life.augmented

RM0492
参考手册

STM32H503 系列基于 Arm® 内核的 32 位 MCU

简介

本参考手册面向应用开发人员，提供有关使用 STM32H503xx 微控制器存储器与外设的完整信息。

有关订购信息以及器件的机械与电气特性，请参见相应的数据手册。

有关 Arm® Cortex®-M33 内核的信息，请参见相应的 Arm® 技术参考手册，可从 http://infocenter.arm.com 获取。

STM32H5 系列 微控制器搭载 ST 最先进的专利技术。

相关文档

- STM32H503xx 数据手册 (DS14053)
- STM32H503CB/EB/KB/RB 器件勘误表 (ES0561)
- STM32 Cortex®-M33 MCU 编程手册 (PM0264)

目录

1 文档约定 63
1.1 概述 63
1.2 寄存器相关缩写词列表 63
1.3 词汇表 64

2 存储器和总线架构 65
2.1 系统架构 65
2.1.1 快速 C 总线 66
2.1.2 S 总线 66
2.1.3 GPDMA1 和 GPDMA2 总线 66
2.1.4 总线矩阵 67
2.1.5 AHB/APB 总线桥 67
2.2 存储器结构 68
2.2.1 简介 68
2.2.2 存储器映射和寄存器边界地址 69
2.2.3 嵌入式 SRAM 74
2.2.4 Flash 概述 74

3 系统安全性 75
3.1 主要安全特性 75
3.2 资源隔离 75
3.2.1 使用安全隐藏保护 (HDP) 实现时域隔离 75
3.2.2 使用 Cortex 特权模式实现资源隔离 76
3.2.3 使用 GTZC 特权控制实现资源隔离 77
3.3 安全执行 79
3.3.1 存储器保护单元 (MPU) 79
3.3.2 嵌入式 Flash 写保护 80
3.3.3 入侵检测和响应 80
3.4 唯一 ID 82
3.5 产品生命周期 82
3.5.1 生命周期管理 84
3.5.2 推荐的产品设置 85
3.6 软件知识产权保护与协作开发 86

3.6.1 软件知识产权保护 87
3.6.2 其他软件知识产权保护 87

4 自举模式 88
4.1 STM32H503 自举模式 88

5 RAM 配置控制器 (RAMCFG) 89
5.1 简介 89
5.2 RAMCFG 主要特性 89
5.3 RAMCFG 功能说明 89
5.3.1 内部 SRAM 特性 89
5.3.2 错误代码校正（SRAM1、SRAM2、BKPSRAM） 90
5.3.3 写保护 (SRAM2) 92
5.3.4 软件擦除 92
5.4 RAMCFG 低功耗模式 92
5.5 RAMCFG 中断 93
5.6 RAMCFG 寄存器 93
5.6.1 RAMCFG 存储器 x 控制寄存器 (RAMCFG_MxCR) 93
5.6.2 RAMCFG 存储器 x 中断使能寄存器 (RAMCFG_MxIER) 94
5.6.3 RAMCFG 存储器中断状态寄存器 (RAMCFG_MxISR) 95
5.6.4 RAMCFG 存储器 x ECC 单错误地址寄存器 (RAMCFG_MxSEAR) 95
5.6.5 RAMCFG 存储器 x ECC 双重错误地址寄存器 (RAMCFG_MxDEAR) . . 96
5.6.6 RAMCFG 存储器 x 中断清零寄存器 x(RAMCFG_MxICR) 96
5.6.7 RAMCFG 存储器 2 写保护寄存器 1 (RAMCFG_M2WPR1) 97
5.6.8 RAMCFG 存储器 x ECC 密钥寄存器 (RAMCFG_MxECCKEYR) 97
5.6.9 RAMCFG 存储器 x 擦除密钥寄存器 (RAMCFG_MxERKEYR) 98
5.6.10 RAMCFG 寄存器映射 98

6 全局特权控制器 (GTZC) 101
6.1 简介 101
6.2 GTZC 主要特性 101
6.3 GTZC 实现 102
6.4 GTZC 功能说明 104
6.4.1 GTZC 框图 104
6.4.2 非法的非特权访问 104
6.4.3 特权控制器 (TZSC) 104

6.4.4 基于块的存储器保护控制器 (MPCBB) 105
6.4.5 上电/复位状态 105
6.5 GTZC1 TZSC 寄存器 106
6.5.1 GTZC1 TZSC 特权配置寄存器 1 (GTZC1_TZSC_PRIVCFGR1) 106
6.5.2 GTZC1 TZSC 特权配置寄存器 2 (GTZC1_TZSC_PRIVCFGR2) 108
6.5.3 GTZC1 TZSC 特权配置寄存器 3 (GTZC1_TZSC_PRIVCFGR3) 109
6.5.4 GTZC1 TZSC BKPSRAM 子区域 A 水印配置寄存器 (GTZC1_TZSC_MPCWM4ACFGR) 110
6.5.5 GTZC1 TZSC BKPSRAM 子区域 A 水印寄存器 (GTZC1_TZSC_MPCWM4AR) 111
6.5.6 GTZC1 TZSC 寄存器映射 112
6.6 GTZC1 MPCBBz 寄存器（z = 1 和 2） 113
6.6.1 GTZC1 SRAMz MPCBB 超级块 x 的特权配置寄存器 (GTZC1_MPCBBz_PRIVCFGRx)（z = 1 和 2） 113
6.6.2 GTZC1 MPCBBz 寄存器映射（z = 1 和 2） 113

7 嵌入式 Flash (FLASH) 114
7.1 简介 114
7.2 FLASH 主要特性 114
7.3 FLASH 功能说明 115
7.3.1 FLASH 框图 115
7.3.2 FLASH 信号 115
7.3.3 Flash 架构和使用 117
7.3.4 FLASH 读操作 119
7.3.5 FLASH 编程操作 121
7.3.6 FLASH 擦除操作 123
7.3.7 FLASH 并行操作 125
7.3.8 Flash 错误保护 125
7.3.9 OTP 和 RO 存储器访问 126
7.3.10 Flash 存储区交换 128
7.3.11 FLASH 复位和时钟 130
7.4 FLASH 选项字节 131
7.4.1 关于选项字节 131
7.4.2 选项字节加载 131
7.4.3 选项字节修改 131
7.4.4 选项字节概述 132
7.4.5 用户和系统选项字节说明 134

7.4.6 数据保护选项字节说明 135
7.4.7 自举地址选项字节说明 135
7.4.8 修改选项字节的特定规则 136
7.5 FLASH 安全性和保护 137
7.5.1 隐藏保护 (HDP) 137
7.5.2 特权 Flash 区域保护 139
7.5.3 Flash 寄存器的特权模式和非特权模式 140
7.5.4 Flash 存储区在存储区交换情况下的属性 141
7.5.5 Flash 配置保护 142
7.5.6 写保护 143
7.5.7 生命周期管理 143
7.5.8 产品状态转换 144
7.5.9 一次性可编程和只读存储器保护 145
7.6 系统存储器 146
7.6.1 系统存储器简介 146
7.6.2 RSS 用户函数 146
7.7 FLASH 低功耗模式 148
7.8 FLASH 错误管理 149
7.8.1 简介 149
7.8.2 写保护错误 (WRPERR) 149
7.8.3 编程顺序错误 (PGSERR) 150
7.8.4 选通错误 (STRBERR) 150
7.8.5 不一致错误 (INCERR) 151
7.8.6 错误校正代码错误 (ECCC/ECCD) 151
7.8.7 选项字节更改错误 (OPTCHANGEERR) 152
7.8.8 其他 HardFault 错误 153
7.9 FLASH 中断 153
7.10 FLASH 寄存器 154
7.10.1 FLASH 访问控制寄存器 (FLASH_ACR) 154
7.10.2 FLASH 密钥寄存器 (FLASH_NSKEYR) 155
7.10.3 FLASH 选项密钥寄存器 (FLASH_OPTKEYR) 156
7.10.4 FLASH 操作状态寄存器 (FLASH_OPSR) 157
7.10.5 FLASH 选项控制寄存器 (FLASH_OPTCR) 157
7.10.6 FLASH 状态寄存器 (FLASH_NSSR) 158
7.10.7 FLASH 控制寄存器 (FLASH_NSCR) 160
7.10.8 FLASH 清零控制寄存器 (FLASH_NSCCR) 162

7.10.9 FLASH 特权配置寄存器 (FLASH_PRIVCFGR) 163
7.10.10 FLASH HDP 扩展寄存器 (FLASH_HDPEXTR) 164
7.10.11 FLASH 选项状态寄存器 (FLASH_OPTSR_CUR) 164
7.10.12 FLASH 选项状态寄存器 (FLASH_OPTSR_PRG) 166
7.10.13 FLASH 选项状态寄存器 2 (FLASH_OPTSR2_CUR) 168
7.10.14 FLASH 选项状态寄存器 2 (FLASH_OPTSR2_PRG) 169
7.10.15 FLASH 唯一自举入口寄存器 (FLASH_NSBOOTR_CUR) 170
7.10.16 FLASH 唯一自举入口地址 (FLASH_NSBOOTR_PRG) 170
7.10.17 FLASH OTP 块锁定 (FLASH_OTPBLR_CUR) 171
7.10.18 FLASH OTP 块锁定 (FLASH_OTPBLR_PRG) 171
7.10.19 FLASH 存储区 1 特权寄存器 (FLASH_PRIVBB1R1) 172
7.10.20 FLASH 存储区 1 扇区写保护 (FLASH_WRP1R_CUR) 172
7.10.21 FLASH 存储区 1 扇区写保护 (FLASH_WRP1R_PRG) 173
7.10.22 FLASH HDP 存储区 1 寄存器 (FLASH_HDP1R_CUR) 173
7.10.23 FLASH HDP 存储区 1 寄存器 (FLASH_HDP1R_PRG) 174
7.10.24 FLASH ECC 校正寄存器 (FLASH_ECCCORR) 174
7.10.25 FLASH ECC 检测寄存器 (FLASH_ECCDETR) 175
7.10.26 FLASH ECC 数据 (FLASH_ECCDR) 176
7.10.27 FLASH 存储区 2 特权寄存器 (FLASH_PRIVBB2R1) 176
7.10.28 FLASH 存储区 2 扇区写保护 (FLASH_WRP2R_CUR) 177
7.10.29 FLASH 存储区 2 扇区写保护 (FLASH_WRP2R_PRG) 177
7.10.30 FLASH HDP 存储区 2 寄存器 (FLASH_HDP2R_CUR) 178
7.10.31 FLASH HDP 存储区 2 寄存器 (FLASH_HDP2R_PRG) 178
7.11 FLASH 寄存器映射 179

8 指令缓存 (ICACHE) 183
8.1 ICACHE 简介 183
8.2 ICACHE 主要特性 183
8.3 ICACHE 实现 184
8.4 ICACHE 功能说明 184
8.4.1 ICACHE 框图 185
8.4.2 ICACHE 复位和时钟 185
8.4.3 ICACHE 标记存储器 186
8.4.4 直接映射 ICACHE（单通路缓存） 187
8.4.5 ICACHE 使能 188
8.4.6 可缓存和不可缓存的通信 188
8.4.7 可缓存访问 189

8.4.8 ICACHE 维护 189
8.4.9 ICACHE 性能监视 190
8.4.10 ICACHE 自举 190
8.5 ICACHE 低功耗模式 190
8.6 ICACHE 错误管理和中断 190
8.7 ICACHE 寄存器 191
8.7.1 ICACHE 控制寄存器 (ICACHE_CR) 191
8.7.2 ICACHE 状态寄存器 (ICACHE_SR) 192
8.7.3 ICACHE 中断使能寄存器 (ICACHE_IER) 192
8.7.4 ICACHE 标志清零寄存器 (ICACHE_FCR) 193
8.7.5 ICACHE 命中监视器寄存器 (ICACHE_HMONR) 194
8.7.6 ICACHE 未命中监视器寄存器 (ICACHE_MMONR) 194
8.7.7 ICACHE 寄存器映射 194

9 电源控制 (PWR) 196
9.1 简介 196
9.2 PWR 主要特性 196
9.3 PWR 引脚和内部信号 197
9.4 PWR 电源和电源域 198
9.4.1 外部电源 198
9.4.2 内部调压器 199
9.4.3 上电和掉电序列 200
9.4.4 独立模拟外设电源 200
9.4.5 USB 收发器电源 201
9.4.6 独立 I/O 电源轨 201
9.4.7 备份域 201
9.5 PWR 系统电源调压 202
9.5.1 LDO 嵌入式调压器 202
9.5.2 V_{CORE} 供电与复位、电压调节和低功耗模式 202
9.5.3 嵌入式调压器工作模式 203
9.6 PWR 电源和温度监视 203
9.6.1 上电复位 (POR)/掉电复位 (PDR) 204
9.6.2 欠压复位 (BOR) 204
9.6.3 可编程电压检测器 (PVD) 205
9.6.4 模拟电压检测器 (AVD) 207
9.6.5 VDDIO2 电压监视器 (IO2VM) 207

9.6.6 备份域电压监视 208
9.6.7 温度监视 208
9.7 PWR 电源管理 209
9.7.1 电压调节 209
9.7.2 电源管理示例 210
9.8 电源模式 211
9.8.1 降低系统时钟速度 214
9.8.2 外设时钟门控 214
9.8.3 低功耗模式 214
9.8.4 睡眠模式 215
9.8.5 停止模式 216
9.8.6 待机模式 218
9.8.7 功耗模式输出引脚 220
9.9 PWR 特权保护 220
9.10 PWR 中断 221
9.11 PWR 寄存器 221
9.11.1 PWR 电源模式控制寄存器 (PWR_PMCR) 221
9.11.2 PWR 状态寄存器 (PWR_PMSR) 223
9.11.3 PWR 电压调节控制寄存器 (PWR_VOSCR) 223
9.11.4 PWR 电压调节状态寄存器 (PWR_VOSSR) 224
9.11.5 PWR 备份域控制寄存器 (PWR_BDCR) 225
9.11.6 PWR 禁止备份保护控制寄存器 (PWR_DBPCR) 226
9.11.7 PWR 备份域状态寄存器 (PWR_BDSR) 226
9.11.8 PWR 电源配置控制寄存器 (PWR_SCCR) 227
9.11.9 PWR 电压监视器控制寄存器 (PWR_VMCR) 228
9.11.10 PWR 电压监视器状态寄存器 (PWR_VMSR) 229
9.11.11 PWR 唤醒状态清零寄存器 (PWR_WUSCR) 230
9.11.12 PWR 唤醒状态寄存器 (PWR_WUSR) 230
9.11.13 PWR 唤醒配置寄存器 (PWR_WUCR) 231
9.11.14 PWR I/O 保留寄存器 (PWR_IORETR) 232
9.11.15 PWR 特权配置寄存器 (PWR_PRIVCFGR) 232
9.11.16 PWR 寄存器映射 233

10 复位和时钟控制 (RCC) 235

10.1 简介 235
10.2 RCC 引脚和内部信号 235

10.3 RCC 复位功能说明 235
10.3.1 电源复位 235
10.3.2 系统复位 236
10.3.3 备份域复位 237
10.3.4 复位源标识 237
10.4 RCC 时钟功能说明 238
10.4.1 HSE 时钟 240
10.4.2 HSI 时钟 241
10.4.3 CSI 振荡器 242
10.4.4 HSI48 时钟 243
10.4.5 PLL 说明 244
10.4.6 LSE 时钟 247
10.4.7 LSI 时钟 248
10.4.8 系统时钟 (SYSCLK) 选择 249
10.4.9 在停止和待机模式下处理时钟发生器 249
10.4.10 时钟安全系统 (CSS) 250
10.4.11 时钟输出生成 (MCO1/MCO2) 251
10.4.12 内核时钟选择 252
10.4.13 基于 TIMx 和 LPTIMx 的时钟测量 254
10.4.14 内部振荡器校准 254
10.4.15 RTC 和 TAMP 时钟 255
10.4.16 定时器时钟 255
10.4.17 看门狗时钟 255
10.4.18 外设时钟门控 255
10.4.19 总线时钟门控 256
10.5 RCC 特权保护模式 256
10.6 RCC 低功耗模式 258
10.7 RCC 中断 258
10.8 RCC 寄存器 259
10.8.1 RCC 时钟控制寄存器 (RCC_CR) 259
10.8.2 RCC HSI 校准寄存器 (RCC_HSICFGR) 262
10.8.3 RCC 时钟恢复 RC 寄存器 (RCC_CRRCR) 262
10.8.4 RCC CSI 校准寄存器 (RCC_CSICFGR) 263
10.8.5 RCC 时钟配置寄存器 1 (RCC_CFGR1) 263
10.8.6 RCC CPU 域时钟配置寄存器 2 (RCC_CFGR2) 266
10.8.7 RCC PLL 时钟源选择寄存器 (RCC_PLL1CFGR) 268

10.8.8 RCC PLL 时钟源选择寄存器 (RCC_PLL2CFGR) 270
10.8.9 RCC PLL1 分频器寄存器 (RCC_PLL1DIVR) 272
10.8.10 RCC PLL1 小数分频器寄存器 (RCC_PLL1FRACR) 273
10.8.11 RCC PLL1 分频器寄存器 (RCC_PLL2DIVR) 274
10.8.12 RCC PLL2 小数分频器寄存器 (RCC_PLL2FRACR) 275
10.8.13 RCC 时钟源中断使能寄存器 (RCC_CIER) 276
10.8.14 RCC 时钟源中断标志寄存器 (RCC_CIFR) 277
10.8.15 RCC 时钟源中断清零寄存器 (RCC_CICR) 278
10.8.16 RCC AHB1 复位寄存器 (RCC_AHB1RSTR) 280
10.8.17 RCC AHB2 外设复位寄存器 (RCC_AHB2RSTR) 281
10.8.18 RCC APB1 外设低位复位寄存器 (RCC_APB1LRSTR) 282
10.8.19 RCC APB1 外设高位复位寄存器 (RCC_APB1HRSTR) 284
10.8.20 RCC APB2 外设复位寄存器 (RCC_APB2RSTR) 285
10.8.21 RCC APB3 外设复位寄存器 (RCC_APB3RSTR) 286
10.8.22 RCC AHB1 外设时钟寄存器 (RCC_AHB1ENR) 287
10.8.23 RCC AHB2 外设时钟寄存器 (RCC_AHB2ENR) 288
10.8.24 RCC APB1 外设时钟寄存器 (RCC_APB1LENR) 290
10.8.25 RCC APB1 外设时钟寄存器 (RCC_APB1HENR) 292
10.8.26 RCC APB2 外设时钟寄存器 (RCC_APB2ENR) 293
10.8.27 RCC APB3 外设时钟寄存器 (RCC_APB3ENR) 294
10.8.28 RCC AHB1 睡眠时钟寄存器 (RCC_AHB1LPENR) 295
10.8.29 RCC AHB2 睡眠时钟寄存器 (RCC_AHB2LPENR) 296
10.8.30 RCC APB1 睡眠时钟寄存器 (RCC_APB1LLPENR) 298
10.8.31 RCC APB1 睡眠时钟寄存器 (RCC_APB1HLPENR) 300
10.8.32 RCC APB2 睡眠时钟寄存器 (RCC_APB2LPENR) 301
10.8.33 RCC APB3 睡眠时钟寄存器 (RCC_APB3LPENR) 302
10.8.34 RCC 内核时钟配置寄存器 (RCC_CCIPR1) 303
10.8.35 RCC 内核时钟配置寄存器 (RCC_CCIPR2) 304
10.8.36 RCC 内核时钟配置寄存器 (RCC_CCIPR3) 305
10.8.37 RCC 内核时钟配置寄存器 (RCC_CCIPR4) 306
10.8.38 RCC 内核时钟配置寄存器 (RCC_CCIPR5) 307
10.8.39 RCC 备份域控制寄存器 (RCC_BDCR) 308
10.8.40 RCC 复位状态寄存器 (RCC_RSR) 310
10.8.41 RCC 特权配置寄存器 (RCC_PRIVCFGR) 312
10.8.42 RCC 寄存器映射 312

11 时钟恢复系统 (CRS) 318
11.1 简介 318
11.2 CRS 主要特性 318
11.3 CRS 实现 318
11.4 CRS 功能说明 319
11.4.1 CRS 框图 319
11.4.2 同步输入 319
11.4.3 频率误差测量 320
11.4.4 频率误差评估和自动微调 321
11.4.5 CRS 初始化和配置 321
11.5 CRS 低功耗模式 322
11.6 CRS 中断 322
11.7 CRS 寄存器 323
11.7.1 CRS 控制寄存器 (CRS_CR) 323
11.7.2 CRS 配置寄存器 (CRS_CFGR) 324
11.7.3 CRS 中断和状态寄存器 (CRS_ISR) 325
11.7.4 CRS 中断标志清零寄存器 (CRS_ICR) 327
11.7.5 CRS 寄存器映射 328

12 通用 I/O (GPIO) 329
12.1 简介 329
12.2 GPIO 主要特性 329
12.3 GPIO 功能说明 329
12.3.1 通用 I/O (GPIO) 331
12.3.2 I/O 引脚复用功能复用器和映射 332
12.3.3 I/O 端口控制寄存器 332
12.3.4 I/O 端口数据寄存器 333
12.3.5 I/O 数据位操作 333
12.3.6 GPIO 锁定机制 333
12.3.7 I/O 复用功能输入/输出 333
12.3.8 外部中断线/唤醒线 334
12.3.9 输入配置 334
12.3.10 输出配置 334
12.3.11 复用功能配置 335
12.3.12 模拟配置 335
12.3.13 将 HSE 或 LSE 振荡器引脚用作 GPIO 336

12.3.14 在 RTC 电源域中使用 GPIO 引脚 336
12.3.15 待机模式期间的 I/O 状态保持 336
12.3.16 特权模式和非特权模式 336
12.3.17 高速低电压模式 (HSLV) 337
12.3.18 I/O 补偿单元 337
12.4 GPIO 寄存器 337
12.4.1 GPIO 端口模式寄存器 (GPIOx_MODER)（x = A 到 D 和 H） 337
12.4.2 GPIO 端口输出类型寄存器 (GPIOx_OTYPER)（x = A 到 D 和 H） . . 338
12.4.3 GPIO 端口输出速度寄存器 (GPIOx_OSPEEDR)（x = A 到 D 和 H） . 338
12.4.4 GPIO 端口上拉/下拉寄存器 (GPIOx_PUPDR)（x = A 到 D 和 H） . . 339
12.4.5 GPIO 端口输入数据寄存器 (GPIOx_IDR)（x = A 到 D 和 H） 339
12.4.6 GPIO 端口输出数据寄存器 (GPIOx_ODR)（x = A 到 D 和 H） 340
12.4.7 GPIO 端口位置 1/复位寄存器 (GPIOx_BSRR)（x = A 到 D 和 H） . . 340
12.4.8 GPIO 端口配置锁定寄存器 (GPIOx_LCKR)（x = A 到 D 和 H） 341
12.4.9 GPIO 复用功能低位寄存器 (GPIOx_AFRL)（x = A 到 D 和 H） 342
12.4.10 GPIO 复用功能高位寄存器 (GPIOx_AFRH)（x = A 到 D 和 H） 342
12.4.11 GPIO 端口位复位寄存器 (GPIOx_BRR)（x = A 到 D 和 H） 343
12.4.12 GPIO 高速低电压寄存器 (GPIOx_HSLVR)（x = A 到 D 和 H） 344
12.4.13 GPIO 寄存器映射 344

13 系统配置、自举和安全性 (SBS) 346
13.1 SBS 简介 346
13.2 SBS 主要特性 346
13.3 SBS 功能说明 347
13.3.1 SBS 框图 347
13.3.2 SBS 信号 347
13.3.3 SBS 复位和时钟 348
13.3.4 SBS 系统配置 348
13.3.5 SBS 自举控制 349
13.3.6 SBS 调试控制 351
13.4 SBS 中断 354
13.5 SBS 寄存器 354
13.5.1 SBS 时域隔离控制寄存器 (SBS_HDPLCR) 354
13.5.2 SBS 时域隔离状态寄存器 (SBS_HDPLSR) 354
13.5.3 SBS 调试控制寄存器 (SBS_DBGCR) 355
13.5.4 SBS 调试锁定寄存器 (SBS_DBGLOCKR) 356

13.5.5 SBS 产品模式和配置寄存器 (SBS_PMCR) 356
13.5.6 SBS FPU 中断屏蔽寄存器 (SBS_FPUIMR) 357
13.5.7 SBS 存储器擦除状态寄存器 (SBS_MESR) 357
13.5.8 SBS I/O 补偿单元控制和状态寄存器 (SBS_CCCSR) 358
13.5.9 SBS I/O 补偿单元值寄存器 (SBS_CCVALR) 359
13.5.10 SBS I/O 补偿单元软件代码寄存器 (SBS_CCSWCR) 360
13.5.11 SBS B 类寄存器 (SBS_CFGR2) 361
13.5.12 SBS CPU 锁定寄存器 (SBS_CNSLCKR) 362
13.5.13 SBS FLITF ECC NMI 屏蔽寄存器 (SBS_ECCNMIR) 362
13.5.14 SBS 寄存器映射 363

14 外设互连矩阵 365
14.1 简介 365
14.2 连接汇总 365
14.3 互连详细信息 366
14.3.1 主从模式的定时器互连 366
14.3.2 ADC 的触发信号 367
14.3.3 ADC 模拟看门狗作为定时器的触发器 367
14.3.4 DAC 的触发器 368
14.3.5 DTS 的触发器 368
14.3.6 两个 LP 定时器之间的连接 369
14.3.7 定时器的内部时钟源 369
14.3.8 低功耗定时器的触发器 370
14.3.9 RTC 唤醒信号作为定时器的输入 370
14.3.10 比较器的消隐源 371
14.3.11 比较器作为定时器的输入、触发或刹车信号 371
14.3.12 系统错误作为定时器的刹车信号 372
14.3.13 GPDMA1/2 的触发信号 372
14.3.14 模拟外设的内部模拟信号 373
14.3.15 DAC 采样和保持模式下的时钟源 373
14.3.16 内部入侵源 374
14.3.17 对 RTC 的入侵输出 374

15 通用直接存储器访问控制器 (GPDMA) 375
15.1 GPDMA 简介 375
15.2 GPDMA 主要特性 375
15.3 GPDMA 实现 376

15.3.1 GPDMA 实例 376
15.3.2 GPDMA 通道 376
15.3.3 低功耗模式下的 GPDMA 自主模式 377
15.3.4 GPDMA 请求 377
15.3.5 GPDMA 块请求 381
15.3.6 支持外设提前终止的 GPDMA 通道 382
15.3.7 GPDMA 触发信号 382
15.4 GPDMA 功能说明 384
15.4.1 GPDMA 框图 384
15.4.2 GPDMA 通道状态和直接编程（无任何链表） 384
15.4.3 GPDMA 通道挂起和恢复 386
15.4.4 GPDMA 通道中止和重启 387
15.4.5 GPDMA 链表数据结构 388
15.4.6 链表项传输执行 391
15.4.7 运行至完成模式下的 GPDMA 通道状态和链表编程 391
15.4.8 链接步进模式下的 GPDMA 通道状态和链表编程 395
15.4.9 GPDMA 通道状态和链表编程 402
15.4.10 基于 FIFO 的 GPDMA 传输 404
15.4.11 GPDMA 传输请求和仲裁 410
15.4.12 GPDMA 触发传输 414
15.4.13 GPDMA 循环缓冲与链表编程 416
15.4.14 外设流控制模式下的 GPDMA 传输 417
15.4.15 GPDMA 特权/非特权通道 418
15.4.16 GPDMA 错误管理 418
15.4.17 GPDMA 自主模式 420
15.5 调试模式下的 GPDMA 420
15.6 低功耗模式下的 GPDMA 421
15.7 GPDMA 中断 421
15.8 GPDMA 寄存器 422
15.8.1 GPDMA 特权配置寄存器 (GPDMA_PRIVCFGR) 422
15.8.2 GPDMA 屏蔽中断状态寄存器 (GPDMA_MISR) 423
15.8.3 GPDMA 通道 x 链表基址寄存器 (GPDMA_CxLBAR) 424
15.8.4 GPDMA 通道 x 标志清零寄存器 (GPDMA_CxFCR) 424
15.8.5 GPDMA 通道 x 状态寄存器 (GPDMA_CxSR) 425
15.8.6 GPDMA 通道 x 控制寄存器 (GPDMA_CxCR) 427
15.8.7 GPDMA 通道 x 传输寄存器 1 (GPDMA_CxTR1) 429

15.8.8 GPDMA 通道 x 传输寄存器 2 (GPDMA_CxTR2) 432
15.8.9 GPDMA 通道 x 块寄存器 1 (GPDMA_CxBR1) 435
15.8.10 GPDMA 通道 x 复用块寄存器 1 (GPDMA_CxBR1) 436
15.8.11 GPDMA 通道 x 源地址寄存器 (GPDMA_CxSAR) 438
15.8.12 GPDMA 通道 x 目标地址寄存器 (GPDMA_CxDAR) 439
15.8.13 GPDMA 通道 x 传输寄存器 3 (GPDMA_CxTR3) 440
15.8.14 GPDMA 通道 x 块寄存器 2 (GPDMA_CxBR2) 441
15.8.15 GPDMA 通道 x 链表地址寄存器 (GPDMA_CxLLR) 442
15.8.16 GPDMA 通道 x 复用链表地址寄存器 (GPDMA_CxLLR) 443
15.8.17 GPDMA 寄存器映射 445

16 嵌套向量中断控制器 (NVIC) 447
16.1 NVIC 主要特性 447
16.2 SysTick 校准值寄存器 447
16.3 中断和异常向量 447

17 扩展中断和事件控制器 (EXTI) 452
17.1 EXTI 主要特性 452
17.2 EXTI 框图 452
17.2.1 外设和 CPU 之间的 EXTI 连接 454
17.2.2 EXTI 中断/事件映射 454
17.3 EXTI 功能说明 455
17.3.1 EXTI 可配置事件输入唤醒 456
17.3.2 EXTI 复用器选择 456
17.4 EXTI 功能行为 457
17.5 EXTI 事件保护 458
17.5.1 EXTI 特权保护 458
17.6 EXTI 寄存器 458
17.6.1 EXTI 上升沿触发选择寄存器 (EXTI_RTSR1) 459
17.6.2 EXTI 下降沿触发选择寄存器 (EXTI_FTSR1) 459
17.6.3 EXTI 软件中断事件寄存器 (EXTI_SWIER1) 460
17.6.4 EXTI 上升沿挂起寄存器 (EXTI_RPR1) 460
17.6.5 EXTI 下降沿挂起寄存器 (EXTI_FPR1) 461
17.6.6 EXTI 特权配置寄存器 (EXTI_PRIVCFGR1) 462
17.6.7 EXTI 上升沿触发选择寄存器 2 (EXTI_RTSR2) 463
17.6.8 EXTI 下降沿触发选择寄存器 2 (EXTI_FTSR2) 463

17.6.9 EXTI 软件中断事件寄存器 2 (EXTI_SWIER2) 464
17.6.10 EXTI 上升沿挂起寄存器 2 (EXTI_RPR2) 465
17.6.11 EXTI 下降沿挂起寄存器 2 (EXTI_FPR2) 466
17.6.12 EXTI 特权配置寄存器 2 (EXTI_PRIVCFGR2) 466
17.6.13 EXTI 外部中断选择寄存器 (EXTI_EXTICR1) 467
17.6.14 EXTI 外部中断选择寄存器 (EXTI_EXTICR2) 468
17.6.15 EXTI 外部中断选择寄存器 (EXTI_EXTICR3) 469
17.6.16 EXTI 外部中断选择寄存器 (EXTI_EXTICR4) 470
17.6.17 EXTI CPU 唤醒中断屏蔽寄存器 (EXTI_IMR1) 471
17.6.18 EXTI CPU 唤醒事件屏蔽寄存器 (EXTI_EMR1) 472
17.6.19 EXTI CPU 唤醒中断屏蔽寄存器 2 (EXTI_IMR2) 473
17.6.20 EXTI CPU 唤醒事件屏蔽寄存器 2 (EXTI_EMR2) 474
17.6.21 EXTI 寄存器映射 475

18 循环冗余校验计算单元 (CRC) 478

18.1 简介 478
18.2 CRC 主要特性 478
18.3 CRC 功能说明 479
18.3.1 CRC 框图 479
18.3.2 CRC 内部信号 479
18.3.3 CRC 运算 479
18.4 CRC 寄存器 481
18.4.1 CRC 数据寄存器 (CRC_DR) 481
18.4.2 CRC 独立数据寄存器 (CRC_IDR) 481
18.4.3 CRC 控制寄存器 (CRC_CR) 482
18.4.4 CRC 初始值 (CRC_INIT) 483
18.4.5 CRC 多项式 (CRC_POL) 483
18.4.6 CRC 寄存器映射 484

19 模数转换器 (ADC1) 485

19.1 简介 485
19.2 ADC 主要特性 485
19.3 ADC 实现 487
19.4 ADC 功能说明 488
19.4.1 ADC 框图 488
19.4.2 ADC 引脚和内部信号 489

19.4.3 ADC 时钟 491
19.4.4 ADC 连接 493
19.4.5 从设备 AHB 接口 494
19.4.6 ADC 深度掉电模式 (DEEPPWD) 和 ADC 调压器 (ADVREGEN) 494
19.4.7 单端输入通道和差分输入通道 494
19.4.8 校准（ADCAL、ADCALDIF、ADC_CALFACT） 495
19.4.9 ADC 开关控制（ADEN、ADDIS、ADRDY） 497
19.4.10 写入 ADC 控制位时的限制 498
19.4.11 通道选择（SQRx、JSQRx） 499
19.4.12 可独立设置各通道采样时间（SMPR1、SMPR2） 500
19.4.13 单次转换模式 (CONT = 0) 501
19.4.14 连续转换模式 (CONT = 1) 502
19.4.15 开始转换（ADSTART、JADSTART） 503
19.4.16 ADC 时序 503
19.4.17 停止正在进行的转换（ADSTP、JADSTP） 504
19.4.18 外部触发转换和触发极性（EXTSEL、EXTEN、JEXTSEL、JEXTEN） 506
19.4.19 注入通道管理 507
19.4.20 不连续模式（DISCEN、DISCNUM、JDISCEN） 508
19.4.21 注入转换的上下文队列 509
19.4.22 可编程分辨率 (RES)——快速转换模式 517
19.4.23 转换结束、采样阶段结束（EOC、JEOC、EOSMP） 517
19.4.24 转换序列结束（EOS、JEOS） 517
19.4.25 时序图示例（单次模式/连续模式，硬件/软件触发） 518
19.4.26 数据管理 520
19.4.27 动态低功耗特性 525
19.4.28 模拟窗口看门狗（AWD1EN、JAWD1EN、AWD1SGL、AWD1CH、AWD2CH、AWD3CH、AWD_HTx、AWD_LTx、AWDx） 530
19.4.29 过采样器 534
19.4.30 温度传感器 538
19.4.31 VBAT 电源监测 540
19.4.32 监测内部参考电压 540
19.4.33 监视电源电压 542
19.5 ADC 中断 542
19.6 ADC 寄存器 543
19.6.1 ADC 中断和状态寄存器 (ADC_ISR) 543
19.6.2 ADC 中断使能寄存器 (ADC_IER) 544

19.6.3 ADC 控制寄存器 (ADC_CR) 546
19.6.4 ADC 配置寄存器 (ADC_CFGR) 549
19.6.5 ADC 配置寄存器 2 (ADC_CFGR2) 553
19.6.6 ADC 采样时间寄存器 1 (ADC_SMPR1) 555
19.6.7 ADC 采样时间寄存器 2 (ADC_SMPR2) 555
19.6.8 ADC 看门狗阈值寄存器 1 (ADC_TR1) 556
19.6.9 ADC 看门狗阈值寄存器 2 (ADC_TR2) 557
19.6.10 ADC 看门狗阈值寄存器 3 (ADC_TR3) 558
19.6.11 ADC 常规序列寄存器 1 (ADC_SQR1) 558
19.6.12 ADC 常规序列寄存器 2 (ADC_SQR2) 559
19.6.13 ADC 常规序列寄存器 3 (ADC_SQR3) 560
19.6.14 ADC 常规序列寄存器 4 (ADC_SQR4) 561
19.6.15 ADC 常规数据寄存器 (ADC_DR) 562
19.6.16 ADC 注入序列寄存器 (ADC_JSQR) 562
19.6.17 ADC 偏移 y 寄存器 (ADC_OFRy) 564
19.6.18 ADC 注入通道 y 数据寄存器 (ADC_JDRy) 565
19.6.19 ADC 模拟看门狗 2 配置寄存器 (ADC_AWD2CR) 565
19.6.20 ADC 模拟看门狗 3 配置寄存器 (ADC_AWD3CR) 566
19.6.21 ADC 差分模式选择寄存器 (ADC_DIFSEL) 566
19.6.22 ADC 校准系数 (ADC_CALFACT) 567
19.6.23 ADC 选项寄存器 (ADC_OR) 567
19.7 ADC 通用寄存器 568
19.7.1 ADC 通用控制寄存器 (ADC_CCR) 568
19.7.2 ADC 硬件配置寄存器 (ADC_HWCFGR0) 569
19.7.3 ADC 版本触发器 (ADC_VERR) 570
19.7.4 ADC 标识寄存器 (ADC_IPDR) 571
19.7.5 ADC 大小标识寄存器 (ADC_SIDR) 571
19.8 ADC 寄存器映射 572

20 数字温度传感器 (DTS) 575
20.1 简介 575
20.2 DTS 主要特性 575
20.3 DTS 功能说明 576
20.3.1 DTS 框图 576
20.3.2 DTS 内部信号 576
20.3.3 DTS 模块操作 577
20.3.4 工作模式 577

20.3.5 校准 577
20.3.6 预分频器 577
20.3.7 温度测量原理 578
20.3.8 采样时间 579
20.3.9 快速测量模式 579
20.3.10 触发输入 580
20.3.11 开关控制和就绪标志 580
20.3.12 温度测量序列 581
20.4 DTS 低功耗模式 581
20.5 DTS 中断 582
20.5.1 温度窗口比较器 582
20.5.2 同步中断 582
20.5.3 异步唤醒 582
20.6 DTS 寄存器 583
20.6.1 温度传感器配置寄存器 1 (DTS_CFGR1) 583
20.6.2 温度传感器 T0 值寄存器 1 (DTS_T0VALR1) 584
20.6.3 温度传感器斜坡值寄存器 (DTS_RAMPVALR) 585
20.6.4 温度传感器中断阈值寄存器 1 (DTS_ITR1) 585
20.6.5 温度传感器数据寄存器 (DTS_DR) 586
20.6.6 温度传感器状态寄存器 (DTS_SR) 586
20.6.7 温度传感器中断使能寄存器 (DTS_ITENR) 588
20.6.8 温度传感器清除中断标志寄存器 (DTS_ICIFR) 589
20.6.9 温度传感器选项寄存器 (DTS_OR) 590
20.6.10 DTS 寄存器映射 591

21 数模转换器 (DAC) 592
21.1 简介 592
21.2 DAC 主要特性 592
21.3 DAC 实现 592
21.4 DAC 功能说明 593
21.4.1 DAC 框图 593
21.4.2 DAC 引脚和内部信号 594
21.4.3 DAC 时钟 595
21.4.4 DAC 通道使能 595
21.4.5 DAC 数据格式 595
21.4.6 DAC 转换 597

21.4.7 DAC 输出电压 598
21.4.8 DAC 触发选择 598
21.4.9 DMA 请求 598
21.4.10 生成噪声 599
21.4.11 生成三角波 600
21.4.12 DAC 通道模式 601
21.4.13 DAC 通道缓冲器校准 604
21.4.14 DAC 双通道转换模式（如果有两个通道可用） 605
21.5 低功耗模式下的 DAC 609
21.6 DAC 中断 609
21.7 DAC 寄存器 610
21.7.1 DAC 控制寄存器 (DAC_CR) 610
21.7.2 DAC 软件触发寄存器 (DAC_SWTRGR) 613
21.7.3 DAC 通道 1 12 位右对齐数据保持寄存器 (DAC_DHR12R1) 614
21.7.4 DAC 通道 1 12 位左对齐数据保持寄存器 (DAC_DHR12L1) 614
21.7.5 DAC 通道 1 8 位右对齐数据保持寄存器 (DAC_DHR8R1) 615
21.7.6 DAC 通道 2 12 位右对齐数据保持寄存器 (DAC_DHR12R2) 615
21.7.7 DAC 通道 2 12 位左对齐数据保持寄存器 (DAC_DHR12L2) 616
21.7.8 DAC 通道 2 8 位右对齐数据保持寄存器 (DAC_DHR8R2) 616
21.7.9 双 DAC 12 位右对齐数据保持寄存器 (DAC_DHR12RD) 617
21.7.10 双 DAC 12 位左对齐数据保持寄存器 (DAC_DHR12LD) 617
21.7.11 双 DAC 8 位右对齐数据保持寄存器 (DAC_DHR8RD) 618
21.7.12 DAC 通道 1 数据输出寄存器 (DAC_DOR1) 618
21.7.13 DAC 通道 2 数据输出寄存器 (DAC_DOR2) 619
21.7.14 DAC 状态寄存器 (DAC_SR) 619
21.7.15 DAC 校准控制寄存器 (DAC_CCR) 621
21.7.16 DAC 模式控制寄存器 (DAC_MCR) 621
21.7.17 DAC 通道 1 采样和保持采样时间寄存器 (DAC_SHSR1) 623
21.7.18 DAC 通道 2 采样和保持采样时间寄存器 (DAC_SHSR2) 623
21.7.19 DAC 采样和保持时间寄存器 (DAC_SHHR) 624
21.7.20 DAC 采样和保持刷新时间寄存器 (DAC_SHRR) 625
21.7.21 DAC 寄存器映射 626

22 比较器 (COMP) 628
22.1 简介 628
22.2 COMP 主要特性 628

22.3 COMP 功能描述 629
22.3.1 COMP 框图 629
22.3.2 COMP 引脚和内部信号 629
22.3.3 COMP 复位和时钟 631
22.3.4 比较器锁定机制 631
22.3.5 迟滞 631
22.3.6 比较器输出消隐功能 632
22.3.7 COMP 启动稳定时间 632
22.3.8 COMP 功耗和速度模式 632
22.4 COMP 低功耗模式 633
22.5 COMP 中断 633
22.5.1 通过 EXTI 实现的中断 633
22.5.2 通过 CPU 的 NVIC 实现的中断 633
22.6 SCALER 功能 634
22.7 COMP 寄存器 635
22.7.1 比较器状态寄存器 (COMP_SR) 635
22.7.2 比较器中断清零标志寄存器 (COMP_ICFR) 635
22.7.3 比较器配置寄存器 1 (COMP_CFGR1) 636
22.7.4 比较器配置寄存器 2 (COMP_CFGR2) 638
22.7.5 COMP 寄存器映射 639

23 运算放大器 (OPAMP) 640
23.1 简介 640
23.2 OPAMP 主要特性 640
23.3 OPAMP 功能描述 640
23.3.1 OPAMP 复位和时钟 640
23.3.2 初始配置 641
23.3.3 信号走线 641
23.3.4 OPAMP 模式 641
23.3.5 校准 648
23.4 OPAMP 低功耗模式 649
23.5 OPAMP PGA 增益 649
23.6 OPAMP 寄存器 650
23.6.1 OPAMP1 控制/状态寄存器 (OPAMP1_CSR) 650
23.6.2 正常模式下的 OPAMP1 微调寄存器 (OPAMP1_OTR) 652
23.6.3 高速模式下的 OPAMP1 微调寄存器 (OPAMP1_HSOTR) 652

23.6.4 OPAMP 寄存器映射 653

24 真随机数发生器 (RNG) 654

24.1 简介 654

24.2 RNG 主要特性 654

24.3 RNG 功能描述 655

24.3.1 RNG 框图 655

24.3.2 RNG 内部信号 655

24.3.3 随机数生成 656

24.3.4 RNG 初始化 658

24.3.5 RNG 操作 659

24.3.6 RNG 时钟 660

24.3.7 错误管理 660

24.3.8 RNG 低功耗使用 661

24.4 RNG 中断 662

24.5 RNG 处理时间 662

24.6 RNG 熵源验证 662

24.6.1 简介 662

24.6.2 验证条件 663

24.6.3 数据采集 663

24.7 RNG 寄存器 664

24.7.1 RNG 控制寄存器 (RNG_CR) 664

24.7.2 RNG 状态寄存器 (RNG_SR) 666

24.7.3 RNG 数据寄存器 (RNG_DR) 667

24.7.4 RNG 运行状态测试控制寄存器 (RNG_HTCR) 667

24.7.5 RNG 寄存器映射 668

25 哈希处理器 (HASH) 669

25.1 简介 669

25.2 HASH 主要特性 669

25.3 HASH 实现 670

25.4 HASH 功能描述 670

25.4.1 HASH 框图 670

25.4.2 HASH 内部信号 670

25.4.3 关于安全哈希算法 671

25.4.4 消息数据输送 671

25.4.5 消息摘要计算 672
25.4.6 消息填充 674
25.4.7 HMAC 运算 675
25.4.8 HASH 挂起/恢复操作 677
25.4.9 HASH DMA 接口 679
25.4.10 HASH 错误管理 679
25.5 HASH 中断 679
25.6 HASH 处理时间 680
25.7 HASH 寄存器 680
25.7.1 HASH 控制寄存器 (HASH_CR) 680
25.7.2 HASH 数据输入寄存器 (HASH_DIN) 682
25.7.3 HASH 启动寄存器 (HASH_STR) 683
25.7.4 HASH 摘要寄存器 683
25.7.5 HASH 中断使能寄存器 (HASH_IMR) 685
25.7.6 HASH 状态寄存器 (HASH_SR) 686
25.7.7 HASH 上下文交换寄存器 687
25.7.8 HASH 寄存器映射 688

26 高级控制定时器 (TIM1) 689
26.1 TIM1 简介 689
26.2 TIM1 主要特性 689
26.3 TIM1 功能描述 690
26.3.1 框图 690
26.3.2 TIM1 引脚和内部信号 691
26.3.3 时基单元 695
26.3.4 计数器模式 697
26.3.5 重复计数器 707
26.3.6 外部触发输入 708
26.3.7 时钟选择 709
26.3.8 捕获/比较通道 712
26.3.9 输入捕获模式 715
26.3.10 PWM 输入模式 716
26.3.11 强制输出模式 717
26.3.12 输出比较模式 717
26.3.13 PWM 模式 718
26.3.14 不对称 PWM 模式 726

26.3.15 组合 PWM 模式 727
26.3.16 组合三相 PWM 模式 728
26.3.17 互补输出和死区插入 729
26.3.18 使用刹车功能 732
26.3.19 双向刹车输入 737
26.3.20 发生外部事件时清除 tim_ocxref 信号 738
26.3.21 生成 6 步 PWM 740
26.3.22 单脉冲模式 741
26.3.23 可再触发单脉冲模式 742
26.3.24 基于比较的脉冲模式 743
26.3.25 编码器接口模式 745
26.3.26 方向位输出 762
26.3.27 UIF 位重映射 763
26.3.28 定时器输入异或功能 763
26.3.29 连接霍尔传感器 763
26.3.30 定时器同步 765
26.3.31 ADC 触发 769
26.3.32 DMA 分组传送模式 769
26.3.33 TIM1DMA 请求 770
26.3.34 调试模式 771
26.4 TIM1低功耗模式 771
26.5 TIM1 中断 771
26.6 TIM1 寄存器 772
26.6.1 TIM1 控制寄存器 1 (TIM1_CR1) 772
26.6.2 TIM1 控制寄存器 2 (TIM1_CR2) 773
26.6.3 TIM1 从模式控制寄存器 (TIM1_SMCR) 776
26.6.4 TIM1 DMA/中断使能寄存器 (TIM1_DIER) 779
26.6.5 TIM1 状态寄存器 (TIM1_SR) 781
26.6.6 TIM1 事件生成寄存器 (TIM1_EGR) 783
26.6.7 TIM1 捕获/比较模式寄存器 1 (TIM1_CCMR1) 785
26.6.8 TIM1 捕获/比较模式寄存器 1 [复用] (TIM1_CCMR1) 786
26.6.9 TIM1 捕获/比较模式寄存器 2 (TIM1_CCMR2) 789
26.6.10 TIM1 捕获/比较模式寄存器 2 [复用] (TIM1_CCMR2) 790
26.6.11 TIM1 捕获/比较使能寄存器 (TIM1_CCER) 793
26.6.12 TIM1 计数器 (TIM1_CNT) 796
26.6.13 TIM1 预分频器 (TIM1_PSC) 797
26.6.14 TIM1 自动重载寄存器 (TIM1_ARR) 797

26.6.15 TIM1 重复计数器寄存器 (TIM1_RCR) 797
26.6.16 TIM1 捕获/比较寄存器 1 (TIM1_CCR1) 798
26.6.17 TIM1 捕获/比较寄存器 2 (TIM1_CCR2) 799
26.6.18 TIM1 捕获/比较寄存器 3 (TIM1_CCR3) 799
26.6.19 TIM1 捕获/比较寄存器 4 (TIM1_CCR4) 800
26.6.20 TIM1 刹车和死区寄存器 (TIM1_BDTR) 801
26.6.21 TIM1 捕获/比较寄存器 5 (TIM1_CCR5) 804
26.6.22 TIM1 捕获/比较寄存器 6 (TIM1_CCR6) 805
26.6.23 TIM1 捕获/比较模式寄存器 3 (TIM1_CCMR3) 806
26.6.24 TIM1 定时器死区寄存器 2 (TIM1_DTR2) 806
26.6.25 TIM1 定时器编码器控制寄存器 (TIM1_ECR) 807
26.6.26 TIM1 定时器输入选择寄存器 (TIM1_TISEL) 808
26.6.27 TIM1 复用功能选项寄存器 1 (TIM1_AF1) 809
26.6.28 TIM1 复用功能寄存器 2 (TIM1_AF2) 812
26.6.29 TIM1 DMA 控制寄存器 (TIM1_DCR) 814
26.6.30 TIM1 全传输 DMA 地址 (TIM1_DMAR) 815
26.6.31 TIM1 寄存器映射 816

27 通用定时器 (TIM2/TIM3) 819

27.1 TIM2/TIM3 简介 819
27.2 TIM2/TIM3 主要特性 819
27.3 TIM2/TIM3 实现 819
27.4 TIM2/TIM3 功能描述 820
27.4.1 框图 820
27.4.2 TIM2/TIM3 引脚和内部信号 821
27.4.3 时基单元 824
27.4.4 计数器模式 826
27.4.5 时钟选择 836
27.4.6 捕获/比较通道 839
27.4.7 输入捕获模式 841
27.4.8 PWM 输入模式 842
27.4.9 强制输出模式 843
27.4.10 输出比较模式 843
27.4.11 PWM 模式 845
27.4.12 非对称 PWM 模式 853
27.4.13 组合 PWM 模式 854
27.4.14 发生外部事件时清除 tim_ocxref 信号 855

27.4.15 单脉冲模式 857
27.4.16 可再触发单脉冲模式 858
27.4.17 基于比较的脉冲模式 859
27.4.18 编码器接口模式 861
27.4.19 方向位输出 878
27.4.20 UIF 位重映射 879
27.4.21 定时器输入异或功能 879
27.4.22 定时器与外部触发同步 879
27.4.23 定时器同步 883
27.4.24 ADC 触发 887
27.4.25 DMA 突发传输模式 888
27.4.26 TIM2/TIM3DMA 请求 889
27.4.27 调试模式 889
27.4.28 TIM2/TIM3 低功耗模式 889
27.4.29 TIM2/TIM3 中断 890
27.5 TIM2/TIM3 寄存器 891
27.5.1 TIMx 控制寄存器 1 (TIMx_CR1)（x = 2、3） 891
27.5.2 TIMx 控制寄存器 2 (TIMx_CR2)（x = 2、3） 892
27.5.3 TIMx 从模式控制寄存器 (TIMx_SMCR)（x = 2、3） 894
27.5.4 TIMx DMA/中断使能寄存器 (TIMx_DIER)（x = 2、3） 896
27.5.5 TIMx 状态寄存器 (TIMx_SR)（x = 2、3） 898
27.5.6 TIMx 事件生成寄存器 (TIMx_EGR)（x = 2、3） 900
27.5.7 TIMx 捕获/比较模式寄存器 1 (TIMx_CCMR1)（x = 2、3） 900
27.5.8 TIMx 捕获/比较模式寄存器 1 [复用] (TIMx_CCMR1)（x = 2、3） . . . 902
27.5.9 TIMx 捕获/比较模式寄存器 2 (TIMx_CCMR2)（x = 2、3） 904
27.5.10 TIMx 捕获/比较模式寄存器 2 [复用] (TIMx_CCMR2)（x = 2、3） . . . 905
27.5.11 TIMx 捕获/比较使能寄存器 (TIMx_CCER)（x = 2、3） 906
27.5.12 TIM3 计数器 (TIM3_CNT) 908
27.5.13 TIM2 计数器 (TIM2_CNT) 908
27.5.14 TIMx 预分频器 (TIMx_PSC)（x = 2、3） 909
27.5.15 TIM3 自动重载寄存器 (TIM3_ARR) 909
27.5.16 TIM2 自动重载寄存器 (TIM2_ARR) 910
27.5.17 TIM3 捕获/比较寄存器 1 (TIM3_CCR1) 910
27.5.18 TIM2 捕获/比较寄存器 1 (TIM2_CCR1) 911
27.5.19 TIM3 捕获/比较寄存器 2 (TIM3_CCR2) 912
27.5.20 TIM2 捕获/比较寄存器 2 (TIM2_CCR2) 912
27.5.21 TIM3 捕获/比较寄存器 3 (TIM3_CCR3) 913

27.5.22 TIM2 捕获/比较寄存器 3 (TIM2_CCR3) 914
27.5.23 TIM3 捕获/比较寄存器 4 (TIM3_CCR4) 914
27.5.24 TIM2 捕获/比较寄存器 4 (TIM2_CCR4) 915
27.5.25 TIMx 定时器编码器控制寄存器 (TIMx_ECR)（x = 2、3） 916
27.5.26 TIMx 定时器输入选择寄存器 (TIMx_TISEL)（x = 2、3） 917
27.5.27 TIMx 复用功能寄存器 1 (TIMx_AF1)（x = 2、3） 918
27.5.28 TIMx 复用功能寄存器 2 (TIMx_AF2)（x = 2、3） 918
27.5.29 TIMx DMA 控制寄存器 (TIMx_DCR)（x = 2、3） 919
27.5.30 TIMx 全传输 DMA 地址 (TIMx_DMAR)（x = 2、3） 920
27.5.31 TIMx 寄存器映射 921

28 基本定时器 (TIM6/TIM7) 924

28.1 TIM6/TIM7 简介 924
28.2 TIM6/TIM7 主要特性 924
28.3 TIM6/TIM7 功能描述 925
28.3.1 TIM6/TIM7 框图 925
28.3.2 TIM6/TIM7 内部信号 925
28.3.3 TIM6/TIM7 时钟 926
28.3.4 时基单元 926
28.3.5 计数模式 928
28.3.6 UIF 位重映射 935
28.3.7 ADC 触发 936
28.3.8 TIM6/TIM7 DMA 请求 936
28.3.9 调试模式 936
28.3.10 TIM6/TIM7 低功耗模式 936
28.3.11 TIM6/TIM7 中断 936
28.4 TIM6/TIM7 寄存器 937
28.4.1 TIMx 控制寄存器 1 (TIMx_CR1)（x = 6 到 7） 937
28.4.2 TIMx 控制寄存器 2 (TIMx_CR2)（x = 6 到 7） 938
28.4.3 TIMx DMA/中断使能寄存器 (TIMx_DIER)（x = 6 到 7） 939
28.4.4 TIMx 状态寄存器 (TIMx_SR)（x = 6 到 7） 939
28.4.5 TIMx 事件生成寄存器 (TIMx_EGR)（x = 6 到 7） 940
28.4.6 TIMx 计数器 (TIMx_CNT)（x = 6 到 7） 940
28.4.7 TIMx 预分频器 (TIMx_PSC)（x = 6 到 7） 941
28.4.8 TIMx 自动重载寄存器 (TIMx_ARR)（x = 6 到 7） 941
28.4.9 TIMx 寄存器映射 942

29 低功耗定时器 (LPTIM) 943
29.1 简介 943
29.2 LPTIM 主要特性 943
29.3 LPTIM 实现 944
29.4 LPTIM 功能描述 945
29.4.1 LPTIM 框图 945
29.4.2 LPTIM 引脚和内部信号 946
29.4.3 LPTIM 输入和触发映射 947
29.4.4 LPTIM 复位和时钟 948
29.4.5 干扰滤波器 948
29.4.6 预分频器 949
29.4.7 触发多路复用器 949
29.4.8 工作模式 950
29.4.9 超时功能 952
29.4.10 生成波形 952
29.4.11 寄存器更新 953
29.4.12 计数器模式 954
29.4.13 定时器使能 954
29.4.14 定时器计数器复位 954
29.4.15 编码器模式 955
29.4.16 重复计数器 956
29.4.17 捕获/比较通道 957
29.4.18 输入捕获模式 958
29.4.19 PWM 模式 960
29.4.20 DMA 请求 962
29.4.21 调试模式 962
29.5 LPTIM 低功耗模式 962
29.6 LPTIM 中断 963
29.7 LPTIM 寄存器 963
29.7.1 LPTIMx 中断和状态寄存器 [复用] (LPTIMx_ISR)（x = 1、2） 964
29.7.2 LPTIMx 中断和状态寄存器 [复用] (LPTIMx_ISR)（x = 1、2） 966
29.7.3 LPTIMx 中断清零寄存器 [复用] (LPTIMx_ICR)（x = 1、2） 968
29.7.4 LPTIMx 中断清零寄存器 [复用] (LPTIMx_ICR)（x = 1、2） 969
29.7.5 LPTIMx 中断使能寄存器 [复用] (LPTIMx_DIER)（x = 1、2） 970
29.7.6 LPTIMx 中断使能寄存器 [复用] (LPTIMx_DIER)（x = 1、2） 972
29.7.7 LPTIM 配置寄存器 (LPTIM_CFGR) 973

29.7.8 LPTIM 控制寄存器 (LPTIM_CR) 976
29.7.9 LPTIM 比较寄存器 1 (LPTIM_CCR1) 977
29.7.10 LPTIM 自动重载寄存器 (LPTIM_ARR) 977
29.7.11 LPTIM 计数器寄存器 (LPTIM_CNT) 978
29.7.12 LPTIM 配置寄存器 2 (LPTIM_CFGR2) 978
29.7.13 LPTIM 重复寄存器 (LPTIM_RCR) 979
29.7.14 LPTIM 捕获/比较模式寄存器 1 (LPTIM_CCMR1) 980
29.7.15 LPTIM 比较器寄存器 2 (LPTIM_CCR2) 982
29.7.16 LPTIM 寄存器映射 983

30 独立看门狗 (IWDG) 985

30.1 简介 985
30.2 IWDG 主要特性 985
30.3 IWDG 实现 985
30.4 IWDG 功能描述 986
30.4.1 IWDG 框图 986
30.4.2 IWDG 内部信号 986
30.4.3 软件和硬件看门狗模式 987
30.4.4 窗口选项 988
30.4.5 调试 990
30.4.6 寄存器访问保护 990
30.5 IWDG 低功耗模式 990
30.6 IWDG 中断 990
30.7 IWDG 寄存器 992
30.7.1 IWDG 密钥寄存器 (IWDG_KR) 993
30.7.2 IWDG 预分频器寄存器 (IWDG_PR) 993
30.7.3 IWDG 重载寄存器 (IWDG_RLR) 994
30.7.4 IWDG 状态寄存器 (IWDG_SR) 995
30.7.5 IWDG 窗口寄存器 (IWDG_WINR) 996
30.7.6 IWDG 提前唤醒中断寄存器 (IWDG_EWCR) 996
30.7.7 IWDG 寄存器映射 997

31 系统窗口看门狗 (WWDG) 998

31.1 简介 998
31.2 WWDG 主要特性 998
31.3 WWDG 实现 998

31.4 WWDG 功能描述 999
31.4.1 WWDG 框图 999
31.4.2 WWDG 内部信号 999
31.4.3 使能看门狗 1000
31.4.4 控制递减计数器 1000
31.4.5 如何设置看门狗超时 1000
31.4.6 调试模式 1001
31.5 WWDG 中断 1002
31.6 WWDG 寄存器 1002
31.6.1 WWDG 控制寄存器 (WWDG_CR) 1002
31.6.2 WWDG 配置寄存器 (WWDG_CFR) 1003
31.6.3 WWDG 状态寄存器 (WWDG_SR) 1004
31.6.4 WWDG 寄存器映射 1004

32 实时时钟 (RTC) 1005
32.1 简介 1005
32.2 RTC 主要特性 1005
32.3 RTC 功能描述 1005
32.3.1 RTC 框图 1005
32.3.2 RTC 引脚和内部信号 1007
32.3.3 RTC 和 TAMP 控制的 GPIO 1008
32.3.4 RTC 特权保护模式 1011
32.3.5 时钟和预分频器 1012
32.3.6 实时时钟和日历 1013
32.3.7 日历超低功耗模式 1013
32.3.8 可编程闹钟 1013
32.3.9 周期性自动唤醒 1014
32.3.10 RTC 初始化和配置 1014
32.3.11 读取日历 1017
32.3.12 复位 RTC 1017
32.3.13 RTC 同步 1018
32.3.14 RTC 参考时钟检测 1018
32.3.15 RTC 精密数字校准 1019
32.3.16 时间戳功能 1021
32.3.17 校准时钟输出 1021
32.3.18 入侵和闹钟输出 1022

32.4 RTC 低功耗模式 1022
32.5 RTC 中断 1023
32.6 RTC 寄存器 1023
32.6.1 RTC 时间寄存器 (RTC_TR) 1023
32.6.2 RTC 日期寄存器 (RTC_DR) 1024
32.6.3 RTC 亚秒寄存器 (RTC_SSR) 1025
32.6.4 RTC 初始化控制和状态寄存器 (RTC_ICSR) 1026
32.6.5 RTC 预分频器寄存器 (RTC_PRER) 1027
32.6.6 RTC 唤醒定时器寄存器 (RTC_WUTR) 1028
32.6.7 RTC 控制寄存器 (RTC_CR) 1029
32.6.8 RTC 特权模式控制寄存器 (RTC_PRIVCFGR) 1032
32.6.9 RTC 写保护寄存器 (RTC_WPR) 1033
32.6.10 RTC 校准寄存器 (RTC_CALR) 1034
32.6.11 RTC 平移控制寄存器 (RTC_SHIFTR) 1035
32.6.12 RTC 时间戳时间寄存器 (RTC_TSTR) 1036
32.6.13 RTC 时间戳日期寄存器 (RTC_TSDR) 1037
32.6.14 RTC 时间戳亚秒寄存器 (RTC_TSSSR) 1037
32.6.15 RTC 闹钟 A 寄存器 (RTC_ALRMAR) 1038
32.6.16 RTC 闹钟 A 亚秒寄存器 (RTC_ALRMASSR) 1039
32.6.17 RTC 闹钟 B 寄存器 (RTC_ALRMBR) 1040
32.6.18 RTC 闹钟 B 亚秒寄存器 (RTC_ALRMBSSR) 1041
32.6.19 RTC 状态寄存器 (RTC_SR) 1042
32.6.20 RTC 屏蔽中断状态寄存器 (RTC_MISR) 1043
32.6.21 RTC 状态清零寄存器 (RTC_SCR) 1044
32.6.22 RTC 闹钟 A 二进制模式寄存器 (RTC_ALRABINR) 1045
32.6.23 RTC 闹钟 B 二进制模式寄存器 (RTC_ALRBBINR) 1045
32.6.24 RTC 寄存器映射 1046

33 入侵和备份寄存器 (TAMP) 1048
33.1 简介 1048
33.2 TAMP 主要特性 1048
33.3 TAMP 功能描述 1049
33.3.1 TAMP 框图 1049
33.3.2 TAMP 引脚和内部信号 1050
33.3.3 RTC 和 TAMP 控制的 GPIO 1052
33.3.4 TAMP 寄存器写保护 1052

33.3.5 备份寄存器保护区 1052
33.3.6 TAMP 特权保护模式 1052
33.3.7 入侵检测 1053
33.3.8 TAMP 备份寄存器和其他器件机密信息擦除 1053
33.3.9 入侵检测配置和初始化 1054
33.4 TAMP 低功耗模式 1059
33.5 TAMP 中断 1060
33.6 TAMP 寄存器 1060
33.6.1 TAMP 控制寄存器 1 (TAMP_CR1) 1060
33.6.2 TAMP 控制寄存器 2 (TAMP_CR2) 1062
33.6.3 TAMP 控制寄存器 3 (TAMP_CR3) 1063
33.6.4 TAMP 滤波器控制寄存器 (TAMP_FLTCR) 1065
33.6.5 TAMP 主动入侵控制寄存器 1 (TAMP_ATCR1) 1066
33.6.6 TAMP 主动入侵种子寄存器 (TAMP_ATSEEDR) 1068
33.6.7 TAMP 主动入侵输出寄存器 (TAMP_ATOR) 1068
33.6.8 TAMP 主动入侵控制寄存器 2 (TAMP_ATCR2) 1069
33.6.9 TAMP 配置寄存器 (TAMP_CFGR) 1070
33.6.10 TAMP 特权配置寄存器 (TAMP_PRIVCFGR) 1072
33.6.11 TAMP 中断使能寄存器 (TAMP_IER) 1073
33.6.12 TAMP 状态寄存器 (TAMP_SR) 1075
33.6.13 TAMP 屏蔽中断状态寄存器 (TAMP_MISR) 1076
33.6.14 TAMP 状态清零寄存器 (TAMP_SCR) 1078
33.6.15 TAMP 单调计数器 1 寄存器 (TAMP_COUNT1R) 1079
33.6.16 TAMP 资源保护配置寄存器 (TAMP_RPCFGR) 1080
33.6.17 TAMP 备份 x 寄存器 (TAMP_BKPxR) 1081
33.6.18 TAMP 寄存器映射 1082

34 内部集成电路 (I2C) 接口 1084
34.1 简介 1084
34.2 I2C 主要特性 1084
34.3 I2C 特性实现 1085
34.4 I2C 功能描述 1085
34.4.1 I2C 框图 1086
34.4.2 I2C 引脚和内部信号 1087
34.4.3 I2C 时钟要求 1087
34.4.4 模式选择 1088

34.4.5 I2C 初始化 1089
34.4.6 软件复位 1093
34.4.7 数据传输 1093
34.4.8 I2C 从模式 1095
34.4.9 I2C 主模式 1103
34.4.10 I2C_TIMINGR 寄存器配置示例 1114
34.4.11 SMBus 特性 1116
34.4.12 SMBus 初始化 1118
34.4.13 SMBus：I2C_TIMEOUTR 寄存器配置示例 1120
34.4.14 SMBus 从模式 1121
34.4.15 地址匹配时从停止模式唤醒 1127
34.4.16 错误条件 1127
34.4.17 DMA 请求 1129
34.4.18 调试模式 1129
34.5 I2C 低功耗模式 1130
34.6 I2C 中断 1130
34.7 I2C 寄存器 1131
34.7.1 I2C 控制寄存器 1 (I2C_CR1) 1131
34.7.2 I2C 控制寄存器 2 (I2C_CR2) 1134
34.7.3 I2C 自身地址 1 寄存器 (I2C_OAR1) 1136
34.7.4 I2C 自身地址 2 寄存器 (I2C_OAR2) 1137
34.7.5 I2C 时序寄存器 (I2C_TIMINGR) 1138
34.7.6 I2C 超时寄存器 (I2C_TIMEOUTR) 1139
34.7.7 I2C 中断和状态寄存器 (I2C_ISR) 1140
34.7.8 I2C 中断清零寄存器 (I2C_ICR) 1142
34.7.9 I2C PEC 寄存器 (I2C_PECR) 1143
34.7.10 I2C 接收数据寄存器 (I2C_RXDR) 1144
34.7.11 I2C 发送数据寄存器 (I2C_TXDR) 1144
34.7.12 I2C 寄存器映射 1145

35 改进型内部集成电路 (I3C) 1147
35.1 简介 1147
35.2 I3C 主要特性 1147
35.3 I3C 特性实现 1149
35.3.1 I3C 实例化 1149
35.3.2 I3C 从低功耗模式唤醒 1149

35.3.3 I3C FIFO 1149
35.3.4 I3C 触发信号 1149
35.3.5 I3C 中断 1149
35.3.6 I3C MIPI® 支持 1150
35.4 I3C 框图 1151
35.5 I3C 引脚和内部信号 1151
35.6 I3C 复位和时钟 1152
35.6.1 I3C 复位 1152
35.6.2 I3C 时钟和要求 1152
35.7 I3C 外设状态和编程 1154
35.7.1 I3C 外设状态 1154
35.7.2 I3C 控制器状态和编程序列 1155
35.7.3 I3C 目标状态和编程序列 1158
35.8 I3C 寄存器和编程 1162
35.8.1 I3C 寄存器组（作为控制器/目标） 1162
35.8.2 I3C 寄存器和位域用法与外设状态的关系（作为控制器） 1163
35.8.3 I3C 寄存器和位域用法与外设状态的关系（作为目标） 1165
35.9 I3C 总线传输和编程 1167
35.9.1 I3C 命令集 (CCC)（作为控制器/目标） 1167
35.9.2 I3C 广播/直接 CCC 传输（ENTDAA 和 RSTACT 除外）（作为控制器） 1171
35.9.3 I3C 广播 ENTDAA CCC 传输（作为控制器） 1173
35.9.4 I3C 广播/直接 RSTACT CCC 传输（作为控制器） 1173
35.9.5 I3C 广播/直接 CCC 传输（ENTDAA、DEFTGTS、DEFGRPA 除外）（作为目标） 1175
35.9.6 I3C 广播 ENTDAA CCC 传输（作为目标） 1177
35.9.7 I3C 广播 DEFTGTS CCC 传输（作为目标） 1178
35.9.8 I3C 广播 DEFGRPA CCC 传输（作为目标） 1179
35.9.9 I3C 直接 GETSTATUS CCC 响应（作为目标） 1179
35.9.10 I3C 私有读/写传输（作为控制器） 1181
35.9.11 I3C 私有读/写传输（作为目标） 1182
35.9.12 传统 I2C 读/写传输（作为控制器） 1183
35.9.13 I3C IBI 传输（作为控制器/目标） 1184
35.9.14 I3C 热加入请求传输（作为控制器/目标） 1185
35.9.15 I3C 控制器角色请求传输（作为控制器/目标） 1186
35.10 I3C FIFO 管理（作为控制器） 1187
35.10.1 C-FIFO 管理（作为控制器） 1187

35.10.2 TX-FIFO 管理（作为控制器） 1188
35.10.3 RX-FIFO 管理（作为控制器） 1190
35.10.4 S-FIFO 管理（作为控制器） 1192
35.11 I3C FIFO 管理（作为目标） 1194
35.11.1 RX-FIFO 管理（作为目标） 1194
35.11.2 TX-FIFO 管理（作为目标） 1196
35.12 I3C 错误管理 1199
35.12.1 控制器错误管理 1199
35.12.2 目标错误管理 1201
35.13 I3C 从低功耗模式唤醒 1202
35.13.1 从停止模式唤醒 1202
35.14 低功耗模式下的 I3C 1204
35.15 I3C 中断 1205
35.16 I3C 寄存器 1206
35.16.1 I3C 消息控制寄存器 (I3C_CR) 1206
35.16.2 I3C 消息控制寄存器 [复用] (I3C_CR) 1208
35.16.3 I3C 配置寄存器 (I3C_CFGR) 1210
35.16.4 I3C 接收数据字节寄存器 (I3C_RDR) 1214
35.16.5 I3C 接收数据字寄存器 (I3C_RDWR) 1215
35.16.6 I3C 发送数据字节寄存器 (I3C_TDR) 1216
35.16.7 I3C 发送数据字寄存器 (I3C_TDWR) 1217
35.16.8 I3C IBI 有效负载数据寄存器 (I3C_IBIDR) 1218
35.16.9 I3C 目标发送配置寄存器 (I3C_TGTTDR) 1219
35.16.10 I3C 状态寄存器 (I3C_SR) 1220
35.16.11 I3C 状态错误寄存器 (I3C_SER) 1221
35.16.12 I3C 接收消息寄存器 (I3C_RMR) 1223
35.16.13 I3C 事件寄存器 (I3C_EVR) 1223
35.16.14 I3C 中断使能寄存器 (I3C_IER) 1227
35.16.15 I3C 清零事件寄存器 (I3C_CEVR) 1230
35.16.16 I3C 自身器件特性寄存器 (I3C_DEVR0) 1232
35.16.17 I3C 设备 x 特性寄存器 (I3C_DEVRx) 1234
35.16.18 I3C 最大读取长度寄存器 (I3C_MAXRLR) 1236
35.16.19 I3C 最大写入长度寄存器 (I3C_MAXWLR) 1237
35.16.20 I3C 时序寄存器 0 (I3C_TIMINGR0) 1237
35.16.21 I3C 时序寄存器 1 (I3C_TIMINGR1) 1238
35.16.22 I3C 时序寄存器 2 (I3C_TIMINGR2) 1240

35.16.23 I3C 总线特性寄存器 (I3C_BCR) 1242
35.16.24 I3C 设备特性寄存器 (I3C_DCR) 1243
35.16.25 I3C 获取功能寄存器 (I3C_GETCAPR) 1244
35.16.26 I3C 控制器角色功能寄存器 (I3C_CRCAPR) 1245
35.16.27 I3C 获取最大数据速度寄存器 (I3C_GETMXDSR) 1246
35.16.28 I3C 扩展配置 ID 寄存器 (I3C_EPIDR) 1248
35.16.29 I3C 寄存器映射 1248

36 通用同步/异步收发器 (USART/UART) 1252

36.1 简介 1252
36.2 USART 主要特性 1252
36.3 USART 扩展特性 1253
36.4 USART 实现 1253
36.5 USART 功能描述 1254
36.5.1 USART 框图 1254
36.5.2 USART 引脚和内部信号 1255
36.5.3 USART 时钟 1256
36.5.4 USART 字符说明 1257
36.5.5 USART FIFO 和阈值 1259
36.5.6 USART 发送器 1259
36.5.7 USART 接收器 1262
36.5.8 USART 波特率生成 1269
36.5.9 USART 接收器对时钟偏差的容差 1271
36.5.10 USART 自动波特率检测 1272
36.5.11 USART 多处理器通信 1273
36.5.12 USART Modbus 通信 1275
36.5.13 USART 奇偶性控制 1276
36.5.14 USART LIN（局域互连网络）模式 1277
36.5.15 USART 同步模式 1279
36.5.16 USART 单线半双工通信 1283
36.5.17 USART 接收器超时 1283
36.5.18 USART 智能卡模式 1284
36.5.19 USART IrDA SIR ENDEC 模块 1288
36.5.20 使用 USART 和 DMA 进行连续通信 1290
36.5.21 RS232 硬件流控制和 RS485 驱动器使能 1292
36.5.22 USART 低功耗管理 1295

36.6 低功耗模式下的 USART 1298
36.7 USART 中断 1298
36.8 USART 寄存器 1300
36.8.1 USART 控制寄存器 1 (USART_CR1) 1300
36.8.2 USART 控制寄存器 1 [备用] (USART_CR1) 1303
36.8.3 USART 控制寄存器 2 (USART_CR2) 1306
36.8.4 USART 控制寄存器 3 (USART_CR3) 1310
36.8.5 USART 波特率寄存器 (USART_BRR) 1314
36.8.6 USART 保护时间和预分频器寄存器 (USART_GTPR) 1314
36.8.7 USART 接收器超时寄存器 (USART_RTOR) 1315
36.8.8 USART 请求寄存器 (USART_RQR) 1316
36.8.9 USART 中断和状态寄存器 (USART_ISR) 1317
36.8.10 USART 中断和状态寄存器 [备用] (USART_ISR) 1322
36.8.11 USART 中断标志清零寄存器 (USART_ICR) 1327
36.8.12 USART 接收数据寄存器 (USART_RDR) 1328
36.8.13 USART 发送数据寄存器 (USART_TDR) 1329
36.8.14 USART 预分频器寄存器 (USART_PRESC) 1329
36.8.15 USART 寄存器映射 1330

37 低功耗通用异步收发器 (LPUART) 1332
37.1 简介 1332
37.2 LPUART 主要特性 1332
37.3 LPUART 实现 1333
37.4 LPUART 功能描述 1334
37.4.1 LPUART 框图 1334
37.4.2 LPUART 引脚和内部信号 1335
37.4.3 LPUART 时钟 1336
37.4.4 LPUART 字符说明 1336
37.4.5 LPUART FIFO 和阈值 1338
37.4.6 LPUART 发送器 1338
37.4.7 LPUART 接收器 1341
37.4.8 LPUART 波特率生成 1344
37.4.9 LPUART 接收器对时钟偏差的容差 1346
37.4.10 LPUART 多处理器通信 1347
37.4.11 LPUART 极性控制 1348
37.4.12 LPUART 单线半双工通信 1349

37.4.13 使用 DMA 和 LPUART 进行连续通信 1350
37.4.14 RS232 硬件流控制和 RS485 驱动器使能 1353
37.4.15 LPUART 低功耗管理 1355
37.5 低功耗模式下的 LPUART 1358
37.6 LPUART 中断 1359
37.7 LPUART 寄存器 1360
37.7.1 LPUART 控制寄存器 1 (LPUART_CR1) 1360
37.7.2 LPUART 控制寄存器 1 [备用] (LPUART_CR1) 1363
37.7.3 LPUART 控制寄存器 2 (LPUART_CR2) 1365
37.7.4 LPUART 控制寄存器 3 (LPUART_CR3) 1367
37.7.5 LPUART 波特率寄存器 (LPUART_BRR) 1370
37.7.6 LPUART 请求寄存器 (LPUART_RQR) 1370
37.7.7 LPUART 中断和状态寄存器 (LPUART_ISR) 1371
37.7.8 LPUART 中断和状态寄存器 [备用] (LPUART_ISR) 1375
37.7.9 LPUART 中断标志清零寄存器 (LPUART_ICR) 1378
37.7.10 LPUART 接收数据寄存器 (LPUART_RDR) 1379
37.7.11 LPUART 发送数据寄存器 (LPUART_TDR) 1379
37.7.12 LPUART 预分频器寄存器 (LPUART_PRESC) 1380
37.7.13 LPUART 寄存器映射 1380

38 串行外设接口 (SPI) 1382
38.1 简介 1382
38.2 SPI 主要特性 1382
38.3 SPI 实现 1383
38.4 SPI 功能描述 1384
38.4.1 SPI 框图 1384
38.4.2 SAI 引脚和内部信号 1385
38.4.3 SPI 通信一般情况 1386
38.4.4 一个主器件和一个从器件之间的通信 1386
38.4.5 标准多从器件通信 1388
38.4.6 多主器件通信 1391
38.4.7 从器件选择 (SS) 引脚管理 1391
38.4.8 就绪引脚 (RDY) 管理 1395
38.4.9 通信格式 1395
38.4.10 配置 SPI 1397
38.4.11 使能 SPI 1398

38.4.12 SPI 数据发送和接收过程 1398
38.4.13 关闭 SPI 1401
38.4.14 数据封装 1403
38.4.15 使用 DMA（直接存储器寻址）进行通信 1404
38.5 SPI 特定模式和控制 1405
38.5.1 TI 模式 1405
38.5.2 SPI 错误标志 1406
38.5.3 CRC 计算 1408
38.6 SPI 低功耗模式 1409
38.7 SPI 中断 1410
38.8 I2S 主要特性 1411
38.9 I2S 功能描述 1411
38.9.1 I2S 一般描述 1411
38.9.2 与 SPI 功能共享的引脚 1412
38.9.3 I2S/PCM 模式下可用的位域 1412
38.9.4 从模式和主模式 1413
38.9.5 支持的音频协议 1413
38.9.6 更灵活的串行接口 1419
38.9.7 启动序列 1421
38.9.8 停止序列 1423
38.9.9 时钟发生器 1423
38.9.10 内部 FIFO 1425
38.9.11 FIFO 状态标志 1426
38.9.12 下溢情况的处理 1427
38.9.13 上溢情况的处理 1428
38.9.14 帧错误检测 1428
38.9.15 DMA 接口 1430
38.9.16 编程示例 1430
38.10 I2S 中断 1432
38.11 SPI/I2S 寄存器 1433
38.11.1 SPI/I2S 控制寄存器 1 (SPI_CR1) 1433
38.11.2 SPI/I2S 控制寄存器 2 (SPI_CR2) 1435
38.11.3 SPI/I2S 配置寄存器 1 (SPI_CFG1) 1435
38.11.4 SPI/I2S 配置寄存器 2 (SPI_CFG2) 1438
38.11.5 SPI/I2S 中断使能寄存器 (SPI_IER) 1440
38.11.6 SPI/I2S 状态寄存器 (SPI_SR) 1441

38.11.7 SPI/I2S 中断/状态标志清零寄存器 (SPI_IFCR) 1444
38.11.8 SPI/I2S 发送数据寄存器 (SPI_TXDR) 1445
38.11.9 SPI/I2S 接收数据寄存器 (SPI_RXDR) 1445
38.11.10 SPI/I2S 多项式寄存器 (SPI_CRCPOLY) 1446
38.11.11 SPI/I2S 发送器 CRC 寄存器 (SPI_TXCRC) 1447
38.11.12 SPI/I2S 接收器 CRC 寄存器 (SPI_RXCRC) 1447
38.11.13 SPI/I2S 下溢数据寄存器 (SPI_UDRDR) 1448
38.11.14 SPI/I2S 配置寄存器 (SPI_I2SCFGR) 1449
38.11.15 SPI/I2S 寄存器映射 1450

39 FD 控制器局域网络 (FDCAN) 1452

39.1 简介 1452
39.2 FDCAN 主要特性 1454
39.3 FDCAN 功能说明 1455
39.3.1 位时序 1456
39.3.2 工作模式 1457
39.3.3 消息 RAM 1464
39.3.4 FIFO 确认处理 1472
39.3.5 FDCAN 接收 FIFO 元素 1472
39.3.6 FDCAN 发送缓冲区元素 1474
39.3.7 FDCAN 发送事件 FIFO 元素 1476
39.3.8 FDCAN 标准消息 ID 过滤器元素 1477
39.3.9 FDCAN 扩展消息 ID 过滤器元素 1478
39.4 FDCAN 寄存器 1480
39.4.1 FDCAN 内核释放寄存器 (FDCAN_CREL) 1480
39.4.2 FDCAN 字节序寄存器 (FDCAN_ENDN) 1480
39.4.3 FDCAN 数据位定时和预分频器寄存器 (FDCAN_DBTP) 1481
39.4.4 FDCAN 测试寄存器 (FDCAN_TEST) 1482
39.4.5 FDCAN RAM 看门狗寄存器 (FDCAN_RWD) 1483
39.4.6 FDCAN CC 控制寄存器 (FDCAN_CCCR) 1483
39.4.7 FDCAN 标称位定时和预分频器寄存器 (FDCAN_NBTP) 1485
39.4.8 FDCAN 时间戳计数器配置寄存器 (FDCAN_TSCC) 1486
39.4.9 FDCAN 时间戳计数器值寄存器 (FDCAN_TSCV) 1486
39.4.10 FDCAN 超时计数器配置寄存器 (FDCAN_TOCC) 1487
39.4.11 FDCAN 超时计数器值寄存器 (FDCAN_TOCV) 1488
39.4.12 FDCAN 错误计数器寄存器 (FDCAN_ECR) 1488
39.4.13 FDCAN 协议状态寄存器 (FDCAN_PSR) 1489

39.4.14 FDCAN 发送器延迟补偿寄存器 (FDCAN_TDCR) 1491
39.4.15 FDCAN 中断寄存器 (FDCAN_IR) 1491
39.4.16 FDCAN 中断使能寄存器 (FDCAN_IE) 1494
39.4.17 FDCAN 中断线选择寄存器 (FDCAN_ILS) 1496
39.4.18 FDCAN 中断线使能寄存器 (FDCAN_ILE) 1497
39.4.19 FDCAN 全局过滤器配置寄存器 (FDCAN_RXGFC) 1497
39.4.20 FDCAN 扩展 ID 和掩码寄存器 (FDCAN_XIDAM) 1499
39.4.21 FDCAN 高优先级消息状态寄存器 (FDCAN_HPMS) 1499
39.4.22 FDCAN 接收 FIFO 0 状态寄存器 (FDCAN_RXF0S) 1500
39.4.23 CAN 接收 FIFO 0 确认寄存器 (FDCAN_RXF0A) 1501
39.4.24 FDCAN 接收 FIFO 1 状态寄存器 (FDCAN_RXF1S) 1501
39.4.25 FDCAN 接收 FIFO 1 确认寄存器 (FDCAN_RXF1A) 1502
39.4.26 FDCAN 发送缓冲区配置寄存器 (FDCAN_TXBC) 1502
39.4.27 FDCAN 发送 FIFO/队列状态寄存器 (FDCAN_TXFQS) 1503
39.4.28 FDCAN 发送缓冲区请求挂起寄存器 (FDCAN_TXBRP) 1503
39.4.29 FDCAN 发送缓冲区添加请求寄存器 (FDCAN_TXBAR) 1504
39.4.30 FDCAN 发送缓冲区取消请求寄存器 (FDCAN_TXBCR) 1505
39.4.31 FDCAN 发送缓冲区发送已发生寄存器 (FDCAN_TXBTO) 1505
39.4.32 FDCAN 发送缓冲区取消完成寄存器 (FDCAN_TXBCF) 1506
39.4.33 FDCAN 发送缓冲区发送中断使能寄存器 (FDCAN_TXBTIE) 1506
39.4.34 FDCAN 发送缓冲区取消完成中断使能寄存器 (FDCAN_TXBCIE) . . . 1507
39.4.35 FDCAN 发送事件 FIFO 状态寄存器 (FDCAN_TXEFS) 1507
39.4.36 FDCAN 发送事件 FIFO 确认寄存器 (FDCAN_TXEFA) 1508
39.4.37 FDCAN CFG 时钟分频寄存器（FDCAN_CKDIV) 1508
39.4.38 FDCAN 寄存器映射 1509

40 通用串行总线全速主机/设备接口 (USB) 1513

40.1 简介 1513
40.2 USB 主要特性 1513
40.3 USB 实现 1513
40.4 USB 功能描述 1514
40.4.1 设备和主机模式下所用 USB 模块的说明 1515
40.4.2 特定于主机模式的主机帧调度器 (HFS) 的说明 1516
40.5 设备模式和主机模式下的编程注意事项 1517
40.5.1 通用 USB 设备编程 1517
40.5.2 系统复位和上电复位 1517
40.5.3 设备模式下双缓冲端点和使用 1523

40.5.4 双缓冲通道：主机模式下的使用 1525
40.5.5 设备模式下的同步传输 1526
40.5.6 主机模式下的同步传输 1527
40.5.7 挂起/恢复事件 1527
40.6 USB 和 USB SRAM 寄存器 1529
40.6.1 通用寄存器 1530
40.6.2 缓冲区描述符表 1545
40.6.3 USB 寄存器映射 1549

41 调试支持 (DBG) 1551
41.1 简介 1551
41.2 DBG 功能描述 1552
41.2.1 DBG 框图 1552
41.2.2 DBG 引脚和内部信号 1552
41.2.3 DBG 复位和时钟 1553
41.2.4 DBG 电源域 1553
41.2.5 调试和低功耗模式 1553
41.2.6 安全性 1554
41.2.7 调试身份验证 1555
41.3 串行线和 JTAG 调试端口 (SWJ-DP) 1556
41.3.1 JTAG 调试端口 1557
41.3.2 串行线调试端口 1559
41.3.3 调试端口寄存器 1560
41.3.4 调试端口寄存器映射和复位值 1567
41.4 访问端口 1568
41.4.1 访问端口寄存器 1569
41.4.2 访问端口寄存器映射 1575
41.5 ROM 表 1576
41.5.1 系统 ROM 表寄存器 1579
41.5.2 系统 ROM 表寄存器映射 1583
41.5.3 MCU ROM 表寄存器 1584
41.5.4 MCU ROM 表寄存器映射 1589
41.5.5 处理器 ROM 表寄存器 1590
41.5.6 处理器 ROM 表寄存器映射 1595
41.6 数据观察点和跟踪单元 (DWT) 1596
41.6.1 DWT 寄存器 1597

41.6.2 DWT 寄存器映射 1612
41.7 指令跟踪宏单元 (ITM) 1614
41.7.1 ITM 寄存器 1615
41.7.2 ITM 寄存器映射 1623
41.8 断点单元 (BPU) 1625
41.8.1 BPU 寄存器 1625
41.8.2 BPU 寄存器映射 1631
41.9 嵌入式跟踪宏单元(ETM) 1633
41.9.1 ETM 寄存器 1633
41.9.2 ETM 寄存器映射 1660
41.10 跟踪端口接口单元 (TPIU) 1664
41.10.1 TPIU 寄存器 1665
41.10.2 TPIU 寄存器映射 1675
41.11 交叉触发接口 (CTI) 1677
41.11.1 CTI 寄存器 1679
41.11.2 CTI 寄存器映射 1690
41.12 微控制器调试单元 (DBGMCU) 1692
41.12.1 器件 ID 1692
41.12.2 低功耗模式仿真 1692
41.12.3 外设时钟冻结 1693
41.12.4 DBGMCU 寄存器 1694
41.12.5 DBGMCU 寄存器映射 1708
41.13 参考 1710

42 器件电子签名 1711
42.1 唯一器件 ID 寄存器（96 位） 1711
42.2 Flash 大小数据寄存器 1712
42.3 封装数据寄存器 1713

43 重要安全说明 1714

44 版本历史 1715

表格索引

表 1. 存储器映射和外设寄存器边界地址 70
表 2. DMA 通道使用（特权） 78
表 3. TAMP 中的内部入侵 80
表 4. 低功耗模式对 TAMP 的影响 82
表 5. 主要产品生命周期转换 83
表 6. 典型产品生命周期阶段 84
表 7. 通过 PRODUCT_STATE 实现软件知识产权保护 87
表 8. STM32H503 自举模式 88
表 9. 内部 SRAM 特性 90
表 10. 低功耗模式对 RAMCFG 的影响 92
表 11. RAMCFG 中断请求 93
表 12. RAMCFG 寄存器映射和复位值 98
表 13. GTZC 实现 102
表 14. GTZC 特性 102
表 15. GTZC1 子模块偏移地址 103
表 16. MPCWM 资源分配 103
表 17. MPCBB 资源分配 103
表 18. GTZC1 TZSC 寄存器映射和复位值 112
表 19. GTZC1 MPCBBz 寄存器映射和复位值（z = 1 和 2） 113
表 20. FLASH 推荐的等待周期数和编程延迟 120
表 21. Flash OTP 结构 126
表 22. 只读公共数据结构 128
表 23. Flash 只读结构 128
表 24. 存储器映射和交换选项 128
表 25. 针对 FLASH_OPSR 内容推荐的反应 130
表 26. 选项字节构成 132
表 27. 特定 OB 修改规则概述 136
表 28. 关闭状态的产品中可修改的 OB 136
表 29. HDP 保护定义 138
表 30. 安全隐藏保护 139
表 31. HDP 保护汇总 139
表 32. 特权保护汇总 140
表 33. 特权与批量/存储区擦除 140
表 34. 特权配置寄存器访问条件 140
表 35. Flash 接口寄存器保护汇总 143
表 36. 产品状态、调试状态和调试策略 143
表 37. PRODUCT_STATE 转换 144
表 38. OTP/RO 访问限制 146
表 39. RSS 服务的 C 定义宏 146
表 40. NSSlib 接口函数列表 146
表 41. 低功耗模式对嵌入式 Flash 的影响 148
表 42. 定位 ECC 故障 152
表 43. Flash 中断请求 154
表 44. 寄存器映射与复位值表 179
表 45. ICACHE 特性 184
表 46. n 路组相联工作模式（默认）的标记存储器尺寸参数值 186
表 47. 直接映射缓存模式下的标记存储器尺寸参数 187
表 48. AHB 事务的 ICACHE 可缓存性 188

表 49. ICACHE 中断 . . . 191
表 50. ICACHE 寄存器映射和复位值 . . . 194
表 51. PWR 输入/输出引脚 . . . 197
表 52. PWR 内部输入/输出信号 . . . 197
表 53. 推荐的模拟开关配置 . . . 200
表 54. 低功耗模式汇总 . . . 212
表 55. 功能取决于工作模式 . . . 212
表 56. 睡眠模式 . . . 215
表 57. 停止模式 . . . 217
表 58. 待机模式 . . . 219
表 59. 功耗模式输出状态与 MCU 功耗模式的关系 . . . 220
表 60. PWR 特权配置汇总 . . . 220
表 61. PWR 中断请求 . . . 221
表 62. PWR 寄存器映射和复位值 . . . 233
表 63. 连接到封装引脚或焊球的 RCC 输入/输出信号 . . . 235
表 64. 复位源标识 (RCC_RSR) . . . 237
表 65. STOPWUCK 和 STOPKERWUCK 说明 . . . 249
表 66. HSIKERON 和 CSIKERON 行为 . . . 250
表 67. 内核时钟分配概述 . . . 252
表 68. RCC 特权配置汇总 . . . 256
表 69. 中断源和控制 . . . 258
表 70. RCC 寄存器映射和复位值 . . . 312
表 71. CRS 特性 . . . 318
表 72. CRS 内部输入/输出信号 . . . 318
表 73. 低功耗模式对 CRS 的影响 . . . 322
表 74. 中断控制位 . . . 322
表 75. CRS 寄存器映射和复位值 . . . 328
表 76. 端口位配置 . . . 330
表 77. GPIO 寄存器映射和复位值 . . . 344
表 78. SBS 内部输入/输出信号 . . . 347
表 79. HDPL 编码值 . . . 350
表 80. SBS 自举逻辑 . . . 351
表 81. 调试认证逻辑 . . . 353
表 82. SBS 寄存器映射和复位值 . . . 363
表 83. 外设互连矩阵 . . . 365
表 84. GPDMA1/2 通道实现 . . . 376
表 85. 低功耗模式下的 GPDMA1/2 自主模式和唤醒功能 . . . 377
表 86. 编程设定的 GPDMA1/2 请求 . . . 377
表 87. 编程设定为块请求的 GPDMA1/2 请求 . . . 381
表 88. 支持外设提前终止的 GPDMA1/2 通道 . . . 382
表 89. 编程设定为支持外设提前终止的 GPDMA1/2 请求 . . . 382
表 90. 编程设定的 GPDMA1/2 触发信号 . . . 382
表 91. 编程设定的 GPDMA 源/目标突发 . . . 404
表 92. 编程设定的数据处理 . . . 408
表 93. 低功耗模式对 GPDMA 的影响 . . . 421
表 94. GPDMA 中断请求 . . . 421
表 95. GPDMA 寄存器映射和复位值 . . . 445
表 96. STM32H503xx 向量表 . . . 448
表 97. EXTI 引脚概述 . . . 453
表 98. EVG 引脚概述 . . . 453
表 99. EXTI 线连接 . . . 454
表 100. 屏蔽功能 . . . 457

表 101. 寄存器保护概述 . . . 458
表 102. EXTI 寄存器映射的各个部分 . . . 458
表 103. EXTI 寄存器映射和复位值 . . . 475
表 104. CRC 内部输入/输出信号 . . . 479
表 105. CRC 寄存器映射和复位值 . . . 484
表 106. ADC 特性 . . . 487
表 107. ADC 输入/输出引脚 . . . 489
表 108. ADC 内部输入/输出信号 . . . 489
表 109. ADC 互连 . . . 489
表 110. 为常规外部触发配置触发极性 . . . 506
表 111. 为注入外部触发配置触发极性 . . . 506
表 112. TSAR 与分辨率的对应关系 . . . 517
表 113. 偏移计算与数据分辨率 . . . 520
表 114. 模拟看门狗通道选择 . . . 530
表 115. 模拟看门狗 1 比较 . . . 531
表 116. 模拟看门狗 2 和 3 比较 . . . 531
表 117. 最大输出结果与 N 和 M 的对应关系（灰色单元格表示截断的部分） . . . 535
表 118. ADC 中断 . . . 542
表 119. ADC 全局寄存器映射 . . . 572
表 120. ADC 寄存器映射和复位值 . . . 572
表 121. ADC 寄存器映射和复位值（主 ADC 和从 ADC 通用寄存器） . . . 574
表 122. DTS 内部输入/输出信号 . . . 576
表 123. 采样时间配置 . . . 579
表 124. 触发配置 . . . 580
表 125. 低功耗模式下的温度传感器行为 . . . 581
表 126. 中断控制位 . . . 582
表 127. DTS 寄存器映射和复位值 . . . 591
表 128. DAC 特性 . . . 592
表 129. DAC 输入/输出引脚 . . . 594
表 130. DAC 内部输入/输出信号 . . . 594
表 131. DAC 互连 . . . 595
表 132. 数据格式（以 12 位数据为例） . . . 597
表 133. HFSEL 说明 . . . 597
表 134. 采样时间和刷新时间 . . . 602
表 135. 通道输出模式汇总 . . . 604
表 136. 低功耗模式对 DAC 的影响 . . . 609
表 137. DAC 中断 . . . 609
表 138. DAC 寄存器映射和复位值 . . . 626
表 139. COMP1 输入/输出内部信号 . . . 629
表 140. COMP1 消隐源选择 . . . 630
表 141. COMP1 同相输入分配 . . . 630
表 142. COMP1 反相输入分配 . . . 630
表 143. COMP1 消隐源分配 . . . 630
表 144. 低功耗模式下的比较器行为 . . . 633
表 145. 通过 EXTI 控制位实现的中断 . . . 633
表 146. 通过 NVIC 控制位实现的中断 . . . 633
表 147. COMP 寄存器映射和复位值 . . . 639
表 148. OPAMP 的可能连接 . . . 641
表 149. 工作模式和校准 . . . 648
表 150. 低功耗模式对 OPAMP 的影响 . . . 649
表 151. OPAMP 寄存器映射和复位值 . . . 653
表 152. RNG 内部输入/输出信号 . . . 655

表 153. RNG 中断请求 . . . 662
表 154. RNG 配置 . . . 663
表 155. RNG 寄存器映射和复位值 . . . 668
表 156. HASH 内部输入/输出信号 . . . 670
表 157. 所支持哈希算法的相关信息 . . . 671
表 158. 哈希处理器输出 . . . 673
表 159. HASH 中断请求 . . . 679
表 160. 处理时间（时钟周期数） . . . 680
表 161. HASH1 寄存器映射和复位值 . . . 688
表 162. TIM 输入/输出引脚 . . . 691
表 163. TIM 内部输入/输出信号 . . . 691
表 164. tim_ti1 输入复用器的互连 . . . 692
表 165. tim_ti2 输入复用器的互连 . . . 692
表 166. tim_ti3 输入复用器的互连 . . . 692
表 167. tim_ti4 输入复用器的互连 . . . 693
表 168. 内部触发连接 . . . 693
表 169. tim_etr 输入复用器的互连 . . . 693
表 170. 定时器刹车互连 . . . 694
表 171. 定时器刹车 2 互连 . . . 694
表 172. 系统刹车互连 . . . 694
表 173. CCR 和 ARR 寄存器变化抖动模式 . . . 724
表 174. 中心对齐 PWM 模式下的 CCR 寄存器变化抖动模式 . . . 725
表 175. 定时器输出行为与 tim_brk/tim_brk2 输入的关系 . . . 736
表 176. 刹车保护解除条件 . . . 737
表 177. 计数方向与编码器信号的关系 (CC1P = CC2P = 0) . . . 746
表 178. 计数方向与编码器信号和极性设置的关系 . . . 750
表 179. DMA 请求 . . . 770
表 180. 低功耗模式对 TIM1 的影响 . . . 771
表 181. 中断请求 . . . 771
表 182. 具有刹车功能的互补通道 tim_ocx 和 tim_ocxn 的输出控制位 . . . 795
表 183. TIM1 寄存器映射和复位值 . . . 816
表 184. STM32H503xx 通用定时器 . . . 819
表 185. TIM 输入/输出引脚 . . . 821
表 186. TIM 内部输入/输出信号 . . . 821
表 187. tim_ti1 输入复用器的互连 . . . 822
表 188. tim_ti2 输入复用器的互连 . . . 822
表 189. tim_ti3 输入复用器的互连 . . . 822
表 190. tim_ti4 输入复用器的互连 . . . 822
表 191. TIMx 内部触发连接 . . . 823
表 192. tim_etr 输入复用器的互连 . . . 823
表 193. CCR 和 ARR 寄存器变化抖动模式 . . . 852
表 194. 中心对齐 PWM 模式下的 CCR 寄存器变化抖动模式 . . . 853
表 195. 计数方向与编码器信号的关系 (CC1P = CC2P = 0) . . . 862
表 196. 计数方向与编码器信号和极性设置的关系 . . . 866
表 197. DMA 请求 . . . 889
表 198. 低功耗模式对 TIM2/TIM3 的影响 . . . 889
表 199. 中断请求 . . . 890
表 200. 标准 tim_ocx 通道的输出控制位 . . . 907
表 201. TIM2/TIM3 寄存器映射和复位值 . . . 921
表 202. TIM 内部输入/输出信号 . . . 925
表 203. TIMx_ARR 寄存器变化抖动模式 . . . 935
表 204. DMA 请求 . . . 936

表 205. 低功耗模式对 TIM6/TIM7 的影响 936
表 206. 中断请求 936
表 207. TIMx 寄存器映射和复位值 942
表 208. STM32H503xx LPTIM 特性 944
表 209. LPTIM1/2 输入/输出引脚 946
表 210. LPTIM1/2 内部信号 946
表 211. LPTIM1/2 外部触发连接 947
表 212. LPTIM1/2 输入 1 连接 947
表 213. LPTIM1/2 输入 2 连接 947
表 214. LPTIM1/2 输入捕获 1 连接 947
表 215. LPTIM1/2 输入捕获 2 连接 948
表 216. 预分频器的分频比 949
表 217. 编码器计数方案 955
表 218. 输入捕获干扰滤波器延迟（在计步器单元中） 959
表 219. 低功耗模式对 LPTIM 的影响 962
表 220. 中断事件 963
表 221. LPTIM 寄存器映射和复位值 983
表 222. IWDG 特性 985
表 223. IWDG 内部输入/输出信号 986
表 224. 低功耗模式对 IWDG 的影响 990
表 225. IWDG 中断请求 992
表 226. IWDG 寄存器映射和复位值 997
表 227. WWDG 特性 998
表 228. WWDG 内部输入/输出信号 999
表 229. WWDG 中断请求 1002
表 230. WWDG 寄存器映射和复位值 1004
表 231. RTC 输入/输出引脚 1007
表 232. RTC 内部输入/输出信号 1007
表 233. RTC 互连 1008
表 234. RTC 引脚 PC13 配置 1008
表 235. RTC_OUT 映射 1010
表 236. 低功耗模式对 RTC 的影响 1022
表 237. 各个模式下的 RTC 引脚功能 1022
表 238. 中断请求 1023
表 239. RTC 寄存器映射和复位值 1046
表 240. TAMP 输入/输出引脚 1050
表 241. TAMP 内部输入/输出信号 1050
表 242. TAMP 互连 1051
表 243. 器件资源 x 入侵保护 1054
表 244. 主动入侵输出变化周期 1056
表 245. 最小 ATPER 值 1057
表 246. 主动入侵滤波脉冲的持续时间 1058
表 247. 低功耗模式对 TAMP 的影响 1059
表 248. 各个模式下的 TAMP 引脚功能 1059
表 249. 中断请求 1060
表 250. TAMP 寄存器映射和复位值 1082
表 251. STM32H503xx I2C 特性实现 1085
表 252. I2C 输入/输出引脚 1087
表 253. I2C 内部输入/输出信号 1087
表 254. 模拟滤波器与数字滤波器对比 1089
表 255. I2C-SMBus 规范数据建立和保持时间 1091
表 256. I2C 配置 1095

表 257. I2C-SMBus 规范时钟时序 . . . 1105
表 258. fI2CCLK = 8 MHz 时的时序设置示例 . . . 1115
表 259. fI2CCLK = 16 MHz 时的时序设置示例 . . . 1115
表 260. SMBus 超时规范 . . . 1117
表 261. 带 PEC 的 SMBus 配置 . . . 1119
表 262. TIMEOUTA 设置示例（最大 $t_{TIMEOUT}$ = 25 ms） . . . 1120
表 263. TIMEOUTB 设置示例 . . . 1120
表 264. TIMEOUTA 设置示例（最大 t_{IDLE} = 50 µs） . . . 1120
表 265. 低功耗模式对 I2C 的影响 . . . 1130
表 266. I2C 中断请求 . . . 1130
表 267. I2C 寄存器映射和复位值 . . . 1145
表 268. I3C 唤醒 . . . 1149
表 269. I3C FIFO 实现 . . . 1149
表 270. I3C 中断 . . . 1149
表 271. I3C 外设控制器/目标特性与 MIPI v1.1 的关系 . . . 1150
表 272. I3C 输入/输出引脚 . . . 1151
表 273. I3C 内部输入/输出信号 . . . 1151
表 274. I3C 寄存器用法 . . . 1162
表 275. I3C 寄存器/位域用法与控制器状态的关系 . . . 1163
表 276. I3C 寄存器/位域用法与目标状态的关系 . . . 1165
表 277. 支持的 I3C CCC 列表（作为控制器/目标） . . . 1168
表 278. I3C 控制器错误管理 . . . 1199
表 279. I3C 目标错误管理 . . . 1201
表 280. 低功耗模式的影响 . . . 1204
表 281. I3C 中断请求 . . . 1205
表 282. I3C 寄存器映射和复位值 . . . 1248
表 283. STM32H503xx 特性 . . . 1253
表 284. USART/LPUART 特性 . . . 1253
表 285. USART 输入/输出引脚 . . . 1255
表 286. USART 内部输入/输出信号 . . . 1256
表 287. 通过采样数据进行噪声检测 . . . 1268
表 288. BRR [3:0] = 0000 时的 USART 接收器容差 . . . 1271
表 289. BRR[3:0] 不等于 0000 时的 USART 接收器容差 . . . 1272
表 290. USART 帧格式 . . . 1276
表 291. 低功耗模式对 USART 的影响 . . . 1298
表 292. USART 中断请求 . . . 1298
表 293. USART 寄存器映射和复位值 . . . 1330
表 294. STM32H503xx 特性 . . . 1333
表 295. USART/LPUART 特性 . . . 1333
表 296. LPUART 输入/输出引脚 . . . 1335
表 297. LPUART 内部输入/输出信号 . . . 1335
表 298. lpuart_ker_ck_pres = 32.768 kHz 时编程的波特率的误差计算 . . . 1345
表 299. fCK = 100 MHz 时编程的波特率的误差计算 . . . 1345
表 300. LPUART 接收器的容差 . . . 1346
表 302. 低功耗模式对 LPUART 的影响 . . . 1358
表 303. LPUART 中断请求 . . . 1359
表 304. LPUART 寄存器映射和复位值 . . . 1380
表 305. SPI 特性 . . . 1383
表 306. SPI/I2S 输入/输出引脚 . . . 1385
表 307. SPI 内部输入/输出信号 . . . 1386
表 308. 低功耗模式对 SPI 的影响 . . . 1409
表 309. SPI 唤醒和中断请求 . . . 1410

表 310. PCM/I2S 模式下可用的位域 1412
表 311. AFCNTR = 1 时在使能 SPI/I2S 之前的 WS 和 CK 电平 1420
表 312. 串行数据线交换 1421
表 313. 适合常见 I2S 频率的 CLKGEN 编程示例 1425
表 314. I2S 中断请求 1432
表 315. SPI 寄存器映射和复位值 1450
表 316. CAN 子系统 I/O 信号 1452
表 317. FDCAN 中的 DLC 编码 1459
表 318. 帧发送的可能配置 1469
表 319. 接收 FIFO 元素 1472
表 320. 接收 FIFO 元素说明 1473
表 321. 发送缓冲区和 FIFO 元素 1474
表 322. 发送缓冲区元素说明 1475
表 323. 发送事件 FIFO 元素 1476
表 324. 发送事件 FIFO 元素说明 1476
表 325. 标准消息 ID 过滤器元素 1477
表 326. 标准消息 ID 过滤器元素位域说明 1478
表 327. 扩展消息 ID 过滤器元素 1478
表 328. 扩展消息 ID 过滤器元素位域说明 1479
表 329. FDCAN 寄存器映射和复位值 1509
表 330. STM32H503xxUSB 实现 1513
表 331. 双缓冲缓冲区标志定义 1523
表 332. 批量双缓冲存储器缓冲区用法 1524
表 333. 批量双缓冲存储器缓冲区用法（主机模式） 1525
表 334. 同步存储器缓冲区的用法 1526
表 335. 同步存储器缓冲区的用法 1527
表 336. 恢复事件检测 1528
表 337. 针对主机恢复事件检测 1529
表 338. 接收状态编码 1544
表 339. 端点/通道类型编码 1544
表 340. 端点/通道种类的意义 1545
表 341. 发送状态编码 1545
表 342. 已分配缓冲区存储器的定义 1547
表 343. USB 寄存器映射和复位值 1549
表 344. SPIRIT 地址块 DBG 1551
表 345. JTAG/串行线调试端口引脚 1552
表 346. 跟踪端口引脚 1552
表 347. 单线跟踪端口引脚 1553
表 348. 验证信号状态 1554
表 349. 生命周期状态和调试状态 1554
表 350. JTAG-DP 数据寄存器 1558
表 351. 数据包请求 1559
表 352. ACK 响应 1560
表 353. 数据传输 1560
表 354. 调试端口寄存器 1561
表 355. 调试端口寄存器映射和复位值 1567
表 356. MEM-AP寄存器 1569
表 357. 访问端口寄存器映射和复位值 1575
表 358. 系统 ROM 表 1576
表 359. MCU ROM 表 1577
表 360. 处理器 ROM 表 1577
表 361. 系统 ROM 表寄存器映射和复位值 1583

表 362. MCU ROM 表寄存器映射和复位值 1589
表 363. CPU ROM 表寄存器映射和复位值 1595
表 364. DWT 寄存器映射和复位值 1612
表 365. ITM 寄存器映射和复位值 1623
表 366. BPU 寄存器映射和复位值 1631
表 367. ETM 寄存器映射和复位值 1660
表 368. TPIU 寄存器映射和复位值 1675
表 369. CTI 输入 1678
表 370. CTI 输出 1678
表 371. CTI 寄存器映射和复位值 1690
表 372. 外设时钟冻结控制位 1693
表 373. 调试模式下的外设行为 1693
表 374. DBGMCU 寄存器映射和复位值 1708
表 375. 文档版本历史 1715

图片索引

图 1. 系统架构 . . . 66
图 2. 存储器映射 . . . 69
图 3. Flash 安全 HDP 区域 . . . 76
图 4. 器件生命周期安全性 . . . 83
图 5. 产品生命周期 . . . 85
图 6. 协作开发原理 . . . 86
图 7. 带 ECC 的 SRAM1 和 SRAM2 存储器映射 . . . 91
图 8. GTZC 特权架构 . . . 102
图 9. GTZC 框图 . . . 104
图 10. 水印存储器保护控制器（区域 x/子区域 A） . . . 105
图 11. MPCBB 框图 . . . 105
图 12. Flash 框图（简化版） . . . 115
图 13. 嵌入式 Flash 结构 . . . 117
图 14. 嵌入式 Flash 的使用 . . . 118
图 15. Flash 存储区交换顺序 . . . 129
图 16. 用户 Flash 中的 HDP . . . 138
图 17. 存储区交换情况下的保护属性图示 . . . 142
图 18. ICACHE 框图 . . . 185
图 19. ICACHE 标记和数据存储器功能视图 . . . 187
图 20. 具有 LDO 的电源概览 . . . 198
图 21. 系统电源配置 . . . 199
图 22. 上电复位 (POR)/掉电复位 (PDR) 波形 . . . 204
图 23. BOR 阈值 . . . 205
图 24. PVD 阈值 . . . 206
图 25. AVD 阈值 . . . 207
图 26. VBAT 阈值 . . . 208
图 27. 温度阈值 . . . 209
图 28. 运行模式下的动态电压调节 . . . 210
图 29. 待机模式下的 I/O 状态 . . . 218
图 30. 复位电路简化框图 . . . 236
图 31. 时钟树 . . . 239
图 32. HSE/LSE 时钟源 . . . 240
图 33. HSE/LSE 旁路 . . . 240
图 34. HSI 校准流程 . . . 242
图 35. CSI 校准流程 . . . 243
图 36. PLL 框图 . . . 244
图 37. PLL 初始化流程图 . . . 247
图 38. CRS 框图 . . . 319
图 39. CRS 计数器行为 . . . 320
图 40. 三伏或五伏容限 GPIO（TT 或 FT）的结构 . . . 330
图 41. 输入浮空/上拉/下拉配置 . . . 334
图 42. 输出配置 . . . 335
图 43. 复用功能配置 . . . 335
图 44. 高阻态模拟配置 . . . 336
图 45. SBS 框图 . . . 347
图 46. 补偿单元管理 . . . 348
图 47. 补偿单元使用 . . . 349
图 48. SBS 自举控制 . . . 349

图 49. SBS 调试控制 . . . 352
图 50. GPDMA 框图 . . . 384
图 51. 不使用链表的 GPDMA 通道直接编程 (GPDMA_CxLLR = 0) . . . 385
图 52. GPDMA 通道挂起和恢复序列 . . . 386
图 53. GPDMA 通道中止和重启序列 . . . 387
图 54. 线性寻址通道 x 的静态链表数据结构（所有 Uxx = 1） . . . 389
图 55. 2D 寻址通道 x 的静态链表数据结构（所有 Uxx = 1） . . . 390
图 56. 线性寻址通道 x 的 GPDMA 动态链表数据结 . . . 390
图 57. 2D 寻址通道 x 的 GPDMA 动态链表数据结构 . . . 391
图 58. 运行至完成模式 (GPDMA_CxCR.LSM = 0) 下的 GPDMA 通道执行和链表编程 . . . 393
图 59. 通过辅助 GPDMA 通道 y 插入 LLIn . . . 395
图 60. 链接步进模式 (GPDMA_CxCR.LSM = 1) 下的 GPDMA 通道执行和链表编程 . . . 397
图 61. 建立 LLIn+1：链接步进模式下的 GPDMA 动态链表. . . . 398
图 62. 在链接步进模式下使用寄存器文件中新的 LLIn' 进行替换. . . . 399
图 63. 在链接步进模式下使用存储器中新的 LLIn' 和 LLIn+1' 进行替换（选项 1） . . . 400
图 64. 在链接步进模式下使用存储器中新的 LLIn' 和 LLIn+1' 进行替换（选项 2） . . . 401
图 65. GPDMA 通道执行和链表编程 . . . 403
图 66. 编程设定的 2D 寻址 . . . 405
图 67. GPDMA 仲裁策略 . . . 412
图 68. 触发命中、记录和上溢波形. . . . 415
图 69. GPDMA 循环缓冲区编程：更新存储器起始地址（通过线性寻址通道） . . . 416
图 70. 共享 GPDMA 通道与循环缓冲：通过线性寻址通道更新存储器起始地址 . . . 417
图 71. EXTI 框图 . . . 453
图 72. 可配置事件触发逻辑 CPU 唤醒. . . . 456
图 73. EXTI 复用器 GPIO 选择 . . . 457
图 74. CRC 计算单元框图 . . . 479
图 75. ADC 框图 . . . 488
图 76. ADC 时钟方案. . . . 492
图 77. ADC1 连接功能. . . . 493
图 78. ADC 校准 . . . 496
图 79. 更新 ADC 校准系数. . . . 496
图 80. 混合单端通道和差分通道 . . . 497
图 81. 使能/禁止 ADC . . . 498
图 82. Bulb 模式时序图 . . . 501
图 83. 模数转换时间 . . . 504
图 84. 停止正在进行的常规转换 . . . 505
图 85. 停止正在进行的常规转换和注入转换 . . . 505
图 86. 触发选择 . . . 507
图 87. 注入转换延迟 . . . 508
图 88. JSQR 上下文队列示例（队列更改） . . . 511
图 89. JSQR 上下文队列示例（触发更改） . . . 511
图 90. 转换前发生溢出的 JSQR 上下文队列示例 . . . 512
图 91. 转换期间发生溢出的 JSQR 上下文队列示例 . . . 512
图 92. 队列为空时的 JSQR 上下文队列示例（JQM = 0 的情况） . . . 513
图 93. 队列为空时的 JSQR 上下文队列示例（JQM = 1 的情况） . . . 513
图 94. 通过将 JADSTP 置 1 的方式清空 JSQR 上下文队列 (JQM = 0)——
正在进行转换时发生 JADSTP. . . . 514
图 95. 通过将 JADSTP 置 1 的方式清空 JSQR 上下文队列 (JQM = 0)——
正在进行转换时发生 JADSTP 并出现新触发 . . . 514
图 96. 通过将 JADSTP 置 1 的方式清空 JSQR 上下文队列 (JQM = 0)——
在正在进行转换的范围之外发生 JADSTP. . . . 515
图 97. 通过将 JADSTP 置 1 的方式清空 JSQR 上下文队列 (JQM = 1). . . . 515

图 98. 通过将 ADDIS 置 1 的方式清空 JSQR 上下文队列 (JQM = 0) 516
图 99. 通过将 ADDIS 置 1 的方式清空 JSQR 上下文队列 (JQM = 1) 516
图 100. 单次序列转换，软件触发 518
图 101. 连续序列转换，软件触发 518
图 102. 单次序列转换，硬件触发 519
图 103. 连续序列转换，硬件触发 519
图 104. 右对齐（偏移禁止，无符号值） 521
图 105. 右对齐（偏移使能，有符号值） 521
图 106. 左对齐（偏移禁止，无符号值） 522
图 107. 左对齐（偏移使能，有符号值） 522
图 108. 溢出示例 (OVRMOD = 0) 523
图 109. 溢出示例 (OVRMOD = 1) 524
图 110. AUTODLY = 1，连续模式下的常规转换，软件触发 526
图 111. AUTODLY = 1，被注入转换中断的常规硬件转换（DISCEN = 0；JDISCEN = 0） 527
图 112. AUTODLY = 1，被注入转换中断的常规硬件转换（DISCEN = 1，JDISCEN = 1） 528
图 113. AUTODLY = 1，被注入转换中断的常规连续转换 529
图 114. AUTODLY = 1，自动注入模式 (JAUTO = 1) 529
图 115. 模拟看门狗的保护区域 530
图 116. ADCy_AWDx_OUT 信号生成（所有常规通道上） 532
图 117. ADCy_AWDx_OUT 信号生成（AWDx 标志未通过软件清零） 532
图 118. ADCy_AWDx_OUT 信号生成（单条常规通道上） 533
图 119. ADCy_AWDx_OUT 信号生成（所有注入通道上） 533
图 120. 20 位到 16 位结果的截断过程 534
图 121. 移 5 位并进行舍入的数值示例 534
图 122. 已触发的常规过采样模式（TROVS 位 = 1） 536
图 123. 常规过采样模式（4x 过采样率） 537
图 124. 同时使用常规和注入过采样模式 537
图 125. 已触发常规过采样支持注入 538
图 126. 在自动注入模式下进行过采样 538
图 127. 温度传感器通道框图 539
图 128. VBAT 通道框图 540
图 129. VREFINT 通道框图 541
图 130. 温度传感器功能框图 576
图 131. 低 REF_CLK 频率时的方法 578
图 132. 高 REF_CLK 频率时的方法 578
图 133. 温度传感器序列 581
图 134. 双通道 DAC 框图 593
图 135. DAC 单通道模式下的数据寄存器 596
图 136. DAC 双通道模式下的数据寄存器 596
图 137. 关闭触发 (TEN = 0) 时的转换时序图 598
图 138. DAC LFSR 寄存器计算算法 600
图 139. LFSR 生成波形的 DAC 转换（使能软件触发） 600
图 140. 生成 DAC 三角波 601
图 141. 生成三角波波形的 DAC 转换（使能软件触发） 601
图 142. DAC 采样和保持模式阶段图 603
图 143. 比较器功能框图 629
图 144. 比较器迟滞 631
图 145. 比较器输出消隐 632
图 146. 调节器框图 634
图 147. 独立模式：外部增益设置模式 642
图 148. 跟随器配置 643
图 149. PGA 模式，内部增益设置 (x2/x4/x8/x16)，未使用反相输入 644

图 150. PGA 模式，内部增益设置 (x2/x4/x8/x16)，用于滤波的反相输入 645
图 151. PGA 模式，同相增益设置 (x2/x4/x8/x16) 或反相增益设置 (x-1/x-3/x-7/x-15). 646
图 152. 配置示例 646
图 153. PGA 模式，同相增益设置 (x2/x4/x8/x16) 或使用滤波的反相增益设置 (x-1/x-3/x-7/x-15). . . 647
图 154. 配置示例 647
图 155. RNG 框图 655
图 156. NIST SP800-90B 熵源模型 656
图 157. RNG 初始化概述 658
图 158. HASH 框图 670
图 159. 消息数据交换功能 672
图 160. HASH 挂起/恢复机制 677
图 161. 高级控制定时器框图 690
图 162. 预分频器分频由 1 变为 2 时的计数器时序图. 696
图 163. 预分频器分频由 1 变为 4 时的计数器时序图. 696
图 164. 计数器时序图，1 分频内部时钟 697
图 165. 计数器时序图，2 分频内部时钟 698
图 166. 计数器时序图，4 分频内部时钟 698
图 167. 计数器时序图，N 分频内部时钟 699
图 168. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载） 699
图 169. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 预装载） 700
图 170. 计数器时序图，1 分频内部时钟 701
图 171. 计数器时序图，2 分频内部时钟 701
图 172. 计数器时序图，4 分频内部时钟 702
图 173. 计数器时序图，N 分频内部时钟 702
图 174. 计数器时序图，未使用重复计数器时更新事件 703
图 175. 计数器时序图，1 分频内部时钟，TIMx_ARR = 0x6 704
图 176. 计数器时序图，2 分频内部时钟 705
图 177. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36 705
图 178. 计数器时序图，N 分频内部时钟 706
图 179. 计数器时序图，ARPE=1 时的更新事件（计数器下溢） 706
图 180. 计数器时序图，ARPE=1 时的更新事件（计数器上溢） 707
图 181. 不同模式和 TIMx_RCR 寄存器设置下的更新频率示例 708
图 182. 外部触发输入模块 709
图 183. 正常模式下的控制电路，1 分频内部时钟 709
图 184. tim_ti2 外部时钟连接示例 710
图 185. 外部时钟模式 1 下的控制电路. 711
图 186. 外部触发输入模块 711
图 187. 外部时钟模式 2 下的控制电路. 712
图 188. 捕获/比较通道（示例：通道 1 输入阶段） 713
图 189. 捕获/比较通道 1 主电路 713
图 190. 捕获/比较通道的输出阶段（通道 1、通道 2、通道 3 和通道 4） 714
图 191. 捕获/比较通道的输出阶段（通道 5 和通道 6） 714
图 192. PWM 输入模式时序 716
图 193. 输出比较模式，翻转 tim_oc1 718
图 194. 边沿对齐模式的 PWM 波形 (ARR=8) 719
图 195. 中心对齐模式 PWM 波形 (ARR=8) 720
图 196. 抖动原理 721
图 197. 抖动模式下的数据格式和寄存器编码 722
图 198. PWM 分辨率与频率的关系 723
图 199. PWM 抖动模式 724
图 200. 中心对齐 PWM 模式下的抖动对占空比的影响 725
图 201. 50% 占空比时产生的 2 个相移 PWM 信号 726

图 202. 通道 1 和通道 3 上的组合 PWM 模式 728
图 203. 三相组合 PWM 信号（每个周期多个触发脉冲） 729
图 204. 带对称死区插入的互补输出 730
图 205. 非对称死区 730
图 206. 延迟时间大于负脉冲宽度的死区波形 731
图 207. 延迟时间大于正脉冲宽度的死区波形 731
图 208. 刹车事件和刹车 2 电路概述 733
图 209. 响应 tim_brk 上的刹车事件的不同输出行为 (OSSI = 1) 735
图 210. tim_brk 和 tim_brk2 使能后的 PWM 输出状态 (OSSI = 1) 736
图 211. tim_brk 使能后的 PWM 输出状态 (OSSI = 0) 736
图 212. 输出重定向（图中未显示 tim_brk2 请求） 738
图 213. tim_ocref_clr 输入选择复用器 739
图 214. 清除 TIMx 的 tim_ocxref 739
图 215. COM 事件生成 6 步 PWM 的示例 (OSSR=1) 740
图 216. 单脉冲模式示例 741
图 217. 可再触发单脉冲模式 742
图 218. 脉冲发生器电路 743
图 219. 发生比较事件时产生脉冲（边沿对齐模式和编码器模式） 744
图 220. 并发触发情况下的扩展脉宽 745
图 221. 编码器接口模式下的计数器工作示例 747
图 222. tim_ti1fp1 极性反相时的编码器接口模式示例 747
图 223. 正交编码器计数模式 748
图 224. 方向 + 时钟编码器模式 749
图 225. 定向时钟编码器模式（CC1P = CC2P = 0） 749
图 226. 定向时钟编码器模式（CC1P = CC2P = 1） 750
图 227. 索引门控选项 751
图 228. 抖动索引信号 751
图 229. IPOS[1:0] = 11 时的索引生成 752
图 230. 在通道 A 上对索引进行门控时的计数器读取 (IPOS[1:0] = 11) 752
图 231. 不对索引进行门控时的计数器读取 (IPOS[1:0] = 00) 753
图 232. 在通道 A 和 B 上对索引进行门控时的计数器读取 753
图 233. 窄索引脉冲情况下的编码器模式行为 (IPOS[1:0] = 11) 754
图 234. 计数器复位窄索引脉冲（放大视图，ARR = 0x07） 755
图 235. x1 和 x2 模式下的索引行为 (IPOS[1:0] = 01) 756
图 236. 定向索引有效性 756
图 237. 计数器复位与 FIDX 位置 1 的关系 757
图 238. 索引消隐 757
图 239. 时钟 + 方向模式下的索引行为 (IPOS[0] = 1) 758
图 240. 定向时钟模式下的索引行为 (IPOS[0] = 1) 758
图 241. 正交编码信号的状态图 759
图 242. 递增计数编码器错误检测 760
图 243. 递减计数编码器错误检测 761
图 244. 更新时传输预装载值情况下的编码器模式切换 (SMSPS = 0) 762
图 245. 测量 3 个信号上边沿之间的时间间隔 763
图 246. 霍尔传感器接口的示例 765
图 247. 复位模式下的控制电路 766
图 248. 门控模式下的控制电路 767
图 249. 触发模式下的控制电路 767
图 250. 外部时钟模式 2 + 触发模式下的控制电路 769
图 251. 通用定时器框图 820
图 252. 预分频器分频由 1 变为 2 时的计数器时序图 825
图 253. 预分频器分频由 1 变为 4 时的计数器时序图 825

图 254. 计数器时序图，1 分频内部时钟 826
图 255. 计数器时序图，2 分频内部时钟 827
图 256. 计数器时序图，4 分频内部时钟 827
图 257. 计数器时序图，N 分频内部时钟 828
图 258. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载） 828
图 259. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 已预装载） 829
图 260. 计数器时序图，1 分频内部时钟 830
图 261. 计数器时序图，2 分频内部时钟 830
图 262. 计数器时序图，4 分频内部时钟 831
图 263. 计数器时序图，N 分频内部时钟 831
图 264. 计数器时序图，更新事件 832
图 265. 计数器时序图，1 分频内部时钟，TIMx_ARR=0x6 833
图 266. 计数器时序图，2 分频内部时钟 834
图 267. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36 834
图 268. 计数器时序图，N 分频内部时钟 835
图 269. 计数器时序图，ARPE=1 时的更新事件（计数器下溢） 835
图 270. 计数器时序图，ARPE=1 时的更新事件（计数器上溢） 836
图 271. 正常模式下的控制电路，1 分频内部时钟 837
图 272. tim_ti2 外部时钟连接示例 837
图 273. 外部时钟模式 1 下的控制电路 838
图 274. 外部触发输入模块 838
图 275. 外部时钟模式 2 下的控制电路 839
图 276. 捕获/比较通道（示例：通道 1 输入阶段） 840
图 277. 捕获/比较通道 1 主电路 840
图 278. 捕获/比较通道的输出阶段（通道 1、通道 2、通道 3 和通道 4） 841
图 279. PWM 输入模式时序 843
图 280. 输出比较模式，翻转 tim_oc1 844
图 281. 边沿对齐模式的 PWM 波形 (ARR=8) 846
图 282. 中心对齐模式 PWM 波形 (ARR=8) 847
图 283. 抖动原理 848
图 284. 抖动模式下的数据格式和寄存器编码 849
图 285. PWM 分辨率与频率的关系（16 位模式） 850
图 286. PWM 分辨率与频率的关系（32 位模式） 850
图 287. PWM 抖动模式 851
图 288. 中心对齐 PWM 模式下的抖动对占空比的影响 852
图 289. 50% 占空比时产生的 2 个相移 PWM 信号 854
图 290. 通道 1 和通道 3 上的组合 PWM 模式 855
图 291. OCREF_CLR 输入选择复用器 856
图 292. 清除 TIMx 的 tim_ocxref 856
图 293. 单脉冲模式示例 857
图 294. 可再触发单脉冲模式 859
图 295. 脉冲发生器电路 859
图 296. 发生比较事件时产生脉冲（边沿对齐模式和编码器模式） 860
图 297. 并发触发情况下的扩展脉宽 861
图 298. 编码器接口模式下的计数器工作示例 863
图 299. tim_ti1fp1 极性反相时的编码器接口模式示例 863
图 300. 正交编码器计数模式 864
图 301. 方向 + 时钟编码器模式 865
图 302. 定向时钟编码器模式（CC1P = CC2P = 0） 865
图 303. 定向时钟编码器模式（CC1P = CC2P = 1） 866
图 304. 索引门控选项 867
图 305. 抖动索引信号 867

图 306. IPOS[1:0] = 11 时的索引生成 . . . 868
图 307. 在通道 A 上对索引进行门控时的计数器读取 (IPOS[1:0] = 11) . . . 868
图 308. 不对索引进行门控时的计数器读取 (IPOS[1:0] = 00) . . . 869
图 309. 在通道 A 和 B 上对索引进行门控时的计数器读取 . . . 869
图 310. 窄索引脉冲情况下的编码器模式行为 (IPOS[1:0] = 11) . . . 870
图 311. 计数器复位窄索引脉冲（放大视图，ARR = 0x07） . . . 871
图 312. x1 和 x2 模式下的索引行为 (IPOS[1:0] = 01) . . . 872
图 313. 定向索引有效性 . . . 872
图 314. 计数器复位与 FIDX 位置 1 的关系 . . . 873
图 315. 索引消隐 . . . 873
图 316. 时钟 + 方向模式下的索引行为 (IPOS[0] = 1) . . . 874
图 317. 定向时钟模式下的索引行为 (IPOS[0] = 1) . . . 874
图 318. 正交编码信号的状态图 . . . 875
图 319. 递增计数编码器错误检测 . . . 876
图 320. 递减计数编码器错误检测 . . . 877
图 321. 更新时传输预装载值情况下的编码器模式切换 (SMSPS = 0) . . . 878
图 322. 复位模式下的控制电路 . . . 880
图 323. 门控模式下的控制电路 . . . 880
图 324. 触发模式下的控制电路 . . . 881
图 325. 外部时钟模式 2 + 触发模式下的控制电路 . . . 882
图 326. 主/从定时器示例 . . . 883
图 327. 单通道定时器的主/从连接示例 . . . 883
图 328. 通过 TIM_mstr 的 tim_oc1ref 对 TIM_slv 进行门控 . . . 884
图 329. 通过 TIM_mstr 的使能信号对 TIM_slv 进行门控 . . . 885
图 330. 通过 TIM_mstr 的更新事件触发 TIM_slv . . . 886
图 331. 通过 TIM_mstr 的使能信号触发 TIM_slv . . . 886
图 332. 使用 TIM_mstr 的 tim_ti1 输入触发 TIM_mstr 和 TIM_slv . . . 887
图 333. 基本定时器框图 . . . 925
图 334. 正常模式下的控制电路，1 分频内部时钟 . . . 926
图 335. 预分频器分频由 1 变为 2 时的计数器时序图 . . . 927
图 336. 预分频器分频由 1 变为 4 时的计数器时序图 . . . 928
图 337. 计数器时序图，1 分频内部时钟 . . . 929
图 338. 计数器时序图，2 分频内部时钟 . . . 929
图 339. 计数器时序图，4 分频内部时钟 . . . 930
图 340. 计数器时序图，N 分频内部时钟 . . . 930
图 341. 计数器时序图，ARPE = 0 时更新事件（TIMx_ARR 未预装载） . . . 931
图 342. 计数器时序图，ARPE=1 时的更新事件（TIMx_ARR 已预装载） . . . 931
图 343. 抖动原理 . . . 932
图 344. 抖动模式下的数据格式和寄存器编码 . . . 933
图 345. FCnt 分辨率与频率的关系 . . . 934
图 346. PWM 抖动模式 . . . 934
图 347. LPTIM1/2 定时器框图(1) . . . 945
图 348. 干扰滤波器时序图 . . . 949
图 349. LPTIM 输出波形，单次计数模式配置（重复寄存器内容不等于零时）(PRELOAD = 1) . . . 950
图 350. LPTIM 输出波形，单次计数模式配置且激活置 1 一次模式（WAVE 位置 1） . . . 951
图 351. LPTIM 输出波形、连续计数模式配置 . . . 951
图 352. 生成波形 . . . 953
图 353. 编码器模式计数序列 . . . 956
图 354. 连续计数模式（重复寄存器 LPTIM_RCR 不等于零时）(PRELOAD = 1) . . . 956
图 355. 捕获/比较输入阶段（通道 1） . . . 957
图 356. 捕获/比较输出阶段（通道 1） . . . 958
图 357. 边沿对齐 PWM 模式 (PRELOAD = 1) . . . 960

图 358. 边沿对齐 PWM 波形（ARR = 8 且 CCxP = 0） 960
图 359. PWM 模式下的立即更新与预装载更新 961
图 360. 独立看门狗框图. 986
图 361. 由超时引起的复位时序 987
图 362. 由于在不允许刷新的区域中执行刷新而引起的复位时序 988
图 363. 窗口比较器更新示例 989
图 364. 独立看门狗中断时序图 991
图 365. 提前唤醒比较器更新示例 992
图 366. 看门狗框图 999
图 367. 窗口看门狗时序图. 1001
图 368. RTC 框图 1006
图 369. TAMP 框图 1049
图 370. 备份寄存器保护区. 1052
图 371. 主动入侵滤波 1057
图 372. I2C 框图 1086
图 373. I2C 总线协议. 1088
图 374. 建立和保持时序. 1090
图 375. I2C 初始化流程 1092
图 376. 数据接收 1093
图 377. 数据发送 1094
图 378. 从器件初始化流程 1097
图 379. I2C 从发送器的传输序列流程 (NOSTRETCH = 0) 1098
图 380. I2C 从发送器的传输序列流程 (NOSTRETCH = 1) 1099
图 381. I2C 从发送器的传输总线图（仅限强制性事件） 1100
图 382. 从接收器的传输序列流程 (NOSTRETCH = 0) 1101
图 383. 从接收器的传输序列流程 (NOSTRETCH = 1) 1102
图 384. I2C 从接收器的传输总线图（仅限强制性事件） 1102
图 385. 主时钟生成 1104
图 386. 主器件初始化流程 1106
图 387. 10 位地址读访问 (HEAD10R = 0). 1106
图 388. 10 位地址读访问 (HEAD10R = 1). 1106
图 389. I2C 主发送器的传输序列流程（N ≤ 255 字节） 1108
图 390. I2C 主发送器的传输序列流程（N > 255 字节） 1109
图 391. I2C 主发送器的传输总线图（仅限强制性事件） 1110
图 392. I2C 主接收器的传输序列流程（N ≤ 255 字节） 1112
图 393. I2C 主接收器的传输序列流程（N > 255 字节） 1113
图 394. I2C 主接收器的传输总线图（仅限强制性事件） 1114
图 395. $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的超时间隔. 1118
图 396. SMBus 从发送器的传输序列流程（N 字节 + PEC） 1121
图 397. SMBus 从发送器的传输总线图 (SBC = 1) 1122
图 398. SMBus 从接收器的传输序列流程（N 字节 + PEC） 1123
图 399. SMBus 从接收器的总线传输图 (SBC = 1) 1124
图 400. SMBus 主发送器的总线传输图 1125
图 401. SMBus 主接收器的总线传输图 1126
图 402. I3C 框图 1151
图 403. I3C（主）控制器状态和编程序列图 1155
图 404. I3C 目标状态和编程序列图 1159
图 405. I3C CCC 消息（作为控制器） 1172
图 406. I3C 广播 ENTDAA CCC（作为控制器） 1173
图 407. I3C 广播、直接读/写 RSTACT CCC（作为控制器） 1174
图 408. I3C CCC 消息（作为目标） 1176
图 409. I3C 广播 ENTDAA CCC（作为目标） 1177

图 410. I3C 广播 DEFTGTS CCC（作为目标） 1178
图 411. I3C 广播 DEFGRPA CCC（作为目标） 1179
图 412. I3C 私有读/写消息（作为控制器） 1181
图 413. I3C 私有读/写消息（作为目标） 1182
图 414. 传统 I2C 读/写消息（作为控制器） 1183
图 415. IBI 传输（作为控制器/目标） 1184
图 416. 热加入请求传输（作为控制器/目标） 1185
图 417. 控制器角色请求传输（作为控制器/目标） 1186
图 418. C-FIFO 管理（作为控制器） 1187
图 419. TX-FIFO 管理（作为控制器） 1189
图 420. RX-FIFO 管理（作为控制器） 1191
图 421. S-FIFO 管理（作为控制器） 1193
图 422. RX-FIFO 管理（作为目标） 1195
图 423. 通过 I3C_TGTTDR 管理 TX-FIFO（作为目标） 1197
图 424. 如果读取的字节数小于 TX-FIFO 大小，则不使用 I3C_TGTTDR
而通过软件管理 TX-FIFO（作为目标） 1199
图 425. USART 框图 1254
图 426. 字长编程 1258
图 427. 可配置的停止位 1260
图 428. 发送时的 TC/TXE 行为 1262
图 429. 16 倍或 8 倍过采样时的起始位检测 1263
图 430. usart_ker_ck 时钟分频器框图 1266
图 431. 16 倍过采样时的数据采样 1267
图 432. 8 倍过采样时的数据采样 1268
图 433. 使用空闲线路检测时的静默模式 1274
图 434. 使用地址标记检测时的静默模式 1275
图 435. LIN 模式下的刹车检测（11 位刹车长度——LBDL 位置 1） 1278
图 436. LIN 模式下的刹车检测与帧错误检测 1279
图 437. USART 同步主发送示例 1280
图 438. USART 在同步主模式下的数据时钟时序图（M 位 = 00） 1280
图 439. USART 在同步主模式下的数据时钟时序图（M 位 = 01） 1281
图 440. USART 在同步从模式下的数据时钟时序图（M 位 = 00） 1282
图 441. ISO 7816-3 异步协议 1284
图 442. 使用 1.5 个停止位检测奇偶校验错误 1285
图 443. IrDA SIR ENDEC 框图 1289
图 444. IrDA 数据调制 (3/16)——正常模式 1289
图 445. 使用 DMA 进行发送 1291
图 446. 使用 DMA 进行接收 1292
图 447. 2 个 USART 间的硬件流控制 1292
图 448. RS232 RTS 流控制 1293
图 449. RS232 CTS 流控制 1294
图 450. 唤醒事件通过验证（唤醒事件 = 地址匹配，禁止 FIFO） 1296
图 451. 唤醒事件未通过验证（唤醒事件 = 地址匹配，禁止 FIFO） 1297
图 452. LPUART 框图 1334
图 453. LPUART 字长编程 1337
图 454. 可配置的停止位 1339
图 455. 发送时的 TC/TXE 行为 1340
图 456. lpuart_ker_ck 时钟分频器框图 1343
图 457. 使用空闲线路检测时的静默模式 1347
图 458. 使用地址标记检测时的静默模式 1348
图 459. 使用 DMA 进行发送 1351
图 460. 使用 DMA 进行接收 1352

图 461. 2 个 LPUART 间的硬件流控制 1353
图 462. RS232 RTS 流控制 1353
图 463. RS232 CTS 流控制 1354
图 464. 唤醒事件通过验证（唤醒事件 = 地址匹配，禁止 FIFO） 1356
图 465. 唤醒事件未通过验证（唤醒事件 = 地址匹配，禁止 FIFO） 1357
图 466. SPI/I2S 框图 1384
图 467. 全双工单个主器件/单个从器件应用 1387
图 468. 半双工单个主器件/单个从器件应用 1387
图 469. 单工单个主器件/单个从器件应用（主器件处于只发送模式/从器件处于只接收模式） . . 1388
图 470. 使用星形拓扑连接的主器件和三个独立从器件 1389
图 471. 一主三从环形连接（菊花链）拓扑 1390
图 472. 多主器件应用 1391
图 473. SS 控制逻辑原理图 1393
图 474. 数据流时序控制（SSOE = 1，SSOM = 0，SSM = 0） 1393
图 475. 数据之间的 SS 交错脉冲（SSOE = 1，SSOM = 1，SSM = 0） 1394
图 476. 数据时钟时序图 1396
图 477. 数据大小不等于 8 位、16 位或 32 位时的数据对齐 1397
图 478. 将数据封装在 FIFO 中，以便以全面功能集实例进行发送和接收 1404
图 479. TI 模式传输 1405
图 480. 从器件检测下溢情况的可选配置 1407
图 481. 波形示例 1414
图 482. 主模式 I2S Philips 协议波形（16/32 位全精度） 1414
图 483. I2S Philips 标准波形 1415
图 484. 主模式 MSB 对齐的 16 位或 32 位全精度长度 1415
图 485. 主模式 MSB 对齐的 16 位或 24 位数据长度 1416
图 486. 从模式 MSB 对齐 1416
图 487. LSB 对齐的 16 位或 24 位数据长度 1417
图 488. 帧长度等于数据长度时的主模式 PCM 1418
图 489. 主模式 PCM 标准波形（16 位或 24 位数据长度） 1418
图 490. 从模式 PCM 波形 1419
图 491. 启动序列，I2S Philips 标准，主模式 1422
图 492. 启动序列，I2S Philips 标准，从模式 1422
图 493. 停止序列，I2S Philips 标准，主模式 1423
图 494. I^2S 时钟发生器架构 1423
图 495. 数据格式 1426
图 496. 下溢情况的处理 1427
图 497. 上溢情况的处理 1428
图 498. 帧错误检测，FIXCH = 0 1429
图 499. 帧错误检测，FIXCH = 1 1429
图 500. CAN 子系统 1453
图 501. FDCAN 框图 1455
图 502. 位时序 1456
图 503. 收发器延迟测量 1460
图 504. 总线监控模式下的引脚控制 1461
图 505. 环回模式下的引脚控制 1463
图 506. 消息 RAM 配置 1464
图 507. 标准消息 ID 过滤器路径 1467
图 508. 扩展消息 ID 过滤器路径 1468
图 509. USB 外设框图 1514
图 510. 数据包缓冲区与缓冲区描述表存储单元示例 1519
图 511. 调试支持基础结构框图 1552
图 512. 产品生命周期状态和调试身份验证 1555

图 513. JTAG TAP 状态机 1557
图 514. CoreSight 拓扑 1578
图 515. 跟踪端口接口单元 (TPIU) 1664
图 516. 嵌入式交叉触发 1677

1 文档约定

1.1 概述

STM32H503xx 器件搭载 Arm®[a] Cortex-M33 内核。

1.2 寄存器相关缩写词列表

寄存器说明中使用以下缩写词[b]：

读/写 (rw)	软件可以读写该位。
只读 (r)	软件只能读取该位。
只写 (w)	软件只能写入该位。读取该位时将返回复位值。
读取/写入 0 清零 (rc_w0)	软件可以通过读取该位或通过写入 0 将该位清零。写入 1 对该位的值无影响。
读取/写入 1 清零 (rc_w1)	软件可以通过读取该位或通过写入 1 将该位清零。写入 0 对该位的值无影响。
读取/写入清零 (rc_w)	软件可以通过读取该位或通过写入寄存器将该位清零。写入该位的值并不重要。
读取/读取清零 (rc_r)	软件可以读取该位。读取该位时，将自动清零。写入该位对其值无影响。
读取/读取置 1 (rs_r)	软件可以读取该位。读取该位时，将自动置 1。写入该位对其值无影响。
读取/置位 (rs)	软件可以读取该位，也可将其置 1。写入 0 对该位的值无影响。
读/仅可写入一次 (rwo)	软件仅可写入一次该位，但可随时读取该位。只能通过复位将该位返回到复位值。
切换 (t)	软件可以通过写入 1 来切换该位。写入 0 无影响。
只读，写触发 (rt_w1)	软件可以读取该位。写入 1 时，将触发事件，但不会影响该位的值。
保留 (Res.)	保留位，必须保持复位值。

a. Arm 是 Arm Limited（或其子公司）在美国和/或其他地区的注册商标。

b. 这是一个涵盖适用于意法半导体微控制器的所有缩写词的详尽列表，其中一些缩写词在当前文档中可能并未使用。

1.3 词汇表

本节简要介绍本文档中所用首字母缩略词和缩写词的定义：

- **字**：32 位数据。
- **半字**：16 位数据。
- **字节**：8 位数据。
- **AHB**：高级高性能总线。

2 存储器和总线架构

2.1 系统架构

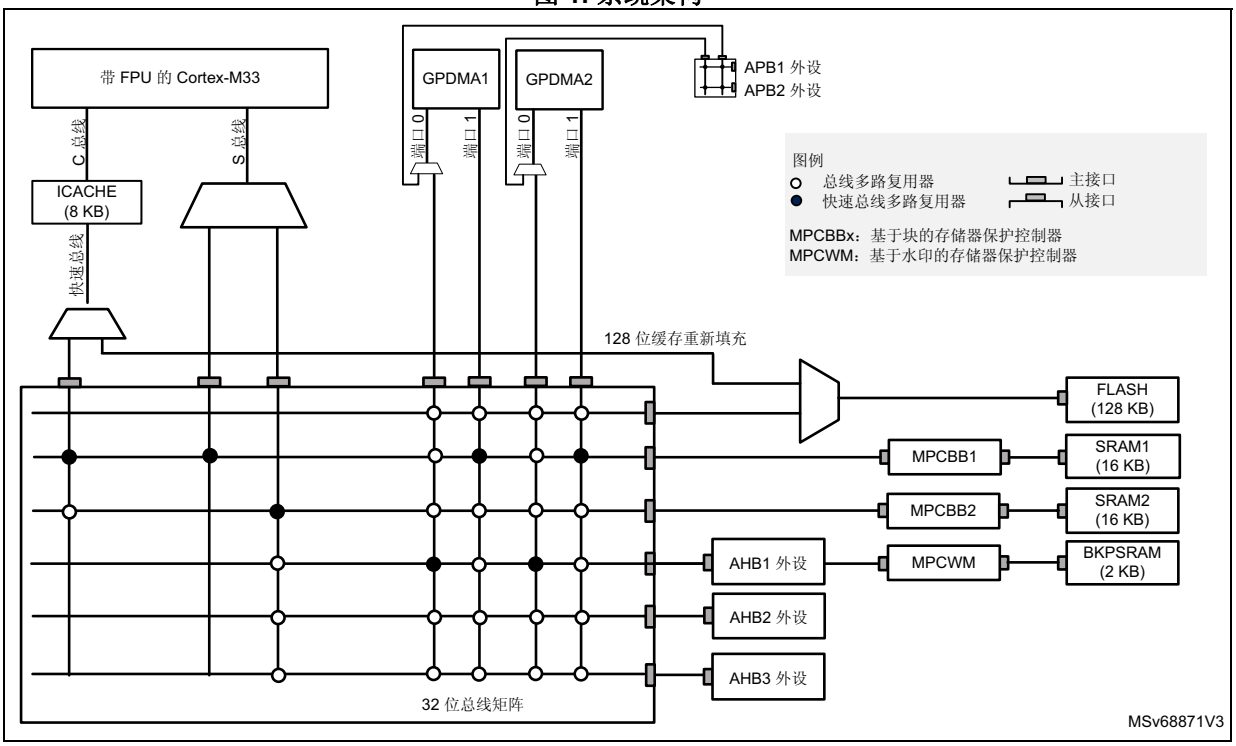
STM32H503xx 架构采用 Arm Cortex-M33 内核，得益于指令缓存可以直接访问嵌入式 Flash，因此在执行速度方面得到了优化。

此架构还采用 32 位多层 AHB 总线矩阵，可实现以下部分的互连：

- 最多 7 条主控总线：
  - 快速 C 总线（可通过指令缓存将带有 FPU 的 Cortex-M33 内核 C 总线连接到内部 SRAM）
  - 带 FPU 的 Cortex-M33 内核 S 总线（可将两条主控总线连接到两个内部 SRAM（无延迟））
  - GPDMA1（具有两个主端口的通用 DMA）
  - GPDMA2（具有两个主端口的通用 DMA）
- 最多 6 条被控总线：
  - 内部 Flash (128 KB)
  - 内部 SRAM1 (16 KB)
  - 内部 SRAM2 (16 KB)
  - AHB1 外设和备份 RAM (2 KB BKPSRAM)（包括 AHB-APB 总线桥和 APB 外设（连接到 APB1 和 APB2））
  - AHB2 外设
  - AHB3 外设（包括 AHB-APB 总线桥和 APB 外设（连接到 APB3））

借助总线矩阵，可以实现主控总线到被控总线的访问，这样即使在多个高速外设同时运行期间，系统也可以实现并发访问和高效运行。此架构如下图所示。

图 1. 系统架构



2.1.1 快速 C 总线

此总线用于将 Cortex-M33 内核的 C 总线连接到内部 Flash 和总线矩阵（通过指令缓存）。使用此总线，可以对映射到代码区域的内部存储器进行取指和数据访问。此总线访问的对象是内部 Flash 和内部 SRAM（SRAM1 和 SRAM2）。

此总线通过连续映射来访问 SRAM1 和 SRAM2。

2.1.2 S 总线

此总线用于将 Cortex-M33 内核的系统总线连接到总线矩阵。内核使用此总线访问位于外设或 SRAM 区域中的数据。此总线访问的对象是内部 SRAM（SRAM1、SRAM2 和 BKPSRAM）、AHB1 外设（包括 APB1 和 APB2）、AHB2 外设和 AHB3 外设。

此总线通过连续映射来访问 SRAM1 和 SRAM2。

注意： *总线矩阵访问 SRAM1 和 SRAM2 时无任何延迟。*

2.1.3 GPDMA1 和 GPDMA2 总线

这两条总线用于将 GPDMA1 和 GPDMA2 的四个 AHB 主接口连接到总线矩阵。这两条总线访问的对象是内部 Flash、内部 SRAM（SRAM1、SRAM2 和 BKPSRAM）、AHB1 外设（包括 APB1 和 APB2）、AHB2 外设和 AHB3 外设。

2.1.4 总线矩阵

总线矩阵用于主控总线之间的访问仲裁管理。仲裁采用循环调度算法。此总线矩阵具有一个快速总线复用器，可以将各主控总线连接到给定的被控总线，而没有任何延迟（请参见*图 1*）。对于同一条主控总线，每次执行新的访问时，其他被控总线都需要延迟至少一个周期。

2.1.5 AHB/APB 总线桥

借助三个 AHB/APB 总线桥，可在 AHB 与 APB 总线之间实现完全同步的连接，从而灵活选择外设频率。

有关连接到这些总线桥的外设的地址映射，请参见*第 2.2.2 节：存储器映射和寄存器边界地址*。

每次芯片复位后，所有外设时钟都将关闭（内部 SRAM 和 Flash 接口除外）。使用外设之前，必须在 RCC_AHBxENR 和 RCC_APBxENR 寄存器中使能其时钟。

注意： *对 APB 寄存器执行 8 位或 16 位访问时，该访问将转换为 32 位访问：总线桥将 8 位或 16 位数据复制后提供给 32 位向量。*

2.2 存储器结构

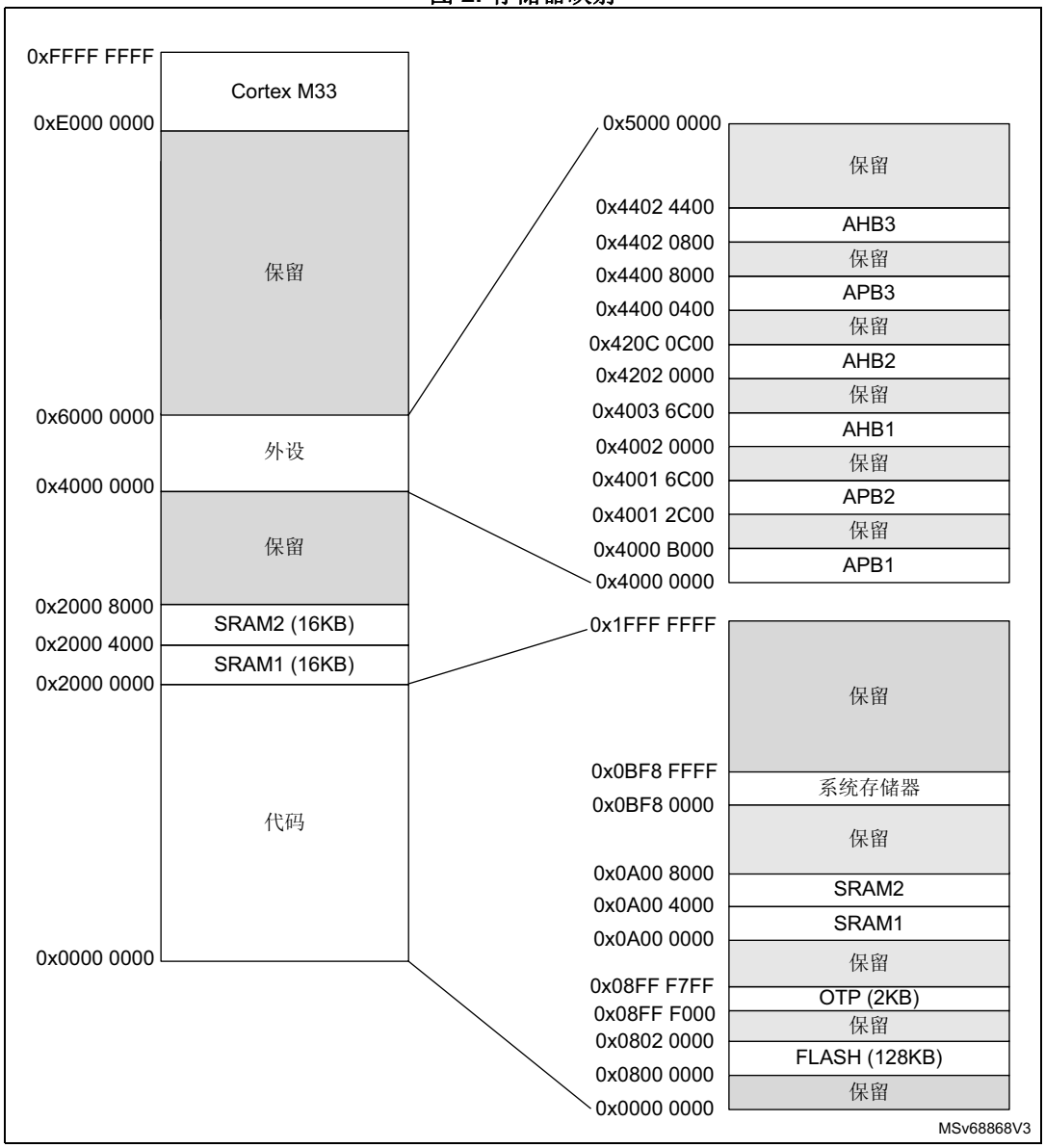
2.2.1 简介

程序存储器、数据存储器、寄存器和 I/O 端口排列在同一个线性（即地址连续）的 4 GB 地址空间内。

各字节按小端格式在存储器中编码。一个存储字单元中编号最低的字节被视为该字的最低有效字节，而编号最高的字节被视为最高有效字节。

2.2.2 存储器映射和寄存器边界地址

图 2. 存储器映射



未分配给片上存储器和外设的所有存储器映射区域均视为“保留区”。下表给出了器件中可用外设的边界地址。

表 1. 存储器映射和外设寄存器边界地址

总线	边界地址	外设	外设寄存器映射
AHB3	0x4402 4400 - 0x44FF FFFF	保留	-
	0x4402 4000 - 0x4402 43FF	DEBUG	DEBUG 寄存器映射
	0x4402 3000 - 0x4402 3FFF	保留	-
	0x4402 2000 - 0x4402 23FF	EXTI	EXTI 寄存器映射
	0x4402 1000 - 0x4402 1FFF	保留	-
	0x4402 0C00 - 0x4402 0FFF	RCC	RCC 寄存器映射
	0x4402 0800 - 0x4402 0BFF	PWR	PWR 寄存器映射
	0x4402 0000 - 0x4402 07FF	保留	-
APB3	0x4400 8000 - 0x4400 FFFF	保留	-
	0x4400 7C00 - 0x4400 7FFF	TAMP	TAMP 寄存器映射
	0x4400 7800 - 0x4400 7BFF	RTC	RTC 寄存器映射
	0x4400 4800 - 0x4400 77FF	保留	-
	0x4400 4400 - 0x4400 47FF	LPTIM1	LPTIM1 寄存器映射
	0x4400 3400 - 0x4400 43FF	保留	-
	0x4400 3000 - 0x4400 33FF	I3C2	I3C2 寄存器映射
	0x4400 2800 - 0x4400 2FFF	保留	-
	0x4400 2400 - 0x4400 27FF	LPUART1	LPUART 寄存器映射
	0x4400 0800 - 0x4400 23FF	保留	-
	0x4400 0400 - 0x4400 07FF	SBS	SBS 寄存器映射
	0x4400 0000 - 0x4400 03FF	保留	-

表 1. 存储器映射和外设寄存器边界地址（续）

总线	边界地址	外设	外设寄存器映射
AHB2	0x420C 0C00 - 0x43FF FFFF	保留	-
	0x420C 0800 - 0x420C 0BFF	RNG	RNG 寄存器映射
	0x420C 0400 - 0x420C 07FF	HASH	HASH 寄存器映射
	0x4202 8800 - 0x420C 03FF	保留	-
	0x4202 8400 - 0x4202 87FF	DAC1	DAC 寄存器映射
	0x4202 8000 - 0x4202 83FF	ADC1	ADC 寄存器映射
	0x4202 2000 - 0x4202 7FFF	保留	-
	0x4202 1C00 - 0x4202 1FFF	GPIOH	GPIO 寄存器映射
	0x4202 1000 - 0x4202 1BFF	保留	-
	0x4202 0C00 - 0x4202 0FFF	GPIOD	GPIO 寄存器映射
	0x4202 0800 - 0x4202 0BFF	GPIOC	
	0x4202 0400 - 0x4202 07FF	GPIOB	
	0x4202 0000 - 0x4202 03FF	GPIOA	
AHB1	0x4003 6C00 - 0x41FF FFFF	保留	-
	0x4003 6400 - 0x4003 6BFF	BKPSRAM	-
	0x4003 3400 - 0x4003 63FF	保留	-
	0x4003 3000 - 0x4003 33FF	GTZC1_MPCBB2	GTZC1 MPCBBz 寄存器（z = 1 和 2）
	0x4003 2C00 - 0x4003 2FFF	GTZC1_MPCBB1	GTZC1 MPCBBz 寄存器（z = 1 和 2）
	0x4003 2800 - 0x4003 2BFF	保留	-
	0x4003 2400 - 0x4003 27FF	GTZC1_TZSC	GTZC1 TZSC 寄存器映射
	0x4003 0800 - 0x4003 23FF	保留	-
	0x4003 0400 - 0x4003 07FF	ICACHE	ICACHE 寄存器映射
	0x4002 7000 - 0x4003 03FF	保留	-
	0x4002 6000 - 0x4002 6FFF	RAMCFG	RAMCFG 寄存器映射
	0x4002 3400 - 0x4002 5FFF	保留	-
	0x4002 3000 - 0x4002 33FF	CRC	CRC 寄存器映射
	0x4002 2400 - 0x4002 2FFF	保留	-
	0x4002 2000 - 0x4002 23FF	FLASH	FLASH 寄存器映射
	0X4002 1000 - 0x4002 1FFF	GPDMA2	GPDMA 寄存器映射
	0x4002 0000 - 0x4002 0FFF	GPDMA1	

表 1. 存储器映射和外设寄存器边界地址（续）

总线	边界地址	外设	外设寄存器映射
APB2	0x4001 6C00 - 0x4001 FFFF	保留	-
	0x4001 6400 - 0x4001 6BFF	USB RAM	USB 寄存器映射
	0x4001 6000 - 0x4001 63FF	USB	
	0x4001 3C00 - 0x4001 5FFF	保留	-
	0x4001 3800 - 0x4001 3BFF	USART1	USART 寄存器映射
	0x4001 3400 - 0x4001 37FF	保留	-
	0x4001 3000 - 0x4001 33FF	SPI1/I2S1	SPI 寄存器映射
	0x4001 2C00 - 0x4001 2FFF	TIM1	TIMx 寄存器映射
	0x4001 0000 - 0x4001 2BFF	保留	-

表 1. 存储器映射和外设寄存器边界地址（续）

总线	边界地址	外设	外设寄存器映射
APB1	0x4000 B000 - 0x4000 FFFF	保留	-
	0x4000 AC00 - 0x4000 AFFF	FDCAN SRAM	FDCAN 寄存器映射
	0x4000 A800 - 0x4000 ABFF	保留	
	0x4000 A400 - 0x4000 A7FF	FDCAN1	FDCAN 寄存器映射
	0x4000 9800 - 0x4000 A3FF	保留	-
	0x4000 9400 - 0x4000 97FF	LPTIM2	LPTIM 寄存器映射
	0x4000 9000 - 0x4000 93FF	保留	-
	0x4000 8C00 - 0x4000 8FFF	DTS	DTS 寄存器映射
	0x4000 6400 - 0x4000 8BFF	保留	-
	0x4000 6000 - 0x4000 63FF	CRS	CRS 寄存器映射
	0x4000 5C00 - 0x4000 5FFF	I3C1	I3C 寄存器映射
	0x4000 5800 - 0x4000 5BFF	I2C2	I2C 寄存器映射
	0x4000 5400 - 0x4000 57FF	I2C1	
	0x4000 4C00 - 0x4000 53FF	保留	-
	0x4000 4800 - 0x4000 4BFF	USART3	USART 寄存器映射
	0x4000 4400 - 0x4000 47FF	USART2	
	0x4000 4000 - 0x4000 43FF	COMP	COMP 寄存器映射
	0x4000 3C00 - 0x4000 3FFF	SPI3/I2S3	SPI 寄存器映射
	0x4000 3800 - 0x4000 3BFF	SPI2/I2S2	
	0x4000 3400 - 0x4000 37FF	OPAMP	OPAMP 寄存器映射
	0x4000 3000 - 0x4000 33FF	IWDG	IWDG 硬件配置寄存器 (IWDG_HWCFGR)
	0x4000 2C00 - 0x4000 2FFF	WWDG	WWDG 寄存器映射
	0x4000 1800 - 0x4000 2BFF	保留	-
	0x4000 1400 - 0x4000 17FF	TIM7	TIM6/TIM7 寄存器映射
	0x4000 1000 - 0x4000 13FF	TIM6	
	0x4000 0800 - 0x4000 0FFF	保留	-
	0x4000 0400 - 0x4000 07FF	TIM3	TIM2/TIM3 寄存器
	0x4000 0000 - 0x4000 03FF	TIM2	

2.2.3 嵌入式 SRAM

器件的 SRAM 高达 32 KB：

- 16 KB SRAM1
- 16 KB SRAM2
- 2 KB BKPSRAM

这些 SRAM 可按字节、半字（16 位）或全字（32 位）访问。这些存储器可通过 CPU 和 DMA 寻址。

CPU 可通过系统总线或 C 总线访问 SRAM1 和 SRAM2，具体取决于所选地址。CPU 只能通过系统总线访问 BKPSRAM。

有关 SRAM 特性的详细信息，请参见*第 5.3.1 节：内部 SRAM 特性*。

2.2.4 Flash 概述

Flash 由两个不同的物理区域组成：

- 主 Flash 块，包含应用程序和用户数据。
- 信息块，由以下部分组成：
  - 用于硬件和存储器保护用户配置的选项字节
  - 包含 ST 专有代码的系统存储器
  - OTP（一次性可编程）区域

Flash 接口根据 AHB 协议实施指令访问和数据访问。此接口还可执行通过 Flash 寄存器进行控制的 Flash 操作（编程/擦除）所必需的逻辑，并实现了安全访问控制功能。更多详细信息，请参见*第 7 章：嵌入式 Flash (FLASH)*。

3 系统安全性

STM32H503xx 具有一套全面的安全特性。

这些安全特性可以简化 IoT 设备的安全标准评估，而且能够促进重复使用、提高互操作性并最大限度地减少 API 碎片化，从而显著降低 OEM 和第三方开发人员的软件开发成本和复杂性。

本章将介绍 STM32H503xx 器件提供的各种安全特性。

3.1 主要安全特性

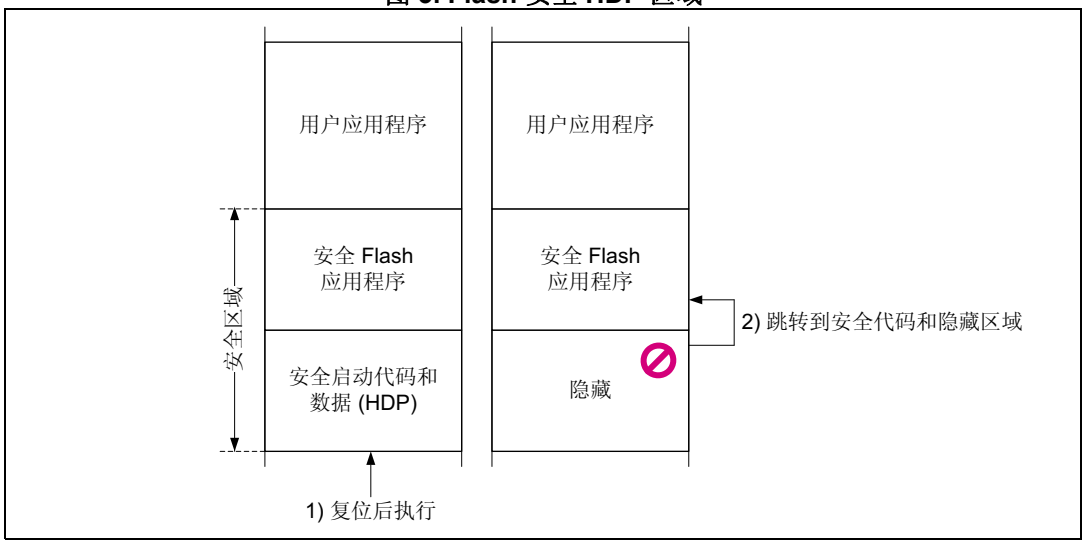
- 使用 Cortex-M33 的特权模式实现资源隔离（扩展到特权、存储器和外设）
- 安全调试控制（回退控制）。借助配置工具，OEM 可以在 OTP 过程中对安全调试控制进行个性化设置
- 时域隔离：使用 HDPL（隐藏保护级别）单调计数器隔离自举级别
- 由电池供电的易失性安全存储（发生入侵时自动擦除）
- 器件 96 位唯一 ID
- HASH 处理器，支持 SHA-1 校验和与 SHA-2 安全散列（SHA2-224 和 SHA2-256）
- 真随机数发生器 (RNG)，通过 NIST SP800-90B 预认证
- 全新的灵活生命周期方案，支持现场产品维护
- 主动入侵检测和保护（针对温度、电压和频率攻击）
  - 不同功耗模式下均支持主动入侵检测（请参见 TAMP 低功耗模式）

3.2 资源隔离

3.2.1 使用安全隐藏保护 (HDP) 实现时域隔离

嵌入式 Flash 允许为每个存储区的水印安全区域定义一个 HDP 区域（8 KB 扇区粒度）。在此 HDP 区域中执行的代码及其相关数据和密钥可在自举后隐藏，直到下一次系统复位。隐藏保护原理如下图所示。

图 3. Flash 安全 HDP 区域



用户 Flash 中 HDP 区域的激活状态与 HDPL1 有关。这意味着只要 HDPL 等于或大于 2，便会拒绝数据读取、写入和取指操作（由 FLASH_HDPxR_PRG 选项字节中 HDPx_PSTRT 和 HDPx_PEND 定义的区域），直到下一次器件复位。

借助 FLASH_HDPEXTR Flash 寄存器，可以对 HDPx 区域的 END 进行扩展（由应用动态扩展）。此扩展可以阻止访问 HDPL3 下的受保护扇区。

注意： *当存储区内包含写保护区域（WRP 或 HDP 区域）时，存储区擦除操作会中止。*

在 STM32H503xx 器件中，可以对用于自举的硬件资源和软件资源进行隔离。这称为时域隔离。

它以一个只能递增级别的单调计数器为基础。当计数器递增时，将隐藏上一级资源（代码和数据）。更多信息，请参见*第 7 章：嵌入式 Flash (FLASH)*。

3.2.2 使用 Cortex 特权模式实现资源隔离

可以将 STM32H503xx 器件的硬件资源与软件资源隔离，从而将其限制在以 Cortex 特权模式运行的软件中。

得益于此硬件隔离技术，可以对关键代码或数据实施保护，以防止有意或无意地入侵更多公开的非特权代码。

使用 MPU 为存储器和外设分配特权

Cortex-M33 MPU 将统一的存储器划分为八个区域，每个区域的大小均为 32 字节的倍数。可以对每个存储区域进行编程，以便在遭受非特权软件的不当访问时生成故障。

3.2.3 使用 GTZC 特权控制实现资源隔离

在 STM32H503 器件中，可以将硬件资源与软件资源隔离，以便二者能够同时存在于特权环境或非特权环境中。请参见*第 6 章：全局特权控制器 (GTZC)*。

使用 GTZC 为存储器和外设分配特权

对于 Cortex-M33 主器件，GTZC 可以针对外设和所选存储器，灵活地增强特权任务与非特权任务之间的隔离，从而对 MPU 的粗略隔离效果加以补充。

对于 Cortex-M33 之外的主器件，GTZC 可以将其分配为非特权发起者，从而对定义为此主器件的特权资源实施自动保护。

- 使用 TZSC 为外设分配特权（仅特权）

  器件中的外设要么通过 GTZC 配置为仅特权，要么本身为特权感知：

  – 仅特权外设或存储器受 AHB/APB 防火墙门保护，此防火墙门由 TZSC 控制。

  – 特权感知外设或存储器直接连接到 AHB 或 APB 互连，从而实现特定行为，例如一个寄存器子集或一个存储区域为仅特权。

    当此类外设通过 GTZC 配置为仅特权时，如果其为互连 (GPDMA) 上的主器件，则会自动发出特权事务。特权感知主器件（如 GPDMA1 和 GPDMA2）根据其内部特权模式来驱动 AHB 互连中的特权信号，而与 GTZC 无关。

    有关特权感知外设和仅特权外设的列表，请参见*第 6 章：全局特权控制器 (GTZC)*。

- 使用 TZSC 和 MPCBB 为存储器分配特权（仅特权）

  GTZC 中的 TZSC 逻辑可以管理 BKPSRAM 的特权级别，即对*第 3.2.2 节*中定义的 MPCWM 资源进行编程。

  同理，GTZC 中的 TZSC 逻辑也可以配置嵌入式 SRAM 块的权限级别，即对*第 3.2.2 节*中定义的 MPCBB 资源进行编程。

- 错误管理（仅特权）

  – 任何尝试访问特权资源的非特权事务都被视为非法。对于非法的非特权读写访问，不会生成非法访问事件。

  – 寻址的资源遵循“静默 - 失败”行为，读访问返回全零数据，所有写访问都会被忽略。

  – 发生非法的非特权访问时，不会生成总线错误，除非此访问是取指操作，并且访问的是特权存储器或外设寄存器。

管理特权感知外设的特权

特权感知外设也可以实现仅特权访问模式：

- *嵌入式 Flash*

  默认情况下，所有嵌入式 Flash 寄存器都可以在特权模式和非特权模式下进行读取或编程。

  当 FLASH_PRIVCFGR 中的特权位 NSPRIV 置 1 时，只能在特权模式下读取和写入 Flash 寄存器。

  关于嵌入式 Flash 的特权保护，器件具有以下特性：

  – 系统 Flash 在特权模式和非特权模式下均可访问。

  – 每个 HDP 区域在特权模式和非特权模式下均可访问（如适用）。

  – 使用基于块的特权配置寄存器 FLASH_PRIVBB1R1 和 FLASH_PRIVBB2R1，可以实时地将嵌入式 Flash 的每个 8 KB 扇区都编程为仅特权。非特权扇区在特权模式和非特权模式下均可访问。

注意： *将扇区从特权属性切换到非特权属性不会擦除扇区的内容。*

如果适用，擦除或编程操作始终可用于特权代码，但仅可用于非特权扇区或非特权存储器的非特权代码。

- *直接存储器访问控制器 (GPDMAx)*

  当 DMA 通道 x 定义为特权（GPDMA_PRIVCFGR 中的 PRIVx = 1）时，访问特权/非特权源或目标时将应用特殊规则。下表总结了这些规则。

表 2. DMA 通道使用（特权）

目标	特权 DMA 通道 x (PRIVx = 1)		非特权 DMA 通道 y (PRIVy = 0)	
	特权源	非特权源	特权源	非特权源
特权	正常		阻止传输(1)	
非特权			阻止传输	正常

1. 阻止传输时，仍可以完成传输，但相应的写操作将被忽略，读操作返回全零。

- *电源控制 (PWR)*

  默认情况下，上电或系统复位后，除了 PWR_PRIVCFGR 之外的所有 PWR 寄存器都可以在特权模式和非特权模式下读取或写入。

  当 PWR_PRIVCFGR 中的特权位 NSPRIV 置 1 时，只能在特权模式下读取和写入 PWR 寄存器。

  有关详细信息，请参见*第 9 章：电源控制 (PWR)*。

- *复位和控制模块 (RCC)*

  默认情况下，上电或系统复位后，除了 RCC_PRIVCFGR 之外的所有 RCC 寄存器都可以在特权模式和非特权模式下读取或写入。

  当 RCC_PRIVCFGR 中的特权位 NSPRIV 置 1 时，只能在特权模式下读取和写入 RCC 位。

  有关详细信息，请参见*第 10 章：复位和时钟控制 (RCC)*。

- *实时时钟 (RTC)*

  默认情况下，任何备份域复位后，除了 RTC_PRIVCFGR 之外的所有 RTC 寄存器都可以在特权模式和非特权模式下读取或写入。

  当仅特权 RTC_PRIVCFGR 中的 PRIV 位置 1 时：

  – 只能在特权模式下写入 RTC 寄存器。

  – 始终可以在特权模式和非特权模式下读取 RTC_PRIVCFGR、RTC_TR、RTC_DR、RTC_SSR、RTC_PRER 和 RTC_CALR。

  所有其他 RTC 寄存器都只能在特权模式下读取。

  当仅特权 RTC_PRIVCFGR 寄存器中的 PRIV 位清零时，仍然可以通过将专用控制位置 1 来限制对某些 RTC 寄存器的特权模式访问：INITPRIV、CALPRIV、TSPRIV、WUTPRIV、ALRAPRV 或 ALRBPRIV。

  有关详细信息，请参见*第 32 章：实时时钟 (RTC)*。

- *入侵和备份寄存器 (TAMP)*

  默认情况下，任何备份域复位后，除了 TAMP_PRIVCFGR 之外的所有 TAMP 寄存器都可以在特权模式和非特权模式下读取或写入。

  当仅特权 TAMP_PRIVCFGR 中的 PRIV 位置 1 时：

  – 只能在特权模式下写入 TAMP 寄存器，但具有自己的保护设置的备份寄存器和单调计数器除外。

  – 始终可以在特权模式和非特权模式下读取 TAMP_CFGR 或 TAMP_PRIVCFGR。所有其他 TAMP 寄存器都只能在特权模式下读取，但具有自己的保护设置的备份寄存器和单调计数器除外。

  应用程序也可以：

  – 将 TAMP_PRIVCFGR 中的 CNTPRIV 位置 1，从而使 TAMP_COUNT1R 寄存器只能进行特权读写

  – 使用 TAMP_PRIVCFGR 中的 BKPRWPRIV 和 BKPWPRIV 位，为备份寄存器的三个保护区中的两个提高安全性：

    – 使保护区 1 进行特权读取和特权写入。

    – 使保护区 2 进行特权/非特权读取和特权写入。

    – 使保护区 3 始终进行特权/非特权读写。

- *通用 I/O (GPIO)*

  所有 GPIO 寄存器均可通过特权访问和非特权访问进行读写操作。

- *扩展中断和事件控制器 (EXTI)*

  EXTI 外设可以保护事件寄存器位免受非特权访问的修改。通过仅特权 EXTI_PRIVCFGRx 寄存器中的寄存器位，可以针对各输入事件单独激活保护。当输入事件配置为特权时，只有特权应用程序才能更改配置、更改屏蔽或清除此输入事件的状态。

  更多详细信息，请参见*第 17 章：扩展中断和事件控制器 (EXTI)*。

- *系统配置自举和安全性 (SBS)*

  除了以下寄存器之外，所有 SBS 寄存器均可在特权模式和非特权模式下进行读写操作：

  – 用于 CPU 配置的 SBS 寄存器：SBS _FPUIMR、SBS _CNSLCKR 和 SBS_ECCNMIR

  更多详细信息，请参见*第 13 章：系统配置、自举和安全性 (SBS)*。

3.3 安全执行

通过将软件特性与硬件特性相结合，器件能够确保其功能正确运行，从而避免因编程工具错误、通过网络访问发动的软件攻击或本地尝试入侵代码执行而导致的异常情况。

本节介绍专为安全执行而设计的硬件特性。

3.3.1 存储器保护单元 (MPU)

Cortex-M33 包含一个存储器保护单元 (MPU)，能够根据以下一个或多个参数来限制对存储区域（包括映射到外设的区域）的读写访问：

- Cortex-M33 工作模式（特权和非特权）
- 数据/取指

存储器映射和 MPU 编程将存储器划分成多达八个区域。

3.3.2 嵌入式 Flash 写保护

写保护 (WRP) 可以防止对嵌入式 Flash 用户区域的选定扇区进行非法或意外的写/擦除操作（系统区域永久写保护）。

只有在 PRODUCT_STATE = 开放时，才能通过选项字节更改来修改受写保护的扇区。

注意： *当存储区内包含写保护区域（WRP 或 HDP 区域）时，存储区擦除操作会中止。更多信息，请参见第 7.5.6 节：写保护。*

3.3.3 入侵检测和响应

原理

器件凭借以下特性来主动保护关键安全资产免遭温度、电压和频率攻击：

- 检测到入侵后擦除器件机密信息
- 进一步确保 CPU 及其相关安全外设的安全执行，包括：
  - 超范围电压（示例：V_{BAT} 和 V_{DDA}）、温度和时钟 (LSE) 检测
  - 安全看门狗 IWDG 的时钟由内部振荡器 LSI 提供
  - 可以将内部振荡器 HSI 选作系统时钟
- 电源保护
  - RTC/TAMP 域通过 V_{DD} 或 V_{BAT} 自动供电

更多详细信息，请参见*第 33 节：入侵和备份寄存器 (TAMP)*。

入侵检测源

器件支持两个有效输入/输出引脚，它们构成了一个独立的主动入侵网格。

有效引脚的时钟由 LSE 提供，可以在不同的系统工作模式（运行、睡眠、停止或待机）和 VBAT 模式下正常工作。有关入侵引脚的列表及其在各电源模式下的可用性，请参见各个模式下的 TAMP 引脚功能。

检测时间是可编程的，并且可以使用数字滤波（在连续四个比较样本中出现两次错误比较后触发入侵）。

注意： *发生入侵事件时将自动生成时间戳。*

下表列出了内部入侵源。

表 3. TAMP 中的内部入侵

入侵输入	TAMP_CR3 中的 NOER 位号	入侵源
itamp1	0	备用域电压连续监测，在 VBAT 模式下正常工作
itamp2	1	温度监视，在 VBAT 模式下正常工作
itamp3	2	LSE 监视(1)，在 VBAT 模式下正常工作
itamp4	3	HSE 监视
itamp5	4	RTC 日历溢出 (rtc_calovf)
itamp6	5	JTAG/SWD 访问
itamp7, 12, 13	6, 11, 12	电压监测 (V_{CORE})，通过 ADC 模拟看门狗

表 3. TAMP 中的内部入侵 （续）

入侵输入	TAMP_CR3 中的 NOER 位号	入侵源
itamp8	7	单调计数器溢出（在内部产生）
itamp9	8	RNG 外设故障生成
itamp11	10	IWDG 超时和潜在入侵（当至少有一个使能的入侵标志置 1 时 IWDG 复位）
itamp15	14	系统故障检测

1. LSE 丢失或超频检测 (> 2 MHz)、干扰滤波器 (> 2 MHz)。

入侵响应

可以对器件中的各个入侵源进行配置，从而触发以下事件：

- 生成中断，可以将器件从停止模式和待机模式唤醒（请参见 TAMP_CR2 寄存器中的 TAMPxMSK 位）。
- 为低功耗定时器生成硬件触发。
- 如果 TAMP_CR2（对应于入侵引脚）或 TAMP_CR3（对应于内部入侵）中相应的 TAMPxNOER 位清零，则擦除器件机密信息。可擦除的机密信息包括：
  - 存储在备份寄存器 (x32) 和 HASH 中的对称密钥
  - 存储在 SRAM2 和 CPU 指令缓存存储器中的其他机密信息（存在 V_{DD} 时擦除 SRAM2）
  - 2 KB 备份 SRAM（取决于配置位）（存在 V_{DD} 时擦除）

将 TAMP_CR2 中的 BKBLOCK 位置 1，可以阻止软件对上述所有机密信息的读/写访问。只有在 BKBLOCK 清零且没有任何使能的入侵源的入侵标志置 1 时，才能对器件机密信息进行访问。

如果不存在 V_{DD}，则存在 V_{DD} 时会擦除的机密信息只能在下一次 V_{DD} 上电时被擦除。

注意： *将 TAMP_CR2 中的 BKERASE 位置 1 或者执行第 3.5.1 节中定义的 PRODUCT_STATE 回退，也可以触发器件机密信息擦除。*

当系统复位或器件从待机模式唤醒时，不会重置器件机密信息。

软件滤波机制

每个入侵源都可以配置为不立即擦除，方法是将 TAMP_CR2（对应于外部入侵引脚）或 TAMP_CR3（对应于内部入侵）中相应的 TAMPxNOER 位置 1。

在这种情况下，当入侵标志置 1 时，将阻止对以下机密信息进行访问，直到所有入侵标志清零：

- 备份寄存器、备份 SRAM 和 SRAM2：读为零，写忽略
- HASH 外设：由 RCC 自动复位

应用程序收到入侵事件通知并对情况进行分析后，可能出现两种情况：

- 应用程序通过软件命令启动机密信息擦除（确认入侵）。
- 应用程序仅清零标志，以允许访问机密信息（虚假入侵）。

注意： *如果入侵软件无法响应此类入侵标志，则 IWDG 复位将自动触发机密信息擦除。*

入侵检测和低功耗模式

下表总结了低功耗模式对入侵检测的影响。

表 4. 低功耗模式对 TAMP 的影响

模式	说明
睡眠	对入侵检测功能没有影响。 TAMP 中断可使器件退出睡眠模式。
停止	对入侵检测功能没有影响，但使用滤波和主动入侵模式的电平检测除外，这些模式仅在时钟源为 LSE 或 LSI 时保持有效。 入侵事件可使器件退出停止模式。
待机	对入侵检测功能没有影响，但使用滤波和主动入侵模式的电平检测除外。这些模式仅在时钟源为 LSE 或 LSI 时保持有效。 入侵事件可使器件退出待机模式。

3.4 唯一 ID

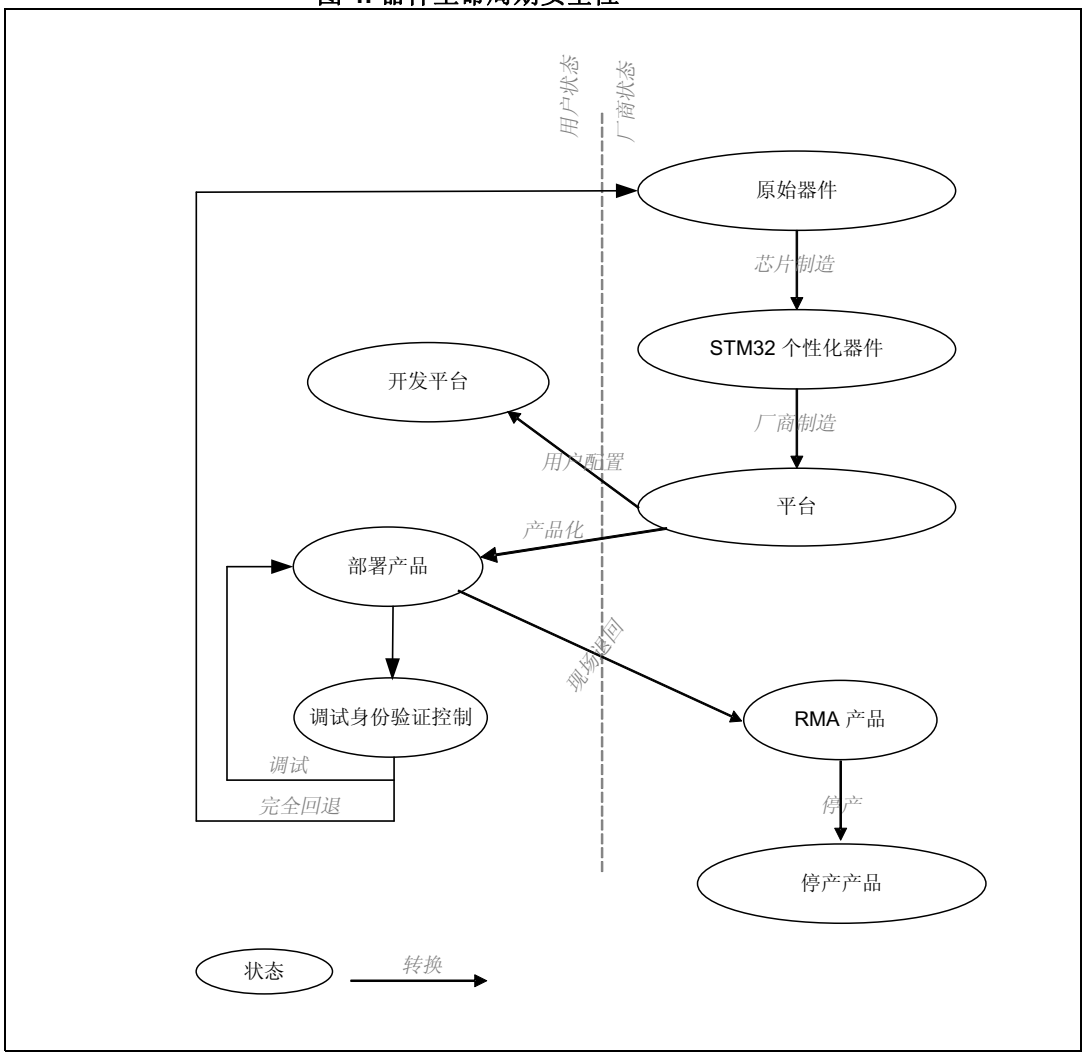
器件内部会存储一个 96 位 ID，每个器件的 ID 都是唯一的（请参见*第 42.1 节：唯一器件 ID 寄存器（96 位）*）。

应用程序可以使用这个唯一的身份密钥来识别云网络中的产品，同时也会使假冒伪劣器件或复制品难以将不受信任的数据注入网络。

3.5 产品生命周期

下图总结了典型的 IoT 设备生命周期。对于每个步骤，器件都提供了内置于硬件中的安全生命周期管理机制。

图 4. 器件生命周期安全性



下表总结了从厂商或最终用户处获取的各个阶段及相关转换的详细信息。

表 5. 主要产品生命周期转换

转换	说明
器件制造	意法半导体创建新的 STM32 器件，并持续关注制造缺陷。在此过程中，为 STM32 配置了 ROM 固件。
厂商制造	一个（或多个）厂商负责平台组装、初始化和配置，然后交付给最终用户。最终用户可以使用最终产品（“生产”转换），也可以使用平台进行软件开发（“用户配置”转换）。
产品化	最终用户获得可供使用的产品。平台的所有安全功能均启用，而调试/测试功能受到限制/禁止。
用户配置	平台厂商准备一个单独的开发平台，此平台不会连接到生产云网络。
现场回退	产品返回现场回退中心进行分析。可以启动完全回退以进行分析。此类访问需要配置厂商提供的密码散列。

3.5.1 生命周期管理

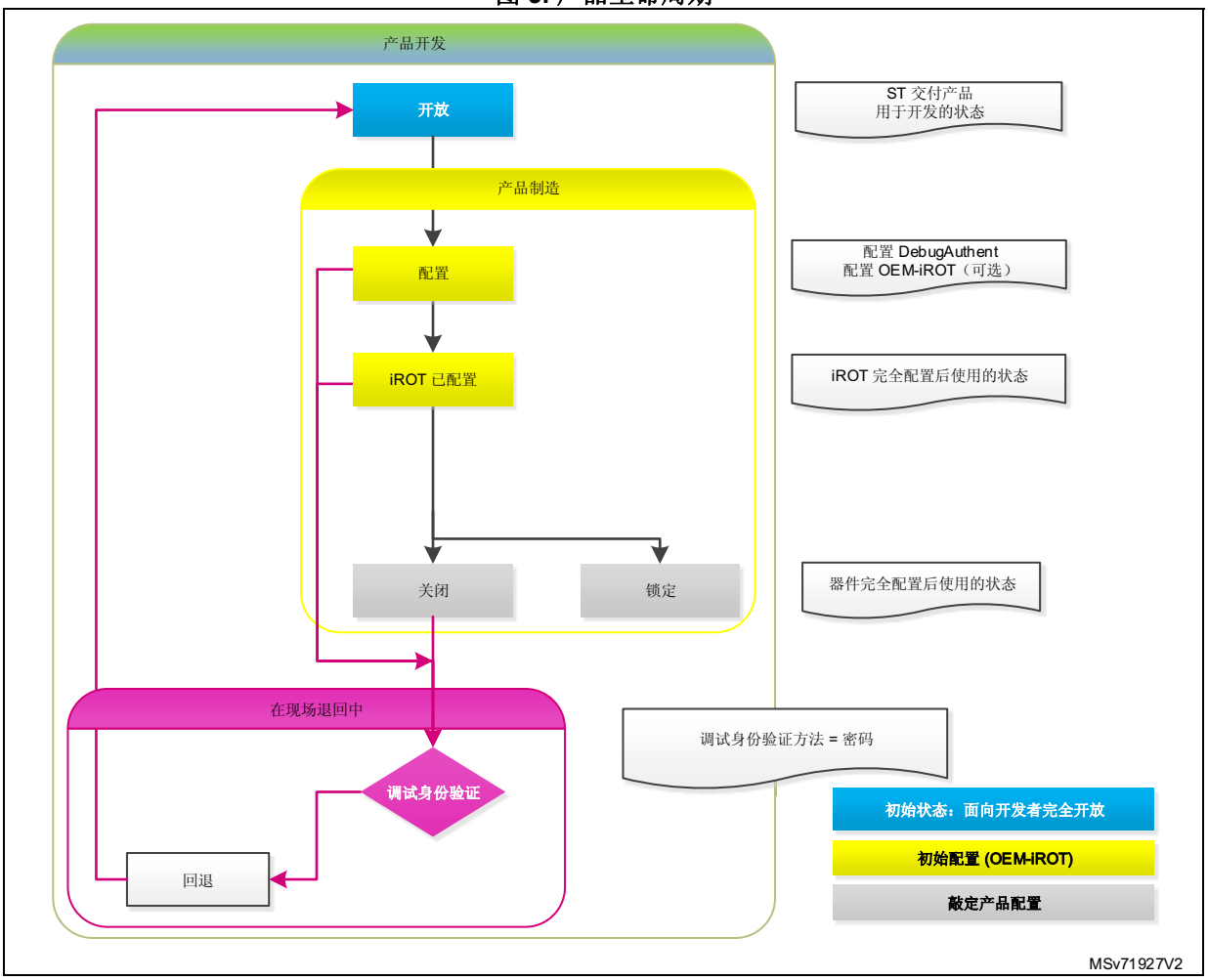
产品生命周期允许控制对产品不同资产（代码和数据）的访问，包括开发、制造和售后阶段。支持使用不同的分配模型来配置产品，同时保护已配置的代码和数据。

表 6. 典型产品生命周期阶段

PRODUCT_STATE + DebugState		调试（默认配置）	备注
开放	器件开放	完全可用	此状态允许开发产品，因为它支持代码的调试。 使用自举引脚可以启动自举程序。
配置	调试部分开放	HDPL3	此状态允许管理产品配置（部分或全部），并且允许启动自举程序来配置产品。
iROT 已配置	调试部分开放	HDPL3	此状态认定安装了不可变可信根，包括相应的配置（代码和选项字节）。
关闭	调试关闭	NoDebug + 调试身份验证控制	此状态认定产品配置已完成。支持对现场维修的调试身份验证（请参见专用应用笔记）。
锁定	调试锁定	无	此状态认定产品配置已完成。不允许进行调试身份验证。产品最终将处于此状态。
回退	调试关闭	NoDebug	此状态是一种临时状态（非易失），用于管理完全回退到开放状态。它会删除所有用户 Flash 代码和数据，包括隐藏保护区域 (HDP)。

支持的转换（如下图汇总）可以通过调试接口或系统自举程序来请求（如果可用）。

图 5. 产品生命周期



3.5.2 推荐的产品设置

为了简化产品维护（现场），建议使用称为调试身份验证控制的功能。此功能可以维护产品激活调试，并且可以在考虑可读信息安全性的同时管理回退。这意味着需要执行以下操作：

- 第一个重要步骤是配置身份验证密码的 HASH。
- 完成产品配置后，将 PRODUCT_STATE 设置为关闭状态。
- 应使用身份验证密码的 HASH（假定用户已在 OTP 中配置）进行调试身份验证，以便启动产品的完全回退。

小心： 用户必须确保在将产品状态从开放更改为任何其他状态（配置、iROT 已配置或关闭）之前配置了身份验证密码的 HASH。
用户必须注意的是，将产品状态更改为已锁定是不可逆的，因此只要更改调试，就一定会丢失器件。

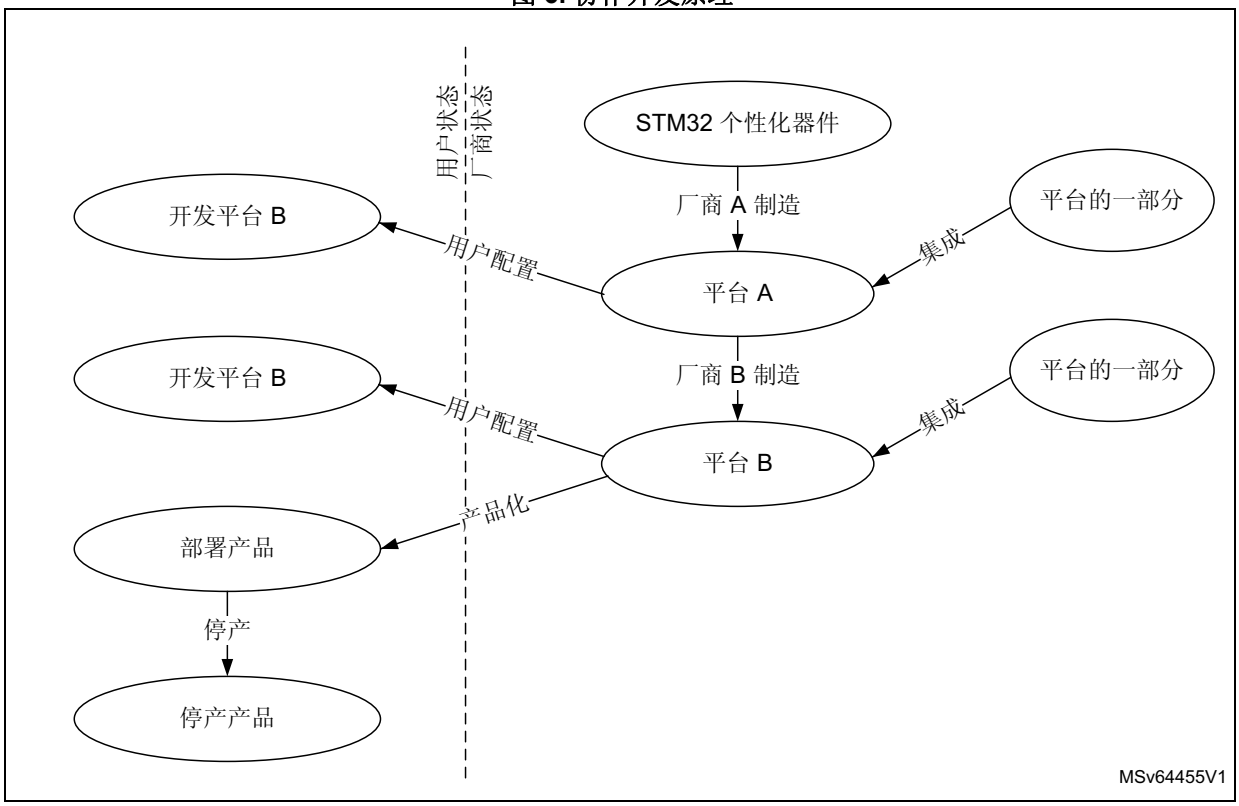
3.6 软件知识产权保护与协作开发

得益于软件知识产权保护与协作模型，器件可用于设计集成创新型第三方库的解决方案。

下图总结了协作开发原理。从意法半导体销售的个性化器件开始，厂商 A 可以在平台 A 上集成部分硬件和软件。厂商 B 在将最终产品部署到最终用户之前也同样可以使用平台 A 进行集成。

注意： *每个平台厂商都可以准备自己的开发平台，这些平台不会连接到生产云网络（“开发平台 X”）。*

图 6. 协作开发原理



下文介绍的功能有助于在此类协作开发中保护软件知识产权。

3.6.1 软件知识产权保护

如*第 3.5.1 节*中所述，硬件 PRODUCT_STATE 机制会自动控制对器件中配置的机密信息的访问。机密信息保护的定义如下表所示。

表 7. 通过 PRODUCT_STATE 实现软件知识产权保护

PRODUCT_STATE 保护级别		机密信息保护
开放	器件开放	无特殊保护。
配置	调试部分开放	所有受 HDPL1 和 HDPL2 保护的区域都无法转存、调试或跟踪。iROT 可以设置更高级别的保护。
iROT 已配置	调试部分开放	所有受 HDPL1 和 HDPL2 保护的区域都无法转存、调试或跟踪。iROT 可以设置更高级别的保护。
关闭	禁止调试	除非在完全回退后通过调试身份验证，否则无法进行调试。
锁定	禁止调试	所有存储在器件中或在外部 Flash 中加密的数据和代码都无法转存明文、调试或跟踪。

3.6.2 其他软件知识产权保护

器件对软件知识产权的其他保护包括：

- 通过检测来抵御侵入性攻击（如物理入侵或干扰）。然后，在检测到的攻击成功之前停用器件。
- 不在侧信道（如时序、功率和电磁辐射）泄露机密信息，以此应对非侵入性攻击（如侧信道攻击）。

4 自举模式

启动时，BOOT0 引脚和 NSBOOTADD[31:8] 选项字节用于选择自举存储器地址，具体包括：

- 从用户 Flash 的任意地址自举
- 从系统存储器自举
  - 自举程序
  - 调试身份验证库 (ST-DA)

选择从用户 Flash 自举时，自举地址由 NSBOOTADD 定义。此地址可通过 NSBOOT_LOCK 锁定。

内部自举程序

嵌入式自举程序位于系统存储器中，由意法半导体在生产阶段编程。它使用 USART、I2C、I3C、SPI、FDCAN 或 USB（设备模式）通过 DFU（器件固件升级）对 Flash 重新编程。

请参见应用笔记《STM32 微控制器系统存储器自举模式》(AN2606)。

嵌入式调试身份验证 (ST-DA)

嵌入式 ST-DA 位于系统存储器中，由 ST 在生产阶段编程。ST-DA 是用于管理调试身份验证协议的库，在实际应用环境中保证以安全的方式启动调试和产品的安全回退。

可以从 ST 工具（STM32CubeProgrammer 或 IDE）启动 ST-DA。应使用验证密码进行调试身份验证，以便启动产品的完全回退。

4.1 STM32H503 自举模式

下表提供了 STM32H503 器件自举模式的详细信息。

表 8. STM32H503 自举模式

PRODUCT_STATE	BOOT0 引脚	自举地址选项字节选择	自举区域	ST 编程的默认值
开放	0	NSBOOTADD[31:8]	自举地址由用户选择字节 NSBOOTADD[31:8] 定义	Flash：0x0800 0000
	1	NA	自举程序	自举程序
配置	x	NA	自举程序	自举程序
已配置、关闭和锁定	x	NSBOOTADD[31:8]	自举地址由用户选择字节 NSBOOTADD[31:8] 定义	Flash：0x0800 0000

5 RAM 配置控制器 (RAMCFG)

5.1 简介

RAMCFG 用于配置内部 SRAM（SRAM1、SRAM2 和 BKPSRAM）的特性。

5.2 RAMCFG 主要特性

下面列出了内部 SRAM 支持的所有特性，具体每个内部 SRAM 支持哪些特性在 RAMCFG 中进行配置：

- 错误代码校正 (ECC)：
  - 单错误检测和校正与中断生成
  - 双重错误检测与中断或 NMI 生成
  - 失效地址的状态
- 写保护（1 KB 粒度）
- SRAM 软件擦除

5.3 RAMCFG 功能说明

5.3.1 内部 SRAM 特性

器件内置三个 SRAM，每个 SRAM 都具有各自的特性：

- SRAM1 和 SRAM2 是主要的 SRAM。

  这两个 SRAM 由块构成，可以在停止模式下掉电以降低功耗：
  - SRAM1：1 block（块）-16KB
  - SRAM2：1 block（块）-16KB

  备份 SRAM (BKPSRAM) 在所有低功耗模式以及 VBAT 模式下 V_{DD} 断开时都可以保留数据。

  更多详细信息，请参见*第 9 章：电源控制 (PWR)*。
- 如果在 Flash 用户选项字节中选择了 SRAM2_RST 选项位，则系统复位时将擦除 SRAM2。如果在 Flash 用户选项字节中选择了 SRAM1_RST 选项位，则系统复位时将擦除 SRAM1。更多详细信息，请参见*第 7 章：嵌入式 Flash (FLASH)*。
- SRAM2 和备份 SRAM（可选）由入侵检测电路保护，检测到入侵时将由硬件擦除。如果备份域复位，它们也将由硬件擦除。更多详细信息，请参见*第 33 章：入侵和备份寄存器 (TAMP)*。
- RAMCFG 内置与内部 SRAM ECC、写保护和软件擦除相关的寄存器。

下表总结了每个内部 SRAM 支持的特性。

表 9. 内部 SRAM 特性

SRAM 特性	SRAM1 (16 KB)	SRAM2 (16 KB)	BKPSRAM (2 KB)
可选择在待机模式下保留数据	-	-	X
可选择在 VBAT 模式下保留数据	-	-	X
可选择在停止模式下保留数据	X	X	X
通过产品状态回退来擦除	X	X	X
可选择通过入侵检测来擦除	-	X	X
备份域复位	-	-	X
可选择通过系统复位来擦除	X	X	-
软件擦除	X	X	X
ECC	X	X	X
写保护	-	X	X(1)

1. 在备份域中，可使用禁止备份域写保护 (DBP) 位来使能/禁止 BKPSRAM 写保护。

5.3.2 错误代码校正（SRAM1、SRAM2、BKPSRAM）

SRAM1、SRAM2 和 BKPSRAM 支持 ECC，但需事先通过 SRAM1_ECC、SRAM2_ECC 和 BKPRAM_ECC 用户选项位将其使能。更多详细信息，请参见*第 7 章：嵌入式 Flash (FLASH)*。

每 32 位 SRAM 增加 7 个 ECC 位，这样可在发生存储器读访问时进行两位错误检测和一位错误校正。

针对 32 位字计算和检查 ECC 时，字节和半字写访问由 SRAM 接口管理，后者首先读取整个字，然后用新的字节/半字值再次写入该字。在这些字节或半字 AHB 写访问期间（由接口完成读/修改/写操作），还会检测 ECC 双重错误。字节或半字写访问延迟为 2 个 AHB 时钟周期。

小心： 如果在支持 ECC 的 SRAM 上进行字节或半字写操作，则读/修改/写操作在缓冲区中完成。在 SRAM AHB 释放后（不再访问 SRAM 时）两个 AHB 时钟周期，缓冲区内容写入 SRAM。

ECC 单错误和双重错误

当检测到单错误时，会自动进行校正，并且 *RAMCFG 存储器中断状态寄存器 (RAMCFG_MxISR)* 和 *RAMCFG 存储器 x 中断清零寄存器 x(RAMCFG_MxICR)* 中的 SEDC/CSEDC 位相应置 1。如果通过 *RAMCFG 存储器 x 中断使能寄存器 (RAMCFG_MxIER)* 中的 SEIE 位使能了中断，则还会产生中断。如果 *RAMCFG 存储器 x 控制寄存器 (RAMCFG_MxCR)* 中的 ALE 位置 1，则失效地址存储在 *RAMCFG 存储器 x ECC 单错误地址寄存器 (RAMCFG_MxSEAR)* 中。

小心： 当 SEDC 位置 1 时，无法检测单错误。

当检测到双重错误时，*RAMCFG 存储器中断状态寄存器 (RAMCFG_MxISR)* 和 *RAMCFG 存储器 x 中断清零寄存器 x(RAMCFG_MxICR)* 中的 DED 和 CDED 位相应置 1。如果通过 *RAMCFG 存储器 x 中断使能寄存器 (RAMCFG_MxIER)* 中的 DEIE 或 ECCNMI 位使能了中断或 NMI，则还会产生中断或 NMI。如果 *RAMCFG 存储器 x 控制寄存器 (RAMCFG_MxCR)* 中的 ALE 位置 1，则失效地址存储在 *RAMCFG 存储器 x ECC 双重错误地址寄存器 (RAMCFG_MxDEAR)* 中。

小心： 当 DED 位置 1 时，无法检测双重错误。

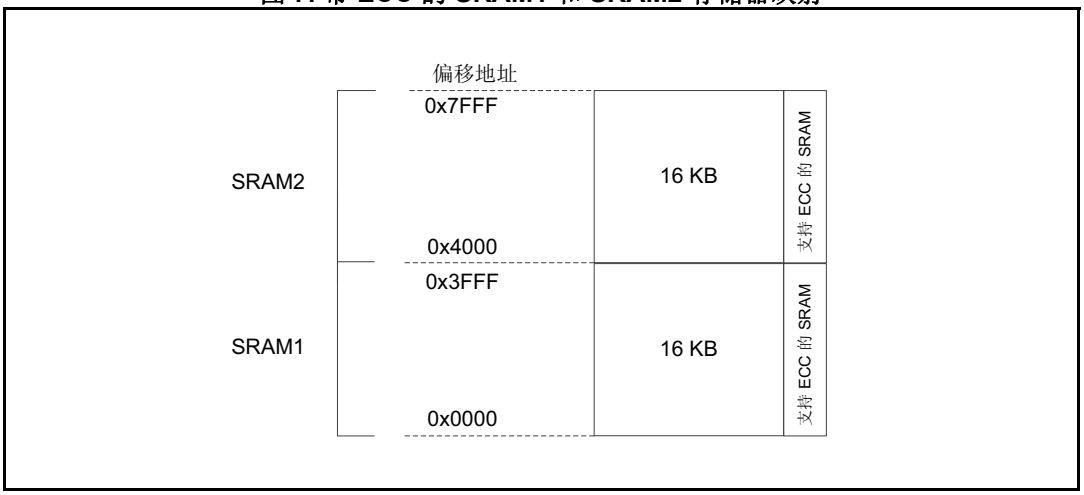
带 ECC 的 SRAM1/SRAM2 存储器映射

当使能 SRAM1 ECC 时，整个 16 KB SRAM1 都支持 ECC 功能。

当使能 SRAM2 ECC 时，整个 16 KB SRAM2 都支持 ECC 功能。

下图显示了对 SRAM1 和 SRAM2 使能 ECC 后的 SRAM 区域。

图 7. 带 ECC 的 SRAM1 和 SRAM2 存储器映射



当通过用户选项位使能 ECC 时，相关 *RAMCFG 存储器 x 控制寄存器 (RAMCFG_MxCR)* 中的 ECCE 位在系统复位后自动置 1。

可通过执行以下软件序列来禁用 ECC：

1. 将 0xAE 写入 *RAMCFG 存储器 x ECC 密钥寄存器 (RAMCFG_MxECCKEYR)*。
2. 将 0x75 写入 *RAMCFG 存储器 x ECC 密钥寄存器 (RAMCFG_MxECCKEYR)*。
3. 向 *RAMCFG 存储器 x 控制寄存器 (RAMCFG_MxCR)* 的 ECCE 位写入 0。

测试 ECC 机制

要测试 ECC 机制，需分别按字 (WORD) 修改 1 个 bit 位和 2 个 bit 位，分别用于 ECC 的 1-bit 位和 2-bit 位错误测试。

检查 ECC 的步骤如下：

1. 使能 ECC，然后在已擦除的存储器上写入数据。
2. 禁止 ECC。
3. 写入修改过 1-bit 或 2-bit 的相同数据（用于 1-bit 位或 2-bit 位的 ECC 错误测试）。
4. 使能 ECC。
5. 读取数据。将产生 ECC 错误中断（由于单或双 ECC 错误中断前面已经使能）。

5.3.3 写保护 (SRAM2)

SRAM2 由 16 个 1 KB 页组成。通过将 *RAMCFG 存储器 2 写保护寄存器 1 (RAMCFG_M2WPR1)* 中的相应 PxWP（x = 0 到 15）位置 1，可以对每个 1 KB 页进行写保护。

5.3.4 软件擦除

可通过执行以下软件序列来请求擦除 SRAM：

1. 将 0xCA 写入 *RAMCFG 存储器 x 擦除密钥寄存器 (RAMCFG_MxERKEYR)*。
2. 将 0x53 写入 *RAMCFG 存储器 x 擦除密钥寄存器 (RAMCFG_MxERKEYR)*。
3. 向 *RAMCFG 存储器 x 控制寄存器 (RAMCFG_MxCR)* 的 SRAMER 位写入 1。

只要正在进行擦除操作，相关 SRAM 中断状态寄存器中的 SRAMBUSY 标志就会置 1。

每次 SRAM 擦除的总持续时间为 N 个 AHB 时钟周期，其中 N 是以 32 位字表示的 SRAM 大小。

如果在进行擦除操作时读取或写入 SRAM，则在 AHB 总线上插入等待周期，直到擦除操作结束为止。

5.4 RAMCFG 低功耗模式

表 10. 低功耗模式对 RAMCFG 的影响

模式	说明
睡眠	无影响。RAMCFG 中断可使器件退出睡眠模式。
停止	RAMCFG 寄存器的内容会保留。
待机	RAMCFG 外设掉电，退出待机模式后必须重新初始化。

5.5 RAMCFG 中断

下表列出了 RAMCFG 中断请求。

表 11. RAMCFG 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止模式	退出待机模式
RAMCFG	ECC 单错误检测和校正	SEDC	SEIE	向 CSEDC 中写入 1	是	否	否
	ECC 双重错误检测	DED	DEIE = 1 且 ECCNMI = 0	向 CDED 中写入 1	是	否	否
NMI	ECC 双重错误检测	DED	ECCNMI	向 CDED 中写入 1	是	否	否

5.6 RAMCFG 寄存器

在下述寄存器中，x 的值如下：

- SRAM1/2 (x = 1/2)
- BKPSRAM (x = 5)

5.6.1 RAMCFG 存储器 x 控制寄存器 (RAMCFG_MxCR)

RAMCFG memory x control register

偏移地址：0x040 * (x - 1)，（x = 1、2 和 5）

复位值：0x0000 000X

ECCE 复位值取决于 ECC 使能用户选项位。

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

位 31:9 保留，必须保持复位值。

位 8 **SRAMER**：SRAM 擦除 (SRAM erase)

只有在 RAMCFG_MxERKEYR 寄存器的 ERASEKEY 位域中写入解锁序列后，此位才可由软件置 1。将此位置 1 可启动 SRAM 擦除。擦除操作结束后，此位由硬件自动清零。

0：未在进行擦除操作

1：正在进行擦除操作

位 7:5 保留，必须保持复位值。

位 4 **ALE**：地址锁存使能 (Address latch enable)

0：失效地址不存储到 SRAMx ECC 单错误/双重错误地址寄存器中

1：失效地址存储到 SRAMx ECC 单错误/双重错误地址寄存器中

注意：在 SRAM1 控制寄存器中，此位保留且必须保持复位值。

位 3:1 保留，必须保持复位值。

位 0 **ECCE**：ECC 使能 (ECC enable)

此位的复位值由用户选项位配置定义。此位置 1 时，只有在 RAMCFG_MxECCKEYR 寄存器中写入解锁序列后，才可由软件清零。

0：禁止 ECC

1：使能 ECC

注意：在 SRAM1 控制寄存器中，此位保留且必须保持复位值。

5.6.2 RAMCFG 存储器 x 中断使能寄存器 (RAMCFG_MxIER)

RAMCFG memory x interrupt enable register

偏移地址：0x004 + 0x040 * (x - 1),（x = 2、3、5）

复位值：0x0000 0000

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

位 31:4 保留，必须保持复位值。

位 3 **ECCNMI**：双重错误 NMI (Double error NMI)

此位由软件置 1，只能通过全局 RAMCFG 复位清零。

0：存在 ECC 双重错误时不会产生 NMI

1：存在 ECC 双重错误时产生 NMI

注意：如果 ECCNMI 置 1，则无论 DEIE 位值如何，都不会产生 RAMCFG 可屏蔽中断。

位 2 保留，必须保持复位值。

位 1 **DEIE**：ECC 双重错误中断使能 (ECC double error interrupt enable)

0：禁止双重错误中断

1：使能双重错误中断

位 0 **SEIE**：ECC 单错误中断使能 (ECC single error interrupt enable)

0：禁止单错误中断

1：使能单错误中断

5.6.3 RAMCFG 存储器中断状态寄存器 (RAMCFG_MxISR)

RAMCFG memory interrupt status register

偏移地址：0x008 + 0x040 * (x - 1)，（x = 1、2、3 和 5）

复位值：0x0000 0000

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

位 31:9 保留，必须保持复位值。

位 8 **SRAMBUSY**：SRAM 忙于擦除操作 (SRAM busy with erase operation)

0：未在进行擦除操作

1：正在进行擦除操作

注意：对于不同的 SRAM，可以由于软件请求、系统复位（如果使能了相应的选项位）、检测到入侵或产品状态回退而执行擦除。请参见表 9：内部 SRAM 特性。

位 7:2 保留，必须保持复位值。

位 1 **DED**：检测到 ECC 双重错误 (ECC double error detected)

0：无双重错误

1：检测到双重错误

注意：在 SRAM1 中断状态寄存器中，此位保留且必须保持复位值。

位 0 **SEDC**：检测到 ECC 单错误并进行校正 (ECC single error detected and corrected)

0：无单错误

1：检测到单错误并进行校正

注意：在 SRAM1 中断状态寄存器中，此位保留且必须保持复位值。

5.6.4 RAMCFG 存储器 x ECC 单错误地址寄存器 (RAMCFG_MxSEAR)

RAMCFG memory x ECC single error address register

偏移地址：0x00C + 0x040 * (x - 1),（x = 2、3、5）

复位值：0x0000 0000

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

位 31:0 **ESEA[31:0]**：ECC 单错误地址 (ECC single error address)

当 RAMCFG_MxCR 寄存器中的 ALE 位置 1 时，此位域将更新为 ECC 单错误对应的地址。

5.6.5 RAMCFG 存储器 x ECC 双重错误地址寄存器 (RAMCFG_MxDEAR)

RAMCFG memory x ECC double error address register

偏移地址：0x010 + 0x040 * (x - 1),（x = 2、3、5）

复位值：0x0000 0000

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

位 31:0 **EDEA[31:0]**：ECC 双重错误地址 (ECC double error address)

当 RAMCFG_MxCR 寄存器中的 ALE 位置 1 时，此位域将更新为 ECC 双重错误对应的地址。

5.6.6 RAMCFG 存储器 x 中断清零寄存器 x(RAMCFG_MxICR)

RAMCFG memory x interrupt clear register x

偏移地址：0x014 + 0x040 * (x - 1),（x = 2、3、5）

复位值：0x0000 0000

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

位 31:2 保留，必须保持复位值。

位 1 **CDED**：清零检测到 ECC 双重错误位 (Clear ECC double error detected)

将 1 写入此标志时，RAMCFG_MxISR 寄存器中的 DED 位将清零。读取此标志时将返回 DED 值。

位 0 **CSEDC**：清零检测到 ECC 单错误并进行校正位 (Clear ECC single error detected and corrected)

将 1 写入此标志时，RAMCFG_MxISR 寄存器中的 SEDC 位将清零。读取此标志时将返回 SEDC 值。

5.6.7 RAMCFG 存储器 2 写保护寄存器 1 (RAMCFG_M2WPR1)

RAMCFG memory 2 write protection register 1

偏移地址：0x058

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15WP	P14WP	P13WP	P12WP	P11WP	P10WP	P9WP	P8WP	P7WP	P6WP	P5WP	P4WP	P3WP	P2WP	P1WP	P0WP
rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs

位 31:16 保留，必须保持复位值。

位 15:0 **PyWP**：SRAM2 1 KB 页 y 写保护 (SRAM2 1-Kbyte page y write protection)（y = 15 到 0）
这些位由软件置 1，只能通过全局 RAMCFG 复位清零。
0：禁止 SRAM2 1 KB 页 y 的写保护。
1：使能 SRAM2 1 KB 页 y 的写保护。

5.6.8 RAMCFG 存储器 x ECC 密钥寄存器 (RAMCFG_MxECCKEYR)

RAMCFG memory x ECC key register

偏移地址：0x024 + 0x040 * (x - 1),（x = 2、3、5）

复位值：0x0000 0000

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

位 31:8 保留，必须保持复位值。

位 7:0 **ECCKEY[7:0]**：ECC 写保护密钥 (ECC write protection key)
要解锁 RAMCFG_MxCR 寄存器中 ECCE 位的写保护，需要执行以下步骤。
1) 将 0xAE 写入 ECCKEY[7:0]。
2) 将 0x75 写入 ECCKEY[7:0]。
注意：写入一个错误的关键字会再次激活写保护。

5.6.9 RAMCFG 存储器 x 擦除密钥寄存器 (RAMCFG_MxERKEYR)

RAMCFG memory x erase key register

偏移地址：0x028 + 0x040 * (x - 1)，（x = 1 到 5）

复位值：0x0000 0000

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

位 31:8 保留，必须保持复位值。

位 7:0 **ERASEKEY[7:0]**：擦除写保护密钥 (Erase write protection key)

要解锁 RAMCFG_MxCR 寄存器中 SRAMER 位的写保护，需要执行以下步骤。

1) 将 0xCA 写入 ERASEKEY[7:0]。

2) 将 0x53 写入 ERASEKEY[7:0]。

注意：写入一个错误的关键字会再次激活写保护。

5.6.10 RAMCFG 寄存器映射

表 12. RAMCFG 寄存器映射和复位值

偏移	寄存器名称和复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**RAMCFG_M1CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRAMER	Res.	Res.	Res.	ALE	Res.	Res.	Res.	ECCE
	Reset value																								0				0				x
0x40	**RAMCFG_M1IER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCNMI	Res.	DEIE	SEIE
	Reset value																													0		0	0
0x08	**RAMCFG_M1ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRAMBUSY	Res.	Res.	Res.	Res.	Res.	Res.	DED	SEDC
	Reset value																								0							0	0
0x0C	**RAMCFG_ M1SEAR**	ESEA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	**RAMCFG_ M1DEAR**	EDEA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	**RAMCFG_M1ICR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CDED	CSEDC
	Reset value																															0	0
0x18 to 0x20	Reserved	Reserved																															
0x024	**RAMCFG_ M1ECCKEYR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCKEY[7:0]							
	Reset value																									0	0	0	0	0	0	0	0

表 12. RAMCFG 寄存器映射和复位值（续）

偏移	寄存器名称和复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x28	**RAMCFG_ M1ERKEYR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERASEKEY[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x2C to 0x3C	Reserved	Reserved																															
0x40	**RAMCFG_M2CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRAMER	Res.	Res.	Res.	ALE	Res.	Res.	Res.	ECCE
	Reset value																								0				0				x
0x44	**RAMCFG_M2IER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCNMI	Res.	DEIE	SEIE
	Reset value																													0		0	0
0x048	**RAMCFG_M2ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRAMBUSY	Res.	Res.	Res.	Res.	Res.	Res.	DED	SEDC
	Reset value																								0							0	0
0x04C	**RAMCFG_ M2SEAR**	ESEA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x050	**RAMCFG_ M2DEAR**	EDEA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x054	**RAMCFG_M2ICR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CDED	CSEDC
	Reset value																															0	0
0x058	**RAMCFG_ M2WPR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	P15WP	P14WP	P13WP	P12WP	P11WP	P10WP	P9WP	P8WP	P7WP	P6WP	P5WP	P4WP	P3WP	P2WP	P1WP	P0WP
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x05C to 0x060	Reserved	Reserved																															
0x064	**RAMCFG_ M2ECCKEYR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCKEY[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x068	**RAMCFG_ M2ERKEYR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERASEKEY[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x06c to 0x0FC	Reserved	Reserved																															
0x100	**RAMCFG_M5CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRAMER	Res.	Res.	Res.	ALE	Res.	Res.	Res.	ECCE
	Reset value																								0				0				x
0x104	**RAMCFG_M5IER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCNMI	Res.	DEIE	SEIE
	Reset value																													0		0	0
0x108	**RAMCFG_M5ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRAMBUSY	Res.	Res.	Res.	Res.	Res.	Res.	DED	SEDC
	Reset value																								0							0	0
0x10C	**RAMCFG_ M5SEAR**	ESEA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 12. RAMCFG 寄存器映射和复位值（续）

偏移	寄存器名称和复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x110	**RAMCFG_ M5DEAR**	EDEA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x114	**RAMCFG_M5ICR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CDED	CSEDC
	Reset value																															0	0
0x118 to 0x120	Reserved	Reserved																															
0x124	**RAMCFG_ M5ECCKEYR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCKEY[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x128	**RAMCFG_ M5ERKEYR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERASEKEY[7:0]							
	Reset value																									0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 2.2 节*。

6 全局特权控制器 (GTZC)

6.1 简介

本节介绍了全局特权控制器 (GTZC) 模块，该模块包含以下子模块：

- **TZSC**：特权控制器

  此子模块定义从外设的特权状态。它还控制水印存储器外设控制器 (MPCWM) 的子区域大小和属性。

- **MPCBB**：基于块的存储器保护控制器

  该子模块通过特权属性配置具有分段 SRAM（512 字节的页）的系统中的内部 SRAM（SRAM1 和 SRAM2）。

此类子模块用于通过特权属性配置具有总线代理的产品中的特权，例如：

- 带可编程特权块（页）的片上 RAM
- 具有可编程特权访问权限的 AHB 和 APB 外设
- 授权为特权的 AHB 主设备

6.2 GTZC 主要特性

GTZC 的主要特性如下：

- 2 个独立 32 位 AHB 接口，用于 TZSC 和 MPCBB
- 可对 TZSC 和 MPCBB 进行特权和非特权访问
- 用于定义产品特权设置的寄存器集：
  - 用于内部 SRAM 的特权块（带 MPCBB）
  - 用于内部备份 SRAM 的特权区域（带 MPCWM）
  - 对特权外设和特权感知外设的特权访问模式
  - 对特权感知主器件的特权访问模式

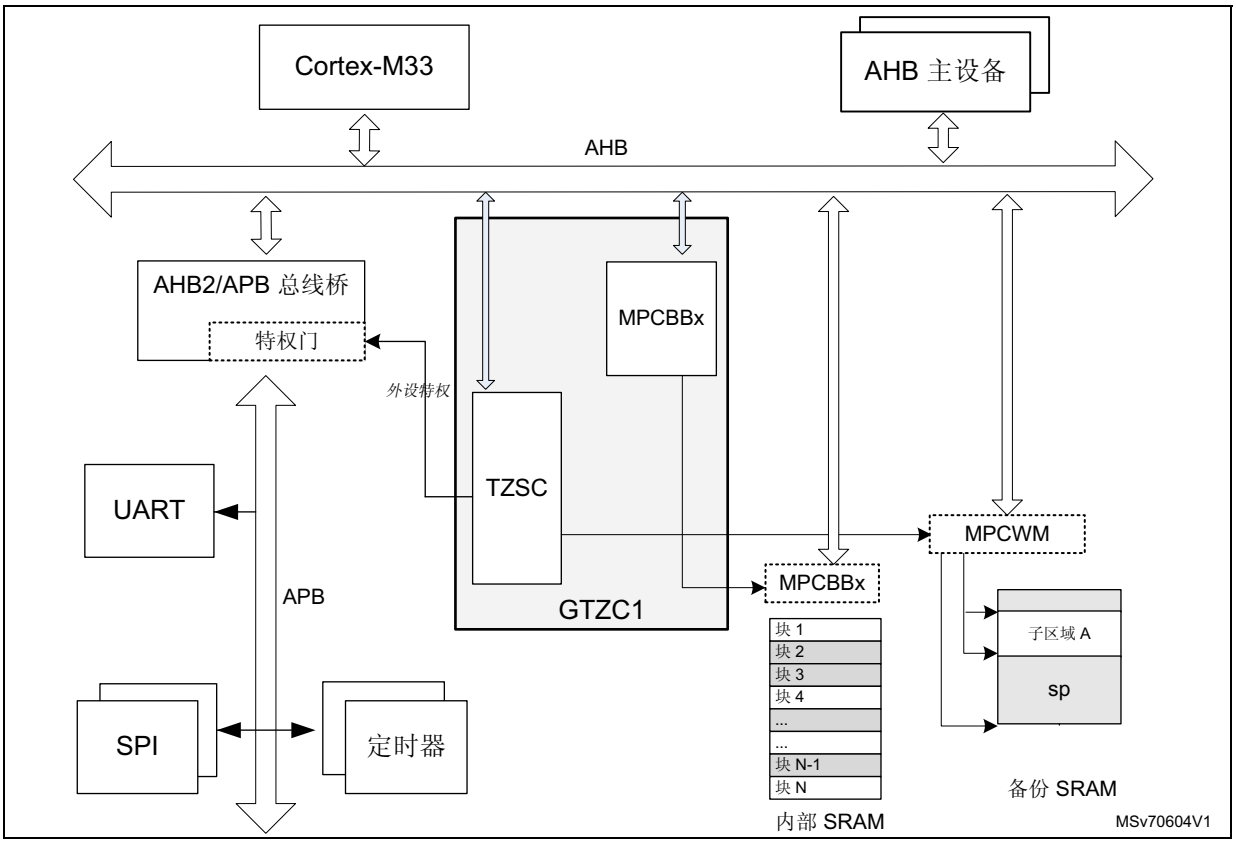
GTZC 特权系统架构

AHB 和 APB 外设可分为：

- **特权**：由 AHB/APB 防火墙桩保护的外设，该桩由 TZSC 控制以定义特权属性
- **非特权**：直接连接到 AHB/APB 互连的外设，没有任何特权门
- **特权感知**：直接连接到 AHB 或 APB 总线、并实现特定特权行为的外设（例如特权寄存器子集）。特权感知 AHB 主器件始终根据其特权模式（如 CPU 或 DMA）驱动 HPROT[1] 信号

包含特权外设和特权感知外设的特权保护架构如下图所示。

图 8. GTZC 特权架构



6.3 GTZC 实现

STM32H503xx 器件嵌入一个 GTZC 实例。

表 13. GTZC 实现

GTZC 子模块	GTZC1
TZSC	X
TZIC	-
MPCBB 子模块数量	2

表 14. GTZC 特性

特性	GTZC
Arm® TrustZone®	-
特权	X

下表显示了 GTZC 子模块相对于 GTZC 基址的偏移地址（有关 GTZC1 基址，请参见*第 2.2 节*）。

表 15. GTZC1 子模块偏移地址

GTZC1 子模块	偏移地址
GTZC1_TZSC	0x0
GTZC1_MPCBB1	0x800
GTZC1_MPCBB2	0xC00

下表介绍了可用 MPCWM 的特性。

表 16. MPCWM 资源分配

GTZC	MPC	目标存储器接口	特权/非特权区域数量	水印粒度（字节）
GTZC1	MPCWM	BKPSRAM	1	32

下表介绍了可用 MPCBB 的特性。

表 17. MPCBB 资源分配

GTZC	MPC	资源	存储器大小 (KB)	块大小（字节）	块数	超级块数
GTZC1	MPCBB1	SRAM1	16	512	32	1
	MPCBB2	SRAM2				

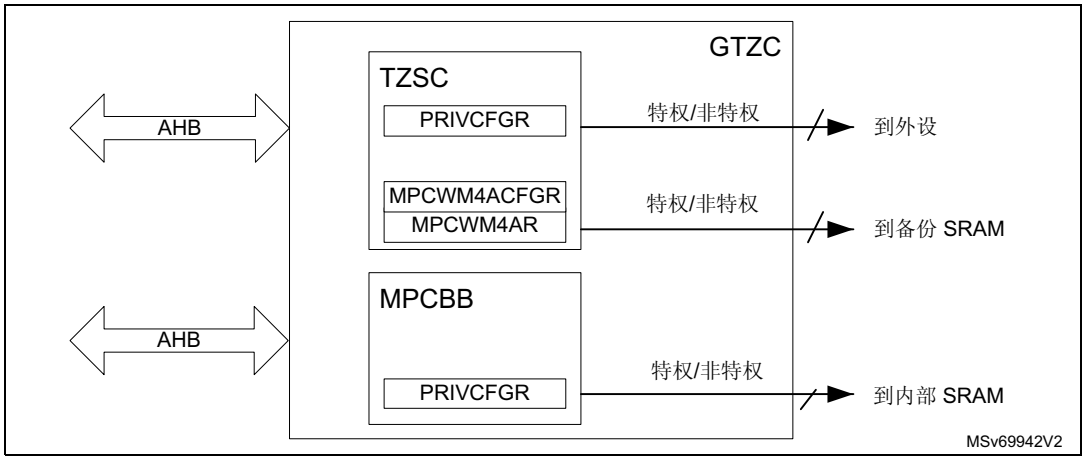
6.4 GTZC 功能说明

6.4.1 GTZC 框图

下图说明了 TZSC 和 MPCBB 的组合功能。每个子模块由自带的 AHB 配置端口控制。

TZSC 定义特权的外设。可以通过特权事务修改外设的特权配置位。

图 9. GTZC 框图



6.4.2 非法的非特权访问

任何试图访问特权资源的非特权事务都被视为非法。针对非法读写访问不生成非法访问事件。寻址的资源遵循“静默-失败”行为，读访问返回全零数据，所有写访问都会被忽略。不会生成总线错误。任何非特权执行事务试图访问特权存储器时，都会生成总线错误。

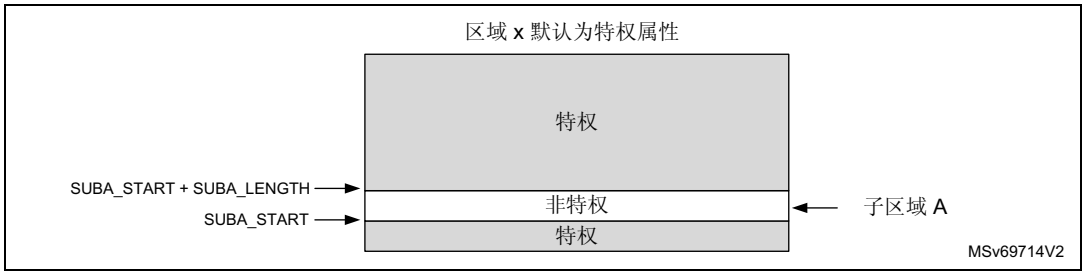
6.4.3 特权控制器 (TZSC)

TZSC 由一组可配置的寄存器组成，具有以下功能：

- 控制所有外设的特权状态，具体方式为通过 GTZC1_TZSC_PRIVCFGRx 寄存器控制特权外设的 AHB/APB 防火墙桩
- 对于水印存储器保护控制器（备份 SRAM），可以定义一个区域，并使用以下位域进行编程：
  - 备份 SRAM 上受保护子区域的起始地址：SUBA_START[10:0]
  - 备份 SRAM 上受保护子区域的长度：SUBA_LENGTH[11:0]

子区域的控制寄存器可用于使能/禁止水印存储器保护控制器以及定义合适属性。

图 10. 水印存储器保护控制器（区域 x/子区域 A）

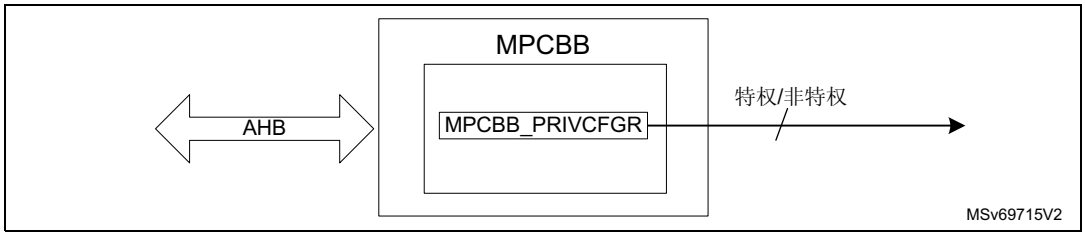


在上图中，区域 x 表示备份 SRAM 区域。子区域 A 的特权属性是可配置的。如果在区域 x 上未定义或使能子区域，则区域 x 的默认属性设为特权。

6.4.4 基于块的存储器保护控制器 (MPCBB)

MPCBB 由一组可配置的寄存器组成，允许定义内部 SRAM 存储器的特权策略。可针对 SRAM 的每个 512 字节块分别配置特权策略。

图 11. MPCBB 框图



为了设置 MPCBB，特权固件必须通过将 GTZC1_MPCBBz_PRIVCFGRx 中的合适位置 1 来定义有特权的存储器块。
MPCBB 超级块由 32 个连续块组成。

注意： *块大小为 512 字节。超级块大小为 512 * 32 = 16 KB。*

6.4.5 上电/复位状态

TZSC 的上电和复位状态时将 GTZC1_TZSC_PRIVCFGRx 的所有位都清零，这意味着所有外设都设置为非特权。

对于内部 SRAMx（x = 1 到 2），所有 GTZC1_MPCBBz_PRIVCFGRx 都设置为 0x0000 0000，默认将这些内部存储器块设为非特权。

对于备份 SRAM，GTZC1_TZSC_MPCWM4AR 设置为 0x0800 0000，默认将存储器设为非特权。

6.5 GTZC1 TZSC 寄存器

所有寄存器只能按字（32 位）进行访问。

6.5.1 GTZC1 TZSC 特权配置寄存器 1 (GTZC1_TZSC_PRIVCFGR1)

GTZC1 TZSC privilege configuration register 1

偏移地址：0x020

复位值：0x0000 0000

仅限写特权访问。

此寄存器只能通过特权事务进行读写操作。

读访问授权可用于任何类型的事务，与特权状态无关。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIM2PRIV	DTSPRIV	Res.	Res.	Res.	Res.	DAC1PRIV	Res.	Res.	Res.	Res.	CRSPRIV	I3C1PRIV	I2C2PRIV	I2C1PRIV	Res.
rw	rw					rw					rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	USART3PRIV	USART2PRIV	SPI3PRIV	SPI2PRIV	IWDGPRIV	WWDGPRIV	Res.	Res.	Res.	TIM7PRIV	TIM6PRIV	Res.	Res.	TIM3PRIV	TIM2PRIV
	rw	rw	rw	rw	rw	rw				rw	rw			rw	rw

位 31 **LPTIM2PRIV**：LPTIM2 的特权访问模式 (privileged access mode for LPTIM2)
0：非特权
1：特权

位 30 **DTSPRIV**：DTS 的特权访问模式 (privileged access mode for DTS)
0：非特权
1：特权

位 29:26 保留，必须保持复位值。

位 25 **DAC1PRIV**：DAC1 的特权访问模式 (privileged access mode for DAC1)
0：非特权
1：特权

位 24:21 保留，必须保持复位值。

位 20 **CRSPRIV**：CRS 的特权访问模式 (privileged access mode for CRS)
0：非特权
1：特权

位 19 **I3C1PRIV**：I3C1 的特权访问模式 (privileged access mode for I3C1)
0：非特权
1：特权

位 18 **I2C2PRIV**：I2C2 的特权访问模式 (privileged access mode for I2C2)
0：非特权
1：特权

位 17 **I2C1PRIV**：I2C1 的特权访问模式 (privileged access mode for I2C1)
0：非特权
1：特权

位 16:15 保留，必须保持复位值。

位 14 **USART3PRIV**：USART3 的特权访问模式 (privileged access mode for USART3)
0：非特权
1：特权

位 13 **USART2PRIV**：USART2 的特权访问模式 (privileged access mode for USART2)
0：非特权
1：特权

位 12 **SPI3PRIV**：SPI3 的特权访问模式 (privileged access mode for SPI3)
0：非特权
1：特权

位 11 **SPI2PRIV**：SPI2 的特权访问模式 (privileged access mode for SPI2)
0：非特权
1：特权

位 10 **IWDGPRIV**：IWDG 的特权访问模式 (privileged access mode for IWDG)
0：非特权
1：特权

位 9 **WWDGPRIV**：WWDG 的特权访问模式 (privileged access mode for WWDG)
0：非特权
1：特权

位 8:6 保留，必须保持复位值。

位 5 **TIM7PRIV**：TIM7 的特权访问模式 (privileged access mode for TIM7)
0：非特权
1：特权

位 4 **TIM6PRIV**：TIM6 的特权访问模式 (privileged access mode for TIM6)
0：非特权
1：特权

位 3:2 保留，必须保持复位值。

位 1 **TIM3PRIV**：TIM3 的特权访问模式 (privileged access mode for TIM3)
0：非特权
1：特权

位 0 **TIM2PRIV**：TIM2 的特权访问模式 (privileged access mode for TIM2)
0：非特权
1：特权

6.5.2 GTZC1 TZSC 特权配置寄存器 2 (GTZC1_TZSC_PRIVCFGR2)

GTZC1 TZSC privilege configuration register 2

偏移地址：0x024

复位值：0x0000 0000

仅限写特权访问。

此寄存器只能通过特权事务进行读写操作。

读访问授权可用于任何类型的事务，与特权状态无关。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	LPTIM1PRIV	Res.	Res.	LPUART1PRIV	Res.	Res.	Res.	Res.	Res.	USBPRIV	Res.	Res.	Res.
			rw			rw						rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	USART1PRIV	Res.	SPI1PRIV	TIM1PRIV	Res.	Res.	Res.	COMPPRIV	OPAMPPRIV	Res.	Res.	FDCAN1PRIV
				rw		rw	rw				rw	rw			rw

位 31:29 保留，必须保持复位值。

位 28 **LPTIM1PRIV**：LPTIM1 的特权访问模式 (privileged access mode for LPTIM1)
0：非特权
1：特权

位 27:26 保留，必须保持复位值。

位 25 **LPUART1PRIV**：LPUART 的特权访问模式 (privileged access mode for LPUART)
0：非特权
1：特权

位 24:20 保留，必须保持复位值。

位 19 **USBPRIV**：USBSF 的特权访问模式 (privileged access mode for USBSF)
0：非特权
1：特权

位 18:12 保留，必须保持复位值。

位 11 **USART1PRIV**：USART1 的特权访问模式 (privileged access mode for USART1)
0：非特权
1：特权

位 10 保留，必须保持复位值。

位 9 **SPI1PRIV**：SPI1 的特权访问模式 (privileged access mode for SPI1)
0：非特权
1：特权

位 8 **TIM1PRIV**：TIM1 的特权访问模式 (privileged access mode for TIM1)
0：非特权
1：特权

位 7:5 保留，必须保持复位值。

位 4 **COMPPRIV**：COMP 的特权访问模式 (privileged access mode for COMP)
0：非特权
1：特权

位 3 **OPAMPPRIV**：OPAMP 的特权访问模式 (privileged access mode for OPAMP)
0：非特权
1：特权

位 2:1 保留，必须保持复位值。

位 0 **FDCAN1PRIV**：FDCAN1 的特权访问模式 (privileged access mode for FDCAN1)
0：非特权
1：特权

6.5.3 GTZC1 TZSC 特权配置寄存器 3 (GTZC1_TZSC_PRIVCFGR3)

GTZC1 TZSC privilege configuration register 3

偏移地址：0x028

复位值：0x0000 0000

仅限写特权访问。

此寄存器只能通过特权事务进行读写操作。

读访问授权可用于任何类型的事务，与特权状态无关。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	RAMCFGPRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNGPRIV	HASHPRIV	Res.
					rw								rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	ADC1PRIV	Res.	ICACHEPRIV	Res.	Res.	Res.	CRCPRIV	Res.	Res.	Res.	Res.	Res.	I3C2PRIV	Res.	Res.
	rw		rw				rw						rw		

位 31:27 保留，必须保持复位值。

位 26 **RAMCFGPRIV**：RAMSCFG 的特权访问模式 (privileged access mode for RAMSCFG)
0：非特权
1：特权

位 25:19 保留，必须保持复位值。

位 18 **RNGPRIV**：RNG 的特权访问模式 (privileged access mode for RNG)
0：非特权
1：特权

位 17 **HASHPRIV**：HASH 的特权访问模式 (privileged access mode for HASH)
0：非特权
1：特权

位 16:15 保留，必须保持复位值。

位 14 **ADC1PRIV**：ADC1 的特权访问模式 (privileged access mode for ADC1)
0：非特权
1：特权

位 13 保留，必须保持复位值。

位 12 **ICACHEPRIV**：ICACHE 的特权访问模式 (privileged access mode for ICACHE)
0：非特权
1：特权

位 11:9 保留，必须保持复位值。

位 8 **CRCPRIV**：CRC 的特权访问模式 (privileged access mode for CRC)
0：非特权
1：特权

位 7:3 保留，必须保持复位值。

位 2 **I3C2PRIV**：I3C2 的特权访问模式 (privileged access mode for I3C2)
0：非特权
1：特权

位 1:0 保留，必须保持复位值。

6.5.4 GTZC1 TZSC BKPSRAM 子区域 A 水印配置寄存器 (GTZC1_TZSC_MPCWM4ACFGR)

GTZC1 TZSC BKPSRAM subregion A watermark configuration register
偏移地址：0x070
复位值：0x0000 0000

仅限特权访问。

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

位 31:10 保留，必须保持复位值。

位 9 **PRIV**：特权子区域 A (Privileged subregion A)
仅当 SREN 置 1 时，才考虑该位。
0：在子区域 A 中授予特权和非特权访问权限。
1：在子区域 A 中仅授予特权访问权限。

位 8:2 保留，必须保持复位值。

位 1 **SRLOCK**：子区域 A 锁定 (Subregion A lock)
此位置 1 后，只能通过系统复位清零。
0：可对 GTZC1_TZSC_MPCWM4ACFGR 和 GTZC1_TZSC_MPCWM4AR 执行写操作。
1：忽略对 GTZC1_TZSC_MPCWM4ACFGR 和 GTZC1_TZSC_MPCWM4AR 的写操作。

位 0 **SREN**：子区域 A 使能 (Subregion A enable)
0：禁止子区域 A。基础区域的访问控制适用于该子区域起始地址和结束地址之间的任何访问。
1：使能子区域 A。GTZC1_TZSC_MPCWM4ACFGR 中定义的访问控制适用于该子区域起始地址和结束地址之间的任何访问，这两个地址在 GTZC1_TZSC_MPCWM4AR 中定义。

6.5.5 GTZC1 TZSC BKPSRAM 子区域 A 水印寄存器 (GTZC1_TZSC_MPCWM4AR)

GTZC1 TZSC BKPSRAM subregion A watermark register
偏移地址：0x074
复位值：0x0800 0000

仅限特权访问。
SUBA_START + SUBA_LENGTH 高于存储器允许的最大大小时，会自动应用 SUBA_LENGTH 的饱和。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	SUBA_LENGTH[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

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

位 31:28 保留，必须保持复位值。

位 27:16 **SUBA_LENGTH[11:0]**：子区域 A 的长度 (Length of subregion A)
该位域定义子区域 A 的长度，将乘以*表 16* 中定义的粒度。
SUBA_START + SUBA_LENGTH 高于存储器允许的最大大小时，会自动应用 SUBA_LENGTH 的饱和。
如果 SUBA_LENGTH = 0，则会禁止子区域 A（GTZC1_TZSC_MPCMW4ACFGR 中的 SREN 位清零）。

位 15:11 保留，必须保持复位值。

位 10:0 **SUBA_START[10:0]**：子区域 A 的起始地址 (Start of subregion A)
该位域定义子区域 A 的偏移地址，将乘以*表 16* 中定义的粒度。

6.5.6 GTZC1 TZSC 寄存器映射

表 18. GTZC1 TZSC 寄存器映射和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000-0x01C	Reserved	Reserved																															
0x020	GTZC1_TZSC_PRIVCFGR1	LPTIM2PRIV	DTSPRIV	Res.	Res.	Res.	Res.	DAC1PRIV	Res.	Res.	Res.	Res.	CRSPRIV	I3C1PRIV	I2C2PRIV	I2C1PRIV	Res.	Res.	USART3PRIV	USART2PRIV	SPI3PRIV	SPI2PRIV	IWDGPRIV	WWDGPRIV	Res.	Res.	Res.	TIM7PRIV	TIM6PRIV	Res.	Res.	TIM3PRIV	TIM2PRIV
	Reset value	0	0					0					0	0	0	0			0	0	0	0	0	0				0	0			0	0
0x024	GTZC1_TZSC_PRIVCFGR2	Res.	Res.	Res.	LPTIM1PRIV	Res.	Res.	LPUART1PRIV	Res.	Res.	Res.	Res.	Res.	USBPRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USART1PRIV	Res.	SPI1PRIV	TIM1PRIV	Res.	Res.	Res.	COMPPRIV	OPAMPPRIV	Res.	Res.	FDCAN1PRIV
	Reset value				0			0						0								0		0	0				0	0			0
0x028	GTZC1_TZSC_PRIVFGR3	Res.	Res.	Res.	Res.	Res.	RAMCFGPRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNGPRIV	HASHPRIV	Res.	Res.	ADC1PRIV	Res.	ICACHEPRIV	Res.	Res.	Res.	CRCPRIV	Res.	Res.	Res.	Res.	Res.	I3C2PRIV	Res.	Res.
	Reset value						0								0	0			0		0				0						0		
0x02C-0x06C	Reserved	Reserved																															
0x070	GTZC1_TZSC_MPCWM4ACFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRLOCK	SREN
	Reset value																							0								0	0
0x74	GTZC1_TZSC_MPCWM4AR	Res.	Res.	Res.	Res.	SUBA_LENGTH[11:0]												Res.	Res.	Res.	Res.	Res.	SUBA_START[10:0]										
	Reset value					1	0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0

请参见*表 15：GTZC1 子模块偏移地址*。

6.6 GTZC1 MPCBBz 寄存器（z = 1 和 2）

所有寄存器只能按字（32 位）进行访问。

6.6.1 GTZC1 SRAMz MPCBB 超级块 x 的特权配置寄存器 (GTZC1_MPCBBz_PRIVCFGRx)（z = 1 和 2）

GTZC1 SRAMz MPCBB privileged configuration for super-block x register

偏移地址：模块 1：0x200 + 0x04 * x，（x = 0 到 31）

偏移地址：模块 2：0x600 + 0x04 * x，（x = 0 到 31）

复位值：0x0000 0000

对该寄存器的写访问处于仅特权状态。允许任意读访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRIV31	PRIV30	PRIV29	PRIV28	PRIV27	PRIV26	PRIV25	PRIV24	PRIV23	PRIV22	PRIV21	PRIV20	PRIV19	PRIV18	PRIV17	PRIV16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRIV15	PRIV14	PRIV13	PRIV12	PRIV11	PRIV10	PRIV9	PRIV8	PRIV7	PRIV6	PRIV5	PRIV4	PRIV3	PRIV2	PRIV1	PRIV0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **PRIVy**：块 y 的特权配置，属于超级块 x (Privileged configuration for block y, belonging to super-block x) (y = 31 到 0)

0：对块 y 的特权和非特权访问，属于超级块 x

1：仅对块 y 进行特权访问，属于超级块 x

6.6.2 GTZC1 MPCBBz 寄存器映射（z = 1 和 2）

表 19. GTZC1 MPCBBz 寄存器映射和复位值（z = 1 和 2）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x200 + 0x04 *x (x = 0 to 31)	GTZC1_MPCBB1_PRIVCFGRx	PRIV31	PRIV30	PRIV29	PRIV28	PRIV27	PRIV26	PRIV25	PRIV24	PRIV23	PRIV22	PRIV21	PRIV20	PRIV19	PRIV18	PRIV17	PRIV16	PRIV15	PRIV14	PRIV13	PRIV12	PRIV11	PRIV10	PRIV9	PRIV8	PRIV7	PRIV6	PRIV5	PRIV4	PRIV3	PRIV2	PRIV1	PRIV0
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x600 + 0x04 *x (x = 0 to 31)	GTZC1_MPCBB2_PRIVCFGRx	PRIV31	PRIV30	PRIV29	PRIV28	PRIV27	PRIV26	PRIV25	PRIV24	PRIV23	PRIV22	PRIV21	PRIV20	PRIV19	PRIV18	PRIV17	PRIV16	PRIV15	PRIV14	PRIV13	PRIV12	PRIV11	PRIV10	PRIV9	PRIV8	PRIV7	PRIV6	PRIV5	PRIV4	PRIV3	PRIV2	PRIV1	PRIV0
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

请参见*表 15：GTZC1 子模块偏移地址*。

7 嵌入式 Flash (FLASH)

7.1 简介

嵌入式 Flash (FLASH) 用于管理任何主器件对 128 KB 嵌入式非易失性存储器的访问。它可以执行读取、编程和擦除操作以及错误校正，以及各种完整性和机密性保护机制。

嵌入式 Flash 可管理在上电复位时对非易失性用户选项字节的自动加载，并实施这些选项的动态更新。

嵌入式 Flash 还具有一次性可编程 (OTP) 区域和只读区域，由意法半导体在产品制造阶段配置。

7.2 FLASH 主要特性

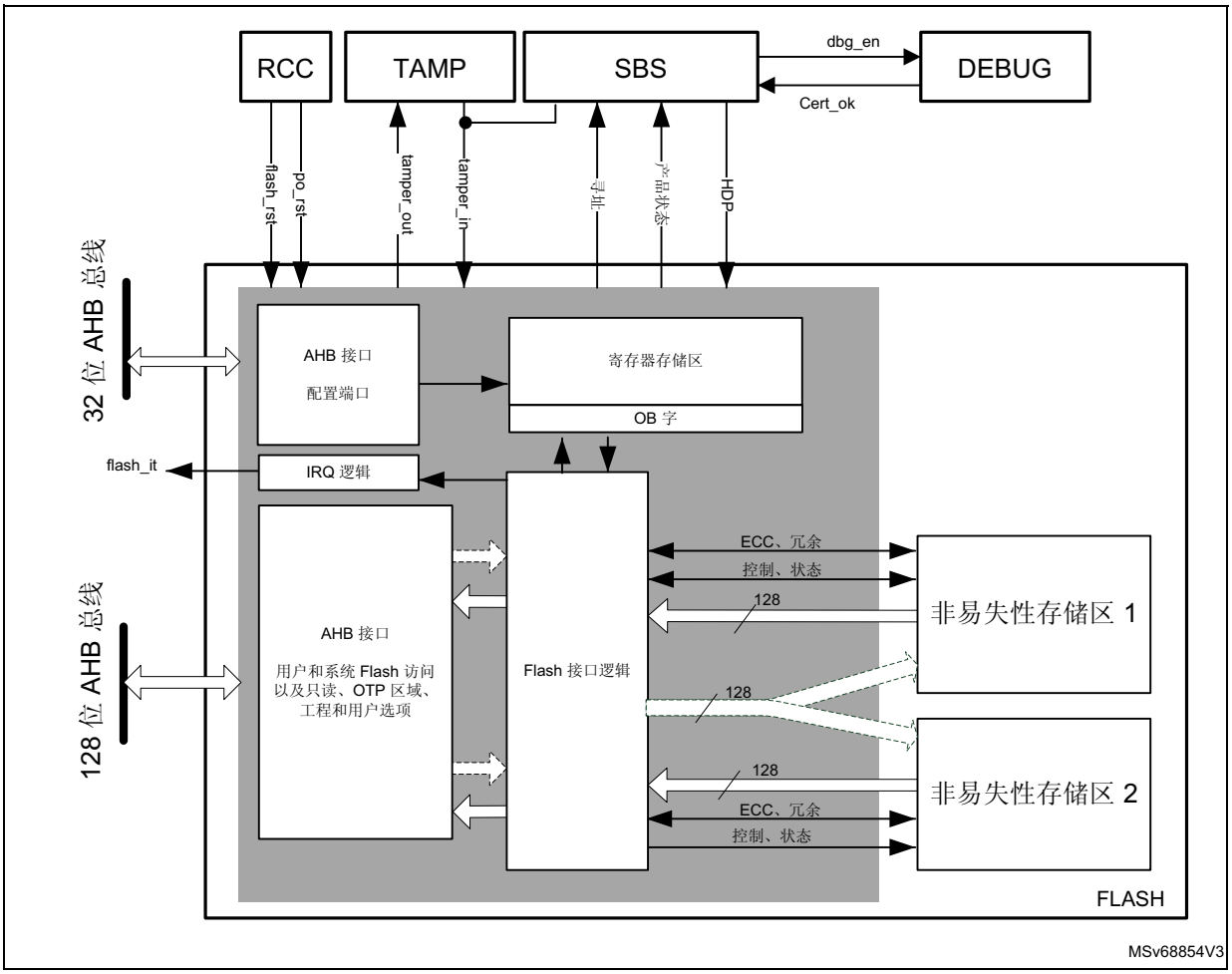
- 128 KB 非易失性存储器，划分为两个 64 KB 存储区
- 支持多种长度的 Flash 读操作：128 位、64 位、32 位、16 位或 1 字节
- 128 位（用户区域）和 16 位 (OTP) Flash 编程
- 8 KB 扇区擦除、存储区擦除和双存储区批量擦除
- 双存储区结构，支持：
  - 同时操作：支持同步读写（编程和擦除）。
  - 两个存储区共用同一个接口，不能并行执行写入和擦除操作（不支持同步写入）。
  - 存储区交换：每个存储区的用户 Flash 的地址映射及其相应的寄存器都可以进行交换。安全标志对物理存储区保持有效，因此交换到具有较低级别安全配置的存储区后不会泄露数据。
- 错误代码校正 (ECC)：使用 9 个 ECC 位对每个 128 位 Flash 字执行单个错误检测/校正或两个错误检测
- 用户可配置的非易失性选项字节
- Flash 增强保护，通过选项字节激活
  - 不同的产品状态，防止对存储器内容进行调试访问
  - 扇区写保护 (WRPSG)，可按扇区配置
  - HDP 保护，可为启动代码提供暂时隔离
- 2 KB 一次性可编程 (OTP) 区域
- 由意法半导体配置的只读区域
- 读取 Flash 中下一个连续存放的指令的预取操作

7.3 FLASH 功能说明

7.3.1 FLASH 框图

下图显示了嵌入式 Flash 框图。

图 12. Flash 框图（简化版）



7.3.2 FLASH 信号

Flash 具有两个 AHB 连接：Flash AHB 寄存器接口和主 AHB 接口。

Flash AHB 寄存器接口

- 数据大小为 32 位
- 除一些寄存器（FLASH_NSKEYR 和 FLASH_OPTKEYR 寄存器用于为控制和选项寄存器插入解锁序列，仅支持 32 位写操作）外，所有寄存器都支持 8 位、16 位和 32 位读写操作。
- 当控制和选项寄存器的解锁序列错误时，将生成总线错误，其他情况时总线上不会生成任何读或写错误

主 AHB 接口

AHB 数据总线大小为 128 位。

此接口用于处理三个不同的目标：

- 放置在用户和系统存储器中的代码（受 9 位 ECC 保护）
- OTP 区域和只读区域（受 6 位 ECC 保护）

主 AHB 接口的实现如下：

- 用户和系统存储器：
  - 支持多种长度：128 位、64 位、32 位、16 位和 8 位数据宽度
  - 每个存储区都有一个 128 位的读缓冲区，用于存储最后一次读取的数据。如果数据在读缓冲区中可用，则不会对 Flash 进行读访问。发生写访问、OTP 访问、用户选项更改请求或擦除操作时，将会刷新缓冲区
  - 预取操作的位数与读缓冲区的大小相同
  - 每个 128 位数据 Flash 字都与一个 9 位 ECC 相关联
- OTP 区域和只读区域
  - 两个 137 位的专用数据缓冲区，用于管理带 6 位 ECC 的 16 位数据
  - 两次读取同一地址会触发两次 Flash 读访问
  - 读访问期间，除存储器等待周期外，还会引入两个等待周期。这些等待周期是解析数据缓冲区所必需的
  - 每次写访问都会触发对 Flash 的写操作
  - 每个 16 位数据都与一个 6 位 ECC 相关联。

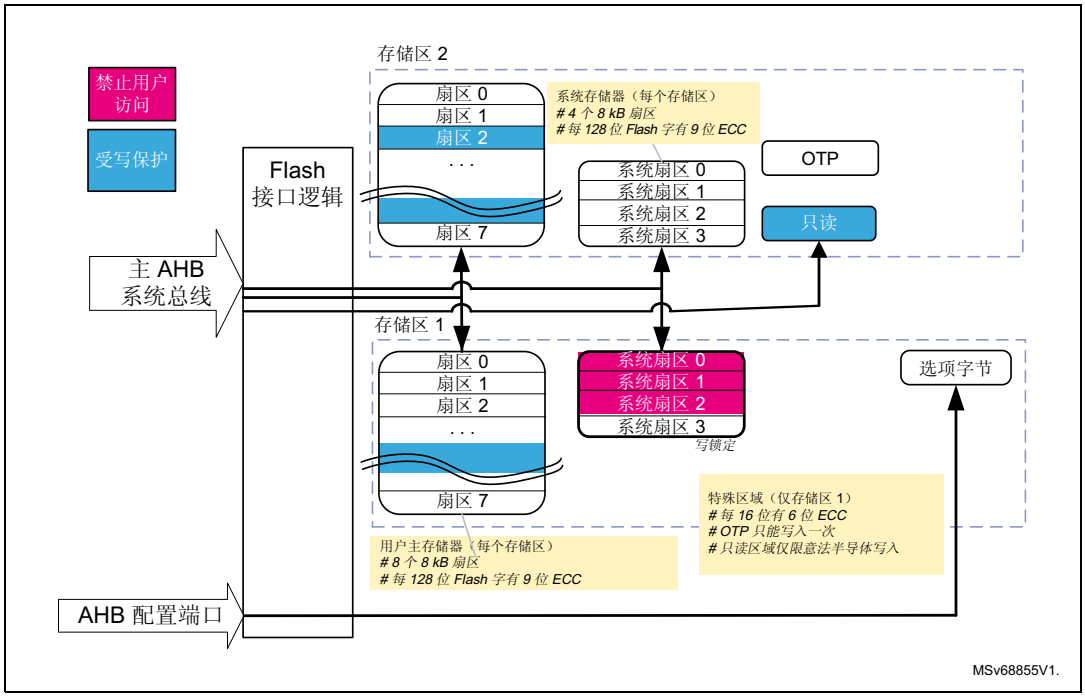
默认情况下，所有 AHB 存储器范围都可以缓存。对于无法缓存的区域（OTP、RO），必须使用 MPU 禁止本地可缓存性。

7.3.3 Flash 架构和使用

Flash 架构

下图显示了嵌入式 Flash 支持的非易失性存储器结构。

图 13. 嵌入式 Flash 结构



嵌入式 Flash 非易失性存储器由以下部分组成：

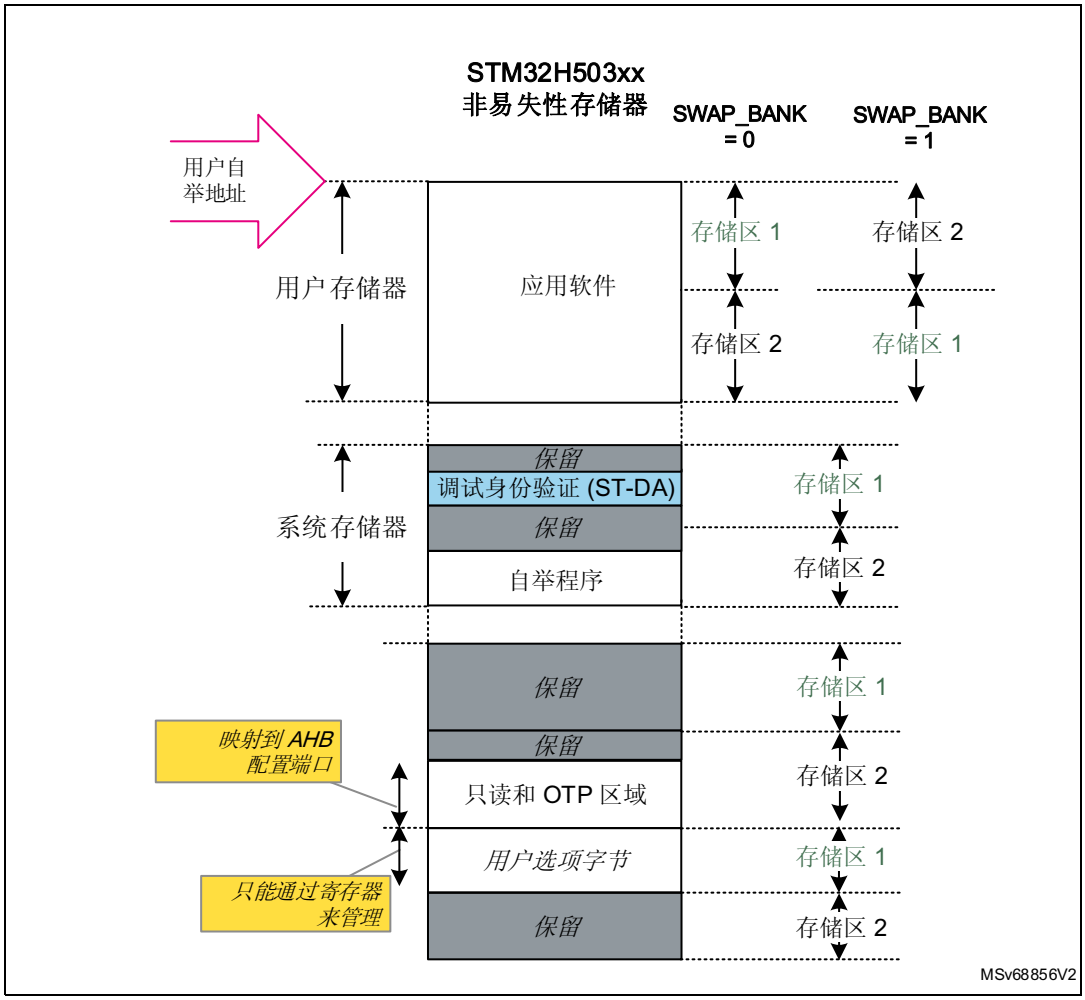
- 一个 128 KB 主存储器块，分为两个存储区，每个存储区 64 KB。每个存储区分为 8 个 8 KB 扇区，并且具有多行 128 位的 Flash 字，每个字有 9 位 ECC
- 一个 64 KB 系统存储器块，分为两个 32 KB 存储区。每个存储区分为四个 8 KB 扇区。系统 Flash 受 ECC 保护（每个 128 位字有 9 位 ECC）
- 一组非易失性选项字节，由嵌入式 Flash 在复位时加载，仅限应用软件通过 AHB 配置寄存器接口进行访问。
- 一个 2 KB 一次性可编程 (OTP) 区域，仅限应用软件执行一次写操作
- 一个 2 KB 只读区域。其中包含唯一的器件 ID 和产品信息

表 24 中汇总了整个 Flash 架构及其相应的访问接口。

分区使用

下图说明了意法半导体和应用软件如何使用嵌入式 Flash。

图 14. 嵌入式 Flash 的使用



对于不同的产品状态和其他选项字节设置，用户存储器和系统存储器的使用会有所不同：

- 用户存储器包含应用代码和数据、根安全服务 (RSS)、调试身份验证码 (ST-DA) 和 STM32 自举程序。发生复位时，内核将跳转到自举地址，此地址通过 BOOT 引脚、BOOT_ADDR 选项字节和产品状态配置
- 如果调试器与产品相连，则入口点是调试身份验证策略，用于通过 SBS 解锁器件（连接到调试器时）。必须提供数字签名以退回到允许调试的产品状态

注意： *有关自举选项字节设置的更多信息，请参见第 7.4.7 节。*

7.3.4 FLASH 读操作

读操作概述

对主 Flash 和系统 Flash 的读访问按如下方式进行：

- 每个存储区都有一个关联的 128 位读数据缓冲区，用于存储最后一次读取的数据。如果多个连续读访问请求的数据属于同一 Flash 数据字（128 位），则会直接从当前的数据读缓冲区中读取数据，而不会触发额外的 Flash 读操作。每次授予读访问权限时，都会运行此机制。当读访问因安全原因遭到拒绝时，嵌入式 Flash 会发出相应的读错误响应，并且不会触发对 Flash 的读操作
- 当发生写访问、OTP 访问、用户选项更改请求或其他擦除操作时，将会禁止读数据缓冲区

对 OTP 和 RO 的读访问按如下方式进行：

1. 读取 137 位 Flash 数据字并将其存储在临时缓冲区中
2. 接口解析此 137 位数据字并选择请求的 16 位或 32 位数据
3. 解析此 137 位数据字时，会引入两个等待周期，AHB 总线停止
4. 如果应用程序读取的 OTP 数据之前并未写入，则会报告双重 ECC 错误，并且只返回一个所有位都置 1 的字（有关详细信息，请参见*第 7.3.9 节：OTP 和 RO 存储器访问*）读取的数据（16 位）存储在 FLASH_ECCDR 寄存器中，以便用户识别双重 ECC 错误是由初始数据所致还是一个真正的 ECC 错误
5. 两次读取同一地址会触发两次 Flash 读访问
6. 对于 8 位访问，生成 AHB 总线错误。

注意： *嵌入式 Flash 可以在执行读操作的同时进行单错误校正和双重错误检测（请参见第 7.3.8 节：Flash 错误保护）。*

指令预取

Cortex-M33 通过 C 总线和指令缓存（若使能）获取指令和数据缓冲池（常量/数据）。预取块旨在使能指令缓存时，通过降低缓存重填延迟来提高 C 总线访问的效率。

对于顺序执行的代码，预取效率很高；Flash 中的预取功能允许从 Flash 中读取下一个顺序执行的指令行，同时当前指令行填入指令缓存并由 CPU 执行。

将 Flash 访问控制寄存器 (FLASH_ACR) 中的 PRFTEN 位置 1 可使能预取功能。只有当访问 Flash 至少需要一个等待周期时，PRFTEN 才必须置 1。

调整读时序限制

在从非易失性存储器读取数据之前，嵌入式 Flash 时钟必须使能并处于运行状态。

为了准确读取 Flash 数据，必须根据嵌入式 Flash 主 AHB 接口时钟频率和器件的内部电压范围 (V_{core})，在 Flash 访问控制寄存器 (FLASH_ACR) 中正确地编程等待周期数 (LATENCY)。

下表显示了等待周期数 (LATENCY)、编程延迟参数 (WRHIGHFREQ)、嵌入式 Flash 时钟频率和电源电压范围之间的对应关系。

表 20. FLASH 推荐的等待周期数和编程延迟

等待周期数 (LATENCY)	编程延迟 (WRHIGH FREQ)	接口时钟频率与V_{CORE} 范围			
		VOS3 范围 0.95 V - 1.05 V	VOS2 范围 1.05 V - 1.15 V	VOS1 范围 1.15 V - 1.26 V	VOS0 范围 1.30 V - 1.40 V
0 WS（1 个 FLASH 时钟周期）	00	[0 MHz; 20 MHz]	[0 MHz; 30 MHz]	[0 MHz; 34 MHz]	[0 MHz; 42 MHz]
1 WS（2 个 FLASH 时钟周期）		[20 MHz; 40 MHz]	[30 MHz; 60 MHz]	[34 MHz; 68 MHz]	[42 MHz; 84 MHz]
2 WS（3 个 FLASH 时钟周期）	01	[40 MHz; 60 MHz]	[60 MHz; 90 MHz]	[68 MHz; 102 MHz]	[84 MHz; 126 MHz]
3 WS（4 个 FLASH 时钟周期）		[60 MHz; 80 MHz]	[90 MHz; 120 MHz]	[102 MHz; 136 MHz]	[126 MHz; 168 MHz]
4 WS（5 个 FLASH 时钟周期）	10	[80 MHz; 100 MHz]	[120 MHz; 150 MHz]	[136 MHz; 170 MHz]	[168 MHz; 210 MHz]
5 WS（6 个 FLASH 时钟周期）		N/A	N/A	[170 MHz; 200 MHz]	[210 MHz; 250 MHz]

注意： *不支持 1.26 V 到 1.30 V 的电压范围。*

调整系统频率

上电后，嵌入式 Flash 的时钟由 32 MHz 高速内部振荡器 (HSI) 提供，电压范围设置为 VOS3 的电压调节值。因此，FLASH_ACR 寄存器中指定了 3 个等待周期的保守延迟（请参见上表）。

更改总线频率时，应用软件必须遵循下述步骤来调整访问非易失性存储器所需的等待周期数。

要提高 CPU 频率：

1. 必要时，将 FLASH_ACR 寄存器中的 LATENCY 和 WRHIGHFREQ 位编程为合适的值，如*表 20* 中所述。
2. 通过回读 FLASH_ACR 寄存器，检查新的等待周期数是否设置成功。
3. 在复位和时钟控制器 (RCC) 的 RCC_CFGR 寄存器中修改嵌入式 Flash 时钟源和/或时钟预分频值。
4. 通过从复位和时钟控制器 (RCC) 的 RCC_CFGR 寄存器中回读嵌入式 Flash 时钟源状态和/或预分频值，检查新的嵌入式 Flash 存储器时钟源和/或新的 AHB 时钟预分频值是否设置成功。

要降低 CPU 频率：

1. 在复位和时钟控制器 (RCC) 的 RCC_CFGR 寄存器中修改嵌入式 Flash 时钟源和/或时钟预分频值。
2. 通过从复位和时钟控制器 (RCC) 的 RCC_CFGR 寄存器中回读嵌入式 Flash 时钟源状态和/或 AHB 接口预分频值，检查嵌入式 Flash 存储器新的时钟源和/或新的时钟预分频值是否设置成功。
3. 必要时，将 FLASH_ACR 寄存器中的 LATENCY 和 WRHIGHFREQ 位编程为合适的值，如*表 20* 中所述。
4. 通过回读 FLASH_ACR 寄存器，检查新的等待周期数是否设置成功。

错误代码校正 (ECC)

嵌入式 Flash 内置了一个错误校正机制。对于每个读操作，都会执行单错误校正和双重错误检测。更多详细信息，请参见*第 7.3.8 节：Flash 错误保护*。

读错误

当 ECC 机制无法校正读操作时，嵌入式 Flash 会报告读错误（如*第 7.8.6 节：错误校正代码错误 (ECCC/ECCD)* 中所述）。

读中断

有关详细信息，请参见*第 7.9 节：FLASH 中断*。

7.3.5 FLASH 编程操作

编程操作概述

编程操作包括发出写命令。嵌入式 Flash 支持一次仅执行一个写命令。不支持同步写入操作。不会阻止重写非初始 Flash 字，但不建议这样做，否则可能会导致数据无效和 ECC 代码不一致。

用户 Flash 和系统 Flash 扇区：

对于用户 Flash 和系统 Flash，每个 128 位数据 Flash 字都与一个 9 位 ECC 相关联。在这种情况下，嵌入式 Flash 必须始终以 128 位字粒度对非易失性存储器执行写操作。当写缓冲区（128 位）已满后，忙标志会置 1 并触发编程操作。

存储区 1 和存储区 2 共用一个写缓冲区，支持多种写访问类型（128 位、64 位、32 位、16 位或 8 位）。应用程序可以将最少 8 位数据写入 128 位 Flash 字。此时，将使用对 128 位 + ECC 的强制写入机制（请参见 FLASH_NSCR 寄存器的 NSFW 位）。

当向存储器发出写请求后，任何新的写请求都会导致主 AHB 总线阻塞。

此外，当写操作正在进行时，对同一存储区的任何新的读请求都会导致主 AHB 总线阻塞。

OTP 和 RO：

当目标存储器是 OTP 和 RO 扇区时，每个 16 位数据 Flash 字都与一个 6 位 ECC 代码相关联。嵌入式 Flash 支持 16 位或 32 位写操作（不支持 8 位写操作）。对于 8 位访问，写访问将被忽略。没有写数据缓冲区。每次写访问都会触发对 Flash 的写操作。

注意： *OTP 区域通常在最终产品中受写保护，如第 7.3.9 节：OTP 和 RO 存储器访问中所述。*

写保护检查在收到写请求时（地址阶段）执行。写缓冲区输出时不会再检查写保护。如果检测到写保护违例，则会取消写操作，并且 FLASH_NSSR 寄存器中的写保护错误标志 (WRPERR) 会置 1。

有关主 Flash、ICP 和 OTP 的写保护，请参见第 7.5 节。

监视正在执行的写操作

应用软件可以使用 FLASH_NSSR 中的状态标志来监视正在执行的写操作。由于一次只能执行一个操作，因此该标志用于指示当前是否正在进行任何操作（写入、擦除或选项更改）；无论哪个存储区。

- **BSY**：该位指示正在对非易失性存储器进行有效的写入、擦除或选项字节更改操作。此标志不专用于特定的存储区。无论哪个存储区，只要对存储器启动操作，此标志就会置 1。具体由以下操作触发：
  - 擦除 (FLASH_NSCR.STRT)
  - 写入（FLASH_NSCR.PG + AHB 写操作）
  - 选项修改 (FLASH_OPTCR.OPTSTRT)

这些位会在当前操作结束或出现错误时清零。

- **NSWBNE**：该位指示嵌入式 Flash 正在等待新数据填满 128 位写缓冲区。在此状态下，写缓冲区不为空。只要应用软件填满写缓冲区，或者使用 FLASH_NSCR 中的 NSFW 位强制写入，又或者检测到错误，该位就会立即复位。如果 NSWBNE 为 1，则无法启动对 Flash 的擦除和选项修改操作。
- **DBNE**：该位指示用于解析 16 位数据的数据缓冲区不为空：
  - 接收到 16 位数据写访问，并且正在填充数据缓冲区。该位在接收到有效写访问后置 1，在处理完写请求准备后立即复位。

注意： *如果 Flash 在接收到 AHB 写请求时处于繁忙状态，则 CPU 将停止执行操作。*

使能写操作

在对用户 Flash 的存储区 1 或存储区 2 进行编程之前，应用软件必须确保 FLASH_NSCR 中的 PG 位置 1。如果该位为 0，则必须使用解锁序列（请参见 *第 7.5.5 节：Flash 配置保护*），并且必须将 PG 位置 1。

必须修改选项字节时，应用软件必须确保 FLASH_OPTCR 已解锁。如果未解锁，则必须使用解锁序列（请参见 *第 7.5.5 节：Flash 配置保护*）。

注意： *应用软件不得解锁已解锁的寄存器，否则该寄存器将保持锁定状态，直到下一次系统复位。*

如果需要，应用软件可以更新编程延迟，如 *调整编程时序限制*中所述。

写入 FLASH 控制寄存器 FLASH_NSCR 和 FLASH_OPTCR

如果 BSY 位置 1，则无法在写模式下访问 FLASH_NSCR 和 FLASH_OPTCR 寄存器。在 BSY 位置 1 时，尝试对上述寄存器进行写操作会导致 AHB 总线阻塞，直到 BSY 位清零。

单次写操作顺序

建议按照以下顺序执行单次写操作：

1. 确保保护机制不会阻止编程操作
2. 检查 FLASH_NSSR 寄存器中的 BSY 位和 FLASH_NSSR 寄存器中的 DBNE 位，以确认当前是否未执行任何 Flash 操作。检查 FLASH_NSSR 寄存器中的 NSWBNE 位，以确认写缓冲区是否为空
3. 检查并清零之前的编程/擦除操作所导致的全部错误标志
4. 解锁 FLASH_NSCR 寄存器，如 *第 7.5.5 节：Flash 配置保护*中所述（仅在寄存器尚未解锁时）
5. 将 FLASH_NSCR 寄存器中的 PG 位置 1 以使能写操作
6. 在对齐地址处写入一个 Flash 字

注意： *NSWBNE 标志用于指示 128 位写缓冲区是否正在等待新数据。*

注意： *在第一次写入到 Flash 写操作完成的过程中，不允许任何擦除请求和选项更改请求。*

7. 等待相应 FLASH_NSSR 寄存器中的 BSY 位清零
8. 如果不再有编程请求，则将 FLASH_NSCR 寄存器中的 PG 位清零

如果逐步执行第 6 步（例如逐字节执行），则写缓冲区会部分填充。在这种情况下，应用软件可以使用 FLASH_NSCR 寄存器中的 NSFW 位来强制写入存储在写缓冲区中的内容。在这种特殊情况下，未写入的位会自动设为 1。如果写缓冲区中没有任何位清零，则 NSFW 位不起作用。

注意： *使用强制写入操作可以防止应用程序稍后使用不等于 1 的值更新丢失的位，否则可能会导致出现意外或不一致的数据或 ECC。*

调整编程时序限制

编程操作时序限制取决于直接影响性能的嵌入式 Flash 时钟频率。如果时序限制过严，非易失性存储器将无法正常工作，如果限制过松，编程速度不会达到最佳水平。
因此，用户必须通过 FLASH_ACR 寄存器中的 WRHIGHFREQ 参数调整最佳编程延时。有关嵌入式 Flash 时钟频率对应的建议编程延时，请参见*第 7.3.4 节：FLASH 读操作*中的*表 20*。
FLASH_ACR 配置寄存器由两个存储区共用。

在修改 WRHIGHFREQ 参数之前，应用软件必须确认当前未执行任何编程/擦除操作。

小心： 如果在编程/擦除 Flash 时修改 WRHIGHFREQ，可能会损坏 Flash 内容。

编程错误

编程操作失败时，可以报错，如*第 7.8 节：FLASH 错误管理*中所述。

编程中断

有关详细信息，请参见*第 7.9 节：FLASH 中断*。

7.3.6 FLASH 擦除操作

擦除操作概述

嵌入式 Flash 可以对 8 KB 用户扇区或一个用户 Flash 存储区执行擦除操作，甚至还可以对两个 Flash 存储区执行擦除操作（例如批量擦除）。

有关用户 Flash、用户选项和 OTP 擦除保护的更多详细信息，请参见第 7.5 节：FLASH 安全性和保护。

擦除命令通过 AHB 配置接口发出。由于嵌入式 Flash 仅支持一次执行一个操作，因此如果同时接收到写请求和擦除请求，则错误标志将会置 1，并且会取消这两个操作。有关详细信息，请参见*第 7.8 节：FLASH 错误管理*。

擦除和 WRP

如果应用软件尝试擦除受写保护的用户扇区，则扇区擦除操作将中止，并且 FLASH_NSSR 寄存器中的 WRPERR 标志将置1，如*第 7.8.2 节：写保护错误 (WRPERR)* 中所述。

Flash 繁忙

有关忙信号的说明，请参见*第 7.3.5 节：FLASH 编程操作*的*监视正在执行的写操作*一节中。

写入 FLASH 控制寄存器 FLASH_NSCR 和 FLASH_OPTCR

请参见*第 7.3.5 节：FLASH 编程操作*中的*写入 FLASH 控制寄存器 FLASH_NSCR 和 FLASH_OPTCR*。

使能擦除操作

擦除扇区之前，应用软件必须确保 FLASH_NSCR 已解锁。如果未解锁，则必须使用解锁序列（请参见*第 7.5.5 节：Flash 配置保护*）。

注意： *应用软件不得解锁已解锁的寄存器，否则该寄存器将保持锁定状态，直到下一次系统复位。这可用于特意锁定寄存器，以防遭到进一步访问。*

类似的限制也适用于存储区擦除请求。

Flash 扇区擦除顺序

要擦除 8 KB 用户扇区，请按以下顺序操作：

1. 确保保护机制不会阻止扇区擦除操作（WRP、HDP）。
2. 检查 FLASH_NSSR 寄存器中的 BSY 和 DBNE 位，以确认当前是否未执行任何 Flash 操作；检查 FLASH_NSSR 寄存器中的 WBNE 位，以确认写缓冲区是否为空。
3. 检查并清零之前的编程/擦除操作所导致的全部错误标志。有关详细信息，请参见*第 7.8 节：FLASH 错误管理*
4. 解锁 FLASH_NSCR 寄存器，如*第 7.5.5 节：Flash 配置保护*中所述（仅在寄存器尚未解锁时）
5. 设置 FLASH_NSCR 寄存器中的 BKSEL 位、SER 位和 SNB 位域。BKSEL 用于指示必须擦除的物理存储区扇区，SER 用于指示扇区擦除操作，而 SNB 则包含目标扇区编号。
6. 将 FLASH_NSCR 寄存器中的 STRT 位置 1。
7. 等待 FLASH_NSSR 寄存器中的 BSY 位清零。
8. STRT 位在扇区擦除操作结束或出现错误时自动清零。
9. 如果不再发出扇区擦除请求，则将 FLASH_NSCR SER 寄存器中的 SER 清零。

注意： *如果在执行扇区擦除操作的同时请求另一个擦除标志，则会生成 PGSERR 错误。*

标准 Flash 存储区擦除顺序

要擦除存储区：

1. 确保保护机制不会阻止扇区擦除操作。
2. 检查 FLASH_NSSR 寄存器中的 BSY 和 DBNE 位，以确认当前是否未执行任何 Flash 操作；检查 FLASH_NSSR 寄存器中的 WBNE 位，以确认写缓冲区是否为空。
3. 检查并清零之前的编程/擦除操作所导致的全部错误标志。有关详细信息，请参见*第 7.8 节：FLASH 错误管理*。
4. 解锁 FLASH_NSCR 寄存器，如*第 7.5.5 节：Flash 配置保护*中所述（仅在寄存器尚未解锁时）。
5. 设置 FLASH_NSCR 寄存器中的 BKSEL 位和 BER 位，以指示目标物理存储区（忽略交换设置）。
6. 将 FLASH_NSCR 寄存器中的 STRT 位置 1 以启动存储区擦除操作。然后等待，直到 FLASH_NSSR 寄存器中的 BSY 位清零。
7. STRT 位在擦除序列结束或出现错误时自动清零。
8. 如果不再发出其他存储区擦除请求，则将 FLASH_NSCR 寄存器中的 BER 清零。

Flash 批量擦除顺序

应用软件可以将 FLASH_NSCR 寄存器中的 MER 位置 1，如下所述：

1. 确保保护机制不会阻止批量擦除操作（*第 7.5 节*）
2. 检查 FLASH_NSSR 寄存器中的 BSY 和 DBNE 位，以确认当前是否未执行任何 Flash 操作；检查 FLASH_NSSR 寄存器中的 WBNE 位，以确认写缓冲区是否为空
3. 检查并清零之前的编程/擦除操作所导致的全部错误标志。有关详细信息，请参见*第 7.8 节：FLASH 错误管理*
4. 解锁 FLASH_NSCR 寄存器，如*第 7.5.5 节：Flash 配置保护*中所述（仅在寄存器尚未解锁时）
5. 将 FLASH_NSCR 寄存器中的 MER 位置 1
6. 将 FLASH_NSCR 寄存器中的 STRT 位置 1。然后等待，直到 FLASH_NSSR 寄存器中的 BSY 位清零
7. STRT 位在擦除序列结束或出现错误时自动清零
8. 将 FLASH_NSCR 寄存器中的 MER 清零

7.3.7 FLASH 并行操作

由于非易失性存储器分为两个独立的存储区，但它们封装在同一个宏内，因此嵌入式 Flash 接口仅支持在一个存储区中执行读操作的同时在另一个存储区中执行写操作或擦除操作。此功能称为同步读写 (RWW)。不支持同步写入或同步读取功能。

在所有情况下，*第 7.3.4 节：FLASH 读操作*、*第 7.3.5 节：FLASH 编程操作*和*第 7.3.6 节：FLASH 擦除操作*中介绍的顺序都适用。

7.3.8 Flash 错误保护

错误校正代码 (ECC)

嵌入式 Flash 支持错误校正代码 (ECC) 机制。它基于 SECDED 算法，可以校正单错误和检测双重错误。

此机制针对每个 128 位 Flash 字使用 9 个 ECC 位，适用于用户存储器和系统存储器。对于只读区域和 OTP 区域，此机制有所增强，针对每个 16 位字使用 6 个 ECC 位。对于 OTP 初始字（例如，22 个位为 1 的字），将生成双重 ECC 错误。当此 OTP 字不再是初始字时，ECC 错误将会消失。

更具体地说，每次对 128 位 Flash 字执行读操作期间，嵌入式 Flash 都会检索 9 位 ECC 信息并计算 Flash 字的 ECC，然后将结果与参考值进行比较。如果它们不匹配，则会生成相应的 ECC 错误，如*第 7.8.6 节：错误校正代码错误 (ECCC/ECCD)* 中所述。

在每次编程操作期间，每个 128 位数据 Flash 字都与一个 9 位 ECC 代码相关联，二者组成的 137 位 Flash 字信息会写入非易失性存储器。

类似的机制也适用于只读区域和 OTP 区域，只是针对每个 16 位数据使用 6 位 ECC。

7.3.9 OTP 和 RO 存储器访问

OTP 和 RO 存储器通过主 AHB 接口进行访问。OTP 可在 0x08FF_F000 到 0x08FF F7FF 地址范围内进行访问，而只读部分可在 0x08FF F800 到 0x08FF_FFFF 地址范围内进行访问。

FLASH 一次性可编程区域

嵌入式 Flash 提供了一个 2048 字节的存储区域，专用于应用非机密一次性可编程数据 (OTP)。此区域由 1024 个 16 位字（加 6 位 ECC）组成。不能擦除，并且只能写入一次。OTP 区域可通过主 AHB 接口访问，地址范围为 0x08FF F000 到 0x08FF F7FE（最后一个完整的可读 16 位字）。

应用软件可以 16 位块的形式编程 OTP 数据。覆盖已编程的 16 位半字会导致数据和 ECC 错误，因此不支持该操作。

注意： *意法半导体交付器件时，OTP 区域处于初始状态。*

如果在读取 OTP 数据时校正了单错误或检测到双重错误，则嵌入式 Flash 会报告读错误，如*第 7.8.6 节：错误校正代码错误 (ECCC/ECCD)* 中所述。

在读取非应用软件写入的 OTP 数据（如初始 OTP）时，ECC 校正会报告检测到双重错误 (ECCD)，并且数据将位于 FLASH_ECCDR 寄存器中。ECCD 意味着生成 NMI，除非 SBS 中的设置不支持生成 NMI。

OTP 写保护

OTP 数据包含 32 个块，每块 32 个 OTP 字，如*表 21* 所示。将 FLASH_OTPBLR 选项字节寄存器中对应于每个 OTP 块（i = 0 到 31）的 LOCKBLi（i = 0 到 31）位置 1，可以对整个 OTP 块实施写访问保护（锁定）。无论块是否已编程（甚至部分编程），都可以对其进行写保护。

OTP 块锁定操作是不可逆的，与产品生命周期状态无关。

注意： *OTP 区域只能在读模式下访问。*

表 21. Flash OTP 结构

OTP 块	AHB 地址	AHB 字		锁定位
		[31:16]	[15:0]	
块 0	0x08FF F000	OTP001	OTP000	LOCKBL0
	0x08FF F004	OTP003	OTP002	
	...			
	0x08FF F03C	OTP031	OTP030	
块 1	0x08FF F040	OTP033	OTP032	LOCKBL1
	0x08FF F044	OTP035	OTP034	
	...			
	0x08FF F07C	OTP063	OTP062	

表 21. Flash OTP 结构（续）

OTP 块	AHB 地址	AHB 字		锁定位
		[31:16]	[15:0]	
块 2	0x08FF F080	OTP065	OTP064	LOCKBL2
	0x08FF F084	OTP067	OTP066	
	...			
	0x08FF F0BC	OTP95	OTP94	
...				
块 31	0x08FF F7C0	OTP993	OTP992	LOCKBL31
	0x08FF F7C4	OTP995	OTP994	
	...			
	0x08FF F7FC	OTP1023	OTP1022	

OTP 写操作顺序

请按以下顺序写入 OTP 字：

1. 检查 FLASH_NSSR 寄存器中的 BSY 位，以确认当前是否未执行任何 Flash 操作；检查 FLASH_NSSR 寄存器中的 DBNE 位，以确认数据缓冲区是否为空
2. 检查并清零之前的编程/擦除操作所导致的全部错误标志。
3. 将 FLASH_NSCR 寄存器中的 PG 位置 1
4. 检查目标 OTP 字的保护状态（请参见*表 21*）。相应的 LOCKBLi 位不得置 1
5. 写入对应于*表 21* 中所示的 4 字节对齐地址的两个 OTP 字（32 位）。或者，应用软件也可以单独编程 16 位 MSB 或 16 位 LSB。在这种情况下，第一个 16 位写操作会立即启动，而无需等待第二个操作
6. 等待 FLASH_NSSR 寄存器中的 BSY 位清零
7. 如果存储区中不再有编程请求，则将 FLASH_NSCR 寄存器中的 PG 位清零
8. （可选）使用 LOCKBLi 锁定 OTP 块以防止进一步更改数据。

注意： *不要写入两次 OTP 16 位字，否则会生成 ECC 错误。*

不支持按字节写入 OTP 数据，会生成总线错误。

为避免数据损坏，务必在开始选项更改操作之前完成 OTP 写过程（例如，通过回读 OTP 值）。

Flash 只读区域

嵌入式 Flash 提供了一个 2 KB 存储区来存储只读数据。此区域映射如*表 22：只读公共数据结构*中所述。可以通过 AHB 主端口访问此区域。ECC 方案可为此只读区域提供稳定的保护，如*第 7.3.8 节：Flash 错误保护*中所述。

下表介绍了应用软件可使用的只读信息。此信息由意法半导体编程设定。

表 22. 只读公共数据结构

只读数据名称	地址	备注
唯一器件 ID	0x08FF F800	U_ID[31:0]
	0x08FF F804	U_ID[63:32]
	0x08FF F808	U_ID[96:64]
Flash 大小/封装	0x08FF F80C	Flash 大小[15:0] || 封装代码[15:0]
保留	0x08FF F810 到 0x08FF FFFF	保留信息

7.3.10 Flash 存储区交换

对于用户 Flash，嵌入式 Flash 存储区 1 和存储区 2 可以进行交换。固件升级后可使用此功能使器件以新固件重启。存储区交换是一个用户选项字节标志，由 FLASH_OPTCR 寄存器的 SWAP_BANK 位控制。

关于数据区域和安全属性的存储区特定设置遵循原始存储区及其内容。控制位始终作用于物理存储区，而非 SWAP_BANK 设置。

下表显示了根据 SWAP_BANK 位配置可以访问的存储器映射。

表 24. 存储器映射和交换选项

Flash 区域	Flash 相应存储区		起始地址	结束地址	大小（字节）	区域名称
	SWAP_BANK=0	SWAP_BANK=1				
用户主存储器	存储区 1	存储区 2	0x0800 0000	0x0800 1FFF	8 K	扇区 0
			0x0800 2000	0x0800 3FFF	8 K	扇区 1
			...	...	...	...
			0x0800 E000	0x0800 FFFF	8 K	扇区 7
	存储区 2	存储区 1	0x0801 0000	0x0801 1FFF	8 K	扇区 0
			0x0801 2000	0x0801 3FFF	8 K	扇区 1
			...	...	...	...
			0x0801 E000	0x0801 FFFF	8 K	扇区 7
系统存储器	存储区 1		0x0BF8 0000	0x0BF8 1FFF	8 K	系统 1 扇区 0
			0x0BF8 2000	0x0BF8 3FFF	8 K	系统 1 扇区 1
			...	...	...	...
			0x0BF8 6000	0x0BF8 7FFF	8 K	系统 1 扇区 3
	存储区 2		0x0BF9 8000	0x0BF9 9FFF	8 K	系统 2 扇区 0
			0x0BF9 A000	0x0BF9 BFFF	8 K	系统 2 扇区 1
			...	...	...	...
			0x0BF8 E000	0x0BF9 FFFF	8 K	系统 2 扇区 3

只有在系统复位或 POR 后，FLASH_OPTCR 寄存器中的 SWAP_BANK 位才能从 SWAP_BANK 选项位进行加载。

要更改 SWAP_BANK 位（例如，应用新的固件更新），需遵循以下顺序：

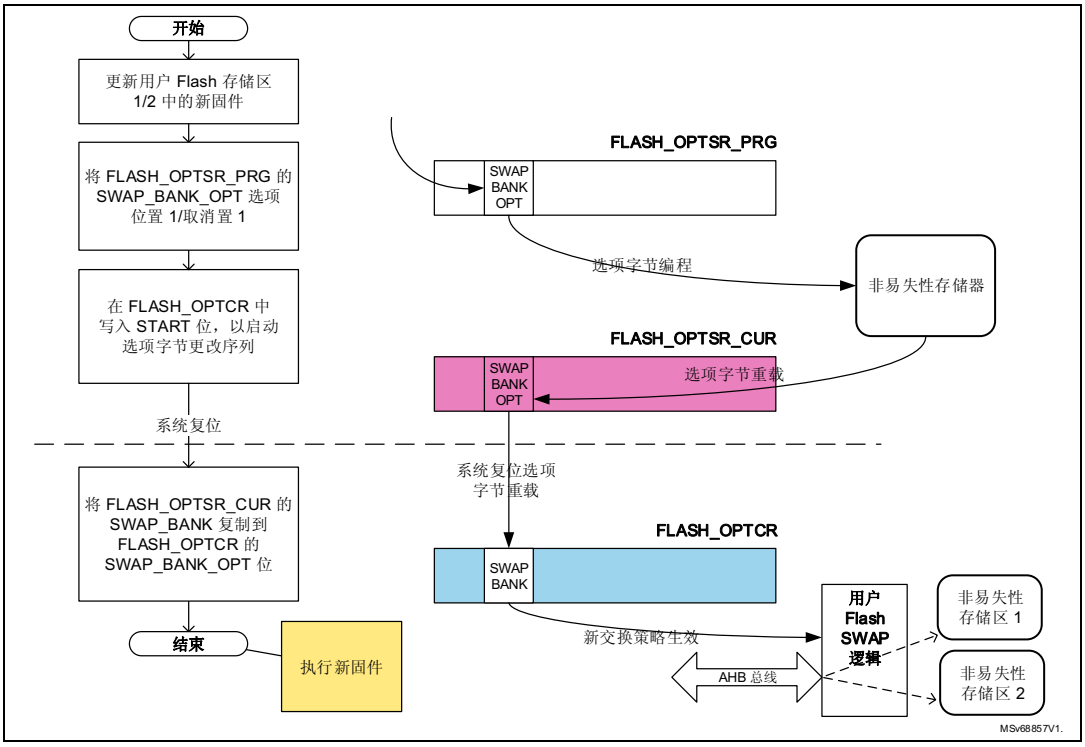
1. 检查 FLASH_NSSR 寄存器中的 BSY 和 DBNE 位，以确认当前是否未执行任何 Flash 操作；检查 FLASH_NSSR 寄存器中的 NSWBNE 位，以确认写缓冲区是否为空
2. 清零之前的操作所导致的全部错误标志
3. 解锁 OPTLOCK 位（如果尚未解锁）
4. 根据需要在 FLASH_OPTSR_PRG 寄存器中设置新的 SWAP_BANK 值
5. 将 FLASH_OPTCR 寄存器中的 OPTSTRT 位置 1，开始执行选项字节更改序列
6. 选项字节更改完成后，FLASH_OPTSR_CUR 中会包含预期的 SWAP_BANK 值，但 FLASH_OPTCR 中的 SWAP_BANK 位尚未修改，存储区交换尚未生效
7. 强制进行系统复位或 POR。发生复位后，存储区交换生效（FLASH_OPTCR 中的 SWAP_BANK 值已更新），并且应执行新的固件。

注意： *FLASH_OPTCR 中的 SWAP_BANK 位为只读位，不能由应用软件进行修改。*

无论产品状态如何，都可以修改 FLASH_OPTSR_PRG 中的 SWAP_BANK 选项位。该位由 NSBOOT_LOCK 用户 OB 锁定，而非由 PRODUCT_STATE 锁定。

下图简要说明了存储区交换顺序。

图 15. Flash 存储区交换顺序



7.3.11 FLASH 复位和时钟

复位管理

嵌入式 Flash 可以通过内核域复位来复位，具体由复位和时钟控制 (RCC) 进行驱动。此复位的主要影响如下：

- 除选项字节寄存器外的所有寄存器都清零，包括读写延迟。如果存储区交换选项发生更改，则应用新设置。
- 大多数控制寄存器会自动获得保护，以防遭到写操作。要取消对这些寄存器的保护，必须使用新的解锁序列，如*第 7.5.5 节：Flash 配置保护*中所述。

嵌入式 Flash 可以通过上电内核域复位来复位，具体由复位和时钟控制 (RCC) 进行驱动。发生复位时，所有选项字节寄存器都会复位。发生复位后，将加载选项字节，并可能应用新功能。在这个加载过程中，器件保持在复位状态下，嵌入式 Flash 不可访问。

Flash 操作期间发生复位

如果在 Flash 操作（编程、擦除或选项更改）期间发生复位，则无法保证 Flash 中的内容。用户必须重新启动操作来确保 Flash 的完整性。状态寄存器 FLASH_OPSR 用于指示 Flash 操作是否已被复位中断。

FLASH_OPSR.CODE_OP 用于提供操作的操作码。下表说明了如何使用 FLASH_OPSR 以及需要执行的操作。

表 25. 针对 FLASH_OPSR 内容推荐的反应

CODE_OP	中断的操作	OTP_OP	SYSF_OP	BK_OP	Flash 区域	ADDR_OP 最小值	ADDR_OP 最大值	推荐操作
0x000	复位时未在进行 Flash 操作	0	0	0	-(1)	-	-	无需额外操作
0x001	写操作	0	0	0/1	用户 Flash	0x0000	0xFFFF	擦除扇区并重写
		1	0	0	OTP	0x0600	0x07FF	
0x011	扇区擦除(2)	0	0	0/1	用户 Flash	0x0000	0xFFFF	重新启动扇区擦除
0x100	存储区擦除	0	0	0/1	用户 Flash	-	-	重新启动存储区擦除
0x101	批量擦除	0	0	0	用户 Flash	-	-	重新启动批量擦除
0x110	选项更改	0	0	0	用户配置	-	-	重新尝试更改选项

1. 破折号代表“无关”。
2. 指示的地址与扇区起始地址对齐，数据区域扇区通过向相应的用户 Flash 扇区发出擦除请求来擦除。

时钟管理

嵌入式 Flash 使用微控制器系统时钟 (sys_ck)，这里使用 AHB 接口时钟。

7.4 FLASH 选项字节

7.4.1 关于选项字节

嵌入式 Flash 包括一组非易失性选项字节。它们在上电复位时加载，只能通过配置寄存器读取和修改。本节内容如下：

- 何时加载选项字节
- 应用软件如何修改选项字节
- 选项字节及其初始值（首次更改选项字节前，用户默认配置）的详细列表。

7.4.2 选项字节加载

有多种方法可以将选项字节加载到嵌入式 Flash 中：

1. **上电唤醒**

   器件首次上电时，嵌入式 Flash 会自动加载所有选项字节。在选项字节的加载过程中，器件保持在复位状态下，嵌入式 Flash 不可访问。

2. **从系统待机模式唤醒**

   当包含嵌入式 Flash 的内核电源域从待机模式切换到运行模式时，嵌入式 Flash 的行为与上电序列期间相同。**在加载期间，器件不处于复位状态，这一点与上电序列期间不同。**

3. **应用程序重载专用选项字节**

   在用户应用程序通过嵌入式 Flash 寄存器成功修改选项字节的内容后，非易失性选项字节将经过编程，嵌入式 Flash 将自动重载所有选项字节以更新选项寄存器。

注意： *选项字节读序列受错误校正代码保护。*

出现错误时，将使用默认值加载选项字节（请参见第 7.4.3 节：选项字节修改）。此值与初始值（用户默认配置）不同，通常限制更多。

7.4.3 选项字节修改

更改用户选项字节

用户选项字节更改操作可用于修改保存在非易失性选项字节区域中的配置和保护设置。

尝试更改用户选项字节时，需强制遵守多项规则，*第 7.4.8 节：修改选项字节的特定规则*中对此进行了总结。不遵守这些规则通常会导致错误，如*第 7.8.7 节：选项字节更改错误 (OPTCHANGEERR)* 中所述。

嵌入式 Flash 具有两组选项字节寄存器：

- 第一组寄存器包含选项字节的当前值，其扩展名为 _CUR。所有“_CUR”寄存器均为只读寄存器。在上电复位、从系统待机模式唤醒或执行选项字节更改操作后，将自动从非易失性存储器加载寄存器的值。
- 第二组寄存器可用于修改选项字节，其扩展名为 _PRG。所有“_PRG”寄存器均可在读取/写入模式下访问。

如果 FLASH_OPTCR 寄存器中的 OPTLOCK 位置 1，则无法修改 FLASH_XXX_PRG 寄存器。

当 OPTSTRT 位置 1 时，嵌入式 Flash 将检查编程序列 (PGSERR) 和*第 7.4.8 节：修改选项字节的特定规则* (OPTCHANGEERR) 中介绍的条件。如果未检测到错误 (PGSERR/OPTCHANGEERR)，则嵌入式 Flash 将在其非易失性存储器中启动选项字节修改操作，并使用扩展名 _CUR 更新选项字节寄存器。

如果未满足*第 7.4.8 节：修改选项字节的特定规则*、*第 7.8.7 节：选项字节更改错误 (OPTCHANGEERR)* 或*第 7.8.3 节：编程顺序错误 (PGSERR)* 中所述的条件之一，则嵌入式 Flash 将中止选项字节更改操作。在这种情况下，FLASH_XXX_PRG 寄存器不会被当前的选项值覆盖。用户应用程序可以检查其配置中出现的问题。

解锁选项字节修改

复位后，OPTLOCK 位会置 1，FLASH_OPTCR 会锁定。因此，在尝试更改选项字节之前，应用软件必须先解锁选项配置寄存器。*第 7.5.5 节：Flash 配置保护*中介绍了 FLASH_OPTCR 解锁序列。

选项字节修改顺序

要修改用户选项字节，需遵循以下顺序：

1. 检查 FLASH_NSSR 寄存器中的 BSY 位，以确认当前是否未执行任何 Flash 操作；检查 FLASH_NSSR 寄存器中的 NSWBNE 位，以确认写缓冲区是否为空
2. 通过 FLASH_NSSR 寄存器确认数据缓冲区是否为空 (DBNE=0)。
3. 清零之前的操作所导致的全部错误标志
4. 解锁 FLASH_OPTCR 寄存器，如*第 7.5.5 节：Flash 配置保护*中所述（除非寄存器已经解锁）
5. 将所需的新选项字节值写入相应的选项寄存器 (FLASH_XXX_PRG)。
6. 将 FLASH_OPTCR 寄存器中的选项字节开始更改位 OPTSTRT 置 1。
7. 等待 FLASH_NSSR 寄存器中的 BSY 位清零
8. OPTSTRT 位在序列结束时（或出现错误时）自动清零
9. 复位器件。如果修改内容包含以下任一 OB，则必须执行这一步：产品状态和 NSBOOTADDR。建议对其他选项字节进行复位，但不作严格要求。

注意： *如果在进行选项字节修改时发生复位或掉电，则将保留原始选项字节值。需要新的选项字节修改序列才能设定新值*

7.4.4 选项字节概述

下表列出了通过嵌入式 Flash 寄存器管理的所有用户选项字节，以及首次更改选项字节前的初始值（用户默认配置）。

表 26. 选项字节构成

寄存器	位域															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLASH_OPTSR[31:16]	SWAP_BANK	Res	Res.								IWDG_STDBY	IWDG_STOP	Res.		IO_VDDIO2_HSLV	IO_VDD_HSLV
默认出厂值	*0*	*0*	*1*	*0*	*1*	*1*	*0*	*1*	*0*	*0*	*1*	*1*	*0*	*0*	*0*	*0*

表 26. 选项字节构成（续）

寄存器	位域															
FLASH_OPTSR[15:0]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PRODUCT_STATE								NRST_STDBY	NRST_STOP	Res.	WWDG_SW	IWDG_SW	BORH_EN	BOR_LEV	
默认出厂值	*1*	*1*	*1*	*0*	*1*	*1*	*0*	*1*	*1*	*1*	*0*	*1*	*1*	*0*	*0*	*0*
FLASH_OPTSR2[31:16]	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
默认出厂值	*1*	*1*	*0*	*0*	*0*	*0*	*1*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
FLASH_OPTSR2[15:0]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Res.	Res.	Res.	Res.	Res.	SRAM1_ECC	SRAM1_RST	Res.	Res.	SRAM2_ECC	Res.	BKPRAM_ECC	SRAM2_RST	Res.	Res.	Res.
默认出厂值	*0*	*0*	*0*	*0*	*0*	*1*	*1*	*0*	*0*	*1*	*0*	*1*	*1*	*0*	*0*	*0*
FLASH_NSBOOTR[31:16]	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NSBOOTADD[23:8]															
默认出厂值	*0x0800*															
FLASH_NSBOOTR[15:0]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NSBOOTADD[7:0]								NSBOOT_LOCK							
默认出厂值	*0x00*								*0xC3*							
FLASH_WRP1R[31:16]	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
默认出厂值	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*
FLASH_WRP1R[15:0]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP[7]	WRP[6]	WRP[5]	WRP[4]	WRP[3]	WRP[2]	WRP[1]	WRP[0]
默认出厂值	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*
FLASH_WRP2R[31:16]	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
默认出厂值	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*

表 26. 选项字节构成（续）

寄存器	位域															
FLASH_WRP2R[15:0]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP[7]	WRP[6]	WRP[5]	WRP[4]	WRP[3]	WRP[2]	WRP[1]	WRP[0]
默认出厂值	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*
FLASH_OTPBLR[31:16]	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	LOCKBL															
默认出厂值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
FLASH_OTPBLR[15:0]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LOCKBL															
默认出厂值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
FLASH_HDP1R[31:16]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						HDP1_END								HDP1_STRT		
默认出厂值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
FLASH_HDP2R[15:0]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						HDP2_END								HDP2_STRT		
默认出厂值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

7.4.5 用户和系统选项字节说明

下面列出了应用程序可以使用的通用选项字节：

- 看门狗管理
  - IWDG_STOP：该位为 1 时，在停止模式下激活独立看门狗 IWDG 计数器（该位为 0 时，停止计数或冻结）
  - IWDG_STDBY：该位为 1 时，在待机模式下激活独立看门狗 IWDG 计数器（该位为 0 时，停止计数或冻结）
  - IWDG_SW：硬件 (0) 或软件 (1) IWDG 看门狗控制选择

注意： *如果使能了硬件看门狗“控制选择”功能（设为 0），则会在上电时自动使能看门狗，从而生成复位，除非在计数结束之前写入看门狗密钥寄存器或重载递减计数器。*

根据 IWDG_STOP 和 IWDG_STDBY 选项的配置，IWDG 可分别在器件处于停止模式或待机模式时继续计数 (1) 或停止计数 (0)。当 IWDG 在停止模式或待机模式期间保持运行时，它可从这些模式中唤醒器件。

- 复位管理
  - BOR_LEV：欠压复位 (BOR) 电压，用于指示激活/释放复位的供电电压阈值。将 BORH_EN 选项位置 1 可以使能 BOR。请参见*第 9 章：电源控制 (PWR)*
  - NRST_STDBY：如果该位清零，则会在进入待机模式时生成复位
  - NRST_STOP：如果该位清零，则会在进入停止或停止 2 模式时生成复位
  - BORH_EN：使能欠压复位 (BOR)。请参见*第 9 章：电源控制 (PWR)*

注意： *如果 NRST_STDBY（或 NRST_STOP）清零，则只要成功执行待机（或停止）模式进入序列，器件就会复位，而不是进入待机（或停止）模式。*

- 器件选项
  - IO_VDDIO2_HSLV：置 1 时，可以为 VDDIO2 电源轨配置低于 2.7 V
  - IO_VDD_HSLV：置 1 时，可以为 VDD 电源轨配置低于 2.7 V

意法半导体交付器件时，通用选项字节中设定的值如下：

- 看门狗管理
  - IWDG 在待机和停止模式下处于激活状态（选项值 = 0x1）
  - IWDG 不会在上电时自动使能（选项字节值 = 0x1）
- 复位管理：
  - BOR：欠压电压选项（复位电压）等于 2.1 V（选项字节值 = 0x0）。当器件进入待机、停止或停止 2 低功耗模式时，不产生复位（选项字节值 = 0x1）
- 器件在全电压范围内工作，禁止在低压条件下进行 I/O 速度优化 (IO_VDDIO2_HSLV = IO_VDD_HSLV = 0)

有关详细信息，请参见*第 7.10 节：FLASH 寄存器*。

7.4.6 数据保护选项字节说明

下面列出了可用于增强数据保护的选项字节：

- PRODUCT_STATE[7:0]：产品生命周期状态（有关详细信息，请参见*第 7.5.8 节：产品状态转换*）。
- WRP1/2：存储区 1（或存储区 2）扇区写保护选项，低电平有效。有关详细信息，请参见*第 7.5.6 节：写保护*。
  - 位 N：扇区 N 写保护。
- HDPx：用户 Flash 中的 HDPL 专用区域控制。

出厂编程时，数据保护选项字节的值如下：

- 产品状态取决于销售类型
- 禁止写保护（所有选项字节位均置 1）

有关详细信息，请参见*第 7.10 节：FLASH 寄存器*。

7.4.7 自举地址选项字节说明

下面列出的选项字节可用于为应用程序配置相应的自举地址：

- PRODUCT_STATE

  间接影响自举序列。
- NSBOOTADD

  选择默认的自举地址。
- BOOT_LOCK

  防止对自举配置进行进一步的修改。
- SWAP_BANK：存储区交换选项，置 1 时将在自举后交换用户 Flash 存储区（请参见*第 7.3.10 节：Flash 存储区交换*）。如果 BOOT_LOCK 有效，则 SWAP_BANK 中的值为固定的只读值。

意法半导体交付器件时，PRODUCT_STATE 为开放，BOOT_LOCK 未设置 (0xC3)。地址 NSBOOTADD = 0x0800 0000。

有关详细信息，请参见*第 7.10 节：FLASH 寄存器*。

7.4.8 修改选项字节的特定规则

除 OPTLOCK 位和寄存器访问规则外，对于选定的安全敏感型选项字节位域，还有其他的保护方法。

可以同时修改多个选项字节，但如果它们出于保护目的相互依赖，则会检查两种状态。

除本章中列出的少数例外情况外，违反规则将会导致 OPTCHANGEERR 标志置 1（更多详细信息，请参见*第 7.8.7 节：选项字节更改错误 (OPTCHANGEERR)*）。

表 27. 特定 OB 修改规则概述

OB	HDPL	值	产品状态
PRODUCT_STATE	0,1(1)	一组可能的转换	一组可能的转换（*表 37*）
HDP	0,1(1)	-	-
NSBOOTADD	-	-	开放或配置 禁止 NSBOOT_LOCK
LOCKBL	-	单向切换(1)	-
SWAP_BANK	-	-	-
NSBOOT_LOCK	-	-	开放、退回或配置 切换到解锁

1. 大多数转换都可以执行，而与 HDPL 无关。只有几个需要特定的 HDPL（见下文）。

即使在关闭 PRODUCT_STATE 下，如果满足所有其他限制，也仍然可以修改一些 OB。下表提供了概览。

表 28. 关闭状态的产品中可修改的 OB

PRODUCT_STATE	可修改的 OB
关闭	SWAP_BANK、LOCKBL、PRODUCT_STATE
锁定	SWAP_BANK、LOCKBL

要更新以下 OB，必须遵守特定的规则：

- **PRODUCT_STATE**

  PRODUCT_STATE 转换遵循具有两种转换类型的状态机。可以不受任何限制地锁定产品。只能使用调试接口并通过数字签名验证来开放产品。退回到“开放 PRODUCT_STATE”会擦除受保护的内容。更多详细信息，请参见*第 7.5.8 节：产品状态转换*。

  总结：只能在 HDPL1 下执行选定的更改，可以在 HDPL0 下执行退回。

- **HDP**

  只能在 HDPL0 和 HDPL1 下进行修改。

- **NSBOOTADD**

  只能在开放和配置状态下进行更改。由 NSBOOT_LOCK 锁定。

- **LOCKBL**

  只能在一个方向上随意更改。此切换永久不可逆。

- **SWAP_BANK**

  当 NSBOOT_LOCK 有效 (0xB4) 时，此 OB 不可修改。
- **NSBOOT_LOCK**

  只能在锁定方向上随意更改。在开放、配置和退回状态下可以解锁。

注意： *对于上述所有用户选项字节：加载默认值，并在 OBL 期间发生双重 ECC 错误时发出入侵信号。*

7.5 FLASH 安全性和保护

由于敏感信息存储在 Flash 中，因此防止其遭受意外操作（例如读取机密区域、对不可变扇区的非法编程或恶意擦除 Flash）非常重要。

为此，Flash 实现了以下保护机制：

- 暂时隔离保护 (HDP)
- 配置保护
- 用户 Flash 写保护
- 器件自举状态管理
- OTP 锁定

本节详细介绍了上述所有安全机制。

支持两种访问模式：

– 非特权

– 特权

Flash 接口按以下顺序评估访问限制：

1. 写保护（用于写访问）
2. HDP
3. 特权

7.5.1 隐藏保护 (HDP)

对于每个存储区，可以按照扇区粒度定义一个非易失性安全隐藏保护 (HDP) 区域。通过提高 SBS 中的 HDPL 级别，可以拒绝对隐藏保护区域的访问。

当 HDPL = 1 时，用户 Flash 不受 HDP 保护。当 HDPL ≥ 2 时，每个存储区中的用户 OB 定义的 HDP 区域都会关闭。不允许对 HDP 区域进行读、写、取指或擦除操作。

HDP 区域可以扩展，扩展设置仅为一个寄存器值，并不存储在 OB 中。当 HDPL = 3 时，HDP 区域保持激活状态，并可以添加扩展。如果在基本用户 OB 定义的区域未激活时添加扩展，则扩展将多覆盖一个扇区，因为 HDPx_END 扇区也受到保护。

HDPL 级别只能通过系统复位来清除，无法禁用 HDP 区域。

定义受保护 HDP 区域的方法是使用起始扇区和结束扇区设置其大小。

可以使用寄存器 FLASH_HDPEXTR 值 HDPx_EXT 来扩展不可访问区域的大小。HDPx_EXT 值是添加到 HDP 区域（相对于 HDPx_END 扇区）的扇区数。寄存器中的易失性 HDPx_EXT 值可以递增，但不能递减。

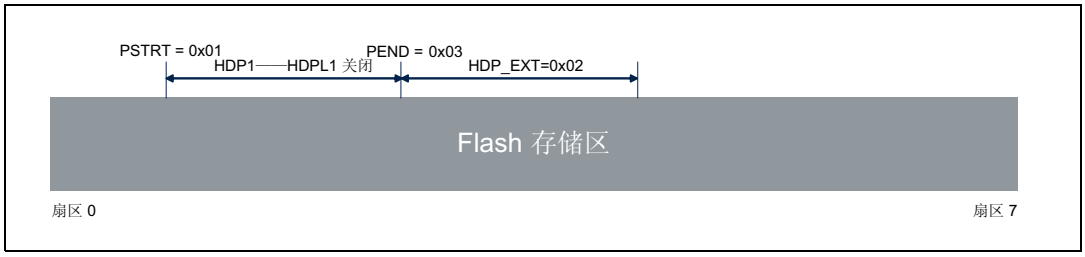
如果 HDPx_STRT > HDPx_END，扩展将覆盖由 HDPx_END 和 HDPx_END + HDPx_EXT 标记的扇区之间 (HDPx_EXT + 1) 大小的区域。

默认情况下，HDPx_END 设置为 0，HDPx_STRT 设置为 1。这意味着用户 Flash 中的 HDP 区域大小为 0，将从 Flash 起始地址（第一个扇区）开始扩展。

例如，通过从地址 0x0800 2000（包含）到地址 0x0800 5FFF（包含）的 HDP 来保护区域：

- 对于物理存储区 1，必须对选项字节寄存器进行如下编程：
  - HDP1_STRT = 0x01
  - HDP1_END = 0x03

图 16. 用户 Flash 中的 HDP



- 一旦 HDPL 递增到 2，将无法访问标记的区域。之后，HDPL = 2 代码将扩展大小设置为 2。HDPL 递增到 3 后，扩展区域也将无法访问。

  或者，如果：

  HDP1_END = 0

  HDP1_START = 1

  HDPx_EXT = 1

  扇区 0 和扇区 1 受 HDP 保护。

  此外，一个相对不常见的示例：

  HDP1_END = 2

  HDP1_START = 7

  HDPx_EXT = 3

- 扇区 2 到扇区 5 受 HDP 保护。此时，用户应该了解到如果应用程序只需要扇区 2 到扇区 4 是 HDP 区域，则应该将 HDPx_EXT 编程为 2 而不是 3。
- 如果交换两个存储区，为物理存储区 1 定义的保护会保留在物理存储区 1 上，不受交换的影响。单独的保护适用于物理存储区 2，选项字节寄存器也必须进行如下编程：
  - HDP2_STRT = 0x01
  - HDP2_END = 0x03

注意： *有关存储区交换机制的详细信息，请参见第 7.5.4 节：Flash 存储区在存储区交换情况下的属性。*

表 29. HDP 保护定义

用户 Flash	使能 OB HDP 区域	使能 HDP 扩展
HDPL = 0,1	HDP 保护 = 0	HDP 保护 = 0
HDPL = 2	HDP 保护 = 1	HDP 保护 = 0
HDPL = 3	HDP 保护 = 1	HDP 保护 = 1

表 30. 安全隐藏保护

HDPx 水印选项字节值 (x = 1,2)		隐藏保护区域
HDPL ≤ 1	-	用户 Flash 中无不可访问的 HDP 区域
HDPL = 2	HDPx_END < HDPx_STRT	用户 Flash 中无不可访问的 HDP 区域
	HDPx_END = HDPx_STRT	仅有一个扇区受 HDP 保护
	HDPx_END > HDPx_STRT	HDPx_STRT 与 HDPx_END 之间的区域受 HDP 保护
HDPL = 3	HDPx_END < HDPx_STRT 且 HDPx_EXT=0	用户 Flash 中无不可访问的 HDP 区域
	HDPx_END ≥ HDPx_STRT 或 HDPx_EXT>0	min(HDPx_STRT, HDPx_END) 与 (HDPx_END + HDPx_EXT) 之间的区域受 HDP 保护

表 31. HDP 保护汇总

HDP 保护	用户 Flash			
	访问 OB HDP 区域		访问 EXT HDP 区域	
	HDPL2 或 HDPL3	HDPL0 或 HDPL1	HDPL3	HDPL0、HDPL1 或 HDPL2
取指	总线错误	正常	总线错误	正常
读	RAZ		RAZ	
写	WI、WRPERR	如果禁止 WRP：正常 否则：WI、WRPERR	WI、WRPERR	如果禁止 WRP：正常 否则：WI、WRPERR
擦除（批量擦除）				

7.5.2 特权 Flash 区域保护

任何扇区都可以使用配置寄存器实时编程为特权或非特权状态。FLASH_PRIVBB1x（或 FLASH_PRIVBB2x）寄存器用于为存储区 1（或存储区 2）中的扇区配置特权属性。

当扇区特权属性位 PRIVBByx[i] 置 1 时，只能通过特权访问模式来访问扇区。非特权页可以通过特权或非特权访问模式进行访问。

要修改基于块的特权属性，建议执行以下操作：

- 检查相关扇区，以确认当前未执行任何 Flash 操作
- 在修改扇区安全属性 PRIVBByx[i] 之后，添加 ISB 指令

小心： 将扇区从特权属性切换到非特权属性不会擦除相关扇区的内容。

表 32. 特权保护汇总

访问		主 Flash	
		非特权扇区	特权扇区
特权	取指	正常	
	读		
	写		
	擦除		
非特权	取指	正常	RAZ
	读		
	写		WI、WRPERR
	擦除		

表 33. 特权与批量/存储区擦除

访问	主 Flash		
	非特权 FLASH	特权 FLASH	非特权 FLASH 和 特权 FLASH 组合
特权批量擦除	正常		
非特权批量擦除	正常	WI、WRPERR	

注意： *有关 PRIVBByx[i] 访问控制，请参见表 34：特权配置寄存器访问条件。*

表 34. 特权配置寄存器访问条件

访问		PRIVBBRxy 中的扇区设置访问
X	P/NP	总线错误
R	P/NP	正常
W	P	正常
W	NP	WI

7.5.3 Flash 寄存器的特权模式和非特权模式

可以通过特权访问和非特权访问模式读写 Flash 寄存器，具体取决于 *FLASH 特权配置寄存器 (FLASH_PRIVCFGR)* 中的 NSPRIV 位。

- 当 NSPRIV 位复位时，所有 Flash 寄存器都可以通过特权访问或非特权访问进行读写操作。
- 当 NSPRIV 位置 1 时，所有 Flash 寄存器都只能通过特权访问进行读写操作。对特权寄存器的非特权访问为 RAZ/WI。

7.5.4 Flash 存储区在存储区交换情况下的属性

SWAP_BANK 选项位用于修改存储器映射中每个存储区的地址。当 SWAP_BANK 复位时，Flash 存储区 1 映射在低地址范围。当 SWAP_BANK 置 1 时，Flash 存储区 1 映射在高地址范围。Flash 存储区属性与其存储区内容保持一致，因此交换存储区时无需修改其设置寄存器：

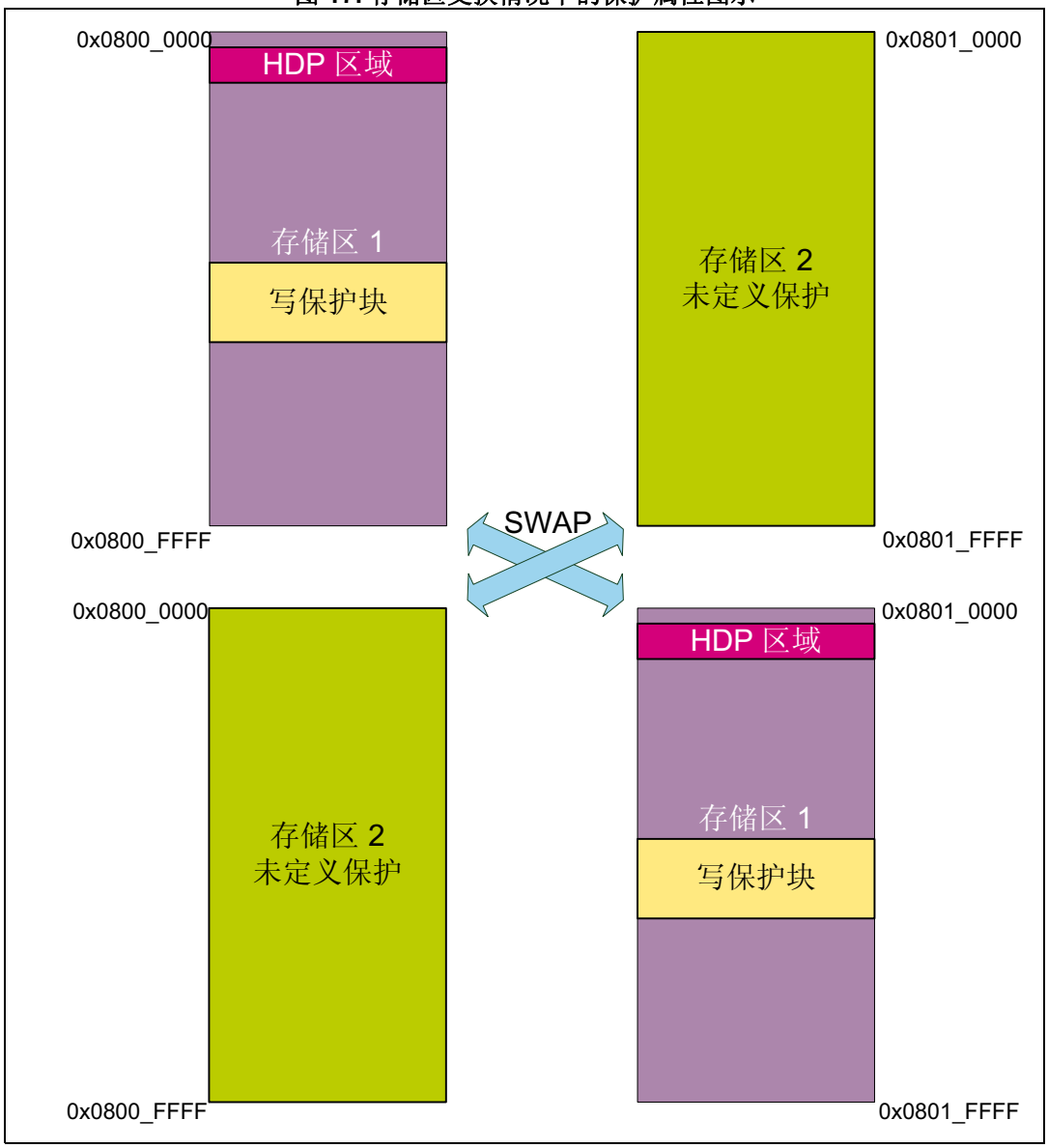
- Flash 写保护扇区 FLASH_WRPSG（请参见*第 7.5.6 节：写保护*）
- FLASH_HDPx 寄存器中的隐藏保护
- Flash 基于块的特权存储区 x 寄存器 y FLASH_PRIVBBxy

将 NSBOOT_LOCK 置 1 可以使 SWAP_BANK 不可变。

注意： *无论 SWAP_BANK 的值为何，FLASH ECC 校正寄存器 (FLASH_ECCCORR) 和 FLASH ECC 检测寄存器 (FLASH_ECCDETR) 中的 BK_ECC 位以及 FLASH 控制寄存器 (FLASH_NSCR) 中的 BKSEL 位在处于低电平（或高电平）时都始终是指存储区 1（或存储区 2）。*

下图显示了在存储区交换情况下的安全属性和保护行为。

图 17. 存储区交换情况下的保护属性图示



7.5.5 Flash 配置保护

嵌入式 Flash 使用硬件机制来保护以下资产，以防其遭受意外的修改或误修改（如软件错误）：

- 选项字节更改
- 写操作
- 擦除命令
- 中断屏蔽

下表总结了嵌入式 Flash 配置寄存器保护。此表中未显示的寄存器不受密钥保护。

表 35. Flash 接口寄存器保护汇总

寄存器名称	解锁寄存器	保护的资产
FLASH_NSCR	FLASH_NSKEYR	写/擦除控制
FLASH_OPTCR	FLASH_OPTKEYR	Flash 存储区选项字节字更改

7.5.6 写保护

嵌入式 Flash 写保护的作用是保护嵌入式非易失性代码和/或数据免遭意外修改。

将 FLASH_WRP1/2 寄存器中的相应 WRP1/2 位清零/置 1，可为任何 8 KB 扇区单独执行写保护或取消保护（请参见 *FLASH 存储区 1 扇区写保护 (FLASH_WRP1R_CUR)* 和 *FLASH 存储区 2 扇区写保护 (FLASH_WRP2R_CUR)*）。

受到写保护的扇区既不能擦除，也不能编程。因此，如果某个扇区处于写保护状态，则无法执行存储区擦除操作。

注意： *有关写保护错误的信息，请参见第 7.8 节：FLASH 错误管理。*

7.5.7 生命周期管理

非易失性状态或调试状态取决于用户选项字节所设置的产品状态。能否进行调试或更改所选安全设置与此状态相关。虽然此状态存储在 OB 中，但控制调试策略的是 SBS。

无论调试状态如何，HDP 都可以工作。即使在调试模式下，当 HDPL 递增时，HDP 区域也会隐藏。

定义的状态如下：

表 36. 产品状态、调试状态和调试策略

PRODUCT_STATE	OB 中的代码	说明
开放	0xED	用户 Flash 开放 (~RDP0)
配置	0x17	配置——正在安装 OEM 不可变可信根。
已配置	0x2E	OEM 完成——已安装 OEM 不可变可信根。(~RDP 0.5)
关闭	0x72	正在运行应用程序的状态。禁止调试，可以执行退回。
锁定	0x5C	不允许转换到其他状态、更改策略或调试。(1)
退回	0x9A	调试身份验证系统在转换为开放时触发的临时状态

1. 实际上，与关闭状态的情况相同（调试证书不正确或未定义）。

1. 无调试保护

可对 Flash 主存储器区域执行读取、编程和擦除操作。对选项字节的限制最少。

2. 全产品保护

所有调试功能均禁止：

- 用户模式：在用户模式（自举 Flash）下执行的代码可访问 Flash 主存储器和所选的选项字节，可对其进行所有操作（读取、擦除和编程）。
- 自举 RAM 模式：不可从 SRAM 自举。
- 可使用 Flash 批量擦除操作转换到开放状态。具体取决于调试解锁策略（请参见下文）。

7.5.8 产品状态转换

正常的产品生命周期最终会进入更加封闭的状态，无需采取安全措施。向开放状态的转换称为退回，由调试身份验证控制逻辑实施控制。如果调试解锁策略设置为“锁定”，则不接受退回。使调试身份验证证书失效可以强制进入锁定状态。

本章未列出的所有转换均无效。

读取当前的产品状态没有任何限制。

转换过程方向

从开放状态到任意封闭状态的转换关键在于适当的产品配置和预配置。必须按正确的顺序执行转换。

正常的转换方向是：

- 开放到配置或已配置
- 配置到已配置
- 已配置到锁定或关闭

转换由器件上运行的软件管理，也可以直接使用调试接口进行管理。软件假定转换是通过自举程序或可信根服务触发的，但通常在器件上运行的任何软件都可以执行发展转换。

转换到开放调试状态

此转换为完全退回。起始状态是除开放和锁定外的任何其他状态。借助器件上运行 (HDPL = 1) 的调试身份验证库，可以使用调试工具来验证调试退回访问权限（请参见*第 13.3.6 节：SBS 调试控制*）。验证凭证后，器件将置于中间状态，即退回状态。从此状态开始，器件在 HDPL = 0 的条件下安全地退回到开放状态。

转换会产生以下结果：

- 更新了产品状态
- 更新了用户选项（安全设置复位）
- Flash 批量擦除（甚至包括受写保护的扇区）
- 备份 RAM 擦除（即使使能了备份域写保护）

表 37. PRODUCT_STATE 转换

从	至(1)					
	开放	配置	已配置	关闭	锁定	（完全）退回
开放	-	正常	正常	X	X	X
配置	X	-	正常	正常	正常	正常 (HDPL1)

表 37. PRODUCT_STATE 转换（续）

从	至(1)					
	开放	配置	已配置	关闭	锁定	（完全）退回
已配置	X	X	-	正常	正常	正常 (HDPL1)
关闭	X	X	X	-	X	正常 (HDPL1)
锁定	X	X	X	X	-	X
（完全）退回	正常 (HDPL0)	X	X	X	X	-

1. X = 不允许转换（OPTCHANGEERR 置 1）
- = 无变化
正常 = 有效转换

在*表 37：PRODUCT_STATE 转换*中，X 表示不允许执行的转换，破折号表示无变化，正常表示有效转换。一些转换只能在括号中提供的 HDPL 下进行。尝试进行非法转换会导致 OPTCHANGEERR 置 1。

错误的 PRODUCT_STATE（OBL 时）将被解析为锁定。

7.5.9 一次性可编程和只读存储器保护

第 7.3.9 节：OTP 和 RO 存储器访问 中详细介绍了 Flash 中 OTP/RO 的各个部分。除专用写保护外，Flash 接口不提供任何保护。

一旦在制造阶段创建了 RO 区域，便无法再写入此区域。*OTP 写保护*部分介绍了一种锁定 OTP 区域的方法。

下表总结了访问条件。

表 38. OTP/RO 访问限制

-	RO/OTP 专用 8 KB 扇区			
	0x08FF_E000 - E7FF	0x08FF_E800 - EFFF 保留	0x08FF_F000 - F7FF OTP	0x08FF_F800 - FFFF RO
X	总线错误	总线错误		
R		如果大小正确，则为(1) RAZ，否则将生成总线错误	如果大小正确，则为 OK，否则将生成总线错误	
W		如果大小正确，则为 WI，否则将生成总线错误	如果大小错误，则生成总线错误。如果块受 LOCKBL 保护，则 WRPERR 置 1，否则为 OK。	如果大小正确，则为 WI，否则将生成总线错误
擦除	WI			

1. 字大小为 16 位，支持 32 位访问。尝试进行其他访问会导致总线错误。

7.6 系统存储器

7.6.1 系统存储器简介

系统存储器存储了由意法半导体在 STM32H503xx 生产期间编程的 RSS（根安全服务）固件。RSS 为用户固件提供运行时服务。本节稍后将介绍这些服务。

7.6.2 RSS 用户函数

得益于 RSS 库，RSS 可以提供运行时服务。与其他微控制器外设功能和映射一样，RSS 库函数在 STM32CubeH5 固件包提供的 CMSIS 器件头文件中面向用户公开。有关 STM32CubeH5 固件包的更多详细信息，请参见 UM2656。RSS 库通过带有前缀的 NSSLIB 函数提供运行时服务。

表 39. RSS 服务的 C 定义宏

C 定义宏	在 Flash 中的位置
NSSLIB_PFUNC	0x0BF8FE6C

NSSLIB

用户固件使用 NSSLIB_PFUNC C 定义宏调用 NSSLIB 函数，此宏指向系统存储器中的位置。

表 40. NSSlib 接口函数列表

库	函数
NSSLIB_PFUNC	JumpHDPLvl2
	JumpHDPLvl3

NSSLIB 函数将在后面的章节中进行介绍。

a) JumpHDPLvl2

原型：

```
uint32_t JumpHDPLvl2(uint32_t VectorTableAddr, uint32_t MPUIndex)
```

用户代码函数调用示例：

```
NSSLIB_PFUNC->JumpHDPLvl2((uint32_t)NextVectorTableAddr, 1U );
```

参数：

- VectorTableAddr：

输入参数，要应用的下一个向量表地址。

向量表格式为 Cortex-M33 内核使用的格式。

- MPUIndex：

输入参数，MPU 区域索引。在调用 JumpHDPLvl2 之前，调用函数应定义相应的 MPU 区域但使其保持禁止状态。此函数应在跳转到向量表的复位处理程序前使能 MPU 区域。向量表复位处理程序函数应属于 MPU 区域。

说明：

用户调用 JumpHDPLvl2 可以：

- 将 HDPL 递增到 2，从而关闭用户 Flash HDPL1 区域
- 跳转到向量表中内置的复位处理程序（地址作为输入参数传递）

关闭 HDPL1 后，JumpHDPLvl2 将使能作为输入参数提供的 MPU 区域。使能 MPU 后，此函数会将 SP 设置为传递的向量表所提供的地址，并跳转到向量表支持的复位处理程序函数。JumpHDPLvl2 不会设置新的向量表。

成功执行后，该函数既不会返回值，也不会将 LR 压入堆栈。

错误：

如果执行失败（错误的输入参数值），则 RSSLIB_Sec_JumpHDPLvl2 将返回 0xF5F5F5F5。

b) JumpHDPLvl3

原型

```
uint32_t JumpHDPLvl3(uint32_t VectorTableAddr, uint32_t MPUIndex)
```

用户代码函数调用示例：

```
NSSLIB_PFUNC->JumpHDPLvl3((uint32_t)NextVectorTableAddr, 1U );
```

参数：

- VectorTableAddr：
  输入参数，要应用的下一个向量表地址。
  向量表格式为 Cortex-M33 内核使用的格式。
- MPUIndex：
  输入参数，MPU 区域索引。在调用 JumpHDPLvl3 之前，调用函数应定义相应的 MPU 区域但使其保持禁止状态。此函数应在跳转到向量表的复位处理程序前使能 MPU 区域。向量表复位处理程序函数应属于 MPU 区域。

说明：

用户调用 JumpHDPLvl3 可以：

- 将 HDPL 递增到 3，从而关闭用户 Flash HDPL1 和 HDPL2 区域
- 然后，跳转到向量表中内置的复位处理程序（地址作为输入参数传递）

关闭 HDPL1/2 后，JumpHDPLvl3 将使能作为输入参数提供的 MPU 区域。使能 MPU 后，此函数会将 SP 设置为传递的向量表所提供的地址，并跳转到向量表支持的复位处理程序函数。JumpHDPLvl3 不会设置新的向量表。

成功执行后，该函数既不会返回值，也不会将 LR 压入堆栈。

错误：

如果执行失败（错误的输入参数值），则 JumpHDPLvl3 将返回 0xF5F5F5F5。

7.7 FLASH 低功耗模式

表 41 总结了 STM32 低功耗模式下嵌入式 Flash 的行为。嵌入式 Flash 属于内核域。

表 41. 低功耗模式对嵌入式 Flash 的影响

电源模式	内核域电压范围	FLASH 繁忙时是否允许	FLASH 功耗模式
运行/睡眠	VOS0/1/2/3	是	运行
停止（时钟停止）	SVOS3/4/5	否	时钟门控（在正常或低功耗模式下）(1)
待机	关闭	否	关闭

1. 选择 SVOS5 时，强制 Flash 进入低功耗模式。

当系统状态发生变化或处于给定系统状态时，嵌入式 Flash 可能会获得不同的供电电压范围 (VOS)，具体取决于应用。将嵌入式 Flash 切换到各种电源模式（运行、时钟门控、停止或关闭）的过程如下文所述。

注意： *有关微控制器电源状态的更多信息，请参见电源控制部分 (PWR)。*

FLASH 域到停止或待机模式的切换管理

如*表 41* 中所述，如果嵌入式 Flash 通知复位和时钟控制器 (RCC) 其正忙（BSY、DBNE 和 NS/WBNE 置 1），则微控制器无法将内核域切换到停止或待机模式。

可通过两种方法释放嵌入式 Flash：

- 通过以下任一操作复位 FLASH_NSSR 寄存器中的 NSWBNE 忙标志：
  a) 用缺失的数据填满写缓冲区。
  b) 通过激活 FLASH_NSCR 寄存器中的 NSFW 位强制执行写操作，而不填入缺失的数据。此操作会强制将所有缺失的数据设置为“高电平”。
- 轮询 FLASH_NSSR 寄存器中的 BSY 忙位，直到其清零。这种情况说明记录的所有写入、擦除和选项更改操作均已完成。

微控制器随后可以将内核域切换到停止或待机模式。

7.8 FLASH 错误管理

7.8.1 简介

在读取、编程或擦除操作期间，如果发生错误，嵌入式 Flash 会自动报告。可报告多种错误：

- *写保护错误 (WRPERR)*
- *编程顺序错误 (PGSERR)*
- *错误校正代码错误 (ECCC/ECCD)*
- *选项字节更改错误 (OPTCHANGEERR)*

应用软件可分别针对每个错误使能中断，如*第 7.9 节：FLASH 中断*中详述。

注意： *对于所有错误，应用软件都必须在尝试执行新的修改操作前清零错误标志。*

由于只有一个写缓冲区并且一次只允许执行一个操作，因此写控制位和状态位由两个存储区共用：

- 写控制位（PG、NSFW、EOPIE、WRPERRIE、PGSERRIE、STRBERRIE、INCERRIE、CLR_EOP、CLR_WRPERR、CLR_STRBERR、CLR_INCERR 和 CLR_PGSERR）同时控制存储区 1 和存储区 2。
- 写状态标志同时为存储区 1 和存储区 2 报告（NSWBNE、DBNE、EOP、WRPERR、PGSERR、STRBERR、INCERR 和 BSY）错误。

7.8.2 写保护错误 (WRPERR)

当尝试对非易失性存储器进行非法擦除/编程操作时，嵌入式 Flash 会将 FLASH_NSSR 寄存器中的写保护错误标志 WRPERR 置 1。

如果擦除操作针对以下存储区域之一，则会被拒绝并标记为非法：

- 受写保护 (WRP) 的扇区。
- 通过设置 HDPL 使其无法访问的 HDP 区域。
- 尝试在非特权模式下擦除特权扇区。

如果编程操作针对以下存储区域之一，则会被忽略并标记为非法：

- 系统 Flash
- 受写保护 (WRP) 的用户扇区
- 通过 LOCKBL 锁定的 OTP 块
- 只读部分
- 保留区域
- 非特权模式下的特权区域

当 WRPERR 标志置 1 时，操作会被拒绝，相应的存储区中也不会进行任何更改。

如果检测到此错误，则写缓冲区失效。

注意： *在执行任何擦除/编程操作之前，必须清零 WRPERR 标志。*

将 FLASH_NSCCR 寄存器中的 CLR_WRPERR 位置 1 可清零 WRPERR 标志。

如果 FLASH_NSCR 寄存器中的 WRPERRIE 位置 1，当 WRPERR 标志置 1 时，会生成中断（有关详细信息，请参见*第 7.9 节：FLASH 中断*）。

7.8.3 编程顺序错误 (PGSERR)

当编程顺序不正确时，Flash 接口会将 FLASH_NSSR 寄存器中的编程顺序错误标志 PGSERR 置 1。

更具体地说，当满足下列条件之一时，PGSERR 标志将会置 1：

- 在请求新的写/擦除操作或选项更改之前，尚未清除 INCERR、WRPERR、PGSERR、STRBERR 和 OPTCHANGEERR 等错误。
- 对于擦除操作，出现以下情况时，PGSERR 将会置 1：
  - 丢失擦除操作 (FLASH_NSCR)：STRT = 1 并且（MER = 0、SER = 0 且 BER = 0）
  - 同时请求扇区擦除和批量擦除操作（STRT 以及 MER、SER 和 BER 中的多个位置 1）
  - PG 在擦除操作期间置 1（该位可确保不会同时发出擦除请求和写请求）：(STRT=1) 并且 (PG=1)
  - 擦除操作在写缓冲区等待下一个数据时启动：(STRT = 1) 并且 WBNE = 1。
  - 擦除操作在 DBNE = 1 时启动
- 对于编程操作，出现以下情况时，PGSERR 将会置 1：
  - 丢失写标志：请求写操作，但在请求之前尚未将 FLASH_NSCR 寄存器中的编程使能位 PG 置 1。
  - 接收到 AHB 写请求，并且（SER = 1、BER = 1 或 MER = 1）
  - 在 WBNE=1 时请求 16 位数据访问
- 对于选项更改操作，出现以下情况时，PGSERR 将会置 1：
  - 选项更改操作在写缓冲区等待下一个数据时启动：OPTSTRT = 1 并且 NSWBNE = 1
  - OPTSTRT 置 1 并且 DBNE = 1

当因尝试执行写操作（Flash 编程）失败而导致 PGSERR 标志置 1 时，当前的编程操作将会中止，并且相应的存储区中不会进行任何更改。写数据缓冲区也会失效。

注意： *当 PGSERR 标志置 1 时，受上述保护机制影响，应用程序执行的最后一个写操作会有丢失的风险。因此，建议在 PGSERR 置 1 时生成中断，并在中断处理程序中通过回读 Flash 中的值来验证上一次写操作是否成功。*

PGSERR 标志也阻止了任何新的编程操作。这意味着在开始新的编程操作之前，必须将 PGSERR 清零。

将 FLASH_NSCCR 寄存器中的 CLR_PGSERR 位置 1 可清零 PGSERR 标志。

如果 FLASH_NSCR 寄存器中的 PGSERRIE 位置 1，当 PGSERR 标志置 1 时，会生成中断。有关详细信息，请参见*第 7.9 节：FLASH 中断*。

7.8.4 选通错误 (STRBERR)

当应用软件多次在写缓冲区中写入同一字节时，无论写访问的目标存储区为何，Flash 接口都会将选通错误标志 STRBERR (FLASH_NSSR) 置 1。

当 STRBERR 标志置 1 时，当前的编程操作将会中止，写缓冲区失效。

将 FLASH_NSCCR 寄存器中的 CLR_STRBERR 位置 1 可清零 STRBERR 标志。

如果 FLASH_NSCR 寄存器中的 STRBERRIE 位置 1，当 STRBERR 标志置 1 时，会生成中断。有关详细信息，请参见*第 7.9 节：FLASH 中断*。

7.8.5 不一致错误 (INCERR)

当检测到访问中的编程不一致时，Flash 接口会将 FLASH_NSSR 寄存器中的不一致错误标志 INCERR 置 1。

更具体地说，当满足下列条件之一时，INCERR 标志将会置 1：

- 在上一个写操作完成之前尝试进行写操作，例如：
  - 应用软件启动了写操作以填充 128 位写缓冲区，但在缓冲区满之前，向其他的 Flash 地址发送了新的突发写请求。
  - 一个主器件启动了写操作，但在缓冲区满之前，另一个主器件对同一地址或其他地址启动了新的写操作。

注意： *在开始新的写操作之前，必须清零 INCERR 标志，否则将生成顺序错误 (PGSERR)。*

建议按照以下顺序操作，以免在发生不一致错误时丢失数据：

1. 当 INCERR 标志置 1 时，执行处理程序。
2. 停止对嵌入式 Flash 的所有写请求。
3. 将 INCERR 位清零。
4. 重新启动被中断的写操作。

将 FLASH_NSCCR 寄存器中的 CLR_INCERR 位置 1 可清零 INCERR 标志。

如果 FLASH_NSCR 寄存器中的 INCERRIE 位置 1，当 INCERR 标志置 1 时，会生成中断（有关详细信息，请参见*第 7.9 节：FLASH 中断*）。

7.8.6 错误校正代码错误 (ECCC/ECCD)

当在读操作过程中检测到单错误校正时，Flash 接口会将 FLASH_ECCCORR 寄存器中的单错误校正标志 ECCC 置 1。

当在读操作过程中检测到双重 ECC 错误时，Flash 接口会将 FLASH_ECCDETR 寄存器中的双重错误检测标志 ECCD 置 1。

当 ECCC 标志置 1 时，会返回经过校正的读取数据。因此，应用程序可以忽略此错误并请求新的读操作。当 ECCD 标志置 1 时，会生成 NMI。如果 ECCD 标志置 1，软件就必须在 NMI 中断服务程序中使指令缓存失效 (CACHEINV = 1)。可以在 SBS 寄存器 *SBS FLITF ECC NMI 屏蔽寄存器 (SBS_ECCNMIR)* 中屏蔽此 NMI，以进行数据访问（OTP、数据区域和 RO 数据）。

当 ECCC 或 ECCD 标志置 1 时，生成错误的 Flash 字的地址会保存在 FLASH_ECCCORR (FLASH_ECCDETR) 寄存器中。如果地址对应于只读区域或 OTP 区域，则 FLASH_ECCCORRR (FLASH_ECCDETR) 寄存器中的 OTP_ECC 位也会置 1。当生成错误的相关标志复位后，该寄存器会自动清空。

BK_ECC 标志用于指示发生错误的 Flash 存储区。

SYSF_ECC 标志用于指示在系统 Flash 区域中检测到了错误。

表 42. 定位 ECC 故障

OTP_ECC	SYSF_ECC	BK_ECC	Flash 区域	ADDR_ECC 最小值	ADDR_ECC 最大值
0	0	0/1	用户 Flash	0x0000	0xFFFF
0	1	0/1	系统 Flash	0x0000	0x0FFF
1	0	0	OTP	0x0600	0x07FF

注意： *连续出现单校正错误或双重检测错误时，只有对应于第一个错误的地址会存储在 FLASH_ECCCORR (FLASH_ECCDETR) 寄存器中。*

注意： *在开始新的读操作之前，必须将 ECCC 或 ECCD 标志清零。*

注意： *由于 ECC 接口由两个存储区共用，因此，如果两个存储区同时出现同一错误，则会显示物理存储区 1 错误。如果一个存储区报告了 ECCD，而另一个存储区报告了 ECCC，则会显示两个错误。*

将 FLASH_ECCCORR (FLASH_ECCDETR) 寄存器中的 ECCC 位（或 ECCD 位）置 1 可清零 ECCC（或 ECCD）标志。

如果 FLASH_ECCCORR 寄存器中的 ECCC 位置 1，当 ECCC 标志置 1 时，会生成中断。仅对 ECCD 生成 NMI。有关详细信息，请参见*第 7.9 节：FLASH 中断*。

7.8.7 选项字节更改错误 (OPTCHANGEERR)

当 Flash 接口在选项更改操作期间发现错误时，此接口会中止操作，并将 FLASH_NSSR 寄存器中的选项字节更改错误标志 OPTCHANGEERR 置 1。

出现以下情况时，会在 OPTSTRT 位置 1 后产生错误：

- 任何禁止的 PRODUCT_STATE 转换（*表 37*）。
- SWAP_BANK 由 NSBOOT_LOCK 锁定。
- 在错误的 HDPL 下尝试修改 OB（所选 PRODUCT_STATE 转换）
- 尝试使用规范未识别的无效值修改 OB（仅限枚举的魔术数）
- 尝试在不允许修改 OB 的产品状态下修改 OB（通常需要在支持调试的开放状态下进行更改）。

注意： *OB 说明（请参见第 7.4.8 节：修改选项字节的特定规则）和表 27 中提供了例外情况和详细信息。*

将 FLASH_CCR 寄存器中的 CLR_OPTCHANGEERR 位置 1 可清零 OPTCHANGEERR 标志。

如果 FLASH_NSCR 寄存器中的 OPTCHANGEERRIE 位置 1，当 OPTCHANGEERR 标志置 1 时，会生成中断（有关详细信息，请参见*第 7.9 节：FLASH 中断*）。

在开始新的操作（选项更改、擦除或写入）之前，必须将 OPTCHANGEERR 标志清零。

7.8.8 其他 HardFault 错误

以下事件会在相应的总线接口上生成总线错误：

- 通过主 AHB 系统总线，访问目标代码和数据（使用 9 位 ECC）。
  - 访问无效地址（包括禁止代码使用的数据地址）。
  - 在 HDPL 值不正确的条件下从 HDP 区域获取代码和数据。
  - 在非特权模式下从特权区域获取代码和数据。
- 通过 AHB 配置或系统总线，访问目标 OTP/RO（所有地址都使用 6 位 ECC）：
  - FLASH_KEYR 或 FLASH_OPTKEYR 的密钥输入错误。
  - 对系统 AHB 接口进行 8 位访问。
  - 寄存器解锁序列错误。

7.9 FLASH 中断

Flash 接口可以生成可屏蔽中断，用以在给定的存储区上通知以下事件：

- 读写错误（请参见*第 7.8 节：FLASH 错误管理*）
  - 读操作期间发生单 ECC 错误校正
  - 写入不一致错误
  - 编程顺序错误
  - 写操作期间发生选通错误
  - 选项更改操作错误
- 安全错误（请参见*第 7.8 节：FLASH 错误管理*）
  - 写保护错误
- 其他事件（见下文）
  - 编程结束

在读操作期间检测到双重 ECC 错误时会生成 NMI。

用户可以通过更改 FLASH_CR、FLASH_ECCCORR 和 FLASH_ECCDETR 寄存器中的屏蔽位，单独使能或禁止 Flash 接口的各个中断源。将相应的屏蔽位置 1 可使能中断。

注意： *在执行写操作之前，必须解锁 FLASH_CR 寄存器，如第 7.5.5 节：Flash 配置保护中所述。*

表 43 对可用的 Flash 接口中断功能进行了总结。如下表所述，在触发新的操作之前需要将一些标志清零。

表 43. Flash 中断请求

中断事件 事件标志	寄存器中的 错误标志标签	使能控制位	清零标志以 恢复操作
操作结束事件	EOP	EOPIE	N/A
写保护错误	WRPERR	WRPERRIE	是
编程顺序错误	PGSERR	PGSERRIE	是
选通错误	STRBERR	STRBERRIE	是
不一致错误	INCERR	INCERRIE	是
选项字节错误	OPTCHANGEERR	-	是
ECC 单错误校正事件	ECCC	ECCCIE	否
ECC 双重错误检测事件(1)	ECCD	禁止	否

1. NMI。

表 43 中所述的各个可屏蔽中断源的状态（除选项字节错误和 ECC 外）都可以从 FLASH_NSSR 寄存器中读取。这些中断源可通过将 FLASH_NSCCR 寄存器中的相应位置 1 来清除。

注意： *无需解锁机制即可清除中断。*

操作结束事件

如果将 FLASH_NSCR 寄存器中的操作结束中断使能位 (EOPIE) 置 1，则在擦除操作、编程操作或选项字节更改结束时会生成中断。

将 FLASH_NSCCR 寄存器中的 CLR_EOP 位置 1，可将 EOP 标志清零。

7.10 FLASH 寄存器

每个寄存器都分配有一个偏移地址和一个复位值。对于表示选项字节值的寄存器，复位值由 OBL 过程决定。如果 OBL 成功，则从 OB 加载复位值。如果 OBL 失败，则会设置严格限制的默认值。

7.10.1 FLASH 访问控制寄存器 (FLASH_ACR)

FLASH access control register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

更多详细信息，请参见*第 7.3.4 节：FLASH 读操作*和*第 7.3.5 节：FLASH 编程操作*。

偏移地址：0x000

复位值：0x0000 0013

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

位 31:9 保留，必须保持复位值。

位 8 **PRFTEN**：预取使能 (Prefetch enable)。修改位值时，用户必须回读 ACR 寄存器以确保已将 PRFTEN 考虑在内。

该位用于控制预取。

0：禁止预取。

1：当至少延迟一个等待周期时使能预取。

位 7:6 保留，必须保持复位值。

位 5:4 **WRHIGHFREQ[1:0]**：Flash 信号延时 (Flash signal delay)

这些位用于控制编程操作期间各非易失性存储器信号之间的延时。应用软件必须根据嵌入式 Flash 接口频率将这些位编程为正确值。有关详细信息，请参见*表 20*。

注意：不会执行任何检查来验证配置是否正确。

对于某些频率，可以选择两个 WRHIGHFREQ 值。

位 3:0 **LATENCY[3:0]**：读延迟 (Read latency)

这些位用于控制对两个非易失性存储区执行读操作期间所使用的等待周期数。应用软件必须根据嵌入式 Flash 接口频率和电压条件将这些位编程为正确值。

0000：从非易失性存储器中读取字时需要使用 0 个等待周期

0001：从非易失性存储器中读取字时需要使用 1 个等待周期

0010：从非易失性存储器中读取字时需要使用 2 个等待周期

...

0111：从非易失性存储器中读取字时需要使用 7 个等待周期

1111：从非易失性存储器中读取字时需要使用 15 个等待周期

注意：硬件不会执行任何检查来验证配置是否正确。

7.10.2 FLASH 密钥寄存器 (FLASH_NSKEYR)

FLASH key register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

FLASH_NSKEYR 为只写寄存器。要将 FLASH_NSCR 寄存器解锁并允许对其执行编程/擦除操作，必须顺序编程以下值。如果操作顺序不正确，会锁定 FLASH_NSCR 寄存器，下次系统复位才会解锁。

第一个密钥 = 0x4567 0123

第二个密钥 = 0xCDEF 89AB

偏移地址：0x004

复位值：0x0000 0000

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

位 31:0 **NSKEY[31:0]**：非易失性存储器配置访问解锁密钥 (Non-volatile memory configuration access unlock key)

7.10.3 FLASH 选项密钥寄存器 (FLASH_OPTKEYR)

FLASH option key register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

FLASH_OPTKEYR 为只写寄存器。要将 FLASH_OPTCR 寄存器解锁，必须顺序编程以下值。如果操作顺序不正确，会锁定 FLASH_OPTCR 寄存器，下次系统复位才会解锁。

第一个密钥 = 0x0819 2A3B

第二个密钥 = 0x4C5D 6E7F

偏移地址：0x00C

复位值：0x0000 0000

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

位 31:0 **OPTKEY[31:0]**：FLASH 选项字节控制访问解锁密钥 (FLASH option bytes control access unlock key)

7.10.4 FLASH 操作状态寄存器 (FLASH_OPSR)

FLASH operation status register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x0018

复位值：0xXXXX XXXX

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

位 31:29 **CODE_OP[2:0]**：Flash 操作代码 (Flash memory operation code)

000：上一次复位期间未执行任何 Flash 操作
001：单次写操作中断
010：保留
011：扇区擦除操作中断
100：存储区擦除操作中断
101：批量擦除操作中断
110：选项更改操作中断
111：保留

位 28:25 保留，必须保持复位值。

位 24 **OTP_OP**：OTP 操作中断 (OTP operation interrupted)
指示复位中断了 OTP 区域中正在执行的操作。

位 23 **SYSF_OP**：系统 Flash 中的操作中断 (Operation in system flash memory interrupted)
指示复位中断了系统 Flash 中正在执行的操作。

位 22 **BK_OP**：已中断操作存储区 (Interrupted operation bank)
指示操作涉及的存储区。

位 21:20 保留，必须保持复位值。

位 19:0 **ADDR_OP[19:0]**：已中断操作地址 (Interrupted operation address)

7.10.5 FLASH 选项控制寄存器 (FLASH_OPTCR)

FLASH option control register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

访问：当前未执行任何 Flash 操作时无等待周期。当 BSY 位置 1 时，将不能在写模式下访问 FLASH_OPTCR 寄存器。此时尝试对该寄存器进行写操作会导致 AHB 总线阻塞，直到 BSY 位清零。

偏移地址：0x01C

复位值：0xX000 0001

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

位 31 **SWAP_BANK**：存储区交换选项配置位 (Bank swapping option configuration bit)

SWAP_BANK 用于控制是否交换存储区 1 和存储区 2。只有在复位或 POR 后，该位才会载入 FLASH_OPTSR_CUR 寄存器的 SWAP_BANK 位。

0：不交换存储区 1 和存储区 2

1：交换存储区 1 和存储区 2

位 30:2 保留，必须保持复位值。

位 1 **OPTSTRT**：选项字节开始更改选项配置位 (Option byte start change option configuration bit)

OPTSTRT 可触发选项字节更改操作。仅当 OPTLOCK 清零时，用户才能将 OPTSTRT 置 1。该位只能通过软件置 1，并在完成选项字节更改操作或发生错误（PGSERR 或 OPTCHANGEERR）时清零。该位与 BSY 位同时复位。

在选项更改操作完成之前，用户应用程序不能修改任何 FLASH_XXX_PRG Flash 寄存器。

将该位置 1 之前，用户必须将所需值写入 FLASH_XXX_PRG 寄存器。FLASH_XXX_PRG 寄存器将保持锁定，直至选项字节更改操作已在非易失性存储器中执行。

位 0 **OPTLOCK**：FLASH_OPTCR 锁定选项配置位 (FLASH_OPTCR lock option configuration bit)

OPTLOCK 位用于锁定 FLASH_OPTCR 寄存器以及所有 _PRG 寄存器。对 FLASH_OPTKEYR 寄存器执行正确的写入序列可解锁该位。如果执行顺序不正确，或者对 FLASH_OPTKEYR 执行了两次解锁序列，该位会保持锁定状态，直到下一次系统复位。

将 OPTLOCK 编程为 1 可将其置 1。OPTLOCK 置 1 后，必须执行新的解锁序列将其解锁。如果 OPTLOCK 从 0 变为 1，FLASH_OPTCR 寄存器的其他位不会发生变化。

0：FLASH_OPTCR 寄存器已解锁

1：FLASH_OPTCR 寄存器已锁定

7.10.6 FLASH 状态寄存器 (FLASH_NSSR)

FLASH status register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x020

复位值：0x0000 000X

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPTCHANGEERR	Res.	Res.	INCERR	STRBERR	PGSERR	WRPERR	EOP
								r			r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBNE	Res.	WBNE	BSY
												r		r	r

位 31:24 保留，必须保持复位值。

位 23 **OPTCHANGEERR**：选项字节更改错误标志 (Option byte change error flag)

OPTCHANGEERR 标志指示选项字节更改操作期间出错。如果 OPTCHANGEERR 置 1，则表示选项字节更改操作未成功完成。如果 FLASH_NSCR 寄存器的 OPTCHANGEERRIE 位置 1，则会在此标志置 1 时产生中断。

向寄存器 FLASH_CCR 的 CLR_OPTCHANGEERR 写入 1 会将 OPTCHANGEERR 清零。

0：未发生选项字节更改错误

1：选项字节更改操作期间发生一个或多个错误。

注意：当 OPTCHANGEERR 置 1 时，FLASH_OPTCR 中的 OPTSTRT 位不可置 1。

位 22:21 保留，必须保持复位值。

位 20 **INCERR**：不一致错误标志 (inconsistency error flag)

发生不一致错误时，INCERR 标志将会置 1。如果 INCERRIE 置 1，将产生中断。向 FLASH_NSCCR 寄存器中的 CLR_INCERR 位写入 1 可将 INCERR 清零。

0：未发生不一致错误

1：发生不一致错误

位 19 **STRBERR**：选通错误标志 (strobe error flag)

发生选通错误（主器件多次尝试将同一字节写入写缓冲区中）时，STRBERR 标志将会置 1。如果 STRBERRIE 位置 1，将产生中断。向 FLASH_NSCCR 寄存器中的 CLR_STRBERR 位写入 1 可将 STRBERR 清零。

0：未发生选通错误

1：发生选通错误

位 18 **PGSERR**：编程顺序错误标志 (programming sequence error flag)

发生顺序错误时，PGSERR 标志将会置 1。如果 PGSERRIE 位置 1，将产生中断。向 FLASH_NSCCR 寄存器中的 CLR_PGSERR 位写入 1 可将 PGSERR 清零。

0：未发生顺序错误

1：发生顺序错误

位 17 **WRPERR**：写保护错误标志 (write protection error flag)

在编程操作期间发生保护错误时，WRPERR 标志将会置 1。如果 WRPERRIE 置 1，也会产生中断。向 FLASH_NSCCR 寄存器中的 CLR_WRPERR 位写入 1 可将 WRPERR 清零。

0：未发生写保护错误

1：发生写保护错误

位 16 **EOP**：操作结束标志 (end of operation flag)

操作（编程/擦除）完成时，EOP 标志将会置 1。如果 EOPIE 置 1，将产生中断。不需要先复位 EOP 便可开始执行新操作。向 FLASH_NSCCR 寄存器中的 CLR_EOP 位写入 1 可将 EOP 位清零。

0：操作未完成

1：操作已完成

位 15:4 保留，必须保持复位值。

位 3 **DBNE**：数据缓冲区非空标志 (data buffer not empty flag)

当 Flash 接口在专用缓冲区中处理 6 位 ECC 数据时，DBNE 标志将会置 1。该位不能通过软件清零，而是在缓冲区空闲后由硬件复位。

0：未使用数据缓冲区

1：使用数据缓冲区，等待

位 2 保留，必须保持复位值。

位 1 **WBNE**：写缓冲区非空标志 (write buffer not empty flag)

嵌入式 Flash 等待新数据填满写缓冲区时，WBNE 标志将会置 1。在此状态下，写缓冲区不为空。每次填满写缓冲区或者在发生以下其中一个事件之后清空写缓冲区时，硬件都会将 WBNE 复位：

– 应用软件使用 FLASH_NSCR 中的 FW 位强制执行写操作

– 嵌入式 Flash 检测到涉及数据丢失的错误

通过软件直接写入 0 无法复位该位。要复位该位，应执行上述任一操作来清空写缓冲区，或发送丢失的数据。

0：写缓冲区为空或已满

1：写缓冲区正在等待数据填满

位 0 **BSY**：忙标志 (busy flag)

BSY 标志指示 Flash 正忙于操作（写入、擦除或选项字节更改）。该位在 Flash 操作开始时置 1，在操作结束或出现错误时清零。

0：当前未执行任何编程、擦除或选项字节更改操作

1：正在执行编程、擦除或选项字节更改操作

7.10.7 FLASH 控制寄存器 (FLASH_NSCR)

FLASH control register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

访问：当前未执行任何 Flash 操作时无等待周期。当 BSY 位置 1 时，将不能在写模式下访问 FLASH_NSCR 寄存器。此时尝试对该寄存器进行写操作会导致 AHB 总线阻塞，直到 BSY 位清零。

偏移地址：0x028

复位值：0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BKSEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPTCHANGEERRIE	Res.	Res.	INCERRIE	STRBERRIE	PGSERRIE	WRPERRIE	EOPIE
rw								rw			rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MER	Res.	Res.	Res.	Res.	Res.	Res.	SNB[2:0]			STRT	FW	BER	SER	PG	LOCK
rw							rw	rw	rw	rs	rw	rw	rw	rw	rs

位 31 **BKSEL**：存储区选择器位 (Bank selector bit)

只有在 LOCK 清零时，才能编程 BKSEL。该位用于选择物理存储区，忽略 SWAP_BANK 设置。

0：选择存储区 1 进行存储区擦除/扇区擦除/中断使能

1：选择存储区 2 进行 BER/SER

位 30:24 保留，必须保持复位值。

位 23 **OPTCHANGEERRIE**：选项字节更改错误中断使能位 (Option byte change error interrupt enable bit)

OPTCHANGEERRIE 位用于控制选项字节更改期间出错时是否产生中断。只有在 LOCK 位清零时，才能编程该位。

0：选项字节更改期间出错时不产生中断

1：选项字节更改期间出错时产生中断。

位 22:21 保留，必须保持复位值。

位 20 **INCERRIE**：不一致错误中断使能位 (inconsistency error interrupt enable bit)

如果 INCERRIE 位置 1，在写操作期间发生不一致错误时，会产生中断。只有在 LOCK 清零时，才能编程 INCERRIE。

0：发生不一致错误时不产生中断

1：发生不一致错误时产生中断

位 19 **STRBERRIE**：选通错误中断使能位 (strobe error interrupt enable bit)

如果 STRBERRIE 位置 1，在写操作期间发生选通错误时（主器件多次向写缓冲区中编程同一字节），会产生中断。只有在 LOCK 清零时，才能编程 STRBERRIE。

0：发生选通错误时不产生中断

1：发生选通错误时产生中断

位 18 **PGSERRIE**：编程顺序错误中断使能位 (programming sequence error interrupt enable bit)

如果 PGSERRIE 位置 1，在编程操作期间发生顺序错误时，会产生中断。只有在 LOCK 清零时，才能编程 PGSERRIE。

0：发生顺序错误时不产生中断

1：发生顺序错误时产生中断

位 17 **WRPERRIE**：写保护错误中断使能位 (write protection error interrupt enable bit)

如果 WRPERRIE 位置 1，在编程操作期间发生保护错误时，会产生中断。只有在 LOCK 清零时，才能编程 WRPERRIE。

0：发生保护错误时不产生中断

1：发生保护错误时产生中断

位 16 **EOPIE**：操作结束中断控制位 (end of operation interrupt control bit)

将 EOPIE 位置 1 可在编程或擦除操作结束时产生中断。只有在 LOCK 清零时，才能编程 EOPIE。

0：操作结束时不产生中断

1：操作结束时使能中断

位 15 **MER**：批量擦除请求 (Mass erase request)

将 MER 位置 1 会请求批量擦除操作（仅限用户 Flash）。只有在 LOCK 清零时，才能编程 MER。

如果 BER 和 SER 都置 1，PGSERR 会置 1。

0：未请求批量擦除操作

1：请求批量擦除操作

当需要执行批量擦除操作而某些扇区受保护时，会触发错误。

位 14:9 保留，必须保持复位值。

位 8:6 **SNB[2:0]**：扇区擦除选择编号 (sector erase selection number)

这些位用于选择擦除操作的目标扇区（否则不使用）。只有在 LOCK 清零时，才能编程 SNB。

0x00：选择扇区 0

0x01：选择扇区 1

...

0x07：选择扇区 7

位 5 **STRT**：擦除启动控制位 (erase start control bit)

STRT 位用于启动扇区擦除或存储区擦除操作。只有在 LOCK 清零时，才能编程 STRT。

STRT 在操作结束或出现错误时复位。无法通过软件复位。

位 4 **FW**：写入强制控制位 (write forcing control bit)

即使写缓冲区未满，FW 也可以强制执行写操作。在这种情况下，所有未写入的位都将由硬件置 1。只有在 LOCK 清零时，才能编程 FW。

相应操作得到确认后，嵌入式 Flash 会复位 FW。

注意：使用强制写操作可以防止应用程序稍后使用 1 以外的其他值更新丢失的位，否则可能会导致永久 ECC 错误。

仅当写缓冲区非空时，写入强制才有效（特别要说明的是，当连续执行强制写操作时，FW 不会启动多次写操作）。

由于只有一个写缓冲区，因此 FW 可以对存储区 1 或存储区 2 强制执行写操作。

位 3 **BER**：擦除请求 (erase request)

将 BER 位置 1 会请求存储区擦除操作（仅限用户 Flash）。只有在 LOCK 清零时，才能编程 BER。

如果 MER 和 SER 也置 1，PGSERR 会置 1。

0：未请求存储区擦除操作

1：请求存储区擦除操作

注意：当需要执行存储区擦除操作而某些扇区受保护时，会触发写保护错误。

位 2 **SER**：扇区擦除请求 (sector erase request)

将 SER 位置 1 会请求扇区擦除操作。只有在 LOCK 清零时，才能编程 SER。

如果 MER 和 SER 也置 1，PGSERR 会置 1。

0：未请求扇区擦除操作

1：请求扇区擦除操作

位 1 **PG**：编程控制位 (programming control bit)

只有在 LOCK 清零时，才能编程 PG。

PG 用于允许在存储区 1 和存储区 2 中进行编程。

0：禁止编程

1：使能编程

位 0 **LOCK**：配置锁定位 (configuration lock bit)

该位用于锁定 FLASH_NSCR 寄存器。对 FLASH_NSKEYR 寄存器执行正确的写入序列可解锁该位。如果执行顺序不正确，或者对 FLASH_NSKEYR 执行了两次解锁序列，该位会保持锁定状态，直到下一次系统复位。

将 LOCK 编程为 1 可将其置 1。OPTLOCK 置 1 后，必须执行新的解锁序列将其解锁。当 LOCK 从 0 变为 1，FLASH_NSCR 寄存器的其他位不会发生变化。

0：解锁 FLASH_NSCR 寄存器

1：锁定 FLASH_NSCR 寄存器

7.10.8 FLASH 清零控制寄存器 (FLASH_NSCCR)

FLASH clear control register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x030

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLR_OPTCHANGEERR	Res.	Res.	CLR_INCERR	CLR_STRBERR	CLR_PGSERR	CLR_WRPERR	CLR_EOP
								w			w	w	w	w	w

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

位 31:24 保留，必须保持复位值。

位 23 **CLR_OPTCHANGEERR**：写入该位可将 FLASH_NSSR 中的相应标志清零。

位 22:21 保留，必须保持复位值。

位 20 **CLR_INCERR**：INCERR 标志清零位 (INCERR flag clear bit)
将该位置 1 可将 FLASH_NSSR 寄存器中的 INCERR 标志复位为 0。

位 19 **CLR_STRBERR**：STRBERR 标志清零位 (STRBERR flag clear bit)
将该位置 1 可将 FLASH_NSSR 寄存器中的 STRBERR 标志复位为 0。

位 18 **CLR_PGSERR**：PGSERR 标志清零位 (PGSERR flag clear bit)
将该位置 1 可将 FLASH_NSSR 寄存器中的 PGSERR 标志复位为 0。

位 17 **CLR_WRPERR**：WRPERR 标志清零位 (WRPERR flag clear bit)
将该位置 1 可将 FLASH_NSSR 寄存器中的 WRPERR 标志复位为 0。

位 16 **CLR_EOP**：EOP 标志清零位 (EOP flag clear bit)
将该位置 1 可将 FLASH_NSSR 寄存器中的 EOP 标志复位为 0。

位 15:0 保留，必须保持复位值。

7.10.9 FLASH 特权配置寄存器 (FLASH_PRIVCFGR)

FLASH privilege configuration register

偏移地址：0x03C

复位值：0x0000 0000

该寄存器可通过特权访问和非特权访问进行读取，但只能通过特权模式写入。

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

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

位 31:2 保留，必须保持复位值。

位 1 **NSPRIV**：特权属性 (privilege attribute)
0：始终允许访问寄存器
1：拒绝对寄存器进行非特权访问

位 0 保留，必须保持复位值。

7.10.10 FLASH HDP 扩展寄存器 (FLASH_HDPEXTR)

FLASH HDP extension register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

只能在 HDPL <= 2 时修改此寄存器。

该寄存器中的值不能递减。尝试写入小于当前值的值将被忽略。

偏移地址：0x048

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP2_EXT[2:0]		
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_EXT[2:0]		
													rw	rw	rw

位 31:19 保留，必须保持复位值。

位 18:16 **HDP2_EXT[2:0]**：在存储区 2 的 8 KB 扇区中扩展 HDP 区域。在 HDP2_END 扇区（包含）后添加扩展。

位 15:3 保留，必须保持复位值。

位 2:0 **HDP1_EXT[2:0]**：在存储区 1 的 8 KB 扇区中扩展 HDP 区域。在 HDP1_END 扇区（包含）后添加扩展。

7.10.11 FLASH 选项状态寄存器 (FLASH_OPTSR_CUR)

FLASH option status register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

默认值：0x0030 5CD8（在 OBL 期间发生双重 ECC 错误时的值）。

该只读寄存器用于反映相应选项位的当前值。

偏移地址：0x050

复位值：0xXXXX XXXX

（OBL 时，将 Flash 中的值加载到寄存器位 0 到 31）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SWAP_BANK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IWDG_STDBY	IWDG_STOP	Res.	Res.	IO_VDDIO2_HSLV	IO_VDD_HSLV
r										r	r			r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRODUCT_STATE[7:0]								NRST_STDBY	NRST_STOP	Res.	WWDG_SW	IWDG_SW	BORH_EN	BOR_LEV[1:0]	
r	r	r	r	r	r	r	r	r	r		r	r	r	r	r

位 31 **SWAP_BANK**：存储区交换选项状态位 (Bank swapping option status bit)
SWAP_BANK 用于反映是否交换存储区 1 和存储区 2。
复位后，SWAP_BANK 会加载到 FLASH_OPTCR 的 SWAP_BANK 中。
0：不交换存储区 1 和存储区 2
1：交换存储区 1 和存储区 2

位 30:22 保留，必须保持复位值。

位 21 **IWDG_STDBY**：IWDG 待机模式冻结选项状态位 (IWDG Standby mode freeze option status bit)
置 0 后，会在系统待机模式下冻结独立看门狗 IWDG。
0：独立看门狗在待机模式下冻结
1：独立看门狗在待机模式下保持运行

位 20 **IWDG_STOP**：IWDG 停止模式冻结选项状态位 (IWDG Stop mode freeze option status bit)
置 0 后，会在系统停止模式下冻结独立看门狗 IWDG。
0：独立看门狗在系统停止模式下冻结
1：独立看门狗在系统停止模式下保持运行

位 19:18 保留，必须保持复位值。

位 17 **IO_VDDIO2_HSLV**：低 VDDIO2 电压高速 IO 状态位 (High-speed IO at low VDDIO2 voltage status bit)。该位只能在 VDDIO2 低于 2.7 V 时置 1。
0：禁止低 VDDIO2 电压高速 IO 功能（VDDIO2 可超过 2.7 V）
1：使能低 VDDIO2 电压高速 IO 功能（VDDIO2 保持低于 2.7 V）

位 16 **IO_VDD_HSLV**：低 VDD 电压高速 IO 状态位 (High-speed IO at low VDD voltage status bit)。该位只能在 VDD 低于 2.7 V 时置 1。
0：禁止低 VDD 电压高速 IO 功能（VDD 可超过 2.7 V）
1：使能低 VDD 电压高速 IO 功能（VDD 保持低于 2.7 V）

位 15:8 **PRODUCT_STATE[7:0]**：生命周期状态代码 (Life state code)（基于 Hamming 8 和 4）。更多信息，请参见*第 7.5.8 节：产品状态转换*。

位 7 **NRST_STDBY**：内核域进入待机模式的复位选项状态位 (Core domain Standby entry reset option status bit)
0：内核域进入待机模式时产生复位
1：内核域进入待机模式时不产生复位

位 6 **NRST_STOP**：内核域进入停止模式的复位选项状态位 (Core domain Stop entry reset option status bit)
0：内核域进入停止模式时产生复位
1：内核域进入停止模式时不产生复位

位 5 保留，必须保持复位值。

位 4 **WWDG_SW**：WWDG 控制模式选项状态位 (WWDG control mode option status bit)
0：WWDG 看门狗由硬件控制
1：WWDG 看门狗由软件控制

位 3 **IWDG_SW**：IWDG 控制模式选项状态位 (IWDG control mode option status bit)
0：IWDG 看门狗由硬件控制
1：IWDG 看门狗由软件控制

位 2 **BORH_EN**：欠压高电平使能状态位 (Brownout high enable status bit)
0：禁止欠压复位 (BOR)
1：使能欠压复位 (BOR)

位 1:0 **BOR_LEV[1:0]**：欠压电压选项状态位 (Brownout level option status bit)
这些位用于反映生成系统复位的电源电压。
00 或 11：BOR 级别 1，阈值电压低（VBOR1 约为 2.1 V）
01：BOR 级别 2，阈值电压中等（VBOR2 约为 2.4 V）
10：BOR 级别 3，阈值电压高（VBOR3 约为 2.7 V）

7.10.12 FLASH 选项状态寄存器 (FLASH_OPTSR_PRG)

FLASH option status register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器用于对相应选项位中的值进行编程。复位后的值会反映相应选项位的当前值。

偏移地址：0x054

复位值：0xXXXX XXXX

（OBL 时，将 Flash 中的值加载到寄存器位 0 到 31）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SWAP_BANK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IWDG_STDBY	IWDG_STOP	Res.	Res.	IO_VDDIO2_HSLV	IO_VDD_HSLV
rw										rw	rw			rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRODUCT_STATE[7:0]								NRST_STDBY	NRST_STOP	Res.	WWDG_SW	IWDG_SW	BORH_EN	BOR_LEV[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw

位 31 **SWAP_BANK**：存储区交换选项配置位 (Bank swapping option configuration bit)
SWAP_BANK 选项位用于配置是否交换存储区 1 和存储区 2。该位会在复位后载入 FLASH_OPTSR_CUR 寄存器的 SWAP_BANK 位。
0：不交换存储区 1 和存储区 2
1：交换存储区 1 和存储区 2

位 30:22 保留，必须保持复位值。

位 21 **IWDG_STDBY**：IWDG 待机模式冻结选项配置位 (IWDG Standby mode freeze option configuration bit)
置 0 后，会在系统待机模式下冻结独立看门狗 IWDG。
0：在待机模式下冻结独立看门狗
1：独立看门狗在待机模式下保持运行

位 20 **IWDG_STOP**：IWDG 停止模式冻结选项配置位 (IWDG Stop mode freeze option configuration bit)
置 0 后，会在系统停止模式下冻结独立看门狗 IWDG。
0：在系统停止模式下冻结独立看门狗
1：独立看门狗在系统停止模式下保持运行

位 19:18 保留，必须保持复位值。

位 17 **IO_VDDIO2_HSLV**：低 VDDIO2 电压高速 IO 配置位 (High-speed IO at low VDDIO2 voltage configuration bit)。该位只能在 VDDIO2 低于 2.7 V 时置 1。
0：禁止低 VDDIO2 电压高速 IO 功能（VDDIO2 可超过 2.7 V）
1：使能低 VDDIO2 电压高速 IO 功能（VDDIO2 保持低于 2.7 V）

位 16 **IO_VDD_HSLV**：低 VDD 电压高速 IO 配置位 (High-speed IO at low VDD voltage configuration bit)。该位只能在 VDD 低于 2.7 V 时置 1。

0：禁止低 VDD 电压高速 IO 功能（VDD 可超过 2.7 V）

1：使能低 VDD 电压高速 IO 功能（VDD 保持低于 2.7 V）

位 15:8 **PRODUCT_STATE[7:0]**：生命周期状态代码 (Life state code)（基于 Hamming 8 和 4）。更多信息，请参见*第 7.5.8 节：产品状态转换*。

位 7 **NRST_STDBY**：内核域进入待机模式的复位选项配置位 (Core domain Standby entry reset option configuration bit)

0：内核域进入待机模式时产生复位

1：内核域进入待机模式时不产生复位

位 6 **NRST_STOP**：内核域进入停止模式的复位选项配置位 (Core domain Stop entry reset option configuration bit)

0：内核域进入停止模式时产生复位

1：内核域进入停止模式时不产生复位

位 5 保留，必须保持复位值。

位 4 **WWDG_SW**：WWDG 控制模式选项配置位 (WWDG control mode option configuration bit)

0：WWDG 看门狗由硬件控制

1：WWDG 看门狗由软件控制

位 3 **IWDG_SW**：IWDG 控制模式选项配置位 (IWDG control mode option configuration bit)

0：IWDG 看门狗由硬件控制

1：IWDG 看门狗由软件控制

位 2 **BORH_EN**：欠压高电平使能配置位 (Brownout high enable configuration bit)

0：禁止欠压复位 (BOR)

1：使能欠压复位 (BOR)

位 1:0 **BOR_LEV[1:0]**：欠压电压选项配置位 (Brownout level option configuration bit)

这些位用于反映生成系统复位的电源电压。

00 或 11：BOR 级别 1，阈值电压低（VBOR1 约为 2.1 V）

01：BOR 级别 2，阈值电压中等（VBOR2 约为 2.4 V）

10：BOR 级别 3，阈值电压高（VBOR3 约为 2.7 V）

7.10.13 FLASH 选项状态寄存器 2 (FLASH_OPTSR2_CUR)

FLASH option status register 2

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

默认值：0xC300 0658

该只读寄存器用于反映相应选项位的当前值。

偏移地址：0x070

复位值：0xXXXX XXXX

（OBL 时，将 Flash 中的值加载到寄存器位 0 到 31）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	SRAM1_ECC	SRAM1_RST	Res.	Res.	SRAM2_ECC	Res.	BKPRAM_ECC	SRAM2_RST	Res.	Res.	Res.
					r	r			r		r	r			

位 31:11 保留，必须保持复位值。

位 10 **SRAM1_ECC**：SRAM1 ECC 检测和校正禁止 (SRAM1 ECC detection and correction disable)
- 0：使能 SRAM1 ECC 检查
- 1：禁止 SRAM1 ECC 检查

位 9 **SRAM1_RST**：系统复位时擦除 SRAM1 (SRAM1 erase upon system reset)
- 0：发生系统复位时擦除 SRAM1
- 1：发生系统复位时不擦除 SRAM1

位 8:7 保留，必须保持复位值。

位 6 **SRAM2_ECC**：SRAM2 ECC 检测和校正禁止 (SRAM2 ECC detection and correction disable)
- 0：使能 SRAM2 ECC 检查
- 1：禁止 SRAM2 ECC 检查

位 5 保留，必须保持复位值。

位 4 **BKPRAM_ECC**：备份 RAM ECC 检测和校正禁止 (Backup RAM ECC detection and correction disable)
- 0：使能 BKPRAM ECC 检查
- 1：禁止 BKPRAM ECC 检查

位 3 **SRAM2_RST**：系统复位时擦除 SRAM2 (SRAM2 erase when system reset)
- 0：发生系统复位时擦除 SRAM2
- 1：发生系统复位时不擦除 SRAM2

位 2:0 保留，必须保持复位值。

7.10.14 FLASH 选项状态寄存器 2 (FLASH_OPTSR2_PRG)

FLASH option status register 2

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器用于对相应选项位中的值进行编程。复位后的值会反映相应选项位的当前值。

偏移地址：0x074

复位值：0xXXXX XXXX

（OBL 时，将 Flash 中的值加载到寄存器位 0 到 31）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	SRAM1_ECC	SRAM1_RST	Res.	Res.	SRAM2_ECC	Res.	BKPRAM_ECC	SRAM2_RST	Res.	Res.	Res.
					rw	rw			rw		rw	rw			

位 31:11 保留，必须保持复位值。

位 10 **SRAM1_ECC**：SRAM1 ECC 检测和校正禁止 (SRAM1 ECC detection and correction disable)
- 0：使能 SRAM1 ECC 检查
- 1：禁止 SRAM1 ECC 检查

位 9 **SRAM1_RST**：系统复位时擦除 SRAM1 (SRAM1 erase upon system reset)
- 0：发生系统复位时擦除 SRAM1
- 1：发生系统复位时不擦除 SRAM1

位 8:7 保留，必须保持复位值。

位 6 **SRAM2_ECC**：SRAM2 ECC 检测和校正禁止 (SRAM2 ECC detection and correction disable)
- 0：使能 SRAM2 ECC 检查
- 1：禁止 SRAM2 ECC 检查

位 5 保留，必须保持复位值。

位 4 **BKPRAM_ECC**：备份 RAM ECC 检测和校正禁止 (Backup RAM ECC detection and correction disable)
- 0：使能 BKPRAM ECC 检查
- 1：禁止 BKPRAM ECC 检查

位 3 **SRAM2_RST**：系统复位时擦除 SRAM2 (SRAM2 erase when system reset)
- 0：发生系统复位时擦除 SRAM2
- 1：发生系统复位时不擦除 SRAM2

位 2:0 保留，必须保持复位值。

7.10.15 FLASH 唯一自举入口寄存器 (FLASH_NSBOOTR_CUR)

FLASH unique boot entry register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器用于反映相应选项位的当前值。

偏移地址：0x080

复位值：0xXXXX XXXX

（OBL 时，将 Flash 中的值加载到寄存器位 0 到 31）

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

位 31:8 **NSBOOTADD[23:0]**：唯一自举入口地址 (unique boot entry address)
这些位用于反映 UBE 地址

位 7:0 **NSBOOT_LOCK[7:0]**：该位域用于锁定 SWAP_BANK 的值和 NSBOOTADD 设置。
0xC3：SWAP_BANK 和 NSBOOTADD 仍然可以遵循各自不同的规则进行修改。
0xB4：冻结 NSBOOTADD 和 SWAP_BANK。

7.10.16 FLASH 唯一自举入口地址 (FLASH_NSBOOTR_PRG)

FLASH unique boot entry address

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器用于对相应选项位中的值进行编程。

偏移地址：0x084

复位值：0xXXXX XXXX

（OBL 时，将 Flash 中的值加载到寄存器位 0 到 31）

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

位 31:8 **NSBOOTADD[23:0]**：唯一自举入口地址 (Unique boot entry address)
这些位可用于配置自举地址

位 7:0 **NSBOOT_LOCK[7:0]**：该位域用于锁定 SWAP_BANK 的值和 NSBOOTADD 设置。
0xC3：SWAP_BANK 和 NSBOOTADD 仍然可以遵循各自不同的规则进行修改。
0xB4：冻结 NSBOOTADD 和 SWAP_BANK。

7.10.17 FLASH OTP 块锁定 (FLASH_OTPBLR_CUR)

FLASH OTP block lock

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器用于反映相应选项位的当前值。

偏移地址：0x090

复位值：0xXXXX XXXX

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

位 31:0 **LOCKBL[31:0]**：OTP 块锁定 (OTP block lock)

块 *n* 对应于 OTP 16 位字 32 x *n* 到 32 x *n* + 31。

LOCKBL[n] = 1 表示 OTP 块 *n* 中的所有 OTP 16 位字都已锁定，尝试对其进行编程会导致 WRPERR。

LOCKBL[n] = 0 表示 OTP 块 *n* 中的所有 OTP 16 位字均未锁定。

当一个块处于锁定状态时，无法移除写保护。

如果未锁定，则无法擦除 OTP 字。

7.10.18 FLASH OTP 块锁定 (FLASH_OTPBLR_PRG)

FLASH OTP block lock

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器用于对相应选项位中的值进行编程。

偏移地址：0x094

复位值：0xXXXX XXXX

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

位 31:0 **LOCKBL[31:0]**：OTP 块锁定 (OTP block lock)

块 *n* 对应于 OTP 16 位字 32 x *n* 到 32 x *n* + 31。

LOCKBL[n] = 1 表示 OTP 块 *n* 中的所有 OTP 16 位字都已锁定，尝试对其进行编程会导致 WRPERR。

LOCKBL[n] = 0 表示 OTP 块 *n* 中的所有 OTP 16 位字均未锁定。

当一个块处于锁定状态时，无法移除写保护。

将 FLASH_OTPBLR_CUR 中的相应位清零，可将 LOCKBL 位置 1。

7.10.19 FLASH 存储区 1 特权寄存器 (FLASH_PRIVBB1R1)

FLASH privilege register for bank 1

该寄存器为特权寄存器，只能通过特权访问进行读写操作。

偏移地址：0x0C0

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIVBB1[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:0 **PRIVBB1[7:0]**：特权/非特权 8 KB Flash 存储区 1 扇区属性 (Privileged / unprivileged 8 Kbytes Flash Bank1 sector attribute)（y = 0 到 7）

0：存储区 1 中的扇区 y 为非特权扇区

1：存储区 1 中的扇区 y 为特权扇区

7.10.20 FLASH 存储区 1 扇区写保护 (FLASH_WRP1R_CUR)

FLASH write sector protection for Bank1

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该只读寄存器用于反映相应选项位的当前值。

偏移地址：0x0E8

复位值：0xXXXX XXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRPSG1[7:0]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **WRPSG1[7:0]**：存储区 1 扇区保护选项状态字节 (Bank1 sector protection option status byte)
将 WRPSG1 位置 0 可对存储区 1 的相应扇区进行写保护（0：写保护；1：无写保护）

7.10.21 FLASH 存储区 1 扇区写保护 (FLASH_WRP1R_PRG)

FLASH write sector protection for Bank1

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器用于对相应选项位中的值进行编程。

偏移地址：0x0EC

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRPSG1[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:0 **WRPSG1[7:0]**：存储区 1 扇区保护选项状态字节 (Bank1 sector protection option status byte)
将 WRPSG1 位置 0 可对存储区 1 的相应扇区进行写保护（0：写保护；1：无写保护）

7.10.22 FLASH HDP 存储区 1 寄存器 (FLASH_HDP1R_CUR)

FLASH HDP Bank1 register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器最初会加载相应选项位的值。

偏移地址：0x0F8

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_END[2:0]		
													w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_STRT[2:0]		
													w	w	w

位 31:19 保留，必须保持复位值。

位 18:16 **HDP1_END[2:0]**：以 8 KB 扇区的编号设置的 HDPL 屏障结束位置

位 15:3 保留，必须保持复位值。

位 2:0 **HDP1_STRT[2:0]**：以 8 KB 扇区的编号设置的 HDPL 屏障起始位置

7.10.23 FLASH HDP 存储区 1 寄存器 (FLASH_HDP1R_PRG)

FLASH HDP Bank1 register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器只能在 HDPL0 和 HDPL1 下进行访问。当 HDPL = 2 或 3 时，则为 WI 或 RAZ。

该寄存器用于对相应选项位中的值进行编程。

偏移地址：0x0FC

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_END[2:0]		
													r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_STRT[2:0]		
													r	r	r

位 31:19 保留，必须保持复位值。

位 18:16 **HDP1_END[2:0]**：以 8 KB 扇区的编号设置的存储区 1 HDPL 屏障结束位置

位 15:3 保留，必须保持复位值。

位 2:0 **HDP1_STRT[2:0]**：以 8 KB 扇区的编号设置的存储区 1 HDPL 屏障起始位置

7.10.24 FLASH ECC 校正寄存器 (FLASH_ECCCORR)

FLASH ECC correction register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x100

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	ECCC	Res.	Res.	Res.	Res.	ECCCIE	OTP_ECC	SYSF_ECC	BK_ECC	Res.	Res.	Res.	Res.	Res.	Res.
	rw					rw	r	r	r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_ECC[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31 保留，必须保持复位值。

位 30 **ECCC**：当检测到单 ECC 错误并对其进行校正时，ECC 校正位由硬件置 1。
写入 1 即可将该位清零。

位 29:26 保留，必须保持复位值。

位 25 **ECCCIE**：ECC 单校正错误中断使能位 (ECC single correction error interrupt enable bit)。如果 ECCCIE 位置 1，当在读操作期间发生 ECC 单校正错误时，会产生中断。
0：发生 ECC 单校正错误时不产生中断
1：发生 ECC 单校正错误时产生中断

位 24 **OTP_ECC**：OTP ECC 错误位 (OTP ECC error bit)

当最后一次从只读/OTP 区域成功执行读操作期间发生单 ECC 校正错误时，该位置 1。ECC 错误的地址会出现在 ADDR_ECC 位域中。

位 23 **SYSF_ECC**：针对系统 Flash 中已校正 ECC 错误的 ECC 标志 (ECC flag for corrected ECC error in system FLASH)

用于指示 ECC 错误是否涉及系统 Flash。

位 22 **BK_ECC**：针对已校正 ECC 错误的 ECC 存储区标志 (ECC bank flag for corrected ECC error)

用于指示 ECC 错误涉及的存储区

位 21:16 保留，必须保持复位值。

位 15:0 **ADDR_ECC[15:0]**：ECC 错误地址 (ECC error address)

如果在读操作期间发生 ECC 错误（单校正错误），ADDR_ECC 会包含生成此错误的地址。错误标志复位时，ADDR_ECC 也会复位。

仅当没有 ECC 错误标志置 1 时，嵌入式 Flash 才会在该寄存器中编程地址。这意味着仅会保存第一个生成 ECC 错误的地址。

ADDR_ECC 中的地址与发生错误的 Flash 区域（用户 Flash、系统 Flash、数据区域、只读/OTP 区域）相对应。

7.10.25 FLASH ECC 检测寄存器 (FLASH_ECCDETR)

FLASH ECC detection register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x104

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCD	Res.	Res.	Res.	Res.	Res.	Res.	OTP_ECC	SYSF_ECC	BK_ECC	Res.	Res.	Res.	Res.	Res.	Res.
rw							r	r	r						

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

位 31 **ECCD**：当检测到双重 ECC 错误时，ECC 检测位由硬件置 1。

该位置 1 时，会生成 NMI。

写入 1 即可将该位清零。需要清零才能检测后续的双重 ECC 错误。

位 30:25 保留，必须保持复位值。

位 24 **OTP_ECC**：OTP ECC 错误位 (OTP ECC error bit)

当最后一次从只读/OTP 区域执行读操作期间发生双重 ECC 检测错误时，该位置 1。ECC 错误的地址会出现在 ADDR_ECC 位域中。

位 23 **SYSF_ECC**：针对系统 Flash 中双重 ECC 错误的 ECC 失效 (ECC fail for double ECC error in system flash memory)

用于指示 ECC 错误是否涉及系统 Flash。

位 22 **BK_ECC**：针对双重 ECC 错误的 ECC 失效存储区 (ECC fail bank for double ECC Error)
用于指示 ECC 错误涉及的存储区

位 21:16 保留，必须保持复位值。

位 15:0 **ADDR_ECC[15:0]**：ECC 错误地址 (ECC error address)
如果在读操作期间发生 ECC 错误（双重检测错误），ADDR_ECC 会包含生成此错误的地址。错误标志复位时，ADDR_ECC 也会复位。
仅当没有 ECC 错误标志置 1 时，嵌入式 Flash 才会在该寄存器中编程地址。这意味着仅会保存第一个生成双重 ECC 错误的地址。
ADDR_ECC 中的地址与发生错误的 Flash 区域（用户 Flash、系统 Flash、数据区域、只读/OTP 区域）相对应。

7.10.26 FLASH ECC 数据 (FLASH_ECCDR)

FLASH ECC data

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x108

复位值：0x0000 0000

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

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

位 31:16 保留，必须保持复位值。

位 15:0 **DATA_ECC[15:0]**：ECC 错误数据 (ECC error data)
当在使用 6 位 ECC 的特殊 16 位数据区域（数据区域、只读/OTP 区域）中发生双重检测 ECC 错误时，失效数据会读取到该寄存器中。
检查该位可以确定是否真的是数据失效，还是由于访问未初始化的存储器所致。

7.10.27 FLASH 存储区 2 特权寄存器 (FLASH_PRIVBB2R1)

FLASH privilege register for Bank2

该寄存器为特权寄存器，只能通过特权访问进行读写操作。

偏移地址：0x1C0

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIVBB2[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:0 **PRIVBB2[7:0]**：特权/非特权 8 KB Flash 存储区 2 扇区属性 (Privileged / unprivileged 8 Kbytes Flash Bank2 sector attribute)（y = 0 到 7）
0：存储区 2 中的扇区 y 为非特权扇区
1：存储区 2 中的扇区 y 为特权扇区

7.10.28 FLASH 存储区 2 扇区写保护 (FLASH_WRP2R_CUR)

FLASH write sector protection for Bank2

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该只读寄存器用于反映相应选项位的当前值。

偏移地址：0x1E8

复位值：0xXXXX XXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRPSG2[7:0]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **WRPSG2[7:0]**：存储区 2 扇区保护选项状态字节 (Bank2 sector protection option status byte)
将 WRPSG2 位置 0 可对存储区 2 的相应扇区进行写保护（0：写保护；1：无写保护）

7.10.29 FLASH 存储区 2 扇区写保护 (FLASH_WRP2R_PRG)

FLASH write sector protection for Bank2

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器用于对相应选项位中的值进行编程。

偏移地址：0x1EC

复位值：0xXXXX XXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRPSG2[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:0 **WRPSG2[7:0]**：存储区 2 扇区保护选项状态字节 (Bank2 sector protection option status byte)
将 WRPSG2 位置 0 可对存储区 2 的相应扇区进行写保护（0：写保护；1：无写保护）

7.10.30 FLASH HDP 存储区 2 寄存器 (FLASH_HDP2R_CUR)

FLASH HDP Bank2 register

该寄存器最初会加载相应选项位的值。

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x1F8

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP2_END[2:0]		
													r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP2_STRT[2:0]		
													r	r	r

位 31:19 保留，必须保持复位值。

位 18:16 **HDP2_END[2:0]**：以 8 KB 扇区的编号设置的存储区 2 HDPL 屏障结束位置

位 15:3 保留，必须保持复位值。

位 2:0 **HDP2_STRT[2:0]**：以 8 KB 扇区的编号设置的存储区 2 HDPL 屏障起始位置

7.10.31 FLASH HDP 存储区 2 寄存器 (FLASH_HDP2R_PRG)

FLASH HDP Bank2 register

当 FLASH_PRIVCFGR 寄存器中的 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

该寄存器只能在 HDPL0 和 HDPL1 下进行访问。当 HDPL = 2 或 3 时，则为 WI 或 RAZ。

该寄存器用于对相应选项位中的值进行编程。

偏移地址：0x1FC

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP2_END[2:0]		
													rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP2_STRT[2:0]		
													rw	rw	rw

位 31:19 保留，必须保持复位值。

位 18:16 **HDP2_END[2:0]**：以 8 KB 扇区的编号设置的存储区 2 HDPL 屏障结束位置

位 15:3 保留，必须保持复位值。

位 2:0 **HDP2_STRT[2:0]**：以 8 KB 扇区的编号设置的存储区 2 HDPL 屏障起始位置

7.11 FLASH 寄存器映射

表 44. 寄存器映射与复位值表

偏移	寄存器名称和复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	FLASH_ACR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRFTEN	Res.	Res.	WRHIGHFREQ[1:0]		LATENCY[3:0]			
	0x0000 0013																								0			0	1	0	0	1	1
0x004	FLASH_NSKEYR	NSKEY[31:0]																															
	0x0000 0000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00C	FLASH_OPTKEYR	OPTKEY																															
	0x0000 0000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x018	FLASH_OPSR		CODE_OP[2:0]		Res.	Res.	Res.	Res.	OTP_OP	SYSF_OP	BK_OP	Res.	Res.	ADDR_OP[19:0]																			
	0xXXXX XXXX	X	X	X					X	X	X			X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x01C	FLASH_OPTCR	SWAP_BANK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPTSTRT	OPTLOCK
	0xX000 0001	0																														0	1
0x020	FLASH_NSSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPTCHANGERR	Res.	Res.	INCERR	STRBERR	PGSERR	WRPERR	EOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBNE	Res.	WBNE	BSY
	0x0000 000X									0			0	0	0	0	0													0		0	0
0x028	FLASH_NSCR	BKSEL.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPTCHANGEERRIE	Res.	Res.	INCERRIE.	STRBERRIE	PGSERRIE	WRPERRIE	EOPIE	MER	Res.	Res.	Res.	Res.	Res.	Res.	SNB[2:0]			STRT	FW	BER	SER	PG	LOCK
	0x0000 0001	0											0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	1
0x030	FLASH_NSCCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLR_OPTCHANGEERR.	Res.	Res.	CLR_INCERR	CLR_STRBERR	CLR_PGSERR	CLR_WRPERR	CLR_EOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	0x0000 0000									0			0	0	0	0	0																

表 44. 寄存器映射与复位值表（续）

偏移	寄存器名称和复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x03C	FLASH_PRIVCFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NSPRIV	Res.
	0x0000 0000																															0	
0x048	FLASH_HDPEXTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP2_EXT[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_EXTT[2:0]		
	0x0000 0000														0	0	0														0	0	0
0x050	FLASH_OPTSR_CUR	SWAP_BANK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IWDG_STDBY	IWDG_STOP	Res.	Res.	IO_VDDIO2_HSLV	IO_VDD_HSLV	PRODUCT_STATE[7:0]								NRST_STDBY	NRST_STOP	Res.	WWDG_SW	IWDG_SW	BORH_EN	BOR_LEV[1:0]	
	0xXXXX XXXX	X										X	X			X	X	X	X	X	X	X	X	X	X	X	X		X	X	X	X	X
0x054	FLASH_OPTSR_PRG	SWAP_BANK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IWDG_STDBY	IWDG_STOP	Res.	Res.	IO_VDDIO2_HSLV	IO_VDD_HSLV	PRODUCT_STATE[7:0]								NRST_STDBY	NRST_STOP	Res.	WWDG_SW	IWDG_SW	BORH_EN	BOR_LEV[1:0]	
	0xXXXX XXXX	0										1	1					1	1	1	0	1	1	1	1	1	1		1	1	1	1	0
0x070	FLASH_OPTSR2_CUR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRAM1_ECC	SRAM1_RST	Res.	Res.	SRAM2_ECC	Res.	BKPRAM_ECC	SRAM2_RST	Res.	Res.
	0xXXXX XXXX																							X	X			X		X	X		
0x074	FLASH_OPTSR2_PRG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRAM1_ECC	SRAM1_RST	Res.	Res.	SRAM2_ECC	Res.	BKPRAM_ECC	SRAM2_RST	Res.	Res.
	0xXXXX XXXX																							X	X			X		X	X		
0x080	FLASH_NSBOOTR_CUR	NSBOOTADD[23:0]																								NSBOOT_LOCK[7:0]							
	0xXXXX XXXX	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x084	FLASH_NSBOOTR_PRG	NSBOOTADD[23:0]																								NSBOOT_LOCK[7:0]							
	0xXXXX XXXX	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x090	FLASH_OTPBLR_CUR	LOCKBL[31:0]																															
	0xXXXX XXXX	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x094	FLASH_OTPBLR_PRG	LOCKBL[31:0]																															
	0xXXXX XXXX	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

表 44. 寄存器映射与复位值表（续）

偏移	寄存器名称和复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0C0	FLASH_PRIVBB1R1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIVBB1[7:0]							
	0x0000 0000																									0	0	0	0	0	0	0	0
0x0E8	FLASH_WRP1RCUR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP1[7:0]							
	0xXXXX XXXX																									X	X	X	X	X	X	X	X
0x0EC	FLASH_WRP1R_PRG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP1[7:0]							
	0xXXXX XXXX																									X	X	X	X	X	X	X	X
0x0F8	FLASH_HDP1R_CUR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_END[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_STRT[2:0]		
	0xXXXX XXXX														X	X	X														X	X	X
0x0FC	FLASH_HDP1R_PRG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_END[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDP1_STRT[2:0]		
	0xXXXX XXXX														X	X	X														X	X	X
0x100	FLASH_ECCCORR	Res.	ECCR	Res.	Res.	Res.	Res.	ECCCIE	OTP_ECC	SYSF_ECC	BK_ECC	Res.	Res.	Res.	Res.	Res.	Res.	ADDR_ECC[15:0]															
	0x0000 0000		0					0	0	0	0							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x104	FLASH_ECCDETR	ECCR	Res.	Res.	Res.	Res.	Res.	Res.	OTP_ECC	SYSF_ECC	BK_ECC	Res.	Res.	Res.	Res.	Res.	Res.	ADDR_ECC[15:0]															
	0x0000 0000	0							0	0	0																						
0x108	FLASH_ECCDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATA_ECC[15:0]															
	0x0000 0000																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C0	FLASH_PRIVBB2R1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIVBB2[7:0]							
	0x0000 0000																									0	0	0	0	0	0	0	0
0x1E8	FLASH_WRP2R_CUR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRPSG2[7:0]							
	0xXXXX XXXX																									X	X	X	X	X	X	X	X
0x1EC	FLASH_WRP2R_PRG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRPSG2[7:0]							
	0xXXXX XXXX																									X	X	X	X	X	X	X	X

表 44. 寄存器映射与复位值表（续）

偏移	寄存器名称和复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1F8	FLASH_HDP2R_CUR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	HDP2_END[2:0]			Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	HDP2_STRT[2:0]		
	0xXXXX XXXX														X	X	X														X	X	X
0x1FC	FLASH_HDP2R_PRG	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	HDP2_END[2:0]			Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	HDP2_STRT[2:0]		
	0xXXXX XXXX														X	X	X														X	X	X

有关寄存器边界地址的信息，请参见*第 2.2 节*。

8 指令缓存 (ICACHE)

8.1 ICACHE 简介

Cortex-M33 处理器的 C-AHB 代码总线上引入了指令缓存 (ICACHE)，以此提高从内部存储器中获取指令和数据时的性能。

一些特定的功能（例如 hit-under-miss 和“关键字优先”重新填充策略）使得在大多数用例中几乎可以实现“零等待周期”性能。

8.2 ICACHE 主要特性

ICACHE 的主要特性如下：

- 总线接口
  - 一个 32 位 AHB 从端口：执行端口（从 Cortex-M33 C-AHB 代码接口输入）
  - 一个 128 位 AHB 主端口：master1 端口（输出到主 AHB 总线矩阵的快速总线）
  - 一个用于控制的 32 位 AHB 从端口（从 AHB 外设互连输入，用于 ICACHE 寄存器访问）
- 缓存访问
  - 命中时零等待周期
  - Hit-under-miss 功能：因上一缓存未命中而对行重新填充期间，能够对处理器请求（访问已缓存数据）进行处理
  - 得益于缓存行大小的 WRAPw 突发（32 位字大小 w 与缓存行大小一致），优化了缓存行重新填充
  - N 路组相联默认配置可以配置为单通路，表示直接映射缓存，适用于需要极低功耗的应用
- 替换和重新填充
  - pLRU-t 替换策略（伪-最近最少使用，基于二进制树），复杂性与性能达到最佳平衡点的算法
  - 关键字优先重新填充策略，最大程度减少处理器停顿
- 性能计数器

  ICACHE 实现两种性能计数器：
  - 命中监视器计数器（32 位）
  - 未命中监视器计数器（16 位）
- 错误管理
  - 可以检测意外可缓存写访问，以标记错误，并产生中断（可选）
- 维护操作
  - 缓存失效：缓存完全失效，快速命令，不可中断

8.3 ICACHE 实现

表 45. ICACHE 特性

特性	ICACHE
通路数	2
缓存大小	8 KB
缓存行宽度	16 字节
要重映射的区域数	0
AHB 从接口的数据大小	32 位
AHB 快速 master1 接口的数据大小	128 位
AHB 慢速 master2 接口的数据大小	0

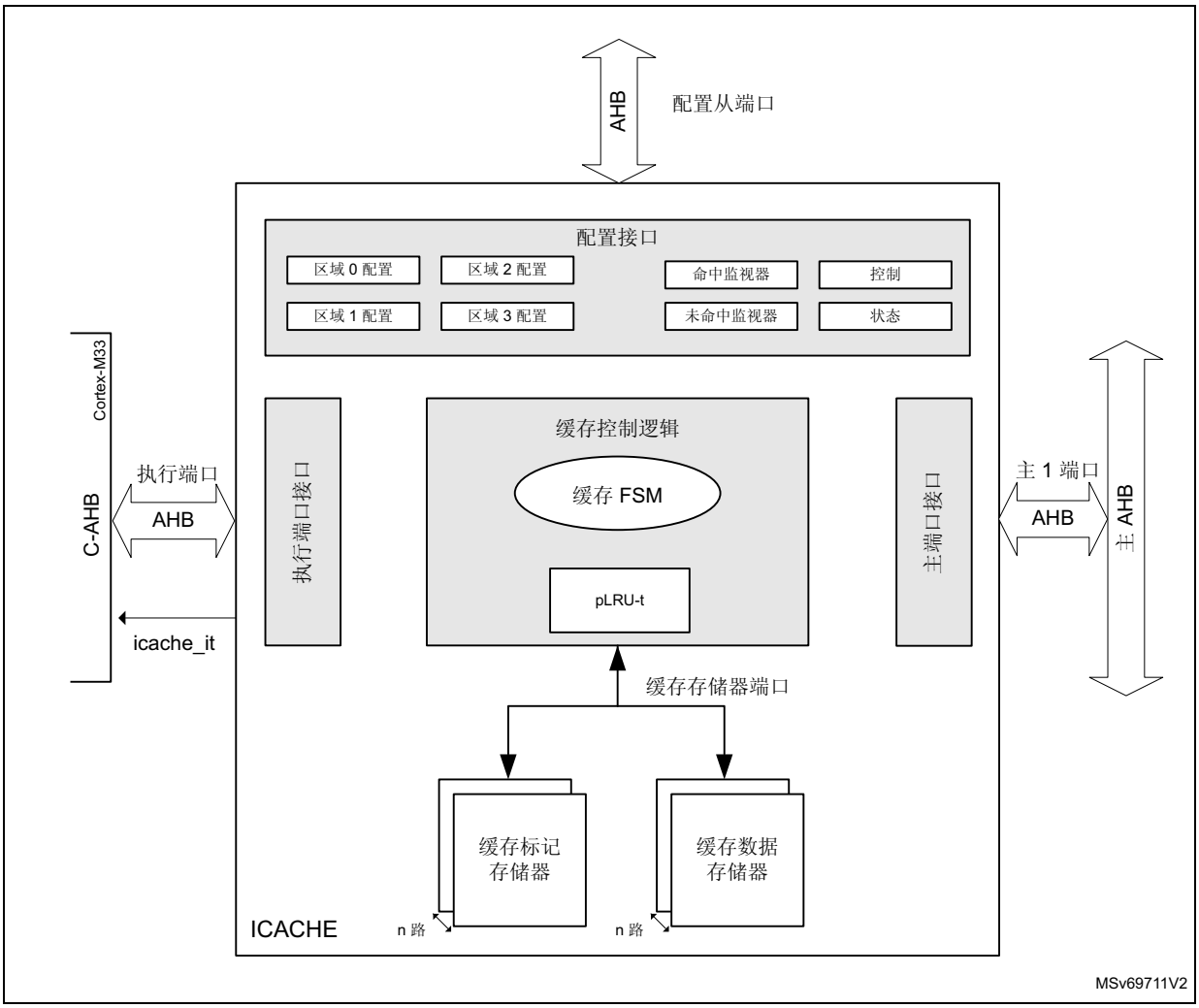
8.4 ICACHE 功能说明

指令缓存用于缓存获取的指令或存储器加载的指令（来自处理器）。因此，ICACHE 只管理读事务，不管理写事务。

为进行错误管理，发生写可缓存事务（仅在软件编程错误时发生）时，ICACHE 会将相应的错误标记置 1，如果允许，还会向处理器产生一个中断。

8.4.1 ICACHE 框图

图 18. ICACHE 框图



8.4.2 ICACHE 复位和时钟

ICACHE 的时钟由 Cortex-M33 C-AHB 总线时钟提供。

释放 ICACHE 复位信号时，缓存失效程序会自动启动，将 ICACHE 状态设置为繁忙状态 (ICACHE_SR = 0x0000 0001)。

该程序结束后：

- ICACHE 失效：“冷缓存”，所有缓存行有效位均为 0（全部 Cache 需要填充）
- ICACHE_SR = 0x0000 0002（表示缓存不再是繁忙状态）
- ICACHE 被禁止：ICACHE_CR 中的 EN 位保持其复位状态 (=0)。

注意： *禁止后，将绕过 ICACHE：从输入请求仅转发到主端口。*

8.4.3 ICACHE 标记存储器

ICACHE 标记存储器包含以下内容：

- 地址标记，用于指示缓存数据存储器中包含哪些数据
- 有效性位

每个缓存行（每个通路）有一个有效位。

有效位在缓存行重新填充（一次未命中之后）时置 1。

有效位将在以下任一情况下复位：

- 释放 ICACHE 复位信号后
- 缓存被禁止时（由软件在 ICACHE_CR 设置 EN = 0 来实现）
- 执行 ICACHE 失效命令时（由软件在 ICACHE_CR 中设置 CACHEINV = 1 来实现）

在执行输入端口接收到可缓存事务时，其 AHB 地址 (HADDR_in) 拆分成以下位域（有关 B 和 W 定义，请参见*表 46*）：

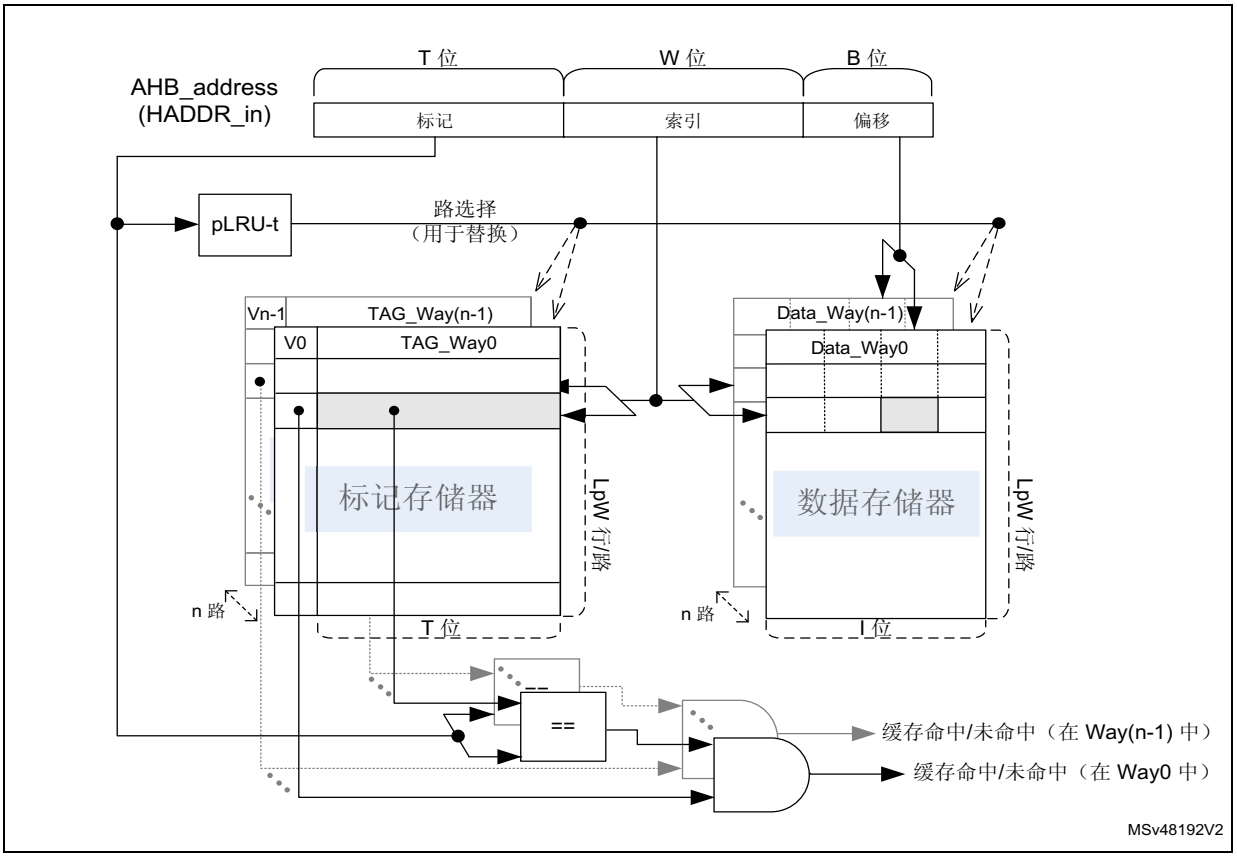
- HADDR_in[B-1:0]：地址字节偏移，指示在缓存行中选择的字节。
- HADDR_in[B+W-1:B]：地址通路索引，指示在每个通路内选择的缓存行。
- HADDR_in[31:B+W]：标记地址，将与标记存储器地址进行比较，以确认所请求的数据在 ICACHE 内是否已经可用（即有效）。

下表概括了用于标记存储器尺寸的 ICACHE 主要参数。*图 19* 显示了标记存储器和数据存储器的功能视图（针对 n 路组相联 ICACHE）。

表 46. n 路组相联工作模式（默认）的标记存储器尺寸参数值

参数	值	示例
缓存大小	S KB = s 字节 (s = 1024 x S)	8 KB = 8192 字节
缓存通路数	n	2
缓存行大小	L 字节 = l 位 (l = 8 x L)	16 字节 = 128 位
缓存行数（每通路）	LpW = s / (n x L) 行/通路	256 行/通路
地址字节偏移大小	B = log2(L) 位	4 位
地址通路索引大小	W = log2(LpW) 位	8 位
标记地址大小	T = (32 - W - B) 位	20 位

图 19. ICACHE 标记和数据存储器功能视图



8.4.4 直接映射 ICACHE（单通路缓存）

默认配置（复位时）为 n 路组相联缓存（ICACHE_CR 中的 WAYSEL = 1），但用户可以通过写入 WAYSEL = 0 将 ICACHE 配置为直接映射（仅适用于缓存已禁止，即 ICACHE_CR 中的 EN = 0 的情况）。

下表概括了选择直接映射缓存工作模式时，标记存储器的 ICACHE 主要参数。

表 47. 直接映射缓存模式下的标记存储器尺寸参数

参数	值	示例
缓存大小	S KB = s 字节 (s = 1024 x S)	8 KB = 8192 字节
缓存通路数	1	1
缓存行大小	L 字节 = I 位 (I = 8 x L)	16 字节 = 128 位
缓存行的数量	LpW = s/L 行	512 行
地址字节偏移大小	B = log2(L) 位	4 位
地址通路索引大小	W = log2(LpW) 位	9 位
标记地址大小	T = (32 - W - B) 位	19 位

在采用直接映射配置时，所有缓存操作（例如读、重新填充、失效）均保持不变。唯一的区别是在缓存行回收时没有替换算法（如*第 8.4.7 节*所述）：只有一个通路（唯一的一个）可用于任何数据重新填充。

8.4.5 ICACHE 使能

要激活 ICACHE，必须将 ICACHE_CR 中的 EN 位置 1。

ICACHE 在禁止时会被绕过，所有事务均在同一个时钟周期内自从端口复制到主端口。

建议在禁止 ICACHE 的情况下初始化或修改主存储器内容（稍后要缓存的区域），并且仅在该区域保持不变的情况下使能 ICACHE（使能的 ICACHE 会将可缓存的写事务检测为错误）。

为确保性能稳定，建议等到可能的缓存失效程序结束后再使能 ICACHE。当硬件复位信号释放、CACHEINV 置 1 或 ICACHE_CR 中的 EN 位清零时，会执行此失效程序。在该程序执行期间，ICACHE_SR 中的 BUSYF 位置 1，程序结束后，BUSYF 清零，同一个寄存器中的 BSYENDF 置 1（如果使能了 ICACHE 中断，则在这种繁忙状态结束时会产生该中断）。

软件在使能 ICACHE 之前必须先测试 BUSYF 和/或 BSYENDF 的值。否则，如果在失效程序结束前就使能 ICACHE，则任何缓存访问（此时 BUSYF = 1）都将被视为不可缓存，其性能取决于主存储器的访问时间。

默认情况下，ICACHE 在自举时处于禁止状态。

8.4.6 可缓存和不可缓存的通信

ICACHE 针对 Cortex-M33 内核而开发。其位于 C-AHB 总线上，因此用于缓存代码存储器区域，存储器映射的地址范围为 0x0000 0000 到 0x1FFF FFFF。

传入到 ICACHE 的存储器请求是否可缓存将基于其 AHB 事务存储器查找属性来定义，如*表 48* 所示。此 AHB 属性取决于被寻址区域的 MPU（存储器保护单元）编程设置。

表 48. AHB 事务的 ICACHE 可缓存性

AHB 查找属性	可缓存性
1	可缓存
0	不可缓存

如果是不可缓存访问，将绕过 ICACHE，表示 AHB 事务原样传播到主输出端口。

绕过 ICACHE 并不会增加目标存储器的访问延迟。

如果是可缓存访问，则 ICACHE 行为将如*第 8.4.7 节*所述。

可缓存的存储器区域由用户在 MPU 中定义和编程，负责为寻址指定区域的任何事务生成 AHB 属性信号。

8.4.7 可缓存访问

当 ICACHE 从 Cortex-M33 接收到可缓存事务时，ICACHE 会检查其标记存储器中是否存在请求的地址，以及对应的缓存行是否有效。

有三种情况：

- 标记存储器中存在请求的地址，缓存行有效：**缓存命中**，数据从缓存中读取，在同一个周期内提供给处理器。
- 标记存储器中不存在请求的地址：**缓存未命中**，数据从主存储器读取并提供给处理器，然后执行缓存行重新填充。

  关键字优先策略可确保处理器等待周期数最少，这是因为在缓存仍在执行缓存行重新填充时，即可提供读取的数据（相关延迟是从主存储器获取一个字时的延迟）。

  ICACHE 主端口总线上产生的突变是 WRAPw（w 为缓存行宽度，以字为单位）。

  AHB 事务属性也传播到主端口上的主 AHB 总线矩阵。
- 地址不在标记存储器中，但属于当前正在进行的主存储器重新填充突发：**缓存命中**（hit-under-miss 特性）。

  在缓存行重新填充期间发生。只要请求的数据在其主接口可用，ICACHE 就可以提供该数据，从而避免未命中（从主存储器获取数据）的情况。

发生缓存行重新填充时（由于缓存未命中），ICACHE 选择在哪个缓存行中写入重新填充的数据。

- 在直接映射（单通路）模式下，只有一个行可用来存储重新填充数据：由输入地址索引所指向的行。
- 在 n 路组相联模式下，n 个通路中各有一个行（由 n 个通路中每个通路的地址索引所指定的行）可供使用。通路选择应基于 pLRU-t 替换算法，对于每一个索引，算法会指定下一次重新填充时的候选通路。

如果必须写入重新填充数据的缓存行已经有效，则必须先使目标缓存行失效。无论是直接映射模式还是 n 路组相联模式，均是如此。

8.4.8 ICACHE 维护

可以通过软件设定 ICACHE_CR 寄存器中的 CACHEINV，使 ICACHE 的全部内容失效。

当 CACHEINV = 1 时，ICACHE 控制逻辑将 ICACHE_SR 中的 BUSYF 标志置 1，并启动缓存失效操作，将每个标记有效位复位为 0（每个缓存行对应一个有效位）。CACHEINV 自动清零。

失效操作完成后，ICACHE 自动将 BUSYF 清零，并将 ICACHE_SR 寄存器中的 BSYENDF 置 1。

如果在此标记条件下（ICACHE_IER 中的 BSYENDIE = 1）使能了 ICACHE 中断，则会产生该中断。之后，（空）缓存会再次可用。

8.4.9 ICACHE 性能监视

ICACHE 提供以下监视器以进行性能分析：

- 32 位命中监视器，计算在缓存从端口上命中 ICACHE 内容的可缓存 AHB 事务的数量。
  该监视器还会将地址存在于标记存储器或重新填充缓冲区（由于上一次未命中，其数据正在或即将从缓存主端口传入）中的所有访问考虑在内（参见*第 8.4.7 节*）。
- 16 位未命中监视器，计算在缓存从端口上未命中 ICACHE 内容的可缓存 AHB 事务的数量。
  该监视器还会将地址既不在标记存储器中也不在重新填充缓冲区内的所有访问考虑在内。

达到最大计数值时，这些监视器并不会回卷。

命中监视器和未命中监视器均可以由软件使能和复位，从而可以对特定代码段进行分析。

软件可以执行以下任务：

- 通过 ICACHE_CR 中的 HITMEN 使能/停止命中监视器。
- 通过将 ICACHE_CR 中的 HITMRST 置 1 复位命中监视器。
- 通过 ICACHE_CR 中的 MISSMEN 使能/停止未命中监视器。
- 通过将 ICACHE_CR 中的 MISSMRST 置 1 复位未命中监视器。

为降低功耗，默认情况下禁止（停止）这些监视器。

8.4.10 ICACHE 自举

自举时，ICACHE 处于禁止状态（ICACHE_CR 中的 EN = 0）。

自举过程结束后，可以使能 ICACHE（通过软件将 CACHE_CR 中的 EN 置 1）。

8.5 ICACHE 低功耗模式

如果在器件级使用 ICACHE，则大多数时候都可以从内部 ICACHE 获取指令，而不是从更大、更耗能的主存储器获取，从而降低功耗。

对于采用低性能配置（针对命中率而言）以及具有严格的低功耗限制的应用，会受益于因 ICACHE 配置为直接映射模式而具有的低功耗特性。这种单通路缓存配置可通过将 ICACHE_CR 中的 WAYSEL 设定为 0 来获得（参见*图 19*）。对于每个请求，仅访问标记和数据存储器中的必要片段，从而降低功耗。同时，缓存的效果依然提高了获取性能。即使对于大多数的代码执行，它的效率也只比 n 路组相联模式略低一点。

8.6 ICACHE 错误管理和中断

如果检测到不受支持的可缓存写请求（功能错误），ICACHE 会将 ICACHE_SR 中的 ERRF 标记置 1 以生成错误。如果对应的中断使能位置 1（ICACHE_IER 中 ERRIE = 1），则会产生中断。

在缓存失效操作结束时也会产生中断。缓存繁忙状态结束后，ICACHE 会将 ICACHE_SR 中的 BSYENDF 标志置 1。如果对应的中断使能位置 1（ICACHE_IER 中 BSYENDIE = 1），则会产生中断。

所有的 ICACHE 中断源均发出相同的唯一中断信号 icache_it，因而使用相同的中断向量。

表 49. ICACHE 中断

中断向量	中断事件	事件标志	使能控制位	中断清除方法
ICACHE	功能错误	ICACHE_SR 中的 ERRF	ICACHE_IER 中的 ERRIE	将 ICACHE_FCR 中的 CERRF 置 1
	繁忙状态结束（失效完成）	ICACHE_SR 中的 BSYENDF	ICACHE_IER 中的 BSYENDIE	将 ICACHE_FCR 中的 CBSYENDF 置 1

ICACHE 还会将所有 AHB 总线错误（例如地址解码问题）从 master1 端口传播回执行端口。

8.7 ICACHE 寄存器

8.7.1 ICACHE 控制寄存器 (ICACHE_CR)

ICACHE control register

偏移地址：0x000

复位值：0x0000 0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MISSMRST	HITMRST	MISSMEN	HITMEN
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WAYSEL	CACHEINV	EN
													rw	w	rw

位 31:20 保留，必须保持复位值。

位 19 **MISSMRST**：未命中监视器复位 (miss monitor reset)
0：无影响
1：复位缓存未命中监视器

位 18 **HITMRST**：命中监视器复位 (hit monitor reset)
0：无影响
1：复位缓存命中监视器

位 17 **MISSMEN**：未命中监视器使能 (miss monitor enable)
0：关闭缓存未命中监视器。停止监视器并不会将其复位。
1：使能缓存未命中监视器

位 16 **HITMEN**：命中监视器使能 (hit monitor enable)
0：关闭缓存命中监视器。停止监视器并不会将其复位。
1：使能缓存命中监视器

位 15:3 保留，必须保持复位值。

位 2 **WAYSEL**：缓存组相联模式选择 (cache associativity mode selection)
该位允许用户选择 ICACHE 组相联。只能在缓存已禁止时 (EN = 0) 通过软件写入。
0：直接映射缓存（单通路缓存）
1：n 路组相联缓存（复位值）

位 1 **CACHEINV**：缓存失效 (cache invalidation)
由软件置 1，当 BUSYF 标志置 1 时（缓存维护操作期间）由硬件清零。写入 0 无影响。
0：无影响
1：整个缓存失效（所有缓存行有效位 = 0）

位 0 **EN**：使能 (enable)
0：禁止缓存
1：使能缓存

8.7.2 ICACHE 状态寄存器 (ICACHE_SR)

ICACHE status register

偏移地址：0x004

复位值：0x0000 0001

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERRF	BSYENDF	BUSYF
													r	r	r

位 31:3 保留，必须保持复位值。

位 2 **ERRF**：缓存错误标志 (cache error flag)
0：无错误
1：操作（可缓存写）期间出错

位 1 **BSYENDF**：繁忙状态结束标志 (busy end flag)
0：缓存繁忙
1：CACHEINV 完全失效操作结束

位 0 **BUSYF**：繁忙标志 (busy flag)
0：缓存没在执行 CACHEINV 操作
1：缓存执行 CACHEINV 完全失效操作

8.7.3 ICACHE 中断使能寄存器 (ICACHE_IER)

ICACHE interrupt enable register

偏移地址：0x008

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERRIE	BSYENDIE	Res.
													rw	rw	

位 31:3 保留，必须保持复位值。

位 2 **ERRIE**：缓存出错时中断使能 (interrupt enable on cache error)
由软件置 1，以使能在缓存出现功能错误（可缓存写访问）时产生中断
0：禁止在出错时产生中断
1：使能在出错时产生中断

位 1 **BSYENDIE**：繁忙状态结束时中断使能 (interrupt enable on busy end)
由软件置 1，以使能在缓存失效操作结束时产生中断。
0：禁止在繁忙状态结束时产生中断
1：使能在繁忙状态结束时产生中断

位 0 保留，必须保持复位值。

8.7.4 ICACHE 标志清零寄存器 (ICACHE_FCR)

ICACHE flag clear register

偏移地址：0x00C

复位值：0x0000 0000

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

位 31:3 保留，必须保持复位值。

位 2 **CERRF**：清零缓存错误标志 (clear cache error flag)
由软件置 1。
0：无影响
1：将 ICACHE_SR 中的 ERRF 标志清零

位 1 **CBSYENDF**：清零繁忙状态结束标志 (clear busy end flag)
由软件置 1。
0：无影响
1：将 ICACHE_SR 中的 BSYENDF 标志清零。

位 0 保留，必须保持复位值。

8.7.5 ICACHE 命中监视器寄存器 (ICACHE_HMONR)

ICACHE hit monitor register

偏移地址：0x010

复位值：0x0000 0000

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

位 31:0 **HITMON[31:0]**：缓存命中监视器计数器 (cache hit monitor counter)

8.7.6 ICACHE 未命中监视器寄存器 (ICACHE_MMONR)

ICACHE miss monitor register

偏移地址：0x014

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15:0 **MISSMON[15:0]**：缓存未命中监视器计数器 (cache miss monitor counter)

8.7.7 ICACHE 寄存器映射

表 50. ICACHE 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	ICACHE_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MISSMRST	HITMRST	MISSMEN	HITMEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WAYSEL	CACHEINV	EN
	Reset value													0	0	0	0														1	0	0
0x004	ICACHE_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERRF	BSYENDF	BUSYF
	Reset value																														0	0	1

表 50. ICACHE 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x008	ICACHE_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERRIE	BSYENDIE	Res.
	Reset value																														0	0	
0x00C	ICACHE_FCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CERRF	CBSYENDF	Res.
	Reset value																														0	0	
0x010	ICACHE_HMONR	HITMON[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x014	ICACHE_MMONR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MISSMON[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 2.2 节*。

9 电源控制 (PWR)

9.1 简介

电源控制器负责管理所有器件电源及功耗模式转换。

9.2 PWR 主要特性

电压控制器 (PWR) 的主要特性包括：

- 电源和电源域
  - 内核域 (V_{CORE})
  - V_{DD} 域
  - 备份域 (V_{SW})
  - 模拟域 (V_{DDA})
  - 为 9 个 I/O（PA8、PA9、PA15 和 PB3:8）供电的 V_{DDIO2} 电源
- 系统电源调压
  - 线性调压器 (LDO)
- 电源监控
  - POR/PDR 监视器
  - BOR 监视器
  - PVD 监视器
  - AVD 监视器
  - 超出功能范围的温度监视器
  - 超出功能范围的备份域电压监视器
- 电源管理
  - 工作模式
  - 电压调节控制
  - 低功耗模式
- V_{BAT} 电池充电
- 特权保护

9.3 PWR 引脚和内部信号

表 51. PWR 输入/输出引脚

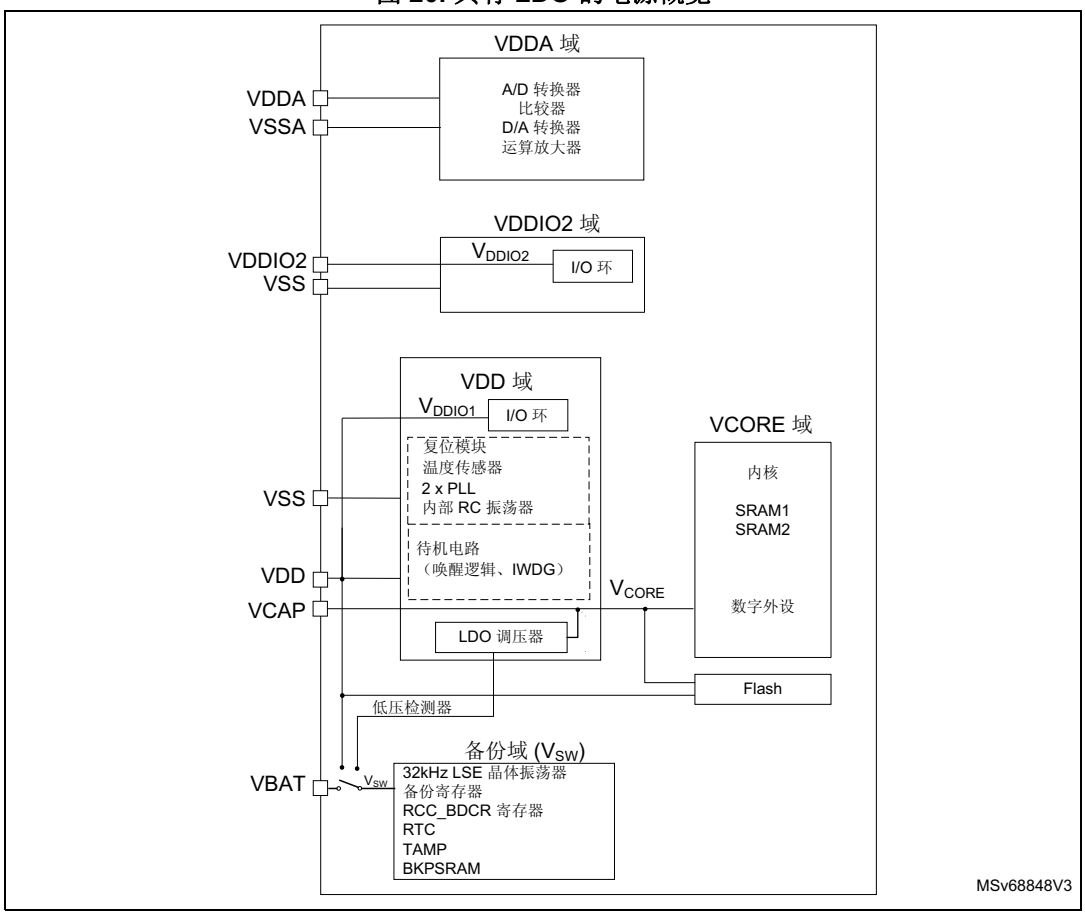
引脚名称	信号类型	说明
VDD	电源	主电源
GND	电源	主接地
VDDA	电源	模拟外设电源
VSSA	电源	模拟外设接地
VDDIO2	电源	独立 I/O 电源
VCAP	电源	逻辑电源 (V_{CORE})
VBAT	电源	备份域电源
VREF+	电源	ADC/DAC 高参考电压
VREF-	电源	ADC/DAC 低参考电压
WKUPx（x = 1 到 5）	输入	唤醒引脚
CSLEEP	输出	MCU 处于睡眠模式
CSTOP	输出	CPU 处于停止模式

表 52. PWR 内部输入/输出信号

内部信号名称	信号类型	说明
WKUPx（x = 1 到 5）	输入	唤醒事件源

9.4 PWR 电源和电源域

图 20. 具有 LDO 的电源概览



9.4.1 外部电源

器件需要 1.71 V 到 3.6 V 的 V_{DD} 工作电压电源。对于有特定电压需求的几个外设，可以单独提供相应的电源。这些电源必须使用 VDD 引脚上的有效工作电源提供：

- V_{DD} = 1.71 V 到 3.6 V

  V_{DD} 是为 I/O、内部调压器和系统模拟信号（如复位、电源管理和内部时钟）供电的外部电源。通过 VDD 引脚从外部提供。

- V_{DDA} = 1.62 V (ADC/COMP)/1.8 V (DAC)/2.0 V (OPAMP) 到 3.6 V

  V_{DDA} 是为 A/D 转换器和 D/A 转换器供电的外部模拟电源。V_{DDA} 电压与 V_{DD} 电压无关，不使用这些外设时，必须连接到 V_{DD}。

- V_{DDIO2} = 1.08 V 到 3.6 V

  V_{DDIO2} 是为 9 个 I/O（PA8、PA9、PA15 和 PB3:8）供电的外部电源。V_{DDIO2} 电压与 V_{DD} 电压无关，不使用这些引脚时，必须连接到 V_{DD}。

注意： *VDDIO2 引脚仅在一些封装中可用，更多信息，请参见器件数据手册。*

- V_{BAT} = 1.2 V 到 3.6 V
  当 V_{DD} 不存在时（通过电源开关），V_{BAT} 作为 RTC、外部时钟 32 kHz 振荡器、备份寄存器，以及备份 SRAM（可选）的电源。
- V_{CAP}：V_{CORE} 电源，其值取决于电压调节。
  通过 PWR_VOSCR 寄存器中的 VOS 位和 PWR_PMCR 寄存器中的 SVOS 位进行配置。
- V_{REF-} 和 V_{REF+}
  V_{REF+} 为 ADC 和 DAC 的输入参考电压。
  不提供单独的 VREF- 和 VREF+ 引脚，它们分别与 VSSA 和 VDDA 绑定。

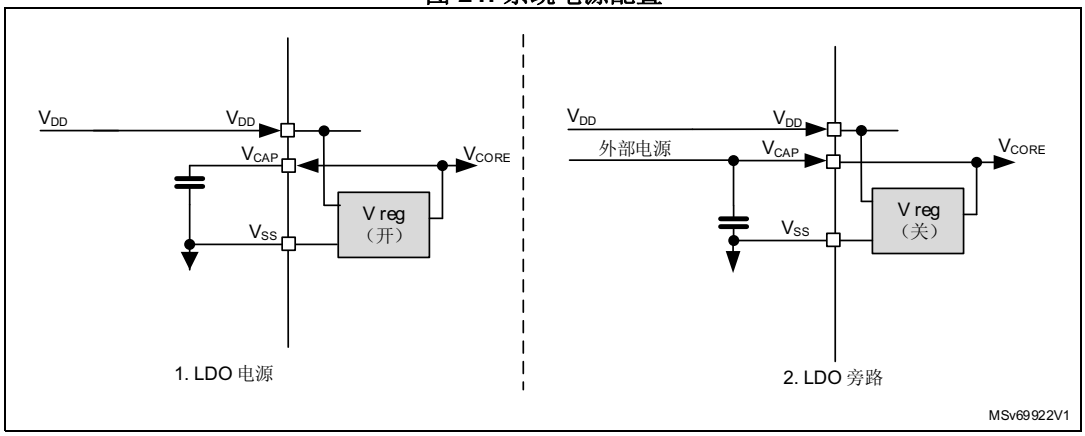
9.4.2 内部调压器

器件内置一个 LDO 调压器，用于为数字外设、SRAM（BKPSRAM 除外）和嵌入式 Flash 提供 V_{CORE} 电源。

LDO 在总外部电容典型值为 4.7 µF 的 VCAP（一个或两个引脚，视封装而定）上产生此电压。LDO 可以提供四个不同的电压（电压调节），并可以在停止模式下运行。

VCORE 内核域电源支持的电源配置如下图所示。

图 21. 系统电源配置



由外部电源提供 VCORE（旁路）时的启动情况

在旁路模式下提供 VCORE 时，VCORE 电压必须首先稳定在高于 1.1 V 的默认电压。由于上电后 LDO 默认为使能状态，因此外部 VCORE 电压必须保持在 1.1 V 以上，直到通过软件将 LDO 禁止。

禁止 LDO 后，外部 VCORE 电压可以根据用户应用需求进行调整（有关 VCORE 电压与最高工作频率的详细信息，请参见数据手册的*通用工作条件*部分）。

在旁路模式下工作时，应用程序必须通过 PWR_VOSCR 寄存器中的电源电压调节选择位 VOS[1:0] 来调整 VOS 电压。VOS[1:0] 必须根据外部提供的内核电压和相关性能来设置。

要调整 VOS 电压，软件必须依序选择各个中间电压。

要提高性能：

a) 必须先递增电压调节值（例如，当从 VOS3 更改为 VOS0 时，必须在 VOS[1:0] 位中选择以下电压：先选 VOS2，再选 VOS1，最后选 VOS0）。

b) 可以增加外部电压。

c) 可以提高系统频率。

要降低性能：

a) 必须降低系统频率。

b) 必须降低外部电压。

c) 随后可以递减电压调节值（例如，当从 VOS1 更改为 VOS3 时，必须在 VOS[1:0] 位中选择以下电压：先选 VOS2，再选 VOS3）。

9.4.3 上电和掉电序列

在上电和掉电阶段，必须遵守以下电源序列要求：

- 当 V_{DD} 低于 1 V 时，其他电源（V_{DDA} 和 V_{DDIO2}）必须保持低于 V_{DD} + 300 mV。
- 当 V_{DD} 高于 1 V 时，所有电源均独立。

在掉电阶段，只有当提供给 MCU 的能量保持在 1 mJ 以下时，V_{DD} 才能暂时低于其他电源。这样，外部去耦电容便可以在掉电瞬态阶段以不同的时间常量进行放电。

9.4.4 独立模拟外设电源

为了提高 ADC 和 DAC 转换精度、扩展供电的灵活性，模拟外设配有独立电源，可以单独滤波并屏蔽 PCB 上的噪声。

- 模拟外设电压源从单独的 VDDA 引脚输入。
- VSSA 引脚提供了独立的电源接地连接。

V_{DDA} 供电电压可与 V_{DD} 不同。在使能任一由 V_{DDA} 供电的模拟外设之前，必须检查是否存在 V_{DDA}。

电源电压监视可通过 PWR_VMSR 寄存器中的 AVDO 位在 VDDA 上实现（请参见*第 9.6.4 节：模拟电压检测器 (AVD)*）。

当使用单电源时，可从外部将 V_{DDA} 通过外部滤波电路连接到 V_{DD}，以得到无噪声的 V_{DDA} 参考电压。

模拟开关升压器

通过将 PWR_PMCR 寄存器中的 BOOSTE 和 AVD_READY 位置 1，应用可以在电源电压低于 2.7 V 时降低 I/O 模拟开关的总谐波失真 (THD)，从而保证与全电压范围时的性能相同。

为避免在 V_{DDA} < 2.7 V 且 V_{DD} > 2.7 V 时因升压器激活而导致电流消耗，可以通过将 PWR_PMCR 寄存器中的 AVD_READY 位保持为 0 来选择 V_{DD} 作为模拟开关的电源电压。

表 53. 推荐的模拟开关配置

VDDA	VDD	BOOSTE	AVD_READY	模拟开关电源电压
≤ 2.7 V	≤ 2.7 V	1	1	升压器
	> 2.7 V	0	0	VDD
> 2.7 V	无关	0	1	VDDA

9.4.5 USB 收发器电源

USB 收发器通过 V_{DD} 电源引脚供电。为确保正常工作，USB 需要 V_{DD} 范围介于 3.0 V 到 3.6 V 之间。

9.4.6 独立 I/O 电源轨

一些 I/O（PA8、PA9、PA15 和 PB3:8）由单独的电源轨供电。此电源轨的电源电压范围为 1.08 到 3.6 V，可通过 VDDIO2 引脚从外部提供。V_{DDIO2} 电压完全独立于 V_{DD} 或 V_{DDA}。VDDIO2 引脚仅在部分封装上可用。关于 I/O 列表，请参见相关器件数据手册中的引脚分配图表。

电源电压监视 (IO2VM) 可通过 PWR_VMSR 寄存器中的 VDDIO2RDY 位在 VDDIO2 上实现。

9.4.7 备份域

为了在 V_{DD} 掉电时，还能保留备份寄存器的内容并且让 RTC 继续工作，可将 VBAT 引脚连接到由电池或者其他电源供电的可选备用电源上。

VBAT 引脚为 RTC 单元、LSE 振荡器和 PC13 到 PC15 I/O 供电，允许 RTC 在主电源关闭时也可工作。当 *PWR 备份域控制寄存器 (PWR_BDCR)* 中的 BREN 位置 1 时，可选择通过 VBAT 引脚为备份 SRAM 供电。V_{BAT} 电源的开关由复位模块中内置的掉电复位电路进行控制。

警告： **在 $t_{RSTTEMPO}$（V_{DD} 启动后的一段延迟）期间或检测到 PDR 后，V_{BAT} 与 V_{DD} 之间的电源开关仍连接到 V_{BAT}。**
在启动阶段，如果 V_{DD} 的建立时间小于 $t_{RSTTEMPO}$（有关 $t_{RSTTEMPO}$ 的值，请参见数据手册）且 $V_{DD} > V_{BAT} + 0.6$ V，会有电流经由 V_{DD} 和电源开关 (V_{BAT}) 之间连接的内部二极管注入 V_{BAT} 引脚。
如果连接到 VBAT 引脚的电源/电池无法承受此注入电流，则强烈建议在该电源与 VBAT 引脚之间连接一个外部低压降二极管。

如果应用中没有使用任何外部电池，建议从外部将该 V_{BAT} 通过 100 nF 外部去耦电容连接到 V_{DD}。

通过 V_{BAT} 对备份域供电时（模拟开关连接到 V_{BAT}），以下引脚可用：

- PC13、PC14 和 PC15，可由 RTC 或 LSE 配置（请参见 *第 32.3 节：RTC 功能描述*）
- PC13 和 PA0，可通过 TAMP 外设配置为入侵引脚。

注意： *由于模拟开关仅能传递有限的电流，因此使用输出模式的 GPIO PC13 到 PC15 是受限的：速率必须限制在 2 MHz，最大负载为 30 pF，并且这些 I/O 不得作为电流源使用（例如，驱动 LED）。*

备份域访问

系统复位后，备份域（RCC 备份域控制寄存器 RCC_BDCR、RTC 寄存器、TAMP 寄存器、备份寄存器和备份 SRAM）将受到保护，以防止意外的写访问。要访问备份域，需将 *PWR 禁止备份保护控制寄存器 (PWR_DBPCR)* 中的 DBP 位置 1 以使能对备份域的访问。

备份 RAM

备份域包括可在 32 位、16 位或 8 位数据模式下访问的 2 KB 备份 RAM。备份 RAM 通过备份域中的备份调压器供电。当备份调压器通过 PWR_BDCR 中的 BREN 位使能时，即使处于待机和/或 VBAT 模式，备份 RAM 的内容也能保留（如果 VBAT 始终存在，则可将其视作内部 EEPROM）。

备份调压器既可以处于开启状态，也可以处于关闭状态，具体取决于应用在待机模式或 VBAT 模式下是否需要备份 RAM 功能。

备份 RAM 受读保护，在发生入侵事件时会被批量擦除，这样可以防止对加密私钥等机密数据进行访问。

可通过以下方式擦除备份 RAM：

- 在产品状态完全退回后通过 Flash 接口
- 入侵事件发生后
- 备份域复位后

V_{BAT} 电池充电

当 V_{DD} 存在时，可通过内部电阻为 VBAT 上的外部电池充电。

V_{BAT} 充电可通过一个 5 kΩ 的电阻或 1.5 kΩ 的电阻来实现，具体取决于 PWR_BDCR 寄存器中 VBRS 位的值。

将 PWR_BDCR 寄存器中的 VBE 位置 1 可使能电池充电。在 V_{BAT} 模式下自动禁止。

9.5 PWR 系统电源调压

9.5.1 LDO 嵌入式调压器

调压器在上电复位时使能。要通过外部电源为 V_{CORE} 供电，可以将 PWR_SCCR 寄存器中的 BYPASS 位置 1 来禁止调压器。

BYPASS 位会在上电复位后写入一次。写入一次的机制将锁定寄存器，因此会忽略其他任何写访问。系统必须在写入新值前掉电再上电。

当 V_{CORE} 通过外部电源供电时，必须在 PWR_VOSCR 寄存器的 VOSx 位反映外部施加的电压。

LDO 调压器可以提供四个不同的电压（电压调节），并可以在停止模式下运行。

9.5.2 V_{CORE} 供电与复位、电压调节和低功耗模式

复位后，V_{CORE} 处于 VOS3。

退出停止模式或待机模式时，电压范围为 VOS3。

9.5.3 嵌入式调压器工作模式

有三种不同的电源模式：运行、停止和待机模式。

运行模式

调压器 (LDO) 为 V_{CORE} 域（内核、存储器和数字外设）提供全功率。调压器输出电压可通过软件调节为不同电压级别（VOS0、VOS1、VOS2 和 VOS3），这些级别通过 PWR 电压调节控制寄存器 (PWR_VOSCR) 中的 VOS 位配置。

当系统时钟频率低于最高频率时，VOS 电压调节特性可使功耗得到优化。默认情况下，系统复位后选择 VOS3。

可实时更改 VOSx 位来适应所需的系统性能。

停止模式

调压器 (LDO) 为 V_{CORE} 域供电以保留寄存器和内部存储器的内容。调压器模式通过 PWR 电源模式控制寄存器 (PWR_PMCR) 中的 SVOS 位选择。

使用 SVO4（电压低于 VOS3）可以进一步降低停止模式的功耗，使用 SVOS5 甚至可以更进一步地降低功耗。

待机模式

调压器 (LDO) 关闭且 V_{CORE} 域掉电。除待机电路和备份域外，寄存器和存储器的内容都将丢失。

9.6 PWR 电源和温度监视

可对以下电源执行电源电压监视：

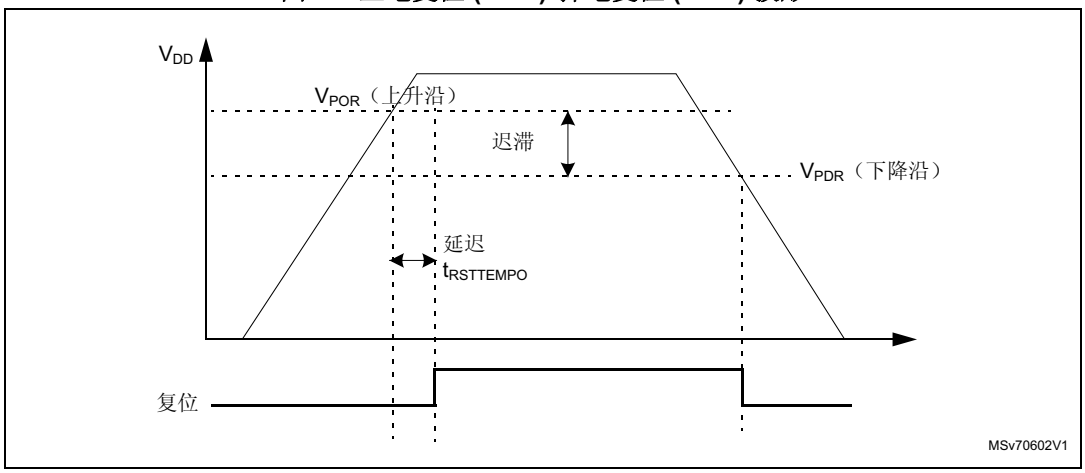
- VDD，通过 POR/PDR（请参见*第 9.6.1 节*）、BOR（请参见*第 9.6.2 节*）和 PVD 监视器（请参见*第 9.6.3 节*）监视
- VDDA，通过 AVD 监控器监视（请参见*第 9.6.4 节*）
- VDDIO2，通过 VDDIO2RDY 位监视（请参见*第 9.6.5 节*）
- VBAT，通过 VBAT 阈值监视（请参见*第 9.6.6 节*）
- 温度监视（请参见*第 9.6.7 节*）

9.6.1 上电复位 (POR)/掉电复位 (PDR)

系统内部集成有 POR/PDR 电路，可确保正常的启动操作。

当 VDD 低于指定的 VPOR 阈值时，系统无需外部复位电路便会保持复位模式。一旦 VDD 电源电压高于 VPOR 阈值，系统便会退出复位状态（请参见*图 22*）。有关上电/掉电复位阈值的详细信息，请参见数据手册的电气特性部分。

图 22. 上电复位 (POR)/掉电复位 (PDR) 波形



1. 有关阈值和迟滞值的信息，请参见数据手册。

9.6.2 欠压复位 (BOR)

上电期间，欠压复位 (BOR) 将使系统保持复位状态，直到 VDD 电源电压达到指定的 VBOR 阈值。

VBOR 阈值通过系统选项字节进行配置。BOR 默认处于关闭状态，但可通过将 BORH_EN 选项位置 1 来使能。

可以选择下列可编程 VBOR 阈值：

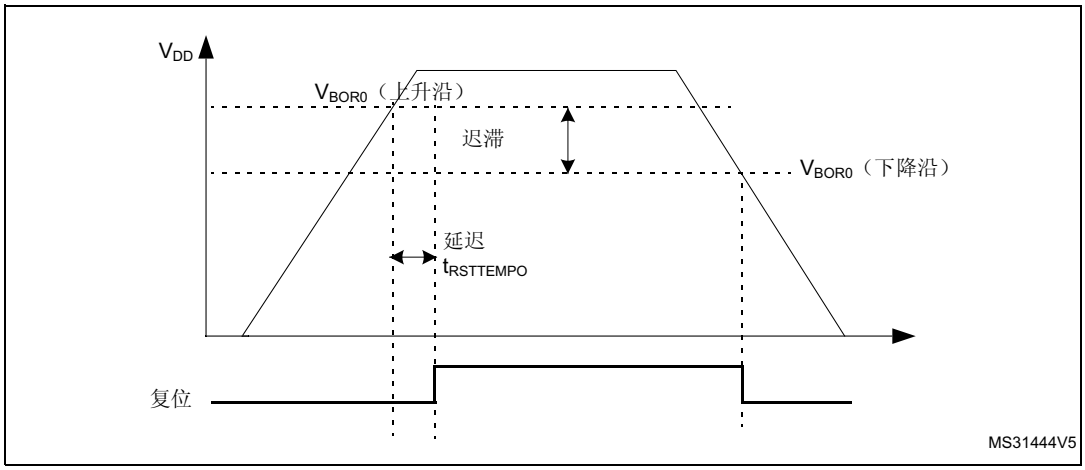
- BOR 关闭 (BORH_EN = 0)
- BOR 级别 1 (VBOR1)
- BOR 级别 2 (VBOR2)
- BOR 级别 3 (VBOR3)

有关欠压复位阈值的更多详细信息，请参见产品数据手册的“*电气特性*”部分。

BOR 使能且 V_{DD} 电源电压降至所选 VBOR 阈值以下时，将产生系统复位。

将 BORH_EN 选项位编程为 0 可以禁止 BOR。要禁止 BOR 功能，V_{DD} 必须高于 POR 阈值，以启动系统选项字节编程序列。禁止 BOR 后，就只能由 PDR 监视掉电过程（请参见*第 9.6.6 节*）

图 23. BOR 阈值



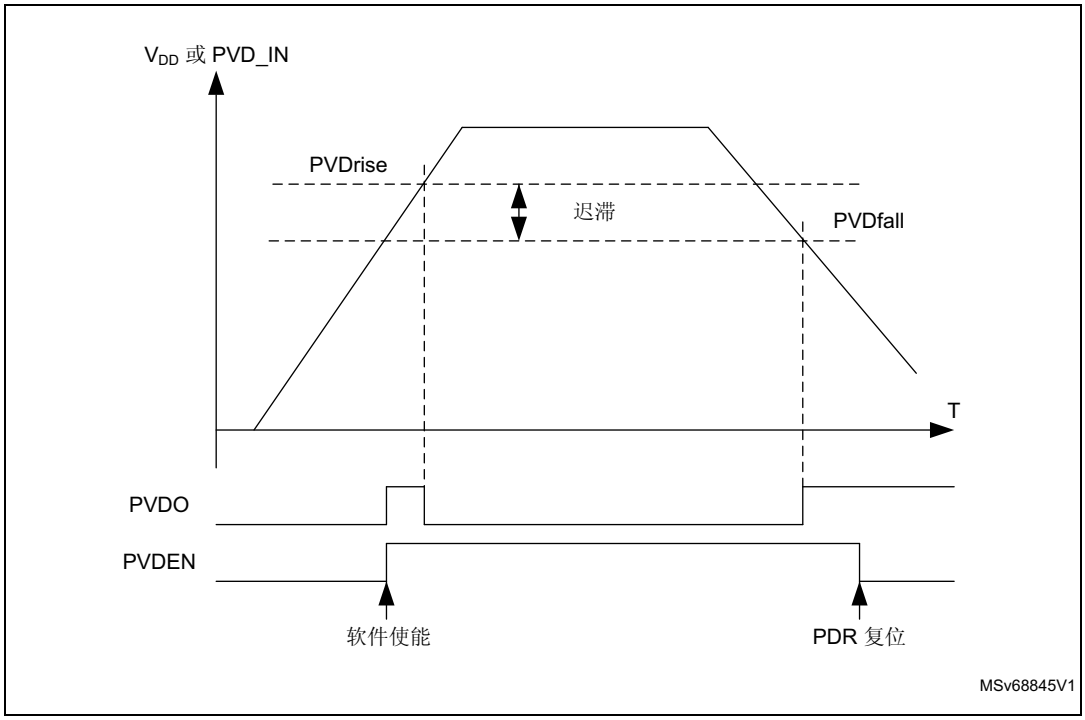
9.6.3 可编程电压检测器 (PVD)

可以使用 PVD 监视 V_{DD} 电源，方法是将其与 *PWR 电压监视器控制寄存器 (PWR_VMCR)* 中的 PLS[2:0] 位所选的阈值进行比较。也可以使用 PVD 来监视 PVD_IN 引脚上的电压。在这种情况下，会将 PVD_IN 电压与内部 VREFINT 值进行比较。

通过将 *PWR 电压监视器控制寄存器 (PWR_VMCR)* 中的 PVDE 位置 1 来使能 PVD。

PWR 电压监视器状态寄存器 (PWR_VMSR) 中提供了 PVDO 标志，用于指示 V_{DD} 或 PVD_IN 电压是大于还是小于 PVD 阈值。此事件内部连接到 EXTI，如果通过 EXTI 寄存器使能了中断，则可以产生中断。必须根据 PVD 的输出行为配置 EXTI 线上升沿/下降沿有效性，即，如果 EXTI 线配置为上升沿有效，则会在 V_{DD} 或 PVD_IN 电压下降到 PVD 阈值以下时产生中断。此功能的用处之一就是可以在服务程序中执行紧急关闭。

图 24. PVD 阈值



1. 有关阈值和迟滞值的信息，请参见数据手册。

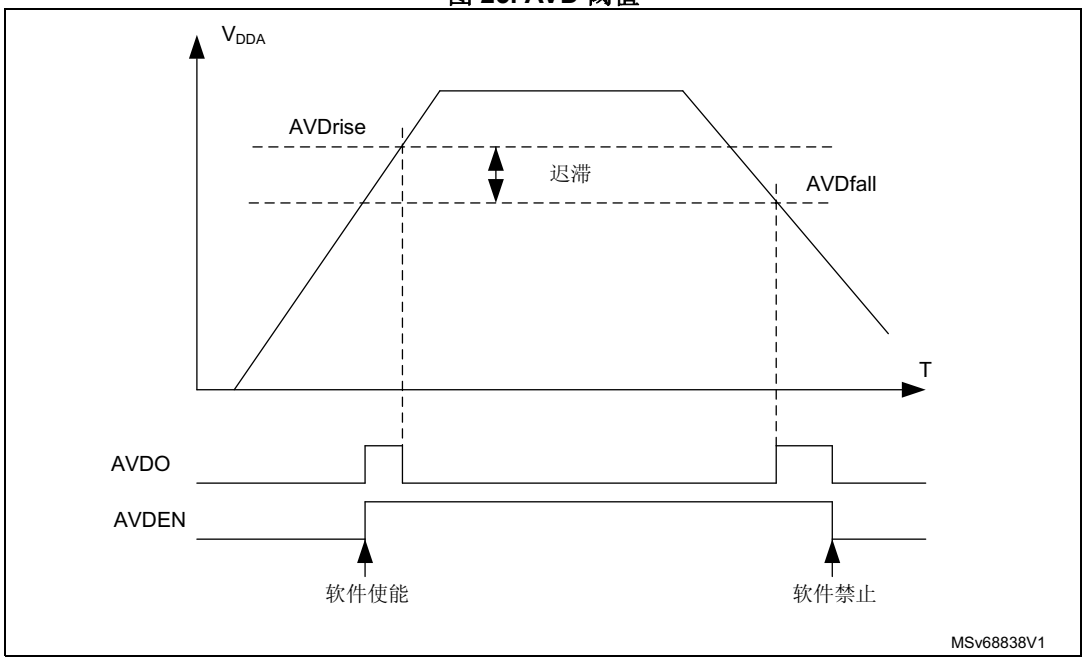
9.6.4 模拟电压检测器 (AVD)

可以使用 AVD 监视 V_{DDA} 电源，方法是将其与 *PWR 电压监视器控制寄存器 (PWR_VMCR)* 中的 ALS[1:0] 位所选的阈值进行比较。

通过将 *PWR 电压监视器控制寄存器 (PWR_VMCR)* 中的 AVDEN 位置 1 来使能 AVD。

PWR 电压监视器状态寄存器 (PWR_VMSR) 中提供了 AVDO 标志，用于指示 V_{DDA} 是大于还是小于 AVD 阈值。此事件内部连接到 EXTI，如果通过 EXTI 寄存器使能了中断，则可以产生中断。当 V_{DDA} 降至 AVD 阈值以下以及/或者当 V_{DDA} 升至 AVD 阈值以上时，可以产生 AVDO 中断，具体取决于 EXTI 上升沿/下降沿配置。例如，服务程序可以指示 V_{DDA} 电源何时降至最小值以下。

图 25. AVD 阈值



1. 有关阈值和迟滞值的信息，请参见数据手册。

9.6.5 VDDIO2 电压监视器 (IO2VM)

IO2VM 用于监视独立的电源电压 VDDIO2，以确保外设处于其功能电压范围内。VDDIO2RDY 标志（在*第 9.11.10 节：PWR 电压监视器状态寄存器 (PWR_VMSR)* 中提供）用于指示是否存在有效的 VDDIO2 电源。

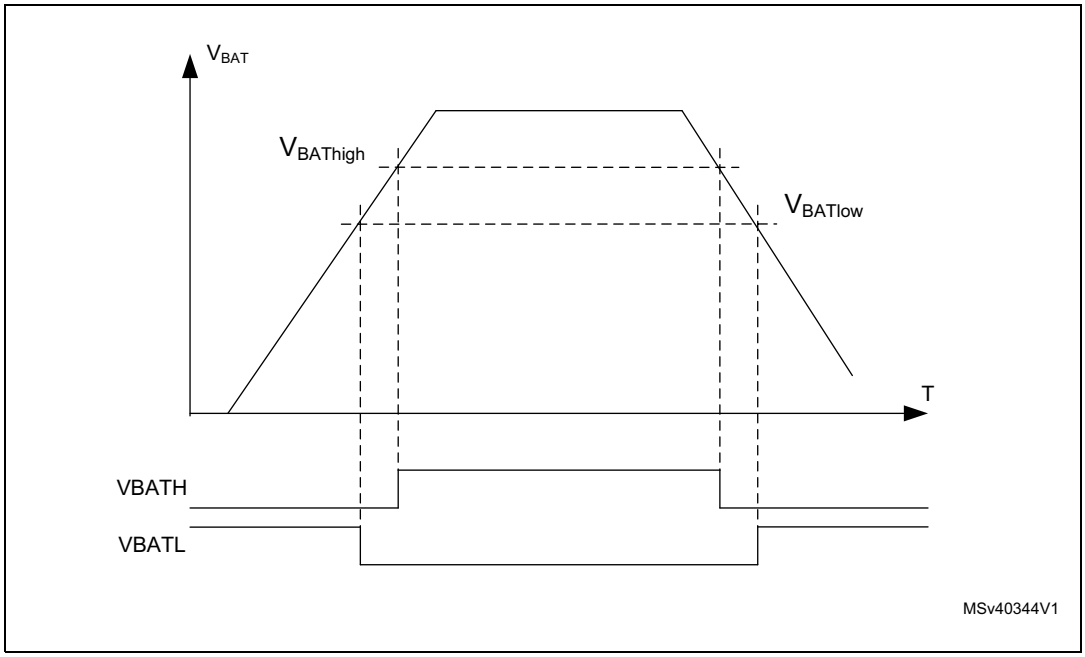
9.6.6 备份域电压监视

在 VBAT 模式下，电池电源电压（备份域）可以通过将其与以下两个阈值进行比较来监视：VBAThigh 和 VBATlow。可通过 *PWR 备份域控制寄存器 (PWR_BDCR)* 中的 MONEN 位使能/禁止 VBAT 监视。当使能时，电池电压阈值会增加功耗。

如果在 TAMP 外设中使能了备份域电压监视内部入侵（TAMP_CR1 寄存器中的 ITAMP1E = 1），则当电池电压超出功能范围上限时，将生成入侵事件。

注意： *VDD 如果存在，则它即是备份域电压，如果不存在，则备份域电压为 VBAT。*

图 26. VBAT 阈值



1. 有关阈值和迟滞值的信息，请参见数据手册。

9.6.7 温度监视

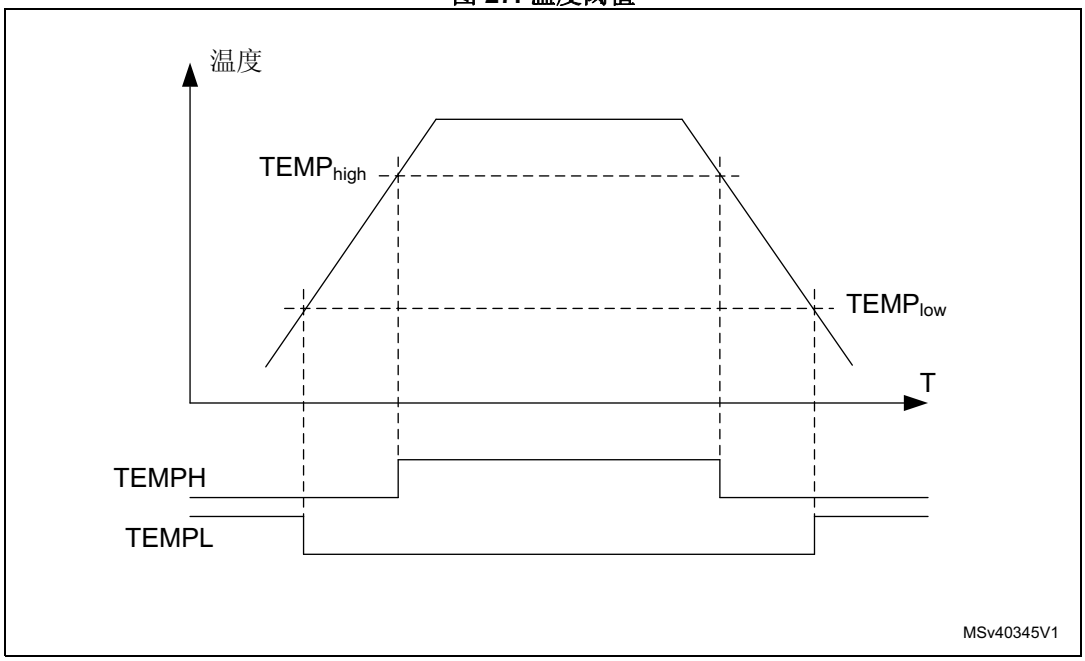
电源控制模块中内置专用的温度传感器单元。可以监视结温，方法是将其与 TEMPhigh 和 TEMPlow 两个阈值进行比较。*PWR 备份域状态寄存器 (PWR_BDSR)* 中的 TEMPH 和 TEMPL 标志用于指示器件温度是大于还是小于阈值。可通过 *PWR 备份域控制寄存器 (PWR_BDCR)* 中的 MONEN 位使能/禁止温度监视。

当使能时，温度阈值会增加功耗。例如，温度值可用于触发程序以执行温度控制任务。

如果在 TAMP 外设中使能了温度监视内部入侵（TAMP_CR1 寄存器中的 ITAMP2E = 1），则当温度超出功能范围上限或下限时，将生成入侵事件。

RTC 入侵信号上提供了 TEMPH 和 TEMPL 唤醒中断（请参见*第 33 节：入侵和备份寄存器 (TAMP)*）。

图 27. 温度阈值



1. 有关阈值和迟滞值的信息，请参见数据手册。

9.7 PWR 电源管理

9.7.1 电压调节

调压器通过以下特性支持电压调节：

- 运行模式电压调节
  - VOS0：电压调节 0
  - VOS1：电压调节 1
  - VOS2：电压调节 2
  - VOS3：电压调节 3
- 停止模式电压调节
  - SVOS3：电压调节 3
  - SVOS4：电压调节 4
  - SVOS5：电压调节 5

有关电压调节值的详细信息，请参见产品数据手册。

复位后，系统以最低的运行模式电压调节 (VOS3) 启动。之后，可通过软件根据所需的系统性能编程 *PWR 电压调节控制寄存器 (PWR_VOSCR)* 中的 VOS 位，实时更改电压调节。当从停止模式或待机模式退出时，运行模式电压调节将复位为默认的 VOS3 值。

进入停止模式前，软件应在 *PWR 电源模式控制寄存器 (PWR_PMCR)* 中预先选择 SVOS 级别。SVOS4 和 SVOS5 的停止模式电压调节还会将调压器设置为低功耗模式，以进一步降低功耗。

9.7.2 电源管理示例

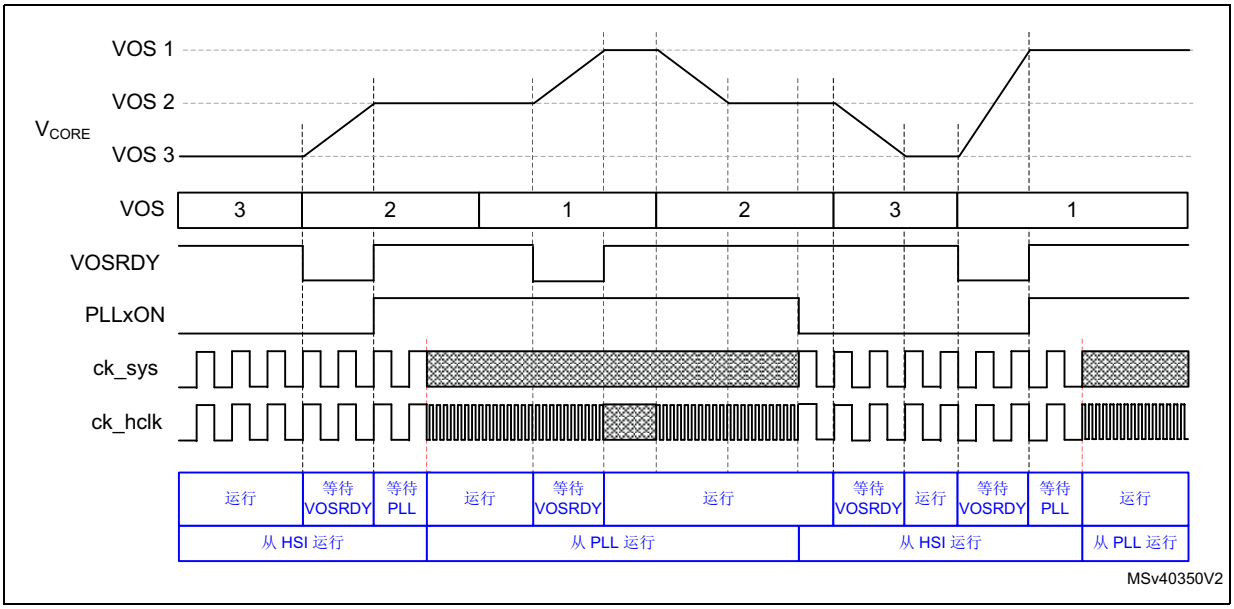
运行模式下的 VCORE 电压调节行为示例。

图 28 说明了以下系统操作序列：

1. 复位后，系统以 VOS3 从 HSI 启动。
2. 首先，通过 PLL 和电压调节 VOS2 使系统性能升到中速时钟。为此：
   a) 将电压调节编程到 VOS2。
   b) VCORE 电源达到 VOSRDY 指示的所需值后，通过使能 PLL 增大时钟频率。
   c) PLL 锁定后，切换系统时钟。
3. 之后，通过 PLL 和电压调节 VOS1 使系统性能升到高速时钟。为此：
   a) 将电压调节编程到 VOS1。
   b) VCORE 电源达到 VOSRDY 指示的所需值后，增大时钟频率。
4. 然后，通过电压调节 VOS2 使系统性能降到中速时钟。为此：
   a) 首先，减小系统频率。
   b) 将电压调节降为 VOS2。
5. 下一步是通过电压调节 VOS3 使系统性能降到 HSI 时钟。为此：
   a) 将时钟切换到 HSI。
   b) 禁止 PLL。
   c) 将电压调节降为 VOS3。
6. 之后，可通过 PLL 使系统性能升到高速时钟。为此：
   a) 将电压调节编程到 VOS1。
   b) VCORE 电源达到 VOSRDY 指示的所需值后，通过使能 PLL 增大时钟频率。
   c) PLL 锁定后，切换系统时钟。

当系统性能（时钟频率）发生变化时，应相应地设置 VOS，否则系统可能会不可靠。

图 28. 运行模式下的动态电压调节



9.8 电源模式

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

器件有以下几种低功耗模式：

- 睡眠模式：

  CPU 时钟关闭，包括 Cortex-M33 内核（例如 NVIC 和 SysTick）在内的所有外设都可以运行，并在发生中断或事件时唤醒 CPU。请参见*第 9.8.4 节：睡眠模式*。

- 停止模式：

  停止模式在保持 SRAM 和寄存器内容的同时，实现最低功耗。内核域中的所有时钟都会停止。PLL、HSE 晶振、HSI（HSIKERON 置 1 时除外）、HSI48 和 CSI RC（CSIKERON 置 1 时除外）都处于禁止状态。LSE 或 LSI 仍在运行中。

  RTC 可以保持有效（带 RTC 的停止模式，不带 RTC 的停止模式）。

  当从停止模式退出时，系统时钟可以是 HSI（最多 64 MHz）

  或 CSI，具体取决于软件配置。

  请参见*第 9.8.5 节：停止模式*。

- 待机模式：

  待机模式下可以使用 BOR 达到最低功耗。此时，内部调压器关闭，因此内核域将掉电。PLL、HSI RC、HSI48、CSI RC 和 HSE 晶振也会关闭。

  RTC 可以保持有效（带 RTC 的待机模式，不带 RTC 的待机模式）。

  在待机模式下，欠压复位 (BOR) 时钟保持工作状态。

  I/O（待机模式使用的 I/O 除外）的状态在待机模式下可以保留。

  进入待机模式后，除备份域和待机电路中的寄存器和备份 SRAM 外，SRAM 和寄存器内容都将丢失。

  发生外部复位（NRST 引脚）、IWDG 复位、WKUP 引脚事件（可配置的上升沿或下降沿）、RTC 事件（闹钟、周期性唤醒、时间戳），或者入侵检测时，器件退出待机模式。入侵检测可由外部引脚或者内部故障检测触发。

  唤醒后的系统时钟为 HSI (32 MHz)。

  请参见*第 9.8.6 节：待机模式*。

此外，可通过下列方法之一降低运行模式下的功耗：

- 降低系统时钟速度，将电压调节配置为低功耗范围。
- 不使用 APB 和 AHB 外设时，将对应的外设时钟关闭。

下表显示了功耗模式概览。

表 54. 低功耗模式汇总(1)

模式名称	进入	唤醒源(2)	唤醒系统时钟	对时钟的影响	调压器
睡眠（立即睡眠或退出时睡眠）	WFI 或从 ISR 返回	任意中断	与进入睡眠模式之前相同	CPU 时钟关闭 对其他时钟或模拟时钟源无影响	VOS3、VOS2、VOS1 或 VOS0
	WFE	唤醒事件			
停止	LPMS = 0 + SLEEPDEEP 位 + WFI 或从 ISR 或 WFE 返回	任意 EXTI 线（在 EXTI 寄存器中配置） 特定外设事件	当 RCC_CFGR 中 STOPWUCK = 1 时为 CSI 当 STOPWUCK = 0 时为 HSI，频率是进入停止模式前的频率（最高 64 MHz）	除 LSI 和 LSE 之外的所有时钟关闭 HSI 或 CSI 可在软件请求时临时使能	SVOS3、SVOS4 或 SVOS5
待机	LPMS = 1 + SLEEPDEEP 位 + WFI 或从 ISR 或 WFE 返回	WKUP 引脚边沿、RTC 事件、NRST 引脚的外部复位、IWDG 复位	32 MHz HSI 时钟	除 LSI 和 LSE 之外的所有时钟关闭	关闭

1. 外设可以将系统从停止模式唤醒，但只有在进入停止模式之前选择了 SVOS3 时才可能实现。请参见*表 55*。功能取决于工作模式。
2. 请参见下一个表格。

表 55. 功能取决于工作模式(1)

外设	运行	睡眠	停止				待机		VBAT
			可用	唤醒能力			可用	唤醒能力	
				SVOS3	SVOS4	SVOS5			
CPU	Y	-	-	-	-	-	-	-	-
Flash (128K MB)	O	O	-(2)	-	-	-	-	-	-
SRAM1 (16 KB)	Y	Y(3)	O(4)	-	-	-	-	-	-
SRAM2 (16 KB)	Y(3)	Y(3)	O(4)	-	-	-	-	-	-
BKPSRAM	O	O	O	-	-	-	O	-	O
备份寄存器	Y	Y	Y	-	-	-	Y	-	Y
欠压复位 (BOR)	Y	Y	Y	O	O	O	Y	O	-
可编程电压检测器 (PVD)	O	O	O	O	O	O	-	-	-
模拟电压检测器 (AVD)	O	O	O	O	O	O	-	-	-
GPDMA	O	O	-	-	-	-	-	-	-
高速内部时钟 (HSI)	O	O	O	-	-	-	-	-	-
振荡器 HSI48	O	O	-	-	-	-	-	-	-
高速外部时钟 (HSE)	O	O	-	-	-	-	-	-	-
低速内部时钟 (LSI)	O	O	O	O	-	-	O	-	-
低速外部时钟 (LSE)	O	O	O	-	-	-	O	-	O
低功耗 RC 振荡器 (CSI)	O	O	O	-	-	-	-	-	-
HSE 上的时钟安全系统 (CSS)	O	O	-	-	-	-	-	-	-

表 55. 功能取决于工作模式(1)（续）

外设	运行	睡眠	停止				待机		VBAT
			可用	唤醒能力			可用	唤醒能力	
				SVOS3	SVOS4	SVOS5			
LSE 上的时钟安全系统	O	O	O	O	O(5)	O(5)	O	O	O
备份域电压监视、温度监视	O	O	O	O	O(5)	O(5)	O	O	O
RTC/TAMP	O	O	O	O	O	O	O	O	O
TAMP 引脚数	2	2	2	2	2	2	2	2	2
USB	O	O	O	O	-	-	-	-	-
USARTx (x = 1,2,3)	O	O	O	O	-	-	-	-	-
低功耗 UART (LPUART)	O	O	O	O	-	-	-	-	-
I2Cx (x = 1,2)	O	O	O	O	-	-	-	-	-
I3Cx (x = 1,2)	O	O	O	O	-	-	-	-	-
SPIx (x = 1,2,3)	O	O	O	O	-	-	-	-	-
FDCAN1	O	O	-	-	-	-	-	-	-
ADC1	O	O	-	-	-	-	-	-	-
DAC1（2 个转换器）	O	O	O	-	-	-	-	-	-
COMP1	O	O	O	O	-	-	-	-	-
OPAMP1	O	O	-	-	-	-	-	-	-
温度传感器 (DTS)	O	O	O	O	-	-	-	-	-
定时器 (TIMx)	O	O	-	-	-	-	-	-	-
低功耗定时器 LPTIMx (x = 1,2)	O	O	O	O	-	-	-	-	-
独立看门狗 (IWDG)	O	O	O	O	O	O	O	O	-
窗口看门狗 (WWDG)	O	O	-	-	-	-	-	-	-
SysTick 定时器 (SYSTICK)	O	O	-	-	-	-	-	-	-
随机数发生器 (RNG)	O	O	-	-	-	-	-	-	-
HASH 加速器	O	O	-	-	-	-	-	-	-
CRC 计算单元	O	O	-	-	-	-	-	-	-
GPIO	O	O	-	-	-	-	-(6)	O(7)	-
EXTI	O	O	O	O	O	O	-	-	-

1. Y = 支持（使能）。O = 可选（默认禁止，可由软件使能）。- = 不提供。
2. Flash 可配置为低功耗模式。默认情况下，不会在停止模式下进入低功耗模式（SVOS3、SVOS4）。
3. SRAM 时钟可以门控方式独立开启或关闭。默认情况下，时钟在运行和睡眠模式下处于使能状态。
4. SRAM 可以独立上电或掉电。默认情况下，不会在停止模式下进入掉电模式。
5. 通过内部入侵事件唤醒。
6. GPIO 状态在待机模式下可以保留，但默认不保留。
7. PA0、PA2、PB7、PC1 和 PC13 这 5 个引脚可以从待机模式唤醒。

调试模式

默认情况下，如果使用调试功能时应用程序将 MCU 置于停止模式或待机模式，调试连接将中断。这是因为 Cortex-M33 内核不再受时钟驱动。

不过，通过设置 DBGMCU 控制寄存器中的一些配置位，即使 MCU 进入低功耗模式，仍可使用软件对其进行调试。更多详细信息，请参见*第 41.2.5 节：调试和低功耗模式*。

9.8.1 降低系统时钟速度

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

更多详细信息，请参见*第 10 章：复位和时钟控制 (RCC)*。

9.8.2 外设时钟门控

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

要进一步降低睡眠模式的功耗，可在执行 WFI 或 WFE 指令之前禁止外设时钟。

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

在睡眠模式下，复位 RCC_AHBxLPENR 和 RCC_APBxLPENR 寄存器中的对应位可以自动禁止外设时钟。

9.8.3 低功耗模式

进入低功耗模式

MCU 通过执行 WFI（等待中断）或 WFE（等待事件）指令进入低功耗模式，或者当*从 ISR 返回*后，Cortex-M33 系统控制寄存器中的 SLEEPONEXIT 位置 1 时，MCU 也将进入低功耗模式。

仅当没有挂起的中断或事件时，才能通过 WFI 或 WFE 进入低功耗模式。

退出低功耗模式

MCU 根据进入睡眠或停止模式的方式退出这两种低功耗模式：

- 如果使用 WFI 指令或从 ISR 返回进入低功耗模式，通过 NVIC 确认的任何外设中断都能唤醒器件。
- 如果使用 WFE 指令进入低功耗模式，MCU 将在有事件发生时立即退出低功耗模式。唤醒事件可通过以下方式产生：
  - NVIC IRQ 中断：
    当 Cortex-M33 系统控制寄存器中的 SEVONPEND = 0 时
    通过在外设控制寄存器和 NVIC 中使能中断。当 MCU 从 WFE 恢复时，外设中断挂起位和 NVIC 外设 IRQ 通道挂起位（在 NVIC 中断清零挂起寄存器中）必须清零。只有当 NVIC 中断的优先级足够高时，才能唤醒和中断 MCU。

    当 Cortex-M33 系统控制寄存器中的 SEVONPEND = 1 时
    通过在外设控制寄存器和 NVIC（可选）中使能中断。当 MCU 从 WFE 恢复时，外设中断挂起位和使能的 NVIC 外设 IRQ 通道挂起位（在 NVIC 中断清零挂起寄存器中）必须清零。所有 NVIC 中断（甚至已禁止的 NVIC 中断）都将唤醒 MCU。只有当已使能 NVIC 中断的优先级足够高时，才能唤醒和中断 MCU。

– 事件：
将一个 EXTI 线配置为事件模式。当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置 1，不必清零 EXTI 外设的中断挂起位或 NVIC IRQ 通道挂起位。可能需要清零外设中的中断标志。

当发生外部复位（NRST 引脚）、IWDG 复位、使能的 WKUPx 引脚上出现上升沿或者触发 RTC/TAMP 事件时，MCU 退出待机模式（请参见*图 368：RTC 框图*）。

从待机模式唤醒后，程序将按照复位（启动引脚采样、选项字节加载和复位向量获取）后的方式重新执行。

小心： 当器件处于停止模式时，外设中断依赖于内部振荡器进行工作。必须使能相应的 NVIC 中断通道，以允许器件通过中断退出停止模式。如果仅禁止 NVIC 通道而同时保持外设中断使能，则无法禁止外设中断，因为器件可能会在时钟开启的情况下保持在停止模式下。

9.8.4 睡眠模式

睡眠模式下的 I/O 状态

在睡眠模式下，所有 I/O 引脚的状态与运行模式下相同。

进入睡眠模式

当 Cortex-M33 系统控制寄存器中的 SLEEPDEEP 位清零时，MCU 会按照*进入低功耗模式*所述进入睡眠模式（有关如何进入睡眠模式的详细信息，请参见下表）。

退出睡眠模式

MCU 会按照*退出低功耗模式*所述退出睡眠模式（有关如何退出睡眠模式的详细信息，请参见下表）。

表 56. 睡眠模式

睡眠模式	说明
进入模式	WFI（等待中断）或 WFE（等待事件），且： – SLEEPDEEP = 0 – 没有中断（对于 WFI）或事件（对于 WFE）挂起 请参见 Cortex-M33 系统控制寄存器。
	从 ISR 返回，且： – SLEEPDEEP = 0 及 – SLEEPONEXIT = 1 – 没有中断挂起 请参见 Cortex-M33 系统控制寄存器。
退出模式	如果使用 WFI 或从 ISR 返回进入 中断（请参见*表 96：STM32H503xx 向量表*） 如果使用 WFE 进入且 SEVONPEND = 0： 唤醒事件（请参见*第 17.3 节：EXTI 功能说明*） 如果使用 WFE 进入且 SEVONPEND = 1： 中断（当在 NVIC 中禁止时，请参见*表 96：STM32H503xx 向量表*）或唤醒事件（请参见*第 17.3 节：EXTI 功能说明*）
唤醒延迟	无

9.8.5 停止模式

停止模式基于 Cortex-M33 深度睡眠模式与外设时钟门控。调压器通过 SVOSx 位配置（退出系统停止模式时，所选的 SVOS4 和 SVOS5 值会额外增加一段启动延时）。在停止模式下，内核域中的所有时钟都会停止。PLL、HSI、HSI48、CSI 和 HSE 振荡器被禁止。

HSI 或 CSI 时钟可以在停止模式下保持使能状态，从而可快速用作外设的内核时钟。

所有 SRAM 和寄存器内容都将保留，但 SRAM 可以整体关闭，从而进一步降低功耗。用户可以通过 *PWR 电源模式控制寄存器 (PWR_PMCR)* 中的 xxSO 位来选择在停止模式下丢弃的存储器。

停止模式下 BOR 始终可用。

停止模式下的 I/O 状态

在停止模式下，所有 I/O 引脚的状态与运行模式下相同。

进入停止模式

当 Cortex-M33 系统控制寄存器中的 SLEEPDEEP 位置 1 时，MCU 会按照*进入低功耗模式*所述进入停止模式（有关如何进入停止模式的详细信息，请参见*表 57*）。

如果正在执行 Flash 编程，将延迟到存储器访问结束后再进入停止模式。

如果正在访问 APB 域，将延迟到 APB 访问结束后再进入停止模式。

在停止模式下，可以通过对各控制位进行编程来选择以下功能：

- 独立看门狗 (IWDG) 通过写入其密钥寄存器或使用硬件选项来启动。而且一旦启动便无法停止，除非复位（请参见*第 30.4 节：IWDG 功能描述*）。
- 实时时钟 (RTC) 通过 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 RTCEN 位进行配置。
- 内部 RC 振荡器 LSI 时钟通过 RCC_BCDR 中的 LSION 位进行配置。
- 外部 32.768 kHz 振荡器 (LSE) 通过 RCC_BCDR 中的 LSEON 位进行配置。

可以在停止模式下使用 AVD 和 PVD。如果不需要上述功能，可以通过软件禁止功能，以节省其功耗。

在停止模式下，ADC1、DAC1（两个通道）和温度传感器也会产生功耗，除非在进入停止模式前将其禁止。

退出停止模式

MCU 可通过使能 EXTI 中断或事件退出停止模式，具体取决于进入低功耗模式的方式。一些外设可以将系统从停止模式唤醒（请参见*表 99：EXTI 线连接*），但只有在进入停止模式之前选择了 SVOS3 时才可能实现（请参见*表 55：功能取决于工作模式*）。配置为 EXTI 外部中断或事件的 GPIO 引脚可以在包括 SVOS4 和 SVOS5 在内的所有电压调节值下将系统从停止模式唤醒。

注意： *当需要通过外设从停止模式唤醒时，必须选择 SVOS3。*

通过发出中断或唤醒事件退出停止模式时，如果 *RCC 时钟配置寄存器 1 (RCC_CFGR)* 中的 STOPWUCK 位置 1，则会选择 CSI 作为系统时钟。如果 STOPWUCK 清零，则会选择 HSI 振荡器作为系统时钟。当选择 CSI 作为唤醒系统时钟时，唤醒时间更短。选择 HSI 能够以更高频率（最高 64 MHz）进行唤醒操作。
MCU 可通过使能 EXTI 中断或事件退出停止模式，具体取决于进入低功耗模式的方式。

退出停止模式时，MCU 处于运行模式下 (VOS3)。

表 57. 停止模式

停止模式	说明
进入模式	WFI（等待中断）或 WFE（等待事件），且： – Cortex-M33 系统控制寄存器中的 SLEEPDEEP 位置 1 – 没有中断（对于 WFI）或事件（对于 WFE）挂起 – PWR_CR1 中的 LPMS = 000
	从 ISR 返回，且： – Cortex-M33 系统控制寄存器中的 SLEEPDEEP 位置 1 – SLEEPONEXIT = 1 – 没有中断挂起 – PWR_PMCR 中的 LPMS = 0
	注意：要进入停止模式，必须清零所有 EXTI 线挂起位（在 EXTI 上升沿挂起寄存器 (EXTI_RPR1) 中）和生成唤醒中断的外设标志。否则将忽略进入停止模式这一过程，继续执行程序。
退出模式	如果使用 WFI 或从 ISR 返回进入： - 任何配置为中断模式的 EXTI 线（必须在 NVIC 中使能对应的 EXTI 中断向量）。中断源可以是外部中断或具有唤醒功能的外设（请参见*表 96：STM32H503xx 向量表*）。 如果使用 WFE 进入且 SEVONPEND = 0： - 任何配置为事件模式的 EXTI 线（请参见*第 17.3 节：EXTI 功能说明*）。 如果使用 WFE 进入且 SEVONPEND = 1： - 任何配置为中断模式的 EXTI 线（即使在 NVIC 中禁止了对应的 EXTI 中断向量）。中断源可以是外部中断或具有唤醒功能的外设（请参见*表 96：STM32H503xx 向量表*）。 - 任何配置为事件模式的 EXTI 线（请参见*第 17.3 节：EXTI 功能说明*）。
	注意：所有外设时钟均必须使能才允许此外设产生从停止模式唤醒的中断（RCC 中的 [PERIPH]EN 和 [PERIPH]LPEN 位必须置 1，且必须选择功能无关的时钟）。
唤醒延迟	最长唤醒时间介于：HSI 或 CSI 唤醒时间与 Flash 从停止模式唤醒的时间之间。

9.8.6 待机模式

BOR 可以处于激活状态的最低功耗模式为待机模式。待机模式基于 Cortex-M33 深度睡眠模式，其中调压器处于禁止状态。PLL、HSI、HSI48、CSI 和 HSE 振荡器都会关闭。

除备份域和待机电路中的寄存器外，SRAM 和寄存器内容都将丢失（请参见*图 20：具有 LDO 的电源概览*）。

在待机模式下 BOR 始终可用。

待机模式下可以使用 PC13 的 RTC 输出功能。也可以为 LSE 使用 PC14 和 PC15 功能。5 个唤醒引脚（WKUPx，x = 1 到 5）和 4 个 RTC 入侵引脚可用。

待机模式下的 I/O 状态

在待机模式下，I/O 默认处于浮空状态。如果 PWR_IORETR 寄存器中的 IORETEN 位置 1，则会保留 I/O 输出状态。对所有 IO 使能 IO 保留模式，但支持待机功能的 IO 和 JTAG IO（PA13、PA14、PA15 和 PB4）除外。在进入待机模式时，会对输出的状态进行采样，并设置上拉电阻或下拉电阻以在待机模式下保持 IO 输出。

如果 PWR_IORETR 寄存器中的 JTAGIORETEN 位置 1，则会保留 I/O 输出状态。对 PA13、PA14、PA15 和 PB4 使能 IO 保留模式（默认在唤醒后不会使能 JTAG 上拉/下拉）。

从待机模式唤醒后，只要 IORETEN（或 JTAG IO 的 JTAGIORETEN）置 1，保留的状态（上拉/下拉）就会保持应用。

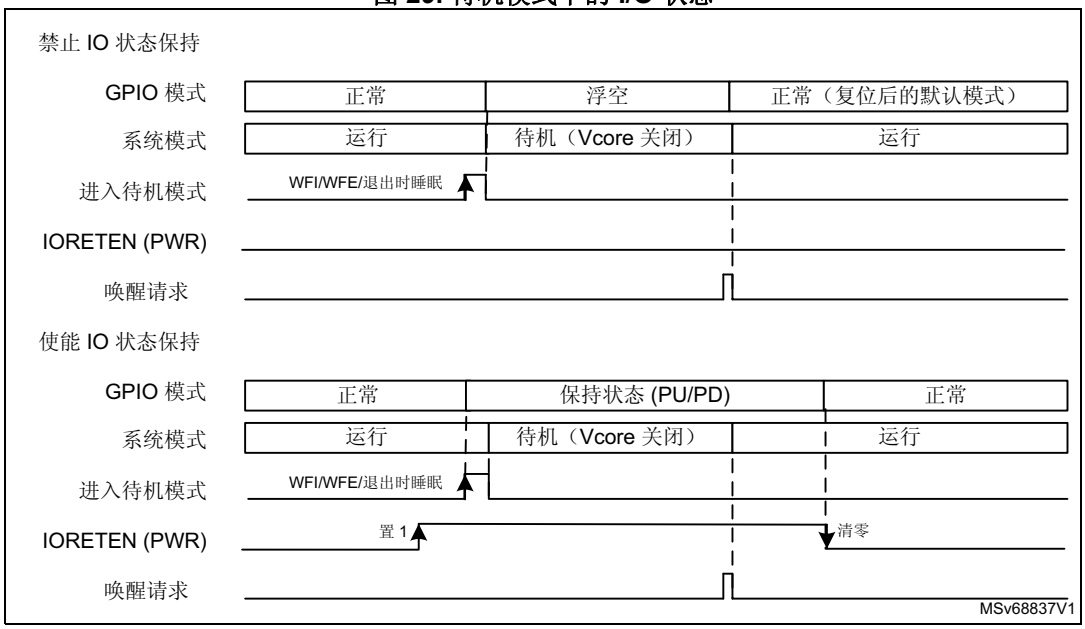
待机模式之前的 GPIO 引脚状态可通过 GPIO_IDR 寄存器来识别（当 GPIO 端口时钟和输入缓冲区都使能时）。

应用程序可通过将 IORETEN（或 JTAG IO 的 JTAGIORETEN）位清零来释放 IO 状态（清除保留的上拉/下拉），此操作可以在重新配置 GPIO 及相关外设之前或之后完成。

例如，应用程序可以在释放保留的状态之前将 GPIO 配置为已知状态。

注意： *仅保留 I/O 数字输出状态。*

图 29. 待机模式下的 I/O 状态



进入待机模式

当 Cortex-M33 系统控制寄存器中的 SLEEPDEEP 位置 1 时，MCU 便按照*进入低功耗模式*所述进入待机模式（有关如何进入待机模式的详细信息，请参见*表 58*）。

在待机模式下，可以通过对各控制位进行编程来选择以下功能：

- 独立看门狗 (IWDG) 通过写入其密钥寄存器或使用硬件选项来启动。而且一旦启动便无法停止，除非复位（请参见*第 30.4 节：IWDG 功能描述*）。
- 实时时钟 (RTC) 通过 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 RTCEN 位进行配置。
- 内部 RC 振荡器 LSI 时钟通过 RCC_BDCR 中的 LSION 位进行配置。
- 外部 32.768 kHz 振荡器 (LSE) 通过 RCC_BDCR 中的 LSEON 位进行配置。
- IO 保留模式通过 PWR_IORETR 寄存器中的 IORETEN 位进行配置。

退出待机模式

MCU 按照*退出低功耗模式*所述退出待机模式。*PWR 状态寄存器 (PWR_PMSR)* 中的 SBF 状态标志指示 MCU 已处于待机模式。除 *PWR 备份域控制寄存器 (PWR_BDCR)* 和 *PWR I/O 保留寄存器 (PWR_IORETR)* 外，所有寄存器都将在从待机模式唤醒后复位（有关如何退出待机模式的详细信息，请参见*表 58*）。

退出待机模式时，在待机期间通过 IORETEN 位保留的 I/O 输出状态仍会保持此配置，直到 PWR_IORETR 寄存器中的 IORETEN 位清零。一旦 IORETEN 清零，I/O 便会配置为复位值或根据 GPIOx_PUPDR 寄存器配置为上拉/下拉状态。

对于在复位后预定义上拉或下拉的 I/O（一些 JTAG/SWD I/O），如果其上拉或下拉与待机期间保留的值不同，将会同时应用下拉和上拉，直到 IORETEN 清零，此时会释放保留值。

此外，如果在从待机模式退出后编程的 GPIOx_PUPDR 值与待机期间保留的值不同，也会同时应用下拉和上拉，直到 IORETEN 清零，此时会释放保留值。

表 58. 待机模式

待机模式	说明
进入模式	WFI（等待中断）或 WFE（等待事件），且： – Cortex-M33 系统控制寄存器中的 SLEEPDEEP 位置 1 – 没有中断（对于 WFI）或事件（对于 WFE）挂起 – PWR_PMCR 中的 LPMS = 1 – PWR_WUSR 中的 WUFx 位清零
	从 ISR 返回，且： – Cortex-M33 系统控制寄存器中的 SLEEPDEEP 位置 1 – SLEEPONEXIT = 1 – 没有中断挂起 – PWR_PMCR 中的 LPMS = 1 – PWR_WUSR 中的 WUFx 位清零 – 将与所选唤醒源对应的 RTC/TAMP 标志清零
退出模式	WKUPx 引脚边沿、RTC 事件、NRST 引脚的外部复位、IWDG 复位、BOR 复位
唤醒延迟	复位阶段

9.8.7 功耗模式输出引脚

为了帮助进行调试，提供两种信号作为器件引脚备用功能：

- **CSLEEP**

  CSLEEP 置 1 时表示系统处于睡眠模式：已执行 WFI 或 WFE。

  CSLEEP 清零时表示系统处于运行模式。

- **CSTOP**

  CSTOP 置 1 时表示系统处于停止模式，即满足以下条件：

  – 已在 CPU SLEEPDEEP = 1 的条件下执行 WFI 或 WFE。

  – 系统中没有 AHB/APB 时钟在运行。

  CSTOP 清零时表示系统未处于停止模式：AHB/APB 时钟正在运行。

下表给出了不同信号状态与 MCU 功耗模式的关系。

表 59. 功耗模式输出状态与 MCU 功耗模式的关系

CSLEEP	CSTOP	MCU 功耗模式(1)
0	0	运行模式
1	0	睡眠模式
1	1	停止模式

1. CSLEEP 和 CSTOP 在内核域中生成，因此在待机模式下不会被驱动。

9.9 PWR 特权保护

默认情况下，复位后，除了 PWR_PRIVCFGR 之外的全部 PWR 寄存器都可以通过特权访问和非特权访问进行读/写操作，而 PWR_PRIVCFGR 寄存器只能通过特权访问进行写操作。PWR_PRIVCFGR 可通过特权访问和非特权访问进行读操作。

PWR_PRIVCFGR 的 NSPRIV 位只能通过特权访问进行写操作。该位用于配置*表 60：PWR 特权配置汇总*中定义的所有 PWR 功能的特权访问。

PWR_PRIVCFGR 中的 NSPRIV 置 1 时：

- PWR 特权位只能通过特权访问进行写操作。
- PWR 特权位只能通过特权访问进行读操作，但 PWR_PRIVCFGR 除外，该寄存器可以通过特权访问或非特权访问进行读操作。
- PWR_VOSSR、PWR_PMSR、PWR_VMSR、PWR_BDSR 和 PWR_WUSR 中的 VOSRDY 位可通过特权访问或非特权访问进行读操作。
- 对特权 PWR 位或寄存器进行的非特权访问将被丢弃：位读为零，并忽略对这些位的写操作 (RAZ/WI)。

表 60. PWR 特权配置汇总

PWR 功能	寄存器名称	特权位
低功耗模式	PWR_PMCR	所有位
备份域和 VBAT 模式配置	PWR_BDCR	所有位
	PWR_DBPCR	所有位
I/O 保留配置	PWR_IORETR	所有位

表 60. PWR 特权配置汇总（续）

PWR 功能	寄存器名称	特权位
唤醒引脚 WKUPx（x = 1 到 5）	PWR_WUCR	WUPENx
		WUPPx
		WUPPUPD
	PWR_WUSCR	CWUFx
电压调节 (VOS) 配置	PWR_VOSCR	VOS[1:0]
电压检测和监视	PWR_SCCR	所有位
	PWR_VMCR	所有位

9.10 PWR 中断

下表汇总了中断源及其控制方式。

表 61. PWR 中断请求

中断向量	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠、停止模式	退出待机模式
PVD/AVD 输出	可编程电压检测器（通过 EXTI 线 16）	PVDO/AVDO	使能 EXTI 线 16	写入 EXTI PIF16 = 1	是	否

9.11 PWR 寄存器

除非另外说明，否则可按字、半字和字节格式访问 PWR 寄存器。

9.11.1 PWR 电源模式控制寄存器 (PWR_PMCR)

PWR power mode control register

当 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x000

复位值：0x0000 000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	SRAM1SO	SRAM2SO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
					rw	rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	AVD_READY	BOOSTE	Res.	Res.	FLPS	Res.	CSSF	Res.	Res.	Res.	SVOS[1:0]		Res.	LPMS
		rw	rw			rw		rw				rw	rw		rw

位 31:27 保留，必须保持复位值。

位 26 **SRAM1SO**：停止模式下 AHB SRAM1 关断 (AHB SRAM1 shut-off in Stop mode)
0：在停止模式下，AHB RAM1 的内容将会保留。
1：在停止模式下，AHB RAM1 的内容将会丢失。

位 25 **SRAM2SO**：停止模式下 AHB SRAM2 关断 (AHB SRAM2 shut-off in Stop mode)
0：在停止模式下，AHB RAM2 的内容将会保留。
1：在停止模式下，AHB RAM2 的内容将会丢失。

位 24:14 保留，必须保持复位值。

位 13 **AVD_READY**：模拟电压就绪 (analog voltage ready)
该位仅在需要使能模拟开关升压时使用（请参见 BOOSTE 位）。
当预期的 V_{DDA} 模拟电源电压可用时，该位必须由软件置 1。
在将 AVDEN 位（PWR_VMCR 寄存器）置 1 并选择要监视的电源电压（ALS 位）之后，AVDO 位（PWR_VMSR 寄存器）会指示正确的模拟电源电压。
0：外设模拟电压 V_{DDA} 未就绪（默认）
1：外设模拟电压 V_{DDA} 就绪。

位 12 **BOOSTE**：模拟开关 V_{BOOST} 控制 (analog switch V_{BOOST} control)
该位用于使能升压器，以保证在 V_{DD} 电源电压低于 2.7 V 时模拟开关的交流性能（降低总谐波失真，以便在整个电源电压范围内实现相同的开关性能）。V_{DD} 电源电压可通过 PVD 和 PLS 位进行监视。
0：禁止升压器（默认）
1：如果模拟电压就绪，则使能升压器 (AVD_READY = 1)。

位 11:10 保留，必须保持复位值。

位 9 **FLPS**：停止模式下的 Flash 低功耗模式 (Flash memory low-power mode in Stop mode)
该位用于在退出停止模式时实现低功耗与重启时间之间的最佳平衡。
置 1 后，Flash 将在系统处于停止模式时进入低功耗模式。
0：当系统进入停止模式时，Flash 将保持正常模式（快速重启时间）。
1：当系统进入停止模式时，Flash 将进入低功耗模式（低功耗）。
注意：当系统在已使能 SVOS5 的情况下进入停止模式时，Flash 将自动强制进入低功耗模式。

位 8 保留，必须保持复位值。

位 7 **CSSF**：将待机和停止标志清零 (clear Standby and Stop flags)（始终读为 0）
该位通过硬件清零。
0：无影响
1：将 STOPF 和 SBF 标志清零。

位 6:4 保留，必须保持复位值。

位 3:2 **SVOS[1:0]**：系统停止模式电压调节选择 (system Stop mode voltage scaling selection)
这些位用于控制系统停止模式下的 V_{CORE} 电压级别，以在功耗与性能之间实现最佳平衡。
00：保留
01：SVOS5 电压调节 5
10：SVOS4 电压调节 4
11：SVOS3 电压调节 3（默认）

位 1 保留，必须保持复位值。

位 0 **LPMS**：低功耗模式选择 (low-power mode selection)
该位用于定义深度睡眠模式。
0：在进入深度睡眠时保持停止模式。
1：在进入深度睡眠时允许待机模式。

9.11.2 PWR 状态寄存器 (PWR_PMSR)

PWR status register

偏移地址：0x004

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SBF	STOPF	Res.	Res.	Res.	Res.	Res.
									r	r					

位 31:7 保留，必须保持复位值。

位 6 **SBF**：系统待机标志 (System standby flag)

该位由硬件置 1，并且只能通过 POR 或将 CSSF 位置 1 来清零。

0：系统未处于待机模式

1：系统已处于待机模式

位 5 **STOPF**：停止标志 (Stop flag)

该位由硬件置 1，并且只能通过复位或将 CSSF 位置 1 来清零。

0：系统未处于停止模式

1：系统已处于停止模式

位 4:0 保留，必须保持复位值。

9.11.3 PWR 电压调节控制寄存器 (PWR_VOSCR)

PWR voltage scaling control register

这些位域还可基于 PWR_PRIVCFGR 寄存器配置获得保护，以免遭到非特权访问。

偏移地址：0x010

复位值：0x0000 0000

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

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

位 31:6 保留，必须保持复位值。

位 5:4 **VOS[1:0]**：根据性能选择电压调节 (voltage scaling selection according to performance)
这些位用于控制 V_{CORE} 电压级别，以在功耗与性能之间实现最佳平衡：
- 在旁路模式下，这些位必须根据外部提供的内核电压及相关性能来设置。
- 提高性能时，必须在提高系统频率前更改电压调节。
- 降低性能时，必须先降低系统频率，再更改电压调节。

00：电压调节 3（默认）
01：电压调节 2
10：电压调节 1
11：电压调节 0

位 3:0 保留，必须保持复位值。

9.11.4 PWR 电压调节状态寄存器 (PWR_VOSSR)

PWR voltage scaling status register

偏移地址：0x014

复位值：0x0000 2008

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

位 31:16 保留，必须保持复位值。

位 15:14 **ACTVOS[1:0]**：当前应用于 V_{CORE} 的电压输出调节 (voltage output scaling currently applied to V_{CORE})
该位域用于提供最后一个 VOS 值。
00：VOS3（最低功耗）
01：VOS2
10：VOS1
11：VOS0（最高频率）

位 13 **ACTVOSRDY**：当前使用的 VOS 电压级别就绪 (Voltage level ready for currently used VOS)
0：V_{CORE} 高于或低于 ACTVOS[1:0] 提供的当前电压调节值。
1：V_{CORE} 等于 ACTVOS[1:0] 提供的当前电压调节值。

位 12:4 保留，必须保持复位值。

位 3 **VOSRDY**：V_{CORE} 电压调节输出选择就绪位 (Ready bit for V_{CORE} voltage scaling output selection)
0：未就绪，电压级别低于 VOS 选择值
1：已就绪，电压级别等于或高于 VOS 选择值。

注意：VOSRDY 标志只能在从低电压调节切换到高电压调节时使用（例如从 VOS3 切换到 VOS0）。

位 2:0 保留，必须保持复位值。

9.11.5 PWR 备份域控制寄存器 (PWR_BDCR)

PWR backup domain control register

当 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

从待机模式唤醒、RESET 信号和 VDD POR 不会复位该寄存器。只有 VSW POR 和 VSWRST 复位可以复位该寄存器。当 RCC_BDCR 寄存器中的 VSWRST 位复位 VSW 域时，不得访问该寄存器。

当 RCC_BDCR 寄存器中的 VSWRST 位复位 VSW 域时，不得访问 MONEN 和 BREN 位。复位后，该寄存器的 MONEN 和 BREN 位受写保护。修改其内容前，必须将 PWR_DBPCR 寄存器中的 DBP 位设置为禁止写保护。

偏移地址：0x020

上电复位值：0x0000 0000

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

位 31:10 保留，必须保持复位值。

位 9 **VBRS**：V_{BAT} 充电电阻选择 (V_{BAT} charging resistor selection)
0：通过 5 kΩ 电阻为 V_{BAT} 充电。
1：通过 1.5 kΩ 电阻为 V_{BAT} 充电。

位 8 **VBE**：V_{BAT} 充电使能 (V_{BAT} charging enable)
0：禁止 V_{BAT} 电池充电。
1：使能 V_{BAT} 电池充电。
注意：仅通过 POR 复位。

位 7:2 保留，必须保持复位值。

位 1 **MONEN**：备份域电压和温度监视使能 (Backup domain voltage and temperature monitoring enable)
0：禁止备份域电压和温度监视
1：使能备份域电压和温度监视

位 0 **BREN**：在待机和 V_{BAT} 模式下保留备份 RAM (Backup RAM retention in Standby and V_{BAT} modes)
当该位置 1 时，使能备份调压器（用于在待机模式和 V_{BAT} 模式下保持备份 RAM 的内容）。如果 BREN 清零，则备份调压器关闭。备份 RAM 仍可在运行和停止模式下使用。但其内容会在待机和 V_{BAT} 模式下丢失。
如果 BREN 置 1，则在备份调压器就绪标志 (BRRDY) 置 1 以指示在待机和 V_{BAT} 模式下会保持写入 SRAM 中的数据之前，应用程序必须等待。
0：在待机和 V_{BAT} 模式下，备份 RAM 内容丢失
1：在待机和 V_{BAT} 模式下，备份 RAM 内容保留

9.11.6 PWR 禁止备份保护控制寄存器 (PWR_DBPCR)

PWR disable backup protection control register

当 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x024

复位值：0x0000 0000

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

位 31:1 保留，必须保持复位值。

位 0 **DBP**：禁止备份域写保护 (Disable backup domain write protection)

在复位状态下，备份域中的所有寄存器和 SRAM 均受到保护，防止收到寄生写访问。必须将该位置 1 才能使能对这些寄存器的写访问。

0：禁止备份域写访问

1：使能备份域写访问

9.11.7 PWR 备份域状态寄存器 (PWR_BDSR)

PWR backup domain status register

从待机模式唤醒、RESET 信号和 VDD POR 不会复位该寄存器。只有 VSW POR 和 VSWRST 复位可以复位该寄存器。

偏移地址：0x028

复位值：0x0000 0000

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

位 31:24 保留，必须保持复位值。

位 23 **TEMPH**：温度级别监视与高阈值 (temperature level monitoring versus high threshold)

0：温度低于高阈值级别

1：温度等于或高于高阈值级别

位 22 **TEMPL**：温度级别监视与低阈值 (temperature level monitoring versus low threshold)

0：温度高于低阈值级别

1：温度等于或低于低阈值级别

位 21 **VBATH**：V_{BAT} 级别监视与高阈值 (V_{BAT} level monitoring versus high threshold)
0：V_{BAT} 级别低于高阈值级别
1：V_{BAT} 级别等于或高于高阈值级别

位 20 **VBATL**：V_{BAT} 级别监视与低阈值级别 (V_{BAT} level monitoring versus low threshold)
0：V_{BAT} 级别高于低阈值级别
1：V_{BAT} 级别等于或低于低阈值级别

位 19:17 保留，必须保持复位值。

位 16 **BRRDY**：备份调压器就绪 (backup regulator ready)
该位由硬件置 1，用于指示备份调压器已就绪。
0：备份调压器未就绪
1：备份调压器已就绪

位 15:0 保留，必须保持复位值。

9.11.8 PWR 电源配置控制寄存器 (PWR_SCCR)

PWR supply configuration control register

当 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x030

复位值：0x0000 0100

仅通过 POR 复位，而不会通过从待机模式唤醒和 POR 引脚复位。该寄存器的 BYPASS 位在 POR 后写入一次。写入一次的机制将锁定寄存器，因此会忽略其他任何写访问。系统必须在写入新值前掉电再上电。

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

位 31:9 保留，必须保持复位值。

位 8 **LDOEN**：LDO 使能 (LDO enable)
当封装使用 LDO 调压器时，该位值由硬件置 1。

位 7:1 保留，必须保持复位值。

位 0 **BYPASS**：电源管理单元旁路 (power management unit bypass)
0：电源管理单元正常工作。使用内部调压器。
1：电源管理单元旁路。使用外部电源（电压监视仍处于激活状态）。

9.11.9 PWR 电压监视器控制寄存器 (PWR_VMCR)

PWR voltage monitor control register

当 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

PVDE 和 PLS 位受锁定机制保护。锁定控制位于 SBS 模块中，此模块由 SBS_CFGR2 寄存器中的 PVDL 位控制。默认情况下，PVDE 和 PLS 处于解锁状态。

偏移地址：0x034

复位值：0x0000 0000

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

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

位 31:11 保留，必须保持复位值。

位 10:9 **ALS[1:0]**：模拟电压检测器 (AVD) 电压选择 (analog voltage detector (AVD) level selection)

这些位用于选择模拟电压检测器 (AVD) 的阈值。

00：AVD 电压 0（V_{AVD0} 约为 1.7 V）

01：AVD 电压 1（V_{AVD1} 约为 2.1 V）

10：AVD 电压 2（V_{AVD2} 约为 2.5 V）

11：AVD 电压 3（V_{AVD3} 约为 2.8 V）

位 8 **AVDEN**：V_{DDA} 上的外设电压监视器使能 (peripheral voltage monitor on V_{DDA} enable)

0：禁止 V_{DDA} 上的外设电压监视器

1：使能 V_{DDA} 上的外设电压监视器

位 7:4 保留，必须保持复位值。

位 3:1 **PLS[2:0]**：可编程电压检测器 (PVD) 电压选择 (programmable voltage detector (PVD) level selection)

这些位用于选择可编程电压检测器 (PVD) 的阈值。

000：PVD 电压 0（V_{PVD0} 约为 1.95 V）

001：PVD 电压 1（V_{PVD1} 约为 2.1 V）

010：PVD 电压 2（V_{PVD2} 约为 2.25 V）

011：PVD 电压 3（V_{PVD3} 约为 2.4 V）

100：PVD 电压 4（V_{PVD4} 约为 2.55 V）

101：PVD 电压 5（V_{PVD5} 约为 2.7 V）

110：PVD 电压 6（V_{PVD6} 约为 2.85 V）

111：PVD_IN 引脚

位 0 **PVDE**：PVD 使能 (PVD enable)

0：禁止 PVD

1：使能 PVD

9.11.10 PWR 电压监视器状态寄存器 (PWR_VMSR)

PWR voltage monitor status register

偏移地址：0x03C

复位值：0x0000 0000

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

位 31:23 保留，必须保持复位值。

位 22 **PVDO**：可编程电压检测输出 (programmable voltage detect output)
该位通过硬件置 1 和清零。仅当通过 PVDE 位使能 PVD 时该位才有效。
0：V_{DD} 等于或高于通过 PLS[2:0] 位选择的 PVD 阈值。
1：V_{DD} 低于通过 PLS[2:0] 位选择的 PVD 阈值。
注意：由于在待机模式下禁止 PVD，因此，进入待机模式或执行复位后，该位等于 0，直到 PVDE 位置 1。

位 21 保留，必须保持复位值。

位 20 **VDDIO2RDY**：V_{DDIO2} 上的电压检测器输出 (voltage detector output on V_{DDIO2})
该位通过硬件置 1 和清零。
0：V_{DDIO2} 低于 V_{DDIO2} 电压监视器的阈值。
1：V_{DDIO2} 等于或高于 V_{DDIO2} 电压监视器的阈值。

位 19 **AVDO**：V_{DDA} 上的模拟电压检测器输出 (analog voltage detector output on V_{DDA})
该位通过硬件置 1 和清零。仅当通过 AVDEN 位使能 VDDA 上的 AVD 时该位才有效。
0：V_{DDA} 等于或高于通过 ALS[2:0] 位选择的 AVD 阈值。
1：V_{DDA} 低于通过 ALS[2:0] 位选择的 AVD 阈值。
注意：由于在待机模式下禁止 AVD，因此，进入待机模式或执行复位后，该位等于 0，直到 AVDEN 位置 1。

位 18:0 保留，必须保持复位值。

9.11.11 PWR 唤醒状态清零寄存器 (PWR_WUSCR)

PWR wakeup status clear register

当 NSPRIV = 1 时，可以保护各个 CWUFx 位免遭非特权访问。

偏移地址：0x040

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CWUF5	CWUF4	CWUF3	CWUF2	CWUF1
											w	w	w	w	w

位 31:5 保留，必须保持复位值。

位 4:0 **CWUFx**：将 WUFx 唤醒引脚标志清零 (clear wakeup pin flag for WUFx)（x = 5 到 1）
该位始终读为 0
0：无影响
1：写入 1 会将 WUFx 唤醒引脚标志清零（该位通过硬件清零）

9.11.12 PWR 唤醒状态寄存器 (PWR_WUSR)

PWR wakeup status register

偏移地址：0x044

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUF5	WUF4	WUF3	WUF2	WUF1
											r	r	r	r	r

位 31:5 保留，必须保持复位值。

位 4:0 **WUFx**：WUFx 唤醒引脚标志 (wakeup pin WUFx flag)（x = 5 到 1）
该位由硬件置 1，只能通过 RESET 引脚或通过将 PWR_WUSCR 寄存器中的 CWUFx 位置 1 来清零。
0：未发生唤醒事件
1：从 WUFx 引脚接收到唤醒事件

9.11.13 PWR 唤醒配置寄存器 (PWR_WUCR)

PWR wakeup configuration register

当 NSPRIV = 1 时，可保护各个 WUPENx 位免遭非特权访问。

偏移地址：0x048

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	WUPPUPD5[1:0]		WUPPUPD4[1:0]		WUPPUPD3[1:0]		WUPPUPD2[1:0]		WUPPUPD1[1:0]	
						rw	w	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	WUPP5	WUPP4	WUPP3	WUPP2	WUPP1	Res.	Res.	Res.	WUPEN5	WUPEN4	WUPEN3	WUPEN2	WUPEN1
			rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

位 31:26 保留，必须保持复位值。

位 25:16 **WUPPUPDx[1:0]**：WKUPx 唤醒引脚上拉/下拉配置 (wakeup pin pull configuration for WKUPx)（x = 5 到 1）

这些位用于定义 WUPENx = 1 时所使用的 I/O 引脚上拉/下拉配置。相关的 GPIO 端口上拉/下拉配置必须设置为相同值或 00。唤醒引脚的上拉/下拉配置会在待机模式下保持不变。

00：无上拉

01：上拉

10：下拉

11：保留

位 15:13 保留，必须保持复位值。

位 12:8 **WUPPx**：WUPx 唤醒引脚极性位 (wakeup pin polarity bit for WUPx)（x = 5 到 1）

这些位用于定义在 WUPx 外部唤醒引脚上进行事件检测时所用的极性。

0：检测高电平（上升沿）

1：检测低电平（下降沿）

位 7:5 保留，必须保持复位值。

位 4:0 **WUPENx**：WUPx 唤醒引脚使能 (enable wakeup pin WUPx)（x = 5 到 1）

这些位由软件置 1 和清零。

0：WUPx 引脚上的事件不会将系统从待机模式唤醒。

1：WUPx 引脚上的上升沿或下降沿会将系统从待机模式唤醒。

注意：如果 WUPx 引脚电平在 WUPPx 选择上升沿时已为高电平，或者在 WUPPx 选择下降沿时已为低电平，则当 WUPx 引脚使能时（将 WUPENx 位置 1），会检测到额外的唤醒事件。

9.11.14 PWR I/O 保留寄存器 (PWR_IORETR)

PWR I/O retention register

当 NSPRIV = 1 时，可以保护该寄存器免遭非特权访问。

偏移地址：0x050

复位值：0x0001 0001

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

位 31:17 保留，必须保持复位值。

位 16 **JTAGIORETEN**：JTAG IO 的 IO 保留使能 (IO retention enable for JTAG IOs)
0：禁止 IO 保留模式。
1：对 PA13、PA14、PA15 和 PB4 使能 IO 保留模式。
在进入待机模式时，会对输出进行采样，并在待机功耗模式下应用于输出 IO

位 15:1 保留，必须保持复位值。

位 0 **IORETEN**：IO 保留使能 (IO retention enable)
0：禁止 IO 保留模式。
1：对所有 IO 使能 IO 保留模式，但支持待机功能的 I/O（PC13、PC14 和 PC15）和 JTAG I/O（PA13、PA14、PA15 和 PB4）除外。
在进入待机模式时，会对输出进行采样，并在待机电源模式下应用于输出 IO。
注意：如果 DBGMCU_CR 寄存器中的 DBG_STANDBY 位置 1，则不会保留 IO 状态。

9.11.15 PWR 特权配置寄存器 (PWR_PRIVCFGR)

PWR privilege configuration register

该寄存器只能通过特权访问进行写操作，但可以通过特权或非特权访问进行读操作。

偏移地址：0x104

复位值：0x0000 0000

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

位 31:2 保留，必须保持复位值。

位 1 **NSPRIV**：PWR 功能特权配置 (PWR functions privilege configuration)
由软件置 1 和复位。该位只能通过特权访问进行写操作。
0：可通过特权或非特权访问对 PWR 功能进行读写操作。
1：只能通过特权访问对 PWR 功能进行读写操作。

位 0 保留，必须保持复位值。

9.11.16 PWR 寄存器映射

表 62. PWR 寄存器映射和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**PWR_PMCR**	Res.	Res.	Res.	Res.	Res.	SRAM1SO	SRAM2SO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AVD_READY	BOOSTE	Res.	Res.	FLPS	Res.	CSSF	Res.	Res.	Res.	SVOS[1:0]		Res.	LPMS
	Reset value						0	0												0	0			0		0				1	1		0
0x004	**PWR_PMSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SBF	STOPF	Res.	Res.	Res.	Res.	Res.
	Reset value																										0	0					
0x008 to 0x00F	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x010	**PWR_VOSCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VOS[1:0]		Res.	Res.	Res.	Res.
	Reset value																											0	0				
0x014	**PWR_VOSSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ACTVOS[1:0]		ACTVOSRDY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VOSRDY	Res.	Res.	Res.
	Reset value																	0	0	1										1			
0x018 to 0x01F	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x020	**PWR_BDCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VBRS	VBE	Res.	Res.	Res.	Res.	Res.	Res.	MONEN	BREN
	Reset value																							0	0							0	0
0x024	**PWR_DBPCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBP
	Reset value																																0
0x028	**PWR_BDSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TEMPH	TEMPL	VBATH	VBATL	Res.	Res.	Res.	BRDY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value									0	0	0	0				0																
0x02C	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x030	**PWR_SCCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LDOEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BYPASS
	Reset value																								1								0
0x034	**PWR_VMCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ALS		AVDEN	Res.	Res.	Res.	Res.	PLS			PVDE
	Reset value																						0	0	0					0	0	0	0

表 62. PWR 寄存器映射和复位值（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x03C	**PWR_VMSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PVDD	Res.	VDDIO2RDY	AVDD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value										0		0	0																			
0x040	**PWR_WUSCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CWUF5	CWUF4	CWUF3	CWUF2	CWUF1
	Reset value																												0	0	0	0	0
0x044	**PWR_WUSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUF5	WUF4	WUF3	WUF2	WUF1
	Reset value																												0	0	0	0	0
0x048	**PWR_WUCR**	Res.	Res.	Res.	Res.	Res.	Res.	WUPPUPD5		WUPPUPD4		WUPPUPD3		WUPPUPD2		WUPPUPD1		Res.	Res.	Res.	WUPP5	WUPP4	WUPP3	WUPP2	WUPP1	Res.	Res.	Res.	WUPEN5	WUPEN4	WUPEN3	WUPEN2	WUPEN1
	Reset value							0	0	0	0	0	0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x04C	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x050	**PWR_IORETR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JTAGIORETEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IORETEN
	Reset value																1																1
0x054 to 0x0FF	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x104	**PWR_PRIVCFGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NSPRIV	Res.
	Reset value																															0	

有关寄存器边界地址的信息，请参见*第 2.2 节*。

10 复位和时钟控制 (RCC)

10.1 简介

复位和时钟控制 (RCC) 用于管理不同类型的复位，并为总线和外设生成所有时钟。

10.2 RCC 引脚和内部信号

下表列出了连接到封装引脚或焊球的 RCC 输入与输出信号。

表 63. 连接到封装引脚或焊球的 RCC 输入/输出信号

信号名称	信号类型	说明
NRST	I/O	系统复位，可用于为外部器件提供复位信号
OSC32_IN	I	32 kHz 振荡器输入
OSC32_OUT	O	32 kHz 振荡器输出
OSC_IN	I	系统振荡器输入
OSC_OUT	O	系统振荡器输出
MCO1	O	外部器件的输出时钟 1
MCO2	O	外部器件的输出时钟 2
LSCO	O	外部器件的低速输出时钟
AUDIOCLK	I	I2S1、I2S2 和 I2S3 的外部内核时钟输入

10.3 RCC 复位功能说明

有三种类型的复位：

- 系统复位
- 电源复位
- 备份域复位

10.3.1 电源复位

只要发生以下事件之一，就会产生电源复位：

- 欠压复位 (BOR)
- 退出待机模式时

欠压复位，包括上电或掉电复位 (POR/PDR)，将所有寄存器设置为其复位值，但备份域中的寄存器除外。

退出待机模式时，内核域中的所有寄存器都设置为其复位值。内核域外的寄存器不受影响。这些寄存器包括：RTC、WKUP、IWDG 和 GPIO 上拉/下拉配置（处于待机模式和从待机模式退出期间）。

10.3.2 系统复位

除了 *RCC 复位状态寄存器 (RCC_RSR)* 中的复位标志和备份域中的寄存器外，系统复位会将其他全部寄存器都复位为复位值。

只要发生以下事件之一，就会产生系统复位：

- NRST 引脚低电平（外部复位）
- 窗口看门狗事件（WWDG 复位）
- 独立看门狗事件（IWDG 复位）
- 软件复位（SW 复位）（请参见*软件复位*）
- 低功耗模式安全复位（请参见*低功耗模式安全复位*）
- 欠压复位

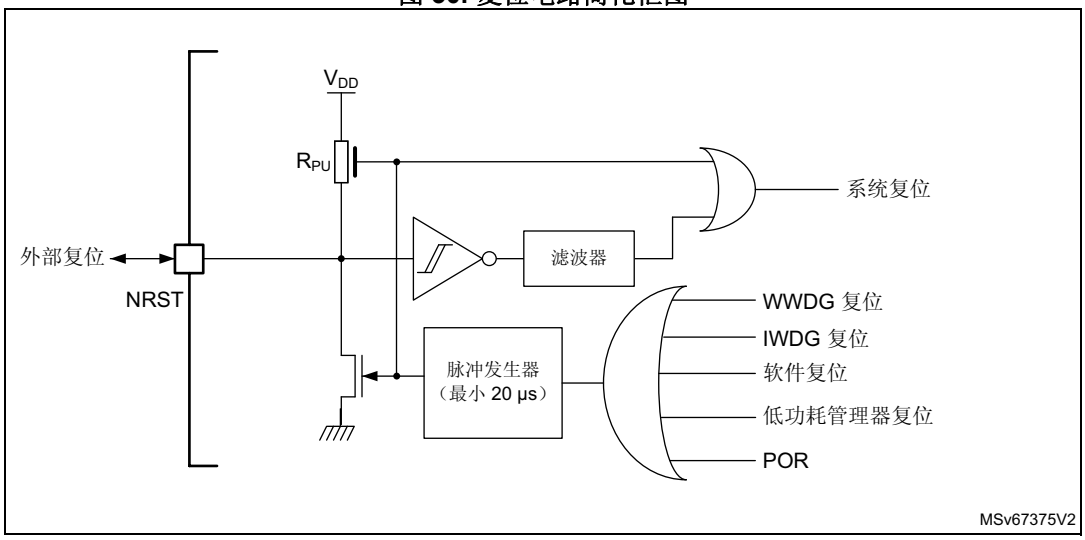
可通过查看 *RCC 复位状态寄存器 (RCC_RSR)* 中的复位标志确定复位源。

这些源均作用于 NRST 引脚，该引脚在复位过程中始终保持低电平。复位服务程序向量根据产品状态和/或启动选项字节进行选择。

芯片内部的复位信号会向 NRST 引脚上输出一个低电平脉冲。脉冲发生器可确保每个内部复位源的复位脉冲都至少持续 20 µs。对于外部复位，在 NRST 引脚处于低电平时产生复位脉冲。

内部复位情况下，内部上拉 R_{PU} 失效，通过上拉电阻节约功耗。

图 30. 复位电路简化框图



软件复位

要对器件进行软件复位，必须将 Cortex-M33 应用中断和复位控制寄存器中的 SYSRESETREQ 位置 1。

低功耗模式安全复位

为了避免关键应用错误地进入低功耗模式，提供了以下低功耗模式安全复位。如果在选项字节中使能，则在下列任意情况下会产生这种复位：

- 进入待机模式：此复位的使能方式是清零用户选项字节中的 nRST_STDBY 位。使能后，只要成功执行进入待机模式序列，器件就将复位，而非进入待机模式。
- 进入停止模式：此复位的使能方式是清零用户选项字节中的 nRST_STOP 位。使能后，只要成功执行进入停止模式序列，器件就将复位，而非进入停止模式。

有关用户选项字节的详细信息，请参见*第 7.4.1 节：关于选项字节*。

10.3.3 备份域复位

只要发生以下事件之一，就会产生备份域复位：

- 软件复位，通过将 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 VSWRST 位置 1 触发。在将 VSWRST 位置 1 以执行复位之前，必须使能对备份域的写访问。
- 在电源 V_{DD} 和 V_{BAT} 都已掉电后，其中任何一个又再上电。

在以下这些情况下，备份 RAM 会被擦除：

- 在备份域复位后
- 在产品状态完全回退后（请参见*第 7 章：嵌入式 Flash (FLASH)*）
- 发生入侵事件时

更多信息，请参见*第 9.6.5 节：VDDIO2 电压监视器 (IO2VM)*。

备份域复位会影响 LSE 振荡器、RTC、备份寄存器、备份 SRAM 和 RCC_BDCR 寄存器。

10.3.4 复位源标识

应用程序可通过检查 RCC_RSR 寄存器中的复位标志来识别复位源。

软件可通过将 RMVF 位置 1 来复位标志。

下表给出了不同复位源条件下 RCC_RSR 寄存器状态位的值。例如，发生 IWDG 超时时（第 6 行），如果 CPU 正在启动阶段读取 RCC_RSR 寄存器，则 PINRSTF 和 IWDGRSTF 位都会置 1，指示 IWDG 也生成了一个引脚复位信号。

表 64. 复位源标识 (RCC_RSR)(1)

#	生成复位的条件	LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	BORRSTF	PINRSTF
1	上电复位	0	0	0	0	1	1
2	引脚/焊盘复位	0	0	0	0	0	1
3	欠压（低电平或高电平）复位	0	0	0	0	1	1
4	CPU 生成的系统复位	0	0	0	1	0	1

表 64. 复位源标识 (RCC_RSR)(1)（续）

#	生成复位的条件	LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	BORRSTF	PINRSTF
5	WWDG 复位	0	1	0	0	0	1
6	IWDG 复位	0	0	1	0	0	1
7	非法进入停止模式复位	1	0	0	0	0	1

1. 灰色单元格突出显示了置 1 的寄存器位。

10.4 RCC 时钟功能说明

可以使用四种不同的时钟源来驱动系统时钟 (SYSCLK)：

- HSI：高速内部 64 MHz RC 振荡器时钟
- CSI：低功耗内部 4 MHz RC 振荡器时钟
- HSE：高速外部晶振或时钟，从 4 MHz 到 50 MHz
- PLL1 时钟

从复位中启动后，HSI 用作系统时钟源，配置为 32 MHz。

器件具有以下附加时钟源：

- LSI：32 kHz 低速内部 RC，该 RC 用于驱动独立看门狗，也可选择提供给 RTC 用于停止/待机模式下的自动唤醒
- LSE：32.768 kHz 低速外部晶振或时钟，可选择用于驱动实时时钟 (rtc_ck)
- HSI48：内部 48 MHz RC，可用于驱动 USB 和 RNG
- PLL2 时钟

对于每个时钟源来说，在未使用时都可单独开启或者关闭，以降低功耗。

可通过多个预分频器配置 AHB 频率、APB1 和 APB2 域。AHB 和 APB 域的最大频率为 250 MHz。

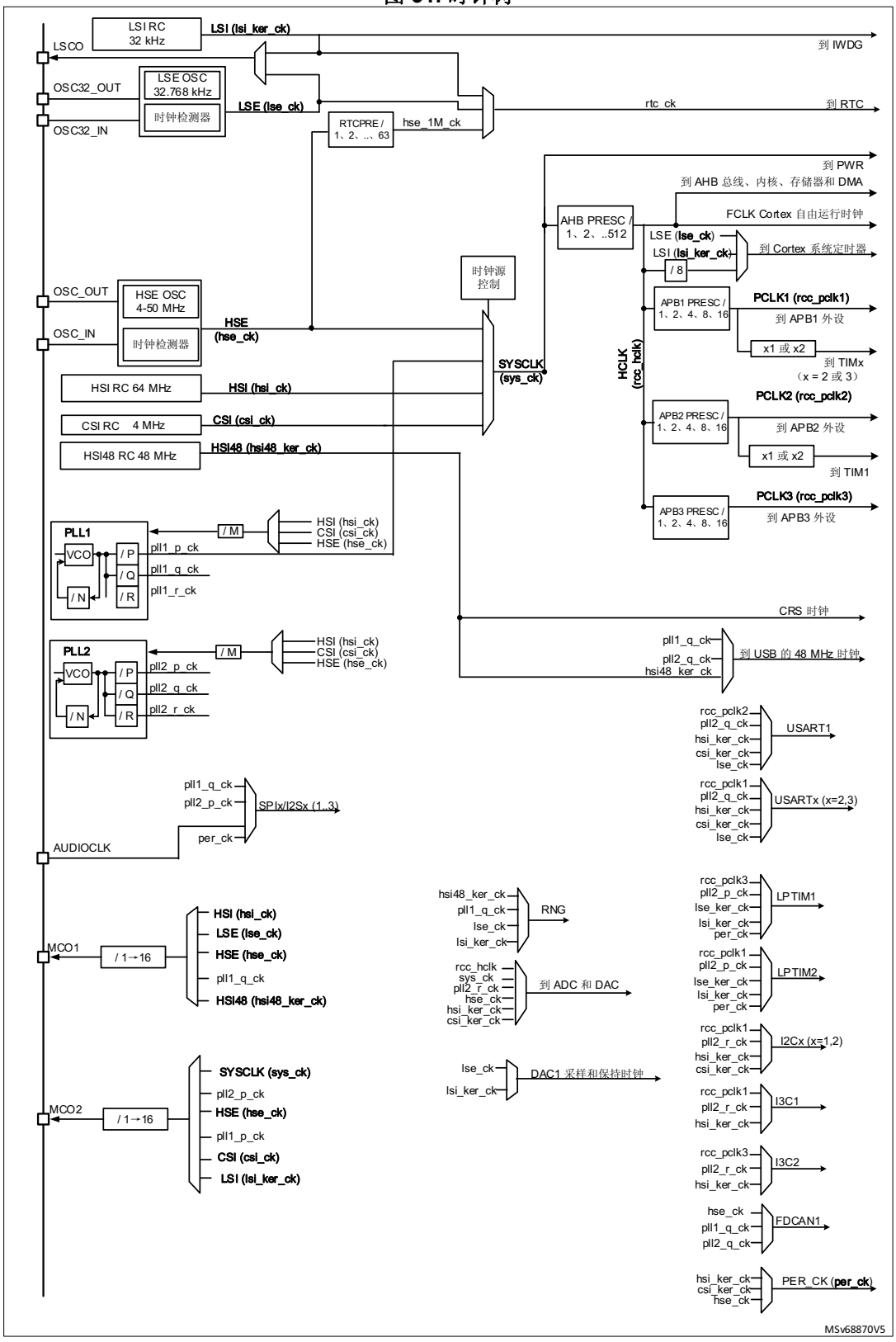
所有外设时钟均由其总线时钟（HCLK、PCLK1、PCLK2 或 PCLK3）提供，但以下接收独立内核时钟的外设时钟除外。该内核时钟可以通过软件在 RCC_CCIPRx 寄存器 (x = 1, 2, 3, 4, 5) 中选择以下几种时钟源：USB、RNG、ADC1、DAC1、U(S)ARTx（x = 1 到 3）、LPUART1、I2Cx（x = 1 和 2）、I3Cx（x = 1 和 2）、SPIx（x = 1 到 3）、FDCAN1、LPTIMx（x = 1 和 2）。

此外，RTC 内核时钟通过软件在 RCC_BDCR 中进行选择。IWDG 时钟始终为 LSI 32 kHz 时钟。

RCC 向 Cortex 系统定时器 (SysTick) 外部时钟馈送八分频的 AHB 时钟 (HCLK)，或馈送 LSE 或 LSI。SysTick 可使用此时钟作为时钟源，也可直接使用 Cortex 时钟 (HCLK) 作为时钟源，具体可在 SysTick 控制和状态寄存器中配置。

FCLK 充当 Cortex-M33 的自由运行时钟。

图 31. 时钟树



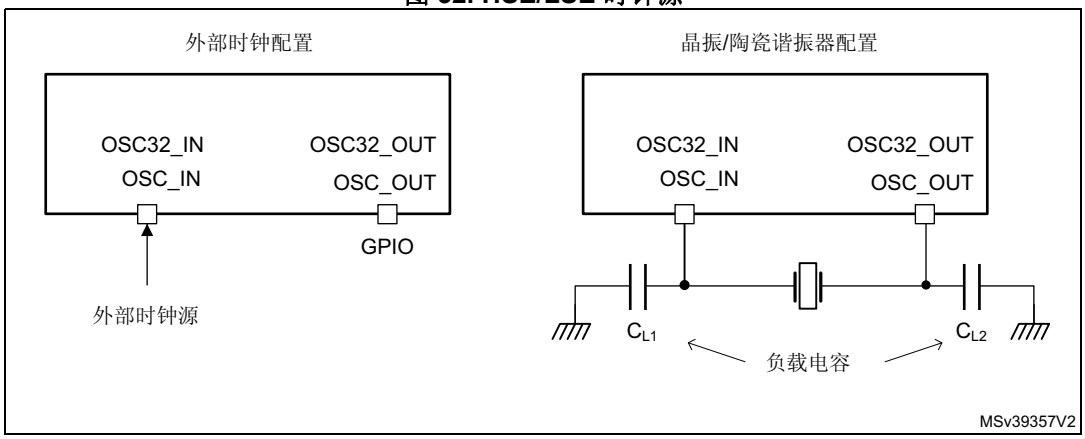
1. 有关内部/外部时钟源的详细信息，请参见数据手册的电气特性部分。

10.4.1 HSE 时钟

HSE 模块可通过两种可能的时钟源来生成时钟：

- 外部晶振/陶瓷谐振器
- 外部时钟源

图 32. HSE/LSE 时钟源



外部时钟源（HSE 旁路）

在此模式下，必须为 OSC_IN 引脚提供外部时钟源。外部时钟可以是低摆幅（模拟）时钟或数字时钟。

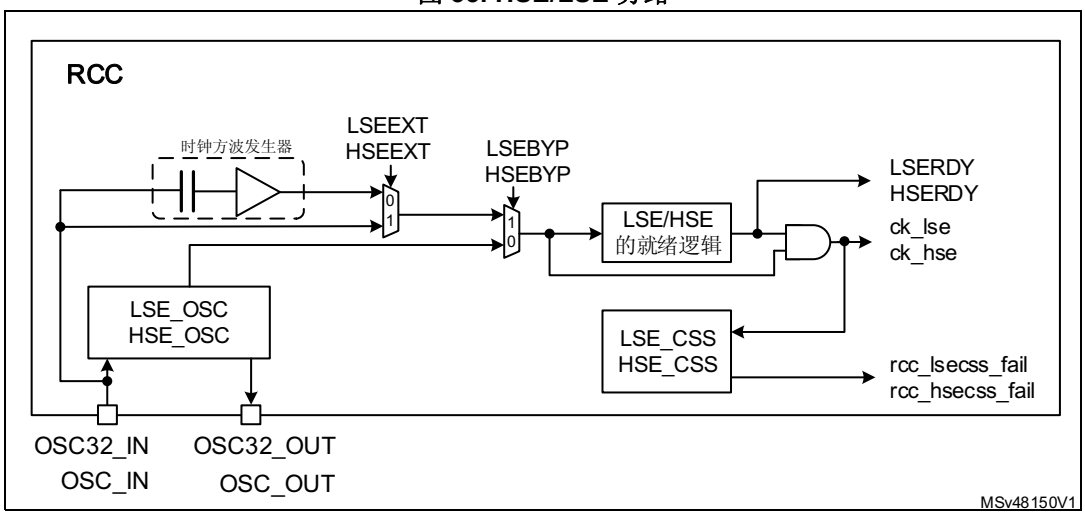
如果该时钟直接由外设使用，则占空比要求由改外设与应用定义（更多详细信息，请参见数据手册）。

该外部时钟提供给 OSC_IN 引脚，而 OSC_OUT 引脚可用作 GPIO（请参见*图 32*）。

如果使用模拟时钟（低摆幅），则必须将 *RCC 时钟控制寄存器 (RCC_CR)* 中的 HSEBYP 和 HSEON 位置 1。

如果使用数字时钟，则必须将 *RCC 时钟控制寄存器 (RCC_CR)* 中的 HSEBYP 和 HSEEXT 位置 1，然后再将 HSEON 位置 1。

图 33. HSE/LSE 旁路



外部晶振/陶瓷谐振器

通过将 HSEBYP 位设为 0 并将 HSEON 位置 1 来使能该振荡器。

如果产品需要非常精确的高速时钟，可使用 HSE。

相关硬件配置如*图 32* 所示：谐振器和负载电容必须尽可能地靠近振荡器的引脚，以尽量减小输出失真和起振稳定时间。负载电容值必须根据所选晶振或陶瓷谐振器的不同做适当调整。有关详细信息，请参见数据手册的电气特性部分。

RCC 时钟控制寄存器 (RCC_CR) 的 HSERDY 标志指示 HSE 振荡器是否稳定。在启动时，硬件将此位置 1 后，**hse_ck** 时钟才可以使用。如在 *RCC 时钟源中断使能寄存器 (RCC_CIER)* 中使能中断，则可产生中断。

HSE 可通过 HSEON 位开启和关闭。请注意，如果满足以下两个条件之一，则不能关闭 HSE：

- HSE 直接（通过软件复用）用作系统时钟。
- 在使能并选择 PLL1 来提供系统时钟的情况下，将 HSE 选作 PLL1 的参考时钟（通过软件复用）。

此时，硬件不允许将 HSEON 位设为 0。

系统进入停止或待机模式时，HSE 会自动由硬件禁止。

此外，HSE 时钟可被驱动到 MCO1 和 MCO2 输出，以及用作其他应用组件的时钟源。

10.4.2 HSI 时钟

HSI 模块为产品提供默认时钟。

HSI 是高速内部 RC 振荡器，可直接用作系统时钟、外设时钟或 PLL 输入。应用可使用预分频器来选择 8 MHz、16 MHz、32 MHz 或 64 MHz 的 HSI 输出频率。此预分频器受 HSIDIV 控制。

HSI 的优势如下：

- 无需外部晶振，因此时钟源成本较低
- 与 HSE 相比，启动时间更短（几微秒）

即使经过频率校准后，HSI 频率也不如外部晶振或陶瓷谐振器的频率精度高。

HSI 可通过 HSION 位开启和关闭。请注意，如果满足以下两个条件之一，则不能关闭 HSI：

- HSI 直接（通过软件复用）用作系统时钟
- 在使能并选择 PLL1 来提供系统时钟的情况下，将 HSI 选作 PLL1 的参考时钟（通过软件复用）。

此时，硬件不允许将 HSION 位设为 0。请注意，如果将 HSI 选作至少一个已使能 PLL（PLLxON 位置 1）的参考时钟，则不能更改 HSIDIV。此时，硬件不使用新值更新 HSIDIV。不过，如果将 HSI 直接用作系统时钟，则可更改 HSIDIV。

HSIRDY 标志指示 HSI 是否稳定。在启动时，硬件将此位置 1 后，HSI 输出时钟才可以使用。

HSI 时钟还可作为备份源（辅助时钟）使用，以防 HSE 发生故障（请参见*第 10.4.10 节：时钟安全系统 (CSS)*）。系统进入停止模式时，是否禁止 HSI 均可。

此外，HSI 时钟可被驱动到 MCO1 输出，以及用作其他应用组件的时钟源。

将 HSI 用作通信外设的内核时钟时必须谨慎，应用必须考虑以下参数：

- 从外设生成内核时钟请求到时钟实际可用的时间间隔
- 频率精度

注意： *系统处于停止模式时，HSI 可保持使能状态。*

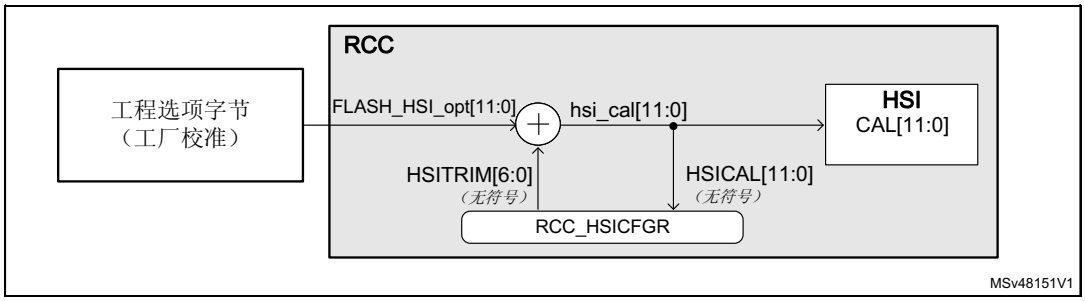
HSI 校准

因为生产工艺不同，不同芯片的 RC 振荡器频率也不同。因此意法半导体会对每个器件进行出厂校准，以满足 ACCHSI 精度要求（更多相关信息，请参见产品数据手册）。

上电复位后，工厂校准值将加载到 HSICAL[11:0] 位中。如果应用受到电压或温度变化影响，则这可能也会影响到 RC 振荡器的频率。用户应用可使用 HSITRIM[6:0] 位对 HSI 频率进行微调。

注意： *HSICAL[11:0] 和 HSITRIM[6:0] 位在 RCC HSI 校准寄存器 (RCC_HSICFGR) 中。*

图 34. HSI 校准流程



10.4.3 CSI 振荡器

CSI 为低功耗 RC 振荡器，可直接用作系统时钟、外设时钟或 PLL 输入。

CSI 的优势如下：

- 无需外部晶振，因此时钟源成本较低
- 与 HSE 相比，启动时间更短（几微秒）
- 极低功耗

CSI 提供的时钟频率为 4 MHz 左右，而 HSI 能提供高达 64 MHz 的时钟。

即使经过频率校准后，CSI 频率也不如外部晶振或陶瓷谐振器的频率精度高。

CSI 可通过 CSION 位开启和关闭。CSIRDY 标志指示 CSI 是否稳定。在启动时，硬件将此位置 1 后，CSI 输出时钟才可以使用。

如果满足以下两个条件之一，则不能关闭 CSI：

- CSI 直接（通过软件复用）用作系统时钟。
- 在使能并选择 PLL1 来提供系统时钟的情况下，将 CSI 选作 PLL1 的参考时钟（通过软件复用）。

此时，硬件不允许将 CSION 位设为 0。

系统进入停止模式时，是否禁止 CSI 均可。

此外，CSI 时钟可被驱动到 MCO2 输出，以及用作其他应用组件的时钟源。

虽然 CSI 的稳定时间要比 HSI 的稳定时间短，但是将 CSI 用作通信外设的内核时钟时必须谨慎，应用必须考虑以下参数：

- 从外设生成内核时钟请求到时钟实际可用的时间间隔
- 频率精度

注意： *CSION 和 CSIRDY 位在 RCC 时钟控制寄存器 (RCC_CR) 中。*

CSI 校准

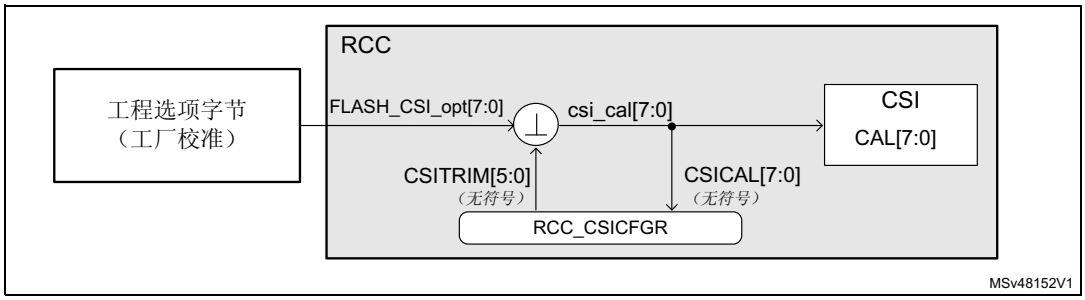
因为生产工艺不同，不同芯片的 RC 振荡器频率也不同，因此意法半导体会对每个器件进行出厂校准，以满足 ACC_{CSI} 精度要求（更多相关信息，请参见产品数据手册）。

复位后，工厂校准值将加载到 CSICAL[7:0] 位中。

如果应用受到电压或温度变化影响，则这可能也会影响到 RC 振荡器的频率。用户应用可使用 CSITRIM[5:0] 位对 CSI 频率进行微调。

注意： *CSICAL[7:0] 和 CSITRIM[5:0] 位在 RCC CSI 校准寄存器 (RCC_CSICFGR) 中。*

图 35. CSI 校准流程



10.4.4 HSI48 时钟

HSI48 是一款 RC 振荡器，其提供的 48 MHz 时钟可直接用作某些外设（例如 USB 或 RNG）的内核时钟。

HSI48 振荡器主要用于通过特殊时钟恢复系统 (CRS) 电路为 USB 外设提供高精度时钟，该电路可使用 USB SOF 信号、LSE 或外部信号以精细粒度实时自动调整振荡器频率。

系统进入停止或待机模式后，HSI48 振荡器会立即被禁止。如果未使用 CRS，则该振荡器将自由运行，并因此受到生产工艺差异的影响。因此意法半导体会对每个器件进行出厂校准，以满足 ACCHSI48 精度要求（更多相关信息，请参见产品数据手册）。

有关如何配置和使用 CRS 的更多详细信息，请参见*第 11 章：时钟恢复系统 (CRS)*。

HSI48RDY 标志指示 HSI48 振荡器是否稳定。在启动时，硬件将此位置 1 后，HSI48 输出时钟才可以使用。

HSI48 可通过 HSI48ON 位开启和关闭。

HSI48 时钟还可被驱动到 MCO1 复用器，以及用作其他应用组件的时钟源。

注意： *HSI48ON 和 HSI48RDY 位在 RCC 时钟控制寄存器 (RCC_CR) 中。*

10.4.5 PLL 说明

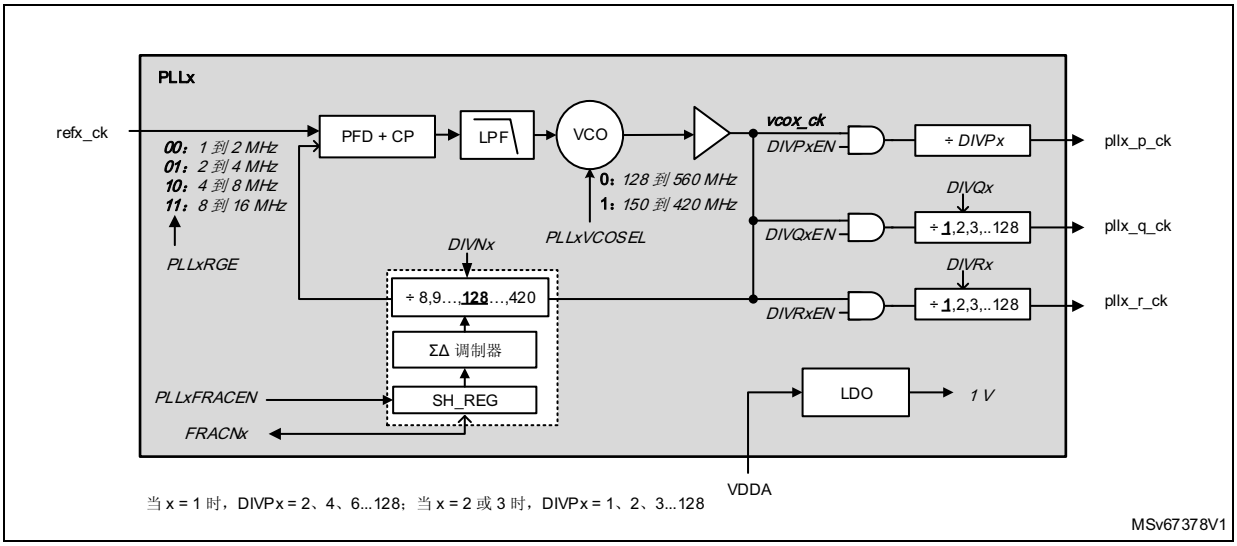
RCC 具有两个 PLL：

- 一个主 PLL (PLL1)，通常用于为 CPU 和某些外设提供时钟
- 一个专用 PLL (PLL2)，用于为外设生成内核时钟

集成在 RCC 中的 PLL 完全独立。它们具有以下特性：

- VCO 支持两种模式：
  - 宽范围模式
  - 低范围模式，适用于音频用例等
- 输入频率范围：
  - 在宽范围模式下，VCO 的输入频率范围为 2 MHz 到 16 MHz
  - 在低范围模式下，VCO 的输入频率范围为 1 MHz 到 2 MHz
- 能够工作在整数或小数模式下
- 13 位 Σ−Δ 调制器，能够对 VCO 频率进行 11 档微调（误差最高 0.3 ppm）。
- Σ−Δ 调制器可实时更新，不会在 PLL 输出端产生频率过冲。
- 每个 PLL 均可通过后分频器提供三种输出。

图 36. PLL 框图



PLL 通过 RCC_PLLxDIVR、RCC_PLLxFRACR、RCC_PLLxCFGR 和 RCC_CR 寄存器进行控制。

为 PLL 提供的参考时钟的频率 (**refx_ck**) 必须介于 1 MHz 到 16 MHz 范围内。为满足此条件，必须正确编程 *RCC PLL 时钟源选择寄存器 (RCC_PLL1CFGR)* 的 DIVMx 分频器。此外，必须根据参考输入频率设置 *RCC_PLLxCFGR* 寄存器的 PLLxRGE[1:0] 位域，以确保 PLL 达到最佳性能。

用户应用随后可配置合适的 VCO。当参考时钟的频率低于 2 MHz 时，必须选择较小的范围 (150 - 420 MHz)。

要降低功耗，建议将 VCO 输出配置为较小的范围。

必须对 DIVNx 环分频器进行编程，以使 VCO 输出达到预期频率。此外，必须满足 VCO 输出范围。

将 FRACNx 影子寄存器的 SH_REG 位的值设为 0 时，PLL 工作在整数模式下。PLLxFRACEN 位从 0 变为 1 时，用 FRACNx 值更新 SH_REG 位。Σ−Δ 调制器采用专门设计，可最大程度地降低抖动影响，同时实现非常小的频率阶跃。

可通过将 PLLxON 置 1 来使能 PLL。PLLxRDY 位指示 PLL 已就绪（表示已锁定）。

注意： *在使能 PLL 之前，请确保为 PLL 提供的参考频率 (**refx_ck**) 稳定，以使硬件在 PLLx 开启时不允许更改 DIVMx，以及在其中的一个 PLL 开启时无法更改 PLLSRC。*

如果当前使用 PLL1 提供系统时钟，则硬件可防止向 PLL1ON 写入 0。此外，时钟发生器还具备其他硬件保护（请参见第 10.4.1 节：HSE 时钟、第 10.4.2 节：HSI 时钟和第 10.4.3 节：CSI 振荡器）。

一旦使能了 PLL，以下 PLL 参数就无法更改：DIVNx、PLLxRGE、PLLxVCOSEL、DIVPx、DIVQx、DIVRx、DIVPxEN、DIVQxEN 和 DIVRxEN。

为确保 PLL 达到最佳性能，如果未使用其中的一个后分频器（DIVP、DIVQ 或 DIVR），则应用程序必须将使能位 (DIVyEN) 置 1，并将相应的后分频器位（DIVP、DIVQ 或 DIVR）设为 0。

如果未遵循上述规则，则无法保证 PLL 输出频率。

输出频率计算

将 PLL 配置为整数模式 (SH_REG = 0) 时，VCO 频率 (F_{VCO}) 通过以下表达式进行计算：

$$F_{VCO} = F_{REF_CK} \times DIVN$$

$$F_{PLL_y_CK} = (F_{VCO} / (DIVy + 1))，其中\ y = P、Q\ 或\ R$$

将 PLL 配置为小数模式（SH_REG 不为 0）时，在使能 PLL 之前必须先初始化 DIVN 分频器。不过，可实时更改 FRACNx 值，而不会干扰 PLL 输出。

可使用该功能基于精度较高的晶振值生成特定频率或实时微调频率。

对于每个 PLL，VCO 频率均通过以下公式进行计算：

$$F_{VCO} = F_{ref_ck} \times \left(DIVN + \frac{FRACN}{2^{(13)}} \right)$$

注意： *对于 PLL1，DIVP 只能使用奇数值。*

在以下情况下，PLL 将由硬件禁止：

- 系统进入停止或待机模式
- 将 HSE 或 PLL（由 HSE 提供时钟）用作系统时钟时发生 HSE 故障

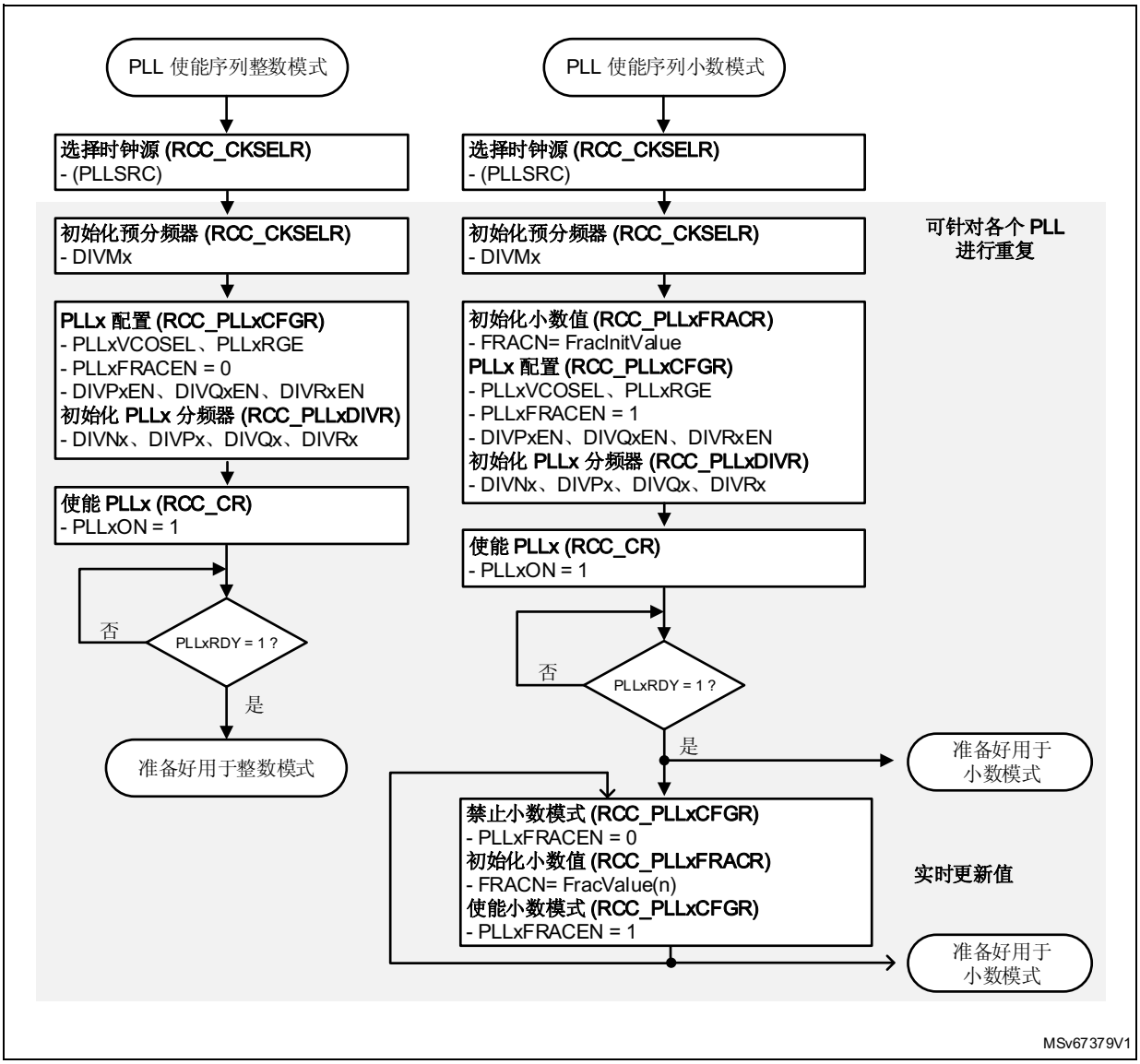
PLL 初始化阶段

图 37 给出了整数和小数模式下推荐的 PLL 初始化序列。在初始化序列开始时，PLLx 应处于禁止状态：

1. 根据所需频率初始化 PLL 寄存器。
   - 对于整数模式，将 *RCC_PLLxCFGR 寄存器*的 PLLxFRACEN 设为 0。
   - 对于小数模式，将 FRACN 设为所需初始值 (FracInitValue)，然后将 PLLxFRACEN 置 1。
2. 将 PLLxON 位置 1 后，用户应用程序必须等待到 PLLxRDY 位置 1。如果 PLLx 处于小数模式，则只要 PLLxRDY = 0，PLLxFRACEN 位就不能设回 0。
3. 将 PLLxRDY 位置 1 后，PLLx 即准备好进行使用。
4. 如果应用程序要实时调整 PLLx 频率（仅限小数模式），则：
   a) PLLxFRACEN 必须设为 0。如果 PLLxFRACEN = 0，则 Σ−Δ 调制器仍使用锁存到 SH_REG 中的值。
   b) 新值必须上传到 PLLxFRACR (FracValue(n)) 中。
   c) PLLxFRACEN 必须置 1，以将 PLLxFRACR 的内容锁存到其影子寄存器中。

注意： *PLLxRDY 变为 1 时即表示，PLLx 输出频率和目标值之差小于 ±2%。*

图 37. PLL 初始化流程图



10.4.6 LSE 时钟

LSE 模块可通过两种可能的时钟源来生成时钟：

- 外部晶振/陶瓷谐振器
- 外部用户时钟

外部时钟源（LSE 旁路）

在此模式下，必须为 OSC32_IN 引脚提供外部时钟源。输入时钟可以是低摆幅（模拟）时钟或数字时钟，最高频率不超过 1 MHz。建议使用接近 50% 的占空比。

此外部时钟提供给 OSC32_IN 引脚，而 OSC32_OUT 引脚必须保持高阻态（请参见*图 32*）。

如果使用模拟时钟（低摆幅），则必须将 LSEBYP 和 LSEON 位置 1 (*RCC 备份域控制寄存器 (RCC_BDCR)*)。

如果使用数字时钟，则必须将 LSEBYP 和 LSEEXT 位置 1，然后再将 LSEON 位置 1 (*RCC 备份域控制寄存器 (RCC_BDCR)*)，如果使用 RTC，则 LSE 旁路不得配置为数字模式，而必须配置为低摆幅模拟模式（复位后的默认值）。

外部晶振/陶瓷谐振器（LSE 晶振）

LSE 时钟由 32.768 kHz 晶振或陶瓷谐振器生成。可作为实时时钟 (RTC) 的时钟源来提供时钟/日历或其他定时功能，具有功耗低且精度高的优点。

RCC 备份域控制寄存器 (RCC_BDCR) 的 LSERDY 标志指示 LSE 晶振是否稳定。在启动时，硬件将该位置 1 后，LSE 晶振输出时钟信号才可以使用。如在 *RCC 时钟源中断使能寄存器 (RCC_CIER)* 中使能中断，则可产生中断。

LSE 振荡器通过 LSEON 位开启和关闭。系统进入停止或待机模式时，LSE 保持使能状态。

此外，LSE 时钟可被驱动到 MCO1 输出，以及用作其他应用组件的时钟源。

LSE 还具有可编程驱动功能 (LSEDRV[1:0])，可用于调节放大器驱动能力。此驱动能力是根据外部晶振/陶瓷元件的要求来选择的，以确保稳定的振荡频率。

必须在使能 LSE 振荡器之前设置驱动能力。

10.4.7 LSI 时钟

系统处于停止或待机模式时，LSI 可作为低功耗时钟源保持运行，供独立看门狗 (IWDG) 和自动唤醒单元 (AWU) 使用。

时钟频率在 32 kHz 左右。有关详细信息，请参见数据手册的电气特性部分。

LSI 可通过 LSION 位开启和关闭。LSIRDY 标志指示 LSI 振荡器是否稳定。如果独立看门狗已通过硬件或软件启动，则 LSI 将强制开启且不可禁止。

系统进入停止或待机模式时，LSI 保持使能状态。

LSI 启动时，硬件将 LSIRDY 位置 1 后，才会提供相应时钟。如在 *RCC 时钟源中断使能寄存器 (RCC_CIER)* 中使能中断，则可产生中断。

此外，LSI 时钟可被驱动到 MCO2 输出，以及用作其他应用组件的时钟源。

注意： *LSION 和 LSIRDY 位在 RCC 备份域控制寄存器 (RCC_BDCR) 中。*

10.4.8 系统时钟 (SYSCLK) 选择

可以使用四种不同的时钟源来驱动系统时钟 (SYSCLK)：

- HSI 振荡器
- CSI 振荡器
- HSE 振荡器
- PLL

系统时钟最大频率为 250 MHz。系统复位后（或在退出待机模式后），选择 HSI 振荡器（32 MHz）作为系统时钟。在直接使用或者通过 PLL 使用某一时钟源来作为系统时钟时，该时钟源无法停止。

只有在目标时钟源已就绪时（时钟在启动延迟或 PLL 锁相后稳定时），才可从一个时钟源切换到另一个。如果选择尚未就绪的时钟源，则切换在该时钟源就绪时才会进行。*RCC 时钟控制寄存器 (RCC_CR)* 中的状态位指示哪些时钟已就绪，以及当前哪个时钟正充当系统时钟。

10.4.9 在停止和待机模式下处理时钟发生器

当整个系统进入停止模式时，所有时钟（系统时钟和内核时钟）均停止，以下时钟源也会停止：

- CSI、HSI（取决于 HSIKERON 和 CSIKERON 位）
- HSE
- PLL1 和 PLL2
- HSI48

除设为 0 的 PLL1ON、PLL2ON、HSEON 和 HSI48ON 外，RCC 寄存器的内容未发生更改。

退出停止模式

当系统通过唤醒事件退出停止模式时，应用程序可选择用于重启的振荡器（HSI 和/或 CSI）。STOPWUCK 位将振荡器选作系统时钟。STOPKERWUCK 位选择用作外设内核时钟的振荡器。如果在系统停止后，外设需要由非系统时钟振荡器来生成内核时钟，则 STOPKERWUCK 位十分有用。

所有这些位均属于*第 10.8.5 节：RCC 时钟配置寄存器 1 (RCC_CFGR1)*。

表 65 给出了相应行为的详细说明。

表 65. STOPWUCK 和 STOPKERWUCK 说明

<table>
<tr><th rowspan="2">STOPWUCK</th><th rowspan="2">STOPKERWUCK</th><th rowspan="2">-</th><th rowspan="2">系统退出停止模式时激活的振荡器</th><th colspan="2">系统退出
停止模式时分配的时钟</th></tr>
<tr><th>系统时钟</th><th>内核时钟</th></tr>
<tr><td rowspan="2">0</td><td>0</td><td>→</td><td>HSI</td><td rowspan="2">HSI</td><td>HSI</td></tr>
<tr><td>1</td><td>→</td><td rowspan="2">HSI 和 CSI</td><td rowspan="2">HSI 和/或 CSI</td></tr>
<tr><td rowspan="2">1</td><td>0</td><td>→</td><td rowspan="2">CSI</td></tr>
<tr><td>1</td><td>→</td><td>CSI</td><td>CSI</td></tr>
</table>

停止模式期间

系统处于停止模式时，在以下两种特定情况下可使能 HSI 或 CSI。

- 专用外设请求内核时钟时：
  外设接收到 HSI 或 CSI，具体取决于为此外设选择的内核时钟源（通过 CKPERSEL[1:0]）。
- HSIKERON 或 CSIKERON 位置 1 时：
  HSI 和 CSI 在停止模式期间保持运行，但会对输出进行门控。在系统退出停止模式时或在外设请求内核时钟时，时钟将立即可用（有关详细信息，请参见*表 63*）。

HSIKERON 和 CSIKERON 位属于 RCC 源控制寄存器 (RCC_CR)。

表 66 给出了相应行为的详细说明。

表 66. HSIKERON 和 CSIKERON 行为

HSIKERON (CSIKERON)	-	停止模式期间的 HSI (CSI) 状态	HSI (CSI) 状态稳定时间
0	→	关闭	$t_{su(HSI)}$ $t_{su(CSI)}$ (1)
1	→	正在运行且受门控	立即

1. $t_{su(HSI)}$ 和 $t_{su(CSI)}$ 为 HSI 和 CSI 振荡器的启动时间（有关这些参数值的信息，请参见产品数据手册）。

当微控制器使系统退出待机模式时，将 HSI 选作系统和内核时钟。RCC 寄存器均复位为其初始值，但 RCC_RSR 和

RCC_BDCR 寄存器除外。

注意： *HSI 和 CSI 输出提供两条时钟路径：*

- 一条路径用于系统时钟（hsi_ck 或 csi_ck）
- 一条路径用于外设内核时钟（hsi_ker_ck 或 csi_ker_ck）。

如果外设在系统处于停止模式时请求内核时钟，则只会激活提供 hsi_ker_ck 或 csi_ker_ck 的路径。

10.4.10 时钟安全系统 (CSS)

HSE 上的时钟安全系统

时钟安全系统可由软件通过 HSECSSON 位使能。即使将 HSEON 设为 0，也仍可使能 HSECSSON 位。

在 HSE 已使能并就绪且 HSECSSON 置 1 时，HSE 上的 CSS 将由硬件使能。

HSE 禁止时，将禁止 HSE 上的 CSS。因此当系统处于停止模式时，该功能不起作用。

不能由软件直接将 HSECSSON 位清零。

发生系统复位或系统进入待机模式时，由硬件将 HSECSSON 位清零。

如果检测到 HSE 时钟故障，则系统会根据 STOPWUCK 位配置（在 RCC 时钟配置寄存器 1 (RCC_CFGR1) 中）自动切换到 HSI 或 CSI，以提供安全时钟。HSE 随后将自动禁止，一个时钟故障事件将发送到高级控制定时器 (TIM1) 的刹车输入，并自动生成 NMI 来向应用程序通知此故障，从而允许 MCU 执行救援操作。如果在发生故障时已将 HSE 输出用作 PLL 的时钟源，则还会禁止 PLL。

如果在使能 CSS 时发生 HSE 时钟故障，则 CSS 会生成一个中断，这会导致自动生成 NMI。*RCC 时钟源中断标志寄存器 (RCC_CIFR)* 中的 HSECSSF 标志置 1，以允许应用程序识别故障源。NMI 程序将无限期执行，除非将 HSECSSF 位清零。因此，应用程序必须通过将 *RCC 时钟源中断清零寄存器 (RCC_CICR)* 中的 HSECSSC 位置 1 来清零 NMI ISR 中的 HSECSSF 标志。

LSE 上的时钟安全系统

可由软件通过写入 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 LSECSSON 位，来激活 LSE 上的时钟安全系统。只能通过硬件复位、RTC 软件复位或在 LSE 上检测到故障后禁止该位。LSE 使能（LSEON 使能）并就绪（LSERDY 由硬件置 1）后，以及通过 RTCSEL 选择 RTC 时钟后，必须写入 LSECSSON。

LSE 上的 CSS 可在所有模式（包括 VBAT 模式）下工作，并且在发生系统复位（上电复位除外）时也能保持工作状态。

LSE 上的时钟安全系统可以检测 LSE 何时消失或发生超频。此外，LSE 上的干扰可通过将 LSEGFON 置 1 来滤除。当 LSE 禁止（LSEON = 0 且 LSERDY = 0）时，必须写入 LSEGFON。

如果在外部 32 kHz 振荡器上检测到故障，则不会再向 RTC 提供 LSE 时钟，但不会对寄存器进行硬件操作。

LSE 上的 CSS 检测事件连接到 TAMP 外设的内部入侵 3。必须使能内部入侵 3（TAMP_CR1 寄存器中的 ITAMP3E = 1），并使能相关中断（TAMP_IER 中的 ITAMP3IE）以从低功耗模式唤醒。除非 TAMP_CR3 中的 ITAMP3NOER = 1，否则还会擦除 TAMP 备份寄存器和备份 SRAM（更多详细信息，请参见*第 33 章：入侵和备份寄存器 (TAMP)*）。

对于 LSE 上的 CSS 检测事件（RCC_BDCR 中的 LSECSSD = 1），软件随后必须禁止 LSECSSON 位并停止出现故障的 32 kHz 振荡器（禁止 LSEON），同时更改 RTC 时钟源（无时钟、LSI 或 HSE，通过 RTCSEL），或者采取任何必要措施来确保应用的安全。

有关 LSE 上的 CSS 电气特性，请参见数据手册。

10.4.11 时钟输出生成 (MCO1/MCO2)

此微控制器有两个时钟输出引脚 (MCO) 可供使用，分别为 MCO1 和 MCO2。可以为每个输出选择一个时钟源。所选时钟可通过可配置的预分频器进行分频（有关信号选择的更多信息，请参见*图 31*）。

MCO1 和 MCO2 输出通过 *RCC 时钟配置寄存器 1 (RCC_CFGR1)* 中的 MCO1PRE[3:0]、MCO1[2:0]、MCO2PRE[3:0] 和 MCO2[2:0] 进行控制。

GPIO 端口对应于各个 MCO 引脚，必须在复用功能模式下进行编程。

为 MCO 输出提供的时钟不能超出最大引脚速度（有关支持的引脚速度的信息，请参见产品数据手册）。

- **LSCO**

  另一个输出 (LSCO) 允许将以下低速时钟之一输出到外部 LSCO 引脚上：
  - LSI
  - LSE

  此输出在停止模式下仍然可用。此输出在待机和 VBAT 模式下不可用。此选择由 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 LSCOSEL 控制，并通过 LSCOEN 使能。

MCO 时钟输出需要在 MCO 引脚上选择相应的复用功能。LSCO 引脚必须保持默认的 POR 状态。

10.4.12 内核时钟选择

某些外设经过专门设计，支持两种可异步运行的不同时钟域：

- 与寄存器和总线接口同步的时钟域（**ckg_bus_perx** 时钟）
- 通常与外设同步的时钟域（内核时钟）

拥有支持这两种时钟域的外设具备以下优势：用户应用能更加自由地为 CPU、总线矩阵和外设的内核部分选择最佳时钟频率。

因此，用户应用无需对外设进行重新编程即可更改总线频率。例如，即使实时更改相应的 APB 时钟，也不会干扰正在通过 UART 进行的传输过程。

下表显示了 RCC 可为外设提供的内核时钟。下表的每一行都表示一个复用器以及与其输出相连的外设。

表 67. 内核时钟分配概述

外设	时钟复用器控制位	pll1_q_ck	pll2_p_ck	pll2_q_ck	pll2_r_ck	sys_ck	总线时钟(1)	hse_ck	hsi_ker_ck	csi_ker_ck	hsi48_ck	lse_ck	lsi_ck	per_ck(2)	AUDIOCLK	禁止
FDCAN1	FDCANSEL	1	-	2	-	-	-	0	-	-	-	-	-	-	-	-
I2C1	I2C1SEL	-	-	-	1	-	0	-	2	3	-	-	-	-	-	-
I2C2	I2C2SEL	-	-	-	1	-	0	-	2	3	-	-	-	-	-	-
I3C1	I3C1SEL	-	-	-	1	-	0	-	2	-	-	-	-	-	-	-
I3C2	I3C2SEL	-	-	-	1	-	0	-	2	-	-	-	-	-	-	-
LPTIM1	LPTIM1SEL	-	1	-	-	-	0	-	-	-	-	3	4	5	-	-
LPTIM2	LPTIM2SEL	-	1	-	-	-	0	-	-	-	-	3	4	5	-	-
TIM1	-	-	-	-	-	-	x	-	-	-	-	-	-	-	-	-
TIM2、TIM3 和 LPTIM2	TIMICSEL	-	-	-	-	-	-	-	1	1	-	-	-	-	-	0
RNG	RNGSEL	1	-	-	-	-	-	-	-	-	0	2	3	-	-	-
SPI(I2S)1	SPI1SEL	0	1	-	-	-	-	-	-	-	-	-	-	4	3	-
SPI(I2S)2	SPI2SEL	0	1	-	-	-	-	-	-	-	-	-	-	4	3	-
SPI(I2S)3	SPI3SEL	0	1	-	-	-	-	-	-	-	-	-	-	4	3	-
USARTx	USARTxSEL	-	-	1	-	-	0	-	3	4	-	5	-	-	-	-
LPUART 1	LPUART 1SEL	-	-	1	-	-	0	-	3	4	-	5	-	-	-	-
USB	USBSEL	1	-	2	-	-	-	-	-	-	3	-	-	-	-	0
ADCDAC(3)	ADCDACSEL	-	-	-	2	1	0	3	4	5	-	-	-	-	-	-
DAC1	DAC1SEL	-	-	-	-	-	-	-	-	-	-	0	1	-	-	-

表 67. 内核时钟分配概述（续）

外设	时钟复用器控制位	pll1_q_ck	pll2_p_ck	pll2_q_ck	pll2_r_ck	sys_ck	总线时钟(1)	hse_ck	hsi_ker_ck	csi_ker_ck	hsi48_ck	lse_ck	lsi_ck	per_ck(2)	AUDIOCLK	禁止
RTC/AWU	RTCSEL	-	-	-	-	-	-	3 (4)	-	-	-	1	2	-	-	0
全部	CKPERSEL	-	-	-	-	-	-	2	0	1	-	-	-	-	-	3

1. 总线时钟是与外设相连的总线接口时钟，可以是 APB 或 AHB 时钟。
2. per_ck 时钟可以是 hse_ck、hsi_ker_ck 或 csi_ker_ck，具体取决于 CKPERSEL 选择。
3. 占空比接近 50%，这意味着 DIV[P/Q/R]x 值应为偶数。对于 ADC，如果支持 DDR，则占空比应为 50%。
4. 时钟 HSE 由 RTCPRE 分频。

为减少开关数，某些外设共用同一内核时钟源。不过，所有外设均具有各自的专用使能信号。

专用于音频应用的外设

音频外设通常需要特别精确的频率，SPI(I2S) 的内核时钟可由以下时钟源生成：

- PLL1，适用于必须减小活动 PLL 数目的情况（对于 SPI/I2S1 到 I2S3）
- PLL2，用于在频率生成方面实现最佳灵活性
- HSE、HSI 或 CSI，适用于对电流消耗极为敏感的用例
- AUDIOCLK，适用于需要使用外部时钟参考的情况

专用于控制和数据传输的外设

SPI、I2C、I3C、UART 等外设无需特定内核时钟频率，但时钟频率应足以生成合适的波特率，或串行接口上所需的位时钟。为此，可从以下时钟源中选择：

- PLL1，适用于必须减小活动 PLL 数目的情况
- PLL2，适用于需要更高灵活性的情况。例如，该解决方案可通过 PLL1 更改频率总线，且不会影响某些串行接口的速度。
- HSI 或 CSI，适用于低功耗用例，或在外设必须快速从停止模式唤醒的情况（例如 UART、I2C 或 I3C）

注意： *如果无需高波特率，则 UART 还需要 LSE 时钟。*

RTC/AWU 时钟

rtc_ck 时钟源可以是以下源之一：

- **hse_1M_ck**（**hse_ck** 经过可编程预分频器分频获得）
- **lse_ck**
- **lsi_ck** 时钟

时钟源通过 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 RTCSEL[1:0] 位和 *RCC 时钟配置寄存器 1 (RCC_CFGR1)* 中的 RTCPRE[5:0] 位进行选择。

所做的选择只能通过复位备份域的方式修改。

如果选择 LSE 作为 RTC 时钟，则即使在备份或 V_{DD} 电源丢失时 RTC 仍可正常工作。

LSE 时钟位于备份域中，而其他振荡器则不是。因此：

- 如果选择 LSE 作为 RTC 时钟，则即使 V_{DD} 电源关断，RTC 也仍会继续工作（假设 V_{BAT} 电源保持工作）。
- 如果选择 LSI 作为 RTC 时钟，则在 V_{DD} 电源关断时将无法保证 AWU 状态。
- 如果将 HSE 时钟用作 RTC 时钟，则在 V_{DD} 电源关断或 V_{CORE} 电源关断时，无法保证 RTC 状态。

rtc_ck 时钟通过 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 RTCEN 位使能。

RTC 总线接口时钟（APB 时钟）通过 RCC_APB3ENR/LPENR 寄存器中的 RTCAPBEN 和 RTCAPBLPEN 位使能。

注意： *在 APB 时钟频率低于 RTC 时钟频率的七倍 ($F_{APB} < 7 \times F_{RTCLCK}$) 时读取 RTC 日历寄存器，软件必须读取日历时间寄存器和日期寄存器两次。如果对 RTC_TR 的第二次读取访问得到的结果与第一次相同，则表明数据正确无误。否则必须执行第三次读访问。*

看门狗时钟

RCC 为电路中的两个看门狗模块提供时钟。独立看门狗 (IWDG) 连接到 LSI。窗口看门狗 (WWDG) 连接到 APB 时钟。

如果独立看门狗已通过硬件选项或软件访问的方式启动，则 LSI 将强制开启且不可禁止。在 LSI 振荡器设置延迟之后，时钟将提供给 IWDG。

10.4.13 基于 TIMx 和 LPTIMx 的时钟测量

嵌入式定时器（输入捕获模式下的 TIM2/3 和 LPTIM1/2）可用于测量应用中使用的时钟源的频率。

10.4.14 内部振荡器校准

大多数时钟源发生器频率都可以通过对 TIMx 或 LPTIMx 的输入捕获进行测量。

- **使用 LSE 校准 HSI 或 CSI：**

  将 LSE 连接到 TIMx 或 LPTIMx 输入捕获的主要目的是为了精确测量 HSI 或 CSI。这需要直接或通过 PLL1 将 HSI 或 CSI 用作系统时钟源。借助 LSE 信号连续边沿之间的系统时钟计数数量，即可对内部时钟周期进行测量。利用 LSE 晶振的高精度（通常为几十 ppm），用户能以同一分辨率测定内部时钟频率，并可通过对时钟源进行微调来补偿由生产工艺造成的和/或与温度和电压相关的频率偏差。

  基本原理基于相对的测量（例如，HSI/LSE 比）。因此，精度与两个时钟源之比紧密相关。比率越大，测量结果越准确。

  HSI 和 CSI 振荡器均设有针对此目的的专用校准位，且支持用户访问（请参见 *RCC CSI 校准寄存器 (RCC_CSICFGR)*）。通过 PLLx 使用 HSI 或 CSI 时，还可通过 PLL 的小数分频器对系统时钟进行微调。

- **使用 HSI 校准 LSI：**

  同时也可测量 LSI 频率：这对于没有晶振的应用场合非常实用。超低功耗 LSI 振荡器具有较大的生产工艺偏差。可以使用更加精确的 HSI 时钟源来测量 LSI 时钟频率。通过此测量，可以实现更加精确的 RTC 时基超时（当使用 LSI 作为 RTC 时钟源时）和/或实现精度水平可以接受的 IWDG 超时。

10.4.15 RTC 和 TAMP 时钟

RTCCLK 时钟源由 RTC 和 TAMP 使用，可以是 HSE/32、LSE 或 LSI 时钟。选择方式是编程 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 RTCSEL[1:0] 位。所做的选择只能通过复位备份域的方式修改。配置系统时，必须始终使 PCLK 频率大于或等于 RTCCLK 频率，以便 RTC 正常工作。如果只使用备份寄存器，并且以边沿检测模式来进行入侵检测，则 TAMP 不需要任何内核时钟。所有其他入侵检测模式都需要内核时钟（更多详细信息，请参见*第 33 章：入侵和备份寄存器 (TAMP)*）。

LSE 和 LSI 时钟在备份域中，而 HSE 时钟不在。因此：

- 如果选择 LSE 或 LSI 作为 RTC 和 TAMP 时钟，则即使 V_{DD} 电源关断，这些外设也仍会继续工作（假设 V_{BAT} 电源保持工作）。
- 如果使用通过预分频器分频的 HSE 时钟作为 RTC 或 TAMP 时钟，则 V_{DD} 电源掉电或内部调压器关闭（切断内核域的供电）时，无法保证 RTC 状态。根据 TAMP 配置，如果在不需要任何内核时钟的模式下使用，则该时钟可以保持正常工作。

如果 RTC 和 TAMP 时钟为 LSE 或 LSI，则 RTC 在系统复位后仍可获得时钟并保持正常工作。

10.4.16 定时器时钟

定时器时钟频率由硬件自动定义为固定值。

分为两种情况：

- 如果 APB 预分频器等于 1，定时器时钟频率等于 APB 域频率。
- 否则，等于 APB 域频率的两倍 (×2)。

10.4.17 看门狗时钟

如果独立看门狗 (IWDG) 已通过硬件选项或软件访问的方式启动，则 LSI 振荡器将强制开启且不可禁止。在 LSI 振荡器稳定后，LSI 32 kHz 时钟将提供给 IWDG。

10.4.18 外设时钟门控

运行模式下的外设时钟门控

每个外设时钟均可由 RCC_AHBxENR 和 RCC_APBxENR 寄存器中的相应 EN 位使能。

如果外设时钟未激活，则不支持对外设寄存器进行读访问或写访问。

使能位的同步机制可为外设产生无干扰时钟。在使能位置 1 后，时钟将在外设总线时钟的 2 个周期之后激活。

小心： 使能外设时钟后，软件必须等待 2 个时钟周期才能访问外设寄存器。

睡眠模式下的外设时钟门控

如果外设已使能，则当器件处于睡眠模式时，可以通过将 RCC_AHBxLPENR 和 RCC_APBxLPENR 寄存器中的外设 LPEN 位清零来使外设时钟自动关闭。外设的 EN 和 LPEN 位都必须均置 1，才能使时钟在睡眠模式下保持开启状态。

10.4.19 总线时钟门控

要进一步降低运行或睡眠模式下的功耗，可以通过 RCC_CFGR2 寄存器中的 APBxDIS 和 AHBxDIS 位来禁止 APBx 或 AHBx 总线时钟，前提是没有使用任何映射到这些总线上的外设并且在 RCC_APBxENR 或 RCC_AHBxENR 中禁止了它们的时钟。

10.5 RCC 特权保护模式

默认情况下，复位后，除了 *RCC 特权配置寄存器 (RCC_PRIVCFGR)* 之外的全部 RCC 寄存器都可以通过特权访问和非特权访问进行读/写操作，而 RCC_PRIVCFGR 寄存器只能通过特权访问进行写操作。RCC_PRIVCFGR 可通过特权访问和非特权访问进行读操作。

RCC_PRIVCFGR 中的 PRIV 位只能通过特权访问进行写操作。此位用于配置所有 RCC 功能的特权访问（如*表 68*、特权外设的 GTZC 或特权感知外设本身所定义）。

当 RCC_PRIVCFGR 中的 PRIV 置 1 时：

- 只能通过特权访问对 RCC 特权位进行写操作。
- RCC 特权位只能通过特权访问进行读操作，但 RCC_PRIVCFGR 除外，该寄存器可通过特权访问或非特权访问进行读操作。
- 对特权 RCC 位或寄存器进行的非特权访问将被丢弃：位读为零，并忽略对这些位的写操作 (RAZ/WI)。

表 68. RCC 特权配置汇总

RCC 功能	特权位	相应的寄存器
HSI	HSION, HSIKERON, HSIRDY	RCC_CR
	HSICAL[11:0], HSITRIM[6:0]	RCC_HSICFGR
	HSIRDYIE	RCC_CIER
	HSIRDYIF	RCC_CIFR
	HSIRDYC	RCC_CICR
HSE	HSEON, HSERDY, HSEBYP, HSECSSON, HSEEXT	RCC_CR
	HSERDYIE	RCC_CIER
	HSERDYIF, HSECSSF	RCC_CIFR
	HSERDYC, HSECSSC	RCC_CICR
CSI	CSISON, CSIKERON, CSISRDY	RCC_CR
	CSICAL[7:0], CSITRIM[5:0]	RCC_CSICFGR
	CSISRDYIE	RCC_CIER
	CSISRDYIF	RCC_CIFR
	CSISRDYIC	RCC_CICR
LSI	LSION, LSIRDY, LSIPREDIV, LSCOSEL, LSCOEN	RCC_BDCR
	LSIRDYIE	RCC_CIER
	LSIRDYIF	RCC_CIFR
	LSIRDYC	RCC_CICR

表 68. RCC 特权配置汇总（续）

RCC 功能	特权位	相应的寄存器
LSE	LSECSSON, LSECSSD, LSEDRV[1:0], LSEBYP, LSERDY, LSEON, LSESYSRDY, LSESYSEN, LSCOSEL, LSCOEN	RCC_BDCR
	LSERDYIE	RCC_CIER
	LSERDYF	RCC_CIFR
	LSERDYC	RCC_CICR
SYSCLK	SW[1:0], SWS[1:0], STOPWUCK, STOPKERWUCK, MCO1SEL[3:0], MCO1PRE[2:0], MCO2SEL[3:0], MCO2PRE[2:0]	RCC_CFGR
	SYSTICKSEL[1:0]	RCC_CCIPR4
	VOS[1:0]	PWR_VOSR
	TIMPRE	RCC_CFGR
PRESC	HPRE[3:0], PPRE1[2:0], PPRE2[2:0], PPRE3[2:0]	RCC_CFGR2
PLL1	PLL1SRC[1:0], PLL1RGE[1:0], PLL1FRACEN, PLL1M[3:0], PLL1VCOSEL, PLL1PEN, PLL1QEN, PLL1REN	RCC_PLL1CFGR
	PLL1N[8:0], PLL1P[6:0], PLL1Q[6:0], PLL1R[6:0]	RCC_PLL1DIVR
	PLL1FRACN[12:0]	RCC_PLL1FRACR
	PLL1RDY, PLL1ON	RCC_CR
	PLL1RDYIE	RCC_CIER
	PLL1RDYF	RCC_CIFR
	PLL1RDYC	RCC_CICR
PLL2	PLL2SRC[1:0], PLL2RGE[1:0], PLL2FRACEN, PLL2M[3:0], PLL2PEN, PLL2QEN, PLL2REN, PLL2VCOSEL	RCC_PLL2CFGR
	PLL2N[8:0], PLL2P[6:0], PLL2Q[6:0], PLL2R[6:0]	RCC_PLL2DIVR
	PLL2FRACN[12:0]	RCC_PLL2FRACR
	PLL2RDY, PLL2ON	RCC_CR
	PLL2RDYIE	RCC_CIER
	PLL2RDYF	RCC_CIFR
	PLL2RDYC	RCC_CICR
HSI48	HSI48ON, HSI48RDY	RCC_CR
	HSI48CAL[9:0]	RCC_CRRCR
	HSI48RDYE	RCC_CIER
	HSI48RDYF	RCC_CIFR
	HSI48RDYC	RCC_CICR
IPKERSECCFG	CKERPSEL[1:0]	RCC_CCIPR5
清除复位标志	RMVF	RCC_RSR

10.6 RCC 低功耗模式

- 可通过软件禁止 AHB 和 APB 外设时钟，包括 DMA 时钟。
- 在睡眠模式下，CPU 时钟停止工作。在睡眠模式下，可通过软件停止存储器接口时钟（Flash、缓存和所有 SRAM 接口）。当连接到 AHB-APB 总线桥时钟的所有外设时钟均被禁止时，睡眠模式期间将通过硬件禁止 AHB-APB 总线桥时钟。
- 在停止模式下，将停止内核域中的所有时钟，并禁止 PLL、HSI、HSI48、CSI 和 HSE 振荡器。但是，可以开启 HSI（最高 64 MHz）或 CSI 以生成唤醒中断。LSI 和 LSE 在停止模式下保持工作状态。
- 在待机模式下，将停止内核域中的所有时钟，并禁止 PLL、HSI、HSI48、CSI 和 HSE 振荡器。

通过将 DBGMCU_CR 寄存器中的 DBG_STOP 或 DBG_STANDBY 位置 1，可以覆盖 CPU 的深度睡眠模式以进行调试。

退出停止模式时，系统时钟为 HSI（最高 64 MHz）或 CSI，具体取决于 *RCC 时钟配置寄存器 1 (RCC_CFGR1)* 中 STOPWUCK 的软件配置。HSI 的频率（范围和用户微调）是进入停止模式之前配置的频率。

除了系统时钟使用的振荡器外，其他内部振荡器可以自动唤醒，以避免在器件返回运行模式时等待其他振荡器唤醒时间。这是通过 RCC_CFGR1 中的 STOPKERWUCK 实现的。

退出待机模式时，系统时钟为 HSI (32 MHz)。用户微调将丢失。

如果正在执行 Flash 编程操作，则将延迟到 Flash 接口访问结束后再进入停止或待机模式。如果正在访问 APB 域，则将延迟到 APB 访问结束后再进入停止或待机模式。

10.7 RCC 中断

下表汇总了中断源及其控制方式。

表 69. 中断源和控制

中断向量	中断事件标志	说明	使能控制位	中断清除方法	退出睡眠模式	退出停止、待机模式
RCC	LSIRDYF	LSI 就绪	LSIRDYIE	将 LSIRDYC 置 1	是	否
	LSERDYF	LSE 就绪	LSERDYIE	将 LSERDYC 置 1	是	否
	HSIDRYF	HSI 就绪	HSIDRYIE	将 HSIRDYC 置 1	是	否
	HSERDYF	HSE 就绪	HSERDYIE	将 HSERDYC 置 1	是	否
	CSISRDYF	CSIS 就绪	CSISRDYIE	将 CSISRDYC 置 1	是	否
	HSI48RDYF	HSI48 就绪	HSI48RDYIE	将 HSI48RDYC 置 1	是	否
	PLL1RDYF	PLL1 就绪	PLL1RDYIE	将 PLL1RDYC 置 1	是	否
	PLL2RDYF	PLL2 就绪	PLL2RDYIE	将 PLL2RDYC 置 1	是	否
TAMP	ITAMP3F(1)	LSE CSS 故障	LSECSSON、ITAMP3E(1) 和 ITAMP3IE(1)	将 CITAMP3F(1) 置 1	是	是
NMI	HSECSSF	HSE CSS 故障	-(2)	将 HSECSSC 置 1	是	否

1. LSE CSS 故障事件 (LSECSSD) 连接到 TAMP 内部入侵 3。为了使中断与该事件相关联，必须使能内部入侵 3 和内部入侵 3 中断。ITAMP3F、ITAMP3E、ITAMP3IE 和 CITAMP3F 位在 TAMP 外设中。
2. 使能安全系统功能（HSECSSON = 1）时，不能屏蔽该中断。

10.8 RCC 寄存器

10.8.1 RCC 时钟控制寄存器 (RCC_CR)

RCC clock control register

偏移地址：0x000

复位值：0x0000 002B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	PLL2RDY	PLL2ON	PLL1RDY	PLL1ON	Res.	Res.	Res.	HSEEXT	HSECSSON	HSEBYP	HSERDY	HSEON
				r	rw	r	rw				rw	rs	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	HSI48RDY	HSI48ON	Res.	CSIKERON	CSIRDY	CSION	Res.	Res.	HSIDIVF	HSIDIV[1:0]		HSIKERON	HSIRDY	HSION
		r	rw		rw	r	rw			r	rw	rw	rw	r	rw

位 31:28 保留，必须保持复位值。

位 27 **PLL2RDY**：PLL2 时钟就绪标志 (PLL2 clock ready flag)
由硬件置 1，用于指示 PLL 已锁定。
0：PLL2 未锁定
1：PLL2 已锁定

位 26 **PLL2ON**：PLL2 使能 (PLL2 enable)
由软件置 1 和清零，用于使能 PLL2。
当进入停机或待机模式时由硬件清零。
0：PLL2 关闭（复位后的默认值）
1：PLL2 开启

位 25 **PLL1RDY**：PLL1 时钟就绪标志 (PLL1 clock ready flag)
由硬件置 1，用以指示 PLL1 已锁定。
0：PLL1 未锁定（复位后的默认值）
1：PLL1 已锁定

位 24 **PLL1ON**：PLL1 使能 (PLL1 enable)
由软件置 1 和清零，用于使能 PLL1。
当进入停机或待机模式时由硬件清零。请注意，如果将 PLL1 输出用作系统时钟，则硬件可防止向该位写入 0。
0：PLL1 关闭（复位后的默认值）
1：PLL1 开启

位 23:21 保留，必须保持复位值。

位 20 **HSEEXT**：旁路模式下的外部高速时钟类型 (external high speed clock type in Bypass mode)
由软件置 1 和复位，用于选择外部时钟的类型（模拟或数字）。
外部时钟必须通过 HSEON 位使能才能为器件使用。HSEEXT 只有在 HSE 振荡器已禁止的情况下才可写入。
0：HSE 处于模拟模式（复位后的默认值）
1：HSE 处于数字模式

位 19 **HSECSSON**：HSE 时钟安全系统使能 (HSE clock security system enable)

由软件置 1，用于使能 HSE 上的时钟安全系统。

该位“仅置 1”（由系统复位或在系统进入待机模式时禁止）。当 HSECSSON 置 1 时，时钟监测器将在 HSE 就绪时由硬件使能，并在检测到振荡器故障时由硬件禁止。

0：HSE 上的 CSS 关闭（时钟监测器关闭）（复位后的默认值）

1：HSE 上的 CSS 开启（如果 HSE 振荡器稳定，则时钟监测器开启；如果不稳定，则关闭）

位 18 **HSEBYP**：HSE 时钟旁路 (HSE clock bypass)

由软件置 1 和清零，用于用外部时钟旁路振荡器。外部时钟必须通过 HSEON 位使能才能为器件使用。

HSEBYP 只有在 HSE 振荡器已禁止的情况下才可写入。

0：不旁路 HSE 振荡器（复位后的默认值）

1：外部时钟旁路 HSE 振荡器

位 17 **HSERDY**：HSE 时钟就绪标志 (HSE clock ready flag)

由硬件置 1，用以指示 HSE 振荡器已稳定。

0：HSE 时钟未就绪（复位后的默认值）

1：HSE 时钟已就绪

位 16 HSEON：HSE 时钟使能 (HSE clock enable)

由软件置 1 和清零。

由硬件清零，用于在进入停止或待机模式时停止 HSE。

如果直接将 HSE 用作系统时钟，或将 HSE 选作已使能的 PLL1（PLL1ON 位置 1）的参考时钟，则不能将该位清零。

0：HSE 关闭（复位后的默认值）

1：HSE 开启

位 15:14 保留，必须保持复位值。

位 13 **HSI48RDY**：HSI48 时钟就绪标志 (HSI48 clock ready flag)

由硬件置 1，用以指示 HSI48 振荡器已稳定。

0：HSI48 时钟未就绪（复位后的默认值）

1：HSI48 时钟已就绪

位 12 **HSI48ON**：HSI48 时钟使能 (HSI48 clock enable)

由软件置 1，在系统进入停止

或待机模式时由软件或硬件清零。

0：HSI48 关闭（复位后的默认值）

1：HSI48 开启

位 11 保留，必须保持复位值。

位 10 **CSIKERON**：停止模式下的 CSI 时钟使能 (CSI clock enable in stop mode)

由软件置 1 和复位，即使在停止模式下也可强制 CSI 开启，从而可快速用作某些外设的内核时钟。该位对 CSION 的值没有任何影响。

0：对 CSI 无影响（复位后的默认值）

1：即使在停止模式下也强制 CSI 开启

位 9 **CSIRDY**：CSI 时钟就绪标志 (CSI clock ready flag)

由硬件置 1，用以指示 CSI 振荡器已稳定。仅在由 CSION 使能 RC 时才会激活该位（由 CSIKERON 或外设请求使能 CSI 时不会激活该位）。

0：CSI 时钟未就绪（复位后的默认值）

1：CSI 时钟已就绪

位 8 **CSION**：CSI 时钟使能 (CSI clock enable)

由软件置 1 和复位，用于使能/禁止系统和/或外设的 CSI 时钟。

如果 STOPWUCK = 1 或 STOPKERWUCK = 1，则由硬件置 1，以在系统退出停止模式时强制 CSI 开启。

如果直接将 CSI 用作系统时钟，或将 CSI 选作已使能的 PLL1（PLL1ON 位置 1）的参考时钟，则不能将该位清零。

0：CSI 关闭（复位后的默认值）

1：CSI 开启

位 7:6 保留，必须保持复位值。

位 5 **HSIDIVF**：HSI 分频器标志 (HSI divider flag)

由硬件置 1 和复位。

由于对 HSIDIV 的写操作不会直接对频率起作用，因此该标志指示的是 HSI 分频器的当前状态。HSIDIVF 会在 HSIDIV 值发生变化时立即变为 0，并会在输出频率与编程到 HSIDIV 中的值匹配时重新置 1。

0：新分频比尚未传播到 hsi_ker_ck（复位后的默认值）

1：hsi_ker_ck 时钟频率反映新 HSIDIV 值（时钟设置完成时的默认寄存器值）

位 4:3 **HSIDIV[1:0]**：HSI 时钟分频器 (HSI clock divider)

由软件置 1 和复位。

可使用这些位来选择一个分频比，以配置所需 HSI 时钟频率。

如果将 HSI 选作至少一个已使能 PLL（PLLxON 位置 1）的参考时钟，则不能更改 HSIDIV。在这种情况下，将忽略新 HSIDIV 值。

00：采用 1 分频，hsi_ker_ck = 64 MHz

01：采用 2 分频，hsi_ker_ck = 32 MHz（系统复位和退出待机模式后的默认值）

10：采用 4 分频，hsi_ker_ck = 16 MHz

11：采用 8 分频，hsi_ker_ck = 8 MHz

位 2 **HSIKERON**：停止模式下的 HSI 时钟使能 (HSI clock enable in stop mode)

由软件置 1 和复位，即使在停止模式下也可强制 HSI 开启，从而可快速用作外设的内核时钟。该位对 HSION 的值没有任何影响。

0：对 HSI 无影响（复位后的默认值）

1：即使在停止模式下也强制 HSI 开启

位 1 **HSIRDY**：HSI 时钟就绪标志 (HSI clock ready flag)

由硬件置 1，用以指示 HSI 振荡器已稳定。

0：HSI 时钟未就绪（复位后的默认值）

1：HSI 时钟已就绪

位 0 **HSION**：HSI 时钟使能 (HSI clock enable)

由软件置 1 和清零。

如果 STOPWUCK = 1 或 STOPKERWUCK = 1，则由硬件置 1，以在产品退出停止模式时强制 HSI 开启。

由硬件置 1，用于在产品退出待机模式时或在用作系统时钟源的 HSE 发生故障时强制 HSI 开启。

如果直接将 HSI 用作系统时钟，或将 HSI 选作已使能的 PLL1（PLL1ON 位置 1）的参考时钟，则不能将该位清零。

0：HSI 关闭

1：HSI 开启（复位后的默认值）

10.8.2 RCC HSI 校准寄存器 (RCC_HSICFGR)

RCC HSI calibration register

偏移地址：0x010

复位值：0x0040 0XXX

复位值取决于 Flash 选项字节设置。

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

位 31:23 保留，必须保持复位值。

位 22:16 **HSITRIM[6:0]**：HSI 时钟微调 (HSI clock trimming)

由软件置 1，用于调整校准。

HSITRIM 位域与复位阶段加载的工程选项字节 (FLASH_HSI_OPT) 相加，以构成校准微调值。

HSICAL = HSITRIM + FLASH_HSI_OPT。

更改 HSITRIM 后，需要一个系统时钟周期才能更新为新的 HSITRIM 值

注意：位域的复位值为 0x40。

位 15:12 保留，必须保持复位值。

位 11:0 **HSICAL[11:0]**：HSI 时钟校准 (HSI clock calibration)

由硬件通过系统复位 nreset 期间的选项字节加载来置 1。由软件通过微调位 HSITRIM 进行调节。

该位域表示工程选项字节校准值与 HSITRIM 位值之和。

10.8.3 RCC 时钟恢复 RC 寄存器 (RCC_CRRCR)

RCC clock recovery RC register

偏移地址：0x014

复位值：0x0000 0XXX

复位值取决于 Flash 选项字节设置。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	HSI48CAL[9:0]									
						r	r	r	r	r	r	r	r	r	r

位 31:10 保留，必须保持复位值。

位 9:0 **HSI48CAL[9:0]**：内部 RC 48 MHz 时钟校准 (Internal RC 48 MHz clock calibration)

由硬件通过系统复位 NRESET 期间的选项字节加载来置 1。只读。

10.8.4 RCC CSI 校准寄存器 (RCC_CSICFGR)

RCC CSI calibration register

偏移地址：0x018

复位值：0x0020 00XX

复位值取决于 Flash 选项字节设置。

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

位 31:22 保留，必须保持复位值。

位 21:16 **CSITRIM[5:0]**：CSI 时钟微调 (CSI clock trimming)

由软件置 1，用于调整校准。

CSITRIM 位域与复位阶段加载的工程选项字节 (FLASH_CSI_OPT) 相加，以构成校准微调值。

CSICAL = CSITRIM + FLASH_CSI_OPT。

注意：位域的复位值为 0x20。

位 15:8 保留，必须保持复位值。

位 7:0 **CSICAL[7:0]**：CSI 时钟校准 (CSI clock calibration)

由硬件通过系统复位 NRESET 期间的选项字节加载来置 1。由软件通过微调位 CSITRIM 进行调节。

该位域表示工程选项字节校准值与 CSITRIM 位值之和。

10.8.5 RCC 时钟配置寄存器 1 (RCC_CFGR1)

RCC clock configuration register 1

偏移地址：0x01C

复位值：0x0000 0000

访问：0 ≤ 等待周期 ≤ 2；按字、半字和字节访问。

只有在时钟源切换期间进行访问时才会插入 1 或 2 个等待周期。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCO2SEL[2:0]			MCO2PRE[3:0]				MCO1SEL[2:0]			MCO1PRE[3:0]				Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMPRE	Res.	RTCPRE[5:0]						STOPKERWUCK	STOPWUCK	Res.	SWS[1:0]		Res.	SW[1:0]	
rw		rw	rw	rw	rw	rw	rw	rw	rw		r	r		rw	rw

位 31:29 **MCO2SEL[2:0]**：微控制器时钟输出 2 (microcontroller clock output 2)
由软件置 1 和清零。时钟源选择可能会造成对 MCO2 的干扰。
强烈建议仅在复位后但在使能外部振荡器和 PLL 之前来配置这些位。
000：选择系统时钟 (**sys_ck**)（复位后的默认值）
001：选择 PLL2 振荡器时钟 (**pll2_p_ck**)
010：选择 HSE 时钟 (**hse_ck**)
011：选择 PLL1 时钟 (**pll1_p_ck**)
100：选择 CSI 时钟 (**csi_ck**)
101：选择 LSI 时钟 (**lsi_ck**)
其他值：保留

位 28:25 **MCO2PRE[3:0]**：MCO2 预分频器 (MCO2 prescaler)
由软件置 1 和清零，用于配置 MCO2 的预分频器。对此预分频器进行修改可能会对 MCO2 造成干扰。强烈建议仅在复位后且在使能外部振荡器和 PLL 之前进行此分频器的更改。
0000：禁止预分频器（复位后的默认值）
0001：1 分频（旁路）
0010：2 分频
0011：3 分频
0100：4 分频
...
1111：15 分频

位 24:22 **MCO1SEL[2:0]**：微控制器时钟输出 1 (Microcontroller clock output 1)
由软件置 1 和清零。时钟源选择可能会造成对 MCO1 的干扰。
强烈建议仅在复位后但在使能外部振荡器和 PLL 之前来配置这些位。
000：选择 HSI 时钟 (**hsi_ck**)（复位后的默认值）
001：选择 LSE 振荡器时钟 (**lse_ck**)
010：选择 HSE 时钟 (**hse_ck**)
011：选择 PLL1 时钟 (**pll1_q_ck**)
100：选择 HSI48 时钟 (**hsi48_ck**)
其他值：保留

位 21:18 **MCO1PRE[3:0]**：MCO1 预分频器 (MCO1 prescaler)
由软件置 1 和清零，用于配置 MCO1 的预分频器。对此预分频器进行修改可能会对 MCO1 造成干扰。强烈建议仅在复位后且在使能外部振荡器和 PLL 之前进行此分频器的更改。
0000：禁止预分频器（复位后的默认值）
0001：1 分频（旁路）
0010：2 分频
0011：3 分频
0100：4 分频
...
1111：15 分频

位 17:16 保留，必须保持复位值。

位 15 **TIMPRE**：定时器时钟预分频器选择 (timers clocks prescaler selection)
此位由软件置 1 和复位，用于控制连接到 APB1 和 APB2 域的所有定时器的时钟频率。
0：如果 PPRE1 或 PPRE2 对应于 1 或 2 分频，则定时器内核时钟为 **rcc_hclk1**；否则为 2 x F_{rcc_pclk1} 或 2 x F_{rcc_pclk2}（复位后的默认值）
1：如果 PPRE1 或 PPRE2 对应于 1、2 或 4 分频，则定时器内核时钟为 2 x F_{rcc_pclk1} 或 2 x F_{rcc_pclk2}；否则为 4 x F_{rcc_pclk1} 或 4 x F_{rcc_pclk2}

位 14 保留，必须保持复位值。

位 13:8 **RTCPRE[5:0]**：适用于 RTC 时钟的 HSE 分频系数 (HSE division factor for RTC clock)

由软件置 1 和清零，用于对 HSE 进行分频，进而为 RTC 生成时钟。

小心：软件必须正确设置这些位，以确保提供给 RTC 的时钟小于 1 MHz。在选择 RTC 时钟源之前必须配置这些位。

000000：无时钟（复位后的默认值）

000001：无时钟

000010：HSE/2

000011：HSE/3

000100：HSE/4

...

111110：HSE/62

111111：HSE/63

位 7 **STOPKERWUCK**：将系统从停止模式唤醒后的内核时钟选择 (kernel clock selection after a wakeup from system Stop)

由软件置 1 和复位，用于选择将系统从停止模式唤醒后的内核时钟。

0：选择 HSI（最高 64 MHz）作为将系统从停止模式唤醒后的时钟（复位后的默认值）

1：选择 CSI 作为将系统从停止模式唤醒后的时钟

位 6 **STOPWUCK**：将系统从停止模式唤醒后的系统时钟选择 (system clock selection after a wakeup from system Stop)

由软件置 1 和复位，用于选择将系统从停止模式唤醒后的系统时钟。

所选时钟还用作 HSE 上的时钟安全系统 (CSS) 的紧急时钟。

0：选择 HSI（最高 64 MHz）作为将系统从停止模式唤醒后的时钟（复位后的默认值）

1：选择 CSI 作为将系统从停止模式唤醒后的时钟

小心： 当 CSS 已使能（通过 HSECSSON 位）且系统时钟为 HSE (SWS = 10) 或者请求开启 HSE (SW =10) 时，不能修改 STOPWUCK。

位 5 保留，必须保持复位值。

位 4:3 **SWS[1:0]**：系统时钟切换状态 (system clock switch status)

由硬件置 1 和复位，用于指示用作系统时钟的时钟源。

00：将 HSI 用作系统时钟 (**hsi_ck**)（复位后的默认值）

01：将 CSI 用作系统时钟 (**csi_ck**)

10：将 HSE 用作系统时钟 (**hse_ck**)

11：将 PLL1 用作系统时钟 (**pll1_p_ck**)

位 2 保留，必须保持复位值。

位 1:0 **SW[1:0]**：系统时钟和跟踪时钟切换 (system clock and trace clock switch)

由软件置 1 和复位，用于选择系统时钟和跟踪时钟源 (**sys_ck**)。

由硬件置 1，用于：

-在系统退出停止模式时强制选择 HSI 或 CSI（具体取决于 STOPWUCK 选择）

-在 HSE 直接或间接用作系统时钟的情况下发生故障时，强制选择 HSI

00：将 HSI 选作系统时钟 (**hsi_ck**)（复位后的默认值）

01：将 CSI 选作系统时钟 (**csi_ck**)

10：将 HSE 选作系统时钟 (**hse_ck**)

11：将 PLL1 选作系统时钟 (**pll1_p_ck** 用作 **sys_ck**)

10.8.6 RCC CPU 域时钟配置寄存器 2 (RCC_CFGR2)

RCC CPU domain clock configuration register 2

偏移地址：0x020

复位值：0x0000 0000

只有在时钟源切换期间进行访问时才会插入 1 或 2 个等待周期。

如果在更新 APB 或 AHB 预分频器值时进行访问，则插入 0 到 15 个等待周期。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	APB3DIS	APB2DIS	APB1DIS	Res.	Res.	AHB2DIS	AHB1DIS
									rw	rw	rw			rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PPRE3[2:0]			Res.	PPRE2[2:0]			Res.	PPRE1[2:0]			HPRE[3:0]			
	rw	rw	rw		rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

位 31:23 保留，必须保持复位值。

位 22 **APB3DIS**：APB3 时钟禁止值 (APB3 clock disable value)。由软件置 1 和清零。

当所有 APB3 外设均未使用并且其时钟已在 RCC_APB3ENR 中禁止时，可将该位置 1 以进一步降低功耗。该位置 1 时，所有 APB3 外设的时钟都将关闭。

0：使能 APB3 时钟，并根据其专用时钟使能控制位分配给相应的外设

1：禁止 APB3 时钟

位 21 **APB2DIS**：APB2 时钟禁止值 (APB2 clock disable value)

当所有 APB2 外设均未使用并且其时钟已在 RCC_APB2ENR 中禁止时，可将该位置 1 以进一步降低功耗。当此位置 1 时，所有 APB2 外设的时钟都将关闭。

0：使能 APB2 时钟，并根据其专用时钟使能控制位分配给相应的外设

1：禁止 APB2 时钟

位 20 **APB1DIS**：APB1 时钟禁止值 (APB1 clock disable value)

当所有 APB1 外设（除 IWDG 外）均未使用并且其时钟已在 RCC_APB1ENR 中禁止时，可将该位置 1 以进一步降低功耗。

该位置 1 时，所有 APB1 外设（IWDG 除外）的时钟都将关闭。

0：使能 APB1 时钟，并根据其专用时钟使能控制位分配给相应的外设

1：禁止 APB1 时钟

位 19:18 保留，必须保持复位值。

位 17 **AHB2DIS**：AHB2 时钟禁止 (AHB2 clock disable)

当 RCC_AHB2ENR 中涉及的所有 AHB2 外设均未使用并且其时钟已在 RCC_AHB2ENR 中禁止时，可将该位置 1 以进一步降低功耗。该位置 1 时，RCC_AHB2ENR 中涉及的所有 AHB2 外设的时钟都将关闭。

0：使能 AHB2 时钟，并根据其专用时钟使能控制位分配给相应的外设

1：禁止 AHB2 时钟

位 16 **AHB1DIS**：AHB1 时钟禁止 (AHB1 clock disable)

当所有 AHB1 外设（后面列出的外设除外）均未使用并且其时钟已在 RCC_AHB1ENR 中禁止时，可将该位置 1 以进一步降低功耗。该位置 1 时，RCC_AHB1ENR 中涉及的所有 AHB1 外设（FLASH、BKPSRAM、ICACHE 和 SRAM1 除外）的时钟都将关闭。

0：使能 AHB1 时钟，并根据其专用时钟使能控制位分配给相应的外设

1：禁止 AHB1 时钟

位 15 保留，必须保持复位值。

位 14:12 **PPRE3[2:0]**：APB 低速预分频器 (APB3) (APB low-speed prescaler (APB3))
由软件置 1 和复位，用于控制 APB 低速时钟分频系数。
在 PPRE3 写入后，时钟将通过 1 APB 到 16 APB 周期新预分频系数进行分频。
0xx：rcc_pclk3 = rcc_hclk1
100：rcc_pclk3 = rcc_hclk1 / 2
101：rcc_pclk3 = rcc_hclk1 / 4
110：rcc_pclk3 = rcc_hclk1 / 8
111：rcc_pclk3 = rcc_hclk1 / 16

位 11 保留，必须保持复位值。

位 10:8 **PPRE2[2:0]**：APB 高速预分频器 (APB2) (APB high-speed prescaler (APB2))
由软件置 1 和复位，用于控制 APB 高速时钟分频系数。
在 PPRE2 写入后，时钟将通过 1 APB 到 16 APB 周期新预分频系数进行分频。
0xx：rcc_pclk2 = rcc_hclk1
100：rcc_pclk2 = rcc_hclk1 / 2
101：rcc_pclk2 = rcc_hclk1 / 4
110：rcc_pclk2 = rcc_hclk1 / 8
111：rcc_pclk2 = rcc_hclk1 / 16

位 7 保留，必须保持复位值。

位 6:4 **PPRE1[2:0]**：APB 低速预分频器 (APB1) (APB low-speed prescaler (APB1))
由软件置 1 和复位，用于控制 rcc_pclk1 的分频系数。
在 PPRE 写入后，时钟将由介于 1 到 16 个 rcc_hclk 周期之间的新预分频系数进行分频。
0xx：rcc_pclk1 = rcc_hclk1（复位后的默认值）
100：rcc_pclk1 = rcc_hclk1 / 2
101：rcc_pclk1 = rcc_hclk1 / 4
110：rcc_pclk1 = rcc_hclk1 / 8
111：rcc_pclk1 = rcc_hclk1 / 16

位 3:0 **HPRE[3:0]**：AHB 预分频器 (AHB prescaler)
由软件置 1 和复位，用于控制 rcc_hclk 的分频系数。更改
此分频比会影响所有总线矩阵时钟的频率
0xxx：rcc_hclk = sys_ck（复位后的默认值）
1000：rcc_hclk = sys_ck / 2
1001：rcc_hclk = sys_ck / 4
1010：rcc_hclk = sys_ck / 8
1011：rcc_hclk = sys_ck / 16
1100：rcc_hclk = sys_ck / 64
1101：rcc_hclk = sys_ck / 128
1110：rcc_hclk = sys_ck / 256
1111：rcc_hclk = sys_ck / 512

小心： 使用电压调节功能时必须谨慎。由于新分频系数存在传播延迟，因此在预分频系数发生更改之后以及在 VCORE 电压下降之前，必须对该寄存器进行读取，以检查是否已将新预分频器值考虑在内。
根据时钟源频率和电压范围，软件应用程序必须在 HPRE 中编程一个合适的值，以确保系统频率不会超出最大频率。

10.8.7 RCC PLL 时钟源选择寄存器 (RCC_PLL1CFGR)

RCC PLL clock source selection register

偏移地址：0x028

复位值：0x0000 0000

访问：无等待周期，按字、半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL1REN	PLL1QEN	PLL1PEN
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PLL1M[5:0]						Res.	Res.	PLL1VCOSEL	PLL1FRACEN	PLL1RGE[1:0]		PLL1SRC[1:0]	
		rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw

位 31:19 保留，必须保持复位值。

位 18 **PLL1REN**：PLL1 DIVR 分频器输出使能 (PLL1 DIVR divider output enable)
由软件置 1 和复位，用于使能 PLL1 的 pll1_r_ck 输出。
为实现节能，不使用 pll1_r_ck 时，必须将 DIVR1EN 和 DIVR1 位设为 0。该位只能在 PLL1 禁止（PLL1ON = 0 且 PLL1RDY = 0）时写入。
0：禁止 pll1_r_ck 输出（复位后的默认值）
1：使能 pll1_r_ck 输出

位 17 PLL1QEN：PLL1 DIVQ 分频器输出使能 (PLL1 DIVQ divider output enable)
由软件置 1 和复位，用于使能 PLL1 的 pll1_q_ck 输出。
为实现节能，不使用 PLL1 的 pll1_q_ck 输出时，必须禁止 pll1_q_ck。
该位只能在 PLL1 禁止（PLL1ON = 0 且 PLL1RDY = 0）时写入。
0：禁止 pll1_q_ck 输出（复位后的默认值）
1：使能 pll1_q_ck 输出

位 16 **PLL1PEN**：PLL1 DIVP 分频器输出使能 (PLL1 DIVP divider output enable)
由软件置 1 和复位，用于使能 PLL1 的 pll1_p_ck 输出。
该位只能在 PLL1 禁止（PLL1ON = 0 且 PLL1RDY = 0）时写入。
为实现节能，不使用 PLL1 的 pll1_p_ck 输出时，必须禁止 pll1_p_ck。
0：禁止 pll1_p_ck 输出（复位后的默认值）
1：使能 pll1_p_ck 输出

位 15:14 保留，必须保持复位值。

位 13:8 **PLL1M[5:0]**：PLL1 的预分频器 (prescaler for PLL1)

由软件置 1 和清零，用于配置 PLL1 的预分频器。

使能 PLL1（PLL1ON = 1 或 PLL1RDY = 1）时，硬件不允许对该预分频器进行任何修改。

为实现节能，不使用 PLL1 时，必须将 PLL1M 的值设为 0。

000000：禁止预分频器（复位后的默认值）

000001：1 分频（旁路）

000010：2 分频

000011：3 分频

...

100000：32 分频

...

111111：63 分频

位 7:6 保留，必须保持复位值。

位 5 **PLL1VCOSEL**：PLL1 VCO 选择 (PLL1 VCO selection)

由软件置 1 和复位，用于为 PLL1 选用合适的 VCO 频率范围。该位必须在使能 PLL1 之前写入。

0：宽 VCO 范围，192 MHz 到 836 MHz（复位后的默认值）

1：中等 VCO 范围，150 MHz 到 420 MHz

位 4 **PLL1FRACEN**：PLL1 小数锁存使能 (PLL1 fractional latch enable)

由软件置 1 和复位，用于将 FRACN1 的内容锁存到 sigma-delta 调制器中。

为将 FRACN1 值锁存到 sigma-delta 调制器中，必须将 PLL1FRACEN 设为 0，然后将其置 1。从 0 到 1 的转换会将 FRACN1 的内容传送到调制器中。

位 3:2 **PLL1RGE[1:0]**：PLL1 输入频率范围 (PLL1 input frequency range)

由软件置 1 和复位，用于为 PLL1 选用合适的参考频率范围。该位必须在使能 PLL1 之前写入。

00：PLL1 输入 (ref1_ck) 时钟频率范围介于 1 MHz 到 2 MHz 之间（复位后的默认值）01：PLL1 输入 (ref1_ck) 时钟频率范围介于 2 MHz 到 4 MHz 之间

10：PLL1 输入 (ref1_ck) 时钟频率范围介于 4 MHz 到 8 MHz 之间

11：PLL1 输入 (ref1_ck) 时钟频率范围介于 8 MHz 到 16 MHz 之间

位 1:0 **PLL1SRC[1:0]**：DIVMx 和 PLL 时钟源选择 (DIVMx and PLLs clock source selection)

由软件置 1 和复位，用于选择 PLL 时钟源。这些位只能在所有 PLL 均已禁止时写入。

为实现节能，不使用 PLL 时，必须将 PLL1SRC 的值设为“00”。00：未向 DIVMx 分频器和 PLL 发送任何时钟（复位后的默认值）

01：将 HSI 选作 PLL 时钟 (hsi_ck)

10：将 CSI 选作 PLL 时钟 (csi_ck)

11：将 HSE 选作 PLL 时钟 (hse_ck)

10.8.8 RCC PLL 时钟源选择寄存器 (RCC_PLL2CFGR)

RCC PLL clock source selection register

偏移地址：0x02C

复位值：0x0000 0000

访问：无等待周期，按字、半字和字节访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL2REN	PLL2QEN	PLL2PEN
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PLL2M[5:0]						Res.	Res.	PLL2VCOSEL	PLL2FRACEN	PLL2RGE[1:0]		PLL2SRC[1:0]	
		rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw

位 31:19 保留，必须保持复位值。

位 18 **PLL2REN**：PLL2 DIVR 分频器输出使能 (PLL2 DIVR divider output enable)
由软件置 1 和复位，用于使能 PLL2 的 pll2_r_ck 输出。
为实现节能，不使用 pll2_r_ck 时，必须将 DIVR2EN 和 DIVR2 位设为 0。
0：禁止 pll2_r_ck 输出（复位后的默认值）
1：使能 pll2_r_ck 输出

位 17 **PLL2QEN**：PLL2 DIVQ 分频器输出使能 (PLL2 DIVQ divider output enable)
由软件置 1 和复位，用于使能 PLL2 的 pll2_q_ck 输出。
为实现节能，不使用 PLL2 的 pll2_q_ck 输出时，必须禁止 pll2_q_ck。
0：禁止 pll2_q_ck 输出（复位后的默认值）
1：使能 pll2_q_ck 输出

位 16 **PLL2PEN**：PLL2 DIVP 分频器输出使能 (PLL2 DIVP divider output enable)
由软件置 1 和复位，用于使能 PLL2 的 pll2_p_ck 输出。
为实现节能，不使用 PLL2 的 pll2_p_ck 输出时，必须禁止 pll2_p_ck。
0：禁止 pll2_p_ck 输出（复位后的默认值）
1：使能 pll2_p_ck 输出

位 15:14 保留，必须保持复位值。

位 13:8 **PLL2M[5:0]**：PLL2 的预分频器 (prescaler for PLL2)
由软件置 1 和清零，用于配置 PLL2 的预分频器。
使能 PLL2（PLL2ON = 1 或 PLL2RDY = 1）时，硬件不允许对该预分频器进行任何修改。
为实现节能，不使用 PLL2 时，必须将 PLL2M 的值设为 0。
000000：禁止预分频器（复位后的默认值）
000001：1 分频（旁路）
000010：2 分频
000011：3 分频
...
100000：32 分频
...
111111：63 分频

位 7:6 保留，必须保持复位值。

位 5 **PLL2VCOSEL**：PLL2 VCO 选择 (PLL2 VCO selection)
由软件置 1 和复位，用于为 PLL2 选用合适的 VCO 频率范围。
该位必须在使能 PLL2 之前写入。
0：宽 VCO 范围，192 MHz 到 836 MHz（复位后的默认值）
1：中等 VCO 范围，150 MHz 到 420 MHz

位 4 **PLL2FRACEN**：PLL2 小数锁存使能 (PLL2 fractional latch enable)
由软件置 1 和复位，用于使能 PLL2 的 pll2_p_ck 输出。
为实现节能，不使用 PLL2 的 pll2_p_ck 输出时，必须禁止 pll2_p_ck。
0：禁止 pll2_p_ck 输出（复位后的默认值）
1：使能 pll2_p_ck 输出

位 3:2 **PLL2RGE[1:0]**：PLL2 输入频率范围 (PLL2 input frequency range)
由软件置 1 和复位，用于为 PLL2 选用合适的参考频率范围。这些位必须在使能 PLL2 之前写入。
00：PLL2 输入 (ref2_ck) 时钟频率范围介于 1 MHz 到 2 MHz 之间（复位后的默认值）01：PLL2 输入 (ref2_ck) 时钟频率范围介于 2 MHz 到 4 MHz 之间
10：PLL2 输入 (ref2_ck) 时钟频率范围介于 4 MHz 到 8 MHz 之间
11：PLL2 输入 (ref2_ck) 时钟频率范围介于 8 MHz 到 16 MHz 之间

位 1:0 **PLL2SRC[1:0]**：DIVMx 和 PLL 时钟源选择 (DIVMx and PLLs clock source selection)
由软件置 1 和复位，用于选择 PLL 时钟源。
这些位只能在所有 PLL 均已禁止时写入。
为实现节能，不使用 PLL 时，必须将 PLL2SRC 的值设为“00”。
00：未向 DIVMx 分频器和 PLL 发送任何时钟（复位后的默认值）
01：将 HSI 选作 PLL 时钟 (hsi_ck)
10：将 CSI 选作 PLL 时钟 (csi_ck)
11：将 HSE 选作 PLL 时钟 (hse_ck)

10.8.9 RCC PLL1 分频器寄存器 (RCC_PLL1DIVR)

RCC PLL1 dividers register

偏移地址：0x034

复位值：0x0101 0280

访问：无等待周期，按字、半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	PLL1R[6:0]							Res.	PLL1Q[6:0]							
	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLL1P[6:0]								PLL1N[8:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30:24 **PLL1R[6:0]**：PLL1 DIVR 分频系数 (PLL1 DIVR division factor)
由软件置 1 和复位，用于控制 pll1_r_ck 时钟的频率。
这些位只能在 PLL1 禁止（PLL1ON = 0 且 PLL1RDY = 0）时写入。
0000000：pll1_r_ck = vco1_ck
0000001：pll1_r_ck = vco1_ck / 2（复位后的默认值）
0000010：pll1_r_ck = vco1_ck / 3
0000011：pll1_r_ck = vco1_ck / 4
...
1111111：pll1_r_ck = vco1_ck / 128

位 23 保留，必须保持复位值。

位 22:16 **PLL1Q[6:0]**：PLL1 DIVQ 分频系数 (PLL1 DIVQ division factor)
由软件置 1 和复位，用于控制 pll1_q_ck 时钟的频率。
这些位只能在 PLL1 禁止（PLL1ON = 0 且 PLL1RDY = 0）时写入。
0000000：pll1_q_ck = vco1_ck
0000001：pll1_q_ck = vco1_ck / 2（复位后的默认值）
0000010：pll1_q_ck = vco1_ck / 3
0000011：pll1_q_ck = vco1_ck / 4
...
1111111：pll1_q_ck = vco1_ck / 128

位 15:9 PLL1P[6:0]：PLL1 DIVP 分频系数 (PLL1 DIVP division factor)
由软件置 1 和复位，用于控制 pll1_p_ck 时钟的频率。
这些位只能在 PLL1 禁止（PLL1ON = 0 且 PLL1RDY = 0）时写入。
请注意，不允许使用奇数分频系数。
0000000：不允许
0000001：pll1_p_ck = vco1_ck / 2（复位后的默认值）
0000010：不允许
0000011：pll1_p_ck = vco1_ck / 4
...
1111111：pll1_p_ck = vco1_ck / 128

位 8:0 **PLL1N[8:0]**：适用于 PLL1VCO 的倍频系数 (Multiplication factor for PLL1VCO)

由软件置 1 和复位，用于控制 VCO 的倍频系数。

这些位只能在 PLL 禁止（PLL1ON = 0 且 PLL1RDY = 0）时写入。

0x003：PLL1N = 4

0x004：PLL1N = 5

0x005：PLL1N = 6

...

0x080：PLL1N = 129（复位后的默认值）

...

0x1FF：PLL1N = 512

其他值：保留

10.8.10 RCC PLL1 小数分频器寄存器 (RCC_PLL1FRACR)

RCC PLL1 fractional divider register

偏移地址：0x038

复位值：0x0000 0000

访问：无等待周期；按字和半字访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLL1FRACN[12:0]													Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			

位 31:16 保留，必须保持复位值。

位 15:3 **PLL1FRACN[12:0]**：PLL1 VCO 的倍频系数的小数部分 (fractional part of the multiplication factor for PLL1 VCO)

由软件置 1 和复位，用于控制 VCO 的倍频系数的小数部分。这些位可随时写入，进而对 PLL1 VCO 进行动态微调。

小心： 软件必须正确设置这些位，以确保 VCO 输出频率介于其有效频率范围内，即：
* 128 MHz 到 560 MHz（PLL1VCOSEL = 0）时
* 150 MHz 到 420 MHz（PLL1VCOSEL = 1）时
VCO 输出频率 = Fref1_ck x (PLL1N + (PLL1FRACN / 213))，其中
* PLL1N 介于 8 到 420 之间
* PLL1FRACN 可介于 0 到 213- 1 之间
* 输入频率 Fref1_ck 必须介于 1 MHz 到 16 MHz 之间。
如果执意要在已使能 PLL 的情况下实时更改 PLL1FRACN 值，应用程序必须按如下过程进行操作：
* 将位 PLL1FRACEN 设为 0
* 将新的小数值写入 PLL1FRACN
* 将位 PLL1FRACEN 置 1

位 2:0 保留，必须保持复位值。

10.8.11 RCC PLL1 分频器寄存器 (RCC_PLL2DIVR)

RCC PLL1 dividers register

偏移地址：0x03C

复位值：0x0101 0280

访问：无等待周期，按字、半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	PLL2R[6:0]							Res.	PLL2Q[6:0]						
	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLL2P[6:0]							PLL2N[8:0]								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30:24 **PLL2R[6:0]**：PLL2 DIVR 分频系数 (PLL2 DIVR division factor)

由软件置 1 和复位，用于控制 pll2_r_ck 时钟的频率。

这些位只能在 PLL1 禁止（PLL2ON = 0 且 PLL2RDY = 0）时写入。

0000000：pll2_r_ck = vco2_ck

0000001：pll2_r_ck = vco2_ck / 2（复位后的默认值）

0000010：pll2_r_ck = vco2_ck / 3

0000011：pll2_r_ck = vco2_ck / 4

...

1111111：pll2_r_ck = vco2_ck / 128

位 23 保留，必须保持复位值。

位 22:16 **PLL2Q[6:0]**：PLL2 DIVQ 分频系数 (PLL2 DIVQ division factor)

由软件置 1 和复位，用于控制 pll2_q_ck 时钟的频率。

这些位只能在 PLL2 禁止（PLL2ON = 0 且 PLL2RDY = 0）时写入。

0000000：pll2_q_ck = vco2_ck

0000001：pll2_q_ck = vco2_ck / 2（复位后的默认值）

0000010：pll2_q_ck = vco2_ck / 3

0000011：pll2_q_ck = vco2_ck / 4

...

1111111：pll2_q_ck = vco2_ck / 128

位 15:9 **PLL2P[6:0]**：PLL2 DIVP 分频系数 (PLL2 DIVP division factor)

由软件置 1 和复位，用于控制 pll2_p_ck 时钟的频率。

这些位只能在 PLL2 禁止（PLL2ON = 0 且 PLL2RDY = 0）时写入。

0000000：pll2_p_ck = vco2_ck

0000001：pll2_p_ck = vco2_ck / 2（复位后的默认值）

0000010：pll2_p_ck = vco2_ck / 3

0000011：pll2_p_ck = vco2_ck / 4

...

1111111：pll2_p_ck = vco2_ck / 128

位 8:0 **PLL2N[8:0]**：适用于 PLL2VCO 的倍频系数 (Multiplication factor for PLL2VCO)

由软件置 1 和复位，用于控制 VCO 的倍频系数。

这些位只能在 PLL 禁止（PLL2ON = 0 且 PLL2RDY = 0）时写入。

0x003：PLL2N = 4

0x004：PLL2N = 5

0x005：PLL2N = 6

...

0x080：PLL2N = 129（复位后的默认值）

...

0x1FF：PLL2N = 512

其他值：保留

10.8.12 RCC PLL2 小数分频器寄存器 (RCC_PLL2FRACR)

RCC PLL2 fractional divider register

偏移地址：0x040

复位值：0x0000 0000

访问：无等待周期；按字和半字访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLL2FRACN[12:0]													Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			

位 31:16 保留，必须保持复位值。

位 15:3 **PLL2FRACN[12:0]**：PLL2 VCO 的倍频系数的小数部分 (fractional part of the multiplication factor for PLL2 VCO)

由软件置 1 和复位，用于控制 VCO 的倍频系数的小数部分。这些位可随时写入，进而对 PLL2 VCO 进行动态微调。

小心： 软件必须正确设置这些位，以确保 VCO 输出频率介于其有效频率范围内，即：
* 128 MHz 到 560 MHz（PLL2VCOSEL = 0）时
* 150 MHz 到 420 MHz（PLL2VCOSEL = 1）时
VCO 输出频率 = Fref2_ck x (PLL2N + (PLL2FRACN / 213))，其中
* PLL2N 介于 8 到 420 之间
* PLL2FRACN 可介于 0 到 213- 1 之间
* 输入频率 Fref2_ck 必须介于 1 MHz 到 16 MHz 之间。
如果执意要在已使能 PLL 的情况下实时更改 PLL2FRACN 值，应用程序必须按如下过程进行操作：
* 将位 PLL2FRACEN 设为 0
* 将新的小数值写入 PLL2FRACN
* 将位 PLL2FRACEN 置 1

位 2:0 保留，必须保持复位值。

10.8.13 RCC 时钟源中断使能寄存器 (RCC_CIER)

RCC clock source interrupt enable register

偏移地址：0x050

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL2RDYIE	PLL1RDYIE	HSI48RDYIE	HSERDYIE	HSIRDYIE	CSIRDYIE	LSERDYIE	LSIRDYIE
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7 **PLL2RDYIE**：PLL2 就绪中断使能 (PLL2 ready interrupt enable)
由软件置 1 和复位，用于使能/禁止由 PLL2 锁定引起的中断。
0：禁止 PLL2 锁定中断（复位后的默认值）
1：使能 PLL2 锁定中断

位 6 **PLL1RDYIE**：PLL1 就绪中断使能 (PLL1 ready interrupt enable)
由软件置 1 和复位，用于使能/禁止由 PLL1 锁定引起的中断。
0：禁止 PLL1 锁定中断（复位后的默认值）
1：使能 PLL1 锁定中断

位 5 **HSI48RDYIE：HSI48 就绪中断使能 (HSI48 ready interrupt enable)**
由软件置 1 和复位，用于使能/禁止由 HSI48 振荡器稳定所引起的中断。
0：禁止 HSI48 就绪中断（复位后的默认值）
1：使能 HSI48 就绪中断

位 4 **HSERDYIE**：HSE 就绪中断使能 (HSE ready interrupt enable)
由软件置 1 和复位，用于使能/禁止由 HSE 振荡器稳定所引起的中断。
0：禁止 HSE 就绪中断（复位后的默认值）
1：使能 HSE 就绪中断

位 3 **HSIRDYIE**：HSI 就绪中断使能 (HSI ready interrupt enable)
由软件置 1 和复位，用于使能/禁止由 HSI 振荡器稳定所引起的中断。
0：禁止 HSI 就绪中断（复位后的默认值）
1：使能 HSI 就绪中断

位 2 **CSIRDYIE**：CSI 就绪中断使能 (CSI ready interrupt enable)
由软件置 1 和复位，用于使能/禁止由 CSI 振荡器稳定所引起的中断。
0：禁止 CSI 就绪中断（复位后的默认值）
1：使能 CSI 就绪中断

位 1 **LSERDYIE**：LSE 就绪中断使能 (LSE ready interrupt enable)
由软件置 1 和复位，用于使能/禁止由 LSE 振荡器稳定所引起的中断。
0：禁止 LSE 就绪中断（复位后的默认值）
1：使能 LSE 就绪中断

位 0 **LSIRDYIE**：LSI 就绪中断使能 (LSI ready interrupt enable)
由软件置 1 和复位，用于使能/禁止由 LSI 振荡器稳定所引起的中断。
0：禁止 LSI 就绪中断（复位后的默认值）
1：使能 LSI 就绪中断

10.8.14 RCC 时钟源中断标志寄存器 (RCC_CIFR)

RCC clock source interrupt flag register

偏移地址：0x054

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	HSECSSF	Res.	Res.	PLL2RDYF	PLL1RDYF	HSI48RDYF	HSERDYF	HSIRDYF	CSIRDYF	LSERDYF	LSIRDYF
					r			r	r	r	r	r	r	r	r

位 31:11 保留，必须保持复位值。

位 10 **HSECSSF**：HSE 时钟安全系统中断标志 (HSE clock security system interrupt flag)
通过软件写入 HSECSSC 位进行复位。
如果 HSE 时钟出现故障，则由硬件置 1。
0：未因 HSE 时钟故障而引起时钟安全中断（复位后的默认值）
1：因 HSE 时钟故障而引起时钟安全中断

位 9:8 保留，必须保持复位值。

位 7 **PLL2RDYF**：PLL2 就绪中断标志 (PLL2 ready interrupt flag)
通过软件写入 PLL2RDYC 位进行复位。
当 PLL2 锁定并且 PLL2RDYIE 置 1 时由硬件置 1。
0：未因 PLL2 锁定而引起时钟就绪中断（复位后的默认值）
1：因 PLL2 锁定而引起时钟就绪中断

位 6 **PLL1RDYF**：PLL1 就绪中断标志 (PLL1 ready interrupt flag)
通过软件写入 PLL1RDYC 位进行复位。
当 PLL1 锁定并且 PLL1RDYIE 置 1 时由硬件置 1。
0：未因 PLL1 锁定而引起时钟就绪中断（复位后的默认值）
1：因 PLL1 锁定而引起时钟就绪中断

位 5 **HSI48RDYF**：HSI48 就绪中断标志 (HSI48 ready interrupt flag)
通过软件写入 HSI48RDYC 位进行复位。
当 HSI48 时钟稳定且 HSI48RDYIE 置 1 时由硬件置 1。
0：未因 HSI48 振荡器而引起时钟就绪中断（复位后的默认值）
1：因 HSI48 振荡器引起时钟就绪中断

位 4 **HSERDYF**：HSE 就绪中断标志 (HSE ready interrupt flag)
通过软件写入 HSERDYC 位进行复位。
当 HSE 时钟稳定且 HSERDYIE 置 1 时由硬件置 1。
0：未因 HSE 而引起时钟就绪中断（复位后的默认值）
1：因 HSE 引起时钟就绪中断

位 3 **HSIRDYF**：HSI 就绪中断标志 (HSI ready interrupt flag)
通过软件写入 HSIRDYC 位进行复位。
当 HSI 时钟稳定且 HSIRDYIE 置 1 时由硬件置 1。
0：未因 HSI 而引起时钟就绪中断（复位后的默认值）
1：因 HSI 引起时钟就绪中断

位 2 **CSIRDYF**：CSI 就绪中断标志 (CSI ready interrupt flag)
通过软件写入 CSIRDYC 位进行复位。
当 CSI 时钟稳定且 CSIRDYIE 置 1 时由硬件置 1。
0：未因 CSI 而引起时钟就绪中断（复位后的默认值）
1：因 CSI 引起时钟就绪中断

位 1 LSERDYF：LSE 就绪中断标志 (LSE ready interrupt flag)
通过软件写入 LSERDYC 位进行复位。
当 LSE 时钟稳定且 LSERDYIE 置 1 时由硬件置 1。
0：未因 LSE 而引起时钟就绪中断（复位后的默认值）
1：因 LSE 引起时钟就绪中断

位 0 **LSIRDYF**：LSI 就绪中断标志 (LSI ready interrupt flag)
通过软件写入 LSIRDYC 位进行复位。
当 LSI 时钟稳定且 LSIRDYIE 置 1 时由硬件置 1。
0：未因 LSI 而引起时钟就绪中断（复位后的默认值）
1：因 LSI 引起时钟就绪中断

10.8.15 RCC 时钟源中断清零寄存器 (RCC_CICR)

RCC clock source interrupt clear register

偏移地址：0x058

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	HSECSSC	Res.	Res.	PLL2RDYC	PLL1RDYC	HSI48RDYC	HSERDYC	HSIRDYC	CSIRDYC	LSERDYC	LSIRDYC
					rc_w1			rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

位 31:11 保留，必须保持复位值。

位 10 **HSECSSC**：HSE 时钟安全系统中断清零 (HSE clock security system interrupt clear)
由软件置 1，用于将 HSECSSF 清零。
完成清零后由硬件复位。
0：对 HSECSSF 无影响（复位后的默认值）
1：将 HSECSSF 清零

位 9:8 保留，必须保持复位值。

位 7 **PLL2RDYC**：PLL2 就绪中断清零 (PLL2 ready interrupt clear)
由软件置 1，用于将 PLL2RDYF 清零。
完成清零后由硬件复位。
0：对 PLL2RDYF 无影响（复位后的默认值）
1：将 PLL2RDYF 清零

位 6 **PLL1RDYC**：PLL1 就绪中断清零 (PLL1 ready interrupt clear)
由软件置 1，用于将 PLL1RDYF 清零。
完成清零后由硬件复位。
0：对 PLL1RDYF 无影响（复位后的默认值）
1：将 PLL1RDYF 清零

位 5 **HSI48RDYC**：HSI48 就绪中断清零 (HSI48 ready interrupt clear)
由软件置 1，用于将 HSI48RDYF 清零。
完成清零后由硬件复位。
0：对 HSI48RDYF 无影响（复位后的默认值）
1：将 HSI48RDYF 清零

位 4 **HSERDYC**：HSE 就绪中断清零 (HSE ready interrupt clear)
由软件置 1，用于将 HSERDYF 清零。
完成清零后由硬件复位。
0：对 HSERDYF 无影响（复位后的默认值）
1：将 HSERDYF 清零

位 3 **HSIRDYC**：HSI 就绪中断清零 (HSI ready interrupt clear)
由软件置 1，用于将 HSIRDYF 清零。
完成清零后由硬件复位。
0：对 HSIRDYF 无影响（复位后的默认值）
1：将 HSIRDYF 清零

位 2 **CSIRDYC**：HSI 就绪中断清零 (HSI ready interrupt clear)
由软件置 1，用于将 CSIRDYF 清零。
完成清零后由硬件复位。
0：对 CSIRDYF 无影响（复位后的默认值）
1：将 CSIRDYF 清零

位 1 **LSERDYC**：LSE 就绪中断清零 (LSE ready interrupt clear)
由软件置 1，用于将 LSERDYF 清零。
完成清零后由硬件复位。
0：对 LSERDYF 无影响（复位后的默认值）
1：将 LSERDYF 清零

位 0 **LSIRDYC**：LSI 就绪中断清零 (LSI ready interrupt clear)
由软件置 1，用于将 LSIRDYF 清零。
完成清零后由硬件复位。
0：对 LSIRDYF 无影响（复位后的默认值）
1：将 LSIRDYF 清零

10.8.16 RCC AHB1 复位寄存器 (RCC_AHB1RSTR)

RCC AHB1 reset register

偏移地址：0x060

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RAMCFGRST	Res.
														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRCRST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPDMA2RST	GPDMA1RST
			rw											rw	rw

位 31:18 保留，必须保持复位值。

位 17 **RAMCFGRST**：RAMCFG 模块复位 (RAMCFG block reset)
由软件置 1 和复位。
0：不复位 RAMCFG 模块（复位后的默认值）
1：复位 RAMCFG 模块

位 16:13 保留，必须保持复位值。

位 12 **CRCRST**：CRC 模块复位 (CRC block reset)。由软件置 1 和复位。
0：不复位 CRC 模块（复位后的默认值）
1：复位 CRC 模块

位 11:2 保留，必须保持复位值。

位 1 **GPDMA2RST**：GPDMA2 模块复位 (GPDMA2 block reset)
由软件置 1 和复位。
0：不复位 GPDMA2 模块（复位后的默认值）
1：复位 GPDMA2 模块

位 0 **GPDMA1RST**：GPDMA1 模块复位 (GPDMA1 block reset)
由软件置 1 和复位。
0：不复位 GPDMA1 模块（复位后的默认值）
1：复位 GPDMA1 模块

10.8.17 RCC AHB2 外设复位寄存器 (RCC_AHB2RSTR)

RCC AHB2 peripheral reset register

偏移地址：0x064

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNGRST	HASHRST	Res.
													rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DAC1RST	ADCRST	Res.	Res.	GPIOHRST	Res.	Res.	Res.	GPIODRST	GPIOCRST	GPIOBRST	GPIOARST
				rw	rw			rw				rw	rw	rw	rw

位 31:19 保留，必须保持复位值。

位 18 **RNGRST**：RNG 模块复位 (RNG block reset)
由软件置 1 和复位。
0：不复位 RNG 模块（复位后的默认值）
1：复位 RNG 模块

位 17 **HASHRST**：HASH 模块复位 (HASH block reset)
由软件置 1 和复位。
0：不复位 HASH 模块（复位后的默认值）
1：复位 HASH 模块

位 16:12 保留，必须保持复位值。

位 11 **DAC1RST**：DAC 模块复位 (DAC block reset)
由软件置 1 和复位。
0：不复位 DAC 模块（复位后的默认值）
1：复位 DAC 模块

位 10 **ADCRST**：ADC1 模块复位 (ADC1 block reset)
由软件置 1 和复位。
0：不复位 ADC1 模块（复位后的默认值）
1：复位 ADC1 模块

位 9:8 保留，必须保持复位值。

位 7 **GPIOHRST**：GPIOH 模块复位 (GPIOH block reset)
由软件置 1 和复位。
0：不复位 GPIOH 模块（复位后的默认值）
1：复位 GPIOH 模块

位 6:4 保留，必须保持复位值。

位 3 **GPIODRST**：GPIOD 模块复位 (GPIOD block reset)
由软件置 1 和复位。
0：不复位 GPIOD 模块（复位后的默认值）
1：复位 GPIOD 模块

位 2 **GPIOCRST**：GPIOC 模块复位 (GPIOC block reset)
由软件置 1 和复位。
0：不复位 GPIOC 模块（复位后的默认值）
1：复位 GPIOC 模块

位 1 **GPIOBRST**：GPIOB 模块复位 (GPIOB block reset)
由软件置 1 和复位。
0：不复位 GPIOB 模块（复位后的默认值）
1：复位 GPIOB 模块

位 0 **GPIOARST**：GPIOA 模块复位 (GPIOA block reset)
由软件置 1 和复位。
0：不复位 GPIOA 模块（复位后的默认值）
1：复位 GPIOA 模块

10.8.18 RCC APB1 外设低位复位寄存器 (RCC_APB1LRSTR)

RCC APB1 peripheral low reset register

偏移地址：0x074

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRSRST	I3C1RST	I2C2RST	I2C1RST	Res.	Res.	USART3RST	USART2RST	COMPRST
							rw	rw	rw	rw			rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3RST	SPI2RST	OPAMPRST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TIM7RST	TIM6RST	Res.	Res.	TIM3RST	TIM2RST
rw	rw	rw								rw	rw			rw	rw

位 31:25 保留，必须保持复位值。

位 24 **CRSRST**：CRS 模块复位 (CRS block reset)
由软件置 1 和复位。
0：不复位 CRS 模块（复位后的默认值）
1：复位 CRS 模块

位 23 **I3C1RST**：I3C1 模块复位 (I3C1 block reset)
由软件置 1 和复位。
0：不复位 I3C1 模块（复位后的默认值）
1：复位 I3C1 模块

位 22 **I2C2RST**：I2C2 模块复位 (I2C2 block reset)
由软件置 1 和复位。
0：不复位 I2C2 模块（复位后的默认值）
1：复位 I2C2 模块

位 21 **I2C1RST**：I2C1 模块复位 (I2C1 block reset)
由软件置 1 和复位。
0：不复位 I2C1 模块（复位后的默认值）
1：复位 I2C1 模块

位 20:19 保留，必须保持复位值。

位 18 **USART3RST**：USART3 模块复位 (USART3 block reset)
由软件置 1 和复位。
0：不复位 USART3 模块（复位后的默认值）
1：复位 USART3 模块

位 17 **USART2RST**：USART2 模块复位 (USART2 block reset)
由软件置 1 和复位。
0：不复位 USART2 模块（复位后的默认值）
1：复位 USART2 模块

位 16 **COMPRST**：COMP 模块复位 (COMP block reset)
由软件置 1 和复位。
0：不复位 COMP 模块（复位后的默认值）
1：复位 COMP 模块

位 15 **SPI3RST**：SPI3 模块复位 (SPI3 block reset)
由软件置 1 和复位。
0：不复位 SPI3 模块（复位后的默认值）
1：复位 SPI3 模块

位 14 **SPI2RST**：SPI2 模块复位 (SPI2 block reset)
由软件置 1 和复位。
0：不复位 SPI2 模块（复位后的默认值）
1：复位 SPI2 模块

位 13 **OPAMPRST**：OPAMP 模块复位 (OPAMP block reset)
由软件置 1 和复位。
0：不复位 OPAMP 模块（复位后的默认值）
1：复位 OPAMP 模块

位 12:6 保留，必须保持复位值。

位 5 **TIM7RST**：TIM7 模块复位 (TIM7 block reset)
由软件置 1 和复位。
0：不复位 TIM7 模块（复位后的默认值）
1：复位 TIM7 模块

位 4 **TIM6RST**：TIM6 模块复位 (TIM6 block reset)
由软件置 1 和复位。
0：不复位 TIM6 模块（复位后的默认值）
1：复位 TIM6 模块

位 3:2 保留，必须保持复位值。

位 1 **TIM3RST**：TIM3 模块复位 (TIM3 block reset)
由软件置 1 和复位。
0：不复位 TIM3 模块（复位后的默认值）
1：复位 TIM3 模块

位 0 **TIM2RST**：TIM2 模块复位 (TIM2 block reset)
由软件置 1 和复位。
0：不复位 TIM2 模块（复位后的默认值）
1：复位 TIM2 模块

10.8.19 RCC APB1 外设高位复位寄存器 (RCC_APB1HRSTR)

RCC APB1 peripheral high reset register

偏移地址：0x078

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	FDCANRST	Res.	Res.	Res.	LPTIM2RST	Res.	DTSRST	Res.	Res.	Res.
						rw				rw		rw			

位 31:10 保留，必须保持复位值。

位 9 **FDCANRST**：FDCAN1 模块复位 (FDCAN1 block reset)
由软件置 1 和复位。
0：不复位 FDCAN1 模块（复位后的默认值）
1：复位 FDCAN1 模块

位 8:6 保留，必须保持复位值。

位 5 **LPTIM2RST**：LPTIM2 模块复位 (LPTIM2 block reset)
由软件置 1 和复位。
0：不复位 LPTIM2 模块（复位后的默认值）
1：复位 LPTIM2 模块

位 4 保留，必须保持复位值。

位 3 **DTSRST**：DTS 模块复位 (DTS block reset)
由软件置 1 和复位。
0：不复位 DTS 模块（复位后的默认值）
1：复位 DTS 模块

位 2:0 保留，必须保持复位值。

10.8.20 RCC APB2 外设复位寄存器 (RCC_APB2RSTR)

RCC APB2 peripheral reset register

偏移地址：0x07C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	USBRST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
							rw								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	USART1RST	Res.	SPI1RST	TIM1RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	rw		rw	rw											

位 31:25 保留，必须保持复位值。

位 24 **USBRST**：USB 模块复位 (USB block reset)
由软件置 1 和复位。
0：不复位 USB 模块（复位后的默认值）
1：复位 USB 模块

位 23:15 保留，必须保持复位值。

位 14 **USART1RST**：USART1 模块复位 (USART1 block reset)
由软件置 1 和复位。
0：不复位 USART1 模块（复位后的默认值）
1：复位 USART1 模块

位 13 保留，必须保持复位值。

位 12 **SPI1RST**：SPI1 模块复位 (SPI1 block reset)
由软件置 1 和复位。
0：不复位 SPI1 模块（复位后的默认值）
1：复位 SPI1 模块

位 11 **TIM1RST**：TIM1 模块复位 (TIM1 block reset)
由软件置 1 和复位。
0：不复位 TIM1 模块（复位后的默认值）
1：复位 TIM1 模块

位 10:0 保留，必须保持复位值。

10.8.21 RCC APB3 外设复位寄存器 (RCC_APB3RSTR)

RCC APB3 peripheral reset register

偏移地址：0x080

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	LPTIM1RST	Res.	I3C2RST	Res.	Res.	LPUART1RST	Res.	Res.	Res.	Res.	Res.	Res.
				rw		rw			rw						

位 31:12 保留，必须保持复位值。

位 11 **LPTIM1RST**：LPTIM1 模块复位 (LPTIM1 block reset)
由软件置 1 和复位。
0：不复位 LPTIM1 模块（复位后的默认值）
1：复位 LPTIM1 模块

位 10 保留，必须保持复位值。

位 9 **I3C2RST**：I3C2RST 模块复位 (I3C2RST block reset)
由软件置 1 和复位。
0：不复位 I3C2RST 模块（复位后的默认值）
1：复位 I3C2RST 模块

位 8:7 保留，必须保持复位值。

位 6 **LPUART1RST**：LPUART1 模块复位 (LPUART1 block reset)
由软件置 1 和复位。
0：不复位 LPUART1 模块（复位后的默认值）
1：复位 LPUART1 模块

位 5:0 保留，必须保持复位值。

10.8.22 RCC AHB1 外设时钟寄存器 (RCC_AHB1ENR)

RCC AHB1 peripherals clock register

偏移地址：0x088

复位值：0x9000 0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRAM1EN	Res.	Res.	BKPRAMEN	Res.	Res.	Res.	GTZC1EN	Res.	Res.	Res.	Res.	Res.	Res.	RAMCFGEN	Res.
rw			rw				rw							rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRCEN	Res.	Res.	Res.	FLITFEN	Res.	Res.	Res.	Res.	Res.	Res.	GPDMA2EN	GPDMA1EN
			rw				rw							rw	rw

位 31 **SRAM1EN**：SRAM1 时钟使能 (SRAM1 clock enable)
由软件置 1 和复位。
0：禁止 SRAM1 时钟
1：使能 SRAM1 时钟（复位后的默认值）

位 30:29 保留，必须保持复位值。

位 28 **BKPRAMEN**：BKPRAM 时钟使能 (BKPRAM clock enable)
由软件置 1 和复位
0：禁止 BKPRAM 外设时钟（复位后的默认值）
1：使能 BKPRAM 外设时钟

位 27:25 保留，必须保持复位值。

位 24 **GTZC1EN**：GTZC1 时钟使能 (GTZC1 clock enable)
由软件置 1 和复位
0：禁止 GTZC1 外设时钟（复位后的默认值）
1：使能 GTZC1 外设时钟

位 23:18 保留，必须保持复位值。

位 17 **RAMCFGEN**：RAMCFG 时钟使能 (RAMCFG clock enable)
由软件置 1 和复位。
0：禁止 RAMCFG 外设时钟（复位后的默认值）
1：使能 RAMCFG 外设时钟

位 16:13 保留，必须保持复位值。

位 12 **CRCEN**：CRC 时钟使能 (CRC clock enable)
由软件置 1 和复位。
0：禁止 CRC 外设时钟（复位后的默认值）
1：使能 CRC 外设时钟

位 11:9 保留，必须保持复位值。

位 8 **FLITFEN**：Flash 接口时钟使能 (Flash interface clock enable)
由软件置 1 和复位。
0：禁止 FLASH 接口时钟
1：使能 FLASH 接口时钟（复位后的默认值）

位 7:2 保留，必须保持复位值。

位 1 **GPDMA2EN**：GPDMA2 时钟使能 (GPDMA2 clock enable)
由软件置 1 和复位。
0：禁止 GPDMA2 外设时钟（复位后的默认值）
1：使能 GPDMA2 外设时钟

位 0 **GPDMA1EN**：GPDMA1 时钟使能 (GPDMA1 clock enable)
由软件置 1 和复位。
0：禁止 GPDMA1 外设时钟（复位后的默认值）
1：使能 GPDMA1 外设时钟

10.8.23 RCC AHB2 外设时钟寄存器 (RCC_AHB2ENR)

RCC AHB2 peripheral clock register

偏移地址：0x08C

复位值：0x4000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	SRAM2EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNGEN	HASHEN	Res.
	rw												rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DAC12EN	ADCEN	Res.	Res.	GPIOHEN	Res.	Res.	Res.	GPIODEN	GPIOCEN	GPIOBEN	GPIOAEN
				rw	rw			rw				rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30 **SRAM2EN**：SRAM2 时钟使能 (SRAM2 clock enable)
由软件置 1 和复位。
0：禁止 SRAM2 时钟
1：使能 SRAM2 时钟（复位后的默认值）

位 29:19 保留，必须保持复位值。

位 18 **RNGEN**：RNG 时钟使能 (RNG clock enable)
由软件置 1 和复位。
0：禁止 RNG 外设时钟（复位后的默认值）
1：使能 RNG 外设时钟

位 17 **HASHEN**：HASH 时钟使能 (HASH clock enable)
由软件置 1 和复位。
0：禁止 HASH 外设时钟（复位后的默认值）
1：使能 HASH 外设时钟

位 16:12 保留，必须保持复位值。

位 11 **DAC12EN**：DAC 时钟使能 (DAC clock enable)
由软件置 1 和复位。
0：禁止 DAC 外设时钟（复位后的默认值）
1：使能 DAC 外设时钟

位 10 **ADCEN**：ADC1 外设时钟使能 (ADC1 peripherals clock enabled)
由软件置 1 和复位。
0：禁止 ADC1 外设时钟（复位后的默认值）
1：使能 ADC1 外设时钟

位 9:8 保留，必须保持复位值。

位 7 **GPIOHEN**：GPIOH 时钟使能 (GPIOH clock enable)
由软件置 1 和复位。
0：禁止 GPIOH 外设时钟（复位后的默认值）
1：使能 GPIOH 外设时钟

位 6:4 保留，必须保持复位值。

位 3 **GPIODEN**：GPIOD 时钟使能 (GPIOD clock enable)
由软件置 1 和复位。
0：禁止 GPIOD 外设时钟（复位后的默认值）
1：使能 GPIOD 外设时钟

位 2 **GPIOCEN**：GPIOC 时钟使能 (GPIOC clock enable)
由软件置 1 和复位。
0：禁止 GPIOC 外设时钟（复位后的默认值）
1：使能 GPIOC 外设时钟

位 1 **GPIOBEN**：GPIOB 时钟使能 (GPIOB clock enable)
由软件置 1 和复位。
0：禁止 GPIOB 外设时钟（复位后的默认值）
1：使能 GPIOB 外设时钟

位 0 **GPIOAEN**：GPIOA 时钟使能 (GPIOA clock enable)
由软件置 1 和复位。
0：禁止 GPIOA 外设时钟（复位后的默认值）
1：使能 GPIOA 外设时钟

10.8.24 RCC APB1 外设时钟寄存器 (RCC_APB1LENR)

RCC APB1 peripheral clock register

偏移地址：0x09C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRSEN	I3C1EN	I2C2EN	I2C1EN	Res.	Res.	USART3EN	USART2EN	COMPEN
							rw	rw	rw	rw			rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3EN	SPI2EN	OPAMPEN	Res.	WWDGEN	Res.	Res.	Res.	Res.	Res.	TIM7EN	TIM6EN	Res.	Res.	TIM3EN	TIM2EN
rw	rw	rw		rw						rw	rw			rw	rw

位 31:25 保留，必须保持复位值。

位 24 **CRSEN**：CRS 时钟使能 (CRS clock enable)
由软件置 1 和复位。
0：禁止 CRS 外设时钟（复位后的默认值）
1：使能 CRS 外设时钟

位 23 **I3C1EN**：I3C1 时钟使能 (I3C1 clock enable)
由软件置 1 和复位。
0：禁止 I3C1 外设时钟（复位后的默认值）
1：使能 I3C1 外设时钟

位 22 **I2C2EN**：I2C2 时钟使能 (I2C2 clock enable)
由软件置 1 和复位。
0：禁止 I2C2 外设时钟（复位后的默认值）
1：使能 I2C2 外设时钟

位 21 **I2C1EN**：I2C1 时钟使能 (I2C1 clock enable)
由软件置 1 和复位。
0：禁止 I2C1 外设时钟（复位后的默认值）
1：使能 I2C1 外设时钟

位 20:19 保留，必须保持复位值。

位 18 **USART3EN**：USART3 时钟使能 (USART3 clock enable)
由软件置 1 和复位。
0：禁止 USART3 外设时钟（复位后的默认值）
1：使能 USART3 外设时钟

位 17 **USART2EN**：USART2 时钟使能 (USART2 clock enable)
由软件置 1 和复位。
0：禁止 USART2 外设时钟（复位后的默认值）
1：使能 USART2 外设时钟

位 16 **COMPEN**：COMP 时钟使能 (COMP clock enable)
由软件置 1 和复位。
0：禁止 COMP 外设时钟（复位后的默认值）
1：使能 COMP 外设时钟

位 15 **SPI3EN**：SPI3 时钟使能 (SPI3 clock enable)
由软件置 1 和复位。
0：禁止 SPI3 外设时钟（复位后的默认值）
1：使能 SPI3 外设时钟

位 14 **SPI2EN**：SPI2 时钟使能 (SPI2 clock enable)
由软件置 1 和复位。
0：禁止 SPI2 外设时钟（复位后的默认值）
1：使能 SPI2 外设时钟

位 13 **OPAMPEN**：OPAMP 时钟使能 (OPAMP clock enable)
由软件置 1 和复位。
0：禁止 OPAMP 外设时钟（复位后的默认值）
1：使能 OPAMP 外设时钟

位 12 保留，必须保持复位值。

位 11 **WWDGEN**：WWDG 时钟使能 (WWDG clock enable)
由软件置 1 和复位。
0：禁止 WWDG 外设时钟（复位后的默认值）
1：使能 WWDG 外设时钟

位 10:6 保留，必须保持复位值。

位 5 **TIM7EN**：TIM7 时钟使能 (TIM7 clock enable)
由软件置 1 和复位。
0：禁止 TIM7 外设时钟（复位后的默认值）
1：使能 TIM7 外设时钟

位 4 **TIM6EN**：TIM6 时钟使能 (TIM6 clock enable)
由软件置 1 和复位。
0：禁止 TIM6 外设时钟（复位后的默认值）
1：使能 TIM6 外设时钟

位 3:2 保留，必须保持复位值。

位 1 **TIM3EN**：TIM3 时钟使能 (TIM3 clock enable)
由软件置 1 和复位。
0：禁止 TIM3 外设时钟（复位后的默认值）
1：使能 TIM3 外设时钟

位 0 **TIM2EN**：TIM2 时钟使能 (TIM2 clock enable)
由软件置 1 和复位。
0：禁止 TIM2 外设时钟（复位后的默认值）
1：使能 TIM2 外设时钟

10.8.25 RCC APB1 外设时钟寄存器 (RCC_APB1HENR)

RCC APB1 peripheral clock register

偏移地址：0x0A0

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	FDCANEN	Res.	Res.	Res.	LPTIM2EN	Res.	DTSEN	Res.	Res.	Res.
						rw				rw		rw			

位 31:10 保留，必须保持复位值。

位 9 **FDCANEN**：FDCAN1 外设时钟使能 (FDCAN1 peripheral clock enable)
由软件置 1 和复位。
0：禁止 FDCAN1 外设时钟（复位后的默认值）
1：使能 FDCAN1 外设时钟

位 8:6 保留，必须保持复位值。

位 5 **LPTIM2EN**：LPTIM2 时钟使能 (LPTIM2 clock enable)
由软件置 1 和复位。
0：禁止 LPTIM2 外设时钟（复位后的默认值）
1：使能 LPTIM2 外设时钟

位 4 保留，必须保持复位值。

位 3 **DTSEN**：DTS 时钟使能 (DTS clock enable)
由软件置 1 和复位。
0：禁止 DTS 外设时钟（复位后的默认值）
1：使能 DTS 外设时钟

位 2:0 保留，必须保持复位值。

10.8.26 RCC APB2 外设时钟寄存器 (RCC_APB2ENR)

RCC APB2 peripheral clock register

偏移地址：0x0A4

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	USBEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
							rw								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	USART1EN	Res.	SPI1EN	TIM1EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	rw		rw	rw											

位 31:25 保留，必须保持复位值。

位 24 **USBEN**：USB 时钟使能 (USB clock enable)
由软件置 1 和复位。
0：禁止 USB 外设时钟（复位后的默认值）
1：使能 USB 外设时钟

位 23:15 保留，必须保持复位值。

位 14 **USART1EN**：USART1 时钟使能 (USART1 clock enable)
由软件置 1 和复位。
0：禁止 USART1 外设时钟（复位后的默认值）
1：使能 USART1 外设时钟

位 13 保留，必须保持复位值。

位 12 **SPI1EN**：SPI1 时钟使能 (SPI1 clock enable)
由软件置 1 和复位。
0：禁止 SPI1 外设时钟（复位后的默认值）
1：使能 SPI1 外设时钟

位 11 **TIM1EN**：TIM1 时钟使能 (TIM1 clock enable)
由软件置 1 和复位。
0：禁止 TIM1 外设时钟（复位后的默认值）
1：使能 TIM1 外设时钟

位 10:0 保留，必须保持复位值。

10.8.27 RCC APB3 外设时钟寄存器 (RCC_APB3ENR)

RCC APB3 peripheral clock register

偏移地址：0x0A8

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTCAPBEN	Res.	Res.	Res.	Res.	Res.
										rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	LPTIM1EN	Res.	I3C2EN	Res.	Res.	LPUART1EN	Res.	Res.	Res.	Res.	SBSEN	Res.
				rw		rw			rw					rw	

位 31:22 保留，必须保持复位值。

位 21 **RTCAPBEN**：RTC APB 接口时钟使能 (RTC APB interface clock enable)
由软件置 1 和复位。
0：禁止 RTC APB 接口时钟（复位后的默认值）
1：使能 RTC APB 接口时钟

位 20:12 保留，必须保持复位值。

位 11 **LPTIM1EN**：LPTIM1 时钟使能 (LPTIM1 clock enable)
由软件置 1 和复位。
0：禁止 LPTIM1 外设时钟（复位后的默认值）
1：使能 LPTIM1 外设时钟

位 10 保留，必须保持复位值。

位 9 **I3C2EN**：I3C2EN 时钟使能 (I3C2EN clock enable)
由软件置 1 和复位。
0：禁止 I3C2EN 外设时钟（复位后的默认值）
1：使能 I3C2EN 外设时钟

位 8:7 保留，必须保持复位值。

位 6 **LPUART1EN**：LPUART1 时钟使能 (LPUART1 clock enable)
由软件置 1 和复位。
0：禁止 LPUART1 外设时钟（复位后的默认值）
1：使能 LPUART1 外设时钟

位 5:2 保留，必须保持复位值。

位 1 **SBSEN**：SBS 时钟使能 (SBS clock enable)
由软件置 1 和复位。
0：禁止 SBS 外设时钟（复位后的默认值）
1：使能 SBS 外设时钟

位 0 保留，必须保持复位值。

10.8.28 RCC AHB1 睡眠时钟寄存器 (RCC_AHB1LPENR)

RCC AHB1 sleep clock register

偏移地址：0x0B0

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRAM1LPEN	Res.	ICACHELPEN	BKPRAMLPEN	Res.	Res.	Res.	GTZC1LPEN	Res.	Res.	Res.	Res.	Res.	Res.	RAMCFGLPEN	Res.
rw		rw	rw				rw							rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRCLPEN	Res.	Res.	Res.	FLITFLPEN	Res.	Res.	Res.	Res.	Res.	Res.	GPDMA2LPEN	GPDMA1LPEN
			rw				rw							rw	rw

位 31 **SRAM1LPEN**：睡眠模式期间的 SRAM1 时钟使能 (SRAM1 clock enable during sleep mode)
由软件置 1 和复位
0：睡眠模式期间禁止 SRAM1 外设时钟
1：睡眠模式期间使能 SRAM1 外设时钟（复位后的默认值）

位 30 保留，必须保持复位值。

位 29 **ICACHELPEN**：睡眠模式期间的 ICACHE 时钟使能 (ICACHE clock enable during sleep mode)
由软件置 1 和复位
0：睡眠模式期间禁止 ICACHE 外设时钟
1：睡眠模式期间使能 ICACHE 外设时钟（复位后的默认值）

位 28 **BKPRAMLPEN**：睡眠模式期间的 BKPRAM 时钟使能 (BKPRAM clock enable during sleep mode)
由软件置 1 和复位
0：睡眠模式期间禁止 BKPRAM 外设时钟
1：睡眠模式期间使能 BKPRAM 外设时钟（复位后的默认值）

位 27:25 保留，必须保持复位值。

位 24 **GTZC1LPEN**：睡眠模式期间的 GTZC1 时钟使能 (GTZC1 clock enable during sleep mode)
由软件置 1 和复位
0：睡眠模式期间禁止 GTZC1 外设时钟
1：睡眠模式期间使能 GTZC1 外设时钟（复位后的默认值）

位 23:18 保留，必须保持复位值。

位 17 **RAMCFGLPEN**：睡眠模式期间的 RAMCFG 时钟使能 (RAMCFG clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 RAMCFG 外设时钟
1：睡眠模式期间使能 RAMCFG 外设时钟（复位后的默认值）

位 16:13 保留，必须保持复位值。

位 12 **CRCLPEN**：睡眠模式期间的 CRC 时钟使能 (CRC clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 CRC 外设时钟
1：睡眠模式期间使能 CRC 外设时钟（复位后的默认值）

位 11:9 保留，必须保持复位值。

位 8 **FLITFLPEN**：睡眠模式期间的 Flash 接口 (FLITF) 时钟使能 (Flash interface (FLITF) clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 FLITF 外设时钟
1：睡眠模式期间使能 FLITF 外设时钟（复位后的默认值）

位 7:2 保留，必须保持复位值。

位 1 **GPDMA2LPEN**：睡眠模式期间的 GPDMA2 时钟使能 (GPDMA2 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 GPDMA2 外设时钟
1：睡眠模式期间使能 GPDMA2 外设时钟（复位后的默认值）

位 0 **GPDMA1LPEN**：睡眠模式期间的 GPDMA1 时钟使能 (GPDMA1 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 GPDMA1 外设时钟
1：睡眠模式期间使能 GPDMA1 外设时钟（复位后的默认值）

10.8.29 RCC AHB2 睡眠时钟寄存器 (RCC_AHB2LPENR)

RCC AHB2 sleep clock register

偏移地址：0x0B4

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	SRAM2LPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNGLPEN	HASHLPEN	Res.
	rw												rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DAC12LPEN	ADCLPEN	Res.	Res.	GPIOHLPEN	Res.	Res.	Res.	GPIODLPEN	GPIOCLPEN	GPIOBLPEN	GPIOALPEN
				rw	rw			rw				rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30 **SRAM2LPEN**：睡眠模式期间的 SRAM2 时钟使能 (SRAM2 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 SRAM2 外设时钟
1：睡眠模式期间使能 SRAM2 外设时钟（复位后的默认值）

位 29:19 保留，必须保持复位值。

位 18 **RNGLPEN**：睡眠模式期间的 RNG 时钟使能 (RNG clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 RNG 外设时钟
1：睡眠模式期间使能 RNG 外设时钟（复位后的默认值）

位 17 **HASHLPEN**：睡眠模式期间的 HASH 时钟使能 (HASH clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 HASH 外设时钟
1：睡眠模式期间使能 HASH 外设时钟（复位后的默认值）

位 16:12 保留，必须保持复位值。

位 11 **DAC12LPEN**：睡眠模式期间的 DAC 时钟使能 (DAC clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 DAC 外设时钟
1：睡眠模式期间使能 DAC 外设时钟（复位后的默认值）

位 10 **ADCLPEN**：睡眠模式期间的 ADC1 外设时钟使能 (ADC1 peripherals clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 ADC1 外设时钟
1：睡眠模式期间使能 ADC1 外设时钟（复位后的默认值）

位 9:8 保留，必须保持复位值。

位 7 **GPIOHLPEN**：睡眠模式期间的 GPIOH 时钟使能 (GPIOH clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 GPIOH 外设时钟
1：睡眠模式期间使能 GPIOH 外设时钟（复位后的默认值）

位 6:4 保留，必须保持复位值。

位 3 **GPIODLPEN**：睡眠模式期间的 GPIOD 时钟使能 (GPIOD clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 GPIOD 外设时钟
1：睡眠模式期间使能 GPIOD 外设时钟（复位后的默认值）

位 2 **GPIOCLPEN**：睡眠模式期间的 GPIOC 时钟使能 (GPIOC clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 GPIOC 外设时钟
1：睡眠模式期间使能 GPIOC 外设时钟（复位后的默认值）

位 1 **GPIOBLPEN**：睡眠模式期间的 GPIOB 时钟使能 (GPIOB clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 GPIOB 外设时钟
1：睡眠模式期间使能 GPIOB 外设时钟（复位后的默认值）

位 0 **GPIOALPEN**：睡眠模式期间的 GPIOA 时钟使能 (GPIOA clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 GPIOA 外设时钟
1：睡眠模式期间使能 GPIOA 外设时钟（复位后的默认值）

10.8.30 RCC APB1 睡眠时钟寄存器 (RCC_APB1LLPENR)

RCC APB1 sleep clock register

偏移地址：0x0C4

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRSLPEN	I3C1LPEN	I2C2LPEN	I2C1LPEN	Res.	Res.	USART3LPEN	USART2LPEN	COMPLPEN
							rw	rw	rw	rw			rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3LPEN	SPI2LPEN	OPAMPLPEN	Res.	WWDGLPEN	Res.	Res.	Res.	Res.	Res.	TIM7LPEN	TIM6LPEN	Res.	Res.	TIM3LPEN	TIM2LPEN
rw	rw	rw		rw						rw	rw			rw	rw

位 31:25 保留，必须保持复位值。

位 24 **CRSLPEN**：睡眠模式期间的 CRS 时钟使能 (CRS clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 CRS 外设时钟
1：睡眠模式期间使能 CRS 外设时钟（复位后的默认值）

位 23 **I3C1LPEN**：睡眠模式期间的 I3C1 时钟使能 (I3C1 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 I3C1 外设时钟
1：睡眠模式期间使能 I3C1 外设时钟（复位后的默认值）

位 22 **I2C2LPEN**：睡眠模式期间的 I2C2 时钟使能 (I2C2 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 I2C2 外设时钟
1：睡眠模式期间使能 I2C2 外设时钟（复位后的默认值）

位 21 **I2C1LPEN**：睡眠模式期间的 I2C1 时钟使能 (I2C1 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 I2C1 外设时钟
1：睡眠模式期间使能 I2C1 外设时钟（复位后的默认值）

位 20:19 保留，必须保持复位值。

位 18 **USART3LPEN**：睡眠模式期间的 USART3 时钟使能 (USART3 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 USART3 外设时钟
1：睡眠模式期间使能 USART3 外设时钟（复位后的默认值）

位 17 **USART2LPEN**：睡眠模式期间的 USART2 时钟使能 (USART2 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 USART2 外设时钟
1：睡眠模式期间使能 USART2 外设时钟（复位后的默认值）

位 16 **COMPLPEN**：睡眠模式期间的 COMP 时钟使能 (COMP clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 COMP 外设时钟
1：睡眠模式期间使能 COMP 外设时钟（复位后的默认值）

位 15 **SPI3LPEN**：睡眠模式期间的 SPI3 时钟使能 (SPI3 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 SPI3 外设时钟
1：睡眠模式期间使能 SPI3 外设时钟（复位后的默认值）

位 14 **SPI2LPEN**：睡眠模式期间的 SPI2 时钟使能 (SPI2 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 SPI2 外设时钟
1：睡眠模式期间使能 SPI2 外设时钟（复位后的默认值）

位 13 **OPAMPLPEN**：睡眠模式期间的 OPAMP 时钟使能 (OPAMP clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 OPAMP 外设时钟
1：睡眠模式期间使能 OPAMP 外设时钟（复位后的默认值）

位 12 保留，必须保持复位值。

位 11 **WWDGLPEN**：睡眠模式期间的 WWDG 时钟使能 (WWDG clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 WWDG 外设时钟
1：睡眠模式期间使能 WWDG 外设时钟（复位后的默认值）

位 10:6 保留，必须保持复位值。

位 5 **TIM7LPEN**：睡眠模式期间的 TIM7 时钟使能 (TIM7 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 TIM7 外设时钟
1：睡眠模式期间使能 TIM7 外设时钟（复位后的默认值）

位 4 **TIM6LPEN**：睡眠模式期间的 TIM6 时钟使能 (TIM6 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 TIM6 外设时钟
1：睡眠模式期间使能 TIM6 外设时钟（复位后的默认值）

位 3:2 保留，必须保持复位值。

位 1 **TIM3LPEN**：睡眠模式期间的 TIM3 时钟使能 (TIM3 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 TIM3 外设时钟
1：睡眠模式期间使能 TIM3 外设时钟（复位后的默认值）

位 0 **TIM2LPEN**：睡眠模式期间的 TIM2 时钟使能 (TIM2 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 TIM2 外设时钟
1：睡眠模式期间使能 TIM2 外设时钟（复位后的默认值）

10.8.31 RCC APB1 睡眠时钟寄存器 (RCC_APB1HLPENR)

RCC APB1 sleep clock register

偏移地址：0x0C8

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	FDCANLPEN	Res.	Res.	Res.	LPTIM2LPEN	Res.	DTSLPEN	Res.	Res.	Res.
						rw				rw		rw			

位 31:10 保留，必须保持复位值。

位 9 **FDCANLPEN**：睡眠模式期间的 FDCAN1 外设时钟使能 (FDCAN1 peripheral clock enable during sleep mode)

由软件置 1 和复位。

0：睡眠模式期间禁止 FDCAN1 外设时钟

1：睡眠模式期间使能 FDCAN1 外设时钟（复位后的默认值）

位 8:6 保留，必须保持复位值。

位 5 **LPTIM2LPEN**：睡眠模式期间的 LPTIM2 时钟使能 (LPTIM2 clock enable during sleep mode)

由软件置 1 和复位。

0：睡眠模式期间禁止 LPTIM2 外设时钟

1：睡眠模式期间使能 LPTIM2 外设时钟（复位后的默认值）

位 4 保留，必须保持复位值。

位 3 **DTSLPEN**：睡眠模式期间的 DTS 时钟使能 (DTS clock enable during sleep mode)

由软件置 1 和复位。

0：睡眠模式期间禁止 DTS 外设时钟

1：睡眠模式期间使能 DTS 外设时钟（复位后的默认值）

位 2:0 保留，必须保持复位值。

10.8.32 RCC APB2 睡眠时钟寄存器 (RCC_APB2LPENR)

RCC APB2 sleep clock register

偏移地址：0x0CC

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	USBLPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
							rw								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	USART1LPEN	Res.	SPI1LPEN	TIM1LPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	rw		rw	rw											

位 31:25 保留，必须保持复位值。

位 24 **USBLPEN**：睡眠模式期间的 USB 时钟使能 (USB clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 USB 外设时钟
1：睡眠模式期间使能 USB 外设时钟（复位后的默认值）

位 23:15 保留，必须保持复位值。

位 14 **USART1LPEN**：睡眠模式期间的 USART1 时钟使能 (USART1 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 USART1 外设时钟
1：睡眠模式期间使能 USART1 外设时钟（复位后的默认值）

位 13 保留，必须保持复位值。

位 12 **SPI1LPEN**：睡眠模式期间的 SPI1 时钟使能 (SPI1 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 SPI1 外设时钟
1：睡眠模式期间使能 SPI1 外设时钟（复位后的默认值）

位 11 **TIM1LPEN**：睡眠模式期间的 TIM1 时钟使能 (TIM1 clock enable during sleep mode)
由软件置 1 和复位。
0：睡眠模式期间禁止 TIM1 外设时钟
1：睡眠模式期间使能 TIM1 外设时钟（复位后的默认值）

位 10:0 保留，必须保持复位值。

10.8.33 RCC APB3 睡眠时钟寄存器 (RCC_APB3LPENR)

RCC APB3 sleep clock register

偏移地址：0x0D0

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTCAPBLPEN	Res.	Res.	Res.	Res.	Res.
										rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	LPTIM1LPEN	Res.	I3C2LPEN	Res.	Res.	LPUART1LPEN	Res.	Res.	Res.	Res.	SBSLPEN	Res.
				rw		rw			rw					rw	

位 31:22 保留，必须保持复位值。

位 21 **RTCAPBLPEN**：睡眠模式期间的 RTC APB 接口时钟使能 (RTC APB interface clock enable during sleep mode)

由软件置 1 和复位。
0：睡眠模式期间禁止 RTC APB 接口时钟
1：睡眠模式期间使能 RTC APB 接口时钟（复位后的默认值）

位 20:12 保留，必须保持复位值。

位 11 **LPTIM1LPEN**：睡眠模式期间的 LPTIM1 时钟使能 (LPTIM1 clock enable during sleep mode)

由软件置 1 和复位。
0：睡眠模式期间禁止 LPTIM1 外设时钟
1：睡眠模式期间使能 LPTIM1 外设时钟（复位后的默认值）

位 10 保留，必须保持复位值。

位 9 **I3C2LPEN**：睡眠模式期间的 I3C2 时钟使能 (I3C2 clock enable during sleep mode)

由软件置 1 和复位。
0：睡眠模式期间禁止 I3C2 外设时钟
1：睡眠模式期间使能 I3C2 外设时钟（复位后的默认值）

位 8:7 保留，必须保持复位值。

位 6 LPUART1LPEN：睡眠模式期间的 LPUART1 时钟使能 (LPUART1 clock enable during sleep mode)

由软件置 1 和复位。
0：睡眠模式期间禁止 LPUART1 外设时钟
1：睡眠模式期间使能 LPUART1 外设时钟（复位后的默认值）

位 5:2 保留，必须保持复位值。

位 1 **SBSLPEN**：睡眠模式期间的 SBS 时钟使能 (SBS clock enable during sleep mode)

由软件置 1 和复位。
0：睡眠模式期间禁止 SBS 外设时钟
1：睡眠模式期间使能 SBS 外设时钟（复位后的默认值）

位 0 保留，必须保持复位值。

10.8.34 RCC 内核时钟配置寄存器 (RCC_CCIPR1)

RCC kernel clock configuration register

偏移地址：0x0D8

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMICSEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	USART3SEL[2:0]			USART2SEL[2:0]			USART1SEL[2:0]		
							rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **TIMICSEL**：TIM2、TIM3 和 LPTIM2 输入捕获源选择 (TIM2, TIM3 and LPTIM2 input capture source selection)

由软件置 1 和复位。

0：没有可用于定时器输入捕获的内部时钟（复位后的默认值）

1：选择 hsi_ker_ck/1024、hsi_ker_ck/8 和 csi_ker_ck/128 用于定时器输入捕获

位 30:9 保留，必须保持复位值。

位 8:6 **USART3SEL[2:0]**：USART3 内核时钟源选择 (USART3 kernel clock source selection)

由软件置 1 和复位。

000：将 rcc_pclk1 选作内核时钟（复位后的默认值）

001：将 pll2_q_ck 选作内核时钟

011：将 hsi_ker_ck 选作内核时钟

100：将 csi_ker_ck 选作内核时钟

101：将 lse_ck 选作内核时钟

其他：保留，禁止内核时钟

位 5:3 **USART2SEL[2:0]**：USART2 内核时钟源选择 (USART2 kernel clock source selection)

由软件置 1 和复位。

000：将 rcc_pclk1 选作内核时钟（复位后的默认值）

001：将 pll2_q_ck 选作内核时钟

011：将 hsi_ker_ck 选作内核时钟

100：将 csi_ker_ck 选作内核时钟

101：将 lse_ck 选作内核时钟

其他：保留，禁止内核时钟

位 2:0 **USART1SEL[2:0]**：USART1 内核时钟源选择 (USART1 kernel clock source selection)
由软件置 1 和复位。
000：将 rcc_pclk2 选作内核时钟（复位后的默认值）
001：将 pll2_q_ck 选作内核时钟
011：将 hsi_ker_ck 选作内核时钟
100：将 csi_ker_ck 选作内核时钟
101：将 lse_ck 选作内核时钟
其他：保留，禁止内核时钟

10.8.35 RCC 内核时钟配置寄存器 (RCC_CCIPR2)

RCC kernel clock configuration register

偏移地址：0x0DC

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	LPTIM2SEL[2:0]			Res.	LPTIM1SEL[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	rw	rw	rw		rw	rw	rw								

位 31:15 保留，必须保持复位值。

位 14:12 **LPTIM2SEL[2:0]**：LPTIM2 内核时钟源选择 (LPTIM2 kernel clock source selection)
000：将 rcc_pclk1 选作内核时钟（复位后的默认值）
001：将 pll2_p_ck 选作内核时钟
011：将 lse_ker_ck 选作内核时钟
100：将 lsi_ker_ck 选作内核时钟
101：将 per_ck 选作内核时钟
其他：保留，禁止内核时钟

位 11 保留，必须保持复位值。

位 10:8 **LPTIM1SEL[2:0]**：LPTIM1 内核时钟源选择 (LPTIM1 kernel clock source selection)
000：将 rcc_pclk3 选作内核时钟（复位后的默认值）
001：将 pll2_p_ck 选作内核时钟
011：将 lse_ker_ck 选作内核时钟
100：将 lsi_ker_ck 选作内核时钟
101：将 per_ck 选作内核时钟
其他：保留，禁止内核时钟

位 7:0 保留，必须保持复位值。

10.8.36 RCC 内核时钟配置寄存器 (RCC_CCIPR3)

RCC kernel clock configuration register

偏移地址：0x0E0
复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	LPUART1SEL[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
					rw	rw	rw								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPI3SEL[2:0]			SPI2SEL[2:0]			SPI1SEL[2:0]		
							rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:27 保留，必须保持复位值。

位 26:24 **LPUART1SEL[2:0]**：LPUART1 内核时钟源选择 (LPUART1 kernel clock source selection)
000：将 rcc_pclk3 选作内核时钟（复位后的默认值）
001：将 pll2_q_ck 选作内核时钟
011：将 hsi_ker_ck 选作内核时钟
100：将 csi_ker_ck 选作内核时钟
101：将 lse_ck 选作内核时钟
其他：保留，禁止内核时钟

位 23:9 保留，必须保持复位值。

位 8:6 **SPI3SEL[2:0]**：SPI3 内核时钟源选择 (SPI3 kernel clock source selection)
由软件置 1 和复位。
000：将 pll1_q_ck 选作内核时钟（复位后的默认值）
001：将 pll2_p_ck 选作内核时钟
011：将 AUDIOCLK 选作内核时钟
100：将 per_ck 选作内核时钟
其他：保留，禁止内核时钟

位 5:3 **SPI2SEL[2:0]**：SPI2 内核时钟源选择 (SPI2 kernel clock source selection)
由软件置 1 和复位。
000：将 pll1_q_ck 选作内核时钟（复位后的默认值）
001：将 pll2_p_ck 选作内核时钟
011：将 AUDIOCLK 选作内核时钟
100：将 per_ck 选作内核时钟
其他：保留，禁止内核时钟

位 2:0 **SPI1SEL[2:0]**：SPI1 内核时钟源选择 (SPI1 kernel clock source selection)
由软件置 1 和复位。
000：将 pll1_q_ck 选作内核时钟（复位后的默认值）
001：将 pll2_p_ck 选作内核时钟
011：将 AUDIOCLK 选作内核时钟
100：将 per_ck 选作内核时钟
其他：保留，禁止内核时钟

10.8.37 RCC 内核时钟配置寄存器 (RCC_CCIPR4)

RCC kernel clock configuration register

偏移地址：0x0E4

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	I3C2SEL[1:0]		I3C1SEL[1:0]		Res.	Res.	Res.	Res.	I2C2SEL[1:0]		I2C1SEL[1:0]	
				rw	rw	rw	rw					rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USBSEL[1:0]		SYSTICKSEL[1:0]		Res.	Res.
										rw	rw	rw	rw		

位 31:28 保留，必须保持复位值。

位 27:26 **I3C2SEL[1:0]**：I3C2 内核时钟源选择 (I3C2 kernel clock source selection)
- 00：将 rcc_pclk3 选作内核时钟（复位后的默认值）
- 01：将 pll2_r_ck 选作内核时钟
- 10：将 hsi_ker_ck 选作内核时钟
- 11：未选择时钟

位 25:24 **I3C1SEL[1:0]**：I3C1 内核时钟源选择 (I3C1 kernel clock source selection)
- 00：将 rcc_pclk1 选作内核时钟（复位后的默认值）
- 01：将 pll2_r_ck 选作内核时钟
- 10：将 hsi_ker_ck 选作内核时钟
- 11：未选择时钟

位 23:20 保留，必须保持复位值。

位 19:18 **I2C2SEL[1:0]**：I2C2 内核时钟源选择 (I2C2 kernel clock source selection)
- 00：将 rcc_pclk1 选作内核时钟（复位后的默认值）
- 01：将 pll2_r_ck 选作内核时钟
- 10：将 hsi_ker_ck 选作内核时钟
- 11：将 csi_ker_ck 选作内核时钟

位 17:16 **I2C1SEL[1:0]**：I2C1 内核时钟源选择 (I2C1 kernel clock source selection)
- 00：将 rcc_pclk1 选作内核时钟（复位后的默认值）
- 01：将 pll2_r_ck 选作内核时钟
- 10：将 hsi_ker_ck 选作内核时钟
- 11：将 csi_ker_ck 选作内核时钟

位 15:6 保留，必须保持复位值。

位 5:4 **USBSEL[1:0]**：USB 内核时钟源选择 (USB kernel clock source selection)
- 00：未选择任何时钟作为内核时钟（复位后的默认值）
- 01：将 pll1_q_ck 选作内核时钟
- 10：将 pll2_q_ck 选作内核时钟
- 11：将 hsi48_ker_ck 选作内核时钟

位 3:2 **SYSTICKSEL[1:0]**：SYSTICK 时钟源选择 (SYSTICK clock source selection)
00：将 rcc_hclk/8 选作时钟源（复位后的默认值）
01：将 lsi_ker_ck[1] 选作时钟源
10：将 lse_ck[1] 选作时钟源
11：保留，禁止内核时钟
*注意：rcc_hclk 频率必须是lsi_ker_ck/lse_ck 的四倍（周期 (LSI/LSE) ≥ 4 * 周期 (HCLK)）。*

位 1:0 保留，必须保持复位值。

10.8.38 RCC 内核时钟配置寄存器 (RCC_CCIPR5)

RCC kernel clock configuration register

偏移地址：0x0E8

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CKPERSEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
rw	rw														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	FDCANSEL [1:0]		Res.	Res.	RNGSEL[1:0]		DAC1SEL	ADCDACSEL[2:0]		
						rw	rw			rw	rw	rw	rw	rw	rw

位 31:30 **CKPERSEL[1:0]**：per_ck 时钟源选择 (per_ck clock source selection)
00：将 hsi_ker_ck 选作内核时钟（复位后的默认值）
01：将 csi_ker_ck 选作内核时钟
10：将 hse_ck 选作内核时钟
11：保留，禁止 per_ck 时钟

位 29:10 保留，必须保持复位值。

位 9:8 **FDCANSEL[1:0]**：FDCAN1 内核时钟源选择 (FDCAN1 kernel clock source selection)
00：将 hse_ck 选作内核时钟（复位后的默认值）
01：将 pll1_q_ck 选作内核时钟
10：将 pll2_q_ck 选作内核时钟
11：保留，禁止内核时钟

位 7:6 保留，必须保持复位值。

位 5:4 **RNGSEL[1:0]**：RNG 内核时钟源选择 (RNG kernel clock source selection)
00：将 hsi48_ker_ck 选作内核时钟（复位后的默认值）
01：将 pll1_q_ck 选作内核时钟
10：将 lse_ck 选作内核时钟
11：将 lsi_ker_ck 选作内核时钟

位 3 **DAC1SEL**：DAC1 采样和保持时钟源选择 (DAC1 sample and hold clock source selection)
该位用于选择 DAC1 采样和保持时钟源 (dac_hold_ck)。
0：选择 LSE
1：选择 LSI

位 2:0 **ADCDACSEL[2:0]**：ADC 和 DAC 内核时钟源选择 (ADC and DAC kernel clock source selection)
000：将 rcc_hclk 选作内核时钟（复位后的默认值）
001：将 sys_ck 选作内核时钟
010：将 pll2_r_ck 选作内核时钟
011：将 hse_ck 选作内核时钟
100：将 hsi_ker_ck 选作内核时钟
101：将 csi_ker_ck 选作内核时钟
其他：保留，禁止内核时钟

10.8.39 RCC 备份域控制寄存器 (RCC_BDCR)

RCC backup domain control register

偏移地址：0x0F0

复位值：0x0000 0000

通过备份域复位实现复位。

访问：0 ≤ 等待周期 ≤ 3；按字、半字和字节访问。

连续访问该寄存器时，则插入等待周期。

注意： *系统复位后，RCC_BDCR 寄存器受写保护（位 27:26 除外）。要修改备份域位，PWR_DBPCR 寄存器中的 DBP 位必须置 1。只有备份域复位后，RCC_BDCR 位（位 27:24 和位 16 除外）才能复位（请参见第 10.3.3 节：备份域复位）。其他复位对这些位不会有任何影响。*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	LSIRDY	LSION	LSCOSEL	LSCOEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VSWRST
				r	rw	rw	rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCEN	Res.	Res.	Res.	Res.	Res.	RTCSEL[1:0]		LSEEXT	LSECSSD	LSECSSON	LSEDRV[1:0]		LSEBYP	LSERDY	LSEON
rw						rw	rw	rw	r	rw	rw	rw	rw	r	rw

位 31:28 保留，必须保持复位值。

位 27 **LSIRDY**：LSI 振荡器就绪 (LSI oscillator ready)
由硬件置 1 和清零，用于指示 LSI 振荡器已稳定。
在 LSION 位清零后，LSIRDY 将在三个内部低速振荡器时钟周期后转为低电平。
当 LSI 由 IWDG 或 RTC 使用时，即使 LSION = 0，该位也将置 1。
0：LSI 振荡器未就绪
1：LSI 振荡器就绪

位 26 **LSION**：LSI 振荡器使能 (LSI oscillator enable)
由软件置 1 和清零。
0：LSI 振荡器关闭
1：LSI 振荡器开启

位 25 **LSCOSEL**：低速时钟输出选择 (Low-speed clock output selection)
由软件置 1 和清零。
0：选择 LSI 时钟
1：选择 LSE 时钟

位 24 **LSCOEN**：低速时钟输出 (LSCO) 使能 (Low-speed clock output (LSCO) enable)
由软件置 1 和清零。
0：禁止 LSCO 输出
1：使能 LSCO 输出

位 23:17 保留，必须保持复位值。

位 16 **VSWRST**：VSwitch 域软件复位 (VSwitch domain software reset)
由软件置 1 和复位。
0：复位未激活（备份域复位后的默认值）
1：复位整个 VSW 域

位 15 **RTCEN**：RTC 时钟使能 (RTC clock enable)
由软件置 1 和复位。
0：禁止 rtc_ck（备份域复位后的默认值）
1：使能 rtc_ck

位 14:10 保留，必须保持复位值。

位 9:8 **RTCSEL[1:0]**：RTC 时钟源选择 (RTC clock source selection)
由软件置 1，用于选择 RTC 的时钟源。
这些位只能写入一次（在 LSE 上检测到故障的情况除外）。
这些位必须在使能 LSECSSON 之前写入。
VSWRST 位可用于复位这些位，然后该位可再次写入。
如果将 HSE 选作 RTC 时钟，在系统处于停止模式或在发生引脚复位 (NRST) 时，该时钟将丢失。
00：无时钟（备份域复位后的默认值）
01：将 LSE 选作 RTC 时钟
10：将 LSI 选作 RTC 时钟
11：将以 RTCPRE 值进行分频的 HSE 时钟选作 RTC 时钟

位 7 **LSEEXT**：旁路模式下的低速外部时钟类型 (low-speed external clock type in bypass mode)
由软件置 1 和复位，用于选择外部时钟的类型（模拟或数字）。
外部时钟必须通过 LSEON 位使能才能为器件使用。
LSEEXT 位只有在 LSE 振荡器已禁止的情况下才可写入。
0：LSE 处于模拟模式（备份域复位后的默认值）
1：LSE 处于数字模式（如果 RTC 处于工作状态，则不使用）

位 6 **LSECSSD**：LSE 时钟安全系统故障检测 (LSE clock security system failure detection)
由硬件置 1，用于指示外部 32 kHz 振荡器上的时钟安全系统何时检测到故障。
0：未在 32 kHz 振荡器上检测到故障（备份域复位后的默认值）
1：在 32 kHz 振荡器上检测到故障

位 5 **LSECSSON**：LSE 时钟安全系统使能 (LSE clock security system enable)
由软件置 1，用于使能 32 kHz 振荡器上的时钟安全系统。
在 LSE 使能（已使能 LSEON）和就绪（由硬件将 LSERDY 置 1）后以及在选择 RTCSEL 后，LSECSSON 必须使能。
该位一旦使能便不能禁止，但检测到 LSE 故障 (LSECSSD = 1) 后除外。此时，软件必须禁止 LSECSSON。
0：32 kHz 振荡器上的 CSS 关闭（备份域复位后的默认值）
1：32 kHz 振荡器上的 CSS 开启

位 4:3 **LSEDRV[1:0]**：LSE 振荡器驱动能力 (LSE oscillator driving capability)

由软件置 1，用于选择 LSE 振荡器的驱动能力。

这些位只有在 LSE 振荡器已禁止的情况下（LSEON = 0 且 LSERDY = 0）才可写入。

00：最低驱动（备份域复位后的默认值）

01：中低驱动

10：中高驱动

11：最高驱动

位 2 **LSEBYP**：LSE 振荡器旁路 (LSE oscillator bypass)

由软件置 1 和复位，用于在调试模式下旁路振荡器。在 LSE 使能（通过 LSEON）或就绪 (LSERDY = 1) 时，不得写入该位

0：不旁路 LSE 振荡器（备份域复位后的默认值）

1：旁路 LSE 振荡器

位 1 **LSERDY**：LSE 振荡器就绪 (LSE oscillator ready)

由硬件置 1 和复位，用于指示 LSE 已稳定。

将 LSEON 设为 0 后，该位需要经过 6 个 lse_ck 时钟周期才能变为低电平。

0：LSE 振荡器未就绪（备份域复位后的默认值）

1：LSE 振荡器就绪

位 0 **LSEON**：LSE 振荡器使能 (LSE oscillator enabled)

由软件置 1 和复位。

0：LSE 振荡器关闭（备份域复位后的默认值）

1：LSE 振荡器开启

10.8.40 RCC 复位状态寄存器 (RCC_RSR)

RCC reset status register

偏移地址：0x0F4

复位值：0x0C00 0000

仅通过上电复位实现复位

访问：0 ≤ 等待周期 ≤ 3；按字、半字和字节访问。

连续访问该寄存器时，则插入等待周期。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	BORRSTF	PINRSTF	Res.	Res.	RMVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r	r	r	r	r			rw							

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

位 31 **LPWRRSTF**：低功耗复位标志 (Low-power reset flag)

当由于进入停止或待机模式而导致复位时，该位由硬件置 1，而相应的 nRST_STOP 或 nRST_STBY 选项位将清零。

通过写入 RMVF 位清零。

0：未发生非法低功耗模式复位

1：发生非法低功耗模式复位

位 30 **WWDGRSTF**：窗口看门狗复位标志 (window watchdog reset flag)
通过软件写入 RMVF 位进行复位。
发生窗口看门狗复位时，由硬件置 1。
0：WWDG 未发生窗口看门狗复位（上电复位后的默认值）
1：WWDG 发生窗口看门狗复位

位 29 **IWDGRSTF**：独立看门狗复位标志 (independent watchdog reset flag)
通过软件写入 RMVF 位进行复位。
发生独立看门狗复位时，由硬件置 1。
0：未发生独立看门狗复位（上电复位后的默认值）
1：发生独立看门狗复位

位 28 **SFTRSTF**：通过 CPU 实现系统复位的复位标志 (system reset from CPU reset flag)
通过软件写入 RMVF 位进行复位。
系统复位是因 CPU 引起时，由硬件置 1。CPU 可通过写入内核 M33 AIRCR 寄存器的 SYSRESETREQ 位来生成系统复位。
0：未发生 CPU 软件复位（上电复位后的默认值）
1：CPU 已生成系统复位

位 27 **BORRSTF**：BOR 复位标志 (BOR reset flag)
通过软件写入 RMVF 位进行复位。
发生 BOR 复位 (pwr_bor_rst) 时，由硬件置 1。
0：未发生 BOR 复位
1：发生 BOR 复位（上电复位后的默认值）

位 26 **PINRSTF**：引脚复位标志 (NRST) (pin reset flag (NRST))
通过软件写入 RMVF 位进行复位。
发生来自引脚的复位时，由硬件置 1。
0：未发生来自引脚的复位
1：发生来自引脚的复位（上电复位后的默认值）

位 25:24 保留，必须保持复位值。

位 23 **RMVF**：清除复位标志 (remove reset flag)
由软件置 1 和复位，用于对复位标志的值进行复位。
0：未激活对复位标志的复位（上电复位后的默认值）
1：对复位标志的值进行复位

位 22:0 保留，必须保持复位值。

10.8.41 RCC 特权配置寄存器 (RCC_PRIVCFGR)

RCC privilege configuration register

偏移地址：0x114

复位值：0x0000 0000

访问：无等待周期；按字、半字和字节访问

该寄存器只能通过特权访问进行写操作，但可以通过特权或非特权访问进行读操作。

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

位 31:2 保留，必须保持复位值。

位 1 **PRIV**：RCC 功能特权配置 (RCC functions privilege configuration)

由软件置 1 和复位。该位只能通过特权访问进行写操作。

0：可通过特权或非特权访问对 RCC 功能进行读写操作。

1：只能通过特权访问对 RCC 功能进行读写操作。

位 0 保留，必须保持复位值。

10.8.42 RCC 寄存器映射

表 70. RCC 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	RCC_CR	Res.	Res.	Res.	Res.	PLL2RDY	PLL2ON	PLL1RDY	PLL1ON	Res.	Res.	Res.	HSEEXT	HSECSSON	HSEBYP	HSERDY	HSEON	Res.	Res.	HSI48RDY	HSI48ON	Res.	CSIKERON	CSIRDY	CSION	Res.	Res.	HSIDIVF	HSIDIV[1:0]		HSIKERON	HSIRDY	HSION
	Reset value					0	0	0	0				0	0	0	0	0			0	0		0	0	0			1	0	1	0	1	1
0x010	RCC_HSICFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSITRIM[6:0]							Res.	Res.	Res.	Res.	HSICAL[11:0]											
	Reset value										1	0	0	0	0	0	0					X	X	X	X	X	X	X	X	X	X	X	X
0x014	RCC_CRRCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSI48CAL[9:0]									
	Reset value																							X	X	X	X	X	X	X	X	X	X
0x018	RCC_CSICFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSITRIM[5:0]						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSICAL[7:0]							
	Reset value											1	0	0	0	0	0									X	X	X	X	X	X	X	X

表 70. RCC 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x01C	RCC_CFGR1	MCO2SEL[2:0]			MCO2PRE[3:0]				MCO1SEL[2:0]			MCO1PRE[3:0]				Res.	Res.	TIMPRE	Res.	RTCPRE[5:0]						STOPKERWUCK	STOPWUCK	Res.	SWS[2:0]		Res.	SW[2:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0			0		0	0	0	0	0	0	0	0		0	0		0	0
0x020	RCC_CFGR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	APB3DIS	APB2DIS	APB1DIS	Res.	Res.	AHB2DIS	AHB1DIS	Res.	Res.	PPRE3[2:0]			Res.	PPRE2[2:0]		Res.	PPRE1[2:0]		HPRE[3:0]				
	Reset value										0	0	0			0	0		0	0	0		0	0	0		0	0	0	0	0	0	0	0
0x028	RCC_PLL1CFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL1REN	PLL1QEN	PLL1PEN	Res.	Res.	PLL1M[5:0]						Res.	Res.	PLL1VCOSEL	PLL1FRACEN	PLL1RGE[1:0]		PLL1SRC[1:0]	
	Reset value														0	0	0			0	0	0	0	0	0			0	0	0	0	0	0
0x02C	RCC_PLL2CFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL2REN	PLL2QEN	PLL2PEN	Res.	Res.	PLL2M[5:0]						Res.	Res.	PLL2VCOSEL	PLL2FRACEN	PLL2RGE[1:0]		PLL2SRC[1:0]	
	Reset value														0	0	0			0	0	0	0	0	0			0	0	0	0	0	0
0x034	RCC_PLL1DIVR	Res.	PLL1R[6:0]							Res.	PLL1Q[6:0]							PLL1P[6:0]								PLL1N[8:0]							
	Reset value		0	0	0	0	0	0	1		0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0
0x038	RCC_PLL1FRACR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL1FRACN[12:0]													Res.	Res.	Res.
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x03C	RCC_PLL2DIVR	Res.	PLL2R[6:0]							Res.	PLL2Q[6:0]							PLL2P[6:0]								PLL2N[8:0]							
	Reset value		0	0	0	0	0	0	1		0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0
0x040	RCC_PLL2FRACR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL2FRACN[12:0]													Res.	Res.	Res.
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x050	RCC_CIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL2RDYIE	PLL1RDYIE	HSI48RDYIE	HSERDYIE	HSIRDYIE	CSIRDYIE	LSERDYIE	LSIRDYIE
	Reset value																									0	0	0	0	0	0	0	0

表 70. RCC 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x054	RCC_CIFR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSECSSF	Res.	Res.	PLL2RDYF	PLL1RDYF	HSI48RDYF	HSERDYF	HSIRDYF	CSIRDYF	LSERDYF	LSIRDYF
	Reset value																						0			0	0	0	0	0	0	0	0
0x058	RCC_CICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSECSSC	Res.	Res.	PLL2RDYC	PLL1RDYC	HSI48RDYC	HSERDYC	HSIRDYC	CSIRDYC	LSERDYC	LSIRDYC
	Reset value																						0			0	0	0	0	0	0	0	0
0x060	RCC_AHB1RSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RAMCFGRST	Res.	Res.	Res.	Res.	CRCRST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPDMA2RST	GPDMA1RST
	Reset value															0					0											0	0
0x064	RCC_AHB2RSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNGRST	HASHRST	Res.	Res.	Res.	Res.	Res.	DAC1RST	ADCRST	Res.	Res.	GPIOHRST	Res.	Res.	Res.	GPIODRST	GPIOCRST	GPIOBRST	GPIOARST
	Reset value														0	0						0	0			0				0	0	0	0
0x074	RCC_ APB1LRSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRSRST	I3C1RST	I2C2RST	I2C1RST	Res.	Res.	USART3RST	USART2RST	COMPRST	SPI3RST	SPI2RST	OPAMPRST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TIM7RST	TIM6RST	Res.	Res.	TIM3RST	TIM2RST
	Reset value								0	0	0	0			0	0	0	0	0	0								0	0			0	0
0x078	RCC_ APB1HRSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FDCAN1RST	Res.	Res.	Res.	LPTIM2RST	Res.	DTSRST	Res.	Res.	Res.
	Reset value																							0				0		0			
0x07C	RCC_APB2RSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USBRST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USART1RST	Res.	SPI1RST	TIM1RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value								0										0		0	0												
0x080	RCC_APB3RSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPTIM1RST	Res.	I3C2RST	Res.	Res.	LPUART1RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																					0		0			0							
0x088	RCC_AHB1ENR	SRAM1EN	Res.	Res.	BKPRAMEN	Res.	Res.	Res.	GTZC1EN	Res.	Res.	Res.	Res.	Res.	Res.	RAMCFGEN	Res.	Res.	Res.	Res.	CRCEN	Res.	Res.	Res.	FLITFEN	Res.	Res.	Res.	Res.	Res.	Res.	GPDMA2EN	GPDMA1EN
	Reset value	1			1				0							0					0			0							0	0
0x08C	RCC_AHB2ENR	Res.	SRAM2EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNGEN	HASHEN	Res.	Res.	Res.	Res.	Res.	DAC12EN	ADCEN	Res.	Res.	GPIOHEN	Res.	Res.	Res.	GPIODEN	GPIOCEN	GPIOBEN	GPIOAEN
	Reset value		1												0	0						0	0			0				0	0	0	0

表 70. RCC 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x09C	RCC_APB1LENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRSEN	I3C1EN	I2C2EN	I2C1EN	Res.	Res.	USART3EN	USART2EN	COMPEN	SPI3EN	SPI2EN	OPAMPEN	Res.	WWDGEN	Res.	Res.	Res.	Res.	Res.	TIM7EN	TIM6EN	Res.	Res.	TIM3EN	TIM2EN
	Reset value								0	0	0	0			0	0	0	0	0	0		0						0	0			0	0
0x0A0	RCC_APB1HENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FDCANEN	Res.	Res.	Res.	LPTIM2EN	Res.	DTSEN	Res.	Res.	Res.
	Reset value																							0				0		0			
0x0A4	RCC_APB2ENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USBEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USART1EN	Res.	SPI1EN	TIM1EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value								0										0		0	0											
0x0A8	RCC_APB3ENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTCAPBEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPTIM1EN	Res.	I3C2EN	Res.	Res.	LPUART1EN	Res.	Res.	Res.	Res.	SBSEN	Res.
	Reset value											0										0		0			0					0	
0x0B0	RCC_ AHB1LPENR	SRAM1LPEN	Res.	ICACHELPEN	BKPRAMLPEN	Res.	Res.	Res.	GTZC1LPEN	Res.	Res.	Res.	Res.	Res.	Res.	RAMCFGLPEN	Res.	Res.	Res.	Res.	CRCLPEN	Res.	Res.	Res.	FLITFLPEN	Res.	Res.	Res.	Res.	Res.	GPDMA2LPEN	GPDMA1LPEN
	Reset value	1		1	1				1							1					1			1							1	1
0x0B4	RCC_ AHB2LPENR	Res.	SRAM2LPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNGLPEN	HASHLPEN	Res.	Res.	Res.	Res.	Res.	DAC12LPEN	ADCLPEN	Res.	Res.	GPIOHLPEN	Res.	Res.	Res.	GPIODLPEN	GPIOCLPEN	GPIOBLPEN	GPIOALPEN
	Reset value		1												1	1						1	1			1				1	1	1	1
0x0C4	RCC_ APB1LLPENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRSLPEN	I3C1LPEN	I2C2LPEN	I2C1LPEN	Res.	Res.	USART3LPEN	USART2LPEN	COMPLPEN	SPI3LPEN	SPI2LPEN	OPAMPLPEN	Res.	WWDGLPEN	Res.	Res.	TIM14LPEN	TIM13LPEN	TIM12LPEN	TIM7LPEN	TIM6LPEN	TIM5LPEN	TIM4LPEN	TIM3LPEN	TIM2LPEN
	Reset value								1	1	1	1			1	1	1	1	1	1		1			1	1	1	1	1	1	1	1	1
0x0C8	RCC_ APB1HLPENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FDCANLPEN	Res.	Res.	Res.	LPTIM2LPEN	Res.	DTSLPEN	Res.	Res.	Res.
	Reset value																							1				1		1			
0x0CC	RCC_ APB2LPENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USBLPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USART1LPEN	Res.	SPI1LPEN	TIM1LPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value								1										1		1	1											

表 70. RCC 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0D0	RCC_ APB3LPENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTCAPBLPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPTIM1LPEN	Res.	I3C2LPEN	Res.	Res.	LPUART1LPEN	Res.	Res.	Res.	Res.	SBSLPEN	Res.
	Reset value											1										1		1			1					1	
0x0D8	RCC_CCIPR1	TIMICSEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USART3SEL[2:0]			USART2SEL[2:0]			USART1SEL[2:0]		
	Reset value	0																							0	0	0	0	0	0	0	0	0
0x0DC	RCC_CCIPR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPTIM2SEL[2:0]			Res.	Res.	LPTIM1SEL[2:0]				Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																		0	0	0		0	0	0								
0x0E0	RCC_CCIPR3	Res.	Res.	Res.	Res.	Res.	LPUART1SEL[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPI3SEL[2:0]			SPI2SEL[2:0]			SPI1SEL[2:0]		
	Reset value						0	0	0							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0E4	RCC_CCIPR4	Res.	Res.	Res.	Res.	I3C2SEL[1:0]		I3C1SEL[1:0]		Res.	Res.	Res.	Res.	I2C2SEL[1:0]		I2C1SEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USBSEL[1:0]		SYSTICKSEL[1:0]		Res.	Res.
	Reset value					0	0	0	0					0	0	0	0											0	0	0	0		
0x0E8	RCC_CCIPR5	CKPERSEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FDCANSEL[1:0]		Res.	Res.	RNGSEL[1:0]		DAC1SEL		ADCDACSEL[2:0]	
	Reset value	0	0																					0	0			0	0	0	0	0	0
0x0F0	RCC_BDCR	Res.	Res.	Res.	Res.	LSIRDY	LSION	LSCOSEL	LSCOEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VSWRST	RTCEN	Res.	Res.	Res.	Res.	Res.	RTCSEL[1:0]		LSEEXT	LSECSSD	LSECSSON		LSEDRV[1:0]		LSEBYP	LSERDY	LSEON
	Reset value					0	0	0	0								0	0						0	0	0	0	0	0	0	0	0	0
0x0F4	RCC_RSR	LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	BORRSTF	PINRSTF	Res.	Res.	RMVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	1	1			0																							
0x0F8 to 0x10C	Reserved	Reserved																															

表 70. RCC 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x114	RCC_PRIVCFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIV	Res.
	Reset value																															0	

有关寄存器边界地址的信息，请参见*第 2.2 节*。

11 时钟恢复系统 (CRS)

11.1 简介

时钟恢复系统 (CRS) 是一个作用于内部精细粒度可调 RC 振荡器 HSI48 的高级数字控制器。CRS 提供一种十分强大的方法，可通过与可选同步信号进行比较来评估振荡器输出频率。CRS 能够根据测得的频率误差值自动进行微调，同时还可以进行手动微调。

CRS 非常适合为 USB 外设提供精密时钟。在这种情况下，同步信号可由 USB 总线上的帧起始 (SOF) 数据包信号提供，USB 主机以 1 ms 的时间间隔发送该信号。

同步信号也可由 LSE 振荡器输出提供，或者由用户软件生成。

11.2 CRS 主要特性

- 具有可编程预分频器和极性的可选同步源：
  - LSE 振荡器输出
  - USB SOF 数据包接收
- 可由软件生成同步脉冲
- 振荡器自动微调功能，无需 CPU 操作
- 手动控制选项，可加快启动融合
- 16 位频率误差计数器，用于自动误差值捕获和重载
- 可编程限值，用于自动频率误差值评估和状态报告
- 可屏蔽中断/事件：
  - 预期同步 (ESYNC)
  - 同步正常 (SYNCOK)
  - 同步警告 (SYNCWARN)
  - 同步或微调错误 (ERR)

11.3 CRS 实现

表 71. CRS 特性

特性	CRS1
TRIM 宽度	6 位

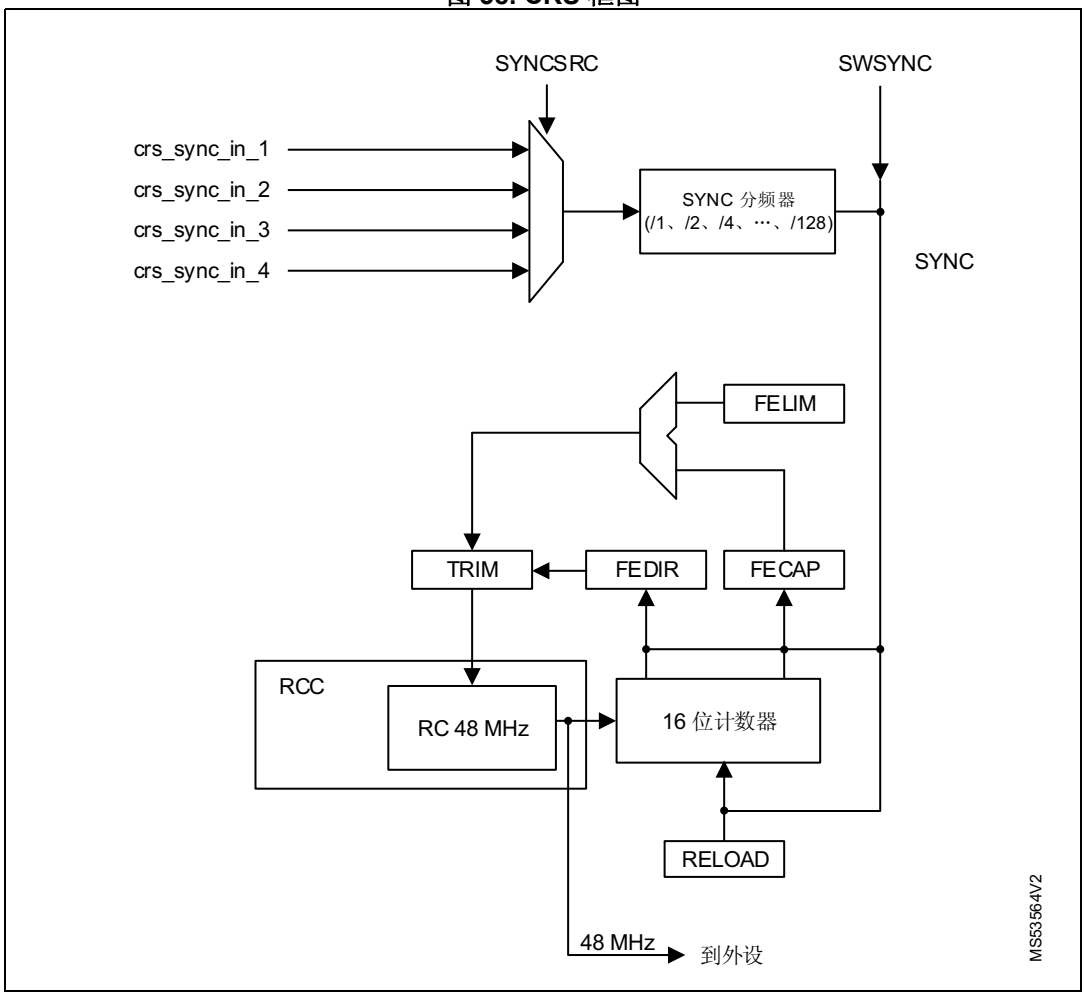
表 72. CRS 内部输入/输出信号

内部信号名称	信号类型	说明
crs_sync_in_1	输入	00：选择 GPIO AF 作为 SYNC 信号源
crs_sync_in_2	输入	01：选择 LSE 作为 SYNC 信号源
crs_sync_in_3	输入	10：选择 USB SOF 作为 SYNC 信号源（默认）
crs_sync_in_4	输入	11：保留

11.4 CRS 功能说明

11.4.1 CRS 框图

图 38. CRS 框图



11.4.2 同步输入

CRS 同步 (SYNC) 源可通过 CRS_CFGR 寄存器选择，它可以是来自 LSE 时钟的信号或 USB SOF 信号。为使 SYNC 输入更稳定，可使用一个简单的数字滤波器（三分之二多数表决，通过 HSI48 时钟采样）来滤除干扰。该信号源具有可配置极性，可由可编程二进制预分频器分频，以获得合理频率范围内的同步信号（通常为 1 kHz 左右）。

有关 CRS 同步源配置的更多信息，请参见*第 11.7.2 节*。

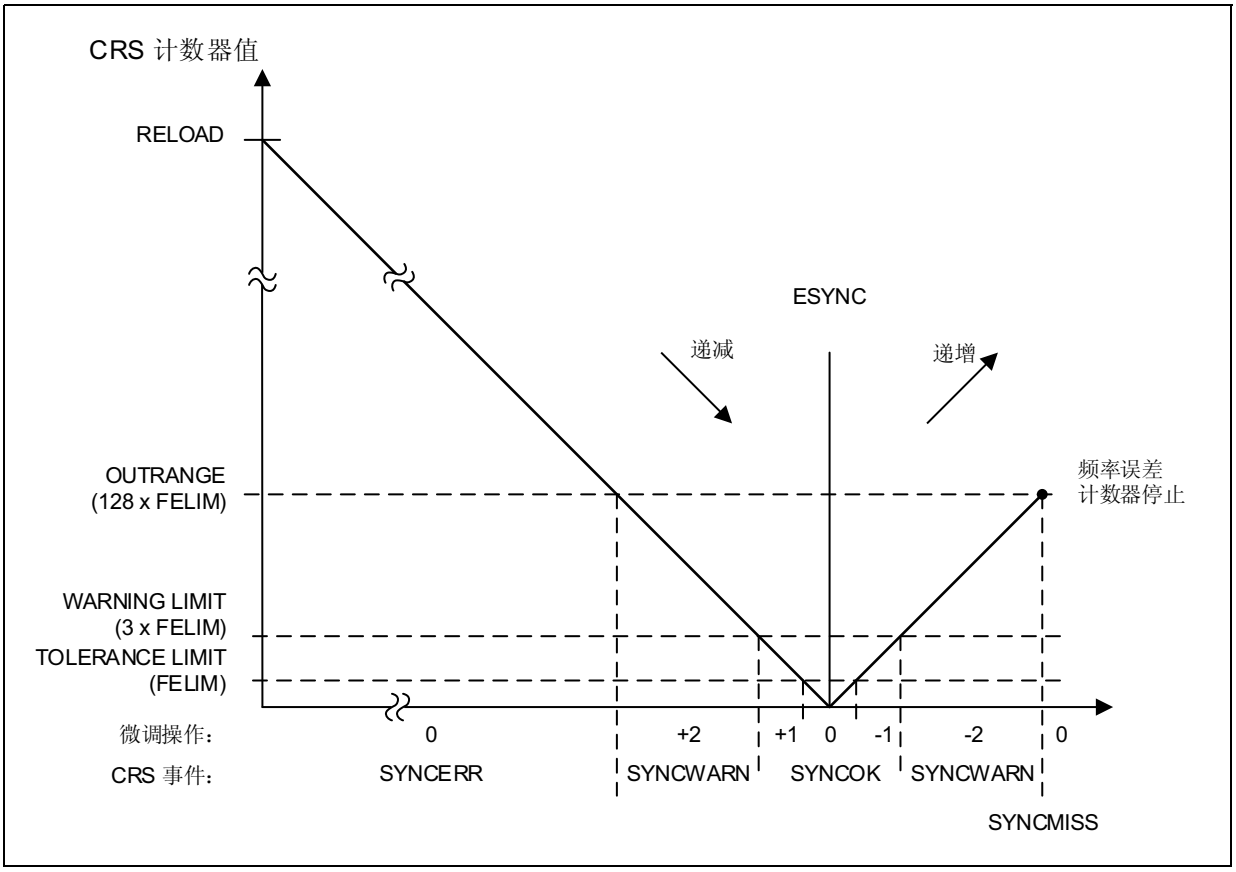
还可通过用软件将 CRS_CR 寄存器中的 SWSYNC 位置 1 的方法来生成同步事件。

11.4.3 频率误差测量

频率误差计数器是一个 16 位递减/递增计数器，会在每个 SYNC 事件发生时重载 RELOAD 值。它开始递减计数，直至达到 0 值，此时会生成 ESYNC（预期同步）事件。随后它将递增计数到 OUTRANGE 限值，这种情况下最终将停止计数（如果未接收到 SYNC 事件）并生成 SYNCMISS 事件。OUTRANGE 限值定义为频率误差限值（CRS_CFGR 寄存器的 FELIM 位域）乘以 128。

当检测到 SYNC 事件时，频率误差计数器的实际值及其方向存储在 CRS_ISR 寄存器的 FECAP（频率误差捕获）位域和 FEDIR（频率误差方向）位中。当在递减计数阶段（达到 0 值之前）检测到 SYNC 事件时，意味着实际频率小于目标频率（TRIM 值必须递增）。当在递增计数阶段检测到此事件时，意味着实际频率大于目标频率（TRIM 值必须递减）。

图 39. CRS 计数器行为



11.4.4 频率误差评估和自动微调

测得的频率误差通过将频率值与一组限值进行比较进行估算：

- TOLERANCE LIMIT，在 CRS_CFGR 寄存器的 FELIM 位域中直接给出
- WARNING LIMIT，定义为 3 × FELIM 值
- OUTRANGE（误差限值），定义为 128 × FELIM 值

该比较结果用于生成状态指示以及控制自动微调，自动微调通过将 CRS_CR 寄存器中的 AUTOTRIMEN 位置 1 来使能。

- 当频率误差低于容差限值时，意味着 TRIM 位域中的实际微调值是最优值，因此无需任何微调操作。
  - 指示 SYNCOK 状态
  - AUTOTRIM 模式下的 TRIM 值无变化
- 当频率误差低于警告限值但高于或等于容差限值时，意味着需要某种微调操作，但只需一步微调便可达到最优 TRIM 值。
  - 指示 SYNCOK 状态
  - 在 AUTOTRIM 模式下通过一步微调来调整 TRIM 值
- 当频率误差高于或等于警告限值但低于误差限值时，需要更强的微调操作，并且存在下一周期无法达到最优 TRIM 值的风险。
  - 指示 SYNCWARN 状态
  - 在 AUTOTRIM 模式下通过两步微调来调整 TRIM 值
- 当频率误差高于或等于误差限值时，频率超出微调范围。当 SYNC 输入不干净或某个 SYNC 脉冲丢失时（例如，当一个 USB SOF 被破坏时），也会发生这种情况。
  - 指示 SYNCERR 或 SYNCMISS 状态
  - AUTOTRIM 模式下的 TRIM 值无变化

注意： *如果 TRIM 位域的实际值接近其限值，则自动微调会强制其上溢或下溢，之后 TRIM 值会设置为限值，此时将指示 TRIMOVF 状态。*

在 AUTOTRIM 模式（CRS_CR 寄存器中的 AUTOTRIMEN 位置 1）下，CRS_CR 的 TRIM 位域由硬件调整并且是只读的。

11.4.5 CRS 初始化和配置

RELOAD 值

RELOAD 值必须根据预分频后目标频率与同步源频率之比进行选择。该值随后会减 1，0 值时达到预期同步。具体公式如下：

$$RELOAD = (f_{TARGET} / f_{SYNC}) - 1$$

RELOAD 位域的复位值对应于 48 MHz 的目标频率和 1 kHz 的同步信号频率（来自 USB 的 SOF 信号）。

FELIM 值

FELIM 值的选择与 HSI48 振荡器的特性及其典型微调步长紧密相关。最优值对应于微调步长的一半，以振荡器时钟节拍数表示。可使用以下公式：

$$FELIM = (f_{TARGET} / f_{SYNC}) * STEP[\%] / 100\% / 2$$

结果必须始终舍入为最接近的整数值以获得最佳微调响应。如果应用中不需要频繁的微调操作，可通过稍微增大 FELIM 值来增加滞后。

FELIM 位域的复位值对应于 (f_{TARGET} / f_{SYNC}) = 48000 以及 0.14% 的典型微调步长。

注意： *微调步长取决于产品，请查看数据手册获取准确设置。*

小心： 错误配置 RELOAD 和 FELIM 位域无法实现硬件保护，从而导致不定微调响应。预期工作模式需要正确设置 RELOAD 值（根据同步源频率），该值还大于 128 * FELIM 值（OUTRANGE 限值）。

11.5 CRS 低功耗模式

表 73. 低功耗模式对 CRS 的影响

模式	说明
睡眠	无影响。CRS 中断可使器件退出睡眠模式。
停止	CRS 寄存器被冻结。CRS 停止工作，直到退出停止模式且 HSI48 振荡器重启。
待机	CRS 外设掉电，退出待机模式后必须重新初始化。

11.6 CRS 中断

表 74. 中断控制位

中断事件	事件标志	使能控制位	清零标志位
预期同步	ESYNCF	ESYNCIE	ESYNCC
同步正常	SYNCOKF	SYNCOKIE	SYNCOKC
同步警告	SYNCWARNF	SYNCWARNIE	SYNCWARNC
同步或微调错误（TRIMOVF、SYNCMISS 和 SYNCERR）	ERRF	ERRIE	ERRC

11.7 CRS 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节*。

外设寄存器只能按字（32 位）进行访问。

11.7.1 CRS 控制寄存器 (CRS_CR)

CRS control register

偏移地址：0x00

复位值：0x0000 2000

复位值：0x0000 4000（支持 7 位 TRIM 宽度的产品）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	TRIM[5:0]						SW SYNC	AUTO TRIMEN	CEN	Res.	ESYNCIE	ERRIE	SYNC WARNIE	SYNC OKIE
		rw	rw	rw	rw	rw	rw	rt_w1	rw	rw		rw	rw	rw	rw

位 31:14 保留，必须保持复位值。

位 13:8 **TRIM[5:0]**：HSI48 振荡器平滑微调 (HSI48 oscillator smooth trimming)

这些位为 HSI48 振荡器提供用户可编程的微调值。可通过编程使其适应电压和温度的差异，使振荡器频率更为准确。

默认值为 32，该值对应于微调间隔的中间值。微调步在产品数据手册中指定。较高的 TRIM 值对应于较高的输出频率。

当 AUTOTRIMEN 位置 1 时，该位域由硬件控制并且是只读的。

位 7 **SWSYNC**：生成软件 SYNC 事件 (Generate software SYNC event)

该位由软件置 1 以生成软件 SYNC 事件。它由硬件自动清零。

0：无操作

1：生成软件 SYNC 事件。

位 6 **AUTOTRIMEN**：自动微调使能 (Automatic trimming enable)

该位根据在两个 SYNC 事件间测得的频率误差使能 TRIM 位的自动硬件调整。如果该位置 1，则 TRIM 位是只读的。TRIM 值可通过硬件一次调整一步或两步，具体取决于测得的频率误差值。更多详细信息，请参见*第 11.4.4 节*。

0：禁止自动微调，TRIM 位可由用户调整。

1：使能自动微调，TRIM 位是只读的且由硬件控制。

位 5 **CEN**：频率误差计数器使能 (Frequency error counter enable)

该位为频率误差计数器使能振荡器时钟。

0：禁止频率误差计数器

1：使能频率误差计数器

当该位置 1 时，CRS_CFGR 寄存器被写保护，无法修改。

位 4 保留，必须保持复位值。

位 3 **ESYNCIE**：预期 SYNC 中断使能 (Expected SYNC interrupt enable)

0：禁止预期 SYNC (ESYNCF) 中断

1：使能预期 SYNC (ESYNCF) 中断

位 2 **ERRIE**：同步或微调错误中断使能 (Synchronization or trimming error interrupt enable)
0：禁止同步或微调错误 (ERRF) 中断
1：使能同步或微调错误 (ERRF) 中断

位 1 **SYNCWARNIE**：SYNC 警告中断使能 (SYNC warning interrupt enable)
0：禁止 SYNC 警告 (SYNCWARNF) 中断
1：使能 SYNC 警告 (SYNCWARNF) 中断

位 0 **SYNCOKIE**：SYNC 事件正常中断使能 (SYNC event OK interrupt enable)
0：禁止 SYNC 事件正常 (SYNCOKF) 中断
1：使能 SYNC 事件正常 (SYNCOKF) 中断

11.7.2 CRS 配置寄存器 (CRS_CFGR)

CRS configuration register

只有禁止频率误差计数器时，才可写入该寄存器（CRS_CR 中的 CEN 位清零）。当计数器使能时，该寄存器被写保护。

偏移地址：0x04

复位值：0x2022 BB7F

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

位 31 **SYNCPOL**：SYNC 极性选择 (SYNC polarity selection)
该位由软件置 1 和清零，用于选择 SYNC 信号源的输入极性。
0：SYNC 在上升沿有效（默认）
1：SYNC 在下降沿有效

位 30 保留，必须保持复位值。

位 29:28 **SYNCSRC[1:0]**：SYNC 信号源选择 (SYNC signal source selection)
这些位由软件置 1 和清零，用于选择 SYNC 信号源（请参见*表 72*）：
00：选择 crs_sync_in_1 作为 SYNC 信号源
01：选择 crs_sync_in_2 作为 SYNC 信号源
10：选择 crs_sync_in_3 作为 SYNC 信号源
11：选择 crs_sync_in_4 作为 SYNC 信号源

注意：当使用 USB LPM（链路层电源管理）且器件处于睡眠模式时，周期性 USB SOF 不会由主机产生。因此，不会向 CRS 提供任何 SYNC 信号来校准运行状态下的 HSI48 振荡器。要确保从睡眠模式唤醒后所需的时钟精度，GPIO 上的 LSE 或参考时钟必须用作 SYNC 信号。

位 27 保留，必须保持复位值。

位 26:24 **SYNCDIV[2:0]**：SYNC 分频器 (SYNC divider)

这些位由软件置 1 和清零，用于控制 SYNC 信号的分频系数。

000：SYNC 不进行分频（默认）

001：SYNC 2 分频

010：SYNC 4 分频

011：SYNC 8 分频

100：SYNC 16 分频

101：SYNC 32 分频

110：SYNC 64 分频

111：SYNC 128 分频

位 23:16 **FELIM[7:0]**：频率误差限值 (Frequency error limit)

FELIM 包含用于评估 CRS_ISR 寄存器的 FECAP[15:0] 位中锁存的已捕获频率误差值的值。

有关 FECAP 评估的更多详细信息，请参见*第 11.4.4 节*。

位 15:0 **RELOAD[15:0]**：计数器重载值 (Counter reload value)

RELOAD 是每次发生 SYNC 事件时要装载到频率误差计数器中的值。

有关计数器行为的更多详细信息，请参见*第 11.4.3 节*。

11.7.3 CRS 中断和状态寄存器 (CRS_ISR)

CRS interrupt and status register

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
FECAP[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEDIR	Res.	Res.	Res.	Res.	TRIM OVF	SYNC MISS	SYNC ERR	Res.	Res.	Res.	Res.	ESYNCF	ERRF	SYNC WARNF	SYNC OKF
r					r	r	r					r	r	r	r

位 31:16 **FECAP[15:0]**：频率误差捕获 (Frequency error capture)

FECAP 是最后一次 SYNC 事件发生时锁存的频率误差计数器值。

有关 FECAP 使用的更多详细信息，请参见*第 11.4.4 节*。

位 15 **FEDIR**：频率误差方向 (Frequency error direction)

FEDIR 是最后一次 SYNC 事件发生时锁存的频率误差计数器的计数方向。它显示实际频率低于目标频率还是高于目标频率。

0：递增计数方向，实际频率高于目标频率

1：递减计数方向，实际频率低于目标频率

位 14:11 保留，必须保持复位值。

位 10 **TRIMOVF**：微调上溢或下溢 (Trimming overflow or underflow)

当自动微调尝试使 TRIM 值上溢或下溢时，该标志由硬件置 1。如果 CRS_CR 寄存器中的 ERRIE 位置 1，将生成中断。该位用软件清零，方法是将 CRS_ICR 寄存器中的 ERRC 位置 1。

0：未发出微调错误信号

1：已发出微调错误信号

位 9 **SYNCMISS**：SYNC 丢失 (SYNC missed)

当频率误差计数器达到值 FELIM * 128 且未检测到 SYNC 时，该标志由硬件置 1，这意味着 SYNC 脉冲丢失或频率误差过大（内部频率过高）而无法通过调整 TRIM 值来补偿，必须采取其他措施。此时，频率误差计数器停止（等待下一个 SYNC），并且会在 CRS_CR 寄存器中的 ERRIE 位置 1 时生成中断。该位用软件清零，方法是将 CRS_ICR 寄存器中的 ERRC 位置 1。

0：未发出 SYNC 丢失错误信号

1：已发出 SYNC 丢失错误信号

位 8 **SYNCERR**：SYNC 错误 (SYNC error)

当 SYNC 脉冲在 ESYNC 事件之前达到且所测得的频率误差大于或等于 FELIM * 128 时，该标志由硬件置 1。这意味着频率误差过大（内部频率过低）而无法通过调整 TRIM 值进行补偿，必须采取其他措施。如果 CRS_CR 寄存器中的 ERRIE 位置 1，将生成中断。该位用软件清零，方法是将 CRS_ICR 寄存器中的 ERRC 位置 1。

0：未发出 SYNC 错误信号

1：已发出 SYNC 错误信号

位 7:4 保留，必须保持复位值。

位 3 **ESYNCF**：预期 SYNC 标志 (Expected SYNC flag)

当频率误差计数器达到零值时，此标志由硬件置 1。如果 CRS_CR 寄存器中的 ESYNCIE 位置 1，将生成中断。该位用软件清零，方法是将 CRS_ICR 寄存器中的 ESYNCC 位置 1。

0：未发出预期 SYNC 信号

1：已发出预期 SYNC 信号

位 2 **ERRF**：错误标志 (Error flag)

发生同步或微调错误时，该标志由硬件置 1。它是 TRIMOVF、SYNCMISS 和 SYNCERR 的逻辑或运算的结果。如果 CRS_CR 寄存器中的 ERRIE 位置 1，将生成中断。该位用软件清零，方法是将 CRS_ICR 寄存器中的 ERRC 位置 1，这将清零 TRIMOVF、SYNCMISS 和 SYNCERR 位。

0：未发出同步或微调错误信号

1：已发出同步或微调错误信号

位 1 **SYNCWARNF**：SYNC 警告标志 (SYNC warning flag)

当所测得的频率误差大于或等于 FELIM * 3 但小于 FELIM * 128 时，该标志由硬件置 1。这意味着要补偿频率误差，必须通过两步或多步来调整 TRIM 值。如果 CRS_CR 寄存器中的 SYNCWARNIE 位置 1，将生成中断。该位用软件清零，方法是将 CRS_ICR 寄存器中的 SYNCWARNC 位置 1。

0：未发出 SYNC 警告信号

1：已发出 SYNC 警告信号

位 0 **SYNCOKF**：SYNC 事件正常标志 (SYNC event OK flag)

当所测得的频率误差小于 FELIM * 3 时，该标志由硬件置 1。这意味着无需调整 TRIM 值或只需一步微调便足以补偿频率误差。如果 CRS_CR 寄存器中的 SYNCOKIE 位置 1，将生成中断。该位用软件清零，方法是将 CRS_ICR 寄存器中的 SYNCOKC 位置 1。

0：未发出 SYNC 事件正常信号

1：已发出 SYNC 事件正常信号

11.7.4 CRS 中断标志清零寄存器 (CRS_ICR)

CRS interrupt flag clear register

偏移地址：0x0C

复位值：0x0000 0000

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

位 31:4 保留，必须保持复位值。

位 3 **ESYNCC**：预期 SYNC 清零标志 (Expected SYNC clear flag)
将 1 写入该位时，CRS_ISR 寄存器中 ESYNCF 标志将清零。

位 2 **ERRC**：错误清零标志 (Error clear flag)
将 1 写入该位将清零 TRIMOVF、SYNCMISS 和 SYNCERR 位，进而会清零 CRS_ISR 寄存器中的 ERRF 标志。

位 1 **SYNCWARNC**：SYNC 警告清零标志 (SYNC warning clear flag)
将 1 写入该位时，CRS_ISR 寄存器中 SYNCWARNF 标志将清零。

位 0 **SYNCOKC**：SYNC 事件正常清零标志 (SYNC event OK clear flag)
将 1 写入该位时，CRS_ISR 寄存器中 SYNCOKF 标志将清零。

11.7.5 CRS 寄存器映射

表 75. CRS 寄存器映射和复位值

偏移	寄存器名	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CRS_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIM[5:0]						SWSYNC	AUTOTRIMEN	CEN	Res.	ESYNCIE	ERRIE	SYNCWARNIE	SYNCOKIE
	Reset value																			1	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	CRS_CFGR	SYNCPOL	Res.	SYNC SRC [1:0]		Res.	SYNC DIV [2:0]			FELIM[7:0]								RELOAD[15:0]															
	Reset value	0		1	0		0	0	0	0	0	1	0	0	0	1	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1	1
0x08	CRS_ISR	FECAP[15:0]																FEDIR	Res.	Res.	Res.	Res.	TRIMOVF	SYNCMISS	SYNCERR	Res.	Res.	Res.	Res.	ESYNCF	ERRF	SYNCWARNF	SYNCOKF
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					0	0	0					0	0	0	0
0x0C	CRS_ICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ESYNCC	ERRC	SYNCWARNC	SYNCOKC
	Reset value																													0	0	0	0

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

12 通用 I/O (GPIO)

12.1 简介

每个通用 I/O 端口包括 4 个 32 位配置寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR 和 GPIOx_PUPDR）、2 个 32 位数据寄存器（GPIOx_IDR 和 GPIOx_ODR）、1 个 16 位复位寄存器 (GPIOx_BRR) 和 1 个 32 位置 1/复位寄存器 (GPIOx_BSRR)。

此外，所有 GPIO 都包括 1 个 32 位锁定寄存器 (GPIOx_LCKR)、2 个 32 位复用功能选择寄存器（GPIOx_AFRH 和 GPIOx_AFRL）和 1 个高速低电压寄存器 (GPIOx_HSLVR)。

12.2 GPIO 主要特性

- 输出状态：推挽或开漏 + 上拉/下拉
- 从输出数据寄存器 (GPIOx_ODR) 或外设（复用功能输出）输出数据
- 可为每个 I/O 选择不同的速度
- 输入状态：浮空、上拉/下拉、模拟
- 将数据输入到输入数据寄存器 (GPIOx_IDR) 或外设（复用功能输入）
- 置 1 和复位寄存器 (GPIOx_BSRR)，对 GPIOx_ODR 具有按位写权限
- 锁定机制 (GPIOx_LCKR)，可冻结 I/O 端口配置
- 模拟功能
- 复用功能选择寄存器
- 快速翻转，每次翻转最快只需要两个时钟周期
- 引脚复用非常灵活，允许将 I/O 引脚用作 GPIO 或多种外设功能中的一种
- 待机模式期间的 I/O 状态保持

12.3 GPIO 功能说明

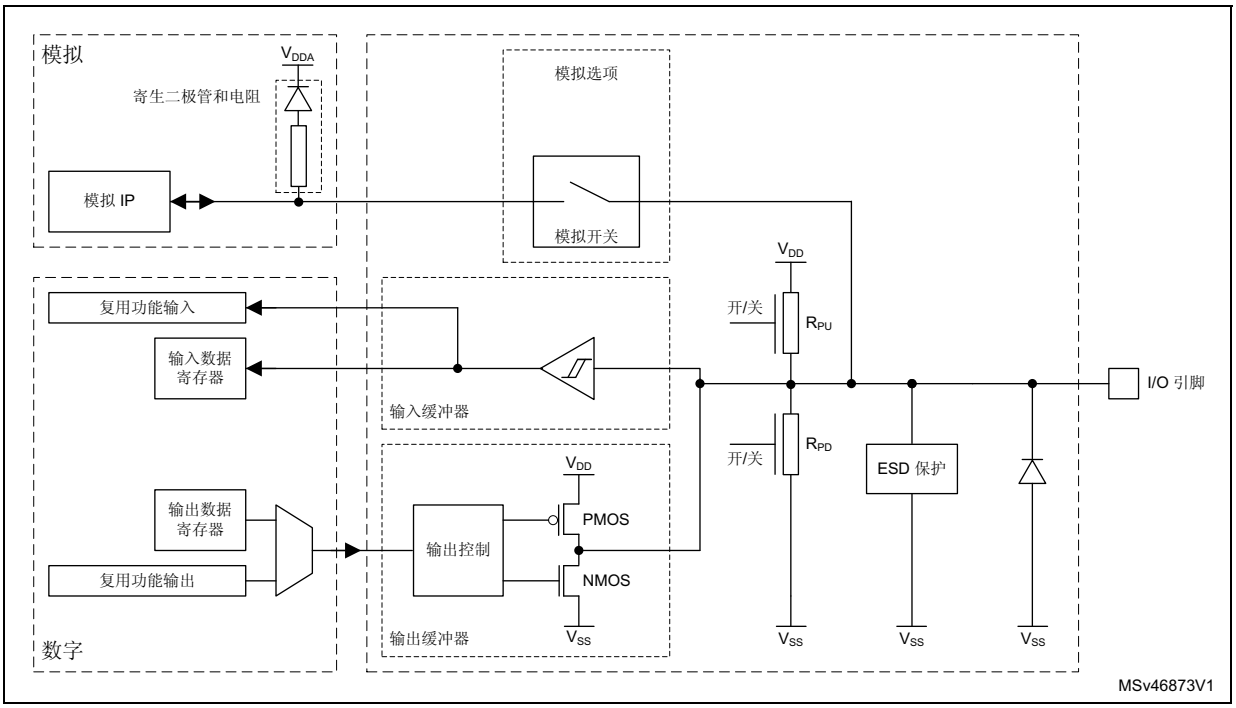
根据数据手册中列出的每个 I/O 端口的特性，可通过软件将通用 I/O (GPIO) 端口的各个端口位分别配置为多种模式：

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

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

下图显示了三伏或五伏容限 GPIO（TT 或 FT）的基本结构。*表 76* 给出了可能的端口位配置方案。

图 40. 三伏或五伏容限 GPIO（TT 或 FT）的结构



注意： *TT GPIO 上没有模拟开关，而是直接进行连接。模拟模块寄生电路不支持五伏容限。*

表 76. 端口位配置(1)

MODE(i) [1:0]	OTYPE(i)	OSPEED(i) [1:0]	PUPD(i) [1:0]		I/O 配置	
01	0	SPEED [1:0]	0	0	GP 输出	PP
	0		0	1	GP 输出	PP + PU
	0		1	0	GP 输出	PP + PD
	0		1	1	保留	
	1		0	0	GP 输出	OD
	1		0	1	GP 输出	OD + PU
	1		1	0	GP 输出	OD + PD
	1		1	1	保留（GP 输出 OD）	

表 76. 端口位配置(1)（续）

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

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

12.3.1 通用 I/O (GPIO)

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

复位后，调试引脚处于复用功能上拉/下拉状态：

- PA15：JTDI 处于上拉状态
- PA14：JTCK/SWCLK 处于下拉状态
- PA13：JTMS/SWDIO 处于上拉状态
- PB4：NJTRST 处于上拉状态
- PB3：JTDO/TRACESWO 处于浮空状态，未处于上拉/下拉状态

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

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

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

12.3.2 I/O 引脚复用功能复用器和映射

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

每个 I/O 引脚都有一个复用器，该复用器采用多达 16 路复用功能输入（AF0 到 AF15），可通过 GPIOx_AFRL（针对引脚 0 到 7）和 GPIOx_AFRH（针对引脚 8 到 15）寄存器对这些输入进行配置：

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

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

要在指定配置下使用 I/O，用户必须按照以下步骤操作：

- **调试功能**：每个器件复位后，立即将这些引脚分配为可由调试主机使用的复用功能引脚。
- **GPIO**：在 GPIOx_MODER 寄存器中将所需 I/O 配置为输出、输入或模拟通道。
- **外设复用功能**：
  - 在 GPIOx_AFRL 或 GPIOx_AFRH 寄存器中，将 I/O 连接到所需的 AFx。
  - 通过 GPIOx_OTYPER、GPIOx_PUPDR 和 GPIOx_OSPEEDR 寄存器分别选择类型、上拉/下拉以及输出速度。
  - 在 GPIOx_MODER 寄存器中将所需 I/O 配置为复用功能。
- Cortex-M33 复用功能 (EVENTOUT)
  - 通过将 I/O 引脚配置为输出 (AF15)，可以使用 Cortex-M33 输出 EVENTOUT 信号。执行 SEV 指令后，可以通过配置的引脚发出事件信号。
  - EVENTOUT 信号可在内部用于触发一些外设（请参见*第 14 章：外设互连矩阵*）。
- **其他功能**：
  - 对于 ADC、DAC、OPAMP 和 COMP，在 GPIOx_MODER 寄存器中将所需 I/O 配置为模拟模式，并在 ADC、DAC、OPAMP 和 COMP 寄存器中配置所需功能。
  - 对于 RTC、WKUPx 和振荡器等其他功能，在相关的 RTC、PWR 和 RCC 寄存器中配置所需功能。这些功能优先于标准 GPIO 寄存器中的配置。

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

12.3.3 I/O 端口控制寄存器

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

12.3.4 I/O 端口数据寄存器

每个 GPIO 都具有 2 个 16 位存储器映射的数据寄存器：输入和输出数据寄存器（*GPIO 端口输入数据寄存器 (GPIOx_IDR)（x = A 到 D 和 H）* 和 *GPIO 端口输出数据寄存器 (GPIOx_ODR)（x = A 到 D 和 H）*）。

GPIOx_ODR 用于存储待输出数据，可对其进行读/写访问。通过 I/O 输入的数据存储到输入数据寄存器 (GPIOx_IDR) 中，它是一个只读寄存器。

12.3.5 I/O 数据位操作

位设置/复位寄存器 (GPIOx_BSRR) 是一个 32 位寄存器，它允许应用程序在输出数据寄存器 (GPIOx_ODR) 中设置和复位每个单独的位。位设置/复位寄存器的大小是 GPIOx_ODR 的两倍。

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

在 GPIOx_BSRR 中向任何位写入 0 都不会对 GPIOx_ODR 中的对应位产生任何影响。如果在 GPIOx_BSRR 中同时尝试对某个位执行置 1 和复位操作，则置 1 操作优先。

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

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

12.3.6 GPIO 锁定机制

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

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

LOCK 序列只能通过对 GPIOx_LCKR 寄存器进行字（32 位长）访问的方式来执行，因为 GPIOx_LCKR 的位 16 必须与 [15:0] 位同时设置。

12.3.7 I/O 复用功能输入/输出

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

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

要了解在每个 GPIO 引脚上复用了哪些功能，请参见器件数据手册。

12.3.8 外部中断线/唤醒线

所有端口都具有外部中断功能。要使用外部中断线，可将端口配置为输入、输出或复用功能模式（端口不能配置为模拟模式）。请参见*第 17 章：扩展中断和事件控制器 (EXTI)*。

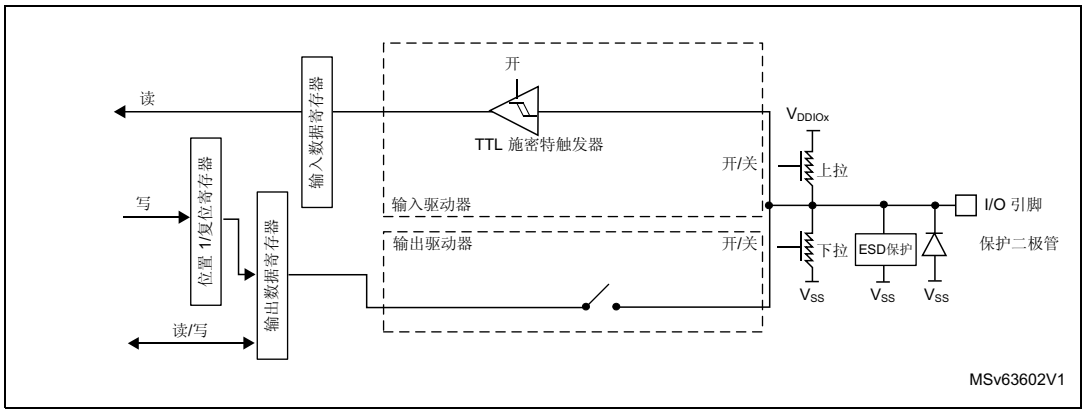
12.3.9 输入配置

对 I/O 端口进行编程作为输入时：

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

下图说明了 I/O 端口位的输入配置。

图 41. 输入浮空/上拉/下拉配置



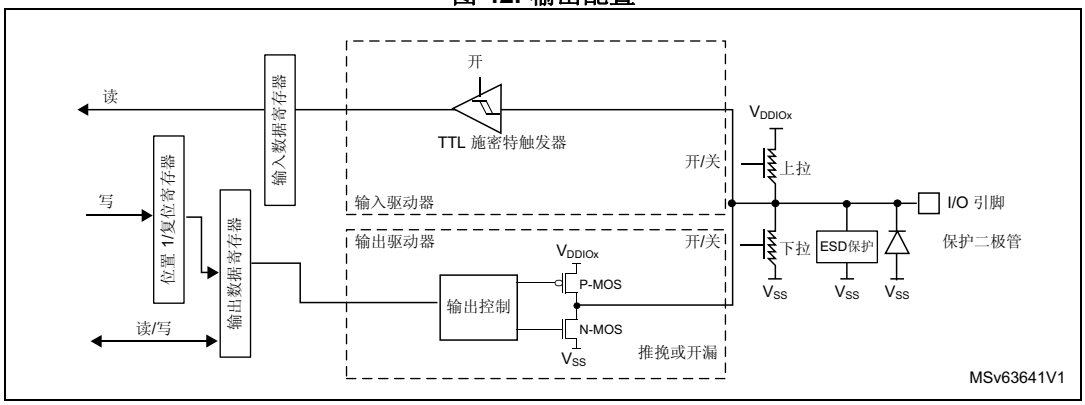
12.3.10 输出配置

对 I/O 端口进行编程作为输出时：

- 使能输出缓冲器：
  - 开漏模式：输出寄存器中的 0 可激活 N-MOS，而输出寄存器中的 1 会使端口保持高阻态 (Hi-Z)（P-MOS 始终不激活）。
  - 推挽模式：输出寄存器中的 0 可激活 N-MOS，而输出寄存器中的 1 可激活 P-MOS。
- 激活施密特触发器输入。
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开上拉和下拉电阻。
- 输入数据寄存器每个 AHB 时钟周期对 I/O 引脚上的数据进行一次采样。
- 对输入数据寄存器的读访问可获取 I/O 状态。
- 对输出数据寄存器的读访问可获取最后的写入值。

下图说明了 I/O 端口位的输出配置。

图 42. 输出配置



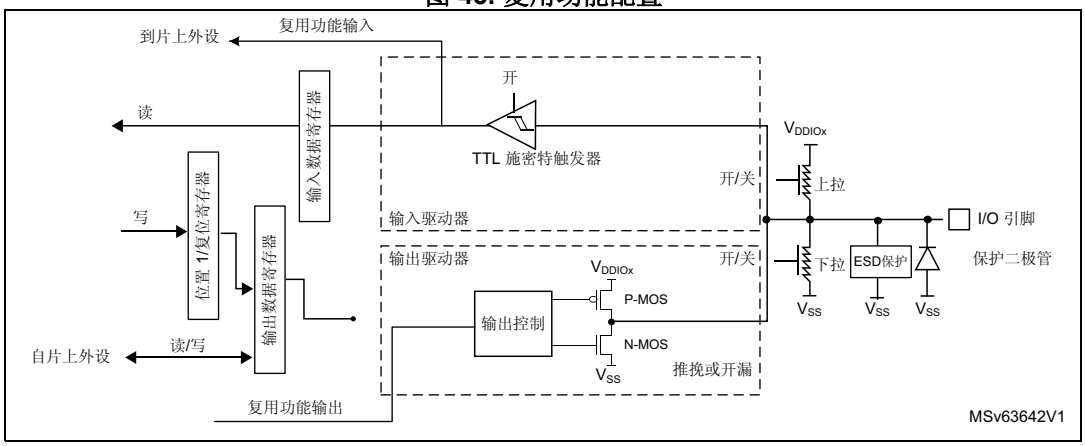
12.3.11 复用功能配置

对 I/O 端口进行编程作为复用功能时：

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

下图说明了 I/O 端口位的复用功能配置。

图 43. 复用功能配置



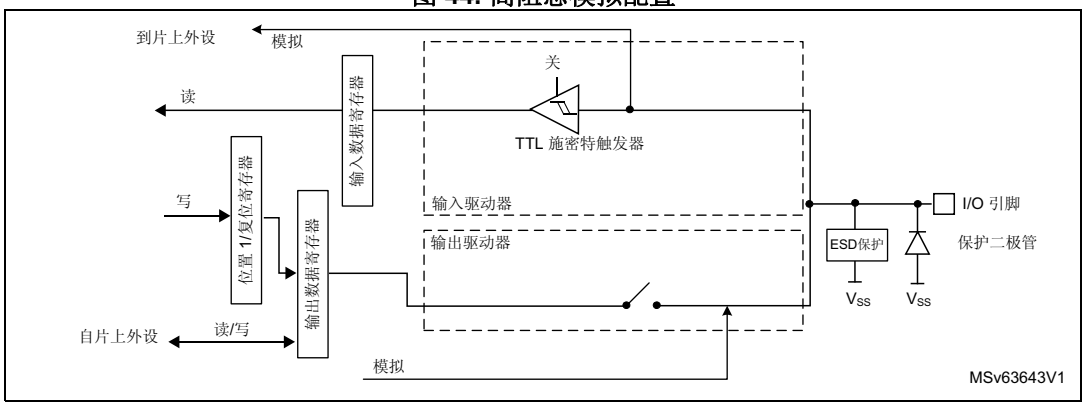
12.3.12 模拟配置

对 I/O 端口进行编程作为模拟配置时：

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

下图说明了 I/O 端口位的高阻态模拟输入配置。

图 44. 高阻态模拟配置



12.3.13 将 HSE 或 LSE 振荡器引脚用作 GPIO

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

当 HSE 或 LSE 振荡器打开（通过将 RCC_CSR 寄存器中的 HSEON 或 LSEON 位置 1）时，振荡器会控制与其相关联的引脚，这些引脚的 GPIO 配置不起作用。

将振荡器配置为用户外部时钟模式时，仅为时钟输入保留此引脚，OSC_OUT 或 OSC32_OUT 引脚仍可用作常规 GPIO。

12.3.14 在 RTC 电源域中使用 GPIO 引脚

当内核电源域掉电时（器件进入待机模式时），PC13/PC14/PC15 GPIO 功能会丢失。在这种情况下，如果不通过 RTC 配置旁通其 GPIO 配置，这些引脚将被设置为模拟输入模式。

有关 RTC 的 I/O 控制的详细信息，请参见*第 32.3 节：RTC 功能描述*。

12.3.15 待机模式期间的 I/O 状态保持

在待机模式下，I/O 默认处于浮空状态。

如果 PWR_IORETR 寄存器中的 IORETEN 位置 1，则会在进入待机模式期间对 I/O 状态进行采样。I/O 状态通过上拉和下拉电阻施加给引脚。从待机模式唤醒后仍会继续应用上拉和下拉电阻，直到软件将 PWR_IORETR 寄存器中的 IORETEN 位清零。

12.3.16 特权模式和非特权模式

所有 GPIO 寄存器均可通过特权访问和非特权访问进行读写操作。

12.3.17 高速低电压模式 (HSLV)

一些 I/O 可通过配置为 HSLV 模式来提高其在低电压下的最大速度。I/O HSLV 位用于控制 I/O 输出速度是优化为在 3.3 V（默认设置）下工作还是在 1.8 V (HSLV = 1) 下工作。

小心： 如果 I/O 电源（V_{DD} 或 V_{DDIO2}）高于 2.7 V，则 I/O HSLV 配置位不得置 1。在电压高于 2.7 V 时将该位置 1 可能会损坏器件。只有当相应的选项位激活时（IO_VDD_HSLV 或 IO_VDDIO2_HSLV，具体取决于 I/O 电源，请参见*第 7.4.1 节：关于选项字节*），I/O HSLV 位才能置 1。该功能没有相关的硬件保护，因此建议仅将其用作固定 I/O 电源的静态配置。

12.3.18 I/O 补偿单元

I/O 换向斜率 (tfall/trise) 可通过软件根据制程、电压和温度条件进行调整，从而减少电源上的 I/O 噪声。更多详细信息，请参见*第 13 章：系统配置、自举和安全性 (SBS)*。

12.4 GPIO 寄存器

本节对 GPIO 寄存器进行了详细介绍。

可按字、半字或字节模式写入外设寄存器。

12.4.1 GPIO 端口模式寄存器 (GPIOx_MODER)（x = A 到 D 和 H）

GPIO port mode register

偏移地址：0x00

复位值：0xABFF FFFF（端口 A）

复位值：0xFF33 FEBF（端口 B）

复位值：0xFFFF FFFF（端口 C）

复位值：0x0000 0030（端口 D）

复位值：0x0000 000F（端口 H）

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

位 31:0 **MODEy[1:0]**：端口 x 配置 I/O 引脚 y (Port x configuration I/O pin y)（y = 15 到 0）

这些位通过软件写入，用于配置 I/O 模式。

00：输入模式

01：通用输出模式

10：复用功能模式

11：模拟模式（复位状态）

注意：当所选封装上没有相应的 I/O 时，该位域保留且必须保持复位值。

12.4.2 GPIO 端口输出类型寄存器 (GPIOx_OTYPER)（x = A 到 D 和 H）

GPIO port output type register

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:0 **OTy**：端口 x 配置 I/O 引脚 y (Port x configuration I/O pin y)（y = 15 到 0）
这些位通过软件写入，用于配置 I/O 输出类型。
0：输出推挽（复位状态）
1：输出开漏
注意：当所选封装上没有相应的 I/O 时，该位保留且必须保持复位值。

12.4.3 GPIO 端口输出速度寄存器 (GPIOx_OSPEEDR)（x = A 到 D 和 H）

GPIO port output speed register

偏移地址：0x08

复位值：0x0C00 0000（端口 A）

复位值：0x0000 00C0（端口 B）

复位值：0x0000 0000（其他端口）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSPEED15[1:0]		OSPEED14[1:0]		OSPEED13[1:0]		OSPEED12[1:0]		OSPEED11[1:0]		OSPEED10[1:0]		OSPEED9[1:0]		OSPEED8[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSPEED7[1:0]		OSPEED6[1:0]		OSPEED5[1:0]		OSPEED4[1:0]		OSPEED3[1:0]		OSPEED2[1:0]		OSPEED1[1:0]		OSPEED0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **OSPEEDy[1:0]**：端口 x 配置 I/O 引脚 y (Port x configuration I/O pin y)（y = 15 到 0）
这些位通过软件写入，用于配置 I/O 输出速度。
00：低速
01：中速
10：高速
11：超高速
注意：关于每种速度的频率规范以及电源和负载条件，请参见器件数据手册。
当所选封装上没有相应的 I/O 时，该位域保留且必须保持复位值。

12.4.4 GPIO 端口上拉/下拉寄存器 (GPIOx_PUPDR)（x = A 到 D 和 H）

GPIO port pull-up/pull-down register

偏移地址：0x0C

复位值：0x6400 0000（端口 A）

复位值：0x0000 0100（端口 B）

复位值：0x0000 0000（其他端口）

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

位 31:0 **PUPDy[1:0]**：端口 x 配置 I/O 引脚 y (Port x configuration I/O pin y)（y = 15 到 0）

这些位通过软件写入，用于配置 I/O 上拉或下拉。

00：无上拉或下拉

01：上拉

10：下拉

11：保留

注意：当所选封装上没有相应的 I/O 时，该位域保留且必须保持复位值。

12.4.5 GPIO 端口输入数据寄存器 (GPIOx_IDR)（x = A 到 D 和 H）

GPIO port input data register

偏移地址：0x10

复位值：0x0000 XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 保留，必须保持复位值。

位 15:0 **IDy**：端口 x 输入数据 I/O 引脚 y (Port x input data I/O pin y)（y = 15 到 0）

这些位为只读。它们包含相应 I/O 端口的输入值。

注意：当所选封装上没有相应的 I/O 时，该位保留且必须保持复位值。

12.4.6 GPIO 端口输出数据寄存器 (GPIOx_ODR)（x = A 到 D 和 H）

GPIO port output data register

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OD15	OD14	OD13	OD12	OD11	OD10	OD9	OD8	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:0 **ODy**：端口输出数据 I/O 引脚 y (Port output data I/O pin y)（y = 15 到 0）
这些位可通过软件读取和写入。
注意：对于原子位置 1/复位，通过写入 GPIOx_BSRR 或 GPIOx_BRR 寄存器，可分别置 1 和/或复位 OD 位（x = A 到 D 和 H）。
当所选封装上没有相应的 I/O 时，该位保留且必须保持复位值。

12.4.7 GPIO 端口位置 1/复位寄存器 (GPIOx_BSRR)（x = A 到 D 和 H）

GPIO port bit set/reset register

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位 31:16 **BRy**：端口 x 复位 I/O 引脚 y (Port x reset I/O pin y)（y = 15 到 0）
这些位为只写。读取这些位会返回值 0x0000。
0：不会对相应的 ODy 位执行任何操作
1：复位相应的 ODy 位
注意：如果同时将 BSy 和 BRy 置 1，则 BSy 的优先级更高。
当所选封装上没有相应的 I/O 时，该位保留且必须保持复位值。

位 15:0 **BSy**：端口 x 置 1 I/O 引脚 y (Port x set I/O pin y)（y = 15 到 0）
这些位为只写。读取这些位会返回值 0x0000。
0：不会对相应的 ODy 位执行任何操作
1：将相应的 ODy 位置 1
注意：当所选封装上没有相应的 I/O 时，该位保留且必须保持复位值。

12.4.8 GPIO 端口配置锁定寄存器 (GPIOx_LCKR)（x = A 到 D 和 H）

GPIO port configuration lock register

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

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

每个锁定位冻结一个特定的配置寄存器（控制寄存器和复用功能寄存器）。

偏移地址：0x1C

复位值：0x0000 0000

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

位 31:17 保留，必须保持复位值。

位 16 **LCKK**：锁定键 (Lock key)

可随时读取该位。只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。直到下一次 MCU 复位或外设复位时，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]

- 锁定键读取

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 MCU 复位或外设复位为止。

位 15:0 **LCKy**：端口 x 锁定 I/O 引脚 y (Port x lock I/O pin y)（y = 15 到 0）

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

注意： *当所选封装上没有相应的 I/O 时，该位保留且必须保持复位值。*

12.4.9 GPIO 复用功能低位寄存器 (GPIOx_AFRL)（x = A 到 D 和 H）

GPIO alternate function low register

偏移地址：0x20

复位值：0x0000 0000

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

位 31:0 **AFSELy[3:0]**：端口 x I/O 引脚 y 的复用功能选择 (Alternate function selection for port x I/O pin y)（y = 7 到 0）

这些位通过软件写入，用于配置复用功能 I/O。

0000：AF0
0001：AF1
0010：AF2
0011：AF3
0100：AF4
0101：AF5
0110：AF6
0111：AF7
1000：AF8
1001：AF9
1010：AF10
1011：AF11
1100：AF12
1101：AF13
1110：AF14
1111：AF15

注意：当所选封装上没有相应的 I/O 时，该位域保留且必须保持复位值。

12.4.10 GPIO 复用功能高位寄存器 (GPIOx_AFRH)（x = A 到 D 和 H）

GPIO alternate function high register

偏移地址：0x24

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL15[3:0]				AFSEL14[3:0]				AFSEL13[3:0]				AFSEL12[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFSEL11[3:0]				AFSEL10[3:0]				AFSEL9[3:0]				AFSEL8[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **AFSELy[3:0]**：端口 x I/O 引脚 y 的复用功能选择 (Alternate function selection for port x I/O pin y)（y = 15 到 8）

这些位通过软件写入，用于配置复用功能 I/O。

0000：AF0
0001：AF1
0010：AF2
0011：AF3
0100：AF4
0101：AF5
0110：AF6
0111：AF7
1000：AF8
1001：AF9
1010：AF10
1011：AF11
1100：AF12
1101：AF13
1110：AF14
1111：AF15

注意：当所选封装上没有相应的 I/O 时，该位域保留且必须保持复位值。

12.4.11 GPIO 端口位复位寄存器 (GPIOx_BRR)（x = A 到 D 和 H）

GPIO port bit reset register

偏移地址：0x28

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15:0 **BRy**：端口 x 复位 IO 引脚 y (Port x reset IO pin y)（y = 15 到 0）

这些位为只写。读取这些位会返回值 0x0000。

0：不会对相应的 ODy 位执行任何操作
1：复位相应的 ODy 位

注意：当所选封装上没有相应的 I/O 时，该位保留且必须保持复位值。

12.4.12 GPIO 高速低电压寄存器 (GPIOx_HSLVR)（x = A 到 D 和 H）

GPIO high-speed low-voltage register

偏移地址：0x2C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSLV 15	HSLV 14	HSLV 13	HSLV 12	HSLV 11	HSLV 10	HSLV9	HSLV8	HSLV7	HSLV6	HSLV5	HSLV4	HSLV3	HSLV2	HSLV1	HSLV0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:0 **HSLVy**：端口 x 高速低电压配置 (Port x high-speed low-voltage configuration)（y = 15 到 0）
这些位通过软件写入，用于在 I/O 电源处于低电压时优化 I/O 速度。
每个位都仅在相应的 IO_VDD_HSLV/IO_VDDIO2_HSLV 用户选项位置 1 时有效。该位必须仅在 I/O 电源电压低于 2.7 V 时使用。
I/O 电源（VDD 或 VDDIO2）高于 2.7 V 时将这些位置 1 可能会导致破坏性结果。
0：禁止 I/O 速度优化
1：使能 I/O 速度优化

注意：并非所有 I/O 都支持 HSLV 模式。有关支持该特性的 I/O 列表，请参见相应数据手册中的 I/O 结构。其他 I/O HSLV 配置必须保持复位值。
当所选封装上没有相应的 I/O 时，该位保留且必须保持复位值。

12.4.13 GPIO 寄存器映射

表 77. GPIO 寄存器映射和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**GPIOx_MODER (x = A to D and H)**	MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]		MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
	Reset value for port A	1	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	Reset value for port B	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1
	Reset value for port C	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	Reset value for port D	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
	Reset value for port H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
0x04	**GPIOx_OTYPER (x = A to D and H)**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 77. GPIO 寄存器映射和复位值（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x08	**GPIOx_ OSPEEDR (x = A to D and H)**	OSPEED15[1:0]		OSPEED14[1:0]		OSPEED13[1:0]		OSPEED12[1:0]		OSPEED11[1:0]		OSPEED10[1:0]		OSPEED9[1:0]		OSPEED8[1:0]		OSPEED7[1:0]		OSPEED6[1:0]		OSPEED5[1:0]		OSPEED4[1:0]		OSPEED3[1:0]		OSPEED2[1:0]		OSPEED1[1:0]		OSPEED0[1:0]	
	Reset value for port A	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Reset value for port B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
	Reset value for ports C, D and H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	**GPIOx_PUPDR (x = A to D and H)**	PUPD15[1:0]		PUPD14[1:0]		PUPD13[1:0]		PUPD12[1:0]		PUPD11[1:0]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[1:0]		PUPD7[1:0]		PUPD6[1:0]		PUPD5[1:0]		PUPD4[1:0]		PUPD3[1:0]		PUPD2[1:0]		PUPD1[1:0]		PUPD0[1:0]	
	Reset value for port A	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Reset value for port B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	Reset value for ports C, D and H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	**GPIOx_IDR (x = A to D and H)**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	Reset value																	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x14	**GPIOx_ODR (x = A to D and H)**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OD15	OD14	OD13	OD12	OD11	OD10	OD9	OD8	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	**GPIOx_BSRR (x = A to D and H)**	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0	BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	**GPIOx_LCKR (x = A to D and H)**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LCKK	LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	**GPIOx_AFRL (x = A to D and H)**	AFSEL7[3:0]				AFSEL6[3:0]				AFSEL5[3:0]				AFSEL4[3:0]				AFSEL3[3:0]				AFSEL2[3:0]				AFSEL1[3:0]				AFSEL0[3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	**GPIOx_AFRH (x = A to D and H)**	AFSEL15[3:0]				AFSEL14[3:0]				AFSEL13[3:0]				AFSEL12[3:0]				AFSEL11[3:0]				AFSEL10[3:0]				AFSEL9[3:0]				AFSEL8[3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	**GPIOx_BRR (x = A to D and H)**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	**GPIOx_HSLVR (x = A to D and H)**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSLV15	HSLV14	HSLV13	HSLV12	HSLV11	HSLV10	HSLV9	HSLV8	HSLV7	HSLV6	HSLV5	HSLV4	HSLV3	HSLV2	HSLV1	HSLV0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 2.2 节*。

13 系统配置、自举和安全性 (SBS)

13.1 SBS 简介

STM32H503xx 器件配有一组位于 SBS 中的配置寄存器。除各种器件配置之外，此 SBS 外设还控制关键的自举和安全功能，包括调试控制。

13.2 SBS 主要特性

- 系统配置
  - 管理稳健性功能
  - 使能/禁止一些 I/O 的 FMP 高驱动模式
  - 管理 I/O 补偿单元
  - 配置寄存器安全访问
- 自举控制
  - 系统复位后，根据当前配置（例如 PRODUCT_STATE (sbs_product_state)）配置 Cortex-M33 自举地址和时域隔离级别
  - 管理通过隐藏保护级别 (HDPL) 单调计数器实现的时域隔离功能
- 调试控制
  - 控制器件调试接口的开启，保证事件的顺序排列，从而确保器件的安全性

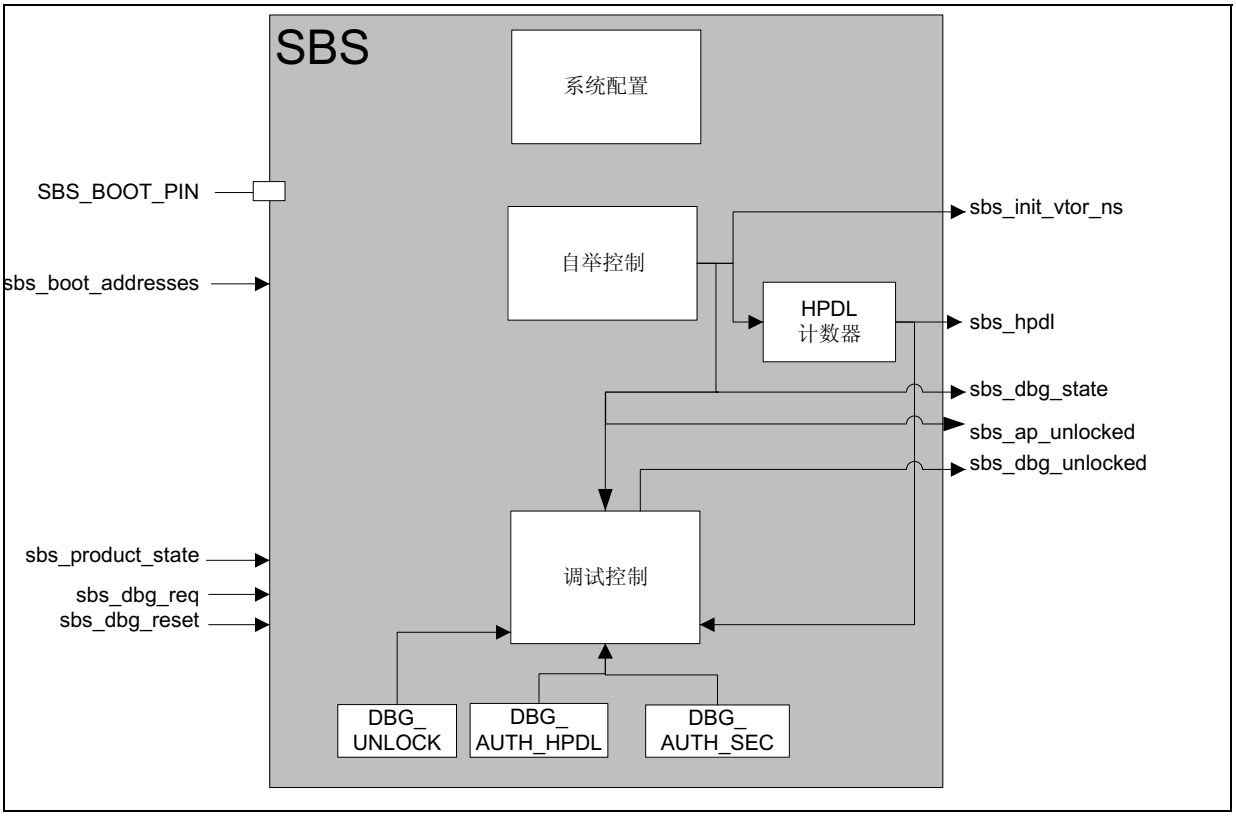
13.3 SBS 功能说明

13.3.1 SBS 框图

下图显示了 SBS 框图，包括四个主要功能：

- 系统配置
- 自举控制
- 调试控制

图 45. SBS 框图



13.3.2 SBS 信号

下表详细介绍了连接 SBS 的用户相关内部信号。

表 78. SBS 内部输入/输出信号

信号名称	类型	说明
SBS_BOOT_PIN	输入	选择在用户 Flash 或自举程序上自举
sbs_boot_addresses		Flash 定义的地址列表： – BOOT_ADDR_NS：自举地址
sbs_product_state		基于 PRODUCT_STATE 选项字节的信号，会根据产品的使用情况激活不同的安全机制。关于预期值，请参见*第 7 章：嵌入式 Flash (FLASH)*。
sbs_dbg_req		自举时启动调试认证协议。

表 78. SBS 内部输入/输出信号（续）

信号名称	类型	说明
sbs_init_vtor_ns	输出	Cortex-M33 的向量地址
sbs_hdpl		HDPL（时域隔离级别，自举级别 ID，用于隔离自举级别） 此信号用于反映单调计数器，该计数器可以从 0 递增到 3，但只能在软件复位或 POR 时复位为 0。
sbs_ap_unlocked		控制 Cortex-M33 访问端口。
sbs_dbg_unlocked		解锁 Cortex-M33 的调试（置 1 时）
sbs_dbg_state		使用 OPEN (0xB4)、CLOSE_SEC (0x8A) 或 CLOSE (0x51) 填充 Flash 的调试状态。
sbs_dbg_reset	输入	复位 SBS_DBGCR（来自 RCC）。通过 DBGMCU 进行配置时，可通过系统复位或 POR 实现复位。

13.3.3 SBS 复位和时钟

SBS 配置端口由 AHB 总线时钟提供时钟。在 DBGMCU 中控制常规复位和调试配置复位。

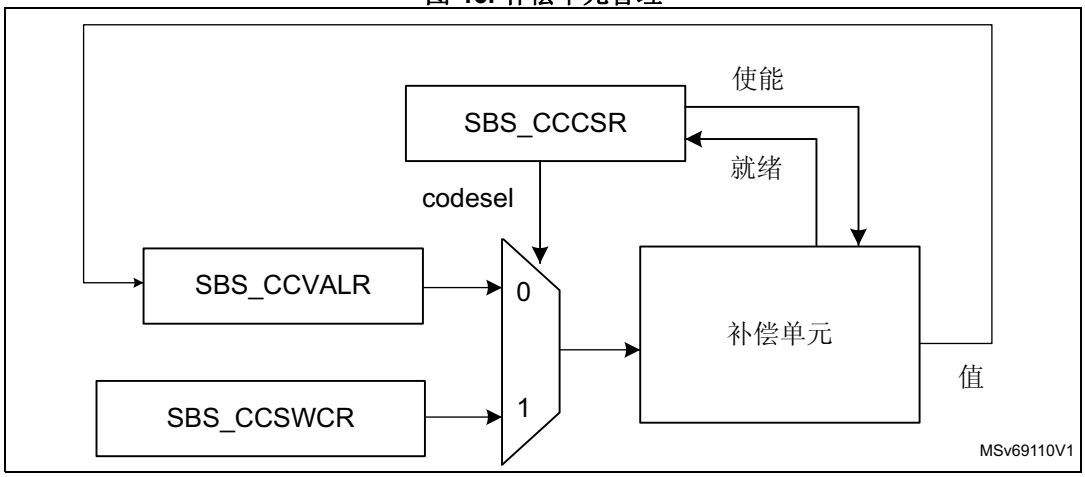
13.3.4 SBS 系统配置

SBS I/O 补偿单元管理

I/O 补偿单元会为 I/O 缓冲区生成一个 8 位值：4 位用于 N-MOS，4 位用于 P-MOS。该值取决于过程、电压和温度 (PVT) 工作条件。这些位用于控制 I/O 缓冲区的输出阻抗和 I/O 换向的压摆率（t_{fall} 和 t_{rise} 时间），从而降低 I/O 噪声对电源的影响。

如下图所示，补偿单元分为两个模块：一个模块提供当前 PVT 的最佳代码，另一个模块用于驱动由软件控制的模块。

图 46. 补偿单元管理



当 SBS_CCCSR 中的 READY 标志置 1 时，可以读取补偿单元值。通过 SBS_CCCSR 中的 CODESEL，应用程序可以在以下两个选项之间选择要应用的值：单元中的代码或 SBS_CCSWCR 中的代码。

STM32H503xx 器件内置两个补偿单元：

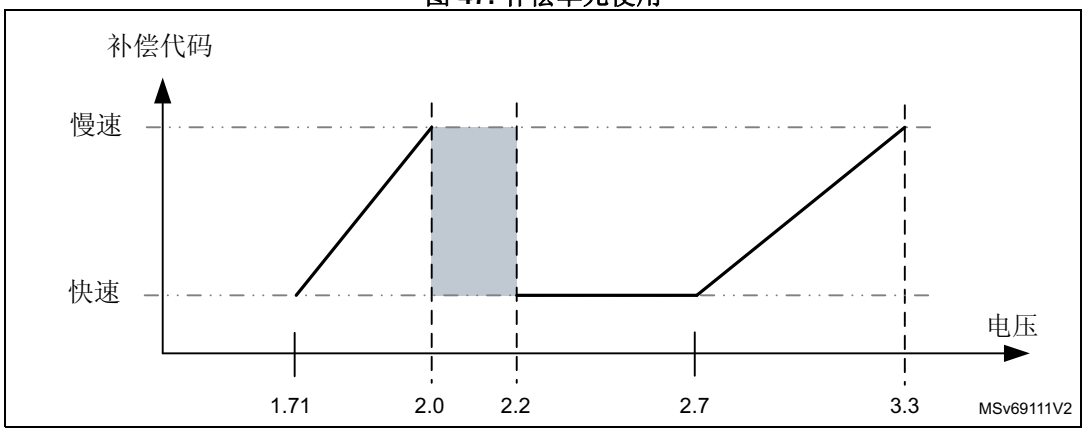
- 一个用于由 V_{DDIO} 电源轨供电的 I/O
- 一个用于由 V_{DDIO2} 电源轨供电的 I/O

默认情况下，补偿单元处于禁止状态，所有 I/O 应用同一个固定的代码。

注意： *仅当 2.7 V ≤ V_{DDIOx} ≤ 3.6 V 或 1.71 V ≤ V_{DDIOx} ≤ 2 V 时，才能使用补偿单元（见下图）。*

注意： *补偿单元只能在使能 CSI 振荡器时使用，有关 CSI 振荡器的更多详细信息，请参见第 10 章：复位和时钟控制 (RCC)。*

图 47. 补偿单元使用

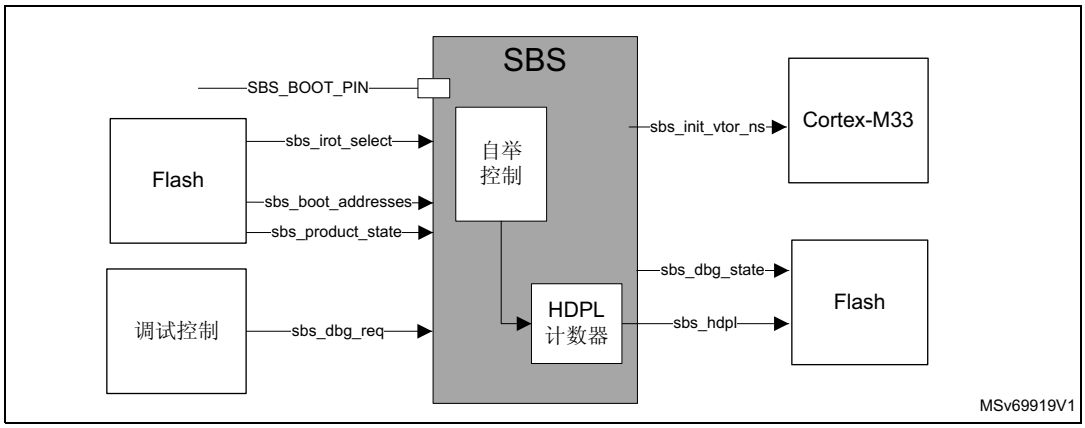


13.3.5 SBS 自举控制

可以根据产品设置使用 SBS 来控制自举入口点。下面列出了主要的自举控制操作：

- 在启动调试认证序列时自举。
- 选择从自举程序或从用户 Flash 自举。
- 初始化 HDPL 自举值。

图 48. SBS 自举控制



选择自举配置时，需考虑以下产品设置：

- SBS_BOOT_PIN：选择在用户 Flash 上自举
  - sbs_boot_addresses：Flash 定义的地址列表：BOOT_ADDR_NS：自举地址
- PRODUCT_STATE：选项字节，用于根据产品的使用情况激活不同的安全机制。关于预期值，请参见*第 7 章：嵌入式 Flash (FLASH)*。
- sbs_dbg_req：用于在自举时启动调试认证协议

自举控制逻辑设置以下数据：

- sbs_init_vtor_ns：Cortex-M33 的向量地址
- HDPL（请参见下文介绍）

SBS HDPL（时域隔离级别）管理

HDPL 是一个在自举阶段递增的单调计数器。只有在上电或系统复位后，HDPL 才会复位为其默认值。此默认值（0 或 1）取决于器件生命周期（如自举逻辑中所定义）。

STM32H503xx 器件使用 HDPL 信息在自举过程中自动隔离代码及其相关机密信息（例如密钥）。递增 HPDL 可以确保一个自举阶段的私有代码和数据不会受到后续自举阶段的直接访问。

HDPL 由用户 Flash 使用。更多详细信息，请参见*第 7 章：嵌入式 Flash (FLASH)*。

HDPL 可以取 0 到 3 之间的值。当达到值 3 时，HDPL 将保持此值直到复位。当前的 HDPL 值可在 SBS_HDPLSR 的 HDPL 位域中读取。

要将 HDPL 递增 1，应用程序必须向 SBS_HDPLCR 中的 INCR_HDPL 写入 0x6A。在这种递增之后，以及在执行任何后续操作之前，用户必须读取 SBS_HDPLSR 来检查 HDPL 是否已有效递增。

表 79. HDPL 编码值

HDPL	代码
0	0xB4
1	0x51
2	0x8A
3	0x6F
	所有其他值

表 80. SBS 自举逻辑

输入				输出		
sbs_product_state	sbs_dbg_req	SBS_BOOT_PIN	sbs_boot_addresses	sbs_init_vtor_ns	sbs_hdpl	sbs_dbg_state (FLASH_STATE)
除锁定外的任何状态	1	x	BOOT_DBG_AUTH_ADD	x	1	取决于 PRODUCT_STATE
开放	0	0	BOOT_ADDR_NS	BOOT_ADDR_NS	1	OPEN
	0	1	START_ADD_BOOT_LOADER	x	≥1	OPEN
配置	0	x	START_ADD_BOOT_LOADER	x	≥1	CLOSE
iROT 已配置	0	x	BOOT_ADDR_NS	BOOT_ADDR_NS	1	CLOSE
关闭	0	x	BOOT_ADDR_NS	BOOT_ADDR_NS	1	CLOSE
锁定	0	x	BOOT_ADDR_NS	BOOT_ADDR_NS	1	CLOSE

13.3.6 SBS 调试控制

SBS 调试控制用于管理调试启动，在寄存器设置中或通过调试认证控制处理产品上下文（PRODUCT_STATE、HDPL）。

禁止调试时，调试访问端口 AP0（通过邮箱）、Cortex-M33 访问端口和 CPU 调试接口将被锁定。在这种情况下，调试器无法访问 CPU，并且无法进行有效的调试。更多详细信息，请参见*第 41 章：调试支持 (DBG)*。

认证调试序列

1. 外部主机请求通过 DBGMCU 访问端口邮箱启动调试认证协议。器件的其他部分保持在复位状态下。
2. SBS 选择 ST-DA（调试认证库）自举地址，并请求从复位状态释放 CPU。
3. 运行 ST-DA 库的 CPU 在系统 Flash 中执行调试认证协议。如果器件关闭，则访问端口邮箱也将关闭，直到 ST-DA 确认认证序列启动请求。
4. 只要从主机输入了正确的密码，就会立即启动完全回退。

注意： *得益于密码认证方法，只能控制完全回退（未重新启动调试）。必须事先使用密码 HASH 配置产品。*

调试复位

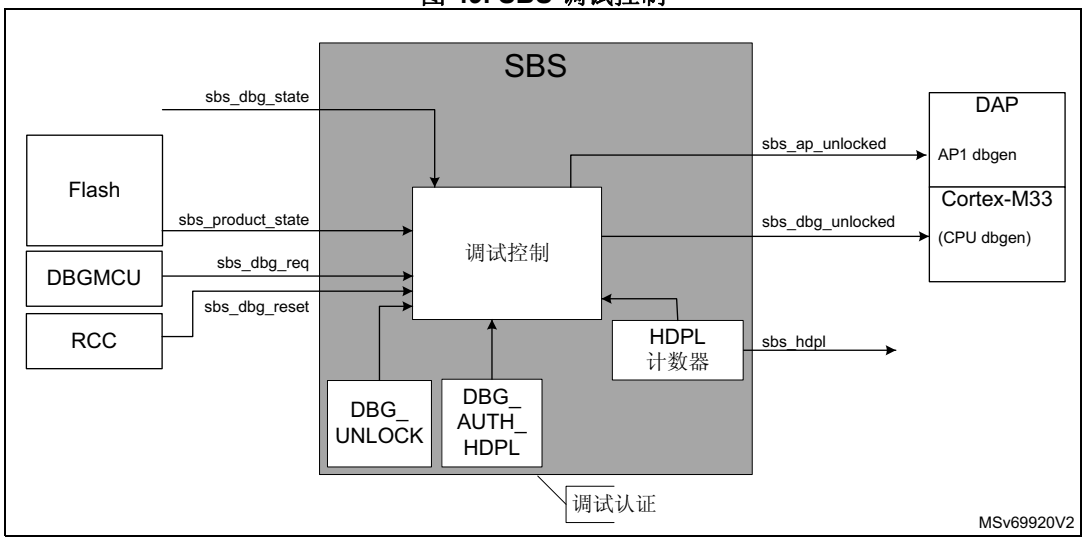
调试启动可能支持通过系统复位或上电复位实现复位的配置，具体取决于 DBGMCU 寄存器位域。来自 RCC 的特定复位引脚 DBGRESETn 用于复位 SBS_DBGCR 寄存器。

调试锁定

通过 SBS_DBGLOCKR 中的 DBGCFG_LOCK，可以锁定调试配置。SBS_DBGCR 随后将不再可写。

当 DBGCFG_LOCK 置 1 时，只能通过硬件 DBGRESETn 复位来复位。

图 49. SBS 调试控制



用于控制调试启动的输入

- sbs_dbg_req：主机请求启动调试认证协议的输入信号。此信号置位后，自举地址将设置为启动调试认证库。
- sbs_product_state：提供当前的 PRODUCT_STATE（来自 Flash）。更多详细信息，请参见*第 7 章：嵌入式 Flash (FLASH)*。
- Sbs_dbg_reset：来自 RCC 的复位信息，如果该寄存器配置为复位，将复位 SBS_DBGCR。

配置

- SBS_DBGCR 中的 DBG_UNLOCK：在达到 DBG_AUTH_HDPL 时解锁调试。
- SBS_DBGCR 中的 AP_UNLOCK：访问端口解锁。
- SBS_DBGCR 中的 DBG_AUTH_HDPL：认证调试时域隔离级别。定义可以启动调试的 HDPL 值。该值只能取 1 到 3 之间的值。
- SBS_DBGLOCKR 中的 DBGC_LOCK：锁定当前的调试配置（仅通过复位释放）。

输出

- sbs_ap_unlocked：用于控制 Cortex-M33 访问端口的信号
  - 1：0xB4
  - 0：所有其他代码
- sbs_dbg_unlocked：解锁 Cortex-M33 的调试（置 1 时）。
  - 1：0xB4
  - 0：所有其他代码
- sbs_dbg_state：将 sbs_product_state 转换为调试状态值（OPEN (0xB4)、CLOSE_SEC (0x8A) 或 CLOSE (0x51)）

表 81. 调试认证逻辑

	输入信号	调试认证控制		输出信号
sbs_dbg_state	**sbs_ hdpl**	**sbs_dbg_ unlock**	**sbs_dbg_ auth_hdpl**	**sbs_dbg_ unlocked**
OPEN	1	x	x	1
	2	x	x	1
	3	x	x	1
CLOSED	x	0	x	0
	1	1	0	0
	1	1	-	1
	1	1	2	0
	2	1	1	1
	2	1	2	1
	2	1	3	0
	3	1	1	1
	3	1	2	1
	3	1	3	1

13.4 SBS 中断

SBS 不支持中断。

13.5 SBS 寄存器

13.5.1 SBS 时域隔离控制寄存器 (SBS_HDPLCR)

SBS temporal isolation control register

偏移地址：0x010

复位值：0x0000 00B4

复位：系统复位

寄存器安全性：无限制

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **INCR_HDPL[7:0]**：递增 HDPL 值 (increment HDPL value)

0xB4：不递增

0x6A：建议值（将 HDPL 级别递增 1）

其他：所有其他值（允许 HDPL 级别递增）

13.5.2 SBS 时域隔离状态寄存器 (SBS_HDPLSR)

SBS temporal isolation status register

偏移地址：0x014

复位值：0xXXXX XXXX

复位值取决于自举情况、HDPL0 或 HDPL1。自举逻辑定义复位后的值。请参见*第 13.3.5 节：SBS 自举控制*。

复位：系统复位

寄存器安全性：无限制

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **HDPL[7:0]**：时域隔离级别 (temporal isolation level)

该位域返回当前的时域隔离级别。

0xB4：HDPL0，为 ST 代码和数据保留的隐藏保护级别

0x51：HDPL1，用于执行和保护不可变信任根 (IROT) 阶段的隐藏保护级别。每次平台复位后最先执行的代码。为确保安全自举，此代码必须不可变

0x8A：HDPL2，用于执行和保护可更新信任根 (UROT) 阶段的隐藏保护级别。从 IROT 开始执行 UROT，从而在自举阶段完成不可变信任根的执行。

0x6F：HDPL3，用于执行应用程序的隐藏保护级别。该值为时域隔离级别，可用于在必须与 IROT 和 UROT 隔离时运行应用程序

13.5.3 SBS 调试控制寄存器 (SBS_DBGCR)

SBS debug control register

偏移地址：0x020

复位值：0x0000 0000

复位：调试复位（系统复位或上电复位）

寄存器安全性：HDPL0/1（其他情况为 RAZ/WI）

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

位 31:24 保留，必须保持复位值。

位 23:16 **DBG_AUTH_HDPL[7:0]**：认证调试时域隔离级别 (authenticated debug temporal isolation level)

写入该位域可定义认证调试以哪个 HDPL 启动。

0x51：HDPL1

0x8A：HDPL2

0x6F：HDPL3

注意：写入任何其他值都将被忽略。读取任何其他值都意味着调试始终不会启动。

位 15:8 **DBG_UNLOCK[7:0]**：在达到 DBG_AUTH_HDPL 时解锁调试 (debug unlock when DBG_AUTH_HDPL is reached)

将 0xB4 写入该位域，可在 SBS_HDPLSR 中的 HDPL 等于此寄存器中的 DBG_AUTH_HDPL 时启动调试。

位 7:0 **AP_UNLOCK[7:0]**：访问端口解锁 (access port unlock)

将 0xB4 写入该位域，可开启器件访问端口。

13.5.4 SBS 调试锁定寄存器 (SBS_DBGLOCKR)

SBS debug lock register

偏移地址：0x024

复位值：0x0000 00B4

复位：调试复位（系统复位或上电复位）

寄存器安全性：HDPL0/1（其他情况为 RAZ/WI）

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **DBGCFG_LOCK[7:0]**：调试配置锁定 (debug configuration lock)

如果位域值不是 0xB4，读取该位域将返回 0x6A。

0xC3 是使用该位域锁定调试配置的推荐值。

0xB4：允许写入 SBS_DBGCR（默认值）

其他值：忽略对 SBS_DBGCR 的写操作

13.5.5 SBS 产品模式和配置寄存器 (SBS_PMCR)

SBS product mode and configuration register

偏移地址：0x100

复位值：0x0000 0000

复位：系统复位

寄存器安全性：无访问限制。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PB8_FMPLUS	PB7_FMPLUS	PB6_FMPLUS
													rw	rw	rw

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

位 31:19 保留，必须保持复位值。

位 18 **PB8_FMPLUS**：PB(8) 上的超快速模式命令 (Fast-mode Plus command on PB(8))

0：禁止 PB(8) 上的超快速模式

1：使能 PB(8) 上的超快速模式

位 17 **PB7_FMPLUS**：PB(7) 上的超快速模式命令 (Fast-mode Plus command on PB(8))

0：禁止 PB(7) 上的超快速模式

1：使能 PB(7) 上的超快速模式

位 16 **PB6_FMPLUS**：PB(6) 上的超快速模式命令 (Fast-mode Plus command on PB(8))
0：禁止 PB(6) 上的超快速模式
1：使能 PB(6) 上的超快速模式

位 15:10 保留，必须保持复位值。

位 9:0 保留，必须保持复位值。

13.5.6 SBS FPU 中断屏蔽寄存器 (SBS_FPUIMR)

SBS FPU interrupt mask register

偏移地址：0x104

复位值：0x0000 001F

复位：系统复位

此寄存器只能通过特权事务进行访问。

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

位 31:6 保留，必须保持复位值。

位 5:0 **FPU_IE[5:0]**：FPU 中断使能 (FPU interrupt enable)
由软件置 1 和清零，用于使能 Cortex-M33 FPU 中断
FPU_IE[5]：不精确的中断使能（复位时禁止中断）
FPU_IE[4]：输入非常规中断使能
FPU_IE[3]：上溢中断使能
FPU_IE[2]：下溢中断使能
FPU_IE[1]：0 分频中断使能
FPU_IE[0]：无效操作中断使能

13.5.7 SBS 存储器擦除状态寄存器 (SBS_MESR)

SBS memory erase status register

偏移地址：0x108

系统复位值：0x000X 000X（位 0 和位 16 不受系统复位的影响）

寄存器安全性：始终可以访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IPMEE
															rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MCLR
															rc_w1

位 31:17 保留，必须保持复位值。

位 16 **IPMEE**：ICACHE 擦除状态 (ICACHE erase status)
当在检测到潜在入侵或产品状态回退后完成 ICACHE 擦除时，该位由硬件置 1（更多详细信息，请参见*第 33 章：入侵和备份寄存器 (TAMP)*）。
该位可通过由软件向其写入 1 的方式来清零。
0：ICACHE 擦除正在进行中
1：ICACHE 擦除已结束

位 15:1 保留，必须保持复位值。

位 0 **MCLR**：器件存储器擦除状态 (device memories erase status)
当在上电复位、检测到入侵或产品状态回退后完成 SRAM2、BKPSRAM 和 ICACHE 擦除时，该位由硬件置 1（更多详细信息，请参见*第 33 章：入侵和备份寄存器 (TAMP)*）。
该位不会因系统复位而复位，而是由软件通过向其写入 1 来清零。
0：如果尚未由软件清零，则存储器擦除正在进行中
1：存储器擦除已完成

13.5.8 SBS I/O 补偿单元控制和状态寄存器 (SBS_CCCSR)

SBS compensation cell for I/Os control and status register

偏移地址：0x110

复位值：0x0000 0000

复位：系统复位

寄存器安全性：始终可以访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	RDY2	RDY1	Res.	Res.	Res.	Res.	CS2	EN2	CS1	EN1
						r	r					rw	rw	rw	rw

位 31:10 保留，必须保持复位值。

位 9 **RDY2**：VDDIO2 补偿单元就绪标志 (VDDIO2 compensation cell ready flag)
该位提供 VDDIO2 补偿单元的状态。
0：VDDIO2 补偿单元未就绪
1：VDDIO2 补偿单元已就绪（可以使用单元提供的代码值）

位 8 **RDY1**：VDDIO 补偿单元就绪标志 (VDDIO compensation cell ready flag)
该位提供补偿单元的状态。
0：VDDIO 补偿单元未就绪
1：VDDIO 补偿单元已就绪（可以使用单元提供的代码值）

位 7:4 保留，必须保持复位值。

位 3 **CS2**：VDDIO2 电源轨的代码选择 (code selection for VDDIO2 power rail)（复位值设为 1）
该位选择 I/O 补偿单元要应用的代码。
0：单元的代码（在 SBS_CCVR 中提供）
1：SBS_CCCR 的代码

位 2 **EN2**：VDDIO2 电源轨补偿单元使能 (enable compensation cell for VDDIO2 power rail)
该位使能 I/O 补偿单元。
0：禁止 I/O 补偿单元
1：使能 I/O 补偿单元

位 1 **CS1**：VDDIO 电源轨的代码选择 (code selection for VDDIO power rail)（复位值设为 1）
该位选择 I/O 补偿单元要应用的代码。
0：单元的代码（在 SBS_CCVR 中提供）
1：SBS_CCCR 的代码

位 0 **EN1**：VDDIO 电源轨补偿单元使能 (enable compensation cell for VDDIO power rail)
该位使能 I/O 补偿单元。
0：禁止 I/O 补偿单元
1：使能 I/O 补偿单元

13.5.9 SBS I/O 补偿单元值寄存器 (SBS_CCVALR)

SBS compensation cell for I/Os value register

偏移地址：0x114

复位值：0x0000 0088

复位：系统复位

寄存器安全性：始终可以访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
APSRC2[3:0]				ANSRC2[3:0]				APSRC1[3:0]				ANSRC1[3:0]			
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 保留，必须保持复位值。

位 15:12 **APSRC2[3:0]**：PMOS 晶体管的补偿值 (compensation value for the PMOS transistor)
该值由补偿单元提供，处理器必须对其进行解析以补偿功能范围内的压摆率。

位 11:8 **ANSRC2[3:0]**：NMOS 晶体管的补偿值 (Compensation value for the NMOS transistor)
该值由补偿单元提供，处理器必须对其进行解析以补偿功能范围内的压摆率。

位 7:4 **APSRC1[3:0]**：PMOS 晶体管的补偿值 (compensation value for the PMOS transistor)
该值由补偿单元提供，处理器必须对其进行解析以补偿功能范围内的压摆率。

位 3:0 **ANSRC1[3:0]**：NMOS 晶体管的补偿值 (compensation value for the NMOS transistor)
该值由补偿单元提供，处理器必须对其进行解析以补偿功能范围内的压摆率。

13.5.10 SBS I/O 补偿单元软件代码寄存器 (SBS_CCSWCR)

SBS compensation cell for I/Os software code register

偏移地址：0x118

复位值：0x0000 7878

复位：系统复位

寄存器安全性：始终可以访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SW_APSRC2[3:0]				SW_ANSRC2[3:0]				SW_APSRC1[3:0]				SW_ANSRC1[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:12 **SW_APSRC2[3:0]**：V_{DDIO} 电源轨的 PMOS 补偿代码 (PMOS compensation code for the V_{DDIO} power rails)

该位域通过软件写入，用于定义 VDDIO 电源轨的 PMOS 晶体管的 I/O 补偿单元代码。SBS_CCSR 中的 CS2 置 1 时，此代码将应用于 I/O。

位 11:8 **SW_ANSRC2[3:0]**：VDDIO 电源轨的 NMOS 补偿代码 (NMOS compensation code for VDDIO power rails)

该位域通过软件写入，用于定义 VDD 电源轨的 NMOS 晶体管的 I/O 补偿单元代码。SBS_CCSR 中的 CS2 置 1 时，此代码将应用于 I/O。

位 7:4 **SW_APSRC1[3:0]**：VDD 电源轨的 PMOS 补偿代码 (PMOS compensation code for the VDD power rails)

该位域通过软件写入，用于定义 VDDIO 电源轨的 PMOS 晶体管的 I/O 补偿单元代码。SBS_CCSR 中的 CS1 置 1 时，此代码将应用于 I/O。

位 3:0 **SW_ANSRC1[3:0]**：VDD 电源轨的 NMOS 补偿代码 (NMOS compensation code for VDD power rails)

该位域通过软件写入，用于定义 VDD 电源轨的 NMOS 晶体管的 I/O 补偿单元代码。SBS_CCSR 中的 CS1 置 1 时，此代码将应用于 I/O。

13.5.11 SBS B 类寄存器 (SBS_CFGR2)

SBS Class B register

偏移地址：0x120

复位值：0x0000 0000

复位：系统复位

寄存器安全性：始终可以访问

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

位 31:4 保留，必须保持复位值。

位 3 **ECCL**：ECC 锁定 (ECC lock)

该位由软件置 1 和清零。它可用于使能并锁定 TIM1 刹车输入的 Flash 双重 ECC 错误信号连接。

0：双重 ECC 错误标志与定时器刹车输入断开连接

1：双重 ECC 错误标志与定时器刹车输入相连

位 2 **PVDL**：PVD 锁定 (PVD lock)

该位由软件置 1，只能通过系统复位清零。它可用于使能并锁定 TIM1 刹车输入的 PVD 连接。

0：PVD 中断与定时器刹车输入断开连接。PWR 寄存器中的 PVD_EN 和 PVD_SEL[2:0] 为读/写位。

1：PVD 中断与定时器刹车输入相连。PWR 寄存器中的 PVD_EN 和 PVD_SEL[2:0] 为只读位。

位 1 **SEL**：SRAM ECC 错误锁定 (SRAM ECC error lock)

该位由软件置 1，只能通过系统复位清零。它可用于使能并锁定 TIM1 刹车输入的 SRAM 双重 ECC 错误信号连接。

0：SRAM 双重 ECC 错误标志与定时器刹车输入断开连接

1：SRAM 双重 ECC 错误标志与定时器刹车输入相连

位 0 **CLL**：内核 lockup 锁定 (core lockup lock)

该位由软件置 1，只能通过系统复位清零。它可用于使能并锁定 TIM1 刹车输入的 Cortex-M33 lockup (HardFault) 输出连接。

0：lockup 输出与定时器刹车输入断开连接

1：lockup 输出与定时器刹车输入相连

13.5.12 SBS CPU 锁定寄存器 (SBS_CNSLCKR)

SBS CPU lock register

偏移地址：0x144

复位值：0x0000 0000

复位：系统复位

寄存器安全性：该寄存器只能通过特权访问进行读写操作。非特权访问为 RAZ/WI。

该寄存器用于锁定 Cortex-M33 MPU 和 VTOR_NS 寄存器的配置。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LOCKNSMPU	LOCKNSVTOR
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **LOCKNSMPU**：MPU 寄存器锁定 (MPU register lock)
该位由软件置 1，只能通过系统复位清零。置 1 时，该位会禁止对 MPU_CTRL_NS、MPU_RNR_NS 和 MPU_RBAR_NS 寄存器的写访问。
0：使能 MPU 寄存器写访问
1：禁止 MPU 寄存器写访问

位 0 **LOCKNSVTOR**：VTOR_NS 寄存器锁定 (VTOR_NS register lock)
该位由软件置 1，只能通过系统复位清零。
0：使能 VTOR_NS 寄存器写访问
1：禁止 VTOR_NS 寄存器写访问

13.5.13 SBS FLITF ECC NMI 屏蔽寄存器 (SBS_ECCNMIR)

SBS flift ECC NMI mask register

偏移地址：0x14C

复位值：0x0000 0000

复位：系统复位

寄存器安全性：此寄存器只能通过特权事务进行访问。

该寄存器设置关于 Flash 双重 ECC 错误的 NMI 预期行为。

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

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

位 31:1 保留，必须保持复位值。

位 0 **ECCNMI_MASK_EN**：FLITF 数据部分发生双重 ECC 错误时的 NMI 行为设置 (NMI behavior setup when a double ECC error occurs on FLITF data part)

0：如果 FLITF 数据部分发生双重 ECC 错误，将生成 NMI

1：如果 FLITF 数据部分发生双重 ECC 错误，不生成 NMI

13.5.14 SBS 寄存器映射

表 82. SBS 寄存器映射和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x010	**SBS_HDPLCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INCR_HDPL[7:0]							
	Reset value																									1	0	1	1	0	1	0	0
0x014	**SBS_HDPLSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HDPL[7:0]							
	Reset value																									x	x	x	x	x	x	x	x
0x020	**SBS_DBGCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_AUTH_HDPL[7:0]								DBG_UNLOCK[7:0]								AP_UNLOCK[7:0]							
	Reset value									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x024	**SBS_DBGLOCKR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBGCFG_LOCK[7:0]							
	Reset value																									1	0	1	1	0	1	0	0
0x028 to 0x030	**Reserved**	Reserved																															
0x038 to 0x09C	**Reserved**	Reserved																															
0x0A4 to 0x0BC	**Reserved**	Reserved																															
0x0C4 to 0x0FC	**Reserved**	Reserved																															
0x100	**SBS_PMCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PB8_FMPLUS	PB7_FMPLUS	PB6_FMPLUS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value														0	0	0																
0x104	**SBS_FPUIMR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FPU_IE[5:0]					
	Reset value																											1	1	1	1	1	1
0x108	**SBS_MESR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IPMEE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MCLR
	Reset value																0																0
0x110	**SBS_CCCSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDY2	RDY1	Res.	Res.	Res.	Res.	CS2	EN2	CS1	EN1
	Reset value																							0	0					0	0	0	0
0x114	**SBS_CCVALR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	APSRC2 [3:0]				ANSRC2 [3:0]				APSRC1 [3:0]				ANSRC1 [3:0]			
	Reset value																	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
0x118	**SBS_CCSWCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SW_APSRC2 [3:0]				SW_ANSRC2 [3:0]				SW_APSRC1 [3:0]				SW_ANSRC1 [3:0]			
	Reset value																	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0
0x120	**SBS_CFGR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCL	PVDL	SEL	CLL
	Reset value																													0	0	0	0

表 82. SBS 寄存器映射和复位值（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x124 to 0x140	**Reserved**	Reserved																															
0x144	**SBS_CNSLCKR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LOCKNSMPU	LOCKNSVTOR
	Reset value																															0	0
0x14C	**SBS_ECCNMIR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCNMI_MASK_EN
	Reset value																																0

有关寄存器边界地址的信息，请参见*第 2.2 节*。

14 外设互连矩阵

14.1 简介

多个外设间有直接连接。这可以在外设间实现自动通信和/或同步，节省了 CPU 资源，进而降低电源功耗。

此外，这些硬件连接避免了软件延迟，允许设计可预测系统。

这些互连可以在多种功耗模式下工作，具体取决于外设：运行、睡眠和停止模式。

14.2 连接汇总

表 83. 外设互连矩阵(1)(2)

源	目标														
	TIM1	TIM2	TIM3	TIM6	TIM7	LPTIM1	LPTIM2	ADC1	OPAMP1	DAC1/2	COMP1	GPDMA	TAMP	RTC	DTS
TIM1	-	1	1	-	-	-	-	2	-	4	10	-	-	-	-
TIM2	1	-	1	-	-	-	-	2	-	4	10	13	-	-	-
TIM3	1	1	-	-	-	-	-	2	-	4	10	-	-	-	-
TIM6	-	-	-	-	-	-	-	2	-	4	-	-	-	-	-
TIM7	-	-	-	-	-	-	-	2	-	4	-	-	-	-	-
LPTIM1	-	-	-	-	-	-	6	2	-	4	10	13	-	-	5
LPTIM2	-	-	-	-	-	6	-	2	-	4	10	13	-	-	5
ADC1	3	-	-	-	-	-	-	-	-	-	-	-	16	-	-
DAC1/2	-	-	-	-	-	-	-	-	14	-	14	-	-	-	-
OPAMP	-	-	-	-	-	-	-	14	-	-	-	-	-	-	-
COMP1	11	11	11	-	-	8	8	-	-	-	-	13	-	-	-
GPDMA1	-	-	-	-	-	8	8	-	-	-	-	13	-	-	-
GPDMA2	-	-	-	-	-	8	8	-	-	-	-	13	-	-	-
EXTI	-	-	-	-	-	-	-	2	-	4	-	13	-	-	5
RTC 唤醒	-	9	-	-	-	-	-	-	-	-	-	13	-	-	-
RTC 闹钟	-	-	-	-	-	8	8	-	-	-	-	13	-	-	-
RTC 日历溢出	-	-	-	-	-	-	-	-	-	-	-	-	16	-	-
TAMP	-	-	-	-	-	8	8	-	-	-	-	13	-	17	-
HSE	-	-	7	-	-	7	-	-	-	-	-	-	-	-	-
LSE	-	7	-	-	-	7	-	-	-	15	-	-	-	-	-
LSE 上的 CSS	-	-	-	-	-	-	-	-	-	-	-	-	16	-	-
HSE 上的 CSS	12	-	-	-	-	-	-	-	-	-	-	-	16	-	-
CSI	7	7	7	-	-	-	7	-	-	-	-	-	-	-	-
HSI	7	7	7	-	-	-	7	-	-	-	-	-	-	-	-
LSI	-	7	-	-	-	7	-	-	-	15	-	-	-	-	-
MCO1	-	7	7	-	-	7	-	-	-	-	-	-	-	-	-
MCO2	-	7	7	-	-	-	7	-	-	-	-	-	-	-	-
VCORE	-	-	-	-	-	-	-	14	-	-	-	-	-	-	-
VREFINT	-	-	-	-	-	-	-	14	-	-	14	-	-	-	-

表 83. 外设互连矩阵(1)(2)（续）

源	目标														
	TIM1	TIM2	TIM3	TIM6	TIM7	LPTIM1	LPTIM2	ADC1	OPAMP1	DAC1/2	COMP1	GPDMA	TAMP	RTC	DTS
T 传感器 (V_{SENSE})	-	-	-	-	-	-	-	*14*	-	-	*14*	-	-	-	-
VBAT	-	-	-	-	-	-	-	*14*	-	-	*14*	-	-	-	-
VBAT 和温度监视器	-	-	-	-	-	-	-	-	-	-	-	-	*16*	-	-
系统错误	*12*	-	-	-	-	-	-	-	-	-	-	-	-	-	-
EVENTOUT	-	-	-	-	-	*8*	*8*	-	-	-	-	*13*	-	-	-

1. 该表中的编号是*第 14.3 节*中相应小节的链接。
2. “-”表示无互连。

14.3 互连详细信息

14.3.1 主从模式的定时器互连

从定时器 (TIM1/TIM2/TIM3) 到定时器 (TIM1/TIM2/TIM3)

目的

某些 TIMx 定时器从内部连接在一起，以实现定时器同步或链接。

当某个定时器配置为主模式时，可对另一个配置为从模式的定时器的计数器执行复位、启动、停止操作或为其提供时钟。

关于功能说明，请参见*第 27.4.23 节：定时器同步*。

关于同步模式的详细信息，请参见：

- *第 26.3.30 节*（面向高级控制定时器 TIM1）
- *第 27.4.22 节*（面向通用定时器 TIM2/TIM3）

触发信号

在可配置定时器事件发生后，输出（来自主器件）出现在 TIMx_TRGO 信号上（对于 TIM1，还出现在 TIMx_TRGO2 信号上）。输入（到从器件）信号在 TIMx_ITR0/ITR1/ITR2 上。

关于可能的主/从连接，请参见：

- *表 168：内部触发连接*（面向高级控制定时器 TIM1）
- *表 191：TIMx 内部触发连接*（面向通用定时器 TIM2/TIM3）

有效功耗模式

定时器可选择在运行模式和睡眠模式下激活。关于低功耗模式对 TIMx 的影响，请参见：

- *表 180：低功耗模式对 TIM1 的影响*
- *表 198：低功耗模式对 TIM2/TIM3 的影响*

14.3.2 ADC 的触发信号

从 EXTI、定时器 (TIM1/TIM2/TIM3/TIM6/TIM7) 和 LP 定时器 (LPTIM1/LPTIM2) 到 ADC1

目的

可通过软件或外部事件（例如定时器输出通道、定时器触发输出或带 EXTI 的输入引脚）触发转换或转换序列。对于 ADC1，如果 EXTEN[1:0] 控制位（对于常规转换）或 JEXTEN[1:0] 位（对于注入转换）不等于 0b00，则外部事件能够以所选极性触发转换。更多详细信息，请参见：

- *第 19.4.18 节：外部触发转换和触发极性（EXTSEL、EXTEN、JEXTSEL、JEXTEN）*
- *ADC 配置寄存器 (ADC_CFGR)* 中定义的 EXTEN[1:0]
- *ADC 注入序列寄存器 (ADC_JSQR)* 中定义的 JEXTEN[1:0]

通用定时器 (TIM2/TIM3)、基本定时器 (TIM6) 和高级控制定时器 (TIM1) 可用于通过定时器输出 tim_oc 和 tim_trgo 生成 ADC 触发事件（关于 TIMx 同步的说明，请参见 *第 26.3.31 节：ADC 触发*（对于 TIM1））

通用定时器 TIM2、基本定时器 (TIM6) 和高级控制定时器 (TIM1) 可用于通过定时器输出 tim_oc 和 tim_trgo 生成 ADC 触发事件。除了通道 11 和 15 上的 EXTI 之外，还可以使用低功耗定时器 (LPTIM1/LPTIM2) 通过 LPTIM 通道生成 ADC 触发事件。

触发信号

对于 ADC1，输入触发信号以及 ADC1 与定时器之间的互连说明如下：

- *表 109：ADC 互连*
- *第 19.4.18 节：外部触发转换和触发极性（EXTSEL、EXTEN、JEXTSEL、JEXTEN）*
- *第 19.4.25 节：时序图示例（单次模式/连续模式，硬件/软件触发）*

有效功耗模式

该互连在运行模式和睡眠模式下激活，假定其触发事件线也处于激活状态。定时器仅在运行模式和睡眠模式下激活。关于低功耗模式的影响，请参见：

- *表 180：低功耗模式对 TIM1 的影响*
- *表 198：低功耗模式对 TIM2/TIM3 的影响*
- *表 219：低功耗模式对 LPTIM 的影响*

14.3.3 ADC 模拟看门狗作为定时器的触发器

从 ADC1 到 TIM1

目的

来自 ADC1 的内部模拟看门狗输出信号连接到片上定时器。ADC1 可通过模拟看门狗信号向高级控制定时器 (TIM1) 提供触发事件，从而复位、启动、停止计数器或为其提供时钟。

关于 ADC 模拟看门狗和定时器触发设置的说明，请参见：

- *第 26.3.6 节：外部触发输入*（对于 TIM1）
- *表 169：tim_etr 输入复用器的互连*（对于连接到 TIM1 (tim_etr) 输入复用器的内部 ADC1 源）
- *第 19.4.28 节：模拟窗口看门狗（AWD1EN、JAWD1EN、AWD1SGL、AWD1CH、AWD2CH、AWD3CH、AWD_HTx、AWD_LTx、AWDx）*（对于 ADC1/ADC_AWDx 信号输出生成）

触发信号

（来自 ADC 的）输出在信号 ADC_AWDx 上，其中 x = 1、2、3（每个 ADC 有三个看门狗）。（到定时器的）输入在信号 TIMx_ETR（外部触发）上。

有效功耗模式

ADC1 在运行模式和睡眠模式下激活。

14.3.4 DAC 的触发器

从定时器 (TIM1/TIM2/TIM3/TIM6/TIM7)、低功耗定时器 (LPTIM1/LPTIM2) 和 EXTI 到 DAC (DAC1/DAC2)

目的

通用定时器 (TIM2/TIM3)、基本定时器 (TIM6/TIM7)、高级控制定时器 (TIM1)、LP 定时器 (LPTIM1/LPTIM2) 输出通道 (lptim1_ch1 和 lptim2_ch1)和 EXTI 可用作启动 DAC 转换的触发事件。

触发信号

TIMx_TRGO 信号的输出（来自定时器）和来自 LP 定时器的输出直接连接到相应的 DAC 输入。

关于 DAC 上的输入触发选择，请参见：

- *表 131：DAC 互连*
- *第 21.4.8 节：DAC 触发选择*

有效功耗模式

此互连在运行模式、睡眠模式和停止模式下激活。

14.3.5 DTS 的触发器

从 EXTI 和 LP 定时器 (LPTIM1/LPTIM2) 到 DTS。

目的

温度测量可通过软件、LP 定时器或外部事件触发。

触发信号

关于输入触发信号以及 DTS 与 LP 定时器之间的互连说明，请参见*表 124：触发配置*。

14.3.6 两个 LP 定时器之间的连接

从 LPTIM1 到 LPTIM2，从 LPTIM2 到 LPTIM1。

目的

LPTIM1 输出通道可用作时钟源，通过 lptim2_in1_mux3 输入递增 LPTIM2 计数器。

LPTIM2 输出通道可用作时钟源，通过 lptim1_in1_mux3 输入递增 LPTIM1 计数器。

触发信号

LPTIM1 输出 lptim1_ch2 是 lptim2_in1_mux3 的输入。

LPTIM2 输出 lptim2_ch2 是 lptim1_in1_mux3 的输入。

表 212：LPTIM1/2 输入 1 连接 给出了 LPTIM1/2 输入 1 连接。

有效功耗模式

该功能在运行模式、睡眠模式和停止模式下可用。

14.3.7 定时器的内部时钟源

从 HSE、LSE、LSI、CSI、HSI 和 MCO1/MCO2 到定时器 (TIM2/TIM3) 和 LP 定时器 (LPTIM1/LPTIM2)

目的

定时器输入或定时器计数器可接收不同的时钟源，并可用于校准参考时钟上的内部振荡器等。

外部时钟（HSE、LSE）、内部时钟（LSI、CSI、HSI）和微控制器输出时钟 (MCO1 和 MCO2) 可用作定时器的输入：

- HSE、LSE、CSI、HSI 和 LSI 作为外部输入信号分配给通用定时器 TIM2/TIM3。CSI/HSI/LSI 可选作由 mode1 外部时钟源（tim_ti1 或 tim_ti2 信号）和 mode2 外部时钟源（外部触发输入 tim_etr_in）提供的计数器时钟。关于输入分配和时钟选择的详细说明，请参见：
  - *第 27.4.5 节：时钟选择*（对于 TIM2/TIM3）
  - 外部时钟 mode1：*表 187：tim_ti1 输入复用器的互连*（对于 tim_ti1_in1 (LSI-TIM2)、tim_ti1_in2 (LSE-TIM2)、tim_ti1_in4 (RCC_HSE_1MHz-TIM3)）。*表 188：tim_ti2 输入复用器的互连*（对于 tim_ti1_in1 (HSI/1024 -TIM2)、tim_ti1_in2 (CSI/128 -TIM2)、tim_ti1_in3 (HSI/1024 -TIM3)、tim_ti1_in1 (CSI/128 -TIM3)）
  - 外部时钟 mode2：*表 192：tim_etr 输入复用器的互连*中的 tim_etr3 (LSE)（对于 TIM2）
- 微控制器输出时钟 (MCO1/MCO2) 作为外部输入连接到通用定时器 TIM2/TIM3。这可以校准 HSI/CSI 系统时钟（使用 TIM2 和 LSE）或 LSI（使用 TIM3 和 HSE）。也可用于精确测量 LSI 或 CSI（使用 TIM3 和 HSI）振荡器频率。使用低速外部 (LSE) 振荡器时，无需额外的硬件连接。
- RCC_HSE_1MHz、LSI 和 LSE 可选作 LPTIM1 的输入捕获 2，如*表 215：LPTIM1/2 输入捕获 2 连接*所述。
- MCO1 和 MCO2 可分别选作 LPTIM1 和 LPTIM2 的输入捕获 1，如*表 212：LPTIM1/2 输入 1 连接*所述。
- HSI/1024、CSI/128 和 HSI/8 可选作 LPTIM2 的输入捕获 2，如*表 215：LPTIM1/2 输入捕获 2 连接*所述。

触发信号

可通过设置 LPTIM 配置寄存器 2 (LPTIM_CFGR2) 来选择 lptim_ic2_mux1 LPTIM 输入捕获。对于定时器，内部时钟信号可选作由 mode1 外部时钟源 (tim_ti1_in) 和 mode2 外部时钟源（外部触发输入 tim_etr_in）提供的计数器时钟。

有效功耗模式

该功能在运行模式和睡眠模式下可用。

14.3.8 低功耗定时器的触发器

从比较器 (COMP1)、CPU (EVENTOUT)、GPDMA1/2、TAMP 和 RTC 闹钟到 LP 定时器 (LPTIM1/LPTIM2)

目的

LPTIM1/LPTIM2 计数器可通过软件启动，也可以在八个触发输入之一上检测到有效边沿后启动（请参见*第 29.4.7 节：触发多路复用器*）。

RTC 闹钟 A/B、TAMP 输入检测、COMP1_OUT、GPDMA 传输完成和 CPU EVENTOUT 可用作启动 LPTIM 计数器 (LPTIM1/2) 的触发信号。

触发信号

关于功能说明，请参见后续章节。关于输入选择的说明，请参见*表 211：LPTIM1/2 外部触发连接*。

有效功耗模式

该互连在低至停止模式下保持激活状态。

14.3.9 RTC 唤醒信号作为定时器的输入

从 RTC 到定时器 (TIM2)

目的

RTC 唤醒中断可用作通用定时器 (TIM2) 通道 1 的输入。

触发信号

对于 TIM2，RTC 唤醒信号连接至 tim_ti1_in3 信号，如*表 187：tim_ti1 输入复用器的互连*所述。

有效功耗模式

该互连在低至停止模式下处于激活状态。计时器未激活，但在唤醒后会执行计数。

14.3.10 比较器的消隐源

从定时器 (TIM1/TIM2/TIM3) 到比较器 COMP1

目的

高级控制定时器 TIM1 和通用定时器 (TIM2/TIM3) 可用作 COMP1 的消隐窗口输入。关于消隐功能的说明，请参见*第 22.3.6 节：比较器输出消隐功能*。

关于消隐源，请参见*比较器配置寄存器 1 (COMP_CFGR1)* 中的 BLANKING[3:0]。

触发信号

定时器输出信号 TIMx_Ocx 是 COMP1 的消隐源输入。

有效功耗模式

该功能在运行模式和睡眠模式下可用。

14.3.11 比较器作为定时器的输入、触发或刹车信号

从比较器 COMP1 到定时器 (TIM1/TIM2/TIM3)

目的

比较器 (COMP1) 输出可以连接到定时器 (TIM1/TIM2/TIM3) 输入捕获、TIMx_ETR 或定时器刹车信号。*第 26.3.6 节：外部触发输入*对与 ETR 的连接进行了说明。

比较器 (COMP1) 输出值也可生成定时器（例如 TIM1）的刹车输入信号。刹车 (tim_brk) 通道的来源为以下两种之一：

- 外部：连接到 TIMx_BKIN 引脚之一（由 AFIO 控制器选择），具有极性选择和可选的数字滤波
- 内部：来自比较器、tim_brk_cmpx 输入（有关产品特定的实现，请参见*第 26.3.2 节：TIM1 引脚和内部信号*）。

触发信号

连接 TIM1（来自 COMP1）的 tim_etr 和 tim_brk 信号如下：

- tim_etr（*表 169：tim_etr 输入复用器的互连*）：外部触发内部输入总线
  这些输入可用作触发、外部时钟或用于硬件逐周期脉宽控制。
- tim_brk（*表 170：定时器刹车互连*和*表 171：定时器刹车 2 互连*）
- *第 26.3.6 节：外部触发输入*
- *第 26.3.18 节：使用刹车功能*

对于 TIM2/TIM3，连接到 tim_ti[1..4] 输入复用器的触发源（来自比较器和一些其他外设）如下：

- *表 187：tim_ti1 输入复用器的互连*
- *表 188：tim_ti2 输入复用器的互连*
- *表 189：tim_ti3 输入复用器的互连*
- *表 190：tim_ti4 输入复用器的互连*
- *表 192：tim_etr 输入复用器的互连*
- *第 27.4.22 节：定时器与外部触发同步*
- *第 27.5.26 节：TIMx 定时器输入选择寄存器 (TIMx_TISEL)（x = 2、3）*

有效功耗模式

触发源在停止模式下具有运行、睡眠和唤醒功能。在运行和睡眠模式下，输入和刹车在与定时器活动所处的相同低功耗模式下保持激活状态。

14.3.12 系统错误作为定时器的刹车信号

从系统错误到定时器。

目的

CSS、CPU 自锁、SRAM1/SRAM2 ECC 双重错误、FLASH ECC 双重错误检测和 PVD 可向定时器 TIM1 以定时器刹车形式生成系统错误。

刹车功能的目的是保护由这些定时器生成的 PWM 信号所驱动的功率器件。

更多详细信息，请参见*第 13.5.11 节：SBS B 类寄存器 (SBS_CFGR2)*。

触发信号

关于可能的刹车源，请参见：

- *第 26.3.18 节：使用刹车功能*（对于 TIM1）
- *表 172：系统刹车互连*（对于 TIM1）

有效功耗模式

定时器可选择在运行模式和睡眠模式下激活。关于低功耗模式对 TIMx 的影响，请参见：

- *表 180：低功耗模式对 TIM1 的影响*
- *表 198：低功耗模式对 TIM2/TIM3 的影响*

14.3.13 GPDMA1/2 的触发信号

从 EXTI、RTC（闹钟/唤醒）、TAMP、定时器 (TIM2)、比较器 (COMP1)、GPDMA1/2 传输完成 (gpdma1_chx_tcf/gpdma2_chx_tcf) 到 GPDMA1/2。

目的

GPDMA 触发信号可分配给 GPDMA 通道 x。编程设定的 GPDMA 传输可以由所选输入触发事件的上升沿/下降沿触发。触发模式也可以编程为 LLI 链路传输的条件。更多详细信息，请参见以下章节：

- *第 15.3.7 节：GPDMA 触发信号*
- *第 15.4.12 节：GPDMA 触发传输*
- *GPDMA 通道 x 传输寄存器 2 (GPDMA_CxTR2)*，其中包含有关以下内容的更多详细信息：
  - 触发选择 TRIGSEL[5:0] 位域
  - 由 TRIGM[1:0] 定义的触发模式 (LLI)
  - 由 TRIGPOL[1:0] 定义的触发极性

触发信号

*表 86：编程设定的 GPDMA1/2 请求*中指定了 GPDMA 触发映射（依据 GPDMA_CxTR2.TRIGSEL[5:0]）。

有效功耗模式

假设触发源在低至睡眠模式下处于激活状态，则该互连在睡眠模式下仍可正常工作。

请参见：

- *第 15.6 节：低功耗模式下的 GPDMA*

14.3.14 模拟外设的内部模拟信号

从内部模拟源到 ADC1、比较器 COMP1 和 OPAMP1

目的

内部参考电压 (VREFINT)、内部温度传感器 (VSENSE) 和 VBAT 监视通道连接到 ADC1 输入通道。此外，内部数字内核电压 (VCORE) 连接到 ADC1 输入通道。

DAC 通道 (DAC1_OUT/DAC2_OUT)、VREFINT、VSENSE 和 VBAT 监视通道连接到比较器 (COMP1)。

OPAMP1 输出可通过 GPIO 连接到 ADC1 输入通道。DAC1_OUT1 可连接到 OPAMP1_VINP。

请参见*表 130：DAC 内部输入/输出信号*

- dac_out1 模拟输出 DAC 通道 1，输出（对于片上外设）
- dac_out2 模拟输出 DAC 通道 2，输出（对于片上外设）

其依据包括：

- *第 33.2 节*和*第 19.2 节：ADC 主要特性*
- *第 19.4.11 节：通道选择（SQRx、JSQRx）*
- *表 148：OPAMP 的可能连接*
- *第 22.3.2 节：COMP 引脚和内部信号*

有效功耗模式

在停止模式下，如果所选外设保持激活，则这些互连保持激活。请参见：

- *第 22.4 节：COMP 低功耗模式*
- *第 23.4 节：OPAMP 低功耗模式*
- *第 21.5 节：低功耗模式下的 DAC*

14.3.15 DAC 采样和保持模式下的时钟源

从 LSI/LSE 到 DAC1/DAC2

目的

DAC1/DAC2 可在停止模式下运行。采样和保持模块及其相关寄存器在停止模式下使用 LSI 或 LSE 时钟源 (dac_hold_ck)。

请参见*表 130：DAC 内部输入/输出信号*：

dac_hold_ck，输入，在采样和保持模式下使用的 DAC 低功耗时钟

有效功耗模式

该功能在低至停止模式下仍然可用。

14.3.16 内部入侵源

从内部外设、时钟或监视到入侵。

目的

为检测任何异常活动或破坏器件的意图，引入了入侵检测并适时提醒系统存在此类意外事件。采取不同的操作可能会产生不同的后果。

- 当电池电压高于功能范围时，备用域电压监视器可以生成入侵事件。更多详细信息，请参见*第 9.6.6 节：备份域电压监视*。
- 当温度高于或低于功能范围时，温度监视器可以生成入侵事件。更多详细信息，请参见*第 9.6.7 节：温度监视*。
- LSE 上的时钟安全系统可以在检测到 LSE 故障时（LSE 消失或发生过频时）生成入侵事件。更多详细信息，请参见*第 10.4.10 节：时钟安全系统 (CSS)*。
- HSE 上的时钟安全系统可以在检测到 HSE 故障时（HSE 消失或发生过频时）生成入侵事件。更多详细信息，请参见*第 10.4.10 节：时钟安全系统 (CSS)*。

关于入侵源列表，请参见*表 242：TAMP 互连*。

有效功耗模式

如果入侵源激活，该互连在所有功耗模式下均激活。

14.3.17 对 RTC 的入侵输出

从 TAMP 到 RTC

目的

RTC 可为入侵事件添加时间戳，以便在检测到此类事件时检索历史记录。RTC 还可控制 GPIO，并根据 MCU 之外的入侵或闹钟状态设置信号。

更多详细信息，请参见*第 32.3.3 节：RTC 和 TAMP 控制的 GPIO*。

有效功耗模式

该互连在所有功耗模式下均保持激活。

15 通用直接存储器访问控制器 (GPDMA)

15.1 GPDMA 简介

通用直接存储器访问 (GPDMA) 控制器是一个主控总线系统外设。

当通过控制为 CPU 减负时，可以使用 GPDMA 通过链表在存储器映射的外设和/或存储器之间执行可编程的数据传输。

15.2 GPDMA 主要特性

- 双路双向 AHB 主控
- 从源到目标进行存储器映射的数据传输：
  - 外设到存储器
  - 存储器到外设
  - 存储器到存储器
  - 外设到外设
- 低功耗模式期间进行自主数据传输（请参见*第 15.3.3 节*）

  在通道级基于 4 级编程设定的优先级进行传输仲裁：
  - 一个高优先级数据传输类，用于时间敏感型通道（队列 3）
  - 三个低优先级数据传输类，可进行加权循环调度分配，用于非时间敏感型通道（队列 0、1、2）
- 在发生以下任意事件时，按通道生成事件：传输完成、半传输完成、数据传输错误、用户设置错误、链路传输错误、完成挂起和触发上溢
- 每个通道生成中断，并且每个事件单独设定了中断使能
- 8 个并发 GPDMA 通道：
  - 对于源和目标传输队列，每个通道都遵循 FIFO 原则（请参见*第 15.3.2 节*）
  - 通道内 GPDMA 通过存储器中的可编程链表将传输链接起来，并支持两种执行模式：运行至完成模式和链接步进模式
  - 通道内和通道间 GPDMA 传输通过与 GPDMA 任务完成事件关联的可编程 GPDMA 输入触发信号来链接起来
- 通道内每个链表项：
  - 单独编程设定源和目标传输
  - 源和目标之间的可编程数据处理：基于字节进行重新排序、封装或解封、填充或截断、符号扩展和左/右重新对齐
  - 可编程要从源传输的数据字节数，以定义块级别
  - 源和目标线性寻址：在连续突发传输之间进行固定寻址或连续增量式寻址（在块级别设定）
  - 2D 源和目标寻址：连续突发传输之间的可编程有符号地址偏移（在第二个 2D/重复的块级别进行块内非连续寻址，并结合连续块之间的可编程有符号地址偏移，适用于通道数量减少的情况）（请参见*第 15.3.2 节*）
  - 支持分散收集（多缓冲区传输），通过 2D 寻址实现数据交错和去交错
  - 可编程设定 GPDMA 请求和触发信号选择

  - 可编程设定 GPDMA 半传输和传输完成事件生成
  - 指向存储器中下一个链表项及其数据结构的指针，并自动更新 GPDMA 链表控制寄存器
- 调试：
  - 支持通道挂起和恢复
  - 报告通道状态，包括 FIFO 级别和事件标志
- 支持特权/非特权传输：
  - 在通道级单独支持特权和非特权 GPDMA 传输
  - 特权感知 AHB 从设备端口

15.3 GPDMA 实现

15.3.1 GPDMA 实例

器件中有两个 GPDMA 实例，分别命名为 GPDMA1 和 GPDMA2。

两个 GPDMA 实例以相同的方式基于通道实现，并连接到相同的请求和触发信号，以下小节对此进行了详细介绍。

15.3.2 GPDMA 通道

指定的 GPDMA 通道 x 实现了以下特性和预期用途。为了发挥 GPDMA 的最佳性能，可参考下表提供的一些常规建议，用户可以根据通道实现的 FIFO 大小和请求的 GPDMA 传输，来选择和分配通道。

表 84. GPDMA1/2 通道实现

通道 x	硬件参数		特性
	dma_fifo_size[x]	dma_addressing[x]	
x = 0 到 3	2	0	通道 x（x = 0 到 3）实现了以下特性： – FIFO 大小为 8 个字节，2 个字 – 固定寻址/连续增量式寻址 通常情况下，在 APB 或 AHB 外设与 SRAM 之间进行 GPDMA 传输时必须分配这些通道。
x = 4、5	4	0	通道 x（x = 4、5）实现了以下特性： – FIFO 大小为 32 个字节，8 个字 – 固定寻址/连续增量式寻址 这些通道可用于请求的 AHB 外设与 SRAM 之间进行 GPDMA 传输，或者与外部存储器之间的双向传输。
x = 6、7	4	1	通道 x（x = 6、7）实现了以下特性： – FIFO 大小为 32 个字节，8 个字 – 2D 寻址 这些通道可用于请求的 AHB 外设与 SRAM 之间进行 GPDMA 传输，或者与外部存储器之间的双向传输。

15.3.3 低功耗模式下的 GPDMA 自主模式

在器件低功耗模式下实现了 GPDMA 自主模式和唤醒功能，如下表所示。

表 85. 低功耗模式下的 GPDMA1/2 自主模式和唤醒功能

功能	低功耗模式
自主模式和唤醒	GPDMA1/2 处于睡眠模式

15.3.4 GPDMA 请求

当控制位 SWREQ = 0 时，来自外设的 GPDMA 请求可以通过 GPDMA_CxTR2 中的 REQSEL[7:0] 分配给 GPDMA 通道 x。

下表列出了规定的 GPDMA 请求映射。

表 86. 编程设定的 GPDMA1/2 请求

GPDMA_CxTR2.REQSEL[7:0]	所选 GPDMA 请求
0	adc1_dma
1	保留
2	dac1_ch1_dma
3	dac1_ch2_dma
4	tim6_upd_dma
5	tim7_upd_dma
6	spi1_rx_dma
7	spi1_tx_dma
8	spi2_rx_dma
9	spi2_tx_dma
10	spi3_rx_dma
11	spi3_tx_dma
12	i2c1_rx_dma
13	i2c1_tx_dma
14	保留
15	i2c2_rx_dma
16	i2c2_tx_dma
17	保留
18	保留
19	保留
20	保留
21	usart1_rx_dma
22	usart1_tx_dma

表 86. 编程设定的 GPDMA1/2 请求（续）

GPDMA_CxTR2.REQSEL[7:0]	所选 GPDMA 请求
23	usart2_rx_dma
24	usart2_tx_dma
25	usart3_rx_dma
26	usart3_tx_dma
27	保留
28	保留
29	保留
30	保留
31	保留
32	保留
33	保留
34	保留
35	保留
36	保留
37	保留
38	保留
39	保留
40	保留
41	保留
42	保留
43	保留
44	保留
45	lpuart1_rx_dma
46	lpuart1_tx_dma
47	保留
48	保留
49	保留
50	保留
51	保留
52	保留
53	保留
54	保留
55	保留
56	保留
57	保留

表 86. 编程设定的 GPDMA1/2 请求（续）

GPDMA_CxTR2.REQSEL[7:0]	所选 GPDMA 请求
58	tim1_cc1_dma
59	tim1_cc2_dma
60	tim1_cc3_dma
61	tim1_cc4_dma
62	tim1_upd_dma
63	tim1_trg_dma
64	tim1_com_dma
65	保留
66	保留
67	保留
68	保留
69	保留
70	保留
71	保留
72	tim2_cc1_dma
73	tim2_cc2_dma
74	tim2_cc3_dma
75	tim2_cc4_dma
76	tim2_upd_dma
77	tim3_cc1_dma
78	tim3_cc2_dma
79	tim3_cc3_dma
80	tim3_cc4_dma
81	tim3_upd_dma
82	tim3_trg_dma
83	保留
84	保留
85	保留
86	保留
87	保留
88	保留
89	保留
90	保留
91	保留
92	保留

表 86. 编程设定的 GPDMA1/2 请求（续）

GPDMA_CxTR2.REQSEL[7:0]	所选 GPDMA 请求
93	保留
94	保留
95	保留
96	保留
97	保留
98	保留
99	保留
100	保留
101	保留
102	lptim1_ic1_dma
103	lptim1_ic2_dma
104	lptim1_ue_dma
105	lptim2_ic1_dma
106	lptim2_ic2_dma
107	lptim2_ue_dma
108	保留
109	保留
110	保留
111	hash_in_dma
112	保留
113	保留
114	保留
115	保留
116	保留
117	保留
118	保留
119	保留
120	i3c1_rx_dma
121	i3c1_tx_dma
122	i3c1_tc_dma
123	i3c1_rs_dma
124	保留
125	保留
126	保留
127	保留

表 86. 编程设定的 GPDMA1/2 请求（续）

GPDMA_CxTR2.REQSEL[7:0]	所选 GPDMA 请求
128	保留
129	保留
130	保留
131	保留
132	保留
133	保留
134	保留
135	保留
136	i3c2_rx
137	i3c2_tx
138	i3c2_tc
139	i3c2_rs

15.3.5 GPDMA 块请求

有些 GPDMA 请求必须设定为块请求，而不能设定为突发请求。然后，GPDMA_CxTR2 中的 BREQ 必须置 1，这样 GPDMA 才能在硬件层面正确执行所请求的外设传输。

表 87. 编程设定为块请求的 GPDMA1/2 请求

GPDMA 块请求
lptim1_ue_dma
lptim2_ue_dma

15.3.6 支持外设提前终止的 GPDMA 通道

如果 GPDMA 通道支持外设提前终止功能，则可以提前终止来自同样支持此功能的外设的数据传输。

表 88. 支持外设提前终止的 GPDMA1/2 通道

支持外设提前终止的 GPDMA 通道 x
x = 0 和 x = 7

当使用 GPDMA_CxTR2.PFREQ = 1 对通道 x 进行编程时，将激活此 GPDMA 支持。然后，在 GPDMA 传输整个块之前，外设自身可以启动并请求数据传输完成（更多详细信息，请参见*第 15.4.14 节*）。

表 89. 编程设定为支持外设提前终止的 GPDMA1/2 请求

编程设定为支持外设提前终止的 GPDMA 通道 x 请求
i3c1_rx_dma
i3c2_rx_dma

15.3.7 GPDMA 触发信号

GPDMA 触发信号可以通过 GPDMA_CxTR2 中的 TRIGSEL[5:0] 分配给 GPDMA 通道 x，前提是 TRIGPOL[1:0] 定义了所选触发信号的上升沿或下降沿（TRIGPOL[1:0] = 01 或 TRIGPOL[1:0] = 10）。

表 90.编程设定的 GPDMA1/2 触发信号

GPDMA_CxTR2.TRIGSEL[5:0]	所选 GPDMA 触发信号
0	exti0
1	exti1
2	exti2
3	exti3
4	exti4
5	exti5
6	exti6
7	exti7
8	tamp_trg1
9	tamp_trg2
10	保留
11	lptim1_ch1
12	lptim1_ch2
13	lptim2_ch1
14	lptim2_ch2

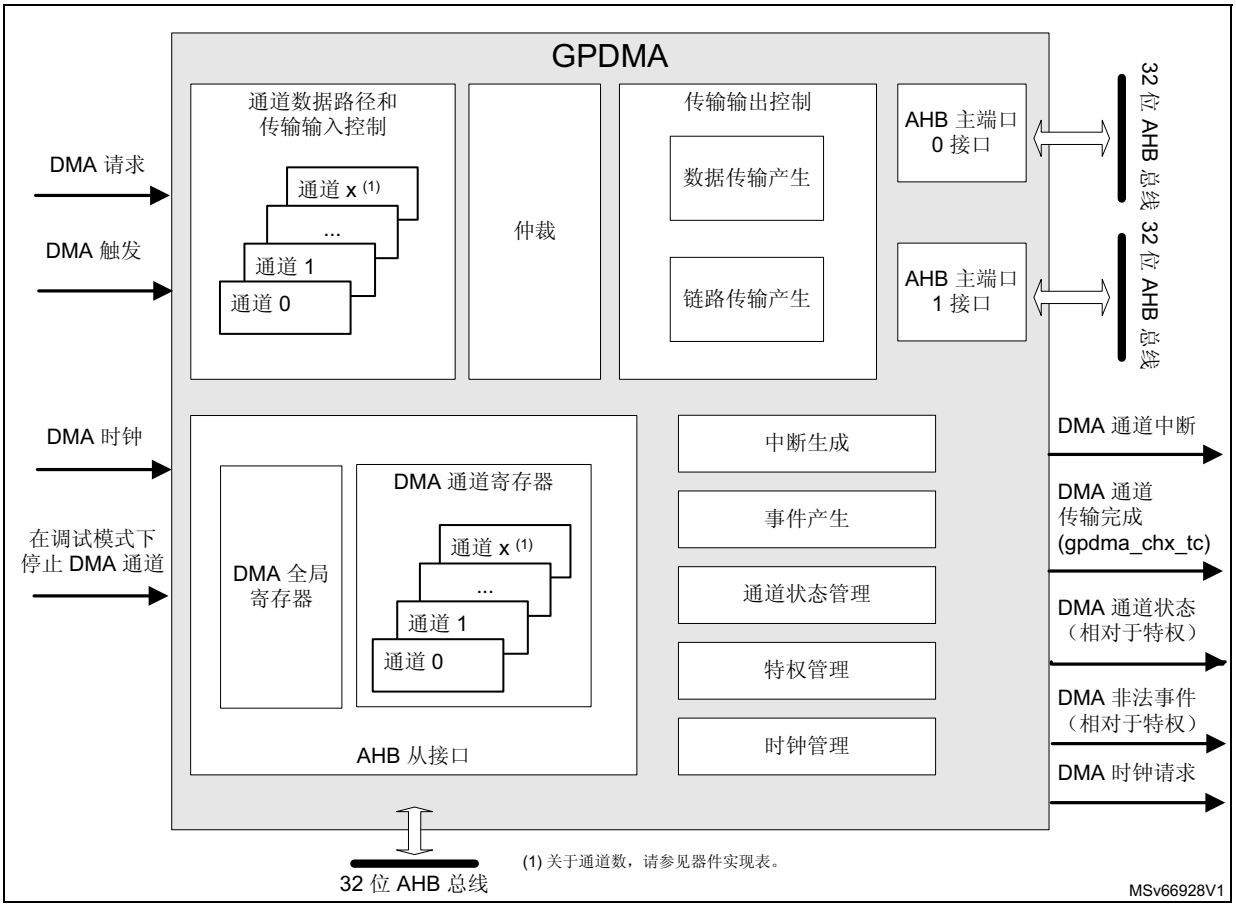
表 90.编程设定的 GPDMA1/2 触发信号（续）

GPDMA_CxTR2.TRIGSEL[5:0]	所选 GPDMA 触发信号
15	rtc_alra_trg
16	rtc_alrb_trg
17	rtc_wut_trg
18	gpdma1_ch0_tc
19	gpdma1_ch1_tc
20	gpdma1_ch2_tc
21	gpdma1_ch3_tc
22	gpdma1_ch4_tc
23	gpdma1_ch5_tc
24	gpdma1_ch6_tc
25	gpdma1_ch7_tc
26	gpdma2_ch0_tc
27	gpdma2_ch1_tc
28	gpdma2_ch2_tc
29	gpdma2_ch3_tc
30	gpdma2_ch4_tc
31	gpdma2_ch5_tc
32	gpdma2_ch6_tc
33	gpdma2_ch7_tc
34	tim2_trgo
35	保留
36	保留
37	保留
38	保留
39	保留
40	保留
41	保留
42	保留
43	保留
44	comp1_out
45	eventout

15.4 GPDMA 功能说明

15.4.1 GPDMA 框图

图 50. GPDMA 框图



注意： *STM32H503xx 器件上不会生成 DMA 非法事件。*

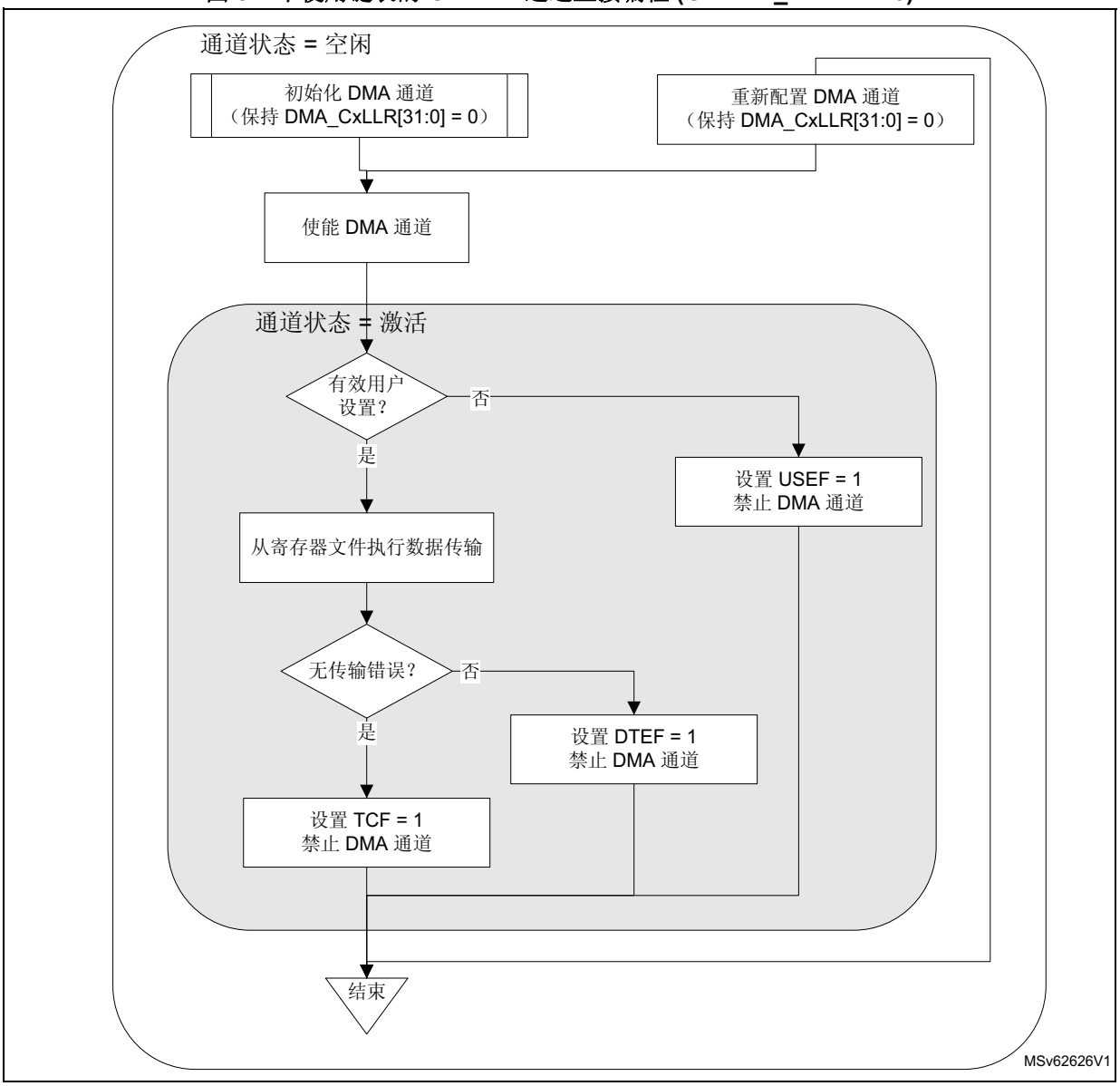
15.4.2 GPDMA 通道状态和直接编程（无任何链表）

GPDMA 复位后，GPDMA 通道 x 处于空闲状态。当软件在 GPDMA_CxCR.EN 中写入 1 后，通道可接受通道配置寄存器 (GPDMA_CxXXX) 的值，切换到活动/非空闲状态，并开始执行相应的已请求数据传输。

通过在 GPDMA_CxCR.EN 中写入 1 使能/启动 GPDMA 通道传输后，传输完成时 GPDMA 通道中断会向软件发出通知，指示 GPDMA 通道返回到空闲状态（然后 EN 位通过硬件禁止），并且通道已准备好重新配置然后再次使能。

下图说明了这种不使用任何链表的 GPDMA 直接编程 (GPDMA_CxLLR = 0)。

图 51. 不使用链表的 GPDMA 通道直接编程 (GPDMA_CxLLR = 0)



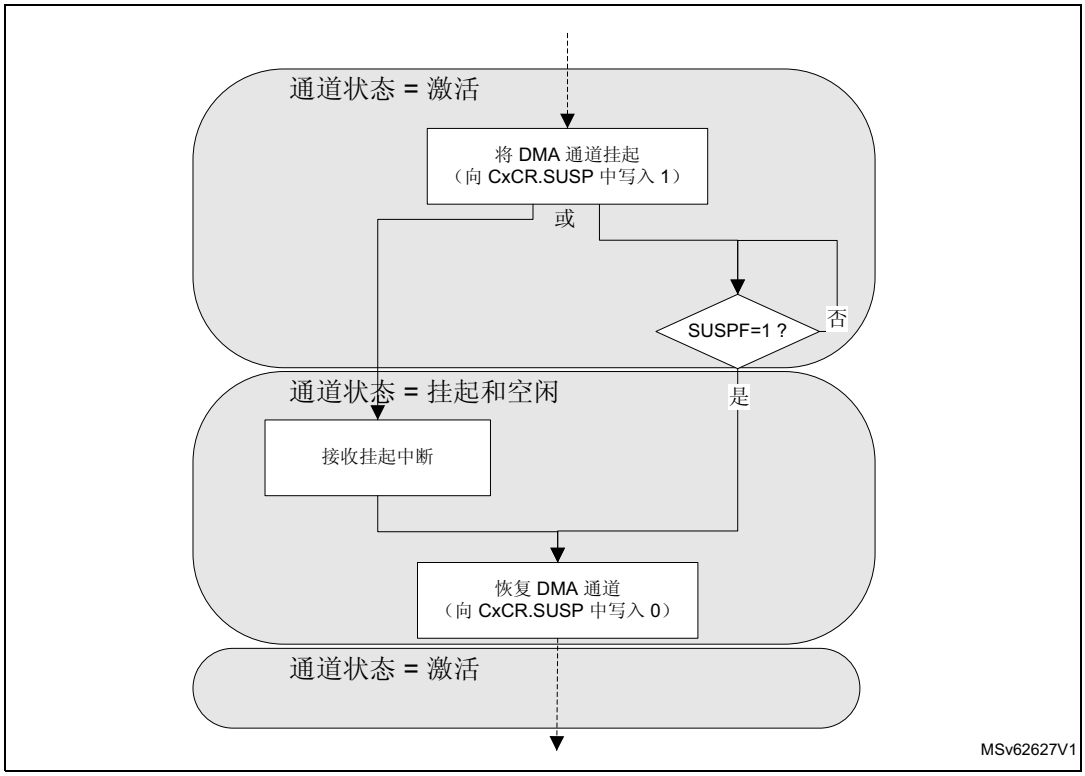
15.4.3 GPDMA 通道挂起和恢复

软件可以按以下顺序自行将仍处于活动状态的通道挂起：

1. 软件将 1 写入 GPDMA_CxCR.SUSP 位。
2. 软件轮询挂起标志 GPDMA_CxSR.SUSPF 直至 SUSPF = 1，或者等待之前通过将 1 写入 GPDMA_CxCR.SUSPIE 使能的中断。等待挂起状态的通道生效，意味着等待任何正在通过其主设备端口执行的 GPDMA 传输完成。然后软件可以在稳定状态下识别到任何可由硬件修改的读寄存器或寄存器位域。

   注意，进行中的 GPDMA 传输可以是数据传输（源/目标突发传输），也可以是链路传输，用于从下一个链表项对链表寄存器文件进行内部更新。
3. 软件将 0 写入 GPDMA_CxCR.SUSP，安全地恢复挂起的通道。

图 52. GPDMA 通道挂起和恢复序列



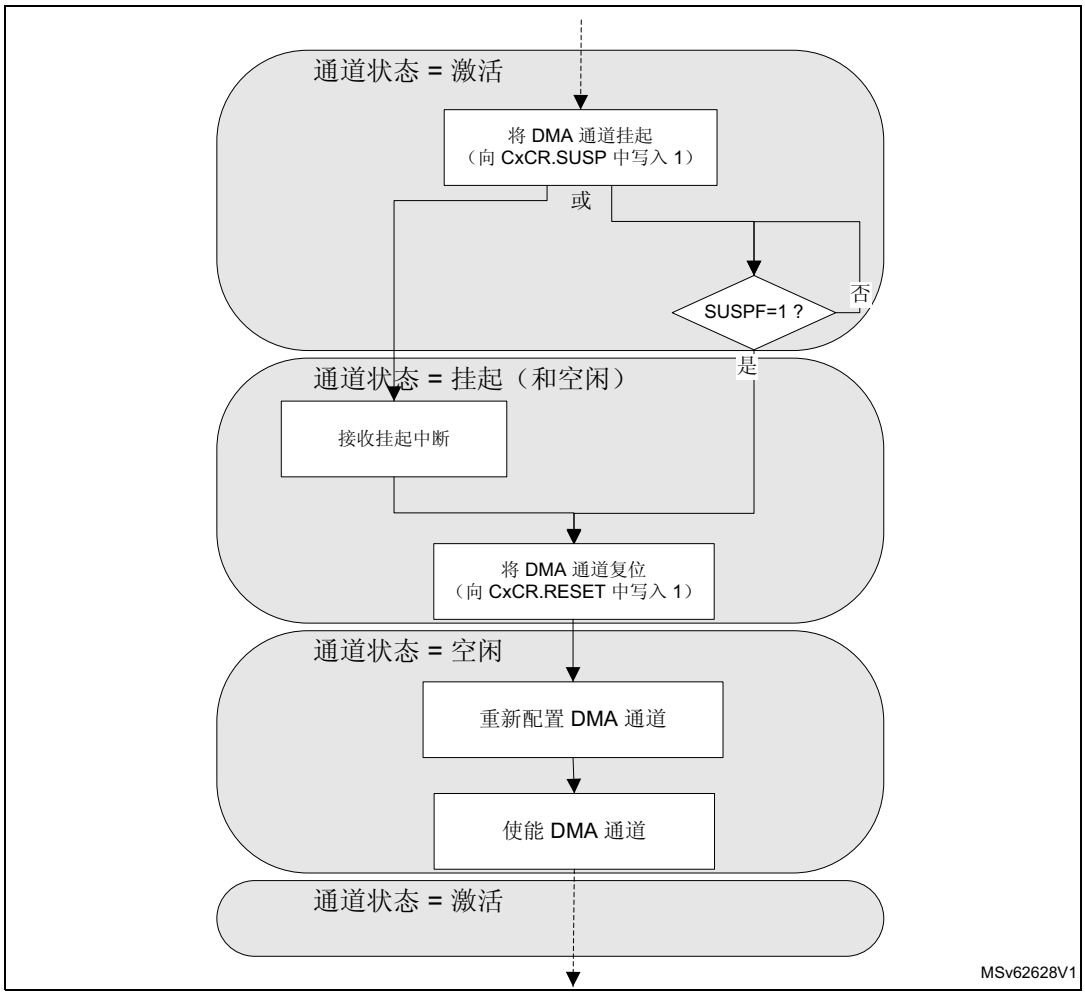
注意： *挂起和恢复序列不会影响 GPDMA_CxCR.EN 位。挂起通道（传输）时并不会挂起已启动的触发信号检测。*

15.4.4 GPDMA 通道中止和重启

软件可以按以下顺序自行中止仍处于活动状态的通道，类似于循环缓冲或双缓冲模式下中止连续的 GPDMA 传输：

1. 软件将 1 写入 GPDMA_CxCR.SUSP 位。
2. 软件轮询挂起标志 GPDMA_CxSR.SUSPF 直至 SUSPF = 1，或者等待之前通过将 1 写入 GPDMA_CxCR.SUSPIE 使能的中断。等待挂起状态的通道生效，意味着等待任何正在通过其主设备端口执行的 GPDMA 传输完成。
3. 软件将 1 写入 GPDMA_CxCR.RESET，复位通道。这会重置通道内部状态并将 FIFO、GPDMA_CxCR.EN 位以及 GPDMA_CxCR.SUSP 位复位。
4. 软件安全地重新配置通道。软件必须重新设定由硬件修改的 GPDMA_CxBR1、GPDMA_CxSAR 和 GPDMA_CxDAR 寄存器。
5. 为了重启中止后重新编程的通道，软件会将 1 写入 GPDMA_CxCR.EN 位，再次使能该通道。

图 53. GPDMA 通道中止和重启序列



15.4.5 GPDMA 链表数据结构

除了直接编程模式，还可以通过传输列表对通道进行编程，该传输列表称为链表项 (LLI) 列表。各个 LLI 均由其数据结构定义。

存储器中，通道 x 的下一个 LLI_{n+1} 的数据结构的基址，为以下各项之和：

- 通道 x 的链路基址（在 GPDMA_CxLBAR 中）
- 链路地址偏移（GPDMA_CxLLR 的 LA[15:2] 位域）。链表寄存器 GPDMA_CxLLR 是来自通道 x 的上一个 LLIn 数据结构的更新结果。

各个 LLI 的数据结构都是特定的。

链表数据结构按照 GPDMA_CxLLR 中 UT1、UT2、UB1、USA、UDA 和 ULL 位以及 UB2 和 UT3 的值进行寻址。

在链表模式下，每个 GPDMA 链表寄存器（GPDMA_CxTR1、GPDMA_CxTR2、GPDMA_CxBR1、GPDMA_CxSAR、GPDMA_CxDAR 或 GPDMA_CxLLR 以及 GPDMA_CxTR3 或 GPDMA_CxBR2）会按照从上一个 LLI 的链表数据结构有条件地更新的 GPDMA_CxLLR 寄存器的当前值，从存储器中的下一个链表数据结构有条件地自动进行更新。

静态链表数据结构

例如，当 GPDMA_CxLLR 中的更新位（UT1、UT2、UB1、USA、UDA 和 ULL 以及 UB2 和 UT3）均置 1 时，存储器中的链表数据结构达到最大，并且：

- 通道 x（x = 0 到 5），连续的 32 位单元，包括 GPDMA_CxTR1、GPDMA_CxTR2、GPDMA_CxBR1、GPDMA_CxSAR、GPDMA_CxDAR 和 GPDMA_CxLLR（参见*图 54*），并包括存储器中的第一个链表寄存器文件 (LLI_0) 以及后面的 LLI（LLI_1、LLI_2）
- 通道 x（x = 6 到 7），连续的 32 位单元，包括 GPDMA_CxTR1、GPDMA_CxTR2、GPDMA_CxBR1、GPDMA_CxSAR、GPDMA_CxDAR 和 GPDMA_CxLLR，以及 GPDMA_CxTR3 和 GPDMA_CxBR2（参见*图 55*），并包括存储器中的第一个链表寄存器文件 (LLI_0) 以及后面的 LLI（例如 LLI_1、LLI_2）

图 54. 线性寻址通道 x 的静态链表数据结构（所有 Uxx = 1）

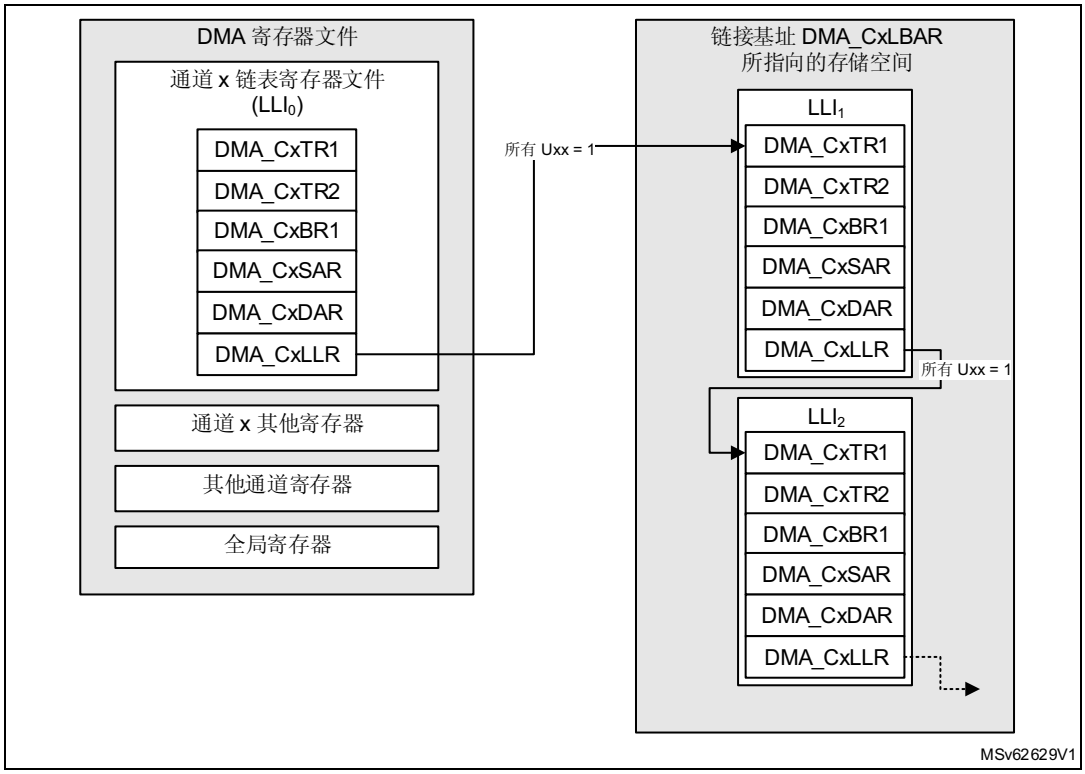
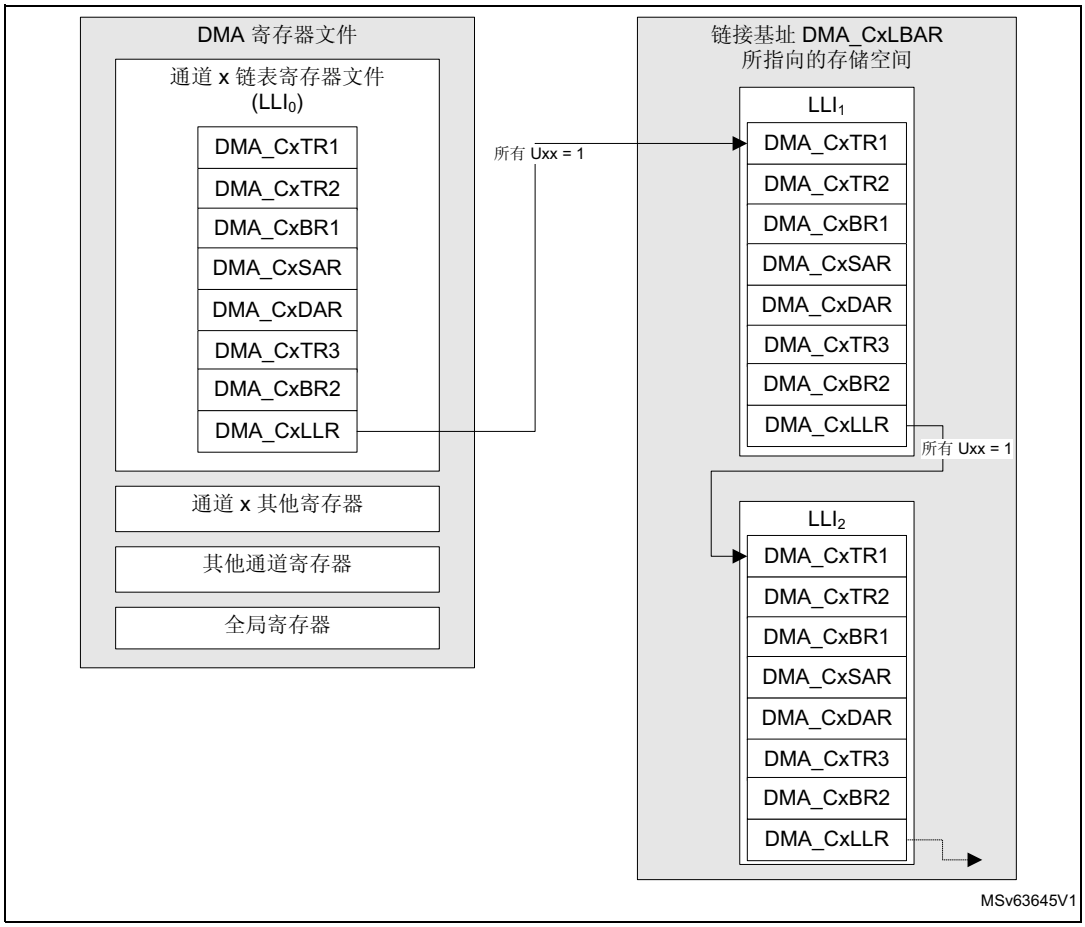


图 55. 2D 寻址通道 x 的静态链表数据结构（所有 Uxx = 1）



动态链表数据结构

另一种情况是，LLI 完整列表的存储器结构可以通过各个 LLI 的特定数据结构进行压缩。

如果 UT1 = 0 并且 UT2 = 1，则寄存器 GPDMA_CxLLR 的链路地址偏移指向 GPDMA_CxTR2 的更新值而不是 GPDMA_CxTR1 的更新值（不对其修改）（参见*图 56*）。

例如：如果 UT1 = UB1 = USA = 0 并且 UT3 = UB2 = 0，则当通道 x 具有 2D 寻址特性时，如果 UT2 = UDA = ULL = 1，则下一个 LLI 不包含 GPDMA_CxTR1 的（更新）值，也不包含 GPDMA_CxBR1、GPDMA_CxSAR、GPDMA_CxTR3、GPDMA_CxBR2 的值。下一个 LLI 包含 GPDMA_CxTR2、GPDMA_CxDAR 和 GPDMA_CxLLR 的更新值，如*图 57* 所示。

图 56. 线性寻址通道 x 的 GPDMA 动态链表数据结

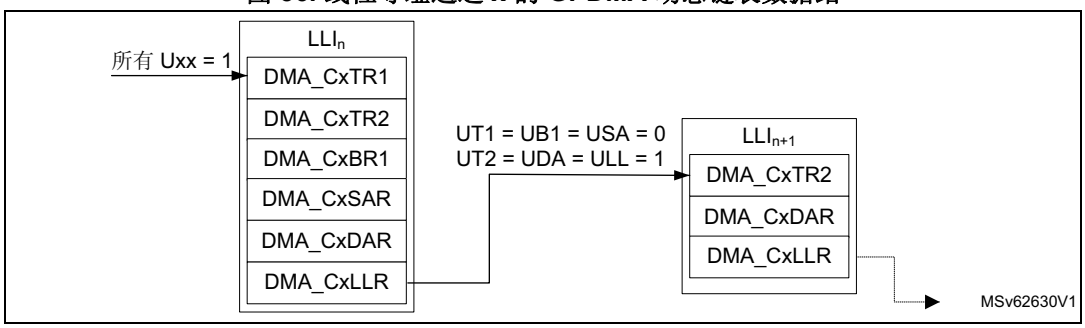
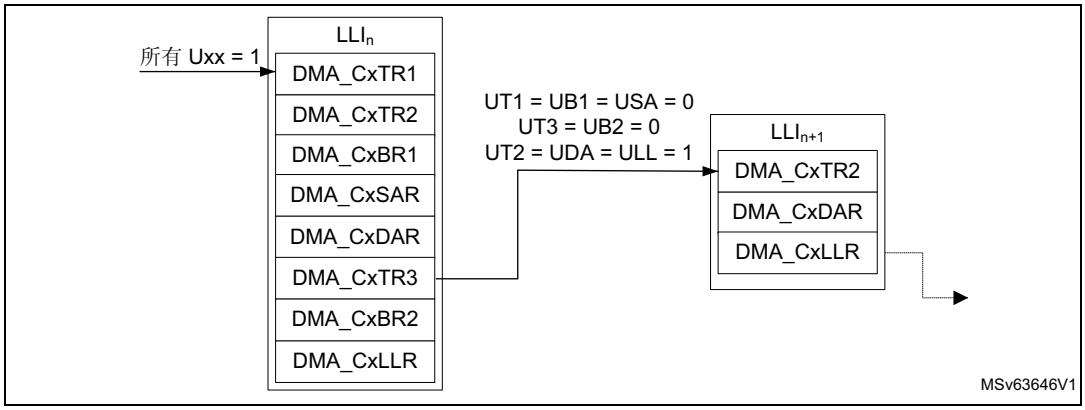


图 57. 2D 寻址通道 x 的 GPDMA 动态链表数据结构



用户必须将各个 LLI_n 的 GPDMA_CxLLR 设定为 32 位对齐，不能超出 GPDMA_CxLBAR 所指向的 64 KB 可寻址空间。

15.4.6 链表项传输执行

LLI_n 传输是一系列以下传输：

1. 数据传输：GPDMA 按 GPDMA 内部寄存器文件所述执行数据传输（对于 LLI_0，此数据传输可以为空）
2. 有条件的链路传输：GPDMA 会按照 LLI_n 的 GPDMA_CxLLR 值，通过下一个 LLI_{n+1} 的数据结构有条件地自动更新其内部寄存器文件。

注意： *由内部寄存器文件 (LLI_0) 定义的初始数据传输可以为空（GPDMA_CxBR1.BNDT[15:0] = 0 且 GPDMA_CxTR2.PFREQ = 0），前提是 GPDMA_CxLLR 中的有条件更新位 UB1 置 1（意味着有一个由存储器中下一个 LLI_1 所描述的非空数据传输将要执行）。*

根据 GPDMA 的预期用途，GPDMA 通道 x 可以按照完整链表所述执行（运行至完成模式，GPDMA_CxCR.LSM = 0），也可以设定为单独执行 LLI（链接步进模式，GPDMA_CxCR.LSM = 1），如后文所述。

15.4.7 运行至完成模式下的 GPDMA 通道状态和链表编程

当 GPDMA_CxCR.LSM = 0 时（在完整列表执行模式下，执行完整的 LLI 序列，称为运行至完成模式），对 GPDMA 通道 x 进行初始设定，通过将 1 写入 GPDMA_CxCR.EN 启动，然后在通道级完成。通道传输满足以下条件：

- 至少具有以下配置：
  - 第一个 LLI_0，内部链表寄存器文件：GPDMA_CxTR1、GPDMA_CxTR2、GPDMA_CxBR1、GPDMA_CxSAR、GPDMA_CxDAR 和 GPDMA_CxLLR 以及 GPDMA_CxTR3 和 GPDMA_CxBR2
  - 存储器中的链表数据结构所描述的最后一个 LLI_N，该链表数据结构由反映倒数第二个 LLI_{N-1} 的 GPDMA_CxLLR 所定义
- 当 GPDMA_CxLLR[31:0] = 0、GPDMA_CxBR1.BRC[10:0] = 0 并且 GPDMA_CxBR1.BNDT[15:0] = 0 时，在最后一个 LLI_{N-1} 传输结束后完成

GPDMA_CxLLR[31:0] = 0 是基于链表的通道传输完成的条件，其含义如下：

- 下一个链路地址的 16 个低有效位 GPDMA_CxLLR.LA[15:0] 为空。
- 所有更新位 GPDMA_CxLLR.Uxx 为空（UT1、UT2、UB1、USA、UDA 和 ULL 以及 UB2 和 UT3）。

如果 GPDMA_CxLLR.LSM = 0，则通道传输可能永远不会完成：

- 如果最后一个 LLI_N 具有递归特性，即指向本身作为下一个 LLI：
  - GPDMA_CxLLR.ULL = 1 并且 GPDMA_CxLLR.LA[15:2] 更新为相同的值
  - 或者 GPDMA_CxLLR.ULL = 0
- 如果 LLI_N 指向上一个 LLI

在以块级别完成常规数据传输的过程中，GPDMA_CxBR1.BNDT[15:0] = 0 并且 GPDMA_CxBR1.BRC[10:0] = 0（如果存在）。或者，块传输可由外设（例如 Rx 模式下的 I3C）提前完成，之后 BNDT[15:0] 不为空（更多详细信息，请参见*第 15.4.14 节*）。

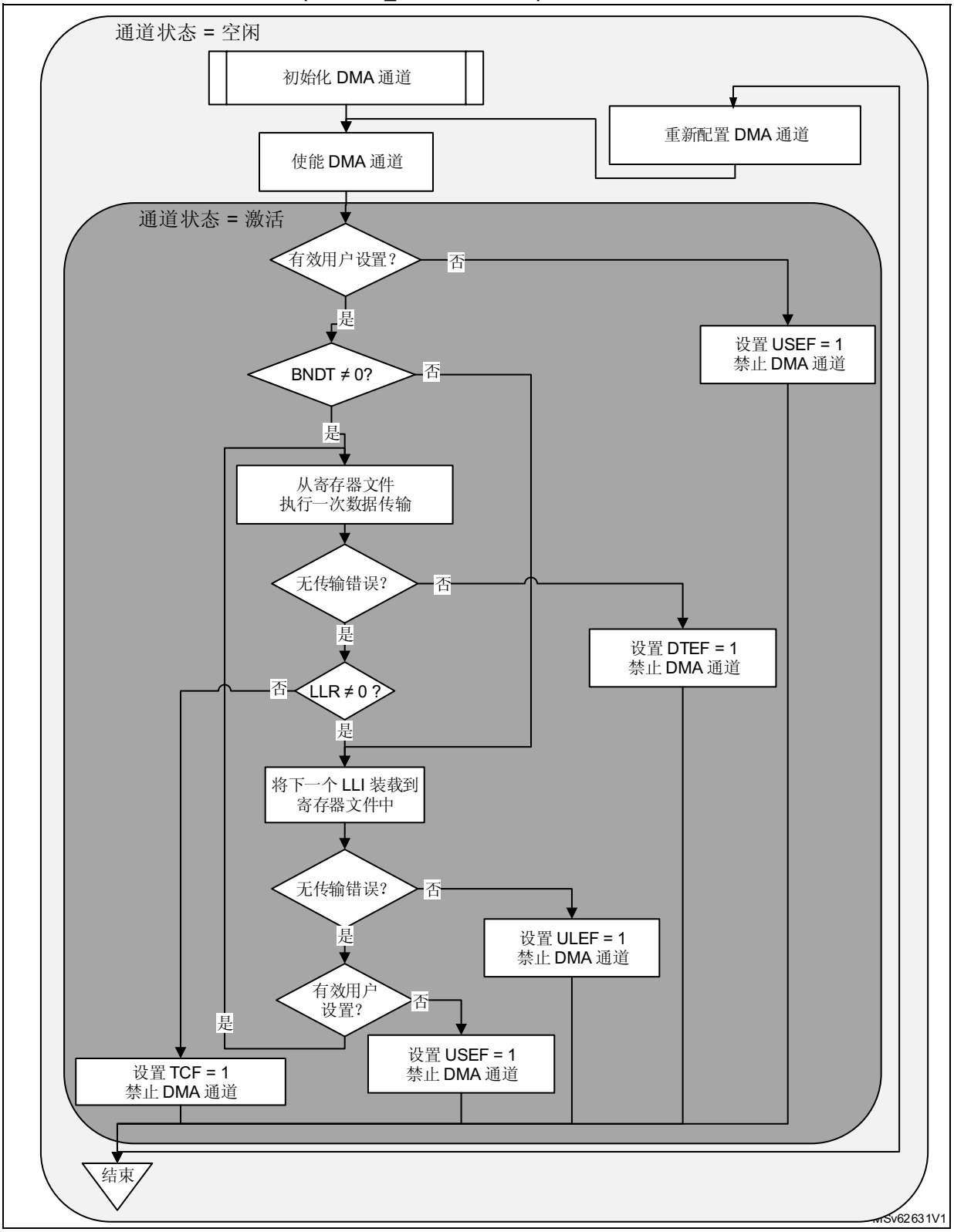
在典型的运行至完成模式下，在 GPDMA 初始化期间执行一次 GPDMA 通道分配，包括其精细编程。为了在运行期间拥有保留的数据通信链路和 GPDMA 服务，实现连续重复传输（在外设与存储器之间双向传输或者在存储器之间传输），此保留数据通信链路可以包含一个通道，或者可以共享通道且重复传输中包含一系列 LLI。

图 58 说明了运行至完成模式下的 GPDMA 通道传输及其寄存器编程。

注意： *图 58 并不是为了说明 TCEF 的发生频率，其取决于 GPDMA_CxTR2 中 TCEM[1:0] 的设定值。它可以在（每个）块传输完成、（每个）2D 块传输完成、（每个）LLI 传输完成时发生，或仅在通道传输完成时发生。在运行至完成模式下，无论 TCEM[1:0] 的值如何，通道传输完成时，硬件都始终会将 TCEF 置 1，并禁止通道。*

在图 58 中，BNDT ≠ 0 是开始第一次数据传输的典型条件。如果外设请求提前终止数据传输，则此条件变为（BNDT ≠ 0 且 PFREQ = 1）（请参见第 15.3.6 节）。

图 58. 运行至完成模式 (GPDMA_CxCR.LSM = 0) 下的 GPDMA 通道执行和链表编程



在运行至完成模式下，运行期间通过辅助通道插入 LLI_n

对 LLI_{n-1} 数据结构中 GPDMA_CxTR2 的以下位域进行编程设定后，LLI_{n-1} 链路传输（LLI_n 加载的开始）的启动以出现触发信号为条件：

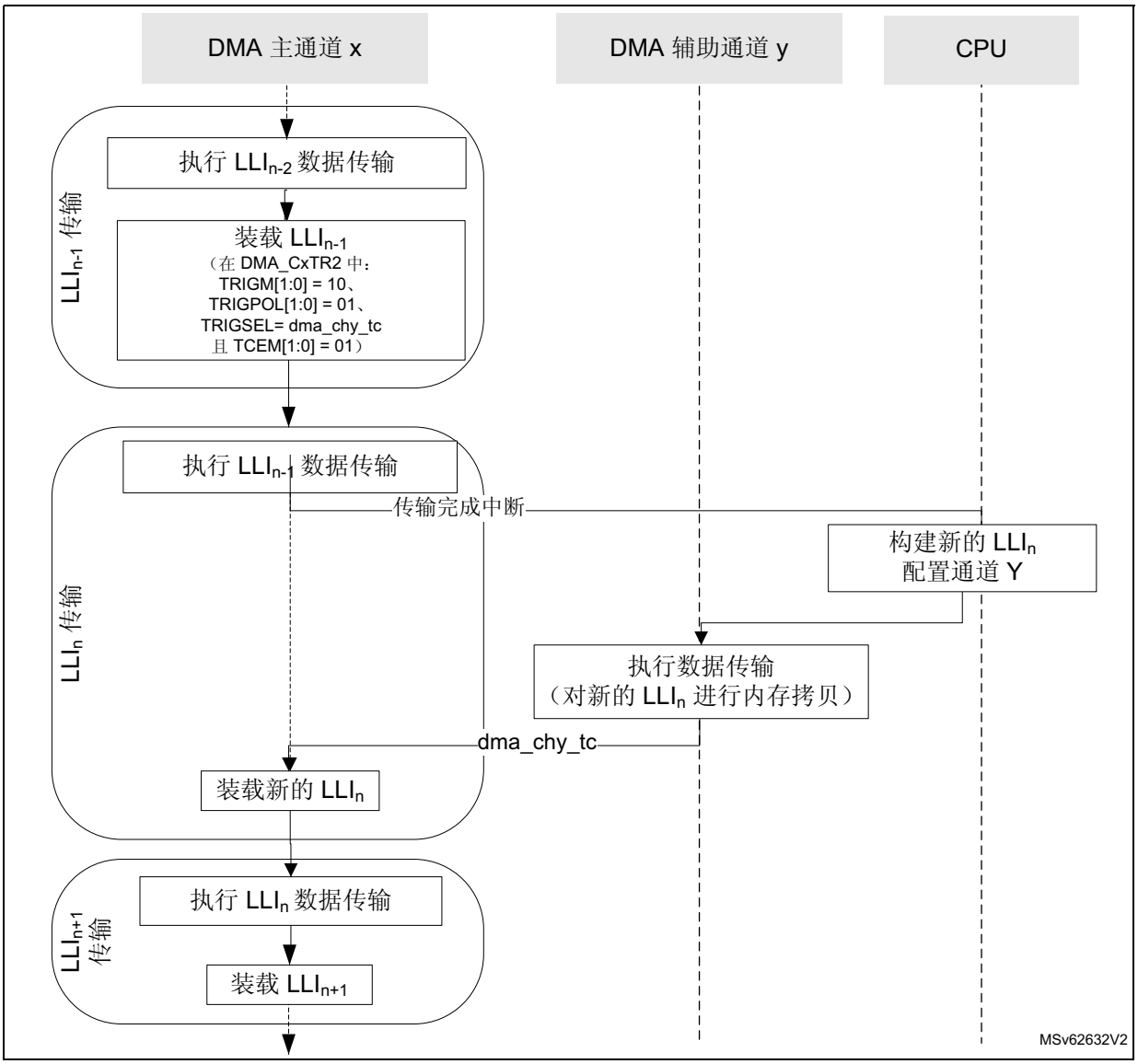
- TRIGM[1:0] = 10（链路传输触发模式）
- TRIGPOL[1:0] = 01 或 10（上升沿或下降沿）
- TRIGSEL[5:0]（有关触发信号选择的详细信息，请参见*第 15.3.7 节*）

另有一个辅助通道 y 可用于在存储器中存储通道 x LLI_n，以及生成传输完成事件 gpdma_chy_tc。通过选择该事件作为通道 x 的 LLI_{n-1} 链路传输输入触发信号，软件可以在主通道 x 的 LLI_{n-1} 数据传输后将主通道 x 暂停，直至确定确实对 LLI_n 执行写操作为止。

下图说明了主通道 x 借助另一个辅助通道 y 确立链表的动态过程。

小心： 此用例仅限于 LLI_{n-1} 数据传输不需要触发信号的应用。此 LLI_{n-1} 的触发模式用于加载下一个 LLI_n。

图 59. 通过辅助 GPDMA 通道 y 插入 LLI_n



15.4.8 链接步进模式下的 GPDMA 通道状态和链表编程

当 GPDMA_CxCR.LSM = 1 时（在链接步进模式下，单独执行一个 LLI），通道传输将在每次单独执行 LLI 之后执行并完成，包括其（有条件的）数据传输和（有条件的）链路传输。

GPDMA 通道传输可以在 LLI 级设定，通过将 1 写入 GPDMA_CxCR.EN 启动，然后在 LLI 级完成：

- 当前 LLI_n 传输的具体描述如下：
  - GPDMA_CxTR1 定义源/目标基本单次/突发传输。
  - GPDMA_CxBR1 定义块级别的字节数 (BNDT[15:0])，对于通道 x（x = 6 到 7），定义 2D/重复块级别的块数 (BRC[10:0]+1) 以及偏移地址的递增/递减模式。
  - GPDMA_CxTR2 定义传输的输入控制（请求，触发信号）和输出控制（传输完成事件）。

  - GPDMA_CxSAR/GPDMA_CxDAR 定义源/目标传输起始地址。
  - GPDMA_CxTR3 针对通道 x（x = 6 到 7）定义突发传输之间的源/目标附加地址偏移。
  - GPDMA_CxBR2 针对通道 x（x = 6 到 7）定义 2D/重复块级别的块之间的源/目标附加地址偏移。
  - GPDMA_CxLLR 定义存储器中下一个 LLI_{n+1} 的数据结构和偏移地址。
- 当前 LLI_n 传输在单次执行当前 LLI_n 之后完成：
  - （有条件的）数据传输完成后（当 GPDMA_CxBR1.BRC[10:0] = 0 并且 GPDMA_CxBR1.BNDT[15:0] = 0 时）
  - 从存储器中下一个 LLI_{n+1} 的数据结构（有条件地）更新 GPDMA 链路寄存器文件之后

注意： *如果 LLI 具有递归特性（即指向本身作为下一个 LLI，GPDMA_CxLLR.ULL = 1 并且 GPDMA_CxLLR.LA[15:2] 更新为相同的值，或者 GPDMA_CxLLR.ULL = 0），则在每次重复此 LLI 单次执行之后，处于链接步进模式的通道完成传输。*

在以块级别完成常规数据传输的过程中，GPDMA_CxBR1.BNDT[15:0] = 0 并且 GPDMA_CxBR1.BRC[10:0] = 0。或者，块传输可由外设（例如 Rx 模式下的 I3C）提前完成，之后 BNDT[15:0] 不为空（更多详细信息，请参见*第 15.4.14 节*）。

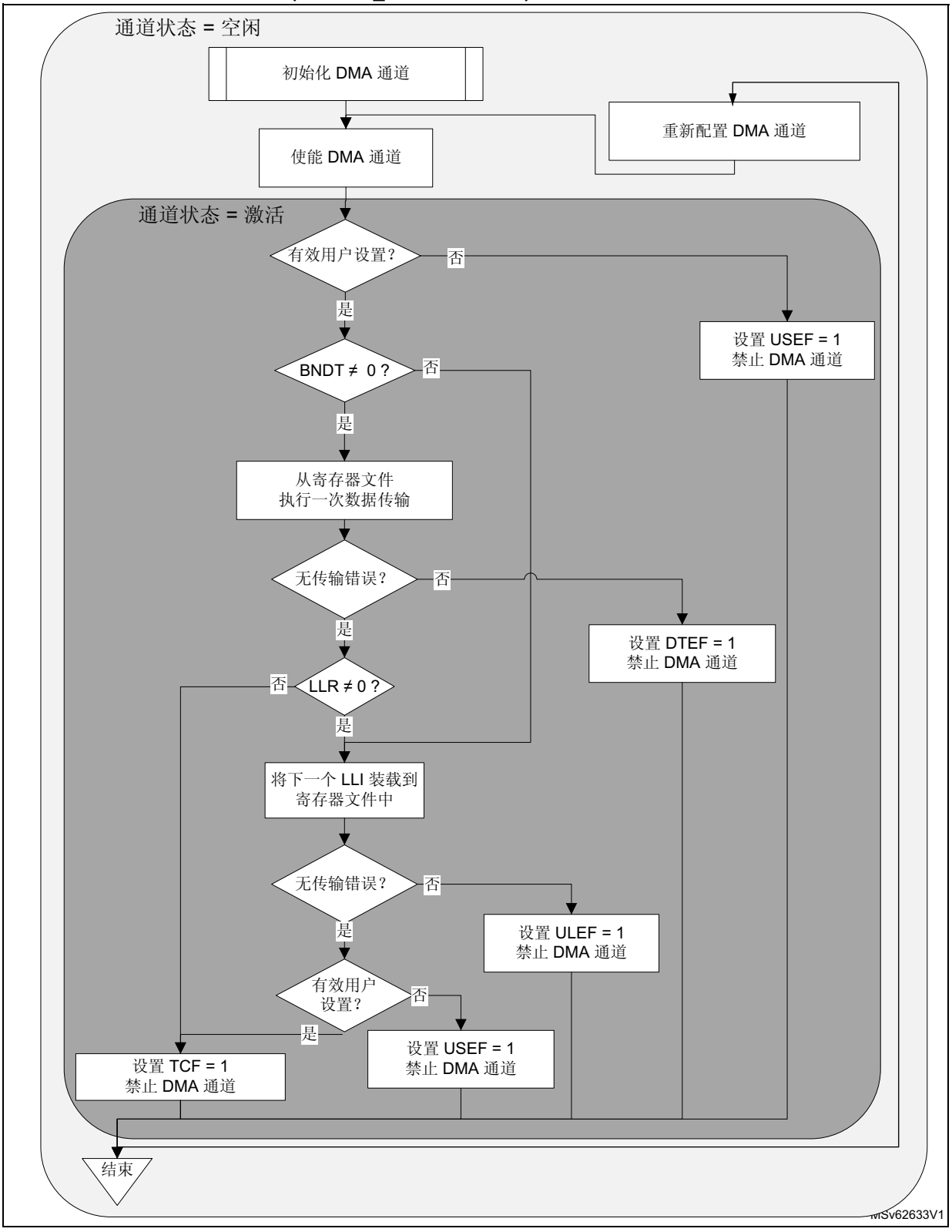
链接步进模式可用于运行期间在存储器中动态建立 LLI。对任何 LLI_n 使用静态数据结构（GPDMA_CxLLR 的所有更新位均具有静态值，LLI_n.LLR.LA = LLI_{n-1}.LLR.LA + 常数），可以让软件运行更顺利。

图 60 说明了链接步进模式下 GPDMA 通道的执行模式及其编程。

注意： *图 60 并不是为了说明 TCEF 的发生频率，其取决于 GPDMA_CxTR2 中 TCEM[1:0] 的设定值。它可以在（每个）块传输完成、（每个）2D 块传输完成、（每个）LLI 传输完成时发生，或仅在最后一个 LLI 数据传输完成时发生。在链接步进模式下，在每次单次执行 LLI 之后，通道即被禁止，并且 TCEF 是否发生取决于 TCEM[1:0] 的值。*

在图 60 中，BNDT ≠ 0 是开始第一次数据传输的典型条件。如果外设请求提前终止数据传输，则此条件变为（BNDT ≠ 0 且 PFREQ = 1）（请参见第 15.3.6 节）。

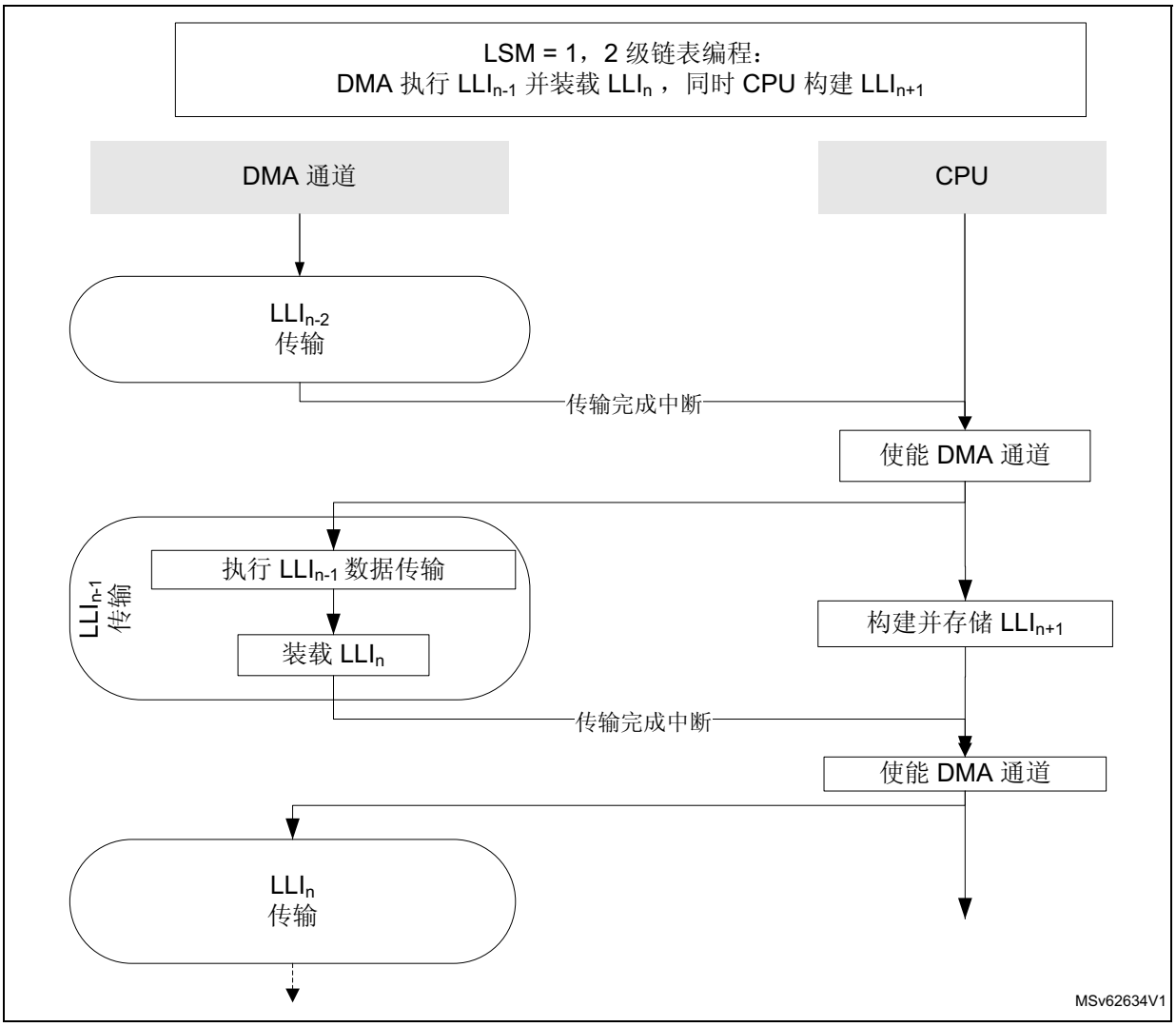
图 60. 链接步进模式 (GPDMA_CxCR.LSM = 1) 下的 GPDMA 通道执行和链表编程



在链接步进模式下，运行期间添加 LLI_{n+1}

运行期间，软件可以将 LLI_{n+1}（和下一个 LLI）的建立推迟，直至 GPDMA 从 LLI_{n-1} 执行了传输并从存储器加载了 LLI_n（或者在这一过程之后），如下图所示。

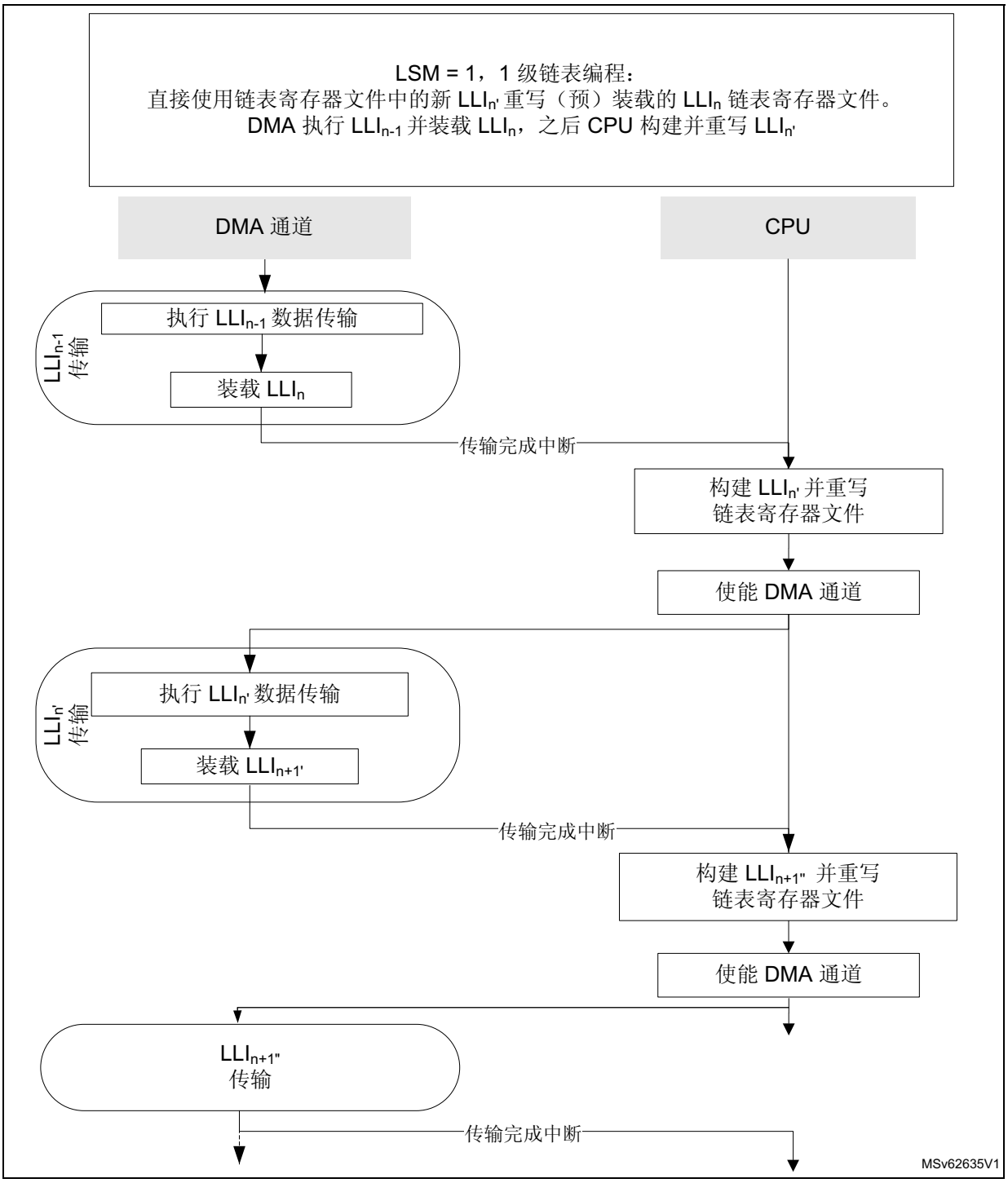
图 61. 建立 LLI_{n+1}：链接步进模式下的 GPDMA 动态链表



在链接步进模式，在运行期间将 LLI_n 替换为新的 LLI_n'（在链表寄存器文件中）

在此链接步进模式下，运行期间，软件可以在 GPDMA 从 LLI_{n-1} 执行了传输并从存储器加载了之前建立的 LLI_n 之后，用新的 LLI_n' 直接重写链表寄存器文件，以此建立并插入新的 LLI_n'，如下图所示。

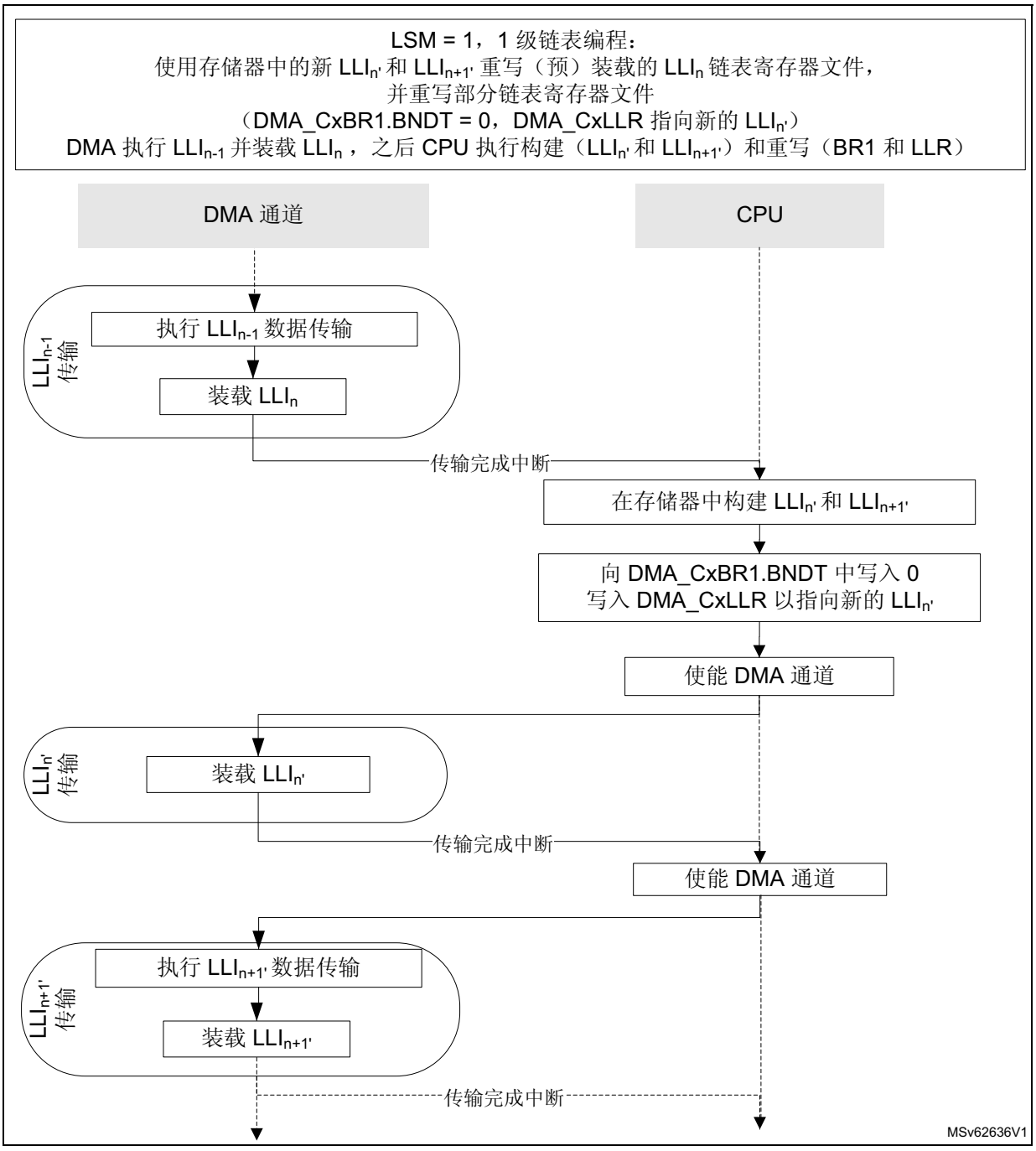
图 62. 在链接步进模式下使用寄存器文件中新的 LLI_n' 进行替换



以链接步进模式在运行期间将 LLI_n 替换为新的 LLI_n'（在存储器中）

在 GPDMA 从 LLI_{n-1} 执行了传输并从存储器加载了之前建立的 LLI_n 之后，软件可以重写部分链表寄存器文件（GPDMA_CxBR1.BNDT[15:0] 重写为空，GPDMA_CxLLR 重写为指向新的 LLI_n'），以此在存储器中建立并插入新的 LLI_n' 和 LLI_{n+1}'，如下图所示。

图 63. 在链接步进模式下使用存储器中新的 LLI_n, 和 LLI_{n+1}, 进行替换（选项 1）



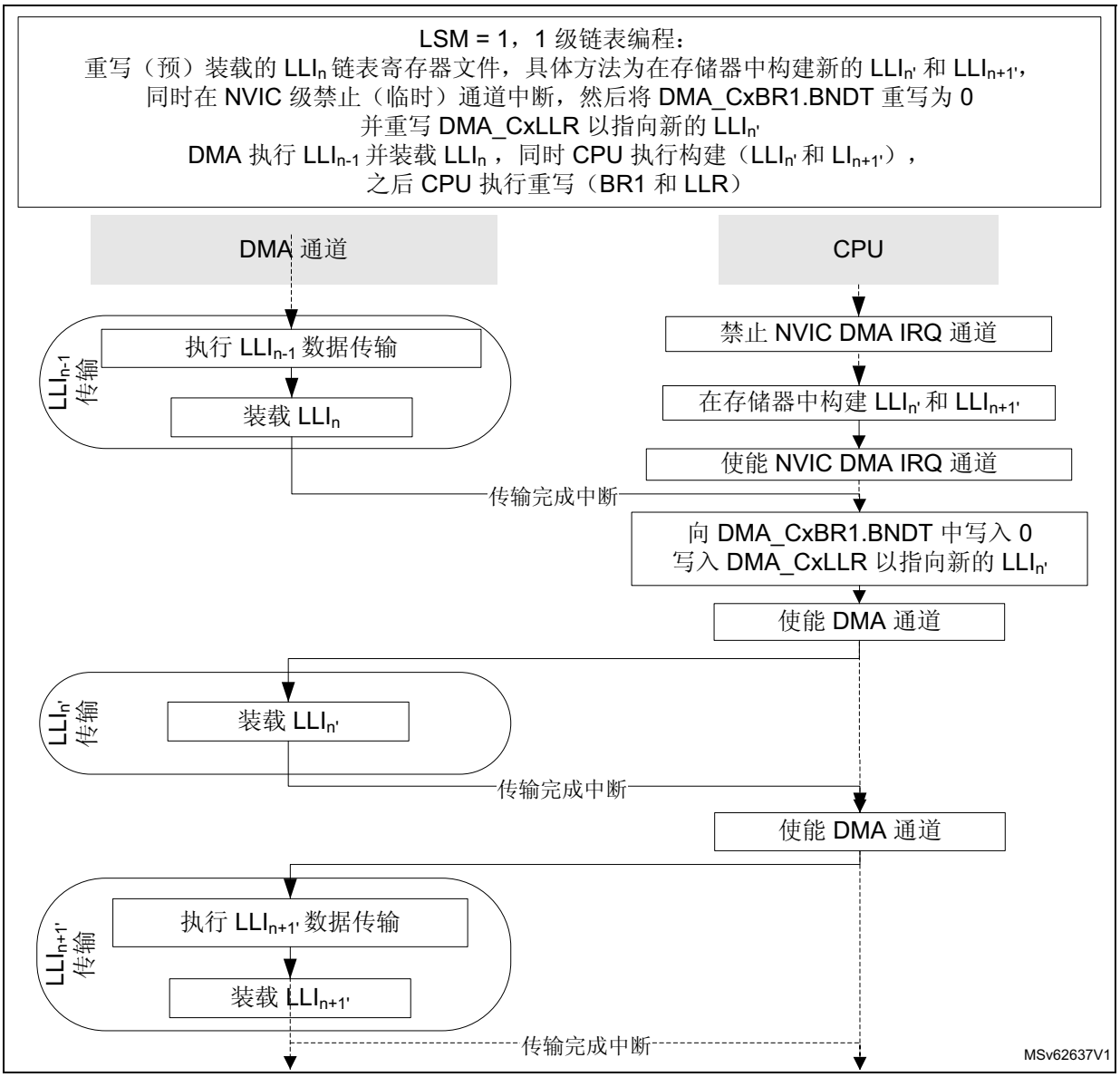
在链接步进模式下，运行期间将 LLI_n 替换为新的 LLI_n'

存在其他软件实现方案。在 GPDMA 从 LLI_{n-1} 执行传输并从存储器加载之前（甚至更早）建立的 LLI_n 的同时，软件可以执行以下操作：

1. 禁止 NVIC，以便不会因为处理中断而中断。
2. 建立新的 LLI_n' 和新的 LLI_{n+1}'。
3. 再次使能 NVIC 以获得通道中断（传输完成）通知。

LLI_{n-1} 中断处理程序中的软件随后被限定为将 GPDMA_CxBR1.BNDT[15:0] 重写为空，将 GPDMA_CxLLR 重写为指向新的 LLI_n'，如下图所示。

图 64. 在链接步进模式下使用存储器中新的 LLI_n' 和 LLI_{n+1}' 进行替换（选项 2）



15.4.9 GPDMA 通道状态和链表编程

当通道处于禁止状态 (GPDMA_CxCR.EN = 0) 时，软件可以重新配置通道，并更新执行模式 (GPDMA_CxCR.LSM)，在运行至完成模式和链接步进模式之间双向切换。

在任意执行模式下，软件均可执行以下操作：

- 将存储器中的 LLI_{n+1} 重新设定为最终通过该 LLI_{n+1}（将该 LLI_{n+1} 的 GPDMA_CxLLR 清零）完成通道传输，在此之前，该 LLI_{n+1} 由 GPDMA 通道加载/使用
- 中止通道并通过 LSM 更新重新配置通道（参见*第 15.4.4 节*。）

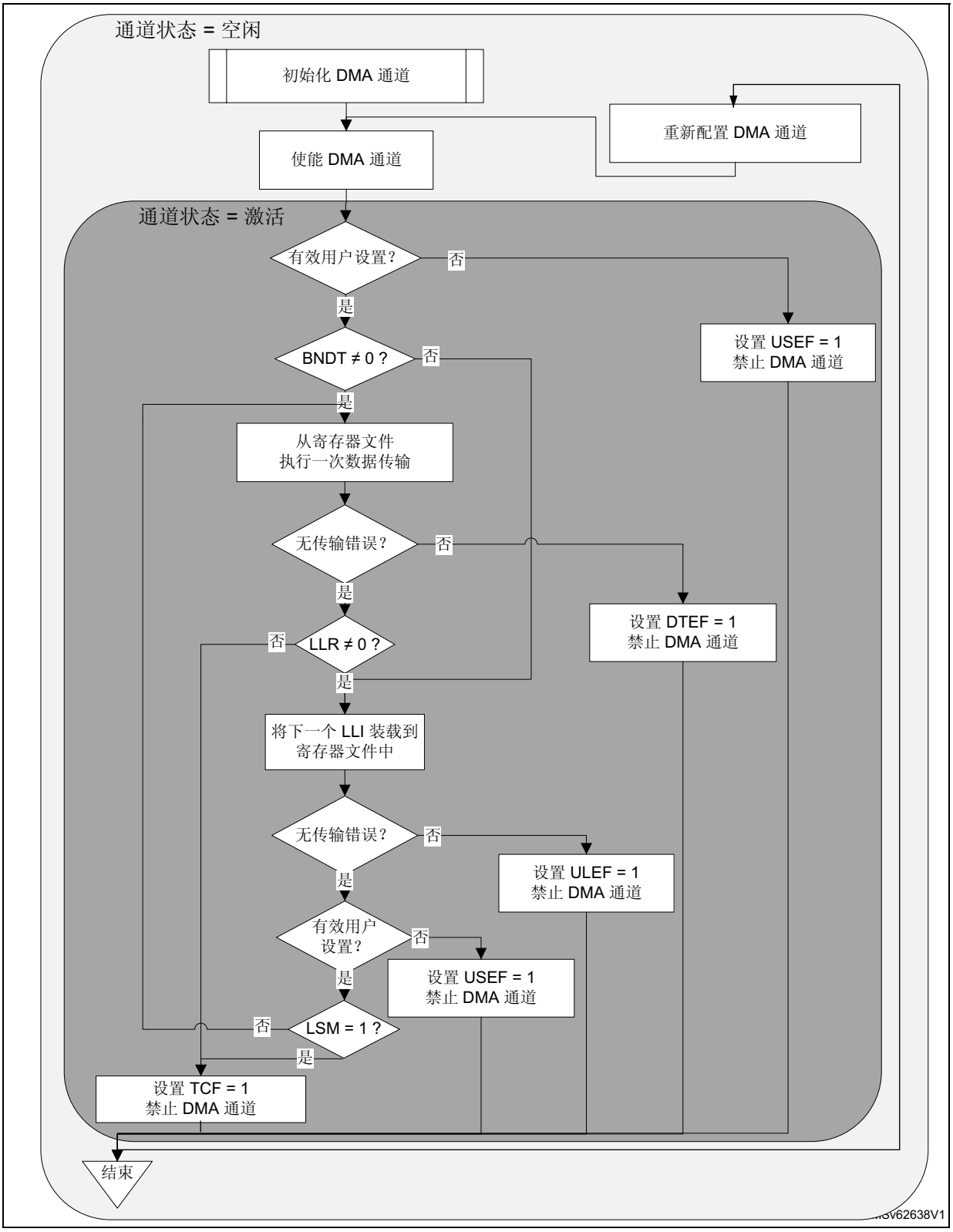
在链接步进模式下，在 LLI_{n-1} 期间，软件在每次单次执行任何 LLI 之后都可以将 LSM 清零。

图 65 显示对任何执行模式均适用的整体、统一的 GPDMA 链表编程流程。

注意： *图 65 并不是为了说明 TCEF 的发生频率，其取决于 GPDMA_CxTR2 中 TCEM[1:0] 的设定值。它可以在（每个）块传输完成、（每个）2D 块传输完成、（每个）LLI 传输完成时发生，或仅在最后一个 LLI 数据传输完成时发生。在运行至完成模式下，无论 TCEM[1:0] 的值如何，通道传输完成时，硬件都始终会将 TCEF 置 1，并禁止通道。在链接步进模式下，在每次单次执行 LLI 之后，通道即被禁止，并且 TCEF 是否发生取决于 TCEM[1:0] 的值。*

在图 65 中，BNDT ≠ 0 是开始第一次数据传输的典型条件。如果外设请求提前终止数据传输，则此条件变为（BNDT ≠ 0 且 PFREQ = 1）（请参见第 15.3.6 节）。

图 65. GPDMA 通道执行和链表编程



15.4.10 基于 FIFO 的 GPDMA 传输

系统提供一种单次传输运行模式：FIFO 模式。提供多种基于 FIFO 的传输。任意通道 x 均通过一个专用 FIFO 来实现，其大小由 dma_fifo_size[x] 定义（更多详细信息，请参见*第 15.3.2 节*）。

GPDMA 突发

设定的最低级别传输是 GPDMA 突发。

GPDMA 突发是从源接收或发送到目标的一连串数据。源（或目标）突发由 GPDMA_CxTR1 寄存器中的位域 SBL_1[5:0]（或 DBL_1[5:0]）所定义的突发长度，以及位域 SDW_LOG2[1:0]（或 DDW_LOG2[1:0]）所定义的数据宽度来设定。

GPDMA 突发的各个数据（称为节拍）的寻址模式由 GPDMA_CxTR1 的 SINC（源突发）和 DINC（目标突发）定义：固定寻址或连续数据增量式寻址。

GPDMA 源/目标突发的起始地址及随后的地址（由 GPDMA_CxSAR 和 GPDMA_CxDAR 定义）必须与各自的数据宽度对齐。

下表列出了 GPDMA 突发的主要特性。

表 91. 编程设定的 GPDMA 源/目标突发

SDW_LOG2[1:0] DDW_LOG2[1:0]	数据宽度：（字节）	SINC/DINC	SBL_1[5:0] DBL_1[5:0]	突发长度（数据/节拍）	下一个数据/节拍地址	下一个突发地址	突发地址对齐
00	1	0（固定）	n = 0 到 63(1)	n+1	+ 0	+ 0	1
01	2						2
10	4						4
00	1	1（连续增量）			+ 1	+ (n + 1)	1
01	2				+ 2	+ 2 * (n + 1)	2
10	4				+ 4	+ 4 * (n + 1)	4
11	禁止使用的用户设置，导致 USEF 信号生成，并且不会发出任何突发传输信号。						

1. 当 S/DBL_1[5:0] = 0 时，突发的长度为 1。此时突发传输也称为单次传输。

一旦设定的源/目标突发传输完成，上表中的下一个突发传输地址即为由 GPDMA_CxSAR 或 GPDMA_CxDAR 所指向的下一个源/目标的默认地址。该默认值是指固定地址/连续增量式地址。

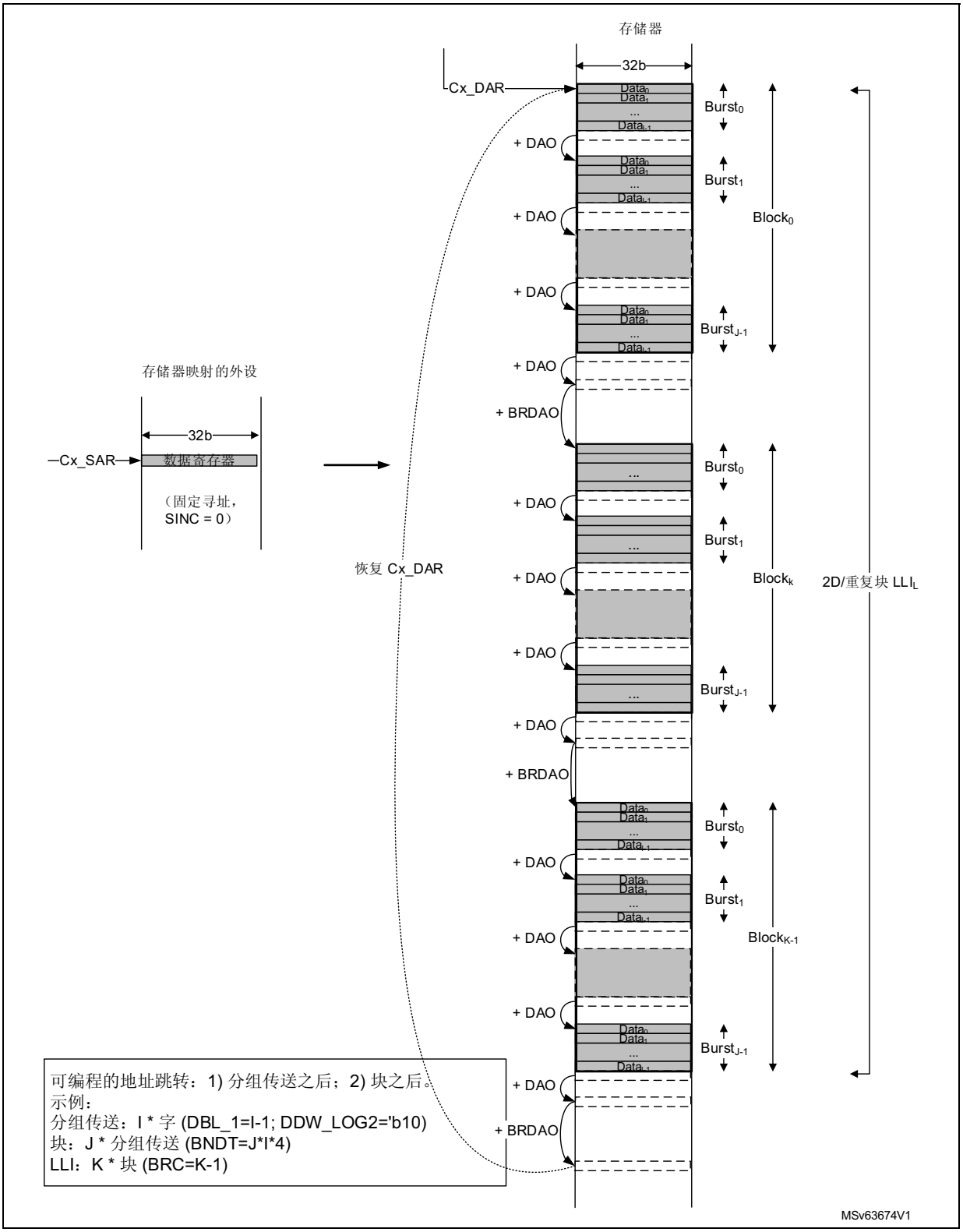
具有 2D 寻址特性的 GPDMA 突发（通道 x，x = 6、7）

如果通道具有额外的 2D 寻址特性，则该默认值未将两个设定的递增或递减偏移考虑在内。这两个额外偏移（默认值为空）的应用条件为：

- 在每个源/目标突发完成之后，由 GPDMA_CxTR2.SAO[12:0]/DAO[12:0] 和 GPDMA_CxBR1.SDEC/DDEC 分别定义
- 在每个块完成之后，由 GPDMA_CxBR2.BRSAO[15:0]/BRDAO[15:0] 和 GPDMA_CxBR1.BRSDEC/BRDDEC 分别定义

这样，可以通过第一个设定的地址跳转（在每个突发传输完成后）以及第二个设定的地址跳转（在每个块传输完成后）对 2D/重复块进行寻址，如下图所示（含 2D 目标缓冲区）。

图 66. 编程设定的 2D 寻址



基于 FIFO 的 GPDMA 突发

在 FIFO 模式下，传输通常由两个流水线式的独立突发传输组成：

- 一个突发通过分配的源设备主端口从源传输到 FIFO，由 GPDMA_CxTR1.SAP 所定义
- 一个突发通过分配的目标设备主端口从 FIFO 传输到目标，由 GPDMA_CxTR1.DAP 所定义

GPDMA 源突发

可以通过分配的端口尽早调度所请求的传输到 FIFO 的源突发传输，具体取决于当前的 FIFO 级别与设定的突发大小（当 FIFO 准备从源获取一个新的突发时）：

当 FIFO 级别 ≤ $2^{dma_fifo_size[x]}$ - (SBL_1[5:0]+1) * $2^{SDW_LOG2[1:0]}$ 时

其中：

- FIFO 级别表示 FIFO 的当前填充级别（字节）。
- $2^{dma_fifo_size[x]}$ 表示通道 x 的 FIFO 大小的一半（字节）（有关实现的详细信息和 dma_fifo_size[x] 值，请参见*第 15.3.2 节*）。
- (SBL_1[5:0]+1) * $2^{SDW_LOG2[1:0]}$ 表示设定的源突发传输的大小（字节）。

基于通道优先级 (GPDMA_CxCR.PRIO[1:0])，这个准备就绪的基于 FIFO 的源传输在内部与其他请求的活动通道进行仲裁。

GPDMA 目标突发

可以通过分配的端口尽早调度所请求的源自 FIFO 的目标突发传输，具体取决于当前的 FIFO 级别与设定的突发大小（当 FIFO 准备向目标推送一个新的突发时）：

当 FIFO 级别 ≥ (DBL_1[5:0]+1) * $2^{DDW_LOG2[1:0]}$ 时

其中：

- FIFO 级别表示 FIFO 的当前填充级别（字节）。
- (DBL_1[5:0]+1) * $2^{DDW_LOG2[1:0]}$ 表示设定的目标突发传输的大小（字节）。

基于通道优先级，这个准备就绪的基于 FIFO 的目标传输在内部与其他请求的活动通道进行仲裁。

GPDMA 突发与源块大小、1 KB 地址边界和 FIFO 大小

设定的源/目标 GPDMA 突发传输通过一个 AHB 突发传输原样实现，除非符合以下条件之一：

- 当通道 x 的 FIFO 大小的一半小于设定的源/目标突发大小时，设定的源/目标 GPDMA 突发通过一系列较小的单次传输或突发传输实现，每个传输的大小小于或等于 FIFO 大小的一半，没有任何用户限制。
- 如果源块大小 (GPDMA_CxBR1.BNDT[15:0]) 不是源突发大小的倍数，而是源突发的数据宽度 (GPDMA_CxTR1.SDW_LOG2[1:0]) 的倍数，则 GPDMA 会将突发修改并缩短为较小长度的单次传输或突发传输，以便在没有任何用户限制的情况下准确传输源块大小。
- 如果源/目标突发传输在 AHB 传输时跨过了 1 KB 的地址边界，则 GPDMA 会将设定的突发修改并缩短为较小长度的单次传输或突发传输以符合 AHB 协议，没有任何用户限制。
- 如果源/目标突发长度在 AHB 传输时超出 16，则 GPDMA 会将设定的突发修改并缩短为较小长度的单次传输或突发传输以符合 AHB 协议，没有任何用户限制。

在任何情况下，GPDMA 始终都能确保源/目标数据（和地址）的完整性，没有任何用户限制。当前 FIFO 级别（软件通过 GPDMA_CxSR 读取）将与有效传输大小进行比较，并进行更新，GPDMA 会在各个 AHB 单次或突发传输之间重新仲裁，还可能进行修改。

基于通道优先级，与设定的突发相比，每个突发大小较小的单次或突发传输都会在内部与其他所请求的活动通道进行仲裁。

注意： *在链表模式下，与存储器到内部 GPDMA 寄存器的链表参数更新相关的 GPDMA 读传输，将通过链接分配的端口进行调度，由 GPDMA_CxCR.LAP 设定。*

GPDMA 数据处理：按字节进行重新排序、封装/解封、填充/截断、符号扩展和左/右对齐

数据处理由 GPDMA_CxTR1 控制。按照 SDW_LOG2[21:0] 和 DDW_LOG2[1:0] 位域，设定突发的源/目标数据宽度为字节、半字或字（参见*表 92*）。

用户可以配置从源传输数据和传输到目标之间的数据处理。具体而言，编程设定的数据处理按以下方式顺序执行：

1. 基于字节的源重新排序
   - 如果 SBX = 1 并且源数据宽度为字，则每个源数据字中间未对齐半字的两个字节交换。
2. 如果目标数据宽度与源数据宽度不同，则基于 PAM[1:0]，数据宽度通过封装、解封、填充或截断来转换：
   - 如果目标数据宽度 > 源数据宽度，则后 SBX 源数据可以右对齐并用 0 填充，或者符号扩展到目标数据宽度，或者进入 FIFO 队列并打包为目标数据宽度。
   - 如果目标数据宽度 < 源数据宽度，则后 SBX 数据可以右对齐并从左侧截断到目标数据宽度，或者进入 FIFO 队列并解包，并向下移动达到目标数据宽度。
3. 基于字节的目标重新排序：
   - 如果 DBX = 1 并且目标数据宽度不是字节，则对齐的后 PAM[1:0] 半字中的两个字节交换。
   - 如果 DHX = 1 并且目标数据宽度既不是字节也不是半字，则对齐的后 PAM[1:0] 字中的两个对齐半字交换。

注意： *通过编程设定 0 填充右对齐和基于字节的目标重新排序，可以实现 0 填充左对齐。*

下表列出了从源到目标可能的数据处理。

表 92. 编程设定的数据处理

SDW_LOG2 [1:0]	源数据	源数据流(1)	SBX	DDW_LOG2 [1:0]	目标数据	PAM[1:0](2)	DBX	DHX	目标数据流(1)
00	字节	B_7、B_6、B_5、B_4、B_3、B_2、B_1、B_0	x	00	字节	xx	x	x	B_7、B_6、B_5、B_4、B_3、B_2、B_1、B_0
				01	半字	00（RA、0P）	0		$0B_3$、$0B_2$、$0B_1$、$0B_0$
							1		B_30、B_20、B_10、B_00
						01（RA、SE）	0		SB_3、SB_2、SB_1、SB_0
							1		B_3S、B_2S、B_1S、B_0S
						1x（封装）	0		B_7B_6、B_5B_4、B_3B_2、B_1B_0
							1		B_6B_7、B_4B_5、B_2B_3、B_0B_1
				10	字	00（RA、0P）	0	0	$000B_1$、$000B_0$
							1		$00B_10$、$00B_00$
							0	1	$0B_100$、$0B_000$
							1		B_1000、B_0000
						01（RA、SE）	0	0	$SSSB_1$、$SSSB_0$
							1		SSB_1S、SSB_0S
							0	1	SB_1SS、SB_0SS
							1		B_1SSS、B_0SSS
						1x（封装）	0	0	$B_7B_6B_5B_4$、$B_3B_2B_1B_0$
							1		$B_6B_7B_4B_5$、$B_2B_3B_0B_1$
							0	1	$B_5B_4B_7B_6$、$B_1B_0B_3B_2$
							1		$B_4B_5B_6B_7$、$B_0B_1B_2B_3$
01	半字	B_7B_6、B_5B_4、B_3B_2、B_1B_0		00	字节	00（RA、LT）	x	x	B_6、B_4、B_2、B_0
						01（LA、RT）			B_7、B_5、B_3、B_1
						1x（解封）			B_7、B_6、B_5、B_4、B_3、B_2、B_1、B_0

表 92. 编程设定的数据处理（续）

SDW_LOG2 [1:0]	源数据	源数据流(1)	SBX	DDW_LOG2 [1:0]	目标数据	PAM[1:0](2)	DBX	DHX	目标数据流(1)
01	半字	B_7B_6、B_5B_4、B_3B_2、B_1B_0	x	01	半字	xx	0	x	B_7B_6、B_5B_4、B_3B_2、B_1B_0
							1		B_6B_7、B_4B_5、B_2B_3、B_0B_1
				10	字	00（RA、0P）	0	0	$00B_3B_2$、$00B_1B_0$
							1		$00B_2B_3$、$00B_0B_1$
							0	1	B_3B_200、B_1B_000
							1		B_2B_300、B_0B_100
						01（RA、SE）	0	0	SSB_3B_2、SSB_1B_0
							1		SSB_2B_3、SSB_0B_1
							0	1	B_3B_2SS、B_1B_0SS
							1		B_2B_3SS、B_0B_1SS
						1x（封装）	0	0	$B_7B_6B_5B_4$、$B_3B_2B_1B_0$
							1		$B_6B_7B_4B_5$、$B_2B_3B_0B_1$
							0	1	$B_5B_4B_7B_6$、$B_1B_0B_3B_2$
							1		$B_4B_5B_6B_7$、$B_0B_1B_2B_3$
10	字	$B_7B_6B_5B_4$、$B_3B_2B_1B_0$	0	00	字节	00（RA、LT）	x	x	B_{12}、B_8、B_4、B_0
						01（LA、RT）			B_{15}、B_{11}、B_7、B_3
						10（解封）			B_7、B_6、B_5、B_4、B_3、B_2、B_1、B_0
				01	半字	00（RA、LT）	0		B_5B_4、B_1B_0
							1		B_4B_5、B_0B_1
						01（LA、RT）	0		B_7B_6、B_3B_2
							1		B_6B_7、B_2B_3
						1x（解封）	0		B_7B_6、B_5B_4、B_3B_2、B_1B_0
							1		B_6B_7、B_4B_5、B_2B_3、B_0B_1

表 92. 编程设定的数据处理（续）

SDW_LOG2 [1:0]	源数据	源数据流(1)	SBX	DDW_LOG2 [1:0]	目标数据	PAM[1:0](2)	DBX	DHX	目标数据流(1)
10	字	$B_7B_6B_5B_4$、$B_3B_2B_1B_0$	0	10	字	xx	0	0	$B_7B_6B_5B_4$、$B_3B_2B_1B_0$
							1		$B_6B_7B_4B_5$、$B_2B_3B_0B_1$
							0	1	$B_5B_4B_7B_6$、$B_1B_0B_3B_2$
							1		$B_4B_5B_6B_7$、$B_0B_1B_2B_3$
			1	00	字节	00（RA、LT）	x	x	B_{12}、B_8、B_4、B_0
						01（LA、RT）			B_{15}、B_{11}、B_7、B_3
						1x（解封）			B_7、B_5、B_6、B_4、B_3、B_1、B_2、B_0
				01	半字	00（RA、LT）	0		B_6B_4、B_2B_0
							1		B_4B_6、B_0B_2
						01（LA、RT）	0		B_7B_5、B_3B_1
							1		B_5B_7、B_1B_3
						1x（解封）	0		B_7B_5、B_6B_4、B_3B_1、B_2B_0
							1		B_5B_7、B_4B_6、B_1B_3、B_0B_2
				10	字	xx	0	0	$B_7B_5B_6B_4$、$B_3B_1B_2B_0$
							1		$B_5B_7B_4B_6$、$B_1B_3B_0B_2$
							0	1	$B_6B_4B_7B_5$、$B_2B_0B_3B_1$
							1		$B_4B_6B_5B_7$、$B_0B_2B_1B_3$

1. 数据流从最低索引 (B_0) 按字节开始适时排序。
2. RA = 右对齐，LA = 左对齐，RT = 右截断，LT = 左截断，0P = 用零填充至目标数据宽度，SE = 符号扩展至目标数据宽度。

15.4.11 GPDMA 传输请求和仲裁

GPDMA 传输请求

如 GPDMA_CxTR2 所定义，设定的 GPDMA 数据传输通过以下任一方式请求：

- 软件请求（如果控制位 SWREQ = 1）：这种方式通常由 CPU 用于在存储器映射的地址之间（存储器之间，GPIO 与存储器之间）进行数据传输
- 来自外设的输入硬件请求（如果 SWREQ = 0）：选择的 GPDMA 硬件外设请求取决于 REQSEL[7:0] 位域（参见*第 15.3.4 节*）。所选硬件请求可为下列之一：
  - 在 GPDMA 模式下配置的外设硬件请求（用于在外设数据寄存器与存储器之间传输）
  - 来自外设的硬件请求，用于从存储器更新其控制寄存器
  - 来自外设的硬件请求，用于读取传输到存储器的状态寄存器

小心： 如果在指定时间，外设将输入硬件外设 GPDMA 请求置为有效，并且每个通道均已准备就绪，可以执行该请求的数据传输，则用户不得通过 GPDMA_CxTR.REQSEL[7:0] 将相同的此类请求分配给两个不同的通道。不会报告用户设置错误。

需要仲裁的 GPDMA 传输请求

在分配的主端口 (GPDMA_CxTR1.SAP) 上将要执行的、已准备就绪的、基于 FIFO 的 GPDMA 源单次传输/突发传输（从源地址到 FIFO），与其他在同一主端口上同时产生请求的 GPDMA 传输，基于通道优先级 (GPDMA_CxCR.PRIO[1:0]) 进行仲裁。

在分配的主端口 (GPDMA_CxTR1.DAP) 上将要执行的、已准备就绪的、基于 FIFO 的 GPDMA 目标单次传输/突发传输（从 FIFO 到目标地址），与其他在同一主端口上同时产生请求的 GPDMA 传输，基于通道优先级 (GPDMA_CxCR.PRIO[1:0]) 进行仲裁。

需要仲裁的 GPDMA 请求的链路传输，包括一次从存储器中的链表数据结构到其中一个链表寄存器（GPDMA_CxTR1、GPDMA_CxTR2、GPDMA_CxBR1、GPDMA_CxSAR、GPDMA_CxDAR 或 GPDMA_CxLLR 以及 GPDMA_CxTR3、GPDMA_CxBR2）的 32 位数据读操作。为了在分配的主端口 (GPDMA_CxCR.LAP) 上进行调度，每个从存储器读取数据的 32 位数据读操作在仲裁时均具有与数据传输相同的通道优先级。

无论请求的数据传输设定为存储器到存储器数据传输的软件请求 (GPDMA_CxTR2.SWREQ = 1)、存储器到外设数据传输或外设到存储器数据传输的硬件请求 (GPDMA_CxTR2.SWREQ = 0)，也无论硬件请求为何种类型，在授予优先权的每个单次传输/突发传输完成之后，都需要重新仲裁。

当硬件请求编程设定为来自目标外设（GPDMA_CxTR2.SWREQ = 0 并且 GPDMA_CxTR2.DREQ = 1）时，（2D/重复）块的第一次存储器读请求（第一个基于 FIFO 的已就绪源突发请求），均由出现的相应选定硬件请求进行门控。。仲裁器不会提前考虑第一个存储器读请求（并非在块传输已使能并且可执行时就立即考虑）。

GPDMA 仲裁

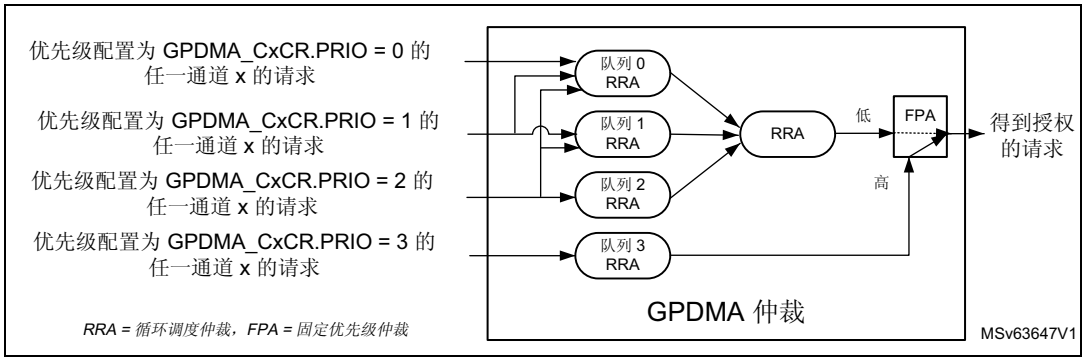
GPDMA 仲裁根据分配的 4 级通道优先级 (GPDMA_CxCR.PRIO[1:0]) 执行。仲裁策略如*图 67* 所示，定义如下：

- 一个专用于被分配为优先级 3 的高优先级数据传输类（队列 3），用来进行时间敏感通道的数据传输

  此数据传输类相对于任何其他低优先级数据传输类具有固定的高优先级仲裁而获得优先权。在该类内，所请求的单次传输/突发传输进行循环调度仲裁。
- 三个低优先级数据通信类（队列 0、1 或 2），用于优先级为 0、1 或 2 的非时间敏感通道

  在该类内，每个请求的单次传输/突发传输均进行循环调度仲裁，并具有根据设定的优先级分配的带宽权重：
  - 优先级为 0 的请求分配给队列 0。
  - 优先级为 1 的请求分配并复制到队列 0 和队列 1。
  - 优先级为 2 的请求分配并复制到队列 0、队列 1 和队列 2。
  - 如果存在同步请求，则任何队列 0、1 或 2 均以循环调度的方式均衡地分配有任何活动输入请求。
  - 此外，对于采用循环调度仲裁器的低优先级通信流量还存在第二个阶段，该仲裁器在来自队列 0、队列 1 和队列 2 的同步选定请求之间公平地进行交替。

图 67. GPDMA 仲裁策略



GPDMA 仲裁和带宽

采用此仲裁策略后，将保证：

- 为优先级相同的请求分配的最大带宽相同
- 为时间敏感请求（优先级为 3）保留带宽（记为 B_{Q3}）
- 在不同的低优先级请求（优先级 0、优先级 1、优先级 2）之间分配剩余加权带宽。

下面两个示例重点说明了加权循环调度仲裁由设定的优先级驱动：

- **示例 1**：具有两个非时间敏感 GPDMA 请求 req0 和 req1 的基本应用。可能的编程设定如下：
  - 如果二者分配的优先级相同，则冲裁器分配给 req0 (B_{req0}) 的带宽**等于**分配给 req1 (B_{req1}) 的带宽。
    $B_{req0} = B_{req1} = 1/2 * (1 - B_{Q3})$
  - 如果分配给 req0 的优先级为 0，分配给 req1 的优先级为 1，则分配给 req0 (B_{P0}) 的带宽是分配给 req1 (B_{P1}) 的带宽**的三分之一**。
    $B_{req0} = B_{P0} = 1/2 * 1/2 * (1 - B_{Q3}) = 1/4 * (1 - B_{Q3})$
    $B_{req1} = B_{P1} = (1/2 + 1) * 1/3 * (1 - B_{Q3}) = 3/4 * (1 - B_{Q3})$
  - 如果分配给 req0 的优先级为 0，分配给 req1 的优先级为 2，则分配给 req0 (B_{P0}) 的带宽是分配给 req1 (B_{P2}) 的带宽**的五分之一**。
    $B_{req0} = B_{P0} = 1/2 * 1/3 * (1 - B_{Q3}) = 1/6 * (1 - B_{Q3})$
    $B_{req1} = B_{P2} = (1/2 + 1 + 1) * 1/3 * (1 - B_{Q3}) = 5/6 * (1 - B_{Q3})$

  上述带宽计算基于理论上的输入请求，这种输入请求对于任何 GPDMA 时钟周期始终有效。这个通过仲裁器计算的带宽必须按应用的请求频率进行加权，该带宽分配结果无法始终有效，在不同的 GPDMA 客户端（例如 400 kHz 的 I2C 与 1 kHz 的 PWM）之间，可能比上述 3 倍和 5 倍的比率变动更大。

- **示例 2**：在该应用中，用户将相同的 N 个非空 GPDMA 请求分配给了每一个非时间敏感的优先级 0、1 和 2。带宽计算如下：
  - 分配给优先级为 0 的一组请求的带宽 (B_{P0}) 为
    $B_{P0} = 1/3 * 1/3 * (1 - B_{Q3}) = 1/9 * (1 - B_{Q3})$
  - 分配给优先级为 1 的一组请求的带宽 (B_{P1})为
    $B_{P1} = (1/3 + 1/2) * 1/3 * (1 - B_{Q3}) = 5/18 * (1 - B_{Q3})$
  - 分配给优先级为 2 的一组请求的带宽 (B_{P2}) 为
    $B_{P2} = (1/3 + 1/2 + 1) * 1/3 * (1 - B_{Q3}) = 11/18 * (1 - B_{Q3})$
  - 分配给 N 个请求中优先级为 Pi（i = 0 到 2）的任意请求 n 的带宽 (B_n) 为 $B_n = 1/N * B_{Pi}$
  - 分配给优先级为 0i 任意请求 n 的带宽 ($B_{n, Pi}$) 为
    $B_{n, P0} = 1/N *1/9 * (1 - B_{Q3})$
    $B_{n, P1} = 1/N *5/18 * (1 - B_{Q3})$
    $B_{n, P2} = 1/N *11/18 * (1 - B_{Q3})$

  在本例中，如果主设备端口总线带宽并不是完全由时间敏感队列 3 使用，则剩余带宽的分配情况为，分配给优先级为 0 的任意请求的带宽，是分配给优先级为 1 的任意请求的带宽的五分之二，是分配给优先级为 2 的任意请求的带宽的十一分之二。

一般而言，假设存在以下请求：

- I 个请求 (I ≥ 0) 分配有优先级 0
  如果 I > 0，这些请求的标记为 i = 0 到 I-1。
- J 个请求 (J ≥ 0) 分配有优先级 1
  如果 J > 0，这些请求的标记为 j = 0 到 J-1。
- K 个请求 (K > 0) 分配有优先级 2
  这些请求标记为 k = 0 到 K-1。
- L 个请求 (L ≥ 0) 分配有优先级 3
  如果 L > 0，这些请求的标记为 l = 0 到 L-1。

由于 B_{Q3} 是为时间敏感请求保留的带宽，优先级为 3 的每个请求 L 的带宽为：

- 当 L > 0 时，$B_l = B_{Q3} / L$（否则：$B_l = 0$）

每个非时间敏感队列的带宽为：

- $B_{Q0} = 1/3 * (1 - B_{Q3})$
- $B_{Q1} = 1/3 * (1 - B_{Q3})$
- $B_{Q2} = 1/3 * (1 - B_{Q3})$

优先级为 0 的一组请求的带宽为：

- $B_{P0} = I / (I + J + K) * B_{Q0}$

每个优先级为 0 的请求 i 的带宽为：

- 当 L > 0 时，$B_i = B_{P0} / I$（否则 $B_{P0} = 0$）

优先级为 0 且路由到队列 0 的一组请求的带宽为：

- $B_{P1,Q0} = J / (I + J + K) * B_{Q0}$

优先级为 1 且路由到队列 1 的一组请求的带宽为：

- $B_{P1,Q1} = J / (J + K) * B_{Q1}$

优先级为 1 的一组请求的总带宽为：

- $B_{P1} = B_{P1,Q0} + B_{P1,Q1}$

每个优先级为 1 的请求 j 的带宽为：

- 当 J > 0 时，$B_j = B_{P1} / J$（否则 $B_j = 0$）

优先级为 2 且路由到队列 0 的一组请求的带宽为：

- $B_{P2,Q0} = K / (I + J + K) * B_{Q0}$

优先级为 2 且路由到队列 1 的一组请求的带宽为：

- $B_{P2,Q1} = K / (J + K) * B_{Q1}$

优先级为 2 且路由到队列 2 的一组请求的带宽为：

- $B_{P2,Q2} = B_{Q2}$

优先级为 2 的一组请求的总带宽为：

- $B_{P2} = B_{P2,Q0} + B_{P2,Q1} + B_{P2,Q2}$

每个优先级为 2 的请求 k 的带宽为：

- $B_k = B_{P2} / K$（通常情况下 K>0）

因此，剩余带宽中最终分配给任意非时间敏感请求 i、j、k 的最大带宽为：

- 通常情况（至少有一个优先级为 2 的请求 k (K > 0)）下：
  - $B_i = 1/I * 1/3 * I/(I + J + K) * (1 - B_{Q3})$
  - $B_j = 1/J * 1/3 *[J/(I + J + K) + J/(J + K)] * (1 - B_{Q3})$
  - $B_k = 1/K * 1/3 *[K/(I + J + K) + K/(J + K) + 1] * (1 - B_{Q3})$
- 特定情况（无优先级为 2 的请求 k (K = 0)）下：
  - $B_i = 1/I * 1/2 * I/(I + J) * (1 - B_{Q3})$
  - $B_j = 1/J * 1/2 *[J/(I + J) + 1] * (1 - B_{Q3})$

因此，GPDMA 仲裁器可用作可编程加权带宽限制器，可用于每个队列，更一般地说是可以用于每个请求/通道。不同的权重从设定的通道优先级单调生成。

15.4.12 GPDMA 触发传输

编程设定的 GPDMA 传输可以由所选输入触发事件的上升沿/下降沿触发，如 GPDMA_CxTR2.TRIGPOL[1:0] 和 GPDMA_CxTR2.TRIGSEL[5:0] 所定义（有关触发信号的选择，请参见*第 15.3.7 节*）。

触发的传输，由 GPDMA_CxTR2.TRIGM[1:0] 中的触发模式定义，可以在 LLI 数据传输级别上进行，以使块的第一个突发读取、通道 x（x = 12 到 15）的 2D/ 重复块的第一个突发读取或每个编程的单个读取受到条件限制。触发模式还可以编程为调节 LLI 链接传输（有关更多详细信息，请参见 GPDMA_CxTR2 中的 TRIGM[1:0]）。

触发命中记录和触发上溢标志生成

当通过以下新的活动触发信号配置使能/加载通道后，GPDMA 会开始对通道 x 的触发信号进行监视：所选触发信号的上升沿或下降沿（分别为 TRIGPOL[1:0] = 01 或 TRIGPOL[1:0] = 10）。

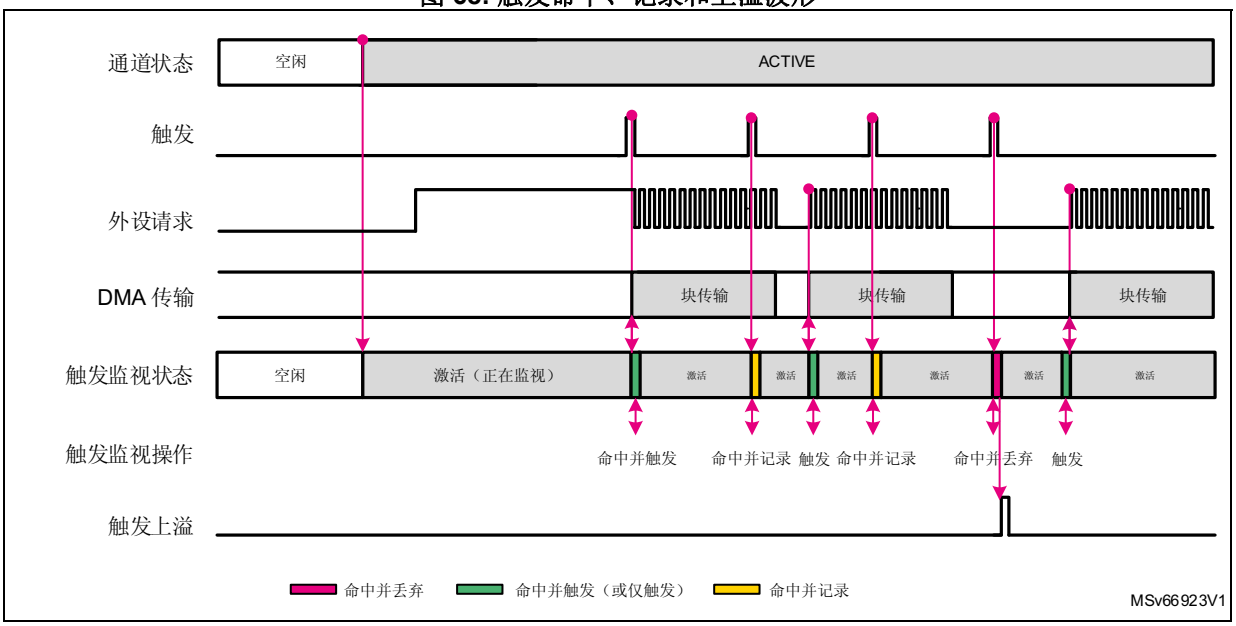
在已触发且未完成的（数据或链路）传输期间，始终会对此触发信号进行监视。如果检测到新的触发信号，则只要定义的上升沿/下降沿和 TRIGSEL[5:0] 未修改，并且通道已使能，就会在内部记录本次命中以准许下一次传输。

传输下一个 LLI_{n+1} 时如果更新了 GPDMA_CxTR2 中 TRIGSEL[5:0] 或 TRIGPOL[1:0] 的任意一个为新值，则复位对先前定义的 LLI_n 触发信号的监视，并丢弃已记录的可能的命中。

小心： 记录第一个新触发信号命中 hit_{n+1} 后，如果检测到另一个触发信号命中 hit_{n+2}，并且 hit_n 所触发的传输仍未完成，则 hit_{n+2} 将会丢失，不会被记录下来。此时，将报告触发信号上溢标志 (GPDMA_CxSR.TOF = 1)，如果已使能，会生成中断（如果 GPDMA_CxCR.TOIE = 1）。由于触发信号上溢，通道不会自动由硬件禁止。

下图说明了块级别触发模式和上升沿触发极性的配置示例中的触发命中、记录和上溢。

图 68. 触发命中、记录和上溢波形



注意： *用户可以为不同的通道分配相同的输入触发事件。这可用于在发生广播触发事件时触发不同的通道。*

15.4.13 GPDMA 循环缓冲与链表编程

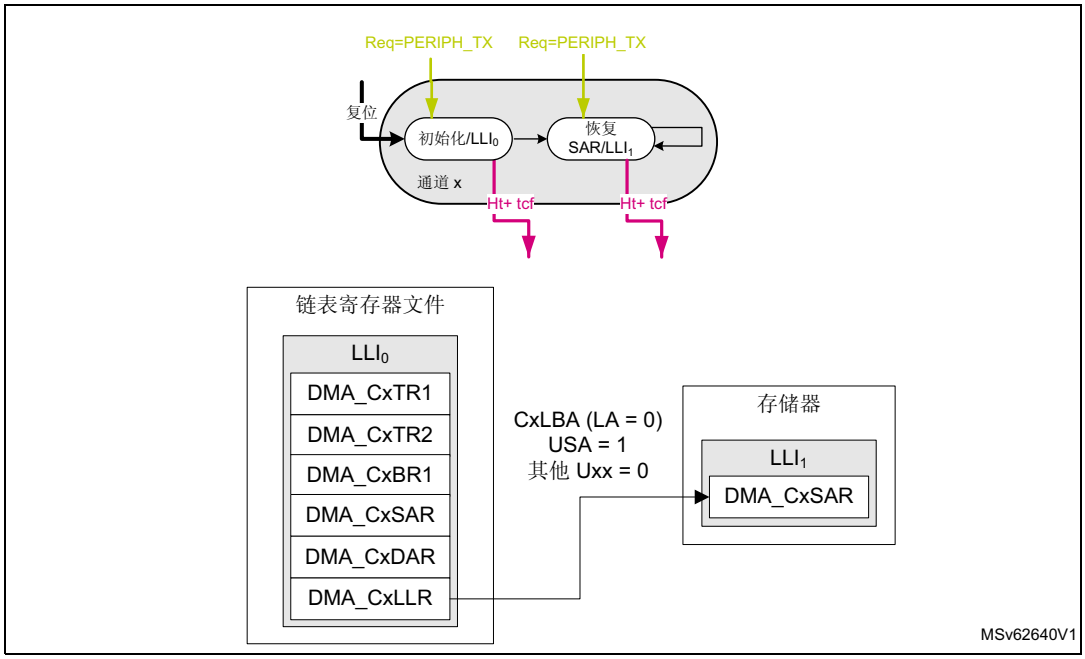
存储器到外设传输以及外设到存储器传输的 GPDMA 循环缓冲（采用线性寻址通道）

要实现循环缓冲，对于从存储器到外设（或外设到存储器）的连续传输，必须由软件建立用来生成半传输和传输完成事件/中断的通道（GPDMA_CxCR.HTIE = 1 且 GPDMA_CxCR.TCIE = 1），以便使能软件对缓冲区的同步处理。

LLI_0 通过线性寻址通道配置为第一个块传输。进行存储器到外设（或外设到存储器的）的传输时，需要连续执行的 LLI_1 来恢复存储器源（或目标）起始地址。当块传输完成后，GPDMA 自动重新加载初始设定的 GPDMA_CxBR1.BNDT[15:0]，无需恢复 GPDMA_CxBR1。

下图说明了采用线性寻址 GPDMA 通道和源循环缓冲的这一编程过程。

图 69. GPDMA 循环缓冲区编程：更新存储器起始地址（通过线性寻址通道）

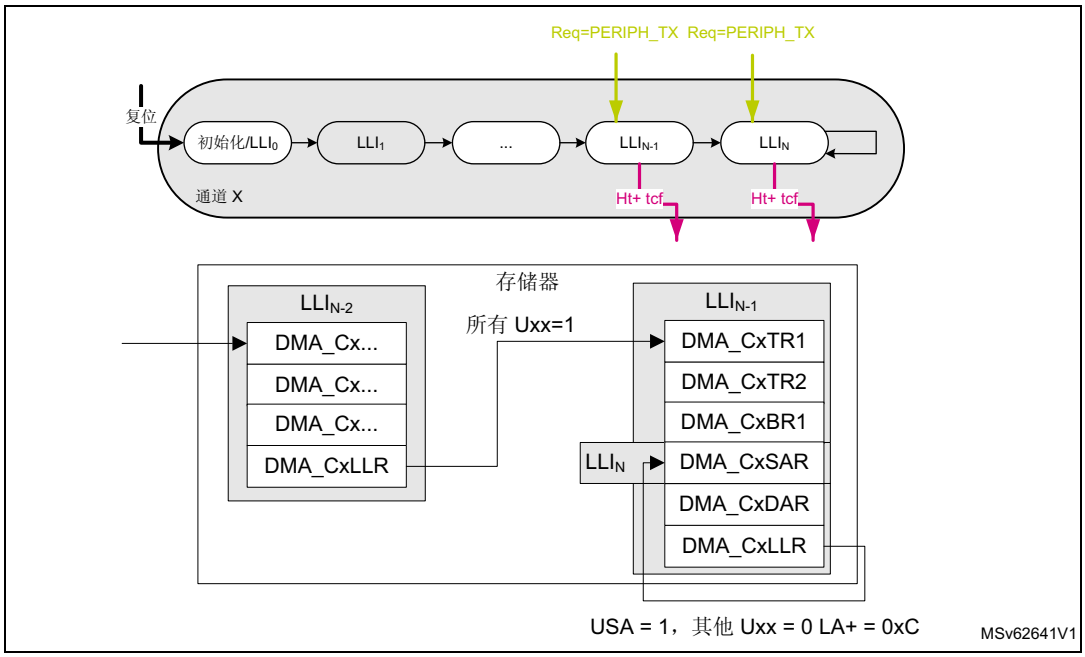


注意： *如果采用 2D 寻址通道，用户可以在 GPDMA_CxBR1.BRC[10:0] = 1 时使用单个 LLI，并通过 GDMA_CxBR2 和 GDMA_CxBR1 设定一个负的存储器块地址偏移，以跳回存储器源或目标起始地址。*

如果在通过共享 GPDMA 通道 x 进行了其他一些传输后必须执行循环缓冲，则需要使用存储器中的倒数第二个 LLI (LLI_{N-1}) 来配置第一个块传输。最后一个 LLI (LLI_N) 可恢复存储器到外设（或外设到存储器）传输中的存储器源（或目标）起始地址。

下图说明了采用线性寻址共享 GPDMA 通道和源循环缓冲的这一编程过程。

图 70. 共享 GPDMA 通道与循环缓冲：通过线性寻址通道更新存储器起始地址



15.4.14 外设流控制模式下的 GPDMA 传输

具有外设流控制模式功能的外设可以决定提前终止 GPDMA 块传输，前提是分配的通道也支持此功能（请参见*第 15.3.6 节*）。

如果相关的 GPDMA 通道 x 也编程设定为外设流控制模式 (GPDMA_CxTR2.PFREQ = 1)：

- GPDMA 块传输的启动方式如下：
  - 如果 GPDMA_CxBR1.BNDT[15:0] ≠ 0，则 GPDMA 硬件在内部使用编程设定的值。
  - 如果 GPDMA_CxBR1.BNDT[15:0] = 0，则 GPDMA 硬件在内部将 64 KB 值用作要传输的最大源块大小。
- 当第一次发生下列情况之一时，GPDMA 块传输即完成：
  - 当 GPDMA_CxBR1.BNDT[15:0] = 0 时
  - 当外设提前终止块时。将生成完成传输事件（如果已编程设定），具体取决于 GPDMA_CxTR2（请参见 *GPDMA 通道 x 传输寄存器 2 (GPDMA_CxTR2)*）。随后，软件可以从源读取当前传输的字节数 (GPDMA_CxBR1.BNDT[15:0])，和/或读取存储器中缓冲区的当前源或目标地址（GPDMA_CxSAR[31:0] 或 GPDMA_CxDAR[31:0]）。

在外设流控制模式下：

- 不支持采用硬件请求传输的目标外设：不支持存储器到外设的传输。
- 不支持来自源外设的数据封装。
- 不支持 2D/ 重复块。
- GPDMA_CxBR1.BNDT[15:0] 必须编程设定为源（外设）突发大小的倍数。

15.4.15 GPDMA 特权/非特权通道

任何通道 x 都是特权或非特权硬件资源，具体取决于特权代理通过 GPDMA_PRIVCFGR.PRIVx 进行的配置。

当通道 x 由特权代理配置为特权状态时，以下访问控制规则适用：

- 对该通道的寄存器位域进行非特权读访问时，将强制返回 0，但 GPDMA_PRIVCFGR 除外，它可由非特权代理读取。
- 对该通道的寄存器位域进行非特权写访问不会产生任何影响。

如果通道配置为特权（或非特权）状态，则源和目标数据传输为通过 AHB 主设备端口进行的特权（或非特权）传输。

当通道配置为特权（或非特权）状态和链表模式时，将下一个链表数据结构从 GPDMA 存储器加载到其寄存器文件时，会通过 GPDMA_CxCR.LAP 分配的主端口以特权（或非特权）传输方式自动执行。

GPDMA 生成一个可反映 GPDMA_PRIVCFGR 的特权总线，以便其他外设始终能获知各个 GPDMA 通道 x 的特权/非特权状态。

如果特权软件必须将通道从特权状态切换到非特权状态，则特权软件必须中止通道，或者等待至特权通道完成传输，之后才能进行切换。如此才能动态地将通道重新分配给下一次非特权传输，因为非特权软件无法这样操作，而是必须先使 GPDMA_CxCR.EN = 0，之后非特权软件才能重新编程 GPDMA_CxCR 以进行下一次传输。特权软件不仅可以复位通道 x (GPDMA_CxCR.RESET = 1)，还可以将完整通道 x 寄存器文件复位为其复位值。

15.4.16 GPDMA 错误管理

根据传输错误的根本原因，GPDMA 将能够管理传输错误并将其上报给用户，如下文所述。

数据传输错误

对源或目标执行总线访问（AHB 单次传输或突发传输）

- 源或目标报告 AHB 错误。
- 设定的通道传输停止（GPDMA_CxCR.EN 由 GPDMA 硬件清零）。通道状态寄存器报告空闲状态 (GPDMA_CxSR.IDLEF = 1) 和数据错误 (GPDMA_CxSR.DTEF = 1)。
- 在发生 GPDMA 数据传输错误后，用户必须执行调试会话，同时需考虑产品所定义的源和目标的存储器映射，包括保护属性。
- 在发生 GPDMA 数据传输错误后，用户必须发出通道复位信号（将 GPDMA_CxCR.RESET 置 1），以复位硬件 GPDMA 通道数据路径和 FIFO 的内容，然后用户再重新使能该通道，以进行下一次传输。

链路传输错误

尝试从存储器中已设定的 LLI 更新 GPDMA 通道寄存器时

- 链表存储器报告 AHB 错误。
- 设定的通道传输停止（GPDMA_CxCR.EN 由 GPDMA 硬件清零），通道状态寄存器报告空闲状态 (GPDMA_CxSR.IDLEF = 1) 和链路错误 (GPDMA_CxSR.ULEF = 1)。
- 在发生 GPDMA 链路错误后，用户必须执行调试会话，同时需考虑产品所定义的链表数据结构的存储器映射（GPDMA_CxLBAR 和 GPDMA_CxLLR），包括保护属性。

- 在发生 GPDMA 链路错误后，用户必须对链表寄存器文件（GPDMA_CxTR1、GPDMA_CxTR2、GPDMA_CxBR1、GPDMA_CxSAR、GPDMA_CxDAR 和 GPDMA_CxLLR 以及 GPDMA_CxTR3 和 GPDMA_CxBR2）执行显式写入，然后用户才能再次使能该通道，以进行下一次传输。

用户设置错误

尝试以未授权用户设置执行 GPDMA 传输时：

- 禁止设定的通道传输（GPDMA_CxCR.EN 由 GPDMA 硬件强制清零），以防执行下一个未授权的设定数据传输。通道状态寄存器报告空闲状态 (GPDMA_CxSR.IDLEF = 1) 和用户设置错误 (GPDMA_CxSR.USEF = 1)。
- 在发生 GPDMA 用户设置错误后，用户必须执行调试会话，同时需考虑 GPDMA 通道编程。用户设置错误可能由以下原因引起：
  - 编程设定的空源块大小未编程为通过下一个 LLI_1 更新该值（GPDMA_CxBR1.BNDT[15:0] = 0 且 GPDMA_CxLLR.UB1 = 0）
  - 编程设定的非空源块大小不是设定的源突发传输数据宽度的倍数（GPDMA_CxBR1.BNDT[2:0] 与 GPDMA_CxTR1.SDW_LOG2[1:0]）
  - 处于封装/解封模式时（如果 PAM[1] = 1），编程设定的非空源块大小不是设定的目标突发传输数据宽度的倍数（GPDMA_CxBR1.BNDT[2:0] 与 GPDMA_CxTR1.DDW_LOG2[1:0]）
  - 编程设定的未对齐源起始地址不是设定的源突发传输数据宽度的倍数（GPDMA_CxSAR[2:0] 与 GPDMA_CxTR1.SDW_LOG2[1:0]）
  - 对于通道 x（x = 6 到 7）：编程设定的未对齐源地址偏移不是设定的源突发传输数据宽度的倍数（GPDMA_CxTR3.SAO[2:0] 与 GPDMA_CxTR1.SDW_LOG2[1:0]）
  - 对于通道 x（x = 6 到 7）：编程设定的未对齐重复块源地址偏移不是设定的源突发传输数据宽度的倍数（GPDMA_CxBR2.BRSAO[2:0] 与 GPDMA_CxTR1.SDW_LOG2[1:0]）
  - 编程设定的未对齐目标起始地址不是设定的目标突发传输数据宽度的倍数（GPDMA_CxDAR[2:0] 与 GPDMA_CxTR1.DDW_LOG2[1:0]）
  - 对于通道 x（x = 6 到 7）：编程设定的未对齐目标地址偏移不是设定的目标突发传输数据宽度的倍数（GPDMA_CxTR3.DAO[2:0] 与 GPDMA_CxTR1.DDW_LOG2[1:0]）
  - 对于通道 x（x=6 到 7）：编程设定的未对齐块重复目标地址偏移不是设定的目标突发传输数据宽度的倍数（GPDMA_CxBR2.BRDAO[2:0] 与 GPDMA_CxTR1.DDW_LOG2[1:0]）
  - 编程设定的双字源数据宽度 (GPDMA_CxTR1.SDW_LOG2[1:0] = 11)
  - 编程设定的双字目标数据宽度 (GPDMA_CxTR1.DDW_LOG2[1:0] = 11)
  - 编程设定的链表项 LLI_{n+1}，数据传输为空值（GPDMA_CxLLR.UB1 = 1 且 GPDMA_CxBR1.BNDT = 0）

15.4.17 GPDMA 自主模式

为了在 GPDMA 执行编程设定的链表传输时节省动态功耗，只要器件处于运行模式或低功耗模式，GPDMA 硬件就会自动管理其本身的时钟门控，并生成传输至 RCC 的时钟请求输出信号，前提是对 RCC 相应的 GPDMA 使能控制位进行了编程。

有关 RCC 编程的详细信息，参见参考手册中的 RCC 章节。

有关 GPDMA 自主特性与器件低功耗模式可用性的更多详细信息，请参见*第 15.3.3 节*。

用户可以利用 GPDMA_CxTR2 在 GPDMA 通道 x 的 LLI_n 级别设定并调度指定 GPDMA 传输的执行，如下所述：

- 软件通过 TRIGM[1:0]、TRIGPOL[1:0]、TRIGSEL[5:0]、SWREQ 和 REQSEL[7:0] 控制并调节传输的输入，以实现输入触发和请求。
- 软件通过 TCEM[1:0] 控制传输的输出并发出是否生成传输完成事件、是否生成相关的数据半传输事件的信号。

更多详细信息，请参见 *GPDMA 通道 x 传输寄存器 2 (GPDMA_CxTR2)*。

在低功耗模式下使用时，该功能可在特定传输完成时通过使能的 GPDMA 传输完成中断 (GPDMA_CxCR.TCIE = 1) 使能 CPU 唤醒或/和允许通过自主 GPDMA 继续在同一个通道中运行另一个 LLI_{n+1} 传输。

如果输出通道 x 传输完成事件 (gpdma_chx_tc) 在 GPDMA 级回送并连接，则可以将该事件编程为通道的选定输入触发信号（参见*第 15.3.7 节*），从而允许以自主方式精细调度 GPDMA 通道间传输，无需将传输完成标志 (TCF) 清零。

在以下任一条件下，指定的 GPDMA 通道 x 会使其时钟请求生效：

- 如果下一个待执行的传输设定为由触发信号调节（GPDMA_CxTR2.TRIGPOL[1:0] 和 GPDMA_CxTR2.TRIGM[1:0]），则仅在触发命中时才会如此操作。
- 如果下一个待执行的传输不由触发信号调节：
  - 如果 GPDMA_CxTR2.SWREQ = 0，则仅当硬件请求通过所选外设置为生效时才会如此操作
  - 如果 GPDMA_CxTR2.SWREQ = 1（存储器之间，GPIO 与存储器之间），则在 GPDMA 使能时立即如此操作

当满足以下所有条件时，GPDMA 通道 x 就会释放其时钟请求：

- 待执行传输完成。
- GPDMA 通道 x 未立即就绪，也未被请求执行下一个传输。
- 如果发生通道 x 中断，可导致此中断的状态寄存器的所有标志都将由软件代理清零。

当一个通道将其时钟请求置为有效，GPDMA 会将其 RCC 时钟请求置为有效。如果没有通道将其时钟请求置为有效，则 GPDMA 会释放 RCC 时钟请求。

15.5 调试模式下的 GPDMA

当微控制器进入调试模式时（内核停止），根据 DBGMCU 模块中的可编程控制位，任意通道 x 都可以独立地继续工作（默认）或挂起。

注意： *在调试模式下，GPDMA_CxSR.SUSPF 不会因 DBGMCU 模块中的可编程控制位引起的挂起而改变。在这种情况下，可以查看 GPDMA_CxSR.IDLEF，了解通道挂起的完成状态。*

15.6 低功耗模式下的 GPDMA

表 93. 低功耗模式对 GPDMA 的影响

模式	说明
睡眠	无影响。GPDMA 中断可使器件退出睡眠模式。
停止(1)	进入停止模式时，GPDMA 寄存器的内容保留。GPDMA 寄存器的内容可以由存储器中的下一个链表项自主更新，以执行自主数据传输。GPDMA 中断可使器件退出停止模式(1)。
待机	GPDMA 掉电，退出待机模式后必须重新初始化。

1. 如需了解支持哪种停止模式，请参见*第 15.3.3 节*。

15.7 GPDMA 中断

每个通道都有一个 GPDMA 中断线，每个 CPU 也分别有一个（如果器件中有多个 CPU）。

表 94. GPDMA 中断请求

中断缩写	中断事件	中断使能	事件标志	事件清除方法
GPDMA_CHx	传输完成	GPDMA_CxCR.TCIE	GPDMA_CxSR.TCF	向 GPDMA_CxFCR.TCF 写入 1
	半传输	GPDMA_CxCR.HTIE	GPDMA_CxSR.HTF	向 GPDMA_CxFCR.HTF 写入 1
	数据传输错误	GPDMA_CxCR.DTEIE	GPDMA_CxSR.DTEF	向 GPDMA_CxFCR.DTEF 写入 1
	更新链路错误	GPDMA_CxCR.ULEIE	GPDMA_CxSR.ULEF	向 GPDMA_CxFCR.ULEF 写入 1
	用户设置错误	GPDMA_CxCR.USEIE	GPDMA_CxSR.USEF	向 GPDMA_CxFCR.USEF 写入 1
	挂起	GPDMA_CxCR.SUSPIE	GPDMA_CxSR.SUSPF	向 GPDMA_CxFCR.SUSPF 写入 1
	触发上溢	GPDMA_CxCR.TOFIE	GPDMA_CxSR.TOF	向 GPDMA_CxFCR.TOF 写入 1

GPDMA 通道 x 事件可以是：

- 传输完成
- 半传输完成
- 传输错误，原因如下：
  - 数据传输错误
  - 更新链路错误
  - 用户设置错误（挂起已完成）
- 触发上溢

注意： *发生通道 x 传输完成事件时，将以一个时钟周期的高电平脉冲形式生成输出信号 gpdma_chx_tc。*

在发生任意 xx 事件后都会生成中断，但必须同时满足以下条件：

- 相应的中断事件 xx 已使能 (GPDMA_CxCR.xxIE = 1)
- 相应的事件标志已清零 (GPDMA_CxSR.xxF = 0)。这意味着，在先前发生相同的 xx 事件后，软件代理必须已将 1 写入相应的 xx 标志清零控制位（将 1 写入 GPDMA_CxFCR.xxF）。

TCF（传输完成）和 HTF（半传输）事件的生成由 GPDMA_CxTR2.TCEM[1:0] 控制，具体如下所述：

- 根据传输完成事件模式 GPDMA_CxTR2.TCEM[1:0]，传输完成事件可以是块传输完成事件、2D/重复传输完成事件或 LLI 传输完成事件，包括下一个 LLI（如果有）上传完成或完整链表传输完成。
- 半传输事件是指半块传输事件或 2D/重复半块传输事件，具体取决于传输完成事件模式 GPDMA_CxTR2.TCEM[1:0]。

  当源块大小字节数的一半（GPDMA_CxBR1.BNDT[15:0] / 2 的舍入整数）传输到目标时，会发生半块传输事件。

  当重复块的一半（(GPDMA_CxBR1.BRC[10:0] + 1) / 2 的舍入整数）传输到目标时，会发生 2D/重复半块传输事件。

更多详细信息，请参见 *GPDMA 通道 x 传输寄存器 2 (GPDMA_CxTR2)*。

在以下情况之一，会出现传输错误：

- 从源进行单次/突发数据传输或到目标的单次/突发传输期间 (DTEF)
- 从存储器中编程设定的 LLI 更新 GPDMA 通道寄存器期间 (ULEF)
- 尝试以未授权设置执行 GPDMA 通道期间 (USEF)

  用户必须执行调试会话，以根据 USEF 根本原因列表更正 GPDMA 通道编程（参见*第 15.4.16 节*）。

有关触发上溢的说明，请参见*触发命中记录和触发上溢标志生成*。

15.8 GPDMA 寄存器

访问 GPDMA 寄存器时，必须使用对齐的 32 位字数据访问。

15.8.1 GPDMA 特权配置寄存器 (GPDMA_PRIVCFGR)

GPDMA privileged configuration register

偏移地址：0x04

复位值：0x0000 0000

该寄存器的写访问必须是特权访问。读访问可以是特权或非特权访问。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN =1 时为只读。

在 GPDMA 通道初始化/关闭时 (GPDMA_CxCR.EN = 0)，该寄存器必须按位编程，以单独将任何通道 x 分配给特权或非特权环境。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIV7	PRIV6	PRIV5	PRIV4	PRIV3	PRIV2	PRIV1	PRIV0
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:0 **PRIVx**：通道 x 的特权状态 (privileged state of channel x)（x = 7 到 0）
0：非特权
1：特权

15.8.2 GPDMA 屏蔽中断状态寄存器 (GPDMA_MISR)

GPDMA masked interrupt status register

偏移地址：0x0C

复位值：0x0000 0000

此寄存器为读寄存器。

此寄存器包含各个通道 x 的屏蔽中断状态位 MISx。它是所有 GPDMA_CxSR 标志的逻辑或运算结果，每个源标志通过相应的 GPDMA_CxCR 中断使能位使能。

将 1 写入相应的 GPDMA_CxFCR 标志清零位时，每个位均由硬件清零。

该寄存器可以混合存储特权和非特权信息，具体取决于各个通道 GPDMA_PRIVCFGR.PRIVx 的特权状态。特权软件可以读取完整的中断状态。非特权软件只能读取非特权通道的状态，读取其他特权位域则返回零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MIS7	MIS6	MIS5	MIS4	MIS3	MIS2	MIS1	MIS0
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **MISx**：通道 x 的屏蔽中断状态 (masked interrupt status of channel x)（x = 7 到 0）
0：通道 x 上未发生中断
1：通道 x 上发生中断

15.8.3 GPDMA 通道 x 链表基址寄存器 (GPDMA_CxLBAR)

GPDMA channel x linked-list base address register

偏移地址：0x50 + 0x80 * x（x = 0 到 7）

复位值：0x0000 0000

该寄存器的写操作必须由特权软件执行。它可以执行特权读操作或不可执行特权读操作，具体取决于通道 x 的特权状态 GPDMA_PRIVCFGR.PRIVx。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN = 1 时为只读。

这个基于通道的寄存器用于存储给定通道 x 的存储区域的链表基址，描述编程设定 GPDMA 传输序列的 LLI 从该基址开始有条件地自动更新。

该 64K 字节对齐的通道 x 链表基址与定义了各个 LLI 字对齐偏移量的 16 位 GPDMA_CxLLR 寄存器偏移地址进行地址组合。

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

位 31:16 **LBA[31:16]**：GPDMA 通道 x 的链表基址 (linked-list base address of GPDMA channel x)

位 15:0 保留，必须保持复位值。

15.8.4 GPDMA 通道 x 标志清零寄存器 (GPDMA_CxFCR)

GPDMA channel x flag clear register

偏移地址：0x5C+ 0x80 * x（x = 0 到 7）

复位值：0x0000 0000

该只写寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TOF	SUSPF	USEF	ULEF	DTEF	HTF	TCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	w	w	w	w	w	w	w								

位 31:15 保留，必须保持复位值。

位 14 **TOF**：触发上溢标志清零 (trigger overrun flag clear)

0：无影响

1：将相应的 TOF 标志清零

位 13 **SUSPF**：挂起完成标志清零 (completed suspension flag clear)
0：无影响
1：将相应的 SUSPF 标志清零

位 12 **USEF**：用户设置错误标志清零 (user setting error flag clear)
0：无影响
1：将相应的 USEF 标志清零

位 11 **ULEF**：更新链路传输错误标志清零 (update link transfer error flag clear)
0：无影响
1：将相应的 ULEF 标志清零

位 10 **DTEF**：数据传输错误标志清零 (data transfer error flag clear)
0：无影响
1：将相应的 DTEF 标志清零

位 9 **HTF**：半传输标志清零 (half transfer flag clear)
0：无影响
1：将相应的 HTF 标志清零

位 8 **TCF**：传输完成标志清零 (transfer complete flag clear)
0：无影响
1：将相应的 TCF 标志清零

位 7:0 保留，必须保持复位值。

15.8.5 GPDMA 通道 x 状态寄存器 (GPDMA_CxSR)

GPDMA channel x status register

偏移地址：0x60 + 0x80 * x（x = 0 到 7）

复位值：0x0000 0001

这是一个只读寄存器，用于报告通道状态。

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFOL[7:0]							
								r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TOF	SUSPF	USEF	ULEF	DTEF	HTF	TCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDLEF
	r	r	r	r	r	r	r								r

位 31:24 保留，必须保持复位值。

位 23:16 **FIFOL[7:0]**：监视到的 FIFO 级别 (monitored FIFO level)
FIFO 中的可用写节拍数，以编程设定的目标数据宽度为单位（若以字节、半字或字为单位，请参见 GPDMA_CxTR1.DDW_LOG2[1:0]）。
注意：将有效传输挂起后，除 GPDMA_CxBR1.BDNT[15:0] 和 GPDMA_CxBR1.BRC[10:0] 外，用户可能还需要读取 FIFOL[7:0]，以了解已传输到目标的数据量。读取前，用户可能需要等待传输挂起完成 (GPDMA_CxSR.SUSPF = 1)。

位 15 保留，必须保持复位值。

位 14 **TOF**：触发上溢标志 (trigger overrun flag)
0：未发生触发上溢事件
1：发生触发上溢事件

位 13 **SUSPF**：挂起完成标志 (completed suspension flag)
0：未发生挂起完成事件
1：发生挂起完成事件

位 12 **USEF**：用户设置错误标志 (user setting error flag)
0：未发生用户设置错误事件
1：发生用户设置错误事件

位 11 **ULEF**：更新链路传输错误标志 (update link transfer error flag)
0：未发生更新链路传输错误事件
1：从存储器更新链表寄存器时发生主控总线错误事件

位 10 **DTEF**：数据传输错误标志 (data transfer error flag)
0：未发生数据传输错误事件
1：数据传输时发生主控总线错误事件

位 9 **HTF**：半传输标志 (half transfer flag)
0：未发生半传输事件
1：发生半传输事件
半传输事件是指半块传输事件或 2D/ 重复半块传输事件，具体取决于传输完成事件模式 (GPDMA_CxTR2.TCEM[1:0])。
当源块大小字节数的一半（GPDMA_CxBR1.BNDT[15:0]/2 的舍入整数）已传输到目标时，会发生半块传输事件。
当重复块的一半（(GPDMA_CxBR1.BRC[10:0] + 1)/2 的舍入整数）传输到目标时，会发生 2D/ 重复半块传输事件。

位 8 **TCF**：传输完成标志 (transfer complete flag)
0：未发生传输完成事件
1：发生传输完成事件
根据传输完成事件模式 GPDMA_CxTR2.TCEM[1:0]，传输完成事件可以是块传输完成事件、2D/ 重复传输完成事件或 LLI 传输完成事件，包括下一个 LLI（如果有）上传完成或完整链表传输完成。

位 7:1 保留，必须保持复位值。

位 0 **IDLEF**：空闲标志 (idle flag)
0：通道不处于空闲状态
1：通道处于空闲状态
当使用有效通道配置（未立即报告 USEF）使能通道 (GPDMA_CxCR.EN = 1) 时，该空闲标志将由硬件清零。
当通道返回到空闲状态（处于挂起或禁止状态）时，该空闲标志在硬复位后置 1 或由硬件置 1。

15.8.6 GPDMA 通道 x 控制寄存器 (GPDMA_CxCR)

GPDMA channel x control register

偏移地址：0x64 + 0x80 * x（x = 0 到 7）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于控制通道（激活、挂起、中止或禁止通道）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIO[1:0]		Res.	Res.	Res.	Res.	LAP	LSM
								rw	rw					rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TOIE	SUSPIE	USEIE	ULEIE	DTEIE	HTIE	TCIE	Res.	Res.	Res.	Res.	Res.	SUSP	RESET	EN
	rw	rw	rw	rw	rw	rw	rw						rw	w	rw

位 31:24 保留，必须保持复位值。

位 23:22 **PRIO[1:0]**：通道 x GPDMA 传输相对于其他传输的优先级 (priority level of the channel x GPDMA transfer versus others)

00：低优先级，低权重
01：低优先级，中等权重
10：低优先级，高权重
11：高优先级

注意：该位必须在 EN = 0 时执行写操作。该位在 EN = 1 时为只读。

位 21:18 保留，必须保持复位值。

位 17 **LAP**：链表分配的端口 (linked-list allocated port)

该位用于分配从存储器更新 GPDMA 链表寄存器时使用的主端口。
0：分配端口 0 (AHB)
1：分配端口 1 (AHB)

注意：该位必须在 EN = 0 时执行写操作。该位在 EN = 1 时为只读。

位 16 **LSM**：链接步进模式 (Link step mode)

0：通道针对完整链表执行并在最后一个 LLI 结束时完成 (GPDMA_CxLLR = 0)。链路地址的 16 个低有效位为空 (LA[15:0] = 0)，所有更新位为空（UT1 = UB1 = UT2 = USA = UDA = ULL = 0 且 UT3 = UB2 = 0）。然后，GPDMA_CxBR1.BNDT[15:0] = 0 且 GPDMA_CxBR1.BRC[10:0] = 0。

1：通道针对当前 LLI 执行**一次**
首先，按照当前内部寄存器文件的定义执行（可能是 1D/重复）块传输，直到 GPDMA_CxBR1.BNDT[15:0] = 0 且 GPDMA_CxBR1.BRC[10:0] = 0。其次，下一个链表数据结构按照 GPDMA_CxLLR 定义有条件地从存储器上传。然后，通道执行完成。

注意：该位必须在 EN = 0 时执行写操作。该位在 EN = 1 时为只读。

位 15 保留，必须保持复位值。

位 14 **TOIE**：触发上溢中断使能 (trigger overrun interrupt enable)

0：禁止中断
1：使能中断

位 13 **SUSPIE**：挂起完成中断使能 (completed suspension interrupt enable)
0：禁止中断
1：使能中断

位 12 **USEIE**：用户设置错误中断使能 (user setting error interrupt enable)
0：禁止中断
1：使能中断

位 11 **ULEIE**：更新链路传输错误中断使能 (update link transfer error interrupt enable)
0：禁止中断
1：使能中断

位 10 **DTEIE**：数据传输错误中断使能 (data transfer error interrupt enable)
0：禁止中断
1：使能中断

位 9 **HTIE**：半传输完成中断使能 (half transfer complete interrupt enable)
0：禁止中断
1：使能中断

位 8 **TCIE**：传输完成中断使能 (transfer complete interrupt enable)
0：禁止中断
1：使能中断

位 7:3 保留，必须保持复位值。

位 2 **SUSP**：挂起 (suspend)
无论向该位 2 写入何值，向位域 RESET（位 1）写入 1 都会导致硬件将该位置为无效。否则：
软件必须写入 1 才能将活动的通道（正通过主端口进行 GPDMA 传输的通道）挂起。
软件必须写入 0 才能恢复挂起的通道，请参照*图 52* 中详述的编程序列。
0：写操作：恢复通道，读操作：通道未挂起
1：写操作：挂起通道，读操作：通道已挂起。

位 1 **RESET**：复位 (reset)
该位为只写位。写入 0 无影响。写入 1 意味着复位 FIFO、通道内部状态、SUSP 和 EN 位（无论向位 2 和位 0 写入何值都是如此）。
当通道处于稳定状态时，复位有效，这意味着以下情况之一：
- 活动通道处于挂起状态（GPDMA_CxSR.SUSPF = 1 且 GPDMA_CxSR.IDLEF = GPDMA_CxCR.EN = 1）
- 通道处于禁止状态（GPDMA_CxSR.IDLEF = 1 且 GPDMA_CxCR.EN = 0）。
对 RESET 执行写操作后，为了继续使用该通道，用户必须在再次使能通道之前以显式方式重新配置该通道，包括硬件修改的配置寄存器（GPDMA_CxBR1、GPDMA_CxSAR 和 GPDMA_CxDAR）。
（请参见*图 53* 中的编程顺序）。
0：无通道复位
1：通道复位

位 0 **EN**：使能 (enable)
无论向该位（位 0）写入何值，向位域 RESET（位 1）写入 1 都会导致硬件将该位置为无效。否则：
当出现传输错误（主控总线错误或用户设置错误）或通道传输完成（通道已做好配置准备，例如：如果在 LLI 单次执行结束时 LSM = 1）时，硬件会将该位清零。
否则，该位可由软件置 1。
将 0 写入该 EN 位的操作将被忽略。
0：写操作：忽略，读操作：通道禁止
1：写操作：使能通道，读操作：通道使能

15.8.7 GPDMA 通道 x 传输寄存器 1 (GPDMA_CxTR1)

GPDMA channel x transfer register 1

偏移地址：0x90 + 0x80 * x（x = 0 到 7）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于控制通道 x 的传输。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN = 1 时为只读。

该寄存器的写操作必须在通道传输完成后执行。硬件随后会将 GPDMA_CxCR.EN 置为无效。通道传输可以在不同的级别完成和编程：块级别、2D/ 重复块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间，如果 GPDMA_CxLLR.UT1 = 1，则该寄存器将由 GPDMA 从存储器自动更新。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	DAP	Res.	Res.	DHX	DBX	DBL_1[5:0]						DINC	Res.	DDW_LOG2[1:0]	
	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SAP	SBX	PAM[1:0]		Res.	SBL_1[5:0]						SINC	Res.	SDW_LOG2[1:0]	
	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw		rw	rw

位 31 保留，必须保持复位值。

位 30 **DAP**：目标分配的端口 (destination allocated port)
该位用于分配目标传输所使用的主端口
0：分配端口 0 (AHB)
1：分配端口 1 (AHB)
注意：该位必须在 EN = 0 时执行写操作。该位在 EN = 1 时为只读。

位 29:28 保留，必须保持复位值。

位 27 **DHX**：目标半字交换 (destination half-word exchange)
如果目标数据大小短于一个字，则忽略该位。
如果目标数据大小是一个字：
0：字中不进行基于半字的交换
1：将每个目标字中两个连续的（后 PAM）半字进行交换。

位 26 **DBX**：目标字节交换 (destination byte exchange)
如果目标数据大小是一个字节，则忽略该位。
如果目标数据大小不是一个字节：
0：半字中不进行基于字节的交换
1：将每个目标半字中两个连续的（后 PAM）字节进行交换。

位 25:20 **DBL_1[5:0]**：目标突发长度减 1（介于 0 和 63 之间）(destination burst length minus 1, between 0 and 63)

突发长度单位是突发内称为节拍的一个数据。如果 DBL_1[5:0] = 0，则突发可称为单次传输。每个数据/节拍的宽度由目标数据宽度 DDW_LOG2[1:0] 定义。

注意：如果突发传输在 AHB 传输时跨过了 1 KB 的地址边界，则 GPDMA 会将设定的突发修改并缩短为较小长度的单次传输或突发传输以符合 AHB 协议。

如果突发传输的长度大于通道 x 的 FIFO 大小，则 GPDMA 会修改编程设定的突发并将其缩短为较短长度的单次传输或突发传输，以符合 FIFO 大小。虽然 GPDMA 在有效传输与较短长度的单次传输/突发传输之间进行重新仲裁会导致传输性能相对较低，但可以保证数据的完整性。

位 19 **DINC**：目标增量式突发传输 (destination incrementing burst)

0：固定突发传输

1：连续增量式突发传输

GPDMA_CxDAR 指向的目标地址在突发节拍/单次传输后保持不变，或者在突发节拍/单次传输后增加对应于连续数据的偏移值。

位 18 保留，必须保持复位值。

位 17:16 **DDW_LOG2[1:0]**：突发传输目标数据宽度（以字节为单位）的二进制对数 (binary logarithm of the destination data width of a burst, in bytes)

00：字节

01：半字（2 字节）

10：字（4 字节）

11：报告用户设置错误且不会发出任何传输信号

注意：设置 8 字节数据宽度会导致报告用户设置错误，并且不会发出任何传输信号。

目标突发传输的地址必须与其数据宽度对齐（起始地址 GPDMA_CxDAR[2:0] 和偏移地址 GPDMA_CxTR3.DAO[2:0] 与 DDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

位 15 保留，必须保持复位值。

位 14 **SAP**：源分配的端口 (source allocated port)

该位用于分配源传输的主端口

0：分配端口 0 (AHB)

1：分配端口 1 (AHB)

注意：该位必须在 EN = 0 时执行写操作。该位在 EN = 1 时为只读。

位 13 **SBX**：每个源字的未对齐半字内的源字节交换 (source byte exchange within the unaligned half-word of each source word)

如果源数据宽度短于一个字，则忽略该位。

如果源数据宽度是一个字：

0：每个源字的未对齐半字内不进行基于字节的交换

1：将每个源字的未对齐半字内两个连续的字节进行交换。

位 12:11 **PAM[1:0]**：填充/对齐模式 (padding/alignment mode)

如果 DDW_LOG2[1:0] = SDW_LOG2[1:0]：如果突发目标传输的数据宽度等于突发源传输的数据宽度，则忽略这些位。

否则，在以下枚举值中，条件 PAM_1 表示目标数据宽度大于源数据宽度，条件 PAM_2 表示源数据宽度大于目标数据宽度。

条件：PAM_1

00：源数据以右对齐方式传输，并用 0 填充至目标数据宽度

01：源数据以右对齐方式传输，符号扩展至目标数据宽度

10-11：在目标传输之前，连续的源数据按照从左 (LSB) 到右 (MSB) 的顺序（称为小端序）进入 FIFO 队列并封装至目标数据宽度

条件：PAM_2

00：源数据以右对齐方式传输，左截断至目标数据宽度

01：源数据以左对齐方式传输，右截断至目标数据宽度

10-11：源数据进入 FIFO 队列并解封装至目标数据宽度，按照从左 (LSB) 到右 (MSB) 的顺序（称为小端序）传输到目标

注意：如果传输来自配置为外设流控制模式（SWREQ = 0、PFREQ = 1 且 DREQ = 0）的源外设，并且目标数据宽度 > 源数据宽度，则不支持封装。

位 10 保留，必须保持复位值。

位 9:4 **SBL_1[5:0]**：源突发长度减 1（介于 0 和 63 之间）(source burst length minus 1, between 0 and 63)

突发长度单位是突发内称为节拍的一个数据。如果 SBL_1[5:0] = 0，则突发可称为单次传输。每个数据/节拍的宽度由目标数据宽度 SDW_LOG2[1:0] 定义。

注意：如果突发传输在 AHB 传输时跨过了 1 KB 的地址边界，则 GPDMA 会将设定的突发修改并缩短为较小长度的单次传输或突发传输以符合 AHB 协议。

如果突发传输的长度大于通道 x 的 FIFO 大小，则 GPDMA 会修改编程设定的突发并将其缩短为较短长度的单次传输或突发传输，以符合 FIFO 大小。虽然 GPDMA 在有效传输与较短长度的单次传输/突发传输之间进行重新仲裁会导致传输性能相对较低，但可以保证数据的完整性。

位 3 **SINC**：源增量式突发传输 (source incrementing burst)

0：固定突发传输

1：连续增量式突发传输

GPDMA_CxSAR 指向的源地址在突发节拍/单次传输后保持不变，或者在突发节拍/单次传输后增加对应于连续数据的偏移值。

位 2 保留，必须保持复位值。

位 1:0 **SDW_LOG2[1:0]**：突发传输源数据宽度（以字节为单位）的二进制对数 (binary logarithm of the source data width of a burst in bytes)

00：字节

01：半字（2 字节）

10：字（4 字节）

11：报告用户设置错误且不会发出任何传输信号

注意：设置 8 字节数据宽度会导致报告用户设置错误，并且不会发出任何传输信号。
源块大小必须是源数据宽度的倍数（GPDMA_CxBR1.BNDT[2:0] 与 SDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。
源突发传输的地址必须与其数据宽度对齐（起始地址 GPDMA_CxSAR[2:0] 与 SDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

15.8.8 GPDMA 通道 x 传输寄存器 2 (GPDMA_CxTR2)

GPDMA channel x transfer register 2

偏移地址：0x94 + 0x80 * x（x = 0 到 7）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于控制通道 x 的传输。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN = 1 时为只读。

该寄存器的写操作必须在通道传输完成后执行（硬件将 GPDMA_CxCR.EN 清零）。通道传输可以在不同的级别完成和编程：块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间，如果 GPDMA_CxLLR.UT2 = 1，则该寄存器将由 GPDMA 从存储器自动更新。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCEM[1:0]		Res.	Res.	Res.	Res.	TRIGPOL[1:0]		Res.	Res.	TRIGSEL[5:0]					
rw	rw					rw	rw			rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGM[1:0]		Res.	PFREQ	BREQ	DREQ	SWREQ	Res.	REQSEL[7:0]							
rw	rw		rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw

位 31:30 **TCEM[1:0]**：传输完成事件模式 (transfer complete event mode)

这些位定义了传输完成和半传输完成事件生成对应的传输粒度。

00：在块级别（当 GPDMA_CxBR1.BNDT[15:0] = 0 时）：传输（和半传输）完成事件在块传输结束（或传输一半）时生成。

注意：如果初始 LLI_0 数据传输为空/无效（在 GPDMA_CxBR1.BNDT[15:0] = 0 时由内部寄存器文件直接编程），则传输完成事件和半传输完成事件均不会生成。

01：通道 x（x = 0 到 5），与 00 相同；通道 x（x = 6 到 7），在 2D/ 重复块级别（当 GPDMA_CxBR1.BRC[10:0] = 0 且 GPDMA_CxBR1.BNDT[15:0] = 0 时）。传输（和半传输）完成事件在 2D/ 重复块传输结束（或传输一半）时生成。

注意：如果初始 LLI_0 数据传输为空/无效（在 GPDMA_CxBR1.BNDT[15:0] = 0 时由内部寄存器文件直接编程），则传输完成事件和半传输完成事件均不会生成。

10：在 LLI 级别：传输完成事件在 LLI 传输结束时生成，包括 LLI 更新（如有）。半传输事件在 LLI 数据传输一半时生成。LLI 数据传输是通道 x（x = 6 到 7）的块传输或 2D/ 重复块传输（如果有数据传输）。

注意：如果初始 LLI_0 数据传输为空/无效（在 GPDMA_CxBR1.BNDT[15:0] = 0 时由内部寄存器文件直接编程），则半传输事件不会生成，传输完成事件在 LLI_1 加载完成后生成。

11：在通道级别：传输完成事件在最后一个 LLI 传输结束时生成。半传输事件在最后一个 LLI 数据传输一半时生成。最后一个 LLI 会将链路地址 GPDMA_CxLLR.LA[15:2] 更新为零，并将所有 GPDMA_CxLLR 更新位（UT1、UT2、UB1、USA、UDA 和 ULL 以及 UT3 和 UB2）清零。如果通道传输是连续的/无限的，则不会生成任何事件。

位 29:26 保留，必须保持复位值。

位 25:24 **TRIGPOL[1:0]**：触发事件极性 (trigger event polarity)

这些位用于定义由 TRIGSEL[5:0] 所定义的所选输入触发事件的极性。

00：无触发（屏蔽触发事件）

01：上升沿触发

10：下降沿触发

11：与 00 相同

位 23:22 保留，必须保持复位值。

位 21:16 **TRIGSEL[5:0]**：触发事件输入选择 (trigger event input selection)

这些位用于选择 GPDMA 传输的触发事件输入（参见*第 15.3.7 节*），如果 TRIGPOL[1:0] ≠ 00，还会选择有效的触发事件。

位 15:14 **TRIGM[1:0]**：触发模式 (trigger mode)

这些位用于定义各传输粒度的触发条件。

如果在 TRIGPOL[1:0] = 00 或 11 时使能通道 x（GPDMA_CxCR.EN 置 1），将忽略这些 TRIGM[1:0] 位。

否则，GPDMA 传输的条件是至少有一次触发命中：

00：在块级别：每个块传输的首个突发读操作的条件是一次命中触发（对于通道 x = 12 到 15，如果通过 GPDMA_CxBR1.BRC[10:0] ≠ 0 配置了 2D/ 重复块，则此条件适用于每个块）。

01：通道 x（x = 0 到 5），与 00 相同；通道 x（x = 6 到 7），在 2D/ 重复块级别。2D/ 重复块传输的首个突发读操作的条件是一次触发命中。

10：在链路级别：LLI 链路传输的条件是一次命中触发。LLI 数据传输（如有）没有条件。

11：在编程设定的突发级别：如果 SWREQ = 1，则每个编程设定的突发读操作的条件是一次命中触发。如果 SWREQ = 0，则所选外设请求的每个编程设定的突发传输的条件是一次命中触发。

– 如果外设编程设定为 LLI 数据传输的源 (DREQ = 0)，则每个编程设定的突发读操作都受此条件限制。

– 如果外设编程设定为 LLI 数据传输的目标 (DREQ = 1)，则每个编程设定的突发写操作都受此条件限制。（可能是 2D/ 重复）块的首个存储器突发读操作（也称为首个基于 FIFO 的就绪源突发），由硬件请求和首个触发命中进行门控。

当通过以下新的活动触发信号配置使能 / 加载通道后，GPDMA 会开始对通道 x 的触发信号进行监视：所选触发信号的上升沿或下降沿（分别为 TRIGPOL[1:0] = 01 或 TRIGPOL[1:0] = 10）。

在已触发且未完成的（数据或链路）传输期间，始终会对此触发信号进行监视；如果检测到新的触发信号，则只要定义的上升沿或下降沿和 TRIGSEL[5:0] 均未修改，并且通道已使能，就会在内部记录本次命中以准许下一次传输。

如果对于 TRIGSEL[5:0] 或 TRIGPOL[1:0] 的任何值，传输下一个 LLI_{n+1} 都会将 GPDMA_CxTR2 更新为新值，则可以复位对先前定义的 LLI_n 触发信号的监视，并丢弃已记录的命中。

记录第一个新触发信号命中 hit_{n+1} 后，如果检测到另一个触发信号命中 hit_{n+2}，并且 hit_n 所触发的传输仍未完成，则 hit_{n+2} 将会丢失，不会被记录下来。此时，将报告触发信号上溢标志 (GPDMA_CxSR.TOF = 1)，如果已使能，会生成中断（GPDMA_CxCR.TOIE = 1）。由于触发信号上溢，通道不会自动由硬件禁止。

注意：当源块大小不是源突发大小的倍数，而是源数据宽度的倍数时，最后一个编程设定的源突发传输不会完成，并且其大小会在内部缩小以匹配块大小。在这种情况下，如果 TRIGM[1:0] = 11 且（SWREQ = 1 或（SWREQ = 0 且 DREQ = 0）），则缩小的突发传输（通过较短长度的单次传输或 / 和突发传输）的条件是一次触发。

当编程设定的目标突发在内部通过较短长度的单次传输或 / 和突发传输（相对于 FIFO 大小、块大小、1 KB 边界地址跨越）缩小时：如果编程设定的目标突发传输的条件是触发（如果 TRIGM[1:0] = 11、SWREQ = 0 且 DREQ = 1），则此缩小的目标突发传输的条件是一次触发。

位 13 保留，必须保持复位值。

位 12 **PFREQ**：外设流控制模式下的硬件请求 (Hardware request in peripheral flow control mode)

重要提示：如果给定的通道 x 未通过此功能实现，则该位将保留且 PFREQ 不存在（有关使用此功能实现的通道列表，请参见第 15.3.2 节）。

如果通道 x 在 SWREQ = 1（存储器到存储器数据传输的软件请求）时激活（GPDMA_CxCR.EN 置 1），则将忽略该位。否则：

0：所选硬件请求在 GPDMA 控制模式下由外设根据硬件请求/确认协议驱动。GPDMA 通过 GPDMA_CxTR1.BNDT[15:0] 编程，硬件在内部使用该配置来完成块传输。

1：所选硬件请求在外设控制模式下由外设根据硬件请求/确认协议驱动。GPDMA 块传输可由外设本身提前完成（更多详细信息，请参见第 15.3.6 节）。

注意：在外设流控制模式下，具有以下限制：

- 不支持 2D/ 重复块（GPDMA_CxBR1.BRC[10:0] 必须设置为 0）

- 外设必须设置为传输源 (DREQ = 0)。

- 不支持将数据封装为较大的目标数据宽度（如果目标数据宽度 > 源数据宽度，则 GPDMA_CxTR1.PAM[1] 必须设置为 0）。

- GPDMA_CxBR1.BNDT[15:0] 必须编程设定为源（外设）突发大小的倍数。

位 11 **BREQ**：块硬件请求 (block hardware request)

如果通道 x 在 SWREQ = 1（存储器到存储器数据传输的软件请求）时激活（GPDMA_CxCR.EN 置 1），则将忽略该位。否则：

0：所选硬件请求在突发传输级别由外设根据硬件请求/确认协议驱动。

1：所选硬件请求在块级别由外设根据硬件请求/确认协议驱动（参见第 15.3.4 节）。

位 10 **DREQ**：目标硬件请求 (destination hardware request)

如果通道 x 在 SWREQ = 1（存储器到存储器数据传输的软件请求）时激活（GPDMA_CxCR.EN 置 1），则将忽略该位。否则：

0：所选硬件请求由源外设驱动（GPDMA 传输调度器通过源/读端口接收请求信号）

1：所选硬件请求由目标外设驱动（GPDMA 传输调度器通过目标/写端口接收请求信号）

注意：如果通道 x 在 SWREQ = 0 且 PFREQ = 1（外设流控制模式下的外设硬件请求）时激活（GPDMA_CxCR.EN 置 1），则对该 DREQ 位的任何软件置 1 都将被忽略：在外设流控制模式下，仅支持外设到存储器的传输。

位 9 **SWREQ**：软件请求 (software request)

当 GPDMA_CxCR.EN 置 1 时，在内部会考虑该位。

0：无软件请求。考虑所选的硬件请求 REQSEL[7:0]。

1：存储器到存储器数据传输的软件请求。将忽略根据 REQSEL[7:0] 默认选择的硬件请求。

位 8 保留，必须保持复位值。

位 7:0 **REQSEL[7:0]**：GPDMA 硬件请求选择 (GPDMA hardware request selection)

如果通道 x 在 SWREQ = 1（存储器到存储器数据传输的软件请求）时激活（GPDMA_CxCR.EN 置 1），则将忽略该位。否则，将根据第 15.3.4 节在内部考虑所选硬件请求。

小心： 用户不得向不同的活动 GPDMA 通道（对于这些通道，GPDMA_CxCR.EN = 1 且 GPDMA_CxTR2.SWREQ = 0）分配相同的输入硬件请求（相同的 REQSEL[7:0] 值）。GPDMA 的设计并不能在硬件上支持同时使能的通道不当配置有同一硬件外设请求信号的情况，并且不会报告用户设置错误。

15.8.9 GPDMA 通道 x 块寄存器 1 (GPDMA_CxBR1)

GPDMA channel x block register 1

偏移地址：0x98 + 0x80 * x（x = 0 到 5）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于控制通道 x 在块级别的传输。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN = 1 时为只读。

该寄存器的写操作必须在通道 x 传输完成后执行（然后，硬件将 GPDMA_CxCR.EN 清零）。通道传输可以在不同的级别完成和编程：块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间：

- 如果 GPDMA_CxLLR.UB1 = 1，该寄存器由 GPDMA 从存储器中的下一个 LLI 自动更新。
- 如果 GPDMA_CxLLR.UB1 = 0 且如果至少有一个链表寄存器将从存储器中的下一个 LLI 更新，则该寄存器在内部自动恢复为 BNDT[15:0] 位域的编程值。
- 如果所有更新位 GPDMA_CxLLR.Uxx 均为空并且 GPDMA_CxLLR.LA[15:0] ≠ 0，则当前 LLI 为最后一个 LLI 且连续执行：在每次执行此最后的 LLI 之后，该寄存器在内部自动恢复为 BNDT[15:0] 的编程值
- 如果 GPDMA_CxLLR = 0，该寄存器和 BNDT[15:0] 保持为空，通道 x 传输完成。

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

位 31:16 保留，必须保持复位值。

位 15:0 **BNDT[15:0]**：要从源传输的块数据字节数 (block number of data bytes to transfer from the source)

从源传输的块大小。当通道使能后，该位域变为只读并递减，指示当前待传输源块中的剩余数据项数。BNDT[15:0] 以字节数进行编程，最大源块大小为 64 KB - 1。

在最后一次数据传输完成后 (BNDT[15:0] = 0)：

- 如果 GPDMA_CxLLR.UB1 = 1，则该位域按存储器中的 LLI 更新。
- 如果 GPDMA_CxLLR.UB1 = 0 且至少有一个非空 Uxx 更新位，则该位域在内部恢复为编程值。
- 如果所有 GPDMA_CxLLR.Uxx = 0 且 GPDMA_CxLLR.LA[15:0] = 0，则该位域在内部恢复为编程值（最后一个 LLI 传输无限/连续）。
- 如果 GPDMA_CxLLR = 0，则该位域在最后一个 LLI 数据传输后保持为零。

注意：非空源块大小必须是源数据宽度的倍数（BNDT[2:0] 与 GPDMA_CxTR1.SDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

配置为封装模式（GPDMA_CxTR1.PAM[1] = 1 并且目标数据宽度与源数据宽度不同）时，非空源块大小必须是目标数据宽度的倍数（BNDT[2:0] 与 GPDMA_CxTR1.DDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

15.8.10 GPDMA 通道 x 复用块寄存器 1 (GPDMA_CxBR1)

GPDMA channel x alternate block register 1

偏移地址：0x98 + 0x80 * x（x = 6 到 7）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于控制通道 x 在块级别的传输。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN = 1 时为只读。

该寄存器的写操作必须在通道 x 传输完成后执行（然后，硬件将 GPDMA_CxCR.EN 清零）。通道传输可以在不同的级别完成和编程：块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间：

- 如果 GPDMA_CxLLR.UB1 = 1，该寄存器由 GPDMA 从存储器中的下一个 LLI 自动更新。
- 如果 GPDMA_CxLLR.UB1 = 0 且如果至少有一个链表寄存器将从存储器中的下一个 LLI 更新，则该寄存器在内部自动恢复为 BNDT[15:0] 和 BRC[10:0] 位域的编程值。
- 如果所有更新位 GPDMA_CxLLR.Uxx 均为空并且 GPDMA_CxLLR.LA[15:0] ≠ 0，则当前 LLI 为最后一个 LLI 且连续执行：在每次执行此最后的 LLI 之后，该寄存器在内部自动恢复为 BNDT[15:0] 和 BRC[10:0] 位域的编程值。
- 如果 GPDMA_CxLLR = 0，BNDT[15:0] 和 BRC[10:0] 保持为空，通道 x 传输完成。

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

位 31 **BRDDEC**：块重复目标地址递减 (Block repeat destination address decrement)

0：块传输结束后，通过将编程设定的偏移量 GPDMA_CxBR2.BRDAO 与当前 GPDMA_CxDAR 值（当前目标地址）相加来更新 GPDMA_CxDAR 寄存器

1：块传输结束后，通过从当前 GPDMA_CxDAR 值（当前目标地址）减去编程设定的偏移量 GPDMA_CxBR2.BRDAO 来更新 GPDMA_CxDAR 寄存器

注意：除此增量/减量（取决于 BRDDEC）外，GPDMA_CxDAR 同时还会通过 GPDMA_CxTR3.DAO 值的增量/减量（取决于 DDEC）进行更新，因为该更新通常会在每个编程设定的突发传输结束后执行。

位 30 **BRSDEC**：块重复源地址递减 (Block repeat source address decrement)

0：块传输结束后，通过将编程设定的偏移量 GPDMA_CxBR2.BRSAO 与当前 GPDMA_CxSAR 值（当前源地址）相加来更新 GPDMA_CxSAR 寄存器

1：块传输结束后，通过从当前 GPDMA_CxSAR 值（当前源地址）减去编程设定的偏移量 GPDMA_CxBR2.BRSAO 来更新 GPDMA_CxSAR 寄存器

注意：除此增量/减量（取决于 BRSDEC）外，GPDMA_CxSAR 同时还会通过 GPDMA_CxTR3.SAO 值的增量/减量（取决于 SDEC）进行更新，因为该更新会在所有编程设定的突发传输结束后执行。

位 29 **DDEC**：目标地址递减 (destination address decrement)

0：编程设定的到目标的突发传输结束后，通过将编程设定的偏移量 GPDMA_CxTR3.DAO 与当前 GPDMA_CxDAR 值（当前目标地址）相加来更新 GPDMA_CxDAR 寄存器

1：编程设定的到目标的突发传输结束后，通过从当前 GPDMA_CxDAR 值（当前目标地址）减去编程设定的偏移量 GPDMA_CxTR3.DAO 来更新 GPDMA_CxDAR 寄存器

位 28 **SDEC**：源地址递减 (source address decrement)

0：编程设定的从源进行的突发传输结束后，通过将编程设定的偏移量 GPDMA_CxTR3.SAO 与当前 GPDMA_CxSAR 值（当前源地址）相加来更新 GPDMA_CxSAR 寄存器

1：编程设定的从源进行的突发传输结束后，通过从当前 GPDMA_CxSAR 值（当前源地址）减去编程设定的偏移量 GPDMA_CxTR3.SAO 来更新 GPDMA_CxSAR 寄存器

位 27 保留，必须保持复位值。

位 26:16 **BRC[10:0]**：块重复计数器 (Block repeat counter)

该位域包含当前块的重复数（0 到 2047）。

当通道使能后，该位域变为只读。递减后，该位域指示剩余块数（不包括当前块）。在每个块传输完成后，该计数器将由硬件递减。

在最后一个块传输完成后 (BRC[10:0] = BNDT[15:0] = 0)：

– 如果 GPDMA_CxLLR.UB1 = 1，则所有 GPDMA_CxBR1 位域均按存储器中的下一个 LLI 更新。

– 如果 GPDMA_CxLLR.UB1 = 0 且至少有一个非空 Uxx 更新位，则该位域在内部恢复为编程值。

– 如果所有 GPDMA_CxLLR.Uxx = 0 且 GPDMA_CxLLR.LA[15:0] ≠ 0，则该位域在内部恢复为编程值（最后一个 LLI 传输无限/连续）。

– 如果 GPDMA_CxLLR = 0，则该位域在最后一个 LLI 和数据传输后保持为零。

位 15:0 **BNDT[15:0]**：要从源传输的块数据字节数 (block number of data bytes to transfer from the source)

从源传输的块大小。当通道使能后，该位域变为只读并递减，指示当前待传输源块中的剩余数据项数。BNDT[15:0] 以字节数进行编程，最大源块大小为 64 KB - 1。

在最后一次数据传输完成后 (BNDT[15:0] = 0)：

- 如果 GPDMA_CxLLR.UB1 = 1，则该位域按存储器中的 LLI 更新。

- 如果 GPDMA_CxLLR.UB1 = 0 且至少有一个非空 Uxx 更新位，则该位域在内部恢复为编程值。

- 如果所有 GPDMA_CxLLR.Uxx = 0 且 GPDMA_CxLLR.LA[15:0] ≠ 0，则该位域在内部恢复为编程值（最后一个 LLI 传输无限/连续）。

- 如果 GPDMA_CxLLR = 0，则该位域在最后一个 LLI 数据传输后保持为零。

注意：非空源块大小必须是源数据宽度的倍数（BNDT[2:0] 与 GPDMA_CxTR1.SDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

配置为封装模式（GPDMA_CxTR1.PAM[1] = 1 并且目标数据宽度与源数据宽度不同）时，非空源块大小必须是目标数据宽度的倍数（BNDT[2:0] 与 GPDMA_CxTR1.DDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

15.8.11 GPDMA 通道 x 源地址寄存器 (GPDMA_CxSAR)

GPDMA channel x source address register

偏移地址：0x9C + 0x80 * x（x = 0 到 7）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于配置源中传输的起始地址。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

当 GPDMA_CxCR.EN = 1 时，该寄存器为只读，并由硬件连续更新，以反映从源进行的下一个突发传输的地址。

该寄存器的写操作必须在通道传输完成后执行（然后，硬件将 GPDMA_CxCR.EN 清零）。通道传输可以在不同的级别完成和编程：块级别、2D/ 重复块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间，如果 GPDMA_CxLLR.USA = 1，则该寄存器将由 GPDMA 从存储器自动更新。

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

位 31:0 **SA[31:0]**：源地址 (source address)

该位域指向待读取的下一个数据对应的地址。

在通道活动期间，根据源寻址模式 (GPDMA_CxTR1.SINC)，该位域在每个源数据突发传输之后保持不变或按数据宽度 (GPDMA_CxTR1.SDW_LOG2[1:0]) 递增，以反映待读取的下一个数据对应的地址。

在通道活动期间，该地址在每次源突发传输完成后更新，因此：

– 编程设定的源突发；固定寻址模式或连续数据增量式寻址模式。在连续增量式寻址模式下 (GPDMA_CxTR1.SINC = 1)，附加地址偏移值是编程设定的突发大小（由 GPDMA_CxTR1.SBL_1[5:0] 和 GPDMA_CxTR1.SDW_LOG2[21:0] 定义）

– 附加的源递增 / 递减偏移值由 GPDMA_CxBR1.SDEC 和 GPDMA_CxTR3.SAO[12:0] 编程设定。

– 源块传输完成后，对于具有 2D 寻址能力的通道 x（x = 6 到 7），附加的块重复源递增 / 递减偏移值由 GPDMA_CxBR1.BRSDEC 和 GPDMA_CxBR2.BRSAO[15:0] 编程设定

在链表模式下，在 LLI 数据传输完成后，如果在 GPDMA_CxLLR.USA = 1 时设置 LLI，则该寄存器会由 GPDMA 从存储器自动更新。

注意：源地址必须与编程设定的源突发传输数据宽度对齐（SA[2:0] 与 GPDMA_CxTR1.SDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

当源块大小不是源突发大小的倍数，而是源数据宽度的倍数时，最后一个编程设定的源突发传输不会完成，并且其大小会在内部缩小以匹配块大小。在这种情况下，不应用附加的 GPDMA_CxTR3.SAO[12:0]。

15.8.12 GPDMA 通道 x 目标地址寄存器 (GPDMA_CxDAR)

GPDMA channel x destination address register

偏移地址：0xA0 + 0x80 * x（x = 0 到 7）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于配置传输的目标起始地址。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

当 GPDMA_CxCR.EN = 1 时，该寄存器为只读，并由硬件连续更新，以反映下一个到目标的突发传输的地址。

该寄存器的写操作必须在通道传输完成后执行（然后，硬件将 GPDMA_CxCR.EN 清零）。通道传输可以在不同的级别完成和编程：块级别、2D/ 重复块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间，如果 GPDMA_CxLLR.UDA = 1，则该寄存器将由 GPDMA 从存储器自动更新。

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

位 31:0 **DA[31:0]**：目标地址 (destination address)

该位域指向待写入下一个数据的地址。

在通道活动期间，根据目标寻址模式 (GPDMA_CxTR1.DINC)，该位域在每个目标突发数据传输之后保持不变或按数据宽度 (GPDMA_CxTR1.DDW_LOG2[21:0]) 递增，以反映待写入下一个数据的地址。

在通道活动期间，该地址在每次目标突发传输完成后更新，因此：

– 编程设定的目标突发；固定寻址模式或连续数据增量式寻址模式。在连续增量式寻址模式下 (GPDMA_CxTR1.DINC = 1)，附加地址偏移值是编程设定的突发大小（由 GPDMA_CxTR1.DBL_1[5:0] 和 GPDMA_CxTR1.DDW_LOG2[1:0] 定义）

– 附加的目标递增/递减偏移值由 GPDMA_CxBR1.DDEC 和 GPDMA_CxTR3.DAO[12:0] 编程设定。

– 目标块传输完成后，对于具有 2D 寻址能力的通道 x（x = 6 到 7），附加的块重复目标递增/递减偏移值由 GPDMA_CxBR1.BRDDEC 和 GPDMA_CxBR2.BRDAO[15:0] 编程设定

在链表模式下，在 LLI 数据传输完成后，如果在 GPDMA_CxLLR.UDA = 1 时设置 LLI，则该寄存器会由 GPDMA 从存储器自动更新。

注意：目标地址必须与编程设定的目标突发传输数据宽度对齐（DA[2:0] 与 GPDMA_CxTR1.DDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

15.8.13 GPDMA 通道 x 传输寄存器 3 (GPDMA_CxTR3)

GPDMA channel x transfer register 3

偏移地址：0xA4 + 0x80 * x（x = 6 到 7）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于控制通道 x 的传输。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN = 1 时为只读。

该寄存器的写操作必须在通道传输完成后执行（然后，硬件将 GPDMA_CxCR.EN 清零）。通道传输可以在不同的级别完成和编程：块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间，如果 GPDMA_CxLLR.UT3 = 1，则该寄存器将由 GPDMA 从存储器自动更新。

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

位 31:29 保留，必须保持复位值。

位 28:16 **DAO[12:0]**：目标地址偏移增量 (destination address offset increment)

GPDMA_CxDAR 指向的目标地址在每个编程设定的目标突发传输后按此偏移量 DAO[12:0] 递增或递减（具体取决于 GPDMA_CxBR1.DDEC）。通过增量式模式 (GPDMA_CxTR1.DINC = 1) 寻址完成的突发传输时，该偏移量将加到编程设定的突发大小中，而不是目标地址中。

注意：目标地址偏移必须与编程设定的目标突发传输数据宽度对齐（DAO[2:0] 与 GPDMA_CxTR1.DDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

位 15:13 保留，必须保持复位值。

位 12:0 **SAO[12:0]**：源地址偏移增量 (source address offset increment)

GPDMA_CxSAR 指向的源地址在每个编程设定的源突发传输后按此偏移量 SAO[12:0] 递增或递减（具体取决于 GPDMA_CxBR1.SDEC）。通过增量式模式 (GPDMA_CxTR1.SINC = 1) 寻址完成的突发传输时，该偏移量将加到编程设定的突发大小中，而不是目标地址中。

注意：源地址偏移必须与编程设定的源突发传输数据宽度对齐（SAO[2:0] 与 GPDMA_CxTR1.SDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

当源块大小不是目标突发大小的倍数，而是源数据宽度的倍数时，最后一个编程设定的源突发传输不会完成，并且其大小会在内部缩小以匹配块大小。在这种情况下，不应用附加的 GPDMA_CxTR3.SAO[12:0]。

15.8.14 GPDMA 通道 x 块寄存器 2 (GPDMA_CxBR2)

GPDMA channel x block register 2

偏移地址：0xA8 + 0x80 * x（x = 6 到 7）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于控制通道 x 在 2D/ 重复块级别的传输。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN = 1 时为只读。

该寄存器的写操作必须在通道传输完成后执行（然后，硬件将 GPDMA_CxCR.EN 清零）。通道传输可以在不同的级别完成和编程：块级别、2D/ 重复块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间，如果 GPDMA_CxLLR.UB2 = 1，则该寄存器将由 GPDMA 从存储器自动更新。

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

位 31:16 **BRDAO[15:0]**：块重复目标地址偏移 (Block repeated destination address offset)

对于具有 2D 寻址能力的通道，该位域用于在块传输结束后更新（通过加法或减法，具体取决于 GPDMA_CxBR1.BRDDEC）当前目标地址 (GPDMA_CxDAR)。

注意：块重复目标地址偏移必须与编程设定的目标突发传输数据宽度对齐（BRDAO[2:0] 与 GPDMA_CxTR1.DDW_LOG2[1:0]）。否则，将报告用户设置错误且不会发出任何传输信号。

在外设流控制模式下 (GPDMA_CxTR2.PFREQ = 1)，BRDAO[15:0] 必须设为 0。

位 15:0 **BRSAO[15:0]**：块重复源地址偏移 (Block repeated source address offset)

对于具有 2D 寻址能力的通道，该位域用于在块传输结束后更新（通过加法或减法，具体取决于 GPDMA_CxBR1.BRSDEC）当前源地址 (GPDMA_CxSAR)。

注意：块重复源地址偏移必须与编程设定的源突发传输数据宽度对齐（BRSAO[2:0] 与 GPDMA_CxTR1.SDW_LOG2[1:0]）。

否则，将报告用户设置错误且不会发出任何传输信号。

在外设流控制模式下 (GPDMA_CxTR2.PFREQ = 1)，BRSAO[15:0] 必须设为 0。

15.8.15 GPDMA 通道 x 链表地址寄存器 (GPDMA_CxLLR)

GPDMA channel x linked-list address register

偏移地址：0xCC + 0x80 * x（x = 0 到 5）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于配置存储器中下一个 LLI 的数据结构及其地址指针。该寄存器为空时，通道传输完成。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN = 1 时为只读。

该寄存器的写操作必须在通道传输完成后执行（然后，硬件将 GPDMA_CxCR.EN 清零）。通道传输可以在不同的级别完成和编程：块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间，如果 GPDMA_CxLLR.ULL = 1，则该寄存器将由 GPDMA 从存储器自动更新。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UT1	UT2	UB1	USA	UDA	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ULL
rw	rw	rw	rw	rw											rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LA[15:2]														Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		

位 31 **UT1**：从存储器更新 GPDMA_CxTR1 (Update GPDMA_CxTR1 from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxTR1。
0：不更新 GPDMA_CxTR1
1：更新 GPDMA_CxTR1

位 30 **UT2**：从存储器更新 GPDMA_CxTR2 (Update GPDMA_CxTR2 from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxTR2。
0：不更新 GPDMA_CxTR2
1：更新 GPDMA_CxTR2

位 29 **UB1**：从存储器更新 GPDMA_CxBR1 (Update GPDMA_CxBR1 from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxBR1。
如果 UB1 = 0 且 GPDMA_CxLLR ≠ 0，则链表未完成。随后，GPDMA_CxBR1.BNDT[15:0] 将在数据传输完成后和链路传输前恢复为编程值。
0：不从存储器更新 GPDMA_CxBR1（如果存在任何链路传输，则恢复 GPDMA_CxBR1.BNDT[15:0]）
1：更新 GPDMA_CxBR1

位 28 **USA**：从存储器更新 GPDMA_CxSAR (update GPDMA_CxSAR from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxSAR。
0：不更新 GPDMA_CxSAR
1：更新 GPDMA_CxSAR

位 27 **UDA**：从存储器更新 GPDMA_CxDAR 寄存器 (Update GPDMA_CxDAR register from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxDAR。
0：不更新 GPDMA_CxDAR
1：更新 GPDMA_CxDAR

位 26:17 保留，必须保持复位值。

位 16 **ULL**：从存储器更新 GPDMA_CxLLR 寄存器 (Update GPDMA_CxLLR register from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxLLR。
0：不更新 GPDMA_CxLLR
1：更新 GPDMA_CxLLR

位 15:2 **LA[15:2]**：指向下一个链表数据结构的指针（16 位低有效地址）(pointer (16-bit low-significant address) to the next linked-list data structure)
如果 UT1 = UT2 = UB1 = USA = UDA = ULL = 0 并且 LA[15:20] = 0，则当前 LLI 为最后一个 LLI。通道传输完成时，不更新链表 GPDMA 寄存器文件。
否则，该位域是指向存储器偏移地址的指针，一旦数据传输完成，为了有条件地更新链表 GPDMA 内部寄存器文件（GPDMA_CxTR1、GPDMA_CxTR2、GPDMA_CxBR1、GPDMA_CxSAR、GPDMA_CxDAR 和 GPDMA_CxLLR），将自动从该存储器偏移地址中获取下一个链表数据结构。
注意：用户必须将该指针编程为 32 位对齐。将忽略两个低有效位的写入。

位 1:0 保留，必须保持复位值。

15.8.16 GPDMA 通道 x 复用链表地址寄存器 (GPDMA_CxLLR)

GPDMA channel x alternate linked-list address register

偏移地址：0xCC + 0x80 * x（x = 6 到 7）

复位值：0x0000 0000

该寄存器可以是特权寄存器或非特权寄存器，具体取决于通道 x 的特权状态 (GPDMA_PRIVCFGR.PRIVx)。

该寄存器用于配置存储器中下一个 LLI 的数据结构及其地址指针。该寄存器为空时，通道传输完成。

该寄存器的写操作必须在 GPDMA_CxCR.EN = 0 时执行。

该寄存器在 GPDMA_CxCR.EN = 1 时为只读。

该寄存器的写操作必须在通道传输完成后执行（然后，硬件将 GPDMA_CxCR.EN 清零）。通道传输可以在不同的级别完成和编程：块级别、LLI 级别或完整链表级别。

在链表模式下，链路传输期间，如果 GPDMA_CxLLR.ULL = 1，则该寄存器将由 GPDMA 从存储器自动更新。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UT1	UT2	UB1	USA	UDA	UT3	UB2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ULL
rw	rw	rw	rw	rw	rw	rw									rw

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

位 31 **UT1**：从存储器更新 GPDMA_CxTR1 (Update GPDMA_CxTR1 from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxTR1。
0：不更新 GPDMA_CxTR1
1：更新 GPDMA_CxTR1

位 30 **UT2**：从存储器更新 GPDMA_CxTR2 (Update GPDMA_CxTR2 from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxTR2。
0：不更新 GPDMA_CxTR2
1：更新 GPDMA_CxTR2

位 29 **UB1**：从存储器更新 GPDMA_CxBR1 (Update GPDMA_CxBR1 from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxBR1。
如果 UB1 = 0 且 GPDMA_CxLLR ≠ 0，则链表未完成。随后，GPDMA_CxBR1.BNDT[15:0] 将在数据传输完成后和链路传输前恢复为编程值。
0：不从存储器更新 GPDMA_CxBR1（如果存在任何链路传输，则恢复 GPDMA_CxBR1.BNDT[15:0]）
1：更新 GPDMA_CxBR1

位 28 **USA**：从存储器更新 GPDMA_CxSAR (update GPDMA_CxSAR from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxSAR。
0：不更新 GPDMA_CxSAR
1：更新 GPDMA_CxSAR

位 27 **UDA**：从存储器更新 GPDMA_CxDAR 寄存器 (Update GPDMA_CxDAR register from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxDAR。
0：不更新 GPDMA_CxDAR
1：更新 GPDMA_CxDAR

位 26 **UT3**：从存储器更新 GPDMA_CxTR3 (Update GPDMA_CxTR3 from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxTR3。
0：不更新 GPDMA_CxTR3
1：更新 GPDMA_CxTR3

位 25 **UB2**：从存储器更新 GPDMA_CxBR2 (Update GPDMA_CxBR2 from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxBR2。
0：不更新 GPDMA_CxBR2
1：更新 GPDMA_CxBR2

位 24:17 保留，必须保持复位值。

位 16 **ULL**：从存储器更新 GPDMA_CxLLR 寄存器 (Update GPDMA_CxLLR register from memory)
该位用于控制链路传输期间是否从存储器更新 GPDMA_CxLLR。
0：不更新 GPDMA_CxLLR
1：更新 GPDMA_CxLLR

位 15:2 **LA[15:2]**：指向下一个链表数据结构的指针（16 位低有效地址）(pointer (16-bit low-significant address) to the next linked-list data structure)
如果 UT1 = UT2 = UB1 = USA = UDA = ULL = 0 并且 LA[15:20] = 0，则当前 LLI 为最后一个 LLI。通道传输完成时，不更新链表 GPDMA 寄存器文件。
否则，该位域是指向存储器偏移地址的指针，一旦数据传输完成，为了有条件地更新链表 GPDMA 内部寄存器文件（GPDMA_CxTR1、GPDMA_CxTR2、GPDMA_CxBR1、GPDMA_CxSAR、GPDMA_CxDAR 和 GPDMA_CxLLR），将自动从该存储器偏移地址中获取下一个链表数据结构。
注意：用户必须将该指针编程为 32 位对齐。将忽略两个低有效位的写入。

位 1:0 保留，必须保持复位值。

15.8.17 GPDMA 寄存器映射

表 95. GPDMA 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x04	GPDMA_ PRIVCFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIV7	PRIV6	PRIV5	PRIV4	PRIV3	PRIV2	PRIV1	PRIV0
	Reset value																									0	0	0	0	0	0	0	0
0x08	Reserved	Reserved																															
0x0C	GPDMA_MISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MIS7	MIS6	MIS5	MIS4	MIS3	MIS2	MIS1	MIS0
	Reset value																									0	0	0	0	0	0	0	0
0x14 - 0x4C	Reserved	Reserved																															
0x50+0x80 * x (x=0 to 7)	GPDMA_ CxLBAR	LBA[31:16]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																
0x54 to 0x58+0x80*x (x=0 to 7)	Reserved	Reserved																															
0x5C+0x80 * x (x=0 to 7)	GPDMA_CxFCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TOF	SUSPF	USEF	ULEF	DTEF	HTF	TCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																		0	0	0	0	0	0	0								
0x60+ 0x80 * x (x=0 to 7)	GPDMA_CxSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFOL[7:0]								Res.	TOF	SUSPF	USEF	ULEF	DTEF	HTF	TCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDLEF
	Reset value									0	0	0	0	0	0	0	0		0	0	0	0	0	0	0								1
0x64+ 0x80 * x (x=0 to 7)	GPDMA_CxCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIO[1:0]		Res.	Res.	Res.	Res.	LAP	LSM	Res.	TOIE	SUSPIE	USEIE	ULEIE	DTEIE	HTIE	TCIE	Res.	Res.	Res.	Res.	Res.	SUSP	RESET	EN
	Reset value									0	0					0	0		0	0	0	0	0	0	0						0	0	0
0x68 to 0x8C+0x80 * x (x=0 to 7)	Reserved	Reserved																															
0x90+0x80 * x (x=0 to 7)	GPDMA_CxTR1	Res.	DAP	Res.	Res.	DHX	DBX	DBL_1[5:0]						DINC	Res.	DDW_LOG2 [1:0]		Res.	SAP	SBX	PAM[1:0]		Res.	SBL_1[5:0]						SINC	Res.	SDW_LOG2 [1:0]	
	Reset value		0			0	0	0	0	0	0	0	0	0		0	0		0	0	0	0		0	0	0	0	0	0	0		0	0
0x94+0x80 * x (x=0 to 7)	GPDMA_CxTR2	TCEM[1:0]		Res.	Res.	Res.	Res.	TRIGPOL[1:0]		Res.	Res.	TRIGSEL[5:0]						TRIGM[1:0]		Res.	PFREQ	BREQ	DREQ	SWREQ	Res.	REQSEL[7:0]							
	Reset value	0	0					0	0			0	0	0	0	0	0	0	0		0	0	0	0		0	0		0	0	0	0	0
0x98+0x80 * x (x=0 to 5)	GPDMA_CxBR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BNDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x98+0x80 * x (x=6 to 7)	GPDMA_CxBR1	BRDDEC	BRSDEC	DDEC	SDEC	Res.	BRC[10:0]											BNDT[15:0]															
	Reset value	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x9C+0x80 * x (x=0 to 7)	GPDMA_CxSAR	SA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xA0+0x80 * x (x=0 to 7)	GPDMA_CxDAR	DA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xA4+0x80 * x (x=6 to 7)	GPDMA_CxTR3	Res.	Res.	Res.	DAO[12:0]													Res.	Res.	Res.	SAO[12:0]												
	Reset value				0	0	0	0	0	0	0	0	0	0	0	0	0				0	0	0	0	0	0	0	0	0	0	0	0	0

表 95. GPDMA 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xA8+0x80 * x (x=6 to 7)	GPDMA_CxBR2	BRDAO[15:0]																BRSAO[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xCC+0x80 * x (x=0 to 5)	GPDMA_CxLLR	UT1	UT2	UB1	USA	UDA	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ULL	LA[15:2]														Res.	Res.
	Reset value	0	0	0	0	0											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
0xCC+0x80 * x (x=6 to 7)	GPDMA_CxLLR	UT1	UT2	UB1	USA	UDA	UT3	UB2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ULL	LA[15:2]														Res.	Res.
	Reset value	0	0	0	0	0	0	0									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

有关寄存器边界地址的信息，请参见*第 2.2 节*。

16 嵌套向量中断控制器 (NVIC)

16.1 NVIC 主要特性

- 134 个可屏蔽中断通道（不包括带 FPU 的 Cortex-M33 的 16 根中断线）
- 16 个可编程优先级（使用了 4 位中断优先级）
- 低延迟异常和中断处理
- 电源管理控制
- 系统控制寄存器的实现

NVIC 和处理器内核接口紧密配合，可以实现低延迟的中断处理和晚到中断的高效处理。

包括内核异常在内的所有中断均通过 NVIC 进行管理。

16.2 SysTick 校准值寄存器

Cortex-M33 内置一个 SysTick 定时器。SysTick 定时器的校准值 (STCALIB) 为 0x3E8，会基于 1 MHz 的 SysTick 时钟频率，产生 1 ms 时间基准。对于以给定频率运行的应用程序，为匹配 1 ms 时基，必须在 SYST_RVR 寄存器中按如下方式对 SysTick 重载值进行编程：

- SysTick 时钟源为 CPU 时钟 HCLK 时
  重载值 = (HCLK x STCALIB) - 1
- SysTick 时钟源为外部时钟 (HCLK/8) 时
  重载值 = ((HCLK/8) x STCALIB) - 1

HCLK 是指以 MHz 为单位的 AHB 频率值。

示例：SysTick 时钟源为 100 MHz 的 CPU 时钟 HCLK，以匹配 1 ms 的时基：

SysTick 重载值 = (100 x STCALIB) - 1 = 0x1869F

16.3 中断和异常向量

下表中的灰色行表示无特定位置的向量。

表 96. STM32H503xx 向量表

位置	优先级	优先级类型	缩略语	说明	地址
-	-	-	-	保留	0x0000 0000
-	-4	固定	复位	复位	0x0000 0004
-	-2	固定	NMI	不可屏蔽中断。RCC 时钟安全系统 (CSS) 连接到 NMI 向量。	0x0000 0008
-	-1	固定	HardFault	硬性故障。所有类型的错误	0x0000 000C
-	0	可设置	MemManage	存储器管理	0x0000 0010
-	1	可设置	BusFault	预取指失败，存储器访问失败	0x0000 0014
-	2	可设置	UsageFault	未定义的指令或非法状态	0x0000 0018
-	-	-	-	保留	0x0000 001C - 0x0000 0028
-	4	-	SVC	通过 SWI 指令调用的系统服务	0x0000 002C
-	5	-	调试监控器	监控	0x0000 0030
-	-	-	-	保留	0x0000 0034
-	6	可设置	PendSV	可挂起的系统服务请求	0x0000 0038
-	7	可设置	SysTick	系统节拍定时器	0x0000 003C
0	8	可设置	WWDG	窗口看门狗中断	0x0000 0040
1	9	可设置	PVD_AVD	电源电压监测器/模拟电压监测器	0x0000 0044
2	10	可设置	RTC	RTC 全局中断	0x0000 0048
-	-	-	-	保留	0x0000 004C
4	12	可设置	TAMP	入侵全局中断	0x0000 0050
5	13	可设置	RAMCFG	RAM 配置全局中断	0x0000 0054
6	14	可设置	FLASH	Flash 全局中断	0x0000 0058
-	-	-	-	保留	0x0000 005C
-	-	-	-	保留	0x0000 0060
9	17	可设置	RCC	RCC 全局中断	0x0000 0064
-	-	-	-	保留	0x0000 0068
11	19	可设置	EXTI0	EXTI 线 0 中断	0x0000 006C
12	20	可设置	EXTI1	EXTI 线 1 中断	0x0000 0070
13	21	可设置	EXTI2	EXTI 线 2 中断	0x0000 0074
14	22	可设置	EXTI3	EXTI 线 3 中断	0x0000 0078
15	23	可设置	EXTI4	EXTI 线 4 中断	0x0000 007C

表 96. STM32H503xx 向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
16	24	可设置	EXTI5	EXTI 线 5 中断	0x0000 0080
17	25	可设置	EXTI6	EXTI 线 6 中断	0x0000 0084
18	26	可设置	EXTI7	EXTI 线 7 中断	0x0000 0088
19	27	可设置	EXTI8	EXTI 线 8 中断	0x0000 008C
20	28	可设置	EXTI9	EXTI 线 9 中断	0x0000 0090
21	29	可设置	EXTI10	EXTI 线 10 中断	0x0000 0094
22	30	可设置	EXTI11	EXTI 线 11 中断	0x0000 0098
23	31	可设置	EXTI12	EXTI 线 12 中断	0x0000 009C
24	32	可设置	EXTI13	EXTI 线 13 中断	0x0000 00E4
25	33	可设置	EXTI14	EXTI 线 14 中断	0x0000 00A0
26	34	可设置	EXTI15	EXTI 线 15 中断	0x0000 00A4
27	35	可设置	GPDMA1_CH0	GPDMA1 通道 0 全局中断	0x0000 00A8
28	36	可设置	GPDMA1_CH1	GPDMA1 通道 1 全局中断	0x0000 00AC
29	37	可设置	GPDMA1_CH2	GPDMA1 通道 2 全局中断	0x0000 00B0
30	38	可设置	GPDMA1_CH3	GPDMA1 通道 3 全局中断	0x0000 00B4
31	39	可设置	GPDMA1_CH4	GPDMA1 通道 4 全局中断	0x0000 00B8
32	40	可设置	GPDMA1_CH5	GPDMA1 通道 5 全局中断	0x0000 00C0
33	41	可设置	GPDMA1_CH6	GPDMA1 通道 6 全局中断	0x0000 00C4
34	42	可设置	GPDMA1_CH7	GPDMA1 通道 7 全局中断	0x0000 00C8
35	43	可设置	IWDG	独立看门狗中断	0x0000 00CC
-	-	-	-	保留	0x0000 00D0
37	45	可设置	ADC1	ADC1 全局中断	0x0000 00D4
38	46	可设置	DAC1	DAC1 全局中断	0x0000 00D8
39	47	可设置	FDCAN1_IT0	FDCAN1 中断 0	0x0000 00DC
40	48	可设置	FDCAN1_IT1	FDCAN1 中断 1	0x0000 00E0
41	49	可设置	TIM1_BRK/TIM1_TERR/ TIM1_IERR	TIM1 刹车/TIM1 转换错误/TIM1 索引错误	0x0000 00E4
42	50	可设置	TIM1_UP	TIM1 更新	0x0000 00E8
43	51	可设置	TIM1_TRG_COM/TIM1_ DIR/TIM1_IDX	TIM1 触发和换向/TIM1 方向改变中断/ TIM1 索引	0x0000 00EC
44	52	可设置	TIM1_CC	TIM1 捕获/比较中断	0x0000 00F0

表 96. STM32H503xx 向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
45	53	可设置	TIM2	TIM2 全局中断	0x0000 00F4
46	54	可设置	TIM3	TIM3 全局中断	0x0000 00F8
-	-	-	-	保留	0x0000 00FC - 0x0000 0100
49	57	可设置	TIM6	TIM6 全局中断	0x0000 0104
50	58	可设置	TIM7	TIM7 全局中断	0x0000 0108
51	59	可设置	I2C1_EV	I2C1 事件中断	0x0000 010C
52	60	可设置	I2C1_ER	I2C1 错误中断	0x0000 0110
53	61	可设置	I2C2_EV	I2C2 事件中断	0x0000 0114
54	62	可设置	I2C2_ER	I2C2 错误中断	0x0000 0118
55	63	可设置	SPI1	SPI1 全局中断	0x0000 011C
56	64	可设置	SPI2	SPI2 全局中断	0x0000 0120
57	65	可设置	SPI3	SPI3 全局中断	0x0000 0124
58	66	可设置	USART1	USART1 全局中断	0x0000 0128
59	67	可设置	USART2	USART2 全局中断	0x0000 012C
60	68	可设置	USART3	USART3 全局中断	0x0000 0130
-	-	-	-	保留	0x0000 0134 - 0x0000 0138
63	71	可设置	LPUART1	LPUART1 全局中断或 LPUART1 R 唤醒或 LPUART1 T 唤醒（通过 EXTI 线）	0x0000 013C
64	72	可设置	LPTIM1 或 LPTIM1_AIT	LPTIM1 全局中断或 LPTimer1 AIT（通过 EXTI 线）	0x0000 0140
-	-	-	-	保留	0x0000 0144 - 0x0000 0154
70	78	可设置	LPTIM2 或 LPTIM2_AIT	LPTIM2 全局中断或 LPTimer2 AIT（通过 EXTI 线）	0x0000 0158
-	-	-	-	保留	0x0000 015C - 0x0000 0164
74	82	可设置	USB	USB 全局中断	0x0000 0168
75	83	可设置	CRS	时钟恢复系统全局中断	0x0000 016C
-	-	-	-	保留	0x0000 0170 - 0x0000 018C
90	98	可设置	GPDMA2_CH0	GPDMA2 通道 0 全局中断	0x0000 0190

表 96. STM32H503xx 向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
91	99	可设置	GPDMA2_CH1	GPDMA2 通道 1 全局中断	0x0000 0194
92	100	可设置	GPDMA2_CH2	GPDMA2 通道 2 全局中断	0x0000 0198
93	101	可设置	GPDMA2_CH3	GPDMA2 通道 3 全局中断	0x0000 019C
94	102	可设置	GPDMA2_CH4	GPDMA2 通道 4 全局中断	0x0000 01A0
95	103	可设置	GPDMA2_CH5	GPDMA2 通道 5 全局中断	0x0000 01A4
96	104	可设置	GPDMA2_CH6	GPDMA2 通道 6 全局中断	0x0000 01A8
97	105	可设置	GPDMA2_CH7	GPDMA2 通道 7 全局中断	0x0000 01AC
-	-	-	-	保留	0x0000 01B0 - 0x0000 01C0
103	111	可设置	FPU	浮点中断	0x0000 01C4
104	112	可设置	ICACHE	指令缓存全局中断	0x0000 01C8
-	-	-	-	保留	0x0000 01CC - 0x0000 01E8
113	121	可设置	DTS 或 DTS_WKUP	DTS 中断或 DTS AIT（通过 EXTI 线）	0x0000 01EC
114	122	可设置	RNG	RNG 全局中断	0x0000 01F0
-	-	-	-	保留	0x0000 01F4 - 0x0000 01F8
117	125	可设置	HASH	HASH 中断	0x0000 01FC
-	-	-	-	保留	0x0000 0200 - 0x0000 0210
123	131	可设置	I3C1_EV	I3C1 事件中断	0x0000 0214
124	132	可设置	I3C1_ER	I3C1 错误中断	0x0000 0218
-	-	-	-	保留	0x0000 021C - 0x0000 0230
131	139	可设置	I3C2_EV	I3C2 事件中断	0x0000 0234
132	140	可设置	I3C2_ER	I3C2 错误中断	0x0000 0238
133	141	可设置	COMP	COMP 全局中断	0x0000 023C

17 扩展中断和事件控制器 (EXTI)

扩展中断和事件控制器 (EXTI) 通过可配置的事件输入来管理单独的 CPU 和系统唤醒。它可以为电源控制提供唤醒请求，给 CPU NVIC 产生中断请求以及给 CPU 的事件输入产生事件。对于 CPU，需通过额外的事件生成模块 (EVG) 来生成 CPU 事件信号。

EXTI 唤醒请求可以唤醒处于停止模式的系统。

此外，还可以在运行模式下生成中断请求和事件请求。

EXTI 还包含 EXTI 复用器 IO 端口选择。

17.1 EXTI 主要特性

EXTI 的主要特性如下：

- 支持 54 个输入事件
- 所有事件输入都可以唤醒系统。
- 对于在外设中没有相关唤醒标志的事件，均在 EXTI 中有一个标志，并且会从 EXTI 生成一个 CPU 中断。
- 事件可用于生成 CPU 唤醒事件。

可配置事件具有以下特性：

- 可选择的有效触发边沿
- 独立于上升沿和下降沿的中断挂起状态寄存器位
- 单独的中断和事件生成掩码，用于调节 CPU 唤醒、中断和事件生成
- 支持软件触发
- EXTI IO 端口选择

17.2 EXTI 框图

EXTI 包含一个通过 AHB 接口访问的寄存器模块、一个事件输入触发模块、一个屏蔽模块和一个 EXTI 复用器，如*图 71* 所示。

寄存器模块包含所有 EXTI 寄存器。

事件输入触发模块提供事件输入边沿触发逻辑。

屏蔽模块为不同的唤醒、中断和事件输出及其屏蔽功能提供事件输入分配。

EXTI 复用器为 EXTI 事件信号选择 IO 端口。

图 71. EXTI 框图

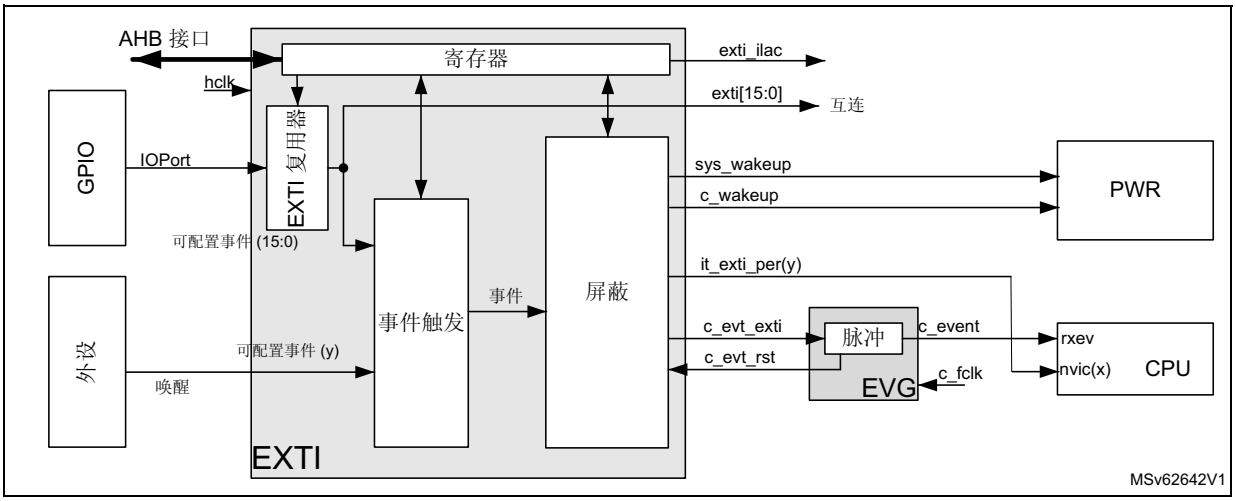


表 97. EXTI 引脚概述

引脚名称	I/O	说明
AHB 接口	I/O	EXTI 寄存器总线接口。
hclk	I	AHB 总线时钟和 EXTI 系统时钟
可配置事件 (y)	I	来自没有相关中断和标志的外设的异步唤醒事件
exti_ilac	O	非法访问事件
IOPort(n)	I	GPIO 模块 IO 端口[15:0]
exti[15:0]	O	用于触发其他外设的 EXTI GPIO 输出端口
it_exti_per (y)	O	与可配置事件 (y) 关联的 CPU 的中断
c_evt_exti	O	CPU 的高电平触发事件输出，与 hclk 同步
c_evt_rst	I	用于清除 c_evt_exti 的异步复位输入
sys_wakeup	O	ck_sys 和 hclk 向 PWR 发出的异步系统唤醒请求
c_wakeup	O	CPU 向 PWR 发出的唤醒请求，与 hclk 同步

表 98. EVG 引脚概述

引脚名称	I/O	说明
c_fclk	I	CPU 自由运行时钟
c_evt_in	I	来自 EXTI 的高电平触发事件输入，与 CPU 时钟异步
c_event	O	事件脉冲，与 CPU 时钟同步
c_evt_rst	O	事件复位信号，与 CPU 时钟同步

17.2.1 外设和 CPU 之间的 EXTI 连接

对于能够在系统处于停止模式时生成唤醒或中断事件的一些外设，将连接至 EXTI。

- 对于生成脉冲或在外设中没有中断状态位的外设唤醒信号，将连接至 EXTI 可配置事件输入。对于这类事件，EXTI 提供的状态挂起位需清零。与状态位相关的 EXTI 中断会中断 CPU。
- EXTI 复用器的所有 GPIO 端口输入可以选择端口引脚以便通过可配置的事件来唤醒系统。

EXTI 可配置事件中断连接到 NVIC。

专用 EXTI/EVG CPU 事件连接至 CPU rxev 输入。

EXTI CPU 唤醒信号连接至 PWR，用于唤醒系统和 CPU 子系统总线时钟。

17.2.2 EXTI 中断/事件映射

EXTI 线的连接方式如下表所示。

表 99. EXTI 线连接

EXTI 线	线源	线类型
0-15	GPIO	可配置
16	PVD/AVD 输出	可配置
17	RTC	-
18	保留	-
19	TAMP	-
20	Reserved	-
21	I2C1 唤醒	-
22	I2C2 唤醒	-
23	Reserved	-
24	I3C 唤醒	-
25	USART1 唤醒	-
26	USART2 唤醒	-
27	USART3 唤醒	-
28	I3C2 唤醒	-
29	COMP1 输出唤醒	-
30 - 36	保留	-
37	LPUART1 唤醒	-
38	LPTIM1	-
39	LPTIM2	-
40	SPI1 唤醒	-
41	SPI2 唤醒	-
42	SPI3 唤醒	-
43 - 46	保留	-

表 99. EXTI 线连接（续）

EXTI 线	线源	线类型
47	USB 唤醒	-
48	保留	-
49	LPTIM2 CH1	-
50	DTS 唤醒	可配置
51 - 52	保留	-
53	V_{DDIO2} 电压监测	可配置

17.3 EXTI 功能说明

事件特性通过寄存器位进行控制，如下所述：

- 有效触发边沿使能
  - 由在 *EXTI 上升沿触发选择寄存器 (EXTI_RTSR1)* 中选择的上升沿使能
  - 由在 *EXTI 下降沿触发选择寄存器 (EXTI_FTSR1)* 中选择的下降沿使能
- 软件触发由 *EXTI 软件中断事件寄存器 (EXTI_SWIER1)* 控制
- 以下寄存器是设置中断挂起标志

  EXTI 上升沿挂起寄存器 (EXTI_RPR1)

  EXTI 下降沿挂起寄存器 (EXTI_FPR1)
- 以下寄存器是设置 CPU 唤醒和中断使能

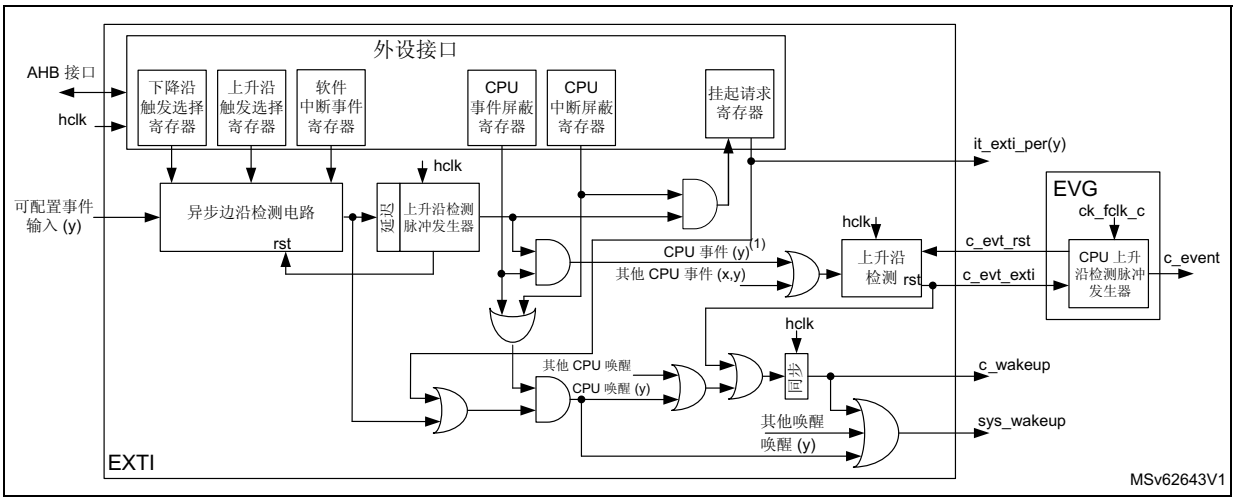
  EXTI CPU 唤醒中断屏蔽寄存器 (EXTI_IMR1)
- 以下寄存器是设置 CPU 唤醒和事件使能

  EXTI CPU 唤醒中断屏蔽寄存器 (EXTI_IMR1)

17.3.1 EXTI 可配置事件输入唤醒

下图所示为具有如下用途的可配置事件输入相关的逻辑详图：唤醒 CPU 子系统总线时钟并生成 CPU 的 EXTI 挂起标志与中断和/或 CPU 唤醒事件。

图 72. 可配置事件触发逻辑 CPU 唤醒



1. 仅适用于支持 CPU rxev 生成 c_event 的输入事件。

软件中断事件寄存器允许通过软件对相应的寄存器位执行写操作来触发可配置事件，与边沿选择设置无关。

可配置事件有效触发边沿（或两个边沿）在上升沿/下降沿选择寄存器中选择并使能。

CPU 具有专用的唤醒（中断）屏蔽寄存器和专用的事件屏蔽寄存器。事件使能后，会为 CPU 生成此事件。CPU 的所有事件会一起进行逻辑或运算，生成单个 CPU 事件信号。事件挂起寄存器（EXTI_RPR 和 EXTI_FPR）不会针对未屏蔽的 CPU 事件而置位。

可配置事件具有唯一的中断挂起请求寄存器。挂起寄存器只会针对未屏蔽的中断而置位。每个可配置事件均提供对 CPU 的中断。可配置事件中断必须由软件通过 EXTI_RPR 和/或 EXTI_FPR 寄存器确认。

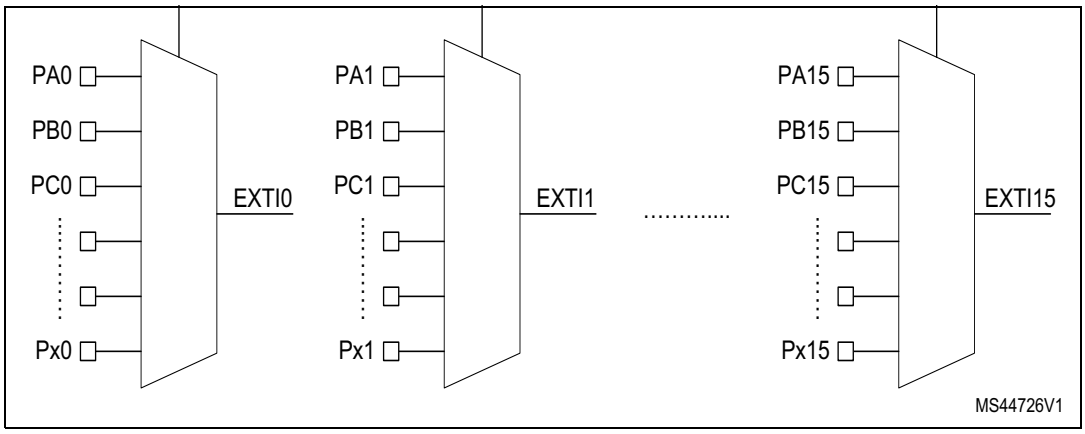
使能 CPU 唤醒（中断）或 CPU 事件时，异步边沿检测电路由时钟延迟和上升沿检测脉冲发生器复位。这可以确保 EXTI hclk 时钟在异步边沿检测电路复位之前唤醒。

注意： *检测到的可配置事件中断挂起请求可由具有适当访问权限的 CPU 清除。只要中断挂起请求处于活动状态，系统就无法进入低功耗模式。*

17.3.2 EXTI 复用器选择

EXTI 复用器可选择作为中断和唤醒的 GPIO。各 GPIO 通过 16 根 EXTI 复用器线连接到前 16 个 EXTI 事件（均为可配置事件）。选择哪个 GPIO 端口作为 EXTI 复用器输出，由 *EXTI 外部中断选择寄存器 (EXTI_EXTICR1)* 控制。

图 73. EXTI 复用器 GPIO 选择



EXTI 复用器输出可用作 EXTI 的输出信号，用于触发其他外设，与 EXTI_IMR 和 EXTI_EMR 寄存器中的屏蔽无关。

17.4 EXTI 功能行为

可配置事件通过使能至少一个触发边沿来使能。

使能事件输入后，是否生成 CPU 唤醒取决于 CPU 中断屏蔽和 CPU 事件屏蔽。

表 100. 屏蔽功能

CPU 中断使能（EXTI_IMR.IMn 中）	CPU 事件使能（EXTI_EMR.EMn 中）	可配置事件输入（EXTI_RPR.RPIFn 和 EXTI_FPR.FPIFn 中）	Exti(n) 中断(1)	CPU 事件	CPU 唤醒
0	0	否	已屏蔽	已屏蔽	已屏蔽
	1	否	已屏蔽	是	是
1	0	状态已锁存	是	已屏蔽	是(2)
	1	状态已锁存	是	是	是

1. 单个 exti(n) 中断将面向 CPU。如果 CPU(m) 不需要中断，则必须在 CPU NVIC 中屏蔽 exti(n) 中断。
2. 仅限 EXTI_IMR.IMn 中已使能 CPU 中断的情况。

对于可配置事件输入，当事件输入上出现使能的边沿时，会生成一个事件请求。相关 CPU 中断未被屏蔽时，相应的挂起位 EXTI_RPR.RPIFn 和/或 EXTI_FPR.FPIFn 会置 1：CPU 子系统被唤醒，CPU 中断信号被激活。EXTI_RPR.RPIFn 和/或 EXTI_FPR.FPIFn 挂起位必须通过软件写入 1 来清零。该操作会清除 CPU 中断。

对于可配置事件输入，可通过软件在软件中断/事件寄存器 EXTI_SWIER 中写入“1”来生成事件请求，从而基于事件生成上升沿。EXTI_RPR 中的上升沿事件挂起位置 1，与 EXTI_RTSR 中的设置无关。

17.5 EXTI 事件保护

EXTI 可以保护事件寄存器位免受非特权访问的修改。通过 EXTI_PRIVCFGR 中的寄存器位，可以针对各输入事件单独激活保护。在 EXTI 级别，保护包括防止以下未经授权的写访问：

- 更改特权可配置事件的设置。
- 更改特权输入事件的屏蔽。
- 清除特权输入事件的挂起状态。

表 101. 寄存器保护概述

寄存器名称	访问类型	保护(1)
EXTI_RTSR	RW	特权可在 EXTI_PRIVCFGR 中逐位使能。
EXTI_FTSR	RW	
EXTI_SWIER	RW	
EXTI_RPR	RW	
EXTI_FPR	RW	
EXTI_PRIVCFGR	RW	始终为特权。
EXTI_EXTICRn	RW	特权可在 EXTI_PRIVCFGR 中逐位使能。
EXTI_IMR	RW	特权可在 EXTI_PRIVCFGR 中逐位使能。
EXTI_EMR	RW	

1. 特权通过单独的输入事件（EXTI_PRIVCFGR 寄存器）使能。

17.5.1 EXTI 特权保护

为输入事件使能特权后，相关输入事件配置和控制位只能通过特权访问进行修改和读取。非特权写访问会被忽略，非特权读访问会返回 0。

输入事件为非特权事件时，特权被禁止。相关输入事件配置和控制位可通过特权访问和非特权访问进行修改和读取。

17.6 EXTI 寄存器

EXTI 寄存器映射分为以下几个部分：

表 102. EXTI 寄存器映射的各个部分

偏移地址	说明
0x000 - 0x01C	通用可配置事件 [31:0] 配置
0x020 - 0x03C	通用可配置事件 [57:32] 配置
0x060 - 0x06C	EXTI IO 端口复用器选择
0x080 - 0x0BC	CPU 输入事件配置

所有寄存器均可支持按字（32 位）、半字（16 位）或字节（8 位）访问。

17.6.1 EXTI 上升沿触发选择寄存器 (EXTI_RTSR1)

EXTI rising trigger selection register

偏移地址：0x000

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RT16
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:17 保留，必须保持复位值。

位 16:0 **RTx**：可配置事件输入 x 的上升沿触发事件配置位 (Rising trigger event configuration bit of configurable event input x)(1)（x = 16 到 0）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，RTx 可通过非特权和特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，RTx 只能通过特权访问进行访问。对该位 x 执行的非特权写访问会被忽略，非特权读访问会返回 0。

0：禁止输入线上升沿触发（事件和中断）

1：使能输入线上升沿触发（事件和中断）

1. 可配置事件输入为边沿触发，在这些输入上不能出现毛刺信号。
如果在对寄存器执行写操作时可配置事件输入上出现上升沿，则相关挂起位不会被置 1。
在同一可配置事件输入上，可同时设置上升沿和下降沿触发。在这种情况下，两种边沿均会生成触发信号。

17.6.2 EXTI 下降沿触发选择寄存器 (EXTI_FTSR1)

EXTI falling trigger selection register

偏移地址：0x004

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FT16
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:17 保留，必须保持复位值。

位 16:0 **FTx**：可配置事件输入 x 的下降沿触发事件配置位 (Falling trigger event configuration bit of configurable event input x)(1)（x = 16 到 0）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，FTx 可通过非特权和特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，FTx 只能通过特权访问进行访问。对该 FTx 执行的非特权写访问会被忽略，非特权读访问会返回 0。

0：禁止输入线下降沿触发（事件和中断）

1：使能输入线下降沿触发（事件和中断）

1. 可配置事件输入为边沿触发，在这些输入上不能出现毛刺信号。
如果在对寄存器执行写操作时可配置事件输入上出现下降沿，则相关挂起位不会被置 1。
在同一可配置事件输入上，可同时设置上升沿和下降沿触发。在这种情况下，两种边沿均会生成触发信号。

17.6.3 EXTI 软件中断事件寄存器 (EXTI_SWIER1)

EXTI software interrupt event register

偏移地址：0x008

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWI16
															rw

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

位 31:17 保留，必须保持复位值。

位 16:0 **SWIx**：事件 x 的软件中断 (Software interrupt on event x)（x = 16 到 0）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，SWIx 可通过非特权和特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，SWIx 只能通过特权访问进行访问。对该 SWIx 执行的非特权写访问会被忽略，非特权读访问会返回 0。

软件中断的生成与 EXTI_RTSR 和 EXTI_FTSR 中的设置无关。执行读操作时将始终返回 0。

0：写入 0 无影响。

1：写入 1 会触发事件 x 的上升沿事件。此位由硬件自动清零。

17.6.4 EXTI 上升沿挂起寄存器 (EXTI_RPR1)

EXTI rising edge pending register

偏移地址：0x00C

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RPIF16
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPIF15	RPIF14	RPIF13	RPIF12	RPIF11	RPIF10	RPIF9	RPIF8	RPIF7	RPIF6	RPIF5	RPIF4	RPIF3	RPIF2	RPIF1	RPIF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:17 保留，必须保持复位值。

位 16:0 **RPIFx**：可配置事件输入 x 上升沿挂起位 (configurable event inputs x rising edge pending bit)（x = 16 到 0）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，RPIFx 可通过非特权和特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，RPIFx 只能通过特权访问进行访问。对该 RPIFx 执行的非特权写访问会被忽略，非特权读访问会返回 0。

0：未发生上升沿触发请求

1：发生了上升沿触发请求

当在可配置事件线上发生了上升沿事件或 EXTI_SWIER 软件触发时，此位将置 1。该位可通过向其写入 1 的方式清零。

17.6.5 EXTI 下降沿挂起寄存器 (EXTI_FPR1)

EXTI falling edge pending register

偏移地址：0x010

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FPIF16
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FPIF15	FPIF14	FPIF13	FPIF12	FPIF11	FPIF10	FPIF9	FPIF8	FPIF7	FPIF6	FPIF5	FPIF4	FPIF3	FPIF2	FPIF1	FPIF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:17 保留，必须保持复位值。

位 16:0 **FPIFx**：可配置事件输入 x 下降沿挂起位 (configurable event inputs x falling edge pending bit)（x = 16 到 0）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，FPIFx 可通过非特权和特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，FPIFx 只能通过特权访问进行访问。对该 FPIFx 执行的非特权写访问会被忽略，非特权读访问会返回 0。

0：未发生下降沿触发请求

1：发生了下降沿触发请求

当可配置事件线上发生了下降沿事件时，此位被置 1。该位可通过向其写入 1 的方式清零。

17.6.6 EXTI 特权配置寄存器 (EXTI_PRIVCFGR1)

EXTI privilege configuration register

偏移地址：0x018

复位值：0x0000 0000

此寄存器提供特权写访问保护。非特权读访问会返回寄存器数据。

仅包含特权输入事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	PRIV29	PRIV28	PRIV27	PRIV26	PRIV25	PRIV24	Res.	PRIV22	PRIV21	Res.	PRIV19	Res.	PRIV17	PRIV16
		rw	rw	rw	rw	rw	rw		rw	rw		rw		rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRIV15	PRIV14	PRIV13	PRIV12	PRIV11	PRIV10	PRIV9	PRIV8	PRIV7	PRIV6	PRIV5	PRIV4	PRIV3	PRIV2	PRIV1	PRIV0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:30 保留，必须保持复位值。

位 29:24 **PRIVx**：事件输入 x 上的特权使能 (Privilege enable on event input x)（x = 29 到 24）
0：禁止事件特权（非特权）
1：使能事件特权（特权）

位 23 保留，必须保持复位值。

位 22:21 **PRIVx**：事件输入 x 上的特权使能 (Privilege enable on event input x)（x = 22 到 21）
0：禁止事件特权（非特权）
1：使能事件特权（特权）

位 20 保留，必须保持复位值。

位 19 **PRIV19**：事件输入 19 上的特权使能 (Privilege enable on event input 19)
0：禁止事件特权（非特权）
1：使能事件特权（特权）

位 18 保留，必须保持复位值。

位 17:0 **PRIVx**：事件输入 x 上的特权使能 (Privilege enable on event input x)（x = 17 到 0）
0：禁止事件特权（非特权）
1：使能事件特权（特权）

17.6.7 EXTI 上升沿触发选择寄存器 2 (EXTI_RTSR2)

EXTI rising trigger selection register 2

偏移地址：0x020

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RT53	Res.	Res.	RT50	Res.	Res.
										rw			rw		

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

位 31:22、20:19、17:0 保留，必须保持复位值。

位 21 **RT53**：可配置事件输入 x 的上升沿触发事件配置位 (Rising trigger event configuration bit of configurable event input x)(1)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，RTx 可通过非特权和特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，RTx 只能通过特权访问进行访问。对该位 x 执行的非特权写访问会被忽略，非特权读访问会返回 0。
0：禁止输入线上升沿触发（事件和中断）
1：使能输入线上升沿触发（事件和中断）

位 18 **RT50**：可配置事件输入 x(1) 的上升沿触发事件配置位 (Rising trigger event configuration bit of configurable event input x)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，RTx 可通过非特权和特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，RTx 只能通过特权访问进行访问。对该位 x 执行的非特权写访问会被忽略，非特权读访问会返回 0。
0：禁止输入线上升沿触发（事件和中断）
1：使能输入线上升沿触发（事件和中断）

1. 可配置事件输入为边沿触发，在这些输入上不能出现毛刺信号。
如果在对寄存器执行写操作时可配置事件输入上出现上升沿，则相关挂起位不会被置 1。
在同一可配置事件输入上，可同时设置上升沿和下降沿触发。在这种情况下，两种边沿均会生成触发信号。

17.6.8 EXTI 下降沿触发选择寄存器 2 (EXTI_FTSR2)

EXTI falling trigger selection register 2

偏移地址：0x024

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FT53	Res.	Res.	FT50	Res.	Res.
										rw			rw		

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

位 31:22、20:19、17:0 保留，必须保持复位值。

位 21 **FT53**：可配置事件输入 x 的下降沿触发事件配置位 (Falling trigger event configuration bit of configurable event input x) (1)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，FTx 可通过非特权和特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，FTx 只能通过特权访问进行访问。对该 FTx 执行的非特权写访问会被忽略，非特权读访问会返回 0。
0：禁止输入线下降沿触发（事件和中断）
1：使能输入线下降沿触发（事件和中断）

位 18 **FT50**：可配置事件输入 x(1) 的下降沿触发事件配置位 (Falling trigger event configuration bit of configurable event input x)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，FTx 可通过非特权和特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，FTx 只能通过特权访问进行访问。对该 FTx 执行的非特权写访问会被忽略，非特权读访问会返回 0。
0：禁止输入线下降沿触发（事件和中断）
1：使能输入线下降沿触发（事件和中断）

1. 可配置事件输入为边沿触发，在这些输入上不能出现毛刺信号。
如果在对寄存器执行写操作时可配置事件输入上出现下降沿，则相关挂起位不会被置 1。
在同一可配置事件输入上，可同时设置上升沿和下降沿触发。在这种情况下，两种边沿均会生成触发信号。

17.6.9 EXTI 软件中断事件寄存器 2 (EXTI_SWIER2)

EXTI software interrupt event register 2

偏移地址：0x028

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWI53	Res.	Res.	SWI50	Res.	Res.
										rw			rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

位 31:22、20:19、17:0 保留，必须保持复位值。

位 21 **SWI53**：事件 x 的软件中断 (Software interrupt on event x)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，SWIx 可通过非特权和特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，SWIx 只能通过特权访问进行访问。对该 SWIx 执行的非特权写访问会被忽略，非特权读访问会返回 0。
软件中断的生成与 EXTI_RTSR 和 EXTI_FTSR 中的设置无关。执行读操作时将始终返回 0。
0：写入 0 无影响。
1：写入 1 会触发事件 x 的上升沿事件。此位由硬件自动清零。

位 18 **SWI50**：事件 x 的软件中断 (Software interrupt on event x)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，SWIx 可通过非特权和特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，SWIx 只能通过特权访问进行访问。对该 SWIx 执行的非特权写访问会被忽略，非特权读访问会返回 0。

软件中断的生成与 EXTI_RTSR 和 EXTI_FTSR 中的设置无关。执行读操作时将始终返回 0。

0：写入 0 无影响。

1：写入 1 会触发事件 x 的上升沿事件。此位由硬件自动清零。

17.6.10 EXTI 上升沿挂起寄存器 2 (EXTI_RPR2)

EXTI rising edge pending register 2

偏移地址：0x02C

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RPIF53	Res.	Res.	RPIF50	Res.	Res.
										rw			rw		

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

位 31:22、20:19、17:0 保留，必须保持复位值。

位 21 **RPIF53**：可配置事件输入 x 上升沿挂起位

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，RPIFx 可通过非特权和特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，RPIFx 只能通过特权访问进行访问。对该 RPIFx 执行的非特权写访问会被忽略，非特权读访问会返回 0。

0：未发生上升沿触发请求

1：发生了上升沿触发请求

当在可配置事件线上发生了上升沿事件或 EXTI_SWIER 软件触发时，此位将置 1。该位可通过向其写入 1 的方式清零。

位 18 **RPIF50**：可配置事件输入 x 上升沿挂起位 (configurable event inputs x rising edge pending bit)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，RPIFx 可通过非特权和特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，RPIFx 只能通过特权访问进行访问。对该 RPIFx 执行的非特权写访问会被忽略，非特权读访问会返回 0。

0：未发生上升沿触发请求

1：发生了上升沿触发请求

当在可配置事件线上发生了上升沿事件或 EXTI_SWIER 软件触发时，此位将置 1。该位可通过向其写入 1 的方式清零。

17.6.11 EXTI 下降沿挂起寄存器 2 (EXTI_FPR2)

EXTI falling edge pending register 2

偏移地址：0x030

复位值：0x0000 0000

仅包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FPIF53	Res.	Res.	FPIF50	Res.	Res.
										rw			rw		

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

位 31:22、20:19、17:0 保留，必须保持复位值。

位 21 **FPIF53**：可配置事件输入 x 下降沿挂起位 (configurable event inputs x falling edge pending bit)
EXTI_PRIVCFGR.PRIVx 处于禁止状态时，FPIFx 可通过非特权和特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，FPIFx 只能通过特权访问进行访问。对该 FPIFx 执行的非特权写访问会被忽略，非特权读访问会返回 0。
0：未发生下降沿触发请求
1：发生了下降沿触发请求
当可配置事件线上发生了下降沿事件时，此位被置 1。该位可通过向其写入 1 的方式清零。

位 18 **FPIF50**：可配置事件输入 x 下降沿挂起位 (configurable event inputs x falling edge pending bit)
EXTI_PRIVCFGR.PRIVx 处于禁止状态时，FPIFx 可通过非特权和特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，FPIFx 只能通过特权访问进行访问。对该 FPIFx 执行的非特权写访问会被忽略，非特权读访问会返回 0。
0：未发生下降沿触发请求
1：发生了下降沿触发请求
当可配置事件线上发生了下降沿事件时，此位被置 1。该位可通过向其写入 1 的方式清零。

17.6.12 EXTI 特权配置寄存器 2 (EXTI_PRIVCFGR2)

EXTI privilege configuration register 2

偏移地址：0x038

复位值：0x0000 0000

此寄存器提供特权写访问保护。非特权读访问会返回寄存器数据。

仅包含特权输入事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIV53	Res.	Res.	PRIV50	PRIV49	Res.
										rw			rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRIV47	Res.	Res.	Res.	Res.	PRIV42	PRIV41	PRIV40	PRIV39	PRIV38	PRIV37	Res.	Res.	Res.	Res.	Res.
rw					rw	rw	rw	rw	rw	rw					

位 31:22、20:19、16、14:11、4:0 保留，必须保持复位值。

位 21 **PRIV53**：事件输入 x 上的特权使能 (Privilege enable on event input x)
0：禁止事件特权（非特权）
1：使能事件特权（特权）

位 18:17 **PRIVx**：事件输入 x 上的特权使能 (Privilege enable on event input x)（x = 50 到 49）
0：禁止事件特权（非特权）
1：使能事件特权（特权）

位 15 **PRIV47**：事件输入 x 上的特权使能 (Privilege enable on event input x)
0：禁止事件特权（非特权）
1：使能事件特权（特权）

位 10:5 **PRIVx**：事件输入 x 上的特权使能 (Privilege enable on event input x)（x = 42 到 37）
0：禁止事件特权（非特权）
1：使能事件特权（特权）

17.6.13 EXTI 外部中断选择寄存器 (EXTI_EXTICR1)

EXTI external interrupt selection register

偏移地址：0x060 EXTI 复用器 0、1、2、3

复位值：0x0000 0000

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

位 31:24 **EXTI3[7:0]**：EXTI3 GPIO 端口选择 (EXTI3 GPIO port selection)
这些位通过软件写入，用于选择 EXTI3 外部中断的源输入。
EXTI_PRIVCFGR.PRIV3 处于禁止状态时，EXTI3 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV3 处于使能状态时，EXTI3 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[3] 引脚
0x01：PB[3] 引脚
0x02：PC[3] 引脚
其他值：保留

位 23:16 **EXTI2[7:0]**：EXTI2 GPIO 端口选择 (EXTI2 GPIO port selection)
这些位通过软件写入，用于选择 EXTI2 外部中断的源输入。
EXTI_PRIVCFGR.PRIV2 处于禁止状态时，EXTI2 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV2 处于使能状态时，EXTI2 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[2] 引脚
0x01：PB[2] 引脚
0x02：PC[2] 引脚
0x03：PD[2] 引脚
其他值：保留

位 15:8 **EXTI1[7:0]**：EXTI1 GPIO 端口选择 (EXTI1 GPIO port selection)
这些位通过软件写入，用于选择 EXTI1 外部中断的源输入。
EXTI_PRIVCFGR.PRIV1 处于禁止状态时，EXTI1 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV1 处于使能状态时，EXTI1 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[1] 引脚
0x01：PB[1] 引脚
0x02：PC[1] 引脚
0x07：PH[1] 引脚
其他值：保留

位 7:0 **EXTI0[7:0]**：EXTI0 GPIO 端口选择 (EXTI0 GPIO port selection)
这些位通过软件写入，用于选择 EXTI0 外部中断的源输入。
EXTI_PRIVCFGR.PRIV0 处于禁止状态时，EXTI0 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV0 处于使能状态时，EXTI0 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[0] 引脚
0x01：PB[0] 引脚
0x02：PC[0] 引脚
0x07：PH[0] 引脚
其他值：保留

17.6.14 EXTI 外部中断选择寄存器 (EXTI_EXTICR2)

EXTI external interrupt selection register

偏移地址：0x064 EXTI 复用器 4、5、6、7

复位值：0x0000 0000

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

位 31:24 **EXTI7[7:0]**：EXTI7 GPIO 端口选择 (EXTI7 GPIO port selection)
这些位通过软件写入，用于选择 EXTI7 外部中断的源输入。
EXTI_PRIVCFGR.PRIV7 处于禁止状态时，EXTI7 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV7 处于使能状态时，EXTI7 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[7] 引脚
0x01：PB[7] 引脚
0x02：PC[7] 引脚
其他值：保留

位 23:16 **EXTI6[7:0]**：EXTI6 GPIO 端口选择 (EXTI6 GPIO port selection)

这些位通过软件写入，用于选择 EXTI6 外部中断的源输入。

EXTI_PRIVCFGR.PRIV6 处于禁止状态时，EXTI6 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIV6 处于使能状态时，EXTI6 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0x00：PA[6] 引脚

0x01：PB[6] 引脚

0x02：PC[6] 引脚

其他值：保留

位 15:8 **EXTI5[7:0]**：EXTI5 GPIO 端口选择 (EXTI5 GPIO port selection)

这些位通过软件写入，用于选择 EXTI5 外部中断的源输入。

EXTI_PRIVCFGR.PRIV5 处于禁止状态时，EXTI5 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIV5 处于使能状态时，EXTI5 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0x00：PA[5] 引脚

0x01：PB[5] 引脚

0x02：PC[5] 引脚

其他值：保留

位 7:0 **EXTI4[7:0]**：EXTI4 GPIO 端口选择 (EXTI4 GPIO port selection)

这些位通过软件写入，用于选择 EXTI4 外部中断的源输入。

EXTI_PRIVCFGR.PRIV4 处于禁止状态时，EXTI4 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIV4 处于使能状态时，EXTI4 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0x00：PA[4] 引脚

0x01：PB[4] 引脚

0x02：PC[4] 引脚

其他值：保留

17.6.15 EXTI 外部中断选择寄存器 (EXTI_EXTICR3)

EXTI external interrupt selection register

偏移地址：0x068 EXTI 复用器 8、9、10、11

复位值：0x0000 0000

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

位 31:24 **EXTI11[7:0]**：EXTI11 GPIO 端口选择 (EXTI11 GPIO port selection)

这些位通过软件写入，用于选择 EXTI11 外部中断的源输入。

EXTI_PRIVCFGR.PRIV11 处于禁止状态时，EXTI11 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIV11 处于使能状态时，EXTI11 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0x00：PA[11] 引脚

0x02：PC[11] 引脚

其他值：保留

位 23:16 **EXTI10[7:0]**：EXTI10 GPIO 端口选择 (EXTI10 GPIO port selection)
这些位通过软件写入，用于选择 EXTI10 外部中断的源输入。
EXTI_PRIVCFGR.PRIV10 处于禁止状态时，EXTI10 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV10 处于使能状态时，EXTI10 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[10] 引脚
0x01：PB[10] 引脚
0x02：PC[10] 引脚
其他值：保留

位 15:8 **EXTI9[7:0]**：EXTI9 GPIO 端口选择 (EXTI9 GPIO port selection)
这些位通过软件写入，用于选择 EXTI9 外部中断的源输入。
EXTI_PRIVCFGR.PRIV9 处于禁止状态时，EXTI9 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV9 处于使能状态时，EXTI9 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[9] 引脚
0x02：PC[9] 引脚
其他值：保留

位 7:0 **EXTI8[7:0]**：EXTI8 GPIO 端口选择 (EXTI8 GPIO port selection)
这些位通过软件写入，用于选择 EXTI8 外部中断的源输入。
EXTI_PRIVCFGR.PRIV8 处于禁止状态时，EXTI8 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV8 处于使能状态时，EXTI8 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[8] 引脚
0x01：PB[8] 引脚
0x02：PC[8] 引脚
其他值：保留

17.6.16 EXTI 外部中断选择寄存器 (EXTI_EXTICR4)

EXTI external interrupt selection register

偏移地址：0x06C EXTI 复用器 12、13、14、15

复位值：0x0000 0000

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

位 31:24 **EXTI15[7:0]**：EXTI15 GPIO 端口选择 (EXTI15 GPIO port selection)
这些位通过软件写入，用于选择 EXTI15 外部中断的源输入。
EXTI_PRIVCFGR.PRIV15 处于禁止状态时，EXTI15 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV15 处于使能状态时，EXTI15 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[15] 引脚
0x01：PB[15] 引脚
0x02：PC[15] 引脚
其他值：保留

位 23:16 **EXTI14[7:0]**：EXTI14 GPIO 端口选择 (EXTI14 GPIO port selection)

这些位通过软件写入，用于选择 EXTI14 外部中断的源输入。
EXTI_PRIVCFGR.PRIV14 处于禁止状态时，EXTI14 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV14 处于使能状态时，EXTI14 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[14] 引脚
0x01：PB[14] 引脚
0x02：PC[14] 引脚
其他值：保留

位 15:8 **EXTI13[7:0]**：EXTI13 GPIO 端口选择 (EXTI13 GPIO port selection)

这些位通过软件写入，用于选择 EXTI13 外部中断的源输入。
EXTI_PRIVCFGR.PRIV13 处于禁止状态时，EXTI13 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV13 处于使能状态时，EXTI13 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[13] 引脚
0x01：PB[13] 引脚
0x02：PC[13] 引脚
其他值：保留

位 7:0 **EXTI12[7:0]**：EXTI12 GPIO 端口选择

这些位通过软件写入，用于选择 EXTI12 外部中断的源输入。
EXTI_PRIVCFGR.PRIV12 处于禁止状态时，EXTI12 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIV12 处于使能状态时，EXTI12 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0x00：PA[12] 引脚
0x01：PB[12] 引脚
0x02：PC[12] 引脚
其他值：保留

17.6.17 EXTI CPU 唤醒中断屏蔽寄存器 (EXTI_IMR1)

EXTI CPU wakeup with interrupt mask register

偏移地址：0x080

复位值：0xFFFE 0000

包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	IM29	IM28	IM27	IM26	IM25	IM24	Res.	IM22	IM21	Res.	IM19	Res.	IM17	IM16
		rw	rw	rw	rw	rw	rw		rw	rw		rw		rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:30、23、20、18 保留，必须保持复位值。

位 29:24 **IMx**：事件输入 x 上的 CPU 唤醒中断屏蔽 (CPU wakeup with interrupt mask on event input x)(1)（x = 29 到 24）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，IMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，IMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽来自输入事件 x 的唤醒中断请求。

1：取消屏蔽来自输入事件 x 的唤醒中断请求。

位 22:21 **IMx**：事件输入 x 上的 CPU 唤醒中断屏蔽 (CPU wakeup with interrupt mask on event input x)(1)（x = 22 到 21）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，IMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，IMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽来自输入事件 x 的唤醒中断请求。

1：取消屏蔽来自输入事件 x 的唤醒中断请求。

位 19 **IM19**：事件输入 x 上的 CPU 唤醒中断屏蔽 (CPU wakeup with interrupt mask on event input x)(1)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，IMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，IMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽来自输入事件 x 的唤醒中断请求。

1：取消屏蔽来自输入事件 x 的唤醒中断请求。

位 17:0 **IMx**：事件输入 x 上的 CPU 唤醒中断屏蔽 (CPU wakeup with interrupt mask on event input x)(1)（x = 17 到 0）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，IMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，IMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽来自输入事件 x 的唤醒中断请求。

1：取消屏蔽来自输入事件 x 的唤醒中断请求。

1. 可配置事件输入的复位值置 0，以在默认情况下禁止中断。

17.6.18 EXTI CPU 唤醒事件屏蔽寄存器 (EXTI_EMR1)

EXTI CPU wakeup with event mask register

偏移地址：0x084

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	EM29	EM28	EM27	EM26	EM25	EM24	Res.	EM22	EM21	Res.	EM19	Res.	EM17	EM16
		rw	rw	rw	rw	rw	rw		rw	rw		rw		rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:30、23、20、18 保留，必须保持复位值。

位 29:24 **EMx**：事件输入 x 上的 CPU 唤醒事件生成屏蔽 (CPU wakeup with event generation mask on event input x)（x = 29 到 24）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，EMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，EMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽线 x 上生成的唤醒事件。

1：取消屏蔽线 x 上生成的唤醒事件。

位 22:21 **EMx**：事件输入 x 上的 CPU 唤醒事件生成屏蔽 (CPU wakeup with event generation mask on event input x)（x = 22 到 21）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，EMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，EMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽线 x 上生成的唤醒事件。

1：取消屏蔽线 x 上生成的唤醒事件。

位 19 **EM19**：事件输入 x 上的 CPU 唤醒事件生成屏蔽 (CPU wakeup with event generation mask on event input x)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，EMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，EMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽线 x 上生成的唤醒事件。

1：取消屏蔽线 x 上生成的唤醒事件。

位 17:0 **EMx**：事件输入 x 上的 CPU 唤醒事件生成屏蔽 (CPU wakeup with event generation mask on event input x)（x = 17 到 0）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，EMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，EMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽线 x 上生成的唤醒事件。

1：取消屏蔽线 x 上生成的唤醒事件。

17.6.19 EXTI CPU 唤醒中断屏蔽寄存器 2 (EXTI_IMR2)

EXTI CPU wakeup with interrupt mask register 2

偏移地址：0x090

复位值：0x001B FFFF

包含可配置事件的寄存器位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IM53	Res.	Res.	IM50	IM49	Res.
										rw			rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IM47	Res.	Res.	Res.	Res.	IM42	IM41	IM40	IM39	IM38	IM37	Res.	Res.	Res.	Res.	Res.
rw					rw	rw	rw	rw	rw	rw					

位 31:22、20:19、16、14:11、4:0 保留，必须保持复位值。

位 21 **IM53**：事件输入 x 上的 CPU 唤醒中断屏蔽 (CPU wakeup with interrupt mask on event input x) (1)
EXTI_PRIVCFGR.PRIVx 处于禁止状态时，IMx 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，IMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0：屏蔽来自输入事件 x 的唤醒中断请求。
1：取消屏蔽来自输入事件 x 的唤醒中断请求。

位 18:17 **IMx**：事件输入 x 上的 CPU 唤醒中断屏蔽 (CPU wakeup with interrupt mask on event input x)(1)（x = 50 到 49）
EXTI_PRIVCFGR.PRIVx 处于禁止状态时，IMx 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，IMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0：屏蔽来自输入事件 x 的唤醒中断请求。
1：取消屏蔽来自输入事件 x 的唤醒中断请求。

位 15 **IM47**：事件输入 x 上的 CPU 唤醒中断屏蔽 (CPU wakeup with interrupt mask on event input x)(1)
EXTI_PRIVCFGR.PRIVx 处于禁止状态时，IMx 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，IMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0：屏蔽来自输入事件 x 的唤醒中断请求。
1：取消屏蔽来自输入事件 x 的唤醒中断请求。

位 10:5 **IMx**：事件输入 x 上的 CPU 唤醒中断屏蔽 (CPU wakeup with interrupt mask on event input x)(1)（x = 42 到 37）
EXTI_PRIVCFGR.PRIVx 处于禁止状态时，IMx 可通过特权和非特权访问进行访问。
EXTI_PRIVCFGR.PRIVx 处于使能状态时，IMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。
0：屏蔽来自输入事件 x 的唤醒中断请求。
1：取消屏蔽来自输入事件 x 的唤醒中断请求。

1. 可配置事件输入的复位值置 0，以在默认情况下禁止中断。

17.6.20 EXTI CPU 唤醒事件屏蔽寄存器 2 (EXTI_EMR2)

EXTI CPU wakeup with event mask register 2

偏移地址：0x094

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EM53	Res.	Res.	EM50	EM49	Res.
										rw			rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM47	Res.	Res.	Res.	Res.	EM42	EM41	EM40	EM39	EM38	EM37	Res.	Res.	Res.	Res.	Res.
rw					rw	rw	rw	rw	rw	rw					

位 31:22、20:19、16、14:11、4:0 保留，必须保持复位值。

位 21 **EM53**：事件输入 x 上的 CPU 唤醒事件生成屏蔽 (CPU wakeup with event generation mask on event input x)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，EMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，EMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽线 x 上生成的唤醒事件。

1：取消屏蔽线 x 上生成的唤醒事件。

位 18:17 **EMx**：事件输入 x 上的 CPU 唤醒事件生成屏蔽 (CPU wakeup with event generation mask on event input x)（x = 50 到 49）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，EMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，EMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽线 x 上生成的唤醒事件。

1：取消屏蔽线 x 上生成的唤醒事件。

位 15 **EM47**：事件输入 x 上的 CPU 唤醒事件生成屏蔽 (CPU wakeup with event generation mask on event input x)

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，EMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，EMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽线 x 上生成的唤醒事件。

1：取消屏蔽线 x 上生成的唤醒事件。

位 10:5 **EMx**：事件输入 x 上的 CPU 唤醒事件生成屏蔽 (CPU wakeup with event generation mask on event input x)（x = 42 到 37）

EXTI_PRIVCFGR.PRIVx 处于禁止状态时，EMx 可通过特权和非特权访问进行访问。

EXTI_PRIVCFGR.PRIVx 处于使能状态时，EMx 只能通过特权访问进行访问。对该位执行的非特权写访问会被忽略。

0：屏蔽线 x 上生成的唤醒事件。

1：取消屏蔽线 x 上生成的唤醒事件。

17.6.21 EXTI 寄存器映射

表 103. EXTI 寄存器映射和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**EXTI_RTSR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RT16	RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x004	**EXTI_FTSR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FT16	FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x008	**EXTI_SWIER1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWI16	SWI15	SWI14	SWI13	SWI12	SWI11	SWI10	SWI9	SWI8	SWI7	SWI6	SWI5	SWI4	SWI3	SWI2	SWI1	SWI0
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00C	**EXTI_RPR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RPIF1	RPIF1	RPIF1	RPIF1	RPIF1	RPIF1	RPIF1	RPIF9	RPIF8	RPIF7	RPIF6	RPIF5	RPIF4	RPIF3	RPIF2	RPIF1	RPIF0
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x010	**EXTI_FPR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FPIF16	FPIF15	FPIF14	FPIF13	FPIF12	FPIF11	FPIF10	FPIF9	FPIF8	FPIF7	FPIF6	FPIF5	FPIF4	FPIF3	FPIF2	FPIF1	FPIF0
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 103. EXTI 寄存器映射和复位值（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x018	**EXTI_ PRIVCFGR1**	Res.	Res.	PRIV29	PRIV28	PRIV27	PRIV26	PRIV25	PRIV24	Res.	PRIV22	PRIV21	Res.	PRIV19	Res.	PRIV17	PRIV16	PRIV15	PRIV14	PRIV13	PRIV12	PRIV11	PRIV10	PRIV9	PRIV8	PRIV7	PRIV6	PRIV5	PRIV4	PRIV3	PRIV2	PRIV1	PRIV0
	Reset value			0	0	0	0	0	0		0	0		0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x020	**EXTI_RTSR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RT53	Res.	Res.	RT50	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value											0			0																		
0x024	**EXTI_FTSR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FT53	Res.	Res.	FT50	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value											0			0																		
0x028	**EXTI_SWIER2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWI53	Res.	Res.	SWI50	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value											0			0																		
0x02C	**EXTI_RPR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RPIF53	Res.	Res.	RPIF50	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value											0			0																		
0x030	**EXTI_FPR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FPIF53	Res.	Res.	FPIF50	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value											0			0																		
0x038	**EXTI_ PRIVCFGR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIV53	Res.	Res.	PRIV50	PRIV49	Res.	PRIV47	Res.	Res.	Res.	Res.	PRIV42	PRIV41	PRIV40	PRIV39	PRIV38	PRIV37	Res.	Res.	Res.	Res.	Res.
	Reset value											0			0	0		0					0	0	0	0	0	0					
0x03C to 0x05C	**Reserved**	Reserved																															
0x060	**EXTI_EXTICR1**	EXTI3[7:0]								EXTI2[7:0]								EXTI1[7:0]								EXTI0[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x064	**EXTI_EXTICR2**	EXTI7[7:0]								EXTI6[7:0]								EXTI5[7:0]								EXTI4[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x068	**EXTI_EXTICR3**	EXTI11[7:0]								EXTI10[7:0]								EXTI9[7:0]								EXTI8[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x06C	**EXTI_EXTICR4**	EXTI15[7:0]								EXTI14[7:0]								EXTI13[7:0]								EXTI12[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x074 to 0x07C	**Reserved**	Reserved																															
0x080	**EXTI_IMR1**	Res.	Res.	IM29	IM28	IM27	IM26	IM25	IM24	Res.	IM22	IM21	Res.	IM19	Res.	IM17	IM16	IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
	Reset value			1	1	1	1	1	1		1	1		1		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x084	**EXTI_EMR1**	Res.	Res.	EM29	EM28	EM27	EM26	EM25	EM24	Res.	EM22	EM21	Res.	EM19	Res.	EM17	EM16	EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
	Reset value			0	0	0	0	0	0		0	0		0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x090	**EXTI_IMR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IM53	Res.	Res.	IM50	IM49	Res.	IM47	Res.	Res.	Res.	Res.	IM42	IM41	IM40	IM39	IM38	IM37	Res.	Res.	Res.	Res.	Res.
	Reset value											0			0	1		1					1	1	1	1	1	1					

表 103. EXTI 寄存器映射和复位值（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x094	**EXTI_EMR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EM53	Res.	Res.	EM50	EM49	Res.	EM47	Res.	Res.	Res.	Res.	EM42	EM41	EM40	EM39	EM38	EM37	Res.	Res.	Res.	Res.	Res.
	Reset value											0			0	0		0					0	0	0	0	0	0					

有关寄存器边界地址的信息，请参见*第 2.2 节*。

18 循环冗余校验计算单元 (CRC)

18.1 简介

CRC（循环冗余校验）计算单元使用多项式发生器从一个 8 位、16 位或 32 位的数据字中产生 CRC 码。

在众多的应用中，基于 CRC 的技术还常用来验证数据传输或存储的完整性。根据功能安全标准的规定，这些技术提供了验证 Flash 完整性的方法。CRC 计算单元有助于在运行期间计算软件的签名，并将该签名与链接时生成并存储在指定存储单元的参考签名加以比较。

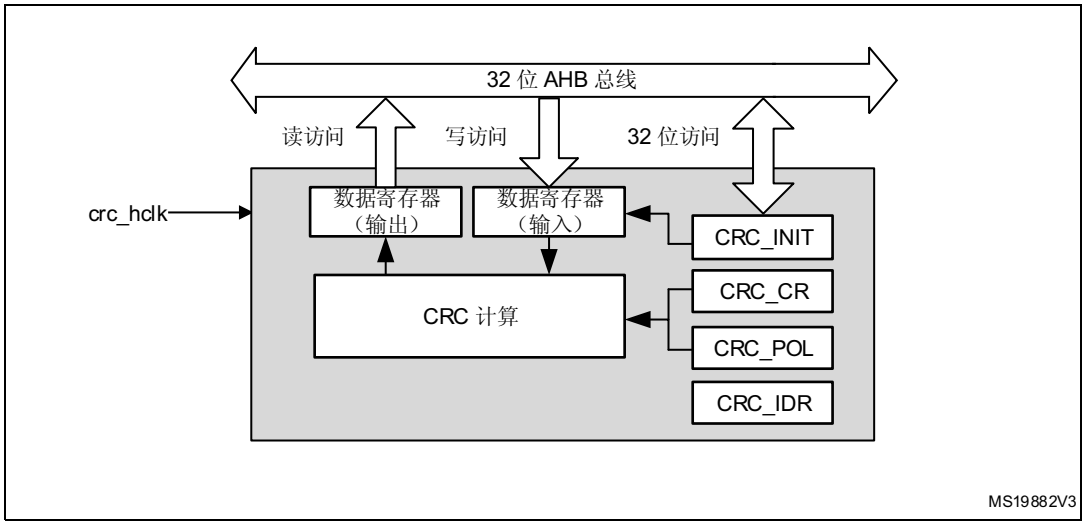
18.2 CRC 主要特性

- 使用 CRC-32（以太网）多项式：0x4C11DB7
  $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- 或者，使用位数可编程的（7 位、8 位、16 位和 32 位）的完全可编程多项式
- 处理 8 位、16 位、32 位数据大小
- 可编程 CRC 初始值
- 单输入/输出 32 位数据寄存器
- 输入缓冲器可避免计算期间发生总线阻塞
- 对于 32 位数据大小，CRC 计算在 4 个 AHB 时钟周期 (HCLK) 内完成
- 8 位通用寄存器（可用于临时存储）
- I/O 数据有可逆性选项
- 仅通过 32 位字对 AHB 从外设进行访问（例外情况：CRC_DR 寄存器可按字、右对齐半字和右对齐字节进行访问）

18.3 CRC 功能说明

18.3.1 CRC 框图

图 74. CRC 计算单元框图



18.3.2 CRC 内部信号

表 104. CRC 内部输入/输出信号

信号名称	信号类型	说明
crc_hclk	数字输入	AHB 时钟

18.3.3 CRC 运算

CRC 计算单元具有单个 32 位读/写数据寄存器 (CRC_DR)。它用于输入新数据（写访问）和保存之前 CRC 计算的结果（读访问）。

对数据寄存器的每个写操作都会对之前的 CRC 值（存储在 CRC_DR 中）和新值再做一次 CRC 计算。CRC 计算针对整个 32 位数据字或逐个字节完成，具体取决于数据的写入格式。

CRC_DR 寄存器可按字、右对齐半字和右对齐字节进行访问。对于其他寄存器，只允许按 32 位进行访问。

计算时间取决于数据宽度：

- 32 位数据需要 4 个 AHB 时钟周期
- 16 位数据需要 2 个 AHB 时钟周期
- 8 位数据需要 1 个 AHB 时钟周期

输入缓冲区允许立即写入第二个数据，而无需等待由于先前的 CRC 计算而导致的任何等待状态。

可动态调整数据大小，从而能最大程度地减少给定字节数的写访问次数。例如，对 5 个字节进行 CRC 计算时，可先写入字，然后写入字节。

为管理各种数据存放方式（大端/小端），输入数据的顺序可以反转。可对 8 位、16 位和 32 位数据执行反转操作，具体取决于 CRC_CR 寄存器中的 REV_IN[1:0] 位。

例如，0x1A2B3C4D 输入数据在 CRC 计算中用作：

- 0x58D43CB2，按字节执行位反转
- 0xD458B23C，按半字执行位反转
- 0xB23CD458，按全字执行位反转

通过将 CRC_CR 寄存器中 REV_OUT 位置 1 也可以将输出数据反转。

该操作按位进行。例如，输出数据 0x11223344 将转换为 0x22CC4488。

使用 CRC_CR 寄存器中的 RESET 控制位可将 CRC 计算器初始化为可编程值（默认值为 0xFFFFFFFF）。

可使用 CRC_INIT 寄存器对 CRC 初始值进行编程。对 CRC_INIT 寄存器进行写访问时会自动初始化 CRC_DR 寄存器。

CRC_IDR 寄存器可用于保存与 CRC 计算相关的临时值。它不受 CRC_CR 寄存器中的 RESET 位影响。

多项式可编程

多项式系数可完全通过 CRC_POL 寄存器进行编程，通过编程 CRC_CR 寄存器中的 POLYSIZE[1:0] 位可将多项式大小配置为 7 位、8 位、16 位或 32 位。不支持偶数多项式。

注意： *偶数多项式的形式为 $X+X^2+..+X^n$，而奇数多项式的形式为 $1+X+X^2+..+X^n$。*

如果 CRC 数据小于 32 位，可从 CRC_DR 寄存器的最低有效位读取 CRC 的值。

为实现可靠的 CRC 计算，CRC 计算期间不能实时更改多项式的值或大小。因此，如果正在执行 CRC 计算，则在更改多项式前，应用程序必须先复位，或者先执行 CRC_DR 读操作。

默认的多项式值为 CRC-32（以太网）多项式：0x4C11DB7。

18.4 CRC 寄存器

CRC_DR 寄存器可按字、右对齐半字和右对齐字节进行访问。对于其他寄存器，只允许按 32 位进行访问。

18.4.1 CRC 数据寄存器 (CRC_DR)

CRC data register

偏移地址：0x00

复位值：0xFFFF FFFF

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

位 31:0 **DR[31:0]**：数据寄存器位

该寄存器用于向 CRC 计算器写入新数据。

读取寄存器时可读出之前的 CRC 计算结果。

如果数据大小小于 32 位，则最低有效位可用于写入/读取正确值。

18.4.2 CRC 独立数据寄存器 (CRC_IDR)

CRC independent data register

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
IDR[31:16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDR[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **IDR[31:0]**：通用的 32 位数据寄存器位

这些位可用作四个字节的临时存储单元。

该寄存器不受 CRC_CR 寄存器中 RESET 位产生的 CRC 复位影响

18.4.3 CRC 控制寄存器 (CRC_CR)

CRC control register

偏移地址：0x08

复位值：0x0000 0000

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

位 31:8 保留，必须保持复位值。

位 7 **REV_OUT**：反转输出数据 (Reverse output data)

该位用于控制输出数据位顺序的反转。

0：位顺序不受影响

1：按位反转输出

位 6:5 **REV_IN[1:0]**：反转输入数据 (Reverse input data)

该位域用于控制输入数据位顺序的反转

00：位顺序不受影响

01：按字节执行位反转

10：按半字执行位反转

11：按字执行位反转

位 4:3 **POLYSIZE[1:0]**：多项式大小 (Polynomial size)

这些位用于控制多项式的大小。

00：32 位多项式

01：16 位多项式

10：8 位多项式

11：7 位多项式

位 2:1 保留，必须保持复位值。

位 0 **RESET**：RESET 位

该位由软件置 1，用于复位 CRC 计算单元并将数据寄存器设置为存储在 CRC_INIT 寄存器中的值。该位只能置 1，将由硬件自动进行清零

18.4.4 CRC 初始值 (CRC_INIT)

CRC initial value

偏移地址：0x10

复位值：0xFFFF FFFF

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

位 31:0 **CRC_INIT[31:0]**：可编程 CRC 初始值 (Programmable initial CRC value)
该寄存器用于写入 CRC 初始值。

18.4.5 CRC 多项式 (CRC_POL)

CRC polynomial

偏移地址：0x14

复位值：0x04C1 1DB7

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

位 31:0 **POL[31:0]**：可编程多项式 (Programmable polynomial)
该寄存器用于写入要用于 CRC 计算的多项式系数。
如果多项式大小小于 32 位，则必须使用最低有效位编程正确值。

18.4.6 CRC 寄存器映射

表 105. CRC 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CRC_DR	DR[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x04	CRC_IDR	IDR[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	CRC_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REV_OUT	REV_IN[1:0]		POLYSIZE[1:0]		Res.	Res.	RESET
	Reset value																									0	0	0	0	0			0
0x10	CRC_INIT	CRC_INIT[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x14	CRC_POL	POL[31:0]																															
	Reset value	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	1	0	0	0	1	1	1	0	1	1	0	1	1	0	1	1	1

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

19 模数转换器 (ADC1)

19.1 简介

ADC 由 12 位逐次逼近模数转换器组成。

ADC 的复用通道多达 20 条。各种不同通道的 A/D 转换可在单次、连续、扫描或不连续采样模式下进行。ADC 的结果存储在一个左对齐或右对齐的 16 位数据寄存器中。

ADC 映射到 AHB 总线，从而可实现快速数据处理。

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

内置硬件过采样器，可提高模拟性能，同时还能减轻 CPU 进行相关计算的负担。

采用了高效的低功耗模式，在低频下可实现极低的功耗。

19.2 ADC 主要特性

- 高性能特性
  - 16 个外部通道和 4 个内部通道
  - 可配置 12 位、10 位、8 位或 6 位分辨率
  - ADC 转换时间与 AHB 总线时钟频率无关
  - 通过降低分辨率缩短转换时间
  - 可管理单端或差分输入
  - AHB 从总线接口，可实现快速数据处理
  - 自校准
  - 可独立设置各通道采样时间
  - 灵活控制采样时间
  - 多达 4 条注入通道（对常规通道或注入通道的模拟输入分配完全可配置）
  - 硬件辅助准备注入通道的上下文，从而实现快速上下文切换
  - 数据对齐以保持内置数据一致性
  - 数据可由 DMA 管理，以实现常规通道转换
  - 四个专用数据寄存器供注入通道使用
- 低功耗特性
  - 速度自适应低功耗模式，可降低 ADC 在低频下工作时的功耗
  - 可在实现慢速总线频率应用的同时保持最佳 ADC 性能
  - 提供自动控制，可避免 ADC 在低 AHB 总线时钟频率应用中溢出（自动延迟模式）
- 过采样器
  - 16 位数据寄存器
  - 过采样率可在 2x 到 256x 之间进行调整
  - 可编程数据最多可移位 8 位

- 数据预处理
  - 偏移补偿
- 模拟输入通道
  - 外部模拟输入：
    GPIO 引脚有多达 5 条快速通道
    GPIO 引脚有多达 11 条慢速通道
  - 1 条用于内部温度传感器 (V_{SENSE}) 的通道
  - 1 条用于 内部参考电压的通道 (V_{REFINT})
  - 1 条用于监视外部 VBAT 电源引脚的通道
  - 1 条用于监视内部 V_{DDCORE} 电源的通道
- 可通过以下方式启动转换过程：
  - 通过软件启动常规转换和注入转换
  - 通过极性可配置的硬件触发器（内部定时器事件或 GPIO 输入事件）启动常规转换和注入转换
- 转换模式
  - ADC 可转换单条通道，也可扫描一系列通道
  - 单次模式会在每次触发时对选定的输入执行一次转换
  - 连续模式可连续转换选定的输入
  - 不连续模式
- 在 ADC 准备就绪、采样结束、转换结束（常规转换或注入转换）、序列转换结束（常规转换或注入转换）、模拟看门狗 1/2/3 事件或溢出事件时生成中断
- 3 个模拟看门狗
  - 看门狗可以执行滤波以忽略超出范围的数据
- ADC 输入范围：$V_{SSA} \leq V_{IN} \leq V_{REF+}$

图 75 给出了一个 ADC 的框图。

19.3 ADC 实现

表 106. ADC 特性

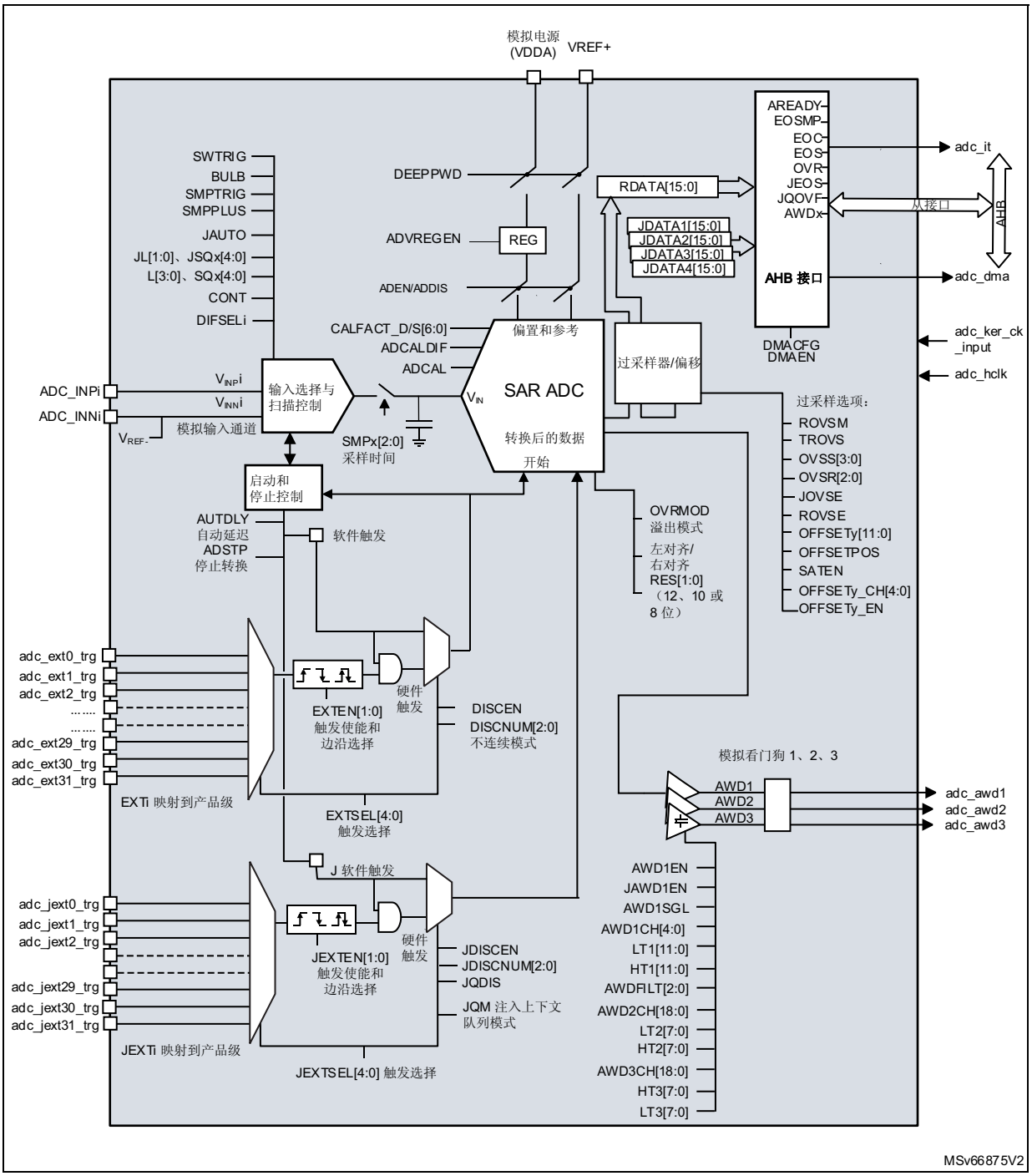
ADC 模式/特性	ADC1
分辨率	12 位
最大采样速度	5 Msps （12 位分辨率）
双模式操作	-
硬件偏移校准	X
硬件线性校准	-
单端输入	X
差分输入	X
注入通道转换	X
过采样	高达 x256
数据寄存器	16 位
数据寄存器 FIFO 深度	3 级
支持 DMA	X
并行数据输出到 ADF	-
偏移补偿	X
增益补偿	-
模拟看门狗数	3
选项寄存器	X

19.4 ADC 功能说明

19.4.1 ADC 框图

图 75 给出了 ADC 框图，*表 107* 列出了 ADC 的引脚说明。

图 75. ADC 框图



19.4.2 ADC 引脚和内部信号

表 107. ADC 输入/输出引脚

引脚名称	信号类型	说明
VDDA	模拟电源输入	ADC 的模拟电源和正参考电压
VSSA	模拟电源地输入	模拟电源地电，压等于 V_{SS}。
VREF+	正模拟参考电压输入	ADC 高/正参考电压。
VREF-	负模拟参考电压输入	ADC 低/负参考电压。V_{REF-} 内部连接到 V_{SSA}
ADC1_INNi/INPi	负/正外部模拟输入信号	20 负/正外部模拟输入通道（有关详细信息，请参见*第 19.4.4 节：ADC 连接*）

表 108. ADC 内部输入/输出信号

内部信号名称	信号类型	说明
$V_{INP}i$	正模拟输入通道	连接到 ADC1_INP*i* 外部通道或者连接到内部通道的正内部模拟输入通道。
$V_{INN}i$	负模拟输入通道	连接到 ADC1_INN*i* 外部通道或者连接到内部通道的内部负模拟输入通道
adc_ext_trgi	输入	用于常规转换的 ADC 外部触发输入。
adc_jext_trgi	输入	用于注入转换的 ADC 外部触发输入。
adc_awdx	输出	内部模拟看门狗输出信号，连接至片上定时器。（x = 模拟看门狗编号 1、2、3）
adc_ker_ck_input	输出	ADC 内核时钟
adc_hclk	输入	ADC 外设时钟
adc_it	输出	ADC 中断
adc_dma	输出	ADC DMA 请求

表 109. ADC 互连

信号名称	源/目标
ADC1 V_{INP}[16]	V_{SENSE}（内部温度传感器输出电压）。
ADC1 V_{INP}[17]	V_{REFINT}（内部参考电压的输出电压）。
ADC1 V_{INP}[2]	V_{BAT}/4（VBAT 引脚输入电压除以 4）。
ADC1 V_{INP}[6]	V_{DDCORE}（内部数字内核电压）。
adc_ext_trg0	tim1_oc1
adc_ext_trg1	tim1_oc2
adc_ext_trg2	tim1_oc3
adc_ext_trg3	tim2_oc2
adc_ext_trg4	tim3_trgo
adc_ext_trg5	保留
adc_ext_trg6	exti11
adc_ext_trg7	保留

表 109. ADC 互连 （续）

信号名称	源/目标
adc_ext_trg8	保留
adc_ext_trg9	tim1_trgo
adc_ext_trg10	tim1_trgo2
adc_ext_trg11	tim2_trgo
adc_ext_trg12	保留
adc_ext_trg13	tim6_trgo
adc_ext_trg14	保留
adc_ext_trg15	tim3_oc4
adc_ext_trg16	exti15
adc_ext_trg17	tim7_trgo
adc_ext_trg18	lptim1_ch1
adc_ext_trg19	lptim2_ch1
adc_ext_trg20	保留
adc_ext_trg21	保留
adc_ext_trg22	保留
adc_ext_trg23	保留
adc_ext_trg24	保留
adc_ext_trg25	保留
adc_ext_trg26	保留
adc_ext_trg27	保留
adc_ext_trg28	保留
adc_ext_trg29	保留
adc_ext_trg30	保留
adc_ext_trg31	保留
adc_jext_trg0	tim1_trgo
adc_jext_trg1	tim1_oc4
adc_jext_trg2	tim2_trgo
adc_jext_trg3	tim2_oc1
adc_jext_trg4	tim3_oc4
adc_jext_trg5	保留
adc_jext_trg6	exti15
adc_jext_trg7	保留
adc_jext_trg8	tim1_trgo2
adc_jext_trg9	保留
adc_jext_trg10	保留

表 109. ADC 互连 （续）

信号名称	源/目标
adc_jext_trg11	tim3_oc3
adc_jext_trg12	tim3_trgo
adc_jext_trg13	tim3_oc1
adc_jext_trg14	tim6_trgo
adc_jext_trg15	保留
adc_jext_trg16	保留
adc_jext_trg17	tim7_trgo
adc_jext_trg18	lptim1_ch1
adc_jext_trg19	lptim2_ch1
adc_jext_trg20	保留
adc_jext_trg21	保留
adc_jext_trg22	保留
adc_jext_trg23	保留
adc_jext_trg24	保留
adc_jext_trg25	保留
adc_jext_trg26	保留
adc_jext_trg27	保留
adc_jext_trg28	保留
adc_jext_trg29	保留
adc_jext_trg30	保留
adc_jext_trg31	保留

19.4.3 ADC 时钟

双时钟域架构

双时钟域架构意味着 ADC 时钟独立于 AHB 总线时钟。

可以在两个不同的时钟源之间选择 ADC 输入时钟（参见*图 76：ADC 时钟方案*）：

1. ADC 时钟可以是特定时钟源 (adc_ker_ck_input)，该时钟源独立于 AHB 时钟，并与 AHB 时钟异步。

   有关如何生成 ADC 专用时钟的更多信息，请参见*复位和时钟控制 (RCC)* 部分。要选择此时钟方案，ADC_CCR 寄存器的 CKMODE[1:0] 位必须设置为 00。

2. ADC 时钟可由 AHB 时钟除以一个可编程的因数 1、2 或 4 来提供。要选择此时钟方案，ADC_CCR 寄存器的 CKMODE[1:0] 位不得为 00。可编程的分频系数可以通过 ADC_CCR 的 CKMODE[1:0] 位来配置。

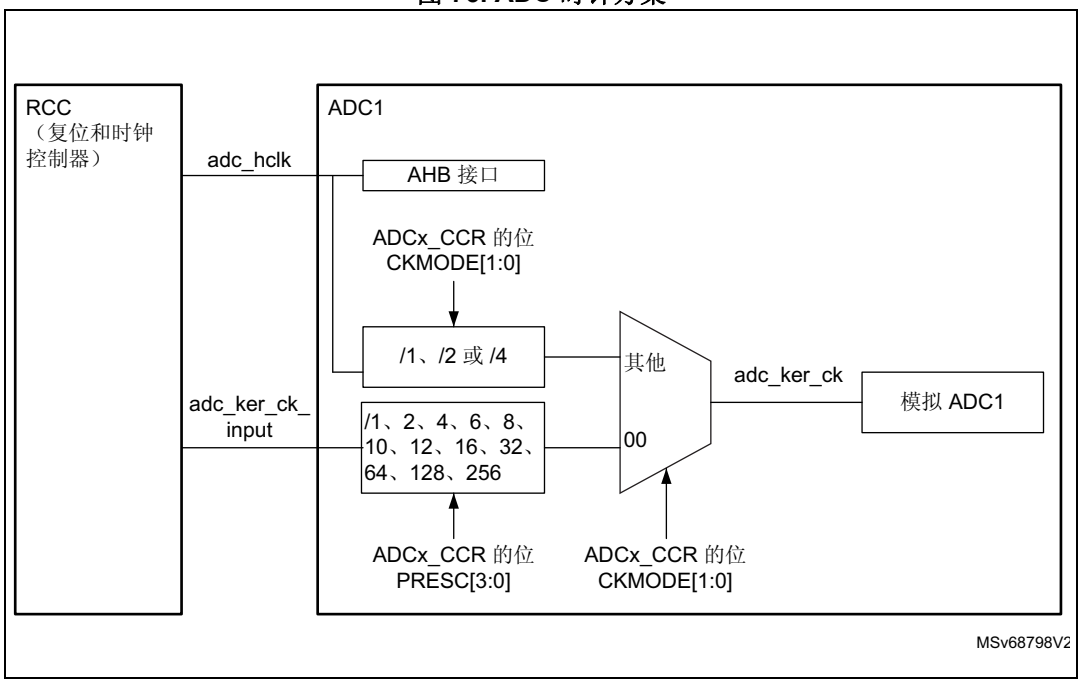
   仅当 AHB 预分频器置 1（RCC_CFGR 寄存器中的 HPRE[3:0] = 0xxx）时，才能使用预分频系数 1 (CKMODE[1:0] = 01)。

选项 **1** 的优点在于无论选择哪种 AHB 时钟方案，都可以达到最大 ADC 时钟频率。使用通过 ADC_CCR 寄存器中的 PRESC[3:0] 位配置的预分频器时，ADC 时钟最后会除以以下比率：1、2、4、6、8、12、16、32、64、128、256。

选项 **2** 的优势在于绕过了时钟域重新同步。如果 ADC 由定时器触发，并且应用要求 ADC 精确触发（不存在任何不确定性），可使用该选项（否则，重新同步两个时钟域会为触发时刻带来不确定性）。

通过 CKMODE[1:0] 位配置的时钟必须符合器件数据手册中指定的工作频率。

图 76. ADC 时钟方案



ADC 时钟与 AHB 时钟的时钟比例限制

通常来讲，ADC 时钟与 AHB 时钟之比没有限制，但一些注入通道已编程的情况除外，此时，ADC 时钟与 AHB 时钟之比必须遵循以下规定：

- 如果所有通道的分辨率均为 12 位或 10 位，则 $F_{adc_hclk} \geq F_{ADC}/4$
- 如果一些通道的分辨率为 8 位（且所有通道的分辨率均不低于 8 位），则 $F_{adc_hclk} \geq F_{ADC}/3$
- 如果一些通道的分辨率为 6 位，则 $F_{adc_hclk} \geq F_{ADC}/2$

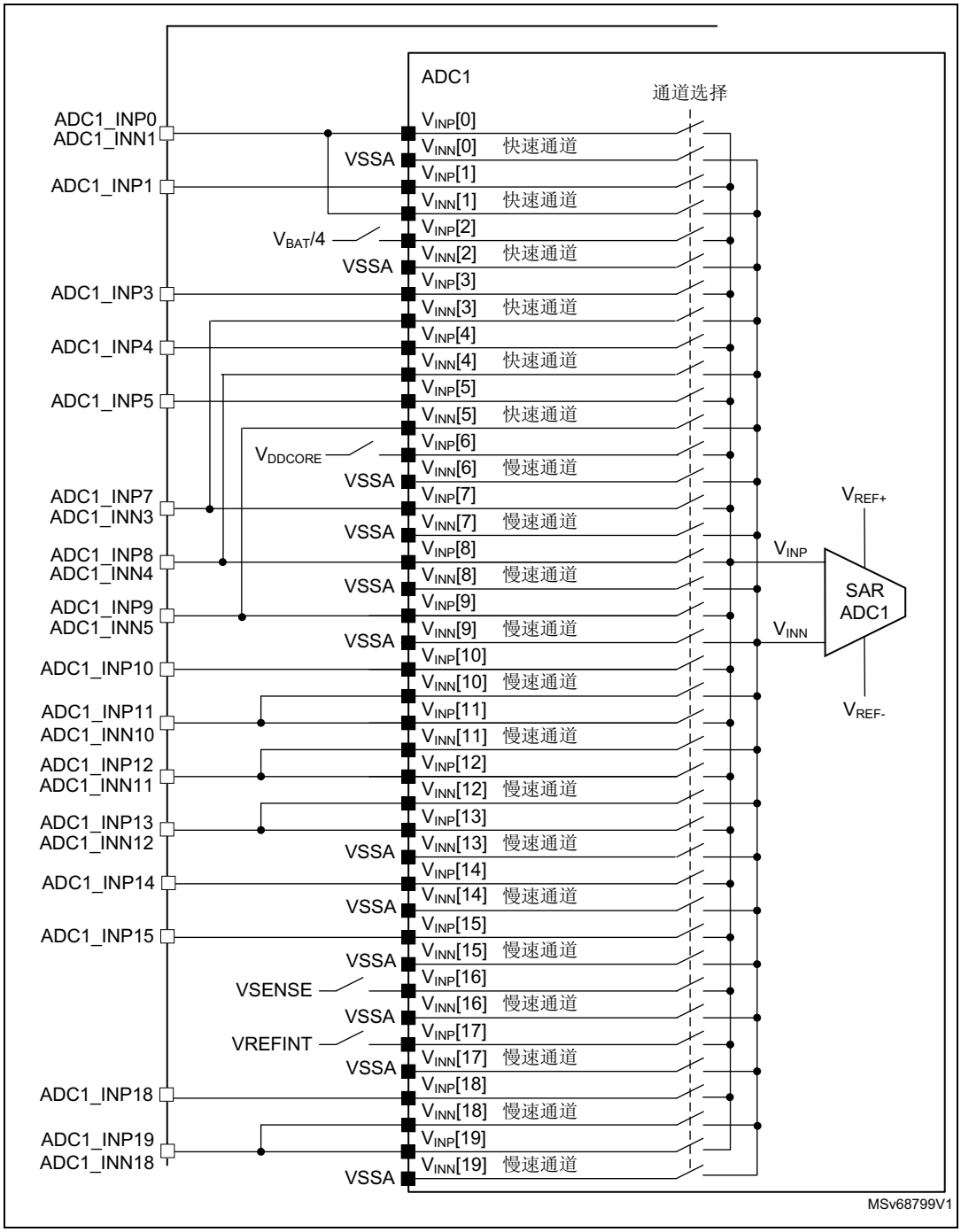
ADC 时钟之间的限制

如果同时使用多个 ADC 接口，则必须对所有 ADC 接口使用来自 RCC 模块的相同时钟，并且不使用预分频比。

19.4.4 ADC 连接

ADC 输入连接到外部通道和内部源，如下所述。

图 77. ADC1 连接功能



19.4.5 从设备 AHB 接口

ADC 利用 AHB 从设备端口实现控制/状态寄存器访问和数据访问。AHB 接口的特性如下：

- 字（32 位）访问
- 单周期响应
- 以零等待状态响应对寄存器的所有读/写访问

AHB 从设备接口不支持分离/重试请求，且绝不会生成 AHB 错误。

19.4.6 ADC 深度掉电模式 (DEEPPWD) 和 ADC 调压器 (ADVREGEN)

ADC 默认处于深度掉电模式，在该模式下，其电源会在内部切断以降低泄漏电流（ADC_CR 寄存器中的 DEEPPWD 位的复位状态为 1）。

要开始 ADC 操作，需要先将 DEEPPWD 位清零，使 ADC 退出深度掉电模式。

然后，必须将 ADC_CR 寄存器中的 ADVREGEN 位置 1，以使能 ADC 内部调压器。软件必须在 ADC 调压器的启动时间 ($T_{ADCVREG_STUP}$) 内等待，之后才能开始校准或使能 ADC。该延迟必须通过软件实现。

有关 ADC 调压器的启动时间，请参见器件数据手册中的 $T_{ADCVREG_STUP}$ 参数。

ADC 操作完成后，可禁止 ADC (ADEN = 0)。此外，还可通过禁止 ADC 调压器实现节能。这可通过向 ADVREGEN 位写入 0 来实现。

之后，为了通过降低泄漏电流的方式进一步实现节能，还可以将 ADC_CR 寄存器中的 DEEPPWD 位置 1，以使 ADC 再次进入深度掉电模式。进入停止模式之前，这一点特别值得关注。

注意： *写入 DEEPPWD = 1 会自动禁止 ADC 调压器，并且 ADVREGEN 位会自动清零。*

如果内部调压器已禁止 (ADVREGEN = 0)，则会保持内部模拟校准。

在 ADC 深度掉电模式下 (DEEPPWD = 1)，内部模拟校准会丢失，需要重新启动校准或重新应用之前保存的校准系数（请参见*第 19.4.8 节：校准（ADCAL、ADCALDIF、ADC_CALFACT）*）。

19.4.7 单端输入通道和差分输入通道

可通过编程 ADC_DIFSEL 寄存器中的 DIFSEL[i] 位将通道配置为单端输入或差分输入。必须在禁止 ADC (ADEN = 0) 时写入此配置。注意，与各个单端通道对应的 DIFSEL[i] 位始终编程为 0。

在单端输入模式下，要通过通道“i”转换的模拟电压是外部电压 $V_{INP[i]}$（正输入）和 V_{REF-}（负输入）之差。

在差分输入模式下，要为通道“i”转换的模拟电压是外部电压 $V_{INP[i]}$（正输入）与 $V_{INN[i]}$（负输入）之差。

差分模式的输出数据是无符号数据。当 $V_{INP[i]}$ 等于 V_{REF-} 时，$V_{INN[i]}$ 等于 V_{REF+}，并且输出数据为 0x000（12 位分辨率模式）。当 $V_{INP[i]}$ 等于 V_{REF+} 时，$V_{INN[i]}$ 等于 V_{REF-}，并且输出数据为 0xFFF。

$$\text{转换后的值} = \frac{\text{ADC_Full_Scale}}{2} \times \left[1 + \frac{V_{INP} - V_{INN}}{V_{REF+}}\right]$$

当 ADC 配置为差分模式时，两路输入的偏置电压均应为 (V_{REF+})/2。

输入信号应为差分信号（共模电压应固定）。

内部通道（例如 V_{REFINT} 和 V_{SENSE}）仅用于单端模式。

有关输入通道连接方式的完整说明，请参见*第 19.4.4 节：ADC 连接*。

小心： 将通道“i”配置为差分输入模式时，其负输入电压 $V_{INN[i]}$ 连接至另一个通道。因此，在单端模式或差分模式下，该通道不再可用，并且不得配置为进行转换。

19.4.8 校准（ADCAL、ADCALDIF、ADC_CALFACT）

ADC 提供有自动校准流程，用于驱动包括 ADC 上电/掉电序列在内的所有校准序列。在校准过程中，ADC 会计算 7 位宽度的校准系数，并会在下一次 ADC 掉电之前在内部应用该系数。在校准过程中，应用不得使用 ADC，必须等待至校准完成。

执行任何 ADC 操作之前都必须进行校准。校准可消除偏移误差，由于制造工艺或带隙的不同，各芯片的偏移误差也有所不同。

单端输入转换与差分输入转换应用的校准系数有所不同：

- 启动要为单端输入转换应用的校准之前，写入 ADCALDIF = 0。
- 启动要为差分输入转换应用的校准之前，写入 ADCALDIF = 1。

随后，通过软件将 ADCAL 位置 1 发起校准。仅当 ADC 禁止 (ADEN = 0) 后，才能发起校准。ADCAL 位在所有校准序列过程中保持为 1。校准完成后，该位会立即由硬件清零。此时，相关校准系数会存储在模拟 ADC 内部，同时还会存储在 ADC_CALFACT 寄存器的 CALFACT_S[6:0] 或 CALFACT_D[6:0] 位中（具体取决于是单端输入校准还是差分输入校准）

如果禁止 ADC (ADEN = 0)，则会保留内部模拟校准。但如果长时间禁止 ADC，建议在重新使能 ADC 之前运行新的校准周期。

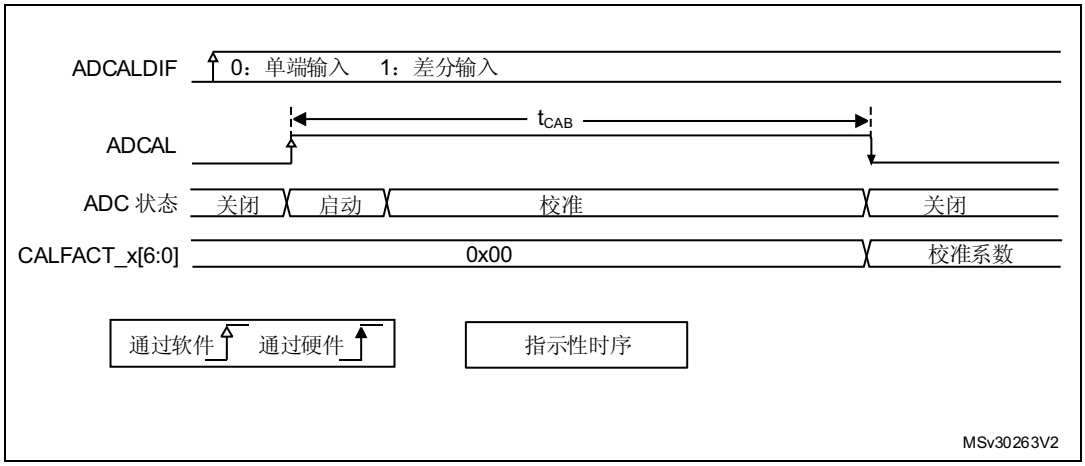
每次 ADC 掉电时，内部模拟校准都会丢失（例如，产品进入待机模式或 VBAT 模式时）。这种情况下，为了避免浪费时间重新校准 ADC，可将校准系数重新写入 ADC_CALFACT 寄存器，但前提是软件之前保存了上次校准时得出的校准系数。

如果 ADC 已使能但未进行转换（ADEN = 1、ADSTART = 0 且 JADSTART = 0），可写入校准系数。随后，下次转换启动时，校准系数会自动添加到模拟 ADC 中。这一载入过程是透明的，不会对转换的启动造成延迟。建议在 V_{REF+} 电压变化超过 10% 时重新进行校准。

ADC 校准的软件流程

1. 确保 DEEPPWD = 0、ADVREGEN = 1，并确保 ADC 调压器启动时间已过。
2. 确保 ADEN = 0。
3. 设置 ADCALDIF = 0（单端输入）或 ADCALDIF = 1（差分输入），选择此校准的输入模式。
4. 将 ADCAL 置 1。
5. 等待至 ADCAL = 0。
6. 可从 ADC_CALFACT 寄存器读取校准系数。

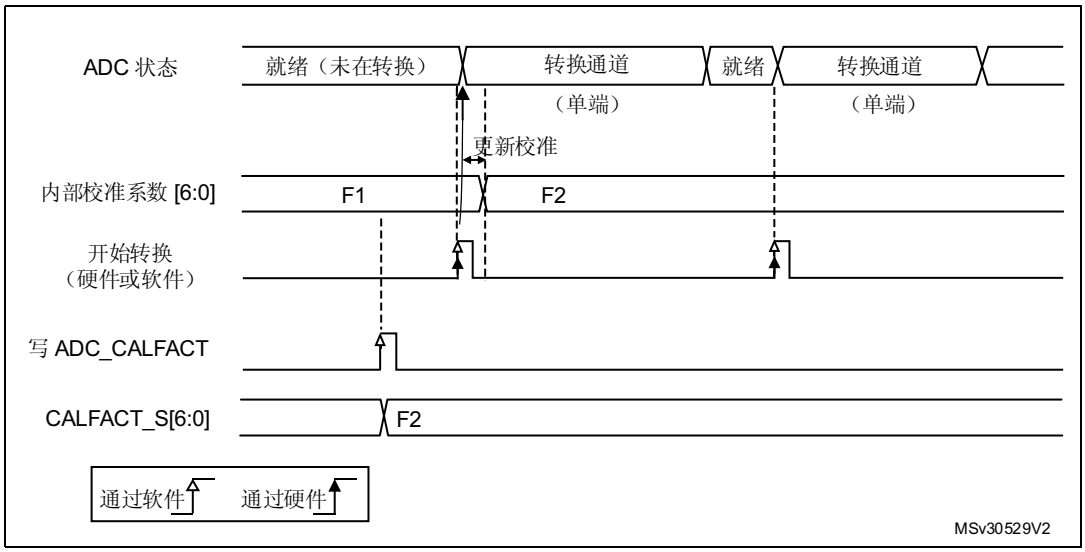
图 78. ADC 校准



将校准系数重新注入 ADC 的软件流程

1. 确保 ADEN = 1、ADSTART = 0 且 JADSTART = 0（ADC 已使能，且未进行任何转换）。
2. 将新的校准系数写入 CALFACT_S 和 CALFACT_D。
3. 启动转换时，仅当内部模拟校准系数与 CALFACT_S 位（单端输入通道）或 CALFACT_D 位（差分输入通道）中存储的值不同时，校准系数才会注入到模拟 ADC。

图 79. 更新 ADC 校准系数

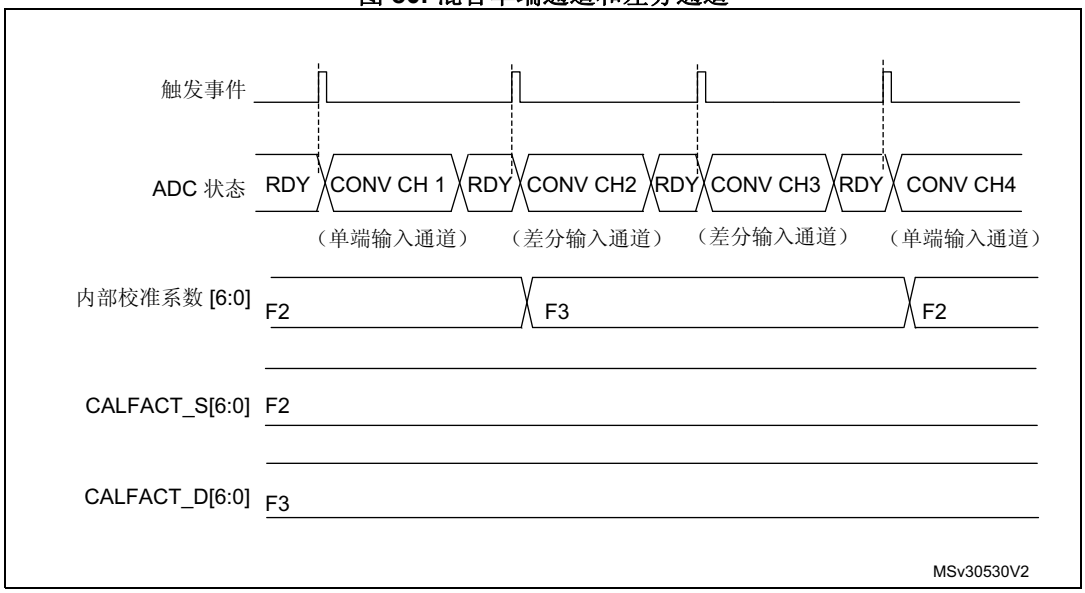


单个 ADC 转换单端模拟输入和差分模拟输入

如果 ADC 既要转换差分输入，也要转换单端输入，则必须执行两次校准，两次校准分别在 ADCALDIF = 0 和 ADCALDIF = 1 的情况下执行。过程如下：

1. 禁止 ADC。
2. 在单端输入模式下 (ADCALDIF = 0) 校准 ADC。此操作会更新寄存器 CALFACT_S[6:0]。
3. 在差分输入模式下 (ADCALDIF = 1) 校准 ADC。此操作会更新寄存器 CALFACT_D[6:0]。
4. 使能 ADC、配置通道并启动转换。每次从单端输入通道切换为差分输入通道（或进行相反切换）时，校准都将自动注入到模拟 ADC 中。

图 80. 混合单端通道和差分通道



19.4.9 ADC 开关控制（ADEN、ADDIS、ADRDY）

首先按照*第 19.4.6 节：ADC 深度掉电模式 (DEEPPWD) 和 ADC 调压器 (ADVREGEN)* 中的流程执行操作。

DEEPPWD = 0 且 ADVREGEN = 1 后，可使能 ADC，并且 ADC 在开始精确转换之前需要 t_{STAB} 的稳定时间，如*图 81* 所示。以下两个控制位可使能或禁止 ADC：

- 将 ADEN 置 1 可使能 ADC。ADC 准备就绪可以运行后，ADRDY 标志会立即置 1。
- 将 ADDIS 置 1 可禁止 ADC。模拟 ADC 被有效禁止后，ADEN 和 ADDIS 会立即由硬件自动清零。

随后可通过将 ADSTART 置 1（请参见*第 19.4.18 节：外部触发转换和触发极性（EXTSEL、EXTEN、JEXTSEL、JEXTEN）*）开始进行常规转换，如果触发器已使能，也可在发生外部触发事件时开始进行常规转换。

可通过将 JADSTART 置 1 开始进行注入转换，如果注入触发器已使能，也可在发生外部注入触发事件时开始进行注入转换。

通过软件使能 ADC 的流程

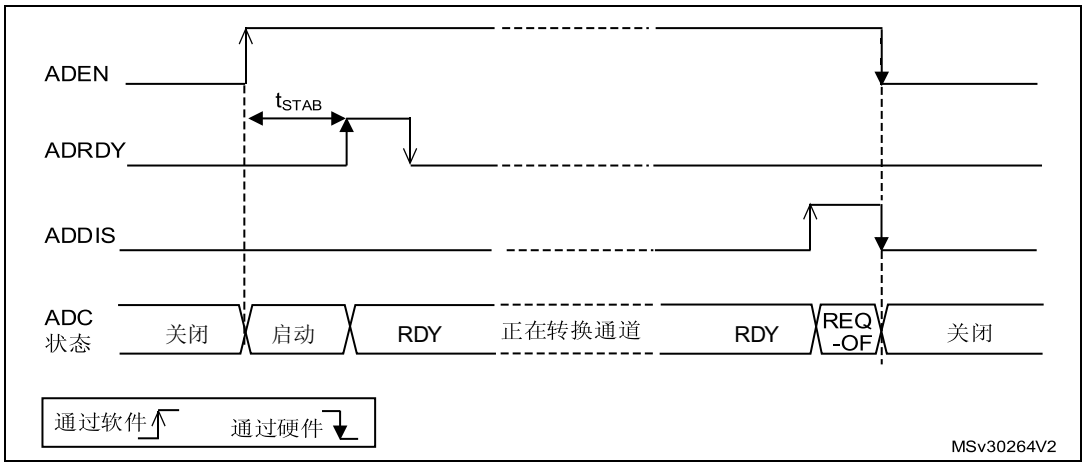
1. 向 ADC_ISR 寄存器中的 ADRDY 位写入“1”，将其清零。
2. 将 ADEN 置 1。
3. 等待至 ADRDY = 1（ADRDY 会在 ADC 启动时间后置 1）。这可以使用关联中断来实现（将 ADRDYIE 置 1）。
4. 向 ADC_ISR 寄存器中的 ADRDY 位写入“1”，将其清零（可选）。

小心： 如果 ADCAL 置 1 并且是在 ADCAL 位由硬件清零（校准结束）后的 4 个 ADC 时钟周期内，则 ADEN 位无法置 1。

通过软件禁止 ADC 的流程

1. 检查 ADSTART 和 JADSTART 是否均为 0，以确保当前未执行任何转换。如有需要，可将 ADSTP 置 1，并将 JADSTP 置 1，随后等待至 ADSTP = 0 且 JADSTP = 0，以停止任何正在进行的常规转换和注入转换。
2. 将 ADDIS 置 1。
3. 如果应用要求，可等待 ADEN = 0，直至模拟 ADC 已有效禁止（ADEN = 0 后，ADDIS 将自动复位）。

图 81. 使能/禁止 ADC



19.4.10 写入 ADC 控制位时的限制

仅当 ADC 已禁止（ADEN 必须等于 0）时，软件才可通过写入 RCC 控制位的方式配置和使能 ADC 时钟（请参见 RCC 部分）、ADC_DIFSEL 寄存器中的 DIFSEL[i] 控制位、ADC_CR 寄存器中的 ADCAL 和 ADEN 控制位。

只有在 ADC 已使能、并且没有待处理的禁止 ADC 的请求（ADEN 必须等于 1 且 ADDIS 必须等于 0）时，才允许软件向 ADC_CR 寄存器的控制位 ADSTART、JADSTART 和 ADDIS 执行写操作。

对于 ADC_CFGR、ADC_SMPRx、ADC_TRy、ADC_SQRy、ADC_JDRy、ADC_OFRy、ADC_OFCHRy 和 ADC_IER 寄存器的所有其他控制位：

- 对于与常规转换配置相关的控制位，仅当 ADC 已使能 (ADEN = 1) 且未进行常规转换（ADSTART 必须等于 0）时，才允许软件对这些位执行写操作。
- 对于与注入转换配置相关的控制位，仅当 ADC 已使能 (ADEN = 1) 且未进行注入转换（JADSTART 必须等于 0）时，才允许软件对这些位执行写操作。
- 可以在进行模数转换时修改 ADC_TRy 寄存器（有关详细信息，请参见*第 19.4.28 节：模拟窗口看门狗（AWD1EN、JAWD1EN、AWD1SGL、AWD1CH、AWD2CH、AWD3CH、AWD_HTx、AWD_LTx、AWDx）*）。

仅当 ADC 已使能（可能会进行转换）且没有待处理的禁止 ADC 的请求（ADSTART 或 JADSTART 必须等于 1 且 ADDIS 必须等于 0）时，软件才可以对 ADC_CR 寄存器的 ADSTP 或 JADSTP 控制位执行写操作。

如果 ADC 已使能 (ADEN = 1)，软件可随时对 ADC_JSQR 寄存器执行写操作。更多详细信息，请参见*第 19.6.16 节：ADC 注入序列寄存器 (ADC_JSQR)*。

注意： *没有硬件保护可以防止执行此类禁止的写操作，ADC 操作可能进入未知状态。要从此状态恢复，必须禁止 ADC（将 ADEN 清零，并且将 ADC_CR 寄存器的所有位都清零）。*

19.4.11 通道选择（SQRx、JSQRx）

ADC 具有多达 20 个复用通道，其中：

- 最多 16 个模拟输入来自 GPIO 引脚 (ADC_INP/INN[i])，视产品而定，并非所有这些输入都在 GPIO 引脚上提供。
- ADC 连接 4 个内部模拟输入：
  - 内部温度传感器 (V_{SENSE})
  - 内部参考电压 (V_{REFINT})
  - V_{BAT} 监测通道 ($V_{BAT}/4$)
  - 内部数字内核电压 (V_{DDCORE})

  要对其中一条内部模拟通道进行转换，必须先对 ADC_CCR 寄存器中的 VREFEN、VREFEN 或 TSEN 位进行编程，以使能相应的模拟源。

有关上述内部模拟输入与外部 ADC 引脚或内部信号的连接，请参见*第 19.4.2 节：ADC 引脚和内部信号*中的 *ADC 互连表*。

转换可以分为两组：常规转换和注入转换。每个组包含一个转换序列，该序列可按任意顺序在任意通道上完成。例如，可按以下顺序对序列进行转换：ADC1_INP/INN3，ADC1_INP/INN8，ADC1_INP/INN2，ADC1_INN/INP2，ADC1_INP/INN0，ADC1_INP/INN2，ADC1_INP/INN2，ADC1_INP/INN15。

- 一个**常规转换组**最多由 16 个转换构成。必须在 ADC_SQRy 寄存器中选择转换序列的常规通道及其顺序。常规转换组中的转换总数必须写入 ADC_SQR1 寄存器中的 L[3:0] 位。
- 一个**注入转换组**最多由 4 个转换构成。必须在 ADC_JSQR 寄存器中选择转换序列的注入通道及其顺序。注入转换组中的转换总数必须写入 ADC_JSQR 寄存器中的 L[1:0] 位。

不得在进行常规转换时对 ADC_SQRy 寄存器进行修改。因此，必须先写入 ADSTP = 1 停止 ADC 常规转换（请参见*第 19.4.17 节：停止正在进行的转换（ADSTP、JADSTP）*）。

仅在使能上下文队列（ADC_CFGR 寄存器中的 JQDIS = 0）时，才会在 JADSTART 置 1（正在进行注入转换）的条件下，允许软件实时修改 ADC_JSQR 寄存器。请参见*第 19.4.21 节：注入转换的上下文队列*

19.4.12 可独立设置各通道采样时间（SMPR1、SMPR2）

开始转换之前，ADC 必须在待测量电压源与 ADC 内置采样电容之间建立直接连接。该采样时间必须足以使输入电压源为内置电容充电至输入电压水平。

对各通道进行采样时可以使用不同的采样时间，采样时间可通过 ADC_SMPR1 和 ADC 寄存器中的 SMP[2:0] 位编程。可选采样时间值如下：

- SMP = 000：2.5 个 ADC 时钟周期
- SMP = 001：6.5 个 ADC 时钟周期
- SMP = 010：12.5 个 ADC 时钟周期
- SMP = 011：24.5 个 ADC 时钟周期
- SMP = 100：47.5 个 ADC 时钟周期
- SMP = 101：92.5 个 ADC 时钟周期
- SMP = 110：247.5 个 ADC 时钟周期
- SMP = 111：640.5 个 ADC 时钟周期

总转换时间的计算公式如下：

T_{CONV} = 采样时间 + 12.5 个 ADC 时钟周期

示例：

如果 $F_{adc_ker_ck}$ = 30 MHz，采样时间为 2.5 个 ADC 时钟周期：

T_{CONV} = (2.5 + 12.5) 个 ADC 时钟周期 = 15 个 ADC 时钟周期 = 500 ns

ADC 通过将状态位 EOSMP 置 1 来指示采样阶段结束（仅限常规转换）。

注意： *在不同的 ADC 转换模式下，实时采样时间可能会不同于上述设定的 SMP 值，而总转换时间 (T_{CONV}) 不变：*

- *对于扫描或连续模式下的第一次转换以及不连续模式下的所有转换，实时采样时间比上述配置值少 0.5 个时钟周期。*
- *对于扫描或连续模式下的第二次转换和后续转换，配置的采样时间值要加上 0.5 个时钟周期。增加的这 0.5 个时钟周期与上一个转换周期重叠。*

采样时间限制

对于每条通道，必须对 SMP[2:0] 位进行编程，以符合数据手册 ADC 特性部分规定的最短采样时间要求。

Bulb 采样模式

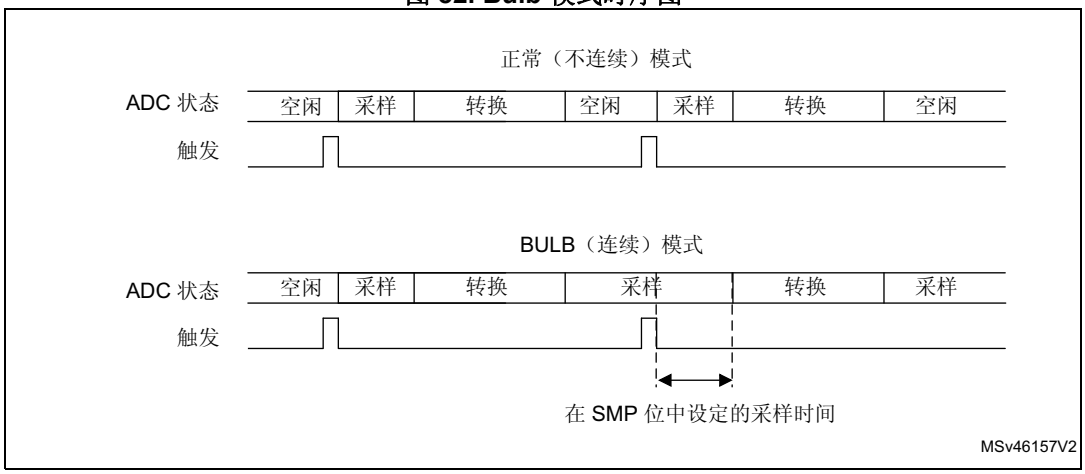
当 ADC 寄存器中的 BULB 位置 1 时，采样周期在上一次 ADC 转换后立即开始。在 ADC_SMPR1 寄存器中设定了采样时间后，会通过硬件触发或软件触发启动转换。在 ADC 使能后执行第一次 ADC 转换时，采样时间在 SMP 位中设定。Bulb 模式从第二次转换开始生效。

最大采样时间存在限制（请参见数据手册中的 ADC 特性部分）。

Bulb 模式既不兼容连续转换模式，也不兼容注入通道转换。

当 BULB 位置 1 时，不允许将 ADC_CFGR2 中的 SMPTRIG 位置 1。

图 82. Bulb 模式时序图



采样时间控制触发模式

当 SMPTRIG 位置 1 时，通过 SMPx 位设定的采样时间不适用。采样时间受触发信号边沿控制。

如果选择了硬件触发，则触发信号的每个上升沿均会启动采样周期。下降沿会结束采样周期，并开始进行转换。

如果选择了软件触发，则软件触发并不是 ADC_CR 中的 ADSTART 位，而是 SWTRIG 位。SWTRIG 位必须置 1 才能启动采样周期，SWTRIG 位必须清零才能结束采样周期并开始进行转换。

最大采样时间存在限制（请参见数据手册中的 ADC 特性部分）。

此模式既不兼容连续转换模式，也不兼容注入通道转换。

当 SMPTRIG 位置 1 时，不允许将 BULB 位置 1。

I/O 模拟开关升压器

当 V_{DDA} 电压过低时，I/O 模拟开关的电阻会增大。因此必须相应地调整采样时间（有关相应的电气特性，请参见器件数据手册）。通过 BOOSTE 位使能内部升压器，或者通过 PWR_PMCR 寄存器的 ADV_READY 位选择 V_{DD} 升压器电压（如果 V_{DD} > 2.7 V），均可以在 V_{DDA} 为低电压值时最小化该电阻。

SMPPLUS 控制位

SMPPLUS 位可用于将采样时间从 2.5 个 ADC 时钟周期改为 3.5 个 ADC 时钟周期。

19.4.13 单次转换模式 (CONT = 0)

在单次转换模式下，ADC 会将通道的所有转换执行一次。CONT 位为 0 时，可通过以下方式启动此模式：

- 将 ADC_CR 寄存器中的 ADSTART 位置 1（适用于常规通道）
- 将 ADC_CR 寄存器中的 JADSTART 位置 1（适用于注入通道）
- 外部硬件触发事件（适用于常规通道或注入通道）

在常规序列中，每次转换完成后：

- 转换数据存储在 16 位 ADC_DR 寄存器中
- EOC（常规转换结束）标志置 1
- EOCIE 位置 1 时将产生中断

在注入序列中，每次转换完成后：

- 转换数据存储在四个 16 位 ADC_JDRy 寄存器的其中一个寄存器中
- JEOC（注入转换结束）标志置 1
- JEOCIE 位置 1 时将产生中断

常规序列完成后：

- EOS（常规序列结束）标志置 1
- EOSIE 位置 1 时将产生中断

注入序列完成后：

- JEOS（注入序列结束）标志置 1
- JEOSIE 位置 1 时将产生中断

随后，ADC 会停止工作，直至发生新的外部常规或注入触发，或者 ADSTART 或 JADSTART 位再次置 1。

注意： *要转换单个通道，可将序列长度编程为 1。*

19.4.14 连续转换模式 (CONT = 1)

该模式仅适用于常规通道。

在连续转换模式下，如果发生软件或硬件常规触发事件，ADC 会将通道的所有常规转换执行一次，随后会自动重启并持续执行序列的每个转换。CONT 位为 1 时，可通过外部触发或将 ADC_CR 寄存器中的 ADSTART 位置 1 来启动此模式。

在常规序列中，每次转换完成后：

- 转换数据存储在 16 位 ADC_DR 寄存器中
- EOC（转换结束）标志置 1
- EOCIE 位置 1 时将产生中断

转换序列完成后：

- EOS（序列结束）标志置 1
- EOSIE 位置 1 时将产生中断

随后，会立即重启新序列，ADC 会继续重复执行转换序列。

注意： *要转换单个通道，可将序列长度编程为 1。*

不能同时使能不连续模式和连续模式：禁止同时将 DISCEN 和 CONT 位置 1。

注入通道不能连续转换，唯一例外的是，在连续转换模式下（使用 JAUTO 位）注入通道配置为在常规通道后的自动转换，请参见自动注入模式一节。

19.4.15 开始转换（ADSTART、JADSTART）

软件通过将 ADSTART 置 1 的方式开始进行 ADC 常规转换。

ADSTART 置 1 后，会开始进行转换：

- 立即开始转换：EXTEN = 0x0 时（软件触发）
- 在所选常规硬件触发的下一有效边沿开始转换：EXTEN 不等于 0x0 时

软件通过将 JADSTART 置 1 的方式开始进行 ADC 注入转换。

JADSTART 置 1 后，会开始进行转换：

- 立即开始转换：JEXTEN = 0x0 时（软件触发）
- 在所选注入硬件触发的下一有效边沿开始转换：JEXTEN 不等于 0x0 时

注意： *在自动注入模式下 (JAUTO = 1)，使用 ADSTART 位开始常规转换，然后再进行自动注入转换（JADSTART 必须保持清零）。*

ADSTART 位和 JADSTART 位还提供当前是否正在进行 ADC 操作的信息。可以在 ADSTART = 0 且 JADSTART = 0（指示 ADC 处于空闲状态）时重新配置 ADC。

ADSTART 通过硬件清零：

- 在使用软件常规触发的单次模式下（CONT = 0，EXTSEL = 0x0）
  - 如果 DISCEN = 1，只要常规转换序列结束（EOS 置 1）或子组处理结束就清零
- 在所有情况下（CONT = x、EXTSEL = x）
  - 执行由软件调用的 ADSTP 程序之后清零。

注意： *在连续模式下 (CONT = 1)，由于序列会自动重新启动，因此，当 EOS 置 1 时，ADSTART 位不会通过硬件清零。*

如果在单次模式下选择了硬件触发（CONT = 0 且 EXTSEL 不等于 0x00），当 EOS 置 1 时，ADSTART 位不会通过硬件清零，借此软件无需再次为下一个硬件触发事件复位 ADSTART。这样可确保不会错过任何后续的硬件触发。

JADSTART 通过硬件清零：

- 在使用软件注入触发的单次模式下 (JEXTSEL = 0x0)
  - 如果 JDISCEN = 1，只要注入转换序列结束（JEOS 置 1）或子组处理结束就清零
- 在所有情况下 (JEXTSEL = x)
  - 执行由软件调用的 JADSTP 程序之后清零。

注意： *选择软件触发时，如果 EOC 标志仍为高电平，则不应将 ADSTART 位置 1。*

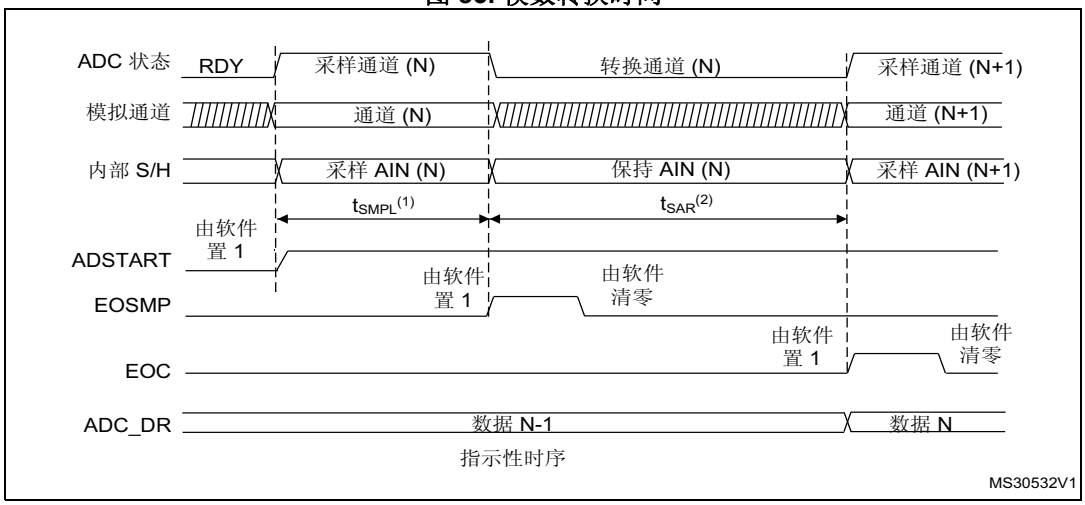
19.4.16 ADC 时序

从转换开始到转换结束所经过的时间是配置的采样时间与逐次逼近时间（具体取决于数据分辨率）的总和：

$T_{CONV} = T_{SMPL} + T_{SAR} = [2.5\ _{|min} + 12.5\ _{|12bit}] \times T_{ADC_CLK}$

$T_{CONV} = T_{SMPL} + T_{SAR} = 83.33\ ns\ _{|min} + 416.67\ ns\ _{|12bit} = 500.0\ ns$（$F_{ADC_CLK} = 30\ MHz$ 时）

图 83. 模数转换时间



1. T_{SMPL} 取决于 SMP[2:0]。
2. T_{SAR} 取决于 RES[2:0]。

19.4.17 停止正在进行的转换（ADSTP、JADSTP）

软件决定是否停止转换，要停止正在进行的常规转换，应将 ADSTP 置 1；要停止正在进行的注入转换，应将 JADSTP 置 1。

停止转换将复位正在进行的 ADC 操作。随后可重新配置 ADC（例如：更改通道选择或触发），为新操作做好准备。

请注意，不能在常规转换仍在执行时停止注入转换，反之亦然。这样便可在常规转换仍在进行时重新配置注入转换序列和触发（反之亦然）。

如果 ADSTP 位由软件置 1，则会中止任何正在进行的常规转换，并会丢弃部分转换结果（ADC_DR 寄存器不会更新为当前转换结果）。

如果 JADSTP 位由软件置 1，则会中止任何正在进行的注入转换，并会丢弃部分转换结果（ADC_JDRy 寄存器不会更新为当前转换结果）。扫描序列也会中止并会复位（这意味着重启 ADC 将重新开始新的序列）。

该程序执行完毕后，ADSTP/ADSTART 位（常规转换时）或 ADSTP/JADSTART 位（注入转换时）会由硬件清零，软件必须轮询 ADSTART（或 JADSTART）直至其复位，然后才能判定 ADC 已完全停止运行。

注意： *在自动注入模式下 (JAUTO = 1)，将 ADSTP 位置 1 会中止常规转换和注入转换（不得使用 JADSTP）。*

图 84. 停止正在进行的常规转换

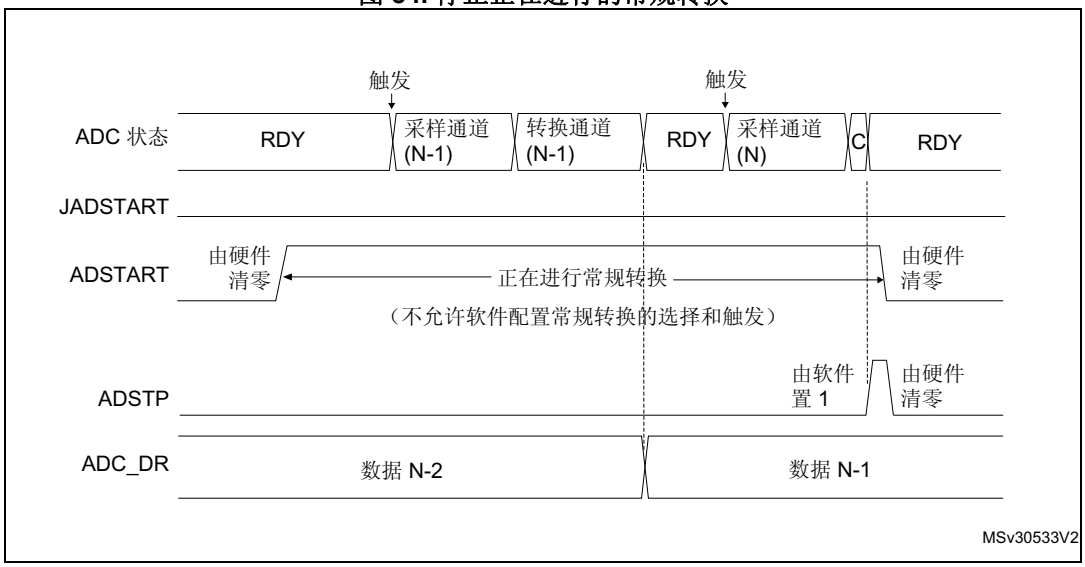
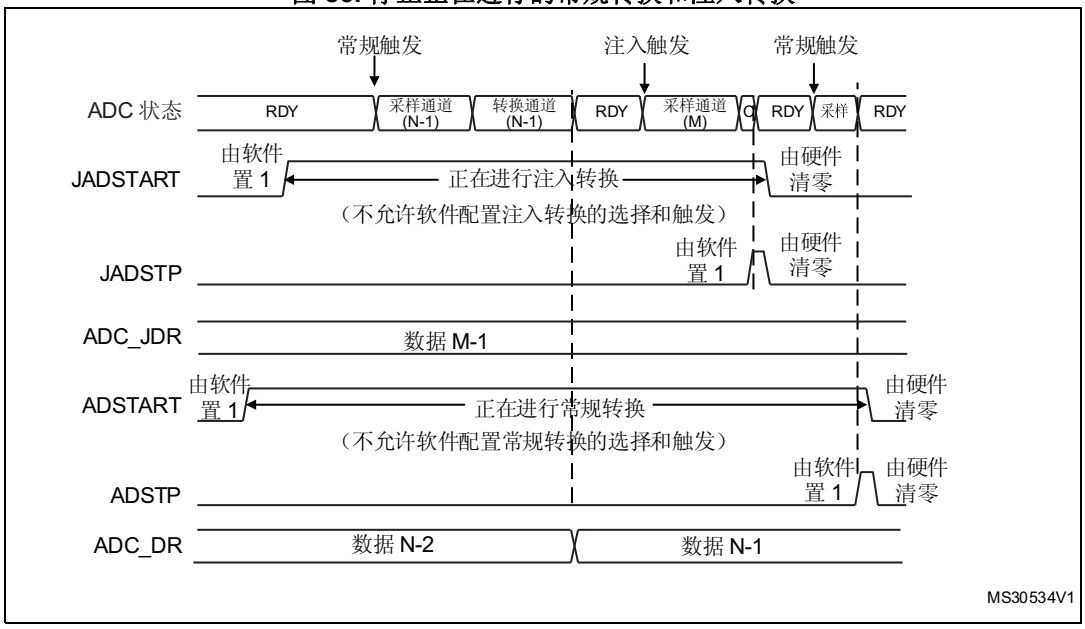


图 85. 停止正在进行的常规转换和注入转换



19.4.18 外部触发转换和触发极性（EXTSEL、EXTEN、JEXTSEL、JEXTEN）

可通过软件或外部事件（例如定时器捕获、输入引脚）触发转换或转换序列。如果 EXTEN[1:0] 控制位（对于常规转换）或 JEXTEN[1:0] 位（对于注入转换）不等于“0b00”，则外部事件能够以所选极性触发转换。

注入队列使能时（位 JQDIS = 0），不能实现注入软件触发。

软件将 ADSTART 位置 1 后，常规触发选择有效；软件将 JADSTART 位置 1 后，注入触发选择有效。

在转换进行时发生的硬件触发会被忽略。

- 如果位 ADSTART = 0，则会忽略发生的任何常规硬件触发。
- 如果位 JADSTART = 0，则会忽略发生的任何注入硬件触发。

表 110 提供了 EXTEN[1:0] 和 JEXTEN[1:0] 值与触发极性之间的对应关系。

表 110. 为常规外部触发配置触发极性

EXTEN[1:0]	源
00	禁止硬件触发检测，使能软件触发检测
01	在上升沿执行硬件触发检测
10	在下降沿执行硬件触发检测
11	在上升沿和下降沿均执行硬件触发检测

注意： *不能实时更改常规触发的极性。*

表 111. 为注入外部触发配置触发极性

JEXTEN[1:0]	源
00	– 如果 JQDIS = 1（队列禁止）：禁止硬件触发检测，使能软件触发检测 – 如果 JQDIS = 0（队列使能），同时禁止硬件和软件触发检测
01	在上升沿执行硬件触发检测
10	在下降沿执行硬件触发检测
11	在上升沿和下降沿均执行硬件触发检测

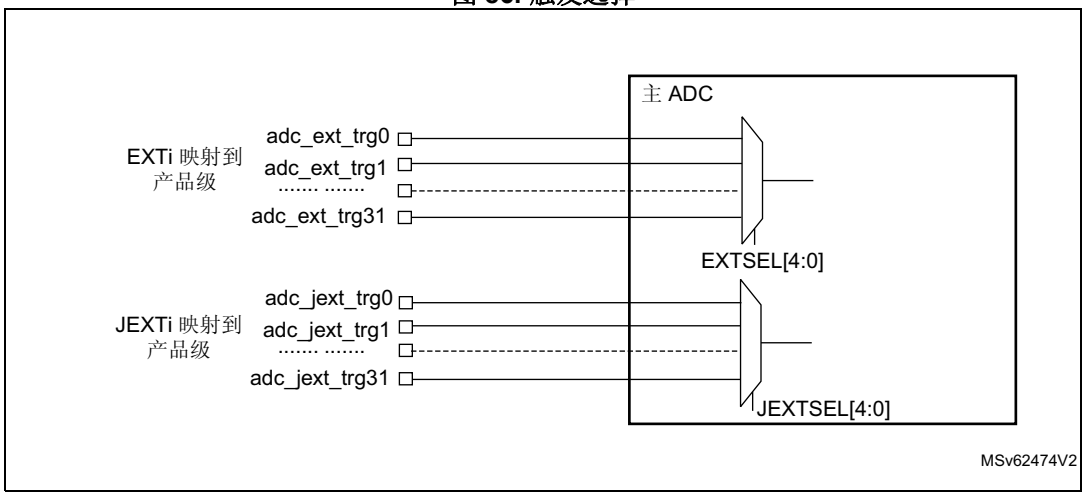
注意： *如果队列已使能 (JQDIS = 0)，可预计并实时更改注入触发的极性。请参见第 19.4.21 节：注入转换的上下文队列。*

EXTSEL 和 JEXTSEL 控制位用于从 32 个可能事件中选择可触发常规组转换和注入组转换的事件。

可通过注入触发中断常规组转换。

注意： *不能实时更改常规触发选择。可以预计并实时更改注入触发选择。请参见第 509 页的第 19.4.21 节：注入转换的上下文队列。*

图 86. 触发选择



有关所有可用于进行常规转换的外部触发，请参见*第 19.4.2 节：ADC 引脚和内部信号*中的 ADC 互连表。

19.4.19 注入通道管理

触发注入模式

要使用触发注入，必须将 ADC_CFGR 寄存器中的 JAUTO 位清零。

1. 通过外部触发或将 ADC_CR 寄存器中的 ADSTART 位置 1 来启动常规通道组转换。
2. 如果在常规通道组转换期间出现外部注入触发或者 ADC_CR 寄存器中的 JADSTART 位置 1，则当前的转换会复位，并会启动注入通道序列切换（所有注入通道都会转换一次）。
3. 然后，常规通道组的常规转换会从上次中断的常规转换处恢复。
4. 如果在注入转换期间出现常规事件，注入转换不会中断，但在注入序列结束时会执行常规序列。*图 87* 显示了相应的时序图。

注意： *使用触发注入时，必须确保触发事件之间的间隔长于注入序列。例如，如果序列长度为 30 个 ADC 时钟周期（即，采样时间为 2.5 个时钟周期的两次转换），则触发事件的最小间隔不能小于 31 个 ADC 时钟周期。*

自动注入模式

如果将 ADC_CFGR 寄存器中的 JAUTO 位置 1，则注入组中的通道会在常规组通道之后自动转换。这可用于转换最多由 20 个转换构成的序列，这些转换在 ADC_SQRy 和 ADC_JSQR 寄存器中编程。

在该模式下，必须将 ADC_CR 寄存器中的 ADSTART 位置 1 以开始常规转换，然后再进行注入转换（JADSTART 必须保持清零）。将 ADSTP 位置 1 会中止常规转换和注入转换（不得使用 JADSTP 位）。

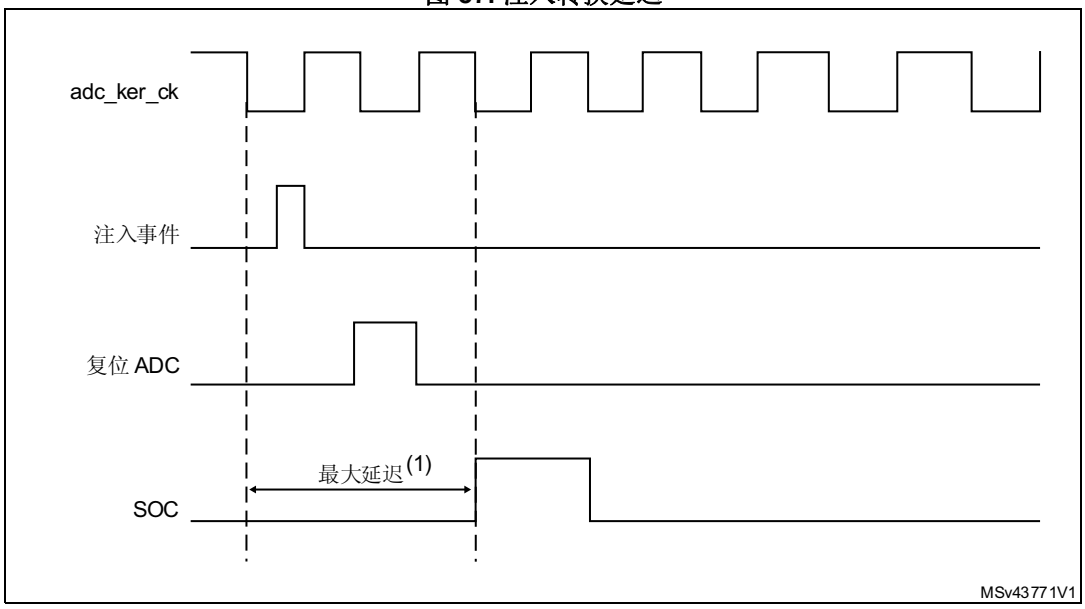
在此模式下，必须禁止注入通道上的外部触发。

如果 CONT 位和 JAUTO 位均已置 1，则在转换常规通道之后会继续转换注入通道。

注意： *不能同时使用自动注入和不连续采样模式。*

如果在 JAUTO 模式下使用 DMA 导出常规定序器的数据，需要将其设定为循环模式（将 DMA_CCRx 寄存器中的 CIRC 位置 1）。如果 CIRC 位复位（单发模式），JAUTO 序列将在出现 DMA 传输完成事件时停止。

图 87. 注入转换延迟



1. 有关延迟的最大值，请参见器件数据手册中的电气特性部分。

19.4.20 不连续模式（DISCEN、DISCNUM、JDISCEN）

常规组模式

可将 ADC_CFGR 寄存器中的 DISCEN 位置 1 来使能此模式。

该模式用于转换含有 n (n ≤ 8) 个转换的短序列（子组），该短序列是在 ADC_SQRy 寄存器中选择的转换序列的一部分。可通过写入 ADC_CFGR 寄存器中的 DISCNUM[2:0] 位来指定 n 的值。

出现外部触发时，将启动在 ADC_SQRy 寄存器中选择的接下来 n 个转换，直到序列中的所有转换均完成为止。通过 ADC_SQR1 寄存器中的 L[3:0] 位定义总序列长度。

示例：

- DISCEN = 1，n = 3，要转换的通道 = 1、2、3、6、7、8、9、10、11
  - 第一次触发：转换的通道为 1、2、3（每次转换时都生成 EOC 事件）。
  - 第二次触发：转换的通道为 6、7、8（每次转换时都生成 EOC 事件）。
  - 第三次触发：转换的通道为 9、10、11（每次转换时都生成 EOC 事件），并会在通道 11 转换完成后生成 EOS 事件。
  - 第四次触发：转换的通道为 1、2、3（每次转换时都生成 EOC 事件）。
  - ...
- DISCEN = 0，要转换的通道 = 1、2、3、6、7、8、9、10、11
  - 第一次触发：转换整个序列：通道 1、然后是通道 2、3、6、7、8、9、10 和 11。每次转换都会生成 EOC 事件，最后一次转换还会生成 EOS 事件。
  - 所有后续触发事件都将重启整个序列。

注意： *上例中提及的通道数并非在所有微控制器上均能提供。*

在不连续模式下转换常规组时，不会出现翻转（序列最后一个子组的转换次数少于 n 次）。

转换完所有子组后，下一个触发信号将启动第一个子组的转换。在上述示例中，第四次触发重新转换了第一个子组中的通道 1、2 和 3。

不能同时使能不连续模式和连续模式。如果同时使能两种模式（即 DISCEN = 1、CONT = 1），ADC 会认定连续模式已禁止并继续执行相关操作。

注入组模式

可将 ADC_CFGR 寄存器中的 JDISCEN 位置 1 来使能此模式。在出现外部注入触发事件之后，该模式会逐通道转换在 ADC_JSQR 寄存器中选择的序列，相当于不连续模式下常规通道“n”固定为 1 的情况。

出现外部触发时，将启动在 ADC_JSQR 寄存器中选择的下一个通道转换，直到序列中的所有转换均完成为止。通过 ADC_JSQR 寄存器中的 JL[1:0] 位定义总序列长度。

示例：

- JDISCEN = 1，要转换的通道 = 1、2、3
  - 第一次触发：转换通道 1（生成 JEOC 事件）
  - 第二次触发：转换通道 2（生成 JEOC 事件）
  - 第三次触发：转换通道 3 并生成 JEOC 事件和 JEOS 事件
  - ...

注意： *上例中提及的通道数并非在所有微控制器上均能提供。*

转换完所有注入通道后，下一个触发信号将启动第一个注入通道的转换。在上述示例中，第四次触发重新转换了第一个注入通道 1。

不能同时使用自动注入模式和不连续模式：当 JAUTO 置 1 时，DISCEN 和 JDISCEN 位必须通过软件保持清零状态。

19.4.21 注入转换的上下文队列

实现上下文队列可为下一个注入转换序列准备多达 2 个上下文。必须将 ADC_CFGR 寄存器的 JQDIS 位复位才能使能此功能。上下文队列使能时，只能进行硬件触发转换。

该上下文包括：

- 注入触发的配置（ADC_JSQR 寄存器中的 JEXTEN[1:0] 位和 JEXTSEL 位）
- 注入序列的定义（ADC_JSQR 寄存器中的 JSQx[4:0] 位和 JL[1:0] 位）

上下文的所有参数都会在 ADC_JSQR 这一寄存器中定义，该寄存器会实现一个双缓冲区队列，可缓冲多达 2 组参数：

- JSQR 寄存器可随时写入，正在进行注入转换时也不例外。
- 每个写入到 JSQR 寄存器中的数据均会存储在上下文队列中。
- 队列开始时为空，对 JSQR 寄存器进行的第一次写访问时会立即更改上下文，随即 ADC 会准备好接收注入触发。
- 注入序列完成后，队列会被占用，上下文会根据队列中存储的后续 JSQR 参数进行更改。这一新的上下文会用于下一个注入转换序列。

- 如果在队列已满的情况下向 JSQR 寄存器执行写操作，会发生队列溢出。这种溢出情况会通过 JQOVF 标志置为有效来指示。发生溢出时，会忽略造成溢出的 JSQR 寄存器写访问，并且上下文队列保持不变。如果 JQOVFIE 位置 1，可产生中断。
- 队列变空时可能执行两种操作，具体取决于寄存器 ADC_CFGR 的控制位 JQM 的值：
  - 如果 JQM = 0，队列刚好在使能 ADC 后为空，但在随后的运行操作期间绝不会变空：队列始终保留上一个有效的上下文，并会根据上一个有效的上下文处理后续的有效注入序列启动。
  - 如果 JQM = 1，队列会在注入序列结束后或队列被清空时变空。这种情况下，队列中不存在任何上下文，硬件触发也会被禁止。因此会忽略后续的所有硬件注入触发，直至软件重新向 JSQR 寄存器写入新的注入上下文。
- 读取 JSQR 寄存器会返回当前有效的 JSQR 上下文。如果 JSQR 上下文为空，JSQR 的读出值为 0x0000。
- 如果通过将 JADSTP 置 1 的方式停止注入转换、或者通过将 ADDIS 置 1 的方式禁止 ADC，队列会被清空：
  - 如果 JQM = 0，队列会保留上一个有效的上下文。
  - 如果 JQM = 1，队列会变空，并会忽略触发。

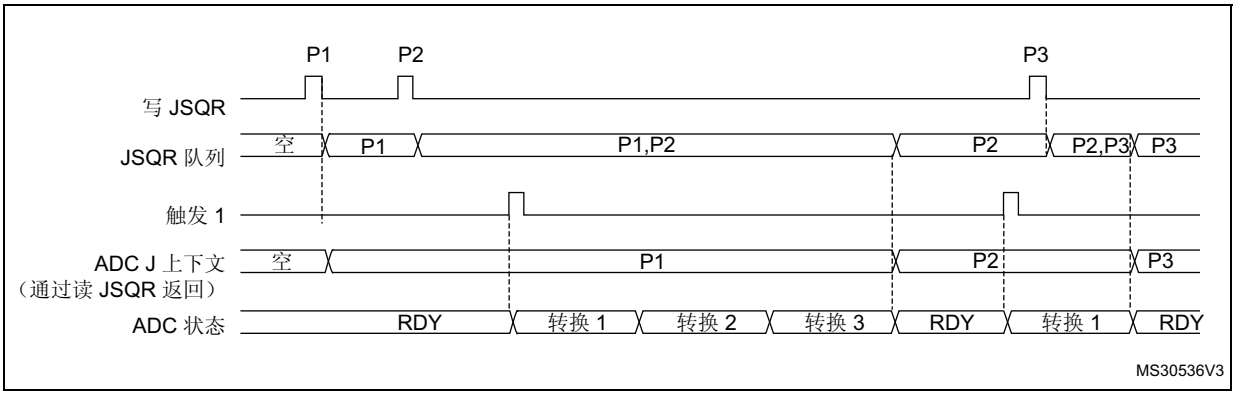
注意： *如果配置为不连续模式（位 JDISCEN = 1），只有注入序列的最后一次触发会更改上下文并占用队列。第一次触发仅会占用队列，但其他触发仍为有效触发，如下文中的不连续模式示例所示（两个上下文的长度均为 3）：*

- *第一次触发，不连续。序列 1：上下文 1 已占用，已执行第一次转换*
- *第二次触发，不连续。序列 1：第二次转换。*
- *第三次触发，不连续。序列 1：第三次转换。*
- *第四次触发，不连续。序列 2：上下文 2 已占用，已执行第一次转换。*
- *第五次触发，不连续。序列 2：第二次转换。*
- *第六次触发，不连续。序列 2：第三次转换。*

更改触发或序列上下文时的操作

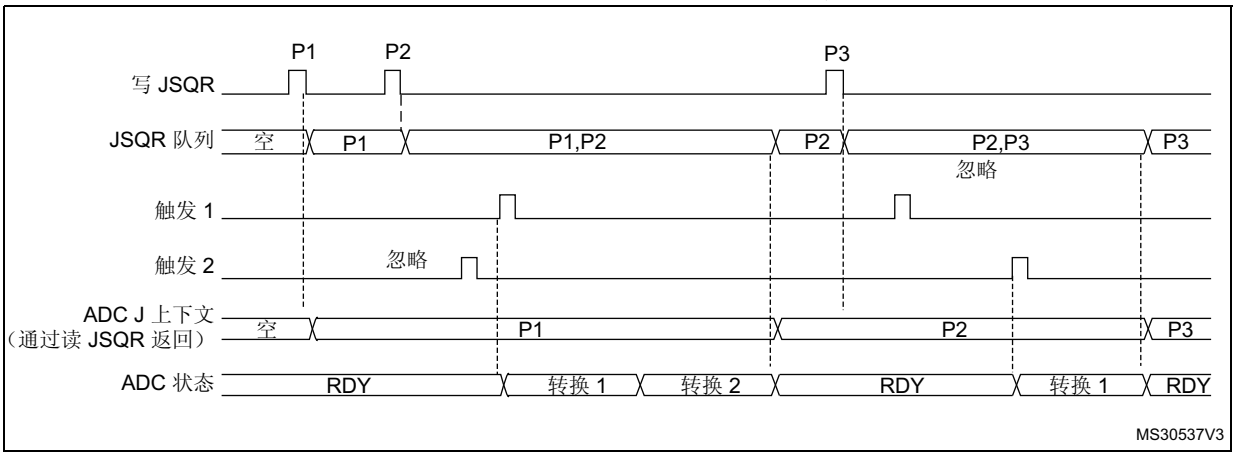
图 88 和*图 89* 显示了更改序列或触发时上下文队列的操作。

图 88. JSQR 上下文队列示例（队列更改）



1. 参数：
   P1：3 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 1
   P3：4 个转换组成的序列，硬件触发 1

图 89. JSQR 上下文队列示例（触发更改）

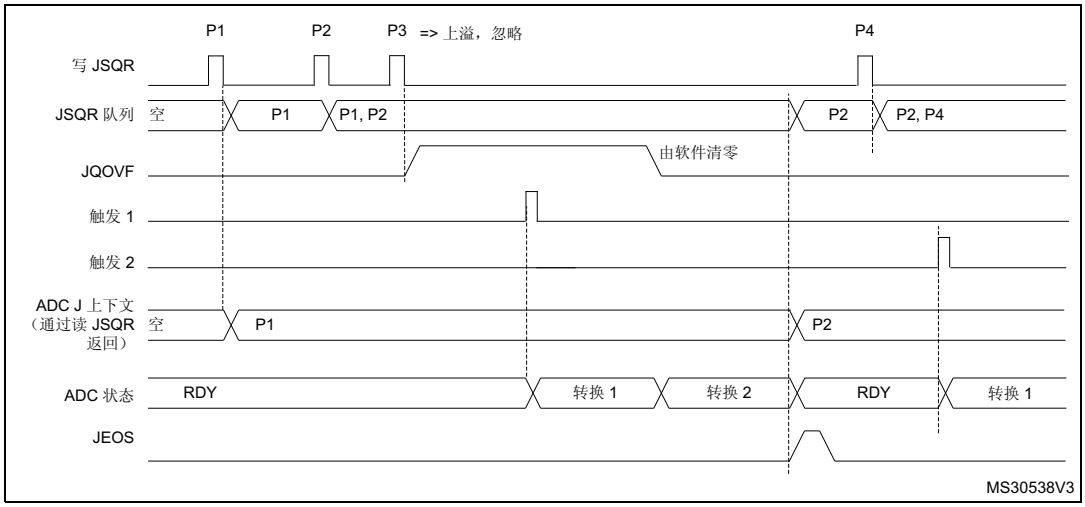


1. 参数：
   P1：2 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 2
   P3：4 个转换组成的序列，硬件触发 1

上下文队列：发生队列溢出时的操作

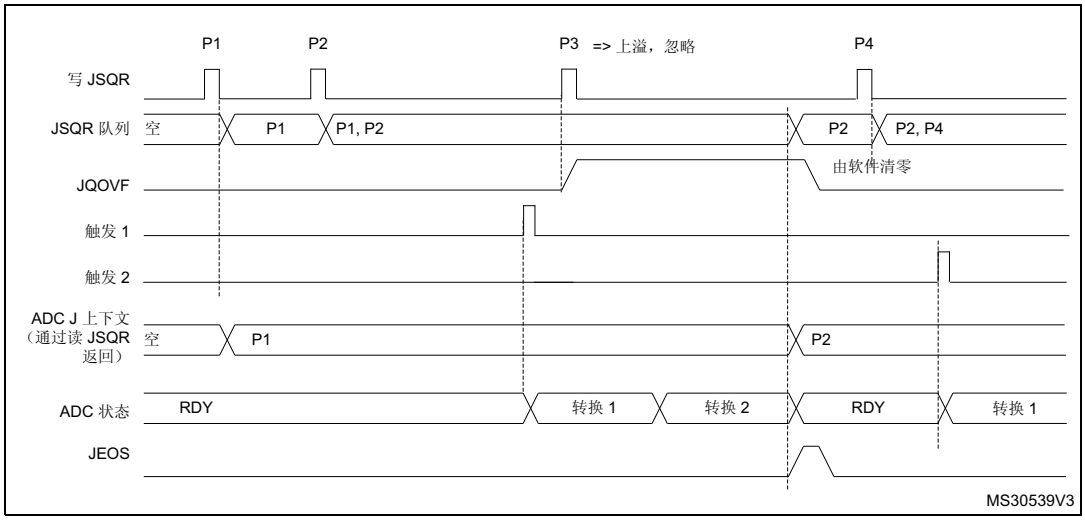
图 90 和*图 91* 显示了转换之前或之后发生溢出时上下文队列的操作。

图 90. 转换前发生溢出的 JSQR 上下文队列示例



1. 参数：
   P1：2 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 2
   P3：3 个转换组成的序列，硬件触发 1
   P4：4 个转换组成的序列，硬件触发 1

图 91. 转换期间发生溢出的 JSQR 上下文队列示例



1. 参数：
   P1：2 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 2
   P3：3 个转换组成的序列，硬件触发 1
   P4：4 个转换组成的序列，硬件触发 1

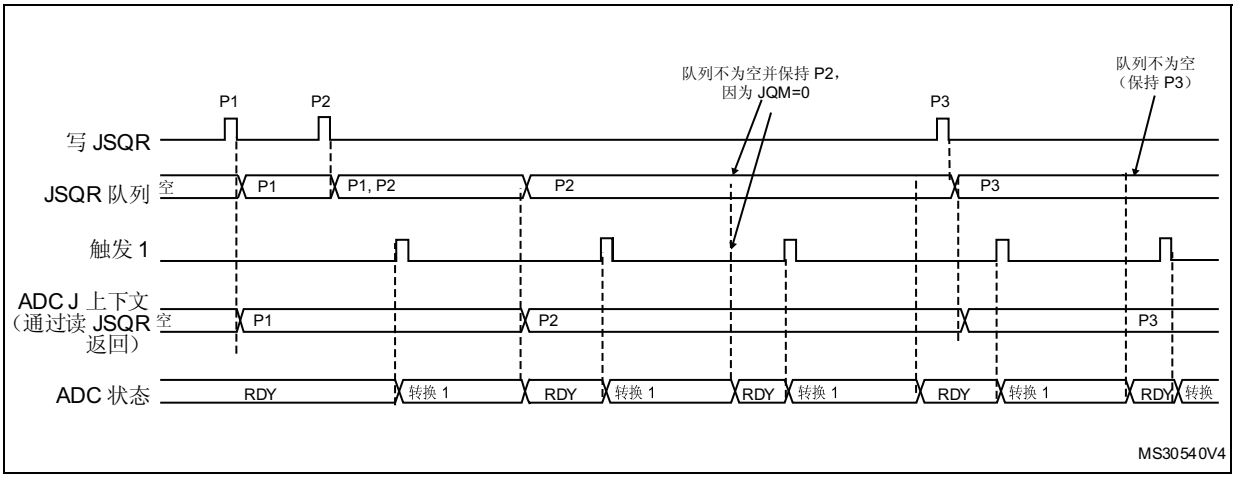
建议按照下述方法管理队列溢出：

- 将每个 P 上下文写入 JSQR 寄存器后，通过标志 JQOVF 指示写操作是否已被忽略（可生成中断）。
- 为避免队列溢出，仅在上一个上下文 P2 的 JEOS 标志置 1 后写入第三个上下文 (P3)。这样可确保上一个上下文已被占用且队列未满。

上下文队列：队列变空时的操作

图 92 和*图 93* 显示了在 JQM = 0 或 1 这两种情况下队列变空时上下文队列的操作。

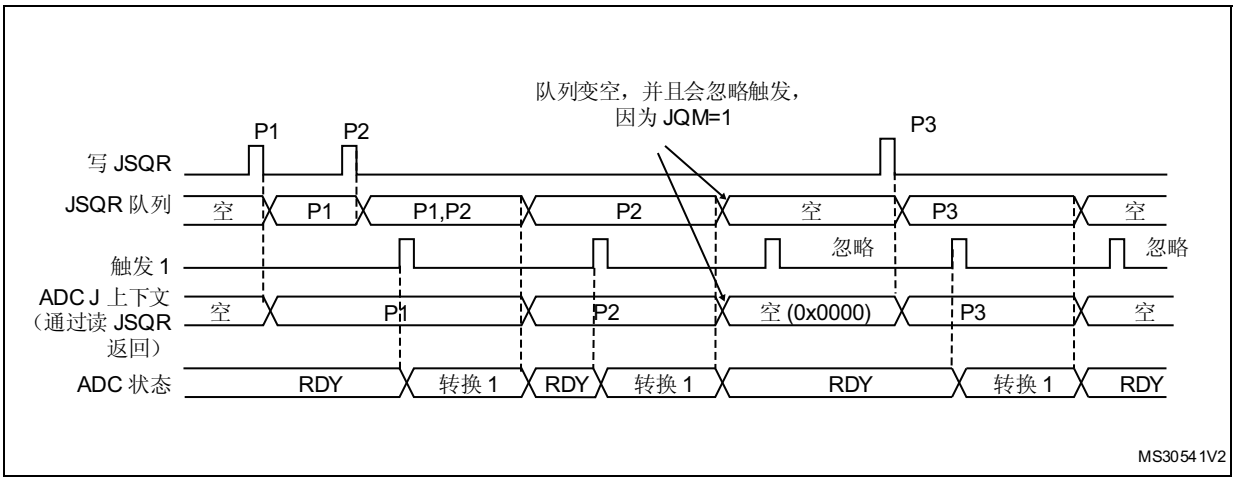
图 92. 队列为空时的 JSQR 上下文队列示例（JQM = 0 的情况）



1. 参数：
   P1：1 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 1
   P3：1 个转换组成的序列，硬件触发 1

注意： *写入 P3 时，上下文会立即改变。但由于内部重新同步的原因，会存在一定的延迟，如果刚好在写入 P3 之后或之前发生触发，可能会发生启动的转换被视为上下文 P2 的情况。为了避免出现这种情况，用户必须确保写入立即应用的新上下文时没有发生 ADC 触发。*

图 93. 队列为空时的 JSQR 上下文队列示例（JQM = 1 的情况）

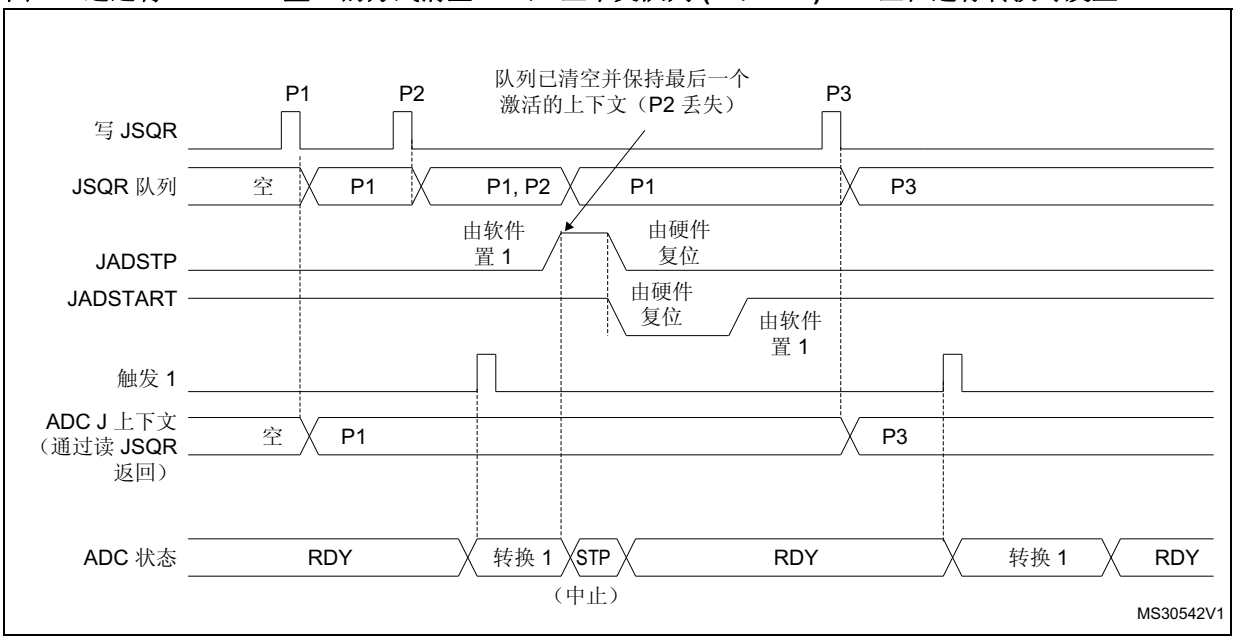


1. 参数：
   P1：1 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 1
   P3：1 个转换组成的序列，硬件触发 1

清空上下文队列

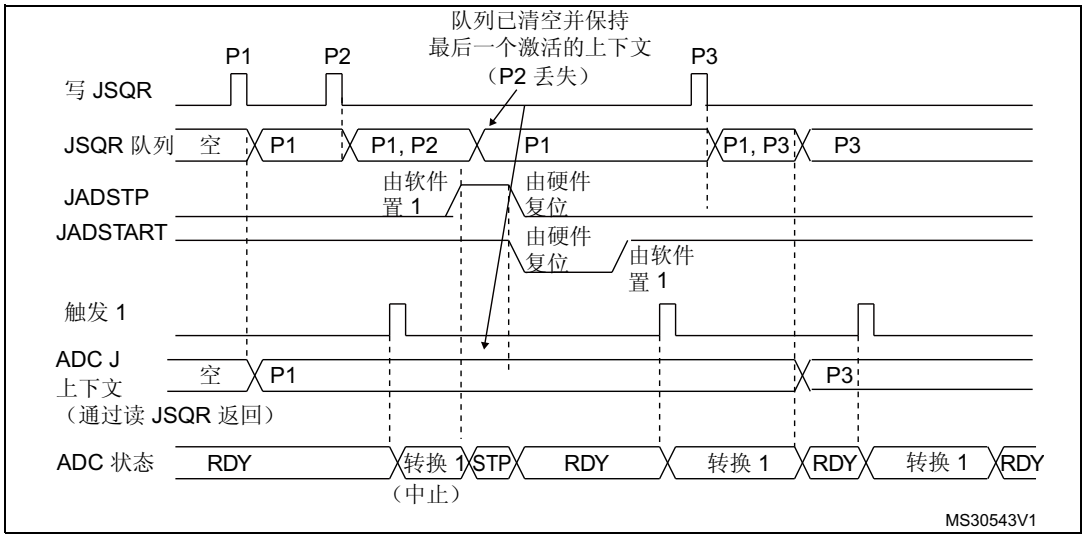
下图显示的是各种情况下清空队列时上下文队列的操作。

图 94. 通过将 JADSTP 置 1 的方式清空 JSQR 上下文队列 (JQM = 0)——正在进行转换时发生 JADSTP



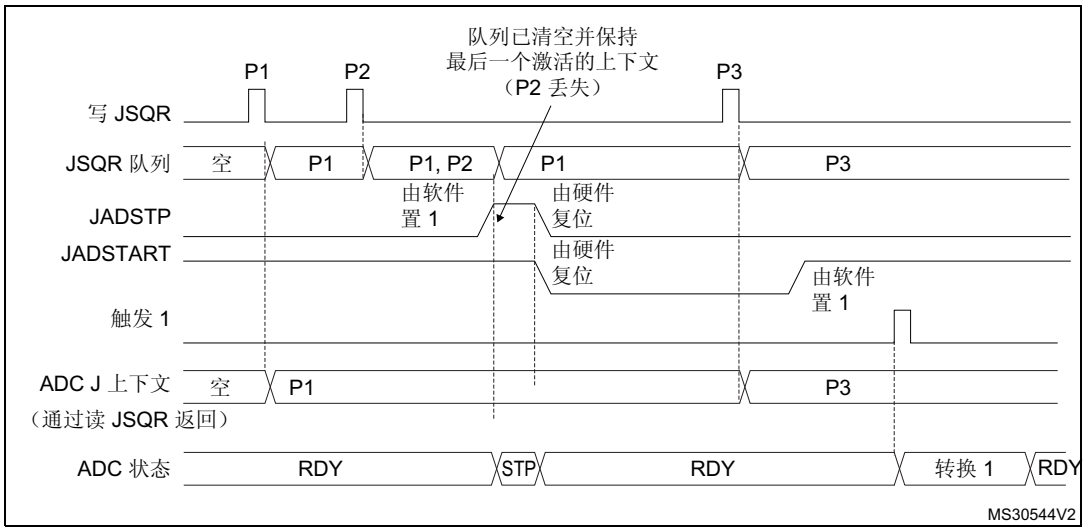
1. 参数：
   P1：1 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 1
   P3：1 个转换组成的序列，硬件触发 1

图 95. 通过将 JADSTP 置 1 的方式清空 JSQR 上下文队列 (JQM = 0)—— 正在进行转换时发生 JADSTP 并出现新触发



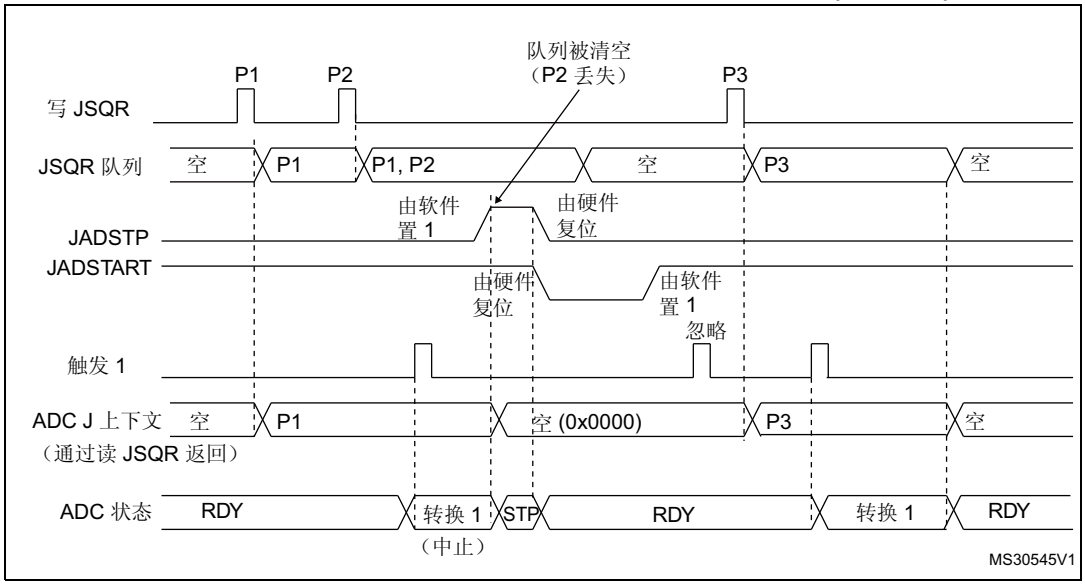
1. 参数：
   P1：1 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 1
   P3：1 个转换组成的序列，硬件触发 1

图 96. 通过将 JADSTP 置 1 的方式清空 JSQR 上下文队列 (JQM = 0)—— 在正在进行转换的范围之外发生 JADSTP



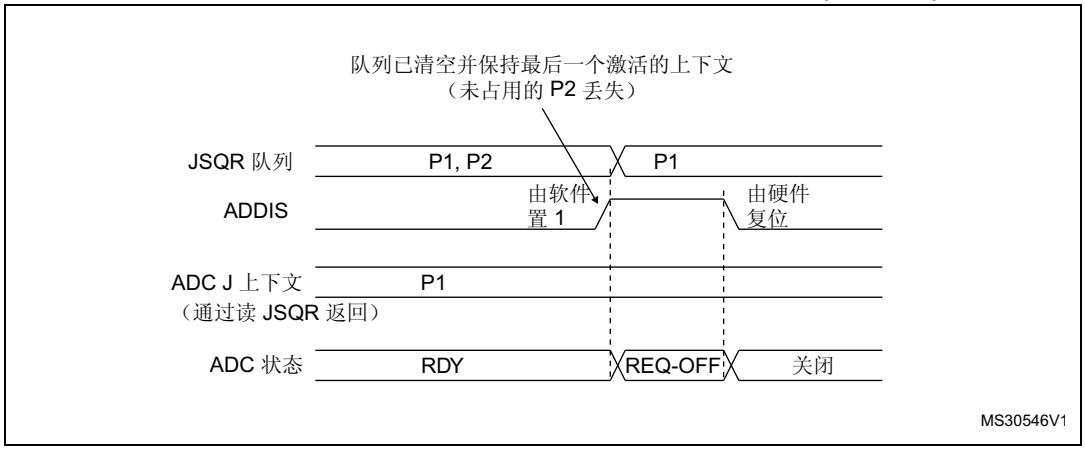
1. 参数：
   P1：1 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 1
   P3：1 个转换组成的序列，硬件触发 1

图 97. 通过将 JADSTP 置 1 的方式清空 JSQR 上下文队列 (JQM = 1)



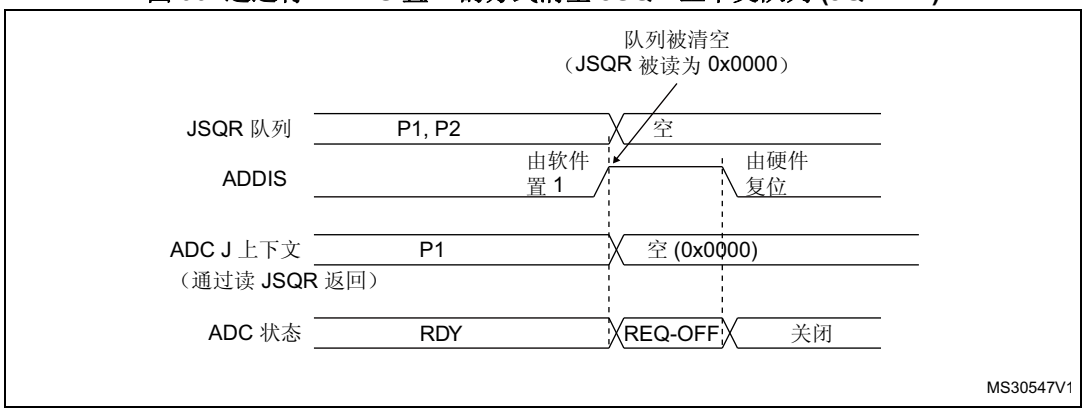
1. 参数：
   P1：1 个转换组成的序列，硬件触发 1
   P2：1 个转换组成的序列，硬件触发 1
   P3：1 个转换组成的序列，硬件触发 1

图 98. 通过将 ADDIS 置 1 的方式清空 JSQR 上下文队列 (JQM = 0)



1. 参数：
P1：1 个转换组成的序列，硬件触发 1
P2：1 个转换组成的序列，硬件触发 1
P3：1 个转换组成的序列，硬件触发 1

图 99. 通过将 ADDIS 置 1 的方式清空 JSQR 上下文队列 (JQM = 1)



1. 参数：
P1：1 个转换组成的序列，硬件触发 1
P2：1 个转换组成的序列，硬件触发 1
P3：1 个转换组成的序列，硬件触发 1

上下文队列：在队列为空时启动 ADC

要在队列为空时启动 ADC 操作，必须按照以下程序进行操作，以免在 ADC 初始化时无法获悉第一个上下文。此程序仅在 JQM 位复位的情况下适用：

5. 写入空 JSQR，JEXTEN 不等于 0（否则会触发软件转换）
6. 将 JADSTART 置 1
7. 将 JADSTP 置 1
8. 等待，直至 JADSTART 复位
9. 将 JADSTART 置 1。

禁止队列

可通过将 ADC_CFGR 寄存器中的 JQDIS 位置 1 的方式禁止队列。

19.4.22 可编程分辨率 (RES)——快速转换模式

可通过降低 ADC 分辨率来执行快速转换。

通过对控制位 RES[1:0] 进行编程，可将分辨率配置为 12 位、10 位、8 位或 6 位。*图 104*、*图 105*、*图 106* 和*图 107* 显示了转换结果格式与分辨率和数据对齐方式的对应关系。

对于不要求使用高数据精度的应用，分辨率越低，转换时间越短，降低分辨率可缩短逐次逼近步骤所需的转换时间，如*表 112* 所示。

表 112. T_{SAR} 与分辨率的对应关系

RES（位）	T_{SAR}（ADC 时钟周期）	F_{ADC} = 30 MHz 时的 T_{SAR} (ns)	T_{CONV}（ADC 时钟周期）（采样时间 = 2.5 个 ADC 时钟周期）	F_{ADC} = 30 MHz 时的 T_{CONV} (ns)
12	12.5 个 ADC 时钟周期	416.67 ns	15 个 ADC 时钟周期	500.0 ns
10	10.5 个 ADC 时钟周期	350.0 ns	13 个 ADC 时钟周期	433.33 ns
8	8.5 个 ADC 时钟周期	203.33 ns	11 个 ADC 时钟周期	366.67 ns
6	6.5 个 ADC 时钟周期	216.67 ns	9 个 ADC 时钟周期	300.0 ns

19.4.23 转换结束、采样阶段结束（EOC、JEOC、EOSMP）

每次出现常规转换结束 (EOC) 事件和注入转换 (JEOC) 事件时，ADC 都会通知应用。

新的常规转换数据出现在 ADC_DR 寄存器中后，ADC 会立即将 EOC 标志置 1。如果 EOCIE 位置 1，可产生中断。EOC 标志可通过由软件向其写入 1 或读取 ADC_DR 的方式来清零。

新的注入转换数据出现在 ADC_JDRy 寄存器中后，ADC 会立即将 JEOC 标志置 1。如果 JEOCIE 位置 1，可产生中断。JEOC 标志可通过由软件向其写入 1 或读取相应 ADC_JDRy 寄存器的方式来清零。

ADC 还通过将状态位 EOSMP 置 1 来指示采样阶段结束（仅限常规转换）。EOSMP 标志可通过由软件向其写入 1 的方式来清零。如果 EOSMPIE 位置 1，可产生中断。

19.4.24 转换序列结束（EOS、JEOS）

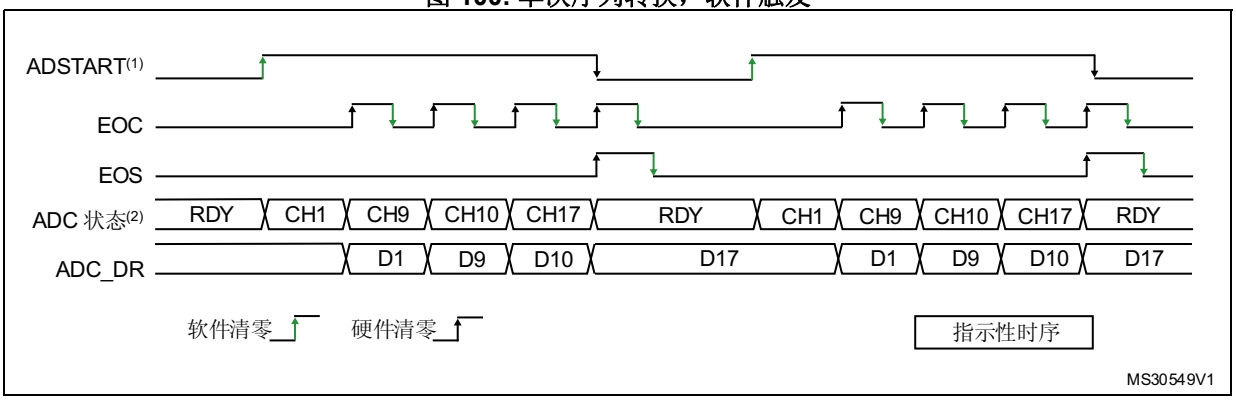
每次出现常规序列结束 (EOS) 事件和注入序列结束 (JEOS) 事件时，ADC 都会通知应用。

常规转换序列的最后一个数据出现在 ADC_DR 寄存器中时，ADC 会立即将 EOS 标志置 1。如果 EOSIE 位置 1，可产生中断。EOS 标志可通过由软件向其写入 1 的方式来清零。

注入转换序列的最后一数据完成时，ADC 会立即将 JEOS 标志置 1。如果 JEOSIE 位置 1，可产生中断。JEOS 标志可通过由软件向其写入 1 的方式来清零。

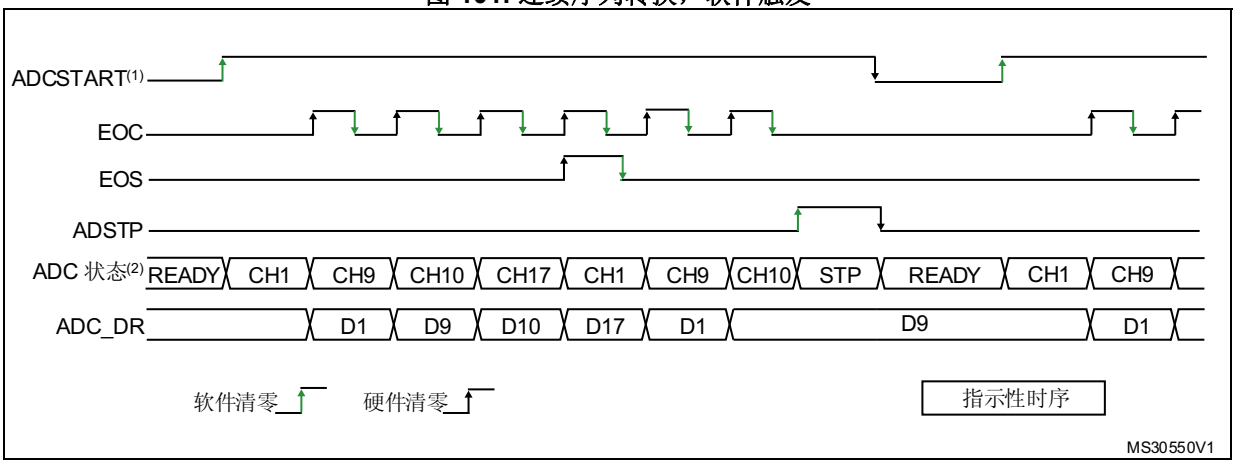
19.4.25 时序图示例（单次模式/连续模式，硬件/软件触发）

图 100. 单次序列转换，软件触发



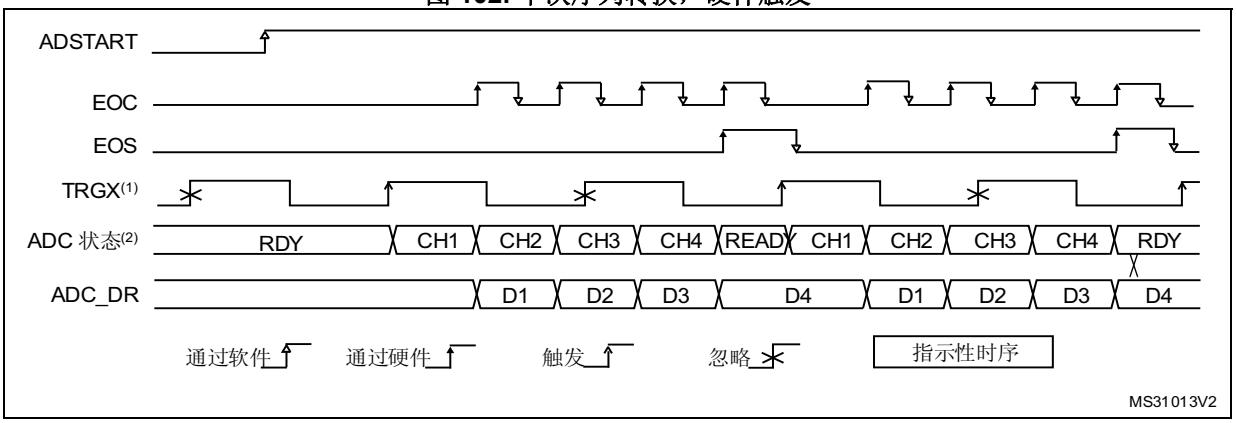
1. EXTEN = 0x0，CONT = 0
2. 所选通道 = 1、9、10、17；AUTDLY = 0。

图 101. 连续序列转换，软件触发



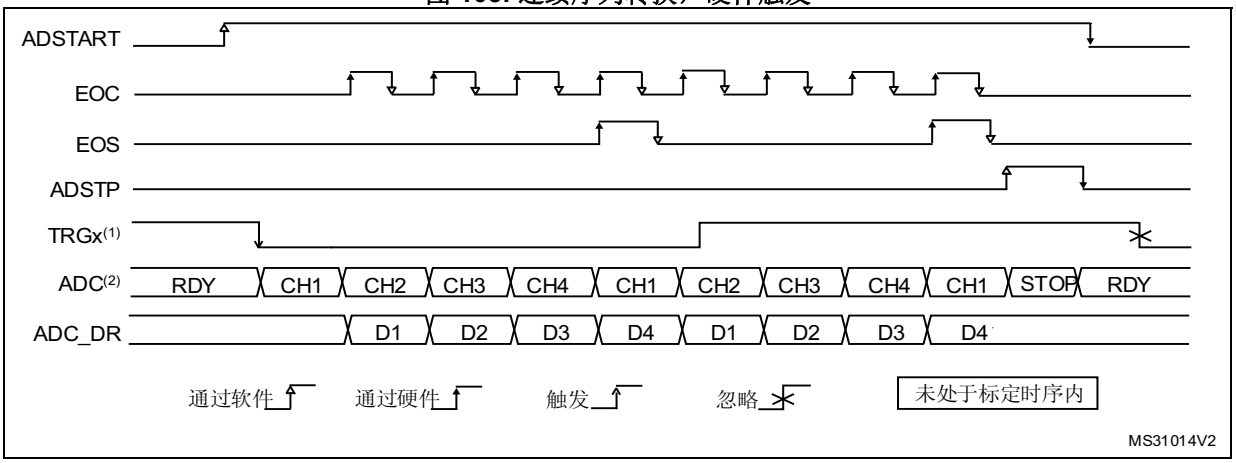
1. EXTEN = 0x0，CONT = 1
2. 所选通道 = 1、9、10、17；AUTDLY = 0。

图 102. 单次序列转换，硬件触发



1. 选择 TRGx（过频）作为触发源，EXTEN = 01，CONT = 0
2. 所选通道 = 1、2、3、4；AUTDLY = 0。

图 103. 连续序列转换，硬件触发



1. 选择 TRGx 作为触发源，EXTEN = 10，CONT = 1
2. 所选通道 = 1、2、3、4；AUTDLY = 0。

19.4.26 数据管理

数据寄存器、数据对齐和偏移（ADC_DR、OFFSET、OFFSET_CH、ALIGN）

数据和对齐

每次常规转换通道结束时（发生 EOC 事件时），转换后数据的结果都会存储在宽度为 16 位的 ADC_DR 数据寄存器中。

每次注入转换通道结束时（发生 JEOC 事件时），转换后数据的结果都会存储在宽度为 16 位的相应 ADC_JDRy 数据寄存器中。

ADC_CFGR 寄存器中的 ALIGN 位用于选择转换后存储的数据的对齐方式。可选择左对齐和右对齐两种方式，如*图 104*、*图 105*、*图 106* 和*图 107* 所示。

特例：采用左对齐时，数据基于半字对齐，分辨率设置为 6 位时除外。当分辨率设置为 6 位时，数据基于字节对齐，如*图 106* 和*图 107* 所示。

注意： *过采样模式下不支持左对齐。当 ROVSE 和/或 JOVSE 位置 1 时，忽略 ALIGN 位的值，并且 ADC 仅提供右对齐数据。*

偏移

可以通过将 ADC_OFRy 寄存器中的 OFFSET_EN 置 1，对通道应用偏移 y（y = 1、2、3、4）。应用偏移的通道会编程到 ADC_OFRy 寄存器的 OFFSET_CH[4:0] 位中。这种情况下，转换后的值会减去写入到 OFFSET[11:0] 位中的用户自定义偏移。结果可能是负值，因此读取的数据为有符号值，SEXT 位代表扩展符号值。

注意： *过采样模式下不支持偏移校正。如果 ROVSE 和/或 JOVSE 位置 1，ADC_OFRy 寄存器中 OFFSET_EN 位的值会被忽略（视为复位）。*

表 115 介绍了如何对模拟看门狗 1 的所有可能的分辨率进行比较。

表 113. 偏移计算与数据分辨率

分辨率（位 RES[1:0]）	原始转换数据与偏移相减		结果	备注
	原始转换数据，左对齐	偏移		
00：12 位	DATA[11:0]	OFFSET[11:0]	有符号 12 位数据	-
01：10 位	DATA[11:2]，00	OFFSET[11:0]	有符号 10 位数据	用户必须将 OFFSET[1:0] 配置为“00”
10：8 位	DATA[11:4]，0000	OFFSET[11:0]	有符号 8 位数据	用户必须将 OFFSET[3:0] 配置为“0000”
11：6 位	DATA[11:6]，000000	OFFSET[11:0]	有符号 6 位数据	用户必须将 OFFSET[5:0] 配置为“000000”

从对应于通道“i”的 ADC_DR（常规通道）或 ADC_JDRy（注入通道，y = 1、2、3、4）读取数据时：

- 如果相应通道的其中一个偏移已使能（位 OFFSET_EN = 1），则读取的数据为有符号数据。
- 如果此通道的四个偏移均未使能，则读取的数据为无符号数据。

图 104、*图 105*、*图 106* 和*图 107* 中显示了有符号数据和无符号数据的对齐方式。

图 104. 右对齐（偏移禁止，无符号值）

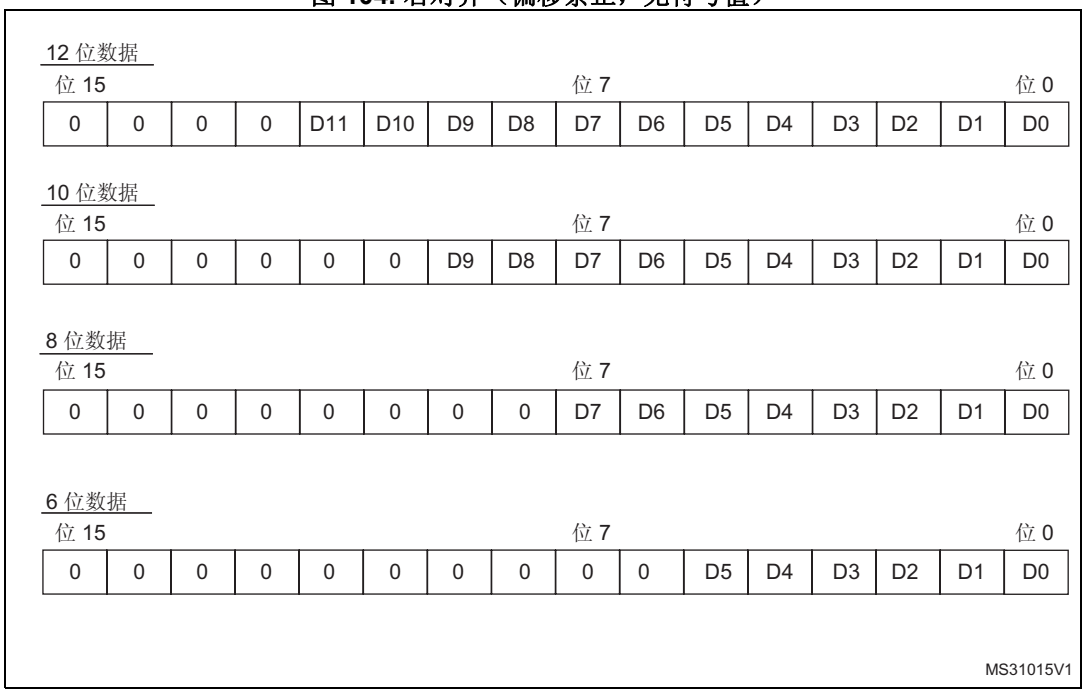


图 105. 右对齐（偏移使能，有符号值）

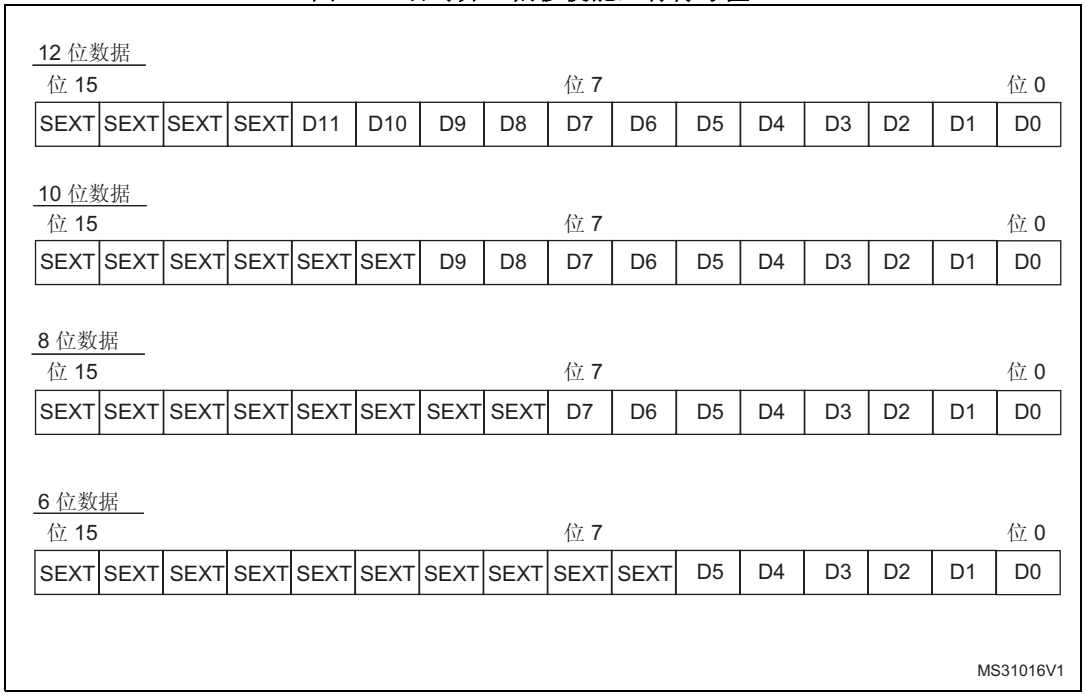


图 106. 左对齐（偏移禁止，无符号值）

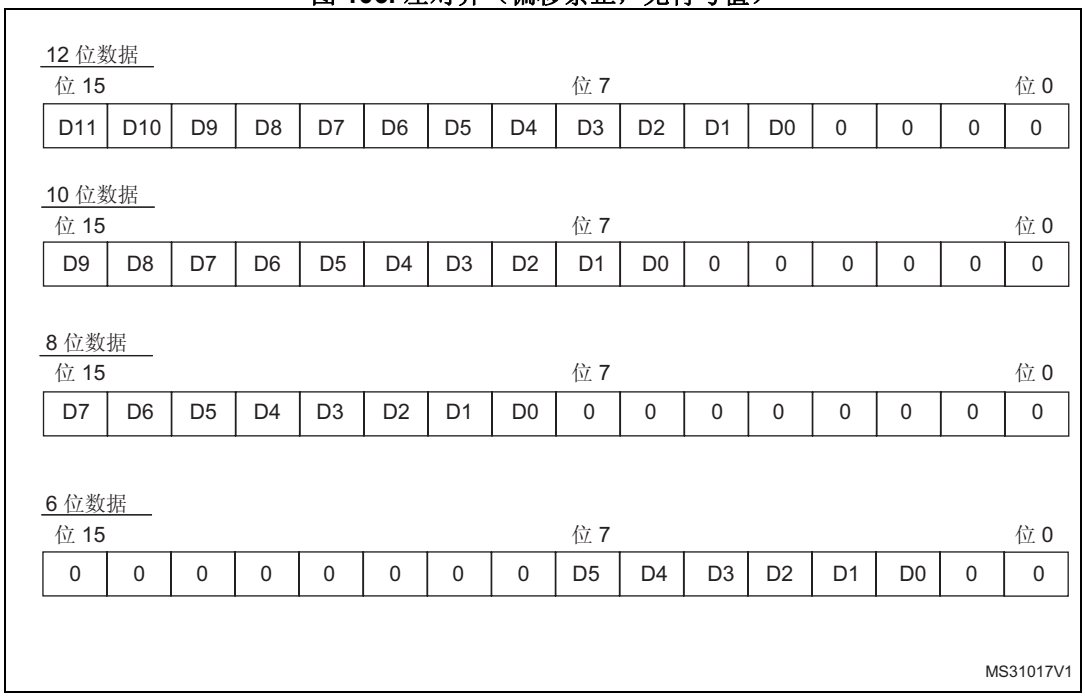
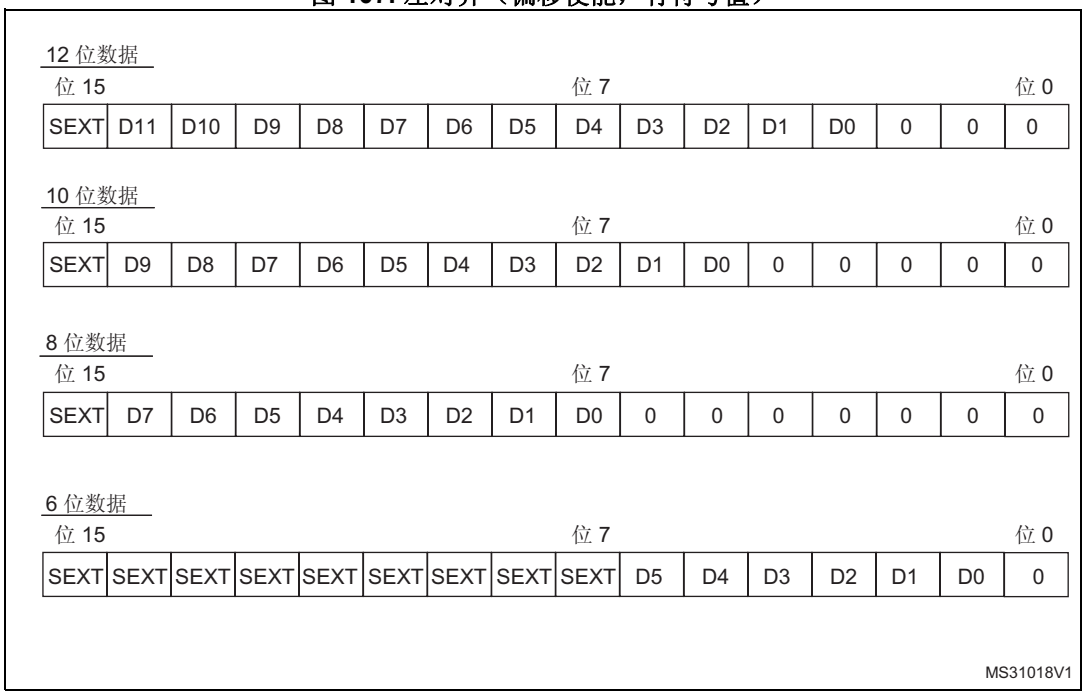


图 107. 左对齐（偏移使能，有符号值）



偏移补偿

如果偏移操作期间 ADC_OFRy 寄存器中的 SATEN 位置 1，则数据为无符号数据。所有偏移数据在 0x000 饱和（12 位模式下）。OFFSETPOS 位置 1 时，偏移方向为正向，数据在 0xFFF 饱和（12 位模式下）。在 8 位模式下，数据分别在 0x00 和 0xFF 饱和。

模拟看门狗比较在偏移补偿之前执行。

ADC 溢出（OVR、OVRMOD）

如果常规转换后的数据未在 ADC_DR FIFO（三个阶段）溢出之前（由 CPU 或 DMA）读取，会由溢出标志 (OVR) 进行指示。

如果新转换在 ADC_CR 寄存器 FIFO 已满的情况下完成，则 OVR 标志置 1。如果 OVRIE 位置 1，则会生成中断。

如果发生溢出情况，ADC 仍会保持工作状态并可继续进行转换，除非通过软件将 ADSTP 置 1，从而停止并复位序列。由于 ADC_DR FIFO 具有三个阶段，因此 FIFO 中最多存储三个数据。

OVR 标志可通过由软件向其写入 1 的方式来清零。

可对控制位 OVRMOD 进行编程，从而配置发生溢出事件时是要保留数据还是要覆盖数据：

- OVRMOD = 0：溢出事件会保留数据寄存器的数据，防止其被覆盖：会保留最多达到 ADC_DR FIFO 深度（三个阶段）的原数据，并会丢弃新的转换结果。在此模式下，将禁止 ADC_DR FIFO。如果 FIFO 已满，尽管可以继续执行转换，但生成的数据还是会被丢弃。EOC 通过读取 ADC_DR 寄存器的方式清零。但是，FIFO 仍可包含之前转换的数据。
- OVRMOD = 1：数据寄存器会由上一次转换结果覆盖，之前未读取的数据会丢失。在此模式下，将禁止 ADC_DR FIFO。如果 OVR 保持为 1，可继续正常执行转换，并且 ADC_DR 寄存器将始终包含最新的转换数据。

图 108. 溢出示例 (OVRMOD = 0)

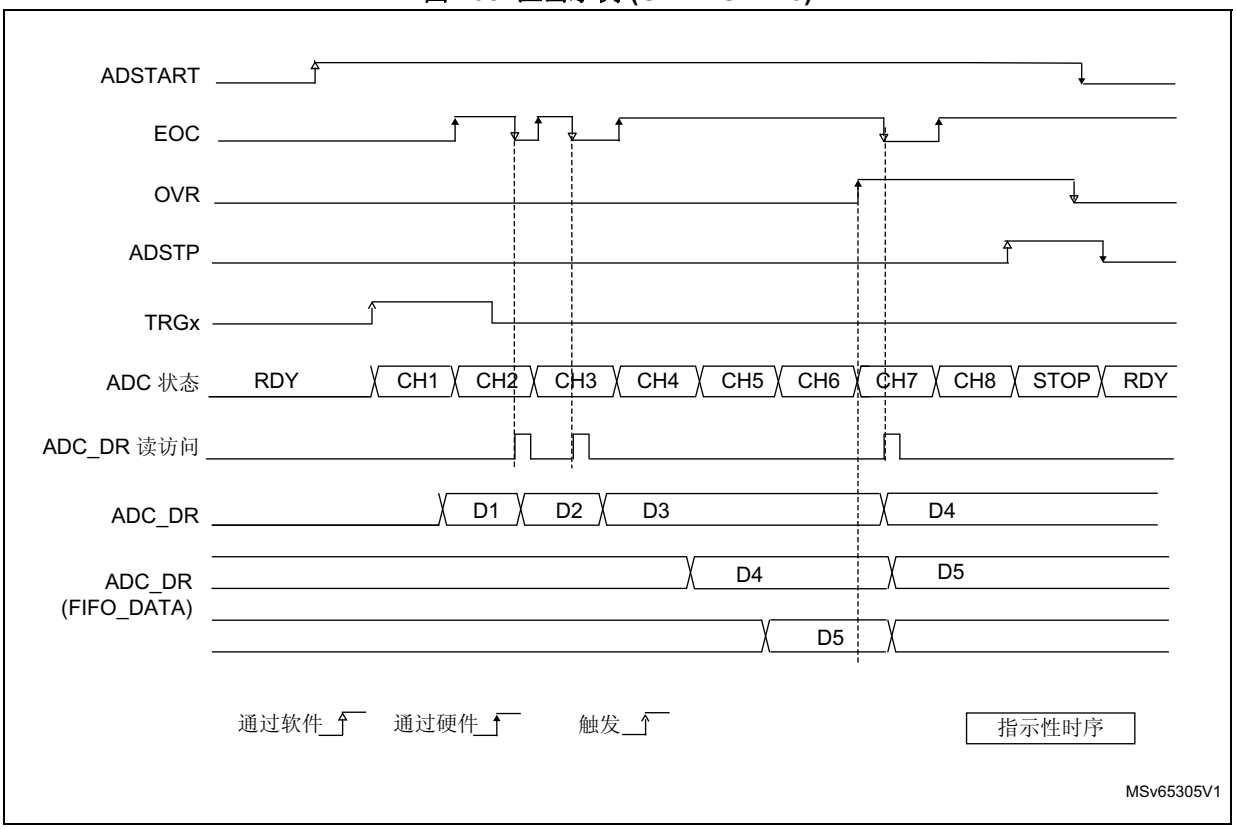
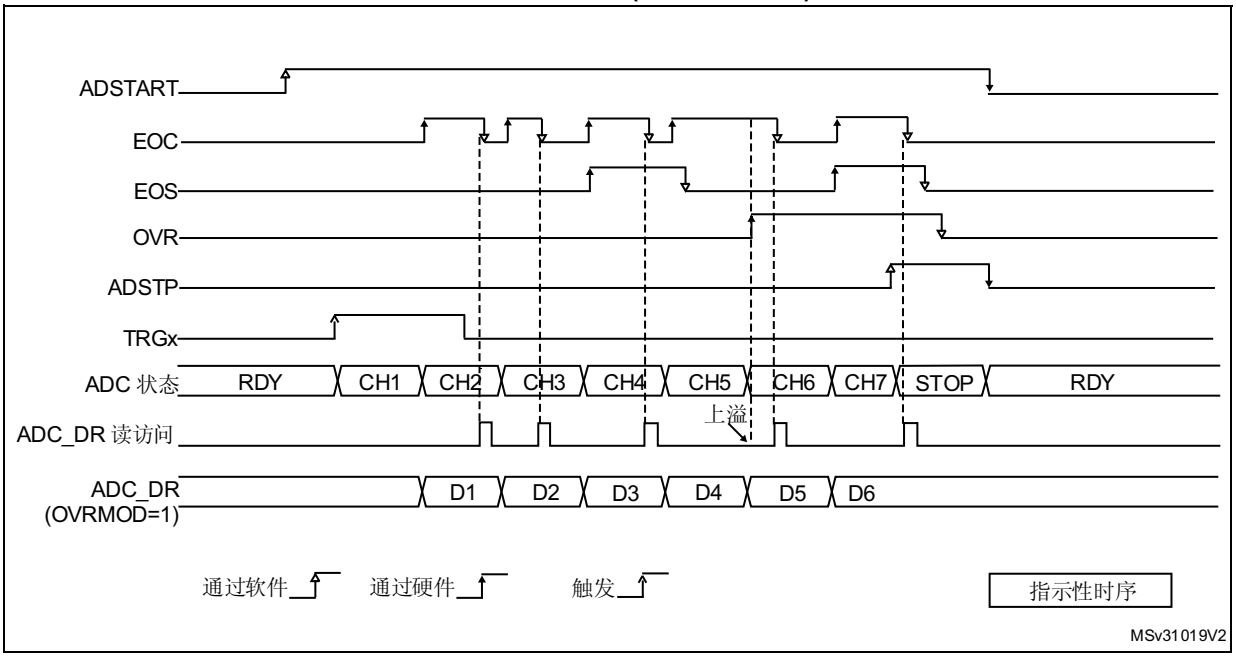


图 109. 溢出示例 (OVRMOD = 1)



注意： *由于四条注入通道均有专用的数据寄存器，因此不会对注入通道进行溢出检测。*

在不使用 DMA 的情况下管理转换序列

如果转换过程足够慢，则可使用软件来处理转换序列。在这种情况下，软件必须使用 EOC 标志及其相关中断来处理各个数据。每当转换结束时，EOC 都会置 1，并且可以读取 ADC_DR 寄存器。OVRMOD 必须配置为 0 才能将溢出事件或 FIFO 上溢作为错误进行管理。

在不使用 DMA 且不发生溢出的情况下管理转换

ADC 在转换一个或多个通道时不是每次都读取数据的情况下，这可能会很有用（例如，存在模拟看门狗时）。在这种情况下，OVRMOD 位必须配置为 1，OVR 标志应被软件忽略。溢出事件不会阻止 ADC 继续进行转换，ADC_DR 寄存器始终包含最新的转换。

使用 DMA 管理转换

由于转换得出的通道值会存储到唯一的数据寄存器中，因此，对于多个通道的转换，使用 DMA 非常有帮助。这样可以避免丢失在下一次写入之前还未被读出的 ADC_DR 寄存器中的数据。

若 DMA 模式已使能（ADC_CFGR 寄存器中的 DMAEN 位置 1），则会在每个通道转换后生成 DMA 请求。这样便可将转换的数据从 ADC_DR 寄存器传输到用软件选择的目标位置。

尽管如此，如果因 DMA 无法及时处理 DMA 传输请求而发生溢出 (OVR = 1)，ADC 会停止生成 DMA 请求，新转换对应的数据不会通过 DMA 进行传输。这意味着可将传输到 RAM 的所有数据都视为有效数据。

根据 OVRMOD 位的配置，可保留或覆盖数据（请参见 *ADC 溢出（OVR、OVRMOD）* 一节）。

DMA 传输请求会禁止，直至软件将 OVR 位清零。

根据应用用途的不同，推荐使用两种不同的 DMA 模式，并使用 ADC_CFGR 寄存器的 DMACFG 位配置相应的模式：

- DMA 单次模式 (DMACFG = 0)。
  如果通过编程将 DMA 设置为传输固定数目的数据，应选择此模式。
- DMA 循环模式 (DMACFG = 1)
  如果在循环模式下对 DMA 进行编程，应选择此模式。

DMA 单次模式 (DMACFG = 0)

在该模式下，每次出现新的转换数据时，ADC 都会生成 DMA 传输请求，并且在 DMA 到达最后一个 DMA 传输操作时（发生传输完成中断，请参见 DMA 部分），即使转换已再次开始，ADC 也会停止生成 DMA 请求。

DMA 传输完成后（在 DMA 控制器中配置的所有传输操作均已完成后）：

- ADC 数据寄存器的内容会冻结。
- 任何正在进行的转换都会中止，其部分结果会被丢弃。
- 不会将任何新的 DMA 请求发送到 DMA 控制器。如果仍存在已开始的转换，这样可避免生成溢出错误。
- 扫描序列会停止并复位。
- DMA 会停止。

DMA 循环模式 (DMACFG = 1)

在该模式下，每次数据寄存器中出现新的转换数据时，ADC 都会生成 DMA 传输请求，即使 DMA 已到达最后一次 DMA 传输操作也不例外。这样可将处于循环模式的 DMA 配置为处理连续的模拟输入数据流。

19.4.27 动态低功耗特性

自动延迟转换模式 (AUTDLY)

ADC 会执行由 AUTDLY 配置位控制的自动延迟转换模式。自动延迟转换可用于简化软件，并可优化采用低频时钟的应用（此类应用可能存在 ADC 溢出的风险）的性能。

AUTDLY = 1 时，仅当同一组内所有之前的数据已处理完毕时，才能开始新的转换：

- 对于常规转换：ADC_DR 寄存器已读取或者 EOC 位已清零时（请参见*图 110*）。
- 对于注入转换：JEOS 位已清零时（请参见*图 111*）。

通过这种方式，可自动调整 ADC 的速度，使其适应系统读取数据的速度。

延时插入到每次常规转换后（无论 DISCEN = 0 还是 1）和每个注入转换序列后（无论 JDISCEN = 0 还是 1）。

注意： *不会在注入转换序列之间插入延时，但会在最后一个序列之后插入延时。*

转换过程中，此延时期间发生的硬件触发事件（针对同一组转换）会被忽略。

注意： *如果延时期间发生软件触发，则不会被忽略，仍可在该延时时间内通过将 ADSTART 或 JADSTART 位置 1 的方式重新启动转换；启动新转换之前，会由软件读取数据。*

不会在不同组的转换之间插入延时（常规转换后即进行注入转换，反之亦然）：

- 如果在常规转换的自动延时期间发生注入触发，注入转换会立即开始（请参见*图 111*）。
- 注入序列完成后，ADC 会等待上一常规转换的延时（如果未结束），然后才会启动新的常规转换（请参见*图 113*）。

自动注入模式 (JAUTO = 1) 下的操作略有不同，仅当上一注入转换序列的自动延时已结束时（JEOS 已清零时），才能开始进行新的常规转换。这是为了确保软件可在启动新序列之前读取给定序列的所有数据（请参见*图 114*）。

要在连续自动注入和自动延时组合模式下停止转换（JAUTO = 1、CONT = 1 且 AUTDLY = 1），请按照以下程序进行操作：

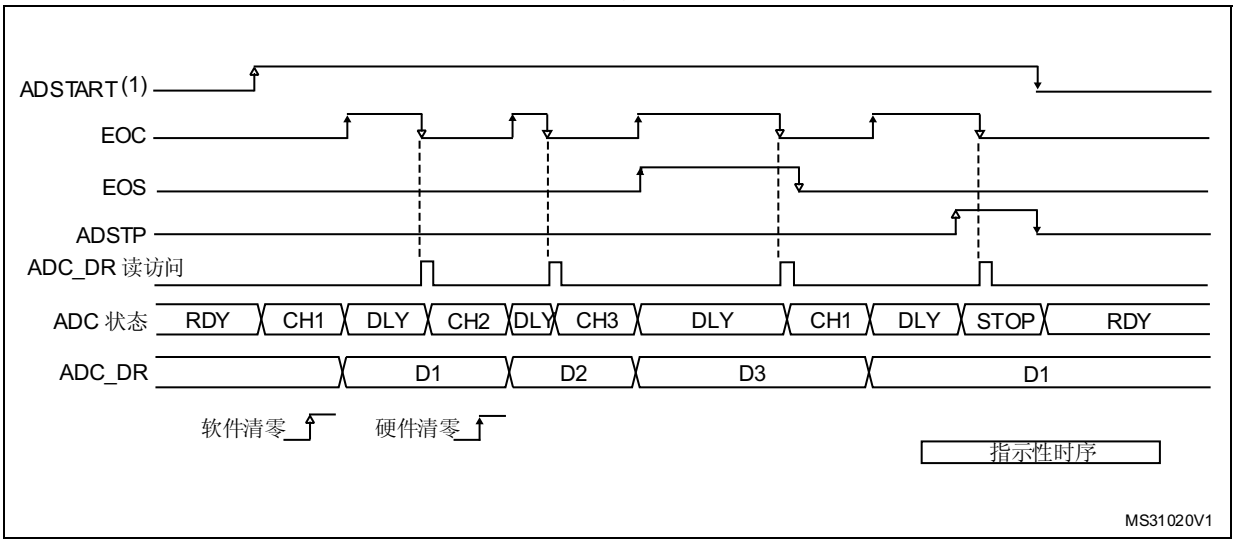
1. 等待，直至 JEOS = 1（不会重新启动任何转换）
2. 将 JEOS 清零
3. 将 ADSTP 置 1
4. 读取常规数据。

否则，如果 JEOS 在 ADSTP 已置 1 后清零，可能重新启动新的常规序列。

在 AUTDLY 模式下，如果在正在进行的常规序列期间或序列的最后一次常规转换之后的延时期间发生硬件常规触发事件，则会忽略该触发事件。但是，如果触发事件发生在该延时之后，即使发生在延时之后的注入序列期间，也会将该触发事件视为待处理事件。随后，会在注入序列的延时结束时开始转换。

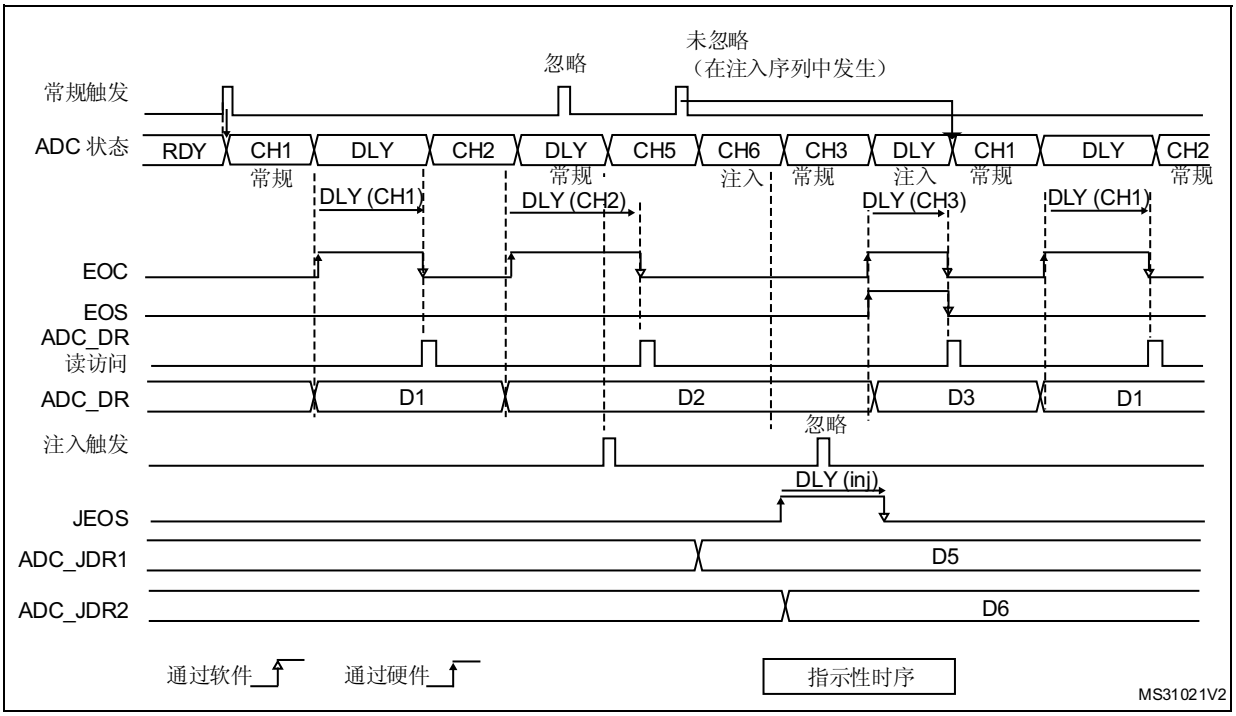
在 AUTDLY 模式下，如果在正在进行的注入序列期间或序列的最后一次注入转换之后的延时期间发生硬件注入触发事件，则会忽略该触发事件。

图 110. AUTODLY = 1，连续模式下的常规转换，软件触发



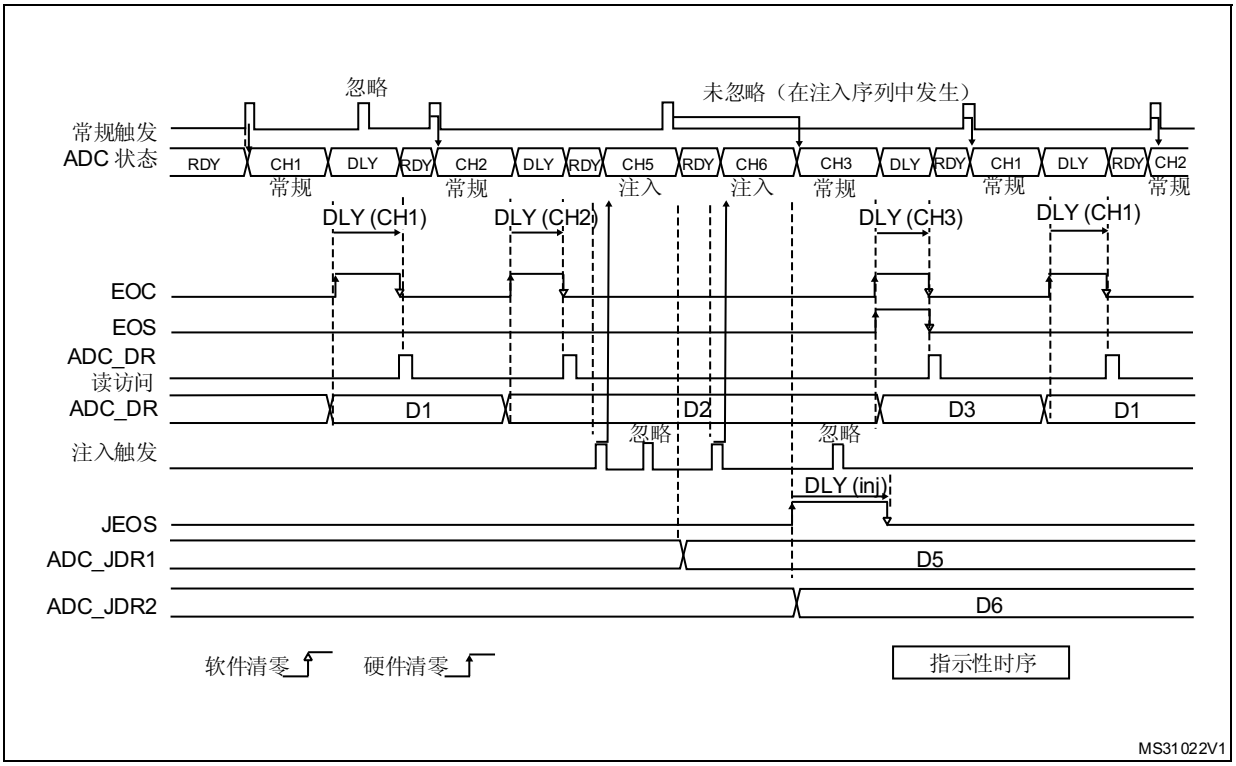
1. AUTDLY = 1
2. 常规配置：EXTEN = 0x0（软件触发），CONT = 1，CHANNELS = 1、2、3
3. 注入配置禁止

图 111. AUTODLY = 1，被注入转换中断的常规硬件转换（DISCEN = 0；JDISCEN = 0）



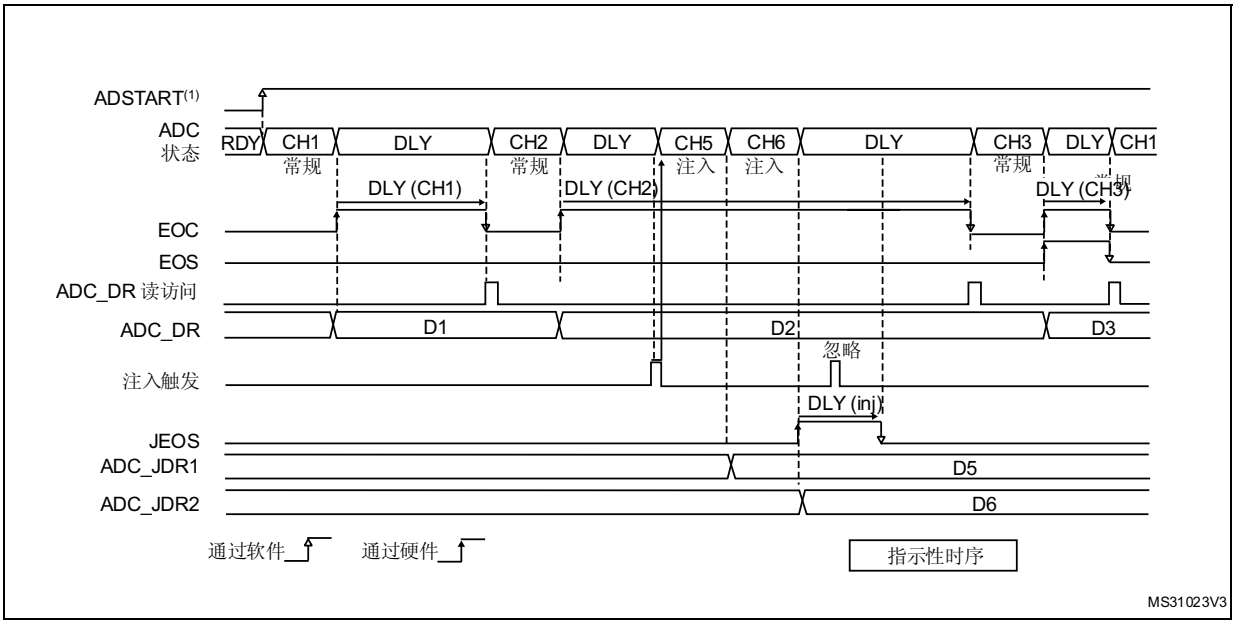
1. AUTDLY = 1
2. 常规配置：EXTEN = 0x1（硬件触发），CONT = 0，DISCEN = 0，CHANNELS = 1、2、3
3. 注入配置：JEXTEN = 0x1（硬件触发），JDISCEN = 0，CHANNELS = 5、6

图 112. AUTODLY = 1，被注入转换中断的常规硬件转换（DISCEN = 1，JDISCEN = 1）



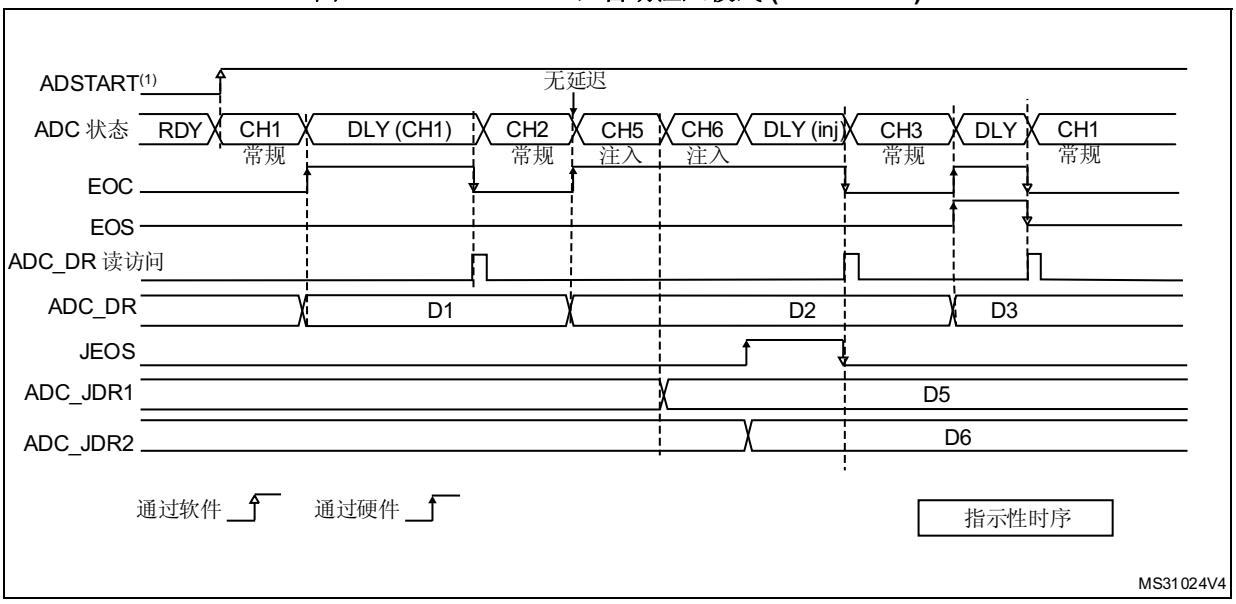
1. AUTDLY = 1
2. 常规配置：EXTEN = 0x1（硬件触发），CONT = 0，DISCEN = 1，DISCNUM = 1，CHANNELS = 1、2、3。
3. 注入配置：JEXTEN = 0x1（硬件触发），JDISCEN = 1，CHANNELS = 5、6

图 113. AUTODLY = 1，被注入转换中断的常规连续转换



1. AUTDLY = 1
2. 常规配置：EXTEN = 0x0（软件触发），CONT = 1，DISCEN = 0，CHANNELS = 1、2、3
3. 注入配置：JEXTEN = 0x1（硬件触发），JDISCEN = 0，CHANNELS = 5、6

图 114. AUTODLY = 1，自动注入模式 (JAUTO = 1)

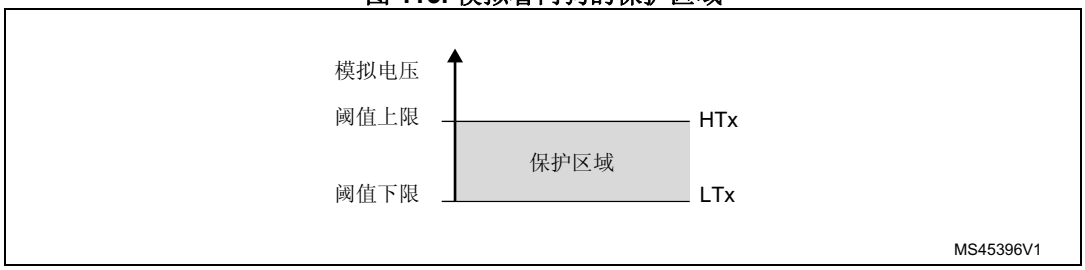


1. AUTDLY = 1
2. 常规配置：EXTEN = 0x0（软件触发），CONT = 1，DISCEN = 0，CHANNELS = 1、2
3. 注入配置：JAUTO = 1，CHANNELS = 5、6

19.4.28 模拟窗口看门狗（AWD1EN、JAWD1EN、AWD1SGL、AWD1CH、AWD2CH、AWD3CH、AWD_HTx、AWD_LTx、AWDx）

三个 AWD 模拟看门狗会监测一些通道是否保持在配置的电压范围（窗口）内。

图 115. 模拟看门狗的保护区域



AWDx 标志和中断

可通过将 ADC_IER 寄存器中的 AWDxIE（x = 1、2、3）置 1 的方式分别为 3 个模拟看门狗使能中断。

AWDx（x = 1、2、3）标志可通过由软件向其写入 1 的方式来清零。

在对齐之前，会将 ADC 转换结果与阈值上限和下限进行比较。

模拟看门狗 1 说明

将 ADC_CFGR 寄存器中的 AWD1EN 位置 1，可使能 AWD 模拟看门狗 1。该看门狗监测一条已选通道或所有已使能通道(1)是否仍在配置的电压范围（窗口）内。

表 114 介绍了应如何配置 ADC_CFGR 寄存器才能在一条或多个通道上使能模拟看门狗。

表 114. 模拟看门狗通道选择

模拟看门狗保护的通道	AWD1SGL 位	AWD1EN 位	JAWD1EN 位
无	x	0	0
所有注入通道	0	0	1
所有常规通道	0	1	0
所有常规通道和注入通道	0	1	1
单个(1)注入通道	1	0	1
单个(1)常规通道	1	1	0
单个(1)常规通道或注入通道	1	1	1

1. 通过 AWD1CH[4:0] 位选择。此外，还必须将通道编程为在合适的常规序列或注入序列中进行转换。

如果 ADC 转换的模拟电压低于阈值下限或高于阈值上限，则 AWD1 模拟看门狗状态位会置 1。

这些阈值编程到模拟看门狗 1 的 ADC_TR1 寄存器的 HT1[11:0] 和 LT1[11:0] 位。如果转换的数据分辨率小于 12 位（取决于 RES[1:0] 位），由于在偏移补偿阶段前始终会在内部对完整的 12 位原始转换数据进行比较（左对齐），因此编程阈值的 LSB 必须保持清零状态。

表 115 介绍了如何对模拟看门狗 1 的所有可能的分辨率进行比较。

表 115. 模拟看门狗 1 比较

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

看门狗 1 的模拟看门狗滤波器

如果 ADC 配置为仅有一个输入通道（不允许在扫描模式下选择多个通道），则可以通过 ADC_TR1 寄存器配置一个有效的 ADC 转换数据区间：

- 如果转换后的数据属于 ADC_TR1 中定义的区间，会生成 DMA 请求。
- 否则，不会发出 DMA 请求。每次转换时都更新 RDATA 寄存器。如果数据超出区间的次数超出 ADC_TR1 的 AWDFILT 位所指定的值，则 AWDx 标志将置 1，并会发出相应的中断。

模拟看门狗 2 和 3 说明

第二个和第三个模拟看门狗更加灵活，可通过编程 AWDxCH[19:0]（x = 2、3）中的相应位来保护多条已选通道。

AWDxCH[19:0]（x = 2、3）的任何位置 1 时，会使能相应的看门狗。

其分辨率被限制为 8 位，仅高 8 位阈值可编程到 HTx[7:0] 和 LTx[7:0] 中。*表 116* 介绍了如何对所有可能的分辨率进行比较。

表 116. 模拟看门狗 2 和 3 比较

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

ADCy_AWDx_OUT 信号输出生成

每个模拟看门狗都关联到一个内部硬件信号 ADCy_AWDx_OUT（y = ADC 编号，x = 看门狗编号），该信号直接连接到一些片上定时器的 ETR 输入（外部触发）。请参见片上定时器部分以了解如何选择 ADCy_AWDx_OUT 信号作为 ETR。

当关联的模拟看门狗使能时，ADCy_AWDx_OUT 会激活：

- 当受保护的转换超出编程阈值时，ADCy_AWDx_OUT 会置 1。
- 在编程阈值范围内的下一受保护转换结束后，ADCy_AWDx_OUT 会复位（如果下一受保护转换仍超出编程阈值范围，该位仍保持置 1）。
- 禁止 ADC 时（将 ADDIS 置 1 时），ADCy_AWDx_OUT 也保持复位状态。请注意，停止常规转换或注入转换（将 ADSTP 或 JADSTP 置 1）对 ADCy_AWDx_OUT 的生成没有任何影响。

注意： *AWDx 标志由硬件置 1，由软件复位。AWDx 标志对生成 ADCy_AWDx_OUT 无影响（例如：如果软件未将 AWDx 标志清零，则在该标志保持为 1 时 ADCy_AWDx_OUT 可以切换）。*

图 116. ADCy_AWDx_OUT 信号生成（所有常规通道上）

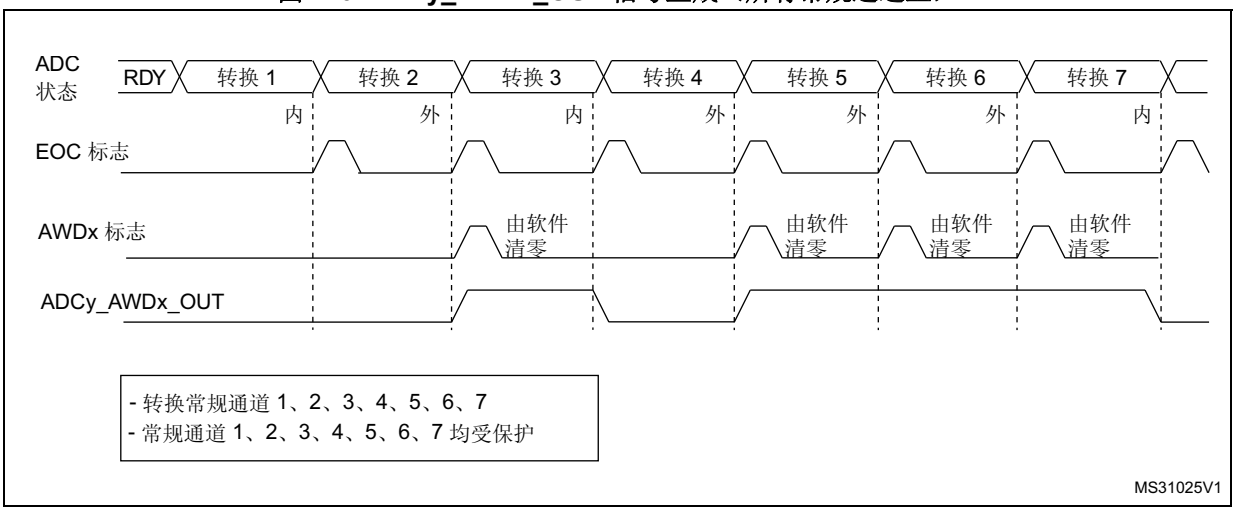


图 117. ADCy_AWDx_OUT 信号生成（AWDx 标志未通过软件清零）

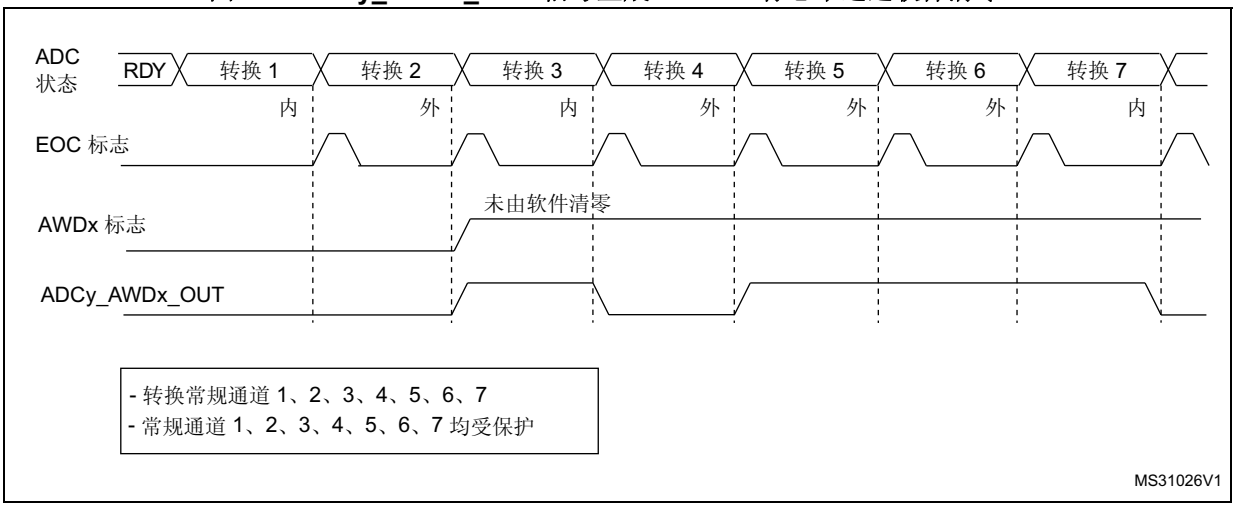


图 118. ADCy_AWDx_OUT 信号生成（单条常规通道上）

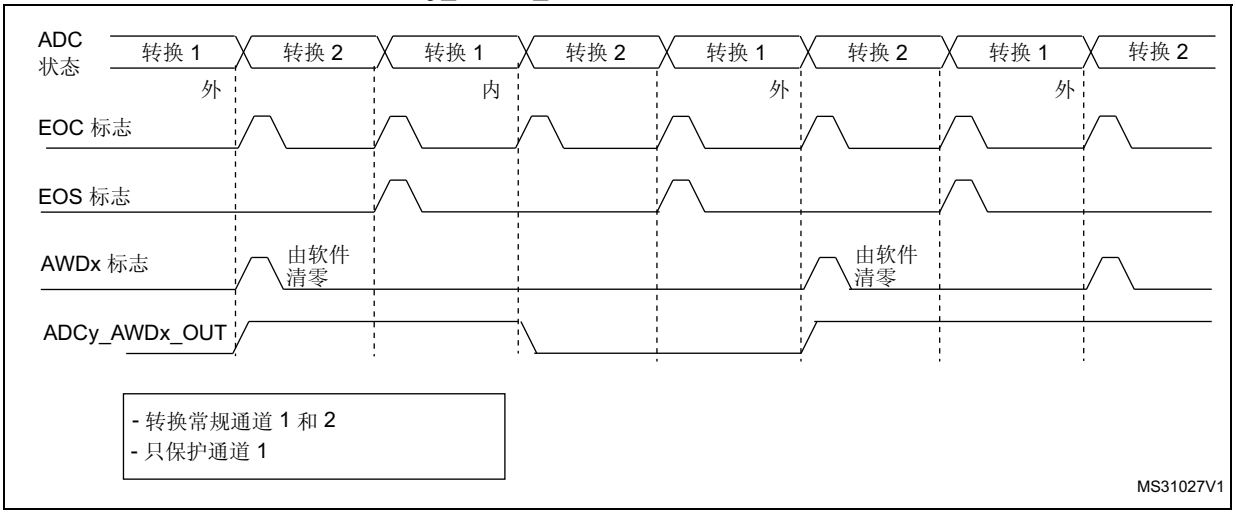
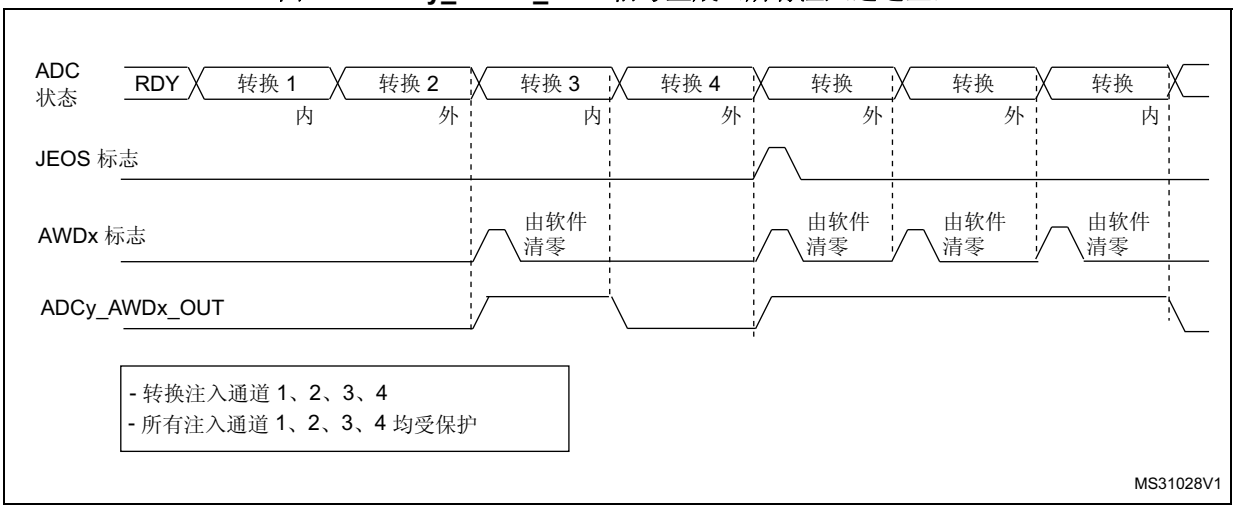


图 119. ADCy_AWDx_OUT 信号生成（所有注入通道上）



模拟看门狗阈值控制

LTx[11:0] 和 HTx[11:0] 可在进行模数转换时（即，从 ADC 内部状态转换开始到转换结束）更改。如果在 ADC 保护的通道上进行 ADC 转换期间 LTx[11:0] 和 HTx[11:0] 发生更新，则会对该转换屏蔽看门狗功能。该屏蔽在下一次转换开始时移除，从而通过下一次 ADC 转换应用模拟看门狗阈值。模拟看门狗比较在每次转换结束时执行。如果当前 ADC 数据超出新的区间，则不会发出中断和 AWDx_OUT 信号。只有在阈值更新后开始的转换结束时，才会生成中断和 AWD。如果 AWD_xOUT 已置为有效，则编程设定新的阈值不会将 AWDx_OUT 信号置为无效。

模拟看门狗与偏移补偿

如果使能了偏移补偿，则模拟看门狗会在数据补偿阶段之前比较阈值。

19.4.29 过采样器

过采样单元会进行数据预处理，以减轻 CPU 的负担。过采样单元能够处理多个转换，并计算多个转换结果的平均值，得到数据宽度增加（高达 16 位）的单个数据。

它提供的结果采用以下形式，其中的 N 和 M 可以进行调整：

$$\text{结果} = \frac{1}{M} \times \sum_{n=0}^{n=N-1} \text{转换}(t_n)$$

允许通过硬件执行以下功能：计算平均值、降低数据速率、改进 SNR 以及基本滤波。

过采样率 N 通过 ADC_CFGR2 寄存器中的 OVFS[2:0] 位进行定义，范围为 2x 到 256x。分频系数 M 通过向右移位来实现（最多可移 8 位），并且通过 ADC_CFGR2 寄存器中的 OVSS[3:0] 位定义。

求和单元可得出多达 20 位的结果（256x 12 位结果），结果会先右移。随后会截断为 16 个最低有效位，这些位使用移位后剩下的最低有效位四舍五入为最接近的数值，然后将最终得到的结果传输到 ADC_DR 数据寄存器中。

注意： *如果移位后得到的中间结果超过 16 位，则会将结果按原样截断（无饱和）。*

图 120. 20 位到 16 位结果的截断过程

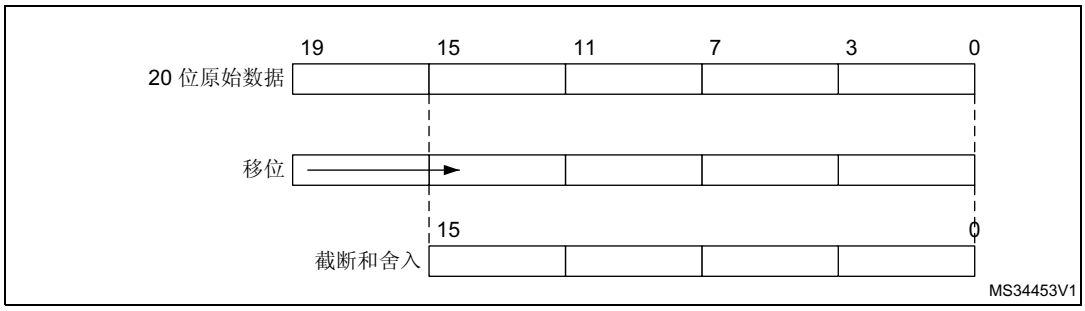


图 121 给出了从原始的 20 位累加数据到最终 16 位结果的数值处理过程示例。

图 121. 移 5 位并进行舍入的数值示例

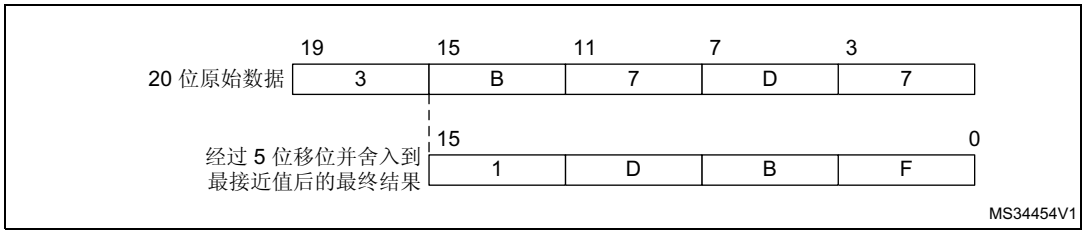


表 117 列出了原始转换数据等于 0xFFF 时对应的各种 N 和 M 组合的数据格式。

表 117. 最大输出结果与 N 和 M 的对应关系（灰色单元格表示截断的部分）

过采样率	最大原始数据	不移位 OVSS = 0000	移 1 位 OVSS = 0001	移 2 位 OVSS = 0010	移 3 位 OVSS = 0011	移 4 位 OVSS = 0100	移 5 位 OVSS = 0101	移 6 位 OVSS = 0110	移 7 位 OVSS = 0111	移 8 位 OVSS = 1000
2x	0x1FFE	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040	0x020
4x	0x3FFC	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040
8x	0x7FF8	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080
16x	0xFFF0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100
32x	0x1FFE0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200
64x	0x3FFC0	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400
128x	0x7FF80	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800
256x	0xFFF00	0xFF00	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF

过采样模式下的转换时序不会发生变化：在整个过采样序列中，采样时间保持不变。每完成 N 次转换都会提供新数据，等效延迟等于 N x T_{CONV} = N x (t_{SMPL} + t_{SAR})。各标志的置 1 情况如下：

- 每个采样阶段后都会将采样阶段结束标志 (EOSMP) 置 1
- 如果过采样结果可用，每完成 N 次转换都会发生转换结束事件 (EOC)
- 过采样数据序列完成后（即 N x 序列长度次转换之后），会发生序列结束事件 (EOS)

过采样时支持的 ADC 工作模式

在过采样模式下，大部分 ADC 工作模式都会保留：

- 单次或连续转换模式
- 可由软件或触发器启动 ADC 转换
- ADC 在转换过程中停止（中止）
- 通过 CPU 或 DMA 在支持溢出检测的情况下读取数据
- 低功耗模式 (AUTDLY)
- 可编程分辨率：在这种情况下，会按照与 12 位转换相同的方式对分辨率降低的转换值（根据 ADC_CFGR 寄存器中的 RES[1:0] 位）进行累加、截断、四舍五入和移位

注意： *处理过采样数据时，不可使用对齐模式。ADC_CFGR 中的 ALIGN 位会被忽略，且数据始终采用右对齐格式。*

过采样模式下不支持偏移校正。如果 ROVSE 和/或 JOVSE 位置 1，ADC_OFRy 寄存器中 OFFSET_EN 位的值会被忽略（视为复位）。

模拟看门狗

可以继续使用模拟看门狗功能，但存在以下区别：

- 会忽略 RES[1:0] 位，始终会使用完整的 12 位值 HT1[11:0] 和 LT1[11:0]（对于 AWD1）以及 HT2/HT3[7:0] 和 LT2/LT3[7:0] 的 8 个最高有效位值（对于 AWD2 和 AWD3）进行比较。
- 会比较 16 位过采样结果 ADC_DR[15:4]（对于 AWD1）以及 ADC_DR[15:8]（对于 AWD2 和 ADW3）的 12 个最高有效位

注意： *移位位数较大时必须多加留意，因为这样会缩小比较范围。例如，如果过采样结果移了 4 位，得到 12 位右对齐数据，那么只能对 8 个数据位执行有效的模拟看门狗比较。比较是在 ADC_DR[11:4] 与 HTx[7:0] / LTx[7:0] (AWD1/2/3) 之间进行，HT1[11:8] 和 LT1[11:8] 保持复位（仅限 AWD1）。*

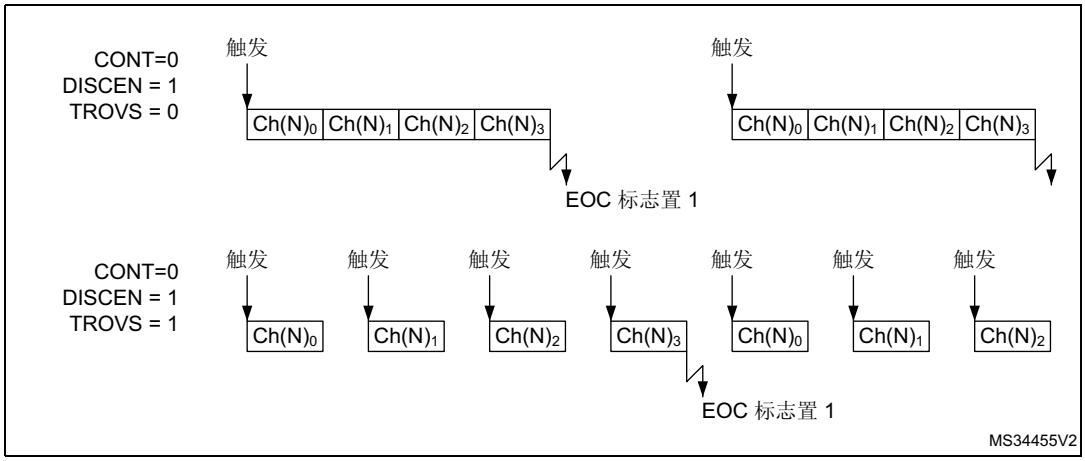
触发模式

平均值计算单元还可用于基本滤波，虽然它不是非常强大的滤波器（衰减缓慢、停止频段衰减受限），但可用作陷波滤波器，用于抑制恒定的寄生频率（通常来自电源或切换模式电源）。为此，可使用 ADC_CFGR2 中的 TROVS 位使能特定的不连续模式，以获得由用户定义、且与转换时间本身无关的过采样频率。

图 122 显示了在不连续模式下如何响应触发从而开始转换。

如果 TROVS 位置 1，则会忽略 DISCEN 位的内容，并将该位视为 1。

图 122. 已触发的常规过采样模式（TROVS 位 = 1）



过采样时的注入和常规定序器管理

在过采样模式下，注入定序器和常规定序器可以执行不同操作。如果两个定序器必须同时使用，则可为它们使能过采样并设定一些限制条件（与唯一的累加单元相关）。

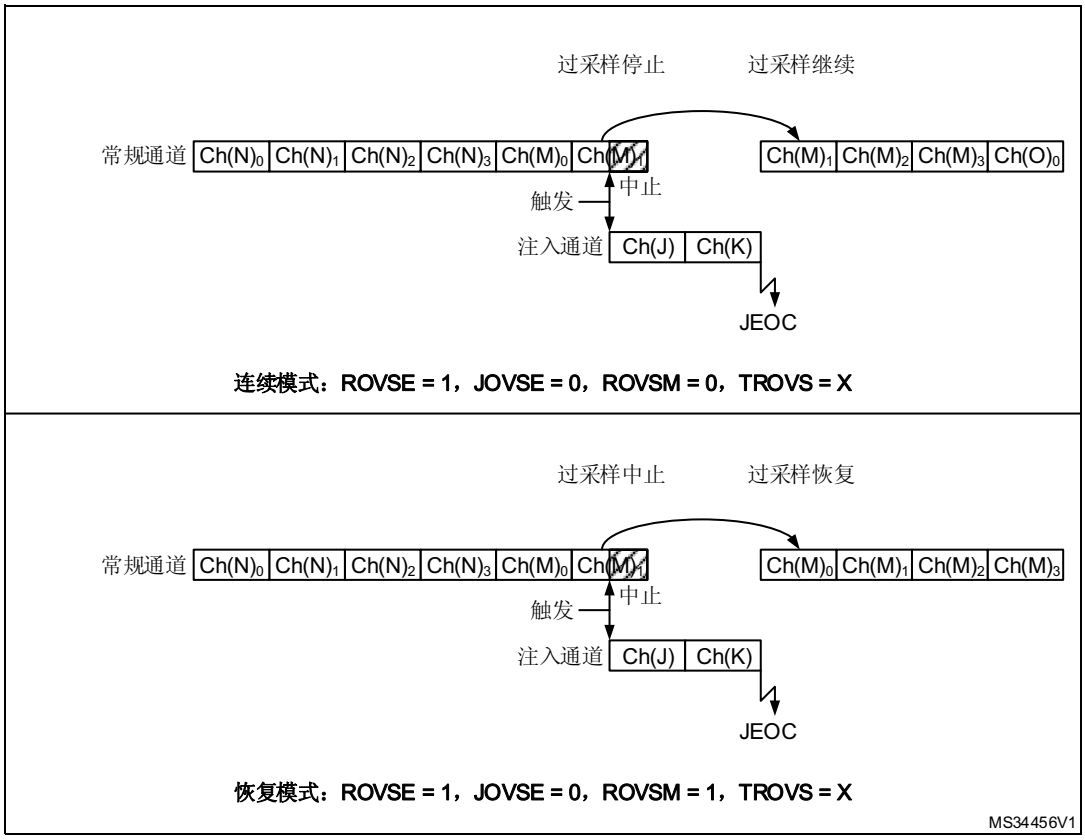
仅对常规通道进行过采样

常规过采样模式位 ROVSM 定义了常规过采样序列在被注入转换中断的情况下如何恢复。

- 在连续模式下，会从上一有效数据开始重新累加（在由于注入触发而发出的转换中止请求之前）。这样可确保在任何注入频率下均可而完成过采样（假设触发之间至少可完成一次常规转换）；
- 在恢复模式下，会从 0 开始重新累加（会忽略之前的转换结果）。该模式可确保所有用于过采样的数据在单个时隙内进行了背靠背转换。需要注意的是，注入触发周期必须超过过采样时长。如果该条件未得到满足，则无法完成过采样，常规定序器将被禁用。

图 123 以 4x 过采样率为例进行了说明。

图 123. 常规过采样模式（4x 过采样率）



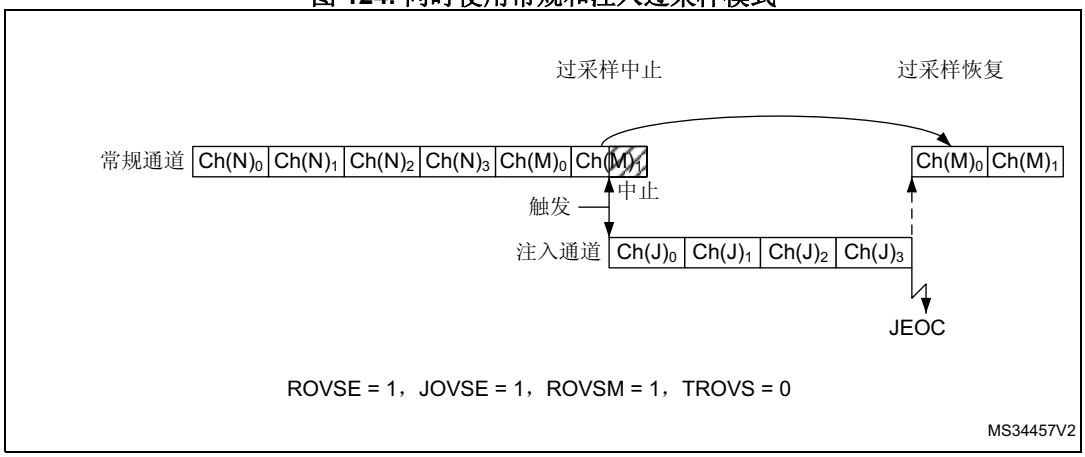
仅对注入通道进行过采样

注入过采样模式位 JOVSE 专为注入定序器中的转换使能过采样。

对常规通道和注入通道进行过采样

可以将 ROVSE 和 JOVSE 位都置 1。在这种情况下，常规过采样模式会强制进入恢复模式（忽略 ROVSM 位），如*图 124* 所示。

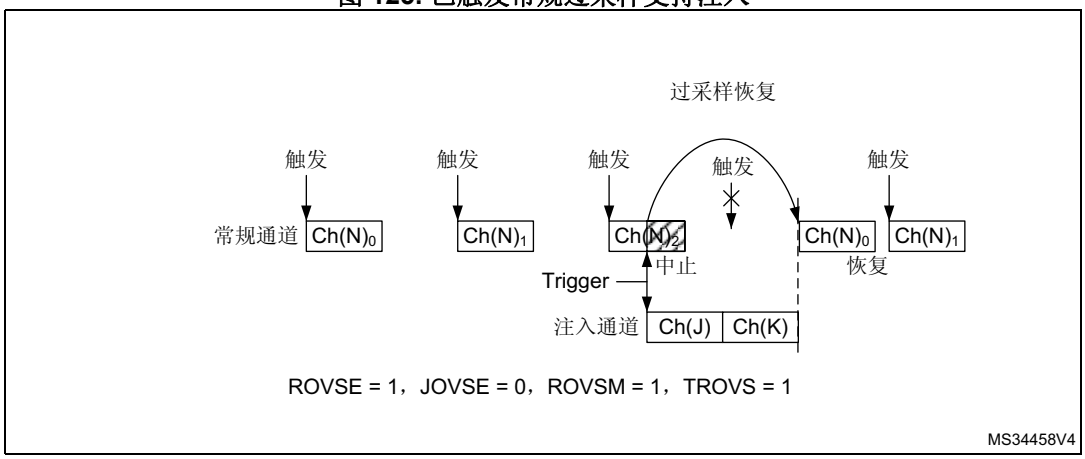
图 124. 同时使用常规和注入过采样模式



已触发常规过采样支持注入转换

已触发常规模式下支持注入转换。在这种情况下，必须禁止注入过采样模式，并且会忽略 ROVSM 位（强制进入恢复模式）。JOVSE 位必须复位。具体操作如*图 125* 所示。

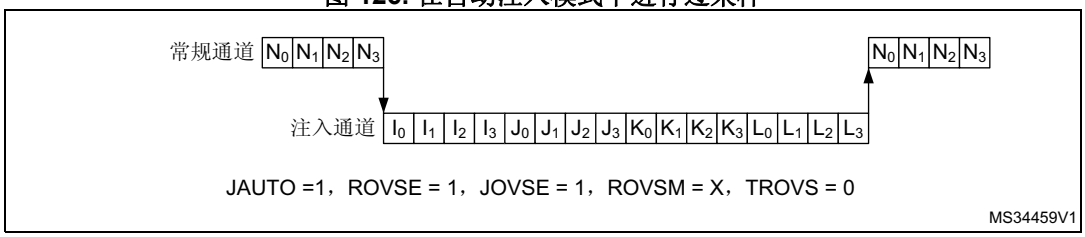
图 125. 已触发常规过采样支持注入



自动注入模式

可以对自动注入序列进行过采样，并将所有转换结果存储在寄存器中以保存 DMA 资源。使用该模式的前提条件是常规和注入过采样均激活：JAUTO = 1、ROVSE = 1 且 JOVSE = 1，不支持其他组合。在自动注入模式下，会忽略 ROVSM 位。*图 126* 显示了转换的排序方式。

图 126. 在自动注入模式下进行过采样



此外，还可以使用 TROVS 位使能触发模式。此时，ADC 必须如下配置：JAUTO = 1、DISCEN = 0、JDISCEN = 0、ROVSE = 1、JOVSE = 1 且 TROVSE = 1。

19.4.30 温度传感器

温度传感器可用于测量器件的结温 (Tj)。

温度传感器内部连接到 ADC 输入通道，这类通道用于将传感器输出电压转换为数字值（更多详细信息，请参见*表：ADC 互连*，该表在*第 19.4.2 节：ADC 引脚和内部信号*中）。不使用时可将传感器置于掉电模式。支持的温度范围：-40 ℃ 到 125 °C。

图 127 显示的是温度传感器与 ADC 之间连接的框图。

温度传感器的输出电压随温度线性变化。由于工艺不同，该线的偏移量取决于各个芯片（芯片之间的温度变化可达 45 °C）。

未校准的内部温度传感器更适用于对温度变量而非绝对温度进行测量的应用。为提高温度传感器测量的准确性，ST 在生产过程中将校准值存储在每个器件的系统存储器中。

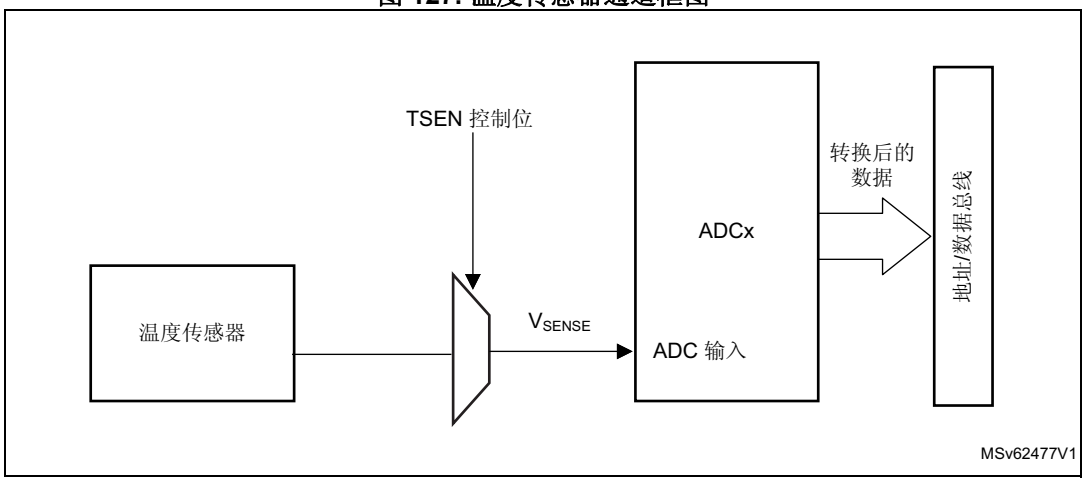
在制造过程中，会将温度传感器的校准数据和内部参考电压存储在系统存储区。随后，用户应用可读取这些数据，并使用这些数据提高温度传感器或内部参考的准确性（更多相关信息，请参见数据手册）。

温度传感器在内部连接到 ADC 输入通道，该通道用于将传感器输出电压转换为数字值。有关转换内部温度传感器时要应用的采样时间值，请参见器件数据手册的电气特性部分。

不使用时可将传感器置于掉电模式。

图 127 显示了温度传感器框图。

图 127. 温度传感器通道框图



读取温度

要使用传感器，请执行以下操作：

1. 选择连接到 V_{SENSE} 的 ADC 输入通道。
2. 设定合适的采样时间（请参见器件数据手册中的电气特性部分）。
3. 将 ADC_CCR 寄存器中的位置 1，以便将温度传感器从掉电模式中唤醒。
4. 开始 ADC 转换。
5. 读取 ADC 数据寄存器中生成的 V_{SENSE} 数据。
6. 使用以下公式计算实际温度：

$$\text{温度 (°C)} = \frac{\text{TS_CAL2_TEMP} - \text{TS_CAL1_TEMP}}{\text{TS_CAL2} - \text{TS_CAL1}} \times (\text{TS_DATA} - \text{TS_CAL1}) + \text{TS_CAL1_TEMP}$$

其中：

- TS_CAL2 是在 TS_CAL2_TEMP 下获得的温度传感器校准值。
- TS_CAL1 是在 TS_CAL1_TEMP 下获得的温度传感器校准值。
- TS_DATA 是由 ADC 转换得到的实际温度传感器输出值。
  更多关于 TS_CAL1 和 TS_CAL2 校准点的信息，请参见器件数据手册。

注意： *传感器从掉电模式中唤醒需要一个启动时间，启动时间过后其才能正确输出 V_{SENSE}。ADC 在上电后同样需要一个启动时间，因此，为尽可能缩短延迟时间，应同时将 ADEN 和相关位置 1。*

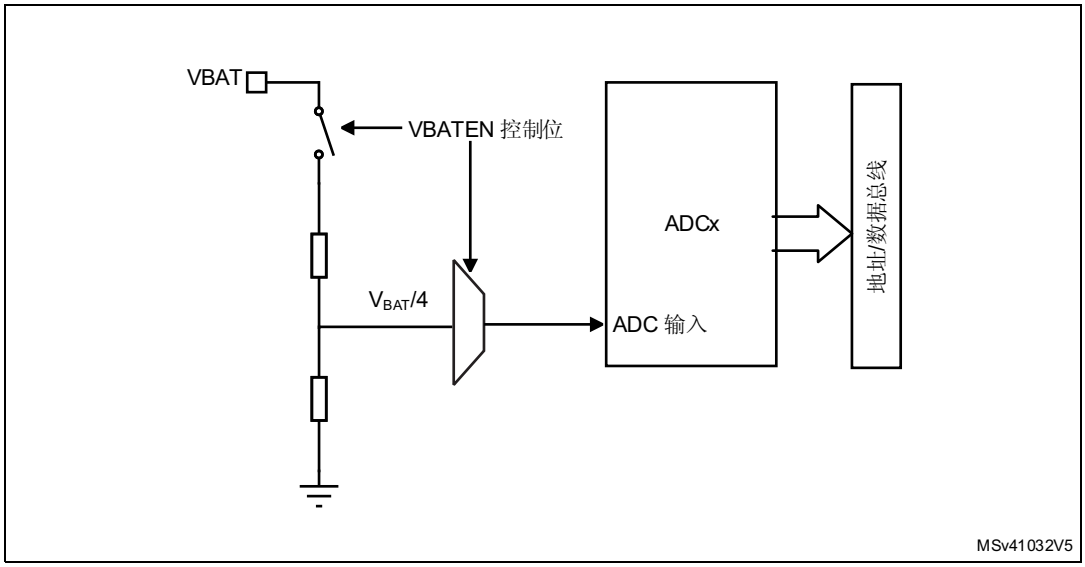
19.4.31 V_{BAT} 电源监测

ADC_CCR 寄存器中的 VBATEN 位用于切换到电池电压。由于 V_{BAT} 电压可能高于 V_{DDA}，因此 V_{BAT} 引脚需要从内部连接到桥接分配器（除以 4），以确保 ADC 正确运行。VBATEN 置 1 时，会自动使能此桥，以将 V_{BAT}/4 连接到 ADC 输入通道（更多详细信息，请参见*表：ADC 互连*，该表在*第 19.4.2 节：ADC 引脚和内部信号*中）。因此，转换出的数字值为 V_{BAT} 电压的三分之一。为防止电池出现意外的电能消耗，建议仅在必要时为 ADC 转换使能桥接分配器。

有关转换 V_{BAT}/4 电压时要应用的采样时间值，请参见器件数据手册的电气特性部分。

图 128 显示了 V_{BAT} 传感特性的框图。

图 128. V_{BAT} 通道框图



1. 必须将 VBATEN 位置 1 才能为 V_{BAT}/4 使能内部通道转换。

19.4.32 监测内部参考电压

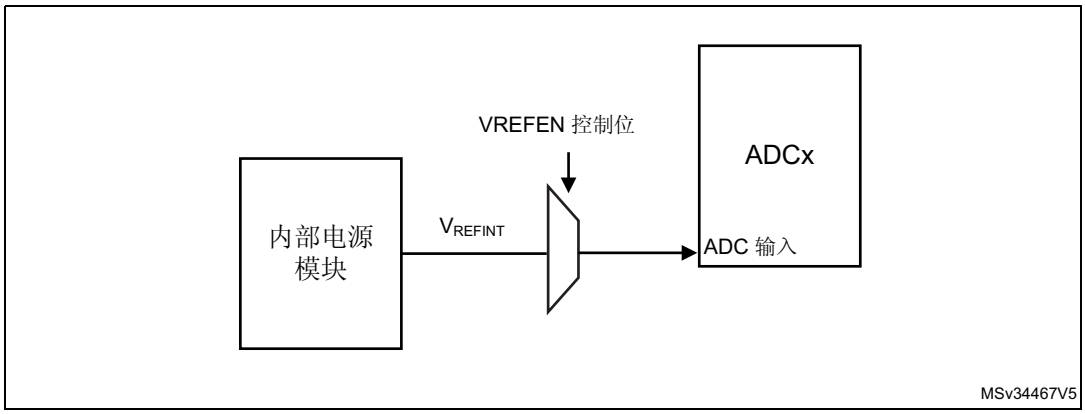
可以通过监测内部参考电压 (V_{REFINT}) 来获得用于评估 ADC V_{REF+} 电压的参考值。

有关内部参考电压在内部连接的 ADC 输入通道的详细信息，请参见*表：ADC 互连*（在*第 19.4.2 节：ADC 引脚和内部信号*中）。

有关转换内部参考电压时要应用的采样时间值，请参见产品数据手册的电气特性部分。

图 129 显示了 V_{REFINT} 传感特性的框图。

图 129. V_{REFINT} 通道框图



1. 必须将 ADC_CCR 寄存器中的 VREFEN 位置 1 才能使能内部通道 (V_{REFINT}) 的转换。

使用内部参考电压计算实际的 V_{REF+} 电压

V_{REF+} 电压可能会有变化，或无法获得准确值。在制造过程中由 ADC 在 V_{REF+_charac} 的条件下获得的内置内部参考电压 (V_{REFINT}) 及其校准数据可用于评估实际的 V_{REF+} 电压。

以下公式可求得为器件供电的实际 V_{REF+} 电压：

$$V_{REF+} = V_{REF+_Charac} \times VREFINT_CAL / VREFINT_DATA$$

其中：

- V_{REF+_Charac} 是制造过程中 V_{REF+} 电压在 V_{REFINT} 下的特性值。该值在器件数据手册中指定。
- VREFINT_CAL 是 VREFINT 校准值
- VREFINT_DATA 是由 ADC 转换得到的实际 VREFINT 输出值

将电源相关的 ADC 测量值转换为绝对电压值

ADC 用于提供对应于 V_{REF+} 与施加给转换通道的电压之比的数字值。

适用于 V_{REF+} 值未知并且 ADC 转换值为右对齐的应用。在这类应用中，需要将该比值转换为与 V_{REF+} 无关的电压：

$$V_{CHANNELx} = \frac{V_{REF+}}{FULL_SCALE} \times ADC_DATA$$

将 V_{REF+} 替换为上面的公式，就可以通过下面的公式求得绝对电压值

$$V_{CHANNELx} = \frac{V_{REF+_Charac} \times VREFINT_CAL \times ADC_DATA}{VREFINT_DATA \times FULL_SCALE}$$

其中：

- V_{REF+_Charac} 是制造过程中 V_{REF+} 电压在 V_{REFINT} 下的特性值。
- VREFINT_CAL 是 VREFINT 校准值
- ADC_DATA 是由 ADC 在通道 x 上测得的值（右对齐）
- VREFINT_DATA 是由 ADC 转换得到的实际 VREFINT 输出值
- FULL_SCALE 是 ADC 输出的最大数字值。例如，如果分辨率为 12 位，该值为 $2^{12} - 1 = 4095$，如果分辨率为 8 位，该值为 $2^8 - 1 = 255$。

注意： *如果执行 ADC 测量时使用的是输出格式而非 12 位右对齐格式，那么必须先将所有参数转换为兼容格式，然后再进行计算。*

19.4.33 监视电源电压

ADC 连接到内部电源电压。若要使用 ADC 测量此电压，可通过 ADC 选项寄存器使能连接。

19.5 ADC 中断

在以下情况下会生成中断：

- ADC 就绪后，ADC 上电（ADRDY 标志）
- 常规组的任何转换结束时（EOC 标志）
- 常规组的转换序列结束时（EOS 标志）
- 注入组的任何转换结束时（JEOC 标志）
- 注入组的转换序列结束时（JEOS 标志）
- 发生模拟看门狗检测时（AWD1、AWD2 和 AWD3 标志）
- 采样阶段结束时（EOSMP 标志）
- 发生数据溢出时（OVR 标志）
- 注入序列上下文队列溢出时（JQOVF 标志）

可以使用单独的中断使能位以提高灵活性。

表 118. ADC 中断

中断向量	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止、待机模式
ADC	ADC 就绪	ADRDY	ADRDYIE	由硬件置 1，由软件清零	是	否
	常规组的转换结束	EOC	EOCIE			
	常规组的转换序列结束	EOS	EOSIE			
	注入组的转换结束	JEOC	JEOCIE			
	注入组的转换序列结束	JEOS	JEOSIE			
	模拟看门狗 1 状态位置 1	AWD1	AWD1IE			
	模拟看门狗 2 状态位置 1	AWD2	AWD2IE			
	模拟看门狗 3 状态位置 1	AWD3	AWD3IE			
	采样阶段结束	EOSMP	EOSMPIE			
	溢出	OVR	OVRIE			
	注入上下文队列溢出	JQOVF	JQOVFIE			

19.6 ADC 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节*。

19.6.1 ADC 中断和状态寄存器 (ADC_ISR)

ADC interrupt and status register

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	JQOVF	AWD3	AWD2	AWD1	JEOS	JEOC	OVR	EOS	EOC	EOSMP	ADRDY
					rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

位 31:11 保留，必须保持复位值。

位 10 **JQOVF**：注入上下文队列溢出 (Injected context queue overflow)
当注入上下文队列溢出时，该位会由硬件置 1。通过软件写入 1 可将该位清零。更多信息，请参见*第 19.4.21 节：注入转换的上下文队列*。
0：未发生注入上下文队列溢出（或标志事件已通过软件确认并清零）
1：发生注入上下文队列溢出

位 9 **AWD3**：模拟看门狗 3 标志 (Analog watchdog 3 flag)
当转换电压超过在 ADC_TR3 寄存器的 LT3[7:0] 和 HT3[7:0] 位域中编程的值时，硬件会将该位置 1。通过软件写入 1 可将该位清零。
0：未发生模拟看门狗 3 事件（或标志事件已通过软件确认并清零）。
1：发生模拟看门狗 3 事件

位 8 **AWD2**：模拟看门狗 2 标志 (Analog watchdog 2 flag)
当转换电压超过在 ADC_TR2 寄存器的 LT2[7:0] 和 HT2[7:0] 位域中编程的值时，硬件会将该位置 1。通过软件写入 1 可将该位清零。
0：未发生模拟看门狗 2 事件（或标志事件已通过软件确认并清零）。
1：发生模拟看门狗 2 事件

位 7 **AWD1**：模拟看门狗 1 标志 (Analog watchdog 1 flag)
当转换电压超过在 ADC_TR1 寄存器的 LT1[11:0] 和 HT1[11:0] 位域中编程的值时，硬件会将该位置 1。通过软件写入 1 可将该位清零。
0：未发生模拟看门狗 1 事件（或标志事件已通过软件确认并清零）。
1：发生模拟看门狗 1 事件

位 6 **JEOS**：注入通道序列结束标志 (Injected channel end of sequence flag)
组内所有注入通道转换结束时，硬件将该位置 1。通过软件写入 1 可将该位清零。
0：注入转换序列未完成（或标志事件已通过软件确认并清零）
1：注入转换已完成

位 5 **JEOC**：注入通道转换结束标志 (Injected channel end of conversion flag)
当通道的每次注入转换结束，新数据出现在相应的 ADC_JDRy 寄存器时，会通过硬件将该位置 1。通过软件向该位写入 1，或读取相应的 ADC_JDRy 寄存器都可将该位清零。
0：注入通道转换未完成（或标志事件已通过软件确认并清零）
1：注入通道转换已完成

位 4 **OVR**：ADC 溢出 (ADC overrun)

该位在常规通道上发生溢出事件时由硬件置 1，这意味着在 EOC 标志已置 1 时，新转换已完成。通过软件写入 1 可将该位清零。

0：未发生溢出（或标志事件已通过软件确认并清零）

1：发生溢出

位 3 **EOS**：常规序列结束标志 (End of regular sequence flag)

常规通道序列转换结束后，硬件将该位置 1。通过软件写入 1 可将该位清零。

0：常规转换序列未完成（或标志事件已通过软件确认并清零）

1：常规转换序列已完成

位 2 **EOC**：转换结束标志 (End of conversion flag)

当通道的每次常规转换结束，新数据出现在 ADC_DR 寄存器时，会通过硬件将该位置 1。通过软件向该位写入 1，或读取 ADC_DR 寄存器都可将该位清零。

0：常规通道转换未完成（或标志事件已通过软件确认并清零）

1：常规通道转换已完成

位 1 **EOSMP**：采样结束标志 (End of sampling flag)

在任何通道（仅限常规通道）的转换过程中，当采样阶段结束时，会通过硬件将该位置 1。

0：采样阶段未结束（或标志事件已通过软件确认并清零）

1：采样阶段已结束

位 0 **ADRDY**：ADC 就绪 (ADC ready)

ADC 使能后 (ADEN = 1) 以及 ADC 达到准备好接收转换请求的状态时，会通过硬件将该位置 1。通过软件写入 1 可将该位清零。

0：ADC 未准备好开始转换（或标志事件已通过软件确认并清零）

1：ADC 已准备好开始转换

19.6.2 ADC 中断使能寄存器 (ADC_IER)

ADC interrupt enable register

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	JQOVFIE	AWD3IE	AWD2IE	AWD1IE	JEOSIE	JEOCIE	OVRIE	EOSIE	EOCIE	EOSMP IE	ADRDY IE
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:11 保留，必须保持复位值。

位 10 **JQOVFIE**：注入上下文队列溢出中断使能 (Injected context queue overflow interrupt enable)

该位由软件置 1 和清零，用于使能/禁止注入上下文队列溢出中断。

0：禁止注入上下文队列溢出中断

1：使能注入上下文队列溢出中断JQOVF 位置 1 时产生中断。

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对该位执行写操作。

位 9 **AWD3IE**：模拟看门狗 3 中断使能 (Analog watchdog 3 interrupt enable)

该位由软件置 1 和清零，用于使能/禁止模拟看门狗 2 中断。

0：禁止模拟看门狗 3 中断

1：使能模拟看门狗 3 中断

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作。

位 8 **AWD2IE**：模拟看门狗 2 中断使能 (Analog watchdog 2 interrupt enable)

该位由软件置 1 和清零，用于使能/禁止模拟看门狗 2 中断。

0：禁止模拟看门狗 2 中断

1：使能模拟看门狗 2 中断

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作。

位 7 **AWD1IE**：模拟看门狗 1 中断使能 (Analog watchdog 1 interrupt enable)

该位由软件置 1 和清零，用于使能/禁止模拟看门狗 1 中断。

0：禁止模拟看门狗 1 中断

1：使能模拟看门狗 1 中断

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作。

位 6 **JEOSIE**：注入转换序列结束中断使能 (End of injected sequence of conversions interrupt enable)

该位由软件置 1 和清零，用于使能/禁止注入转换序列结束中断。

0：禁止 JEOS 中断

1：使能 JEOS 中断。JEOS 位置 1 时产生中断。

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对该位执行写操作。

位 5 **JEOCIE**：注入转换结束中断使能 (End of injected conversion interrupt enable)

该位由软件置 1 和清零，用于使能/禁止注入转换结束中断。

0：禁止 JEOC 中断。

1：使能 JEOC 中断。JEOC 位置 1 时产生中断。

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对该位执行写操作。

位 4 **OVRIE**：溢出中断使能 (Overrun interrupt enable)

该位由软件置 1 和清零，用于使能/禁止常规转换的溢出中断。

0：禁止溢出中断

1：使能溢出中断。OVR 位置 1 时产生中断。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对该位执行写操作。

位 3 **EOSIE**：常规转换序列结束中断使能 (End of regular sequence of conversions interrupt enable)

该位由软件置 1 和清零，用于使能/禁止常规转换序列结束中断。

0：禁止 EOS 中断

1：使能 EOS 中断EOS 位置 1 时产生中断。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对该位执行写操作。

位 2 **EOCIE**：常规转换结束中断使能 (End of regular conversion interrupt enable)

该位由软件置 1 和清零，用于使能/禁止常规转换结束中断。

0：禁止 EOC 中断。

1：使能 EOC 中断EOC 位置 1 时产生中断。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对该位执行写操作。

位 1 **EOSMPIE**：常规转换采样结束中断使能 (End of sampling flag interrupt enable for regular conversions)

该位由软件置 1 和清零，用于使能/禁止常规转换采样阶段结束中断。

0：禁止 EOSMP 中断。

1：使能 EOSMP 中断。EOSMP 位置 1 时产生中断。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对该位执行写操作。

位 0 **ADRDYIE**：ADC 就绪中断使能 (ADC ready interrupt enable)

该位由软件置 1 和清零，用于使能/禁止 ADC 就绪中断。

0：禁止 ADRDY 中断

1：使能 ADRDY 中断。ADRDY 位置 1 时产生中断。

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作

19.6.3 ADC 控制寄存器 (ADC_CR)

ADC control register

偏移地址：0x08

复位值：0x2000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAL	ADCALDIF	DEEPPWD	ADVREGEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
rs	rw	rw	rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JADSTP	ADSTP	JADSTART	ADSTART	ADDIS	ADEN
										rs	rs	rs	rs	rs	rs

位 31 **ADCAL**：ADC 校准 (ADC calibration)

该位由软件置 1，用于开始 ADC 校准。先对 ADCALDIF 位编程，以确定此校准应用于单端输入模式还是差分输入模式。

校准完成后，该位由硬件清零。

0：校准已完成

1：写入 1 可校准 ADC。读取值为 1 表示正在进行校准。

注意：仅当 ADEN = 0 时，才允许通过软件将 ADCAL 置 1 的方式启动校准。

仅当 ADEN = 1、ADSTART = 0 且 JADSTART = 0（ADC 已使能，当前未进行任何转换）时，才允许通过软件对 ADC_CALFACT 执行写操作来更新校准系数。

位 30 **ADCALDIF**：校准的差分模式 (Differential mode for calibration)

该位由软件置 1 和清零，用于为校准配置单端输入模式或差分输入模式。

0：写入 ADCAL 会在单端输入模式下启动校准。

1：写入 ADCAL 会在差分输入模式下启动校准。

注意：仅当 ADC 已禁止、且当前未进行校准时（ADCAL = 0、JADSTART = 0、JADSTP = 0、ADSTART = 0、ADSTP = 0、ADDIS = 0 且 ADEN = 0），才允许通过软件对该位执行写操作。

位 29 **DEEPPWD**：深度掉电使能 (Deep-power-down enable)

该位由软件置 1 和清零，用于使 ADC 进入深度掉电模式。

0：ADC 未处于深度掉电模式

1：ADC 处于深度掉电模式（默认复位状态）

注意：仅当 ADC 已禁止时（ADCAL = 0、JADSTART = 0、JADSTP = 0、ADSTART = 0、ADSTP = 0、ADDIS = 0 且 ADEN = 0），才允许通过软件对该位执行写操作。

位 28 **ADVREGEN**：ADC 调压器使能 (ADC voltage regulator enable)

该位由软件置 1，用于使能 ADC 调压器。

执行启动校准或使能 ADC 等操作之前，必须先使能 ADC 调压器，并且软件必须等待调压器启动时间。

0：禁止 ADC 调压器

1：使能 ADC 调压器。

更多关于 ADC 调压器使能和禁止序列的详细信息，请参见*第 19.4.6 节：ADC 深度掉电模式 (DEEPPWD) 和 ADC 调压器 (ADVREGEN)*。

仅当 ADC 已禁止时（ADCAL = 0、JADSTART = 0、ADSTART = 0、ADSTP = 0、ADDIS = 0 且 ADEN = 0），才能通过软件对该位域进行编程。

位 27:6 保留，必须保持复位值。

位 5 **JADSTP**：ADC 停止注入转换命令 (ADC stop of injected conversion command)

该位由软件置 1，用于停止和丢弃正在进行的注入转换（JADSTP 命令）。

当转换已有效丢弃、并且可重新配置 ADC 注入序列和触发时，会通过硬件将该位清零。随后，ADC 会准备好接收新的开始注入转换命令（JADSTART 命令）。

0：当前未执行 ADC 停止注入转换命令

1：写入 1 可停止正在进行的注入转换。读取值为 1 表示正在执行 ADSTP 命令。

注意：仅当 JJADSTART = 1 且 ADDIS = 0 时（ADC 已使能、最终会进行注入转换、并且没有任何待处理的禁止 ADC 的请求），才允许通过软件将 JADSTP 置 1

在自动注入模式下 (JAUTO = 1)，将 ADSTP 位置 1 会中止常规转换和注入转换（不要使用 JADSTP）。

位 4 **ADSTP**：ADC 停止常规转换命令 (ADC stop of regular conversion command)

该位由软件置 1，用于停止和丢弃正在进行的常规转换（ADSTP 命令）。

当转换已有效丢弃、并且可重新配置 ADC 常规序列和触发时，会通过硬件将该位清零。随后，ADC 会准备好接收新的开始常规转换命令（ADSTART 命令）。

0：当前未执行 ADC 停止常规转换命令

1：写入 1 可停止正在进行的常规转换。读取值为 1 表示正在执行 ADSTP 命令。

注意：仅当 ADSTART = 1 且 ADDIS = 0 时（ADC 已使能、最终会进行常规转换、并且没有任何待处理的禁止 ADC 的请求），才允许通过软件将 ADSTP 置 1。

在自动注入模式下 (JAUTO = 1)，将 ADSTP 位置 1 会中止常规转换和注入转换（不要使用 JADSTP）。

位 3 **JADSTART**：ADC 开始注入转换 (ADC start of injected conversion)

该位由软件置 1，用于开始 ADC 的注入通道转换。根据配置位 JEXTEN 的值，可以立即开始转换（软件触发配置），也可以在发生注入硬件触发事件后开始转换（硬件触发配置）。

该位通过硬件清零：

– 在单次转换模式下，如果选择了软件触发 (JEXTSEL = 0x0)：出现注入转换序列结束 (JEOS) 标志时清零。

– 在所有情况下：执行完 JADSTP 命令后，由硬件将 JADSTP 位清零的同时清零。

0：当前未进行 ADC 注入转换。

1：写入 1 可开始注入转换。读取值为 1 表示 ADC 正在运行，最终会转换注入通道。

注意：仅当 ADEN = 1 且 ADDIS = 0 时（ADC 已使能，并且没有任何待处理的禁止 ADC 的请求），才允许通过软件将 JADSTART 置 1。

在自动注入模式下 (JAUTO = 1)，通过将 ADSTART 位置 1 开始常规转换和自动注入转换（JADSTART 必须保持清零）

位 2 **ADSTART**：ADC 开始常规转换 (ADC start of regular conversion)

该位由软件置 1，用于开始 ADC 的常规通道转换。根据配置位 EXTEN 的值，可以立即开始转换（软件触发配置），也可以在发生常规硬件触发事件后开始转换（硬件触发配置）。

该位通过硬件清零：

– 在单次转换模式下，如果选择了软件触发 (EXTSEL = 0x0)：出现常规转换序列结束 (EOS) 标志时清零。

– 在所有情况下：执行完 ADSTP 命令后，由硬件将 ADSTP 位清零的同时清零。

0：当前未进行 ADC 常规转换。

1：写入 1 可开始常规转换。读取值为 1 表示 ADC 正在运行，最终会转换常规通道。

注意：仅当 ADEN = 1 且 ADDIS = 0 时（ADC 已使能，并且没有任何待处理的禁止 ADC 的请求），才允许通过软件将 ADSTART 置 1。

在自动注入模式下 (JAUTO = 1)，通过将 ADSTART 位置 1 开始常规转换和自动注入转换（JADSTART 必须保持清零）

位 1 **ADDIS**：ADC 禁止命令 (ADC disable command)

该位由软件置 1，用于禁止 ADC（ADDIS 命令）并使其进入掉电状态（OFF 状态）。

ADC 已有效禁止后，会立即通过硬件将该位清零（此时也会通过硬件将 ADEN 清零）。

0：当前未执行 ADDIS 命令

1：写入 1 可禁止 ADC。读取值为 1 表示正在执行 ADDIS 命令。

注意：仅当 ADEN = 1、ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件将 ADDIS 置 1

位 0 **ADEN**：ADC 使能控制 (ADC enable control)

该位由软件置 1，用于使能 ADC。ADRDY 标志置 1 后，ADC 将立即准备好运行。

如果 ADC 已禁止，则执行 ADDIS 命令后，将通过硬件对该位清零。

0：禁止 ADC（OFF 状态）

1：写入 1 可使能 ADC。

注意：仅当 ADC_CR 寄存器的所有位均为 0 时（ADCAL = 0、JADSTART = 0、ADSTART = 0、ADSTP = 0、ADDIS = 0 且 ADEN = 0，但 ADVREGEN 位除外，该位必须为 1），才允许通过软件将 ADEN 置 1（并且软件必须等待调压器的启动时间）

19.6.4 ADC 配置寄存器 (ADC_CFGR)

ADC configuration register

偏移地址：0x0C

复位值：0x8000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
JQDIS	AWD1CH[4:0]					JAUTO	JAWD1EN	AWD1EN	AWD1SGL	JQM	JDISCEN	DISCNUM[2:0]			DISCEN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALIGN	AUTDLY	CONT	OVRMOD	EXTEN[1:0]		EXTSEL4	EXTSEL3	EXTSEL2	EXTSEL1	EXTSEL0	RES[1:0]		Res.	DMACFG	DMAEN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw

位 31 **JQDIS**：注入队列禁止 (Injected queue disable)

这些位由软件置 1 和清零，用于禁止注入队列机制：

0：使能注入队列

1：禁止注入队列

注意：仅当 AADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行常规转换、也未进行注入转换），才允许通过软件对该位执行写操作。

将 JQDIS 位置 1 或复位会导致注入队列被清空，JSQR 寄存器也会被清空。

位 30:26 **AWD1CH[4:0]**：模拟看门狗 1 通道选择 (Analog watchdog 1 channel selection)

这些位将由软件置 1 和清零。它们用于选择由模拟看门狗监控的输入通道。

00000：通过 AWD1 监控 ADC 模拟输入通道 0

00001：通过 AWD1 监控 ADC 模拟输入通道 1

.....

10011：通过 AWD1 监控 ADC 模拟输入通道 19

其他：保留，不得使用

注意：一些通道未进行物理连接。将相应的 AWD1CH[4:0] 设置保持为复位值。

通过 AWD1CH 选择的通道必须也在 SQRi 或 JSQRi 寄存器中选择。

仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 25 **JAUTO**：注入组自动转换 (Automatic injected group conversion)

通过软件将该位置 1 和清零可在常规组转换后分别使能/禁止注入组自动转换。

0：禁止注入组自动转换

1：使能注入组自动转换

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行常规转换、也未进行注入转换），才允许通过软件对该位执行写操作。

位 24 **JAWD1EN**：注入通道上的模拟看门狗 1 使能 (Analog watchdog 1 enable on injected channels)

该位由软件置 1 和清零

0：在注入通道上禁止模拟看门狗 1

1：在注入通道上使能模拟看门狗 1

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对该位执行写操作。

位 23 **AWD1EN**：常规通道上的模拟看门狗 1 使能 (Analog watchdog 1 enable on regular channels)

该位由软件置 1 和清零

0：在常规通道上禁止模拟看门狗 1

1：在常规通道上使能模拟看门狗 1

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对该位执行写操作。

位 22 **AWD1SGL**：在单一通道或所有通道上使能看门狗 1 (Enable the watchdog 1 on a single channel or on all channels)

该位由软件置 1 和清零，用于在通过 AWD1CH[4:0] 位确定的通道或所有通道上使能模拟看门狗。

0：在所有通道上使能模拟看门狗 1

1：在单一通道上使能模拟看门狗 1

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 21 **JQM**：JSQR 队列模式 (JSQR queue mode)

该位由软件置 1 和清零。

该位定义空队列的管理方式。

0：JSQR 模式 0：队列从不为空，并会保留上一次写入 JSQR 的配置。

1：JSQR 模式 1：队列可以为空，队列为空时，会在上一个有效注入序列完成后立即在内部禁止注入序列的软件和硬件触发。

更多信息，请参见*第 19.4.21 节：注入转换的上下文队列*。

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对该位执行写操作。

位 20 **JDISCEN**：注入通道的不连续采样模式 (Discontinuous mode on injected channels)

通过软件将该位置 1 和清零可使能/禁止注入通道的不连续采样模式。

0：禁止注入通道的不连续采样模式

1：使能注入通道的不连续采样模式

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对该位执行写操作。

不能同时使用自动注入模式和不连续模式：当 JAUTO 置 1 时，DISCEN 和 JDISCEN 位必须通过软件保持清零状态。

位 19:17 **DISCNUM[2:0]**：不连续采样模式通道计数 (Discontinuous mode channel count)

这些位由软件写入，用于定义在接收到外部触发后于不连续采样模式下转换的常规通道数。

000：1 个通道

001：2 个通道

...

111：8 个通道

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 16 **DISCEN**：常规通道的不连续模式 (Discontinuous mode for regular channels)

该位由软件置 1 和清零，用于使能/禁止常规通道的不连续模式。

0：禁止常规通道的不连续模式

1：使能常规通道的不连续模式

注意：不能同时使能不连续模式和连续模式：禁止同时将 DISCEN 和 CONT 位置 1。

不能同时使用自动注入模式和不连续模式：当 JAUTO 置 1 时，DISCEN 和 JDISCEN 位必须通过软件保持清零状态。

仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对该位执行写操作。

位 15 **ALIGN**：数据对齐 (Data alignment)

该位由软件置 1 和清零，用于选择右对齐或左对齐。请参见*数据寄存器、数据对齐和偏移（ADC_DR、OFFSET、OFFSET_CH、ALIGN）*一节。

0：右对齐

1：左对齐

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作

位 14 **AUTDLY**：延迟转换模式 (Delayed conversion mode)

该位由软件置 1 和清零，用于使能/禁止自动延迟转换模式。

0：自动延迟转换模式关闭

1：自动延迟转换模式开启

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作

位 13 **CONT**：常规转换的单次/连续转换模式 (Single / continuous conversion mode for regular conversions)

该位由软件置 1 和清零。该位置 1 时，常规转换将持续进行，直到该位清零。

0：单次转换模式

1：连续转换模式

注意：不能同时使能不连续模式和连续模式：禁止同时将 DISCEN 和 CONT 位置 1。

仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对该位执行写操作。

位 12 **OVRMOD**：溢出模式 (Overrun mode)

该位由软件置 1 和清零，用于配置数据溢出的管理方式。

0：如果检测到溢出，ADC_DR 寄存器会保留原有数据。

1：如果检测到溢出，ADC_DR 寄存器会被上一转换结果覆盖。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对该位执行写操作。

位 11:10 **EXTEN[1:0]**：常规通道的外部触发使能和极性选择 (External trigger enable and polarity selection for regular channels)

通过软件将这些位置 1 和清零可选择外部触发极性和使能常规组的触发。

00：禁止硬件触发检测（可通过软件启动转换）

01：在上升沿执行硬件触发检测

10：在下降沿执行硬件触发检测

11：在上升沿和下降沿都执行硬件触发检测

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 9:5 **EXTSEL[4:0]**：常规组的外部触发选择 (External trigger selection for regular group)

这些位可选择用于触发常规组转换的外部事件。

00000：adc_ext_trg0
00001：adc_ext_trg1
00010：adc_ext_trg2
00011：adc_ext_trg3
00100：adc_ext_trg4
00101：adc_ext_trg5
00110：adc_ext_trg6
00111：adc_ext_trg7
...
11111：adc_ext_trg31

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 4:3 **RES[1:0]**：数据分辨率 (Data resolution)

通过软件写入这些位可选择转换的分辨率。

00：12 位
01：10 位
10：8 位
11：6 位

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 2 保留，必须保持复位值。

位 1 **DMACFG**：直接存储器访问配置 (Direct memory access configuration)

该位由软件置 1 和清零，用于在 DMA 两种工作模式之间进行选择，仅当 DMAEN = 1 时，该位才有效。

0：选择 DMA 单次模式
1：选择 DMA 循环模式

更多详细信息，请参见*使用 DMA 管理转换*一节

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作

位 0 **DMAEN**：直接存储器访问使能 (Direct memory access enable)

该位由软件置 1 和清零，用于使能 DMA 请求的生成。这样便可使用 DMA 自动管理转换的数据。更多详细信息，请参见*使用 DMA 管理转换*一节。

0：禁止 DMA
1：使能 DMA

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作

19.6.5 ADC 配置寄存器 2 (ADC_CFGR2)

ADC configuration register 2

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	SMPTRIG	BULB	SWTRIG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
				rw	rw	rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	ROVSM	TROVS	OVSS[3:0]				OVSR[2:0]			JOVSE	ROVSE
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27 **SMPTRIG**：采样时间控制触发模式 (Sampling time control trigger mode)

该位由软件置 1 和清零，用于使能采样时间控制触发模式。

0：禁止采样时间控制触发模式

1：使能采样时间控制触发模式

采样时间始于触发上升沿，转换始于触发下降沿。

EXTEN 位应设置为 01。当 SMPTRIG 位置 1 时，BULB 位不得置 1。

当 EXTEN 位设置为 00 时，将 SWTRIG 置 1 可以开始采样，将 SWTRIG 位清零可以开始转换。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作。

位 26 **BULB**：Bulb 采样模式 (Bulb sampling mode)

该位由软件置 1 和清零，用于使能 Bulb 采样模式。

0：禁止 Bulb 采样模式

1：使能 Bulb 采样模式。采样周期在上一次转换结束后立即开始。

如果 BULB 位置 1，则 SAMPTRIG 位不得置 1。

执行第一次 ADC 转换时，采样时间在 SMPx 位中设定。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作。

位 25 **SWTRIG**：采样时间控制触发模式的软件触发位 (Software trigger bit for sampling time control trigger mode)

该位由软件置 1 和清零，用于使能 Bulb 采样模式。

0：在采样时间控制触发模式下软件触发启动转换

1：在采样时间控制触发模式下软件触发启动采样

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作。

位 24:17 保留，必须保持复位值。

位 16:11 保留，必须保持复位值。

位 10 **ROVSM**：常规过采样模式 (Regular oversampling mode)

该位由软件置 1 和清零，用于选择常规过采样模式。

0：连续模式：如果触发注入转换，过采样会暂时停止，并会在注入序列完成后继续（注入序列过程中会保留过采样缓冲区）

1：恢复模式：如果触发注入转换，当前过采样会中止，并会在注入序列完成后从头开始进行过采样（注入序列开始后，过采样缓冲区会清零）

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作。

位 9 **TROVS**：已触发常规过采样 (Triggered Regular oversampling)

该位由软件置 1 和清零，以使能已触发过采样

0：会在触发后连续完成某一通道的所有过采样转换

1：某一通道的每个过采样转换都需要重新触发

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作。

位 8:5 **OVSS[3:0]**：过采样移位 (Oversampling shift)

该位域由软件置 1 和清零，用于定义应用到原始过采样结果的右移位数。

0000：不移位

0001：移 1 位

0010：移 2 位

0011：移 3 位

0100：移 4 位

0101：移 5 位

0110：移 6 位

0111：移 7 位

1000：移 8 位

其他代码保留

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 4:2 **OVSR[2:0]**：过采样率 (Oversampling ratio)

该位域由软件置 1 和清零，用于定义过采样率。

000：2x

001：4x

010：8x

011：16x

100：32x

101：64x

110：128x

111：256x

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 1 **JOVSE**：注入过采样使能 (Injected oversampling Enable)

该位由软件置 1 和清零，用于使能注入过采样。

0：禁止注入过采样

1：使能注入过采样

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作

位 0 **ROVSE**：常规过采样使能 (Regular oversampling Enable)

该位由软件置 1 和清零，用于使能常规过采样。

0：禁止常规过采样

1：使能常规过采样

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作

19.6.6 ADC 采样时间寄存器 1 (ADC_SMPR1)

ADC sample time register 1

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SMPPLUS	Res.	SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:1]	
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP5[0]	SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **SMPPLUS**：采样时间加一个时钟周期 (Addition of one clock cycle to the sampling time)

1：对于 ADC_SMPR1 和 ADC_SMPR2 寄存器，2.5 个 ADC 时钟周期的采样时间变为 3.5 个 ADC 时钟周期。

0：采样时间始终设置为 2.5 个 ADC 时钟周期

为确保当前未执行任何转换，仅在 ADSTART = 0 且 JADSTART = 0 时才允许通过软件对该位执行写操作。

位 30 保留，必须保持复位值。

位 29:0 **SMPx[2:0]**：通道 x 采样时间选择 (Channel x sampling time selection)（x = 9 到 0）

通过软件写入这些位可分别为各个通道选择采样时间。在采样周期期间，通道选择位必须保持不变。

000：2.5 个 ADC 时钟周期

001：6.5 个 ADC 时钟周期

010：12.5 个 ADC 时钟周期

011：24.5 个 ADC 时钟周期

100：47.5 个 ADC 时钟周期

101：92.5 个 ADC 时钟周期

110：247.5 个 ADC 时钟周期

111：640.5 个 ADC 时钟周期

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

一些通道未进行物理连接。将相应的 SMPx[2:0] 设置保持为复位值。

19.6.7 ADC 采样时间寄存器 2 (ADC_SMPR2)

ADC sample time register 2

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SMP19[2:0]			SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:1]	
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP15[0]	SMP14[2:0]			SMP13[2:0]			SMP12[2:0]			SMP11[2:0]			SMP10[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:30 保留，必须保持复位值。

位 29:0 **SMPx[2:0]**：通道 x 采样时间选择 (Channel x sampling time selection)（x = 19 到 10）

通过软件写入这些位可分别为各个通道选择采样时间。在采样周期期间，通道选择位必须保持不变。

000：2.5 个 ADC 时钟周期
001：6.5 个 ADC 时钟周期
010：12.5 个 ADC 时钟周期
011：24.5 个 ADC 时钟周期
100：47.5 个 ADC 时钟周期
101：92.5 个 ADC 时钟周期
110：247.5 个 ADC 时钟周期
111：640.5 个 ADC 时钟周期

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

一些通道未进行物理连接。将相应的 SMPx[2:0] 设置保持为复位值。

19.6.8 ADC 看门狗阈值寄存器 1 (ADC_TR1)

ADC watchdog threshold register 1

偏移地址：0x20

复位值：0x0FFF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	HT1[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	AWDFILT[2:0]			LT1[11:0]											
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27:16 **HT1[11:0]**：模拟看门狗 1 阈值上限 (Analog watchdog 1 higher threshold)

这些位由软件写入，用于定义模拟看门狗 1 的阈值上限。

请参见*第 19.4.28 节：模拟窗口看门狗（AWD1EN、JAWD1EN、AWD1SGL、AWD1CH、AWD2CH、AWD3CH、AWD_HTx、AWD_LTx、AWDx）*

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 15 保留，必须保持复位值。

位 14:12 **AWDFILT[2:0]**：模拟看门狗滤波参数 (Analog watchdog filtering parameter)

该位由软件置 1 和清零。

000：不滤波
001：两次连续检测会生成一个 AWDx 标志或中断
...
111：八次连续检测会生成一个 AWDx 标志或中断

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作。

位 11:0 **LT1[11:0]**：模拟看门狗 1 阈值下限 (Analog watchdog 1 lower threshold)
这些位由软件写入，用于定义模拟看门狗 1 的阈值下限。
请参见*第 19.4.28 节：模拟窗口看门狗（AWD1EN、JAWD1EN、AWD1SGL、AWD1CH、AWD2CH、AWD3CH、AWD_HTx、AWD_LTx、AWDx）*
注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

19.6.9 ADC 看门狗阈值寄存器 2 (ADC_TR2)

ADC watchdog threshold register 2

偏移地址：0x24

复位值：0x00FF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HT2[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LT2[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23:16 **HT2[7:0]**：模拟看门狗 2 阈值上限 (Analog watchdog 2 higher threshold)
这些位由软件写入，用于定义模拟看门狗 2 的阈值上限。
请参见*第 19.4.28 节：模拟窗口看门狗（AWD1EN、JAWD1EN、AWD1SGL、AWD1CH、AWD2CH、AWD3CH、AWD_HTx、AWD_LTx、AWDx）*
注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 15:8 保留，必须保持复位值。

位 7:0 **LT2[7:0]**：模拟看门狗 2 阈值下限（Analog watchdog 2 lower threshold）
这些位由软件写入，用于定义模拟看门狗 2 的阈值下限。
请参见*第 19.4.28 节：模拟窗口看门狗（AWD1EN、JAWD1EN、AWD1SGL、AWD1CH、AWD2CH、AWD3CH、AWD_HTx、AWD_LTx、AWDx）*
注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

19.6.10 ADC 看门狗阈值寄存器 3 (ADC_TR3)

ADC watchdog threshold register 3

偏移地址：0x28

复位值：0x00FF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HT3[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LT3[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23:16 **HT3[7:0]**：模拟看门狗 3 阈值上限 (Analog watchdog 3 higher threshold)

这些位由软件写入，用于定义模拟看门狗 3 的阈值上限。

请参见*第 19.4.28 节：模拟窗口看门狗（AWD1EN、JAWD1EN、AWD1SGL、AWD1CH、AWD2CH、AWD3CH、AWD_HTx、AWD_LTx、AWDx）*

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 15:8 保留，必须保持复位值。

位 7:0 **LT3[7:0]**：模拟看门狗 3 阈值下限 (Analog watchdog 3 lower threshold)

这些位由软件写入，用于定义模拟看门狗 3 的阈值下限。

此看门狗将 LT3 的 8 位与转换后数据的高 8 位进行比较。

注意：仅当 AADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

19.6.11 ADC 常规序列寄存器 1 (ADC_SQR1)

ADC regular sequence register 1

偏移地址：0x30

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	SQ4[4:0]					Res.	SQ3[4:0]					Res.	SQ2[4]
			rw	rw	rw	rw	rw		rw	rw	rw	rw	rw		rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ2[3:0]				Res.	SQ1[4:0]					Res.	Res.	L[3:0]			
rw	rw	rw	rw		rw	rw	rw	rw	rw			rw	rw	rw	rw

位 31:29 保留，必须保持复位值。

位 28:24 **SQ4[4:0]**：常规序列中的第四次转换 (4th conversion in regular sequence)

通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第四次转换。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 23 保留，必须保持复位值。

位 22:18 **SQ3[4:0]**：常规序列中的第三次转换 (3rd conversion in regular sequence)

通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第三次转换。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 17 保留，必须保持复位值。

位 16:12 **SQ2[4:0]**：常规序列中的第二次转换 (2nd conversion in regular sequence)

通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第二次转换。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 11 保留，必须保持复位值。

位 10:6 **SQ1[4:0]**：常规序列中的第一次转换 (1st conversion in regular sequence)

通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第一次转换。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 5:4 保留，必须保持复位值。

位 3:0 **L[3:0]**：常规通道序列长度（Regular channel sequence length）

通过软件写入这些位可定义常规通道转换序列中的转换总数。

0000：1 次转换

0001：2 次转换

...

1111：16 次转换

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

注意：一些通道未进行物理连接，不得选择这样的通道进行转换。

19.6.12 ADC 常规序列寄存器 2 (ADC_SQR2)

ADC regular sequence register 2

偏移地址：0x34

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	SQ9[4:0]					Res.	SQ8[4:0]					Res.	SQ7[4]
			rw	rw	rw	rw	rw		rw	rw	rw	rw	rw		rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ7[3:0]				Res.	SQ6[4:0]					Res.	SQ5[4:0]				
rw	rw	rw	rw		rw	rw	rw	rw	rw		rw	rw	rw	rw	rw

位 31:29 保留，必须保持复位值。

位 28:24 **SQ9[4:0]**：常规序列中的第九次转换 (9th conversion in regular sequence)

通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第九次转换。

注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 23 保留，必须保持复位值。

位 22:18 **SQ8[4:0]**：常规序列中的第八次转换 (8th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第八次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 17 保留，必须保持复位值。

位 16:12 **SQ7[4:0]**：常规序列中的第七次转换 (7th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第七次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 11 保留，必须保持复位值。

位 10:6 **SQ6[4:0]**：常规序列中的第六次转换 (6th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第六次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 5 保留，必须保持复位值。

位 4:0 **SQ5[4:0]**：常规序列中的第五次转换 (5th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第五次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

注意：一些通道未进行物理连接，不得选择这样的通道进行转换。

19.6.13 ADC 常规序列寄存器 3 (ADC_SQR3)

ADC regular sequence register 3

偏移地址：0x38

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	SQ14[4:0]					Res.	SQ13[4:0]					Res.	SQ12[4]
			rw	rw	rw	rw	rw		rw	rw	rw	rw	rw		rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ12[3:0]				Res.	SQ11[4:0]					Res.	SQ10[4:0]				
rw	rw	rw	rw		rw	rw	rw	rw	rw		rw	rw	rw	rw	rw

位 31:29 保留，必须保持复位值。

位 28:24 **SQ14[4:0]**：常规序列中的第十四次转换 (14th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第十四次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 23 保留，必须保持复位值。

位 22:18 **SQ13[4:0]**：常规序列中的第十三次转换 (13th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第十三次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 17 保留，必须保持复位值。

位 16:12 **SQ12[4:0]**：常规序列中的第十二次转换 (12th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第十二次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 11 保留，必须保持复位值。

位 10:6 **SQ11[4:0]**：常规序列中的第十一次转换 (11th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第十一次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 5 保留，必须保持复位值。

位 4:0 **SQ10[4:0]**：常规序列中的第十次转换 (10th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第十次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

注意：一些通道未进行物理连接，不得选择这样的通道进行转换。

19.6.14 ADC 常规序列寄存器 4 (ADC_SQR4)

ADC regular sequence register 4

偏移地址：0x3C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	SQ16[4:0]					Res.	SQ15[4:0]				
					rw	rw	rw	rw	rw		rw	rw	rw	rw	rw

位 31:11 保留，必须保持复位值。

位 10:6 **SQ16[4:0]**：常规序列中的第十六次转换 (16th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第十六次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

位 5 保留，必须保持复位值。

位 4:0 **SQ15[4:0]**：常规序列中的第十五次转换 (15th conversion in regular sequence)
通过软件写入这些位，并将通道编号（0 到 19）分配为常规转换序列中的第十五次转换。
注意：仅当 ADSTART = 0 时（这可确保当前未进行任何常规转换），才允许通过软件对这些位执行写操作。

注意：一些通道未进行物理连接，不得选择这样的通道进行转换。

19.6.15 ADC 常规数据寄存器 (ADC_DR)

ADC regular data register

偏移地址：0x40

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15:0 **RDATA[15:0]**：已转换的常规数据 (Regular data converted)

这些位为只读。其中包含上一转换常规通道的转换结果。数据有左对齐和右对齐两种方式，如*第 19.4.26 节：数据管理*所述。

19.6.16 ADC 注入序列寄存器 (ADC_JSQR)

ADC injected sequence register

偏移地址：0x4C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
JSQ4[4:0]					Res.	JSQ3[4:0]					Res.	JSQ2[4:1]			
rw	rw	rw	rw	rw		rw	rw	rw	rw	rw		rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JSQ2[0]	Res.	JSQ1[4:0]					JEXTEN[1:0]		JEXTSEL[4:0]					JL[1:0]	
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:27 **JSQ4[4:0]**：注入序列中的第四次转换 (4th conversion in the injected sequence)

通过软件写入这些位，并将通道编号（0 到 19）分配为注入转换序列中的第四次转换。

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对这些位执行写操作。

位 26 保留，必须保持复位值。

位 25:21 **JSQ3[4:0]**：注入序列中的第三次转换 (3rd conversion in the injected sequence)

通过软件写入这些位，并将通道编号（0 到 19）分配为注入转换序列中的第三次转换。

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对这些位执行写操作。

位 20 保留，必须保持复位值。

位 19:15 **JSQ2[4:0]**：注入序列中的第二次转换 (2nd conversion in the injected sequence)

通过软件写入这些位，并将通道编号（0 到 19）分配为注入转换序列中的第二次转换。

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对这些位执行写操作。

位 14 保留，必须保持复位值。

位 13:9 **JSQ1[4:0]**：注入序列中的第一次转换 (1st conversion in the injected sequence)

通过软件写入这些位，并将通道编号（0 到 19）分配为注入转换序列中的第一次转换。

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对这些位执行写操作。

位 8:7 **JEXTEN[1:0]**：注入通道的外部触发使能和极性选择 (External trigger enable and polarity selection for injected channels)

通过软件将这些位置 1 和清零可选择外部触发极性和使能注入组的触发。

00：如果 JQDIS = 0（队列使能），将同时禁止硬件和软件触发检测。否则，禁止队列以及硬件触发检测（可通过软件启动转换）

01：在上升沿执行硬件触发检测

10：在下降沿执行硬件触发检测

11：在上升沿和下降沿都执行硬件触发检测

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对这些位执行写操作。

如果 JQM = 1 且上下文队列为空，则会在内部禁止注入序列的软件和硬件触发（请参见第 19.4.21 节：注入转换的上下文队列）

位 6:2 **JEXTSEL[4:0]**：注入组的外部触发选择 (External Trigger Selection for injected group)

这些位可选择用于触发注入组转换的外部事件：

00000：adc_jext_trg0

00001：adc_jext_trg1

00010：adc_jext_trg2

00011：adc_jext_trg3

00100：adc_jext_trg4

00101：adc_jext_trg5

00110：adc_jext_trg6

00111：adc_jext_trg7

...

11111：adc_jext_trg31

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对这些位执行写操作。

位 1:0 **JL[1:0]**：注入通道序列长度 (Injected channel sequence length)

通过软件写入这些位可定义注入通道转换序列中的转换总数。

00：1 次转换

01：2 次转换

10：3 次转换

11：4 次转换

注意：仅当 JADSTART = 0 时（这可确保当前未进行任何注入转换），才允许通过软件对这些位执行写操作。

注意：一些通道未进行物理连接，不得选择这样的通道进行转换。

19.6.17 ADC 偏移 y 寄存器 (ADC_OFRy)

ADC offset y register

偏移地址：0x60 + 0x04 * (y -1)，（y = 1 到 4）

复位值：0x0000 0000

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

位 31 **OFFSET_EN**：偏移 y 使能 (Offset y enable)

该位由软件写入，用于使能或禁止编程到 OFFSET[11:0] 位中的偏移。

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对该位执行写操作

位 30:26 **OFFSET_CH[4:0]**：数据偏移 y 的通道选择 (Channel selection for the data offset y)

这些位由软件写入，用于定义编程到 OFFSET[11:0] 位中的偏移将应用于哪个通道。

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

一些通道未进行物理连接，不得选择这样的通道用于数据偏移 y。

如果 OFFSET_EN 置 1，则不允许为不同的 ADC_OFRy 寄存器选择同一个通道。

位 25 **SATEN**：饱和使能 (Saturation enable)

该位由软件置 1 和清零，用于使能偏移功能在 0x000 和 0xFFF 的饱和。

0：无饱和控制，偏移结果可以是有符号

1：使能饱和，偏移结果为无符号，在 0x000 和 0xFFF 饱和

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 24 **OFFSETPOS**：正偏移 (Positive offset)

该位由软件置 1 和清零，用于使能正偏移。

0：负偏移

1：正偏移

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

位 23:12 保留，必须保持复位值。

位 11:0 **OFFSET[11:0]**：编程到 OFFSET_CH[4:0] 位中的通道对应的数据偏移 y (Data offset y for the channel programmed into bits OFFSET_CH[4:0])

这些位由软件写入，用于定义在转换通道（可以是常规通道或注入通道）时从原始转换数据中减去的偏移。应用数据偏移的通道必须在 OFFSET_CH[4:0] 位中编程。转换结果可从 ADC_DR（常规转换）或 ADC_JDRyi 寄存器（注入转换）中读取。

注意：仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

如果多个偏移 (OFFSET) 指向同一通道，相减时只会考虑使用 x 值最小的偏移。

例如：如果 OOFFSET1_CH[4:0] = 4 且 OFFSET2_CH[4:0] = 4，转换通道 4 时会减去 OFFSET1[11:0]。

19.6.18 ADC 注入通道 y 数据寄存器 (ADC_JDRy)

ADC injected channel y data register

偏移地址：0x80 + 0x04 * (y - 1)，（y = 1 到 4）

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15:0 **JDATA[15:0]**：注入数据 (Injected data)

这些位为只读。它们包括来自注入通道 y 的转换结果。数据有左对齐和右对齐两种方式，如 *第 19.4.26 节：数据管理* 所述。

19.6.19 ADC 模拟看门狗 2 配置寄存器 (ADC_AWD2CR)

ADC analog watchdog 2 configuration register

偏移地址：0xA0

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWD2CH[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AWD2CH[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **AWD2CH[19:0]**：模拟看门狗 2 通道选择 (Analog watchdog 2 channel selection)

这些位将由软件置 1 和清零。它们用于使能和选择由模拟看门狗 2 监控的输入通道。

AWD2CH[i] = 0：不通过 AWD2 监控 ADC 模拟输入通道 i

AWD2CH[i] = 1：通过 AWD2 监控 ADC 模拟输入通道 i

当 AWD2CH[19:0] = 000..0 时，会禁止模拟看门狗 2

注意：通过 AWD2CH 选择的通道必须也在 SQRi 或 JSQRi 寄存器中选择。

仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

一些通道未进行物理连接，不得选择这样的通道用于模拟看门狗。

19.6.20 ADC 模拟看门狗 3 配置寄存器 (ADC_AWD3CR)

ADC analog watchdog 3 configuration register

偏移地址：0xA4

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWD3CH[19:16]			
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AWD3CH[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **AWD3CH[19:0]**：模拟看门狗 3 通道选择 (Analog watchdog 3 channel selection)

这些位将由软件置 1 和清零。它们用于使能和选择由模拟看门狗 3 监控的输入通道。

AWD3CH[i] = 0：不通过 AWD3 监控 ADC 模拟输入通道 i

AWD3CH[i] = 1：通过 AWD3 监控 ADC 模拟输入通道 i

当 AWD3CH[19:0] = 000..0 时，会禁止模拟看门狗 3

注意：通过 AWD3CH 选择的通道必须也在 SQRi 或 JSQRi 寄存器中选择。

仅当 ADSTART = 0 且 JADSTART = 0 时（这可确保当前未进行任何转换），才允许通过软件对这些位执行写操作。

一些通道未进行物理连接，不得选择这样的通道用于模拟看门狗。

19.6.21 ADC 差分模式选择寄存器 (ADC_DIFSEL)

ADC Differential mode selection register

偏移地址：0xB0

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIFSEL[19:16]			
												rw	rw	rw	rw

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

位 31:20 保留，必须保持复位值。

位 19:0 **DIFSEL[19:0]**：通道 19 到 0 的差分模式 (Differential mode for channels 19 to 0)。

这些位将由软件置 1 和清零。它们用于选择将通道配置为单端模式或差分模式。

DIFSEL[i] = 0：将 ADC 模拟输入通道 i 配置为单端模式

DIFSEL[i] = 1：将 ADC 模拟输入通道 i 配置为差分模式

注意：与连接到单端 I/O 端口或内部通道的通道相对应的 DIFSEL 位必须保持复位值（单端输入模式）。

仅当 ADC 已禁止时（ADCAL = 0、JADSTART = 0、JADSTP = 0、ADSTART = 0、ADSTP = 0、ADDIS = 0 且 ADEN = 0），才允许通过软件对这些位执行写操作。

19.6.22 ADC 校准系数 (ADC_CALFACT)

ADC calibration factors

偏移地址：0xB4

复位值：0x0000 0000

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

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

位 31:23 保留，必须保持复位值。

位 22:16 **CALFACT_D[6:0]**：差分模式下的校准系数 (Calibration Factors in differential mode)

这些位可由硬件或软件写入。

差分输入校准完成后，会立即由硬件更新为校准系数。

软件可向这些位写入新的校准系数。如果新校准系数不同于当前存储于模拟 ADC 中的校准系数，启动新的差分校准后，会立即应用新校准系数。

注意：仅当 ADEN = 1、ADSTART = 0 且 JADSTART = 0 时（ADC 已使能、当前未执行任何校准和转换），才允许通过软件对这些位执行写操作。

位 15:7 保留，必须保持复位值。

位 6:0 **CALFACT_S[6:0]**：单端模式下的校准系数 (Calibration Factors In single-ended mode)

这些位可由硬件或软件写入。

单端输入校准完成后，会立即由硬件更新为校准系数。

软件可向这些位写入新的校准系数。如果新校准系数不同于当前存储于模拟 ADC 中的校准系数，启动新的单端校准后，会立即应用新校准系数。

注意：仅当 ADEN = 1、ADSTART = 0 且 JADSTART = 0 时（ADC 已使能、当前未执行任何校准和转换），才允许通过软件对这些位执行写操作。

19.6.23 ADC 选项寄存器 (ADC_OR)

ADC option register

偏移地址：0xC8

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OP1	OP0
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **OP1**：选项位 1 (Option bit 1)
0：禁止 V_{DDCORE} 通道
1：使能 V_{DDCORE} 通道

位 0 **OP0**：选项位 0
0：禁止 INP0/INN1 GPIO 开关控制
1：使能 INP0/INN1 GPIO 开关控制
注意：如果选择了 ADC1_INP0 或 ADC1_INN1 通道，则该选项位必须置 1。

注意：在一些 ADC 实例上，ADC_OR 寄存器可能会被保留。请参见第 19.3 节：ADC 实现。

19.7 ADC 通用寄存器

19.7.1 ADC 通用控制寄存器 (ADC_CCR)

ADC common control register

偏移地址：0x308

复位值：0x0000 0000

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

位 31:25 保留，必须保持复位值。

位 24 **VBATEN**：VBAT 使能 (VBAT enable)
通过软件将该位置 1 和清零以进行控制。
0：禁止 V_{BAT} 通道
1：使能 V_{BAT} 通道

位 23 **TSEN**：V_{SENSE} 使能 (V_{SENSE} enable)
通过软件将该位置 1 和清零以控制 V_{SENSE}。
0：禁止温度传感器通道
1：使能温度传感器通道

位 22 **VREFEN**：V_{REFINT} 使能 (V_{REFINT} enable)
通过软件将该位置 1 和清零可使能/禁止 V_{REFINT} 通道。
0：禁止 V_{REFINT} 通道
1：使能 V_{REFINT} 通道

位 21:18 **PRESC[3:0]**：ADC 预分频器 (ADC prescaler)

这些位由软件置 1 和清零，用于选择 ADC 的时钟频率。该时钟为所有 ADC 所共用。

0000：输入 ADC 时钟未分频
0001：输入 ADC 时钟 2 分频
0010：输入 ADC 时钟 4 分频
0011：输入 ADC 时钟 6 分频
0100：输入 ADC 时钟 8 分频
0101：输入 ADC 时钟 10 分频
0110：输入 ADC 时钟 12 分频
0111：输入 ADC 时钟 16 分频
1000：输入 ADC 时钟 32 分频
1001：输入 ADC 时钟 64 分频
1010：输入 ADC 时钟 128 分频
1011：输入 ADC 时钟 256 分频
其他：保留

注意：仅当 ADC 已禁止时（ADCAL = 0、JADSTART = 0、ADSTART = 0、ADSTP = 0、ADDIS = 0 且 ADEN = 0），才允许通过软件对这些位执行写操作。仅当 CKMODE[1:0] = 0b00 时，才会应用 ADC 预分频值。

位 17:16 **CKMODE[1:0]**：ADC 时钟模式 (ADC clock mode)

这些位由软件置 1 和清零，用于定义 ADC 时钟方案（主 ADC 和从 ADC 共用）：

00：adc_ker_ck (x = 1)（异步时钟模式），在产品级生成（请参见*第 6 节：复位和时钟控制 (RCC)*）

01：adc_hclk/1（同步时钟模式）。仅当 AHB 时钟预分频器设为 1（RCC_CFGR 寄存器中的 HPRE[3:0] = 0XXX）且系统时钟占空比为 50% 时，才必须使能此配置。

10：adc_hclk/2（同步时钟模式）

11：adc_hclk/4（同步时钟模式）

在所有同步时钟模式下，从定时器触发到转换开始的延迟过程中，不存在抖动。

注意：仅当 ADC 已禁止时（ADCAL = 0、JADSTART = 0、ADSTART = 0、ADSTP = 0、ADDIS = 0 且 ADEN = 0），才允许通过软件对这些位执行写操作。

位 15:0 保留，必须保持复位值。

19.7.2 ADC 硬件配置寄存器 (ADC_HWCFGR0)

ADC hardware configuration register

偏移地址：0x3F0

复位值：0x0000 1211

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

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

位 31:16 保留，必须保持复位值。

位 15:12 **IDLEVALUE[3:0]**：未选通道的空闲值 (Idle value for non-selected channels)
0000：空闲通道选择为 0x13
0001：空闲通道选择为 0x1F

位 11:8 **OPBITS[3:0]**：选项位的数量 (Number of option bits)
0000：未实现选项寄存器
0002：在 ADC 选项寄存器 (ADC_OR) 中实现了 2 个选项位（偏移地址为 0xC8）

位 7:4 **MULPIPE[3:0]**：流水线级数 (Number of pipeline stages)
0001：一级流水线

位 3:0 **ADCNUM[3:0]**：实现的 ADC 数 (Number of ADCs implemented)
0001：实现了一个 ADC 实例
0010：实现了两个 ADC 实例
0011：实现了三个 ADC 实例

19.7.3 ADC 版本触发器 (ADC_VERR)

ADC version register

偏移地址：0x3F4

复位值：0x0000 0012

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **MAJREV[3:0]**：主版本 (Major revision)
这些位返回 ADC IP 主版本
0001：主版本 = 1.X

位 3:0 **MINREV[3:0]**：次版本 (Minor revision)
这些位返回 ADC IP 次版本
0001：次版本 = X.1
0002：次版本 = X.2
0003：次版本 = X.3

19.7.4 ADC 标识寄存器 (ADC_IPDR)

ADC identification register

偏移地址：0x3F8

复位值：0x0011 0006

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

位 31:0 **ID[31:0]**：外设标识符 (Peripheral identifier)

这些位返回 ADC 标识符。

ID[31:0] = 0x0011 0006：c7amba_aditf5_90_v1

19.7.5 ADC 大小标识寄存器 (ADC_SIDR)

ADC size identification register

偏移地址：0x3FC

复位值：0xA3C5 DD01

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

位 31:0 **SID[31:0]**：大小标识 (Size Identification)

SID[31:8]：描述 ADC_SIDR 寄存器特性的固定代码。该位域始终读为 0xA3C5DD。

SID[7:0]：只读数字位域，返回标识寄存器与 IP 基址之间的地址偏移 (KB)：

0x01：1 KB 地址偏移

0x02：2 KB 地址偏移

0x04：4 KB 地址偏移

0x08：8 KB 地址偏移

19.8 ADC 寄存器映射

表 119. ADC 全局寄存器映射

偏移	寄存器
0x000 - 0x0B4	主 ADC1/2
0x0B8 - 0x2FC	保留
0x300 - 0x30C	主 ADC 和从 ADC 通用寄存器

表 120. ADC 寄存器映射和复位值

偏移	寄存器名称 复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**ADC_ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JQOVF	AWD3	AWD2	AWD1	JEOS	JEOC	OVR	EOS	EOC	EOSMP	ADRDY
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0x04	**ADC_IER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JQOVFIE	AWD3IE	AWD2IE	AWD1IE	JEOSIE	JEOCIE	OVRIE	EOSIE	EOCIE	EOSMPIE	ADRDYIE
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0x08	**ADC_CR**	ADCAL	ADCALDIF	DEEPPWD	ADVREGEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JADSTP	ADSTP	JADSTART	ADSTART	ADDIS	ADEN
	Reset value	0	0	1	0																							0	0	0	0	0	0
0x0C	**ADC_CFGR**	JQDIS	AWD1CH[4:0]					JAUTO	JAWD1EN	AWD1EN	AWD1SGL	JQM	JDISCEN	DISCNUM[2:0]			DISCEN	ALIGN	AUTDLY	CONT	OVRMOD	EXTEN[1:0]		EXTSEL4	EXTSEL3	EXTSEL2	EXTSEL1	EXTSEL0	RES[1:0]		Res.	DMACFG	DMAEN
	Reset value	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0
0x0C	**ADC_CFGR2**	Res.	Res.	Res.	Res.	SMPTRIG	BULB	SWTRIG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ROVSM	TROVS	OVSS[3:0]					OVSR[2:0]		JOVSE	ROVSE
	Reset value					0	0	0															0	0	0	0	0	0	0	0	0	0	0
0x14	**ADC_SMPR1**	SMPPLUS	Res.	SMP9 [2:0]			SMP8 [2:0]			SMP7 [2:0]			SMP6 [2:0]			SMP5 [2:0]			SMP4 [2:0]			SMP3 [2:0]			SMP2 [2:0]			SMP1 [2:0]			SMP0 [2:0]		
	Reset value	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	**ADC_SMPR2**	Res.	Res.	SMP19 [2:0]			SMP18 [2:0]			SMP17 [2:0]			SMP16 [2:0]			SMP15 [2:0]			SMP14 [2:0]			SMP13 [2:0]			SMP12 [2:0]			SMP11 [2:0]			SMP10 [2:0]		
	Reset value			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	Reserved	Res.																															
0x20	**ADC_TR1**	Res.	Res.	Res.	Res.	HT1[11:0]												Res.	AWDFILT [2:0]			LT1[11:0]												
	Reset value					1	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	**ADC_TR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HT2[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LT2[7:0]							
	Reset value									1	1	1	1	1	1	1	1									0	0	0	0	0	0	0	0
0x28	**ADC_TR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HT3[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LT3[7:0]							
	Reset value									1	1	1	1	1	1	1	1									0	0	0	0	0	0	0	0

表 120. ADC 寄存器映射和复位值（续）

偏移	寄存器名称 复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2C	Reserved	Res.																															
0x30	**ADC_SQR1**	Res.	Res.	Res.	SQ4[4:0]					Res.	SQ3[4:0]					Res.	SQ2[4:0]					Res.	SQ1[4:0]					Res.	Res.	L[3:0]			
	Reset value				0	0	0	0	0		0	0	0	0	0		0	0	0	0	0		0	0	0	0	0			0	0	0	0
0x34	**ADC_SQR2**	Res.	Res.	Res.	SQ9[4:0]					Res.	SQ8[4:0]					Res.	SQ7[4:0]					Res.	SQ6[4:0]					Res.	SQ5[4:0]				
	Reset value				0	0	0	0	0		0	0	0	0	0		0	0	0	0	0		0	0	0	0	0		0	0	0	0	0	0
0x38	**ADC_SQR3**	Res.	Res.	Res.	SQ14[4:0]					Res.	SQ13[4:0]					Res.	SQ12[4:0]					Res.	SQ11[4:0]					Res.	SQ10[4:0]				
	Reset value				0	0	0	0	0		0	0	0	0	0		0	0	0	0	0		0	0	0	0	0		0	0	0	0	0	0
0x3C	**ADC_SQR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SQ16[4:0]					Res.	SQ15[4:0]				
	Reset value																						0	0	0	0	0		0	0	0	0	0
0x40	**ADC_DR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	regular RDATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44- 0x48	Reserved	Res.																															
0x4C	**ADC_JSQR**	JSQ4[4:0]					Res.	JSQ3[4:0]					Res.	JSQ2[4:0]				Res.	JSQ1[4:0]					JEXTEN[1:0]		JEXTSEL [4:0]				JL[1:0]			
	Reset value	0	0	0	0	0		0	0	0	0	0		0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50- 0x5C	Reserved	Res.																															
0x60	**ADC_OFR1**	OFFSET_EN	OFFSET_ CH[4:0]					SATEN	OFFSETPOS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OFFSET[11:0]										
	Reset value	0	0	0	0	0	0	0	0														0	0	0	0	0	0	0	0	0	0	0
0x64	**ADC_OFR2**	OFFSET_EN	OFFSET_ CH[4:0]					SATEN	OFFSETPOS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OFFSET[11:0]										
	Reset value	0	0	0	0	0	0	0	0														0	0	0	0	0	0	0	0	0	0	0
0x68	**ADC_OFR3**	OFFSET_EN	OFFSET_ CH[4:0]					SATEN	OFFSETPOS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OFFSET[11:0]										
	Reset value	0	0	0	0	0	0	0	0														0	0	0	0	0	0	0	0	0	0	0
0x6C	**ADC_OFR4**	OFFSET_EN	OFFSET_ CH[4:0]					SATEN	OFFSETPOS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OFFSET[11:0]										
	Reset value	0	0	0	0	0	0	0	0														0	0	0	0	0	0	0	0	0	0	0
0x70- 0x7C	Reserved	Res.																															
0x80	**ADC_JDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x84	**ADC_JDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA2[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 120. ADC 寄存器映射和复位值（续）

偏移	寄存器名称 复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x88	**ADC_JDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA3[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x8C	**ADC_JDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA4[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x90-0x9C	Reserved	Res.																															
0xA0	**ADC_AWD2CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWD2CH[19:0]																			
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xA4	**ADC_AWD3CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWD3CH[19:0]																			
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xA8-0xAC	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0xB0	**ADC_DIFSEL**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIFSEL[19:0]																			
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xB4	**ADC_CALFACT**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CALFACT_D[6:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CALFACT_S[6:0]							
	Reset value									0	0	0	0	0	0	0	0									0	0	0	0	0	0	0	0
0xB8-0xC4	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0xC8	**ADC_OR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OP1	OP0
	Reset value																															0	0
0xCC-0xFC	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

表 121. ADC 寄存器映射和复位值（主 ADC 和从 ADC 通用寄存器）

偏移	寄存器名称 Reset value	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x304	Reserved	Res.																															
0x308	**ADC_CCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VBATEN	TSEN	VREFEN	PRESC[3:0]				CKMODE[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value								0	0	0	0	0	0	0	0	0																
x30C-0x3EC	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x3F0	**ADC_HWCFGR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDLEVALUE [3:0]				OPBITS[3:0]				MULPIPE[3:0]				ADCNUM[3:0]			
	Reset value	0x0000 1211																															
0x3F4	**ADC_VERR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MAJREV[3:0]				MINREV[3:0]			
	Reset value	0x0000 0012																															
0x3F8	**ADC_IPDR**	ID[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
0x3FC	**ADC_SIDR**	SID[31:0]																															
	Reset value	1	0	1	0	0	0	1	1	1	1	0	0	0	1	0	1	1	1	0	1	1	1	0	1	0	0	0	0	0	0	0	1

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

20 数字温度传感器 (DTS)

20.1 简介

器件内置一个可将温度转换为方波的传感器，其中方波的频率与温度成正比。该频率基于 PCLK 或 LSE 时钟测得。

20.2 DTS 主要特性

温度传感器模块的主要特性如下：

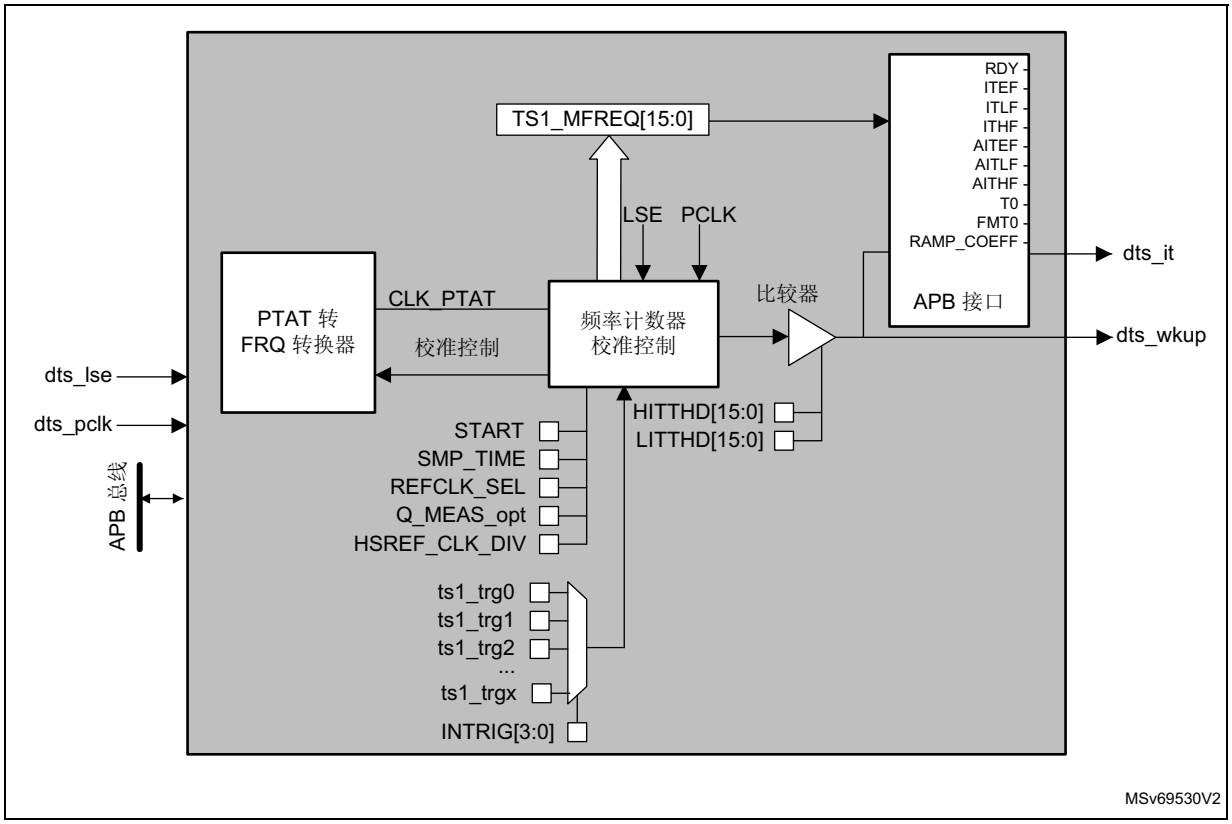
- 测量由软件或 4 个硬件源触发启动
- 可编程采样时间以提高温度测量精度
- 计数器基于 LSE 或 PCLK 时钟进行同步
- 基于阈值下限和阈值上限的温度看门狗
- 可在温度低于或高于预定义阈值时以及测量结束时生成中断
- 可在温度高于/低于预定义阈值时生成异步唤醒信号（仅限 LSE 模式）
- 使用 LSE 时钟实现快速测量

20.3 DTS 功能说明

20.3.1 DTS 框图

温度传感器框图如*图 130* 所示。

图 130. 温度传感器功能框图



20.3.2 DTS 内部信号

表 122. DTS 内部输入/输出信号

信号名称	信号类型	说明
dts_lse	数字输入	LSE 时钟
dts_pclk	数字输入	APB 时钟
dts_it	数字输出	温度传感器中断
dts_wkup	数字输出	温度传感器唤醒

20.3.3 DTS 模块操作

温度传感器的模拟部分用于输出频率，该频率与绝对温度 (CLK_PTAT) 成正比。该频率基于 PCLK 或 LSE 时钟测得。

每次测量前，温度传感器都会对频率生成模块执行校准。

20.3.4 工作模式

可通过设置*温度传感器配置寄存器 1 (DTS_CFGR1)* 中的 REFCLK_SEL 位来选择多种工作模式：

- 仅 PCLK (REFCLK_SEL = 0)
  可以访问温度传感器寄存器。因此，可重新配置接口并使用 PCLK 时钟执行测量序列。
- PCLK 和 LSE (REFCLK_SEL = 1)
  可以访问温度传感器寄存器。因此，可重新配置接口并使用 LSE 时钟执行测量序列。
- 仅 LSE (REFCLK_SEL = 1) 且 PCLK 关闭
  无法访问温度传感器寄存器。可使用 LSE 时钟执行测量。该模式用于通过硬件触发和异步中断线退出睡眠模式。

20.3.5 校准

温度传感器在执行任何频率测量前都必须运行校准。除快速测量模式（DTS_CFGR1 中的 Q_MEAS_OPT 设为 1）外，触发温度测量后都将自动执行校准。

20.3.6 预分频器

在校准过程中，计数器时钟必须低于 1 MHz。这可通过内置在温度传感器中的 PCLK 时钟预分频器来实现。

在温度测量期间，预分频器会被旁路。

- 当 PCLK 用作参考时钟（DTS_CFGR1 中的 REFCLK_SEL 设为 0）时，使用预分频器。配置的分频比最大值为 127（关于分频器设置，请参见 HSREF_CLK_DIV[6:0] 寄存器定义）。
- 当 LSE 用作参考时钟（DTS_CFGR1 中的 REFCLK_SEL 设为 1）时，时基等于 2 个 LSE 周期。在这种情况下，不使用预分频器。

20.3.7 温度测量原理

温度传感器的模拟部分用于输出信号 (CLK_PTAT)，其 FM(T) 频率取决于温度。

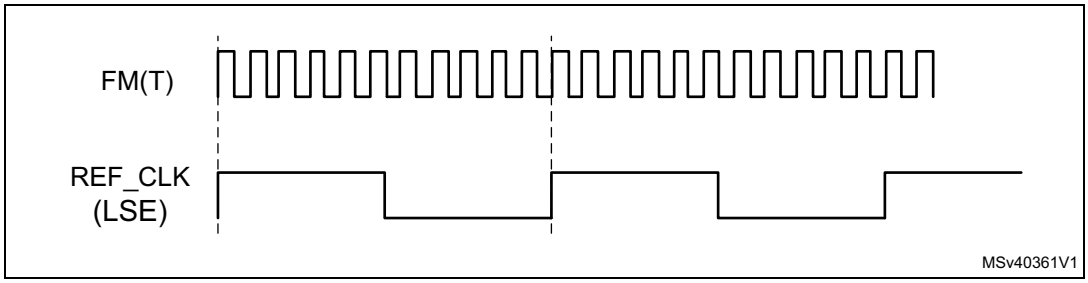
可以通过 DTS_CFGR1 中的 REFCLK_SEL 位将 PCLK 或 LSE 选作参考时钟 (REF_CLK)。

计数方法取决于 REF_CLK 频率。这是因为温度传感器模块中实现了两个计数器：

- 对于低 REF_CLK 频率，在一个或多个 REF_CLK 周期内对 FM(T) 周期进行计数。
- 对于高 REF_CLK 频率，在一个或多个 FM(T) 周期内对 REF_CLK 周期进行计数。

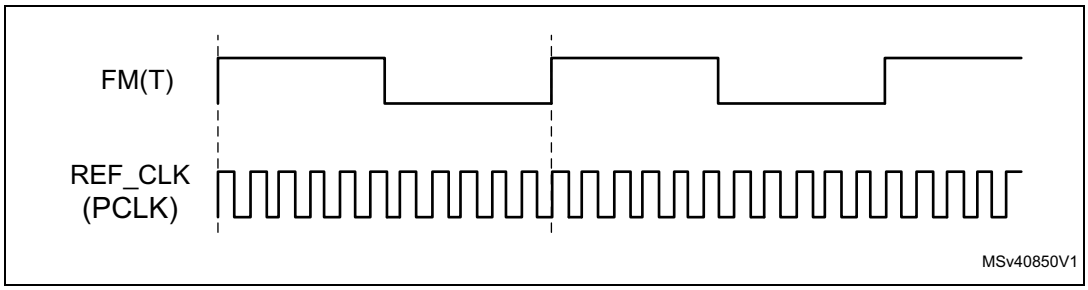
该计数器的行为如*图 131* 和*图 132* 所示。

图 131. 低 REF_CLK 频率时的方法



1. 为提高精度，可以在多个 LSE 周期内进行 FM(T) 测量。

图 132. 高 REF_CLK 频率时的方法



1. 为提高精度，可以在多个 FM(T) 周期内进行 PCLK 测量。

计数结果存储在 DTS_DR 寄存器中（请参见*温度传感器数据寄存器 (DTS_DR)*）。

获得 FM(T) 频率后，可使用以下公式通过软件计算相应的温度：

- 使用 PCLK 时：

$$T = T_0 + ((F_{PCLK} / TS1_MFREQ) \times TS1_SMP_TIME - 100 \times TS1_FMT0) / TS1_RAMP_COEFF$$

其中

T0（出厂校准温度）等于 30°C。

TS1_FMT0 为测量结果，存储在 DTS_T0VALR1 寄存器中。该值以 0.1 kHz 表示。

TS1_RAMP_COEFF 为工厂测试期间的测量结果，存储在 DTS_RAMPVALR 寄存器中。该值以 Hz/°C 表示。

- 使用 LSE 时钟时：

$$T = T_0 + ((F_{LSE} \times TS1_MFREQ / TS1_SMP_TIME) - (100 \times TS1_FMT0)) / TS1_RAMP_COEFF$$

20.3.8 采样时间

可以增加采样周期来提高测量精度。当参考频率 (REF_CLK) 接近 FM(T) 频率时，这种方法非常有用。LSE 模式下的默认值为 1 个 REF_CLK 周期，PCLK 模式下的默认值为 1 个 FM(T) 周期。

采样时间通过 DTS_CFGR1 寄存器中的 TS1_SMP_TIME 位进行配置（请参见*表 123*）。

表 123. 采样时间配置

TS1_SMP_TIME[3:0]	LSE 或 FM(T) 时钟周期
0000	1
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

20.3.9 快速测量模式

如果不要求高精度，可以将 DTS_CFGR1 寄存器中的 Q_MEAS_OPT 设为 1 来跳过每个测量序列中所包含的校准步骤。该方法只能在选择 LSE 时钟作为参考时钟（LSREF_CLK 设为 1）的情况下使用。该模式可以缩短测量时间。

20.3.10 触发输入

温度测量可通过软件或外部事件触发。触发源可通过 DTS_CFGR1 中的 TS1_INTRIG[3:0] 位来选择。

表 124. 触发配置

名称	TS1_INTRIG[3:0]				备注
N.A	0	0	0	0	无硬件触发
ts1_trg0	0	0	0	1	lptim1_ch1
ts1_trg1	0	0	1	0	lptim2_ch1
ts1_trg2	0	0	1	1	保留
ts1_trg3	0	1	0	0	exti13
ts1_trg4	0	1	0	1	保留
ts1_trg5	0	1	1	0	
ts1_trg6	0	1	1	1	
ts1_trg7	1	0	0	0	
ts1_trg8	1	0	0	1	
ts1_trg9	1	0	1	0	
ts1_trg10	1	0	1	1	
ts1_trg11	1	1	0	0	
ts1_trg12	1	1	0	1	
ts1_trg13	1	1	1	0	
ts1_trg14	1	1	1	1	

注意： *硬件触发仅在上升沿有效。*

温度传感器只能在 TS1_RDY 位置 1 时捕获硬件触发上升沿（请参见第 20.3.11 节：开关控制和就绪标志），其他情况下会忽略触发。

如果触发源动态更改，则新的触发源信号应为低电平。如果新的触发源信号为高电平，则温度传感器将检测上升沿，并据此启动测量序列。

20.3.11 开关控制和就绪标志

DTS 模块可通过将 DTS_CFGR1 寄存器中的 TS1_EN 位置 1 来使能。*温度传感器状态寄存器 (DTS_SR)* 中的 TS1_RDY 标志用于指示 DTS 模块是否准备好进行温度测量：当 TS1_RDY 位设为 1 时，可以开始测量。测量开始后，TS1_RDY 位将会复位。随后，新的测量请求将被忽略。测量完成后，TS1_RDY 位会再次置 1，以指示传感器已准备好开始新的测量。

20.3.12 温度测量序列

测量可通过软件或硬件触发启动。

软件触发

当 DTS_CFGR1 中的 TS1_INTRIG_SEL[3:0] 设为“0000”时，选择软件触发。

如果 TS1_RDY 设为 1，则将 1 写入 DTS_CFGR1 中的 TS1_START 位可启动测量。

如果 TS1_RDY 等于 0，则在 TS1_RDY 置 1 前，软件触发都不会启动测量。

如果 TS1_START 在测量完成后保持为 1，则 TS1_RDY 标志将变为 1，并重新启动测量。

硬件触发

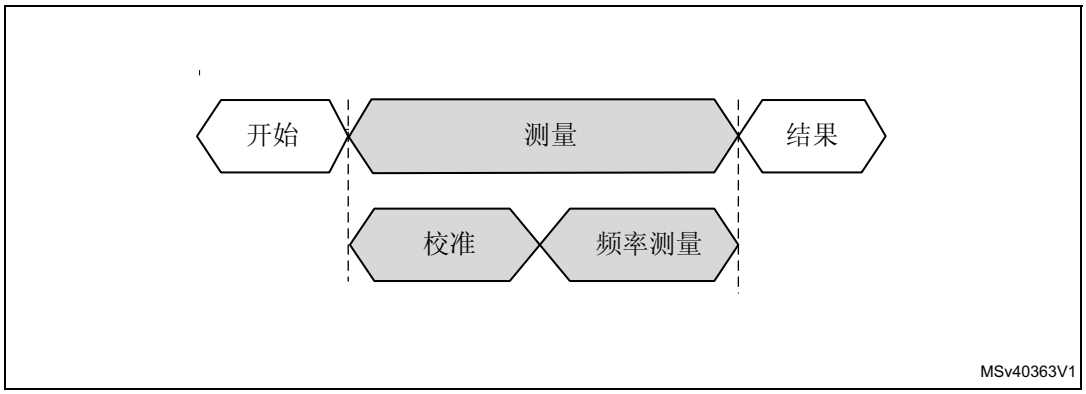
可通过 TS1_INTRIG_SEL[3:0] 位选择 4 个硬件触发信号之一。当 TS1_RDY 设为 1 时，在触发信号的上升沿启动测量。如果 TS1_RDY 为 0，则忽略上升沿。

温度测量序列

一次测量包含两个步骤：校准模拟模块和测量。触发测量后，将自动开始校准（请参见*第 20.3.5 节：校准*）。测量周期取决于以下 DTS_CFGR1 位：

- 参考时钟通过 REFCLK_SEL 位选择
- 分频比由 HSREF_CLK_DIV 位配置
- 采样时间由 TS1_SMP_TIME 位定义。

图 133. 温度传感器序列



20.4 DTS 低功耗模式

表 125. 低功耗模式下的温度传感器行为

模式	说明
睡眠	仅适用于 LSE 模式。 DTS 中断可使器件退出睡眠模式。
停止	仅适用于 LSE 模式。 DTS 中断可使器件退出停止模式。

20.5 DTS 中断

可通过两种方式将 DTS 模块用作中断源。DTS 中断线可以连接到 EXTI 控制器（请参见*第 20.5.3 节：异步唤醒*）或 CPU NVIC（请参见*第 20.5.2 节：同步中断*）。

20.5.1 温度窗口比较器

可通过 DTS_ITR1 寄存器定义用于温度比较的阈值上限和阈值下限。如果温度数据等于/高于 TS1_HITTHD，或者等于/低于 TS1_LITTHD 位，则会生成中断，并且 DTS_SR 寄存器中的相应标志（TS1_ ITLF、TS1_ITHF、TS1_AITLF 和 TS1_AITHF）置 1（请参见*第 20.6.6 节*）。

20.5.2 同步中断

DTS 模块提供了一条全局中断输出线。在测量结束时和/或当测量结果等于/高于或等于/低于预定义阈值时，都会生成中断（请参见*第 20.5.1 节：温度窗口比较器*）。

三个中断事件可通过 DTS_ITENR 寄存器中的 3 位来选择（请参见*第 20.6.7 节*）。三个中断可以进行任意组合来使用。

DTS_SR 寄存器中的 TS1_ITEF、TS1_ITLF 和 TS1_ITHF 标志用于反映中断事件。这些位可通过 DTS_ICIFR 寄存器的相应位来复位（请参见*第 20.6.8 节*）。

20.5.3 异步唤醒

DTS 模块还提供了一条异步中断线。该中断线只能在选择 LSE 作为参考时钟 (REFCLK_SEL = 1) 的情况下使用。

该中断线可生成唤醒信号，以便在测量结束时和/或当测量结果等于/高于或等于/低于预定义阈值时，将系统从睡眠模式唤醒（请参见*第 20.5.1 节：温度窗口比较器*）。

三个异步唤醒事件可通过 DTS_ITENR 寄存器中的 3 位来选择。三个中断可以进行任意组合来使用。

DTS_SR 寄存器中的 TS1_AITEF、TS1_AITLF 和 TS1_AITHF 标志用于反映中断状态。它们可通过 DTS_ICIFR 寄存器的相应位来复位。

下表给出了各个中断位及其说明。

表 126. 中断控制位

中断事件	中断标志	使能控制位	中断清除位	退出睡眠模式	同步/异步
当测量结束时	DTS_SR 中的 TS1_ITEF	DTS_ITENR 中的 TS1_ITEEN	DTS_ICIFR 中的 TS1_CITEF	否	基于 PCLK 进行同步
当测量值等于/低于阈值下限时	DTS_SR 中的 TS1_ITLF	DTS_ITENR 中的 TS1_ITLEN	DTS_ICIFR 中的 TS1_CITLF	否	
当测量值等于/高于阈值上限时	DTS_SR 中的 TS1_ITHF	DTS_ITENR 中的 TS1_ITHEN	DTS_ICIFR 中的 TS1_CITHF	否	
当测量结束时	DTS_SR 中的 TS1_AITEF	DTS_ITENR 中的 TS1_AITEEN	DTS_ICIFR 中的 TS1_CAITEF	是	异步
当测量值等于/低于阈值下限时	DTS_SR 中的 TS1_AITLF	DTS_ITENR 中的 TS1_AITLEN	DTS_ICIFR 中的 TS1_CAITLF	是	
当测量值等于/高于阈值上限时	DTS_SR 中的 TS1_AITHF	DTS_ITENR 中的 TS1_AITHEN	DTS_ICIFR 中的 TS1_CAITHF	是	

20.6 DTS 寄存器

该外设的寄存器只能按字（32 位）进行访问。

20.6.1 温度传感器配置寄存器 1 (DTS_CFGR1)

Temperature sensor configuration register 1

DTS_CFGR1 是温度传感器 1 的配置寄存器。

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	HSREF_CLK_DIV[6:0]							Res.	Res.	Q_MEAS_OPT	REFCLK_SEL	TS1_SMP_TIME[3:0]			
	rw	rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	TS1_INTRIG_SEL[3:0]				Res.	Res.	Res.	TS1_START	Res.	Res.	Res.	TS1_EN
				rw	rw	rw	rw				rw				rw

位 31 保留，必须保持复位值。

位 30:24 **HSREF_CLK_DIV[6:0]**：高速时钟分频比 (High speed clock division ratio)

这些位由软件置 1 和清零，可用于定义主时钟的分频比，以获得校准所需的低于 1 MHz 的内部频率。这些位仅在选择 PCLK 作为参考时钟 (REFCLK_SEL = 0) 的情况下才适用于校准。

0000000：无分频器

0000001：无分频器

0000010：1/2 分频比

...

1111111：1/127 分频比

位 23:22 保留，必须保持复位值。

位 21 **Q_MEAS_OPT**：快速测量选项位 (Quick measurement option bit)

该位由软件置 1 和清零，用于跳过校准步骤以提高测量速度。该位仅在将 LSE 时钟用作参考时钟 (REFCLK_SEL = 1) 的情况下生效。

0：含校准的测量

1：不含校准的测量

位 20 **REFCLK_SEL**：参考时钟选择位 (Reference clock selection bit)

该位由软件置 1 和清零，用于指示参考时钟是高速时钟 (PCLK) 还是低速时钟 (LSE)。

0：高速参考时钟 (PCLK)

1：低速参考时钟 (LSE)

位 19:16 **TS1_SMP_TIME[3:0]**：温度传感器 1 的采样时间 (Sampling time for temperature sensor 1)

这些位用于增加采样时间以提高测量精度。

当选择 PCLK 时钟作为参考时钟 (REFCLK_SEL = 0) 时，以 CLK_PTAT 的 TS1_SMP_TIME 周期执行测量。

当选择 LSE 作为参考时钟 (REFCLK_SEL = 1) 时，以 LSE 的 TS1_SMP_TIME 周期执行测量。

位 15:12 保留，必须保持复位值。

位 11:8 **TS1_INTRIG_SEL[3:0]**：温度传感器 1 的输入触发选择位 (Input trigger selection bit for temperature sensor 1)

这些位由软件置 1 和清零，用于选择温度测量的输入触发。请参见*第 20.3.10 节：触发输入*。

位 7:5 保留，必须保持复位值。

位 4 **TS1_START**：启动温度传感器 1 的频率测量 (Start frequency measurement on temperature sensor 1)

该位由软件置 1 和清零。

0：无软件触发。

1：软件触发频率测量。（仅当 TS1 就绪时）。

位 3:1 保留，必须保持复位值。

位 0 **TS1_EN**：温度传感器 1 使能位 (Temperature sensor 1 enable bit)

该位由软件置 1 和清零。

0：禁止温度传感器 1

1：使能温度传感器 1

注意：使能时，温度传感器将在一段特定的延时后激活。TS1_RDY 标志在传感器就绪时置 1。

20.6.2 温度传感器 T0 值寄存器 1 (DTS_T0VALR1)

Temperature sensor T0 value register 1

DTS_T0VALR1 包含温度传感器 1 的出厂校准温度值 (T0)。复位值经过出厂微调。

偏移地址：0x08

复位值：0x000X XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_T0[1:0]	
														r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS1_FMT0[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:18 保留，必须保持复位值。

位 17:16 **TS1_T0[1:0]**：温度传感器 1 的 T0 温度工程值 (Engineering value of the T0 temperature for temperature sensor 1)

00：30°C

01：130°C

其他值：保留，不得使用。

位 15:0 **TS1_FMT0[15:0]**：温度传感器 1 在 T0 测得的频率工程值 (Engineering value of the frequency measured at T0 for temperature sensor 1)

该值以 0.1 kHz 表示。

20.6.3 温度传感器斜坡值寄存器 (DTS_RAMPVALR)

Temperature sensor ramp value register

DTS_RAMPVALR 是温度传感器的斜坡系数。复位值经过出厂微调。

偏移地址：0x10

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS1_RAMP_COEFF[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 保留，必须保持复位值。

位 15:0 **TS1_RAMP_COEFF[15:0]**：温度传感器 1 的斜坡系数工程值 (Engineering value of the ramp coefficient for the temperature sensor 1)

该值以 Hz/°C 表示。

20.6.4 温度传感器中断阈值寄存器 1 (DTS_ITR1)

Temperature sensor interrupt threshold register 1

DTS_ITR1 包含传感器 1 的阈值。

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TS1_HITTHD[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS1_LITTHD[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 **TS1_HITTHD[15:0]**：温度传感器 1 的中断阈值上限 (High interrupt threshold for temperature sensor 1)

这些位由软件置 1 和清零，用于指示在发出中断信号之前可以达到的最高值。

位 15:0 **TS1_LITTHD[15:0]**：温度传感器 1 的中断阈值下限 (Low interrupt threshold for temperature sensor 1)

这些位由软件置 1 和清零，用于指示在发出中断信号之前可以达到的最低值。

20.6.5 温度传感器数据寄存器 (DTS_DR)

Temperature sensor data register

DTS_DR 包含用于计算 FM(T) 频率的 REF_CLK 周期数。

偏移地址：0x1C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS1_MFREQ[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:0 **TS1_MFREQ[15:0]**：温度传感器 1 的计数器输出值 (Value of the counter output value for temperature sensor 1)

20.6.6 温度传感器状态寄存器 (DTS_SR)

Temperature sensor status register

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS1_RDY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_AITHF	TS1_AITLF	TS1_AITEF	Res.	TS1_ITHF	TS1_ITLF	TS1_ITEF
r									r	r	r		r	r	r

位 31:16 保留，必须保持复位值。

位 15 **TS1_RDY**：温度传感器 1 就绪标志 (Temperature sensor 1 ready flag)

该位由硬件置 1 和复位，
用于指示正在进行测量。
0：温度传感器 1 繁忙
1：温度传感器 1 就绪

位 14:7 保留，必须保持复位值。

位 6 **TS1_AITHF**：温度传感器 1 阈值上限异步中断标志 (Asynchronous interrupt flag for high threshold on temperature sensor 1)

当达到阈值上限时，该位由硬件置 1。
通过软件向 DTS_ICIFR 寄存器中的 TS1_CAITHF 位写入 1 可将该位清零。
0：温度传感器 1 未达到阈值上限
1：温度传感器 1 达到阈值上限
注：该位只有在 TS1_AITHFEN 位置 1 时才有效

位 5 **TS1_AITLF**：温度传感器 1 阈值下限异步中断标志 (Asynchronous interrupt flag for low threshold on temperature sensor 1)

当达到阈值下限时，该位由硬件置 1。

通过软件向 DTS_ICIFR 寄存器中的 TS1_CAITLF 位写入 1 可将该位清零。

0：温度传感器 1 未达到阈值下限

1：温度传感器 1 达到阈值下限

注：该位只有在 TS1_AITLFEN 位置 1 时才有效

位 4 **TS1_AITEF**：温度传感器 1 测量结束异步中断标志 (Asynchronous interrupt flag for end of measure on temperature sensor 1)

当温度测量完成时，该位由硬件置 1。

通过软件向 DTS_ICIFR 寄存器中的 TS1_CAITEF 位写入 1 可将该位清零。

0：未检测到温度传感器 1 测量结束

1：检测到温度传感器 1 测量结束

注：该位只有在 TS1_AITEFEN 位置 1 时才有效

位 3 保留，必须保持复位值。

位 2 **TS1_ITHF**：温度传感器 1 阈值上限中断标志（基于 PCLK 进行同步）(Interrupt flag for high threshold on temperature sensor 1, synchronized on PCLK)

当设置并达到阈值上限时，该位由硬件置 1。

通过软件向 DTS_ICIFR 寄存器中的 TS1_CITHF 位写入 1 可将该位清零。

0：温度传感器 1 未达到阈值上限

1：温度传感器 1 达到阈值上限

注：该位只有在 TS1_ITHFEN 位置 1 时才有效

位 1 **TS1_ITLF**：温度传感器 1 阈值下限中断标志（基于 PCLK 进行同步）(Interrupt flag for low threshold on temperature sensor 1, synchronized on PCLK)

当设置并达到阈值下限时，该位由硬件置 1。

通过软件向 DTS_ICIFR 寄存器中的 TS1_CITLF 位写入 1 可将该位清零。

0：温度传感器 1 未达到阈值下限

1：温度传感器 1 达到阈值下限

注：该位只有在 TS1_ITLFEN 位置 1 时才有效

位 0 **TS1_ITEF**：温度传感器 1 测量结束中断标志（基于 PCLK 进行同步）(Interrupt flag for end of measurement on temperature sensor 1, synchronized on PCLK)

当温度测量完成时，该位由硬件置 1。

通过软件向 DTS_ICIFR 寄存器中的 TS2_CITEF 位写入 1 可将该位清零。

0：未检测到温度传感器 1 测量结束

1：检测到温度传感器 1 测量结束

注：该位只有在 TS1_ITEFEN 位置 1 时才有效

20.6.7 温度传感器中断使能寄存器 (DTS_ITENR)

Temperature sensor interrupt enable register

偏移地址：0x24

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_AITHEN	TS1_AITLEN	TS1_AITEEN	Res.	TS1_ITHEN	TS1_ITLEN	TS1_ITEEN
									rw	rw	rw		rw	rw	rw

位 31:7 保留，必须保持复位值。

位 6 **TS1_AITHEN**：温度传感器 1 阈值上限异步中断使能标志 (Asynchronous interrupt enable flag on high threshold for temperature sensor 1)

该位由软件置 1 和清零，
用于使能在温度高于阈值上限时生成异步中断（仅当 REFCLK_SEL = 1 时）。
0：禁止温度传感器 1 阈值上限异步中断
1：使能温度传感器 1 阈值上限异步中断

位 5 **TS1_AITLEN**：温度传感器 1 阈值下限异步中断使能标志 (Asynchronous interrupt enable flag for low threshold on temperature sensor 1)

该位由软件置 1 和清零，
用于在温度低于阈值下限时使能异步中断（仅当 REFCLK_SEL = 1 时）。
0：禁止温度传感器 1 阈值下限异步中断
1：使能温度传感器 1 阈值下限异步中断

位 4 **TS1_AITEEN**：温度传感器 1 测量结束异步中断使能标志 (Asynchronous interrupt enable flag for end of measurement on temperature sensor 1)

该位由软件置 1 和清零，
用于使能测量结束异步中断（仅当 REFCLK_SEL = 1 时）。
0：禁止温度传感器 1 测量结束异步中断
1：使能温度传感器 1 测量结束异步中断

位 3 保留，必须保持复位值。

位 2 **TS1_ITHEN**：温度传感器 1 阈值上限中断使能标志（基于 PCLK 进行同步）(Interrupt enable flag for high threshold on temperature sensor 1, synchronized on PCLK)

该位由软件置 1 和清零，
用于使能在测量值等于/高于阈值上限时生成中断。
0：禁止温度传感器 1 阈值上限同步中断
1：使能温度传感器 1 阈值上限同步中断

位 1 **TS1_ITLEN**：温度传感器 1 阈值下限中断使能标志（基于 PCLK 进行同步）(Interrupt enable flag for low threshold on temperature sensor 1, synchronized on PCLK)

该位由软件置 1 和清零，
用于使能在测量值等于/低于阈值下限时生成同步中断。
0：禁止温度传感器 1 阈值下限同步中断
1：使能温度传感器 1 阈值下限同步中断

位 0 **TS1_ITEEN**：温度传感器 1 测量结束中断使能标志（基于 PCLK 进行同步）(Interrupt enable flag for end of measurement on temperature sensor 1, synchronized on PCLK)

该位由软件置 1 和清零，
用于使能测量结束同步中断。
0：禁止温度传感器 1 测量结束同步中断
1：使能温度传感器 1 测量结束同步中断

20.6.8 温度传感器清除中断标志寄存器 (DTS_ICIFR)

Temperature sensor clear interrupt flag register

DTS_ICIFR 是中断标志控制寄存器。

偏移地址：0x28

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_CAITHF	TS1_CAITLF	TS1_CAITEF	Res.	TS1_CITHF	TS1_CITLF	TS1_CITEF
									rc_w1	rc_w1	rc_w1		rc_w1	rc_w1	rc_w1

位 31:7 保留，必须保持复位值。

位 6 **TS1_CAITHF**：温度传感器 1 阈值上限异步中断清零标志 (Asynchronous interrupt clear flag for high threshold on temperature sensor 1)

向该位写入 1 可将 DTS_SR 寄存器中的 TS1_AITHF 标志清零。

位 5 **TS1_CAITLF**：温度传感器 1 阈值下限异步中断清零标志 (Asynchronous interrupt clear flag for low threshold on temperature sensor 1)

向该位写入 1 可将 DTS_SR 寄存器中的 TS1_AITLF 标志清零。

位 4 **TS1_CAITEF**：一次写入位，用于清零温度传感器 1 测量结束异步 IT 标志。

向该位写入 1 可将 DTS_SR 寄存器中的 TS1_AITEF 标志清零。

位 3 保留，必须保持复位值。

位 2 **TS1_CITHF**：温度传感器 1 阈值上限中断清零标志 (Interrupt clear flag for high threshold on temperature sensor 1)

向该位写入 1 可将 DTS_SR 寄存器中的 TS1_ITHF 标志清零。

位 1 **TS1_CITLF**：温度传感器 1 阈值下限中断清零标志 (Interrupt clear flag for low threshold on temperature sensor 1)

向该位写入 1 可将 DTS_SR 寄存器中的 TS1_ITLF 标志清零。

位 0 **TS1_CITEF**：温度传感器 1 测量结束中断清零标志 (Interrupt clear flag for end of measurement on temperature sensor 1)

向该位写入 1 可将 DTS_SR 寄存器中的 TS1_ITEF 标志清零。

20.6.9 温度传感器选项寄存器 (DTS_OR)

Temperature sensor option register

DTS_OR 包含通用选项位。

偏移地址：0x2C

复位值：0x0000 0000

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

位 31:0 **TS_OP[31:0]**：通用选项位 (general purpose option bits)

20.6.10 DTS 寄存器映射

下表对温度传感器寄存器进行了汇总。

表 127. DTS 寄存器映射和复位值

偏移	寄存器名称 复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**DTS_ CFGR1**	Res.	HSREF_CLK_DIV [6:0]							Res.	Res.	Q_MEAS_OPT	REFCLK_SEL	TS1_ SMP_TIME [3:0]				Res.	Res.	Res.	Res.	TS1_ INTRIG_SEL [3:0]				Res.	Res.	Res.	TS1_START	Res.	Res.	Res.	TS1_EN
	Reset value		0	0	0	0	0	0	0			0	0	0	0	0	0					0	0	0	0				0				0
0x04	Reserved																																
0x08	**DTS_ T0VALR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_T0[1:0]		TS1_FMT0[15:0]															
	Reset value															X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x0C	Reserved																																
0x10	**DTS_ RAMPVALR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_RAMP_COEFF[15:0]															
	Reset value																	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x14	**DTS_ITR1**	TS1_HITTHD[15:0]																TS1_LITTHD[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	Reserved																																
0x1C	**DTS_DR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_MFREQ[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	**DTS_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_RDY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_AITHF	TS1_AITLF	TS1_AITEF	Res.	TS1_ITHF	TS1_ITLF	TS1_ITEF
	Reset value																	0									0	0	0		0	0	0
0x24	**DTS_ITENR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_AITHEN	TS1_AITLEN	TS1_AITEEN	Res.	TS1_ITHEN	TS1_ITLEN	TS1_ITEEN
	Reset value																										0	0	0		0	0	0
0x28	**DTS_ICIFR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1_CAITHF	TS1_CAITLF	TS1_CAITEF	Res.	TS1_CITHF	TS1_CITLF	TS1_CITEF
	Reset value																										0	0	0		0	0	0
0x2C	**DTS_OR**	TS_OP31	TS_OP30	TS_OP29	TS_OP28	TS_OP27	TS_OP26	TS_OP25	TS_OP24	TS_OP23	TS_OP22	TS_OP21	TS_OP20	TS_OP19	TS_OP18	TS_OP17	TS_OP16	TS_OP15	TS_OP14	TS_OP13	TS_OP12	TS_OP11	TS_OP10	TS_OP9	TS_OP8	TS_OP7	TS_OP6	TS_OP5	TS_OP4	TS_OP3	TS_OP2	TS_OP1	TS_OP0
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

21 数模转换器 (DAC)

21.1 简介

DAC 模块是 12 位电压输出数模转换器。DAC 可以按 8 位或 12 位模式进行配置，并且可与 DMA 控制器配合使用。在 12 位模式下，数据可以采用左对齐或右对齐。DAC 有两个输出通道，每个通道各有一个转换器。在 DAC 双通道模式下，每个通道可以单独进行转换，或者当两个通道组合在一起同步执行更新操作时，也可以同时进行转换。

如果 DAC 输出与输出焊盘断开连接并连接到片上外设，DACx_OUTy 引脚可用作通用输入/输出 (GPIO)。可选择使能 DAC 输出缓冲器，从而获得高驱动输出电流。可分别对每条 DAC 输出通道应用校准。DAC 输出通道支持低功耗模式，即采样和保持模式。

21.2 DAC 主要特性

DAC 的主要特性如下（请参见*图 134：双通道 DAC 框图*）

- 一个 DAC 接口，最多两个输出通道
- 12 位模式下数据采用左对齐或右对齐
- 同步更新功能
- 噪声波和三角波产生
- 双 DAC 通道，可独立或同时进行转换
- 每条通道都有 DMA 功能，包括 DMA 下溢错误检测
- 双数据 DMA 功能，可减少总线活动
- 外部触发信号进行转换
- DAC 输出通道缓冲/非缓冲模式
- 缓冲器偏移校准
- 每路 DAC 输出均可与 DACx_OUTy 输出引脚断开连接
- DAC 输出可与片上外设连接
- 可在 STOP 模式下通过采样和保持模式实现低功耗运行

图 134 所示为 DAC 通道的框图，*表 129* 则给出了引脚说明。

21.3 DAC 实现

表 128. DAC 特性

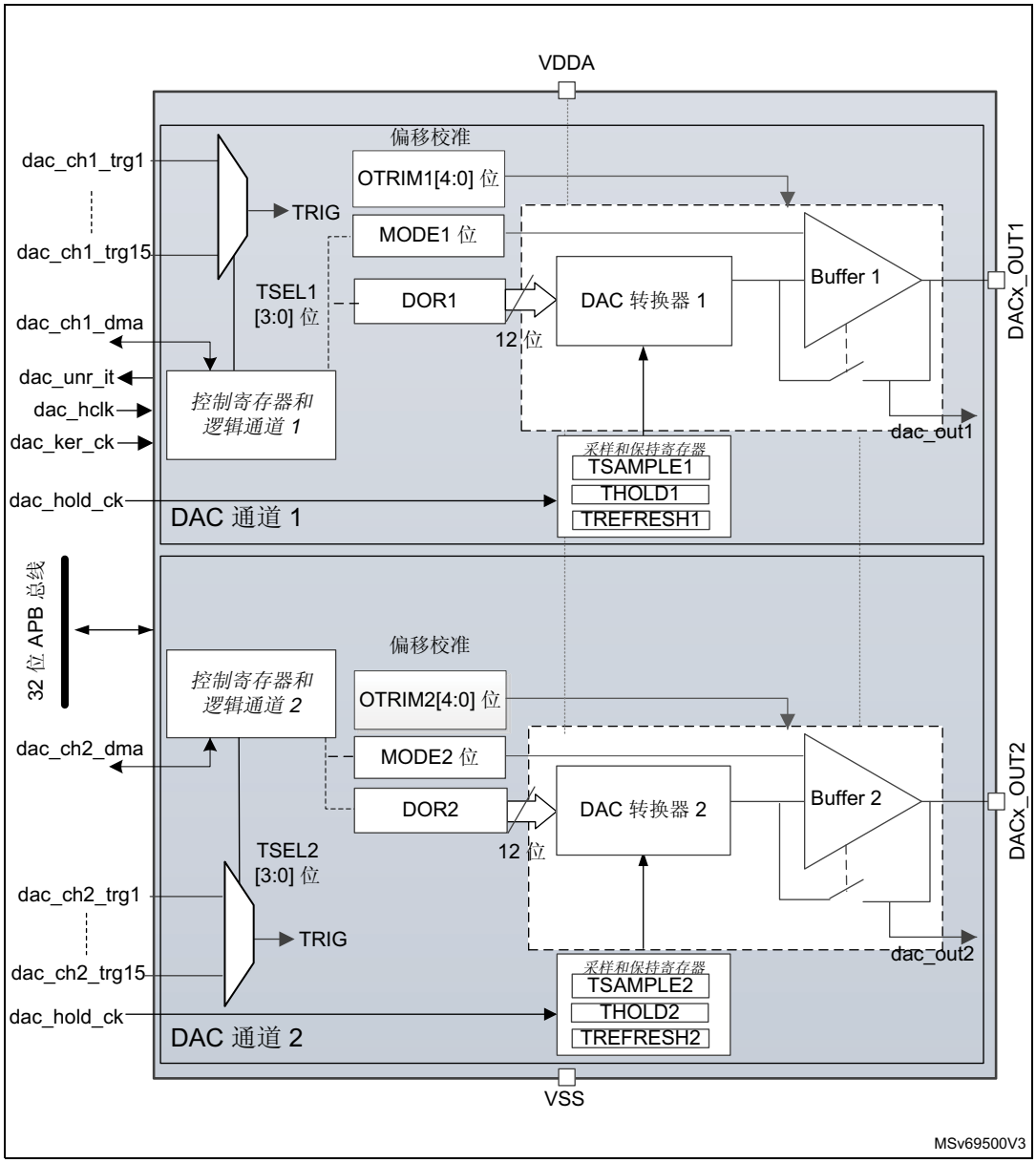
DAC 特性	DAC1
双通道	X
输出缓冲器	X
I/O 连接	PA4 上的 DAC1_OUT1，PA5 上的 DAC1_OUT2
最大采样时间	1 Msps
自主模式	-
VREF+ 引脚	-(1)

1. VREF+ 内部连接到 VDDA。

21.4 DAC 功能说明

21.4.1 DAC 框图

图 134. 双通道 DAC 框图



1. DAC_MCR 中的 MODEx 位用于控制输出模式，并允许在正常模式（采用缓冲/非缓冲配置）与采样和保持模式之间切换。
2. 关于通道 2 的可用性，请参见*第 21.3 节：DAC 实现*。

21.4.2 DAC 引脚和内部信号

DAC 包含：

- 最多两条输出通道
- DACx_OUTy 可与输出引脚断开连接并用作普通 GPIO
- dac_outx 可以使用内部引脚连接到芯片上的外设，如比较器、运算放大器和 ADC（如果可用）
- DAC 输出通道（缓冲/非缓冲）
- 可在 STOP 模式下使用 LSI/LSE 时钟源 (dac_hold_ck) 运行采样和保持模块及其寄存器，以进行静态转换。

DAC 包含 多达两条独立的输出通道。每条输出通道均可连接到片上外设，如比较器、运算放大器和 ADC（如有）。在这种情况下，DAC 输出通道可与 DACx_OUTy 输出引脚断开连接，相应的 GPIO 可用于其他用途。

DAC 输出可缓冲、也可以不缓冲。采样和保持模块及其关联寄存器可在 STOP 模式下使用 LSI/LSE 时钟源 (dac_hold_ck) 运行。

表 129. DAC 输入/输出引脚

引脚名称	信号类型	备注
VDDA	模拟电源输入	模拟电源
VSSA	模拟电源地输入	模拟电源地
DACx_OUTy	模拟输出信号	DACx 通道 y 模拟输出

表 130. DAC 内部输入/输出信号

内部信号名称	信号类型	说明
dac_ch1_dma	双向	DAC 通道 1 DMA 请求/确认
dac_ch2_dma	双向	DAC 通道 2 DMA 请求/确认
dac_ch1_trgx（x = 1 到 15）	输入	DAC 通道 1 触发输入
dac_ch2_trgx（x = 1 到 15）	输入	DAC 通道 2 触发输入
dac_unr_it	输出	DAC 下溢中断
dac_hclk	输入	DAC 外设时钟
dac_ker_ck	输入	DAC 内核时钟
dac_hold_ck	输入	在采样和保持模式下使用的 DAC 低功耗时钟
dac_out1	模拟输出	片上外设的 DAC 通道 1 输出
dac_out2	模拟输出	片上外设的 DAC 通道 2 输出

表 131. DAC 互连

信号名称	源	源类型
dac_hold_ck	ck_lsi 或 ck_lse	在 RCC 中选择的 LSI 或 LSE 时钟
dac_chx_trg1（x = 1、2）	tim1_trgo	片上定时器的内部信号
dac_chx_trg2（x = 1、2）	tim2_trgo	片上定时器的内部信号
dac_chx_trg3（x = 1、2）	tim3_trgo	片上定时器的内部信号
dac_chx_trg5（x = 1、2）	tim6_trgo	片上定时器的内部信号
dac_chx_trg6（x = 1、2）	tim7_trgo	片上定时器的内部信号
dac_chx_trg11（x = 1、2）	lptim1_ch1	片上定时器的内部信号
dac_chx_trg12（x = 1、2）	lptim2_ch1	片上定时器的内部信号
dac_chx_trg13（x = 1、2）	exti9	外部引脚

21.4.3 DAC 时钟

有两个时钟源可用于更新 DAC：

- dac_hclk：DAC 外设时钟（AHB 时钟）
- dac_ker_ck：DAC 内核时钟：此时钟可用于同步 DAC 和 ADC。
- dac_hold_ck：在采样和保持模式下使用的低功耗时钟

DAC 时钟在 RCC 中选择。

21.4.4 DAC 通道使能

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

当 DAC 接口准备好接受数据时，DAC_SR 寄存器中的 DACxRDY 位会置 1。当 ENx 位置 1 且 DACxRDY 信号复位时，不允许写入新数据，也不允许将触发信号置为有效。

注意： *ENx 位仅会使能模拟 DAC 通道 x。即使 ENx 位复位，DAC 通道 x 数字接口仍处于使能状态。*

21.4.5 DAC 数据格式

根据所选配置模式，数据必须按如下方式写入指定寄存器：

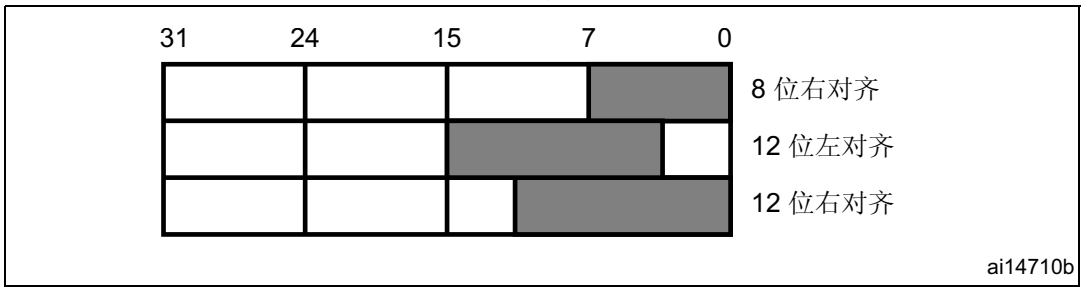
- DAC 单通道

  存在三种可能：

  – 8 位右对齐：软件必须将数据加载到 DAC_DHR8Rx[7:0] 位（存储到 DHRx[11:4] 位）。
  – 12 位左对齐：软件必须将数据加载到 DAC_DHR12Lx[15:4] 位（存储到 DHRx[11:0] 位）。
  – 12 位右对齐：软件必须将数据加载到 DAC_DHR12Rx[11:0] 位（存储到 DHRx[11:0] 位）。

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

图 135. DAC 单通道模式下的数据寄存器



- DAC 双通道（如果可用）

  存在三种可能：

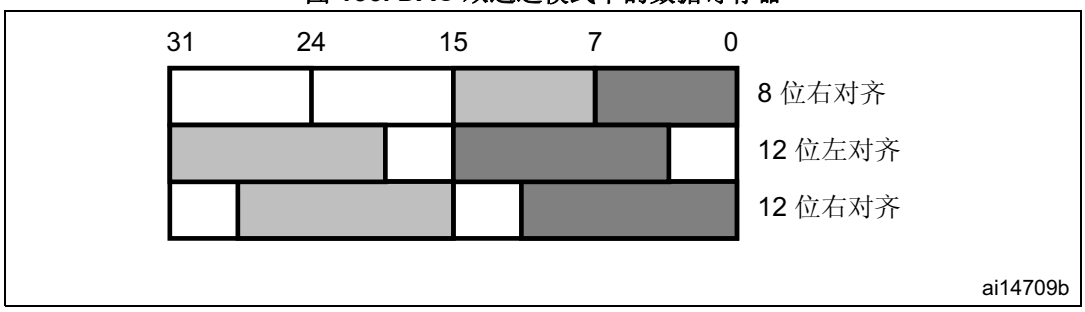
  – 8 位右对齐：将 DAC 通道 1 的数据加载到 DAC_DHR8RD [7:0] 位（存储到 DHR1[11:4] 位），将 DAC 通道 2 的数据加载到 DAC_DHR8RD [15:8] 位（存储到 DHR2[11:4] 位）

  – 12 位左对齐：将 DAC 通道 1 的数据加载到 DAC_DHR12LD [15:4] 位（存储到 DHR1[11:0] 位），将 DAC 通道 2 的数据加载到 DAC_DHR12LD [31:20] 位（存储到 DHR2[11:0] 位）

  – 12 位右对齐：将 DAC 通道 1 的数据加载到 DAC_DHR12RD [11:0] 位（存储到 DHR1[11:0] 位），将 DAC 通道 2 的数据加载到 DAC_DHR12RD [27:16] 位（存储到 DHR2[11:0] 位）

根据加载的 DAC_DHRyyyD 寄存器，用户写入的数据将移位并存储到 DHR1 和 DHR2（数据保持寄存器，即内部非存储器映射寄存器）。之后，DHR1 和 DHR2 寄存器将被自动加载，或者通过软件或外部事件触发分别被加载到 DAC_DOR1 和 DAC_DOR2 寄存器。

图 136. DAC 双通道模式下的数据寄存器



有符号/无符号数据

DAC 输入数据是无符号数据：在 12 位模式下，0x000 对应于最小值，0xFFF 对应于最大值。

DAC 也可以处理采用二进制补码格式的有符号输入数据。实现方式是将 DAC_MCR 寄存器中的 SINFORMATx 位置 1。

当 SINFORMATx 位置 1 时，写入 DHRx 寄存器的数据的 MSB 位在复制到 DAC_DORx 寄存器时会被反转，并且 DAC 接口可以接受有符号数据（Q1.15、Q1.11 或 Q1.7 格式）。DAC_DHR12Lx 寄存器可用于在数据保持寄存器中存储 16 位有符号数据。16 位数据的 12 个 MSB 用于 DAC 输出数据，并且 MSB 位被反转。四个 LSB 则被忽略。

表 132. 数据格式（以 12 位数据为例）

SINFORMATx 位	写入 DHRx 寄存器的数据	传输到 DORx 寄存器的数据
0	0x000	0x000
0	0xFFF	0xFFF
1	0x7FF	0xFFF
1	0x000	0x800
1	0xFFF	0x7FF
1	0x800	0x000

21.4.6 DAC 转换

DAC_DORx 无法直接写入，任何数据都必须通过加载 DAC_DHRx 寄存器（写入 DAC_DHR8Rx、DAC_DHR12Lx、DAC_DHR12Rx、DAC_DHR8RD、DAC_DHR12RD 或 DAC_DHR12LD）才能传输到 DAC 通道x。

如果未选择硬件触发（DAC_CR 寄存器中的 TENx 位复位），那么经过一个 dac_hclk 时钟周期后，DAC_DHRx 寄存器中存储的数据自动传输到 DAC_DORx 寄存器。但是，如果选择硬件触发（DAC_CR 寄存器中的 TENx 位置 1）且触发条件到来，将在触发信号后经过三个 dac_hclk 时钟周期进行传输。

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

要同步 DAC 和 ADC，可以为这两个外设使用相同的时钟源。实现方式是在 RCC 中选择 dac_ker_ck 时钟，而不是 dac_hclk 时钟（AHB 时钟）。

当 dac_hclk 或 dac_ker_ck 时钟速度大于 80 MHz 时，DAC_MCR 的 HFSEL 位必须置 1。之后，从 DAC_DHRx 寄存器传输到 DAC_DORx 寄存器会增加额外的延迟。

关于 DAC_DORx 更新频率的限制与 HFSEL 位和 dac_hclk 时钟频率的关系，请参见下方的 *HFSEL 说明*表。

如果未经允许更新了数据或发生软件/硬件触发事件，外设的行为将无法预测。

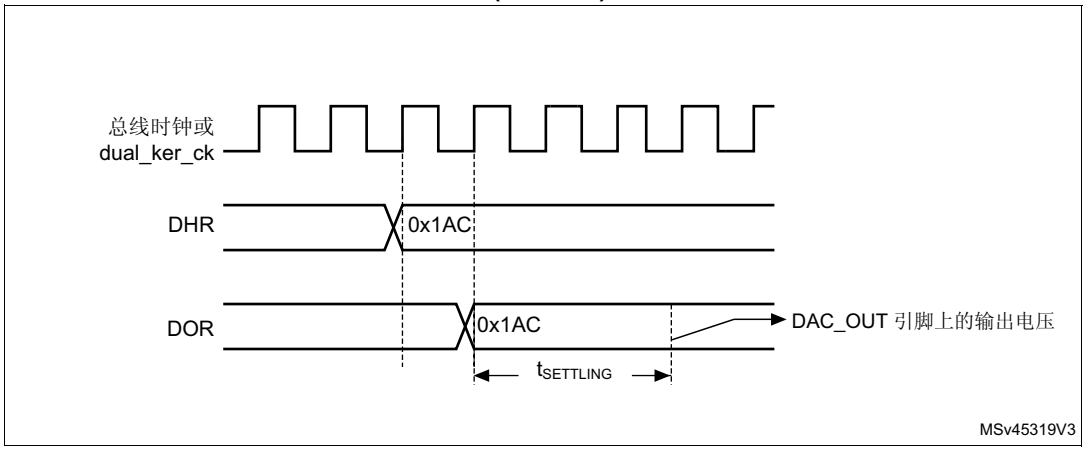
以上时序仅与 DAC 接口的限制相关。另请参见产品数据手册中的 $t_{SETTLING}$ 参数值。

表 133. HFSEL 说明

HFSEL [1:0]	AHB 频率	使用 AHB 时钟 (dac_hclk) 时的延迟	使用 dac_ker_ck 时钟时的延迟	功能
00	< 80 MHz	3	4	DAC_DOR 更新频率最高为 3 个 AHB 时钟周期或 4 个 dac_ker_ck 周期一次。
01	≥ 80 MHz(1)	5	5	DAC_DOR 更新频率最高为 5 个 AHB 时钟周期或 5 个 dac_ker_ck 周期一次。
10	≥ 160 MHz	7	6	DAC_DOR 更新频率最高为 7 个 AHB 时钟周期或 6 个 dac_ker_ck 周期一次。
11	保留	-	-	-

1. 关于最大 AHB 频率值，请参见器件数据手册。

图 137. 关闭触发 (TEN = 0) 时的转换时序图



21.4.7 DAC 输出电压

经过线性转换后，数字输入会转换为 0 到 V_{REF+} 之间的输出电压。

各 DAC 通道引脚的模拟输出电压通过以下公式确定：

$$\text{DAC 输出} = V_{REF} \times \frac{DOR}{4096}$$

21.4.8 DAC 触发选择

如果 TENx 控制位置 1，可通过外部事件（定时计数器、外部中断线）触发转换。TSELx[3:0] 控制位将决定通过 16 个可能事件中的哪一个来触发转换，如 DAC_CR 寄存器的 TSELx[3:0] 位所示。这些事件可以是软件触发，也可以是硬件触发。请参见*第 21.4.2 节：DAC 引脚和内部信号*中的互连表。

每当 DAC 接口在所选触发源上检测到上升沿时（请参见下表），DAC_DHRx 寄存器中存储的最后一个数据即会传输到 DAC_DORx 寄存器中。发生触发后再经过三个 dac_hclk 周期，DAC_DORx 寄存器将会得到更新。

如果选择软件触发，一旦 SWTRIG 位置 1，转换即会开始。DAC_DHRx 寄存器内容加载到 DAC_DORx 寄存器中后，SWTRIG 即由硬件复位。

注意： *ENx 位置 1 时，无法更改 TSELx[3:0] 位。*

如果选择软件触发，DAC_DHRx 寄存器的内容只需一个 dac_hclk 时钟周期即可传输到 DAC_DORx 寄存器。

21.4.9 DMA 请求

每个 DAC 通道都具有 DMA 功能。两个 DMA 通道用于处理 DAC 通道的 DMA 请求。

当 DMAENx 位置 1 时，如果发生外部触发（而不是软件触发），则传输结束后 DAC_DHRx 寄存器的值会传输到 DAC_DORx 寄存器，并且会产生 DMA 请求。

在双通道模式下，如果两个 DMAENx 位均置 1，则将产生两个 DMA 请求。如果只需要一个 DMA 请求，则只需将相应的 DMAENx 位置 1。这样，应用程序可以在双通道模式下通过一个 DMA 请求和一个特定 DMA 通道来管理两个 DAC 通道。

由于 DAC_DHRx 已在 DMA 请求之前向 DAC_DORx 传输数据，必须在发生第一个触发事件之前将第一个数据写入 DAC_DHRx。

DMA 下溢

DAC DMA 请求没有缓冲队列。这样，如果第二个外部触发在第一个外部触发的确认接收之前到达（第一个请求），将不会发出新的请求，并且 DAC_SR 寄存器中的 DMA 通道 x 下溢标志 DMAUDRx 将置 1，以报告这一错误状况。DAC 通道 x 仍将继续转换旧有数据。

软件必须通过写入 1 来将 DMAUDRx 标志清零，将所用 DMA 数据流的 DMAEN 位清零，并重新初始化 DMA 和 DAC 通道 x，以便正确地重新开始 DMA 传输。软件必须修改 DAC 触发转换频率或减轻 DMA 工作负载，以避免再次发生 DMA 下溢。最后，可通过使能 DMA 数据传输和转换触发来继续完成 DAC 转换。

对于各 DAC 通道 x，如果使能 DAC_CR 寄存器中相应的 DMAUDRIEx 位，还将产生中断。

DMA 双数据模式

当 DMA 控制器以正常模式使用时，收到 DMA 请求后只传输 12 位（或 8 位）数据。由于 AHB 宽度为 32 位，因此可以同时传输两个 12 位数据。要使用此模式，应将 DAC_MCR 寄存器的 DMADOUBLEx 位置 1。

当 DMAENx 位置 1 时，每发生两次外部触发（软件触发除外）就会产生一个 DAC DMA 请求：

1. 当检测到第一次触发时，DAC_DHRx 和 DAC_DHRBx 寄存器的值传输到 DAC_DORx 和 DAC_DORBx 寄存器中。实际的 DAC 数据被加载到 DAC_DORx 寄存器中。之后生成一个 DMA 请求。DMA 将新数据写入 DAC_DHRx 和 DAC_DHRBx 数据寄存器。
2. 当检测到第二次触发时，实际的 DAC 数据被加载到 DAC_DHRBx 寄存器中。第二次触发不会生成任何 DMA 请求。DORSTATx 位指示哪个 DOR 数据实际被加载到模拟 DAC 输入中。

在 DMA 双数据模式下，还支持 DMA 下溢功能。

在 DMA 双数据模式下，DMA 请求只能处理一个 DAC 通道。要在 DMA 双数据模式下使用两个通道输出，必须分别配置每个 DMA 通道。

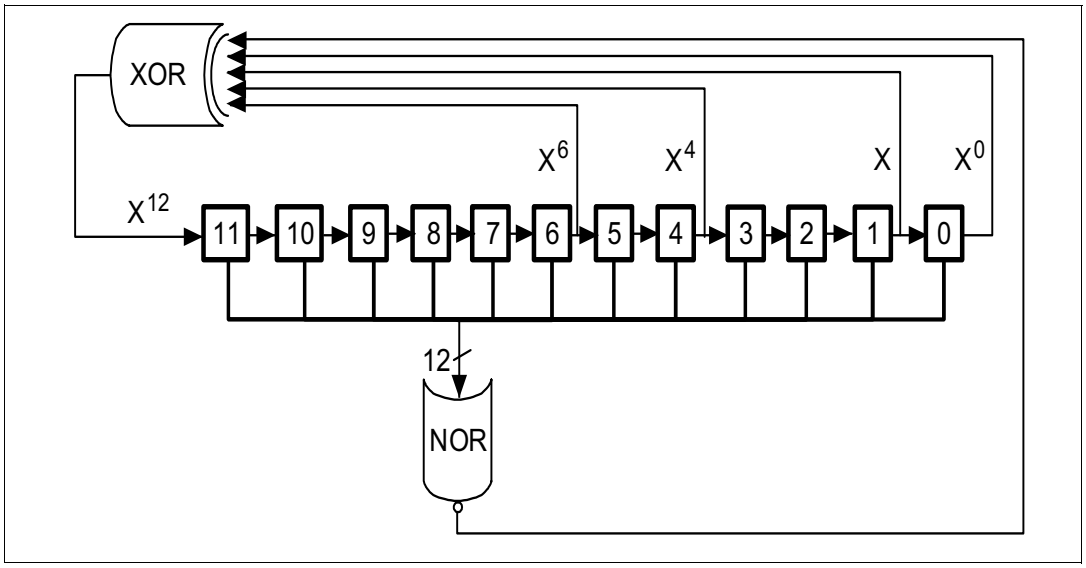
要在双数据模式与单数据模式之间来回切换，必须满足以下条件：

- 必须禁止 DAC。
- 必须清零 DMAEN 位（ENx = 0 且 DMAEN = 0）。

21.4.10 生成噪声

为了生成可变振幅的伪噪声，可使用 LFSR（线性反馈移位寄存器）。将 WAVEx[1:0] 置为 01 即可选择生成 DAC 噪声。LFSR 中的预加载值为 0xAAA。在每次发生触发事件后，经过三个 dac_hclk 时钟周期，该寄存器会依照特定的计算算法完成更新。

图 138. DAC LFSR 寄存器计算算法

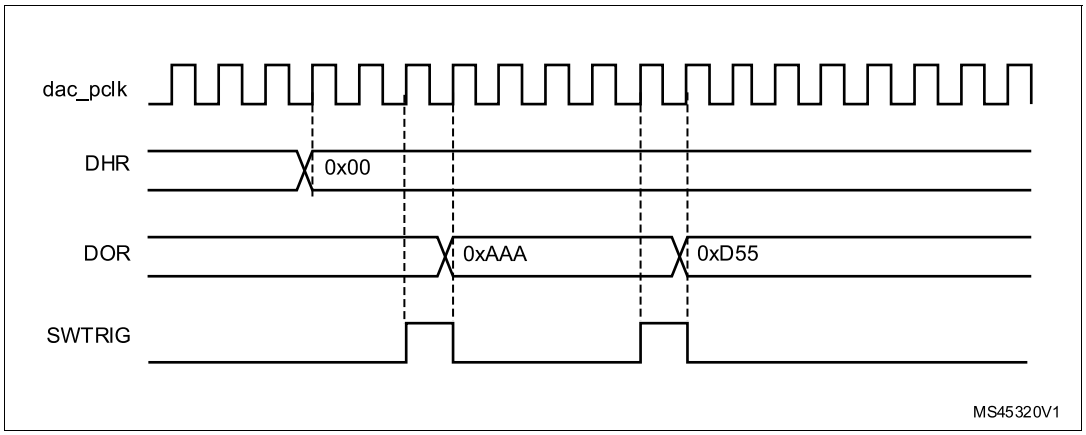


LFSR 值可以通过 DAC_CR 寄存器中的 MAMPx[3:0] 位来部分或完全屏蔽，在不发生溢出的情况下，该值将与 DAC_DHRx 的内容相加，然后传输到 DAC_DORx 寄存器中。

如果 LFSR 为 0x0000，将向其注入“1”（防锁定机制）。

可以通过复位 WAVEx[1:0] 位来将 LFSR 波形生成功能关闭。

图 139. LFSR 生成波形的 DAC 转换（使能软件触发）



注意： *要生成噪声，必须通过将 DAC_CR 寄存器中的 TENx 位置 1 来使能 DAC 触发。*

21.4.11 生成三角波

可以在直流电流或慢变信号上叠加一个小幅三角波。将 WAVEx[1:0] 置为“10”即可选择生成 DAC 三角波。振幅通过 DAC_CR 寄存器中的 MAMPx[3:0] 位进行配置。每次发生触发事件后，经过三个 dac_hclk 时钟周期，内部三角波计数器将会递增。在不发生溢出的情况下，该计数器的值将与 DAC_DHRx 寄存器内容相加，所得总和将传输到 DAC_DORx 寄存器中。只要小于 MAMPx[3:0] 位定义的最大振幅，三角波计数器就会一直递增。一旦达到配置的振幅，计数器将递减至零，然后再递增，以此类推。

可以通过复位 WAVEx[1:0] 位来将三角波生成功能关闭。

图 140. 生成 DAC 三角波

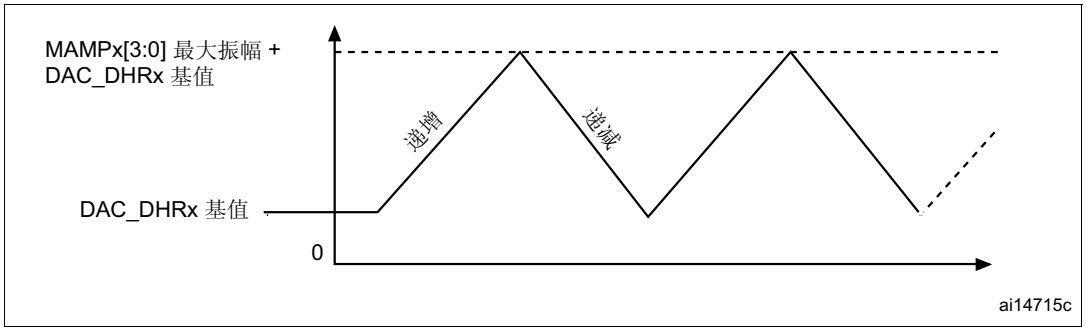
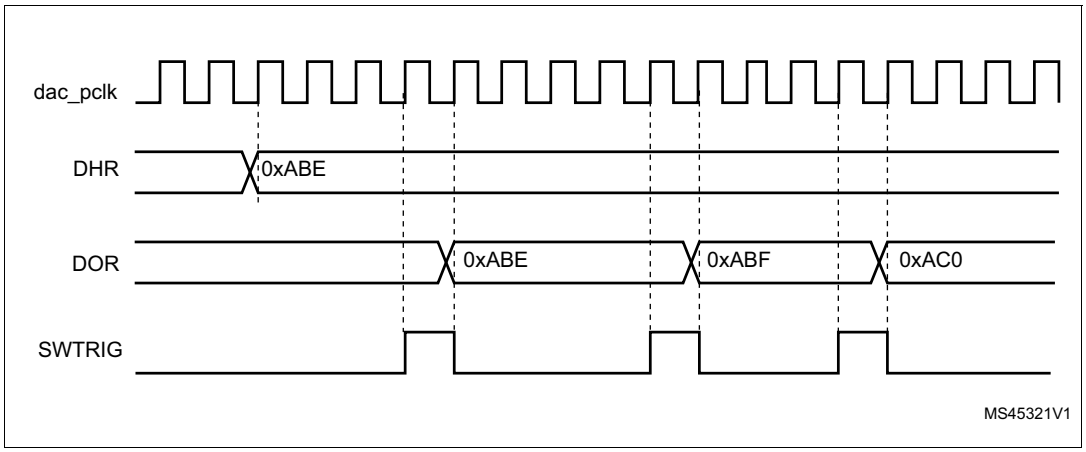


图 141. 生成三角波波形的 DAC 转换（使能软件触发）



注意： *要生成三角波，必须通过将 DAC_CR 寄存器中的 TENx 位置 1 来使能 DAC 触发。MAMPx[3:0] 位必须在使能 DAC 之前进行配置，否则将无法更改。*

21.4.12 DAC 通道模式

每条 DAC 通道均可配置为正常模式或采样和保持模式。可使能输出缓冲器，以实现高驱动能力。使能输出缓冲器之前，需要校准电压偏移。此项校准是在出厂时执行的（复位后加载），可在应用运行期间通过软件进行调整。

正常模式

在正常模式下，可通过更改缓冲器状态以及更改 DACx_OUTy 引脚互连实现四种组合。

要使能输出缓冲器，DAC_MCR 寄存器中的 MODEx[2:0] 位必须满足：

- 000：DAC 连接到外部引脚
- 001：DAC 连接到外部引脚以及片上外设

要禁止输出缓冲器，DAC_MCR 寄存器中的 MODEx[2:0] 位必须满足：

- 010：DAC 连接到外部引脚
- 011：DAC 连接到片上外设

采样和保持模式

在采样和保持模式下，DAC 内核在触发转换后对数据进行转换，然后在电容上保持转换后的电压。不进行转换时，DAC 内核和缓冲器在两次采样之间是完全关闭的，DAC 输出处于三态，因此可降低整体功耗。每次新转换之前都需要一个稳定周期，其值取决于缓冲器状态。

在该模式下，DAC 内核以及所有相应的逻辑和寄存器除了受 dac_hclk 时钟驱动外，还受 LSI/LSE 低速时钟 (dac_hold_ck) 驱动，从而可在深度低功耗模式（如停止模式）下使用 DAC 通道。

使能采样和保持模式时，LSI/LSE 低速时钟 (dac_hold_ck) 不得停止工作。

采样/保持模式运行可分为 3 个阶段：

1. 采样阶段：采样/保持元件充电到所需电压。充电时间取决于电容值（内部还是外部，由用户选择）。采样时间通过 DAC_SHSRx 寄存器中的 TSAMPLEx[9:0] 位进行配置。对 TSAMPLEx[9:0] 位执行写操作期间，DAC_SR 寄存器中的 BWSTx 位会置 1，从而可在两个时钟域（AHB 和低速时钟）之间实现同步，并允许软件在 DAC 通道运行期间更改采样阶段的值
2. 保持阶段：DAC 输出通道处于三态，DAC 内核和缓冲器关闭，以降低电流消耗。保持时间通过 DAC_SHHR 寄存器中的 THOLDx[9:0] 位进行配置
3. 刷新阶段：刷新时间通过 DAC_SHRR 寄存器中的 TREFRESHx[7:0] 位进行配置

以上三个阶段的时间以 LSI/LSE 时钟周期为单位。举例来说，假设选择了 LSI/LSE ~32 KHz，要配置 350 µs 的采样时间，2 ms 的保持时间和 100 µs 的刷新时间：

采样阶段需要 12 个周期：TSAMPLEx[9:0] = 11，

保持阶段需要 62 个周期：THOLDx[9:0] = 62，

刷新阶段需要 4 个周期：TREFRESHx[7:0] = 4

本例中，与正常模式相比，功耗几乎降低了 15 倍。

下表列出了计算正确的采样时间和刷新时间的公式，保持时间取决于漏电流。

表 134. 采样时间和刷新时间

缓冲器状态	t_{SAMP} (1)(2)	$t_{REFRESH}$ (2)(3)
使能	7 µs + (10*R_{BON}*C_{SH})	7 µs + (R_{BON}*C_{SH})*ln(2*N_{LSB})
禁止	3 µs + (10*R_{BOFF}*C_{SH})	3 µs + (R_{BOFF}*C_{SH})*ln(2*N_{LSB})

1. 上述公式中，要实现 12 位分辨率，稳定为理想代码值（½ LSB 或精度）需要 10 个时间常数。要实现 8 位分辨率，稳定时间为 7 个时间常数。
2. C_{SH} 是采样和保持模式下的电容。
3. 保持阶段的容许压降“Vd”通过电容以输出漏电流放电后的 LSB 数表示。稳定回理想值（½ LSB 误差精度）需要 ln(2*Nlsb) 个 DAC 时间常数。

输出缓冲器开启时的采样和刷新时间计算示例

下例中使用的数值仅用于说明目的。有关产品数据，请参见产品数据手册。

C_{SH} = 100 nF

V_{REF+} = 3.0 V

采样阶段：

t_{SAMP} = 7 µs + (10 * 2000 * 100 * 10^{-9}) = 2.007 ms

（其中 R_{BON} = 2 kΩ）

刷新阶段：

$t_{REFRESH}$ = 7 µs + (2000 * 100 * 10^{-9}) * ln(2*10) = 606.1 µs

（其中，N_{LSB} = 10，保持阶段会减少 10 个 LSB）

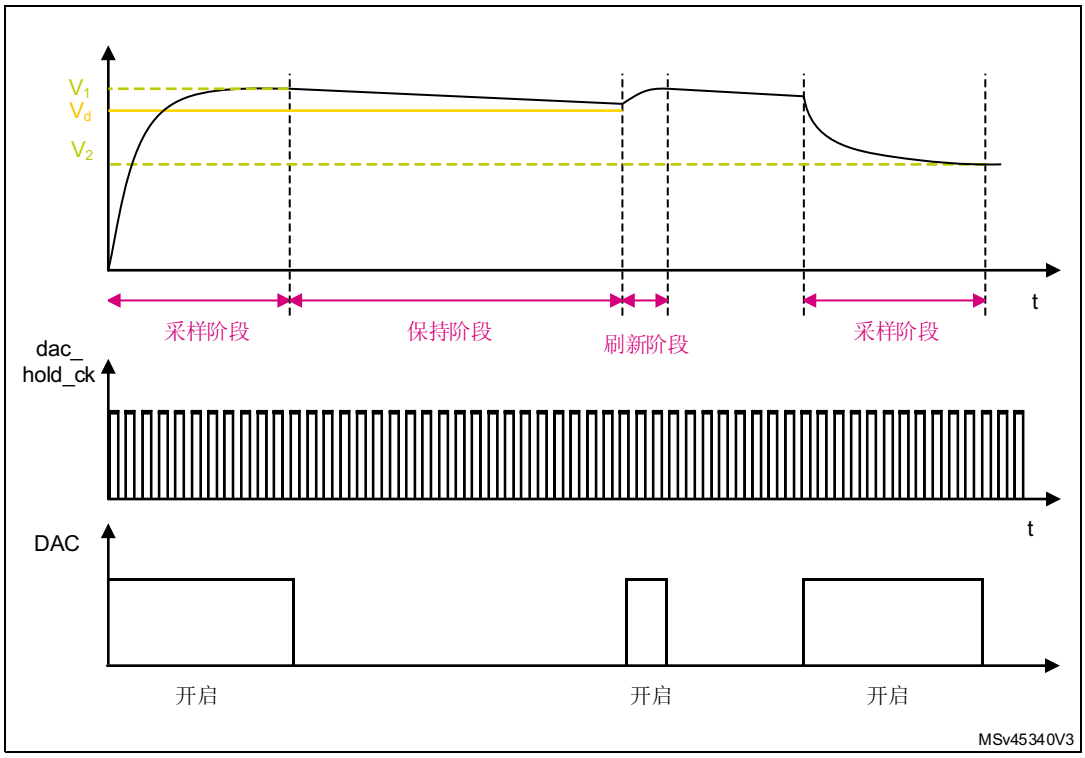
保持阶段：

$D_V = i_{leak} * t_{hold} / C_{SH}$ = 0.0073 V（3 V 时 10 个 LSB 在 12 位分辨率下）

i_{leak} = 150 nA（所有温度范围内 IO 泄漏的最坏情况）

t_{hold} = 0.0073 * 100 * 10^{-9} / (150 * 10^{-9}) = 4.867 ms

图 142. DAC 采样和保持模式阶段图



与正常模式相似，采样和保持模式也具有不同配置。

要使能输出缓冲器，必须将 DAC_MCR 寄存器中的 MODEx[2:0] 位设置为：

- 100：DAC 连接到外部引脚
- 101：DAC 连接到外部引脚以及片上外设

要禁止输出缓冲器，必须将 DAC_MCR 寄存器中的 MODEx[2:0] 位设置为：

- 110：DAC 连接到外部引脚以及片上外设
- 111：DAC 连接到片上外设

  当 MODEx[2:0] 位等于 111 时，内部电容 C_{Lint} 将保持 DAC 内核的电压输出，随后会将该电压驱动到片上外设。

所有采样和保持阶段均可中断，DAC_DHRx 中的任何更改都会立即触发新的采样阶段。

表 135. 通道输出模式汇总

MODEx[2:0]			模式	缓冲器	输出连接
0	0	0	正常模式	使能	连接到外部引脚
0	0	1			连接到外部引脚以及片上外设（如比较器）
0	1	0		禁止	连接到外部引脚
0	1	1			连接到片上外设（如比较器）
1	0	0	采样和保持模式	使能	连接到外部引脚
1	0	1			连接到外部引脚以及片上外设（如比较器）
1	1	0		禁止	连接到外部引脚以及片上外设（如比较器）
1	1	1			连接到片上外设（如比较器）

注意： *某些封装不提供任何 DACx_OUTy 引脚。在这种情况下，应用不得使用以下通道输出模式：缓冲器连接到封装上不存在的外部引脚。更多详细信息，请参见产品数据手册。*

21.4.13 DAC 通道缓冲器校准

N 位数模转换器 (DAC) 的传递函数为：

$$V_{out} = ((D/2^N) \times G \times V_{ref}) + V_{OS}$$

其中，V_{OUT} 为模拟输出，D 为数字输入，G 为增益，V_{ref} 为标称满量程电压，V_{os} 为偏移电压。对于理想的 DAC 通道，G = 1 且 V_{os} = 0。

由于输出缓冲器所具有的特性，不同芯片的电压偏移可能有所不同，并会在模拟输出上引入绝对偏移误差。为补偿 V_{os}，需要通过调整技术进行校准。

仅当 DAC 通道x 在缓冲器使能（MODEx[2:0] = 0b000 或 0b001 或 0b100 或 0b101）的条件下运行时，校准才有效。如果在缓冲器关闭的条件下应用于其他模式，校准无效。校准过程中：

- 缓冲器输出与引脚内部/外部连接断开，并会进入三态模式 (HiZ)。
- 缓冲器作为比较器来检测中间代码值 0x800，并通过内部桥将其与 VREF+/2 信号进行比较，然后根据比较结果（CAL_FLAGx 位）将其输出信号切换为 0 或 1

提供以下两种校准技术：

- 出厂调整（默认设置）

  DAC 缓冲器偏移是在出厂时进行调整的。DAC_CCR 寄存器中 OTRIMx[4:0] 位的默认值是出厂调整值，会在 DAC 数字接口复位时载入。

- 用户调整

  如果运行条件不同于标称出厂调整条件，特别是 V_{DDA} 电压、温度、VREF+ 值发生变化时，可进行用户调整，并且用户可在应用过程中随时通过软件进行调整。

注意： *更多关于标称出厂调整条件的详细信息，请参见数据手册。*

另外，如果 V_{DD} 移除（例如器件进入 STANDBY 或 VBAT 模式），则需要进行校准。

执行用户调整校准的步骤如下：

1. 如果 DAC 通道激活，可向 DAC_CR 中的 ENx 位写入 0 来禁止通道。
2. 写入 DAC_MCR 寄存器，选择使能了缓冲器的模式（MODEx[2:0] = 0b000、0b001、0b100 或 0b101）。
3. 将 DAC_CR 寄存器中的 CENx 位置 1，开始进行 DAC 通道 x 校准。
4. 应用调整算法：
   a) 向 OTRIMx[4:0] 位写入以 0b00000 开头的代码。
   b) 等待 t_{TRIM} 延迟。
   c) 检查 DAC_SR 中的 CAL_FLAGx 位是否已置 1。
   d) 如果 CAL_FLAGx 已置 1，调整代码 OTRIMx[4:0] 已找到，并可在*器件*运行过程中用来补偿输出值，否则会使 OTRIMx[4:0] 递增并重复执行 (a) 到 (d) 的步骤。

软件算法可使用逐次逼近法或二分法较快地计算和设置 OTRIMx[4:0] 位的内容。

CAL_FLAGx 位的换向/切换指示偏移已正确补偿，相应的调整代码必须保留在 DAC_CCR 寄存器的 OTRIMx[4:0] 位中。

注意： *为了获得正确的值，对 DAC_SR 寄存器中的 OTRIMx[4:0] 位进行的写操作与对 CAL_FLAGx 位进行的读操作之间必须保留 t_{TRIM} 延迟。此参数在数据手册的电气特性部分进行了说明。*

如果 V_{DDA}、VREF+ 和温度条件在器件运行期间不发生更改，同时器件更为频繁地进入待机和 VBAT 模式，软件可将第一次用户校准时找到的 OTRIMx[4:0] 位存储在 flash 或备用寄存器中。然后在器件电源再次恢复时直接加载/写入这些位，从而无需等待新的校准时间。

当 CENx 位置 1 时，不允许将 ENx 位置 1。

21.4.14 DAC 双通道转换模式（如果有两个通道可用）

为了在同时需要两个 DAC 通道的应用中有效利用总线带宽，DAC 模块实现了三个双寄存器：DHR8RD、DHR12RD 和 DHR12LD。这样，只需一个寄存器访问即可同时驱动两个 DAC 通道。要生成相应波形，无需访问 DHRxxxD 寄存器。因此，可单独也可同时使用两个输出通道。

通过两个 DAC 通道和这三个双寄存器可以实现 15 种转换模式。但如果需要，所有这些转换模式也都可以通过单独的 DHRx 寄存器来实现。

下面几段内容将介绍所有这些模式。

独立触发（不产生波形）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
2. 将 TSEL1 位域和 TSEL2 位域设置为不同的值，以配置不同的触发源。
3. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

DAC 通道 1 触发信号到达时，DHR1 寄存器的内容传输到 DAC_DOR1（三个 dac_hclk 时钟周期之后）。

DAC 通道 2 触发信号到达时，DHR2 寄存器的内容传输到 DAC_DOR2（三个 dac_hclk 时钟周期之后）。

独立触发（生成单个 LFSR）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
2. 将 TSEL1 位域和 TSEL2 位域设置为不同的值，以配置不同的触发源。
3. 将两个 DAC 通道的 WAVEx[1:0] 设置为 01，并在 MAMPx[3:0] 位中配置相同的 LFSR 掩码值。
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

DAC 通道 1 触发信号到达时，LFSR1 计数器内容（使用相同的掩码）与 DHR1 寄存器内容相加，所得总和传输到 DAC_DOR1 中（三个 dac_hclk 时钟周期之后）。LFSR1 计数器随即更新。

DAC 通道 2 触发信号到达时，LFSR2 计数器内容（使用相同的掩码）与 DHR2 寄存器内容相加，所得总和传输到 DAC_DOR2 中（三个 dac_hclk 时钟周期之后）。LFSR2 计数器随即更新。

独立触发（生成不同 LFSR）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
2. 将 TSEL1 位域和 TSEL2 位域设置为不同的值，以配置不同的触发源。
3. 将两个 DAC 通道的 WAVEx[1:0] 设置为 01，并在 MAMP1[3:0] 和 MAMP2[3:0] 位中设置不同的 LFSR 掩码值。
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

DAC 通道 1 触发信号到达时，LFSR1 计数器内容（使用 MAMP1[3:0] 配置的掩码）与 DHR1 寄存器内容相加，所得总和传输到 DAC_DOR1 中（三个 dac_hclk 时钟周期之后）。LFSR1 计数器随即更新。

DAC 通道 2 触发信号到达时，LFSR2 计数器内容（使用 MAMP2[3:0] 配置的掩码）与 DHR2 寄存器内容相加，所得总和传输到 DAC_DOR2 中（三个 dac_hclk 时钟周期之后）。LFSR2 计数器随即更新。

独立触发（生成单个三角波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
2. 将 TSEL1 位域和 TSEL2 位域设置为不同的值，以配置不同的触发源。
3. 将两个 DAC 通道的 WAVEx[1:0] 设置为 1x，并在 MAMPx[3:0] 位中配置相同的最大振幅值。
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

DAC 通道 1 触发信号到达时，DAC 通道 1 三角波计数器内容（使用相同的三角波振幅）与 DHR1 寄存器内容相加，所得总和传输到 DAC_DOR1 中（三个 dac_hclk 时钟周期之后）。DAC 通道 1 三角波计数器随即更新。

DAC 通道 2 触发信号到达时，DAC 通道 2 三角波计数器内容（使用相同的三角波振幅）与 DHR2 寄存器内容相加，所得总和传输到 DAC_DOR2 中（三个 dac_hclk 时钟周期之后）。DAC 通道 2 三角波计数器随即更新。

独立触发（生成不同三角波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
2. 将 TSEL1 位和 TSEL2 位设置为不同的值，以配置不同的触发源。
3. 将两个 DAC 通道的 WAVEx[1:0] 设置为 1x，并在 MAMP1[3:0] 和 MAMP2[3:0] 位中设置不同的最大振幅值。
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

DAC 通道 1 触发信号到达时，DAC 通道 1 三角波计数器内容（使用 MAMP1[3:0] 配置的三角波振幅）与 DHR1 寄存器内容相加，所得总和传输到 DAC_DOR1 中（三个 dac_hclk 时钟周期之后）。DAC 通道 1 三角波计数器随即更新。

DAC 通道 2 触发信号到达时，DAC 通道 2 三角波计数器内容（使用 MAMP2[3:0] 配置的三角波振幅）与 DHR2 寄存器内容相加，所得总和传输到 DAC_DOR2 中（三个 dac_hclk 时钟周期之后）。DAC 通道 2 三角波计数器随即更新。

同步软件启动

要将 DAC 配置为此转换模式，需要遵循以下顺序：

- 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

在此配置中，DHR1 和 DHR2 寄存器内容会在一个 dac_hclk 时钟周期后分别传输到 DAC_DOR1 和 DAC_DOR2 中。

同步触发（不产生波形）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
2. 将 TSEL1 位域和 TSEL2 位域设置为相同的值，以便为两个 DAC 通道配置相同的触发源。
3. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

当触发信号到达时，DHR1 和 DHR2 寄存器内容将分别传输到 DAC_DOR1 和 DAC_DOR2 中（三个 dac_hclk 时钟周期之后）。

同步触发（生成单个 LFSR）

1. 要将 DAC 配置为此转换模式，需要遵循以下顺序：
2. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
3. 将 TSEL1 位域和 TSEL2 位域设置为相同的值，以便为两个 DAC 通道配置相同的触发源。
4. 将两个 DAC 通道的 WAVEx[1:0] 设置为 01，并在 MAMPx[3:0] 位中配置相同的 LFSR 掩码值。
5. 将 DAC 双通道数据加载到所需 DHR 寄存器（DHR12RD、DHR12LD 或 DHR8RD）。

触发信号到达时，LFSR1 计数器内容（使用相同的掩码）与 DHR1 寄存器内容相加，所得总和传输到 DAC_DOR1 中（三个 dac_hclk 时钟周期之后）。LFSR1 计数器随即更新。同时，LFSR2 计数器内容（使用相同的掩码）与 DHR2 寄存器内容相加，所得总和传输到 DAC_DOR2 中（三个 dac_hclk 时钟周期之后）。LFSR2 计数器随即更新。

同步触发（生成不同 LFSR）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
2. 将 TSEL1 位域和 TSEL2 位域设置为相同的值，以便为两个 DAC 通道配置相同的触发源。
3. 将两个 DAC 通道的 WAVEx[1:0] 设置为 01，并在 MAMP1[3:0] 和 MAMP2[3:0] 位中设置不同的 LFSR 掩码值。
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

触发信号到达时，LFSR1 计数器内容（使用 MAMP1[3:0] 配置的掩码）与 DHR1 寄存器内容相加，所得总和传输到 DAC_DOR1 中（三个 dac_hclk 时钟周期之后）。LFSR1 计数器随即更新。
同时，LFSR2 计数器内容（使用 MAMP2[3:0] 配置的掩码）与 DHR2 寄存器内容相加，所得总和传输到 DAC_DOR2 中（三个 dac_hclk 时钟周期之后）。LFSR2 计数器随即更新。

同步触发（生成单个三角波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
2. 将 TSEL1 位域和 TSEL2 位域设置为相同的值，以便为两个 DAC 通道配置相同的触发源。
3. 将两个 DAC 通道的 WAVEx[1:0] 设置为 1x，并在 MAMPx[3:0] 位中配置相同的最大振幅值。
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

触发信号到达时，DAC 通道 1 三角波计数器内容（使用相同的三角波振幅）与 DHR1 寄存器内容相加，所得总和传输到 DAC_DOR1 中（三个 dac_hclk 时钟周期之后）。DAC 通道 1 三角波计数器随即更新。

同时，DAC 通道 2 三角波计数器内容（使用相同的三角波振幅）与 DHR2 寄存器内容相加，所得总和传输到 DAC_DOR2 中（三个 dac_hclk 时钟周期之后）。DAC 通道 2 三角波计数器随即更新。

同步触发（生成不同三角波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1。
2. 将 TSEL1 位域和 TSEL2 位域设置为相同的值，以便为两个 DAC 通道配置相同的触发源。
3. 将两个 DAC 通道的 WAVEx[1:0] 设置为 1x，并在 MAMP1[3:0] 和 MAMP2[3:0] 位中设置不同的最大振幅值。
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）。

触发信号到达时，DAC 通道 1 三角波计数器内容（使用 MAMP1[3:0] 配置的三角波振幅）与 DHR1 寄存器内容相加，所得总和传输到 DAC_DOR1 中（三个 AHB 时钟周期之后）。DAC 通道 1 三角波计数器随即更新。

同时，DAC 通道 2 三角波计数器内容（使用 MAMP2[3:0] 配置的三角波振幅）与 DHR2 寄存器内容相加，所得总和传输到 DAC_DOR2 中（三个 dac_hclk 时钟周期之后）。DAC 通道 2 三角波计数器随即更新。

21.5 低功耗模式下的 DAC

表 136. 低功耗模式对 DAC 的影响

模式	说明
睡眠	无影响，DAC 与 DMA 结合使用。
停止(1)	DAC 以静态值继续工作。在使用 LSE/LSI 时钟的情况下，可以选择采样和保持模式。
待机	DAC 外设掉电，退出待机模式后必须重新初始化。

1. 有关 DAC 外设支持的停止模式的信息，请参见*第 21.3 节：DAC 实现*。

21.6 DAC 中断

表 137. DAC 中断

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止模式	退出待机模式
DAC	DMA 下溢	DMAUDRx	DMAUDRIEx	写入 DMAUDRx = 1	是	否	否

21.7 DAC 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1 章*。

外设寄存器必须按字（32 位）进行访问。

21.7.1 DAC 控制寄存器 (DAC_CR)

DAC control register

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	CEN2	DMAUDRIE2	DMAEN2	MAMP2[3:0]				WAVE2[1:0]		TSEL2[3]	TSEL2[2]	TSEL2[1]	TSEL2[0]	TEN2	EN2
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	CEN1	DMAUDRIE1	DMAEN1	MAMP1[3:0]				WAVE1[1:0]		TSEL1[3]	TSEL1[2]	TSEL1[1]	TSEL1[0]	TEN1	EN1
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30 **CEN2**：DAC 通道 2 校准使能 (DAC channel2 calibration enable)

该位由软件置 1 和清零，用于使能/禁止 DAC 通道 2 校准，仅当 DAC_CR 中的位 EN2 设置为 0 时才可写入（仅当 DAC 通道已禁止时才可以进入/退出校准模式），否则会忽略写操作。

0：DAC 通道 2 处于正常运行模式

1：DAC 通道 2 处于校准模式

注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 29 **DMAUDRIE2**：DAC 通道 2 DMA 下溢中断使能 (DAC channel2 DMA underrun interrupt enable)

该位由软件置 1 和清零。

0：禁止 DAC 通道 2 DMA 下溢中断

1：使能 DAC 通道 2 DMA 下溢中断

注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 28 **DMAEN2**：DAC 通道 2 DMA 使能 (DAC channel2 DMA enable)

该位由软件置 1 和清零。

0：禁止 DAC 通道 2 DMA 模式

1：使能 DAC 通道 2 DMA 模式

注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 27:24 **MAMP2[3:0]**：DAC 通道 2 掩码/振幅选择器 (DAC channel2 mask/amplitude selector)

这些位由软件写入，用于在生成噪声波模式下选择掩码，或者在生成三角波模式下选择振幅。

0000：不屏蔽 LFSR 的位 0/三角波振幅等于 1

0001：不屏蔽 LFSR 的位 [1:0]/三角波振幅等于 3

0010：不屏蔽 LFSR 的位 [2:0]/三角波振幅等于 7

0011：不屏蔽 LFSR 的位 [3:0]/三角波振幅等于 15

0100：不屏蔽 LFSR 的位 [4:0]/三角波振幅等于 31

0101：不屏蔽 LFSR 的位 [5:0]/三角波振幅等于 63

0110：不屏蔽 LFSR 的位 [6:0]/三角波振幅等于 127

0111：不屏蔽 LFSR 的位 [7:0]/三角波振幅等于 255

1000：不屏蔽 LFSR 的位 [8:0]/三角波振幅等于 511

1001：不屏蔽 LFSR 的位 [9:0]/三角波振幅等于 1023

1010：不屏蔽 LFSR 的位 [10:0]/三角波振幅等于 2047

≥ 1011：不屏蔽 LFSR 的位 [11:0]/三角波振幅等于 4095

注意：这些位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 23:22 **WAVE2[1:0]**：DAC 通道 2 噪声/三角波生成使能 (DAC channel2 noise/triangle wave generation enable)

这些位由软件置 1 或清零。

00：禁止生成波

01：使能生成噪声波

1x：使能生成三角波

注意：只在位 TEN2 = 1（使能 DAC 通道 2 触发）时使用

这些位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 21:18 **TSEL2[3:0]**：DAC 通道 2 触发选择 (DAC channel2 trigger selection)

这些位用于选择 DAC 通道 2 的外部触发事件

0000：SWTRIG2

0001：dac_ch2_trg1

0010：dac_ch2_trg2

...

1111：dac_ch2_trg15

有关触发配置和映射的详细信息，请参见*第 21.4.2 节：DAC 引脚和内部信号*中的触发选择表。

注意：只在位 TEN2 = 1（使能 DAC 通道 2 触发）时使用。

这些位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 17 **TEN2**：DAC 通道 2 触发使能 (DAC channel2 trigger enable)

该位由软件置 1 和清零，用于使能/禁止 DAC 通道 2 触发

0：禁止 DAC 通道 2 触发，写入 DAC_DHR2 寄存器的数据在一个 dac_hclk 时钟周期之后传输到 DAC_DOR2 寄存器

1：使能 DAC 通道 2 触发，DAC_DHR2 寄存器的数据在三个 dac_hclk 时钟周期之后传输到 DAC_DOR2 寄存器

注意：如果选择软件触发，DAC_DHR2 寄存器的内容只需一个 dac_hclk 时钟周期即可传输到 DAC_DOR2 寄存器。

这些位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 16 **EN2**：DAC 通道 2 使能 (DAC channel2 enable)

该位由软件置 1 和清零，用于使能/禁止 DAC 通道 2。

0：禁止 DAC 通道 2

1：使能 DAC 通道 2

注意：这些位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 15 保留，必须保持复位值。

位 14 **CEN1**：DAC 通道 1 校准使能 (DAC channel1 calibration enable)

该位由软件置 1 和清零，用于使能/禁止 DAC 通道 1 校准，仅当 DAC_CR 中的位 EN1 = 0 时才可写入（仅当 DAC 通道已禁止时才可以进入/退出校准模式），否则会忽略写操作。

0：DAC 通道 1 处于正常运行模式

1：DAC 通道 1 处于校准模式

位 13 **DMAUDRIE1**：DAC 通道 1 DMA 下溢中断使能 (DAC channel1 DMA Underrun Interrupt enable)

该位由软件置 1 和清零。

0：禁止 DAC 通道 1 DMA 下溢中断

1：使能 DAC 通道 1 DMA 下溢中断

位 12 **DMAEN1**：DAC 通道 1 DMA 使能 (DAC channel1 DMA enable)

该位由软件置 1 和清零。

0：禁止 DAC 通道 1 DMA 模式

1：使能 DAC 通道 1 DMA 模式

位 11:8 **MAMP1[3:0]**：DAC 通道 1 掩码/振幅选择器 (DAC channel1 mask/amplitude selector)

这些位由软件写入，用于在生成噪声波模式下选择掩码，或者在生成三角波模式下选择振幅。

0000：不屏蔽 LFSR 的位 0/三角波振幅等于 1

0001：不屏蔽 LFSR 的位 [1:0]/三角波振幅等于 3

0010：不屏蔽 LFSR 的位 [2:0]/三角波振幅等于 7

0011：不屏蔽 LFSR 的位 [3:0]/三角波振幅等于 15

0100：不屏蔽 LFSR 的位 [4:0]/三角波振幅等于 31

0101：不屏蔽 LFSR 的位 [5:0]/三角波振幅等于 63

0110：不屏蔽 LFSR 的位 [6:0]/三角波振幅等于 127

0111：不屏蔽 LFSR 的位 [7:0]/三角波振幅等于 255

1000：不屏蔽 LFSR 的位 [8:0]/三角波振幅等于 511

1001：不屏蔽 LFSR 的位 [9:0]/三角波振幅等于 1023

1010：不屏蔽 LFSR 的位 [10:0]/三角波振幅等于 2047

≥ 1011：不屏蔽 LFSR 的位 [11:0]/三角波振幅等于 4095

位 7:6 **WAVE1[1:0]**：DAC 通道 1 噪声/三角波生成使能 (DAC channel1 noise/triangle wave generation enable)

这些位由软件置 1 和清零。

00：禁止生成波

01：使能生成噪声波

1x：使能生成三角波

只在位 TEN1 = 1（使能 DAC 通道 1 触发）时使用。

位 5:2 **TSEL1[3:0]**：DAC 通道 1 触发选择 (DAC channel1 trigger selection)

这些位用于选择 DAC 通道 1 的外部触发事件。

0000：SWTRIG1

0001：dac_ch1_trg1

0010：dac_ch1_trg2

...

1111：dac_ch1_trg15

有关触发配置和映射的详细信息，请参见*第 21.4.2 节：DAC 引脚和内部信号*中的触发选择表。

注意：只在位 TEN1 = 1（使能 DAC 通道 1 触发）时使用。

位 1 **TEN1**：DAC 通道 1 触发使能 (DAC channel1 trigger enable)

该位由软件置 1 和清零，用于使能/禁止 DAC 通道 1 触发。

0：禁止 DAC 通道 1 触发，写入 DAC_DHR1 寄存器的数据在一个 dac_hclk 时钟周期之后传输到 DAC_DOR1 寄存器

1：使能 DAC 通道 1 触发，DAC_DHR1 寄存器的数据在三个 dac_hclk 时钟周期之后传输到 DAC_DOR1 寄存器

注意：如果选择软件触发，DAC_DHR1 寄存器的内容只需一个 dac_hclk 时钟周期即可传输到 DAC_DOR1 寄存器。

位 0 **EN1**：DAC 通道 1 使能 (DAC channel1 enable)

该位由软件置 1 和清零，用于使能/禁止 DAC 通道 1。

0：禁止 DAC 通道 1

1：使能 DAC 通道 1

21.7.2 DAC 软件触发寄存器 (DAC_SWTRGR)

DAC software trigger register

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWTRIG2	SWTRIG1
														w	w

位 31:2 保留，必须保持复位值。

位 1 **SWTRIG2**：DAC 通道 2 软件触发 (DAC channel2 software trigger)

该位由软件置 1，用于在软件触发模式下触发 DAC。

0：不触发

1：触发

注意：一旦 DAC_DHR2 寄存器值加载到 DAC_DOR2 寄存器中，该位即会由硬件清零（一个 dac_hclk 时钟周期之后）。

该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 0 **SWTRIG1**：DAC 通道 1 软件触发 (DAC channel1 software trigger)

该位由软件置 1，用于在软件触发模式下触发 DAC。

0：不触发

1：触发

注意：一旦 DAC_DHR1 寄存器值加载到 DAC_DOR1 寄存器中，该位即会由硬件清零（一个 dac_hclk 时钟周期之后）。

21.7.3 DAC 通道 1 12 位右对齐数据保持寄存器 (DAC_DHR12R1)

DAC channel1 12-bit right-aligned data holding register

偏移地址：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	DACC1DHRB[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DACC1DHR[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27:16 **DACC1DHRB[11:0]**：DAC 通道 1 12 位右对齐数据 B (DAC channel1 12-bit right-aligned data B)

这些位由软件写入，用于在 DAC 工作于双数据模式时为 DAC 通道 1 指定 12 位数据。

位 15:12 保留，必须保持复位值。

位 11:0 **DACC1DHR[11:0]**：DAC 通道 1 12 位右对齐数据 (DAC channel1 12-bit right-aligned data)

这些位由软件写入，用于为 DAC 通道 1 指定 12 位数据。

21.7.4 DAC 通道 1 12 位左对齐数据保持寄存器 (DAC_DHR12L1)

DAC channel1 12-bit left aligned data holding register

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
DACC1DHRB[11:0]												Res.	Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC1DHR[11:0]												Res.	Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw				

位 31:20 **DACC1DHRB[11:0]**：DAC 通道 1 12 位左对齐数据 B (DAC channel1 12-bit left-aligned data B)

这些位由软件写入，用于在 DAC 工作于双数据模式时为 DAC 通道 1 指定 12 位数据。

位 19:16 保留，必须保持复位值。

位 15:4 **DACC1DHR[11:0]**：DAC 通道 1 12 位左对齐数据 (DAC channel1 12-bit left-aligned data)

这些位由软件写入，

用于为 DAC 通道 1 指定 12 位数据。

位 3:0 保留，必须保持复位值。

21.7.5 DAC 通道 1 8 位右对齐数据保持寄存器 (DAC_DHR8R1)

DAC channel1 8-bit right aligned data holding register

偏移地址：0x10

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC1DHRB[7:0]								DACC1DHR[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:8 **DACC1DHRB[7:0]**：DAC 通道 1 8 位右对齐数据 (DAC channel1 8-bit right-aligned data)
这些位由软件写入，用于在 DAC 工作于双数据模式时为 DAC 通道 1 指定 8 位数据。

位 7:0 **DACC1DHR[7:0]**：DAC 通道 1 8 位右对齐数据 (DAC channel1 8-bit right-aligned data)
这些位由软件写入，用于为 DAC 通道 1 指定 8 位数据。

21.7.6 DAC 通道 2 12 位右对齐数据保持寄存器 (DAC_DHR12R2)

DAC channel2 12-bit right aligned data holding register

此寄存器仅适用于双通道 DAC。请参见*第 21.3 节：DAC 实现*。

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	DACC2DHRB[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DACC2DHR[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27:16 **DACC2DHRB[11:0]**：DAC 通道 2 12 位右对齐数据 (DAC channel2 12-bit right-aligned data)
这些位由软件写入，用于在 DAC 工作于 DMA 双数据模式时为 DAC 通道 2 指定 12 位数据。

位 15:12 保留，必须保持复位值。

位 11:0 **DACC2DHR[11:0]**：DAC 通道 2 12 位右对齐数据 (DAC channel2 12-bit right-aligned data)
这些位由软件写入，用于为 DAC 通道 2 指定 12 位数据。

21.7.7 DAC 通道 2 12 位左对齐数据保持寄存器 (DAC_DHR12L2)

DAC channel2 12-bit left aligned data holding register

此寄存器仅适用于双通道 DAC。请参见*第 21.3 节：DAC 实现*。

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DACC2DHRB[11:0]												Res.	Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC2DHR[11:0]												Res.	Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw				

位 31:20 **DACC2DHRB[11:0]**：DAC 通道 2 12 位左对齐数据 B (DAC channel2 12-bit left-aligned data B)
这些位由软件写入，用于在 DAC 工作于双数据模式时为 DAC 通道 2 指定 12 位数据。

位 19:16 保留，必须保持复位值。

位 15:4 **DACC2DHR[11:0]**：DAC 通道 2 12 位左对齐数据 (DAC channel2 12-bit left-aligned data)
这些位由软件写入，用于为 DAC 通道 2 指定 12 位数据。

位 3:0 保留，必须保持复位值。

21.7.8 DAC 通道 2 8 位右对齐数据保持寄存器 (DAC_DHR8R2)

DAC channel2 8-bit right-aligned data holding register

此寄存器仅适用于双通道 DAC。请参见*第 21.3 节：DAC 实现*。

偏移地址：0x1C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC2DHRB[7:0]								DACC2DHR[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:8 **DACC2DHRB[7:0]**：DAC 通道 2 8 位右对齐数据 (DAC channel2 8-bit right-aligned data)
这些位由软件写入，用于在 DAC 工作于双数据模式时为 DAC 通道 2 指定 8 位数据。

位 7:0 **DACC2DHR[7:0]**：DAC 通道 2 8 位右对齐数据 (DAC channel2 8-bit right-aligned data)
这些位由软件写入，用于为 DAC 通道 2 指定 8 位数据。

21.7.9 双 DAC 12 位右对齐数据保持寄存器 (DAC_DHR12RD)

Dual DAC 12-bit right-aligned data holding register

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	DACC2DHR[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DACC1DHR[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27:16 **DACC2DHR[11:0]**：DAC 通道 2 12 位右对齐数据 (DAC channel2 12-bit right-aligned data)
这些位由软件写入，用于为 DAC 通道 2 指定 12 位数据。

位 15:12 保留，必须保持复位值。

位 11:0 **DACC1DHR[11:0]**：DAC 通道 1 12 位右对齐数据 (DAC channel1 12-bit right-aligned data)
这些位由软件写入，用于为 DAC 通道 1 指定 12 位数据。

21.7.10 双 DAC 12 位左对齐数据保持寄存器 (DAC_DHR12LD)

Dual DAC 12-bit left aligned data holding register

偏移地址：0x24

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DACC2DHR[11:0]												Res.	Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC1DHR[11:0]												Res.	Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw				

位 31:20 **DACC2DHR[11:0]**：DAC 通道 2 12 位左对齐数据 (DAC channel2 12-bit left-aligned data)
这些位由软件写入，用于为 DAC 通道 2 指定 12 位数据。

位 19:16 保留，必须保持复位值。

位 15:4 **DACC1DHR[11:0]**：DAC 通道 1 12 位左对齐数据 (DAC channel1 12-bit left-aligned data)
这些位由软件写入，用于为 DAC 通道 1 指定 12 位数据。

位 3:0 保留，必须保持复位值。

21.7.11 双 DAC 8 位右对齐数据保持寄存器 (DAC_DHR8RD)

Dual DAC 8-bit right aligned data holding register

偏移地址：0x28

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC2DHR[7:0]								DACC1DHR[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:8 **DACC2DHR[7:0]**：DAC 通道 2 8 位右对齐数据 (DAC channel2 8-bit right-aligned data)
这些位由软件写入，用于为 DAC 通道 2 指定 8 位数据。

位 7:0 **DACC1DHR[7:0]**：DAC 通道 1 8 位右对齐数据 (DAC channel1 8-bit right-aligned data)
这些位由软件写入，用于为 DAC 通道 1 指定 8 位数据。

21.7.12 DAC 通道 1 数据输出寄存器 (DAC_DOR1)

DAC channel1 data output register

偏移地址：0x2C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	DACC1DORB[11:0]											
				r	r	r	r	r	r	r	r	r	r	r	r

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

位 31:28 保留，必须保持复位值。

位 27:16 **DACC1DORB[11:0]**：DAC 通道 1 数据输出 (DAC channel1 data output)
这些位为只读，其中包含 DAC 通道 1 B 的数据输出。

位 15:12 保留，必须保持复位值。

位 11:0 **DACC1DOR[11:0]**：DAC 通道 1 数据输出 (DAC channel1 data output)
这些位为只读，其中包含 DAC 通道 1 的数据输出。

21.7.13 DAC 通道 2 数据输出寄存器 (DAC_DOR2)

DAC channel2 data output register

此寄存器仅适用于双通道 DAC。请参见*第 21.3 节：DAC 实现*。

偏移地址：0x30

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	DACC2DORB[11:0]											
				r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DACC2DOR[11:0]											
				r	r	r	r	r	r	r	r	r	r	r	r

位 31:28 保留，必须保持复位值。

位 27:16 **DACC2DORB[11:0]**：DAC 通道 2 数据输出 (DAC channel2 data output)
这些位为只读，其中包含 DAC 通道 2 B 的数据输出。

位 15:12 保留，必须保持复位值。

位 11:0 **DACC2DOR[11:0]**：DAC 通道 2 数据输出 (DAC channel2 data output)
这些位为只读，其中包含 DAC 通道 2 的数据输出。

21.7.14 DAC 状态寄存器 (DAC_SR)

DAC status register

偏移地址：0x34

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BWST2	CAL_FLAG2	DMAUDR2	DORSTAT2	DAC2RDY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r	rc_w1	r	r											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BWST1	CAL_FLAG1	DMAUDR1	DORSTAT1	DAC1RDY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r	rc_w1	r	r											

位 31 **BWST2**：DAC 通道 2 写入采样时间忙标志 (DAC channel2 busy writing sample time flag)
该位会在采样和保持模式使能后由系统置 1。每次软件写入 DAC_SHSR2 寄存器时，该位都会置 1，对 DAC_SHSR2 的写操作完成后，该位通过硬件清零。（需要约 3 个 LSI/LSE 同步周期）。
0：目前未对 DAC_SHSR2 进行写操作：DAC_SHSR2 可写入
1：目前正在对 DAC_SHSR2 进行写操作：DAC_SHSR2 不可写入
注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 30 **CAL_FLAG2**：DAC 通道 2 校准偏移状态 (DAC channel2 calibration offset status)
该位由硬件置 1 和清零
0：校准调整值小于偏移校正值
1：校准调整值等于或大于偏移校正值
注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 29 **DMAUDR2**：DAC 通道 2 DMA 下溢标志 (DAC channel2 DMA underrun flag)
该位由硬件置 1，由软件清零（写入 1）。
0：DAC 通道 2 未发生 DMA 下溢错误状况
1：DAC 通道 2 发生 DMA 下溢错误状况（当前所选触发源以高于 DMA 服务能力的频率驱动 DAC 通道 2 转换）。
注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 28 **DORSTAT2**：DAC 通道 2 输出寄存器状态位 (DAC channel2 output register status bit)
该位由硬件置 1 和清零。该位仅适用于 DAC 在双数据模式下工作时。
0：DOR[11:0] 为实际使用的 DAC 输出
1：DORB[11:0] 为实际使用的 DAC 输出
注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 27 **DAC2RDY**：DAC 通道 2 就绪状态位 (DAC channel2 ready status bit)
该位由硬件置 1 和清零。
0：DAC 通道 2 尚未准备好接受触发或输出数据
1：DAC 通道 2 已准备好接受触发或输出数据
注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 26:16 保留，必须保持复位值。

位 15 **BWST1**：DAC 通道 1 写入采样时间忙标志 (DAC channel1 busy writing sample time flag)
该位会在采样和保持模式使能后由系统置 1，每次软件写入 DAC_SHSR1 寄存器时，该位都会置 1，对 DAC_SHSR1 的写操作完成后，该位通过硬件清零。（需要约 3 个 LSI/LSE 同步周期）。
0：目前未对 DAC_SHSR1 进行写操作：DAC_SHSR1 可写入
1：目前正在对 DAC_SHSR1 进行写操作：DAC_SHSR1 不可写入

位 14 **CAL_FLAG1**：DAC 通道 1 校准偏移状态 (DAC channel1 calibration offset status)
该位由硬件置 1 和清零
0：校准调整值小于偏移校正值
1：校准调整值等于或大于偏移校正值

位 13 **DMAUDR1**：DAC 通道 1 DMA 下溢标志 (DAC channel1 DMA underrun flag)
该位由硬件置 1，由软件清零（写入 1）。
0：DAC 通道 1 未发生 DMA 下溢错误状况
1：DAC 通道 1 发生 DMA 下溢错误状况（当前所选触发源以高于 DMA 服务能力的频率驱动 DAC 通道 1 转换）

位 12 **DORSTAT1**：DAC 通道 1 输出寄存器状态位 (DAC channel1 output register status bit)
该位由硬件置 1 和清零。该位仅适用于 DAC 在双数据模式下工作时。
0：DOR[11:0] 为实际使用的 DAC 输出
1：DORB[11:0] 为实际使用的 DAC 输出

位 11 **DAC1RDY**：DAC 通道 1 就绪状态位 (DAC channel1 ready status bit)
该位由硬件置 1 和清零。
0：DAC 通道 1 尚未准备好接受触发或输出数据
1：DAC 通道 1 已准备好接受触发或输出数据

位 10:0 保留，必须保持复位值。

21.7.15 DAC 校准控制寄存器 (DAC_CCR)

DAC calibration control register

偏移地址：0x38

复位值：0x00XX 00XX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTRIM2[4:0]				
											rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTRIM1[4:0]				
											rw	rw	rw	rw	rw

位 31:21 保留，必须保持复位值。

位 20:16 **OTRIM2[4:0]**：DAC 通道 2 偏移调整值 (DAC channel2 offset trimming value)
这些位仅适用于双通道 DAC。请参见*第 21.3 节：DAC 实现*。

位 15:5 保留，必须保持复位值。

位 4:0 **OTRIM1[4:0]**：DAC 通道 1 偏移调整值 (DAC channel1 offset trimming value)

21.7.16 DAC 模式控制寄存器 (DAC_MCR)

DAC mode control register

偏移地址：0x3C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	SINFORMAT2	DMADOUBLE2	Res.	Res.	Res.	Res.	Res.	MODE2[2:0]		
						rw	rw						rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HFSEL[1]	HFSEL[0]	Res.	Res.	Res.	Res.	SINFORMAT1	DMADOUBLE1	Res.	Res.	Res.	Res.	Res.	MODE1[2:0]		
rw	rw					rw	rw						rw	rw	rw

位 31:26 保留，必须保持复位值。

位 25 **SINFORMAT2**：为 DAC 通道 2 使能有符号格式 (Enable signed format for DAC channel2)
该位由软件置 1 和清零。
0：输入数据为无符号格式
1：输入数据为有符号格式（二进制补码）。MSB 位表示符号。
注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 24 **DMADOUBLE2**：DAC 通道 2 DMA 双数据模式 (DAC channel2 DMA double data mode)
该位由软件置 1 和清零。
0：选择 DMA 正常模式
1：选择 DMA 双数据模式
注意：该位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 23:19 保留，必须保持复位值。

位 18:16 **MODE2[2:0]**：DAC 通道 2 模式 (DAC channel2 mode)

仅当 DAC 已禁止且不处于校准模式时（DAC_CR 寄存器中的位 EN2 = 0 且位 CEN2 = 0），才可写入这些位。如果 EN2 = 1 或 CEN2 = 1，则会忽略写操作。

这些位由软件置 1 和清零，用于选择 DAC 通道 2 的模式：

– DAC 通道 2 处于正常模式

000：DAC 通道 2 连接到外部引脚且使能了缓冲器

001：DAC 通道 2 连接到外部引脚以及片上外设且使能了缓冲器

010：DAC 通道 2 连接到外部引脚且禁止了缓冲器

011：DAC 通道 2 连接到片上外设且禁止了缓冲器

– DAC 通道 2 处于采样和保持模式

100：DAC 通道 2 连接到外部引脚且使能了缓冲器

101：DAC 通道 2 连接到外部引脚以及片上外设且使能了缓冲器

110：DAC 通道 2 连接到外部引脚以及片上外设且禁止了缓冲器

111：DAC 通道 2 连接到片上外设且禁止了缓冲器

注意：仅可在 EN2 = 0 时修改该寄存器。

关于 DAC 通道 2 的可用性，请参见第 21.3 节：DAC 实现。

位 15:14 **HFSEL[1:0]**：高频接口模式选择 (High frequency interface mode selection)

00：禁止高频接口模式

01：使能高频接口模式，AHB 时钟频率 > 80 MHz

10：使能高频接口模式，AHB 时钟频率 > 160 MHz

11：保留

位 13:10 保留，必须保持复位值。

位 9 **SINFORMAT1**：为 DAC 通道 1 使能有符号格式 (Enable signed format for DAC channel1)

该位由软件置 1 和清零。

0：输入数据为无符号格式

1：输入数据为有符号格式（二进制补码）。MSB 位表示符号。

位 8 **DMADOUBLE1**：DAC 通道 1 DMA 双数据模式 (DAC channel1 DMA double data mode)

该位由软件置 1 和清零。

0：选择 DMA 正常模式

1：选择 DMA 双数据模式

位 7:3 保留，必须保持复位值。

位 2:0 **MODE1[2:0]**：DAC 通道 1 模式 (DAC channel1 mode)

仅当 DAC 已禁止且不处于校准模式时（DAC_CR 寄存器中的位 EN1 = 0 且位 CEN1 = 0），才可写入这些位。如果 EN1 = 1 或 CEN1 = 1，则会忽略写操作。

这些位由软件置 1 和清零，用于选择 DAC 通道 1 的模式：

– DAC 通道 1 处于正常模式

000：DAC 通道 1 连接到外部引脚且使能了缓冲器

001：DAC 通道 1 连接到外部引脚以及片上外设且使能了缓冲器

010：DAC 通道 1 连接到外部引脚且禁止了缓冲器

011：DAC 通道 1 连接到片上外设且禁止了缓冲器

– DAC 通道 1 处于采样和保持模式

100：DAC 通道 1 连接到外部引脚且使能了缓冲器

101：DAC 通道 1 连接到外部引脚以及片上外设且使能了缓冲器

110：DAC 通道 1 连接到外部引脚以及片上外设且禁止了缓冲器

111：DAC 通道 1 连接到片上外设且禁止了缓冲器

注意：仅可在 EN1 = 0 时修改该寄存器。

21.7.17 DAC 通道 1 采样和保持采样时间寄存器 (DAC_SHSR1)

DAC channel1 sample and hold sample time register

偏移地址：0x40

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	TSAMPLE1[9:0]									
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:10 保留，必须保持复位值。

位 9:0 **TSAMPLE1[9:0]**：DAC 通道 1 采样时间 (DAC channel1 sample time)（仅在采样和保持模式下有效）

当 DAC 通道 1 已禁止或者处于正常运行期间时，可写入这些位。对于后一种情况，仅当 DAC_SR 寄存器的 BWST1 为低电平时，才可进行写操作，如果 BWST1 = 1，会忽略写操作。

注意： *它代表执行采样阶段的 LSI/LSE 时钟数。采样时间* = (TSAMPLE1[9:0] + 1) x *LSI/LSE 时钟周期。*

21.7.18 DAC 通道 2 采样和保持采样时间寄存器 (DAC_SHSR2)

DAC channel2 sample and hold sample time register

此寄存器仅适用于双通道 DAC。请参见*第 21.3 节：DAC 实现*。

偏移地址：0x44

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	TSAMPLE2[9:0]									
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:10 保留，必须保持复位值。

位 9:0 **TSAMPLE2[9:0]**：DAC 通道 2 采样时间 (DAC channel2 sample time)（仅在采样和保持模式下有效）

当 DAC 通道 2 已禁止或者处于正常运行期间时，可写入这些位。对于后一种情况，仅当 DAC_SR 寄存器的 BWST2 为低电平时，才可进行写操作，如果 BWST2 = 1，会忽略写操作。

注意： *它代表执行采样阶段的 LSI/LSE 时钟数。采样时间* = (TSAMPLE1[9:0] + 1) x *LSI/LSE 时钟周期。*

21.7.19 DAC 采样和保持时间寄存器 (DAC_SHHR)

DAC sample and hold time register

偏移地址：0x48

复位值：0x0001 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	THOLD2[9:0]									
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	THOLD1[9:0]									
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:26 保留，必须保持复位值。

位 25:16 **THOLD2[9:0]**：DAC 通道 2 保持时间 (DAC channel2 hold time)（仅在采样和保持模式下有效）
保持时间 = (THOLD[9:0]) x LSI/LSE 时钟周期
注意：仅可在 EN2 = 0 时修改该寄存器。
这些位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 15:10 保留，必须保持复位值。

位 9:0 **THOLD1[9:0]**：DAC 通道 1 保持时间 (DAC channel1 hold time)（仅在采样和保持模式下有效）
保持时间 = (THOLD[9:0]) x LSI/LSE 时钟周期
注意：仅可在 EN1 = 0 时修改该寄存器。

注意：仅当 DAC 通道已禁止且处于正常运行模式时（DAC_CR 寄存器中的位 ENx = 0 且位 CENx = 0），才可写入这些位。如果 ENx = 1 或 CENx = 1，则会忽略写操作。

21.7.20 DAC 采样和保持刷新时间寄存器 (DAC_SHRR)

DAC sample and hold refresh time register

偏移地址：0x4C

复位值：0x0001 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TREFRESH2[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TREFRESH1[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23:16 **TREFRESH2[7:0]**：DAC 通道 2 刷新时间 (DAC channel2 refresh time)（仅在采样和保持模式下有效）

刷新时间 = (TREFRESH[7:0]) x LSI/LSE 时钟周期

注意：仅可在 EN2 = 0 时修改该寄存器。

这些位仅适用于双通道 DAC。请参见第 21.3 节：DAC 实现。

位 15:8 保留，必须保持复位值。

位 7:0 **TREFRESH1[7:0]**：DAC 通道 1 刷新时间 (DAC channel1 refresh time)（仅在采样和保持模式下有效）

刷新时间 = (TREFRESH[7:0]) x LSI/LSE 时钟周期

注意：仅可在 EN1 = 0 时修改该寄存器。

注意：仅当 DAC 通道已禁止且处于正常运行模式时（DAC_CR 寄存器中的位 ENx = 0 且位 CENx = 0），才可写入这些位。如果 ENx = 1 或 CENx = 1，则会忽略写操作。

21.7.21 DAC 寄存器映射

表 138 汇总了 DAC 寄存器。

表 138. DAC 寄存器映射和复位值

偏移	寄存器名称 复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**DAC_CR**	Res.	CEN2	DMAUDRIE2	DMAEN2	MAMP2[3:0]				WAVE2[2:0]		TSEL2[3:1]			TSEL2[0]	TEN2	EN2	Res.	CEN1	DMAUDRIE1	DMAEN1	MAMP1[3:0]				WAVE1[1:0]		TSEL1[3:1]			TSEL1[0]	TEN1	EN1
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	**DAC_ SWTRGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWTRIG2	SWTRIG1
	Reset value																															0	0
0x08	**DAC_ DHR12R1**	Res.	Res.	Res.	Res.	DACC1DHRB[11:0]												Res.	Res.	Res.	Res.	DACC1DHR[11:0]											
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0
0x0C	**DAC_ DHR12L1**	DACC1DHRB[11:0]												Res.	Res.	Res.	Res.	DACC1DHR[11:0]												Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0				
0x10	**DAC_ DHR8R1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC1DHRB[7:0]								DACC1DHR[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	**DAC_ DHR12R2**	Res.	Res.	Res.	Res.	DACC2DHRB[11:0]												Res.	Res.	Res.	Res.	DACC2DHR[11:0]											
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0
0x18	**DAC_ DHR12L2**	DACC2DHRB[11:0]												Res.	Res.	Res.	Res.	DACC2DHR[11:0]												Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0				
0x1C	**DAC_ DHR8R2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DHRB[7:0]								DACC2DHR[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	**DAC_ DHR12RD**	Res.	Res.	Res.	Res.	DACC2DHR[11:0]												Res.	Res.	Res.	Res.	DACC1DHR[11:0]											
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0
0x24	**DAC_ DHR12LD**	DACC2DHR[11:0]												Res.	Res.	Res.	Res.	DACC1DHR[11:0]												Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0				
0x28	**DAC_ DHR8RD**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DHR[7:0]								DACC1DHR[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	**DAC_ DOR1**	Res.	Res.	Res.	Res.	DACC1DORB[11:0]												Res.	Res.	Res.	Res.	DACC1DOR[11:0]											
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0
0x30	**DAC_ DOR2**	Res.	Res.	Res.	Res.	DACC2DORB[11:0]												Res.	Res.	Res.	Res.	DACC2DOR[11:0]											
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0

表 138. DAC 寄存器映射和复位值（续）

偏移	寄存器名称 复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x34	**DAC_SR**	BWST2	CAL_FLAG2	DMAUDR2	DORSTAT2	DAC2RDY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BWST1	CAL_FLAG1	DMAUDR1	DORSTAT1	DAC1RDY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0												0	0	0	0	0											
0x38	**DAC_CCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTRIM2[4]	OTRIM2[3]	OTRIM2[2]	OTRIM2[1]	OTRIM2[0]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTRIM1[4]	OTRIM1[3]	OTRIM1[2]	OTRIM1[1]	OTRIM1[0]
	Reset value												X	X	X	X	X												X	X	X	X	X
0x3C	**DAC_MCR**	Res.	Res.	Res.	Res.	Res.	Res.	SINFORMAT2	DMADOUBLE2	Res.	Res.	Res.	Res.	Res.	MODE2 [2:0]			HFSEL[1]	HFSEL[0]	Res.	Res.	Res.	Res.	SINFORMAT1	DMADOUBLE1	Res.	Res.	Res.	Res.	Res.	MODE1 [2:0]		
	Reset value							0	0						0	0	0	0	0					0	0						0	0	0
0x40	**DAC_ SHSR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSAMPLE1[9:0]									
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x44	**DAC_ SHSR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSAMPLE2[9:0]									
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x48	**DAC_ SHHR**	Res.	Res.	Res.	Res.	Res.	Res.	THOLD2[9:0]										Res.	Res.	Res.	Res.	Res.	Res.	THOLD1[9:0]									
	Reset value							0	0	0	0	0	0	0	0	0	1							0	0	0	0	0	0	0	0	0	1
0x4C	**DAC_ SHRR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TREFRESH2[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TREFRESH1[7:0]							
	Reset value									0	0	0	0	0	0	0	1									0	0	0	0	0	0	0	1
0x50- 0x54	Reserved	Res.																															

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

22 比较器 (COMP)

22.1 简介

器件内置一个超低功耗比较器。该比较器可用于多种功能，包括：

- 在模拟信号的触发下从低功耗模式唤醒
- 模拟信号调理
- 与定时器的 PWM 输出结合使用时，构成逐周期电流控制环路

22.2 COMP 主要特性

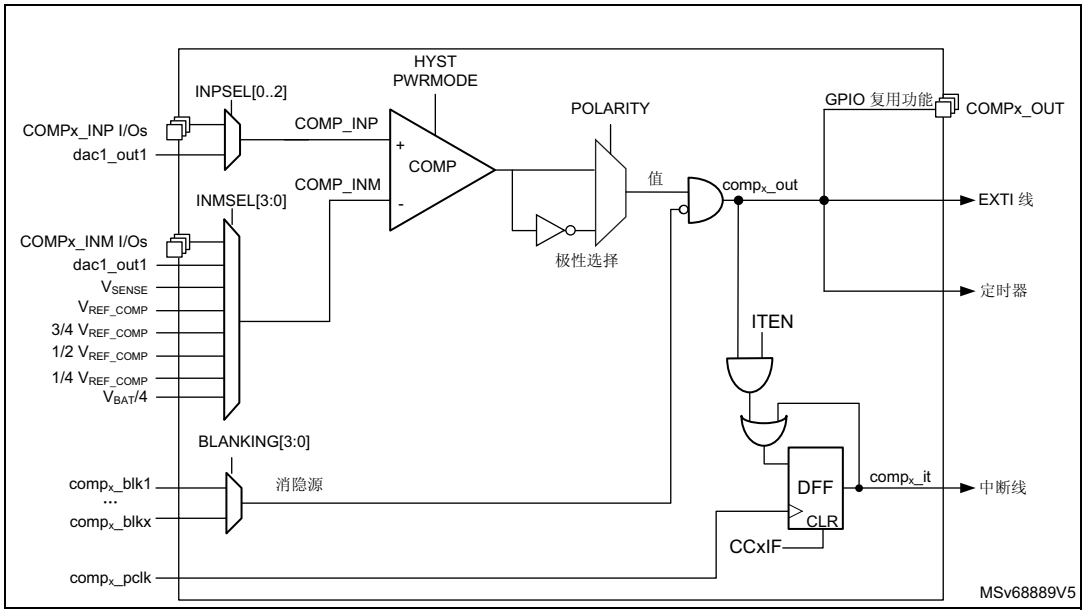
- 可选反相模拟输入：
  - I/O 引脚
  - DAC 通道 1 输出
  - 通过调节器（缓冲分压器）提供的内部参考电压和三个因数分压值（1/4、1/2、3/4）
- 三个 I/O 引脚可选作同相模拟输入
- 可编程迟滞
- 可编程速度/功耗
- 将输出映射到 I/O
- 将输出重定向到用于触发以下事件的定时器输入：
  - 捕获事件
  - OCREF_CLR 事件（用于逐周期电流控制）
  - 刹车事件（用于快速 PWM 关断）
- 消隐比较器输出
- 可生成中断，用于使器件从睡眠模式和停止模式唤醒（通过 EXTI 控制器）
- 将中断输出定向到 CPU

22.3 COMP 功能描述

22.3.1 COMP 框图

比较器的框图如下图所示。

图 143. 比较器功能框图



22.3.2 COMP 引脚和内部信号

用作比较器输入的 I/O 必须在 GPIO 寄存器中配置为模拟模式。

比较器输出可通过其复用功能连接到 I/O。更多详细信息，请参见产品数据手册。

输出也可以在内部重定向到用于以下用途的各种定时器输入：

- 使用 BKIN 和 BKIN2 输入紧急关断 PWM 信号
- 使用定时器的 ETR 输入进行逐周期电流控制
- 用于时序测量的输入捕获

比较器输出可以在内部同时连接到 I/O 引脚。

表 139. COMP1 输入/输出内部信号

信号名称	信号类型	说明
dac1_out1	模拟输入	来自 DAC 外设的模拟信号
comp1_blkx	数字输出	请参见*表 140*
comp1_out	数字输出	COMP1 输出
comp1_it	数字输出	COMP1 中断输出

表 140. COMP1 消隐源选择

信号名称	信号类型	消隐输入源
comp1_blk1	数字输入	tim1_oc5
comp1_blk2	数字输入	tim2_oc3
comp1_blk3	数字输入	tim3_oc3
comp1_blk4	数字输入	tim3_oc4
comp1_blk5	数字输入	lptim1_ch2
comp1_blk6	数字输入	lptim2_ch2

表 141. COMP1 同相输入分配

COMP1_INP	COMP1_INPSEL2	COMP1_INPSEL1	COMP1_INPSEL0
COMP1_INP1	X	X	1
COMP1_INP2	0	0	0
COMP1_INP3	0	1	0
dac1_out1	1	X	0

注意： *其他组合保留。*

表 142. COMP1 反相输入分配

COMP1_INM	COMP1_INMSEL[3:0]
¼ V_{REFINT}	0000
½ V_{REFINT}	0001
¾ V_{REFINT}	0010
V_{REFINT}	0011
dac1_out1	0100
COMP1_INM1	0101
COMP1_INM2	0110
COMP1_INM3	0111
V_{SENSE}	1000
V_{BAT}/4	1001

注意： *其他组合保留。*

表 143. COMP1 消隐源分配

消隐源	COMP1_BLANKING[3:0]
无消隐	0000
tim1_oc5	0001
tim2_oc3	0010

表 143. COMP1 消隐源分配（续）

消隐源	COMP1_BLANKING[3:0]
tim3_oc3	0011
tim3_oc4	0100
lptim1_ch2	0101
lptim2_ch2	0110

22.3.3 COMP 复位和时钟

时钟控制器提供的时钟 comp_pclk 与 APB 时钟同步。

注意： ***重要提示：****极性选择逻辑和到端口的输出重定向独立于 APB 时钟。因此，即使在停止模式下比较器仍能正常工作。连接至 CPU 的 NVIC 的中断线需要 APB 时钟 (comp_pclk) 才能工作。如果没有 APB 时钟，则无法生成中断信号 comp_it。*

22.3.4 比较器锁定机制

比较器可用于过流或热保护等安全用途。对于具有特定功能安全要求的应用，可对比较器配置进行保护，以防发生意外修改（例如，当程序计数器损坏时）。

为此，可以对比较器配置寄存器进行写保护（只读）。

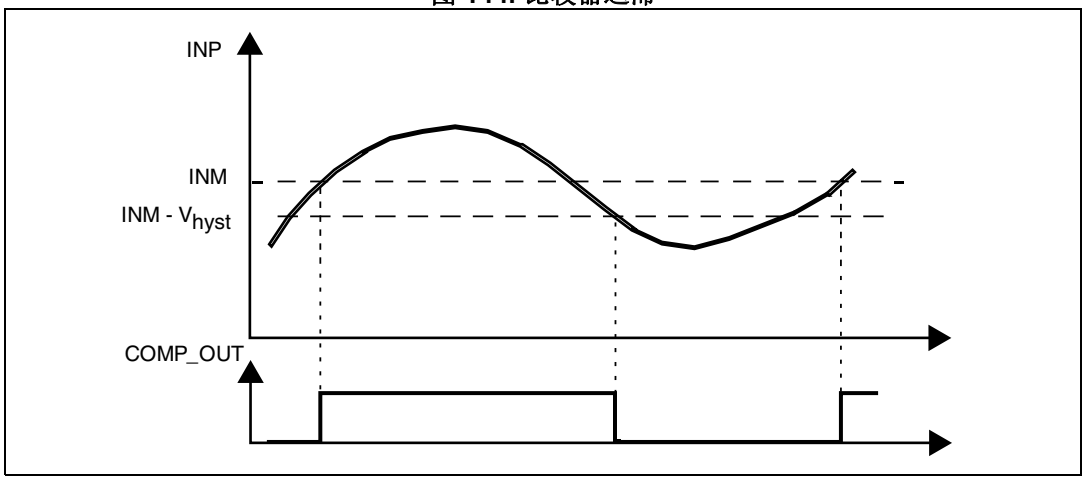
一旦比较器通道配置完成，其 LOCK 位设置为 1。这会导致比较器通道的整个寄存器组变为只读，包括 LOCK 位。

只能通过 MCU 复位来移除写保护。

22.3.5 迟滞

比较器具有可编程迟滞，可在有噪声信号时避免发生意外输出转换。迟滞可在不需要时（例如，退出低功耗模式时）禁止。这样就可以使用外部组件强制迟滞值。

图 144. 比较器迟滞

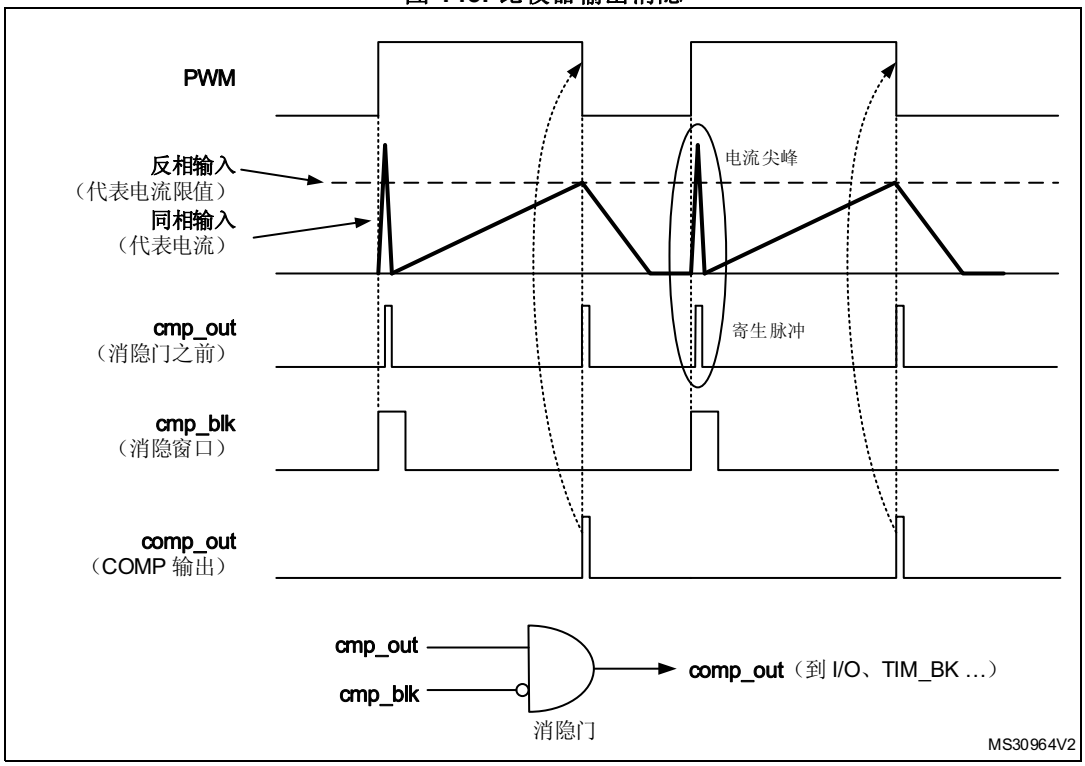


22.3.6 比较器输出消隐功能

消隐功能的目的是防止电流调节在 PWM 周期开始处出现短暂电流尖峰（通常为功率开关反向并联二极管中的恢复电流）时发生跳闸。该功能使用通过定时器输出比较信号定义的消隐窗口。有关可选消隐信号，请参见寄存器说明。

消隐信号对内部比较器输出进行门控，以便能使 comp_out 免受因电流尖峰而导致的寄生脉冲的干扰，如下图所示（图中未显示 COMP 通道编号）。

图 145. 比较器输出消隐



22.3.7 COMP 启动稳定时间

在 COMP 使能（上电）后的一段时间内（定义为启动稳定时间），比较器参数及其输出均未定义。更多详细信息，请参见产品数据手册。如果比较器输出正在驱动设计中的任何关键阶段，建议在稳定时间过后通过 GPIO 复用功能或通过 IP 配置（如定时器）将比较器输出重定向到器件 I/O。

22.3.8 COMP 功耗和速度模式

对于给定的应用，可调节 COMP 通道功耗与传播延迟以获得最佳平衡。

COMP_CFGRx 寄存器的位 PWRMODE[1:0] 可编程为如下值：

- 00：高速/高功耗
- 01：中速/中等功耗
- 10：中速/中等功耗
- 11：极低速/超低功耗

22.4 COMP 低功耗模式

表 144. 低功耗模式下的比较器行为

模式	说明
睡眠	对比较器无影响。比较器中断可使器件退出睡眠模式。
停止	对比较器无影响。比较器中断可使器件退出停止模式。

注意： *当内部参考电压关闭时，不能使用比较器将器件从睡眠模式或停止模式退出。*

22.5 COMP 中断

可通过两种方式将比较器用作中断源。

比较器输出从内部连接到扩展中断和事件控制器 (EXTI)。每个比较器都具有其各自的 EXTI 线，能够产生中断或事件，可使器件退出低功耗模式。

比较器还提供有连接至 CPU 的 NVIC 的中断线。在 CPU 激活时使用该功能处理低延迟中断。这需要 APB 时钟处于运行状态。

22.5.1 通过 EXTI 实现的中断

更多详细信息，请参见中断和事件部分。

通过 EXTI 模块实现 COMP 中断的程序：

1. 使能 COMP 并等待稳定。
2. 将 EXTI 线（用于接收 comp_wkup 信号）配置为中断模式，选择上升沿、下降沿或任一边沿有效，然后使能 EXTI 线。
3. 配置并使能映射到相应 EXTI 线的 NVIC IRQ 通道。

表 145. 通过 EXTI 控制位实现的中断

中断事件	事件标志	使能控制位	退出睡眠模式	退出停止模式	退出待机模式
comp1_wkup	通过 EXTI	通过 EXTI	是	是	N/A

22.5.2 通过 CPU 的 NVIC 实现的中断

通过 CPU 的 NVIC 实现 COMP 中断的程序：

1. 使能 COMP 并等待稳定。
2. 配置并使能映射到 comp_it 线的 NVIC IRQ 通道。
3. 在 COMP_CFGRx 中配置并使能 ITEN。

表 146. 通过 NVIC 控制位实现的中断

中断事件	中断标志	使能控制位	中断清除位	退出睡眠模式	退出停止模式
comp1_it	COMP_CFGR1 中的 C1IF	COMP_CFGR1 中的 ITEN	CC1IF	是（使用 APB 时钟）	否

注意： *要使用该中断，必须使能 APB 时钟。如果未使能该时钟，则不会生成中断。*
该中断为高电平有效，并且无法通过硬件自动清除。必须在比较器中断处理程序中通过固件（将 COMP_CFGR1 寄存器中的 ITEN 设为 0）来清除。

中断信号 comp_it 在比较器输出高电平时生成，因此当中断发生后，必须尽快将 comp_it 信号设置为低电平，以避免比较器中断处理程序重入。这可通过将 ITEN 设为 0 或更改比较器 POLARITY 位来完成。

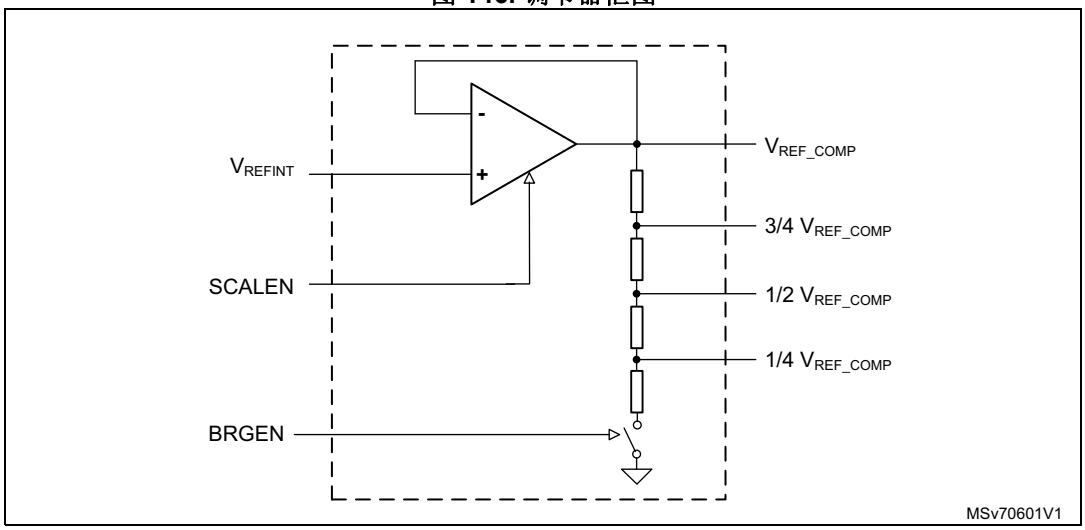
22.6 SCALER 功能

调节器模块用于为各个比较器输入提供不同的参考电压。其基于一个驱动电阻桥的放大器。放大器输入连接至内部参考电压。

放大器和电阻桥可单独使能。放大器由 COMP_CFGRx 寄存器的 SCALEN 位使能。电阻桥由 COMP_CFGRx 寄存器的 BRGEN 位使能。

如果未使用电阻分压，则可断开电阻桥，以降低功耗。断开后，1/4 V_{REF_COMP}、1/2 V_{REF_COMP} 和 3/4 V_{REF_COMP} 值等于 V_{REF_COMP}。

图 146. 调节器框图



22.7 COMP 寄存器

22.7.1 比较器状态寄存器 (COMP_SR)

Comparator status register

COMP_SR 为比较器状态寄存器。

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	C1IF
															r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	C1VAL
															r

位 31:17 保留，必须保持复位值。

位 16 **C1IF**：COMP 通道 1 中断标志 (COMP Channel1 interrupt flag)
当 COMP 通道 1 输出置 1 时，该位由硬件置 1。
该位由软件通过向 COMP_ICFR 寄存器的 CC1IF 位写入 1 来清零。

位 15:1 保留，必须保持复位值。

位 0 **C1VAL**：COMP 通道 1 输出状态位 (COMP Channel1 output status bit)
该位为只读，用于反映当前 COMP 通道 1 输出的状态（考虑了 POLARITY 和 BLANKING 位的影响）。

22.7.2 比较器中断清零标志寄存器 (COMP_ICFR)

Comparator interrupt clear flag register

COMP_ICFR 为比较器中断清零标志寄存器。

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IF
															rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

位 31:17 保留，必须保持复位值。

位 16 **CC1IF**：清零 COMP 通道 1 中断标志 (Clear COMP Channel1 interrupt flag)
写入 1 可将 COMP_SR 寄存器中的 C1IF 标志清零。

位 15:0 保留，必须保持复位值。

22.7.3 比较器配置寄存器 1 (COMP_CFGR1)

Comparator configuration register 1

COMP_CFGR1 为 COMP 配置寄存器。

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
LOCK	Res.	Res.	Res.	BLANKING[3:0]				Res.	INPSEL2	Res.	INPSEL1	INMSEL[3:0]			
rs				rw	rw	rw	rw		rw		rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PWRMODE[1:0]		Res.	Res.	HYST[1:0]		Res.	ITEN	Res.	Res.	POLARITY	SCALEN	BRGEN	EN
		rw	rw			rw	rw		rw			rw	rw	rw	rw

位 31 **LOCK**：锁定 (Lock)

该位由软件置 1，通过硬件系统复位清零，用于锁定 COMP 通道 1 配置寄存器 (COMP_CFGR1[31:0]) 的全部内容。

0：可以读/写 COMP_CFGR1[31:0] 寄存器

1：只能读取 COMP_CFGR1[31:0]

位 30:28 保留，必须保持复位值。

位 27:24 **BLANKING[3:0]**：COMP 通道 1 消隐源选择 (COMP Channel1 blanking source selection)

该位域的各个位由软件置 1 和清零（仅限 LOCK 未置 1 的情况）。

该位域用于选择 COMP 通道 1 输出消隐输入源：

0000：无消隐

0001：comp_blk1

0010：comp_blk2

0011：comp_blk3

0100：comp_blk4

0101：comp_blk5

0110：comp_blk6

所有其他值：保留

位 23 保留，必须保持复位值。

位 22 **INPSEL2**：COMP 同相输入选择 (COMP noninverting input selection)

该位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于选择连接到 COMP 通道正输入的输入。

更多详细信息，请参见*表 141：COMP1 同相输入分配*。

位 21 保留，必须保持复位值。

位 20 **INPSEL1**：COMP 同相输入选择 (COMP noninverting input selection)

该位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于选择连接到 COMP 通道正输入的输入。

注意：更多详细信息，请参见表 141：COMP1 同相输入分配。

位 19:16 **INMSEL[3:0]**：COMP 通道 1 反相输入选择 (COMP channel1 inverting input selection)

这些位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于选择连接到 COMP 通道负输入的输入。

注意：更多详细信息，请参见表 142：COMP1 反相输入分配。

位 15:14 保留，必须保持复位值。

位 13:12 **PWRMODE[1:0]**：COMP 通道 1 的功耗模式 (Power mode of the COMP channel1)
这些位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于控制 COMP 通道 1 的功耗/速度。
00：高速/高功耗
01：中速/中等功耗
10：中速/中等功耗
11：极低速/超低功耗

位 11:10 保留，必须保持复位值。

位 9:8 **HYST[1:0]**：COMP 通道 1 迟滞选择 (COMP channel1 hysteresis selection)
这些位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于选择 COMP 通道 1 的迟滞电压。
00：无迟滞
01：低迟滞
10：中等迟滞
11：高迟滞

位 7 保留，必须保持复位值。

位 6 **ITEN**：COMP 通道 1 中断使能 (COMP channel1 interrupt enable)
该位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于使能 COMP 通道 1 的中断生成。
0：禁止 COMP 通道 1 生成中断
1：使能 COMP 通道 1 生成中断

位 5:4 保留，必须保持复位值。

位 3 **POLARITY**：COMP 通道 1 极性选择 (COMP channel1 polarity selection)
该位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于反转 COMP 通道 1 极性。
0：不反转 COMP 通道 1 输出
1：反转 COMP 通道 1 输出

位 2 **SCALEN**：电压调节器使能 (Voltage scaler enable)
该位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于使能 COMP 通道的 V_{REFINT} 调节器。
0：禁止 V_{REFINT} 调节器
1：使能 V_{REFINT} 调节器

位 1 **BRGEN**：调节器电阻桥使能 (Scaler bridge enable)
该位由软件置 1 和清零（仅限 LOCK 未置 1 的情况）。该位可使能调节器电阻桥。
0：禁止调节器电阻桥
1：使能调节器电阻桥
如果 SCALEN 置 1 且 BRGEN 复位，则全部四个调节器输出都会提供相同值 V_{REF_COMP}（近似于 V_{REFINT}）。
如果 SCALEN 和 BRGEN 均置 1，则四个调节器输出会分别提供 V_{REF_COMP}、3/4 V_{REF_COMP}、1/2 V_{REF_COMP} 和 1/4 V_{REF_COMP} 值。

位 0 **EN**：COMP 1 通道使能 (COMP Channel1 enable)
该位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于使能 COMP 通道 1。
0：禁止
1：使能

22.7.4 比较器配置寄存器 2 (COMP_CFGR2)

Comparator configuration register 2

COMP_CFGR2 为 COMP 配置寄存器。

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
rs															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INPSEL0	Res.	Res.	Res.	Res.
											rw				

位 31 **LOCK**：锁定 (Lock)

该位由软件置 1，通过硬件系统复位清零，用于锁定
COMP 通道 2 配置寄存器 (COMP_CFGR2[31:0]) 的全部内容。
0：可以读/写 COMP_CFGR2[31:0] 寄存器
1：只能读取 COMP_CFGR2[31:0]

位 30:5 保留，必须保持复位值。

位 4 **INPSEL0**：COMP 同相输入选择 (COMP non-inverting input selection)

该位由软件置 1 和清零（仅限 LOCK 未置 1 的情况），用于选择连接到 COMP 通道正输入的输入。
更多详细信息，请参见*表 141：COMP1 同相输入分配*。

位 3:0 保留，必须保持复位值。

22.7.5 COMP 寄存器映射

表 147. COMP 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**COMP_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	C1IF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	C1VAL
	Reset value																0																0
0x04	**COMP_ICFR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																0																
0x0C	**COMP_ CFGR1**	LOCK	Res.	Res.	Res.	BLANKING [3:0]				Res.	.INPSEL2	Res.	INPSEL1	INMSEL [3:0]				Res.	Res.	PWRMODE [1:0]		Res.	Res.	HYST[1:0]		Res.	ITEN	Res.	Res.	POLARITY	SCALEN	BRGEN	EN
	Reset value	0				0	0	0	0		0		0	0	0	0	0			0	0			0	0		0			0	0	0	0
0x10	**COMP_ CFGR2**	LOCK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INPSEL0	Res.	Res.	Res.	Res.
	Reset value	0																											0				

有关寄存器边界地址的信息，请参见*第 2.2 节*。

23 运算放大器 (OPAMP)

23.1 简介

器件内置一个运算放大器，该运算放大器具有两个输入和一个输出。这三个 I/O 可连接到外部引脚，因此能实现任意类型的外部互连。可在内部对运算放大器进行以下三种配置：

- 配置为跟随器
- 配置为放大器（具有从 2 到 16 的同相增益）
- 配置为放大器（具有从 -1 到 -15 的反相增益）

有关 OPAMP 输入和输出与内部外设连接的详细信息，请参见*第 23.3.3 节：信号走线*。

23.2 OPAMP 主要特性

- 轨到轨输入电压范围
- 低输入失调电压（校准后为 1.5 mV，出厂校准时为 10 mV）
- 7 MHz 增益带宽
- 高速模式提供更高的压摆率

注意： *有关详细的 OPAMP 特性，请参见产品数据手册。*

23.3 OPAMP 功能描述

OPAMP 有多种模式。

每个 OPAMP 均可单独使能，如果被禁止，则输出呈高阻态。

使能后，其既可以处于校准模式，此模式下 OPAMP 的所有输入都断开，也可以处于功能模式。

有两种功能模式，高速模式和正常模式。在功能模式下，OPAMP 的输入和输出的连接如*第 23.3.3 节：信号走线*中所述。

23.3.1 OPAMP 复位和时钟

访问寄存器时，需要运算放大器时钟。如果应用程序无需对这些寄存器进行读或写访问，则可使用外设时钟使能寄存器关闭此时钟。请参见*第 10.8.24 节：RCC APB1 外设时钟寄存器 (RCC_APB1LENR)* 中的 OPAMPEN 位。

OPAEN 位使能和禁止 OPAMP 工作。应先更改 OPAMP 寄存器配置，然后再使能 OPAEN 位，以避免对输出产生寄生效应。

如果不再需要运算放大器的输出，则可禁止运算放大器以降低功耗。禁止 OPAMP 时，会保持之前设置的所有配置（包括校准）不变。

23.3.2 初始配置

运算放大器的默认配置为功能模式，在该模式下，三个输入/输出连接到外部引脚。在默认模式下，运算放大器使用出厂微调值进行偏移校准。有关出厂微调的条件，请参见数据手册的电气特性部分，通常温度为 30°C，电压为 3 V。可对微调值进行调节。有关更改微调值的信息，请参见*第 23.3.5 节：校准*。默认配置使用正常模式，该模式提供标准性能。可以将 OPAHSM 位置 1，从而将运算放大器切换至高速模式，以达到更高的压摆率。数据手册的*电气特性*部分对正常模式和高速模式特性进行了定义。

一旦 OPAMPx_CSR 寄存器中的 OPAEN 位置 1，运算放大器即可工作。两个输入引脚和一个输出引脚可按照*第 23.3.3 节：信号走线*中的定义进行连接，并且可对默认连接设置进行更改。

注意： *必须在相应 GPIOx_MODER 寄存器中将输入和输出引脚配置为模拟模式（默认状态）。*

23.3.3 信号走线

运算放大器引脚走线由 OPAMPx_CSR 寄存器确定。

下表介绍了运算放大器 (OPAMP1) 的引脚连接。

表 148. OPAMP 的可能连接

信号	引脚	内部	注释
OPAMP1_VINM	PC5(INM0) PB1(INM1)	ADC1_INP8 ADC1_INM4 ADC1_INP5 COMP1_INM6 OPAMP1_VOUT 或 PGA	PGA_GAIN 和 VM_SEL 位控制该引脚。 ADC 控制 ADC 输入。
OPAMP1_VINP	PB0(INP0) PA0(INP2)	dac1_out1 (INP1) ADC1_INM5 ADC1_INP9 ADC1_INP0 ADC1_INM1 COMP1_INP1	VP_SEL 位控制该引脚。 ADC 控制 ADC 输入。
OPAMP1_VOUT	PA7	ADC1_INM3 ADC1_INP7	在 OPAMP 使能时连接该引脚。 ADC 输入通过 ADC 控制。

23.3.4 OPAMP 模式

运算放大器的输入和输出都可在引脚上访问。放大器可用于多种配置环境：

- 独立模式（外部增益设置模式）
- 跟随器配置模式
- PGA 模式

注意： *放大器输出引脚直接连接到输出引脚的焊盘，以最大限度地降低输出阻抗。如果使能了放大器，那么即使放大器被配置为 PGA 并仅连接到内部通道，它的输出引脚也不能用作通用 I/O。*

输入信号的阻抗必须保持在不能使输入泄漏产生明显伪信号（由于源极阻降而引起）的水平以下。有关更多详细信息，请参见数据手册的电气特性部分。

独立模式（外部增益设置模式）

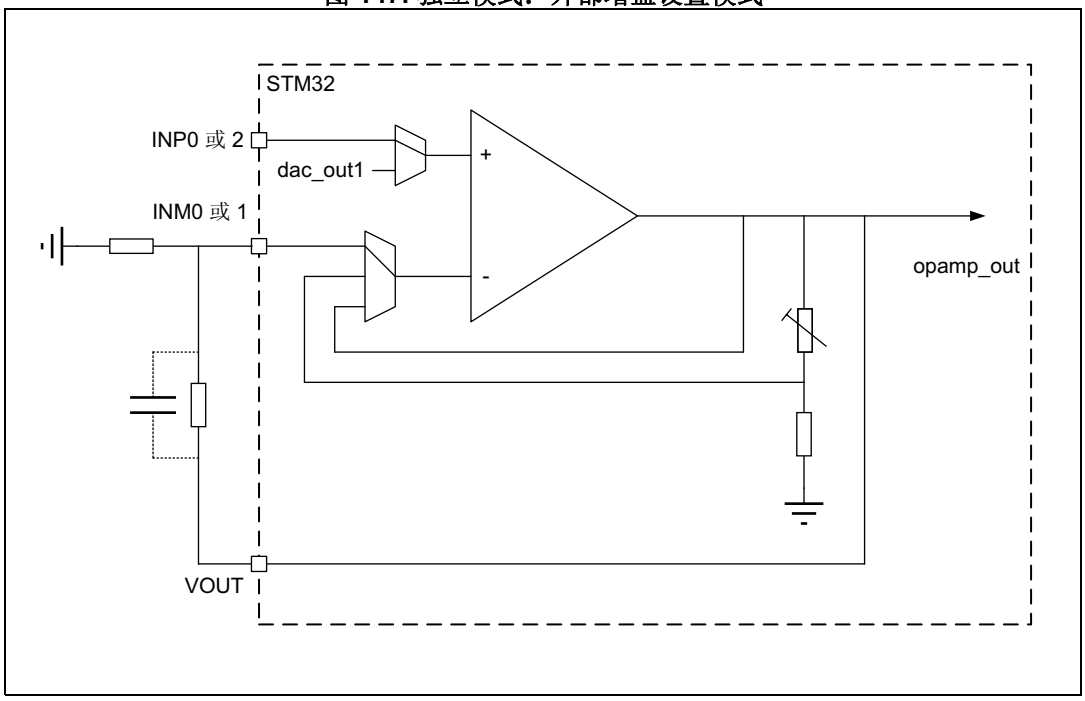
下面介绍在独立模式下使用 OPAMP 的步骤。

假设 OPAMPx_CSR 配置为默认值，GPIOx_MODER 配置为默认状态。只要 OPAEN 位置 1，两个输入引脚和一个输出引脚都连接到运算放大器。

此默认配置使用工厂微调值，并在正常模式下运行（达到最高性能）。可按如下方式更改 OPAMP 的行为：

- 可以将 OPAHSM 设置为“运算放大器高速”模式，以达到高压摆率。
- 可以将 USERTRIM 置 1，以修改输入偏移的微调值。

图 147. 独立模式：外部增益设置模式



跟随器配置模式

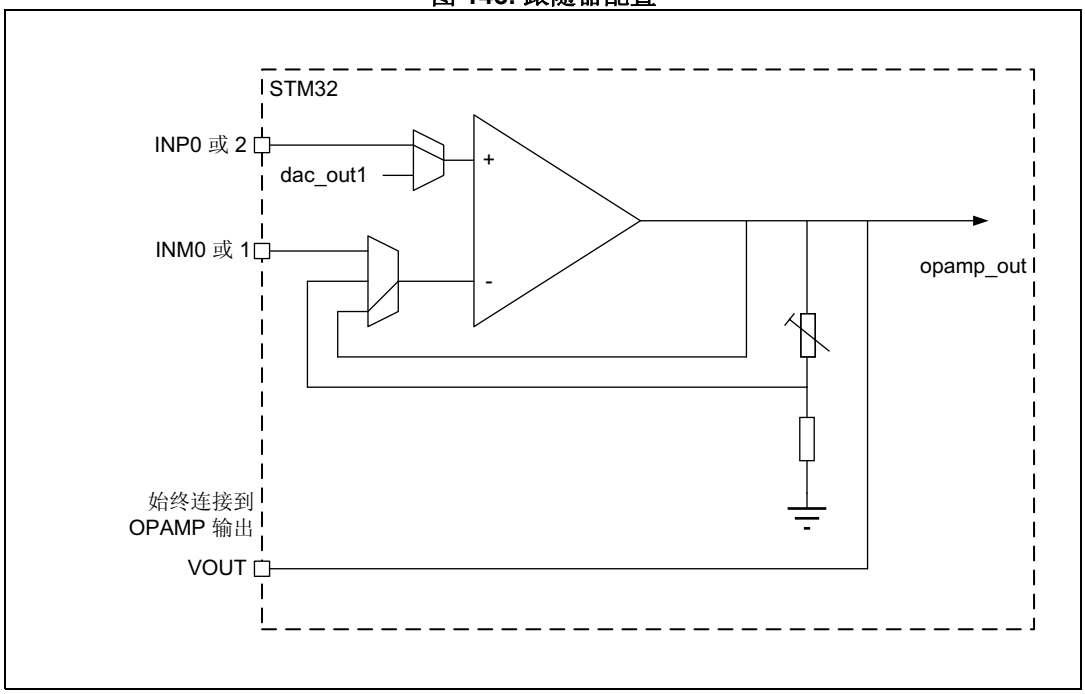
下面介绍在跟随器模式下使用 OPAMP 的步骤。

- 将 VM_SEL 位配置为“opamp_out 连接到 OPAMPx_VINM 输入”(11)。
- 将 VP_SEL 位配置为“GPIO 连接到 OPAMPx_VINP”（00 或 10）。
- 一旦 OPAEN 位置 1，同相输入的电压就会被缓冲到 OPAMP1_VOUT 引脚。

注意： *对应于 OPAMPx_VINM 的引脚可供另外使用。*

OPAMP1 输出上的信号也视为 ADC 输入。因此，如果输入信号频率与运算放大器增益带宽积规范相符，则配置为跟随器模式的 OPAMP 可用于对输入信号进行阻抗调整，然后再将这些信号馈送至 ADC 输入。

图 148. 跟随器配置



可编程增益放大器模式

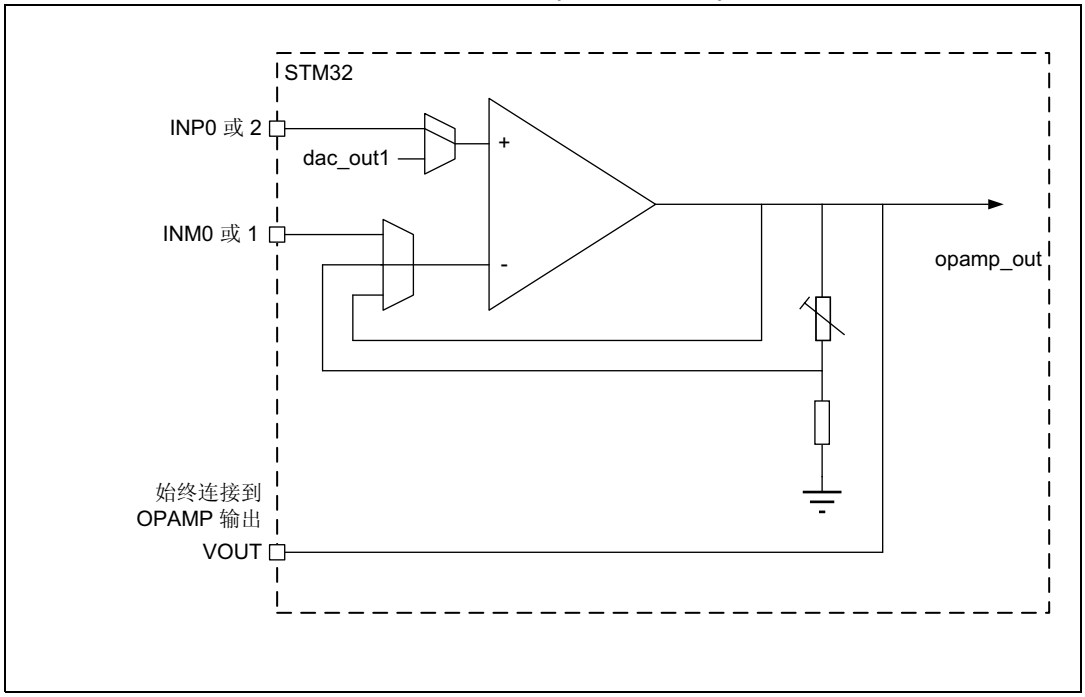
下面介绍将 OPAMP 用作可编程增益放大器的步骤。

- 将 VM_SEL 位配置为“反馈电阻连接到 OPAMPx_VINM 输入”(10)。
- 将 PGA_GAIN 位配置为“内部增益 2、4、8 或 16”（0000 到 0011）。
- 将 VP_SEL 位配置为“GPIO 连接到 OPAMPx_VINP”（00 或 10）。

  一旦 OPAEN 位置 1，所选增益就会放大同相输入电压，并可在引脚 OPAMPx_VOUT 上测得放大后的电压。

注意： *为避免饱和，输入电压应保持低于：V_{DDA} 除以所选增益。*

图 149. PGA 模式，内部增益设置 (x2/x4/x8/x16)，未使用反相输入



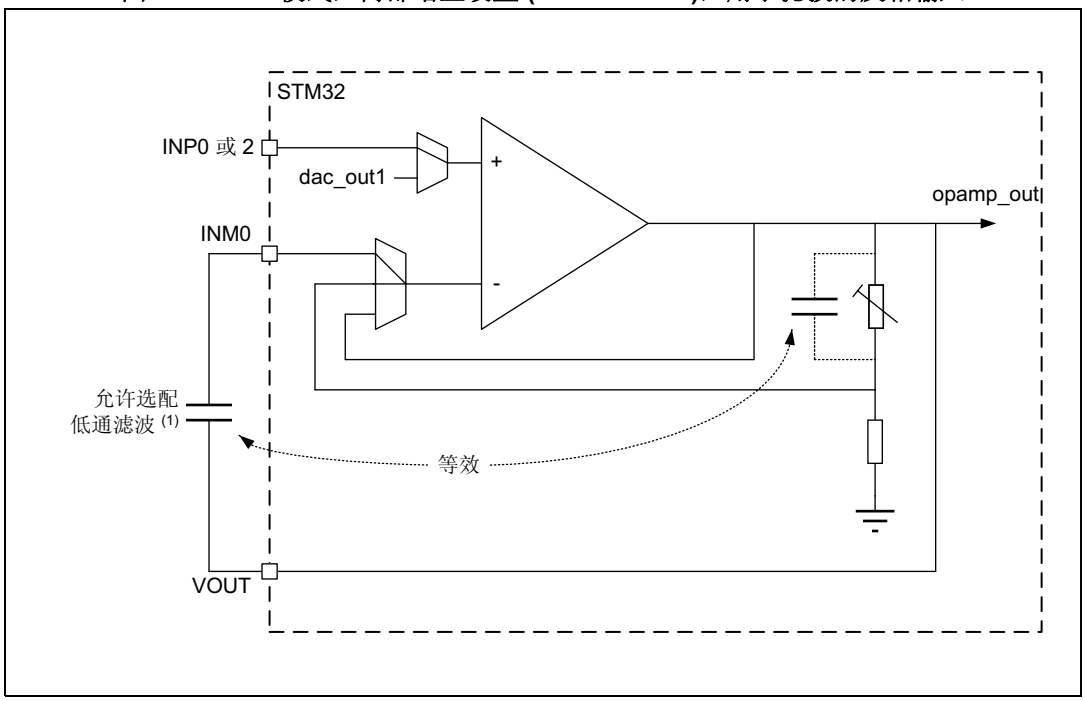
使用外部滤波的可编程增益放大器模式

下面介绍使用 OPAMP 和外部滤波来放大输入信号幅值的步骤。

- 将 VM_SEL 位配置为“反馈电阻连接到 OPAMPx_VINM 输入”(10)。
- 将 PGA_GAIN 位配置为“内部增益 2、4、8 或 16，在 INM0 上使用滤波”（0100 到 0111）。
- 将 VP_SEL 位配置为“GPIO 连接到 OPAMPx_VINP”。

  INM 上的任何外部连接均可与内部 PGA 并联使用。例如，可在 opamp_out 和 INM 之间连接一个电容来实现滤波目的。有关 PGA 电阻网络中使用的电阻值，请参见数据手册。

图 150. PGA 模式，内部增益设置 (x2/x4/x8/x16)，用于滤波的反相输入



1. 增益取决于截止频率。

可编程增益放大器，使用外部偏压的同相模式或反相模式

下面介绍使用 OPAMP 和偏置电压（针对同相模式，反相模式无需偏置电压）来放大输入信号幅值的步骤。

- 将 VM_SEL 位配置为“反馈电阻连接到 OPAMPx_VINM 输入”(10)。
- 将 PGA_GAIN 位配置为“反相增益 = -1、-3、-7、-15/同相增益 = 2、4、8、16，使用 INM0）”（1000 到 1011）。
- 将 VP_SEL 位配置为“GPIO 连接到 OPAMPx_VINP”。

图 151. PGA 模式，同相增益设置 (x2/x4/x8/x16) 或反相增益设置 (x-1/x-3/x-7/x-15)

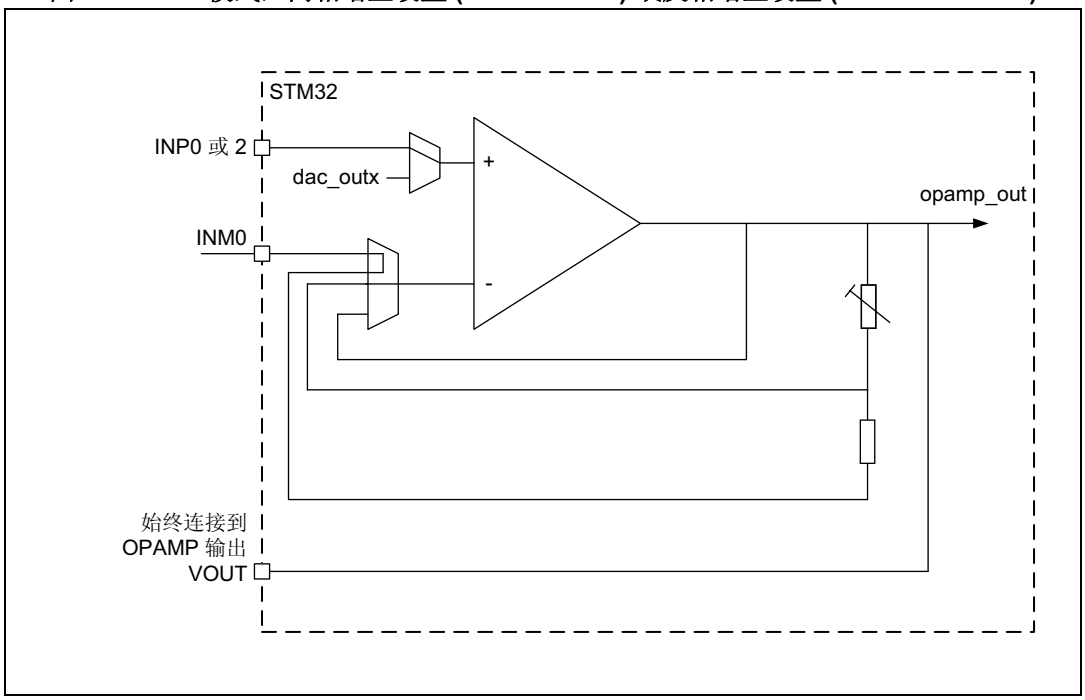
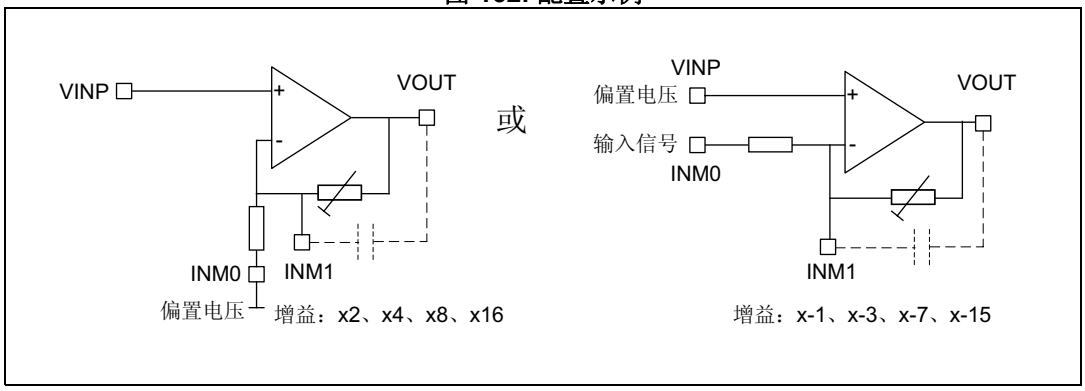


图 152. 配置示例



可编程增益放大器，使用外部偏压的同相模式或使用滤波的反相模式

下面介绍使用 OPAMP 和偏置电压（针对同相模式）或滤波（针对反相模式）来放大输入信号幅值的步骤。

- 将 VM_SEL 位配置为“反馈电阻连接到 OPAMPx_VINM 输入”(10)。
- 将 PGA_GAIN 位配置为“反相增益 = -1、-3、-7、-15/同相增益 = 2、4、8、16，使用 INM0 和用于滤波的 INM1 节点”（1100 到 1111）。
- 将 VP_SEL 位配置为“GPIO 连接到 OPAMPx_VINP”。

  VM1 上的任何外部连接均可与内部 PGA 并联使用。例如，可在 opamp_out 和 VM1 之间连接一个电容来实现滤波目的。有关 PGA 电阻网络中使用的电阻值，请参见数据手册。

图 153. PGA 模式，同相增益设置 (x2/x4/x8/x16) 或使用滤波的反相增益设置 (x-1/x-3/x-7/x-15)

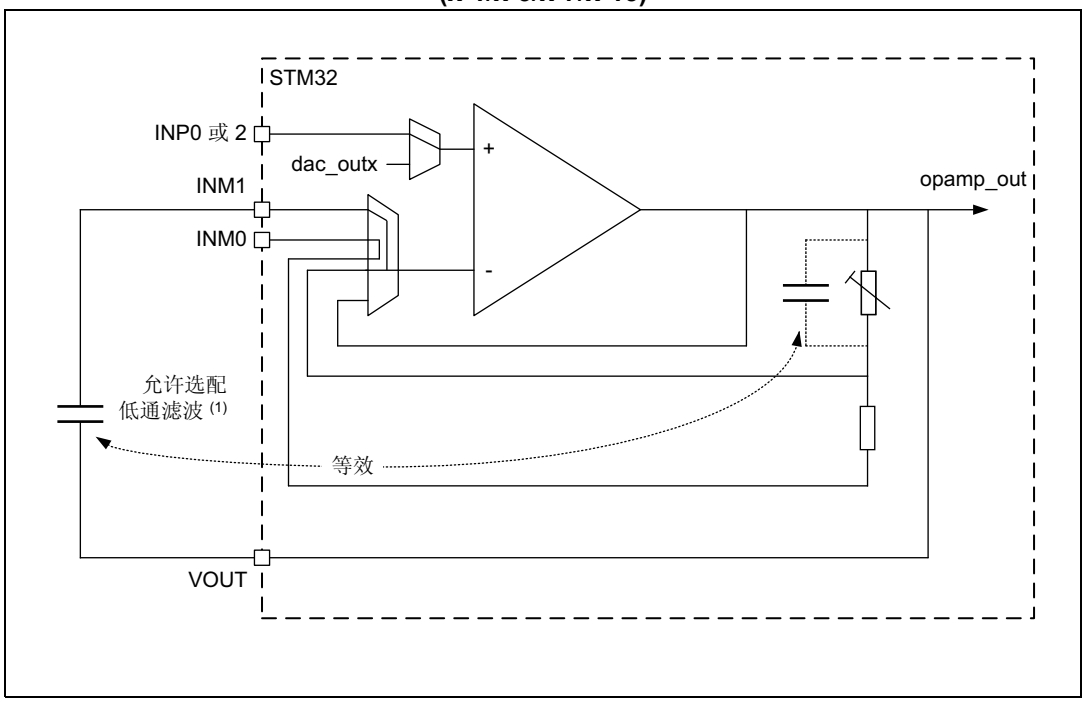
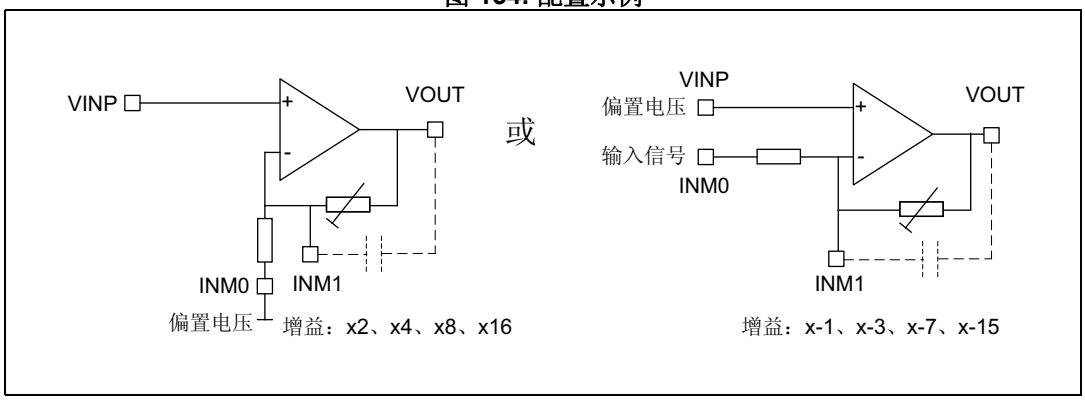


图 154. 配置示例



23.3.5 校准

OPAMP 接口连续将调整的偏移值发送至运算放大器。启动时，使用预设的“出厂”微调值初始化了微调值。

用户可以对每个运算放大器进行微调。对于正常模式和高速模式，特定寄存器允许用户使用不同的微调值。

校准的目的在于尽可能地减小 OPAMP 输入失调电压。在稳定的电压和温度条件下，校准电路允许用户将输入失调电压降至 +/-1.5 mV 以下。

对于每个运算放大器和每种模式，需要对两个微调值进行微调，一个值用于 N 差分对，另一个值用于 P 差分对。

对于每个运算放大器，有两个寄存器可用于微调偏移值，一个用于正常模式 (OPAMPx_OTR)，另一个用于高速模式 (OPAMPx_HSOTR)。每个寄存器都包含 5 个用于 P 差分对微调的位和 5 个用于 N 差分对微调的位。这些均为“用户”微调的值。

用户可使用 OPAMPx_CSR 寄存器的 USERTRIM 位从“出厂”值切换到“用户”微调的值。该位在启动时复位，因此默认情况下，“出厂”值被应用于 OPAMP 选项寄存器。

用户可在校准或功能模式下轻松更改微调值。

通常在将 CALON 位设置为 1 来初始化校准操作之后，才配置偏移微调寄存器。当 CALON = 1 时，运算放大器的输入从工作环境中断开。

- 将 CALSEL 置为 01 可初始化 P 差分对的偏移校准（使用低参考电压）。
- 将 CALSEL 复位为 11 可初始化 N 差分对的偏移校准（使用高参考电压）。

当 CALON = 1 时，CALOUT 位将反映 CALSEL 和 OPAHSM 所选择的微调值的影响。软件应将 OPAMP 控制寄存器中的 TRIMOFFSETN 位从 0x00 增至第一个导致 OPAMP 寄存器的 CALOUT 位从 1 变为 0 的值。如果 CALOUT 位复位，则偏移已正确校准，并且必须存储相应的微调值。在更改微调值之后，CALOUT 标志最多需要 1 ms 的时间达到稳定状态（请参见数据手册电气特性部分的 $t_{OFFTRIM}max$ 延迟规范）。

注意： *微调值越接近于最佳微调值，达到稳定状态所用的时间就越长。在任何情况下，最长稳定时间都低于 1 ms。*

表 149. 工作模式和校准

模式	控制位				输出	
	OPAEN	OPAHSM	CALON	CALSEL	V_{OUT}	CALOUT 标志
正常工作模式	1	0	0	X	模拟	0
高速模式	1	1	0	X	模拟	0
掉电	0	X	X	X	Z	0
正常模式的偏移校准 N 差分	1	0	1	11	模拟	X
正常模式的偏移校准 P 差分	1	0	1	01	模拟	X
高速模式的偏移校准 N 差分	1	1	1	11	模拟	X
高速模式的偏移校准 P 差分	1	1	1	01	模拟	X

校准程序

以下为对任一运算放大器执行完全校准的步骤：

1. 将 OPAMPx_CSR 的 OPAEN 位置 1，以使能运算放大器。
2. 将 OPAMPx_CSR 寄存器中的 USERTRIM 位置 1。
3. 选择一种校准模式（请参见*表 149：工作模式和校准*）。步骤 3 至 4 必须重复执行四次。首先，选择以下内容：
   – 正常模式和 N 差分对

   上述校准模式对应于 OPAMPx_CSR 寄存器中的 OPAHSM = 0，CALSEL = 11。
4. 从 00000b 开始递增 OPAMPx_OTR 的 TRIMOFFSETN[4:0]，直至 OPAMPx_CSR 的 CALOUT 变为 0。

注意： *在写入 OPAMPx_OTR 寄存器和读取 CALOUT 值之间，确保等待数据手册电气特性部分中指定的 $t_{OFFTRIM}max$ 延迟，以获得正确的 CALOUT 值。*

针对以下模式重复步骤 3 到 4：

- 正常模式和 P 差分对，CALSEL = 01
- 高速模式和 N 差分对
- 高速模式和 P 差分对

如果未使用某种模式，则无需执行相应的校准。

注意： *在整个校准阶段，运算放大器输出的外部连接不能上拉或下拉高于 500 µA 的电流。*

23.4 OPAMP 低功耗模式

表 150. 低功耗模式对 OPAMP 的影响

模式	说明
睡眠	无影响。
停止	无影响，OPAMP 寄存器内容保持不变。
待机	OPAMP 寄存器掉电，退出待机模式后必须重新初始化。

23.5 OPAMP PGA 增益

将 OPAMP 配置为 PGA 模式时，可对同相模式选择增益 x2、x4、x8 或 x16，对反相模式选择增益 x-1、x-3、x-7 或 x-15。

将 OPAMP 配置为同相模式时，增益误差在产品数据手册中指定。将其配置为反相模式时，增益系数不仅由片上反馈电阻定义，而且与信号源输出阻抗有关。如果信号源输出阻抗与 PGA 的输入反馈电阻相比，不可忽略不计，则会产生增益误差。请参见产品数据手册中的 PGA 阻抗值。

23.6 OPAMP 寄存器

该外设的寄存器只能按字（32 位）进行访问。

23.6.1 OPAMP1 控制/状态寄存器 (OPAMP1_CSR)

OPAMP1 control/status register

地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	CAL OUT	TST REF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USER TRIM	PGA_GAIN[3:2]	
	r	rw											rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PGA_GAIN[1:0]		CALSEL[1:0]		CALON	Res.	Res.	OPA HSM	Res.	VM_SEL[1:0]		Res.	VP_SEL[1:0]		FORCE _VP	OPAEN
rw	rw	rw	rw	rw			rw		rw	rw		rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30 **CALOUT**：运算放大器校准输出 (Operational amplifier calibration output)
OPAMP 输出状态标志。在校准模式下，OPAMP 用作比较器。
0：同相 < 反相
1：同相 > 反相

位 29 **TSTREF**：OPAMP 校准参考电压输出控制 (OPAMP calibration reference voltage output control)
（为测试保留）
0：未输出 OPAMP 的 INTVREF
1：输出 OPAMP 的 INTVREF

位 28:19 保留，必须保持复位值。

位 18 **USERTRIM**：用户微调使能 (User trimming enable)
该位用于从“出厂”AOP 偏移微调值切换至“用户”AOP 偏移微调值
对于正常模式和高速模式，该位均有效。
0：使用“出厂”微调代码
1：使用“用户”微调代码

位 17:14 **PGA_GAIN[3:0]**：可编程增益放大器增益设置 (Programmable gain amplifier gain setting)
0000：同相内部增益 2，参考 VREF
0001：同相内部增益 4，参考 VREF
0010：同相内部增益 8，参考 VREF
0011：同相内部增益 16，参考 VREF
0100：同相内部增益 2，在 INM0 上使用滤波，参考 VREF
0101：同相内部增益 4，在 INM0 上使用滤波，参考 VREF
0110：同相内部增益 8，在 INM0 上使用滤波，参考 VREF
0111：同相内部增益 16，在 INM0 上使用滤波，参考 VREF
1000：反相增益 = -1/同相增益 = 2，INM0 节点用于输入或偏置
1001：反相增益 = -3/同相增益 = 4，INM0 节点用于输入或偏置
1010：反相增益 = -7/同相增益 = 8，INM0 节点用于输入或偏置
1011：反相增益 = -15/同相增益 = 16，INM0 节点用于输入或偏置
1100：反相增益 = -1/同相增益 = 2，INM0 节点用于输入或偏置，INM1 节点用于滤波
1101：反相增益 = -3/同相增益 = 4，INM0 节点用于输入或偏置，INM1 节点用于滤波
1110：反相增益 = -7/同相增益 = 8，INM0 节点用于输入或偏置，INM1 节点用于滤波
1111：反相增益 = -15/同相增益 = 16，INM0 节点用于输入或偏置，INM1 节点用于滤波

位 13:12 **CALSEL[1:0]**：校准选择 (Calibration selection)
用于选择 CALON = 1 或 FORCE_VP = 1 时生成内部参考电压所用的偏移校准总线。
00：对 OPAMP 输入应用 0.033*VDDA
01：对 OPAMP 输入应用 0.1*VDDA（用于 PMOS 校准）
10：对 OPAMP 输入应用 0.5*VDDA
11：对 OPAMP 输入应用 0.9*VDDA（用于 NMOS 校准）

位 11 **CALON**：使能校准模式 (Calibration mode enabled)
0：正常模式
1：校准模式（通过硬件打开所有开关）

位 10:9 保留，必须保持复位值。

位 8 **OPAHSM**：运算放大器高速模式 (Operational amplifier high-speed mode)
要更改此配置，必须禁止运算放大器。
0：运算放大器处于正常模式
1：运算放大器处于高速模式

位 7 保留，必须保持复位值。

位 6:5 **VM_SEL[1:0]**：反相输入选择 (Inverting input selection)
00：INM0 连接到 OPAMP_VINM 输入
01：INM1 连接到 OPAMP_VINM 输入
10：反馈电阻连接到 OPAMP_VINM 输入（PGA 模式），反相输入选择取决于 PGA_GAIN 设置
11：opamp_out 连接到 OPAMP_VINM 输入（跟随器模式）

位 4 保留，必须保持复位值。

位 3:2 **VP_SEL[1:0]**：同相输入选择 (Non inverted input selection)
00：GPIO INP0 连接到 OPAMP_VINP
01：dac_out1 连接到 OPAMP_VINP
10：GPIO INP2 连接到 OPAMP_VINP
11：保留

位 1 **FORCE_VP**：强制内部 VP 参考 (Force internal reference on VP)（为测试保留）
0：正常工作模式：同相输入连接到输入。
1：校准验证模式：同相输入连接到校准参考电压。

位 0 **OPAEN**：运算放大器使能 (Operational amplifier enable)

0：禁止运算放大器

1：使能运算放大器

注意：如果 OPAMP1 在特定封装中未连接，则必须保持禁止状态（保持 OPAMP1_CSR 寄存器默认值）。

23.6.2 正常模式下的 OPAMP1 微调寄存器 (OPAMP1_OTR)

OPAMP1 trimming register in normal mode

地址：0x04

复位值：0x0000 XXXX（出厂微调值）

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

位 31:13 保留，必须保持复位值。

位 12:8 **TRIMOFFSETP[4:0]**：PMOS 差分对的微调 (Trim for PMOS differential pairs)

位 7:5 保留，必须保持复位值。

位 4:0 **TRIMOFFSETN[4:0]**：NMOS 差分对的微调 (Trim for NMOS differential pairs)

23.6.3 高速模式下的 OPAMP1 微调寄存器 (OPAMP1_HSOTR)

OPAMP1 trimming register in high-speed mode

地址：0x08

复位值：0x0000 XXXX（出厂微调值）

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

位 31:13 保留，必须保持复位值。

位 12:8 **TRIMHSOFFSETP[4:0]**：PMOS 差分对的高速模式微调 (High-speed mode trim for PMOS differential pairs)

位 7:5 保留，必须保持复位值。

位 4:0 **TRIMHSOFFSETN[4:0]**：NMOS 差分对的高速模式微调 (High-speed mode trim for NMOS differential pairs)

23.6.4 OPAMP 寄存器映射

表 151. OPAMP 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**OPAMP1_CSR**	Res.	CALOUT	TSTREF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USERTRIM	PGA_GAIN				CALSEL		CALON	Res.	Res.	OPAHSM	Res.	VM_SEL		Res.	VP_SEL		FORCE_VP	OPAEN
	Reset value		0	0											0	0	0	0	0	0	0	0			0		0	0		0	0	0	0
0x04	**OPAMP1_OTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIM OFFSETP[4:0]					Res.	Res.	Res.	TRIM OFFSETN[4:0]				
	Reset value																				(1)								(1)				
0x08	**OPAMP1_ HSOTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIMHS OFFSETP[4:0]					Res.	Res.	Res.	TRIMHS OFFSETN[4:0]				
	Reset value																				(1)								(1)				

1. 出厂微调值。

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

24 真随机数发生器 (RNG)

24.1 简介

RNG 是一个真随机数发生器，可向应用程序提供作为 32 位样本的全熵输出，它由一个实时熵源（模拟）和一个内部调节组件构成。

RNG 是符合 NIST SP 800-90B 标准的熵源，可用于构建非确定性随机位发生器 (NDRBG)。

RNG 真随机数发生器已通过 NIST SP800-90B 标准预先认证，也已按照德国 BSI AIS-31（T0 到 T8）统计测试进行了测试。

24.2 RNG 主要特性

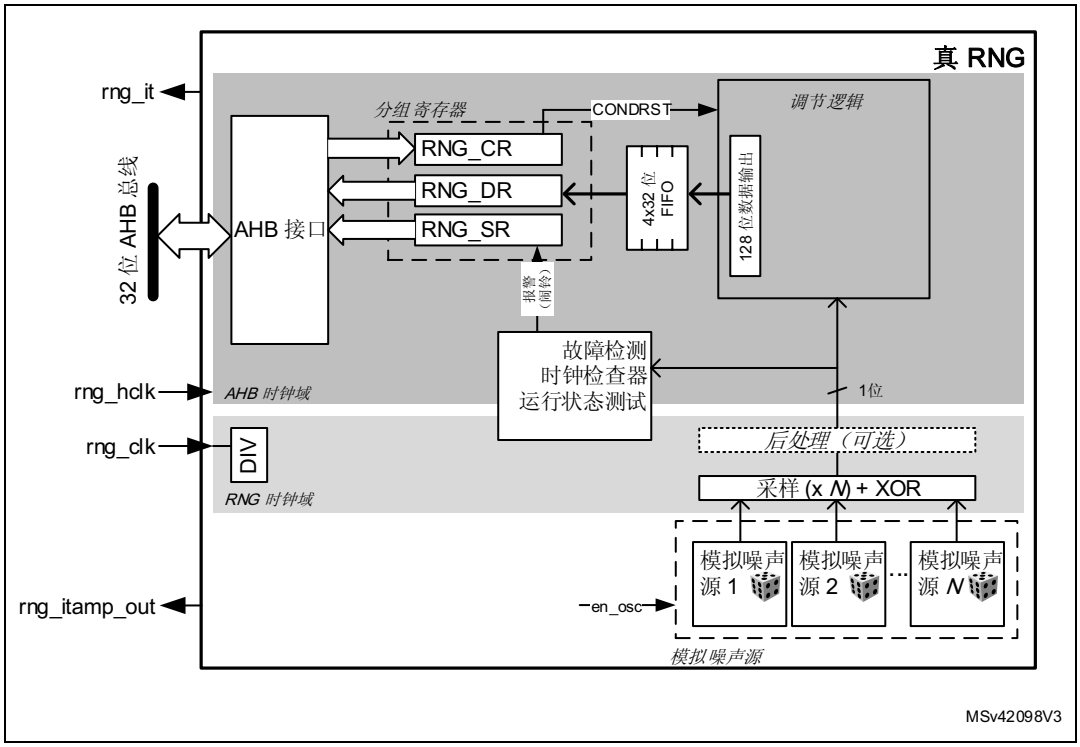
- RNG 提供的 32 位真随机数由模拟熵源生成，并通过符合 NIST SP800-90B 标准要求的调节级进行调节。
- 可用作熵源以构建非确定性随机位发生器 (NDRBG)。
- 在 NIST 配置中，如果 $f_{AHB} < f_{threshold}$，则 RNG 每 412 个 AHB 时钟周期会生成四个 32 位随机样本（否则为 256 个 RNG 时钟周期）。
- 内置启动和符合 NIST SP800-90B 标准认证的连续运行状态测试（重复计数和自适应比例测试）及相关的特定错误管理
- 可被禁止以降低功耗，或以自动低功耗模式使能（默认配置）。
- 具有 AMBA® AHB 从外设，只能通过 32 位字进行单次访问（否则会生成 AHB 总线错误，并会忽略写访问）。

24.3 RNG 功能描述

24.3.1 RNG 框图

图 155 显示了 RNG 框图。

图 155. RNG 框图



24.3.2 RNG 内部信号

表 152 中列出了 RNG 级而非 STM32 产品级（焊盘上）所提供的内部信号，对这些信号有所了解会很有用。

表 152. RNG 内部输入/输出信号

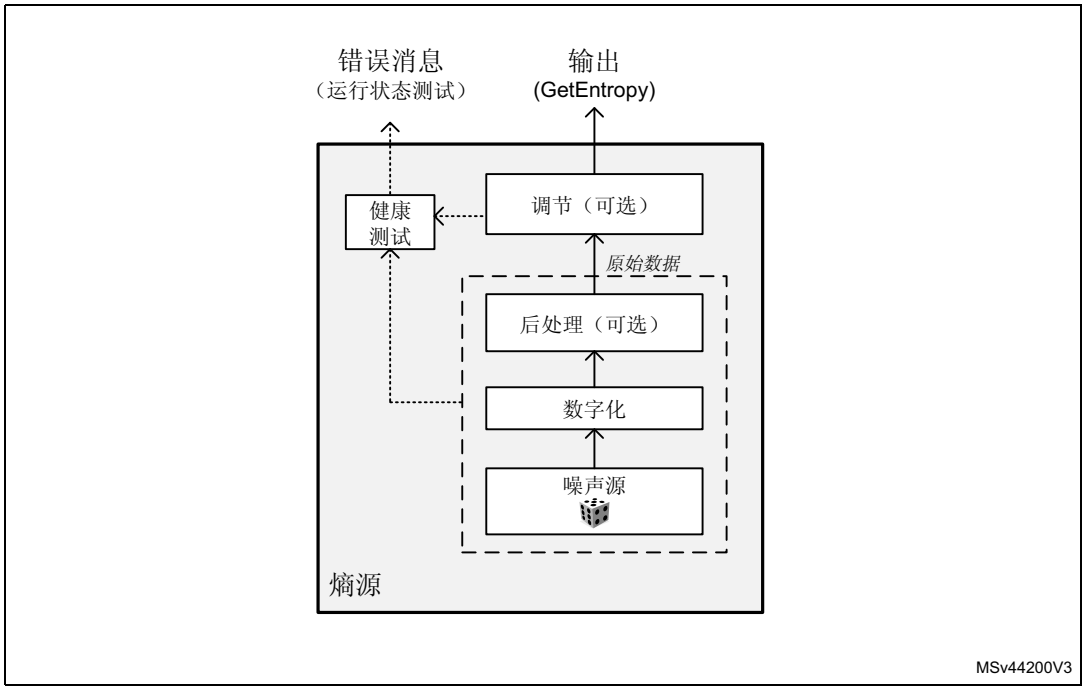
信号名称	信号类型	说明
rng_it	数字输出	RNG 全局中断请求
rng_hclk	数字输入	AHB 时钟
rng_clk	数字输入	RNG 专用时钟，与 rng_hclk 异步
rng_itamp_out	数字输出	发送到 TAMP 的 RNG 内部入侵事件信号（异或运算），在发生意外硬件故障时触发。该信号被触发后，RNG 将停止提供随机样本，需要复位并重新初始化才能再次使用。

24.3.3 随机数生成

真随机数发生器 (RNG) 按确定的间隔通过其 AHB 接口提供真随机数。

RNG 在其边界内集成了所有必需的 NIST 组件（如*图 156* 所示）。这些组件包括模拟噪声源、数字化处理级、调节算法、运行状态监视模块以及两个用于与熵源交互的接口：GetEntropy 和 HealthTest。

图 156. NIST SP800-90B 熵源模型



下文将详细介绍上图所示的组件。

噪声源

噪声源组件包含不确定性的、能够提供熵值的活动，其最终决定了与熵源位串输出相关联的不确定性。该噪声源提供 1 位样本。它包括：

- 多个模拟噪声源 (x6)，每个噪声源由三个自由运行的环形振荡器输出异或 (XOR) 而成。可以禁止这些模拟振荡器以实现节能，如*第 24.3.8 节：RNG 低功耗使用*所述。
- 将所有噪声源的异或运算结果作为单个模拟输出。
- 该输出的采样级由专用时钟输入（rng_clk 通过集成分频器）提供时钟信号，输出 1 位原始数据。

该噪声源采样与 AHB 接口时钟频率 (rng_hclk) 无关，软件可通过使用集成分频器来降低采样频率。

注意： *第 24.6 节：RNG 熵源验证中提供了建议使用的 RNG 时钟频率及相关分频值。*

后期处理

在 NIST 配置中，不会对采样的噪声源进行后期处理。在非 NIST 配置 B 中（如*第 24.6.2 节*中定义），会进行归一化去偏置处理，即一半的位取自采样噪声源，一半的位是对取自采样噪声源的值取反。

调节

RNG 中的调节组件是确定性函数，可以提高生成固定长度位串输出（128 位）的熵率。NIST SP800-90B 的目标是提供全熵输出（128 位）。

*第 24.5 节：RNG 处理时间*中给出了生成两个随机数之间和 RNG 初始化之间所需的时间，以及第一个样本的可用性的说明。

输出缓冲区

数据输出缓冲区最多可存储四个从调节组件输出的 32 位字。如果已通过 RNG_DR 寄存器从输出 FIFO 读取了四个字，则 128 位调节输出寄存器的内容会被推送到输出 FIFO 中，同时自动启动新的一轮调节。经过*第 24.5 节：RNG 处理时间*中指定的时钟周期数后，会将四个新字添加到调节输出寄存器。

只要通过 RNG_DR 寄存器获取随机数时，DRDY 标志就从 0 变为 1。该标志会保持在置 1 状态，直至从 RNG_DR 寄存器读取四个字后输出缓冲区变空。

注意： *如果使能了中断，则当该数据就绪标志从 0 变为 1 时，会产生中断。随后，中断会自动由 RNG 清零，如上所述。*

运行状态检查

该组件可确保整个熵源（包括其噪声源）正常启动并按预期运行，能够快速准确地定位故障，从而提供了高可靠性保证。

根据 NIST SP800-90B 标准，RNG 可实现以下运行状态检查功能。规定的阈值对应于寄存器 RNG_HTCR 的建议值（*第 24.6.2 节*中的配置 A）。

1. 启动运行状态测试（在复位后和在首次将 RNG 用作熵源前执行）：
   - 重复计数测试，在噪声源持续不变地输出了 42 位或以上的 0 或 1 时指示错误。
   - 在 1024 个连续位窗口上运行的自适应比例测试：RNG 验证噪声源输出上的第一个位重复次数是否超过 711 次。
   - 已知答案测试，用于验证调节级。
2. 连续运行状态测试（无限期地在噪声源输出端运行）：
   - 重复计数测试，类似于启动测试中运行的测试。
   - 自适应比例测试，类似于启动测试中运行的测试。
3. 厂商专用连续测试
   - 转换计数测试，当噪声源持续不变地输出了 32 位或以上的 01 或 10 时指示错误。
   - 实时“过慢”采样时钟检测器，如果一个 RNG 时钟周期（分频前）小于 AHB 时钟周期的 32 分之一，则该检测器会指示错误。
4. 数字化噪声源按需测试（原始数据）
   - 通过重新启动熵源并重新运行启动测试来提供支持（请参见*第 24.3.4 节：RNG 初始化*中的软件复位序列）。*不支持*其他类型的按需测试（基于软件）。

RNG_SR 寄存器中的 CECS 和 SECS 状态位会在检测到错误条件时进行指示，*第 24.3.7 节：错误管理*对此进行了详细说明。

注意： *检测到错误时可产生中断。*

可通过更改 RNG_HTCR 寄存器中的值来修改超出运行状态测试阈值的值。详细信息，请参见第 24.6 节：RNG 熵源验证。

24.3.4 RNG 初始化

RNG 简化状态机如*图 157* 所示

在使能 RNG（RNG_CR 中的 RNGEN = 1）后，会发生以下一系列事件：

1. 使能模拟噪声源，默认情况下，RNG 在等待 16 个 RNG 时钟周期（分频前）后开始采样模拟输出并填充 128 位调节移位寄存器。
2. 调节硬件初始化，自动触发对原始数据样本的启动行为测试和已知答案测试。
3. 启动运行状态测试完成。在此期间，使用三个 128 位噪声源样本。
4. 再次使用 128 位数据填充调节级内部输入数据缓冲区并执行调节，调节轮数由 RNG 配置（NIST 或非 NIST）定义。然后使用后期处理结果填充输出缓冲区。
5. 输出缓冲区会按照 RNG 使用情况自动重填。

相关初始化时间请参见*第 24.5 节：RNG 处理时间*。

图 157. RNG 初始化概述

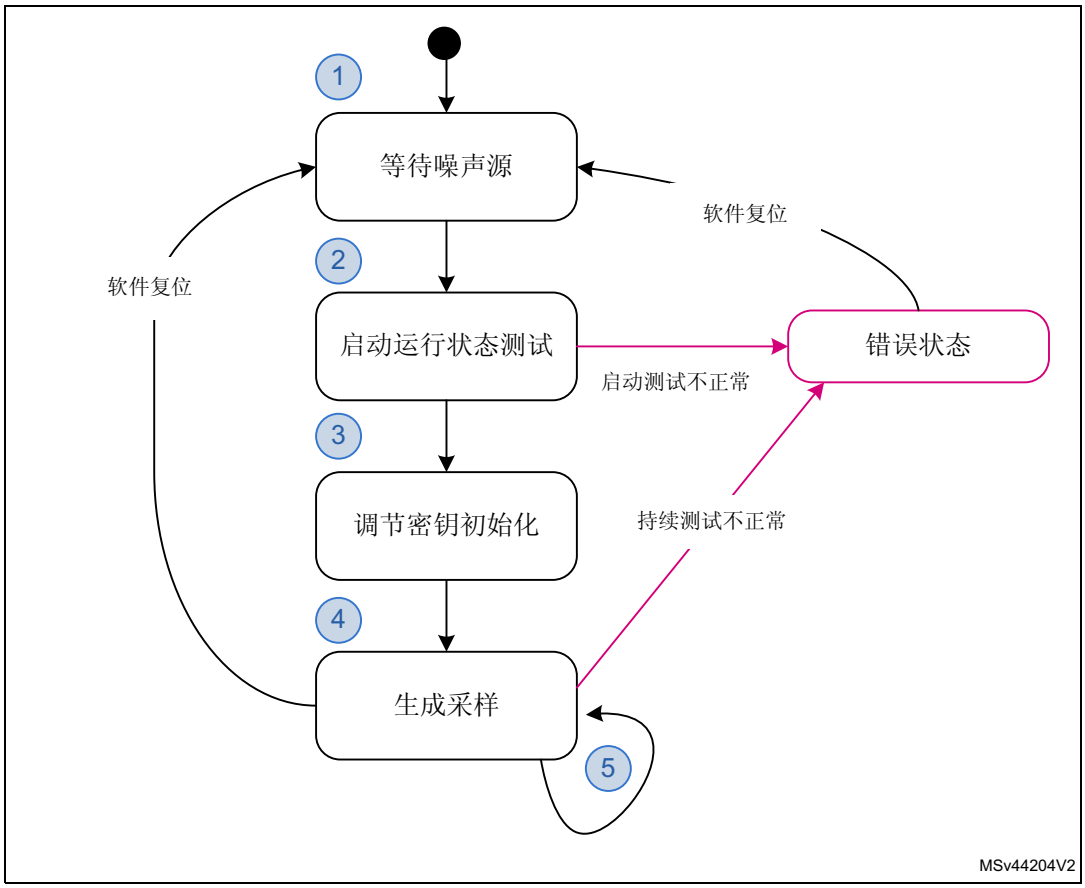


图 157 还重点说明了可能的软件复位序列，其实现方式如下：

1. 向 RNG_CR 寄存器中写入 RNGEN = 0 和 CONDRST = 1，并使用相同的 RNG 配置和新的 CLKDIV（如果需要）。
2. 然后，向 RNG_CR 寄存器中写入 RNGEN = 1 和 CONDRST = 0。
3. 初始化完成后，等待随机数就绪。

注意： *当 RNG 外设通过 RCC（硬件复位）复位时，RNG 寄存器中可实现最佳随机性的 RNG 配置将丢失。通过将 CONFIGLOCK 置 1 来执行软件复位时可保留 RNG 配置。*

24.3.5 RNG 操作

正常工作

要使用中断运行 RNG，建议执行以下步骤：

1. 查阅*第 24.6 节：RNG 熵源验证*并验证应用是否需要特定的 RNG 配置。
   - 如果需要，则向 RNG_CR 寄存器中写入 CONDRST = 1 以及正确的 RNG 配置。然后，对 RNG_CR 寄存器执行第二次写操作：CONDRST = 0、中断使能位 IE = 1 且 RNG 使能位 RNGEN = 1。
   - 如果不需要，则对 RNG_CR 寄存器执行写操作：中断使能位 IE = 1 且 RNG 使能位 RNGEN = 1。
2. 这以后，准备好随机数时或出现错误时就产生中断。因此，每次发生中断时，应检查：
   - 未发生错误。RNG_SR 寄存器中的 SEIS 和 CEIS 位必须设为 0。
   - 随机数准备就绪。RNG_SR 寄存器中的 DRDY 位必须置 1。
   - 如果满足以上两个条件，最多可连续四次读取 RNG_DR 寄存器的内容。如果调节输出缓冲区中存在有效数据，则应用程序可以额外读取四个字（此时 DRDY 位仍保持为 1）。如果上述条件中有一个或两个都无法满足，则不得读取 RNG_DR 寄存器。如果发生错误，则必须使用*第 24.3.7 节*中介绍的错误恢复序列。

要在轮询模式下运行 RNG，建议执行以下步骤：

1. 查阅*第 24.6 节：RNG 熵源验证*并验证应用是否需要特定的 RNG 配置。
   - 如果需要，则向 RNG_CR 寄存器中写入 CONDRST = 1 以及相应的 RNG 配置。然后，对 RNG_CR 寄存器执行第二次写操作：CONDRST = 0 且 RNG 使能位 RNGEN = 1。
   - 如果不需要，则只需将 RNG_CR 寄存器中的 RNGEN 位置 1 即可使能 RNG。
2. 读取 RNG_SR 寄存器并检查：
   - 未发生错误（SEIS 和 CEIS 位必须设为 0）
   - 随机数准备就绪（DRDY 位必须置 1）
3. 如果满足以上两个条件，最多可连续四次读取 RNG_DR 寄存器的内容。如果调节输出缓冲区中存在有效数据，则应用程序可以额外读取四个字（此时 DRDY 位仍保持为 1）。如果上述条件中有一个或两个都无法满足，则不得读取 RNG_DR 寄存器。如果发生错误，则必须使用*第 24.3.7 节*中介绍的错误恢复序列。

注意： *如果数据未就绪 (DRDY = 0)，则 RNG_DR 会返回 0。*
建议始终验证 RNG_DR 是否不为零。因为如果不为零，在轮询 RNG_SR 与读取 RND_DR 输出之间可能会发生种子错误（罕见事件）。

如果应用程序比较关注随机数生成周期，并且不需要符合 NIST 标准，则可以使用 RNG 配置“B”来选择更快的 RNG 配置，如*第 24.6 节：RNG 熵源验证*中所述。*第 24.5 节：RNG 处理时间*中总结了使用随机数生成速度的好处。

低功耗运行

如果应用程序比较关注功耗，则可以使用低功耗策略，如*第 24.3.8 节：RNG 低功耗使用*所述。

软件后期处理

无需进行特定的软件后期处理/调节即可满足 AIS-31 或 NIST SP800-90B 认证。

*第 24.3.3 节：随机数生成*介绍了内置的运行状态检查功能。

24.3.6 RNG 时钟

RNG 在两个不同的时钟上运行：AHB 总线时钟和专用 RNG 时钟。

AHB 时钟用于为 AHB 存储寄存器和调节组件提供时钟信号。RNG 时钟与可编程分频器搭配使用（请参见 RNG_CR 寄存器中的 CLKDIV 位域），用于噪声源采样。*第 24.6 节：RNG 熵源验证*中详细介绍了建议使用的时钟配置。

注意： *如果 RNG_CR 寄存器中的 CED 位置 0，那么 RNG 时钟频率（内部分频前）必须大于 AHB 时钟频率的 32 分之一，否则时钟检查器会始终指示时钟错误（RNG_SR 寄存器中的 CECS = 1）。*

有关 AHB 和 RNG 时钟域的详细信息，请参见*第 24.3.1 节：RNG 框图*。

24.3.7 错误管理

运行状态检查模块在生成随机数的同时还验证噪声源行为以及 RNG 源时钟频率是否正常，具体说明见本部分。另外，还描述了相关的错误状态。

时钟错误检测

如果时钟错误检测已使能 (CED = 0) 且 RNG 时钟频率过低，RNG 会将 CEIS 和 CECS 位都置 1，以指示发生了时钟错误。在这种情况下，应用程序必须检查 RNG 时钟是否配置正确（参见*第 24.3.6 节：RNG 时钟*），随后必须将中断标志 CEIS 位清零。当时钟状态正常时，CECS 位自动清零。

注意： *时钟错误对生成的随机数没有影响，即应用程序仍然可以读取 RNG_DR 寄存器。*

仅当 RNG 将 CECS 置 1 时，CEIS 才会置 1。

噪声源错误检测

如果发生噪声源（或种子）错误，RNG 会停止生成随机数，并会将 SEIS 和 SECS 位都置 1，以指示发生了种子错误。如果 RNG_DR 寄存器中有可用值，不得使用该值，因为它可能没有足够的熵。

要从种子错误中完全恢复，必须使用以下序列：

1. 先后向 CONDRST 中写入 1 和 0 以执行软件复位（有关详细信息，请参见位域说明）。仅当 SECS 置 1 时才需要执行此步骤。事实上，当 SEIS 置 1 且 SECS 清零时，意味着 RNG 会自动执行复位（自动复位）。在这种情况下，应用程序必须将中断标志位 SEIS 清零。
2. 如果在步骤 1 中将 SECS 置 1（不自动复位），则需等待 RNG_CR 寄存器中的 CONDRST 清零，然后确认 RNG_SR 寄存器中的 SEIS 是否清零。否则，只需将 RNG_SR 寄存器中的 SEIS 位清零即可。
3. 如果在步骤 1 中将 SECS 置 1（不自动复位），则需等待 RNG 将 SECS 清零。随机数生成于是恢复正常。

注意： *发生种子错误后，当 SECS 清零时，RNG 会重新开始生成随机数。*

应用程序将 RNG_CR 寄存器中的 ARDIS 位置 1 后，将禁止自动复位。必须在步骤 1 中使用 CONDRST。

RNG 入侵错误

当 RNG 发现意外错误时，TAMP 外设中将会触发内部入侵事件，并且 RNG 将停止提供随机数据。

该事件发生后，安全应用程序需使用中央复位管理或全局 SoC 复位来复位 RNG 外设。之后，需要再次正确初始化 RNG。

24.3.8 RNG 低功耗使用

考虑到功耗问题，可在 DRDY 位置 1 后立即禁止 RNG，具体方法为将 RNG_CR 寄存器中的 RNGEN 位设为 0。由于 RNGEN = 0 时后期处理逻辑和输出缓冲区仍处于工作状态，因此软件可使用以下功能：

- 如果输出缓冲区中存在有效字，仍可从 RNG_DR 寄存器中读取四个随机数。
- 如果调节输出内部寄存器中存在有效位，仍可从 RNG_DR 寄存器中额外读取四个随机数。如果不属于以上两种情况，则必须由应用程序重新使能 RNG，直至达到预期的新噪声源位阈值（NIST 模式下为 128 位）并且完成了一轮完整的调节。
  只有达到预期的调节轮数时（NISTC = 0 时为两轮），四个新的随机字才可用。相关总时间请参见*第 662 页的第 24.5 节：RNG 处理时间*。

禁止 RNG 时，用户会禁用所有模拟种子发生器，各发生器的功耗列于数据手册中的电气特性部分。用户还会对所有由 RNG 时钟提供时钟信号的逻辑系统进行门控。请注意，由于 RNG 需要一定的时间进行初始化，因此该策略导致在一段延时后才可在 RNG_DR 寄存器中提供随机样本。

如果 RNG 模块在初始化期间（也就是刚好在 DRDY 位首次置 1 之前）被禁止，则初始化序列将从 RNGEN 位置 1 时停止的位置恢复运行，但应用程序使用 RNG_CR 寄存器中的 CONDRST 位复位调节逻辑时除外。

当应用程序希望对 RNG 时钟进行门控时，建议在 RNGEN 位清零后等待两个 RNG 内核时钟周期，然后再使用 RCC 对 RNG 内核时钟进行门控。

此外，当应用程序需要进入 RNG 停用的功耗模式时，也建议在 RNGEN 位清零后等待两个 RNG 内核时钟周期，然后再使用 PWR 进入低功耗模式。

在上述两种情况下，为避免在 RNG 模拟振荡器保持工作状态时产生意外消耗，应用程序可将 RNG_CR 寄存器中的位 13 置 1。当 RNGEN 位置 1（RNG 激活）时，将该位置 1 会增加一些边际功耗。

注意： *有关 RNG 停用（保留或不可用）的功耗模式，请参见 PWR 部分。*

24.4 RNG 中断

在 RNG 中，发生以下事件时会产生中断：

- 数据就绪标志
- 种子错误，请参见 *第 24.3.7 节：错误管理*
- 时钟错误，请参见 *第 24.3.7 节：错误管理*

可以使用专用的中断使能控制位，如*表 153* 所示。

表 153. RNG 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法
RNG	数据就绪标志	DRDY	IE	无（自动）
	种子错误标志	SEIS	IE	向 SEIS 写入 0，或向 CONDRST 写入 1 然后写入 0
	时钟错误标志	CEIS	IE	向 CEIS 写入 0

用户可以通过更改 RNG_CR 寄存器中的屏蔽位或通用中断控制位 IE 的方式分别使能或禁止上述中断源。可以从 RNG_SR 寄存器中读取各个中断源的状态。

注意： *仅当 RNG 已使能时，才会产生中断。*

24.5 RNG 处理时间

在*表 154* 中介绍的建议配置 A 中，两组四个 32 位数据之间的时间为：

- 206 x N 个 AHB 周期（如果 $f_{AHB} < f_{threshold}$（调节级实施限制）），或者
- 128 x N 个 RNG 周期（如果 $f_{AHB} \geq f_{threshold}$（噪声源级实施限制））。

其中 $f_{threshold} = 1.6 \times f_{RNG}$，例如 f_{RNG}= 48 MHz 时，该频率为 77 MHz。值 N 为 2。

注意： *当 CLKDIV 不等于 0 时，f_{RNG} 必须考虑内部分频比。*

如果选择配置 B，则性能数据将变为：

- 206 个 AHB 周期（如果 $f_{AHB} < f_{threshold}$），或者
- 32 个 RNG 周期（如果 $f_{AHB} \geq f_{threshold}$）

其中 $f_{threshold} = 6.5 \times f_{RNG}$。

24.6 RNG 熵源验证

24.6.1 简介

为了评估 RNG 可提供的熵大小，意法半导体通过德国 BSI AIS-31 统计测试（T0 到 T8）以及 NIST SP800-90B 测试套件对外设进行了测试。测试结果可根据需要提供，客户也可以重现测量结果。

24.6.2 验证条件

意法半导体在以下条件下对 RNG 真随机数发生器进行了测试：

- RNG 时钟 rng_clk = 48 MHz
- RNG 配置如*表 154：RNG 配置*所述。请注意，只有配置 A 可通过 NIST SP800-90B 标准认证。

表 154. RNG 配置

RNG 配置	RNG_CR 位						轮数 (N)	RNG_HTCR 寄存器
	NISTC 位	RNG_CONFIG1 [5:0]	CLKDIV [3:0]	RNG_CONFIG2 [2:0]	RNG_CONFIG3 [3:0]	CED 位		
A	0	0x0F	0x0(1)	0x0(2)	0xD	0	2	0x0000 AAC7(3)
B	1	0x18	0x0	0x0	0x0	0	1	0x0000 AAC7

1. 对于 NIST 认证，噪声源采样频率必须为 48 MHz 或更低。因此，如果 RNG 时钟频率不等于 48 MHz，则必须调整 CLKDIV 的值。有关详细信息，请参见*第 24.7.1 节*中的 CLKDIV 位域说明。
2. 对于 RNG 功耗极为敏感的情况，建议使用值 0x1。有关详细信息，请参见*第 24.3.8 节：RNG 低功耗使用*结尾部分。
3. 对应于 42（重复计数测试）和 711（自适应比例测试）。详细信息，请参见*第 657 页的运行状态检查*。

24.6.3 数据采集

为了运行统计测试，需要对熵源的原始数据和输出进行采样。有关数据收集和统计测试套件运行的详细信息，请参见“使用 NIST 统计测试套件验证 STM32 微控制器随机数生成”应用笔记 (AN4230)（可访问 *www.st.com* 获取）。

如果需要为产品检索上述样本，请联系意法半导体。

24.7 RNG 寄存器

RNG 与控制寄存器、数据寄存器和状态寄存器相关联。

24.7.1 RNG 控制寄存器 (RNG_CR)

RNG control register

偏移地址：0x000

复位值：0x0080 0D00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CONFIGLOCK	CONDRST	Res.	Res.	Res.	Res.	RNG_CONFIG1[5:0]						CLKDIV[3:0]			
rs	rw					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RNG_CONFIG2[2:0]			NISTC	RNG_CONFIG3[3:0]				ARDIS	Res.	CED	Res.	IE	RNGEN	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw		rw		rw	rw		

位 31 **CONFIGLOCK**：RNG 配置锁定 (RNG Config lock)

0：允许写入 RNG_HTCR 和 RNG_CR 配置位 [29:4]。

1：忽略对 RNG_HTCR 和 RNG_CR 配置位 [29:4] 的写操作，直到下一次 RNG 复位。

该位域只能设置一次：如果该位置 1，则只有在 RNG 复位时才能复位为 0。

位 30 **CONDRST**：调节软复位 (Conditioning soft reset)

写入 1 然后写入 0，可复位调节逻辑、清空所有 FIFO 并开始重新初始化 RNG（RNG_SR 寄存器清零）。CONDRST 不会更改寄存器 RNG_CR 和 RNG_HTCR。

在设置任何配置位 [29:4] 的同一访问中，必须将该位置 1。换句话说，当 CONDRST 位置 1 时，也必须在位 [29:4] 中写入正确配置。

当 CONDRST 由软件设置为 0 时，其值在复位过程完成后将变为 0。大约需要 2 个 AHB 时钟周期 + 2 个 RNG 时钟周期。

位 29:26 保留，必须保持复位值。

位 25:20 **RNG_CONFIG1[5:0]**：RNG 配置 1 (RNG configuration 1)

保留用于 RNG 配置（位域 1）。必须使用*第 24.6 节：RNG 熵源验证*中记录的建议值进行初始化。

仅当在同一访问中 CONDRST 位置 1，而 CONFIGLOCK 保持为 0 时，才允许写入 RNG_CONFIG1 的任意位。如果 CONFIGLOCK = 1，则忽略对该位的写操作。

位 19:16 **CLKDIV[3:0]**：时钟分频系数 (Clock divider factor)

该值用于配置输入 RNG 时钟的内部可编程分频器（从 1 到 16）。这些位只能在内核已禁止 (RNGEN = 0) 时写入。

0x0：分频后的内部 RNG 时钟等同于输入 RNG 时钟。

0x1：每个内部 RNG 时钟周期为两个 RNG 时钟周期。

0x2：每个内部 RNG 时钟周期为 2^2 (= 4) 个 RNG 时钟周期。

...

0xF：每个内部时钟周期为 2^{15} 个 RNG 时钟周期（例如，48 MHz 输入 RNG 时钟变为 1.5 kHz 内部 RNG 时钟）

仅当在同一访问中 CONDRST 位置 1，而 CONFIGLOCK 保持为 0 时，才允许写入这些位。如果 CONFIGLOCK = 1，则忽略对该位的写操作。

位 15:13 **RNG_CONFIG2[2:0]**：RNG 配置 2 (RNG configuration 2)

保留用于 RNG 配置（位域 2）。对于 RNG 功耗极为敏感的情况，可以将位 13 置 1。请参见*第 24.3.8 节：RNG 低功耗使用*。有关详细信息，请参见 RNG_CONFIG1 位域。

位 12 **NISTC**：NIST 自定义 (NIST custom)

0：符合 NIST 标准的 RNG 的硬件默认值。在此配置中，每 128 位输出执行两轮调节并使用 256 位噪声源。

1：符合 NIST 标准的 RNG 的自定义值。关于建议使用的配置，请参见*第 24.6 节：RNG 熵源验证*。

仅当在同一访问中 CONDRST 位置 1，而 CONFIGLOCK 保持为 0 时，才允许写入该位。如果 CONFIGLOCK = 1，则忽略对该位的写操作。

位 11:8 **RNG_CONFIG3[3:0]**：RNG 配置 3 (RNG configuration 3)

保留用于 RNG 配置（位域 3）。有关详细信息，请参见 RNG_CONFIG1 位域。

如果该寄存器中的 NISTC 位清零，RNG 会忽略 RNG_CONFIG3 位域值。

位 7 **ARDIS**：自动复位禁止 (Auto reset disable)

0：当发生噪声源错误时，RNG 执行自动复位以将 SECS 位清零。

1：当发生噪声源错误时，应用程序必须通过向 CONDRST 写入 1 然后写入 0 来复位 RNG，以重新启动随机数生成。

当使能自动复位时，在出现噪声源错误后，应用程序仍需要将 SEIS 位清零。

仅当在同一访问中 CONDRST 位置 1，而 CONFIGLOCK 保持为 0 时，才允许写入该位。如果 CONFIGLOCK = 1，则忽略对该位的写操作。

位 6 保留，必须保持复位值。

位 5 **CED**：时钟错误检测 (Clock error detection)

0：使能时钟错误检测

1：禁止时钟错误检测

当使能 RNG 时，不能实时使能或禁止时钟错误检测，即必须禁止 RNG 才能使能或禁止 CED。

仅当在同一访问中 CONDRST 位置 1，而 CONFIGLOCK 保持为 0 时，才允许写入该位。如果 CONFIGLOCK = 1，则忽略对该位的写操作。

位 4 保留，必须保持复位值。

位 3 **IE**：中断使能 (Interrupt enable)

0：禁止 RNG 中断

1：使能 RNG 中断。只要 RNG_SR 寄存器中的 DRDY = 1、SEIS = 1 或 CEIS = 1，就会挂起中断。

位 2 **RNGEN**：真随机数发生器使能 (True random number generator enable)

0：禁止真随机数发生器。逻辑噪声源关闭，并会对由 RNG 时钟提供时钟信号的逻辑系统进行门控。

1：使能真随机数发生器。

位 1:0 保留，必须保持复位值。

24.7.2 RNG 状态寄存器 (RNG_SR)

RNG status register

偏移地址：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SEIS	CEIS	Res.	Res.	SECS	CECS	DRDY
									rc_w0	rc_w0			r	r	r

位 31:7 保留，必须保持复位值。

位 6 **SEIS**：种子错误中断状态 (Seed error interrupt status)

该位与 SECS 同时置 1，通过写入 0 来清零（除非使用 CONDRST）。写入 1 无效。

0：未检测到错误序列

1：至少检测到一个错误序列。有关详细信息，请参见 SECS 位说明。

如果 RNG_CR 寄存器中 IE = 1，则会挂起中断。

位 5 **CEIS**：时钟错误中断状态 (Clock error interrupt status)

该位与 CECS 同时置 1，通过写入 0 来清零。写入 1 无效。

0：RNG 时钟正确 ($f_{RNGCLK} > f_{HCLK}/32$)

1：检测到 RNG 时钟在内部分频前过慢 ($f_{RNGCLK} < f_{HCLK}/32$)

如果 RNG_CR 寄存器中 IE = 1，则会挂起中断。

位 4:3 保留，必须保持复位值。

位 2 **SECS**：种子错误当前状态 (Seed error current status)

0：目前未检测到错误序列。如果 SEIS 位置 1，则意味着已检测到错误序列并已恢复正常。

1：至少检测到一个以下错误序列：

– 运行时重复计数测试失败（噪声源持续不变地输出了 24 位或以上的 0 或 1，或者 32 个或以上的 01 或 10）。

– 噪声源启动测试或连续自适应比例测试失败。

– 启动后期处理/调节完整性检查失败。

位 1 **CECS**：时钟错误当前状态 (Clock error current status)

0：RNG 时钟正确 ($f_{RNGCLK} > f_{HCLK}/32$)。如果 CEIS 位置 1，则意味着已检测到时钟过慢并已恢复正常。

1：RNG 时钟过慢 ($f_{RNGCLK} < f_{HCLK}/32$)。

注意：只有 RNG_CR 寄存器中的 CED 位置 0 时，CECS 位才有效。

位 0 **DRDY**：数据就绪 (Data ready)

0：RNG_DR 寄存器尚未有效，无可用随机数据。

1：RNG_DR 寄存器包含有效随机数据。

在输出缓冲区变空之后（读取 RNG_DR 寄存器之后），在生成新的随机值之前，该位返回 0。

注意：当禁止外设时（RNG_CR 寄存器中的 RNGEN = 0），DRDY 位可能会被置 1。

如果 RNG_CR 寄存器中的 IE = 1，则在 DRDY = 1 时生成中断。

24.7.3 RNG 数据寄存器 (RNG_DR)

RNG data register

偏移地址：0x008

复位值：0x0000 0000

RNG_DR 寄存器是只读寄存器，在读取时提供 32 位随机数值。当 DRDY = 1 并且该值不为 0x0 时，即使 RNGEN = 0，该寄存器的内容也有效。

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

位 31:0 **RNDATA[31:0]**：随机数据 (Random data)

DRDY = 1 时有效的 32 位随机数据。DRDY = 0 时，RNDATA 值为 0。

当 DRDY 置 1 时，建议始终验证 RNG_DR 是否不为零。因为如果不为零，在轮询 RNG_SR 与读取 RND_DR 输出之间可能会发生种子错误（罕见事件）。

24.7.4 RNG 运行状态测试控制寄存器 (RNG_HTCR)

RNG health test control register

偏移地址：0x010

复位值：0x0000 72AC

仅当 RNG_CR 中的 CONDRST 位置 1 且 CONFIGLOCK 位清零时，才允许写入 RNG_HTCR。如果 CONFIGLOCK = 1，则忽略对该寄存器的写操作。

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

位 31:0 **HTCFG[31:0]**：运行状态测试配置 (health test configuration)

RNG 使用该配置来配置运行状态测试。有关建议值，请参见*第 24.6 节：RNG 熵源验证*。

注意：如果写入的值与建议值不同，则无法保证 RNG 行为（包括对该寄存器的读操作）。

24.7.5 RNG 寄存器映射

表 155. RNG 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	RNG_CR	CONFIGLOCK	CONDRST	Res.	Res.	Res.	Res.	RNG_CONFIG1[5:0]						CLKDIV [3:0]				RNG_ CONFIG2 [2:0]			NISTC	RNG_ CONFIG3 [3:0]				ARDIS	Res.	CED	Res.	IE	RNGEN	Res.	Res.
	Reset value	0	0					0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0		0		0	0		
0x004	RNG_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SEIS	CEIS	Res.	Res.	SECS	CECS	DRDY
	Reset value																										0	0			0	0	0
0x008	RNG_DR	RNDATA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x010	RNG_HTCR	HTCFG[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1	0	1	0	1	1	0	0

有关寄存器边界地址的信息，请参见*第 2.2 节*。

25 哈希处理器 (HASH)

25.1 简介

哈希处理器完全兼容安全哈希算法（SHA-1、SHA-224 和 SHA-256）和 HMAC（密钥哈希消息认证码）算法。HMAC 适用于需要进行消息验证的应用程序。

哈希处理器可以为任何长度小于 2^{64} 位的消息计算经 FIPS（联邦信息处理标准）认可的 160、224 和 256 位长度的摘要。

25.2 HASH 主要特性

- 适合于数据验证应用，符合以下标准：
  - 联邦信息处理标准出版物 FIPS PUB 180-4 *安全哈希标准*（SHA-1 和 SHA-2 系列）
  - 联邦信息处理标准出版物 FIPS PUB 186-4 *数字签名标准 (DSS)*
  - 互联网工程任务组 (IETF) 征求意见 RFC 2104 *HMAC：密钥哈希消息认证*和联邦信息处理标准出版物 FIPS PUB 198-1 *密钥哈希消息认证码 (HMAC)*
- 快速计算 SHA-1、SHA-224 和 SHA-256
  - 使用 SHA-1（或 SHA-256）算法处理一个 512 位数据块需要 82（或 66）个时钟周期
- 支持 HMAC 模式以及所有支持的算法
- 连续消息块中摘要的对应 32 位字添加到彼此之中，以构成整个消息的摘要
  - 可自动进行 32 位字交换，以兼容输入位串的内部小端模式表示法
  - 支持的字交换格式：位、字节、半字和 32 位字
- 单个 32 位只写输入寄存器，与内部输入 FIFO 相关联，对应于一个 64 字节块的大小（16 x 32 位）
- 可自动填充来完成输入位串，从而适应摘要最小块大小
- AHB 从外设，仅支持 32 位字访问（否则会产生 AHB 错误）
- 8×32 位字（H0 到 H7）用于输出消息摘要
- 数据流自动控制，支持使用一个通道进行直接存储器访问 (DMA)。
- 支持 单次 DMA 传输和固定四字 DMA 突发传输。
- 每块（512 位）可中断一次消息摘要计算
  - 可重载摘要寄存器
  - 哈希计算挂起/恢复机制，包括 DMA

25.3 HASH 实现

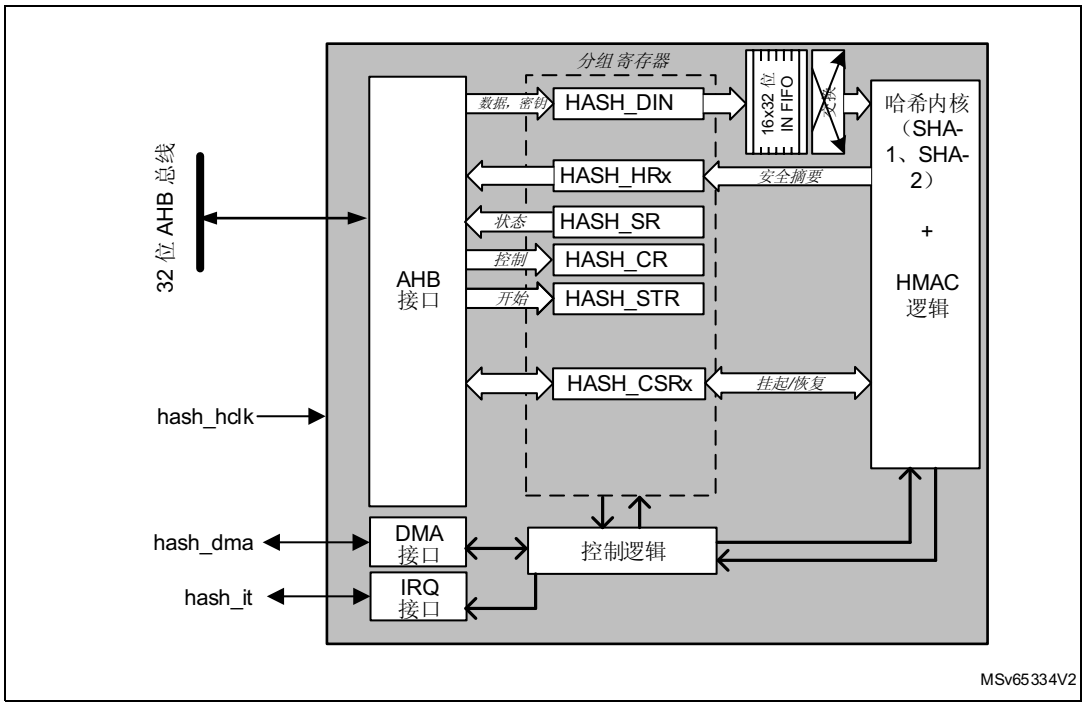
该器件具有单个 HASH 外设实例。

25.4 HASH 功能描述

25.4.1 HASH 框图

图 158 显示了哈希处理器的框图。

图 158. HASH 框图



25.4.2 HASH 内部信号

表 156 列出了 HASH 级而非产品级（焊盘上）所提供的内部信号，对这些信号有所了解会很有用。

表 156. HASH 内部输入/输出信号

信号名称	信号类型	说明
hash_hclk	数字输入	AHB 总线时钟
hash_it	数字输出	哈希处理器全局中断请求
hash_dma	数字输入/输出	DMA 突发请求/确认

25.4.3 关于安全哈希算法

哈希处理器完全兼容由 FIPS PUB 180-4 标准定义的安全哈希算法。

对于每个算法，HASH 计算消息或数据文件的压缩表示。更具体来讲，当输入中出现消息时，HASH 处理内核将生成一个固定长度的输出位串，称为消息摘要（请参见*表 157*）。

表 157. 所支持哈希算法的相关信息

算法	消息摘要大小（位）	数据块大小（字节）	消息长度	位串消息
SHA-1	160	64	$< 2^{64}$ 位	是
SHA-224	224			
SHA-256	256			

然后可以通过数字签名算法来处理此消息摘要，以便生成或验证消息的签名。

对消息摘要而不是对消息签名通常可提高流程的效率，因为消息摘要通常比消息要小得多。数字签名的验证程序和数字签名的创建程序必须使用相同哈希算法。

哈希处理器支持的 SHA-2 功能由 NIST 评定为“安全可靠”，因为要找出与某个给定消息摘要对应的消息，或找出两个生成相同消息摘要的不同消息（自 2017 年 2 月起 SHA-1 不再安全可靠），在计算层面无法实现。对传输中的消息进行任何更改都极有可能产生不同的消息摘要，从而导致签名验证失败。

25.4.4 消息数据输送

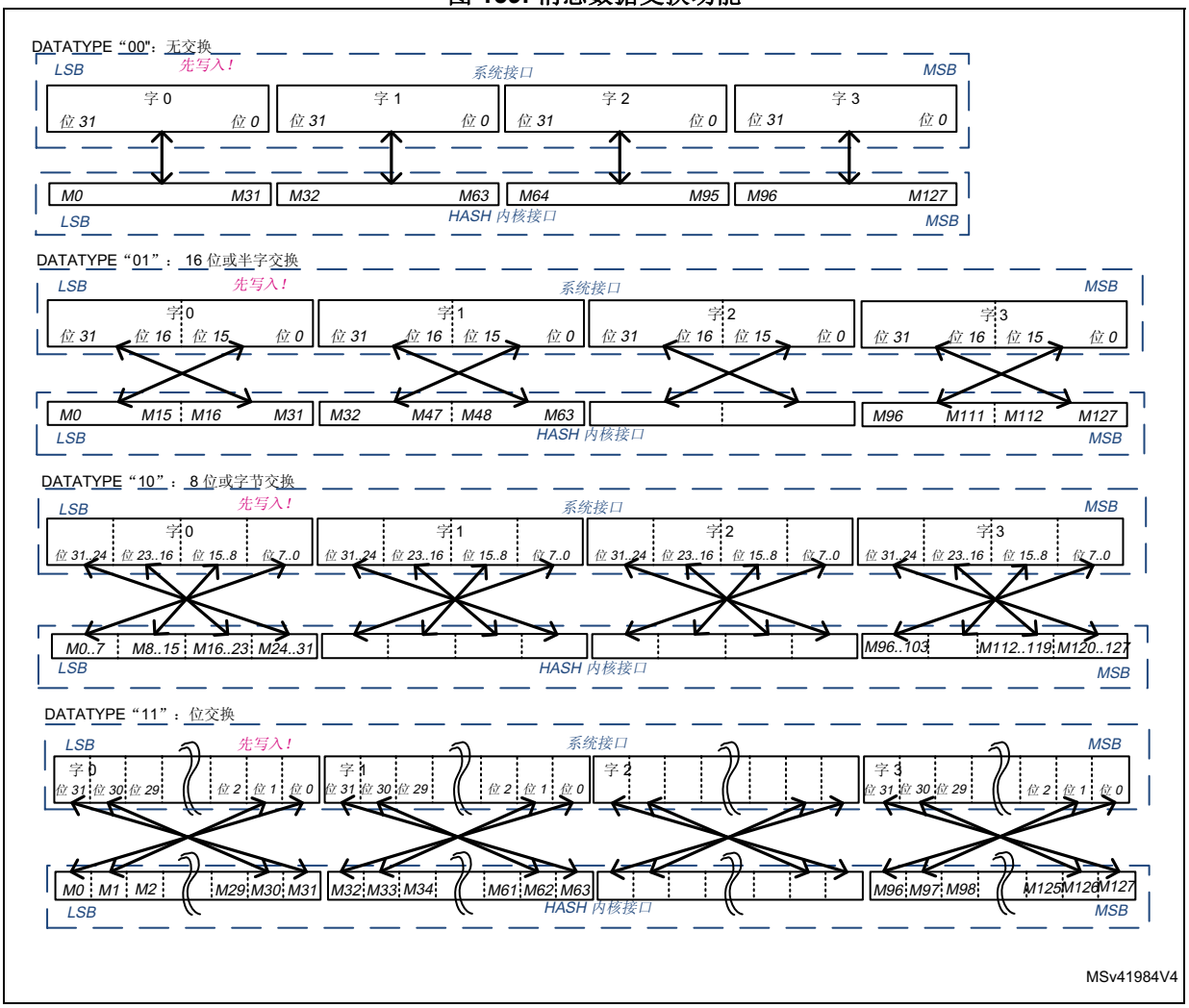
要由 HASH 处理的消息（或数据文件）必须视为位串。根据 FIPS PUB 180-4 标准，该消息位串从左到右增长（十六进制字采用“大端”约定来表示），从而使得在每个字内，最高有效位存储在最左侧位的位置上。以位串表示为“**01100001 01100010 01100011**”的消息串“abc”为例，其 32 位字表示为 **0x00636261**，8 位字表示为 **0x61626300**。

通过将数据写入到 HASH_DIN 寄存器，数据将一次性输入到 HASH 的一个 32 位字中。每次在寄存器中写入新数据时，HASH_DIN 寄存器的当前内容都将传输到 16 字输入 FIFO。因此，HASH_DIN 和 FIFO 构成了一个长度为十七个 32 位字的 FIFO（名为 IN 缓冲器）。

根据要处理的数据类型（例如，当数据是 ASCII 文本流时的字节交换），必须先对从输入 FIFO 读取的数据进行位、字节、半字或无交换操作，然后再将其输入小端哈希处理内核。*图 159* 显示了如何根据 HASH 控制寄存器 (HASH_CR) 中的 DATATYPE 位域，使用由驱动程序弹出到输入 FIFO 中的一个 32 位字构建哈希处理内核 32 位数据块 M0...31。

当禁止位交换时 (DATATYPE = 00)，HASH_DIN 数据字节序如下所述：消息的最低有效位必须在输入到哈希处理器的第一个字中的 MSB 位置，位串的第 32 位必须在输入到哈希处理器的第二个字中的 MSB 位置，依此类推。

图 159. 消息数据交换功能



25.4.5 消息摘要计算

当计算消息摘要时，哈希处理器会按顺序处理多个块。有关块大小的详细信息，可参见*表 157：所支持哈希算法的相关信息*。

DMA 或 CPU 每次将块写入到哈希处理器时，HASH 都会自动开始计算消息摘要。此操作称为部分摘要计算。

如*第 25.4.4 节：消息数据输送*中所述，待处理的消息将一次性输入到 HASH 32 位字中，从而写入 HASH_DIN 寄存器以填充输入 FIFO。应用程序必须使用以下序列对该数据执行哈希计算。

1. 使用 HASH_CR 寄存器初始化哈希处理器：
   – 使用 ALGO[1:0] 位域选择正确的算法。如果需要，可使用 DATATYPE[1:0] 位域为消息输入字编程正确的交换操作。
   – 当需要使用 HMAC 模式时，如果 HMAC 密钥大小大于算法已知的块大小，则将 MODE 位以及 LKEY 位置 1（否则将 LKEY 位保持清零状态）。有关详细信息，请参见*第 25.4.7 节：HMAC 运算*。
   – 如果消息的最后一个字中的有效位数不是 32 位，则更新 HASH_STR 寄存器中的 NBLW[4:0] 以定义最后一个字中的有效位数。NBLW 信息用于在最终消息摘要计算之前正确执行自动消息填充操作。
2. 通过将 HASH_CR 寄存器中的 INIT 位置 1 来完成初始化。如果通过 DMA 传输数据，也将 DMAE 位置 1。

小心： 编程步骤 2 时，务必事先或同时设置正确的配置值（ALGO、DATATYPE、HMAC 模式、密钥长度、NBLW）。

3. 开始通过写入 HASH_DIN 寄存器来填充数据（自动通过 DMA 传输数据的情况除外）。请注意，只有当块的最后一个值进入输入 FIFO 后，才能开始处理块。部分或最终摘要计算的管理方式取决于数据输送到处理器的方式：
   – 通过软件填充数据：

     每次应用程序写入下一个块的第一个字时，都会触发部分摘要计算，块大小由 HASH_SR 中的 NBWE 位定义。一旦处理器再次准备就绪（HASH_SR 中的 DINIS = 1），软件即可将新数据写入 HASH_DIN。这种机制可防止 HASH 引入等待状态。

     当输入最后一个块并且软件将 DCAL 位置 1 时，将触发最终摘要计算。如果消息长度不是块大小的整数倍，则必须在写入 DCAL 位之前写入 HASH_STR 寄存器中的 NBLW 位域（有关详细信息，请参见*第 25.4.6 节*）。
   – 通过单次 DMA 传输 (MDMAT = 0) 填充数据：

     每次 FIFO 已满时，都将自动触发部分摘要计算。

     当最后一个块通过 DMA 传输到 HASH_DIN 寄存器后，将自动触发最终摘要计算（硬件将 DCAL 位置 1）。如果消息长度不是块大小的整数倍，则必须在使能 DMA 之前写入 HASH_STR 寄存器中的 NBLW 位域（有关详细信息，请参见*第 25.4.6 节*）。
   – 使用多次 DMA 传输 (MDMAT = 1) 填充数据：

     可针对单次 DMA 传输触发部分摘要计算（请参见上述说明）。但是，当最后一个块通过 DMA 传输到 HASH_DIN 寄存器后，不会自动触发最终摘要计算（硬件不将 DCAL 位置 1）。这使得哈希处理器能够接收新的 DMA 传输（作为此摘要计算的一部分）。为了启动最终摘要计算，软件必须在最后一次 DMA 传输之前将 MDMAT 位清零，以便如同针对单次 DMA 传输一样触发最终摘要计算。
4. 摘要计算完成 (DCIS = 1) 后，可以从输出寄存器读取生成的摘要，如*表 158* 中所述。

表 158. 哈希处理器输出

算法	有效输出寄存器	最高有效位	摘要大小（位）
SHA-1	HASH_H0 到 HASH_H4	HASH_H0[31]	160
SHA-224	HASH_H0 到 HASH_H6	HASH_H0[31]	224
SHA-256	HASH_H0 到 HASH_H7		256

关于 HMAC 详细指令的更多信息，请参见*第 25.4.7 节：HMAC 运算*。

25.4.6 消息填充

概述

当计算消息的压缩表示时，循环执行将数据输送到哈希处理器（每个块大小传输执行一次自动部分摘要计算）的过程，直到原始消息的最后几位被写入 HASH_DIN 寄存器。

因为消息的长度（位数）可以是任何整数值，因此写入到哈希处理器的最后一个字的有效位数在 1 到 32 之间。这个最后一个字的有效位数必须写到 HASH_STR 寄存器的 NBLW 位域中，以便消息填充能够在最终消息摘要计算之前得到正确处理。

填充处理

*第 25.4.5 节：消息摘要计算*中介绍了使能或禁止 DMA 时的详细填充序列。

填充示例

根据联邦信息处理标准 PUB 180-4 规定，消息填充包括添加一个“1”，后跟 *k* 个“0”以及一个 64 位整数（长度与消息长度 *L*（位数）相等）。经过这三个填充操作后生成的消息长度为 *L* + 1 + *k* + 64（512 位的整数倍）。

对于哈希处理器，“1”将添加到 HASH_DIN 寄存器中写入的最后一个字，位位置由 NBLW 位域定义，而其余更高的位将被清零（“0”）。

FIPS PUB180-4 提供的示例

我们假定原始消息是采用 ASCII 二进制编码格式的“abc”，长度 L=24：

```
字节 0    字节 1   字节 2   字节 3
01100001 01100010 01100011 uuuuuuuu
<-- 写入 HASH_DIN 的第一个字 -->
```

NBLW 必须以值 24 加载：“**1**”在位串中的位 24 处追加（在以上位串中从左到右开始计数），这对应于 HASH_DIN 寄存器中的位 31（小端模式约定）：

```
01100001 01100010 01100011 1uuuuuuu
```

由于 L = 24，因此以上位串中的位数是 25，并将追加 423 个“0”位，现在为 448 位。

这将生成十六进制（大端模式的字节字）：

```
61626380 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000
```

将追加双字格式的消息长度值 **L**，即 00000000 00000018。因此，最后填充的消息为十六进制（大端模式的字节字）：

```
61626380 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000018
```

如果哈希处理器被编程为在 HASH_DIN 输入寄存器（HASH_CR 中的 DATATYPE = 10）中交换字节，则必须使用以下序列输入上述消息：

1. 将 **0xUU636261** 写入到 HASH_DIN 寄存器（其中的“**U**”意味着无关）
2. 将 **0x18** 写入到 HASH_STR 寄存器（写入到 HASH_DIN 寄存器的最后一个字中的有效位数是 24，因为原始消息长度为 24 位）
3. 将 **0x10** 写入到 HASH_STR 寄存器以启动消息填充（如上文所述）以及随后执行摘要计算
4. 完成哈希计算后，HASH_HRx 寄存器 (x = 0...4) 中的消息摘要可供 SHA-1 算法使用。对于此 FIPS 示例，预期值如下：

```
HASH_HR0 = 0xA9993E36
HASH_HR1 = 0x4706816A
HASH_HR2 = 0xBA3E2571
HASH_HR3 = 0x7850C26C
HASH_HR4 = 0x9CD0D89D
```

25.4.7 HMAC 运算

概述

根据互联网工程任务组 RFC2104 和 NIST FIPS PUB 198-1 的规定，HMAC 算法以不可逆方式将正在处理的消息与用户所选的密钥进行绑定，从而用于消息验证。该算法由两个嵌套的哈希运算组成：

```
HMAC(message)= Hash((Key | pad) XOR opad |
                    Hash((Key | pad) XOR ipad | message))
```

其中：

- **Opad** = $[0x5C]_n$（外 pad），**ipad** = $[0x36]_n$（内 pad）。
- $[X]_n$ 表示 X 重复 *n* 次，其中的 *n* 等于底层哈希函数数据块的字节大小（对于 512 位块大小，n = 64）。
- **pad** 是将密钥扩展到上面定义的长度 *n* 所需的零序列。如果密钥长度大于 *n*，应用程序必须首先使用 Hash() 函数对密钥进行哈希处理，然后将得到的字节串用作 HMAC 的实际密钥。
- | 表示串联运算符。

注意： *哈希处理器的 HMAC 模式可与所有支持的算法一起使用。*

HMAC 处理

计算 HMAC 需要下列四个步骤：

1. 软件将 INIT 位置 1，同时将 MODE 位置 1，并使用 ALGO 位选择所需的算法。如果正在使用的密钥的长度超过 64 字节，则还必须将 LKEY 位置 1。在这种情况下，根据 HMAC 规范的要求，哈希处理器将使用密钥的哈希代替真实密钥。
2. 软件提供用于内部哈希函数的密钥，该操作与消息字符串加载操作遵循相同的机制：将密钥数据写入 HASH_DIN 寄存器，然后通过将 DCAL 位置 1 以及将正确的 NBLW 写入 HASH_STR 寄存器来完成传输。
3. 一旦处理器再次准备就绪（HASH_SR 中的 DINIS = 1），软件即可将消息字符串写入 HASH_DIN。当输入最后一个块的最后一个字并且软件将 HASH_STR 寄存器中的 DCAL 位置 1 时，如果消息长度不是块大小的整数倍，则必须同时将 NBLW 位域编程为非零值。请注意，DMA 也可用于输送消息字符串，如*第 25.4.5 节：消息摘要计算*中所述。

4. 一旦处理器再次准备就绪（HASH_SR 中的 DINIS = 1），软件就会提供用于外部哈希函数的密钥，也就是将密钥数据写入 HASH_DIN 寄存器，然后通过将 DCAL 位置 1 以及将正确的 NBLW 编程到 HASH_STR 寄存器来完成传输。DCIS 位置 1 后，HMAC 结果就会存储到有效输出寄存器 (HASH_HRx) 中。

注意： *HMAC 本原的计算延迟取决于密钥和消息的长度，如第 25.6 节：HASH 处理时间中所述。*

有关字节序管理的详细信息，可参见第 25.4.4 节：消息数据输送。

HMAC 示例

下面是一个由 NIST 指定的 HMAC SHA-1 算法示例（HASH_CR 中的 ALGO = 00 且 MODE = 1）。SHA-1 块大小为 64 字节。

假定原始消息是采用 ASCII 二进制编码格式的“**Sample message for keylen = blocklen**”（keylen = blocklen 时的示例消息），长度 L = 34 字节。如果 HASH 编程为无交换模式（HASH_CR 中的 DATATYPE = 00），则必须将以下数据按顺序加载至 HASH_DIN 寄存器：

1. 由 NIST 指定的**内部哈希密钥**输入（长度 = 64，即不填充）。由于密钥长度 = 64，HASH_CR 寄存器中的 LKEY 位清零

```
00010203 04050607 08090A0B 0C0D0E0F 10111213 14151617
18191A1B 1C1D1E1F 20212223 24252627 28292A2B 2C2D2E2F
30313233 34353637 38393A3B 3C3D3E3F
```

2. **消息**输入（长度 = 34，即需要填充）。必须将 HASH_STR 设置为 **0x20**，以开始消息填充和内部哈希计算（将“**U**”视为无关）

```
53616D70 6C65206D 65737361 67652066 6F72206B 65796C65
6E3D626C 6F636B6C 656EUUUU
```

3. **外部哈希密钥**输入（长度 = 64，即不填充）。在此输入与内部哈希密钥相同的密钥。
4. **最终外部哈希计算**由 HASH 执行。HMAC-SHA1 摘要结果存储于 HASH_HRx 寄存器中，如下所示：

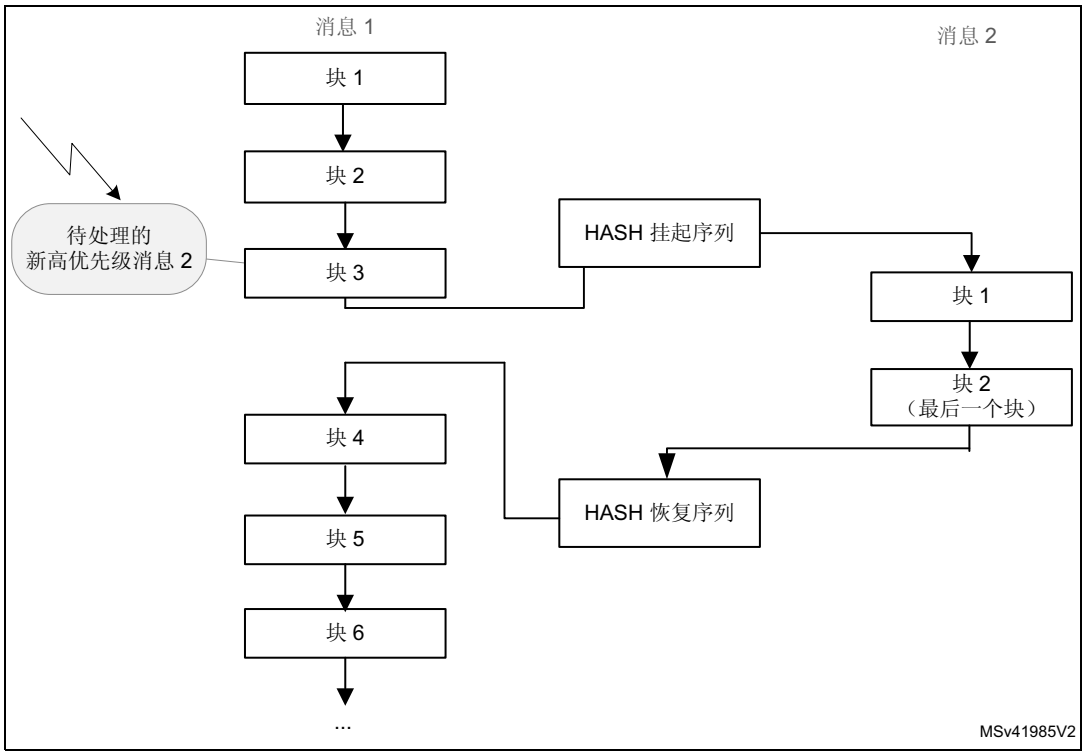
```
HASH_HR0 = 0x5FD596EE
HASH_HR1 = 0x78D5553C
HASH_HR2 = 0x8FF4E72D
HASH_HR3 = 0x266DFD19
HASH_HR4 = 0x2366DA29
```

25.4.8 HASH 挂起/恢复操作

概述

可以中断哈希/HMAC 操作以执行另一个优先级较高的处理。处理完优先级较高的任务后，可以继续完成之前中断的过程，如*图 160* 所示。

图 160. HASH 挂起/恢复机制



要这样做，必须将被中断的任务的上下文从 HASH 寄存器保存到存储器，然后从存储器恢复到 HASH 寄存器。

以下说明由软件或 DMA 控制数据流的过程。

通过软件加载数据

如果不使用 DMA 将消息加载到哈希处理器，则只有在当前未处理任何块时才能保存上下文。

要挂起消息处理，请在写入 NBWE 中定义的字数后执行以下操作：

1. 在轮询模式下，等待 BUSY = 0，然后轮询 DINIS 状态位是否置 1。
   在中断模式下，通过 DINIS 中断处理程序执行下一步（建议使用）。
2. 将以下寄存器的内容存储到存储器中：
   - HASH_IMR
   - HASH_STR
   - HASH_CR
   - HASH_CSR0 到 HASH_CSR37，以及 HASH_CSR38 到 HASH_CSR53（如果 HMAC 操作正在进行）

要恢复消息处理，请执行以下操作：

1. 将存储器中保存的值写入以下寄存器中：HASH_IMR、HASH_STR 和 HASH_CR。
2. 通过将 HASH_CR 寄存器中的 INIT 位置 1 来初始化哈希处理器
3. 将存储器中保存的值写入 HASH_CSRx 寄存器。
4. 从之前的中断点重新开始处理。

通过 DMA 加载数据

如果使用 DMA 将消息加载到哈希处理器中，建议先挂起再恢复安全摘要计算，如下所述。

在此序列中，分配给哈希外设的 DMA 通道仍会分配用于处理消息 1（请参见*图 160*）。

要挂起使用 DMA 进行的消息处理，请执行以下操作：

1. 将 DMAE 位清零以禁止 DMA 接口。哈希外设通过 DMA 自动获取足够的数据来完成当前的突发传输。
2. 等待至最后一次 DMA 传输完成（HASH_SR 中的 DMAS = 0）。
3. 禁止 DMA 通道。
4. 在轮询模式或中断模式（建议使用）下，等待至哈希处理器就绪（未在处理任何块），即等待 HASH_SR 中的 DINIS = 1 如果 HASH_SR 中的 DCIS 也置 1，则哈希结果可用，无需进行上下文交换操作。否则转至步骤 5。
5. 将 HASH_IMR、HASH_STR、HASH_CR 和 HASH_CSR0 保存到 HASH_CSR37 寄存器。如果 HMAC 操作正在进行，则还必须将 HASH_CSR38 保存到 HASH_CSR53 寄存器。

要恢复使用 DMA 进行的消息处理，请执行以下操作：

1. 重新配置 DMA 控制器，以便其在没有再次被中断的情况下继续传输消息直到结束。
2. 将存储器中保存的值编程到 HASH_IMR、HASH_STR 和 HASH_CR 寄存器。
3. 通过将 HASH_CR 寄存器中的 INIT 位置 1 来初始化哈希处理器。
4. 将存储器中保存的值编程到 HASH_CSRx 寄存器。
5. 通过将 DMAE 位置 1，从之前的中断点重新开始处理。

25.4.9 HASH DMA 接口

HASH 支持单次 DMA 传输和固定四字 DMA 突发传输。

哈希处理器可使用一个接口连接 DMA 控制器。通过将 HASH_CR 寄存器中的 DMAE 位置 1，可以使用此 DMA 向 HASH 中写入数据。如果该位置 1，HASH 每次需要向 HASH_DIN 寄存器写入块时，都会发出 DMA 请求。

接收到四个 32 位字后，HASH 会自动触发新的 DMA 请求。有关详细信息，请参见*第 25.4.5 节：消息摘要计算*。

在开始 DMA 传输之前，软件必须对将要复制到 HASH_DIN 寄存器的最后一个字中的有效位数进行编程。这可通过向 HASH_STR 寄存器中写入以下值来完成：

NBLW = Len(Message)% 32，其中“**x%32**”表示 x 除以 32 的余数。

HASH_SR 寄存器的 DMAS 位提供有关 DMA 接口活动的信息。该位与 DMAE 一起置 1，并且在 DMAE 清零且未在进行任何 DMA 传输时清零。

注意： *没有任何中断与 DMAS 位相关联。*

MDMAT 置 1 时，传输的大小必须是四个字的倍数。

25.4.10 HASH 错误管理

哈希处理器不生成任何错误标志。

25.5 HASH 中断

哈希处理器可生成两个单独的可屏蔽中断源，用以通知发生以下事件：

- 摘要计算完成 (DCIS)
- 数据输入缓冲器就绪 (DINIS)

这两个中断源连接到同一全局中断请求信号 (hash_it)，该信号又连接到器件中断控制器。通过更改 HASH_IMR 寄存器中的屏蔽位，可以单独使能或禁止各个中断源。将相应的屏蔽位置 1 可使能中断。

可以从 HASH_SR 寄存器中读取各个可屏蔽中断源的状态。*表 159* 对可用功能进行了总结。

表 159. HASH 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法
HASH	摘要计算完成	DCIS	DCIE	将 DCIS 清零或将 INIT 置 1
	数据输入缓冲器准备好获取新的块	DINIS	DINIE	将 DINIS 清零或写入 HASH_DIN

25.6 HASH 处理时间

表 160 汇总了每种工作模式下处理中间块需要的时间。

表 160. 处理时间（时钟周期数）

工作模式	块大小（以字节为单位）	FIFO 加载(1)	计算阶段	总计
SHA-1	64	16	66	**82**
SHA-224		16	50	**66**
SHA-256				

1. 包括向处理器中加载块所需的时间。

处理消息的最后一个块（或者 HMAC 中某个密钥的最后一个块）所需要的时间可能会更长。此时间取决于最后一个块的长度以及密钥的大小（在 HMAC 模式下）。

与处理中间块相比，此时间可能按以下系数增加：

- 对于哈希消息，该系数为 **1 到 2.5**
- 对于 HMAC 输入密钥，该系数在 **2.5** 左右
- 对于 HMAC 消息，该系数为 **1 到 2.5**
- 对于 HMAC 输出密钥，如果是短密钥，则该系数在 **2.5** 左右
- 对于 HMAC 输出密钥，如果是长密钥，则该系数为 **3.5 到 5**

25.7 HASH 寄存器

哈希内核与多个控制和状态寄存器以及多个消息摘要寄存器相关联。所有这些寄存器都只能通过 32 位字来访问，否则将生成 AHB 错误。

25.7.1 HASH 控制寄存器 (HASH_CR)

HASH control register

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ALGO[1:0]		LKEY
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	MDMAT	DINNE	NBW[3:0]				Res.	MODE	DATATYPE[1:0]		DMAE	INIT	Res.	Res.
		rw	r	r	r	r	r		rw	rw	rw	rw	rw		

位 31:19 保留，必须保持复位值。

位 18:17 **ALGO[1:0]**：算法选择 (Algorithm selection)

这些位用于选择哈希算法：

00：SHA-1

01：保留

10：SHA-224

11：SHA-256

仅当 INIT 位置 1 时，此选择才有效。在计算期间更改该位域不起作用。

在 ALGO 位域更新并且 INIT 位置 1 后，HASH_SR 中的 NBWE 会自动更新为 0x11。

位 16 **LKEY**：长密钥选择 (Long key selection)

HMAC 密钥大小大于块大小（64 字节）时，应用程序必须将该位置 1

0：HMAC 密钥大小小于或等于块大小（短密钥）。在 HMAC 计算过程中，使用 HASH_DIN 中写入的实际密钥值。

1：HMAC 密钥大小大于块大小（长密钥）。在 HMAC 计算过程中，使用密钥的哈希代替真实密钥。

仅当 INIT 位置 1 并且 MODE 位置 1（选择 HMAC 模式）时，此选择才有效。在计算期间更改该位不起作用。

位 15 保留，必须保持复位值。

位 14 保留，必须保持复位值。

位 13 **MDMAT**：多个 DMA 传输 (Multiple DMA transfers)

如果在需要多个 DMA 传输时对大型文件进行哈希运算，则将该位置 1。

0：在 DMA 传输结束时自动将 DCAL 置 1。

1：在 DMA 传输结束时不自动将 DCAL 置 1。

位 12 **DINNE**：DIN 非空 (DIN not empty)

有关 DINNE 位的说明，请参见 HASH_SR 的 DINNE 位。

该位为只读。

位 11:8 **NBW[3:0]**：已推入的字数 (Number of words already pushed)

有关 NBW[3:0] 位域的说明，请参见 HASH_SR 的 NBWP[3:0] 位域。

该位为只读。

位 7 保留，必须保持复位值。

位 6 **MODE**：模式选择 (Mode selection)

该位针对所选算法选择正常或密钥 HMAC 模式：

0：选择哈希模式

1：选择 HMAC 模式。如果正在使用的密钥的长度超过 64 字节，则必须将 LKEY 位置 1。

仅当 INIT 位置 1 时，此选择才有效。在计算期间更改该位不起作用。

位 5:4 **DATATYPE[1:0]**：数据类型选择 (Data type selection)

该位域用于定义在 HASH_DIN 寄存器中输入的数据格式：

00：32 位数据。写入到 HASH_DIN 的数据将由哈希处理直接使用，不会重新排序。

01：16 位数据或半字。写入到 HASH_DIN 的数据被视为两个半字，并且在交换之后由 HASH 处理使用。

10：8 位数据或字节。写入到 HASH_DIN 的数据被视为四个字节，并且在交换之后由 HASH 处理使用。

11：位数据或位串。写入到 HASH_DIN 的数据被视为 32 位（字符串的第一位处于位置 0），并且在交换之后由 HASH 处理使用（字符串的第一位处于位置 31）。

位 3 **DMAE**：DMA 使能 (DMA enable)

0：禁止 DMA 传输

1：使能 DMA 传输。哈希内核可以接收数据时，便发送 DMA 请求。

该位置 1 后，将在消息的最后一个数据被写入哈希处理器时由硬件清零。

当正在进行 DMA 传输时，将该位清零不会中止当前的传输。HASH 的 DMA 接口在内部继续保持使能状态，直到传输完成或 INIT 置 1。

将 INIT 位置 1 不会清零 DMAE 位。

位 2 **INIT**：初始化消息摘要计算 (Initialize message digest calculation)

将该位置 1 将会复位哈希处理器内核，以使 HASH 可以计算新消息的消息摘要。

将该位清零不会产生影响。读取该位时，始终返回 0。

位 1:0 保留，必须保持复位值。

25.7.2 HASH 数据输入寄存器 (HASH_DIN)

HASH data input register

偏移地址：0x04

复位值：0x0000 0000

HASH_DIN 是数据输入寄存器。其宽度为 32 位。该寄存器用于以 64 字节的块来输入消息。

编程 HASH_DIN 寄存器时，在 AHB 数据总线上呈现的值将被“推入”哈希内核，并且寄存器获取 AHB 数据总线上呈现的新值。要获得正确的消息格式，必须先对 HASH_CR 寄存器中的 DATATYPE 位进行配置。

完整的块写入到 HASH_DIN 寄存器后，将启动中间摘要计算。有两种启动方式：

- 通过将下一个块的第一个数据写入 HASH_DIN 寄存器来启动（未使用 DMA 时）
- 自动启动（使用 DMA 时）

最后一个块写入到 HASH_DIN 寄存器之后，可通过将 HASH_STR 寄存器中的 DCAL 位置 1（最终摘要计算）来启动最终摘要计算（包括填充）。如果使用 DMA 并且 MDMAT 位清零，则会自动执行该操作。

读取 HASH_DIN 寄存器将返回零。

注意： *当 HASH 繁忙时，如果摘要计算（中间或最终）未完成，对 HASH_DIN 寄存器进行写访问可能会导致 AHB 总线阻塞。*

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

位 31:0 **DATAIN[31:0]**：数据输入 (Data input)

写入该寄存器会将当前寄存器内容推入到 FIFO，并且寄存器获取在 AHB 数据总线上呈现的新值。

读取该寄存器将返回零。

25.7.3 HASH 启动寄存器 (HASH_STR)

HASH start register

偏移地址：0x08

复位值：0x0000 0000

HASH_STR 寄存器有两个功能：

- 用于定义哈希处理器中输入的消息的最后一个字的有效位数（也就是写入到 HASH_DIN 寄存器的最后一个数据中有效的最低有效位的数量）
- 用于通过将 DCAL 位置 1 来启动对消息中最后一个块的处理。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	DCAL	Res.	Res.	Res.	NBLW[4:0]				
							rw				rw	rw	rw	rw	rw

位 31:9 保留，必须保持复位值。

位 8 **DCAL**：摘要计算 (Digest calculation)

将该位置 1 将开始使用先前写入的 NBLW 值来填充消息，并且，由于 INIT 位已置 1，因此将使用写入到输入 FIFO 的所有数据字开始计算最终消息摘要。

读取该位将返回 0。

位 7:5 保留，必须保持复位值。

位 4:0 **NBLW[4:0]**：最后一个字中的有效位数 (Number of valid bits in the last word)

当消息位串的最后一个字被写入 HASH_DIN 寄存器并且经过内部数据交换后，哈希处理器会只接收如下指定的有效位：

0x00：写入的最后一个数据的全部 32 位均是有效消息位，即 M[31:0]

0x01：写入的最后一个数据中只有一位（在交换后）是有效消息位，即 M[0]

0x02：写入的最后一个数据中只有两位（在交换后）是有效消息位，即 M[1:0]

0x03：写入的最后一个数据中只有三位（在交换后）是有效消息位，即 M[2:0]

...

0x1F：写入的最后一个数据中只有 31 位（在交换后）是有效消息位，即 M[30:0]

上述机制仅在 DCAL = 0 时有效。如果在 DCAL 置 1 时写入 NBLW 位，则 NBLW 位域保持不变。换句话说，配置 NBLW 和设置 DCAL 不能同时进行。

读取 NBLW 位将返回写入到 NBLW 的最后一个值。

25.7.4 HASH 摘要寄存器

HASH digest registers

这些寄存器包含消息摘要结果，其定义如下：

- HASH_HR0、HASH_HR1、HASH_HR2、HASH_HR3 和 HASH_HR4 寄存器返回 SHA-1 摘要结果。
- HASH_HR0 到 HASH_HR6 寄存器返回 SHA-224 摘要结果。
- HASH_HR0 到 HASH_HR7 寄存器返回 SHA-256 摘要结果。

在所有情况下，摘要的最高有效位均存储在 HASH_HR0[31] 中，并且未使用的 HASH_HRx 寄存器读为零。

如果当哈希内核在计算中间摘要或最终消息摘要（DCIS 位等于 0）时，对这些寄存器之一进行读访问，则读取操作将返回零。

注意： *当启动新消息的摘要计算时（通过将 INIT 位置 1），HASH_HRx 寄存器将被强制设为其复位值。*

HASH_HR0 到 HASH_HR4 寄存器可以通过两个不同的地址进行访问（寄存器别名）。

HASH 别名摘要寄存器 x (HASH_HRAx)

HASH aliased digest register x

偏移地址：0x0C + x * 0x4，（x = 0 到 4）

复位值：0x0000 0000

HASH_HRAx 寄存器的内容与位于偏移地址 0x310 处的 HASH_HRx 寄存器之一的内容相同。

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

位 31:0 **Hx[31:0]**：哈希数据 x (Hash data x)

请参见*第 25.7.4 节：HASH 摘要寄存器*简介。

HASH 摘要寄存器 x (HASH_HRx)

HASH digest register x

偏移地址：0x310 + x * 0x4，（x = 0 到 4）

复位值：0x0000 0000

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

位 31:0 **Hx[31:0]**：哈希数据 x (Hash data x)

请参见*第 25.7.4 节：HASH 摘要寄存器*简介。

HASH 补充摘要寄存器 x (HASH_HRx)

HASH supplementary digest register x

偏移地址：0x310 + x * 0x4，（x = 5 到 7）

复位值：0x0000 0000

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

位 31:0 **Hx[31:0]**：哈希数据 x (Hash data x)

请参见*第 25.7.4 节：HASH 摘要寄存器*简介。

25.7.5 HASH 中断使能寄存器 (HASH_IMR)

HASH interrupt enable register

偏移地址：0x20

复位值：0x0000 0000

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

位 31:2 保留，必须保持复位值。

位 1 **DCIE**：摘要计算完成中断使能 (Digest calculation completion interrupt enable)

0：禁止摘要计算完成中断

1：使能摘要计算完成中断

位 0 **DINIE**：数据输入中断使能 (Data input interrupt enable)

0：禁止数据输入中断

1：使能数据输入中断

25.7.6 HASH 状态寄存器 (HASH_SR)

HASH status register

偏移地址：0x24

复位值：0x0011 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NBWE[4:0]				
											r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DINNE	Res.	NBWP[4:0]					Res.	Res.	Res.	Res.	Res.	BUSY	DMAS	DCIS	DINIS
r		r	r	r	r	r						r	r	rc_w0	rc_w0

位 31:21 保留，必须保持复位值。

位 20:16 **NBWE[4:0]**：预期的字数 (Number of words expected)

该位域用于反映消息中必须推入到 FIFO 以触发部分计算的字数。当对 HASH_DIN 寄存器进行写访问时，NBWE 递减 1。

如果 HASH_CR 中的 INIT 位置 1，则 NBWE 将设置为预期的块大小 + 1 (0x11)（以字为单位）。当部分摘要计算结束后，该位将被设置为预期的块大小 (0x10)。

位 15 **DINNE**：DIN 非空 (DIN not empty)

当 HASH_DIN 寄存器包含有效数据时（也就是在至少写入一次之后），该位将置 1。当 INIT 位（初始化）或 DCAL 位（上一个消息处理已完成）置 1 时，该位将清零。

0：数据输入缓冲器中不存在任何数据

1：输入缓冲器至少包含一个字的数据

位 14 保留，必须保持复位值。

位 13:9 **NBWP[4:0]**：已推入的字数 (Number of words already pushed)

该位域用于反映消息中已推入到 FIFO 的准确字数。当对 HASH_DIN 寄存器进行写访问时，NBWP 递增 1。

开始摘要计算后，NBWP 更新为 NBWP 块大小（以字为单位），并且当 INIT 位置 1 时，NBWP 将变为零。

位 8:4 保留，必须保持复位值。

位 3 **BUSY**：繁忙位 (Busy bit)

0：当前未在处理任何块

1：哈希内核正在处理某个数据块

位 2 **DMAS**：DMA 状态 (DMA Status)

该位提供有关 DMA 接口活动的信息。该位与 DMAE 一起置 1，并且在 DMAE = 0 且未在进行任何 DMA 传输时清零。没有任何中断与该位相关联。

0：禁止 DMA 接口 (DMAE = 0) 并且未在进行任何传输

1：使能 DMA 接口 (DMAE = 1) 或者某个传输在进行

位 1 **DCIS**：摘要计算完成中断状态 (Digest calculation completion interrupt status)

该位是在摘要就绪时（也就是整个消息已处理完毕时）由硬件设置。要将该位清零，可写入 0，或者将 HASH_CR 寄存器中的 INIT 位置 1。

0：HASH_HRx 寄存器中无任何摘要（返回零）

1：摘要计算已完成，摘要存储于 HASH_HRx 寄存器中。如果 HASH_IMR 寄存器中将 DCIE 位置 1，则产生中断。

位 0 **DINIS**：数据输入中断状态 (Data input interrupt status)

该位可在 FIFO 准备好获取新块时（16 个位置空闲）由硬件置 1。要将该位清零，可写入 0，或者写 HASH_DIN 寄存器。

0：输入缓冲器中的空闲位置少于 16 个

1：可以将一个新块输入到输入缓冲器中。如果 HASH_IMR 寄存器中将 DINIE 位置 1，则产生中断。

当 DINIS = 0 时，HASH_CSRx 寄存器读为零。

25.7.7 HASH 上下文交换寄存器

HASH context swap registers

这些寄存器包含哈希处理器的完整内部寄存器状态。如果高优先级的任务需要使用哈希处理器，而该处理器正在执行其他任务，则必须使用上述寄存器执行挂起/恢复操作。

发生此类事件时，必须读取 HASH_CSRx 寄存器，并且读取值必须保存到系统存储器空间中。随后，可以使用哈希处理器执行高优先级的任务。完成哈希计算之后，可从存储器中读取保存的上下文并将其写回 HASH_CSRx 寄存器。

仅当 DINIS 等于 1 时才能读取 HASH_CSRx 寄存器，否则读操作将返回零。

HASH 上下文交换寄存器 x (HASH_CSRx)

HASH context swap register x

偏移地址：0x0F8 + x * 0x4，（x = 0 到 53）

复位值：0x0022 0002 (HASH_CSR0)

复位值：0x0000 0000（其他）

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

位 31:0 **CSx[31:0]**：上下文交换 x (Context swap x)

请参见*第 25.7.7 节：HASH 上下文交换寄存器*简介。

25.7.8 HASH 寄存器映射

表 161. HASH1 寄存器映射和复位值

偏移	寄存器名称 复位值	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	HASH_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		ALGO[1:0]		LKEY	Res.	Res.	MDMAT	DINNE	NBW[3:0]				Res.	MODE	DATATYPE[1:0]		DMAE	INIT	Res.	Res.
	Reset value												0	0	0	0	0			0	0	0	0	0	0		0	0	0	0	0		
0x04	HASH_DIN	DATAIN[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	HASH_STR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DCAL	Res.	Res.	Res.	NBLW[4:0]				
	Reset value																								0				0	0	0	0	0
0x0C	HASH_HR0	H0[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	HASH_HR1	H1[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	HASH_HR2	H2[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	HASH_HR3	H3[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	HASH_HR4	H4[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	HASH_IMR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DCIE	DINIE
	Reset value																															0	0
0x24	HASH_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		NBWE[4:0]				DINNE			NBWP[4:0]					Res.	Res.	Res.	Res.	Res.	BUSY	DMAS	DCIS	DINIS
	Reset value												1	0	0	0	1	0		0	0	0	0	0						0	0	0	1
Reserved																																	
0xF8	HASH_CSR0	CSR0[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0xFC	HASH_CSR1	CSR1[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x100	HASH_CSR2	CSR2[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
...																																	
0x1CC	HASH_CSR53	CSR53[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reserved																																	
0x310	HASH_HR0	H0[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x314	HASH_HR1	H1[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x318	HASH_HR2	H2[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x31C	HASH_HR3	H3[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x320	HASH_HR4	H4[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x324	HASH_HR5	H5[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x328	HASH_HR6	H6[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x32C	HASH_HR7	H7[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
...																																	

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

26 高级控制定时器 (TIM1)

在本章中，应将“TIMx”理解为“TIM1”，因为本参考手册所适用的产品中只有一个此类定时器的实例。

26.1 TIM1 简介

高级控制定时器 (TIM1) 包含一个 16 位自动重载计数器，该计数器由可编程预分频器驱动。

此类定时器可用于多种用途，包括测量输入信号的脉冲宽度（输入捕获），或者生成输出波形（输出比较、PWM 和带死区插入的互补 PWM）。

使用定时器预分频器和 RCC 时钟控制器预分频器，可将脉冲宽度和波形周期从几微秒调制到几毫秒。

高级控制定时器 (TIM1) 和通用定时器 (TIMy) 彼此完全独立，不共享任何资源。如*第 26.3.30 节：定时器同步*中所述，它们可以一起同步操作。

26.2 TIM1 主要特性

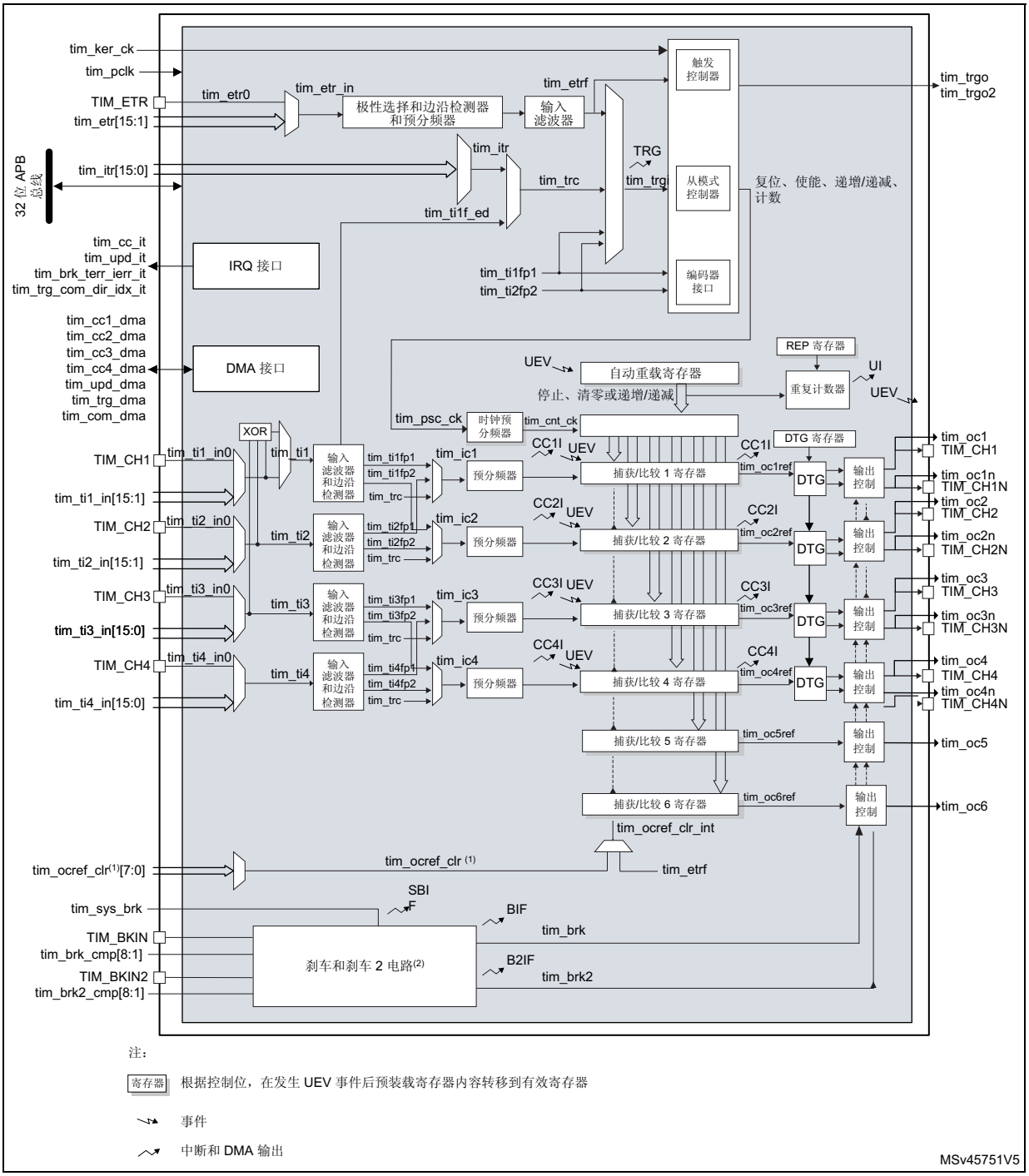
TIM1 定时器主要特性：

- 16 位递增、递减、递增/递减自动重载计数器。
- 16 位可编程预分频器，用于对计数器时钟频率进行分频（可在运行时修改），分频系数介于 1 到 65536 之间。
- 多达 6 个独立通道，可用于：
  - 输入捕获（但通道 5 和通道 6 除外）
  - 输出比较
  - PWM 生成（边沿和中心对齐模式）
  - 单脉冲模式输出
- 带可编程死区的互补输出
- 同步电路，用于实现外部信号对定时器的控制以及多个定时器互联。
- 重复计数器，用于仅在给定数目的计数周期后更新定时器寄存器。
- 2 个刹车输入，用于将定时器的输出信号置于用户可选的安全配置中。
- 发生如下事件时生成中断/DMA 请求：
  - 更新：计数器上溢/下溢、计数器初始化（通过软件或内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或通过内部/外部触发计数）
  - 输入捕获
  - 输出比较
- 支持定位用增量（正交）编码器和霍尔传感器电路
- 触发输入可用作外部时钟或用于逐周期电流管理

26.3 TIM1 功能描述

26.3.1 框图

图 161. 高级控制定时器框图



1. 并非所有定时器都具有此特性，请参见*第 26.3.2 节：TIM1 引脚和内部信号*。
2. 详细信息，请参见*图 208：刹车事件和刹车 2 电路概述*。

26.3.2 TIM1 引脚和内部信号

本节中的表格对 TIM 输入和输出进行了汇总。

表 162. TIM 输入/输出引脚

引脚名称	信号类型	说明
TIM_CH1 TIM_CH2 TIM_CH3 TIM_CH4	输入/输出	定时器多用途通道。 每个通道均可用于捕获、比较或 PWM。 TIM_CH1 和 TIM_CH2 还可用作外部时钟（低于 1/4 tim_ker_ck 时钟频率）、外部触发输入和正交编码器输入。 TIM_CH1、TIM_CH2 和 TIM_CH3 可用于连接数字霍尔效应传感器。
TIM_CH1N TIM_CH2N TIM_CH3N TIM_CH4N	输出	定时器互补输出，由 TIM_CHx 输出提供，可插入死区。
TIM_ETR	输入	外部触发输入。该输入可用作外部触发信号或外部时钟源。如果使用 tim_etr_in 预分频器，则该输入可以接受频率高于 tim_ker_ck 的时钟。
TIM_BKIN TIM_BKIN2	输入/输出	刹车和刹车 2 输入。这些输入也可以配置为双向模式。

表 163. TIM 内部输入/输出信号

内部信号名称	信号类型	说明
tim_ti1_in[15:0] tim_ti2_in[15:0] tim_ti3_in[15:0] tim_ti4_in[15:0]	输入	内部定时器输入总线。tim_ti1_in[15:0] 和 tim_ti2_in[15:0] 输入可用于捕获或用作外部时钟（低于 1/4 tim_ker_ck 时钟频率）以及正交编码器信号。
tim_etr[15:0]	输入	外部触发内部输入总线。这些输入可用作触发信号、外部时钟或用于硬件逐周期脉宽控制。如果使用 tim_etr_in 预分频器，则这些输入可以接受频率高于 tim_ker_ck 的时钟。
tim_itr[15:0]	输入	内部触发输入总线。这些输入可用于从模式控制器或用作输入时钟（低于 1/4 tim_ker_ck 时钟频率）。
tim_trgo/tim_trgo2	输出	内部触发输出。这些触发信号由其他定时器和/或其他外设使用。
tim_ocref_clr[7:0]	输入	定时器 tim_ocref_clr 输入总线。这些输入可用于清除 tim ocxref 信号，通常用于硬件逐周期脉宽控制。
tim_brk_cmp[8:1]	输入	内部信号刹车输入
tim_brk2_cmp[8:1]	输入	内部信号刹车 2 输入
tim_sys_brk[n:0]	输入	系统刹车输入。该输入收集 MCU 的系统级错误。

表 163. TIM 内部输入/输出信号（续）

内部信号名称	信号类型	说明
tim_pclk	输入	定时器 APB 时钟
tim_ker_ck	输入	定时器内核时钟
tim_cc_it	输出	定时器捕获/比较中断
tim_upd_it	输出	定时器更新事件中断
tim_brk_terr_ierr_it	输出	定时器刹车、刹车 2、转换错误和索引错误中断
tim_trg_com_dir_idx_it	输出	定时器触发、换向、方向和索引中断
tim_cc1_dma tim_cc2_dma tim_cc3_dma tim_cc4_dma	输出	定时器捕获/比较事件的 DMA 请求
tim_upd_dma	输出	定时器更新事件的 DMA 请求
tim_trg_dma	输出	定时器触发事件的 DMA 请求
tim_com_dma	输出	定时器换相事件的 DMA 请求

表 164、*表 165*、*表 166* 和*表 167* 列出了连接到 tim_ti[4:1] 输入复用器的源。

表 164. tim_ti1 输入复用器的互连

tim_ti1 输入	源
	TIM1
tim_ti1_in0	TIM1_CH1
tim_ti1_in1	COMP1_OUT
tim_ti1_in[15:2]	保留

表 165. tim_ti2 输入复用器的互连

tim_ti2 输入	源
	TIM1
tim_ti2_in0	TIM1_CH2
tim_ti2_in[15:1]	保留

表 166. tim_ti3 输入复用器的互连

tim_ti3 输入	源
	TIM1
tim_ti3_in0	TIM1_CH3
tim_ti2_in[15:1]	保留

表 167. tim_ti4 输入复用器的互连

tim_ti4 输入	源
	TIM1
tim_ti4_in0	TIM1_CH4
tim_ti4_in[15:1]	保留

表 168 列出了连接到 tim_itr 输入复用器的内部源。

表 168. 内部触发连接

定时器内部触发输入信号	TIM1
tim_itr0	保留
tim_itr1	tim2_trgo
tim_itr2	tim3_trgo
tim_itr[15:3]	保留

表 169 列出了连接到 tim_etr 输入复用器的内部源。

表 169. tim_etr 输入复用器的互连

定时器外部触发输入信号	定时器外部触发信号分配
	TIM1
tim_etr0	TIM1_ETR
tim_etr1	COMP1_OUT
tim_etr2	保留
tim_etr3	adc1_awd1
tim_etr4	adc1_awd2
tim_etr5	adc1_awd3
tim_etr[15:6]	保留

表 170、*表 171* 和*表 172* 列出了连接到 tim_brk 和 tim_brk2 输入的源。

表 170. 定时器刹车互连

tim_brk 输入	TIM1
TIM_BKIN	TIM1_BKIN 引脚
tim_brk_cmp1	COMP1_OUT
tim_brk_cmp[8:2]	保留

表 171. 定时器刹车 2 互连

tim_brk2 输入	TIM1
TIM_BKIN2	TIM1_BKIN2 引脚
tim_brk2_cmp1	COMP1_OUT
tim_brk2_cmp[8:2]	保留

表 172. 系统刹车互连

tim_sys_brk 输入	TIM1	SBS_CFGR2 寄存器中的使能位
tim_sys_brk0	FLASH 双重 ECC 错误	ECCL
tim_sys_brk1	可编程电压检测器 (PVD)	PVDL
tim_sys_brk2	SRAM 双重 ECC 错误	SEL
tim_sys_brk3	Cortex®-M33 LOCKUP	CLL
CSS	时钟安全系统	无（始终使能）

26.3.3 时基单元

可编程高级控制定时器的主要模块是一个 16 位计数器及其相关的自动重载寄存器。计数器可递增计数、递减计数或进行递增递减计数。计数器的时钟可通过预分频器进行分频。

即使在计数器运行时，计数器、自动重载寄存器和预分频器寄存器也可以通过软件进行读写。

时基单元包括：

- 计数器寄存器 (TIMx_CNT)
- 预分频器寄存器 (TIMx_PSC)
- 自动重载寄存器 (TIMx_ARR)
- 重复计数器寄存器 (TIMx_RCR)

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以随时直接写入影子寄存器，也可以在每次发生更新事件 (UEV) 时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位 (ARPE)。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将产生更新事件。该更新事件也可由软件产生。下文将详细介绍每个配置中更新事件的产生方式。

计数器由预分频器输出 tim_cnt_ck 提供时钟，仅当 TIMx_CR1 寄存器中的计数器使能位 (CEN) 置 1 时，才会使能计数器（有关计数器使能的更多详细信息，另请参见从模式控制器的相关说明）。

注意： *计数器将在 TIMx_CR1 寄存器的 CEN 位置 1 时刻的一个时钟周期后开始计数。*

预分频器说明

预分频器可对计数器时钟进行分频，分频系数介于 1 和 65536 之间。该预分频器基于 16 位寄存器 TIMx_PSC 所控制的 16 位计数器。由于该控制寄存器具有缓冲功能，因此预分频器可实现实时更改。而新的预分频比将在下一更新事件发生时被采用。

图 162 和*图 163* 以一些示例说明在预分频比实时变化时计数器的行为。

图 162. 预分频器分频由 1 变为 2 时的计数器时序图

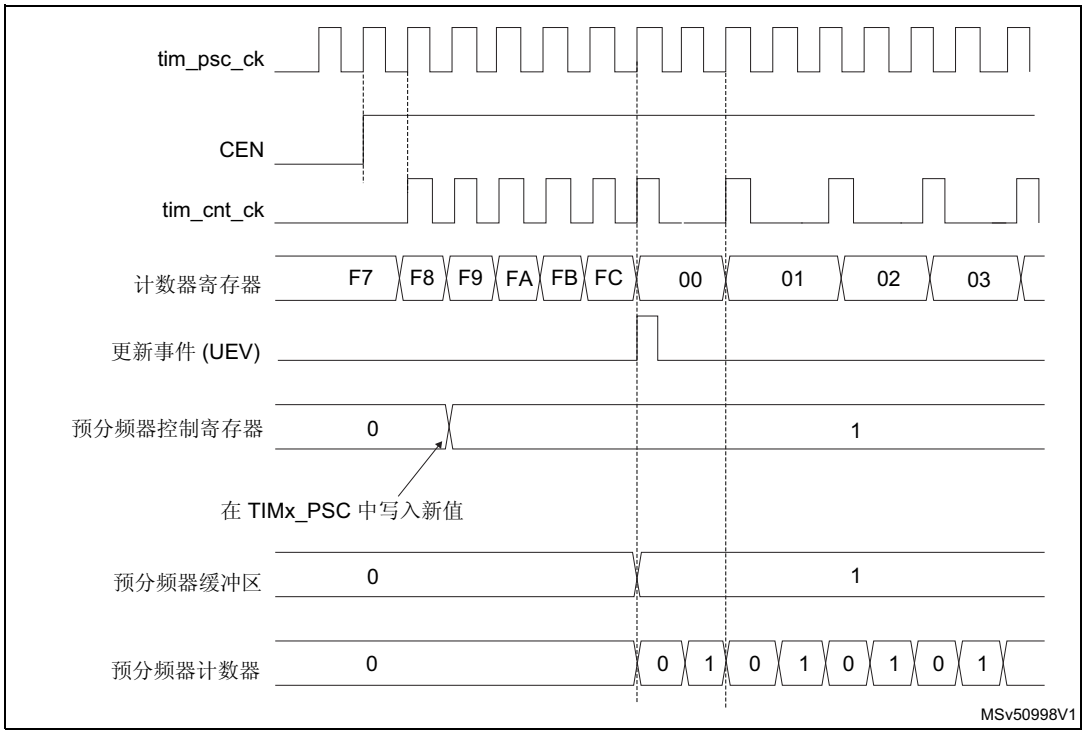
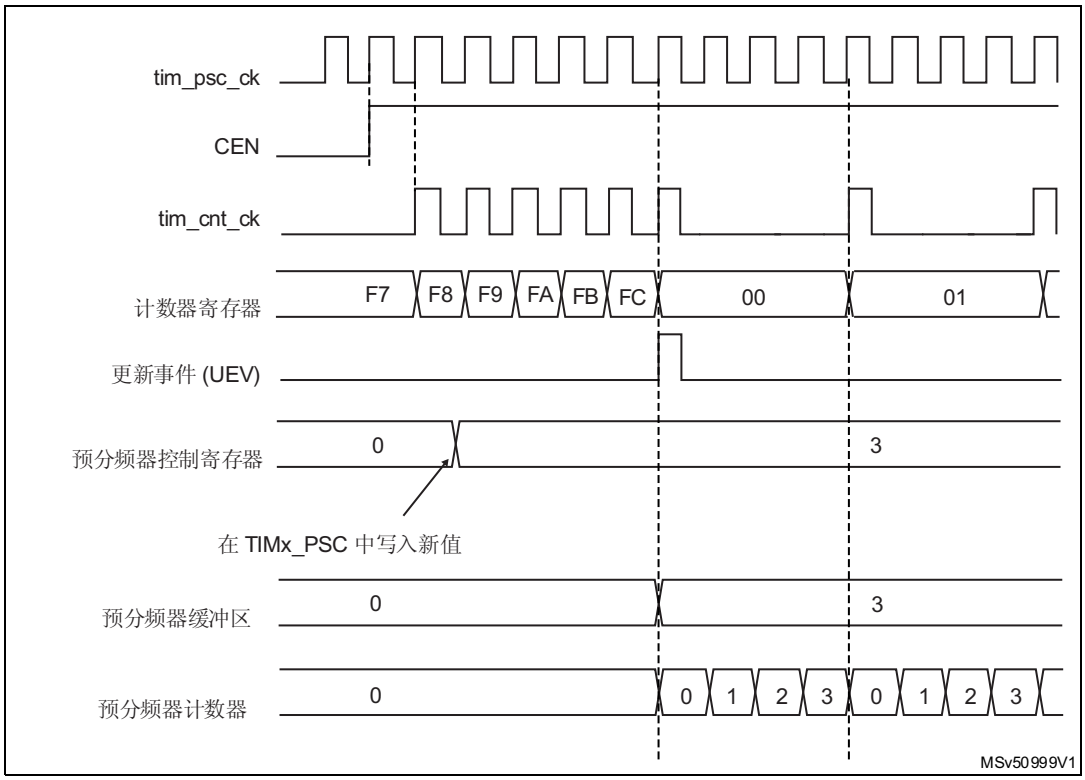


图 163. 预分频器分频由 1 变为 4 时的计数器时序图



26.3.4 计数器模式

递增计数模式

在递增计数模式下，计数器从 0 计数到自动重载值（TIMx_ARR 寄存器的内容），然后重新从 0 开始计数并生成计数器上溢事件。

如果使用重复计数器，则当递增计数的重复次数达到重复计数器寄存器中设定的次数加一次（(TIMx_RCR) + 1）后，将生成更新事件 (UEV)。否则，将在每次计数器上溢时产生更新事件。

将 TIMx_EGR 寄存器的 UG 位置 1（通过软件或使用从模式控制器）时，也将产生更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变）。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。利用这一特性，在定时器的输入捕获操作时可以通过对 UG 位置 1 让计数器复位而不至于触发更新中断，也就避免了在捕获中断里触发更新中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容，
- 自动重载影子寄存器将以预装载值 (TIMx_ARR) 进行更新，
- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

图 164. 计数器时序图，1 分频内部时钟

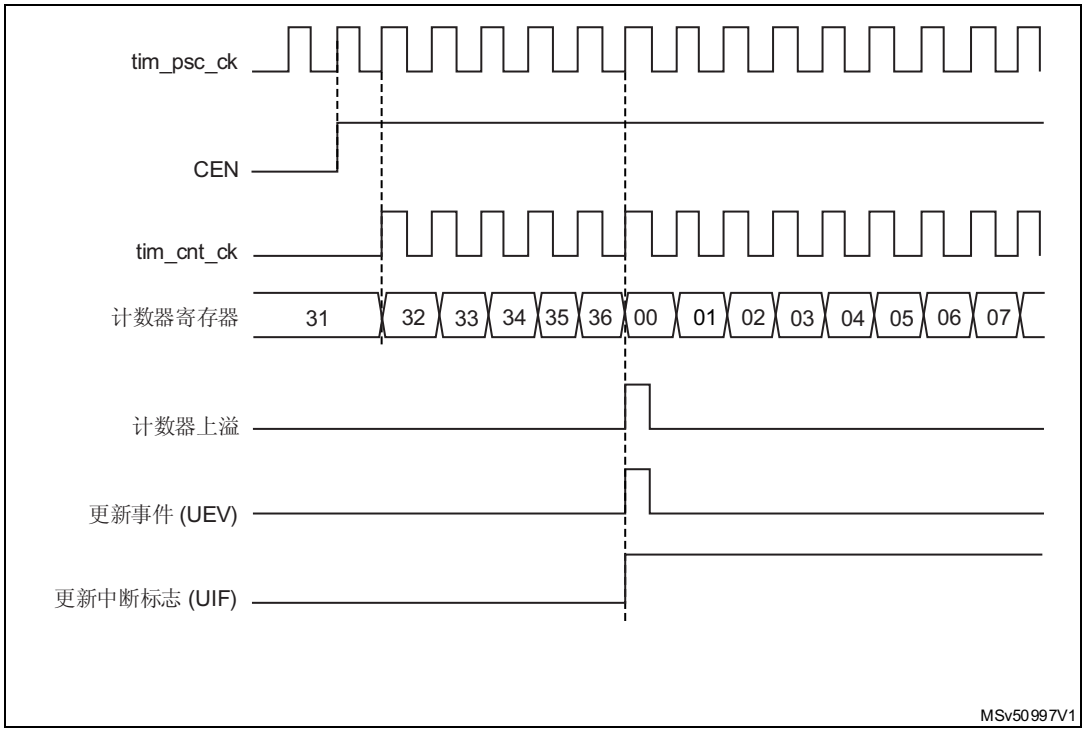


图 165. 计数器时序图，2 分频内部时钟

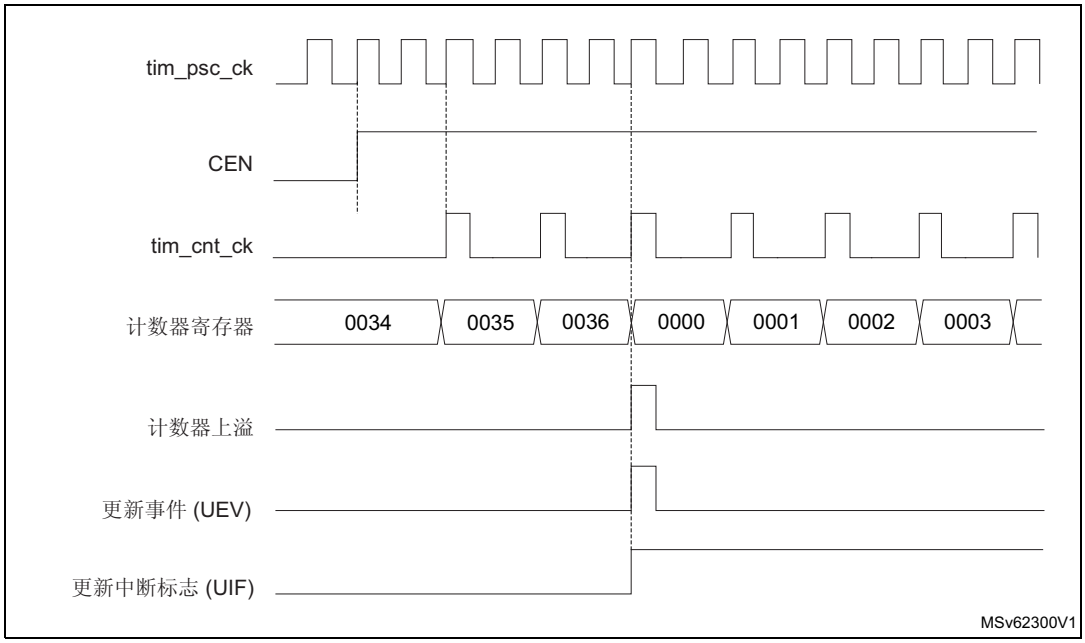


图 166. 计数器时序图，4 分频内部时钟

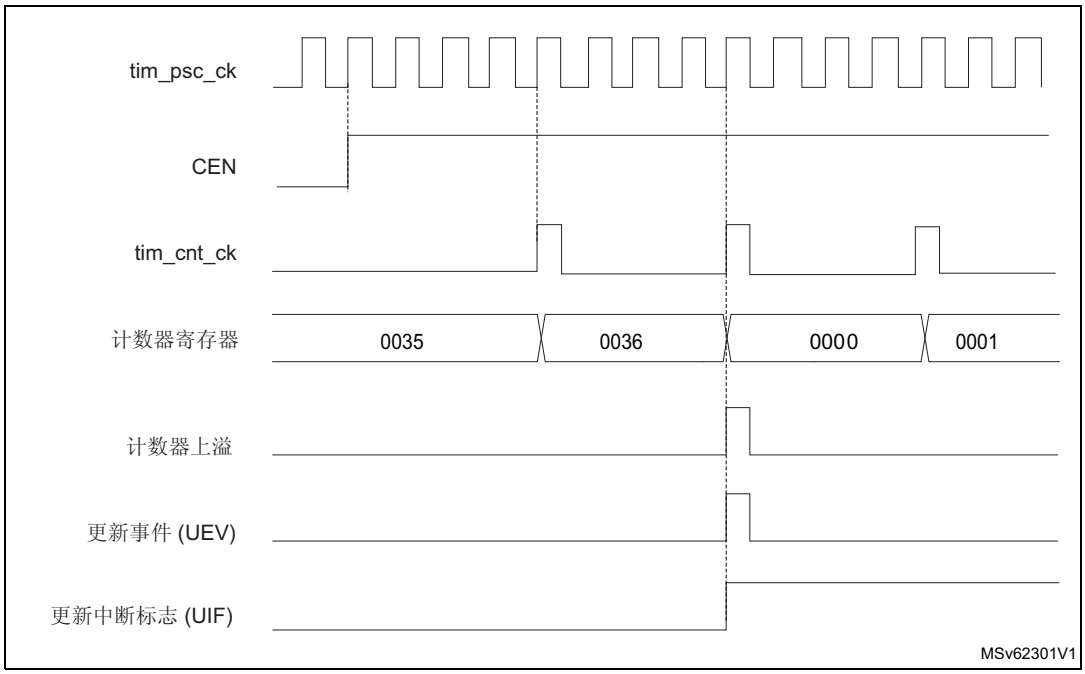


图 167. 计数器时序图，N 分频内部时钟

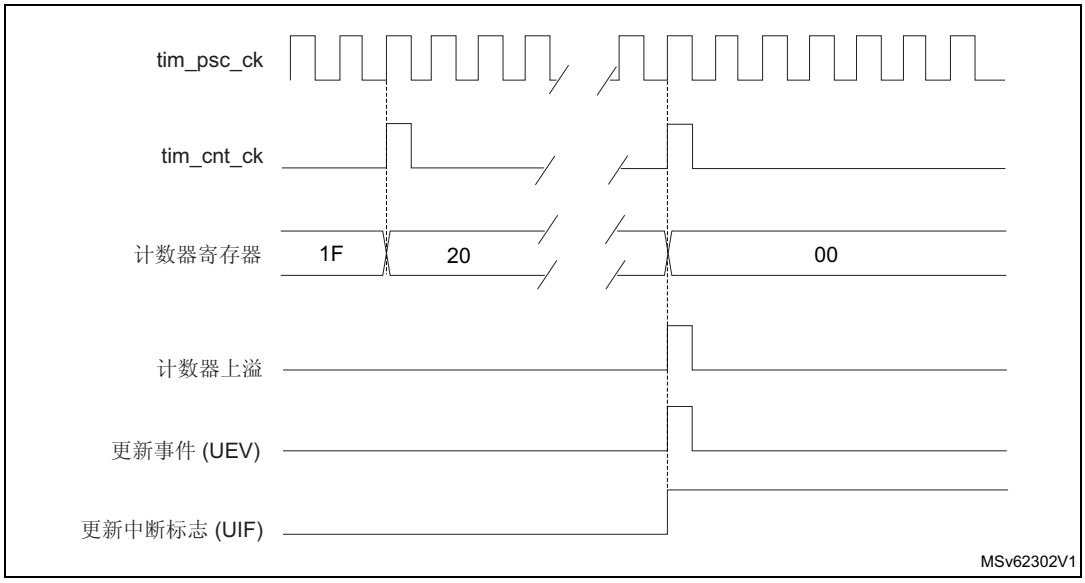


图 168. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载）

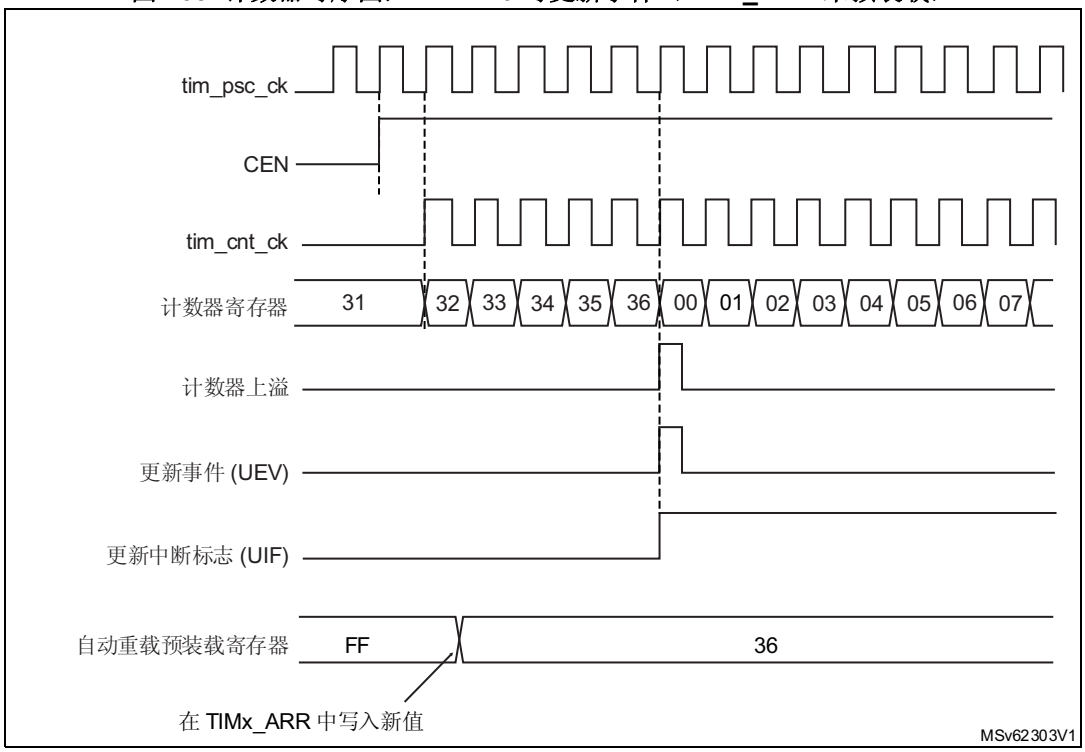
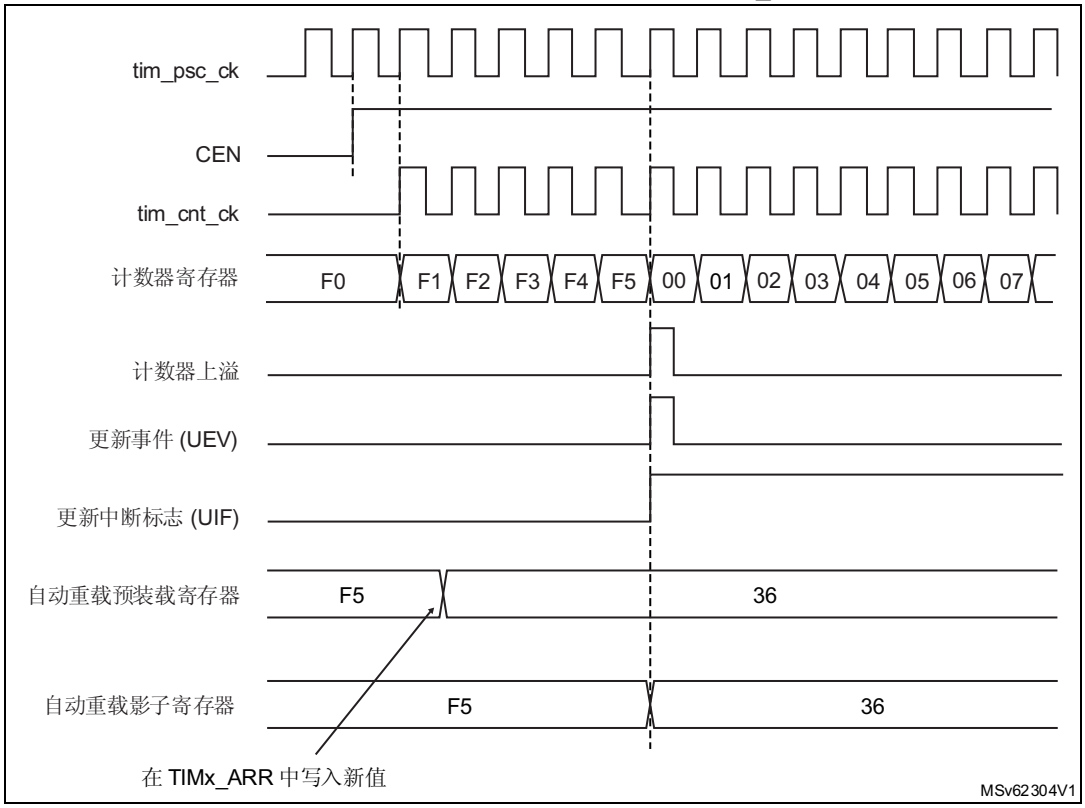


图 169. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 预装载）



递减计数模式

在递减计数模式下，计数器从自动重载值（TIMx_ARR 寄存器的内容）开始递减计数到 0，然后重新从自动重载值开始计数并生成计数器下溢事件。

如果使用重复计数器，则当递减计数的重复次数达到重复计数器寄存器中编程的次数加一次（(TIMx_RCR) + 1）后，将产生更新事件 (UEV)。否则，将在每次计数器下溢时产生更新事件。

将 TIMx_EGR 寄存器的 UG 位置 1（通过软件或使用从模式控制器）时，也将产生更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器会从当前自动重载值开始重新计数，而预分频器计数器则从 0 开始重新计数（但预分频比保持不变）。

此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。利用这一特性，在定时器的输入捕获操作时可以通过对 UG 位置 1 让计数器复位而不至于触发更新中断，也就避免了在捕获中断里触发更新中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容。
- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 自动重载影子寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，ARR 寄存器更新在计数器重载之前被更新，因此下一个周期就是预期的值。

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

图 170. 计数器时序图，1 分频内部时钟

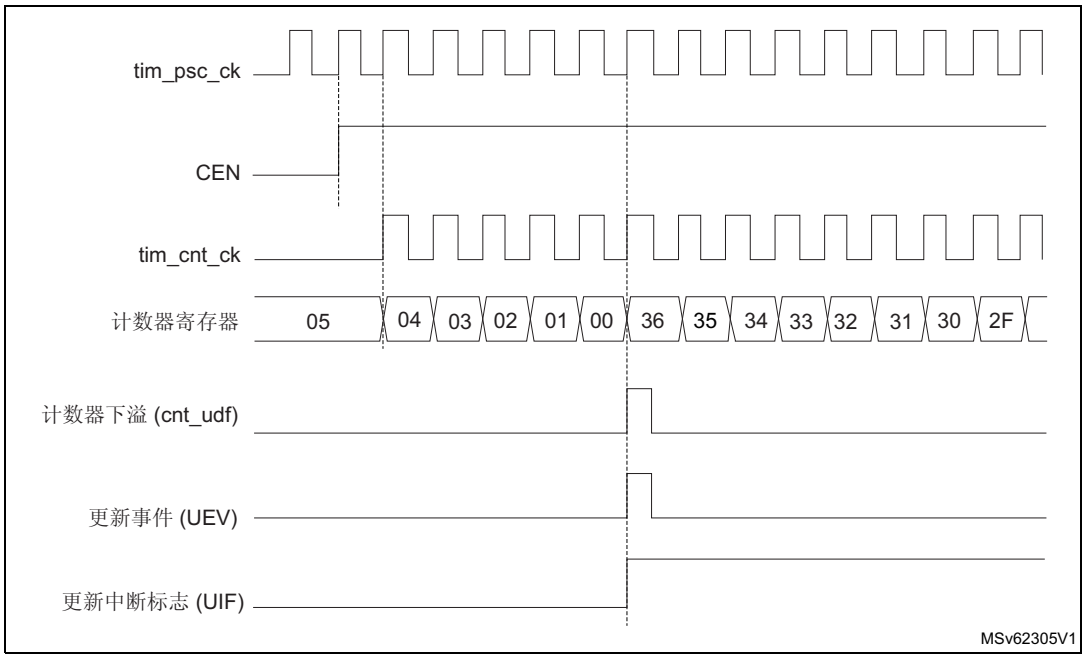


图 171. 计数器时序图，2 分频内部时钟

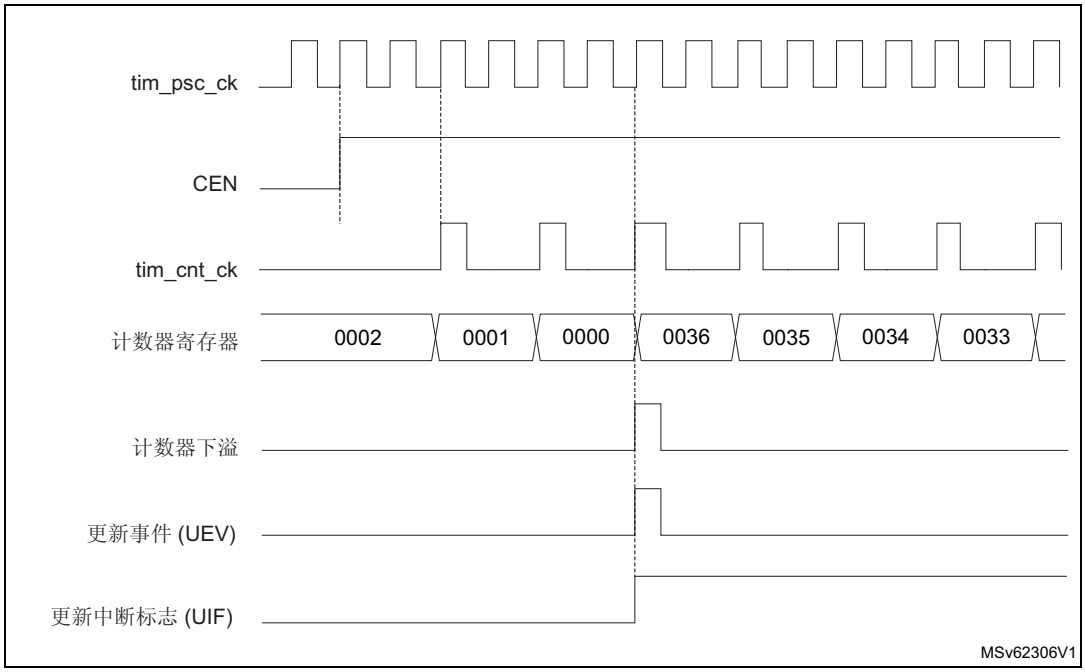


图 172. 计数器时序图，4 分频内部时钟

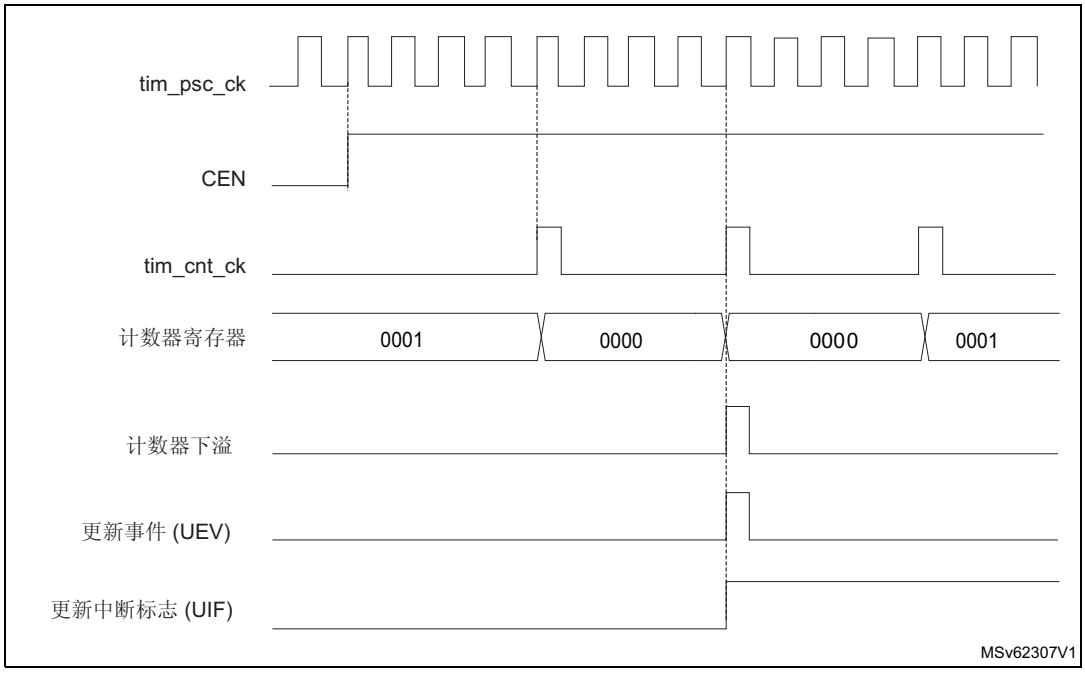


图 173. 计数器时序图，N 分频内部时钟

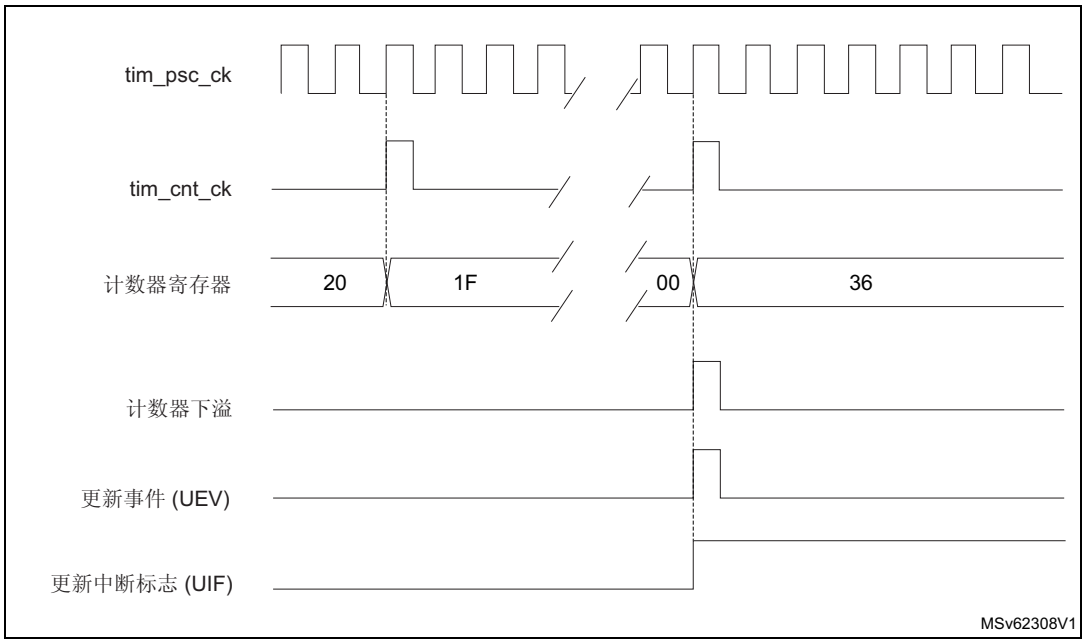
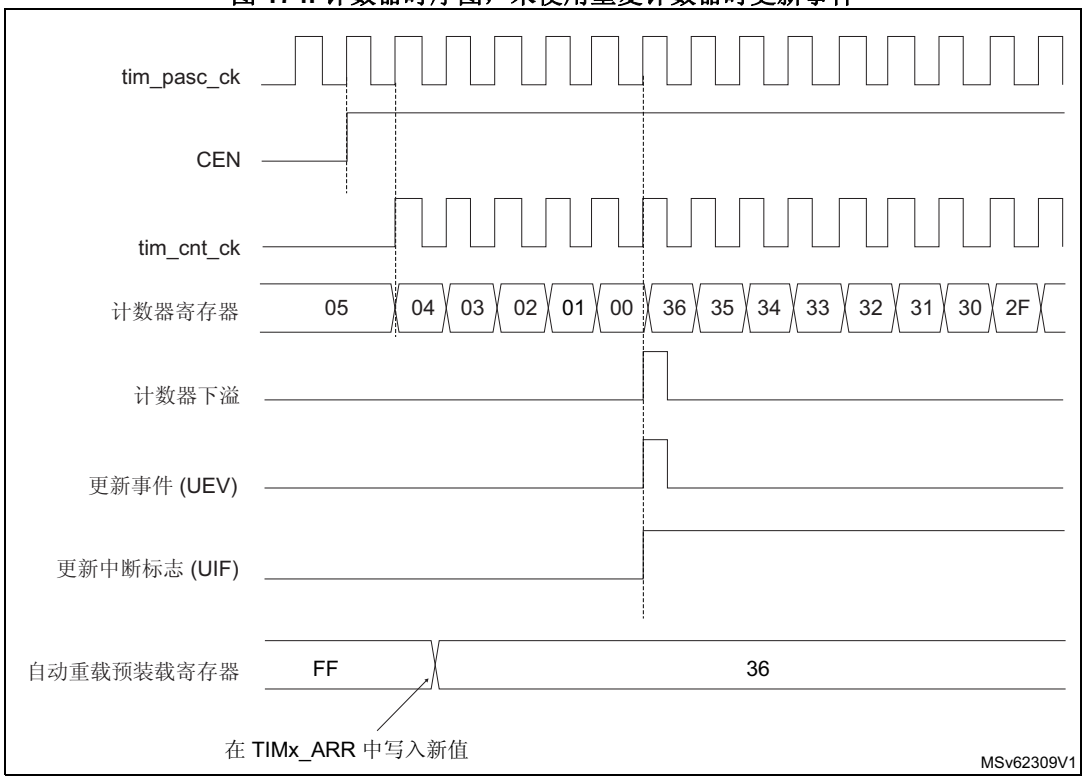


图 174. 计数器时序图，未使用重复计数器时更新事件



中心对齐模式（递增/递减计数）

在中心对齐模式下，计数器从 0 开始计数到自动重载值（TIMx_ARR 寄存器的内容）– 1，生成计数器上溢事件；然后从自动重载值开始向下计数到 1 并生成计数器下溢事件。之后从 0 开始重新计数。

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时，中心对齐模式有效。将通道配置为输出模式时，其输出比较中断标志将在以下模式下置 1，即：计数器递减计数（中心对齐模式 1，CMS =“01”）、计数器递增计数（中心对齐模式 2，CMS =“10”）以及计数器递增/递减计数（中心对齐模式 3，CMS =“11”）。

在此模式下，TIMx_CR1 寄存器的 DIR 方向位不可写入值，而是由硬件更新并指示当前计数器方向。

每次发生计数器上溢和下溢时都会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。这种情况下，计数器以及预分频器计数器将重新从 0 开始计数。

UEV 更新事件可通过软件禁止，只需将 TIMx_CR1 寄存器中的 UDIS 位置 1。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器仍会根据当前自动重载值进行递增和递减计数。

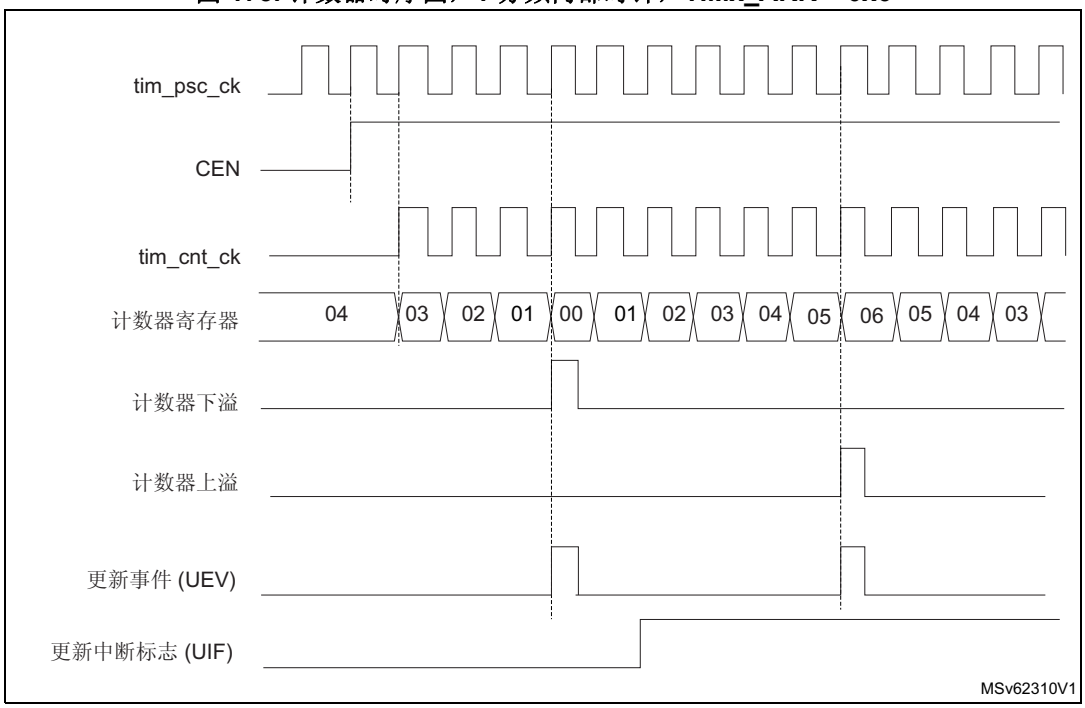
此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成 UEV 更新事件，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。利用这一特性，在定时器的输入捕获操作时可以通过对 UG 位置 1 让计数器复位而不至于触发更新中断，也就避免了在捕获中断里触发更新中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容
- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）
- 自动重载影子寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，如果更新操作是由计数器上溢触发的，则 ARR 寄存器在计数器重载之前更新，因此，下一个计数周期就是我们所希望的新的周期长度（计数器被重载新的值）。

以下各图以一些示例说明不同时钟频率下计数器的行为。

图 175. 计数器时序图，1 分频内部时钟，TIMx_ARR = 0x6



1. 此处使用中心对齐模式 1（有关详细信息，请参见*第 26.6 节：TIM1 寄存器*）。

图 176. 计数器时序图，2 分频内部时钟

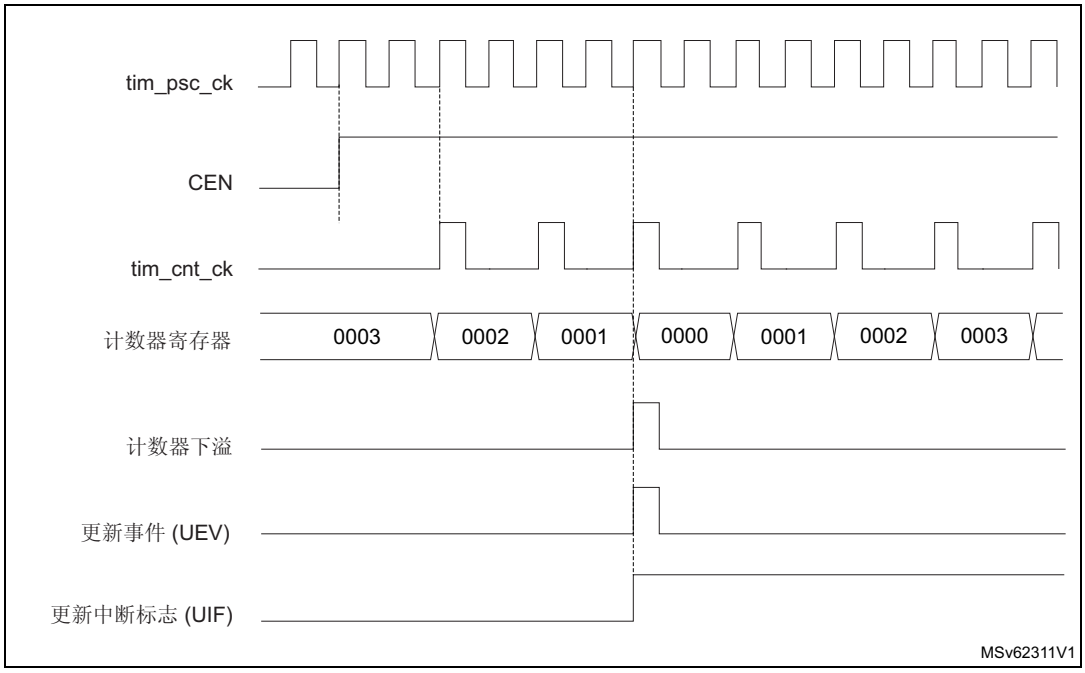


图 177. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36

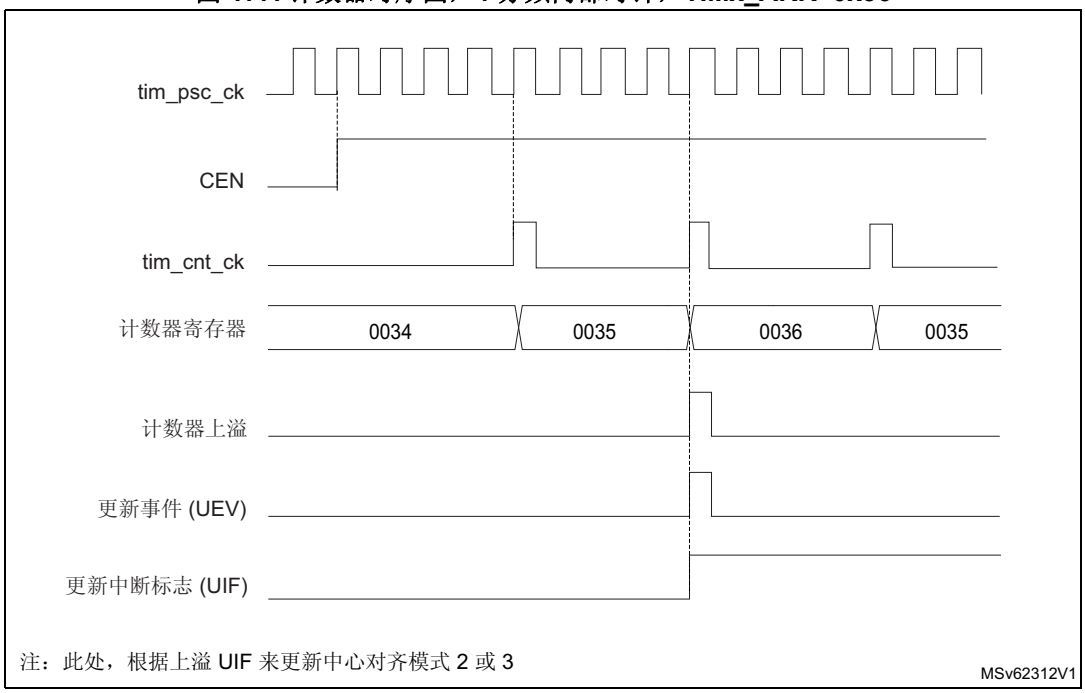


图 178. 计数器时序图，N 分频内部时钟

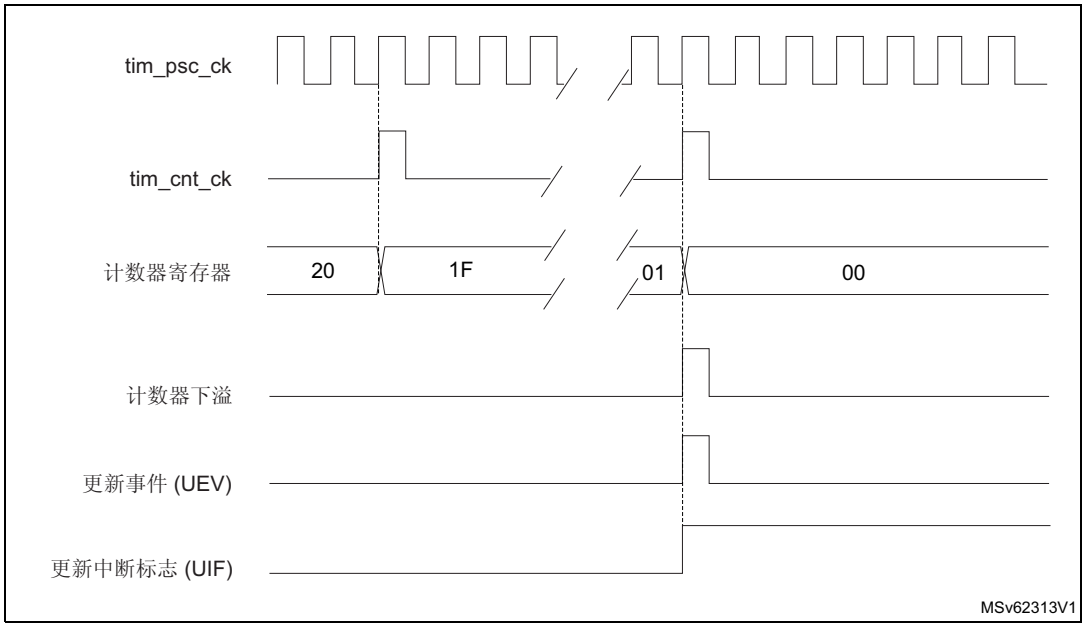


图 179. 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

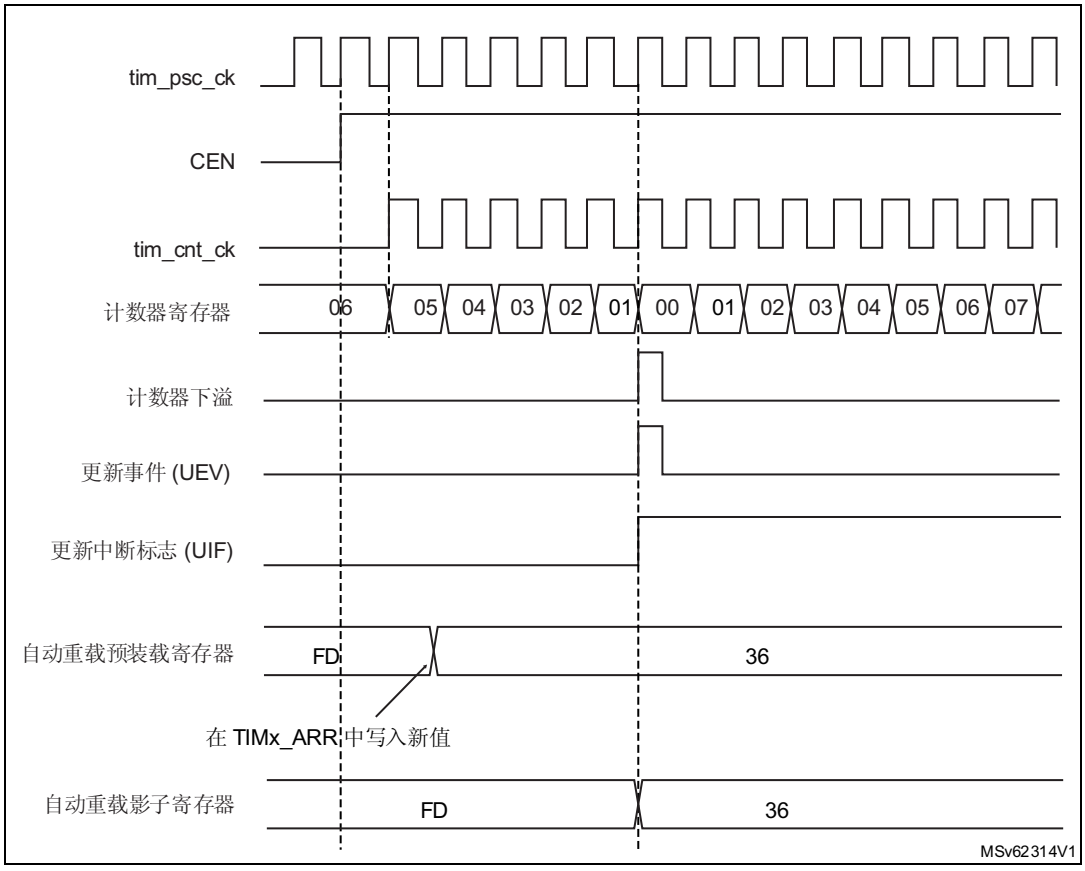
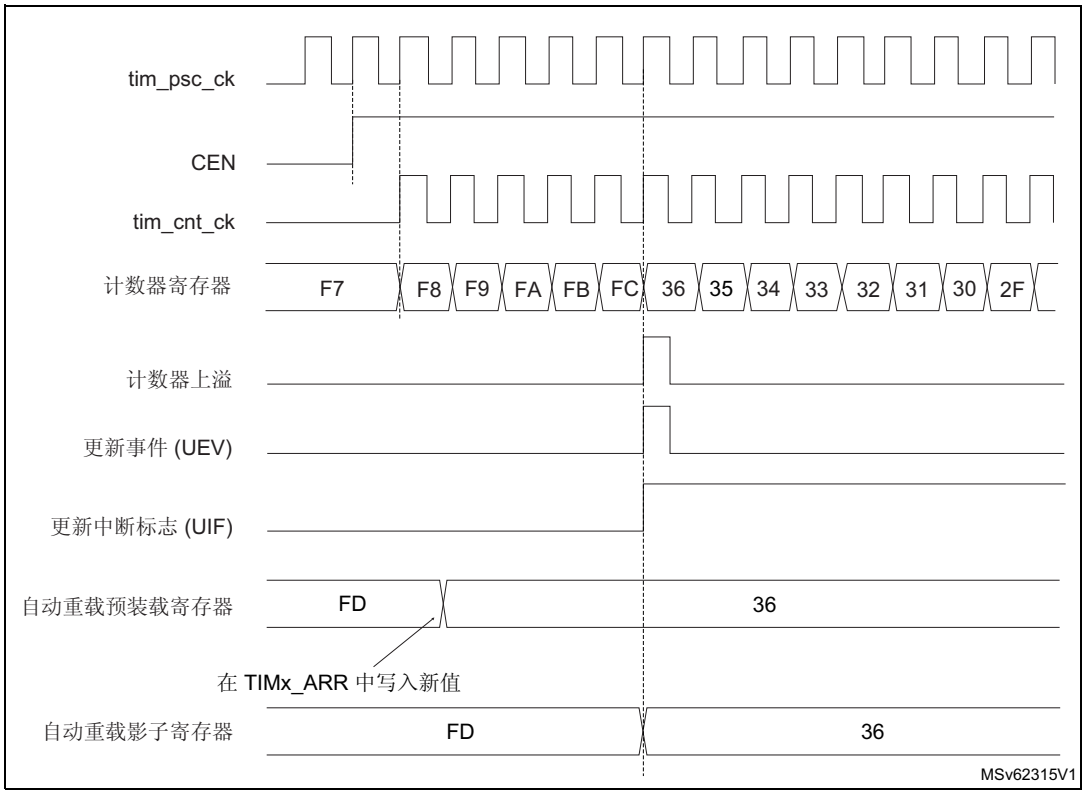


图 180. 计数器时序图，ARPE=1 时的更新事件（计数器上溢）



26.3.5 重复计数器

*第 26.3.3 节：时基单元*介绍如何因计数器上溢/下溢而生成更新事件 (UEV)。实际上，只有当重复计数器达到零时，才会生成更新事件。这在生成 PWM 信号时很有用。

这意味着，每当发生 N+1 个计数器上溢或下溢（其中，N 是 TIMx_RCR 重复计数器寄存器中的值），数据就将从预装载寄存器转移到影子寄存器（TIMx_ARR 自动重载寄存器、TIMx_PSC 预分频器寄存器以及比较模式下的 TIMx_CCRx 捕获/比较寄存器）。

重复计数器在下列情况下递减：

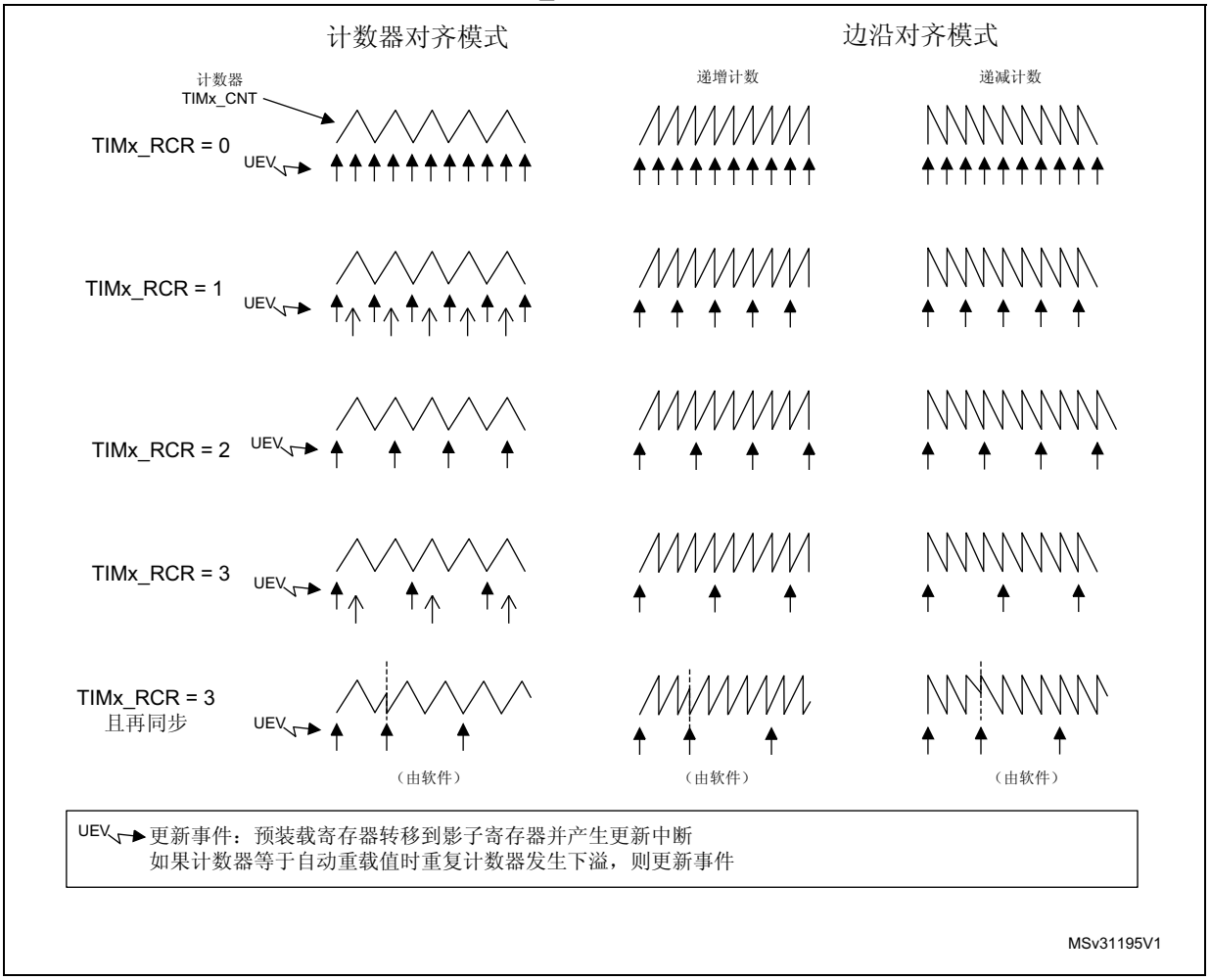
- 递增计数模式下的每个计数器上溢，
- 递减计数模式下的每个计数器下溢，
- 中心对齐模式下每个计数器上溢和计数器下溢。尽管这使得最大重复次数不超过 32768 个 PWM 周期，但在每个 PWM 周期内可更新占空比两次。当在中心对齐模式下，每个 PWM 周期仅刷新一次比较寄存器时，由于模式的对称性，最大分辨率为 $2xT_{ck}$。

重复计数器是自动重载类型；其重复率为 TIMx_RCR 寄存器所定义的值（请参见*图 181*）。当更新事件由软件（通过将 TIMx_EGR 寄存器的 UG 位置 1）或硬件（通过从模式控制器）生成时，无论重复计数器的值为多少，更新事件都将立即发生，并且在重复计数器中重新装载 TIMx_RCR 寄存器的内容。

在中心对齐模式下，如果 RCR 值为奇数，更新事件将在上溢或下溢时发生，这取决于何时写入 RCR 寄存器以及何时启动计数器：如果在启动计数器前写入 RCR，则 UEV 在下溢时发生。如果在启动计数器后写入 RCR，则 UEV 在上溢时发生。

例如，如果 RCR = 3，UEV 将在每个周期的第四个上溢或下溢事件时产生（取决于何时写入 RCR）。

图 181. 不同模式和 TIMx_RCR 寄存器设置下的更新频率示例



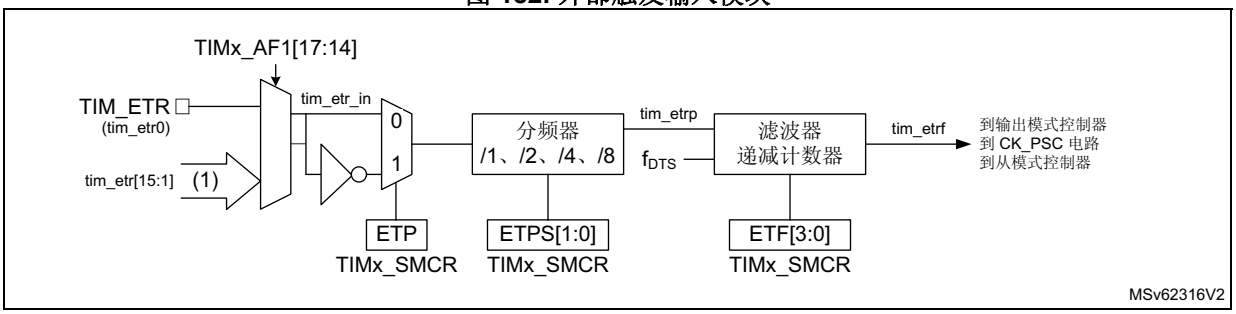
26.3.6 外部触发输入

定时器具有一个外部触发输入 tim_etr_in。它的用途是：

- 外部时钟（外部时钟模式 2，请参见*第 26.3.7 节*）
- 用于从模式的触发信号（请参见*第 26.3.30 节*）
- 用于逐周期电流调节的 PWM 复位输入（请参见*第 26.3.9 节*）

下面的*图 182* 介绍了 tim_etr_in 输入的调节过程。输入极性通过 TIMx_SMCR 寄存器中的 ETP 位定义。触发信号可通过 ETPS[1:0] 位域编程的分频比进行预分频，然后通过 ETF[3:0] 位域设置的滤波方式进行数字滤波。生成的信号 (tim_etrf) 有三种用途：作为外部时钟、调节输出（通常用于复位 PWM 输出以限制电流）以及作为从模式控制器的触发信号。

图 182. 外部触发输入模块



tim_etr_in 输入来自多个源：输入引脚（默认配置）或内部源。使用 TIMx_AF1 寄存器中的 ETRSEL[3:0] 位域进行选择。

有关连接到产品 etr_in 输入的源的列表，请参见*第 26.3.2 节：TIM1 引脚和内部信号*。

26.3.7 时钟选择

计数器时钟可由下列时钟源提供：

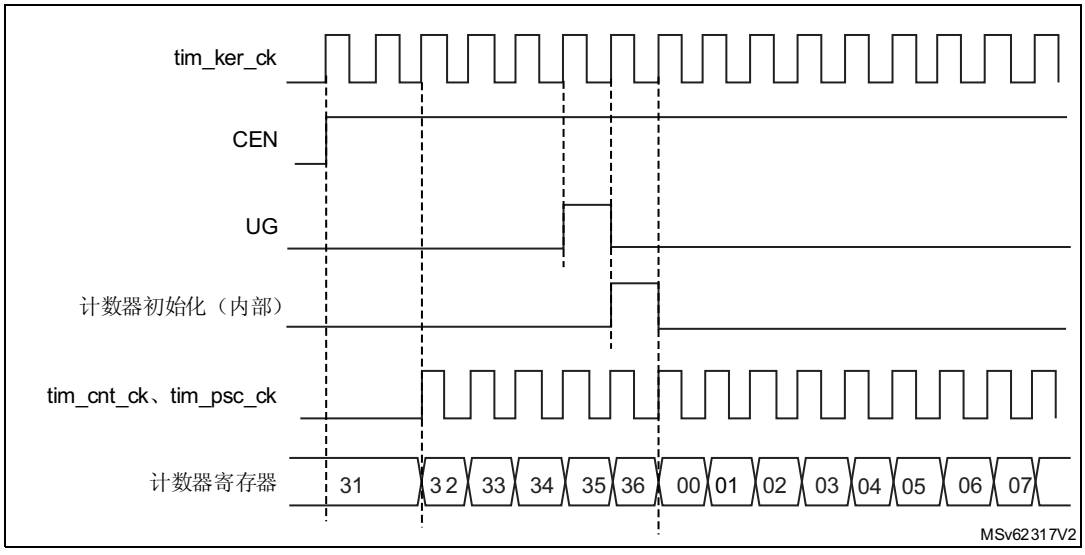
- 内部时钟 (tim_ker_ck)
- 外部时钟模式 1：外部输入引脚（tim_ti1 或 tim_ti2）
- 外部时钟模式 2：外部触发输入 (tim_etr_in)
- 编码器模式

内部时钟源 (tim_ker_ck)

如果禁止从模式控制器 (SMS=000)，则 CEN 位、DIR 位（TIMx_CR1 寄存器中）和 UG 位（TIMx_EGR 寄存器中）为实际控制位，并且只能通过软件进行更改（UG 除外，仍保持自动清零）。当对 CEN 位写入 1 时，预分频器的时钟就由内部时钟 tim_ker_ck 提供。

图 183 显示了正常模式下控制电路与递增计数器的行为（没有预分频的情况下）。

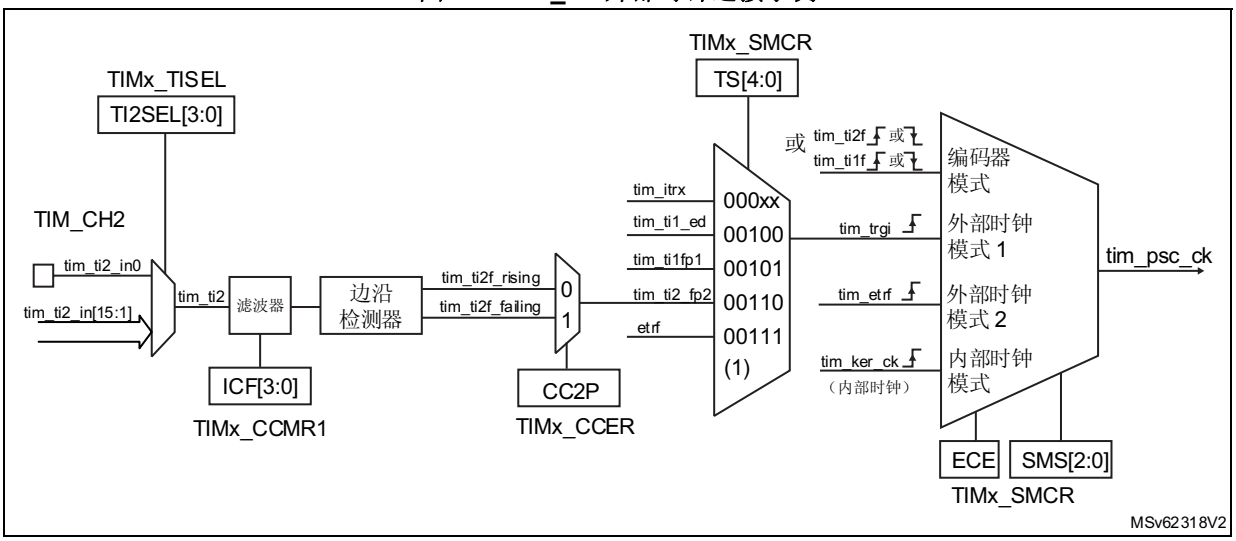
图 183. 正常模式下的控制电路，1 分频内部时钟



外部时钟源模式 1

当 TIMx_SMCR 寄存器中的 SMS=111 时，可选择此模式。计数器可在选定的输入信号上出现上升沿或下降沿时计数。

图 184. tim_ti2 外部时钟连接示例



1. 保留从 01000 到 11111 的代码。

例如，要使递增计数器在 tim_ti2 输入出现上升沿时计数，请执行以下步骤：

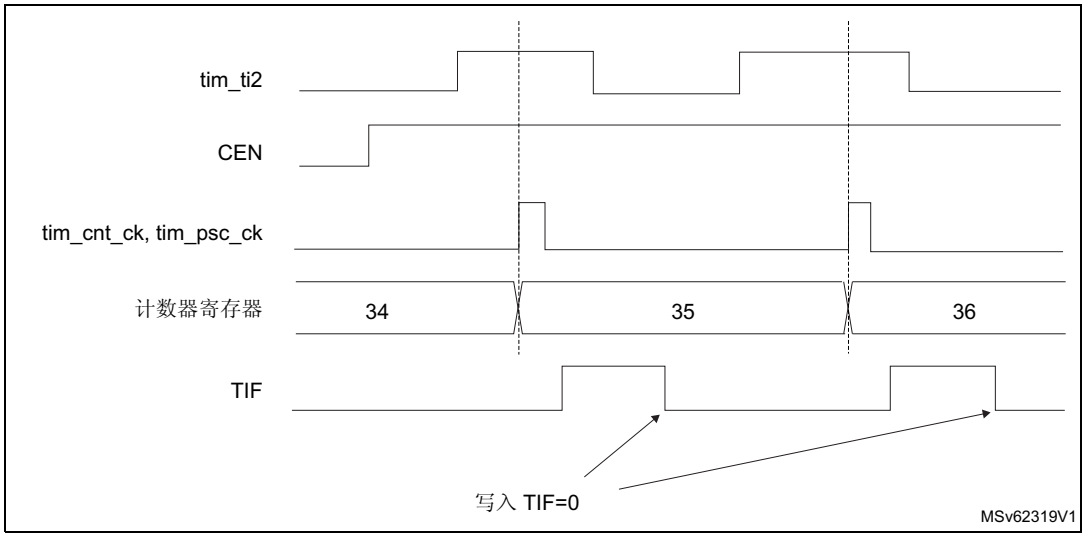
1. 通过在 TIMx_CCMR1 寄存器中写入 CC2S = “01”来配置通道 2，使其能够检测 tim_ti2 输入的上升沿。
2. 通过在 TIMx_CCMR1 寄存器中写入 IC2F[3:0] 位来配置输入滤波带宽（如果不需要任何滤波器，请保持 IC2F=0000）。
3. 通过在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=0 来选择上升沿极性。
4. 通过在 TIMx_SMCR 寄存器中写入 SMS=111，使定时器在外部时钟模式 1 下工作。
5. 通过在 TIMx_SMCR 寄存器中写入 TS = 00110 来选择 tim_ti2 作为触发输入源。
6. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

注意： *捕获预分频器不用于触发操作，无需对其进行配置。*

当 tim_ti2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

tim_ti2 的上升沿与实际计数器时钟之间的延迟是由于 tim_ti2 输入的重新同步电路引起的。

图 185. 外部时钟模式 1 下的控制电路



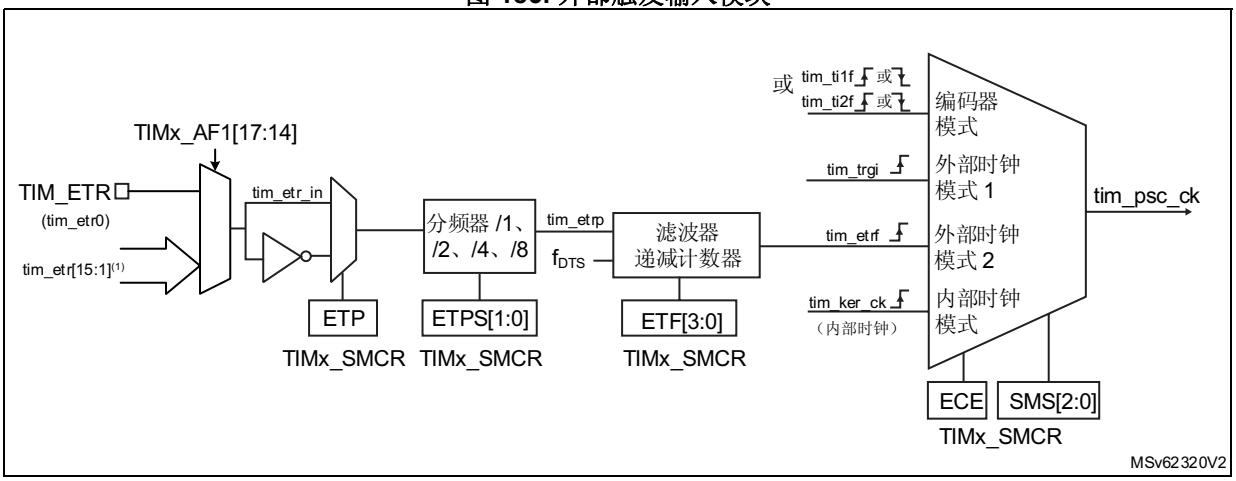
外部时钟源模式 2

通过在 TIMx_SMCR 寄存器中写入 ECE=1 可选择此模式。

计数器在外部触发输入 tim_etr_in 出现上升沿或下降沿时计数。

图 186 简要介绍了外部触发输入模块。

图 186. 外部触发输入模块



1. 请参见*第 26.3.2 节：TIM1 引脚和内部信号*。

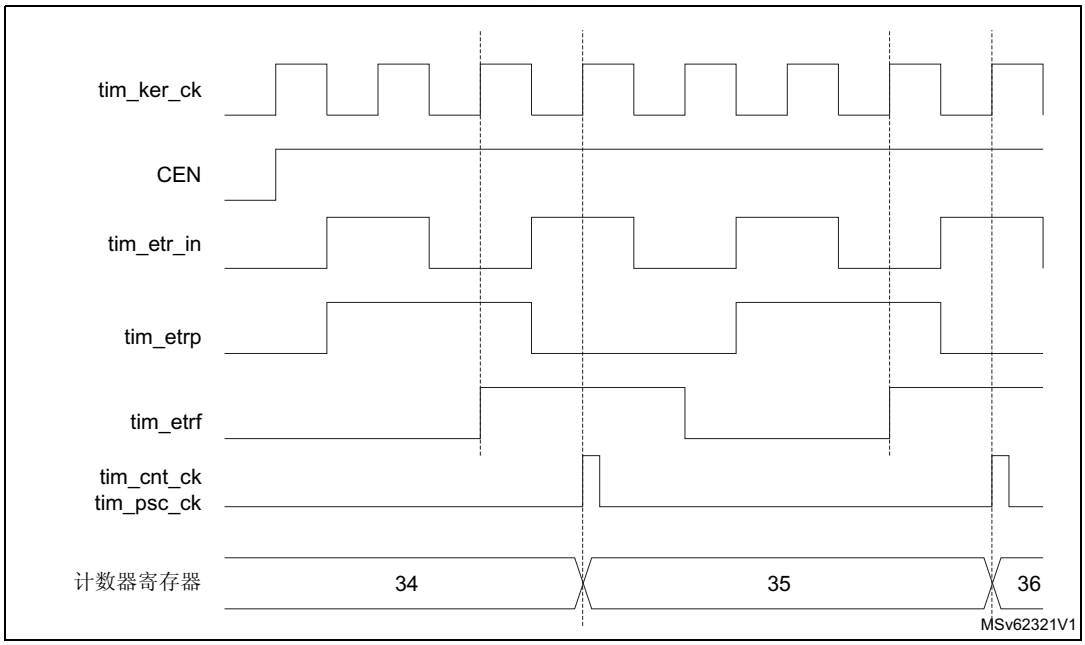
例如，要使递增计数器在 tim_etr_in 每出现 2 个上升沿时计数，请执行以下步骤：

1. 由于此例中不需滤波器，因此在 TIMx_SMCR 寄存器中写入 ETF[3:0]=0000。
2. 通过在 TIMx_SMCR 寄存器中写入 ETPS[1:0]=01 来设置预分频器
3. 通过在 TIMx_SMCR 寄存器中写入 ETP = 0 来选择 tim_etr_in 输入的上升沿检测
4. 通过在 TIMx_SMCR 寄存器中写入 ECE=1 来使能外部时钟模式 2。
5. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

tim_etr_in 每出现 2 个上升沿，计数器计数一次。

tim_etr_in 的上升沿与实际计数器时钟之间的延迟是由于 tim_etrp 信号的重新同步电路引起的。因此，计数器能够正确捕获的最大频率不超过 tim_ker_ck 频率的 ¼。当 ETRP 信号较快时，用户必须通过适当的 ETPS 预分频器设置对外部信号进行分频。

图 187. 外部时钟模式 2 下的控制电路



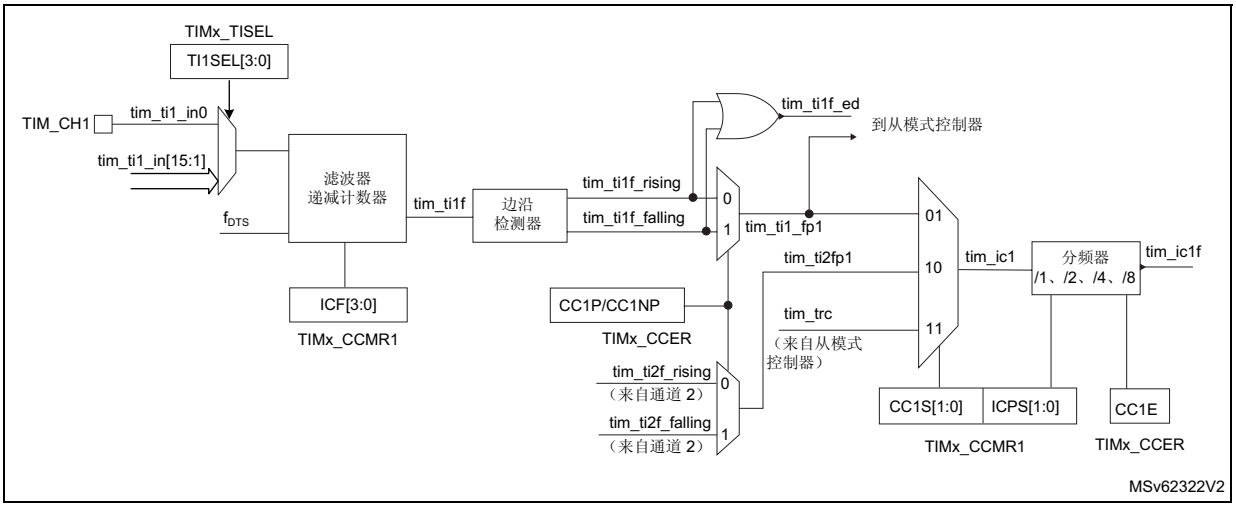
26.3.8 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入阶段（数字滤波、多路复用和预分频器，通道 5 和通道 6 除外）和一个输出阶段（比较器和输出控制）构建而成。

图 188 到*图 191* 简要介绍了一个捕获/比较通道。

输入阶段对相应的 tim_tix 输入进行采样，生成一个滤波后的信号 tim_tixf。然后，带有极性选择功能的边沿检测器生成一个信号 (tim_tixfpy)，该信号可用作从模式控制器的触发输入，也可用作捕获命令。该信号先进行预分频 (ICxPS)，而后再进入捕获寄存器。

图 188. 捕获/比较通道（示例：通道 1 输入阶段）



输出阶段生成一个中间波形作为基准：tim_ocxref（高电平有效）。链的末端决定最终输出信号的极性。

图 189. 捕获/比较通道 1 主电路

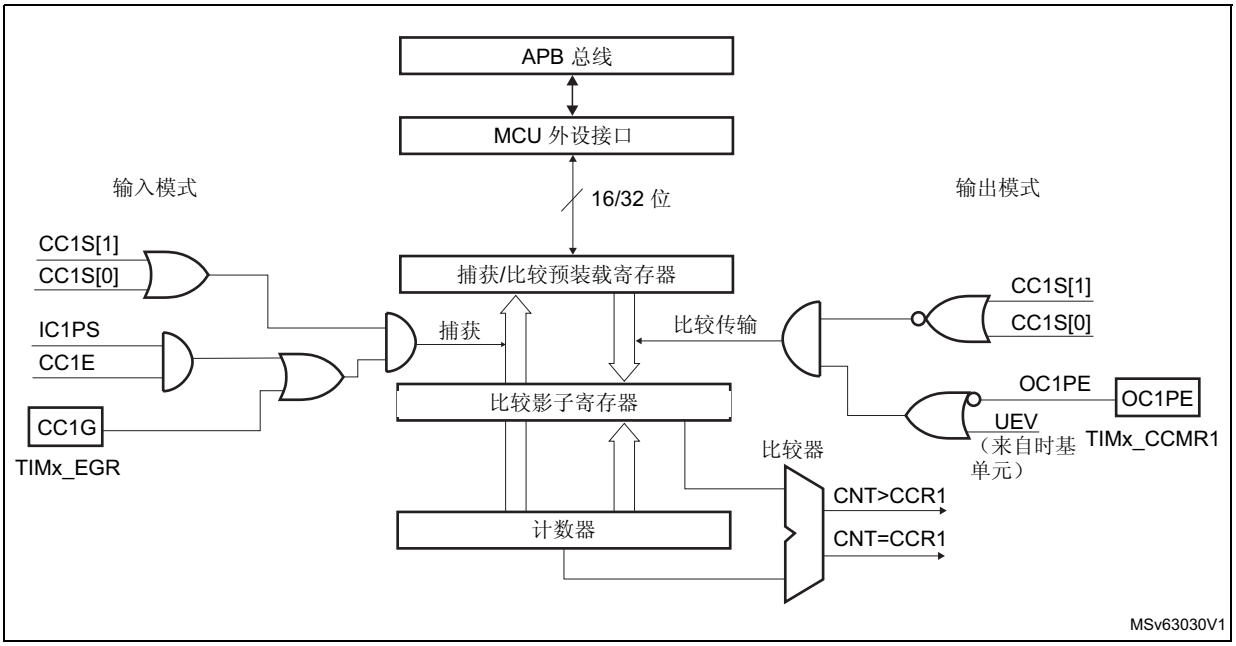
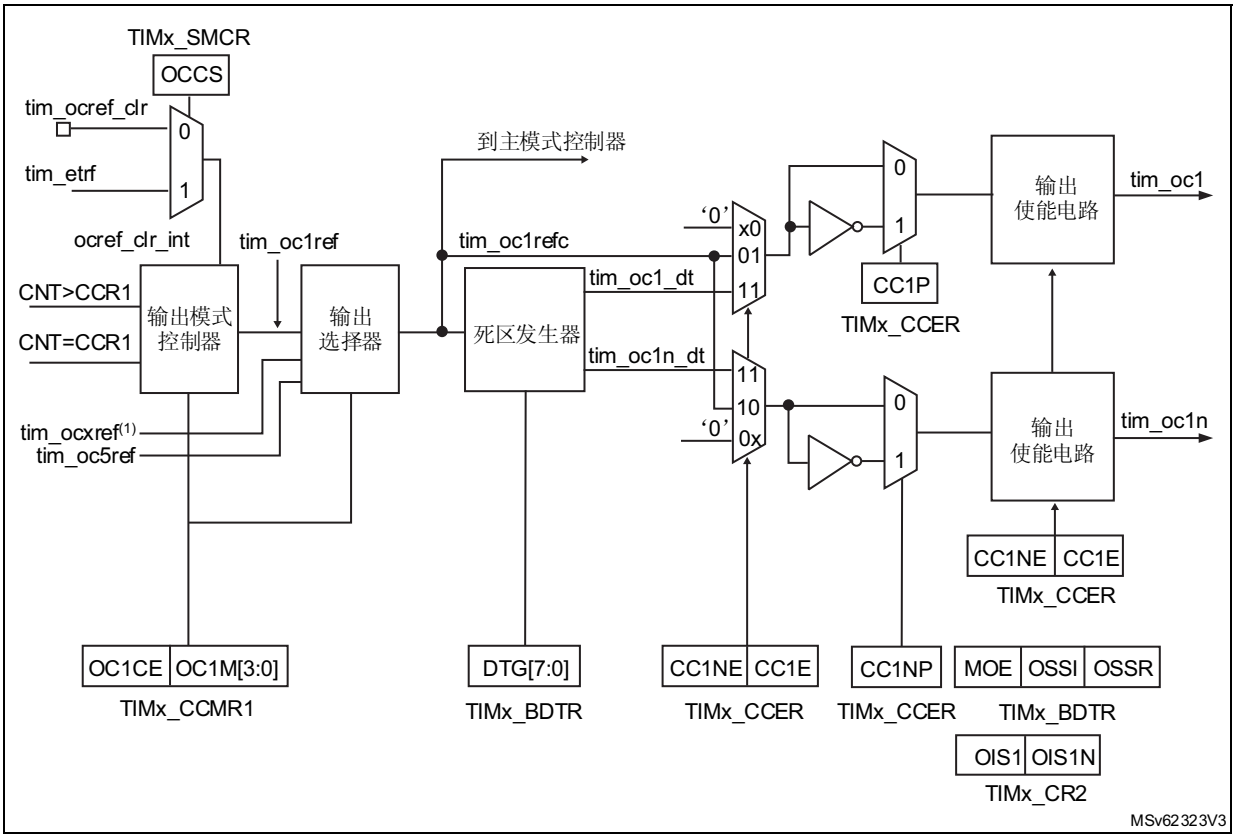
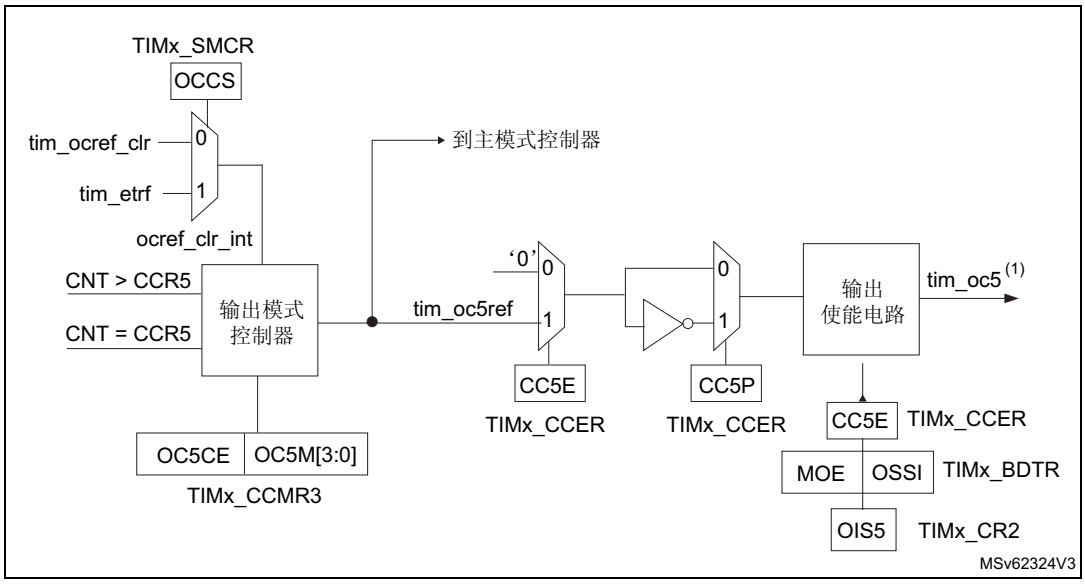


图 190. 捕获/比较通道的输出阶段（通道 1、通道 2、通道 3 和通道 4）



1. tim_ocxref，其中 x 为互补通道的序号

图 191. 捕获/比较通道的输出阶段（通道 5 和通道 6）



1. 不适用于外部。

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。读写访问的始终是预装载寄存器。

在捕获模式下，捕获实际发生在影子寄存器中，然后将影子寄存器的内容复制到预装载寄存器中。

在比较模式下，预装载寄存器的内容将复制到影子寄存器中，然后将影子寄存器的内容与计数器进行比较。

26.3.9 输入捕获模式

在输入捕获模式下，当相应的 ICx 信号检测到跳变沿后，将使用捕获/比较寄存器 (TIMx_CCRx) 来锁存计数器的值。发生捕获事件时，会将相应的 CCXIF 标志（TIMx_SR 寄存器）置 1，并可发送中断或 DMA 请求（如果已使能）。如果发生捕获事件时 CCxIF 标志已处于高位，则会将过捕获标志 CCxOF（TIMx_SR 寄存器）置 1。可通过软件方法向 CCxIF 写入 0 来给 CCxIF 清零，或读取存储在 TIMx_CCRx 寄存器中的已捕获数据。向 CCxOF 写入 0 后会将其清零。

以下示例说明了如何在 tim_ti1 输入出现上升沿时将计数器的值捕获到 TIMx_CCR1 中。具体操作步骤如下：

- 选择有效输入：TIMx_CCR1 必须关联到 tim_ti1 输入，因此向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01。只要 CC1S 不等于 00，就会将通道配置为输入模式，并且 TIMx_CCR1 寄存器将处于只读状态。
- 根据关联到定时器的信号，对相关输入滤波参数进行编程（如果输入为 tim_tix 之一，则对 TIMx_CCMRx 寄存器中的 ICxF 位进行编程）。假设输入信号发生翻转时，信号需要 5 个内部时钟周期才能稳定，则必须设置滤波时间大于 5 个内部时钟周期。若在 tim_ti1 上连续 8 次检测到新电平采样值（以 f_{DTS} 频率采样）判定沿跳变合法，则向 TIMx_CCMR1 寄存器中的 IC1F 位写入 0011。
- 通过在 TIMx_CCER 寄存器中将 CC1P 位和 CC1NP 位写入 0，选择 tim_ti1 上的有效转换边沿（本例中为上升沿）。
- 对输入预分频器进行编程。在本例中，我们希望每次有效转换时都执行捕获操作，因此需要禁止预分频器（向 TIMx_CCMR1 寄存器中的 IC1PS 位写入“00”）。
- 通过将 TIMx_CCER 寄存器中的 CC1E 位置 1，允许将计数器的值捕获到捕获寄存器中。
- 如果需要，可通过将 TIMx_DIER 寄存器中的 CC1IE 位置 1 来使能相关中断请求，并且/或者通过将该寄存器中的 CC1DE 位置 1 来使能 DMA 请求。

发生输入捕获时：

- 发生有效跳变沿时，TIMx_CCR1 寄存器会获取计数器的值。
- 将 CC1IF 标志置 1（中断标志）。如果至少发生了两次连续捕获，但 CC1IF 标志未被清零，这样 CC1OF 捕获溢出标志会被置 1。
- 根据 CC1IE 位生成中断。
- 根据 CC1DE 位生成 DMA 请求。

要处理过捕获，建议在读取捕获溢出标志之前读取数据。这样可避免丢失在读取捕获溢出标志之后与读取数据之前可能出现的过捕获信息。

注意： *通过软件将 TIMx_EGR 寄存器中的相应 CCxG 位置 1 可生成 IC 中断和/或 DMA 请求。*

26.3.10 PWM 输入模式

该模式用于测量连接到单个 tim_tix 输入的 PWM 信号的周期和占空比：

- TIMx_CCR1 寄存器保存周期值（相邻两个上升沿之间的时间间隔）
- TIM_CCR2 寄存器保存脉宽（相邻的上升沿与下降沿之间的时间间隔）

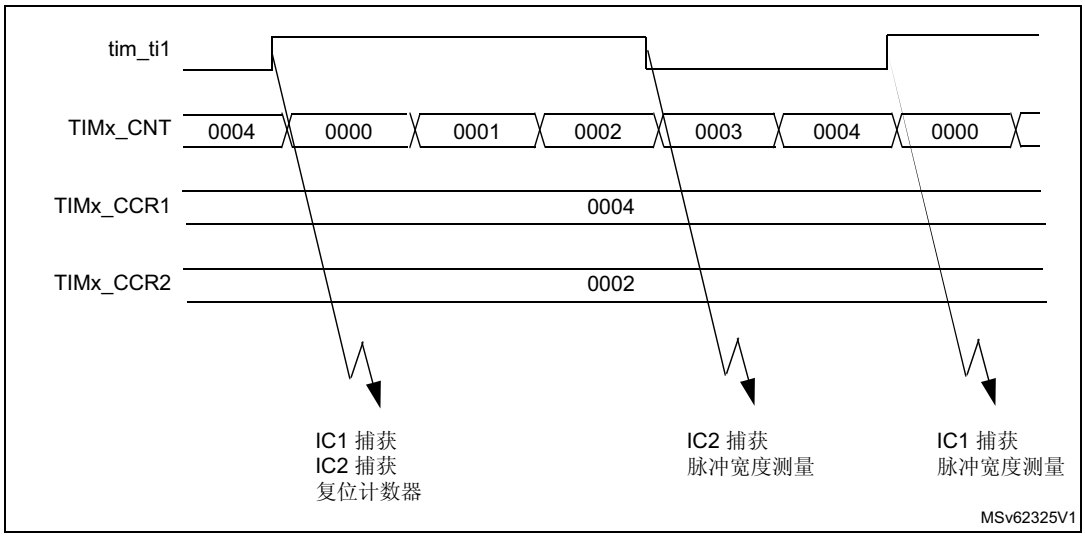
此模式是输入捕获模式的一个特例。设置步骤与之类似，但存在以下区别：

- 两个 ICx 信号被映射至同一个 tim_tixfp1 输入。
- 这两个 ICx 信号在边沿处有效，但极性相反。
- 选择两个 tim_tixfp 信号之一作为触发输入，并将从模式控制器配置为复位模式。

应用于 tim_ti1 的 PWM 信号的周期和脉宽可通过以下步骤测量：

- 选择 TIMx_CCR1 的有效输入：向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01（选择 tim_ti1）。
- 选择 tim_ti1fp1 的有效极性（用于在 TIMx_CCR1 中捕获和计数器清零）：向 CC1P 位和 CC1NP 位写入“0”（上升沿有效）。
- 选择 TIMx_CCR2 的有效输入：向 TIMx_CCMR1 寄存器中的 CC2S 写入 10（选择 tim_ti1）。
- 选择 tim_ti1fp2 的有效极性（用于在 TIMx_CCR2 中捕获）：向 CC2P 位和 CC2NP 位写入 CC2P/CC2NP =“10”（下降沿有效）。
- 选择有效触发输入：向 TIMx_SMCR 寄存器中的 TS 位写入 00101（选择 tim_ti1fp1）。
- 将从模式控制器配置为复位模式：向 TIMx_SMCR 寄存器中的 SMS 位写入 0100。
- 使能捕获：向 TIMx_CCER 寄存器中的 CC1E 位和 CC2E 位写入“1”。

图 192. PWM 输入模式时序



26.3.11 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中的 CCxS 位 = 00）下，可直接由软件将每个输出比较信号（tim_ocxref 和 tim_ocx/tim_ocxn）强制设置为有效电平或无效电平，而无需考虑输出比较寄存器和计数器之间的任何比较结果。

要将输出比较信号 (tim_ocxref/tim_ocx) 强制设置为有效电平，用户只需向相应 TIMx_CCMRx 寄存器中的 OCxM 位写入 0101。tim_ocxref 进而强制设置为高电平（tim_ocxref 始终为高电平有效），同时 tim_ocx 获取 CCxP 极性位的相反值。

例如：CCxP = 0（tim_ocx 高电平有效）=> tim_ocx 强制设置为高电平。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入 0100，可将 tim_ocxref 信号强制设置为低电平。

当然，即使在强制输出模式下，TIMx_CCRx 影子寄存器与计数器之间的比较仍然会执行，而且允许将标志置 1。因此可发送相应的中断和 DMA 请求。下面的输出比较模式一节对此进行了介绍。

26.3.12 输出比较模式

此功能用于控制输出波形，或指示已经过某一时间段。通道 1 到通道 4 可用作输出，而通道 5 和通道 6 只能在单片机内部使用（例如，用于产生混合波形或触发 ADC）。

当捕获/比较寄存器与计数器之间相匹配时，输出比较功能：

- 将为相应的输出引脚分配一个可编程值，该值由输出比较模式（TIMx_CCMRx 寄存器中的 OCxM 位）和输出极性（TIMx_CCER 寄存器中的 CCxP 位）定义。匹配时，输出引脚既可保持其电平 (OCxM=0000)，也可设置为有效电平 (OCxM=0001)、无效电平 (OCxM=0010) 或进行翻转 (OCxM=0011)。
- 将中断状态寄存器中的标志置 1（TIMx_SR 寄存器中的 CCxIF 位）。
- 如果相应中断使能位（TIMx_DIER 寄存器中的 CCXIE 位）置 1，将生成中断。
- 如果相应使能位（TIMx_DIER 寄存器的 CCxDE 位，TIMx_CR2 寄存器的 CCDS 位，用来选择 DMA 请求）置 1，将发送 DMA 请求。

使用 TIMx_CCMRx 寄存器中的 OCxPE 位，可将 TIMx_CCRx 寄存器配置为带或不带预装功能。

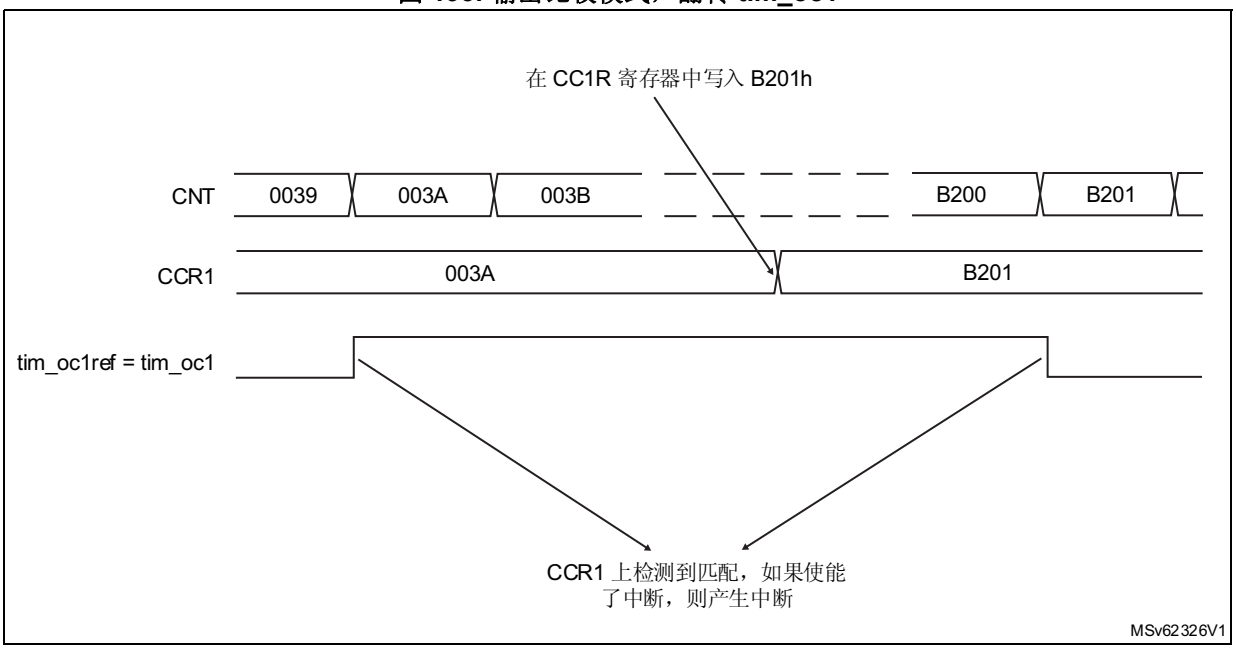
在输出比较模式下，更新事件 UEV 对 tim_ocxref 和 tim_ocx 输出毫无影响。时间分辨率可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

步骤

1. 选择计数器时钟（内部、外部、预分频器）。
2. 在 TIMx_ARR 和 TIMx_CCRx 寄存器中写入所需数据。
3. 如果要生成中断请求，则需将 CCxIE 位置 1。
4. 选择输出模式。例如：
   - 当 CNT 与 CCRx 匹配时，写入 OCxM = 0011 以翻转 tim_ocx 输出引脚
   - 写入 OCxPE = 0 以禁止预装功能
   - 写入 CCxP = 0 以选择高电平有效极性
   - 写入 CCxE = 1 以使能输出
5. 通过将 TIMx_CR1 寄存器中的 CEN 位置 1 来使能计数器。

可随时通过软件更新 TIMx_CCRx 寄存器以控制输出波形，前提是未使能预装功能（OCxPE=0，否则仅当发生下一个更新事件 UEV 时，才会更新 TIMx_CCRx 影子寄存器）。*图 193* 给出了一个示例。

图 193. 输出比较模式，翻转 tim_oc1



26.3.13 PWM 模式

脉宽调制模式用于生成信号，该信号频率由 TIMx_ARR 寄存器值决定，其占空比则由 TIMx_CCRx 寄存器值决定。

各通道可以独立选择 PWM 模式（每个 tim_ocx 输出对应一个 PWM），只需向 TIMx_CCMRx 寄存器的 OCxM 位写入“0110”（PWM 模式 1）或“0111”（PWM 模式 2）。必须通过将 TIMx_CCMRx 寄存器中的 OCxPE 位置 1 使能相应预装载寄存器，最后通过将 TIMx_CR1 寄存器中的 ARPE 位置 1 使能自动重载预装载寄存器（在递增计数或中心对齐模式下）。

由于只有在发生更新事件时预装载寄存器内容才会传送到影子寄存器，因此启动计数器之前，必须通过将 TIMx_EGR 寄存器中的 UG 位置 1 来初始化所有寄存器。

tim_ocx 极性可通过软件来编程（使用 TIMx_CCER 寄存器的 CCxP 位）。可将其编程为高电平有效或低电平有效。通过 CCxE、CCxNE、MOE、OSSI 和 OSSR 位（TIMx_CCER 和 TIMx_BDTR 寄存器）的组合使能 tim_ocx 输出。有关详细信息，请参见 TIMx_CCER 寄存器说明。

在 PWM 模式（1 或 2）下，TIMx_CNT 总是与 TIMx_CCRx 进行比较，以确定是 TIMx_CCRx ≤ TIMx_CNT 还是 TIMx_CNT ≤ TIMx_CCRx（取决于计数器计数方向）。

根据 TIMx_CR1 寄存器中的 CMS 位状态，定时器能够产生边沿对齐模式或中心对齐模式的 PWM 信号。

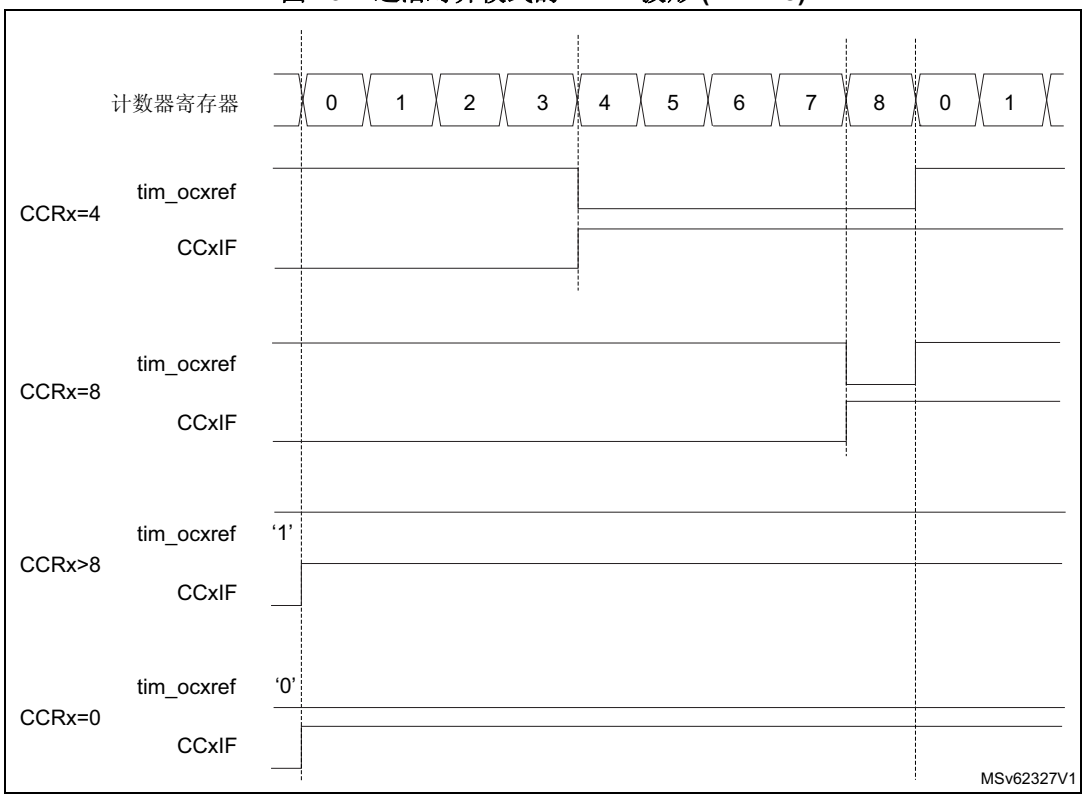
PWM 边沿对齐模式

- 递增计数配置

  当 TIMx_CR1 寄存器中的 DIR 位为低时执行递增计数。请参见*第 697 页的递增计数模式*。

  以下以 PWM 模式 1 为例。只要 TIMx_CNT < TIMx_CCRx，PWM 参考信号 tim_ocxref 便为高电平，否则为低电平。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR 中），则 tim_ocxref 保持为“1”。如果比较值为 0，则 tim_ocxref 保持为“0”。*图 194* 举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

图 194. 边沿对齐模式的 PWM 波形 (ARR=8)



- 递减计数配置

  当 TIMx_CR1 寄存器中的 DIR 位为高时执行递减计数。请参见*第 700 页的递减计数模式*。

  在 PWM 模式 1 下，只要 TIMx_CNT > TIMx_CCRx，参考信号 tim_ocxref 即为低电平，否则其为高电平。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR 中），则 tim_ocxref 保持为“1”。此模式下不可能产生 0% 的 PWM 波形。

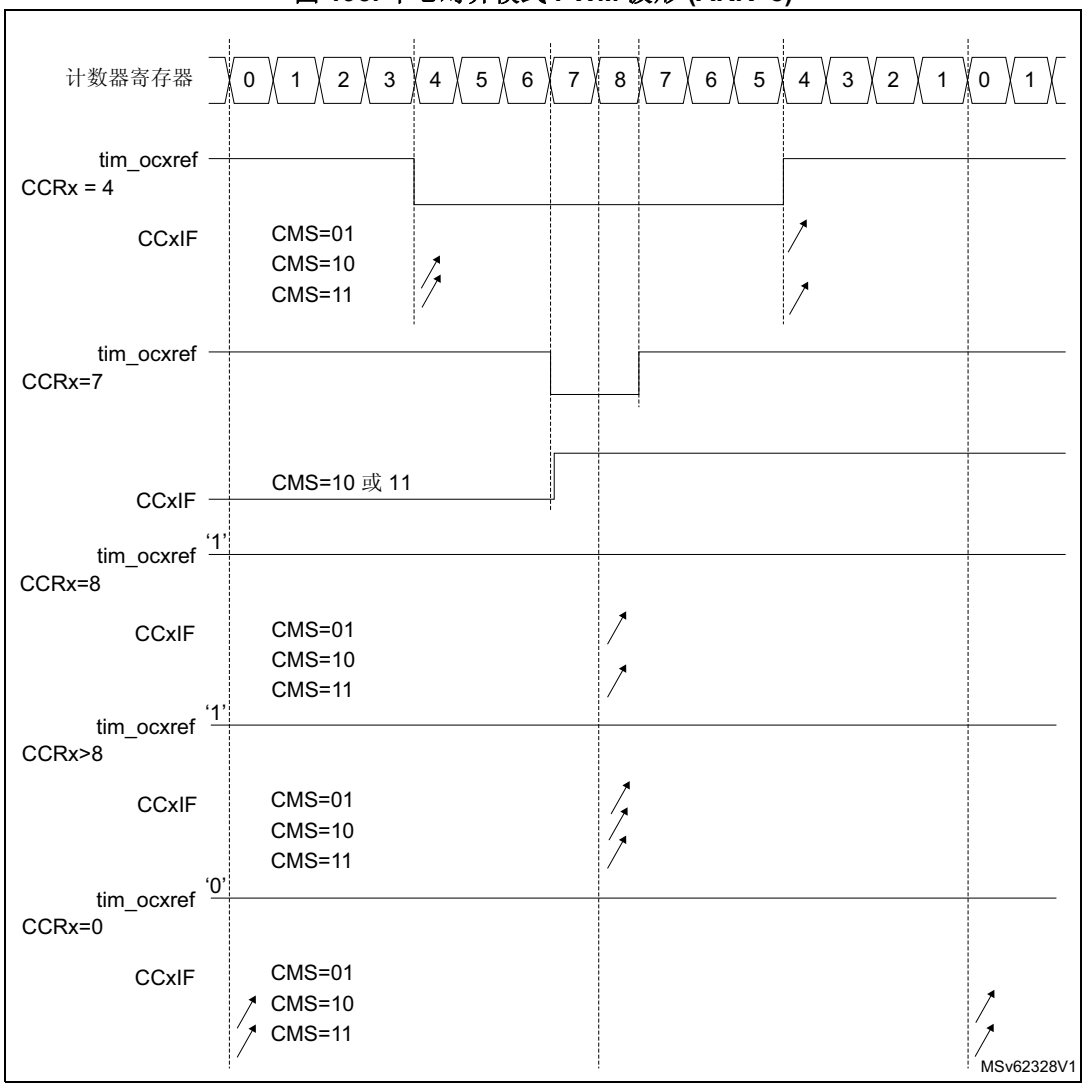
PWM 中心对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时（其余所有配置对 tim_ocxref/tim_ocx 信号具有相同的作用），中心对齐模式生效。根据 CMS 位的配置，可以在计数器递增计数、递减计数或同时递增和递减计数时将比较标志置 1。TIMx_CR1 寄存器中的方向位 (DIR) 由硬件更新，不得通过软件更改。请参见*第 703 页的中心对齐模式（递增/递减计数）*。

图 195 显示了中心对齐模式的 PWM 波形，在此例中：

- TIMx_ARR=8，
- PWM 模式为 PWM 模式 1，
- 在根据 TIMx_CR1 寄存器中 CMS=01 而选择的中心对齐模式 1 下，当计数器递减计数时，比较标志置 1。

图 195. 中心对齐模式 PWM 波形 (ARR=8)



中心对齐模式使用建议：

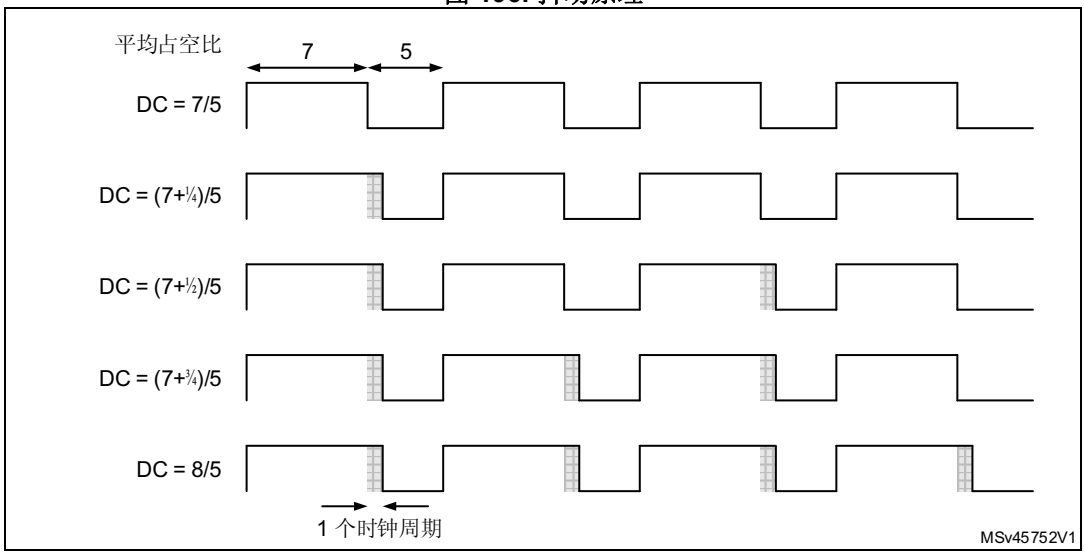
- 启动中心对齐模式时将使用当前的递增 / 递减计数配置。这意味着计数器将根据写入 TIMx_CR1 寄存器中 DIR 位的值进行递增或递减计数。此外，不得通过软件同时修改 DIR 和 CMS 位。
- 不建议在中心对齐模式下对计数器执行写操作，否则将发生意外错误。尤其是：
  - 如果写入计数器的值大于自动重载值 (TIMx_CNT > TIMx_ARR)，计数方向不会更新。例如，如果计数器之前递增计数，则继续递增计数。
  - 如果向计数器写入 0 或 TIMx_ARR 的值，计数方向会更新，但不生成更新事件 UEV。
- 使用中心对齐模式最为保险的方法是：在启动计数器前通过软件生成更新（将 TIMx_EGR 寄存器中的 UG 位置 1），并且不要在计数器运行过程中对其执行写操作。

抖动模式

使用 TIMx_CR1 寄存器中的 DITHEN 位使能抖动模式，可以提高 PWM 模式的有效分辨率。这适用于 CCR（提高占空比分辨率）和 ARR（提高 PWM 频率分辨率）。

工作原理如下：在 16 个连续的 PWM 周期内按照预定义模式略微更改（增加或不增加一个定时器时钟周期）实际的 CCR（或 ARR）值。考虑到平均占空比或 PWM 周期，这样可以将分辨率提高 16 倍。*图 196* 介绍了应用于 4 个连续 PWM 周期的抖动原理。

图 196. 抖动原理

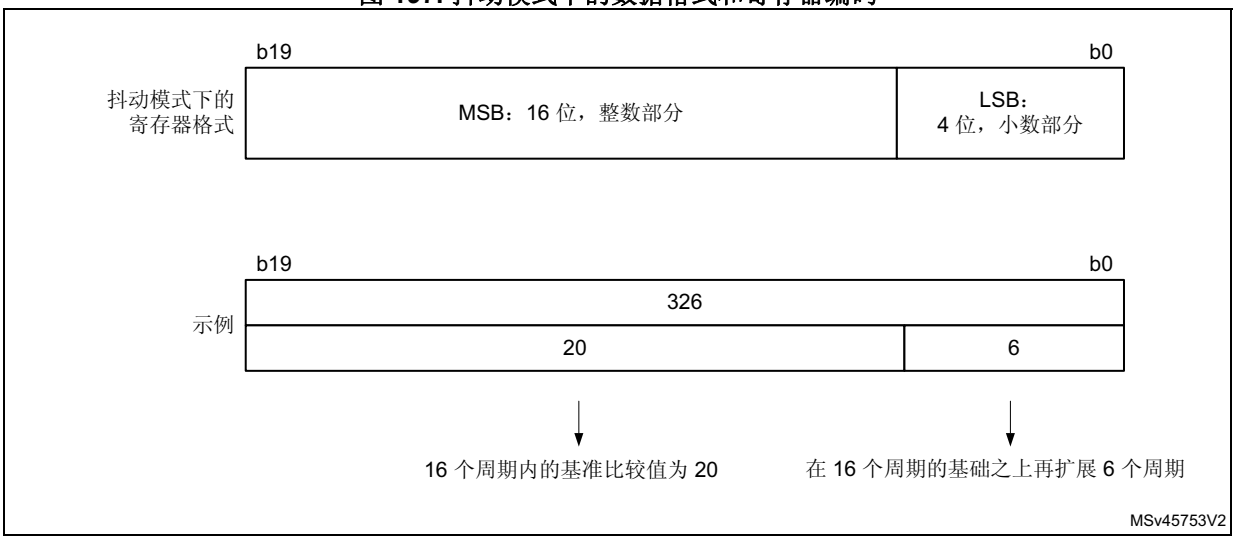


使能抖动模式后，寄存器编码的变化如下（请参见*图 197* 获取示例）：

- 4 个 LSB 用于对提高的分辨率部分（小数部分）进行编码
- MSB 左移至位 19:4，用于对基值进行编码

注意： *复位 DITHEN 位时必须遵循以下顺序：*
1. CEN 位和 ARPE 位必须复位
2. DITHEN 位必须复位
3. CCIF 标志必须清零
4. CEN 位可以置 1（最终 ARPE = 1）。

图 197. 抖动模式下的数据格式和寄存器编码



最低频率可根据以下公式得出：

$$\text{分辨率} = \frac{F_{Tim}}{F_{pwm}} \Rightarrow F_{pwmMin} = \frac{F_{Tim}}{Max_{Resolution}}$$

$$\text{禁止抖动模式：} F_{pwmMin} = \frac{F_{Tim}}{65536}$$

$$\text{使能抖动模式：} F_{pwmMin} = \frac{F_{Tim}}{65535 + \frac{15}{16}}$$

注意： *在抖动模式下，最大 TIMx_ARR 和 TIMxCCRy 值限制为 0xFFFEF，该值对应于 65534（整数部分）和 15（抖动部分）。*

如下面的*图 198* 所示，无论 PWM 频率如何，抖动模式都可以用于提高 PWM 分辨率。

图 198. PWM 分辨率与频率的关系

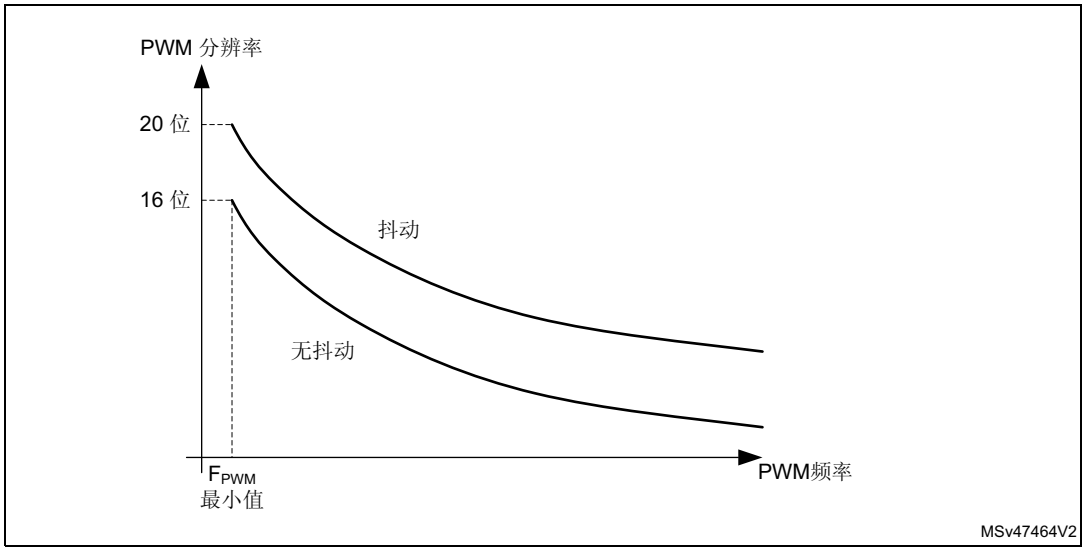
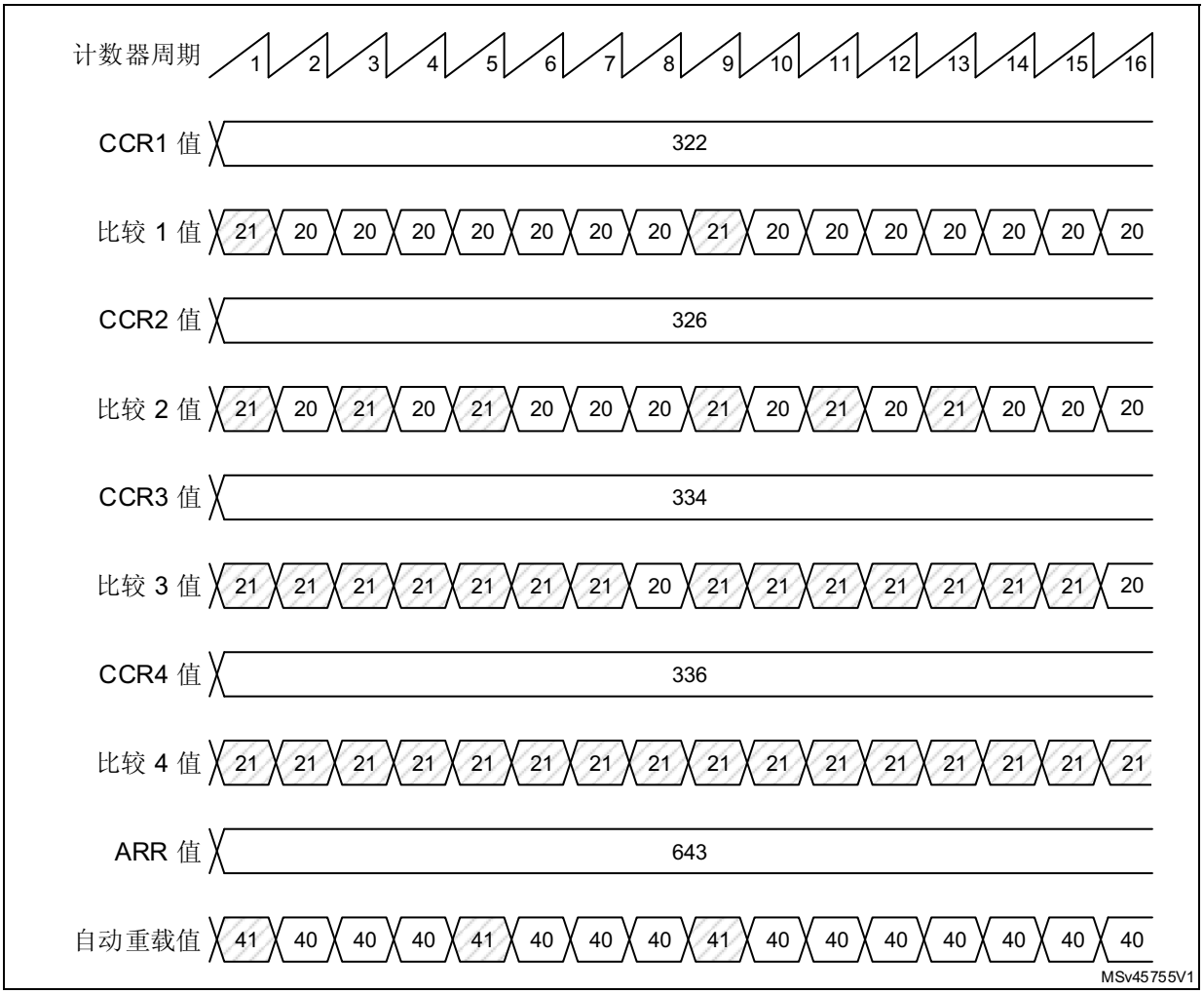


图 199 详细描述了占空比和/或周期在 16 个连续周期内的变化情况。

图 199. PWM 抖动模式



自动重载和比较值增量在特定模式下的分布情况如下面的*表 173* 所示。抖动序列的目的是尽可能均匀地分布增量并将整体纹波降至最低。

表 173. CCR 和 ARR 寄存器变化抖动模式

LSB 值	PWM 周期															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-

表 173. CCR 和 ARR 寄存器变化抖动模式（续）

LSB 值	PWM 周期															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

抖动模式也适用于中心对齐 PWM 模式（TIMx_CR1 寄存器中的 CMS 位不等于“00”）。在这种情况下，考虑到递增和递减计数阶段，可在 8 个连续的 PWM 周期内应用抖动模式，如下面的*图 200* 所示。

图 200. 中心对齐 PWM 模式下的抖动对占空比的影响

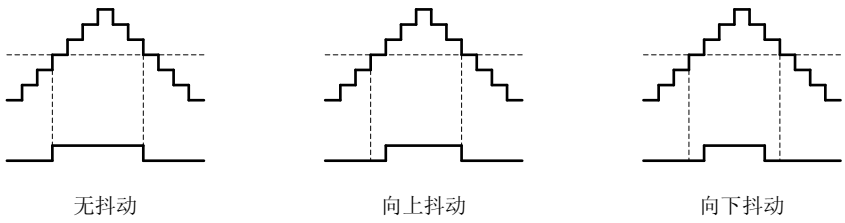


表 174 显示了如何在中心对齐 PWM 模式下应用抖动模式。

表 174. 中心对齐 PWM 模式下的 CCR 寄存器变化抖动模式

LSB 值	PWM 周期															
	1		2		3		4		5		6		7		8	
	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-

表 174. 中心对齐 PWM 模式下的 CCR 寄存器变化抖动模式（续）

LSB 值	PWM 周期															
	1		2		3		4		5		6		7		8	
	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

26.3.14 不对称 PWM 模式

在不对称模式下，生成的两个中心对齐 PWM 信号间允许存在可编程相移。频率由 TIMx_ARR 寄存器的值确定，而占空比和相移则由一对 TIMx_CCRx 寄存器确定。两个寄存器分别控制递增计数和递减计数期间的 PWM，这样每半个 PWM 周期便会调节一次 PWM：

- tim_oc1refc（或 tim_oc2refc）由 TIMx_CCR1 和 TIMx_CCR2 控制
- tim_oc3refc（或 tim_oc4refc）由 TIMx_CCR3 和 TIMx_CCR4 控制

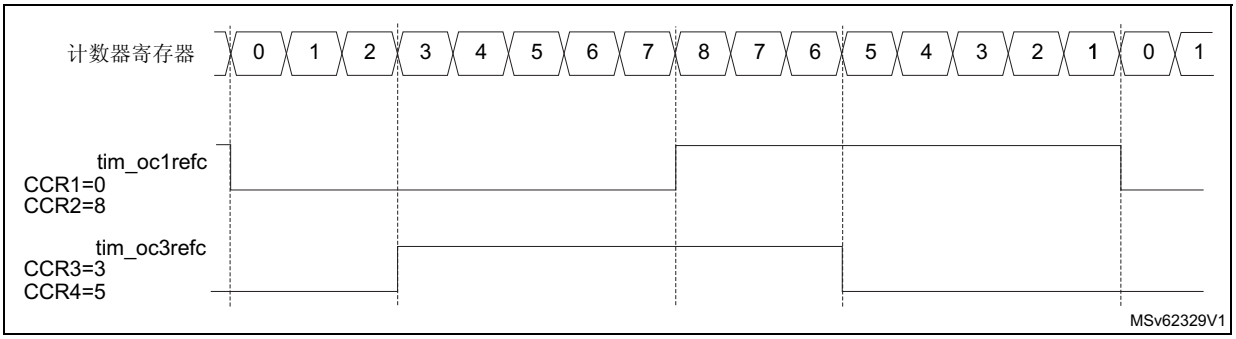
两个通道可以独立选择不对称 PWM 模式（每对 CCR 寄存器一个 tim_ocx 输出），只需向 TIMx_CCMRx 寄存器的 OCxM 位写入“1110”（不对称 PWM 模式 1）或“1111”（不对称 PWM 模式 2）。

注意： *出于兼容性原因，OCxM[3:0] 位域分为两部分，最高有效位与最低有效的 3 位不相邻。*

给定通道用作不对称 PWM 通道时，也可使用其互补通道。例如，如果通道 1 上产生 tim_oc1refc 信号（不对称 PWM 模式 1），则由于不对称 PWM 模式 1 的原因，通道 2 上可输出 tim_oc2ref 信号或 tim_oc2refc 信号。

图 201 显示了不对称 PWM 模式下可以产生的信号示例（通道 1 到通道 4 配置为不对称 PWM 模式 2）。与死区发生器配合使用时，这可控制相移全桥直流到直流转换器。

图 201. 50% 占空比时产生的 2 个相移 PWM 信号



26.3.15 组合 PWM 模式

在组合 PWM 模式下，生成的两个边沿或中心对齐 PWM 信号的各个脉冲间允许存在可编程延时和相移。频率由 TIMx_ARR 寄存器的值确定，而占空比和延时则由两个 TIMx_CCRx 寄存器确定。产生的信号 tim_ocxrefc 由两个参考 PWM 的“逻辑或”运算或者“逻辑与”运算组合组成：

- tim_oc1refc（或 tim_oc2refc）由 TIMx_CCR1 和 TIMx_CCR2 控制
- tim_oc3refc（或 tim_oc4refc）由 TIMx_CCR3 和 TIMx_CCR4 控制

两个通道可以独立选择组合 PWM 模式（每对 CCR 寄存器一个 tim_ocx 输出），只需向 TIMx_CCMRx 寄存器的 OCxM 位写入“1100”（组合 PWM 模式 1）或“1101”（组合 PWM 模式 2）。

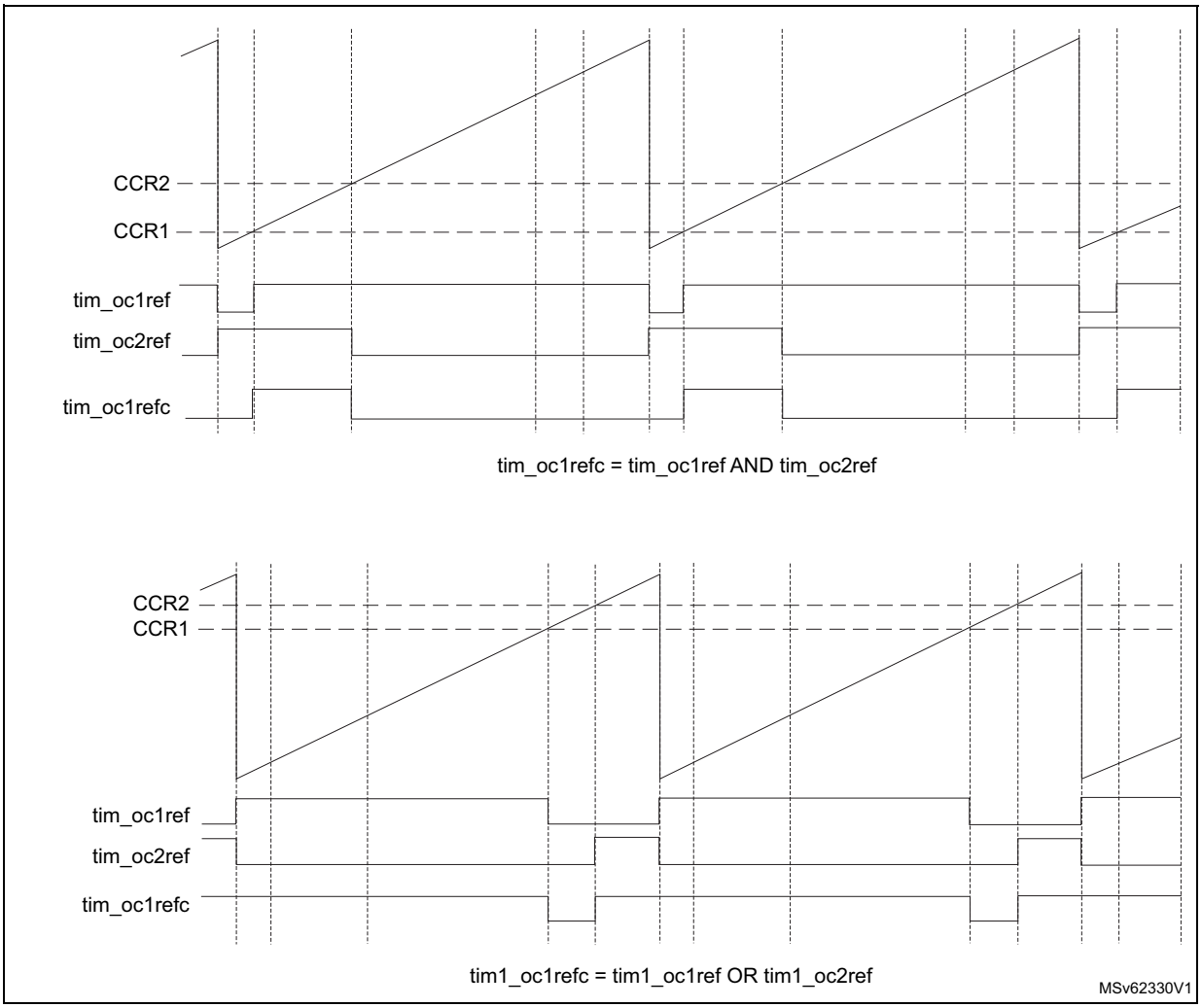
当给定通道用作组合 PWM 通道时，其互补通道必须在相反的 PWM 模式下配置（例如，一个通道在组合 PWM 模式 1 下配置，另一个通道在组合 PWM 模式 2 下配置）。

注意： *出于兼容性原因，OCxM[3:0] 位域分为两部分，最高有效位与最低有效的 3 位不相邻。*

图 202 显示了组合 PWM 模式下可以产生的信号示例，通过以下配置可获得这些信号：

- 通道 1 配置为组合 PWM 模式 2，
- 通道 2 配置为 PWM 模式 1，
- 通道 3 配置为组合 PWM 模式 2，
- 通道 4 在 PWM 模式 1 下配置。

图 202. 通道 1 和通道 3 上的组合 PWM 模式



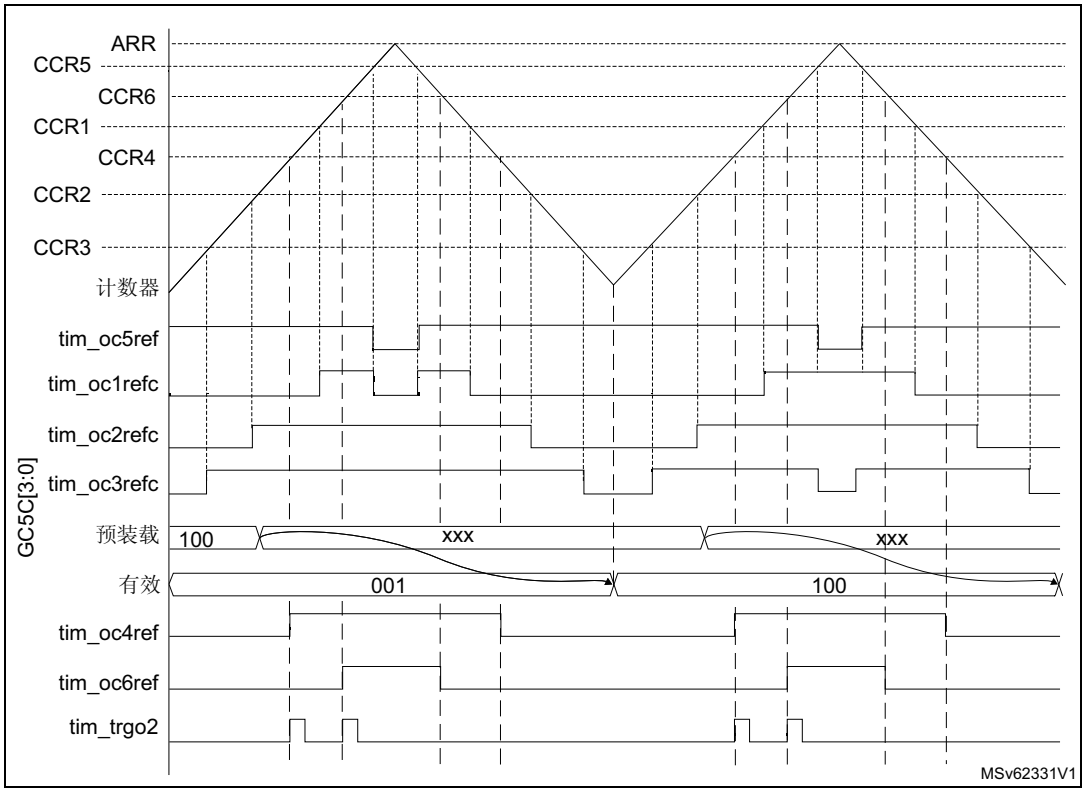
26.3.16 组合三相 PWM 模式

在组合三相 PWM 模式下，产生的一至三个中心对齐 PWM 信号与一个可编程信号间允许在脉冲中间进行逻辑与运算。tim_oc5ref 信号用于定义产生的组合信号。通过 TIMx_CCR5 中的 3 位 GC5C[3:1]，可以选择 tim_oc5ref 与哪个参考信号组合。产生的信号 tim_ocxrefc 由两个参考 PWM 的逻辑与运算组合组成：

- 如果 GC5C1 置 1，则 tim_oc1refc 由 TIMx_CCR1 和 TIMx_CCR5 控制
- 如果 GC5C2 置 1，则 tim_oc2refc 由 TIMx_CCR2 和 TIMx_CCR5 控制
- 如果 GC5C3 置 1，则 tim_oc3refc 由 TIMx_CCR3 和 TIMx_CCR5 控制

通道 1 到通道 3 可独立选择组合三相 PWM 模式，只需将 3 位 GC5C[3:1] 中的至少一位置 1。

图 203. 三相组合 PWM 信号（每个周期多个触发脉冲）



tim_trgo2 波形说明了如何根据给定的三相 PWM 信号同步 ADC。更多详细信息，请参见*第 26.3.31 节：ADC 触发*。

26.3.17 互补输出和死区插入

高级控制定时器 (TIM1) 可以输出两路互补信号，并管理输出的关断与接通瞬间。

这段时间通常称为死区，用户必须根据与输出相连接的器件及其特性（电平转换器的固有延迟、开关器件产生的延迟...）来调整死区时间。

每路输出可以独立选择输出极性（主输出 tim_ocx 或互补输出 tim_ocxn）。可通过对 TIMx_CCER 寄存器中的 CCxP 和 CCxNP 位执行写操作来完成极性选择。

互补信号 tim_ocx 和 tim_ocxn 通过以下多个控制位的组合进行激活：TIMx_CCER 寄存器中的 CCxE 和 CCxNE 位以及 TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位。更多详细信息，请参见*第 795 页的表 182：具有刹车功能的互补通道 tim_ocx 和 tim_ocxn 的输出控制位*。应当注意，切换至空闲状态（MOE 位变为 0）的时刻，死区仍然有效。

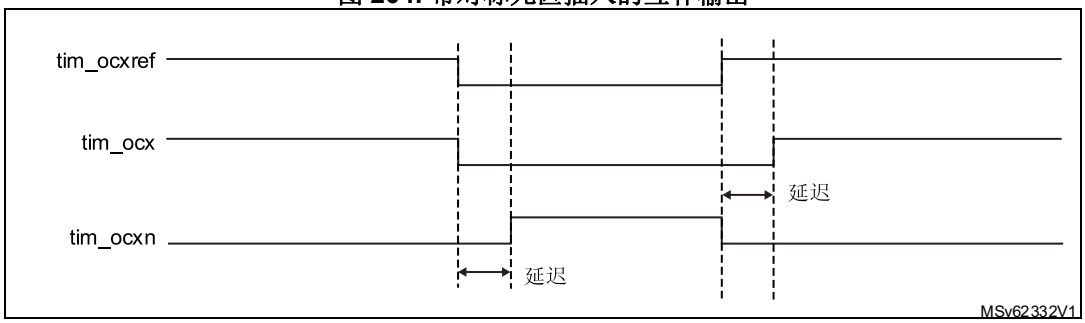
CCxE 和 CCxNE 位同时置 1 并且 MOE 位置 1（若存在刹车电路）时，将使能死区插入。每个通道有一个 10 位死区发生器。将基于参考波形 tim_ocxref 生成 2 个输出 tim_ocx 和 tim_ocxn。如果 tim_ocx 和 tim_ocxn 为高电平有效：

- 输出信号 tim_ocx 与参考信号相同，只是其上升沿相对参考上升沿存在延迟。
- 输出信号 tim_ocxn 与参考信号相反，并且其上升沿相对参考下降沿存在延迟。

如果延迟时间大于有效输出（tim_ocx 或 tim_ocxn）的宽度，则不会产生相应的脉冲。

下图所示为死区发生器的输出信号与参考信号 tim_ocxref 之间的关系。（在这些示例中，假定 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1）

图 204. 带对称死区插入的互补输出



TIMx_DTR2 中的 DTAE 位用于区分参考信号上升沿与下降沿的死区值，如*图 205* 所示。

在不对称模式下 (DTAE = 1)，上升沿参考死区由 TIMx_BDTR 寄存器中的 DTG[7:0] 位域定义，下降沿参考死区由 TIMx_DTR2 寄存器中的 DTGF[7:0] 位域定义。DTAE 位必须在计数器使能之前写入，并且在 CEN = 1 时不得修改。

可以使用预装载机制在 PWM 操作期间实时更新死区值。当 TIMX_DTR2 寄存器中的 DTPE 位置 1 时，将对死区位域 DTG[7:0] 和 DTGF[7:0] 进行预装载。预装载值会在发生下一更新事件时装载到影子寄存器中。

注意： *如果在计数器处于使能状态时将 DTPE 位置 1，则自上一次更新以来写入的任何新值都会被丢弃，而继续使用之前的值。*

图 205. 非对称死区

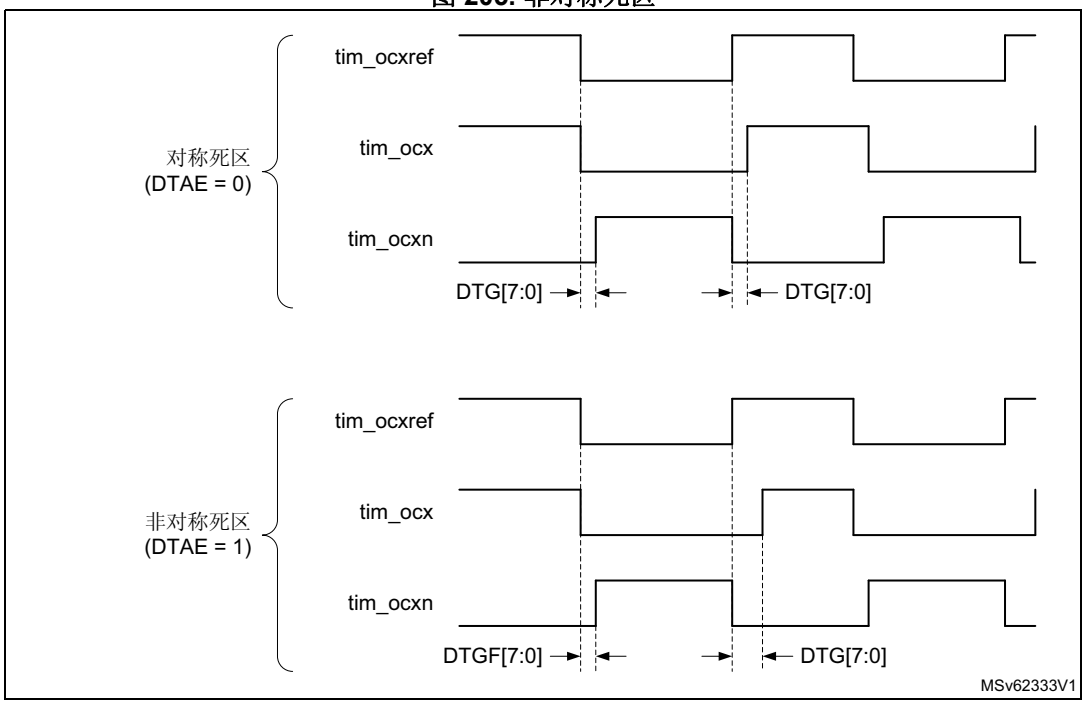


图 206. 延迟时间大于负脉冲宽度的死区波形

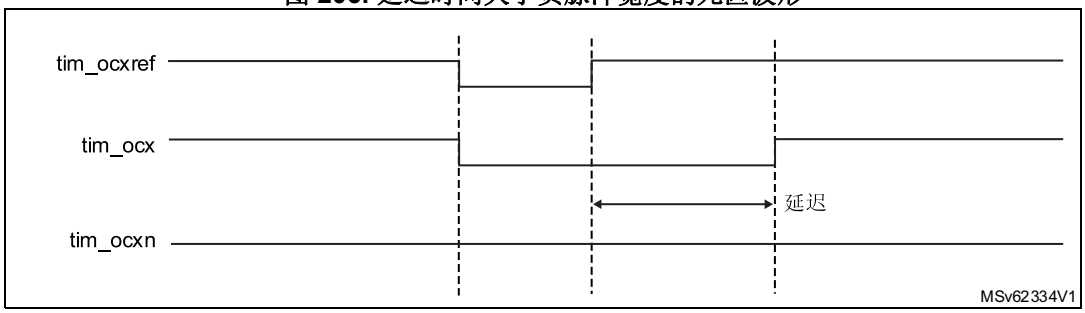
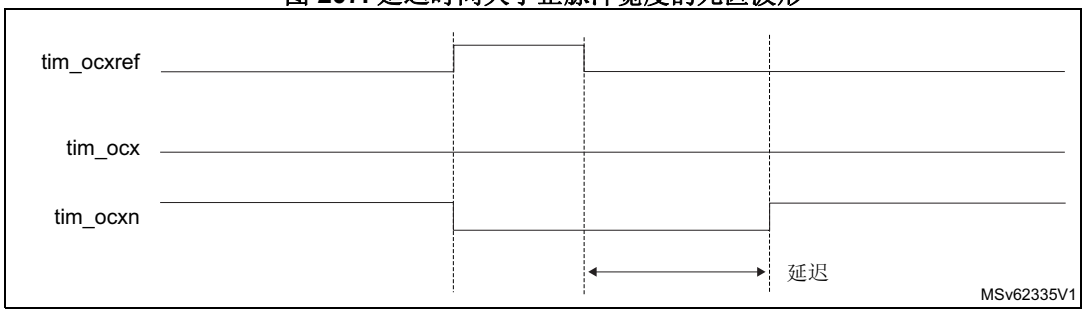


图 207. 延迟时间大于正脉冲宽度的死区波形



死区延迟对于所有通道均相同，可通过 TIMx_BDTR 寄存器中的 DTG 位进行编程。有关延迟时间计算的信息，请参见*第 26.6.20 节：TIM1 刹车和死区寄存器 (TIM1_BDTR)*。

将 tim_ocxref 重定向到 tim_ocx 或 tim_ocxn

在输出模式（强制输出模式、输出比较模式或 PWM 模式）下，通过配置 TIMx_CCER 寄存器中的 CCxE 和 CCxNE 位，可将 tim_ocxref 重定向到 tim_ocx 输出或 tim_ocxn 输出。

此功能可用于在一个输出上发送特定波形（如 PWM 或静态有效电平），而同时使互补输出保持其无效电平。或者，使两个输出同时保持无效电平，或者两个输出同时处于有效电平，两者互补并且带死区。

注意： *如果仅使能 tim_ocxn（CCxE = 0，CCxNE = 1），两者不互补，一旦 tim_ocxref 为高电平，tim_ocxn 即变为有效。例如，如果 CCxNP = 0，则 tim_ocxn = tim_ocxref。另一方面，如果同时使能 tim_ocx 和 tim_ocxn (CCxE = CCxNE = 1)，tim_ocx 在 tim_ocxref 为高电平时变为有效，而 tim_ocxn 则与之互补，在 tim_ocxref 为低电平时变为有效。*

26.3.18 使用刹车功能

刹车功能的目的是保护由定时器生成的 PWM 信号所驱动的电源开关。两个刹车输入通常连接到功率级和三相逆变器的故障输出。激活时，刹车电路会关闭 PWM 输出，并将其强制为预定义的安全状态。也可选择一些内部 MCU 事件来触发输出关断。

有两个刹车通道。一个刹车通道收集系统级故障（时钟失效、ECC/奇偶校验错误等）和应用故障（来自输入引脚和内置比较器），可以在死区持续时间后将输出强制为预定义的电平（有效或无效）。刹车 2 通道只包括应用故障，能够将输出强制为无效状态。

刹车期间的输出使能信号和输出电平取决于多个控制位：

- TIMx_BDTR 寄存器中的 MOE 位用于通过软件使能/禁止输出，在发生刹车事件和刹车 2 事件时复位。
- TIMx_BDTR 寄存器中的 OSSI 位，定义定时器将输出控制在无效状态下，还是将输出控制释放给 GPIO 控制器（通常使其处于高阻态模式）
- TIMx_CR2 寄存器中的 OISx 和 OISxN 位，将输出设置为关断电平（有效或无效）。无论 OISx 和 OISxN 的值为何，均无法在给定时间将 tim_ocx 和 tim_ocxn 输出同时设置为有效电平。更多详细信息，请参见*第 795 页的表 182：具有刹车功能的互补通道 tim_ocx 和 tim_ocxn 的输出控制位*。

退出复位状态后，刹车功能处于禁止状态，MOE 位处于低电平。通过设置 TIMx_BDTR 寄存器中的 BKE 位和 BK2E 位来使能刹车功能。可通过配置同一寄存器中的 BKP 位和 BK2P 位来选择刹车输入的极性。BKEx 和 BKPx 位可同时修改。对 BKEx 和 BKPx 位执行写操作时，写操作会在 1 个 APB 时钟周期的延迟后生效。因此，执行写操作后，需要等待 1 个 APB 时钟周期，才能准确回读该位。

由于 MOE 下降沿可能是异步信号，因此在实际信号（作用于输出）与同步控制位（位于 TIMx_BDTR 寄存器中）之间插入了再同步电路，从而在异步信号与同步信号之间产生延迟。具体而言，如果在 MOE 处于低电平时将其置 1，则必须首先插入延迟（空指令），才能准确进行读取。这是因为写入的是异步信号，而读取的却是同步信号。

刹车 (tim_brk) 通道的源为：

- 连接到 TIMx_BKIN 引脚之一的外部源（由 GPIO 复用功能选择寄存器选定），具有极性选择和可选的数字滤波
- 内部源：
  - 来自 tim_brk_cmpx 输入（有关产品特定的实现，请参见*第 26.3.2 节：TIM1 引脚和内部信号*）
  - 来自系统刹车请求（有关产品特定的实现，请参见*第 26.3.2 节：TIM1 引脚和内部信号*）

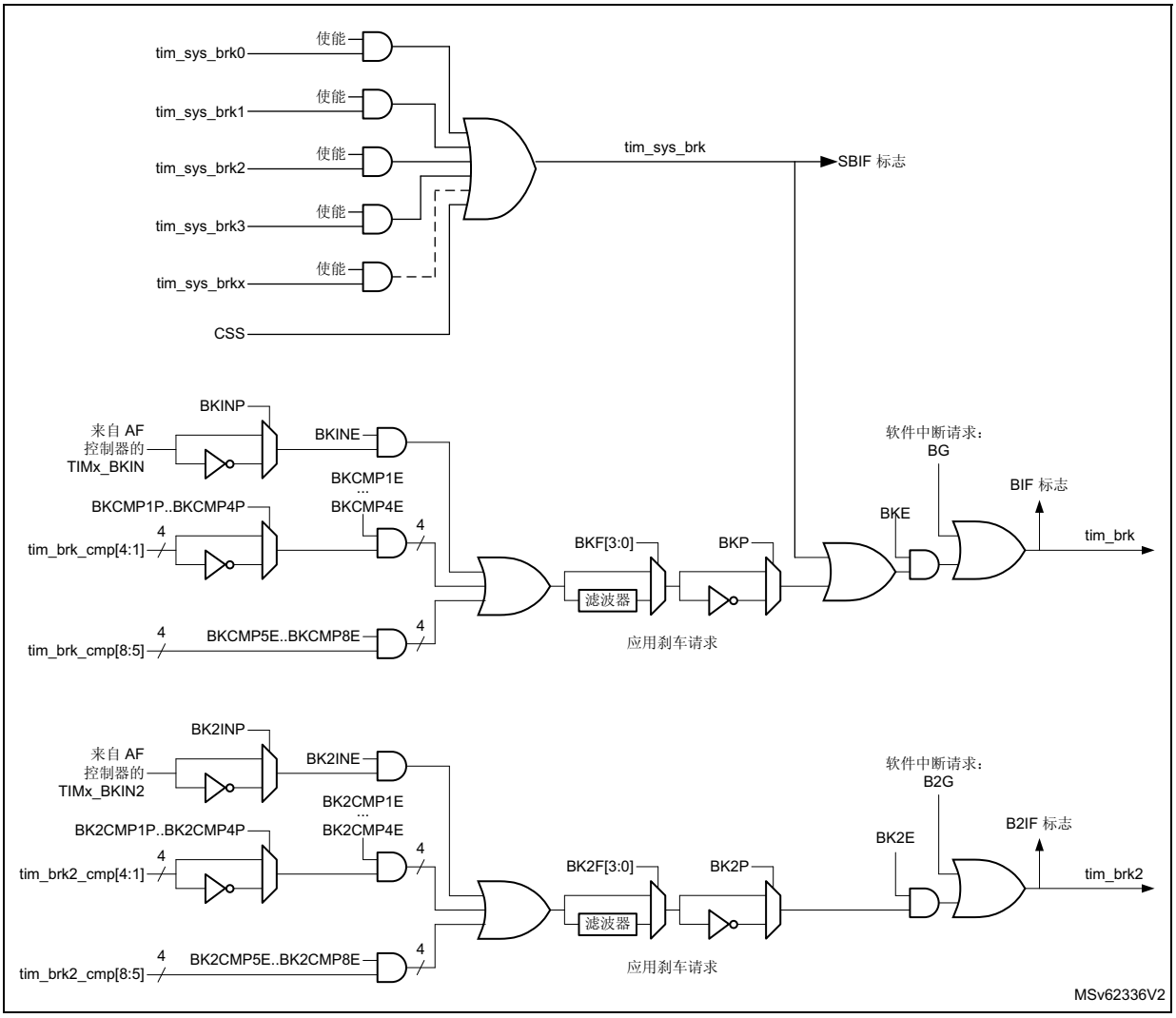
刹车 2 (tim_brk2) 的源为：

- 连接到 TIMx_BKIN2 引脚之一的外部源（由 GPIO 复用功能选择寄存器选定），具有极性选择和可选的数字滤波
- 来自 tim_brk2_cmpx 输入的内部源（有关产品特定的实现，请参见*第 26.3.2 节：TIM1 引脚和内部信号*）

也可由软件通过 TIMx_EGR 寄存器中的 BG 和 B2G 位产生刹车事件。

在所有源进入定时器 tim_brk 或 tim_brk2 输入之前，对其进行或运算，如下面的*图 208* 所示。

图 208. 刹车事件和刹车 2 电路概述



注意： *只有禁止可编程滤波器时才能保证异步（无时钟）操作。如果使能可编程滤波器，必须使用故障安全时钟模式（例如，使用内部 PLL 和/或 CSS）来保证能够处理刹车事件。*

发生刹车之一（其中一个刹车输入上出现所选电平）时：

- MOE 位异步清零，使输出处于无效状态、空闲状态甚至释放控制权给 GPIO 控制器（通过 OSSI 位进行选择）。即使 MCU 振荡器关闭，该功能仍然使能。
- MOE=0 时，将以 TIMx_CR2 寄存器 OISx 位中设定的电平驱动每个输出通道。如果 OSSI=0，定时器将释放输出控制（由 GPIO 控制器接管），否则使能输出保持高电平。
- 使用互补输出时：
  - 输出首先置于无效状态（取决于极性）。这是异步操作，因此即使没有为定时器提供时钟，该操作仍有效。
  - 如果定时器时钟仍存在，则将重新激活死区发生器，进而在死区后以 OISx 和 OISxN 位中编程的电平驱动输出。即使在这种情况下，也不能同时将 tim_ocx 和 tim_ocxn 驱动至其有效电平。请注意，MOE 进行再同步，因此死区的持续时间会比通常情况稍长一些（约 2 个 tim_ker_ck 时钟周期）。

  - 如果 OSSI = 0，定时器将释放输出控制（由强制高阻态的 GPIO 控制器接管），否则使能输出将保持高电平或在 CCxE 或 CCxNE 位之一为高电平时立即变为高电平。
- 将刹车状态标志（TIMx_SR 寄存器中的 SBIF、BIF 和 B2IF 位）置 1。如果 TIMx_DIER 寄存器中的 BIE 位置 1，则会产生中断。如果 TIMx_DIER 寄存器中的 BDE 位置 1，可发送 DMA 请求。
- 如果 TIMx_BDTR 寄存器中的 AOE 位置 1，则 MOE 位会在发生下一更新事件 (UEV) 时自动再次置 1。例如，这可用于执行调节。否则，MOE 将始终保持低电平，直到应用将其再次置 1。这种情况下，这一特性可用于确保安全。可以将刹车输入连接到功率驱动器的警报、温度传感器或任何安全元件。

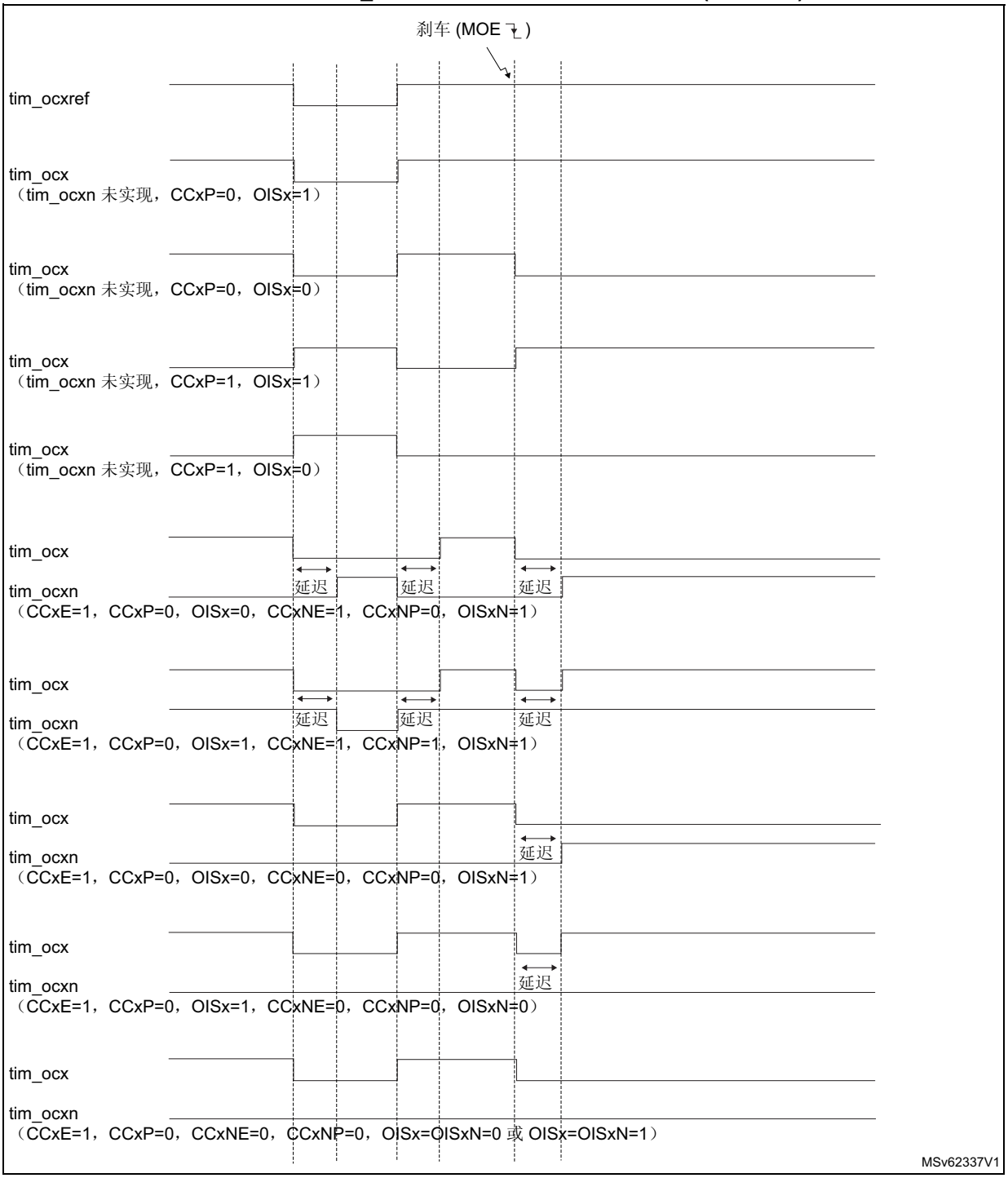
注意： *当 AOE 位置 1 时，如果 CPU 将 MOE 复位，则输出处于空闲状态并根据 OSSI 值被强制设为无效电平或高阻态。如果 CPU 同时将 MOE 和 AOE 位复位，则输出处于禁止状态并由 TIMx_CR2 寄存器中 OISx 位设定的电平驱动。*

注意： *刹车输入为电平有效。因此，当刹车输入为有效电平时，不能将 MOE 位置 1（自动或通过软件）。同时，不能将状态标志 BIF 和 B2IF 清零。*

除刹车输入和输出管理外，刹车电路内部还实施了写保护，用以保护应用的安全。该功能用于冻结多个参数配置（死区持续时间、tim_ocx/tim_ocxn 极性和禁止时的状态、OCxM 配置、刹车使能和极性）。应用可以通过 TIMx_BDTR 寄存器中的 LOCK 位，从 3 种保护级别中进行选择。请参见*第 26.6.20 节：TIM1 刹车和死区寄存器 (TIM1_BDTR)*。MCU 复位后只能对 LOCK 位执行一次写操作。

图 209 所示为输出对刹车响应行为的示例。

图 209. 响应 tim_brk 上的刹车事件的不同输出行为 (OSSI = 1)



两个刹车输入针对定时器输出具有不同的行为：

- tim_brk 输入可禁止（无效状态）PWM 输出，也可将 PWM 输出强制为预定义的安全状态。
- tim_brk2 只能禁止（无效状态）PWM 输出。

tim_brk 输入的优先级高于 tim_brk2 输入，如*表 175* 所示。

注意： *tim_brk2 只能在 OSSR = OSSI = 1 时使用。*

表 175. 定时器输出行为与 tim_brk/tim_brk2 输入的关系

tim_brk	tim_brk2	定时器输出状态	典型用例	
			tim_ocxn 输出（下桥臂开关）	tim_ocx 输出（上桥臂开关）
有效	X	– 无效，之后强制为输出状态（死区后） – 如果 OSSI = 0，则禁止输出（由 GPIO 逻辑接管控制）	死区插入后开启	关闭
无效	有效	无效	关闭	关闭

图 210 给出了 on tim_brk 和 tim_brk2 输入上出现有效信号时 tim_ocx 和 tim_ocxn 输出行为示例。在这种情况下，两个输出的极性均为高电平有效（TIMx_CCER 寄存器中的 CCxP = CCxNP = 0）。

图 210. tim_brk 和 tim_brk2 使能后的 PWM 输出状态 (OSSI = 1)

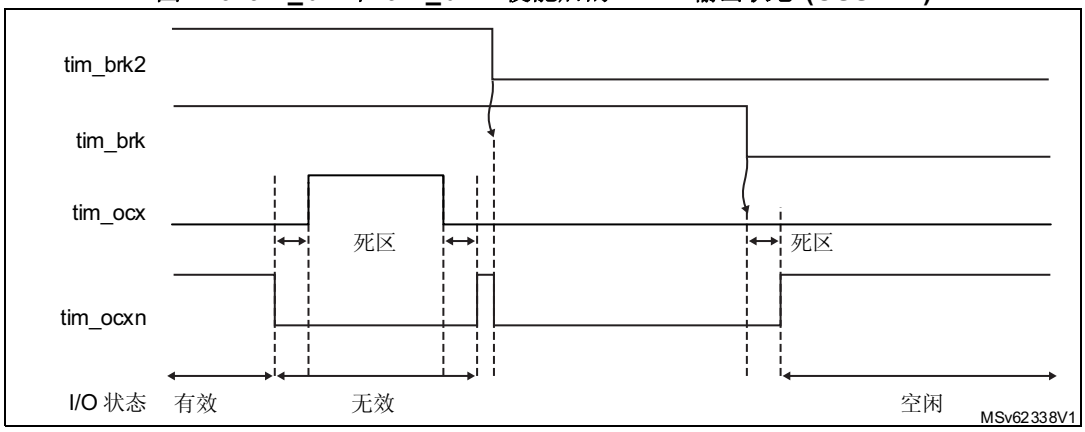
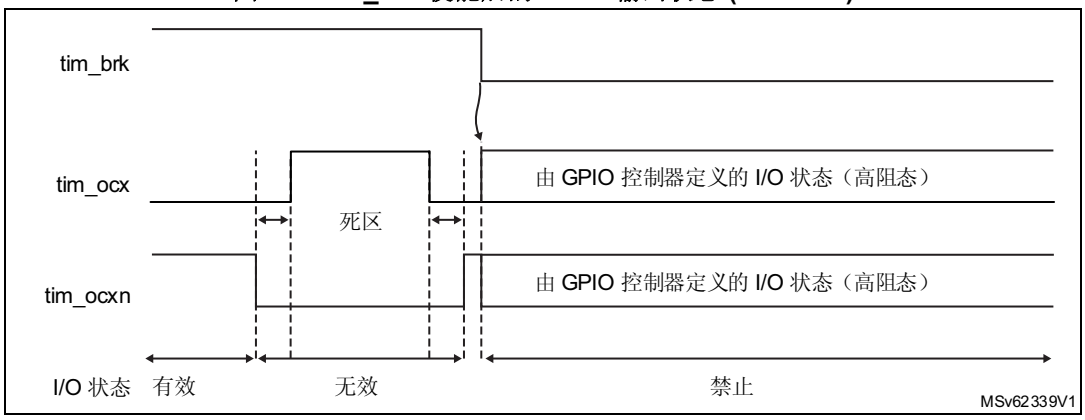


图 211. tim_brk 使能后的 PWM 输出状态 (OSSI = 0)



26.3.19 双向刹车输入

TIM1 具有双向刹车 I/O，如*图 212* 所示。

这为以下方面提供了支持：

- 将板级全局刹车信号用于向外部 MCU 或栅极驱动器发送故障信号，唯一的特定引脚作为输入输出双向状态脚
- 在必须将多个内部和外部刹车输入合并时，将内部刹车源和多个外部开漏源一起进行或运算，触发唯一的刹车事件。

使用 TIMxBDTR 寄存器的 BKBID 和 BK2BID 位，将 tim_brk 和 tim_brk2 输入配置为双向模式。可以使用 TIMxBDTR 寄存器中的 LOCK 位，将 BKBID 编程位锁定在只读模式（锁定级别 1 或更高）。

双向模式可以用于 tim_brk 和 tim_brk2 输入，需要将 I/O 配置为低电平有效极性的开漏模式（使用 BKINP、BKP、BK2INP 和 BK2P 位进行配置）。任何来自系统（例如 CSS）、片上外设或刹车输入的刹车请求都会强制将刹车输入置为低电平，以通知发生了故障事件。如果未正确设置极性位（高电平有效极性），则出于安全目的禁止双向模式。

软件刹车事件（BG 和 B2G）也会导致刹车 I/O 被强制置为“0”，从而向外部组件指示定时器已进入刹车状态。但是仅在刹车使能时（BKE 或 B2KE = 1）有效。当生成软件刹车事件且 BKE 或 B2KE = 0 时，输出将被置于安全状态，并且刹车标志置 1，但对 TIMx_BKIN 和 TIMx_BKIN2 I/O 无影响。

安全解除机制可防止系统最终锁定（刹车输入上的低电平会触发刹车，进而将相同输入强制置为低电平）。

当 BKDSRM (BK2DSRM) 位置 1 时，会释放刹车输出以清除故障信号，从而使系统能够重新获得保护。

在任何情况下都不能禁止刹车保护电路：

- 刹车输入路径始终有效：即使 BKDSRM (BK2DSRM) 位置 1 且释放开漏控制，刹车事件也仍然有效。这样可以在发生刹车期间防止 PWM 输出重新启动。
- 使能输出（MOE 位置 1）后，BKDSRM (BK2DSRM) 位不能解除刹车保护（请参见*表 176*）

表 176. 刹车保护解除条件

MOE	BKBID (BK2BID)	BKDSRM (BK2DSRM)	刹车保护状态
0	0	X	启动
0	1	0	启动
0	1	1	解除
1	X	X	启动

启动和重新启动刹车电路

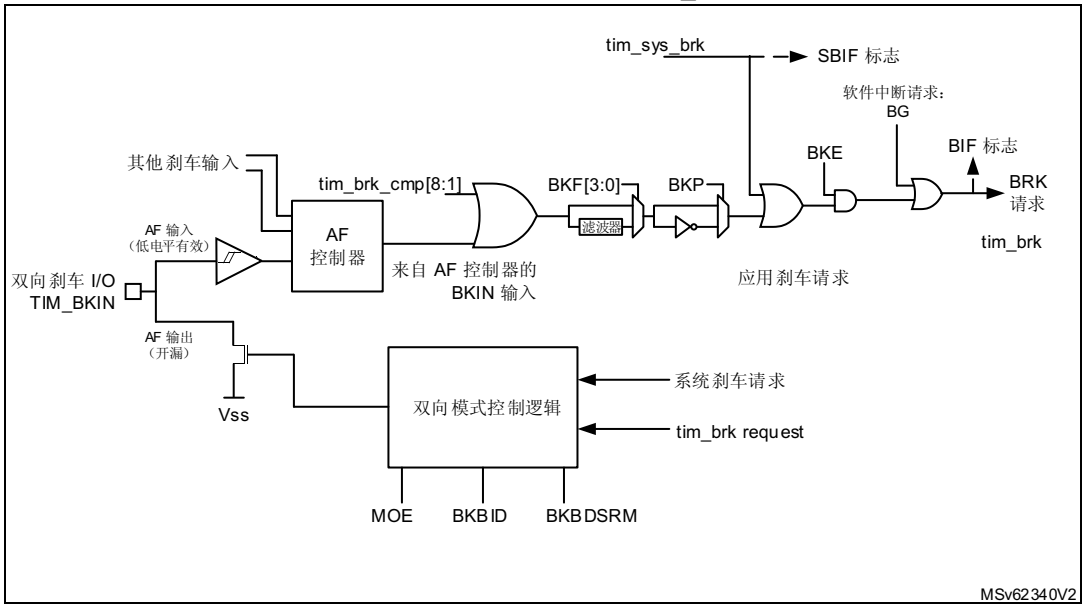
默认情况下（外设复位配置）会启动刹车电路（在输入或双向模式下）。

发生刹车（刹车 2）事件后，必须按照以下步骤重新启动保护：

- 必须将 BKDSRM (BK2DSRM) 位置 1，以释放输出控制
- 软件必须等待系统刹车条件消失（如果有），并清零 SBIF 状态标志（或在重新启动前由系统清零）
- 软件必须轮询 BKDSRM (BK2DSRM) 位，直到该位由硬件清零（当应用程序刹车条件消失时）

此后，刹车电路即启动并激活，可以通过将 MOE 位置 1 来重新使能 PWM 输出。

图 212. 输出重定向（图中未显示 tim_brk2 请求）

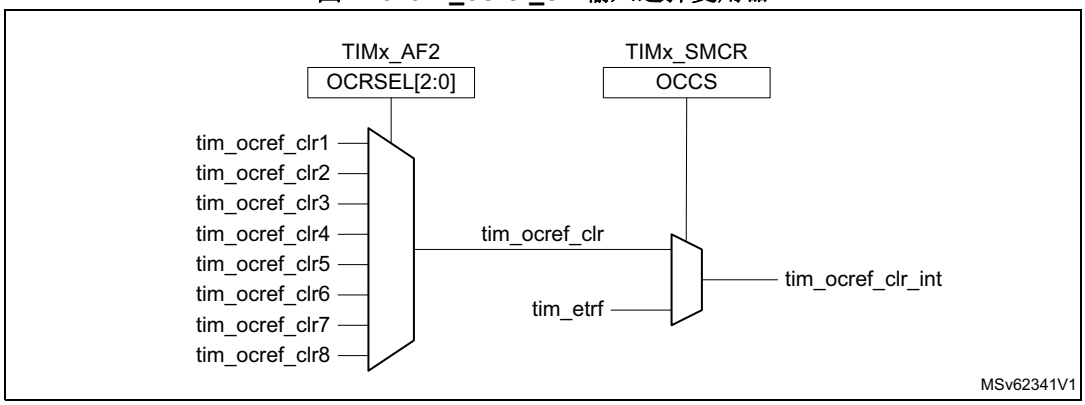


26.3.20 发生外部事件时清除 tim_ocxref 信号

对于给定通道，在 tim_ocref_clr_int 输入上施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置 1），可将 tim_ocxref 信号清零。tim_ocxref 信号将保持低电平，直到在下一 PWM 周期再次变为有效状态。该功能只能在输出比较模式和 PWM 模式下使用。在强制模式下不起作用。通过配置 TIMx_SMCR 寄存器中的 OCCS 位，可在 tim_ocref_clr 输入和 tim_etrf（滤波器后的 tim_etr_in）之间选择 tim_ocref_clr_int 输入。

可以使用 TIMx_AF2 寄存器中的 OCRSEL[2:0] 位域在多个输入中选择 tim_ocref_clr 输入，如下面的*图 213* 所示。有关产品中可用的源，请参见*第 26.3.2 节：TIM1 引脚和内部信号*。

图 213. tim_ocref_clr 输入选择复用器

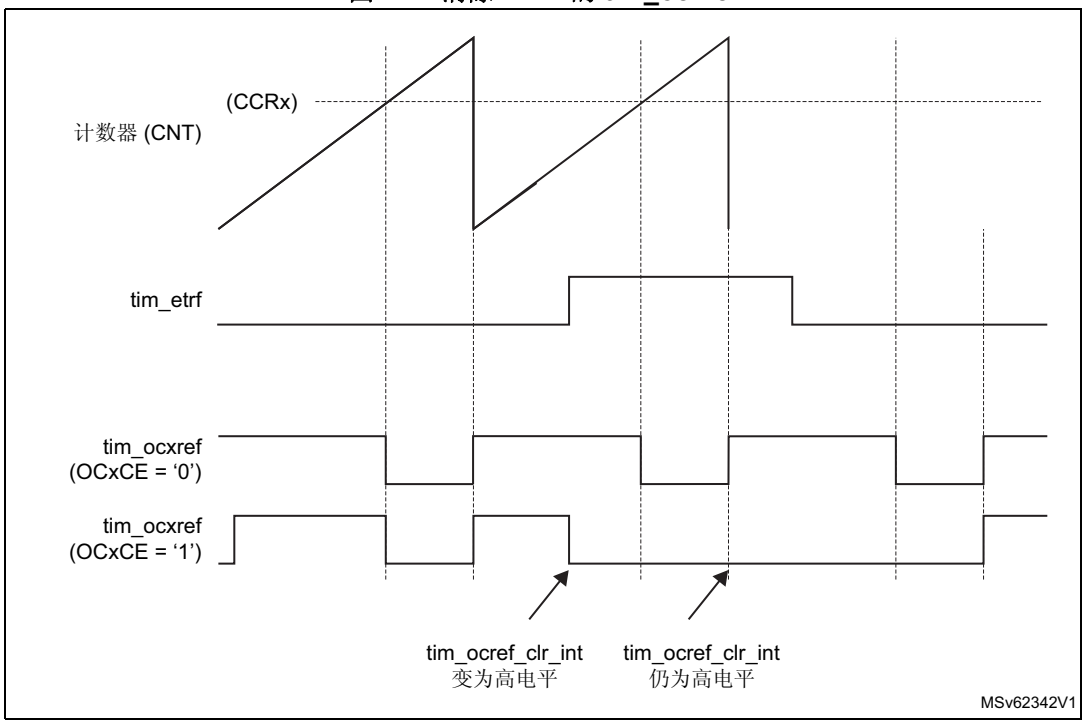


选择 tim_etrf 时，tim_etr_in 必须配置如下：

1. 必须关闭外部触发预分频器：TIMx_SMCR 寄存器中的 ETPS[1:0] 位置“00”。
2. 必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE 位置“0”。
3. 外部触发极性 (ETP) 和外部触发滤波器 (ETF) 可根据应用需要进行配置（根据触发源的极性以及最终是否需要使用滤波器来消除噪声）。

图 214 对比了使能位 OCxCE 在不同值下的情况，显示了当 tim_etrf 输入变为高电平时 tim_ocxref 信号的行为。在本例中，定时器 TIMx 编程为 PWM 模式。

图 214. 清除 TIMx 的 tim_ocxref



注意： *如果 PWM 的占空比为 100% (CCRx > ARR)，则下次计数器溢出时会再次使能 tim_ocxref。*

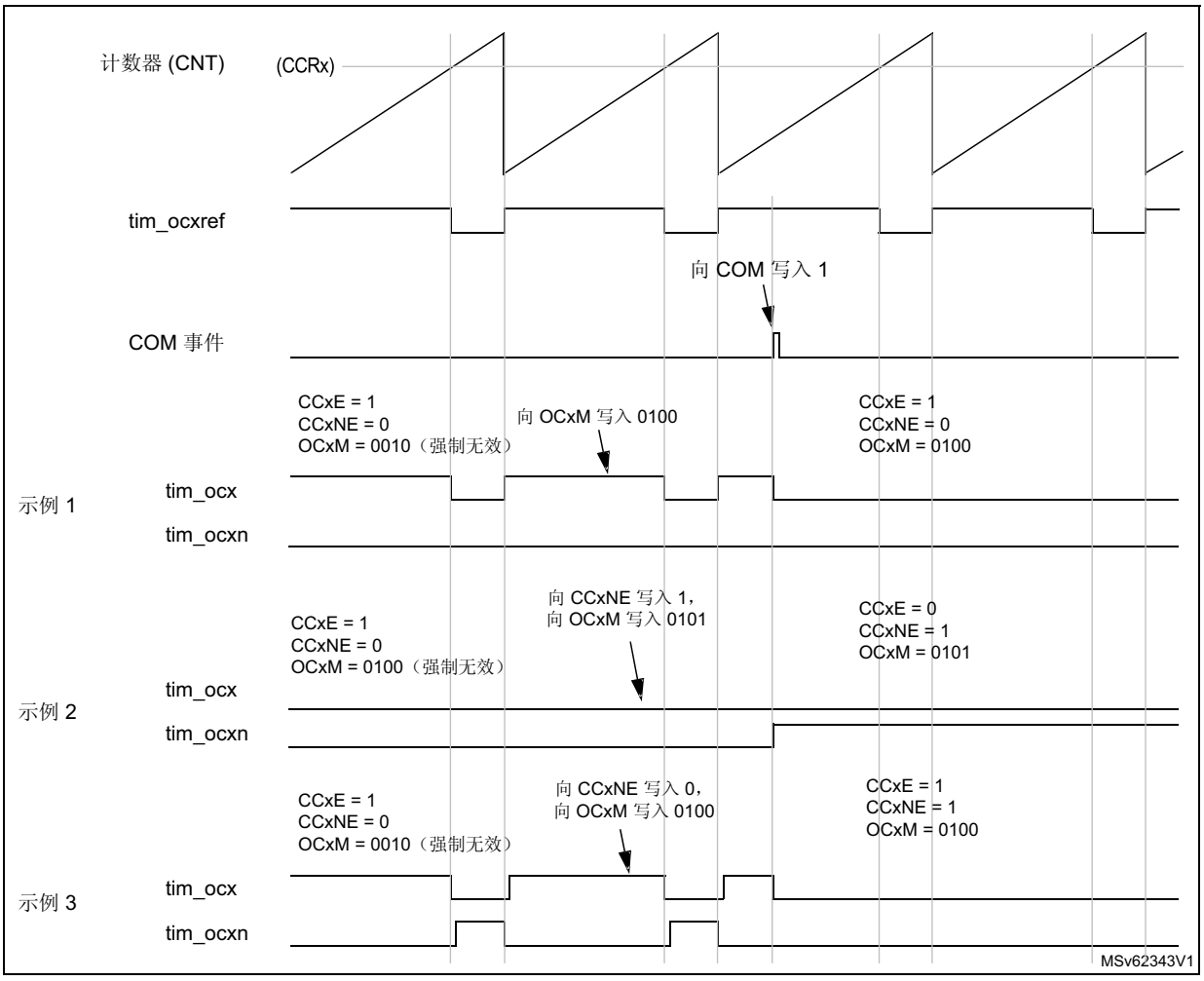
26.3.21 生成 6 步 PWM

当通道使用互补输出时，OCxM、CCxE 和 CCxNE 位上提供预装载位。发生 COM 换向事件时，这些预装载位将传输到影子位。因此，用户可以预先编程下一步骤的配置，并同时更改所有通道的配置。COM 可由软件通过将 TIMx_EGR 寄存器中的 COM 位置 1 而生成，也可以由硬件在 tim_trgi 上升沿生成。

发生 COM 事件时，某个标志位（TIMx_SR 寄存器中的 COMIF 位）将会置 1。这时，如果 TIMx_DIER 寄存器中的 COMIE 位置 1，将产生中断；如果 TIMx_DIER 寄存器中的 COMDE 位置 1，则将产生 DMA 请求。

图 215 以 3 种不同的编程配置为例，显示了发生 COM 事件时 tim_ocx 和 tim_ocxn 输出的行为。

图 215. COM 事件生成 6 步 PWM 的示例 (OSSR=1)



26.3.22 单脉冲模式

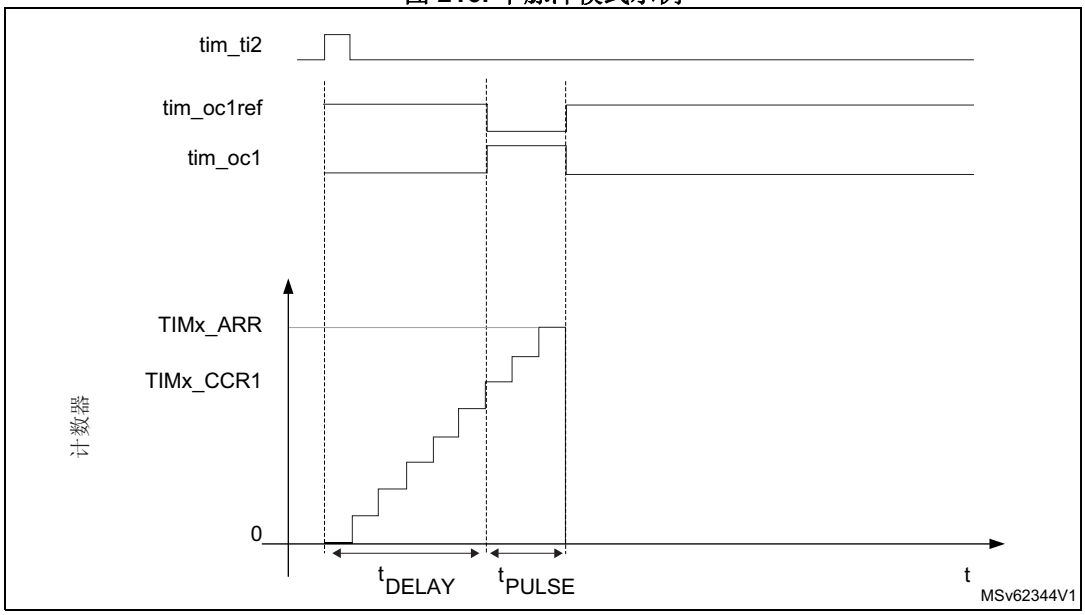
单脉冲模式 (OPM) 是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。通过将 TIMx_CR1 寄存器中的 OPM 位置 1 选择单脉冲模式。这样，发生下一更新事件 UEV 时，计数器将自动停止。

只有当比较值与计数器初始值不同时，才能正确产生一个脉冲。启动前（定时器等待触发时），必须进行如下配置：

- 递增计数模式下：CNT < CCRx ≤ ARR（特别注意，0 < CCRx）
- 递减计数模式下：CNT > CCRx

图 216. 单脉冲模式示例



例如，用户希望达到这样的效果：在 tim_ti2 输入引脚检测到上升沿时，经过 t_{DELAY} 的延迟，在 tim_oc1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 tim_ti2fp2 作为触发 1：

- 在 TIMx_CCMR1 寄存器中写入 CC2S = “01”，以将 tim_ti2fp2 映射到 tim_ti2。
- 在 TIMx_CCER 寄存器中写入 CC2P = “0”和 CC2NP = “0”，使 tim_ti2fp2 能够检测上升沿。
- 在 TIMx_SMCR 寄存器中写入 TS = 00110，将 tim_ti2fp2 配置为从模式控制器的触发 (tim_trgi)。
- 在 TIMx_SMCR 寄存器中写入 SMS = “110”（触发模式），以使用 tim_ti2fp2 启动计数器。

OPM 波形通过对比较寄存器执行写操作来定义（考虑时钟频率和计数器预分频器）。

- t_{DELAY} 由写入 TIMx_CCR1 寄存器的值定义。
- t_{PULSE} 由自动重载值与比较值之差 (TIMx_ARR - TIMx_CCR1) 来定义。
- 假设希望产生这样的波形：信号在发生比较匹配时从“0”变为“1”，在计数器达到自动重载值时由“1”变为“0”。为此，必须通过在 TIMx_CCMR1 寄存器中写入 OC1M=111，来使能 PWM 模式 2。可选择在 TIMx_CCMR1 寄存器的 OC1PE 和

TIMx_CR1 寄存器的 ARPE 中写入“1”，以使能预装载寄存器。这种情况下，必须在 TIMx_CCR1 寄存器中写入比较值并在 TIMx_ARR 寄存器中写入自动重载值，通过将 UG 位置 1 来产生更新，然后等待 tim_ti2 上的外部触发事件。本例中，CC1P 的值为“0”。

在本例中，TIMx_CR1 寄存器中的 DIR 和 CMS 位必须为低。

由于仅需要 1 个脉冲（单脉冲模式），因此必须向 TIMx_CR1 寄存器的 OPM 位写入 1，以便在发生下一更新事件（计数器从自动重载值返回到 0）时使计数器停止计数。TIMx_CR1 寄存器中的 OPM 位置“0”时，即选择重复模式。

特殊情况：tim_ocx 快速使能：

在单脉冲模式下，tim_tix 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时，将切换输出。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟（t_{DELAY} 最小值）。

如果要输出延迟时间最短的波形，可以将 TIMx_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 tim_ocxref（和 tim_ocx）对激励信号做出响应，而不再考虑比较的结果。其新电平与发生比较匹配时相同。仅当通道配置为 PWM1 或 PWM2 模式时，OCxFE 才会起作用。

26.3.23 可再触发单脉冲模式

该模式可以启动计数器来响应激励信号，并且能产生长度可编程的脉冲，但与不可再触发单脉冲模式间存在以下差别，如*第 26.3.22 节*所述：

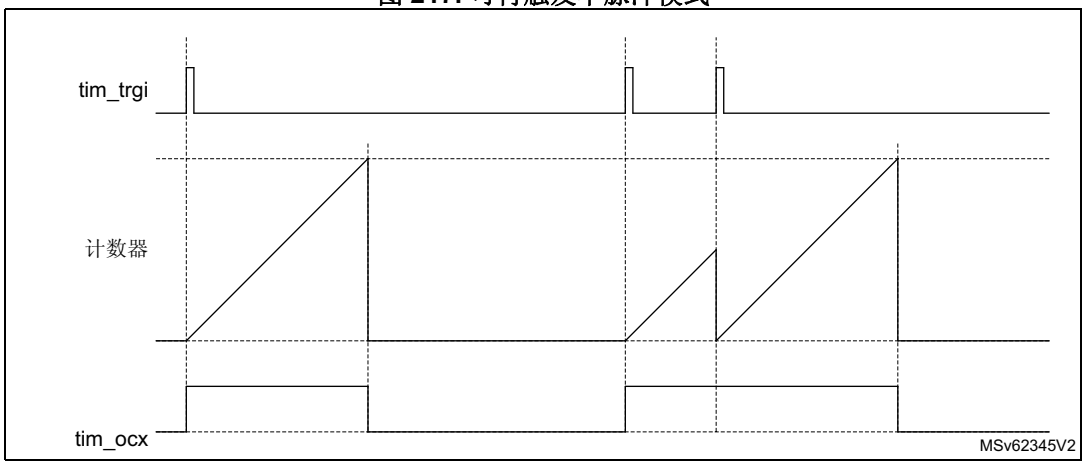
- 发生触发时，脉冲立即产生（无可编程延时）
- 如果在上一个触发完成前发生新的触发，脉冲将延长

定时器必须处于从模式，TIMx_SMCR 寄存器中的位 SMS[3:0] = “1000”（组合复位 + 触发模式），针对可再触发 OPM 模式 1 或模式 2 将 OCxM[3:0] 位设置为“1000”或“1001”。

定时器配置为递增计数模式时，相应的 CCRx 必须置 0（ARR 寄存器设置脉冲长度）。如果定时器配置为递减计数模式，CCRx 必须高于或等于 ARR。

注意： *出于兼容性原因，OCxM[3:0] 和 SMS[3:0] 位域分为两部分，最高有效位与最低有效的 3 位不相邻。*
此模式不能与中心对齐 PWM 模式一起使用。在 TIMx_CR1 中必须设置 CMS[1:0] = 00。

图 217. 可再触发单脉冲模式



26.3.24 基于比较的脉冲模式

可以在发生比较匹配事件时生成脉冲。当计数器值等于给定的比较值时可产生脉宽可编程的信号，该信号可用于调试或同步。

该模式可用于任何选定的从模式，包括编码器模式、边沿和中心对齐计数模式。该模式仅适用于通道 3 和通道 4。脉冲发生器只有一个，由两个通道共用，如下面的*图 218* 所示。

图 218. 脉冲发生器电路

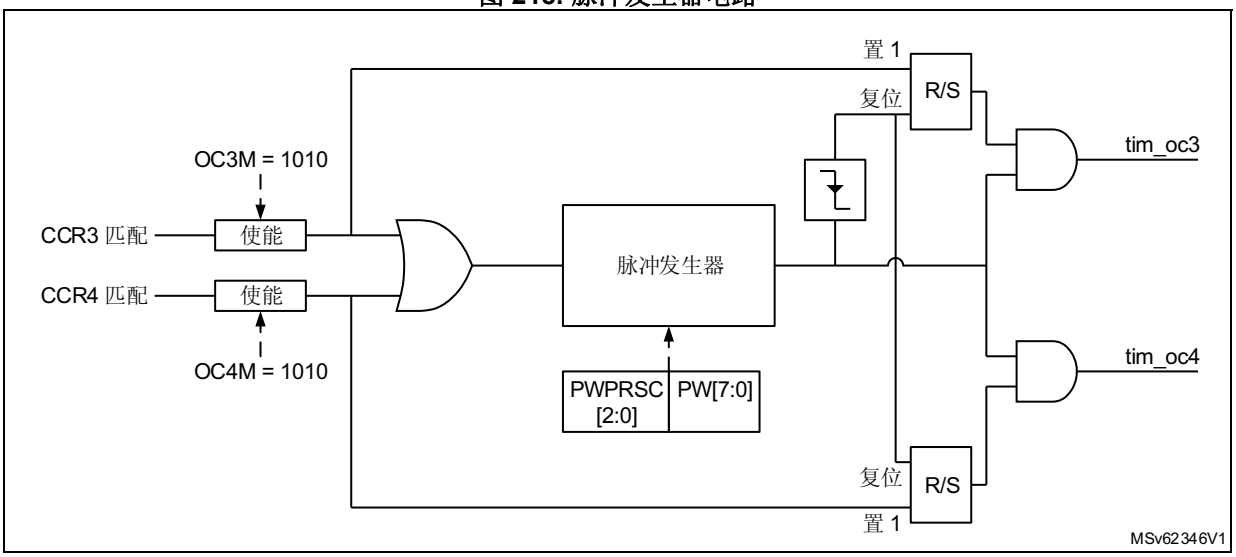
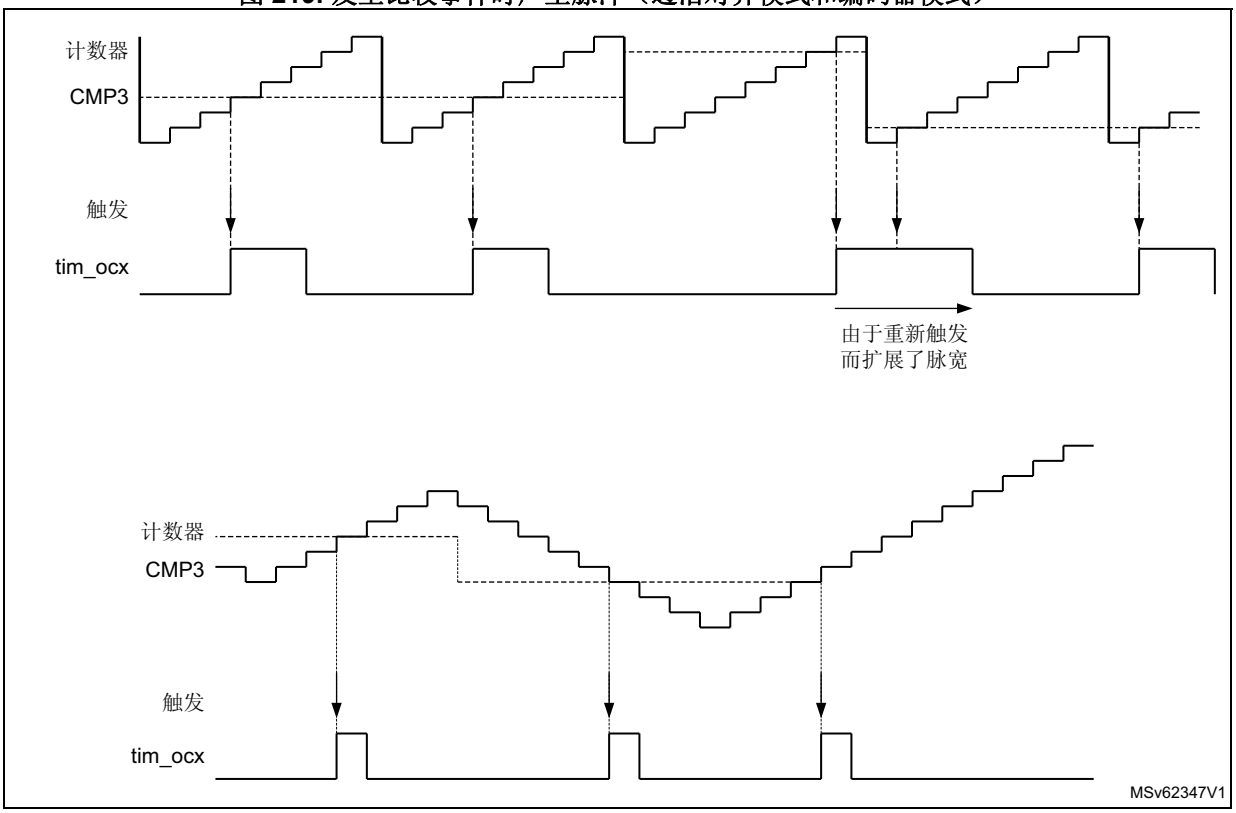


图 219 显示了如何在边沿对齐模式和编码器模式下产生脉冲。

图 219. 发生比较事件时产生脉冲（边沿对齐模式和编码器模式）



该输出比较模式使用 TIMx_CCMR2 寄存器中的 OC3M[3:0] 和 OC4M[3:0] 位域进行选择。

脉宽使用寄存器中的 PW[7:0] 位域以及根据 PWPRSC[2:0] 位预分频的特定时钟进行编程，具体如下所示：

$t_{PW} = PW[7:0] \times t_{PWG}$

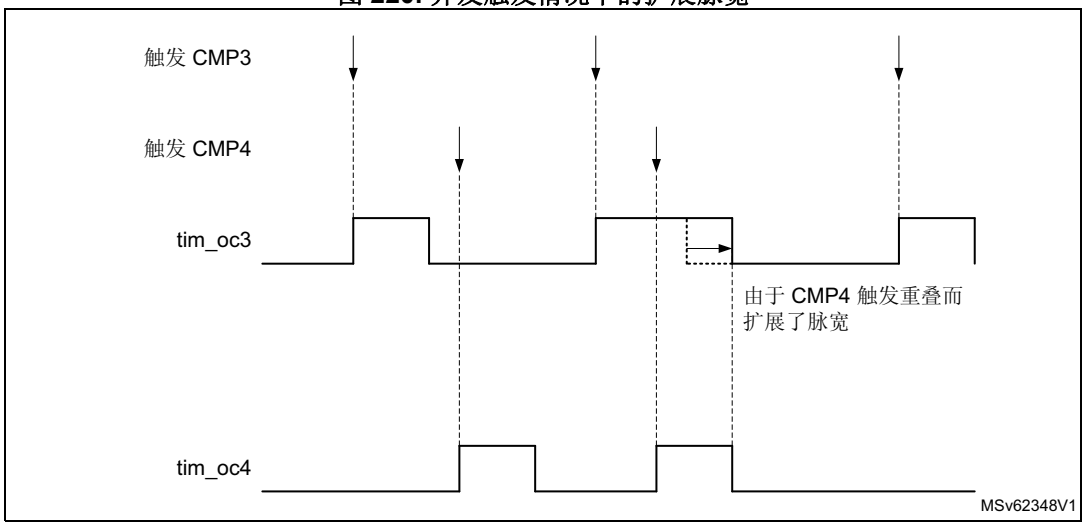
其中 $t_{PWG} = (2^{(PWPRSC[2:0])}) \times t_{tim_ker_ck}$。

给出了根据预分频值得出的分辨率和最大值。

脉冲可再触发：如果正在产生脉冲时发生新的触发，脉冲将会延长。

注意： *如果同时使能两个通道，则只要一个通道上的触发信号未与并发输出上产生的脉冲重叠，就可以独立发出脉冲。相反地，如果两个触发信号发生重叠，则先到达的触发信号的脉冲宽度将被延长（因为再触发），而后到达的触发信号的脉冲宽度则正确无误（如下面的图 220 所示）。*

图 220. 并发触发情况下的扩展脉宽



26.3.25 编码器接口模式

正交编码器

选择编码器接口模式时，如果计数器仅在 tim_ti1 边沿处计数，在 TIMx_SMCR 寄存器中写入 SMS = “0001”；如果计数器仅在 tim_ti2 边沿处计数，写入 SMS = “0010”；如果计数器在 tim_ti1 和 tim_ti2 边沿处均计数，则写入 SMS = “0011”。

通过编程 TIMx_CCER 寄存器的 CC1P 和 CC2P 位，选择 tim_ti1 和 tim_ti2 极性。如果需要，还可对输入滤波器进行编程。CC1NP 和 CC2NP 必须保持低电平。

tim_ti1 和 tim_ti2 两个输入用于连接正交编码器。请参见 *表 177*。如果使能计数器（在 TIMx_CR1 寄存器的 CEN 位中写入“1”），则计数器的时钟由 tim_ti1fp1 或 tim_ti2fp2 上的每次有效信号转换提供。tim_ti1fp1 和 tim_ti2fp2 是 tim_ti1 和 tim_ti2 进行输入滤波和极性选择后的信号，如果不进行滤波和反相，则 tim_ti1fp1 = tim_ti1，tim_ti2fp2 = tim_ti2。将根据两个输入的信号转换序列，产生计数脉冲和方向信号。根据该信号转换序列，计数器相应递增或递减计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应修改。任何输入（tim_ti1 或 tim_ti2）发生信号转换时，都会计算 DIR 位，无论计数器是仅在 tim_ti1 或 tim_ti2 边沿处计数，还是同时在 tim_ti1 和 tim_ti2 处计数。

编码器接口模式就相当于带有方向选择的外部时钟。这意味着，计数器仅在 0 到 TIMx_ARR 寄存器中的自动重载值之间进行连续计数（根据具体方向，从 0 递增计数到 ARR，或从 ARR 递减计数到 0）。因此，在启动前必须先配置 TIMx_ARR。同样，捕获、比较、预分频器、重复计数器和触发输出功能继续正常工作。编码器模式和外部时钟模式 2 不兼容，因此不能同时选择。

在此模式下，计数器会根据正交编码器的速度和方向自动进行修改，因此，其内容始终表示编码器的位置。计数方向对应于所连传感器的旋转方向。下表汇总了可能的组合（假设 tim_ti1 和 tim_ti2 不同时切换）。

表 177. 计数方向与编码器信号的关系 (CC1P = CC2P = 0)

有效边沿	SMS[3:0]	相反信号的电平（tim_ti1fp1 对应 tim_ti2，tim_ti2fp2 对应 tim_ti1）	tim_ti1fp1 信号		tim_ti2fp2 信号	
			上升	下降	上升	下降
仅在 tim_ti1 处计数 x1 模式	1110	高	递减	递增	不计数	不计数
		低	不计数	不计数	不计数	不计数
仅在 tim_ti2 处计数 x1 模式	1111	高	不计数	不计数	递增	递减
		低	不计数	不计数	不计数	不计数
仅在 tim_ti1 处计数 x2 模式	0001	高	递减	递增	不计数	不计数
		低	递增	递减	不计数	不计数
仅在 tim_ti2 处计数 x2 模式	0010	高	不计数	不计数	递增	递减
		低	不计数	不计数	递减	递增
在 tim_ti1 和 tim_ti2 处均计数 x4 模式	0011	高	递减	递增	递增	递减
		低	递增	递减	递减	递增

正交编码器可直接与 MCU 相连，无需外部接口逻辑。不过，通常使用比较器将编码器的差分输出转换为数字信号。这样大幅提高了抗噪声性能。用于指示机械零位的第三个编码器输出可与外部触发输入相连，用以触发计数器复位。

图 221 以计数器工作为例，说明了计数信号的产生和方向控制。同时也说明了选择双边沿时如何对输入抖动进行补偿。将传感器靠近其中一个切换点放置时可能出现这种情况。本例中假设配置如下：

- CC1S =“01”（TIMx_CCMR1 寄存器，tim_ti1fp1 映射到 tim_ti1 上）。
- CC2S =“01”（TIMx_CCMR2 寄存器，tim_ti1fp2 映射到 tim_ti2 上）。
- CC1P =“0”，CC1NP =“0”（TIMx_CCER 寄存器，tim_ti1fp1 未反相，tim_ti1fp1 = tim_ti1）。
- CC2P =“0”，CC2NP =“0”（TIMx_CCER 寄存器，tim_ti1fp2 未反相，tim_ti1fp2 = tim_ti2）。
- SMS =“0011”（TIMx_SMCR 寄存器，两个输入在上升沿和下降沿均有效）。
- CEN=“1”（TIMx_CR1 寄存器，使能计数器）。

图 221. 编码器接口模式下的计数器工作示例

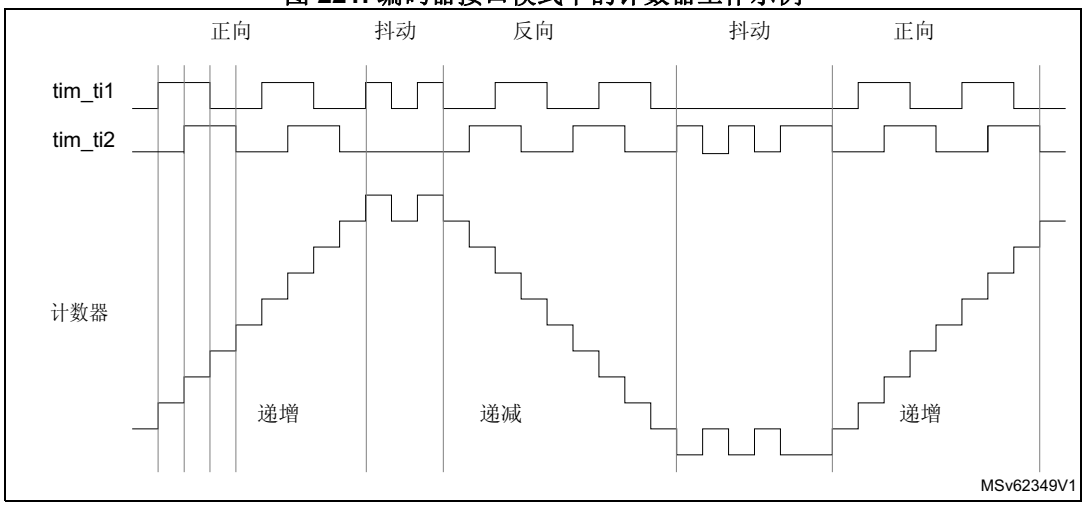
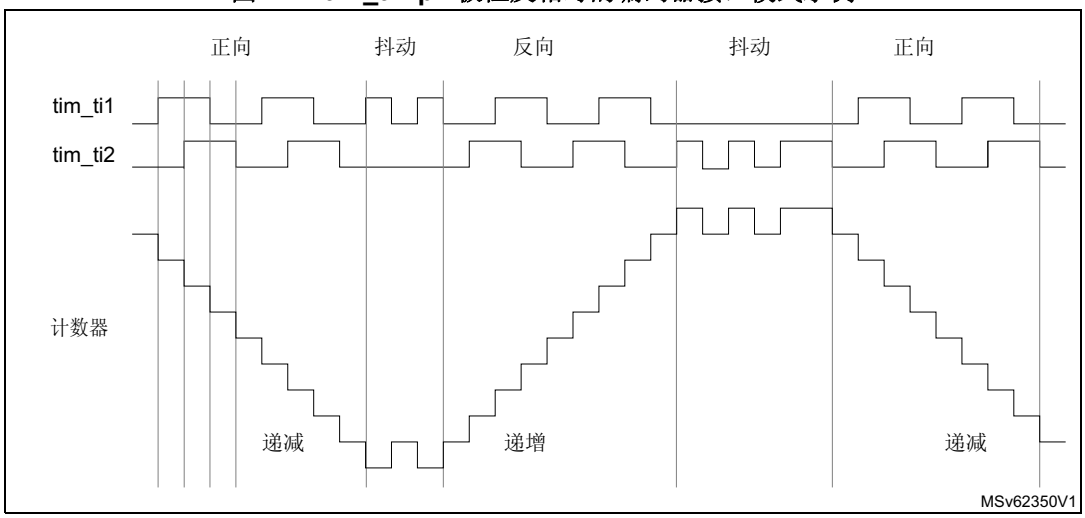


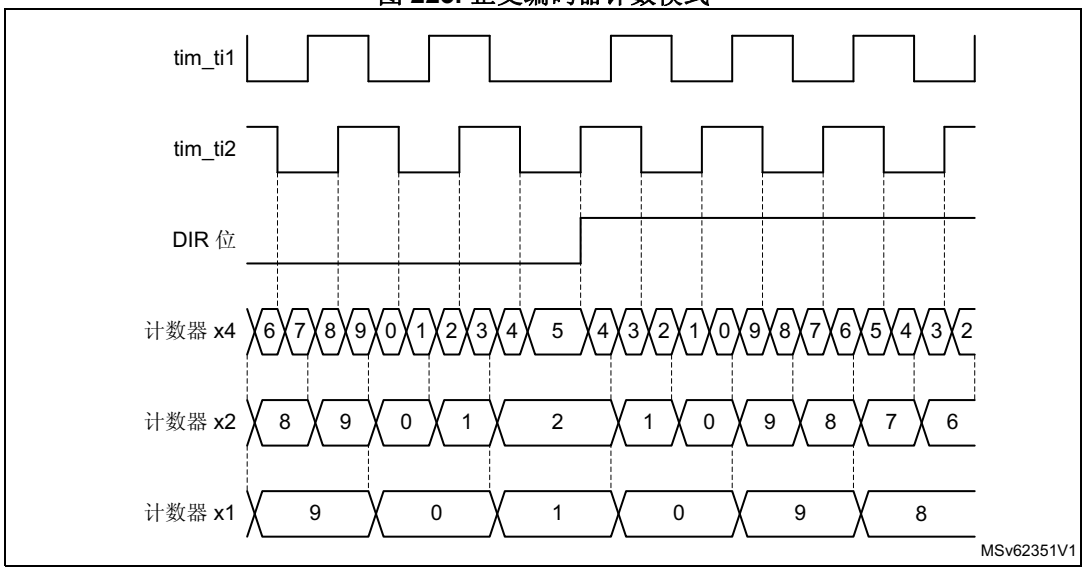
图 222 举例说明 tim_ti1fp1 极性反相时计数器的行为（除 CC1P = “1”外，其他配置与上例相同）。

图 222. tim_ti1fp1 极性反相时的编码器接口模式示例



下面的*图 223* 显示了在不同计数模式下速度反转期间的定时器计数器值。

图 223. 正交编码器计数模式



定时器配置为编码器接口模式时，会提供传感器当前位置的相关信息。使用另一个配置为捕获模式的定时器测量两个编码器事件之间的周期，可获得动态信息（速度、加速度和减速度）。指示机械零位的编码器输出即可用于此目的。根据两个事件之间的时间间隔，还可定期读取计数器。如果可能，可以将计数器值锁存到第三个输入捕获寄存器来实现此目的（捕获信号必须为周期性信号，可以由另一个定时器产生）；还可以通过 DMA 请求读取计数器值。

TIMx_CR1 寄存器中的 IUFREMAP 位强制将更新中断标志 (UIF) 连续复制到定时计数器寄存器的位 31 (TIMxCNT[31]) 中。这样便可按原子方式读取计数器值以及由 UIFCPY 标志发出的电位翻转条件。这可避免在后台任务（计数器读）和中断（更新中断）之间共享处理时产生竞争条件，从而简化角速度的计算。

UIF 和 UIFCPY 标志之间的置位没有延迟。

在 32 位定时器实现中，当 IUFREMAP 位置 1 时，计数器的位 31 在读访问时由 UIFCPY 标志覆盖（计数器的最高有效位只能在写模式下访问）。

时钟 + 方向编码器模式

除了正交编码器模式之外，定时器也支持其他类型的编码器。

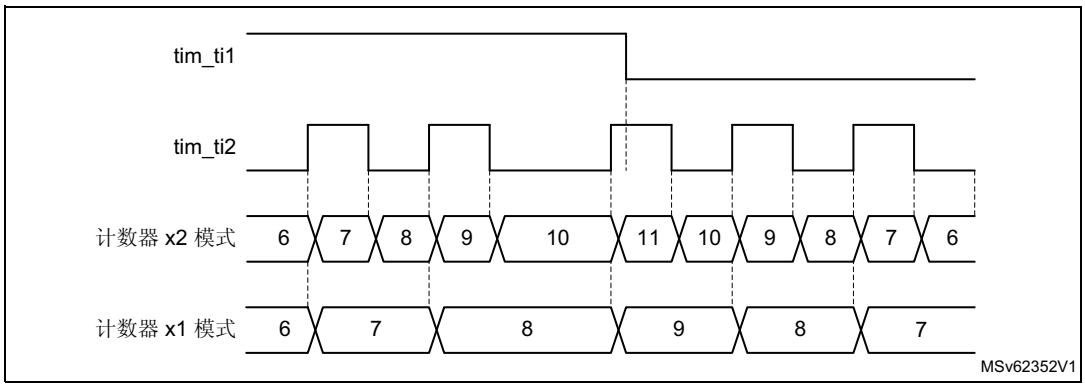
在“时钟 + 方向”模式下（如*图 224* 所示），时钟由 tim_ti2 单线提供，方向则由 tim_ti1 输入强制设定。

该模式通过 TIMx_SMCR 寄存器中的 SMS[3:0] 位域使能，具体如下所述：

- 1010：x2 模式，计数器在时钟的上升沿和下降沿均更新
- 1011：x1 模式，计数器在单个时钟边沿更新，具体哪个边沿取决于 CC2P 位的值：CC2P = 0 对应上升沿有效，CC2P = 1 对应下降沿有效

tim_ti1 上方向信号的极性由 CC1P 位设置：0 对应正极性（tim_ti1 为高电平时递增计数，tim_ti1 为低电平时递减计数），CC1P = 1 对应负极性（tim_ti1 为低电平时递增计数）。

图 224. 方向 + 时钟编码器模式



定向时钟编码器模式

在“定向时钟”模式下（如*图 225* 所示），时钟由两条线路提供，每次只使用一条线路，具体使用哪条线路取决于计数方向，因此具有一条递增计数时钟线和一条递减计数时钟线。

该模式通过 TIMx_SMCR 寄存器中的 SMS[3:0] 位域使能，具体如下所述：

- 1100：x2 模式，计数器在任一时钟线的上升沿和下降沿均更新。CC1P 和 CC2P 位为时钟空闲状态编码。CCxP = 0 对应高电平空闲状态（请参见*图 225*），CCxP = 1 对应低电平空闲状态（请参见*图 226*）。
- 1101：x1 模式，计数器在单个时钟边沿更新，具体哪个边沿取决于 CC1P 位和 CC2P 位的值：CCxP = 0 对应下降沿有效和高电平空闲状态（请参见*图 225*），CCxP = 1 对应上升沿有效和低电平空闲状态（请参见*图 226*）。

图 225. 定向时钟编码器模式（CC1P = CC2P = 0）

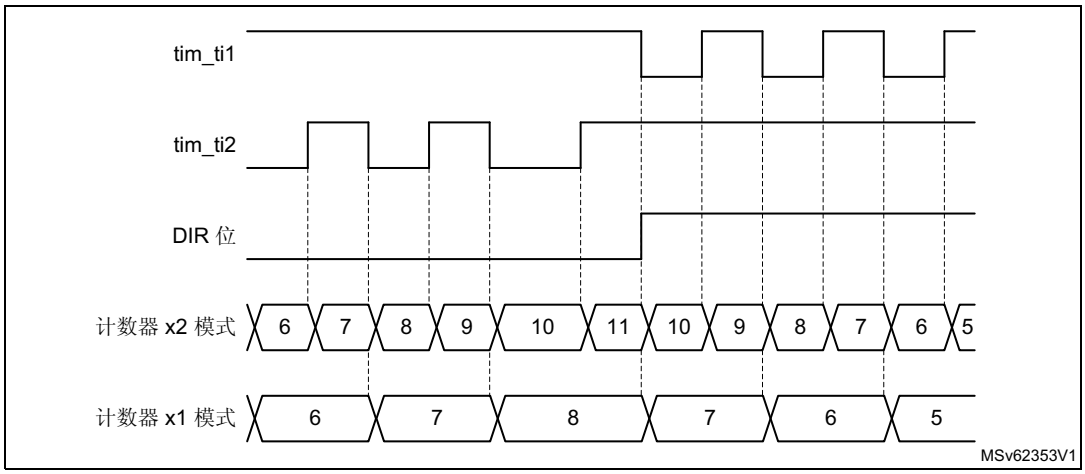
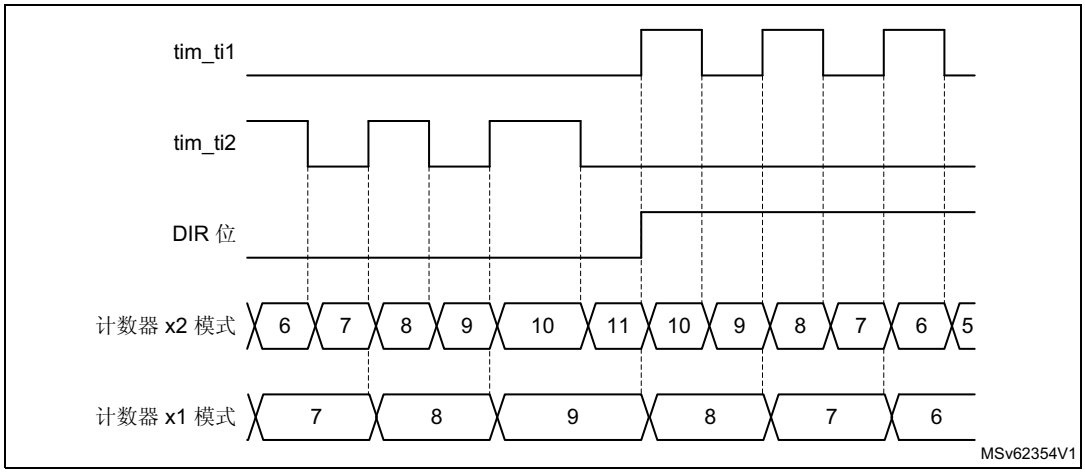


图 226. 定向时钟编码器模式（CC1P = CC2P = 1）



下面的*表 178* 详细介绍了针对任何输入转换，定向时钟模式的工作方式。

表 178. 计数方向与编码器信号和极性设置的关系

定向时钟模式	SMS[3:0]	相反信号的电平（tim_ti1fp1 对应 tim_ti2，tim_ti2fp2 对应 tim_ti1）	tim_ti1fp1 信号		tim_ti2fp2 信号	
			上升	下降	上升	下降
x2 模式 CCxP = 0	1100	高	递减	递减	递增	递增
		低	不计数	不计数	不计数	不计数
x2 模式 CCxP = 1	1100	高	不计数	不计数	不计数	不计数
		低	递减	递减	递增	递增
x1 模式 CCxP = 0	1101	高	不计数	递减	不计数	递增
		低	不计数	不计数	不计数	不计数
x1 模式 CCxP = 1	1101	高	不计数	不计数	不计数	不计数
		低	递减	不计数	递增	不计数

索引输入

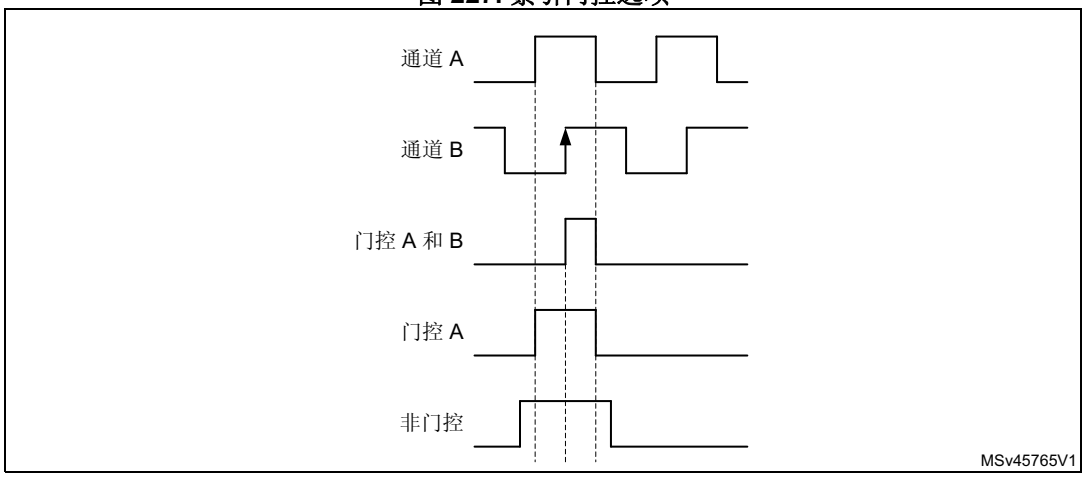
计数器可通过来自编码器的索引信号来复位，该信号指示绝对参考位置。索引信号必须连接到 tim_etr_in 输入。可使用数字输入滤波器对其进行滤波。

索引功能通过 TIMX_ECR 寄存器中的 IE 位来使能。仅当 SMS[3:0] 位域为以下值时，才可以在编码器模式下将 IE 位置 1：0001、0010、011、1010、1011、1100、1101、1110、1111。

商用编码器提供了多种索引脉冲调节选项，如下面的*图 227* 所示：

- 在 A 和 B 上进行门控：脉冲宽度为 1/4 个通道周期，与 A 和 B 边沿均对齐
- 在 A（或 B）上进行门控：脉冲宽度为 1/2 个通道周期，与通道 A（或通道 B）上的两个边沿对齐
- 不进行门控：脉冲宽度最多为一个通道周期，不与任何边沿对齐

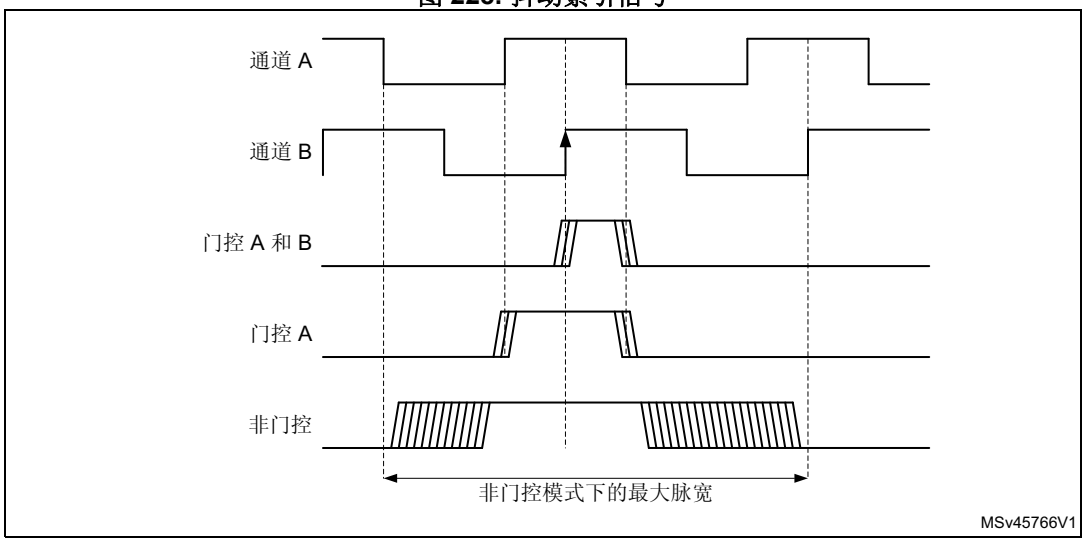
图 227. 索引门控选项



无论采用哪种门控模式，电路都能容忍索引信号的抖动，如下面的*图 228* 所示。

在非门控模式下，信号脉宽必须严格小于 2 个编码器周期。如果脉宽大于或等于 2 个编码器周期，计数器会多次复位。

图 228. 抖动索引信号



定时器平等支持 3 个门控选项，无需任何特定编程。只需使用 TIMx_ECR 寄存器中的 IPOS[1:0] 位域定义索引必须基于哪种编码器状态（即通道 A 和通道 B 状态组合）进行同步。

对于不同的计数方向，索引检测事件的作用会有所不同，从而确保了在速度反转期间能够执行对称操作：

- 计数器在递增计数期间复位（DIR 位 = 0）。
- 计数器在递减计数时设置为 TIMx_ARR。

因此，无论哪种计数方向，都可以在完全相同的机械角位置上生成索引。下面的*图 229* 通过一个简单的示例说明了生成索引的位置（每次机械旋转编码器提供 4 个边沿）。

图 229. IPOS[1:0] = 11 时的索引生成

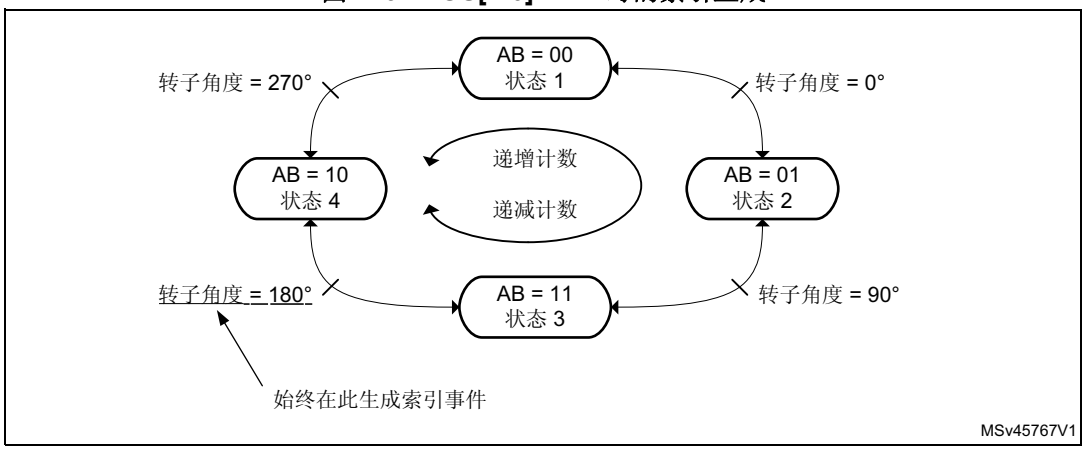


图 230 给出了 IPOS[1:0] = 11 时的波形和相应值。该图说明了计数器值根据计数方向强制自动调整的情况：

- 如果编码器状态为“11”（ChA = 1，ChB = 1）并且递增计数（DIR 位 = 0），则计数器设置为 0。
- 如果编码器退出“11”状态并且递减计数（DIR 位 = 1），则计数器设置为 TIMx_ARR。

可以在发生索引检测事件时发出中断。

箭头指示在哪次转换时生成索引事件中断。

图 230. 在通道 A 上对索引进行门控时的计数器读取 (IPOS[1:0] = 11)

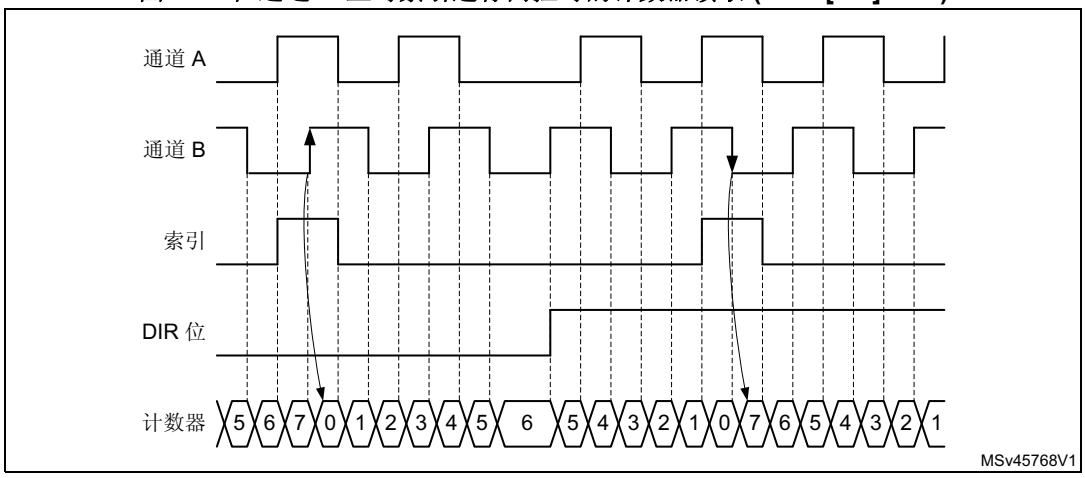
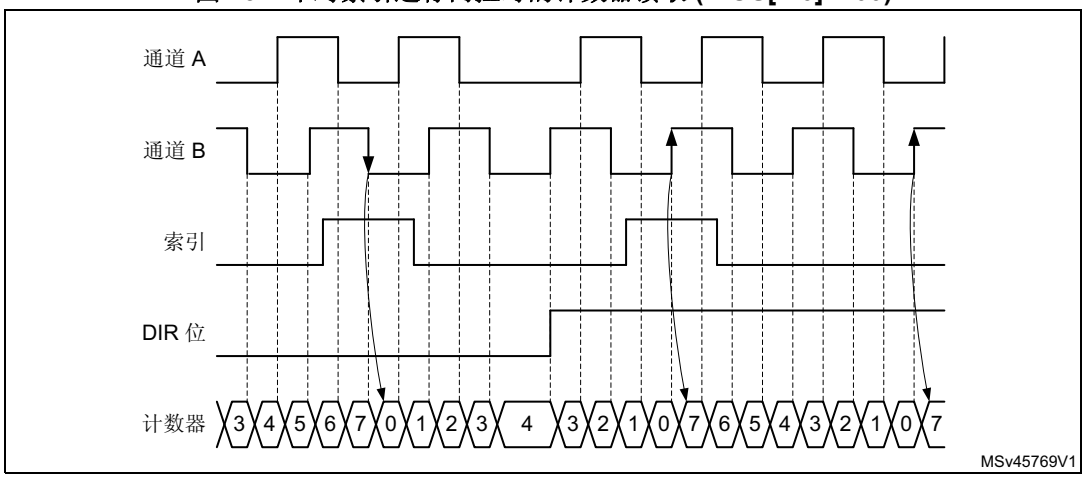


图 231. 显示了非门控模式下的波形和相应值。箭头指示在哪次转换时生成索引事件。

图 231. 不对索引进行门控时的计数器读取 (IPOS[1:0] = 00)



下面的*图 232.* 说明了在各种脉冲对齐场景下如何处理“在 A 和 B 上进行门控”模式。箭头指示在哪次转换时生成索引事件。

图 232. 在通道 A 和 B 上对索引进行门控时的计数器读取

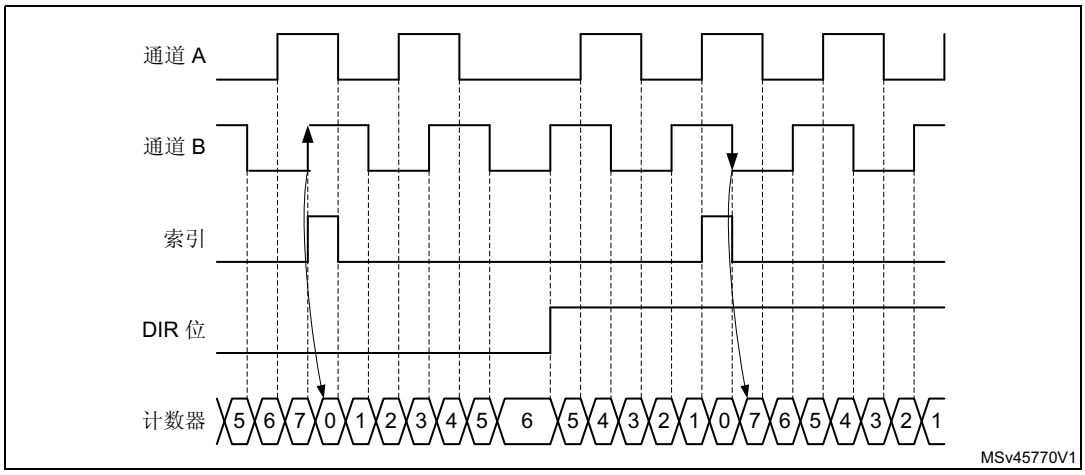


图 233 和*图 234* 详细说明了后续索引脉冲的宽度可能小于 1/4 个编码器时钟周期的情况。

图 233. 窄索引脉冲情况下的编码器模式行为 (IPOS[1:0] = 11)

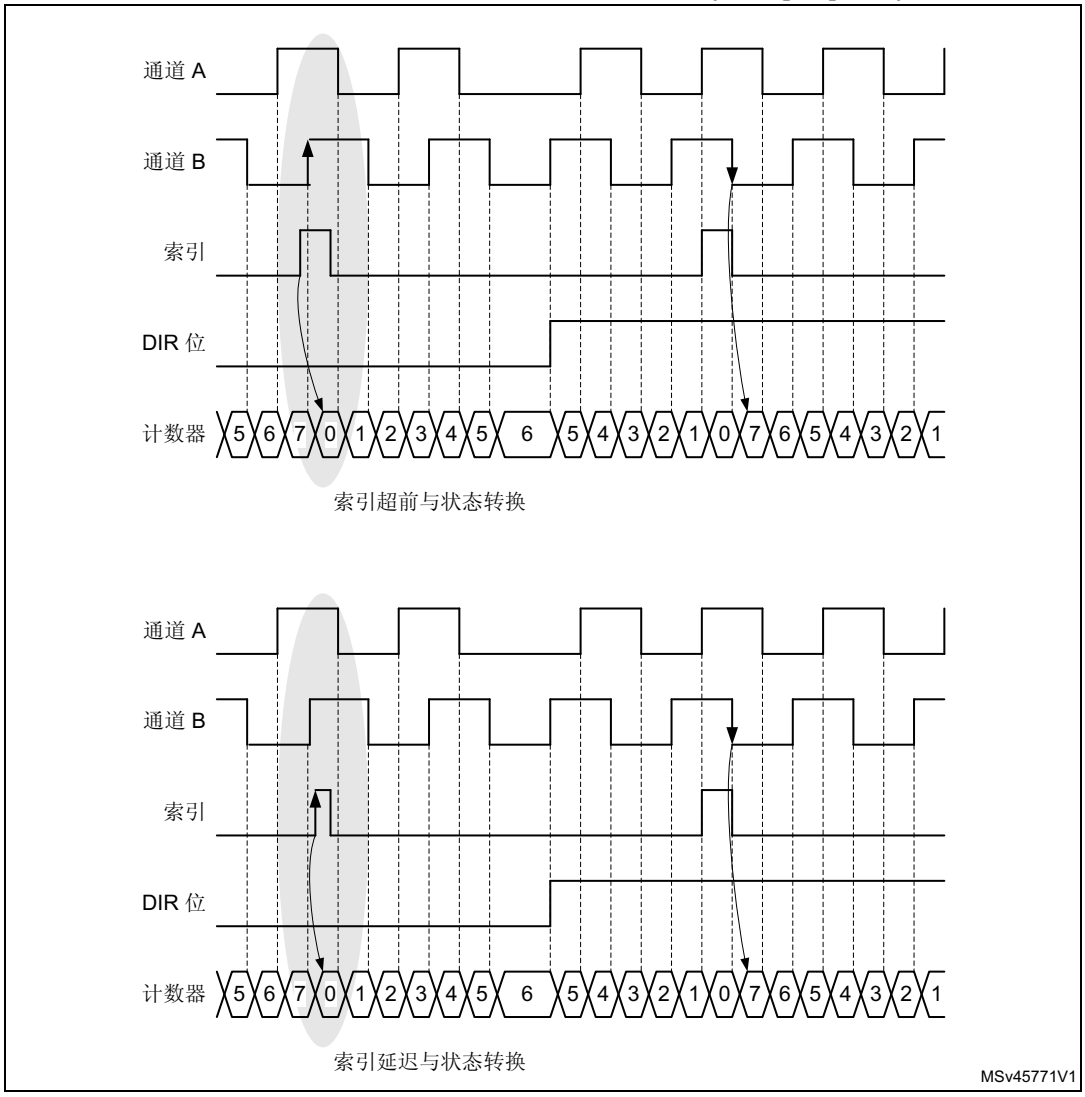


图 234. 计数器复位窄索引脉冲（放大视图，ARR = 0x07）

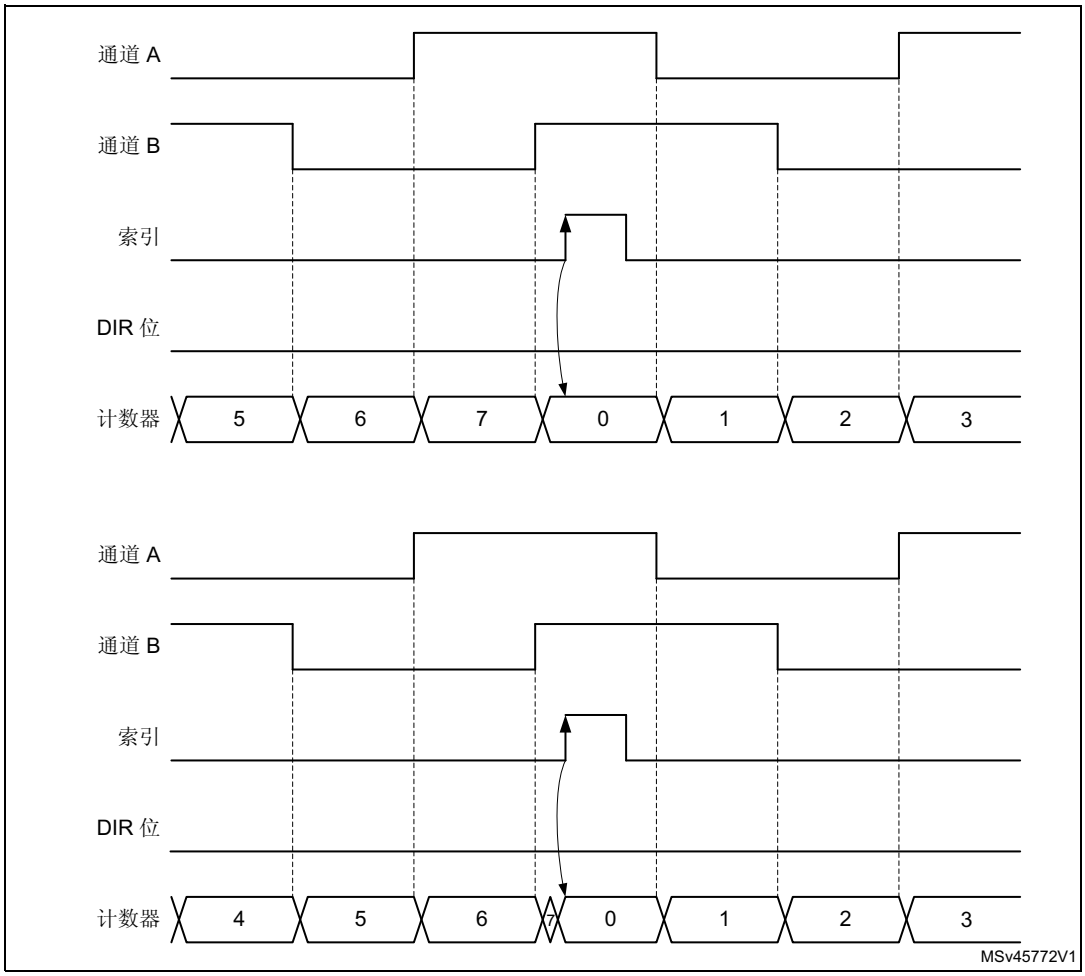
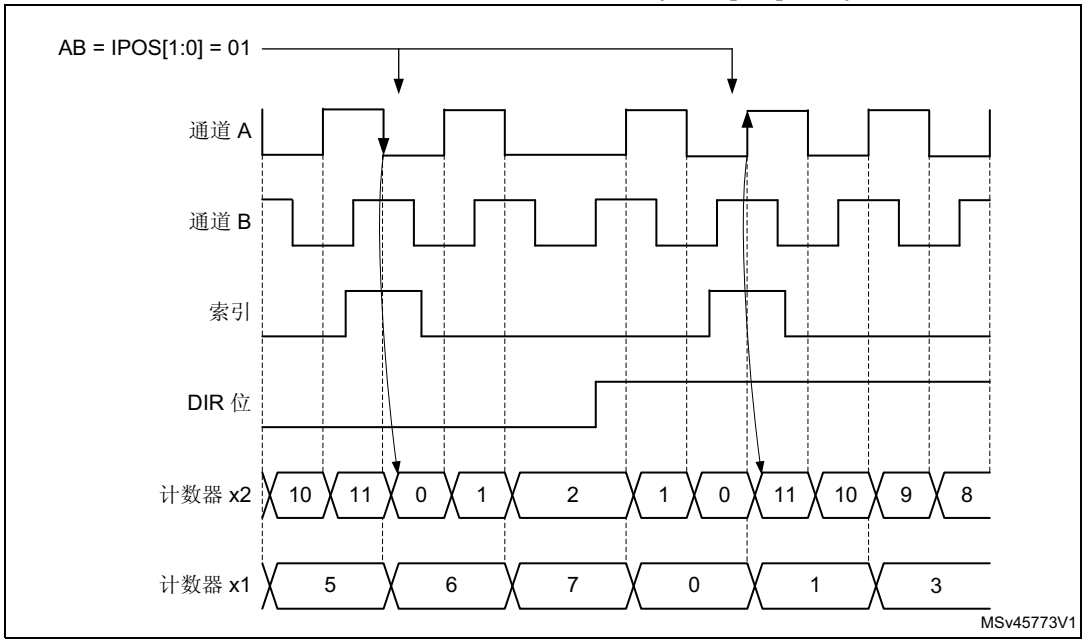


图 235 说明了如何在 x1 和 x2 模式下管理索引。

图 235. x1 和 x2 模式下的索引行为 (IPOS[1:0] = 01)

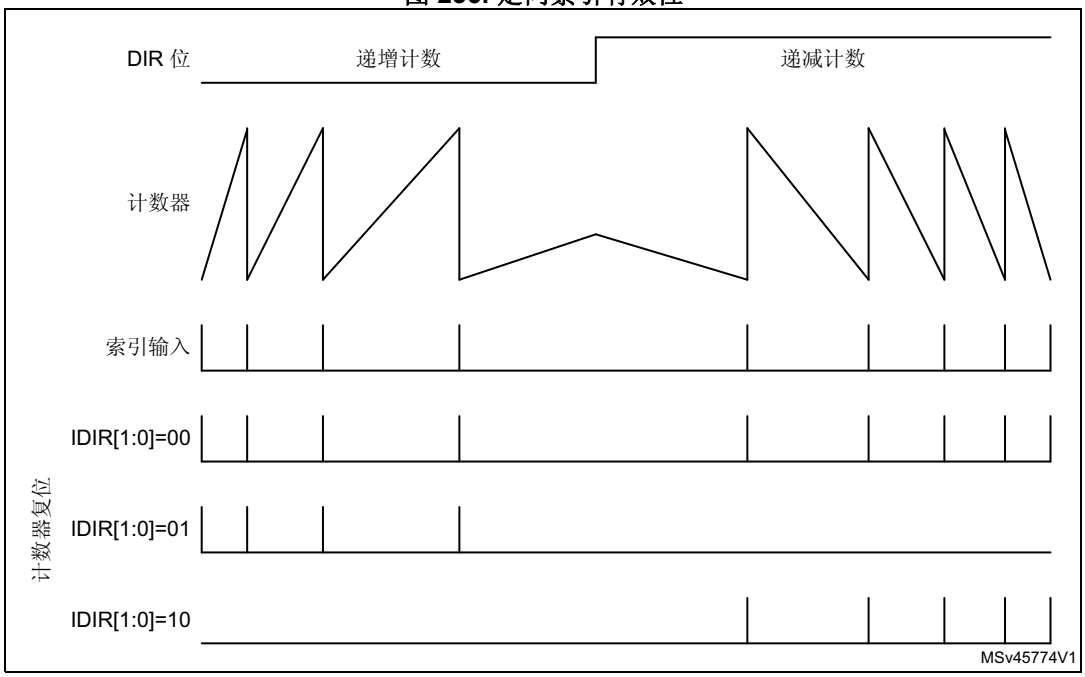


定向索引有效性

TIMx_ECR 寄存器中的 IDIR[1:0] 位域允许索引仅在选定的计数方向上有效。

图 236 显示了索引与计数器复位事件之间的关系，具体取决于 IDIR[1:0] 值。

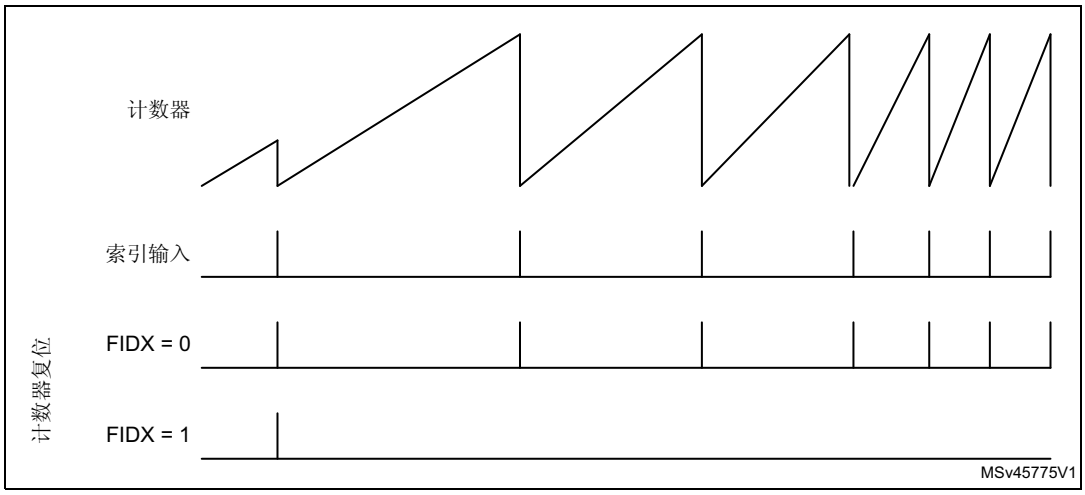
图 236. 定向索引有效性



特殊的第一个索引事件管理

TIMx_ECR 寄存器中的 FIDX 位允许仅获取一次索引，如下面的*图 237* 所示。第一个索引到达后，任何后续索引都将被忽略。如果需要，可以通过向 FIDX 位写入 0 然后再置 1 的方式重新启动电路。

图 237. 计数器复位与 FIDX 位置 1 的关系



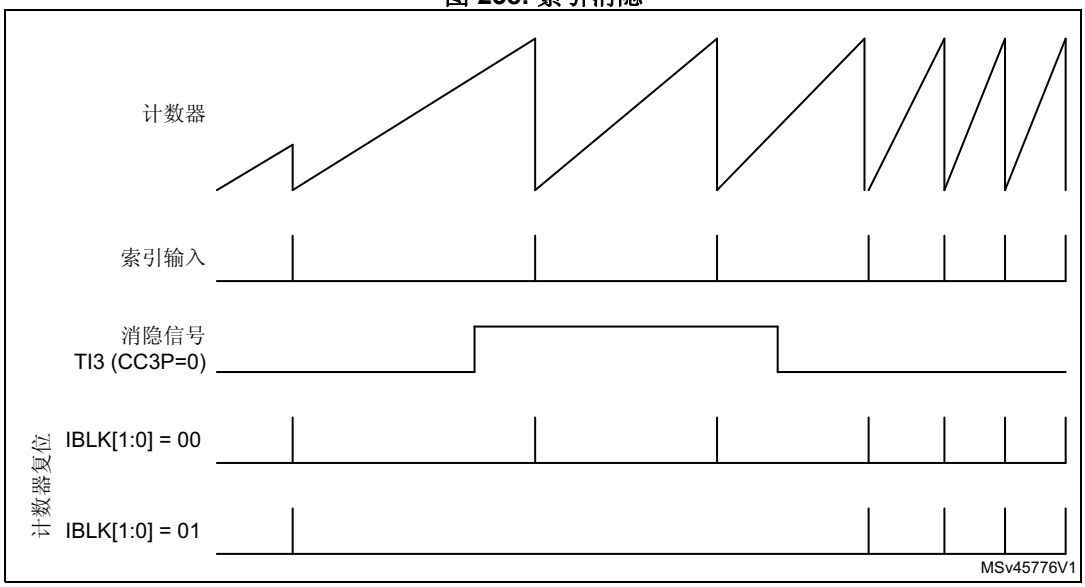
索引消隐

可使用 tim_ti3 或 tim_ti4 输入来消隐索引事件。在消隐窗口期间，索引事件不再复位计数器，如下面的*图 238* 所示。

该模式通过 TIMx_ECR 寄存器中的 IBLK[1:0] 位域使能，具体如下所述：

- IBLK[1:0] = 00：索引信号始终有效
- IBLK[1:0] = 01：tim_ti3 输入上的索引信号消隐
- IBLK[1:0] = 10：tim_ti4 输入上的索引信号消隐

图 238. 索引消隐



非正交模式下的索引管理

下面的*图 239* 和*图 240* 详细介绍了当 SMS[3:0] 位域等于 1010、1011、1100 和 1101 时，在定向时钟模式和时钟 + 方向模式下如何管理索引。

对于这两种模式，索引有效性通过 IPOS[0] 位设置，具体如下所述：

- IPOS[0] = 0：在时钟低电平时检测索引
- IPOS[0] = 1：在时钟高电平时检测索引

IPOS[1] 位无意义。

图 239. 时钟 + 方向模式下的索引行为 (IPOS[0] = 1)

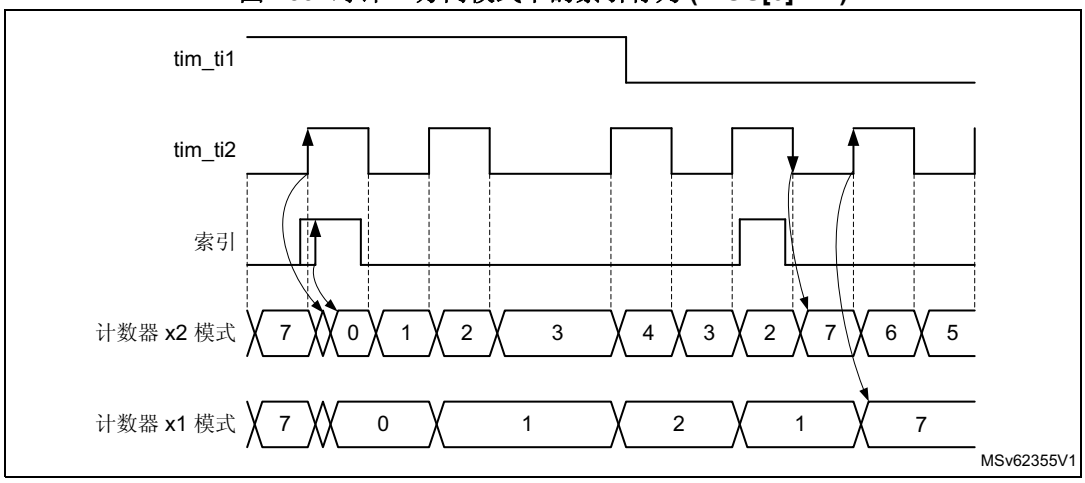
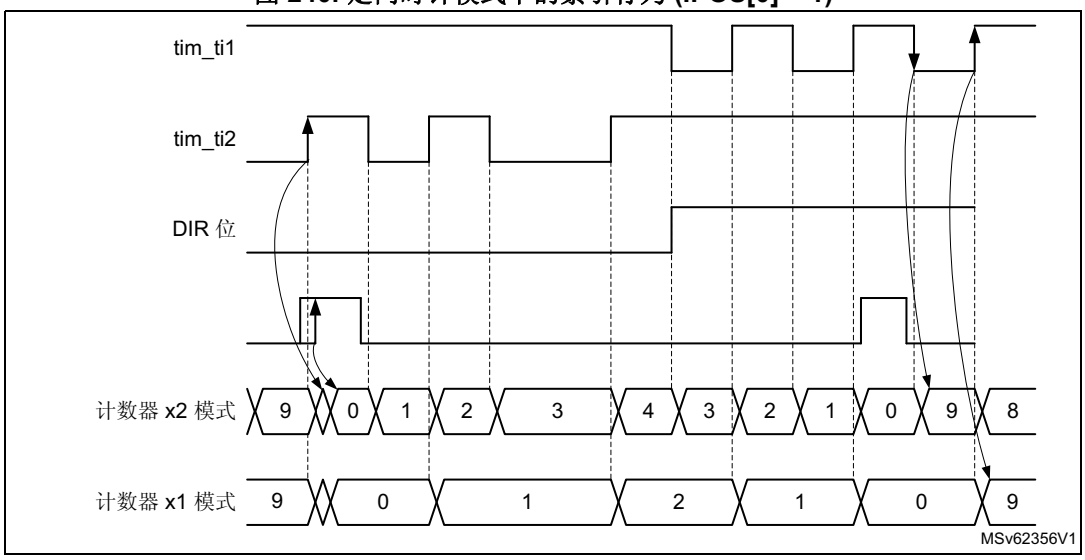


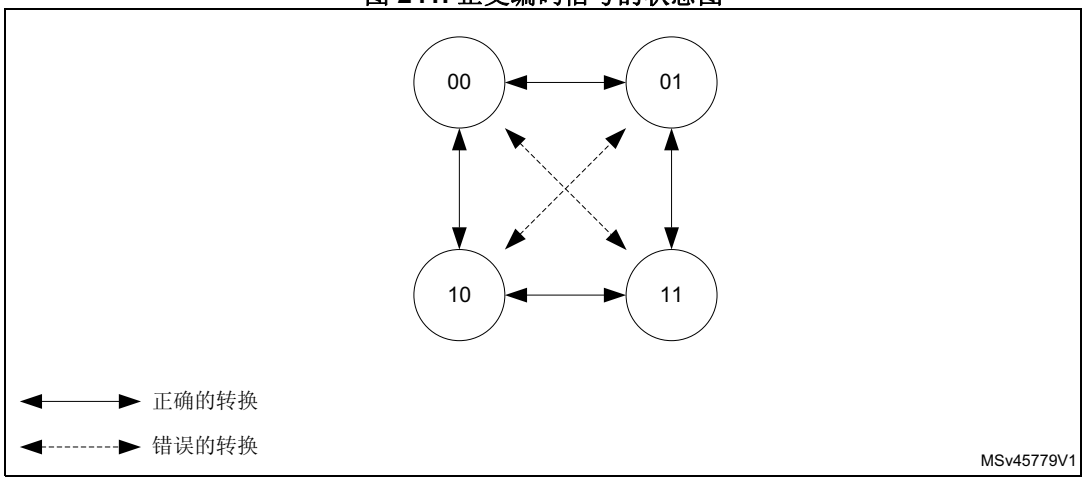
图 240. 定向时钟模式下的索引行为 (IPOS[0] = 1)



编码器错误管理

在使用 2 个正交信号的编码器配置下，可以检测转换错误。对 2 个输入的读操作对应于 2 位格雷码，可表示为状态图（如下面的*图 241.* 所示）。单个位会立即改变。错误的转换会将 TIMx_SR 状态寄存器中的 TERRF 中断标志置 1。如果 TIMx_DIER 寄存器中的 TERRIE 位置 1，将生成转换错误中断。

图 241. 正交编码信号的状态图



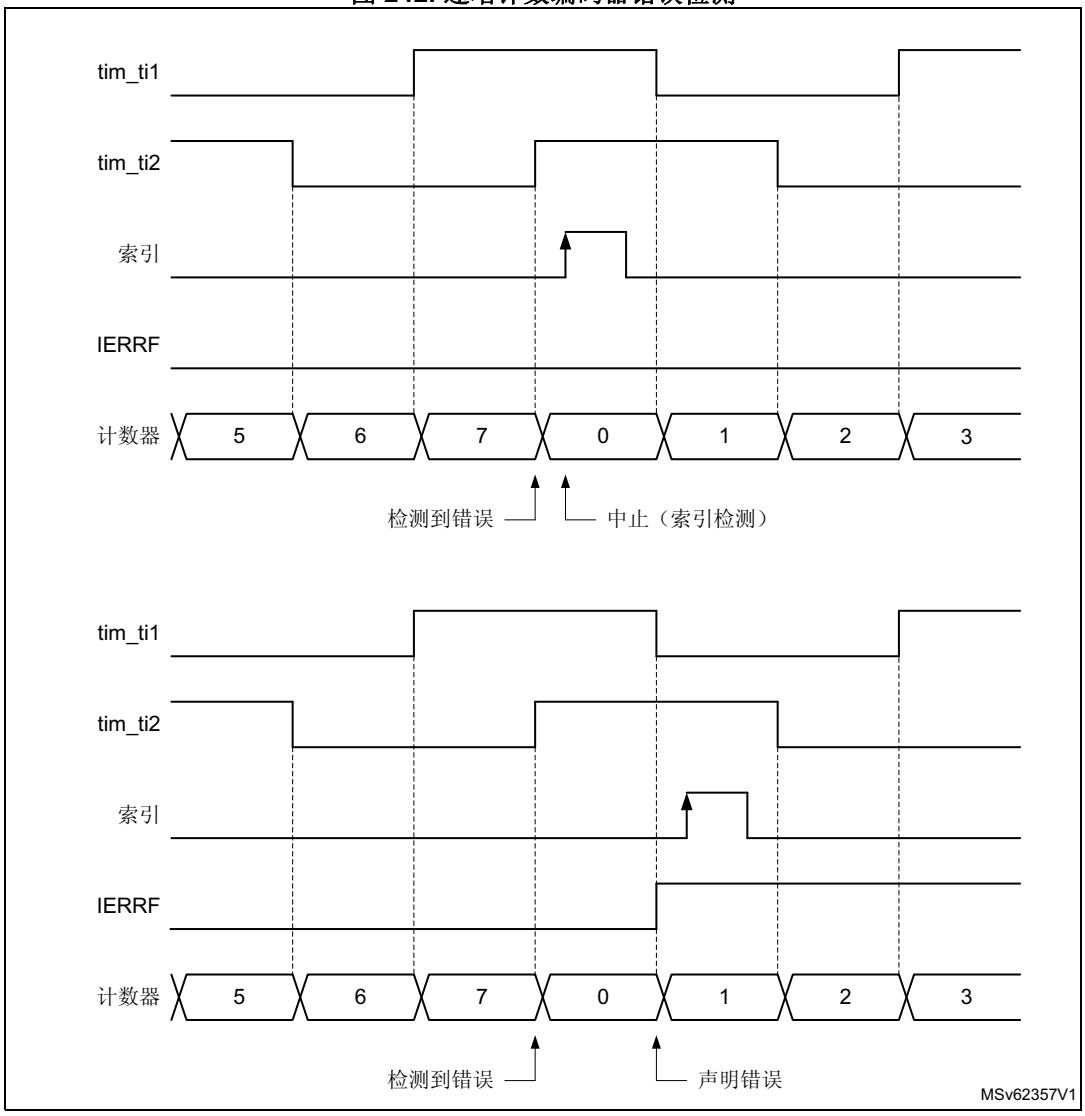
对于具有索引信号的编码器，可以检测会导致每转脉冲数过多的异常操作。每转 N 个脉冲的编码器每转可提供 4xN 个计数。索引信号每 4xN 个时钟周期复位一次计数器。

如果在没有发生任何索引事件的情况下计数器值从 TIMx_ARR 递增到 0 或从 0 递减到 TIMxARR 值，将报告索引位置错误。

上溢阈值使用 TIMx_ARR 寄存器编程设定。1000 线编码器生成的计数器值介于 0 与 3999 之间（在 4x 读取模式下）。上溢检测阈值必须通过设置 TIMx_ARR = 3999 + 1 = 4000 进行编程。

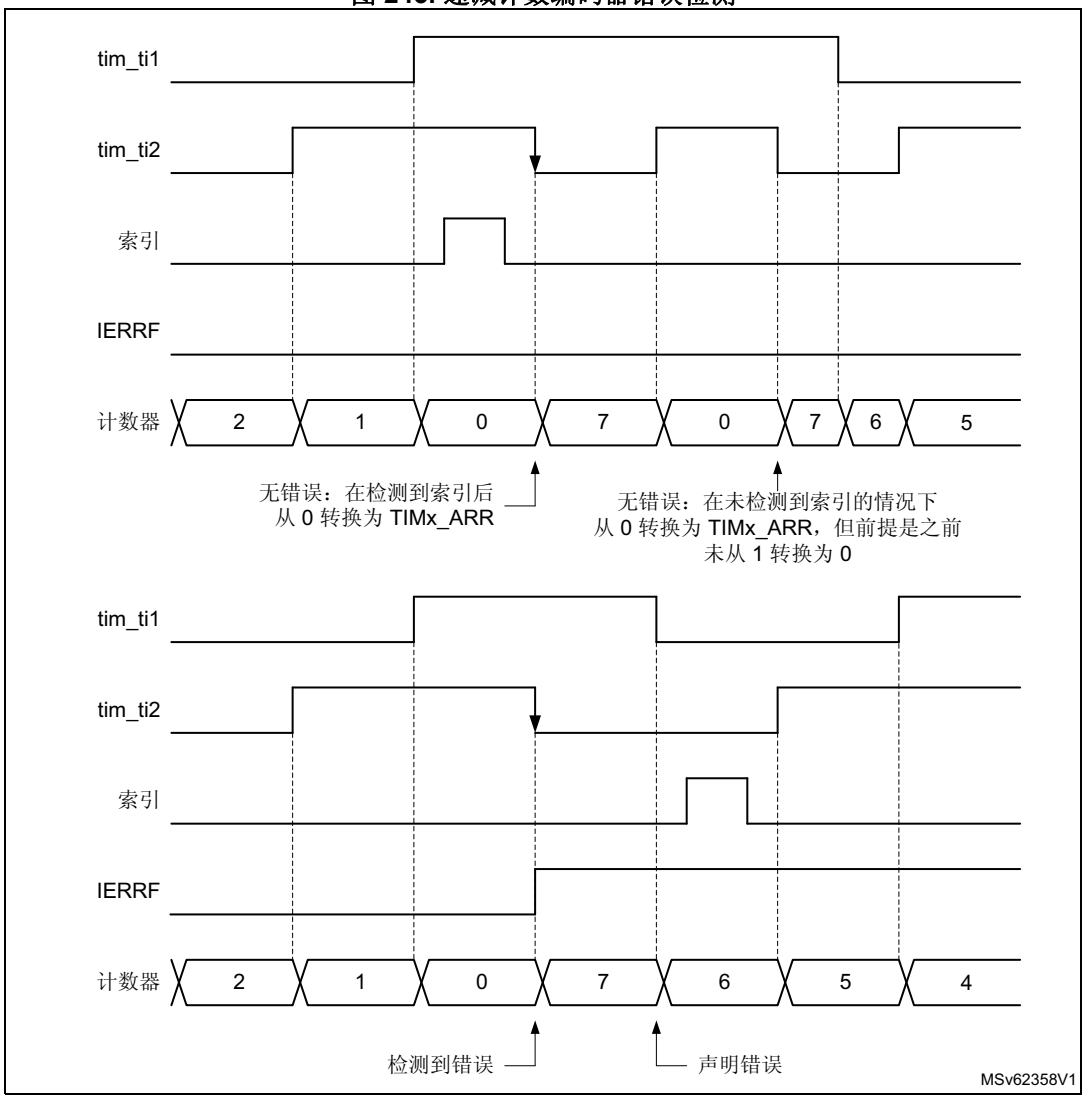
递增计数时，错误标志将延迟到从 0 到 1 的转换后再置 1。这是为了应对 A 和 B 门控模式下的窄索引脉冲，如下面的*图 242* 所示。

图 242. 递增计数编码器错误检测



在递减计数模式下，检测的触发条件是初次从 1 到 0 的转换。这是为了应对 A 和 B 门控模式下的窄索引脉冲（如下面的*图 243* 所示），以避免在检测到索引后编码器立即在 TIMx_ARR 与 0 之间抖动时发生误检。

图 243. 递减计数编码器错误检测



索引错误会将 TIMx_SR 状态寄存器中的 IERRF 中断标志置 1。如果 TIMx_DIER 寄存器中的 IERRIE 位置 1，将生成索引错误中断。

功能编码器中断

以下中断也可用于编码器模式

- 方向更改：编码器模式下计数方向的任何更改都会导致 TIMx_CR1 寄存器中的 DIR 位翻转。方向更改会将 TIMx_SR 状态寄存器中的 DIRF 中断标志置 1。如果 TIMx_DIER 寄存器中的 DIRIE 位置 1，将生成方向更改中断。
- 索引事件：索引事件会将 TIMx_SR 状态寄存器中的 IDXF 中断标志置 1。如果 TIMx_DIER 寄存器中的 IDXIE 位置 1，将生成索引中断。

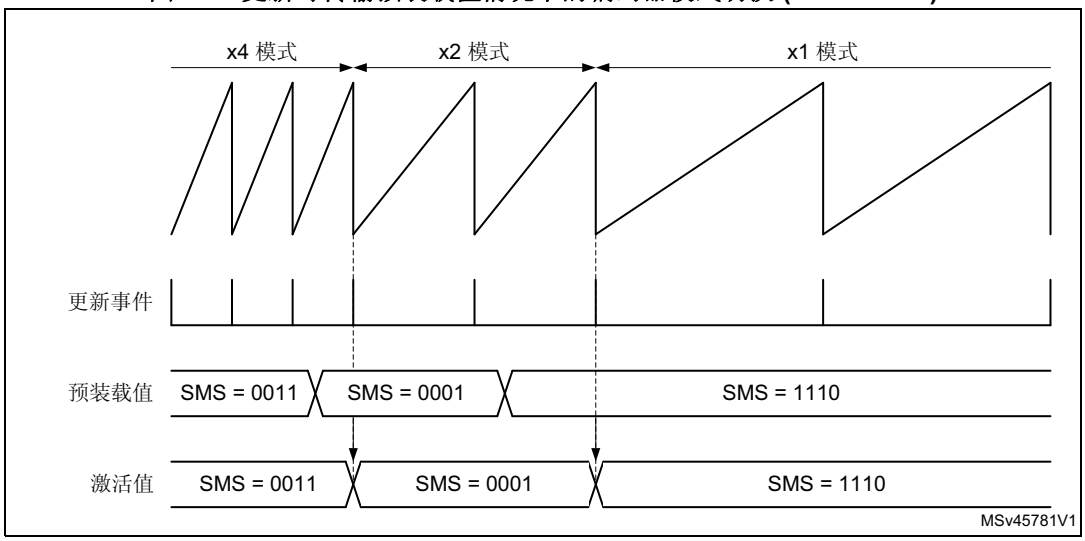
运行时编码器模式更新情况下的从模式选择预装载

在运行过程中，有时可能需要从一种编码器模式切换到另一种编码器模式。这通常发生在高速情况下，用于降低更新中断频率，先从 x4 模式切换到 x2 模式再切换到 x1 模式，如下面的*图 244* 所示。

为此，可以预装载 SMS[3:0] 位。该功能通过将 TIMx_SMCR 寄存器中的 SMSPE 使能位置 1 来使能。使用 TIMx_SMCR 寄存器中的 SMSPS 位，可以选择将 SMS[3:0] 预装载值传输到有效值的触发信号。

- SMSPS = 0：传输由计数器递增计数至上溢或递减计数至下溢时发生的更新事件 (UEV) 触发。
- SMSPS = 1：传输由索引事件触发。

图 244. 更新时传输预装载值情况下的编码器模式切换 (SMSPS = 0)



编码器时钟输出

编码器模式的工作原理并不是特别适合低速下的高分辨率速度测量，因为它需要相对较长的积分时间才能具有足够数量的时钟边沿并获取精确的测量结果。

在低速条件下，更好的解决方案是进行边沿到边沿的时钟周期测量。这可以使用从定时器来实现。定时器可通过 tim_trgo 输出来输出编码器时钟信息。之后，从定时器便可执行周期测量并提供每个编码器时钟边沿的速度信息。

该模式通过将 TIMx_CR2 寄存器中的 MMS[3:0] 位域设置为 1000 来使能，适用于以下 SMS[3:0] 值：0001、0010、0011、1010、1011、1100、1101、1110、1111。不允许使用任何其他 SMS[3:0] 代码，否则可能会导致意外行为。

26.3.26 方向位输出

定时器可通过 tim_oc3n 和 tim_oc4 输出信号（TIMx_CR1 寄存器中的 DIR 位的副本）来输出方向信号。这通过将 TIMx_CCMR2 寄存器中的 OC3M[3:0] 或 OC4M[3:0] 位域设置为 1011 来实现。

该功能可用于在编码器模式下监视计数方向（或旋转方向），或者用于在中心对齐 PWM 模式下获得指示递增/递减计数阶段的信号。

26.3.27 UIF 位重映射

TIMx_CR1 寄存器中的 IUFREMAP 位强制将更新中断标志 UIF 连续复制到定时器计数器寄存器的位 31 (TIMxCNT[31]) 中。这样便可按原子方式读取计数器值以及由 UIFCPY 标志发出的电位翻转条件。在特定情况下，这可避免在后台任务（计数器读）和中断（更新中断）之间共享处理时产生竞争条件，从而简化计算。

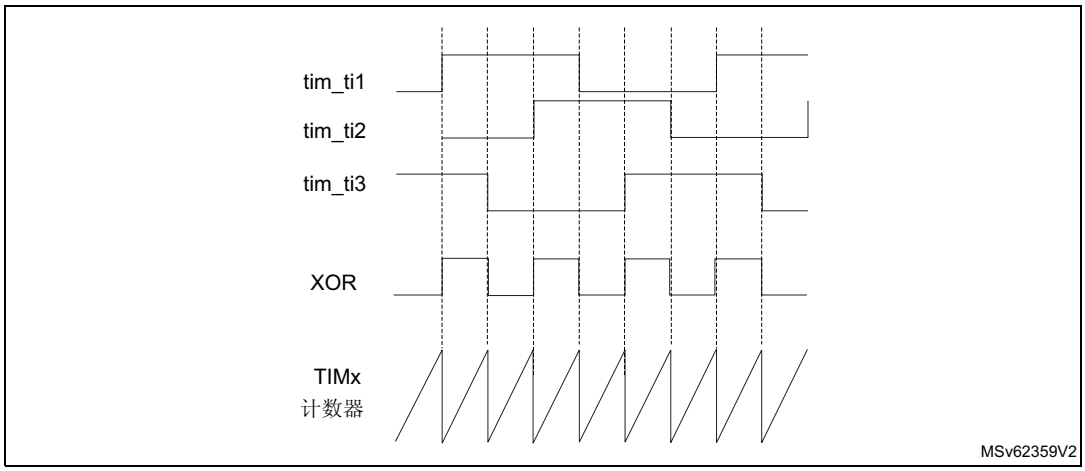
UIF 和 UIFCPY 标志之间的置位没有延迟。

26.3.28 定时器输入异或功能

通过 TIMx_CR2 寄存器中的 TI1S 位，可将通道 1 的输入滤波器连接到异或门的输出，从而将 tim_ti1、tim_ti2 和 tim_ti3 这三个输入引脚组合在一起。

异或输出可与触发或输入捕获等所有定时器输入功能配合使用。这样便于测量两个输入信号上边沿之间的间隔（如下面的*图 245* 所示）。

图 245. 测量 3 个信号上边沿之间的时间间隔



26.3.29 连接霍尔传感器

可通过用于产生电机驱动 PWM 信号的高级控制定时器以及*图 246* 中称为“接口定时器”的另一个定时器 TIMx，实现与霍尔传感器的连接。3 个定时器输入引脚（tim_ti1、tim_ti2 和 tim_ti3）通过异或门连接到 tim_ti1 输入通道（通过将 TIMx_CR2 寄存器中的 TI1S 位置 1 来选择），并由“接口定时器”进行捕获。

从模式控制器配置为复位模式；从输入为 tim_ti1f_ed。这样，每当 3 个输入中有一个输入发生切换时，计数器会从 0 开始重新计数。这样将产生由霍尔输入的任何变化而触发的时基。

在“接口定时器”上，捕获/比较通道 1 配置为捕获模式，捕获信号为 tim_trc（请参见*第 713 页的图 188：捕获/比较通道（示例：通道 1 输入阶段）*）。捕获值对应于输入上两次变化的间隔时间，可提供与电机转速相关的信息。

“接口定时器”可用于在输出模式下产生脉冲，以通过触发 COM 事件更改高级控制定时器各个通道的配置。高级控制定时器用于生成电机驱动 PWM 信号。为此，必须对接口定时器通道进行编程，以便在编程的延迟过后产生正脉冲（在输出比较或 PWM 模式中）。该脉冲通过 tim_trgo 输出发送到高级控制定时器。

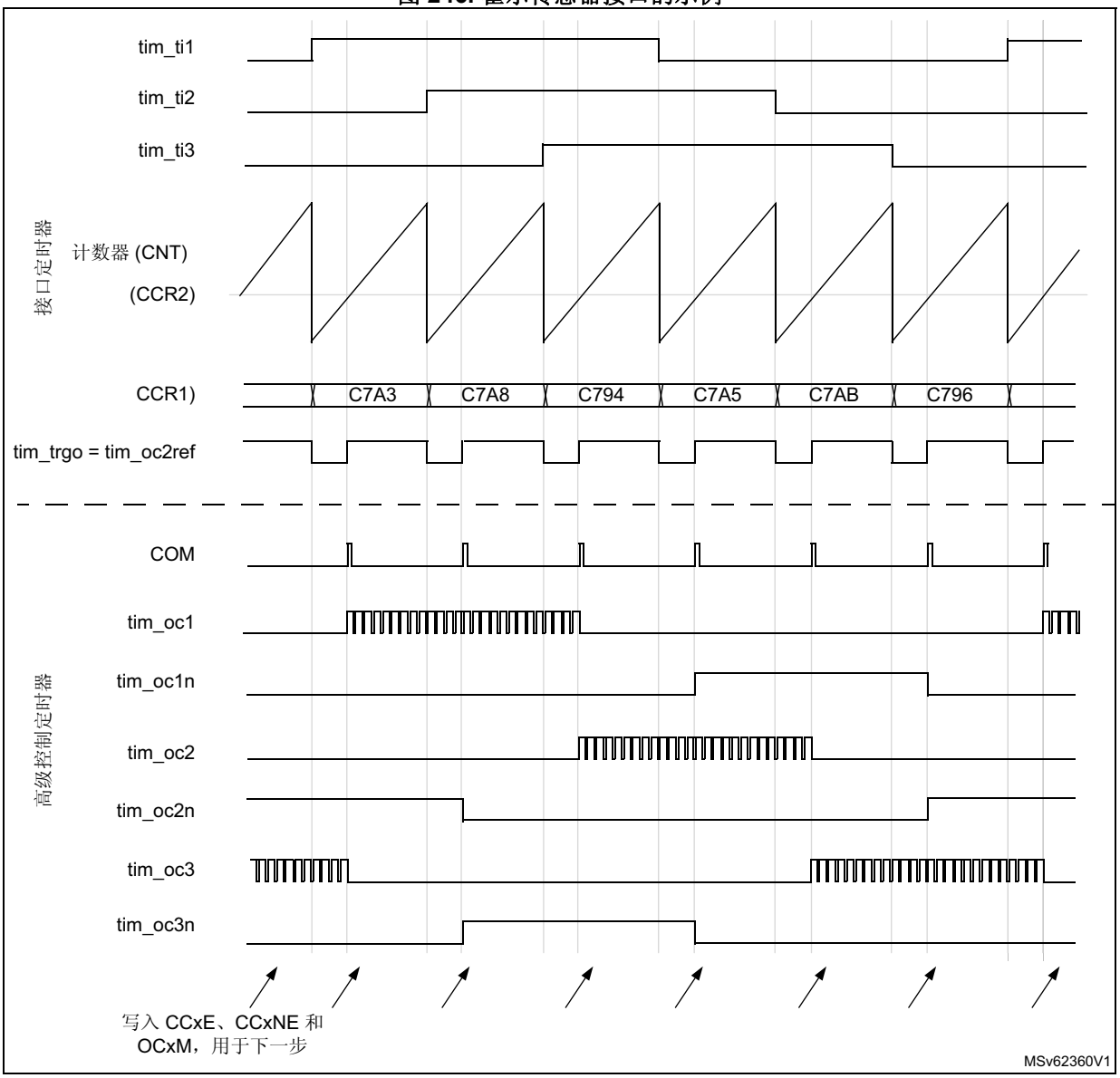
示例：霍尔输入与一个 TIMx 定时器相连接，每当霍尔输入发生更改，需要在所编程的延迟过后更改高级控制定时器的 PWM 配置。

- 向 TIMx_CR2 寄存器的 TI1S 位写入“1”，使 3 个定时器输入经过异或运算后进入 tim_ti1 输入通道。
- 时基编程：向 TIMx_ARR 写入其最大值（计数器必须通过 tim_ti1 的变化清零）。设置预分频器，以得到最大计数器周期，该周期长于传感器上两次变化的间隔时间。
- 将通道 1 编程为捕获模式（选择 tim_trc）：向 TIMx_CCMR1 寄存器的 CC1S 位写入“01”。如果需要，还可以编程数字滤波器。
- 将通道 2 编程为 PWM 2 模式，并具有所需延迟：向 TIMx_CCMR1 寄存器的 OC2M 位写入“111”，CC2S 位写入“00”。
- 选择 tim_oc2ref 作为 tim_trgo 上的触发输出：向 TIMx_CR2 寄存器的 MMS 位写入“101”。

在高级控制定时器中，必须选择正确的 tim_itrx 输入作为触发输入，定时器编程为可产生 PWM 信号，捕获/比较控制信号进行预装载（TIMx_CR2 寄存器的 CCPC = 1），并且 COM 事件由触发输入控制（TIMx_CR2 寄存器中 CCUS = 1）。发生 COM 事件后，在 PWM 控制位（CCxE、OCxM）中写入下一步的配置，此操作可在由 tim_oc2ref 上升沿产生的中断子程序中完成。

图 246 为本示例的示意图。

图 246. 霍尔传感器接口的示例



26.3.30 定时器同步

TIMx 定时器从内部连接在一起，以实现定时器同步或链接。有关详细信息，请参见*第 27.4.23 节：定时器同步*。它们可在以下几种模式下实现同步：复位模式、门控模式、触发模式、复位 + 触发模式和门控 + 复位模式。

从模式：复位模式

当触发输入信号发生变化时，计数器及其预分频器可重新初始化。此外，如果 TIMx_CR1 寄存器中的 URS 位处于低电平，则会生成更新事件 UEV。然后，所有预装载寄存器（TIMx_ARR 和 TIMx_CCRx）都将更新。

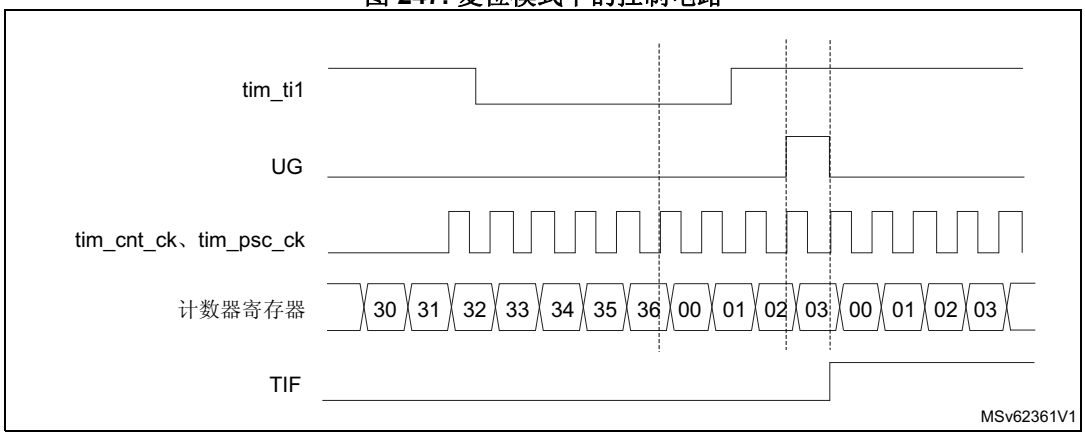
在以下示例中，tim_ti1 输入上出现上升沿时，递增计数器清零：

- 将通道 1 配置为检测 tim_ti1 上的上升沿。配置输入滤波时间（本例中不需要任何滤波器，因此保持 IC1F=0000）。由于捕获预分频器不用于触发操作，因此无需对其进行配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中的 CC1S = 01。在 TIMx_CCER 寄存器中写入 CC1P=0 和 CC1NP=“0”，验证极性（仅检测上升沿）。
- 在 TIMx_SMCR 寄存器中写入 SMS=100，将定时器配置为复位模式。在 TIMx_SMCR 寄存器中写入 TS=00101，选择 tim_ti1 作为输入源。
- 在 TIMx_CR1 寄存器中写入 CEN=1，启动计数器。

计数器开始根据内部时钟计数，然后正常运转，直到出现 tim_ti1 上升沿。当 tim_ti1 出现上升沿时，计数器清零，然后重新从 0 开始计数。同时，触发标志（TIMx_SR 寄存器中的 TIF 位）置 1，使能中断或 DMA 后，还可发送中断或 DMA 请求（取决于 TIMx_DIER 寄存器中的 TIE 和 TDE 位）。

下图显示了自动重载寄存器 TIMx_ARR=0x36 时的相关行为。tim_ti1 的上升沿与实际计数器复位之间的延迟是由于 tim_ti1 输入的重新同步电路引起的。

图 247. 复位模式下的控制电路



从模式：门控模式

输入信号的电平可用来使能计数器。

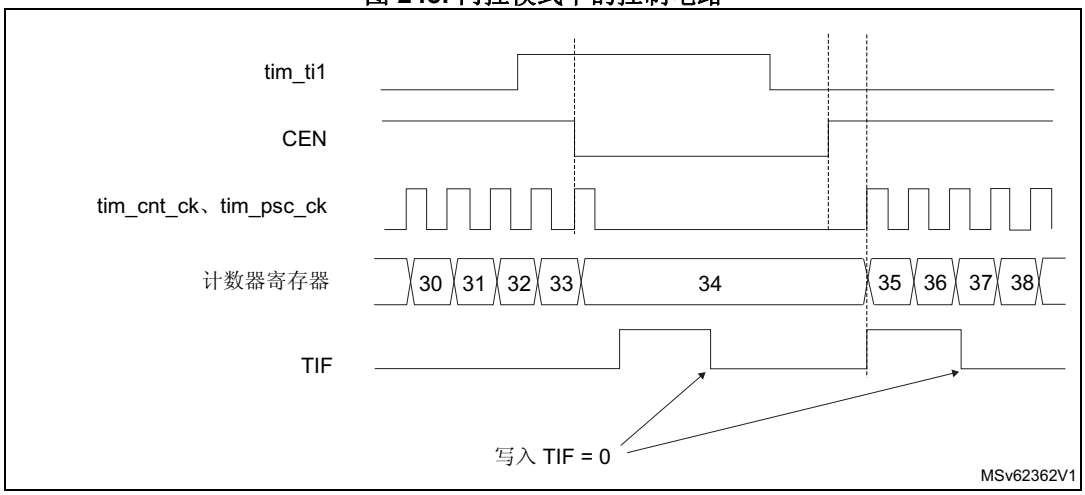
在以下示例中，递增计数器仅在 tim_ti1 输入为低电平时计数：

- 将通道 1 配置为检测 tim_ti1 上的低电平。配置输入滤波时间（本例中不需要任何滤波器，因此保持 IC1F=0000）。由于捕获预分频器不用于触发操作，因此无需对其进行配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中的 CC1S=01。在 TIMx_CCER 寄存器中写入 CC1P=1 和 CC1NP=“0”，以确定极性（仅检测低电平）。
- 在 TIMx_SMCR 寄存器中写入 SMS=101，将定时器配置为门控模式。在 TIMx_SMCR 寄存器中写入 TS=00101，选择 tim_ti1 作为输入源。
- 在 TIMx_CR1 寄存器中写入 CEN=1，使能计数器（在门控模式下，如果 CEN=0，则无论触发输入电平如何，计数器都不启动）。

只要 tim_ti1 为低电平，计数器就开始根据内部时钟计数，直到 tim_ti1 变为高电平时停止计数。计数器启动或停止时，TIMx_SR 寄存器中的 TIF 标志都会置 1。

tim_ti1 的上升沿与实际计数器停止之间的延迟是由于 tim_ti1 输入的重新同步电路引起的。

图 248. 门控模式下的控制电路



从模式：触发模式

所选输入上发生某一事件时可以启动计数器。

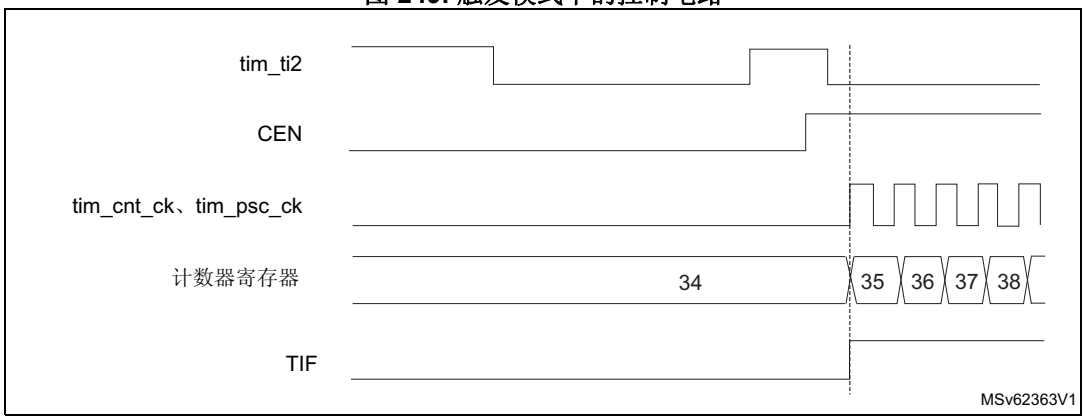
在以下示例中，tim_ti2 输入上出现上升沿时，递增计数器启动：

- 将通道 2 配置为检测 tim_ti2 上的上升沿。配置输入滤波时间（本例中不需要任何滤波器，因此保持 IC2F=0000）。由于捕获预分频器不用于触发操作，因此无需对其进行配置。CC2S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中的 CC2S=01。在 TIMx_CCER 寄存器中写入 CC2P=1 和 CC2NP=0，以确定极性（仅检测低电平）。
- 在 TIMx_SMCR 寄存器中写入 SMS=110，将定时器配置为触发模式。在 TIMx_SMCR 寄存器中写入 TS=00110，选择 tim_ti2 作为输入源。

当 tim_ti2 出现上升沿时，计数器开始根据内部时钟计数，并且 TIF 标志置 1。

tim_ti2 的上升沿与实际计数器启动之间的延迟是由于 tim_ti2 输入的重新同步电路引起的。

图 249. 触发模式下的控制电路



从模式：复位 + 触发组合模式

在这种情况下，在出现所选触发输入 (tim_trgi) 上升沿时，重新初始化计数器，生成一个寄存器更新事件，并启动计数器。

该模式用于单脉冲模式。

从模式：门控 + 复位组合模式

触发输入 (tim_trgi) 为高电平时使能计数器时钟。只要触发输入变为低电平，计数器立即停止计数并复位。计数器的启动和停止都被控制。

该模式用于检测超范围 PWM 信号（占空比超过最大预期值）。

从模式：外部时钟模式 2 + 触发模式

外部时钟模式 2 可与另一种从模式（外部时钟模式 1 和编码器模式除外）结合使用。这种情况下，tim_etr_in 信号用作外部时钟输入，在复位模式、门控模式或触发模式下工作时，可选择另一个输入作为触发输入。不建议通过 TIMx_SMCR 寄存器中的 TS 位来选择 tim_etr_in 作为 tim_trgi。

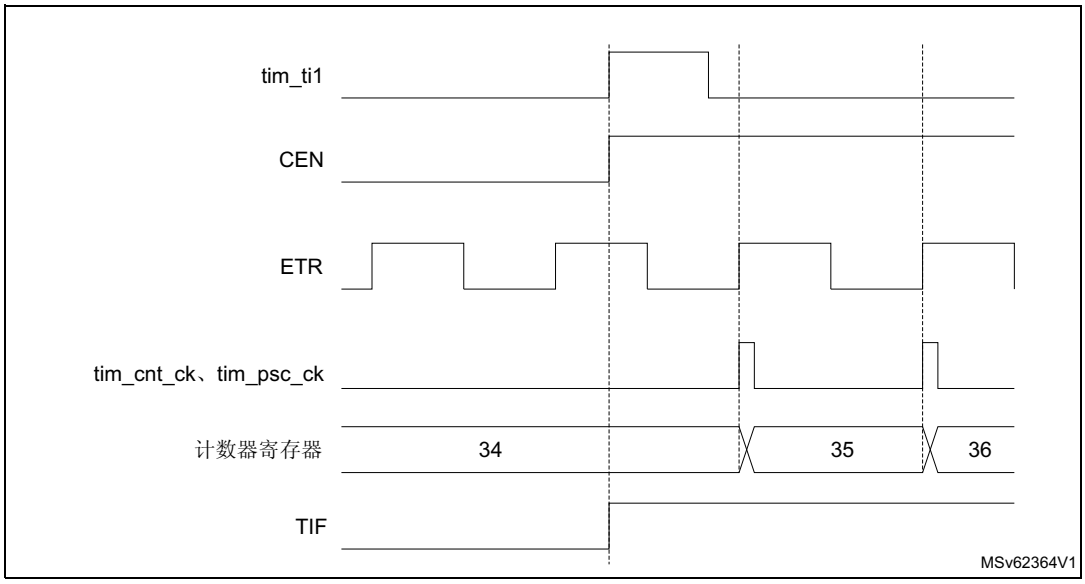
在以下示例中，只要 tim_ti1 出现上升沿，递增计数器即会在 tim_etr_in 信号的每个上升沿处递增：

1. 通过对 TIMx_SMCR 寄存器进行如下编程，配置外部触发输入电路：
   - ETF = 0000：无滤波器
   - ETPS = 00：禁止预分频器
   - ETP = 0：检测 tim_etr_in 的上升沿，并写入 ECE=1，以使能外部时钟模式 2。
2. 如下配置通道 1，以检测 TI 的上升沿：
   - IC1F=0000：无滤波器。
   - 由于捕获预分频器不用于触发操作，因此无需对其进行配置。
   - TIMx_CCMR1 寄存器中 CC1S=01，只选择输入捕获源。
   - TIMx_CCER 寄存器中 CC1P=0 且 CC1NP=“0”，以确定极性（仅检测上升沿）。
3. 在 TIMx_SMCR 寄存器中写入 SMS=110，将定时器配置为触发模式。在 TIMx_SMCR 寄存器中写入 TS=00101，选择 tim_ti1 作为输入源。

tim_ti1 出现上升沿时将使能计数器并且 TIF 标志置 1。然后计数器在 tim_etr_in 出现上升沿时计数。

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

图 250. 外部时钟模式 2 + 触发模式下的控制电路



注意：必须先使能接收 tim_trgo 或 tim_trgo2 信号的从外设（定时器、ADC 等）的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改时钟频率（预分频器）。

26.3.31 ADC 触发

定时器可通过多种内部信号产生 ADC 触发事件，例如复位、使能或比较事件。也可生成由内部边沿检测器发出的脉冲，例如：

- OC4ref 的上升沿和下降沿
- OC5ref 上的上升沿或 OC6ref 上的下降沿

在重定向到 ADC 的 tim_trgo2 内部线路上发出触发信号。共有 16 个可能的事件，它们可通过 TIMx_CR2 寄存器中的 MMS2[3:0] 位选择。

第 729 页的图 203 给出了三相电机驱动的应用示例。

注意：必须先使能接收 tim_trgo 或 tim_trgo2 信号的从外设（定时器、ADC 等）的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改时钟频率（预分频器）。

注意：必须先使能 ADC 时钟，才能从主定时器接收事件；从定时器接收触发信号时，不得实时更改 ADC 时钟。

26.3.32 DMA 分组传送模式

TIMx 定时器能够根据一个事件生成多个 DMA 请求。主要目的是能够对定时器的一部分多次重新编程而无需软件开销，但也可以用于定期地一次性地批量读取定时器的多个寄存器的内容。

DMA 控制器访问目标唯一，必须指向虚拟寄存器 TIMx_DMAR。发生给定的定时器事件时，定时器会启动 DMA 请求序列（分组）。每次写入 TIMx_DMAR 寄存器都会重定向到定时器其中的一个寄存器。

TIMx_DCR 寄存器中的 DBL[4:0] 位设置 DMA 连续传送长度。当对 TIMx_DMAR 地址进行读或写访问时，定时器进行一次分组传送，即传送次数（按半字或字节）。

TIMx_DCR 寄存器中的 DBA[4:0] 位定义 DMA 传送的 DMA 基址（通过 TIMx_DMAR 地址执行读/写访问时）。DBA 定义为从 TIMx_CR1 寄存器地址开始计算的偏移量：

示例：

00000：TIMx_CR1

00001：TIMx_CR2

00010：TIMx_SMCR

TIMx_DCR 寄存器中的 DBSS[3:0] 位定义触发 DMA 分组传送的中断源（详细信息，请参见 *第 26.6.29 节：TIM1 DMA 控制寄存器 (TIM1_DCR)*）。

例如，定时器 DMA 分组传送功能用于在发生更新事件后将 CCRx 寄存器（x = 2、3、4）的内容更新为通过 DMA 传输到 CCRx 寄存器中的多个半字。

具体操作步骤如下：

1. 将相应的 DMA 通道配置如下：
   - DMA 通道外设地址为 DMAR 寄存器地址。
   - DMA 通道存储器地址为包含要通过 DMA 传输到 CCRx 寄存器的数据的 RAM 缓冲区地址。
   - 要传输的数据量 = 3（参见下文注释）。
   - 禁止循环模式。
2. 通过将 DBA 和 DBL 位域配置如下来配置 DCR 寄存器：
   DBL = 3 次传输，DBA = 0xE 并且 DBSS = 1。
3. 使能 TIMx 更新 DMA 请求（DIER 寄存器中的 UDE 位置 1）。
4. 使能 TIMx。
5. 使能 DMA 通道。

本例适用于每个 CCRx 寄存器只更新一次的情况。如果每个 CCRx 寄存器要更新两次，则要传输的数据量必须为 6。下面以包含 data1、data2、data3、data4、data5 和 data6 的 RAM 缓冲区为例。数据将按照如下方式传输到 CCRx 寄存器：在第一个更新 DMA 请求期间，data1 传输到 CCR2，data2 传输到 CCR3，data3 传输到 CCR4；在第二个更新 DMA 请求期间，data4 传输到 CCR2，data5 传输到 CCR3，data6 传输到 CCR4。

注意： *可以将空值写入保留的寄存器中。*

26.3.33 TIM1DMA 请求

TIM1 可生成 DMA 请求，如下表所示。

表 179. DMA 请求

DMA 请求信号	DMA 请求	使能控制位
tim_upd_dma	更新	UDE
tim_cc1_dma	捕获/比较 1	CC1DE
tim_cc2_dma	捕获/比较 2	CC2DE
tim_cc3_dma	捕获/比较 3	CC3DE
tim_cc4_dma	捕获/比较 4	CC4DE
tim_com_dma	换向 (COM)	COMDE
tim_trg_dma	触发	TDE

26.3.34 调试模式

当微控制器进入调试模式（Cortex-M33 内核停止）时，TIMx 计数器可以根据 DBG 模块中的 DBG_TIMx_STOP 配置位选择继续正常工作或者停止工作。

调试模式下的行为可以使用调试支持 (DBG) 模块中每个定时器的专用配置位进行编程。

为了安全起见，当计数器停止时，输出被禁止（就像 MOE 位被复位一样）。可以将输出强制变为无效状态（OSSI 位 = 1），或者通过 GPIO 控制器（OSSI 位 = 0）来控制输出，通常将其强制为高阻态。

更多详细信息，请参见调试支持 (DBG) 部分。

26.4 TIM1 低功耗模式

表 180. 低功耗模式对 TIM1 的影响

模式	说明
睡眠	无影响，外设处于激活状态。中断可使器件退出睡眠模式。
停止	定时器停止工作，寄存器的内容保持不变。不产生任何中断。
待机	定时器掉电，退出待机模式后必须重新初始化。

26.5 TIM1 中断

TIM1 可产生多个中断，如*表 181* 所示。

表 181. 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止模式和待机模式
TIM_UP	更新	UIF	UIE	向 UIF 中写入 0	是	否
TIM_CC	捕获/比较 1	CC1IF	CC1IE	向 CC1IF 中写入 0	是	否
	捕获/比较 2	CC2IF	CC2IE	向 CC2IF 中写入 0	是	否
	捕获/比较 3	CC3IF	CC3IE	向 CC3IF 中写入 0	是	否
	捕获/比较 4	CC4IF	CC4IE	向 CC4IF 中写入 0	是	否
TIM_TRG_COM	换向 (COM)	COMIF	COMIE	向 COMIF 中写入 0	是	否
	触发	TIF	TIE	向 TIF 中写入 0	是	否
TIM_DIR_IDX	索引	IDXF	IDXIE	向 IDXF 中写入 0	是	否
	方向	DIRF	DIRIE	向 DIRF 中写入 0	是	否
TIM_BRK	刹车	BIF	BIE	向 BIF 中写入 0	是	否
	刹车 2	B2IF		向 B2IF 中写入 0	是	否
	系统刹车	SBIF		向 SBIF 中写入 0	是	否
TIM_IERR	索引错误	IERRF	IERRIE	向 IERRF 中写入 0	是	否
TIM_TER	转换错误	TERRF	TERRIE	向 TERRF 中写入 0	是	否

26.6 TIM1 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.2 节*。

26.6.1 TIM1 控制寄存器 1 (TIM1_CR1)

TIM1 control register 1

偏移地址：0x000

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DITHEN	UIFREMAP	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
			rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15:13 保留，必须保持复位值。

位 12 **DITHEN**：抖动使能 (Dithering enable)

0：禁止抖动
1：使能抖动

注意：只有在 CEN 位复位时才能修改 DITHEN 位。

位 11 **UIFREMAP**：UIF 状态位重映射 (UIF status bit remapping)

0：无重映射。UIF 状态位不复制到 TIMx_CNT 寄存器的位 31。
1：使能重映射。UIF 状态位复制到 TIMx_CNT 寄存器的位 31。

位 10 保留，必须保持复位值。

位 9:8 **CKD[1:0]**：时钟分频 (Clock division)

该位域指示定时器时钟 (tim_ker_ck) 频率与死区发生器以及数字滤波器（tim_etr_in、tim_tix）所使用的死区及采样时钟 (t_{DTS}) 之间的分频比。

00：$t_{DTS}=t_{tim_ker_ck}$
01：$t_{DTS}=2*t_{tim_ker_ck}$
10：$t_{DTS}=4*t_{tim_ker_ck}$
11：保留，不要设置成此值

位 7 **ARPE**：自动重载预装载使能 (Auto-reload preload enable)

0：TIMx_ARR 寄存器不进行缓冲
1：TIMx_ARR 寄存器进行缓冲

位 6:5 **CMS[1:0]**：中心对齐模式选择 (Center-aligned mode selection)

00：边沿对齐模式。计数器根据方向位 (DIR) 递增计数或递减计数。
01：中心对齐模式 1。计数器交替进行递增计数和递减计数。仅当计数器递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志才置 1。
10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志才置 1。
11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志都会置 1。

注意：只要计数器处于使能状态 (CEN=1)，就不得从边沿对齐模式切换为中心对齐模式。

位 4 **DIR**：方向 (Direction)

0：计数器递增计数
1：计数器递减计数

注意：当定时器配置为中心对齐模式或编码器模式时，该位为只读状态。

位 3 **OPM**：单脉冲模式 (One-pulse mode)

0：计数器在发生更新事件时不会停止计数

1：计数器在发生下一更新事件时停止计数（将 CEN 位清零）

位 2 **URS**：更新请求源 (Update request source)

该位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能时，所有以下事件都会生成更新中断或 DMA 请求。此类事件包括：

– 计数器上溢/下溢

– 将 UG 位置 1

– 通过从模式控制器生成的更新事件

1：使能时，只有计数器上溢/下溢会生成更新中断或 DMA 请求。

位 1 **UDIS**：更新禁止 (Update disable)

该位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新 (UEV) 事件可通过以下事件之一生成：

– 计数器上溢/下溢

– 将 UG 位置 1

– 通过从模式控制器生成的更新事件

带缓冲的寄存器被加载为预加载数值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

位 0 **CEN**：计数器使能 (Counter enable)

0：禁止计数器

1：使能计数器

注意：只有事先通过软件将 CEN 位置 1，才可以使用外部时钟、门控模式和编码器模式。而触发模式可通过硬件自动将 CEN 位置 1。

26.6.2 TIM1 控制寄存器 2 (TIM1_CR2)

TIM1 control register 2

偏移地址：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	MMS[3]	Res.	MMS2[3:0]				Res.	OIS6	Res.	OIS5
						rw		rw	rw	rw	rw		rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OIS4N	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Res.	CCPC
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw

位 31:26 保留，必须保持复位值。

位 24 保留，必须保持复位值。

位 23:20 **MMS2[3:0]**：主模式选择 2 (Master mode selection 2)

这些位可选择将发送到 ADC 以实现同步的信息 (tim_trgo2)。这些位的组合如下：

0000： **复位**——TIMx_EGR 寄存器中的 UG 位用作触发输出 (tim_trgo2)。如果复位由触发输入生成（从模式控制器配置为复位模式），则 tim_trgo2 上的信号相比实际复位会有延迟。

0001： **使能**——计数器使能信号 CNT_EN 用作触发输出 (tim_trgo2)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号由 CEN 控制位与门控模式下的触发输入的逻辑与运算组合而成。当计数器使能信号由触发输入控制时，tim_trgo2 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中 MSM 位的说明）。

0010： **更新**——选择更新事件作为触发输出 (tim_trgo2)。例如，主定时器可用作从定时器的预分频器。

0011： **比较脉冲**——CC1IF 标志置 1 时（即使已为高），只要发生捕获或比较匹配，触发输出 (tim_trgo2) 都会发送一个正脉冲。

0100： **比较**——tim_oc1refc 信号用作触发输出 (tim_trgo2)

0101： **比较**——tim_oc2refc 信号用作触发输出 (tim_trgo2)

0110： **比较**——tim_oc3refc 信号用作触发输出 (tim_trgo2)

0111： **比较**——tim_oc4refc 信号用作触发输出 (tim_trgo2)

1000： **比较**——tim_oc5refc 信号用作触发输出 (tim_trgo2)

1001： **比较**——tim_oc6refc 信号用作触发输出 (tim_trgo2)

1010： **比较脉冲**——tim_oc4refc 上升沿或下降沿时，tim_trgo2 上生成脉冲

1011： **比较脉冲**——tim_oc6refc 上升沿或下降沿时，tim_trgo2 上生成脉冲

1100： **比较脉冲**——tim_oc4refc 或 tim_oc6refc 上升沿时，tim_trgo2 上生成脉冲

1101： **比较脉冲**——tim_oc4refc 上升沿或 tim_oc6refc 下降沿时，tim_trgo2 上生成脉冲

1110： **比较脉冲**——tim_oc5refc 或 tim_oc6refc 上升沿时，tim_trgo2 上生成脉冲

1111： **比较脉冲**——tim_oc5refc 上升沿或 tim_oc6refc 下降沿时，tim_trgo2 上生成脉冲

注意：必须先使能从定时器或 ADC 的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改从定时器或 ADC 的时钟。

位 19 保留，必须保持复位值。

位 18 **OIS6**：输出空闲状态 6（tim_oc6 输出）(Output idle state 6 (tim_oc6 output))
请参见 OIS1 位

位 17 保留，必须保持复位值。

位 16 **OIS5**：输出空闲状态 5（tim_oc5 输出）(Output idle state 5 (tim_oc5 output))
请参见 OIS1 位

位 15 **OIS4N**：输出空闲状态 4（tim_oc4n 输出）(Output idle state 4 (tim_oc4n output))
请参见 OIS1N 位

位 14 **OIS4**：输出空闲状态 4（tim_oc4 输出）(Output idle state 4 (tim_oc4 output))
请参见 OIS1 位

位 13 **OIS3N**：输出空闲状态 3（tim_oc3n 输出）(Output idle state 3 (tim_oc3n output))
请参见 OIS1N 位

位 12 **OIS3**：输出空闲状态 3（tim_oc3n 输出）(Output idle state 3 (tim_oc3n output))
请参见 OIS1 位

位 11 **OIS2N**：输出空闲状态 2（tim_oc2n 输出）(Output idle state 2 (tim_oc2n output))
请参见 OIS1N 位

位 10 **OIS2**：输出空闲状态 2（tim_oc2 输出）(Output idle state 2 (tim_oc2 output))
请参见 OIS1 位

位 9 **OIS1N**：输出空闲状态 1（tim_oc1n 输出）(Output idle state 1 (tim_oc1n output))

0：当 MOE=0 时，经过死区时间后 tim_oc1n=0

1：当 MOE=0 时，经过死区时间后 tim_oc1n=1

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，该位即无法修改。

位 8 **OIS1**：输出空闲状态 1（tim_oc1 输出）(Output idle state 1 (tim_oc1 output))

0：当 MOE=0 时，经过死区时间后 tim_oc1=0

1：当 MOE=0 时，经过死区时间后 tim_oc1=1

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，该位即无法修改。

位 7 **TI1S**：tim_ti1 选择 (tim_ti1 selection)

0：tim_ti1_in[15:0] 复用器输出连接到 tim_ti1 输入

1：tim_ti1_in[15:0]、tim_ti2_in[15:0] 和 tim_ti3_in[15:0] 复用器输出进行异或运算，并连接到 tim_ti1 输入

位 25，6:4 **MMS[3:0]**：主模式选择 (Master mode selection)

这些位用于选择主模式下将要发送到从定时器以实现同步的信息 (tim_trgo)。这些位的组合如下：

0000：**复位**——TIMx_EGR 寄存器中的 UG 位用作触发输出 (tim_trgo)。如果复位由触发输入生成（从模式控制器配置为复位模式），则 tim_trgo 上的信号相比实际复位会有延迟。

0001：**使能**——计数器使能信号 CNT_EN 用作触发输出 (tim_trgo)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号由 CEN 控制位与门控模式下的触发输入的逻辑与运算组合而成。当计数器使能信号由触发输入控制时，tim_trgo 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中 MSM 位的说明）。

0010：**更新**——选择更新事件作为触发输出 (tim_trgo)。例如，主定时器可用作从定时器的预分频器。

0011：**比较脉冲**——CC1IF 标志置 1 时（即使已为高电平），只要发生捕获或比较匹配，触发输出都会发送一个正脉冲。(tim_trgo)。

0100：**比较**——tim_oc1refc 信号用作触发输出 (tim_trgo)

0101：**比较**——tim_oc2refc 信号用作触发输出 (tim_trgo)

0110：**比较**——tim_oc3refc 信号用作触发输出 (tim_trgo)

0111：**比较**——tim_oc4refc 信号用作触发输出 (tim_trgo)

1000：**编码器时钟输出**——编码器时钟信号用作触发输出 (tim_trgo)。该代码适用于以下 SMS[3:0] 值：0001、0010、0011、1010、1011、1100、1101、1110、1111。不允许使用任何其他 SMS[3:0] 代码，否则可能会导致意外行为。

其他代码保留

注意：必须先使能从定时器或 ADC 的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改从定时器或 ADC 的时钟。

位 3 **CCDS**：捕获/比较 DMA 选择 (Capture/compare DMA selection)

0：发生 CCx 事件时发送 CCx DMA 请求

1：发生更新事件时发送 CCx DMA 请求

位 2 **CCUS**：捕获/比较控制更新选择 (Capture/compare control update selection)

0：如果捕获/比较控制位 (CCPC=1) 进行预装载，仅通过将 COMG 位置 1 来对这些位进行更新

1：如果捕获/比较控制位 (CCPC=1) 进行预装载，可通过将 COMG 位置 1 或 tim_trgi 的上升沿对这些位进行更新

注意：该位仅对具有互补输出的通道有效。

位 1 保留，必须保持复位值。

位 0 **CCPC**：捕获/比较预装载控制 (Capture/compare preloaded control)

0：CCxE、CCxNE 和 OCxM 位未进行预装载

1：CCxE、CCxNE 和 OCxM 位进行了预装载，写入这些位后，仅当发生换向事件 (COM)（COMG 位置 1 或在 tim_trgi 上检测到上升沿，取决于 CCUS 位）时才会对这些位进行更新。

注意：该位仅对具有互补输出的通道有效。

26.6.3 TIM1 从模式控制寄存器 (TIM1_SMCR)

TIM1 slave mode control register

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	SMSPS	SMSPE	Res.	Res.	TS[4:3]		Res.	Res.	Res.	SMS[3]
						rw	rw			rw	rw				rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:26 保留，必须保持复位值。

位 25 **SMSPS**：SMS 预装载源 (SMS preload source)

该位用于选择将 SMS[3:0] 位域预装载值传输到有效值的触发事件

0：传输由定时器的更新事件触发

1：传输由索引事件触发

位 24 **SMSPE**：SMS 预装载使能 (SMS preload enable)

该位用于选择是否预装载 SMS[3:0] 位域

0：禁止 SMS[3:0] 位域预装载

1：使能 SMS[3:0] 位域预装载

位 23:22 保留，必须保持复位值。

位 21:20 **TS[4:3]**：触发选择——位 4:3 (Trigger selection - bit 4:3)

请参见 TS[2:0] 说明——位 6:4

位 19:17 保留，必须保持复位值。

位 15 **ETP**：外部触发极性 (External trigger polarity)

该位可选择将 tim_etr_in 还是 $\overline{\text{tim_etr_in}}$ 用于触发操作

0：tim_etr_in 未反相，高电平或上升沿有效。

1：tim_etr_in 反相，低电平或下降沿有效。

位 14 **ECE**：外部时钟使能 (External clock enable)

该位可使能外部时钟模式 2。

0：禁止外部时钟模式 2

1：使能外部时钟模式 2。计数器时钟由 tim_etrf 信号的任意有效边沿提供。

注意：将 ECE 位置 1 与选择外部时钟模式 1 并将 tim_trgi 连接到 tim_etrf（SMS=111 且 TS=00111）具有相同效果。

外部时钟模式 2 可以和以下从模式同时使用：复位模式、门控模式和触发模式。不过此类情况下 tim_trgi 不得连接 tim_etrf（TS 位不得为 00111）。

如果同时使能外部时钟模式 1 和外部时钟模式 2，则外部时钟输入为 tim_etrf。

位 13:12 **ETPS[1:0]**：外部触发预分频器 (External trigger prescaler)

外部触发信号 tim_etrp 频率不得超过 TIMxCLK 频率的 1/4。可通过使能预分频器来降低 tim_etrp 频率。这种方法在通过 tim_etr_in 输入快速外部时钟时非常有用。

00：预分频器关闭
01：2 分频 tim_etr_in 频率
10：4 分频 tim_etr_in 频率
11：8 分频 tim_etr_in 频率

位 11:8 **ETF[3:0]**：外部触发滤波器 (External trigger filter)

该位域可定义 tim_etrp 信号的采样频率和适用于 tim_etrp 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿：

0000：无滤波器，按 f_{DTS} 频率进行采样
0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2
0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4
0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8
0100：$f_{SAMPLING}=f_{DTS}/2$，N=6
0101：$f_{SAMPLING}=f_{DTS}/2$，N=8
0110：$f_{SAMPLING}=f_{DTS}/4$，N=6
0111：$f_{SAMPLING}=f_{DTS}/4$，N=8
1000：$f_{SAMPLING}=f_{DTS}/8$，N=6
1001：$f_{SAMPLING}=f_{DTS}/8$，N=8
1010：$f_{SAMPLING}=f_{DTS}/16$，N=5
1011：$f_{SAMPLING}=f_{DTS}/16$，N=6
1100：$f_{SAMPLING}=f_{DTS}/16$，N=8
1101：$f_{SAMPLING}=f_{DTS}/32$，N=5
1110：$f_{SAMPLING}=f_{DTS}/32$，N=6
1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 7 **MSM**：主/从模式 (Master/slave mode)

0：无操作
1：当前定时器的触发输入事件 (tim_trgi) 的动作被推迟，以使当前定时器与其从定时器实现完美同步（通过 tim_trgo）。此设置适用于由单个外部事件对多个定时器进行同步的情况。

位 6:4 **TS[2:0]**：触发选择 (Trigger selection)

此位域与 TS[4:3] 位组合在一起。

该位域可选择将要用于同步计数器的触发输入。

00000：内部触发 0 (tim_itr0)
00001：内部触发 1 (tim_itr1)
00010：内部触发 2 (tim_itr2)
00011：内部触发 3 (tim_itr3)
00100：tim_ti1 边沿检测信号 (tim_ti1f_ed)
00101：滤波后的定时器输入 1 (tim_ti1fp1)
00110：滤波后的定时器输入 2 (tim_ti2fp2)
00111：外部触发输入 (tim_etrf)
01000：内部触发 4 (tim_itr4)
01001：内部触发 5 (tim_itr5)
01010：内部触发 6 (tim_itr6)
01011：内部触发 7 (tim_itr7)
01100：内部触发 8 (tim_itr8)
01101：内部触发 9 (tim_itr9)
01110：内部触发 10 (tim_itr10)
01111：内部触发 11 (tim_itr11)
10000：内部触发 12 (tim_itr12)
10001：内部触发 13 (tim_itr13)
10010：内部触发 14 (tim_itr14)
10011：内部触发 15 (tim_itr15)
其他值：保留

有关各定时器 tim_itrx 含义的详细信息，请参见*表 168：内部触发连接*。

注意：这些位只能在未使用的情况下（例如，SMS = 000 时）进行更改，以避免转换时出现错误的边沿检测。

位 3 **OCCS**：OCREF 清零选择 (OCREF clear selection)

该位用于选择 OCREF 清零源。

0：tim_ocref_clr_int 连接到 tim_ocref_clr 输入
1：tim_ocref_clr_int 连接到 tim_etrf

位 16，2:0 **SMS[3:0]**：从模式选择 (Slave mode selection)

选择外部信号时，触发信号 (tim_trgi) 的有效边沿与外部输入上所选的极性相关（请参见输入控制寄存器和控制寄存器说明）。

0000： 禁止从模式——如果 CEN = “1”，预分频器时钟直接由内部时钟提供。

0001： 正交编码器模式 1（x2 模式）——计数器根据 tim_ti2fp2 电平在 tim_ti1fp1 边沿递增/递减计数。

0010： 正交编码器模式 2（x2 模式）——计数器根据 tim_ti1fp1 电平在 tim_ti2fp2 边沿递增/递减计数。

0011： 正交编码器模式 3（x4 模式）——计数器在 tim_ti1fp1 和 tim_ti2fp2 的边沿计数，计数的方向取决于另外一个输入的电平。

0100： 复位模式——在出现所选触发输入 (tim_trgi) 上升沿时，重新初始化计数器并生成一个寄存器更新事件。

0101： 门控模式——触发输入 (tim_trgi) 为高电平时使能计数器时钟。只要触发输入变为低电平，计数器立即停止计数（但不复位）。计数器的启动和停止都被控制。

0110： 触发模式——触发信号 tim_trgi 出现上升沿时启动计数器（但不复位）。只控制计数器的启动。

0111： 外部时钟模式 1——由所选触发信号 (tim_trgi) 的上升沿提供计数器时钟。

1000： 复位 + 触发组合模式——在出现所选触发输入 (tim_trgi) 上升沿时，重新初始化计数器，生成一个寄存器更新事件并启动计数器。

1001： 门控 + 复位组合模式——触发输入 (tim_trgi) 为高电平时使能计数器时钟。只要触发输入变为低电平，计数器立即停止计数并复位。计数器的启动和停止都被控制。

1010： 编码器模式：时钟 + 方向，x2 模式。

1011： 编码器模式：时钟 + 方向，x1 模式，tim_ti2fp2 边沿有效性由 CC2P 设置

1100： 编码器模式：定向时钟，x2 模式。

1101： 编码器模式：定向时钟，x1 模式，tim_ti1fp1 和 tim_ti2fp2 的边沿有效性由 CC1P 和 CC2P 设置。

1110： 正交编码器模式：x1 模式，仅在 tim_ti1fp1 边沿处计数，边沿有效性由 CC1P 设置。

1111： 正交编码器模式：x1 模式，仅在 tim_ti2fp2 边沿处计数，边沿有效性由 CC2P 设置。

注意：如果将 tim_ti1f_ed 选作触发输入 (TS=00100)，则不得使用门控模式。实际上，TI1F 每次转换时，tim_ti1f_ed 都输出 1 个脉冲，而门控模式检查的则是触发信号的电平。

注意：必须先使能接收 tim_trgo 或 tim_trgo2 信号的从外设（定时器、ADC 等）的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改时钟频率（预分频器）。

26.6.4 TIM1 DMA/中断使能寄存器 (TIM1_DIER)

TIM1 DMA/interrupt enable register

偏移地址：0x00C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TERRIE	IERRIE	DIRIE	IDXIE	Res.	Res.	Res.	Res.
								rw	rw	rw	rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23 **TERRIE**：转换错误中断使能 (Transition error interrupt enable)
0：禁止转换错误中断
1：使能转换错误中断

位 22 **IERRIE**：索引错误中断使能 (Index error interrupt enable)
0：禁止索引错误中断
1：使能索引错误中断

位 21 **DIRIE**：方向更改中断使能 (Direction change interrupt enable)
0：禁止方向更改中断
1：使能方向更改中断

位 20 **IDXIE**：索引中断使能 (Index interrupt enable)
0：禁止索引中断
1：使能索引中断

位 19:15 保留，必须保持复位值。

位 14 **TDE**：触发信号 (TGRI) DMA 请求使能 (Trigger DMA request enable)
0：禁止触发信号 (TGRI) DMA 请求
1：使能触发信号 (TGRI) DMA 请求

位 13 **COMDE**：COM DMA 请求使能 (COM DMA request enable)
0：禁止 COM DMA 请求
1：使能 COM DMA 请求

位 12 **CC4DE**：捕获/比较 4 DMA 请求使能 (Capture/compare 4 DMA request enable)
0：禁止 CC4 DMA 请求
1：使能 CC4 DMA 请求

位 11 **CC3DE**：捕获/比较 3 DMA 请求使能 (Capture/compare 3 DMA request enable)
0：禁止 CC3 DMA 请求
1：使能 CC3 DMA 请求

位 10 **CC2DE**：捕获/比较 2 DMA 请求使能 (Capture/compare 2 DMA request enable)
0：禁止 CC2 DMA 请求
1：使能 CC2 DMA 请求

位 9 **CC1DE**：捕获/比较 1 DMA 请求使能 (Capture/compare 1 DMA request enable)
0：禁止 CC1 DMA 请求
1：使能 CC1 DMA 请求

位 8 **UDE**：更新 DMA 请求使能 (Update DMA request enable)
0：禁止更新 DMA 请求
1：使能更新 DMA 请求

位 7 **BIE**：刹车中断使能 (Break interrupt enable)
0：禁止刹车中断
1：使能刹车中断

位 6 **TIE**：触发信号 (TGRI) 中断使能 (Trigger interrupt enable)
0：禁止触发信号 (TGRI) 中断
1：使能触发信号 (TGRI) 中断

位 5 **COMIE**：COM 中断使能 (COM interrupt enable)
0：禁止 COM 中断
1：使能 COM 中断

位 4 **CC4IE**：捕获/比较 4 中断使能 (Capture/compare 4 interrupt enable)
0：禁止 CC4 中断
1：使能 CC4 中断

位 3 **CC3IE**：捕获/比较 3 中断使能 (Capture/compare 3 interrupt enable)
0：禁止 CC3 中断
1：使能 CC3 中断

位 2 **CC2IE**：捕获/比较 2 中断使能 (Capture/compare 2 interrupt enable)
0：禁止 CC2 中断
1：使能 CC2 中断

位 1 **CC1IE**：捕获/比较 1 中断使能 (Capture/compare 1 interrupt enable)
0：禁止 CC1 中断
1：使能 CC1 中断

位 0 **UIE**：更新中断使能 (Update interrupt enable)
0：禁止更新中断
1：使能更新中断

26.6.5 TIM1 状态寄存器 (TIM1_SR)

TIM1 status register

偏移地址：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TERRF	IERRF	DIRF	IDXF	Res.	Res.	CC6IF	CC5IF
								rc_w0	rc_w0	rc_w0	rc_w0			rc_w0	rc_w0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	SBIF	CC4OF	CC3OF	CC2OF	CC1OF	B2IF	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

位 31:24 保留，必须保持复位值。

位 23 **TERRF**：转换错误中断标志 (Transition error interrupt flag)
在编码器模式下检测到转换错误时，该标志由硬件置 1。通过软件写入“0”可将该位清零。
0：未检测到编码器转换错误
1：检测到编码器转换错误

位 22 **IERRF**：索引错误中断标志 (Index error interrupt flag)
检测到索引错误时，该标志由硬件置 1。通过软件写入“0”可将该位清零。
0：未检测到索引错误
1：检测到索引错误

位 21 **DIRF**：方向更改中断标志 (Direction change interrupt flag)
在编码器模式下发生方向更改时（TIMx_CR 中 DIR 位的值发生变化），该标志由硬件置 1。通过软件写入“0”可将该位清零。
0：方向未更改
1：方向更改

位 20 **IDXF**：索引中断标志 (Index interrupt flag)
检测到索引事件时，该标志由硬件置 1。通过软件写入“0”可将该位清零。
0：未发生索引事件
1：发生了索引事件

位 19:18 保留，必须保持复位值。

位 17 **CC6IF**：比较 6 中断标志 (Compare 6 interrupt flag)
请参见 CC1IF 说明
注意：通道 6 只能配置为输出。

位 16 **CC5IF**：比较 5 中断标志 (Compare 5 interrupt flag)
请参见 CC1IF 说明
注意：通道 5 只能配置为输出。

位 15:14 保留，必须保持复位值。

位 13 **SBIF**：系统刹车中断标志 (System break interrupt flag)
只要系统刹车输入变为有效状态，该标志便由硬件置 1。系统刹车输入无效后可通过软件对其清零。
此标志必须复位以使 PWM 重新开始工作。
0：未发生刹车事件。
1：在系统刹车输入上检测到有效电平。如果 TIMx_DIER 寄存器中 BIE=1，则会生成中断。

位 12 **CC4OF**：捕获/比较 4 过捕获标志 (Capture/compare 4 overcapture flag)
请参见 CC1OF 说明

位 11 **CC3OF**：捕获/比较 3 过捕获标志 (Capture/compare 3 overcapture flag)
请参见 CC1OF 说明

位 10 **CC2OF**：捕获/比较 2 过捕获标志 (Capture/compare 2 overcapture flag)
请参见 CC1OF 说明

位 9 **CC1OF**：捕获/比较 1 过捕获标志 (Capture/compare 1 overcapture flag)
仅当将相应通道配置为输入捕获模式时，此标志位才会由硬件置 1。通过软件写入“0”可将该位清零。
0：未检测到过捕获。
1：TIMx_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。

位 8 **B2IF**：刹车 2 中断标志 (Break 2 interrupt flag)
只要刹车 2 输入变为有效状态，此标志便由硬件置 1。刹车 2 输入无效后可通过软件对其清零。
0：未发生刹车事件。
1：在刹车 2 输入上检测到有效电平。如果 TIMx_DIER 寄存器中 BIE=1，则会生成中断。

位 7 **BIF**：刹车中断标志 (Break interrupt flag)
只要刹车输入变为有效状态，此标志便由硬件置 1。刹车输入无效后可通过软件对其清零。
0：未发生刹车事件。
1：在刹车输入上检测到有效电平。如果 TIMx_DIER 寄存器中 BIE=1，则会生成中断。

位 6 **TIF**：触发中断标志 (Trigger interrupt flag)
在除门控模式以外的所有模式下，当使能从模式控制器后在 tim_trgi 输入上检测到 TRG 触发事件有效边沿时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动或停止时置 1。但需要通过软件清零。
0：未发生触发事件。
1：触发中断挂起。

位 5 **COMIF**：COM 中断标志 (COM interrupt flag)

该标志在发生 COM 事件时（捕获/比较控制位 CCxE、CCxNE 和 OCxM 已更新时）由硬件置 1。但需要通过软件清零。

0：未发生 COM 事件。

1：COM 中断挂起。

位 4 **CC4IF**：捕获/比较 4 中断标志 (Capture/compare 4 interrupt flag)

请参见 CC1IF 说明

位 3 **CC3IF**：捕获/比较 3 中断标志 (Capture/compare 3 interrupt flag)

请参见 CC1IF 说明

位 2 **CC2IF**：捕获/比较 2 中断标志 (Capture/compare 2 interrupt flag)

请参见 CC1IF 说明

位 1 **CC1IF**：捕获/比较 1 中断标志 (Capture/compare 1 interrupt flag)

该标志由硬件置 1，但需要通过软件清零（输入捕获或输出比较模式）或通过读取 TIMx_CCR1 寄存器清零（仅限输入捕获模式）。

0：未发生比较匹配/输入捕获

1：发生了比较匹配/输入捕获

如果通道 CC1 配置为输出：当计数器 TIMx_CNT 的值与 TIMx_CCR1 的值匹配时，此标志置 1。当 TIMx_CCR1 的值大于 TIMx_ARR 的值时，CC1IF 位将在计数器发生上溢（递增计数模式和增减计数模式下）或下溢（递减计数模式下）时变为高电平。在中心对齐模式下，可通过 3 种方式将此标志置 1，完整说明请参见 TIMx_CR1 寄存器中的 CMS 位。

如果通道 CC1 配置为输入：当 TIMx_CCR1 寄存器中捕获到计数器值时（如果通过设置 TIMx_CCER 中的 CC1P 和 CC1NP 位定义为边沿有效，则改为在 IC1 上检测到边沿时），该位置 1。

位 0 **UIF**：更新中断标志 (Update interrupt flag)

该位在发生更新事件时通过硬件置 1。但需要通过软件清零。

0：未发生更新。

1：更新中断挂起。该位在以下情况下更新寄存器时由硬件置 1：

– TIMx_CR1 寄存器中的 UDIS=0，并且重复计数器值上溢或下溢时（重复计数器 = 0 时更新）。

– TIMx_CR1 寄存器中的 URS = 0 且 UDIS = 0，并且由软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。

– TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且 CNT 由触发事件重新初始化时（请参见 *第 26.6.3 节：TIM1 从模式控制寄存器 (TIM1_SMCR)*）。

26.6.6 TIM1 事件生成寄存器 (TIM1_EGR)

TIM1 event generation register

偏移地址：0x014

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	B2G	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
							w	w	w	w	w	w	w	w	w

位 15:9 保留，必须保持复位值。

位 8 **B2G**：刹车 2 生成 (Break 2 generation)
该位由软件置 1 以生成事件，并由硬件自动清零。
0：无操作
1：生成刹车 2 事件。MOE 位清零且 B2IF 标志置 1。使能后可发生相关中断。

位 7 **BG**：刹车生成 (Break generation)
该位由软件置 1 以生成事件，并由硬件自动清零。
0：无操作
1：生成刹车事件。MOE 位清零且 BIF 标志置 1。使能后可发生相关中断或 DMA 传输事件。

位 6 **TG**：触发生成 (Trigger generation)
该位由软件置 1 以生成事件，并由硬件自动清零。
0：无操作
1：TIMx_SR 寄存器中的 TIF 标志置 1。使能后可发生相关中断或 DMA 传输事件。

位 5 **COMG**：捕获/比较控制更新生成 (Capture/compare control update generation)
该位可通过软件置 1，并由硬件自动清零
0：无操作
1：CCxE、CCxNE 和 OCxM 位更新（前提是 CCPC 位置 1）
注意：该位仅对具有互补输出的通道有效。

位 4 **CC4G**：捕获/比较 4 生成 (Capture/compare 4 generation)
请参见 CC1G 说明

位 3 **CC3G**：捕获/比较 3 生成 (Capture/compare 3 generation)
请参见 CC1G 说明

位 2 **CC2G**：捕获/比较 2 生成 (Capture/compare 2 generation)
请参见 CC1G 说明

位 1 **CC1G**：捕获/比较 1 生成 (Capture/compare 1 generation)
该位由软件置 1 以生成事件，并由硬件自动清零。
0：无操作
1：通道 1 上生成捕获/比较事件：
如果通道 CC1 配置为输出：
使能时，CC1IF 标志置 1 并发送相应的中断或 DMA 请求。
如果通道 CC1 配置为输入：
TIMx_CCR1 寄存器中将捕获到计数器当前值。使能时，CC1IF 标志置 1 并发送相应的中断或 DMA 请求。如果 CC1IF 标志已为高电平，CC1OF 标志将置 1。

位 0 **UG**：更新生成 (Update generation)
该位可通过软件置 1，并由硬件自动清零。
0：无操作
1：重新初始化计数器并生成一个寄存器更新事件。请注意，预分频器计数器也将清零（但预分频比不受影响）。如果选择中心对齐模式或 DIR=0（递增计数），计数器将清零；如果 DIR=1（递减计数），计数器将使用自动重载值 (TIMx_ARR)。

26.6.7 TIM1 捕获/比较模式寄存器 1 (TIM1_CCMR1)

TIM1 capture/compare mode register 1

偏移地址：0x018

复位值：0x0000 0000

同一寄存器可用于输入捕获模式（本节）或输出比较模式（下一节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。可独立组合两种模式（例如，输入捕获模式下的通道 1 和输出比较模式下的通道 2）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输入捕获模式：

位 31:16 保留，必须保持复位值。

位 15:12 **IC2F[3:0]**：输入捕获 2 滤波器 (Input capture 2 filter)

位 11:10 **IC2PSC[1:0]**：输入捕获 2 预分频器 (Input capture 2 prescaler)

位 9:8 **CC2S[1:0]**：捕获/比较 2 选择 (Capture/compare 2 selection)

该位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道配置为输出

01：CC2 通道配置为输入，tim_ic2 映射到 tim_ti2 上

10：CC2 通道配置为输入，tim_ic2 映射到 tim_ti1 上

11：CC2 通道配置为输入，tim_ic2 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注意：仅当通道关闭时（TIMx_CCER 中的 CC2E = “0”），才可向 CC2S 位写入数据。

位 7:4 **IC1F[3:0]**：输入捕获 1 滤波器 (Input capture 1 filter)

该位域可定义 tim_ti1 输入的采样频率和适用于 tim_ti1 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿：

0000：无滤波器，按 f_{DTS} 频率进行采样

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8

1000：$f_{SAMPLING}=f_{DTS}/8$，N=6

1001：$f_{SAMPLING}=f_{DTS}/8$，N=8

1010：$f_{SAMPLING}=f_{DTS}/16$，N=5

1011：$f_{SAMPLING}=f_{DTS}/16$，N=6

1100：$f_{SAMPLING}=f_{DTS}/16$，N=8

1101：$f_{SAMPLING}=f_{DTS}/32$，N=5

1110：$f_{SAMPLING}=f_{DTS}/32$，N=6

1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 3:2 **IC1PSC[1:0]**：输入捕获 1 预分频器 (Input capture 1 prescaler)

该位域定义 CC1 输入 (tim_ic1) 的预分频比。只要 CC1E=“0”（TIMx_CCER 寄存器），预分频器便立即复位。

00：无预分频器，捕获输入上每检测到一个边沿便执行捕获

01：每发生 2 个事件便执行一次捕获

10：每发生 4 个事件便执行一次捕获

11：每发生 8 个事件便执行一次捕获

位 1:0 **CC1S[1:0]**：捕获/比较 1 选择 (Capture/compare 1 Selection)

该位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出

01：CC1 通道配置为输入，tim_ic1 映射到 tim_ti1 上

10：CC1 通道配置为输入，tim_ic1 映射到 tim_ti2 上

11：CC1 通道配置为输入，tim_ic1 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注意：仅当通道关闭时（TIMx_CCER 中的 CC1E = “0”），才可向 CC1S 位写入数据。

26.6.8 TIM1 捕获/比较模式寄存器 1 [复用] (TIM1_CCMR1)

TIM1 capture/compare mode register 1

偏移地址：0x018

复位值：0x0000 0000

同一寄存器可用于输出比较模式（本节）或输入捕获模式（上一节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。可独立组合两种模式（例如，输入捕获模式下的通道 1 和输出比较模式下的通道 2）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2 CE	OC2M[2:0]			OC2 PE	OC2 FE	CC2S[1:0]		OC1 CE	OC1M[2:0]			OC1 PE	OC1 FE	CC1S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式：

位 31:25 保留，必须保持复位值。

位 23:17 保留，必须保持复位值。

位 15 **OC2CE**：输出比较 2 清零使能 (Output compare 2 clear enable)

位 24、14:12 **OC2M[3:0]**：输出比较 2 模式 (Output compare 2 mode)

位 11 **OC2PE**：输出比较 2 预装载使能 (Output compare 2 preload enable)

位 10 **OC2FE**：输出比较 2 快速使能 (Output compare 2 fast enable)

位 9:8 **CC2S[1:0]**：捕获/比较 2 选择 (Capture/compare 2 selection)

该位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道配置为输出

01：CC2 通道配置为输入，tim_ic2 映射到 tim_ti2 上

10：CC2 通道配置为输入，tim_ic2 映射到 tim_ti1 上

11：CC2 通道配置为输入，tim_ic2 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注意：仅当通道关闭时（TIMx_CCER 中的 CC2E = “0”），才可向 CC2S 位写入数据。

位 7 **OC1CE**：输出比较 1 清零使能 (Output compare 1 clear enable)

0：tim_oc1ref 不受 tim_ocref_clr_int 信号影响

1：tim_ocref_clr_int 信号（tim_ocref_clr 输入或 tim_etrf 输入）上检测到高电平时，tim_oc1ref 立即清零

位 16，6:4 **OC1M[3:0]**：输出比较 1 模式 (Output compare 1 mode)

这些位定义提供 tim_oc1 和 tim_oc1n 的输出参考信号 tim_oc1ref 的行为。tim_oc1ref 为高电平有效，而 tim_oc1 和 tim_oc1n 的有效电平则取决于 CC1P 和 CC1NP 位。

0000：冻结——输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 进行比较不会对输出造成任何影响。当定时器用作软件时基时，可以使用该模式。在定时器操作期间使能冻结模式时，输出保持进入冻结状态前的状态（有效或无效）。

0001：将通道 1 设置为匹配时输出有效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1 (TIMx_CCR1) 匹配时，tim_oc1ref 信号强制变为高电平。

0010：将通道 1 设置为匹配时输出无效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1 (TIMx_CCR1) 匹配时，tim_oc1ref 信号强制变为低电平。

0011：翻转——TIMx_CNT=TIMx_CCR1 时，tim_oc1ref 发生翻转。

0100：强制变为无效电平——tim_oc1ref 强制变为低电平。

0101：强制变为有效电平——tim_oc1ref 强制变为高电平。

0110：PWM 模式 1——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便输出有效电平，否则输出无效电平。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便输出无效电平（tim_oc1ref=“0”），否则输出有效电平（tim_oc1ref=“1”）。

0111：PWM 模式 2——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便输出无效电平，否则输出有效电平。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便输出有效电平，否则输出无效电平。

1000：可再触发 OPM 模式 1——在递增计数模式下，通道为有效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为有效状态。在递减计数模式下，通道为无效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为无效状态。

1001：可再触发 OPM 模式 2——在递增计数模式下，通道为无效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 2 下进行比较，通道会在下一次更新时再次变为无效状态。在递减计数模式下，通道为有效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为有效状态。

1010：保留。

1011：保留。

1100：组合 PWM 模式 1——tim_oc1ref 与在 PWM 模式 1 下的行为相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 的逻辑或运算结果。

1101：组合 PWM 模式 2——tim_oc1ref 与在 PWM 模式 2 下的行为相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 的逻辑与运算结果。

1110：不对称 PWM 模式 1——tim_oc1ref 与在 PWM 模式 1 下的行为相同。计数器递增计数时，tim_oc1refc 输出 tim_oc1ref；计数器递减计数时，tim_oc1refc 输出 tim_oc2ref。

1111：不对称 PWM 模式 2——tim_oc1ref 与在 PWM 模式 2 下的行为相同。计数器递增计数时，tim_oc1refc 输出 tim_oc1ref；计数器递减计数时，tim_oc1refc 输出 tim_oc2ref。

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 3 且 CC1S=“00”（通道配置为输出），这些位即无法修改。

注意：在 PWM 模式下，当比较结果发生改变，输出比较模式从“冻结”模式切换到“PWM”模式以及输出比较模式从“强制有效/无效”模式切换到“PWM”模式时，OCREF 电平会发生更改。

注意：该位域将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成 COM 事件时，OC1M 有效位才会从预装载位获取新值。

位 3 **OC1PE**：输出比较 1 预装载使能 (Output compare 1 preload enable)

0：禁止与 TIMx_CCR1 相关的预装载寄存器。可随时向 TIMx_CCR1 写入数据，写入后将立即使用新值。

1：使能与 TIMx_CCR1 相关的预装载寄存器。可读/写访问预装载寄存器。TIMx_CCR1 预装载值在每次生成更新事件时都会装载到影子寄存器中。

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 3 且 CC1S="00"（通道配置为输出），这些位即无法修改。

位 2 **OC1FE**：输出比较 1 快速使能 (Output compare 1 fast enable)

该位可降低触发事件与定时器输出跳变之间的延迟。单脉冲模式下必须使用（TIMx_CR1 寄存器中的 OPM 位置 1），以便在启动触发后快速发送输出脉冲。

0：即使触发开启，CC1 也将根据计数器和 CCR1 值正常工作。触发输入出现边沿时，激活 CC1 输出的最短延迟时间为 5 个时钟周期。

1：触发输入上出现有效边沿相当于 CC1 输出上的比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和激活 CC1 输出的延迟时间缩短为 3 个时钟周期。仅当通道配置为 PWM1 或 PWM2 模式时，OCFE 才会起作用。

位 1:0 **CC1S[1:0]**：捕获/比较 1 选择 (Capture/compare 1 selection)

该位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出

01：CC1 通道配置为输入，tim_ic1 映射到 tim_ti1 上

10：CC1 通道配置为输入，tim_ic1 映射到 tim_ti2 上

11：CC1 通道配置为输入，tim_ic1 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注意：仅当通道关闭时（TIMx_CCER 中的 CC1E = "0"），才可向 CC1S 位写入数据。

26.6.9 TIM1 捕获/比较模式寄存器 2 (TIM1_CCMR2)

TIM1 capture/compare mode register 2

偏移地址：0x01C

复位值：0x0000 0000

同一寄存器可用于输入捕获模式（本节）或输出比较模式（下一节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。可独立组合两种模式（例如，输入捕获模式下的通道 3 和输出比较模式下的通道 4）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC4F[3:0]				IC4PSC[1:0]		CC4S[1:0]		IC3F[3:0]				IC3PSC[1:0]		CC3S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输入捕获模式

位 31:16 保留，必须保持复位值。

位 15:12 **IC4F[3:0]**：输入捕获 4 滤波器 (Input capture 4 filter)

位 11:10 **IC4PSC[1:0]**：输入捕获 4 预分频器 (Input capture 4 prescaler)

位 9:8 **CC4S[1:0]**：捕获/比较 4 选择 (Capture/compare 4 selection)
该位域定义通道方向（输入/输出）以及所使用的输入。
00：CC4 通道配置为输出
01：CC4 通道配置为输入，tim_ic4 映射到 tim_ti4 上
10：CC4 通道配置为输入，tim_ic4 映射到 tim_ti3 上
11：CC4 通道配置为输入，tim_ic4 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注意：仅当通道关闭时（TIMx_CCER 中的 CC4E = “0”），才可向 CC4S 位写入数据。

位 7:4 **IC3F[3:0]**：输入捕获 3 滤波器 (Input capture 3 filter)

位 3:2 **IC3PSC[1:0]**：输入捕获 3 预分频器 (Input capture 3 prescaler)

位 1:0 **CC3S[1:0]**：捕获/比较 3 选择 (Capture/compare 3 selection)
该位域定义通道方向（输入/输出）以及所使用的输入。
00：CC3 通道配置为输出
01：CC3 通道配置为输入，tim_ic3 映射到 tim_ti3 上
10：CC3 通道配置为输入，tim_ic3 映射到 tim_ti4 上
11：CC3 通道配置为输入，tim_ic3 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注意：仅当通道关闭时（TIMx_CCER 中的 CC3E = “0”），才可向 CC3S 位写入数据。

26.6.10 TIM1 捕获/比较模式寄存器 2 [复用] (TIM1_CCMR2)

TIM1 capture/compare mode register 2

偏移地址：0x01C

复位值：0x0000 0000

同一寄存器可用于输出比较模式（本节）或输入捕获模式（上一节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。可独立组合两种模式（例如，输入捕获模式下的通道 3 和输出比较模式下的通道 4）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC3M[3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4 CE	OC4M[2:0]			OC4 PE	OC4 FE	CC4S[1:0]		OC3 CE	OC3M[2:0]			OC3 PE	OC3 FE	CC3S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

位 31:25 保留，必须保持复位值。

位 23:17 保留，必须保持复位值。

位 15 **OC4CE**：输出比较 4 清零使能 (Output compare 4 clear enable)

位 24、14:12 **OC4M[3:0]**：输出比较 4 模式 (Output compare 4 mode)
请参见 OC3M[3:0] 位的说明

位 11 **OC4PE**：输出比较 4 预装载使能 (Output compare 4 preload enable)

位 10 **OC4FE**：输出比较 4 快速使能 (Output compare 4 fast enable)

位 9:8 **CC4S[1:0]**：捕获/比较 4 选择 (Capture/compare 4 selection)

该位域定义通道方向（输入/输出）以及所使用的输入。

00：CC4 通道配置为输出

01：CC4 通道配置为输入，tim_ic4 映射到 tim_ti4 上

10：CC4 通道配置为输入，tim_ic4 映射到 tim_ti3 上

11：CC4 通道配置为输入，tim_ic4 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注意：仅当通道关闭时（TIMx_CCER 中的 CC4E = “0”），才可向 CC4S 位写入数据。

位 7 **OC3CE**：输出比较 3 清零使能 (Output compare 3 clear enable)

位 16，6:4 **OC3M[3:0]**：输出比较 3 模式 (Output compare 3 mode)

这些位定义提供 tim_oc3 和 tim_oc3n 的输出参考信号 tim_oc3ref 的行为。tim_oc3ref 为高电平有效，而 tim_oc3 和 tim_oc3n 的有效电平则取决于 CC3P 和 CC3NP 位。

0000：冻结——输出比较寄存器 TIMx_CCR3 与计数器 TIMx_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。

0001：将通道 3 设置为匹配时输出有效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 3 (TIMx_CCR3) 匹配时，tim_oc3ref 信号强制变为高电平。

0010：将通道 3 设置为匹配时输出无效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 3 (TIMx_CCR3) 匹配时，tim_oc3ref 信号强制变为低电平。

0011：翻转——TIMx_CNT=TIMx_CCR3 时，tim_oc3ref 发生翻转。

0100：强制变为无效电平——tim_oc3ref 强制变为低电平。

0101：强制变为有效电平——tim_oc3ref 强制变为高电平。

0110：PWM 模式 1——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR3，通道 3 便输出有效电平，否则输出无效电平。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR3，通道 3 便输出无效电平（tim_oc3ref=“0”），否则输出有效电平（tim_oc3ref=“1”）。

0111：PWM 模式 2——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR3，通道 3 便输出无效电平，否则输出有效电平。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR3，通道 3 便输出有效电平，否则输出无效电平。

1000：可再触发 OPM 模式 1——在递增计数模式下，通道为有效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为有效状态。在递减计数模式下，通道为无效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为无效状态。

1001：可再触发 OPM 模式 2——在递增计数模式下，通道为无效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 2 下进行比较，通道会在下一次更新时再次变为无效状态。在递减计数模式下，通道为有效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为有效状态。

1010：比较匹配时生成脉冲：根据 TIMxECR 中的 PWPRSC[2:0] 和 PW[7:0] 位域的编程设定，发生 CCR3 匹配事件时在 tim_oc3ref 上生成脉冲。

1011：方向输出。tim_oc3ref 信号被 DIR 位的副本覆盖。

1100：组合 PWM 模式 1——tim_oc3ref 与在 PWM 模式 1 下的行为相同。tim_oc3refc 是 tim_oc3ref 和 tim_oc4ref 的逻辑或运算结果。

1101：组合 PWM 模式 2——tim_oc3ref 与在 PWM 模式 2 下的行为相同。tim_oc3refc 是 tim_oc3ref 和 tim_oc4ref 的逻辑与运算结果。

1110：不对称 PWM 模式 1——tim_oc3ref 与在 PWM 模式 1 下的行为相同。计数器递增计数时，tim_oc3refc 输出 tim_oc3ref；计数器递减计数时，tim_oc3refc 输出 tim_oc4ref。

1111：不对称 PWM 模式 2——tim_oc3ref 与在 PWM 模式 2 下的行为相同。计数器递增计数时，tim_oc3refc 输出 tim_oc3ref；计数器递减计数时，tim_oc3refc 输出 tim_oc4ref。

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 3 且 CC1S=“00”（通道配置为输出），这些位即无法修改。

注意：在 PWM 模式下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“PWM”模式时，OCREF 电平才会发生更改。

该位域将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成 COM 事件时，OC3M 有效位才会从预装载位获取新值。

位 3 **OC3PE**：输出比较 3 预装载使能 (Output compare 3 preload enable)

位 2 **OC3FE**：输出比较 3 快速使能 (Output compare 3 fast enable)

位 1:0 **CC3S[1:0]**：捕获/比较 3 选择 (Capture/compare 3 selection)

该位域定义通道方向（输入/输出）以及所使用的输入。

00：CC3 通道配置为输出

01：CC3 通道配置为输入，tim_ic3 映射到 tim_ti3 上

10：CC3 通道配置为输入，tim_ic3 映射到 tim_ti4 上

11：CC3 通道配置为输入，tim_ic3 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注意：仅当通道关闭时（TIMx_CCER 中的 CC3E = "0"），才可向 CC3S 位写入数据。

26.6.11 TIM1 捕获/比较使能寄存器 (TIM1_CCER)

TIM1 capture/compare enable register

偏移地址：0x020

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC6P	CC6E	Res.	Res.	CC5P	CC5E
										rw	rw			rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	CC4NE	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:22 保留，必须保持复位值。

位 21 **CC6P**：捕获/比较 6 输出极性 (Capture/compare 6 output polarity)
请参见 CC1P 说明

位 20 **CC6E**：捕获/比较 6 输出使能 (Capture/compare 6 output enable)
请参见 CC1E 说明

位 19:18 保留，必须保持复位值。

位 17 **CC5P**：捕获/比较 5 输出极性 (Capture/compare 5 output polarity)
请参见 CC1P 说明

位 16 **CC5E**：捕获/比较 5 输出使能 (Capture/compare 5 output enable)
请参见 CC1E 说明

位 15 **CC4NP**：捕获/比较 4 互补输出极性 (Capture/compare 4 complementary output polarity)
请参见 CC1NP 说明

位 14 **CC4NE**：捕获/比较 4 互补输出使能 (Capture/compare 4 complementary output enable)
请参见 CC1NE 说明

位 13 **CC4P**：捕获/比较 4 输出极性 (Capture/compare 4 output polarity)
请参见 CC1P 说明

位 12 **CC4E**：捕获/比较 4 输出使能 (Capture/compare 4 output enable)
请参见 CC1E 说明

位 11 **CC3NP**：捕获/比较 3 互补输出极性 (Capture/compare 3 complementary output polarity)
请参见 CC1NP 说明

位 10 **CC3NE**：捕获/比较 3 互补输出使能 (Capture/compare 3 complementary output enable)
请参见 CC1NE 说明

位 9 **CC3P**：捕获/比较 3 输出极性 (Capture/compare 3 output polarity)
请参见 CC1P 说明

位 8 **CC3E**：捕获/比较 3 输出使能 (Capture/compare 3 output enable)
请参见 CC1E 说明

位 7 **CC2NP**：捕获/比较 2 互补输出极性 (Capture/compare 2 complementary output polarity)
请参见 CC1NP 说明

位 6 **CC2NE**：捕获/比较 2 互补输出使能 (Capture/compare 2 complementary output enable)
请参见 CC1NE 说明

位 5 **CC2P**：捕获/比较 2 输出极性 (Capture/compare 2 output polarity)
请参见 CC1P 说明

位 4 **CC2E**：捕获/比较 2 输出使能 (Capture/compare 2 output enable)
请参见 CC1E 说明

位 3 **CC1NP**：捕获/比较 1 互补输出极性 (Capture/compare 1 complementary output polarity)
CC1 通道配置为输出：
0：tim_oc1n 高电平有效。
1：tim_oc1n 低电平有效。
CC1 通道配置为输入：
该位与 CC1P 配合使用，用以定义 tim_ti1fp1 和 tim_ti2fp1 的极性。请参见 CC1P 说明。
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 2 或 3 且 CC1S=“00”（通道配置为输出），该位立即变为不可写状态。
注意：该位将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1NP 有效位才会从预装载位获取新值。

位 2 **CC1NE**：捕获/比较 1 互补输出使能 (Capture/compare 1 complementary output enable)
0：关闭——tim_oc1n 未激活。tim_oc1n 电平与 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位存在函数关系。
1：开启——基于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值，在相应输出引脚上输出 tim_oc1n 信号。
注意：该位将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1NE 有效位才会从预装载位获取新值。

位 1 **CC1P**：捕获/比较 1 输出极性 (Capture/compare 1 output polarity)

0：OC1 高电平有效（输出模式）/边沿有效性选择（输入模式，见下文）

1：OC1 低电平有效（输出模式）/边沿有效性选择（输入模式，见下文）

CC1 通道配置为输入时，CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的有效极性。

CC1NP=0，CC1P=0： 非反相/上升沿触发。电路对 TIxFP1 上升沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式或编码器模式下执行触发操作）。

CC1NP=0，CC1P=1： 反相/下降沿触发。电路对 TIxFP1 下降沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 反相（在门控模式或编码器模式下执行触发操作）。

CC1NP=1，CC1P=1： 非反相/上升沿和下降沿均触发。电路对 TIxFP1 上升沿和下降沿都敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式下执行触发操作）。编码器模式下不得使用此配置。

CC1NP=1，CC1P=0： 该配置保留，不得使用。

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 2 或 3，该位立即变为不可写状态。

注意：该位将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1P 有效位才会从预装载位获取新值。

位 0 **CC1E**：捕获/比较 1 输出使能 (Capture/compare 1 output enable)

0：禁止捕获模式/OC1 无效（见下文）

1：使能捕获模式/在相应输出引脚上输出 OC1 信号

当 CC1 通道配置为输出时，OC1 电平取决于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位，与 CC1E 位的状态无关。有关详细信息，请参见*表 182*。

注意：该位将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1E 有效位才会从预装载位获取新值。

表 182. 具有刹车功能的互补通道 tim_ocx 和 tim_ocxn 的输出控制位

控制位					输出状态(1)	
MOE 位	OSSI 位	OSSR 位	CCxE 位	CCxNE 位	tim_ocx 输出状态	tim_ocxn 输出状态
1	X	X	0	0	禁止输出（不由定时器驱动：高阻态） tim_ocx=0，tim_ocxn=0	
		0	0	1	禁止输出（不由定时器驱动：高阻态） tim_ocx=0	tim_ocxref + 极性 tim_ocxn = tim_ocxref 异或 CCxNP
		0	1	0	tim_ocxref + 极性 tim_ocx=tim_ocxref 异或 CCxP	禁止输出（不由定时器驱动：高阻态） tim_ocxn=0
		X	1	1	OCREF + 极性 + 死区	OCREF 互补项（对 OCREF 进行“非”运算）+ 极性 + 死区
		1	0	1	关闭状态（输出使能为无效状态） tim_ocx=CCxP	tim_ocxref + 极性 tim_ocxn = tim_ocxref 异或 CCxNP
		1	1	0	tim_ocxref + 极性 tim_ocx=tim_ocxref 异或 CCxP	关闭状态（输出使能为无效状态） tim_ocxn=CCxNP

表 182. 具有刹车功能的互补通道 tim_ocx 和 tim_ocxn 的输出控制位（续）

控制位					输出状态(1)	
MOE 位	OSSI 位	OSSR 位	CCxE 位	CCxNE 位	tim_ocx 输出状态	tim_ocxn 输出状态
0	0	X	X	X	禁止输出（不由定时器驱动：高阻态）。	
	1		0	0		
			0	1	关闭状态（输出使能为无效状态） 异步：tim_ocx=CCxP，tim_ocxn=CCxNP（如果触发 tim_brk 或 tim_brk2）。 随后（仅当触发 tim_brk 时才有效），如果存在时钟：在死区后 tim_ocx=OISx 且 tim_ocxn=OISxN，假定 OISx 和 OISxN 并没有都设置成 OCX 及 tim_ocxn 的有效电平（否则在半桥配置下驱动开关时可能导致短路）。 *注意：tim_brk2 只能在 OSSI = OSSR = 1 时使用。*	
			1	0		
			1	1		

1. 如果一个通道的两个输出均未使用（由 GPIO 接管控制控制），则 OISx、OISxN、CCxP 和 CCxNP 位必须保持清零状态。

注意： *与互补通道 tim_ocx 和 tim_ocxn 相连的外部 I/O 引脚的状态取决于通道 tim_ocx 和 tim_ocxn 的状态以及 GPIO 寄存器。*

26.6.12 TIM1 计数器 (TIM1_CNT)

TIM1 counter

偏移地址：0x024

复位值：0x0000 0000

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

位 31 **UIFCPY**：UIF 副本 (UIF copy)

该位是 TIMx_ISR 寄存器中 UIF 位的只读副本。如果 TIMxCR1 中的 UIFREMAP 位复位，则位 31 保留，读数 0。

位 30:16 保留，必须保持复位值。

位 15:0 **CNT[15:0]**：计数器值 (Counter value)

不抖动模式 (DITHEN = 0)

该寄存器保存计数器值。

抖动模式 (DITHEN = 1)

该寄存器仅在 CNT[15:0] 中保存不抖动部分。小数部分不保存。

26.6.13 TIM1 预分频器 (TIM1_PSC)

TIM1 prescaler

偏移地址：0x028

复位值：0x0000

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

位 15:0 **PSC[15:0]**：预分频器值 (Prescaler value)

计数器时钟频率 ($f_{tim_cnt_ck}$) 等于 $f_{tim_psc_ck}$ / (PSC[15:0] + 1)。

PSC 包含每次发生更新事件（包括计数器通过 TIMx_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）时要装载到实际预分频器寄存器的值。

26.6.14 TIM1 自动重载寄存器 (TIM1_ARR)

TIM1 auto-reload register

偏移地址：0x02C

复位值：0x0000 FFFF

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

位 31:20 保留，必须保持复位值。

位 19:0 **ARR[19:0]**：自动重载值 (Auto-reload value)

ARR 为要装载到实际自动重载寄存器的值。

有关 ARR 更新和行为的更多详细信息，请参见*第 695 页的第 26.3.3 节：时基单元*。

当自动重载值为空时，计数器不工作。

不抖动模式 (DITHEN = 0)

该寄存器保存自动重载值。

抖动模式 (DITHEN = 1)

该寄存器在 ARR[19:4] 中保存整数部分。ARR[3:0] 位域包含抖动部分。

26.6.15 TIM1 重复计数器寄存器 (TIM1_RCR)

TIM1 repetition counter register

偏移地址：0x030

复位值：0x0000

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

位 15:0 **REP[15:0]**：重复计数器重载值 (Repetition counter reload value)

该位域用于在预装载寄存器使能时定义比较寄存器的更新频率（即，从预装载寄存器向影子寄存器周期性传输数据），以及在更新中断使能时定义更新中断的生成速率。

重复递减计数器计数到 0 时，将生成一个更新事件并且计数器从 REP 值重新开始计数。由于只有生成重复更新事件 UEV 时，重复计数器才会重载 REP 值，因此在生成下一重复更新事件之前，向 TIMx_RCR 寄存器写入任何值都无影响。

这意味着 PWM 模式下 (REP+1) 相当于：

– 边沿对齐模式下的 PWM 周期数

– 中心对齐模式下的 PWM 半周期数。

26.6.16 TIM1 捕获/比较寄存器 1 (TIM1_CCR1)

TIM1 capture/compare register 1

偏移地址：0x034

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **CCR1[19:0]**：捕获/比较 1 值 (Capture/compare 1 value)

如果通道 CC1 配置为输出：CCR1 是捕获/比较寄存器 1 的预装载值。

如果没有通过 TIMx_CCMR1 寄存器中的 OC1PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 1）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc1 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器在 CCR1[15:0] 中保存比较值。CCR1[19:16] 位复位。

抖动模式 (DITHEN = 1)

该寄存器在 CCR1[19:4] 中保存整数部分。CCR1[3:0] 位域包含抖动部分。

如果通道 CC1 配置为输入：CR1 为上一个输入捕获 1 事件 (tim_ic1) 发生时的计数器值。只能读取 TIMx_CCR1 寄存器，无法对其进行编程。

不抖动模式 (DITHEN = 0)

该寄存器在 CCR1[15:0] 中保存捕获值。CCR1[19:16] 位复位。

抖动模式 (DITHEN = 1)

该寄存器在 CCR1[19:4] 中保存捕获值。CCR1[3:0] 位复位。

26.6.17 TIM1 捕获/比较寄存器 2 (TIM1_CCR2)

TIM1 capture/compare register 2

偏移地址：0x038

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **CCR2[19:0]**：捕获/比较 2 值 (Capture/compare 2 value)

如果通道 CC2 配置为输出：CCR2 是捕获/比较寄存器 2 的预装载值。
如果没有通过 TIMx_CCMR1 寄存器中的 OC2PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 2）。
有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc2 输出相应电平的值。
不抖动模式 (DITHEN = 0)
该寄存器在 CCR2[15:0] 中保存比较值。CCR2[19:16] 位复位。
抖动模式 (DITHEN = 1)
该寄存器在 CCR2[19:4] 中保存整数部分。CCR2[3:0] 位域包含抖动部分。
如果通道 CC2 配置为输入：CCR2 为上一个输入捕获 2 事件 (tim_ic2) 发生时的计数器值。只能读取 TIMx_CCR2 寄存器，无法对其进行编程。
不抖动模式 (DITHEN = 0)
该寄存器在 CCR2[15:0] 中保存捕获值。CCR2[19:16] 位复位。
抖动模式 (DITHEN = 1)
该寄存器在 CCR2[19:4] 中保存捕获值。CCR2[3:0] 位复位。

26.6.18 TIM1 捕获/比较寄存器 3 (TIM1_CCR3)

TIM1 capture/compare register 3

偏移地址：0x03C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR3[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **CCR3[19:0]**：捕获/比较值 (Capture/compare value)

如果通道 CC3 配置为输出：CCR3 是捕获/比较寄存器 3 的预装载值。
如果没有通过 TIMx_CCMR2 寄存器中的 OC3PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 3）。
有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc3 输出相应电平的值。
不抖动模式 (DITHEN = 0)
该寄存器在 CCR3[15:0] 中保存比较值。CCR3[19:16] 位复位。
抖动模式 (DITHEN = 1)
该寄存器在 CCR3[19:4] 中保存整数部分。CCR3[3:0] 位域包含抖动部分。
如果通道 CC3 配置为输入：CCR3 为上一个输入捕获 3 事件 (tim_ic3) 发生时的计数器值。只能读取 TIMx_CCR3 寄存器，无法对其进行编程。
不抖动模式 (DITHEN = 0)
该寄存器在 CCR3[15:0] 中保存捕获值。CCR3[19:16] 位复位。
抖动模式 (DITHEN = 1)
该寄存器在 CCR3[19:4] 中保存捕获值。CCR3[3:0] 位复位。

26.6.19 TIM1 捕获/比较寄存器 4 (TIM1_CCR4)

TIM1 capture/compare register 4

偏移地址：0x040

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR4[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **CCR4[19:0]**：捕获/比较值 (Capture/compare value)

如果通道 CC4 配置为输出：CCR4 是捕获/比较寄存器 4 的预装载值。
如果没有通过 TIMx_CCMR2 寄存器中的 OC4PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 4）。
有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc4 输出上发出信号的值。
不抖动模式 (DITHEN = 0)
该寄存器在 CCR4[15:0] 中保存比较值。CCR4[19:16] 位复位。
抖动模式 (DITHEN = 1)
该寄存器在 CCR4[19:4] 中保存整数部分。CCR4[3:0] 位域包含抖动部分。
如果通道 CC4 配置为输入：CCR4 为上一个输入捕获 4 事件 (tim_ic4) 发生时的计数器值。只能读取 TIMx_CCR4 寄存器，无法对其进行编程。
不抖动模式 (DITHEN = 0)
该寄存器在 CCR4[15:0] 中保存捕获值。CCR4[19:16] 位复位。
抖动模式 (DITHEN = 1)
该寄存器在 CCR4[19:4] 中保存捕获值。CCR4[3:0] 位复位。

26.6.20 TIM1 刹车和死区寄存器 (TIM1_BDTR)

TIM1 break and dead-time register

偏移地址：0x044

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	BK2BID	BKBID	BK2 DSRM	BK DSRM	BK2P	BK2E	BK2F[3:0]				BKF[3:0]			
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

注意： *由于可以根据 LOCK 配置锁定位 BKBID/BK2BID/BK2P、BK2E、BK2F[3:0]、BKF[3:0]、AOE、BKP、BKE、OSSI、OSSR 和 DTG[7:0] 的写操作，因此必须在第一次对 TIMx_BDTR 寄存器执行写访问时对这些位进行配置。*

位 31:30 保留，必须保持复位值。

位 29 **BK2BID**：刹车 2 双向 (Break2 bidirectional)
请参见 BKBID 说明

位 28 **BKBID**：刹车双向 (Break bidirectional)
0：刹车输入 tim_brk 为输入模式
1：刹车输入 tim_brk 为双向模式
在双向模式下（BKBID 位置 1），刹车输入配置为输入模式和开漏输出模式。任何激活的刹车事件都将使刹车输入上呈逻辑低电平，以向外部器件指示发生了内部刹车事件。
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。
注意：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 27 **BK2DSRM**：刹车 2 解除 (Break2 disarm)
请参见 BKDSRM 说明

位 26 **BKDSRM**：刹车解除 (Break disarm)
0：启动刹车输入 tim_brk
1：解除刹车输入 tim_brk
当没有刹车源被激活时，该位由硬件清零。
必须通过软件将 BKDSRM 位置 1 以释放双向输出控制（开漏输出处于高阻态），然后不断轮询该位，直到其由硬件复位，指示故障条件已消失。
注意：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 25 **BK2P**：刹车 2 极性 (Break2 polarity)
0：刹车输入 tim_brk2 为低电平有效
1：刹车输入 tim_brk2 为高电平有效
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。
注意：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 24 **BK2E**：刹车 2 使能 (Break2 enable)

该位可使能完整的刹车 2 保护（包括连接到 bk_acth 的所有源和相应的 BKIN 源，如*图 208：刹车事件和刹车 2 电路概述*所示）。

0：禁止刹车 2 功能

1：使能刹车 2 功能

注意：BRKIN2 只能在 OSSR = OSSI = 1 时使用。

注意：编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1 后，该位即无法修改。

注意：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 23:20 **BK2F[3:0]**：刹车 2 滤波器 (Break 2 filter)

该位域可定义 tim_brk2 输入的采样频率和适用于 tim_brk2 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿：

0000：无滤波器，tim_brk2 异步工作

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8

1000：$f_{SAMPLING}=f_{DTS}/8$，N=6

1001：$f_{SAMPLING}=f_{DTS}/8$，N=8

1010：$f_{SAMPLING}=f_{DTS}/16$，N=5

1011：$f_{SAMPLING}=f_{DTS}/16$，N=6

1100：$f_{SAMPLING}=f_{DTS}/16$，N=8

1101：$f_{SAMPLING}=f_{DTS}/32$，N=5

1110：$f_{SAMPLING}=f_{DTS}/32$，N=6

1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

注意：编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1 后，该位即无法修改。

位 19:16 **BKF[3:0]**：刹车滤波器 (Break filter)

该位域可定义 tim_brk 输入的采样频率和适用于 tim_brk 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿：

0000：无滤波器，tim_brk 异步工作

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8

1000：$f_{SAMPLING}=f_{DTS}/8$，N=6

1001：$f_{SAMPLING}=f_{DTS}/8$，N=8

1010：$f_{SAMPLING}=f_{DTS}/16$，N=5

1011：$f_{SAMPLING}=f_{DTS}/16$，N=6

1100：$f_{SAMPLING}=f_{DTS}/16$，N=8

1101：$f_{SAMPLING}=f_{DTS}/32$，N=5

1110：$f_{SAMPLING}=f_{DTS}/32$，N=6

1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

注意：编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1 后，该位即无法修改。

位 15 **MOE**：主输出使能 (Main output enable)

只要刹车输入（tim_brk 或 tim_brk2）为有效状态，该位便由硬件异步清零。该位由软件置 1，也可根据 AOE 位状态自动置 1。该位仅对配置为输出的通道有效。

0：响应刹车事件（2 个）。禁止 OC 和 OCN 输出

响应刹车事件或向 MOE 写入 0 时：OC 和 OCN 输出被禁止或被强制为空闲状态，具体取决于 OSSI 位。

1：如果 OC 和 OCN 输出的相应使能位（TIMx_CCER 寄存器中的 CCxE 和 CCxNE 位）均置 1，则使能 OC 和 OCN 输出。

有关详细信息，请参见 OC/OCN 使能说明（*第 26.6.11 节：TIM1 捕获/比较使能寄存器 (TIM1_CCER)*）。

位 14 **AOE**：自动输出使能 (Automatic output enable)

0：MOE 只能由软件置 1

1：MOE 可由软件置 1，也可在发生下一更新事件时自动置 1（如果刹车输入 tim_brk 和 tim_brk2 均无效）

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 13 **BKP**：刹车极性 (Break polarity)

0：刹车输入 tim_brk 为低电平有效

1：刹车输入 tim_brk 为高电平有效

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

注意：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 12 **BKE**：刹车使能 (Break enable)

该位可使能完整的刹车保护（包括连接到 bk_acth 的所有源和相应的 BKIN 源，如*图 208：刹车事件和刹车 2 电路概述*所示）。

0：禁止刹车功能

1：使能刹车功能

注意：编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1 后，该位即无法修改。

注意：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 11 **OSSR**：运行模式下的关闭状态选择 (Off-state selection for Run mode)

该位在 MOE=1 时作用于配置为输出模式且具有互补输出的通道。如果定时器中没有互补输出，则不存在 OSSR。

有关详细信息，请参见 OC/OCN 使能说明（*第 26.6.11 节：TIM1 捕获/比较使能寄存器 (TIM1_CCER)*）。

0：处于无效状态时，禁止 OC/OCN 输出（输出控制器呈高阻态，相应输出由 GPIO 接管）。

1：处于无效状态时，一旦 CCxE=1 或 CCxNE=1，便使能 OC/OCN 输出并将其设为无效电平（输出仍由定时器控制）。

注意：编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 2 后，该位即无法修改。

位 10 **OSSI**：空闲模式下的关闭状态选择 (Off-state selection for idle mode)

当由于刹车事件或软件写操作而使 MOE=0 时，该位作用于配置为输出的通道。

有关详细信息，请参见 OC/OCN 使能说明（*第 26.6.11 节：TIM1 捕获/比较使能寄存器 (TIM1_CCER)*）。

0：处于无效状态时，禁止 OC/OCN 输出（定时器释放输出控制，由强制高阻态的 GPIO 逻辑接管）。

1：处于无效状态时，首先将 OC/OCN 输出强制为其无效电平，然后在死区后将其强制为空闲电平。定时器始终控制输出。

注意：编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 2 后，该位即无法修改。

位 9:8 **LOCK[1:0]**：锁定配置 (Lock configuration)

这些位用于针对软件错误提供写保护。

00：关闭锁定——不对任何位提供写保护。

01：锁定级别 1，此时无法对 TIMx_BDTR 寄存器中的 DTG 位、TIMx_CR2 寄存器中的 OISx 和 OISxN 位以及 TIMx_BDTR 寄存器中的 BKBID/BK2BID/BKE/BKP/AOE 位执行写操作。

10：锁定级别 2，此时无法对锁定级别 1 中适用的各位、CC 极性位（TIMx_CCER 寄存器中的 CCxP/CCxNP 位，只要通过 CCxS 位将相关通道配置为输出）以及 OSSR 和 OSSI 位执行写操作。

11：锁定级别 3，此时无法对锁定级别 2 中适用的各位、CC 控制位（TIMx_CCMRx 寄存器中的 OCxM 和 OCxPE 位，只要通过 CCxS 位将相关通道配置为输出）执行写操作。

注意：复位后只能对 LOCK 位执行一次写操作。对 TIMx_BDTR 寄存器执行写操作后其中的内容将冻结，直到下一次复位。

位 7:0 **DTG[7:0]**：配置死区发生器 (Dead-time generator setup)

该位域定义插入到互补输出之间的死区持续时间。DT 与该持续时间相对应。

DTG[7:5]=0xx => DT=DTG[7:0]x t_{dtg}，其中 $t_{dtg}=t_{DTS}$。

DTG[7:5]=10x => DT=(64+DTG[5:0])xt_{dtg}，其中 $T_{dtg}=2xt_{DTS}$。

DTG[7:5]=110 => DT=(32+DTG[4:0])xt_{dtg}，其中 $T_{dtg}=8xt_{DTS}$。

DTG[7:5]=111 => DT=(32+DTG[4:0])xt_{dtg}，其中 $T_{dtg}=16xt_{DTS}$。

示例：如果 T_{DTS}=125ns (8MHz)，则可能的死区值为：

0 到 15875 ns（步长为 125 ns），

16 us 到 31750 ns（步长为 250 ns），

32 us 到 63us（步长为 1 us），

64 us 到 126 us（步长为 2 us）

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，该位域即无法修改。

26.6.21 TIM1 捕获/比较寄存器 5 (TIM1_CCR5)

TIM1 capture/compare register 5

偏移地址：0x048

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GC5C3	GC5C2	GC5C1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR5[19:16]			
rw	rw	rw										rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR5[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **GC5C3**：通道 5 和通道 3 组合 (Group channel 5 and channel 3)

通道 3 输出上失真：

0：tim_oc5ref 对 tim_oc3refc 无影响

1：tim_oc3refc 是 tim_oc3ref 和 tim_oc5ref 的逻辑与运算结果

该位可以立即生效，也可预装载并在更新事件后执行（如果在 TIMxCCMR2 中选择了预装载功能）。

注意：也可在组合 PWM 信号上应用此失真。

位 30 **GC5C2**：通道 5 和通道 2 组合 (Group channel 5 and channel 2)

通道 2 输出上失真：

0：tim_oc5ref 对 tim_oc2refc 无影响

1：tim_oc2refc 是 tim_oc2ref 和 tim_oc5ref 的逻辑与运算结果

该位可以立即生效，也可预装载并在更新事件后执行（如果在 TIMxCCMR1 中选择了预装载功能）。

注意：也可在组合 PWM 信号上应用此失真。

位 29 **GC5C1**：通道 5 和通道 1 组合 (Group channel 5 and channel 1)

通道 1 输出上失真：

0：oc5ref 对 oc1refc 无影响

1：oc1refc 是 oc1ref 和 oc5ref 的逻辑与运算结果

该位可以立即生效，也可预装载并在更新事件后执行（如果在 TIMxCCMR1 中选择了预装载功能）。

注意：也可在组合 PWM 信号上应用此失真。

位 28:20 保留，必须保持复位值。

位 19:0 **CCR5[19:0]**：捕获/比较 5 值 (Capture/compare 5 value)

CCR5 是捕获/比较寄存器 5 的预装载值。

如果没有通过 TIMx_CCMR3 寄存器中的 OC5PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 5）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc5 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器在 CCR5[15:0] 中保存比较值。CCR5[19:16] 位复位。

抖动模式 (DITHEN = 1)

该寄存器在 CCR5[19:4] 中保存整数部分。CCR5[3:0] 位域包含抖动部分。

26.6.22 TIM1 捕获/比较寄存器 6 (TIM1_CCR6)

TIM1 capture/compare register 6

偏移地址：0x04C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR6[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR6[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **CCR6[19:0]**：捕获/比较 6 值 (Capture/compare 6 value)

CCR6 是捕获/比较寄存器 6 的预装载值。

如果没有通过 TIMx_CCMR3 寄存器中的 OC6PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 6）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc6 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器在 CCR6[15:0] 中保存比较值。CCR6[19:16] 位复位。

抖动模式 (DITHEN = 1)

该寄存器在 CCR6[19:4] 中保存整数部分。CCR6[3:0] 位域包含抖动部分。

26.6.23 TIM1 捕获/比较模式寄存器 3 (TIM1_CCMR3)

TIM1 capture/compare mode register 3

偏移地址：0x050

复位值：0x0000 0000

请参见上述 CCMR1 寄存器说明。通道 5 和通道 6 只能配置为输出。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC6M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC5M[3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC6 CE	OC6M[2:0]			OC6 PE	OC6FE	Res.	Res.	OC5 CE	OC5M[2:0]			OC5PE	OC5FE	Res.	Res.
rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw		

位 31:25 保留，必须保持复位值。

位 23:17 保留，必须保持复位值。

位 15 **OC6CE**：输出比较 6 清零使能 (Output compare 6 clear enable)

位 24、14:12 **OC6M[3:0]**：输出比较 6 模式 (Output compare 6 mode)

位 11 **OC6PE**：输出比较 6 预装载使能 (Output compare 6 preload enable)

位 10 **OC6FE**：输出比较 6 快速使能 (Output compare 6 fast enable)

位 9:8 保留，必须保持复位值。

位 7 **OC5CE**：输出比较 5 清零使能 (Output compare 5 clear enable)

位 16，6:4 **OC5M[3:0]**：输出比较 5 模式 (Output compare 5 mode)

位 3 **OC5PE**：输出比较 5 预装载使能 (Output compare 5 preload enable)

位 2 **OC5FE**：输出比较 5 快速使能 (Output compare 5 fast enable)

位 1:0 保留，必须保持复位值。

26.6.24 TIM1 定时器死区寄存器 2 (TIM1_DTR2)

TIM1 timer deadtime register 2

偏移地址：0x054

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTPE	DTAE
														rw	rw

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

位 31:18 保留，必须保持复位值。

位 17 **DTPE**：死区预装载使能 (Deadtime preload enable)

0：禁止死区值预装载

1：使能死区值预装载

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，该位即无法修改。

位 16 **DTAE**：死区不对称使能 (Deadtime asymmetric enable)

0：上升沿和下降沿的死区相同，由寄存器 DTG[7:0] 定义

1：上升沿死区由寄存器 DTG[7:0] 定义，下降沿死区由 DTGF[7:0] 位定义。

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，该位即无法修改。

位 15:8 保留，必须保持复位值。

位 7:0 **DTGF[7:0]**：配置下降沿死区发生器 (Dead-time falling edge generator setup)

该位域定义插入到互补输出之间的下降沿死区持续时间。

DTGF[7:5]=0xx => DTF=DTGF[7:0]x t_{dtg}，其中 $t_{dtg}=t_{DTS}$。

DTGF[7:5]=10x => DTF=(64+DTGF[5:0])xt_{dtg}，其中 $T_{dtg}=2xt_{DTS}$。

DTGF[7:5]=110 => DTF=(32+DTGF[4:0])xt_{dtg}，其中 $T_{dtg}=8xt_{DTS}$。

DTGF[7:5]=111 => DTF=(32+DTGF[4:0])xt_{dtg}，其中 $T_{dtg}=16xt_{DTS}$。

示例：如果 T_{DTS}=125ns (8MHz)，则可能的死区值为：

0 到 15875 ns（步长为 125 ns），

16 us 到 31750 ns（步长为 250 ns），

32 us 到 63us（步长为 1 us），

64 us 到 126 us（步长为 2 us）

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，该位域即无法修改。

26.6.25 TIM1 定时器编码器控制寄存器 (TIM1_ECR)

TIM1 timer encoder control register

偏移地址：0x058

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	PWPRSC[2:0]			PW[7:0]							
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IPOS[1:0]		FIDX	IBLK[1:0]		IDIR[1:0]		IE
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:27 保留，必须保持复位值。

位 26:24 **PWPRSC[2:0]**：脉冲宽度预分频器 (Pulse width prescaler)

该位域用于设置脉冲发生器的时钟预分频器，具体如下所示：

$t_{PWG} = (2^{(PWPRSC[2:0])}) \times t_{tim_ker_ck}$

位 23:16 **PW[7:0]**：脉冲宽度 (Pulse width)

该位域用于定义脉冲的持续时间，具体如下所示：

$t_{PW} = PW[7:0] \times t_{PWG}$

位 15:8 保留，必须保持复位值。

位 7:6 **IPOS[1:0]**：索引定位 (Index positioning)

在正交编码器模式下（SMS[3:0] = 0001、0010、0011、1110、1111），该位指示索引事件在哪种 AB 输入配置下复位计数器。

00：当 AB = 00 时，索引事件复位计数器
01：当 AB = 01 时，索引事件复位计数器
10：当 AB = 10 时，索引事件复位计数器
11：当 AB = 11 时，索引事件复位计数器

在定向时钟模式或时钟 + 方向模式（SMS[3:0] = 1010、1011、1100、1101）下，这些位指示索引事件在哪种电平状态下复位计数器。在双向时钟模式下，对两个时钟输入均适用。

x0：当时钟为低电平时，索引事件复位计数器
x1：当时钟为高电平时，索引事件复位计数器

注意：IPOS[1] 位无意义

位 5 **FIDX**：第一个索引 (First index)

该位指示是否只需考虑第一个索引

0：索引始终有效
1：第一个索引仅复位计数器

位 4:3 **IBLK[1:0]**：索引消隐 (Index blanking)

该位指示索引事件由 tim_ti3 输入还是 tim_ti4 输入限制

00：索引始终有效
01：根据 CC3P 位域，当 tim_ti3 输入有效时禁止索引
10：根据 CC4P 位域，当 tim_ti4 输入有效时禁止索引
11：保留

位 2:1 **IDIR[1:0]**：索引方向 (Index direction)

该位指示索引事件在哪种计数方向下复位计数器。

00：无论计数方向如何，索引事件都会复位计数器
01：索引事件仅在递增计数时复位计数器
10：索引事件仅在递减计数时复位计数器
11：保留

位 0 **IE**：索引使能 (Index enable)

该位指示索引事件是否复位计数器。

0：禁止索引
1：使能索引

26.6.26 TIM1 定时器输入选择寄存器 (TIM1_TISEL)

TIM1 timer input selection register

偏移地址：0x05C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TI4SEL[3:0]				Res.	Res.	Res.	Res.	TI3SEL[3:0]			
				rw	rw	rw	rw					rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
				rw	rw	rw	rw					rw	rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27:24 **TI4SEL[3:0]**：选择 tim_ti4[15:0] 输入 (Selects tim_ti4[15:0] input)
0000：tim_ti4_in0：TIMx_CH4
0001：tim_ti4_in1
...
1111：tim_ti4_in15
有关互连表，请参见*第 26.3.2 节：TIM1 引脚和内部信号*。

位 23:20 保留，必须保持复位值。

位 19:16 **TI3SEL[3:0]**：选择 tim_ti3[15:0] 输入 (Selects tim_ti3[15:0] input)
0000：tim_ti3_in0：TIMx_CH2
0001：tim_ti3_in1
...
1111：tim_ti3_in15
有关互连表，请参见*第 26.3.2 节：TIM1 引脚和内部信号*。

位 15:12 保留，必须保持复位值。

位 11:8 **TI2SEL[3:0]**：选择 tim_ti2[15:0] 输入 (Selects tim_ti2[15:0] input)
0000：tim_ti2_in0：TIMx_CH2
0001：tim_ti2_in1
...
1111：tim_ti2_in15
有关互连表，请参见*第 26.3.2 节：TIM1 引脚和内部信号*。

位 7:4 保留，必须保持复位值。

位 3:0 **TI1SEL[3:0]**：选择 tim_ti1[15:0] 输入 (Selects tim_ti1[15:0] input)
0000：tim_ti1_in0：TIMx_CH1
0001：tim_ti1_in1
...
1111：tim_ti1_in15
有关互连表，请参见*第 26.3.2 节：TIM1 引脚和内部信号*。

26.6.27 TIM1 复用功能选项寄存器 1 (TIM1_AF1)

TIM1 alternate function option register 1

偏移地址：0x060

复位值：0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETRSEL[3:2]	
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL[1:0]		BK CMP4P	BK CMP3P	BK CMP2P	BK CMP1P	BKINP	BK CMP8E	BK CMP7E	BK CMP6E	BK CMP5E	BK CMP4E	BK CMP3E	BK CMP2E	BK CMP1E	BKINE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:18 保留，必须保持复位值。

位 17:14 **ETRSEL[3:0]**：etr_in 源选择 (etr_in source selection)
这些位选择 etr_in 输入源。
0000：tim_etr0：TIMx_ETR 输入
0001：tim_etr1
...
1111：tim_etr15
有关产品特定的实现，请参见*第 26.3.2 节：TIM1 引脚和内部信号*。
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，这些位即无法修改。

位 13 **BKCMP4P**：tim_brk_cmp4 输入极性 (tim_brk_cmp4 input polarity)
该位选择 tim_brk_cmp4 输入有效电平，必须与 BKP 极性位一起编程。
0：tim_brk_cmp4 输入极性不反相（BKP = 0 时低电平有效，BKP = 1 时高电平有效）
1：tim_brk_cmp4 输入极性反相（BKP = 0 时高电平有效，BKP = 1 时低电平有效）
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 12 **BKCMP3P**：tim_brk_cmp3 输入极性 (tim_brk_cmp3 input polarity)
该位选择 tim_brk_cmp3 输入有效电平，必须与 BKP 极性位一起编程。
0：tim_brk_cmp3 输入极性不反相（BKP = 0 时低电平有效，BKP = 1 时高电平有效）
1：tim_brk_cmp3 输入极性反相（BKP = 0 时高电平有效，BKP = 1 时低电平有效）
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 11 **BKCMP2P**：tim_brk_cmp2 输入极性 (tim_brk_cmp2 input polarity)
该位选择 tim_brk_cmp2 输入有效电平，必须与 BKP 极性位一起编程。
0：tim_brk_cmp2 输入极性不反相（BKP = 0 时低电平有效，BKP = 1 时高电平有效）
1：tim_brk_cmp2 输入极性反相（BKP = 0 时高电平有效，BKP = 1 时低电平有效）
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 10 **BKCMP1P**：tim_brk_cmp1 输入极性 (tim_brk_cmp1 input polarity)
该位选择 tim_brk_cmp1 输入有效电平，必须与 BKP 极性位一起编程。
0：tim_brk_cmp1 输入极性不反相（BKP = 0 时低电平有效，BKP = 1 时高电平有效）
1：tim_brk_cmp1 输入极性反相（BKP = 0 时高电平有效，BKP = 1 时低电平有效）
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 9 **BKINP**：TIMx_BKIN 输入极性 (TIMx_BKIN input polarity)
该位选择 TIMx_BKIN 复用功能输入有效电平，必须与 BKP 极性位一起编程。
0：TIMx_BKIN 输入极性不反相（BKP = 0 时低电平有效，BKP = 1 时高电平有效）
1：TIMx_BKIN 输入极性反相（BKP = 0 时高电平有效，BKP = 1 时低电平有效）
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 8 **BKCMP8E**：tim_brk_cmp8 使能 (tim_brk_cmp8 enable)
该位使能定时器 tim_brk 输入的 tim_brk_cmp8。tim_brk_cmp8 输出与其他 tim_brk 源进行“或”运算。
0：禁止 tim_brk_cmp8 输入
1：使能 tim_brk_cmp8 输入
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 7 **BKCMP7E**：tim_brk_cmp7 使能 (tim_brk_cmp7 enable)
该位使能定时器 tim_brk 输入的 tim_brk_cmp7。tim_brk_cmp7 输出与其他 tim_brk 源进行“或”运算。
0：禁止 tim_brk_cmp7 输入
1：使能 tim_brk_cmp7 输入
注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 6 **BKCMP6E**：tim_brk_cmp6 使能 (tim_brk_cmp6 enable)

该位使能定时器 tim_brk 输入的 tim_brk_cmp6。tim_brk_cmp6 输出与其他 tim_brk 源进行“或”运算。

0：禁止 tim_brk_cmp6 输入

1：使能 tim_brk_cmp6 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 5 **BKCMP5E**：tim_brk_cmp5 使能 (tim_brk_cmp5 enable)

该位使能定时器 tim_brk 输入的 tim_brk_cmp5。tim_brk_cmp5 输出与其他 tim_brk 源进行“或”运算。

0：禁止 tim_brk_cmp5 输入

1：使能 tim_brk_cmp5 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 4 **BKCMP4E**：tim_brk_cmp4 使能 (tim_brk_cmp4 enable)

该位使能定时器 tim_brk 输入的 tim_brk_cmp4。tim_brk_cmp4 输出与其他 tim_brk 源进行“或”运算。

0：禁止 tim_brk_cmp4 输入

1：使能 tim_brk_cmp4 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 3 **BKCMP3E**：tim_brk_cmp3 使能 (tim_brk_cmp3 enable)

该位使能定时器 tim_brk 输入的 tim_brk_cmp3。tim_brk_cmp3 输出与其他 tim_brk 源进行“或”运算。

0：禁止 tim_brk_cmp3 输入

1：使能 tim_brk_cmp3 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 2 **BKCMP2E**：tim_brk_cmp2 使能 (tim_brk_cmp2 enable)

该位使能定时器 tim_brk 输入的 tim_brk_cmp2。tim_brk_cmp2 输出与其他 tim_brk 源进行“或”运算。

0：禁止 tim_brk_cmp2 输入

1：使能 tim_brk_cmp2 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 1 **BKCMP1E**：tim_brk_cmp1 使能 (tim_brk_cmp1 enable)

该位使能定时器 tim_brk 输入的 tim_brk_cmp1。tim_brk_cmp1 输出与其他 tim_brk 源进行“或”运算。

0：禁止 tim_brk_cmp1 输入

1：使能 tim_brk_cmp1 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 0 **BKINE**：TIMx_BKIN 输入使能 (TIMx_BKIN input enable)

该位使能定时器 tim_brk 输入的 TIMx_BKIN 复用功能。TIMx_BKIN 输入与其他 tim_brk 源进行“或”运算。

0：禁止 TIMx_BKIN 输入

1：使能 TIMx_BKIN 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

注意：　有关产品特定的实现，请参见第 26.3.2 节：TIM1 引脚和内部信号。

26.6.28 TIM1 复用功能寄存器 2 (TIM1_AF2)

TIM1 alternate function register 2

偏移地址：0x064

复位值：0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OCRSEL[2:0]		
													rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	BK2CMP4P	BK2CMP3P	BK2CMP2P	BK2CMP1P	BK2INP	BK2CMP8E	BK2CMP7E	BK2CMP6E	BK2CMP5E	BK2CMP4E	BK2CMP3E	BK2CMP2E	BK2CMP1E	BK2INE
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:19 保留，必须保持复位值。

位 18:16 **OCRSEL[2:0]**：ocref_clr 源选择 (ocref_clr source selection)

这些位选择 ocref_clr 输入源。

000：tim_ocref_clr0

001：tim_ocref_clr1

...

111：tim_ocref_clr7

有关产品特定的信息，请参见*第 26.3.2 节：TIM1 引脚和内部信号*。

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，这些位即无法修改。

位 15:14 保留，必须保持复位值。

位 13 **BK2CMP4P**：tim_brk2_cmp4 输入极性 (tim_brk2_cmp4 input polarity)

该位选择 tim_brk2_cmp4 输入有效电平，必须与 BK2P 极性位一起编程。

0：tim_brk2_cmp4 输入极性不反相（BK2P = 0 时低电平有效，BK2P = 1 时高电平有效）

1：tim_brk2_cmp4 输入极性反相（BK2P = 0 时高电平有效，BK2P = 1 时低电平有效）

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 12 **BK2CMP3P**：tim_brk2_cmp3 输入极性 (tim_brk2_cmp3 input polarity)

该位选择 tim_brk2_cmp3 输入有效电平，必须与 BK2P 极性位一起编程。

0：tim_brk2_cmp3 输入极性不反相（BK2P = 0 时低电平有效，BK2P = 1 时高电平有效）

1：tim_brk2_cmp3 输入极性反相（BK2P = 0 时高电平有效，BK2P = 1 时低电平有效）

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 11 **BK2CMP2P**：tim_brk2_cmp2 输入极性 (tim_brk2_cmp2 input polarity)

该位选择 tim_brk2_cmp2 输入有效电平，必须与 BK2P 极性位一起编程。

0：tim_brk2_cmp2 输入极性不反相（BK2P = 0 时低电平有效，BK2P = 1 时高电平有效）

1：tim_brk2_cmp2 输入极性反相（BK2P = 0 时高电平有效，BK2P = 1 时低电平有效）

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 10 **BK2CMP1P**：tim_brk2_cmp1 输入极性 (tim_brk2_cmp1 input polarity)

该位选择 tim_brk2_cmp1 输入有效电平，必须与 BK2P 极性位一起编程。

0：tim_brk2_cmp1 输入极性不反相（BK2P = 0 时低电平有效，BK2P = 1 时高电平有效）

1：tim_brk2_cmp1 输入极性反相（BK2P = 0 时高电平有效，BK2P = 1 时低电平有效）

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 9 **BK2INP**：TIMx_BKIN2 输入极性 (TIMx_BKIN2 input polarity)

该位选择 TIMx_BKIN2 复用功能输入有效电平，必须与 BK2P 极性位一起编程。

0：TIMx_BKIN2 输入极性不反相（BK2P = 0 时低电平有效，BK2P = 1 时高电平有效）

1：TIMx_BKIN2 输入极性反相（BK2P = 0 时高电平有效，BK2P = 1 时低电平有效）

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 8 **BK2CMP8E**：tim_brk2_cmp8 使能 (tim_brk2_cmp8 enable)

该位使能定时器 tim_brk2 输入的 tim_brk2_cmp8。tim_brk2_cmp8 输出与其他 tim_brk2 源进行“或”运算。

0：禁止 tim_brk2_cmp8 输入

1：使能 tim_brk2_cmp8 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 7 **BK2CMP7E**：tim_brk2_cmp7 使能 (tim_brk2_cmp7 enable)

该位使能定时器 tim_brk2 输入的 tim_brk2_cmp7。tim_brk2_cmp7 输出与其他 tim_brk2 源进行“或”运算。

0：禁止 tim_brk2_cmp7 输入

1：使能 tim_brk2_cmp7 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 6 **BK2CMP6E**：tim_brk2_cmp6 使能 (tim_brk2_cmp6 enable)

该位使能定时器 tim_brk2 输入的 tim_brk2_cmp6。tim_brk2_cmp6 输出与其他 tim_brk2 源进行“或”运算。

0：禁止 tim_brk2_cmp6 输入

1：使能 tim_brk2_cmp6 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 5 **BK2CMP5E**：tim_brk2_cmp5 使能 (tim_brk2_cmp5 enable)

该位使能定时器 tim_brk2 输入的 tim_brk2_cmp5。tim_brk2_cmp5 输出与其他 tim_brk2 源进行“或”运算。

0：禁止 tim_brk2_cmp5 输入

1：使能 tim_brk2_cmp5 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 4 **BK2CMP4E**：tim_brk2_cmp4 使能 (tim_brk2_cmp4 enable)

该位使能定时器 tim_brk2 输入的 tim_brk2_cmp4。tim_brk2_cmp4 输出与其他 tim_brk2 源进行“或”运算。

0：禁止 tim_brk2_cmp4 输入

1：使能 tim_brk2_cmp4 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 3 **BK2CMP3E**：tim_brk2_cmp3 使能 (tim_brk2_cmp3 enable)

该位使能定时器 tim_brk2 输入的 tim_brk2_cmp3。tim_brk2_cmp3 输出与其他 tim_brk2 源进行“或”运算。

0：禁止 tim_brk2_cmp3 输入

1：使能 tim_brk2_cmp3 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 2 **BK2CMP2E**：tim_brk2_cmp2 使能 (tim_brk2_cmp2 enable)

该位使能定时器 tim_brk2 输入的 tim_brk2_cmp2。tim_brk2_cmp2 输出与其他 tim_brk2 源进行“或”运算。

0：禁止 tim_brk2_cmp2 输入

1：使能 tim_brk2_cmp2 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 1 **BK2CMP1E**：tim_brk2_cmp1 使能 (tim_brk2_cmp1 enable)

该位使能定时器 tim_brk2 输入的 tim_brk2_cmp1。tim_brk2_cmp1 输出与其他 tim_brk2 源进行“或”运算。

0：禁止 tim_brk2_cmp1 输入

1：使能 tim_brk2_cmp1 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

位 0 **BK2INE**：TIMx_BKIN2 输入使能 (TIMx_BKIN2 input enable)

该位使能定时器 tim_brk2 输入的 TIMx_BKIN2 复用功能。TIMx_BKIN2 输入与其他 tim_brk2 源进行“或”运算。

0：禁止 TIMx_BKIN2 输入

1：使能 TIMx_BKIN2 输入

注意：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，该位即无法修改。

注意：有关产品特定的实现，请参见第 26.3.2 节：TIM1 引脚和内部信号。

26.6.29 TIM1 DMA 控制寄存器 (TIM1_DCR)

TIM1 DMA control register

偏移地址：0x3DC

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBSS[3:0]			
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DBL[4:0]					Res.	Res.	Res.	DBA[4:0]				
			rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:16 **DBSS[3:0]**：DMA 分组传送源选择 (DMA burst source selection)

该位域定义触发 DMA 分组传送的中断源（当对 TIMx_DMAR 地址进行读或写访问时，定时器进行一次分组传送）。

0000：保留

0001：更新

0010：CC1

0011：CC2

0100：CC3

0101：CC4

0110：COM

0111：触发

其他值：保留

位 15:13 保留，必须保持复位值。

位 12:8 **DBL[4:0]**：DMA 分组传送长度 (DMA burst length)

该 5 位向量定义了 DMA 的传送长度（当对 TIMx_DMAR 地址进行读或写访问时，定时器进行一次分组传送），即传送次数。可按半字或字节进行传送（请参见下面的示例）。

00000：1 次传输

00001：2 次传输

00010：3 次传输

...

11010：26 次传输

示例：考虑以下传输：DBL = 7 字节，DBA = TIM2_CR1。

– 如果 DBL = 7 字节且 DBA = TIM2_CR1 表示待传送字节的地址，则该地址可通过以下公式计算得出：

(TIMx_CR1 地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL

在本例中，将为 (TIMx_CR1 地址) + DBA 加上 7 个字节，得到将要复制数据的源/目标地址。这种情况下将向自以下地址开始的 7 个寄存器传输数据：(TIMx_CR1 地址) + DBA

根据 DMA 数据大小的配置，可能发生下面几种情况：

– 如果按半字配置 DMA 数据大小，则将向 7 个寄存器中的每一个传送 16 位数据。

– 如果按字节配置 DMA 数据大小，也将向 7 个寄存器传送数据：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，依此类推。因此，使用传送定时器时，还必须指定 DMA 传送的数据大小。

位 7:5 保留，必须保持复位值。

位 4:0 **DBA[4:0]**：DMA 基址 (DMA base address)

该 5 位向量定义 DMA 传输的基址（通过 TIMx_DMAR 地址进行读/写访问时）。DBA 定义为从 TIMx_CR1 寄存器地址开始计算的偏移量。

示例：

00000：TIMx_CR1

00001：TIMx_CR2

00010：TIMx_SMCR

...

26.6.30 TIM1 全传输 DMA 地址 (TIM1_DMAR)

TIM1 DMA address for full transfer

偏移地址：0x3E0

复位值：0x0000 0000

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

位 31:0 **DMAB[31:0]**：DMA 分组传送寄存器 (DMA register for burst accesses)

对 DMAR 寄存器执行读或写操作将访问位于如下地址的寄存器：(TIMx_CR1 地址) + (DBA + DMA 索引) x 4

其中 TIMx_CR1 地址为控制寄存器 1 的地址，DBA 为 TIMx_DCR 寄存器中配置的 DMA 基址，DMA 索引由 DMA 传输自动控制，其范围介于 0 到 DBL（TIMx_DCR 寄存器中配置的 DBL）之间。

26.6.31 TIM1 寄存器映射

TIM1 寄存器可映射为 16 位可寻址寄存器，如下表所述：

表 183. TIM1 寄存器映射和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**TIMx_CR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DITHEN	UIFREMAP	Res.	CKD [1:0]		ARPE	CMS [1:0]		DIR	OPM	URS	UDIS	CEN
	Reset value																				0	0		0	0	0	0	0	0	0	0	0	0
0x004	**TIMx_CR2**	Res.	Res.	Res.	Res.	Res.	Res.	MMS[3]	Res.	MMS2[3:0]				Res.	OIS6	Res.	OIS5	OIS4N	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS [2:0]			CCDS	CCUS	Res.	CCPC
	Reset value							0		0	0	0	0		0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0
0x008	**TIMx_SMCR**	Res.	Res.	Res.	Res.	Res.	Res.	SMSPS	SMSPE	Res.	Res.	TS [4:3]		Res.	Res.	Res.	SMS[3]	ETP	ECE	ETP S [1:0]		ETF[3:0]			MSM	TS[2:0]			OCCS	SMS[2:0]			
	Reset value							0	0			0	0				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00C	**TIMx_DIER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TERRIE	IERRIE	DIRIE	IDXIE	Res.	Res.	Res.	Res.	Res.	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset value									0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x010	**TIMx_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TERRF	IERRF	DIRF	IDXF	Res.	Res.	CC6IF	CC5IF	Res.	Res.	SBIF	CC4OF	CC3OF	CC2OF	CC1OF	B2IF	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset value									0	0	0	0			0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x014	**TIMx_EGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	B2G	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																								0	0	0	0	0	0	0	0	0
0x018	**TIMx_CCMR1** Input Capture mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2F[3:0]				IC2 PSC [1:0]		CC2 S [1:0]		IC1F[3:0]				IC1 PSC [1:0]		CC1 S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIMx_CCMR1 Output Compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[3]	OC2CE	OC2M [2:0]			OC2PE	OC2FE	CC2 S [1:0]		OC1CE	OC1M [2:0]			OC1PE	OC1FE	CC1 S [1:0]	
	Reset value								0								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x01C	**TIMx_CCMR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC3M[3]	OC4CE	OC4M [2:0]			OC4PE	OC4FE	CC4 S [1:0]		OC3CE	OC3M [2:0]			OC3PE	OC3FE	CC3 S [1:0]	
	Reset value								0								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIMx_CCMR2	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4F[3:0]				IC4 PSC [1:0]		CC4 S [1:0]		IC3F[3:0]				IC3 PSC [1:0]		CC3 S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x020	**TIMx_CCER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC6P	CC6E	Res.	Res.	CC5P	CC5E	CC4NP	CC4NE	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
	Reset value											0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x024	**TIMx_CNT**	UIFCPY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 183. TIM1 寄存器映射和复位值（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x028	**TIMx_PSC**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x02C	**TIMx_ARR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[19:0]																			
	Reset value													0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x030	**TIMx_RCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x034	**TIMx_CCR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[19:0]																			
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x038	**TIMx_CCR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[19:0]																			
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x03C	**TIMx_CCR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR3[19:0]																			
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x040	**TIMx_CCR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR4[19:0]																			
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x044	**TIMx_BDTR**	Res.	Res.	BK2BID	BKBID	BK2DSRM	BKDSRM	BK2P	BK2E	BK2F[3:0]				BKF[3:0]				MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK [1:0]		DT[7:0]							
	Reset value			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x048	**TIMx_CCR5**	GC5C3	GC5C2	GC5C1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR5[19:0]																			
	Reset value	0	0	0										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04C	**TIMx_CCR6**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR6[19:0]																			
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x050	**TIMx_CCMR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC6M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC5M[3]	OC6CE	OC6M [2:0]			OC6PE	OC6FE	Res.	Res.	OC5CE	OC5M [2:0]			OC5PE	OC5FE	Res.	Res.
	Reset value								0								0	0	0	0	0	0	0			0	0	0	0	0	0		
0x054	**TIMx_DTR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTPE	DTAE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTGF[7:0]							
	Reset value															0	0									0	0	0	0	0	0	0	0
0x058	**TIMx_ECR**	Res.	Res.	Res.	Res.	Res.	PWPRSC[2:0]			PW[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IPOS [1:0]		FIDX	IBLK [1:0]		IDIR [1:0]		IE
	Reset value						0	0	0	0	0	0	0	0	0	0	0									0	0	0	0	0	0	0	0
0x05C	**TIMx_TISEL**	Res.	Res.	Res.	Res.	TI4SEL[3:0]				Res.	Res.	Res.	Res.	TI3SEL[3:0]				Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
	Reset value					0	0	0	0					0	0	0	0					0	0	0	0					0	0	0	0

表 183. TIM1 寄存器映射和复位值（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x060	**TIMx_AF1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res ETRSEL [3:0]				BKCMP4P	BKCMP3P	BKCMP2P	BKCMP1P	BKINP	BKCMP8E	BKCMP7E	BKCMP6E	BKCMP5E	BKCMP4E	BKCMP3E	BKCMP2E	BKCMP1E	BKINE
	Reset value															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x064	**TIMx_AF2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OCRSEL [2:0]			Res.	Res.	BK2CMP4P	BK2CMP3P	BK2CMP2P	BK2CMP1P	BK2INP	BK2CMP8E	BK2CMP7E	BK2CMP6E	BK2CMP5E	BK2CMP4E	BK2CMP3E	BK2CMP2E	BK2CMP1E	BK2INE
	Reset value														0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x068.. 0x3D8	Reserved	Res.																															
0x3DC	**TIMx_DCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBSS[3:0]				Res.	Res.	Res	DBL[4:0]					Res.	Res.	Res.	DBA[4:0]				
	Reset value													0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x3E0	**TIMx_DMAR**	DMAB[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

27 通用定时器 (TIM2/TIM3)

27.1 TIM2/TIM3 简介

通用定时器包含一个 16 位或 32 位自动重载计数器，该计数器由可编程预分频器驱动。

它们可用于多种用途，包括测量输入信号的脉冲宽度（*输入捕获*）或生成输出波形（*输出比较和 PWM*）。

使用定时器预分频器和 RCC 时钟控制器预分频器，可将脉冲宽度和波形周期从几微秒调制到几毫秒。

这些定时器彼此完全独立，不共享任何资源。如*第 27.4.23 节：定时器同步*中所述，它们可以一起同步操作。

27.2 TIM2/TIM3 主要特性

通用 TIMx 定时器具有以下特性：

- 16 位或 32 位递增、递减、递增/递减自动重载计数器。
- 16 位可编程预分频器，用于对计数器时钟频率进行分频（可在运行时修改），分频系数介于 1 和 65535 之间。
- 多达 4 个独立通道，可用于：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿对齐模式和中心对齐模式）
  - 单脉冲模式输出
- 同步电路可实现外部信号对定时器的控制以及多个定时器互联。
- 发生如下事件时生成中断/DMA 请求：
  - 更新：计数器上溢/下溢、计数器初始化（通过软件或内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或通过内部/外部触发计数）
  - 输入捕获
  - 输出比较
- 支持定位用增量（正交）编码器和霍尔传感器电路
- 触发输入可用作外部时钟或用于逐周期电流管理

27.3 TIM2/TIM3 实现

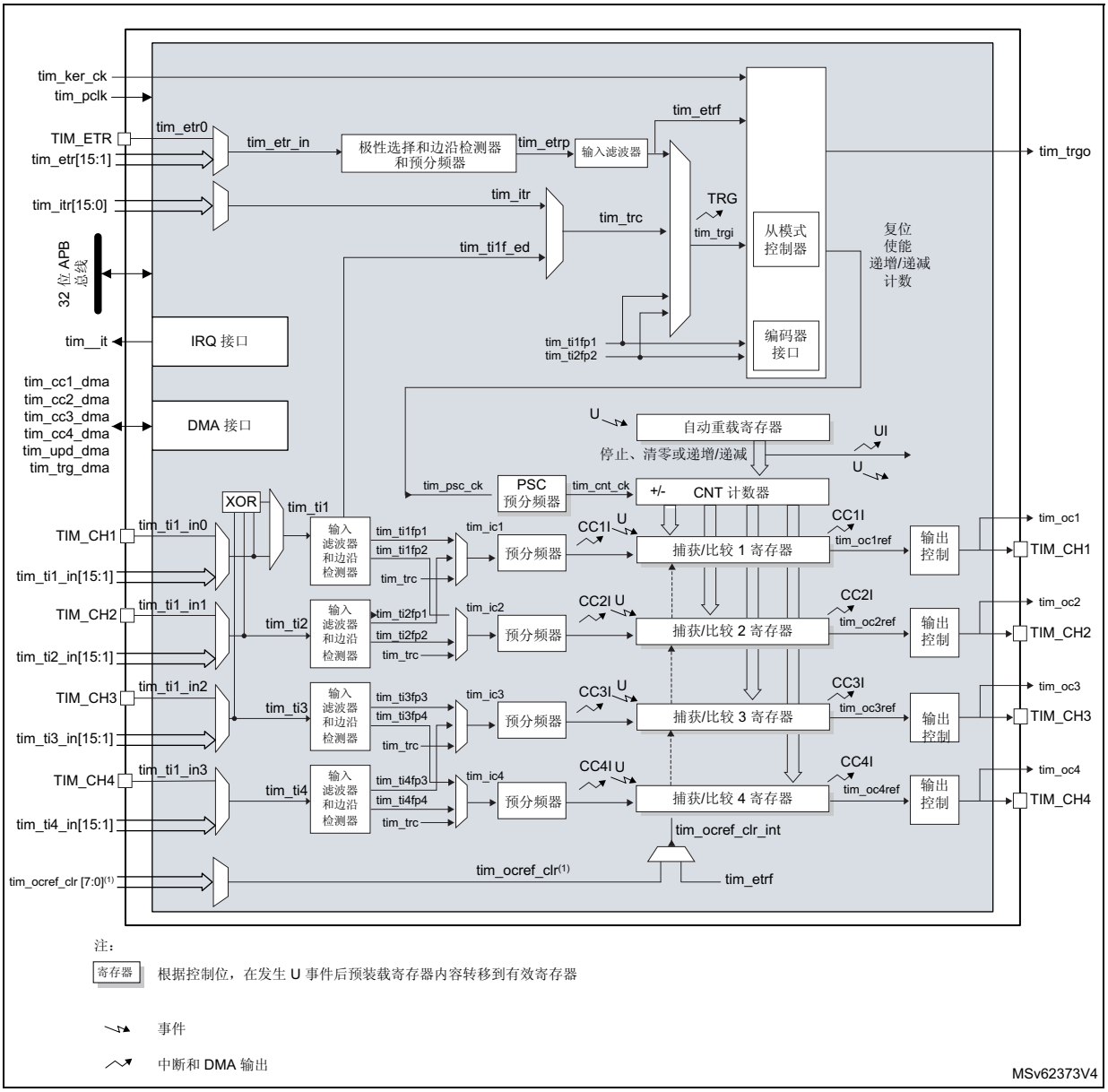
表 184. STM32H503xx 通用定时器

定时器实例	TIM2	TIM3
分辨率	32 位	16 位
OCREF 清零选择源	是 tim_etrf 保留	是 tim_etrf 保留

27.4 TIM2/TIM3 功能描述

27.4.1 框图

图 251. 通用定时器框图



1. 并非所有定时器都具有此特性，请参见*第 27.3 节：TIM2/TIM3 实现*。

27.4.2 TIM2/TIM3 引脚和内部信号

本节中的*表 185* 和*表 186* 对 TIM 输入和输出进行了汇总。

表 185. TIM 输入/输出引脚

引脚名称	信号类型	说明
TIM_CH1 TIM_CH2 TIM_CH3 TIM_CH4	输入/输出	定时器多用途通道。 每个通道均可用于捕获、比较或 PWM。 TIM_CH1 和 TIM_CH2 还可用作外部时钟（低于 1/4 tim_ker_ck 时钟频率）、外部触发输入和正交编码器输入。 TIM_CH1、TIM_CH2 和 TIM_CH3 可用于连接数字霍尔效应传感器。
TIM_ETR	输入	外部触发输入。该输入可用作外部触发信号或外部时钟源。如果使用 tim_etr_in 预分频器，则该输入可以接收频率高于 tim_ker_ck 的时钟。

表 186. TIM 内部输入/输出信号

内部信号名称	信号类型	说明
tim_ti1_in[15:0] tim_ti2_in[15:0] tim_ti3_in[15:0] tim_ti4_in[15:0]	输入	内部定时器输入总线。tim_ti1_in[15:0] 和 tim_ti2_in[15:0] 输入可用于捕获或用作外部时钟（低于 1/4 tim_ker_ck 时钟频率）以及正交编码器信号。
tim_etr[15:0]	输入	外部触发内部输入总线。这些输入可用作触发信号、外部时钟或用于硬件逐周期脉宽控制。如果使用 tim_etr_in 预分频器，则这些输入可以接收频率高于 tim_ker_ck 的时钟。
tim_itr[15:0]	输入	内部触发输入总线。这些输入可用于从模式控制器或用作输入时钟（低于 1/4 tim_ker_ck 时钟频率）。
tim_trgo	输出	内部触发输出。该触发信号可触发其他片上外设。
tim_ocref_clr[7:0]	输入	定时器 tim_ocref_clr 输入总线。这些输入可用于清除 tim_ocxref 信号，通常用于硬件逐周期脉宽控制。
tim_pclk	输入	定时器 APB 时钟。
tim_ker_ck	输入	定时器内核时钟
tim_it	输出	全局定时器中断，收集捕获/比较、更新和刹车触发请求。
tim_cc1_dma tim_cc2_dma tim_cc3_dma tim_cc4_dma	输出	定时器捕获/比较 1..4 DMA 请求。
tim_upd_dma	输出	定时器更新 DMA 请求。
tim_trg_dma	输出	定时器触发 DMA 请求。

表 187、*表 188*、*表 189* 和*表 190* 列出了连接到 tim_ti[4:1] 输入复用器的源。

表 187. tim_ti1 输入复用器的互连

tim_ti1 输入	源	
	TIM2	TIM3
tim_ti1_in0	TIM2_CH1	TIM3_CH1
tim_ti1_in1	RCC_LSI	COMP1_OUT
tim_ti1_in2	RCC_LSE	RCC_MCO1
tim_ti1_in3	RTC_WKUP_IT	AFI_TIM2_TI1
tim_ti1_in4	AFI_TIM3_TI1	RCC_HSE_1MHz
tim_ti1_in[15:5]	保留	

表 188. tim_ti2 输入复用器的互连

tim_ti2 输入	源	
	TIM2	TIM3
tim_ti2_in0	TIM2_CH2	TIM3_CH2
tim_ti2_in1	RCC_HSI/1024	RCC_CSI/128
tim_ti2_in2	RCC_CSI/128	RCC_MCO2
tim_ti2_in3	RCC_MCO2	RCC_HSI/1024
tim_ti2_in4	RCC_MCO1	保留
tim_ti2_in[15:5]	保留	

表 189. tim_ti3 输入复用器的互连

tim_ti3 输入	源	
	TIM2	TIM3
tim_ti3_in0	TIM2_CH3	TIM3_CH3
tim_ti3_in[15:1]	保留	

表 190. tim_ti4 输入复用器的互连

tim_ti4 输入	源	
	TIM2	TIM3
tim_ti4_in0	TIM2_CH4	TIM3_CH4
tim_ti4_in1	COMP1_OUT	保留
tim_ti4_in[15:2]	保留	

表 191 列出了连接到 tim_etr 输入复用器的内部源。

表 191. TIMx 内部触发连接

TIMx	TIM2	TIM3
tim_itr0	tim1_trgo	tim1_trgo
tim_itr1	保留	tim2_trgo
tim_itr2	tim3_trgo	保留
tim_itr[11:3]	保留	
tim_itr12	USBSOF	保留
tim_itr[15:13]	保留	

表 192 列出了连接到 tim_etr 输入复用器的内部源。

表 192. tim_etr 输入复用器的互连

<table>
<tr><th rowspan="2">定时器外部触发输入信号</th><th colspan="2">定时器外部触发信号分配</th></tr>
<tr><th>TIM2</th><th>TIM3</th></tr>
<tr><td>tim_etr0</td><td>TIM2_ETR</td><td>TIM3_ETR</td></tr>
<tr><td>tim_etr1</td><td>COMP1_OUT</td><td>COMP1_OUT</td></tr>
<tr><td>tim_etr2</td><td colspan="2">保留</td></tr>
<tr><td>tim_etr3</td><td>RCC_LSE</td><td rowspan="4">保留</td></tr>
<tr><td>tim_etr4</td><td rowspan="4">保留</td></tr>
<tr><td>tim_etr5</td></tr>
<tr><td>tim_etr6</td></tr>
<tr><td>tim_etr7</td></tr>
<tr><td>tim_etr8</td><td>TIM2_ETR</td></tr>
<tr><td>tim_etr9</td><td>TIM3_ETR</td><td>保留</td></tr>
<tr><td>tim_etr[15:10]</td><td colspan="2">保留</td></tr>
</table>

27.4.3 时基单元

可编程定时器的主要模块由一个 16 位/32 位计数器及其相关的自动重装寄存器组成。计数器可递增计数、递减计数或进行递增递减计数。计数器的时钟可通过预分频器进行分频。

计数器、自动重载寄存器和预分频器寄存器可通过软件进行读写。即使在计数器运行时也可执行读写操作。

时基单元包括：

- 计数器寄存器 (TIMx_CNT)
- 预分频器寄存器 (TIMx_PSC)
- 自动重载寄存器 (TIMx_ARR)

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以随时直接写入影子寄存器，也可以在每次发生更新事件 (UEV) 时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位 (ARPE)。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将产生更新事件。该更新事件也可由软件产生。下文将详细介绍各配置的更新事件的产生过程。

计数器由预分频器输出 tim_cnt_ck 提供时钟，仅当 TIMx_CR1 寄存器中的计数器使能位 (CEN) 置 1 时，才会使能计数器（有关计数器使能的更多详细信息，另请参见从模式控制器的相关说明）。

请注意，实际的计数器使能信号 CNT_EN 在 CEN 置 1 的一个时钟周期后被置 1。

预分频器说明

预分频器可对计数器时钟进行分频，分频系数介于 1 和 65536 之间。该预分频器基于 16 位 /32 位寄存器（TIMx_PSC 寄存器）所控制的 16 位计数器。由于该控制寄存器具有缓冲功能，因此预分频器可实现实时更改。而新的预分频比将在下一更新事件发生时被采用。

图 252 和 *图 253* 以一些示例说明在预分频比实时变化时计数器的行为：

图 252. 预分频器分频由 1 变为 2 时的计数器时序图

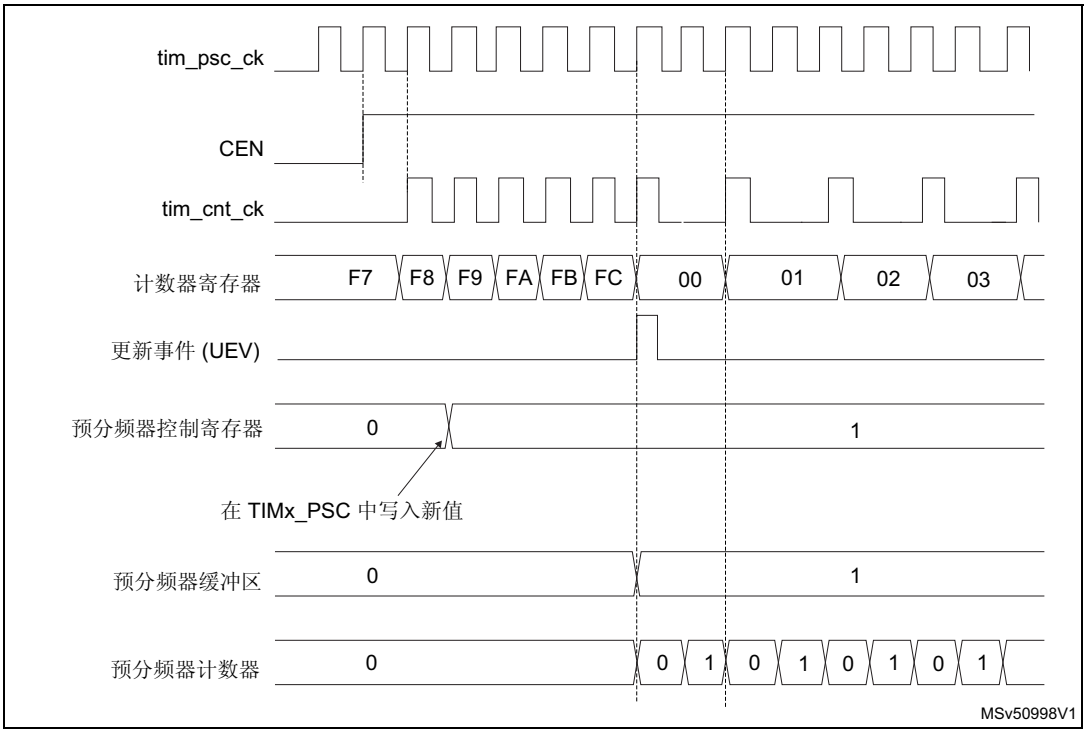
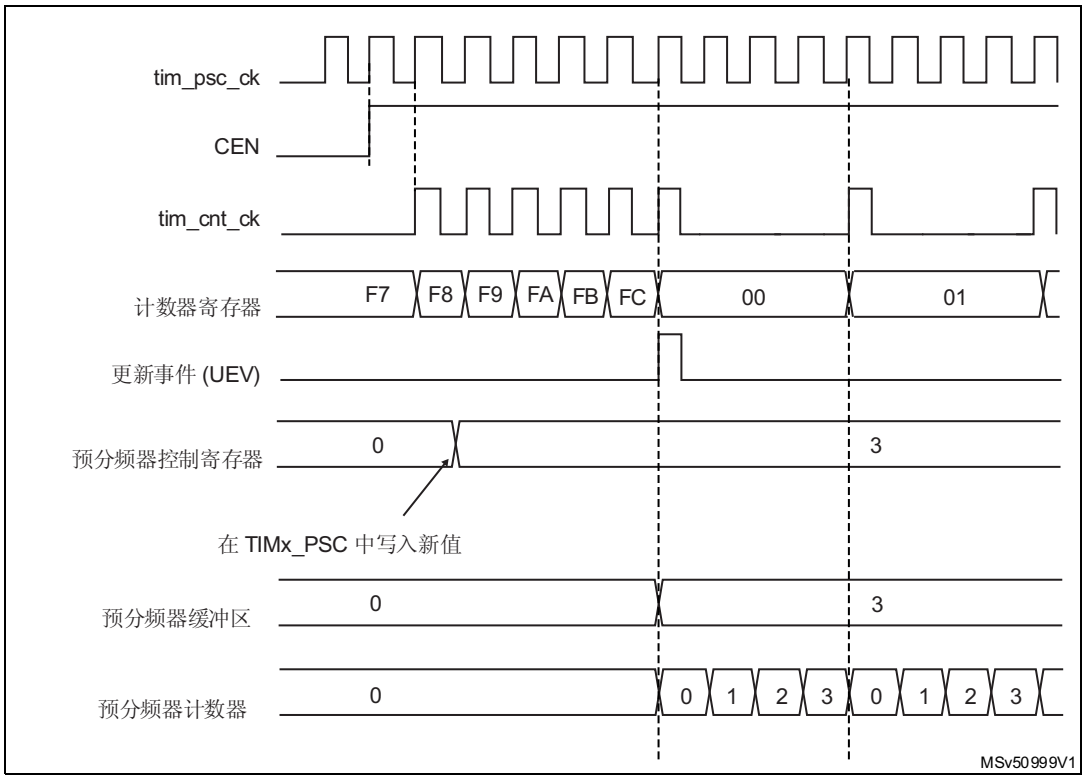


图 253. 预分频器分频由 1 变为 4 时的计数器时序图



27.4.4 计数器模式

递增计数模式

在递增计数模式下，计数器从 0 计数到自动重载值（TIMx_ARR 寄存器的内容），然后重新从 0 开始计数并生成计数器上溢事件。

每次发生计数器上溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变）。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。利用这一特性，在定时器的输入捕获操作时可以通过对 UG 位置 1 让计数器清零而不至于触发更新中断，也就避免了在捕获中断里触发更新中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）
- 使用预装载值 (TIMx_ARR) 更新自动重载影子寄存器

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

图 254. 计数器时序图，1 分频内部时钟

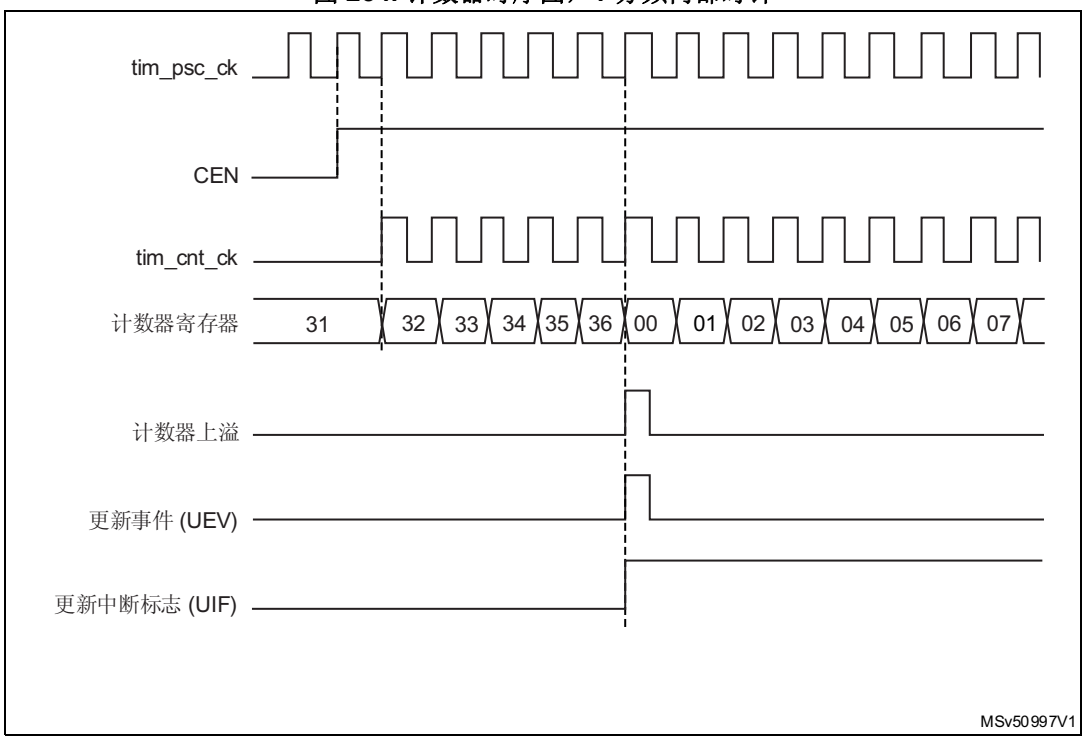


图 255. 计数器时序图，2 分频内部时钟

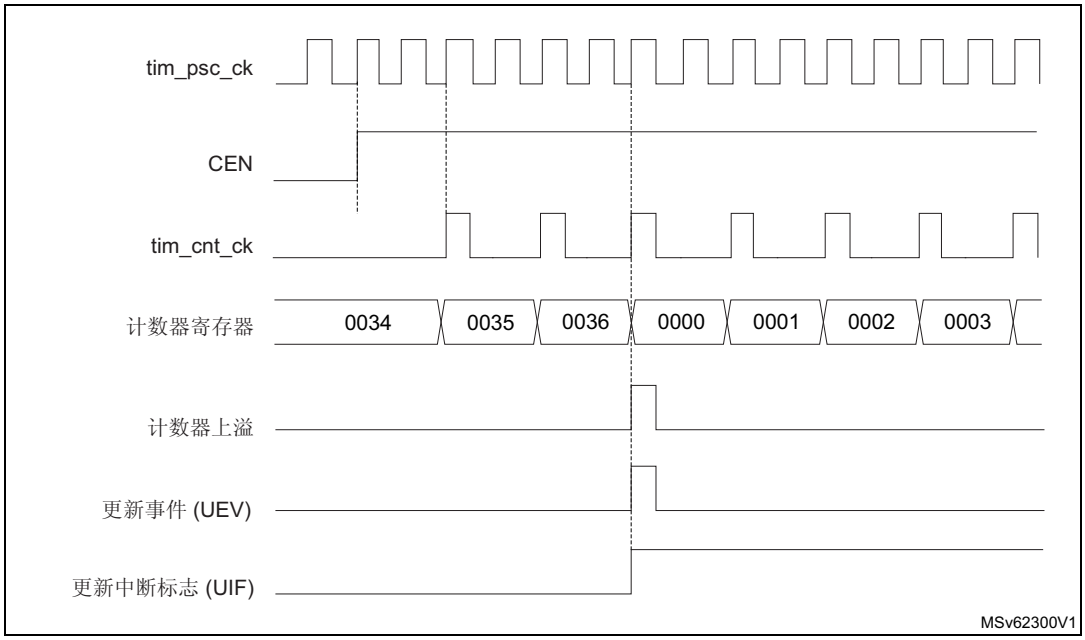


图 256. 计数器时序图，4 分频内部时钟

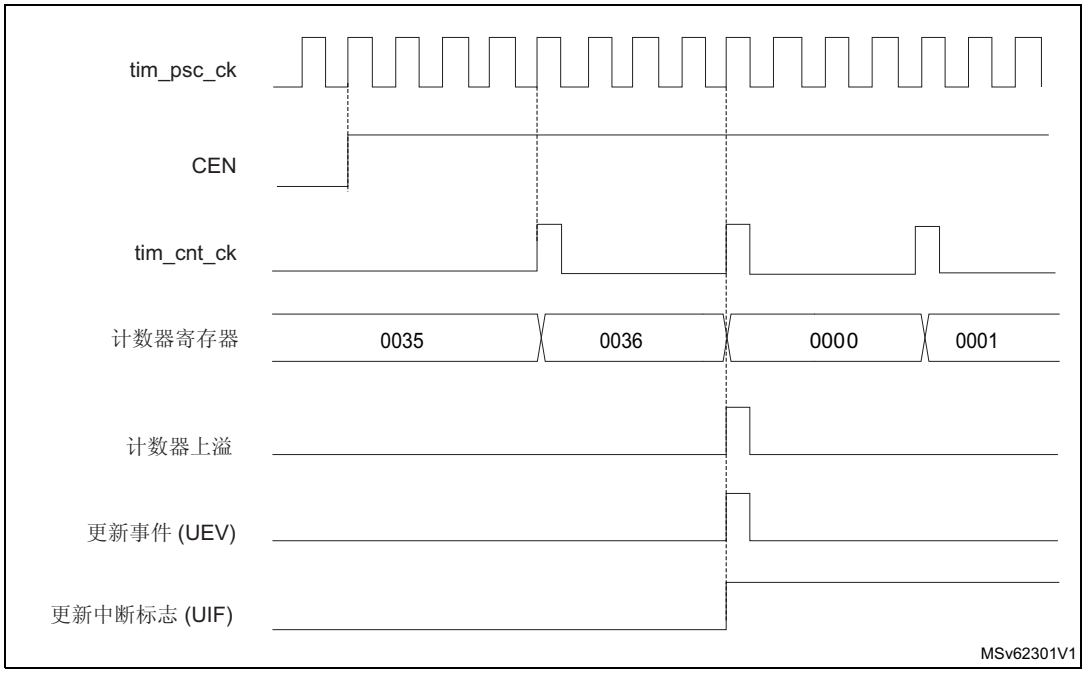


图 257. 计数器时序图，N 分频内部时钟

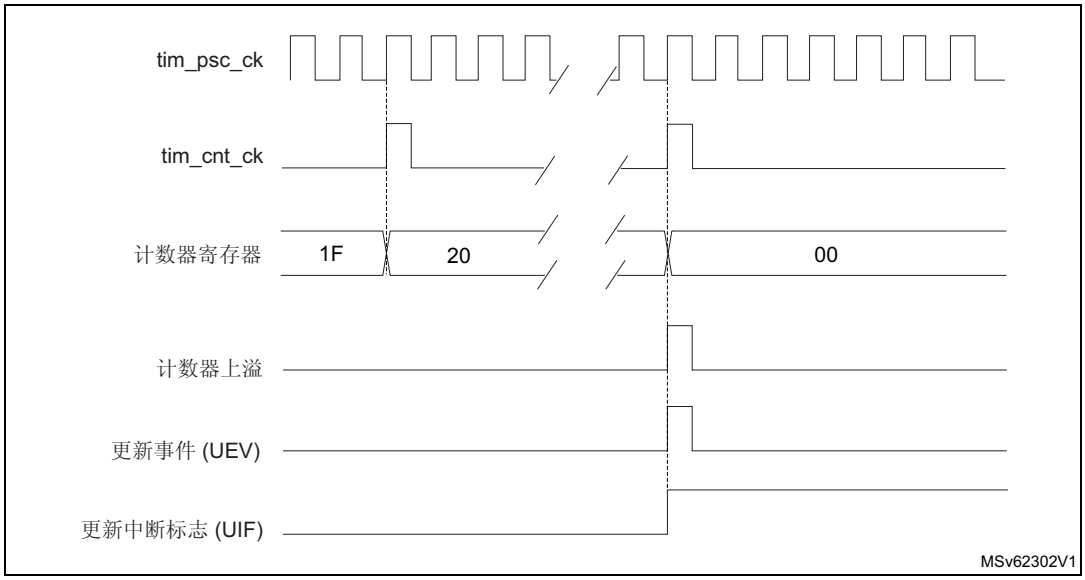


图 258. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载）

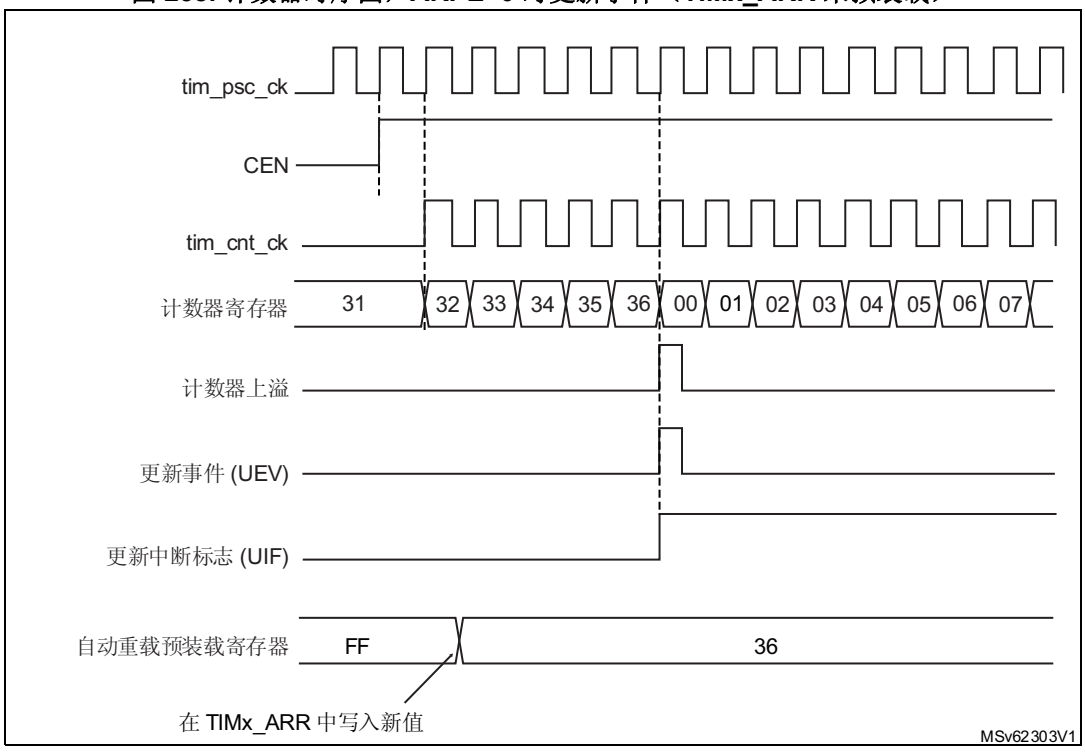
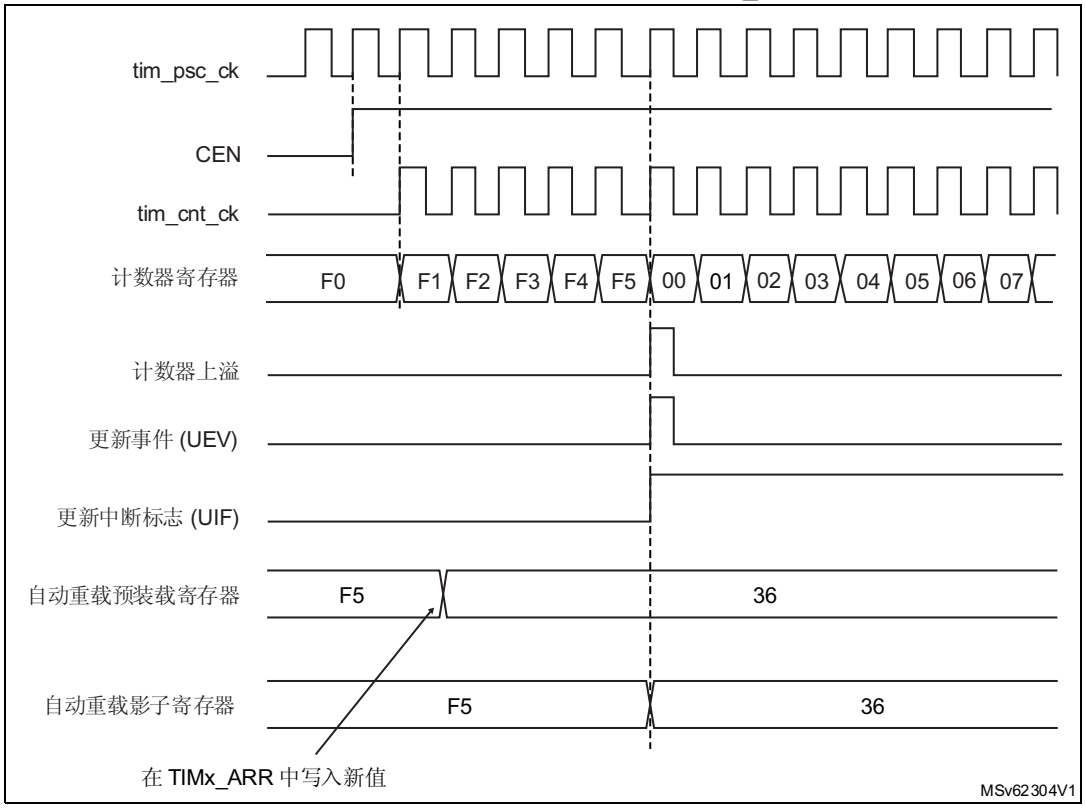


图 259. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 已预装载）



递减计数模式

在递减计数模式下，计数器从自动重载值（TIMx_ARR 寄存器的内容）开始递减计数到 0，然后重新从自动重载值开始计数并生成计数器下溢事件。

每次发生计数器下溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器会从当前自动重载值开始重新计数，而预分频器计数器则从 0 开始重新计数（但预分频比保持不变）。

此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。利用这一特性，在定时器的输入捕获操作时可以通过对 UG 位置 1 让计数器清零而不至于触发更新中断，也就避免了在捕获中断里触发更新中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 自动重载影子寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，ARR 寄存器更新在计数器重载之前被更新，因此下一个周期就是预期的值。

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

图 260. 计数器时序图，1 分频内部时钟

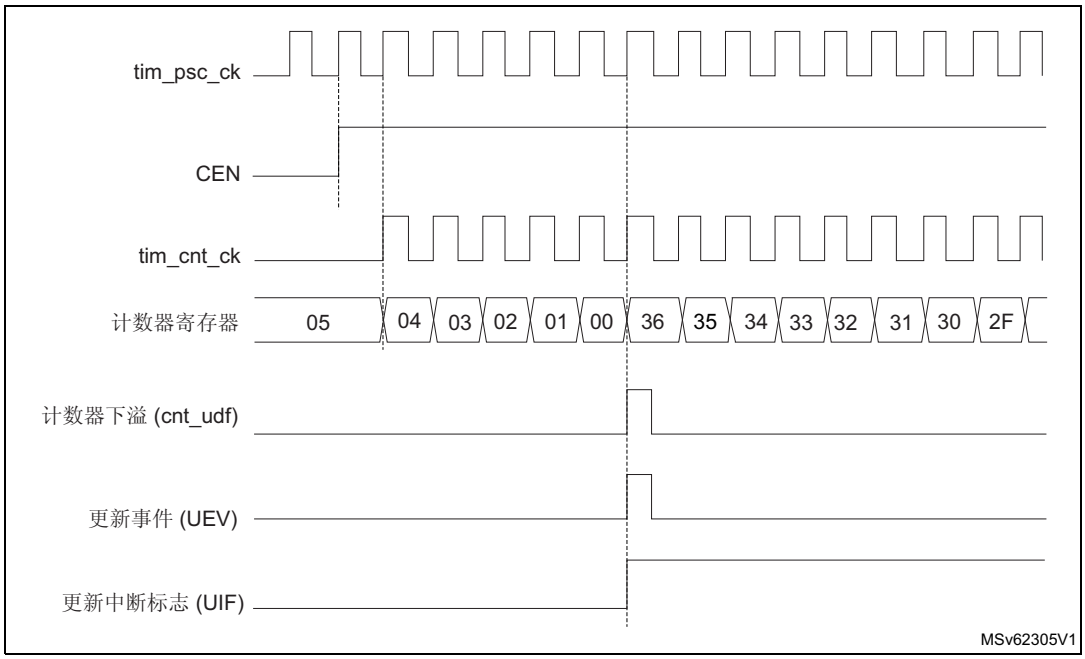


图 261. 计数器时序图，2 分频内部时钟

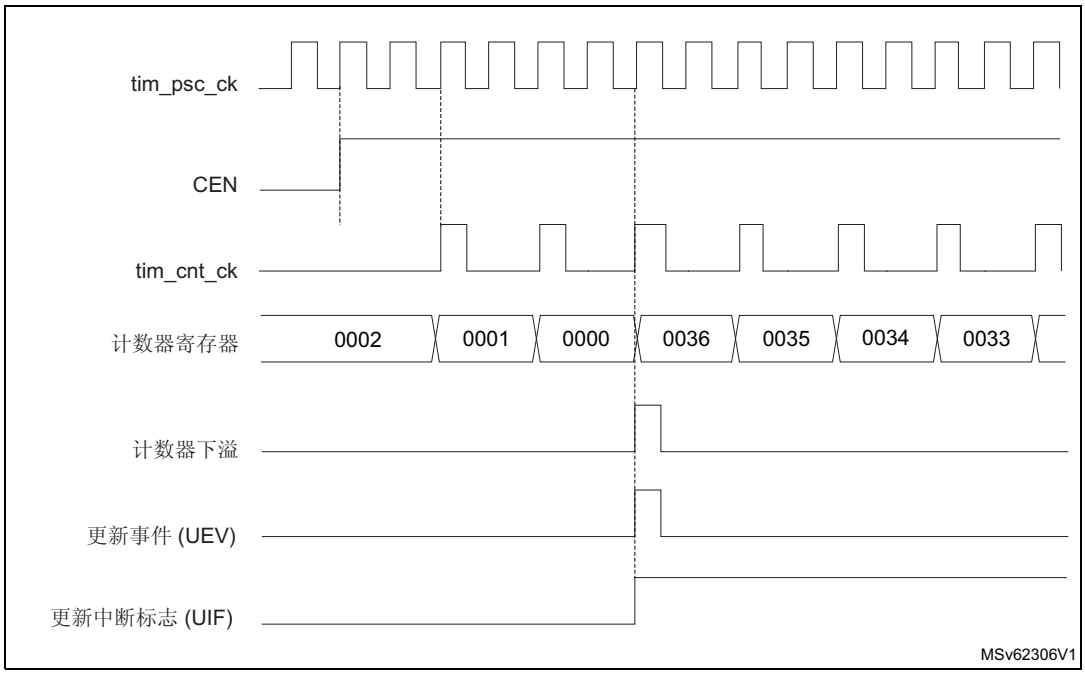


图 262. 计数器时序图，4 分频内部时钟

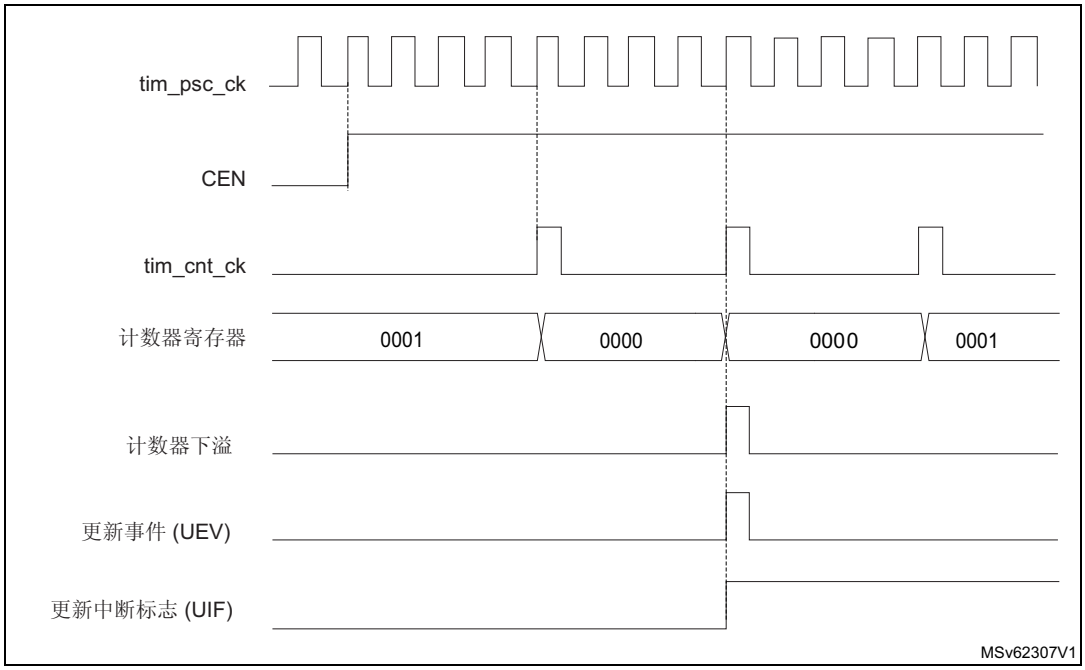


图 263. 计数器时序图，N 分频内部时钟

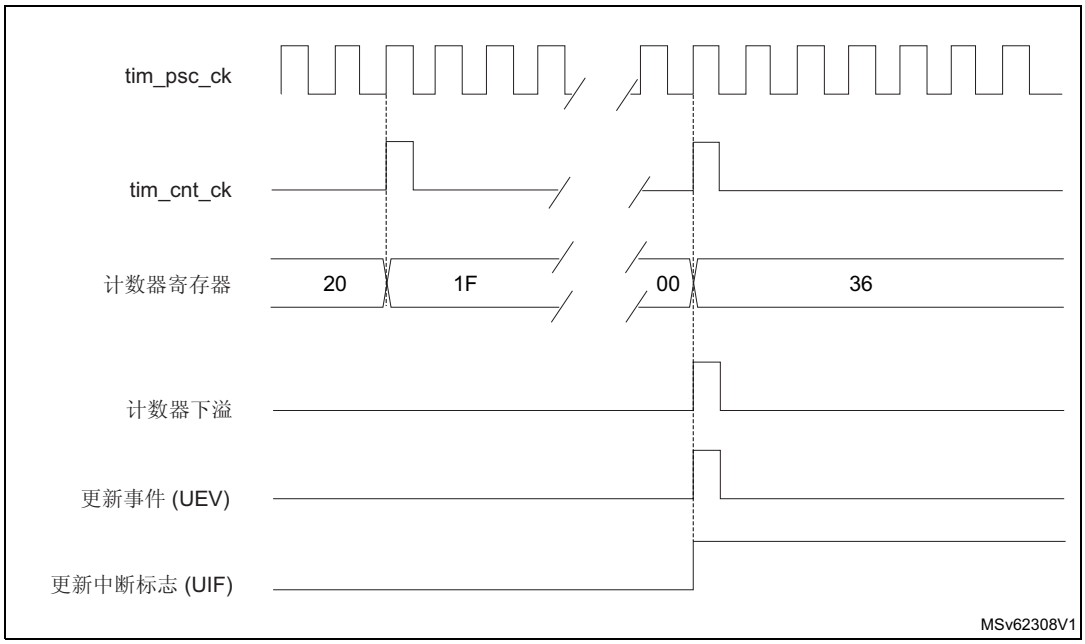
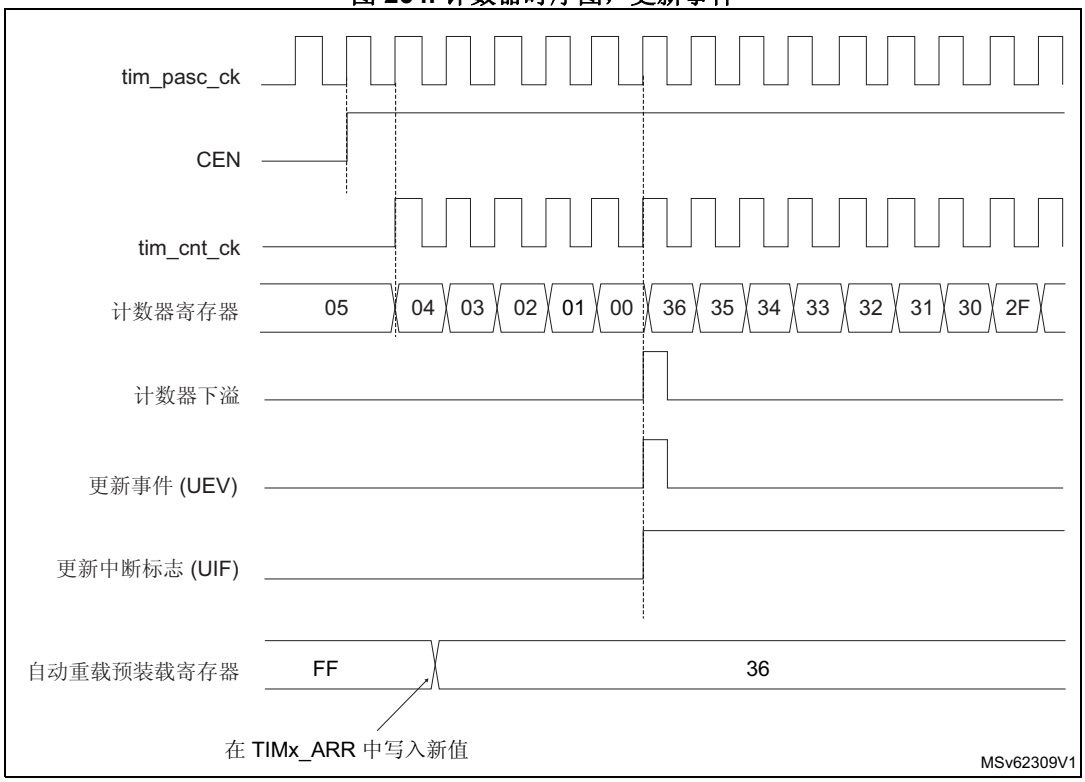


图 264. 计数器时序图，更新事件



中心对齐模式（递增/递减计数）

在中心对齐模式下，计数器从 0 开始计数到自动重载值（TIMx_ARR 寄存器的内容）– 1，生成计数器上溢事件；然后从自动重载值开始向下计数到 1 并生成计数器下溢事件。之后从 0 开始重新计数。

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时，中心对齐模式有效。将通道配置为输出模式时，其输出比较中断标志将在以下模式下置 1，即：计数器递减计数（中心对齐模式 1，CMS =“01”）、计数器递增计数（中心对齐模式 2，CMS =“10”）以及计数器递增/递减计数（中心对齐模式 3，CMS =“11”）。

此模式下无法写入方向位（TIMx_CR1 寄存器中的 DIR 位）。而是由硬件更新并指示当前计数器方向。

每次发生计数器上溢和下溢时都会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。这种情况下，计数器以及预分频器计数器将重新从 0 开始计数。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器仍会根据当前自动重载值进行递增和递减计数。

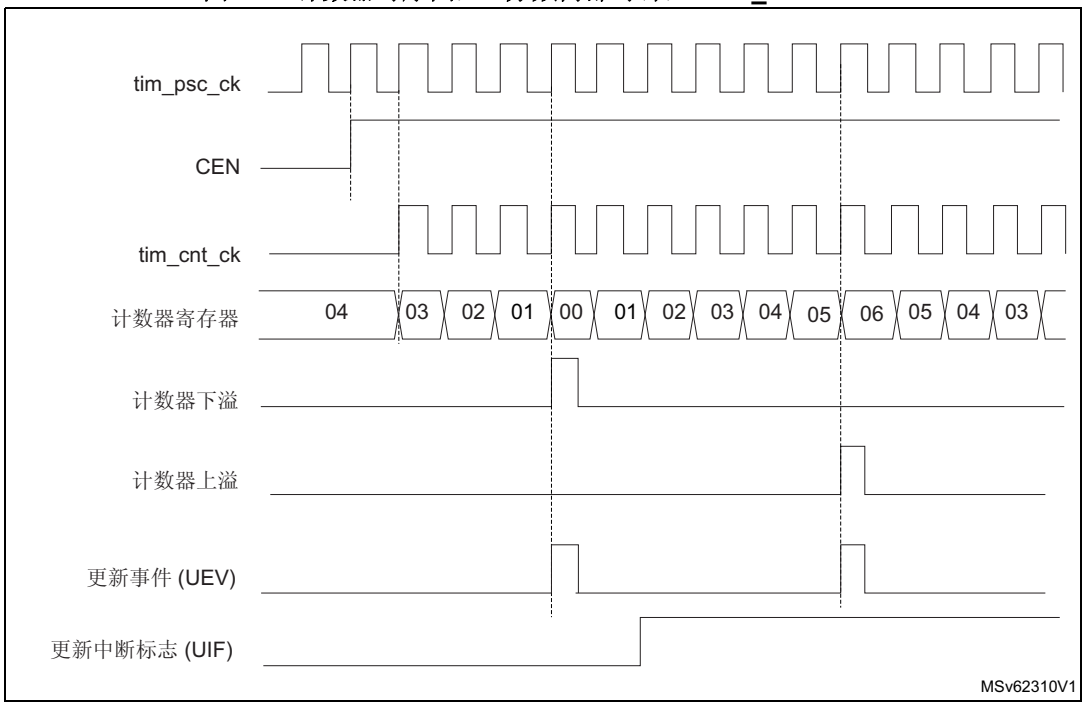
此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。利用这一特性，在定时器的输入捕获操作时可以通过对 UG 位置 1 让计数器清零而不至于触发更新中断，也就避免了在捕获中断里触发更新中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 自动重载影子寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，如果更新操作是由计数器上溢触发的，则 ARR 寄存器在计数器重载之前更新，因此，下一个计数周期就是我们所希望的新的周期长度（计数器被重载新的值）。

以下各图以一些示例说明不同时钟频率下计数器的行为。

图 265. 计数器时序图，1 分频内部时钟，TIMx_ARR=0x6



1. 此处使用中心对齐模式 1（有关详细信息，请参见 *第 891 页的第 27.5.1 节：TIMx 控制寄存器 1 (TIMx_CR1)（x = 2、3）*）。

图 266. 计数器时序图，2 分频内部时钟

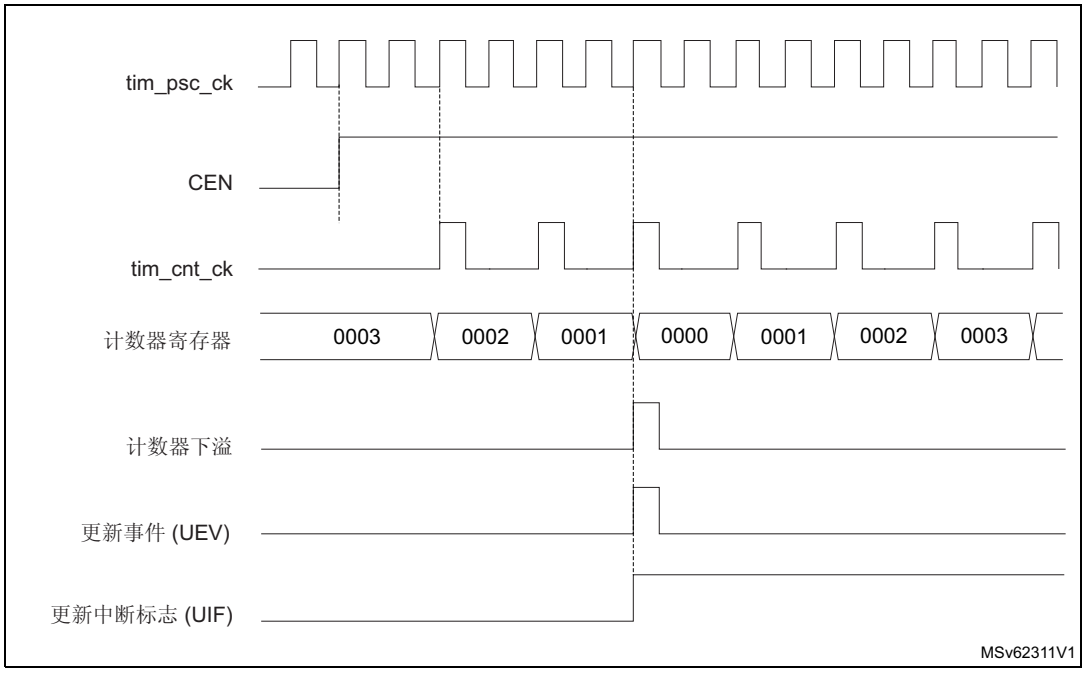
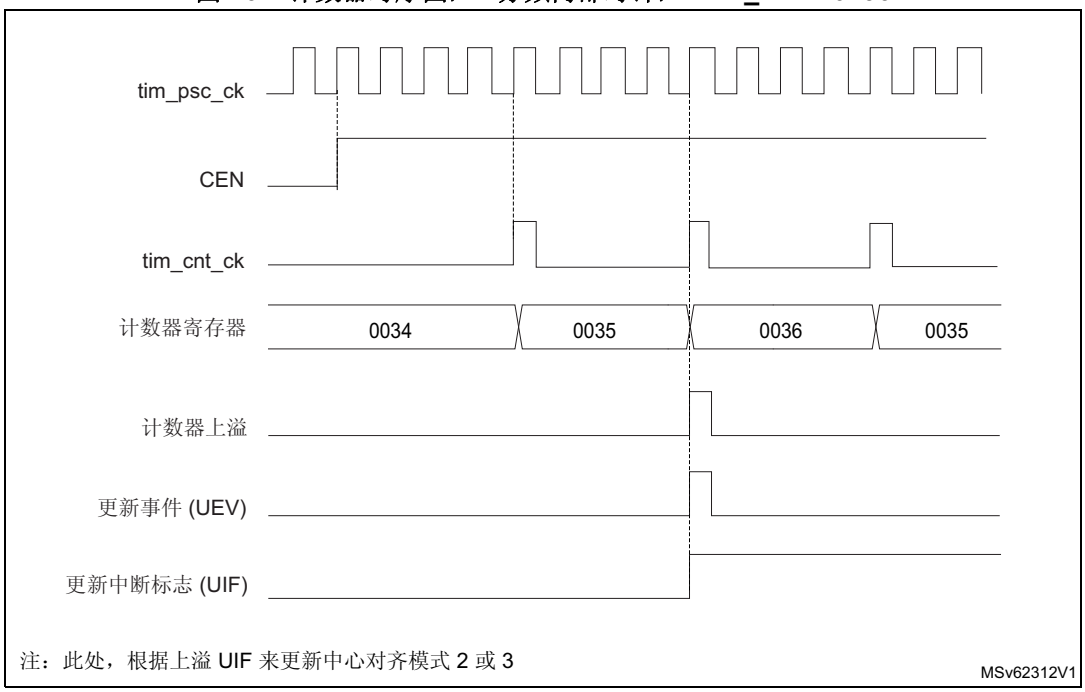


图 267. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36



1. 中心对齐模式 2 或模式 3 与上溢 UIF 结合使用。

图 268. 计数器时序图，N 分频内部时钟

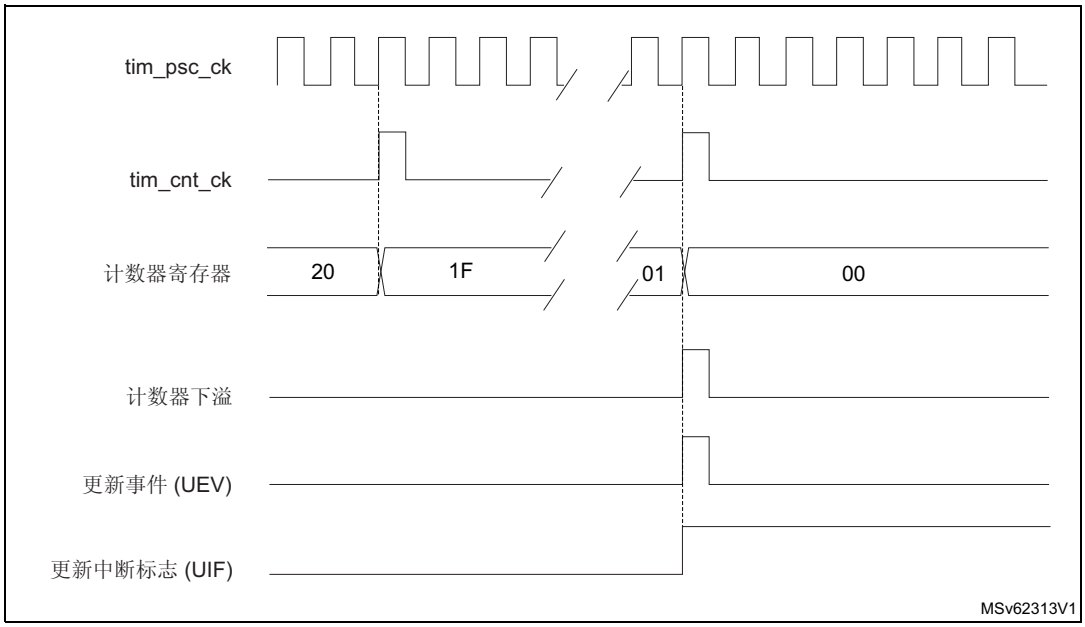


图 269. 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

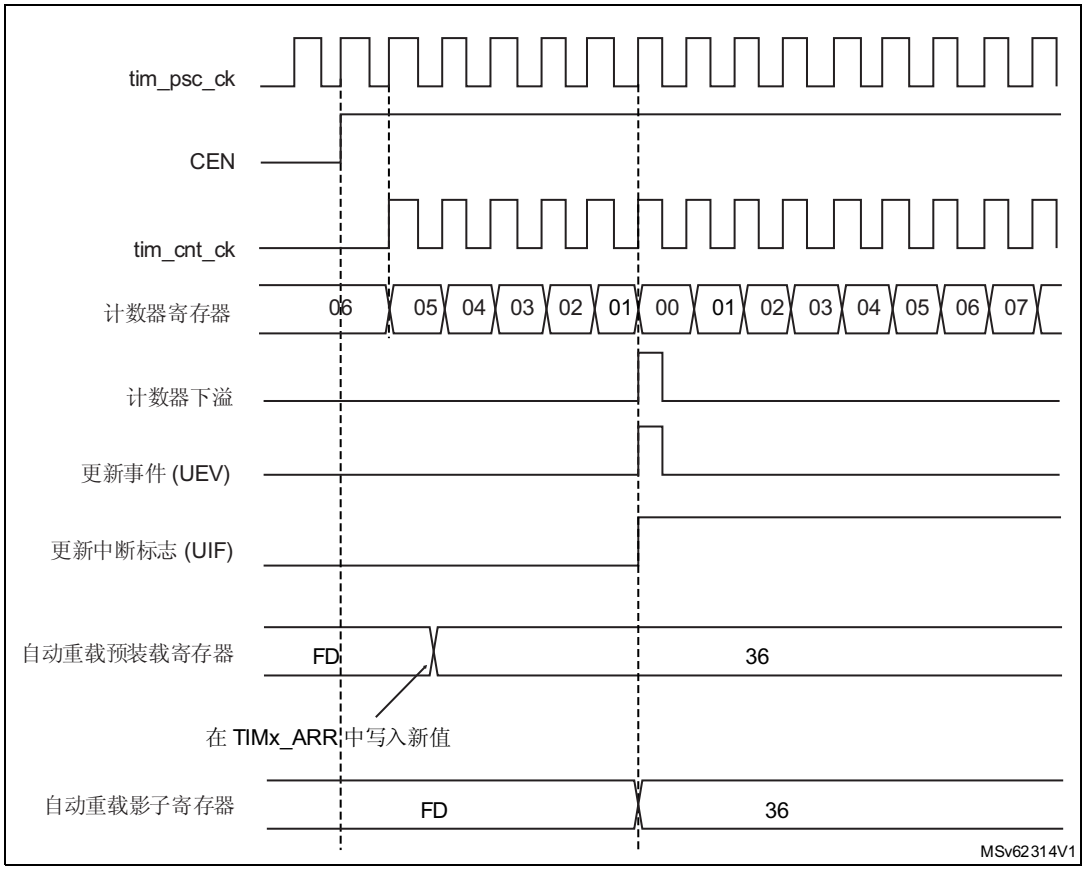
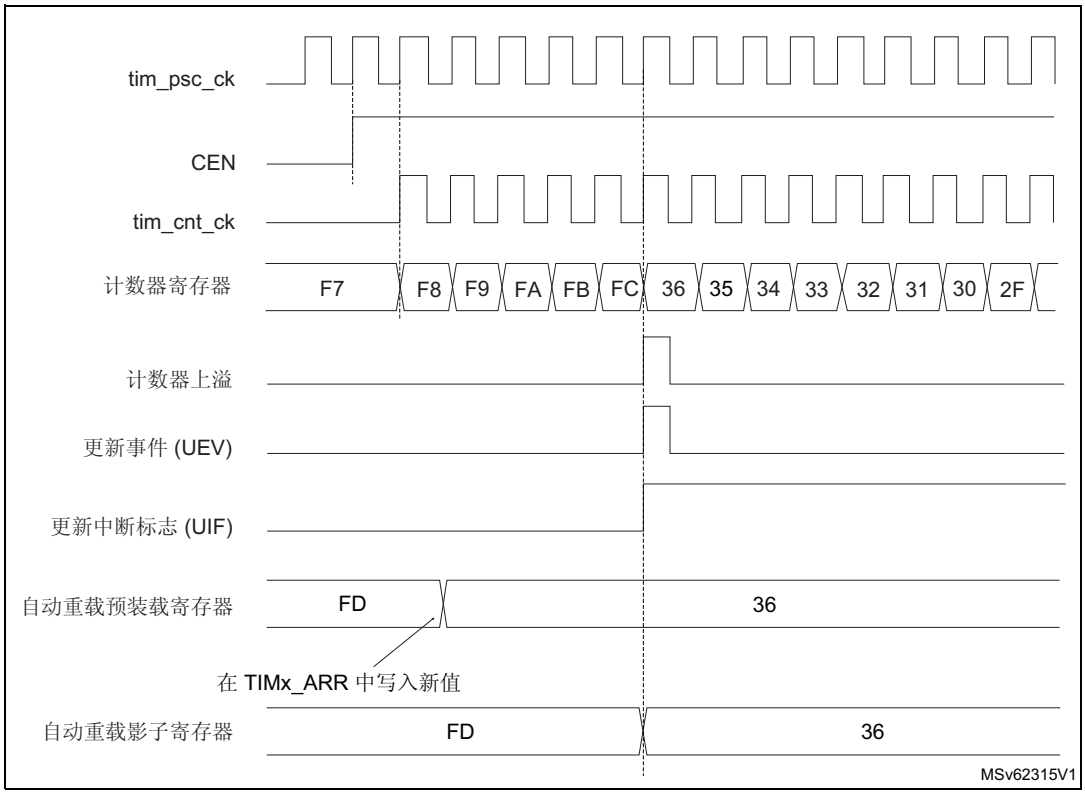


图 270. 计数器时序图，ARPE=1 时的更新事件（计数器上溢）



27.4.5 时钟选择

计数器时钟可由下列时钟源提供：

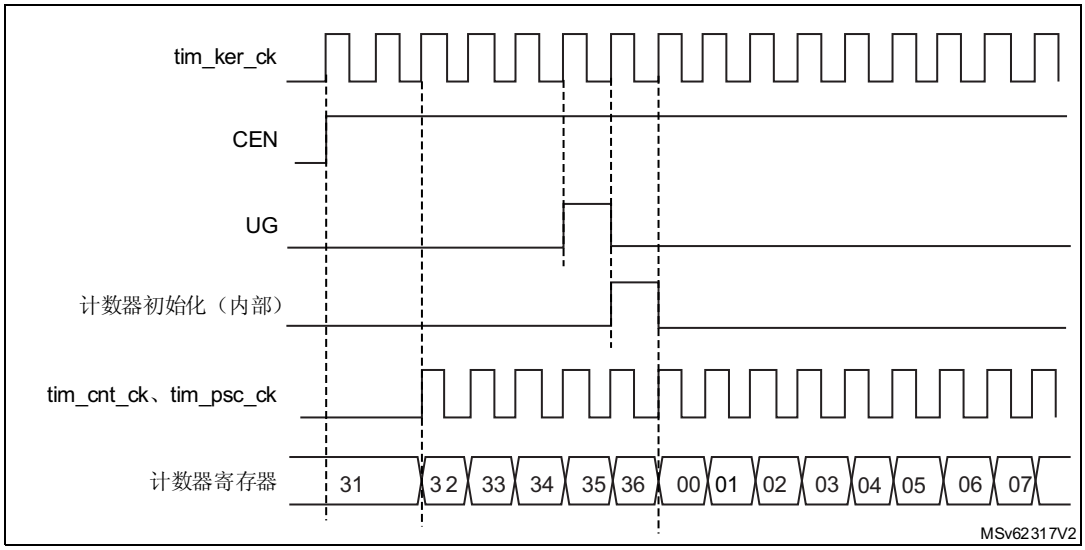
- 内部时钟 (tim_ker_ck)
- 外部时钟模式 1：外部输入引脚（tim_ti1 或 tim_ti2）
- 外部时钟模式 2：外部触发输入 (tim_etr_in)
- 内部触发输入 (tim_itr)：使用一个定时器作为另一个定时器的预分频器，例如可以将定时器 1 配置为定时器 2 的预分频器。更多详细信息，请参见*第 884 页的将一个定时器用作另一个定时器的预分频器*。

内部时钟源 (tim_ker_ck)

如果禁止从模式控制器（TIMx_SMCR 寄存器中 SMS=000），则 CEN 位、DIR 位（TIMx_CR1 寄存器中）和 UG 位（TIMx_EGR 寄存器中）为实际控制位，并且只能通过软件进行更改（UG 除外，仍自动清零）。当对 CEN 位写入 1 时，预分频器的时钟就由内部时钟 tim_ker_ck 提供。

图 271 显示了正常模式下控制电路与递增计数器的行为（没有预分频的情况下）。

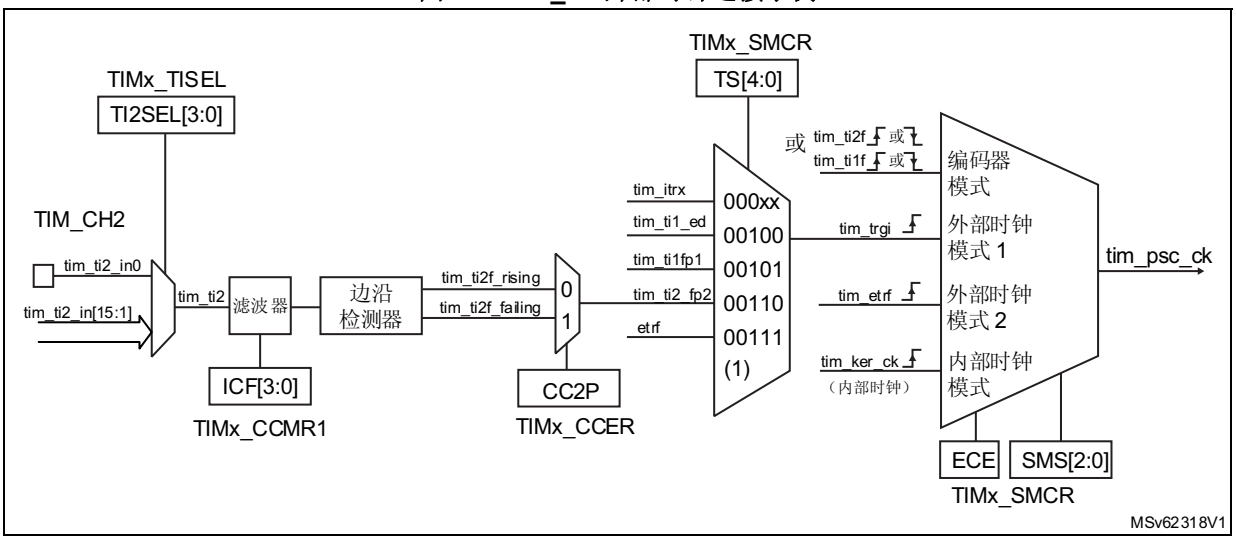
图 271. 正常模式下的控制电路，1 分频内部时钟



外部时钟源模式 1

当 TIMx_SMCR 寄存器中的 SMS=111 时，可选择此模式。计数器可在选定的输入信号上出现上升沿或下降沿时计数。

图 272. tim_ti2 外部时钟连接示例



1. 从 01000 到 11111 的代码：tim_itr[15:0]。

例如，要使递增计数器在 tim_ti2 输入出现上升沿时计数，请执行以下步骤：

1. 使用 TIMx_TISEL 寄存器中的 TI2SEL[3:0] 位选择正确的 tim_ti2_in[15:0] 源（内部或外部）。
2. 通过在 TIMx_CCMR1 寄存器中写入 CC2S = “01” 来配置通道 2，使其能够检测 tim_ti2 输入的上升沿。
3. 通过在 TIMx_CCMR1 寄存器中写入 IC2F[3:0] 位来配置输入滤波带宽（如果不需要任何滤波器，请保持 IC2F=0000）。

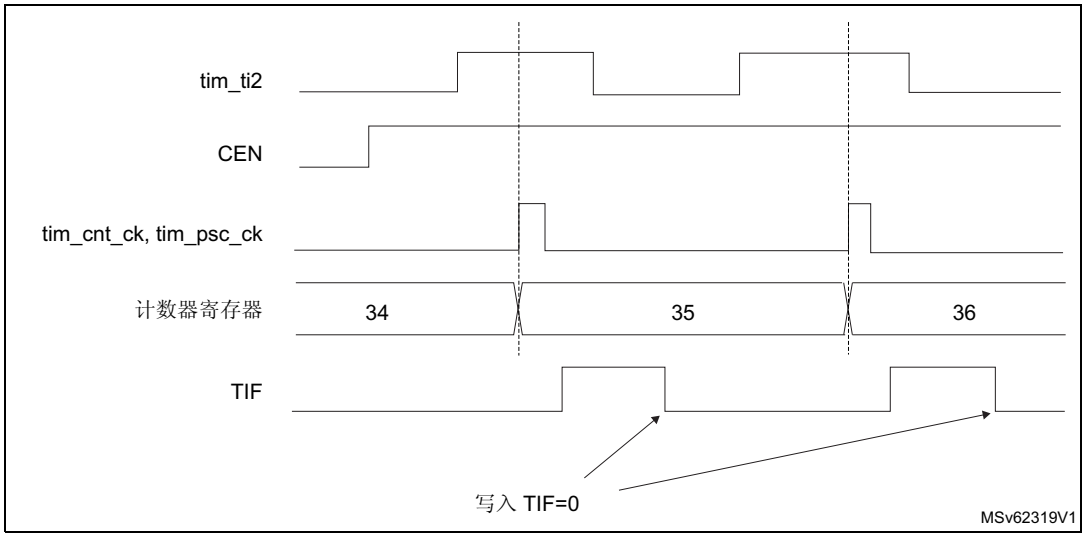
注意： *由于捕获预分频器不用于触发操作，因此无需对其进行配置。*

4. 通过在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=0 来选择上升沿极性。
5. 通过在 TIMx_SMCR 寄存器中写入 SMS=111，使定时器在外部时钟模式 1 下工作。
6. 通过在 TIMx_SMCR 寄存器中写入 TS=00110 来选择 tim_ti2 作为输入源。
7. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

当 tim_ti2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

tim_ti2 的上升沿与实际计数器时钟之间的延迟是由于 tim_ti2 输入的重新同步电路引起的。

图 273. 外部时钟模式 1 下的控制电路



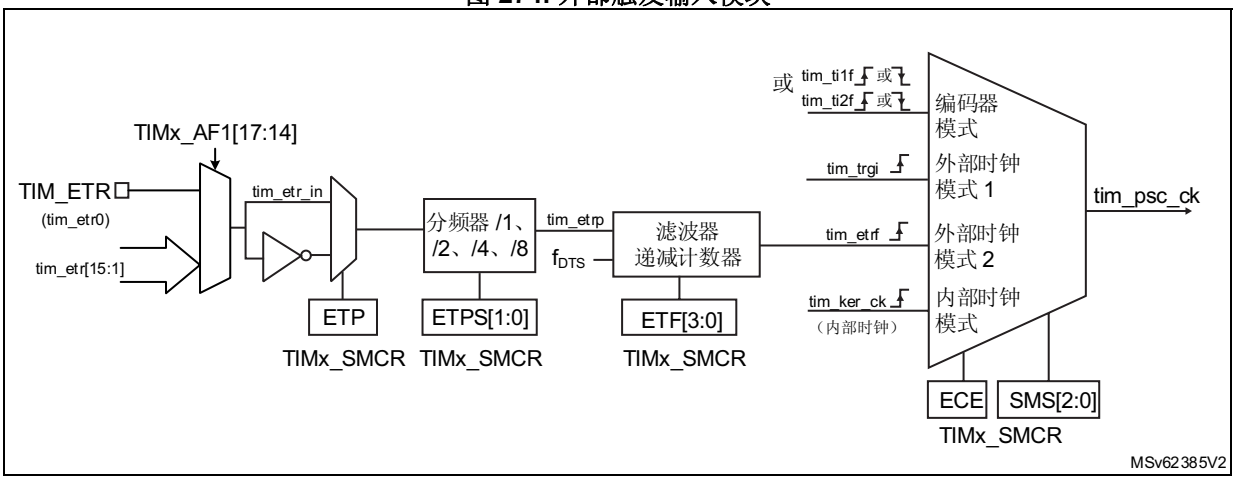
外部时钟源模式 2

通过在 TIMx_SMCR 寄存器中写入 ECE=1 可选择此模式。

计数器可在外部触发输入 tim_etr_in 出现上升沿或下降沿时计数。

图 274 简要介绍了外部触发输入模块。

图 274. 外部触发输入模块



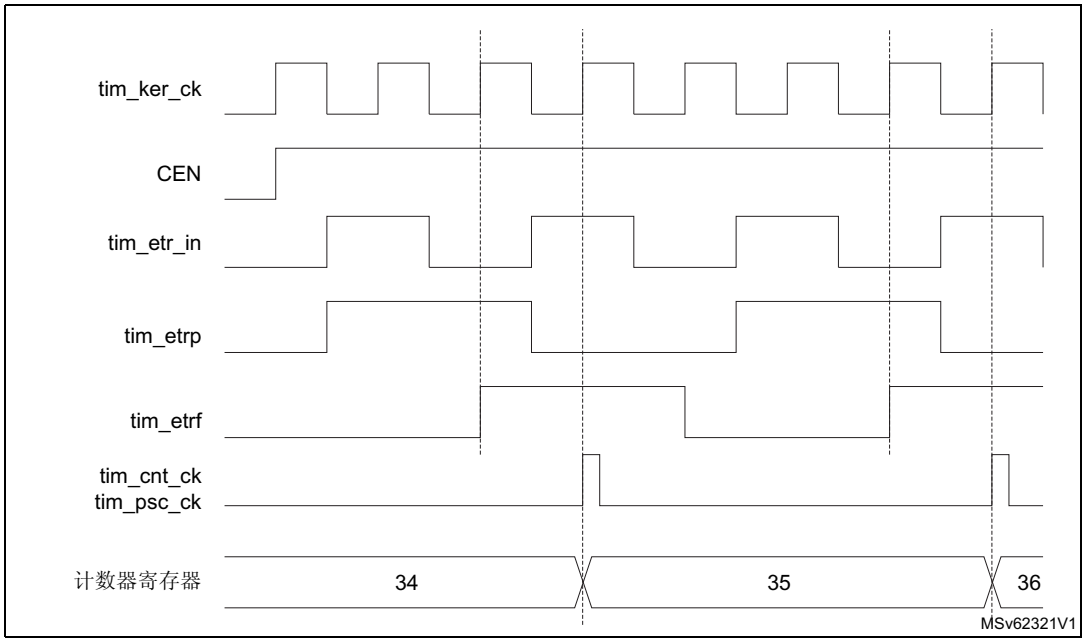
例如，要使递增计数器在 tim_etr_in 每出现 2 个上升沿时计数，请执行以下步骤：

1. 使用 TIMx_AF1 寄存器中的 ETRSEL[3:0] 位选择正确的 tim_etr_in 源（内部或外部）。
2. 由于此例中不需滤波器，因此在 TIMx_SMCR 寄存器中写入 ETF[3:0]=0000。
3. 通过在 TIMx_SMCR 寄存器中写入 ETPS[1:0]=01 来设置预分频器。
4. 通过在 TIMx_SMCR 寄存器中写入 ETP=0 来选择 tim_etr_in 的上升沿检测。
5. 通过在 TIMx_SMCR 寄存器中写入 ECE=1 来使能外部时钟模式 2。
6. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

tim_etr_in 每出现 2 个上升沿，计数器计数一次。

tim_etr_in 的上升沿与实际计数器时钟之间的延迟是由于 tim_etrp 信号的重新同步电路引起的。因此，计数器能够正确捕获的最大频率不超过 TIMxCLK 频率的 ¼。当 ETRP 信号较快时，用户必须通过适当的 ETPS 预分频器设置对外部信号进行分频。

图 275. 外部时钟模式 2 下的控制电路



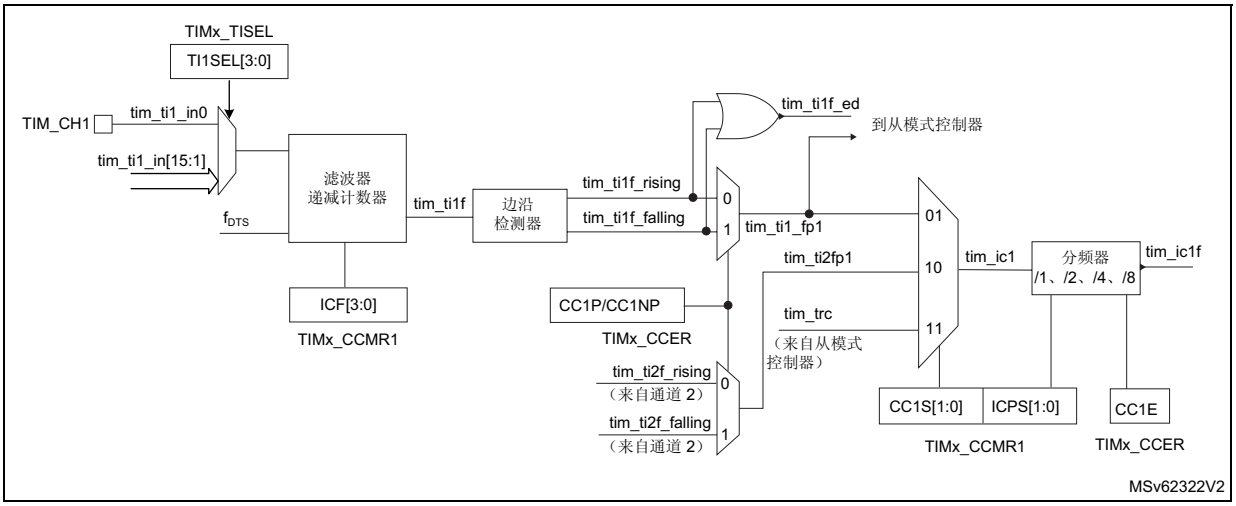
27.4.6 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入单元（数字滤波、多路复用和预分频器）和一个输出单元（比较器和输出控制）构建而成。

下图简要介绍了一路捕获/比较通道。

输入阶段对相应的 tim_tix 输入进行采样，生成一个滤波后的信号 tim_tixf。然后，带有极性选择功能的边沿检测器生成一个信号 (tim_tixfpy)，该信号可用作从模式控制器的触发输入，也可用作捕获命令。该信号先进行预分频 (ICxPS)，而后再进入捕获寄存器。

图 276. 捕获/比较通道（示例：通道 1 输入阶段）



输出阶段生成一个中间波形作为基准：tim_ocxref（高电平有效）。链的末端决定最终输出信号的极性。

图 277. 捕获/比较通道 1 主电路

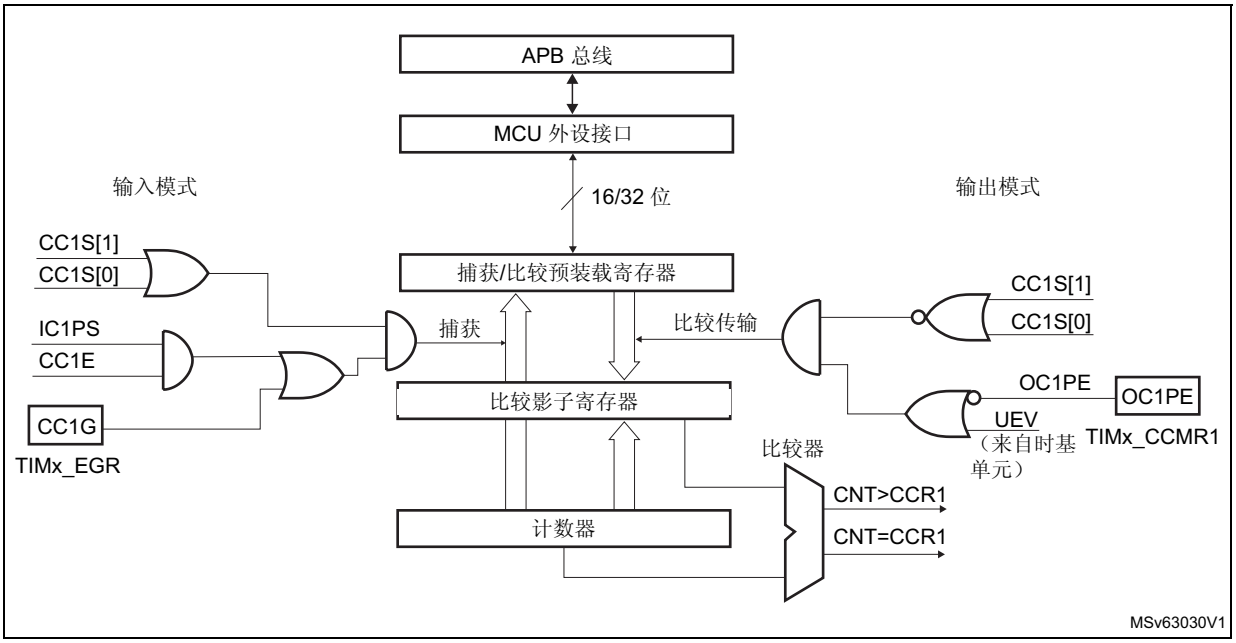
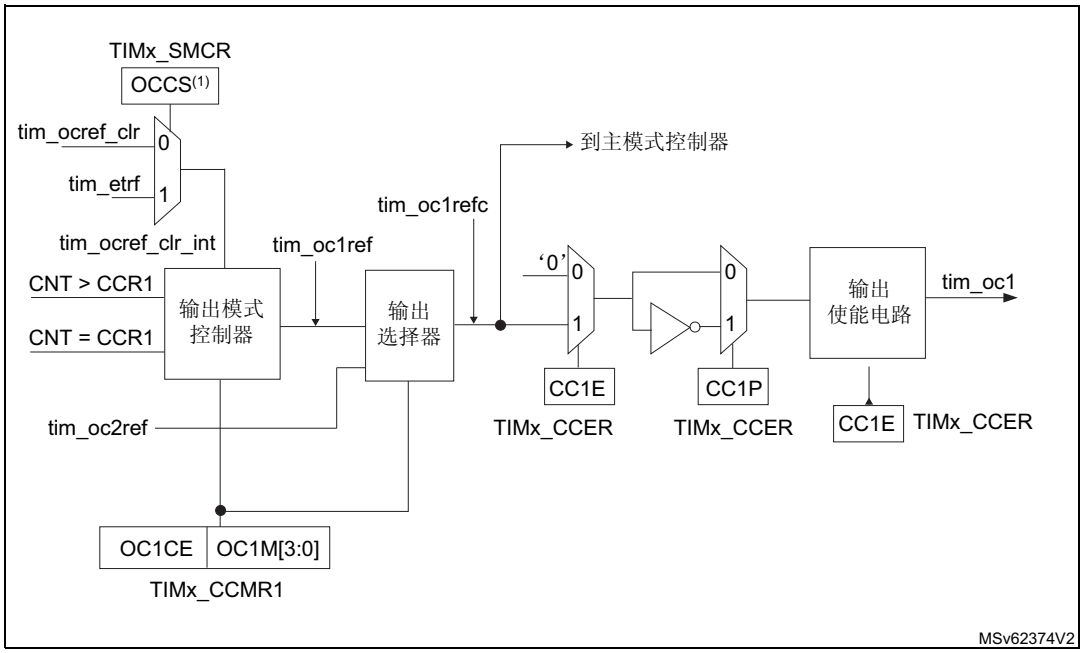


图 278. 捕获/比较通道的输出阶段（通道 1、通道 2、通道 3 和通道 4）



1. 仅一些实例提供。如果不提供，则 tim_etrf 直接连接到 tim_ocref_clr_int。

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。读写访问的始终是预装载寄存器。

在捕获模式下，捕获实际发生在影子寄存器中，然后将影子寄存器的内容复制到预装载寄存器中。

在比较模式下，预装载寄存器的内容将复制到影子寄存器中，然后将影子寄存器的内容与计数器进行比较。

27.4.7 输入捕获模式

在输入捕获模式下，当相应的 ICx 信号检测到跳变沿后，将使用捕获/比较寄存器 (TIMx_CCRx) 来锁存计数器的值。发生捕获事件时，会将相应的 CCXIF 标志（TIMx_SR 寄存器）置 1，并可发送中断或 DMA 请求（如果已使能）。如果发生捕获事件时 CCxIF 标志已处于高位，则会将过捕获标志 CCxOF（TIMx_SR 寄存器）置 1。可通过软件向 CCxIF 写入 0 使其清零，或读取存储在 TIMx_CCRx 寄存器中的已捕获数据。向 CCxOF 写入 0 后会将其清零。

以下示例说明了如何在 tim_ti1 输入出现上升沿时将计数器的值捕获到 TIMx_CCR1 中。具体操作步骤如下：

1. 使用 TIMx_TISEL 寄存器中的 TI1SEL[3:0] 位选择正确的 tim_tix_in[15:0] 源（内部或外部）。
2. 选择有效输入：TIMx_CCR1 必须关联到 tim_ti1 输入，因此向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01。只要 CC1S 不等于 00，就会将通道配置为输入模式，并且 TIMx_CCR1 寄存器将处于只读状态。
3. 根据关联到定时器的信号，对所需输入滤波参数进行编程（如果输入为 tim_tix 之一，则对 TIMx_CCMRx 寄存器中的 ICxF 位进行编程）。假设输入信号发生翻转时，信号需要 5 个内部时钟周期才能稳定，则必须设置滤波时间大于 5 个内部时钟周期。若在 tim_ti1 上连续 8 次检测到新电平采样值（以 f_{DTS} 频率采样）判定沿跳变合法，则向 TIMx_CCMR1 寄存器中的 IC1F 位写入 0011。

4. 通过向 TIMx_CCER 寄存器中的 CC1P 位和 CC1NP 位写入 000，选择 tim_ti1 通道的有效跳变沿（本例中为上升沿）。
5. 对输入预分频器进行编程。在本例中，我们希望每次有效转换时都执行捕获操作，因此需要禁止预分频器（向 TIMx_CCMR1 寄存器中的 IC1PS 位写入 00）。
6. 通过将 TIMx_CCER 寄存器中的 CC1E 位置 1，允许将计数器的值捕获到捕获寄存器中。
7. 如果需要，可通过将 TIMx_DIER 寄存器中的 CC1IE 位置 1 来使能相关中断请求，并且/或者通过将该寄存器中的 CC1DE 位置 1 来使能 DMA 请求。

发生输入捕获时：

- 发生有效跳变沿时，TIMx_CCR1 寄存器会获取计数器的值。
- 将 CC1IF 标志置 1（中断标志）。如果至少发生了两次连续捕获，但 CC1IF 标志未被清零，这样 CC1OF 捕获溢出标志会被置 1。
- 根据 CC1IE 位生成中断。
- 根据 CC1DE 位生成 DMA 请求。

要处理过捕获，建议在读取捕获溢出标志之前读取数据。这样可避免丢失在读取捕获溢出标志之后与读取数据之前可能出现的过捕获信息。

注意： *通过软件将 TIMx_EGR 寄存器中的相应 CCxG 位置 1 可生成 IC 中断和/或 DMA 请求。*

27.4.8 PWM 输入模式

该模式用于测量连接到单个 tim_tix 输入的 PWM 信号的周期和占空比：

- TIMx_CCR1 寄存器保存周期值（相邻两个上升沿之间的时间间隔）
- TIM_CCR2 寄存器保存脉宽（相邻的上升沿与下降沿之间的时间间隔）

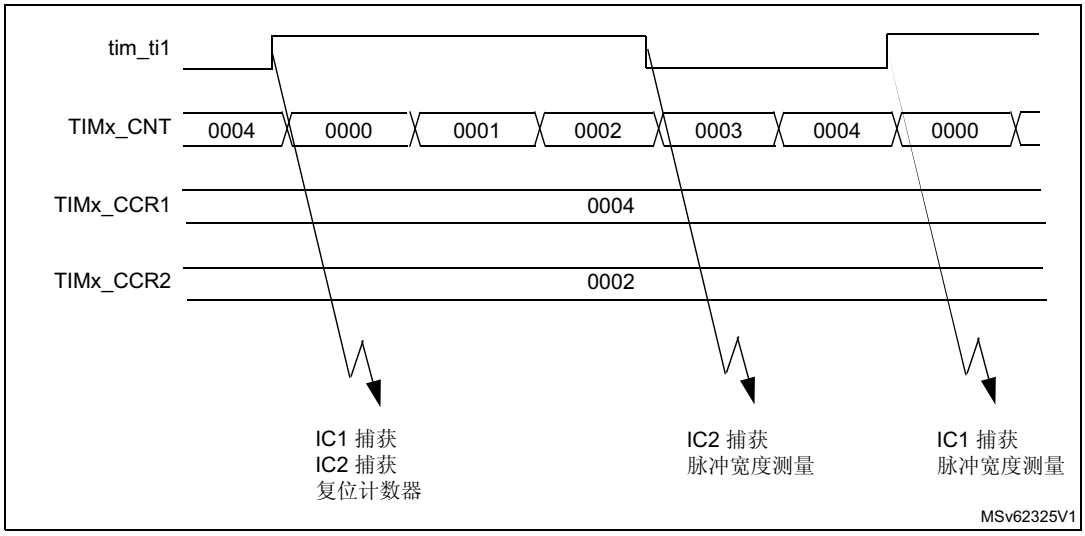
此模式是输入捕获模式的一个特例。设置步骤与之类似，但存在以下区别：

- 两个 ICx 信号被映射至同一个 tim_tix 输入。
- 这两个 ICx 信号在边沿处有效，但极性相反。
- 选择两个 TIxFP 信号之一作为触发输入，并将从模式控制器配置为复位模式。

应用于 tim_ti1 的 PWM 信号的周期和脉宽可通过以下步骤测量：

1. 使用 TIMx_TISEL 寄存器中的 TI1SEL[3:0] 位选择正确的 tim_tix_in[15:0] 源（内部或外部）。
2. 选择 TIMx_CCR1 的有效输入：向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01（选择 tim_ti1）。
3. 选择 tim_ti1fp1 的有效极性（同时用于 TIMx_CCR1 中的捕获和计数器清零）：向 CC1P 位和 CC1NP 位写入“0”（上升沿有效）。
4. 选择 TIMx_CCR2 的有效输入：向 TIMx_CCMR1 寄存器中的 CC2S 写入 10（选择 tim_ti1）。
5. 选择 tim_ti1fp2 的有效极性（用于 TIMx_CCR2 中的捕获）：向 CC2P 位写入“1”，向 CC2NP 位写入“0”（下降沿有效）。
6. 选择有效触发输入：向 TIMx_SMCR 寄存器中的 TS 位写入 00101（选择 tim_ti1fp1）。
7. 将从模式控制器配置为复位模式：向 TIMx_SMCR 寄存器中的 SMS 位写入 100。
8. 使能捕获：向 TIMx_CCER 寄存器中的 CC1E 位和 CC2E 位写入“1”。

图 279. PWM 输入模式时序



1. PWM 输入模式只能与 TIMx_CH1/TIMx_CH2 信号配合使用，因为只有 tim_ti1fp1 和 tim_ti2fp2 与从模式控制器相连。

27.4.9 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中的 CCxS 位 = 00）下，可直接由软件将每个输出比较信号（tim_ocxref 和 tim_ocx）强制设置为有效电平或无效电平，而无需考虑输出比较寄存器和计数器之间的任何比较结果。

要将输出比较信号 (tim_ocxref/tim_ocx) 强制设置为有效电平，只需向相应 TIMx_CCMRx 寄存器中的 OCxM 位写入 101。tim_ocxref 进而强制设置为高电平（tim_ocxref 始终为高电平有效），同时 tim_ocx 获取 CCxP 极性位的相反值。

例如：CCxP = 0（tim_ocx 高电平有效）=> tim_ocx 强制设置为高电平。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入 100，可将 tim_ocxref 信号强制设置为低电平。

当然，即使在强制输出模式下，TIMx_CCRx 影子寄存器与计数器之间的比较仍然会执行，而且允许将标志置 1。因此可发送相应的中断和 DMA 请求。输出比较模式一节对此进行了介绍。

27.4.10 输出比较模式

此功能用于控制输出波形，或指示已经过某一时间段。

当捕获/比较寄存器与计数器之间相匹配时，输出比较功能：

- 将为相应的输出引脚分配一个可编程值，该值由输出比较模式（TIMx_CCMRx 寄存器中的 OCxM 位）和输出极性（TIMx_CCER 寄存器中的 CCxP 位）定义。匹配时，输出引脚既可保持其电平 (OCXM=000)，也可设置为有效电平 (OCXM=001)、无效电平 (OCXM=010) 或进行翻转 (OCxM=011)。
- 将中断状态寄存器中的标志置 1（TIMx_SR 寄存器中的 CCxIF 位）。
- 如果相应中断使能位（TIMx_DIER 寄存器中的 CCXIE 位）置 1，将生成中断。
- 如果相应使能位（TIMx_DIER 寄存器的 CCxDE 位，TIMx_CR2 寄存器的 CCDS 位，用来选择 DMA 请求）置 1，将发送 DMA 请求。

使用 TIMx_CCMRx 寄存器中的 OCxPE 位，可将 TIMx_CCRx 寄存器配置为带或不带预装功能。

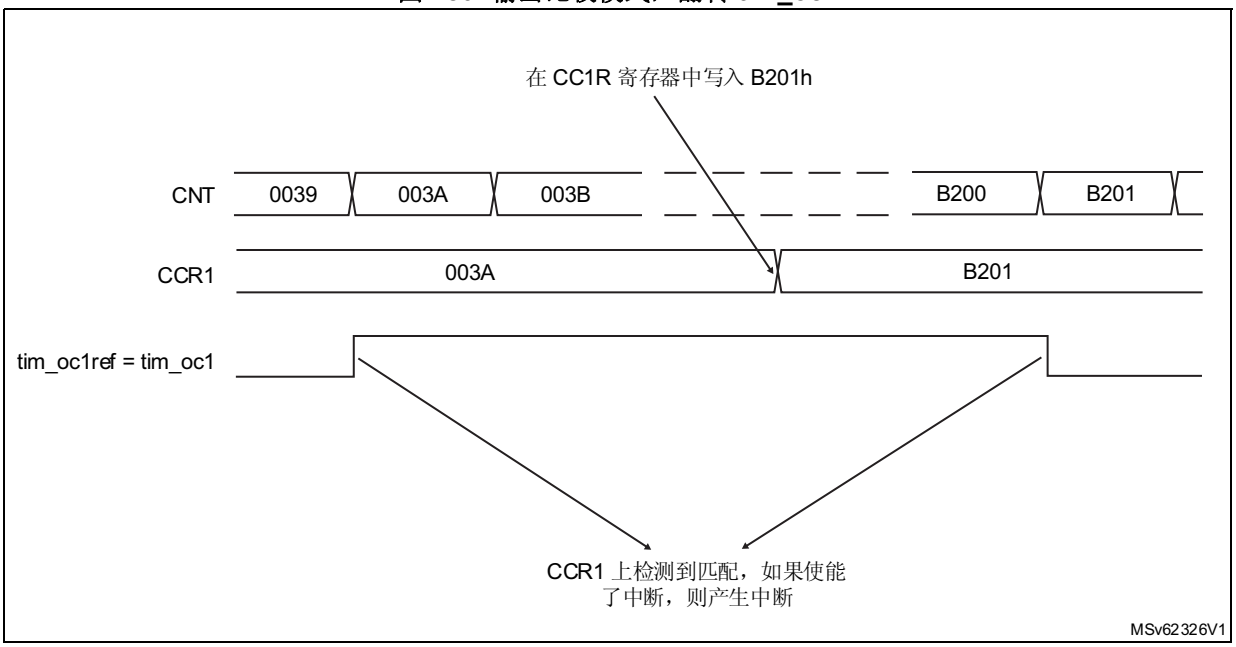
在输出比较模式下，更新事件 UEV 对 tim_ocxref 和 tim_ocx 输出毫无影响。时间分辨率可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

步骤

1. 选择计数器时钟（内部、外部、预分频器）。
2. 在 TIMx_ARR 和 TIMx_CCRx 寄存器中写入所需数据。
3. 如果要生成中断和/或 DMA 请求，将 CCxIE 位和/或 CCxDE 位置 1。
4. 选择输出模式。例如：
   a) 当 CNT 与 CCRx 匹配时，写入 OCxM = 0011 以翻转 tim_ocx 输出引脚
   b) 写入 OCxPE = 0 以禁止预装功能
   c) 写入 CCxP = 0 以选择高电平有效极性
   d) 写入 CCxE = 1 以使能输出
5. 通过将 TIMx_CR1 寄存器中的 CEN 位置 1 来使能计数器。

可随时通过软件更新 TIMx_CCRx 寄存器以控制输出波形，前提是未使能预装功能（OCxPE=0，否则仅当发生下一个更新事件 UEV 时，才会更新 TIMx_CCRx 影子寄存器）。*图 280* 给出了一个示例。

图 280. 输出比较模式，翻转 tim_oc1



27.4.11 PWM 模式

脉宽调制模式用于生成信号，该信号频率由 TIMx_ARR 寄存器值决定，其占空比则由 TIMx_CCRx 寄存器值决定。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），可以独立选择各通道（每个 tim_ocx 输出对应一个 PWM）的 PWM 模式。必须通过将 TIMx_CCMRx 寄存器中的 OCxPE 位置 1 使能相应预装载寄存器，最后通过将 TIMx_CR1 寄存器中的 ARPE 位置 1 使能自动重载预装载寄存器（在递增计数或中心对齐模式下）。

由于只有在发生更新事件时预装载寄存器内容才会传送到影子寄存器，因此启动计数器之前，必须通过将 TIMx_EGR 寄存器中的 UG 位置 1 来初始化所有寄存器。

tim_ocx 极性可通过软件来编程（使用 TIMx_CCER 寄存器的 CCxP 位）。可将其编程为高电平有效或低电平有效。可通过 TIMx_CCER 寄存器中的 CCxE 位使能 tim_ocx 输出。有关详细信息，请参见 TIMx_CCERx 寄存器说明。

在 PWM 模式（1 或 2）下，TIMx_CNT 与 TIMx_CCRx 进行实时比较，以确定是 TIMx_CCRx ≤ TIMx_CNT 还是 TIMx_CNT ≤ TIMx_CCRx（取决于计数器计数方向）。tim_ocref_clr 可通过 tim_etr_in 或 tim_oceref_clr 信号由外部事件清除。tim_ocref_clr 信号仅在以下情况下变为有效状态：

- 发生比较匹配事件后
- 输出比较模式（TIMx_CCMRx 寄存器中的 OCxM 位）从“冻结”配置（不进行比较，OCxM=“000”）切换为任一 PWM 模式（OCxM=“110”或“111”）。定时器运行期间，可以通过软件强制 PWM 输出。

根据 TIMx_CR1 寄存器中的 CMS 位状态，定时器能够产生边沿对齐模式或中心对齐模式的 PWM 信号。

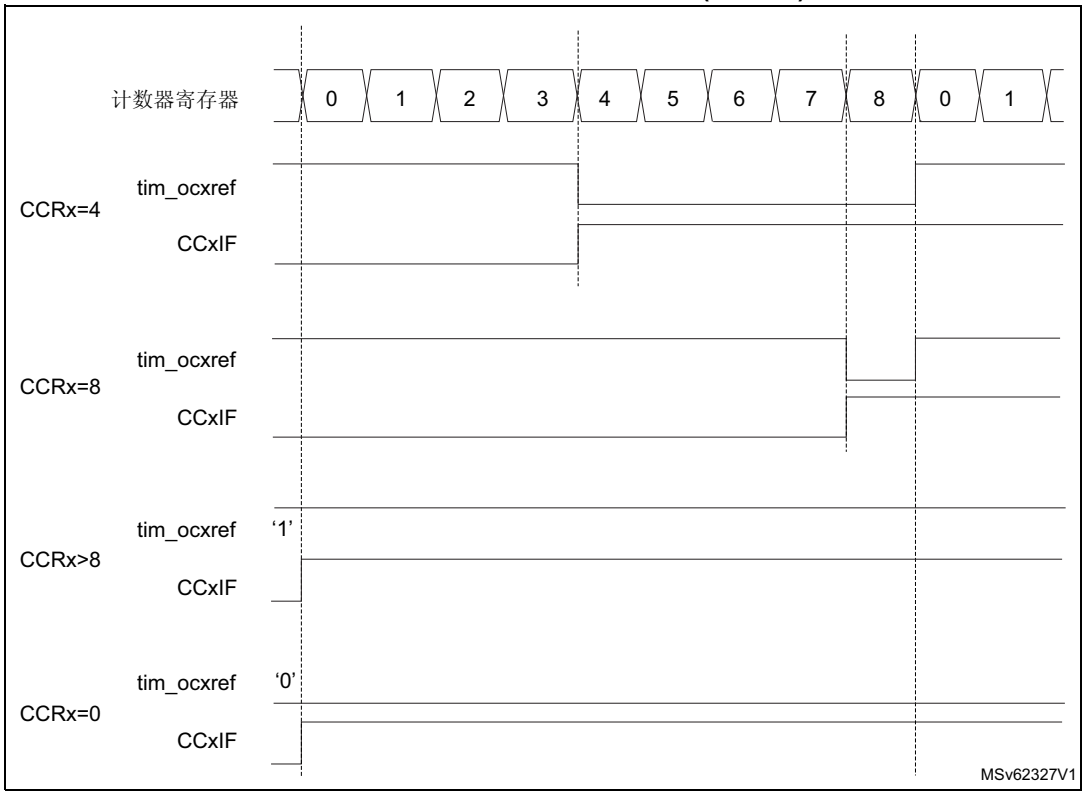
PWM 边沿对齐模式

递增计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低时执行递增计数。请参见*第 826 页的递增计数模式*。

以下以 PWM 模式 1 为例。只要 TIMx_CNT < TIMx_CCRx，PWM 参考信号 tim_ocxref 便为高电平，否则为低电平。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR 中），则 tim_ocxref 保持为“1”。如果比较值为 0，则 tim_ocxref 保持为“0”。*图 281* 举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

图 281. 边沿对齐模式的 PWM 波形 (ARR=8)



递减计数配置

当 TIMx_CR1 寄存器中的 DIR 位为高时执行递减计数。请参见*第 829 页的递减计数模式*。

在 PWM 模式 1 下，只要 TIMx_CNT > TIMx_CCRx，参考信号 tim_ocxref 即为低电平，否则为高电平。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重载值，则 tim_ocxref 保持为 100%。此模式下不可能产生 PWM。

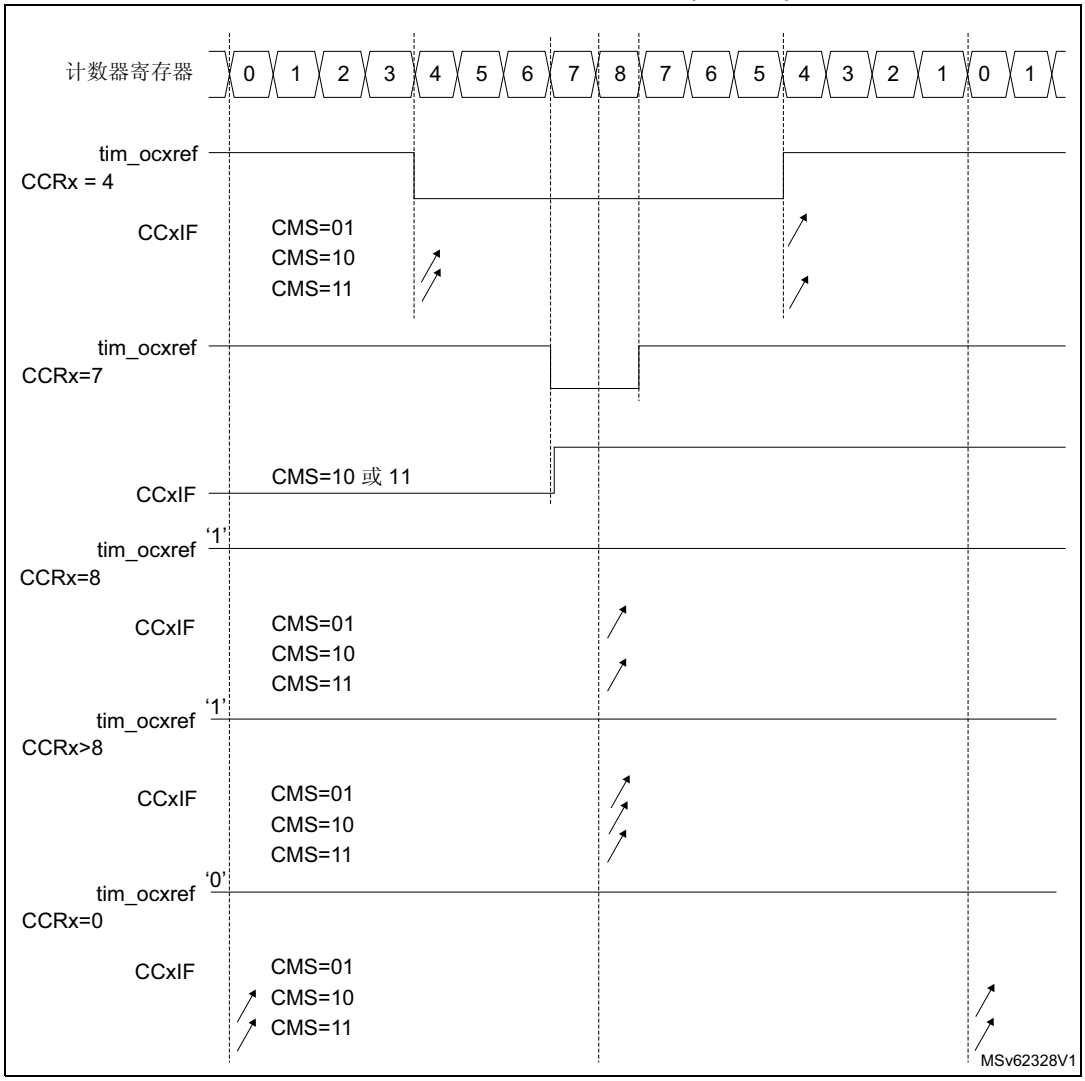
PWM 中心对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时（其余所有配置对 tim_ocxref/tim_ocx 信号具有相同的作用），中心对齐模式生效。根据 CMS 位的配置，可以在计数器递增计数、递减计数或同时递增和递减计数时将比较标志置 1。TIMx_CR1 寄存器中的方向位 (DIR) 由硬件更新，不得通过软件更改。请参见*第 832 页的中心对齐模式（递增/递减计数）*。

图 282 显示了中心对齐模式的 PWM 波形，在此例中：

- TIMx_ARR=8，
- PWM 模式为 PWM 模式 1，
- 在根据 TIMx_CR1 寄存器中 CMS=01 而选择的中心对齐模式 1 下，当计数器递减计数时，比较标志置 1。

图 282. 中心对齐模式 PWM 波形 (ARR=8)



中心对齐模式使用建议：

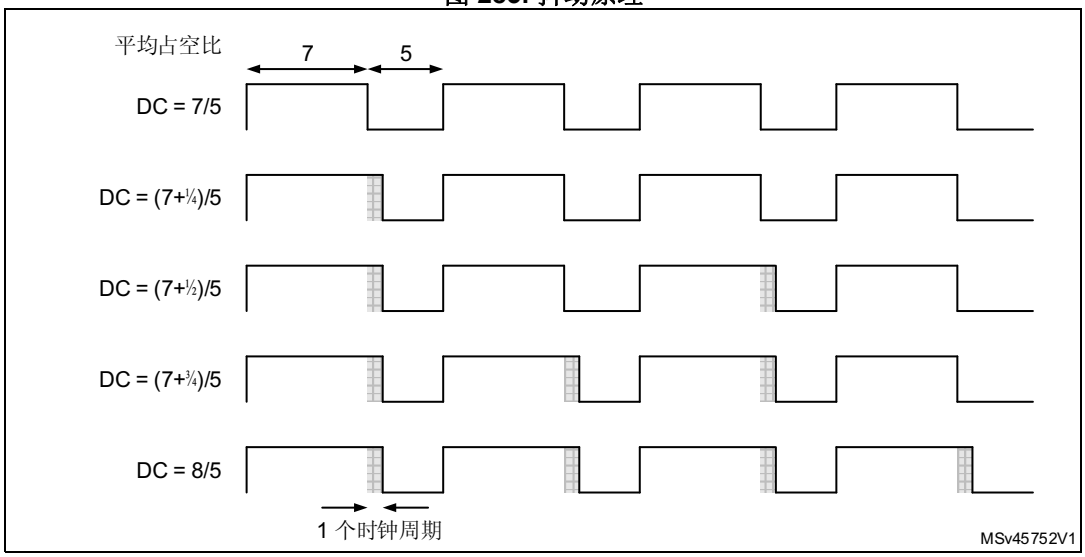
- 启动中心对齐模式时将使用当前的递增/递减计数配置。这意味着计数器将根据写入 TIMx_CR1 寄存器中 DIR 位的值进行递增或递减计数。此外，不得通过软件同时修改 DIR 和 CMS 位。
- 不建议在中心对齐模式下对计数器执行写操作，否则将发生意外错误。尤其是：
  - 如果写入计数器的值大于自动重载值 (TIMx_CNT>TIMx_ARR)，计数方向不会更新。例如，如果计数器之前递增计数，则继续递增计数。
  - 如果向计数器写入 0 或 TIMx_ARR 的值，计数方向会更新，但不生成更新事件 UEV。
- 使用中心对齐模式最为保险的方法是：在启动计数器前通过软件生成更新（将 TIMx_EGR 寄存器中的 UG 位置 1），并且不要在计数器运行过程中对其执行写操作。

抖动模式

使用 TIMx_CR1 寄存器中的 DITHEN 位使能抖动模式，可以提高 PWM 模式的有效分辨率。这适用于 CCR（提高占空比分辨率）和 ARR（提高 PWM 频率分辨率）。

工作原理如下：在 16 个连续的 PWM 周期内按照预定义模式略微更改（增加或不增加一个定时器时钟周期）实际的 CCR（或 ARR）值。考虑到平均占空比或 PWM 周期，这样可以将分辨率提高 16 倍。*图 283* 介绍了应用于 4 个连续 PWM 周期的抖动原理。

图 283. 抖动原理



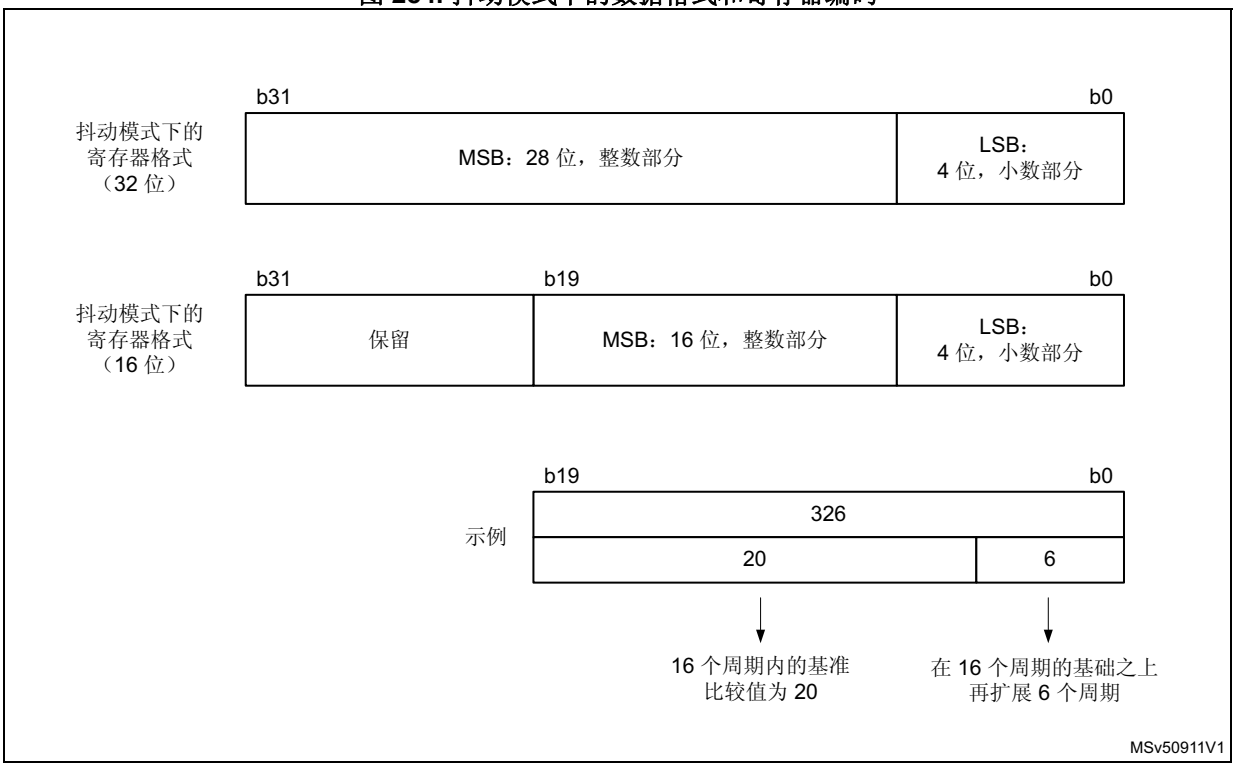
使能抖动模式后，寄存器编码的变化如下（请参见*图 284* 获取示例）：

- 4 个 LSB 用于对提高的分辨率部分（小数部分）进行编码。
- MSB 左移 4 位，用于对基值进行编码。在 16 位模式下，保持 16 位格式。

注意： *复位 DITHEN 位时必须遵循以下顺序：*

1. *CEN 和 ARPE 位必须复位。*
2. *DITHEN 位必须复位。*
3. *CCIF 标志必须清零。*
4. *CEN 位可以置 1（最终 ARPE = 1）。*

图 284. 抖动模式下的数据格式和寄存器编码



最低频率可根据以下公式得出：

$$\text{分辨率} = \frac{F_{Tim}}{F_{pwm}} \Rightarrow F_{pwmMin} = \frac{F_{Tim}}{Max_{Resolution}}$$

$$\text{禁止抖动模式：} F_{pwmMin} = \frac{F_{Tim}}{65536}$$

$$\text{抖动模式（16 位定时器）：} F_{pwmMin} = \frac{F_{Tim}}{65535 + \frac{15}{16}}$$

$$\text{抖动模式（32 位定时器）：} F_{pwmMin} = \frac{F_{Tim}}{268435454 + \frac{15}{16}}$$

注意： *对于 16 位定时器，抖动模式下的最大 TIMx_ARR 和 TIMxCCRy 值限制为 0xFFFEF，该值对应于 65534（整数部分）和 15（抖动部分）。*
对于 32 位定时器，抖动模式下的最大 TIMx_ARR 和 TIMxCCRy 值限制为 0xFFFFFFEF，该值对应于 264435454（整数部分）和 15（抖动部分）。

如下面的*图 285* 和*图 286* 所示，抖动模式用于提高 PWM 分辨率。

图 285. PWM 分辨率与频率的关系（16 位模式）

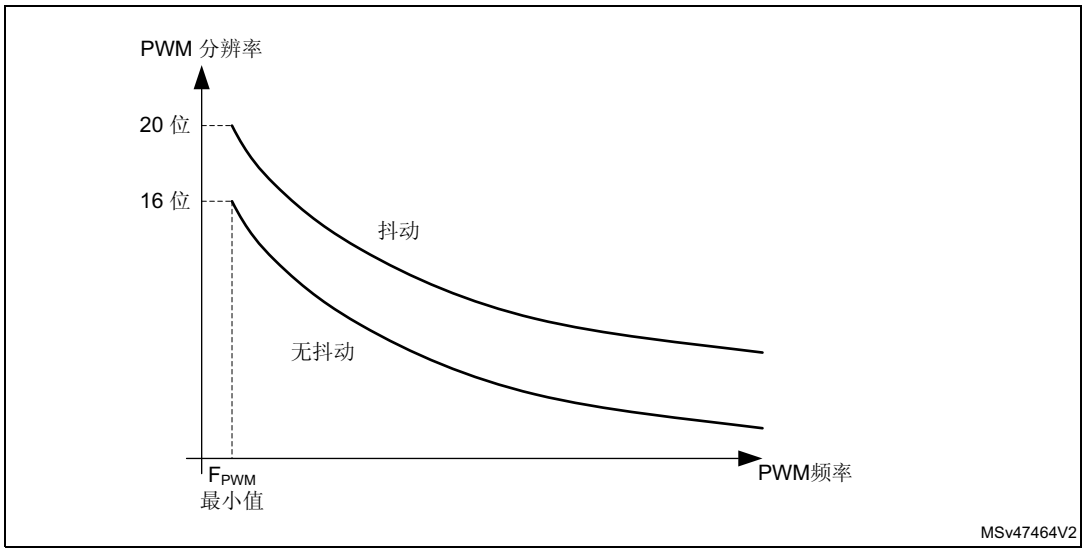


图 286. PWM 分辨率与频率的关系（32 位模式）

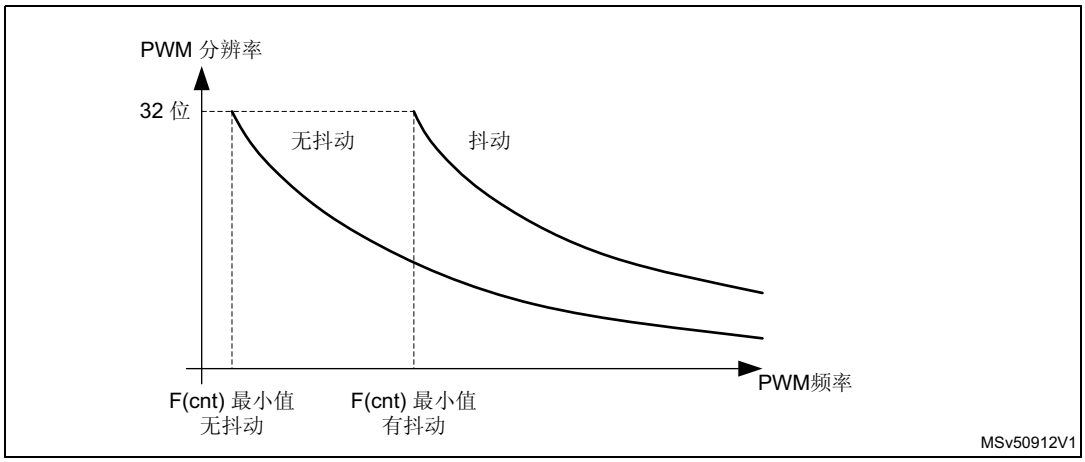
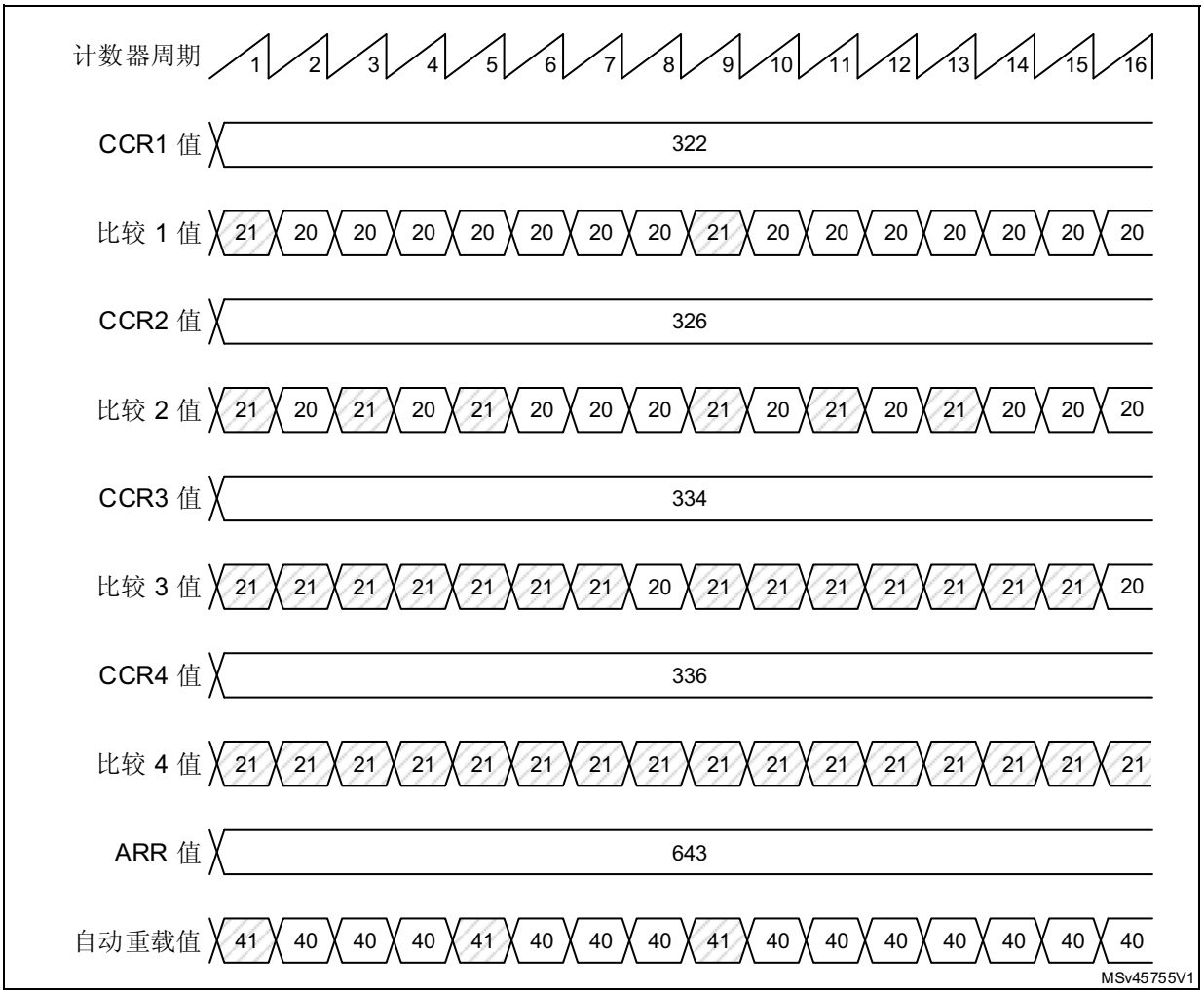


图 287 详细描述了占空比和/或周期在 16 个连续周期内的变化情况。

图 287. PWM 抖动模式



自动重载和比较值增量在特定模式下的分布情况如下面的*表 193* 所示。抖动序列的目的是尽可能均匀地分布增量并将整体纹波降至最低。

表 193. CCR 和 ARR 寄存器变化抖动模式

LSB 值	PWM 周期															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

抖动模式也适用于中心对齐 PWM 模式（TIMx_CR1 寄存器中的 CMS 位不等于“00”）。在这种情况下，考虑到递增和递减计数阶段，可在 8 个连续的 PWM 周期内应用抖动模式，如下面的*图 288* 所示。

图 288. 中心对齐 PWM 模式下的抖动对占空比的影响

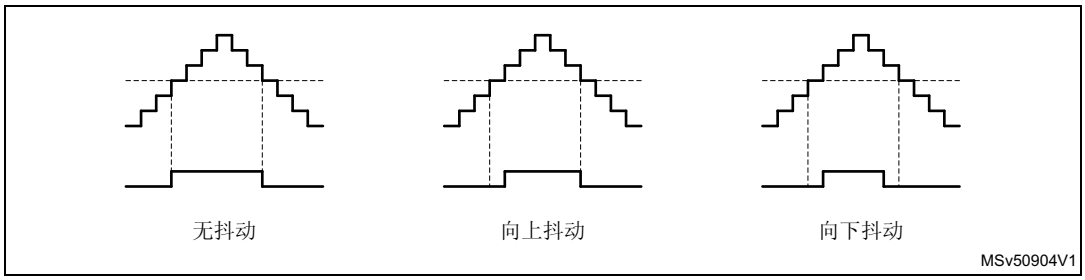


表 194 显示了如何在中心对齐 PWM 模式下应用抖动模式。

表 194. 中心对齐 PWM 模式下的 CCR 寄存器变化抖动模式

LSB 值	PWM 周期															
	1		2		3		4		5		6		7		8	
	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减	递增	递减
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

27.4.12 非对称 PWM 模式

在非对称模式下，生成的两个中心对齐 PWM 信号间允许存在可编程相移。频率由 TIMx_ARR 寄存器的值确定，而占空比和相移则由一对 TIMx_CCRx 寄存器确定。两个寄存器分别控制递增计数和递减计数期间的 PWM，这样每半个 PWM 周期便会调节一次 PWM：

- tim_oc1refc（或 tim_oc2refc）由 TIMx_CCR1 和 TIMx_CCR2 控制
- tim_oc3refc（或 tim_oc4refc）由 TIMx_CCR3 和 TIMx_CCR4 控制

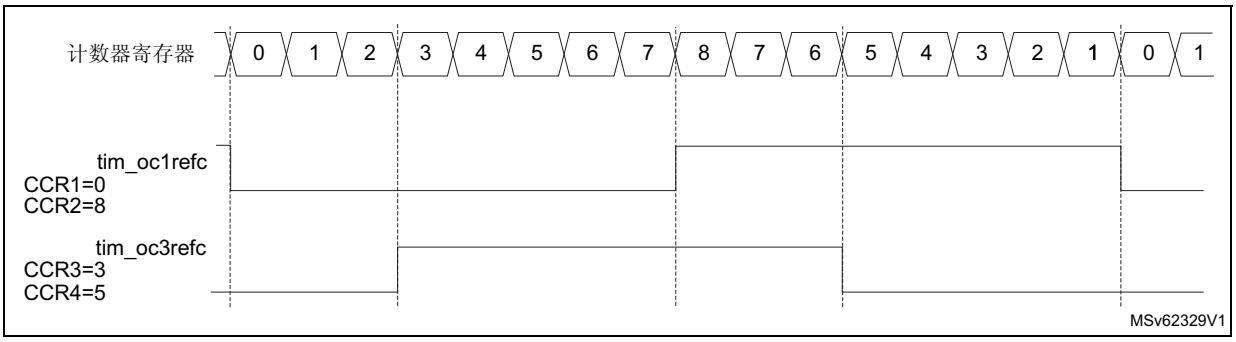
两个通道可以独立选择非对称 PWM 模式（每对 CCR 寄存器一个 tim_ocx 输出），只需向 TIMx_CCMRx 寄存器的 OCxM 位写入“1110”（非对称 PWM 模式 1）或“1111”（非对称 PWM 模式 2）。

注意： *出于兼容性原因，OCxM[3:0] 位域分为两部分，最高有效位与最低有效的 3 位不相邻。*

给定通道用作不对称 PWM 通道时，也可使用其辅助通道。例如，如果通道 1 上产生 tim_oc1refc 信号（非对称 PWM 模式 1），则由于非对称 PWM 模式 2 的原因，通道 2 上可输出 tim_oc2ref 信号或 tim_oc2refc 信号。

图 289 显示了非对称 PWM 模式下可以产生的信号示例（通道 1 到通道 4 配置为非对称 PWM 模式 2）。

图 289. 50% 占空比时产生的 2 个相移 PWM 信号



27.4.13 组合 PWM 模式

在组合 PWM 模式下，生成的两个边沿或中心对齐 PWM 信号的各个脉冲间允许存在可编程延时和相移。频率由 TIMx_ARR 寄存器的值确定，而占空比和延时则由两个 TIMx_CCRx 寄存器确定。产生的信号 tim_ocxrefc 由两个参考 PWM 的“逻辑或”运算或者“逻辑与”运算组合组成：

——tim_oc1refc（或 tim_oc2refc）由 TIMx_CCR1 和 TIMx_CCR2 控制

——tim_oc3refc（或 tim_oc4refc）由 TIMx_CCR3 和 TIMx_CCR4 控制

两个通道可以独立选择组合 PWM 模式（每对 CCR 寄存器一个 tim_ocx 输出），只需向 TIMx_CCMRx 寄存器的 OCxM 位写入“1100”（组合 PWM 模式 1）或“1101”（组合 PWM 模式 2）。

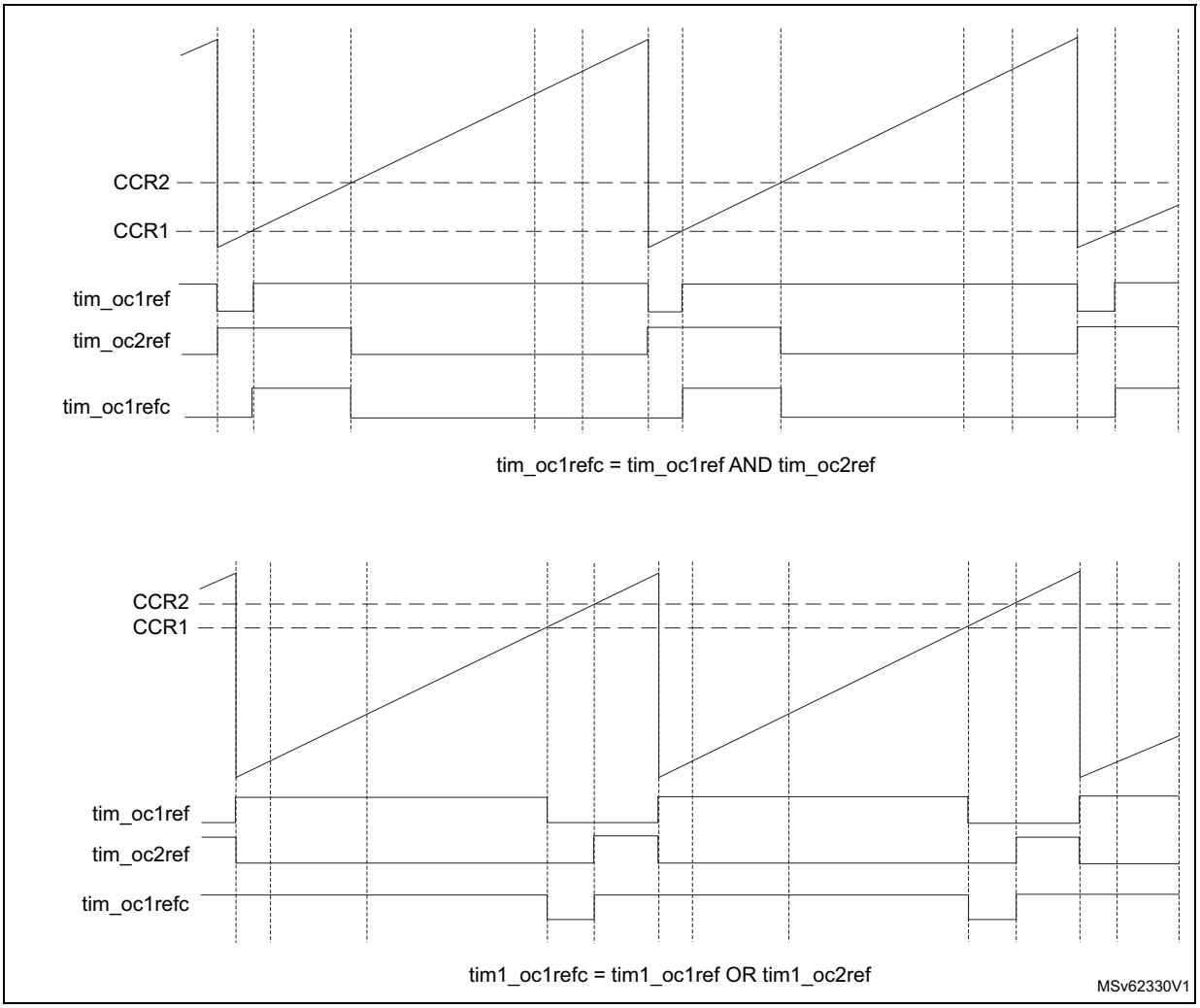
当给定通道用作组合 PWM 通道时，其辅助通道必须在相反的 PWM 模式下配置（例如，一个通道在组合 PWM 模式 1 下配置，另一个通道在组合 PWM 模式 2 下配置）。

注意： *出于兼容性原因，OCxM[3:0] 位域分为两部分，最高有效位与最低有效的 3 位不相邻。*

图 290 显示了组合 PWM 模式下可以产生的信号示例，通过以下配置可获得这些信号：

- 通道 1 配置为组合 PWM 模式 2，
- 通道 2 配置为 PWM 模式 1，
- 通道 3 配置为组合 PWM 模式 2，
- 通道 4 配置为 PWM 模式 1

图 290. 通道 1 和通道 3 上的组合 PWM 模式



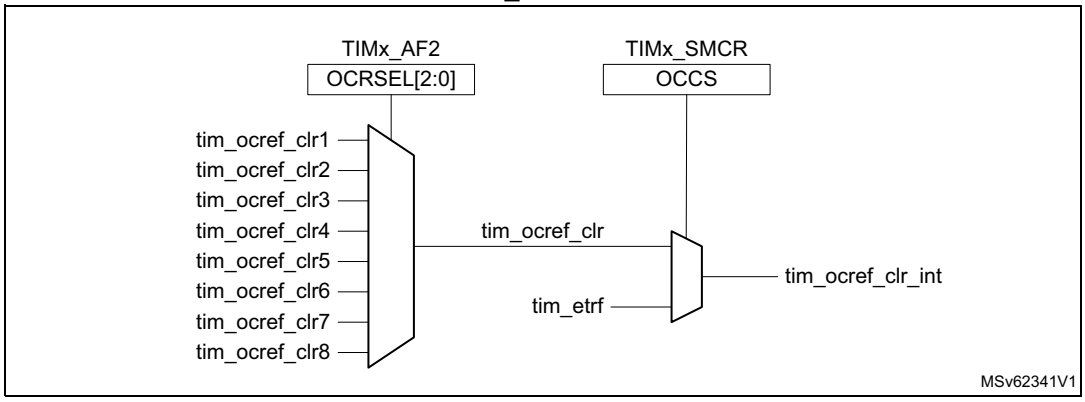
27.4.14 发生外部事件时清除 tim_ocxref 信号

对于给定通道，在 tim_ocref_clr_int 输入上施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置 1），可将 tim_ocxref 信号清零。tim_ocxref 信号将保持低电平，直到在下一 PWM 周期再次变为有效状态。该功能只能在输出比较模式和 PWM 模式下使用。在强制模式下不起作用。

tim_ocref_clr_int 源取决于 OCREF 清零选择功能的实现，请参见*第 27.3 节：TIM2/TIM3 实现*。

如果实现了 OCREF 清零选择功能，则可以通过配置 TIMx_SMCR 寄存器中的 OCCS 位，tim_ocref_clr_int 可以在 tim_ocref_clr 输入和 tim_etrf 输入之间选择。可以使用 TIMx_AF2 寄存器中的 OCRSEL[2:0] 位域在多个 tim_ocref_clr[7:0] 输入中选择 tim_ocref_clr 输入，如下面的*图 291* 所示。

图 291. OCREF_CLR 输入选择复用器



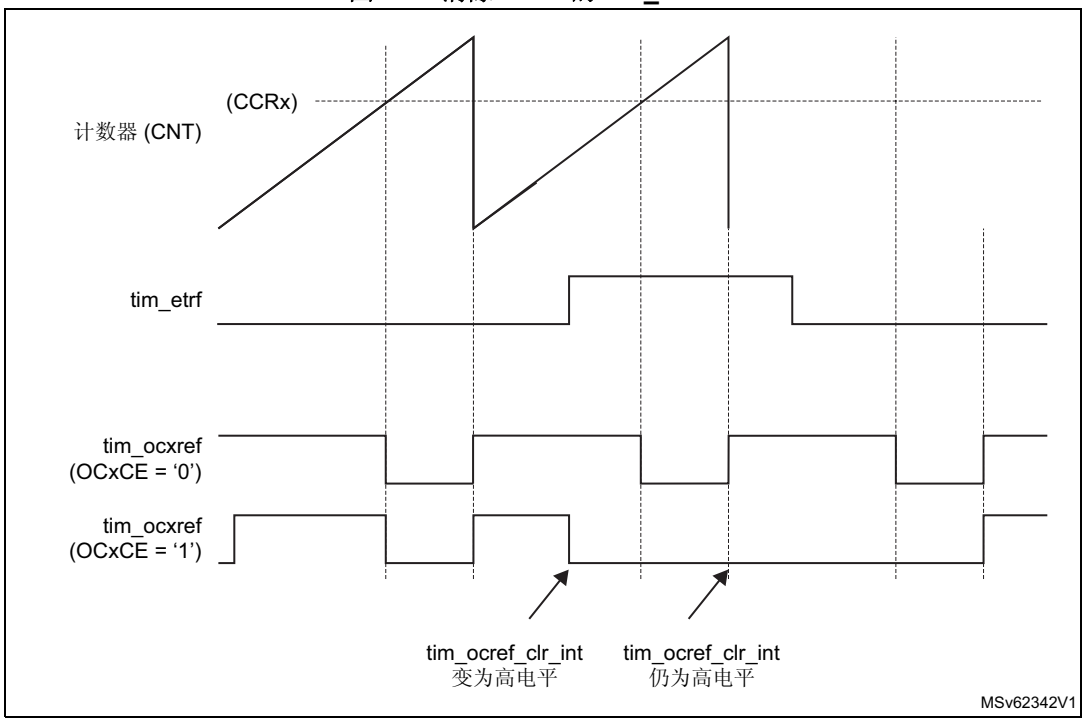
如果未实现 OCREF 清零选择功能，则 tim_ocref_clr_int 输入直接连接到 tim_etrf 输入。

例如，tim_ocref_clr_int 信号可以连接到比较器的输出，用于控制电流。此时，tim_etr_in 必须配置如下：

1. 必须关闭外部触发预分频器：TIMx_SMCR 寄存器中的 ETPS[1:0] 位清为 00。
2. 必须禁止外部时钟模式 2：TIM1_SMCR 寄存器中的 ECE 位清为 0。
3. 外部触发极性 (ETP) 和外部触发滤波器 (ETF) 可根据应用需要进行配置。

图 292 对比了不同使能位 OCxCE 值的情况，显示了当 tim_etrf 输入变为高电平时 tim_ocxref 信号的行为。在本例中，定时器 TIMx 编程为 PWM 模式。

图 292. 清除 TIMx 的 tim_ocxref



注意： *如果 PWM 的占空比为 100% (CCRx>ARR)，则下次计数器溢出时会再次使能 tim_ocxref。*

27.4.15 单脉冲模式

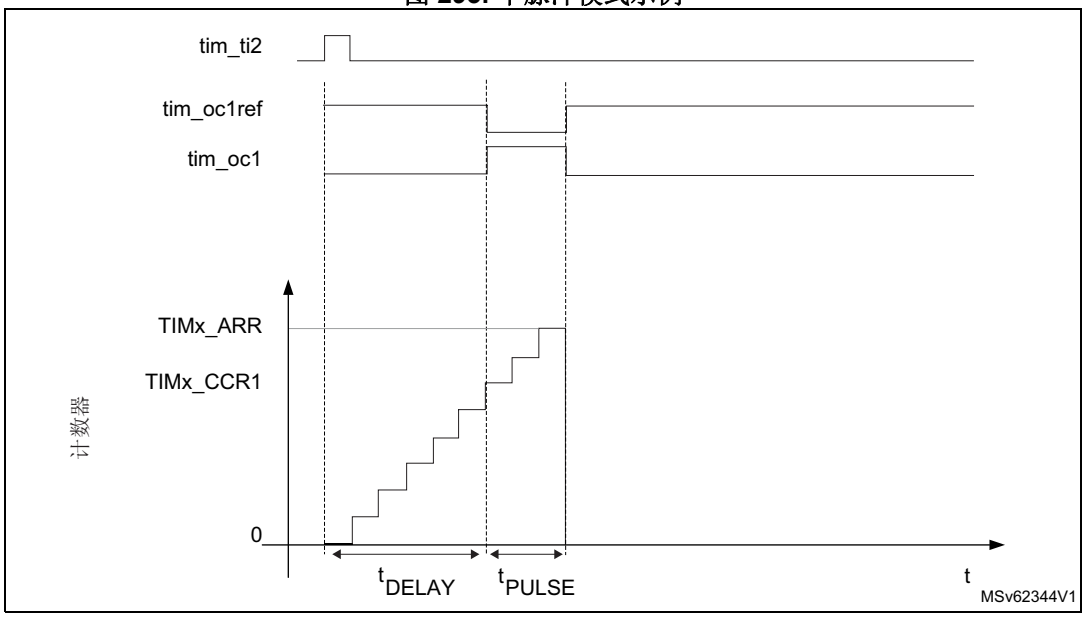
单脉冲模式 (OPM) 是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。通过将 TIMx_CR1 寄存器中的 OPM 位置 1 选择单脉冲模式。这样，发生下一更新事件 UEV 时，计数器将自动停止。

只有当比较值与计数器初始值不同时，才能正确产生一个脉冲。启动前（定时器等待触发时），必须进行如下配置：

- CNT < CCRx ≤ ARR（特别注意，0 < CCRx），

图 293. 单脉冲模式示例



例如，用户希望达到这样的效果：在 tim_ti2 输入引脚检测到上升沿时，经过 t_{DELAY} 的延迟，在 tim_oc1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 tim_ti2fp2 作为触发 1：

1. 使用 TIMx_TISEL 寄存器中的 TI2SEL[3:0] 位选择正确的 tim_ti2_in[15:0] 源（内部或外部）。
2. 在 TIMx_CCMR1 寄存器中写入 CC2S=01，将 tim_ti2fp2 映射到 tim_ti2。
3. 在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=“0”，使 tim_ti2fp2 能够检测上升沿。
4. 在 TIMx_SMCR 寄存器中写入 TS = 00110，将 tim_ti2fp2 配置为从模式控制器的触发 (tim_trgi)。
5. 在 TIMx_SMCR 寄存器中写入 SMS=“110”（触发模式），以使用 tim_ti2fp2 启动计数器。

OPM 波形通过对比较寄存器执行写操作来定义（考虑时钟频率和计数器预分频器）。

- t_{DELAY} 由写入 TIMx_CCR1 寄存器的值定义。
- t_{PULSE} 由自动重载值与比较值之差 (TIMx_ARR - TIMx_CCR1) 来定义。
- 假设希望产生这样的波形：信号在发生比较匹配时从“0”变为“1”，在计数器达到自动重载值时由“1”变为“0”。为此，必须通过在 TIMx_CCMR1 寄存器中写入 OC1M=111，来使能 PWM 模式 2。可选择在 TIMx_CCMR1 寄存器的 OC1PE 和 TIMx_CR1 寄存器的 ARPE 中写入“1”，以使能预装载寄存器。这种情况下，必须在 TIMx_CCR1 寄存器中写入比较值并在 TIMx_ARR 寄存器中写入自动重载值，通过将 UG 位置 1 来产生更新，然后等待 tim_ti2 上的外部触发事件。本例中，CC1P 的值为“0”。

在本例中，TIMx_CR1 寄存器中的 DIR 和 CMS 位必须为低。

由于仅需要 1 个脉冲（单脉冲模式），因此必须向 TIMx_CR1 寄存器的 OPM 位写入 1，以便在发生下一更新事件（计数器从自动重载值返回到 0）时使计数器停止计数。TIMx_CR1 寄存器中的 OPM 位置“0”时，即选择重复模式。

特殊情况：tim_ocx 快速使能：

在单脉冲模式下，tim_tix 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时，将切换输出。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟（t_{DELAY} 最小值）。

如果要输出延迟时间最短的波形，可以将 TIMx_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 tim_ocxref（和 tim_ocx）对激励信号做出响应，而不再考虑比较的结果。其新电平与发生比较匹配时相同。仅当通道配置为 PWM1 或 PWM2 模式时，OCxFE 才会起作用。

27.4.16 可再触发单脉冲模式

该模式可以启动计数器来响应激励信号，并且能产生长度可编程的脉冲，但与不可再触发单脉冲模式间存在以下差别，如*第 27.4.15 节*所述：

- 发生触发时，脉冲立即产生（无可编程延时）
- 如果在上一个触发完成前发生新的触发，脉冲将延长

定时器必须处于从模式，TIMx_SMCR 寄存器中的位 SMS[3:0] =“1000”（组合复位 + 触发模式），针对可再触发 OPM 模式 1 或模式 2 将 OCxM[3:0] 位设置为“1000”或“1001”。

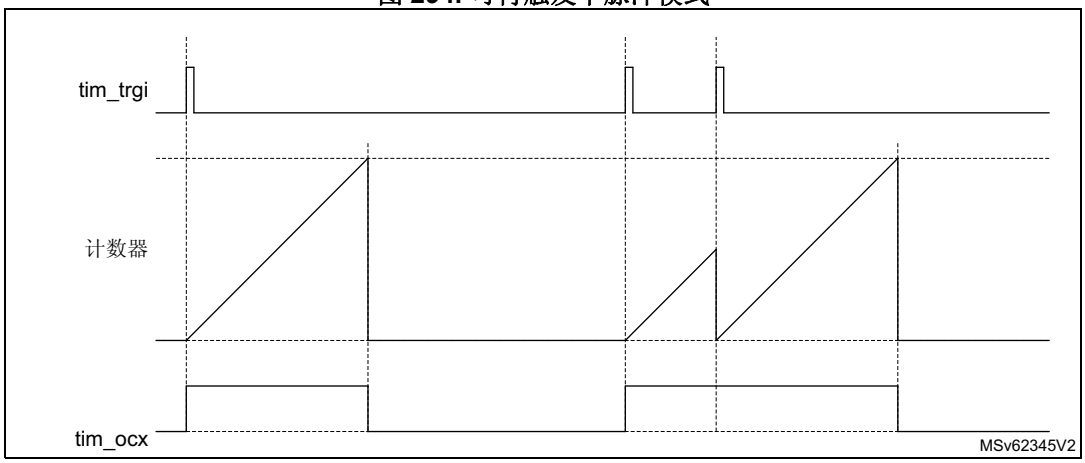
定时器配置为递增计数模式时，相应的 CCRx 必须置 0（ARR 寄存器设置脉冲长度）。如果定时器配置为递减计数模式，CCRx 必须高于或等于 ARR。

注意： *在可再触发单脉冲模式下，CCxIF 标志没有意义。*

出于兼容性原因，OCxM[3:0] 和 SMS[3:0] 位域分为两部分，最高有效位与最低有效的 3 位不相邻。

此模式不能与中心对齐 PWM 模式一起使用。在 TIMx_CR1 中必须设置 CMS[1:0] = 00。

图 294. 可再触发单脉冲模式



27.4.17 基于比较的脉冲模式

可以在发生比较匹配事件时生成脉冲。当计数器值等于给定的比较值时可产生脉宽可编程的信号，该信号可用于调试或同步。

该模式可用于任何选定的从模式，包括编码器模式、边沿和中心对齐计数模式。该模式仅适用于通道 3 和通道 4。脉冲发生器只有一个，由两个通道共用，如下面的*图 295* 所示。

图 295. 脉冲发生器电路

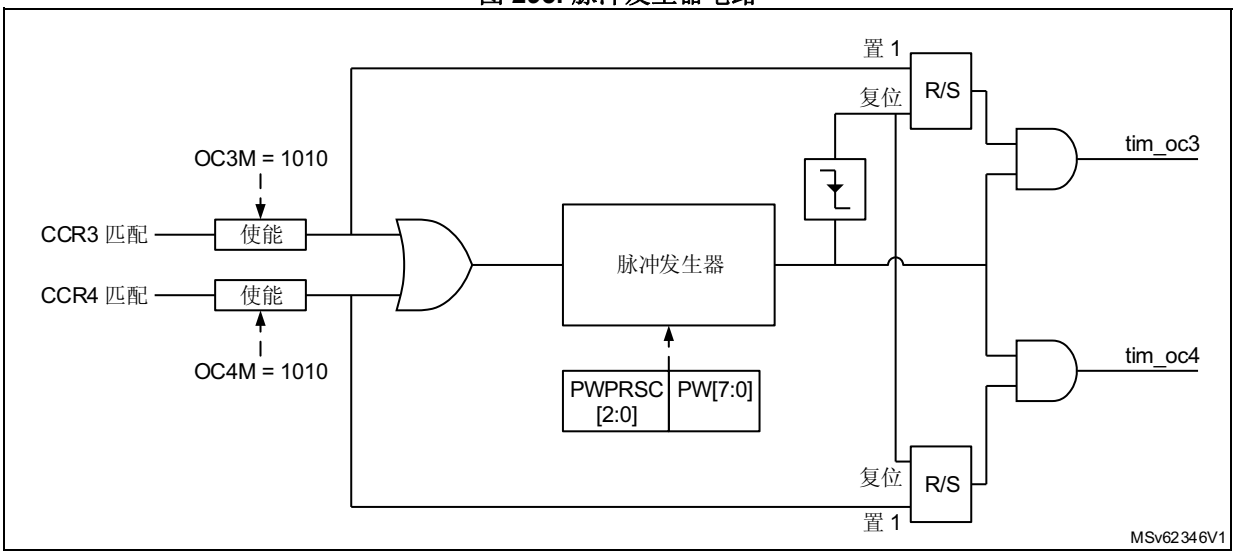
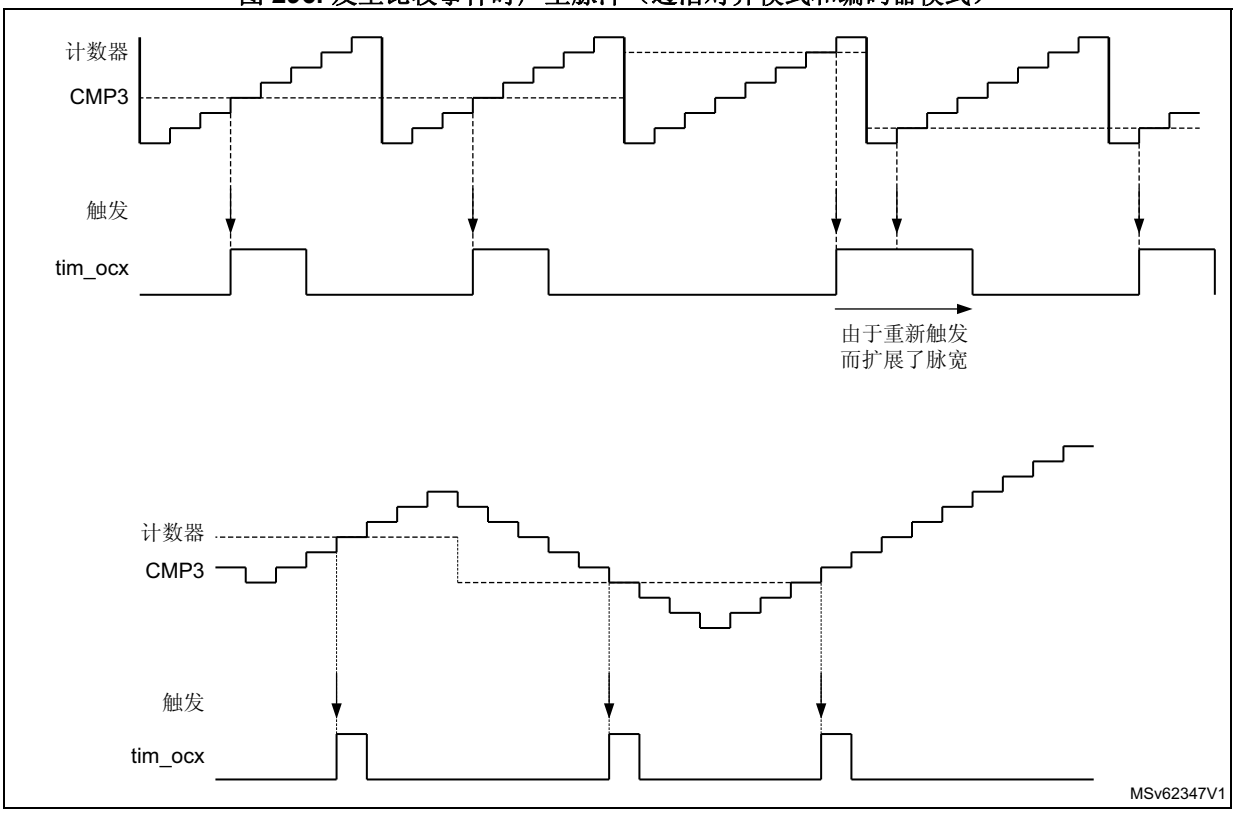


图 296 显示了如何在边沿对齐模式和编码器模式下产生脉冲。

图 296. 发生比较事件时产生脉冲（边沿对齐模式和编码器模式）



该输出比较模式使用 TIMx_CCMR2 寄存器中的 OC3M[3:0] 和 OC4M[3:0] 位域进行选择。

脉宽使用寄存器中的 PW[7:0] 位域以及根据 PWPRSC[2:0] 位预分频的特定时钟进行编程，具体如下所示：

$t_{PW} = PW[7:0] \times t_{PWG}$

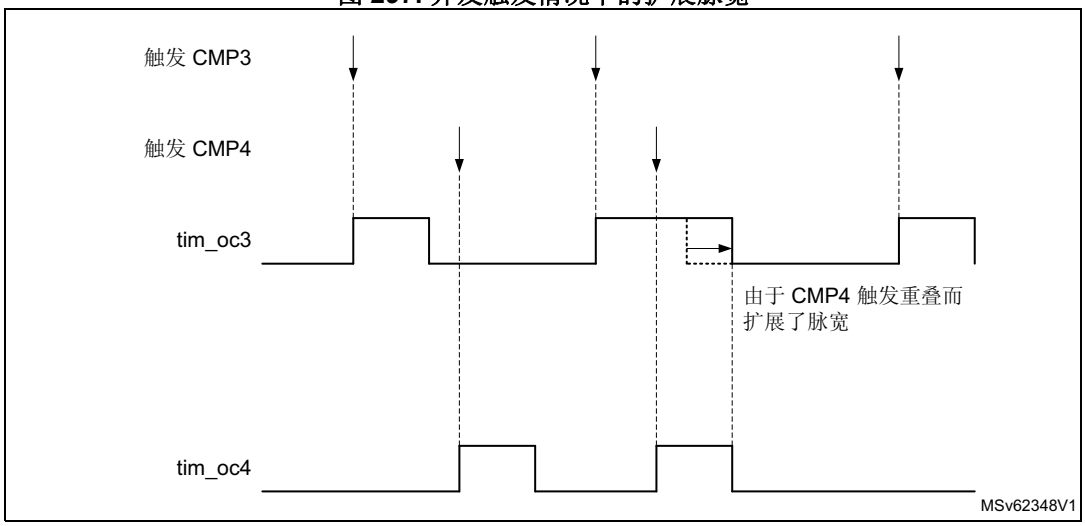
其中 $t_{PWG} = (2^{(PWPRSC[2:0])}) \times t_{tim_ker_ck}$。

给出了根据预分频值得出的分辨率和最大值。

脉冲可再触发：如果正在产生脉冲时发生新的触发，脉冲将会延长。

注意： *如果同时使能两个通道，则只要一个通道上的触发信号未与并发输出上产生的脉冲重叠，就可以独立发出脉冲。相反地，如果两个触发信号发生重叠，则先到达的触发信号的脉冲宽度将被延长（因为再触发），而后到达的触发信号的脉冲宽度则正确无误（如下面的图 297 所示）。*

图 297. 并发触发情况下的扩展脉宽



27.4.18 编码器接口模式

正交编码器

选择编码器接口模式时，如果计数器仅在 tim_ti1 边沿处计数，在 TIMx_SMCR 寄存器中写入 SMS=0001；如果计数器仅在 tim_ti2 边沿处计数，写入 SMS=0010；如果计数器在 tim_ti1 和 tim_ti2 边沿处均计数，则写入 SMS =0011。

通过编程 TIMx_CCER 寄存器的 CC1P 和 CC2P 位，选择 tim_ti1 和 tim_ti2 极性。CC1NP 和 CC2NP 必须保持清零。如果需要，还可对输入滤波器进行编程。

tim_ti1 和 tim_ti2 两个输入用于连接增量编码器。请参见 *表 195*。如果使能计数器（在 TIMx_CR1 寄存器的 CEN 位中写入“1”），则计数器的时钟由 tim_ti1fp1 或 tim_ti2fp2 上的每次有效信号转换提供。tim_ti1fp1 和 tim_ti2fp2 是 tim_ti1 和 tim_ti2 进行输入滤波和极性选择后的信号，如果不进行滤波和反相，则 tim_ti1fp1=tim_ti1，tim_ti2fp2=tim_ti2。将根据两个输入的信号转换序列，产生计数脉冲和方向信号。根据该信号转换序列，计数器相应递增或递减计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应修改。任何输入（tim_ti1 或 tim_ti2）发生信号转换时，都会计算 DIR 位，无论计数器是仅在 tim_ti1 或 tim_ti2 边沿处计数，还是同时在 tim_ti1 和 tim_ti2 处计数。

编码器接口模式就相当于带有方向选择的外部时钟。这意味着，计数器仅在 0 到 TIMx_ARR 寄存器中的自动重载值之间进行连续计数（根据具体方向，从 0 递增计数到 ARR，或从 ARR 递减计数到 0）。因此，在启动前必须先配置 TIMx_ARR。同样，捕获、比较、预分频器、触发输出功能继续正常工作。编码器模式和外部时钟模式 2 不兼容，因此不能同时选择。

在此模式下，计数器会根据正交编码器的速度和方向自动进行修改，因此，其内容始终表示编码器的位置。计数方向对应于所连传感器的旋转方向。下表汇总了可能的组合（假设 tim_ti1 和 tim_ti2 不同时切换）。

表 195. 计数方向与编码器信号的关系 (CC1P = CC2P = 0)

有效边沿	SMS[3:0]	相反信号的电平（tim_ti1fp1 对应 tim_ti2，tim_ti2fp2 对应 tim_ti1）	tim_ti1fp1 信号		tim_ti2fp2 信号	
			上升	下降	上升	下降
仅在 tim_ti1 处计数 x1 模式	1110	高	递减	递增	不计数	不计数
		低	不计数	不计数	不计数	不计数
仅在 tim_ti2 处计数 x1 模式	1111	高	不计数	不计数	递增	递减
		低	不计数	不计数	不计数	不计数
仅在 tim_ti1 处计数 x2 模式	0001	高	递减	递增	不计数	不计数
		低	递增	递减	不计数	递减
仅在 tim_ti2 处计数 x2 模式	0010	高	不计数	不计数	递增	递减
		低	不计数	不计数	递减	递增
在 tim_ti1 和 tim_ti2 处均计数 x4 模式	0011	高	递减	递增	递增	递减
		低	递增	递减	递减	递增

正交编码器可直接与 MCU 相连，无需外部接口逻辑。不过，通常使用比较器将编码器的差分输出转换为数字信号。这样大幅提高了抗噪声性能。用于指示机械零位的第三个编码器输出可与外部触发输入相连，用以触发计数器复位。

图 298 以计数器工作为例，说明了计数信号的生成和方向控制。同时也说明了选择双边沿时如何对输入抖动进行补偿。将传感器靠近其中一个切换点放置时可能出现这种情况。本例中假设配置如下：

- CC1S=01（TIMx_CCMR1 寄存器，tim_ti1fp1 映射到 tim_ti1 上）
- CC2S=01（TIMx_CCMR2 寄存器，tim_ti2fp2 映射到 tim_ti2 上）
- CC1P 和 CC1NP = “0”（TIMx_CCER 寄存器，tim_ti1fp1 未反相，tim_ti1fp1=tim_ti1）
- CC2P 和 CC2NP = “0”（TIMx_CCER 寄存器，tim_ti2fp2 未反相，tim_ti2fp2=tim_ti2）
- SMS=0011（TIMx_SMCR 寄存器，两个输入在上升沿和下降沿均有效）
- CEN = 1（TIMx_CR1 寄存器，使能计数器）

图 298. 编码器接口模式下的计数器工作示例

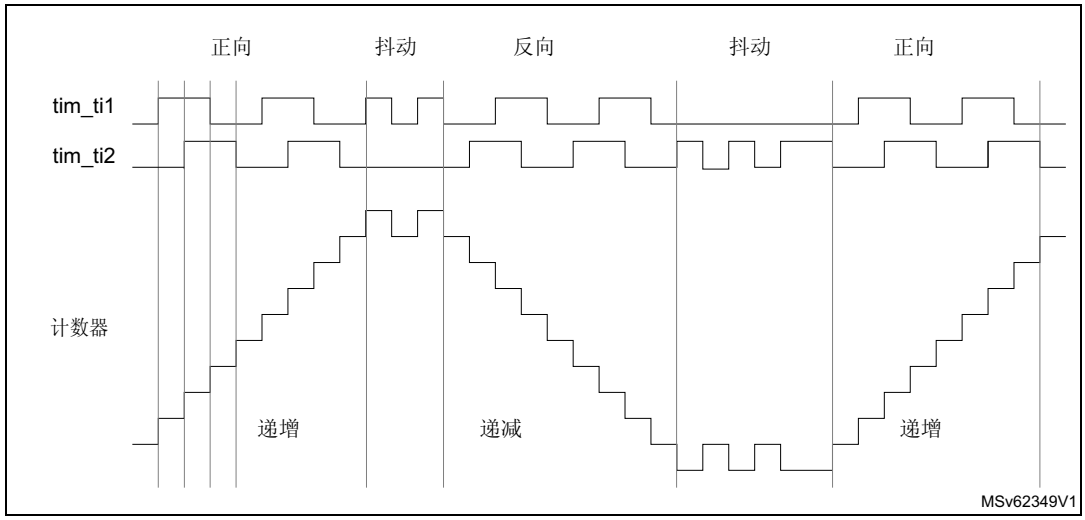
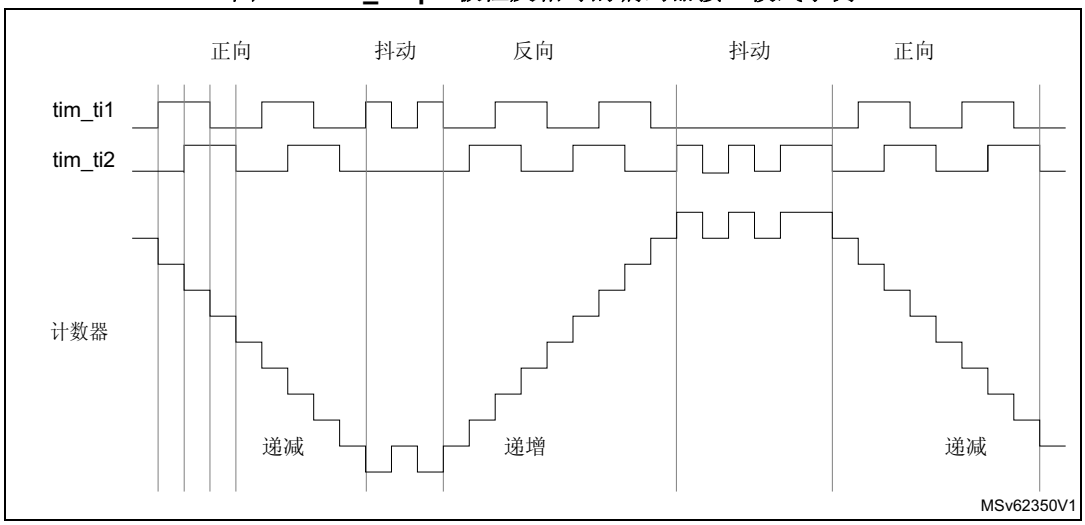


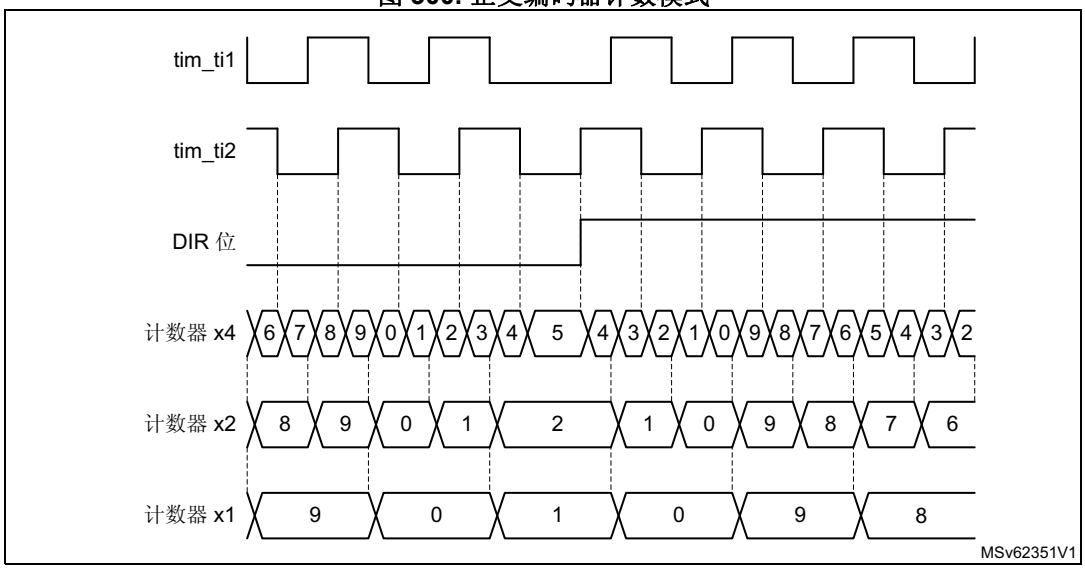
图 299 举例说明了 tim_ti1fp1 极性反相时的计数器行为（除 CC1P=1 外，其他配置与上例相同）。

图 299. tim_ti1fp1 极性反相时的编码器接口模式示例



下面的*图 300* 显示了在不同计数模式下速度反转期间的定时器计数器值。

图 300. 正交编码器计数模式



定时器配置为编码器接口模式时，会提供传感器当前位置的相关信息。使用另一个配置为捕获模式的定时器测量两个编码器事件之间的周期，可获得动态信息（速度、加速度和减速度）。指示机械零位的编码器输出即可用于此目的。根据两个事件之间的时间间隔，还可定期读取计数器。如果可能，可以将计数器值锁存到第三个输入捕获寄存器来实现此目的（捕获信号必须为周期性信号，可以由另一个定时器产生）；还可以通过 DMA 请求读取计数器值。

TIMx_CR1 寄存器中的 UIFREMAP 位强制将更新中断标志 (UIF) 连续复制到定时计数器寄存器的位 31 (TIMxCNT[31]) 中。这样便可按原子方式读取计数器值以及由 UIFCPY 标志发出的电位翻转条件。这可避免在后台任务（计数器读）和中断（更新中断）之间共享处理时产生竞争条件，从而简化角速度的计算。

UIF 和 UIFCPY 标志之间的置位没有延迟。

在 32 位定时器实现中，当 UIFREMAP 位置 1 时，计数器的位 31 在读访问时由 UIFCPY 标志覆盖（计数器的最高有效位只能在写模式下访问）。

时钟 + 方向编码器模式

除了正交编码器模式之外，定时器也支持其他类型的编码器。

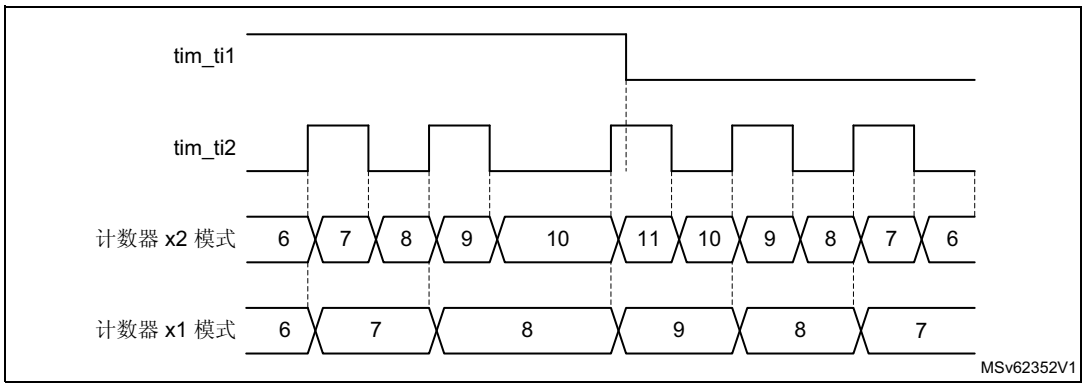
在“时钟 + 方向”模式下（如*图 301* 所示），时钟由 tim_ti2 单线提供，方向则由 tim_ti1 输入强制设定。

该模式通过 TIMx_SMCR 寄存器中的 SMS[3:0] 位域使能，具体如下所述：

- 1010：x2 模式，计数器在时钟的上升沿和下降沿均更新
- 1011：x1 模式，计数器在单个时钟边沿更新，具体哪个边沿取决于 CC2P 位的值：CC2P = 0 对应上升沿有效，CC2P = 1 对应下降沿有效

tim_ti1 上方向信号的极性由 CC1P 位设置：0 对应正极性（tim_ti1 为高电平时递增计数，tim_ti1 为低电平时递减计数），CC1P = 1 对应负极性（tim_ti1 为低电平时递增计数）。

图 301. 方向 + 时钟编码器模式



定向时钟编码器模式

在“定向时钟”模式下（如*图 302* 所示），时钟由两条线路提供，每次只使用一条线路，具体使用哪条线路取决于计数方向，因此具有一条递增计数时钟线和一条递减计数时钟线。

该模式通过 TIMx_SMCR 寄存器中的 SMS[3:0] 位域使能，具体如下所述：

- 1100：x2 模式，计数器在任一时钟线的上升沿和下降沿均更新。CC1P 和 CC2P 位为时钟空闲状态编码。CCxP = 0 对应高电平空闲状态（请参见*图 302*），CCxP = 1 对应低电平空闲状态（请参见*图 303*）。
- 1101：x1 模式，计数器在单个时钟边沿更新，具体哪个边沿取决于 CC1P 位和 CC2P 位的值：CCxP = 0 对应下降沿有效和高电平空闲状态（请参见*图 302*），CCxP = 1 对应上升沿有效和低电平空闲状态（请参见*图 303*）。

图 302. 定向时钟编码器模式（CC1P = CC2P = 0）

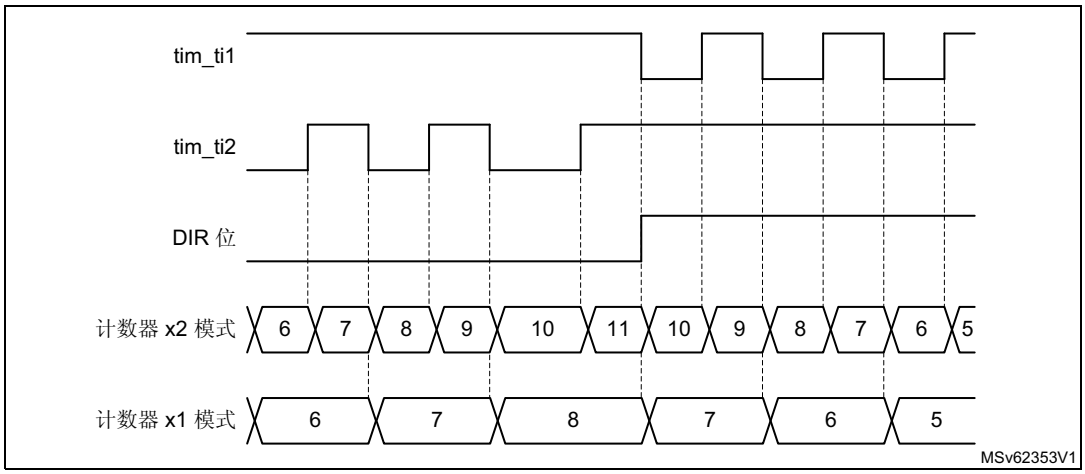
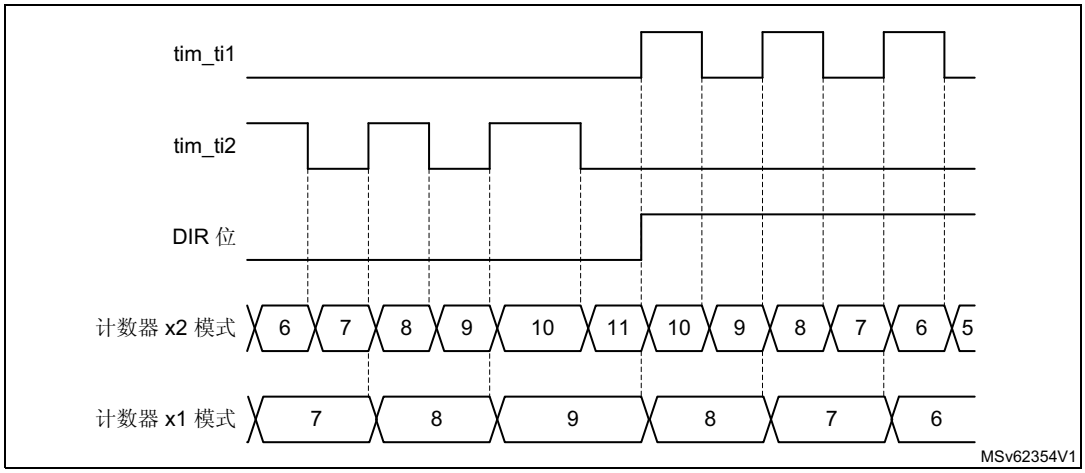


图 303. 定向时钟编码器模式（CC1P = CC2P = 1）



下面的*表 196* 详细介绍了针对任何输入转换，定向时钟模式的工作方式。

表 196. 计数方向与编码器信号和极性设置的关系

定向时钟模式	SMS[3:0]	相反信号的电平（tim_ti1fp1 对应 tim_ti2，tim_ti2fp2 对应 tim_ti1）	tim_ti1fp1 信号		tim_ti2fp2 信号	
			上升	下降	上升	下降
x2 模式 CCxP = 0	1100	高	递减	递减	递增	递增
		低	不计数	不计数	不计数	不计数
x2 模式 CCxP = 1	1100	高	不计数	不计数	不计数	不计数
		低	递减	递减	递增	递增
x1 模式 CCxP = 0	1101	高	不计数	递减	不计数	递增
		低	不计数	不计数	不计数	不计数
x1 模式 CCxP = 1	1101	高	不计数	不计数	不计数	不计数
		低	递减	不计数	递增	不计数

索引输入

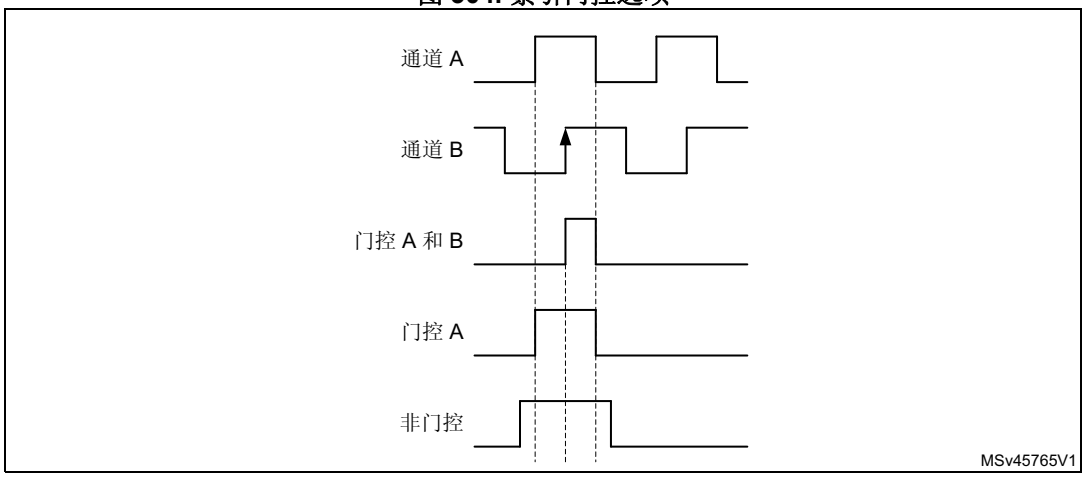
计数器可通过来自编码器的索引信号来复位，该信号指示绝对参考位置。索引信号必须连接到 tim_etr_in 输入。可使用数字输入滤波器对其进行滤波。

索引功能通过 TIMx_ECR 寄存器中的 IE 位来使能。仅当 SMS[3:0] 位域为以下值时，才可以在编码器模式下将 IE 位置 1：0001、0010、011、1010、1011、1100、1101、1110、1111。

商用编码器提供了多种索引脉冲调节选项，如下面的*图 304* 所示：

- 在 A 和 B 上进行门控：脉冲宽度为 1/4 个通道周期，与 A 和 B 边沿均对齐
- 在 A（或 B）上进行门控：脉冲宽度为 1/2 个通道周期，与通道 A（或通道 B）上的两个边沿对齐
- 不进行门控：脉冲宽度最多为一个通道周期，不与任何边沿对齐

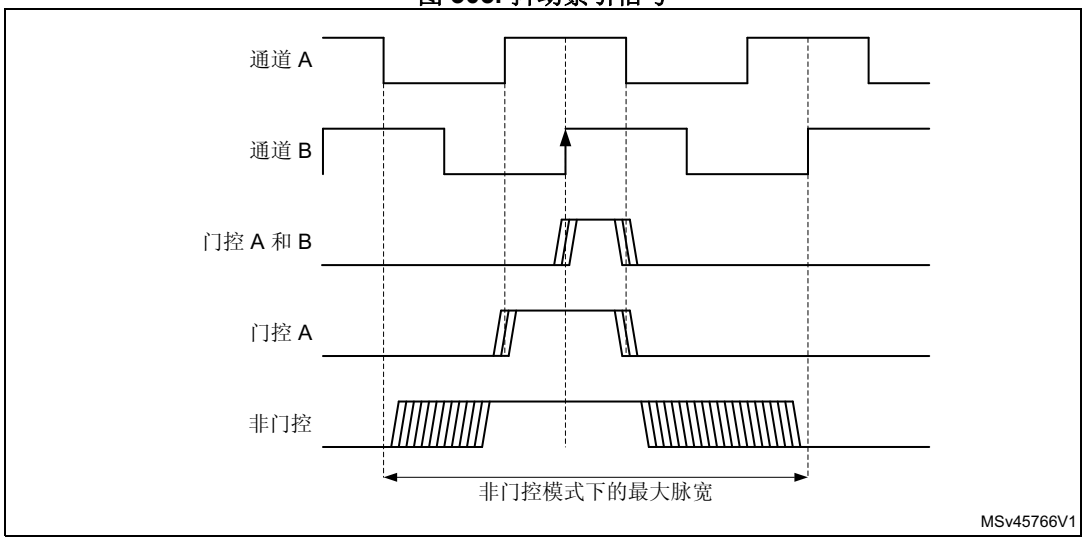
图 304. 索引门控选项



无论采用哪种门控模式，电路都能容忍索引信号的抖动，如下面的*图 305* 所示。

在非门控模式下，信号脉宽必须严格小于 2 个编码器周期。如果脉宽大于或等于 2 个编码器周期，计数器会多次复位。

图 305. 抖动索引信号



定时器平等支持 3 个门控选项，无需任何特定编程。只需使用 TIMx_ECR 寄存器中的 IPOS[1:0] 位域定义索引必须基于哪种编码器状态（即通道 A 和通道 B 状态组合）进行同步。

对于不同的计数方向，索引检测事件的作用会有所不同，从而确保了在速度反转期间能够执行对称操作：

- 计数器在递增计数期间复位（DIR 位 = 0）
- 计数器在递减计数时设置为 TIMx_ARR

因此，无论哪种计数方向，都可以在完全相同的机械角位置上生成索引。下面的*图 306* 通过一个简单的示例说明了生成索引的位置（每次机械旋转编码器提供 4 个边沿）。

图 306. IPOS[1:0] = 11 时的索引生成

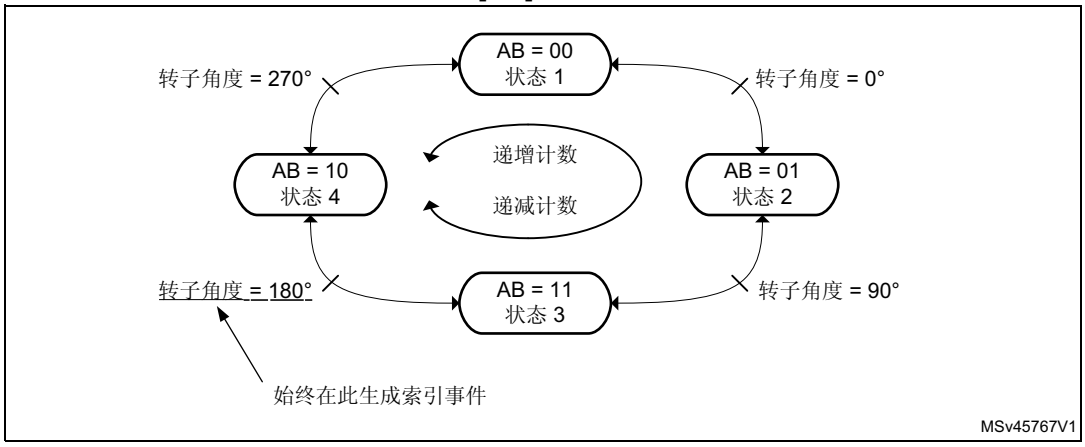


图 307 给出了 IPOS[1:0] = 11 时的波形和相应值。该图说明了计数器值根据计数方向强制自动调整的情况：

- 如果编码器状态为“11”（ChA = 1，ChB = 1）并且递增计数（DIR 位 = 0），则计数器设置为 0。
- 如果编码器退出“11”状态并且递减计数（DIR 位 = 1），则计数器设置为 TIMx_ARR。

可以在发生索引检测事件时发出中断。

箭头指示在哪次转换时生成索引事件中断。

图 307. 在通道 A 上对索引进行门控时的计数器读取 (IPOS[1:0] = 11)

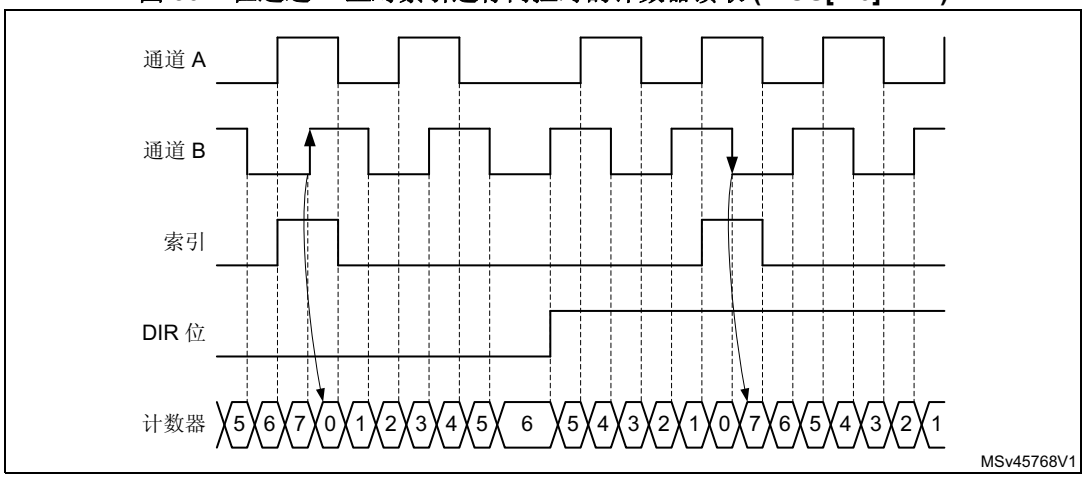
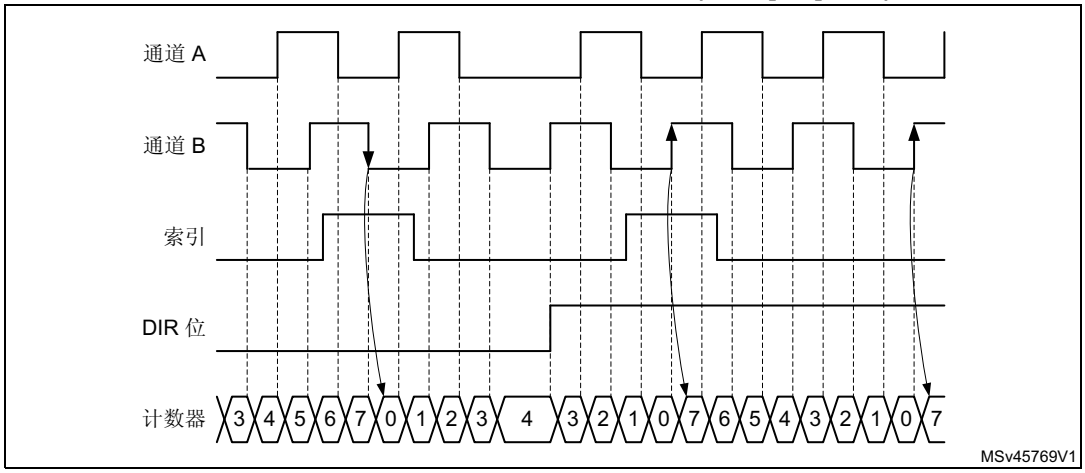


图 308 显示了非门控模式下的波形和相应值。箭头指示在哪次转换时生成索引事件。

图 308. 不对索引进行门控时的计数器读取 (IPOS[1:0] = 00)



下面的*图 309* 说明了在各种脉冲对齐场景下如何处理“在 A 和 B 上进行门控”模式。箭头指示在哪次转换时生成索引事件。

图 309. 在通道 A 和 B 上对索引进行门控时的计数器读取

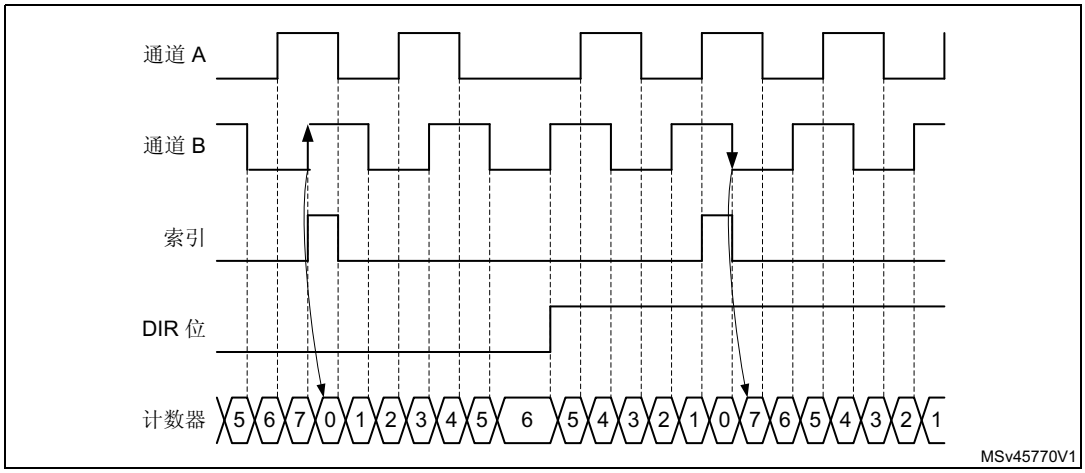


图 310 和*图 311* 详细说明了后续索引脉冲的宽度可能小于 1/4 个编码器时钟周期的情况。

图 310. 窄索引脉冲情况下的编码器模式行为 (IPOS[1:0] = 11)

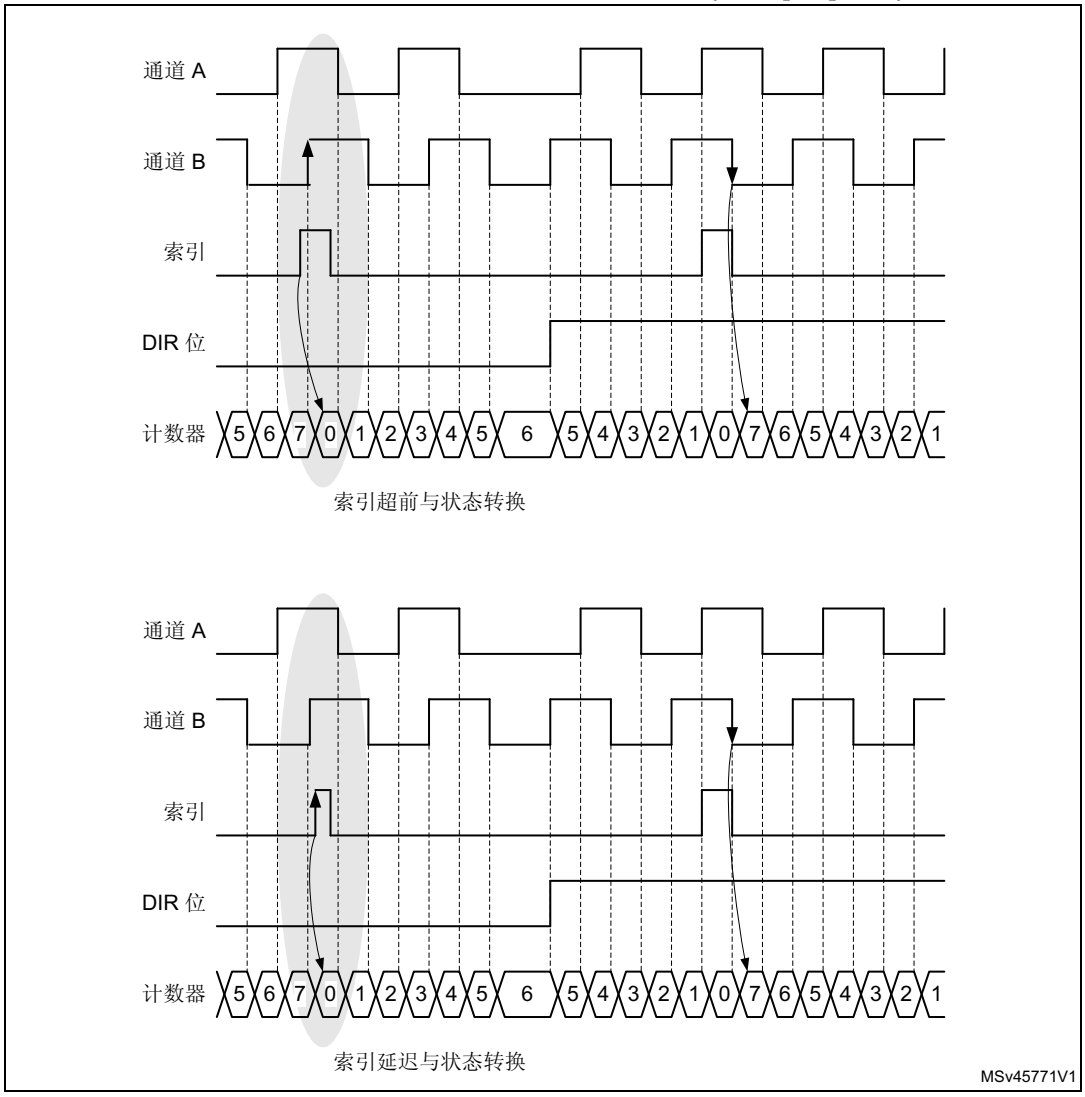


图 311. 计数器复位窄索引脉冲（放大视图，ARR = 0x07）

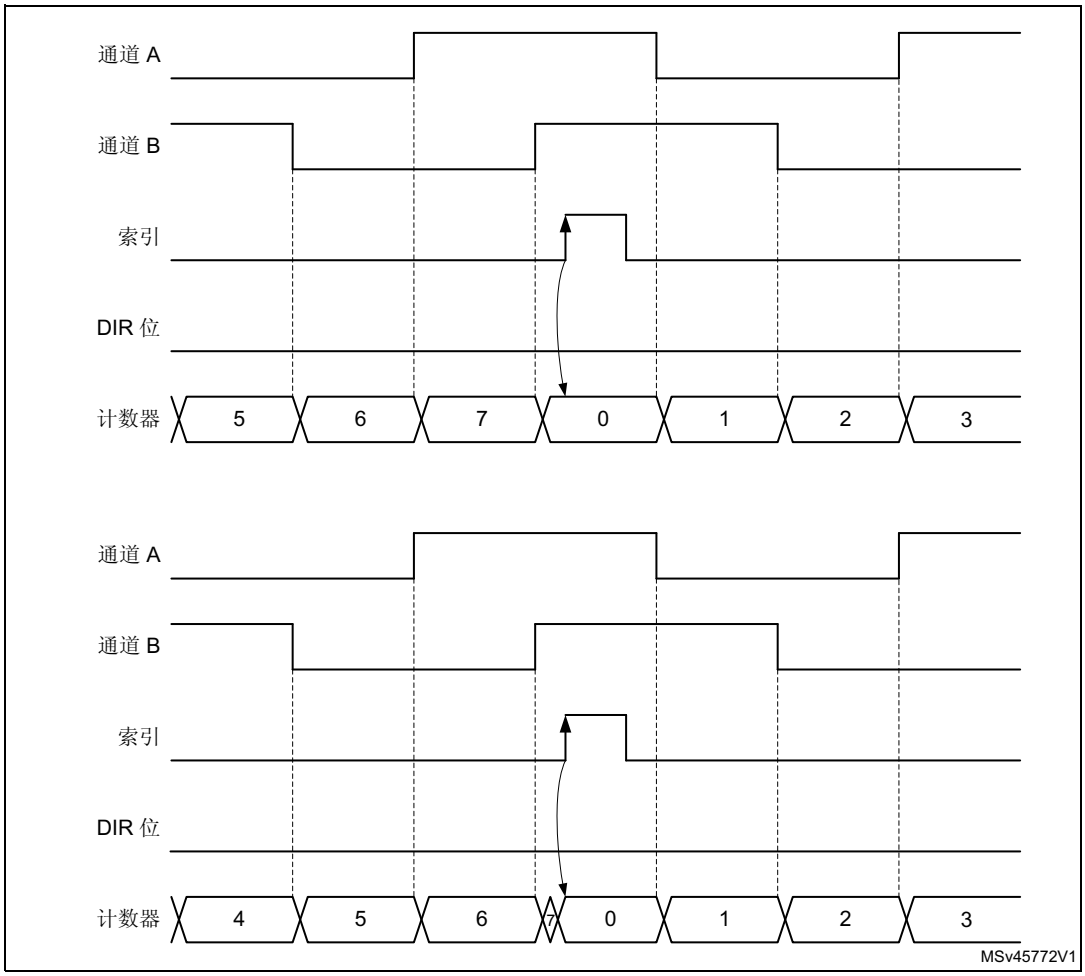
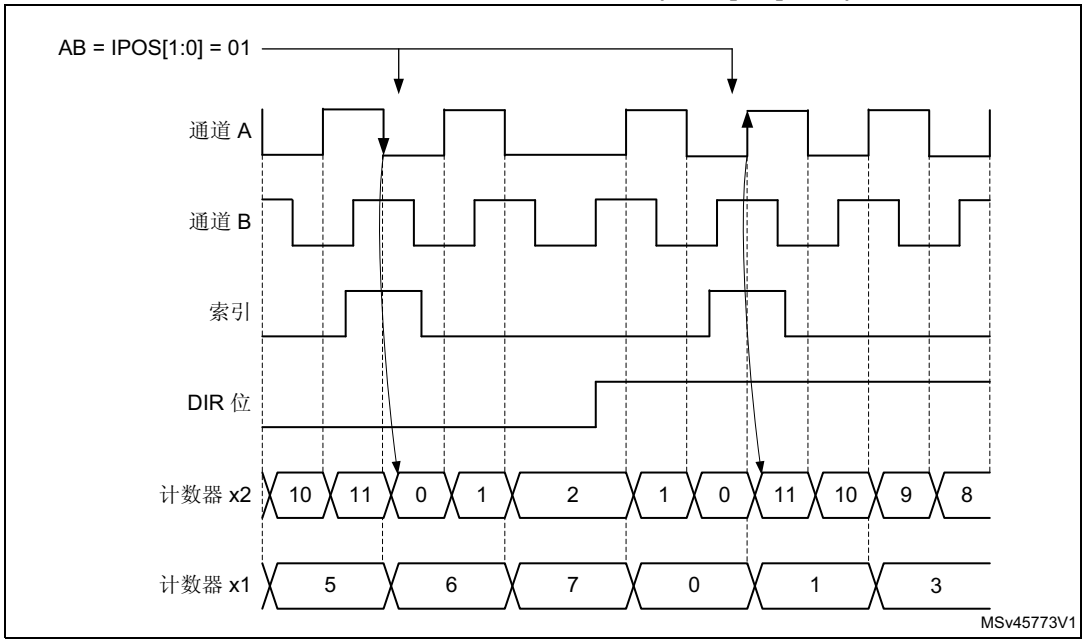


图 312 说明了如何在 x1 和 x2 模式下管理索引。

图 312. x1 和 x2 模式下的索引行为 (IPOS[1:0] = 01)

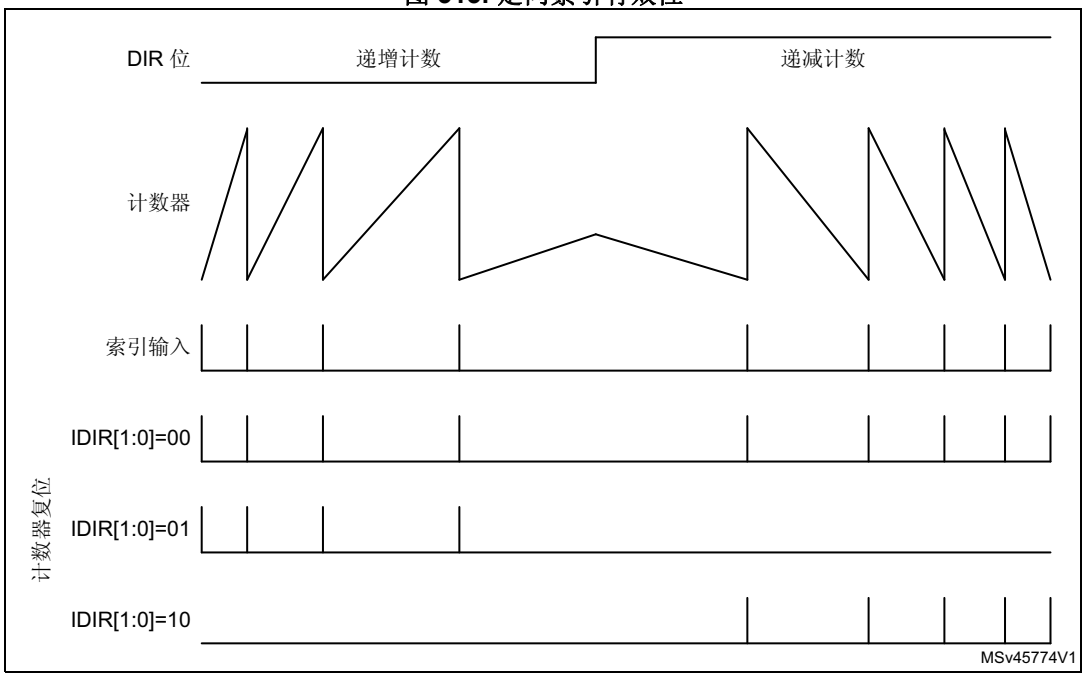


定向索引有效性

TIMx_ECR 寄存器中的 IDIR[1:0] 位域允许索引仅在选定的计数方向上有效。

图 313 显示了索引与计数器复位事件之间的关系，具体取决于 IDIR[1:0] 值。

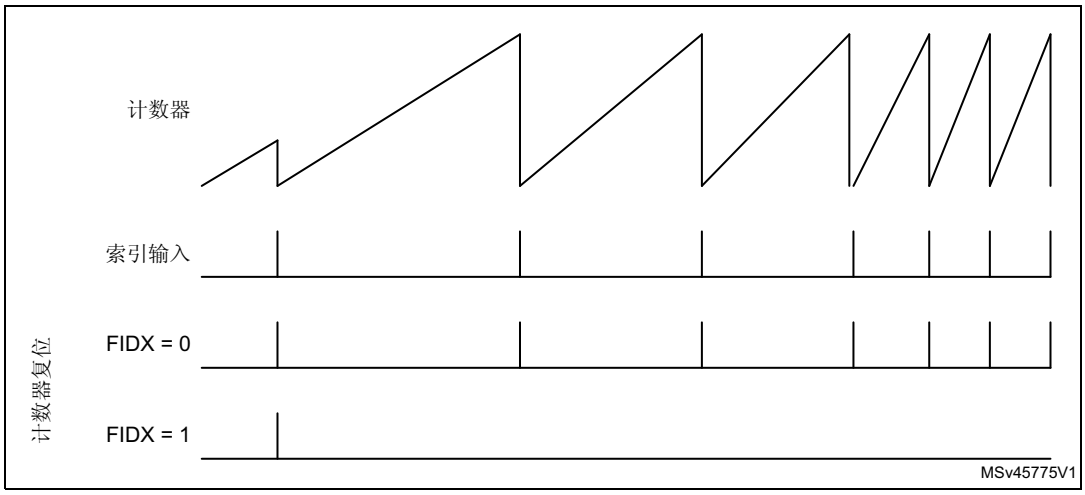
图 313. 定向索引有效性



特殊的第一个索引事件管理

TIMx_ECR 寄存器中的 FIDX 位允许仅获取一次索引，如下面的*图 314* 所示。第一个索引到达后，任何后续索引都将被忽略。如果需要，可以通过向 FIDX 位写入 0 然后再置 1 的方式重新启动电路。

图 314. 计数器复位与 FIDX 位置 1 的关系



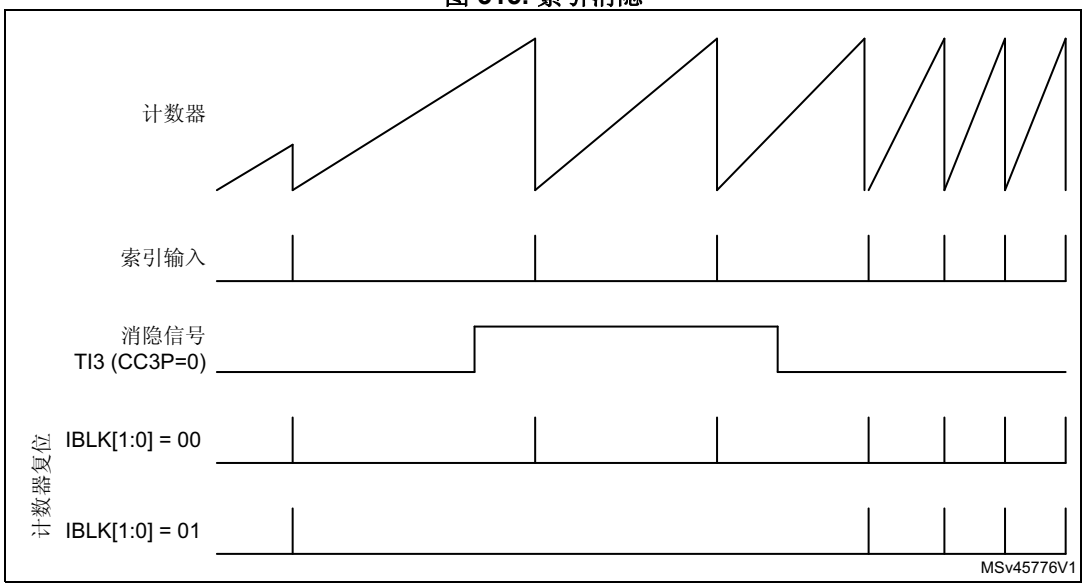
索引消隐

可使用 tim_ti3 或 tim_ti4 输入来消隐索引事件。在消隐窗口期间，索引事件不再复位计数器，如下面的*图 315* 所示。

该模式通过 TIMx_ECR 寄存器中的 IBLK[1:0] 位域使能，具体如下所述：

- IBLK[1:0] = 00：索引信号始终有效
- IBLK[1:0] = 01：tim_ti3 输入上的索引信号消隐
- IBLK[1:0] = 10：tim_ti4 输入上的索引信号消隐

图 315. 索引消隐



非正交模式下的索引管理

下面的*图 316* 和*图 317* 详细介绍了当 SMS[3:0] 位域等于 1010、1011、1100 和 1101 时，在定向时钟模式和时钟 + 方向模式下如何管理索引。

对于这两种模式，索引有效性通过 IPOS[0] 位设置，具体如下所述：

- IPOS[0] = 0：在时钟低电平时检测索引
- IPOS[0] = 1：在时钟高电平时检测索引

IPOS[1] 位无意义。

图 316. 时钟 + 方向模式下的索引行为 (IPOS[0] = 1)

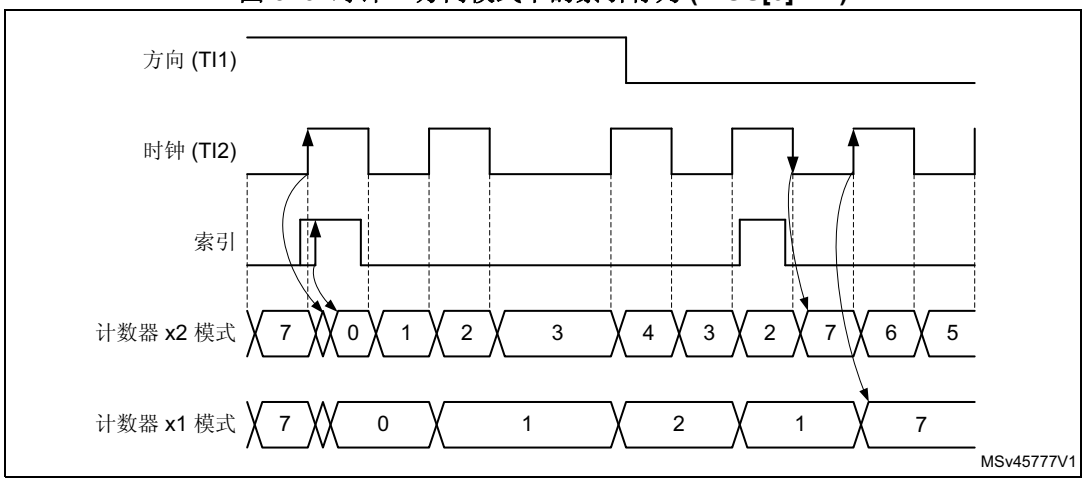
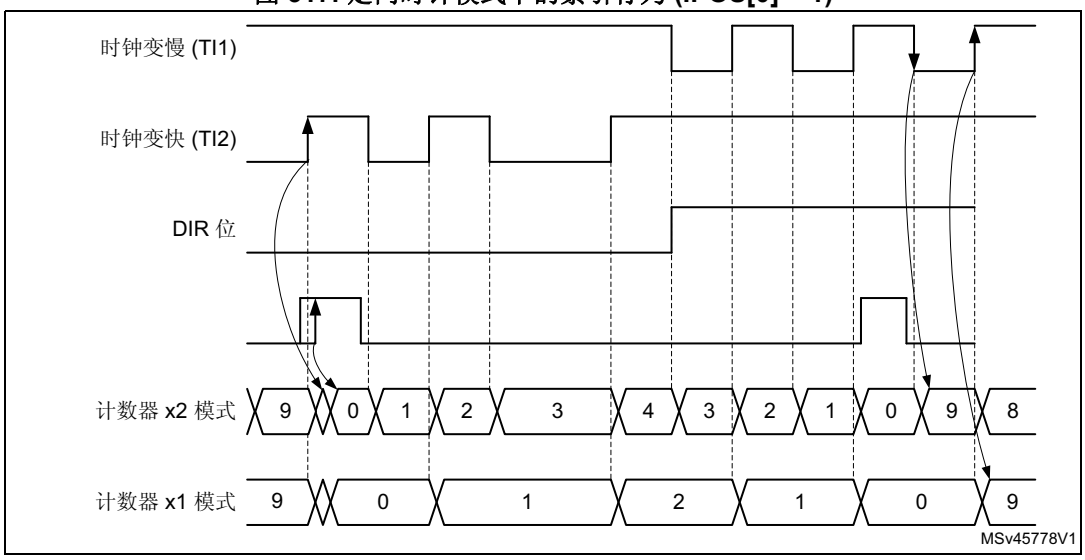


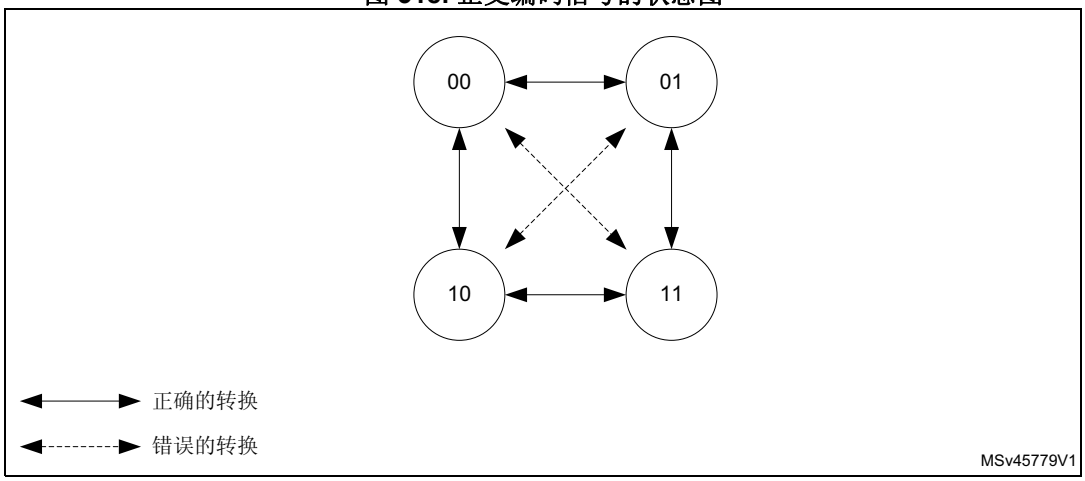
图 317. 定向时钟模式下的索引行为 (IPOS[0] = 1)



编码器错误管理

在使用 2 个正交信号的编码器配置下，可以检测转换错误。对 2 个输入的读操作对应于 2 位格雷码，可表示为状态图（如下面的*图 318.* 所示）。单个位会立即改变。错误的转换会将 TIMx_SR 状态寄存器中的 TERRF 中断标志置 1。如果 TIMx_DIER 寄存器中的 TERRIE 位置 1，将生成转换错误中断。

图 318. 正交编码信号的状态图



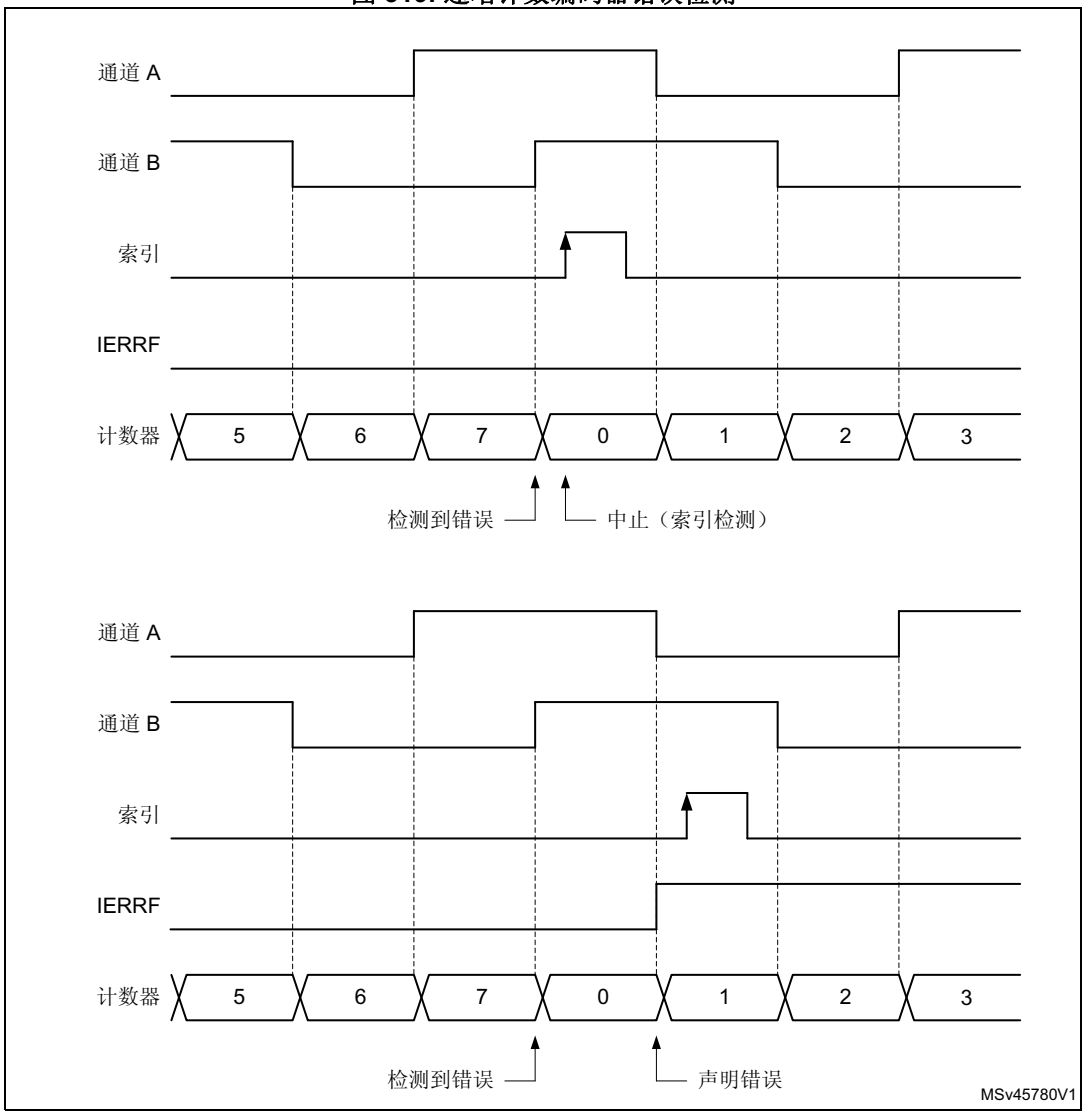
对于具有索引信号的编码器，可以检测会导致每转脉冲数过多的异常操作。每转 N 个脉冲的编码器每转可提供 4xN 个计数。索引信号每 4xN 个时钟周期复位一次计数器。

如果在没有发生任何索引事件的情况下计数器值从 TIMx_ARR 递增到 0 或从 0 递减到 TIMxARR 值，将报告索引位置错误。

上溢阈值使用 TIMx_ARR 寄存器编程设定。1000 线编码器生成的计数器值介于 0 与 3999 之间（在 4x 读取模式下）。上溢检测阈值必须通过设置 TIMx_ARR = 3999 + 1 = 4000 进行编程。

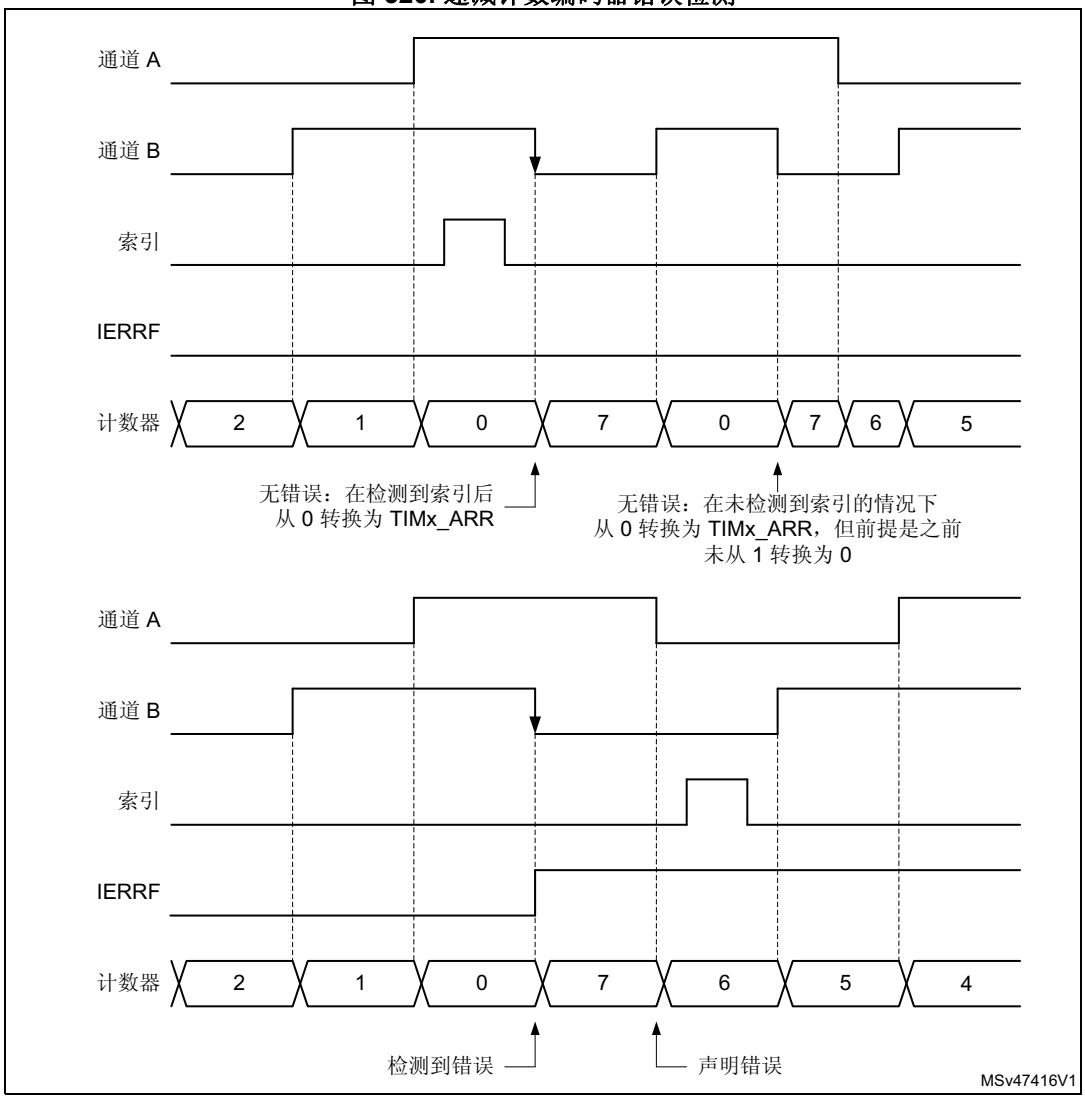
递增计数时，错误标志将延迟到从 0 到 1 的转换后再置 1。这是为了应对 A 和 B 门控模式下的窄索引脉冲，如下面的*图 319* 所示。

图 319. 递增计数编码器错误检测



在递减计数模式下，检测的触发条件是初次从 1 到 0 的转换。这是为了应对 A 和 B 门控模式下的窄索引脉冲（如下面的*图 320* 所示），以避免在检测到索引后编码器立即在 TIMx_ARR 与 0 之间抖动时发生误检。

图 320. 递减计数编码器错误检测



索引错误会将 TIMx_SR 状态寄存器中的 IERRF 中断标志置 1。如果 TIMx_DIER 寄存器中的 IERRIE 位置 1，将生成索引错误中断。

功能编码器中断

以下中断也可用于编码器模式

- 方向更改：编码器模式下计数方向的任何更改都会导致 TIMx_CR1 寄存器中的 DIR 位翻转。方向更改会将 TIMx_SR 状态寄存器中的 DIRF 中断标志置 1。如果 TIMx_DIER 寄存器中的 DIRIE 位置 1，将生成方向更改中断。
- 索引事件：索引事件会将 TIMx_SR 状态寄存器中的 IDXF 中断标志置 1。如果 TIMx_DIER 寄存器中的 IDXIE 位置 1，将生成索引中断。

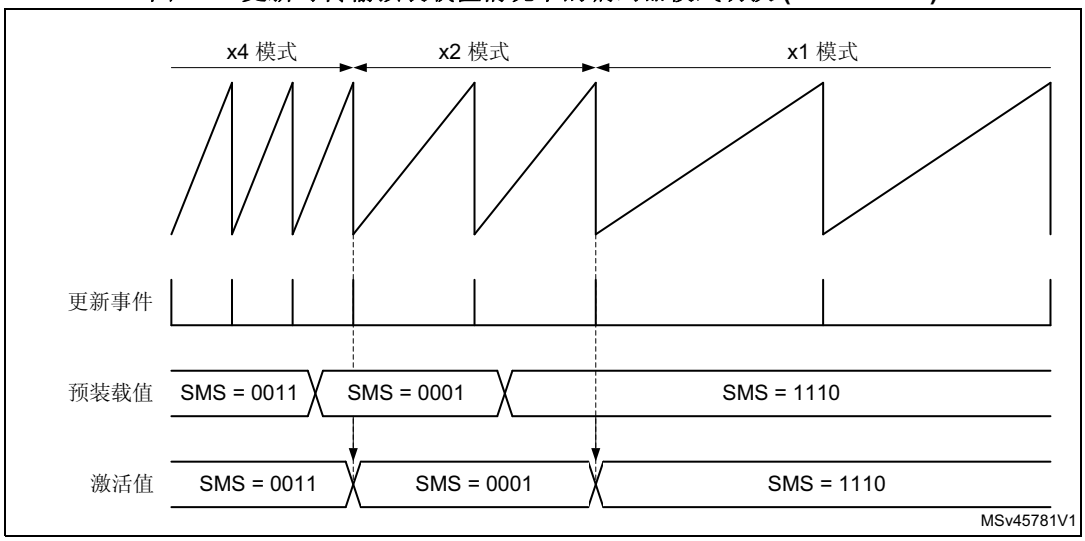
运行时编码器模式更新情况下的从模式选择预装载

在运行过程中，有时可能需要从一种编码器模式切换到另一种编码器模式。这通常发生在高速情况下，用于降低更新中断频率，先从 x4 模式切换到 x2 模式再切换到 x1 模式，如下面的*图 321* 所示。

为此，可以预装载 SMS[3:0] 位。该功能通过将 TIMx_SMCR 寄存器中的 SMSPE 使能位置 1 来使能。使用 TIMx_SMCR 寄存器中的 SMSPS 位，可以选择将 SMS[3:0] 预装载值传输到有效值的触发信号。

- SMSPS = 0：传输由计数器递增计数至上溢或递减计数至下溢时发生的更新事件 (UEV) 触发。
- SMSPS = 1：传输由索引事件触发。

图 321. 更新时传输预装载值情况下的编码器模式切换 (SMSPS = 0)



编码器时钟输出

编码器模式的工作原理并不是特别适合低速下的高分辨率速度测量，因为它需要相对较长的积分时间才能具有足够数量的时钟边沿并获取精确的测量结果。

在低速条件下，更好的解决方案是进行边沿到边沿的时钟周期测量。这可以使用从定时器来实现。定时器可通过 tim_trgo 输出来输出编码器时钟信息。之后，从定时器便可执行周期测量并提供每个编码器时钟边沿的速度信息。

该模式通过将 TIMx_CR2 寄存器中的 MMS[3:0] 位域设置为 1000 来使能，适用于以下 SMS[3:0] 值：0001、0010、0011、1010、1011、1100、1101、1110、1111。不允许使用任何其他 SMS[3:0] 代码，否则可能会导致意外行为。

27.4.19 方向位输出

定时器可通过 tim_oc3 和 tim_oc4 输出信号（TIMx_CR1 寄存器中的 DIR 位的副本）来输出方向信号。这通过将 TIMx_CCMR2 寄存器中的 OC3M[3:0] 或 OC4M[3:0] 位域设置为 1011 来实现。

该功能可用于在编码器模式下监视计数方向（或旋转方向），或者用于在中心对齐 PWM 模式下获得指示递增/递减计数阶段的信号。

27.4.20 UIF 位重映射

TIMx_CR1 寄存器中的 UIFREMAP 位强制将更新中断标志 (UIF) 连续复制到定时计数器寄存器的位 31 (TIMxCNT[31]) 中。该功能用于按原子方式读取计数器值以及由 UIFCPY 标志发出的更新事件情况。这可避免在后台任务（计数器读）和中断（更新中断）之间共享处理时产生竞争条件，从而简化角速度的计算。

UIF 和 UIFCPY 标志之间的置位没有延迟。

在 32 位定时器实现中，当 UIFREMAP 位置 1 时，计数器的位 31 在读访问时由 UIFCPY 标志覆盖（计数器的最高有效位只能在写模式下访问）。

27.4.21 定时器输入异或功能

通过 TIM1xx_CR2 寄存器中的 TI1S 位，可将通道 1 的输入滤波器连接到异或门的输出，从而将 tim_ti1、tim_ti2 和 tim_ti3 这三个输入引脚组合在一起。

异或输出可与触发或输入捕获等所有定时器输入功能配合使用。

*第 763 页的第 26.3.29 节：连接霍尔传感器*以连接霍尔传感器为例介绍了此功能。

27.4.22 定时器与外部触发同步

TIMx 定时器可在多种模式下与外部触发实现同步：复位模式、门控模式、触发模式、复位 + 触发模式和门控 + 复位模式。

从模式：复位模式

当触发输入信号发生变化时，计数器及其预分频器可重新初始化。此外，如果 TIMx_CR1 寄存器中的 URS 位处于低电平，则会生成更新事件 UEV。然后，所有预装载寄存器（TIMx_ARR 和 TIMx_CCRx）都将更新。

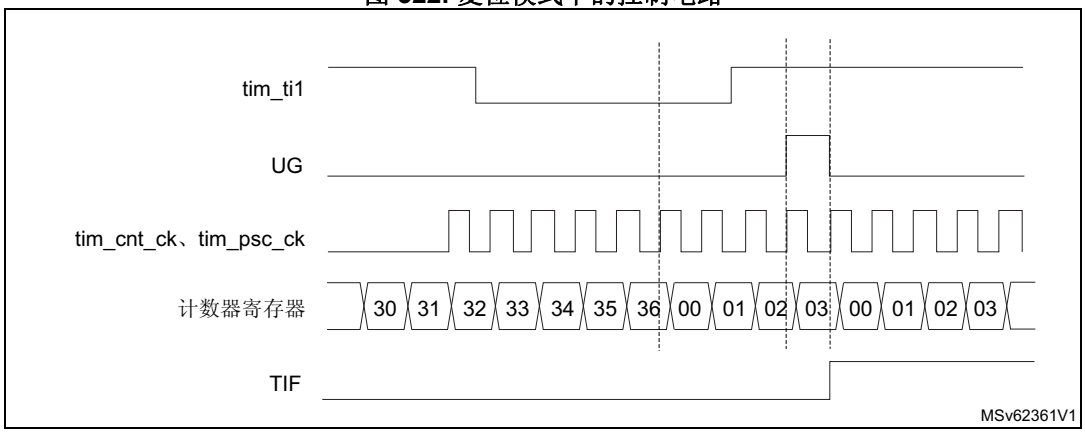
在以下示例中，tim_ti1 输入上出现上升沿时，递增计数器清零：

1. 将通道 1 配置为检测 tim_ti1 上的上升沿。配置输入滤波时间（本例中不需要任何滤波器，因此保持 IC1F=0000）。由于捕获预分频器不用于触发操作，因此无需对其进行配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中的 CC1S = 01。在 TIMx_CCER 寄存器中写入 CC1P=0 和 CC1NP=0，验证极性（仅检测上升沿）。
2. 在 TIMx_SMCR 寄存器中写入 SMS=100，将定时器配置为复位模式。在 TIMx_SMCR 寄存器中写入 TS=00101，选择 tim_ti1 作为输入源。
3. 在 TIMx_CR1 寄存器中写入 CEN=1，启动计数器。

计数器开始根据内部时钟计数，然后正常运转，直到出现 tim_ti1 上升沿。当 tim_ti1 出现上升沿时，计数器清零，然后重新从 0 开始计数。同时，触发标志（TIMx_SR 寄存器中的 TIF 位）置 1，使能中断或 DMA 后，还可发送中断或 DMA 请求（取决于 TIMx_DIER 寄存器中的 TIE 和 TDE 位）。

下图显示了自动重载寄存器 TIMx_ARR=0x36 时的相关行为。tim_ti1 的上升沿与实际计数器复位之间的延迟是由于 tim_ti1 输入的重新同步电路引起的。

图 322. 复位模式下的控制电路



从模式：门控模式

输入信号的电平可用来使能计数器。

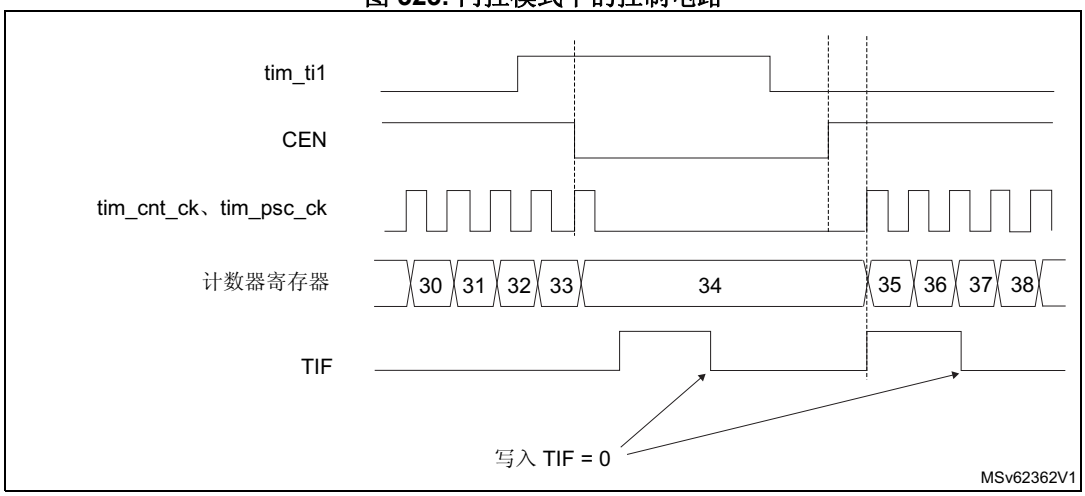
在以下示例中，递增计数器仅在 tim_ti1 输入为低电平时计数：

1. 将通道 1 配置为检测 tim_ti1 上的低电平。配置输入滤波时间（本例中不需要任何滤波器，因此保持 IC1F=0000）。由于捕获预分频器不用于触发操作，因此无需对其进行配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中的 CC1S=01。在 TIMx_CCER 寄存器中写入 CC1P=1 和 CC1NP=0，以确定极性（仅检测低电平）。
2. 在 TIMx_SMCR 寄存器中写入 SMS=101，将定时器配置为门控模式。在 TIMx_SMCR 寄存器中写入 TS=00101，选择 tim_ti1 作为输入源。
3. 在 TIMx_CR1 寄存器中写入 CEN=1，使能计数器（在门控模式下，如果 CEN=0，则无论触发输入电平如何，计数器都不启动）。

只要 tim_ti1 为低电平，计数器就开始根据内部时钟计数，直到 tim_ti1 变为高电平时停止计数。计数器启动或停止时，TIMx_SR 寄存器中的 TIF 标志都会置 1。

tim_ti1 的上升沿与实际计数器停止之间的延迟是由于 tim_ti1 输入的重新同步电路引起的。

图 323. 门控模式下的控制电路



注意： *由于门控模式作用于电平而非边沿，因此在门控模式下，“CCxP=CCxNP=1”（同时检测上升沿和下降沿）不发挥任何作用。*

从模式：触发模式

所选输入上发生某一事件时可以启动计数器。

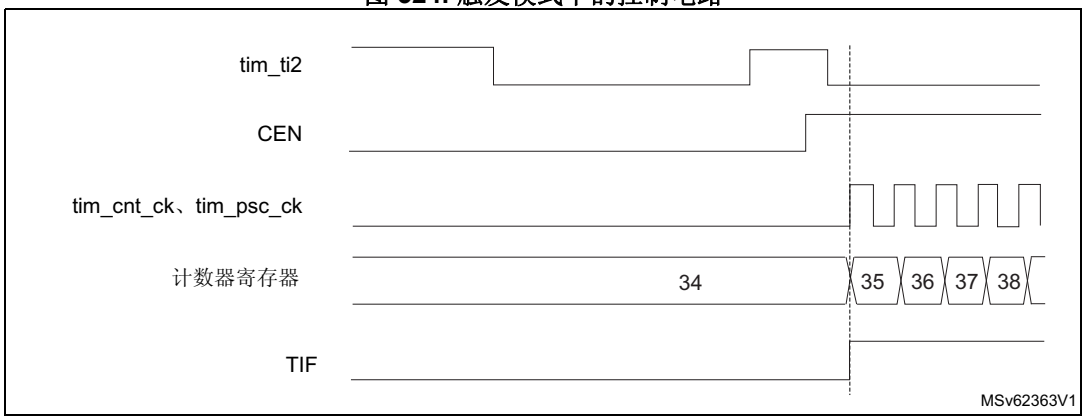
在以下示例中，tim_ti2 输入上出现上升沿时，递增计数器启动：

1. 将通道 2 配置为检测 tim_ti2 上的上升沿。配置输入滤波时间（本例中不需要任何滤波器，因此保持 IC2F=0000）。由于捕获预分频器不用于触发操作，因此无需对其进行配置。CC2S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中的 CC2S=01。在 TIMx_CCER 寄存器中写入 CC1P=0 和 CC1NP=0，以确定极性（仅检测高电平）。
2. 在 TIMx_SMCR 寄存器中写入 SMS=110，将定时器配置为触发模式。在 TIMx_SMCR 寄存器中写入 TS=00110，选择 tim_ti2 作为输入源。

当 tim_ti2 出现上升沿时，计数器开始根据内部时钟计数，并且 TIF 标志置 1。

tim_ti2 的上升沿与实际计数器启动之间的延迟是由于 tim_ti2 输入的重新同步电路引起的。

图 324. 触发模式下的控制电路



运行时编码器模式更新情况下的从模式选择预装载

可以预装载 SMS[3:0] 位。该功能通过将 TIMx_SMCR 寄存器中的 SMSPE 使能位置 1 来使能。计数器发生上溢时产生的更新事件 (UEV) 会触发将 SMS[3:0] 预装载值传输到有效值。

从模式——组合复位 + 触发模式

在这种情况下，在出现所选触发输入 (tim_trgi) 上升沿时，重新初始化计数器，生成一个寄存器更新事件，并启动计数器。

该模式用于单脉冲模式。

从模式——组合门控 + 复位模式

触发输入 (tim_trgi) 为高电平时使能计数器时钟。只要触发输入变为低电平，计数器立即停止计数并复位。计数器的启动和停止都被控制。

该模式用于检测超范围 PWM 信号（占空比超过最大预期值）。

从模式——外部时钟模式 2 + 触发模式

外部时钟模式 2 可与另一种从模式（外部时钟模式 1 和编码器模式除外）结合使用。这种情况下，tim_etr_in 信号用作外部时钟输入，在复位模式、门控模式或触发模式下工作时，可选择另一个输入作为触发输入。不建议通过 TIMx_SMCR 寄存器中的 TS 位来选择 tim_etr_in 作为 tim_trgi。

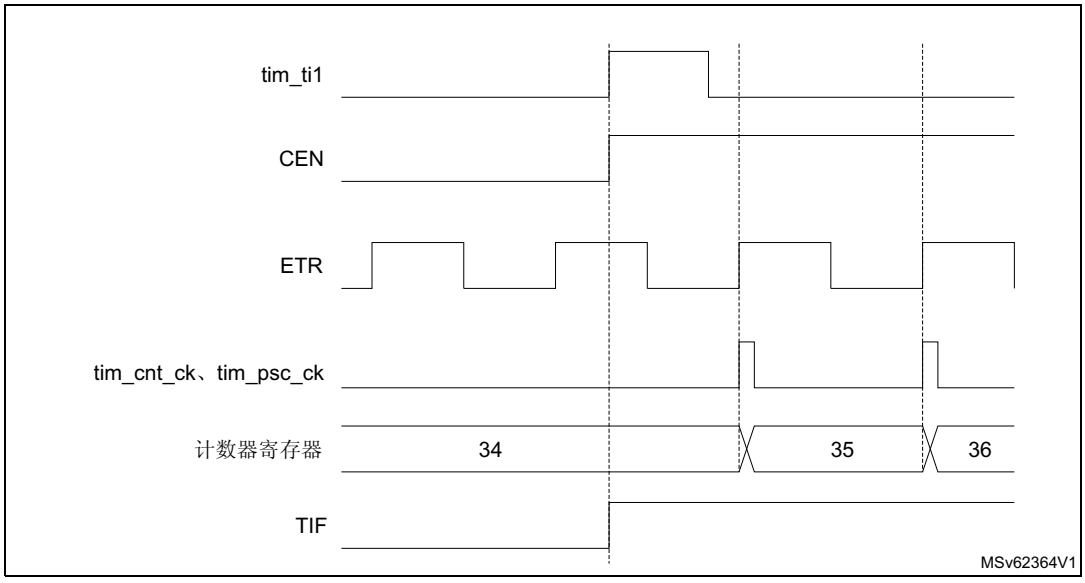
在以下示例中，只要 tim_ti1 出现上升沿，递增计数器即会在 tim_etr_in 信号的每个上升沿处递增：

1. 通过对 TIMx_SMCR 寄存器进行如下编程，配置外部触发输入电路：
   - ETF = 0000：无滤波器
   - ETPS = 00：禁止预分频器
   - ETP = 0：检测 tim_etr_in 的上升沿，并写入 ECE=1，以使能外部时钟模式 2。
2. 如下配置通道 1，以检测 TI 的上升沿：
   - IC1F=0000：无滤波器。
   - 由于捕获预分频器不用于触发操作，因此无需对其进行配置。
   - TIMx_CCMR1 寄存器中 CC1S=01，只选择输入捕获源。
   - TIMx_CCER 寄存器中 CC1P=0 且 CC1NP= 0，以确定极性（仅检测上升沿）。
3. 在 TIMx_SMCR 寄存器中写入 SMS=110，将定时器配置为触发模式。在 TIMx_SMCR 寄存器中写入 TS=00101，选择 tim_ti1 作为输入源。

tim_ti1 出现上升沿时将使能计数器并且 TIF 标志置 1。然后计数器在 tim_etr_in 出现上升沿时计数。

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

图 325. 外部时钟模式 2 + 触发模式下的控制电路



27.4.23 定时器同步

TIMx 定时器从内部连接在一起，以实现定时器同步或链接。当某个定时器配置为主模式时，可对另一个配置为从模式的定时器的计数器执行复位、启动、停止操作或为其提供时钟。

图 326 和*图 327* 显示了主/从定时器的连接示例。

图 326. 主/从定时器示例

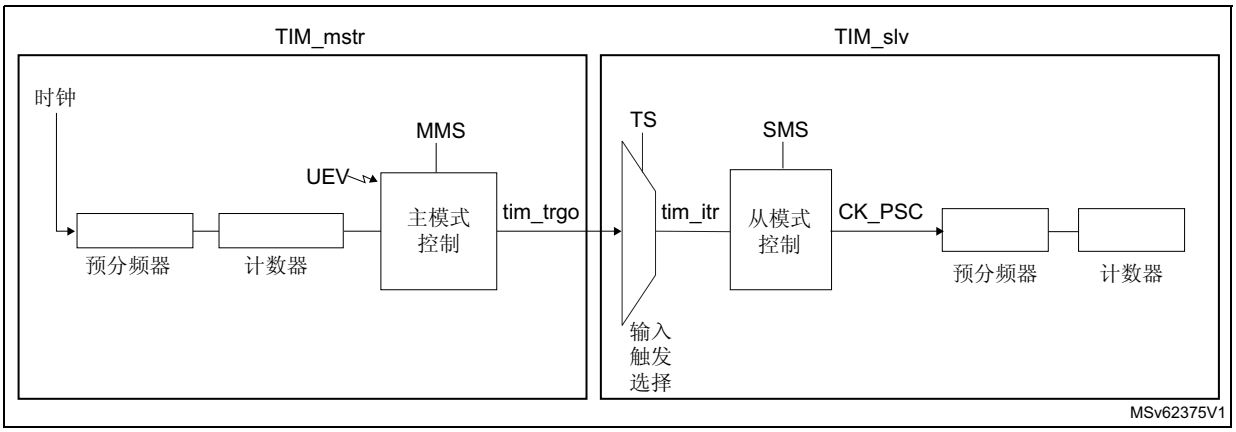
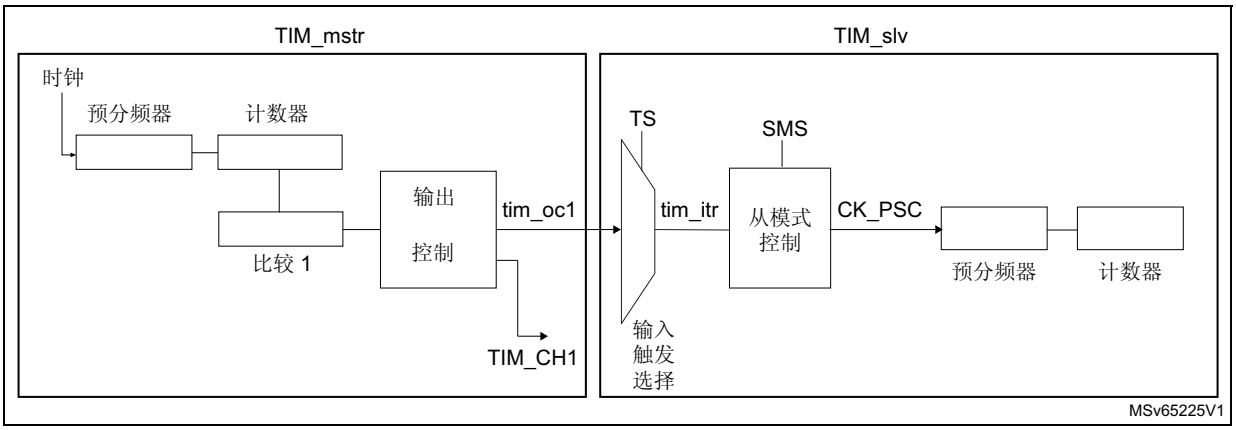


图 327. 单通道定时器的主/从连接示例



注意： *单通道定时器（请参见图 327）没有主模式。不过，tim_oc1 输出信号可用作从定时器的触发信号（请参见第 27.4.2 节：TIM2/TIM3 引脚和内部信号中的 TIMx 内部触发连接表）。*
tim_oc1 信号的脉冲宽度必须至少设为目标定时器时钟周期的 2 倍，以便确保从定时器能够检测到触发。
例如，如果目标定时器 tim_ker_ck 时钟频率是源定时器的 1/4，则 OC1 的脉冲宽度必须至少是 8 个源定时器时钟周期。

将一个定时器用作另一个定时器的预分频器

例如，可以将 TIM_mstr 配置为 TIM_slv 的预分频器。请参见*图 326*。为此：

1. 将 TIM_mstr 配置为主模式，以便每次发生更新事件 UEV 时都输出一个周期性触发信号。如果在 TIM_mstr_CR2 寄存器中写入 MMS=010，则每次生成更新事件时，tim_trgo 都会输出一个上升沿。
2. 要将 TIM_mstr 的 tim_trgo 输出连接到 TIM_slv，必须将 TIM_slv 配置为从模式，使用 ITR2 作为内部触发。通过 TIM_slv_SMCR 寄存器中的 TS 位（写入 TS=00010）可对此进行选择。
3. 然后必须将从模式控制器设为外部时钟模式 1（在 TIM_slv_SMCR 寄存器中写入 SMS=111）。这样一来，TIM_slv 的时钟将由 TIM_mstr 周期性触发信号的上升沿（与 TIM_mstr 的计数器上溢对应）提供。
4. 最后必须通过将这两个定时器的相应 CEN 位（TIMx_CR1 寄存器）置 1 同时使能二者。

注意： *如果选择 TIM_mstr 的 tim_ocx 信号作为触发输出 (MMS=1xx)，该信号的上升沿将用于驱动 TIM_slv 的计数器。*

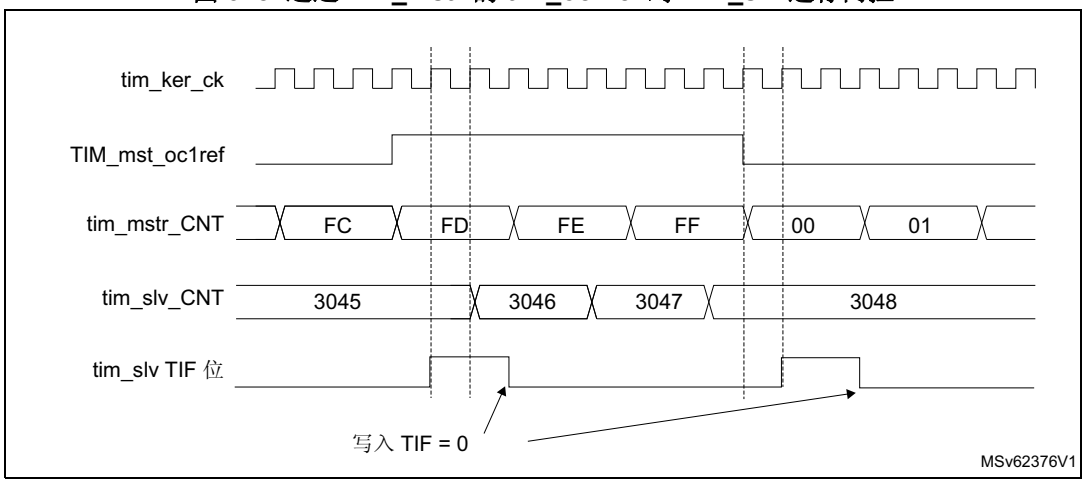
使用一个定时器使能另一个定时器

本例中通过定时器 TIM_mstr 的输出比较 1 来使能 TIM_slv。相关连接图，请参见*图 326*。仅当 TIM_mstr 的 tim_oc1ref 为高电平时，TIM_slv 才根据分频后的内部时钟进行计数。两个计数器的时钟频率都基于 tim_ker_ck 通过预分频器执行 3 分频 ($f_{tim_cnt_ck} = f_{tim_ker_ck}/3$)。

1. 将 TIM_mstr 配置为主模式，发送其输出比较 1 参考信号 (tim_oc1ref) 作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=100）。
2. 配置 TIM_mstr 的 tim_oc1ref 波形（TIM_mstr_CCMR1 寄存器）。
3. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=00010）。
4. 将 TIM_slv 配置为门控模式（TIM_slv_SMCR 寄存器中的 SMS=101）。
5. 通过向 CEN 位（TIM_slv_CR1 寄存器）写入“1”使能 TIM_slv。
6. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“1”启动 TIM_mstr。

注意： *从定时器计数器的时钟与主定时器计数器的时钟不同步，该模式仅影响 TIM_slv 的计数器使能信号。*

图 328. 通过 TIM_mstr 的 tim_oc1ref 对 TIM_slv 进行门控

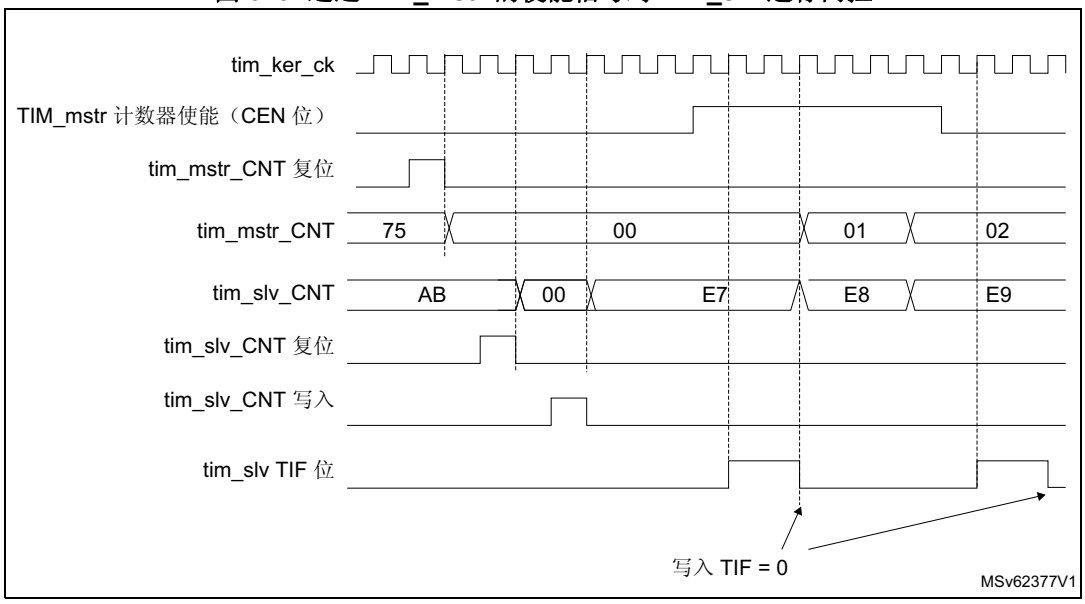


在*图 328* 的示例中，TIM_slv 的计数器和预分频器在启动前未进行初始化。因此从各自的当前值开始计数。启动定时器 TIM_mstr 之前，通过复位这两个定时器可以从指定值开始计数。然后便可以在定时器计数器中写入任意值。两个定时器都可通过软件使用 TIMx_EGR 寄存器中的 UG 位轻松复位。

在下一示例中（请参见*图 329*），TIM_mstr 与 TIM_slv 同步。TIM_mstr 为主模式，从 0 开始计数。TIM_slv 为从模式，从 0xE7 开始计数。两个定时器的预分频比相同。在 TIM_mstr_CR1 寄存器中通过向 CEN 位写入“0”来禁止 TIM_mstr 时，TIM_slv 将停止：

1. 将 TIM_mstr 配置为主模式，发送其输出比较 1 参考信号 (tim_oc1ref) 作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=100）。
2. 配置 TIM_mstr 的 tim_oc1ref 波形（TIM_mstr_CCMR1 寄存器）。
3. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=00010）。
4. 将 TIM_slv 配置为门控模式（TIM_slv_SMCR 寄存器中的 SMS=101）。
5. 通过向 UG 位（TIM_mstr_EGR 寄存器）写入“1”复位 TIM_mstr。
6. 通过向 UG 位（TIM_slv_EGR 寄存器）写入“1”复位 TIM_slv。
7. 通过在 TIM_slv 的计数器 (tim_slv_CNT) 中写入“0xE7”使 TIM_slv 初始化为 0xE7。
8. 通过向 CEN 位（TIM_slv_CR1 寄存器）写入“1”使能 TIM_slv。
9. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“1”启动 TIM_mstr。
10. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“0”停止 TIM_mstr。

图 329. 通过 TIM_mstr 的使能信号对 TIM_slv 进行门控

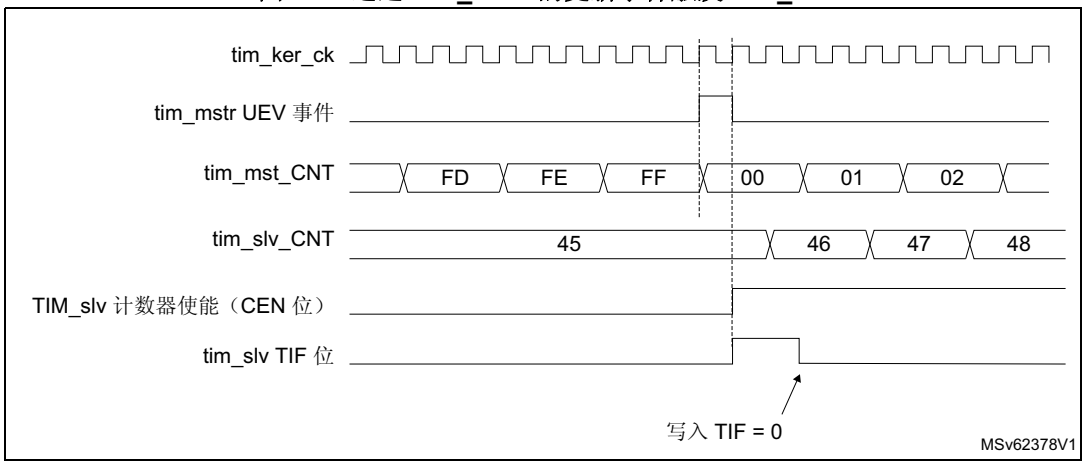


使用一个定时器启动另一个定时器

本例中，使用 TIM_mstr 的更新事件使能 TIM_slv。相关连接图，请参见*图 326*。只要 TIM_mstr 生成更新事件，TIM_slv 便根据分频后的内部时钟从当前值（可以不为 0）开始计数。TIM_slv 接收到触发信号时，其 CEN 位自动置 1，并且计数器开始计数，直到向 TIM_slv_CR1 寄存器的 CEN 位写入“0”后停止计数。两个计数器的时钟频率都基于 tim_ker_ck 通过预分频器执行 3 分频 ($f_{tim_cnt_ck} = f_{tim_ker_ck}/3$)。

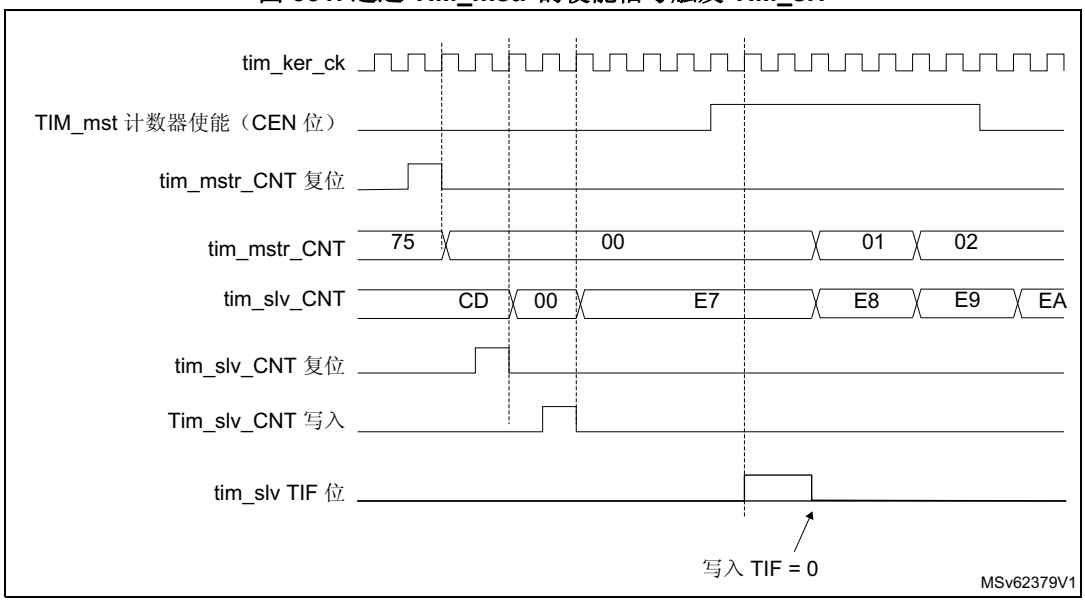
1. 将 TIM_mstr 配置为主模式，发送其更新事件 (UEV) 作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=010）。
2. 配置 TIM_mstr 的周期（TIM_mstr_ARR 寄存器）。
3. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=00010）。
4. 将 TIM_slv 配置为触发模式（TIM_slv_SMCR 寄存器中的 SMS=110）。
5. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“1”启动 TIM_mstr。

图 330. 通过 TIM_mstr 的更新事件触发 TIM_slv



如上述示例所示，用户可以在开始计数之前初始化两个计数器。*图 331* 显示了与*图 330* 具有相同配置，只不过处于触发模式（TIM_slv_SMCR 寄存器中的 SMS=110）而非门控模式的计数行为。

图 331. 通过 TIM_mstr 的使能信号触发 TIM_slv



使用一个外部触发同步启动 2 个定时器

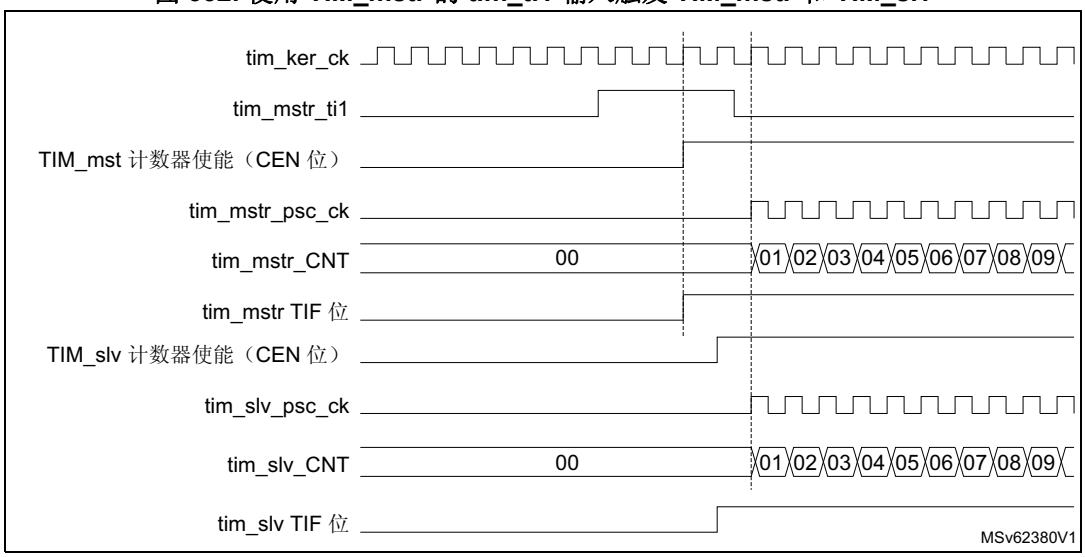
本例中，TIM_mstr 的 tim_ti1 输入出现上升沿时使能 TIM_mstr，使能 TIM_mstr 的同时使能 TIM_slv。相关连接图，请参见*图 326*。要确保计数器对齐，TIM_mstr 必须配置为主/从模式（对应的 tim_ti1 为从，对应 TIM_slv 为主）：

1. 将 TIM_mstr 配置为主模式，发送其使能信号作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=001）。
2. 将 TIM_mstr 配置为从模式以接收来自 tim_ti1 的输入触发（TIM_mstr_SMCR 寄存器中的 TS=00100）。
3. 将 TIM_mstr 配置为触发模式（TIM_mstr_SMCR 寄存器中的 SMS=110）。
4. 通过写入 MSM=1（TIM_mstr_SMCR 寄存器）将 TIM_mstr 配置为主/从模式。
5. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=00000）。
6. 将 TIM_slv 配置为触发模式（TIM_slv_SMCR 寄存器中的 SMS=110）。

当 tim_ti1 (TIM_mstr) 出现上升沿时，两个计数器开始根据内部时钟同步计数，并且两个 TIF 标志都置 1。

注意：本例中，两个定时器都在启动之前进行了初始化（通过将各自的 UG 位置 1）。两个计数器都从 0 开始计数，但可以通过对任意一个计数器寄存器 (TIMx_CNT) 进行写操作，在二者之间轻松插入一个偏移量。请注意，主/从模式在 TIM_mstr 的 CNT_EN 与 CK_PSC 之间产生了延迟。

图 332. 使用 TIM_mstr 的 tim_ti1 输入触发 TIM_mstr 和 TIM_slv



注意：必须先使能接收 tim_trgo 信号的从外设（定时器、ADC 等）的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改时钟频率（预分频器）。

27.4.24 ADC 触发

定时器可通过多种内部信号产生 ADC 触发事件，例如复位、使能或比较事件。

注意：必须先使能接收 tim_trgo 信号的从外设（例如定时器和 ADC）的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改时钟频率（预分频器）。

27.4.25 DMA 突发传输模式

TIMx 定时器能够根据一个事件生成多个 DMA 请求。主要目的是能够对定时器的一部分多次重新编程而无需软件开销，但也可以用于定期地一次性地批量读取定时器的多个寄存器的内容。

DMA 控制器访问目标唯一，必须指向虚拟寄存器 TIMx_DMAR。发生给定的定时器事件时，定时器会启动 DMA 请求序列（突发传输）。每次写入 TIMx_DMAR 寄存器都会重定向到定时器其中的一个寄存器。

TIMx_DCR 寄存器中的 DBL[4:0] 位设置 DMA 连续传送长度。当对 TIMx_DMAR 地址进行读或写访问时，定时器进行一次突发传输，即传输数量（按半字或字节）。

TIMx_DCR 寄存器中的 DBA[4:0] 位定义 DMA 传送的 DMA 基址（通过 TIMx_DMAR 地址执行读/写访问时）。DBA 定义为从 TIMx_CR1 寄存器地址开始计算的偏移量：

示例：

00000：TIMx_CR1

00001：TIMx_CR2

00010：TIMx_SMCR

TIMx_DCR 寄存器中的 DBSS[3:0] 位定义触发 DMA 突发传输的中断源（详细信息，请参见 *第 27.5.29 节：TIMx DMA 控制寄存器 (TIMx_DCR)（x = 2、3）*）。

例如，定时器 DMA 突发传送功能用于在发生更新事件后将 CCRx 寄存器（x = 2、3、4）的内容更新为通过 DMA 传输到 CCRx 寄存器中的多个半字。

具体操作步骤如下：

1. 将相应的 DMA 通道配置如下：
   - DMA 通道外设地址为 DMAR 寄存器地址。
   - DMA 通道存储器地址为包含要通过 DMA 传输到 CCRx 寄存器的数据的 RAM 缓冲区地址。
   - 要传输的数据量 = 3（参见下文注释）。
   - 禁止循环模式。
2. 通过将 DBA 和 DBL 位域配置如下来配置 DCR 寄存器：
   DBL = 3 次传输，DBA = 0xE 并且 DBSS = 1。
3. 使能 TIMx 更新 DMA 请求（DIER 寄存器中的 UDE 位置 1）。
4. 使能 TIMx
5. 使能 DMA 通道

本例适用于每个 CCRx 寄存器必须更新一次的情况。如果每个 CCRx 寄存器要更新两次，则要传输的数据量必须为 6。下面以包含 data1、data2、data3、data4、data5 和 data6 的 RAM 缓冲区为例。数据将按照如下方式传输到 CCRx 寄存器：在第一个更新 DMA 请求期间，data1 传输到 CCR2，data2 传输到 CCR3，data3 传输到 CCR4；在第二个更新 DMA 请求期间，data4 传输到 CCR2，data5 传输到 CCR3，data6 传输到 CCR4。

注意： *可以将空值写入保留的寄存器中。*

27.4.26 TIM2/TIM3DMA 请求

TIM2/TIM3 可生成 DMA 请求，如*表 197* 所示。

表 197. DMA 请求

DMA 请求信号	DMA 请求	使能 控制位
tim_upd_dma	更新	UDE
tim_cc1_dma	捕获/比较 1	CC1DE
tim_cc2_dma	捕获/比较 2	CC2DE
tim_cc3_dma	捕获/比较 3	CC3DE
tim_cc4_dma	捕获/比较 4	CC4DE
tim_trg_dma	触发	TDE

注意： *一些定时器的 DMA 请求可能未连接到 DMA 控制器。更多详细信息，请参见 DMA 控制器部分。*

27.4.27 调试模式

当微控制器进入调试模式（Cortex-M33 内核停止）时，TIMx 计数器可以继续正常工作，也可以停止工作。

调试模式下的行为可以使用调试支持 (DBG) 模块中每个定时器的专用配置位进行编程。

更多详细信息，请参见调试支持 (DBG) 部分。

27.4.28 TIM2/TIM3 低功耗模式

表 198. 低功耗模式对 TIM2/TIM3 的影响

模式	说明
睡眠	无影响，外设处于激活状态。中断可使器件退出睡眠模式。
停止	定时器停止工作，寄存器的内容保持不变。不产生任何中断。
待机	定时器掉电，退出待机模式后必须重新初始化。

27.4.29 TIM2/TIM3 中断

TIM2/TIM3 可产生多个中断，如*表 199* 所示。

表 199. 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止模式和待机模式
TIM_UP	更新	UIF	UIE	向 UIF 中写入 0	是	否
TIM_CC	捕获/比较 1	CC1IF	CC1IE	向 CC1IF 中写入 0	是	否
	捕获/比较 2	CC2IF	CC2IE	向 CC2IF 中写入 0	是	否
	捕获/比较 3	CC3IF	CC3IE	向 CC3IF 中写入 0	是	否
	捕获/比较 4	CC4IF	CC4IE	向 CC4IF 中写入 0	是	否
TIM_TRG	触发	TIF	TIE	向 TIF 中写入 0	是	否
TIM_DIR_IDX	索引	IDXF	IDXIE	向 IDXF 中写入 0	是	否
	方向	DIRF	DIRIE	向 DIRF 中写入 0	是	否
TIM_IERR	索引错误	IERRF	IERRIE	向 IERRF 中写入 0	是	否
TIM_TER	转换错误	TERRF	TERRIE	向 TERRF 中写入 0	是	否

27.5 TIM2/TIM3 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.2 节*。

外设寄存器可按半字（16 位）或字（32 位）访问。

27.5.1 TIMx 控制寄存器 1 (TIMx_CR1)（x = 2、3）

TIMx control register 1

偏移地址：0x000

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DITHEN	UIFREMAP	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
			rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15:13 保留，必须保持复位值。

位 12 **DITHEN**：抖动使能 (Dithering Enable)

0：禁止抖动

1：使能抖动

注意：只有在 CEN 位复位时才能修改 DITHEN 位。

位 11 **UIFREMAP**：UIF 状态位重映射 (UIF status bit remapping)

0：无重映射。UIF 状态位不复制到 TIMx_CNT 寄存器的位 31。

1：使能重映射。UIF 状态位复制到 TIMx_CNT 寄存器的位 31。

位 10 保留，必须保持复位值。

位 9:8 **CKD[1:0]**：时钟分频 (Clock division)

该位域指示定时器时钟 (tim_ker_ck) 频率与数字滤波器所使用的采样时钟（tim_etr_in、tim_tix）之间的分频比。

00：$t_{DTS} = t_{tim_ker_ck}$

01：$t_{DTS} = 2 \times t_{tim_ker_ck}$

10：$t_{DTS} = 4 \times t_{tim_ker_ck}$

11：保留

位 7 **ARPE**：自动重载预装载使能 (Auto-reload preload enable)

0：TIMx_ARR 寄存器不进行缓冲

1：TIMx_ARR 寄存器进行缓冲

位 6:5 **CMS[1:0]**：中心对齐模式选择 (Center-aligned mode selection)

00：边沿对齐模式。计数器根据方向位 (DIR) 递增计数或递减计数。

01：中心对齐模式 1。计数器交替进行递增计数和递减计数。仅当计数器递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志才置 1。

10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志才置 1。

11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志都会置 1。

注意：只要计数器处于使能状态 (CEN=1)，就不得从边沿对齐模式切换为中心对齐模式。

位 4 **DIR**：方向 (Direction)

0：计数器递增计数

1：计数器递减计数

注意：当定时器配置为中心对齐模式或编码器模式时，该位为只读状态。

位 3 **OPM**：单脉冲模式 (One-pulse mode)

0：计数器在发生更新事件时不会停止计数

1：计数器在发生下一更新事件时停止计数（将 CEN 位清零）

位 2 **URS**：更新请求源 (Update request source)

该位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能时，所有以下事件都会生成更新中断或 DMA 请求。此类事件包括：

– 计数器上溢/下溢

– 将 UG 位置 1

– 通过从模式控制器生成的更新事件

1：使能时，只有计数器上溢/下溢会生成更新中断或 DMA 请求。

位 1 **UDIS**：更新禁止 (Update disable)

该位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新 (UEV) 事件可通过以下事件之一生成：

– 计数器上溢/下溢

– 将 UG 位置 1

– 通过从模式控制器生成的更新事件

带缓冲的寄存器被加载为预加载数值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

位 0 **CEN**：计数器使能 (Counter enable)

0：禁止计数器

1：使能计数器

注意：只有事先通过软件将 CEN 位置 1，才可以使用外部时钟、门控模式和编码器模式。而触发模式可通过硬件自动将 CEN 位置 1。

在单脉冲模式下，当发生更新事件时会自动将 CEN 位清零。

27.5.2 TIMx 控制寄存器 2 (TIMx_CR2)（x = 2、3）

TIMx control register 2

偏移地址：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	MMS[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
						rw									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1S	MMS[2:0]			CCDS	Res.	Res.	Res.
								rw	rw	rw	rw	rw			

位 31:26 保留，必须保持复位值。

位 24:8 保留，必须保持复位值。

位 7 **TI1S**：tim_ti1 选择 (tim_ti1 selection)

0：tim_ti1_in[15:0] 复用器输出到 tim_ti1 输入

1：tim_ti1_in[15:0]、tim_ti2_in[15:0] 和 tim_ti3_in[15:0] 复用器输出进行异或运算，并连接到 tim_ti1 输入。另请参见*第 763 页的第 26.3.29 节：连接霍尔传感器*。

位 25、6、5、4 **MMS[3:0]**：主模式选择 (Master mode selection)

这些位用于选择主模式下将要发送到从定时器以实现同步的信息 (tim_trgo)。这些位的组合如下：

0000：**复位**——TIMx_EGR 寄存器中的 UG 位用作触发输出 (tim_trgo)。如果复位由触发输入生成（从模式控制器配置为复位模式），则 tim_trgo 上的信号相比实际复位会有延迟。

0001：**使能**——计数器使能信号 CNT_EN 用作触发输出 (tim_trgo)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号由 CEN 控制位与门控模式下的触发输入的逻辑与运算组合而成。
当计数器使能信号由触发输入控制时，tim_trgo 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中 MSM 位的说明）。

0010：**更新**——选择更新事件作为触发输出 (tim_trgo)。例如，主定时器可用作从定时器的预分频器。

0011：**比较脉冲**——CC1IF 标志置 1 时（即使已为高电平），只要发生捕获或比较匹配，触发输出都会发送一个正脉冲 (tim_trgo)。

0100：**比较**——tim_oc1refc 信号用作触发输出 (tim_trgo)

0101：**比较**——tim_oc2refc 信号用作触发输出 (tim_trgo)

0110：**比较**——tim_oc3refc 信号用作触发输出 (tim_trgo)

0111：**比较**——tim_oc4refc 信号用作触发输出 (tim_trgo)

1000：**编码器时钟输出**——编码器时钟信号用作触发输出 (tim_trgo)。该代码适用于以下 SMS[3:0] 值：0001、0010、0011、1010、1011、1100、1101、1110、1111。不允许使用任何其他 SMS[3:0] 代码，否则可能会导致意外行为。

其他值：保留

注意：必须先使能从定时器或 ADC 的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改从定时器或 ADC 的时钟。

位 3 **CCDS**：捕获/比较 DMA 选择 (Capture/compare DMA selection)

0：发生 CCx 事件时发送 CCx DMA 请求

1：发生更新事件时发送 CCx DMA 请求

位 2:0 保留，必须保持复位值。

27.5.3 TIMx 从模式控制寄存器 (TIMx_SMCR)（x = 2、3）

TIMx slave mode control register

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	SMSPS	SMSPE	Res.	Res.	TS[4:3]		Res.	Res.	Res.	SMS[3]
						rw	rw			rw	rw				rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:26 保留，必须保持复位值。

位 25 **SMSPS**：SMS 预装载源 (SMS preload source)

该位用于选择将 SMS[3:0] 位域预装载值传输到有效值的触发事件

0：传输由定时器的更新事件触发

1：传输由索引事件触发

位 24 **SMSPE**：SMS 预装载使能 (SMS preload enable)

该位用于选择是否预装载 SMS[3:0] 位域

0：禁止 SMS[3:0] 位域预装载

1：使能 SMS[3:0] 位域预装载

位 23:22 保留，必须保持复位值。

位 19:17 保留，必须保持复位值。

位 15 **ETP**：外部触发极性 (External trigger polarity)

该位可选择将 tim_etr_in 还是 $\overline{\text{tim_etr_in}}$ 用于触发操作

0：tim_etr_in 未反相，高电平或上升沿有效

1：tim_etr_in 反相，低电平或下降沿有效

位 14 **ECE**：外部时钟使能 (External clock enable)

该位可使能外部时钟模式 2。

0：禁止外部时钟模式 2

1：使能外部时钟模式 2。计数器时钟由 tim_etrf 信号的任意有效边沿提供。

注意：将 ECE 位置 1 与选择外部时钟模式 1 并将 tim_trgi 连接到 tim_etrf（SMS=111 且 TS=00111）具有相同效果。

外部时钟模式 2 可以和以下从模式同时使用：复位模式、门控模式和触发模式。不过此类情况下 tim_trgi 不得连接 tim_etrf（TS 位不得为 00111）。

如果同时使能外部时钟模式 1 和外部时钟模式 2，则外部时钟输入为 tim_etrf。

位 13:12 **ETPS[1:0]**：外部触发预分频器 (External trigger prescaler)

外部触发信号 tim_etrp 频率不得超过 tim_ker_ck 频率的 1/4。可通过使能预分频器来降低 tim_etrp 频率。这种方法在通过 tim_etr_in 输入快速外部时钟时非常有用。

00：预分频器关闭

01：2 分频 tim_etrp 频率

10：4 分频 tim_etrp 频率

11：8 分频 tim_etrp 频率

位 11:8 **ETF[3:0]**：外部触发滤波器 (External trigger filter)

该位域可定义 tim_etrp 信号的采样频率和适用于 tim_etrp 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿：

0000：无滤波器，按 f_{DTS} 频率进行采样
0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2
0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4
0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8
0100：$f_{SAMPLING}=f_{DTS}/2$，N=6
0101：$f_{SAMPLING}=f_{DTS}/2$，N=8
0110：$f_{SAMPLING}=f_{DTS}/4$，N=6
0111：$f_{SAMPLING}=f_{DTS}/4$，N=8
1000：$f_{SAMPLING}=f_{DTS}/8$，N=6
1001：$f_{SAMPLING}=f_{DTS}/8$，N=8
1010：$f_{SAMPLING}=f_{DTS}/16$，N=5
1011：$f_{SAMPLING}=f_{DTS}/16$，N=6
1100：$f_{SAMPLING}=f_{DTS}/16$，N=8
1101：$f_{SAMPLING}=f_{DTS}/32$，N=5
1110：$f_{SAMPLING}=f_{DTS}/32$，N=6
1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 7 **MSM**：主/从模式 (Master/Slave mode)

0：无操作
1：当前定时器的触发输入事件 (tim_trgi) 的动作被推迟，以使当前定时器与其从定时器实现完美同步（通过 tim_trgo）。此设置适用于由单个外部事件对多个定时器进行同步的情况。

位 21、20、6、5、4 **TS[4:0]**：触发选择 (Trigger selection)

该位域可选择将要用于同步计数器的触发输入。

00000：内部触发 0 (tim_itr0)
00001：内部触发 1 (tim_itr1)
00010：内部触发 2 (tim_itr2)
00011：内部触发 3 (tim_itr3)
00100：tim_ti1 边沿检测器 (tim_ti1f_ed)
00101：滤波后的定时器输入 1 (tim_ti1fp1)
00110：滤波后的定时器输入 2 (tim_ti2fp2)
00111：外部触发输入 (tim_etrf)
01000：内部触发 4 (tim_itr4)
01001：内部触发 5 (tim_itr5)
01010：内部触发 6 (tim_itr6)
01011：内部触发 7 (tim_itr7)
01100：内部触发 8 (tim_itr8)
01101：内部触发 9 (tim_itr9)
01110：内部触发 10 (tim_itr10)
01111：内部触发 11 (tim_itr11)
10000：内部触发 12 (tim_itr12)
10001：内部触发 13 (tim_itr13)
10010：内部触发 14 (tim_itr14)
10011：内部触发 15 (tim_itr15)
其他值：保留

有关产品特定实现的详细信息，请参见*第 27.4.2 节：TIM2/TIM3 引脚和内部信号*。

注意：这些位只能在未使用的情况下（例如，SMS = 000 时）进行更改，以避免转换时出现错误的边沿检测。

位 3 **OCCS**：OCREF 清零选择 (OCREF clear selection)

该位用于选择 OCREF 清零源

0：tim_ocref_clr_int 连接到 tim_ocref_clr 输入

1：tim_ocref_clr_int 连接到 tim_etrf

注意：如果不支持 OCREF 清零选择功能，该位保留并由硬件强制清零。第 27.3 节：TIM2/TIM3 实现。

位 16、2、1、0 **SMS[3:0]**：从模式选择 (Slave mode selection)

选择外部信号时，触发信号 (tim_trgi) 的有效边沿与外部输入上所选的极性相关（请参见输入控制寄存器和控制寄存器说明）。

0000：禁止从模式——如果 CEN = “1”，预分频器时钟直接由内部时钟提供。

0001：编码器模式 1——计数器根据 tim_ti2fp2 电平在 tim_ti1fp1 边沿递增/递减计数。

0010：编码器模式 2——计数器根据 tim_ti1fp1 电平在 tim_ti2fp2 边沿递增/递减计数。

0011：编码器模式 3——计数器在 tim_ti1fp1 和 tim_ti2fp2 的边沿计数，计数的方向取决于另外一个信号的电平。

0100：复位模式——在出现所选触发输入 (tim_trgi) 上升沿时，重新初始化计数器并生成一个寄存器更新事件。

0101：门控模式——触发输入 (tim_trgi) 为高电平时使能计数器时钟。只要触发输入变为低电平，计数器立即停止计数（但不复位）。计数器的启动和停止都被控制。

0110：触发模式——触发信号 tim_trgi 出现上升沿时启动计数器（但不复位）。只控制计数器的启动。

0111：外部时钟模式 1——由所选触发信号 (tim_trgi) 的上升沿提供计数器时钟。

1000：复位 + 触发组合模式——在出现所选触发输入 (tim_trgi) 上升沿时，重新初始化计数器，生成一个寄存器更新事件并启动计数器。

1001：门控 + 复位组合模式——触发输入 (tim_trgi) 为高电平时使能计数器时钟。只要触发输入变为低电平，计数器立即停止计数并复位。计数器的启动和停止都被控制。

1010：编码器模式：时钟 + 方向，x2 模式。

1011：编码器模式：时钟 + 方向，x1 模式，tim_ti2fp2 边沿有效性由 CC2P 设置。

1100：编码器模式：定向时钟，x2 模式。

1101：编码器模式：定向时钟，x1 模式，tim_ti1fp1 和 tim_ti2fp2 的边沿有效性由 CC1P 和 CC2P 设置。

1110：正交编码器模式：x1 模式，仅在 tim_ti1fp1 边沿处计数，边沿有效性由 CC1P 设置。

1111：正交编码器模式：x1 模式，仅在 tim_ti2fp2 边沿处计数，边沿有效性由 CC2P 设置。

注意：如果将 tim_ti1f_ed 选作触发输入 (TS=00100)，则不得使用门控模式。实际上，tim_ti1f 每次转换时，tim_ti1f_ed 都输出 1 个脉冲，而门控模式检查的则是触发信号的电平。

注意：必须先使能接收 tim_trgo 信号的从外设（定时器、ADC 等）的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改时钟频率（预分频器）。

27.5.4 TIMx DMA/中断使能寄存器 (TIMx_DIER)（x = 2、3）

TIMx DMA/Interrupt enable register

偏移地址：0x00C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TERRIE	IERRIE	DIRIE	IDXIE	Res.	Res.	Res.	Res.
								rw	rw	rw	rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw		rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23 **TERRIE**：转换错误中断使能 (Transition error interrupt enable)
0：禁止转换错误中断
1：使能转换错误中断

位 22 **IERRIE**：索引错误中断使能 (Index error interrupt enable)
0：禁止索引错误中断
1：使能索引错误中断

位 21 **DIRIE**：方向更改中断使能 (Direction change interrupt enable)
0：禁止方向更改中断
1：使能方向更改中断

位 20 **IDXIE**：索引中断使能 (Index interrupt enable)
0：禁止索引中断
1：使能索引中断

位 19:15 保留，必须保持复位值。

位 14 **TDE**：触发信号 DMA 请求使能 (Trigger DMA request enable)
0：禁止触发信号 DMA 请求。
1：使能触发信号 DMA 请求。

位 13 保留，必须保持复位值。

位 12 **CC4DE**：捕获/比较 4 DMA 请求使能 (Capture/Compare 4 DMA request enable)
0：禁止 CC4 DMA 请求。
1：使能 CC4 DMA 请求。

位 11 **CC3DE**：捕获/比较 3 DMA 请求使能 (Capture/Compare 3 DMA request enable)
0：禁止 CC3 DMA 请求。
1：使能 CC3 DMA 请求。

位 10 **CC2DE**：捕获/比较 2 DMA 请求使能 (Capture/Compare 2 DMA request enable)
0：禁止 CC2 DMA 请求。
1：使能 CC2 DMA 请求。

位 9 **CC1DE**：捕获/比较 1 DMA 请求使能 (Capture/Compare 1 DMA request enable)
0：禁止 CC1 DMA 请求。
1：使能 CC1 DMA 请求。

位 8 **UDE**：更新 DMA 请求使能 (Update DMA request enable)
0：禁止更新 DMA 请求。
1：使能更新 DMA 请求。

位 7 保留，必须保持复位值。

位 6 **TIE**：触发信号中断使能 (Trigger interrupt enable)
0：禁止触发信号中断。
1：使能触发信号中断。

位 5 保留，必须保持复位值。

位 4 **CC4IE**：捕获/比较 4 中断使能 (Capture/Compare 4 interrupt enable)
0：禁止 CC4 中断。
1：使能 CC4 中断。

位 3 **CC3IE**：捕获/比较 3 中断使能 (Capture/Compare 3 interrupt enable)
0：禁止 CC3 中断。
1：使能 CC3 中断。

位 2 **CC2IE**：捕获/比较 2 中断使能 (Capture/Compare 2 interrupt enable)
0：禁止 CC2 中断。
1：使能 CC2 中断。

位 1 **CC1IE**：捕获/比较 1 中断使能 (Capture/Compare 1 interrupt enable)
0：禁止 CC1 中断。
1：使能 CC1 中断。

位 0 **UIE**：更新中断使能 (Update interrupt enable)
0：禁止更新中断。
1：使能更新中断。

27.5.5 TIMx 状态寄存器 (TIMx_SR)（x = 2、3）

TIMx status register

偏移地址：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TERRF	IERRF	DIRF	IDXF	Res.	Res.	Res.	Res.
								rc_w0	rc_w0	rc_w0	rc_w0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	Res.	TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0			rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

位 31:24 保留，必须保持复位值。

位 23 **TERRF**：转换错误中断标志 (Transition error interrupt flag)
在编码器模式下检测到转换错误时，该标志由硬件置 1。通过软件写入“0”可将该位清零。
0：未检测到编码器转换错误
1：检测到编码器转换错误

位 22 **IERRF**：索引错误中断标志 (Index error interrupt flag)
检测到索引错误时，该标志由硬件置 1。通过软件写入“0”可将该位清零。
0：未检测到索引错误
1：检测到索引错误

位 21 **DIRF**：方向更改中断标志 (Direction change interrupt flag)
在编码器模式下发生方向更改时（TIMx_CR 中 DIR 位的值发生变化），该标志由硬件置 1。通过软件写入“0”可将该位清零。
0：方向未更改
1：方向更改

位 20 **IDXF**：索引中断标志 (Index interrupt flag)
检测到索引事件时，该标志由硬件置 1。通过软件
写入“0”可将该位清零。
0：未发生索引事件
1：发生了索引事件

位 19:13 保留，必须保持复位值。

位 12 **CC4OF**：捕获/比较 4 过捕获标志 (Capture/Compare 4 overcapture flag)
请参见 CC1OF 说明

位 11 **CC3OF**：捕获/比较 3 过捕获标志 (Capture/Compare 3 overcapture flag)
请参见 CC1OF 说明

位 10 **CC2OF**：捕获/比较 2 过捕获标志 (Capture/compare 2 overcapture flag)
请参见 CC1OF 说明

位 9 **CC1OF**：捕获/比较 1 过捕获标志 (Capture/Compare 1 overcapture flag)
仅当将相应通道配置为输入捕获模式时，此标志位才会由硬件置 1。通过软件写入“0”可将该位清零。
0：未检测到过捕获。
1：TIMx_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。

位 8:7 保留，必须保持复位值。

位 6 **TIF**：触发中断标志 (Trigger interrupt flag)
在除门控模式以外的所有模式下，当使能从模式控制器后在 tim_trgi 输入上检测到 TRG 触发事件有效边沿时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动或停止时置 1。但需要通过软件清零。
0：未发生触发事件。
1：触发中断挂起。

位 5 保留，必须保持复位值。

位 4 **CC4IF**：捕获/比较 4 中断标志 (Capture/Compare 4 interrupt flag)
请参见 CC1IF 说明

位 3 **CC3IF**：捕获/比较 3 中断标志 (Capture/Compare 3 interrupt flag)
请参见 CC1IF 说明

位 2 **CC2IF**：捕获/比较 2 中断标志 (Capture/Compare 2 interrupt flag)
请参见 CC1IF 说明

位 1 **CC1IF**：捕获/比较 1 中断标志 (Capture/compare 1 interrupt flag)
该标志由硬件置 1，但需要通过软件清零（输入捕获或输出比较模式）或通过读取 TIMx_CCR1 寄存器清零（仅限输入捕获模式）。
0：未发生比较匹配/输入捕获
1：发生了比较匹配/输入捕获
如果通道 CC1 配置为输出：当计数器 TIMx_CNT 的值与 TIMx_CCR1 的值匹配时，此标志置 1。当 TIMx_CCR1 的值大于 TIMx_ARR 的值时，CC1IF 位将在计数器发生上溢（递增计数模式和增减计数模式下）或下溢（递减计数模式下）时变为高电平。在中心对齐模式下，可通过 3 种方式将此标志置 1，完整说明请参见 TIMx_CR1 寄存器中的 CMS 位。
如果通道 CC1 配置为输入：当 TIMx_CCR1 寄存器中捕获到计数器值时（如果通过设置 TIMx_CCER 中的 CC1P 和 CC1NP 位定义为边沿有效，则改为在 IC1 上检测到边沿时），该位置 1。

位 0 **UIF**：更新中断标志 (Update interrupt flag)
该位在发生更新事件时通过硬件置 1。但需要通过软件清零。
0：未发生更新
1：更新中断挂起。该位在以下情况下更新寄存器时由硬件置 1：
上溢或下溢并且当 TIMx_CR1 寄存器中 UDIS = 0 时。
TIMx_CR1 寄存器中的 URS = 0 且 UDIS = 0，并且由软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。
TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且 CNT 由触发事件重新初始化时（参见同步控制寄存器说明）。

27.5.6 TIMx 事件生成寄存器 (TIMx_EGR)（x = 2、3）

TIMx event generation register

偏移地址：0x014

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
									w		w	w	w	w	w

位 15:7 保留，必须保持复位值。

位 6 **TG**：触发生成 (Trigger generation)
该位由软件置 1 以生成事件，并由硬件自动清零。
0：无操作
1：TIMx_SR 寄存器中的 TIF 标志置 1。使能后可发生相关中断或 DMA 传输事件。

位 5 保留，必须保持复位值。

位 4 **CC4G**：捕获/比较 4 生成 (Capture/compare 4 generation)
请参见 CC1G 说明

位 3 **CC3G**：捕获/比较 3 生成 (Capture/compare 3 generation)
请参见 CC1G 说明

位 2 **CC2G**：捕获/比较 2 生成 (Capture/compare 2 generation)
请参见 CC1G 说明

位 1 **CC1G**：捕获/比较 1 生成 (Capture/compare 1 generation)
该位由软件置 1 以生成事件，并由硬件自动清零。
0：无操作
1：通道 1 上生成捕获/比较事件：
如果通道 CC1 配置为输出：
使能时，CC1IF 标志置 1 并发送相应的中断或 DMA 请求。
如果通道 CC1 配置为输入：
TIMx_CCR1 寄存器中将捕获到计数器当前值。使能时，CC1IF 标志置 1 并发送相应的中断或 DMA 请求。如果 CC1IF 标志已为高电平，CC1OF 标志将置 1。

位 0 **UG**：更新生成 (Update generation)
该位可通过软件置 1，并由硬件自动清零。
0：无操作
1：重新初始化计数器并生成寄存器更新事件。请注意，预分频器计数器也将清零（但预分频比不受影响）。如果选择中心对齐模式或 DIR=0（递增计数），计数器将清零；如果 DIR=1（递减计数），计数器将使用自动重载值 (TIMx_ARR)。

27.5.7 TIMx 捕获/比较模式寄存器 1 (TIMx_CCMR1)（x = 2、3）

TIMx capture/compare mode register 1

偏移地址：0x018

复位值：0x0000 0000

同一寄存器可用于输入捕获模式（本节）或输出比较模式（下一节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。可独立组合两种模式（例如，输入捕获模式下的通道 1 和输出比较模式下的通道 2）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输入捕获模式

位 31:16 保留，必须保持复位值。

位 15:12 **IC2F[3:0]**：输入捕获 2 滤波器 (Input capture 2 filter)

位 11:10 **IC2PSC[1:0]**：输入捕获 2 预分频器 (Input capture 2 prescaler)

位 9:8 **CC2S[1:0]**：捕获/比较 2 选择 (Capture/compare 2 selection)

该位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道配置为输出。

01：CC2 通道配置为输入，tim_ic2 映射到 tim_ti2 上。

10：CC2 通道配置为输入，tim_ic2 映射到 tim_ti1 上。

11：CC2 通道配置为输入，tim_ic2 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注意：仅当通道关闭时（TIMx_CCER 中的 CC2E = 0），才可向 CC2S 位写入数据。

位 7:4 **IC1F[3:0]**：输入捕获 1 滤波器 (Input capture 1 filter)

该位域可定义 tim_ti1 输入的采样频率和适用于 tim_ti1 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿：

0000：无滤波器，按 f_{DTS} 频率进行采样

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8

1000：$f_{SAMPLING}=f_{DTS}/8$，N=6

1001：$f_{SAMPLING}=f_{DTS}/8$，N=8

1010：$f_{SAMPLING}=f_{DTS}/16$，N=5

1011：$f_{SAMPLING}=f_{DTS}/16$，N=6

1100：$f_{SAMPLING}=f_{DTS}/16$，N=8

1101：$f_{SAMPLING}=f_{DTS}/32$，N=5

1110：$f_{SAMPLING}=f_{DTS}/32$，N=6

1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 3:2 **IC1PSC[1:0]**：输入捕获 1 预分频器 (Input capture 1 prescaler)

该位域定义 CC1 输入 (tim_ic1) 的预分频比。只要 CC1E=0（TIMx_CCER 寄存器），预分频器便立即复位。

00：无预分频器，捕获输入上每检测到一个边沿便执行捕获

01：每发生 2 个事件便执行一次捕获

10：每发生 4 个事件便执行一次捕获

11：每发生 8 个事件便执行一次捕获

位 1:0 **CC1S[1:0]**：捕获/比较 1 选择 (Capture/Compare 1 selection)
该位域定义通道方向（输入/输出）以及所使用的输入。
00：CC1 通道配置为输出
01：CC1 通道配置为输入，tim_ic1 映射到 tim_ti1 上
10：CC1 通道配置为输入，tim_ic1 映射到 tim_ti2 上
11：CC1 通道配置为输入，tim_ic1 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注意：仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

27.5.8 TIMx 捕获/比较模式寄存器 1 [复用] (TIMx_CCMR1)（x = 2、3）

TIMx capture/compare mode register 1

偏移地址：0x018

复位值：0x0000 0000

同一寄存器可用于输出比较模式（本节）或输入捕获模式（上一节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。可独立组合两种模式（例如，输入捕获模式下的通道 1 和输出比较模式下的通道 2）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M [3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

位 31:25 保留，必须保持复位值。

位 23:17 保留，必须保持复位值。

位 15 **OC2CE**：输出比较 2 清零使能 (Output compare 2 clear enable)

位 24、14:12 **OC2M[3:0]**：输出比较 2 模式 (Output compare 2 mode)
请参见 OC1M 说明——位 6:4

位 11 **OC2PE**：输出比较 2 预装载使能 (Output compare 2 preload enable)

位 10 **OC2FE**：输出比较 2 快速使能 (Output compare 2 fast enable)

位 9:8 **CC2S[1:0]**：捕获/比较 2 选择 (Capture/Compare 2 selection)
该位域定义通道方向（输入/输出）以及所使用的输入。
00：CC2 通道配置为输出
01：CC2 通道配置为输入，tim_ic2 映射到 tim_ti2 上
10：CC2 通道配置为输入，tim_ic2 映射到 tim_ti1 上
11：CC2 通道配置为输入，tim_ic2 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注意：仅当通道关闭时（TIMx_CCER 中的 CC2E = 0），才可向 CC2S 位写入数据。

位 7 **OC1CE**：输出比较 1 清零使能 (Output compare 1 clear enable)
0：tim_oc1ref 不受 tim_ocref_clr_int 输入影响
1：tim_ocref_clr_int 输入上检测到高电平时，tim_oc1ref 立即清零

位 16，6:4 **OC1M[3:0]**：输出比较 1 模式 (Output compare 1 mode)

这些位定义了输出参考信号 tim_oc1ref 的行为，tim_oc1 就是从这个信号派生出来的。tim_oc1ref 为高电平有效，而 tim_oc1 的有效电平则取决于 CC1P 位。

0000：冻结——输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 进行比较不会对输出造成任何影响。当定时器用作软件时基时，可以使用该模式。在定时器操作期间使能冻结模式时，输出保持进入冻结状态前的状态（有效或无效）。

0001：将通道 1 设置为匹配时输出有效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1 (TIMx_CCR1) 匹配时，tim_oc1ref 信号强制变为高电平。

0010：将通道 1 设置为匹配时输出无效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1 (TIMx_CCR1) 匹配时，tim_oc1ref 信号强制变为低电平。

0011：翻转——TIMx_CNT=TIMx_CCR1 时，tim_oc1ref 发生翻转。

0100：强制变为无效电平——tim_oc1ref 强制变为低电平。

0101：强制变为有效电平——tim_oc1ref 强制变为高电平。

0110：PWM 模式 1——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便输出有效电平，否则输出无效电平。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便输出无效电平（tim_oc1ref=0），否则输出有效电平（tim_oc1ref=1）。

0111：PWM 模式 2——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便输出无效电平，否则输出有效电平。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便输出有效电平，否则输出无效电平。

1000：可再触发 OPM 模式 1——在递增计数模式下，通道为有效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为无效状态。在递减计数模式下，通道为无效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为无效状态。

1001：可再触发 OPM 模式 2——在递增计数模式下，通道为无效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 2 下进行比较，通道会在下一次更新时再次变为无效状态。在递减计数模式下，通道为有效状态，直至（在 tim_trgi 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为有效状态。

1010：脉冲比较。

1011：保留。

1100：组合 PWM 模式 1——tim_oc1ref 与在 PWM 模式 1 下的行为相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 的逻辑或运算结果。

1101：组合 PWM 模式 2——tim_oc1ref 与在 PWM 模式 2 下的行为相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 的逻辑与运算结果。

1110：非对称 PWM 模式 1——tim_oc1ref 与在 PWM 模式 1 下的行为相同。计数器递增计数时，tim_oc1refc 输出 tim_oc1ref；计数器递减计数时，tim_oc1refc 输出 tim_oc2ref。

1111：非对称 PWM 模式 2——tim_oc1ref 与在 PWM 模式 2 下的行为相同。计数器递增计数时，tim_oc1refc 输出 tim_oc1ref；计数器递减计数时，tim_oc1refc 输出 tim_oc2ref。

注意：在 PWM 模式下，当比较结果发生改变，输出比较模式从“冻结”模式切换到“PWM”模式以及输出比较模式从“强制有效/无效”模式切换到“PWM”模式时，OCREF 电平会发生更改。

位 3 **OC1PE**：输出比较 1 预装载使能 (Output compare 1 preload enable)

0：禁止与 TIMx_CCR1 相关的预装载寄存器。可随时向 TIMx_CCR1 写入数据，写入后将立即使用新值。

1：使能与 TIMx_CCR1 相关的预装载寄存器。可读/写访问预装载寄存器。TIMx_CCR1 预装载值在每次生成更新事件时都会装载到影子寄存器中。

位 2 **OC1FE**：输出比较 1 快速使能 (Output compare 1 fast enable)

该位可降低触发事件与定时器输出跳变之间的延迟。单脉冲模式下必须使用（TIMx_CR1 寄存器中的 OPM 位置 1），以便在启动触发后快速发送输出脉冲。

0：即使触发开启，CC1 也将根据计数器和 CCR1 值正常工作。触发输入出现边沿时，激活 CC1 输出的最短延迟时间为 5 个时钟周期。

1：触发输入上出现有效边沿相当于 CC1 输出上的比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和激活 CC1 输出的延迟时间缩短为 3 个时钟周期。仅当通道配置为 PWM1 或 PWM2 模式时，OCFE 才会起作用。

位 1:0 **CC1S[1:0]**：捕获/比较 1 选择 (Capture/Compare 1 selection)

该位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出。

01：CC1 通道配置为输入，tim_ic1 映射到 tim_ti1 上。

10：CC1 通道配置为输入，tim_ic1 映射到 tim_ti2 上。

11：CC1 通道配置为输入，tim_ic1 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注意：仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

27.5.9 TIMx 捕获/比较模式寄存器 2 (TIMx_CCMR2)（x = 2、3）

TIMx capture/compare mode register 2

偏移地址：0x01C

复位值：0x0000 0000

同一寄存器可用于输入捕获模式（本节）或输出比较模式（下一节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。可独立组合两种模式（例如，输入捕获模式下的通道 1 和输出比较模式下的通道 2）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC4F[3:0]				IC4PSC[1:0]		CC4S[1:0]		IC3F[3:0]				IC3PSC[1:0]		CC3S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输入捕获模式

位 31:16 保留，必须保持复位值。

位 15:12 **IC4F[3:0]**：输入捕获 4 滤波器 (Input capture 4 filter)

位 11:10 **IC4PSC[1:0]**：输入捕获 4 预分频器 (Input capture 4 prescaler)

位 9:8 **CC4S[1:0]**：捕获/比较 4 选择 (Capture/Compare 4 selection)
该位域定义通道方向（输入/输出）以及所使用的输入。
00：CC4 通道配置为输出
01：CC4 通道配置为输入，tim_ic4 映射到 tim_ti4 上
10：CC4 通道配置为输入，tim_ic4 映射到 tim_ti3 上
11：CC4 通道配置为输入，tim_ic4 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注意：仅当通道关闭时（TIMx_CCER 中的 CC4E = 0），才可向 CC4S 位写入数据。

位 7:4 **IC3F[3:0]**：输入捕获 3 滤波器 (Input capture 3 filter)

位 3:2 **IC3PSC[1:0]**：输入捕获 3 预分频器 (Input capture 3 prescaler)

位 1:0 **CC3S[1:0]**：捕获/比较 3 选择 (Capture/Compare 3 selection)
该位域定义通道方向（输入/输出）以及所使用的输入。
00：CC3 通道配置为输出
01：CC3 通道配置为输入，tim_ic3 映射到 tim_ti3 上
10：CC3 通道配置为输入，tim_ic3 映射到 tim_ti4 上
11：CC3 通道配置为输入，tim_ic3 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注意：仅当通道关闭时（TIMx_CCER 中的 CC3E = 0），才可向 CC3S 位写入数据。

27.5.10 TIMx 捕获/比较模式寄存器 2 [复用] (TIMx_CCMR2)（x = 2、3）

TIMx capture/compare mode register 2

偏移地址：0x01C

复位值：0x0000 0000

同一寄存器可用于输出比较模式（本节）或输入捕获模式（上一节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。可独立组合两种模式（例如，输入捕获模式下的通道 1 和输出比较模式下的通道 2）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4M [3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC3M [3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

位 31:25 保留，必须保持复位值。

位 23:17 保留，必须保持复位值。

位 15 **OC4CE**：输出比较 4 清零使能 (Output compare 4 clear enable)

位 24、14:12 **OC4M[3:0]**：输出比较 4 模式 (Output compare 4 mode)
请参见 OC1M 说明（TIMx_CCMR1 寄存器中的位 6:4）

位 11 **OC4PE**：输出比较 4 预装载使能 (Output compare 4 preload enable)

位 10 **OC4FE**：输出比较 4 快速使能 (Output compare 4 fast enable)

位 9:8 **CC4S[1:0]**：捕获/比较 4 选择 (Capture/Compare 4 selection)
该位域定义通道方向（输入/输出）以及所使用的输入。
00：CC4 通道配置为输出
01：CC4 通道配置为输入，tim_ic4 映射到 tim_ti4 上
10：CC4 通道配置为输入，tim_ic4 映射到 tim_ti3 上
11：CC4 通道配置为输入，tim_ic4 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注意：仅当通道关闭时（TIMx_CCER 中的 CC4E = 0），才可向 CC4S 位写入数据。

位 7 **OC3CE**：输出比较 3 清零使能 (Output compare 3 clear enable)

位 16，6:4 **OC3M[3:0]**：输出比较 3 模式 (Output compare 3 mode)
请参见 OC1M 说明（TIMx_CCMR1 寄存器中的位 6:4）

位 3 **OC3PE**：输出比较 3 预装载使能 (Output compare 3 preload enable)

位 2 **OC3FE**：输出比较 3 快速使能 (Output compare 3 fast enable)

位 1:0 **CC3S[1:0]**：捕获/比较 3 选择 (Capture/Compare 3 selection)
该位域定义通道方向（输入/输出）以及所使用的输入。
00：CC3 通道配置为输出
01：CC3 通道配置为输入，tim_ic3 映射到 tim_ti3 上
10：CC3 通道配置为输入，tim_ic3 映射到 tim_ti4 上
11：CC3 通道配置为输入，tim_ic3 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注意：仅当通道关闭时（TIMx_CCER 中的 CC3E = 0），才可向 CC3S 位写入数据。

27.5.11 TIMx 捕获/比较使能寄存器 (TIMx_CCER)（x = 2、3）

TIMx capture/compare enable register

偏移地址：0x020

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Res.	CC4P	CC4E	CC3NP	Res.	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
rw		rw	rw	rw		rw	rw	rw		rw	rw	rw		rw	rw

位 15 **CC4NP**：*捕获/比较 4 输出极性 (Capture/Compare 4 output Polarity)*。
请参见 CC1NP 说明

位 14 保留，必须保持复位值。

位 13 **CC4P**：*捕获/比较 4 输出极性 (Capture/Compare 4 output Polarity)*。
请参见 CC1P 说明

位 12 **CC4E**：*捕捉/比较 4 输出使能 (Capture/Compare 4 output enable)*。
请参见 CC1E 说明

位 11 **CC3NP**：*捕获/比较 3 输出极性 (Capture/Compare 3 output polarity)*。
请参见 CC1NP 说明

位 10 保留，必须保持复位值。

位 9 **CC3P**：*捕获/比较 3 输出极性 (Capture/Compare 3 output polarity)*。
请参见 CC1P 说明

位 8 **CC3E**：*捕捉/比较 3 输出使能 (Capture/Compare 3 output enable)*。
请参见 CC1E 说明

位 7 **CC2NP**：*捕获/比较 2 输出极性 (Capture/Compare 2 output polarity)*。
请参见 CC1NP 说明

位 6 保留，必须保持复位值。

位 5 **CC2P**：*捕获/比较 2 输出极性 (Capture/Compare 2 output polarity)*。
请参见 CC1P 说明

位 4 **CC2E**：*捕捉/比较 2 输出使能 (Capture/Compare 2 output enable)*。
请参见 CC1E 说明

位 3 **CC1NP**：*捕获/比较 1 输出极性 (Capture/Compare 1 output polarity)*。
CC1 通道配置为输出：在这种情况下，CC1NP 必须保持清零。
CC1 通道配置为输入：该位与 CC1P 配合使用，用以定义 tim_ti1fp1/tim_ti2fp1 的极性。请参见 CC1P 说明。

位 2 保留，必须保持复位值。

位 1 **CC1P**：*捕获/比较 1 输出极性 (Capture/Compare 1 output polarity)*。
0：OC1 高电平有效（输出模式）/边沿有效性选择（输入模式，见下文）
1：OC1 低电平有效（输出模式）/边沿有效性选择（输入模式，见下文）
CC1 通道配置为输入时，CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的有效极性。
CC1NP=0，CC1P=0：非反相/上升沿触发。电路对 TIxFP1 上升沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式或编码器模式下执行触发操作）。
CC1NP=0，CC1P=1：反相/下降沿触发。电路对 TIxFP1 下降沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 反相（在门控模式或编码器模式下执行触发操作）。
CC1NP=1，CC1P=1：非反相/上升沿和下降沿均触发。电路对 TIxFP1 上升沿和下降沿都敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式下执行触发操作）。编码器模式下不得使用此配置。
CC1NP=1，CC1P=0：该配置保留，不得使用。

位 0 **CC1E**：*捕获/比较 1 输出使能 (Capture/Compare 1 output enable)*。
0：禁止捕获模式/OC1 无效
1：使能捕获模式/在相应输出引脚上输出 OC1 信号

表 200. 标准 tim_ocx 通道的输出控制位

CCxE 位	tim_ocx 输出状态
0	禁止输出（不由定时器驱动：高阻态）
1	使能输出（tim_ocx = tim_ocxref + 极性）

注意：　当 CCxE=0 时，与标准 tim_ocx 通道相连的外部 IO 引脚的状态仅取决于 GPIO 寄存器。

27.5.12 TIM3 计数器 (TIM3_CNT)

TIM3 counter

偏移地址：0x024

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UIF CPY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
rw															

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

位 31 **UIFCPY**：值取决于 TIMx_CR1 中的 IUFREMAP。

如果 UIFREMAP = 0

保留

如果 UIFREMAP = 1

UIFCPY：UIF 副本 (UIF Copy)

该位是 TIMx_ISR 寄存器中 UIF 位的只读副本

位 30:16 保留，必须保持复位值。

位 15:0 **CNT[15:0]**：计数器值 (Counter value)

不抖动模式 (DITHEN = 0)

该寄存器保存计数器值。

抖动模式 (DITHEN = 1)

该寄存器在 CNT[15:0] 中保存不抖动部分。小数部分不保存。

27.5.13 TIM2 计数器 (TIM2_CNT)

TIM2 counter

偏移地址：0x024

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UIF CPY_ CNT [31]	CNT[30:16]														
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

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

位 31 **UIFCPY_CNT[31]**：值取决于 TIMx_CR1 中的 UIFREMAP。

如果 UIFREMAP = 0

CNT[31]：计数器值的最高有效位 (Most significant bit of counter value)

如果 UIFREMAP = 1

UIFCPY：UIF 副本 (UIF Copy)

该位是 TIMx_ISR 寄存器中 UIF 位的只读副本

位 30:0 **CNT[30:0]**：计数器值的最低有效部分 (Least significant part of counter value)

不抖动模式 (DITHEN = 0)

该寄存器保存计数器值。

抖动模式 (DITHEN = 1)

该寄存器在 CNT[30:0] 中保存不抖动部分。小数部分不保存。

27.5.14 TIMx 预分频器 (TIMx_PSC)（x = 2、3）

TIMx prescaler

偏移地址：0x028

复位值：0x0000

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

位 15:0 **PSC[15:0]**：预分频器值 (Prescaler value)

计数器时钟频率 tim_cnt_ck 等于 $f_{tim_psc_ck}$ / (PSC[15:0] + 1)。

PSC 包含每次发生更新事件（包括计数器通过 TIMx_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）时要装载到实际预分频器寄存器的值。

27.5.15 TIM3 自动重载寄存器 (TIM3_ARR)

TIM3 auto-reload register

偏移地址：0x02C

复位值：0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[19:16]			
												rw	rw	rw	rw

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

位 31:20 保留，必须保持复位值。

位 19:0 **ARR[19:0]**：低自动重载值 (Low Auto-reload value)

ARR 为要装载到实际自动重载寄存器的值。

有关 ARR 更新和行为的更多详细信息，请参见*第 824 页的第 27.4.3 节：时基单元*。

当自动重载值为空时，计数器不工作。

不抖动模式 (DITHEN = 0)

该寄存器保存自动重载值。

抖动模式 (DITHEN = 1)

该寄存器在 ARR[19:4] 中保存整数部分。ARR[3:0] 位域包含抖动部分。

27.5.16 TIM2 自动重载寄存器 (TIM2_ARR)

TIM2 auto-reload register

偏移地址：0x02C

复位值：0xFFFF FFFF

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

位 31:0 **ARR[31:0]**：自动重载值 (Auto-reload value)

ARR 为要装载到实际自动重载寄存器的值。

有关 ARR 更新和行为的更多详细信息，请参见*第 824 页的第 27.4.3 节：时基单元*。

当自动重载值为空时，计数器不工作。

不抖动模式 (DITHEN = 0)

该寄存器保存自动重载值。

抖动模式 (DITHEN = 1)

该寄存器在 ARR[31:4] 中保存整数部分。ARR[3:0] 位域包含抖动部分。

27.5.17 TIM3 捕获/比较寄存器 1 (TIM3_CCR1)

TIM3 capture/compare register 1

偏移地址：0x034

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **CCR1[19:0]**：捕获/比较 1 值 (Capture/compare 1 value)

如果通道 CC1 配置为输出：

CCR1 是捕获/比较寄存器 1 的预装载值。如果没有通过 TIMx_CCMR1 寄存器中的 OC1PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 1）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc1 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器在 CCR1[15:0] 中保存比较值。CCR1[19:16] 位复位。

抖动模式 (DITHEN = 1)

该寄存器在 CCR1[19:4] 中保存整数部分。CCR1[3:0] 位域包含抖动部分。

如果通道 CC1 配置为输入：

CCR1 为最后一个输入捕获 1 事件 (tim_ic1) 发生时的计数器值。只能读取 TIMx_CCR1 寄存器，无法对其进行编程。

不抖动模式 (DITHEN = 0)

CCR1[15:0] 位保存捕获值。CCR1[19:16] 位保留。

抖动模式 (DITHEN = 1)

该寄存器在 CCR1[19:0] 中保存捕获值。CCR1[3:0] 位复位。

27.5.18 TIM2 捕获/比较寄存器 1 (TIM2_CCR1)

TIM2 capture/compare register 1

偏移地址：0x034

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1[31:16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **CCR1[31:0]**：捕获/比较 1 值 (Capture/compare 1 value)

如果通道 CC1 配置为输出：

CCR1 是捕获/比较寄存器 1 的预装载值。如果没有通过 TIMx_CCMR1 寄存器中的 OC1PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 1）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc1 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器保存比较值。

抖动模式 (DITHEN = 1)

该寄存器在 CCR1[31:4] 中保存整数部分。CCR1[3:0] 位域包含抖动部分。

如果通道 CC1 配置为输入：

CCR1 为上一个输入捕获 1 事件 (tim_ic1) 发生时的计数器值。只能读取 TIMx_CCR1 寄存器，无法对其进行编程。

不抖动模式 (DITHEN = 0)

该寄存器保存捕获值。

抖动模式 (DITHEN = 1)

该寄存器在 CCR1[31:0] 中保存捕获值。CCR1[3:0] 位复位。

27.5.19 TIM3 捕获/比较寄存器 2 (TIM3_CCR2)

TIM3 capture/compare register 2

偏移地址：0x038

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **CCR2[19:0]**：捕获/比较 1 值 (Capture/compare 1 value)

如果通道 CC2 配置为输出：

CCR2 是捕获/比较寄存器 2 的预装载值。如果没有通过 TIMx_CCMR2 寄存器中的 OC2PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 2）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc2 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器在 CCR2[15:0] 中保存比较值。CCR2[19:16] 位复位。

抖动模式 (DITHEN = 1)

该寄存器在 CCR2[19:4] 中保存整数部分。CCR2[3:0] 位域包含抖动部分。

如果通道 CC2 配置为输入：

CCR2 为上一个输入捕获 2 事件 (tim_ic2) 发生时的计数器值。只能读取 TIMx_CCR2 寄存器，无法对其进行编程。

不抖动模式 (DITHEN = 0)

CCR2[15:0] 位保存捕获值。CCR2[19:16] 位保留。

抖动模式 (DITHEN = 1)

该寄存器在 CCR2[19:0] 中保存捕获值。CCR2[3:0] 位复位。

27.5.20 TIM2 捕获/比较寄存器 2 (TIM2_CCR2)

TIM2 capture/compare register 2

偏移地址：0x038

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2[31:16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **CCR2[31:0]**：捕获/比较 2 值 (Capture/compare 2 value)

如果通道 CC2 配置为输出：

CCR2 是捕获/比较寄存器 2 的预装载值。如果没有通过 TIMx_CCMR2 寄存器中的 OC2PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 2）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc2 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器保存比较值。

抖动模式 (DITHEN = 1)

该寄存器在 CCR2[31:4] 中保存整数部分。CCR2[3:0] 位域包含抖动部分。

如果通道 CC2 配置为输入：

CCR2 为上一个输入捕获 2 事件 (tim_ic2) 发生时的计数器值。只能读取 TIMx_CCR2 寄存器，无法对其进行编程。

不抖动模式 (DITHEN = 0)

该寄存器保存捕获值。

抖动模式 (DITHEN = 1)

该寄存器在 CCR2[31:0] 中保存捕获值。CCR2[3:0] 位复位。

27.5.21 TIM3 捕获/比较寄存器 3 (TIM3_CCR3)

TIM3 capture/compare register 3

偏移地址：0x03C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR3[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **CCR3[19:0]**：捕获/比较 3 值 (Capture/compare 3 value)

如果通道 CC3 配置为输出：

CCR3 是捕获/比较寄存器 3 的预装载值。如果没有通过 TIMx_CCMR3 寄存器中的 OC3PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 3）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc3 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器在 CCR3[15:0] 中保存比较值。CCR3[19:16] 位复位。

抖动模式 (DITHEN = 1)

该寄存器在 CCR3[19:4] 中保存整数部分。CCR3[3:0] 位域包含抖动部分。

如果通道 CC3 配置为输入：

CCR3 为上一个输入捕获 3 事件 (tim_ic3) 发生时的计数器值。只能读取 TIMx_CCR3 寄存器，无法对其进行编程。

不抖动模式 (DITHEN = 0)

CCR3[15:0] 位保存捕获值。CCR3[19:16] 位保留。

抖动模式 (DITHEN = 1)

该寄存器在 CCR3[19:0] 中保存捕获值。CCR3[3:0] 位复位。

27.5.22 TIM2 捕获/比较寄存器 3 (TIM2_CCR3)

TIM2 capture/compare register 3

偏移地址：0x03C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3[31:16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **CCR3[31:0]**：捕获/比较 3 值 (Capture/compare 3 value)

如果通道 CC3 配置为输出：

CCR3 是捕获/比较寄存器 3 的预装载值。如果没有通过 TIMx_CCMR3 寄存器中的 OC3PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 3）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc3 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器保存比较值。

抖动模式 (DITHEN = 1)

该寄存器在 CCR3[31:4] 中保存整数部分。CCR3[3:0] 位域包含抖动部分。

如果通道 CC3 配置为输入：

CCR3 为上一个输入捕获 3 事件 (tim_ic3) 发生时的计数器值。只能读取 TIMx_CCR3 寄存器，无法对其进行编程。

不抖动模式 (DITHEN = 0)

该寄存器保存捕获值。

抖动模式 (DITHEN = 1)

该寄存器在 CCR3[31:0] 中保存捕获值。CCR3[3:0] 位复位。

27.5.23 TIM3 捕获/比较寄存器 4 (TIM3_CCR4)

TIM3 capture/compare register 4

偏移地址：0x040

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR4[19:16]			
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:0 **CCR4[19:0]**：捕获/比较 4 值 (Capture/compare 4 value)

如果通道 CC4 配置为输出：

CCR4 是捕获/比较寄存器 4 的预装载值。如果没有通过 TIMx_CCMR4 寄存器中的 OC4PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 4）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc4 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器在 CCR4[15:0] 中保存比较值。CCR4[19:16] 位复位。

抖动模式 (DITHEN = 1)

该寄存器在 CCR4[19:4] 中保存整数部分。CCR4[3:0] 位域包含抖动部分。

如果通道 CC4 配置为输入：

CCR4 为上一个输入捕获 4 事件 (tim_ic4) 发生时的计数器值。只能读取 TIMx_CCR4 寄存器，无法对其进行编程。

不抖动模式 (DITHEN = 0)

CCR4[15:0] 位保存捕获值。CCR4[19:16] 位保留。

抖动模式 (DITHEN = 1)

该寄存器在 CCR4[19:0] 中保存捕获值。CCR4[3:0] 位复位。

27.5.24 TIM2 捕获/比较寄存器 4 (TIM2_CCR4)

TIM2 capture/compare register 4

偏移地址：0x040

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4[31:16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **CCR4[31:0]**：捕获/比较 4 值 (Capture/compare 4 value)

如果通道 CC4 配置为输出：

CCR4 是捕获/比较寄存器 4 的预装载值。如果没有通过 TIMx_CCMR4 寄存器中的 OC4PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 4）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc4 输出相应电平的值。

不抖动模式 (DITHEN = 0)

该寄存器保存比较值。

抖动模式 (DITHEN = 1)

该寄存器在 CCR4[31:4] 中保存整数部分。CCR4[3:0] 位域包含抖动部分。

如果通道 CC4 配置为输入：

CCR4 为上一个输入捕获 4 事件 (tim_ic4) 发生时的计数器值。只能读取 TIMx_CCR4 寄存器，无法对其进行编程。

不抖动模式 (DITHEN = 0)

该寄存器保存捕获值。

抖动模式 (DITHEN = 1)

该寄存器在 CCR4[31:0] 中保存捕获值。CCR4[3:0] 位复位。

27.5.25 TIMx 定时器编码器控制寄存器 (TIMx_ECR)（x = 2、3）

TIMx timer encoder control register

偏移地址：0x058

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	PWPRSC[2:0]			PW[7:0]							
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IPOS[1:0]		FIDX	IBLK[1:0]		IDIR[1:0]		IE
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:27 保留，必须保持复位值。

位 26:24 **PWPRSC[2:0]**：脉冲宽度预分频器 (Pulse width prescaler)

该位域用于设置脉冲发生器的时钟预分频器，具体如下所示：

$t_{PWG} = (2^{(PWPRSC[2:0])}) \times t_{tim_ker_ck}$

位 23:16 **PW[7:0]**：脉冲宽度 (Pulse width)

该位域用于定义脉冲的持续时间，具体如下所示：

$t_{PW} = PW[7:0] \times t_{PWG}$

位 15:8 保留，必须保持复位值。

位 7:6 **IPOS[1:0]**：索引定位 (Index positioning)

在正交编码器模式下（SMS[3:0] = 0001、0010、0011、1110、1111），该位指示索引事件在哪种 AB 输入配置下复位计数器。

00：当 AB = 00 时，索引事件复位计数器
01：当 AB = 01 时，索引事件复位计数器
10：当 AB = 10 时，索引事件复位计数器
11：当 AB = 11 时，索引事件复位计数器

在定向时钟模式或时钟 + 方向模式（SMS[3:0] = 1010、1011、1100、1101）下，这些位指示索引事件在哪种电平状态下复位计数器。在双向时钟模式下，对两个时钟输入均适用。

x0：当时钟为低电平时，索引事件复位计数器
x1：当时钟为高电平时，索引事件复位计数器

注意：IPOS[1] 位无意义

位 5 **FIDX**：第一个索引 (First index)

该位指示是否只需考虑第一个索引

0：索引始终有效
1：第一个索引仅复位计数器

位 4:3 **IBLK[1:0]**：索引消隐 (Index blanking)

该位指示索引事件是否由 tim_ti3 输入触发

00：索引始终有效
01：根据 CC3P 位域，当 tim_ti3 输入有效时禁止索引
10：根据 CC4P 位域，当 tim_ti4 输入有效时禁止索引
11：保留

位 2:1 **IDIR[1:0]**：索引方向 (Index direction)
该位指示索引事件在哪种计数方向下复位计数器。
00：无论计数方向如何，索引事件都会复位计数器
01：索引事件仅在递增计数时复位计数器
10：索引事件仅在递减计数时复位计数器
11：保留

位 0 **IE**：索引使能 (Index enable)
该位指示索引事件是否复位计数器。
0：禁止索引
1：使能索引

27.5.26 TIMx 定时器输入选择寄存器 (TIMx_TISEL)（x = 2、3）

TIMx timer input selection register

偏移地址：0x05C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TI4SEL[3:0]				Res.	Res.	Res.	Res.	TI3SEL[3:0]			
				rw	rw	rw	rw					rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
				rw	rw	rw	rw					rw	rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27:24 **TI4SEL[3:0]**：选择 tim_ti4[15:0] 输入 (Selects tim_ti4[15:0] input)
0000：tim_ti4_in0：TIMx_CH4
0001：tim_ti4_in1
...
1111：tim_ti4_in15
有关产品特定的实现，请参见*第 27.4.2 节：TIM2/TIM3 引脚和内部信号*。

位 23:20 保留，必须保持复位值。

位 19:16 **TI3SEL[3:0]**：选择 tim_ti3[15:0] 输入 (Selects tim_ti3[15:0] input)
0000：tim_ti3_in0：TIMx_CH3
0001：tim_ti3_in1
...
1111：tim_ti3_in15
有关产品特定的实现，请参见*第 27.4.2 节：TIM2/TIM3 引脚和内部信号*。

位 15:12 保留，必须保持复位值。

位 11:8 **TI2SEL[3:0]**：选择 tim_ti2[15:0] 输入 (Selects tim_ti2[15:0] input)
0000：tim_ti2_in0：TIMx_CH2
0001：tim_ti2_in1
...
1111：tim_ti2_in15
有关产品特定的实现，请参见*第 27.4.2 节：TIM2/TIM3 引脚和内部信号*。

位 7:4 保留，必须保持复位值。

位 3:0 **TI1SEL[3:0]**：选择 tim_ti1[15:0] 输入 (Selects tim_ti1[15:0] input)
0000：tim_ti1_in0：TIMx_CH1
0001：tim_ti1_in1
...
1111：tim_ti1_in15
有关产品特定的实现，请参见*第 27.4.2 节：TIM2/TIM3 引脚和内部信号*。

27.5.27 TIMx 复用功能寄存器 1 (TIMx_AF1)（x = 2、3）

TIMx alternate function register 1

偏移地址：0x060

复位值：0x0000 0000

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

位 31:18 保留，必须保持复位值。

位 17:14 **ETRSEL[3:0]**：etr_in 源选择 (etr_in source selection)
这些位选择 etr_in 输入源。
0000：tim_etr0：TIMx_ETR 输入
0001：tim_etr1
...
1111：tim_etr15
有关产品特定的实现，请参见*第 27.4.2 节：TIM2/TIM3 引脚和内部信号*。

位 13:0 保留，必须保持复位值。

27.5.28 TIMx 复用功能寄存器 2 (TIMx_AF2)（x = 2、3）

TIMx alternate function register 2

偏移地址：0x064

复位值：0x0000 0000

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

位 31:19 保留，必须保持复位值。

位 18:16 **OCRSEL[2:0]**：ocref_clr 源选择 (ocref_clr source selection)

这些位选择 ocref_clr 输入源。

000：tim_ocref_clr0

001：tim_ocref_clr1

...

111：tim_ocref_clr7

有关产品特定的实现，请参见*第 27.4.2 节：TIM2/TIM3 引脚和内部信号*。

位 15:0 保留，必须保持复位值。

27.5.29 TIMx DMA 控制寄存器 (TIMx_DCR)（x = 2、3）

TIMx DMA control register

偏移地址：0x3DC

复位值：0x0000 0000

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

位 31:20 保留，必须保持复位值。

位 19:16 **DBSS[3:0]**：DMA 突发源选择 (DMA burst source selection)

该位域定义了触发 DMA 突发传输的中断源（当对 TIMx_DMAR 进行读或写访问时，定时器识别突发传输）。

0000：保留

0001：更新

0010：CC1

0011：CC2

0100：CC3

0101：CC4

0110：COM

0111：触发

其他值：保留

位 15:13 保留，必须保持复位值。

位 12:8 **DBL[4:0]**：DMA 突发传输长度 (DMA burst length)

该 5 位向量定义了 DMA 的传送长度（当对 TIMx_DMAR 地址进行读或写访问时，定时器识别突发传输），即传输的数量。可按半字或字节进行传送（请参见下面的示例）。

00000：1 次传输
00001：2 次传输
00010：3 次传输
...
11010：26 次传输

示例：考虑以下传输：DBL = 7 字节，DBA = TIM2_CR1。

– 如果 DBL = 7 字节且 DBA = TIM2_CR1 表示待传送字节的地址，则该地址可通过以下公式计算得出：

(TIMx_CR1 地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL

在本例中，将为 (TIMx_CR1 地址) + DBA 加上 7 个字节，得到将要复制数据的源/目标地址。这种情况下将向自以下地址开始的 7 个寄存器传输数据：(TIMx_CR1 地址) + DBA

根据 DMA 数据大小的配置，可能发生下面几种情况：

– 如果按半字配置 DMA 数据大小，则将向 7 个寄存器中的每一个传送 16 位数据。

– 如果按字节配置 DMA 数据大小，也将向 7 个寄存器传送数据：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，依此类推。因此，使用传送定时器时，还必须指定 DMA 传送的数据大小。

位 7:5 保留，必须保持复位值。

位 4:0 **DBA[4:0]**：DMA 基址 (DMA base address)

该 5 位向量定义 DMA 传输的基址（通过 TIMx_DMAR 地址进行读/写访问时）。DBA 定义为从 TIMx_CR1 寄存器地址开始计算的偏移量。

示例：

00000：TIMx_CR1，
00001：TIMx_CR2，
00010：TIMx_SMCR，
...

27.5.30 TIMx 全传输 DMA 地址 (TIMx_DMAR)（x = 2、3）

TIMx DMA address for full transfer

偏移地址：0x3E0

复位值：0x0000 0000

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

位 31:0 **DMAB[31:0]**：DMA 突发传输寄存器 (DMA register for burst accesses)

对 DMAR 寄存器执行读或写操作将访问位于如下地址的寄存器

(TIMx_CR1 地址) + (DBA + DMA 索引) x 4

其中 TIMx_CR1 地址为控制寄存器 1 的地址，DBA 为 TIMx_DCR 寄存器中配置的 DMA 基址，DMA 索引由 DMA 传输自动控制，其范围介于 0 到 DBL（TIMx_DCR 寄存器中配置的 DBL）之间。

27.5.31 TIMx 寄存器映射

TIMx 寄存器按照下表所述方式映射：

表 201. TIM2/TIM3 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**TIMx_CR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DITHEN	UIFREMA	Res.	CKD [1:0]		ARPE	CMS [1:0]		DIR	OPM	URS	UDIS	CEN
	Reset value																				0	0		0	0	0	0	0	0	0	0	0	0
0x004	**TIMx_CR2**	Res.	Res.	Res.	Res.	Res.	Res.	MMS[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1S	MMS[2:0]			CCDS	Res.	Res.	Res.
	Reset value							0																		0	0	0	0	0			
0x008	**TIMx_SMCR**	Res.	Res.	Res.	Res.	Res.	Res.	SMSPS	SMSPE	Res.	Res.	TS [4:3]		Res.	Res.	Res.	SMS[3]	ETP	ECE	ETPS [1:0]		ETF[3:0]				MSM	TS[2:0]			Res.	SMS[2:0]		
	Reset value							0	0			0	0				0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0
0x00C	**TIMx_DIER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TERRIE	IERRIE	DIRIE	IDXIE	Res.	Res.	Res.	Res.	Res.	Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset value								0	0	0	0							0		0	0	0	0	0		0		0	0	0	0	0
0x010	**TIMx_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TERRF	IERRF	DIRF	IDXF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	Res.	TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset value								0	0	0	0									0	0	0	0			0		0	0	0	0	0
0x014	**TIMx_EGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																										0		0	0	0	0	0
0x018	**TIMx_CCMR1** Input Capture mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2F[3:0]				IC2 PSC [1:0]		CC2S [1:0]		IC1F[3:0]				IC1 PSC [1:0]		CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIMx_CCMR1 Output Compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[3]	OC2CE	OC2M [2:0]			OC2PE	OC2FE	CC2S [1:0]		OC1CE	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
	Reset value								0								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x01C	**TIMx_CCMR2** Input Capture mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4F[3:0]				IC4 PSC [1:0]		CC4S [1:0]		IC3F[3:0]				IC3 PSC [1:0]		CC3S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIMx_CCMR2 Output Compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC3M[3]	O24CE	OC4M [2:0]			OC4PE	OC4FE	CC4S [1:0]		OC3CE	OC3M [2:0]			OC3PE	OC3FE	CC3S [1:0]	
	Reset value								0								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x020	**TIMx_CCER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4NP	Res.	CC4P	CC4E	CC3NP	Res.	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
	Reset value																	0		0	0	0		0	0	0		0	0	0		0	0

表 201. TIM2/TIM3 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x024	**TIMx_CNT**	UIFCPY_CNT[31]	CNT[30:16] (CNT[31:16] on 32-bit timers only)															CNT[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x028	**TIMx_PSC**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x02C	**TIMx_ARR** (x = 3)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[19:0]																			
	Reset value													0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x02C	**TIMx_ARR** (x = 2)	ARR[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x030	Reserved																																
0x034	**TIMx_CCR1**	CCR1[31:20] (32-bit timers only)												CCR1[19:0]																			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x038	**TIMx_CCR2**	CCR2[31:20] (32-bit timers only)												CCR2[19:0]																			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x03C	**TIMx_CCR3**	CCR3[31:20] (32-bit timers only)												CCR3[19:0]																			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x040	**TIMx_CCR4**	CCR4[31:20] (32-bit timers only)												CCR4[19:0]																			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x044.. 0x054	Reserved	Res.																															
0x058	**TIMx_ECR**	Res.	Res.	Res.	Res.	Res.	PWPRSC [2:0]			PW[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IPOS [1:0]		FIDX	IBLK [1:0]		IDIR [1:0]		IE
	Reset value						0	0	0	0	0	0	0	0	0	0	0									0	0	0	0	0	0	0	0
0x05C	**TIMx_TISEL**	Res.	Res.	Res.	Res.	TI4SEL[3:0]				Res.	Res.	Res.	Res.	TI3SEL[3:0]				Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
	Reset value					0	0	0	0					0	0	0	0					0	0	0	0					0	0	0	0
0x060	**TIMx_AF1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETRSEL [3:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value															0	0	0	0														
0x064	**TIMx_AF2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OCRSEL[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value														0	0	0																
0x068.. 0x3D8	Reserved	Res.																															
0x3DC	**TIMx_DCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBSS[3:0]				Res.	Res.	Res.	DBL[4:0]					Res.	Res.	Res.	DBA[4:0]				
	Reset value													0	0	0	0				0	0	0	0	0				0	0	0	0	0

表 201. TIM2/TIM3 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x3E0	**TIMx_DMAR**	DMAB[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

28 基本定时器 (TIM6/TIM7)

28.1 TIM6/TIM7 简介

基本定时器TIM6/TIM7 包含一个 16 位自动重载计数器，该计数器由可编程预分频器驱动。

基本定时器可用作通用定时器以生成时基。

基本定时器也可用于触发数模转换器。这是通过定时器的触发输出来完成的。

这些定时器彼此完全独立，不共享任何资源。

28.2 TIM6/TIM7 主要特性

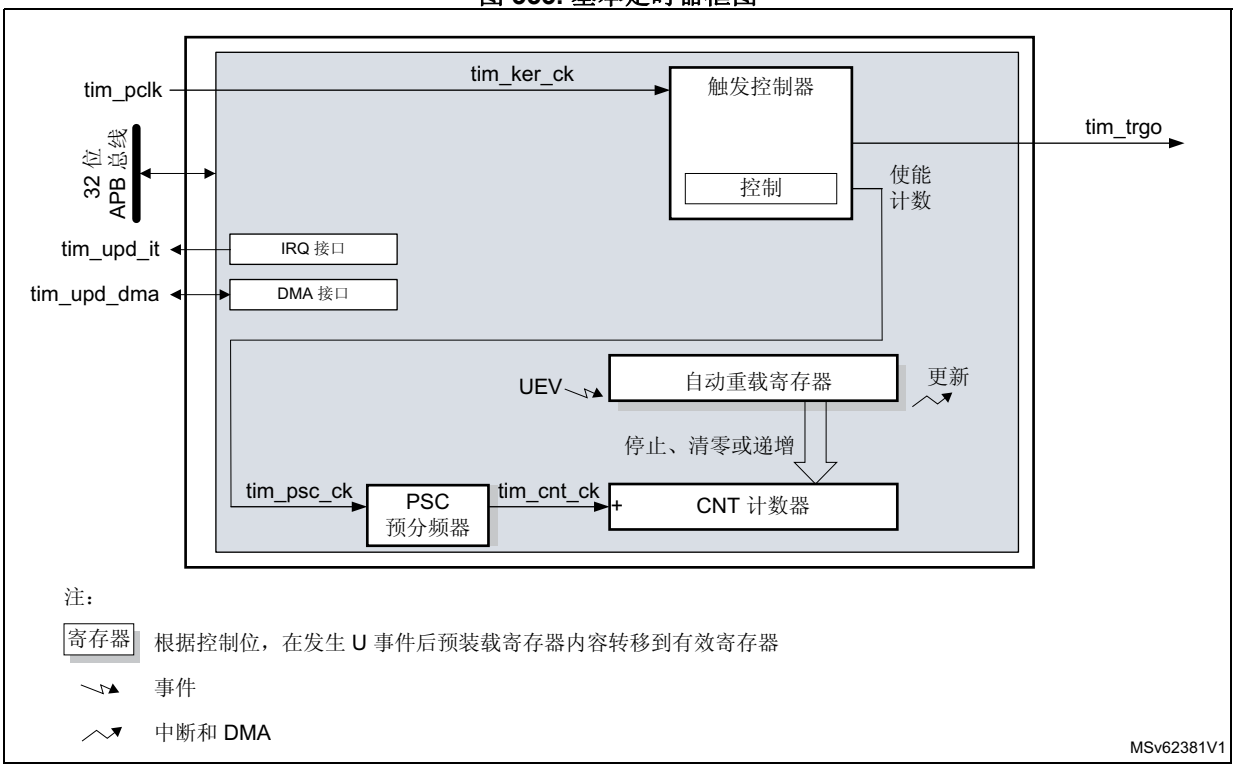
基本定时器 (TIM6/TIM7) 的特性包括：

- 16 位自动重载递增计数器
- 16 位可编程预分频器，用于对计数器时钟频率进行分频（可在运行时修改），分频系数介于 1 和 65535 之间
- 用于触发 DAC 的同步电路
- 发生如下更新事件时会生成中断/DMA 请求：计数器上溢

28.3 TIM6/TIM7 功能描述

28.3.1 TIM6/TIM7 框图

图 333. 基本定时器框图



28.3.2 TIM6/TIM7 内部信号

本节中的表格对 TIM 输入和输出进行了汇总。

表 202. TIM 内部输入/输出信号

内部信号名称	信号类型	说明
tim_pclk	输入	定时器 APB 时钟
tim_ker_ck	输入	定时器内核时钟。该时钟必须与 tim_pclk 同步（来自相同源）。时钟比 tim_ker_ck/tim_pclk 必须是整数：1、2、3 ... 16（最大值）
tim_trgo	输出	内部触发输出。该触发信号可触发其他片上外设 (DAC)。
tim_upd_it	输出	定时器更新事件中断
tim_upd_dma	输出	定时器更新 DMA 请求

28.3.3 TIM6/TIM7 时钟

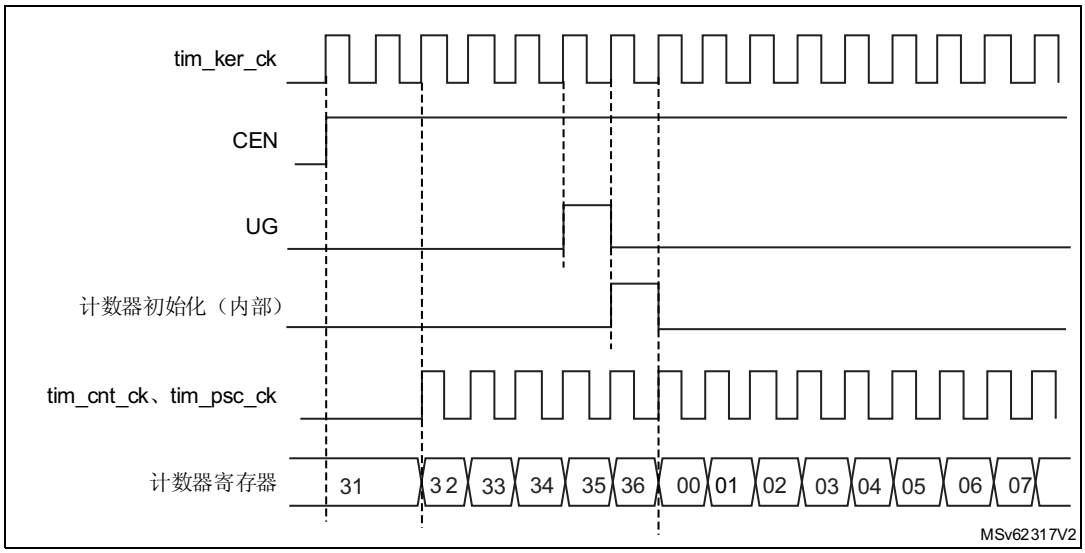
定时器总线接口由 tim_pclk APB 时钟提供时钟。

计数器时钟 tim_ker_ck 连接到 tim_pclk 输入。

CEN（TIMx_CR1 寄存器中）和 UG 位（TIMx_EGR 寄存器中）为实际控制位，并且只能通过软件进行更改（保持自动清零的 UG 除外）。当对 CEN 位写入 1 时，预分频器的时钟就由内部时钟 tim_ker_ck 提供。

图 334 显示了正常模式下控制电路与递增计数器的行为（没有预分频的情况下）。

图 334. 正常模式下的控制电路，1 分频内部时钟



28.3.4 时基单元

可编程定时器的主要模块由一个 16 位递增计数器及其相关的自动重载寄存器组成。计数器的时钟可通过预分频器进行分频。

计数器、自动重载寄存器和预分频器寄存器可通过软件进行读写。即使在计数器运行时也可执行读写操作。

时基单元包括：

- 计数器寄存器 (TIMx_CNT)
- 预分频器寄存器 (TIMx_PSC)
- 自动重载寄存器 (TIMx_ARR)

自动重载寄存器是预装载的。每次尝试对自动重载寄存器执行读写操作时，都会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件 UEV 时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位 (ARPE)。当计数器达到上溢值并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。下文将详细介绍各配置的更新事件的产生过程。

计数器由预分频器输出 tim_cnt_ck 提供时钟，仅当 TIMx_CR1 寄存器中的计数器使能位 (CEN) 置 1 时，才会使能计数器。

请注意，实际的计数器使能信号 tim_cnt_en 在 CEN 位置 1 的一个时钟周期后被置 1。

预分频器说明

预分频器可对计数器时钟进行分频，分频系数介于 1 和 65536 之间。该预分频器基于 TIMx_PSC 寄存器中的 16 位寄存器所控制的 16 位计数器。由于 TIMx_PSC 控制寄存器有缓冲，因此可对预分频器进行实时更改。而新的预分频比将在下一更新事件发生时被采用。

图 335 和 *图 336* 以一些示例说明在预分频比实时变化时计数器的行为。

图 335. 预分频器分频由 1 变为 2 时的计数器时序图

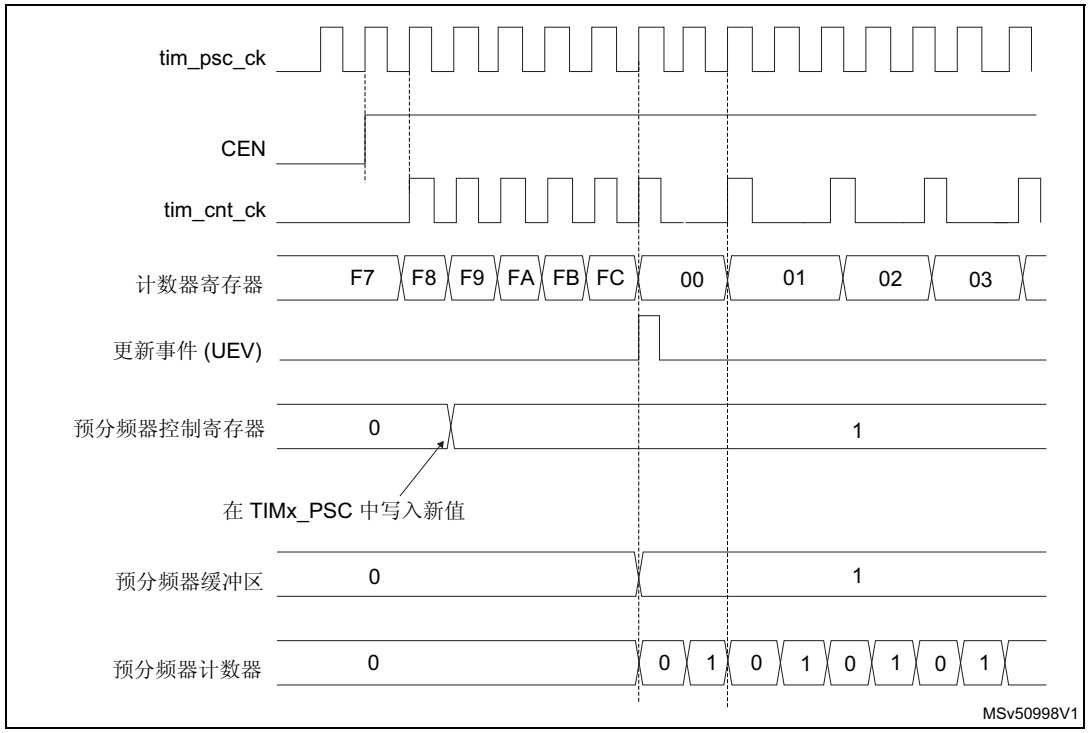
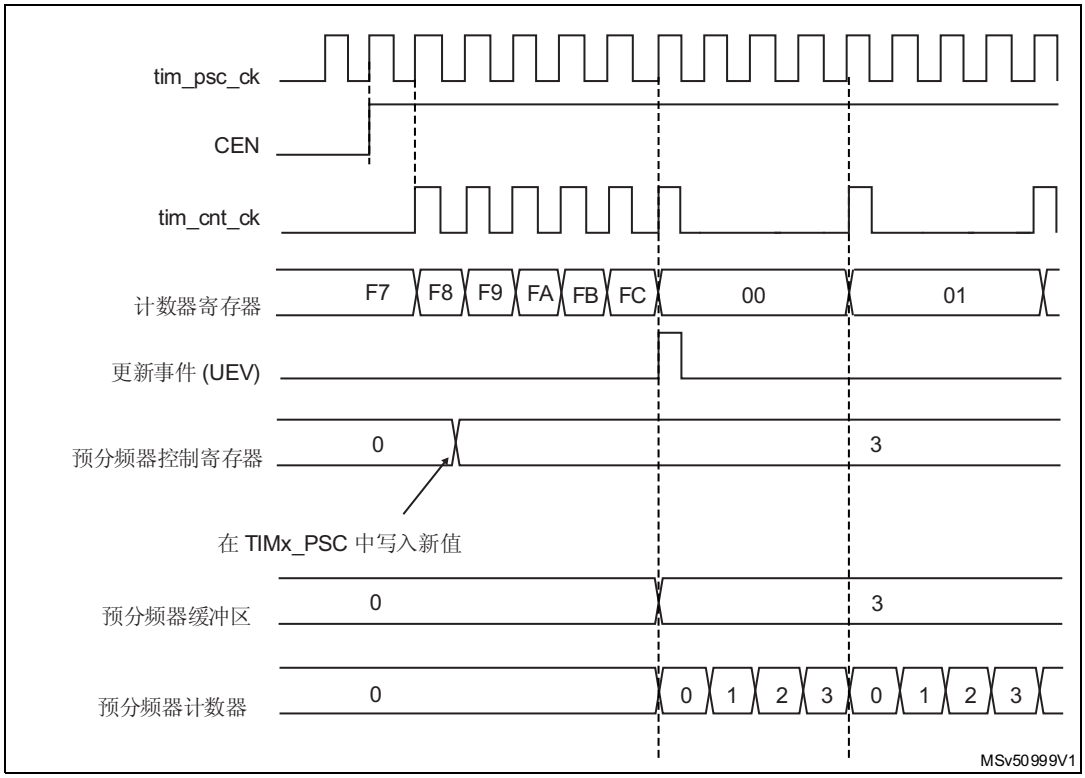


图 336. 预分频器分频由 1 变为 4 时的计数器时序图



28.3.5 计数模式

计数器从 0 计数到自动重载值（TIMx_ARR 寄存器的内容），然后重新从 0 开始计数并生成计数器上溢事件。

每次发生计数器上溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以产生更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。这样，直到 UDIS 位中写入 0 前便不会生成任何更新事件，但计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变）。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）
- 使用预装载值 (TIMx_ARR) 更新自动重载影子寄存器

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

图 337. 计数器时序图，1 分频内部时钟

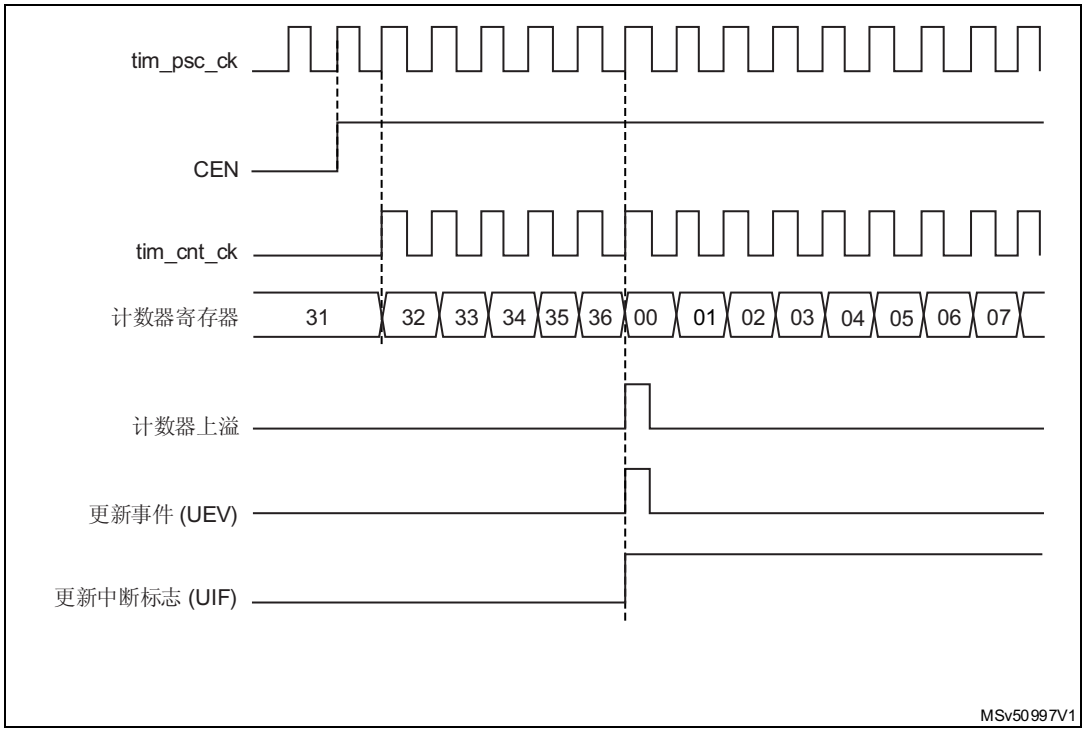


图 338. 计数器时序图，2 分频内部时钟

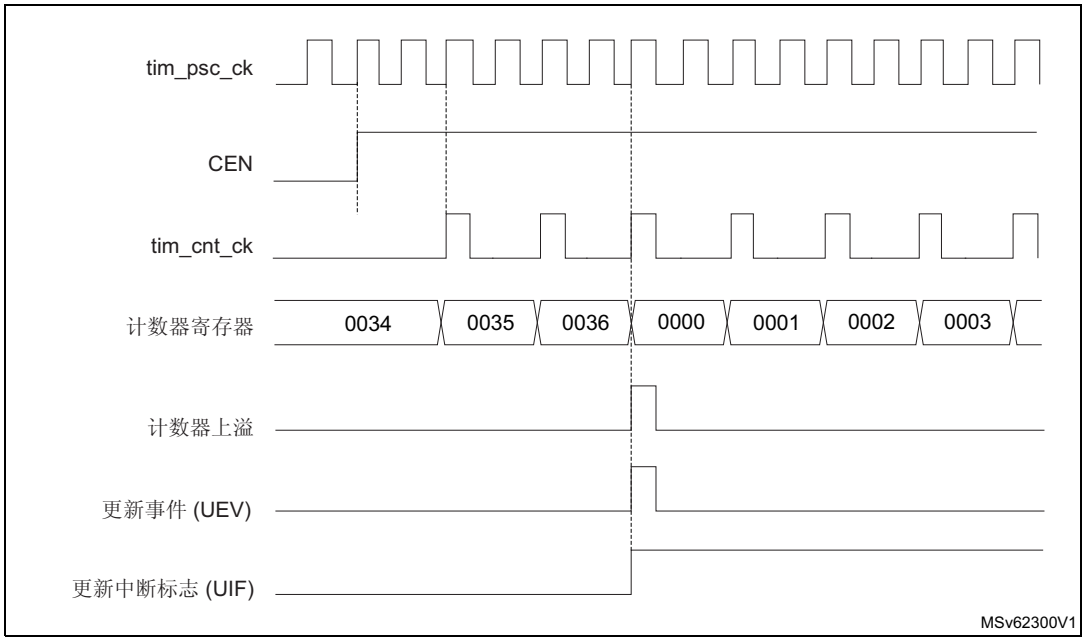


图 339. 计数器时序图，4 分频内部时钟

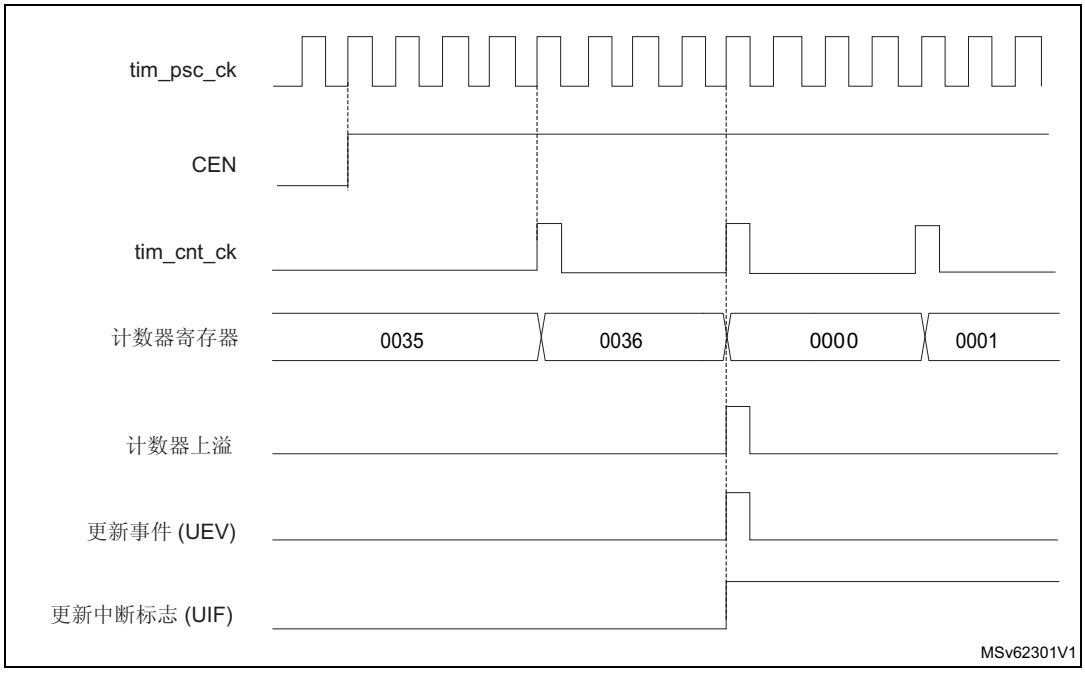


图 340. 计数器时序图，N 分频内部时钟

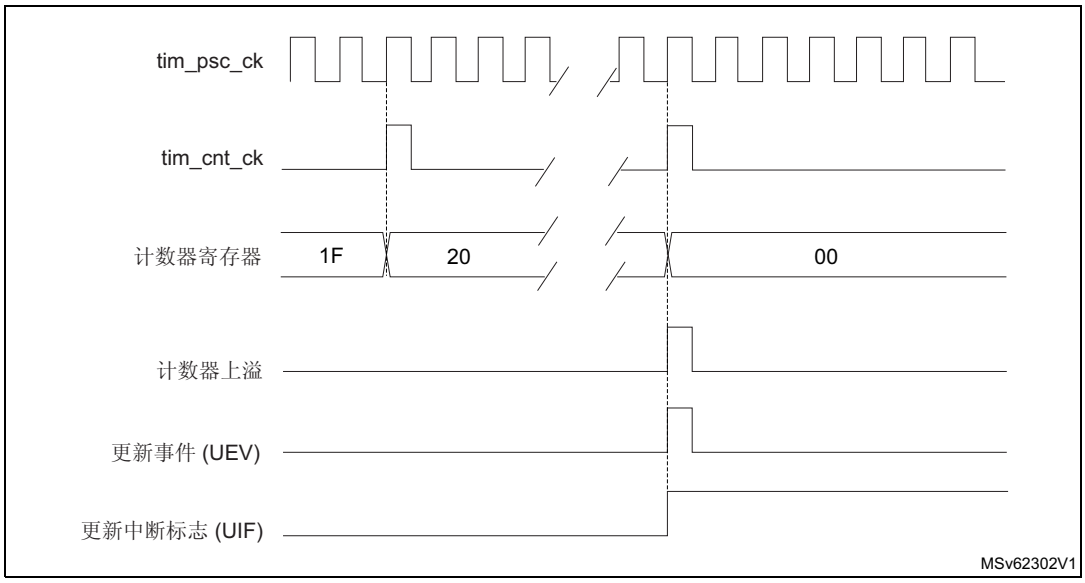


图 341. 计数器时序图，ARPE = 0 时更新事件（TIMx_ARR 未预装载）

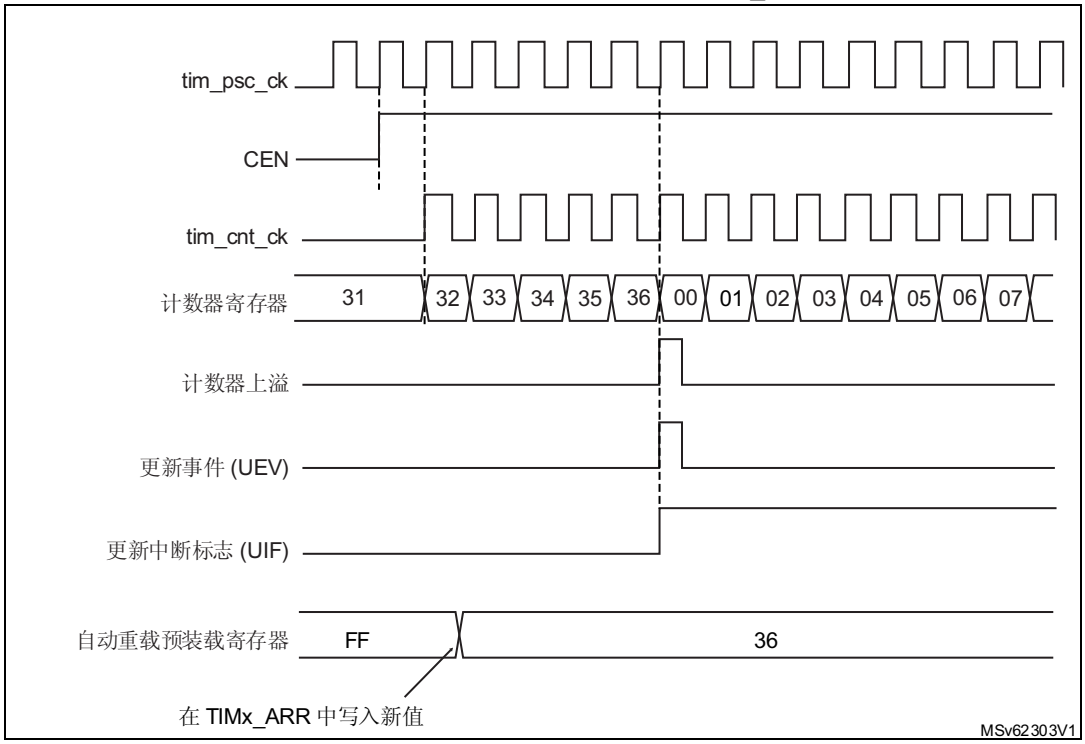
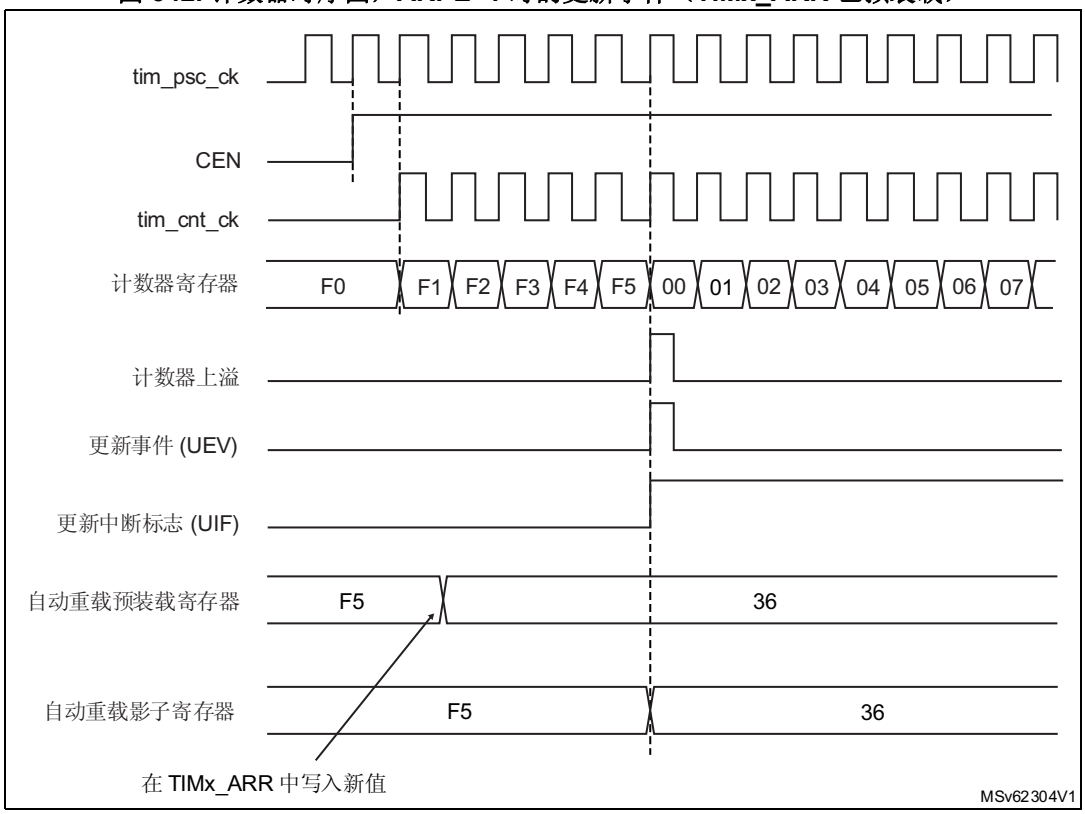


图 342. 计数器时序图，ARPE=1 时的更新事件（TIMx_ARR 已预装载）



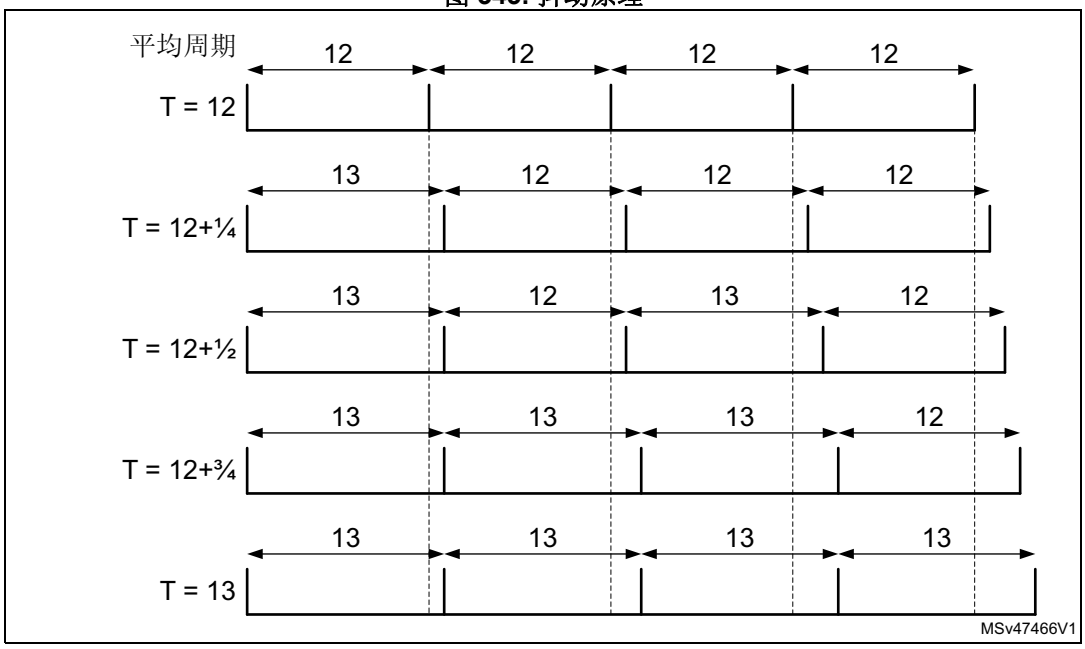
抖动模式

使用 TIMx_CR1 寄存器中的 DITHEN 位使能抖动模式，可以提高时基的有效分辨率。这会影响 TIMx_ARR 的行为方式，并且在定时器用作触发输出（通常用于 DAC）时对于调整平均计数器周期很有用。

工作原理如下：在 16 个连续的计数周期内按照预定义模式略微更改（增加或不增加一个定时器时钟周期）实际的 ARR 值。考虑到平均计数周期，这样可以将分辨率提高 16 倍。

下面的*图 343* 介绍了应用于 4 个连续计数周期的抖动原理。

图 343. 抖动原理

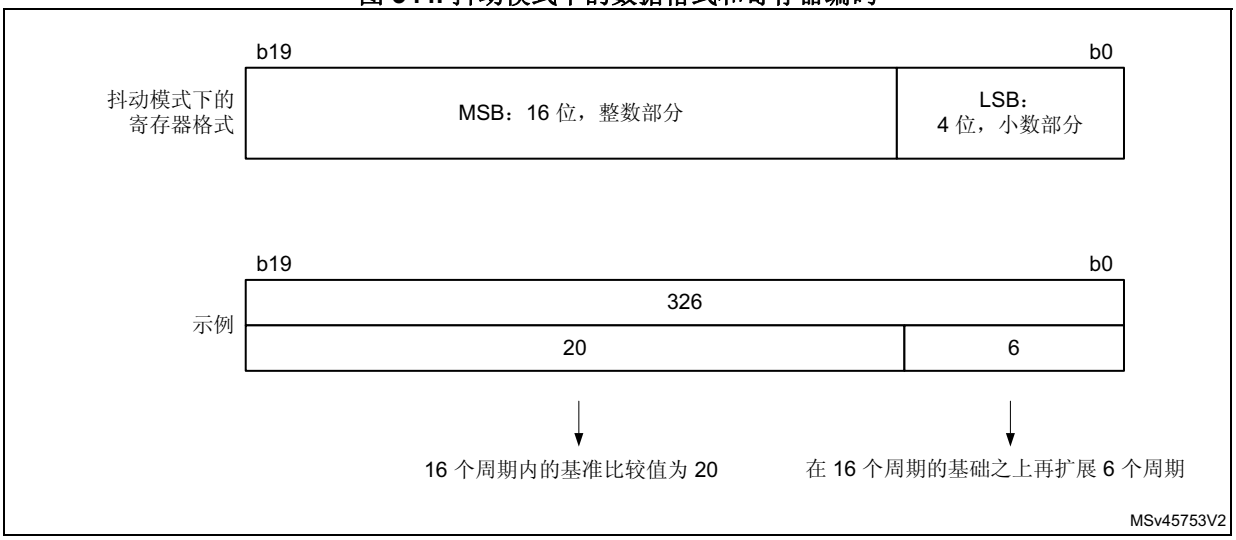


使能抖动模式后，寄存器编码的变化如下（请参见*图 344* 获取示例）：

- 4 个 LSB 用于对提高的分辨率部分（小数部分）进行编码
- MSB 左移至位 19:4，用于对基值进行编码

注意： *复位 DITHEN 位时必须遵循以下顺序：*
1. CEN 位和 ARPE 位必须复位
2. DITHEN 位必须复位
3. CEN 位可以置 1（最终 ARPE = 1）。

图 344. 抖动模式下的数据格式和寄存器编码



最低频率可根据以下公式得出：

$$\text{分辨率} = \frac{F_{Tim}}{F_{pwm}} \Rightarrow F_{pwmMin} = \frac{F_{Tim}}{Max_{Resolution}}$$

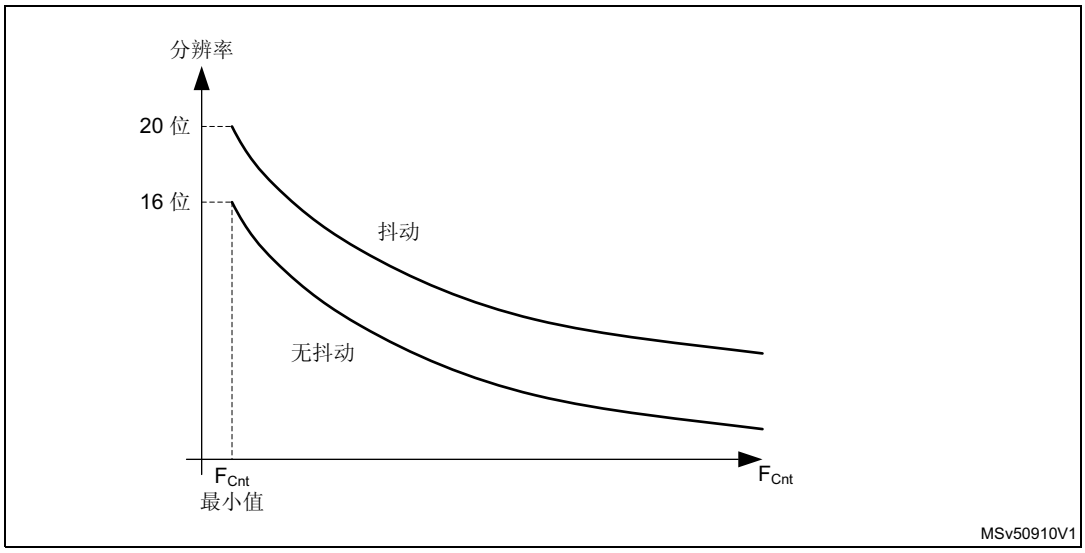
$$\text{禁止抖动模式：} F_{pwmMin} = \frac{F_{Tim}}{65536}$$

$$\text{使能抖动模式：} F_{pwmMin} = \frac{F_{Tim}}{65535 + \frac{15}{16}}$$

注意： *在抖动模式下，最大 TIMx_ARR 值限制为 0xFFFEF，该值对应于 65534（整数部分）和 15（抖动部分）。*

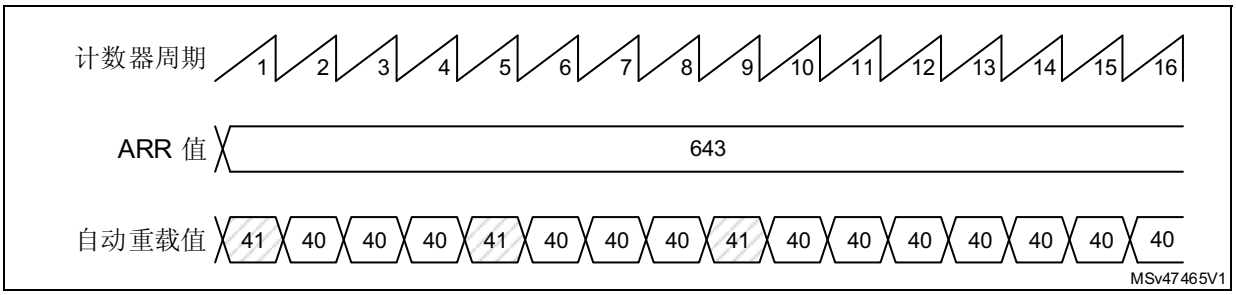
如下面的*图 345* 所示，无论 PWM 频率如何，抖动模式都可以用于提高 PWM 分辨率。

图 345. F_{Cnt} 分辨率与频率的关系



下面的*图 346* 详细描述了 16 个连续周期内的变化情况。

图 346. PWM 抖动模式



自动重载和比较值增量在特定模式下的分布情况如下面的*表 203* 所示。抖动序列的目的是尽可能均匀地分布增量并将整体纹波降至最低。

表 203. TIMx_ARR 寄存器变化抖动模式

-	PWM 周期															
LSB 值	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

28.3.6 UIF 位重映射

TIMx_CR1 寄存器中的 IUFREMAP 位强制将更新中断标志 UIF 连续复制到定时器计数器寄存器的位 31 (TIMxCNT[31]) 中。该功能用于按原子方式读取计数器值以及由 UIFCPY 标志发出的更新事件情况。在特定情况下，这可避免在后台任务（计数器读）和中断（更新中断）之间共享处理时产生竞争条件，从而简化计算。

UIF 和 UIFCPY 标志之间的置位没有延迟。

28.3.7 ADC 触发

定时器可通过多种内部信号产生 ADC 触发事件，例如复位、使能或比较事件。

注意： *必须先使能接收 tim_trgo 信号的从外设（例如定时器和 ADC）的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改时钟频率（预分频器）。*

28.3.8 TIM6/TIM7 DMA 请求

TIM6/TIM7 可生成单个 DMA 请求，如*表 204* 所示。

表 204. DMA 请求

DMA 缩写	DMA 请求	使能控制位
tim_upd_dma	更新	UDE

28.3.9 调试模式

当微控制器进入调试模式（Cortex-M33 内核停止）时，TIMx 计数器可以继续正常工作，也可以停止工作。

调试模式下的行为可以使用调试支持 (DBG) 模块中每个定时器的专用配置位进行编程。

更多详细信息，请参见调试支持 (DBG) 部分。

28.3.10 TIM6/TIM7 低功耗模式

表 205. 低功耗模式对 TIM6/TIM7 的影响

模式	说明
睡眠	无影响，外设处于激活状态。中断可使器件退出睡眠模式。
停止	定时器停止工作，寄存器的内容保持不变。不产生任何中断。
待机	定时器掉电，退出待机模式后必须重新初始化。

28.3.11 TIM6/TIM7 中断

TIM6/TIM7 可产生单个中断，如*表 206* 所示。

表 206. 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止模式和待机模式
TIM6 TIM7	更新	UIF	UIE	向 UIF 中写入 0	是	否

28.4 TIM6/TIM7 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节*。

外设寄存器可按半字（16 位）或字（32 位）访问。

28.4.1 TIMx 控制寄存器 1 (TIMx_CR1)（x = 6 到 7）

TIMx control register 1

偏移地址：0x00

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DITHEN	UIFREMAP	Res.	Res.	Res.	ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
			rw	rw				rw				rw	rw	rw	rw

位 15:13 保留，必须保持复位值。

位 12 **DITHEN**：抖动使能 (Dithering enable)
0：禁止抖动
1：使能抖动
注意：只有在 CEN 位复位时才能修改 DITHEN 位。

位 11 **UIFREMAP**：UIF 状态位重映射 (UIF status bit remapping)
0：无重映射。UIF 状态位不复制到 TIMx_CNT 寄存器的位 31。
1：使能重映射。UIF 状态位复制到 TIMx_CNT 寄存器的位 31。

位 10:8 保留，必须保持复位值。

位 7 **ARPE**：自动重载预装载使能 (Auto-reload preload enable)
0：TIMx_ARR 寄存器不进行缓冲。
1：TIMx_ARR 寄存器进行缓冲。

位 6:4 保留，必须保持复位值。

位 3 **OPM**：单脉冲模式 (One-pulse mode)
0：计数器在发生更新事件时不会停止计数
1：计数器在发生下一更新事件时停止计数（将 CEN 位清零）。

位 2 **URS**：更新请求源 (Update request source)
该位由软件置 1 和清零，用以选择 UEV 事件源。
0：使能时，所有以下事件都会生成更新中断或 DMA 请求。此类事件包括：
– 计数器上溢/下溢
– 将 UG 位置 1
– 通过从模式控制器生成的更新事件
1：使能时，只有计数器上溢/下溢会生成更新中断或 DMA 请求。

位 1 **UDIS**：更新禁止 (Update disable)

该位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新 (UEV) 事件可通过以下事件之一生成：

– 计数器上溢/下溢

– 将 UG 位置 1

– 通过从模式控制器生成的更新事件

带缓冲的寄存器被加载为预加载数值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR 和 PSC）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

位 0 **CEN**：计数器使能 (Counter enable)

0：禁止计数器

1：使能计数器

在单脉冲模式下，当发生更新事件时会自动将 CEN 位清零。

28.4.2 TIMx 控制寄存器 2 (TIMx_CR2)（x = 6 到 7）

TIMx control register 2

偏移地址：0x04

复位值：0x0000

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

位 15:7 保留，必须保持复位值。

位 6:4 **MMS[2:0]**：主模式选择 (Master mode selection)

这些位用于选择主模式下将要发送到从定时器以实现同步的信息 (TRGO)。这些位的组合如下：

000：**复位**——TIMx_EGR 寄存器中的 UG 位用作触发输出 (tim_trgo)。

001：**使能**——计数器使能信号 tim_cnt_en 用作触发输出 (tim_trgo)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。写入 CEN 控制位后，将产生计数器使能信号。

010：**更新**——选择更新事件作为触发输出 (tim_trgo)。例如，主定时器可用作从定时器的预分频器。

注意：必须先使能从定时器或接收 tim_trgo 的外设的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改从定时器或该外设的时钟。

位 3:0 保留，必须保持复位值。

28.4.3 TIMx DMA/中断使能寄存器 (TIMx_DIER)（x = 6 到 7）

TIMx DMA/Interrupt enable register

偏移地址：0x0C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIE
							rw								rw

位 15:9 保留，必须保持复位值。

位 8 **UDE**：更新 DMA 请求使能 (Update DMA request enable)

0：禁止更新 DMA 请求。
1：使能更新 DMA 请求。

位 7:1 保留，必须保持复位值。

位 0 **UIE**：更新中断使能 (Update interrupt enable)

0：禁止更新中断。
1：使能更新中断。

28.4.4 TIMx 状态寄存器 (TIMx_SR)（x = 6 到 7）

TIMx status register

偏移地址：0x10

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIF
															rc_w0

位 15:1 保留，必须保持复位值。

位 0 **UIF**：更新中断标志 (Update interrupt flag)

该位在发生更新事件时通过硬件置 1。但需要通过软件清零。
0：未发生更新。
1：更新中断挂起。该位在以下情况下更新寄存器时由硬件置 1：
– 计数器上溢且当 TIMx_CR1 寄存器中 UDIS = 0 时。
– 如果 TIMx_CR1 寄存器中 URS = 0 且 UDIS = 0，通过软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。

28.4.5 TIMx 事件生成寄存器 (TIMx_EGR)（x = 6 到 7）

TIMx event generation register

偏移地址：0x14

复位值：0x0000

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

位 15:1 保留，必须保持复位值。

位 0 **UG**：更新生成 (Update generation)

该位可通过软件置 1，并由硬件自动清零。

0：不执行任何操作。

1：重新初始化定时器计数器并生成寄存器更新事件。请注意，预分频器计数器也将清零（但预分频比不受影响）。

28.4.6 TIMx 计数器 (TIMx_CNT)（x = 6 到 7）

TIMx counter

偏移地址：0x24

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UIF CPY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r															

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

位 31 **UIFCPY**：UIF 副本 (UIF copy)

该位是 TIMx_ISR 寄存器中 UIF 位的只读副本。如果 TIMx_CR1 中的 UIFREMAP 位复位，则位 31 保留，读为 0。

位 30:16 保留，必须保持复位值。

位 15:0 **CNT[15:0]**：计数器值 (Counter value)

不抖动模式 (DITHEN = 0)

该寄存器保存计数器值。

抖动模式 (DITHEN = 1)

该寄存器仅在 CNT[15:0] 中保存不抖动部分。小数部分不保存。

28.4.7 TIMx 预分频器 (TIMx_PSC)（x = 6 到 7）

TIMx prescaler

偏移地址：0x28

复位值：0x0000

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

位 15:0 **PSC[15:0]**：预分频器值 (Prescaler value)

计数器时钟频率 $f_{tim_cnt_ck}$ 等于 $f_{tim_psc_ck}$ / (PSC[15:0] + 1)。

PSC 包含在每次发生更新事件时要装载到有效预分频器寄存器的值。

（包括通过 TIMx_EGR 寄存器的 UG 位清零计数器时。）

28.4.8 TIMx 自动重载寄存器 (TIMx_ARR)（x = 6 到 7）

TIMx auto-reload register

偏移地址：0x2C

复位值：0x0000 FFFF

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

位 31:20 保留，必须保持复位值。

位 19:0 **ARR[19:0]**：自动重载值 (Auto-reload value)

ARR 为要装载到有效自动重载寄存器的值。

有关 ARR 更新和行为的详细信息，请参见*第 926 页的第 28.3.4 节：时基单元*。

当自动重载值为空时，计数器不工作。

不抖动模式 (DITHEN = 0)

该寄存器在 ARR[15:0] 中保存自动重载值。ARR[19:16] 位保留。

抖动模式 (DITHEN = 1)

该寄存器在 ARR[19:4] 中保存整数部分。ARR[3:0] 位域包含抖动部分。

28.4.9 TIMx 寄存器映射

TIMx 寄存器可映射为 16 位可寻址寄存器，如下表所述：

表 207. TIMx 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**TIMx_CR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DITHEN	UIFREMA	Res.	Res.	Res.	ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
	Reset value																				0	0				0				0	0	0	0
0x04	**TIMx_CR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MMS [2:0]			Res.	Res.	Res.	Res.
	Reset value																										0	0	0				
0x08	Reserved																																
0x0C	**TIMx_DIER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIE
	Reset value																								0								0
0x10	**TIMx_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIF
	Reset value																																0
0x14	**TIMx_EGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UG
	Reset value																																0
0x18-0x20	Reserved																																
0x24	**TIMx_CNT**	UIFCPY or Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	**TIMx_PSC**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	**TIMx_ARR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[19:0]																			
	Reset value													0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

29 低功耗定时器 (LPTIM)

29.1 简介

LPTIM 是一个 16 位定时器，可从降低功耗的最终发展中受益。由于 LPTIM 的时钟源具有多样性，因此 LPTIM 能够在所有电源模式（待机模式除外）下保持运行状态。即使没有内部时钟源，LPTIM 也能运行，鉴于这一点，可将其用作“脉冲计数器”，这种脉冲计数器在某些应用中十分有用。此外，LPTIM 还能将系统从低功耗模式唤醒，因此非常适合实现“超时功能”，而且功耗极低。

LPTIM 引入了一个灵活的时钟方案，该方案能够提供所需的功能和性能，同时还能最大程度地降低功耗。

29.2 LPTIM 主要特性

- 16 位递增计数器
- 3 位预分频器，可采用 8 种分频系数（1、2、4、8、16、32、64 和 128）
- 可选时钟
  - 内部时钟源：可配置的内部时钟源（请参见 RCC 部分）
  - LPTIM 输入的外部时钟源（在没有 LP 振荡器运行的情况下工作，由脉冲计数器应用使用）
- 16 位 ARR 自动重载寄存器
- 16 位捕获/比较寄存器
- 连续/单触发模式
- 可选软件/硬件输入触发
- 可编程数字防抖动干扰滤波器
- 可配置输出：脉冲、PWM
- 可配置 I/O 极性
- 编码器模式
- 重复计数器
- 多达 2 个独立通道，可用于：
  - 输入捕获
  - PWM 生成（边沿对齐模式）
  - 单脉冲模式输出
- 10 个中断产生事件
- 发生如下事件时生成 DMA 请求：
  - 更新事件
  - 输入捕获

29.3 LPTIM 实现

表 208 介绍了 STM32H503xx 器件上的 LPTIM 实现。

表 208. STM32H503xx LPTIM 特性

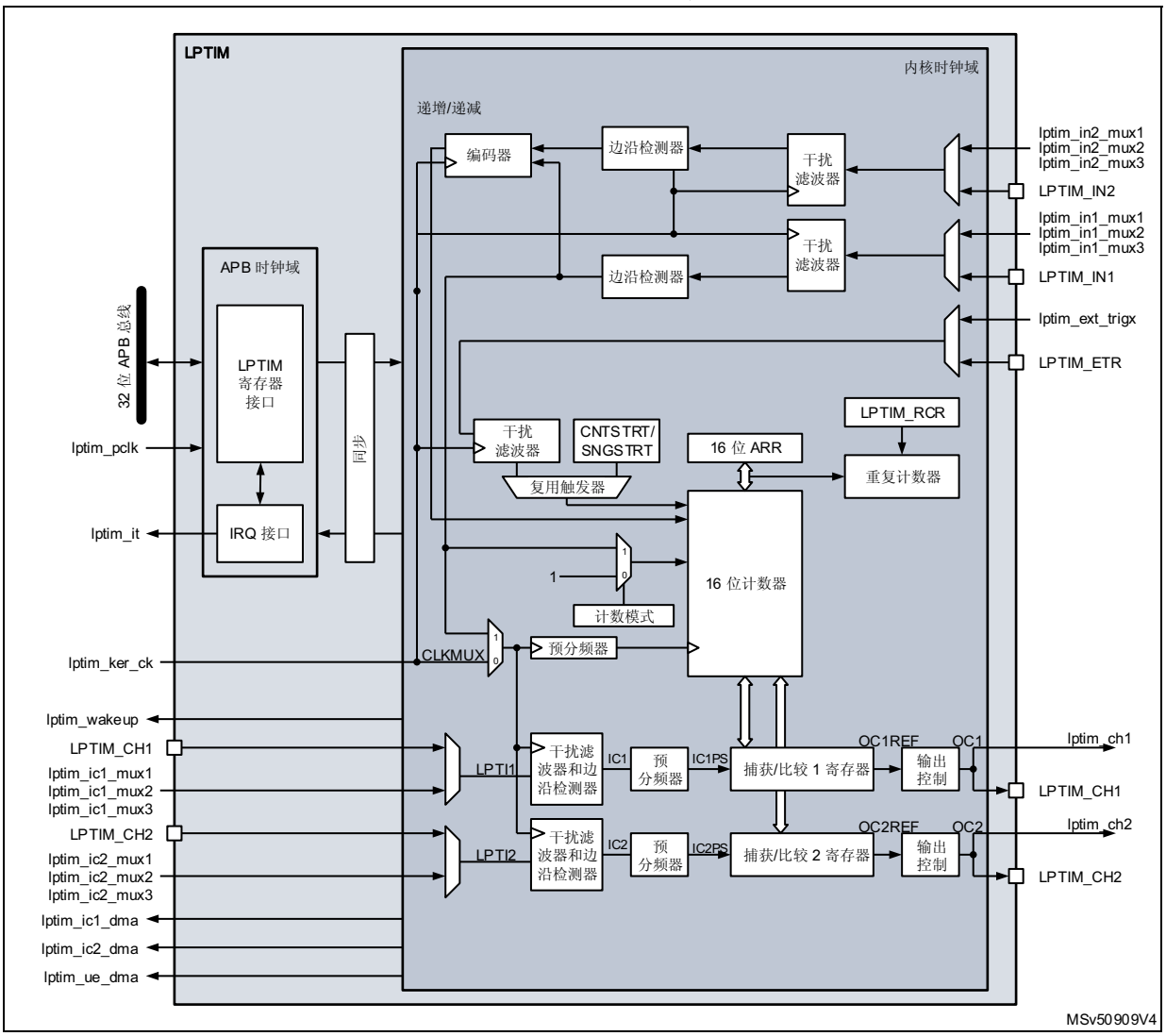
LPTIM 模式/特性(1)	LPTIM1	LPTIM2
编码器模式	X	X
PWM 模式	X	X
输入捕获	X	X
通道数	2	2
DMA 请求数	3	3
从停止模式唤醒	X	X

1. X = 支持。

29.4 LPTIM 功能描述

29.4.1 LPTIM 框图

图 347. LPTIM1/2 定时器框图(1)



1. 一些 IO 可能不可用，请参见*第 29.4.2 节：LPTIM 引脚和内部信号*。

29.4.2 LPTIM 引脚和内部信号

以下各表分别列出了 LPTIM 引脚和内部信号。

表 209. LPTIM1/2 输入/输出引脚

名称	信号类型	说明
LPTIM_IN1	数字输入	复用器输入 0 上 GPIO 引脚的 LPTIM 输入 1
LPTIM_IN2	数字输入	复用器输入 0 上 GPIO 引脚的 LPTIM 输入 2
LPTIM_ETR	数字输入	LPTIM 外部触发 GPIO 引脚
LPTIM_CH1	数字输入/输出	LPTIM 通道 1 输入/输出 GPIO 引脚
LPTIM_CH2	数字输入/输出	LPTIM 通道 2 输入/输出 GPIO 引脚

表 210. LPTIM1/2 内部信号

名称	信号类型	说明
lptim_pclk	数字输入	LPTIM APB 时钟域
lptim_ker_ck	数字输入	LPTIM 内核时钟
lptim_in1_mux1	数字输入	连接至复用器输入 1 的内部 LPTIM 输入 1
lptim_in1_mux2	数字输入	连接至复用器输入 2 的内部 LPTIM 输入 1
lptim_in1_mux3	数字输入	连接至复用器输入 3 的内部 LPTIM 输入 1
lptim_in2_mux1	数字输入	连接至复用器输入 1 的内部 LPTIM 输入 2
lptim_in2_mux2	数字输入	连接至复用器输入 2 的内部 LPTIM 输入 2
lptim_in2_mux3	数字输入	连接至复用器输入 3 的内部 LPTIM 输入 2
lptim_ic1_mux1	数字输入	连接至复用器输入 1 的内部 LPTIM 输入捕获 1
lptim_ic1_mux2	数字输入	连接至复用器输入 2 的内部 LPTIM 输入捕获 1
lptim_ic1_mux3	数字输入	连接至复用器输入 3 的内部 LPTIM 输入捕获 1
lptim_ic2_mux1	数字输入	连接至复用器输入 1 的内部 LPTIM 输入捕获 2
lptim_ic2_mux2	数字输入	连接至复用器输入 2 的内部 LPTIM 输入捕获 2
lptim_ic2_mux3	数字输入	连接至复用器输入 3 的内部 LPTIM 输入捕获 2
lptim_ext_trigx	数字输入	LPTIM 外部触发输入 x
lptim_it	数字输出	LPTIM 全局中断
lptim_wakeup	数字输出	LPTIM 唤醒事件
lptim_ic1_dma	数字输出	LPTIM 输入捕获 1 DMA 请求
lptim_ic2_dma	数字输出	LPTIM 输入捕获 2 DMA 请求
lptim_ue_dma	数字输出	LPTIM 更新事件 DMA 请求

29.4.3 LPTIM 输入和触发映射

下面详细给出了 LPTIM 外部触发连接和输入连接。

表 211. LPTIM1/2 外部触发连接

TRIGSEL	外部触发信号	
	LPTIM1	LPTIM2
lptim_ext_trig0	GPIO	
lptim_ext_trig1	rtc_alra_trg	
lptim_ext_trig2	rtc_alrb_trg	
lptim_ext_trig3	tamp_trg1	
lptim_ext_trig4	tamp_trg2	gpdma_ch0_tcf
lptim_ext_trig5	gpdma_ch1_tcf	gpdma_ch4_tcf
lptim_ext_trig6	comp1_out	
lptim_ext_trig7	EVENTOUT	

表 212. LPTIM1/2 输入 1 连接

lptim_in1_mux	LPTIM1 输入 1 连接位置	LPTIM2 输入 1 连接位置
lptim_in1_mux0	GPIO	
lptim_in1_mux1	comp1_out	
lptim_in1_mux2	lptim2_ch1	lptim1_ch2
lptim_in1_mux3	-	

表 213. LPTIM1/2 输入 2 连接

lptim_in2_mux	LPTIM1/2 输入 2 连接位置
lptim_in2_mux0	GPIO
lptim_in2_mux1	-
lptim_in2_mux2	-
lptim_in2_mux3	-

表 214. LPTIM1/2 输入捕获 1 连接

lptim_ic1_mux	LPTIM1 输入捕获 1 连接位置	LPTIM2 输入捕获 1 连接位置
lptim_ic1_mux0	GPIO	
lptim_ic1_mux1	comp1_out	
lptim_ic1_mux2	EVENTOUT	
lptim_ic1_mux3	RCC_MCO1	RCC_MCO2

表 215. LPTIM1/2 输入捕获 2 连接

lptim_ic2_mux	LPTIM1 输入捕获 2 连接位置	LPTIM2 输入捕获 2 连接位置
lptim_ic2_mux0	I/O	
lptim_ic2_mux1	LSI	HSI/1024
lptim_ic2_mux2	LSE	CSI/128
lptim_ic2_mux3	RCC_HSE_1MHZ	HSI/8

29.4.4 LPTIM 复位和时钟

LPTIM 可通过多个时钟源提供时钟。它可以由内部时钟信号提供时钟，内部时钟信号可通过 RCC 来选择任意可配置的内部时钟源（更多详细信息，请参见 RCC 部分）。此外，LPTIM 还可通过注入到其外部 Input1 上的外部时钟信号提供时钟。当通过外部时钟源提供时钟时，LPTIM 可以在下述两种可能配置中的其中一种配置下运行：

- 第一种配置是，LPTIM 通过外部信号提供时钟，但同时通过可配置的内部时钟源为 LPTIM 提供内部时钟信号（请参见 RCC 部分）。
- 第二种配置是，LPTIM 仅由外部时钟源通过外部 Input1 提供时钟。此配置可在进入低功耗模式后所有内置振荡器关闭时，用于实现超时功能或脉冲计数器功能。

对 CKSEL 和 COUNTMODE 位进行编程，可控制 LPTIM 使用外部时钟源还是内部时钟源。

当使用外部时钟源时，可使用 CKPOL 位选择外部时钟信号的有效边沿。如果上升沿和下降沿均为有效边沿，则还必须提供内部时钟信号（第一种配置）。在这种情况下，内部时钟信号频率必须至少为外部时钟信号频率的五倍。

29.4.5 干扰滤波器

外部（映射到 GPIO）或内部（映射到芯片级或其他嵌入式外设）LPTIM 输入由数字滤波器保护，避免任何毛刺和噪声干扰在 LPTIM 内部传播，从而防止产生意外计数或触发。

在激活数字滤波器之前，必须先向 LPTIM 提供内部时钟源。这是保证滤波器正常工作的必要条件。

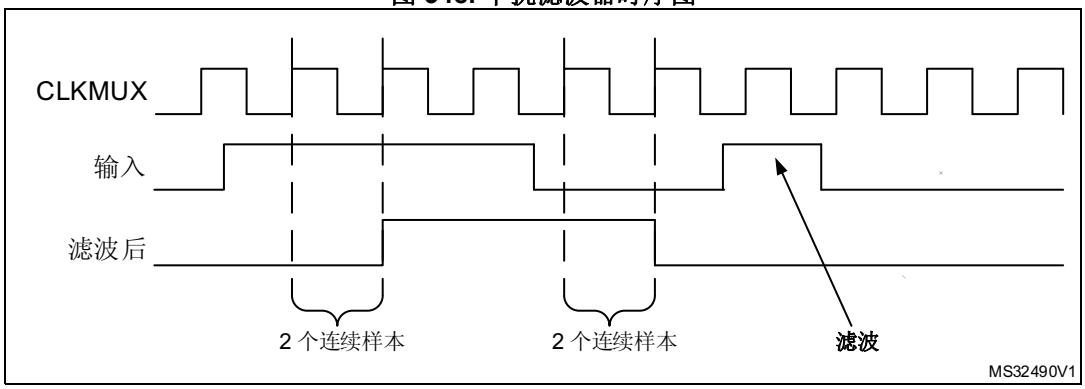
数字滤波器分为三组：

- 第一组数字滤波器保护 LPTIM 内部或外部输入。数字滤波器的敏感性由 CKFLT 位控制。
- 第二组数字滤波器保护 LPTIM 内部或外部触发输入。数字滤波器的敏感性由 TRGFLT 位控制。
- 第三组数字滤波器保护 LPTIM 内部或外部输入捕获。数字滤波器的敏感性由 ICxF 位控制。

注意： *数字滤波器的敏感性以组为单位进行控制。无法单独配置同一组内各个数字滤波器的敏感性。*

滤波器的敏感性会影响相同的连续采样的数量，在其中一个 LPTIM 输入上检测到此类连续采样时，才能将某信号电平变化视为有效跳变。*图 348* 给出了编程 2 个连续采样时，干扰滤波器行为的示例。

图 348. 干扰滤波器时序图



注意： *不提供内部时钟信号时，必须通过将 CKFLT、ICxF 和 TRGFLT 位设为 0 来停用数字滤波器。在这种情况下，可使用外部模拟滤波器来防止 LPTIM 外部输入产生干扰。*

29.4.6 预分频器

LPTIM 16 位计数器前面要有一个可配置的 2 次幂预分频器。预分频器的分频比由 PRESC[2:0] 3 位域进行控制。下表列出了所有可能的分频比：

表 216. 预分频器的分频比

编程	分频系数
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

29.4.7 触发多路复用器

LPTIM 计数器可通过软件启动，也可以在 8 个触发输入之一上检测到有效边沿后启动。

TRIGEN[1:0] 用于确定 LPTIM 触发源：

- TRIGEN[1:0] 等于“00”时，LPTIM 计数器会在通过软件将 CNTSTRT 位或 SNGSTRT 位其中之一置 1 后立即启动。TRIGEN[1:0] 的其余三个可能的值用于配置触发输入使用的有效边沿。LPTIM 计数器会在检测到有效边沿后立即启动。
- TRIGEN[1:0] 不等于“00”时，TRIGSEL[2:0] 用于选择使用 8 个触发输入中的哪一个来启动计数器。

外部触发信号视为 LPTIM 的异步信号。因此，检测到触发信号后，由于同步问题，需要延迟两个计数器时钟周期，定时器才能开始运行。

如果在定时器已启动时发生新的触发事件，则此事件将被忽略（除非已使能超时功能）。

注意： *必须使能定时器，才能将 SNGSTRT/CNTSTRT 位置 1。当定时器禁止时，对这些位执行的任何写操作都将被硬件丢弃。*

注意： *通过软件启动计数器后 (TRIGEN[1:0] = 00)，从 LPTIM_CR 寄存器更新（SNGSTRT 位或 CNTSTRT 位置 1）到计数器真正启动之间会有 3 个内核时钟周期的延迟。*

29.4.8 工作模式

LPTIM 支持以下两种工作模式：

- 连续模式：定时器自由运行，由触发事件启动并且直到被禁止才会停止。
- 单触发模式：定时器由触发事件启动，当生成 LPTIM 更新事件时停止。

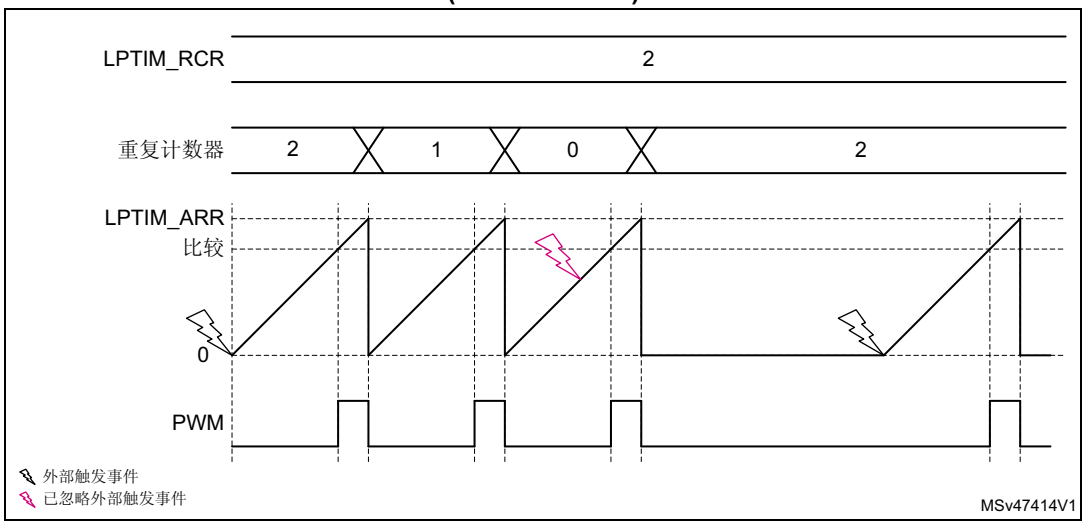
单触发模式

要使能单触发计数，必须将 SNGSTRT 位置 1。

新的触发事件将重新启动定时器。从计数器启动到生成下一个 LPTIM 更新事件，这段时间内发生的任何触发事件都将被丢弃。

选择外部触发时，每个外部触发事件将在 SNGSTRT 位置 1 后以及重复计数器停止后（更新事件后）到达，并且如果重复寄存器的内容不等于零，则将为重复计数器重新装载重复寄存器中包含的值，并启动新的单触发计数周期，如*图 349* 所示。

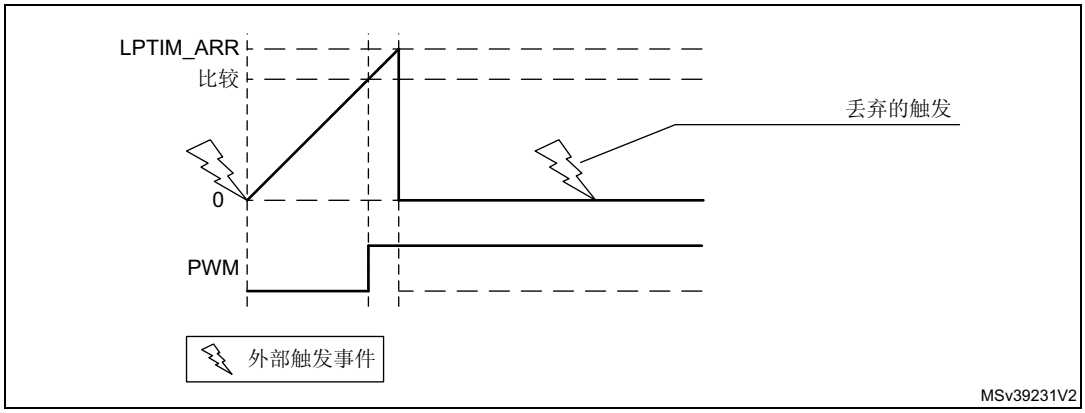
图 349. LPTIM 输出波形，单次计数模式配置（重复寄存器内容不等于零时）(PRELOAD = 1)



- 置 1 一次模式激活：

请注意，当 LPTIM_CFGR 寄存器中的 WAVE 位域置 1 时，将激活置 1 一次模式。在这种情况下，计数器仅会在第一个触发事件后启动一次，任何后续触发事件都将被丢弃，如*图 350* 所示。

图 350. LPTIM 输出波形，单次计数模式配置且激活置 1 一次模式（WAVE 位置 1）



若通过软件启动（TRIGEN[1:0] =“00”），将 SNGSTRT 置 1 会使计数器进行单触发计数。

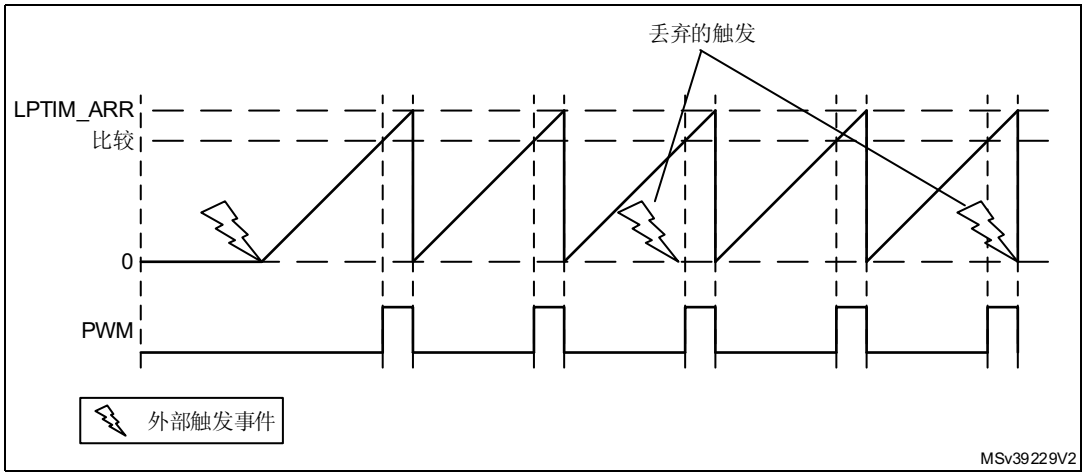
连续模式

要使能连续计数，必须将 CNTSTRT 位置 1。

若选择外部触发，则在 CNTSTRT 置 1 后到达的外部触发事件将启动计数器进行连续计数。任何后续的外部触发事件都将被丢弃，如*图 351* 所示。

若通过软件启动（TRIGEN[1:0] =“00”），将 CNTSTRT 置 1 会使计数器开始连续计数。

图 351. LPTIM 输出波形、连续计数模式配置



SNGSTRT 和 CNTSTRT 位只能在定时器使能时（ENABLE 位置 1）置 1。可以“实时”从单触发模式切换为连续模式。

如果之前选择的是连续模式，则将 SNGSTRT 置 1 会使 LPTIM 切换为单触发模式。计数器（激活时）在生成 LPTIM 更新事件后立即停止。

如果之前选择的是单触发模式，则将 CNTSTRT 置 1 会使 LPTIM 切换为连续模式。计数器（激活时）在达到 ARR 后立即重新启动。

29.4.9 超时功能

若在一个选定的触发输入上检测到有效边沿，则可用于复位 LPTIM 计数器。该功能通过 TIMOUT 位进行控制。

第一个触发事件会启动定时器，任何后续的触发事件都将复位 LPTIM 计数器和重复计数器，并且定时器将重新启动。

可实现低功耗超时功能。超时值对应于比较值；如果在预期的时间帧内未发生触发事件，MCU 将由比较匹配事件唤醒。

29.4.10 生成波形

两个 16 位寄存器、LPTIM_ARR（自动重载寄存器）和 LPTIM_CCRx（捕获/比较寄存器）用于在 LPTIM 输出上生成多个不同的波形

定时器可生成以下波形：

- PWM 模式：只要 LPTIM_CNT 中的计数器值超过 LPTIM_CCRx 中的比较值，LPTIM 输出就会置位。只要 LPTIM_ARR 和 LPTIM_CNT 寄存器之间发生匹配，LPTIM 输出就会复位。有关更多详细信息，请参见*第 29.4.19 节：PWM 模式*。
- 单脉冲模式：对于第一个脉冲，输出波形与 PWM 模式输出波形类似，随后输出将永久复位。
- 置 1 一次模式：除输出保持最后一个信号电平外（取决于配置的输出极性），输出波形与单脉冲模式输出波形类似。

上述模式要求 LPTIM_ARR 寄存器的值严格大于 LPTIM_CCRx 寄存器的值。

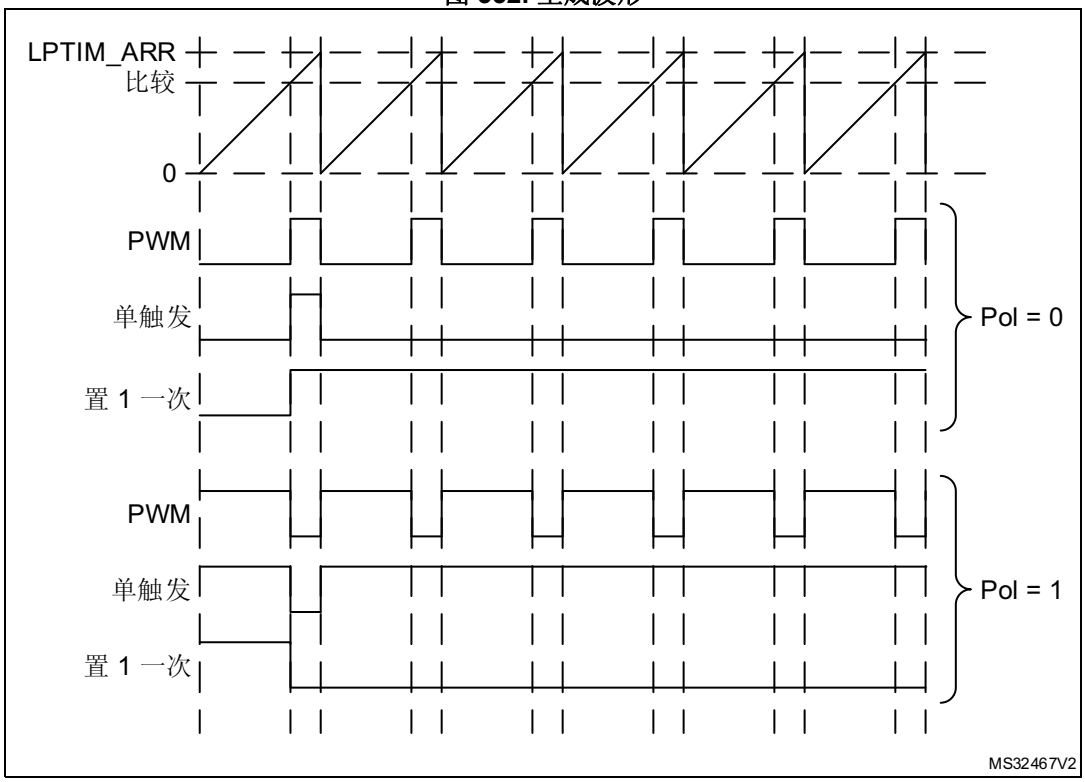
LPTIM 输出波形可通过 WAVE 位配置，具体如下：

- 若将 WAVE 位复位为 0，则会强制 LPTIM 生成 PWM 波形或单脉冲波形，具体取决于将哪个位（CNTSTRT 或 SNGSTRT）置 1。
- 若将 WAVE 位置 1，则会强制 LPTIM 生成置 1 一次模式波形。

CCxP 位控制 LPTIM 输出极性。更改立即生效，因此输出默认值将在极性重新配置后立即更改，甚至会在定时器使能前进行更改。

生成的信号的频率高达 LPTIM 时钟频率 2 分频。*图 352* 给出了可能在 LPTIM 输出上生成的三种波形。此外，该图还显示了通过 CCxP 位更改极性所产生的效果。

图 352. 生成波形



29.4.11 寄存器更新

LPTIM_ARR 寄存器、LPTIM_RCR 寄存器和 LPTIM_CCRx 寄存器在 APB 总线写操作后会立即更新，若定时器已启动，也可以在下一个 LPTIM 更新事件生成时同步更新。

PRELOAD 位控制 LPTIM_ARR 寄存器、LPTIM_RCR 寄存器和 LPTIM_CCRx 寄存器的更新方式：

- 当 PRELOAD 位复位为 0 时，LPTIM_ARR 寄存器、LPTIM_RCR 寄存器和 LPTIM_CCRx 寄存器会在写访问后立即更新。
- 当 PRELOAD 位置 1 时，如果定时器已启动，则 LPTIM_ARR 寄存器、LPTIM_RCR 寄存器和 LPTIM_CCRx 寄存器将在下一个 LPTIM 更新事件生成时更新。

LPTIM APB 接口和 LPTIM 内核逻辑使用的时钟不同，因此在 APB 写操作后，需要经过一定的延迟，写入值才能用于计数器比较器。在此延迟期间，必须避免向这些寄存器执行其他写操作。

LPTIM_ISR 寄存器中的 ARROK 标志、REPOK 标志和 CMPxOK 标志分别指示 LPTIM_ARR 寄存器、LPTIM_RCR 寄存器和 LPTIM_CCRx 寄存器的写操作已完成。

向 LPTIM_ARR 寄存器、LPTIM_RCR 寄存器或 LPTIM_CCRx 寄存器执行写操作后，只有在前一次写操作完成后，才能对同一寄存器执行新的写操作。在 ARROK 标志、REPOK 标志或 CMPxOK 标志置 1 前执行连续的写操作会造成无法预知的结果。

29.4.12 计数器模式

LPTIM 计数器可用于对 LPTIM Input1 上的外部事件进行计数，也可用于对内部时钟周期进行计数。CKSEL 位和 COUNTMODE 位用于控制使用哪些源更新计数器。

若使用 LPTIM 对 Input1 上的外部事件进行计数，计数器可在上升沿、下降沿或两种边沿进行更新，具体取决于写入 CKPOL[1:0] 位的值。

根据 CKSEL 和 COUNTMODE 值，可选择以下计数模式：

- CKSEL = 0：LPTIM 由内部时钟源提供时钟
  - COUNTMODE = 0

    LPTIM 配置为由内部时钟源提供时钟，LPTIM 计数器配置为在每个内部时钟脉冲后进行更新。
  - COUNTMODE = 1

    LPTIM 外部 Input1 通过提供给 LPTIM 的内部时钟进行采样。

    因此，为了不丢失任何事件，外部 Input1 信号变化的频率绝不能超过提供给 LPTIM 的内部时钟的频率。此外，不得对提供给 LPTIM 的内部时钟进行预分频 (PRESC[2:0] = 000)。
- CKSEL = 1：LPTIM 由外部时钟源提供时钟

  COUNTMODE 值不相关。

  在这种配置下，LPTIM 无需内部时钟源（已使能干扰滤波器时除外）。注入到 LPTIM 外部 Input1 的信号用作 LPTIM 的系统时钟。该配置适合未使能任何内置振荡器的工作模式。

  对于这种配置，LPTIM 计数器可以在 Input1 时钟信号的上升沿或下降沿进行更新，但不可在上升沿和下降沿均更新。

  由于注入到 LPTIM 外部 Input1 的信号也可用于为 LPTIM 内核逻辑提供时钟，计数器递增计数前存在一些初始延时（使能 LPTIM 后）。更确切地说，LPTIM 外部 Input1 的前五个有效边沿将丢失（使能 LPTIM 后）。

29.4.13 定时器使能

LPTIM_CR 寄存器中的 ENABLE 位用于使能/禁止 LPTIM 内核逻辑。将 ENABLE 位置 1 后，需要延迟两个计数器时钟周期，才能真正使能 LPTIM。

LPTIM_CFGR 寄存器必须在禁止 LPTIM 后才能修改。

29.4.14 定时器计数器复位

为了将 LPTIM_CNT 寄存器的内容复位为零，实现了两种复位机制：

- 同步复位机制：同步复位由 LPTIM_CR 寄存器中的 COUNTRST 位控制。在将 COUNTRST 位域置 1 后，复位信号在 LPTIM 内核时钟域中传播。因此，重要的是要注意，要在经历几个 LPTIM 内核逻辑时钟脉冲之后再考虑复位。这会使 LPTIM 计数器在复位触发和生效之间额外计数几个脉冲。由于 COUNTRST 位位于 APB 时钟域中，并且 LPTIM 计数器位于 LPTIM 内核时钟域中，因此当将 1 写入到 COUNTRST 位时，内核时钟需要 3 个时钟周期的延迟用以同步由 APB 时钟域发出的复位信号。
- 异步复位机制：异步复位由位于 LPTIM_CR 寄存器中的 RSTARE 位控制。当该位置 1 时，对 LPTIM_CNT 寄存器的任何读访问都会将其内容复位为零。必须在不提供 LPTIM 内核时钟的时间范围内触发异步复位。例如，当 LPTIM Input1 用作外部时钟源时，必须在充分保证 LPTIM Input1 不会发生翻转的情况下，才能应用异步复位。

请注意，为了实现可靠的 LPTIM_CNT 寄存器内容读取，必须执行两次连续的读访问并进行比较。当两次读访问的值相等时，可认为读访问可靠。然而，当使能了异步复位时，不可能两次读取 LPTIM_CNT 寄存器。

警告： **LPTIM 内没有防止两个复位机制同时使用的机制。所以开发人员必须确保这两个机制是互斥地使用。**

29.4.15 编码器模式

此模式用于处理来自正交编码器的信号，此正交编码器用于检测旋转元件的角度位置。编码器接口模式就相当于带有方向选择的外部时钟。这意味着，计数器仅在 0 到 LPTIM_ARR 寄存器中编程的自动重载值之间进行连续计数（根据具体方向，从 0 递增计数到 ARR，或从 ARR 递减计数到 0）。因此必须在启动计数器之前配置 LPTIM_ARR。通过两个外部输入信号（Input1 和 Input2）生成时钟信号作为 LPTIM 计数器的时钟。这两个信号间的相位确定计数方向。

仅当 LPTIM 由内部时钟源提供时钟时才可使用编码器模式。Input1 和 Input2 输入上的信号频率不得超过 LPTIM 内部时钟频率 4 分频。必须满足此条件才能确保 LPTIM 正常工作。

方向变化由 LPTIM_ISR 寄存器中的两个递减和递增标志指示。此外，如果通过 DOWNIE 位使能，还可为两种方向变化事件产生中断。

要激活编码器模式，必须将 ENC 位置 1。LPTIM 必须首先配置为连续模式。

当编码器模式激活时，LPTIM 计数器按照增量编码器的速度和方向自动修改。因此，其内容始终代表编码器的位置。计数方向由递增和递减标志指示，对应于编码器转子的旋转方向。

根据使用 CKPOL[1:0] 位配置的边沿敏感性，可得几种不同的计数方案。下表汇总了可能的组合（假设 Input1 和 Input2 不同时切换）。

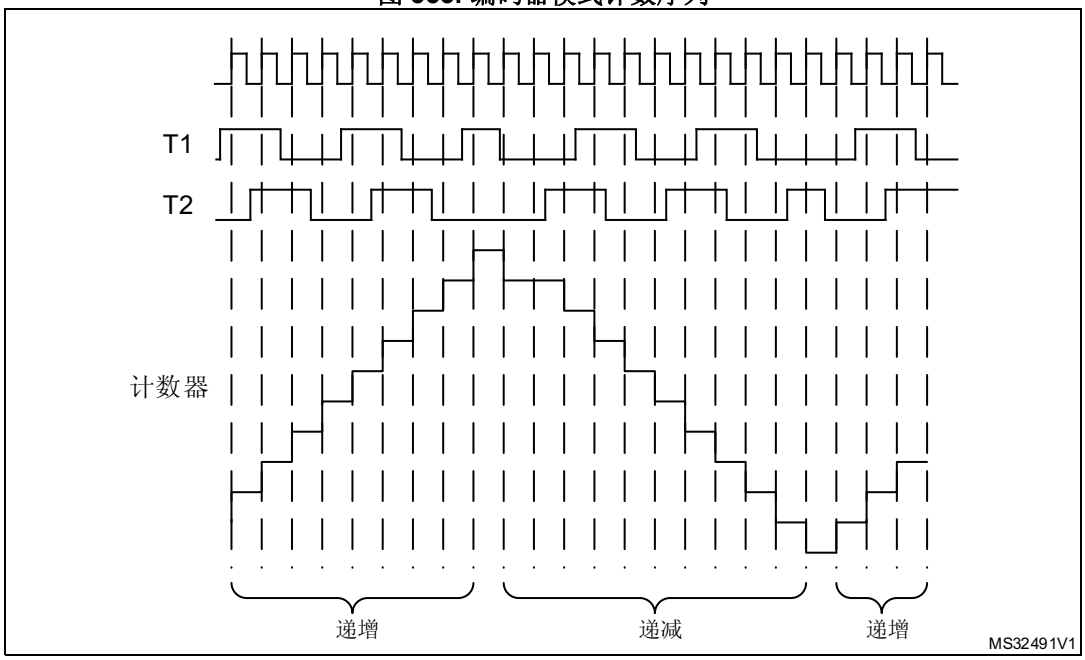
表 217. 编码器计数方案

有效边沿	相反信号的电平（Input1 对应 Input2，Input2 对应 Input1）	Input1 信号		Input2 信号	
		上升	下降	上升	下降
上升沿	高	递减	不计数	递增	不计数
	低	递增	不计数	递减	不计数
下降沿	高	不计数	递增	不计数	递减
	低	不计数	递减	不计数	递增
两种边沿	高	递减	递增	递增	递减
	低	递增	递减	递减	递增

下图所示为编码器模式下配置了两种边沿敏感性的计数序列。

小心： 在此模式下，LPTIM 必须由内部时钟源提供时钟，因此 CKSEL 位必须保持其复位值 0。另外，预分频器分频比必须等于其复位值 1（PRESC[2:0] 位必须为“000”）。

图 353. 编码器模式计数序列



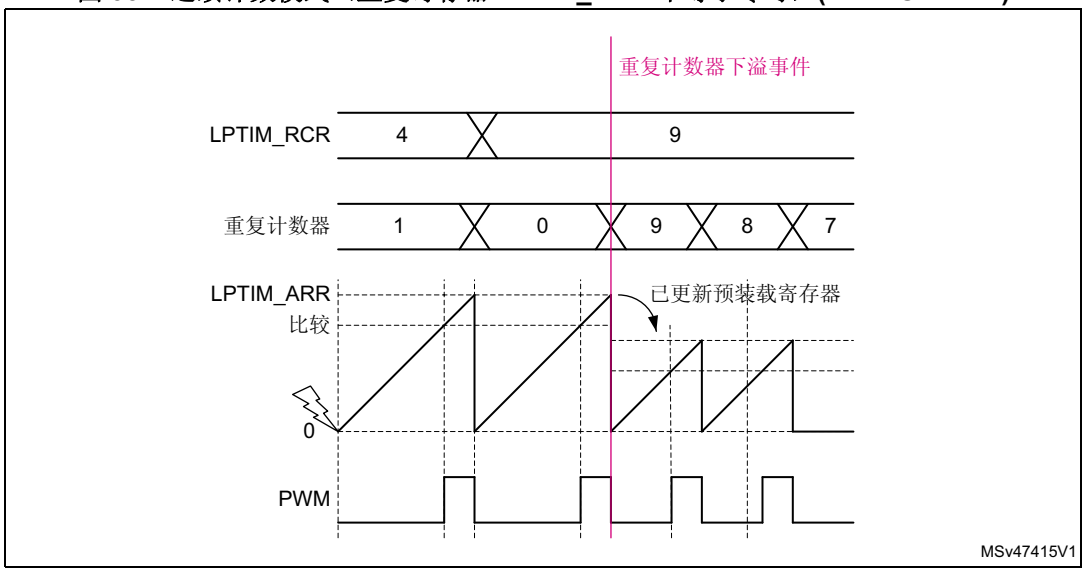
29.4.16 重复计数器

LPTIM 具有一个重复计数器，该计数器在每次发生 LPTIM 计数器上溢事件时递减 1。当重复计数器为零且 LPTIM 计数器上溢时，将生成重复计数器下溢事件。在每个重复计数器下溢事件后，重复计数器都会装载重复寄存器 LPTIM_RCR 的 REP[7:0] 位域的内容。

当 REP[7:0] 寄存器设为 0 时，每次 LPTIM 计数器上溢都会生成重复下溢事件。

当 PRELOAD = 1 时，写入 REP[7:0] 位域对重复计数器的内容没有任何影响，直到发生下一个重复下溢事件为止。重复计数器在每次发生 LPTIM 计数器上溢事件时继续递减计数，而只有在生成重复下溢事件时，重复计数器才会装载写入 REP[7:0] 的新值。*图 354* 说明了这种行为。

图 354. 连续计数模式（重复寄存器 LPTIM_RCR 不等于零时）(PRELOAD = 1)



重复计数器下溢事件与 LPTIM 预装载寄存器更新之间存在系统性关联（有关更多详细信息，请参见“寄存器更新”部分）。

重复计数器下溢事件通过映射到 LPTIM_ISR 寄存器的更新事件 (UE) 标志向软件发送信号。当 UE 标志置 1 时，如果映射到 LPTIM_DIER 寄存器的相应更新事件中断使能 (UEIE) 控制位置 1，则可触发 LPTIM 中断。

重复寄存器 LPTIM_RCR 位于 APB 总线接口时钟域中，而重复计数器自身则位于 LPTIM 内核时钟域中。每次向 LPTIM_RCR 寄存器中写入新值时，都会将该值从 APB 总线接口时钟域传播到 LPTIM 内核时钟域，以便在重复计数器下溢事件发生后立即将新写入的值装载到重复计数器。同步新写入内容的延迟为 4 个 APB 时钟周期加上 3 个 LPTIM 内核时钟周期，延迟结束后由 LPTIM_ISR 寄存器中的 REPOK 标志发出信号。如果 LPTIM 内核时钟周期相对较长（例如当 LPTIM 内核由 LSI 时钟源提供时钟时），则软件持续轮询 REPOK 标志以检测 LPTIM_RCR 寄存器的内容是否完成同步所需要的时间可能会很长。因此，如果 LPTIM_DIER 寄存器中与 REPOK 标志相关的 REPOKIE 控制位置 1，则 REPOK 标志置 1 时可生成中断。

注意： *向 LPTIM_RCR 寄存器执行写操作后，只有在前一次写操作完成后，才能对该寄存器执行新的写操作。在 REPOK 标志置 1 前执行连续的写操作会造成无法预知的结果。*

小心： 在 PRELOAD = 0 的情况下使用重复计数器时，必须在自动重载匹配事件前至少五个计数器周期更改 LPTIM_RCR 寄存器，否则可能会发生无法预知的行为。

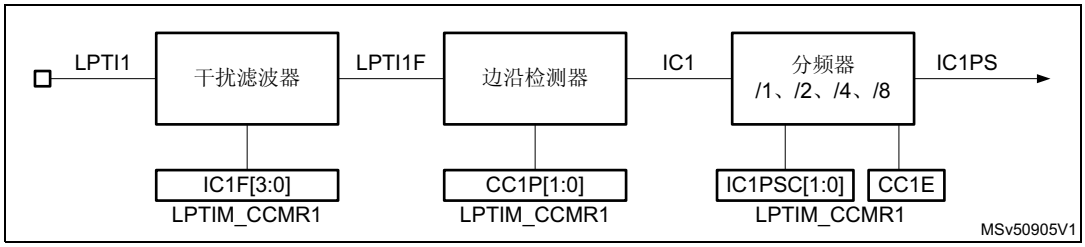
29.4.17 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器、一个捕获输入阶段（数字滤波、多路复用和预分频器）和一个 PWM 输出阶段（比较器和输出控制）构建而成。

输入阶段

输入阶段对相应的 LPTIx 输入进行采样，生成一个滤波后的信号 LPTIxF。然后，带有极性选择功能的边沿检测器生成可用作捕获命令的 ICx 信号。该信号经过预分频后，生成捕获命令信号 (ICxPS)。

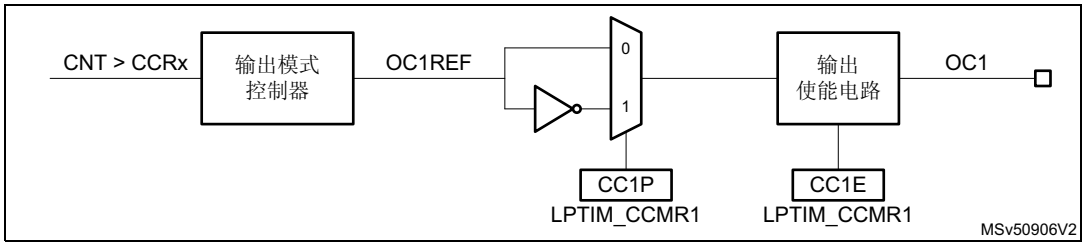
图 355. 捕获/比较输入阶段（通道 1）



输出阶段

输出阶段生成一个中间波形作为基准：OCxREF（高电平有效）。链的末端决定最终输出信号的极性。

图 356. 捕获/比较输出阶段（通道 1）



29.4.18 输入捕获模式

在输入捕获模式下，当相应的 ICx 信号检测到跳变沿后，将使用捕获/比较寄存器 (LPTIM_CCRx) 来锁存计数器的值。假设通道 x 上使能了输入捕获（CCxE 置 1），则当发生捕获事件时，会将相应的 CCxIF 标志（LPTIM_ISR 寄存器）置 1，并可发送中断或 DMA 请求（如果已使能）。如果发生捕获事件时 CCxIF 标志已处于高位，则会将过捕获标志 CCxOF（LPTIM_ISR 寄存器）置 1。可通过软件向 CCxIF 写入 1 使其清零，或读取存储在 LPTIM_CCRx 寄存器中的已捕获数据。通过向 CCxOCF 写入 1 来清零 CCxOF。

注意： *在 DMA 模式下，必须先使能 IC DMA 请求再启动计数器，最后才能使能输入捕获通道（将 CCxE 位置 1）。这是为了防止在计数器尚未启动时生成输入捕获 DMA 请求。*

输入捕获干扰滤波器延迟

当触发事件到达通道 x 输入 (LPTIx) 时，根据配置的干扰滤波器（CCMRx 寄存器中的 ICxF[1:0] 位域）和内核时钟预分频器值（CFGR 寄存器中的 PRESC[2:0] 位域），可能存在可变延迟，该延迟会导致 CCRx 寄存器中存储的捕获值与捕获触发所对应的实际值之间存在系统偏移（请参见*表 218*）。

该偏移对脉宽测量没有影响，因为它是系统偏移，在两次捕获之间会得到补偿。

可使用以下公式计算输入捕获触发对应的实际捕获值：

实际捕获值 = 已捕获(LPTIM_CCRx) - 偏移

必须根据干扰滤波器和内核时钟预分频器值（CFGR 寄存器中的 PRESC 位域）使用相关偏移

示例：确定 PRESC[2:0] = 0x2 且 ICxF = 0x3 时的实际捕获值。

对于该配置（PRESC[2:0] = 0x2 且 ICxF = 0x3），根据*表 218*，偏移为 5。

假设 CCRx 中的捕获值为 9 (LPTIM_CNT = 9)，这意味着当 LPTIM_CNT 等于 9 - 5 = 4 时发生捕获触发。

表 218. 输入捕获干扰滤波器延迟（在计步器单元中）

预分频器 PRESC[2:0]	ICxF[1:0]	偏移
0	0	2
	1	7
	2	9
	3	13
1	0	3
	1	5
	2	6
	3	8
2	0	2
	1	3
	2	4
	3	5
3	0	2
	1	2
	2	3
	3	3
4	0	2
	1	2
	2	2
	3	2
5	0	2
	1	2
	2	2
	3	2
6	0	2
	1	2
	2	2
	3	2
7	0	2
	1	2
	2	2
	3	2

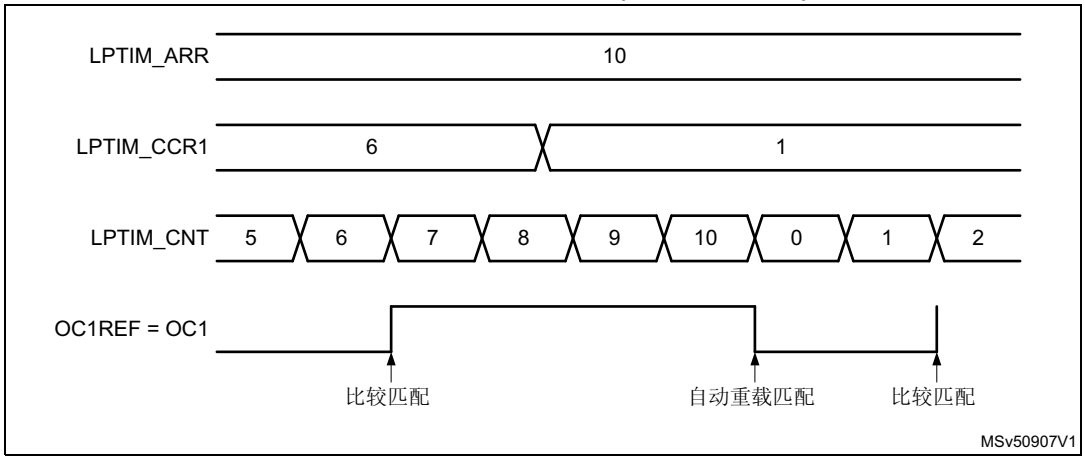
29.4.19 PWM 模式

PWM 模式可以生成一个信号，该信号频率由 LPTIM_ARR 寄存器值决定，其占空比则由 LPTIM_CCRx 寄存器值决定。LPTIM 可在边沿对齐模式下生成 PWM。

OCx 极性可通过软件来编程（使用 LPTIM_CCMRx 寄存器的 CCxP 位）。可将其编程为高电平有效或低电平有效。OCx 输出通过将 LPTIM_CCMRx 寄存器中的 CCxE 位置 1 来使能。有关详细信息，请参见 LPTIM_CCMRx 寄存器说明。

图 357 给出了一个示例，其中 LPTIM 通道 1 配置为 PWM 模式，LPTIM_CCR1 = 6（稍后变为 1），LPTIM_ARR = 10。

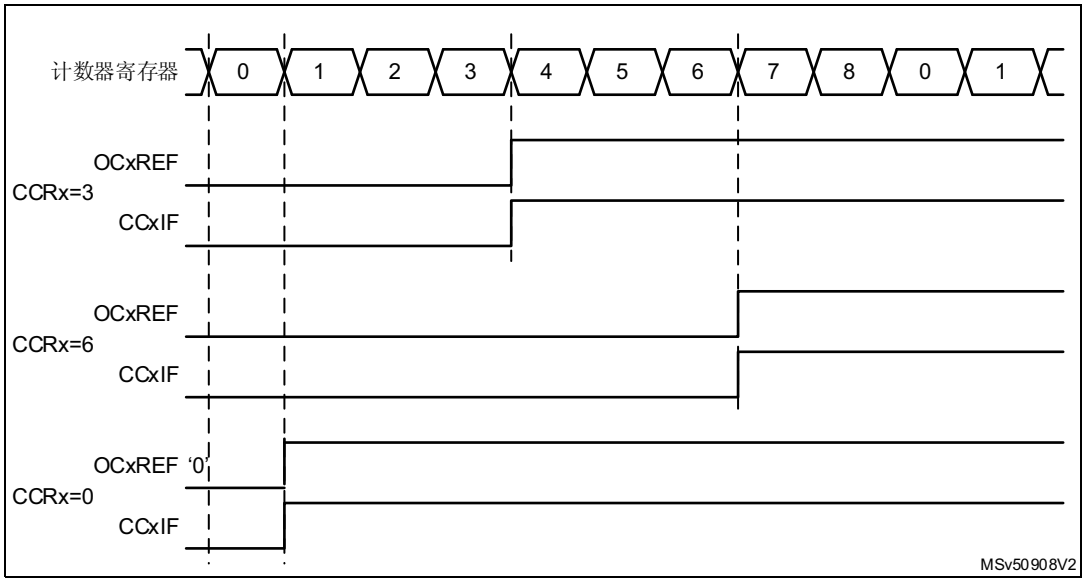
图 357. 边沿对齐 PWM 模式 (PRELOAD = 1)



在以下示例中，只要 LPTIM_CNT ≤ LPTIM_CCRx，参考 PWM 信号 OCxREF 即为低电平，否则为高电平。

图 358 举例介绍边沿对齐模式的一些 PWM 波形 (LPTIM_ARR = 8)。

图 358. 边沿对齐 PWM 波形（ARR = 8 且 CCxP = 0）



PWM 模式下的立即更新 PRELOAD = 0

PWM (PRELOAD = 0) 模式允许在当前 PWM 周期内提前更改输出电平。基于立即更新 LPTIM_CCRx 寄存器 (PRELOAD = 0) 以及连续比较 LPTIM_CNT 寄存器与 LPTIM_CCRx 寄存器，该模式允许在当前 PWM 周期内尽快应用新的占空比值，而无需等待当前 PWM 周期完成。

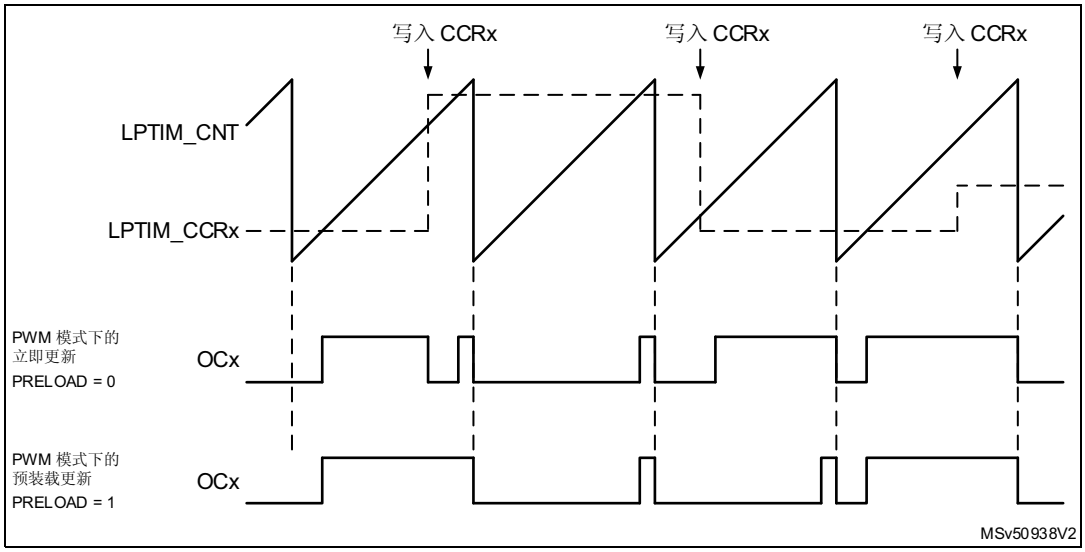
当 PRELOAD = 0 时，可以由软件（或 DMA）通过更新 LPTIM_CCRx 寄存器中的比较值来实时更改 OCxREF 信号的电平。

根据写入的比较值以及当前的计数器值和比较值，将对 OCxREF 电平进行重新分配，具体说明如下：

- 如果新比较值未超过当前计数器值而当前比较值超过了计数器值，则写入新的比较值后，会立即将 OCxREF 电平重新分配为高电平。
- 如果新比较值超过了计数器值而当前比较值未超过计数器值，则写入新的比较值后，会立即将 OCxREF 电平重新分配为低电平。

如果新比较值和当前比较值均未超过计数器值，则输出参考信号 OCxREF 的电平保持不变。*图 359* 说明了当 PRELOAD = 0 和 PRELOAD = 1 时 OCxREF 信号电平的行为。

图 359. PWM 模式下的立即更新与预装载更新



注意： *对于这两种 PWM 模式，比较匹配标志、自动重载匹配标志和更新事件标志会在相应事件发生后的一个 LPTIM 计数器周期后置 1，OCxREF 电平也会在相应事件后的一个 LPTIM 计数器周期后发生更改。例如，如果 LPTIM_CCRx 设为 3，CCxIF 将在 LPTIM_CNT = 4 时置 1。图 357 说明了这种行为。*

29.4.20 DMA 请求

LPTIM 能够生成两类 DMA 请求：

- 用于检索输入捕获计数器值的 DMA 请求
- 用于定期对 LPTIMER 的一部分多次重新编程的 DMA 更新请求（无需软件开销）。

输入捕获 DMA 请求

每个 LPTIM 通道都有专用的输入捕获 DMA 请求。每次 CCRx 寄存器中的捕获就绪时，都会生成 DMA 请求（LPTIM_DIER 中的 CCxDE 位置 1 时），CCxIF 也会置 1。之后，CCRx 中的捕获值可以通过 DMA 定期传输到所需的存储器目标位置。读取 CCRx 寄存器中的捕获值时，CCxIF 由硬件自动清零。

注意： *ICx DMA 请求信号 lptim_icx_dma 会在下列情况下复位：*

- 如果相应的 DMA 请求被禁止（将 LPTIM_DIER 寄存器中的 CCxDE 位清零）

- 如果通道 x 被禁止（将 CCxE 位清零）

- 如果 LPTIM 被禁止（将 LPTIM_CR 寄存器中的 ENABLE 位清零）

更新事件 DMA 请求

每次发生更新事件时，都会生成 DMA 请求（LPTIM_DIER 中的 UEDE 置 1 时），UE 标志也会置 1。DMA 请求可用于定期更新 LPTIM_ARR 寄存器、LPTIM_RCR 寄存器或 LPTIM_CCRx 寄存器，从而允许生成自定义 PWM 波形。

任何总线主设备（例如 CPU 或 DMA）对 LPTIM_ARR 寄存器进行写访问后，UE 都会立即由硬件自动清零。

注意： *UE DMA 请求信号 lptim_ue_dma 会在下列情况下复位：*

- 如果相应的 DMA 请求被禁止（将 LPTIM_DIER 寄存器中的 UEDE 位清零）

- 如果 LPTIM 被禁止（将 LPTIM_CR 寄存器中的 ENABLE 位清零）

29.4.21 调试模式

当微控制器进入调试模式时（内核停止），LPTIM 计数器会根据 DBG 模块中的 DBG_LPTIM_STOP 配置位选择继续正常工作或者停止工作。

29.5 LPTIM 低功耗模式

表 219. 低功耗模式对 LPTIM 的影响

模式	说明
睡眠	无影响。LPTIM 中断可使器件退出睡眠模式。
停止	如果 LPTIM 由停止模式下可用的振荡器提供时钟，则 LPTIM 可以正常工作，中断会使器件退出停止模式。
待机	LPTIM 外设掉电，退出待机模式后必须重新初始化。

注意： *在进入睡眠、停止和待机模式前，必须禁止所有 DMA 请求（复位 UEDE 和 CCxDE 位）。*

29.6 LPTIM 中断

若以下事件通过 LPTIM_DIER 寄存器使能，则这些事件会生成中断/唤醒事件：

- 比较匹配
- 自动重载匹配（编码器模式下无论哪种方向）
- 外部触发事件
- 自动重载寄存器写操作完成
- 比较寄存器写操作完成
- 方向变化（编码器模式），可编程（递增/递减/同时递增和递减）。
- 更新事件
- 重复寄存器更新成功
- 发生输入捕获事件
- 发生过捕获事件
- 中断使能寄存器更新成功

注意： *只要 LPTIM_DIER 寄存器中的位在 LPTIM_ISR 寄存器（状态寄存器）中相应标志置 1 后置 1，就不会触发中断。*

表 220. 中断事件

中断向量	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止模式(1)
LPTIMx	比较匹配	CCxIF	CCxIE	向 CCxCF 写入 1	是	是
	输入捕获	CCxIF	CCxIE	向 CCxCF 写入 1	是	是
	过捕获	CCxOF	CCxOIE	向 CCxOCF 写入 1	是	是
	自动重载匹配	ARRM	ARRMIE	向 ARRMCF 写入 1	是	是
	外部触发事件	EXTTRIG	EXTTRIGIE	向 EXTTRIGCF 写入 1	是	是
	自动重载寄存器更新成功	ARROK	ARROKIE	向 ARROKCF 写入 1	是	是
	捕获/比较寄存器更新成功	CMPxOK	CMPxOKIE	向 CMPxOKCF 写入 1	是	是
	方向更改为递增(2)	UP	UPIE	向 UPCF 写入 1	是	是
	方向更改为递减(2)	DOWN	DOWNIE	向 DOWNCF 写入 1	是	是
	更新事件	UE	UEIE	向 UECF 写入 1	是	是
	重复寄存器更新成功	REPOK	REPOKIE	向 REPOKCF 写入 1	是	是

1. 仅当 LPTIM 实例支持从停止模式唤醒功能时，各 LPTIM 事件才能将器件从停止模式唤醒。请参见*第 29.3 节：LPTIM 实现*。
2. 如果 LPTIM 不支持编码器模式功能，则该事件不存在。请参见*第 29.3 节：LPTIM 实现*。

29.7 LPTIM 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节：寄存器相关缩写词列表*。

外设寄存器只能按字（32 位）进行访问。

29.7.1 LPTIMx 中断和状态寄存器 [复用] (LPTIMx_ISR)（x = 1、2）

LPTIMx interrupt and status register [alternate]

该寄存器说明只适用于输出比较模式。关于输入捕获模式，请参见下一节。

偏移地址：0x000

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIEROK	Res.	Res.	Res.	Res.	CMP2OK	Res.	Res.	Res.
							r					r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CC2IF	REPOK	UE	DOWN	UP	ARROK	CMP1OK	EXTTRIG	ARRM	CC1IF
						r	r	r	r	r	r	r	r	r	r

位 31:25 保留，必须保持复位值。

位 24 **DIEROK**：中断使能寄存器更新成功 (Interrupt enable register update OK)

由硬件将 DIEROK 置 1 时，会通知应用 LPTIM_DIER 寄存器的 APB 总线写操作已成功完成。可以通过向 LPTIM_ICR 寄存器中的 DIEROKCF 位写入 1 来将 DIEROK 标志清零。

位 23:22 保留，必须保持复位值。

位 21 保留，必须保持复位值。

位 20 保留，必须保持复位值。

位 19 **CMP2OK**：比较寄存器 2 更新成功 (Compare register 2 update OK)

由硬件将 CMP2OK 置 1 时，会通知应用 LPTIM_CCR2 寄存器的 APB 总线写操作已成功完成。可以通过向 LPTIM_ICR 寄存器中的 CMP2OKCF 位写入 1 来将 CMP2OK 标志清零。

注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 18:12 保留，必须保持复位值。

位 11 保留，必须保持复位值。

位 10 保留，必须保持复位值。

位 9 **CC2IF**：比较 2 中断标志 (Compare 2 interrupt flag)

如果通道 CC2 配置为输出：

由硬件将 CC2IF 标志置 1 时，会通知应用 LPTIM_CNT 寄存器的值与比较寄存器的值匹配。可以通过向 LPTIM_ICR 寄存器中的 CC2CF 位写入 1 来将 CC2IF 标志清零。

0：不匹配

1：计数器 LPTIM_CNT 寄存器的值与 LPTIM_CCR2 寄存器的值匹配

注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 8 **REPOK**：重复寄存器更新成功 (Repetition register update OK)

由硬件将 REPOK 置 1 时，会通知应用 LPTIM_RCR 寄存器的 APB 总线写操作已成功完成。可以通过向 LPTIM_ICR 寄存器中的 REPOKCF 位写入 1 来将 REPOK 标志清零。

位 7 **UE**：发生 LPTIM 更新事件 (LPTIM update event occurred)

由硬件将 UE 置 1 时，会通知应用生成了更新事件。如果使能了相应的中断或 DMA 请求，则会生成该中断或 DMA 请求。可以通过向 LPTIM_ICR 寄存器中的 UECF 位写入 1 来将 UE 标志清零。只要总线主设备（例如 CPU 或 DMA）对 LPTIM_ARR 寄存器进行写访问，UE 标志就会由硬件自动清零。

位 6 **DOWN**：计数方向从递增变为递减 (Counter direction change up to down)

在编码器模式下，由硬件将 DOWN 位置 1 时，会通知应用计数方向由递增变为递减。可以通过向 LPTIM_ICR 寄存器中的 DOWNCF 位写入 1 来将 DOWN 标志清零。

注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 5 **UP**：计数方向从递减变为递增 (Counter direction change down to up)

在编码器模式下，由硬件将 UP 位置 1 时，会通知应用计数方向由递减变为递增。可以通过向 LPTIM_ICR 寄存器中的 UPCF 位写入 1 来将 UP 标志清零。

注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 4 **ARROK**：自动重载寄存器更新成功 (Autoreload register update OK)

由硬件将 ARROK 置 1 时，会通知应用 LPTIM_ARR 寄存器的 APB 总线写操作已成功完成。可以通过向 LPTIM_ICR 寄存器中的 ARROKCF 位写入 1 将 ARROK 标志清零。

位 3 **CMP1OK**：比较寄存器 1 更新成功 (Compare register 1 update OK)

由硬件将 CMP1OK 置 1 时，会通知应用 LPTIM_CCR1 寄存器的 APB 总线写操作已成功完成。可以通过向 LPTIM_ICR 寄存器中的 CMP1OKCF 位写入 1 来将 CMP1OK 标志清零。

位 2 **EXTTRIG**：外部触发边沿事件 (External trigger edge event)

由硬件将 EXTTRIG 置 1 时，会通知应用所选的外部触发输入上产生有效边沿。如果由于定时器已启动而忽略触发事件，则不会将此标志置 1。可以通过向 LPTIM_ICR 寄存器中的 EXTTRIGCF 位写入 1 来将 EXTTRIG 标志清零。

位 1 **ARRM**：自动重载匹配 (Autoreload match)

由硬件将 ARRM 置 1 时，会通知应用 LPTIM_CNT 寄存器的值已达到 LPTIM_ARR 寄存器的值。可以通过向 LPTIM_ICR 寄存器中的 ARRMCF 位写入 1 来将 ARRM 标志清零。

位 0 **CC1IF**：比较 1 中断标志 (Compare 1 interrupt flag)

如果通道 CC1 配置为输出：

由硬件将 CC1IF 标志置 1 时，会通知应用 LPTIM_CNT 寄存器的值与比较寄存器的值匹配。可以通过向 LPTIM_ICR 寄存器中的 CC1CF 位写入 1 来将 CC1IF 标志清零。

0：不匹配

1：计数器 LPTIM_CNT 寄存器的值与 LPTIM_CCR1 寄存器的值匹配

29.7.2 LPTIMx 中断和状态寄存器 [复用] (LPTIMx_ISR)（x = 1、2）

LPTIMx interrupt and status register [alternate]

该寄存器说明只适用于输入捕获模式。关于输出比较模式，请参见上一节。

偏移地址：0x000

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIEROK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
							r								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	CC2OF	CC1OF	Res.	Res.	CC2IF	REPOK	UE	DOWN	UP	ARROK	Res.	EXTTRIG	ARRM	CC1IF
		r	r			r	r	r	r	r	r		r	r	r

位 31:25 保留，必须保持复位值。

位 24 **DIEROK**：中断使能寄存器更新成功 (Interrupt enable register update OK)
由硬件将 DIEROK 置 1 时，会通知应用 LPTIM_DIER 寄存器的 APB 总线写操作已成功完成。可以通过向 LPTIM_ICR 寄存器中的 DIEROKCF 位写入 1 来将 DIEROK 标志清零。

位 23:16 保留，必须保持复位值。

位 15 保留，必须保持复位值。

位 14 保留，必须保持复位值。

位 13 **CC2OF**：捕获 2 过捕获标志 (Capture 2 over-capture flag)
仅当将相应通道配置为输入捕获模式时，此标志位才会由硬件置 1。通过软件向 LPTIM_ICR 寄存器中的 CC2OCF 位写入 1 可将该位清零。
0：未检测到过捕获。
1：LPTIM_CCR2 寄存器中已捕获到计数器值且 CC2IF 标志已置 1。
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 12 **CC1OF**：捕获 1 过捕获标志 (Capture 1 over-capture flag)
仅当将相应通道配置为输入捕获模式时，此标志位才会由硬件置 1。通过软件向 LPTIM_ICR 寄存器中的 CC1OCF 位写入 1 可将该位清零。
0：未检测到过捕获。
1：LPTIM_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。
注意：如果 LPTIM 没有实现至少 1 个通道，则保留该位。请参见第 29.3 节。

位 11 保留，必须保持复位值。

位 10 保留，必须保持复位值。

位 9 **CC2IF**：捕获 2 中断标志 (Capture 2 interrupt flag)
如果通道 CC2 配置为输入：
由硬件将 CC2IF 置 1 时，会通知应用 LPTIM_CCR2 寄存器中已捕获到计数器的当前值。如果使能了相应的中断或 DMA 请求，则会生成该中断或 DMA 请求。如果 CC2IF 标志已为高电平，CC2OF 标志将置 1。
0：未发生输入捕获事件
1：LPTIM_CCR2 寄存器中已捕获到计数器值。（IC2 上已检测到与所选极性匹配的边沿）。CPU 或 DMA 读取捕获值后，CC2IF 标志就会由硬件自动清零。可以通过向 LPTIM_ICR 寄存器中的 CC2CF 位写入 1 来将 CC2IF 标志清零。
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 8 **REPOK**：重复寄存器更新成功 (Repetition register update OK)

由硬件将 REPOK 置 1 时，会通知应用 LPTIM_RCR 寄存器的 APB 总线写操作已成功完成。可以通过向 LPTIM_ICR 寄存器中的 REPOKCF 位写入 1 来将 REPOK 标志清零。

位 7 **UE**：发生 LPTIM 更新事件 (LPTIM update event occurred)

由硬件将 UE 置 1 时，会通知应用生成了更新事件。可以通过向 LPTIM_ICR 寄存器中的 UECF 位写入 1 来将 UE 标志清零。

位 6 **DOWN**：计数方向从递增变为递减 (Counter direction change up to down)

在编码器模式下，由硬件将 DOWN 位置 1 时，会通知应用计数方向由递增变为递减。可以通过向 LPTIM_ICR 寄存器中的 DOWNCF 位写入 1 来将 DOWN 标志清零。

注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 5 **UP**：计数方向从递减变为递增 (Counter direction change down to up)

在编码器模式下，由硬件将 UP 位置 1 时，会通知应用计数方向由递减变为递增。可以通过向 LPTIM_ICR 寄存器中的 UPCF 位写入 1 来将 UP 标志清零。

注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 4 **ARROK**：自动重载寄存器更新成功 (Autoreload register update OK)

由硬件将 ARROK 置 1 时，会通知应用 LPTIM_ARR 寄存器的 APB 总线写操作已成功完成。可以通过向 LPTIM_ICR 寄存器中的 ARROKCF 位写入 1 将 ARROK 标志清零。

位 3 保留，必须保持复位值。

位 2 **EXTTRIG**：外部触发边沿事件 (External trigger edge event)

由硬件将 EXTTRIG 置 1 时，会通知应用所选的外部触发输入上产生有效边沿。如果由于定时器已启动而忽略触发事件，则不会将此标志置 1。可以通过向 LPTIM_ICR 寄存器中的 EXTTRIGCF 位写入 1 来将 EXTTRIG 标志清零。

位 1 **ARRM**：自动重载匹配 (Autoreload match)

由硬件将 ARRM 置 1 时，会通知应用 LPTIM_CNT 寄存器的值已达到 LPTIM_ARR 寄存器的值。可以通过向 LPTIM_ICR 寄存器中的 ARRMCF 位写入 1 来将 ARRM 标志清零。

位 0 **CC1IF**：捕获 1 中断标志 (capture 1 interrupt flag)

如果通道 CC1 配置为输入：

由硬件将 CC1IF 置 1 时，会通知应用 LPTIM_CCR1 寄存器中已捕获到计数器的当前值。如果使能了相应的中断或 DMA 请求，则会生成该中断或 DMA 请求。如果 CC1IF 标志已为高电平，CC1OF 标志将置 1。

0：未发生输入捕获事件

1：LPTIM_CCR1 寄存器中已捕获到计数器值。（IC1 上已检测到与所选极性匹配的边沿）。CPU 或 DMA 读取捕获值后，CC1IF 标志就会由硬件自动清零。可以通过向 LPTIM_ICR 寄存器中的 CC1CF 位写入 1 来将 CC1IF 标志清零。

29.7.3 LPTIMx 中断清零寄存器 [复用] (LPTIMx_ICR)（x = 1、2）

LPTIMx interrupt clear register [alternate]

该寄存器说明只适用于输出比较模式。关于输入捕获模式，请参见下一节。

偏移地址：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIEROKCF	Res.	Res.	Res.	Res.	CMP2OKCF	Res.	Res.	Res.
							w					w			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CC2CF	REPOKCF	UECF	DOWNCF	UPCF	ARROKCF	CMP1OKCF	EXTTRIGCF	ARRMCF	CC1CF
						w	w	w	w	w	w	w	w	w	w

位 31:25 保留，必须保持复位值。

位 24 **DIEROKCF**：中断使能寄存器更新成功清零标志 (Interrupt enable register update OK clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 DIEROK 标志将清零。

位 23:22 保留，必须保持复位值。

位 21 保留，必须保持复位值。

位 20 保留，必须保持复位值。

位 19 **CMP2OKCF**：比较寄存器 2 更新成功清零标志 (Compare register 2 update OK clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 CMP2OK 标志将清零。
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 18:12 保留，必须保持复位值。

位 11 保留，必须保持复位值。

位 10 保留，必须保持复位值。

位 9 **CC2CF**：捕获/比较 2 清零标志 (Capture/compare 2 clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 CC2IF 标志将清零。
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 8 **REPOKCF**：重复寄存器更新成功清零标志 (Repetition register update OK clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 REPOK 标志将清零。

位 7 **UECF**：更新事件清零标志 (Update event clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 UE 标志将清零。

位 6 **DOWNCF**：方向变为递减清零标志 (Direction change to down clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 DOWN 标志将清零。
注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 5 **UPCF**：方向变为递增清零标志 (Direction change to UP clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 UP 标志将清零。
注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 4 **ARROKCF**：自动重载寄存器更新成功清零标志 (Autoreload register update OK clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 ARROK 标志将清零

位 3 **CMP1OKCF**：比较寄存器 1 更新成功清零标志 (Compare register 1 update OK clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 CMP1OK 标志将清零。

位 2 **EXTTRIGCF**：外部触发有效边沿清零标志 (External trigger valid edge clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 EXTTRIG 标志将清零

位 1 **ARRMCF**：自动重载匹配清零标志 (Autoreload match clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 ARRM 标志将清零

位 0 **CC1CF**：捕获/比较 1 清零标志 (Capture/compare 1 clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 CC1IF 标志将清零。

29.7.4 LPTIMx 中断清零寄存器 [复用] (LPTIMx_ICR)（x = 1、2）

LPTIMx interrupt clear register [alternate]

该寄存器说明只适用于输入捕获模式。关于输出比较模式，请参见上一节。

偏移地址：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIEROKCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
							w								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	CC2OCF	CC1OCF	Res.	Res.	CC2CF	REPOKCF	UECF	DOWNCF	UPCF	ARROKCF	Res.	EXTTRIGCF	ARRMCF	CC1CF
		w	w			w	w	w	w	w	w		w	w	w

位 31:25 保留，必须保持复位值。

位 24 **DIEROKCF**：中断使能寄存器更新成功清零标志 (Interrupt enable register update OK clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 DIEROK 标志将清零。

位 23:16 保留，必须保持复位值。

位 15 保留，必须保持复位值。

位 14 保留，必须保持复位值。

位 13 **CC2OCF**：捕获/比较 2 过捕获清零标志 (Capture/compare 2 over-capture clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 CC2OF 标志将清零。
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 12 **CC1OCF**：捕获/比较 1 过捕获清零标志 (Capture/compare 1 over-capture clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 CC1OF 标志将清零。
注意：如果 LPTIM 没有实现至少 1 个通道，则保留该位。请参见第 29.3 节。

位 11 保留，必须保持复位值。

位 10 保留，必须保持复位值。

位 9 **CC2CF**：捕获/比较 2 清零标志 (Capture/compare 2 clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 CC2IF 标志将清零。
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 8 **REPOKCF**：重复寄存器更新成功清零标志 (Repetition register update OK clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 REPOK 标志将清零。

位 7 **UECF**：更新事件清零标志 (Update event clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 UE 标志将清零。

位 6 **DOWNCF**：方向变为递减清零标志 (Direction change to down clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 DOWN 标志将清零。
注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 5 **UPCF**：方向变为递增清零标志 (Direction change to UP clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 UP 标志将清零。
注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 4 **ARROKCF**：自动重载寄存器更新成功清零标志 (Autoreload register update OK clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 ARROK 标志将清零。

位 3 保留，必须保持复位值。

位 2 **EXTTRIGCF**：外部触发有效边沿清零标志 (External trigger valid edge clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 EXTTRIG 标志将清零。

位 1 **ARRMCF**：自动重载匹配清零标志 (Autoreload match clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 ARRM 标志将清零。

位 0 **CC1CF**：捕获/比较 1 清零标志 (Capture/compare 1 clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 CC1IF 标志将清零。

29.7.5 LPTIMx 中断使能寄存器 [复用] (LPTIMx_DIER)（x = 1、2）

LPTIMx interrupt enable register [alternate]

该寄存器说明只适用于输出比较模式。关于入捕获模式，请参见下一节。

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UEDE	Res.	Res.	Res.	CMP2OKIE	Res.	Res.	Res.
								rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CC2IE	REPOKIE	UEIE	DOWNIE	UPIE	ARROKIE	CMP1OKIE	EXTTRIGIE	ARRMIE	CC1IE
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23 **UEDE**：更新事件 DMA 请求使能 (Update event DMA request enable)
0：禁止 UE DMA 请求。将“0”写入 UEDE 位可复位相关的 ue_dma_req 信号。
1：使能 UE DMA 请求
注意：如果 LPTIM 没有实现至少 1 个通道，则保留该位。请参见第 29.3 节。

位 22 保留，必须保持复位值。

位 21 保留，必须保持复位值。

位 20 保留，必须保持复位值。

位 19 **CMP2OKIE**：比较寄存器 2 更新成功中断使能 (Compare register 2 update OK interrupt enable)
0：禁止 CMPOK 寄存器 2 中断
1：使能 CMPOK 寄存器 2 中断
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 18:12 保留，必须保持复位值。

位 11 保留，必须保持复位值。

位 10 保留，必须保持复位值。

位 9 **CC2IE**：捕获/比较 2 中断使能 (Capture/compare 2 interrupt enable)
0：禁止捕获/比较 2 中断
1：使能捕获/比较 2 中断
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 8 **REPOKIE**：重复寄存器更新成功中断使能 (Repetition register update OK interrupt Enable)
0：禁止重复寄存器更新成功中断
1：使能重复寄存器更新成功中断

位 7 **UEIE**：更新事件中断使能 (Update event interrupt enable)
0：禁止更新事件中断
1：使能更新事件中断

位 6 **DOWNIE**：方向变为递减中断使能 (Direction change to down Interrupt Enable)
0： 禁止 DOWN 中断
1： 使能 DOWN 中断
注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 5 **UPIE**：方向变为递增中断使能 (Direction change to UP Interrupt Enable)
0： 禁止 UP 中断
1： 使能 UP 中断
注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 4 **ARROKIE**：自动重载寄存器更新成功中断使能 (Autoreload register update OK Interrupt Enable)
0： 禁止 ARROK 中断
1： 使能 ARROK 中断

位 3 **CMP1OKIE**：比较寄存器 1 更新成功中断使能 (Compare register 1 update OK interrupt enable)
0：禁止 CMPOK 寄存器 1 中断
1：使能 CMPOK 寄存器 1 中断

位 2 **EXTTRIGIE**：外部触发有效边沿中断使能 (External trigger valid edge Interrupt Enable)
0： 禁止 EXTTRIG 中断
1： 使能 EXTTRIG 中断

位 1 **ARRMIE**：自动重载匹配中断使能 (Autoreload match Interrupt Enable)
0： 禁止 ARRM 中断
1： 使能 ARRM 中断

位 0 **CC1IE**：捕获/比较 1 中断使能 (Capture/compare 1 interrupt enable)
0：禁止捕获/比较 1 中断
1：使能捕获/比较 1 中断

29.7.6 LPTIMx 中断使能寄存器 [复用] (LPTIMx_DIER)（x = 1、2）

LPTIMx interrupt enable register [alternate]

该寄存器说明只适用于输入捕获模式。关于输出比较模式，请参见上一节。

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	CC2DE	Res.	UEDE	Res.	Res.	Res.	Res.	Res.	Res.	CC1DE
						rw		rw							rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	CC2OIE	CC1OIE	Res.	Res.	CC2IE	REPOKIE	UEIE	DOWNIE	UPIE	ARROKIE	Res.	EXTTRIGIE	ARRMIE	CC1IE
		rw	rw			rw	rw	rw	rw	rw	rw		rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27 保留，必须保持复位值。

位 26 保留，必须保持复位值。

位 25 **CC2DE**：捕获/比较 2 DMA 请求使能 (Capture/compare 2 DMA request enable)
0：禁止 CC2 DMA 请求。将“0”写入 CC2DE 位可复位相关的 ic2_dma_req 信号。
1：使能 CC2 DMA 请求
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 24 保留，必须保持复位值。

位 23 **UEDE**：更新事件 DMA 请求使能 (Update event DMA request enable)
0：禁止 UE DMA 请求。将“0”写入 UEDE 位可复位相关的 ue_dma_req 信号。
1：使能 UE DMA 请求
注意：如果 LPTIM 没有实现至少 1 个通道，则保留该位。请参见第 29.3 节。

位 22:17 保留，必须保持复位值。

位 16 **CC1DE**：捕获/比较 1 DMA 请求使能 (Capture/compare 1 DMA request enable)
0：禁止 CC1 DMA 请求。将“0”写入 CC1DE 位可复位相关的 ic1_dma_req 信号。
1：使能 CC1 DMA 请求
注意：如果 LPTIM 没有实现至少 1 个通道，则保留该位。请参见第 29.3 节。

位 15 保留，必须保持复位值。

位 14 保留，必须保持复位值。

位 13 **CC2OIE**：捕获/比较 2 过捕获中断使能 (Capture/compare 2 over-capture interrupt enable)
0：禁止 CC2 过捕获中断
1：使能 CC2 过捕获中断
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 12 **CC1OIE**：捕获/比较 1 过捕获中断使能 (Capture/compare 1 over-capture interrupt enable)
0：禁止 CC1 过捕获中断
1：使能 CC1 过捕获中断
注意：如果 LPTIM 没有实现至少 1 个通道，则保留该位。请参见第 29.3 节。

位 11 保留，必须保持复位值。

位 10 保留，必须保持复位值。

位 9 **CC2IE**：捕获/比较 2 中断使能 (Capture/compare 2 interrupt enable)
0：禁止捕获/比较 2 中断
1：使能捕获/比较 2 中断
注意：如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见第 29.3 节。

位 8 **REPOKIE**：重复寄存器更新成功中断使能 (Repetition register update OK interrupt Enable)
0：禁止重复寄存器更新成功中断
1：使能重复寄存器更新成功中断

位 7 **UEIE**：更新事件中断使能 (Update event interrupt enable)
0：禁止更新事件中断
1：使能更新事件中断

位 6 **DOWNIE**：方向变为递减中断使能 (Direction change to down Interrupt Enable)
0：禁止 DOWN 中断
1：使能 DOWN 中断
注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 5 **UPIE**：方向变为递增中断使能 (Direction change to UP Interrupt Enable)
0：禁止 UP 中断
1：使能 UP 中断
注意：如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 4 **ARROKIE**：自动重载寄存器更新成功中断使能 (Autoreload register update OK Interrupt Enable)
0：禁止 ARROK 中断
1：使能 ARROK 中断

位 3 保留，必须保持复位值。

位 2 **EXTTRIGIE**：外部触发有效边沿中断使能 (External trigger valid edge Interrupt Enable)
0：禁止 EXTTRIG 中断
1：使能 EXTTRIG 中断

位 1 **ARRMIE**：自动重载匹配中断使能 (Autoreload match Interrupt Enable)
0：禁止 ARRM 中断
1：使能 ARRM 中断

位 0 **CC1IE**：捕获/比较 1 中断使能 (Capture/compare 1 interrupt enable)
0：禁止捕获/比较 1 中断
1：使能捕获/比较 1 中断

29.7.7 LPTIM 配置寄存器 (LPTIM_CFGR)

LPTIM configuration register

偏移地址：0x00C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENC	COUNT MODE	PRE LOAD	Res.	WAVE	TIMOUT	TRIGEN[1:0]		Res.
							rw	rw	rw		rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGSEL[2:0]			Res.	PRESC[2:0]			Res.	TRGFLT[1:0]		Res.	CKFLT[1:0]		CKPOL[1:0]		CKSEL
rw	rw	rw		rw	rw	rw		rw	rw		rw	rw	rw	rw	rw

位 31:30 保留，必须保持复位值。

位 29 保留，必须保持复位值。

位 28:25 保留，必须保持复位值。

位 24 **ENC**：编码器模式使能 (Encoder mode enable)

ENC 位控制编码器模式

0： 禁止编码器模式

1： 使能编码器模式

注意： 如果 LPTIM 不支持编码器模式功能，则保留该位。请参见第 29.3 节。

位 23 **COUNTMODE**：计数器模式使能 (counter mode enabled)

COUNTMODE 位用于选择 LPTIM 使用哪个时钟源来为计数器提供时钟：

0：计数器在每个内部时钟脉冲后递增

1：计数器在 LPTIM 外部 Input1 上的每个有效时钟脉冲后递增

位 22 **PRELOAD**：寄存器更新模式 (Registers update mode)

PRELOAD 位控制 LPTIM_ARR 寄存器、LPTIM_RCR 寄存器和 LPTIM_CCRx 寄存器的更新方式

0： 寄存器在每次 APB 总线写访问后更新

1： 寄存器在当前 LPTIM 周期结束时更新

位 21 保留，必须保持复位值。

位 20 **WAVE**：波形 (Waveform shape)

WAVE 位控制输出波形

0： 停用置 1 一次模式

1： 激活置 1 一次模式

位 19 **TIMOUT**：超时使能 (Timeout enable)

TIMOUT 位控制超时功能

0： 定时器已启动时到达的触发事件将被忽略

1： 定时器已启动时到达的触发事件将复位并重新启动 LPTIM 计数器和重复计数器

位 18:17 **TRIGEN[1:0]**：触发使能和极性 (Trigger enable and polarity)

TRIGEN 位控制 LPTIM 计数器是否由外部触发信号启动。如果已选择由外部触发信号启动，触发有效边沿的配置有以下三种：

00：软件触发（由软件启动计数）

01：上升沿为有效边沿

10：下降沿为有效边沿

11：上升沿和下降沿均为有效边沿

位 16 保留，必须保持复位值。

位 15:13 **TRIGSEL[2:0]**：触发选择器 (Trigger selector)

TRIGSEL 位用于选择作为 LPTIM 触发事件的触发源，可用触发源包括以下 8 种：

000：lptim_ext_trig0

001：lptim_ext_trig1

010：lptim_ext_trig2

011：lptim_ext_trig3

100：lptim_ext_trig4

101：lptim_ext_trig5

110：lptim_ext_trig6

111：lptim_ext_trig7

详细信息，请参见*第 29.4.3 节：LPTIM 输入和触发映射*。

位 12 保留，必须保持复位值。

位 11:9 **PRESC[2:0]**：时钟预分频器 (Clock prescaler)

PRESC 位配置预分频器的分频系数。分频系数可从以下分频系数中选择：

000：/1
001：/2
010：/4
011：/8
100：/16
101：/32
110：/64
111：/128

位 8 保留，必须保持复位值。

位 7:6 **TRGFLT[1:0]**：触发信号的可配置数字滤波器 (Configurable digital filter for trigger)

TRGFLT 值用于设置连续相同采样的数量，若在内部触发信号电平发生变化时检测到此类连续采样，才会将此电平变化视为有效电平切换。必须存在内部时钟源才能使用此功能

00：任何触发信号有效电平变化均视为有效触发
01：触发信号有效电平变化必须至少稳定 2 个时钟周期，才能将其视为有效触发。
10：触发信号有效电平变化必须至少稳定 4 个时钟周期，才能将其视为有效触发。
11：触发信号有效电平变化必须至少稳定 8 个时钟周期，才能将其视为有效触发。

位 5 保留，必须保持复位值。

位 4:3 **CKFLT[1:0]**：外部时钟的可配置数字滤波器 (Configurable digital filter for external clock)

CKFLT 值用于设置连续相同采样的数量，若在外部时钟信号电平发生变化时检测到此类连续采样，才会将此电平变化视为有效电平切换。必须存在内部时钟源才能使用此功能

00：任何外部时钟信号电平变化均视为有效切换
01：外部时钟信号电平变化必须至少稳定 2 个时钟周期，才能将其视为有效切换。
10：外部时钟信号电平变化必须至少稳定 4 个时钟周期，才能将其视为有效切换。
11：外部时钟信号电平变化必须至少稳定 8 个时钟周期，才能将其视为有效切换。

位 2:1 **CKPOL[1:0]**：时钟极性 (Clock polarity)

当 LPTIM 由外部时钟源提供时钟时，CKPOL 位用于配置计数所使用的有效边沿：

00：上升沿为用于计数的有效边沿。
如果将 LPTIM 配置为编码器模式（ENC 位置 1），则编码器子模式 1 有效。
01：下降沿为用于计数的有效边沿。
如果将 LPTIM 配置为编码器模式（ENC 位置 1），则编码器子模式 2 有效。
10：上升沿和下降沿均为有效边沿。当外部时钟信号的上升沿和下降沿均视为有效边沿时，LPTIM 还必须由内部时钟源提供时钟，且内部时钟源频率至少等于外部时钟频率的四倍。
如果将 LPTIM 配置为编码器模式（ENC 位置 1），则编码器子模式 3 有效。
11：不允许

有关编码器模式子模式的更多详细信息，请参见*第 29.4.15 节：编码器模式*。

位 0 **CKSEL**：时钟选择器 (Clock selector)

CKSEL 位用于选择 LPTIM 使用的时钟源：

0： LPTIM 由内部时钟源（APB 时钟或任意内置振荡器）提供时钟
1： LPTIM 由外部时钟源通过 LPTIM 外部 Input1 提供时钟

小心： 必须在 LPTIM 已禁止时（ENABLE 位复位为 0）才能修改 LPTIM_CFGR 寄存器。

29.7.8 LPTIM 控制寄存器 (LPTIM_CR)

LPTIM control register

偏移地址：0x010

复位值：0x0000 0000

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

位 31:5 保留，必须保持复位值。

位 4 **RSTARE**：读操作后复位使能 (Reset after read enable)

该位由软件置 1 和清零。当 RSTARE 置 1 时，对 LPTIM_CNT 寄存器的任何读取访问都会异步复位 LPTIM_CNT 寄存器的内容。

只有在使能 LPTIM 时才能将此位置 1。

位 3 **COUNTRST**：计数器复位 (Counter reset)

此位通过软件置 1，通过硬件清零。当设置为 1 时，该位将触发 LPTIM_CNT 计数器寄存器的同步复位。由于该复位的同步性质，它仅在 3 个 LPTimer 内核时钟周期（LPTimer 内核时钟可能不同于 APB 时钟）的同步延迟之后发生。

只有在使能 LPTIM 时才能将此位置 1。此位由硬件自动复位。

小心： COUNTRST 在已由硬件清零之前，不得由软件置“1”。因此，软件必须在尝试将 COUNTRST 位置“1”之前检查其是否已清零。

位 2 **CNTSTRT**：定时器以连续模式启动 (Timer start in Continuous mode)

此位通过软件置 1，通过硬件清零。

若通过软件启动（TRIGEN[1:0] =“00”），将此位置 1 会使 LPTIM 以连续模式启动。

如果禁止软件启动（TRIGEN[1:0] 不等于“00”），将此位置 1 会使定时器在检测到外部触发信号后立即以连续模式启动。

如果在进行单脉冲模式计数时将此位置 1，则在 LPTIM_ARR 寄存器和 LPTIM_CNT 寄存器下一次匹配时定时器不会停止，LPTIM 计数器将继续以连续模式计数。

只有在使能 LPTIM 时才能将此位置 1。此位由硬件自动复位。

位 1 **SNGSTRT**：LPTIM 以单脉冲模式启动 (LPTIM start in Single mode)

此位通过软件置 1，通过硬件清零。

若通过软件启动（TRIGEN[1:0] =“00”），将此位置 1 会使 LPTIM 以单脉冲模式启动。

如果禁止软件启动（TRIGEN[1:0] 不等于“00”），将此位置 1 会使 LPTIM 在检测到外部触发信号后立即以单脉冲模式启动。

如果在 LPTIM 处于连续计数模式时将此位置 1，LPTIM 将在 LPTIM_ARR 寄存器和 LPTIM_CNT 寄存器下一次匹配时停止。

只有在使能 LPTIM 时才能将此位置 1。此位由硬件自动复位。

位 0 **ENABLE**：LPTIM 使能 (LPTIM enable)

ENABLE 位由软件置 1 和清零。

0：禁止 LPTIM。将“0”写入 ENABLE 位可复位所有 DMA 请求信号（输入捕获和更新事件 DMA 请求）。

1：使能 LPTIM

29.7.9 LPTIM 比较寄存器 1 (LPTIM_CCR1)

LPTIM compare register 1

偏移地址：0x014

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:0 **CCR1[15:0]**：捕获/比较 1 值 (Capture/compare 1 value)

如果通道 CC1 配置为输出：

CCR1 是要装载到捕获/比较 1 寄存器的值。

根据 PRELOAD 选项，如果 PRELOAD 位复位，则 CCR1 寄存器立即更新；如果 PRELOAD 位置 1，则 CCR1 寄存器在生成下一个 LPTIM 更新事件时更新。

捕获/比较寄存器 1 中包含要与计数器 LPTIM_CNT 进行比较并在 OC1 输出相应电平的值。

如果通道 CC1 配置为输入：

CCR1 变为只读，其中包含上一个输入捕获 1 事件发生时的计数器值。只能读取 LPTIM_CCR1 寄存器，无法对其进行编程。

小心： 必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIM_CCR1 寄存器。

29.7.10 LPTIM 自动重载寄存器 (LPTIM_ARR)

LPTIM autoreload register

偏移地址：0x018

复位值：0x0000 0001

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

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

位 31:16 保留，必须保持复位值。

位 15:0 **ARR[15:0]**：自动重载值 (Auto reload value)

ARR 为 LPTIM 的自动重载值。

该值必须严格大于 CCRx[15:0] 的值。

小心： 必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIM_ARR 寄存器。

29.7.11 LPTIM 计数器寄存器 (LPTIM_CNT)

LPTIM counter register

偏移地址：0x01C

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15:0 **CNT[15:0]**：计数器值 (Counter value)

当 LPTIM 通过异步时钟运行时，读取 LPTIM_CNT 寄存器会返回不可靠的值。因此在这种情况下，必须连续执行读访问两次，并验证两次返回的值是否相同。

29.7.12 LPTIM 配置寄存器 2 (LPTIM_CFGR2)

LPTIM configuration register 2

偏移地址：0x024

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2SEL[1:0]		Res.	Res.	IC1SEL[1:0]	
										rw	rw			rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IN2SEL[1:0]		Res.	Res.	IN1SEL[1:0]	
										rw	rw			rw	rw

位 31:22 保留，必须保持复位值。

位 21:20 **IC2SEL[1:0]**：LPTIM 输入捕获 2 选择 (LPTIM input capture 2 selection)

IC2SEL 位用于控制 LPTIM 输入捕获 2 多路复用器，它将 LPTIM 输入捕获 2 连接至其中一个可用输入。

00：lptim_ic2_mux0

01：lptim_ic2_mux1

10：lptim_ic2_mux2

11：lptim_ic2_mux3

有关连接详细信息，请参见*第 29.4.3 节：LPTIM 输入和触发映射*。

位 19:18 保留，必须保持复位值。

位 17:16 **IC1SEL[1:0]**：LPTIM 输入捕获 1 选择 (LPTIM input capture 1 selection)
IC1SEL 位用于控制 LPTIM 输入捕获 1 多路复用器，它将 LPTIM 输入捕获 1 连接至其中一个可用输入。
00：lptim_ic1_mux0
01：lptim_ic1_mux1
10：lptim_ic1_mux2
11：lptim_ic1_mux3
有关连接详细信息，请参见*第 29.4.3 节：LPTIM 输入和触发映射*。

位 15:6 保留，必须保持复位值。

位 5:4 **IN2SEL[1:0]**：LPTIM 输入 2 选择 (LPTIM input 2 selection)
IN2SEL 位用于控制 LPTIM 输入 2 多路复用器，它将 LPTIM 输入 2 连接至其中一个可用输入。
00：lptim_in2_mux0
01：lptim_in2_mux1
10：lptim_in2_mux2
11：lptim_in2_mux3
有关连接详细信息，请参见*第 29.4.3 节：LPTIM 输入和触发映射*。

位 3:2 保留，必须保持复位值。

位 1:0 **IN1SEL[1:0]**：LPTIM 输入 1 选择 (LPTIM input 1 selection)
IN1SEL 位用于控制 LPTIM 输入 1 多路复用器，它将 LPTIM 输入 1 连接至其中一个可用输入。
00：lptim_in1_mux0
01：lptim_in1_mux1
10：lptim_in1_mux2
11：lptim_in1_mux3
有关连接详细信息，请参见*第 29.4.3 节：LPTIM 输入和触发映射*。

29.7.13 LPTIM 重复寄存器 (LPTIM_RCR)

LPTIM repetition register

偏移地址：0x028

复位值：0x0000 0000

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

位 31:8 保留，必须保持复位值。

位 7:0 **REP[7:0]**：重复寄存器值 (Repetition register value)
REP 为 LPTIM 的重复值。

小心： 必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIM_RCR 寄存器。在 PRELOAD = 0 的情况下使用重复计数器时，必须在自动重载匹配事件前至少五个计数器周期更改 LPTIM_RCR 寄存器，否则可能会发生无法预知的行为。

29.7.14 LPTIM 捕获/比较模式寄存器 1 (LPTIM_CCMR1)

LPTIM capture/compare mode register 1

偏移地址：0x02C

复位值：0x0000 0000

这些通道可用于输入（捕获模式）或输出（PWM 模式）模式。通道方向通过配置相应的 CCxSEL 位进行定义。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	IC2F[1:0]		Res.	Res.	IC2PSC[1:0]		Res.	Res.	Res.	Res.	CC2P[1:0]		CC2E	CC2 SEL
		rw	rw			rw	rw					rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	IC1F[1:0]		Res.	Res.	IC1PSC[1:0]		Res.	Res.	Res.	Res.	CC1P[1:0]		CC1E	CC1 SEL
		rw	rw			rw	rw					rw	rw	rw	rw

位 31:30 保留，必须保持复位值。

位 29:28 **IC2F[1:0]**：输入捕获 2 滤波器 (Input capture 2 filter)

该位域用于定义连续相同采样的数量，若在外部输入捕获信号电平发生变化时检测到此类连续采样，才会将此电平变化视为有效电平切换。必须存在内部时钟源才能使用此功能。

00：任何外部输入捕获信号电平变化均视为有效切换

01：外部输入捕获信号电平变化必须至少稳定 2 个时钟周期，才能将其视为有效切换。

10：外部输入捕获信号电平变化必须至少稳定 4 个时钟周期，才能将其视为有效切换。

11：外部输入捕获信号电平变化必须至少稳定 8 个时钟周期，才能将其视为有效切换。

位 27:26 保留，必须保持复位值。

位 25:24 **IC2PSC[1:0]**：输入捕获 2 预分频器 (Input capture 2 prescaler)

该位域用于定义 CC2 输入 (IC2) 的预分频比。

00：无预分频器，捕获输入上每检测到一个边沿便执行捕获

01：每发生 2 个事件便执行一次捕获

10：每发生 4 个事件便执行一次捕获

11：每发生 8 个事件便执行一次捕获

位 23:20 保留，必须保持复位值。

位 19:18 **CC2P[1:0]**：捕获/比较 2 输出极性 (Capture/compare 2 output polarity)

条件：CC2 配置为输出

使能输出模式时，只有位 2 用于设置极性，位 3 为无关位。

0：OC2 高电平有效

1：OC2 低电平有效

条件：CC2 配置为输入

该位域用于选择捕获操作的 IC2 极性。

00：上升沿，电路对 IC2 上升沿敏感

01：下降沿，电路对 IC2 下降沿敏感

10：保留，不使用此配置。

11：上升沿和下降沿，电路对 IC2 的上升沿和下降沿均敏感。

位 17 **CC2E**：捕获/比较 2 输出使能 (Capture/compare 2 output enable)

条件：CC2 配置为输出

0：关闭——OC2 未激活。仅当所有其他 LPTIM 通道均被禁止时，将“0”写入 CC2E 位才可复位 ue_dma_req 信号。

1：开启——在相应输出引脚上输出 OC2 信号

条件：CC2 配置为输入

此位决定了是否可以实际将计数器值捕获到输入捕获/比较寄存器 2 (LPTIM_CCR2) 中。

0：禁止捕获。将“0”写入 CC2E 位可复位相关的 ic2_dma_req 信号。

1：使能捕获。

位 16 **CC2SEL**：捕获/比较 2 选择 (Capture/compare 2 selection)

该位域用于定义通道的方向、输入（捕获）或输出模式。

0：CC2 通道配置为输出 PWM 模式

1：CC2 通道配置为输入捕获模式

位 15:14 保留，必须保持复位值。

位 13:12 **IC1F[1:0]**：输入捕获 1 滤波器 (Input capture 1 filter)

该位域用于定义连续相同采样的数量，若在外部输入捕获信号电平发生变化时检测到此类连续采样，才会将此电平变化视为有效电平切换。必须存在内部时钟源才能使用此功能。

00：任何外部输入捕获信号电平变化均视为有效切换

01：外部输入捕获信号电平变化必须至少稳定 2 个时钟周期，才能将其视为有效切换。

10：外部输入捕获信号电平变化必须至少稳定 4 个时钟周期，才能将其视为有效切换。

11：外部输入捕获信号电平变化必须至少稳定 8 个时钟周期，才能将其视为有效切换。

位 11:10 保留，必须保持复位值。

位 9:8 **IC1PSC[1:0]**：输入捕获 1 预分频器 (Input capture 1 prescaler)

该位域用于定义 CC1 输入 (IC1) 的预分频比。

00：无预分频器，捕获输入上每检测到一个边沿便执行捕获

01：每发生 2 个事件便执行一次捕获

10：每发生 4 个事件便执行一次捕获

11：每发生 8 个事件便执行一次捕获

位 7:4 保留，必须保持复位值。

位 3:2 **CC1P[1:0]**：捕获/比较 1 输出极性 (Capture/compare 1 output polarity)

条件：CC1 配置为输出

使能输出模式时，只有位 2 用于设置极性，位 3 为无关位。

0：OC1 高电平有效，LPTIM 输出反映 LPTIM_ARR 寄存器和 LPTIM_CCRx 寄存器之间的比较结果

1：OC1 低电平有效，LPTIM 输出反映与 LPTIM_ARR 寄存器和 LPTIM_CCRx 寄存器之间的比较结果相反的值

条件：CC1 配置为输入

该位域用于选择捕获操作的 IC1 极性。

00：上升沿，电路对 IC1 上升沿敏感

01：下降沿，电路对 IC1 下降沿敏感

10：保留，不使用此配置。

11：上升沿和下降沿，电路对 IC1 的上升沿和下降沿均敏感。

位 1 **CC1E**：捕获/比较 1 输出使能 (Capture/compare 1 output enable)

条件：CC1 配置为输出

0：关闭——OC1 未激活。仅当所有其他 LPTIM 通道均被禁止时，将“0”写入 CC1E 位才可复位 ue_dma_req 信号。

1：开启——在相应输出引脚上输出 OC1 信号

条件：CC1 配置为输入

此位决定了是否可以实际将计数器值捕获到输入捕获/比较寄存器 1 (LPTIM_CCR1) 中。

0：禁止捕获。将“0”写入 CC1E 位可复位相关的 ic1_dma_req 信号。

1：使能捕获。

位 0 **CC1SEL**：捕获/比较 1 选择 (Capture/compare 1 selection)

该位域用于定义通道的方向、输入（捕获）或输出模式。

0：CC1 通道配置为输出 PWM 模式

1：CC1 通道配置为输入捕获模式

小心： 对 LPTIM_CCMRx 寄存器执行写操作后，必须延迟一段时间才能对该寄存器执行新的写操作，该延迟时间必须等于或大于 (PRESC × 3) 个内核时钟周期的值，其中 PRESC[2:0] 为时钟十进制分频系数（1，2，4，..128）。任何违反此延迟要求的连续写操作都会造成不可预知的结果。

小心： 仅当通道 x 被禁止（CCxE 位复位为 0）时，才可以修改 CCxSEL、ICxF[1:0]、CCxP[1:0] 和 ICxPSC[1:0] 位域。

29.7.15 LPTIM 比较器寄存器 2 (LPTIM_CCR2)

LPTIM compare register 2

偏移地址：0x034

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:0 **CCR2[15:0]**：捕获/比较 2 值 (Capture/compare 2 value)

如果通道 CC2 配置为输出：

CCR2 是要装载到捕获/比较 2 寄存器的值。

根据 PRELOAD 选项，如果 PRELOAD 位复位，则 CCR2 寄存器立即更新；如果 PRELOAD 位置 1，则 CCR2 寄存器在生成下一个 LPTIM 更新事件时更新。

捕获/比较寄存器 2 中包含要与计数器 LPTIM_CNT 进行比较并在 OC2 输出相应电平的值。

如果通道 CC2 配置为输入：

CCR2 变为只读，其中包含上一个输入捕获 2 事件发生时的计数器值。只能读取 LPTIM_CCR2 寄存器，无法对其进行编程。

小心： 必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIM_CCR2 寄存器。

注意： 如果 LPTIM 实现的通道少于 2 个，则保留该寄存器。请参见*第 29.3 节：LPTIM 实现*。

29.7.16 LPTIM 寄存器映射

下表对 LPTIM 寄存器进行了汇总。

表 221. LPTIM 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	LPTIMx_ISR（x = 1、2）Output compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIEROK	Res.	Res.	Res.	Res.	CMP2OK(1)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2IF(1)	REPOK	UE	DOWN(2)	UP(2)	ARROK	CMP1OK	EXTTRIG	ARRM	CC1IF
	Reset value								0					0										0	0	0	0	0	0	0	0	0	0
	LPTIMx_ISR（x = 1、2）Input capture mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIEROK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2OF(1)	CC1OF	Res.	Res.	CC2IF(1)	REPOK	UE	DOWN(2)	UP(2)	ARROK	Res.	EXTTRIG	ARRM	CC1IF
	Reset value								0											0	0			0	0	0	0	0	0		0	0	0
0x004	LPTIMx_ICR（x = 1、2）Output compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIEROKCF	Res.	Res.	Res.	Res.	CMP2OKCF(1)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2CF(1)	REPOKCF	UECF	DOWNCF(2)	UPCF(2)	ARROKCF	CMP1OKCF	EXTTRIGCF	ARRMCF	CC1CF
	Reset value								0					0										0	0	0	0	0	0	0	0	0	0
	LPTIMx_ICR（x = 1、2）Input capture mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIEROKCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2OCF(1)	CC1OCF	Res.	Res.	CC2CF(1)	REPOKCF	UECF	DOWNCF(2)	UPCF(2)	ARROKCF	Res.	EXTTRIGCF	ARRMCF	CC1CF
	Reset value								0											0	0			0	0	0	0	0	0		0	0	0
0x008	LPTIMx_DIER（x = 1、2）Output compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UEDE	Res.	Res.	Res.	CMP2OKIE(1)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2IE(1)	REPOKIE	UEIE	DOWNIE(2)	UPIE(2)	ARROKIE	CMP1OKIE	EXTTRIGIE	ARRMIE	CC1IE
	Reset value									0				0										0	0	0	0	0	0	0	0	0	0
	LPTIMx_DIER（x = 1、2）Input capture mode	Res.	Res.	Res.	Res.	Res.	Res.	CC2DE(1)	Res.	UEDE	Res.	Res.	Res.	Res.	Res.	Res.	CC1DE	Res.	Res.	CC2OIE(1)	CC1OIE	Res.	Res.	CC2IE(1)	REPOKIE	UEIE	DOWNIE(2)	UPIE(2)	ARROKIE	Res.	EXTTRIGIE	ARRMIE	CC1IE
	Reset value							0		0							0			0	0			0	0	0	0	0	0		0	0	0
0x00C	LPTIM_CFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENC(2)	COUNTMODE	PRELOAD	Res.	WAVE	TIMOUT	TRIGEN		Res.	TRIGSEL[2:0]		Res.	PRESC			Res.	TRGFLT		CKFLT		Res.	CKPOL		CKSEL
	Reset value								0	0	0		0	0	0	0		0	0	0		0	0	0		0	0		0	0	0	0	0
0x010	LPTIM_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSTARE	COUNTRST	CNTSTRT	SNGSTRT	ENABLE
	Reset value																												0	0	0	0	0
0x014	LPTIM_CCR1	CCR1[15:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x018	LPTIM_ARR	ARR[15:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
0x01C	LPTIM_CNT	CNT[15:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 221. LPTIM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x024	LPTIM_ CFGR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2SEL[1:0]		Res.	Res.	IC1SEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IN2SEL[1:0]		Res.	Res.	IN1SEL[1:0]	
	Reset value											0	0			0	0											0	0			0	0
0x028	LPTIM_RCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x02C	LPTIM_CCMR1	Res.	Res.	IC2F[1:0]		Res.	Res.	IC2PSC[1:0]		Res.	Res.	Res.	Res.	CC2P[1:0]		CC2E	CC2SEL	Res.	Res.	IC1F[1:0]		Res.	Res.	IC1PSC[1:0]		Res.	Res.	Res.	Res.	CC1P[1:0]		CC1E	CC1SEL
	Reset value			0	0			0	0					0	0	0	0			0	0			0	0					0	0	0	0
0x034	LPTIM_CCR2(3)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. 如果 LPTIM 没有实现至少 2 个通道，则保留该位。请参见*第 29.3 节：LPTIM 实现*。
2. 如果 LPTIM 不支持编码器模式功能，则保留该位。请参见*第 29.3 节：LPTIM 实现*。
3. 如果 LPTIM 实现的通道少于 2 个，则保留该寄存器。请参见*第 29.3 节：LPTIM 实现*。

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

30 独立看门狗 (IWDG)

30.1 简介

独立看门狗 (IWDG) 外设可检测由于软件或硬件故障而引起的故障，因此具有较高的安全性。

IWDG 的时钟由独立时钟提供，即便在主时钟发生故障时仍然保持工作状态。

此外，看门狗功能在 V_{DD} 电压域中执行，因此即便在低功耗模式下，IWDG 仍然保持正常工作。有关该产品中 IWDG 的功能，请参见*第 30.3 节*。

IWDG 最适合那些需要将看门狗作为一个在主程序之外的进程完全独立运行的应用，它能够在这类应用中非常可靠地检测任何意外行为。

30.2 IWDG 主要特性

- 12 位递减计数器
- 双电压域，可在低功耗模式下工作
- 独立时钟
- 提前唤醒中断生成
- 复位生成
  - 发生超时时
  - 刷新超出预期窗口时

30.3 IWDG 实现

表 222. IWDG 特性 (1)

IWDG 模式/特性	IWDG
LSI 用作 IWDG 内核时钟 (iwdg_ker_ck)	X
窗口功能	X
提前唤醒中断生成	X
系统复位生成 (2)	X
可在系统停止模式下工作	X
可在系统待机模式下工作	X
可在系统停止模式下生成中断	X
可在系统待机模式下生成中断	-
可在微控制器进入调试模式时被冻结 (3)	X
用于控制停止模式下活动的选项字节 (4)	X
用于控制待机模式下活动的选项字节 (5)	X
用于控制硬件模式的选项字节 (6)	X

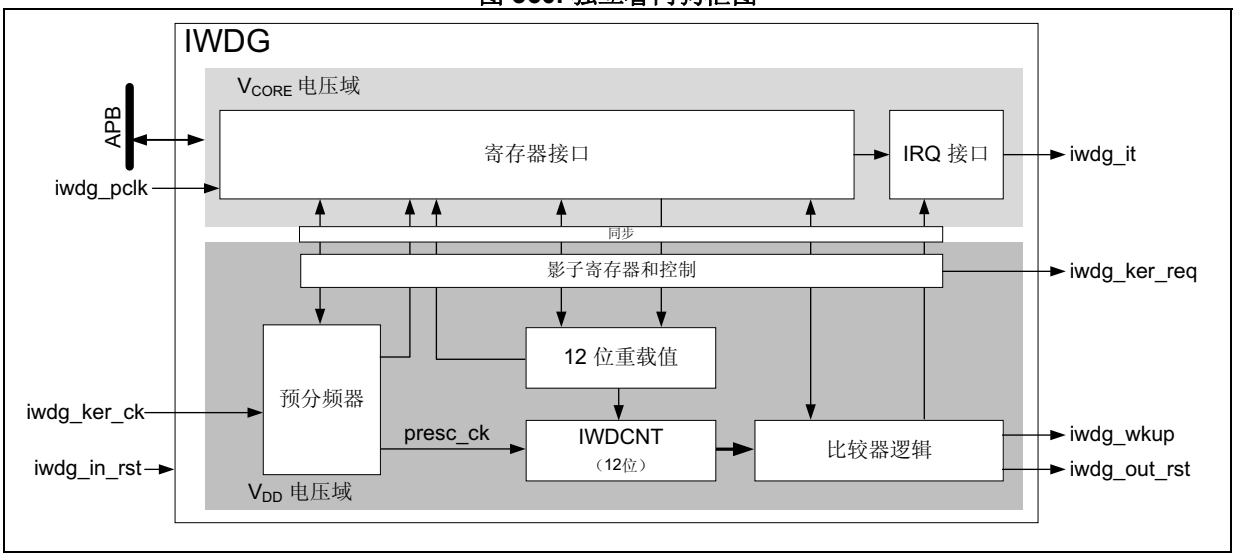
1. “X” = 支持，“-” = 不支持。
2. 有关详细信息，请参见 RCC 部分。
3. 通过 DBG 部分的 DBG_IWDG_STOP 进行控制。
4. 通过 FLASH 部分的选项字节 IWDG_STOP 进行控制。
5. 通过 FLASH 部分的选项字节 IWDG_STDBY 进行控制。
6. 通过 FLASH 部分的选项字节 IWDG_SW 进行控制。

30.4 IWDG 功能描述

30.4.1 IWDG 框图

图 360 给出了独立看门狗模块的功能框图。

图 360. 独立看门狗框图



寄存器和 IRQ 接口位于 V_{CORE} 电压域。看门狗功能本身位于 V_{DD} 电压域，在低功耗模式下仍然保持正常工作。有关 IWDG 功能，请参见*第 30.3 节*。

寄存器接口和 IRQ 接口的时钟主要由 APB 时钟 (iwdg_pclk) 提供，而看门狗功能的时钟则由专用内核时钟 (iwdg_ker_ck) 提供。可通过同步机制在两个电压域之间进行数据交换。请注意，寄存器接口中的大多数寄存器均映射到 V_{DD} 电压域中。

IWDG 递减计数器 (IWDCNT) 的时钟由预分频时钟 (presc_ck) 提供。预分频时钟是由内核时钟 iwdg_ker_ck 根据 PR[3:0] 位域经预分频器分频后生成的。

30.4.2 IWDG 内部信号

表 223 详细列出了 IWDG 内部信号。

表 223. IWDG 内部输入/输出信号

信号名称	信号类型	说明
iwdg_ker_ck	输入	IWDG 内核时钟
iwdg_ker_req	输入	IWDG 内核时钟请求
iwdg_pclk	输入	IWDG APB 时钟
iwdg_out_rst	输出	IWDG 复位输出
iwdg_in_rst	输入	IWDG 复位输入
iwdg_wkup	输出	IWDG 唤醒事件
iwdg_it	输出	IWDG 提前唤醒中断

30.4.3 软件和硬件看门狗模式

这两种看门狗模式允许应用选择 IWDG 的使能方式：通过软件命令使能（软件看门狗模式）或自动使能（硬件看门狗模式）。软件和硬件看门狗模式对于所有其他功能的工作方式没有影响。

软件看门狗模式为默认工作模式。将值 0x0000 CCCC 写入 *IWDG 密钥寄存器 (IWDG_KR)* 可启动独立看门狗，IWDCNT 从复位值 0xFFF 开始递减计数。

在硬件看门狗模式下，独立看门狗在上电时或每次复位（通过 iwdg_in_rst）时自动启动。IWDCNT 递减计数器从复位值 0xFFF 开始递减计数。硬件看门狗模式功能通过器件选项位使能（有关详细信息，请参见*第 30.3 节*）。

使能 IWDG 时，ONF 标志置 1。

当 IWDCNT 计数到 0x000 时，会生成复位信号（iwdg_out_rst 置为有效）。

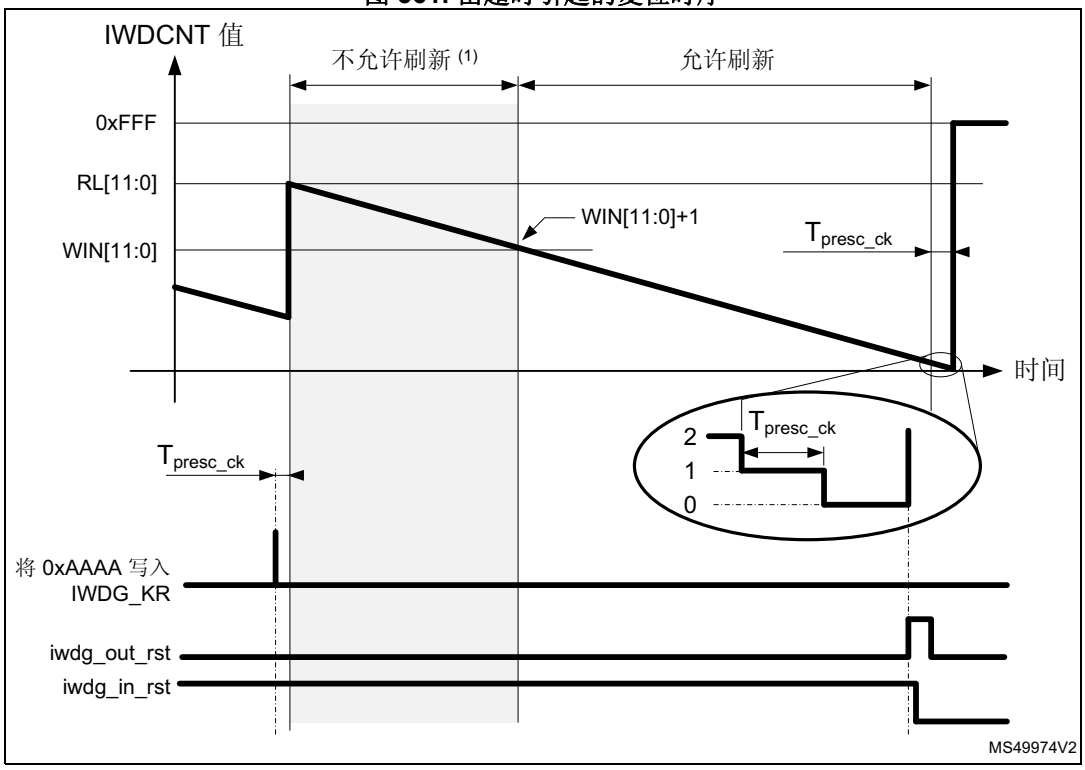
任何时候将键值 0x0000 AAAA 写入 *IWDG 密钥寄存器 (IWDG_KR)*，IWDG_RLR 的值都会被重载到 IWDCNT，从而避免产生看门狗复位。

由于存在再同步延迟，因此必须在 IWDCNT 递减计数器计数到 1 之前刷新 IWDG。

IWDG 一旦启动，便只有在复位（iwdg_in_rst 置为有效）时才能停止。

如*图 361* 所示，在执行刷新命令的一个 presc_ck 周期后，IWDCNT 会重新装载 RL[11:0] 的内容。

图 361. 由超时引起的复位时序



1. 激活窗口选项时。

如果在 IWDCNT 计数到 1 前未刷新 IWDG，IWDG 会产生复位（即 iwdg_out_rst 置为有效）。随后，RCC 会复位 IWDG（iwdg_in_rst 置为有效）以清除复位源。

30.4.4 窗口选项

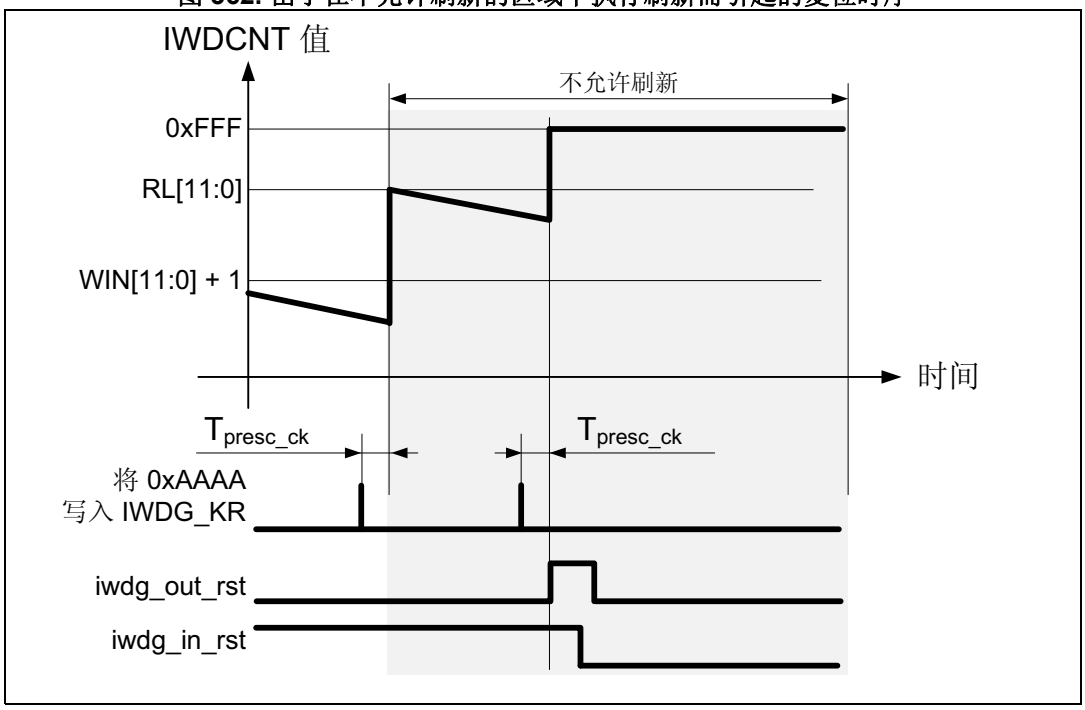
通过在 *IWDG 窗口寄存器 (IWDG_WINR)* 中设置合适的窗口，IWDG 也可以用作窗口看门狗。

当计数器值大于 WIN[11:0] + 1 时，如果执行重载操作，则会产生复位。WIN[11:0] 位于 *IWDG 窗口寄存器 (IWDG_WINR)* 中。如*图 362* 所示，复位信号在意外执行刷新命令的一个 presc_ck 周期后生成。

IWDG 窗口寄存器 (IWDG_WINR) 的默认值为 0x0000 0FFF，因此，如果不更新此默认值，将禁止窗口选项。

只要窗口值发生变化，递减计数器 (IWDCNT) 就会立即重载 RL[11:0] 的值，以便估计下一次执行刷新命令的位置。

图 362. 由于在不允许刷新的区域中执行刷新而引起的复位时序



使能窗口选项时配置 IWDG

1. 通过在 *IWDG 密钥寄存器 (IWDG_KR)* 中写入 0x0000 CCCC 来使能 IWDG。
2. 通过在 *IWDG 密钥寄存器 (IWDG_KR)* 中写入 0x0000 5555 来使能寄存器访问。
3. 通过编程 *IWDG 预分频器寄存器 (IWDG_PR)* 来配置 IWDG 预分频器。
4. 对 *IWDG 重载寄存器 (IWDG_RLR)* 进行写操作。
5. 如果需要，可以通过在 *IWDG 提前唤醒中断寄存器 (IWDG_EWCR)* 中写入适当的值来使能提前唤醒中断并编程提前唤醒比较器。
6. 对 *IWDG 窗口寄存器 (IWDG_WINR)* 进行写操作。这将自动使用 RL[11:0] 的值重载 IWDCNT 递减计数器。
7. 等待寄存器更新 (IWDG_SR = 0x0000 0000)。
8. 将 0x0000 0000 写入 *IWDG 密钥寄存器 (IWDG_KR)* 以对寄存器进行写保护。

注意： *如果应用无需在此序列完成后禁止 APB 时钟，则可以跳过步骤 7。*

禁止窗口选项时配置 IWDG

不使用窗口选项时，可按以下步骤配置 IWDG：

1. 通过在 *IWDG 密钥寄存器 (IWDG_KR)* 中写入 0x0000 CCCC 来使能 IWDG。
2. 通过在 *IWDG 密钥寄存器 (IWDG_KR)* 中写入 0x0000 5555 来使能寄存器访问。
3. 通过编程 *IWDG 预分频器寄存器 (IWDG_PR)* 来配置预分频器。
4. 对 *IWDG 重载寄存器 (IWDG_RLR)* 进行写操作。
5. 如果需要，可以通过在 *IWDG 提前唤醒中断寄存器 (IWDG_EWCR)* 中写入适当的值来使能提前唤醒中断并编程提前唤醒比较器。
6. 等待寄存器更新 (IWDG_SR = 0x0000 0000)。
7. 使用 RL[11:0] 的值刷新计数器，然后将 0x0000 AAAA 写入 *IWDG 密钥寄存器 (IWDG_KR)* 以对寄存器进行写保护。

更新窗口比较器

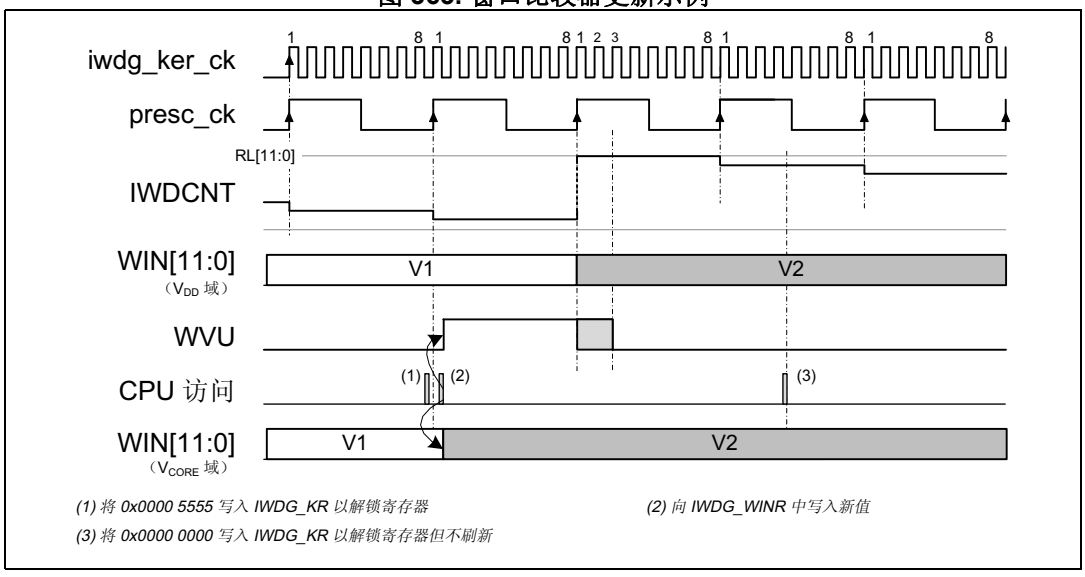
窗口比较器可在 IWDG 运行期间更新。IWDCNT 也会重新装载。可执行以下序列来更新窗口比较器：

1. 通过在 IWDG 密钥寄存器 (IWDG_KR) 中写入 0x0000 5555 来使能寄存器访问。
2. 写入 IWDG 窗口寄存器 (IWDG_WINR)。这将自动使用 RL[11:0] 的值重载 IWDCNT 递减计数器。
3. 等待 WVU = 0
4. 通过向 IWDG_KR 中写入 0x0000 0000 来锁定寄存器

如果应用无需在此序列完成后禁止 APB 时钟，则可以跳过步骤 *3*。

图 363 显示了该序列。只要写入 IWDG_WINR 寄存器，WVU 标志就会置 1。新的窗口值以及通过 RL[11:0] 重载 IWDCNT 的操作会在 presc_ck 的下一个上升沿生效。在最坏情况下，WVU 标志会在两个内核时钟周期后恢复为 0。因此，WVU 保持置 1 的时间最长为一个 presc_ck 周期加上两个内核时钟周期。

图 363. 窗口比较器更新示例



30.4.5 调试

当处理器进入调试模式时（内核停止），IWDCNT 递减计数器会根据产品的调试功能选择继续正常工作或者停止工作。有关该产品功能的详细信息，请参见*第 30.3 节*。

30.4.6 寄存器访问保护

IWDG 预分频器寄存器 (IWDG_PR)、*IWDG 重载寄存器 (IWDG_RLR)*、*IWDG 提前唤醒中断寄存器 (IWDG_EWCR)* 和 *IWDG 窗口寄存器 (IWDG_WINR)* 具有写访问保护。要修改上述寄存器，首先向 *IWDG 密钥寄存器 (IWDG_KR)* 写入 0x0000 5555。而写入其他值则会破坏该序列，从而使寄存器访问保护再次生效。这表示重载操作（即写入 0x0000 AAAA）也会启动写保护功能。

状态寄存器指示预分频值、递减计数器重载值或窗口值是否正在被更新。

30.5 IWDG 低功耗模式

根据选项字节配置，IWDG 可在低功耗模式下继续计数或停止计数。有关详细信息，请参见*第 30.3 节*。

表 224. 低功耗模式对 IWDG 的影响

模式	说明
睡眠	无影响。IWDG 中断可使器件退出睡眠模式。
停止	IWDG 保持工作状态或不工作，具体取决于选项字节配置。 有关详细信息，请参见*第 30.3 节*。 IWDG 中断可使器件退出停止模式。
待机	IWDG 保持工作状态或不工作，具体取决于选项字节配置。 有关详细信息，请参见*第 30.3 节*。 IWDG 中断不会使器件退出待机模式。

30.6 IWDG 中断

IWDG 可根据递减计数器的值生成提前中断。将 *IWDG 提前唤醒中断寄存器 (IWDG_EWCR)* 寄存器中的 EWIE 位置 1 可使能提前中断。

应用可以使用比较器值 (EWIT[11:0]) 定义必须生成提前中断的位置。

当 IWDCNT 递减计数器计数到 EWIT[11:0] - 1 的值时，iwdg_wkup 会激活，系统可根据需要退出低功耗模式。

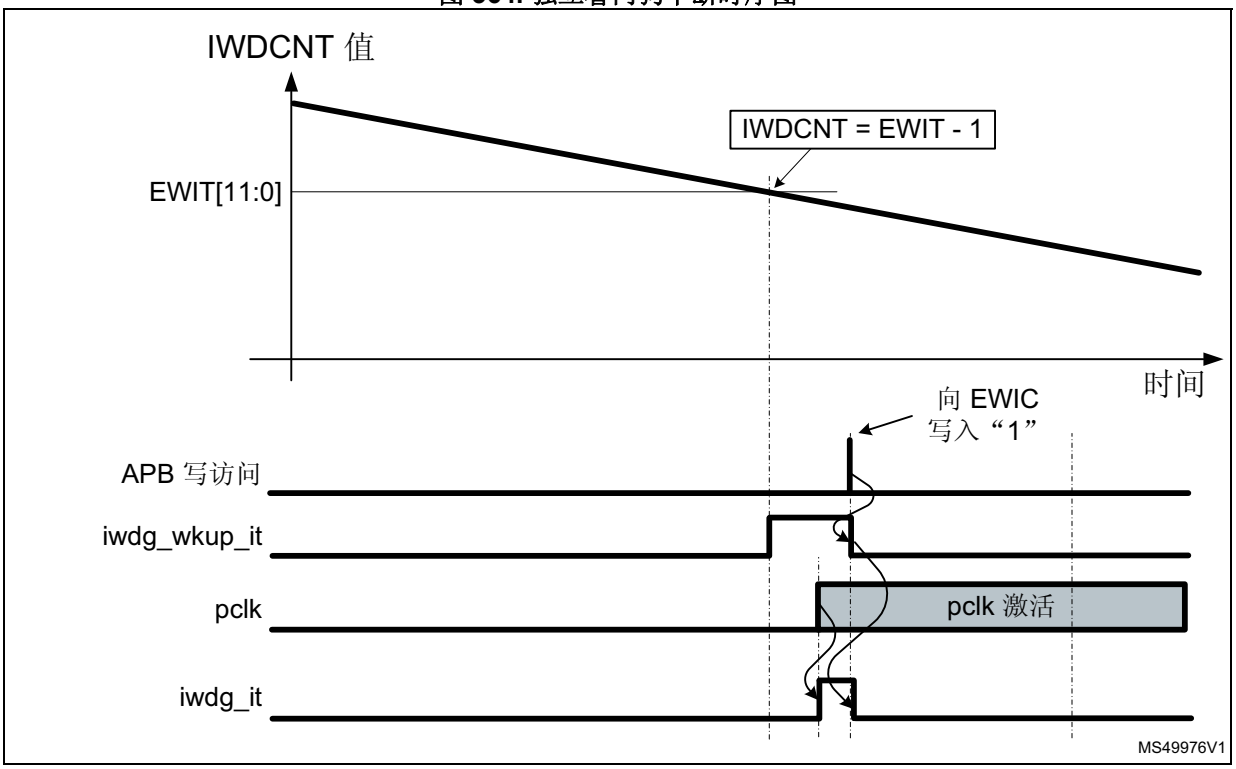
当 APB 时钟可用时，iwdg_it 也会激活。

此外，*IWDG 状态寄存器 (IWDG_SR)* 的 EWIF 标志置 1。

通过向 *IWDG 提前唤醒中断寄存器 (IWDG_EWCR)* 的 EWIC 位写入“1”来确认 EWI 中断。

对 IWDG_EWCR 寄存器的写操作还会触发将递减计数器 (IWDCNT) 刷新为重载值 RL[11:0]。

图 364. 独立看门狗中断时序图



如果在产生看门狗复位之前必须执行特定的安全操作或数据记录，则可使用提前唤醒中断 (EWI)。

更改提前唤醒比较器值

通过执行以下序列，可以随时更改提前唤醒比较器值或使能/禁止中断生成：

1. 通过在 *IWDG 密钥寄存器 (IWDG_KR)* 中写入 0x0000 5555 来使能寄存器访问。
2. 通过在 *IWDG 提前唤醒中断寄存器 (IWDG_EWCR)* 中写入适当的值来使能/禁止提前唤醒中断和/或编程提前唤醒比较器。
3. 等待 EWU =“0”，EWU 位于 *IWDG 状态寄存器 (IWDG_SR)* 中。
4. 将 0x0000 0000 写入 *IWDG 密钥寄存器 (IWDG_KR)* 以对寄存器进行写保护。

如果应用无需在此序列完成后禁止 APB 时钟，则可以跳过步骤 *3*。

图 364 显示了该序列。只要写入 IWDG_EWCR 寄存器，EWU 标志就会置 1。新的比较器值以及通过 RL[11:0] 重载 IWDCNT 的操作会在 presc_ck 的下一个上升沿生效。在最坏情况下，EWU 标志会在两个内核时钟周期后恢复为 0。因此，EWU 保持置 1 的时间最长为一个 presc_ck 周期加上两个内核时钟周期。

图 365. 提前唤醒比较器更新示例

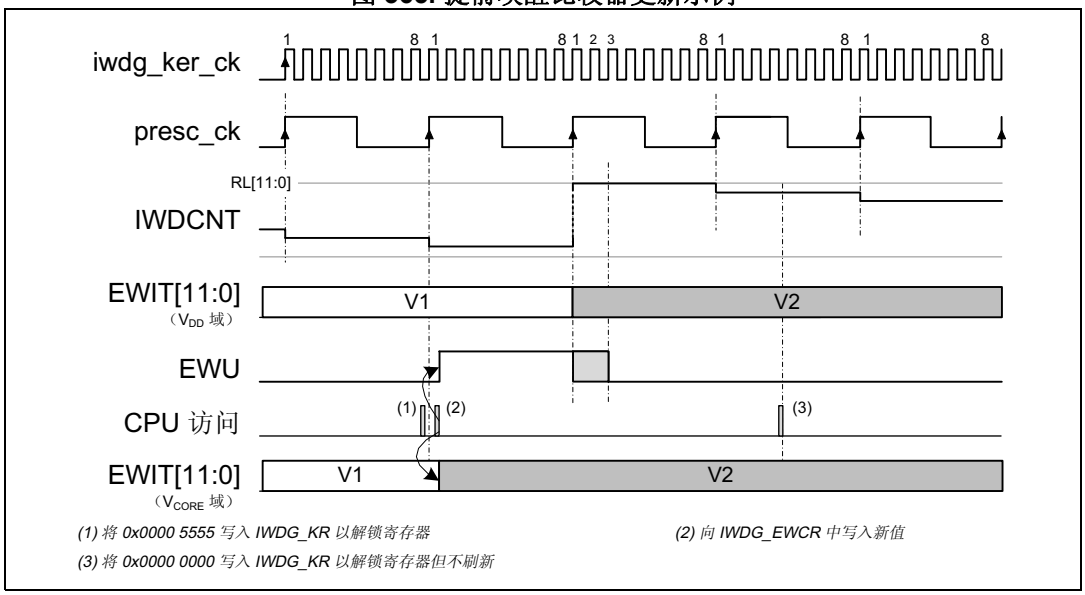


表 225 汇总了 IWDG 中断请求。

表 225. IWDG 中断请求

中断事件	事件标志	中断清除方法	中断使能控制位	激活的中断	
				iwdg_it	iwdg_wkup_it
IWDCNT 计数到 EWIT 值	EWIF	向 EWIC 写入“1”	EWIE	是(1)	是(2)

1. 当 iwdg_pclk 输入上存在时钟时生成。
2. 当 iwdg_ker_ck 输入上存在时钟时生成。

30.7 IWDG 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节*。

外设寄存器可按半字（16 位）或字（32 位）访问。

寄存器接口中的大多数寄存器均映射到 V_{DD} 电压域中。当 iwdg_in_rst 置为有效时，位于 V_{DD} 电压域中的看门狗逻辑和影子寄存器将会复位。

当应用回读看门狗寄存器时，硬件会将相应影子寄存器的值传输到寄存器接口。

当应用写入看门狗寄存器时，硬件会更新相应的影子寄存器。

30.7.1 IWDG 密钥寄存器 (IWDG_KR)

IWDG key register

偏移地址：0x00

复位值：0x0000 0000

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

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

位 31:16 保留，必须保持复位值。

位 15:0 **KEY[15:0]**：键值 (Key value)（只能写，读为 0x0000）

这些位可提供多种功能，具体取决于应用写入的值：

- 0xAAAA：将 RL[11:0] 值重载到 IWDCNT 递减计数器（看门狗刷新），并对寄存器进行写保护。必须每隔一段时间便通过软件写入该值，否则当计数器计数到 0 时，看门狗会产生复位。
- 0x5555：使能对寄存器的写访问。
- 0xCCCC：使能看门狗（选择硬件看门狗选项时除外）并对寄存器进行写保护。
- 除 0x5555 以外的其他值：对寄存器进行写保护。

请注意，只有 IWDG_PR、IWDG_RLR、IWDG_EWCR 和 IWDG_WINR 寄存器具有写保护机制。

30.7.2 IWDG 预分频器寄存器 (IWDG_PR)

IWDG prescaler register

偏移地址：0x04

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **PR[3:0]**：预分频系数 (Prescaler divider)

这些位受写访问保护，请参见*第 30.4.6 节*。通过软件设置这些位来选择计数器时钟的预分频系数。若要更改预分频系数，*IWDG 状态寄存器 (IWDG_SR)* 的 PVU 位必须复位。

0000：4 分频
0001：8 分频
0010：16 分频
0011：32 分频
0100：64 分频
0101：128 分频
0110：256 分频
0111：512 分频
其他值：1024 分频

注意：读取该寄存器会返回 V_{DD} 电压域的预分频器值。如果正在对该寄存器执行写操作，则读取的值可能不是最新的/ 有效的。因此，只有在 IWDG 状态寄存器 (IWDG_SR) 中的 PVU 位复位时，从寄存器读取的值才有效。

30.7.3 IWDG 重载寄存器 (IWDG_RLR)

IWDG reload register

偏移地址：0x08

复位值：0x0000 0FFF

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

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

位 31:12 保留，必须保持复位值。

位 11:0 **RL[11:0]**：看门狗计数器重载值 (Watchdog counter reload value)

这些位受写访问保护，请参见*第 30.4.6 节*。这个值由软件设置，每次对 *IWDG 密钥寄存器 (IWDG_KR)* 写入值 0xAAAA 时，这个值就会重载到看门狗计数器中。之后，看门狗计数器便从该装载的值开始递减计数。超时周期由该值和预分频器时钟共同决定。不建议将 RL[11:0] 设置为小于 2 的值。

若要更改重载值，*IWDG 状态寄存器 (IWDG_SR)* 中的 RVU 位必须复位。

注意：读取该寄存器会返回 V_{DD} 电压域的重载值。如果正在对该寄存器执行写操作，则读取的值可能不是最新的/ 有效的。因此，只有在 IWDG 状态寄存器 (IWDG_SR) 中的 RVU 位复位时，从寄存器读取的值才有效。

30.7.4 IWDG 状态寄存器 (IWDG_SR)

IWDG status register

偏移地址：0x0C

复位值：0x0000 0000 (0xFFFF FEFF)

该寄存器包含各种状态标志。请注意，括号内的掩码值表示 ONF 位的复位值尚未定义。当 IWDG 配置为软件模式时，ONF 位的复位值为 0；当 IWDG 配置为硬件模式时，ONF 位的复位值为 1。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	EWIF	Res.	Res.	Res.	Res.	Res.	ONF	Res.	Res.	Res.	Res.	EWU	WVU	RVU	PVU
	r						r					r	r	r	r

位 31:15 保留，必须保持复位值。

位 14 **EWIF**：看门狗提前中断标志 (Watchdog early interrupt flag)

该位由硬件置“1”以指示存在挂起的提前中断。该位必须由软件通过向 IWDG_EWCR 的 EWIC 位写入“1”来清零。

位 13:9 保留，必须保持复位值。

位 8 **ONF**：看门狗使能状态位 (Watchdog enable status bit)

IWDG 启动后，该位立即由硬件置“1”。在软件模式下，该位保持为“1”，直到 IWDG 复位为止。在硬件模式下，该位始终置“1”。

0：未激活 IWDG

1：已激活 IWDG，需要应用定期刷新

位 7:4 保留，必须保持复位值。

位 3 **EWU**：看门狗中断比较器值更新 (Watchdog interrupt comparator value update)

该位由硬件置 1 以指示中断比较器值 (EWIT[11:0]) 或 EWIE 正在更新。当在 V_{DD} 电压域下完成更新操作后（最多需要一个 presc_ck 周期加上两个 IWDG 内核时钟 (iwdg_ker_ck) 周期），会通过硬件将该位复位。

EWIT[11:0] 和 EWIE 位域只有在 EWU 位复位时才可更新。

位 2 **WVU**：看门狗计数器窗口值更新 (Watchdog counter window value update)

该位由硬件置 1 以指示窗口值正在更新。当在 V_{DD} 电压域下完成重载值更新操作后（最多需要一个 presc_ck 周期加上两个 IWDG 内核时钟 (iwdg_ker_ck) 周期），会通过硬件将该位复位。

窗口值只有在 WVU 位复位时才可更新。

该位只有在通用“窗口”= 1 时才生成。

位 1 **RVU**：看门狗计数器重载值更新 (Watchdog counter reload value update)

该位由硬件置 1 以指示重载值正在更新。当在 V_{DD} 电压域下完成重载值更新操作后（最多需要六个 IWDG 内核时钟 (iwdg_ker_ck) 周期），会通过硬件将该位复位。

重载值只有在 RVU 位复位时才可更新。

位 0 **PVU**：看门狗预分频器值更新 (Watchdog prescaler value update)

该位由硬件置 1 以指示预分频器值正在更新。当在 V_{DD} 电压域下完成预分频器更新操作后（最多需要六个 IWDG 内核时钟 (iwdg_ker_ck) 周期），会通过硬件将该位复位。

预分频器值只有在 PVU 位复位时才可更新。

注意： *如果应用使用多个重载值、预分频器值、提前中断位置值或窗口值，则必须等到 RVU 位复位后才能更改重载值，必须等到 PVU 位复位后才能更改预分频器值，必须等到 WVU 位复位后才能更改窗口值，而且必须等到 EWU 位复位后才能更改提前中断位置值。在更新预分频器和/或重载/窗口/提前中断值之后，无需等到 RVU、PVU、WVU 或 EWU 复位后再继续执行代码（进入低功耗模式时除外）。*

30.7.5 IWDG 窗口寄存器 (IWDG_WINR)

IWDG window register

偏移地址：0x10

复位值：0x0000 0FFF

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

位 31:12 保留，必须保持复位值。

位 11:0 **WIN[11:0]**：看门狗计数器窗口值 (Watchdog counter window value)
这些位受写访问保护，请参见*第 30.4.6 节*。它们包含用于与递减计数器进行比较的窗口值上限。
为防止发生复位，当 IWDCNT 递减计数器的值小于 WIN[11:0] + 1 且大于 1 时必须重载。
若要更改重载值，*IWDG 状态寄存器 (IWDG_SR)* 中的 WVU 位必须复位。
注意：读取该寄存器会返回 V_{DD} 电压域的重载值。如果正在对该寄存器执行写操作，则读取的值可能无效。因此，只有在 IWDG 状态寄存器 (IWDG_SR) 中的 WVU 位复位时，从寄存器读取的值才有效。

30.7.6 IWDG 提前唤醒中断寄存器 (IWDG_EWCR)

IWDG early wake-up interrupt register

偏移地址：0x14

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15 **EWIE**：看门狗提前中断使能 (Watchdog early interrupt enable)

由软件置 1 和复位。

0：禁止提前中断接口。

1：使能提前中断接口。

若要更改该位的值，*IWDG 状态寄存器 (IWDG_SR)* 中的 EWU 位必须复位。

位 14 **EWIC**：看门狗提前中断确认 (Watchdog early interrupt acknowledge)

必须由软件向该位写入 1，以确认提前唤醒中断并将 EWIF 标志清零。写入 0 无影响，读取该标志将返回 0。

位 13:12 保留，必须保持复位值。

位 11:0 **EWIT[11:0]**：看门狗计数器窗口值 (Watchdog counter window value)

这些位受写访问保护（请参见*第 30.4.6 节*）。通过软件写入这些位，可以定义在 IWDCNT 递减计数器计数到哪个位置时必须生成提前唤醒中断。当 IWDCNT 小于或等于 EWIT[11:0] - 1 时，生成提前中断。

EWIT[11:0] 必须大于 1。

只有在 EWIE = 1 时才会生成中断。

若要更改重载值，*IWDG 状态寄存器 (IWDG_SR)* 中的 EWU 位必须复位。

注意：读取该寄存器将返回 V_{DD} 电压域的提前唤醒比较器值和中断使能位值。如果正在对该寄存器执行写操作，则读取的值可能不是最新的/有效的。因此，只有在 IWDG 状态寄存器 (IWDG_SR) 中的 EWU 位复位时，从寄存器读取的值才有效。

30.7.7 IWDG 寄存器映射

表 226. IWDG 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	IWDG_KR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	KEY[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	IWDG_PR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR[3:0]			
	Reset value																													0	0	0	0
0x08	IWDG_RLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RL[11:0]											
	Reset value																					1	1	1	1	1	1	1	1	1	1	1	1
0x0C	IWDG_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWIF	Res.	Res.	Res.	Res.	Res.	ONF	Res.	Res.	Res.	Res.	EWU	WVU	RVU	PVU
	Reset value																		0						x					0	0	0	0
0x10	IWDG_WINR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WIN[11:0]											
	Reset value																					1	1	1	1	1	1	1	1	1	1	1	1
0x14	IWDG_EWCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWIE	EWIC	Res.	Res.	EWIT[11:0]											
	Reset value																	0	0			0	0	0	0	0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

31 系统窗口看门狗 (WWDG)

31.1 简介

系统窗口看门狗 (WWDG) 通常用来监测特定软件故障，这些软件故障来源于外部干扰或错误的逻辑条件造成的应用程序运行序列异常。

除非程序在 T6 位清零前刷新递减计数器的值，否则看门狗电路在达到预置的时间周期时，会产生复位。如果在递减计数器达到窗口寄存器值之前刷新控制寄存器中的 7 位递减计数器值，看门狗电路也会产生复位。这意味着必须在限定的时间窗口内刷新计数器。

WWDG 时钟由 APB 时钟经预分频后提供，通过可配置的时间窗口来检测应用程序非正常的过迟或过早的操作。

WWDG 最适合那些要求看门狗在精确计时窗口起作用的应用。

31.2 WWDG 主要特性

- 可编程的自由运行递减计数器
- 复位条件
  - 当递减计数器值小于 0x40 时复位（如果看门狗已激活）
  - 在窗口之外重载递减计数器时复位（如果看门狗已激活）（请参见*图 367*）
- 提前唤醒中断 (EWI)：当递减计数器等于 0x40 时触发（如果已使能且看门狗已激活）

31.3 WWDG 实现

表 227. WWDG 特性(1)

WWDG 模式/特性	WWDG
窗口功能	X
提前唤醒中断生成	X
系统复位生成(2)	X
可在系统停止模式下工作	-
可在系统待机模式下工作	-
可在微控制器进入调试模式时被冻结(3)	X
用于控制硬件模式的选项字节(4)	X

1. “X” = 支持，“-” = 不支持
2. 有关详细信息，请参见 RCC 部分。
3. 通过 DBG 模块中的 DBG_WWDG_STOP 进行控制。
4. 通过选项字节 WWDG_SW 进行控制。当 WWDG_SW 设置为 HW 模式时，只要 CPU 处于运行模式或睡眠模式，WWDG 就会运行。

31.4 WWDG 功能描述

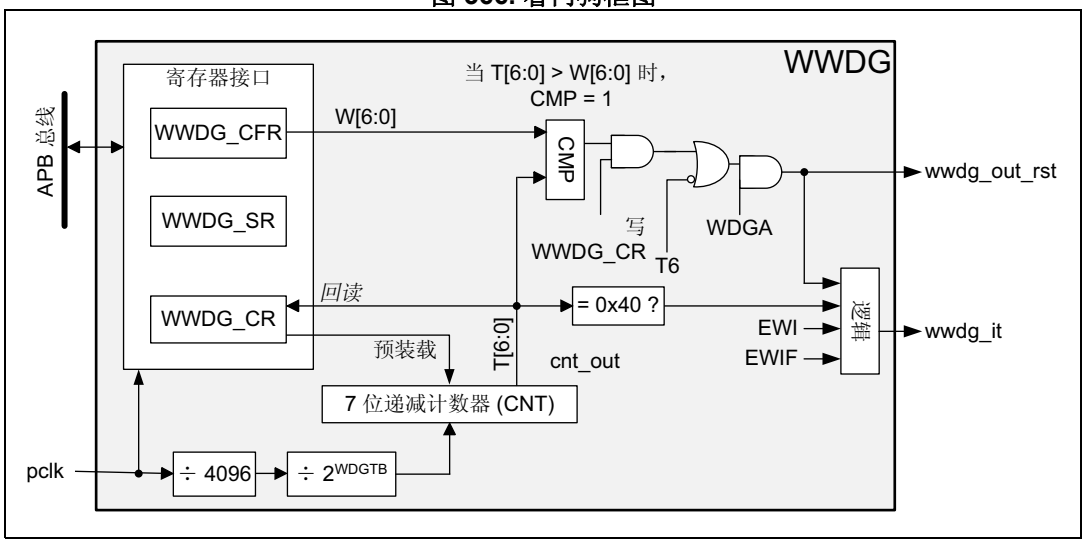
如果激活看门狗（WWDG_CR 寄存器中的 WDGA 位置 1），则当 7 位递减计数器（T[6:0] 位）从 0x40 递减到 0x3F（T6 已清零）时会引发复位。当计数器值大于窗口寄存器中所存储的值时，如果软件重载计数器，则会产生复位。

应用程序在正常运行过程中必须定期写入 WWDG_CR 寄存器以防止复位。只有当计数器值低于窗口寄存器值且高于 0x3F 时，才能执行此操作。存储在 WWDG_CR 寄存器中的值必须介于 0xFF 和 0xC0 之间。

请参见*图 366* 了解 WWDG 框图。

31.4.1 WWDG 框图

图 366. 看门狗框图



31.4.2 WWDG 内部信号

表 228 给出了 WWDG 内部信号列表。

表 228. WWDG 内部输入/输出信号

信号名称	信号类型	说明
pclk	数字输入	APB 总线时钟
wwdg_out_rst	数字输出	WWDG 复位信号输出
wwdg_it	数字输出	WWDG 提前中断输出

31.4.3 使能看门狗

当用户选项 WWDG_SW 选择“软件窗口看门狗”时，看门狗在复位后总处于关闭状态。可通过设置 WWDG_CR 寄存器中的 WDGA 位来使能看门狗，之后除非执行复位操作，否则不能再次关闭。

当用户选项 WWDG_SW 选择“硬件窗口看门狗”时，看门狗在复位后始终使能，无法禁止。

31.4.4 控制递减计数器

递减计数器处于自由运行状态，即使禁止看门狗，递减计数器仍继续递减计数。当使能看门狗时，必须将 T6 位置 1，以防止立即复位。

T[5:0] 位包含了看门狗产生复位之前的计时数目；复位前的延时时间在一个最小值和一个最大值之间变化，这是因为写入 WWDG_CR 寄存器时，预分频器的状态是未知的（请参见*图 367*）。*WWDG 配置寄存器 (WWDG_CFR)* 包含窗口的上限：为防止发生复位，当递减计数器的值低于或等于窗口寄存器值且大于 0x3F 时必须重载。*图 367* 介绍了窗口看门狗的工作过程。

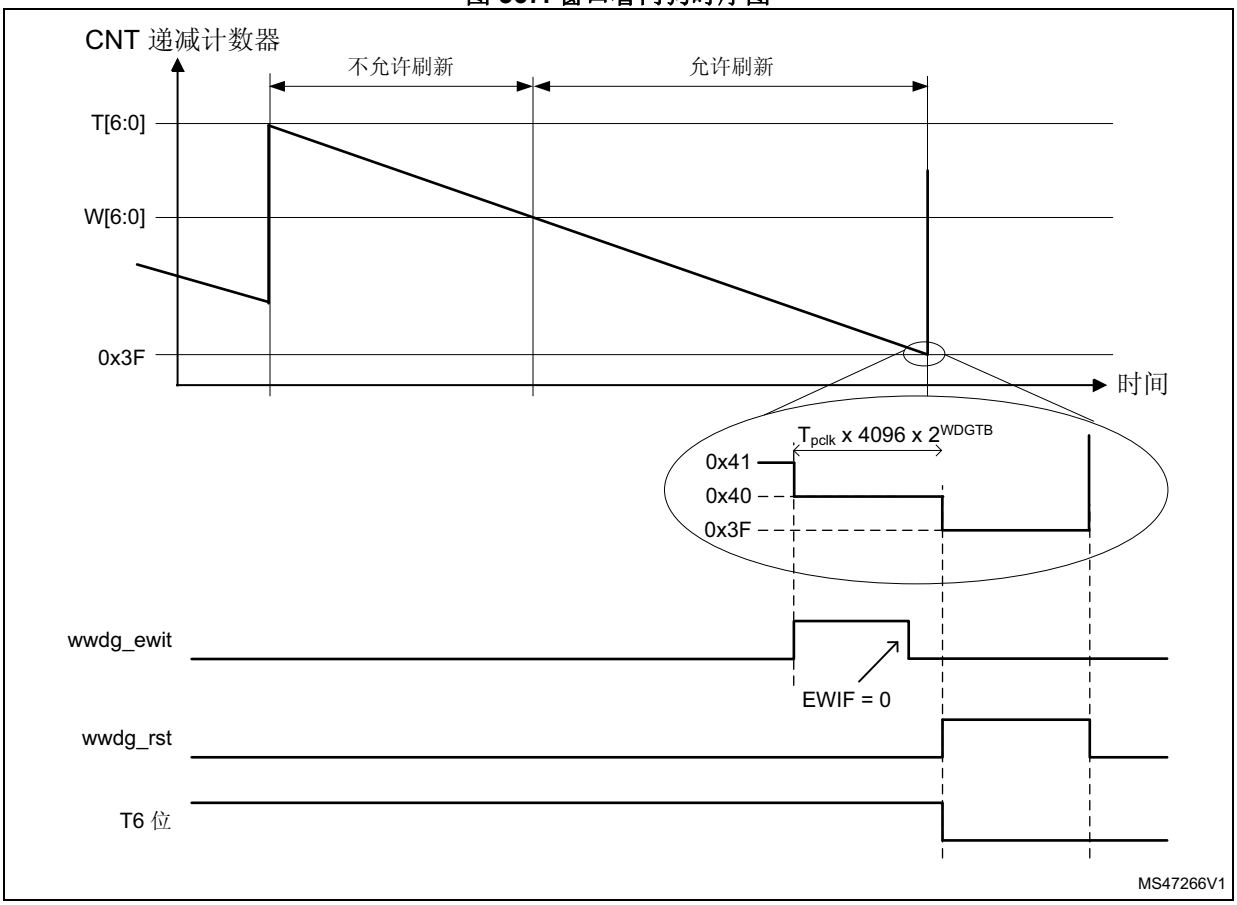
注意： *可使用 T6 位产生软件复位（将 WDGA 位置 1 并将 T6 位清零）。*

31.4.5 如何设置看门狗超时

使用*图 367* 中的公式来计算 WWDG 超时。

警告： **写入 WWDG_CR 寄存器时，始终将 1 写入 T6 位，以避免生成立即复位。**

图 367. 窗口看门狗时序图



超时值的计算公式如下：

$$t_{WWDG} = t_{PCLK} \times 4096 \times 2^{WDGTB[2:0]} \times (T[5:0] + 1) \quad (ms)$$

其中：

t_{WWDG}：WWDG 超时

t_{PCLK}：APB 时钟周期，以 ms 为测量单位

4096：对应于内部分频器的值

例如，如果 APB 频率为 48 MHz，将 WDGTB[2:0] 设置为 3 并将 T[5:0] 设置为 63：

$$t_{WWDG} = (1/48000) \times 4096 \times 2^{3} \times (63 + 1) = 43.69ms$$

有关 t_{WWDG} 的最小值和最大值，请参见数据手册。

31.4.6 调试模式

当器件进入调试模式时（内核停止），WWDG 计数器会根据 DBG 模块中的配置位选择继续正常工作或者停止工作。有关详细信息，请参见。

31.5 WWDG 中断

如果在产生复位之前必须执行特定的安全操作或数据记录，则可使用提前唤醒中断 (EWI)。要使能提前唤醒中断，应用程序必须：

- 向 WWDG_SR 寄存器的 EWIF 位写入 0，以清除意外的挂起中断
- 向 WWDG_CFR 寄存器的 EWI 位写入 1，以使能中断

当递减计数器的值为 0x40 时，将生成看门狗中断。在复位器件之前，可以使用相应的中断服务程序 (ISR) 来触发特定操作（例如通信或数据记录）。

在某些应用中，可以使用 EWI 中断来管理软件系统检查和/或系统恢复/功能退化，而不会生成 WWDG 复位。在这种情况下，相应的 ISR 必须重载 WWDG 计数器以避免 WWDG 复位，然后再触发所需操作。

通过将 0 写入 WWDG_SR 寄存器中的 EWIF 位来清除看门狗中断。

注意： *当由于在更高优先级任务中有系统锁定等原因而无法使用看门狗中断时，最终会产生 WWDG 复位。*

表 229. WWDG 中断请求

中断		事件标志	使能控制位	中断清除方法	退出以下模式		
向量	事件				睡眠	停止(1)	待机(1)
WWDG(2)	提前唤醒中断	EWIF	EWI	向 EWIF 标志写入 0	是	否	否

1. WWDG 中断可能具有附加功能，有关详细信息，请参见*第 31.3 节*。
2. WWDG 向量与 wwdg_it 信号置为有效相对应。

31.6 WWDG 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节*。

外设寄存器可按半字（16 位）或字（32 位）访问。

31.6.1 WWDG 控制寄存器 (WWDG_CR)

WWDG control register

偏移地址：0x000

复位值：0x0000 007F

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

位 31:8 保留，必须保持复位值。

位 7 **WDGA**：激活位 (Activation bit)

该位由软件置 1，只有复位后才由硬件清零。当 WDGA = 1 时，看门狗可产生复位。

0：禁止看门狗

1：使能看门狗

位 6:0 **T[6:0]**：7 位计数器（MSB 到 LSB）

这些位用来存储看门狗计数器的值，每隔 (4096 x $2^{WDGTB[2:0]}$) PCLK 个周期递减一次。当它从 0x40 递减到 0x3F（T6 清零）时会产生复位。

31.6.2 WWDG 配置寄存器 (WWDG_CFR)

WWDG configuration register

偏移地址：0x004

复位值：0x0000 007F

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

位 31:14 保留，必须保持复位值。

位 13:11 **WDGTB[2:0]**：定时器时基 (Timer base)

可按如下方式修改预分频器的时基：

000：CK 计数器时钟 (PCLK div 4096) 分频器 1

001：CK 计数器时钟 (PCLK div 4096) 分频器 2

010：CK 计数器时钟 (PCLK div 4096) 分频器 4

011：CK 计数器时钟 (PCLK div 4096) 分频器 8

100：CK 计数器时钟 (PCLK div 4096) 分频器 16

101：CK 计数器时钟 (PCLK div 4096) 分频器 32

110：CK 计数器时钟 (PCLK div 4096) 分频器 64

111：CK 计数器时钟 (PCLK div 4096) 分频器 128

位 10 保留，必须保持复位值。

位 9 **EWI**：提前唤醒中断使能 (Early wake-up interrupt enable)

由软件置 1，只有复位后才由硬件清零。置 1 后，只要计数器值达到 0x40 就会产生中断。

位 8:7 保留，必须保持复位值。

位 6:0 **W[6:0]**：7 位窗口值 (7-bit window value)

这些位包含用于与递减计数器进行比较的窗口值。

31.6.3 WWDG 状态寄存器 (WWDG_SR)

WWDG status register

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWIF
															rc_w0

位 31:1 保留，必须保持复位值。

位 0 **EWIF**：提前唤醒中断标志 (Early wake-up interrupt flag)

当计数器值达到 0x40 时，该位由硬件置 1。它必须由软件通过写入 0 来清零。写入 1 无效。如果不使能中断，该位也会被置 1。

31.6.4 WWDG 寄存器映射

下表提供了 WWDG 寄存器映射和复位值。

表 230. WWDG 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	WWDG_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDGA	T[6:0]						
	Reset value																									0	1	1	1	1	1	1	1
0x004	WWDG_CFR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDGTB [2:0]			Res.	EWI	Res.	Res.	W[6:0]						
	Reset value																			0	0	0		0			1	1	1	1	1	1	1
0x008	WWDG_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWIF
	Reset value																																0

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

32 实时时钟 (RTC)

32.1 简介

实时时钟 (RTC) 提供用于管理所有低功耗模式的自动唤醒单元。

实时时钟 (RTC) 是一个独立的 BCD 定时器/计数器。RTC 提供具有可编程闹钟中断功能的日历时钟/日历。

无论器件状态如何（运行模式、低功耗模式或处于复位状态），只要电源电压保持在工作范围内，RTC 便不会停止工作。

RTC 在 V_{BAT} 模式下正常工作。

32.2 RTC 主要特性

RTC 支持以下特性（请参见*图 368：RTC 框图*）：

- 日历具有亚秒、秒、分、小时（12 或 24 格式）、星期几、日、月、年，格式为 BCD（二进码十进数）。
- 可实现 32 位自由运行计数器的二进制模式。
- 自动调整每月是 28、29（闰年）、30 还是 31 天。
- 两个可编程闹钟。
- 可运行时纠正 1 到 32767 个 RTC 时钟脉冲。这可用于与主时钟同步。
- 参考时钟检测：可使用更加精确的第二时钟源（50 或 60 Hz）来提高日历的精确度。
- 数字校准电路具有 0.95 ppm 的分辨率，以补偿石英晶振的不准确性。
- 时间戳特性可用于保存日历内容。 该功能可由时间戳引脚上的事件触发，或由入侵事件触发，也可以通过切换到 V_{BAT} 模式来触发。
- 17 位自动重载唤醒定时器 (WUT)，适用于分辨率和周期可编程的周期性事件。
- 闹钟 A、闹钟 B、唤醒定时器和时间戳各自具有独立的特权保护

RTC 通过开关供电，当 V_{DD} 电源存在时，该开关选择 V_{DD} 供电，否则选择由 VBAT 引脚供电。

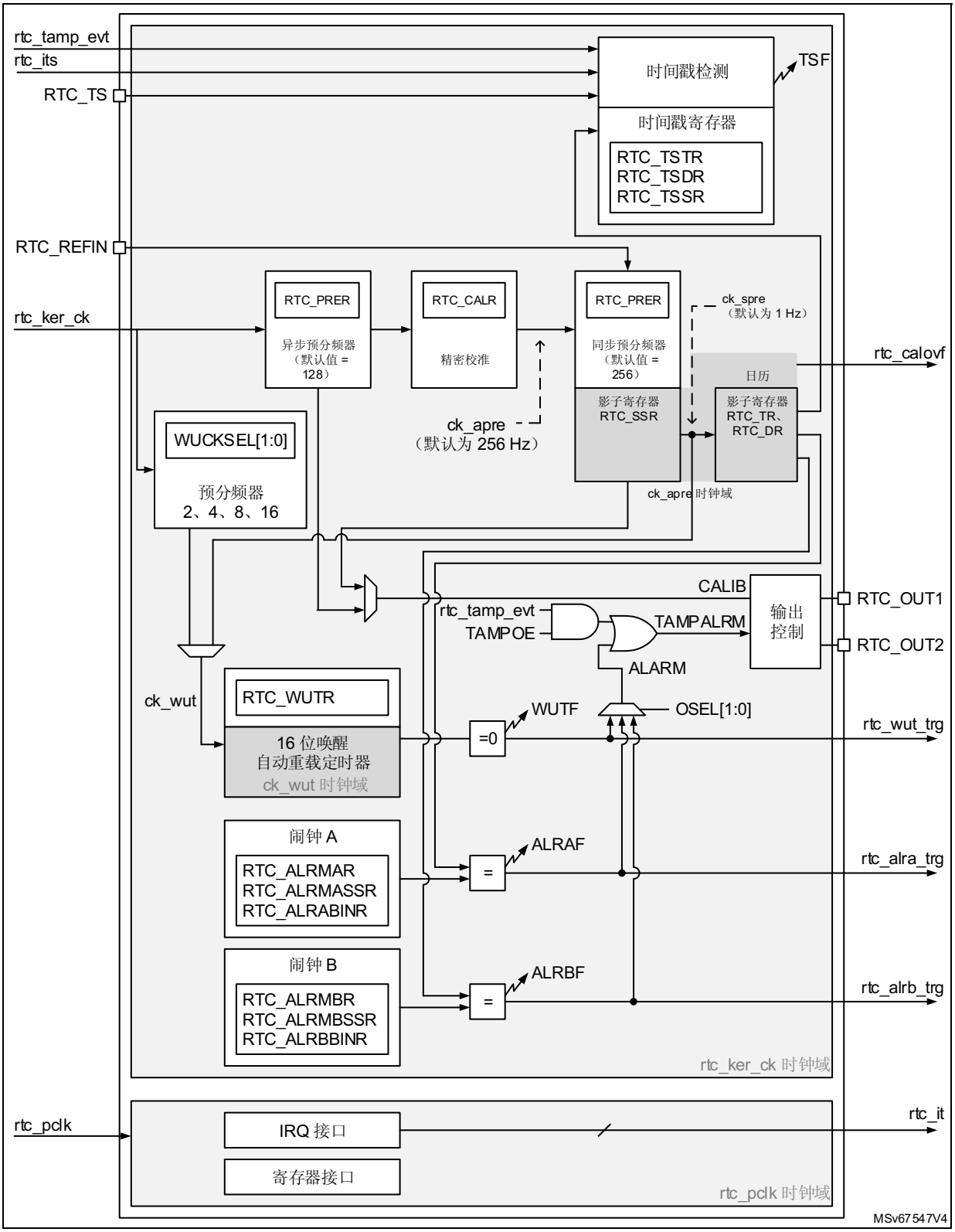
当 RTC 的时钟由 LSE 提供时，RTC 可以在 V_{BAT} 模式和所有低功耗模式下正常工作。

所有 RTC 事件（闹钟、唤醒定时器、时间戳）都可以产生中断，将器件从低功耗模式唤醒。

32.3 RTC 功能描述

32.3.1 RTC 框图

图 368. RTC 框图


rtc_tamp_evt
rtc_its
RTC_TS
时间戳检测
TSF
时间戳寄存器
RTC_TSTR
RTC_TSDR
RTC_TSSR
RTC_REFIN
rtc_ker_ck
RTC_PRER
异步预分频器
（默认值 = 128）
RTC_CALR
精密校准
RTC_PRER
同步预分频器
（默认值 = 256）
ck_spre
（默认为 1 Hz）
日历
rtc_calovf
影子寄存器
RTC_SSR
影子寄存器
RTC_TR、
RTC_DR
ck_apre
（默认为 256 Hz）
WUCKSEL[1:0]
预分频器
2、4、8、16
ck_apre 时钟域
CALIB
rtc_tamp_evt
TAMPOE
TAMPALRM
输出
控制
RTC_OUT1
RTC_OUT2
ALARM
ck_wut
RTC_WUTR
16 位唤醒
自动重载定时器
ck_wut 时钟域
=0
WUTF
OSEL[1:0]
rtc_wut_trg
闹钟 A
RTC_ALRMAR
RTC_ALRMASSR
RTC_ALRABINR
=
ALRAF
rtc_alra_trg
闹钟 B
RTC_ALRMBR
RTC_ALRMBSSR
RTC_ALRBBINR
=
ALRBF
rtc_alrb_trg
rtc_ker_ck 时钟域
rtc_pclk
IRQ 接口
寄存器接口
rtc_it
rtc_pclk 时钟域
MSv67547V4

32.3.2 RTC 引脚和内部信号

表 231. RTC 输入/输出引脚

引脚名称	信号类型	说明
RTC_TS	输入	RTC 时间戳输入
RTC_REFIN	输入	RTC 50 Hz 或 60 Hz 参考时钟输入
RTC_OUT1	输出	RTC 输出 1
RTC_OUT2	输出	RTC 输出 2

RTC_OUT1 和 RTC_OUT2 可选择以下两个输出之一：

- CALIB：512 Hz 或 1 Hz 时钟输出（LSE 频率为 32.768 kHz）。可通过将 RTC_CR 寄存器中的 COE 位置 1 来使能此输出。
- TAMPALRM：该输出是 rtc_tamp_evt 与 ALARM 信号的逻辑或运算结果。

ALARM 通过配置 RTC_CR 寄存器中的 OSEL[1:0] 位（可选择闹钟 A、闹钟 B 或唤醒输出）来使能。rtc_tamp_evt 通过将 RTC_CR 寄存器中的 TAMPOE 位置 1（将选择入侵事件输出）来使能。

表 232. RTC 内部输入/输出信号

内部信号名称	信号类型	说明
rtc_ker_ck	输入	RTC 内核时钟，在本文档中也称为 RTCCLK
rtc_pclk	输入	RTC APB 时钟
rtc_its	输入	RTC 内部时间戳事件
rtc_tamp_evt	输入	在 TAMP 外设中检测到的入侵事件（内部或外部）
rtc_it	输出	RTC 中断（有关详细信息，请参见*第 32.5 节：RTC 中断*）
rtc_alra_trg	输出	RTC 闹钟 A 事件检测触发
rtc_alrb_trg	输出	RTC 闹钟 B 事件检测触发
rtc_wut_trg	输出	RTC 唤醒定时器事件检测触发
rtc_calovf	输出	RTC 日历上溢：当 RTC 日历达到其最大值时（99 年 12 月 31 日 23:59:59），会生成该信号。随后，日历将被冻结，不会上溢。

RTC 内核时钟通常是 32.768 kHz 的 LSE，但也可以在 RCC 中选择其他时钟源（更多详细信息，请参见 RCC）。当所选时钟不是 LSE 时，一些功能在某些低功耗模式或 V_{BAT} 模式下不可用。更多详细信息，请参见*第 32.4 节：RTC 低功耗模式*。

表 233. RTC 互连

信号名称	源/目标
rtc_its	从电源控制器 (PWR)：主电源丢失/切换到 V_{BAT} 检测输出
rtc_tamp_evt	从 TAMP 外设：tamp_evt
rtc_calovf	到 TAMP 外设：tamp_itamp5

触发输出可以用作其他外设的触发信号。

32.3.3 RTC 和 TAMP 控制的 GPIO

电池备份域 (V_{BAT}) 中包含的 GPIO 由使用这些 I/O 提供功能的外设直接控制（无论 GPIO 的配置如何）。

RTC_OUT1、RTC_TS、TAMP_IN1 和 TAMP_OUT2 映射到同一引脚 (PC13)。映射在 PC13 上的 RTC 和 TAMP 功能可用于所有低功耗模式和 V_{BAT} 模式。

输出机制遵循*表 234* 中所示的优先级顺序。

表 234. RTC 引脚 PC13 配置(1)

PC13 引脚功能	OSEL[1:0]（ALARM 输出使能）	TAMPOE（TAMPER 输出使能）	COE（CALIB 输出使能）	OUT2EN	TAMPALRM_TYPE	TAMPALRM_PU	TAMP2E=TAMP2AM=1 (ATOSHARE=0) 或 TAMPxE=TAMPxAM=1（ATOSHARE=1 且 ATOSELx=1）	TAMP1E（TAMP_IN1 输入使能）	TSE（RTC_TS 输入使能）
TAMPALRM 输出推挽	01 或 10 或 11	0	无关	无关	0	0	无关	无关	无关
	00	1							
	01 或 10 或 11	1							

表 234. RTC 引脚 PC13 配置(1)（续）

PC13 引脚功能		OSEL[1:0]（ALARM 输出使能）	TAMPOE（TAMPER 输出使能）	COE（CALIB 输出使能）	OUT2EN	TAMPALRM_TYPE	TAMPALRM_PU	TAMP2E=TAMP2AM=1 (ATOSHARE=0) 或 TAMPxE=TAMPxAM=1（ATOSHARE=1 且 ATOSELx=1）	TAMP1E（TAMP_IN1 输入使能）	TSE（RTC_TS 输入使能）
TAMPALRM 输出开漏(2)	无上下拉	01 或 10 或 11	0	无关	无关	1	0	无关	无关	无关
		00	1							
		01 或 10 或 11	1							
	内部上拉	01 或 10 或 11	0	无关	无关	1	1	无关	无关	无关
		00	1							
		01 或 10 或 11	1							
CALIB 输出 PP		00	0	1	0	无关	无关	无关	无关	无关
TAMP_OUT2 输出 PP		00	0	0	0	无关	无关	1	无关	无关
TAMP_IN1 输入浮空		00	0	0	无关	无关	无关	0	1	0
		00	0	1	1					
		无关	无关	0						
RTC_TS 和 TAMP_IN1 输入浮空		00	0	0	无关	无关	无关	0	1	1
		00	0	1	1			0		
		无关	无关	0				0		
RTC_TS 输入浮空		00	0	0	无关	无关	无关	0	0	1
		00	0	1	1			0		
		无关	无关	0				0		

表 234. RTC 引脚 PC13 配置(1)（续）

PC13 引脚功能	OSEL[1:0]（ALARM 输出使能）	TAMPOE（TAMPER 输出使能）	COE（CALIB 输出使能）	OUT2EN	TAMPALRM_TYPE	TAMPALRM_PU	TAMP2E=TAMP2AM=1 (ATOSHARE=0) 或 TAMPxE=TAMPxAM=1（ATOSHARE=1 且 ATOSELx=1）	TAMP1E（TAMP_IN1 输入使能）	TSE（RTC_TS 输入使能）
唤醒引脚	00	0	0	无关	无关	无关	0	0	0
	00	0	1	1			0		
	无关	无关	0				0		
标准 GPIO	00	0	0	无关	无关	无关	0	0	0
	00	0	1	1			0		
	无关	无关	0				0		

1. OD：开漏；PP：推挽。
2. 在此配置下，GPIO 必须配置为输入。

此外，可借助 OUT2EN 位在 PA0 或 PB2 引脚上输出 RTC_OUT2。可以将不同的功能映射到 RTC_OUT1 或 RTC_OUT2 上，具体取决于 OSEL、COE 和 OUT2EN 的配置，如*表 235* 所示。

表 235. RTC_OUT 映射

OSEL[1:0] 位（ALARM 输出使能）	COE 位（CALIB 输出使能）	OUT2EN 位	RTC_OUT1 映射到 PC13	RTC_OUT2 映射到 PA0 或 PB2
00	0	0	-	-
00	1		CALIB	-
01 或 10 或 11	无关		TAMPALRM	-
00	0	1	-	-
00	1		-	CALIB
01 或 10 或 11	0		-	TAMPALRM
01 或 10 或 11	1		TAMPALRM	CALIB

32.3.4 RTC 特权保护模式

默认情况下，备份域上电复位后，除了 RTC 特权模式控制寄存器 (RTC_PRIVCFGR) 只能在特权模式下写入之外，所有其他 RTC 寄存器都可以在特权模式和非特权模式下读取或写入。RTC 保护配置不受系统复位影响。

当 RTC_PRIVCFGR 寄存器中的 PRIV 位置 1 时：

- 只能在特权模式下写入 RTC 寄存器。
- 始终可以在特权模式和非特权模式下读取 RTC_PRIVCFGR、RTC_TR、RTC_DR、RTC_SSR、RTC_PRER 和 RTC_CALR。所有其他 RTC 寄存器都只能在特权模式下读取。

当 PRIV 位清零时，仍然可以通过将专用控制位 INITPRIV、CALPRIV、TSPRIV、WUTPRIV、ALRAPRV 或 ALRBPRIV 置 1 来保护某些寄存器。如果所有这些位也都清零，则所有 RTC 寄存器都可以在特权模式和非特权模式下读取或写入。

- 当 INITPRIV 置 1 时：
  - RTC_TR、RTC_DR、RTC_PRER 寄存器，RTC_ICSR 寄存器中的 INIT、BIN 和 BCDU，以及 RTC_CR 寄存器中的 FMT 控制位只能在特权模式下写入。
  - 这些寄存器和控制位可以在特权模式和非特权模式下读取。
- 当 CALPRIV 置 1 时：
  - RTC_SHIFTR 和 RTC_CALR 寄存器，以及 RTC_CR 寄存器中的 ADD1H、SUB1H 和 REFCKON 控制位只能在特权模式下写入。
  - 这些寄存器和控制位可以在特权模式和非特权模式下读取。
- 当 ALRAPRIV 置 1 时：
  - RTC_ALRMAR、RTC_ALRMASSR 和 RTC_ALRABINR 寄存器，RTC_CR 寄存器中的 ALRAE ALRAFCLR、ALRAIE 和 SSRUIE，RTC_SCR 寄存器中的 CALRAF 和 CSSRUF，RTC_SR 寄存器中的 ALRAF 和 SSRUF，以及 RTC_MISR 寄存器中的 ALRAMF 和 SSRUMF 只能在特权模式下读取和写入。
- 当 ALRBPRIV 置 1 时：
  - RTC_ALRMBR、RTC_ALRMBSSR 和 RTC_ALRBBINR 寄存器，RTC_CR 寄存器中的 ALRBE ALRBFCLR 和 ALRBIE，RTC_SCR 寄存器中的 CALRBF，RTC_SR 寄存器中的 ALRBF 以及 RTC_MISR 寄存器中的 ALRBMF 只能在特权模式下读取和写入。
- 当 WUTPRIV 置 1 时：
  - RTC_WUTR 寄存器，RTC_CR 寄存器中的 WUTE、WUTIE 和 WUCKSEL 控制位，RTC_SCR 寄存器中的 CWUTF、RTC_SR 寄存器中的 WUTF 以及 RTC_MISR 寄存器中的 WUTMF 只能在特权模式下读取和写入。
- 当 TSPRIV 置 1 时：
  - RTC_TSTR 和 RTC_TSDR 和 RTC_TSSSR 寄存器，RTC_CR 寄存器中的 TAMPTS、ITSE、TSE、TSIE、TSEDGE 控制位，RTC_SCR 寄存器中的 CITSF、CTSOVF 和 CTSF，RTC_SR 寄存器中的 TSF、TSOVF 和 ITSF，以及 RTC_MISR 寄存器中的 TSMF、TSOVMF 和 ITSMF 只能在特权模式下读取和写入。

对受特权保护的寄存器的非特权访问将被拒绝：

- 不生成任何总线错误。
- 受写保护时，不会写入这些位。
- 受读保护时，这些位读为 0。

32.3.5 时钟和预分频器

有关 RTC 时钟 (RTCCLK) 源配置的更多信息，请参见“复位和时钟控制 (RCC)”。

BCD 模式 (BIN=00)

可编程的预分频器阶段可生成 1 Hz 的时钟，用于更新日历。为最大程度地降低功耗，预分频器分为 2 个可编程的预分频器（参见*图 368：RTC 框图*）：

- 一个通过 RTC_PRER 寄存器的 PREDIV_A 位配置的 7 位异步预分频器。
- 一个通过 RTC_PRER 寄存器的 PREDIV_S 位配置的 15 位同步预分频器。

注意： *使用两个预分频器时，推荐将异步预分频器配置为较高的值，以最大程度降低功耗。*

要使用频率为 32.768 kHz 的 LSE 获得频率为 1 Hz 的内部时钟 (ck_spre)，需要将异步预分频系数设置为 128，并将同步预分频系数设置为 256。

分频系数的最小值为 1，最大值为 2^{22}。

这对应于约为 4 MHz 的最大输入频率。

f_{ck_apre} 可根据以下公式得出：

$$f_{CK_APRE} = \frac{f_{RTCCLK}}{PREDIV_A + 1}$$

ck_apre 时钟用于为二进制 RTC_SSR 亚秒递减计数器提供时钟。当该计数器计数到 0 时，会使用 PREDIV_S 的内容重载 RTC_SSR。

f_{ck_spre} 可根据以下公式得出：

$$f_{CK_SPRE} = \frac{f_{RTCCLK}}{(PREDIV_S + 1) \times (PREDIV_A + 1)}$$

ck_spre 时钟既可以用于更新日历，也可以用作 16 位唤醒自动重载定时器的时基。为获得较短的超时周期，还可以将 16 位唤醒自动重载定时器与经可编程的 4 位异步预分频器分频的 RTCCLK 一同运行（有关详细信息，请参见*第 32.3.9 节：周期性自动唤醒*）。

二进制模式 (BIN=01)

SSR 二进制递减计数器扩展到 32 位长度并自由运行。时间和日期日历 BCD 寄存器不可用。

该递减计数器的时钟由 ck_apre 提供：通过 RTC_PRER 寄存器的 PREDIV_A 位配置的 7 位异步预分频器的输出。

PREDIV_S 值无意义。

混合模式（BIN=10 或 11）

SSR 二进制递减计数器扩展到 32 位长度并自由运行。时间和日期日历 BCD 寄存器也可用。

该递减计数器的时钟由 ck_apre 提供：通过 RTC_PRER 寄存器的 PREDIV_A 位配置的 7 位异步预分频器的输出。位 BCDU[2:0] 用于定义日历何时使用 SSR 最低有效位递增 1 秒。

32.3.6 实时时钟和日历

RTC 日历时间和日期寄存器可通过与 PCLK（APB 时钟）同步的影子寄存器来访问。这些时间和日期寄存器也可以直接访问，无同步时间。

- RTC_SSR 对应于亚秒
- RTC_TR 对应于时间
- RTC_DR 对应于日期

每个 RTCCLK 周期，都将当前日历值复制到影子寄存器，并将 RTC_ICSR 寄存器的 RSF 位置 1（请参见*第 32.6.11 节：RTC 平移控制寄存器 (RTC_SHIFTR)*）。在停止和待机模式下不会执行复制操作。退出这两种模式时，影子寄存器会在最长 4 个 RTCCLK 周期后进行更新。

当应用读取日历寄存器时，它会访问影子寄存器的内容。也可以通过将 RTC_CR 寄存器的 BYPSHAD 控制位置 1 来直接访问日历寄存器。默认情况下，该位被清零，用户访问影子寄存器。

在 BYPSHAD=0 模式下读取 RTC_SSR、RTC_TR 或 RTC_DR 寄存器时，APB 时钟频率 (f_{APB}) 必须至少为 RTC 时钟频率 (f_{RTCCLK}) 的 7 倍。

影子寄存器通过系统复位来复位。

32.3.7 日历超低功耗模式

将 RTC_CALR 寄存器中的 LPCAL 位置 1，可以大幅降低 RTC 功耗。在这种配置下，整个 RTC 仅由 ck_apre 提供时钟，而不是同时由 RTCCLK 和 ck_apre 提供时钟。因此，一些标志的延迟较长，校准窗口也较长（请参见*RTC 超低功耗模式*一节）。

当异步预分频器的分频系数 (PREDIV_A+1) 不是 2 的幂数时，会忽略 LPCAL 位（假定为 0）。

从 LPCAL=0 切换到 LPCAL=1 或从 LPCAL=1 切换到 LPCAL=0 的操作不会立即生效，需要几个 ck_apre 周期才能完成。

32.3.8 可编程闹钟

RTC 单元提供两个可编程闹钟，即闹钟 A 和闹钟 B。以下说明针对闹钟 A，但同样适用于闹钟 B。

可通过 RTC_CR 寄存器中的 ALRAE 位来使能可编程闹钟功能。

如果日历亚秒、秒、分钟、小时、日期或日分别与闹钟寄存器 RTC_ALRMASSR 和 RTC_ALRMAR 中编程的值相匹配，则 ALRAF 标志会被置为 1。可通过 RTC_ALRMAR 寄存器的 MSKx 位以及 RTC_ALRMASSR 寄存器的 MASKSSx 位单独选择各日历字段。

使用二进制模式时，可以在闹钟二进制寄存器 RTC_ALRABINR 中编程亚秒字段。

可通过 RTC_CR 寄存器中的 ALRAIE 位来使能闹钟中断。

如果闹钟用于为其他外设生成触发事件，则可通过将 RTC_CR 寄存器中的 ALRAFCLR 位置 1 使 ALRAF 由硬件自动清零。在这种配置下，如果只需将 ALRAF 标志清零，则无需任何软件干预。

小心： 如果选择秒字段（RTC_ALRMAR 中的 MSK1 位复位），则 RTC_PRER 寄存器中设置的同步预分频器分频系数必须至少为 3，才能确保闹钟正确地运行。

闹钟 A 和闹钟 B（如果已通过 RTC_CR 寄存器中的位 OSEL[0:1] 使能）可连接到 TAMPALRM 输出。可通过 RTC_CR 寄存器的 POL 位配置 TAMPALRM 输出极性。

32.3.9 周期性自动唤醒

周期性唤醒标志由 16 位可编程自动重载递减计数器生成。唤醒定时器范围可扩展至 17 位。

可通过 RTC_CR 寄存器中的 WUTE 位来使能此唤醒功能。

唤醒定时器时钟输入 ck_wut 可以是：

- 2、4、8 或 16 分频的 RTC 时钟 (RTCCLK)。

  当 RTCCLK 为 LSE (32.768 kHz) 时，可配置的唤醒中断周期介于 122 µs 和 32 s 之间，且分辨率低至 61 µs。
- BCD 模式下的 ck_spre（通常为 1 Hz 内部时钟），或用于在二进制或混合（BCD 二进制）模式下根据 BCDU 的定义更新日历的时钟。

  当 ck_spre 频率为 1 Hz 时，可以实现的唤醒时间为 1s 到 36h 左右，分辨率为 1 秒。这一较大的可编程时间范围分为两部分：
  - WUCKSEL [2:1] = 10 时为 1s 到 18h
  - WUCKSEL [2:1] = 11 时约为 18h 到 36h。在后一种情况下，会将 2^{16} 加到 16 位计数器当前值中。当初始化序列完成之后（请参见*第 1016 页的编程唤醒定时器*），定时器开始递减计数。在低功耗模式下使能唤醒功能时，递减计数保持有效。此外，当计数器计数到 0 时，RTC_SR 寄存器的 WUTF 标志会置 1，并且唤醒计数器会使用其重载值（RTC_WUTR 寄存器值）自动重载。

根据 RTC_WUTR 寄存器中的 WUTOCLR，WUTF 标志必须由软件清零 (WUTOCLR = 0x0000)，或者在自动重载递减计数器计数到 WUTOCLR 值 (0x0000 < WUTOCLR ≤ WUT) 时由硬件自动清零。

唤醒标志基于内部信号 rtc_wut 输出，该信号可用于其他外设（请参见*第 32.3.1 节：RTC 框图*部分）。

通过将 RTC_CR 寄存器中的 WUTIE 位置 1 来使能周期性唤醒中断时，它会使器件退出低功耗模式。

如果已通过 RTC_CR 寄存器的位 OSEL[1:0] 使能周期性唤醒标志，则该标志可连接到 TAMPALRM 输出。可通过 RTC_CR 寄存器的 POL 位配置 TAMPALRM 输出极性。

系统复位以及低功耗模式（睡眠、停机和待机）对唤醒定时器没有任何影响。

32.3.10 RTC 初始化和配置

RTC 二进制、BCD 或混合模式

默认情况下，RTC 处于 BCD 模式（RTC_ICSR 寄存器中的 BIN = 00）：RTC_SSR 寄存器包含亚秒字段值 SS[15:0]，由 ck_apre 提供时钟，允许生成 1 Hz 时钟来更新 BCD 格式的日历寄存器（RTC_TR 和 RTC_DR）。

当 RTC 配置为二进制模式时（RTC_ICSR 寄存器中的 BIN = 01）：RTC_SSR 寄存器包含二进制计数器值 SS[31:0]，由 ck_apre 提供时钟。不使用 BCD 格式的日历寄存器（RTC_TR 和 RTC_DR）。

当 RTC 配置为混合模式时（RTC_ICSR 寄存器中的 BIN = 10 或 11）：RTC_SSR 寄存器包含二进制计数器值 SS[31:0]，由 ck_apre 提供时钟。每次 SSR[BCDU+7:0] 计数到 0 时，日历都会更新（增量为 1 秒）。

RTC 寄存器写保护

系统复位后，可通过电源控制外设中的 DBP 位来保护 RTC 寄存器以防止非正常的写访问（请参见 PWR 电源控制部分）。必须将 DBP位置 1 才能使能 RTC 寄存器的写访问。

备份域复位后，部分 RTC 寄存器受到写保护：RTC_TR、RTC_DR、RTC_PRER、RTC_CALR、RTC_SHIFTR、RTC_ICSR 中的 INIT、BIN 和 BCDU 位以及 RTC_CR 中的 FMT、SUB1H、ADD1H、REFCKON 位。

要解锁 RTC 寄存器的写保护，需要执行以下步骤。

1. 将 0xCA 写入 RTC_WPR 寄存器。
2. 将 0x53 写入 RTC_WPR 寄存器。

写入一个错误的关键字会再次激活写保护。

保护机制不受系统复位影响。

受 INITPRIV 保护的寄存器由 INIT KEY 进行写保护。

受 CALDPRIV 保护的寄存器由 CAL KEY 进行写保护。

如果 RTC_PRIVCFGR 中的 PRIV 或 INITPRIV 置 1：只有在 PRIV 和 INITPRIV 配置定义的特权模式下完成对 RTC_WPR 寄存器的写访问时，才可以解锁和锁定 INIT KEY。

如果 RTC_PRIVCFGR 中的 PRIV 或 CALPRIV 置 1：只有在 PRIV 和 CALPRIV 配置定义的特权模式下完成对 RTC_WPR 寄存器的写访问时，才可以解锁和锁定 CAL KEY。

日历初始化和配置

要编程包括时间格式和预分频器配置在内的初始时间和日期日历值，需按照以下顺序操作：

1. 将 RTC_ICSR 寄存器中的 INIT 位置 1 以进入初始化模式。在此模式下，日历计数器将停止工作并且其值可更新。
2. 轮询 RTC_ICSR 寄存器中的 INITF 位。当 INITF 置 1 时进入初始化阶段模式。
   - 如果 LPCAL=0：INITF 在 INIT 位置 1 后的 2 个 RTCCLK 周期左右置 1。
   - 如果 LPCAL=1：INITF 在 INIT 位置 1 后的 2 个 ck_apre 周期内置 1。
3. 要为日历计数器生成 1 Hz 时钟，应编程 RTC_PRER 寄存器中的两个预分频系数以及 RTC_ICSR 寄存器中的 BIN 和 BCDU。
4. 在影子寄存器（RTC_TR 和 RTC_DR）中加载初始时间和日期值，然后通过 RTC_CR 寄存器中的 FMT 位配置时间格式（12 或 24 小时制）。
5. 通过清零 INIT 位退出初始化模式。随后，自动加载实际日历计数器值。
   - 如果 LPCAL=0：在 4 个 RTCCLK 时钟周期后重新开始计数。
   - 如果 LPCAL=1：在最多 (2 RTCCLK + 1 ck_apre) 个周期后重新开始计数。

当初始化序列完成之后，日历开始计数。RTC_SSR 的内容初始化为：

- 在 BCD 模式下 (BIN=00) 初始化为 PREDIV_S
- 在二进制或混合（BCD 二进制）模式下（BIN=01、10 或 11）初始化为 0xFFFF FFFF。

在 BCD 模式下，RTC_SSR 包含同步预分频器计数器的值。这样，便可计算分辨率低至 1 / (PREDIV_S + 1) 秒的 RTC 的准确时间。因此，可通过增大同步预分频器的值 (PREDIV_S[14:0]) 来提高分辨率。将 PREDIV_S 设置为 0x7FFF 时，可得到允许的最大分辨率（30.52 µs，时钟频率为 32768 Hz）。

但是，提高 PREDIV_S 意味着必须降低 PREDIV_A 才能将同步预分频器的输出维持在 1 Hz。这样，异步预分频器的输出频率会增大，RTC 的动态功耗也会相应增加。RTC 的动态功耗针对 PREDIV_A+1 为 2 的幂数的情况进行了优化。

注意： *系统复位后，应用可读取 RTC_ICSR 寄存器中的 INITS 标志，以检查日历是否已初始化。如果该标志为 0，表明自系统复位以来，日历还尚未初始化过，其年份位域一直还保持着备份域复位默认值 (0x00)。*

注意： *要在初始化之后读取日历，必须首先用软件检查 RTC_ICSR 寄存器的 RSF 标志是否置 1。*

夏令时

可通过 RTC_CR 寄存器的 SUB1H、ADD1H 和 BKP 位管理夏令时。

利用 SUB1H 或 ADD1H，软件只需单次操作便可在日历中减去或增加一个小时，无需执行整个初始化步骤。

此外，软件还可以使用 BKP 位来记录是否曾经执行过此操作。

编程闹钟

要对可编程的闹钟进行编程或更新，必须执行类似的步骤。以下步骤针对闹钟 A，但同样适用于闹钟 B。

1. 将 RTC_CR 中的 ALRAE 位清零以禁止闹钟 A。
2. 编程闹钟 A 寄存器（RTC_ALRMASSR/RTC_ALRMAR 或 RTC_ALRABINR）。
3. 将 RTC_CR 寄存器中的 ALRAE 位置 1 以再次使能闹钟 A。

注意： *由于时钟同步的缘故，RTC_CR 寄存器的每次更改都将需要大约 2 个 RTCCLK 时钟周期来完成。*

编程唤醒定时器

要配置或更改唤醒定时器的自动重载值（RTC_WUTR 中的 WUT[15:0]），需要按照以下顺序操作：

1. 将 RTC_CR 中的 WUTE 清零以禁止唤醒定时器。
2. 轮询 RTC_ICSR 中的 WUTWF，直到该位置 1，以确保可以访问唤醒自动重载定时器和 WUCKSEL[2:0] 位。在日历初始化模式下必须跳过该步骤。
   - 如果 WUCKSEL[2] = 0：WUTWF 在 WUTE 位清零后的 (1 ck_wut + 1 RTCCLK) 个周期左右置 1。
   - 如果 WUCKSEL[2] = 1：WUTWF 在 WUTE 位清零后的 (1 ck_apre + 1 RTCCLK) 个周期内置 1。
3. 编程唤醒自动重载值 WUT[15:0] 和 WUTOCLR[15:0]，并选择唤醒时钟（RTC_CR 中的 WUCKSEL[2:0] 位）。将 RTC_CR 寄存器中的 WUTE 位置 1 以再次使能定时器。唤醒定时器重新开始递减计数。
   - 如果 WUCKSEL[2] = 0：WUTWF 在 WUTE 位置 1 后的 (1 ck_wut + 1 RTCCLK) 个周期左右清零。
   - 如果 WUCKSEL[2] = 1：WUTWF 在 WUTE 位置 1 后的 (1 ck_apre + 1 RTCCLK) 个周期内清零。

32.3.11 读取日历

当 RTC_CR 寄存器中的 BYPSHAD 控制位清零时

要正确读取 RTC 日历寄存器（RTC_SSR、RTC_TR 和 RTC_DR），APB 时钟频率 (f_{PCLK}) 必须等于或大于 RTC 时钟频率 (f_{RTCCLK}) 的七倍。这可以确保同步机制的安全性。

如果 APB 时钟频率低于 RTC 时钟频率的七倍，则软件必须分两次读取日历时间寄存器和日期寄存器。这样，当两次读取的 RTC_TR 结果相同时，才能确保数据正确。否则必须执行第三次读访问。任何情况下，APB 的时钟频率都不能低于 RTC 的时钟频率。

每次将日历寄存器中的值复制到 RTC_SSR、RTC_TR 和 RTC_DR 影子寄存器时，RTC_ICSR 寄存器中的 RSF 位都会置 1。每个 RTCCLK 周期执行一次复制。为确保这 3 个值一致，读取 RTC_SSR 或 RTC_TR 时会锁定高阶日历影子寄存器中的值，直到读取 RTC_DR。为避免软件对日历执行读访问的时间间隔小于 1 个 RTCCLK 周期：第一次读取日历之后必须通过软件将 RSF 清零，并且软件必须等待到 RSF 置 1 之后才可再次读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

从低功耗模式（停止模式或待机模式）唤醒之后，必须通过软件将 RSF 清零。之后，软件必须等待至 RSF 再次置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

RSF 位必须在唤醒之后而不是进入低功耗模式之前进行清零。

系统复位之后，软件必须等待至 RSF 置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。实际上，系统复位会将影子寄存器复位为其默认值。

初始化之后（请参见*第 1015 页的日历初始化和配置*），软件必须等待至 RSF 置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

同步之后（请参见*第 32.3.13 节：RTC 同步*），软件必须等待至 RSF 置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

当 RTC_CR 寄存器中的 BYPSHAD 控制位置 1 时（旁路影子寄存器）

读取日历寄存器时会直接从日历计数器获取值，这样便无需等待至 RSF 位置 1。这对于从低功耗模式（停止模式或待机模式）退出后的情况特别有用，因为影子寄存器在这些模式下不更新。

当 BYPSHAD 位置 1 时，如果在对寄存器的两次读访问之间出现 RTCCLK 沿，则不同寄存器的结果彼此可能不一致。此外，如果在读操作期间出现 RTCCLK 沿，则可能导致其中一个寄存器的值不正确。软件必须分两次读取所有寄存器，然后将两次结果加以比较来确认数据是否一致和正确。此外，软件也可以只比较两次读取日历寄存器得到的结果的最低位。

注意： *当 BYPSHAD = 1 时，读取日历寄存器的指令需要一个额外的 APB 周期才能完成。*

32.3.12 复位 RTC

日历影子寄存器（RTC_SSR、RTC_TR 和 RTC_DR）以及 RTC 状态寄存器 (RTC_ICSR) 的某些位通过所有可用的系统复位源复位为各自的默认值。

而以下寄存器则通过备份域复位来复位为各自的默认值并且不受系统复位的影响：RTC 当前日历寄存器、RTC 控制寄存器 (RTC_CR)、预分频器寄存器 (RTC_PRER)、RTC 校准寄存器 (RTC_CALR)、RTC 移位寄存器 (RTC_SHIFTR)、RTC 时间戳寄存器（RTC_TSSSR、RTC_TSTR 和 RTC_TSDR）、唤醒定时器寄存器 (RTC_WUTR)、闹钟 A 寄存器和闹钟 B 寄存器（RTC_ALRMASSR/RTC_ALRMAR/RTC_ALRABINR 和 RTC_ALRMBSSR/RTC_ALRMBR/RTC_ALRBBINR）。

此外，当由 LSE 提供时钟时，如果复位源并非备份域复位源（有关不受系统复位影响的 RTC 时钟源的详细信息，请参见 RCC），则 RTC 将在系统复位时保持运行状态。发生备份域复位时，RTC 会停止工作，并且所有 RTC 寄存器都会设置为各自的复位值。

32.3.13 RTC 同步

RTC 可与高精度的远程时钟同步。在读取亚秒字段后（RTC_SSR 或 RTC_TSSSR），即可计算远程时钟的时间与 RTC 之间的精准偏差。之后，可使用 RTC_SHIFTR 对 RTC 的时钟进行零点几秒的“平移”，经过调整后可消除此偏差。

可以使用 RTC 平移控制寄存器 (RTC_SHIFTR) 对 RTC 进行微调。可以用大小为 1 个 ck_apre 周期的分辨率对 RTC_SHIFTR 进行写操作来平移（延迟或提前）时钟。

在这种平移操作中，会将 SUBFS[14:0] 值加到同步预分频器计数器 SS[15:0] 中：这会使时钟产生延迟。

在 BCD 或混合模式下，如果同时将 ADD1S 位置 1，则会增加一秒，与此同时减去的时间为零点几秒，因此会使时钟提前。ADD1S 在二进制模式下不起作用。

对 RTC_SHIFTR 寄存器执行写操作以启动平移操作时，硬件会将 SHPF 标志置 1 以指示平移操作挂起。完成平移操作时，硬件会将该位清零。

小心： 在混合模式下（BIN=10 或 11），SUBFS[14:BCDU+8] 必须写入 0。

小心： 在 BCD 模式下初始化平移操作前，用户必须检查确认 SS[15] = 0，以确保不会发生上溢。在混合模式下，用户必须检查确认位 SS[BCDU+8] = 0。

小心： 该同步功能与参考时钟检测功能不兼容：当 REFCKON = 1 时，固件不能对 RTC_SHIFTR 执行写操作。

32.3.14 RTC 参考时钟检测

该功能仅在 BCD 模式 (BIN=00) 下可用。

RTC 日历更新可与参考时钟 RTC_REFIN（通常为市电频率，50 Hz 或 60 Hz）同步。RTC_REFIN 参考时钟的精度应高于 32.768 kHz LSE 时钟。使能 RTC_REFIN 检测时（将 RTC_CR 的 REFCKON 位置 1），日历仍由 LSE 提供时钟，而 RTC_REFIN 用于补偿不准确的日历更新频率 (1 Hz)。

每个 1 Hz 时钟边沿都与最近的 RTC_REFIN 时钟边沿进行比较（如果在给定的时间窗口内发现一个边沿）。在大多数情况下，两个时钟边沿恰好对齐。当 1 Hz 时钟由于 LSE 时钟不精确而发生偏离时，RTC 会稍微偏移 1 Hz 时钟，以便后续的 1 Hz 时钟边沿能够对齐。利用这种机制，可使日历像参考时钟一样精确。

RTC 使用 32.768 kHz 石英产生的 256 Hz 时钟 (ck_apre) 检测是否存在参考时钟源。大约在日历每次更新时（每 1 秒钟），便会在时间窗口期间执行一次检测。检测到第一个参考时钟边沿时，该窗口等于 7 个 ck_apre 周期。随后的日历更新使用长度为 3 个 ck_apre 周期的较小窗口。

每次在窗口中检测到参考时钟时，都会强制输出 ck_spre 时钟的异步预分频器进行重载。当参考时钟与 1 Hz 时钟对齐时，此操作不起作用，因为预分频器会在同一时刻重载。当时钟不对齐时，重载操作会微调后续的 1 Hz 时钟边沿，使其与参考时钟对齐。

如果参考时钟停止（在 3 个 ck_apre 窗口内未出现参考时钟边沿），日历将仅根据 LSE 时钟进行连续更新。RTC 随后使用 ck_spre 边沿上居中的大检测窗口（7 个 ck_apre 周期）等待参考时钟。

使能 RTC_REFIN 检测后，必须将 PREDIV_A 和 PREDIV_S 设置为各自的默认值：

- PREDIV_A = 0x007F
- PREVID_S = 0x00FF

注意： *RTC_REFIN 时钟检测在待机模式下不可用。*

32.3.15 RTC 精密数字校准

RTC 频率可采用约 0.954 ppm 的分辨率进行数字校准，校准范围为 -487.1 ppm 到 +488.5 ppm。使用一系列微调（增加和/或减少单独的 ck_cal 脉冲）进行频率校正。

如果 LPCAL=0：ck_cal = RTCCLK

如果 LPCAL=1：ck_cal = ck_apre

这些微调的分布非常均匀，因此 RTC 的校准效果相当好，即使在短时间内持续观察也是如此。

RTC 超低功耗模式

将 RTC 校准寄存器 (RTC_CALR) 中的 LPCAL 位置 1，可以降低 RTC 的功耗。在这种情况下，校准机制应用于 ck_apre 而不是 RTCCLK。最终产生的精度是相同的，但校准的周期约为 2^{20} x PREDIV_A x RTCCLK 个脉冲，而非 2^{20} 个 RTCCLK 脉冲（当 LPCAL=0 时）。

精密校准机制

精密校准寄存器 (RTC_CALR) 可指定校准周期内要减少的 ck_cal 时钟周期数：

- 将位 CALM[0] 置 1 时，校准周期内将只减少一个时钟。
- 将 CALM[1] 置 1 时，将减少两个时钟。
- 将 CALM[2] 置 1 时，将减少四个时钟。
- 依此类推，将 CALM[8] 置 1 时，将减少 256 个时钟。

注意： *CALM[8:0] (RTC_CALR) 可指定校准周期内要减少的 ck_cal 脉冲数。将位 CALM[0] 置 1 时，校准周期内将只减少 1 个脉冲（当 cal_cnt[19:0] = 0x80000 时）；将 CALM[1] 置 1 时，将减少 2 个周期（当 cal_cnt = 0x40000 和 0xC0000 时）；将 CALM[2] 置 1 时，将减少 4 个周期（当 cal_cnt = 0x20000/0x60000/0xA0000/0xE0000 时）；依此类推，将 CALM[8] 置 1 时，将减少 256 个时钟（当 cal_cnt = 0xXX800 时）。*

使用适当分辨率时，CALM 可使 RTC 频率减少最多 487.1 ppm，而 CALP 可用于使频率增加 488.5 ppm。将 CALP 置 1，可每隔 2^{11} 个 ck_cal 周期有效插入一个额外的 ck_cal 脉冲，这意味着每个校准周期内可增加 512 个时钟。

将 CALM 和 CALP 配合使用时，可在校准周期内增加一个范围为 -511 到 +512 ck_cal 周期的偏差，对应的校准范围为 -487.1 ppm 到 +488.5 ppm，分辨率约为 0.954 ppm。

若输入频率 (F_{RTCCLK}) 已知，可通过以下公式计算有效校准频率 (F_{CAL})：

$$F_{CAL} = F_{RTCCLK} \times [1 + (CALP \times 512 - CALM) / (2^{20} + CALM - CALP \times 512)]$$

小心： PREDIV_A 必须大于或等于 3。

PREDIV_A < 3 条件下的校准

当异步预分频器值（RTC_PRER 寄存器中的 PREDIV_A 位）小于 3 时，不能将 CALP 位置 1。如果 CALP 已置 1 并且 PREDIV_A 位的值小于 3，则会忽略 CALP，即假定 CALP 等于 0 而执行校准。

但是，在 BCD 模式下，要在 PREDIV_A 小于 3 的条件下执行校准，应降低同步预分频器值 (PREDIV_S) 以便每秒内可加速 8 个 ck_cal 时钟周期，这意味着每个校准周期可增加 256 个时钟周期。因此，仅使用 CALM 位，可在每个校准周期内有效增加 255 到 256 个时钟脉冲（对应的校准范围为 243.3 ppm 到 244.1 ppm）。

在标称 RTCCLK 频率 32768 Hz 下，当 PREDIV_A 等于 1 时（分频系数为 2），应将 PREDIV_S 设置为 16379 而不是 16383（少 4）。唯一相关的其他情况是，当 PREDIV_A 等于 0 时，应将 PREDIV_S 设置为 32759 而不是 32767（少 8）。

如果以这种方式减少 PREDIV_S，则采用以下公式计算校准输入时钟的有效频率：

$F_{CAL} = F_{RTCCLK} \times [1 + (256 - CALM) / (2^{20} + CALM - 256)]$

在这种情况下，如果 RTCCLK 恰好为 32768.00 Hz，则当 CALM[7:0] 等于 0x100 时（CALM 范围的中值），说明设置正确。

验证 RTC 校准

建议在 LPCAL = 0 的条件下验证 RTC 校准，以获得 32 秒校准周期。

通过测量 RTCCLK 的精确频率，计算正确的 CALM 和 CALP 值以确保 RTC 精度。此外，还为应用提供了一个可选的 1 Hz 输出，用来测量和验证 RTC 精度。

如果在有限的间隔内测量 RTC 的精确频率，则会导致测量期间产生最多 2 个 RTCCLK 时钟周期的测量误差，具体取决于数字校准周期与测量周期的对齐方式。

但是，如果测量周期与校准周期的长度相同，则可以消除此测量误差。在这种情况下，观测到的唯一误差是由数字校准的分辨率导致的误差。

- 默认情况下，校准周期为 32 秒。

  在此模式下，测量整个 32 秒内 1 Hz 输出的精度，可确保测量误差在 0.477 ppm 内（32 秒内为 0.5 个 RTCCLK 周期，受校准分辨率限制）。
- 可将 RTC_CALR 寄存器的 CALW16 位置 1，以强制 16 秒的校准周期。

  此时，可在 16 秒内测量 RTC 精度，产生的最大误差为 0.954 ppm（16 秒内为 0.5 个 RTCCLK 周期）。但是，由于校准分辨率降低，长期的 RTC 精度也会降到 0.954 ppm：将 CALW16 置 1 时，CALM[0] 位将始终保持为 0。
- 可将 RTC_CALR 寄存器的 CALW8 位置 1，以强制 8 秒的校准周期。

  此时，可在 8 秒内测量 RTC 精度，产生的最大误差为 1.907 ppm（8 秒内为 0.5 个 RTCCLK 周期）。长期的 RTC 精度也会降到 1.907 ppm：将 CALW8 置 1 时，CALM[1:0] 位将始终保持为 00。

动态重校准

当 RTC_ICSR/INITF = 0 时，可动态更新校准寄存器 (RTC_CALR)，具体步骤如下：

1. 轮询 RTC_ICSR/RECALPF（重新校准挂起标志）。
2. 如果该标志为 0，则可以根据需要向 RTC_CALR 写入新值。随后 RECALPF 位会被自动置为 1。
3. 新校准设置将在对 RTC_CALR 执行写操作之后的三个 ck_apre 周期内生效。

32.3.16 时间戳功能

将 RTC_CR 寄存器的 TSE 或 ITSE 位置 1 可使能时间戳。

将 TSE 置 1 时：

当在 RTC_TS 引脚上检测到时间戳事件时，日历会保存到时间戳寄存器（RTC_TSSSR、RTC_TSTR 和 RTC_TSDR）中。

将 TAMPTS 置 1 时：

当检测到内部或外部时间戳事件时，日历会保存到时间戳寄存器（RTC_TSSSR、RTC_TSTR 和 RTC_TSDR）中。请参见 *RTC 控制寄存器 (RTC_CR)* 和*入侵事件的时间戳*一节。

将 ITSE 置 1 时：

当检测到内部时间戳事件时，日历会保存到时间戳寄存器（RTC_TSSSR、RTC_TSTR 和 RTC_TSDR）中。内部时间戳事件通过切换到 V_{BAT} 电源生成。

由于内部事件或外部事件而发生时间戳事件时，RTC_SR 寄存器中的时间戳标志位 (TSF) 将置 1。如果是内部事件，RTC_SR 寄存器中的 ITSF 标志也将置 1。

通过将 RTC_CR 寄存器中的 TSIE 位置 1，可在发生时间戳事件时生成中断。

如果在时间戳标志 (TSF) 已置 1 的条件下检测到新的时间戳事件，则时间戳上溢标志 (TSOVF) 将置 1，而时间戳寄存器（RTC_TSTR 和 RTC_TSDR）将保持上一事件的结果。

注意： *由于同步过程，TSF 将在来自 RTC_TS 引脚或 rtc_its 内部信号的时间戳事件发生后的 2 个 ck_apre 周期内置 1。TSF 在入侵标志置 1 后的 3 个 ck_apre 周期内置 1。*

TSOVF 在事件发生后的仅 1 个 ck_apre 周期内就会置 1。这意味着，如果两个时间戳事件接连发生，则 TSOVF 可能为“1”而 TSF 仍为“0”。因此，建议只在检测到 TSF 为“1”后再轮询 TSOVF。

小心： 如果在 TSF 位清零后紧接着发生时间戳事件，则 TSF 和 TSOVF 位都将置 1。为防止在时间戳事件发生的同时屏蔽该事件，除非已将 TSF 位读取为 1，否则应用程序不得将 0 写入 TSF 位。

32.3.17 校准时钟输出

将 RTC_CR 寄存器中的 COE 位置 1 时，会在 CALIB 器件输出上提供一个参考时钟。

如果 RTC_CR 寄存器中的 COSEL 位复位且 PREDIV_A = 0x7F，则 CALIB 频率为 $f_{RTCCLK/64}$。这相当于 RTCCLK 频率为 32.768 kHz 时，512 Hz 的校准输出。CALIB 占空比是不规则的：下降沿上存在轻微抖动。因此推荐使用上升沿。

当 COSEL 置 1 且“PREDIV_S+1”为 256 的非零整数倍（即：PREDIV_S[7:0] = 0xFF）时，CALIB 频率为 $f_{RTCCLK}/(256 * (PREDIV_A+1))$。这相当于 RTCCLK 频率为 32.768 kHz 时，1 Hz 的校准输出，其中预分频器为默认值（PREDIV_A = Ox7F、PREDIV_S = 0xFF）。

注意： *COSEL 清零时，CALIB 输出为异步预分频器的第 6 级输出。如果 LPCAL 从 0 变为 1，则在 LPCAL 切换过程中输出可能是不规则的（干扰...）。LPCAL = 1 时，该输出始终可用。LPCAL = 0 且 PREDIV_A < 0x20 时，不存在任何输出。*

COSEL 置 1 时，CALIB 输出为同步预分频器的第 8 级输出。

32.3.18 入侵和闹钟输出

RTC_CR 寄存器中的 OSEL[1:0] 控制位用于激活闹钟输出 TAMPALRM，以及选择输出的功能。这些功能可反映 RTC_SR 寄存器中相应标志的内容。

当 RTC_CR 中的 TAMPOE 控制位置 1 时，所有外部和内部入侵标志将进行逻辑或运算并连接到 TAMPALRM 输出。如果 OSEL = 00，则 TAMPALRM 输出仅反映入侵标志。如果 OSEL ≠ 00，则 TAMPALRM 上的信号反映入侵标志以及闹钟 A、闹钟 B 或唤醒标志。

TAMPALRM 输出的极性由 RTC_CR 中的 POL 控制位确定，这样当 POL 置 1 时会输出选定标志位的相反值。

TAMPALRM 输出

使用 RTC_CR 寄存器中的控制位 TAMPALRM_TYPE 可将 TAMPALRM 引脚配置为输出开漏或输出上拉。借助 RTC_CR 中的 TAMPALRM_PU，可以在输出模式下应用内部上拉。

注意： *使能* TAMPALRM *输出后，其优先级高于 RTC_OUT1 上的 CALIB。*

如果在 RTC 中将 TAMPALRM 配置为开漏，则必须将 RTC_OUT1 GPIO 配置为输入。

32.4 RTC 低功耗模式

表 236. 低功耗模式对 RTC 的影响

模式	说明
睡眠	无影响 RTC 中断可使器件退出睡眠模式。
停止	当 RTC 时钟源为 LSE 或 LSI 时，RTC 保持工作状态。RTC 中断可使器件退出停止模式。
待机	当 RTC 时钟源为 LSE 或 LSI 时，RTC 保持工作状态。RTC 中断可使器件退出待机模式。

下表汇总了所有模式下的 RTC 引脚和功能。

表 237. 各个模式下的 RTC 引脚功能

功能	待机模式以外的所有低功耗模式下的功能正常	待机模式下的功能正常	V_{BAT} 模式下的功能正常
RTC_TS	是	是	是
RTC_REFIN	是	否	否
RTC_OUT1	是	是	是
RTC_OUT2	是	否	否

32.5 RTC 中断

中断通道在屏蔽中断状态寄存器中置 1。中断输出也会激活。

表 238. 中断请求

中断缩写	中断事件	事件标志(1)	使能控制位(2)	中断清除方法	退出低功耗模式
RTC	闹钟 A	ALRAF	ALRAIE	向 CALRAF 中写入 1	是(3)
	闹钟 B	ALRBF	ALRBIE	向 CALRBF 中写入 1	是(3)
	时间戳	TSF	TSIE	向 CTSF 中写入 1	是(3)
	唤醒定时器	WUTF	WUTIE	向 CWUTF 中写入 1	是(3)
	SSR 下溢	SSRUF	SSRUIE	向 CSSRUF 中写入 1	是(3)

1. 事件标志位于 RTC_SR 寄存器中。
2. 中断屏蔽标志（由事件标志和使能控制位的逻辑与运算得出）位于 RTC_MISR 寄存器中。
3. 通过振荡器提供时钟时，RTC 可以在低功耗模式下正常工作。

32.6 RTC 寄存器

有关寄存器说明中使用的缩写，请参见参考手册中的*第 1.2 节*。

外设寄存器可按字（32 位）进行访问。

32.6.1 RTC 时间寄存器 (RTC_TR)

RTC time register

RTC_TR 是日历时间影子寄存器。只能在初始化模式下对该寄存器执行写操作。请参见*第 1015 页的日历初始化和配置*和*第 1017 页的读取日历*。

此寄存器受写保护。*第 1015 页的RTC 寄存器写保护*中介绍了写访问的过程。

可以对该寄存器进行写保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x00

备份域复位值：0x0000 0000

系统复位值：0x0000 0000（当 BYPSHAD = 0 时，当 BYPSHAD = 1 时不受影响）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PM	HT[1:0]		HU[3:0]			
									rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	MNT[2:0]			MNU[3:0]				Res.	ST[2:0]			SU[3:0]			
	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

位 31:23 保留，必须保持复位值。

位 22 **PM**：AM/PM 符号 (AM/PM notation)
0：AM 或 24 小时制
1：PM

位 21:20 **HT[1:0]**：小时的十位（BCD 格式）(Hour tens in BCD format)

位 19:16 **HU[3:0]**：小时的个位（BCD 格式）(Hour units in BCD format)

位 15 保留，必须保持复位值。

位 14:12 **MNT[2:0]**：分钟的十位（BCD 格式）(Minute tens in BCD format)

位 11:8 **MNU[3:0]**：分钟的个位（BCD 格式）(Minute units in BCD format)

位 7 保留，必须保持复位值。

位 6:4 **ST[2:0]**：秒的十位（BCD 格式）(Second tens in BCD format)

位 3:0 **SU[3:0]**：秒的个位（BCD 格式）(Second units in BCD format)

32.6.2 RTC 日期寄存器 (RTC_DR)

RTC date register

RTC_DR 是日历日期影子寄存器。只能在初始化模式下对该寄存器执行写操作。请参见*第 1015 页的日历初始化和配置*和*第 1017 页的读取日历*。

此寄存器受写保护。*第 1015 页的RTC 寄存器写保护*中介绍了写访问的过程。

可以对该寄存器进行写保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x04

备份域复位值：0x0000 2101

系统复位值：0x0000 2101（当 BYPSHAD = 0 时，当 BYPSHAD = 1 时不受影响）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	YT[3:0]				YU[3:0]			
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDU[2:0]			MT	MU[3:0]				Res.	Res.	DT[1:0]		DU[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23:20 **YT[3:0]**：年份的十位（BCD 格式）(Year tens in BCD format)

位 19:16 **YU[3:0]**：年份的个位（BCD 格式）(Year units in BCD format)

位 15:13 **WDU[2:0]**：星期几的个位 (Week day units)
000：禁止
001：星期一
...
111：星期日

位 12 **MT**：月份的十位（BCD 格式）(Month tens in BCD format)

位 11:8 **MU[3:0]**：月份的个位（BCD 格式）(Month units in BCD format)

位 7:6 保留，必须保持复位值。

位 5:4 **DT[1:0]**：日期的十位（BCD 格式）(Date tens in BCD format)

位 3:0 **DU[3:0]**：日期的个位（BCD 格式）(Date units in BCD format)

注意： *日历在达到最大值时会冻结，而不会翻转。*

32.6.3 RTC 亚秒寄存器 (RTC_SSR)

RTC subsecond register

偏移地址：0x08

备份域复位值：0x0000 0000

系统复位值：0x0000 0000（当 BYPSHAD = 0 时，当 BYPSHAD = 1 时不受影响）

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

位 31:0 **SS[31:0]**：同步二进制计数器 (Synchronous binary counter)

SS[31:16]：同步二进制计数器 MSB 值 (Synchronous binary counter MSB values)

选择二进制模式或混合模式时（BIN = 01、10 或 11）：

SS[31:16] 是 SS[31:0] 自由运行递减计数器的高 16 位。

选择 BCD 模式时 (BIN=00)：

SS[31:16] 由硬件强制设为 0x0000。

SS[15:0]：亚秒值/同步二进制计数器 LSB 值 (Subsecond value/synchronous binary counter LSB values)

选择二进制模式时（BIN = 01、10 或 11）：

SS[15:0] 是 SS[31:0] 自由运行递减计数器的低 16 位。

选择 BCD 模式时 (BIN=00)：

SS[15:0] 是同步预分频器计数器的值。此亚秒值可根据以下公式得出：

亚秒值 = (PREDIV_S - SS) / (PREDIV_S + 1)

仅当执行平移操作之后，SS 才能大于 PREDIV_S。在这种情况下，正确的时间/日期比 RTC_TR/RTC_DR 所指示的时间/日期慢一秒钟。

32.6.4 RTC 初始化控制和状态寄存器 (RTC_ICSR)

RTC initialization control and status register

此寄存器受写保护。*第 1015 页的RTC 寄存器写保护*中介绍了写访问的过程。

既可以为该寄存器设置全局保护，也可以为该寄存器的每个位单独设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x0C

备份域复位值：0x0000 0007

系统复位：不受影响（INIT、INITF 和 RSF 位除外，它们在复位时被清零）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RECALPF
															r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	BCDU[2:0]			BIN[1:0]		INIT	INITF	RSF	INITS	SHPF	WUTWF	Res.	Res.
			rw	rw	rw	rw	rw	rw	r	rc_w0	r	r	r		

位 31:17 保留，必须保持复位值。

位 16 **RECALPF**：重新校准挂起标志 (Recalibration pending Flag)

当软件对 RTC_CALR 寄存器执行写操作时，RECALPF 状态标志将自动置 1，指示 RTC_CALR 寄存器已屏蔽。当采用新的校准设置时，该位恢复为 0。请参见*动态重校准*。

位 15:13 保留，必须保持复位值。

位 12:10 **BCDU[2:0]**：BCD 更新 (BCD update)（BIN = 10 或 11）

在混合模式下，如果同时使用 BCD 日历和二进制扩展计数器（BIN = 10 或 11），则将使用 SSR 最低有效位递增日历秒单元。

0x0：每当 SS[7:0] = 0 时，就会产生 1s 日历增量
0x1：每当 SS[8:0] = 0 时，就会产生 1s 日历增量
0x2：每当 SS[9:0] = 0 时，就会产生 1s 日历增量
0x3：每当 SS[10:0] = 0 时，就会产生 1s 日历增量
0x4：每当 SS[11:0] = 0 时，就会产生 1s 日历增量
0x5：每当 SS[12:0] = 0 时，就会产生 1s 日历增量
0x6：每当 SS[13:0] = 0 时，就会产生 1s 日历增量
0x7：每当 SS[14:0] = 0 时，就会产生 1s 日历增量

位 9:8 **BIN[1:0]**：二进制模式 (Binary mode)

00：自由运行 BCD 日历模式（禁止二进制模式）
01：自由运行二进制模式（禁止 BCD 模式）
10：自由运行 BCD 日历模式和二进制模式
11：自由运行 BCD 日历模式和二进制模式

位 7 **INIT**：初始化模式 (Initialization mode)

0：自由运行模式
1：初始化模式，用于编程时间和日期寄存器（RTC_TR 和 RTC_DR）和预分频器寄存器 (RTC_PRER) 以及 BIN 和 BCDU 位域。计数器停止计数，当 INIT 被复位后，计数器从新值开始计数。

位 6 **INITF**：初始化标志 (Initialization flag)

当该位置 1 时，RTC 处于初始化状态，此时可更新事件、日期和预分频器寄存器。

0：不允许更新日历寄存器

1：允许更新日历寄存器

位 5 **RSF**：寄存器同步标志 (Registers synchronization flag)

每次将日历寄存器的值复制到影子寄存器（RTC_SSR、RTC_TR 和 RTC_DR）时，都会由硬件将该位置 1。在初始化模式下、平移操作挂起时 (SHPF = 1) 或在旁路影子寄存器模式 (BYPSHAD = 1) 下，该位由硬件清零。该位还可由软件清零。

在初始化模式下，该位可由软件或硬件清零。

0：日历影子寄存器尚未同步

1：日历影子寄存器已同步

位 4 **INITS**：初始化状态标志 (Initialization status flag)

当日历年份字段不为 0 时（备份域复位状态），由硬件将该位置 1。

0：日历尚未初始化

1：日历已经初始化

位 3 **SHPF**：平移操作挂起 (Shift operation pending)

只要通过对 RTC_SHIFTR 寄存器执行写操作来启动平移操作，此标志便由硬件置 1。执行完相应的平移操作后，此标志由硬件清零。对 SHPF 位执行写入操作不起作用。

0：没有平移操作挂起

1：某个平移操作挂起

位 2 **WUTWF**：唤醒定时器写标志 (Wake-up timer write flag)

在 RTC_CR 寄存器中的 WUTE 位置 0 后，当 WUT 的值可更改时，由硬件将该位置 1。该位在初始化模式下由硬件清零。

0：除初始化模式外，不允许更新唤醒定时器配置

1：允许更新唤醒定时器配置

位 1:0 保留，必须保持复位值。

32.6.5 RTC 预分频器寄存器 (RTC_PRER)

RTC prescaler register

只能在初始化模式下对该寄存器执行写操作。必须通过两次独立的写访问执行初始化。请参见*第 1015 页的日历初始化和配置*。

此寄存器受写保护。*第 1015 页的RTC 寄存器写保护*中介绍了写访问的过程。

可以对该寄存器进行写保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x10

备份域复位值：0x007F 00FF

系统复位：不受影响

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

位 31:23 保留，必须保持复位值。

位 22:16 **PREDIV_A[6:0]**：异步预分频系数 (Asynchronous prescaler factor)
下面是异步分频系数的公式：
ck_apre 频率 = RTCCLK 频率/(PREDIV_A+1)

位 15 保留，必须保持复位值。

位 14:0 **PREDIV_S[14:0]**：同步预分频系数 (Synchronous prescaler factor)
下面是同步分频系数的公式：
ck_spre 频率 = ck_apre 频率/(PREDIV_S+1)

32.6.6 RTC 唤醒定时器寄存器 (RTC_WUTR)

RTC wake-up timer register

仅当 RTC_ICSR 中的 WUTWF 置 1 时才可对该寄存器执行写操作。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x14

备份域复位值：0x0000 FFFF

系统复位：不受影响

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

位 31:16 **WUTOCLR[15:0]**：唤醒自动重载输出清零值 (Wake-up auto-reload output clear value)
如果 WUTOCLR[15:0] 不等于 0x0000，当自动重载递减计数器计数到 0 时，WUTF 由硬件置 1；当自动重载递减计数器计数到 WUTOCLR[15:0] 时，WUTF 由硬件清零。
如果 WUTOCLR[15:0] = 0x0000，则 WUTF 由软件清零，并在 WUT 递减计数器计数到 0 时由硬件置 1。

位 15:0 **WUT[15:0]**：唤醒自动重载值位 (Wake-up auto-reload value bits)
当使能唤醒定时器时（WUTE 置 1），每 (WUT[15:0] + 1) 个 ck_wut 周期将 WUTF 标志置 1 一次。ck_wut 周期通过 RTC_CR 寄存器的 WUCKSEL[2:0] 位进行选择。
当 WUCKSEL[2] = 1 时，唤醒定时器变为 17 位，WUCKSEL[1] 等效为 WUT[16]，即要重载到定时器的最高有效位。
WUTF 第一次置 1 发生在 WUTE 置 1 后的 WUT 个周期到 (WUT + 2) 个 ck_wut 周期之间。
禁止在 WUCKSEL[2:0] = 011 (RTCCLK/2) 时将 WUT[15:0] 设置为 0x0000。

32.6.7 RTC 控制寄存器 (RTC_CR)

RTC control register

此寄存器受写保护。第 1015 页的RTC 寄存器写保护中介绍了写访问的过程。

既可以为该寄存器设置全局保护，也可以为该寄存器的每个位单独设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x18

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OUT2EN	TAMPALRM_TYPE	TAMPALRM_PU	ALRBFCLR	ALRAFCLR	TAMPOE	TAMPTS	ITSE	COE	OSEL[1:0]		POL	COSEL	BKP	SUB1H	ADD1H
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSIE	WUTIE	ALRBIE	ALRAIE	TSE	WUTE	ALRBE	ALRAE	SSRUIE	FMT	BYPSHAD	REFCKON	TSEDGE	WUCKSEL[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **OUT2EN**：RTC_OUT2 输出使能 (RTC_OUT2 output enable)

当该位置 1 时，可将 RTC 输出重映射到 RTC_OUT2，具体如下：

OUT2EN = 0：RTC 输出 2 禁止

如果 OSEL ≠ 00 或 TAMPOE = 1：在 RTC_OUT1 上输出 TAMPALRM

如果 OSEL = 00、TAMPOE = 0 且 COE = 1：在 RTC_OUT1 上输出 CALIB

OUT2EN = 1：RTC 输出 2 使能

如果 (OSEL ≠ 00 或 TAMPOE = 1) 且 COE = 0：在 RTC_OUT2 上输出 TAMPALRM

如果 OSEL = 00、TAMPOE = 0 且 COE = 1：在 RTC_OUT2 上输出 CALIB

如果（OSEL ≠ 00 或 TAMPOE = 1）且 COE = 1：在 RTC_OUT2 上输出 CALIB，在 RTC_OUT1 上输出 TAMPALRM。

位 30 **TAMPALRM_TYPE**：TAMPALRM 输出类型 (TAMPALRM output type)

0：TAMPALRM 为推挽输出

1：TAMPALRM 为开漏输出

位 29 **TAMPALRM_PU**：TAMPALRM 上拉使能 (TAMPALRM pull-up enable)

0：不对 TAMPALRM 输出应用上拉

1：对 TAMPALRM 输出应用上拉

位 28 **ALRBFCLR**：闹钟 B 标志自动清零 (Alarm B flag automatic clear)

0：闹钟 B 事件生成触发事件，必须通过软件将 ALRBF 清零来允许下一次闹钟事件。

1：闹钟 B 事件生成触发事件。ALRBF 在 1 个 ck_apre 周期后由硬件自动清零。

位 27 **ALRAFCLR**：闹钟 A 标志自动清零 (Alarm A flag automatic clear)

0：闹钟 A 事件生成触发事件，必须通过软件将 ALRAF 清零来允许下一次闹钟事件。

1：闹钟 A 事件生成触发事件。ALRAF 在 1 个 ck_apre 周期后由硬件自动清零。

位 26 **TAMPOE**：TAMPALRM 上的入侵检测输出使能 (Tamper detection output enable on TAMPALRM)

0：入侵标志未发送到 TAMPALRM

1：入侵标志发送到 TAMPALRM，并与 OSEL 提供的信号和 POL 提供的极性相结合。

位 25 **TAMPTS**：发生入侵检测事件时激活时间戳 (Activate timestamp on tamper detection event)

0：发生入侵检测事件时不保存 RTC 时间戳

1：发生入侵检测事件时保存 RTC 时间戳

即便 RTC_CR 寄存器中的 TSE = 0，TAMPTS 仍有效。时间戳标志在入侵标志置 1 后的 3 个 ck_apre 周期内置 1。

注意：必须在进入 RTC 初始化模式之前，将 TAMPTS 清零。

位 24 **ITSE**：内部事件时间戳使能 (timestamp on internal event enable)

0：禁止内部事件时间戳

1：使能内部事件时间戳

位 23 **COE**：校准输出使能 (Calibration output enable)

该位使能 CALIB 输出

0：禁止校准输出

1：使能校准输出

位 22:21 **OSEL[1:0]**：输出选择 (Output selection)

这些位用于选择要连接到 TAMPALRM 输出的标志。

00：禁止输出

01：使能闹钟 A 输出

10：使能闹钟 B 输出

11：使能唤醒输出

位 20 **POL**：输出极性 (Output polarity)

该位用于配置 TAMPALRM 输出的极性。

0：当 ALRAF/ALRBF/WUTF 置 1 时（取决于 OSEL[1:0]），或者当 TAMPxF/ITAMPxF 置 1 时（如果 TAMPOE = 1），该引脚为高电平。

1：当 ALRAF/ALRBF/WUTF 置 1 时（取决于 OSEL[1:0]），或者当 TAMPxF/ITAMPxF 置 1 时（如果 TAMPOE = 1），该引脚为低电平。

位 19 **COSEL**：校准输出选择 (Calibration output selection)

当 COE = 1 时，该位可选择 CALIB 上输出的信号。

0：校准输出为 512 Hz

1：校准输出为 1 Hz

在 RTCCLK 为 32.768 kHz 且预分频器为其默认值（PREDIV_A = 127 且 PREDIV_S = 255）的条件下，这些频率有效。请参见*第 32.3.17 节：校准时钟输出*。

位 18 **BKP**：备份 (Backup)

用户可对该位执行写操作以记录是否已对夏令时进行更改。

位 17 **SUB1H**：减少 1 小时（冬季时间更改）(Subtract 1 hour (winter time change))

当该位在初始化模式以外的模式下置 1 时，如果当前小时不是 0，则日历时间将减少 1 小时。该位始终读为 0。

当前小时为 0 时，将该位置 1 没有任何作用。

0：无影响

1：将当前时间减少 1 小时。这可用于冬季时间更改。

位 16 **ADD1H**：增加 1 小时（夏季时间更改）(Add 1 hour (summer time change))

当该位在初始化模式以外的模式下置 1 时，日历时间将增加 1 小时。该位始终读为 0。

0：无影响

1：将当前时间增加 1 小时。这可用于夏季时间更改

位 15 **TSIE**：时间戳中断使能 (Timestamp interrupt enable)

0：禁止时间戳中断

1：使能时间戳中断

位 14 **WUTIE**：唤醒定时器中断使能 (Wake-up timer interrupt enable)
0：禁止唤醒定时器中断
1：使能唤醒定时器中断

位 13 **ALRBIE**：闹钟 B 中断使能 (Alarm B interrupt enable)
0：禁止闹钟 B 中断
1：使能闹钟 B 中断

位 12 **ALRAIE**：闹钟 A 中断使能 (Alarm A interrupt enable)
0：禁止闹钟 A 中断
1：使能闹钟 A 中断

位 11 **TSE**：时间戳使能 (timestamp enable)
0：禁止时间戳
1：使能时间戳

位 10 **WUTE**：唤醒定时器使能 (Wake-up timer enable)
0：禁止唤醒定时器
1：使能唤醒定时器
注意：唤醒定时器禁止时，需要等到 WUTWF = 1 后才能重新使能。

位 9 **ALRBE**：闹钟 B 使能 (Alarm B enable)
0：禁止闹钟 B
1：使能闹钟 B

位 8 **ALRAE**：闹钟 A 使能 (Alarm A enable)
0：禁止闹钟 A
1：使能闹钟 A

位 7 **SSRUIE**：SSR 下溢中断使能 (SSR underflow interrupt enable)
0：禁止 SSR 下溢中断
1：使能 SSR 下溢中断

位 6 **FMT**：小时格式 (Hour format)
0：24 小时/天格式
1：AM/PM 小时格式

位 5 **BYPSHAD**：旁路影子寄存器 (Bypass the shadow registers)
0：日历值（从 RTC_SSR、RTC_TR 和 RTC_DR 读取时）取自影子寄存器，该影子寄存器每两个 RTCCLK 周期更新一次。
1：日历值（从 RTC_SSR、RTC_TR 和 RTC_DR 读取时）直接取自日历计数器。
注意：如果 APB 时钟的频率低于 7 倍的 RTCCLK 频率，则必须将 BYPSHAD 置 1。

位 4 **REFCKON**：RTC_REFIN 参考时钟检测使能（50 Hz 或 60 Hz）(RTC_REFIN Reference clock detection enable (50 or 60 Hz))
0：禁止 RTC_REFIN 检测
1：使能 RTC_REFIN 检测
注意：BIN 必须为 0x00 并且 PREDIV_S 必须为 0x00FF。

位 3 **TSEDGE**：时间戳事件有效边沿 (Timestamp event active edge)
0：RTC_TS 输入上升沿生成时间戳事件
1：RTC_TS 输入下降沿生成时间戳事件
TSEDGE 发生更改时，必须复位 TSE 以避免将 TSF 意外置 1。

位 2:0 **WUCKSEL[2:0]**：ck_wut 唤醒时钟选择 (ck_wut wake-up clock selection)

000：选择 RTC/16 时钟

001：选择 RTC/8 时钟

010：选择 RTC/4 时钟

011：选择 RTC/2 时钟

10x：在 BCD 模式下，选择 ck_spre 时钟（通常为 1 Hz）。在二进制模式或混合模式下，则为 BCDU 选择的时钟。

11x：在 BCD 模式下，选择 ck_spre 时钟（通常为 1 Hz）。在二进制模式或混合模式下，则为 BCDU 选择的时钟。此外，将 2^{16} 加到 WUT 计数器值中。

注意： *只能在初始化模式下 (RTC_ICSR/INITF = 1) 对该寄存器的位 6 和 4 执行写操作。*

WUT = 唤醒单元计数器值。当 WUCKSEL[2:1 = 11] 时，WUT = (0x0000 到 0xFFFF) + 0x10000（增加的值）。

只能在 RTC_CR WUTE 位 = 0 且 RTC_ICSR WUTWF 位 = 1 时对该寄存器的位 2 到 0 执行写操作。

建议不要在日历小时递增时更改小时，因为这样做可能会屏蔽日历小时的增量。

ADD1H 和 SUB1H 的更改在下一秒生效。

32.6.8 RTC 特权模式控制寄存器 (RTC_PRIVCFGR)

RTC privilege mode control register

该寄存器只能通过 APB 特权访问进行写操作。

偏移地址：0x1C

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRIV	INIT PRIV	CAL PRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS PRIV	WUT PRIV	ALRB PRIV	ALRA PRIV
rw	rw	rw										rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15 **PRIV**：RTC 特权保护 (RTC privilege protection)

0：所有 RTC 寄存器都可以通过 APB 特权或非特权访问进行写操作（由其他特权保护位保护的寄存器除外）。

1：所有 RTC 寄存器都只能通过 APB 特权访问进行写操作。

位 14 **INITPRIV**：初始化特权保护 (Initialization privilege protection)

0：在 RTC 初始化模式下，日历寄存器和预分频器寄存器可以通过 APB 特权或非特权访问进行写操作。

1：在 RTC 初始化模式下，日历寄存器和预分频器寄存器只能通过 APB 特权访问进行写操作。

位 13 **CALPRIV**：移位寄存器、夏令时存储、校准和参考时钟特权保护 (Shift register, Delight saving, calibration and reference clock privilege protection)

0：移位寄存器、夏令时存储、校准和参考时钟可以通过 APB 特权或非特权访问进行写操作。

1：移位寄存器、夏令时存储、校准和参考时钟只能通过 APB 特权访问进行写操作。

位 12:4 保留，必须保持复位值。

位 3 **TSPRIV**：时间戳特权保护 (Timestamp privilege protection)

0：RTC 时间戳配置和中断清除方法可以通过 APB 特权或非特权访问进行写操作。

1：RTC 时间戳配置和中断清除方法只能通过 APB 特权访问进行写操作。

位 2 **WUTPRIV**：唤醒定时器特权保护 (Wake-up timer privilege protection)

0：RTC 唤醒定时器配置和中断清除方法可以通过 APB 特权或非特权访问进行写操作。

1：RTC 唤醒定时器配置和中断清除方法只能通过 APB 特权访问进行写操作。

位 1 **ALRBPRIV**：闹钟 B 特权保护 (Alarm B privilege protection)

0：RTC 闹钟 B 配置和中断清除方法可以通过 APB 特权或非特权访问进行写操作。

1：RTC 闹钟 B 配置和中断清除方法只能通过 APB 特权访问进行写操作。

位 0 **ALRAPRIV**：闹钟 A 和 SSR 下溢特权保护 (Alarm A and SSR underflow privilege protection)

0：RTC 闹钟 A 和 SSR 下溢配置以及中断清除方法可以通过 APB 特权或非特权访问进行写操作。

1：RTC 闹钟 A 和 SSR 下溢配置以及中断清除方法只能通过 APB 特权访问进行写操作。

注意： *有关读保护的详细信息，请参见第 32.3.4 节：RTC 特权保护模式。*

既可以为该寄存器设置全局写保护，也可以为该寄存器的每个位单独设置写保护，以防其遭到非特权访问，具体取决于 RTC_PRIVCFGR 配置（请参见*第 32.3.4 节：RTC 特权保护模式*）。

32.6.9 RTC 写保护寄存器 (RTC_WPR)

RTC write protection register

偏移地址：0x24

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **KEY[7:0]**：写保护关键字 (Write protection key)

可通过软件对该字节执行写操作。

读取该字节时，始终返回 0x00。

有关如何解锁 RTC 寄存器写保护的介绍，请参见 *RTC 寄存器写保护*。

32.6.10 RTC 校准寄存器 (RTC_CALR)

RTC calibration register

此寄存器受写保护。*第 1015 页的RTC 寄存器写保护*中介绍了写访问的过程。

可以对该寄存器进行写保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x28

备份域复位值：0x0000 0000

系统复位：不受影响

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CALP	CALW8	CALW 16	LPCAL	Res.	Res.	Res.	CALM[8:0]								
rw	rw	rw	rw				rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15 **CALP**：将 RTC 的频率增加 488.5 ppm (Increase frequency of RTC by 488.5 ppm)

0：不增加 RTCCLK 脉冲。

1：每 2^{11} 个脉冲有效插入一个 RTCCLK 脉冲（将频率增加 488.5 ppm）。

此功能应与 CALM 结合使用，后者在高分辨率下会降低日历的频率。如果输入频率为 32768 Hz，则在 32 秒窗口中增加的 RTCCLK 脉冲数按如下公式计算：(512 × CALP) - CALM。

请参见*第 32.3.15 节：RTC 精密数字校准*。

位 14 **CALW8**：使用 8 秒校准周期 (Use an 8-second calibration cycle period)

当 CALW8 置 1 时，选择 8 秒校准周期。

注意：当 CALW8 = 1 时，CALM[1:0] 将始终保持为 00。请参见第 32.3.15 节：RTC 精密数字校准。

位 13 **CALW16**：使用 16 秒校准周期 (Use a 16-second calibration cycle period)

当 CALW16 置 1 时，选择 16 秒校准周期。如果 CALW8 = 1，则该位不可以置 1。

注意：当 CALW16= 1 时，CALM[0] 将始终保持为 0。请参见第 32.3.15 节：RTC 精密数字校准。

位 12 **LPCAL**：RTC 低功耗模式 (RTC low-power mode)

0：校准窗口为 2^{20} 个 RTCCLK 脉冲，这是一种高功耗模式。只有在需要小于 32 秒的校准窗口时，才可以设置该模式。

1：校准窗口为 2^{20} 个 ck_apre 周期，这是超低功耗模式所需的配置。

位 11:9 保留，必须保持复位值。

位 8:0 **CALM[8:0]**：负校准 (Calibration minus)

在 2^{20} 个 RTCCLK 脉冲内屏蔽 CALM 个脉冲（如果输入频率为 32768 Hz，则为 32 秒）来降低日历的频率。其分辨率为 0.9537 ppm。

要提高日历的频率，则应将此功能与 CALP 结合使用。请参见*第 1019 页的第 32.3.15 节：RTC 精密数字校准*。

32.6.11 RTC 平移控制寄存器 (RTC_SHIFTR)

RTC shift control register

此寄存器受写保护。*第 1015 页的RTC 寄存器写保护*中介绍了写访问的过程。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x2C

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD1S	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
w															

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

位 31 **ADD1S**：增加一秒钟 (Add one second)

0：无影响

1：对时钟/日历增加一秒钟

该位为只写位且始终读为 0。当平移操作挂起（RTC_ICSR 中的 SHPF = 1）时，对该位执行写操作无作用。

此函数应与 SUBFS 配合使用（请参见下文介绍），以便有效地向原子操作机制的时钟添加亚秒值。

位 30:15 保留，必须保持复位值。

位 14:0 **SUBFS[14:0]**：减少亚秒值 (Subtract a fraction of a second)

这些位为只写位且始终读为 0。当平移操作挂起（RTC_ICSR 中的 SHPF = 1）时，对该位执行写操作无作用。

写入 SUBFS 的值将加到同步预分频器计数器中。由于该计数器递减计数，此操作可有效地从时钟减去（延迟）以下时间：

延迟（秒）= SUBFS / (PREDIV_S + 1)

当 ADD1S 函数与 SUBFS 结合使用时，可有效地将亚秒值增加到时钟（提前时钟），使时钟提前以下时间：

提前（秒）= (1 - (SUBFS / (PREDIV_S + 1)))。

在混合 BCD 二进制模式下（BIN=10 或 11），SUBFS[14:BCDU+8] 必须写入 0。

注意：对 SUBFS 执行写操作将使 RSF 清零。软件随后会等待至 RSF = 1 以确定影子寄存器已更新为平移后的时间。

32.6.12 RTC 时间戳时间寄存器 (RTC_TSTR)

RTC timestamp time register

仅当 RTC_SR 中的 TSF 置 1 时，该寄存器的内容才有效。当 TSF 位复位时，清零该寄存器。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x30

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PM	HT[1:0]		HU[3:0]			
									r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	MNT[2:0]			MNU[3:0]				Res.	ST[2:0]			SU[3:0]			
	r	r	r	r	r	r	r		r	r	r	r	r	r	r

位 31:23 保留，必须保持复位值。

位 22 **PM**：AM/PM 符号 (AM/PM notation)
0：AM 或 24 小时制
1：PM

位 21:20 **HT[1:0]**：小时的十位（BCD 格式）(Hour tens in BCD format)。

位 19:16 **HU[3:0]**：小时的个位（BCD 格式）(Hour units in BCD format)。

位 15 保留，必须保持复位值。

位 14:12 **MNT[2:0]**：分钟的十位（BCD 格式）(Minute tens in BCD format)。

位 11:8 **MNU[3:0]**：分钟的个位（BCD 格式）(Minute units in BCD format)。

位 7 保留，必须保持复位值。

位 6:4 **ST[2:0]**：秒的十位（BCD 格式）(Second tens in BCD format)。

位 3:0 **SU[3:0]**：秒的个位（BCD 格式）(Second units in BCD format)。

32.6.13 RTC 时间戳日期寄存器 (RTC_TSDR)

RTC timestamp date register

仅当 RTC_SR 中的 TSF 置 1 时，该寄存器的内容才有效。当 TSF 位复位时，清零该寄存器。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x34

备份域复位值：0x0000 0000

系统复位：不受影响

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDU[2:0]			MT	MU[3:0]				Res.	Res.	DT[1:0]		DU[3:0]			
r	r	r	r	r	r	r	r			r	r	r	r	r	r

位 31:16 保留，必须保持复位值。

位 15:13 **WDU[2:0]**：星期几的个位 (Week day units)

位 12 **MT**：月份的十位（BCD 格式）(Month tens in BCD format)

位 11:8 **MU[3:0]**：月份的个位（BCD 格式）(Month units in BCD format)

位 7:6 保留，必须保持复位值。

位 5:4 **DT[1:0]**：日期的十位（BCD 格式）(Date tens in BCD format)

位 3:0 **DU[3:0]**：日期的个位（BCD 格式）(Date units in BCD format)

32.6.14 RTC 时间戳亚秒寄存器 (RTC_TSSSR)

RTC timestamp subsecond register

仅当 RTC_SR 中的 TSF 置 1 时，该寄存器的内容才有效。当 TSF 位复位时，清零该寄存器。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x38

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SS[31:16]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

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

位 31:0 SS[31:0]：亚秒值/同步二进制计数器值 (Subsecond value/synchronous binary counter values)

当发生时间戳事件时，SS[31:0] 是同步预分频器计数器的值。

32.6.15 RTC 闹钟 A 寄存器 (RTC_ALRMAR)

RTC alarm A register

仅当 RTC_CR 中的 ALRAE 复位时或在初始化模式下，才可以对该寄存器执行写操作。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x40

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSK4	WDSEL	DT[1:0]		DU[3:0]				MSK3	PM	HT[1:0]		HU[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **MSK4**：闹钟 A 日期掩码 (Alarm A date mask)
0：如果日期/日匹配，则闹钟 A 置 1
1：在闹钟 A 比较中，日期/日无关

位 30 **WDSEL**：星期几选择 (Week day selection)
0：DU[3:0] 代表日期的个位
1：DU[3:0] 代表星期几。DT[1:0] 为无关位。

位 29:28 **DT[1:0]**：日期的十位（BCD 格式）(Date tens in BCD format)

位 27:24 **DU[3:0]**：日期个位或日（BCD 格式）(Date units or day in BCD format)

位 23 **MSK3**：闹钟 A 小时掩码 (Alarm A hours mask)
0：如果小时匹配，则闹钟 A 置 1
1：在闹钟 A 比较中，小时无关

位 22 **PM**：AM/PM 符号 (AM/PM notation)
0：AM 或 24 小时制
1：PM

位 21:20 **HT[1:0]**：小时的十位（BCD 格式）(Hour tens in BCD format)

位 19:16 **HU[3:0]**：小时的个位（BCD 格式）(Hour units in BCD format)

位 15 **MSK2**：闹钟 A 分钟掩码 (Alarm A minutes mask)
0：如果分钟匹配，则闹钟 A 置 1
1：在闹钟 A 比较中，分钟无关

位 14:12 **MNT[2:0]**：分钟的十位（BCD 格式）(Minute tens in BCD format)

位 11:8 **MNU[3:0]**：分钟的个位（BCD 格式）(Minute units in BCD format)

位 7 **MSK1**：闹钟 A 秒掩码 (Alarm A seconds mask)
0：如果秒匹配，则闹钟 A 置 1
1：在闹钟 A 比较中，秒无关

位 6:4 **ST[2:0]**：秒的十位（BCD 格式）(Second tens in BCD format)。

位 3:0 **SU[3:0]**：秒的个位（BCD 格式）(Second units in BCD format)。

32.6.16 RTC 闹钟 A 亚秒寄存器 (RTC_ALRMASSR)

RTC alarm A subsecond register

仅当 RTC_CR 中的 ALRAE 复位时或在初始化模式下，才可以对该寄存器执行写操作。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x44

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31 **SSCLR**：闹铃时清零同步计数器（仅限二进制模式）(Clear synchronous counter on alarm (Binary mode only))

0：同步二进制计数器（RTC_SSR 中的 SS[31:0]）处于自由运行状态。

1：同步二进制计数器（RTC_SSR 中的 SS[31:0]）从 0xFFFF FFFF 计数到 RTC_ALRABINR.SS[31:0] 值，并在计数到 RTC_ALRABINR.SS[31:0] 的一个 ck_apre 周期后自动重载为 0xFFFF FFFF。

注意：使用 BCD 模式或混合模式时（BIN = 00、10 或 11），SSCLR 必须保持为 0。

位 30 保留，必须保持复位值。

位 29:24 **MASKSS[5:0]**：屏蔽从该位开始的最高有效位 (Mask the most-significant bits starting at this bit)

0：不对闹钟 A 的亚秒进行比较。当秒单元递增时设置闹钟（假定其余位域均匹配）。

1：在闹钟 A 比较中，SS[31:1] 为无关位。仅比较 SS[0]。

2：在闹钟 A 比较中，SS[31:2] 为无关位。仅比较 SS[1:0]。

...

31：在闹钟 A 比较中，SS[31] 为无关位。仅比较 SS[30:0]。

从 32 到 63：所有 32 个 SS 位均进行比较，并且必须全部匹配才能激活闹钟。

注意：在 BCD 模式下 (BIN=00)，同步计数器的上溢位（位 31:15）从不进行比较。仅当执行平移操作之后，这些位才不为 0。

位 23:15 保留，必须保持复位值。

位 14:0 **SS[14:0]**：亚秒值 (Subseconds value)

该值与同步预分频器计数器的内容进行比较以确定是否要激活闹钟 A。仅比较位 0 到 MASKSS-1。

该位域是 RTC_ALRABINR 中 SS[14:0] 的镜像，因此也可以通过 RTC_ALRABINR 对其进行读/写操作。

注意：如果在 TAMP_ATCR1 中的 ATCKSEL[3] = 1 时将 SSCLR 置 1，则 SS[3:0] 必须为 0000。

32.6.17 RTC 闹钟 B 寄存器 (RTC_ALRMBR)

RTC alarm B register

仅当 RTC_CR 中的 ALRBE 复位时或在初始化模式下，才可以对该寄存器执行写操作。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x48

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSK4	WD SEL	DT[1:0]		DU[3:0]				MSK3	PM	HT[1:0]		HU[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **MSK4**：闹钟 B 日期掩码 (Alarm B date mask)
0：如果日期和日匹配，则闹钟 B 置 1
1：在闹钟 B 比较中，日期和日无关

位 30 **WDSEL**：星期几选择 (Week day selection)
0：DU[3:0] 代表日期的个位
1：DU[3:0] 代表星期几。DT[1:0] 为无关位。

位 29:28 **DT[1:0]**：日期的十位（BCD 格式）(Date tens in BCD format)

位 27:24 **DU[3:0]**：日期个位或日（BCD 格式）(Date units or day in BCD format)

位 23 **MSK3**：闹钟 B 小时掩码 (Alarm B hours mask)
0：如果小时匹配，则闹钟 B 置 1
1：在闹钟 B 比较中，小时无关

位 22 **PM**：AM/PM 符号 (AM/PM notation)
0：AM 或 24 小时制
1：PM

位 21:20 **HT[1:0]**：小时的十位（BCD 格式）(Hour tens in BCD format)

位 19:16 **HU[3:0]**：小时的个位（BCD 格式）(Hour units in BCD format)

位 15 **MSK2**：闹钟 B 分钟掩码 (Alarm B minutes mask)
0：如果分钟匹配，则闹钟 B 置 1
1：在闹钟 B 比较中，分钟无关

位 14:12 **MNT[2:0]**：分钟的十位（BCD 格式）(Minute tens in BCD format)

位 11:8 **MNU[3:0]**：分钟的个位（BCD 格式）(Minute units in BCD format)

位 7 **MSK1**：闹钟 B 秒掩码 (Alarm B seconds mask)
0：如果秒匹配，则闹钟 B 置 1
1：在闹钟 B 比较中，秒无关

位 6:4 **ST[2:0]**：秒的十位（BCD 格式）(Second tens in BCD format)

位 3:0 **SU[3:0]**：秒的个位（BCD 格式）(Second units in BCD format)

32.6.18 RTC 闹钟 B 亚秒寄存器 (RTC_ALRMBSSR)

RTC alarm B subsecond register

仅当 RTC_CR 中的 ALRBE 复位时或在初始化模式下，才可以对该寄存器执行写操作。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x4C

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31 **SSCLR**：闹铃时清零同步计数器（仅限二进制模式）(Clear synchronous counter on alarm (Binary mode only))

0：同步二进制计数器（RTC_SSR 中的 SS[31:0]）处于自由运行状态。

1：同步二进制计数器（RTC_SSR 中的 SS[31:0]）从 0xFFFF FFFF 计数到 RTC_ALRBBINR.SS[31:0] 值，并在计数到 RTC_ALRBBINR.SS[31:0] 的一个 ck_apre 周期后自动重载为 0xFFFF FFFF。

注意：使用 BCD 模式或混合模式时（BIN = 00、10 或 11），SSCLR 必须保持为 0。

位 30 保留，必须保持复位值。

位 29:24 **MASKSS[5:0]**：屏蔽从该位开始的最高有效位 (Mask the most-significant bits starting at this bit)

0：不对闹钟 B 的亚秒进行比较。当秒单元递增时设置闹钟（假定其余位域均匹配）。

1：在闹钟 B 比较中，SS[31:1] 为无关位。仅比较 SS[0]。

2：在闹钟 B 比较中，SS[31:2] 为无关位。仅比较 SS[1:0]。

...

31：在闹钟 B 比较中，SS[31] 为无关位。仅比较 SS[30:0]。

从 32 到 63：所有 32 个 SS 位均进行比较，并且必须全部匹配才能激活闹钟。

注意：在 BCD 模式下 (BIN=00)，同步计数器的上溢位（位 15）从不进行比较。仅当执行平移操作之后，该位才不为 0。

位 23:15 保留，必须保持复位值。

位 14:0 **SS[14:0]**：亚秒值 (Subseconds value)

该值与同步预分频器计数器的内容进行比较以确定是否要激活闹钟 B。仅比较位 0 到 MASKSS-1。

该位域是 RTC_ALRBBINR 中 SS[14:0] 的镜像，因此也可以通过 RTC_ALRBBINR 对其进行读/写操作。

注意：如果在 TAMP_ATCR1 中的 ATCKSEL[3] = 1 时将 SSCLR 置 1，则 SS[3:0] 必须为 0000。

32.6.19 RTC 状态寄存器 (RTC_SR)

RTC status register

既可以为该寄存器设置全局保护，也可以为该寄存器的每个位单独设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x50

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SSRUF	ITSF	TSOVF	TSF	WUTF	ALRBF	ALRAF
									r	r	r	r	r	r	r

位 31:7 保留，必须保持复位值。

位 6 **SSRUF**：SSR 下溢标志 (SSR underflow flag)

当 SSR 到达 0 以下时，由硬件将该标志置 1。当 SSCLR=1 时，SSRUF 不会置 1。

位 5 **ITSF**：内部时间戳标志 (Internal timestamp flag)

发生内部时间戳事件时，由硬件将该标志置 1。

位 4 **TSOVF**：时间戳上溢标志 (Timestamp overflow flag)

当在 TSF 已置 1 的情况下发生时间戳事件时，由硬件将此标志置 1。

建议仅在 TSF 位清零之后再检查并清零 TSOVF 位。否则，如果时间戳事件恰好在清零 TSF 位之前刚刚发生，则上溢事件可能会被漏掉。

位 3 **TSF**：时间戳标志 (Timestamp flag)

发生时间戳事件时，由硬件将此标志置 1。

如果 ITSF 标志已置 1，则必须将 TSF 与 ITSF 一起清零。

注意：如果 TAMPTS = 1，并在入侵事件后的 3 个 ck_apre 周期内读取入侵标志，则 TSF 不会置 1。更多详细信息，请参见入侵事件的时间戳。

位 2 **WUTF**：唤醒定时器标志 (Wake-up timer flag)

当唤醒自动重载计数器计数到 0 时，由硬件将该标志置 1。

如果 WUTOCLR[15:0] 不等于 0x0000，则当唤醒自动重载计数器计数到 WUTOCLR 值时，WUTF 由硬件清零。

如果 WUTOCLR[15:0] 等于 0x0000，则 WUTF 必须由软件清零。

软件必须在 WUTF 再次置 1 的 1.5 个 RTCCLK 周期之前将该标志清零。

位 1 **ALRBF**：闹钟 B 标志 (Alarm B flag)

当时间/日期寄存器（RTC_TR 和 RTC_DR）与闹钟 B 寄存器 (RTC_ALRMBR) 匹配时，由硬件将该标志置 1。

位 0 **ALRAF**：闹钟 A 标志 (Alarm A flag)

当时间/日期寄存器（RTC_TR 和 RTC_DR）与闹钟 A 寄存器 (RTC_ALRMAR) 匹配时，由硬件将该标志置 1。

注意：该寄存器的位在 RTC_SCR 寄存器中相应的清零位置 1 后的 2 个 APB 时钟周期之后清零。

32.6.20 RTC 屏蔽中断状态寄存器 (RTC_MISR)

RTC masked interrupt status register

既可以为该寄存器设置全局保护，也可以为该寄存器的每个位单独设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x54

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SSRUMF	ITSMF	TSOVMF	TSMF	WUTMF	ALRBMF	ALRAMF
									r	r	r	r	r	r	r

位 31:7 保留，必须保持复位值。

位 6 **SSRUMF**：SSR 下溢屏蔽标志 (SSR underflow masked flag)
发生 SSR 下溢中断时，由硬件将该标志置 1。

位 5 **ITSMF**：内部时间戳屏蔽标志 (Internal timestamp masked flag)
当发生内部时间戳事件并产生时间戳中断时，由硬件将该标志置 1。

位 4 **TSOVMF**：时间戳上溢屏蔽标志 (Timestamp overflow masked flag)
当在 TSMF 已置 1 的情况下发生时间戳中断时，由硬件将此标志置 1。
建议仅在 TSF 位清零之后再检查并清零 TSOVF 位。否则，如果时间戳事件恰好在清零 TSF 位之前刚刚发生，则上溢事件可能会被漏掉。

位 3 **TSMF**：时间戳屏蔽标志 (Timestamp masked flag)
发生时间戳中断时，由硬件将此标志置 1。
如果 ITSF 标志已置 1，则必须将 TSF 与 ITSF 一起清零。

位 2 **WUTMF**：唤醒定时器屏蔽标志 (Wake-up timer masked flag)
发生唤醒定时器中断时，由硬件将该标志置 1。
软件必须在 WUTF 再次置 1 的 1.5 个 RTCCLK 周期之前将该标志清零。

位 1 **ALRBMF**：闹钟 B 屏蔽标志 (Alarm B masked flag)
发生闹钟 B 中断时，由硬件将该标志置 1。

位 0 **ALRAMF**：闹钟 A 屏蔽标志 (Alarm A masked flag)
发生闹钟 A 中断时，由硬件将此标志置 1。

既可以为该寄存器设置全局保护，也可以为该寄存器的每个位单独设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

32.6.21 RTC 状态清零寄存器 (RTC_SCR)

RTC status clear register

既可以为该寄存器设置全局保护，也可以为该寄存器的每个位单独设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x5C

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSSRUF	CITSF	CTSOVF	CTSF	CWUTF	CALRBF	CALRAF
									w	w	w	w	w	w	w

位 31:7 保留，必须保持复位值。

位 6 **CSSRUF**：清零 SSR 下溢标志 (Clear SSR underflow flag)
向该位写入“1”可清零 RTC_SR 寄存器中的 SSRUF 位。

位 5 **CITSF**：清零内部时间戳标志 (Clear internal timestamp flag)
向该位写入 1 可清零 RTC_SR 寄存器中的 ITSF 位。

位 4 **CTSOVF**：清零时间戳上溢标志 (Clear timestamp overflow flag)
向该位写入 1 可清零 RTC_SR 寄存器中的 TSOVF 位。
建议仅在 TSF 位清零之后再检查并清零 TSOVF 位。否则，如果时间戳事件恰好在清零 TSF 位之前刚刚发生，则上溢事件可能会被漏掉。

位 3 **CTSF**：清零时间戳标志 (Clear timestamp flag)
向该位写入 1 可清零 RTC_SR 寄存器中的 TSF 位。
如果 ITSF 标志已置 1，则必须通过将 CRSF 和 CITSF 置 1 来将 TSF 与 ITSF 一起清零。

位 2 **CWUTF**：清零唤醒定时器标志 (Clear wake-up timer flag)
向该位写入 1 可清零 RTC_SR 寄存器中的 WUTF 位。

位 1 **CALRBF**：清零闹钟 B 标志 (Clear alarm B flag)
向该位写入 1 可清零 RTC_SR 寄存器中的 ALRBF 位。

位 0 **CALRAF**：清零闹钟 A 标志 (Clear alarm A flag)
向该位写入 1 可清零 RTC_SR 寄存器中的 ALRAF 位。

32.6.22 RTC 闹钟 A 二进制模式寄存器 (RTC_ALRABINR)

RTC alarm A binary mode register

仅当 RTC_CR 中的 ALRAE 复位时或在初始化模式下，才可以对该寄存器执行写操作。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x70

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31:0 **SS[31:0]**：二进制模式下的同步计数器闹钟值 (Synchronous counter alarm value in Binary mode)

该值与同步计数器的内容进行比较以确定是否要激活闹钟 A。仅比较位 0 到 MASKSS-1。SS[14:0] 是 RTC_ALRMASSRR 中 SS[14:0] 的镜像，因此也可以通过 RTC_ALRMASSR 对其进行读/写操作。

注意：如果在 TAMP_ATCR1 中的 ATCKSEL[3] = 1 时将 SSCLR 置 1，则 SS[3:0] 必须为 0000。

32.6.23 RTC 闹钟 B 二进制模式寄存器 (RTC_ALRBBINR)

RTC alarm B binary mode register

仅当 RTC_CR 中的 ALRBE 复位时或在初始化模式下，才可以对该寄存器执行写操作。

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 32.3.4 节：RTC 特权保护模式*。

偏移地址：0x74

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31:0 **SS[31:0]**：二进制模式下的同步计数器闹钟值 (Synchronous counter alarm value in Binary mode)

该值与同步计数器的内容进行比较以确定是否要激活闹钟 B。仅比较位 0 到 MASKSS-1。SS[14:0] 是 RTC_ALRMBSSRR 中 SS[14:0] 的镜像，因此也可以通过 RTC_ALRMBSSR 对其进行读/写操作。

注意：如果在 TAMP_ATCR1 中的 ATCKSEL[3] = 1 时将 SSCLR 置 1，则 SS[3:0] 必须为 0000。

32.6.24 RTC 寄存器映射

表 239. RTC 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	RTC_TR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PM	HT[1:0]		HU[3:0]				Res.	MNT[2:0]			MNU[3:0]				Res.	ST[2:0]			SU[3:0]			
	Reset value										0	0	0	0	0	0	0		0	0	0	0	0	0	0		0	0	0	0	0	0	0
0x04	RTC_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	YT[3:0]				YU[3:0]				WDU[2:0]			MT	MU[3:0]				Res.	Res.	DT[1:0]		DU[3:0]			
	Reset value									0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1			0	0	0	0	0	1
0x08	RTC_SSR	SS[31:16]																SS[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	RTC_ICSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RECALPF	Res.	Res.	Res.	BCDU[2:0]			BIN[1:0]		INIT	INITF	RSF	INITS	SHPF	WUT WF	Res.	Res.
	Reset value																0				0	0	0	0	0	0	0	0	0	0	1		
0x10	RTC_PRER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREDIV_A[6:0]							PREDIV_S[14:0]															
	Reset value										1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0x14	RTC_WUTR	WUTOCLR[15:0]																WUT[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x18	RTC_CR	OUT2EN	TAMPALRM_TYPE	TAMPALRM_PU	ALRBFCLR	ALRAFCLR	TAMPOE	TAMPTS	ITSE	COE	OSEL[1:0]		POL	COSEL	BKP	SUB1H	ADD1H	TSIE	WUTIE	ALRBIE	ALRAIE	TSE	WUTE	ALRBE	ALRAE	SSRUIE	FMT	BYPSHAD	REFCKON	TSEDGE	WUCKSEL[2:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	RTC_PRIVCFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIV	INITPRIV	CALPRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSPRIV	WUTPRIV	ALRBPRIV	ALRAPRIV
	Reset value																	0	0	0										0	0	0	0
0x24	RTC_WPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	KEY[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x28	RTC_ CALR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CALP	CALW8	CALW16	LPCAL	Res.	Res.	Res.	CALM[8:0]								
	Reset value																	0	0	0	0				0	0	0	0	0	0	0	0	0
0x2C	RTC_SHIFTR	ADD1S	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUBFS[14:0]														
	Reset value	0																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	RTC_TSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PM	HT[1:0]		HU[3:0]				Res.	MNT[2:0]			MNU[3:0]				Res.	ST[2:0]			SU[3:0]			
	Reset value										0	0	0	0	0	0	0		0	0	0	0	0	0	0		0	0	0	0	0	0	0

表 239. RTC 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x34	RTC_TSDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDU[2:0]			MT	MU[3:0]				Res.	Res.	DT [1:0]		DU[3:0]			
	Reset value																	0	0	0	0	0	0	0	0			0	0	0	0	0	0
0x38	RTC_TSSSR	SS[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	RTC_ALRMAR	MSK4	WDSEL	DT [1:0]		DU[3:0]				MSK3	PM	HT [1:0]		HU[3:0]				MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	RTC_ ALRMASSR	SSCLR	Res.	MASKSS [5:0]						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SS[14:0]														
	Reset value	0		0	0	0	0	0	0										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	RTC_ALRMBR	MSK4	WDSEL	DT [1:0]		DU[3:0]				MSK3	PM	HT [1:0]		HU[3:0]				MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	RTC_ ALRMBSSR	SSCLR	Res.	MASKSS [5:0]						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SS[14:0]														
	Reset value	0		0	0	0	0	0	0										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	RTC_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SSRUF	ITSF	TSOVF	TSF	WUTF	ALRBF	ALRAF
	Reset value																										0	0	0	0	0	0	0
0x54	RTC_MISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SSRUMF	ITSMF	TSOVMF	TSMF	WUTMF	ALRBMF	ALRAMF
	Reset value																										0	0	0	0	0	0	0
0x5C	RTC_SCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSSRUF	CITSF	CTSOVF	CTSF	CWUTF	CALRBF	CALRAF
	Reset value																										0	0	0	0	0	0	0
0x60	RTC_OR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OUT2_RMP
	Reset value																																0
0x70	RTC_ ALRABINR	SS[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x74	RTC_ ALRBBINR	SS[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 2.2 节*。

33 入侵和备份寄存器 (TAMP)

33.1 简介

防入侵检测电路用于保护敏感数据免受外部攻击。在所有低功耗模式和 V_{BAT} 模式下，32 个 32 位备份寄存器的内容均保持不变。该防入侵检测电路具有 2 个入侵引脚和 13 个内部入侵事件，可对备份寄存器以及器件中的其他机密信息实施保护。外部入侵引脚可配置为边沿检测、带或不带滤波的电平检测或主动入侵模式，该模式通过自动检查入侵引脚是否存在外部开路或短路的情况来提高安全性。

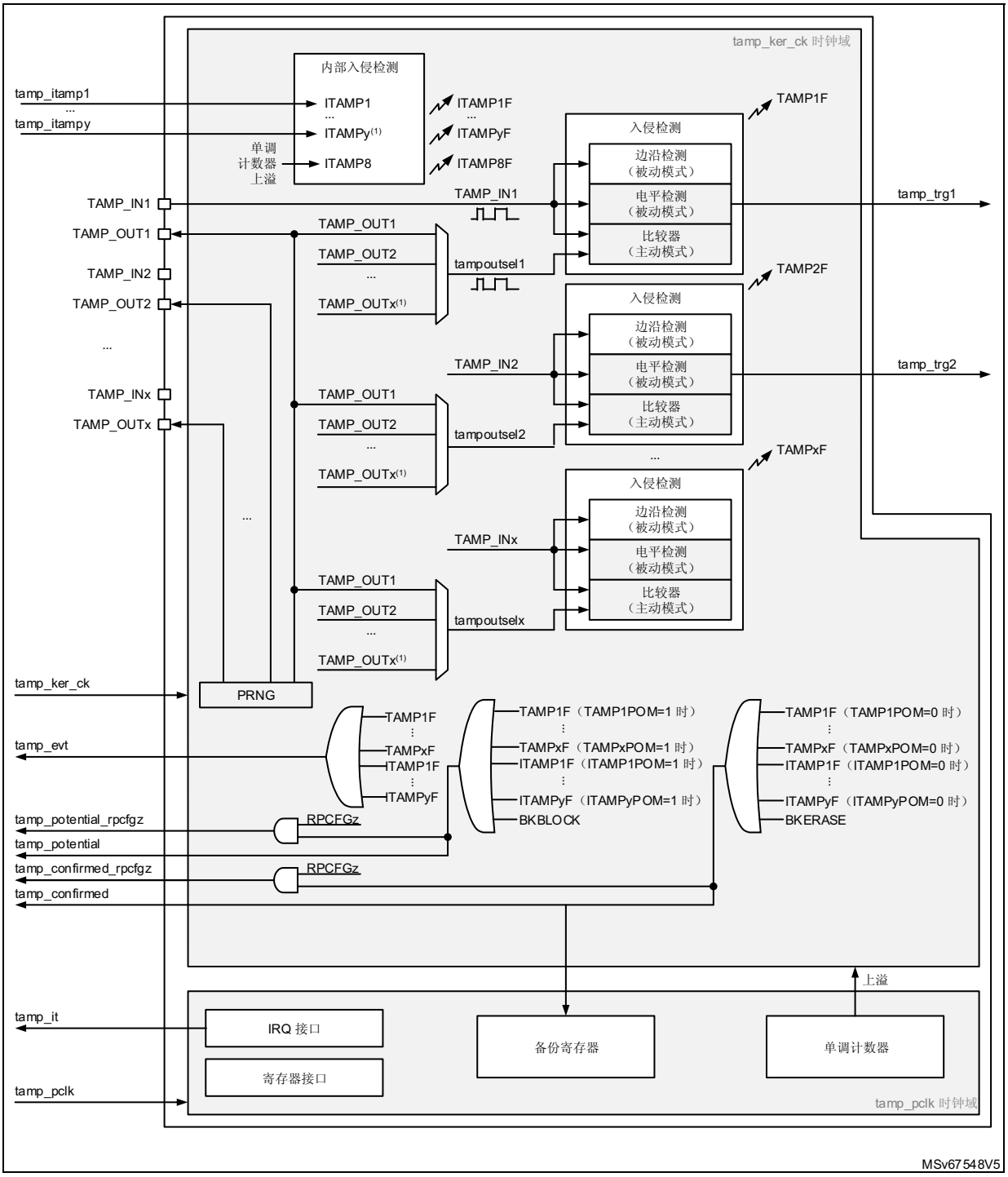
33.2 TAMP 主要特性

- 入侵检测可以选择擦除备份寄存器、备份SRAM、SRAM2 和缓存。受入侵保护的器件资源被称为“器件机密信息”。
- 32 个 32 位备份寄存器：
  - 备份寄存器 (TAMP_BKPxR) 在备份域中实现，可在 V_{DD} 电源关闭时通过 V_{BAT} 保持上电状态。
- 多达 2 个入侵引脚（针对 2 个外部入侵检测事件）：
  - 主动入侵模式：连续比较入侵输出与输入，以防止物理开路/短路攻击。
  - 被动入侵：超低功耗边沿或电平检测，支持内部上拉硬件管理。
  - 可配置数字滤波器。
- 13 个内部入侵事件，可防止瞬态或环境扰动攻击
- 每个入侵源都可以配置为两种模式：
  - 确认模式：检测到入侵后立即擦除机密信息（包括擦除备份寄存器）
  - 潜在模式：检测到入侵后擦除大多数机密信息（通过软件启动）
- 任何入侵检测都可以生成一个 RTC 时间戳事件。
- 入侵配置和备份寄存器特权保护
- 单调计数器。

33.3 TAMP 功能描述

33.3.1 TAMP 框图

图 369. TAMP 框图



1. 外部和内部入侵引脚的数量取决于产品。

33.3.2 TAMP 引脚和内部信号

表 240. TAMP 输入/输出引脚

引脚名称	信号类型	说明
TAMP_INx（x = 引脚索引）	输入	入侵输入引脚
TAMP_OUTx（x = 引脚索引）	输出	入侵输出引脚（仅限主动模式）

表 241. TAMP 内部输入/输出信号

内部信号名称	信号类型	说明
tamp_ker_ck	输入	TAMP 内核时钟，连接到 rtc_ker_ck，在本文档中也称为 RTCCLK
tamp_pclk	输入	TAMP APB 时钟，连接到 rtc_pclk
tamp_itamp[y]（y = 信号索引）	输入	内部入侵事件源
tamp_evt	输出	入侵事件检测标志（内部或外部入侵），无论采用确认模式配置还是潜在模式配置。
tamp_potential	输出	潜在入侵检测信号，用于器件机密信息(1)保护。 该信号在以下情况下有效： – 在潜在模式下生成入侵事件检测标志（内部或外部入侵）。 – 或者通过向 BKBLOCK 写入 1 来完成软件请求
tamp_confirmed	输出	确认入侵检测信号，用于器件机密信息(1)保护。 该信号在以下情况下有效： – 在确认模式下生成入侵事件检测标志（内部或外部入侵）。 – 或者通过向 BKERASE 写入 1 来完成软件请求
tamp_potential_rpcfgz（z = 信号索引）	输出	仅在 RPCFGz = 1 时生成的潜在入侵检测信号。 该信号在以下情况下有效： – 在潜在模式下生成入侵事件检测标志（内部或外部入侵）。 – 或者通过向 BKBLOCK 写入 1 来完成软件请求
tamp_confirmed_rpcfgz（z = 信号索引）	输出	仅在 RPCFGz = 1 时生成的确认入侵检测信号。 该信号在以下情况下有效： – 在确认模式下生成入侵事件检测标志（内部或外部入侵）。 – 或者通过向 BKERASE 写入 1 来完成软件请求
tamp_it	输出	TAMP 中断（有关详细信息，请参见*第 33.5 节：TAMP 中断*）
tamp_trg[x]（x = 信号索引）	输出	入侵检测触发信号

1. 请参见*表 242：TAMP 互连*。

TAMP 内核时钟通常是 32.768 kHz 的 LSE，但也可以在 RCC 中选择其他时钟源（更多详细信息，请参见 RCC）。一些检测模式在某些低功耗模式或 V_{BAT} 模式下不可用，具体取决于所选时钟（更多详细信息，请参见*第 33.4 节：TAMP 低功耗模式*）。

表 242. TAMP 互连

信号名称	源/目标
tamp_evt	rtc_tamp_evt 用于生成时间戳事件
tamp_potential	tamp_potential 信号用于阻止对下列器件机密信息的读写访问： – 备份寄存器 – SRAM2 tamp_potential 信号用于擦除下列器件机密信息： – ICACHE 的内容 正在执行擦除操作时，会阻止对器件机密信息的访问。
tamp_confirmed	tamp_confirmed 信号用于擦除下列器件机密信息： – 备份寄存器 – SRAM2 – ICACHE 的内容 正在执行擦除操作时，会阻止对器件机密信息的访问。
tamp_potential_rpcfg0	当 TAMP_RPCFGR 中的 RPCFG0 位置 1 时，tamp_potential_rpcfg0 信号用于阻止对下列器件机密信息的读写访问： – 备份 SRAM
tamp_confirmed_rpcfg0	当 TAMP_RPCFGR 中的 RPCFG0 位置 1 时，tamp_confirmed_rpcfg0 信号用于擦除下列器件机密信息： – 备份 SRAM 正在执行擦除操作时，会阻止对器件机密信息的访问。
tamp_itamp1	备份域电压阈值监视(1)
tamp_itamp2	温度监视(1)
tamp_itamp3	LSE 监视 (LSECSS)(2)
tamp_itamp4	HSE 监视 (rcc_hsecss_fail)
tamp_itamp5	RTC 日历上溢 (rtc_calovf)
tamp_itamp6	JTAG/SWD 访问（NVSTATE ≠ OPEN 时）
tamp_itamp7	ADC1 看门狗监视 1
tamp_itamp8(3)	单调计数器 1 上溢
tamp_itamp9	TRNG 故障
tamp_itamp11	入侵标志置 1 时产生的 IWDG 复位（潜在入侵超时）
tamp_itamp12	ADC1 模拟看门狗监视 2
tamp_itamp13	ADC1 模拟看门狗监视 3
tamp_itamp15	系统故障

1. 必须通过将 *PWR 备份域控制寄存器 (PWR_BDCR)* 中的 MONEN 置 1 来使能该监视功能。
2. 必须通过将 *PWR 备份域控制寄存器 (PWR_BDCR)* 中的 LSEON 置 1 来使能该监视功能。
3. 该信号在 TAMP 外设中生成。

33.3.3 RTC 和 TAMP 控制的 GPIO

请参见*第 32.3.3 节：RTC 和 TAMP 控制的 GPIO*。

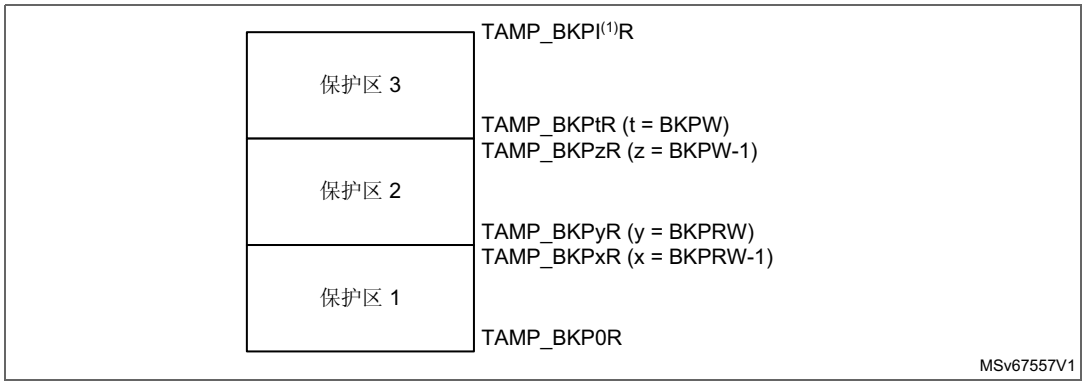
33.3.4 TAMP 寄存器写保护

系统复位后，可通过电源控制外设中的 DBP 位来保护 TAMP 寄存器（包括备份寄存器），以防止非正常的写访问（请参见 PWR 电源控制部分）。必须将 DBP 位置 1 才能使能 TAMP 寄存器的写访问。

33.3.5 备份寄存器保护区

备份寄存器保护通过 BKPRW[7:0] 和 BKPW[7:0] 进行配置（请参见下图）：

图 370. 备份寄存器保护区



1. I = 最后一个备份寄存器索引

33.3.6 TAMP 特权保护模式

默认情况下，备份域上电复位后，除了 TAMP 特权配置寄存器 (TAMP_PRIVCFGR) 只能在特权模式下写入之外，所有其他 TAMP 寄存器都可以在特权模式和非特权模式下读取或写入。TAMP 保护配置不受系统复位影响。

当 TAMP_PRIVCFGR 寄存器中的 TAMPPRIV 位置 1 时：

- 只能在特权模式下写入 TAMP 寄存器，但具有自己的保护设置的备份寄存器和单调计数器除外。
- 当 TAMP_PRIVCFGR 寄存器中的 CNT1PRIV 位置 1 时：TAMP_COUNT1R 只能在特权模式下读取和写入。
- 始终可以在特权模式和非特权模式下读取 TAMP_CFGR 和 TAMP_PRIVCFGR。所有其他 TAMP 寄存器都只能在特权模式下读取，但具有自己的保护设置的备份寄存器和单调计数器除外。

对于保护区 1，备份寄存器保护通过 BKPRW[7:0] 和 BKPRWPRIV 进行配置；对于保护区 2，备份寄存器保护通过 BKPRW[7:0]、BKPW[7:0] 和 BKPWPRIV 进行配置（请参见*图 370*）。

对受特权保护的寄存器的非特权访问将被拒绝：

- 不生成任何总线错误。
- 受写保护时，不会写入这些位。
- 受读保护时，这些位读为 0。

33.3.7 入侵检测

入侵检测的主要目的是保护器件机密信息免受外部攻击。检测的对象是 TAMP_INx（x = 引脚索引）I/O 上的事件，或通过内部监视器检测超范围的器件条件。

入侵检测可配置用于以下目的：

- 擦除备份寄存器和存储在 SRAM 或外设中的其他器件机密信息（如*表 242：TAMP 互连*所列）。器件机密信息列表可通过 *TAMP 资源保护配置寄存器 (TAMP_RPCFGR)* 进行配置。
- 阻止对备份寄存器和存储在 SRAM 或外设中的其他器件机密信息（如*表 242：TAMP 互连*所列）的读/写访问。器件机密信息列表可通过 TAMP_RPCFGR 进行配置。
- 生成中断，可使器件从低功耗模式唤醒
- 为低功耗定时器生成硬件触发信号，或生成 RTC 时间戳事件

外部 I/O 入侵检测支持 2 种主要配置：

- 被动模式：监视 TAMP_INx I/O，并基于边沿或电平执行入侵检测。
- 主动模式：连续比较 TAMP_INx（x = 引脚索引）与 TAMP_OUTy（y = 引脚索引），以实现开路/短路检测。

可以使用数字滤波器进行外部入侵检测，以避免发生误检。此外，可以将每个入侵源都配置为潜在模式，以避免硬件在入侵检测时启动机密信息擦除操作。经过软件检查后，可由软件启动机密信息擦除操作。

33.3.8 TAMP 备份寄存器和其他器件机密信息擦除

备份寄存器 (TAMP_BKPxR) 不会通过系统复位来复位，也不会在器件从待机模式唤醒时复位。

当相应的屏蔽位置 1（TAMP_CR2 寄存器中的 TAMPxMSK=1）时，备份寄存器和其他器件机密信息不会重置。

入侵检测——确认模式

当 TAMP_CR2 寄存器中的 TAMPxPOM = 0 时，为 TAMPx（外部入侵 x）选择确认模式。
当 TAMP_CR3 寄存器中的 ITAMPxPOM = 0 时，为 ITAMPx（内部入侵 x）选择确认模式。
*表 242：TAMP 互连*中的 tamp_confirmed 和 tamp_confirmed_rpcfgx 信号描述了确认模式下入侵检测的作用。

选择该模式后，可以在检测到入侵时自动擦除器件机密信息。

入侵检测——潜在模式

当 TAMP_CR2 寄存器中的 TAMPxPOM = 1 时，为 TAMPx（外部入侵 x）选择潜在模式。
当 TAMP_CR3 寄存器中的 ITAMPxPOM = 1 时，为 ITAMPx（内部入侵 x）选择潜在模式。
*表 242：TAMP 互连*中的 tamp_potential 和 tamp_potential_rpcfgx 信号描述了潜在模式下入侵检测的作用。

选择该模式后，可以避免在检测到入侵时不可逆地擦除某些器件机密信息。在该模式下，当检测到相应的入侵事件时，一些器件机密信息不会被擦除。此外，在潜在模式下，只要入侵检测标志置 1，就会阻止对这些器件机密信息的读写访问，直到通过将 TAMP_SCR 寄存器中的相应清零标志置 1 的方式将该标志清零。因此，软件可以执行一些检查来区分虚假入侵与真实入侵，从而能够在潜在入侵被确认为真实入侵的情况下启动机密信息擦除操作。可通过软件将 TAMP_CR2 寄存器中的 BKERASE 位置 1 来擦除器件机密信息。

潜在入侵切换为确认入侵时的超时

对于一些内部入侵，如果在其他入侵标志置 1 时发生独立看门狗复位，则将生成入侵事件（请参见 *表 242：TAMP 互连*）。IWDG 入侵必须在 ITAMPxPOM = 0 的情况下配置。这样，当上一个入侵事件处于潜在模式时，可以通过硬件在超时后强制擦除器件机密信息。如果在未对潜在入侵事件做出任何软件决策之前发生看门狗复位，则相当于将“潜在入侵”切换为“确认入侵”。

器件资源保护配置

可以对一些器件资源进行配置，将其列入受入侵检测保护的器件机密信息列表。

当 TAMP_RPCFGR 中的 RPCFGz = 0 时，与 RPCFGz 相关联的器件资源不受 TAMP 外设保护：

- 不受入侵检测影响（无论是在确认模式下还是在潜在模式下）
- 不受 BKERASE 软件命令影响
- 不受 BKBLOCK 软件命令影响

当 TAMP_RPCFGR 中的 RPCFGz = 1 时，与 RPCFGz 相关联的器件资源受 TAMP 外设保护：

- 受确认入侵检测和 BKERASE 软件命令的影响，如*表 242：TAMP 互连*中的 tamp_confirmed_rpcfgz 信号部分所述
- 受潜在入侵检测和 BKBLOCK 软件命令的影响，如*表 242：TAMP 互连*中的 tamp_potential_rpcfgz 信号部分所述

表 243. 器件资源 x 入侵保护

-	潜在入侵 或 BKBLOCK	确认入侵 或 BKERASE
RPCFGx = 0	对器件资源 x 无影响	对器件资源 x 无影响
RPCFGx = 1	对器件机密信息 x 进行保护（如 tamp_potential_rpcfgx 所述）(1)	对器件机密信息 x 进行保护（如 tamp_confirmed_rpcfgx(1) 所述）

1. 请参见*表 242：TAMP 互连*。

通过软件阻止访问器件机密信息

默认情况下，应用程序可以访问器件机密信息，但检测到入侵事件标志的情况除外：只要入侵标志置 1，就不能访问器件机密信息。

可通过软件将 TAMP_CR2 寄存器的 BKBLOCK 位置 1 来阻止访问器件机密信息。只有在 BKBLOCK = 0 且没有任何入侵标志置 1 时，才能访问器件机密信息。

33.3.9 入侵检测配置和初始化

可通过将 TAMP_CR 寄存器中相应的 TAMPxE 位置 1 来使能各输入。

每个 TAMP_INx 入侵检测输入都分别与 TAMP_SR 寄存器中的标志 TAMPxF 相关联。

通过将 TAMP_IER 寄存器中的 TAMPxIE 位置 1，可在发生入侵检测事件时（当 TAMPxF 置 1 时）生成中断。当相应的 TAMPxMSK 置 1 时，不允许将 TAMPxIE 置 1。

出现入侵事件时生成触发输出

低功耗定时器可将入侵事件检测用作触发输入。

将 TAMP_CR 寄存器中的 TAMPxMSK 位清零时，必须通过软件将 TAMPxF 标志清零以便在同一引脚上检测新入侵。

将 TAMPxMSK 位置 1 时，TAMPxF 标志会被屏蔽，并在 TAMP_SR 寄存器中保持清零。这种配置允许在停止模式下自动触发低功耗定时器，无需通过唤醒系统来执行 TAMPxF 清零。在这种情况下，备份寄存器不清零。

该功能仅在入侵配置为*对入侵输入的带滤波电平检测（被动模式）*模式（TAMPFLT ≠ 00 且未选择主动模式）时可用。

入侵事件的时间戳

当 RTC_CR 中的 TAMPTS 置 1 时，任何内部或外部入侵事件都会导致时间戳事件的发生。因入侵事件而引发时间戳事件时，如同发生正常时间戳事件一样，RTC_SR 中的 TSF 位或 TSOVF 位会置 1。

注意： *TSF 在 TAMPxF 标志置 1 后的 3 个 ck_apre 周期内置 1。如果 RTCCLK 停止，则 TSF 不会置 1（RTCCLK 重启时置 1）。*

注意： *如果 TAMPxF 在 TSF 按预期置 1 前被清零，则 TSF 不会置 1。因此，如果 TAMPTS = 1，软件应在清零入侵标志前等待时间戳标志，或者应读取 TAMP 中断程序中的 RTC 计数器值。*

对入侵输入的边沿检测（被动模式）

如果 TAMPFLT 位设置为 00，当相应的 TAMPxTRG 位上出现上升沿或下降沿时，TAMP_INx 引脚将生成入侵检测事件。选择边沿检测时，会禁止 TAMP_INx 输入上的内部上拉电阻。

小心： 当 TAMPFLT = 00 且 TAMPxTRG = 0（上升沿检测）时，如果在使能入侵检测前入侵输入已处于高电平，则可通过硬件来检测入侵事件。

检测到入侵事件并清零后，应当在重新编程备份寄存器 (TAMP_BKPxR) 之前禁止 TAMP_INx，然后再重新使能（TAMPxE 置 1）。这可防止应用在 TAMP_INx 输入值仍指示入侵检测时，对备份寄存器执行写操作。这相当于对 TAMP_INx 输入的电平检测。

注意： *当 V_{DD} 电源关闭时，入侵检测仍有效。要避免意外复位备份寄存器，应将 TAMPx 映射到的引脚从外部连接到正确的电平。*

对入侵输入的带滤波电平检测（被动模式）

通过将 TAMPFLT 设置为非零值可执行带滤波的电平检测。在 TAMPxTRG 位指定的电平连续出现 2、4 或 8 个（取决于 TAMPFLT 值）采样时生成入侵检测事件。

除非通过将 TAMPPUDIS 置 1 禁止入侵输入，否则在入侵输入的状态被采样前，将通过 I/O 内部上拉电阻对 TAMP_INx 输入预充电。预充电的持续时间由 TAMPPRCH 位确定，允许增大 TAMP_INx 输入上的电容。

可使用 TAMPFREQ 确定用于电平检测的采样频率，以便使入侵检测延迟与上拉电阻功耗之间达到最佳平衡。

注意： *有关上拉电阻的电气特性，请参见微控制器数据手册。*

主动入侵检测

当 TAMP_ATCR 中的 TAMPxAM 位置 1 时，入侵事件配置为主动模式，该模式基于 TAMP_OUTy 引脚与 TAMP_INx 引脚之间的比较。默认情况下 (ATOSHARE = 0)，在 TAMP_INx 与 TAMP_OUTx (y = x) 之间进行比较。当 ATOSHARE 位置 1 时，同一输出可用于多个入侵输入。借助 ATOSHARE 和 ATOSELx 位，只要选择了与主动入侵输入 TAMP_INx 进行比较的输出 (TAMPxEN = TAMPxAM = 1)，就会在 I/O 上使能 TAMP_OUTy 功能。请参见 TAMP_ATCRx（x = 1 和 2）寄存器中 ATOSHARE 和 ATOSEL 位的说明。

每两个 CK_ATPER 周期 (CK_ATPER = 2^{ATPER} × CK_ATPRE)，TAMP_OUTy 输出引脚就会提供一个由伪随机数发生器 (PRNG) 生成的值。输出该值后，TAMP_OUTy 引脚会在一个 CK_ATPER 周期后输出其相反值。

表 244. 主动入侵输出变化周期

ATCKSEL[3:0]	CK_ATPRE 频率	ATPER[2:0]	入侵输出变化 (CK_ATPER) 频率	入侵输出变化周期(1) (ms)
0x0	f_{RTCCLK}	0x0	f_{RTCCLK}	0.030
		0x1	$f_{RTCCLK}/2$	0.061
		0x2	$f_{RTCCLK}/4$	0.122
		0x3	$f_{RTCCLK}/8$	0.244
		0x4	$f_{RTCCLK}/16$	0.488
		0x5	$f_{RTCCLK}/32$	0.977
		0x6	$f_{RTCCLK}/64$	1.953
		0x7	$f_{RTCCLK}/128$	3.906
…	…	…	…	…
0x7	$f_{RTCCLK}/128$	0x0	$f_{RTCCLK}/128$	3.906
		0x1	$f_{RTCCLK}/256$	7.8125
		0x2	$f_{RTCCLK}/512$	15.625
		0x3	$f_{RTCCLK}/1024$	31.250
		0x4	$f_{RTCCLK}/2048$	62.5
		0x5	$f_{RTCCLK}/4096$	125
		0x6	$f_{RTCCLK}/8192$	250
		0x7	$f_{RTCCLK}/16384$	500
0xB	$f_{RTCCLK}/2048$(2)	0x0	$f_{RTCCLK}/2048$	62.5
		0x1	$f_{RTCCLK}/4096$	125
		0x2	$f_{RTCCLK}/8192$	250
		0x3	$f_{RTCCLK}/16384$	500
		0x4	$f_{RTCCLK}/32768$	1000
		0x5	$f_{RTCCLK}/65536$	2000
		0x6	$f_{RTCCLK}/131072$	4000
		0x7	$f_{RTCCLK}/262144$	8000

1. 假设 f_{RTCCLK} = 32768 Hz。
2. 该设置要求 (PREDIV_A+1) = 128 且 (PREDIV_S+1) 是 16 的倍数。

根据选择的入侵输出数量的不同，所选入侵输出使用 PRNG 的频率也有所不同。选择的输出数量取决于 TAMPxAM、TAMPxE、ATOSEL 和 ATOSHARE。

- 当仅选择 1 个输出时：每 16 个 CK_ATPER 周期使用一次 PRNG。
- 当选择 2 个输出时：每 8 个 CK_ATPER 周期使用一次 PRNG。
- 当选择 3 个或 4 个输出时：每 4 个 CK_ATPER 周期使用一次 PRNG。
- 当选择 5 个或更多输出时：每 2 个 CK_ATPER 周期使用一次 PRNG。

PRNG 需要至少 9 个 CK_ATPRE 周期才能输出新值。因此，下表提供了确保正常工作的最小 ATPER 值：

表 245. 最小 ATPER 值

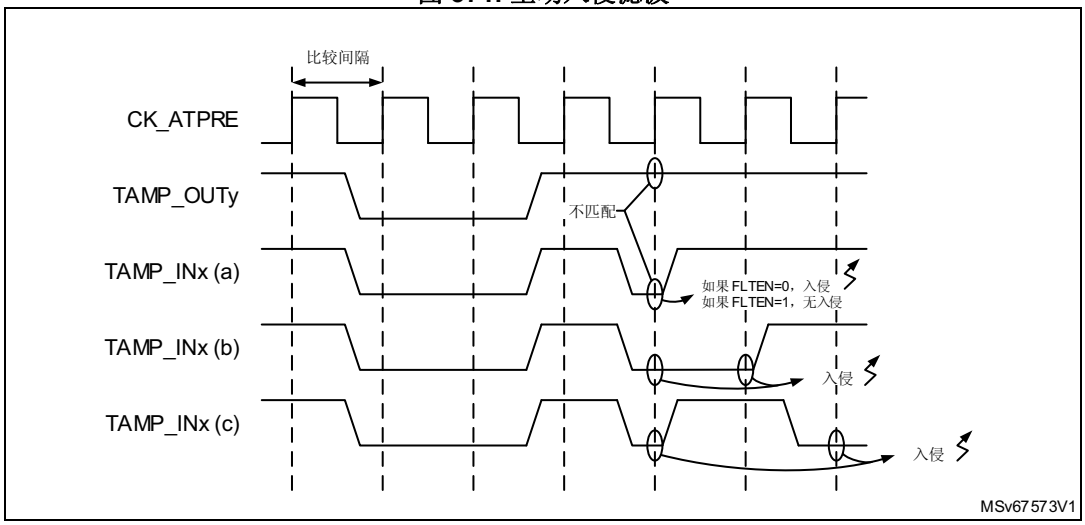
选择的输出数量	ATPER 最小值
1	0
2	1
3 或 4	2
5 或以上	3

TAMP_INx 引脚从外部连接到 TAMP_OUTy 引脚。每个 CK_ATPRE 周期都会将 TAMP_OUTy 输出值与 TAMP_INx 接收值进行比较。如果比较不匹配，TAMP_SR 寄存器中的 TAMPxF 位将会置 1。

例如，可以使用 TAMP_OUT1 与 TAMP_IN1 和 TAMP_IN2 进行比较，方法是在主动模式下配置并使能 TAMP1 和 TAMP2（ATOSHARE = 1、ATOSEL1 = 000 且 ATOSEL2 = 000）。

当 FLTEN = 1 时，主动入侵模式可与输入滤波结合使用。在这种情况下，只有在连续 4 个比较样本中出现 2 次比较错误时，才会检测到入侵。

图 371. 主动入侵滤波



如*图 371* 所示，如果 FLTEN = 0，在对 TAMP_INx 输入进行采样时，TAMP_OUTy 输出与相关 TAMP_INx 输入之间的任何不匹配都会生成入侵事件。所有三个示例 (a)、(b) 和 (c) 都显示了这种情况。

如果 FLTEN = 1，则示例 (a) 不会生成入侵事件，因为在连续四次比较中仅检测到一次不匹配。在示例 (b) 中，由于连续两次检测到不匹配，因此生成了入侵事件。在示例 (c) 中，即使在连续采样过程中并未发生不匹配，但由于在连续四次比较中出现了两次不匹配，因此也生成了入侵事件。

将 FLTEN 置 1 可以避免因 TAMP_INx 输入上的干扰、突发或瞬态状态而触发不必要的入侵检测，方法是忽略小于一个 CK_ATPRE 周期的单个脉冲（对 TAMP_ATCR1 寄存器的 ATCKSEL 位域进行编程）。*表 246* 列出了 ATCKSEL 各种可能设置的最小滤波脉宽（假设 f_{RTCCLK} = 32.768 kHz）。

表 246. 主动入侵滤波脉冲的持续时间

ATCKSEL[3:0]	CK_ATPRE 频率	最小滤波脉宽 (ms)
0x0	f_{RTCCLK}	0.030
0x1	$f_{RTCCLK}/2$	0.061
0x2	$f_{RTCCLK}/4$	0.122
0x3	$f_{RTCCLK}/8$	0.244
0x4	$f_{RTCCLK}/16$	0.488
0x5	$f_{RTCCLK}/32$	0.977
0x6	$f_{RTCCLK}/64$	1.953
0x7	$f_{RTCCLK}/128$	3.906
0xB	$f_{RTCCLK}/2048$	62.500(1)

1. 该设置要求 (PREDIV_A+1) = 128 且 (PREDIV_S+1) 是 16 的倍数。

注意： *但是，如果小于一个 CK_ATPRE 周期的多个脉冲在连续四次比较中出现了两次不匹配，可能会引发入侵。*

小心： 当 ATCKSEL[3] = 1 时，进入 RTC 初始化模式会停止 CK_ATPRE 和 CK_ATPER 时钟。因此，TAMP_OUTy 引脚停止切换，直到退出 INIT 模式。

请参见 *日历初始化和配置*一节。

如果在 ATCKSEL[3] = 1 时使用 RTC 二进制模式，则另请参见 *RTC 闹钟 A 亚秒寄存器 (RTC_ALRMASSR)*、*RTC 闹钟 B 亚秒寄存器 (RTC_ALRMBSSR)*、*RTC 闹钟 A 二进制模式寄存器 (RTC_ALRABINR)* 和 *RTC 闹钟 B 二进制模式寄存器 (RTC_ALRBBINR)*。

小心： 小心：当 ATCKSEL[3] = 1 时，如果日历上溢，主动入侵检测将不再起作用。当日历上溢时，必须使能内部入侵 5 以确保实施入侵保护。

伪随机数发生器必须使用种子进行初始化。可通过在 TAMP_ATSEEDR 寄存器中连续写入四个 32 位随机值来完成该初始化操作。对种子进行编程会自动将其发送到 PRNG。只要新种子由 PRNG 进行传输和转化，TAMP_ATOR 中的 SEEDF 位就会置 1，并且不允许关闭 TAMP APB 时钟。写入第四个种子后，转化的持续时间将长达 184 个 APB 时钟周期。因此，写入新种子后，用户必须等待至 SEEDF 清零后才能进入低功耗模式。

主动入侵输出只有在写入第一个种子并完成转化后才会激活。随后，可以在主动入侵活动期间写入并转化新种子。

主动入侵初始化

下面给出了系统复位后通过软件初始化主动入侵的流程：
读取 TAMP_ATOR 寄存器中的 INITS。

- 如果 INITS = 0x0（初始化未完成）：
  a) 写入 TAMP_ATCR 以配置主动入侵时钟、滤波器和共用输出（如果有）以及主动模式。
  b) 写入 TAMP_CR1 以使能入侵（必须在同一个写访问中使能所需的所有入侵）。
  c) 对 TAMP_ATSEEDR 执行四次写操作，以写入 SEED。
  d) 等待至 TAMP_ATOR 中的 SEEDF = 0。备份寄存器随即受到主动入侵保护。
- 如果 INITS = 0x1（初始化已完成）：
  无需初始化。为了增加随机性，应定期提供新的 SEED。提供新 SEED 时，需等待至 SEEDF = 0 之后才能进入低功耗模式，该模式会关闭 TAMP APB 时钟。
- 如果通过软件禁止了入侵，之后又重新使能，则在使能入侵后，必须写入 SEED：
  a) 写入 TAMP_CR1 以使能入侵（必须在同一个写访问中使能所需的所有入侵）。
  b) 对 TAMP_ATSEEDR 执行四次写操作，以写入 SEED。
  c) 等待至 TAMP_ATOR 中的 SEEDF = 0。备份寄存器随即受到主动入侵保护。

33.4 TAMP 低功耗模式

表 247. 低功耗模式对 TAMP 的影响

模式	说明
睡眠	无影响。 TAMP 中断可使器件退出睡眠模式。
停止	对所有功能都没有影响，但使用滤波和主动入侵模式的电平检测除外，这些模式仅在时钟源为 LSE 或 LSI 时保持有效。 入侵事件可使器件退出停止模式。
待机	对所有功能都没有影响，但使用滤波和主动入侵模式的电平检测除外，这些模式仅在时钟源为 LSE 或 LSI 时保持有效。 入侵事件可使器件退出待机模式。

表 248. 各个模式下的 TAMP 引脚功能

引脚名称	所有低功耗模式下的功能正常	V_{BAT} 模式下的功能正常
TAMP_IN[2:1]	是(1)	是(1)
TAMP_OUT[2:1]	是(2)	是(2)

1. 只有 PC13 和 PA0 可以在待机模式和 V_{BAT} 模式下正常工作。
2. 只有 PC13 可以在待机模式和 V_{BAT} 模式下正常工作。

33.5 TAMP 中断

表 249. 中断请求

中断缩写	中断事件	事件标志(1)	使能控制位	中断清除方法	退出低功耗模式
TAMP	入侵 x(2)	TAMPxF	TAMPxIE	向 CTAMPxF 中写入 1	是(3)
	内部入侵 y(2)	ITAMPyF	ITAMPyIE	向 CITAMPyF 中写入 1	是(3)

1. 事件标志位于 TAMP_SR 寄存器中。
2. 入侵事件和内部入侵事件的数量取决于产品。
3. 有关低功耗模式下可用功能的更多详细信息，请参见*表 247：低功耗模式对 TAMP 的影响*。

33.6 TAMP 寄存器

有关寄存器说明中使用的缩写，请参见参考手册中的*第 1.2 节*。外设寄存器可按字（32 位）进行访问。

33.6.1 TAMP 控制寄存器 1 (TAMP_CR1)

TAMP control register 1

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x00

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	ITAMP15E	Res.	ITAMP13E	ITAMP12E	ITAMP11E	Res.	ITAMP9E	ITAMP8E	ITAMP7E	ITAMP6E	ITAMP5E	ITAMP4E	ITAMP3E	ITAMP2E	ITAMP1E
	rw		rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2E	TAMP1E
														rw	rw

位 31 保留，必须保持复位值。

位 30 **ITAMP15E**：内部入侵 15 使能 (Internal tamper 15 enable)
0：禁止内部入侵 15。
1：使能内部入侵 15。

位 29 保留，必须保持复位值。

位 28 **ITAMP13E**：内部入侵 13 使能 (Internal tamper 13 enable)
0：禁止内部入侵 13。
1：使能内部入侵 13。

位 27 **ITAMP12E**：内部入侵 12 使能 (Internal tamper 12 enable)
0：禁止内部入侵 12。
1：使能内部入侵 12。

位 26 **ITAMP11E**：内部入侵 11 使能 (Internal tamper 11 enable)
0：禁止内部入侵 11。
1：使能内部入侵 11。

位 25 保留，必须保持复位值。

位 24 **ITAMP9E**：内部入侵 9 使能 (Internal tamper 9 enable)
0：禁止内部入侵 9。
1：使能内部入侵 9。

位 23 **ITAMP8E**：内部入侵 8 使能 (Internal tamper 8 enable)
0：禁止内部入侵 8。
1：使能内部入侵 8。

位 22 **ITAMP7E**：内部入侵 7 使能 (Internal tamper 7 enable)
0：禁止内部入侵 7。
1：使能内部入侵 7。

位 21 **ITAMP6E**：内部入侵 6 使能 (Internal tamper 6 enable)
0：禁止内部入侵 6。
1：使能内部入侵 6。

位 20 **ITAMP5E**：内部入侵 5 使能 (Internal tamper 5 enable)
0：禁止内部入侵 5。
1：使能内部入侵 5。

位 19 **ITAMP4E**：内部入侵 4 使能 (Internal tamper 4 enable)
0：禁止内部入侵 4。
1：使能内部入侵 4。

位 18 **ITAMP3E**：内部入侵 3 使能 (Internal tamper 3 enable)
0：禁止内部入侵 3。
1：使能内部入侵 3。

位 17 **ITAMP2E**：内部入侵 2 使能 (Internal tamper 2 enable)
0：禁止内部入侵 2。
1：使能内部入侵 2。

位 16 **ITAMP1E**：内部入侵 1 使能 (Internal tamper 1 enable)
0：禁止内部入侵 1。
1：使能内部入侵 1。

位 15:8 保留，必须保持复位值。

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 保留，必须保持复位值。

位 4 保留，必须保持复位值。

位 3 保留，必须保持复位值。

位 2 保留，必须保持复位值。

位 1 **TAMP2E**：对 TAMP_IN2 的入侵检测使能 (Tamper detection on TAMP_IN2 enable)(1)

0：禁止对 TAMP_IN2 的入侵检测。

1：使能对 TAMP_IN2 的入侵检测。

位 0 **TAMP1E**：对 TAMP_IN1 的入侵检测使能 (Tamper detection on TAMP_IN1 enable)(1)

0：禁止对 TAMP_IN1 的入侵检测。

1：使能对 TAMP_IN1 的入侵检测。

1. 必须先配置入侵检测模式（通过 TAMP_FLTCR、TAMP_ATCR1、TAMP_ATCR2 寄存器和 TAMP_CR2 中的 TAMPxTRG 位进行选择），然后才能使能入侵检测。

33.6.2 TAMP 控制寄存器 2 (TAMP_CR2)

TAMP control register 2

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x04

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	TAMP2 TRG	TAMP1 TRG	BK ERASE	BK BLOCK	Res.	Res.	Res.	Res.	TAMP2 MSK	TAMP1 MSK
						rw	rw	w	rw					rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2 POM	TAMP1 POM
														rw	rw

位 31 保留，必须保持复位值。

位 30 保留，必须保持复位值。

位 29 保留，必须保持复位值。

位 28 保留，必须保持复位值。

位 27 保留，必须保持复位值。

位 26 保留，必须保持复位值。

位 25 **TAMP2TRG**：入侵 2 输入的有效电平 (Active level for tamper 2 input)

0：如果 TAMPFLT ≠ 00，入侵 2 输入保持低电平会触发入侵检测事件。
如果 TAMPFLT = 00，入侵 2 输入上升沿会触发入侵检测事件。

1：如果 TAMPFLT ≠ 00，入侵 2 输入保持高电平会触发入侵检测事件。
如果 TAMPFLT = 00，入侵 2 输入下降沿会触发入侵检测事件。

位 24 **TAMP1TRG**：入侵 1 输入的有效电平 (Active level for tamper 1 input)

0：如果 TAMPFLT ≠ 00，入侵 1 输入保持低电平会触发入侵检测事件。
如果 TAMPFLT = 00，入侵 1 输入上升沿会触发入侵检测事件。

1：如果 TAMPFLT ≠ 00，入侵 1 输入保持高电平会触发入侵检测事件。
如果 TAMPFLT = 00，入侵 1 输入下降沿会触发入侵检测事件。

位 23 **BKERASE**：备份寄存器和器件机密信息(1)擦除 (Backup registers and device secrets erase)
向该位写入“1”会重置备份寄存器和器件机密信息(1)。写入 0 无影响。该位始终读为 0。

位 22 **BKBLOCK**：阻止备份寄存器和器件机密信息(1)访问 (Backup registers and device secrets access blocked)
0：如果没有任何入侵标志置 1，则可以对备份寄存器和器件机密信息(1)进行访问
1：不可以对备份寄存器和器件机密信息(1)进行访问

位 21:19 保留，必须保持复位值。

位 18 保留，必须保持复位值。

位 17 **TAMP2MSK**：入侵 2 屏蔽 (Tamper 2 mask)
0：入侵 2 事件生成触发事件，必须通过软件将 TAMP2F 清零来允许下一次入侵事件检测。
1：入侵 2 事件生成触发事件。屏蔽 TAMP2F 并由硬件在内部将其清零。不擦除备份寄存器和器件机密信息(1)。
TAMP2MSK 置 1 时，不得使能入侵 2 中断。

位 16 **TAMP1MSK**：入侵 1 屏蔽 (Tamper 1 mask)
0：入侵 1 事件生成触发事件，必须通过软件将 TAMP1F 清零来允许下一次入侵事件检测。
1：入侵 1 事件生成触发事件。屏蔽 TAMP1F 并由硬件在内部将其清零。不擦除备份寄存器和器件机密信息(1)。
TAMP1MSK 置 1 时，不得使能入侵 1 中断。

位 15:8 保留，必须保持复位值。

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 保留，必须保持复位值。

位 4 保留，必须保持复位值。

位 3 保留，必须保持复位值。

位 2 保留，必须保持复位值。

位 1 **TAMP2POM**：入侵 2 潜在模式 (Tamper 2 potential mode)
0：在确认模式(1)下执行入侵 2 事件检测。
1：在潜在模式(2)下执行入侵 2 事件检测。

位 0 **TAMP1POM**：入侵 1 潜在模式 (Tamper 1 potential mode)
0：在确认模式(1)下执行入侵 1 事件检测。
1：在潜在模式(2)下执行入侵 1 事件检测。

1. *表 242：TAMP 互连*中的 tamp_confirmed 和 tamp_confirmed_rpcfgx 信号描述了确认模式下入侵检测的作用。
2. *表 242：TAMP 互连*中的 tamp_potential 和 tamp_potential_rpcfgx 信号描述了潜在模式下入侵检测的作用。

33.6.3 TAMP 控制寄存器 3 (TAMP_CR3)

TAMP control register 3

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x08

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	ITAMP 15POM	Res.	ITAMP 13POM	ITAMP 12POM	ITAMP 11POM	Res.	ITAMP9 POM	ITAMP8 POM	ITAMP7 POM	ITAMP6 POM	ITAMP5 POM	ITAMP4 POM	ITAMP3 POM	ITAMP2 POM	ITAMP1 POM
	rw		rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15 保留，必须保持复位值。

位 14 **ITAMP15POM**：内部入侵 15 潜在模式 (Internal tamper 15 potential mode)
0：在确认模式(1)下执行内部入侵 15 事件检测。
1：在潜在模式(2)下执行内部入侵 15 事件检测。

位 13 保留，必须保持复位值。

位 12 **ITAMP13POM**：内部入侵 13 潜在模式 (Internal tamper 13 potential mode)
0：在确认模式(1)下执行内部入侵 13 事件检测。
1：在潜在模式(2)下执行内部入侵 13 事件检测。

位 11 **ITAMP12POM**：内部入侵 12 潜在模式 (Internal tamper 12 potential mode)
0：在确认模式(1)下执行内部入侵 12 事件检测。
1：在潜在模式(2)下执行内部入侵 12 事件检测。

位 10 **ITAMP11POM**：内部入侵 11 潜在模式 (Internal tamper 11 potential mode)
0：在确认模式(1)下执行内部入侵 11 事件检测。
1：在潜在模式(2)下执行内部入侵 11 事件检测。

位 9 保留，必须保持复位值。

位 8 **ITAMP9POM**：内部入侵 9 潜在模式 (Internal tamper 9 potential mode)
0：在确认模式(1)下执行内部入侵 9 事件检测。
1：在潜在模式(2)下执行内部入侵 9 事件检测。

位 7 **ITAMP8POM**：内部入侵 8 潜在模式 (Internal tamper 8 potential mode)
0：在确认模式(1)下执行内部入侵 8 事件检测。
1：在潜在模式(2)下执行内部入侵 8 事件检测。

位 6 **ITAMP7POM**：内部入侵 7 潜在模式 (Internal tamper 7 potential mode)
0：在确认模式(1)下执行内部入侵 7 事件检测。
1：在潜在模式(2)下执行内部入侵 7 事件检测。

位 5 **ITAMP6POM**：内部入侵 6 潜在模式 (Internal tamper 6 potential mode)
0：在确认模式(1)下执行内部入侵 6 事件检测。
1：在潜在模式(2)下执行内部入侵 6 事件检测。

位 4 **ITAMP5POM**：内部入侵 5 潜在模式 (Internal tamper 5 potential mode)
0：在确认模式(1)下执行内部入侵 5 事件检测。
1：在潜在模式(2)下执行内部入侵 5 事件检测。

位 3 **ITAMP4POM**：内部入侵 4 潜在模式 (Internal tamper 4 potential mode)
0：在确认模式(1)下执行内部入侵 4 事件检测。
1：在潜在模式(2)下执行内部入侵 4 事件检测。

位 2 **ITAMP3POM**：内部入侵 3 潜在模式 (Internal tamper 3 potential mode)
0：在确认模式(1)下执行内部入侵 3 事件检测。
1：在潜在模式(2)下执行内部入侵 3 事件检测。

位 1 **ITAMP2POM**：内部入侵 2 潜在模式 (Internal tamper 2 potential mode)
0：在确认模式(1)下执行内部入侵 2 事件检测。
1：在潜在模式(2)下执行内部入侵 2 事件检测。

位 0 **ITAMP1POM**：内部入侵 1 潜在模式 (Internal tamper 1 potential mode)
0：在确认模式(1)下执行内部入侵 1 事件检测。
1：在潜在模式(2)下执行内部入侵 1 事件检测。

1. *表 242：TAMP 互连*中的 tamp_confirmed 和 tamp_confirmed_rpcfgx 信号描述了确认模式下内部入侵检测的作用。
2. *表 242：TAMP 互连*中的 tamp_potential 和 tamp_potential_rpcfgx 信号描述了潜在模式下内部入侵检测的作用。

33.6.4 TAMP 滤波器控制寄存器 (TAMP_FLTCR)

TAMP filter control register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x0C

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31:8 保留，必须保持复位值。

位 7 **TAMPPUDIS**：TAMP_INx 上拉禁止 (TAMP_INx pull-up disable)
该位决定在每次采样之前是否对每个 TAMPx 引脚都进行预充电。
0：采样之前对 TAMP_INx 引脚进行预充电（使能内部上拉）
1：禁止对 TAMP_INx 引脚进行预充电

位 6:5 **TAMPPRCH[1:0]**：TAMP_INx 预充电持续时间 (TAMP_INx precharge duration)
这些位决定了在每次采样之前激活上拉的持续时间。TAMPPRCH 对每个 TAMP_INx 输入都有效。
0x0：1 个 RTCCLK 周期
0x1：2 个 RTCCLK 周期
0x2：4 个 RTCCLK 周期
0x3：8 个 RTCCLK 周期

位 4:3 **TAMPFLT[1:0]**：TAMP_INx 滤波器计数 (TAMP_INx filter count)

这些位决定了为激活入侵事件所需的指定电平 (TAMP*TRG) 上的连续采样次数。TAMPFLT 对每个 TAMP_INx 输入都有效。

0x0：在 TAMP_INx 输入转变为有效电平的边沿激活入侵事件（TAMP_INx 输入上无内部上拉）。

0x1：在有效电平上连续执行 2 次采样后激活入侵事件。

0x2：在有效电平上连续执行 4 次采样后激活入侵事件。

0x3：在有效电平上连续执行 8 次采样后激活入侵事件。

位 2:0 **TAMPFREQ[2:0]**：入侵采样频率 (Tamper sampling frequency)

决定对每个 TAMP_INx 输入进行采样时的频率。

0x0：RTCCLK / 32768（RTCCLK = 32768 Hz 时为 1 Hz）

0x1：RTCCLK / 16384（RTCCLK = 32768 Hz 时为 2 Hz）

0x2：RTCCLK / 8192（RTCCLK = 32768 Hz 时为 4 Hz）

0x3：RTCCLK / 4096（RTCCLK = 32768 Hz 时为 8 Hz）

0x4：RTCCLK / 2048（RTCCLK = 32768 Hz 时为 16 Hz）

0x5：RTCCLK / 1024（RTCCLK = 32768 Hz 时为 32 Hz）

0x6：RTCCLK / 512（RTCCLK = 32768 Hz 时为 64 Hz）

0x7：RTCCLK / 256（RTCCLK = 32768 Hz 时为 128 Hz）

注意： *该寄存器仅与被动模式下的入侵输入有关。*

33.6.5 TAMP 主动入侵控制寄存器 1 (TAMP_ATCR1)

TAMP active tamper control register 1

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x10

备份域复位值：0x0007 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTEN	ATOSHARE	Res.	Res.	Res.	ATPER[2:0]			Res.	Res.	Res.	Res.	ATCKSEL[3:0]			
rw	rw				rw	rw	rw					rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	ATOSEL2[1:0]		ATOSEL1[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	TAMP2AM	TAMP1AM
				rw	rw	rw	rw							rw	rw
ATOSEL4[1:0]		ATOSEL3[1:0]		ATOSEL2[1:0]		ATOSEL1[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	TAMP2AM	TAMP1AM
rw	rw	rw	rw	rw	rw	rw	rw							rw	rw

位 31 **FLTEN**：主动入侵滤波器使能 (Active tamper filter enable)

0：禁止主动入侵滤波

1：使能主动入侵滤波：当在连续 4 次采样中出现 2 次比较不匹配时，会检测到入侵事件。

位 30 **ATOSHARE**：主动入侵输出共用 (Active tamper output sharing)

0：每个主动入侵输入 TAMP_INi 都与其专用输出 TAMP_OUTi 进行比较

1：每个主动入侵输入 TAMP_INi 都与 ATOSELi 位定义的 TAMPOUTSELi 进行比较。

位 29:27 保留，必须保持复位值。

位 26:24 **ATPER[2:0]**：主动入侵输出变化周期 (Active tamper output change period)
入侵输出每 CK_ATPER = (2^{ATPER} x CK_ATPRE) 个周期变化一次。请参见 *表 245：最小 ATPER 值*。

位 23:20 保留，必须保持复位值。

位 19:16 **ATCKSEL[3:0]**：主动入侵 RTC 异步预分频器时钟选择 (Active tamper RTC asynchronous prescaler clock selection)
这些位用于选择 RTC 异步预分频器级输出。所选时钟为 CK_ATPRE。
0000：选择 RTCCLK
0001：选择 RTCCLK/2
0010：选择 RTCCLK/4
0011：选择 RTCCLK/8
0100：选择 RTCCLK/16
0101：选择 RTCCLK/32
0110：选择 RTCCLK/64
0111：选择 RTCCLK/128
1011：当 (PREDIV_A+1) = 128 且 (PREDIV_S+1) 是 16 的倍数时，选择 RTCCLK/2048。
其他值：保留
注意：这些位只能在所有主动入侵均已禁止时写入。禁止所有主动入侵后，写保护最多可保持 1.5 个 CK_ATPRE 周期。

位 15:12 保留，必须保持复位值。

位 15:14 **ATOSEL4[1:0]**：主动入侵共用输出 4 选择 (Active tamper shared output 4 selection)
00：TAMPOUTSEL4 = TAMP_OUT1
01：TAMPOUTSEL4 = TAMP_OUT2
10：TAMPOUTSEL4 = TAMP_OUT3
11：TAMPOUTSEL4 = TAMP_OUT4
如果封装引脚排列中未提供 TAMP_OUTx 输出，则输出选择值保留，不得使用。

位 13:12 **ATOSEL3[1:0]**：主动入侵共用输出 3 选择 (Active tamper shared output 3 selection)
00：TAMPOUTSEL3 = TAMP_OUT1
01：TAMPOUTSEL3 = TAMP_OUT2
10：TAMPOUTSEL3 = TAMP_OUT3
11：TAMPOUTSEL3 = TAMP_OUT4
如果封装引脚排列中未提供 TAMP_OUTx 输出，则输出选择值保留，不得使用。

位 11:10 **ATOSEL2[1:0]**：主动入侵共用输出 2 选择 (Active tamper shared output 2 selection)
00：TAMPOUTSEL2 = TAMP_OUT1
01：TAMPOUTSEL2 = TAMP_OUT2
10：TAMPOUTSEL2 = TAMP_OUT3
11：TAMPOUTSEL2 = TAMP_OUT4
如果封装引脚排列中未提供 TAMP_OUTx 输出，则输出选择值保留，不得使用。

位 9:8 **ATOSEL1[1:0]**：主动入侵共用输出 1 选择 (Active tamper shared output 1 selection)
00：TAMPOUTSEL1 = TAMP_OUT1
01：TAMPOUTSEL1 = TAMP_OUT2
10：TAMPOUTSEL1 = TAMP_OUT3
11：TAMPOUTSEL1 = TAMP_OUT4
如果封装引脚排列中未提供 TAMP_OUTx 输出，则输出选择值保留，不得使用。

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 保留，必须保持复位值。

位 4 保留，必须保持复位值。

位 3 保留，必须保持复位值。

位 2 保留，必须保持复位值。

位 1 **TAMP2AM**：入侵 2 主动模式 (Tamper 2 active mode)
0：入侵 2 检测使用被动模式。
1：入侵 2 检测使用主动模式。

位 0 **TAMP1AM**：入侵 1 主动模式 (Tamper 1 active mode)
0：入侵 1 检测使用被动模式。
1：入侵 1 检测使用主动模式。

注意： *当 TAMPxAM 位置 1 时，不允许更改该寄存器中的主动入侵配置，除非 TAMP_CR1 寄存器中所有相应的 TAMPxE 位都清零。*

配置为主动模式的所有入侵必须同时使能（通过在同一个 TAMP_CR1 写访问中将所有相关的 TAMPxE 都置 1）。

配置为主动模式的所有入侵必须同时禁止（通过在同一个 TAMP_CR1 写访问中将所有相关的 TAMPxE 都清零）。

禁止主动入侵后，必须等待至少 1 个 CK_ATPRE 周期才能将其重新使能。

33.6.6 TAMP 主动入侵种子寄存器 (TAMP_ATSEEDR)

TAMP active tamper seed register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x14

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31:0 **SEED[31:0]**：伪随机数发生器种子值 (Pseudo-random generator seed value)
必须向该寄存器连续写入四个 32 位值，才能为 PRNG 提供 128 位种子。写入该寄存器会自动将种子值发送到 PRNG。

33.6.7 TAMP 主动入侵输出寄存器 (TAMP_ATOR)

TAMP active tamper output register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x18

备份域复位值：0x0000 0000

系统复位：不受影响（SEEDF 除外，它在复位时被清零）。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INITS	SEEDF	Res.	Res.	Res.	Res.	Res.	Res.	PRNG[7:0]							
r	r							r	r	r	r	r	r	r	r

位 31:16 保留，必须保持复位值。

位 15 **INITS**：主动入侵初始化状态 (Active tamper initialization status)

当 PRNG 接收到第一个 128 位种子时，该标志由硬件置 1，这意味着使能的主动入侵可用。当禁止主动入侵时，该标志被清零。

位 14 **SEEDF**：种子运行标志 (Seed running flag)

将新种子写入 TAMP_ATSEEDR 时，该标志由硬件置 1。当 PRNG 接收到该新种子时，该标志由硬件清零，也可以通过系统复位清零。只要 SEEDF 置 1，就不得关闭 TAMP APB 时钟。

位 13:8 保留，必须保持复位值。

位 7:0 **PRNG[7:0]**：伪随机数发生器值 (Pseudo-random generator value)

该位域提供 PRNG 输出的值。由于同步延迟可能导致不一致，因此必须至少读取 PRNG 两次。如果两次读取的值相等，则表示数值正确。

33.6.8 TAMP 主动入侵控制寄存器 2 (TAMP_ATCR2)

TAMP active tamper control register 2

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x1C

备份域复位值：0x0000 0000

系统复位：不受影响

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	ATOSEL2[2:0]			ATOSEL1[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
		rw	rw	rw	rw	rw	rw								

位 31:14 保留，必须保持复位值。

位 13:11 **ATOSEL2[2:0]**：主动入侵共用输出 2 选择 (Active tamper shared output 2 selection)

000：TAMPOUTSEL2 = TAMP_OUT1
001：TAMPOUTSEL2 = TAMP_OUT2
010：TAMPOUTSEL2 = TAMP_OUT3
011：TAMPOUTSEL2 = TAMP_OUT4
100：TAMPOUTSEL2 = TAMP_OUT5
101：TAMPOUTSEL2 = TAMP_OUT6
110：TAMPOUTSEL2 = TAMP_OUT7
111：TAMPOUTSEL2 = TAMP_OUT8
如果封装引脚排列中未提供 TAMP_OUTx 输出，则输出选择值保留，不得使用。
位 12:11 是 TAMP_ATCR1 中 ATOSEL2[1:0] 的镜像，因此也可以通过 TAMP_ATCR1 对其进行读/写操作。

位 10:8 **ATOSEL1[2:0]**：主动入侵共用输出 1 选择 (Active tamper shared output 1 selection)

000：TAMPOUTSEL1 = TAMP_OUT1
001：TAMPOUTSEL1 = TAMP_OUT2
010：TAMPOUTSEL1 = TAMP_OUT3
011：TAMPOUTSEL1 = TAMP_OUT4
100：TAMPOUTSEL1 = TAMP_OUT5
101：TAMPOUTSEL1 = TAMP_OUT6
110：TAMPOUTSEL1 = TAMP_OUT7
111：TAMPOUTSEL1 = TAMP_OUT8
如果封装引脚排列中未提供 TAMP_OUTx 输出，则输出选择值保留，不得使用。
位 9:8 是 TAMP_ATCR1 中 ATOSEL1[1:0] 的镜像，因此也可以
通过 TAMP_ATCR1 对其进行读/写操作。

位 7:0 保留，必须保持复位值。

注意： *当 TAMPxAM 位置 1 时，不允许更改该寄存器中的主动入侵配置，除非 TAMP_CR1 寄存器中所有相应的 TAMPxE 位都清零。*

配置为主动模式的所有入侵必须同时使能（通过在同一个 TAMP_CR1 写访问中将所有相关的 TAMPxE 都置 1）。

配置为主动模式的所有入侵必须同时禁止（通过在同一个 TAMP_CR1 写访问中将所有相关的 TAMPxE 都清零）。

禁止主动入侵后，必须等待至少 1 个 CK_ATPRE 周期才能将其重新使能。

33.6.9 TAMP 配置寄存器 (TAMP_CFGR)

TAMP configuration register

既可以为该寄存器设置全局写保护，也可以为该寄存器的每个位单独设置写保护，以防其遭到非特权访问，具体取决于 TAMP_PRIVCFGR 配置（请参见*第 33.3.6 节：TAMP 特权保护模式*）。

偏移地址：0x20

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31 保留，必须保持复位值。

位 30 保留，必须保持复位值。

位 29:24 保留，必须保持复位值。

位 23:16 **BKPW[7:0]**：备份寄存器写保护偏移量 (Backup registers write protection offset)
BKPW值必须介于 0 到 32 之间。

保护区 2 针对备份寄存器 TAMP_BKPyR (y = BKPRW) 到 TAMP_BKPzR（z = BKPW-1，其中 BKPW > BKPRW）定义：
如果 BKPWSEC = 0 或 BKPWSEC ≤ BKPRWSEC：无保护区 2。

保护区 3 针对备份寄存器 TAMP_BKPtR（BKPWSEC ≥ BKPRWSEC 时，t = BKPW，否则 t = BKPRWSEC）定义。
如果 BKPWSEC = 32：无保护区 3。

请参见*图 370：备份寄存器保护区*。

注意：如果 BKPWPRIV 置 1，则只能在特权模式下写入 BKPRW[7:0]。

位 15 保留，必须保持复位值。

位 14:8 保留，必须保持复位值。

位 7:0 **BKPRW[7:0]**：备份寄存器读/写保护偏移量 (Backup registers read/write protection offset)
BKPRW值必须介于 0 到 32 之间。

保护区 1 针对备份寄存器 TAMP_BKP0R 到 TAMP_BKPxR（x = BKPRW-1，其中 BKPRW ≥ 1）定义。
如果 BKPRW = 0：无保护区 1。

请参见*图 370：备份寄存器保护区*。

注意：如果 BKPRWPRIV 置 1，则只能在特权模式下写入 BKPRW[7:0]。

33.6.10 TAMP 特权配置寄存器 (TAMP_PRIVCFGR)

TAMP privilege configuration register

该寄存器只能通过 APB 特权访问进行写操作。

偏移地址：0x24

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TAMPPRIV	BKPWPRIV	BKPRWPRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
rw	rw	rw													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT1PRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
rw															

位 31 **TAMPPRIV**：入侵特权保护（不包括备份寄存器）(Tamper privilege protection (excluding backup registers))

0：入侵配置和中断可以通过特权或非特权访问进行写操作。

1：入侵配置和中断只能通过特权访问进行写操作。

注意：有关读保护的详细信息，请参见第 33.3.6 节：TAMP 特权保护模式。

位 30 **BKPWPRIV**：备份寄存器区域 2 特权保护 (Backup registers zone 2 privilege protection)

0：备份寄存器区域 2 可以通过特权或非特权访问进行写操作。

1：备份寄存器区域 2 只能通过特权访问进行写操作。

位 29 **BKPRWPRIV**：备份寄存器区域 1 特权保护 (Backup registers zone 1 privilege protection)

0：备份寄存器区域 1 可以通过特权或非特权访问进行读写操作。

1：备份寄存器区域 1 只能通过特权访问进行读写操作。

位 28:16 保留，必须保持复位值。

位 15 **CNT1PRIV**：单调计数器 1 特权保护 (Monotonic counter 1 privilege protection)

0：单调计数器 1 (TAMP_COUNT1R) 可以通过 APB 特权或非特权访问进行读写操作。

1：单调计数器 1 (TAMP_COUNT1R) 只能通过 APB 特权访问进行读写操作。

位 14:0 保留，必须保持复位值。

33.6.11 TAMP 中断使能寄存器 (TAMP_IER)

TAMP interrupt enable register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x2C

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	ITAMP15IE	Res.	ITAMP13IE	ITAMP12IE	ITAMP11IE	Res.	ITAMP9IE	ITAMP8IE	ITAMP7IE	ITAMP6IE	ITAMP5IE	ITAMP4IE	ITAMP3IE	ITAMP2IE	ITAMP1IE
	rw		rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2IE	TAMP1IE
														rw	rw

位 31 保留，必须保持复位值。

位 30 **ITAMP15IE**：内部入侵 15 中断使能 (Internal tamper 15 interrupt enable)
0：禁止内部入侵 15 中断。
1：使能内部入侵 15 中断。

位 29 保留，必须保持复位值。

位 28 **ITAMP13IE**：内部入侵 13 中断使能 (Internal tamper 13 interrupt enable)
0：禁止内部入侵 13 中断。
1：使能内部入侵 13 中断。

位 27 **ITAMP12IE**：内部入侵 12 中断使能 (Internal tamper 12 interrupt enable)
0：禁止内部入侵 12 中断。
1：使能内部入侵 12 中断。

位 26 **ITAMP11IE**：内部入侵 11 中断使能 (Internal tamper 11 interrupt enable)
0：禁止内部入侵 11 中断。
1：使能内部入侵 11 中断。

位 25 保留，必须保持复位值。

位 24 **ITAMP9IE**：内部入侵 9 中断使能 (Internal tamper 9 interrupt enable)
0：禁止内部入侵 9 中断。
1：使能内部入侵 9 中断。

位 23 **ITAMP8IE**：内部入侵 8 中断使能 (Internal tamper 8 interrupt enable)
0：禁止内部入侵 8 中断。
1：使能内部入侵 8 中断。

位 22 **ITAMP7IE**：内部入侵 7 中断使能 (Internal tamper 7 interrupt enable)
0：禁止内部入侵 7 中断。
1：使能内部入侵 7 中断。

位 21 **ITAMP6IE**：内部入侵 6 中断使能 (Internal tamper 6 interrupt enable)
0：禁止内部入侵 6 中断。
1：使能内部入侵 6 中断。

位 20 **ITAMP5IE**：内部入侵 5 中断使能 (Internal tamper 5 interrupt enable)
0：禁止内部入侵 5 中断。
1：使能内部入侵 5 中断。

位 19 **ITAMP4IE**：内部入侵 4 中断使能 (Internal tamper 4 interrupt enable)
0：禁止内部入侵 4 中断。
1：使能内部入侵 4 中断。

位 18 **ITAMP3IE**：内部入侵 3 中断使能 (Internal tamper 3 interrupt enable)
0：禁止内部入侵 3 中断。
1：使能内部入侵 3 中断。

位 17 **ITAMP2IE**：内部入侵 2 中断使能 (Internal tamper 2 interrupt enable)
0：禁止内部入侵 2 中断。
1：使能内部入侵 2 中断。

位 16 **ITAMP1IE**：内部入侵 1 中断使能 (Internal tamper 1 interrupt enable)
0：禁止内部入侵 1 中断。
1：使能内部入侵 1 中断。

位 15:8 保留，必须保持复位值。

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 保留，必须保持复位值。

位 4 保留，必须保持复位值。

位 3 保留，必须保持复位值。

位 2 保留，必须保持复位值。

位 1 **TAMP2IE**：入侵 2 中断使能 (Tamper 2 interrupt enable)
0：禁止入侵 2 中断。
1：使能入侵 2 中断。

位 0 **TAMP1IE**：入侵 1 中断使能 (Tamper 1 interrupt enable)
0：禁止 1 入侵中断。
1：使能入侵 1 中断。

33.6.12 TAMP 状态寄存器 (TAMP_SR)

TAMP status register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x30

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	ITAMP15F	Res.	ITAMP13F	ITAMP12F	ITAMP11F	Res.	ITAMP9F	ITAMP8F	ITAMP7F	ITAMP6F	ITAMP5F	ITAMP4F	ITAMP3F	ITAMP2F	ITAMP1F
	rw		r	r	r		r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2F	TAMP1F
														r	r

位 31 保留，必须保持复位值。

位 30 **ITAMP15F**：内部入侵 15 标志 (Internal tamper 15 flag)
在内部入侵 15 上检测到入侵检测事件时，由硬件将该标志置 1。

位 29 保留，必须保持复位值。

位 28 **ITAMP13F**：内部入侵 13 标志 (Internal tamper 13 flag)
在内部入侵 13 上检测到入侵检测事件时，由硬件将该标志置 1。

位 27 **ITAMP12F**：内部入侵 12 标志 (Internal tamper 12 flag)
在内部入侵 12 上检测到入侵检测事件时，由硬件将该标志置 1。

位 26 **ITAMP11F**：内部入侵 11 标志 (Internal tamper 11 flag)
在内部入侵 11 上检测到入侵检测事件时，由硬件将该标志置 1。

位 25 保留，必须保持复位值。

位 24 **ITAMP9F**：内部入侵 9 标志 (Internal tamper 9 flag)
在内部入侵 9 上检测到入侵检测事件时，由硬件将该标志置 1。

位 23 **ITAMP8F**：内部入侵 8 标志 (Internal tamper 8 flag)
在内部入侵 8 上检测到入侵检测事件时，由硬件将该标志置 1。

位 22 **ITAMP7F**：内部入侵 7 标志 (Internal tamper 7 flag)
在内部入侵 7 上检测到入侵检测事件时，由硬件将该标志置 1。

位 21 **ITAMP6F**：内部入侵 6 标志 (Internal tamper 6 flag)
在内部入侵 6 上检测到入侵检测事件时，由硬件将该标志置 1。

位 20 **ITAMP5F**：内部入侵 5 标志 (Internal tamper 5 flag)
在内部入侵 5 上检测到入侵检测事件时，由硬件将该标志置 1。

位 19 **ITAMP4F**：内部入侵 4 标志 (Internal tamper 4 flag)
在内部入侵 4 上检测到入侵检测事件时，由硬件将该标志置 1。

位 18 **ITAMP3F**：内部入侵 3 标志 (Internal tamper 3 flag)
在内部入侵 3 上检测到入侵检测事件时，由硬件将该标志置 1。

位 17 **ITAMP2F**：内部入侵 2 标志 (Internal tamper 2 flag)
在内部入侵 2 上检测到入侵检测事件时，由硬件将该标志置 1。

位 16 **ITAMP1F**：内部入侵 1 标志 (Internal tamper 1 flag)
在内部入侵 1 上检测到入侵检测事件时，由硬件将该标志置 1。

位 15:8 保留，必须保持复位值。

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 保留，必须保持复位值。

位 4 保留，必须保持复位值。

位 3 保留，必须保持复位值。

位 2 保留，必须保持复位值。

位 1 **TAMP2F**：TAMP2 检测标志 (TAMP2 detection flag)
在 TAMP2 输入上检测到入侵检测事件时，由硬件将该标志置 1。

位 0 **TAMP1F**：TAMP1 检测标志 (TAMP1 detection flag)
在 TAMP1 输入上检测到入侵检测事件时，由硬件将该标志置 1。

33.6.13 TAMP 屏蔽中断状态寄存器 (TAMP_MISR)

TAMP masked interrupt status register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x34

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	ITAMP15MF	Res.	ITAMP13MF	ITAMP12MF	ITAMP11MF	Res.	ITAMP9MF	ITAMP8MF	ITAMP7MF	ITAMP6MF	ITAMP5MF	ITAMP4MF	ITAMP3MF	ITAMP2MF	ITAMP1MF
	r		r	r	r		r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2MF	TAMP1MF
														r	r

位 31 保留，必须保持复位值。

位 30 **ITAMP15MF**：内部入侵 15 中断屏蔽标志 (internal tamper 15 interrupt masked flag)
发生内部入侵 15 中断时，由硬件将该标志置 1。

位 29 保留，必须保持复位值。

位 28 **ITAMP13MF**：内部入侵 13 中断屏蔽标志 (internal tamper 13 interrupt masked flag)
发生内部入侵 13 中断时，由硬件将该标志置 1。

位 27 **ITAMP12MF**：内部入侵 12 中断屏蔽标志 (internal tamper 12 interrupt masked flag)
发生内部入侵 12 中断时，由硬件将该标志置 1。

位 26 **ITAMP11MF**：内部入侵 11 中断屏蔽标志 (internal tamper 11 interrupt masked flag)
发生内部入侵 11 中断时，由硬件将该标志置 1。

位 25 保留，必须保持复位值。

位 24 **ITAMP9MF**：内部入侵 9 中断屏蔽标志 (internal tamper 9 interrupt masked flag)
发生内部入侵 9 中断时，由硬件将该标志置 1。

位 23 **ITAMP8MF**：内部入侵 8 中断屏蔽标志 (Internal tamper 8 interrupt masked flag)
发生内部入侵 8 中断时，由硬件将该标志置 1。

位 22 **ITAMP7MF**：内部入侵 7 中断屏蔽标志 (Internal tamper 7 interrupt masked flag)
发生内部入侵 7 中断时，由硬件将该标志置 1。

位 21 **ITAMP6MF**：内部入侵 6 中断屏蔽标志 (Internal tamper 6 interrupt masked flag)
发生内部入侵 6 中断时，由硬件将该标志置 1。

位 20 **ITAMP5MF**：内部入侵 5 中断屏蔽标志 (Internal tamper 5 interrupt masked flag)
发生内部入侵 5 中断时，由硬件将该标志置 1。

位 19 **ITAMP4MF**：内部入侵 4 中断屏蔽标志 (Internal tamper 4 interrupt masked flag)
发生内部入侵 4 中断时，由硬件将该标志置 1。

位 18 **ITAMP3MF**：内部入侵 3 中断屏蔽标志 (Internal tamper 3 interrupt masked flag)
发生内部入侵 3 中断时，由硬件将该标志置 1。

位 17 **ITAMP2MF**：内部入侵 2 中断屏蔽标志 (Internal tamper 2 interrupt masked flag)
发生内部入侵 2 中断时，由硬件将该标志置 1。

位 16 **ITAMP1MF**：内部入侵 1 中断屏蔽标志 (Internal tamper 1 interrupt masked flag)
发生内部入侵 1 中断时，由硬件将该标志置 1。

位 15:8 保留，必须保持复位值。

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 保留，必须保持复位值。

位 4 保留，必须保持复位值。

位 3 保留，必须保持复位值。

位 2 保留，必须保持复位值。

位 1 **TAMP2MF**：TAMP2 中断屏蔽标志 (TAMP2 interrupt masked flag)
发生入侵 2 中断时，由硬件将该标志置 1。

位 0 **TAMP1MF**：TAMP1 中断屏蔽标志 (TAMP1 interrupt masked flag)
发生入侵 1 中断时，由硬件将该标志置 1。

33.6.14 TAMP 状态清零寄存器 (TAMP_SCR)

TAMP status clear register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x3C

系统复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	CITAMP15F	Res.	CITAMP13F	CITAMP12F	CITAMP11F	Res.	CITAMP9F	CITAMP8F	CITAMP7F	CITAMP6F	CITAMP5F	CITAMP4F	CITAMP3F	CITAMP2F	CITAMP1F
	w		w	w	w		w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CTAMP2F	CTAMP1F
														w	w

位 31 保留，必须保持复位值。

位 30 **CITAMP15F**：清零 ITAMP15 检测标志 (Clear ITAMP15 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP15F 标志清零。

位 29 保留，必须保持复位值。

位 28 **CITAMP13F**：清零 ITAMP13 检测标志 (Clear ITAMP13 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP13F 标志清零。

位 27 **CITAMP12F**：清零 ITAMP12 检测标志 (Clear ITAMP12 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP12F 标志清零。

位 26 **CITAMP11F**：清零 ITAMP11 检测标志 (Clear ITAMP11 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP11F 标志清零。

位 25 保留，必须保持复位值。

位 24 **CITAMP9F**：清零 ITAMP9 检测标志 (Clear ITAMP9 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP9F 标志清零。

位 23 **CITAMP8F**：清零 ITAMP8 检测标志 (Clear ITAMP8 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP8F 标志清零。

位 22 **CITAMP7F**：清零 ITAMP7 检测标志 (Clear ITAMP7 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP7F 标志清零。

位 21 **CITAMP6F**：清零 ITAMP6 检测标志 (Clear ITAMP6 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP6F 标志清零。

位 20 **CITAMP5F**：清零 ITAMP5 检测标志 (Clear ITAMP5 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP5F 标志清零。

位 19 **CITAMP4F**：清零 ITAMP4 检测标志 (Clear ITAMP4 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP4F 标志清零。

位 18 **CITAMP3F**：清零 ITAMP3 检测标志 (Clear ITAMP3 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP3F 标志清零。

位 17 **CITAMP2F**：清零 ITAMP2 检测标志 (Clear ITAMP2 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP2F 标志清零。

位 16 **CITAMP1F**：清零 ITAMP1 检测标志 (Clear ITAMP1 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 ITAMP1F 标志清零。

位 15:8 保留，必须保持复位值。

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 保留，必须保持复位值。

位 4 保留，必须保持复位值。

位 3 保留，必须保持复位值。

位 2 保留，必须保持复位值。

位 1 **CTAMP2F**：清零 TAMP2 检测标志 (Clear TAMP2 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 TAMP2F 标志清零。

位 0 **CTAMP1F**：清零 TAMP1 检测标志 (Clear TAMP1 detection flag)
向该位写入 1 可将 TAMP_SR 寄存器中的 TAMP1F 标志清零。

33.6.15 TAMP 单调计数器 1 寄存器 (TAMP_COUNT1R)

TAMP monotonic counter 1 register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x040

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31:0 **COUNT[31:0]**：
该寄存器为只读，当对该寄存器进行写访问时，该寄存器递增 1。该寄存器无法计满返回，当计数到最大值时将被冻结。

33.6.16 TAMP 资源保护配置寄存器 (TAMP_RPCFGR)

TAMP resources protection configuration register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x54

备份域复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RP CFG0
															rw

位 31 保留，必须保持复位值。

位 30:8 保留，必须保持复位值。

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 保留，必须保持复位值。

位 4 保留，必须保持复位值。

位 3 保留，必须保持复位值。

位 2 保留，必须保持复位值。

位 1 保留，必须保持复位值。

位 0 **RPCFG0**：可配置资源 0 保护 (Configurable resource 0 protection)(1)

0： 资源 0 不包含在由 TAMP 外设保护的器件机密信息中

1：资源 0 包含在由 TAMP 外设保护的器件机密信息中

1. 请参见*表 240：TAMP 输入/输出引脚*和*表 242：TAMP 互连*中的 tamp_confirmed_rpcfg0 和 tamp_potential_rpcfg0 信号。

33.6.17 TAMP 备份 x 寄存器 (TAMP_BKPxR)

TAMP backup x register

可对该寄存器设置保护，以防其遭到非特权访问。请参见*第 33.3.6 节：TAMP 特权保护模式*。

偏移地址：0x100 + 0x04 * x，（x = 0 到 31）

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31:0 **BKP[31:0]**：

应用可向/从这些寄存器写入/读取数据。

在默认（擦除）配置中，该寄存器将在发生入侵检测事件时复位。只要至少有一个内部或外部入侵标志置 1，该寄存器就会强制设为复位值。

33.6.18 TAMP 寄存器映射

表 250. TAMP 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TAMP_CR1	Res.	ITAMP15E	Res.	ITAMP13E	ITAMP12E	ITAMP11E	Res.	ITAMP9E	ITAMP8E	ITAMP7E	ITAMP6E	ITAMP5E	ITAMP4E	ITAMP3E	ITAMP2E	ITAMP1E	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2E	TAMP1E
	Reset value		0		0	0	0		0	0	0	0	0	0	0	0	0															0	0
0x04	TAMP_CR2	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2TRG	TAMP1TRG	BKERASE	BKBLOCK	Res.	Res.	Res.	Res.	TAMP2MSK	TAMP1MSK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2POM	TAMP1POM
	Reset value							0	0	0	0					0	0															0	0
0x08	TAMP_CR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITAMP15POM	Res.	ITAMP13POM	ITAMP12POM	ITAMP11POM	Res.	ITAMP9POM	ITAMP8POM	ITAMP7POM	ITAMP6POM	ITAMP5POM	ITAMP4POM	ITAMP3POM	ITAMP2POM	ITAMP1POM
	Reset value																		0		0	0	0		0	0	0	0	0	0	0	0	0
0x0C	TAMP_FLTCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMPPUDIS	TAMPPRCH[1:0]		TAMPFLT[1:0]		TAMPFREQ[2:0]		
	Reset value																									0	0	0	0	0	0	0	0
0x10	TAMP_ATCR1	FLTEN	ATOSHARE	Res.	Res.	Res.	AT PER[2:0]			Res.	Res.	Res.	Res.	ATCK SEL[3:0]				Res.	Res.	Res.	Res.	ATO SEL2 [1:0]		ATO SEL1 [1:0]		Res.	Res.	Res.	Res.	Res.	Res.	TAMP2AM	TAMP1AM
	Reset value	0	0				0	0	0					0	1	1	1					0	0	0	0							0	0
0x10	TAMP_ATCR1	FLTEN	ATOSHARE	Res.	Res.	Res.	AT PER[2:0]			Res.	Res.	Res.	Res.	ATCK SEL[3:0]				ATO SEL4 [1:0]		ATO SEL3 [1:0]		ATO SEL2 [1:0]		ATO SEL1 [1:0]		Res.	Res.	Res.	Res.	Res.	Res.	TAMP2AM	TAMP1AM
	Reset value	0	0				0	0	0					0	1	1	1	0	0	0	0	0	0	0	0							0	0
0x14	TAMP_ATSEEDR	SEED[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	TAMP_ATOR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INITS	SEEDF	Res.	Res.	Res.	Res.	Res.	Res.	PRNG[7:0]							
	Reset value																	0	0							0	0	0	0	0	0	0	0
0x1C	TAMP_ATCR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATO SEL2 [2:0]			ATO SEL1 [2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																			0	0	0	0	0	0								
0x20	TAMP_CFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BKPW[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BKPRW[7:0]							
	Reset value									0	0	0	0	0	0	0	0									0	0	0	0	0	0	0	0

表 250. TAMP 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x24	TAMP_PRIVCFGR	TAMPPRIV	BKPWPRIV	BKPRWPRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT1PRIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value	0	0	0														0															
0x2C	TAMP_IER	Res.	ITAMP15IE	Res.	ITAMP13IE	ITAMP12IE	ITAMP11IE	Res.	ITAMP9IE	ITAMP8IE	ITAMP7IF	ITAMP6IE	ITAMP5IE	ITAMP4IE	ITAMP3IE	ITAMP2IE	ITAMP1IE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2IE	TAMP1IE
	Reset value		0		0	0	0		0	0	0	0	0	0	0	0	0														0	0
0x30	TAMP_SR	Res.	ITAMP15F	Res.	ITAMP13F	ITAMP12F	ITAMP11F	Res.	ITAMP9F	ITAMP8F	ITAMP7F	ITAMP6F	ITAMP5F	ITAMP4F	ITAMP3F	ITAMP2F	ITAMP1F	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2F	TAMP1F
	Reset value		0		0	0	0		0	0	0	0	0	0	0	0	0														0	0
0x34	TAMP_MISR	Res.	ITAMP15MF	Res.	ITAMP13MF	ITAMP12MF	ITAMP11MF	Res.	ITAMP9MF	ITAMP8MF	ITAMP7MF	ITAMP6MF	ITAMP5MF	ITAMP4MF	ITAMP3MF	ITAMP2MF	ITAMP1MF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP2MF	TAMP1MF
	Reset value		0		0	0	0		0	0	0	0	0	0	0	0	0														0	0
0x3C	TAMP_SCR	Res.	CITAMP15F	Res.	CITAMP13F	CITAMP12F	CITAMP11F	Res.	CITAMP9F	CITAMP8F	CITAMP7F	CITAMP6F	CITAMP5F	CITAMP4F	CITAMP3F	CITAMP2F	CITAMP1F	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CTAMP2F	CTAMP1F
	Reset value		0		0	0	0		0	0	0	0	0	0	0	0	0														0	0
0x40	TAMP_COUNTR	COUNT[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	TAMP_OR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IN4_RMP	IN3_RMP	IN2_RMP	Res.	Res.	Res.	Res.	OUT5_RMP	OUT3_RMP		Res.
	Reset value																						0	0	0					0	0	0	
0x54	TAMP_RPCFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RPCFG0
	Reset value																																0
0x100 + 0x04*x, (x= 0 to 31)	TAMP_BKPxR	BKP[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 2.2 节*。

34 内部集成电路 (I2C) 接口

34.1 简介

I^2C（内部集成电路）总线接口处理微控制器与串行 I^2C 总线间的通信。它提供多主模式功能，可以控制所有 I^2C 总线特定的序列、协议、仲裁和时序。它支持标准模式 (Sm)、快速模式 (Fm) 和超快速模式 (Fm+)。

I^2C 总线接口还与 SMBus（系统管理总线）和 PMBus®（电源管理总线）兼容。

可使用 DMA 来减轻 CPU 的工作量。

34.2 I2C 主要特性

- 兼容 I^2C 总线规范第 03 版：
  - 从模式和主模式
  - 多主模式功能
  - 标准速度模式（高达 100 kHz）
  - 快速模式（高达 400 kHz）
  - 超快速模式（高达 1 MHz）
  - 7 位和 10 位寻址模式
  - 多个 7 位从地址（2 个从设备地址寄存器，1 个具有可配置的掩码位段）
  - 所有 7 位地址应答模式
  - 广播呼叫
  - 总线上的数据建立和保持时间可软件配置
  - 方便易用的事件管理
  - 可选的时钟延长
  - 软件复位
- 带 DMA 功能的 1 字节缓冲
- 可编程模拟和数字噪声滤波器

还可提供以下特性，具体取决于产品实现（请参见*第 34.3 节*）：

- 兼容 SMBus 规范第 3.0 版：
  - 具有 ACK 控制的硬件 PEC（数据包错误校验）生成和验证
  - 命令和数据应答控制
  - 支持地址解析协议 (ARP)
  - 支持主机和从设备
  - SMBus 报警
  - 超时和空闲条件检测
- 兼容 PMBus 第 1.3 版标准
- 独立时钟：选择独立时钟源可使 I2C 通信速度不受 i2c_pclk 重新编程的影响
- 地址匹配时从停止模式唤醒

34.3 I2C 特性实现

表 251. STM32H503xx I2C 特性实现

I2C 特性(1)	I2C1	I2C2
7 位寻址模式	X	X
10 位寻址模式	X	X
标准速度模式（高达 100 kbit/s）	X	X
快速模式（高达 400 kb/s）	X	X
超快速模式，20 mA 输出驱动 I/O（高达 1 Mb/s）	X	X
独立时钟	X	X
仅从停止模式唤醒（非自主模式）	X	X
SMBus/PMBus	X	X

1. X = 支持。

34.4 I2C 功能描述

除了接收和发送数据之外，此接口还可以从串行格式转换为并行格式，反之亦然。中断由软件使能或禁止。该接口通过数据引脚 (SDA) 和时钟引脚 (SCL) 连接到 I^2C 总线。它可以连接到标准速度（高达 100 kHz）、快速（高达 400 kHz）或超快速（高达 1 MHz）I^2C 总线。

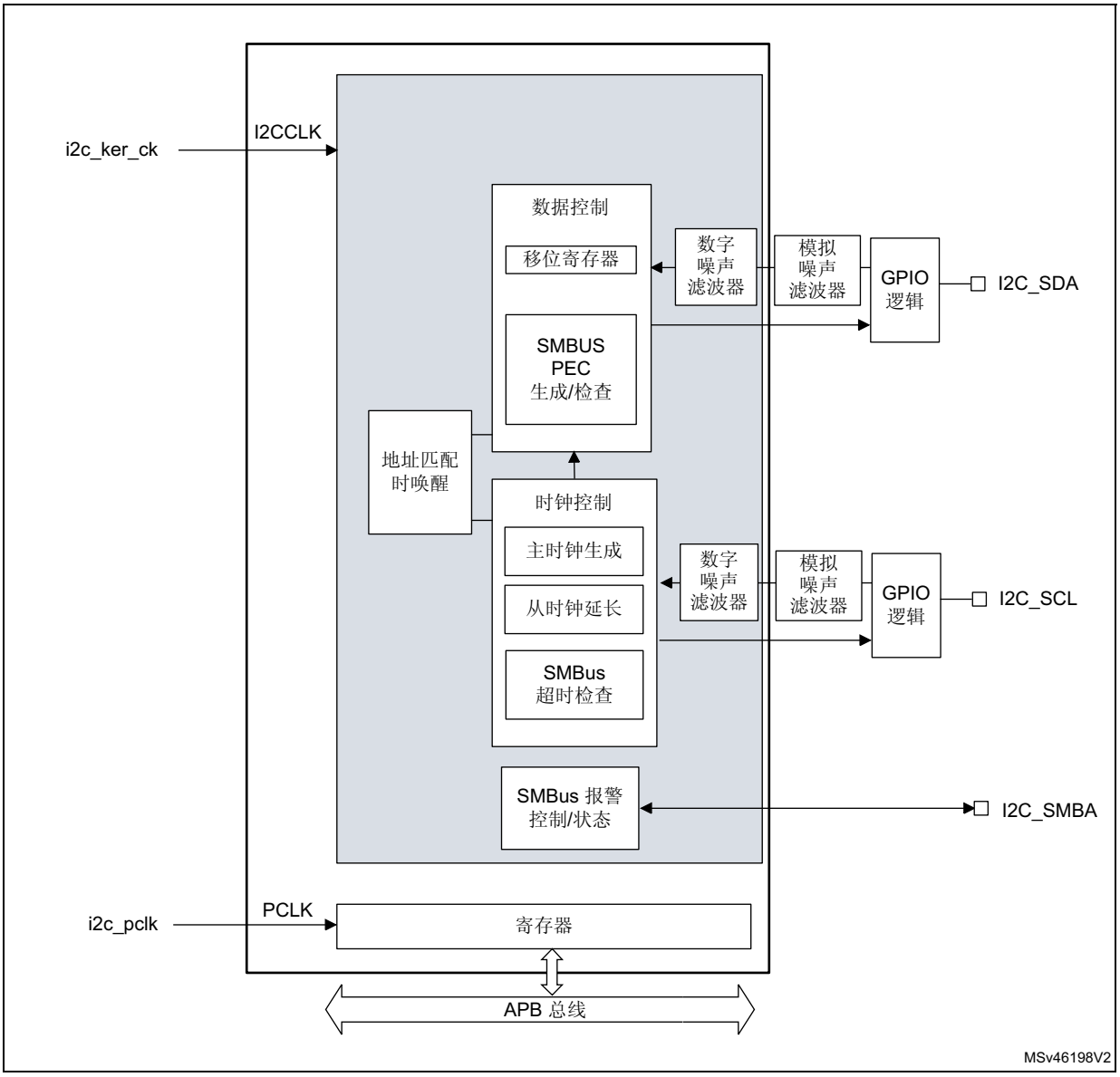
该接口也可通过数据 (SDA) 引脚和时钟 (SCL) 引脚连接到 SMBus。

如果支持 SMBus 功能：还可选用 SMBus 报警引脚 (SMBA)。

34.4.1 I2C 框图

I2C 接口如*图 372* 所示。

图 372. I2C 框图



I2C 的时钟由独立时钟源提供，这使得 I2C 能够独立于 i2c_pclk 频率工作。

34.4.2 I2C 引脚和内部信号

表 252. I2C 输入/输出引脚

引脚名称	信号类型	说明
I2C_SDA	双向	I2C 数据
I2C_SCL	双向	I2C 时钟
I2C_SMBA	双向	SMBus 报警

表 253. I2C 内部输入/输出信号

内部信号名称	信号类型	说明
i2c_ker_ck	输入	I2C 内核时钟，在本文档中也称为 I2CCLK
i2c_pclk	输入	I2C APB 时钟
i2c_it	输出	I2C 中断，有关中断源的列表，请参见*表 266*
i2c_rx_dma	输出	I2C 接收数据 DMA 请求 (I2C_RX)
i2c_tx_dma	输出	I2C 发送数据 DMA 请求 (I2C_TX)

34.4.3 I2C 时钟要求

I2C 内核的时钟由 i2c_ker_ck 提供。

i2c_ker_ck 周期 t_{I2CCLK} 必须遵循以下条件：

- $t_{I2CCLK} < (t_{LOW} - t_{filters}) / 4$
- $t_{I2CCLK} < t_{HIGH}$

其中：

t_{LOW}：SCL 低电平时间，t_{HIGH}：SCL 高电平时间

$t_{filters}$：滤波器使能时，该值为模拟滤波器和数字滤波器引入的延时总和。

数字滤波器延时为 DNF x t_{I2CCLK}。

i2c_pclk 时钟周期 t_{PCLK} 必须遵循以下条件：

- $t_{PCLK} < 4 / 3\ t_{SCL}$（$t_{SCL}$：SCL 周期）

小心： 当 I2C 内核的时钟由 i2c_pclk 提供时，该时钟必须遵循 t_{I2CCLK} 的条件。

34.4.4 模式选择

该接口在工作时可选用以下四种模式之一：

- 从发送器
- 从接收器
- 主发送器
- 主接收器

默认情况下，它以从模式工作。接口在生成起始位后会自动由从模式切换为主模式，并在出现仲裁丢失或生成停止位时从主模式切换为从模式，从而实现多主模式功能。

通信流程

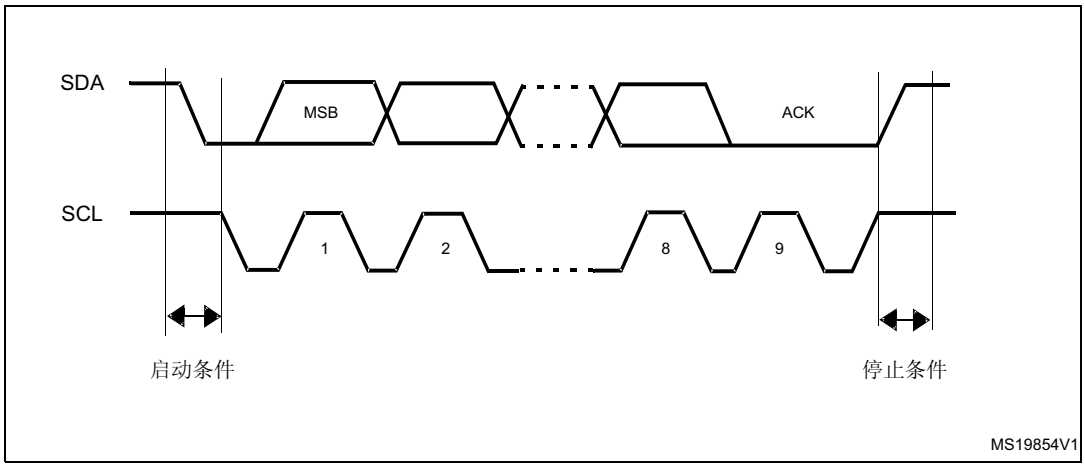
在主模式下，I2C 接口会启动数据传输并生成时钟信号。串行数据传输始终是在出现起始位时开始，在出现停止位时结束。起始位和停止位均在主模式下由软件生成。

在从模式下，该接口能够识别其自身地址（7 或 10 位）以及广播呼叫地址。广播呼叫地址检测可由软件使能或禁止。保留的 SMBus 地址也可由软件使能。

数据和地址均以 8 位字节传输，MSB 在前。起始位后紧随地址字节（7 位地址占据一个字节；10 位地址占据两个字节）。地址始终在主模式下发送。

在字节传输八个时钟周期后是第九个时钟脉冲，在此期间接收器必须向发送器发送一个应答位（请参见*图 373*）。

图 373. I²C 总线协议



应答位可由软件使能或禁止。I2C 接口地址可通过软件进行选择。

34.4.5 I2C 初始化

使能和禁止外设

必须先在时钟控制器中配置并使能 I2C 外设时钟，之后才能通过将 I2C_CR1 寄存器中的 PE 位置 1 来使能 I2C。

当禁止 I2C (PE = 0) 时，I^2C 将执行软件复位。更多详细信息，请参见*第 34.4.6 节*。

噪声滤波器

通过将 I2C_CR1 寄存器中的 PE 位置 1 来使能 I2C 外设之前，如有必要，用户必须配置噪声滤波器。默认情况下，SDA 和 SCL 输入上开启了模拟噪声滤波器。该滤波器符合 I^2C 规范，此规范要求在快速模式和超快速模式下对脉宽在 50ns 以下的脉冲都要抑制。用户可通过将 ANFOFF 位置 1 来禁止该模拟滤波器，和/或通过配置 I2C_CR1 寄存器中的 DNF[3:0] 位来选择数字滤波器。

使能数字滤波器时，SCL 或 SDA 线的电平只有在电平稳定时间超过 DNF x i2c_ker_ck 个周期后才会发生内部变化，这样可抑制的尖峰脉宽在 1 到 15 个 i2c_ker_ck 周期可编程。

表 254. 模拟滤波器与数字滤波器对比

-	模拟滤波器	数字滤波器
抑制尖峰的脉冲宽度	≥ 50 ns	长度可编程为 1 到 15 个 I2C 外设时钟
优点	停止模式中仍可用	– 长度可编程：额外的滤波能力与标准要求 – 稳定长度
缺点	随温度、电压和过程变化会发生变化	当使能数字滤波器后，无法在地址匹配时从停止模式唤醒

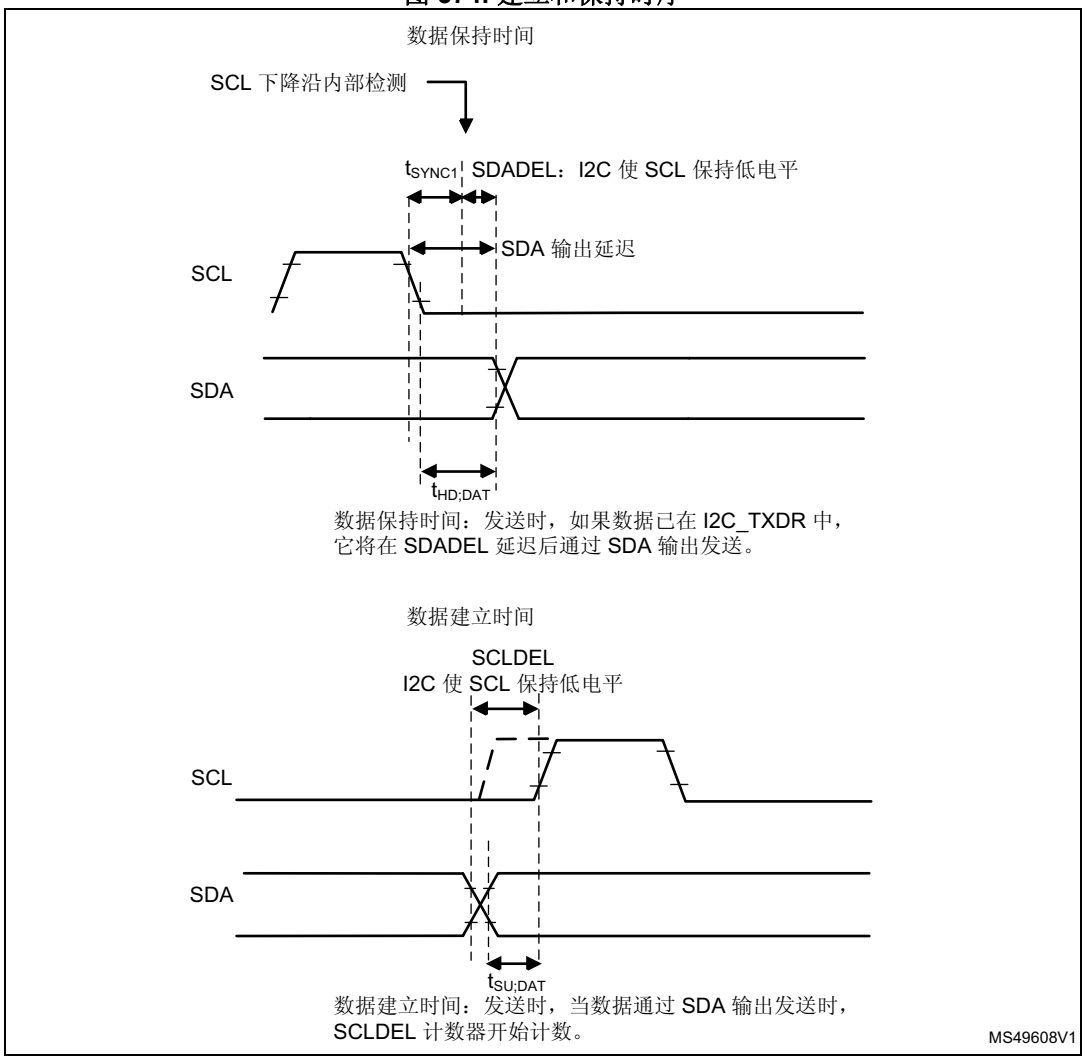
小心： 使能 I2C 时，不能更改滤波器配置。

I2C 时序

必须配置时序，以便保证主模式和从模式下使用正确的数据保持和建立时间。配置方法是编程 I2C_TIMINGR 寄存器中的 PRESC[3:0]、SCLDEL[3:0] 和 SDADEL[3:0] 位。

STM32CubeMX 工具计算 I2C_TIMINGR 内容并在 I2C 配置窗口中进行显示。

图 374. 建立和保持时序



当内部检测到 SCL 下降沿时，会在发送 SDA 输出之前插入一段延时（t_{SDADEL}，这会影响保持时间$t_{HD;DAT}$）：t_{SDADEL} = SDADEL x t_{PRESC} + t_{I2CCLK}，其中 t_{PRESC} = (PRESC + 1) x t_{I2CCLK}。

SDA 总输出延时为：

t_{SYNC1} + {[SDADEL x (PRESC + 1) + 1] x t_{I2CCLK}}

t_{SYNC1} 持续时间取决于：

- SCL 下降斜率
- 模拟滤波器（使能时）引入的输入延时：$t_{AF(min)} < t_{AF} < t_{AF(max)}$
- 数字滤波器（使能时）引入的输入延时：t_{DNF} = DNF x t_{I2CCLK}
- SCL 与 i2c_ker_ck 时钟建立同步而产生的延时（2 到 3 个 i2c_ker_ck 周期）

要桥接 SCL 下降沿的未定义区域，用户编程 SDADEL 时必须遵循以下条件：

{$t_{f\ (max)}$ + $t_{HD;DAT\ (min)}$ - $t_{AF(min)}$ - [(DNF + 3) x t_{I2CCLK}]} / {(PRESC + 1) x t_{I2CCLK}} ≤ SDADEL

SDADEL ≤ {$t_{HD;DAT\ (max)}$ - $t_{AF(max)}$ - [(DNF + 4) x t_{I2CCLK}]} / {(PRESC + 1) x t_{I2CCLK}}

注意： *只有使能模拟滤波器时，公式中才包含 $t_{AF(min)}$ / $t_{AF(max)}$。有关 t_{AF} 值的信息，请参见器件数据手册。*

关于 $t_{HD;DAT}$ 最大值，标准模式下可达 3.45 µs，快速模式可达 0.9 µs，超快速模式下可达 0.45 µs。但必须小于 $t_{VD;DAT}$ 最大值（差值为跳变时间）。只有器件未延长 SCL 信号的低电平周期 (t_{LOW}) 时，才必须满足该最大值条件。如果时钟延长 SCL，数据必须在建立时间内保持有效，之后才能释放时钟。

SDA 上升沿通常为最坏情况。在这种情况下，上述公式变成如下形式：

$SDADEL \leq \{t_{VD;DAT\ (max)} - t_{r\ (max)} - t_{AF\ (max)} - [(DNF + 4) \times t_{I2CCLK}]\} / \{(PRESC + 1) \times t_{I2CCLK}\}$。

注意： *NOSTRETCH=0 时会违反该条件，这是因为器件会根据 SCLDEL 值来延长 SCL 低电平时间，以保证建立时间。*

有关 t_f、t_r、$t_{HD;DAT}$ 和 $t_{VD;DAT}$ 标准值的信息，请参见*表 255*。

- 在 t_{SDADEL} 后，或在因数据未写入 I2C_TXDR 寄存器而导致从器件必须延长时钟时发送 SDA 输出后，SCL 线会在建立时间内保持低电平。该建立时间为 $t_{SCLDEL} = (SCLDEL + 1) \times t_{PRESC}$，其中 $t_{PRESC} = (PRESC + 1) \times t_{I2CCLK}$。$t_{SCLDEL}$ 会影响建立时间 $t_{SU;DAT}$。

要桥接 SDA 跳变（上升沿通常为最坏情况）的未定义区域，编程 SCLDEL 时必须遵循以下条件：

$\{[t_{r\ (max)} + t_{SU;DAT\ (min)}] / [(PRESC + 1)] \times t_{I2CCLK}]\} - 1 \leq SCLDEL$

有关 t_r 和 $t_{SU;DAT}$ 标准值的信息，请参见*表 255*。

将使用的 SDA 和 SCL 跳变时间值就是应用中的值。使用最大值而非标准值会增加 SDADEL 和 SCLDEL 计算的约束条件，但能够确保任意应用的特性。

注意： *在发送和接收模式下，对于每个时钟脉冲，检测到 SCL 下降沿后，I2C 主器件或从器件会至少在 [(SDADEL+SCLDEL+1) x (PRESC+1) + 1] x t_{I2CCLK} 期间内延长 SCL 低电平时间。在发送模式下，如果 SDADEL 计数器计数结束后数据还未写入 I2C_TXDR，则 I2C 会继续延长 SCL 低电平时间，直到写入下一个数据。随后，会将新数据 MSB 发送到 SDA 输出，SCLDEL 计数器将开始计数，同时会继续延长 SCL 低电平时间以确保提供充足的数据建立时间。*

如果从模式下 NOSTRETCH = 1，则 SCL 不会延长。因此，编程 SDADEL 时必须确保提供充足的建立时间。

表 255. I²C-SMBus 规范数据建立和保持时间

符号	参数	标准模式 (Sm)		快速模式 (Fm)		超快速模式 (Fm+)		SMBus		单位
		最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值	
$t_{HD;DAT}$	数据保持时间	0	-	0	-	0	-	0.3	-	µs
$t_{VD;DAT}$	数据有效时间	-	3.45	-	0.9	-	0.45	-	-	
$t_{SU;DAT}$	数据建立时间	250	-	100	-	50	-	250	-	ns
t_r	SDA 和 SCL 信号的下降时间	-	1000	-	300	-	120	-	1000	
t_f	SDA 和 SCL 信号的下降时间	-	300	-	300	-	120	-	300	

此外，在主模式下，必须通过编程 I2C_TIMINGR 寄存器中的 PRESC[3:0]、SCLH[7:0] 和 SCLL[7:0] 位域来配置 SCL 时钟的高电平和低电平。

- 当内部检测到 SCL 下降沿时，会在释放 SCL 输出之前插入一段延时。该延时为 $t_{SCLL} = (SCLL + 1) \times t_{PRESC}$，其中 $t_{PRESC} = (PRESC + 1) \times t_{I2CCLK}$。
  t_{SCLL} 会影响 SCL 低电平时间 t_{LOW}。
- 当内部检测到 SCL 上升沿时，会在将 SCL 输出强制为低电平之前插入一段延时。该延时为 $t_{SCLH} = (SCLH + 1) \times t_{PRESC}$，其中 $t_{PRESC} = (PRESC+ 1) \times t_{I2CCLK}$。$t_{SCLH}$ 会影响 SCL 高电平时间 t_{HIGH}。

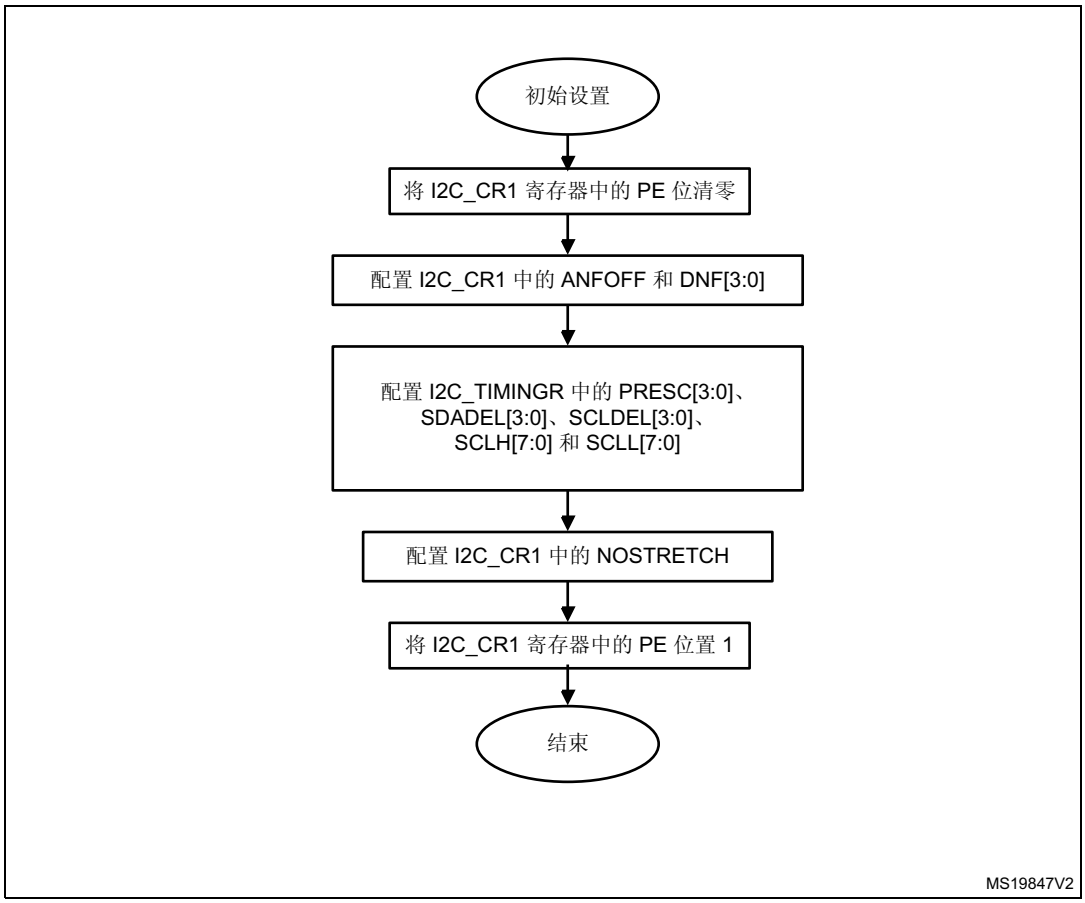
更多详细信息，请参见 *I2C 主模式初始化*。

小心： 使能 I2C 后，不允许更改时序配置。

此外，还必须在使能外设之前配置 I2C 从 NOSTRETCH 模式。更多详细信息，请参见 *I2C 从模式初始化*。

小心： 使能 I2C 后，不允许更改 NOSTRETCH 配置。

图 375. I2C 初始化流程



34.4.6 软件复位

可通过将 I2C_CR1 寄存器中的 PE 位清零来执行软件复位。在这种情况下，I2C 线 SCL 和 SDA 被释放。内部状态机复位，通信控制位和状态位恢复为其复位值。配置寄存器不受影响。

受影响的寄存器位：

1. I2C_CR2 寄存器：START、STOP 和 NACK
2. I2C_ISR 寄存器：BUSY、TXE、TXIS、RXNE、ADDR、NACKF、TCR、TC、STOPF、BERR、ARLO 和 OVR

支持 SMBus 功能时还会影响到以下寄存器位：

1. I2C_CR2 寄存器：PECBYTE
2. I2C_ISR 寄存器：PECERR、TIMEOUT 和 ALERT

必须使 PE 保持低电平持续至少三个 APB 时钟周期，才能成功执行软件复位。使用以下软件序列可确保这一点：

1. 写入 PE = 0
2. 检查 PE = 0
3. 写入 PE = 1

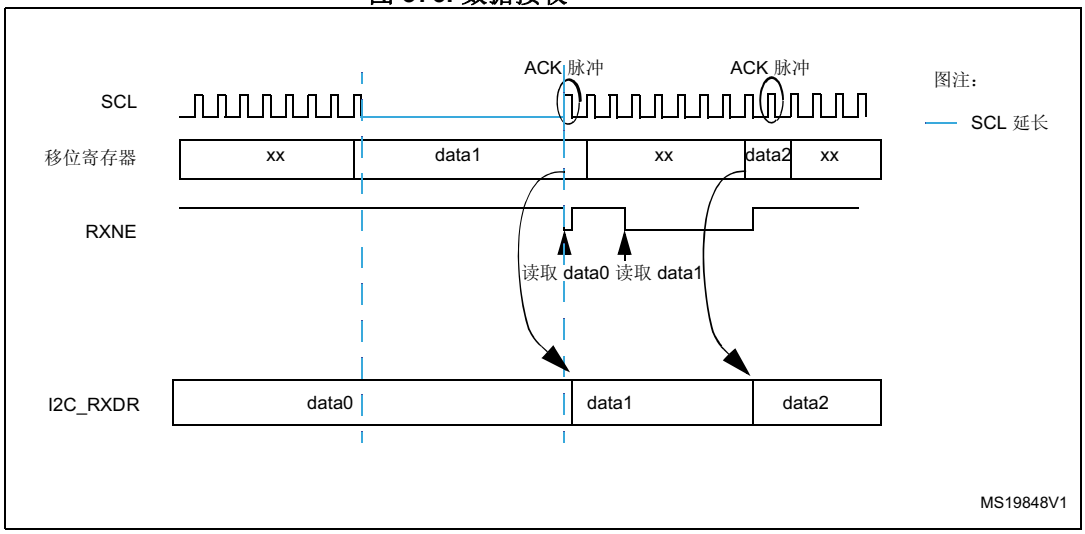
34.4.7 数据传输

数据传输由发送和接收数据寄存器以及移位寄存器来管理。

接收

SDA 输入填充移位寄存器。在第八个 SCL 脉冲后（接收到完整的数据字节时），如果 I2C_RXDR 寄存器为空 (RXNE=0)，则移位寄存器的内容会复制到其中。如果 RXNE = 1（意味着尚未读取上一次接收到的数据字节），则将延长 SCL 线的低电平时间，直到读取了 I2C_RXDR 为止。在第八个和第九个 SCL 脉冲之间（应答脉冲之前）插入一段延长的时间。

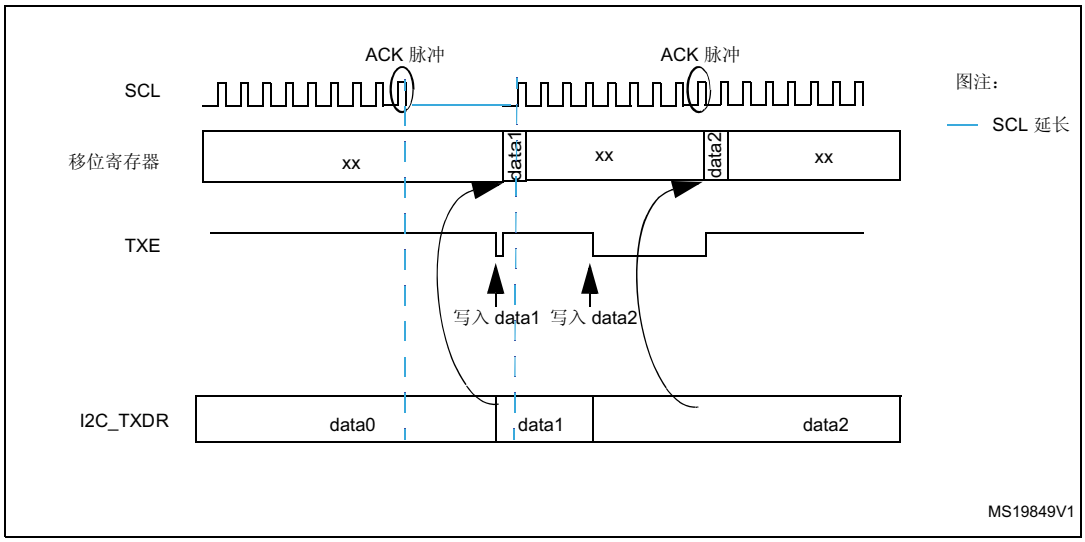
图 376. 数据接收



发送

如果 I2C_TXDR 寄存器不为空 (TXE = 0)，则其内容会在第九个 SCL 脉冲（应答脉冲）后复制到移位寄存器中。然后移位寄存器的内容会移出到 SDA 线上。如果 TXE = 1（意味着 I2C_TXDR 内尚未写入任何数据），则将延长 SCL 线的低电平时间，直到写入了 I2C_TXDR 为止。在第九个 SCL 脉冲后进行延长。

图 377. 数据发送



硬件传输管理

I2C 内置了字节计数器，以便在下列各种模式下管理字节传输和结束通信：

- 主模式下生成 NACK、STOP 和 ReSTART
- 从接收器模式下控制 ACK 是否发出
- SMBus 模式下生成/校验 PEC

字节计数器通常在主模式下使用。在从模式下，字节计数器默认为禁止状态。可以通过软件来使能，方法是将 I2C_CR1 寄存器中的 SBC（从字节控制）位置 1。

待传输的字节数在 I2C_CR2 寄存器的 NBYTES[7:0] 位域中进行编程。如果待传输的字节数 (NBYTES) 大于 255，或者接收方希望控制是否对接收到的数据字节进行应答，则必须选择重载模式，方法是将 I2C_CR2 寄存器的 RELOAD 位置 1。在该模式下，完成 NBYTES 中所编程字节数的数据传输之后，TCR 标志将置 1，并且 TCIE 置 1 时将生成中断。只要 TCR 标志置 1，SCL 便会延长。向 NBYTES 写入一个非零值时，TCR 由软件清零。

在 NBYTES 中设置最后一次传输的字节数前，必须将 RELOAD 位清零。

当主模式下 RELOAD = 0 时，可在以下两种模式下使用计数器：

- **自动结束模式**（I2C_CR2 寄存器中的 AUTOEND = 1）。在该模式下，一旦完成 NBYTES[7:0] 位域中所编程字节数的数据传输，主器件便会自动发送停止位。
- **软件结束模式**（I2C_CR2 寄存器中的 AUTOEND = 0）。在该模式下，一旦完成 NBYTES[7:0] 位域中所编程字节数的数据传输，TC 标志将置 1，并且 TCIE 置 1 时将生成中断。只要 TC 标志置 1，SCL 信号便会延长，需要软件介入操作。当软件将 I2C_CR2 寄存器中的起始位或停止位置 1 时，TC 标志将被清零。当主器件要发送重复起始位时，必须使用该模式。

小心： 当 RELOAD 位置 1 时，AUTOEND 位将不起作用。

表 256. I2C 配置

功能	SBC 位	RELOAD 位	AUTOEND 位
主 Tx/Rx NBYTES + STOP	x	0	1
主 Tx/Rx + NBYTES + RESTART	x	0	0
从 Tx/Rx，接收的所有字节都要回复应答	0	x	x
具有 ACK 控制的从 Rx	1	1	x

34.4.8 I2C 从模式

I2C 从模式初始化

要在从模式下工作，用户必须至少使能一个从地址。可使用 I2C_OAR1 和 I2C_OAR2 这两个寄存器来编程自身从地址 OA1 和 OA2。

- OA1 既可配置为 7 位寻址模式（默认），也可通过将 I2C_OAR1 寄存器的 OA1MODE 位置 1 配置为 10 位寻址模式。

  通过将 I2C_OAR1 寄存器中的 OA1EN 位置 1 来使能 OA1。
- 如果需要额外的从地址，可配置第二个从地址 OA2。将 I2C_OAR2 寄存器的 OA2MSK[2:0] 位置 1 最多可屏蔽七个 OA2 LSB。因此，当 OA2MSK 配置为 1 到 6 时，将分别只有 OA2[7:2]、OA2[7:3]、OA2[7:4]、OA2[7:5]、OA2[7:6] 或 OA2[7] 与接收到的地址作比较。只要 OA2MSK 不等于 0，OA2 的地址比较器便会排除 I2C 保留地址（0000 XXX 和 1111 XXX），这些地址将不会得到应答。如果 OA2MSK = 7，接收到的所有 7 位地址（保留地址除外）均得到应答。OA2 始终为 7 位地址。

  如果这些保留地址在 I2C_OAR1 或 I2C_OAR2 寄存器中进行了编程并且 OA2MSK = 0，则它们可以在通过特定使能位使能后得到应答。

  通过将 I2C_OAR2 寄存器中的 OA2EN 位置 1 来使能 OA2。
- 通过将 I2C_CR1 寄存器中的 GCEN 位置 1 来使能广播呼叫地址。

当通过 I2C 的其中一个使能地址来寻址到该 I2C 设备时，ADDR 中断状态标志将置 1，并且 ADDRIE 位置 1 时将生成中断。

默认情况下，从器件使用其时钟延长功能（即必要时延长 SCL 信号的低电平时间）来为软件操作的执行提供时机。如果主器件不支持时钟延长，则必须对 I2C 进行如下配置：将 I2C_CR1 寄存器中 NOSTRETCH 位置 1。

接收到 ADDR 中断后，如果使能多个地址，则用户必须读取 I2C_ISR 寄存器中的 ADDCODE[6:0] 位，以确定是哪个地址匹配。还必须检查 DIR 标志，以获悉传输方向。

带时钟延长的从模式 (NOSTRETCH = 0)

在默认模式下，I2C 从器件会在以下情况下延长 SCL 时钟：

- ADDR 标志置 1 时：接收到的地址与其中一个使能的从地址匹配。通过软件将 ADDRCF 位置 1 以清零 ADDR 标志时，将释放该时钟延长。
- 发送时，前一次数据传输已完成但 I2C_TXDR 寄存器中未写入任何新数据，或者 ADDR 标志清零 (TXE = 1) 时未写入第一个数据字节。在 I2C_TXDR 寄存器中写入数据时，将释放该时钟延长。
- 接收时，尚未读取 I2C_RXDR 寄存器但新的数据接收已完成。读取 I2C_RXDR 时，将释放该时钟延长。
- 当从器件字节控制模式和重载模式（SBC = 1 且 RELOAD = 1）下 TCR = 1 时，这意味着最后一个数据字节已完成传输。通过向 NBYTES[7:0] 位域写入一个非零值以将 TCR 清零时，将释放该时钟延长。
- 检测到 SCL 下降沿后，I2C 会在 [(SDADEL + SCLDEL + 1) x (PRESC+ 1) + 1] x t_{I2CCLK} 期间内延长 SCL 低电平时间。

不带时钟延长的从模式 (NOSTRETCH = 1)

当 I2C_CR1 寄存器中的 NOSTRETCH = 1 时，I2C 从器件不会延长 SCL 信号。

- ADDR 标志置 1 时，不会延长 SCL 时钟。
- 发送时，必须在与发送数据对应的第一个 SCL 脉冲出现之前，向 I2C_TXDR 寄存器写入数据。否则，会发生下溢，I2C_ISR 寄存器中的 OVR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。当第一次数据发送开始而 STOPF 位仍置 1（尚未清零）时，OVR 标志也将置 1。因此，如果写入下一次传输要发送的第一个数据后才清零上一次传输的 STOPF 标志，则应提供 OVR 状态，甚至对于待发送的第一个数据也是如此。
- 接收时，必须在下一个数据字节的第九个 SCL 脉冲（ACK 脉冲）出现之前，从 I2C_RXDR 寄存器读取数据。否则，会发生上溢，I2C_ISR 寄存器中的 OVR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

从器件字节控制模式

要在从接收模式下实现字节 ACK 控制，必须通过将 I2C_CR1 寄存器中的 SBC 位置 1 来使能从器件字节控制模式。这样符合 SMBus 标准。

要在从接收模式下实现字节 ACK 控制，必须选择重载模式 (RELOAD = 1)。要控制每个字节，必须在 ADDR 中断子程序中将 NBYTES 初始化为 0x1，并在每接收一个字节后将 NBYTES 重载为 0x1。接收到字节后，TCR 位将置 1，从而延长 SCL 信号的第八个和第九个脉冲之间的低电平时间。用户可以从 I2C_RXDR 寄存器中读取数据，然后通过配置 I2C_CR2 寄存器中的 ACK 位来决定是否应答。通过将 NBYTES 编程为非零值来释放 SCL 延长：发送应答或不应答信号，然后可继续接收下一个字节。

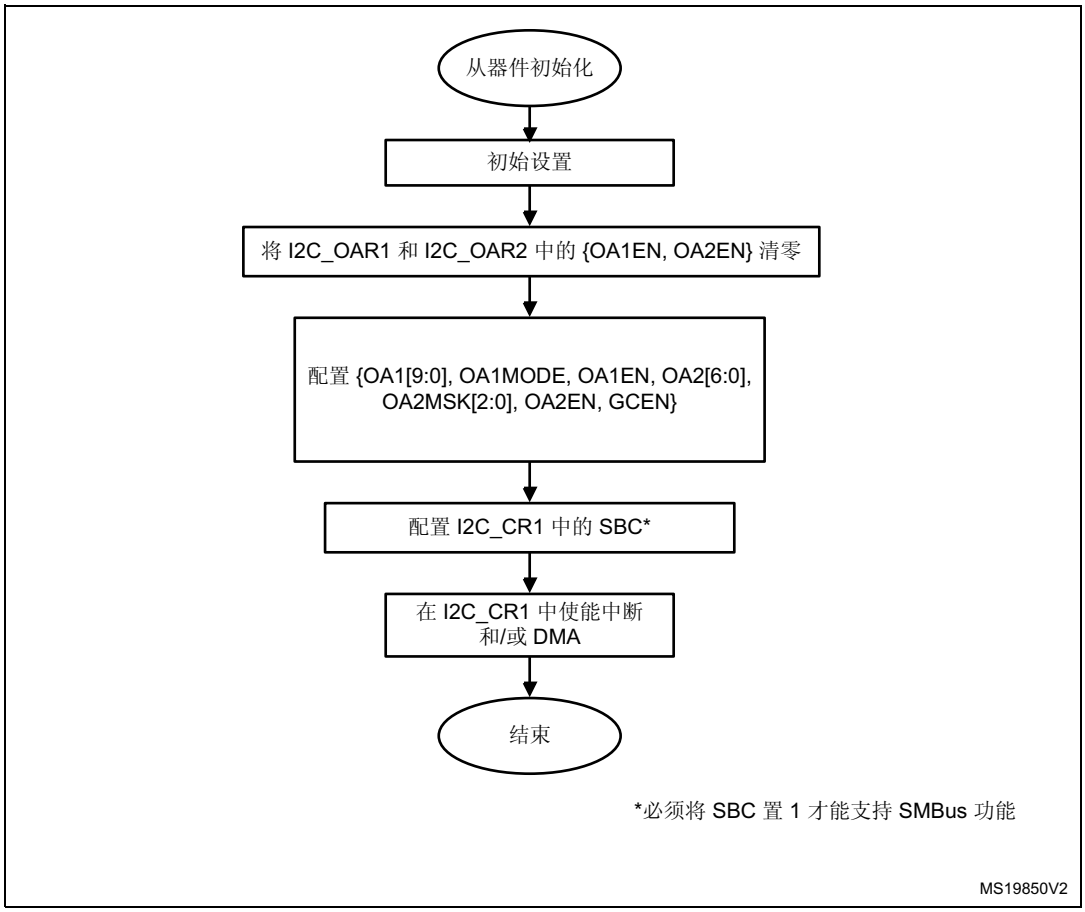
NBYTES 可加载大于 0x1 的值，在这种情况下，接收流在 NBYTES 个数据接收期间是连续的。

注意： *SBC 位只能在 I2C 被禁止时、从器件不被寻址时或 ADDR = 1 时配置。*

ADDR = 1 或 TCR = 1 时，可以更改 RELOAD 位的值。

小心： 从器件字节控制模式与 NOSTRETCH 模式不兼容。不允许在 NOSTRETCH = 1 时将 SBC 位置 1。

图 378. 从器件初始化流程



从发送器

当 I2C_TXDR 寄存器为空时，将生成发送中断状态 (TXIS)。如果 I2C_CR1 寄存器中的 TXIE 位置 1，将生成中断。

I2C_TXDR 寄存器中写入待发送的下一个数据字节时，TXIS 位将被清零。

接收到 NACK 时，I2C_ISR 寄存器中的 NACKF 位将置 1，如果 I2C_CR1 寄存器中的 NACKIE 位置 1，还将生成中断。从器件自动释放 SCL 和 SDA 线，以使主器件执行停止或重复起始位的发送。收到 NACK 时，TXIS 位不会置 1。

当接收到停止位且 I2C_CR1 寄存器中的 STOPIE 位置 1 时，I2C_ISR 寄存器中的 STOPF 标志将置 1 并且会生成中断。在大多数应用中，SBC 位通常编程为 0。在这种情况下，如果接收到从地址 (ADDR=1) 时 TXE = 0，用户可以选择发送 I2C_TXDR 寄存器的内容作为第一个数据字节，也可以选择通过将 TXE 位置 1 来刷新 I2C_TXDR 寄存器以编程新的数据字节。

在从器件字节控制模式 (SBC = 1) 下，必须在地址匹配中断子程序 (ADDR = 1) 中向 NBYTES 写入待发送数据的个数。在这种情况下，传输期间 TXIS 事件的数量对应于 NBYTES 中编程的值。

小心： 如果 NOSTRETCH = 1，当 ADDR 标志置 1 时不会延长 SCL 时钟，因此用户无法在 ADDR 子程序中刷新 I2C_TXDR 寄存器的内容，从而编程第一个数据字节。必须在 I2C_TXDR 寄存器中预编程待发送的第一个数据字节：

- 该数据可以是前一个传输消息的最后一个 TXIS 事件中写入的数据。
- 如果该数据字节不是待发送的数据字节，可通过将 TXE 位置 1 来刷新 I2C_TXDR 寄存器，从而编程新的数据字节。必须仅在执行完这些操作后再清零 STOPF 位，以确保在地址应答之后第一次数据传输之前执行这些操作。

  如果第一次数据传输开始时 STOPF 仍置 1，则将生成下溢错误（OVR 标志置 1）。

  如果需要 TXIS 事件（发送中断或发送 DMA 请求），用户必须将 TXE 位和 TXIS 位均置 1，以便生成 TXIS 事件。

图 379. I2C 从发送器的传输序列流程 (NOSTRETCH = 0)

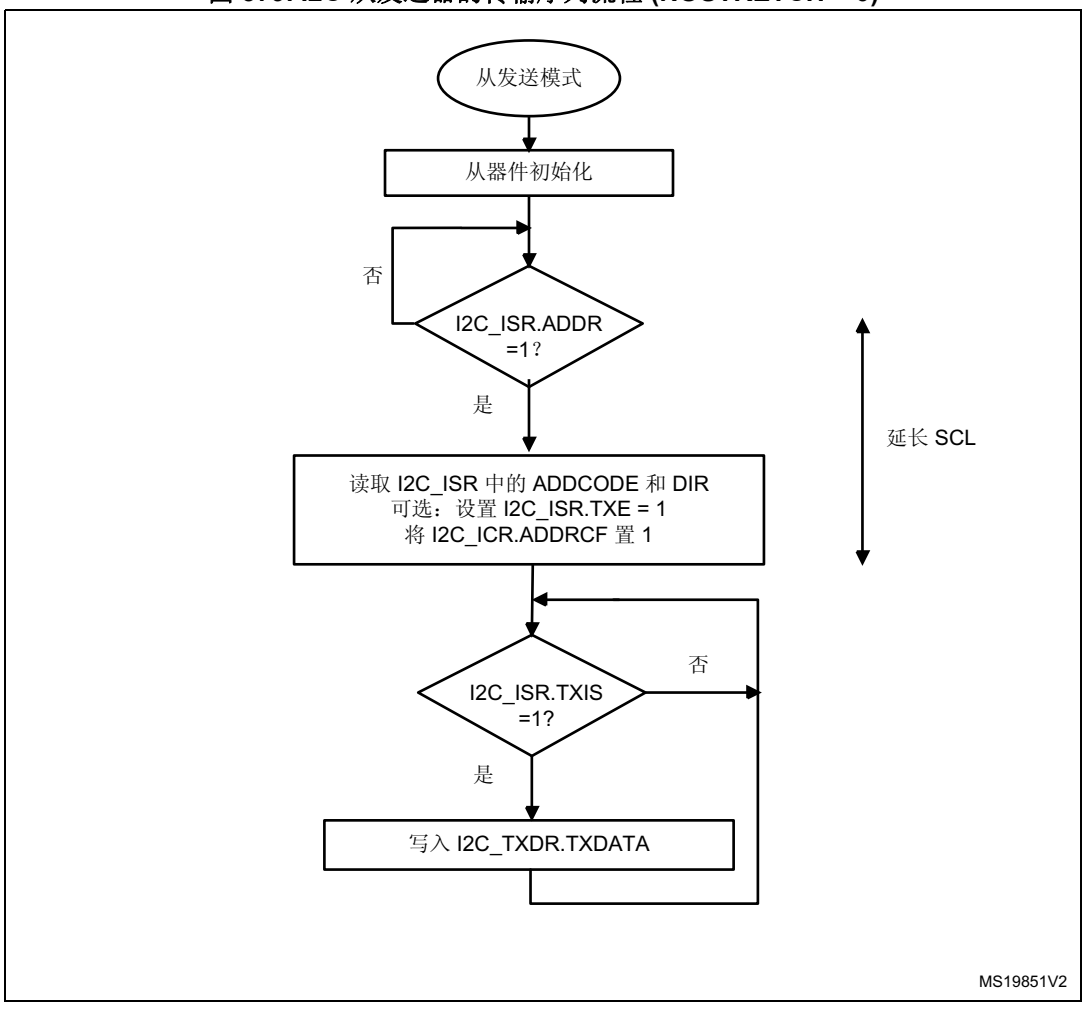


图 380. I2C 从发送器的传输序列流程 (NOSTRETCH = 1)

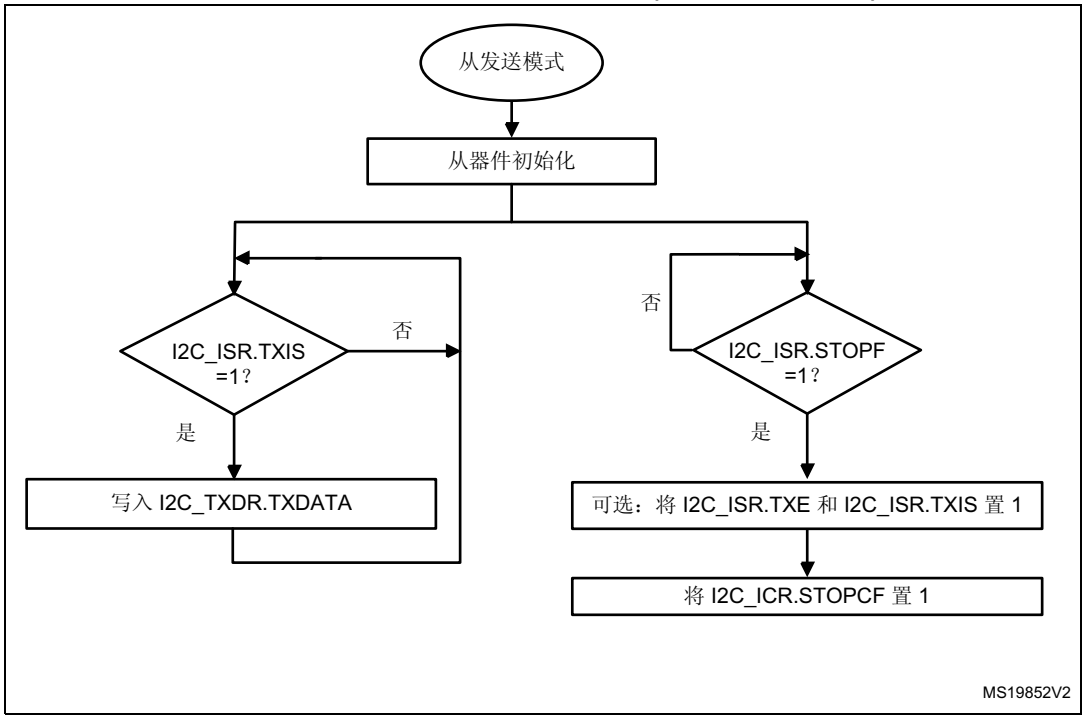
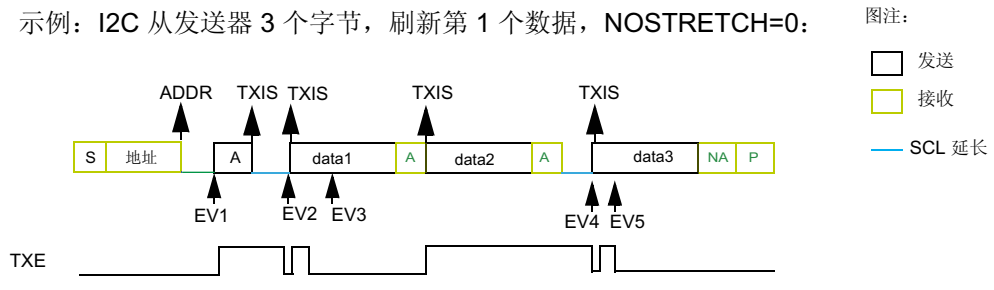


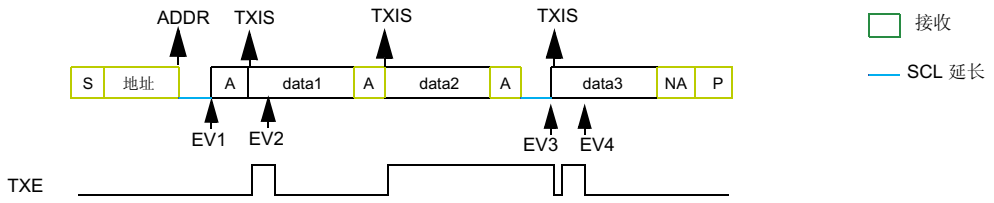
图 381. I2C 从发送器的传输总线图（仅限强制性事件）

示例：I2C 从发送器 3 个字节，刷新第 1 个数据，NOSTRETCH=0：



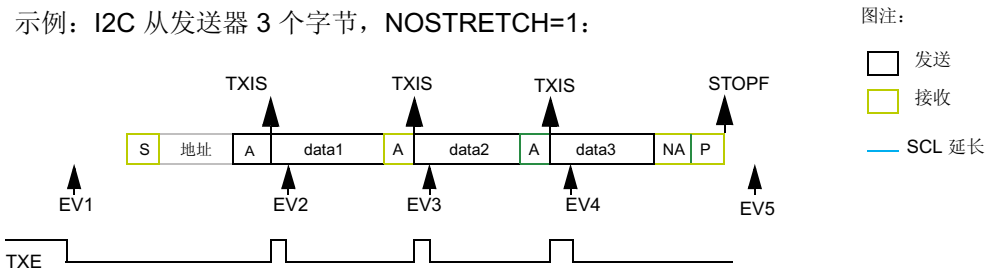
EV1：ADDR ISR：检查 ADDCODE 和 DIR，将 TXE 置 1，将 ADDRCF 置 1
EV2：TXIS ISR：写入 data1
EV3：TXIS ISR：写入 data2
EV4：TXIS ISR：写入 data3
EV5：TXIS ISR：写入 data4（不发送）

示例：I2C 从发送器 3 个字节，不刷新第 1 个数据，NOSTRETCH=0：



EV1：ADDR ISR：检查 ADDCODE 和 DIR，将 ADDRCF 置 1
EV2：TXIS ISR：写入 data2
EV3：TXIS ISR：写入 data3
EV4：TXIS ISR：写入 data4（不发送）

示例：I2C 从发送器 3 个字节，NOSTRETCH=1：



EV1：写入 data1
EV2：TXIS ISR：写入 data2
EV3：TXIS ISR：写入 data3
EV4：TXIS ISR：写入 data4（不发送）
EV5：STOPF ISR：（可选：将 TXE 和 TXIS 置 1），将 STOPCF 置 1

MS19853V2

从接收器

当 I2C_RXDR 满时，I2C_ISR 中的 RXNE 将置 1，如果 I2C_CR1 中的 RXIE 置 1，还将生成中断。读取 I2C_RXDR 时，将清零 RXNE。

接收到停止条件且 I2C_CR1 寄存器中的 STOPIE 置 1 时，I2C_ISR 中的 STOPF 将置 1 并且会生成中断。

图 382. 从接收器的传输序列流程 (NOSTRETCH = 0)

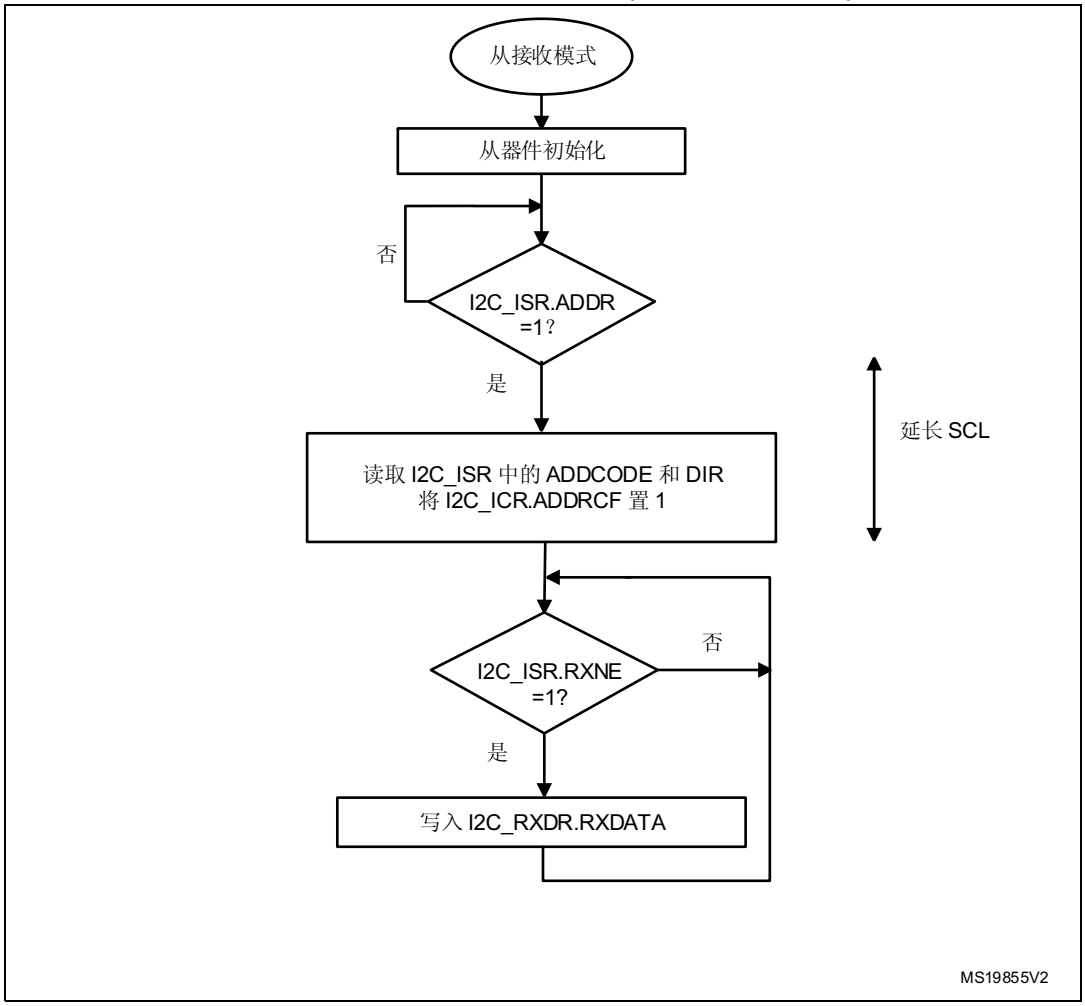


图 383. 从接收器的传输序列流程 (NOSTRETCH = 1)

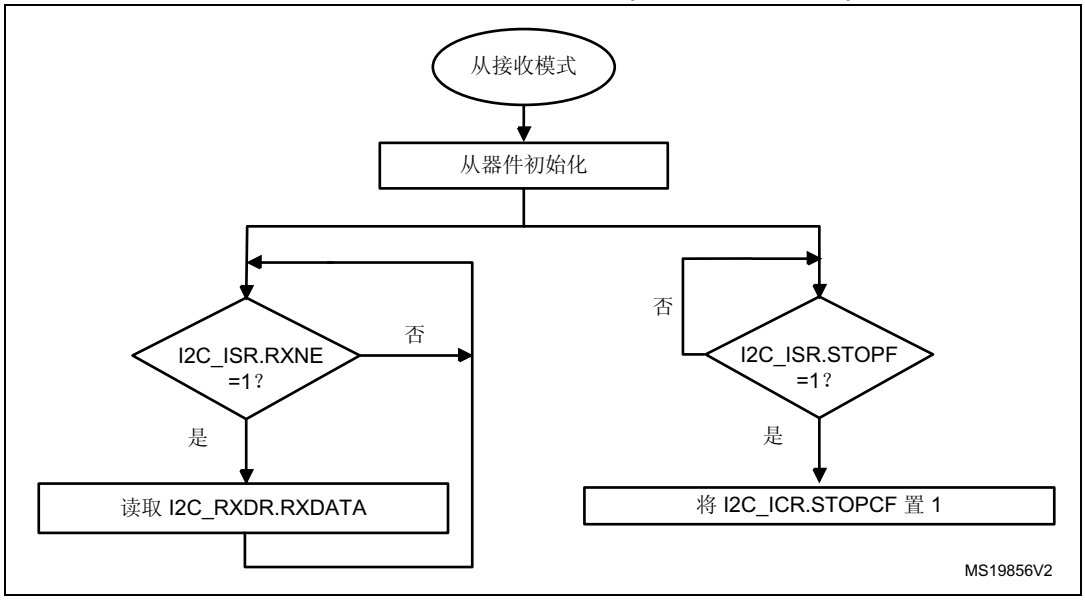
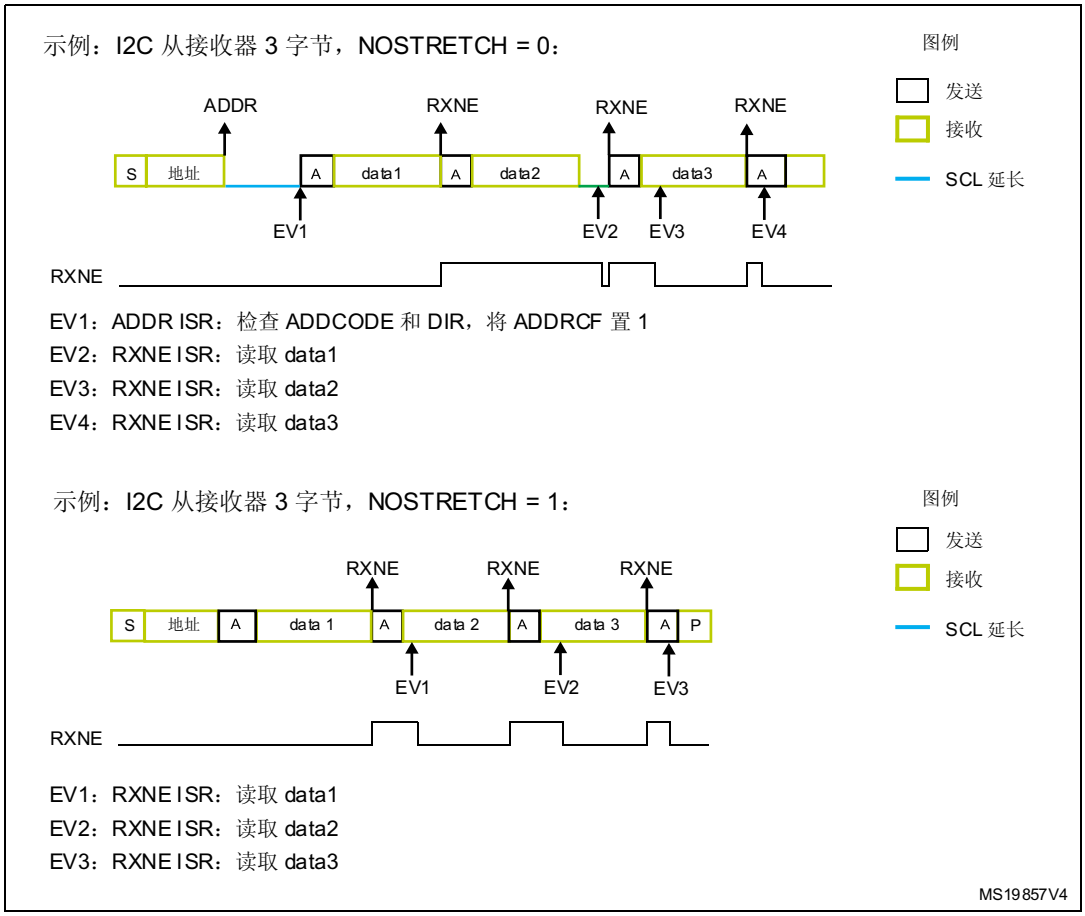


图 384. I2C 从接收器的传输总线图（仅限强制性事件）



34.4.9 I2C 主模式

I2C 主模式初始化

使能外设前，必须通过设置 I2C_TIMINGR 寄存器中的 SCLH 和 SCLL 位来配置 I2C 主时钟。

STM32CubeMX 工具计算 I2C_TIMINGR 内容并在 I2C 配置窗口中进行显示。

为了支持多主环境和从时钟延长，I2C 实现了时钟同步机制。

为了实现时钟同步，需执行以下操作：

- 使用 SCLL 计数器从 SCL 低电平内部检测开始对时钟的低电平进行计数。
- 使用 SCLH 计数器从 SCL 高电平内部检测开始对时钟的高电平进行计数。

I2C 经过 t_{SYNC1} 延时后检测其自身的 SCL 低电平，该延时取决于 SCL 下降沿、SCL 输入噪声滤波器（模拟 + 数字）以及 SCL 与 I2CxCLK 时钟的同步。一旦 SCLL 计数器达到 I2C_TIMINGR 寄存器的 SCLL[7:0] 位中编程的值，I2C 便会将 SCL 释放为高电平。

I2C 经过 t_{SYNC2} 延时后检测其自身的 SCL 高电平，该延时取决于 SCL 上升沿、SCL 输入噪声滤波器（模拟 + 数字）以及 SCL 与 I2CxCLK 时钟的同步。一旦 SCLH 计数器达到 I2C_TIMINGR 寄存器的 SCLH[7:0] 位中编程的值，I2C 便会使 SCL 变为低电平。

因此，主时钟周期为：

$$t_{SCL} = t_{SYNC1} + t_{SYNC2} + \{[(SCLH+ 1) + (SCLL+ 1)] \times (PRESC+ 1) \times t_{I2CCLK}\}$$

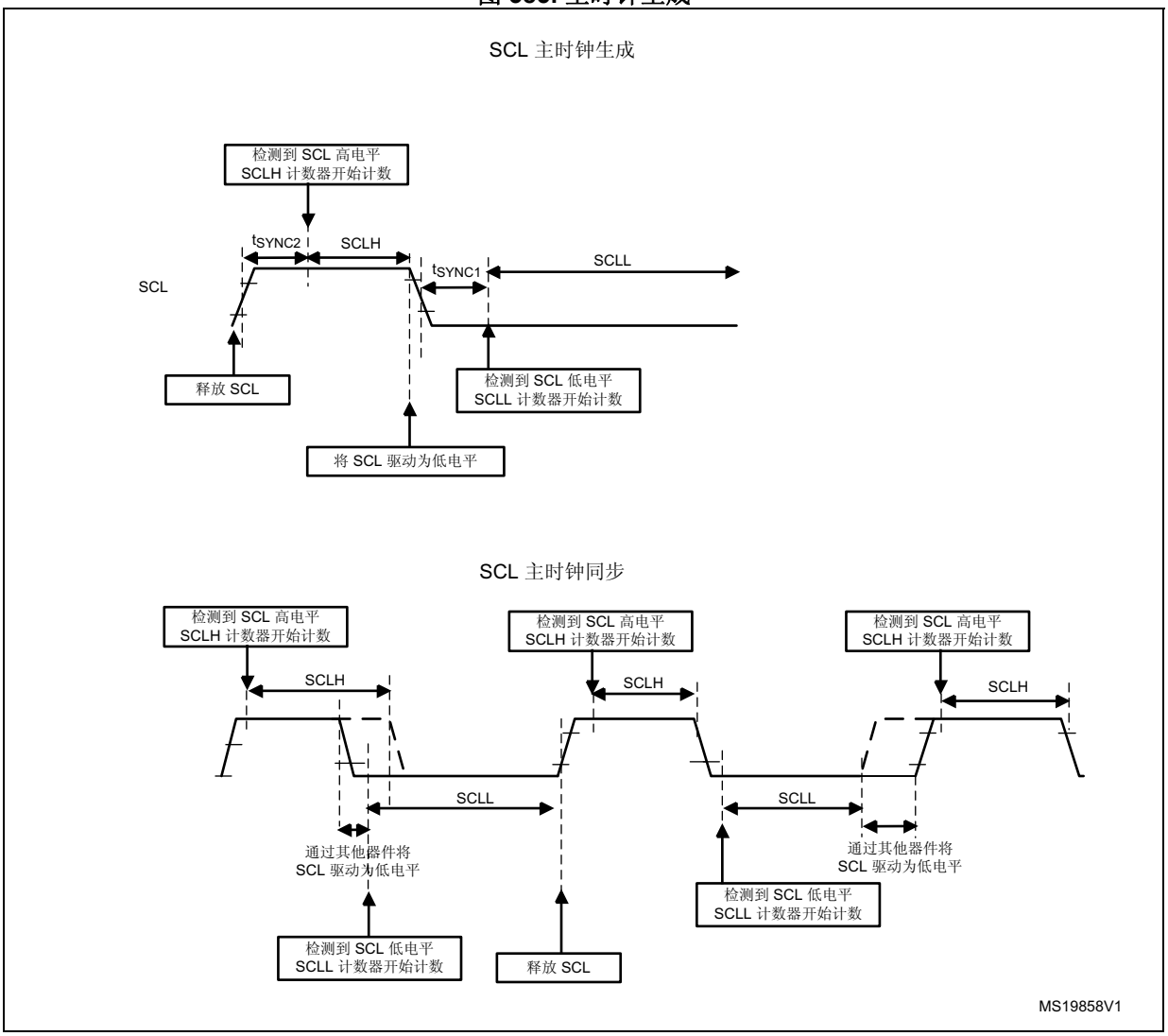
t_{SYNC1} 的持续时间取决于：

- SCL 下降斜率
- 模拟滤波器（使能时）引入的输入延时
- 数字滤波器（使能时）引入的输入延时：DNF x t_{I2CCLK}
- SCL 与 i2c_ker_ck 时钟建立同步而产生的延时（两到三个 i2c_ker_ck 周期）

t_{SYNC2} 的持续时间取决于：

- SCL 上升斜率
- 模拟滤波器（使能时）引入的输入延时
- 数字滤波器（使能时）引入的输入延时：DNF x t_{I2CCLK}
- SCL 与 i2c_ker_ck 时钟建立同步而产生的延时（两到三个 i2c_ker_ck 周期）

图 385. 主时钟生成


SCL 主时钟生成
检测到 SCL 高电平
SCLH 计数器开始计数
tSYNC2
SCLH
tSYNC1
SCLL
SCL
释放 SCL
检测到 SCL 低电平
SCLL 计数器开始计数
将 SCL 驱动为低电平
SCL 主时钟同步
检测到 SCL 高电平
SCLH 计数器开始计数
SCLH
SCLL
通过其他器件将
SCL 驱动为低电平
检测到 SCL 低电平
SCLL 计数器开始计数
释放 SCL
MS19858V1

小心： 为了符合 I²C 或 SMBus 规范，主时钟必须遵循下表中给出的时序。

表 257. I²C-SMBus 规范时钟时序

符号	参数	标准模式 (Sm)		快速模式 (Fm)		超快速模式 (Fm+)		SMBus		单位
		最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值	
f_{SCL}	SCL 时钟频率	-	100	-	400	-	1000	-	100	kHz
$t_{HD:STA}$	（重复）起始条件的保持时间	4.0	-	0.6	-	0.26	-	4.0	-	μs
$t_{SU:STA}$	重复起始条件的建立时间	4.7	-	0.6	-	0.26	-	4.7	-	
$t_{SU:STO}$	停止条件的建立时间	4.0	-	0.6	-	0.26	-	4.0	-	
t_{BUF}	停止条件和起始条件之间的总线空闲时间	4.7	-	1.3	-	0.5	-	4.7	-	
t_{LOW}	SCL 时钟的低电平周期	4.7	-	1.3	-	0.5	-	4.7	-	
t_{HIGH}	SCL 时钟的高电平周期	4.0	-	0.6	-	0.26	-	4.0	50	
t_r	SDA 和 SCL 信号的上升时间	-	1000	-	300	-	120	-	1000	ns
t_f	SDA 和 SCL 信号的下降时间	-	300	-	300	-	120	-	300	

注意： *SCLL 还用于生成 t_{BUF} 和 $t_{SU:STA}$ 时序，SCLH 还用于生成 $t_{HD:STA}$ 和 $t_{SU:STO}$ 时序。*

有关 I2C_TIMINGR 设置与 i2c_ker_ck 频率的示例，请参见*第 34.4.10 节*。

主模式通信初始化（地址阶段）

要发起通信，用户必须在 I2C_CR2 寄存器中为寻址的从器件编程以下参数：

- 寻址模式（7 位或 10 位）：ADD10
- 待发送的从地址：SADD[9:0]
- 传输方向：RD_WRN
- 读取 10 位地址时：HEAD10R 位。必须对 HEAD10R 进行相应配置，以指示传输方向变化时必须发送完整的地址序列，还是只发送地址头。
- 待传输的字节数：NBYTES[7:0]。如果字节数等于或大于 255，则初始化时必须将 NBYTES[7:0] 填充为 0xFF。

然后，用户必须将 I2C_CR2 寄存器中的 START 位置 1。START 位置 1 时，不允许更改上述所有位。

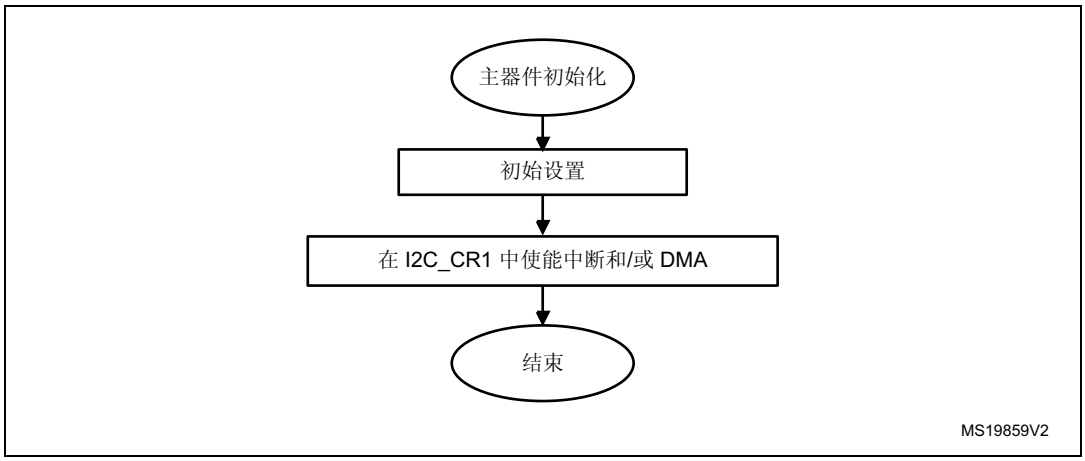
之后，当主器件检测到总线空闲 (BUSY = 0) 时，它会在经过 t_{BUF} 的延时后自动发送起始位，随后发出从器件地址。

仲裁丢失时，主器件将自动切换回从模式，如果作为从器件被寻址，还可对其自身地址进行应答。

注意： *无论接收到的应答值为何，只要已在总线上发送从地址，START 位便会由硬件复位。如果仲裁丢失，START 位也会由硬件复位。*
在 10 位寻址模式下，如果从器件不对从地址的前七个位进行应答，则主器件将自动重新启动从地址发送，直至接收到 ACK。在这种情况下，如果从从器件接收到 NACK，则必须将 ADDRCF 置 1，以停止发送从地址。
如果当 START 位置 1 时，I2C 作为从器件 (ADDR = 1) 被寻址，则 I2C 将切换为从模式，START 位将清零。

注意：　该步骤同样适用于重复起始位。在这种情况下，BUSY = 1。

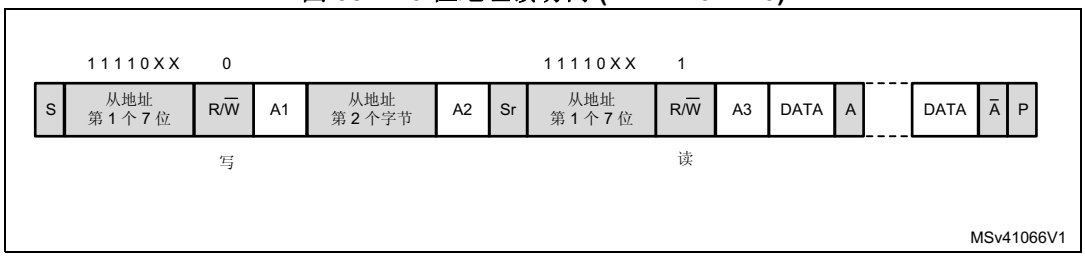
图 386. 主器件初始化流程



主接收器寻址 10 位地址从器件的初始化过程

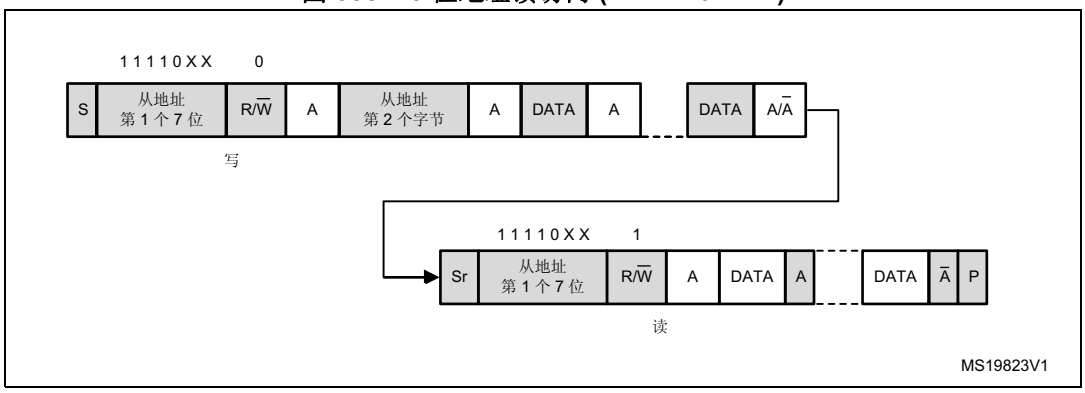
- 如果从地址采用 10 位格式，用户可选择将 I2C_CR2 寄存器中的 HEAD10R 位清零来发送完整的读序列。在这种情况下，主器件会在 START 位置 1 后自动发送以下完整序列：（重复）起始位 + 带写方向的从器件 10 位地址头字节 + 从器件地址第二个字节 + 重复起始位 + 带读方向的从器件 10 位地址头字节

图 387. 10 位地址读访问 (HEAD10R = 0)



- 如果主器件对 10 位地址从器件进行寻址、向该从器件发送数据、然后再从该从器件读取数据，则必须首先完成主器件发送过程。然后，重复起始位置 1，10 位从地址配置为 HEAD10R = 1。在这种情况下，主器件发送以下序列：重复起始位 + 带读方向的从器件 10 位地址头字节。

图 388. 10 位地址读访问 (HEAD10R = 1)



主发送器

写传输时，在发送完每个字节（即第九个 SCL 脉冲（接收到 ACK 时））后，TXIS 标志将置 1。

如果 I2C_CR1 寄存器中的 TXIE 位置 1，TXIS 事件将生成中断。当 I2C_TXDR 寄存器中写入待发送的下一个数据字节时，该标志将被清零。

传输期间的 TXIS 事件的数量对应于 NBYTES[7:0] 中编程的值。如果待发送的数据字节总数大于 255，则必须通过将 I2C_CR2 寄存器中的 RELOAD 位置 1 来选择重载模式。在这种情况下，当 NBYTES 数据传输完成时，TCR 标志将置 1，并且 SCL 线的低电平将被延长，直到 NBYTES[7:0] 被写入非零值。

收到 NACK 时，TXIS 标志不会置 1。

- 当 RELOAD = 0 且 NBYTES 数据传输完成时：
  - 在自动结束模式 (AUTOEND = 1) 下，将自动发送停止位。
  - 在软件结束模式 (AUTOEND = 0) 下，TC 标志将置 1 且 SCL 线的低电平将被延长，以便执行以下软件操作：

    可通过将 I2C_CR2 寄存器中的 START 位置 1 并配置适当的从地址和待传输字节数来请求发送重复起始位。将 START 位置 1 会将 TC 标志清零，并在总线上发送起始位。

    可通过将 I2C_CR2 寄存器中的 STOP 位置 1 来请求停止位。将 STOP 位置 1 会将 TC 标志清零，并在总线上发送停止位。
- 如果接收到 NACK：TXIS 标志不会置 1，并且接收到 NACK 后会自动发送停止位。I2C_ISR 寄存器中的 NACKF 标志置 1，如果 NACKIE 位置 1，还将生成中断。

图 389. I2C 主发送器的传输序列流程（N ≤ 255 字节）

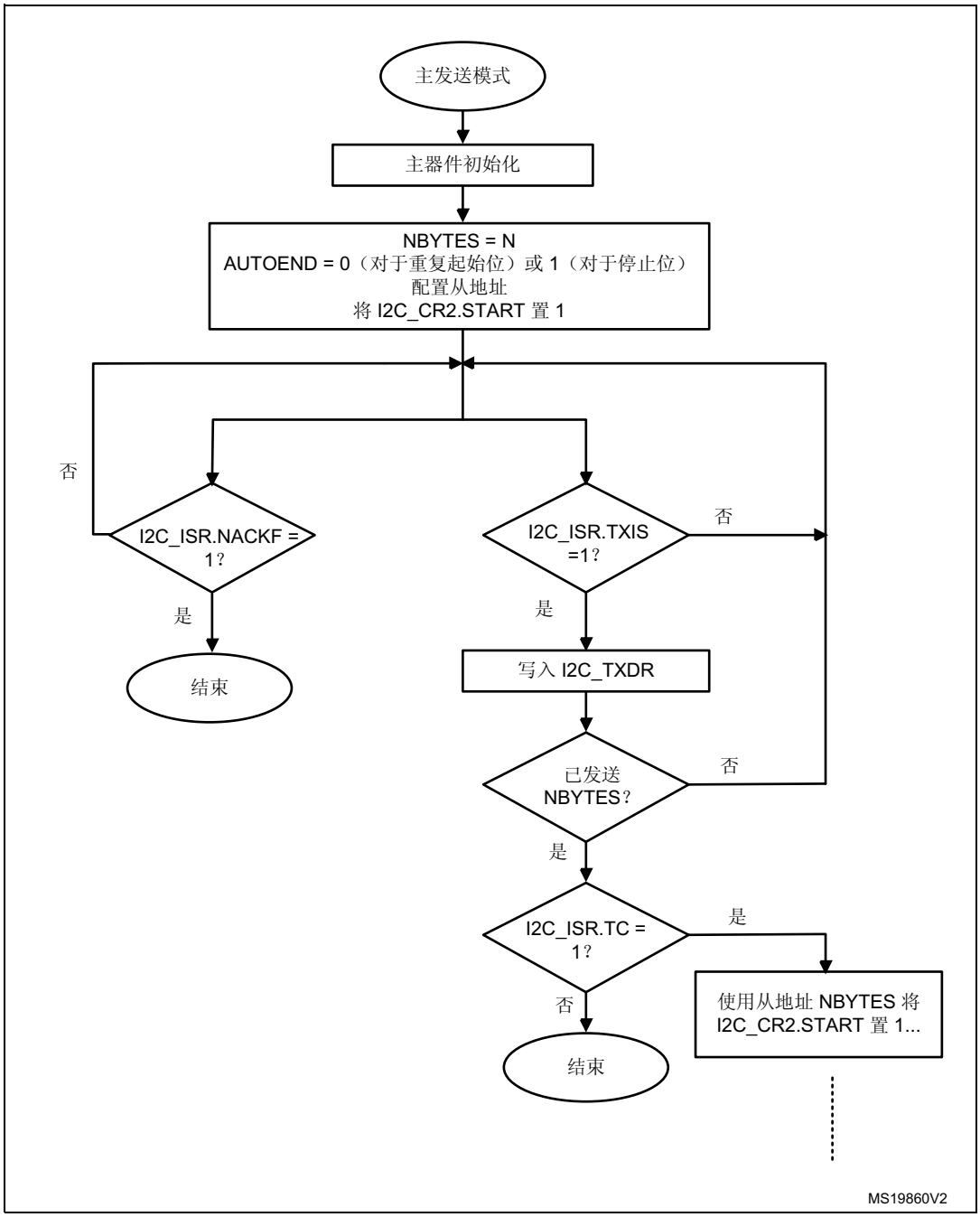


图 390. I2C 主发送器的传输序列流程（N > 255 字节）

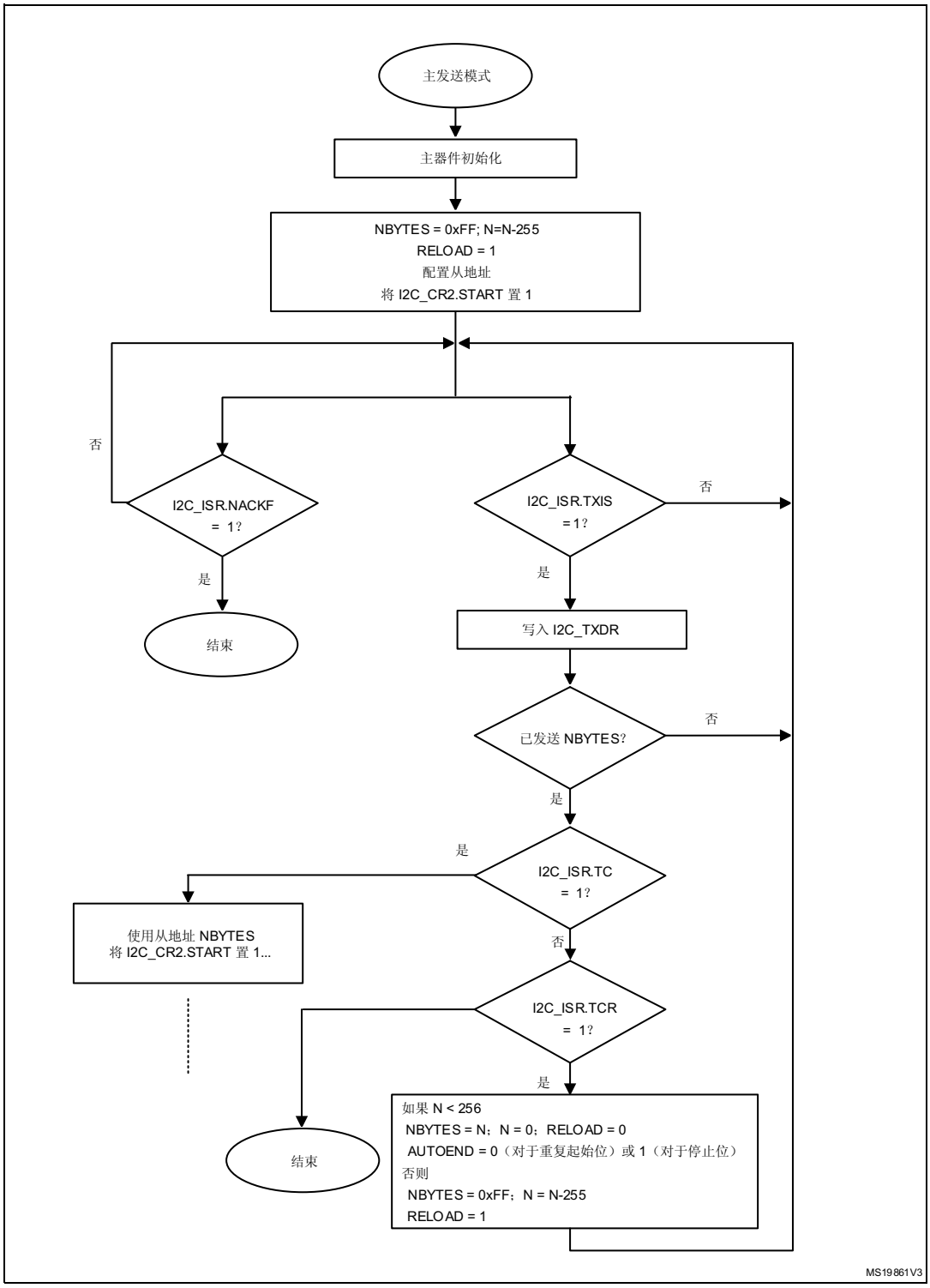
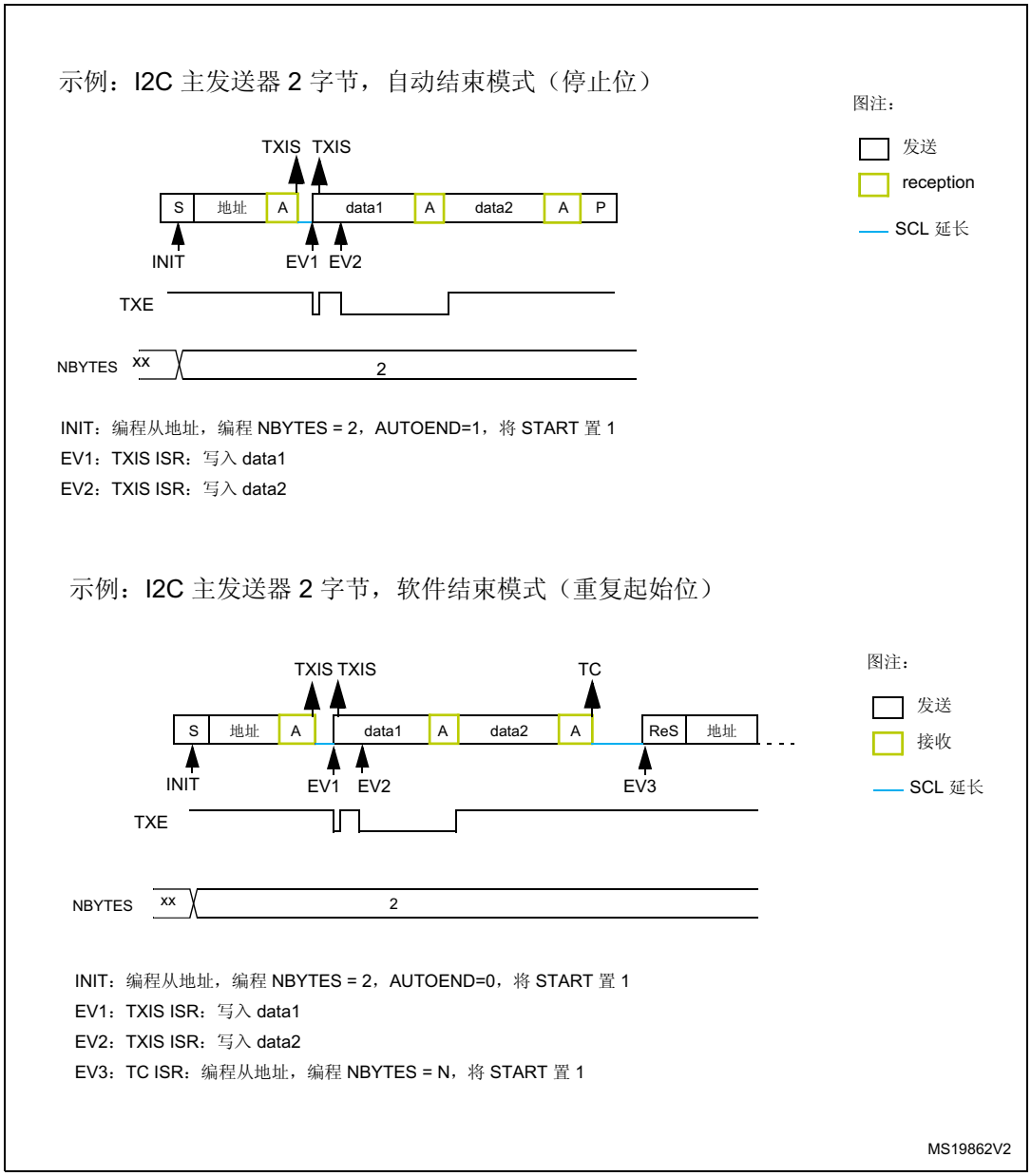


图 391. I2C 主发送器的传输总线图（仅限强制性事件）



主接收器

读传输时，在接收到每个字节（即第八个 SCL 脉冲）后，RXNE 标志将置 1。如果 I2C_CR1 寄存器中的 RXIE 位置 1，RXNE 事件将生成中断。读取 I2C_RXDR 时，将清零该标志。

如果待接收的数据字节总数大于 255，则必须通过将 I2C_CR2 寄存器中的 RELOAD 位置 1 来选择重载模式。在这种情况下，当 NBYTES[7:0] 数据传输完成时，TCR 标志将置 1，并且 SCL 线的低电平将被延长，直到 NBYTES[7:0] 被写入非零值。

- 当 RELOAD = 0 且 NBYTES[7:0] 数据传输完成时：
  - 在自动结束模式 (AUTOEND = 1) 下，接收到最后一个字节后，将自动发送 NACK 和停止位。
  - 在软件结束模式 (AUTOEND = 0) 下，接收到最后一个字节后，将自动发送 NACK，TC 标志将置 1 且 SCL 线的低电平将被延长，以便执行以下软件操作：

    可通过将 I2C_CR2 寄存器中的 START 位置 1 并配置适当的从地址和待传输字节数来请求发送重复起始位。将 START 位置 1 会将 TC 标志清零，并在总线上发送起始位，后跟从地址。

    可通过将 I2C_CR2 寄存器中的 STOP 位置 1 来请求停止位。将 STOP 位置 1 会将 TC 标志清零，并在总线上发送停止位。

图 392. I2C 主接收器的传输序列流程（N ≤ 255 字节）

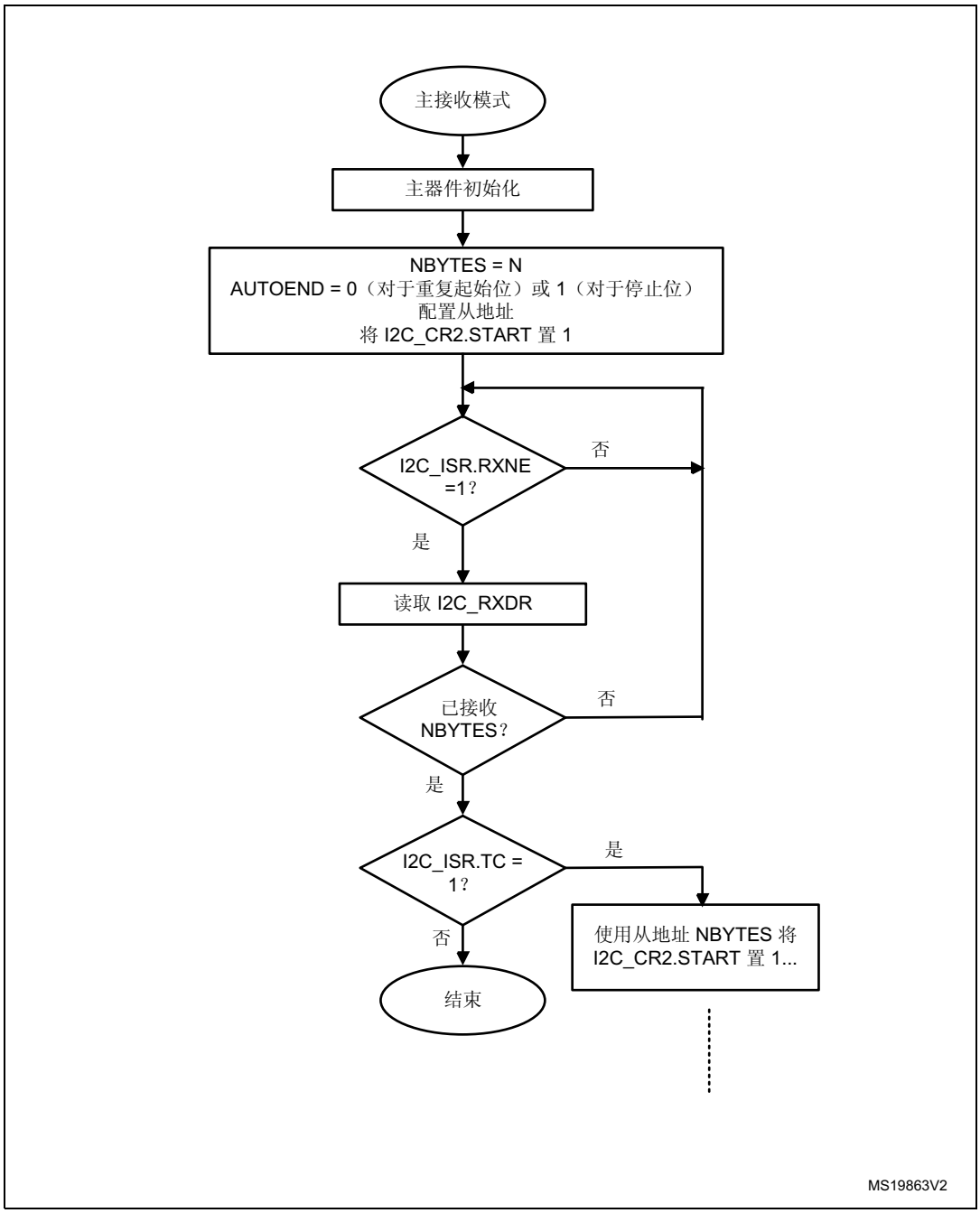


图 393. I2C 主接收器的传输序列流程（N > 255 字节）

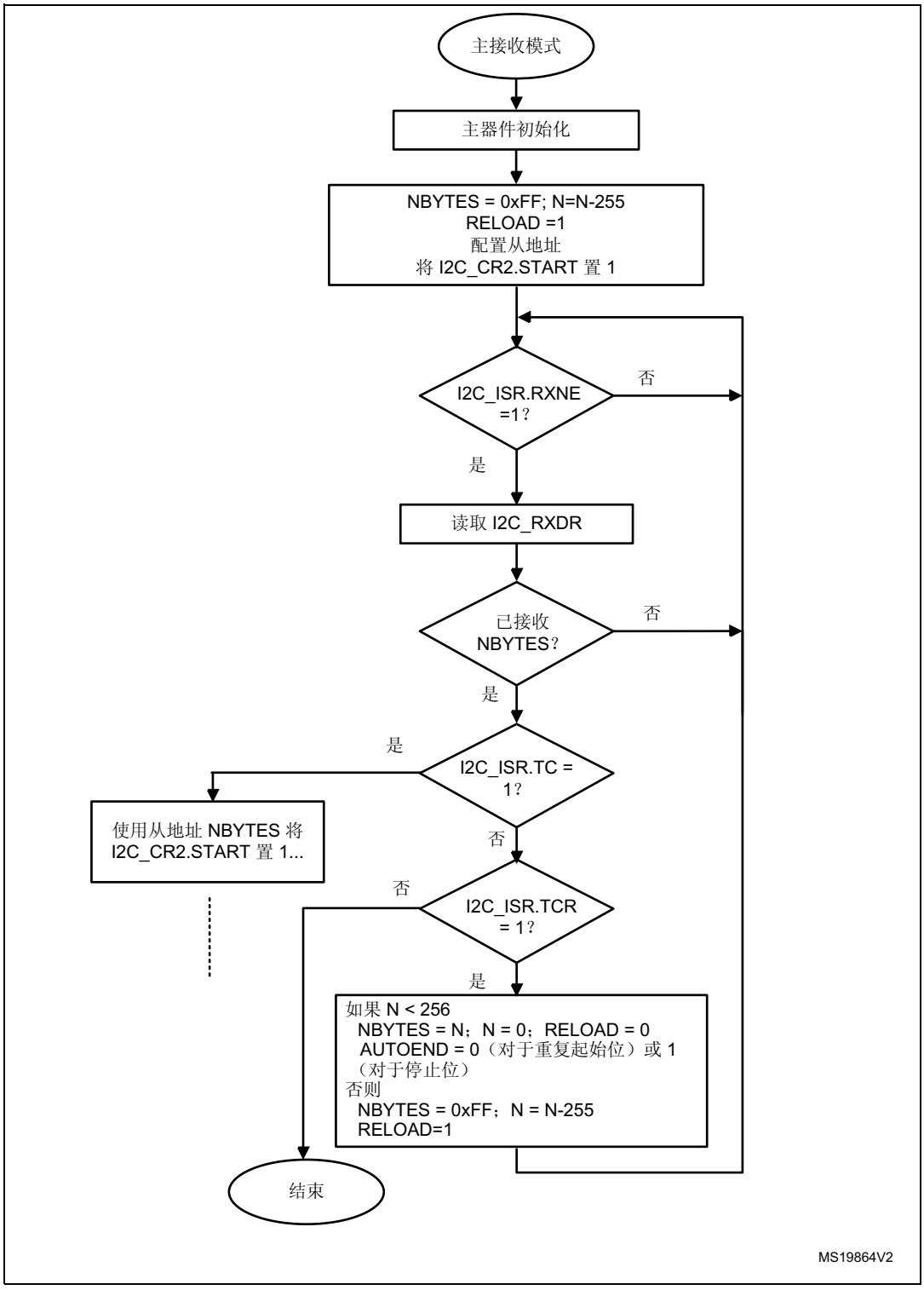
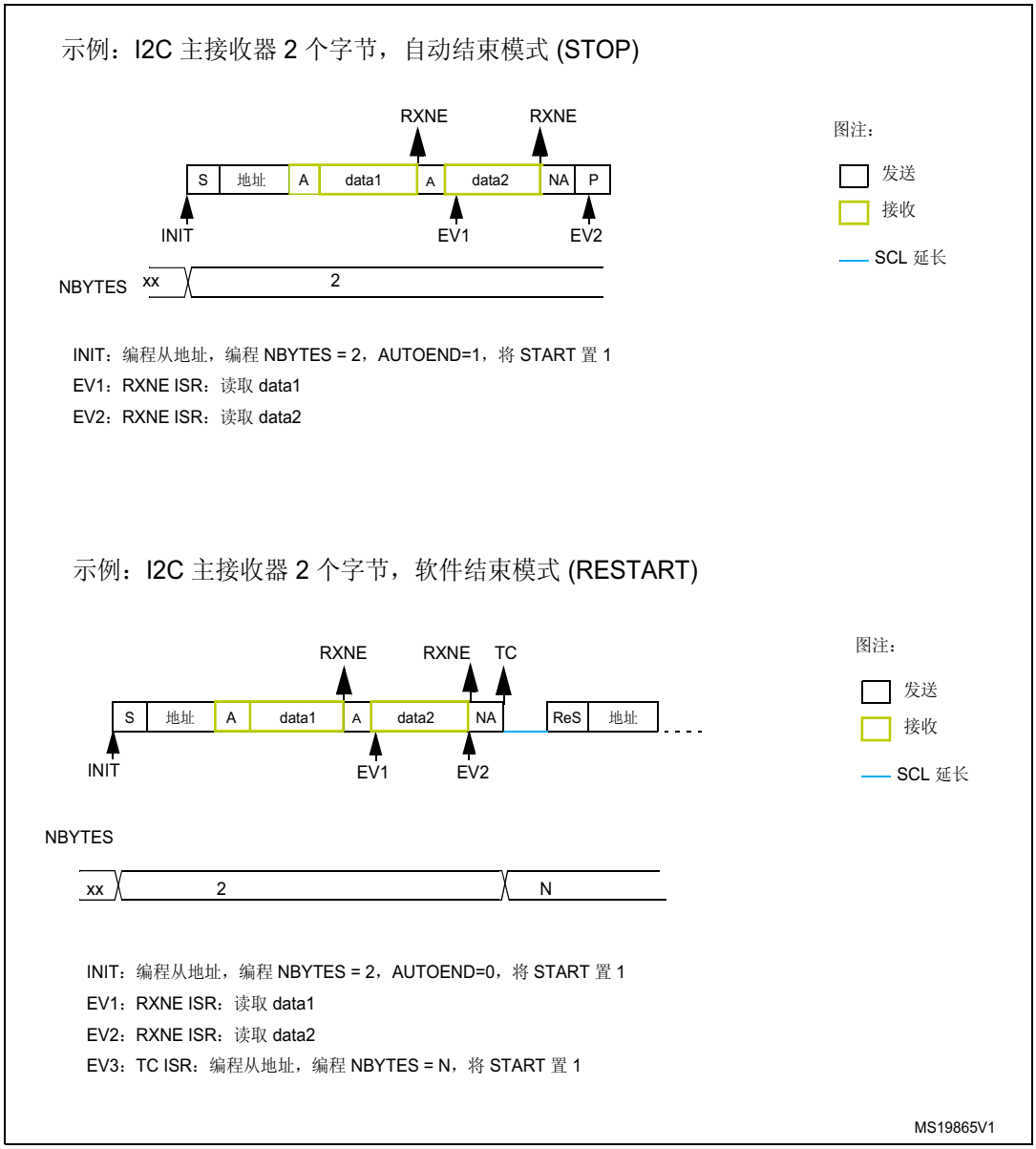


图 394. I2C 主接收器的传输总线图（仅限强制性事件）



34.4.10 I2C_TIMINGR 寄存器配置示例

下文各表提供了相应示例，以介绍如何编程 I2C_TIMINGR 才能获得符合 I^2C 规范的时序。要获取更准确的配置值，应使用 STM32CubeMX 工具（I2C 配置窗口）。

表 258. f_{I2CCLK} = 8 MHz 时的时序设置示例

参数	标准模式 (Sm)		快速模式 (Fm)	超快速模式 (Fm+)
	10 kHz	100 kHz	400 kHz	500 kHz
PRESC	0x1	0x1	0x0	0x0
SCLL	0xC7	0x13	0x9	0x6
t_{SCLL}	200 x 250 ns = 50 µs	20 x 250 ns = 5.0 µs	10 x 125 ns = 1250 ns	7 x 125 ns = 875 ns
SCLH	0xC3	0xF	0x3	0x3
t_{SCLH}	196 x 250 ns = 49 µs	16 x 250 ns = 4.0 µs	4 x 125 ns = 500 ns	4 x 125 ns = 500 ns
t_{SCL}(1)	约 100 µs(2)	约 10 µs(2)	约 2.5 µs(3)	约 2.0 µs(4)
SDADEL	0x2	0x2	0x1	0x0
t_{SDADEL}	2 x 250 ns = 500 ns	2 x 250 ns = 500 ns	1 x 125 ns = 125 ns	0 ns
SCLDEL	0x4	0x4	0x3	0x1
t_{SCLDEL}	5 x 250 ns = 1250 ns	5 x 250 ns = 1250 ns	4 x 125 ns = 500 ns	2 x 125 ns = 250 ns

1. 由于 SCL 内部检测存在延时，t_{SCL} 大于 t_{SCLL} + t_{SCLH}。为 t_{SCL} 提供的值仅用于举例说明。
2. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 500 ns。t_{SYNC1} + t_{SYNC2} = 1000 ns 时的示例。
3. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 500 ns。t_{SYNC1} + t_{SYNC2} = 750 ns 时的示例。
4. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 500 ns。t_{SYNC1} + t_{SYNC2} = 655 ns 时的示例。

表 259. f_{I2CCLK} = 16 MHz 时的时序设置示例

参数	标准模式 (Sm)		快速模式 (Fm)	超快速模式 (Fm+)
	10 kHz	100 kHz	400 kHz	1000 kHz
PRESC	0x3	0x3	0x1	0x0
SCLL	0xC7	0x13	0x9	0x4
t_{SCLL}	200 x 250 ns = 50 µs	20 x 250 ns = 5.0 µs	10 x 125 ns = 1250 ns	5 x 62.5 ns = 312.5 ns
SCLH	0xC3	0xF	0x3	0x2
t_{SCLH}	196 x 250 ns = 49 µs	16 x 250 ns = 4.0 µs	4 x 125 ns = 500 ns	3 x 62.5 ns = 187.5 ns
t_{SCL}(1)	约 100 µs(2)	约 10 µs(2)	约 2.5 µs(3)	约 1.0 µs(4)
SDADEL	0x2	0x2	0x2	0x0
t_{SDADEL}	2 x 250 ns = 500 ns	2 x 250 ns = 500 ns	2 x 125 ns = 250 ns	0 ns
SCLDEL	0x4	0x4	0x3	0x2
t_{SCLDEL}	5 x 250 ns = 1250 ns	5 x 250 ns = 1250 ns	4 x 125 ns = 500 ns	3 x 62.5 ns = 187.5 ns

1. 由于 SCL 内部检测存在延时，t_{SCL} 大于 t_{SCLL} + t_{SCLH}。为 t_{SCL} 提供的值仅用于举例说明。
2. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 250 ns。t_{SYNC1} + t_{SYNC2} = 1000 ns 时的示例。
3. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 250 ns。t_{SYNC1} + t_{SYNC2} = 750 ns 时的示例。
4. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 250 ns。t_{SYNC1} + t_{SYNC2} = 500 ns 时的示例。

34.4.11 SMBus 特性

仅当支持 SMBus 功能（请参见*第 34.3 节*）时，才涉及本节内容。

简介

系统管理总线 (SMBus) 是一个双线制接口，各器件可通过它在彼此之间或者与系统的其余部分进行通信。它以 I^2C 的工作原理为基础。SMBus 可针对系统和电源管理相关的任务提供控制总线。

该外设与 SMBus 规范 (http://smbus.org) 兼容。

系统管理总线规范涉及三类器件。

- 从器件，用于接收或响应命令。
- 主器件，用于发出命令、生成时钟和中止传输。
- 主机，专用的主器件，可提供连接系统 CPU 的主接口。主机必须具有主 - 从设备功能，并且必须支持 SMBus 主机通知协议。系统中只允许存在一个主机。

该外设可配置为主器件或从器件，也可配置为主机。

总线协议

任何给定器件都有十一种可用命令协议。器件既可以在这些协议中任选其一，也可以使用全部协议进行通信。这些协议分别为快速命令、发送字节、接收字节、写入字节、写入字、读取字节、读取字、过程调用、块读取、块写入以及块写入-块读取过程调用。这些协议必须通过用户软件实现。

有关这些协议的详细信息，请参见 SMBus 规范 (http://smbus.org)。

地址解析协议 (ARP)

通过为各个从器件动态分配一个新的唯一地址可解决 SMBus 从地址冲突的问题。为了提供一种机制来针对地址分配隔离各个器件，各器件必须具有唯一的器件标识符 (UDID)。该 128 位数字由软件实现。

该外设支持地址解析协议 (ARP)。通过将 I2C_CR1 寄存器中的 SMBDEN 位置 1 来使能 SMBus 器件默认地址 (0b1100 001)。ARP 命令必须通过用户软件实现。

此外，还将在从模式下执行仲裁以支持 ARP。

有关 SMBus 地址解析协议的详细信息，请参见 SMBus 规范 (http://smbus.org)。

接收的命令和数据应答控制

SMBus 接收器必须能够对接收到的每个命令或数据进行 NACK 应答。要在从模式下实现 ACK 控制，必须通过将 I2C_CR1 寄存器中的 SBC 位置 1 来使能从字节控制模式。更多详细信息，请参见*从器件字节控制模式*。

主机通知协议

该外设通过将 I2C_CR1 寄存器中的 SMBHEN 位置 1 来支持主机通知协议。在这种情况下，主机将应答 SMBus 主机地址 (0b0001 000)。

使用该协议时，器件用作主器件，而主机用作从器件。

SMBus 报警

器件支持 SMBus ALERT 可选信号。只具备从功能的器件可通过 SMBALERT# 引脚向主机发出信号，指示它想要通信。主机会处理该中断并通过报警响应地址 (0b0001 100) 同时访问所有 SMBALERT# 器件。只有那些将 SMBALERT# 拉到低电平的器件会应答报警响应地址。

如果配置为从器件 (SMBHEN = 0)，则通过将 I2C_CR1 寄存器中的 ALERTEN 位置 1 来将 SMBA 引脚拉为低电平。这同时还会使能报警响应地址。

如果配置为主机 (SMBHEN = 1)，则当 SMBA 引脚上检测到下降沿且 ALERTEN = 1 时，I2C_ISR 寄存器中的 ALERT 标志置 1。如果 I2C_CR1 寄存器中的 ERRIE 位置 1，将生成中断。当 ALERTEN = 0 时，即使外部 SMBA 引脚为低电平，ALERT 线也将被视为高电平。

如果无需 SMBus ALERT 引脚，则当 ALERTEN = 0 时，SMBA 引脚可用作标准 GPIO。

数据包错误校验

SMBus 规范中引入了数据包错误校验机制来提高可靠性和通信稳定性。数据包错误校验的实施方式是在每次消息传输结束时附加数据包错误代码 (PEC)。PEC 的计算方式是对所有消息字节（包括地址和读/写位）使用 CRC-8 多项式 $C(x) = x^8 + x^2 + x + 1$。

外设内置了硬件 PEC 计算器，可在接收到的字节与硬件计算的 PEC 不匹配时自动发送否定应答信号。

超时

该外设内置了硬件定时器，以便符合 SMBus 规范中定义的三个超时。

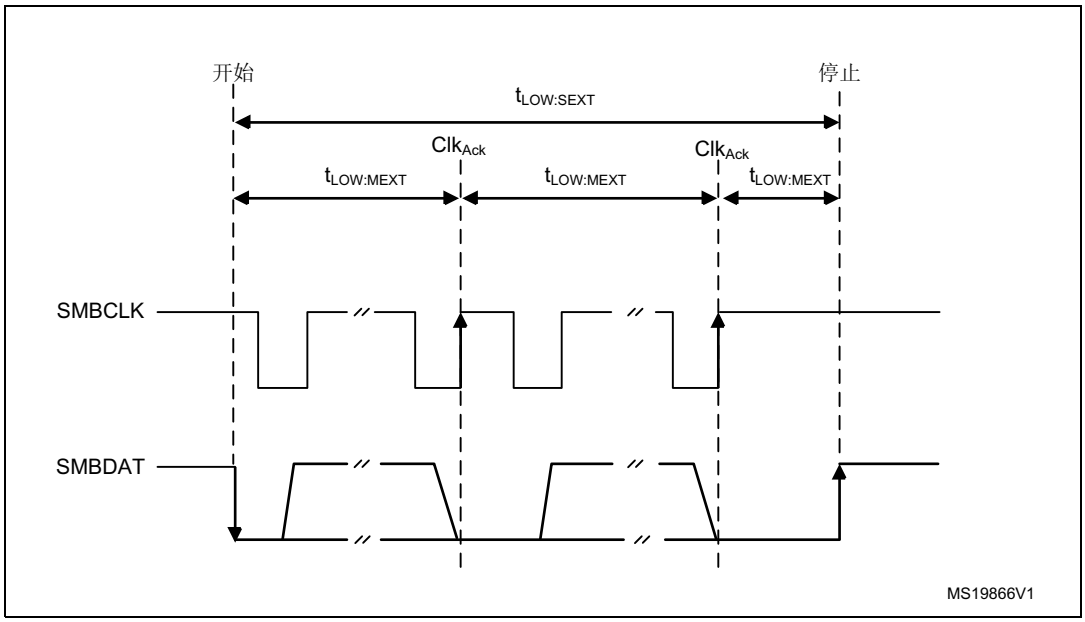
表 260. SMBus 超时规范

符号	参数	限值		单位
		最小值	最大值	
$t_{TIMEOUT}$	检测时钟低电平超时	25	35	ms
$t_{LOW:SEXT}$(1)	累积时钟低电平延长时间（从器件）	-	25	
$t_{LOW:MEXT}$(2)	累积时钟低电平延长时间（主器件）	-	10	

1. $t_{LOW:SEXT}$ 是一段累积时间，即给定从器件在一条消息的最初起始到停止期间时钟信号可延长的时间。其他从器件或主器件也可能延长时钟，进而导致时钟低电平总延长时间超过 $t_{LOW:SEXT}$。因此，测量该参数时该全速主器件只寻址一个从器件。

2. $t_{LOW:MEXT}$ 是一段累积时间，即主器件在消息的每个字节（定义为 START 到 ACK、ACK 到 ACK 或 ACK 到 STOP）内时钟信号可延长的时间。从器件或其他主器件也可能延长时钟，进而导致时钟低电平总时间超过 $t_{LOW:MEXT}$（针对给定字节）。因此，测量该参数时该全速主器件只寻址一个从器件。

图 395. $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的超时间隔



总线空闲检测

如果主器件检测到时钟和数据信号的高电平时间 $t_{IDLE} > t_{HIGH,MAX}$，则认为总线空闲（请参见 *I2C 时序*）。

该时序参数已考虑如下情况：主器件已动态添加至总线，但可能尚未检测到 SMBCLK 或 SMBDAT 线上的状态转换。在这种情况下，主器件必须等待足够长的时间，以确定当前未进行传输。外设支持硬件总线空闲检测。

34.4.12 SMBus 初始化

仅当支持 SMBus 功能（请参见*第 34.3 节*）时，才涉及本节内容。

除了 I2C 初始化之外，还必须进行一些其他的特定初始化，以便执行 SMBus 通信。

接收的命令和数据应答控制（从模式）

SMBus 接收器必须能够对接收到的每个命令或数据进行 NACK 应答。要在从模式下实现 ACK 控制，必须通过将 I2C_CR1 寄存器中的 SBC 位置 1 来使能从字节控制模式。更多详细信息，请参见*从器件字节控制模式*。

特定地址（从模式）

必要时必须使能特定的 SMBus 地址。更多详细信息，请参见*总线空闲检测*。

- 通过将 I2C_CR1 寄存器中的 SMBDEN 位置 1 来使能 SMBus 器件默认地址 (0b1100 001)。
- 通过将 I2C_CR1 寄存器中的 SMBHEN 位置 1 来使能 SMBus 主机地址 (0b0001 000)。
- 通过将 I2C_CR1 寄存器中的 ALERTEN 位置 1 来使能报警响应地址 (0b0001100)。

数据包错误校验

通过将 I2C_CR1 寄存器中的 PECEN 位置 1 来使能 PEC 的计算。然后，借助硬件字节计数器（I2C_CR2 寄存器中的 NBYTES[7:0]）来管理 PEC 传输。使能 I2C 之前，必须配置 PECEN 位。

PEC 传输由硬件字节计数器来管理，因此在从模式下连接 SMBus 时必须将 SBC 位置 1。当 PECBYTE 位置 1 且 RELOAD 位清零时，传输完 NBYTES - 1 字节的数据后会传输 PEC。如果 RELOAD 置 1，PECBYTE 将不起作用。

小心： 使能 I2C 时，不允许更改 PECEN 配置。

表 261. 带 PEC 的 SMBus 配置

模式	SBC 位	RELOAD 位	AUTOEND 位	PECBYTE 位
主 Tx/Rx NBYTES + PEC+ STOP	x	0	1	1
主 Tx/Rx NBYTES + PEC + ReSTART	x	0	0	1
从 Tx/Rx + PEC	1	0	x	1

超时检测

将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 和 TEXTEN 位置 1 来使能超时检测。定时器必须按如下方式编程：即在 SMBus 规范规定的时间最大值之前检测出超时情况。

- $t_{TIMEOUT}$ 检查
  要使能 $t_{TIMEOUT}$ 检查，必须将 12 位 TIMEOUTA[11:0] 位编程为定时器重载值，以检查 $t_{TIMEOUT}$ 参数。必须将 TIDLE 位配置为 0，以检测 SCL 低电平超时。
  然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 位置 1 来使能定时器。
  如果 SCL 的低电平持续时间超过 (TIMEOUTA + 1) x 2048 x t_{I2CCLK}，I2C_ISR 寄存器中的 TIMEOUT 标志将置 1。
  请参见*表 262*。

小心： TIMEOUTEN 位置 1 时，不允许更改 TIMEOUTA[11:0] 位和 TIDLE 位的配置。

- $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 检查
  必须根据外设配置为主器件还是从器件来配置 12 位 TIMEOUTB 定时器，以便为从器件校验 $t_{LOW:SEXT}$，为主器件校验 $t_{LOW:MEXT}$。由于标准只规定了最大值，因此用户可以为这两个参数选择相同的值。然后，通过将 I2C_TIMEOUTR 寄存器中的 TEXTEN 位置 1 来使能定时器。
  如果 SMBus 外设延长 SCL 的累积时间超过 (TIMEOUTB + 1) x 2048 x t_{I2CCLK}，并且达到*总线空闲检测*一节给出的超时间隔，则 I2C_ISR 寄存器中的 TIMEOUT 标志将置 1。
  请参见*表 263*

小心： TEXTEN 位置 1 时，不允许更改 TIMEOUTB 配置。

总线空闲检测

要使能 t_{IDLE} 检查，必须将 12 位 TIMEOUTA[11:0] 位域编程为定时器重载值，以获取 t_{IDLE} 参数。必须将 TIDLE 位配置为“1”，以检测 SCL 和 SDA 高电平超时。然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 位置 1 来使能定时器。

如果 SCL 和 SDA 线的高电平持续时间超过 (TIMEOUTA + 1) x 4 x t_{I2CCLK}，I2C_ISR 寄存器中的 TIMEOUT 标志将置 1。

请参见*表 264*。

小心： TIMEOUTEN 置 1 时，不允许更改 TIMEOUTA 和 TIDLE 配置。

34.4.13 SMBus：I2C_TIMEOUTR 寄存器配置示例

仅当支持 SMBus 功能时，才涉及本节内容。请参见*第 34.3 节*。

- 将 $t_{TIMEOUT}$ 的最大持续时间配置为 25 ms：

表 262. TIMEOUTA 设置示例（最大 $t_{TIMEOUT}$ = 25 ms）

f_{I2CCLK}	TIMEOUTA[11:0] 位	TIDLE 位	TIMEOUTEN 位	$t_{TIMEOUT}$
8 MHz	0x61	0	1	98 x 2048 x 125 ns = 25 ms
16 MHz	0xC3	0	1	196 x 2048 x 62.5 ns = 25 ms

- 将 $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的最大持续时间配置为 8 ms：

表 263. TIMEOUTB 设置示例

f_{I2CCLK}	TIMEOUTB[11:0] 位	TEXTEN 位	$t_{LOW:EXT}$
8 MHz	0x1F	1	32 x 2048 x 125 ns = 8 ms
16 MHz	0x3F	1	64 x 2048 x 62.5 ns = 8 ms

- 将 t_{IDLE} 的最大持续时间配置为 50 µs

表 264. TIMEOUTA 设置示例（最大 t_{IDLE} = 50 µs）

f_{I2CCLK}	TIMEOUTA[11:0] 位	TIDLE 位	TIMEOUTEN 位	t_{TIDLE}
8 MHz	0x63	1	1	100 x 4 x 125 ns = 50 µs
16 MHz	0xC7	1	1	200 x 4 x 62.5 ns = 50 µs

34.4.14 SMBus 从模式

仅当支持 SMBus 功能（请参见*第 34.3 节*）时，才涉及本节内容。

除了 I2C 从模式传输管理（请参见*第 34.4.8 节*）之外，还提供了额外的软件流程来支持 SMBus。

SMBus 从发送器

在 SMBus 模式下作为从发送器时，必须将 SBC 编程为 1，以使能在完成已编程数据字节数的传输后进行 PEC 传输。当 PECBYTE 位置 1 时，NBYTES[7:0] 中编程的字节数包含 PEC 传输。在这种情况下，总 TXIS 中断数为 NBYTES - 1，如果主器件在完成 NBYTES - 1 字节的数据传输后请求传输额外的字节，则将自动发送 I2C_PECR 寄存器的内容。

小心： 当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

图 396. SMBus 从发送器的传输序列流程（N 字节 + PEC）

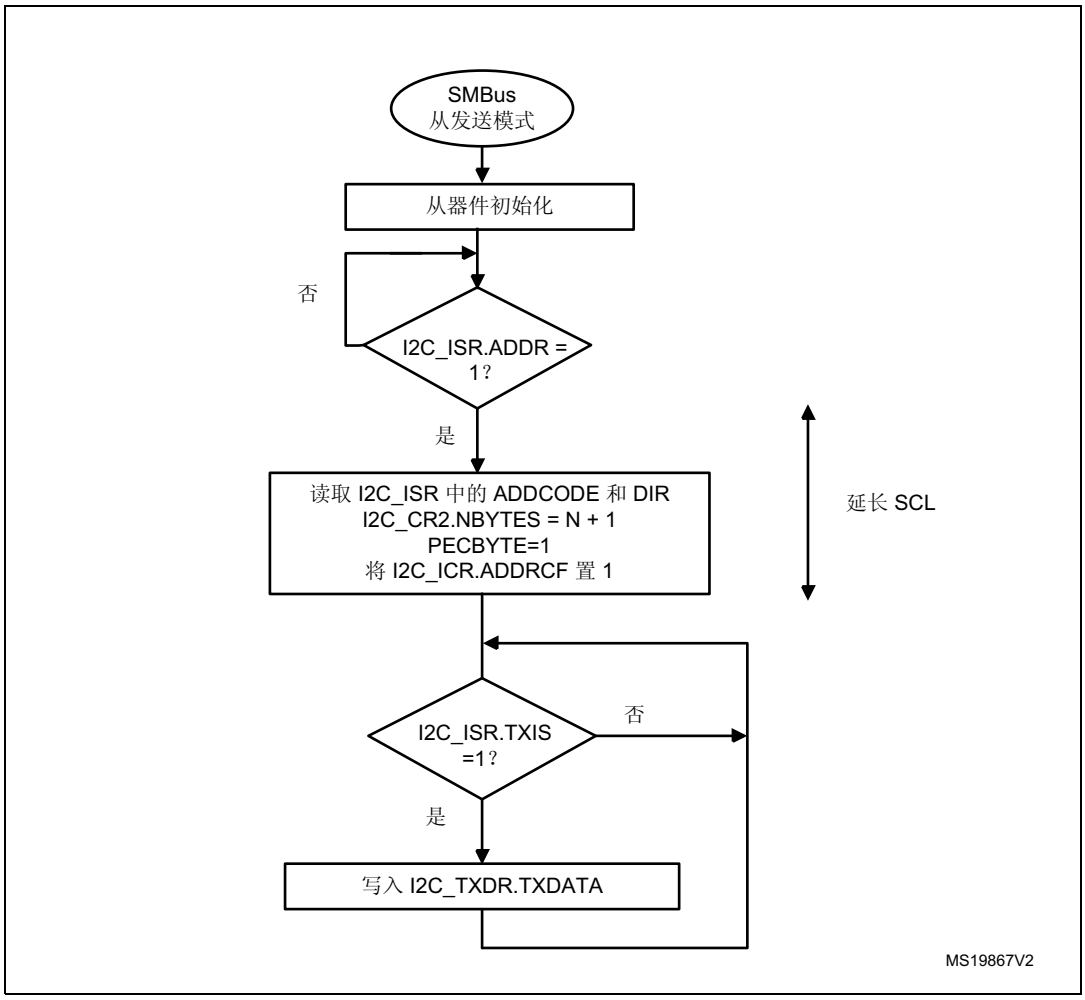
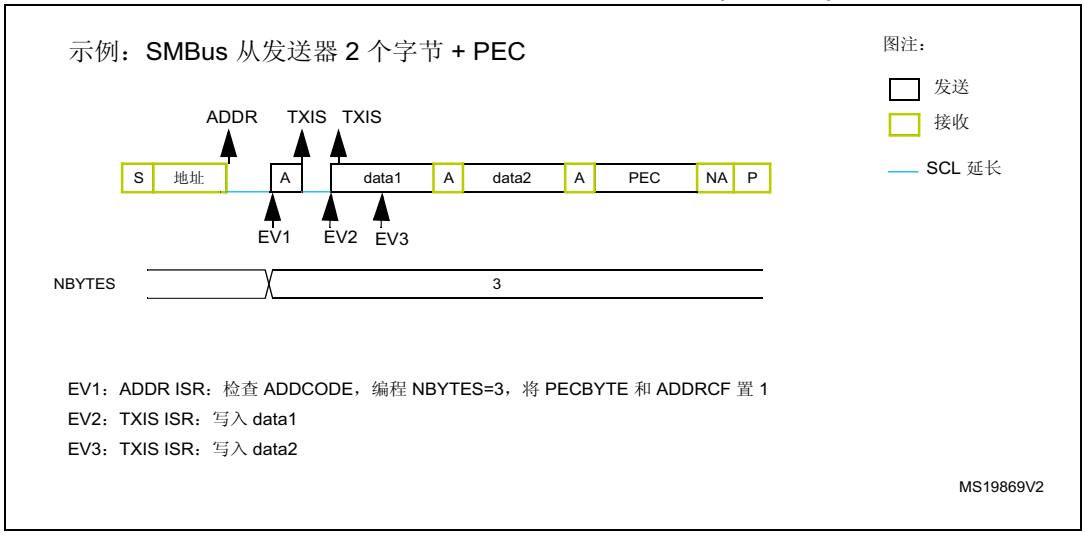


图 397. SMBus 从发送器的传输总线图 (SBC = 1)



SMBus 从接收器

在 SMBus 模式下使用 I2C 时，必须将 SBC 编程为 1，以便在完成已编程数据字节数的传输后进行 PEC 校验。要对每个字节进行 ACK 控制，必须选择重载模式 (RELOAD = 1)。更多详细信息，请参见*从器件字节控制模式*。

要校验 PEC 字节，必须将 RELOAD 位清零并将 PECBYTE 位置 1。在这种情况下，当接收到 NBYTES - 1 字节的数据后，接收的下一个字节将与内部 I2C_PECR 寄存器的内容作比较。如果比较不匹配，则将自动生成 NACK 信号；如果比较匹配，则将自动生成 ACK 信号，而与 ACK 位的值无关。PEC 字节一经接收，便会像任何其它数据一样复制到 I2C_RXDR 寄存器中，并且 RXNE 标志将置 1。

当 PEC 不匹配时，PECERR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

如果无需 ACK 软件控制，用户可编程 PECBYTE = 1，在同一写操作下，将 NBYTES 编程为连续接收的字节数。接收到 NBYTES - 1 字节的数据后，会将接收的下一个字节视为 PEC 进行校验。

小心： 当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

图 398. SMBus 从接收器的传输序列流程（N 字节 + PEC）

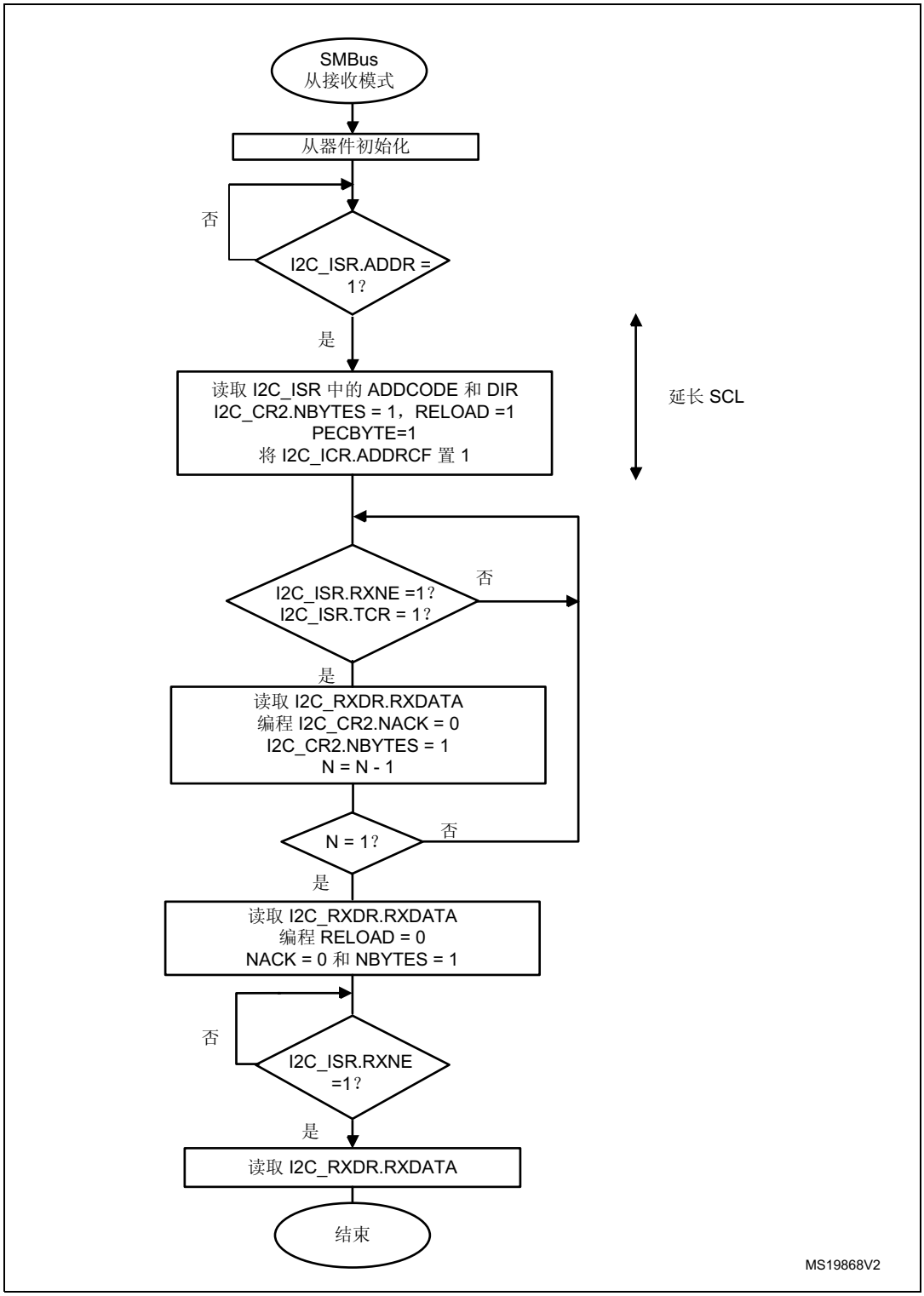
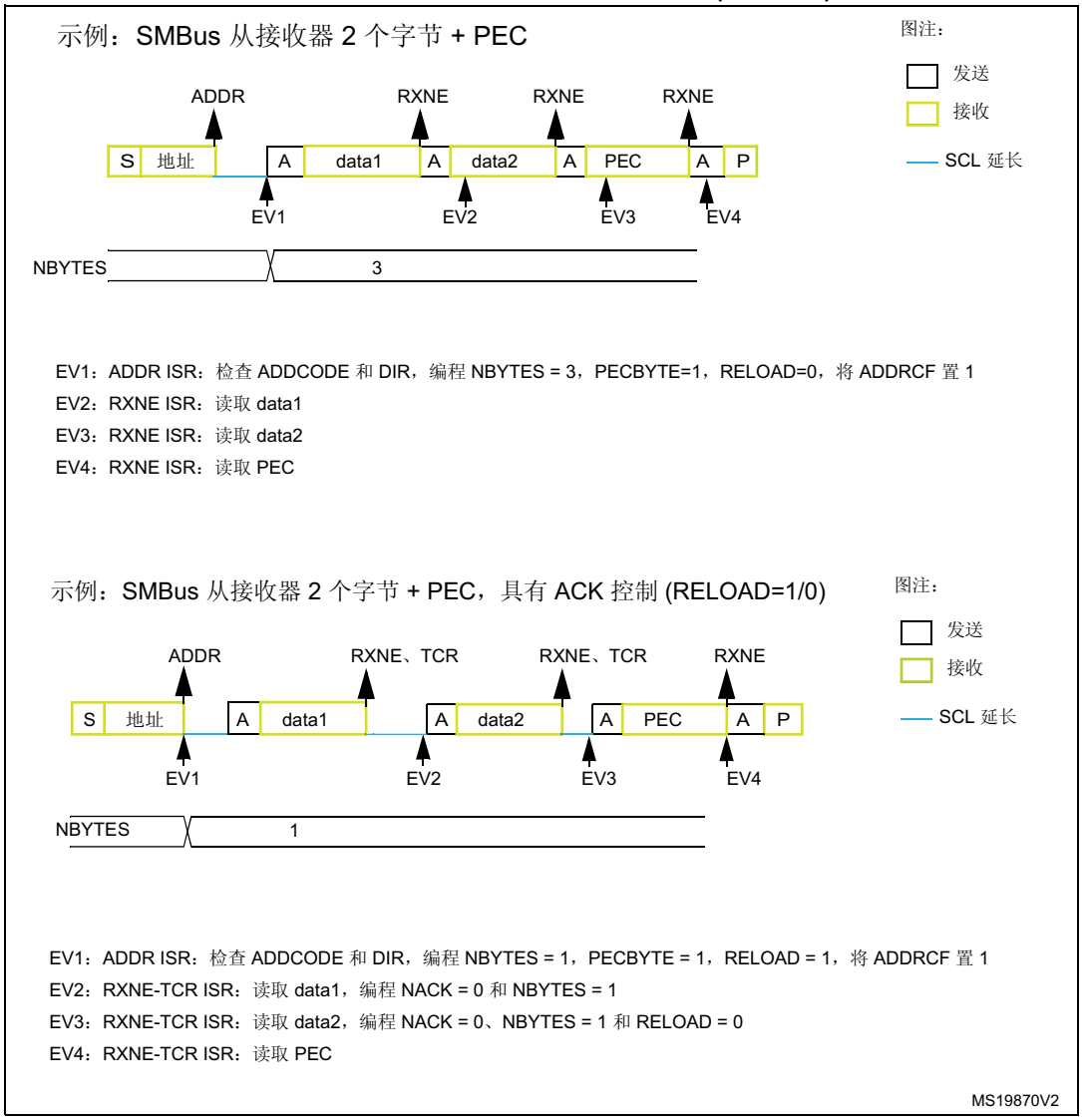


图 399. SMBus 从接收器的总线传输图 (SBC = 1)



仅当支持 SMBus 功能（请参见*第 34.3 节*）时，才涉及本节内容。

除了 I2C 主模式传输管理（请参见*第 34.4.9 节*）之外，还提供了额外的软件流程来支持 SMBus。

SMBus 主发送器

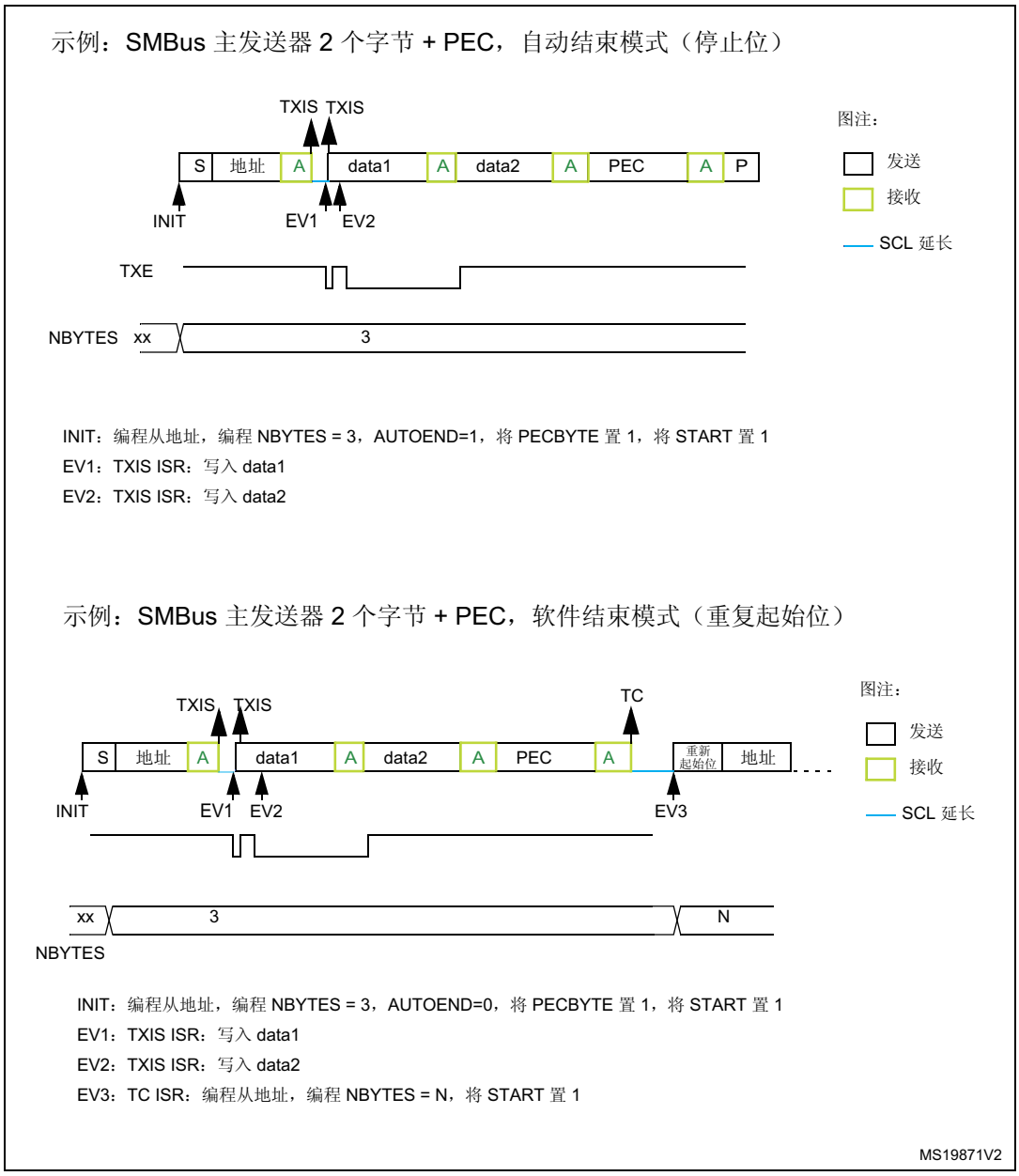
当 SMBus 主器件想要发送 PEC 时，必须在 START 位置 1 前，将 PECBYTE 位置 1 并在 NBYTES[7:0] 位域中设置字节数。在这种情况下，总 TXIS 中断数为 NBYTES - 1。因此，如果 PECBYTE 位在 NBYTES = 0x1 时置 1，则将自动发送 I2C_PECR 寄存器的内容。

如果 SMBus 主器件想要在 PEC 后发送停止位，则必须选择自动结束模式 (AUTOEND = 1)。在这种情况下，传输 PEC 后将自动发送停止位。

如果 SMBus 主器件想要在 PEC 后发送重复起始位，则必须选择软件模式 (AUTOEND = 0)。在这种情况下，发送 NBYTES - 1 字节的数据后，将发送 I2C_PECR 寄存器的内容，TC 标志将在传输完 PEC 之后置 1，SCL 线的低电平时间将延长。必须在 TC 中断子程序中设置重复起始位。

小心： 当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

图 400. SMBus 主发送器的总线传输图



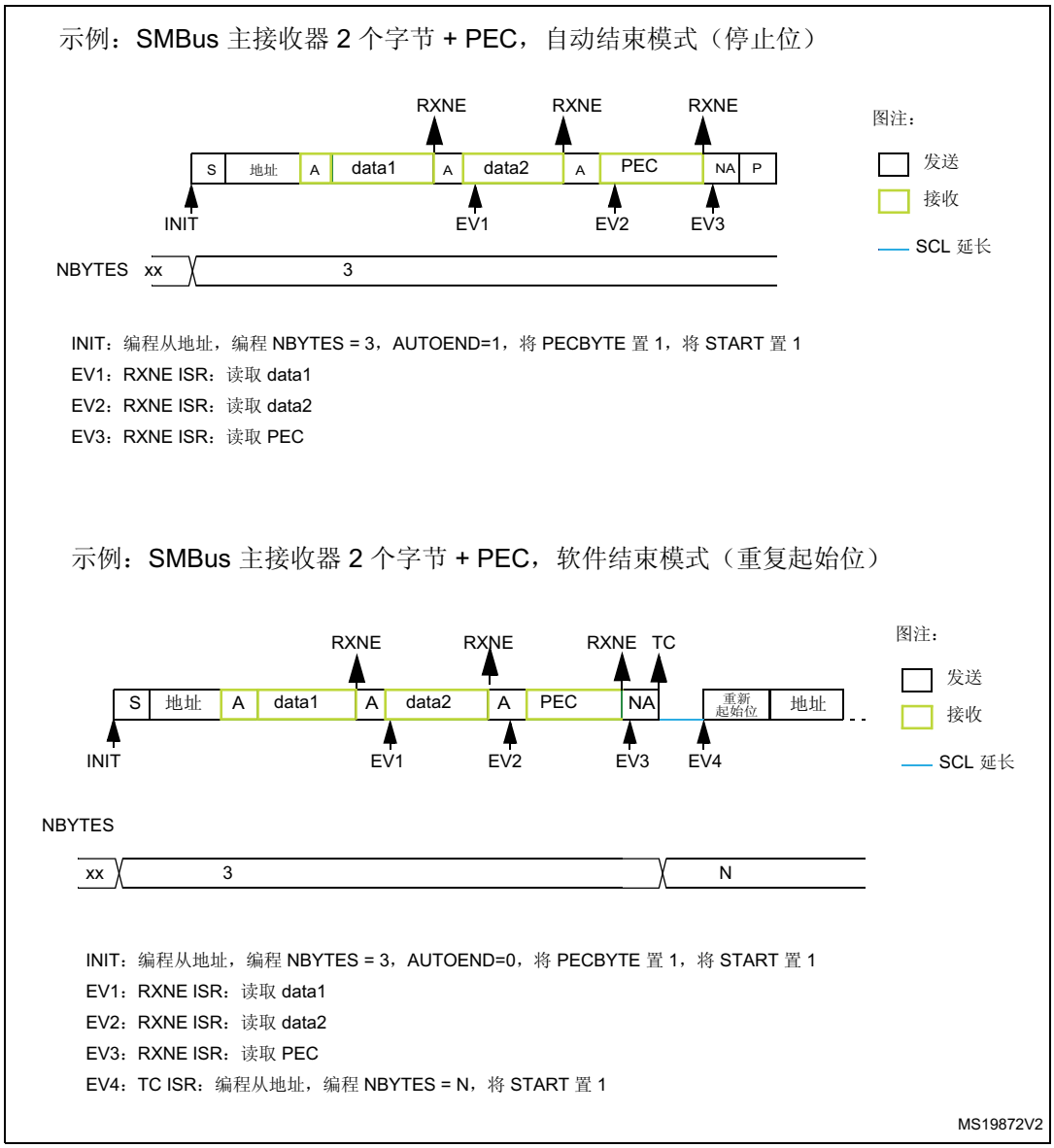
SMBus 主接收器

当 SMBus 主器件想要接收 PEC，并在传输结束后接收 STOP 时，可选择自动结束模式 (AUTOEND = 1)。将 START 位置 1 之前，必须将 PECBYTE 位置 1 并设置从地址。在这种情况下，当接收到 NBYTES - 1 字节的数据后，将自动使用 I2C_PECR 寄存器的内容对接收的下一个字节进行校验。PEC 字节（其后跟有停止位）将得到 NACK 响应。

当 SMBus 主接收器想要接收 PEC 字节，并且在传输结束后接收重复起始位时，必须选择软件模式 (AUTOEND = 0)。将 START 位置 1 之前，必须将 PECBYTE 位置 1 并设置从地址。在这种情况下，当接收到 NBYTES - 1 字节的数据后，将自动使用 I2C_PECR 寄存器的内容对接收的下一个字节进行校验。接收到 PEC 字节后，TC 标志将置 1，SCL 线的低电平时间将延长。可以在 TC 中断子程序中设置重复起始位。

小心： 当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

图 401. SMBus 主接收器的总线传输图



34.4.15 地址匹配时从停止模式唤醒

仅当支持从停止模式唤醒功能（请参见*第 34.3 节*）时，才涉及本节内容。

被寻址时，I2C 能够将 MCU 从停止模式唤醒（APB 时钟关断）。支持所有寻址模式。

将 I2C_CR1 寄存器中的 WUPEN 位置 1，可以使能从停止模式唤醒功能。对于 I2CCLK，必须选择 仅限 HSI 和 CSI 振荡器作为时钟源，以便从停止模式唤醒。

在停止模式期间，仅限 HSI 和 CSI 关闭。当检测到 START 时，I2C 接口将 仅限 HSI 和 CSI 接通，并延长 SCL 使其处于低电平直到唤醒 仅限 HSI 和 CSI。

仅限 HSI 和 CSI 随后用来接收地址。

地址匹配的情况下，MCU 唤醒时间内，I2C 延长 SCL 的低电平持续时间。通过软件清零 ADDR 标志后，此延长操作被释放，传输正常进行。

如果地址不匹配，仅限 HSI 和 CSI 再次关断，MCU 不被唤醒。

注意： *如果 I2C 时钟是系统时钟，或者 WUPEN = 0，则接收到 START 后，*仅限 HSI 和 CSI *不会接通。*

只有 ADDR 中断能够唤醒 MCU。因此，当 I2C 以主器件身份或在 ADDR 标志置 1 后以被寻址从器件身份执行传输时，不要进入停止模式。通过在 ADDR 中断程序中清零 SLEEPDEEP 位，然后仅在 STOPF 标志置 1 后再将其置 1，可对此进行管理。

小心： 数字滤波器与从停止模式唤醒功能不兼容。如果 DNF 位不等于 0，则将 WUPEN 位置 1 将不起任何作用。

小心： 只有当 I2C 时钟源为 仅限 HSI 和 CSI 振荡器时，该功能才可用。

小心： 必须使能时钟延长 (NOSTRETCH = 0) 才能确保从停止模式唤醒功能正常工作。

小心： 如果禁止从停止模式唤醒 (WUPEN = 0)，则在进入停止模式前必须禁止 I2C 外设 (PE = 0)。

34.4.16 错误条件

以下错误是可能导致通信失败的条件。

总线错误 (BERR)

当检测到起始位或停止位但不位于第九个 SCL 时钟脉冲之后时，会检测到总线错误。当 SDA 边沿出现且 SCL 为高电平时，会检测到起始或停止位。

只有当 I2C 在传输过程中用作主器件或被寻址为从器件时（即未处于从模式下的地址阶段），才会将总线错误标志置 1。

在从模式下检测到错位的起始位或重复起始位时，I2C 会像接收到正确的起始位一样进入地址识别状态。

检测到总线错误时，I2C_ISR 寄存器中的 BERR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

仲裁丢失 (ARLO)

当 SDA 线上发送高电平但在 SCL 上升沿却采样到低电平时，会检测到仲裁丢失。

- 在主模式下，将在地址阶段、数据阶段和数据应答阶段检测到仲裁丢失。在这种情况下，SDA 线和 SCL 线被释放，起始控制位由硬件清零，主器件自动切换为从模式。
- 在从模式下，将在数据阶段和数据应答阶段检测到仲裁丢失。在这种情况下，传输停止，SCL 和 SDA 线被释放。

检测到仲裁丢失时，I2C_ISR 寄存器中的 ARLO 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

上溢/下溢错误 (OVR)

当满足 NOSTRETCH = 1 和以下条件时，将在从模式下检测到上溢或下溢错误：

- 在接收过程中接收到一个新的字节，但 RXDR 寄存器的值还未被读取。接收的新字节丢失，自动发送 NACK 来响应新字节。
- 在发送过程中：
  - 当 STOPF = 1 且必须发送第一个数据字节时。TXE = 0 时发送 I2C_TXDR 寄存器的内容，否则发送 0xFF。
  - 必须发送一个新字节但尚未向 I2C_TXDR 寄存器写入数据时，将发送 0xFF。

检测到上溢或下溢错误时，I2C_ISR 寄存器中的 OVR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

数据包错误校验错误 (PECERR)

仅当支持 SMBus 功能（请参见*第 34.3 节*）时，才涉及本节内容。

当接收到的 PEC 字节与 I2C_PECR 寄存器的内容不匹配时，将检测到 PEC 错误。接收到错误的 PEC 后，将自动发送 NACK。

检测到 PEC 错误时，I2C_ISR 寄存器中的 PECERR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

超时错误 (TIMEOUT)

仅当支持 SMBus 功能（请参见*第 34.3 节*）时，才涉及本节内容。

满足以下任何条件均会出现超时错误：

- TIDLE = 0 且 SCL 的低电平持续时间达到 TIMEOUTA[11:0] 位中定义的时间：这用于检测 SMBus 超时。
- TIDLE = 1 且 SDA 和 SCL 的高电平持续时间达到 TIMEOUTA[11:0] 位中定义的时间：这用于检测总线空闲情况。
- 主器件的累积时钟低电平延长时间达到了 TIMEOUTB[11:0] 位中定义的时间（SMBus $t_{LOW:MEXT}$ 参数）。
- 从器件的累积时钟低电平延长时间达到了 TIMEOUTB[11:0] 位中定义的时间（SMBus $t_{LOW:SEXT}$ 参数）。

当在主模式下检测到超时时，将自动发送停止位。

当在从模式下检测到超时时，将自动释放 SDA 和 SCL 线。

检测到超时错误时，I2C_ISR 寄存器中的 TIMEOUT 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

报警 (ALERT)

仅当支持 SMBus 功能（请参见*第 34.3 节*）时，才涉及本节内容。

当 I2C 接口配置为主机 (SMBHEN = 1)、使能了报警引脚检测 (ALERTEN = 1) 并且在 SMBA 引脚上检测到下降沿时，ALERT 标志将置 1。如果 I2C_CR1 寄存器中的 ERRIE 位置 1，将生成中断。

34.4.17 DMA 请求

使用 DMA 进行发送

将 I2C_CR1 寄存器中的 TXDMAEN 位置 1 可以使能 DMA（直接存储器访问）进行发送。当 TXIS 位置 1 时，数据将从由 DMA 外设配置的 SRAM 区（请参见“直接存储器访问控制器 (DMA)”部分）装载进 I2C_TXDR 寄存器。

只有数据字节采用 DMA 进行传输。

- 在主模式下：初始化、从地址、方向、字节数和起始位均由软件编程（发送的从地址无法通过 DMA 传输）。当所有数据均通过 DMA 传输时，必须在起始位置 1 之前初始化 DMA。传输结束由 NBYTES 计数器来管理。请参见*主发送器*。
- 在从模式下：
  - 当 NOSTRETCH = 0 时，如果所有数据均通过 DMA 传输，则必须在地址匹配事件之前（或清零 ADDR 之前在 ADDR 中断子程序中）初始化 DMA。
  - 当 NOSTRETCH = 1 时，必须在地址匹配事件之前初始化 DMA。
- 支持 SMBus 时：PEC 传输由 NBYTES 计数器管理。请参见*SMBus 从发送器*和*SMBus 主发送器*。

注意： *如果使用 DMA 进行发送，则无需使能 TXIE 位。*

使用 DMA 进行接收

将 I2C_CR1 寄存器中的 RXDMAEN 位置 1 可以使能 DMA（直接存储器访问）进行接收。当 RXNE 位置 1 时，数据将从 I2C_RXDR 寄存器装载进由 DMA 外设配置的 SRAM 区（请参见“直接存储器访问控制器 (DMA)”部分）。只有数据字节（包括 PEC）采用 DMA 进行传输。

- 在主模式下：初始化、从地址、方向、字节数和起始位均由软件编程。当所有数据均通过 DMA 传输时，必须在起始位置 1 之前初始化 DMA。传输结束由 NBYTES 计数器来管理。
- 在从模式下，当 NOSTRETCH = 0 时，如果所有数据均通过 DMA 传输，则必须在地址匹配事件之前（或清零 ADDR 标志之前在 ADDR 中断子程序中）初始化 DMA。
- 如果支持 SMBus（请参见*第 34.3 节*）：PEC 传输由 NBYTES 计数器管理。请参见*SMBus 从接收器*和*SMBus 主接收器*。

注意： *如果使用 DMA 进行接收，则无需使能 RXIE 位。*

34.4.18 调试模式

当微控制器进入调试模式时（内核停止），SMBus 超时定时器会根据 DBG 模块中的 DBG_I2Cx_ 配置位选择继续正常工作或者停止工作。

34.5 I2C 低功耗模式

表 265. 低功耗模式对 I2C 的影响

模式	说明
睡眠	无影响。I2C 中断可使器件退出睡眠模式。
停止(1)	I2C 模块的寄存器内容仍被保持。 – WUPEN = 1 并且 I2C 的时钟由内部振荡器 (仅限 HSI 和 CSI) 提供：地址识别功能正常。I2C 地址匹配条件会导致器件退出停止模式。 – WUPEN = 0：必须在进入停止模式之前禁止 I2C。
待机	I2C 外设掉电，退出待机模式后必须重新初始化。

1. 有关每个实例支持的停止模式的信息，请参见*第 34.3 节*。如果不支持从特定停止模式唤醒，则必须在进入此停止模式之前禁止实例。

34.6 I2C 中断

下表列出了 I2C 中断请求。

表 266. I2C 中断请求

中断缩写		中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止模式	退出待机模式
I2C	I2C_EV	接收缓冲区非空	RXNE	RXIE	读取 I2C_RXDR 寄存器	是	否	否
		发送缓冲区中断状态	TXIS	TXIE	写入 I2C_TXDR 寄存器			
		停止位检测中断标志	STOPF	STOPIE	写入 STOPCF = 1			
		传输完成等待重载	TCR	TCIE	写入 I2C_CR2 (NBYTES[7:0] ≠ 0)			
		传输完成	TC		写入 START = 1 或 STOP = 1			
		地址匹配	ADDR	ADDRIE	写入 ADDRCF = 1		是(1)	
		接收到 NACK 应答	NACKF	NACKIE	写入 NACKCF = 1		否	
	I2C_ER	总线错误	BERR	ERRIE	写入 BERRCF = 1	是	否	否
		仲裁丢失	ARLO		写入 ARLOCF = 1			
		上溢/下溢	OVR		写入 OVRCF = 1			
		PEC 错误	PECERR		写入 PECERRCF = 1			
		超时/t_{LOW} 错误	TIMEOUT		写入 TIMEOUTCF = 1			
		SMBus 报警	ALERT		写入 ALERTCF = 1			

1. 仅当 I2C 实例支持从停止模式唤醒功能时，ADDR 匹配事件才能将器件从停止模式唤醒。请参见*第 34.3 节*。

34.7 I2C 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节*。

外设寄存器按字（32 位）进行访问。

34.7.1 I2C 控制寄存器 1 (I2C_CR1)

I2C control register 1

偏移地址：0x00

复位值：0x0000 0000

访问：无等待周期，正在执行写访问时出现另一个写访问的情况除外。在这种情况下，会在第二个写访问中插入等待周期，直到前一个写访问完成为止。第二个写访问的延时最长可达 2 x i2c_pclk + 6 x i2c_ker_ck。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STOPFACLR	ADDRACLR	Res.	Res.	Res.	Res.	Res.	FMP	PECEN	ALERTEN	SMBDEN	SMBHEN	GCEN	WUPEN	NOSTRETCH	SBC
rw	rw						rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDMAEN	TXDMAEN	Res.	ANFOFF	DNF[3:0]				ERRIE	TCIE	STOPIE	NACKIE	ADDRIE	RXIE	TXIE	PE
rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **STOPFACLR**：停止位检测标志 (STOPF) 自动清零 (STOP detection flag (STOPF) automatic clear)

0：STOPF 标志由硬件置 1，由软件通过将 STOPCF 位置 1 来清零。

1：STOPF 标志由硬件保持清零状态。该状态可用于 NOSTRETCH 从模式，以避免在下一次数据传输前因未将 STOPF 标志清零而导致上溢错误。这样，只需通过 DMA 即可实现从数据管理，而无需任何来自外设的中断。

位 30 **ADDRACLR**：地址匹配标志 (ADDR) 自动清零 (Address match flag (ADDR) automatic clear)

0：ADDR 标志由硬件置 1，由软件通过将 ADDRCF 位置 1 来清零。

1：ADDR 标志由硬件保持清零状态。该状态可用于从模式，以避免在 I2C 仅使能一个从地址时延长 ADDR 时钟。这样，只需通过 DMA 即可实现从数据管理，而无需任何来自外设的中断。

位 29:25 保留，必须保持复位值。

位 24 **FMP**：超快速模式 20 mA 驱动使能 (Fast-mode Plus 20 mA drive enable)

0：禁止 20 mA I/O 驱动

1：使能 20 mA I/O 驱动

位 23 **PECEN**：PEC 使能 (PEC enable)

0：禁止 PEC 计算

1：使能 PEC 计算

注意：如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 22 **ALERTEN**：SMBus 报警使能 (SMBus alert enable)

0：在主机模式 (SMBHEN = 1) 下，不支持 SMBus 报警引脚 (SMBA)。在设备模式 (SMBHEN = 0) 下，会释放 SMBA 引脚并禁止报警响应地址头（0001100x 后跟 NACK）。

1：在主机模式 (SMBHEN = 1) 下，支持 SMBus 报警引脚。在设备模式 (SMBHEN = 0) 下，会将 SMBA 引脚驱动为低电平并使能报警响应地址头（0001100x 后跟 ACK）。

注意：当 ALERTEN = 0 时，SMBA 引脚可用作标准 GPIO。

如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 21 **SMBDEN**：SMBus 器件默认地址使能 (SMBus device default address enable)

0：禁止器件默认地址。不对地址 0b1100001x 应答。

1：使能器件默认地址。对地址 0b1100001x 应答。

注意：如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 20 **SMBHEN**：SMBus 主机地址使能 (SMBus host address enable)

0：禁止主机地址。不对地址 0b0001000x 应答。

1：使能主机地址。对地址 0b0001000x 应答。

注意：如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 19 **GCEN**：广播呼叫使能 (General call enable)

0：禁止广播呼叫。不对地址 0b00000000 应答。

1：使能广播呼叫。对地址 0b00000000 应答。

位 18 **WUPEN**：从停止模式唤醒使能 (Wake-up from Stop mode enable)

0：禁止从停止模式唤醒。

1：使能从停止模式唤醒。

注意：如果不支持从停止模式唤醒功能，该位保留并由硬件强制清零。请参见第 34.3 节。

注意：只有当 DNF = “0000” 时，WUPEN 才能置 1

位 17 **NOSTRETCH**：时钟延长禁止 (Clock stretching disable)

该位用于在从模式下禁止时钟延长。它在主模式下必须保持清零。

0：使能时钟延长

1：禁止时钟延长

注意：该位只能在 I2C 禁止时 (PE = 0) 编程。

位 16 **SBC**：从器件字节控制 (Slave byte control)

该位用于在从设备模式下使能硬件字节控制。

0：禁止从器件字节控制

1：使能从器件字节控制

位 15 **RXDMAEN**：DMA 接收请求使能 (DMA reception requests enable)

0：禁止 DMA 接收请求

1：使能 DMA 接收请求

位 14 **TXDMAEN**：DMA 发送请求使能 (DMA transmission requests enable)

0：禁止 DMA 发送请求

1：使能 DMA 发送请求

位 13 保留，必须保持复位值。

位 12 **ANFOFF**：模拟噪声滤波器关闭 (Analog noise filter OFF)

0：使能模拟噪声滤波器

1：禁止模拟噪声滤波器

注意：该位只能在 I2C 禁止时 (PE = 0) 编程。

位 11:8 **DNF[3:0]**：数字噪声滤波器 (Digital noise filter)

这些位用于配置 SDA 和 SCL 输入端的数字噪声滤波器。数字滤波器可滤除脉宽 DNF[3:0] * t_{I2CCLK} 以下的尖峰

0000：禁止数字滤波器

0001：使能数字滤波器，可滤除的噪声尖峰脉宽可达 1 t_{I2CCLK}

...

1111：使能数字滤波器，可滤除的噪声尖峰脉宽可达 15 t_{I2CCLK}

注意：如果模拟滤波器已使能，数字滤波将叠加在模拟滤波之上。

该滤波器只能在 I2C 禁止时 (PE = 0) 编程。

位 7 **ERRIE**：错误中断使能 (Error interrupt enable)

0：禁止错误检测中断

1：使能错误检测中断

注意：以下任一错误均会生成中断：

仲裁丢失 (ARLO)

总线错误检测 (BERR)

上溢/下溢 (OVR)

超时检测 (TIMEOUT)

PEC 错误检测 (PECERR)

报警引脚事件检测 (ALERT)

位 6 **TCIE**：传输完成中断使能 (Transfer complete interrupt enable)

0：禁止传输完成中断

1：使能传输完成中断

注意：以下任一事件均会生成中断：

传输完成 (TC)

传输完成等待重载 (TCR)

位 5 **STOPIE**：停止位检测中断使能 (Stop detection interrupt enable)

0：禁止停止位检测 (STOPF) 中断

1：使能停止位检测 (STOPF) 中断

位 4 **NACKIE**：接收到否定应答中断使能 (Not acknowledge received interrupt enable)

0：禁止接收到否定应答 (NACKF) 中断

1：使能接收到否定应答 (NACKF) 中断

位 3 **ADDRIE**：地址匹配中断使能（仅从模式）(Address match interrupt enable (slave only))

0：禁止地址匹配 (ADDR) 中断

1：使能地址匹配 (ADDR) 中断

位 2 **RXIE**：RX 中断使能 (RX interrupt enable)

0：禁止接收 (RXNE) 中断

1：使能接收 (RXNE) 中断

位 1 **TXIE**：TX 中断使能 (TX interrupt enable)

0：禁止发送 (TXIS) 中断

1：使能发送 (TXIS) 中断

位 0 **PE**：外设使能 (Peripheral enable)

0：禁止外设

1：使能外设

注意：当 PE = 0 时，将释放 I2C SCL 线和 SDA 线。内部状态机和状态位均恢复为复位值。清零时，PE 必须保持低电平状态至少三个 APB 时钟周期。

34.7.2 I2C 控制寄存器 2 (I2C_CR2)

I2C control register 2

偏移地址：0x04

复位值：0x0000 0000

访问：无等待周期，正在执行写访问时出现另一个写访问的情况除外。在这种情况下，会在第二个写访问中插入等待周期，直到前一个写访问完成为止。第二个写访问的延时最长可达 2 x i2c_pclk + 6 x i2c_ker_ck。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	PEC BYTE	AUTO END	RE LOAD	NBYTES[7:0]							
					rs	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NACK	STOP	START	HEAD10R	ADD10	RD_ WRN	SADD[9:0]									
rs	rs	rs	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:27 保留，必须保持复位值。

位 26 **PECBYTE**：数据包错误校验字节 (Packet error checking byte)

该位由软件置 1，并可在 PEC 传输完成时、接收到停止位或匹配地址时、或者 PE=0 时由硬件清零。

0：不传输 PEC。

1：请求 PEC 发送/接收。

注意：向该位写入 0 不起作用。

当 RELOAD 置 1 时，该位不起作用。

当 SBC = 0 时，该位在从模式下不起作用。

如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 25 **AUTOEND**：自动结束模式（主模式）(Automatic end mode (master mode))

该位由软件置 1 和清零。

0：软件结束模式：当 NBYTES 数据传输完成时，TC 标志将置 1，SCL 的低电平时间将延长直到相应软件操作结束。

1：自动结束模式：当 NBYTES 数据传输完成时，将自动发送停止位。

注意：在从模式下，或当 RELOAD 位置 1 时，该位不起作用。

位 24 **RELOAD**：NBYTES 重载模式 (NBYTES reload mode)

该位由软件置 1 和清零。

0：传输 NBYTES 数据（后跟停止位或重复起始位）之后即完成传输。

1：传输 NBYTES 数据之后未完成传输（将重载 NBYTES）。当 NBYTES 数据传输完成时，TCR 标志将置 1，SCL 的低电平时间将延长直到相应软件操作结束。

位 23:16 **NBYTES[7:0]**：字节数 (Number of bytes)

在此设置待发送/接收的字节数。在从模式下，当 SBC = 0 时，该位域为无关位域。

注意：START 位置 1 时，不允许更改这些位。

位 15 **NACK**：NACK 生成（从模式）(NACK generation (slave mode))

该位由软件置 1，并可在发送 NACK 时、接收到停止位或匹配地址时、或者 PE = 0 时由硬件清零。

0：在当前接收的字节后发送 ACK。

1：在当前接收的字节后发送 NACK。

注意：向该位写入 0 不起作用。

该位仅在从模式下使用：在主接收器模式下，无论 NACK 位的值为何，最后一个字节（后跟停止位或重复起始位）后都将自动生成 NACK。

当从接收器 NOSTRETCH 模式下发生上溢时，无论 NACK 位的值为何，都将自动生成 NACK。

使能硬件 PEC 校验时 (PECBYTE = 1)，PEC 应答值与 NACK 值无关。

位 14 **STOP**：停止位生成（主模式）(Stop generation (master mode))

该位由软件置 1，并可在检测到停止位时或 PE = 0 时由硬件清零。

在主模式下：

0：不生成停止位。

1：在当前字节传输完成后生成停止位。

注意：向该位写入 0 不起作用。

位 13 **START**：生成起始位 (Start generation)

该位由软件置 1，并可在发送起始位（后跟地址序列）之后、发生仲裁丢失时、从模式下发生地址匹配时、出现超时错误时、或者 PE = 0 时由硬件清零。

0：不生成起始位。

1：生成重复起始/起始位：

如果 I2C 已处于主模式下且 AUTOEND = 0，则将该位置 1 会在 NBYTES 传输结束后且 RELOAD = 0 的情况下生成重复起始位。

否则，将该位置 1 会在总线释放后立即生成起始位。

注意：向该位写入 0 不起作用。

即使总线繁忙或 I2C 处于从模式，也可将 START 位置 1。

当 RELOAD 置 1 时，该位不起作用。

位 12 **HEAD10R**：读方向传输时，只发送 10 位地址的前 7 位地址头字节（主接收器模式）(10-bit address header only read direction (master receiver mode))

0：主器件发送完整的 10 位从地址读序列：起始位 + 带写方向的 2 字节 10 位地址 + 重复起始位 + 带读方向的 10 位地址的前 7 位。

1：主器件只发送 10 位地址的前 7 位，后跟读方向。

注意：START 位置 1 时，不允许更改该位。

位 11 **ADD10**：10 位寻址模式（主模式）(10-bit addressing mode (master mode))

0：主器件工作在 7 位寻址模式下

1：主器件工作在 10 位寻址模式下

注意：START 位置 1 时，不允许更改该位。

位 10 **RD_WRN**：传输方向（主模式）(Transfer direction (master mode))

0：主器件请求写传输。

1：主器件请求读传输。

注意：START 位置 1 时，不允许更改该位。

位 9:0 **SADD[9:0]**：从地址（主模式）(Slave address (master mode))

在 7 位寻址模式 (ADD10 = 0) 下：

SADD[7:1] 必须写入待发送的 7 位从地址。SADD[9]、SADD[8] 和 SADD[0] 位为无关位。

在 10 位寻址模式 (ADD10 = 1) 下：

SADD[9:0] 必须写入待发送的 10 位从地址。

注意：START 位置 1 时，不允许更改这些位。

34.7.3 I2C 自身地址 1 寄存器 (I2C_OAR1)

I2C own address 1 register

偏移地址：0x08

复位值：0x0000 0000

访问：无等待周期，正在执行写访问时出现另一个写访问的情况除外。在这种情况下，会在第二个写访问中插入等待周期，直到前一个写访问完成为止。第二个写访问的延时最长可达 2 x i2c_pclk + 6 x i2c_ker_ck。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA1EN	Res.	Res.	Res.	Res.	OA1 MODE	OA1[9:0]									
rw					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15 **OA1EN**：设备自身地址 1 使能 (Own address 1 enable)
0：禁止设备自身地址 1。不对接收的从地址 OA1 应答。
1：使能设备自身地址 1。对接收的从地址 OA1 应答。

位 14:11 保留，必须保持复位值。

位 10 **OA1MODE**：设备自身地址 1 10 位模式 (Own address 1 10-bit mode)
0：设备自身地址 1 为 7 位地址。
1：设备自身地址 1 为 10 位地址。
注意：仅可在 OA1EN = 0 时写入该位。

位 9:0 **OA1[9:0]**：接口自身从地址 (Interface own slave address)
7 位寻址模式：OA1[7:1] 包含 7 位自身从地址。OA1[9]、OA1[8] 和 OA1[0] 位为无关位。
10 位寻址模式：OA1[9:0] 包含 10 位自身从地址。
注意：仅可在 OA1EN = 0 时写入这些位。

34.7.4 I2C 自身地址 2 寄存器 (I2C_OAR2)

I2C own address 2 register

偏移地址：0x0C

复位值：0x0000 0000

访问：无等待周期，正在执行写访问时出现另一个写访问的情况除外。在这种情况下，会在第二个写访问中插入等待周期，直到前一个写访问完成为止。第二个写访问的延时最长可达 2 x i2c_pclk + 6 x i2c_ker_ck。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA2EN	Res.	Res.	Res.	Res.	OA2MSK[2:0]			OA2[7:1]							Res.
rw					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

位 31:16 保留，必须保持复位值。

位 15 **OA2EN**：设备自身地址 2 使能 (Own address 2 enable)
0：禁止设备自身地址 2。不对接收的从地址 OA2 应答。
1：使能设备自身地址 2。对接收的从地址 OA2 应答。

位 14:11 保留，必须保持复位值。

位 10:8 **OA2MSK[2:0]**：设备自身地址 2 屏蔽位 (Own Address 2 masks)
000：无屏蔽
001：OA2[1] 被屏蔽，为无关位。仅比较 OA2[7:2]。
010：OA2[2:1] 被屏蔽，为无关位。仅比较 OA2[7:3]。
011：OA2[3:1] 被屏蔽，为无关位。仅比较 OA2[7:4]。
100：OA2[4:1] 被屏蔽，为无关位。仅比较 OA2[7:5]。
101：OA2[5:1] 被屏蔽，为无关位。仅比较 OA2[7:6]。
110：OA2[6:1] 被屏蔽，为无关位。仅比较 OA2[7]。
111：OA2[7:1] 被屏蔽，为无关位。不进行比较，对接收到的全部 7 位地址（保留位除外）应答。

注意：仅可在 OA2EN = 0 时写入这些位。
只要 OA2MSK 不等于 0，即使比较匹配，也不会对保留的 I2C 地址（0b0000xxx 和 0b1111xxx）应答。

位 7:1 **OA2[7:1]**：接口地址 (Interface address)
7 位寻址模式：7 位地址
注意：仅可在 OA2EN = 0 时写入这些位。

位 0 保留，必须保持复位值。

34.7.5 I2C 时序寄存器 (I2C_TIMINGR)

I2C timing register

偏移地址：0x10

复位值：0x0000 0000

访问：无等待周期

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

位 31:28 **PRESC[3:0]**：时序预分频因子 (Timing prescaler)

该位域用于对 i2c_ker_ck 进行预分频，以生成用于数据建立和保持计数器（请参见 *I2C 时序*）以及 SCL 高电平和低电平计数器（请参见 *I2C 主模式初始化*）的时钟周期 t_{PRESC}。

$t_{PRESC} = (PRESC + 1) \times t_{I2CCLK}$

位 27:24 保留，必须保持复位值。

位 23:20 **SCLDEL[3:0]**：数据建立时间 (Data setup time)

该位域用于在 SDA 边沿和 SCL 上升沿之间生成延时 t_{SCLDEL}。在主模式和从模式下，如果 NOSTRETCH = 0，则 SCL 线的低电平时间将在 t_{SCLDEL} 期间延长。

$t_{SCLDEL} = (SCLDEL + 1) \times t_{PRESC}$

注意：t_{SCLDEL} 用于生成 $t_{SU:DAT}$ 时序。

位 19:16 **SDADEL[3:0]**：数据保持时间 (Data hold time)

该位域用于在 SCL 下降沿和 SDA 边沿之间生成延时 t_{SDADEL}。在主模式和从模式下，如果 NOSTRETCH = 0，则 SCL 线的低电平时间将在 t_{SDADEL} 期间延长。

$t_{SDADEL} = SDADEL \times t_{PRESC}$

注意：SDADEL 用于生成 $t_{HD:DAT}$ 时序。

位 15:8 **SCLH[7:0]**：SCL 高电平周期（主模式）(SCL high period (master mode))

在主模式下，该位域用于生成 SCL 高电平周期。

$t_{SCLH} = (SCLH + 1) \times t_{PRESC}$

注意：SCLH 还用于生成 $t_{SU:STO}$ 和 tHD:STA 时序。

位 7:0 **SCLL[7:0]**：SCL 低电平周期（主模式）(SCL low period (master mode))

在主模式下，该位域用于生成 SCL 低电平周期。

$t_{SCLL} = (SCLL + 1) \times t_{PRESC}$

注意：SCLL 还用于生成 t_{BUF} 和 $t_{SU:STA}$ 时序。

注意：该寄存器必须在 I2C 禁止时 (PE = 0) 进行配置。

注意：STM32CubeMX 工具计算 I2C_TIMINGR 内容并在 I2C 配置窗口中进行显示。

34.7.6 I2C 超时寄存器 (I2C_TIMEOUTR)

I2C timeout register

偏移地址：0x14

复位值：0x0000 0000

访问：无等待周期，正在执行写访问时出现另一个写访问的情况除外。在这种情况下，会在第二个写访问中插入等待周期，直到前一个写访问完成为止。第二个写访问的延时最长可达 2 x i2c_pclk + 6 x i2c_ker_ck。

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

位 31 **TEXTEN**：时钟信号延长超时使能 (Extended clock timeout enable)

0：禁止时钟信号延长超时检测

1：使能时钟信号延长超时检测当 I2C 接口执行 SCL 延长的累积时间超过 $t_{LOW:EXT}$ 时，将检测到超时错误 (TIMEOUT = 1)。

位 30:28 保留，必须保持复位值。

位 27:16 **TIMEOUTB[11:0]**：总线超时 B (Bus timeout B)

该位域用于配置累积时钟延长超时：

在主模式下，将检测主器件的累积时钟低电平延长时间 ($t_{LOW:MEXT}$)

在从模式下，将检测从器件的累积时钟低电平延长时间 ($t_{LOW:SEXT}$)

$t_{LOW:EXT}$ = (TIMEOUTB + TIDLE = 01) x 2048 x t_{I2CCLK}

注意：仅可在 TEXTEN = 0 时写入这些位。

位 15 **TIMOUTEN**：时钟超时使能 (Clock timeout enable)

0：禁止 SCL 超时检测

1：使能 SCL 超时检测：当 SCL 的低电平时间超过 $t_{TIMEOUT}$ (TIDLE = 0)，或 SCL 的高电平时间超过 t_{IDLE} (TIDLE = 1) 时，将检测到超时错误 (TIMEOUT = 1)。

位 14:13 保留，必须保持复位值。

位 12 **TIDLE**：空闲时钟超时检测 (Idle clock timeout detection)

0：TIMEOUTA 用于检测 SCL 低电平超时

1：TIMEOUTA 用于检测 SCL 和 SDA 高电平超时（总线空闲条件）

注意：仅可在 TIMOUTEN = 0 时写入该位。

位 11:0 **TIMEOUTA[11:0]**：总线超时 A (Bus timeout A)

该位域用于配置：

SCL 低电平超时条件 $t_{TIMEOUT}$（当 TIDLE = 0 时）

$t_{TIMEOUT}$= (TIMEOUTA + 1) x 2048 x t_{I2CCLK}

总线空闲条件，即 SCL 和 SDA 高电平（当 TIDLE = 1 时）

t_{IDLE}= (TIMEOUTA + 1) x 4 x t_{I2CCLK}

注意：仅可在 TIMOUTEN = 0 时写入这些位。

注意：如果不支持 SMBus 功能，该寄存器保留，并由硬件强制为“0x00000000”。请参见第 34.3 节。

34.7.7 I2C 中断和状态寄存器 (I2C_ISR)

I2C interrupt and status register

偏移地址：0x18

复位值：0x0000 0001

访问：无等待周期

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUSY	Res.	ALERT	TIME OUT	PEC ERR	OVR	ARLO	BERR	TCR	TC	STOPF	NACKF	ADDR	RXNE	TXIS	TXE
r		r	r	r	r	r	r	r	r	r	r	r	r	rs	rs

位 31:24 保留，必须保持复位值。

位 23:17 **ADDCODE[6:0]**：地址匹配代码（从模式）(Address match code (Slave mode))

发生地址匹配事件时 (ADDR = 1)，这些位更新为接收到的地址。

在 10 位地址的情况下，ADDCODE 提供 10 位地址的头字节，后跟地址的两个 MSB。

位 16 **DIR**：传输方向（从模式）(Transfer direction (slave mode))

该标志在发生地址匹配事件时 (ADDR = 1) 更新。

0：写传输，从器件进入接收器模式。

1：读传输，从器件进入发送器模式。

位 15 **BUSY**：总线忙碌 (Bus busy)

该标志用于指示总线上正在进行通信。当检测到起始位时，该位由硬件置 1。当检测到停止位或 PE = 0 时，该位由硬件清零。

位 14 保留，必须保持复位值。

位 13 **ALERT**：SMBus 报警 (SMBus alert)

当 SMBHEN = 1（SMBus 主机配置）、ALERTEN = 1 且在 SMBA 引脚上检测到 SMBALERT 事件（下降沿）时，该标志由硬件置 1。该位由软件清零，方法是将 ALERTCF 位置 1。

注意：当 PE = 0 时，该位由硬件清零。

如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 12 **TIMEOUT**：超时或 t_{LOW} 检测标志 (Timeout or t_{LOW} detection flag)

发生超时或延长时钟超时时，该标志由硬件置 1。该位由软件清零，方法是将 TIMEOUTCF 位置 1。

注意：当 PE = 0 时，该位由硬件清零。

如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 11 **PECERR**：接收期间 PEC 错误 (PEC Error in reception)

当接收到的 PEC 与 PEC 寄存器的内容不匹配时，该标志由硬件置 1。接收到错误的 PEC 后，将自动发送 NACK。该标志由软件清零，方法是将 PECCF 位置 1。

注意：当 PE = 0 时，该位由硬件清零。

如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 10 **OVR**：上溢/下溢（从模式）(Overrun/Underrun (slave mode))

在从模式下且 NOSTRETCH = 1 时，如果发生上溢/下溢错误，该标志由硬件置 1。该标志由软件清零，方法是将 OVRCF 位置 1。

注意：当 PE = 0 时，该位由硬件清零。

位 9 **ARLO**：仲裁丢失 (Arbitration lost)

发生仲裁丢失时，该标志由硬件置 1。该标志由软件清零，方法是将 ARLOCF 位置 1。

注意：当 PE = 0 时，该位由硬件清零。

位 8 **BERR**：总线错误 (Bus error)

当检测到错位的起始位或停止位，而外设也参与传输时，该标志由硬件置 1。在从模式下的地址阶段，该标志不会置 1。该标志由软件清零，方法是将 *BERRCF 位置 1*。

注意：当 PE = 0 时，该位由硬件清零。

位 7 **TCR**：传输完成等待重载 (Transfer Complete Reload)

当 RELOAD = 1 且 NBYTES 数据传输完成时，该标志由硬件置 1。当 NBYTES 写入一个非零值时，该标志由软件清零。

注意：当 PE = 0 时，该位由硬件清零。

该标志单独用于主模式，SBC 位置 1 时单独用于从模式。

位 6 **TC**：传输完成（主模式）(Transfer Complete (master mode))

当 RELOAD = 0、AUTOEND = 0 且 NBYTES 数据传输完成时，该标志由硬件置 1。当 START 位或 STOP 位置 1 时，该标志由软件清零。

注意：当 PE = 0 时，该位由硬件清零。

位 5 **STOPF**：停止位检测标志 (Stop detection flag)

当在总线上检测到停止位，且外设也参与本次传输时，该标志由硬件置 1：

- 外设作为主器件，该位置 1 的前提是外设已经发出停止位。
- 外设作为从器件，该位置位的前提条件是此次传输的寻址对象就是该外设。

该标志由软件清零，方法是将 STOPCF 位置 1。

注意：当 PE = 0 时，该位由硬件清零。

位 4 **NACKF**：接收到否定应答标志 (Not Acknowledge received flag)

传输完字节后接收到 NACK 时，该标志由硬件置 1。该标志由软件清零，方法是将 NACKCF 位置 1。

注意：当 PE = 0 时，该位由硬件清零。

位 3 **ADDR**：地址匹配（从模式）(Address matched (slave mode))

接收到的地址与使能的从设备地址之一匹配时，该位由硬件置 1。该位由软件清零，方法是将 *ADDRCF 位置 1*。

注意：当 PE = 0 时，该位由硬件清零。

位 2 **RXNE**：接收数据寄存器不为空（接收器）(Receive data register not empty (receivers))

当接收到的数据已复制到 I2C_RXDR 寄存器且准备就绪可供读取时，该位由硬件置 1。读取 I2C_RXDR 时，将清零该位。

注意：当 PE = 0 时，该位由硬件清零。

位 1 **TXIS**：发送中断状态（发送器）(Transmit interrupt status (transmitters))

当 I2C_TXDR 寄存器为空时，该位由硬件置 1，待发送的数据必须写入 I2C_TXDR 寄存器。下一个待发送的数据写入 I2C_TXDR 寄存器时，该位被清零。

该位只能在 NOSTRETCH = 1 时由软件写入 1，以生成 TXIS 事件（TXIE = 1 时为中断，TXDMAEN = 1 时为 DMA 请求）。

注意：当 PE = 0 时，该位由硬件清零。

位 0 **TXE**：发送数据寄存器为空（发送器）(Transmit data register empty (transmitters))
当 I2C_TXDR 寄存器为空时，该位由硬件置 1。下一个待发送的数据写入 I2C_TXDR 寄存器时，该位被清零。
该位可由软件写入 1，以刷新发送数据寄存器 I2C_TXDR。
注意：当 PE = 0 时，该位由硬件置 1。

34.7.8 I2C 中断清零寄存器 (I2C_ICR)

I2C interrupt clear register

偏移地址：0x1C

复位值：0x0000 0000

访问：无等待周期

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	ALERTCF	TIMOUTCF	PECCF	OVRCF	ARLOCF	BERRCF	Res.	Res.	STOPCF	NACKCF	ADDRCF	Res.	Res.	Res.
		w	w	w	w	w	w			w	w	w			

位 31:14 保留，必须保持复位值。

位 13 **ALERTCF**：报警标志清零 (Alert flag clear)
将 1 写入该位时，I2C_ISR 寄存器中的 ALERT 标志将清零。
注意：如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 12 **TIMOUTCF**：超时检测标志清零 (Timeout detection flag clear)
将 1 写入该位时，I2C_ISR 寄存器中的 TIMEOUT 标志将清零。
注意：如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 11 **PECCF**：PEC 错误标志清零 (PEC Error flag clear)
将 1 写入该位时，I2C_ISR 寄存器中的 PECERR 标志将清零。
注意：如果不支持 SMBus 功能，该位保留并由硬件强制清零。请参见第 34.3 节。

位 10 **OVRCF**：上溢/下溢标志清零 (Overrun/Underrun flag clear)
将 1 写入该位时，I2C_ISR 寄存器中的 OVR 标志将清零。

位 9 **ARLOCF**：仲裁丢失标志清零 (Arbitration lost flag clear)
将 1 写入该位时，I2C_ISR 寄存器中的 ARLO 标志将清零。

位 8 **BERRCF**：总线错误标志清零 (Bus error flag clear)
将 1 写入该位时，I2C_ISR 寄存器中的 BERRF 标志将清零。

位 7:6 保留，必须保持复位值。

位 5 **STOPCF**：停止位检测标志清零 (STOP detection flag clear)
将 1 写入该位时，I2C_ISR 寄存器中的 STOPF 标志将清零。

位 4 **NACKCF**：否定应答标志清零 (Not Acknowledge flag clear)
向该位写入 1 时，I2C_ISR 寄存器中的 NACKF 标志将清零。

位 3 **ADDRCF**：地址匹配标志清零 (Address Matched flag clear)
将 1 写入该位时，I2C_ISR 寄存器中的 ADDR 标志将清零。将 1 写入该位时，I2C_CR2 寄存器中的 START 位也将清零。

位 2:0 保留，必须保持复位值。

34.7.9 I2C PEC 寄存器 (I2C_PECR)

I2C PEC register

偏移地址：0x20

复位值：0x0000 0000

访问：无等待周期

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

位 31:8 保留，必须保持复位值。

位 7:0 **PEC[7:0]**：数据包错误校验寄存器 (Packet error checking register)
当 PECEN=1 时，该位域包含内部 PEC。
当 PE = 0 时，PEC 由硬件清零。

注意： *如果不支持 SMBus 功能，该寄存器保留，并由硬件强制为“0x00000000”。请参见第 34.3 节。*

34.7.10 I2C 接收数据寄存器 (I2C_RXDR)

I2C receive data register

偏移地址：0x24

复位值：0x0000 0000

访问：无等待周期

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **RXDATA[7:0]**：8 位接收数据 (8-bit receive data)
从 I^2C 总线接收的数据字节。

34.7.11 I2C 发送数据寄存器 (I2C_TXDR)

I2C transmit data register

偏移地址：0x28

复位值：0x0000 0000

访问：无等待周期

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **TXDATA[7:0]**：8 位发送数据 (8-bit transmit data)
待发送到 I^2C 总线的数据字节
注意：仅可在 TXE = 1 时写入这些位。

34.7.12 I2C 寄存器映射

下表提供了 I2C 寄存器映射和复位值。

表 267. I2C 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0	I2C_CR1	STOPFACLR	ADDRACLR	Res.	Res.	Res.	Res.	Res.	FMP	PECEN	ALERTEN	SMBDEN	SMBHEN	GCEN	WUPEN	NOSTRETCH	SBC	RXDMAEN	TXDMAEN	Res.	ANFOFF	DNF[3:0]				ERRIE	TCIE	STOPIE	NACKIE	ADDRIE	RXIE	TXIE	PE
	Reset value	0	0						0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0
0x4	I2C_CR2	Res.	Res.	Res.	Res.	Res.	PECBYTE	AUTOEND	RELOAD	NBYTES[7:0]								NACK	STOP	START	HEAD10R	ADD10	RD_WRN	SADD[9:0]									
	Reset value						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x8	I2C_OAR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OA1EN	Res.	Res.	Res.	Res.	OA1MODE	OA1[9:0]									
	Reset value																	0					0	0	0	0	0	0	0	0	0	0	0
0xC	I2C_OAR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OA2EN	Res.	Res.	Res.	Res.	OA2MSK [2:0]	OA2[7:1]									Res.
	Reset value																	0					0	0	0	0	0	0	0	0	0	0	
0x10	I2C_ TIMINGR	PRESC[3:0]				Res.	Res.	Res.	Res.	SCLDEL [3:0]				SDADEL [3:0]				SCLH[7:0]								SCLL[7:0]							
	Reset value	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	I2C_ TIMEOUTR	TEXTEN	Res.	Res.	Res.	TIMEOUTB[11:0]												TIMOUTEN	Res.	Res.	TIDLE	TIMEOUTA[11:0]											
	Reset value	0				0	0	0	0	0	0	0	0	0	0	0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	I2C_ISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADDCODE[6:0]							DIR	BUSY	Res.	ALERT	TIMEOUT	PECERR	OVR	ARLO	BERR	TCR	TC	STOPF	NACKF	ADDR	RXNE	TXIS	TXE
	Reset value									0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x1C	I2C_ICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ALERTCF	TIMOUTCF	PECCF	OVRCF	ARLOCF	BERRCF	Res.	Res.	STOPCF	NACKCF	ADDRCF	Res.	Res.	Res.
	Reset value																			0	0	0	0	0	0			0	0	0			
0x20	I2C_PECR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PEC[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x24	I2C_RXDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RXDATA[7:0]							
	Reset value																									0	0	0	0	0	0	0	0

表 267. I2C 寄存器映射和复位值 （续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x28	I2C_TXDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXDATA[7:0]							
	Reset value																									0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

35 改进型内部集成电路 (I3C)

35.1 简介

I3C 接口负责处理此设备与连接在 I3C 总线上的其他设备（例如传感器和主机处理器）之间的通信。

I3C 总线是一种双线制串行单端多分支总线，旨在改进传统的 I^2C 总线。

I3C 仅 SDR 外设实现了 MIPI® I3C 规范 v1.1 要求的所有功能。它可以控制所有 I3C 总线特定的序列、协议、仲裁和时序，并且既可以作为控制器（以前称为主设备）也可以作为目标（以前称为从设备）。

当作为控制器时，I3C 外设能够改善 I^2C 接口功能，同时保持一定程度的向后兼容性：即支持 I^2C 目标以传统 I^2C 快速模式 (Fm) 或传统 I^2C 超快速模式 (Fm+) 在 I3C 总线上工作，但前提是目标不执行时钟延长。

I3C 外设可与 DMA 一起使用，以减轻 CPU 的负担。

35.2 I3C 主要特性

I3C 外设支持：

- MIPI® I3C 规范 v1.1（有关详细信息，请参见*表 271*），可用作：
  - I3C 仅 SDR 主控制器
  - I3C 仅 SDR 辅助控制器
  - I3C 仅 SDR 目标
- I3C SCL 总线时钟频率高达 12.5 MHz
- 主机应用通过 APB 从端口配置寄存器
- 排队传输数据：
  - 发送 FIFO (TX-FIFO) 用于在 I3C 总线上发送的数据字节/字
  - 接收 FIFO (RX-FIFO) 用于在 I3C 总线上接收的数据字节/字
  - 每个 FIFO 都可以选择具有专用 DMA 通道的 DMA 模式
- 排队传输控制/状态（作为控制器时）：
  - 控制 FIFO (C-FIFO) 用于在 I3C 总线上发送的控制字
  - 可选状态 FIFO (S-FIFO) 用于在 I3C 总线上接收的状态字
  - 每个 FIFO 都可以选择具有专用 DMA 通道的 DMA 模式
- 消息：
  - 以 Fm/Fm+ 模式发送到传统 I^2C 目标的传统 I^2C 读/写消息
  - I3C SDR 私有读/写消息
  - I3C SDR 广播 CCC 消息（有关详细信息，请参见*表 277*）
  - I3C SDR 直接读/写 CCC 消息（有关详细信息，请参见*表 277*）
- 帧级管理（作为控制器时）：
  - 可选 C-FIFO 和 TX-FIFO 预装载
  - 多消息封装
  - 可选择在 I3C 总线上生成可仲裁头
  - 在 I3C 总线上生成 HDR 退出时序，以从错误中恢复

- 可编程总线时序（作为控制器时）：
  - SCL 高电平和低电平周期
  - SDA 保持时间
  - 总线空闲（最短）时间
  - 总线可用/空闲条件时间
  - 时钟停顿时间
- 管理目标发起的请求：
  - 最多可同时管理四个目标（作为控制器时）
  - 带内中断，可编程 IBI 有效负载（最多 4 个字节），支持挂起读通知
  - 总线控制请求，支持恢复流程，存在切换延迟
  - 热加入机制
- 检测 HDR 退出时序（作为目标时）
- 总线错误管理：
  - CEx，其中 x = 0、1、2、3（作为控制器时）
  - TEx，其中 x = 0、1、...、6（作为目标时）
  - 总线控制切换错误与恢复
  - 目标复位
- 基于事件的独立可编程管理：
  - 通过标志报告和清零控制识别每个事件
  - 通过标志轮询和/或通过可编程中断（可按事件使能）通知主机应用
  - 错误类型识别
- 从停止模式唤醒（作为控制器时）（请参见*第 35.3.2 节*）：
  - 通过带内中断（无有效负载）
  - 通过热加入请求
  - 通过控制器角色请求
- 从停止模式唤醒（作为目标时）（请参见*第 35.3.2 节*）：
  - 通过复位时序
  - 通过丢失的起始位
- 多时钟域管理：
  - 除了 SCL 时钟之外，还支持独立的 APB 时钟和内核时钟，它们由独立编程的时钟源通过 RCC 驱动
  - 对于应用驱动的不同 SCL 时钟频率所需要的内核时钟和 APB 时钟的最低工作频率（请参见*第 35.6.2 节*中的时钟限制）

35.3 I3C 特性实现

35.3.1 I3C 实例化

设备中有一个 I3C 实例。

35.3.2 I3C 从低功耗模式唤醒

外设可以将器件从低功耗模式唤醒，详细信息请参见*表 268*。有关唤醒功能的更多详细信息，请参见*第 35.13 节*。

表 268. I3C 唤醒

唤醒
通过 SVOS3 从停止模式唤醒

35.3.3 I3C FIFO

FIFO 按照*表 269* 中的定义实现。

表 269. I3C FIFO 实现

FIFO	内容	单位	大小	用作控制器/目标（基本原理）
C-FIFO	（32 位）控制字	字	2	控制器（一个帧可基于多个控制字；而用作目标时则不然）
S-FIFO	（32 位）状态字			控制器（目标：仅限寄存器模式下的状态）
TX-FIFO	发送的数据	字节	8	控制器和目标
RX-FIFO	接收的数据			

35.3.4 I3C 触发信号

本产品不具有该特性：没有任何硬件触发信号作为输入连接到 I3C 外设。

35.3.5 I3C 中断

有关中断映射实现的详细信息，请参见*表 270*。

表 270. I3C 中断

信号名称	信号类型	说明
i3c_err_it	O	错误中断线
i3c_evt_it	O	事件中断线

35.3.6 I3C MIPI® 支持

I3C 外设支持 MIPI 规范 v1.1，如*表 271* 所定义。

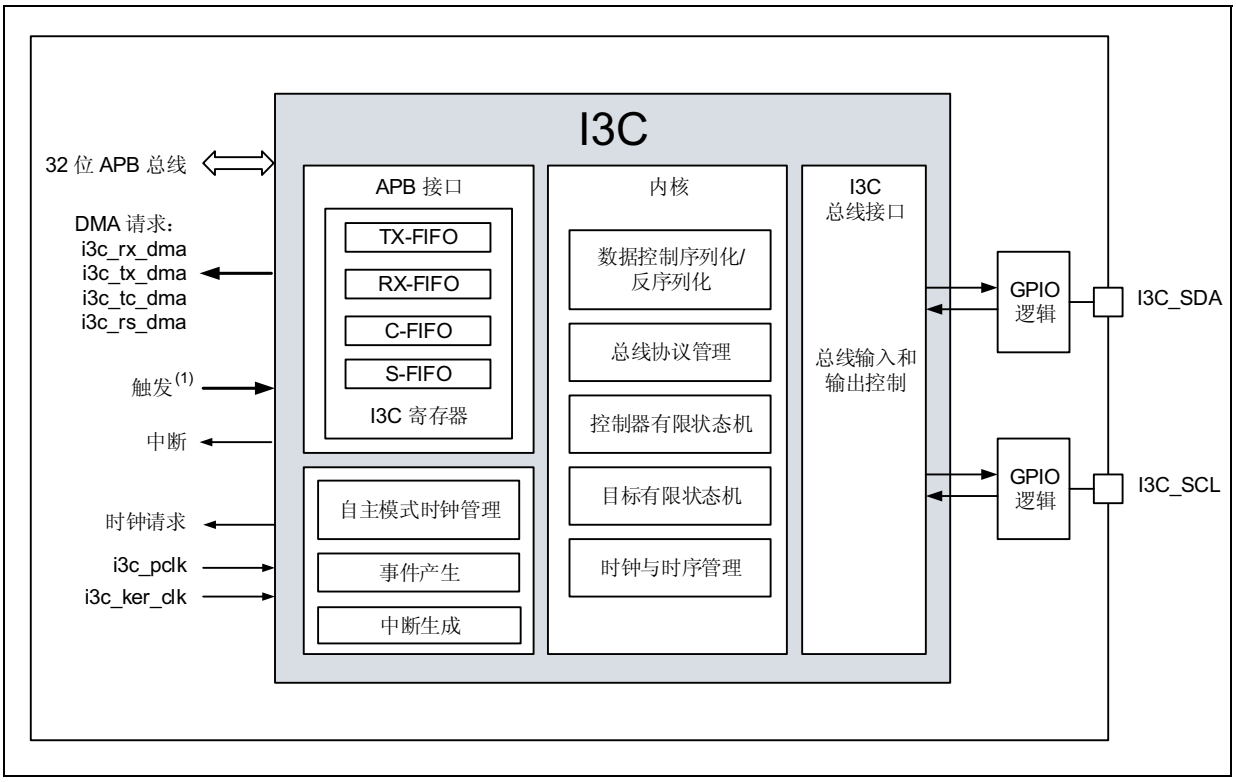
表 271. I3C 外设控制器/目标特性与 MIPI v1.1 的关系

特性	MIPI I3C v1.1	I3C 外设		备注
		作为控制器时	作为目标时	
I3C SDR 消息	X	X	X	-
传统 I^2C 消息 (Fm/Fm+)	X	X	-	作为控制器并且 I3C 总线与（外部）传统 I2C 目标结合使用时必须具备。 作为目标时，在 MIPI v1.1 中可选。
HDR DDR 消息	X	-	-	在 MIPI v1.1 中可选
HDR-TSL/TSP、HDR-BT	X	-	-	在 MIPI v1.1 中可选
动态地址分配	X	X	X	-
静态地址	X	X	-	不（有意）支持外设作为 I^2C 总线上的目标。
分组寻址	X	X	-	在 MIPI v1.1 中可选
CCC	X	X	X	支持必须的以及一些可选的 CCC 命令（作为控制器/目标时，请参见*表 277*）。
错误检测和恢复	X	X	X	-
带内中断（通过 MDB）	X	X	X	-
辅助控制器	X	X	X	-
热加入机制	X	X	X	-
目标复位	X	X	X	-
同步时序控制	X	X	-	在 MIPI v1.1 中可选
异步时序控制 0	X	X	-	作为控制器时，在 MIPI v1.1 中要求必须具备。 作为目标时，在 MIPI v1.1 中可选。
异步时序控制 1、2、3	X	-	-	在 MIPI v1.1 中可选
设备到设备隧道	X	X	-	在 MIPI v1.1 中可选
多线数据传输	X	-	-	在 MIPI v1.1 中可选
监视设备提前终止	X	-	-	在 MIPI v1.1 中可选

35.4 I3C 框图

I3C 框图如*图 402* 所示。

图 402. I3C 框图



1. 该特性取决于具体实现，有可能不可用。请参见*第 35.3.4 节：I3C 触发信号*。

35.5 I3C 引脚和内部信号

表 272. I3C 输入/输出引脚

引脚名称	信号类型	说明
I3C_SDA	双向	I3C 总线串行数据线
I3C_SCL	双向	I3C 总线串行时钟线

表 273. I3C 内部输入/输出信号

信号名称	信号类型	说明
i3c_pclk	I	APB 时钟
i3c_ker_clk	I	内核时钟（也称为 I3CCLK）
i3c_pclk_req	O	APB 时钟请求
i3c_ker_clk_req	O	内核时钟请求
i3c_it(1)	O	全局中断线

表 273. I3C 内部输入/输出信号（续）

信号名称	信号类型	说明
i3c_err_it(1)	O	错误中断线
i3c_evt_it(1)	O	事件中断线
i3c_rx_dma	O	从 RX-FIFO 读取收到的字节/字的 DMA 请求。
i3c_tx_dma	O	将待发送的字节/字写入 TX-FIFO 的 DMA 请求。
i3c_tc_dma	O	将待发送的控制字写入 C-FIFO 的 DMA 请求（外设用作控制器时）。
i3c_rs_dma	O	从 S-FIFO 读取状态字的 DMA 请求（外设用作控制器时）。

1. 该信号取决于具体实现。请参见*第 35.3.5 节*。

35.6 I3C 复位和时钟

35.6.1 I3C 复位

系统复位时，I3C 外设将复位。

或者，也可以通过软件写入复位和时钟控制器 (RCC) 的相应复位控制位 (I3CxRST) 来专门复位外设。更多详细信息，请参见本文档的 RCC 部分。

此外，用作目标的外设在使能（I3C_CFGR 寄存器中的 EN = 1）后可以在 I3C 总线上接收控制器发出的带内复位时序。之后，通知软件（当 I3C_EVR 寄存器中的 RSTF = 1 和/或使能相应的中断时）执行在上一次接收到广播或直接 RSTACT CCC 时请求的操作（如 I3C_DEVR0 寄存器中的 RSTACT[1:0] 所设定）。更多详细信息，请参见*表 277* 和*第 35.16.16 节*。

该复位中断通知可用于从 I3C 外设（通常在 V_{CORE} 域中）处于激活状态的低功耗模式下唤醒。

有关相应低功耗模式的更多详细信息，请参见 PWR 部分中的电源管理。

35.6.2 I3C 时钟和要求

如*图 402* 中所示，I3C 外设实现了以下几个时钟域：

- SCL 总线时钟：用于 I3C 总线接口
  - 作为控制器时：用户必须通过编程 *I3C 时序寄存器 0 (I3C_TIMINGR0)*、*I3C 时序寄存器 1 (I3C_TIMINGR1)* 和 *I3C 时序寄存器 2 (I3C_TIMINGR2)* 设置（并可调整）SCL/SDA 时序，相关信息在*控制器初始化*和*更新传输配置（作为控制器）*中进行了总结。
  - 作为目标时：用户必须通过编程 *I3C 时序寄存器 1 (I3C_TIMINGR1)* 设置并遵守总线可用条件（t_{AVAL} 针对带内中断或控制器角色请求）和总线空闲条件（t_{IDLE} 针对热加入请求），相关信息在*目标初始化*中进行了总结。
- I3CCLK 内核时钟：用于 I3C 协议管理、数据和控制序列化/反序列化、控制器和目标有限状态机、总线时钟和时序管理
- APB 时钟：用于 APB 接口、DMA 接口、事件和中断生成

APB 时钟和内核时钟由独立编程的时钟源通过 RCC 驱动（请参见*第 10 章：复位和时钟控制 (RCC)*）。

I3C 内核时钟要求（作为控制器）

根据总线上 SCL 时钟的预期值，应用必须确保 I3CCLK 内核时钟频率至少是 SCL 时钟频率的 2 倍

注意： *保持 $F_{SCL\ max}$ = 12.9 MHz 意味着 I3CCLK 内核时钟频率 > 25.8 MHz。*

I3C 内核时钟要求（作为目标）

根据总线上 SCL 时钟的预期值，应用必须确保 I3CCLK 内核时钟的最低工作频率满足下列限制条件：

1. I3CCLK 内核时钟周期 < t_{HIGH}（SCL 时钟高电平周期）
   - $t_{HIGH\ min}$ = 24 ns。高于 41.7 MHz 的频率能够确保符合该限制，该限制也可以适当放宽，具体取决于 I3C 总线/控制器
2. I3CCLK 内核时钟周期 < t_{CASr}（重复起始位后的时钟保持时间）
   - $t_{CASr\ min}$ = $t_{CAS\ min}$ / 2 = 19.2 ns。高于 52 MHz 的频率能够确保符合该限制，该限制也可以适当放宽，具体取决于 I3C 总线/控制器
3. 两个 I3CCLK 内核时钟周期 < t_{LOW_OD}（开漏模式下的 SCL 时钟低电平周期）
   - $t_{LOW_OD\ min}$ = 200 ns。高于 10 MHz 的频率能够确保符合该限制，该限制也可以适当放宽，具体取决于 I3C 总线/控制器
4. I3CCLK 内核时钟频率 > 2.5 倍 SCL 时钟频率
   - $F_{SCL\ max}$ = 12.9 MHz。高于 32.3 MHz 的频率能够确保符合该限制，该限制也可以适当放宽，具体取决于 I3C 控制器

APB 时钟要求

根据总线上 SCL 时钟的预期值，应用必须确保 APB 时钟的最低工作频率：

APB 时钟周期 < 3 x SCL 时钟周期 - I3CCLK 内核时钟周期

这意味着 $F_{APB} > [F_{SCL} \times F_{I3CCLK} / (3 \times F_{I3CCLK} - F_{SCL})]$

注意： *对于 APB 频率，该公式可以简化为最小值 5 MHz。*

35.7 I3C 外设状态和编程

35.7.1 I3C 外设状态

I3C 外设既可以用作 I3C 总线控制器，也可以用作 I3C 目标。在任何情况下（请参见*图 403* 和*图 404*），外设都处于以下状态之一：

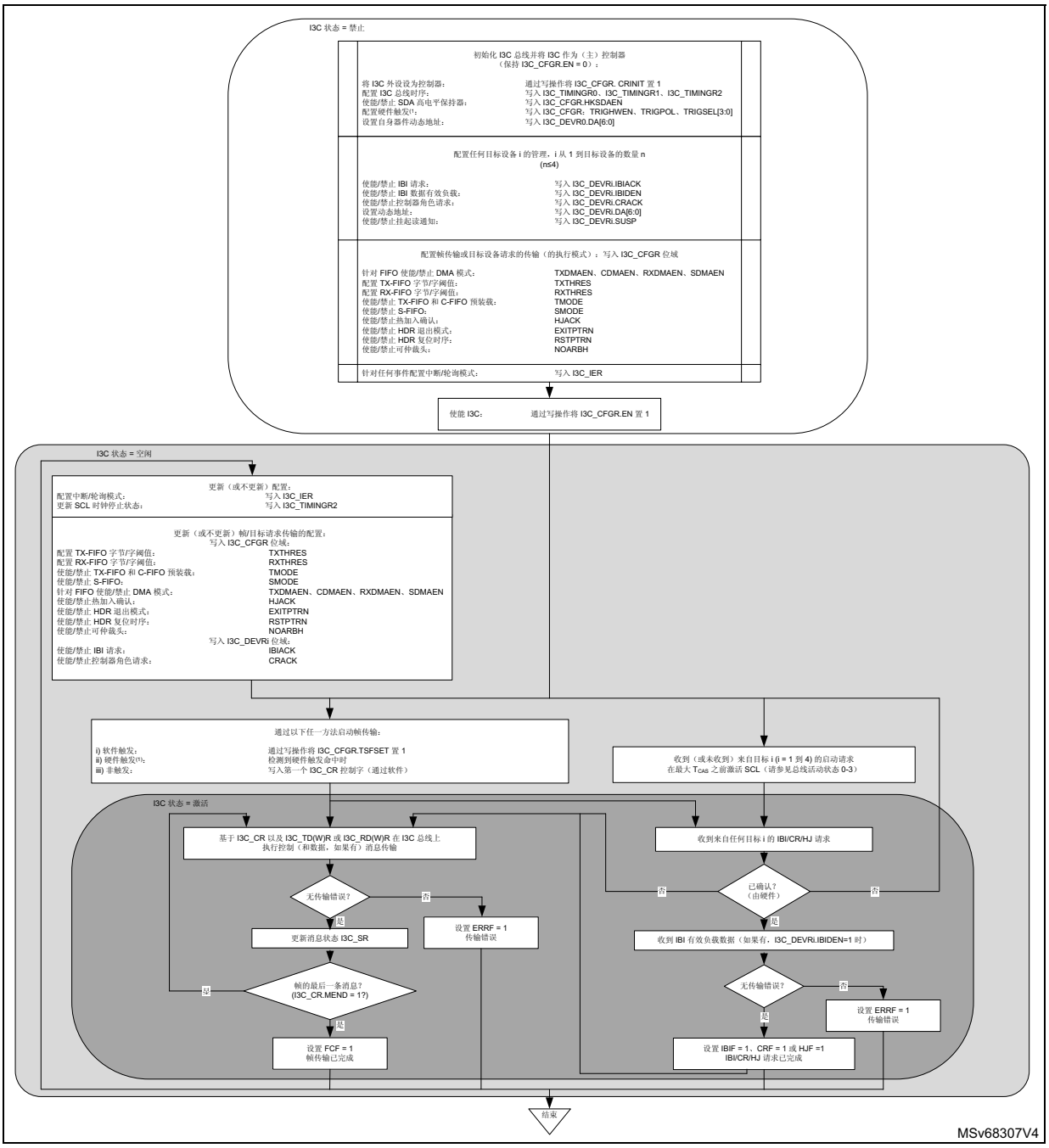
- 禁止状态：
  - I3C 复位（系统复位或通过 RCC 进行 I3C 复位）后，外设处于禁止状态。
  - 当软件将 I3C_CFGR 寄存器中的 EN 位置 1 时，外设会考虑不同配置寄存器的值，然后切换到（使能且）空闲状态。
- 空闲状态：
  - 外设在使能 (EN = 1) 后会激活其 I3CCLK 和 SCL 时钟域，并能够在 I3C 总线上进行通信。
  - 软件可以对 I3C 外设配置进行部分更新，更多详细信息请参见*更新传输配置（作为控制器）*和*更新 I3C 外设的配置（作为目标）*。
  - 在以下两种情况下，外设会切换到（使能且）工作状态：
  a) 软件启动传输后（作为控制器：当软件启动帧传输时；作为目标：当软件发出 IBI/CR/HJ 请求时）
  b) 硬件接收到来自总线上另一个 I3C 设备的请求后（作为控制器：在接收到来自目标的启动请求和最大 T_{CAS} 时间之后；作为目标：在接收到广播/直接 CCC 或私有读/写时）。
- 工作状态：
  - 外设在总线上执行传输
  - 当请求的传输完成时，将通过 *I3C 事件寄存器 (I3C_EVR)* 中的事件通知软件，并通过 *I3C 中断使能寄存器 (I3C_IER)* 使能相应的中断。外设切换到空闲状态后，仍可在总线上进行通信，并且可以进行（部分）重新配置。
  a) 作为控制器：生成的事件/标志可以是帧完成 (FCF)、IBI/控制器角色/热加入请求完成 (IBIF/CRF/HJF) 或传输错误 (ERRF)
  b) 作为目标：生成的事件/标志可以是动态地址分配完成 (DAUPDF)、IBI 完成 (IBIENDF)、控制器角色获得完成 (CRUPDF)、广播/直接 CCC 完成 (xxUPDF/RSTF/GETF/STAF)、私有读/写完成 (FCF) 或传输错误 (ERRF)

注意： *软件可以禁止外设（写入 EN = 0），将其进行部分复位（SCL 时钟域和 I3CCLK 内核时钟域内的部分）。事件、中断和时钟请求的生成也会受到影响。先前写入的 APB 寄存器配置将会保留，不会被修改。*

35.7.2 I3C 控制器状态和编程序列

图 403 给出了 I3C 外设用作（主）控制器时的完整编程序列，包括状态转换、主要子任务和条件，如本节中所述。

图 403. I3C（主）控制器状态和编程序列图



1. 该特性取决于具体实现，有可能不可用。请参见*第 35.3.4 节：I3C 触发信号*。

控制器初始化

当控制器处于禁止状态（I3C_CFGR 寄存器中的 EN = 0）时，软件必须按如下所述进行初始化：

- 配置 *I3C 配置寄存器 (I3C_CFGR)* 中的以下位域：
  - CRINIT = 1：作为 I3C 总线控制器
  - HKSDAEN：SDA 上的高电平保持器使能/禁止
- 配置 I3C 总线时序：
  a) *I3C 时序寄存器 0 (I3C_TIMINGR0)*：
  - 传统 I^2C 和 I3C 开漏/推挽阶段的 SCL 时钟高电平时间周期（t_{DIG_H}、$t_{DIG_H_MIXED}$）
  - 传统 I^2C 和 I3C 开漏/推挽阶段的 SCL 时钟低电平时间周期（t_{DIG_L}、$t_{DIG_OD_L}$）

  b) *I3C 时序寄存器 1 (I3C_TIMINGR1)*：
  - 推挽阶段的 SDA 保持时间 (t_{HD_PP}）
  - 总线空闲条件时间（I3C t_{CAS}、传统 I^2C t_{BUF}）
  - I3C 重复起始位时序（t_{CASr}、t_{CBSr}）
  - I3C 停止时序 (t_{CBP})
  - SCL 时钟低电平最大停顿时间（针对 ENTDAA CCC ($t_{STALLDAA}$)、传统 I^2C 的 ACK/NACK、I3C 传输的地址阶段、写数据传输的奇偶校验位、传统 I^2C 写传输的 ACK/NACK 数据阶段、I3C 读传输的转换位或传统 I^2C 写传输的 ACK/NACK 阶段 (t_{STALL})），可供用作控制器的外设根据自身需要调整 SCL 时钟低电平停顿时间
  - $t_{NEWCRLock}$，针对控制器角色切换过程（在 GETACCCR CCC 命令后）

  c) *I3C 时序寄存器 2 (I3C_TIMINGR2)*：
  - SCL 时钟低电平停顿时间（每个阶段均可单独使能/禁止），用于调整 SCL 时钟低电平停顿时间（如果被寻址的 I3C 目标或传统 I^2C 目标的 SDA 切换过程需要）
- 配置自身动态地址：*I3C 自身器件特性寄存器 (I3C_DEVR0)* 的 DA[6:0] 位域
- 配置任意目标设备 x 的相关管理：*I3C 设备 x 特性寄存器 (I3C_DEVRx)*（x = 1 到 4）
- 配置帧传输或目标请求的传输的执行模式：*I3C 配置寄存器 (I3C_CFGR)* 中的以下位域：
  - TXDMAEN、CDMAEN、RXDMAEN、SDMAEN：分别用于为 TX-FIFO、C-FIFO、RX-FIFO、S-FIFO 使能/禁止 DMA 模式
  - TXTHRES、RXTHRES：分别为 TX-FIFO 和 RX-FIFO 的字节/字阈值
  - TMODE：发送模式（使能/禁止 TX-FIFO 和 C-FIFO 预装载）
  - SMODE：S-FIFO 使能/禁止
  - EXITPTRN、RSTPTRN：退出、复位时序使能/禁止
  - HJACK：热加入确认使能/禁止
  - NOARBH：可仲裁头禁止/使能
- 配置任意事件的中断生成或轮询模式：*I3C 中断使能寄存器 (I3C_IER)*

之后，软件即可使能 I3C 外设（将 EN 置 1）

注意： *软件可以在使能该外设的同时一次写入 I3C_CFGR 的所有位域。*

启动控制器发起的帧传输

当控制器处于使能状态（I3C_CFGR 寄存器中 EN = 1）时，软件可以通过以下任意配置方法启动帧传输：

1. 软件触发：通过写操作将 I3C_CFGR 寄存器中的 TSFSET 置 1 时
   – 这会导致硬件将 I3C_EVR 寄存器中的 CFNFF 标志置 1，以请求写入第一个控制字 I3C_CR。
2. 无触发：通过软件写入第一个控制字 I3C_CR 时。

无论帧采用哪种启动方法，I3C 外设随后都会切换到工作状态。当控制字不是 I3C 帧的的最后一条消息时（当 I3C_CR 寄存器中的 MEND = 0 时），并且没有发生任何传输错误时（当 I3C_EVR 寄存器中的 ERRF = 1 时），硬件会继续请求下一个控制字，并继续进行帧传输：

a) 如果 C-FIFO 未配置为 DMA 模式（I3C_CFGR 寄存器中的 CDMAEN = 0），则软件将在 I3C_EVR 寄存器中的标志 CFNFF = 1 或相应的中断（若已使能，即 I3C_IER 寄存器中的 CFNFIE = 1）后写入下一个控制字

b) 如果 C-FIFO 配置为 DMA 模式 (CDMAEN = 1)，与从而产生的 I3C DMA 请求 (i3c_tc_dma) 对应分配的 DMA 通道将自动推送并写入下一个控制字。

启动和接收目标发起的传输

当控制器处于使能状态（I3C_CFGR 寄存器中的 EN = 1）时，不仅控制器可以启动控制器发起的传输，目标也可以通过发出启动请求（将 SDA 驱动为低电平）来启动传输，但前提是控制器已通过 I3C_DEVR0 寄存器允许热加入请求、IBI 请求或控制器角色请求。

在这种情况下，即使控制器软件无意启动帧传输，硬件也会切换到工作状态（在最大 t_{CAS} 时间（可定义为 1 µs、100 µs、2 ms 或 50 ms，分别对应于总线工作状态 0、1、2 或 3）之前激活 SCL 时钟）以接收来自目标的热加入/带内中断/控制器角色请求。

有关目标发起的 I3C 总线传输的执行以及作为控制器的相关编程的更多信息，请参见*第 35.9 节*中的以下各图：

- *图 415：IBI 传输（作为控制器/目标）*
- *图 416：热加入请求传输（作为控制器/目标）*
- *图 417：控制器角色请求传输（作为控制器/目标）*

执行（控制器发起的）帧传输

控制器在总线上执行帧传输，直到最后一条消息传输完成（I3C_EVR 寄存器中的 FCF = 1），或产生传输错误（I3C_EVR 寄存器中的 ERRF = 1）和相应的中断（如果使能）。这基于 I3C_CR 和 I3C_TD(W)R 寄存器（由软件显式写入或由分配的 DMA 通道推入）以及 I3C_RD(W)R（由软件或由分配的 DMA 通道显式读取）。之后，I3C 控制器切换回空闲状态。

有关控制器发起的 I3C 总线传输的执行以及作为控制器的相关编程的更多信息，请参见*第 35.9 节*中的以下各图：

- *图 405：I3C CCC 消息（作为控制器）*
- *图 406：I3C 广播 ENTDAA CCC（作为控制器）*
- *图 407：I3C 广播、直接读/写 RSTACT CCC（作为控制器）*
- *图 412：I3C 私有读/写消息（作为控制器）*
- *图 414：传统 I2C 读/写消息（作为控制器）*

图 403 不包括 FIFO（TX-FIFO、RX-FIFO、C-FIFO 和 S-FIFO）的管理。有关这部分内容的详细信息，请参见*第 35.10 节*。

当 S-FIFO 被禁止（I3C_CFGR 寄存器中的 SMODE = 0）时，对于每一条无传输错误完成的消息，硬件都会通过更新 *I3C 状态寄存器 (I3C_SR)* 来报告 I3C 总线上完成了一次交换消息的传输，软件可以读取也可以不读取该寄存器。

- 在状态寄存器 I3C_SR 的 S-FIFO 被禁止（I3C_CFGR 寄存器中的 SMODE = 0）的条件下，除了最后一条消息传输完成（I3C_EVR 寄存器中的 FCF = 1）或产生传输错误（I3C_EVR 寄存器中的 ERRF = 1）且产生相应的中断（如果使能）时可通知软件之外，在执行直接 CCC 读或私有读传输时，如果目标提前结束读传输，也会通过 I3C_EVR 寄存器中的 RXTGTENDF = 1 和产生相应的中断（如果使能）来通知软件。然后，软件可以读取 I3C_SR，以获取有关已执行传输的更多信息。

或者，如果使能了 S-FIFO（I3C_CFGR 寄存器中的 SMODE = 1），则无论目标是否提前结束读传输，每执行一条消息都必须读取一次状态寄存器 I3C_SR，要么直接由软件读取（由 I3C_EVR 寄存器中的 SFNEF = 1 和相应的中断（如果使能）通知），要么通过 DMA 读取（如果 I3C_CFGR 寄存器中的 SDMAEN = 1）。只有在读取最后一条消息的状态（S-FIFO 为空）后才会报告帧完成（I3C_EVR 寄存器中的 FCF = 1）。有关详细信息，请参见*第 35.10.4 节*。

更新传输配置（作为控制器）

返回到空闲状态后，软件可以在下一次传输前更新 I3C 外设的配置：

- 通过 *I3C 时序寄存器 2 (I3C_TIMINGR2)* 修改 SCL 时钟停顿
- 通过 *I3C 中断使能寄存器 (I3C_IER)* 修改中断/轮询模式策略
- 修改 *I3C 配置寄存器 (I3C_CFGR)* 的以下位域：
  - TXTHRES、RXTHRES
  - TMODE、SMODE
  - TXDMAEN、CDMAEN、RXDMAEN、SDMAEN
  - EXITPTRN、RSTPTRN
  - NOARBH
- 通过软件和/或 DMA 修改/准备要执行的下一次帧传输的控制字、状态字、读/写数据
  - *I3C 消息控制寄存器 (I3C_CR)*、*I3C 消息控制寄存器 [复用] (I3C_CR)*
  - *I3C 状态寄存器 (I3C_SR)*
  - *I3C 发送数据字节寄存器 (I3C_TDR)*、*I3C 发送数据字寄存器 (I3C_TDWR)*
  - *I3C 接收数据字节寄存器 (I3C_RDR)*、*I3C 接收数据字寄存器 (I3C_RDWR)*
- 通常，在发出并完成广播/直接 DISEC/ENEC CCC 之后：
  - 通过 I3C_CFGR 寄存器中的 HJACK 位修改热加入确认策略
  - 通过 *I3C 设备 x 特性寄存器 (I3C_DEVRx)* 修改对任意目标 x 的 IBI/CR 确认策略

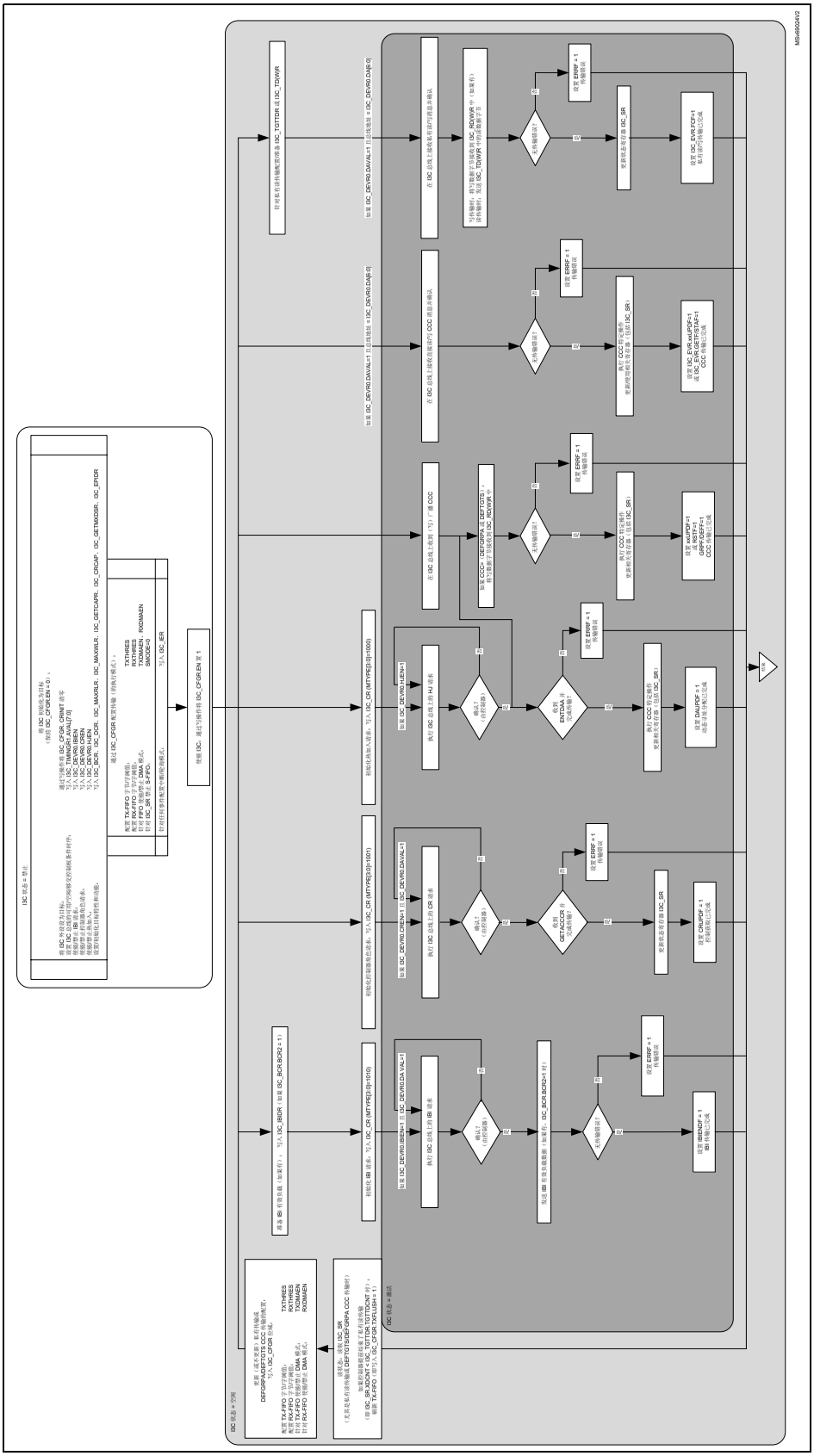
*第 35.8.1 节*中总结了寄存器用法与 I3C 外设控制器角色的关系。

表 275 中总结了外设作为控制器时静态/动态寄存器位域的用法。

35.7.3 I3C 目标状态和编程序列

图 404 给出了外设用作目标时的完整编程序列，包括状态转换、主要子任务和条件，如本节中所述。

图 404. I3C 目标状态和编程序列图



目标初始化

当目标处于禁止状态（I3C_CFGR 寄存器中的 EN = 0）时，软件必须按如下所述进行初始化：

- 将 I3C 外设设置为目标：将 I3C_CFGR 寄存器中的 CRINIT 清零
- 通过 *I3C 时序寄存器 1 (I3C_TIMINGR1)* 设置 I3C 总线时序：写入 AVAL[7:0] 以设置：
  - 总线可用条件时间 (t_{AVAL})，针对 IBI 或控制器角色请求
  - 总线空闲条件时间 (t_{IDLE})，针对热加入请求
  - $t_{NEWCRLock}$，针对控制器角色切换过程（GETACCCR CCC 后）
- 配置目标发起的请求：写入 *I3C 自身器件特性寄存器 (I3C_DEVR0)* 中的以下位
  - IBIEN：带内中断（也称为 IBI）请求使能/禁止
  - CREN：控制器角色请求使能/禁止
  - HJEN：热加入请求使能/禁止
- 初始化目标特性和功能：写入
  - *I3C 总线特性寄存器 (I3C_BCR)*
  - *I3C 设备特性寄存器 (I3C_DCR)*
  - *I3C 最大读取长度寄存器 (I3C_MAXRLR)*
  - *I3C 最大写入长度寄存器 (I3C_MAXWLR)*
  - *I3C 获取功能寄存器 (I3C_GETCAPR)*
  - *I3C 控制器角色功能寄存器 (I3C_CRCAPR)*
  - *I3C 获取最大数据速度寄存器 (I3C_GETMXDSR)*
  - *I3C 扩展配置 ID 寄存器 (I3C_EPIDR)*
- 配置传输的执行模式：*I3C 配置寄存器 (I3C_CFGR)* 中的以下位域：
  - TXDMAEN、RXDMAEN：分别为 TX-FIFO 和 RX-FIFO 使能/禁止 DMA 模式
  - TXTHRES、RXTHRES：分别为 TX-FIFO 和 RX-FIFO 的字节/字阈值
  - 禁止 S-FIFO：SMODE = 0（默认/复位值）
- 配置任意事件的中断生成或轮询模式：*I3C 中断使能寄存器 (I3C_IER)*

然后，软件可使能 I3C 外设（通过写操作将 EN 置 1）。

注意： *软件可以在使能该外设的同时一次写入 I3C_CFGR 寄存器的所有位域。*

从控制器接收（广播 CCC、直接读/写 CCC 或私有读/写）消息

当目标处于空闲状态（I3C_CFGR 寄存器中的 EN = 1）时，即表示已准备好在 I3C 总线上接收来自控制器的通信消息，并且已准备好切换到工作状态。

通常，处于工作状态的目标会先接收广播 ENTDAA CCC（可能在接收到可选的广播 ENEC/DISEC CCC 后），然后再被分配动态地址。I3C_EVR 寄存器中的事件 DAUPF 置 1，并生成相关中断（如果使能），目标随后返回空闲状态。

之后，处于空闲状态的目标便准备好从控制器接收任何其他广播 CCC 消息、直接读/写 CCC 或私有读/写消息。

有关控制器发起的 I3C 总线传输的执行以及作为目标的相关编程（包括更新 I3C 寄存器和位域）的更多信息，请参见*第 35.9 节*中的以下各图：

- *图 408：I3C CCC 消息（作为目标）*
- *图 409：I3C 广播 ENTDAA CCC（作为目标）*
- *图 410：I3C 广播 DEFTGTS CCC（作为目标）*
- *图 411：I3C 广播 DEFGRPA CCC（作为目标）*
- *图 413：I3C 私有读/写消息（作为目标）*

图 404 不包括 FIFO 的管理（TX-FIFO、RX-FIFO）。有关这部分内容的详细信息，请参见*第 35.10 节*。

读取消息状态寄存器

对于每一条无错完成接收的消息，硬件都会通过更新 *I3C 状态寄存器 (I3C_SR)*（软件可通过 *I3C 事件寄存器 (I3C_EVR)* 中的相应标志或 *I3C 中断使能寄存器 (I3C_IER)* 中的相应中断（如果使能）获得通知后读取该寄存器）来报告 I3C 总线上完成了一次交换消息的传输。

出现以下消息后，软件必须读取 *I3C 状态寄存器 (I3C_SR)*：

- 私有读：获取交换的数据字节数，因为控制器可能会比目标预期的更早结束传输（如果 I3C_SR 寄存器中的 XDCNT[15:0] 小于 I3C_TGTTDR 寄存器中的 TGTTDCNT[15:0]）。在这种情况下，软件必须清空 TX-FIFO（在 I3C_CFGR 寄存器中写入 TXFLUSH = 1）。
- DEFTGTS CCC 或 DEFGRPA CCC：获取 RX-FIFO 中接收到的数据字节数

启动（目标发起的）传输

当目标先从禁止状态进入空闲状态（软件在 I3C_CFGR 寄存器中写入 EN = 1），同时能够从控制器接收广播 CCC 时，软件可以发出热加入请求（软件在 I3C_CR 寄存器中写入 MTYPE[3:0] = 1000）以申请资格加入下一个 ENTDAA CCC，前提是已使能热加入请求（I3C_DEVR0 寄存器中的 HJEN = 1）。

分配动态地址（I3C_EVR 寄存器中的 DAUPF = 1）后（与控制器发出的帧传输同时进行的情况更为常见），软件可通过将相关控制字写入 I3C_CR 寄存器来向控制器发出 IBI（带内中断请求）或控制器角色请求。

有关目标发起的 I3C 总线传输的执行以及作为目标的相关编程的更多信息，请参见*第 35.9 节*中的以下各图：

- *图 415：IBI 传输（作为控制器/目标）*
- *图 416：热加入请求传输（作为控制器/目标）*
- *图 417：控制器角色请求传输（作为控制器/目标）*

更新 I3C 外设的配置（作为目标）

返回到空闲状态后，软件可以在下一次传输前更新 I3C 目标的配置：

- 通过 *I3C 中断使能寄存器 (I3C_IER)* 修改中断/轮询模式策略
- 修改 *I3C 配置寄存器 (I3C_CFGR)* 的以下位域：
  - TXTHRES、RXTHRES
  - TXDMAEN、RXDMAEN
- 修改/准备 *I3C IBI 有效负载数据寄存器 (I3C_IBIDR)*，如果有任何有效负载（I3C_BCR 寄存器中的 BCR2 = 1 时），应在启动 IBI 传输（向 *I3C 消息控制寄存器 [复用] (I3C_CR)* 写入 MTYPE[3:0] = 1010）前执行该操作

- 修改/准备 *I3C 目标发送配置寄存器 (I3C_TGTTDR)*，在从控制器接收私有读或直接 CCC 读（GETSTATUS CCC 除外）消息之前，禁止或使能 TX-FIFO 预装载指定数量的待发送数据字节

*第 35.8.1 节*中总结了寄存器用法与 I3C 外设目标角色的关系。

表 276 中总结了外设作为目标时静态/动态寄存器位域的用法。

35.8 I3C 寄存器和编程

35.8.1 I3C 寄存器组（作为控制器/目标）

表 274 列出了寄存器及其用法与 I3C 外设角色的关系。

表 274. I3C 寄存器用法

寄存器	用作控制器	用作目标
I3C_CR	X	X
I3C_CFGR	X	X
I3C_RDR	X	X
I3C_RDWR	X	X
I3C_TDR	X	X
I3C_TDWR	X	X
I3C_IBIDR	X	X
I3C_TGTTDR	-	X
I3C_SR	X	X
I3C_SER	X	X
I3C_RMR	X	X
I3C_EVR	X	X
I3C_IER	X	X
I3C_CEVR	X	X
I3C_DEVR0	X	X
I3C_DEVRx (x = 1..4)	X	-
I3C_MAXRLR	-	X
I3C_MAXWLR	-	X
I3C_TIMINGR0	X	-
I3C_TIMINGR1	X	X
I3C_TIMINGR2	X	-
I3C_BCR	-	X
I3C_DCR	-	X
I3C_GETCAPR	-	X
I3C_CRCAPR	-	X

表 274. I3C 寄存器用法（续）

寄存器	用作控制器	用作目标
I3C_GETMXDSR	-	X
I3C_EPIDR	-	X

35.8.2 I3C 寄存器和位域用法与外设状态的关系（作为控制器）

表 275 列出了当 I3C 外设用作控制器时，寄存器及其用法与控制器状态（禁止、空闲和活动）的关系。

表 275. I3C 寄存器/位域用法与控制器状态的关系

寄存器	用作控制器	只可写（在禁止状态下）	通常可写/读（在空闲状态下）	通常可写/读（在空闲状态或工作状态下）
I3C_CR	X	-	-	X
I3C_CFGR	X	CRINIT HKSDAEN EN(1)	写入：除 {CRINIT、HKSDAEN} 之外使用的任何位域，即 {CDMAEN SDMAEN TXDMAEN RXDMAEN TMODE SMODE TXTHRES RXTHRES HJACK EXITPTRN RSTPTRN NOARBH}(2) CFLUSH SFLUSH TXFLUSH RXFLUSH TSFSET	-
I3C_RDR	X	-	-	读
I3C_RDWR	X	-	-	读
I3C_TDR	X	-	-	写
I3C_TDWR	X	-	-	写
I3C_IBIDR	X	-	-	读
I3C_TGTTDR	-			
I3C_SR	X	-	-	读
I3C_SER	X	-	读	-

表 275. I3C 寄存器/位域用法与控制器状态的关系（续）

寄存器	用作控制器	只可写（在禁止状态下）	通常可写/读（在空闲状态下）	通常可写/读（在空闲状态或工作状态下）
I3C_RMR	X	-	-	读取 RADD[6:0] IBIRDCNT[2:0]
I3C_EVR	X	-	-	读（控制器角色位域）： HJF CRF IBIF FCF ERRF RXTGTENDF RXFNEF TXFNFF SFNEF CFNFF RXLASTF TXLASTF TXFEF CFEF
I3C_IER	X	写入 xIE，其中 x = HJ、CR、IBI、FC、ERR、RXTGTEND、RXFNE、TXFNF、SFNE、CFNF(2)		
I3C_CEVR	X	-	-	写（控制器角色位域，请参见 I3C_EVR）
I3C_DEVR0	X	写入 DA[6:0](2)		-
I3C_DEVRx (x = 1..4)	X	写入 SUSP、IBIDEN、IBIACK、CRACK、DA[6:0](2)		-
I3C_MAXRLR	-			
I3C_MAXWLR	-			
I3C_TIMINGR0	X	X	-	-
I3C_TIMINGR1	X	X	-	-
I3C_TIMINGR2	X	写(2)		
I3C_BCR	-			
I3C_DCR	-			
I3C_GETCAPR	-			
I3C_CRCAPR	-			
I3C_GETMXDSR	-			
I3C_EPIDR	-			

1. 写入 I3C_CFGR 寄存器中的 EN 位，设置为禁止状态（该位为 0 时）。该位域也可以在空闲状态下写入和清零。
2. 这些位域通常在总线配置期间以禁止状态写入和初始化。当 EN = 0 时，它们不受写保护，因此在其他状态下也可以写入和更新这些位域。

35.8.3 I3C 寄存器和位域用法与外设状态的关系（作为目标）

表 276 列出了当外设用作目标时，寄存器及其用法与 I3C 目标状态（禁止、空闲和活动）的关系。

表 276. I3C 寄存器/位域用法与目标状态的关系

寄存器	用作目标	只可写 （在禁止状态下）	通常可写/读 （在空闲状态下）	通常可写/读（在空闲状态或工作状态下）
I3C_CR	X	-	MTYPE[3:0] DCNT[2:0]	-
I3C_CFGR	X	CRINIT EN(1)	写入：除 CRINIT 之外使用的任何位域，即 {TXDMAEN RXDMAEN TXTHRES RXTHRES}(2) TXFLUSH RXFLUSH	-
I3C_RDR	X	-	-	读
I3C_RDWR	X	-	-	读
I3C_TDR	X	-	-	写
I3C_TDWR	X	-	-	写
I3C_IBIDR	X	-	写	-
I3C_TGTTDR	X	-	写	-
I3C_SR	X	-	-	读 DIR、XDCNT[15:0]
I3C_SER	X	-	读取 DOVR、STALL、PERR、CODERR[3:0]	-
I3C_RMR	X	-	-	读取 RCODE[7:0]

表 276. I3C 寄存器/位域用法与目标状态的关系（续）

寄存器	用作目标	只可写 （在禁止状态下）	通常可写/读 （在空闲状态下）	通常可写/读（在空闲状态或工作状态下）
I3C_EVR	X	-	-	读（目标角色位域）： GRPF DEFF INTUPDF ASUPDF RSTF MRLUPDF MWLUPDF DAUPDF STAF GETF WKPF CRUPDF IBIENDF ERRF FCF RXFNEF TXFNFF TXLASTF TXFEF
I3C_IER	X	写入 xIE，其中 x = GRP、DEF、INTUPD、ASUPD、RST、MRLUPD、MWLUPD、DAUPD、STA、GET、WKP、CRUPD、IBIEND、ERR、FC、RXFNE、TXFNF(2)		
I3C_CEVR	X	-	-	写（目标角色位域，请参见 I3C_EVR）
I3C_DEVR0	X	HJEN CREN IBIEN	读取 RSTVAL、RSTACT[1:0] 和 AS[1:0]	-
I3C_DEVRx (x = 1..4)	-			
I3C_MAXRLR	X	X	-	-
I3C_MAXWLR	X	X	-	-
I3C_TIMINGR0	-			
I3C_TIMINGR1	X	AVAL[7:0]	-	-
I3C_TIMINGR2	-			
I3C_BCR	X	X	-	-
I3C_DCR	X	X	-	-
I3C_GETCAPR	X	X	-	-
I3C_CRCAPR	X	X	-	-

表 276. I3C 寄存器/位域用法与目标状态的关系（续）

寄存器	用作目标	只可写（在禁止状态下）	通常可写/读（在空闲状态下）	通常可写/读（在空闲状态或工作状态下）
I3C_GETMXDSR	X	X	-	-
I3C_EPIDR	X	X	-	-

1. 写入 I3C_CFGR 寄存器中的 EN 位，设置为禁止状态（该位为 0 时）。该位域也可以在空闲状态下写入和清零。
2. 这些位域通常在 I3C 总线配置期间以禁止状态写入和初始化。当 EN = 0 时，它们不受写保护。

35.9 I3C 总线传输和编程

35.9.1 I3C 命令集 (CCC)（作为控制器/目标）

表 277 提供了支持的 I3C 命令集列表（例如 CCC 列表、通用命令代码），以及外设作为控制器或目标时如何处理该命令集的概述。

表 277. 支持的 I3C CCC 列表（作为控制器/目标）

CCC 名称	CCC 值	读/写	有/无释义字节 有/无子命令字节	有/无 可选数据字节	用作控制器	用作目标，生成 I3C_EVR 事件	作为目标时：特定操作
广播 CCC							
ENEC	0x00	写	无释义字节/子命令字节	有一个数据字节（使能目标事件字节）	X	X，INTUPDF	更新并使能 I3C_DEVR0：HJEN、CREN、IBIEN（如果有）
DISEC	0x01			有一个数据字节（禁止目标事件字节）	X	X，INTUPDF	更新并禁止 I3C_DEVR0：HJEN、CREN、IBIEN（如果有）
ENTASx (x = 0...3)	0x02 ... 0x05			无数据字节	X	X，ASUPDF	更新 I3C_DEVR0.AS[1:0]
RSTDAA	0x06			-	X	X，DAUPDF	将 I3C_DEVR0.DAVAL 清零
ENTDAA	0x07			-	X	X，DAUPDF	更新 I3C_DEVR0：DA[6:0] 并将 DAVAL 置 1
DEFTGTS	0x08			有 [1+ 4x (1+n)] 个数据字节 (n- 目标数量)	X	X，DEFF	更新 I3C_RDR/I3C_RDWR。请参见*图 410*。
SETMWL	0x09			有两个数据字节	X	X，MWLUPDF	更新 I3C_MAXWLR
SETMRL	0x0A			有两个或三个数据字节	X	X，MRLUPDF	更新 I3C_MAXRLR
ENTTM	0x0B			有一个数据字节	X	-	
SETXTIME	0x28		有子命令字节	无/有一个或多个数据字节	X		
SETAASA	0x29		无释义字节/子命令字节	无数据字节	X		
RSTACT	0x2A		有释义字节（0x00、0x01 或 0x02）		X	X，检测到复位时序后置位 RSTF	更新 I3C_DEVR0：RSTACT[1:0] 并将 RSTVAL 置 1
DEFGRPA	0x2B		无释义字节/子命令字节	有多个数据字节	X	X，GRPF	更新 I3C_RDR/RDWR。请参见*图 411*。
RSTGRPA	0x2C			无数据字节	X	-	-

表 277. 支持的 I3C CCC 列表（作为控制器/目标）（续）

CCC 名称	CCC 值	读/写	有/无释义字节 有/无子命令字节	有/无 可选数据字节	用作控制器	用作目标，生成 I3C_EVR 事件	作为目标时： 特定操作
直接 CCC							ACK 时执行操作（如果 I3C 目标地址 = I3C_DEVR0.DA[6:0] 且 I3C_DEVR0.DAVAL = 1）（否则为 NACK）
ENEC	0x80	写	无释义字节/子命令字节	有一个数据字节（使能目标事件字节）	X	X，INTUPDF	更新并使能 I3C_DEVR0：HJEN、CREN、IBIEN（如果有）
DISEC	0x81			有一个数据字节（禁止目标事件字节）	X	X，INTUPDF	更新并禁止 I3C_DEVR0：HJEN、CREN、IBIEN（如果有）
ENTASx (x = 0...3)	0x82...0x85			无数据字节	X	X，ASUPDF	更新 I3C_DEVR0.AS[1:0]
SETDASA	0x87			无数据字节	X	-	-
SETNEWDA	0x88			有一个数据字节	X	X，DAUPDF	更新 I3C_DEVR0：DA[6:0]（并将 DAVAL 置 1）
SETMWL	0x89			有两个数据字节	X	X，MWLUPDF	更新 I3C_MAXWLR
SETMRL	0x8A			有两个或三个数据字节	X	X，MRLUPDF	更新 I3C_MAXRLR
GETMWL	0x8B	读	无释义字节/子命令字节	有两个数据字节	X	X，GETF	从 I3C_MAXWLR[15:0] 返回数据字节。请参见 *第 35.16.19 节*。
GETMRL	0x8C			有两个或三个数据字节	X	X，GETF	从 I3C_MAXRLR[15:0] 返回数据字节，如果 I3C_BCR.BCR2 = 1，则从 I3C_MAXRLR.IBIP[2:0] 返回第三个字节。请参见 *第 35.16.18 节*。

表 277. 支持的 I3C CCC 列表（作为控制器/目标）（续）

CCC 名称	CCC 值	读/写	有/无释义字节 有/无子命令字节	有/无 可选数据字节	用作控制器	用作目标，生成 I3C_EVR 事件	作为目标时： 特定操作
GETPID	0x8D	读	无释义字节/ 子命令字节	有六个数据字节	X	X，GETF	从 I3C_EPIDR 返回数据字节。请参见 *第 35.16.28 节*。
GETBCR	0x8E			有一个数据字节	X	X，GETF	从 I3C_BCR[7:0] 返回数据字节。请参见 *第 35.16.23 节*。
GETDCR	0x8F				X	X，GETF	返回 I3C_DCR[7:0]。请参见 *第 35.16.24 节*。
GETSTATUS	0x90		有/无释义字节（TGTSTAT、PRECR）	有两个数据字节（格式 1 或格式 2（有 PRECR 时））	X	X，STAF（格式 1 时） X，GETF（格式 2 时）	返回两个数据字节，详细信息请参见 *第 35.9.9 节*。
GETACCCR	0x91		无释义字节/子命令字节	有一个数据字节	X	X，CRUPDF	从 I3C_DEVR0.DA[6:0] 返回带奇偶校验位的数据字节
GETMXDS	0x94		有/无释义字节（WRRDTURN、CRHDLY）	有两个数据字节（格式 1）、五个数据字节（格式 2 或格式 3（有 WRRDTURN 时））或一个数据字节（格式 3（有 CRHDLY 时））	X	X，GETF	从 I3C_GETMXDSR 返回数据字节。 请参见 *第 35.16.27 节*。
GETCAPS	0x95		有/无释义字节（TGTSTAT、CRCAPS）	有三个数据字节（格式 1 或格式 2（有 TGTSTAT 时））或两个数据字节（格式 2（有 CRCAPS 时））	X	X，GETF	从 I3C_GETCAPR 返回 3 个 GETCAPx 数据字节（请参见 *第 35.16.25 节*） 或者 从 I3C_CRCAPR 返回 2 个 CRCAPx 数据字节（请参见 *第 35.16.26 节*）
D2DXFER	0x97	写	有释义字节	有释义字节	X	-	-
SETXTIME	0x98		有子命令字节	有子命令字节	X		
GETXTIME	0x99	读	无释义字节/ 子命令字节	无释义字节/ 子命令字节	X		
RSTACT	0x9A	读/写	有释义字节（0x00、0x01 或 0x02）	有释义字节（0x00、0x01 或 0x02）	X	X，检测到复位时序后置位 RSTF	读：从 I3C_DEVR0 寄存器中的 RSTACT[1:0] 返回数据字节。 写：更新 I3C_DEVR0：RSTACT[1:0] 并将 RSTVAL 置 1

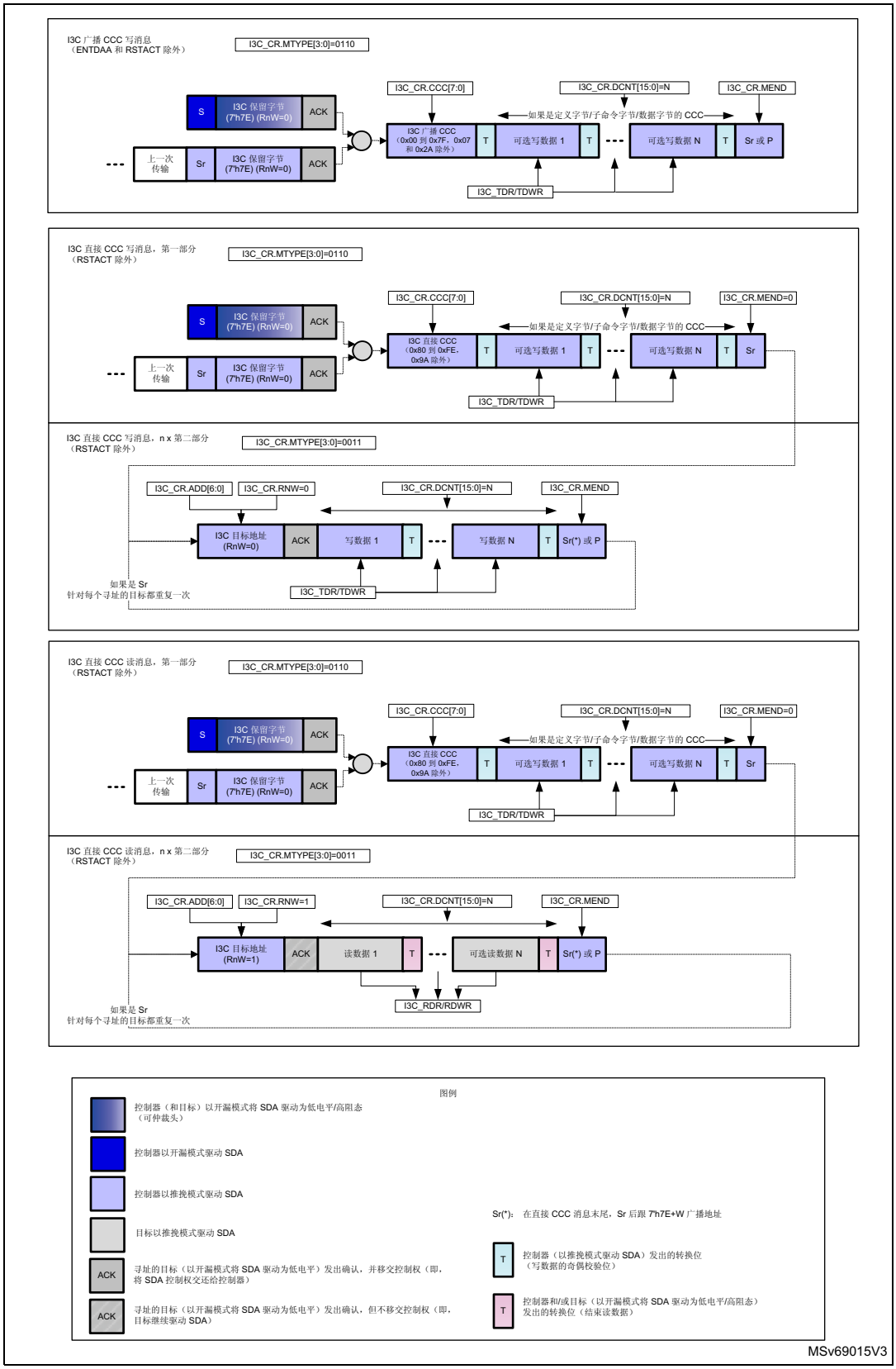
表 277. 支持的 I3C CCC 列表（作为控制器/目标）（续）

CCC 名称	CCC 值	读/写	有/无释义字节 有/无子命令字节	有/无 可选数据字节	用作控制器	用作目标，生成 I3C_EVR 事件	作为目标时：特定操作
SETGRPA	0x9B	写	无释义字节/ 子命令字节	无释义字节/ 子命令字节	X	-	-
RSTGRPA	0x9C				X		

35.9.2 I3C 广播/直接 CCC 传输（ENTDAA 和 RSTACT 除外）（作为控制器）

图 405 说明了当外设用作控制器时在 I3C 总线上进行 I3C 广播 CCC 写传输（ENTDAA、RSTACT 除外）和直接 CCC 读/写传输。

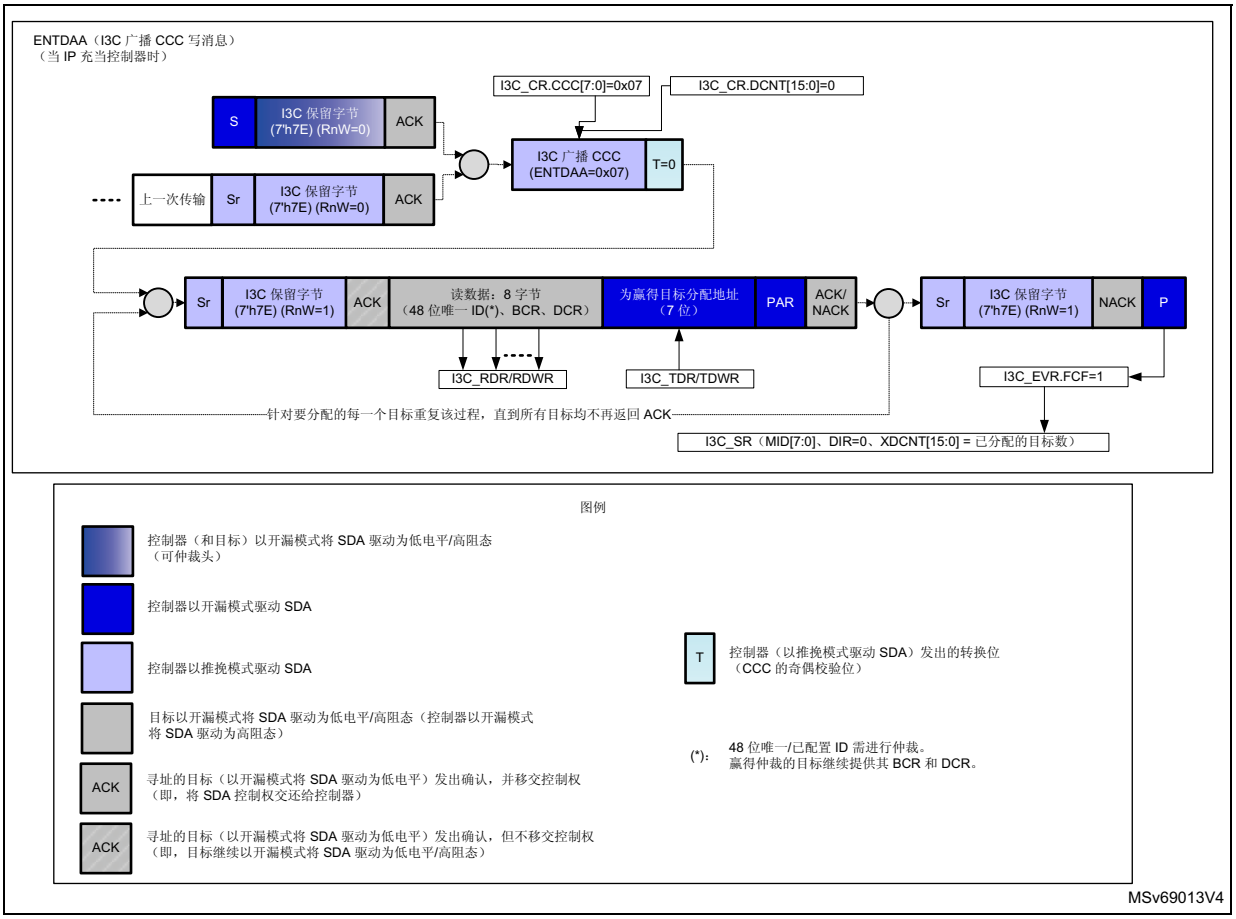
图 405. I3C CCC 消息（作为控制器）



35.9.3 I3C 广播 ENTDAA CCC 传输（作为控制器）

图 406 说明了当外设用作控制器时在 I3C 总线上传输的 I3C 广播 ENTDAA CCC。

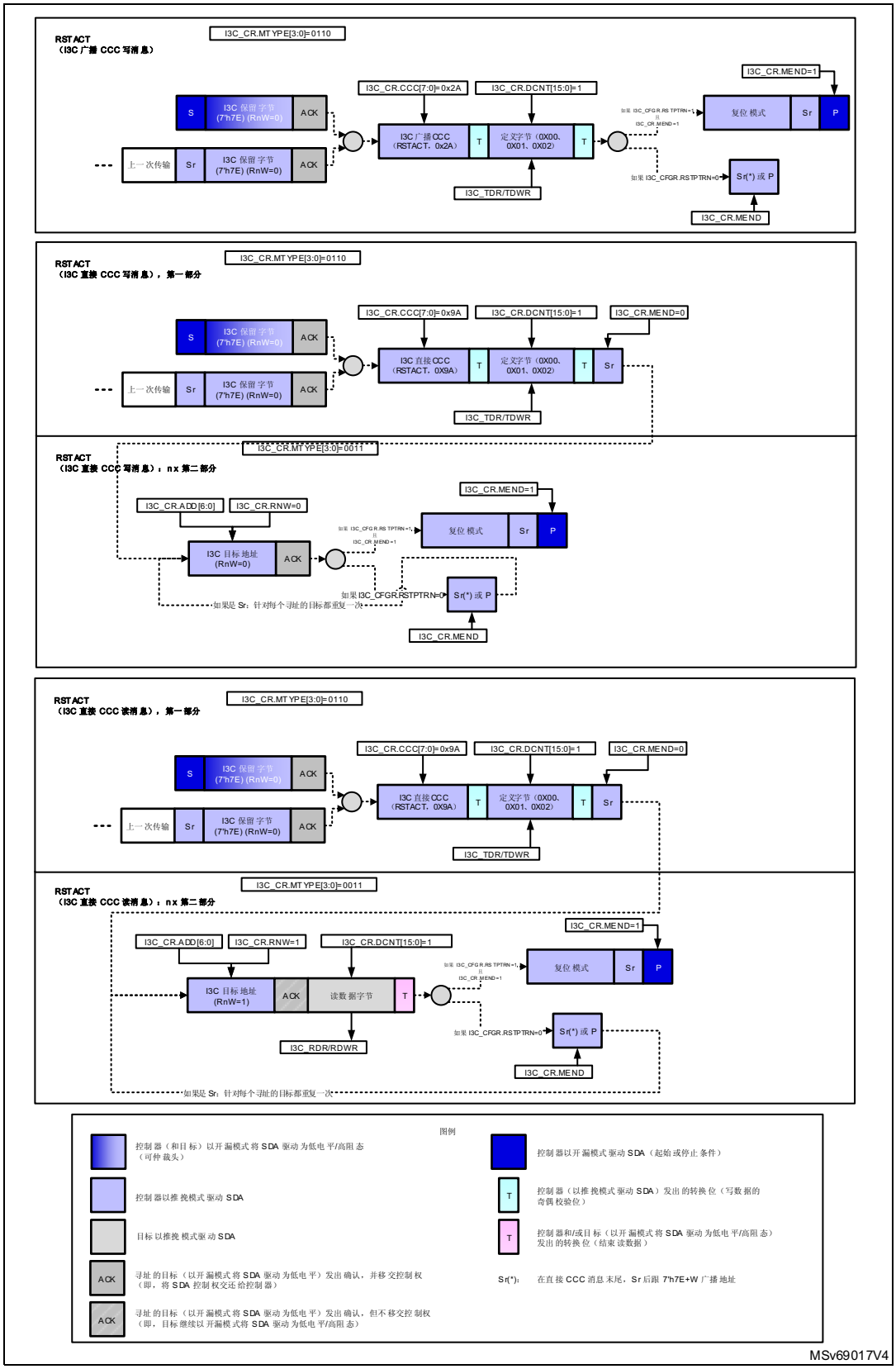
图 406. I3C 广播 ENTDAA CCC（作为控制器）



35.9.4 I3C 广播/直接 RSTACT CCC 传输（作为控制器）

图 407 说明了当外设用作控制器时在 I3C 总线上传输的 I3C 广播（写）、直接写/读 RSTACT CCC。

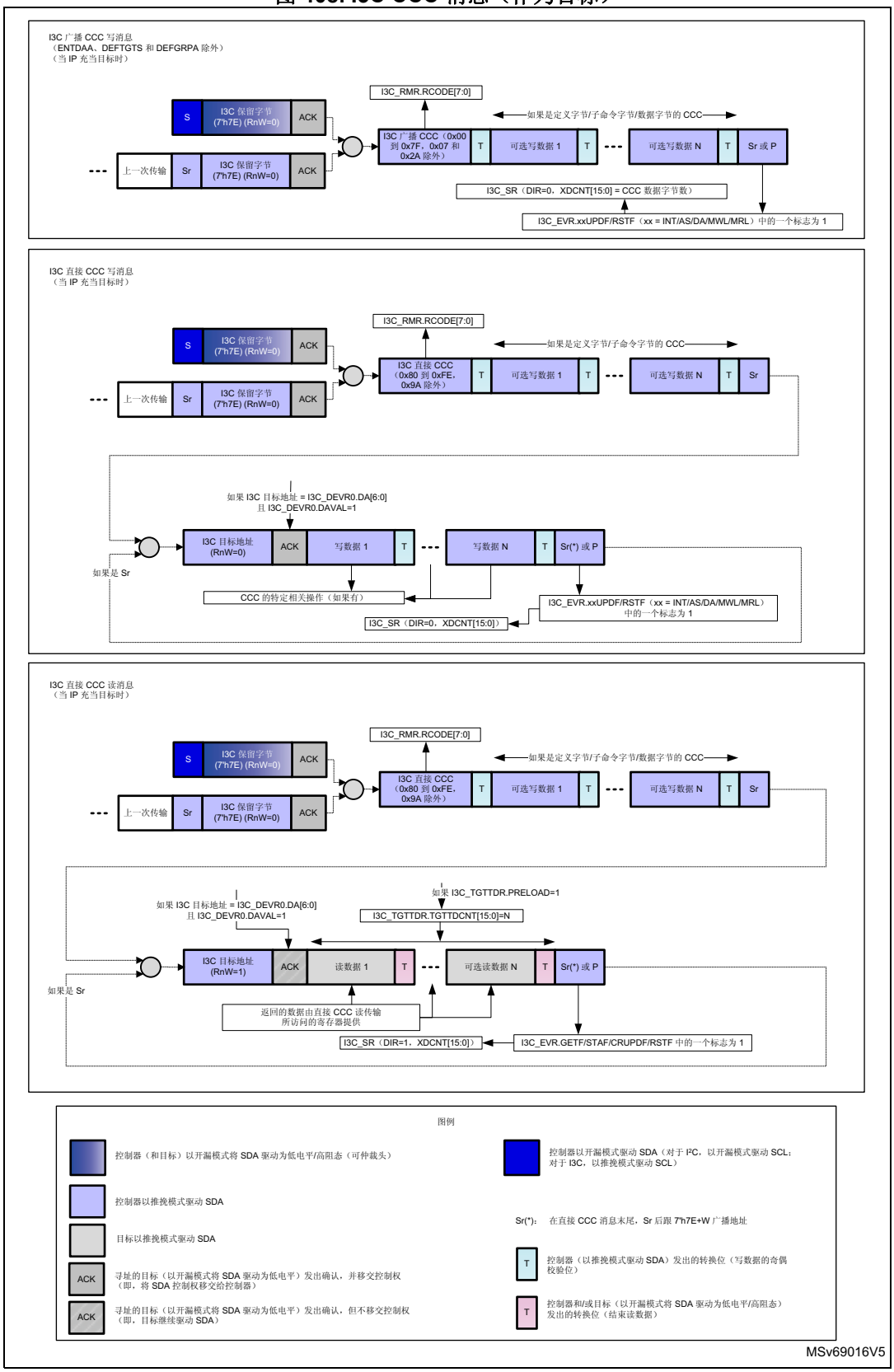
图 407. I3C 广播、直接读/写 RSTACT CCC（作为控制器）



35.9.5 I3C 广播/直接 CCC 传输（ENTDAA、DEFTGTS、DEFGRPA 除外）（作为目标）

图 408 说明了当外设用作目标时在 I3C 总线上进行的 I3C 广播 CCC 写传输（ENTDAA、DEFTGTS、DEFGRPA 除外）、直接 CCC 读/写传输。

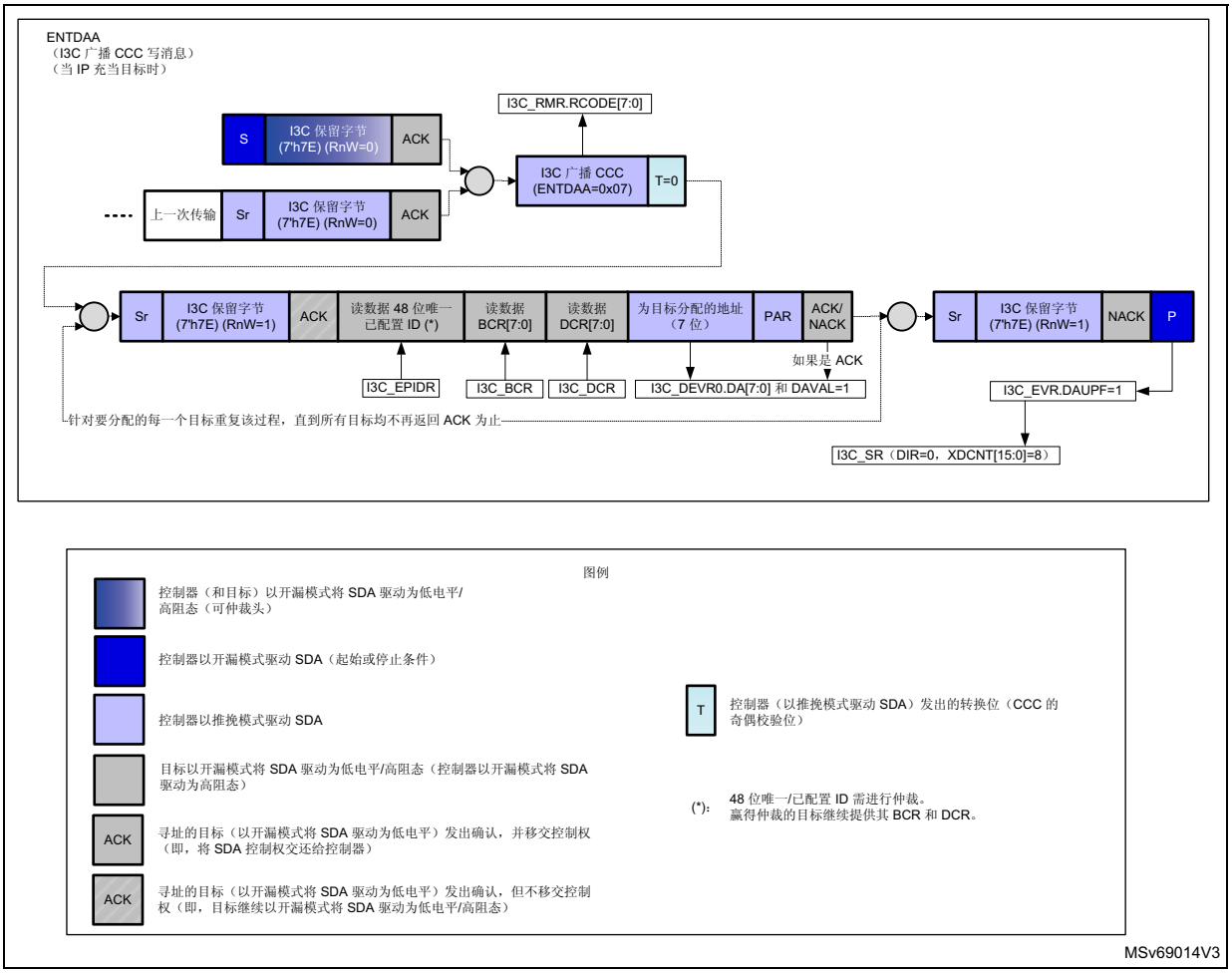
图 408. I3C CCC 消息（作为目标）



35.9.6 I3C 广播 ENTDAA CCC 传输（作为目标）

图 409 说明了当外设作为目标时在 I3C 总线上传输的 I3C 广播 ENTDAA CCC。

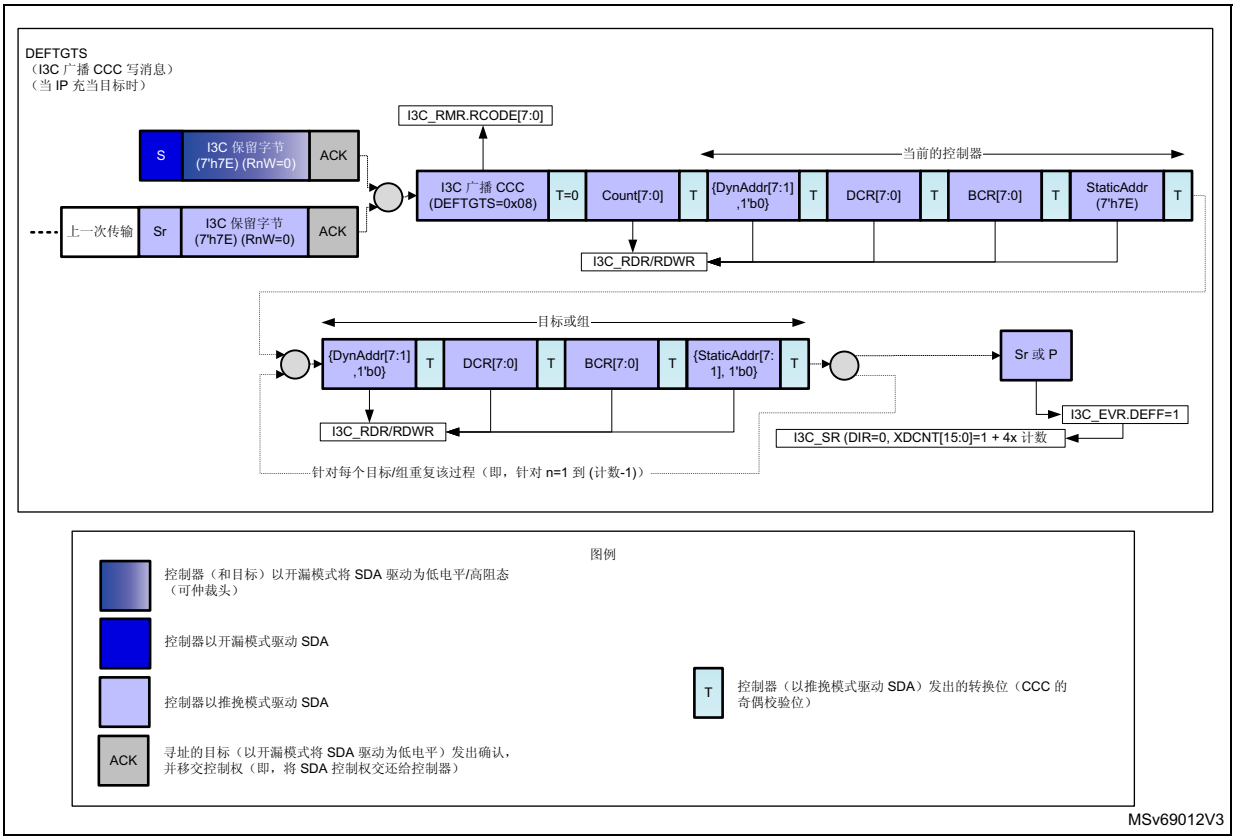
图 409. I3C 广播 ENTDAA CCC（作为目标）



35.9.7 I3C 广播 DEFTGTS CCC 传输（作为目标）

图 410 说明了当外设作为目标时在 I3C 总线上传输的 I3C 广播 DEFTGTS CCC。

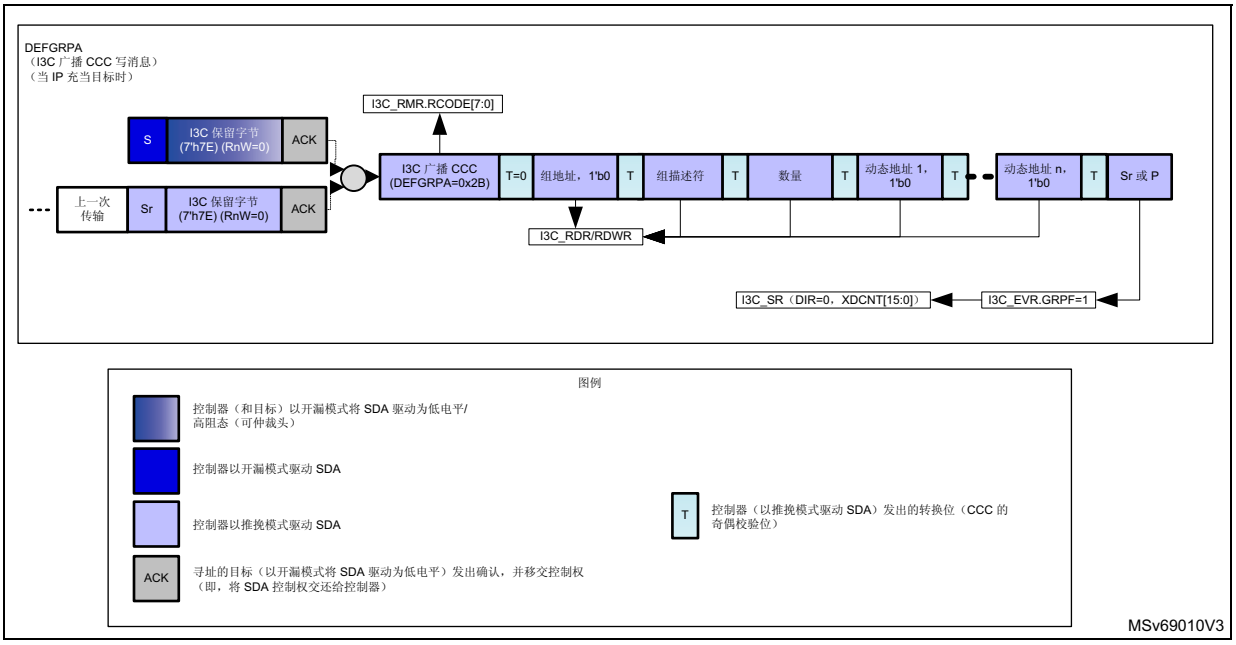
图 410. I3C 广播 DEFTGTS CCC（作为目标）



35.9.8 I3C 广播 DEFGRPA CCC 传输（作为目标）

图 411 说明了当外设作为目标时在 I3C 总线上传输的 I3C 广播 DEFGRPA CCC。

图 411. I3C 广播 DEFGRPA CCC（作为目标）



35.9.9 I3C 直接 GETSTATUS CCC 响应（作为目标）

当 I3C 作为目标时，硬件会在接收到 GETSTATUS CCC 时以格式 1（无释义字节或释义字节 TGTSTAT = 0x00）或格式 2（释义字节 PRECR = 0x91）返回两个数据字节。

I3C 总线上以格式 1 返回的 2 字节 STATUS[15:0] 如下：

- STATUS[15:14] = 00（未使用）
- STATUS[13] = 1（如果自接收到前一个 GETSTATUS CCC 后检测到丢失的起始位）或 0（其他情况）
- STATUS[12] = 1（如果自接收到前一个 GETSTATUS CCC 后检测到上溢/下溢错误）或 0（其他情况）
- STATUS[11] = 1（如果自接收到前一个 GETSTATUS CCC 后，在 SDR 读传输期间检测到 SCL 的稳定时间超过 125 µs）或 0（其他情况）
- STATUS[10:8] = 000 到 110（如果自接收到前一个 GETSTATUS CCC 后检测到协议错误（前提是 STATUS[5] = 1），编码值 x = 0 到 6，对应于目标错误 TEx）或 000（其他情况）
- STATUS[7:6] = 00（准备好执行切换过程）
- STATUS[5] = 1（如果自接收到前一个 GETSTATUS CCC 后检测到协议错误）或 0（其他情况）
- STATUS[4] = 0（保留）
- STATUS[3:1] = 000（未使用）
- STATUS[0] = 1（如果存在挂起的中断（前提是在 I3C_CR 寄存器中配置了 IBI，I3C_DEVR0 寄存器中的 IBIEN = 1 且 DAVAL = 1，而且 IBI 尚未被控制器确认也未通过 DISEC 禁止）或 0（其他情况）

I3C 总线上以格式 2 返回的 2 字节 STATUS[15:0] 如下：

- STATUS[15:8] = 0000 0000（未使用）
- STATUS[7:2] = 00000（未使用）
- STATUS[1] = 1（如果软件仍在处理接收到的 DEFTGTS 或接收到的 DEFGRPA CCC，并且尚未将相关事件清除（I3C_EVR 寄存器中的 DEFF = 1 或 GRPF = 1））；控制器必须等待软件完成上述操作后再发出 GETACCR CCC（否则不会被确认）
- STATUS[0] = 1（如果 DEFTGTS 或 DEFGRPA CCC 可能已丢失）。如果检测到丢失的起始位（I3C_EVR 寄存器中的 WKPF = 1），该位将会置 1；如果 I3C_EVR 寄存器中的 DEFF = 1 或 GRPF = 1，该位将会清零。

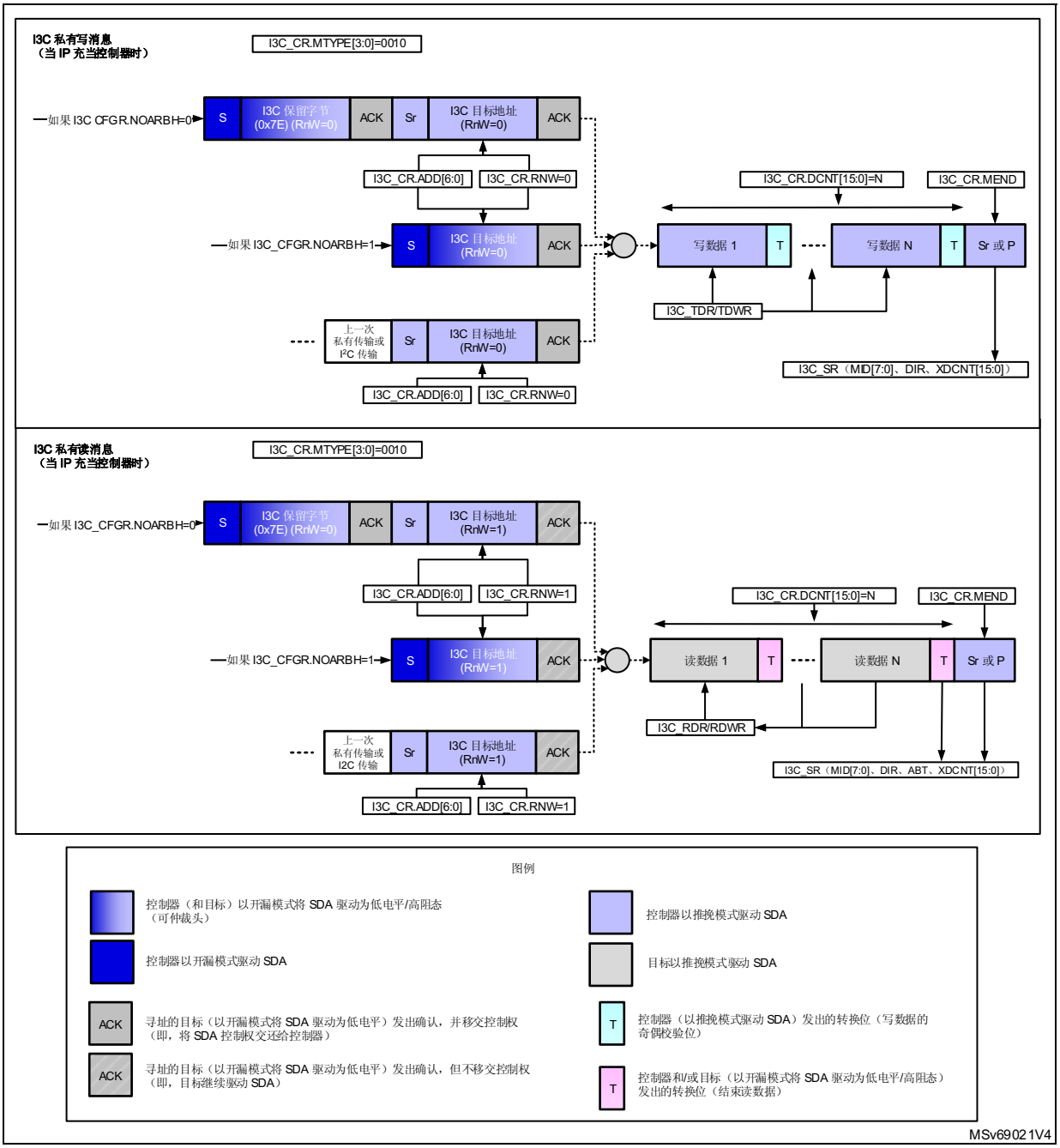
当格式 1 的 GETSTATUS CCC 完成后，将通过 I3C_EVR 寄存器中的 STAF = 1 和相应的中断（如果使能，即 I3C_IER 寄存器中的 STAIE = 1）进行报告。

当格式 2 的 GETSTATUS CCC 完成后，将通过 I3C_EVR 寄存器中的 GETF = 1 和相应的中断（如果使能，即 I3C_IER 寄存器中的 GETIE = 1）进行报告。

35.9.10 I3C 私有读/写传输（作为控制器）

图 412 说明了当外设用作控制器时在 I3C 总线上进行的私有读/写传输。

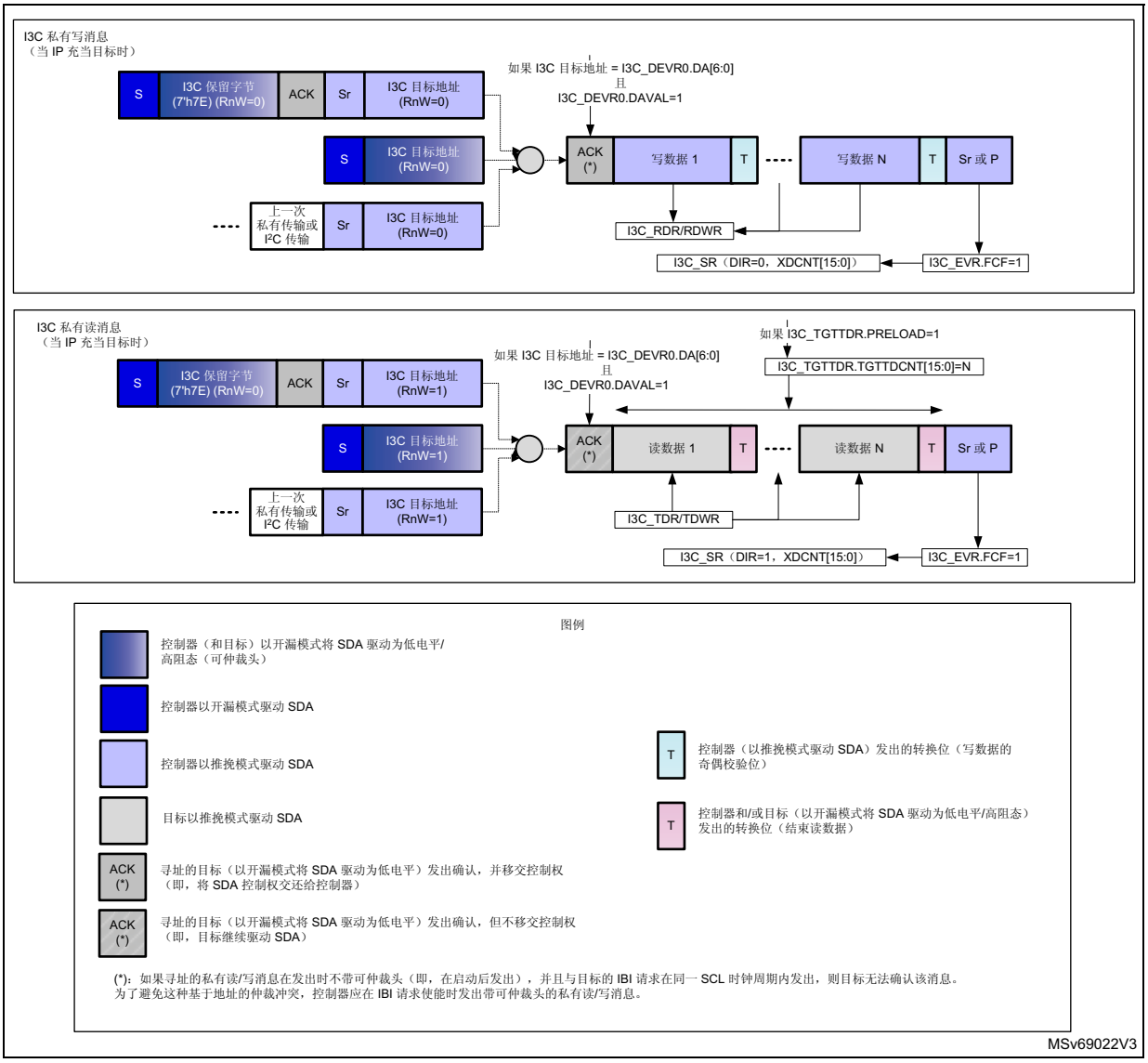
图 412. I3C 私有读/写消息（作为控制器）



35.9.11 I3C 私有读/写传输（作为目标）

图 413 说明了当外设用作目标时在 I3C 总线上进行的 I3C 私有读/写传输。

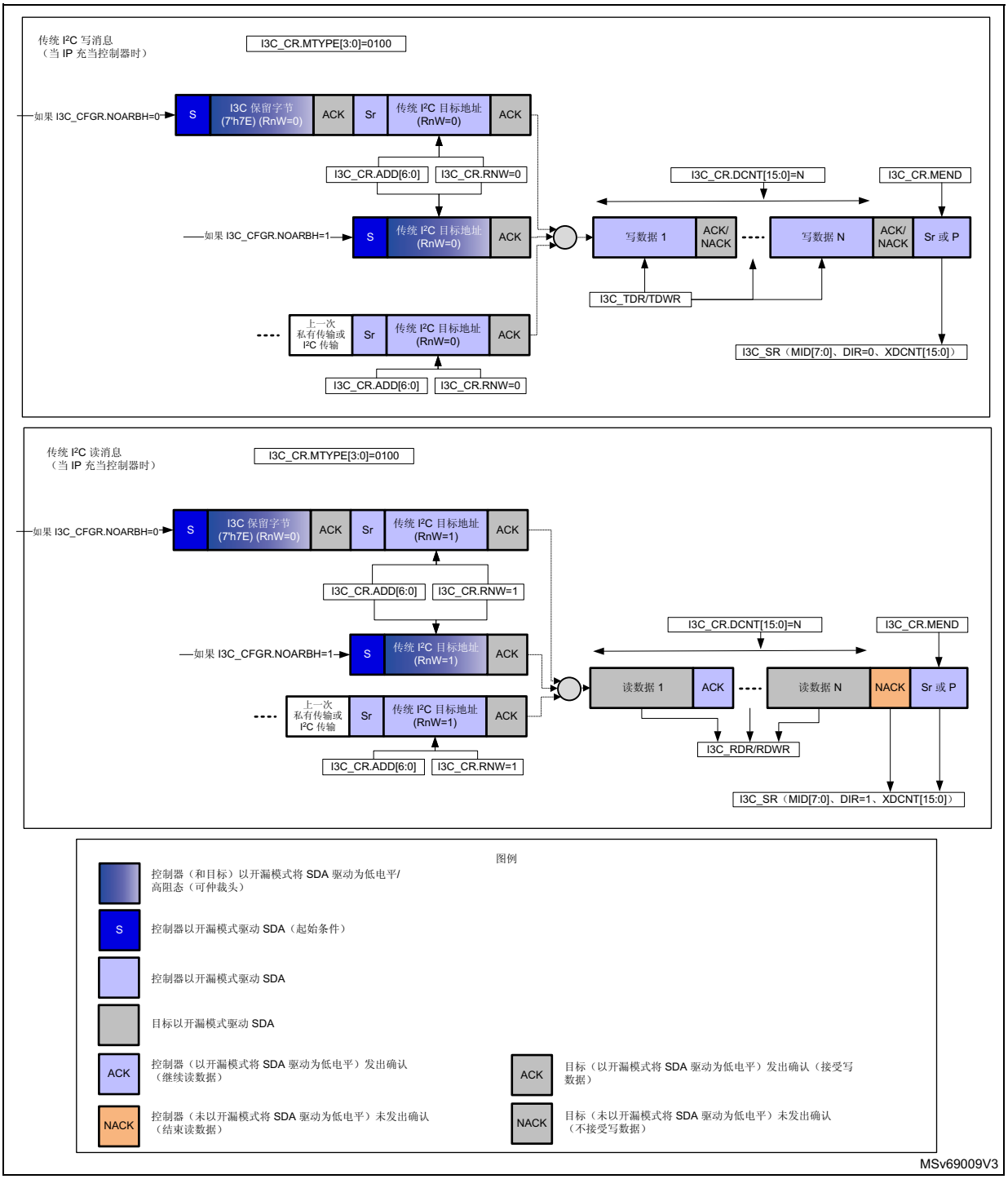
图 413. I3C 私有读/写消息（作为目标）



35.9.12 传统 I²C 读/写传输（作为控制器）

图 414 说明了当外设用作控制器时在 I3C 总线上进行的传统 I²C 读/写传输。

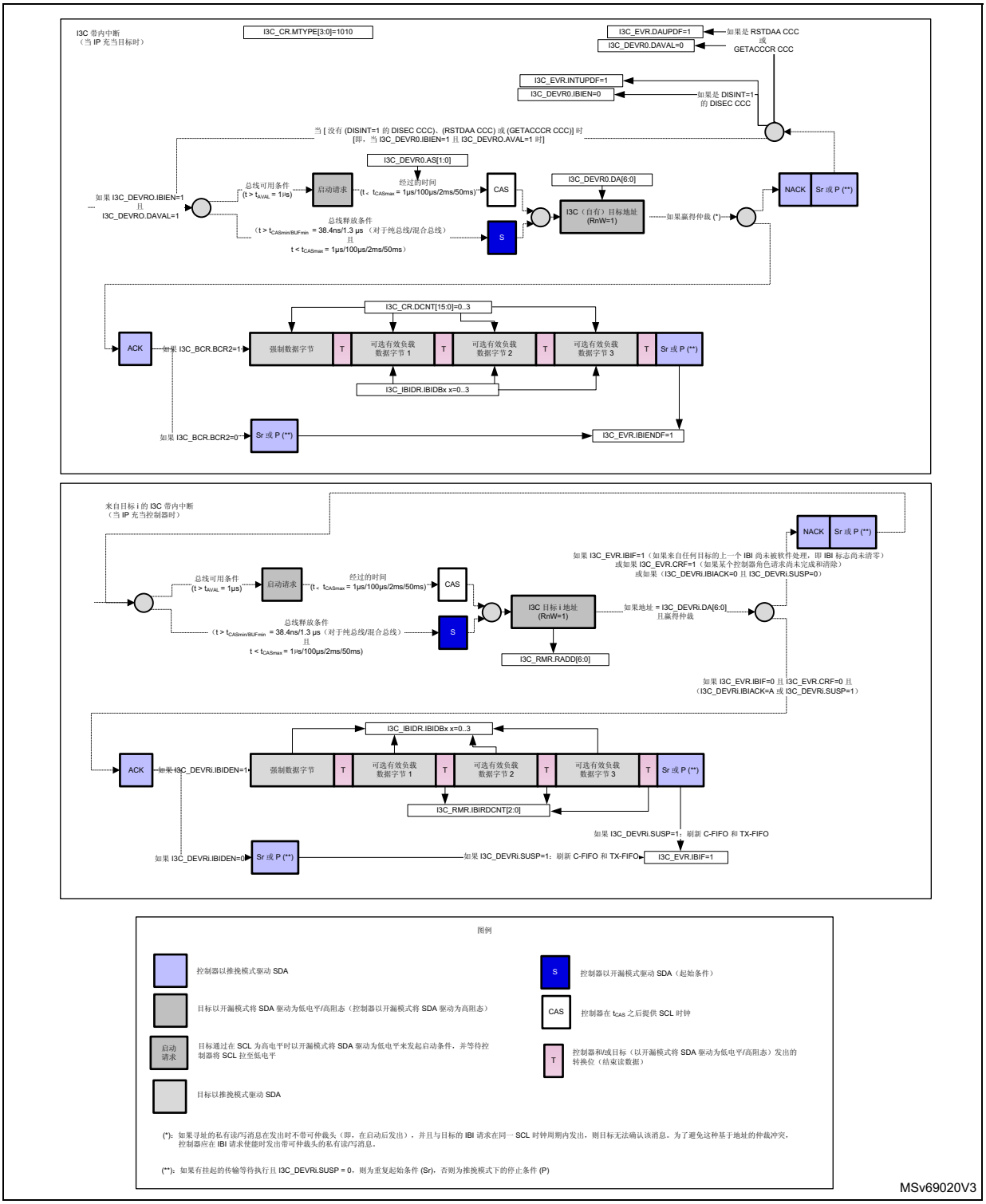
图 414. 传统 I²C 读/写消息（作为控制器）



35.9.13 I3C IBI 传输（作为控制器/目标）

图 415 说明了当外设用作目标或控制器时在 I3C 总线上进行的 IBI（带内中断）传输。

图 415. IBI 传输（作为控制器/目标）

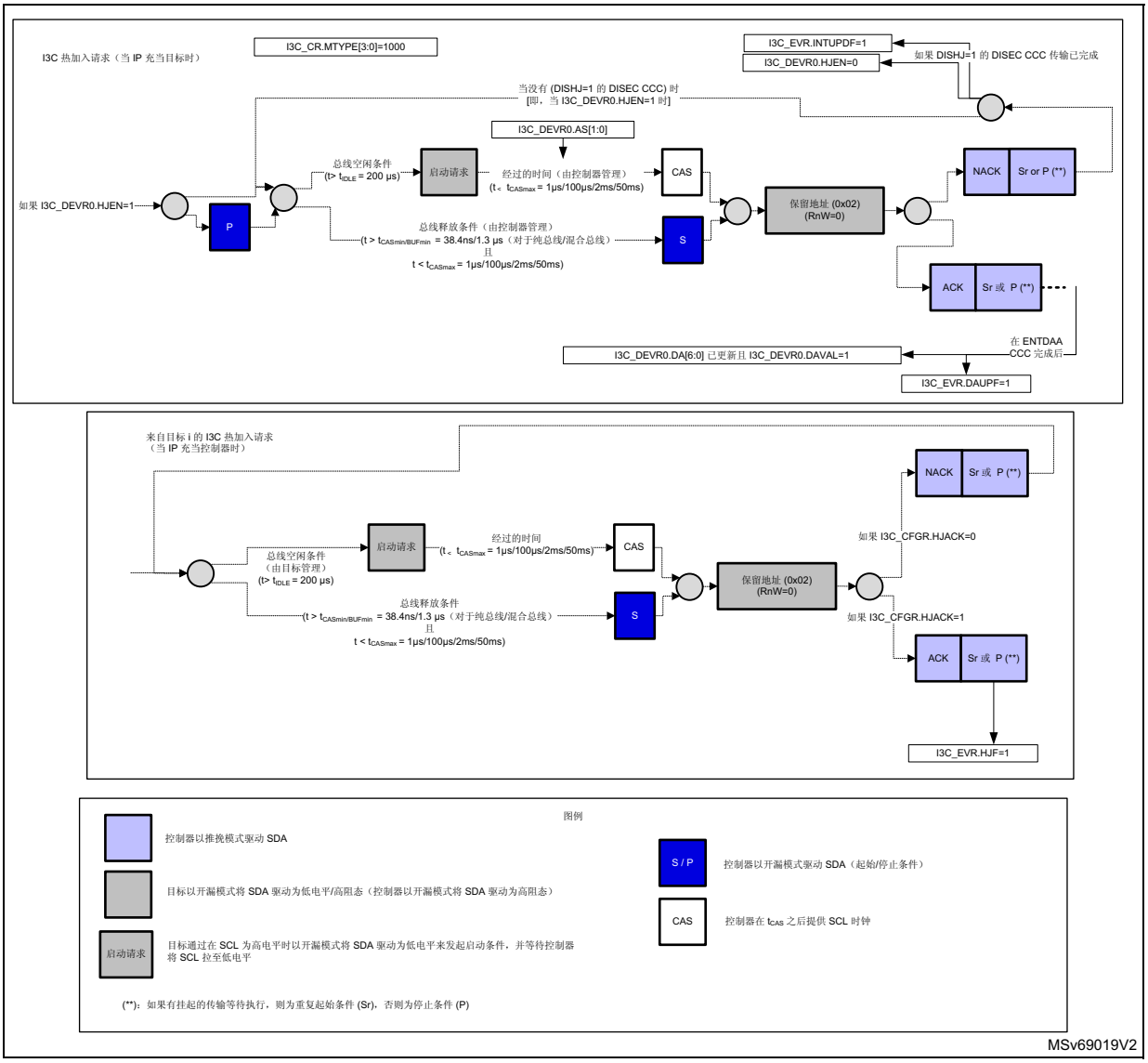


当外设用作控制器时，I3C_IBIDR 寄存器用于接收 IBI 数据有效负载。因此，来自目标的 IBI 请求不得超过 4 字节的数据有效负载。如果需要在该带内中断的上下文中交换更多信息，则控制器软件必须发出私有读控制字。

35.9.14 I3C 热加入请求传输（作为控制器/目标）

图 416 说明了当外设用作目标或控制器时在 I3C 总线上进行的热加入请求传输。

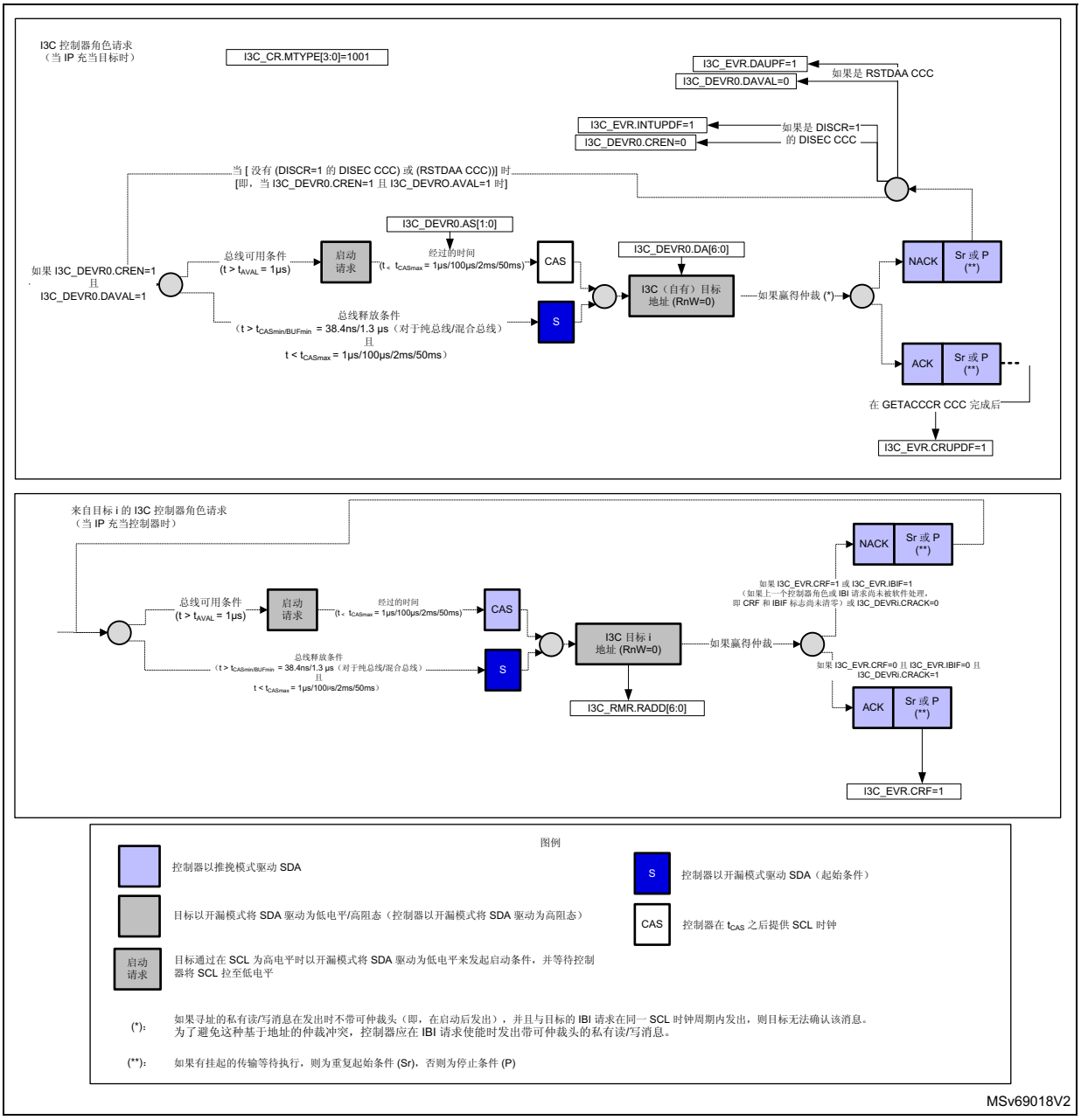
图 416. 热加入请求传输（作为控制器/目标）



35.9.15 I3C 控制器角色请求传输（作为控制器/目标）

图 417 说明了当外设用作目标或控制器时在 I3C 总线上进行的控制器角色请求传输。

图 417. 控制器角色请求传输（作为控制器/目标）



35.10 I3C FIFO 管理（作为控制器）

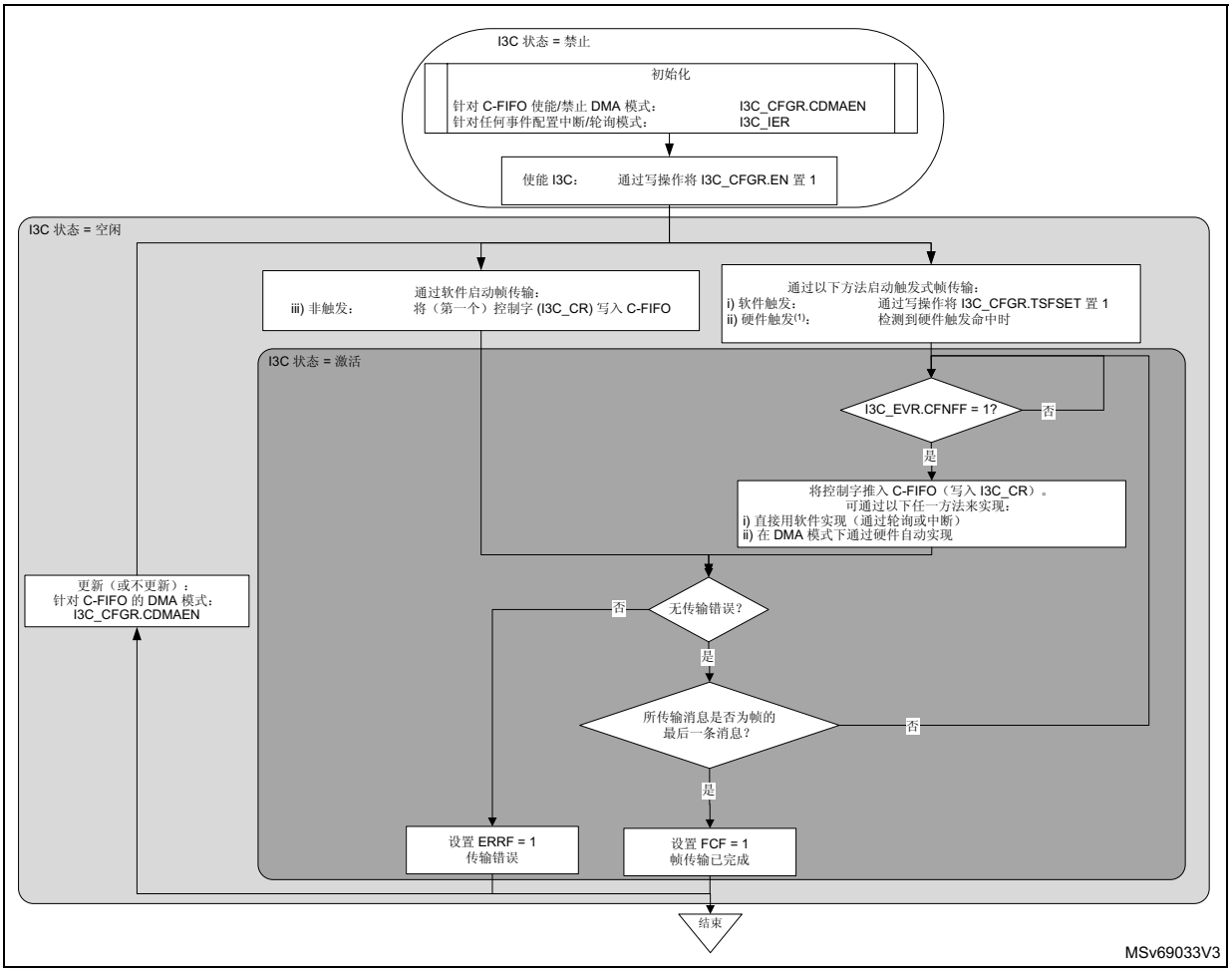
35.10.1 C-FIFO 管理（作为控制器）

用作控制器时（如*第 35.9 节*中的各图所示），可以在以下任何传输期间使用 C-FIFO：

- 广播 CCC（*图 405*、*图 406*、*图 407*）
- 直接读/写 CCC（*图 407*）
  - 命令部分，第一条消息
  - 数据部分，后续的消息
- 私有读/写（*图 412*）
- 传统 I2C 读/写（*图 414*）
- 软件发起的错误恢复（SCL 强制停止，直到下一个报头消息后跟 HDR 退出时序）

图 418 说明了当 I3C 外设用作控制器时，如何管理 C-FIFO 以对 I3C 总线上的控制字进行排队。

图 418. C-FIFO 管理（作为控制器）



1. 该特性取决于具体实现，有可能不可用。请参见*第 35.3.4 节*。

首先，软件必须通过 I3C_CFGR 寄存器中的 CDMAEN 初始化 C-FIFO 管理，方法如下：

- 直接由软件按控制字写入（如果 CDMAEN = 0）：
  - 通过轮询模式（I3C_IER 寄存器中的 CFNFIE = 0）：在显式写入 I3C_CR 寄存器之前等待硬件请求下一个控制字（I3C_IER 寄存器中的 CFNFF = 1）
  - 通过使能的中断通知（如果 CFNFIE = 1）
- 由分配的 DMA 通道写入（如果 CDMAEN = 1），以使能相应的 I3C 外设 DMA 请求 (i3c_tc_dma)：
  - 根据模块级别的配置，DMA 会自动将控制字从其存储器源缓冲区推入/写入 I3C_CR 寄存器，直到帧完成（在帧的最后一条消息后，I3C 总线上会发出停止位），但发生传输错误时除外。

在任何情况下，只要 C-FIFO 为空并且必须发出新控制字的重复起始位，都会报告 C-FIFO 下溢（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 COVR = 1）。如果使能了相应的中断（I3C_IER 寄存器中的 ERRIE = 1），则会生成中断。

当 I3C 外设未处于工作状态时，可以修改 C-FIFO 管理的 DMA 模式配置。

用作控制器时，如果发生传输错误（I3C_EVR 寄存器中的 ERRF = 1），硬件将会自动清空 C-FIFO。

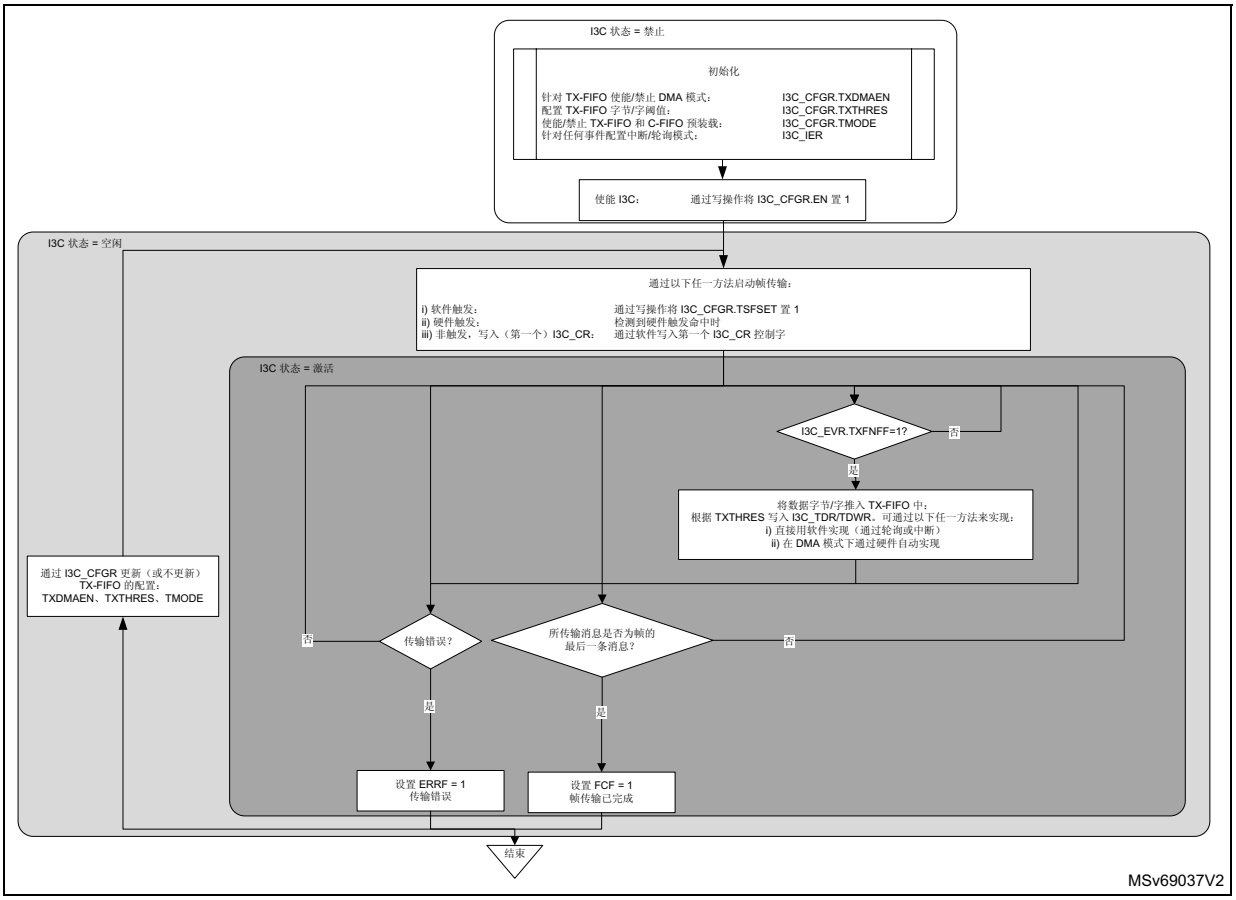
35.10.2 TX-FIFO 管理（作为控制器）

用作控制器时（如*第 35.9 节*中的各图所示），可以在以下任何传输期间使用 TX-FIFO：

- 广播或直接 CCC（包括 ENTDAA 和 RSTACT），如果存在释义字节/子命令字节或数据字节（*图 405*、*图 406*、*图 407*）
- 私有写（*图 412*）
- 传统 I2C 写（*图 414*）

图 419 说明了当 I3C 外设用作控制器时，如何管理 TX-FIFO 以对要在 I3C 总线上发送的数据字节或字进行排队。

图 419. TX-FIFO 管理（作为控制器）



首先，软件必须通过 I3C_CFGR 寄存器中的以下位域初始化 TX-FIFO 管理：

- TXDMAEN：使能/禁止 TX-FIFO 的 DMA 模式
- TXTHRES：将数据字节或字推入 TX-FIFO
- TMODE：使能/禁止 TX-FIFO 和 C-FIFO 预装载

然后，根据 I3C_CFGR 寄存器中的 TXDMAEN 位，通过以下两种方法写入 TX-FIFO：

- 直接由软件按字节/字写入（如果 TXDMAEN = 0）：
  - 通过轮询模式（I3C_IER 寄存器中的 TXFNFIE = 0）：在显式写入 I3C_TDR 或 I3C_TDWR 寄存器之前等待硬件请求下一个数据字节/字（I3C_IER 寄存器中的 TXFNFF = 1），具体取决于 I3C_CGFR 寄存器中的 TXTHRES 位
  - 通过使能的中断通知（如果 TXFNFIE = 1）
- 由分配的 DMA 通道写入（如果 TXDMAEN = 1），以使能相应的 I3C 外设 DMA 请求 (i3c_tx_dma)：
  - 根据 DMA 模块级别的配置，DMA 会自动将数据字节/字从其存储器源缓冲区推入/写入 I3C_TDR 或 I3C_TDWR 寄存器（具体取决于 I3C_CGFR 寄存器中的 TXTHRES 位），直到帧完成（在帧的最后一条消息后，I3C 总线上会发出停止位），但发生传输错误时除外。

I3C 消息以起始位或重复起始位开始，以停止位或重复起始位结束。在消息级别，要发送的最后一个数据字节/字由 I3C_EVR 寄存器中的 TXLASTF = 1 进行指示。当 I3C 帧包含多条消息（由重复起始位分隔）时，软件可以使用该事件来更新相应的指针，使之指向用于存储下一条消息的字节/字的缓冲区。

当报告帧完成时（I3C_EVR 寄存器中的 FCF = 1，且相应的中断已使能），TX-FIFO 为空。

如果 TX-FIFO 为空并且必须在 I3C 总线上发送数据字节，则会报告 TX-FIFO 下溢（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 DOVR = 1）。如果使能了相应的中断（I3C_IER 寄存器中的 ERRIE = 1），则会生成中断。

当 I3C 外设未处于工作状态时，可以修改 TX-FIFO 管理的配置。

用作控制器时，如果发生传输错误 (ERRF = 1)，硬件将会自动清空 TX-FIFO。

无 C-FIFO/TX-FIFO 预装载

根据*表 269* 中的定义，C-FIFO 大小为 2 个字，TX-FIFO 大小为 8 个字节。

当未配置 C-FIFO/TX-FIFO 预装载时（I3C_CFGR 寄存器中的 TMODE = 0），只要第一个控制字写入 C-FIFO，I3C 外设就会立即在 I3C 总线上发出起始位。然后，外设将对 I3C_CR 寄存器进行解码，并要求写入下一个数据字节/字（如果该消息中需要）。只要检测到硬件要求在 I3C 总线上发送下一个控制字（如果必须在 I3C 总线上发出重复起始位或者 C-FIFO 需获取可用空间），就会立即请求将该控制字写入 C-FIFO，直到最后一条消息（I3C_CR 寄存器中的 MEND = 1）。

同理，只要检测到硬件要求在 I3C 总线上发送另一个数据字节/字（如果必须在 I3C 总线上发出重复起始位，TX-FIFO 未满，或者必须在该 I3C 消息期间发送数据字节/字），必须立即将该数据字节/字写入 TX-FIFO。

C-FIFO 和 TX-FIFO 预装载

当配置了 C-FIFO/TX-FIFO 预装载时（I3C_CFGR 寄存器中的 TMODE = 1），在总线上发出起始位之前，I3C 外设需要尽可能地等待 C-FIFO 和 TX-FIFO 装载，具体如下：

- 等待第一个控制字写入 C-FIFO
- 等待数据字节/字写入 TX-FIFO（如果有），如第一个控制字所定义（如果 I3C_CR 寄存器中的 RNW = 0 且 DCNT[15:0] = 0），最多可写入 TX-FIFO 大小
- 如果 TX-FIFO 未满并且第一个控制字不是帧的最后一条消息（I3C_CR 寄存器中的 MEND = 0）：
  - 等待第二个控制字写入 C-FIFO，随即 C-FIFO 装满。
  - 如果 TX-FIFO 未满，则等待数据字节/字写入 TX-FIFO（如果有），如第二个控制字所定义（I3C_CR 寄存器中的 RNW = 0 且 DCNT[15:0]），最多可写入 TX-FIFO 大小。

然后，只要检测到硬件要求在 I3C 总线上发送下一个控制字（如果必须在 I3C 总线上发出重复起始位），就会立即请求将该控制字写入 C-FIFO，直到最后一条消息（I3C_CR 寄存器中的 MEND = 1）。

同理，只要检测到硬件要求在 I3C 总线上发送下一个数据字节/字（如果必须在 I3C 总线上发出重复起始位，TX-FIFO 未满，或者将在该 I3C 消息期间发送数据字节/字），必须立即将该数据字节/字写入 TX-FIFO。

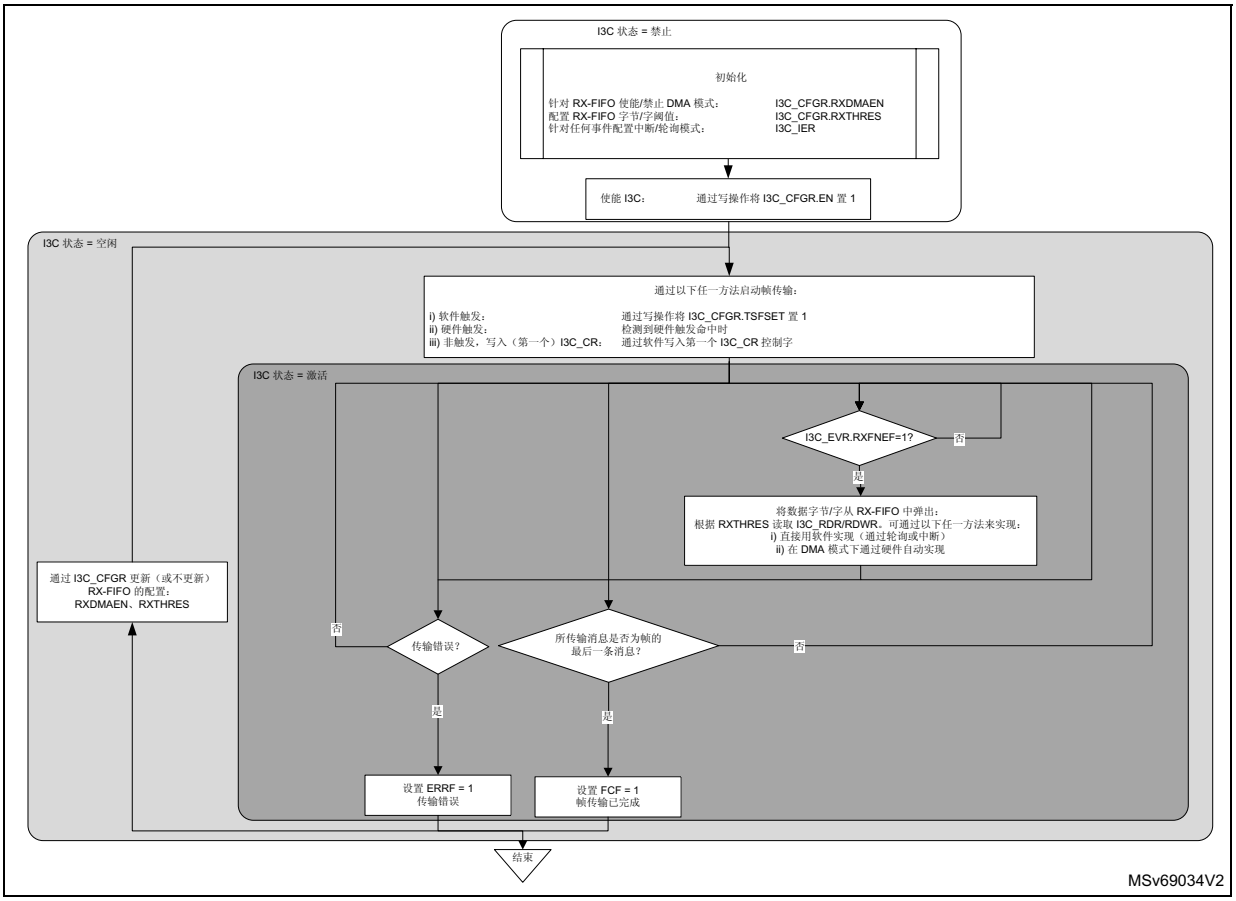
35.10.3 RX-FIFO 管理（作为控制器）

用作控制器时（如*第 35.9 节*中的各图所示），可以在以下任何传输期间使用 RX-FIFO：

- 广播 ENTDAA CCC（*图 406*）
- 直接 CCC 读（*图 405*），包括直接 RSTACT CCC 读（*图 407*）
- 私有读（*图 412*）
- 传统 I2C 读（*图 414*）

图 420 说明了当 I3C 外设用作控制器时，如何管理 RX-FIFO 以对从 I3C 总线接收到的数据字节/字进行排队和弹出。

图 420. RX-FIFO 管理（作为控制器）



首先，软件必须通过 I3C_CFGR 寄存器的以下位域初始化 RX-FIFO 管理：

- RXDMAEN：使能/禁止 RX-FIFO 的 DMA 模式
- RXTHRES：从 RX-FIFO 弹出数据字节或字

然后，根据 RXDMAEN 位读取 RX-FIFO：

- 直接由软件按字节/字读取 (RXDMAEN = 0)：
  - 通过轮询模式（I3C_IER 寄存器中的 RXFNEIE = 0）：在显式读取 I3C_RDR 或 I3C_RDWR 寄存器之前等待硬件请求下一个数据字节/字（I3C_IER 寄存器中的 RXFNEF = 1），具体取决于 I3C_CGFR 寄存器中的 RXTHRES 位
  - 通过使能的中断通知（如果 I3C_IER 寄存器中的 RXFNEIE = 1）
- 由分配的 DMA 通道读取（如果 RXDMAEN = 1），以使能相应的 I3C 外设 DMA 请求 (i3c_rx_dma)：
  - 根据 DMA 模块级别的配置，DMA 会自动从 I3C_RDR 或 I3C_RDWR 寄存器弹出/读取数据字节/字（具体取决于 RXTHRES 位），并将它们写入其存储器目标缓冲区，直到帧完成（在帧的最后一条消息后，I3C 总线上会发出停止位），但发生传输错误时除外。

I3C 消息以起始位或重复起始位开始，以停止位或重复起始位结束。在消息级别，从 I3C 总线接收到的最后一个数据字节/字由 I3C_EVR 寄存器中的 IRXLASTF = 1 进行指示。当 I3C 帧包含多条消息（由重复起始位分隔）时，软件可以使用该事件来更新相应的指针，使之指向用于存储下一条消息的数据字节/字的缓冲区。

如果 RX-FIFO 已满并且在 I3C 总线上接收到数据字节，则会报告 RX-FIFO 上溢（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 DOVR = 1）。如果使能了相应的中断（I3C_IER 寄存器中的 ERRIE = 1），则会生成中断。

当 I3C 外设未处于工作状态时，可以修改 RX-FIFO 管理的配置。

目标提前终止读传输

被寻址的目标可以提前完成（也称为提前结束）私有读消息。

- 如果 RXDMAEN = 1：
  - 软件必须针对 DMA 请求 i3c_rx_dma 分配一个支持外设提前终止的 DMA 通道 x（请参见 DMA 实现部分）。
  - 软件必须对 DMA 通道 x 进行配置，以使能 DMA 外设流控制模式（通过 DMA_CxTR2 寄存器中的 PFREQ = 1），从而能够提前或不提前完成 DMA 块传输。
  - 然后，在块传输完成时，软件可以读取 DMA_Cx 和/或 DMA_CxSAR 寄存器中的 BR1.BNDT[15:0] 以获取 DMA 传输的有效字节数。
- 如果 RXDMAEN = 0：
  - 在消息级别，从 I3C 总线接收到的最后一个数据字节/字由 I3C_EVR 寄存器中的 RXFNEF = 1 和 RXLASTF = 1 进行指示。

在任何情况下，如果禁止了 S-FIFO（I3C_CFGR 寄存器中的 SMODE = 0），将通过 I3C_EVR 寄存器中的 RXTGTENDF = 1 和相应的中断（如果使能）通知软件发生读传输提前终止。然后，软件可以读取状态寄存器 I3C_SR 来检查与最后一条消息相关的信息，并获取提前结束的读传输中接收到的数据字节数（I3C_SR 寄存器中的 XDCNT[15:0]）。

在任何情况下，如果使能了 S-FIFO（I3C_CFGR 寄存器中的 SMODE = 1），软件或 DMA（具体取决于 I3C_CFGR 寄存器中的 SDMAEN）必须针对每一条消息读取一次状态寄存器 I3C_SR。提前结束的读消息中接收到的有效数据字节数由 I3C_SR 寄存器中的 XDCNT[15:0]（随后 ABT = 1）报告。

更多信息，请参见 *I3C 状态寄存器 (I3C_SR)* 和*第 35.10.4 节*。

35.10.4 S-FIFO 管理（作为控制器）

用作控制器时，软件可以使用 S-FIFO 为每一条传输的消息读取 *I3C 状态寄存器 (I3C_SR)*。

禁止 S-FIFO 时读取状态寄存器

如果 I3C_CFGR 寄存器中的 SMODE = 0，则 S-FIFO 处于禁止状态，此时可以像读取普通寄存器一样读取状态寄存器：

- 传输新消息时，寄存器的内容由硬件覆盖
- I3C_SR 包含传输的最后一条消息的状态
- 如果未读取状态寄存器，SCL 时钟不会停顿。

如果 SMODE = 0，对于目标提前结束私有读传输的特定情况：

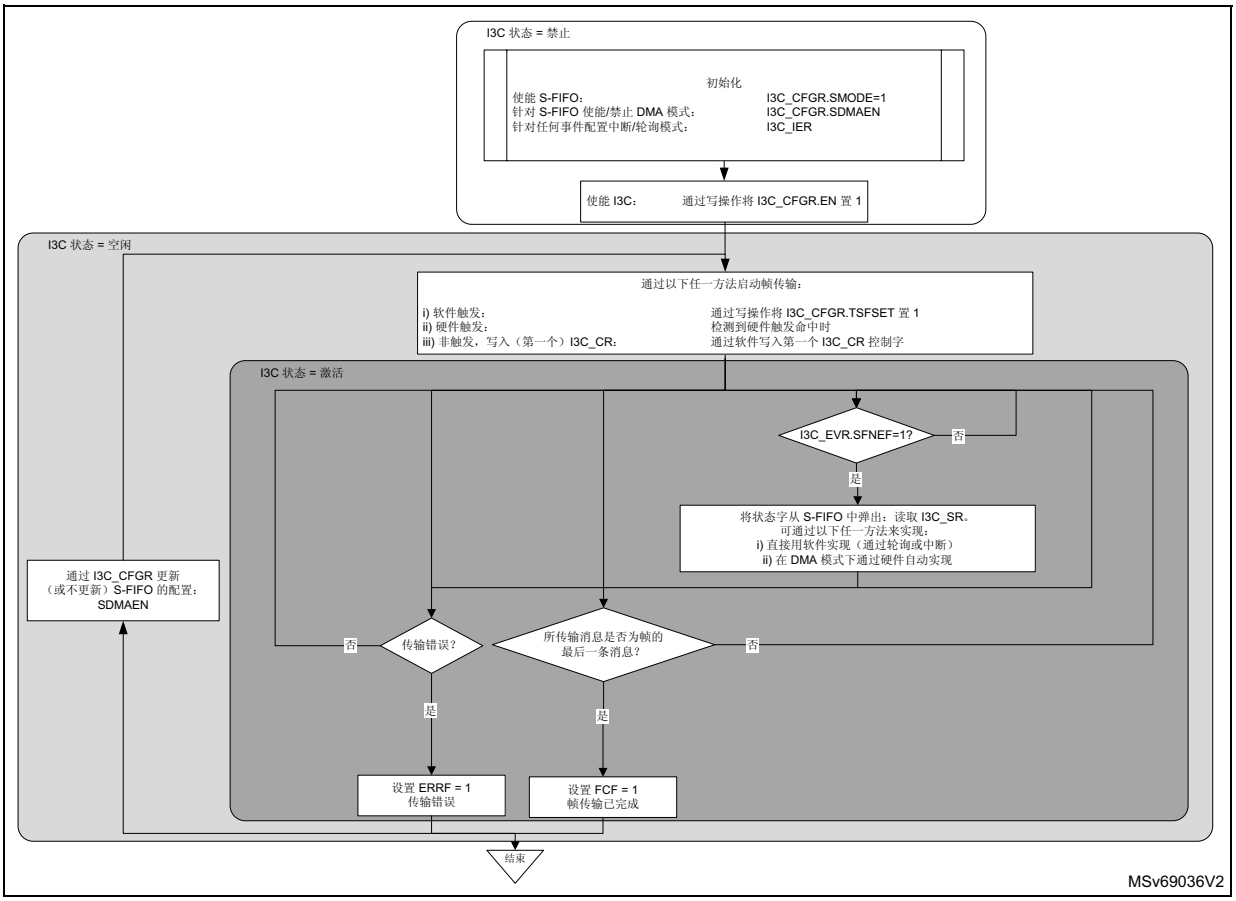
- 通过 I3C_EVR 寄存器中的标志 RXTGTENDF = 1 和相应的中断（如果使能）通知软件。
- 然后，在软件通过写操作将 I3C_CEVR 寄存器中的 CRXTGTENDF 置 1 以清零事件标志之前：
  - I3C 总线上不再接收数据字节并由硬件写入 I3C_RDR/I3C_RDWR 寄存器
  - 不能更新 I3C_SR
  - 使 SCL 时钟停顿（如果需要）

通常，当 I3C_EVR 寄存器中的 FCF = 1、ERRF = 1 或 RXTGTENDF = 1 时，可以读取 I3C_SR。在目标提前终止读传输之后，可以读取 I3C_SR 寄存器中的 XDCNT[15:0] 以获取私有读传输中接收到的有效数据字节数（请参见*第 35.10.3 节*）。

使能 S-FIFO 时读取状态寄存器

如果 I3C_CFGR 寄存器中的 SMODE = 1，则 S-FIFO 处于使能状态。*图 421* 说明了当 I3C 外设用作控制器时，如何管理 S-FIFO 以对 I3C 总线上执行的每条消息的状态字进行排队和弹出。

图 421. S-FIFO 管理（作为控制器）



首先，软件必须通过 I3C_CFGR 寄存器中的 SDMAEN 位域（使能/禁止 S-FIFO 的 DMA 模式）初始化 S-FIFO 管理。然后，根据 SDMAEN 位读取 S-FIFO：

- 直接由软件读取（如果 SDMAEN = 0）：
  - 通过轮询模式（I3C_IER 寄存器中的 SFNEIE = 0）：在显式读取 I3C_SR 寄存器之前等待硬件请求下一个状态字（I3C_IER 寄存器中的 SFNEF = 1）
  - 通过使能的中断通知 (SFNEIE = 1)
- 由分配的 DMA 通道读取（如果 SDMAEN = 1），以使能相应的 I3C 外设 DMA 请求 (i3c_rs_dma)：
  - 根据 DMA 模块级别的配置，DMA 会自动从 I3C_SR 寄存器弹出/读取状态字并将它们写入其存储器目标缓冲区，直到帧完成（在帧的最后一条消息后，I3C 总线上会发出停止位），但发生传输错误时除外。

必须读取每一个消息状态，否则当 S-FIFO 已满并且必须写入下一个消息状态时，硬件会将上溢错误标志置 1（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 COVR = 1）。如果使能了相应的中断（I3C_IER 寄存器中的 ERRIE = 1），则会生成中断。

只有在 S-FIFO 为空时才会报告帧完成（I3C_EVR 寄存器中的 FCF = 1）。

当 I3C 外设未处于工作状态时，可以修改 S-FIFO 管理的配置。

35.11 I3C FIFO 管理（作为目标）

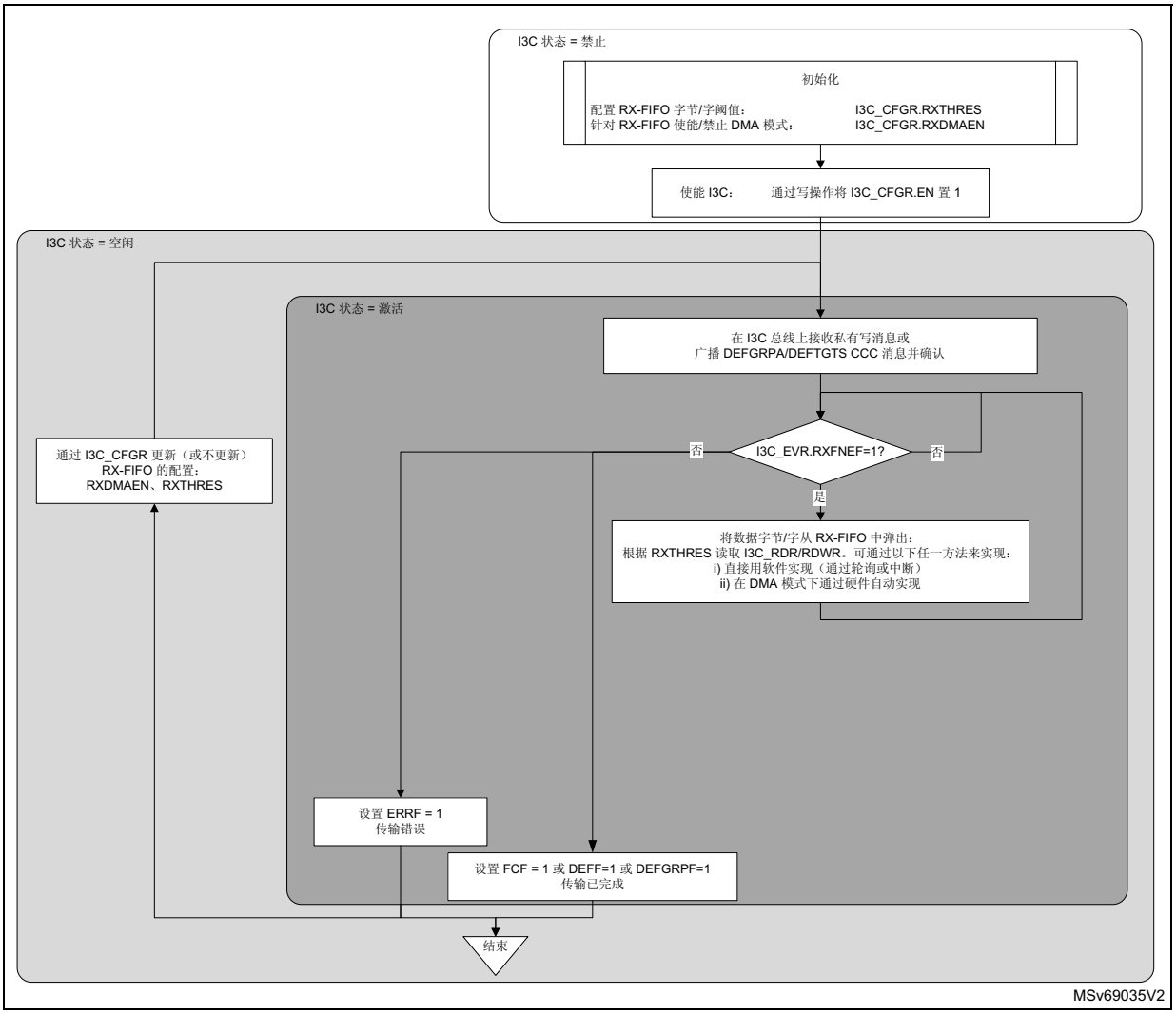
35.11.1 RX-FIFO 管理（作为目标）

用作目标时（如*第 35.9 节*中的各图所示），可以在以下任何已接收和确认的传输期间使用 RX-FIFO：

- 广播 DEFTGTS CCC（*图 410*）
- 广播 DEFGRPA CCC（*图 411*）
- 私有写（*图 413*）

图 422 说明了当 I3C 外设用作目标时，如何管理 RX-FIFO 以对从 I3C 总线接收到的数据字节/字进行排队和弹出。

图 422. RX-FIFO 管理（作为目标）



首先，软件必须通过以下 I3C_CFGR 寄存器位域初始化 RX-FIFO 管理：

- RXDMAEN：使能/禁止 RX-FIFO 的 DMA 模式
- RXTHRES：从 RX-FIFO 弹出数据字节或字

然后，根据 RXDMAEN 位读取 RX-FIFO：

- 直接由软件按字节/字读取（如果 RXDMAEN = 0）：
  - 通过轮询模式（I3C_IER 寄存器中的 RXFNEIE = 0）：在显式读取 I3C_RDR 或 I3C_RDWR 寄存器之前等待硬件请求下一个数据字节/字（I3C_IER 寄存器中的 RXFNEF = 1），具体是字节还是字取决于 I3C_CGFR 寄存器中的 RXTHRES
  - 通过使能的中断通知（如果 RXFNEIE = 1）
- 由分配的 DMA 通道读取（如果 RXDMAEN = 1），以使能相应的 I3C 外设 DMA 请求 (i3c_rx_dma)：
  - 根据 DMA 模块级别的配置，DMA 会自动从 I3C_RDR 或 I3C_RDWR 寄存器弹出/读取数据字节/字（具体取决于 RXTHRES），并将它们写入其存储器目标缓冲区，直到传输完成（I3C_EVR 寄存器中的 FCF = 1（私有写）、GRPF = 1 (DEFGRPA CCC) 或 DEFF = 1 (DEFTGTS CCC)），但发生传输错误时除外。

如果 RX-FIFO 已满并且在 I3C 总线上接收到新的数据字节，则会报告 RX-FIFO 上溢（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 DOVR = 1），并生成相应的中断（如果使能）。

当传输完成时（I3C_EVR 寄存器中的 FCF = 1、GRPF = 1 或 DEFF = 1）：

- RX-FIFO 为空
- 在软件尚未处理完与已完成的私有写/DEFTGTS/DEFGRPA 传输相对应的 RX 数据缓冲区（也就是软件还未通过写操作将 I3C_CEVR 寄存器中的 CFCF/CDEFF/CGRPF 置 1 以清零相应的标志）之前，如果接收到下一个私有写/DEFTGTS CCC/DEFGRPA CCC，并且硬件必须将数据字节写入 RX-FIFO，则会报告 RX-FIFO 上溢错误（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 DOVR = 1）并生成相应的中断（如果使能）。

当 I3C 外设未处于工作状态时，可以修改 RX-FIFO 管理的配置。

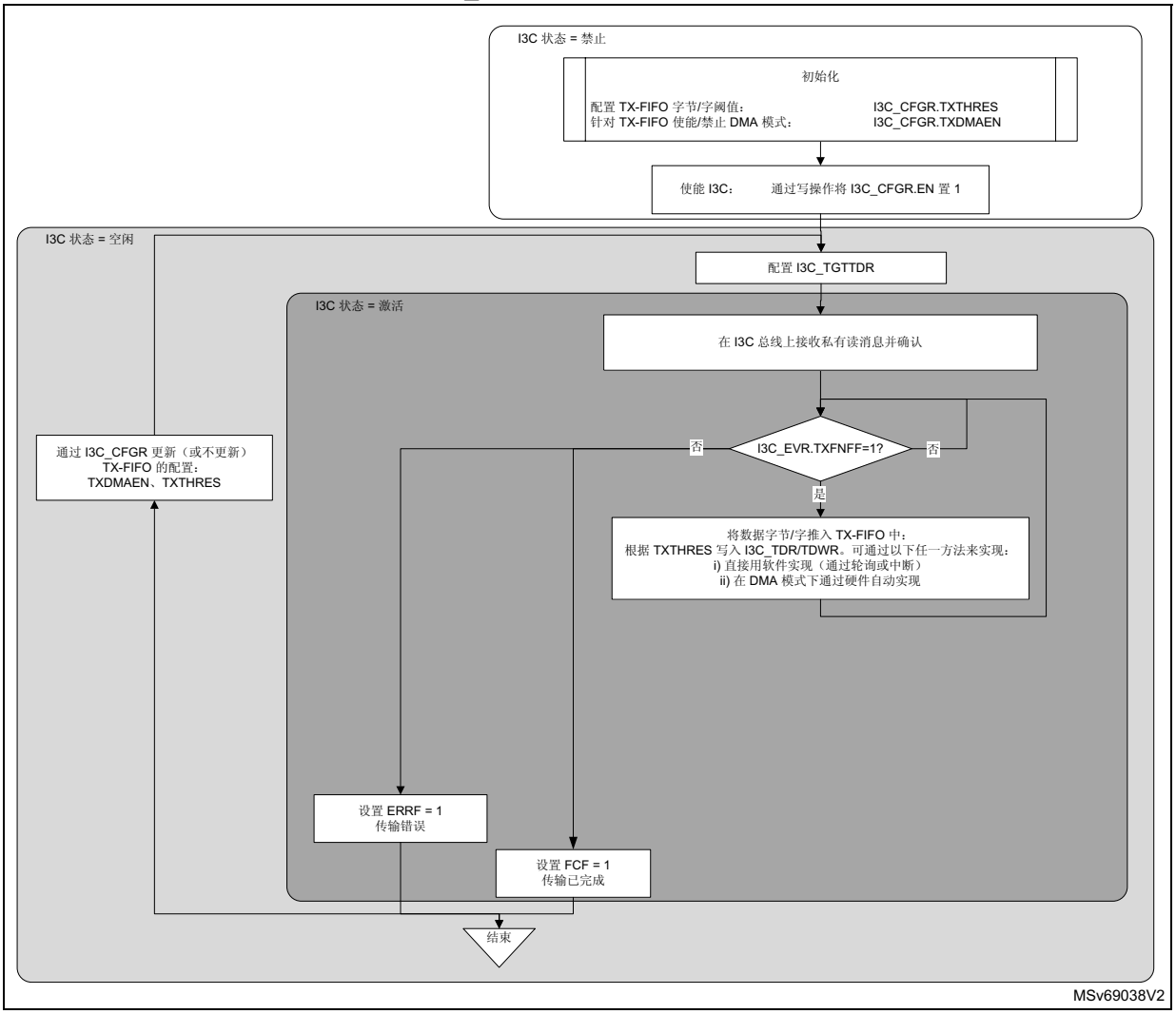
35.11.2 TX-FIFO 管理（作为目标）

主要方案

用作目标时（如*第 35.9 节*中的各图所示），只能在私有读传输（*图 413*）期间使用 TX-FIFO。

图 423 说明了当 I3C 外设用作目标时，如何管理 TX-FIFO 以对要在 I3C 总线上发送的数据字节或字进行排队和推入。

图 423. 通过 I3C_TGTTDR 管理 TX-FIFO（作为目标）



首先，软件必须初始化 TX-FIFO 管理并写入 I3C_CFGR 中的以下位

- TXDMAEN：使能/禁止 TX-FIFO 的 DMA 模式
- TXTHRES：将数据字节或字推入 TX-FIFO

然后，在 I3C 总线上接收私有读传输之前，软件必须配置 *I3C 目标发送配置寄存器 (I3C_TGTTDR)* 以向 TX-FIFO 预装载一定数量的数据字节（在单次访问中写入 TGTTDCNT[15:0] 和 PRELOAD = 1），从而准备好在 I3C 总线上发送来自目标的数据字节：

- 如果 PRELOAD = 1 且 TGTTDCNT[15:0] > TX-FIFO 大小，则 TX-FIFO 会先预装载至 FIFO 大小。
- 如果 PRELOAD = 1 且 TGTTDCNT[15:0] ≤ TX-FIFO 大小，则 TX-FIFO 会预装载至 TGTTDCNT[15:0]。

注意： *TX-FIFO 大小为 8 字节（请参见表 269）。*

根据 TXDMAEN 位预装载 TX-FIFO：

- 直接由软件按字节/字预装载 (TXDMAEN = 0)：
  - 通过轮询模式（I3C_IER 寄存器中的 TXFNFIE = 0）：在显式写入 I3C_TDR 或 I3C_TDWR 寄存器之前等待硬件请求下一个数据字节/字（I3C_IER 寄存器中的 TXFNFF = 1），具体是字节还是字取决于 I3C_CGFR 寄存器中的 TXTHRES
  - 通过使能的中断通知（如果 TXFNFIE = 1）
- 由分配的 DMA 通道预装载 (TXDMAEN = 1)，以使能相应的 I3C 外设 DMA 请求 (i3c_tx_dma)：
  - 根据 DMA 模块级别的配置，DMA 会自动将数据字节/字从其存储器源缓冲区推入/写入 I3C_TDR 或 I3C_TDWR 寄存器（具体取决于 TXTHRES），直到传输完成（IEC_EVR 寄存器中的 FCF = 1），但发生传输错误时除外。

然后，当 I3C_TGTTDR 寄存器的 TGTTDCNT[15:0] 中还有其余数据要装载到 TX-FIFO 以继续私有读传输（PRELOAD 置 1 且 TGTTDCNT[15:0] > TX-FIFO 大小）时，如果控制器尚未完成私有读传输，则采用同样的方式进行预装载（直接通过软件或通过分配的 DMA 通道）：

- 如果 TXTHRES = 0：当在 I3C 总线上发送一个字节时，将下一个字节预装载到 TX-FIFO
- 如果 TXTHRES = 1：当在 I3C 总线上发送四个字节时，将下一个字预装载到 TX-FIFO。

当目标或控制器先终止数据字节传输时，私有读传输完成（I3C_EVR 寄存器中的 FCF = 1）。

传输完成后，软件可以：

- 读取 I3C_SR 寄存器中的 XDCNT[15:0]：发送的有效数据字节数
- 读取 I3C_TGTTDR 寄存器中的 TGTTDCNT[15:0]：I3C 总线上待装载和发送的剩余字节数
- 清空 TX-FIFO：是否通过写操作将 I3C_CFGR 寄存器中的 TXFLUSH 置 1 并继续进行另一次私有读传输，取决于用户应用

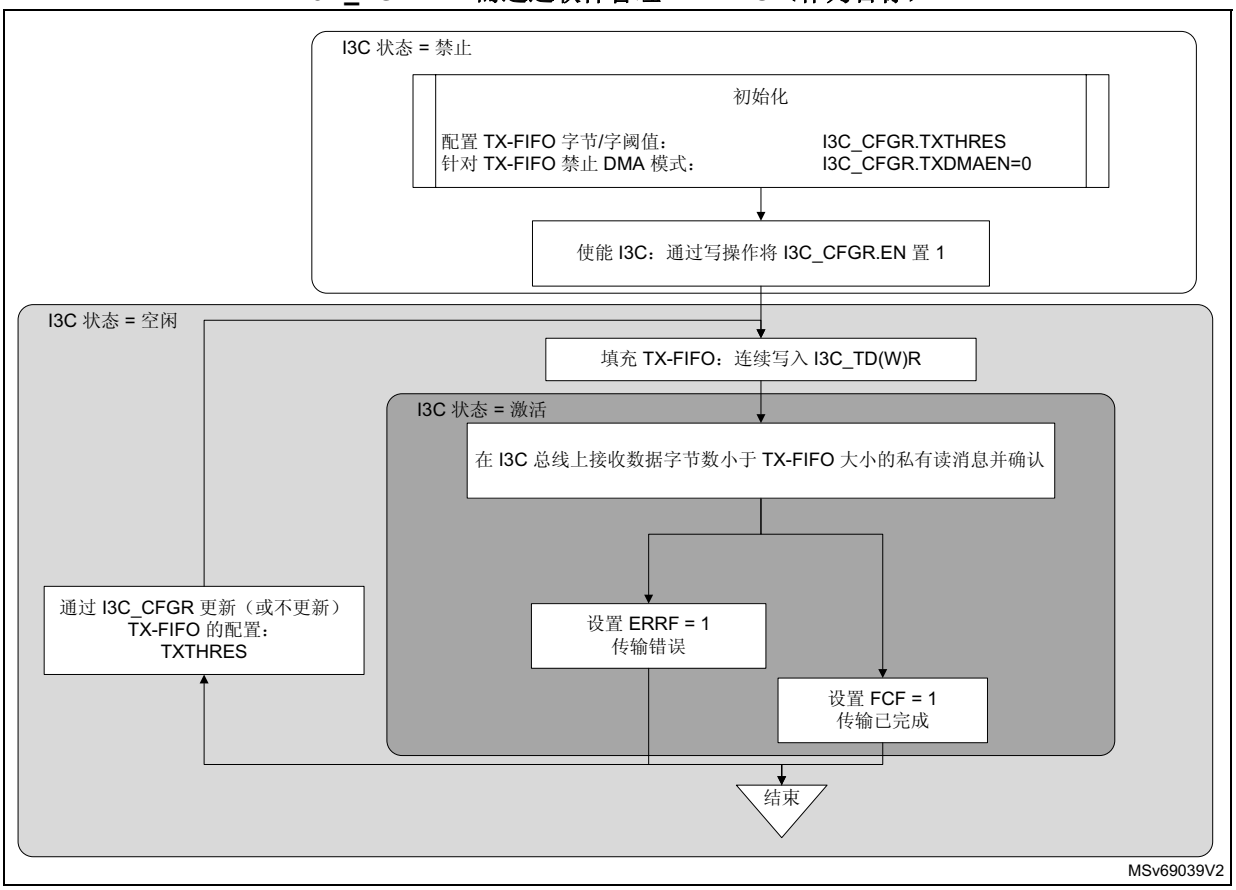
如果 TX-FIFO 为空并且必须在 I3C 总线上发送数据字节，则会报告 TX-FIFO 下溢（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 DOVR = 1）。如果使能了相应的中断（I3C_EVR 寄存器中的 ERRIE = 1），则会生成中断。

当 I3C 外设未处于工作状态时，可以修改 TX-FIFO 管理的配置。

如果字节数小于 TX-FIFO 大小，可以选择不使用 I3C_TGTTDR

除了使用 I3C_TGTTDR 寄存器之外，如*图 424* 所示，当不使用 DMA (TDMAEN = 0) 时，如果 I3C 总线上要读取的数据字节数小于 TX-FIFO 大小，软件可通过连续写入 I3C_TDR 或 I3C_I3C_TDWR（具体取决于 TXTHRES），直接使用自身直接读取的字节数来准备并填充 TX-FIFO。

图 424. 如果读取的字节数小于 TX-FIFO 大小，则不使用 I3C_TGTTDR 而通过软件管理 TX-FIFO（作为目标）



35.12 I3C 错误管理

35.12.1 控制器错误管理

表 278 列举了当 I3C 外设用作控制器时的 I3C 总线错误条件，并针对每个错误条件提供了相应的检测、操作和报告。

表 278. I3C 控制器错误管理

错误类型	说明	错误检测	控制器操作	报告的错误(1)
CE0	非法格式化 CCC（例如返回的数据字节较少）	直接 CCC 读传输时目标提前结束读取数据(2)	硬件发出停止位。	ERRF = 1、PERR = 1 且 CODERR[3:0] = 0000

表 278. I3C 控制器错误管理（续）

错误类型	说明	错误检测	控制器操作	报告的错误(1)
CE1	监视错误	在传统 I²C 读传输结束时检测到不正确的 ACK	硬件使 SCL 保持运行状态并持续九个时钟周期，可以交换另一个字节，然后再次发出 NACK，后跟停止位。	ERRF = 1、PERR = 1 且 CODERR[3:0] = 0001
		I3C SDR 读传输结束时无法生成重复起始位或停止位	SCL 保持运行状态。软件可通过向 I3C_CR 寄存器写入控制字 MTYPE[3:0] = 0000（报头消息）来停止 SCL，然后通常必须等待至少 150 µs 才能发出另一条消息。	
		检测到 CE1 错误后，无法生成起始位		
CE2	对广播地址无响应 (0b111_1110)	在消息发送过程中（升级故障或复位时序消息除外），检测到报头 (0b111_1110 + RnW = 0) 被 NACK	硬件发出 HDR 退出时序，后跟停止位。	ERRF = 1、PERR = 1 且 CODERR[3:0] = 0010
CE3	控制器切换失败	新控制器在 SDA 变为低电平后（通过测试报头或来自目标的启动请求启动后）以及在其工作状态定义的延迟时间结束后尚未将 SCL 驱动为低电平	硬件发出起始位 + 0b111_1110 + RnW = 0，后跟来自目标的 ACK/NACK，最后是停止位。	ERRF = 1、PERR = 1 且 CODERR[3:0] = 0011
-	在 GETACCCR CCC 中返回的带奇偶校验位的 7 位目标地址不正确	检测到不正确的动态地址和/或奇偶校验位	硬件取消 GETACCCR CCC，方法是发出重新起始位 + 0b111_1110 + RnW = 0，后跟来自目标的 ACK/NACK，最后是停止位。	ERRF = 1 且 DERR = 1
-	被寻址的目标首次在直接 CCC 读传输中被 NACK	检测到 NACK	硬件执行单次重试，方法是发出重复起始位 + 7 位相同目标地址 + RnW = 1。	-
-	被寻址的目标第二次在直接 CCC 写传输、I3C 私有读/写传输、传统 I²C 传输或直接 CCC 读传输中被 NACK	检测到 NACK	硬件发出停止位。	ERRF = 1 且 ANACK = 1
-	分配/发送的带奇偶校验位的动态地址首次在 ENTDAA CCC 中被 NACK	检测到 NACK	硬件执行单次重试分配循环，方法是发出重复起始位 + 0b111_1110 + RnW = 1，后跟来自（最高优先级）目标的 ACK 和 8 字节读数据，后面是分配的地址 + 奇偶校验位。	-
-	分配/发送的带奇偶校验位的动态地址第二次在 ENTDAA CCC 中被 NACK	检测到 NACK	硬件发出停止位。	ERRF = 1 且 DNACK = 1
-	写数据在传统 I2C 写传输中被 NACK			

表 278. I3C 控制器错误管理（续）

错误类型	说明	错误检测	控制器操作	报告的错误(1)
-	相对于 I3C 总线时序，未及时写入/读取控制字、状态字、发送数据或读数据	SCL 停顿超时	硬件发出停止位。	ERRF = 1 且（COVR = 1 或 DOVR = 1）

1. I3C_EVR 寄存器中的 ERRF，I3C_SER 寄存器中的 PERR、CODERR[3:0]、DERR、ANACK、DNACK、COVR 和 DOVR。
2. MIPI v1.1：在 GETCAPS CCC 中，接收到的数据字节数可以是 2、3 或 4。但是，符合 MIPI v1.0 规范的目标只能返回第一个字节（之前名为 GETHDRCAP CCC）。因此，如果字节数小于 4，则不会生成 CE0。
在 GETMXDS CCC 中，接收到的数据字节数可以是 2 或 5。如果字节数小于 5，则不会生成 CE0。

35.12.2 目标错误管理

表 279 列举了当 I3C 外设用作目标时的 I3C 总线错误条件，并针对每个错误条件提供了相应的检测、操作和报告。

表 279. I3C 目标错误管理

错误类型	说明	错误检测	下一步操作（由控制器发出时）	报告的错误(1)
TE0	DAA 分配后的广播地址 (0b111_1110+RnW = 0) 无效或 7 位动态地址 + RnW = 1 无效	在起始位或重复起始位后检测到禁止使用的地址	硬件等待 HDR 退出时序	ERRF = 1、PERR = 1 且 ODERR[3:0] = 1000
TE1	CCC 代码	检测到 CCC 代码中有奇偶校验错误		ERRF = 1、PERR = 1 且 ODERR[3:0] = 1001
TE2	写数据	在 I3C 私有写消息中检测到写数据字节中有奇偶校验错误	硬件等待停止位或重复起始位	ERRF = 1、PERR = 1 且 ODERR[3:0] = 1010
TE3	动态地址仲裁期间分配的地址	在 ENTDAA CCC 中检测到分配的动态地址中有奇偶校验错误		ERRF = 1、PERR = 1 且 ODERR[3:0] = 1011
TE4	动态地址仲裁期间 Sr 位后缺少 0b111_1110 + RnW = 1	检测到动态地址仲裁期间 Sr 位后缺少 {0b111_1110 + RnW = 1}		ERRF = 1、PERR = 1 且 ODERR[3:0] = 1100
TE5	检测到 CCC 后的事务	在 CCC 直接读/写传输中，检测到无效的 CCC 方向（相对于地址阶段提供的方向）		ERRF = 1、PERR = 1 且 ODERR[3:0] = 1101
TE6	监视错误	在 SDR 数据读传输期间（在直接 CCC 读、私有读或 IBI 传输期间）检测到以异常值驱动 SDA	硬件释放 SDA，然后等待停止位或重复起始位	ERRF = 1、PERR = 1 且 ODERR[3:0] = 1110
-	SDR 数据读传输期间监视 SCL	在 SDR 数据读传输期间（在直接 CCC 读、私有读或 IBI 传输期间）检测到 SCL 的稳定时间超过 125 µs		ERRF = 1 且 STALL = 1

表 279. I3C 目标错误管理（续）

错误类型	说明	错误检测	下一步操作（由控制器发出时）	报告的错误(1)
-	待发送的数据未及时写入/不可用（相对于 I3C 总线时序）	在私有读传输中，软件或 DMA 未及时写入待发送的数据	硬件检测到停止位或重复起始位	ERRF = 1 且 DOVR = 1
-	无法及时读取接收到的数据（相对于 I3C 总线时序）	在私有写、DEFTGTS 或 DEFGRPA CCC 中，软件或 DMA 未及时读取接收到的数据		

1. I3C_EVR 寄存器中的 ERRF，I3C_SER 寄存器中的 PERR、ODERR[3:0]、STALL 和 DOVR。

35.13 I3C 从低功耗模式唤醒

35.13.1 从停止模式唤醒

用户必须先配置复位和时钟控制器 (RCC) 以设置到 I3C 外设的时钟数据路径：为 I3C 内核时钟选择源振荡器，为 I3C APB 时钟选择源振荡器，并设置时钟频率。通过 RCC 进行初始化时，用户必须使能给定 I3C 外设的 I3C 时钟（分别用于运行/睡眠模式和停止模式）才能正常工作。有关 RCC 编程的更多详细信息，请参见*第 10 章：复位和时钟控制 (RCC)*。

无论器件处于运行、睡眠还是停止模式，I3C 硬件都会自动管理其自身的时钟门控，并分别为其内核时钟和 APB 时钟生成传输至 RCC 的时钟请求输出信号。

当进入停止模式时（对 V_{CORE} 域供电），默认情况下，任何时钟振荡器都处于禁止状态，并且在任何情况下，系统时钟和外设时钟都不会在该域中运行。

作为控制器：通过 IBI（不含 MDB）、热加入请求或控制器角色请求唤醒

当外设用作控制器时，在产品进入低功耗模式之前，软件必须发出 ENTASx CCC（通常 x = 0、1、2 或 3）以通知目标在分别定义为 1 µs、100 µs、2 ms 或 50 ms 的时间间隔结束之前，I3C 控制器不会退出空闲状态，也不会在 I3C 总线上进行通信。该时间间隔定义了控制器在起始位后将 SCL 总线时钟设置为低电平并运行的 T_{CAS} 延时。更具体地说，对于停止模式，CCC 必须限制为 ENTASx（值 x = 1、2 或 3）。

I3C 从停止模式唤醒的总体方案如下：

1. 首先，I3C 外设向系统请求其内核时钟：
   - 在 I3C 总线上检测到起始位时（检测到 SDA 线被驱动为低电平，而 SCL 为高电平）。
   - 作为控制器，由外部 I3C 目标器件发出热加入/带内中断/控制器角色请求。只要提供内核时钟并且该时钟正常运行，I3C 硬件就会使用内部定时器等待相应的 T_{CAS} 时间结束，然后将 SCL 驱动为低电平并继续切换 SCL，以便让目标在 I3C 总线上执行其热加入/带内中断/控制器角色请求。
2. 系统使能 I3C 内核时钟的源振荡器，时钟随即准备就绪（如果是 HSI，则在几微秒后）。用户可以在停止模式下使源振荡器保持开启状态来缩短该启动延迟，但代价是增加功耗。
   - I3C 外设保持内核时钟请求，直到 I3C 总线上生成停止位。

3. 之后，内核时钟继续运行，作为控制器的 I3C 外设向系统请求其 APB 时钟：
   – 在接收到的 IBI 请求的 ACK 地址上（*图 415*），I3C 外设会保持 APB 时钟请求，直到 IBIF 被清零。

     如果 IBI 不含 MDB：即使尚未提供 APB 时钟，I3C 总线上通常也会生成停止位。
   – 在接收到的控制器角色请求的 ACK 地址上（*图 416*），I3C 外设会保持 APB 时钟请求，直到 CRF 被清零。
   – 在热加入请求的 ACK 地址上（*图 417*），I3C 外设会保持 APB 时钟请求，直到 HJF 被清零。
4. 系统收到 I3C APB 时钟请求的通知，PWR 模块的电源管理单元随即唤醒。
5. 如果调压器必须提高运行模式下的电压，则可能需要额外的延时。
6. 系统使能用于驱动 APB 时钟的系统时钟。如果为系统和 APB 时钟选择的振荡器源与用于驱动 I3C 内核时钟的振荡器源不同，则会出现额外的延时。在这种情况下，系统可使能系统时钟的源振荡器，该振荡器在延时后准备就绪。
7. 当 APB 时钟正在运行时，外设可以在其状态和数据寄存器中记录 I3C 传输。当总线传输完成后，外设会生成相应的标志 (IBIF/CRF/HJF)，使能的中断可以唤醒 CPU。

作为目标：通过复位时序唤醒

目标复位时序是控制器从低功耗模式唤醒并可能复位目标的特定方案，包括 RSTACT CCC 和带内复位时序生成。

作为目标，I3C 通过复位时序从停止模式唤醒的序列如下：

1. 当外设在总线上检测到复位时序（当 SCL 保持低电平时发生 14 次 SDA 跳变）时，会向系统请求其 APB 时钟以将 I3C_EVR 寄存器的 RSTF 标志置 1。
2. 系统收到 I3C APB 时钟请求的通知，PWR 模块的电源管理单元随即唤醒（几微秒后）。
3. 如果调压器必须提高运行模式下的电压，则可能需要额外的延时。
4. 系统使能系统时钟的源振荡器，该振荡器在几微秒后准备就绪。
5. 当 APB 时钟正在运行时，I3C 外设可以将 I3C_EVR 寄存器的 RSTF 标志置 1，使能的中断可以唤醒 CPU。

作为目标：通过丢失的起始位唤醒

1. 外设向系统请求其内核时钟：
   – 在 I3C 总线上检测到起始位时（检测到 SDA 线被驱动为低电平，而 SCL 为高电平）
   – 作为目标，由外部 I3C 器件（控制器或目标）发出请求。
   – 如果在 SCL 被外部控制器驱动为低电平之前未提供内核时钟，则 I3C 目标会检测到丢失了 I3C 总线起始位并等待提供内核时钟。请注意，应发送到目标的 I3C 控制器消息可能已丢失（且未确认）。也可能丢失了从控制器发送到另一个被寻址的目标的 I3C CCC 或私有消息，或来自另一个目标的 IBI/CR/HJF 启动请求。
2. 系统使能 I3C 内核时钟的源振荡器，时钟随即准备就绪（如果是 HSI，则在几微秒后）。用户可以在停止模式下使源振荡器保持开启状态来缩短该启动延迟，但代价是增加功耗。特别是在控制器处于 x = 0 的工作状态（具有 1 µs T_{CAS} 延时）时。
3. 之后，内核时钟继续运行，作为目标的 I3C 外设向系统请求其 APB 时钟以将 WKPF 标志置 1（由非用户文本触发），或者：

4. 系统收到 I3C APB 时钟请求的通知，PWR 模块的电源管理单元随即唤醒（几微秒后）。
5. 如果调压器必须提高运行模式下的电压，则可能需要额外的延时。
6. 系统使能驱动 APB 时钟的系统时钟。如果为系统和 APB 时钟选择的振荡器源与用于驱动 I3C 内核时钟的振荡器源不同，则会出现额外的延时。在这种情况下，系统可使能系统时钟的源振荡器，该振荡器在几微秒后准备就绪。
7. 当 APB 时钟正在运行时：
   – 丢失起始位标志（I3C_EVR 寄存器的 WKPF）由非用户文本置 1：如果标志置 1，使能的中断可以唤醒 CPU。然后便可确认 I3C 总线事务已丢失，但无法确认目标是否已被寻址。软件应使用一段超时，如果控制器在此时间间隔内没有再次对目标进行寻址，则返回停止模式。

35.14 低功耗模式下的 I3C

表 280. 低功耗模式的影响

模式	说明
睡眠	无影响。I3C 中断可使器件退出睡眠模式。
停止(1)	进入停止模式时，I3C 寄存器的内容保留。 I3C 硬件会自动管理其自身的时钟门控，并为其内核时钟和 APB 时钟生成传输到 RCC 的时钟请求输出信号。I3C 中断可使器件唤醒并退出停止模式。可以进行 I3C 传输，也可以借助自主 DMA 与存储器之间进行数据传输，前提是自主 DMA 可以在停止模式下工作。 更多详细信息，请参见*第 35.13 节*。
待机	I3C 外设掉电，退出待机模式后必须重新初始化。

1. 如需了解是否支持停止模式以及支持哪种停止模式，请参见*表 268*。

35.15 I3C 中断

表 281. I3C 中断请求

中断缩写		中断事件	用作		中断使能：I3C_IER 中的位域	事件标志：I3C_EVR 中的位域	事件清除方法：将 1 写入 I3C_CEVR 中的位域
			控制器	目标			
I3C	EVT	请求控制字	X	-	CFNFIE	CFNFF	-
		状态字可用	X	-	SFNEIE	SFNEF	-
		请求待发送的数据	X	X	TXFNFIE	TXFNFF	-
		接收到的数据可用	X	X	RXFNEIE	RXFNEF	-
		控制器：帧传输完成 目标：私有传输完成	X	X	FCIE	FCF	CFCF
		目标提前结束私有读传输（并且 I3C_CFGR 寄存器中的 SMODE = 0）	X	-	RXTGTENDIE	RXTGTENDF	CRXTGTENDF
		接收到 IBI 请求	X	-	IBIIE	IBIF	CIBIF
		接收到控制器角色请求	X	-	CRIE	CRF	CCRF
		接收到热加入请求	X	-	HJIE	HJF	CHJF
		IBI 请求完成	-	X	IBIENDIE	IBIENDF	CIBIENDF
		I3C 总线起始位丢失	-	X	WKPIE	WKPF	CWKPF
		接收到直接 GETACCR CCC	-	X	CRUPDIE	CRUPDF	CCRUPDF
		接收到直接 GETSTATUS CCC	-	X	STAIE	STAF	CSTAF
		接收到任何直接 GETxxx CCC（GETSTATUS 除外）	-	X	GETIE	GETF	CGETF
		动态地址更新（接收到广播 ENTDAA、广播 RSTDAA 或直接 SETNEWDA）	-	X	DAUPDIE	DAUPDF	CDAUPDF
		接收到直接 SETMWL CCC	-	X	MWLUPDIE	MWLUPDF	CMWLUPDF
		接收到直接 SETMRL CCC	-	X	MRLUPDIE	MRLUPDF	CMRLUPDF
		检测到复位时序	-	X	RSTIE	RSTF	CRSTF
		总线工作状态更新（接收到直接/广播 ENTASx CCC）	-	X	ASUPDIE	ASUPDF	CASUPDF
		接收到广播/直接 ENEC/DISEC CCC	-	X	INTUPDIE	INTUPDF	CINTUPDF
		接收到广播 DEFTGTS CCC	-	X	DEFIE	DEFF	CDEFF
		接收到广播 DEFGRPA CCC	-	X	GRPIE	GRPF	CGRPF
	ERR	发生错误	X	X	ERRIE	ERRF	CERRF

35.16 I3C 寄存器

访问 I3C 寄存器时，必须使用按 32 位字对齐的地址。

注意： 访问 I3C_RDR 和 I3C_TDR 寄存器时，必须使用单个有效 LSB 数据字节，以便读取 RX-FIFO 和写入 TX-FIFO。

35.16.1 I3C 消息控制寄存器 (I3C_CR)

I3C message control register

偏移地址：0x000

复位值：0x0000 0000

必须使用该寄存器来控制在 I3C 总线上发出的消息：

- 当 I3C 用作控制器时（位[30] = MTYPE[3] = 0）：如果没有要发出的 CCC 代码，则位[29:27] = MTYPE[2:0] ≠ 110；否则必须遵循*第 35.16.2 节*中的复用寄存器说明。
- 当 I3C 用作目标时（位[30] = MTYPE[3] = 1）。

当 I3C 用作控制器时：

- 如果控制 FIFO (C-FIFO) 未满（I3C_EVR 寄存器中的 CFNFF = 1），则写入该寄存器意味着将新的控制字推入 C-FIFO；具体是通过软件完成还是通过 DMA 自动完成取决于 I3C_CFGR 寄存器中的 CDMAEN。
- 如果 C-FIFO 为空并且必须使用新的控制字发出重复起始位，则 I3C 硬件会将控制 FIFO 错误下溢标志置 1（I3C_SER 寄存器中的 COVR = 1）。如果使能了相应的中断（I3C_IER 寄存器中的 ERRIE = 1），则会生成中断。
- 帧的最后一条消息完成传输后（I3C_CR 寄存器中 MEND = 1 时的消息），I3C 硬件会将帧完成标志置 1（I3C_EVR 寄存器中的 FCF = 1）并生成相应的中断（如果使能）。

当 I3C 用作目标时，该寄存器用于寄存器模式：

- 通过软件写入该寄存器以在 I3C 总线上发出命令（IBI、控制器角色或热加入请求）。
- 禁止 C-FIFO，且不支持用于控制字的 DMA 模式。

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

位 31 **MEND**：消息结束类型/帧的最后一条消息（当 I3C 用作控制器时）(Message end type/last message of a frame (when the I3C acts as controller))

0：来自控制器的此消息后跟重复起始位 (Sr)，然后才能发出另一条消息

1：来自控制器的此消息以停止位 (P) 结束，是帧的最后一条消息

位 30:27 **MTYPE[3:0]**：消息类型（I3C 用作控制器/目标）(Message type (whatever I3C acts as controller/target))
条件：当 I3C 用作 I3C 控制器时
0000：**SCL 时钟强制停止**直到执行下一个控制字
位[26:0] 被忽略。在检测到阻止生成起始位/重复起始位/停止位的 CE1 错误时（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 CODERR[3:0] = 0001），软件必须使用此消息类型使 SCL “保持”恢复状态。请参见*表 278*。
0001：**报头消息**
位[26:0] 被忽略。如果被寻址的目标未以 ACK 响应私有/直接消息，作为 GETSTATUS 尝试失败后的升级阶段，软件必须通过向 I3C_CFGR 寄存器中写入 EXITPTRN = 1 对此进行编程，以便在总线上发出 HDR 退出时序，无论报头是被 ACK 还是 NACK（以避免目标认为 I3C 总线处于 HDR 模式）。有关升级处理，请参见*表 278* 以及 MIPI 规范。
0010：**私有信息**（请参见*图 412*）
位[23:17] (ADD[6:0]) 是发出的 7 位动态地址。
位[16] (RNW) 是发出的 RnW 位。
位[15:0] (DCNT[15:0]) 是编程数据的字节数。
传输的私有信息为：
– {S / S + 0b111_1110 + RnW = 0 + Sr/Sr+*} + 7 位 DynAddr + RnW + (8 位数据 + T)* + Sr/P。
– 在 S（起始位）之后，根据 I3C_CFGR 寄存器中的 NOARBH 位决定是否插入可仲裁头 (0b111_1110 + RnW = 0)。
– Sr+*：在 Sr（重复起始位）之后，硬件会根据需要自动插入 (0b111_1110 + RnW = 0)，前提是其跟随在上一条未以 P（停止位）结束的消息之后。
0011：**直接消息（I3C SDR 直接 CCC 命令的第二部分）**（请参见*图 405*）
位[23:17] (ADD[6:0]) 是发出的 7 位动态地址。
位[16] (RNW) 是发出的 RnW 位。
位[15:0] (DCNT[15:0]) 是编程数据的字节数。
传输的直接信息为：Sr + 7 位 DynAddr + RnW + (8 位数据 + T)* + Sr/P
0100：**传统 I²C 消息**（请参见*图 414*）
位[23:17] (ADD[6:0]) 是发出的 7 位静态地址。
位[16] (RNW) 是发出的 RnW 位。
位[15:0] (DCNT[15:0]) 是编程数据的字节数。
传输的传统 I²C 消息为：
– {S / S + 0b111_1110 + RnW = 0 + Sr/Sr+*} + 7 位 StaAddr + RnW + (8 位数据 + T)* + Sr/P。
– 在 S 之后，根据 NOARBH 位决定是否插入可仲裁头 (0b111_1110 + RnW = 0)。
– Sr+*：在 Sr（重复起始位）之后，硬件会根据需要自动插入 (0b111_1110 + RnW = 0)，前提是其跟随在上一条未以 P（停止位）结束的消息之后。
其他值：保留
条件：当 I3C 用作 I3C 目标时
1000：**热加入请求** (W)（请参见*图 416*）
传输的热加入请求为 {S +} 0b000_0010 addr + RnW = 0。
如果目标使能了热加入请求（I3C_DEVR0 寄存器中的 HJEN = 1），则可以通过写入控制字来发出该请求，具体分为主动和被动两种方式。主动方式是在总线空闲条件后通过硬件发出启动请求（SDA 低电平）并等待控制器激活 SCL 时钟，被动方式是由控制器发出并发消息。
1001：**控制器角色请求** (W)（请参见*图 417*）
传输的控制器角色请求为 {S +} DA[6:0] + RnW = 0（I3C_DEVR0 寄存器中的 DA）
如果目标使能了控制器角色请求（I3C_DEVR0 寄存器中的 CREN = 1 且 DAVAL = 1），则可以通过写入控制字来发出该请求，具体分为主动和被动两种方式。主动方式是在总线空闲条件后通过硬件发出启动请求（SDA 低电平）并等待控制器激活 SCL 时钟，被动方式是由控制器发出并发消息。
1010：**IBI（带内中断）请求** (R)（请参见*图 415*）
位[15:0] (DCNT[15:0]) 是 IBI 数据有效负载的字节数（包括第一个 MDB）（如果有）。
传输的 IBI 请求为 {S +} DA[6:0] + RnW = 1 + 可选 IBI 数据有效负载。如果目标使能了 IBI 请求（I3C_DEVR0 寄存器中的 IBIEN = 1 且 DAVAL = 1），则可以通过写入控制字来发出该请求，具体分为主动和被动两种方式。主动方式是在总线空闲条件后通过硬件发出启动请求（SDA 低电平）并等待控制器激活 SCL 时钟，被动方式是由控制器发出并发消息。
当控制器确认后，发送的 IBI 有效负载数据（可选，是否选择取决于 I3C_BCR 寄存器中的 BCR2）由 I3C_CR 寄存器中的 DCNT[15:0] 和 I3C_IBIDR 定义，并且必须与 I3C_IBIDR 寄存器中 IBIP[2:0] 定义的 IBI 有效负载数据大小一致。
其他值：保留

位 26:24 保留，必须保持复位值。

位 23:17 **ADD[6:0]**：7 位 I3C 动态/I²C 静态目标地址（当 I3C 用作控制器时）(7-bit I3C dynamic / I²C static target address (when I3C acts as controller))
当 I3C 用作控制器时，如果 MTYPE[3:0] = 0010（私有消息）、MTYPE[3:0] = 0011（直接消息）或 MTYPE[3:0] = 0100（传统 I²C 消息），则使用该位域

位 16 **RNW**：读/非写消息（当 I3C 用作控制器时）(Read / non-write message (when I3C acts as controller))

当 I3C 用作控制器时，如果 MTYPE[3:0] = 0010（私有消息）、MTYPE[3:0] = 0011（直接消息）或 MTYPE[3:0] = 0100（传统 I^2C 消息），则使用该位域在 I3C 总线上发出 RnW 位。

0：写消息

1：读消息

位 15:0 **DCNT[15:0]**：读/写消息期间要传输的数据计数，以字节为单位（I3C 用作控制器/目标）(Count of data to transfer during a read or write message, in bytes (whatever I3C acts as controller/target))

当 I3C 用作控制器时，如果 MTYPE[3:0] = 0010（私有消息）、MTYPE[3:0] = 0011（直接消息）或 MTYPE[3:0] = 0100（传统 I^2C 消息），则使用该位域来设置在总线上交换的数据字节数。如果是私有或传统 I^2C 读/写消息，则该位域必须非空。

当 I3C 用作目标时，如果 MTYPE[3:0] = 1010（IBI 请求）并且存在任何 IBI 数据有效负载（I3C_BCR 寄存器中 BCR2 = 1 时要发送的数据），则使用该位域来设置 IBI 数据有效负载的字节数（1、2、3 或 4）。

线性编码最长可达 64 KB - 1

0x0000：没有数据要传输

0x0001：1 字节

0x0002：2 字节

...

0xFFFF：64 KB - 1 字节

35.16.2 I3C 消息控制寄存器 [复用] (I3C_CR)

I3C message control register [alternate]

偏移地址：0x000

复位值：0x0000 0000

当控制关于控制器何时必须发出 CCC 命令的消息（无论 CCC 的类型为何：广播 CCC、直接 CCC 或进入 HDR CCC）时，必须遵循该写寄存器说明。

这是当 MTYPE[3:0] = 0110 时的 I3C_CR 寄存器说明（复用）。其他情况时，请参见*第 35.16.1 节*。

如果控制 FIFO（也称为 C-FIFO）未满（I3C_EVR 寄存器中的 CFNFF = 1），则写入该寄存器意味着将新的控制字推入 C-FIFO；具体是通过软件完成还是通过 DMA 自动完成取决于 I3C_CFGR 寄存器中的 CDMAEN 位。

帧的最后一条消息完成传输后（I3C_CR 寄存器中 MEND = 1 时的消息），I3C 硬件会将帧完成标志置 1（I3C_EVR 寄存器中的 FCF = 1）并生成相应的中断（如果使能）。

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

位 31 **MEND**：消息结束类型/帧的最后一条消息（当 I3C 用作控制器时）(Message end type/last message of a frame (when I3C acts as controller))

0：来自控制器的此消息后跟重复起始位 (Sr)，然后才能发出另一条消息

1：来自控制器的此消息以停止位 (P) 结束，是帧的最后一条消息

位 30:27 **MTYPE[3:0]**：消息类型（当 I3C 用作控制器时）(Message type (when I3C acts as controller))

条件：当 I3C 用作 I3C 控制器时

0110：广播/直接 CCC 命令（请参见*表 277*、*图 405*、*图 406* 和*图 407*）

位[23:16] (CCC[7:0]) 是发出的 8 位 CCC 代码

位[15:0] (DCNT[15:0]) 是 CCC 释义字节、CCC 子命令字节或 CCC 数据字节的数量。

如果位[23] = CCC[7] = 1：消息为 I3C SDR 直接 CCC 命令的第一部分

传输的直接 CCC 命令（第一部分）消息为：

– {S / S + 0b111_1110 +RnW = 0 / Sr+*} + (直接 CCC + T) + (8 位数据 + T)* + Sr

– 在 S（起始位）之后，根据 I3C_CFGR 寄存器中的 NOARBH 位决定是否插入可仲裁头 (0b111_1110 + RnW = 0)。

– Sr+*：在 Sr（重复起始位）之后，硬件会自动插入 (0b111_1110 + R/W)。

如果位[23] = CCC[7] = 0：消息为 I3C SDR 广播 CCC 命令（包括特定的 ENTDAA，请参见*图 406*）

传输的广播 CCC 命令消息为：

– {S / S + 0b111_1110 + RnW = 0 / Sr+*} + (广播 CCC + T) + (8 位数据 + T)* + Sr/P

– 在 S（起始位）之后，根据 NOARBH 位决定是否插入可仲裁头 (0b111_1110 + RnW = 0)。

– Sr+*：在 Sr（重复起始位）之后，硬件会自动插入 (0b111_1110 + R/W)。

其他值：保留

位 26:24 保留，必须保持复位值。

位 23:16 **CCC[7:0]**：8 位 CCC 代码（当 I3C 用作控制器时）(8-bit CCC code (when I3C acts as controller))

如果位[23] = CCC[7] = 1，则消息为 I3C SDR 直接 CCC 命令的第一部分。

如果位[23] = CCC[7] = 0，则消息为 I3C SDR 广播 CCC 命令（包括 ENTDAA）。

位 15:0 **DCNT[15:0]**：要作为 CCC 释义字节、CCC 子命令字节或 CCC 数据字节传输的 CCC 命令相关数据的计数（以字节为单位）(Count of related data to the CCC command to transfer as CCC defining bytes, or CCC sub-command bytes, or CCC data bytes, in bytes)

线性编码最长可达 64 KB - 1。

0x0000：没有数据要传输。

注意：发出 ENTDAA 广播 CCC 时强制使用的值（请参见图 406）。

0x0001：1 字节

注意：发出 RSTACT 直接/广播 CCC 时强制使用的值（请参见图 407）。

0x0002：2 字节

...

0xFFFF：64 KB - 1 字节

35.16.3 I3C 配置寄存器 (I3C_CFGR)

I3C configuration register

偏移地址：0x004

复位值：0x0000 0000

该寄存器用于配置：

- I3C 用作控制器或目标时适用的功能：RX-FIFO 和 TX-FIFO 管理（RXDMAEN、RXTHRES、RXFLUSH、TXDMAEN、TXTHRES、TXFLUSH）、I3C 外设角色 (CRINIT)
- I3C 用作控制器时的专用功能：基于帧的控制字触发 (TSFSET)、FIFO 管理（TMODE、SMODE、SFLUSH、SDMAEN、CDMAEN）和其他管理（HJACK、HKSDAEN、EXITPTRN、RSTPATRN、NOARBH）

只能在 EN = 0 时修改配置位域 CRINIT 和 HKSDAEN。如果在修改这两个位域的同时将 EN 置 1，则满足上述条件（无需稍后再执行一次写操作将 EN 置 1）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	TSFSET	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CFLUSH	CDMA EN	TMODE	SMODE	SFLUSH	SDMA EN
	w									w	rw	rw	rw	w	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TX THRES	TX FLUSH	TX DMAEN	Res.	RX THRES	RX FLUSH	RX DMAEN	HJACK	Res.	HK SDAEN	EXIT PTRN	RST PTRN	NO ARBH	CRINIT	EN
	rw	w	rw		rw	w	rw	rw		rw	rw	rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30 **TSFSET**：帧传输设置（软件触发）（当 I3C 用作控制器时）(Frame transfer set (software trigger) (when I3C acts as controller))

该位只能写入。当 I3C 用作 I3C 控制器时：

0：不执行任何操作

1：将该位置 1 后，硬件会将 I3C_EVR 寄存器中的标志 CFNFF 置 1 以启动帧传输（C-FIFO 未满，需要控制字）

注意：如果该位未置 1，可选择通过软件启动帧传输，方法是在 C-FIFO 为空（I3C_EVR 寄存器中的 CFEF = 1）时直接写入第一个控制字寄存器 (I3C_CR)。然后，如果第一个写入的控制字未标记为消息结束（I3C_CR 寄存器中的 MEND = 0），则会导致硬件将 CFNFF 置 1。

位 29:24 保留，必须保持复位值。

位 23 保留，必须保持复位值。

位 22 保留，必须保持复位值。

位 21 **CFLUSH**：C-FIFO 清空（当 I3C 用作控制器时）(C-FIFO flush (when I3C acts as controller))

该位只能写入。

0：不执行任何操作

1：清空 C-FIFO

位 20 **CDMAEN**：C-FIFO DMA 请求使能（当 I3C 用作控制器时）(C-FIFO DMA request enable (when I3C acts as controller))

当 I3C 用作控制器时：

0：针对 C-FIFO 禁止 DMA 模式

- 软件根据给定帧的需要将控制字写入/推入 C-FIFO（写入 I3C_CR 寄存器）。
- 下一个控制字传输可由软件通过轮询 I3C_EVR 寄存器中的标志 CFNFF = 1 或通过中断通知（通过 I3C_IER 寄存器中的 CFNFIE = 1 使能）来写入。

1：针对 C-FIFO 使能 DMA 模式

- DMA 根据给定帧的需要将控制字写入/推入 C-FIFO（写入 I3C_CR 寄存器）。
- 下一个控制字传输由编程设定的硬件自动写入（通过来自 I3C 的有效 C-FIFO DMA 请求和编程设定的 DMA 通道实现）。

位 19 **TMODE**：发送模式（当 I3C 用作控制器时）(Transmit mode (when I3C acts as controller))

当 I3C 用作控制器时，该位用于针对 I3C 总线上发出的帧来管理 C-FIFO 和 TX-FIFO。

0：在开始发起帧传输之前，不预装载 C-FIFO 和 TX-FIFO。

第一个控制字出现在 C-FIFO 中时，帧传输立即开始。

1：在开始发起帧传输之前，先预装载 C-FIFO 和 TX-FIFO（也可以根据需要只预装载 TX-FIFO，具体取决于帧格式）。更多详细信息，请参见*第 35.10.2 节*。

位 18 **SMODE**：S-FIFO 使能/状态接收模式（当 I3C 用作控制器时）(S-FIFO enable / status receive mode (when I3C acts as controller))

当 I3C 用作控制器时，该位用于使能 I3C 总线上交换的消息的状态 FIFO (S-FIFO)。

当 I3C 用作目标时，该位必须清零。

0：禁止 S-FIFO

- 在没有 FIFO 机制的情况下使用状态寄存器 (I3C_SR)。
- 如果未读取新的状态寄存器内容，则不会出现 SCL 停顿。
- 必须在状态寄存器被硬件覆盖之前读取该寄存器。
- 必须将 I3C_CFGR 寄存器中的 SDMAEN 清零。

1：使能 S-FIFO。

- 必须读取每一个消息状态。
- 当 S-FIFO 已满并且必须读取下一个消息状态时，将会出现 SCL 停顿。
- 在最大 SCL 时钟停顿时间过后，将会报告 S-FIFO 上溢错误。

位 17 **SFLUSH**：S-FIFO 清空（当 I3C 用作控制器时）(S-FIFO flush (when I3C acts as controller))

只有在 I3C 用作控制器时，才能写入和使用该位。

0：不执行任何操作

1：清空 S-FIFO

位 16 **SDMAEN**：S-FIFO DMA 请求使能（当 I3C 用作控制器时）(S-FIFO DMA request enable (when I3C acts as controller))

如果 I3C_CFGR 寄存器中的 SMODE = 0（禁止 S-FIFO），则该位必须清零。换句话说，如果禁止 S-FIFO，则无法使用 DMA 模式。然后，可以选择读取或不读取状态寄存器 I3C_SR。

如果 SMODE = 1（使能 S-FIFO），则该位可以置 1 或清零。换句话说，必须通过软件或通过分配的 DMA 通道为每一条消息读取状态寄存器 I3C_SR。

0：禁止通过 DMA 模式读取状态寄存器 I3C_SR

- SMODE = 0：软件可以在帧完成传输（I3C_EVR 寄存器中的 FCF = 1）或出现错误（I3C_EVR 寄存器中的 ERRF = 1）后读取 I3C_SR 寄存器。通过轮询这些寄存器标志或通过中断通知（通过 I3C_IER 寄存器中的 FCIE = 1 和 ERRIE = 1 使能）。

- SMODE = 1：在每次 SFNEF 标志置 1 后，软件都必须从 S-FIFO 读取/弹出一个状态字（读取 I3C_SR 寄存器）。通过轮询该寄存器标志或通过中断通知（通过 I3C_IER 寄存器中的 SFNEIE = 1 使能）。

1：使能通过 DMA 模式读取状态寄存器 I3C_SR

- 必须将 I3C_CFGR 寄存器中的 SMODE 置 1（使能 S-FIFO）

- DMA 从 S-FIFO 读取/弹出状态字（读取 I3C_SR 寄存器）

- 状态字由编程设定的硬件自动读取（通过来自 I3C 的有效 S-FIFO DMA 请求和编程设定的 DMA 通道实现）。

位 15 保留，必须保持复位值。

位 14 **TXTHRES**：TX-FIFO 阈值（I3C 用作控制器/目标）(TX-FIFO threshold (whatever I3C acts as controller/target))

该阈值以 TX-FIFO 级别来定义 I3C_EVR 寄存器中的 TXFNFF 标志何时置 1（如果 TXDMAEN = 1，则还定义 DMA TX 请求何时有效）。

0：1 字节阈值

当必须将 1 个字节写入 TX-FIFO（在 I3C_TDR 中）时，TXFNFF 置 1。

1：1 字/4 字节阈值

当必须将 1 个字/4 个字节写入 TX-FIFO（在 I3C_TDWR 寄存器中）时，TXFNFF 置 1。如果最后发送的数据的字节数不是 4 的倍数（I3C_SR 寄存器中的 XDCNT[1:0] = 00），则硬件仅考虑最后一个字相关的 1、2 或 3 个有效 LSB 字节，并在 I3C 总线上发送。

位 13 **TXFLUSH**：TX-FIFO 清空（I3C 用作控制器/目标）(TX-FIFO flush (whatever I3C acts as controller/target))

该位只能写入。

当 I3C 用作目标时，如果控制器已中止数据读取（将 T 位驱动为低电平），并且 TX-FIFO 中还有剩余数据（ABT = 1，且 I3C_SR 寄存器中的 XDCNT[15:0] < I3C_TGTTDR 寄存器中的 TGTTDCNT[15:0]），则该位可用于在私有读传输期间清空 TX-FIFO。

0：不执行任何操作

1：清空 TX-FIFO

位 12 **TXDMAEN**：TX-FIFO DMA 请求使能（I3C 用作控制器/目标）(TX-FIFO DMA request enable (whatever I3C acts as controller/target))

0：针对 TX-FIFO 禁止 DMA 模式

- 软件将数据字节/字写入/推入 TX-FIFO（写入 I3C_TDR 或 I3C_TDWR 寄存器），以便通过 I3C 总线发送。

- 下一个数据字节/字必须由软件通过轮询标志 TXFNFF = 1 或通过中断通知（通过 TXFNFIE = 1 使能）写入。

1：针对 TX-FIFO 使能 DMA 模式

- DMA 将数据字节/字写入/推入 TX-FIFO（写入 I3C_TDR 或 I3C_TDWR 寄存器）。

- 下一个数据字节/字传输由编程设定的硬件自动推入（通过来自 I3C 的有效 TX-FIFO DMA 请求和编程设定的 DMA 通道实现）。

位 11 保留，必须保持复位值。

位 10 **RXTHRES**：RX-FIFO 阈值（I3C 用作控制器/目标）(RX-FIFO threshold (whatever I3C acts as controller/target))

该阈值以 RX-FIFO 级别来定义 I3C_EVR 寄存器中的 RXFNEF 标志何时置 1（如果 RXDMAEN = 1，则还定义 DMA RX 请求何时有效）。

0：1 字节阈值

当必须在 RX-FIFO（在 I3C_RDR 寄存器中）中读取 1 个字节时，RXFNEF 置 1。

1：1 字/4 字节阈值

当要在 RX-FIFO（在 I3C_RDWR 中）读取 1 个字/4 个字节时，RXFNEF 置 1。如果最后接收到的数据的字节数不是 4 的倍数，则软件仅考虑最后一个字相关的 1、2 或 3 个有效 LSB 字节。接收到的数据有效字节数由 I3C_SR 寄存器中的 XDCNT[15:0] 报告。

位 9 **RXFLUSH**：RX-FIFO 清空（I3C 用作控制器/目标）(RX-FIFO flush (whatever I3C acts as controller/target))

该位只能写入。

0：不执行任何操作

1：清空 RX-FIFO

位 8 **RXDMAEN**：RX-FIFO DMA 请求使能（I3C 用作控制器/目标）(RX-FIFO DMA request enable (whatever I3C acts as controller/target))

0：针对 RX-FIFO 禁止 DMA 模式

- 软件从 RX-FIFO 读取/弹出数据字节/字（读取 I3C_RDR 或 I3C_RDWR 寄存器）。

- 下一个数据字节/字必须由软件通过轮询 I3C_EVR 寄存器中的标志 RXFNEF = 1 或通过中断通知（通过 I3C_IER 寄存器中的 RXFNEIE = 1 使能）来读取。

1：针对 RX-FIFO 使能 DMA 模式

- DMA 从 RX-FIFO 读取/弹出数据字节/字（读取 I3C_RDR 或 I3C_RDWR 寄存器）。

- 下一个数据字节/字由编程设定的硬件自动读取（通过来自 I3C 的有效 RX-FIFO DMA 请求和编程设定的 DMA 通道实现）。

位 7 **HJACK**：热加入请求确认（当 I3C 用作控制器时）(Hot-join request acknowledge (when I3C acts as a controller))

0：热加入请求不被确认

NACK 后，控制器按照初始编程的方式继续工作（热加入目标已接收到 NACK，必须稍后发出另一个热加入请求）。

1：热加入请求被确认

ACK 后，控制器按照初始编程的方式继续工作。软件通过 HJ 中断获得通知（I3C_EVR 寄存器中的标志 HJF 置 1），必须稍后启动 ENTDAA 序列，从而可以使用禁止目标事件命令（DISEC，DISHJ = 1）来防止其他热加入请求。

位 6 保留，必须保持复位值。

位 5 **HKSDAEN**：SDA 线上的高电平保持器使能（当 I3C 用作控制器时）(High-keeper enable on SDA line (when I3C acts as a controller))

0：禁止高电平保持器

1：使能高电平保持器，T 位上的弱上拉有效（而非开漏类上拉）。

注意：只有在 I3C_CFGR 寄存器中的 EN = 0 时，才能修改该位。

位 4 **EXITPTRN**：HDR 退出时序使能（当 I3C 用作控制器时）(HDR exit pattern enable (when I3C acts as a controller))

只有在当前未进行任何帧传输时，才能修改该位。

0：发出消息报头（I3C_CR 寄存器中的 MTYPE[3:0] = 0001）后不发送 HDR 退出时序。这用于发送报头，在控制器角色切换后，如果怀疑存在问题（新控制器未在工作状态定义的延迟时间内通过访问前一个控制器来使能其控制器角色），可以以此测试总线的所有权。

1：发出消息报头 (MTYPE[3:0] = 0001) 后发送 HDR 退出时序。这用于在目标未响应私有消息或直接读 CCC 时进行控制器错误检测和升级处理。

无论消息报头 {S/Sr + 0x7E addr + W} 是被 ACK 还是 NACK，都会发送 HDR 退出时序。

位 3 **RSTPTRN**：HDR 复位时序使能（当 I3C 用作控制器时）(HDR reset pattern enable (when I3C acts as a controller))

只有在当前未进行任何帧传输时，才能修改该位。

0：在帧结束时发出标准停止位

1：在任何发出的帧（包含 RSTACT CCC 命令）的停止位前都插入 HDR 复位时序

位 2 **NOARBH**：起始位后无可仲裁头（当 I3C 用作控制器时）(No arbitrable header after a start (when I3C acts as a controller))

只有在当前未进行任何帧传输时，才能修改该位。

0：在起始位之后、在传统 I^2C 消息或 I3C SDR 私有读/写消息之前（默认）发出可仲裁头 (0b111_1110 + RnW = 0)。

1：无可仲裁头

- 对于传统 I^2C 消息或 I3C SDR 私有读/写消息，在起始位后直接发出目标地址。
- 这是一种更高效的选项（当发出 0x7E 可仲裁头无用时），但只能在控制器确定被寻址的目标器件无法同时发出 IBI 或控制器角色请求时才可以使用（以确保对于 IBI 或 MR，控制器在开漏模式下发出的地址与目标器件在起始位后可以发出的同一地址之间不会出现错误解析和潜在冲突）。

位 1 **CRINIT**：标记控制者/目标角色 (Initial controller/target role)

只有在 I3C_CFGR 寄存器中的 EN = 0 时，才能修改该位。

0：目标角色

通过设置 EN = 1 使能后，外设最初用作目标。I3C 不驱动 SCL 线，也不使能 SDA 拉高，直到其最终获得控制器角色。

1：控制器角色

通过设置 EN = 1 使能后，外设最初用作控制器。获得 I3C 控制器角色后，I3C 将驱动 SCL 线并使能 SDA 拉高，直到其最终将控制器角色提供给 I3C 辅助控制器。

位 0 **EN**：I3C 使能（I3C 用作控制器/目标）(I3C enable (whatever I3C acts as controller/target))

0：禁止 I3C

- 除寄存器外，外设均处于复位状态（部分复位）。
- 在将 EN 清零之前，当 I3C 用作控制器时，必须使用 DISEC CCC 禁止所有可能的目标请求。
- 当 I3C 作为目标时，软件不得禁止 I3C，除非需要部分复位。

1：使能 I3C

在该状态下，无法修改某些寄存器位域（如 I3C_CFGR 的 CRINIT、HKSDAEN）

35.16.4 I3C 接收数据字节寄存器 (I3C_RDR)

I3C receive data byte register

偏移地址：0x010

复位值：0x0000 0000

如果 RX-FIFO 配置为使用基于字节的读访问（I3C_CFGR 寄存器中的 RXTHRES = 0），则该寄存器用于读取从 I3C 总线接收到的数据字节。然后：

- 读取时，I3C_RDR 返回一个 32 位数据字，接收到的数据字节位于 LSB 位置，其他保留位读为 0。
- 当 I3C_CFGR 寄存器中的 RXDMAEN = 1 时：编程设定的 I3C 和 DMA 通过硬件自动管理相关的连续读传输。
- 当 RXDMAEN = 0 时：在读取之前，必须通过 RX-FIFO 非空标志（I3C_EVR 寄存器中的 RXFNEF = 1）或通过相应的中断（如果使能）来通知软件必须读取下一个接收到的数据字节。当用作控制器时，要读取的给定消息的最后一个接收字节也将使用 I3C_EVR 寄存器中的 RXLASTF = 1 进行标记。

- 如果 RX-FIFO 已满，无需等待即可接收新的字节但无法将其推入 RX-FIFO，软件将通过 I3C_EVR 寄存器中的错误标志 ERRF = 1 和 I3C_SER 寄存器中的数据上溢标志 DOVRF = 1（和相应的中断（如果使能））获得通知。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDB0[7:0]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **RDB0[7:0]**：I3C 总线上的 8 位接收数据 (8-bit received data on I3C bus)。

35.16.5 I3C 接收数据字寄存器 (I3C_RDWR)

I3C receive data word register

偏移地址：0x014

复位值：0x0000 0000

如果 RX-FIFO 配置为使用基于 32 位字的读访问（I3C_CFGR 寄存器中的 RXTHRES = 1），则该寄存器用于读取从 I3C 总线接收到的数据字节。然后：

- 当 I3C_CFGR 寄存器中的 RXDMAEN = 1 时：编程设定的 I3C 和 DMA 通过硬件自动管理相关的连续读传输。
- 当 RXDMAEN = 0 时：在读取之前，必须先通过 RX-FIFO 非空标志（I3C_EVR 寄存器中的 RXFNEF = 1）或通过相应的中断（如果使能）来通知软件必须读取下一个接收到的数据字。当用作控制器时，要读取的给定消息的最后一个接收字/字节也将使用 I3C_EVR 寄存器中的 RXLASTF = 1 进行标记。
- 如果 RX-FIFO 中保存的字节数小于 4，则读取 I3C_RDWR 将返回一个以空字节填充的数据字：LSB 位置中的可用字节在 MSB 位置中以零字节填充。
- 如果 RX-FIFO 已满，无需等待即可接收新的字节但无法将其推入 RX-FIFO，软件将通过 I3C_EVR 寄存器中的错误标志 ERRF = 1 和 I3C_SER 寄存器中的数据上溢 DOVR = 1（和相应的中断（如果使能））获得通知。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDB3[7:0]								RDB2[7:0]							
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDB1[7:0]								RDB0[7:0]							
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:24 **RDB3[7:0]**：8 位接收数据（I3C 总线上的最新字节）(8-bit received data (latest byte on I3C bus))。

位 23:16 **RDB2[7:0]**：8 位接收数据（I3C 总线上跟在 RDB1 之后的字节）(8-bit received data (next byte after RDB1 on I3C bus))。

位 15:8 **RDB1[7:0]**：8 位接收数据（I3C 总线上跟在 RDB0 之后的字节）(8-bit received data (next byte after RDB0 on I3C bus))。

位 7:0 **RDB0[7:0]**：8 位接收数据（I3C 总线上最早的字节）(8-bit received data (earliest byte on I3C bus))。

35.16.6 I3C 发送数据字节寄存器 (I3C_TDR)

I3C transmit data byte register

偏移地址：0x018

复位值：0x0000 0000

该寄存器用于写入要通过 I3C 总线发送的数据字节。

该寄存器实现了对发送 FIFO (TX-FIFO) 的基于字节的写访问，在 I3C_CFGR 寄存器中的 TXTHRES = 0 时使用。

当 I3C 用作控制器时：

- 当 TXDMAEN = 1 且 I3C_CFGR 寄存器中的 TXTHRES = 0 时：编程设定的 I3C 和 DMA 通过硬件自动管理相关的连续写传输。
- 当 TXDMAEN = 0 且 TXTHRES = 0 时：在写入之前，必须通过 TX-FIFO 非满标志（I3C_EVR 寄存器中的 TXFNFF = 1）或通过相应的中断（如果使能）来通知软件必须写入下一个数据字节。要写入的给定消息的最后一个发送字节也将使用 I3C_EVR 寄存器中的 TXLASTF = 1 进行标记。

当 I3C 用作目标时：

- 当 TXDMAEN = 1、TXTHRES = 0 且 I3C_TGTTDR 寄存器中的 PRELOAD = 1 时：根据 I3C_TGTDR 寄存器中的 TGTTDCNT[15:0]，编程设定的 I3C 和 DMA 通过硬件自动管理对 I3C_TDR 寄存器的相关连续写传输次数。
- 当 TXDMAEN = 0 且 TXTHRES = 0 时：
  - 当 I3C_TGTTDR 寄存器中的 PRELOAD = 1 时：在写入之前，必须通过 TX-FIFO 非满标志 (TXFNFF = 1) 或通过相应的中断（如果使能）来通知软件必须写入下一个数据字节。初始编程的 TGTTDCNT[15:0] 的最后一个发送字节也将使用 I3C_EVR 寄存器中的 TXLASTF = 1 进行标记。
  - 当 PRELOAD = 0 时：在写入之前，必须通过 TX-FIFO 为空标志（I3C_EVR 寄存器中的 TXFEF = 1）或通过相应的中断（如果使能）来通知软件最多可写入接下来的 8 个数据字节（TX-FIFO 大小）。如果软件需要写入另一个数据字节，则必须等待 TX-FIFO 为空 (TXFEF = 1)，然后在 9 个 SCL 时钟周期结束之前写入要发送的数据字节，以避免生成数据下溢错误标志（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 DOVR = 1）。

如果 TX-FIFO 为空，并且控制器无法再等待数据字节发送，软件将通过 I3C_EVR 寄存器中的错误标志 ERRF = 1 和 I3C_SER 寄存器中的数据下溢标志 DOF = 1（和相应的中断（如果使能））获得通知。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDB0[7:0]							
								w	w	w	w	w	w	w	w

位 31:8 保留，必须保持复位值。

位 7:0 **TDB0[7:0]**：I3C 总线上的 8 位发送数据 (8-bit data to transmit on I3C bus)。

35.16.7 I3C 发送数据字寄存器 (I3C_TDWR)

I3C transmit data word register

偏移地址：0x01C

复位值：0x0000 0000

该寄存器用于写入要通过 I3C 总线发送的 32 位数据字。

该寄存器实现了对发送 FIFO (TX-FIFO) 的基于字的写访问，在 I3C_CFGR 寄存器中的 TXTHRES = 1 时使用。

当 I3C 用作控制器时：

- 当 TXDMAEN = 1 且 I3C_CFGR 寄存器中的 TXTHRES = 1 时，编程设定的 I3C 和 DMA 通过硬件自动管理相关的连续写传输。
- 当 TXDMAEN = 0 且 TXTHRES = 1 时：在写入之前，必须通过 TX-FIFO 非满标志（I3C_EVR 寄存器中的 TXFNFF = 1）或通过相应的中断（如果使能）来通知软件必须写入下一个数据字/字节。要写入 TX-FIFO 的给定消息的最后一个发送字/字节也将使用 I3C_EVR 寄存器中的 TXLASTF = 1 进行标记。

当 I3C 用作目标时：

- 当 TXDMAEN = 1、TXTHRES = 1 且 I3C_TGTTDR 寄存器中的 PRELOAD = 1 时：根据 I3C_TGTDR 寄存器中的 TGTTDCNT[15:0]，编程设定的 I3C 和 DMA 通过硬件自动管理对 I3C_TDWR 寄存器的相关连续写传输次数。
- 当 TXDMAEN = 0 且 TXTHRES = 1 时：
  - 当 PRELOAD = 1 时：在写入之前，必须通过 TX-FIFO 非满标志（I3C_EVR 寄存器中的 TXFNFF = 1）或通过相应的中断（如果使能）来通知软件必须写入下一个数据字。I3C_TGTTDR 寄存器中初始编程的 TGTTDCNT[15:0] 的最后一个发送字也将使用 I3C_EVR 寄存器中的 TXLASTF = 1 进行标记。
  - 当 PRELOAD = 0 时：在写入之前，必须通过 TX-FIFO 为空标志（I3C_EVR 寄存器中的 TXFEF = 1）或通过相应的中断（如果使能）来通知软件最多可写入接下来的两个数据字（TX-FIFO 大小）。如果软件需要写入另一个数据字，则必须等待 TX-FIFO 为空 (TXFEF = 1)，然后在 9 个 SCL 时钟周期结束之前写入要发送的下一个数据字，以避免生成数据下溢错误标志（I3C_EVR 寄存器中的 ERRF = 1 且 I3C_SER 寄存器中的 DOVR = 1）。

如果 TX-FIFO 为空，并且控制器/目标无法再等待数据字节发送，软件将通过 I3C_EVR 寄存器中的错误标志 ERRF = 1 和 I3C_SER 寄存器中的数据下溢标志 DOF = 1（和相应的中断（如果使能））获得通知。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TDB3[7:0]								TDB2[7:0]							
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDB1[7:0]								TDB0[7:0]							
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位 31:24 **TDB3[7:0]**：8 位发送数据（I3C 总线上的最新字节）(8-bit transmit data (latest byte on I3C bus))。

位 23:16 **TDB2[7:0]**：8 位发送数据（I3C 总线上跟在 TDB1[7:0] 之后的字节）(8-bit transmit data (next byte after TDB1[7:0] on I3C bus))。

位 15:8 **TDB1[7:0]**：8 位发送数据（I3C 总线上跟在 TDB0[7:0] 之后的字节）(8-bit transmit data (next byte after TDB0[7:0] on I3C bus))。

位 7:0 **TDB0[7:0]**：8 位发送数据（I3C 总线上最早的字节）(8-bit transmit data (earliest byte on I3C bus))

35.16.8 I3C IBI 有效负载数据寄存器 (I3C_IBIDR)

I3C IBI payload data register

偏移地址：0x020

复位值：0x0000 0000

该寄存器用于 IBI 有效负载数据。

当 I3C 用作目标时：

- 如果 I3C_BCR 寄存器中的 BCR2 = 0，则该寄存器无用。
- 如果 BCR2 = 1，该寄存器必须通过软件写入
  - 在控制器确认 IBI 请求（I3C_CR 寄存器中的 MTYPE[3:0] = 1010）后，作为 IBI 有效负载数据在 I3C 总线上发出；IBI 数据有效负载大小由 I3C_CR 寄存器中的 DCNT[15:0] 定义。
  - 最大（静态）有效负载数据大小由 I3C_MAXRLR 寄存器中的 IBIP[2:0] 指定，可以是 1、2、3 或 4 个字节。
  - DCNT[15:0] 必须设置在 1（针对强制数据字节 MDB[7:0]）和最大 IBIP[2:0] 之间。

当 I3C 用作控制器时：如果 I3C_DEVRx 寄存器中的 IBIACK = 1，它在 I3C 总线上确认来自目标 x 的 IBI 请求后：

- 如果 I3C_DEVRx 寄存器中的 IBIDEN = 0（从目标 x 接收到的 BCR[2] = 0），则该寄存器无用。
- 如果 IBIDEN = 1（从目标 x 接收到的 BCR[2] = 1）：
  - 该寄存器由硬件在内部写入从 I3C 总线上接收到的 IBI 有效负载数据（包括第一个强制数据字节 MDB[7:0]）。
  - 当 I3C_IBIDR 中接收到有效负载的最后一个字节时（当目标将 T 位驱动为 0 时），I3C_EVR 寄存器中的 IBIF 标志置 1（并生成相应的中断（如果使能））。
  - 然后，软件可通过接收和记录的 7 位地址（I3C_RMR 寄存器中的 RADD[6:0]）来识别目标 x。
  - 软件可读取该寄存器并根据有效接收和记录的字节数（I3C_RMR 寄存器中的 IBIRDCNT[2:0]）来解析字节。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IBIDB3[7:0]								IBIDB2[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IBIDB1[7:0]								IBIDB0[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **IBIDB3[7:0]**：8 位 IBI 有效负载数据（I3C 总线上的最新字节）(8-bit IBI payload data (latest byte on I3C bus))。

位 23:16 **IBIDB2[7:0]**：8 位 IBI 有效负载数据（I3C 总线上跟在 IBIDB1[7:0] 之后的字节）(8-bit IBI payload data (next byte on I3C bus after IBIDB1[7:0]))。

位 15:8 **IBIDB1[7:0]**：8 位 IBI 有效负载数据（I3C 总线上跟在 IBIDB0[7:0] 之后的字节）(8-bit IBI payload data (next byte on I3C bus after IBIDB0[7:0]))。

位 7:0 **IBIDB0[7:0]**：8 位 IBI 有效负载数据（I3C 总线上最早的字节，MDB[7:0] 强制数据字节）(8-bit IBI payload data (earliest byte on I3C bus, MDB[7:0] mandatory data byte))。

35.16.9 I3C 目标发送配置寄存器 (I3C_TGTTDR)

I3C target transmit configuration register

偏移地址：0x024

复位值：0x0000 0000

当 I3C 用作目标时，必须使用该寄存器在 TX-FIFO 中预装载一定数量的数据字节，以便当控制器接受/确认这些数据时，它们已准备好在 I3C 总线上进行发送，无论是否使用 DMA 模式（与 I3C_CFGR 寄存器中的 TXDMAEN 无关）。

或者，当 I3C 用作目标时，如果要发送的数据字节数小于或等于 TX-FIFO 大小（8 字节），软件可通过轮询 TX-FIFO 为空标志（I3C_EVR 寄存器中的 TXFEF = 1）或通过已使能的相应中断，直接使用字节并将其写入 I3C_TDR 寄存器（或 I3C_TDWR 寄存器，具体取决于 I3C_CFGR 寄存器中的 TXTHRES）。

在任何情况下，由于 I3C 目标无法延长/停顿 SCL 线，因此软件可以通过 I3C_EVR 寄存器中的 DOR = 1 来识别 TX-FIFO 下溢。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRE LOAD
															rw

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

位 31:17 保留，必须保持复位值。

位 16 **PRELOAD**：TX-FIFO 预装载（当 I3C 配置为目标时）(Preload of the TX-FIFO (when I3C is configured as target))

当写入和定义要预装载到 TX-FIFO 并进行发送的字节数时，该位必须由软件在同一访问中写入并置 1。

当所有要发送的数据字节都装载到 TX-FIFO 中时，该位由硬件清零。

0：不执行 TX-FIFO 预装载

1：执行 TX-FIFO 预装载

位 15:0 **TGTTDCNT[15:0]**：发送数据计数器，以字节为单位（当 I3C 配置为目标时）(Transmit data counter, in bytes (when I3C is configured as target))

当 PRELOAD 置 1 时，该位域必须由软件在同一访问中写入，以便定义要预装载和发送的字节数。

该位域由硬件更新，读取该位域会报告要装载到 TX-FIFO 中的剩余字节数。

35.16.10 I3C 状态寄存器 (I3C_SR)

I3C status register

偏移地址：0x030

复位值：0x0000 0000

该寄存器用于读取 I3C 总线上交换的消息的状态：

- 在 FIFO 模式下：当 I3C 用作控制器并且通过 I3C_CFGR 寄存器中的 SMODE = 1 使能了 S-FIFO 时：
  - 不处于 DMA 模式 (SDMAEN = 0) 时，如果有要读取的状态寄存器，则软件通过 I3C_EVR 寄存器中的 SFNEF = 1（和相应的中断（如果使能））获得通知
  - 在 DMA 模式 (SDMAEN = 1) 下，编程设定的 I3C 和 DMA 通过硬件自动管理相关的连续读传输。
  - S-FIFO 溢出时，软件通过 I3C_SER 寄存器中的 COVR = 1（以及I3C_EVR 寄存器中的 ERRF = 1 和相应的中断（如果使能））获得通知
- 在寄存器模式下：如果 I3C_CFGR 寄存器中的 SMODE = 0
  - 软件可使用 I3C_SER 寄存器中的标志 FCF 和 I3C_EVR 寄存器中的 ERRF = 1（和相应的中断（如果使能））来读取该寄存器
  - 该寄存器可以在新消息完成时被硬件重置，且不会发出任何通知。

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

位 31:24 **MID[7:0]**：消息标识符/给定帧的计数器（当 I3C 用作控制器时）(Message identifier/counter of a given frame (when the I3C acts as controller))

当 I3C 用作控制器时，该位域用于标识 I3C_SR 状态寄存器所引用的控制字消息 (I3C_CR)。帧的第一条消息用 MID[7:0] = 0 标识。

当 I3C 总线上的新消息控制字 (I3C_CR) 完成时，该位域（由硬件）递增。每次开始新帧时都会复位该位域。

位 23:19 保留，必须保持复位值。

位 18 **DIR**：消息方向 (Message direction)

无论 I3C 用作控制器还是目标，该位都指示 I3C 总线上相关消息的方向

0：写

1：读

注意：ENTDAA CCC 被视为写命令。

位 17 **ABT**：目标提前结束私有读消息（当 I3C 用作控制器时）(A private read message is ended prematurely by the target (when the I3C acts as controller))

当 I3C 用作控制器时，该位指示目标发送的私有读数据是否提前终止（与控制器预期的编程设定的读数据字节数（I3C_CR 寄存器中的 DCNT[15:0]）相比，目标提前将 T 位驱动为低电平）。

0：目标未提前完成

1：目标提前完成

位 16 保留，必须保持复位值。

位 15:0 **XDCNT[15:0]**：数据计数器 (Data counter)

条件：动态地址分配过程中 (ENTDAA CCC)

- 当 I3C 用作控制器时：总线上检测到的目标数
- 当 I3C 用作目标时：发送的字节数

条件：对于消息期间的其他传输

- 无论 I3C 用作控制器还是目标：消息期间从 I3C 总线读取或在 I3C 总线上发送的数据字节数

35.16.11 I3C 状态错误寄存器 (I3C_SER)

I3C status error register

偏移地址：0x034

复位值：0x0000 0000

该只读寄存器用于在硬件引发错误时获取有关错误的更多信息，并通过 I3C_EVR 寄存器中的错误标志 ERRF = 1（和相应的中断（如果使能））通知软件。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	DERR	DNACK	ANACK	COVR	DOVR	STALL	PERR	CODERR[3:0]			
					r	r	r	r	r	r	r	r	r	r	r

位 31:11 保留，必须保持复位值。

位 10 **DERR**：数据错误（当 I3C 用作控制器时）(Data error (when the I3C acts as controller))

0：未检测到错误

1：当接收到的目标地址或 / 和奇偶校验位不匹配时，控制器在控制器角色切换过程中（GETACCCR CCC，以前称为 GETACCMST）检测到数据错误。活动控制器保留控制器角色。

位 9 **DNACK**：数据未确认（当 I3C 用作控制器时）(Data not acknowledged (when the I3C acts as controller))

0：未检测到错误

1：控制器在以下过程中检测到目标未确认数据字节：

i) 传统 I2C 写传输

ii) 第二次尝试在 ENTDAA 过程中发送动态地址

位 8 **ANACK**：地址未确认（当 I3C 配置为控制器时）(Address not acknowledged (when the I3C is configured as controller))

0：未检测到错误

1：控制器在以下过程中检测到目标未确认静态/动态地址：

i) 传统 I2C 读/写传输

ii) 直接 CCC 写传输

iii) 第二次尝试直接 CCC 读传输

iv) 私有读/写传输

位 7 **COVR**：C-FIFO 下溢或 S-FIFO 上溢（当 I3C 用作控制器时）(C-FIFO underrun or S-FIFO overrun (when the I3C acts as controller))

0：未检测到错误

1：控制器检测到：

i) C-FIFO 下溢：控制 FIFO 为空，必须发出重复起始位

ii) S-FIFO 上溢：S-FIFO 已满，新消息结束

位 6 **DOVR**：RX-FIFO 上溢或 TX-FIFO 下溢 (RX-FIFO overrun or TX-FIFO underrun)

0：未检测到错误

1：无论用作控制器还是目标，硬件都检测到：

i) TX-FIFO 下溢：TX-FIFO 为空并且必须发送写数据字节

ii) RX-FIFO 上溢：RX-FIFO 已满并且接收到新的数据字节

位 5 **STALL**：SCL 停顿错误（当 I3C 用作目标时）(SCL stall error (when the I3C acts as target))

0：未检测到错误

1：目标在 I3C SDR 数据读传输期间（在直接 CCC 读、私有读或 IB 传输期间）检测到 SCL 的稳定时间超过 125 µs

位 4 **PERR**：协议错误 (Protocol error)

0：未检测到错误

1：无论用作控制器还是目标，硬件都检测到协议错误（详细信息请参见 CODERR[3:0]）

位 3:0 **CODERR[3:0]**：协议错误代码/类型 (Protocol error code/type)

0000：CE0 错误（发送 CCC 后的事务）：

控制器检测到非法格式化 CCC

0001：CE1 错误（监视错误）：

控制器检测到总线上发送的数据与预期不符

0010：CE2 错误（对广播地址无响应）：

控制器检测到未确认的广播地址 (0b111_1110)

0011：CE3 错误（控制器角色切换失败）：

控制器检测到新控制器在控制器角色切换后未驱动总线

1000：TE0 错误（无效广播地址 0b111_1110 + W）：

目标检测到无效的广播地址 0b111_1110 + W

1001：TE1 错误（CCC 代码）：

目标通过奇偶校验检测到 CCC 代码中有奇偶校验错误（相对于 T 位）

1010：TE2 错误（写数据）：

目标通过奇偶校验检测到写数据中有奇偶校验错误（相对于 T 位）

1011：TE3 错误（动态地址仲裁期间分配的地址）：

目标在动态地址仲裁期间通过奇偶校验检测到分配的地址中有奇偶校验错误（相对于 PAR 位）

1100：TE4 错误（动态地址仲裁期间 Sr 位后缺少 0b111_1110 + R）：

目标在动态地址仲裁期间检测到 Sr 位后缺少 0b111_1110 + R

1101：TE5 错误（检测到 CCC 后的事务）：

目标检测到非法格式化 CCC

1110：TE6 错误（监视错误）：

目标检测到总线上发送的数据与预期不符

其他值：保留

35.16.12 I3C 接收消息寄存器 (I3C_RMR)

I3C received message register

偏移地址：0x040

复位值：0x0000 0000

当 I3C 用作控制器时，该只读寄存器用于记录接收到的目标地址以及 IBI 接收到的有效负载数据大小。

当 I3C 用作目标时，该只读寄存器用于记录接收到的 CCC 代码。

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

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

位 31:24 保留，必须保持复位值。

位 23:17 **RADD[6:0]**：接收到的目标地址（当 I3C 配置为控制器时）(Received target address (when the I3C is configured as controller))

当 I3C 配置为控制器时，该位域用于记录在已确认的 IBI 或控制器角色请求期间从目标接收到的动态地址。

位 16 保留，必须保持复位值。

位 15:8 **RCODE[7:0]**：接收到的 CCC 代码（当 I3C 配置为目标时）(Received CCC code (when the I3C is configured as target))

当 I3C 配置为目标时，该位域用于记录接收到的 CCC 代码。

位 7:3 保留，必须保持复位值。

位 2:0 **IBIRDCNT[2:0]**：IBI 接收到的有效负载数据计数（当 I3C 配置为控制器时）(IBI received payload data count (when the I3C is configured as controller))

当 I3C 配置为控制器时，该位域用于记录 I3C_IBIDR 寄存器中有效接收的数据字节数。

35.16.13 I3C 事件寄存器 (I3C_EVR)

I3C event register

偏移地址：0x050

复位值：0x0000 0003

这是一个只读寄存器，用于报告事件标志。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GRPF	DEFF	INT UPDF	AS UPDF	RSTF	MRL UPDF	MWL UPDF	DA UPDF	STAF	GETF	WKPF	Res.	HJF	CR UPDF	CRF	IBI ENDF
r	r	r	r	r	r	r	r	r	r	r		r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IBIF	Res.	Res.	Res.	ERRF	RXTGT ENDF	FCF	Res.	RX LASTF	TX LASTF	RX FNEF	TX FNFF	SFNEF	CFNFF	TXFEF	CFEF
r				r	r	r		r	r	r	r	r	r	r	r

位 31 **GRPF**：组寻址标志（当 I3C 用作目标时）(Group addressing flag (when the I3C acts as target))

当 I3C 用作目标（通常支持控制器角色）时，该标志由硬件置 1，用于指示已接收到广播 DEFGRPA CCC（定义组地址列表）。然后，软件可以存储接收到的数据，以便在获得控制器角色时使用。

该标志的清零方式是通过软件向 I3C_CR 寄存器中的相应 CGRPF 位写入 1。

位 30 **DEFF**：DEFTGTS 标志（当 I3C 用作目标时）(DEFTGTS flag (when the I3C acts as target))

当 I3C 用作目标（通常支持控制器角色）时，该标志由硬件置 1，用于指示已接收到广播 DEFTGTS CCC（定义目标列表）。然后，软件可以存储接收到的数据，以便在获得控制器角色时使用。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CDEFF 位写入 1。

位 29 **INTUPDF**：中断/控制器角色/热加入更新标志（当 I3C 用作目标时）(Interrupt/controller-role/hot-join update flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已接收到直接或广播 ENEC/DISEC CCC（使能/禁止目标事件），其中目标事件为中断/IBI 请求、控制器角色请求或热加入请求。

然后，软件必须分别读取 I3C_DEVR0 寄存器中的 IBIEN、CREN 或 HJEN。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CINTUPDF 位写入 1。

位 28 **ASUPDF**：工作状态更新标志（当 I3C 用作目标时）(Activity state update flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已接收到直接或广播 ENTASx CCC（其中 x = 0...3）。然后，软件必须读取 I3C_DEVR0 寄存器中的 AS[1:0]。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CASUPDF 位写入 1。

位 27 **RSTF**：复位时序标志（当 I3C 用作目标时）(Reset pattern flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已检测到复位时序（SCL 为低电平时发生 14 次 SDA 跳变，后跟重复起始位和停止位）。

然后，当不处于停止模式时，软件必须读取 I3C_DEVR0 寄存器中的 RSTACT[1:0] 和 RSTVAL，以获取所需的复位级别。

- 如果 RSTVAL = 1：当 RSTF 置 1（和/或生成相应的中断（如果使能））时，I3C_DEVR0 寄存器中的 RSTACT[1:0] 指示软件要执行的复位操作（如果有）。
- 如果 RSTVAL = 0：当 RSTF 置 1（和/或生成相应的中断（如果使能））时，软件必须在第一个检测到的复位时序后发出 I3C 复位，并在第二个检测到的复位时序后发出系统复位。

当处于停止模式时，可以使用相应的中断来唤醒器件。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CRSTF 位写入 1。

位 26 **MRLUPDF**：最大读取长度更新标志（当 I3C 用作目标时）(Maximum read length update flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已接收到直接 SETMRL CCC（设置最大读取长度）。然后，软件必须读取 I3C_MAXRLR 寄存器中的 MRL[15:0] 以获取最大读取长度值。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CMRLUPDF 位写入 1。

位 25 **MWLUPDF**：最大写入长度更新标志（当 I3C 用作目标时）(Maximum write length update flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已接收到直接 SETMWL CCC（设置最大写入长度）。然后，软件必须读取 I3C_MAXRLR 寄存器中的 MWL[15:0] 以获取最大写入长度值。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CMWLUPDF 位写入 1。

位 24 **DAUPDF**：动态地址更新标志（当 I3C 用作目标时）(Dynamic address update flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已通过广播 ENTDAA、RSTDAA 和直接 SETNEWDA CCC 中的任何一个接收到动态地址更新。然后，软件必须读取 I3C_DEVR0 寄存器中的 DA[6:0] 和 DAVAL 以获取动态地址更新。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CDAUPDF 位写入 1。

位 23 **STAF**：获取状态标志（当 I3C 用作目标时）(Get status flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已接收到格式 1 的直接 GETSTATUS CCC（无释义字节或有释义字节 TGTSTAT）。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CSTAF 位写入 1。

位 22 **GETF**：获取标志（当 I3C 用作目标时）(Get flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已接收到除格式 1 的 GETSTATUS 之外（但包括格式 2 的 GETSTATUS）的任意获取类型的直接 CCC (GET*** CCC)。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CGETF 位写入 1。

位 21 **WKPF**：唤醒/丢失起始位标志（当 I3C 用作目标时）(Wake-up/missed start flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已检测到起始位（SDA 下降沿后跟 SCL 下降沿），但在下一个 SCL 下降沿，I3C 内核时钟（仍然）被门控。因此，目标可能已丢失 I3C 总线事务。

相应的中断可用于将器件从低功耗（睡眠或停止）模式唤醒。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CWKPF 位写入 1。

位 20 保留，必须保持复位值。

位 19 **HJF**：热加入标志（当 I3C 用作控制器时）(Hot-join flag (when the I3C acts as controller))

当 I3C 用作控制器时，该标志由硬件置 1，用于指示已接收到热加入请求。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CHJF 位写入 1。

位 18 **CRUPDF**：控制器角色更新标志（当 I3C 用作目标时）(Controller-role update flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示在完成控制器角色切换过程后 I3C 现已获得控制器角色。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CCRUPDF 位写入 1。

位 17 **CRF**：控制器角色请求标志（当 I3C 用作控制器时）(Controller-role request flag (when the I3C acts as controller))

当 I3C 用作控制器时，该标志由硬件置 1，用于指示硬件已确认并完成控制器角色请求。然后，软件必须为控制器角色切换过程发出 GETACCCR CCC（获取接受控制器角色）。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CCRF 位写入 1。

位 16 **IBIENDF**：IBI 结束标志（当 I3C 用作目标时）(IBI end flag (when the I3C acts as target))

当 I3C 用作目标时，该标志由硬件置 1，用于指示已接收到并完成 IBI 传输（控制器确认 IBI 并读取 IBI 数据字节（如果有））。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CIBIENDF 位写入 1。

位 15 **IBIF**：IBI 标志（当 I3C 用作控制器时）(IBI flag (when the I3C acts as controller))

当 I3C 用作控制器时，该标志由硬件置 1，用于指示已接收到 IBI 请求。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CIBIF 位写入 1。

位 14:12 保留，必须保持复位值。

位 11 **ERRF**：标志（I3C 用作控制器/目标）(Flag (whatever the I3C acts as controller/target))

该标志由硬件置 1，用于指示发生错误。然后，软件必须读取 I3C_SER 以获取错误类型。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CERRF 位写入 1。

位 10 **RXTGTENDF**：目标发起的读结束标志（当 I3C 用作控制器时）(Target-initiated read end flag (when the I3C acts as controller))

当 I3C 用作控制器时，并且只有在禁止 S-FIFO（I3C_CFGR 寄存器中的 SMODE = 0）时，该标志才会由硬件置 1，用于指示目标已提前结束读传输。然后，软件必须读取状态寄存器 I3C_SR 以检查与最后一条消息相关的信息，并获取提前结束的读传输中接收到的数据字节数（I3C_SR 寄存器中的 XDCNT）。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CRXTGTENDF 位写入 1。

位 9 **FCF**：帧完成标志（I3C 用作控制器/目标）(Frame complete flag (whatever the I3C acts as controller/target))

当 I3C 用作控制器时，该标志由硬件置 1，用于指示 I3C 总线上的帧已（正常）完成，例如在发出停止位时。

当 I3C 用作目标时，该标志由硬件置 1，用于指示在 I3C 总线上发送至该目标或由该目标寻址的消息已（正常）完成，例如在控制器发出下一个停止位或重复起始位时。

该标志的清零方式是通过软件向 I3C_CEVR 寄存器中的相应 CFCF 位写入 1。

位 8 保留，必须保持复位值。

位 7 **RXLASTF**：最后读取的数据字节/字标志（当 I3C 用作控制器时）(Last read data byte/word flag (when the I3C acts as controller))

当 I3C 用作控制器时，该标志由硬件置 1，用于指示必须从 RX-FIFO 读取的消息的最后一个数据字节/字（具体取决于 I3C_CFGR 寄存器中的 RXTHRES）。当读取消息的最后一个数据字节/字时，该标志由硬件清零。

位 6 **TXLASTF**：最后写入的数据字节/字标志（I3C 用作控制器/目标）(Last written data byte/word flag (whatever the I3C acts as controller/target))

该标志由硬件置 1，用于指示必须写入 TX-FIFO 的消息的最后一个数据字节/字（具体取决于 I3C_CFGR 寄存器中的 TXTHRES）。当写入消息的最后一个数据字节/字时，该标志由硬件清零。

位 5 **RXFNEF**：RX-FIFO 非空标志（I3C 用作控制器/目标）(RX-FIFO not empty flag (whatever the I3C acts as controller/target))

该标志由硬件置 1/清零，用于指示必须/不得从 RX-FIFO 读取数据字节。

注意：在从 RX-FIFO 读取数据字节（读取 I3C_RDR 或 I3C_RDWR，具体取决于 RXTHRES）之前，软件必须等待 RXFNEF = 1（通过轮询或通过使能的中断）。

位 4 **TXFNFF**：TX-FIFO 未满标志（I3C 用作控制器/目标）(TX-FIFO not full flag (whatever the I3C acts as controller/target))

该标志由硬件置 1/清零，用于指示必须/不得向 TX-FIFO 写入数据字节。

注意：在写入 TX-FIFO（写入 I3C_TDR 或 I3C_TDWR，具体取决于 TXTHRES）之前，软件必须等待 TXFNFF = 1（通过轮询或通过使能的中断）。

注意：当 I3C 用作目标时，如果软件要使用 TXFNFF 标志写入 I3C_TDR/I3C_TDWR，则必须配置 TX-FIFO 预装载（将 I3C_TGTTDR 寄存器中的 PRELOAD 置 1）。

位 3 **SFNEF**：S-FIFO 非空标志（当 I3C 用作控制器时）(S-FIFO not empty flag (when the I3C acts as controller))

当 I3C 用作控制器时，如果 S-FIFO 已使能（I3C_CFGR 寄存器中的 SMODE = 1），则该标志由硬件置 1，用于指示必须从 S-FIFO 读取状态字。该标志由硬件清零，用于指示不从 S-FIFO 读取状态字。

位 2 **CFNFF**：C-FIFO 未满标志（当 I3C 用作控制器时）(C-FIFO not full flag (when the I3C acts as controller))

当 I3C 用作控制器时，该标志由硬件置 1，用于指示必须向 C-FIFO 写入控制字。该标志由硬件清零，用于指示不向 C-FIFO 写入控制字。

注意：在写入 C-FIFO（写入 I3C_CR）之前，软件必须等待 CFNFF = 1（通过轮询或通过使能的中断）。

位 1 **TXFEF**：TX-FIFO 为空标志（I3C 用作控制器/目标）(TX-FIFO empty flag (whatever the I3C acts as controller/target))

该标志由硬件置 1，用于指示 TX-FIFO 为空。
该标志由硬件清零，用于指示 TX-FIFO 不为空。

位 0 **CFEF**：C-FIFO 为空标志（I3C 用作控制器/目标）(C-FIFO empty flag (whatever the I3C acts as controller/target))

该标志由硬件置 1，用于指示 C-FIFO 为空（用作控制器时），或者 I3C_CR 寄存器中不包含控制字（无 IBI/CR/HJ 请求）（用作目标时）。
该标志由硬件清零，用于指示 C-FIFO 不为空（用作控制器时），或者 I3C_CR 寄存器中包含一个控制字（待处理的 IBI/CR/HJ 请求）（用作目标时）。

注意：当 I3C 用作控制器时，如果配置了 C-FIFO 和 TX-FIFO 预装载（I3C_CFGR 寄存器中的 TMODE = 1），则软件必须等待 TXFEF = 1 和 CFEF = 1，然后才能开始新的帧传输。

35.16.14 I3C 中断使能寄存器 (I3C_IER)

I3C interrupt enable register

偏移地址：0x054

复位值：0x0000 0000

该寄存器用于按位使能/禁止以下每个事件/标志的中断。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GRPIE	DEFIE	INTUPDIE	ASUPDIE	RSTIE	MRLUPDIE	MWLUPDIE	DAUPDIE	STAIE	GETIE	WKPIE	Res.	HJIE	CRUPDIE	CRIE	IBIENDIE
r	r	r	r	r	r	r	r	r	r	r		r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IBIIE	Res.	Res.	Res.	ERRIE	RXTGTENDIE	FCIE	Res.	Res.	Res.	RXFNEIE	TXFNFIE	SFNEIE	CFNFIE	Res.	Res.
r				r	r	r				r	r	r	r		

位 31 **GRPIE**：DEFGRPA CCC 中断使能（当 I3C 用作目标时）(DEFGRPA CCC interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 30 **DEFIE**：DEFTGTS CCC 中断使能（当 I3C 用作目标时）(DEFTGTS CCC interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 29 **INTUPDIE**：ENEC/DISEC CCC 中断使能（当 I3C 用作目标时）(ENEC/DISEC CCC interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 28 **ASUPDIE**：ENTASx CCC 中断使能（当 I3C 用作目标时）(ENTASx CCC interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 27 **RSTIE**：复位时序中断使能（当 I3C 用作目标时）(reset pattern interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 26 **MRLUPDIE**：SETMRL CCC 中断使能（当 I3C 用作目标时）(SETMRL CCC interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 25 **MWLUPDIE**：SETMWL CCC 中断使能（当 I3C 用作目标时）(SETMWL CCC interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 24 **DAUPDIE**：ENTDAA/RSTDAA/SETNEWDA CCC 中断使能（当 I3C 用作目标时）(ENTDAA/RSTDAA/SETNEWDA CCC interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 23 **STAIE**：格式 1 GETSTATUS CCC 中断使能（当 I3C 用作目标时）(format 1 GETSTATUS CCC interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 22 **GETIE**：GETxxx CCC 中断使能（格式 1 的 GETSTATUS 除外）（当 I3C 用作目标时）(GETxxx CCC interrupt enable (except GETSTATUS of format 1) (when the I3C acts as target))

0：禁止中断
1：使能中断

位 21 **WKPIE**：唤醒中断使能（当 I3C 用作目标时）(Wake-up interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 20 保留，必须保持复位值。

位 19 **HJIE**：热加入中断使能（当 I3C 用作控制器时）(Hot-join interrupt enable (when the I3C acts as controller))

0：禁止中断
1：使能中断

位 18 **CRUPDIE**：控制器角色更新中断使能（当 I3C 用作目标时）(Controller-role update interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 17 **CRIE**：控制器角色请求中断使能（当 I3C 用作控制器时）(Controller-role request interrupt enable (when the I3C acts as controller))

0：禁止中断
1：使能中断

位 16 **IBIENDIE**：IBI 结束中断使能（当 I3C 用作目标时）(IBI end interrupt enable (when the I3C acts as target))

0：禁止中断
1：使能中断

位 15 **IBIIE**：IBI 请求中断使能（当 I3C 用作控制器时）(IBI request interrupt enable (when the I3C acts as controller))

0：禁止中断
1：使能中断

位 14:12 保留，必须保持复位值。

位 11 **ERRIE**：错误中断使能（I3C 用作控制器/目标）(error interrupt enable (whatever the I3C acts as controller/target))

0：禁止中断
1：使能中断

位 10 **RXTGTENDIE**：目标发起的读结束中断使能（当 I3C 用作控制器时）(target-initiated read end interrupt enable (when the I3C acts as controller))

0：禁止中断
1：使能中断

位 9 **FCIE**：帧完成中断使能（I3C 用作控制器/目标）(frame complete interrupt enable (whatever the I3C acts as controller/target))

0：禁止中断
1：使能中断

位 8:6 保留，必须保持复位值。

位 5 **RXFNEIE**：RX-FIFO 非空中断使能（I3C 用作控制器/目标）(RX-FIFO not empty interrupt enable (whatever the I3C acts as controller/target))

0：禁止中断
1：使能中断

位 4 **TXFNFIE**：TX-FIFO 未满中断使能（I3C 用作控制器/目标）(TX-FIFO not full interrupt enable (whatever the I3C acts as controller/target))

0：禁止中断
1：使能中断

位 3 **SFNEIE**：S-FIFO 非空中断使能（当 I3C 用作控制器时）(S-FIFO not empty interrupt enable when the I3C acts as controller)

0：禁止中断
1：使能中断

位 2 **CFNFIE**：C-FIFO 未满中断使能（当 I3C 用作控制器时）(C-FIFO not full interrupt enable when the I3C acts as controller)

0：禁止中断
1：使能中断

位 1:0 保留，必须保持复位值。

35.16.15 I3C 清零事件寄存器 (I3C_CEVR)

I3C clear event register

偏移地址：0x058

复位值：0x0000 0000

该只写寄存器用于单独（按位）将 I3C_EVR 寄存器的相应事件标志清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CGRPF	CDEFF	CINTUPDF	CASUPDF	CRSTF	CMRLUPDF	CMWLUPDF	CDAUPDF	CSTAF	CGETF	CWKPF	Res.	CHJF	CCRUPDF	CCRF	CIBIENDF
w	w	w	w	w	w	w	w	w	w	w		w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CIBIF	Res.	Res.	Res.	CERRF	CRXTGTENDF	CFCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
w				w	w	w									

位 31 **CGRPF**：清零 DEFGRPA CCC 标志（当 I3C 用作目标时）(Clear DEFGRPA CCC flag (when the I3C acts as target))

0：无影响

1：将 GRPF 清零

位 30 **CDEFF**：清零 DEFTGTS CCC 标志（当 I3C 用作目标时）(Clear DEFTGTS CCC flag (when the I3C acts as target))

0：无影响

1：将 DEFF 清零

位 29 **CINTUPDF**：清零 ENEC/DISEC CCC 标志（当 I3C 用作目标时）(Clear ENEC/DISEC CCC flag (when the I3C acts as target))

0：无影响

1：将 CINTUPDF 清零

位 28 **CASUPDF**：清零 ENTASx CCC 标志（当 I3C 用作目标时）(Clear ENTASx CCC flag (when the I3C acts as target))

0：无影响

1：将 ASUPDF 清零

位 27 **CRSTF**：清零复位时序标志（当 I3C 用作目标时）(Clear reset pattern flag (when the I3C acts as target))

0：无影响

1：将 RSTF 清零

位 26 **CMRLUPDF**：清零 SETMRL CCC 标志（当 I3C 用作目标时）(Clear SETMRL CCC flag (when the I3C acts as target))

0：无影响

1：将 MRLUPDF 清零

位 25 **CMWLUPDF**：清零 SETMWL CCC 标志（当 I3C 用作目标时）(Clear SETMWL CCC flag (when the I3C acts as target))

0：无影响

1：将 MWLUPDF 清零

位 24 **CDAUPDF**：清零 ENTDAA/RSTDAA/SETNEWDA CCC 标志（当 I3C 用作目标时）(Clear ENTDAA/RSTDAA/SETNEWDA CCC flag (when the I3C acts as target))

0：无影响
1：将 DAUPDF 清零

位 23 **CSTAF**：清零格式 1 GETSTATUS CCC 标志（当 I3C 用作目标时）(Clear format 1 GETSTATUS CCC flag (when the I3C acts as target))

0：无影响
1：将 STAF 清零

位 22 **CGETF**：清零 GETxxx CCC 标志（格式 1 的 GETSTATUS 除外）（当 I3C 用作目标时）(Clear GETxxx CCC flag (except GETSTATUS of format 1) (when the I3C acts as target))

0：无影响
1：将 GETF 清零

位 21 **CWKPF**：清零唤醒标志（当 I3C 用作目标时）(Clear wake-up flag (when the I3C acts as target))

0：无影响
1：将 WKPF 清零

位 20 保留，必须保持复位值。

位 19 **CHJF**：清零热加入标志（当 I3C 用作控制器时）(Clear hot-join flag (when the I3C acts as controller))

0：无影响
1：将 HJF 清零

位 18 **CCRUPDF**：清零控制器角色更新标志（当 I3C 用作目标时）(Clear controller-role update flag (when the I3C acts as target))

0：无影响
1：将 CRUPDF 清零

位 17 **CCRF**：清零控制器角色请求标志（当 I3C 用作控制器时）(Clear controller-role request flag (when the I3C acts as controller))

0：无影响
1：将 CRF 清零

位 16 **CIBIENDF**：清零 IBI 结束标志（当 I3C 用作目标时）(Clear IBI end flag (when the I3C acts as target))

0：无影响
1：将 IBIENDF 清零

位 15 **CIBIF**：清零 IBI 请求标志（当 I3C 用作控制器时）(Clear IBI request flag (when the I3C acts as controller))

0：无影响
1：将 IBIF 清零

位 14:12 保留，必须保持复位值。

位 11 **CERRF**：清零错误标志（I3C 用作控制器/目标）(Clear error flag (whatever the I3C acts as controller/target))

0：无影响
1：将 ERRF 清零

位 10 **CRXTGTENDF**：清零目标发起的读结束标志（当 I3C 用作控制器时）(Clear target-initiated read end flag (when the I3C acts as controller))

0：无影响
1：将 RXTGTENDF 清零

位 9 **CFCF**：清零帧完成标志（I3C 用作控制器/目标）(Clear frame complete flag (whatever the I3C acts as controller/target))

0：无影响

1：将 FCF 清零

位 8:0 保留，必须保持复位值。

35.16.16 I3C 自身器件特性寄存器 (I3C_DEVR0)

I3C own device characteristics register

偏移地址：0x060

复位值：0x0000 0000

当 I3C 外设用作目标时，该寄存器用于写入或读取其自身的器件特性。

当 I3C 外设用作控制器时，位域 DA[6:0] 用于写入和存储其自身的动态地址。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSTVAL	RSTACT[1:0]		AS[1:0]		HJEN	Res.	CREN	IBIEN
							r	r	r	r	r	rw		rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DA[6:0]							DAVAL
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:25 保留，必须保持复位值。

位 24 **RSTVAL**：复位操作有效（当 I3C 用作目标时）(Reset action is valid (when the I3C acts as target))

该位由硬件置 1，用于指示 RSTACT[1:0] 位域已在接收到广播或直接写 RSTACT CCC（目标复位操作）时更新并生效。

当目标接收到帧起始时，该位由硬件清零。

当器件不处于停止模式时：

- 如果 RSTVAL = 1：当 I3C_EVR 寄存器中的 RSTF 置 1（和/或生成相应的中断（如果使能））时，I3C_DEVR0 寄存器中的 RSTACT[1:0] 指示软件要执行的复位操作（如果有）。
- 如果 RSTVAL = 0：当 RSTF 置 1（和/或生成相应的中断（如果使能））时，软件必须在第一个检测到的复位时序后发出 I3C 复位，并在第二个检测到的复位时序后发出系统复位。

当处于停止模式时，可以使用相应的中断来唤醒器件。

位 23:22 **RSTACT[1:0]**：接收到复位时序时的复位操作/级别（当 I3C 用作目标时）(Reset action/level on received reset pattern (when the I3C acts as target))

该只读位域由硬件在接收到直接读 RSTACT CCC 时使用，以便在 I3C 总线上返回相应的数据字节。

该只读位域由硬件在接收到广播或直接写 RSTACT CCC（目标复位操作）时更新。

仅映射释义字节 0x00、0x01 和 0x02，RSTACT[1:0] = 释义字节[1:0]。

00：不执行任何复位操作

01：第一级复位：应用软件必须：

a) 通过将 I3C 配置寄存器的使能位清零（写入 EN = 0）对外设进行部分复位。这会复位 I3C 总线接口和 I3C 内核的一部分，但不会修改 I3C APB 寄存器的内容（EN 位除外）。

b) 通过写操作将 RCC（复位和时钟控制器）寄存器的 I3C 复位控制位置 1，对外设（包括其所有寄存器）进行完全复位。

10：第二级复位：应用软件必须发出热复位，也称为系统复位。这（请参见*第 10 章：复位和时钟控制 (RCC)*）与引脚复位 (NRST = 0) 的作用相同：

- 当器件由 Cortex®-M 控制时，软件会通过写操作将 AITR 寄存器的 SYSRESETREQ 控制位置 1。

- 当器件由 Cortex®-A 控制时，软件会通过写操作将 RCC_GRSTCSETR 寄存器中的 SYSRST 置 1。

11：不执行任何复位操作

位 21:20 **AS[1:0]**：工作状态（当 I3C 用作目标时）(Activity state (when the I3C acts as target))

该只读位域由硬件在接收到 ENTASx CCC 时更新（进入工作状态，其中 x = 0 到 3）：

00：工作状态 0

01：工作状态 1

10：工作状态 2

11：工作状态 3

位 19 **HJEN**：热加入请求使能（当 I3C 用作目标时）(Hot-join request enable (when the I3C acts as target))

该位最初由软件在 EN = 0 时写入，然后在接收到 DISEC CCC (DISHJ = 1) 时由硬件清零，在接收到 ENEC CCC (ENHJ = 1) 时由硬件置 1。只有在 I3C_CFGR 寄存器中 EN = 0 时，才能通过软件写入该位。

0：禁止热加入请求

1：使能热加入请求

位 18 保留，必须保持复位值。

位 17 **CREN**：控制器角色请求使能（当 I3C 用作目标时）(Controller-role request enable (when the I3C acts as target))

该位最初由软件在 EN = 0 时写入，然后在接收到 DISEC CCC (DISCR = 1) 时由硬件清零，在接收到 ENEC CCC (ENCR = 1) 时由硬件置 1。只有在 I3C_CFGR 寄存器中 EN = 0 时，才能通过软件写入该位。

0：禁止控制器角色请求

1：使能控制器角色请求

位 16 **IBIEN**：IBI 请求使能（当 I3C 用作目标时）(IBI request enable (when the I3C acts as target))

该位最初由软件在 EN = 0 时写入，然后在接收到 DISEC CCC (DISINT = 1) 时由硬件清零，在接收到 ENEC CCC (ENINT = 1) 时由硬件置 1。只有在 I3C_CFGR 寄存器中 EN = 0 时，才能通过软件写入该位。

0：禁止 IBI 请求

1：使能 IBI 请求

位 15:8 保留，必须保持复位值。

位 7:1 **DA[6:0]**：7 位动态地址 (7-bit dynamic address)

当 I3C 用作控制器时，该位域可由软件写入，用于定义其自身的动态地址。

当 I3C 用作目标时，该位域由硬件在接收到广播 ENTDAA CCC 或直接 SETNEWDA CCC 时更新。

位 0 **DAVAL**：动态地址有效（当 I3C 用作目标时）(Dynamic address is valid (when the I3C acts as target))

当 I3C 用作控制器时，该位可由软件写入，用于验证其自身的动态地址，例如在控制器角色切换之前。

当 I3C 用作目标时，该位由硬件在确认广播 ENTDAA CCC 或直接 SETNEWDA CCC 时置 1，在确认广播 RSTDAA CCC 时清零。

35.16.17 I3C 设备 x 特性寄存器 (I3C_DEVRx)

I3C device x characteristics register

偏移地址：0x060 + 0x4 * x，（x = 1 到 4）

复位值：0x0000 0000

当 I3C 外设用作控制器时，该寄存器用于定义和存储器件目标 x 的一些特性及其相关的控制器管理，以便通过 I3C 总线与该目标 x 进行相应的通信。然后，硬件可以自主识别并确认来自目标 x 的有效 IBI 或/和控制器角色请求，接收预期的 IBI 有效负载数据（如果有），并通过 I3C_EVR 寄存器中的相应标志 IBIF/CRF（和相应的中断（如果使能））通知软件。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUSP	IBIDEN	CRACK	IBIACK
r												rw	rw	rw	rw

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

位 31 **DIS**：DA[6:0] 写禁止（当 I3C 用作控制器时）(DA[6:0] write disabled (when the I3C acts as controller))

当 I3C 用作控制器时，软件将 IBIACK 或 CRACK 置 1 后，该只读位就会由硬件置 1 (DIS = 1) 以锁定配置的 DA[6:0] 和 IBIDEN 值。

然后，为了能够修改 DA[6:0]、IBIDEN 或 SUSP，软件必须等待 DIS 由硬件清零（轮询 DIS = 0），之后才能修改这三个为目标 x 分配的值。事实上，当控制器要修改 DA[6:0]、IBIDEN 或 SUSP 时，目标可以请求 IBI 或控制器角色。

0：允许对 I3C_DEVRx 寄存器中的 DA[7:0] 和 IBIDEN 的写操作

1：禁止/锁定对 DA[7:0] 和 IBIDEN 的写操作

位 30:20 保留，必须保持复位值。

位 19 **SUSP**：收到 IBI 时挂起/停止 I3C 传输（当 I3C 用作控制器时）(Suspend/stop I3C transfer on received IBI (when the I3C acts as controller))

当 I3C 用作控制器时，该位可用于从具有挂起读取通知功能的目标 x 接收 IBI（接收到的 MDB[7:5] = 3'b101）。

如果该位置 1，当接收到 IBI 请求并完成（I3C_EVR 寄存器中的 IBIF = 1）时，I3C 总线上会发出停止位，并且硬件会自动清空 C-FIFO 和 TX-FIFO。即使编程了下一条控制消息，控制器执行流程也会停止。当 IBI 完成时，控制器软件可以向发起 IBI 请求的目标器件发出新的控制字（例如私有读）。

0：确认并完成来自目标 x 的 IBI 请求后，不清空 C-FIFO 和 TX-FIFO，并根据是否存在下一个控制字，发出重复起始位或停止位

1：接收到来自目标 x 的 IBI 请求后，停止 I3C 传输，并清空 C-FIFO 和 TX-FIFO

位 18 **IBIDEN**：IBI 数据使能（当 I3C 用作控制器时）(IBI data enable (when the I3C acts as controller))

当 I3C 用作控制器时，该位必须由软件写入，用于存储在广播 ENTDAA 或直接 GETBCR CCC 期间通过接收到的 I3C_RDR 从目标 x 接收到的 BCR[2] 位。

当 I3C_DEVRx 寄存器中的 DIS = 1 时，写入该位域没有任何影响。

0：来自目标 x 的已确认 IBI 后没有数据字节

1：来自目标 x 的已确认 IBI 后跟强制数据字节 MDB[7:0]

位 17 **CRACK**：控制器角色请求确认（当 I3C 用作控制器时）(Controller-role request acknowledge (when the I3C acts as controller))

当 I3C 用作控制器时，该位由软件写入，用于定义在接收来自目标 x 的控制器角色请求时应用于 I3C 总线的确认策略：

0：必须 NACK 来自目标 x 的控制器角色请求

NACK 后，消息按照初始编程的方式继续传输（目标已接收到 NACK，稍后可发出另一个控制器角色请求）

1：必须 ACK 来自目标 x 的控制器角色请求（通过 7 位动态地址 DA[6:0]）

- DIS 位域由硬件置 1，用于保护 DA[6:0] 免受软件修改，同时硬件可以在内部将当前的 DA[6:0] 存储到内核时钟域中。

- ACK 后，消息按照初始编程的方式继续传输。软件通过控制器角色请求标志（I3C_EVR 寄存器中的 CRF = 1）和/或相应的中断（如果使能）获得通知；为了将控制器角色有效地授予请求的辅助控制器，软件必须发出 GETACCCR（以前称为 GETACCMST），然后发出停止位。

- 不管该器件或其他器件的 CRACK 配置为多少，其他控制器角色请求都将被 NACK，直到 I3C_EVR 寄存器中的控制器角色请求标志 (CRF) 和 IBI 标志 (IBIF) 都清零。

位 16 **IBIACK**：IBI 请求确认（当 I3C 用作控制器时）(IBI request acknowledge (when the I3C acts as controller))

当 I3C 用作控制器时，该位由软件写入，用于定义在接收来自目标 x 的 IBI 请求时应用于 I3C 总线的确认策略：

0：必须 NACK 来自目标 x 的 IBI 请求

NACK 后，消息按照初始编程的方式继续传输（目标已接收到 NACK，稍后可发出另一个 IBI 请求）

1：必须 ACK 来自目标 x 的 IBI 请求（通过 7 位动态地址 DA[6:0]）

- DIS 位域由硬件置 1，用于保护 DA[6:0] 免受软件修改，同时硬件可以在内部将当前的 DA[6:0] 存储到内核时钟域中。

- ACK 后，控制器根据 I3C_DEVRx.IBIDEN 记录 IBI 有效负载数据（如果有）。

- 软件通过 IBI 标志 (IBIF = 1) 和/或相应的中断（如果使能）获得通知；

- 不管该器件或其他器件的 IBIACK 配置为多少，其他 IBI 请求都将被 NACK，直到 IBI 请求标志 (IBIF) 和控制器角色请求标志 (CRF) 都清零。

位 15:8 保留，必须保持复位值。

位 7:1 **DA[6:0]**：为目标 x 分配的 I3C 动态地址（当 I3C 用作控制器时）(Assigned I3C dynamic address to target x (when the I3C acts as controller))

当 I3C 用作控制器时，该位域必须由软件写入，用于存储控制器通过目标 x 确认的广播 ENTDAA 或直接 SETNEWDA CCC 发送的 7 位动态地址。

当 I3C_DEVRx 寄存器中的只读位域 DIS = 1 时，写入该位域没有任何影响。

位 0 保留，必须保持复位值。

35.16.18 I3C 最大读取长度寄存器 (I3C_MAXRLR)

I3C maximum read length register

偏移地址：0x090

复位值：0x0000 0000

当 I3C 用作目标时，该寄存器用于设置或获取分别在 GETMRL 或 SETMRL CCC 期间与控制器交换的最大读取长度值。该寄存器还用于设置 IBI 数据有效负载大小。

当 I3C_CFGR 寄存器中的 EN = 0 时，可通过软件写入该寄存器。

接收私有读消息时，如果发送数据的计数达到 I3C_MAXRLR 寄存器中的 MRL[15:0]，目标将结束数据发送（通过驱动 T 位 = 0）。

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

位 31:19 保留，必须保持复位值。

位 18:16 **IBIP[2:0]**：IBI 负载数据最大大小，以字节为单位（当 I3C 作为目标时）（IBI payload data maximum size, in bytes (when I3C acts as target))

该位域最初由软件在 EN = 0 时写入，用于设置在确认 IBI 请求后发送到控制器的最大数据字节数。该位域可由硬件在接收到 SETMRL 命令时更新（还可能更新 IBIP[2:0]）。

通过 I3C_EVR 寄存器中的 MRLUPF 和相应的中断（如果使能）向软件通知 MRL 更新。

000：空负载数据大小（仅当 IC3_BCR 寄存器中的 BCR2 = 0 时允许）

001：1 字节（强制数据字节 MDB[7:0]）

010：2 字节（包括第一个 MDB[7:0]）

011：3 字节（包括第一个 MDB[7:0]）

100：4 字节（包括第一个 MDB[7:0]）

其他值：与 100 相同

位 15:0 **MRL[15:0]**：最大数据读取长度（当 I3C 作为目标时）(Maximum data read length (when I3C acts as target))

该位域最初由软件在 EN = 0 时写入，在接收到 SETMRL 命令时由硬件更新（也可能更新 IBIP[2:0]）。

通过 MRLUPF 和相应的中断（如果使能）向软件通知 MRL 更新。

当目标接收到 GETMRL CCC 时，硬件使用该位域在 I3C 总线上返回值。

35.16.19 I3C 最大写入长度寄存器 (I3C_MAXWLR)

I3C maximum write length register

偏移地址：0x094

复位值：0x0000 0000

当 I3C 用作目标时，该寄存器用于设置或获取分别在 GETMWL 或 SETMWL CCC 期间与控制器交换的最大写入长度值。

当 I3C_CFGR 寄存器中的 EN = 0 时，可通过软件写入该寄存器。

接收私有写消息时，如果接收到的数据计数达到 I3C_MAXWLR 寄存器中的 MWL[15:0]，目标将停止数据接收（额外接收的数据不会写入 RX-FIFO）。

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

位 31:16 保留，必须保持复位值。

位 15:0 **MWL[15:0]**：最大数据写入长度（当 I3C 作为目标时）（Maximum data write length (when I3C acts as target)）

该位域最初由软件在 EN = 0 时写入，在接收到 SETMWL 命令时由硬件更新。
通过 I3C_EVR 寄存器中的 MWLUPF 和相应的中断（如果使能）向软件通知 MWL 更新。
当目标接收到 GETMWL CCC 时，硬件使用该位域在 I3C 总线上返回值。

35.16.20 I3C 时序寄存器 0 (I3C_TIMINGR0)

I3C timing register 0

偏移地址：0x0A0

复位值：0x0000 0000

当 I3C 用作控制器时，该寄存器用于配置 SCL 时钟信号波形。

当 I3C_CFGR 寄存器中的 EN = 0 时，可通过软件写入该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SCLH_I2C[7:0]								SCLL_OD[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCLH_I3C[7:0]								SCLL_PP[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **SCLH_I2C[7:0]**：SCL 高电平持续时间，用于传统 I²C 消息，以内核时钟周期数为单位：

t_{SCLH_I2C} = (SCLH_I2C + 1) x t_{I3CCLK}

注意：SCLH_I2C 用于在与 I²C 器件通信时生成 t_{DIG_H} (I²C) 时序。

注意：I²C fm+ 器件 t_{DIG_Hmin} = 260 ns，I²C fm 器件 t_{DIG_Hmin} = 600 ns。

位 23:16 **SCLL_OD[7:0]**：开漏阶段的 SCL 低电平持续时间，用于传统 I²C 消息和 I3C 开漏阶段（起始位后的地址阶段、控制器发起的消息期间的 ACK 阶段以及直接/私有/IBI 有效负载期间的 T 位阶段），以内核时钟周期数为单位：

t_{SCLL_OD} = (SCLL_OD + 1) x t_{I3CCLK}

注意：SCLL_OD 用于生成 t_{DIG_L} (I²C) 和 $t_{DIG_OD_L}$ (I3C) 时序。

注意：I²C fm+ 器件 t_{DIG_Lmin} = 500 ns，I²C fm 器件 t_{DIG_Lmin} = 1320 ns。

注意：I3C 消息：$t_{DIG_OD_Lmin}$ = 200 ns。

注意：如果单个 I3C 帧正在收集 I²C 和 I3C 消息，I3C 开漏阶段期间的 SCL 低电平持续时间将会延长以适应 I²C 时序。

位 15:8 **SCLH_I3C[7:0]**：SCL 高电平持续时间，用于 I3C 消息（在推挽和开漏阶段），以内核时钟周期数为单位：

t_{SCLH_I3C} = (SCLH_I3C + 1) x t_{I3CCLK}

注意：SCLH_I3C 用于生成 t_{DIG_H} (I3C) 和 $t_{DIG_H_MIXED}$ 时序。

注意：对于混合总线（至少有一个 I²C 目标）：$t_{DIG_H_MIXEDmin}$ = 32 ns，$t_{DIG_H_MIXEDmax}$ = 45 ns（由于 I²C 50 ns 尖峰滤波器）。

注意：对于纯 I3C 总线（无 I²C 目标）：t_{DIG_Hmin} = 32 ns。

位 7:0 **SCLL_PP[7:0]**：I3C 推挽阶段的 SCL 低电平持续时间，以内核时钟周期数为单位：

t_{SCLL_PP} = (SCLL_PP + 1) x t_{I3CCLK}

注意：SCLL_PP 用于生成 t_{DIG_L}（PP 中的 I3C）时序。

注意：t_{DIG_Lmin} = 32 ns（12.5 MHz 时最大占空比为 40/60）。

35.16.21 I3C 时序寄存器 1 (I3C_TIMINGR1)

I3C timing register 1

偏移地址：0x0A4

复位值：0x0000 0000

当 I3C 用作控制器时，该寄存器用于配置一些 I3C 时序设置。

当 I3C 用作目标且支持控制器角色时，该寄存器用于配置控制器角色切换过程的时序。

当 I3C_CFGR 寄存器中的 EN = 0 时，可通过软件写入该寄存器。

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

位 31:29 保留，必须保持复位值。

位 28 **SDA_HD**：SDA 保持时间（当 I3C 用作控制器时），以内核时钟周期数为单位（请参见推挽模式下的 MIPI 时序 SDA 保持时间 t_{HD_PP}）：

SDA 保持时间 = (SDA_HD + 0.5) x t_{I3CCLK}

注意：当用作控制器时：t_{HD_PPmiN} = min (t_{CR}，t_{CF}) + 3 ns。

位 27:23 保留，必须保持复位值。

位 22:16 **FREE[6:0]**：用于设置某些 MIPI 时序（例如总线空闲条件时间）的内核时钟周期数（当 I3C 用作控制器时）

当 I3C 用作控制器时：

1. 对于 I3C 起始位时序：在停止位之后和起始位之前必须等待一段（总线空闲条件）时间，请参见 MIPI 时序 (I3C) t_{CAS} 和 (I^2C) t_{BUF}。这些时序定义为：t_{BUF} = t_{CAS} = [(FREE[6:0] + 1) x 2 - (0.5 + SDA_HD)] x t_{I3CCLK}

注意：对于纯 I3C 总线：t_{CASmin} = 38.4 ns，t_{CASmax} = 1 µs、100 µs、2 ms、50 ms（分别对应 ENTAS0、1、2 和 3）。

注意：对于混合总线，I^2C fm+ 器件 t_{BUFmin} = 0.5 µs，I^2C fm 器件 t_{BUFmin} = 1.3 µs。

2. 对于 I3C 重复起始位时序：在重复起始位之后（SDA 置为无效）和将 SCL 驱动为低电平之前必须等待一段时间，请参见MIPI 时序 t_{CASr}。该时序定义为： t_{CASr} = [(FREE[6:0] + 1) x 2 - (0.5 + SDA_HD)] x t_{I3CCLK}。

注意：$t_{CASr,\ miN}$ = 19.2 ns。

3. 对于 I3C 停止位时序：在将 SCL 时钟驱动为高电平之后和停止位（SDA 置为有效）之前必须等待一段时间。该时序定义为：t_{CBP} = (FREE[6:0] + 1) x t_{I3CCLK}。

注意：t_{CBPmiN} = 19.2 ns。

4. 对于 I3C 重复起始位时序（控制器结束读传输并生成重复起始位和停止位时的 T 位）：在将 SCL 时钟驱动为高电平之后和重复起始位之前（SDA 置为无效）必须等待一段时间。该时序定义为：t_{CBSr} = (FREE[6:0] + 1) x t_{I3CCLK}。

注意：$t_{CBSr,\ miN}$ = 19.2 ns。

位 15:10 保留，必须保持复位值。

位 9:8 **ASNCR[1:0]**：新控制器的工作状态（当 I3C 用作活动控制器时）(Activity state of the new controller (when I3C acts as active controller))

该位域用于指示在作为新目标进行访问之前需要等待的时间，请参见 AVAL[7:0]。

只有在 I3C 用作控制器时，才能修改该位域。

位 7:0 **AVAL[7:0]**：用于设置 1 µs 时间单位的内核时钟周期数（I3C 用作控制器/目标）。

硬件随后使用该时间单位构建一些内部定时器，对应于以下 MIPI I3C 时序：

当 I3C 用作目标时：

1. 对于总线可用条件时间：在停止位之后和发出 IBI 启动请求或控制器角色请求之前，必须等待一段（总线可用条件）时间（总线空闲条件至少持续 t_{AVAL}）。请参见 MIPI 时序 t_{AVAL} = 1 µs。该时序定义为：t_{AVAL} = (AVAL[7:0] + 2) x t_{I3CCLK}
2. 对于总线空闲条件时间：在 SDA 和 SCL 都为高电平且保持稳定之后和发出热加入事件之前，必须等待一段（总线空闲条件）时间。请参见 MIPI v1.1 时序 t_{IDLE} = 200 µs。该时序定义为：t_{IDLE} = (AVAL[7:0] + 2) x 200 x t_{I3CCLK}

当 I3C 用作控制器时，其停顿时钟的时间不得超过最大停顿时间（将 SCL 时钟停顿为低电平），具体如下：

1. 对于动态地址分配期间分配的地址的第一位：停顿时钟的时间不得超过 MIPI 时序 $t_{STALLDAA}$ = 15 ms。该时序定义为：$t_{STALLDAAmax}$ = (AVAL[7:0] + 1) x 15000 x t_{I3CCLK}
2. 对于 I3C/I^2C 传输的 ACK/NACK 阶段、写数据传输的奇偶校验位、I3C 读传输的转换位：停顿时钟的时间不得超过 MIPI 时序 t_{STALL} = 100 µs。该时序定义为：$t_{STALLmax}$ = (AVAL[7:0] + 1) x 100 x t_{I3CCLK}

无论 I3C 用作控制器还是（支持控制器角色的）目标，在控制器角色切换过程中：

1. 新控制器必须等待 $t_{NEWCRLock}$，然后才能在完成 GETACCR CCC 后将 SDA 拉低（发出起始位）。然后，新控制器可以在 t_{CAS} 内将 SCL 拉低以激活 SCL 时钟。在测试新控制器是否已通过拉低 SDA 获得总线控制权之前，活动控制器必须等待相同的 $t_{NEWCRLock}$ 时间，或至少 100 µs。等待的时间取决于 I3C_TIMINGR1 寄存器中 ANSCR[1:0] 的值：
   - ASNCR[1:0] = 00：$t_{NEWCRLock}$ = (AVAL[7:0] + 1) x t_{I3CCLK}
   - ASNCR[1:0] = 01：$t_{NEWCRLock}$ = (AVAL[7:0] + 1) x 100 x t_{I3CCLK}
   - ASNCR[1:0] = 10：$t_{NEWCRLock}$ = (AVAL[7:0] + 1) x 2000 x t_{I3CCLK}
   - ASNCR[1:0] = 11：$t_{NEWCRLock}$ = (AVAL[7:0] + 1) x 50000 x t_{I3CCLK}

35.16.22 I3C 时序寄存器 2 (I3C_TIMINGR2)

I3C timing register 2

偏移地址：0x0A8

复位值：0x0000 0000

当 I3C 用作控制器时，该寄存器用于配置和使能 SCL 时钟停顿，或者根据寻址的 I3C 或传统 I^2C 目标的需要使能 SCL 时钟低电平停顿。

只有在 I3C 用作控制器时，才能对该寄存器执行写操作。

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

位 31:16 保留，必须保持复位值。

位 15:8 **STALL[7:0]**：控制器时钟停顿时间，以内核时钟周期数为单位 (Controller clock stall time, in number of kernel clock cycles)

t_{SCLL_STALL} = STALL x t_{I3CCLK}

位 7:4 保留，必须保持复位值。

位 3 **STALLA**：ACK 阶段的控制器时钟停顿使能 (Controller clock stall enable on ACK phase)

SCL 在地址 ACK/NACK 阶段（第九位之前）停顿（在由 STALL 定义的 t_{SCLL_STALL} 期间）。这样可以为目标留出时间准备数据。由控制器本身根据目标发起的请求 (IBI/HJ/CR) 所驱动的 ACK 不受该控制位的影响。

0：不停顿

1：使能停顿

位 2 **STALLC**：CCC PAR 阶段的控制器时钟停顿使能 (Controller clock stall enable on PAR phase of CCC)

SCL 在通用命令代码的 T 位阶段的 STALL x t_{SCLL_PP} 期间（第九位之前）停顿。这样可以为目标留出时间解码命令。

0：不停顿

1：使能停顿

注意：t_{SCLL_PP} = (I3C_TIMINGR0.SCLL_PP[7:0] + 1) x t_{I3CCLK}

位 1 **STALLD**：数据 PAR 阶段的控制器时钟停顿使能 (Controller clock stall enable on PAR phase of Data)

SCL 在 T 位阶段的 STALL x t_{SCLL_PP} 期间（第九位之前）停顿。这样可以为目标留出时间读取接收到的数据。

0：不停顿

1：使能停顿

注意：t_{SCLL_PP} = (I3C_TIMINGR0.SCLL_PP[7:0] + 1) x t_{I3CCLK}

位 0 **STALLT**：数据 T 位阶段（和传统 I^2C 读消息的数据字节的 ACK/NACK 阶段）的控制器时钟停顿使能 (Controller clock stall enable on T-bit phase of data (and on the ACK/NACK phase of data byte of a legacy I^2C read))

SCL 在 T 位阶段的 STALL x t_{SCLL_PP} 期间（第九位之前）停顿。这样可以为目标留出时间准备要发送的数据。

0：不停顿

1：使能停顿

注意：t_{SCLL_PP} = (I3C_TIMINGR0.SCLL_PP[7:0] + 1) x t_{I3CCLK}

35.16.23 I3C 总线特性寄存器 (I3C_BCR)

I3C bus characteristics register

偏移地址：0x0C0

复位值：0x0000 0000

当 I3C 用作目标时，该寄存器用于配置硬件使用的三个位，以在接收到 GETBCR 或 ENTDAA CCC 时返回数据字节 BCR[7:0]。I3C 总线上返回的字节 BCR[7:0] 具体如下：

- BCR[7] = 0（保留）
- BCR[6] = I3C_BCR6（支持控制器角色）
- BCR[5] = 1（高级功能，使用 GETCAPS CCC 来确定返回的字节）
- BCR[4] = 0（非虚拟目标）
- BCR[3] = 1（支持离线）
- BCR[2] = I3C_BCR2（IBI 有效负载）
- BCR[1] = 1（支持 IBI 请求）
- BCR[0] = I3C_BCR0（最大数据速度限制）

只有在 I3C_CFGR 寄存器中的 EN = 0 时，才能通过软件写入该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BCR6	Res.	Res.	Res.	BCR2	Res.	BCR0
									rw				rw		rw

位 31:7 保留，必须保持复位值。

位 6 **BCR6**：支持控制器角色 (Controller capable)
0：I3C 目标（不支持控制器角色）
1：支持 I3C 控制器角色

位 5:3 保留，必须保持复位值。

位 2 **BCR2**：带内中断 (IBI) 有效负载 (in-band interrupt (IBI) payload)
0：接受的 IBI 后没有数据字节
1：接受的 IBI 后至少有 1 个强制数据字节（最多 4 个数据字节）

位 1 保留，必须保持复位值。

位 0 **BCR0**：最大数据速度限制 (max data speed limitation)
0：无限制
1：限制（如 I3C_GETMXDSR 所述）。

35.16.24 I3C 设备特性寄存器 (I3C_DCR)

I3C device characteristics register

偏移地址：0x0C4

复位值：0x0000 0000

当 I3C 用作目标时，该寄存器用于配置设备特性 ID，可供硬件在接收到 GETDCR、ENTDAA 或 DEFTGTS CCC 时返回数据字节 DCR[7:0]。

只有在 I3C_CFGR 寄存器中的 EN = 0 时，才能通过软件写入该寄存器。

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **DCR[7:0]**：设备特性 ID (device characteristics ID)

0x00：通用设备（适用于 v1.0 设备）

其他值：描述 I3C 传感器/设备类型的 ID

注意：最新 MIPI DCR ID 分配可访问 https://www.mipi.org 获取。

35.16.25 I3C 获取功能寄存器 (I3C_GETCAPR)

I3C get capability register

偏移地址：0x0C8

复位值：0x0000 0000

当 I3C 用作目标时，该寄存器用于设置 IBI MDB 以支持挂起读通知，可供硬件在接收到 GETCAPS CCC（格式 1）时返回 GETCAP3 字节。

I3C 总线上返回的字节 GETCAP1[7:0] 具体如下：

- GETCAP1[7:0] = 0（无 HDR）

I3C 总线上返回的字节 GETCAP2[7:0] 具体如下：

- GETCAP2[7:6] = 00（无 HDR）
- GETCAP2[5:4] = 00（无组寻址）
- GETCAP2[3:0] = 0001（符合 MIPI 规范 v1.1）

I3C 总线上返回的字节 GETCAP3[7:0] 具体如下：

- GETCAP3[7] = 0（保留）
- GETCAP3[6] = I3C_GETCAPR 寄存器中的 CAPPEND（IBI MDB 支持挂起读通知）
- GETCAP3[5] = 0（无 HDR）
- GETCAP3[4] = 1（GETSTATUS 中支持释义字节）
- GETCAP3[3] = 1（GETCAPS 中支持释义字节）
- GETCAP3[2] = 0（无器件到器件传输）
- GETCAP3[1] = 0（无器件到器件传输）
- GETCAP3[0] = 0（无多通道数据传输）

只有在 I3C_CFGR 寄存器中的 EN = 0 时，才能通过软件写入该寄存器。

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

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

位 31:15 保留，必须保持复位值。

位 14 **CAPPEND**：IBI MDB 支持挂起读通知 (IBI MDB support for pending read notification)

该位由软件在总线初始化 (EN = 0) 期间写入，用于通过 IBI MDB[7:0] 值指示是否支持挂起读通知。

该位用于返回 GETCAP3 字节以响应 GETCAPS CCC（格式 1）。

0：用作目标时，该 I3C 将发送不带有指示挂起读通知的强制数据字节值的 IBI 请求

1：用作目标时，该 I3C 将发送带有指示挂起读通知的强制数据字节值 (MDB[7:5] = 101) 的 IBI 请求

位 13:0 保留，必须保持复位值。

35.16.26 I3C 控制器角色功能寄存器 (I3C_CRCAPR)

I3C controller-role capability register

偏移地址：0x0CC

复位值：0x0000 0000

当 I3C 用作目标时，该寄存器用于设置在控制器角色切换后 I3C 作为辅助控制器所支持的功能，可供硬件在接收到 GETCAPS CCC（格式 2，具有释义字节 CRCAPS (0x91)）时返回 CRCAP1 字节和 CRCAP2 字节。

I3C 总线上返回的 CRCAP1[7:0] 具体如下：

- CRCAP1[7:3] = 00000（保留）
- CRCAP1[2] = 0（无多线）
- CRCAP1[1] = I3C_CRCAPR 寄存器中的 CAPGRP（组管理）
- CRCAP1[0] = 1（热加入）

I3C 总线上返回的 CRLCAP2[7:0] 具体如下：

- CRCAP2[7:4] = 0000（保留）
- CRCAP2[3] = I3C_CRCAPR 寄存器中的 CAPDHOFF（延迟控制器角色切换）
- CRCAP2[2] = 1（支持深度睡眠）
- CRCAP2[1] = 0（无自动控制器角色回传）
- CRCAP2[0] = 1（支持 IBI ACK）

只有在 I3C_CFGR 寄存器中的 EN = 0 时，才能通过软件写入该寄存器。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CAP GRP	Res.	Res.	Res.	Res.	Res.	CAP DHOFF	Res.	Res.	Res.
						rw						rw			

位 31:10 保留，必须保持复位值。

位 9 **CAPGRP**：组管理支持（用作控制器时）(group management support (when acting as controller))

该位由软件在总线初始化 (EN = 0) 期间写入，用于指示 I3C 在用作控制器时（控制器角色切换之后）是否能够通过发出的 DEFGRPA、RSTGRPA 和 SETGRPA CCC 支持组管理。

该位用于返回 CRCAP1 字节以响应 GETCAPS CCC（格式 2）。

0：该 I3C 不支持组地址功能

1：该 I3C 支持组地址功能（成为控制器后）

位 8:4 保留，必须保持复位值。

位 3 **CAPDHOFF**：延迟控制器角色切换 (delayed controller-role hand-off)

该位由软件在总线初始化 (EN = 0) 期间写入，用于指示该目标 I3C 是否需要额外的时间来处理当前控制器所请求的控制器角色切换。

该位用于返回 CRCAP2 字节以响应 GETCAPS CCC（格式 2）。

0：该 I3C 不需要额外的时间来处理控制器角色切换

1：该 I3C 需要额外的时间来处理控制器角色切换

位 2:0 保留，必须保持复位值。

35.16.27 I3C 获取最大数据速度寄存器 (I3C_GETMXDSR)

I3C get max data speed register

偏移地址：0x0D0

复位值：0x0000 0000

当 I3C 用作目标时，该寄存器用于设置其功能和限制（如果有）。在接收到格式 1（2 字节，无 maxRdTurn）、格式 2（5 字节，有 MaxRdTurn）或格式 3（释义字节 WRRDTURN = 0x00：5 字节（同格式 2）/释义字节 CRHDLY = 0x91：单字节 CRHDLY1）的 GETMXDS CCC 时，硬件使用该寄存器返回数据字节。

I3C 总线上返回的字节 MaxWr[7:0] 具体如下：

- MaxWr[7:4] = 0000（保留）
- MaxWr[3] = 1（支持释义字节 WRRDTURN 和 CRHDLY）
- MaxWr[2:0] = 000（控制器向目标发送的非 CCC 消息的最大持续数据速率设计运行在 12.5 MHz）

I3C 总线上返回的字节 MaxRd[7:0] 具体如下：

- MaxRd[7] = 0（保留）
- MaxRd[6] = 1（允许在写传输和读传输之间停止）
- 如果 I3C_GETMXDSR 寄存器中的 TSCO = 0（时钟到数据切换时间 t_{SCO} ≤ 12 ns），则 MaxRd[5:3] = 100，否则为 111（t_{SCO} > 12 ns，更多详细信息，请参见数据手册）
- MaxRd[2:0] = 000（目标向控制器发送的非 CCC 消息的最大持续数据速率设计运行在 12.5 MHz）

如果 I3C_GETMXDSR 寄存器中的 FMT[1:0] = 00，则 I3C 总线上返回的 3 字节 MaxRdTurn[23:0] 具体如下：

- MaxRdTurn[23:0] 包含 I3C_GETMXDSR 寄存器中 RDTURN[7:0] 的 MSB（65 ms 到 16 s 之间）、中间字节（256 µs 到 65 ms 之间）或 LSB（小于 256 µs）（其他位为 0），具体取决于同一寄存器中的 FMT[1:0]。

I3C 总线上返回的字节 CRHDLY1[7:0] 具体如下：

- CRHDLY1[7:3] = 00000（保留）
- 如果 I3C_GETMXDSR 寄存器中的 HOFFAS[1:0] = 00，则 CRHDLY1[2] = 0，否则为 1（设置总线工作状态）
- CRHDLY1[1:0] = HOFFAS[1:0]（控制器角色切换工作状态）

只有在 I3C_CFGR 寄存器中的 EN = 0 时，才能通过软件写入该寄存器。

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

位 31:25 保留，必须保持复位值。

位 24 **TSCO**：时钟到数据转向时间 (t_{SCO}) (clock-to-data turnaround time)

该位由软件在总线初始化（I3C_CFGR 寄存器中 EN = 0）期间写入，用于指定时钟到数据的转向时间 t_{SCO}（相对于 12 ns 值）。硬件使用该位通过 MaxRd[5:3] 位返回编码的时钟到数据转向时间来响应 GETMXDS CCC。

0：$t_{SCO} \leq 12$ ns

1：$t_{SCO} > 12$ ns（更多详细信息，请参见数据手册）

位 23:16 **RDTURN[7:0]**：3 字节 MaxRdTurn 的编程字节（最大读取转向时间字节）(programmed byte of the 3-byte MaxRdTurn (maximum read turnaround byte))

该位由软件在总线初始化 (EN = 0) 期间写入，用于写入 3 字节 MaxRdTurn 的所选字节的值（通过 FMT[1:0] 位域），返回该值可对最大读取切换转向时间进行编码以响应 GETMXDS CCC（格式 2）。

位 15:10 保留，必须保持复位值。

位 9:8 **FMT[1:0]**：GETMXDS CCC 格式 (GETMXDS CCC format)

该位域由软件在总线初始化 (EN = 0) 期间写入，用于指示如何返回格式 1（无 MaxRdTurn）和格式 2（有 MaxRdTurn）的 GETMXDS。

该位域用于返回 2 字节格式 1（MaxWr、MaxRd）或 5 字节格式 2（MaxWr、MaxRd、3 字节 MaxRdTurn）字节以响应 GETCAPS CCC。

00：格式 1（2 字节：MaxWr（无释义字节）、MaxRd）

01：格式 2：（5 字节：MaxWr（无释义字节）、MaxRd、MaxRdTurn）

- 返回 3 字节 MaxRdTurn，其中 MSB = 0，中间字节 = 0，LSB = RDTURN[7:0]。
- 最大读取转向时间小于 256 µs。

10：格式 2（5 字节：MaxWr（无释义字节）、MaxRd 和 MaxRdTurn 的中间字节）

- 返回 3 字节 MaxRdTurn，其中 MSB = 0，中间字节 = RDTURN[7:0]，LSB = 0。
- 最大读取转向时间介于 256 到 65535 µs 之间

11：格式 2（5 字节：MaxWr（无释义字节）、MaxRd、MaxRdTurn 的 MSB）

- 返回 3 字节 MaxRdTurn，其中 MSB = RDTURN[7:0]，中间字节 = 0，LSB = 0。
- 最大读取转向时间介于 65535 µs 到 16 s 之间。

位 7:2 保留，必须保持复位值。

位 1:0 **HOFFAS[1:0]**：控制器切换工作状态 (Controller hand-off activity state)

该位域由软件在总线初始化 (EN = 0) 期间写入，用于指示当返回释义字节 CRHDLY (0x91) 以响应 GETMXDS CCC 时，在控制器角色切换到该支持控制器角色的 I3C 后，（其他）当前控制器期望 I3C 总线处于哪种初始工作状态。

该 2 位位域用于返回 CRHDLY1 字节以响应 GETCAPS CCC（格式 3），用以声明该 I3C 在控制器角色切换后成为控制器时的工作状态，以及前一个控制器在测试该 I3C 以确认其所有权之前必须等待的时间。

00：工作状态 0 是该 I3C 在成为控制器之前和成为控制器时的初始工作状态

01：工作状态 1 是该 I3C 成为控制器时的初始工作状态

10：工作状态 2 是该 I3C 成为控制器时的初始工作状态

11：工作状态 3 是该 I3C 成为控制器时的初始工作状态

35.16.28 I3C 扩展配置 ID 寄存器 (I3C_EPIDR)

I3C extended provisioned ID register

偏移地址：0xD4

复位值：0x0208 0000

当 I3C 用作目标时，该寄存器用于通过软件设置 4 位 MIPI 实例 ID，以及其他一些用于 48 位配置 ID 的固定位。在接收到 GETPID 和 ENTDAA CCC 时，硬件也会使用该寄存器返回 6 个字节的 48 位配置 ID。

只有在 I3C_CFGR 寄存器中的 EN = 0 时，才能通过软件写入该寄存器。

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

位 31:17 **MIPIMID[14:0]**：15 位 MIPI 制造商 ID (15-bit MIPI manufacturer ID)
该只读位域为 15 位意法半导体 MIPI ID (0x0104)。
该位域代表 48 位配置 ID 的位[47:33]。

位 16 **IDTSEL**：配置 ID 类型选择器 (provisioned ID type selector)
该位域设置为 0（供应商固定值）。
该位域代表 48 位配置 ID 的位[32]。
注意：配置 ID 的位[31:16] 可以为 0。

位 15:12 **MIPIID[3:0]**：4 位 MIPI 实例 ID (4-bit MIPI Instance ID)
该位域由软件写入，用于在单个 I3C 总线上通过特定编号单独设置和标识该 I3C IP 的每个实例。
该位域代表 48 位配置 ID 的位[15:12]。
注意：配置 ID 的位[11:0] 可以为 0。

位 11:0 保留，必须保持复位值。

35.16.29 I3C 寄存器映射

表 282. I3C 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	I3C_CR	MEND	MTYPE[3:0]				Res.	Res.	Res.	ADD[6:0]							RNW	DCNT[15:0]															
										CCC[7:0]																							
	Reset value	0	0	0	0	0				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 282. I3C 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x004	I3C_CFGR	Res.	TSFSET	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CFLUSH	CDMAEN	TMODE	SMODE	SFLUSH	SDMAEN	Res.	TXTHRES	TXFLUSH	TXDMAEN	Res.	RXTHRES	RXFLUSH	RXDMAEN	HJACK	Res.	HKSDAEN	EXITPTRN	RSTPTRN	NOARBH	CRINIT	EN
	Reset value		0									0	0	0	0	0	0		0	0	0		0	0	0	0		0	0	0	0	0	0
0x010	I3C_RDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDB0[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x014	I3C_RDWR	RDB3[7:0]								RDB2[7:0]								RDB1[7:0]								RDB0[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x018	I3C_TDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDB0[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x01C	I3C_TDWR	TDB3[7:0]								TDB2[7:0]								TDB1[7:0]								TDB0[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x020	I3C_IBIDR	IBIDB3[7:0]								IBIDB2[7:0]								IBIDB1[7:0]								IBIDB0[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x024	I3C_TGTTDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRELOAD	TGTTDCNT[15:0]															
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x030	I3C_SR	MID[7:0]								Res.	Res.	Res.	Res.	Res.	DIR	ABT	Res.	XDCNT[15:0]															
	Reset value	0	0	0	0	0	0	0	0						0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x034	I3C_SER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DERR	DNACK	ANACK	COVR	DOVR	STALL	PERR	CODERR[3:0]			
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0x040	I3C_RMR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RADD[6:0]							res. @ 7	RCODE[7:0]								Res.	Res.	Res.	Res.	Res.	IBIRDCNT[2:0]		
	Reset value									0	0	0	0	0	0	0		0	0	0	0	0	0	0	0						0	0	0
0x050	I3C_EVR	GRPF	DEFF	INTUPDF	ASUPDF	RSTF	MRLUPDF	MWLUPDF	DAUPDF	STAF	GETF	WKPF	Res.	HJF	CRUPDF	CRF	IBIENDF	IBIF	Res.	Res.	Res.	ERRF	RXTGTENDF	FCF	Res.	RXLASTF	TXLASTF	RXFNEF	TXFNFF	SFNEF	CFNFF	TXFEF	CFEF
	Reset value	0	0	0	0	0	0	0	0	0	0			0	0	0	0	0				0	0	0		0	0	0	0	0	0	1	1
0x054	I3C_IER	GRPIE	DEFIE	INTUPDIE	ASUPDIE	RSTIE	MRLUPDIE	MWLUPDIE	DAUPDIE	STAIE	GETIE	WKPIE	Res.	HJIE	CRUPDIE	CRIE	IBIENDIE	IBIIE	Res.	Res.	Res.	ERRIE	RXTGTENDIE	FCIE	Res.	Res.	Res.	RXFNEIE	TXFNFIE	SFNEIE	CFNFIE	Res.	Res.
	Reset value	0	0	0	0	0	0	0	0	0	0			0	0	0	0	0				0	0	0				0	0	0	0		

表 282. I3C 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x058	I3C_CEVR	CGRPF	CDEFF	CINTUPDF	CASUPDF	CRSTF	CMRLUPDF	CMWLUPDF	CDAUPDF	CSTAF	CGETF	CWKPF	Res.	CHJF	CRRUPDF	CCRF	CIBIENDF	CIBIF	Res.	Res.	Res.	CERRF	CRXTGTENDF	CFCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0	0	0	0	0			0	0	0	0	0				0	0	0									
0x05C	Reserved																																
0x060	I3C_DEVR0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSTVAL	RSTACT [1:0]		AS [1:0]		HJEN	Res.	CREN	IBIEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DA[6:0]							DAVAL
	Reset value								0	0	0	0	0	0		0	0									0	0	0	0	0	0	0	0
0x060 + 4 * x, (x = 1 to 4)	I3C_DEVRx	DIS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUSP	IBIDEN	CRACK	IBIACK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DA[6:0]							Res.
	Reset value	0												0	0	0	0									0	0	0	0	0	0	0	
0x090	I3C_MAXRLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IBIP [2:0]			MRL15:0]															
	Reset value														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x094	I3C_MAXWLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MWL15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0A0	I3C_TIMINGR0	SCLH_I2C[7:0]								SCLL_OD[7:0]								SCLH_I3C[7:0]								SCLL_PP[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0A4	I3C_TIMINGR1	Res.	Res.	Res.	SDA_HD	Res.	Res.	Res.	Res.	Res.	FREE[6:0]							Res.	Res.	Res.	Res.	Res.	Res.	ASNCR[1:0]		AVAL[7:0]							
	Reset value				0						0	0	0	0	0	0	0							0	0	0	0	0	0	0	0	0	0
0x0A8	I3C_TIMINGR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AVAL[7:0]								Res.	Res.	Res.	Res.	STALLA	STALLC	STALLD	STALLT
	Reset value																	0	0	0	0	0	0	0	0					0	0	0	0
0x0C0	I3C_BCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BCR6	Res.	Res.	Res.	BCR2	Res.	BCR0
	Reset value																										0				0		0
0x0C4	I3C_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DCR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x0C8	I3C_GETCAPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CAPPEND	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																		0														
0x0CC	I3C_CRCAPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CAPGR	Res.	Res.	Res.	Res.	Res.	CAPDHOFF	Res.	Res.	Res.
	Reset value																							0						0			

表 282. I3C 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0D0	I3C_GETMXDSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSCO	RDTURN[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	FMT[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	HOFFAS [1:0]	
	Reset value								0	0	0	0	0	0	0	0	0							0	0							0	0
0x0D4	I3C_EPIDR	MIPIMID[14:0]															IDTSEL	MIPIID[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0												

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

36 通用同步/异步收发器 (USART/UART)

本节介绍通用同步/异步收发器 (USART/UART)。

36.1 简介

USART 能够灵活地与外部设备进行全双工数据交换，满足外部设备对工业标准 NRZ 异步串行数据格式的要求。USART 通过小数波特率发生器实现了多种波特率。

USART 不仅支持同步单向通信和半双工单线通信，以及 LIN（局域互连网络）、智能卡协议、IrDA（红外线数据协会）SIR ENDEC 规范和调制解调器操作 (CTS/RTS)，还支持多处理器通信。

通过配置多个缓冲区使用 DMA（直接存储器访问）可实现高速数据通信。

36.2 USART 主要特性

- 全双工异步通信
- NRZ 标准格式（标记/空格）
- 可配置为 16 倍过采样或 8 倍过采样，从而在速度容差与时钟容差之间取得最佳平衡
- 波特率发生器系统
- 两个用于收发数据的内部 FIFO
  每个 FIFO 均可由软件使能/禁止，并且均带有一个状态标志。
- 通用可编程收发波特率
- 双时钟域，带有独立于 PCLK 的外设专用内核时钟
- 自动波特率检测
- 数据字长度可编程（7 位、8 位或 9 位）
- 可编程的数据顺序，最先移位 MSB 或 LSB
- 停止位可配置（支持 1 个或 2 个停止位）
- 用于同步通信的同步主/从模式和时钟输出/输入
- SPI 从发送下溢错误标志
- 单线半双工通信
- 使用 DMA 实现连续通信
- 使用中央 DMA 在预留的 SRAM 缓冲区中收/发字节。
- 发送器和接收器有单独的使能位
- 发送和接收的单独信号极性控制
- Tx/Rx 引脚可配置互换
- 调制解调器和 RS-485 收发器的硬件流控制
- 通信控制/错误检测标志
- 奇偶校验控制：
  – 发送奇偶校验位
  – 检查接收的数据字节的奇偶性

- 具有标志的中断源
- 多处理器通信：从静默模式唤醒（通过空闲线检测或地址标记检测）
- 从停止模式唤醒

36.3 USART 扩展特性

- LIN 主模式同步刹车发送功能和 LIN 从模式刹车检测功能
  - 对 USART 进行 LIN 硬件配置时可生成 13 位刹车符号和检测 10/11 位刹车符号
- IrDA SIR 编解码器在正常模式下支持 3/16 位持续时间
- 智能卡模式
  - 对于 ISO/IEC 7816-3 标准中定义的智能卡，支持 T=0 和 T=1 异步协议
  - 智能卡工作模式下，支持 0.5 和 1.5 个停止位
- 支持 ModBus 通信
  - 超时功能
  - CR/LF 字符识别

36.4 USART 实现

下表介绍了 USART 实现。其中还包含 LPUART 作为参照。

表 283. STM32H503xx 特性

USART 模式/特性	STM32H503xx
USART1	完全
USART2	完全
USART3	完全
LPUART1	低功耗

表 284. USART/LPUART 特性

USART 模式/特性(1)	全面的功能集	基本的功能集	低功耗功能集
调制解调器的硬件流控制	X	X	X
使用 DMA 进行连续通信	X	X	X
多处理器通信	X	X	X
同步模式（主/从）	X	-	-
智能卡模式	X	-	-
单线半双工通信	X	X	X
IrDA SIR ENDEC 模块	X	X	-
LIN 模式	X	X	-
双时钟域	X	X	X
接收器超时中断	X	X	-
Modbus 通信	X	X	-

表 284. USART/LPUART 特性（续）

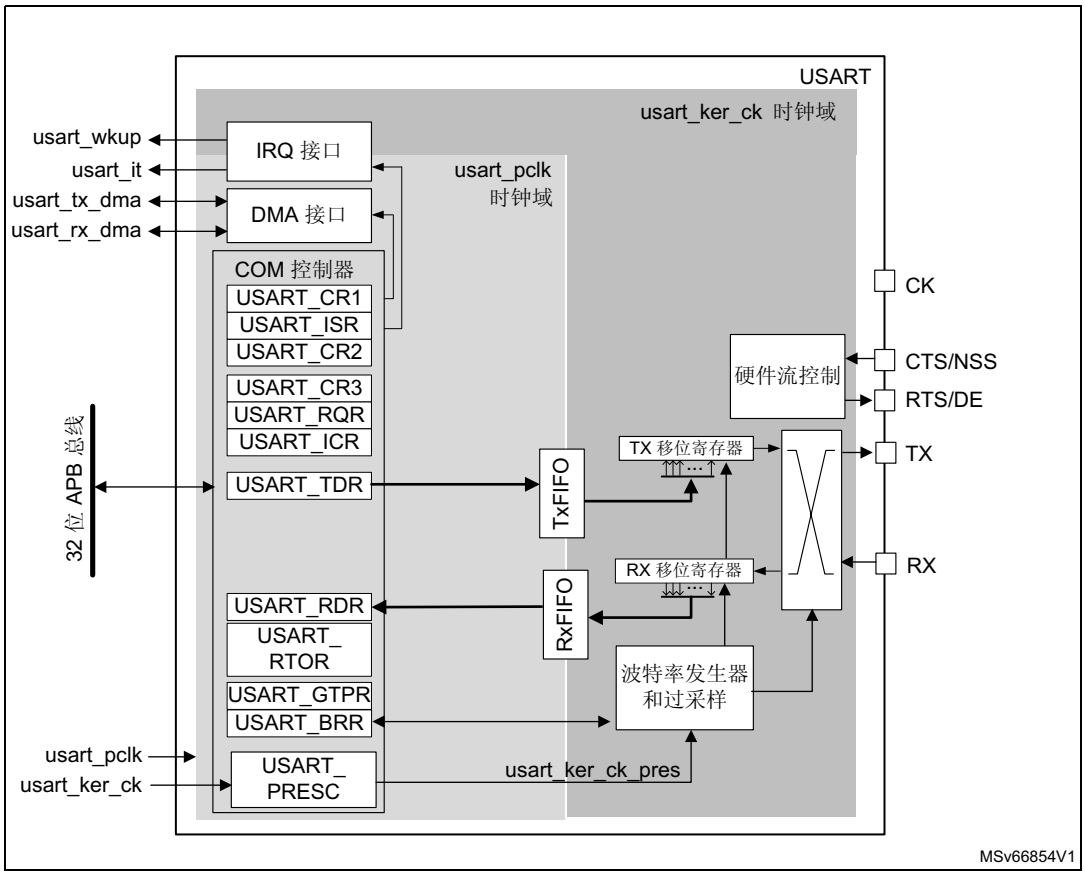
USART 模式/特性(1)	全面的功能集	基本的功能集	低功耗功能集
自动波特率检测	X	X	-
驱动器使能	X	X	X
USART 数据长度	7 位、8 位和 9 位		
Tx/Rx FIFO	X	X	X
Tx/Rx FIFO 大小	8		
从低功耗模式唤醒	X(2)	X(2)	X(2)

1. X = 支持。
2. 支持从停止模式唤醒。

36.5 USART 功能描述

36.5.1 USART 框图

图 425. USART 框图



36.5.2 USART 引脚和内部信号

USART 输入/输出引脚说明

- USART 双向通信

  USART 双向通信需要至少两个引脚：接收数据输入引脚 (RX) 和发送数据输出引脚 (TX)：

  – **RX**（接收数据输入引脚）

  RX 为串行数据输入引脚。采用过采样技术进行数据恢复，即区分有效输入数据和噪声。

  – **TX**（发送数据输出引脚）

  如果关闭发送器，该输出引脚模式由其 I/O 端口配置决定。如果使能了发送器但没有需要发送的数据，则 TX 引脚处于高电平。在单线和智能卡模式下，该 I/O 用于发送和接收数据。

- RS232 硬件流控制模式

  在 RS232 硬件流控制模式下需要以下引脚：

  – **CTS**（清除以发送）

  如果驱动为高电平，则该信号用于在当前传输结束时阻止数据发送。

  – **RTS**（请求以发送）

  如果为低电平，则该信号用于指示 USART 已准备好接收数据。

- RS485 硬件控制模式

  在 RS485 硬件控制模式下需要 **DE**（驱动器使能）引脚。该信号用于激活外部收发器的发送模式。

- 同步主/从模式和智能卡模式

  在同步主/从模式和智能卡模式下需要以下引脚：

  – CK

  该引脚在同步主模式和智能卡模式下用作时钟输出。

  它在同步从模式下用作时钟输入。

  在同步主模式下，该引脚用于输出发送器数据时钟，以便按照 SPI 主设备模式进行同步发送（起始位和结束位上无时钟脉冲，可通过软件向最后一个数据位发送时钟脉冲）。同时，RX 引脚可同步接收数据。该机制可用于控制带移位寄存器的外设（例如 LCD 驱动器）。时钟相位和极性可通过软件编程。

  在智能卡模式下，CK 输出向智能卡提供时钟。

  – NSS

  该引脚在同步从模式下用作从设备选择输入。

有关 USART 输入/输出引脚和内部信号的列表，请参见*表 285* 和*表 286*。

表 285. USART 输入/输出引脚

引脚名称	信号类型	说明
USART_RX	输入	串行数据接收输入。
USART_TX	输出	发送数据输出引脚。
USART_CTS	输入	清除以发送
USART_RTS	输出	请求以发送
USART_DE(1)	输出	驱动器使能

表 285. USART 输入/输出引脚（续）

引脚名称	信号类型	说明
USART_CK	输出	同步主模式和智能卡模式下的时钟输出。
USART_NSS(2)	输入	同步从模式下的从设备选择输入。

1. USART_DE 和 USART_RTS 共用同一个引脚。
2. USART_NSS 和 USART_CTS 共用同一个引脚。

USART 输入/输出信号说明

表 286. USART 内部输入/输出信号

引脚名称	信号类型	说明
usart_pclk	输入	APB 时钟
usart_ker_ck	输入	USART 内核时钟
usart_wkup	输出	USART 提供唤醒中断
usart_it	输出	USART 全局中断
usart_tx_dma	输入/输出	USART 发送 DMA 请求
usart_rx_dma	输入/输出	USART 接收 DMA 请求

36.5.3 USART 时钟

*图 425 中的简化框图*显示的是两个完全独立的时钟域：

- **usart_pclk** 时钟域

  usart_pclk 时钟信号为外设总线接口提供时钟。需要访问 USART 寄存器时，该信号必须有效。

- **usart_ker_ck** 内核时钟域。

  usart_ker_ck 是 USART 时钟源。它独立于 **usart_pclk**，由 RCC 提供。因此，即使 **usart_ker_ck** 时钟停止，也可以对 USART 寄存器执行读/写操作。

  不支持双时钟域功能时，**usart_ker_ck** 时钟与 **usart_pclk** 时钟相同。

usart_pclk 和 **usart_ker_ck** 之间无任何约束：**usart_ker_ck** 可快于/可慢于 **usart_pclk**。唯一的限制是软件以足够快的速度管理通信的能力。

USART 工作在 SPI 从设备模式下时，会使用源自外部 CK 信号（由外部主 SPI 设备提供）的串行接口时钟来处理数据流。**usart_ker_ck** 的时钟频率必须至少 3 倍于 CK 上的输入时钟。

36.5.4 USART 字符说明

可通过对 USART_CR1 寄存器中的 M 位（M0：位 12，M1：位 28）进行编程来将字长设置为 7 位、8 位或 9 位（请参见*图 426*）：

- 7 位字符长度：M[1:0] = 10
- 8 位字符长度：M[1:0] = 00
- 9 位字符长度：M[1:0] = 01

注意： *在 7 位数据长度模式下，不支持智能卡模式、LIN 主模式和自动波特率（0x7F 帧和 0x55 帧检测）。*

在默认情况下，信号（TX 或 RX）在起始位工作期间处于低电平状态。在停止位工作期间处于高电平状态。

通过极性配置控制，可以单独针对每个信号对这些值取反。

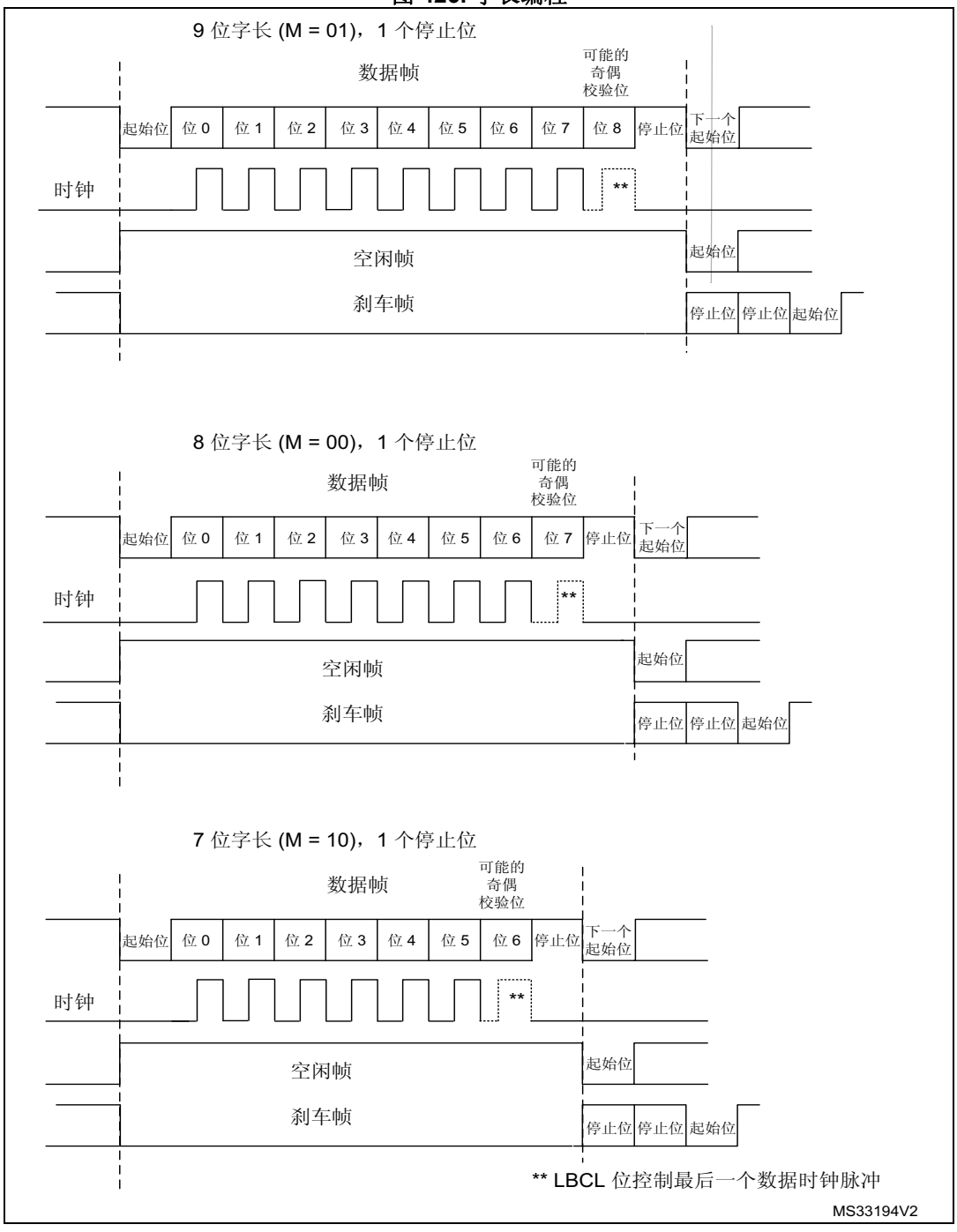
空闲字符可理解为整个帧周期内电平均为“1”（停止位的电平也是“1”）。

刹车字符可理解为在一个帧周期内接收到的电平均为“0”。发送器在刹车帧的末尾插入 2 个停止位。

发送和接收操作由通用波特率发生器驱动。当发送器和接收器的使能位置 1 时，将分别生成发送时钟和接收时钟。

下面给出了各个块的详细说明。

图 426. 字长编程


9 位字长 (M = 01)，1 个停止位
数据帧
可能的
奇偶
校验位
起始位
位 0
位 1
位 2
位 3
位 4
位 5
位 6
位 7
位 8
停止位
下一个
起始位
时钟
**
空闲帧
起始位
刹车帧
停止位
停止位
起始位
8 位字长 (M = 00)，1 个停止位
数据帧
可能的
奇偶
校验位
起始位
位 0
位 1
位 2
位 3
位 4
位 5
位 6
位 7
停止位
下一个
起始位
时钟
**
空闲帧
起始位
刹车帧
停止位
停止位
起始位
7 位字长 (M = 10)，1 个停止位
数据帧
可能的
奇偶
校验位
起始位
位 0
位 1
位 2
位 3
位 4
位 5
位 6
停止位
下一个
起始位
时钟
**
空闲帧
起始位
刹车帧
停止位
停止位
起始位
** LBCL 位控制最后一个数据时钟脉冲
MS33194V2

36.5.5 USART FIFO 和阈值

USART 可工作在 FIFO 模式下。

USART 具有一个发送 FIFO (TXFIFO) 和一个接收 FIFO (RXFIFO)。可通过将 USART_CR1 寄存器中的 FIFOEN（位 29）置 1 使能 FIFO 模式。仅 UART、SPI 和智能卡模式下支持该模式。

最大数据字长度为 9 位，因此 TXFIFO 为 9 位宽。不过，RXFIFO 的默认宽度为 12 位。这是因为接收器不仅在 FIFO 中存储数据，而且还存储与每个字符相关的错误标志（奇偶校验错误、噪声错误和帧错误标志）。

注意： *接收的数据与相应的标志一起存储在 RXFIFO 中，但读取 RDR 时仅读取数据。*

状态标志位于 USART_ISR 寄存器中。

可以配置触发 Tx 和 RX 中断的 TXFIFO 和 RXFIFO 阈值。这些阈值通过 USART_CR3 控制寄存器中的 RXFTCFG 和 TXFTCFG 位域进行编程。

在这种情况下：

- 当 RXFIFO 中接收到的数据量达到 RXFTCFG 位域中编程的阈值时，会生成 Rx 中断。此时，USART_ISR 寄存器中的 RXFT 标志置 1。这意味着已接收到 RXFTCFG 数据：USART_RDR 中有 1 个数据，RXFIFO 中有 (RXFTCFG - 1) 个数据。例如，如果将 RXFTCFG 编程为 101，则在接收到对应于 FIFO 大小的数据量（RXFIFO 中有（FIFO 大小 - 1）个数据，USART_RDR 中有 1 个数据）时，RXFT 标志将置 1。因此，下一个接收到的数据不会将上溢标志置 1。
- 当 TXFIFO 中的空位置数达到在 TXFTCFG 位域中编程的阈值时，会生成 Tx 中断。

36.5.6 USART 发送器

发送器可发送 7 位、8 位或 9 位的数据字，具体取决于 M 位的状态。要激活发送器功能，必须将发送使能位 (TE) 置 1。发送移位寄存器中的数据在 TX 引脚输出，相应的时钟脉冲在 CK 引脚输出。

字符发送

USART 发送期间，首先通过 TX 引脚移出数据的最低有效位（默认配置）。在该模式下，USART_TDR 寄存器的缓冲区 (TDR) 位于内部总线和发送移位寄存器之间。

使能 FIFO 模式时，写入到发送数据寄存器 (USART_TDR) 中的数据会在 TXFIFO 中排队。

每个字符前面都有一个起始位，其对应于一个位周期的逻辑低电平。字符由可配置数量的停止位终止。

停止位的数量可配置为 0.5、1、1.5 或 2。

注意： *向 USART_TDR 中写入要发送的数据前，TE 位必须先置 1。*

数据发送期间不得复位 TE 位。发送期间复位 TE 位会冻结波特率计数器，进而损坏 TX 引脚上的数据。当前发送的数据随即丢失。

使能 TE 位时，将发送空闲帧。

可配置的停止位

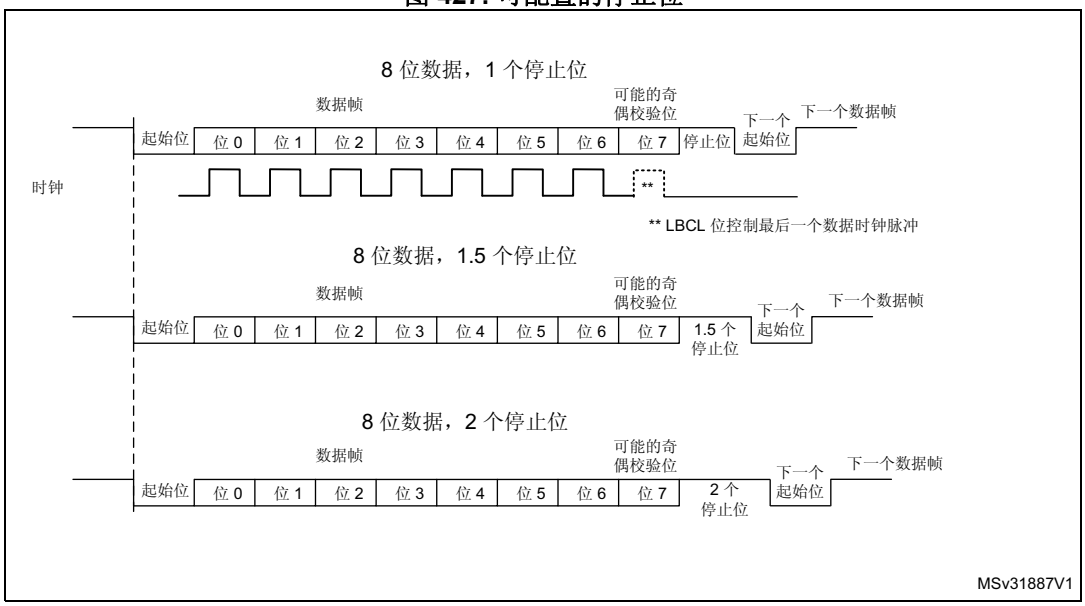
可以在 USART_CR2 的位 13 和位 12 中编程将随各个字符发送的停止位的数量。

- ***1 个停止位***：这是停止位数量的默认值。
- ***2 个停止位***：正常 USART 模式、单线模式和调制解调器模式支持该值。
- ***1.5 个停止位***：用于智能卡模式。

空闲帧发送包括停止位。

刹车传输是 10 个低电平位（M[1:0] = 00 时）、11 个低电平位（M[1:0] = 01 时）或 9 个低电平位（M[1:0] = 10 时），然后是 2 个停止位（请参见*图 427*）。无法传送长刹车（刹车长度大于 9/10/11 个低电平位）。

图 427. 可配置的停止位



字符发送步骤

要发送字符，需遵循以下步骤：

1. 对 USART_CR1 中的 M 位进行编程以定义字长。
2. 使用 USART_BRR 寄存器选择所需波特率。
3. 对 USART_CR2 中的停止位数量进行编程。
4. 通过向 USART_CR1 寄存器中的 UE 位写入 1 使能 USART。
5. 如果必须进行多缓冲区通信，请选择 USART_CR3 中的 DMA 使能 (DMAT)。按照*第 36.5.20 节：使用 USART 和 DMA 进行连续通信*中的说明配置 DMA 寄存器。
6. 将 USART_CR1 中的 TE 位置 1 以便在首次发送时发送一个空闲帧。
7. 在 USART_TDR 寄存器中写入要发送的数据。为每个要在单缓冲区模式下发送的数据重复这一步骤。
   - 禁止 FIFO 模式时，向 USART_TDR 写入数据会将 TXE 标志清零。
   - 使能 FIFO 模式时，向 USART_TDR 写入数据会为 TXFIFO 中增添一个数据。当 TXFNF 标志置 1 时，对 USART_TDR 寄存器的写操作可以执行。该标志会保持置 1，直到 TXFIFO 已满。

8. 将最后一个数据写入 USART_TDR 寄存器后，等待 TC = 1。
   – 禁止 FIFO 模式时，这表示最后一个帧的发送已完成。
   – 使能 FIFO 模式时，这表示 TXFIFO 和移位寄存器均为空。

   当 USART 被禁止或进入暂停模式时，需要执行此检查来避免损坏最后一次发送。

单字节通信

- 禁止 FIFO 模式时

  对发送数据寄存器进行写操作始终会清零 TXE 位。TXE 标志由硬件置 1。它表示：
  – 数据已从 USART_TDR 寄存器移到移位寄存器中且数据发送已开始；
  – USART_TDR 寄存器为空；
  – USART_TDR 寄存器中可写入下一个数据，而不会覆盖前一个数据。

  TXEIE 位置 1 时该标志位会生成中断。

  发送时，写 USART_TDR 寄存器，会将数据存储在 TDR 缓冲区。该数据随后会在当前发送结束时复制到移位寄存器中。

  未发送时，写 USART_TDR 寄存器，会将数据写入移位寄存器，数据发送开始时，TXE 位置 1。
- 使能 FIFO 模式时，TXFNF（TXFIFO 未满）标志由硬件置 1，以指示：
  – TXFIFO 未满；
  – USART_TDR 寄存器为空；
  – USART_TDR 寄存器中可写入下一个数据，而不会覆盖前一个数据。发送时，写 USART_TDR 寄存器，会将数据存储在 TXFIFO。该数据随后会在当前发送结束时从 TXFIFO 复制到移位寄存器中。

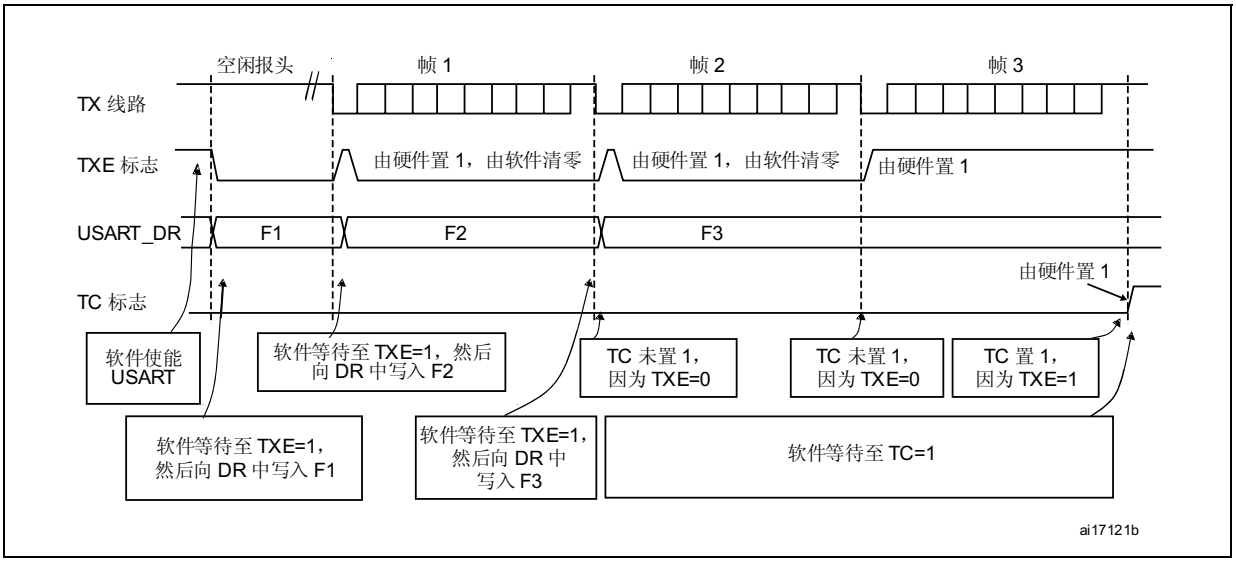
  TXFIFO 未满时，即使在对 USART_TDR 寄存器执行完写操作后，TXFNF 标志也保持为 1。该标志在 TXFIFO 已满时清零。TXFNFIE 位置 1 时该标志位会生成中断。

  或者，当达到 TXFIFO 阈值时，会生成中断并将数据写入 FIFO。在这种情况下，CPU 可写入由编程的触发阈值定义的数据块。

  如果帧已发送（停止位后）且 TXE 标志（FIFO 模式下的 TXFE）置 1，TC 标志将变为高电平。如果 USART_CR1 寄存器中的 TCIE 位置 1，将生成中断。

向 USART_TDR 寄存器中写入最后一个数据后，必须等待至 TC 置 1，之后才可禁止 USART 或使微控制器进入低功耗模式（请参见*图 428：发送时的 TC/TXE 行为*）。

图 428. 发送时的 TC/TXE 行为



注意： *使能 FIFO 管理时，TXFNF 标志将用于数据发送。*

刹车字符

将 SBKRQ 位置 1 将发送一个刹车字符。刹车帧的长度取决于 M 位（请参见*图 426*）。

如果将 1 写入 SBKRQ 位，则当前字符发送完成后，将在 TX 线路上发送一个刹车字符。通过写操作将 SBKF 位置 1 并在刹车字符发送完成时（发送刹车字符后的停止位期间），该位由硬件复位。USART 在刹车帧末尾的两位持续时间内插入一个逻辑“1”信号 (STOP)，以确保识别下个帧的起始位。

如果 SBKRQ 位置 1，则在当前发送结束时，会发送一个刹车字符。

使能 FIFO 模式时，即使 TXFIFO 已满，发送刹车字符的优先级也仍高于发送数据的优先级。

空闲字符

将 TE 位置 1 会驱动 USART 在第一个数据帧之前发送一个空闲帧。

36.5.7 USART 接收器

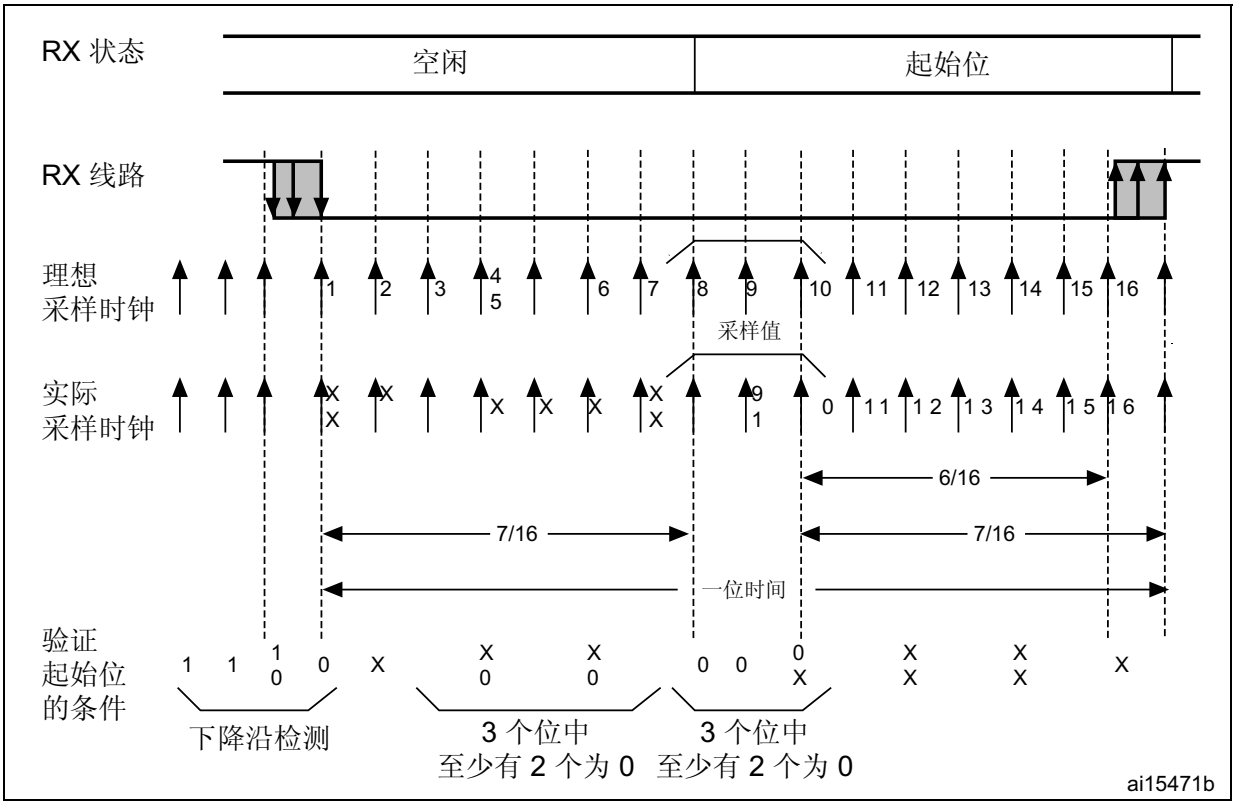
USART 可接收 7 位、8 位或 9 位的数据字，具体取决于 USART_CR1 寄存器中的 M 位。

起始位检测

16 倍或 8 倍过采样时，起始位检测序列相同。

在 USART 中，识别出特定序列的采样时会检测到起始位。该序列为：1 1 1 0 X 0 X 0X 0X 0 X 0X 0。

图 429. 16 倍或 8 倍过采样时的起始位检测



注意： *如果序列不完整，起始位检测将中止，接收器将返回空闲状态（无标志位置 1）等待下降沿。*

如果 3 个采样位均为 0（针对第 3 位、第 5 位和第 7 位进行首次采样时检测到这 3 位均为 0；针对第 8 位、第 9 位和第 10 位进行第二次采样时仍检测到这 3 位均为 0），可确认起始位（RXNE 标志置 1 且 RXNEIE = 1 时生成中断；如果使能 FIFO 模式，则 RXFNE 标志置 1 且 RXFNEIE = 1 时生成中断）。

满足以下条件时，可验证起始位但 NE 噪声标志置 1：

a) 对于两次采样，3 个采样位中有 2 位为 0（针对第 3 位、第 5 位和第 7 位进行采样；针对第 8 位、第 9 位和第 10 位采样）

或

b) 如果其中一次采样时（对第 3 位、第 5 位和第 7 位进行采样或对第 8 位、第 9 位和第 10 位进行采样），3 个采样位中有 2 个为 0。

如果上述条件均不满足，则启动检测中止，接收器返回空闲状态（无标志置 1）。

字符接收

USART 接收期间，首先通过 RX 引脚移出数据的最低有效位（默认配置）。

字符接收步骤

要接收字符，需遵循以下步骤：

1. 对 USART_CR1 中的 M 位进行编程以定义字长。
2. 使用波特率寄存器 USART_BRR 选择所需波特率
3. 对 USART_CR2 中的停止位数量进行编程。
4. 通过向 USART_CR1 寄存器中的 UE 位写入 1 使能 USART。
5. 如果将进行多缓冲区通信，请选择 USART_CR3 中的 DMA 使能 (DMAR)。按照*第 36.5.20 节：使用 USART 和 DMA 进行连续通信*中的说明配置 DMA 寄存器。
6. 将 RE 位 USART_CR1 置 1。这一操作将使能接收器开始搜索起始位。

接收到字符时：

- 如果已禁止 FIFO 模式，则 RXNE 位置 1，这表明移位寄存器的内容已传送到 RDR。也就是说，已接收到并可读取数据（以及其相应的错误标志）。
- 如果已使能 FIFO 模式，则 RXFNE 位置 1，这表示 RXFIFO 非空。读取 USART_RDR 会返回输入到 RXFIFO 中的最早数据。接收到数据时，数据以及相应的错误位将一起被存储在 RXFIFO 中。
- 如果 RXNEIE（使能 FIFO 模式时为 RXFNEIE）位置 1，则会生成中断。
- 如果接收期间已检测到帧错误、噪声错误、奇偶校验错误或上溢错误，错误标志会置 1。
- 在多缓冲区通信模式下：
  - 如果禁止 FIFO 模式，则 RXNE 标志会在每次接收到字节后置 1。该标志在 DMA 读取接收数据寄存器时被清零。
  - 如果使能 FIFO 模式，则 RXFNE 标志在 RXFIFO 非空时置 1。每次收到 DMA 请求后，都会从 RXFIFO 取回数据。当 RXFIFO 非空时（即，当存在要从 RXFIFO 中读取的数据时），会触发 DMA 请求。
- 在单缓冲区模式下：
  - 如果禁止 FIFO 模式，则通过软件对 USART_RDR 寄存器进行读操作来将 RXNE 标志清零。也可以通过将 USART_RQR 寄存器中的 RXFRQ 位编程为 1 来清零 RXNE 标志。RXNE 标志必须在结束接收下一个字符前清零，以避免发生上溢错误。
  - 如果使能 FIFO 模式，则 RXFNE 在 RXFIFO 非空时置 1。每次对 USART_RDR 执行完读操作后，都会从 RXFIFO 中取回数据。RXFIFO 为空时，RXFNE 标志将清零。也可以通过将 USART_RQR 寄存器中的 RXFRQ 位编程为 1 来清零 RXFNE 标志。RXFIFO 已满时，必须在结束接收下一个字符前读取 RXFIFO 中的第一个条目，以避免发生上溢错误。当 RXFNEIE 位置 1 时，RXFNE 标志会生成中断。或者，当达到 RXFIFO 阈值时，会生成中断并从 RXFIFO 中读取数据。在这种情况下，CPU 可读取由编程的阈值定义的数据块。

刹车字符

接收到刹车字符时，USART 将会按照帧错误对其进行处理。

空闲字符

检测到空闲帧时，除了在 IDLEIE 位置 1 时会生成中断外，处理步骤与接收到数据字符的情况相同。

上溢错误

- 禁止 FIFO 模式

  如果在 RXNE 未复位时接收到字符，则会发生上溢错误。

  RXNE 位清零前，数据无法从移位寄存器传送到 RDR 寄存器。每接收到一个字节后，RXNE 标志都将置 1。

  当 RXNE 标志位是 1 时，如果在接收到下一个数据或尚未处理上一个 DMA 请求，则会发生上溢错误。发生上溢错误时：

  – ORE 位置 1。
  – RDR 中的内容不会丢失。可通过读取 USART_RDR 寄存器获得之前的数据。
  – 移位寄存器会被覆盖。之后，上溢期间接收到的任何数据都将丢失。
  – 如果 RXNEIE 或 EIE 位置 1，则会生成中断。

- 使能 FIFO 模式

  移位寄存器准备好传送并且接收 FIFO 已满时，会发生上溢错误。

  在 RXFIFO 中出现一个空闲位置之前，数据无法从移位寄存器传送到 USART_RDR 寄存器。当 RXFIFO 非空时，RXFNE 标志置 1。

  如果 RXFIFO 已满且移位寄存器已准备好传送，则会发生上溢错误。发生上溢错误时：

  – ORE 位置 1。
  – RXFIFO 中的第一个条目不会丢失。通过读取 USART_RDR 寄存器可获得此条目。
  – 移位寄存器会被覆盖。之后，上溢期间接收到的任何数据都将丢失。
  – 如果 RXFNEIE 或 EIE 位置 1，则会生成中断。

通过将 USART_ICR 寄存器中的 ORECF 位置 1 来复位 ORE 位。

注意： *ORE 位置 1 时表示至少 1 个数据丢失。*

禁止 FIFO 模式时，有以下两种可能

- *如果 RXNE = 1，则最后一个有效数据存储于接收寄存器 (RDR) 中并且可进行读取，*
- *如果 RXNE = 0，则最后一个有效数据已被读取，因此 RDR 寄存器中没有要读取的数据。接收到新（丢失）数据的同时已读取 RDR 寄存器中的最后一个有效数据时，会发生该情况。*

选择时钟源和合适的过采样方法

通过时钟控制系统选择时钟源（请参见*章节：复位和时钟控制 (RCC)*）。在使能 USART 之前，必须通过 UE 位选择时钟源。

必须遵循以下两个条件选择时钟源：

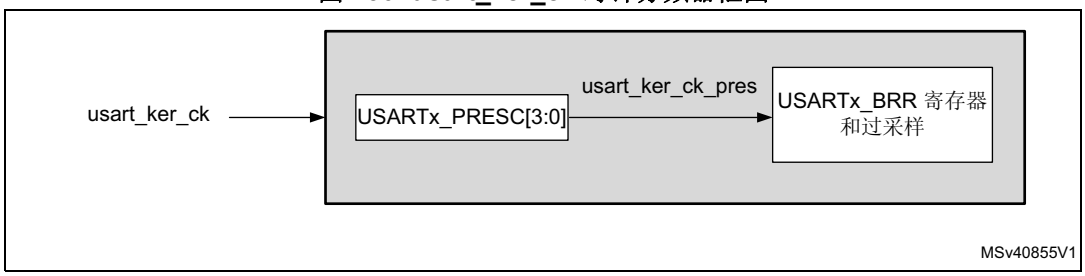
- 可在低功耗模式下使用 USART
- 通信速度。

时钟源频率为 usart_ker_ck。

如果支持双时钟域和从低功耗模式唤醒功能，则 usart_ker_ck 时钟源可在 RCC 中进行配置（请参见*章节：复位和时钟控制 (RCC)*）。否则，usart_ker_ck 时钟与 usart_pclk 相同。

usart_ker_ck 时钟可根据可编程系数（在 USART_PRESC 寄存器中定义）进行分频。

图 430. usart_ker_ck 时钟分频器框图



某些 usart_ker_ck 时钟源允许 USART 在 MCU 处于低功耗模式时接收数据。需要时，USART 可基于所接收的数据和选择的唤醒模式来唤醒 MCU，以通过用软件读取 USART_RDR 寄存器的方式或通过 DMA 的方式传输已接收数据。

对于其他时钟源，系统必须激活才能使能 USART 通信。

时钟源还决定通信速度范围（尤其是最大通信速度）。

接收器采用不同的用户可配置过采样技术（除了同步模式下），可以从噪声中提取有效数据。这可在最大通信速度与抗噪声/时钟误差性能之间实现最佳平衡。

可通过编程 USART_CR1 寄存器中的 OVER8 位来选择采样方法，且采样时钟可以是波特率时钟的 16 倍或 8 倍（请参见*图 431* 和*图 432*）。

根据应用：

- 选择 8 倍过采样（OVER8 = 1）以获得更高的速度（高达 usart_ker_ck_pres/8）。这种情况下接收器对时钟偏差的最大容差将会降低（请参见*第 1271 页的第 36.5.9 节：USART 接收器对时钟偏差的容差*）
- 选择 16 倍过采样 (OVER8=0) 以增加接收器对时钟偏差的容差。在这种情况下，最大速度被限制为最大 usart_ker_ck_pres/16（其中，usart_ker_ck_pres 为 USART 输入时钟通过预分频器进行分频得到的值）。

可通过编程 USART_CR3 寄存器中的 ONEBIT 位选择用于评估逻辑电平的方法。有两种选择可供使用：

- 在已接收位的中心进行三次采样，从而进行多数表决。这种情况下，如果用于多数表决的 3 次采样结果不相等，NE 位置 1。
- 在已接收位的中心进行单次采样

  根据应用：

  – 在噪声环境下工作时，请选择三次采样的多数表决法 (ONEBIT = 0)；在检测到噪声时请拒绝数据（请参见*表 287*），因为这表示采样过程中产生了干扰。

  – 线路无噪声时请选择单次采样法（ONEBIT=1）以增加接收器对时钟偏差的容差（请参见*第 1271 页的第 36.5.9 节：USART 接收器对时钟偏差的容差*）。这种情况下 NE 位始终不会置 1。

帧中检测到噪声时：

- 在 RXNE 位（使能 FIFO 模式时为 RXFNE 位）的上升沿时 NE 位置 1。
- 无效数据从移位寄存器传送到 USART_RDR 寄存器。
- 单字节通信时无中断产生。然而，在 RXNE 位（使能 FIFO 模式时为 RXFNE 位）生成中断时，该位出现上升沿。多缓冲区通信时，USART_CR3 寄存器中的 EIE 位置 1 时会发出中断。

通过设置 ICR 寄存器中的 NFCF 位复位 NE 位。

注意： *SPI 模式不支持噪声错误。*

智能卡、IrDA 和 LIN 模式下不可采用 8 倍过采样。在这些模式下，OVER8 位由硬件强制清零。

图 431. 16 倍过采样时的数据采样

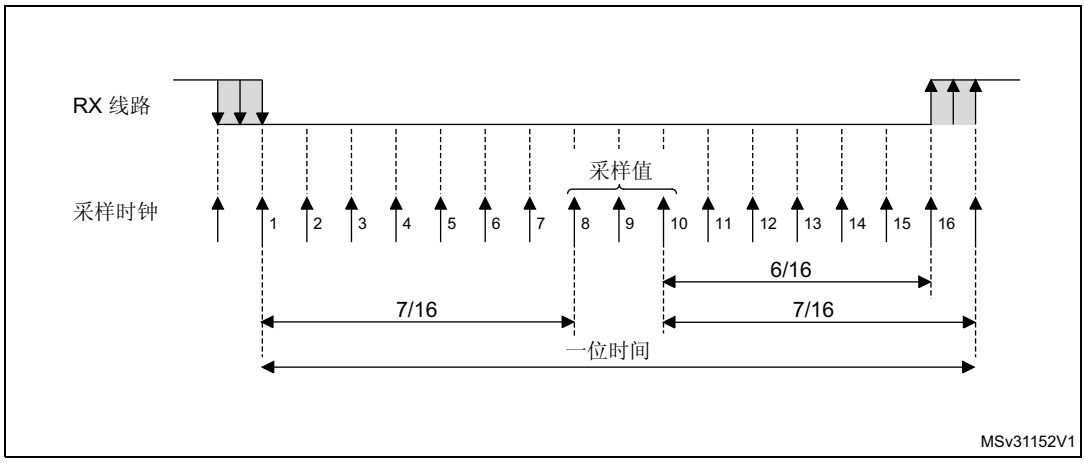


图 432. 8 倍过采样时的数据采样

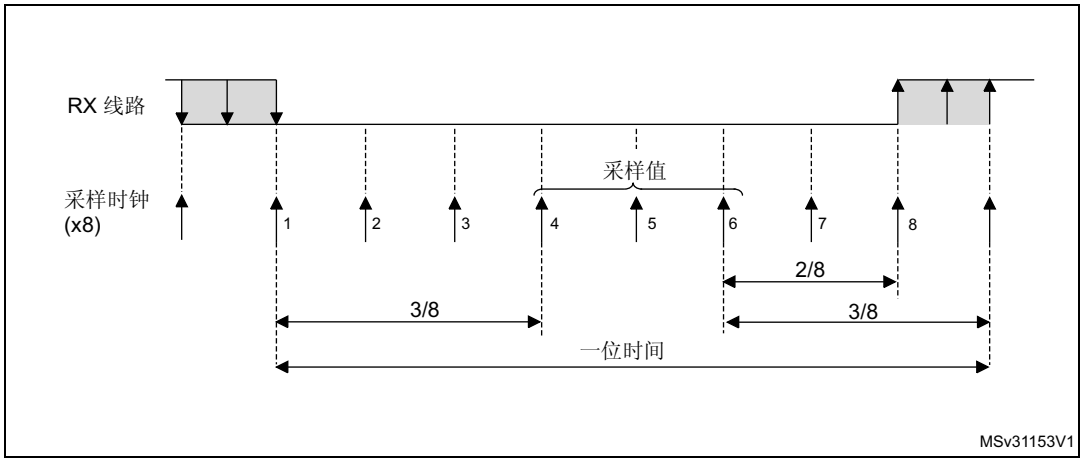


表 287. 通过采样数据进行噪声检测

采样值	NE 状态	接收的位值
000	0	0
001	1	0
010	1	0
011	1	1
100	1	0
101	1	1
110	1	1
111	0	1

帧错误

如果接收数据时未在预期时间内识别出停止位，从而出现同步失效或过度的噪声，则会检测到帧错误。

检测到帧错误时：

- FE 位由硬件置 1
- 无效数据从移位寄存器传送到 USART_RDR 寄存器（使能 FIFO 模式时为 RXFIFO）。
- 单字节通信时无中断产生。然而，在 RXNE 位（使能 FIFO 模式时为 RXFNE 位）生成中断时，该位出现上升沿。多缓冲区通信时，USART_CR3 寄存器中的 EIE 位置 1 时会发出中断。

通过将 1 写入 USART_ICR 寄存器中的 FECF 位来复位 FE 位。

注意： *SPI 模式不支持帧错误。*

接收期间可配置的停止位

可通过 USART_CR 的控制位配置要接收的停止位的数量：可以是 1 或 2 个（正常模式下），也可以是 0.5 或 1.5 个（智能卡模式下）。

- **0.5 个停止位（在智能卡模式下接收时）**：不会对 0.5 个停止位进行采样。结果，选择 0.5 个停止位时，无法检测到帧错误和刹车帧。
- **1 个停止位**：将在第 8、第 9 和第 10 次采样时对 1 个停止位进行采样。
- 1.5 个停止位（在智能卡模式下）

  在智能卡模式下发送时，设备必须检查数据是否正确发送。因此，必须使能接收器块（USART_CR1 中的 RE = 1）并检查停止位，以测试智能卡是否已检测到奇偶校验错误。

  发生奇偶校验错误时，智能卡会在采样时将数据信号强制为低电平（即 NACK 信号），该信号被标记为帧错误。之后，FE 标志在 1.5 个停止位的末尾由 RXNE 标志（使能 FIFO 模式时为 RXFNE）置 1。在第 16、第 17 和第 18 次采样时对 1.5 个停止位进行采样（停止位采样开始后维持 1 个波特时钟周期）。1.5 个停止位可分为 2 个部分：0.5 个波特时钟周期（未发生任何动作），然后是 1 个正常的停止位周期（一半时间处进行采样）（更多详细信息，请参见*第 1283 页的第 36.5.17 节：USART 接收器超时*）。
- 2 个停止位

  采样 2 个停止位时在第 8、第 9 和第 10 次采样时对第一个停止位进行采样。

  如果在第一个停止位期间检测到帧错误，则帧错误标志会置 1。

  发生帧错误时不检测第 2 个停止位。RXNE 标志（使能 FIFO 模式时为 RXFNE）将在第一个停止位结束时置 1。

36.5.8 USART 波特率生成

接收器和发送器（Rx 和 Tx）的波特率均设置为 USART_BRR 寄存器中编程的值。

公式 1：适用于标准 USART（包括 SPI 模式）的波特率（OVER8 = 0 或 1）

在 16 倍过采样的情况下，波特率通过以下公式得出：

$$\text{Tx/Rx 波特率} = \frac{\text{usart_ker_ck_pres}}{\text{USARTDIV}}$$

在 8 倍过采样的情况下，波特率通过以下公式得出：

$$\text{Tx/Rx 波特率} = \frac{2 \times \text{usart_ker_ck_pres}}{\text{USARTDIV}}$$

公式 2：智能卡、LIN 和 IrDA 模式下的波特率 (OVER8 = 0)

波特率可根据以下公式得出：

$$\text{Tx/Rx 波特率} = \frac{\text{usart_ker_ck_pres}}{\text{USARTDIV}}$$

USARTDIV 是一个存放在 USART_BRR 寄存器中的无符号定点数。

- 当 OVER8 = 0 时，BRR = USARTDIV。
- 当 OVER8 = 1 时
  - BRR[2:0] = USARTDIV[3:0] 右移 1 位。
  - BRR[3] 必须保持清零。
  - BRR[15:4] = USARTDIV[15:4]

注意： *对 USART_BRR 执行写操作后，波特率计数器更新为波特率寄存器中的新值。因此，波特率寄存器的值不得在通信时发生更改。*

16 倍和 8 倍过采样时，USARTDIV 必须大于或等于 16。

如何从 USART_BRR 寄存器中获取 USARTDIV

示例 1

要通过 usart_ker_ck_pres = 8 MHz 获得 9600 波特：

- 16 倍过采样时：
  USARTDIV = 8 000 000/9600
  BRR = USARTDIV = 0d833 = 0x0341
- 8 倍过采样时：
  USARTDIV = 2 * 8 000 000/9600
  USARTDIV = 1666,66 (0d1667 = 0x683)
  BRR[3:0] = 0x3 >> 1 = 0x1
  BRR = 0x681

示例 2

要通过 usart_ker_ck_pres = 48 MHz 获得 921.6 K 波特：

- 16 倍过采样时：
  USARTDIV = 48 000 000/921 600
  BRR = USARTDIV = 0x52 = 0x34
- 8 倍过采样时：
  USARTDIV = 2 * 48 000 000/921 600
  USARTDIV = 104 (0d104 = 0x68)
  BRR[3:0] = USARTDIV[3:0] >> 1 = 0x8 >> 1 = 0x4
  BRR = 0x64

36.5.9 USART 接收器对时钟偏差的容差

仅当总时钟系统偏差小于 USART 接收器的容差时，USART 异步接收器才能正常工作。

影响总偏差的因素包括：

- DTRA：发送器误差引起的偏差（其中还包括发送器本地振荡器的偏差）
- DQUANT：接收器的波特率量化引起的误差
- DREC：接收器本地振荡器的偏差
- DTCL：传输线路引起的偏差（通常是由于收发器所引起，它可能会在低电平到高电平转换时序与高电平到低电平转换时序之间引入不对称）

$$DTRA + DQUANT + DREC + DTCL + DWU < USART\ 接收器的容差$$

其中

DWU 是使用从低功耗模式唤醒时因采样点偏差而产生的误差。

M[1:0] = 01 时：

$$DWU = \frac{t_{WUUSART}}{11 \times Tbit}$$

M[1:0] = 00 时：

$$DWU = \frac{t_{WUUSART}}{10 \times Tbit}$$

M[1:0] = 10 时：

$$DWU = \frac{t_{WUUSART}}{9 \times Tbit}$$

$t_{WUUSART}$ 是检测到起始位下降沿与时钟（由外设请求）就绪、达到外设且调压器就绪之间的时间。

USART 接收器在*表 288* 和*表 289* 中指定的最大容许偏差下可正确接收数据，具体取决于以下设置：

- 由 USART_CR1 寄存器中的 M 位定义的 9 位、10 位或 11 位字符长度
- 由 USART_CR1 寄存器中的 OVER8 位定义的 8 倍或 16 倍过采样
- USART_BRR 寄存器的 BRR[3:0] 位等于或不等于 0000。
- 使用 1 位或 3 位对数据进行采样，取决于 USART_CR3 寄存器中 ONEBIT 位的值

表 288. BRR [3:0] = 0000 时的 USART 接收器容差

M 位	OVER8 位 = 0		OVER8 位 = 1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
00	3.75%	4.375%	2.50%	3.75%
01	3.41%	3.97%	2.27%	3.41%
10	4.16%	4.86%	2.77%	4.16%

表 289. BRR[3:0] 不等于 0000 时的 USART 接收器容差

M 位	OVER8 位 = 0		OVER8 位 = 1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
00	3.33%	3.88%	2%	3%
01	3.03%	3.53%	1.82%	2.73%
10	3.7%	4.31%	2.22%	3.33%

注意：当接收的帧恰好包含 10 个（M 位 = 00）、11 个（M 位 = 01）或 9 个（M 位 = 10）位时间的空闲帧时，表 288 和表 289 中指定的数据可能与特例中的数据略微不同。

36.5.10 USART 自动波特率检测

USART 可根据接收一个字符检测并自动设置 USART_BRR 寄存器的值。自动波特率检测在以下两种情况下非常有用：

- 事先不知道系统的通信速度。
- 系统正在使用精确度相对较低的时钟源且该机制允许在不测量时钟偏差的情况下获得正确的波特率。

时钟源频率必须与预期通信速度兼容。

- 16 倍过采样时，波特率范围为 usart_ker_ck_pres/65535 到 usart_ker_ck_pres/16。
- 8 倍过采样时，波特率范围为 usart_ker_ck_pres/65535 到 usart_ker_ck_pres/8。

在激活自动波特率检测之前，必须通过 USART_CR2 寄存器中的 ABRMOD[1:0] 位域选择自动波特率检测模式。根据不同的字符模式，存在四种检测模式。在这些自动波特率模式下，波特率在同步接收数据期间被多次测量，每次测量的结果都与前一次进行比较。

这些模式如下：

- **模式 0**：以 1 位开头的任意字符。

  这种情况下，USART 会测量起始位的持续时间（下降沿到上升沿）。
- **模式 1**：以 10xx 位模式开头的任何字符。

  这种情况下，USART 会测量起始位和第一个数据位的持续时间。测量在下降沿到下降沿期间完成，可在信号斜率较小时确保较高的精度。
- **模式 2**：0x7F 字符帧（可以是 LSB 在前模式下的 0x7F 字符，也可以是 MSB 在前模式下的 0xFE 字符）。

  这种情况下，先在起始位结束时更新波特率 (BR)，然后在位 6 结束时更新波特率（根据从下降沿到下降沿执行的测量：BR6）。以 BR 对位 0 到位 6 进行采样，而以 BR6 对字符的其他位进行采样。
- **模式 3**：0x55 字符帧。

  这种情况下，先在起始位结束时更新波特率 (BR)，然后在位 0 结束时更新波特率（根据从下降沿到下降沿执行的测量：BR0），最后在位 6 结束时更新波特率 (BR6)。以 BR 对位 0 进行采样，以 BR0 对位 1 到位 6 进行采样，以 BR6 对字符的其他位进行采样。同时，对 RX 线路的各个中间跳变执行其他检查。如果 RX 上的跳变与接收器（基于根据位 0 计算的波特率的接收器）未充分同步，则生成错误。

激活自动波特率检测之前，必须先通过向 USART_BRR 寄存器写入非零的波特率值来初始化该寄存器。

通过将 USART_CR2 寄存器中的 ABREN 位置 1 来激活自动波特率检测。之后 USART 会等待 RX 线路上的第一个字符。通过将 USART_ISR 寄存器中的 ABRF 标志置 1 来指示自动波特率操作完成。如果线路噪声较大，则无法保证正确的波特率检测。这种情况下，BRR 值可能会损坏，ABRE 错误标志位会置 1。如果通信速度不在自动波特率检测范围（位持续时间不在 16 个和 65536 个时钟周期（16 倍过采样时）之间，也不在 8 个和 65536 个时钟周期（8 倍过采样时）之间）内，也会出现这种情况。

稍后，可通过复位 ABRF 标志（通过写入 0）重新启动自动波特率检测。

禁止 FIFO 管理且发生自动波特率错误时，ABRE 标志通过 RXNE 和 FE 位置 1。

使能 FIFO 管理且发生自动波特率错误时，ABRE 标志通过 RXFNE 和 FE 位置 1。

如果使能 FIFO 模式，则必须使用第一个 RXFIFO 位置上的数据进行自动波特率检测。因此，在启动自动波特率检测之前，请通过检查 USART_ISR 寄存器的 RXFNE 标志来确保 RXFIFO 为空。

注意： *如果在自动波特率操作期间禁止 USART (UE=0)，则可能损坏 BRR 值。*

36.5.11 USART 多处理器通信

可以执行 USART 多处理器通信（多个 USART 连接在一个网络中）。例如，其中一个 USART 可以是主 USART，其 TX 输出与其他 USART 的 RX 输入相连；而其他 USART 为从 USART，其各自的 TX 输出在逻辑上通过与运算连在一起，并与主 USART 的 RX 输入相连。

在多处理器配置中，理想情况下通常只有预期的消息接收方主动接收完整的消息内容，从而减少由所有未被寻址的接收器造成的冗余 USART 服务开销。

可通过静默功能将未被寻址的器件置于静默模式下。为了使用静默模式功能，必须将 USART_CR1 寄存器中的 MME 位置 1。

注意： *使能 FIFO 管理且 MME 已置 1 时，不得清零 MME 位再快速将其置 1（在两个 usart_ker_ck 周期内），否则静默模式可能保持有效。*

使能静默模式时：

- 不得将接收状态位置 1。
- 禁止任何接收中断。
- USART_ISR 寄存器中的 RWU 位置 1。在某些情况下，RWU 可以由硬件或软件通过 USART_RQR 寄存器中的 MMRQ 位自动控制。

根据 USART_CR1 寄存器中 WAKE 位的设置，USART 可使用以下两种方法进入或退出静默模式：

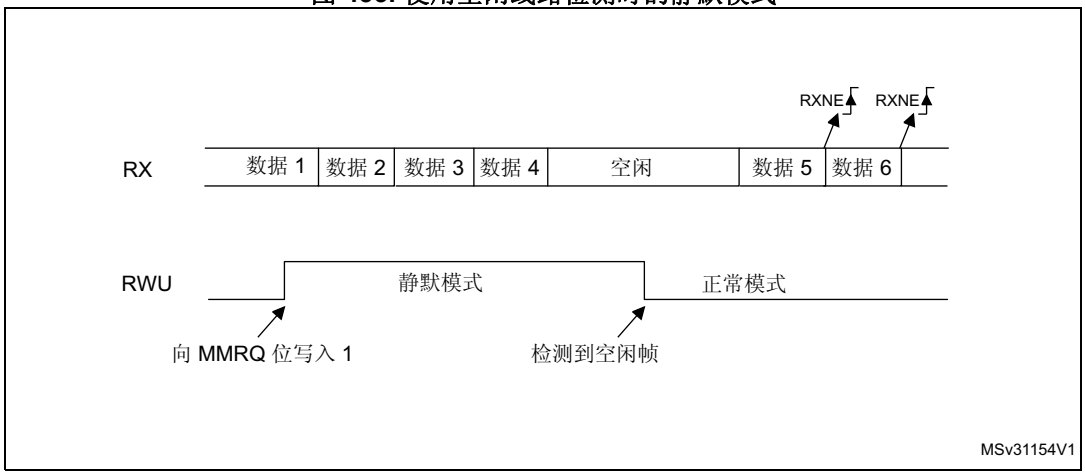
- 如果 WAKE 位被复位，则进行空闲线路检测，
- 如果 WAKE 位置 1，则进行地址标记检测。

空闲线路检测 (WAKE=0)

当向 MMRQ 位写入 1 且 RWU 位自动置 1 时，USART 进入静默模式。

检测到空闲帧时，USART 将唤醒。此时 RWU 位会由硬件清零，但 USART_ISR 寄存器中的 IDLE 位不会置 1。*图 433* 中给出了使用空闲线路检测时静默模式行为的示例。

图 433. 使用空闲线路检测时的静默模式



注意： *如果在 IDLE 字符已经过去时将 MMRQ 位置 1，则不会进入静默模式（RWU 未置 1）。*

如果在线路处于空闲状态时激活 USART，在一个 IDLE 帧持续时间后（不只在接收一个字符帧后）会检测到空闲状态。

4 位/7 位地址标记检测 (WAKE=1)

在此模式下，如果字节的 MSB 为 1，则将这些字节识别为地址，否则将其识别为数据。在地址字节中，目标接收器的地址位于 4 个或 7 个 LSB 中。7 位或 4 位地址检测通过 ADDM7 位来选择。接收器会将此 4 位/7 位字与其地址进行比较，该接收器的地址在 USART_CR2 寄存器的 ADD 位中进行设置。

注意： *在 7 位和 9 位数据模式下，地址检测分别在 6 位和 8 位地址上完成（ADD[5:0] 和 ADD[7:0]）。*

当接收到与其编程地址不匹配的地址字符时，USART 会进入静默模式。此时，RWU 位将由硬件置 1。USART 进入静默模式后，RXNE 标志不会针对此地址字节置 1，也不会发出中断或 DMA 请求。使能 FIFO 管理时，软件必须确保在进入静默模式之前 RXFIFO 中至少有一个空位置。

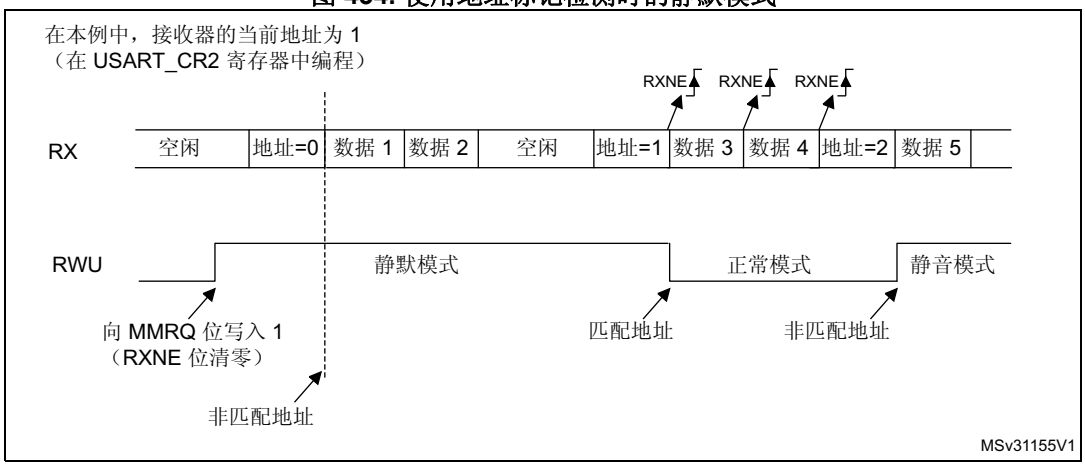
当向 MMRQ 位写入 1 时，USART 也会进入静默模式。这种情况下，RWU 位也自动置 1。

当接收到与编程地址匹配的地址字符时，USART 会退出静默模式。然后 RWU 位被清零，可以开始正常接收后续字节。由于 RWU 位已清零，RXNE/RXFNE 位会针对地址字符置 1。

注意： *使能 FIFO 管理时，如果在接收器对数据的最后一位进行采样时 MMRQ 置 1，则可在有效进入静默模式之前接收该数据*

图 434 中给出了使用地址标记检测时静默模式行为的示例。

图 434. 使用地址标记检测时的静默模式



36.5.12 USART Modbus 通信

USART 为 Modbus/RTU 和 Modbus/ASCII 协议的实现提供基本支持。Modbus/RTU 是一个半双工块传输协议。该协议的控制部分（地址识别、块完整性控制和命令解析）必须用软件实现。

USART 为块结束检测提供基本支持，无需软件开销或其他资源。

Modbus/RTU

在此模式下，一个块的结束通过超过 2 个字符时间的“静默”（空闲线路）来识别。此功能通过可编程的超时功能实现。

超时功能和中断必须分别通过 USART_CR2 寄存器中的 RTOEN 位和 USART_CR1 寄存器中的 RTOIE 位激活。与 2 个字符时间（例如 22 个位时间）的超时相对应的值必须在 RTO 寄存器中编程。如果在此期间接收线路空闲，则在接收到最后一个停止位后，将生成中断，同时通知软件当前块接收已完成。

Modbus/ASCII

在此模式下，块结束通过特定 (CR/LF) 字符序列识别。USART 通过字符匹配功能管理此机制。

通过在 ADD[7:0] 位域中编程 LF ASCII 码以及激活字符匹配中断 (CMIE = 1)，软件可在接收到 LF 时获得通知并检查 DMA 缓存区中的 CR/LF。

36.5.13 USART 奇偶性控制

将 USART_CR1 寄存器中的 PCE 位置 1，可以使能奇偶校验控制（发送时生成奇偶校验位，接收时进行奇偶校验检查）。根据 M 位定义的帧长度，*表 290* 中列出了可能的 USART 帧格式。

表 290. USART 帧格式

M 位	PCE 位	USART 帧(1)
00	0	| SB | 8 位数据 | STB |
00	1	| SB | 7 位数据 | PB | STB |
01	0	| SB | 9 位数据 | STB |
01	1	| SB | 8 位数据 PB | STB |
10	0	| SB | 7 位数据 | STB |
10	1	| SB | 6 位数据 | PB | STB |

1. 图注：SB：起始位，STB：停止位，PB：奇偶校验位。在数据寄存器中，PB 始终位于 MSB 位置（第 8 位或第 7 位，具体取决于 M 位的值）。

偶校验

对奇偶校验位进行计算，使帧和奇偶校验位中“1”的数量为偶数（帧由 6 个、7 个或 8 个 LSB 位组成，具体取决于 M 位的值）。

例如，如果数据 = 00110101 且 4 个位置 1，则在选择偶校验（USART_CR1 寄存器中的 PS 位 = 0）时，奇偶校验位等于 0。

奇校验

对奇偶校验位进行计算，使帧和奇偶校验位中“1”的数量为奇数（帧由 6 个、7 个或 8 个 LSB 位组成，具体取决于 M 位的值）。

例如，如果数据 = 00110101 且 4 个位置 1，则在选择奇校验（USART_CR1 寄存器中的 PS 位 = 1）时，奇偶校验位等于 1。

接收时进行奇偶校验检查

如果奇偶校验检查失败，则 USART_ISR 寄存器中的 PE 标志置 1；如果 USART_CR1 寄存器中 PEIE 位置 1，则会生成中断。通过软件将 1 写入 USART_ICR 寄存器中的 PECF 位来清零 PE 标志。

发送时的奇偶校验生成

如果 USART_CR1 寄存器中的 PCE 位置 1，则在数据寄存器中所写入数据的 MSB 位会进行发送，但是会由奇偶校验位进行更改（如果选择偶校验 (PS=0)，则“1”的数量为偶数；如果选择奇校验 (PS=1)，则“1”的数量为奇数）。

36.5.14 USART LIN（局域互连网络）模式

仅在支持 LIN 模式时才与本节相关。请参见*第 1253 页的第 36.4 节：USART 实现*。

通过将 USART_CR2 寄存器中的 LINEN 位置 1 来选择 LIN 模式。在 LIN 模式下，必须将以下位清零：

- USART_CR2 寄存器中的 CLKEN 位，
- USART_CR3 寄存器中的 STOP[1:0]、SCEN、HDSEL 和 IREN 位。

LIN 发送

与正常的 USART 发送相比，在 LIN 主设备中发送时必须采用*第 36.5.5 节*中介绍的步骤，同时还具有以下区别：

- M 位清零以配置 8 位字长度。
- LINEN 位置 1 以进入 LIN 模式。此时，将 SBKRQ 位置 1 会发送 13 个 0 位作为刹车字符。然后会发送值为“1”的 2 个位以进行下一启动检测。

LIN 接收

使能 LIN 模式后，将激活刹车检测电路。该检测完全独立于正常的 USART 接收器。在空闲状态或某个帧期间，只要发生刹车即可检测出来。

接收器（USART_CR1 寄存器中 RE=1）使能后，电路便开始监测启动信号的 RX 输入。检测起始位的方法与搜索刹车字符或数据的方法相同。检测到起始位后，电路会对接下来的位进行采样，方法与数据采样相同（第 8、第 9 和第 10 次采样）。如果 10 个（USART_CR2 寄存器中 LBDL = 0 时）或 11 个（USART_CR2 寄存器中 LBDL=1 时）连续位均检测为 0，且其后跟随分隔符，则 USART_ISR 寄存器中的 LBDF 标志将会置 1。如果 LBDIE 位=1，则会生成中断。在验证刹车前，会对分隔符进行检查，因为它表示 RX 线路已恢复到高电平。

如果在第 10 或第 11 次采样前采样到“1”信号，则刹车帧检测电路会取消当前检测，并重新搜索起始位。

如果禁止 LIN 模式 (LINEN=0)，接收器会作为正常的 USART 继续工作，不会再进行刹车帧检测。

如果使能 LIN 模式 (LINEN=1)，只要发生帧错误（例如，在 0 处检测到停止位，这种情况可能出现在任何刹车帧中），接收器即会停止，直到刹车帧检测电路接收到“1”（刹车帧字不完整时）或接收到分隔符（检测到刹车帧时）为止。

*第 1278 页的图 435：LIN 模式下的刹车检测（11 位刹车长度——LBDL 位置 1）*中显示了刹车检测器状态机和刹车标志的行为。

*第 1279 页的图 436：LIN 模式下的刹车检测与帧错误检测*中列出了刹车帧的示例。

图 435. LIN 模式下的刹车检测（11 位刹车长度——LBDL 位置 1）

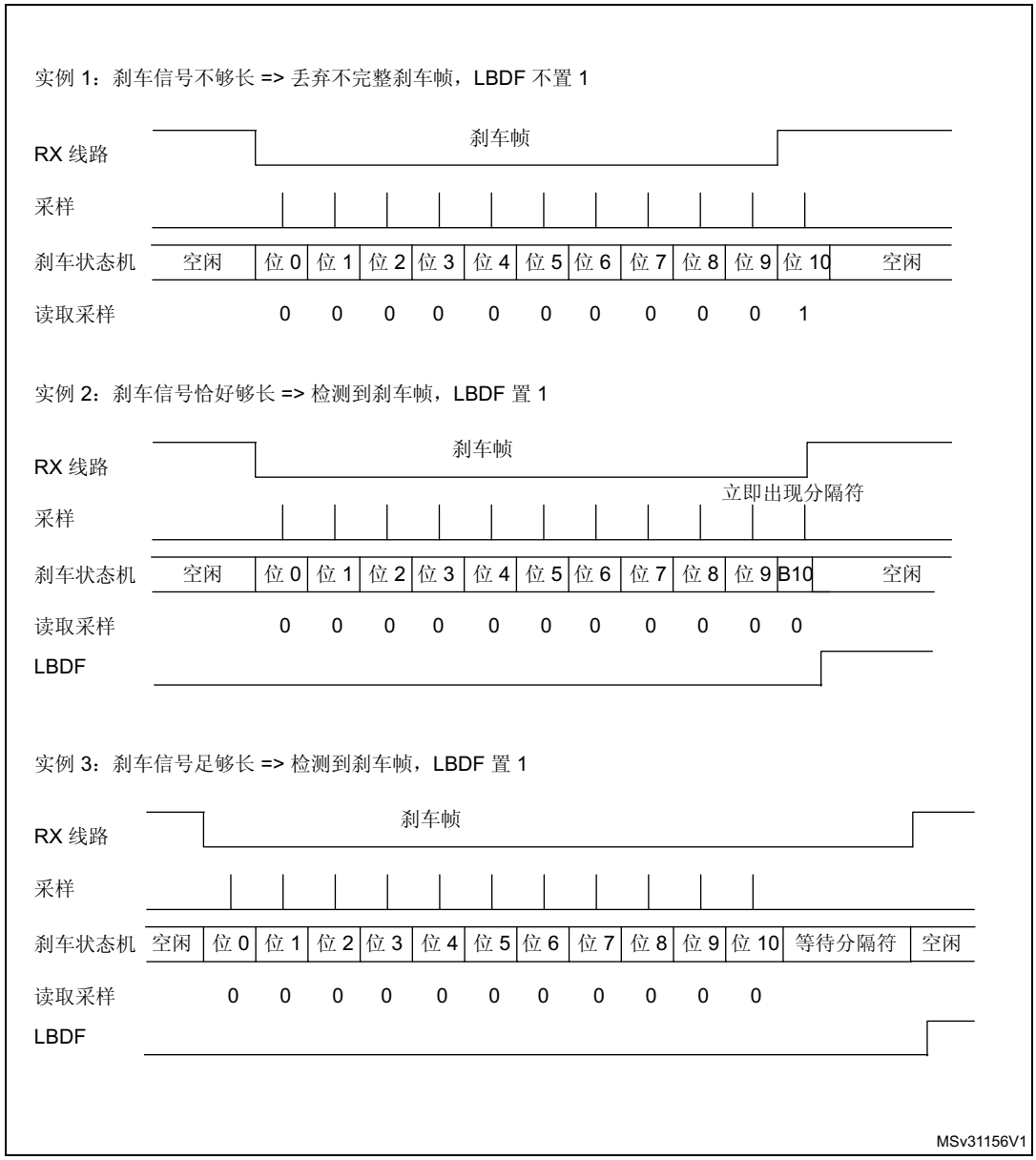
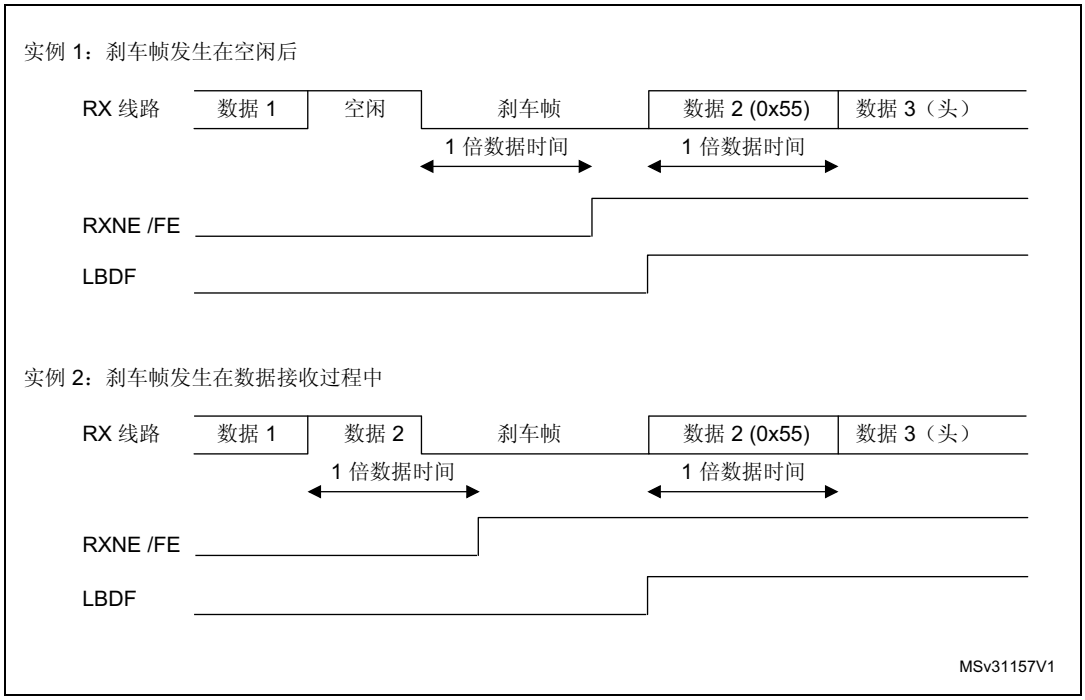


图 436. LIN 模式下的刹车检测与帧错误检测



36.5.15 USART 同步模式

主模式

通过将 USART_CR2 寄存器中的 CLKEN 位编程为 1 来选择同步主模式。在同步模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN 位，
- USART_CR3 寄存器中的 SCEN、HDSEL 和 IREN 位。

在此模式下，USART 可用于在主模式下控制双向同步串行通信。CK 引脚是 USART 发送器时钟的输出。在起始位或停止位期间，不会向 CK 引脚发送时钟脉冲。在最后一个有效数据位（地址标记）期间，是否生成时钟脉冲取决于 USART_CR2 寄存器中 LBCL 位的状态。USART_CR2 寄存器中的 CPOL 位用于选择时钟极性；USART_CR2 寄存器中的 CPHA 位用于选择外部时钟的相位（请参见*图 437*、*图 438* 和*图 439*）。

在空闲状态、报头模式和发送刹车期间，外部 CK 时钟处于未激活状态。

在同步主模式下，USART 发送器的工作方式与异步模式下完全相同。但是，由于 CK 与 TX 同步（根据 CPOL 和 CPHA），因此 TX 上的数据是同步的。

在同步主模式下，USART 接收器的工作方式与异步模式下不同。如果 RE 置 1，则数据在 CK 上采样（上升或下降沿，具体取决于 CPOL 和 CPHA），而不会进行任何过采样。此时必须确保给定建立时间和保持时间（取决于波特率：1/16 位时间）符合要求。

注意： *在主模式下，CK 引脚可与 TX 引脚结合使用。因此，仅当使能发送器 (TE=1) 且正在发送数据时（USART_TDR 数据寄存器已写入），才会提供时钟。这意味着，没有发送数据的情况下无法接收同步数据。*

图 437. USART 同步主发送示例

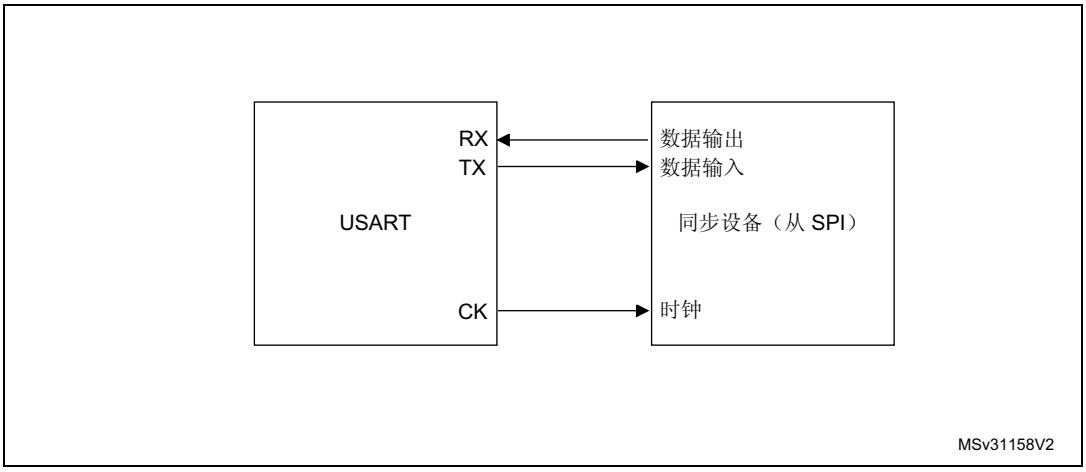


图 438. USART 在同步主模式下的数据时钟时序图（M 位 = 00）

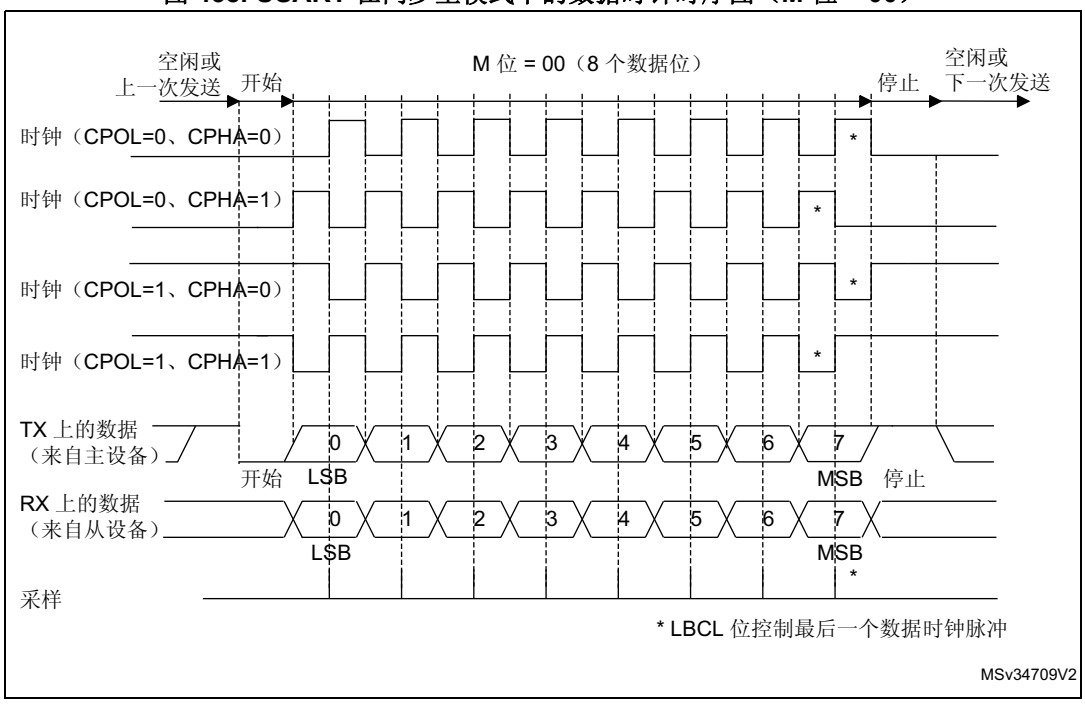
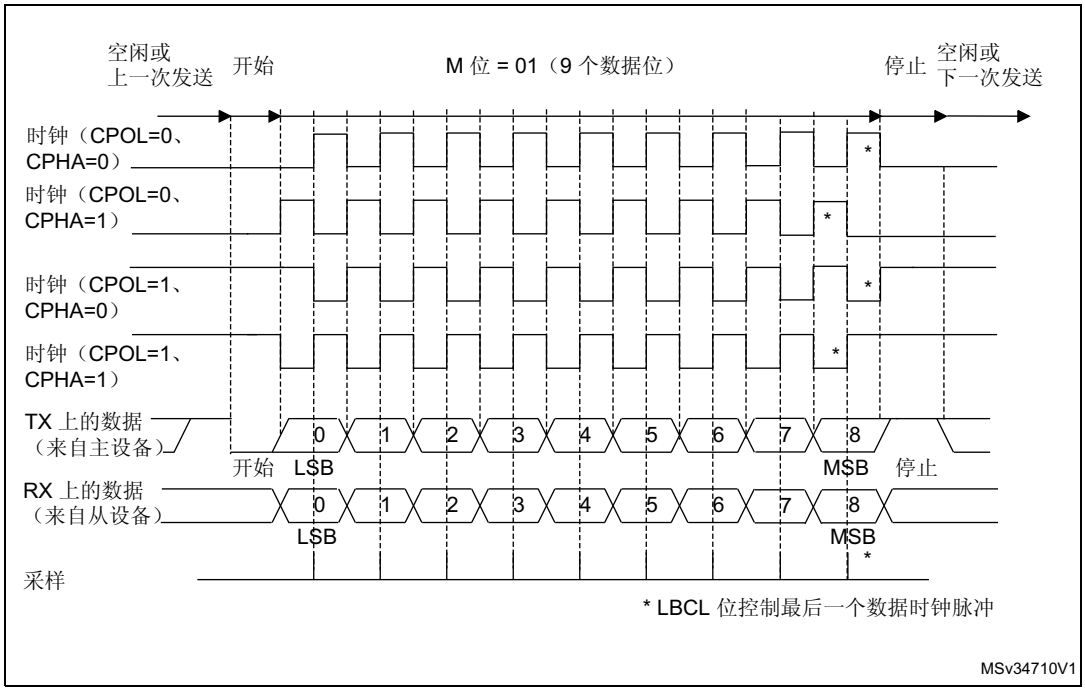


图 439. USART 在同步主模式下的数据时钟时序图（M 位 = 01）



从模式

通过将 USART_CR2 寄存器中的 SLVEN 位编程为 1 来选择同步从模式。在同步从模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN 和 CLKEN 位，
- USART_CR3 寄存器中的 SCEN、HDSEL 和 IREN 位。

在此模式下，USART 可用于在从模式下控制双向同步串行通信。在从模式下，CK 引脚是 USART 的输入。

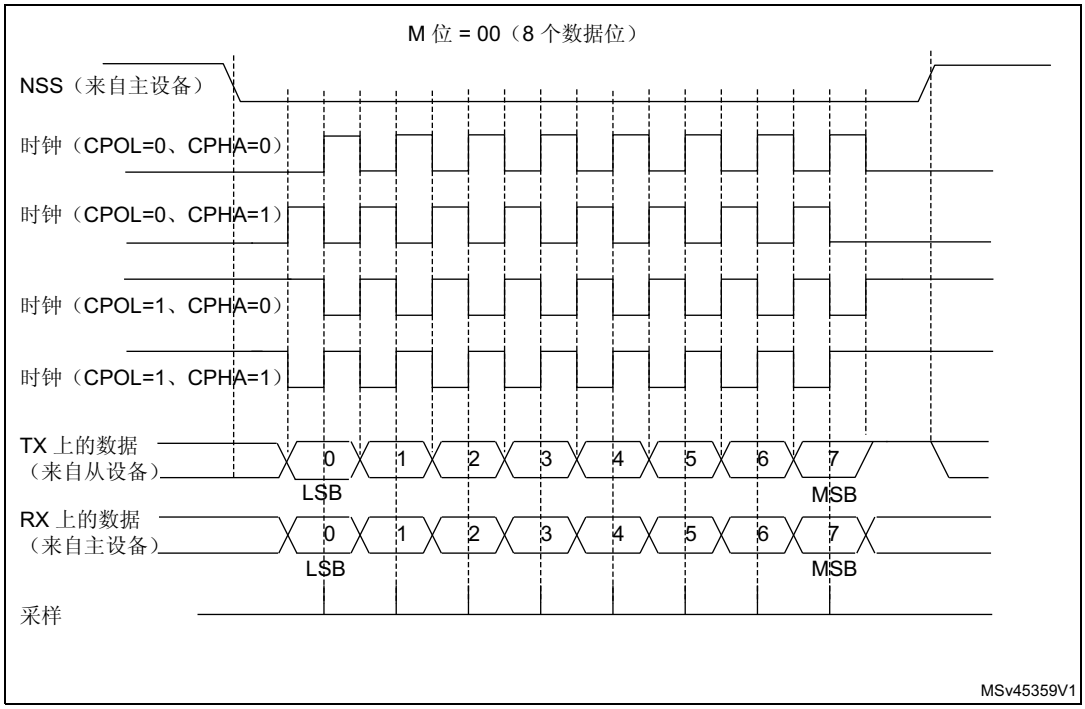
注意： *当外设用于 SPI 从设备模式时，外设时钟源 (*usart_ker_ck_pres*) 的频率必须是 CK 输入频率的 3 倍以上。*

USART_CR2 寄存器中的 CPOL 位和 CPHA 位分别用于选择时钟极性和外部时钟的相位（请参见*图 440*）。

从发送模式支持下溢错误标志。如果在软件尚未将任何值加载到 USART_TDR 之前出现第一个数据发送时钟脉冲，则此标志将置 1。

从设备支持硬件和软件 NSS 管理。

图 440. USART 在同步从模式下的数据时钟时序图（M 位 = 00）



从设备选择 (NSS) 引脚管理

可通过 USART_CR2 寄存器中的 DIS_NSS 位设置硬件或软件从设备选择管理：

- 软件 NSS 管理 (DIS_NSS = 1)
  始终选择 SPI 从设备，忽略 NSS 输入引脚。
  外部 NSS 引脚空闲，可供其他应用使用。
- 硬件 NSS 管理 (DIS_NSS = 0)
  SPI 从设备选择取决于 NSS 输入引脚。NSS 为低电平时选择从设备，NSS 为高电平时取消选择从设备。

注意： *禁止 USART 时 (UE=0)，必须选择 LBCL（仅用于 SPI 主设备模式）、CPOL 和 CPHA 位以确保时钟脉冲正常工作。*

在 SPI 从设备模式下，必须在启动主设备通信之前（或时钟稳定时在相邻帧之间）使能 USART。否则，如果 USART 从设备在主设备处于某个帧中间时使能，则它会与主设备失去同步。在通信时钟的第一个边沿到来之前或者正在进行的通信结束之前，从设备的数据寄存器就需要准备就绪，否则 SPI 从设备会发送零。

SPI 从设备下溢错误

发生下溢错误时，USART_ISR 寄存器中的 UDR 标志会置 1，SPI 从设备继续发送最后一个数据，直到下溢错误标志由软件清零。

下溢标志在帧开始时置 1。如果 USART_CR3 寄存器中的 EIE 位置 1，则会触发下溢错误中断。

通过将 USART_ICR 寄存器中的位 UDRCF 置 1 来清零下溢错误标志。

发生下溢错误时，仍然可以对 TDR 寄存器进行写操作。清除下溢错误将允许发送新数据。

如果发生下溢错误且没有新数据被写入 TDR 中，则 TC 标志在帧结束时置 1。

注意： *如果向 USART_TDR 写入数据的时间点过于接近第一个 CK 发送边沿，则可能会发生下溢错误。为避免发生此下溢错误，必须在第一个 CK 边沿的 3 个* usart_ker_ck *周期之前对 USART_TDR 进行写操作。*

36.5.16 USART 单线半双工通信

通过将 USART_CR3 寄存器中的 HDSEL 位置 1 来选择单线半双工模式。在此模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN 和 CLKEN 位。
- USART_CR3 寄存器中的 SCEN 和 IREN 位。

USART 可以配置为遵循单线半双工协议，其中 TX 和 RX 线路从内部相连接。使用 USART_CR3 寄存器中的控制位 HDSEL 可在半双工通信和全双工通信间进行选择。

一旦向 HDSEL 位写入 1：

- TX 和 RX 线路从内部相连接。
- 不再使用 RX 引脚。
- 无数据传输时，TX 引脚始终处于释放状态。因此，它在空闲状态或接收过程中用作标准 I/O。这意味着，必须对 I/O 进行配置，以便将 TX 配置为复用功能开漏并外接上拉电阻。

除此之外，通信协议与正常 USART 模式下的通信协议相似。此线路上的任何冲突都必须由软件管理（例如，使用中央仲裁器）。尤其要注意，发送过程永远不会被硬件阻塞，只要数据是在 TE 位置 1 的情况下写入，发送就会持续进行。

36.5.17 USART 接收器超时

接收器超时功能可通过将 USART_CR2 控制寄存器中的 RTOEN 位置 1 来使能。

超时间隔通过 USART_RTOR 寄存器中的 RTO 位域进行编程。

接收器超时计数器遵循以下规则开始计数：

- STOP = 00 或 STOP = 11 时从停止位结束时开始计数。
- STOP = 10 时从第二个停止位结束时开始计数。
- STOP = 01 时从停止位开始时开始计数。

经过超时间隔后，USART_ISR 寄存器中的 RTOF 标志置 1。如果 USART_CR1 寄存器中的 RTOIE 位置 1，则会产生超时。

36.5.18 USART 智能卡模式

仅在支持智能卡模式时才涉及本节内容。请参见*第 1253 页的第 36.4 节：USART 实现*。

通过将 USART_CR3 寄存器中的 SCEN 位置 1 选择智能卡模式。在智能卡模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN 位，
- USART_CR3 寄存器中的 HDSEL 和 IREN 位。

此外，还可将 CLKEN 位置 1，以便为智能卡提供时钟。

智能卡接口支持符合 ISO 7816-3 标准的异步智能卡协议。同时支持 T=0（字符模式）和 T=1（块模式）。

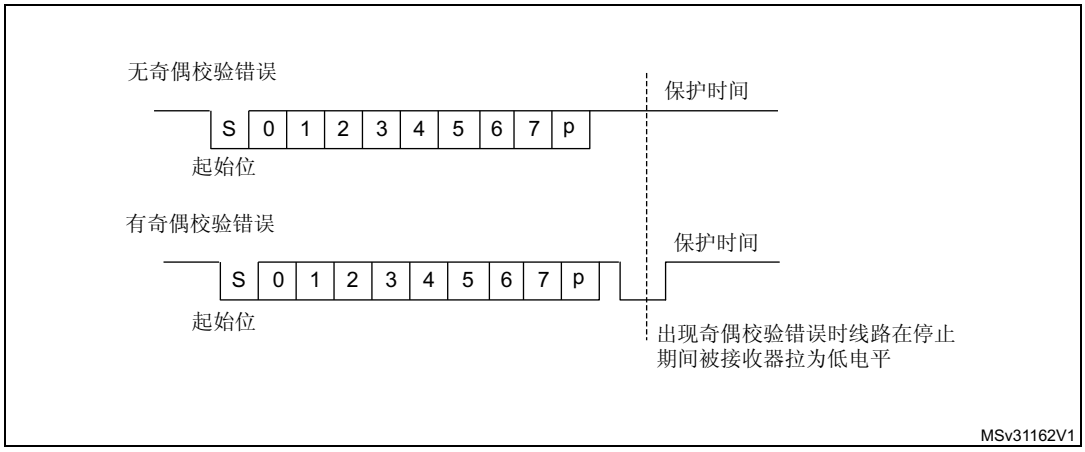
USART 必须如下所示进行配置：

- 8 个位加奇偶校验：当 USART_CR1 寄存器中的 M = 1 且 PCE = 1 时
- 发送和接收数据时使用 1.5 个停止位：USART_CR2 寄存器中的 STOP = 11。接收时也可以选择 0.5 个停止位。

在 T=0（字符）模式下，奇偶校验错误在保护时间周期内的每个字符结束时指示。

图 441 显示了有奇偶校验错误和无奇偶校验错误时数据线上情况的示例。

图 441. ISO 7816-3 异步协议



连接到智能卡时，USART 的 TX 输出会驱动一条双向线（它也由智能卡驱动）。必须将 TX 引脚配置为开漏引脚。

智能卡模式采用单线半双工通信协议。

- 从发送移位寄存器发送数据会经过至少 1/2 个时钟周期的延迟。正常工作时，已满的发送移位寄存器会在下一个波特时钟边沿开始移位。在智能卡模式下，此发送过程还会进一步经过 1/2 波特时钟周期的延迟。
- 发送时，如果智能卡检测到奇偶校验错误，它会通过将线路驱动为低电平 (NACK) 告知 USART 此状态。此 NACK 信号（将发送线拉低 1 个时钟周期）会导致发送器端（配置为 1.5 个停止位）出现帧错误。USART 可根据协议自动重新发送数据。重试次数在 SCARCNT 位域中编程。如果经过编程次数的重试后，USART 继续收到 NACK，USART 会停止发送并将该错误以帧错误形式发出。TXE 位（使能 FIFO 模式时为 TXFNF 位）可使用 USART_RQR 寄存器中的 TXFRQ 位置 1。

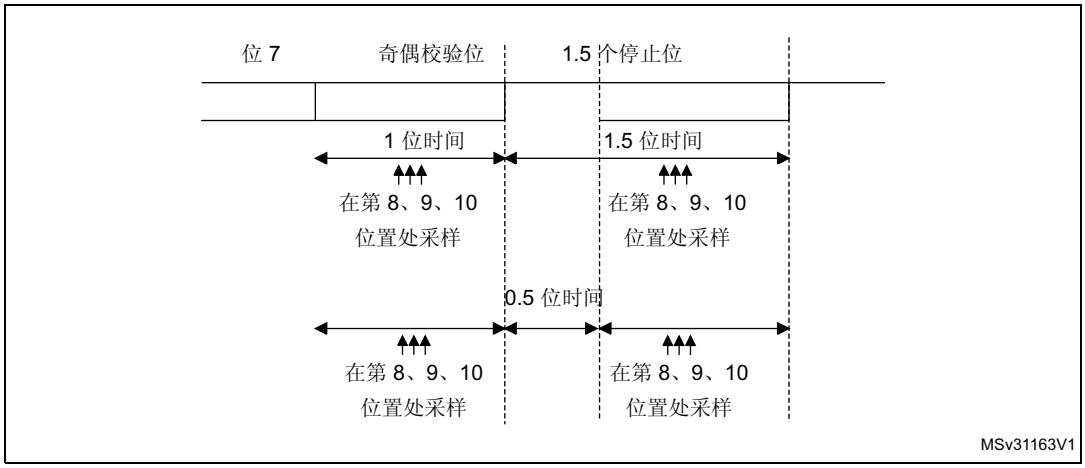
- 发送时智能卡自动重试：在 USART 检测 NACK 与重复字符的起始位之间插入 2.5 个波特率周期的延迟。接收最后一个重复字符后，立即将 TC 位置 1（无保护时间）。如果软件要重复此操作，必须确保标准要求的最短 2 个波特率周期。
- 如果在接收一个使用 1.5 个停止位编程的帧期间检测到奇偶校验错误，则在完成接收帧后，发送线会被拉低一个时钟周期。这是为了向智能卡指出发送到 USART 的数据尚未正确接收。如果 NACK 控制位置 1，接收器会向奇偶校验错误发送“NACK”信号；否则不会发送 NACK 信号（将在 T=1 模式下会使用）。如果接收到的字符错误，则不会激活 RXNE（使能 FIFO 模式时为 RXFNE）/接收 DMA 请求。根据协议规范，智能卡必须重新发送相同的字符。如果经过 SCARCNT 位域中指定的最大重试次数后接收到的字符仍然错误，USART 会停止发送 NACK 信号，并将错误以奇偶校验错误的形式发出。
- 接收时智能卡自动重试：如果 USART 向智能卡发送 NACK 信号，但智能卡不重复字符，则 BUSY 标志将保持置 1。
- 发送时，USART 会在两个连续字符之间插入保护时间（按照保护时间寄存器中编程的值）。由于保护时间在前一个字符的停止位后测量，因此必须将 GT[7:0] 寄存器编程为所需 CGT（字符保护时间，如 7816-3 规范所定义）减去 12（一个字符的持续时间）。
- 通过对保护时间寄存器进行编程，可以延迟 TC 标志的置位。正常工作时，当发送移位寄存器为空且没有新的发送请求出现时，会对 TC 标志进行置位。在智能卡模式下，空的发送移位寄存器会触发保护时间计数器，使其递增计数至保护时间寄存器中的值。在此期间，TC 标志被强制为低电平。当保护时间计数器达到设置值时，TC 置位为高电平。TCBGT 标志可用于检测数据传输是否结束，而无需等待保护时间结束。该标志在帧发送结束之后且未从智能卡接收到 NACK 时置 1。
- TC 标志的释放不受智能卡模式的影响。
- 如果在发送端检测到帧错误（由来自接收器的 NACK 信号引起），则发送端的接收块不会将 NACK 作为起始位进行检测。根据 ISO 协议，接收到的 NACK 信号的持续时间可以是 1 或 2 个时钟周期。
- 在接收端，如果检测到奇偶校验错误并发送了 NACK 信号，则接收端不会将 NACK 作为起始位进行检测。

注意： *刹车字符在智能卡模式下无效。带有帧错误的 0x00 数据被视为数据，而非刹车。*

当翻转 TE 位时，不会发送空闲帧。空闲帧（在其他配置中进行了定义）在 ISO 协议中未进行定义。

图 442 介绍了 USART 如何对 NACK 信号采样。在本例中，USART 正在发送数据并配置了 1.5 个停止位。USART 的接收部分已被使能，以检查数据的完整性和 NACK 信号。

图 442. 使用 1.5 个停止位检测奇偶校验错误



USART 可以通过 CK 输出为智能卡提供时钟。在智能卡模式下，CK 仅通过一个 5 位预分频器由内部外设输入时钟提供。分频比在 USART_GTPR 寄存器中进行配置。CK 频率可在 usart_ker_ck_pres/2 到 usart_ker_ck_pres/62 之间进行编程，其中 usart_ker_ck_pres 为经过编程的预分频器分频的外设输入时钟。

块模式 (T=1)

在 T=1（块）模式下，通过将 UART_CR3 寄存器中的 NACK 位清零可停用奇偶校验错误发送。

在块模式下请求从智能卡执行读操作时，软件必须将 RTOR 寄存器编程为 BWT（块等待时间）- 11。如果在经过此时间段后未从智能卡接收到应答，将生成超时中断。如果在该时间段超时之前接收到第一个字符，则通过 RXNE/RXFNE 中断发出信号指示。

注意： *即使在使用 DMA 模式下的 USART 从块模式下的智能卡读取时，也必须使能 RXNE/RXFNE 中断。同时，只有在接收到第一个字节后才可使能 DMA。*

接收到第一个字符（RXNE/RXFNE 中断）后，为允许自动检查两个连续字符间的最长等待时间，必须将 RTO 寄存器编程为 CWT（字符等待时间 - 值 11）。此时间以波特率时间单位表示。前一个字符结束后，如果智能卡未在小于 CWT 的时间段内发送新字符，USART 将通过 RTOF 标志和中断（当 RTOIE 位置 1 时）向软件指示此情况。

注意： *按照智能卡协议定义，BWT/CWT 的值必须从最后一个字符开始（起始位）时定义。必须将 RTO 寄存器分别编程为 BWT -11 或 CWT -11，并考虑最后一个字符本身的长度。*

块长度计数器用于对 USART 接收到的所有字符进行计数。当 USART 进行发送时，此计数器复位。块长度由智能卡在块的第三个字节（起始字段）中发送。必须将此值编程到 USART_RTOR 寄存器中的 BLEN 位域。使用 DMA 模式时，在块开始之前，必须将该寄存器位域编程为最小值 (0x0)。对于该值，在接收到第四个字符后生成中断。软件必须读取 LEN 字段（第三个字节），其值必须从接收缓存区中读取。

在中断驱动接收模式下，块长度可以由软件或者通过编程 BLEN 值来检查。但是在块开始前，可以编程 BLEN 的最大值 (0xFF)。接收到第三个字符后，将编程实际值。

如果块使用 LRC（纵向冗余校验，1 个结尾字节），则 BLEN = LEN。如果块使用 CRC 机制（2 个结尾字节），则必须编程 BLEN = LEN+1。块总长度（包括起始字段、结尾字段和信息字段）等于 BLEN+4。块结束的信号通过 EOBF 标志和中断（EOBIE 位置 1 时）发送给软件。

如果块长度出现错误，则通过 RTO 中断（字符等待时间上溢）发送块结束信号。

注意： *错误检查代码 (LRC/CRC) 必须通过软件计算/验证。*

正向约定和反向约定

智能卡协议定义了两种约定：正向约定和反向约定。

正向约定定义为：LSB 在前，逻辑位值 1 对应于线路的 H 状态，奇偶校验为偶校验。要使用此约定，必须编程以下控制位：MSBFIRST = 0，DATAINV = 0（默认值）。

反向约定定义为：MSB 在前，逻辑位值 1 对应于信号线路的 L 状态，奇偶校验为偶校验。要使用此约定，必须编程以下控制位：MSBFIRST = 1，DATAINV = 1。

注意： *将逻辑数据值取反（0=H，1=L）时，奇偶校验位将同样取反。*

为识别智能卡约定，智能卡会将初始字符 TS 作为 ATR（复位应答）的第一个字符发送。TS 支持两种格式：LHHL LLL LLH 和 LHHL HHH LLH。

- (H) LHHL LLL LLH 建立反向约定：状态 L 编码为值 1，时刻 2 传送最高有效位（MSB 在前）。按反向约定解码时，传送的字节等于“3F”。
- (H) LHHL HHH LLH 建立正向约定：状态 H 编码为值 1，时刻 2 传送最低有效位（LSB 在前）。按正向约定解码时，传送的字节等于“3B”。

在 2 到 10 的九个时刻中，如果有偶数个位设置为 1，则字符的奇偶校验正确。

由于 USART 不了解智能卡使用哪种约定，因此 USART 需要能够识别任意一种模式并相应操作。模式识别不在硬件中完成，而是通过软件序列完成。此外，假设以正向约定配置 USART（默认）而智能卡以反向约定 (TS = LHHL LLL LLH) 应答，则 USART 接收到的字节将等于 03，奇偶校验将为奇校验。

因此，有两种方法可用于识别 TS 模式：

方法 1

将 USART 编程为标准智能卡模式/正向约定。这种情况下，TS 模式接收会向智能卡生成奇偶校验错误中断和错误信号。

- 奇偶校验错误中断通知软件智能卡未以正向约定正确应答。之后，软件重新将 USART 编程为反向约定
- 为响应错误信号，智能卡会重试同一 TS 字符，此时重新编程后的 USART 将正确接收该字符

或者，为应答奇偶校验错误中断，软件可决定重新编程 USART，同时向智能卡生成新的复位命令，然后再次等待 TS。

方法 2

将 USART 编程为 9 位/无奇偶校验模式，无位反向。在此模式下，按如下方式接收两种 TS 模式中的任一种：

(H) LHHL LLL LLH = 0x103 -> 选择反向约定

(H) LHHL HHH LLH = 0x13B -> 选择正向约定

软件根据这两种模式检查接收到的字符，如果其中任意一种匹配，则相应编程 USART 以接收下一个字符。

如果两种都未被识别，则可能复位智能卡以重新开始协商。

36.5.19 USART IrDA SIR ENDEC 模块

仅在支持 IrDA 模式时才涉及本节内容。请参见*第 1253 页的第 36.4 节：USART 实现*。

通过将 USART_CR3 寄存器中的 IREN 位置 1 来选择 IrDA 模式。在 IrDA 模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN、STOP 和 CLKEN 位，
- USART_CR3 寄存器中的 SCEN 和 HDSEL 位。

IrDA SIR 物理层规定使用反相归零 (RZI) 调制方案，它以红外光脉冲表示逻辑 0（参见*图 443*）。

SIR 发送编码器用于调制 USART 发出的非归零 (NRZ) 位流。输出脉冲流会发送到外部输出驱动器和红外线 LED。USART 支持的 SIR ENDEC 比特率最高为 115.2 kbaud。在正常模式下，所发送的脉冲宽度规定为一个位周期的 3/16。

SIR 接收解码器用于解调由红外探测器发出的归零位流，并将接收到的 NRZ 串行位流输出到 USART。在空闲状态下，解码器输入通常为高电平（标记状态）。发送编码器输出的极性与解码器输入相反。当解码器输入为低电平时，会检测到起始位。

- IrDA 是一个半双工通信协议。如果发送器忙（USART 正在向 IrDA 编码器发送数据时），则 IrDA 解码器会忽略 IrDA 接收线上的所有数据；如果接收器忙（USART 正在接收来自 USART 的解码数据时），则 IrDA 不会对 USART 发送到 IrDA 的 TX 上的数据进行编码。在接收数据时，必须避免同时进行发送，因为这样做可能会破坏要发送的数据。
- 0 作为高电平脉冲发送，而 1 作为 0 发送。在正常模式下，脉冲宽度规定为所选位周期的 3/16（请参见*图 444*）。
- SIR 解码器用于将兼容 IrDA 的接收信号转换为 USART 的位流。
- SIR 接收逻辑将高电平状态视为逻辑“1”，将低电平脉冲视为逻辑“0”。
- 发送编码器输出的极性与解码器输入相反。SIR 输出在空闲时处于低电平状态。
- IrDA 规范要求脉冲容忍值要大于 1.41 µs。可接受的脉冲宽度可通过寄存器设置。接收器端的干扰检测逻辑会滤除宽度小于 2 个 PSC 周期的脉冲（PSC 是在 USART_GTPR 中编程的预分频器值）。宽度小于 1 个 PSC 周期的脉冲都将被拒绝，但宽度大于 1 个而小于 2 个周期的脉冲可能被接受也可能被拒绝，而宽度大于 2 个周期的脉冲将被接受作为有效脉冲。当 PSC=0 时，IrDA 编码器/解码器不工作。
- 接收器能够与低功耗发送器进行通信。
- 在 IrDA 模式下，USART_CR2 寄存器中的停止位必须配置为“1 个停止位”。

IrDA 低功耗模式

- 发送器

  在低功耗模式下，脉冲宽度不再保持为位周期的 3/16。此时的脉冲宽度为低功耗波特率的 3 倍，最小可为 1.42 MHz。通常此值是 1.8432 MHz (1.42 MHz < PSC < 2.12 MHz)。低功耗模式下的可编程分频器会对系统时钟进行分频，以达到此值。
- 接收器

  在低功耗模式下接收与在正常模式下接收类似。为进行干扰检测，USART 必须丢弃持续时间短于 1/PSC 的脉冲。只有当持续时间大于 2 个 IrDA 低功耗波特时钟周期（USART_GTPR 寄存器中的 PSC 值）时，才是有效低电平。

注意： *宽度小于两个但大于一个 PSC 周期的脉冲可能被接受，也可能被拒绝。*

接收器的建立时间必须由软件进行管理。IrDA 物理层规范规定发送和接收之间至少要经过 10 ms 的延迟（IrDA 是一个半双工协议）。

图 443. IrDA SIR ENDEC 框图

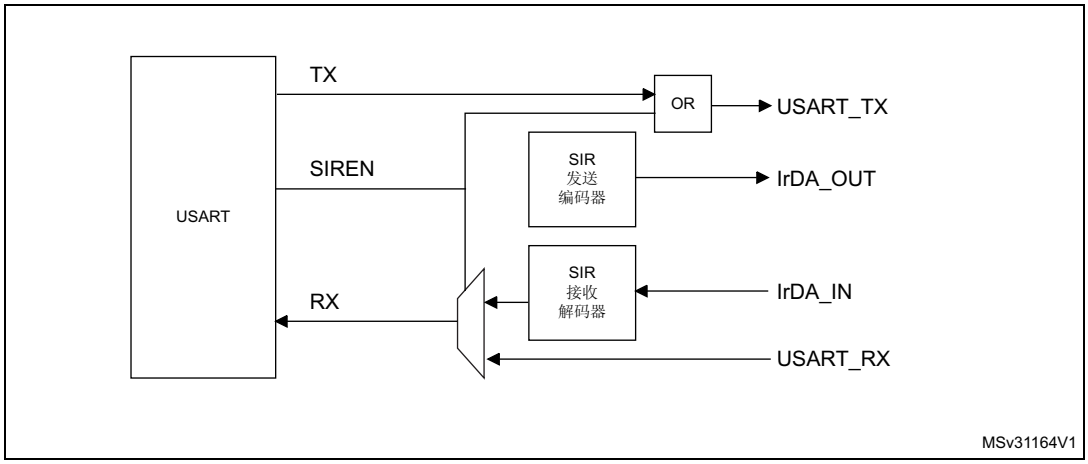
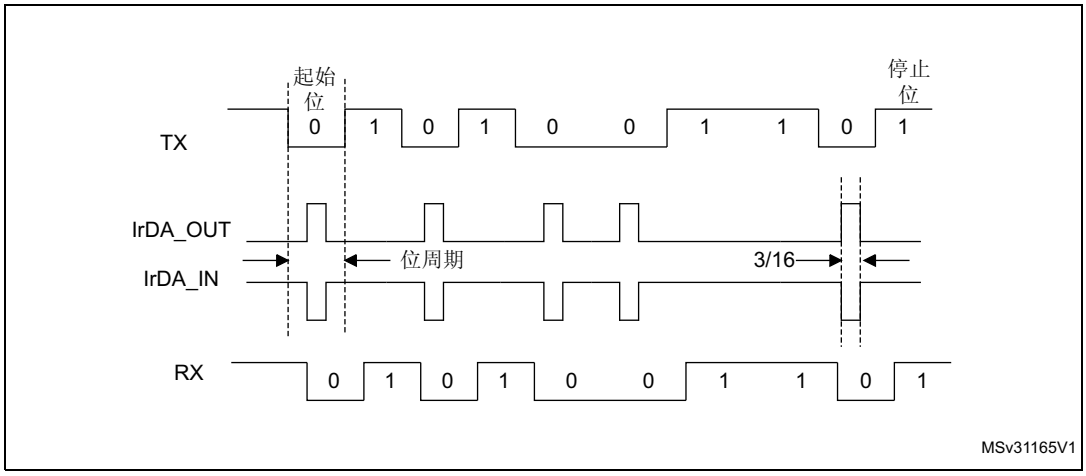


图 444. IrDA 数据调制 (3/16)——正常模式



36.5.20 使用 USART 和 DMA 进行连续通信

USART 能够使用 DMA 进行连续通信。接收缓冲区和发送缓冲区的 DMA 请求是独立的。

注意： *要确定是否支持 DMA 模式，请参见第 1253 页的第 36.4 节：USART 实现。如果不支持 DMA 模式，请按照第 36.5.7 节中的说明使用 USART。可以将 USART_ISR 寄存器中的 TXE/RXNE 标志清零，从而在禁止 FIFO 时实现连续通信。*

使用 DMA 进行发送

将 USART_CR3 寄存器中的 DMAT 位置 1 可以使能 DMA 模式进行发送。当 TXE 标志（使能 FIFO 模式时为 TXFNF 标志）置 1 时，可使用 DMA 外设（请参见*直接存储器访问控制器 (DMA) 部分*）将数据从配置的 SRAM 区域加载到 USART_TDR 寄存器。要映射一个 DMA 通道以进行 USART 发送，请按以下步骤操作（x 表示通道编号）：

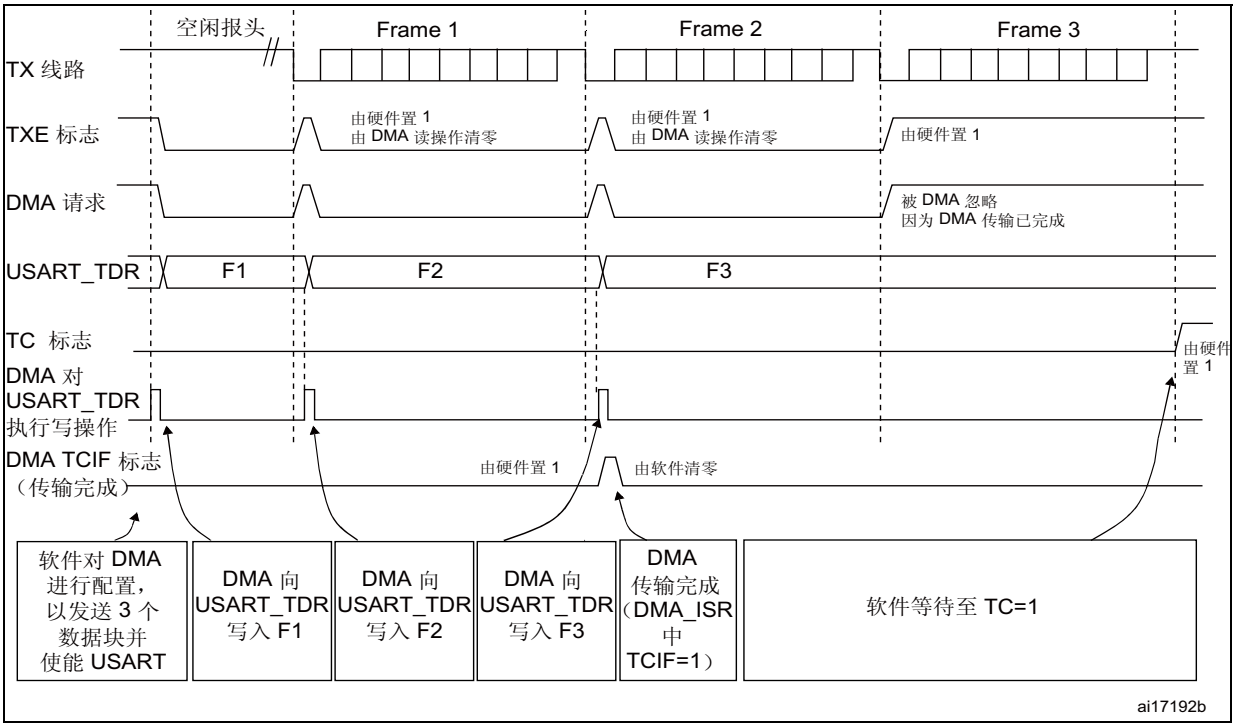
1. 在 DMA 控制寄存器中写入 USART_TDR 寄存器地址，将其配置为传输的目标地址。每次发生 TXE（使能 FIFO 模式时为 TXFNF）事件后，数据都从存储器移动到此地址。
2. 在 DMA 控制寄存器中写入存储器地址，将其配置为传输的源地址。每次发生 TXE（使能 FIFO 模式时为 TXFNF）事件后，数据都从该存储区域加载到 USART_TDR 寄存器中。
3. 在 DMA 控制寄存器中配置要传输的总字节数。
4. 在 DMA 寄存器中配置通道优先级
5. 根据应用的需求，在完成一半或全部传输后产生 DMA 中断。
6. 通过将 USART_ICR 寄存器中的 TCCF 位置 1，将 USART_ISR 寄存器中的 TC 标志清零。
7. 在 DMA 寄存器中激活该通道。

当达到在 DMA 控制器中设置的数据传输量时，DMA 控制器会在 DMA 通道的中断向量上产生一个中断。

在发送模式下，DMA 对所有要发送的数据执行了写操作（DMA_ISR 寄存器中的 TCIF 标志置 1）后，可以对 TC 标志进行监视，以确保 USART 通信已完成。在禁止 USART 之前或系统进入低功耗模式之前（禁止外设时钟时）必须执行此步骤，以避免损坏最后一次发送。软件必须等待直到 TC=1。TC 标志在所有数据发送期间都保持清零状态，然后在最后一帧发送结束时由硬件置 1。

注意： *在 DMA 传输结束之前不得将 DMAT 位清零。*

图 445. 使用 DMA 进行发送



注意： *使能 FIFO 管理时，DMA 请求由发送 FIFO 未满（即 TXFNF = 1）触发。*

使用 DMA 进行接收

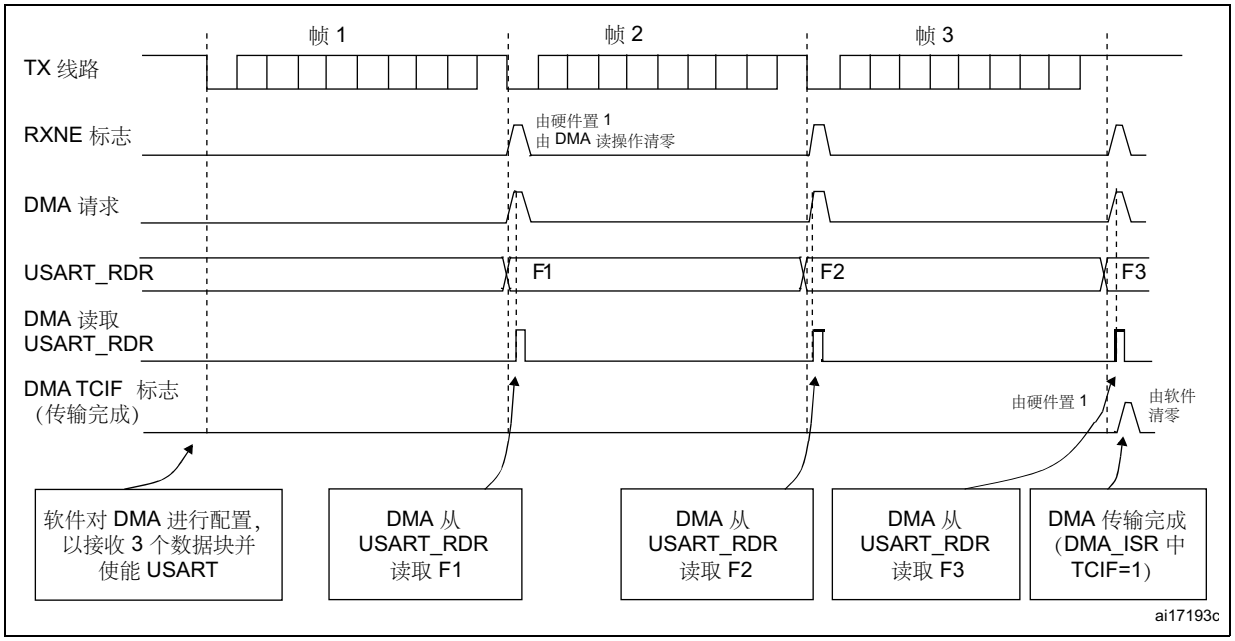
将 USART_CR3 寄存器中的 DMAR 位置 1 可以使能 DMA 模式进行接收。接收数据字节时，可使用 DMA 外设（请参见*直接存储器访问控制器 (DMA) 部分*）将数据从 USART_RDR 寄存器加载到配置的 SRAM 区域中。要映射一个 DMA 通道以进行 USART 接收，请按以下步骤操作：

1. 在 DMA 控制寄存器中写入 USART_RDR 寄存器地址，将其配置为传输的源地址。每次发生 RXNE（使能 FIFO 模式时为 RXFNE）事件后，数据都从该地址移动到存储器。
2. 在 DMA 控制寄存器中写入存储器地址，将其配置为传输的目标地址。每次发生 RXNE（使能 FIFO 模式时为 RXFNE）事件后，数据都从 USART_RDR 加载到此存储区域。
3. 在 DMA 控制寄存器中配置要传输的总字节数。
4. 在 DMA 控制寄存器中配置通道优先级。
5. 根据应用的需求，在完成一半或全部传输后产生中断。
6. 在 DMA 控制寄存器中激活该通道。

当达到在 DMA 控制器中设置的数据传输量时，DMA 控制器会在 DMA 通道的中断向量上产生一个中断。

注意： *在 DMA 传输结束之前不得将 DMAR 位清零。*

图 446. 使用 DMA 进行接收



注意： *使能 FIFO 管理时，DMA 请求由接收 FIFO 非空（即 RXFNE = 1）触发。*

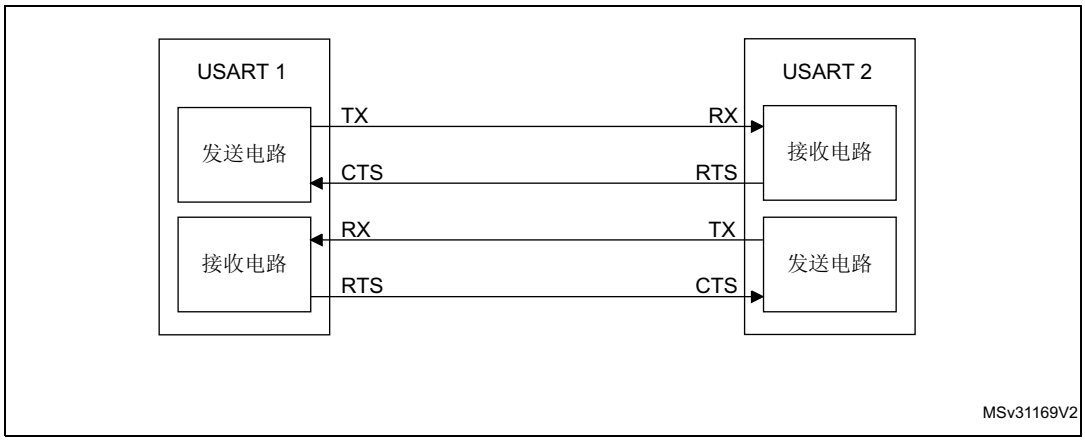
多缓冲区通信中的错误标志和中断生成

在多缓冲区通信模式下，如果事务中发生任何错误，都会在当前字节后将相应错误标志置 1。如果中断使能标志置 1，则会产生中断。在单字节接收过程中，与 RXNE（使能 FIFO 模式时为 RXFNE）一同置位的帧错误、上溢错误和噪声标志具有单独的错误标志中断使能位（USART_CR3 寄存器中的 EIE 位）；如果该位置 1，在发生其中任何一个错误时，都会在当前字节后使能中断。

36.5.21 RS232 硬件流控制和 RS485 驱动器使能

使用 CTS 输入和 RTS 输出可以控制 2 个器件间的串行数据流。*图 447* 显示了在这种模式下如何连接 2 个器件：

图 447. 2 个 USART 间的硬件流控制

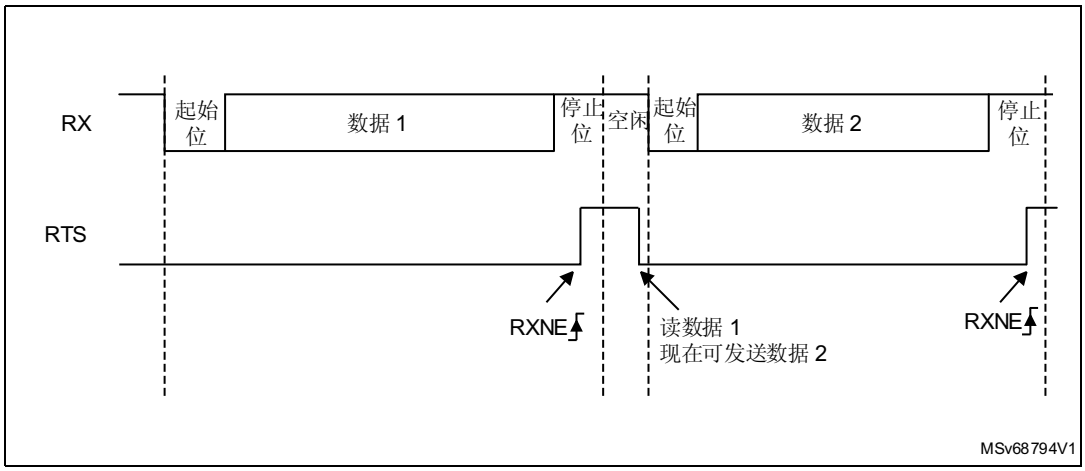


向 USART_CR3 寄存器中的 RTSE 位和 CTSE 位写入 1，可分别使能 RS232 RTS 和 CTS 流控制。

RS232 RTS 流控制

如果使能 RTS 流控制 (RTSE=1)，只要 USART 接收器准备好接收新数据，便会将 RTS 变为无效（连接到低电平）。当接收寄存器已满时，会将 RTS 变为有效，表明发送过程会在当前帧结束后停止。*图 448* 显示了在使能 RTS 流控制的情况下进行通信的示例。

图 448. RS232 RTS 流控制



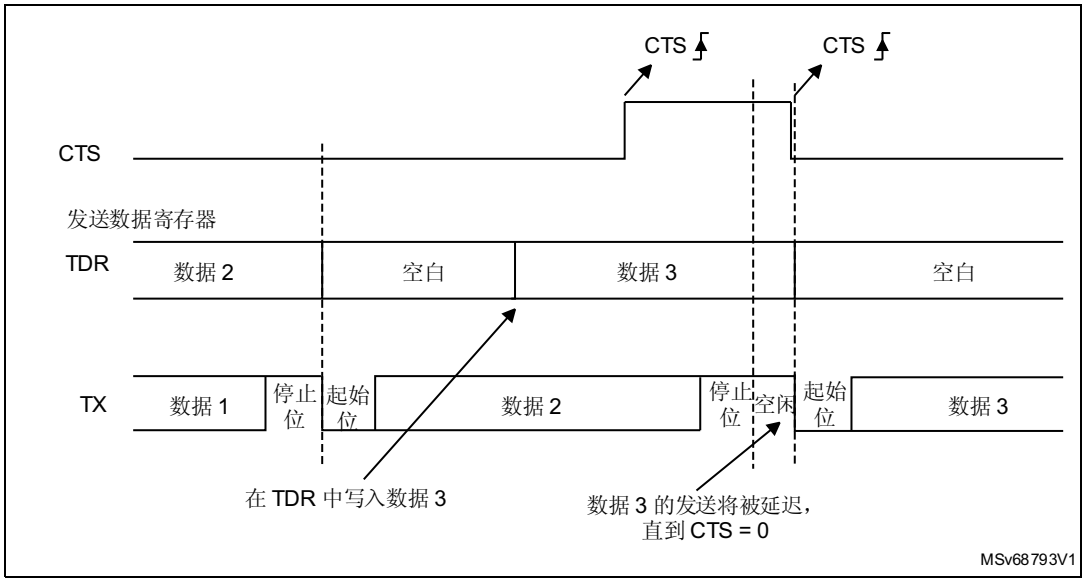
注意： *如果使能 FIFO 模式，则仅当 RXFIFO 已满时，RTS 才会变为有效。*

RS232 CTS 流控制

如果使能 CTS 流控制 (CTSE = 1)，则发送器会在发送下一帧前检查 CTS。如果 CTS 无效（连接到低电平），则会发送下一数据（假设数据已准备好发送，即 TXE/TXFE=0）；否则不会进行发送。如果在发送过程中 CTS 变为有效，则当前发送完成之后，发送器停止。

当 CTSE = 1 时，只要 CTS 发生变化，CTSIF 状态位便会由硬件自动置 1。这指示接收器是否已准备好进行通信。如果 USART_CR3 寄存器中的 CTSIE 位置 1，则会产生中断。*图 449* 显示了在使能 CTS 流控制的情况下进行通信的示例。

图 449. RS232 CTS 流控制



注意： *为正常运行，必须在当前字符结束前至少 3 个 USART 时钟源周期内将 CTS 拉低。此外还必须注意，当脉冲短于 2 个 PCLK 周期时无法将 CTSCF 标志置 1。*

RS485 驱动器使能

驱动器使能功能可通过将 USART_CR3 控制寄存器中的 DEM 位置 1 使能。这样用户便可通过 DE（驱动器使能）信号激活外部收发器控制。使能时间为激活 DE 信号与 START 位开始间的时间。可以通过 USART_CR1 控制寄存器中的 DEAT [4:0] 位域编程禁止时间。禁止时间为发送的消息中最后一个停止位结束与取消激活 DE 信号间的时间。可以通过 USART_CR1 控制寄存器中的 DEDT [4:0] 位域编程禁止时间。DE 信号的极性可使用 USART_CR3 控制寄存器中的 DEP 位配置。

在 USART 中，DEAT 和 DEDT 以采样时间单位表示（1/8 或 1/16 位时间，具体取决于过采样速率）。

36.5.22 USART 低功耗管理

USART 具有高级低功耗模式功能，即使禁止 usart_pclk 时钟，也能正常传输数据。

UESM 位置 1 时，USART 能够将 MCU 从低功耗模式唤醒。

当 usart_pclk 被关闭时，如果某些特定操作需要激活 **usart_pclk** 时钟，则 USART 会提供唤醒中断 (**usart_wkup**)：

- 禁止 FIFO 模式时

  必须激活 usart_pclk 时钟以清空 USART 数据寄存器。

  在这种情况下，usart_wkup 中断源为 RXNE 置 1。RXNEIE 位必须在进入低功耗模式之前置 1。

- 使能 FIFO 模式时

  必须激活 usart_pclk 时钟以：

  – 填充 TXFIFO

  – 或清空 RXFIFO

  在这种情况下，usart_wkup 中断源可以为：

  – RXFIFO 非空。此时，RXFNEIE 位必须在进入低功耗模式之前置 1。

  – RxFIFO 已满。此时，RXFFIE 位必须在进入低功耗模式之前置 1，接收到的数据量对应于 RXFIFO 大小，并且 RXFF 标志未置 1。

  – TXFIFO 为空。此时，TXFEIE 位必须在进入低功耗模式之前置 1。

  这允许在低功耗模式下发送/接收 TXFIFO/RXFIFO 中的数据。

  为了避免发生上溢/下溢错误以及在低功耗模式下发送/接收数据，usart_wkup 中断源可以是以下事件之一：

  – 达到 TXFIFO 阈值。此时，TXFTIE 位必须在进入低功耗模式之前置 1。

  – 达到 RXFIFO 阈值。此时，RXFTIE 位必须在进入低功耗模式之前置 1。

  例如，如果唤醒时间短于在线路上接收单个字节所需的时间，则应用可将阈值设为最大 RXFIFO 大小。

  使用 RXFIFO 已满、TXFIFO 为空、RXFIFO 非空和 RXFIFO/TXFIFO 阈值中断将 MCU 从低功耗模式唤醒时，可在低功耗模式下尽可能多地进行 USART 传输，这样有助于优化功耗。

或者，也可通过 WUS 位域选择特定 **usart_wkup** 中断。

检测到唤醒事件后，WUF 标志会由硬件置 1 并在 WUFIE 位置 1 时生成一个 **usart_wkup** 中断。

注意： *在进入低功耗模式之前，请确保未进行任何 USART 传输。检查 BUSY 标志不能确保在进行数据接收时绝不会进入低功耗模式。*

WUF 标志在检测到唤醒事件时置 1，而与 MCU 处于低功耗模式还是工作模式无关。

如果在初始化和使能接收器后立即进入低功耗模式，则必须检查 REACK 位以确保 USART 确实已使能。

当 DMA 用于接收时，它必须在进入低功耗模式前禁止，并在退出低功耗模式后重新使能。

如果使能 FIFO，则只有在使能静默模式的情况下才能在地址匹配时从低功耗模式唤醒。

使用静默模式和低功耗模式

如果 USART 在进入低功耗模式前处于静默模式：

- 不得使用空闲检测来从静默模式唤醒，因为空闲检测无法在低功耗模式下工作。
- 如果使用地址匹配来从静默模式唤醒，则低功耗模式唤醒源也必须是地址匹配。如果 RXNE 标志在进入低功耗模式时置 1，则接口将在地址匹配时和从低功耗模式唤醒时保持静默模式。

注意： *使能 FIFO 管理时，静默模式可与从低功耗模式唤醒搭配使用，而且没有任何限制（即，上述关于静默和低功耗模式的两点仅在 FIFO 管理禁止时有效）。*

USART 内核时钟 (usart_ker_ck) 在低功耗模式下关闭时从低功耗模式唤醒

在低功耗模式下，如果在 USART 接收线上检测到下降沿时 usart_ker_ck 时钟处于关闭状态，则 USART 接口会借助 usart_ker_ck_req 信号请求开启 usart_ker_ck 时钟。usart_ker_ck 随后用来接收帧。

如果唤醒事件通过验证，则 MCU 将从低功耗模式唤醒，并会正常进行数据接收。

如果唤醒事件未通过验证，则 usart_ker_ck 会再次关闭，MCU 不会被唤醒并保持在低功耗模式下，且内核时钟请求被释放。

以下示例显示了将唤醒事件编程为“地址匹配检测”并禁止 FIFO 管理的情况。

图 450 所示为唤醒事件通过验证时的 USART 行为。

图 450. 唤醒事件通过验证（唤醒事件 = 地址匹配，禁止 FIFO）

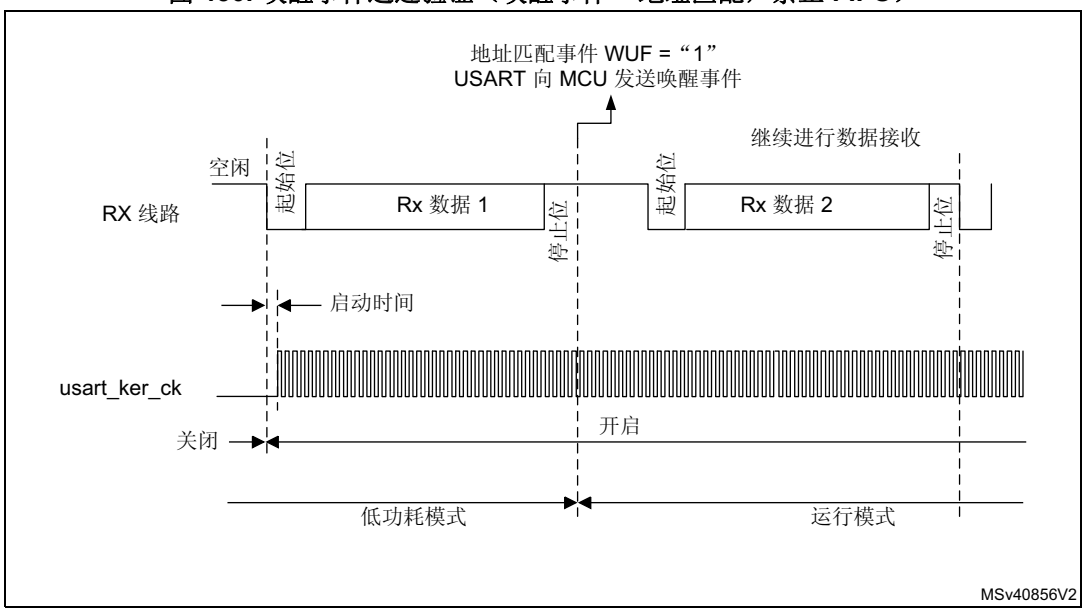
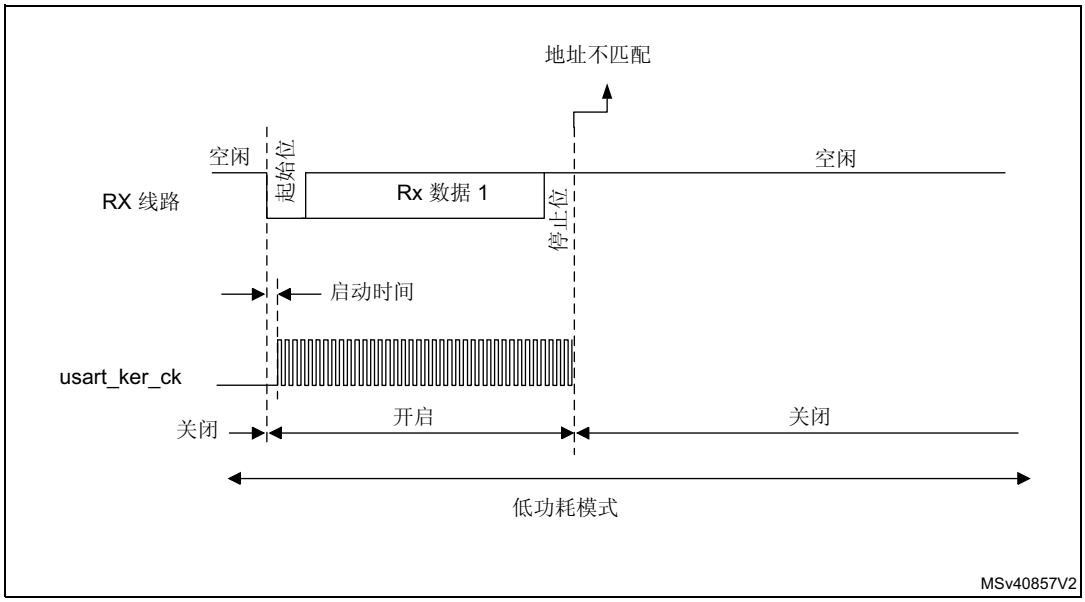


图 451 所示为唤醒事件未通过验证时的 USART 行为。

图 451. 唤醒事件未通过验证（唤醒事件 = 地址匹配，禁止 FIFO）



注意： *将地址匹配或任一接收的帧用作唤醒事件时，如上两图适用。如果唤醒事件为起始位检测，则 USART 会在起始位结束时向 MCU 发送唤醒事件。*

确定允许从低功耗模式正确唤醒微控制器的最大 USART 波特率

允许从低功耗模式正确唤醒微控制器的最大波特率取决于唤醒时间参数（请参见器件数据手册）和 USART 接收器容差（请参见*第 36.5.9 节：USART 接收器对时钟偏差的容差*）。

举例来说：OVER8 = 0，M 位 = 01，ONEBIT = 0，BRR [3:0] = 0000。

在这些条件下，根据*表 288：BRR [3:0] = 0000 时的 USART 接收器容差*，USART 接收器的容差为 3.41%。

DTRA + DQUANT + DREC + DTCL + DWU < USART 接收器的容差

$D_{WUmax} = t_{WUUSART} / (11 \times T_{bitmin})$

$T_{bitmin} = t_{WUUSART} / (11 \times D_{WUmax})$

其中 $t_{WUUSART}$ 为从低功耗模式唤醒的时间。

考虑一种理想情况：参数 DTRA、DQUANT、DREC 和 DTCL 为 0%，则 DWU 的最大值为 3.41%。实际上，我们至少需要考虑 usart_ker_ck 不精确 (DREC) 的情况。

例如，如果将 HSI 用作 usart_ker_ck，HSI 的不精确度为 1%，则可以得到：

$t_{WUUSART}$ = 3 µs（仅提供数值作为参考；有关正确数值，请参见器件数据手册）。

D_{WUmax} = USART 接收器容差 – DREC = 3.41% - 1% = 2.41%

T_{bitmin} = 3 µs/ (11 x 2.41%) = 11.32 µs。

因此，允许从低功耗模式正确唤醒的最大波特率为：1/11.32 µs = 88.36 kbaud。

36.6 低功耗模式下的 USART

表 291. 低功耗模式对 USART 的影响

模式	说明
睡眠	无影响。USART 中断可使器件退出睡眠模式。
停止(1)	USART 寄存器的内容会保留。 当 USART 的时钟由停止模式下可用的振荡器提供时，USART 能够将微控制器从停止模式唤醒。
待机	USART 外设掉电，退出待机模式后必须重新初始化。

1. 请参见*第 36.4 节：USART 实现*以了解给定外设实例是否支持从停止模式唤醒。如果某个实例在给定的停止模式下不起作用，则必须在进入此停止模式之前将其禁止。

36.7 USART 中断

有关所有 USART 中断请求的详细说明，请参见*表 292*。

表 292. USART 中断请求

中断向量	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止(1)模式	退出待机模式
USART 或 UART	发送数据寄存器为空	TXE	TXEIE	写入 TDR	是	否	否
	发送 FIFO 未满	TXFNF	TXFNFIE	TXFIFO 已满		否	
	发送 FIFO 为空	TXFE	TXFEIE	写入 TDR 或向 TXFRQ 中写入 1		是	
	达到发送 FIFO 阈值	TXFT	TXFTIE	写入 TDR		是	
	CTS 中断	CTSIF	CTSIE	向 CTSCF 中写入 1		否	
	发送完成	TC	TCIE	写入 TDR 或向 TCCF 中写入 1		否	
	保护时间前发送完成	TCBGT	TCBGTIE	写入 TDR 或向 TCBGT 中写入 1		否	

表 292. USART 中断请求（续）

中断向量	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止(1)模式	退出待机模式
USART 或 UART	接收数据寄存器不为空（已准备好读取数据）	RXNE	RXNEIE	读取 RDR 或向 RXFRQ 中写入 1	是	是	否
	接收 FIFO 非空	RXFNE	RXFNEIE	读取 RDR 直到 RXFIFO 为空或向 RXFRQ 中写入 1		是	
	接收 FIFO 已满	RXFF(2)	RXFFIE	读取 RDR		是	
	达到接收 FIFO 阈值	RXFT	RXFTIE	读取 RDR		是	
	检测到上溢错误	ORE	RXNEIE/ RXFNEIE	向 ORECF 中写入 1		否	
	检测到空闲线路	IDLE	IDLEIE	向 IDLECF 中写入 1		否	
	奇偶校验错误	PE	PEIE	向 PECF 中写入 1		否	
	LIN 刹车	LBDF	LBDIE	向 LBDCF 中写入 1		否	
	多缓冲区通信中的噪声错误。	NE	EIE	向 NFCF 中写入 1		否	
	多缓冲区通信中的上溢错误。	ORE(3)		向 ORECF 中写入 1		否	
	多缓冲区通信中的帧错误。	FE		向 FECF 中写入 1		否	
	字符匹配	CMF	CMIE	向 CMCF 中写入 1		否	
	接收器超时	RTOF	RTOFIE	向 RTOCCF 中写入 1		否	
	块结束	EOBF	EOBIE	向 EOBCF 中写入 1		否	
	从低功耗模式唤醒	WUF	WUFIE	向 WUC 中写入 1		是	
	SPI 从设备下溢错误	UDR	EIE	向 UDRCF 中写入 1		否	

1. 仅当外设实例支持从停止模式唤醒功能时，USART 才能将器件从停止模式唤醒。关于所支持的停止模式的列表，请参见*第 36.4 节：USART 实现*。
2. 如果 USART 接收 n+1 个数据（n 表示 RXFIFO 大小，RXFIFO 中有 n 个数据，USART_RDR 中有 1 个数据），则 RXFF 标志置 1。在停止模式下，未向 USART_RDR 提供时钟。因此，不会对该寄存器进行写操作，一旦有 n 个数据被接收并写入到 RXFIFO，RXFF 中断便会生效（RXFF 标志未置 1）。
3. 当 OVRDIS = 0 时。

36.8 USART 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节*。

外设寄存器必须按字（32 位）进行访问。

36.8.1 USART 控制寄存器 1 (USART_CR1)

USART control register 1

偏移地址：0x00

复位值：0x0000 0000

同一寄存器可用于使能 FIFO 模式（本节）和禁止 FIFO 模式（下一节）的情况。

使能 FIFO 模式

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXFFIE	TXFEIE	FIFOEN	M1	EOBIE	RTOIE	DEAT[4:0]					DEDT[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVER8	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXFNFIE	TCIE	RXFNEIE	IDLEIE	TE	RE	UESM	UE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **RXFFIE**：RXFIFO 变满时中断使能 (RXFIFO Full interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 USART_ISR 寄存器中的 RXFF=1 时，生成 USART 中断

位 30 **TXFEIE**：TXFIFO 为空时中断使能 (TXFIFO empty interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 USART_ISR 寄存器中的 TXFE=1 时，生成 USART 中断

位 29 **FIFOEN**：FIFO 模式使能 (FIFO mode enable)

该位由软件置 1 和清零。

0：禁止 FIFO 模式。

1：使能 FIFO 模式。

只有在禁止 USART (UE=0) 时才能写入该位域。

注意：FIFO 模式只能在标准 UART 通信、SPI 主/从模式和智能卡模式下使用，不得在 IrDA 和 LIN 模式下使能。

位 28 **M1**：字长 (Word length)

该位必须与位 12 (M0) 搭配使用来确定字长度。该位由软件置 1 或清零。

M[1:0] = 00：1 个起始位，8 个数据位，n 个停止位

M[1:0] = 01：1 个起始位，9 个数据位，n 个停止位

M[1:0] = 10：1 个起始位，7 个数据位，n 个停止位

只有在禁止 USART (UE=0) 时才能写入该位。

注意：在 7 位数据长度模式下，不支持智能卡模式、LIN 主模式和自动波特率（0x7F 帧和 0x55 帧检测）。

位 27 **EOBIE**：块结束中断使能 (End of Block interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：USART_ISR 寄存器中的 EOBF 标志置 1 时生成 USART 中断

注意：如果 USART 不支持智能卡模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 26 **RTOIE**：接收器超时中断使能 (Receiver timeout interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：USART_ISR 寄存器中的 RTOF 位置 1 时生成 USART 中断。

注意：如果 USART 不支持接收器超时功能，该位保留且必须保持复位值。第 1253 页的第 36.4 节：USART 实现。

位 25:21 **DEAT[4:0]**：驱动器使能有效时间 (Driver Enable assertion time)

该 5 位值用于定义激活 DE（启动器使能）信号与起始位开始间的时间。此时间以采样时间单位表示（1/8 或 1/16 位时间，具体取决于过采样速率）。

只有在禁止 USART (UE=0) 时才能写入该位域。

注意：如果不支持驱动器使能功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 20:16 **DEDT[4:0]**：驱动器使能无效时间 (Driver Enable deassertion time)

该 5 位值用于定义发送的消息中最后一个停止位结束与取消激活 DE（驱动器使能）信号间的时间。此时间以采样时间单位表示（1/8 或 1/16 位时间，具体取决于过采样速率）。

如果在 DEDT 时间内对 USART_TDR 寄存器执行写操作，则新数据仅在经过 DEDT 和 DEAT 时间后才会发送。

只有在禁止 USART (UE=0) 时才能写入该位域。

注意：如果不支持驱动器使能功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 15 **OVER8**：过采样模式 (Oversampling mode)

0：16 倍过采样

1：8 倍过采样

只有在禁止 USART (UE=0) 时才能写入该位。

注意：在 LIN、IrDA 和智能卡模式下，该位必须保持清零。

位 14 **CMIE**：字符匹配中断使能 (Character match interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：USART_ISR 寄存器中的 CMF 位置 1 时生成 USART 中断。

位 13 **MME**：静默模式使能 (Mute mode enable)

该位可使能 USART 静默模式功能。该位置 1 时，USART 会按照 WAKE 位的定义在工作模式与静默模式之间切换。该位由软件置 1 和清零。

0：接收器永久处于工作模式

1：接收器可在静默模式和工作模式之间切换。

位 12 **M0**：字长 (Word length)

该位与位 28 (M1) 搭配使用来确定字长度。它由软件置 1 或清零（请参见位 28 (M1) 的说明）。

只有在禁止 USART (UE=0) 时才能写入该位。

位 11 **WAKE**：接收器唤醒方法 (Receiver wake-up method)

该位用于确定 USART 静默模式的唤醒方法。该位由软件置 1 或清零。

0：空闲线路

1：地址标记

只有在禁止 USART (UE=0) 时才能写入该位域。

位 10 **PCE**：奇偶校验控制使能 (Parity control enable)

该位选择硬件奇偶校验控制（生成和检测）。使能奇偶校验控制时，计算出的奇偶校验位被插入到 MSB 位置（如果 M=1，则为第 9 位；如果 M=0，则为第 8 位），并对接收到的数据检查奇偶校验位。该位由软件置 1 和清零。一旦该位置 1，PCE 在当前字节的后面处于活动状态（在接收和发送时）。

0：禁止奇偶校验控制

1：使能奇偶校验控制

只有在禁止 USART (UE=0) 时才能写入该位域。

位 9 **PS**：奇偶校验选择 (Parity selection)

该位用于在使能奇偶校验生成/检测（PCE 位置 1）时选择奇校验或偶校验。该位由软件置 1 和清零。在当前字节的后面选择奇偶校验。

0：偶校验

1：奇校验

只有在禁止 USART (UE=0) 时才能写入该位域。

位 8 **PEIE**：PE 中断使能 (PE interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 USART_ISR 寄存器中的 PE=1 时，生成 USART 中断

位 7 **TXFNFIE**：TXFIFO 未满中断使能 (TXFIFO not full interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 USART_ISR 寄存器中的 TXFNF = 1 时，生成 USART 中断

位 6 **TCIE**：传送完成中断使能 (Transmission complete interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 USART_ISR 寄存器中的 TC=1 时，生成 USART 中断

位 5 **RXFNEIE**：RXFIFO 非空中断使能 (RXFIFO not empty interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 USART_ISR 寄存器中的 ORE=1 或 RXFNE=1 时，生成 USART 中断

位 4 **IDLEIE**：IDLE 中断使能 (IDLE interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 USART_ISR 寄存器中的 IDLE=1 时，生成 USART 中断

位 3 **TE**：发送器使能 (Transmitter enable)

该位使能发送器。该位由软件置 1 和清零。

0：禁止发送器

1：使能发送器

注意：除了在智能卡模式下以外，传送期间 TE 位上的低电平脉冲（0 后紧跟的是 1）会在当前字的后面发送一个报头（空闲线路）。为生成空闲字符，TE 不能立即写入 1。为确保所需的持续时间，软件可轮询 USART_ISR 寄存器中的 TEACK 位。

在智能卡模式下，当 TE 置 1 时，在发送开始前存在 1 位的时间延迟。

位 2 **RE**：接收器使能 (Receiver enable)

该位使能接收器。该位由软件置 1 和清零。

0：禁止接收器

1：使能接收器并开始搜索起始位

位 1 **UESM**：低功耗模式下的 USART 使能 (USART enable in low-power mode)

当该位清零时，USART 无法将 MCU 从低功耗模式唤醒。

当该位置 1 时，USART 可将 MCU 从低功耗模式唤醒。

该位由软件置 1 和清零。

0：USART 无法将 MCU 从低功耗模式唤醒。

1：USART 能够将 MCU 从低功耗模式唤醒。

注意：建议在进入低功耗模式前将 UESM 位置 1，并在退出低功耗模式时将其清零。

位 0 **UE**：USART 使能 (USART enable)

该位清零后，USART 预分频器和输出将立即停止，并丢弃所有当前操作。USART 配置会保留，而所有的 USART_ISR 状态标志均会复位。该位由软件置 1 和清零。

0：禁止 USART 预分频器和输出，低功耗模式

1：使能 USART

注意：为进入低功耗模式而不在线路上生成错误，之前必须复位 TE 位，并且软件必须等待 USART_ISR 中的 TC 位置 1 后才能复位 UE 位。

UE = 0 时也会复位 DMA 请求，因此必须在复位 UE 位前禁止 DMA 通道。

在智能卡模式下 (SCEN = 1)，无论 UE 位值为何，当 CLKEN = 1 时，CK 始终可用。

36.8.2 USART 控制寄存器 1 [备用] (USART_CR1)

USART control register 1 [alternate]

偏移地址：0x00

复位值：0x0000 0000

同一寄存器可用于使能 FIFO 模式（上一节）和禁止 FIFO 模式（本节）的情况。

禁止 FIFO 模式

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	FIFO EN	M1	EOBIE	RTOIE	DEAT[4:0]					DEDT[4:0]				
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVER8	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:30 保留，必须保持复位值。

位 29 **FIFOEN**：FIFO 模式使能 (FIFO mode enable)

该位由软件置 1 和清零。

0：禁止 FIFO 模式。

1：使能 FIFO 模式。

只有在禁止 USART (UE=0) 时才能写入该位域。

注意：FIFO 模式只能在标准 UART 通信、SPI 主/从模式和智能卡模式下使用，不得在 IrDA 和 LIN 模式下使能。

位 28 **M1**：字长 (Word length)

该位必须与位 12 (M0) 搭配使用来确定字长度。该位由软件置 1 或清零。

M[1:0] = 00：1 个起始位，8 个数据位，n 个停止位

M[1:0] = 01：1 个起始位，9 个数据位，n 个停止位

M[1:0] = 10：1 个起始位，7 个数据位，n 个停止位

只有在禁止 USART (UE=0) 时才能写入该位。

注意：在 7 位数据长度模式下，不支持智能卡模式、LIN 主模式和自动波特率（0x7F 帧和 0x55 帧检测）。

位 27 **EOBIE**：块结束中断使能 (End of Block interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：USART_ISR 寄存器中的 EOBF 标志置 1 时生成 USART 中断

注意：如果 USART 不支持智能卡模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 26 **RTOIE**：接收器超时中断使能 (Receiver timeout interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：USART_ISR 寄存器中的 RTOF 位置 1 时生成 USART 中断。

注意：如果 USART 不支持接收器超时功能，该位保留且必须保持复位值。第 1253 页的第 36.4 节：USART 实现。

位 25:21 **DEAT[4:0]**：驱动器使能有效时间 (Driver Enable assertion time)

该 5 位值用于定义激活 DE（启动器使能）信号与起始位开始间的时间。此时间以采样时间单位表示（1/8 或 1/16 位时间，具体取决于过采样速率）。

只有在禁止 USART (UE=0) 时才能写入该位域。

注意：如果不支持驱动器使能功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 20:16 **DEDT[4:0]**：驱动器使能无效时间 (Driver Enable deassertion time)

该 5 位值用于定义发送的消息中最后一个停止位结束与取消激活 DE（驱动器使能）信号间的时间。此时间以采样时间单位表示（1/8 或 1/16 位时间，具体取决于过采样速率）。

如果在 DEDT 时间内对 USART_TDR 寄存器执行写操作，则新数据仅在经过 DEDT 和 DEAT 时间后才会发送。

只有在禁止 USART (UE=0) 时才能写入该位域。

注意：如果不支持驱动器使能功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 15 **OVER8**：过采样模式 (Oversampling mode)

0：16 倍过采样

1：8 倍过采样

只有在禁止 USART (UE=0) 时才能写入该位。

注意：在 LIN、IrDA 和智能卡模式下，该位必须保持清零。

位 14 **CMIE**：字符匹配中断使能 (Character match interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：USART_ISR 寄存器中的 CMF 位置 1 时生成 USART 中断。

位 13 **MME**：静默模式使能 (Mute mode enable)

该位可使能 USART 静默模式功能。该位置 1 时，USART 会按照 WAKE 位的定义在工作模式与静默模式之间切换。该位由软件置 1 和清零。

0：接收器永久处于工作模式

1：接收器可在静默模式和工作模式之间切换。

位 12 **M0**：字长 (Word length)

该位与位 28 (M1) 搭配使用来确定字长度。它由软件置 1 或清零（请参见位 28 (M1) 的说明）。
只有在禁止 USART (UE=0) 时才能写入该位。

位 11 **WAKE**：接收器唤醒方法 (Receiver wake-up method)

该位用于确定 USART 静默模式的唤醒方法。该位由软件置 1 或清零。
0：空闲线路
1：地址标记
只有在禁止 USART (UE=0) 时才能写入该位域。

位 10 **PCE**：奇偶校验控制使能 (Parity control enable)

该位选择硬件奇偶校验控制（生成和检测）。使能奇偶校验控制时，计算出的奇偶校验位被插入到 MSB 位置（如果 M=1，则为第 9 位；如果 M=0，则为第 8 位），并对接收到的数据检查奇偶校验位。该位由软件置 1 和清零。一旦该位置 1，PCE 在当前字节的后面处于活动状态（在接收和发送时）。
0：禁止奇偶校验控制
1：使能奇偶校验控制
只有在禁止 USART (UE=0) 时才能写入该位域。

位 9 **PS**：奇偶校验选择 (Parity selection)

该位用于在使能奇偶校验生成/检测（PCE 位置 1）时选择奇校验或偶校验。该位由软件置 1 和清零。在当前字节的后面选择奇偶校验。
0：偶校验
1：奇校验
只有在禁止 USART (UE=0) 时才能写入该位域。

位 8 **PEIE**：PE 中断使能 (PE interrupt enable)

该位由软件置 1 和清零。
0：禁止中断
1：当 USART_ISR 寄存器中的 PE=1 时，生成 USART 中断

位 7 **TXEIE**：发送数据寄存器为空 (Transmit data register empty)

该位由软件置 1 和清零。
0：禁止中断
1：当 USART_ISR 寄存器中的 TXE = 1 时，生成 USART 中断

位 6 **TCIE**：传送完成中断使能 (Transmission complete interrupt enable)

该位由软件置 1 和清零。
0：禁止中断
1：当 USART_ISR 寄存器中的 TC=1 时，生成 USART 中断

位 5 **RXNEIE**：接收数据寄存器非空 (Receive data register not empty)

该位由软件置 1 和清零。
0：禁止中断
1：当 USART_ISR 寄存器中的 ORE=1 或 RXNE=1 时，生成 USART 中断

位 4 **IDLEIE**：IDLE 中断使能 (IDLE interrupt enable)

该位由软件置 1 和清零。
0：禁止中断
1：当 USART_ISR 寄存器中的 IDLE=1 时，生成 USART 中断

位 3 **TE**：发送器使能 (Transmitter enable)

该位使能发送器。该位由软件置 1 和清零。

0：禁止发送器

1：使能发送器

注意：除了在智能卡模式下以外，传送期间 TE 位上的低电平脉冲（0 后紧跟的是 1）会在当前字的后面发送一个报头（空闲线路）。为生成空闲字符，TE 不能立即写入 1。为确保所需的持续时间，软件可轮询 USART_ISR 寄存器中的 TEACK 位。

在智能卡模式下，当 TE 置 1 时，在发送开始前存在 1 位的时间延迟。

位 2 **RE**：接收器使能 (Receiver enable)

该位使能接收器。该位由软件置 1 和清零。

0：禁止接收器

1：使能接收器并开始搜索起始位

位 1 **UESM**：低功耗模式下的 USART 使能 (USART enable in low-power mode)

当该位清零时，USART 无法将 MCU 从低功耗模式唤醒。

当该位置 1 时，USART 可将 MCU 从低功耗模式唤醒。

该位由软件置 1 和清零。

0：USART 无法将 MCU 从低功耗模式唤醒。

1：USART 能够将 MCU 从低功耗模式唤醒。

注意：建议在进入低功耗模式前将 UESM 位置 1，并在退出低功耗模式时将其清零。

位 0 **UE**：USART 使能 (USART enable)

该位清零后，USART 预分频器和输出将立即停止，并丢弃所有当前操作。USART 配置会保留，而所有的 USART_ISR 状态标志均会复位。该位由软件置 1 和清零。

0：禁止 USART 预分频器和输出，低功耗模式

1：使能 USART

注意：为进入低功耗模式而不在线路上生成错误，之前必须复位 TE 位，并且软件必须等待 USART_ISR 中的 TC 位置 1 后才能复位 UE 位。

UE = 0 时也会复位 DMA 请求，因此必须在复位 UE 位前禁止 DMA 通道。

在智能卡模式下 (SCEN = 1)，无论 UE 位值为何，当 CLKEN = 1 时，CK 始终可用。

36.8.3 USART 控制寄存器 2 (USART_CR2)

USART control register 2

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD[7:0]								RTOEN	ABRMOD[1:0]		ABREN	MSBFIRST	DATAINV	TXINV	RXINV
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWAP	LINEN	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	Res.	LBDIE	LBDL	ADDM7	DIS_NSS	Res.	Res.	SLVEN
rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw			rw

位 31:24 **ADD[7:0]**：USART 节点的地址 (Address of the USART node)

这些位用于指定静默模式下 USART 节点的地址或低功耗模式/运行模式下要识别的字符代码。

- 在静默模式下：它们用于在多处理器通信时通过 4 位/7 位地址标记检测从静默模式唤醒。发送器发送字符的 MSB 应为 1。在 4 位地址标记检测中，只使用 ADD[3:0] 位。
- 在低功耗模式下：它们用于在字符匹配时从低功耗模式唤醒。
  如果将 WUS[1:0] 编程为 0b00（WUF 在地址匹配时激活），并且已通过将 WUFIE 位置 1 使能 WUF 中断，则当收到的字符与通过 ADD[6:0] 或 ADD[3:0] 位域（取决于 ADDM7 位）编程的字符匹配时会从低功耗模式唤醒。发送器发送字符的 MSB 应为 1。
- 在静默模式无效时的运行模式下（例如，ModBus 协议中的块结束检测）：接收到的整个字符（8 位）将与 ADD[7:0] 值进行比较，如果匹配，CMF 标志将置 1。如果 CMIE 位置 1，则会生成中断。

仅在禁止接收 (RE = 0) 或禁止 USART (UE = 0) 时才能写入这些位。

位 23 **RTOEN**：接收器超时使能 (Receiver timeout enable)

该位由软件置 1 和清零。

0：禁止接收器超时功能。

1：使能接收器超时功能。

使能此功能后，如果 RX 线路在 RTOR（接收器超时寄存器）中编程的持续时间内处于空闲状态（无接收），则 USART_ISR 寄存器中的 RTOF 标志置 1。

注意：如果 USART 不支持接收器超时功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 22:21 **ABRMOD[1:0]**：自动波特率模式 (Auto baud rate mode)

这些位将由软件置 1 和清零。

00：通过测量起始位检测波特率。

01：下降沿到下降沿的测量（接收到的帧必须以一个等于 1 的位开头，即帧 = 10xxxxxx）

10：0x7F 帧检测。

11：0x55 帧检测。

仅在 ABREN = 0 时或禁止 USART (UE=0) 时才能写入该位域。

注意：如果 DATAINV=1 且/或 MSBFIRST=1，这些模式必须与在线路上时相同，例如 MSBFIRST 的 0xAA

如果 USART 不支持自动波特率功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 20 **ABREN**：自动波特率检测使能 (Auto baud rate enable)

该位由软件置 1 和清零。

0：禁止自动波特率检测。

1：使能自动波特率检测。

注意：如果 USART 不支持自动波特率功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 19 **MSBFIRST**：最高有效位在前 (Most significant bit first)

该位由软件置 1 和清零。

0：发送/接收数据时位 0 在前，后跟起始位。

1：发送/接收数据时 MSB（位 7/8）在前，后跟起始位。

只有在禁止 USART (UE=0) 时才能写入该位域。

位 18 **DATAINV**：二进制数据反向 (Binary data inversion)

该位由软件置 1 和清零。

0：按正/正向逻辑发送/接收数据寄存器中的逻辑数据。（1=H，0=L）

1：按负/反向逻辑发送/接收数据寄存器中的逻辑数据。（1=L，0=H）。奇偶校验位也取反。

只有在禁止 USART (UE=0) 时才能写入该位域。

位 17 **TXINV**：TX 引脚有效电平反向 (TX pin active level inversion)
该位由软件置 1 和清零。
0：TX 引脚信号使用标准逻辑电平（V_{DD} = 1/空闲，Gnd = 0/标记）工作。
1：对 TX 引脚信号值取反。（V_{DD} = 0/标记，Gnd=1/空闲）。
允许在 TX 线路上使用外部反相器。
只有在禁止 USART (UE=0) 时才能写入该位域。

位 16 **RXINV**：RX 引脚有效电平反向 (RX pin active level inversion)
该位由软件置 1 和清零。
0：RX 引脚信号使用标准逻辑电平（V_{DD} = 1/空闲，Gnd = 0/标记）工作。
1：对 RX 引脚信号值取反。（V_{DD} = 0/标记，Gnd=1/空闲）。
允许在 RX 线路上使用外部反相器。
只有在禁止 USART (UE=0) 时才能写入该位域。

位 15 **SWAP**：交换 TX/RX 引脚 (Swap TX/RX pins)
该位由软件置 1 和清零。
0：按标准引脚排列定义使用 TX/RX 引脚
1：交换 TX 和 RX 引脚功能。允许在与另一个 UART 的交叉连接时工作。
只有在禁止 USART (UE=0) 时才能写入该位域。

位 14 **LINEN**：LIN 模式使能 (LIN mode enable)
该位由软件置 1 和清零。
0：禁止 LIN 模式
1：使能 LIN 模式
LIN 模式可以使用 USART_CR1 寄存器中的 SBKRQ 位发送 LIN 同步刹车（13 个低位），并可检测 LIN 同步刹车。
只有在禁止 USART (UE=0) 时才能写入该位域。
注意：如果 USART 不支持 LIN 模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 13:12 **STOP[1:0]**：停止位 (STOP bits)
这些位用于编程停止位。
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
只有在禁止 USART (UE=0) 时才能写入该位域。

位 11 **CLKEN**：时钟使能 (Clock enable)
该位允许用户使能 CK 引脚。
0：禁止 CK 引脚
1：使能 CK 引脚
只有在禁止 USART (UE=0) 时才能写入该位。
注意：如果既不支持同步模式，也不支持智能卡模式，则该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。
在智能卡模式下，为了向智能卡正确提供 CK 时钟，必须按以下步骤操作：
UE = 0
SCEN = 1
GTPR 配置
CLKEN= 1
UE = 1

位 10 **CPOL**：时钟极性 (Clock polarity)

该位允许用户在同步模式下选择 CK 引脚上时钟输出的极性。它与 CPHA 位结合使用可获得所需的时钟/数据关系

0：空闲时 CK 引脚为低电平

1：空闲时 CK 引脚为高电平

只有在禁止 USART (UE=0) 时才能写入该位。

注意：如果不支持同步模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 9 **CPHA**：时钟相位 (Clock phase)

该位用于在同步模式下选择 CK 引脚上时钟输出的相位。它与 CPOL 位结合使用可获得所需的时钟/数据关系（请参见*图 431* 和*图 432*）

0：从第一个时钟边沿开始采样数据

1：从第二个时钟边沿开始采样数据

只有在禁止 USART (UE=0) 时才能写入该位。

注意：如果不支持同步模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 8 **LBCL**：最后一个位时钟脉冲 (Last bit clock pulse)

该位用于在同步模式下选择与发送的最后一个数据位 (MSB) 关联的时钟脉冲是否必须在 CK 引脚上输出。

0：最后一个数据位的时钟脉冲不在 CK 引脚上输出

1：最后一个数据位的时钟脉冲在 CK 引脚上输出

小心： 最后一位为发送的第 7 个、第 8 个或第 9 个数据位，具体取决于 USART_CR1 寄存器中 M 位所选择的 7 位、8 位或 9 位格式。

只有在禁止 USART (UE=0) 时才能写入该位。

注意：如果不支持同步模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 7 保留，必须保持复位值。

位 6 **LBDIE**：LIN 刹车检测中断使能 (LIN break detection interrupt enable)

刹车中断屏蔽（使用刹车分隔符进行刹车检测）。

0：禁止中断

1：当 USART_ISR 寄存器中 LBDF=1 时，生成中断

注意：如果不支持 LIN 模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 5 **LBDL**：LIN 刹车检测长度 (LIN break detection length)

该位用于选择 11 位刹车检测或 10 位刹车检测。

0：10 位刹车检测

1：11 位刹车检测

只有在禁止 USART (UE=0) 时才能写入该位。

注意：如果不支持 LIN 模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 4 **ADDM7**：7 位地址检测/4 位地址检测 (7-bit Address Detection/4-bit Address Detection)

该位用于选择 4 位地址检测或 7 位地址检测。

0：4 位地址检测

1：7 位地址检测（在 8 位数据模式下）

只有在禁止 USART (UE=0) 时才能写入该位

注意：在 7 位和 9 位数据模式下，地址检测分别在 6 位和 8 位地址上完成（ADD[5:0] 和 ADD[7:0]）。

位 3 **DIS_NSS**：
当 DIS_NSS 位置 1 时，忽略 NSS 引脚输入。
0：SPI 从设备选择取决于 NSS 输入引脚。
1：始终选择 SPI 从设备，忽略 NSS 输入引脚。
注意：不支持 SPI 从设备模式时，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 2:1 保留，必须保持复位值。

位 0 **SLVEN**：同步从模式使能 (Synchronous Slave mode enable)
SLVEN 位置 1 时，使能同步从模式。
0：禁止从模式。
1：使能从模式。
注意：不支持 SPI 从设备模式时，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

注意：使能发送器时不得对 CPOL、CPHA 和 LBCL 位进行写操作。

36.8.4 USART 控制寄存器 3 (USART_CR3)

USART control register 3

偏移地址：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXFTCFG[2:0]			RXF TIE	RXFTCFG[2:0]			TCBG TIE	TXFTIE	WUFIE	WUS1	WUS0	SCARCNT[2:0]			Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEP	DEM	DDRE	OVR DIS	ONE BIT	CTSIE	CTSE	RTSE	DMAT	DMAR	SCEN	NACK	HD SEL	IRLP	IREN	EIE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:29 **TXFTCFG[2:0]**：TXFIFO 阈值配置 (TXFIFO threshold configuration)
000：TXFIFO 达到其深度的 1/8
001：TXFIFO 达到其深度的 1/4
010：TXFIFO 达到其深度的 1/2
011：TXFIFO 达到其深度的 3/4
100：TXFIFO 达到其深度的 7/8
101：TXFIFO 变空
其余组合：保留

位 28 **RXFTIE**：RXFIFO 阈值中断使能 (FIFO threshold interrupt enable)
该位由软件置 1 和清零。
0：禁止中断
1：当接收 FIFO 达到 RXFTCFG 中编程的阈值时，生成 USART 中断。

位 27:25 **RXFTCFG[2:0]**：接收 FIFO 阈值配置 (Receive FIFO threshold configuration)

000：接收 FIFO 达到其深度的 1/8
001：接收 FIFO 达到其深度的 1/4
010：接收 FIFO 达到其深度的 1/2
011：接收 FIFO 达到其深度的 3/4
100：接收 FIFO 达到其深度的 7/8
101：接收 FIFO 已满
其余组合：保留

位 24 **TCBGTIE**：保护时间前发送完成中断使能 (Transmission Complete before guard time, interrupt enable)

该位由软件置 1 和清零。
0：禁止中断
1：当 USART_ISR 寄存器中的 TCBGT=1 时，生成 USART 中断

注意：如果 USART 不支持智能卡模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 23 **TXFTIE**：TXFIFO 阈值中断使能 (TXFIFO threshold interrupt enable)

该位由软件置 1 和清零。
0：禁止中断
1：当 TXFIFO 达到 TXFTCFG 中编程的阈值时，生成 USART 中断。

位 22 **WUFIE**：从低功耗模式唤醒中断使能 (Wake-up from low-power mode interrupt enable)

该位由软件置 1 和清零。
0：禁止中断
1：当 USART_ISR 寄存器中的 WUF=1 时，生成 USART 中断

注意：WUFIE 必须在进入低功耗模式前置 1。
如果 USART 不支持从停止模式唤醒功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 21:20 **WUS[1:0]**：从低功耗模式唤醒中断标志选择 (Wake-up from low-power mode interrupt flag selection)

该位域用于指定激活 WUF（从低功耗模式唤醒标志）的事件。
00：WUF 在地址匹配时激活（按 ADD[7:0] 和 ADDM7 所定义）
01：保留。
10：WUF 在起始位检测时激活
11：WUF 在 RXNE/RXFNE 时激活。
只有在禁止 USART (UE=0) 时才能写入该位域。

注意：如果 USART 不支持从停止模式唤醒功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 19:17 **SCARCNT[2:0]**：智能卡自动重试计数 (Smartcard auto-retry count)

该位域用于指定智能卡模式下发送和接收的重试次数。
在发送模式下，该位域用于指定生成发送错误（FE 位置 1）前自动重新发送的重试次数。
在接收模式下，该位域用于指定生成接收错误（RXNE/RXFNE 位和 PE 位置 1）前错误接收尝试的次数。
只有在禁止 USART (UE=0) 时才能编程该位域。
使能 USART (UE=1) 时，该位域只能写入 0x0，以停止重新发送。
0x0：禁止重新发送——发送模式下不会自动重新发送。
0x1 到 0x7：自动重新发送的尝试次数（发出错误信号前）

注意：如果不支持智能卡模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 16 保留，必须保持复位值。

位 15 **DEP**：驱动器使能极性选择 (Driver enable polarity selection)

0：DE 信号高电平有效。

1：DE 信号低电平有效。

只有在禁止 USART (UE=0) 时才能写入该位。

注意：如果不支持驱动器使能功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 14 **DEM**：驱动器使能模式 (Driver enable mode)

该位用于通过 DE 信号激活外部收发器控制。

0：禁止 DE 功能。

1：使能 DE 功能。DE 信号在 RTS 引脚上输出。

只有在禁止 USART (UE=0) 时才能写入该位。

注意：如果不支持驱动器使能功能，该位保留且必须保持复位值。第 1253 页的第 36.4 节：USART 实现。

位 13 **DDRE**：接收出错时的 DMA 禁止 (DMA Disable on Reception Error)

0：接收出错时不禁止 DMA。相应的错误标志置 1，但 RXNE 保持为 0 以防止上溢。因此，将不使能 DMA 请求，从而不会传送错误数据（无 DMA 请求），但会传送接收到的下一个正确数据。（用于智能卡模式）

1：接收出错后禁止 DMA。相应的错误标志以及 RXNE 均置 1。屏蔽 DMA 请求，直到错误标志清零。这意味着软件必须首先禁止 DMA 请求 (DMAR = 0) 或者将 RXNE（使能 FIFO 模式时为 RXFNE）清零，然后再将错误标志清零。

只有在禁止 USART (UE=0) 时才能写入该位。

注意：接收错误包括：奇偶校验错误、帧错误或噪声错误。

位 12 **OVRDIS**：上溢禁止 (Overrun Disable)

该位用于禁止接收上溢检测。

0：接收新数据前未读取已接收的数据时，上溢错误标志 ORE 置 1。

1：禁止上溢功能。如果在 RXNE 标志仍置 1 时接收到新数据，

则 ORE 标志不会置 1，且新接收的数据会覆盖 USART_RDR 寄存器之前的内容。使能 FIFO 模式时，RXFIFO 将被旁路，数据将直接写入 USART_RDR 寄存器中。即使在使能 FIFO 管理时，也将使用 RXNE 标志。

只有在禁止 USART (UE=0) 时才能写入该位。

注意：此控制位用于检查通信流而不会读取数据

位 11 **ONEBIT**：一个采样位方法使能 (One sample bit method enable)

该位允许用户选择采样方法。选择一个采样位方法后，将禁止噪声检测标志 (NE)。

0：三个采样位方法

1：一个采样位方法

只有在禁止 USART (UE=0) 时才能写入该位。

位 10 **CTSIE**：CTS 中断使能 (CTS interrupt enable)

0：禁止中断

1：当 USART_ISR 寄存器中 CTSIF=1 时，生成中断

注意：如果不支持硬件流控制功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 9 **CTSE**：CTS 使能 (CTS enable)

0：禁止 CTS 硬件流控制

1：使能 CTS 模式，仅当 CTS 输入无效（连接到 0）时才发送数据。如果在发送数据时使 CTS 输入有效，会在停止之前完成发送。如果使 CTS 有效时数据已写入数据寄存器，则将延迟发送，直到 CTS 无效。

只有在禁止 USART (UE=0) 时才能写入该位

注意：如果不支持硬件流控制功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 8 **RTSE**：RTS 使能 (RTS enable)

0：禁止 RTS 硬件流控制

1：使能 RTS 输出，仅当接收缓冲区中有空间时才会请求数据。发送完当前字符后应停止发送数据。可以接收数据时使 RTS 输出无效（拉至 0）。

只有在禁止 USART (UE=0) 时才能写入该位。

注意：如果不支持硬件流控制功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 7 **DMAT**：DMA 使能发送器 (DMA enable transmitter)

该位由软件置 1/复位。

1：针对发送使能 DMA 模式

0：针对发送禁止 DMA 模式

位 6 **DMAR**：DMA 使能接收器 (DMA enable receiver)

该位由软件置 1/复位。

1：针对接收使能 DMA 模式

0：针对接收禁止 DMA 模式

位 5 **SCEN**：智能卡模式使能 (Smartcard mode enable)

该位用于使能智能卡模式。

0：禁止智能卡模式

1：使能智能卡模式

只有在禁止 USART (UE=0) 时才能写入该位域。

注意：如果 USART 不支持智能卡模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 4 **NACK**：智能卡 NACK 使能 (Smartcard NACK enable)

0：出现奇偶校验错误时禁止 NACK 发送

1：出现奇偶校验错误时使能 NACK 发送

只有在禁止 USART (UE=0) 时才能写入该位域。

注意：如果 USART 不支持智能卡模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 3 **HDSEL**：半双工选择 (Half-duplex selection)

选择单线半双工模式

0：未选择半双工模式

1：选择半双工模式

只有在禁止 USART (UE=0) 时才能写入该位。

位 2 **IRLP**：IrDA 低功耗 (IrDA low-power)

该位用于选择正常模式和低功耗 IrDA 模式

0：正常模式

1：低功耗模式

只有在禁止 USART (UE=0) 时才能写入该位。

注意：如果不支持 IrDA 模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 1 **IREN**：IrDA 模式使能 (IrDA mode enable)

该位由软件置 1 和清零。

0：禁止 IrDA

1：使能 IrDA

只有在禁止 USART (UE=0) 时才能写入该位。

注意：如果不支持 IrDA 模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 0 **EIE**：错误中断使能 (Error interrupt enable)

如果出现帧错误、上溢错误、噪声标志或 SPI 从设备下溢错误（USART_ISR 寄存器中的 FE=1、ORE=1、NE=1 或 UDR = 1），则需要使用错误中断使能位来使能中断生成。

0：禁止中断

1：USART_ISR 寄存器中的 FE=1、ORE=1、NE=1 或 UDR = 1（在 SPI 从设备模式下）时，生成中断。

36.8.5 USART 波特率寄存器 (USART_BRR)

USART baud rate register

只有在禁止 USART (UE=0) 时才能写入此寄存器。在自动波特率检测模式下，该位由硬件自动更新。

偏移地址：0x0C

复位值：0x0000 0000

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

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

位 31:16 保留，必须保持复位值。

位 15:0 **BRR[15:0]**：USART 波特率 (USART baud rate)

BRR[15:4]

BRR[15:4] 对应于 USARTDIV[15:4]

BRR[3:0]

当 OVER8 = 0 时，BRR[3:0] = USARTDIV[3:0]。

当 OVER8 = 1 时：

BRR[2:0] = USARTDIV[3:0] 右移 1 位。

BRR[3] 必须保持清零。

36.8.6 USART 保护时间和预分频器寄存器 (USART_GTPR)

USART guard time and prescaler register

偏移地址：0x10

复位值：0x0000 0000

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

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

位 31:16 保留，必须保持复位值。

位 15:8 **GT[7:0]**：保护时间值 (Guard time value)
该位域用于编程保护时间值（以波特时钟周期数为单位）。
该位用于智能卡模式。经过此保护时间后，发送完成标志置 1。
只有在禁止 USART (UE=0) 时才能写入该位域。
注意：如果不支持智能卡模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 7:0 **PSC[7:0]**：预分频器值 (Prescaler value)
条件：IrDA 低功耗和正常的 IrDA 模式
PSC[7:0] = IrDA 正常和低功耗波特率
该位域用于编程预分频器，进行 USART 源时钟分频以获得低功耗频率：
使用寄存器中给出的值（8 个有效位）对源时钟进行分频：
00000000：保留 - 不编程此值
00000001：源时钟 1 分频
00000010：源时钟 2 分频
...
条件：智能卡模式
PSC[4:0]：预分频器值 (Prescaler value)
该位域用于编程预分频器，进行 USART 源时钟分频以提供智能卡时钟。
将寄存器中给出的值（5 个有效位）乘以 2 得出源时钟频率的分频系数：
00000：保留 - 不编程此值
00001：源时钟 2 分频
00010：源时钟 4 分频
00011：源时钟 6 分频
...
只有在禁止 USART (UE=0) 时才能写入该位域。
注意：如果使用智能卡模式，则位 [7:5] 必须保持清零。
不支持智能卡和 IrDA 模式时，该位域保留并由硬件强制清零。请参见第 1253 页的第 36.4 节：USART 实现。

36.8.7 USART 接收器超时寄存器 (USART_RTOR)

USART receiver timeout register

偏移地址：0x14

复位值：0x0000 0000

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

位 31:24 **BLEN[7:0]**：块长度 (Block Length)

该位域用于提供智能卡 T=1 接收下的块长度。其值等于信息字符数 + 结尾字段的长度 (1-LEC/2-CRC) - 1。

例如：

BLEN = 0 -> 0 个信息字符 + LEC

BLEN = 1 -> 0 个信息字符 + CRC

BLEN = 255 -> 254 个信息字符 + CRC（总共 256 个字符）

在智能卡模式下，块长度计数器在 TXE=0（使能 FIFO 模式时为 TXFE = 0）时复位。

该位域也可用于其他模式。这种情况下，块长度计数器在 RE=0（禁止接收器）和/或 EOBCF 位写入 1 时复位。

注意：块接收开始后可编程此值（使用起始字段中 LEN 字符中的数据）。每个接收到的块只能对此值编程一次。

位 23:0 **RTO[23:0]**：接收器超时值 (Receiver timeout value)

该位域用于提供接收器的超时值（以位持续时间数为单位）。

在标准模式下，如果在接收到最后一个字符后，在 RTO 值对应的时间内未检测到新的起始位，则 RTOF 标志置 1。

在智能卡模式下，此值用于实施 CWT 和 BWT。有关更多详细信息，请参见智能卡章节。在标准模式下，从接收到的最后一个字符的起始位开始执行 CWT/BWT 测量。

注意：每个接收到的字符只能对此值编程一次。

注意：可以实时写入 RTOR。如果新值小于或等于计数器的值，RTOF 标志置 1。

如果不支持接收器超时功能，此寄存器保留并由硬件强制为“0x00000000”。请参见第 1253 页的第 36.4 节：USART 实现。

36.8.8 USART 请求寄存器 (USART_RQR)

USART request register

偏移地址：0x18

复位值：0x0000 0000

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

位 31:5 保留，必须保持复位值。

位 4 **TXFRQ**：发送数据刷新请求 (Transmit data flush request)

禁止 FIFO 模式时，向该位写入 1 会将 TXE 标志置 1。这可丢弃发送数据。由于错误 (NACK) 而未发送数据且 USART_ISR 寄存器中的 FE 标志有效时，只能在智能卡模式下使用该位。如果 USART 不支持智能卡模式，该位保留且必须保持复位值。

使能 FIFO 时，TXFRQ 位置 1 以清空整个 FIFO。这会将 TXFE 标志（发送 FIFO 为空，USART_ISR 寄存器中的位 23）置 1。在 UART 模式和智能卡模式下都支持清空发送 FIFO。

注意：在 FIFO 模式下，TXFNF 标志在清空请求期间复位，直到 TxFIFO 为空，以确保数据寄存器中没有写入数据。

位 3 **RXFRQ**：接收数据刷新请求 (Receive data flush request)

向该位写入 1 将清空整个接收 FIFO（即，将 RXFNE 位清零）。

这可以丢弃接收的数据而不对其执行读取操作，并避免发生上溢情况。

位 2 **MMRQ**：静默模式请求 (Mute mode request)

向该位写入 1 可将 USART 置于静默模式，并将 RWU 标志复位。

位 1 **SBKRQ**：发送刹车请求 (Send break request)

向该位写入 1 可将 SBKF 标志置 1 并在发送设备可用后立即请求在线路上发送 BREAK。

注意：如果应用需要在之前插入的所有数据（包括尚未发送的数据）后发送刹车字符，软件必须等到 TXE 标志使能后将 SBKRQ 位置 1。

位 0 **ABRRQ**：自动波特率请求 (Auto baud rate request)

向该位写入 1 可复位 USART_ISR 中的 ABRF 和 ABRE 标志，并请求对下一个接收到的数据帧进行自动波特率测量。

注意：如果 USART 不支持自动波特率功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

36.8.9 USART 中断和状态寄存器 (USART_ISR)

USART interrupt and status register

偏移地址：0x1C

复位值：0x0XX0 00C0

使能 FIFO/智能卡模式时，XX = 28

使能 FIFO 模式且禁止智能卡模式时，XX = 08

同一寄存器可用于使能 FIFO 模式（本节）和禁止 FIFO 模式（下一节）的情况。

使能 FIFO 模式

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TXFT	RXFT	TCBGT	RXFF	TXFE	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY
				r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABRF	ABRE	UDR	EOBF	RTOF	CTS	CTSIF	LBDF	TXFNF	TC	RXFNE	IDLE	ORE	NE	FE	PE
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:28 保留，必须保持复位值。

位 27 **TXFT**：TXFIFO 阈值标志 (FIFO threshold flag)

当 TXFIFO 达到在 USART_CR3 寄存器的 TXFTCFG 中编程的阈值时（即 TXFIFO 包含 TXFTCFG 个空位置），该位由硬件置 1。如果 USART_CR3 寄存器中的 TXFTIE 位 =1（位 31），则会生成中断。

0：TXFIFO 未达到编程的阈值。

1：TXFIFO 已达到编程的阈值。

位 26 **RXFT**：RXFIFO 阈值标志 (RXFIFO threshold flag)

达到在 USART_CR3 寄存器的 RXFTCFG 中编程的阈值时，该位由硬件置 1。这意味着，接收 FIFO 中有 (RXFTCFG - 1) 个数据，USART_RDR 寄存器中有一个数据。如果 USART_CR3 寄存器中的 RXFTIE 位 = 1（位 27），则会生成中断。

0：接收 FIFO 未达到编程的阈值。

1：接收 FIFO 已达到编程的阈值。

注意：当 RXFTCFG 阈值配置为 101 时，如果存在 16 个数据（即 RXFIFO 中有 15 个数据，USART_RDR 中有 1 个数据），则 RXFT 标志将置 1。因此，接收到的第 17 个数据不会导致上溢错误。接收到第 18 个数据后会发生上溢错误。

位 25 **TCBGT**：保护时间前发送完成标志 (Transmission complete before guard time flag)

当写入到 USART_TDR 中的最后一个数据已正确从移位寄存器中发出时，该位置 1。

如果包含数据的帧完成发送，并且智能卡未发回任何 NACK，则该位在智能卡模式下由硬件置 1。如果 USART_CR3 寄存器中的 TCBGTIE=1，则会生成中断。

该位由软件清零，方法是向 USART_ICR 寄存器中的 TCBGTCF 写入 1 或向 USART_TDR 寄存器执行写操作。

0：发送未完成或发送未成功完成（即，从智能卡接收到 NACK）

1：发送成功完成（在保护时间结束之前完成，智能卡未发送 NACK）。

注意：如果 USART 不支持智能卡模式，该位保留且保持复位值。如果 USART 支持智能卡模式并使能智能卡模式，则 TCBGT 复位值为 1。请参见第 1253 页的第 36.4 节：USART 实现。

位 24 **RXFF**：RXFIFO 已满 (RXFIFO Full)

当接收到的数据量对应于 RXFIFO 大小 + 1（RXFIFO 已满 + USART_RDR 寄存器中的 1 个数据）时，该位由硬件置 1。

如果 USART_CR1 寄存器中 RXFFIE 位 = 1，则会生成中断。

0：RXFIFO 未满。

1：RxFIFO 已满。

位 23 **TXFE**：TXFIFO 为空 (TXFIFO Empty)

当 TXFIFO 为空时，该位由硬件置 1。当 TXFIFO 包含至少一个数据时，该标志被清零。也可以通过向 USART_RQR 寄存器中的位 TXFRQ（位 4）写入 1 将 TXFE 标志置 1。

如果 USART_CR1 寄存器中的 TXFEIE 位 = 1（位 30），则会生成中断。

0：TXFIFO 非空。

1：TXFIFO 为空。

位 22 **REACK**：接收使能确认标志 (Receive enable acknowledge flag)

USART 采用接收使能值时，通过硬件将该位置 1/复位。

该位可用于验证 USART 是否准备好在进入低功耗模式前接收数据。

注意：如果 USART 不支持从停止模式唤醒功能，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 21 **TEACK**：发送使能确认标志 (Transmit enable acknowledge flag)

USART 采用发送使能值时，通过硬件将该位置 1/复位。

通过写入 TE=0 生成空闲帧请求，然后在 USART_CR1 寄存器中写入 TE=1 以遵循 TE=0 最短周期时，可使用该位。

位 20 **WUF**：从低功耗模式唤醒标志 (Wake-up from low-power mode flag)

当检测到唤醒事件时，该位由硬件置 1。事件通过 WUS 位域定义。该位由软件清零，方法是向 USART_ICR 寄存器中的 WUCF 写入 1。

如果 USART_CR3 寄存器中 WUFIE=1，则会生成中断。

注意：当 UESM 清零时，WUF 标志也清零。

如果 USART 不支持从停止模式唤醒功能，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 19 **RWU**：*接收器从静默模式唤醒 (Receiver wake-up from Mute mode)*

该位指示 USART 是否处于静默模式。当识别出唤醒/静默序列时，该位由硬件清零/置 1。静默模式控制序列（地址或 IDLE）通过 USART_CR1 寄存器中的 WAKE 位进行选择。

当选择 IDLE 模式下唤醒时，该位只能通过用软件向 USART_RQR 寄存器中的 MMRQ 位写 1 的方式置 1。

0：接收器处于工作模式

1：接收器处于静默模式

注意：如果 USART 不支持从停止模式唤醒功能，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 18 **SBKF**：发送刹车标志 (Send break flag)

该位指示已请求发送刹车字符。通过将 1 写入 USART_CR3 寄存器中的 SBKRQ 位，该位由软件置 1。该位在刹车发送的停止位期间由硬件自动复位。

0：不发送刹车字符

1：刹车字符已发送

位 17 **CMF**：字符匹配标志 (Character match flag)

接收到由 ADD[7:0] 定义的字符后由硬件将该位置 1。通过向 USART_ICR 寄存器中的 CMCF 写入 1，该位由软件清零。

如果 USART_CR1 寄存器中 CMIE=1，则会生成中断。

0：未检测到字符匹配

1：检测到字符匹配

位 16 **BUSY**：忙标志 (Busy flag)

该位由硬件置 1 和复位。当 RX 线路上正在进行通信（成功检测到起始位）时有效。在接收结束（成功或失败）时复位。

0：USART 处于空闲状态（无接收）

1：正在接收

位 15 **ABRF**：自动波特率标志 (Auto baud rate flag)

已设置自动波特率（RXFNE 也置 1，并在 RXFNEIE = 1 时生成中断），或者自动波特率操作未成功完成时，该位由硬件置 1 (ABRE=1)（此时，ABRE、RXFNE 和 FE 也置 1）

为请求新的自动波特率检测，通过向 USART_RQR 寄存器中的 ABRRQ 写入 1，该位由软件清零。

注意：如果 USART 不支持自动波特率功能，该位保留且保持复位值。

位 14 **ABRE**：自动波特率错误 (Auto baud rate error)

如果波特率测量失败（波特率超出范围或字符比较失败），该位由硬件置 1。

通过将 1 写入 USART_RQR 寄存器中的 ABRRQ 位，该位由软件清零。

注意：如果 USART 不支持自动波特率功能，该位保留且保持复位值。

位 13 **UDR**：SPI 从设备下溢错误标志 (SPI slave underrun error flag)

在从发送模式下，如果在软件尚未将任何值加载到 USART_TDR 之前出现第一个数据发送时钟脉冲，此标志将置 1。通过将 USART_ICR 寄存器中的 UDRCF 位置 1 来复位此标志。

0：无下溢错误

1：存在下溢错误

注意：如果 USART 不支持 SPI 从设备模式，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 12 **EOBF**：块结束标志 (End of block flag)

接收到完整块后，该位由硬件置 1（例如 T=1 智能卡模式）。接收到的字节数（从块的起始处，包括起始字段）等于或大于 BLEN + 4 时执行检测。

如果 USART_CR1 寄存器中 EOBIE = 1，则会生成中断。

通过向 USART_ICR 寄存器中的 EOBCF 写入 1，该位由软件清零。

0：未达到块结束

1：已达到块结束（字符数）

注意：如果不支持智能卡模式，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 11 **RTOF**：接收器超时 (Receiver timeout)

已经过在 RTOR 寄存器中编程的超时值后，如果无任何通信，该位由硬件置 1。通过向 USART_ICR 寄存器中的 RTOCF 写入 1，该位由软件清零。

如果 USART_CR2 寄存器中 RTOIE=1，则会生成中断。

在智能卡模式下，该超时对应于 CWT 或 BWT 时间。

0：未达到超时值

1：已达到超时值，未接收到任何数据

注意：如果 RTOR 寄存器中编程的时间值将 2 个字符隔开，则 RTOF 不置 1。如果此时间大于该值 + 2 个采样时间（2/16 或 2/8，具体取决于过采样方法），则 RTOF 标志置 1。

即使 RE = 0，计数器仍会计数，但 RTOF 仅在 RE = 1 时置 1。如果 RE 置 1 时已经超时，则 RTOF 会置 1。

如果 USART 不支持接收器超时功能，该位保留且保持复位值。

位 10 **CTS**：CTS 标志 (CTS flag)

该位由硬件置 1/复位。该位是对 CTS 输入引脚的状态取反。

0：CTS 线置 1

1：CTS 线复位

注意：如果不支持硬件流控制功能，该位保留且保持复位值。

位 9 **CTSIF**：CTS 中断标志 (CTS interrupt flag)

如果 CTSE 位置 1，当 CTS 输入切换时，该位由硬件置 1。通过将 1 写入 USART_ICR 寄存器中的 CTSCF 位，该位由软件清零。

如果 USART_CR3 寄存器中 CTSIE=1，则会生成中断。

0：CTS 状态线上未发生变化

1：CTS 状态线上发生变化

注意：如果不支持硬件流控制功能，该位保留且保持复位值。

位 8 **LBDF**：LIN 刹车检测标志 (LIN break detection flag)

检测到 LIN 刹车时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 LBDCF 写入 1，该位由软件清零。

如果 USART_CR2 寄存器中 LBDIE = 1，则会生成中断。

0：未检测到 LIN 刹车

1：检测到 LIN 刹车

注意：如果 USART 不支持 LIN 模式，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 7 **TXFNF**：TXFIFO 未满 (TXFIFO not full)

TXFNF 会在 TXFIFO 未满时由硬件置 1，表示可向 USART_TDR 中写入数据。每次对 USART_TDR 进行写操作都会将数据置于 TXFIFO 中。该标志会保持置 1，直到 TXFIFO 已满。当 TXFIFO 已满时，该标志清零，表示不能向 USART_TDR 中写入数据。

如果 USART_CR1 寄存器中 TXFNFIE 位 = 1，则会生成中断。

0：发送 FIFO 已满

1：发送 FIFO 未满

注意：在清空请求期间，TXFNF 保持复位，直到 TXFIFO 为空。发送清空请求（通过将 TXFRQ 位置 1）后，必须先检查 TXFNF 标志，然后再写入 TXFIFO（TXFNF 和 TXFE 会同时置 1）。

单缓冲区发送期间使用该位。

位 6 **TC**：发送完成 (Transmission complete)

该位指示写入到 USART_TDR 中的最后一个数据已从移位寄存器中发出。TC 标志的行为如下：

- 当 TDN = 0 时，如果已完成对包含数据的帧的发送并且 TXE/TXFE 置 1，TC 标志将会置 1。
- 当 TDN 等于 TXFIFO 中的数据量时，如果 TXFIFO 为空并且达到 TDN 时，TC 标志将会置 1。
- 当 TDN 大于 TXFIFO 中的数据量时，TC 将保持清零状态，直到 TXFIFO 再次填满以达到设定要传输的数据量。
- 当 TDN 小于 TXFIFO 中的数据量时，即使 TXFIFO 不为空，当达到 TDN 时 TC 也会置 1。

如果 USART_CR1 寄存器中 TCIE = 1，则会生成中断。

TC 位由软件清零，方法是向 USART_ICR 寄存器中的 TCCF 写入 1 或向 USART_TDR 寄存器执行写操作。

位 5 **RXFNE**：RXFIFO 非空 (RXFIFO not empty)

RXFIFO 非空时，RXFNE 位由硬件置 1，这表示可以从 USART_RDR 寄存器读取数据。每次对 USART_RDR 进行读操作都会在 RXFIFO 中释放一个位置。

RXFNE 在 RXFIFO 为空时清零。也可以通过向 USART_RQR 寄存器中的 RXFRQ 位写入 1 将 RXFNE 标志位清零。

如果 USART_CR1 寄存器中 RXFNEIE=1，则会生成中断。

0：未接收到数据

1：已准备好读取接收到的数据

位 4 **IDLE**：检测到空闲线路 (Idle line detected)

检测到空闲线路时，该位由硬件置 1。如果 USART_CR1 寄存器中 IDLEIE = 1，则会生成中断。通过向 USART_ICR 寄存器中的 IDLECF 写入 1，该位由软件清零。

0：未检测到空闲线路

1：检测到空闲线路

注意：直到 RXFNE 位已置 1 时（即，当出现新的空闲线路时）IDLE 位才会被再次置 1。

使能静默模式 (MME=1) 后，如果 USART 未静默 (RWU=0)，则 IDLE 置 1，无论是否通过 WAKE 位选择了静默模式。如果 RWU=1，IDLE 不置 1。

位 3 **ORE**：上溢错误 (Overrun error)

在 RXFF = 1 的情况下，当移位寄存器中当前正在接收的数据准备好传输到 USART_RDR 寄存器时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 ORECF 写入 1，该位由软件清零。

如果 USART_CR1 寄存器中 RXFNEIE = 1 或 USART_CR3 寄存器中 EIE = 1，则会生成中断。

0：无上溢错误

1：检测到上溢错误

注意：当该位置 1 时，USART_RDR 寄存器的内容不会丢失，但移位寄存器会被覆盖。EIE 位置 1 后，如果在多缓冲区通信中 ORE 标志置 1，则会生成中断。

USART_CR3 寄存器中的 OVRDIS 位置 1 时，该位将被永久强制清零（无上溢检测）。

位 2 **NE**：噪声检测标志 (Noise detection flag)

当在接收的帧上检测到噪声时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 NFCF 写入 1，该位由软件清零。

0：未检测到噪声

1：检测到噪声

注意：该位不会生成中断，因为该位出现的时间与本身生成中断的 RXFNE 位出现的时间相同。EIE 位置 1 后，如果在多缓冲区通信中 NE 标志置 1，则会生成中断。

当线路无噪声时，可以通过将 ONEBIT 位编程为 1 提高 USART 对偏差的容差来禁止 NE 标志（请参见第 1271 页的第 36.5.9 节：USART 接收器对时钟偏差的容差）。

此错误与 USART_RDR 中的字符相关联。

位 1 **FE**：帧错误 (Framing error)

当检测到去同步化、过度的噪声或刹车字符时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 FECF 写入 1，该位由软件清零。

在智能卡模式下发送数据时，如果在达到最大发送尝试次数后仍未成功（智能卡向数据帧发送 NACK 信号），则该位置 1。

如果 USART_CR3 寄存器中 EIE = 1，则会生成中断。

0：未检测到帧错误

1：检测到帧错误或刹车字符

注意：此错误与 USART_RDR 中的字符相关联。

位 0 **PE**：奇偶校验错误 (Parity error)

当在接收模式下发生奇偶校验错误时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 PECF 写入 1，该位由软件清零。

如果 USART_CR1 寄存器中的 PEIE = 1，则会生成中断。

0：无奇偶校验错误

1：奇偶校验错误

注意：此错误与 USART_RDR 中的字符相关联。

36.8.10 USART 中断和状态寄存器 [备用] (USART_ISR)

USART interrupt and status register

偏移地址：0x1C

复位值：0x0000 00C0

同一寄存器可用于使能 FIFO 模式（上一节）和禁止 FIFO 模式（本节）的情况。

禁止 FIFO 模式

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	TCBGT	Res.	Res.	RE ACK	TE ACK	WUF	RWU	SBKF	CMF	BUSY
						r			r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABRF	ABRE	UDR	EOBF	RTOF	CTS	CTSIF	LBDF	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:26 保留，必须保持复位值。

位 25 **TCBGT**：保护时间前发送完成标志 (Transmission complete before guard time flag)

当写入到 USART_TDR 中的最后一个数据已正确从移位寄存器中发出时，该位置 1。

如果包含数据的帧完成发送，并且智能卡未发回任何 NACK，则该位在智能卡模式下由硬件置 1。如果 USART_CR3 寄存器中的 TCBGTIE=1，则会生成中断。

该位由软件清零，方法是向 USART_ICR 寄存器中的 TCBGTCF 写入 1 或向 USART_TDR 寄存器执行写操作。

0：发送未完成或发送未成功完成（即，从智能卡接收到 NACK）

1：发送成功完成（在保护时间结束之前完成，智能卡未发送 NACK）。

注意：如果 USART 不支持智能卡模式，该位保留且保持复位值。如果 USART 支持智能卡模式并使能智能卡模式，则 TCBGT 复位值为 1。请参见第 1253 页的第 36.4 节：USART 实现。

位 24:23 保留，必须保持复位值。

位 22 **REACK**：接收使能确认标志 (Receive enable acknowledge flag)

USART 采用接收使能值时，通过硬件将该位置 1/复位。

该位可用于验证 USART 是否准备好在进入低功耗模式前接收数据。

注意：如果 USART 不支持从停止模式唤醒功能，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 21 **TEACK**：发送使能确认标志 (Transmit enable acknowledge flag)

USART 采用发送使能值时，通过硬件将该位置 1/复位。

通过写入 TE=0 生成空闲帧请求，然后在 USART_CR1 寄存器中写入 TE=1 以遵循 TE=0 最短周期时，可使用该位。

位 20 **WUF**：从低功耗模式唤醒标志 (Wake-up from low-power mode flag)

当检测到唤醒事件时，该位由硬件置 1。事件通过 WUS 位域定义。该位由软件清零，方法是向 USART_ICR 寄存器中的 WUCF 写入 1。

如果 USART_CR3 寄存器中 WUFIE=1，则会生成中断。

注意：当 UESM 清零时，WUF 标志也清零。

如果 USART 不支持从停止模式唤醒功能，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 19 **RWU**：*接收器从静默模式唤醒 (Receiver wake-up from Mute mode)*

该位指示 USART 是否处于静默模式。当识别出唤醒/静默序列时，该位由硬件清零/置 1。静默模式控制序列（地址或 IDLE）通过 USART_CR1 寄存器中的 WAKE 位进行选择。

当选择 IDLE 模式下唤醒时，该位只能通过用软件向 USART_RQR 寄存器中的 MMRQ 位写 1 的方式置 1。

0：接收器处于工作模式

1：接收器处于静默模式

注意：如果 USART 不支持从停止模式唤醒功能，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 18 **SBKF**：发送刹车标志 (Send break flag)

该位指示已请求发送刹车字符。通过将 1 写入 USART_CR3 寄存器中的 SBKRQ 位，该位由软件置 1。该位在刹车发送的停止位期间由硬件自动复位。

0：不发送刹车字符

1：刹车字符已发送

位 17 **CMF**：字符匹配标志 (Character match flag)

接收到由 ADD[7:0] 定义的字符后由硬件将该位置 1。通过向 USART_ICR 寄存器中的 CMCF 写入 1，该位由软件清零。

如果 USART_CR1 寄存器中 CMIE=1，则会生成中断。

0：未检测到字符匹配

1：检测到字符匹配

位 16 **BUSY**：忙标志 (Busy flag)

该位由硬件置 1 和复位。当 RX 线路上正在进行通信（成功检测到起始位）时有效。在接收结束（成功或失败）时复位。

0：USART 处于空闲状态（无接收）

1：正在接收

位 15 **ABRF**：自动波特率标志 (Auto baud rate flag)

已设置自动波特率（RXNE 也置 1，在 RXNEIE = 1 时生成中断），或者自动波特率操作未成功完成时，该位由硬件置 1 (ABRE=1)（此时，ABRE、RXNE 和 FE 也置 1）

为请求新的自动波特率检测，通过向 USART_RQR 寄存器中的 ABRRQ 写入 1，该位由软件清零。

注意：如果 USART 不支持自动波特率功能，该位保留且保持复位值。

位 14 **ABRE**：自动波特率错误 (Auto baud rate error)

如果波特率测量失败（波特率超出范围或字符比较失败），该位由硬件置 1。

通过将 1 写入 USART_RQR 寄存器中的 ABRRQ 位，该位由软件清零。

注意：如果 USART 不支持自动波特率功能，该位保留且保持复位值。

位 13 **UDR**：SPI 从设备下溢错误标志 (SPI slave underrun error flag)

在从发送模式下，如果在软件尚未将任何值加载到 USART_TDR 之前出现第一个数据发送时钟脉冲，此标志将置 1。通过将 USART_ICR 寄存器中的 UDRCF 位置 1 来复位此标志。

0：无下溢错误

1：存在下溢错误

注意：如果 USART 不支持 SPI 从设备模式，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 12 **EOBF**：块结束标志 (End of block flag)

接收到完整块后，该位由硬件置 1（例如 T=1 智能卡模式）。接收到的字节数（从块的起始处，包括起始字段）等于或大于 BLEN + 4 时执行检测。

如果 USART_CR1 寄存器中 EOBIE = 1，则会生成中断。

通过向 USART_ICR 寄存器中的 EOBCF 写入 1，该位由软件清零。

0：未达到块结束

1：已达到块结束（字符数）

注意：如果不支持智能卡模式，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 11 **RTOF**：接收器超时 (Receiver timeout)

已经过在 RTOR 寄存器中编程的超时值后，如果无任何通信，该位由硬件置 1。通过向 USART_ICR 寄存器中的 RTOCF 写入 1，该位由软件清零。

如果 USART_CR2 寄存器中 RTOIE=1，则会生成中断。

在智能卡模式下，该超时对应于 CWT 或 BWT 时间。

0：未达到超时值

1：已达到超时值，未接收到任何数据

注意：如果 RTOR 寄存器中编程的时间值将 2 个字符隔开，则 RTOF 不置 1。如果此时间大于该值 + 2 个采样时间（2/16 或 2/8，具体取决于过采样方法），则 RTOF 标志置 1。

即使 RE = 0，计数器仍会计数，但 RTOF 仅在 RE = 1 时置 1。如果 RE 置 1 时已经超时，则 RTOF 会置 1。

如果 USART 不支持接收器超时功能，该位保留且保持复位值。

位 10 **CTS**：CTS 标志 (CTS flag)

该位由硬件置 1/复位。该位是对 CTS 输入引脚的状态取反。

0：CTS 线置 1

1：CTS 线复位

注意：如果不支持硬件流控制功能，该位保留且保持复位值。

位 9 **CTSIF**：CTS 中断标志 (CTS interrupt flag)

如果 CTSE 位置 1，当 CTS 输入切换时，该位由硬件置 1。通过将 1 写入 USART_ICR 寄存器中的 CTSCF 位，该位由软件清零。

如果 USART_CR3 寄存器中 CTSIE=1，则会生成中断。

0：CTS 状态线上未发生变化

1：CTS 状态线上发生变化

注意：如果不支持硬件流控制功能，该位保留且保持复位值。

位 8 **LBDF**：LIN 刹车检测标志 (LIN break detection flag)

检测到 LIN 刹车时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 LBDCF 写入 1，该位由软件清零。

如果 USART_CR2 寄存器中 LBDIE = 1，则会生成中断。

0：未检测到 LIN 刹车

1：检测到 LIN 刹车

注意：如果 USART 不支持 LIN 模式，该位保留且保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 7 **TXE**：发送数据寄存器为空 (Transmit data register empty)

当 USART_TDR 寄存器的内容已传输到移位寄存器时，TXE 由硬件置 1。通过对 USART_TDR 寄存器执行写操作将该位清零。为丢弃数据（仅在智能卡 T=0 模式下出现发送故障时），也可以通过向 USART_RQR 寄存器中的 TXFRQ 写入 1 来将 TXE 标志置 1。

如果 USART_CR1 寄存器中 TXEIE 位 = 1，则会生成中断。

0：数据寄存器已满

1：数据寄存器已满

位 6 **TC**：发送完成 (Transmission complete)

该位指示写入到 USART_TDR 中的最后一个数据已从移位寄存器中发出。如果已完成对包含数据的帧的发送并且 TXE 置 1，TC 标志将会置 1。

如果 USART_CR1 寄存器中 TCIE = 1，则会生成中断。

TC 位由软件清零，方法是向 USART_ICR 寄存器中的 TCCF 写入 1 或向 USART_TDR 寄存器执行写操作。

位 5 **RXNE**：读取数据寄存器不为空 (Read data register not empty)

当 USART_RDR 移位寄存器的内容已传输到 USART_RDR 寄存器时，RXNE 位由硬件置 1。通过对 USART_RDR 寄存器执行读取操作将该位清零。也可以通过将 USART_RQR 寄存器中的 RXFRQ 位置 1 将 RXNE 标志位清零。

如果 USART_CR1 寄存器中 RXNEIE = 1，则会生成中断。

0：未接收到数据

1：已准备好读取接收到的数据

位 4 **IDLE**：检测到空闲线路 (Idle line detected)

检测到空闲线路时，该位由硬件置 1。如果 USART_CR1 寄存器中 IDLEIE = 1，则会生成中断。通过向 USART_ICR 寄存器中的 IDLECF 写入 1，该位由软件清零。

0：未检测到空闲线路

1：检测到空闲线路

注意：直到 RXNE 位已置 1 时（即，当出现新的空闲线路时）IDLE 位才会被再次置 1。

使能静默模式 (MME=1) 后，如果 USART 未静默 (RWU=0)，则 IDLE 置 1，无论是否通过 WAKE 位选择了静默模式。如果 RWU=1，IDLE 不置 1。

位 3 **ORE**：上溢错误 (Overrun error)

在 RXNE = 1 的情况下，当移位寄存器中当前正在接收的数据

准备好传输到 USART_RDR 寄存器时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 ORECF 写入 1，该位由软件清零。

如果 USART_CR1 寄存器中 RXNEIE = 1 或 USART_CR3 寄存器中 EIE = 1，则会生成中断。

1：检测到上溢错误

注意：当该位置 1 时，USART_RDR 寄存器的内容不会丢失，但移位寄存器会被覆盖。EIE 位置 1 后，如果在多缓冲区通信中 ORE 标志置 1，则会生成中断。

USART_CR3 寄存器中的 OVRDIS 位置 1 时，该位将被永久强制清零（无上溢检测）。

位 2 **NE**：噪声检测标志 (Noise detection flag)

当在接收的帧上检测到噪声时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 NFCF 写入 1，该位由软件清零。

0：未检测到噪声

1：检测到噪声

注意：该位不会生成中断，因为该位出现的时间与本身生成中断的 RXNE 位出现的时间相同。EIE 位置 1 后，如果在多缓冲区通信中 NE 标志置 1，则会生成中断。

当线路无噪声时，可以通过将 ONEBIT 位编程为 1 提高 USART 对偏差的容差来禁止 NE 标志（请参见第 1271 页的第 36.5.9 节：USART 接收器对时钟偏差的容差）。

此错误与 USART_RDR 中的字符相关联。

位 1 **FE**：帧错误 (Framing error)

当检测到去同步化、过度的噪声或刹车字符时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 FECF 写入 1，该位由软件清零。

在智能卡模式下发送数据时，如果在达到最大发送尝试次数后仍未成功（智能卡向数据帧发送 NACK 信号），则该位置 1。

如果 USART_CR3 寄存器中 EIE = 1，则会生成中断。

0：未检测到帧错误

1：检测到帧错误或刹车字符

注意：此错误与 USART_RDR 中的字符相关联。

位 0 **PE**：奇偶校验错误 (Parity error)

当在接收模式下发生奇偶校验错误时，该位由硬件置 1。通过向 USART_ICR 寄存器中的 PECF 写入 1，该位由软件清零。

如果 USART_CR1 寄存器中的 PEIE = 1，则会生成中断。

0：无奇偶校验错误

1：奇偶校验错误

注意：此错误与 USART_RDR 中的字符相关联。

36.8.11 USART 中断标志清零寄存器 (USART_ICR)

USART interrupt flag clear register

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUCF	Res.	Res.	CMCF	Res.
											w			w	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	UDRCF	EOBCF	RTOCF	Res.	CTSCF	LBDCF	TCBGTCF	TCCF	TXFECF	IDLECF	ORECF	NECF	FECF	PECF
		w	w	w		w	w	w	w	w	w	w	w	w	w

位 31:21 保留，必须保持复位值。

位 20 **WUCF**：从低功耗模式唤醒清零标志 (Wake-up from low-power mode clear flag)

将 1 写入该位时，USART_ISR 寄存器中 WUF 标志将清零。

注意：如果 USART 不支持从停止模式唤醒功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 19:18 保留，必须保持复位值。

位 17 **CMCF**：字符匹配清零标志 (Character match clear flag)

将 1 写入该位时，USART_ISR 寄存器中 CMF 标志将清零。

位 16:14 保留，必须保持复位值。

位 13 **UDRCF**：SPI 从设备下溢清零标志 (SPI slave underrun clear flag)

将 1 写入该位时，USART_ISR 寄存器中 UDRF 标志将清零。

注意：如果 USART 不支持 SPI 从设备模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 12 **EOBCF**：块结束清零标志 (End of block clear flag)

将 1 写入该位时，USART_ISR 寄存器中 EOBF 标志将清零。

注意：如果 USART 不支持智能卡模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 11 **RTOCF**：接收器超时清零标志 (Receiver timeout clear flag)

将 1 写入该位时，USART_ISR 寄存器中 RTOF 标志将清零。

注意：如果 USART 不支持接收器超时功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 10 保留，必须保持复位值。

位 9 **CTSCF**：CTS 清零标志 (CTS clear flag)
将 1 写入该位时，USART_ISR 寄存器中 CTSIF 标志将清零。
注意：如果不支持硬件流控制功能，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 8 **LBDCF**：LIN 刹车检测清零标志 (LIN break detection clear flag)
将 1 写入该位时，USART_ISR 寄存器中 LBDF 标志将清零。
注意：如果不支持 LIN 模式，该位保留且必须保持复位值。请参见第 1253 页的第 36.4 节：USART 实现。

位 7 **TCBGTCF**：保护时间前发送完成清零标志 (Transmission complete before Guard time clear flag)
将 1 写入该位时，USART_ISR 寄存器中 TCBGT 标志将清零。

位 6 **TCCF**：发送完成清零标志 (Transmission complete clear flag)
将 1 写入该位时，USART_ISR 寄存器中 TC 标志将清零。

位 5 **TXFECF**：TXFIFO 为空清零标志 (TXFIFO empty clear flag)
将 1 写入该位时，USART_ISR 寄存器中 TXFE 标志将清零。

位 4 **IDLECF**：检测到空闲线路清零标志 (Idle line detected clear flag)
将 1 写入该位时，USART_ISR 寄存器中 IDLE 标志将清零。

位 3 **ORECF**：上溢错误清零标志 (Overrun error clear flag)
将 1 写入该位时，USART_ISR 寄存器中 ORE 标志将清零。

位 2 **NECF**：检测到噪声清零标志 (Noise detected clear flag)
将 1 写入该位时，USART_ISR 寄存器中 NE 标志将清零。

位 1 **FECF**：帧错误清零标志 (Framing error clear flag)
将 1 写入该位时，USART_ISR 寄存器中 FE 标志将清零。

位 0 **PECF**：奇偶校验错误清零标志 (Parity error clear flag)
将 1 写入该位时，USART_ISR 寄存器中 PE 标志将清零。

36.8.12 USART 接收数据寄存器 (USART_RDR)

USART receive data register

偏移地址：0x24

复位值：0x0000 0000

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

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

位 31:9 保留，必须保持复位值。

位 8:0 **RDR[8:0]**：接收数据值 (Receive data value)
包含接收到的数据字符。
RDR 寄存器在输入移位寄存器和内部总线之间提供了并行接口（请参见*图 425*）。
在使能奇偶校验位的情况下进行接收时，从 MSB 位中读取的值为接收到的奇偶校验位。

36.8.13 USART 发送数据寄存器 (USART_TDR)

USART transmit data register

偏移地址：0x28

复位值：0x0000 0000

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

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

位 31:9 保留，必须保持复位值。

位 8:0 **TDR[8:0]**：发送数据值 (Transmit data value)

包含要发送的数据字符。

USART_TDR 寄存器在内部总线和输出移位寄存器之间提供了并行接口（请参见*图 425*）。

在使能奇偶校验位的情况下（USART_CR1 寄存器中的 PCE 位被置 1）进行发送时，由于 MSB 的写入值（位 7 或位 8，具体取决于数据长度）会被奇偶校验位所取代，因此该值不起任何作用。

注意：只能在 TXE/TXFNF=1 时写入此寄存器。

36.8.14 USART 预分频器寄存器 (USART_PRESC)

USART prescaler register

只有在禁止 USART (UE=0) 时才能写入此寄存器。

偏移地址：0x2C

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **PRESCALER[3:0]**：时钟预分频器 (Clock prescaler)

USART 输入时钟可通过预分频系数进行分频：

0000：输入时钟未分频
0001：输入时钟 2 分频
0010：输入时钟 4 分频
0011：输入时钟 6 分频
0100：输入时钟 8 分频
0101：输入时钟 10 分频
0110：输入时钟 12 分频
0111：输入时钟 16 分频
1000：输入时钟 32 分频
1001：输入时钟 64 分频
1010：输入时钟 128 分频
1011：输入时钟 256 分频
其余组合：保留

注意：为 PRESCALER 编程不允许的值时，编程的预分频值等于 1011，即输入时钟除以 256。

36.8.15 USART 寄存器映射

表 293. USART 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	USART_CR1 FIFO mode enabled	RXFFIE	TXFEIE	FIFOEN	M1	EOBIE	RTOIE	DEAT[4:0]					DEDT[4:0]					OVER8	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXFNFIE	TCIE	RXFNEIE	IDLEIE	TE	RE	UESM	UE
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00	USART_CR1 FIFO mode disabled	Res.	Res.	FIFOEN	M1	EOBIE	RTOIE	DEAT[4:0]					DEDT[4:0]					OVER8	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
	Reset value			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	USART_CR2	ADD[7:0]								RTOEN	ABRMOD[1:0]		ABREN	MSBFIRST	DATAINV	TXINV	RXINV	SWAP	LINEN	STOP [1:0]		CLKEN	CPOL	CPHA	LBCL	Res.	LBDIE	LBDL	ADDM7	DIS_NSS	Res.	Res.	SLVEN
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0			0
0x08	USART_CR3	TXFTCFG[2:0]			RXFTIE[2:0]		RXFTCFG		TCBGTIE	TXFTIE	WUFIE	WUS 1	WUS 0	SCAR CNT [2:0]			Res.	DEP	DEM	DDRE	OVRDIS	ONEBIT	CTSIE	CTSE	RTSE	DMAT	DMAR	SCEN	NACK	HDSEL	IRLP	IREN	EIE
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	USART_BRR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BRR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	USART_GTPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GT[7:0]								PSC[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	USART_RTOR	BLEN[7:0]								RTO[23:0]																							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 293. USART 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x18	USART_RQR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFRQ	RXFRQ	MMRQ	SBKRQ	ABRRQ
	Reset value																												0	0	0	0	0
0x1C	USART_ISR FIFO mode enabled	Res.	Res.	Res.	Res.	TXFT	RXFT	TCBGT	RXFF	TXFE	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY	ABRF	ABRE	UDR	EOBF	RTOF	CTS	CTSIF	LBDF	TXFNF	TC	RXFNE	IDLE	ORE	NE	FE	PE
	Reset value					0	0	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0x1C	USART_ISR FIFO mode disabled	Res.	Res.	Res.	Res.	Res.	Res.	TCBGT	Res.	Res.	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY	ABRF	ABRE	UDR	EOBF	RTOF	CTS	CTSIF	LBDF	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
	Reset value							0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0x20	USART_ICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUCF	Res.	Res.	CMCF	Res.	Res.	Res.	UDRCF	EOBCF	RTOCF	Res.	CTSCF	LBDCF	TCBGTCF	TCCF	TXFECF	IDLECF	ORECF	NECF	FECF	PECF
	Reset value												0			0				0	0	0		0	0	0	0	0	0	0	0	0	0
0x24	USART_RDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDR[8:0]								
	Reset value																								0	0	0	0	0	0	0	0	0
0x28	USART_TDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDR[8:0]								
	Reset value																								0	0	0	0	0	0	0	0	0
0x2C	USART_PRESC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESCALER[3:0]			
	Reset value																													0	0	0	0

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

37 低功耗通用异步收发器 (LPUART)

本节介绍低功耗通用异步收发器 (LPUART)。

37.1 简介

LPUART 是一种 UART，允许在有限功耗下进行双向 UART 通信。仅需32.768 kHz LSE 时钟即可进行高达 9600 波特的 UART 通信。当 LPUART 使用其他不同于 LSE 的时钟源时，可以达到更高的波特率。

即使当微控制器处于低功耗模式，能耗极低时，LPUART 也会等待 UART 帧的到来。LPUART 包含所有必要的硬件支持，使在最小功耗下可以进行异步串行通信。

它支持半双工单线通信和调制解调器操作 (CTS/RTS)，

还支持多处理器通信。

DMA（直接存储器访问）可用于数据发送/接收。

37.2 LPUART 主要特性

- 全双工异步通信
- NRZ 标准格式（标记/空格）
- 可编程波特率
- 使用 32.768 kHz 时钟源时波特率为 300 波特到 9600 波特。
- 使用高频时钟源可实现更高的波特率
- 两个用于收发数据的内部 FIFO
  每个 FIFO 均可由软件使能/禁止，并且均带有用于指示 FIFO 状态的状态标志。
- 双时钟域，带有独立于 PCLK 的外设专用内核时钟。
- 数据字长度可编程（7 位、8 位或 9 位）
- 可编程的数据顺序，最先移位 MSB 或 LSB
- 停止位可配置（支持 1 个或 2 个停止位）
- 单线半双工通信
- 使用 DMA 实现连续通信
- 使用中心化 DMA 在预留的 SRAM 缓冲区中收/发字节。
- 发射器和接收器有单独的使能位
- 发送和接收的单独信号极性控制
- Tx/Rx 引脚可配置互换
- 调制解调器和 RS-485 收发器的硬件流控制
- 传输检测标志：
  - 接收缓冲区已满
  - 发送缓冲区为空
  - BUSY 标志和发送结束标志

- 奇偶校验控制：
  - 发送奇偶校验位
  - 检查接收的数据字节的奇偶性
- 四个错误检测标志：
  - 上溢错误
  - 噪声检测
  - 帧错误
  - 奇偶校验错误
- 具有标志的中断源
- 多处理器通信：从静默模式唤醒（通过空闲线检测或地址标记检测）
- 从停止模式唤醒

37.3 LPUART 实现

下表介绍了 LPUART 实现。其中还包含 USART 和 UART 作为参照。

表 294. STM32H503xx 特性

USART 模式/特性	STM32H503xx
USART1	完全
USART2	完全
USART3	完全
LPUART1	低功耗

表 295. USART/LPUART 特性

USART 模式/特性(1)	全面的功能集	基本的功能集	低功耗功能集
调制解调器的硬件流控制	X	X	X
使用 DMA 进行连续通信	X	X	X
多处理器通信	X	X	X
同步模式（主/从）	X	-	-
智能卡模式	X	-	-
单线半双工通信	X	X	X
IrDA SIR ENDEC 模块	X	X	-
LIN 模式	X	X	-
双时钟域	X	X	X
接收器超时中断	X	X	-
Modbus 通信	X	X	-
自动波特率检测	X	X	-
驱动器使能	X	X	X

表 295. USART/LPUART 特性（续）

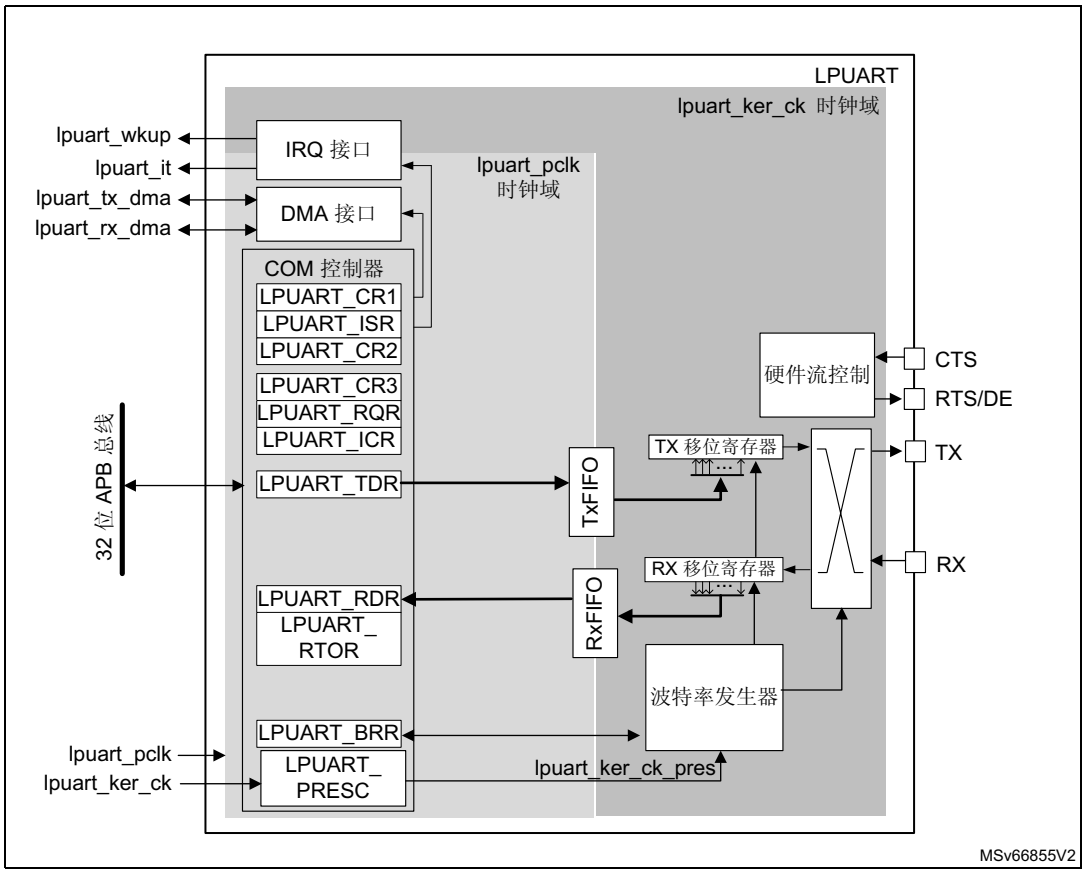
USART 模式/特性(1)	全面的功能集	基本的功能集	低功耗功能集
USART 数据长度	7 位、8 位和 9 位		
Tx/Rx FIFO	X	X	X
Tx/Rx FIFO 大小	8		
从低功耗模式唤醒	X(2)	X(2)	X(2)

1. X = 支持。
2. 支持从停止模式唤醒。

37.4 LPUART 功能描述

37.4.1 LPUART 框图

图 452. LPUART 框图



37.4.2 LPUART 引脚和内部信号

LPUART 输入/输出引脚说明

- LPUART 双向通信

  LPUART 双向通信需要至少两个引脚：接收数据输入引脚 (RX) 和发送数据输出引脚 (TX)：

  – **RX**（接收数据输入引脚）：

  RX 为串行数据输入引脚。

  – **TX**（发送数据输出引脚）

  如果关闭发送器，该输出引脚模式由其 I/O 端口配置决定。如果使能了发送器但没有待发送的数据，则 TX 引脚处于高电平。在单线模式下，该 I/O 用于发送和接收数据。

- RS232 硬件流控制模式

  在 RS232 硬件流控制模式下需要以下引脚：

  – **CTS**（清除以发送）

  如果驱动为高电平，则该信号用于在当前传输结束时阻止数据发送。

  – **RTS**（请求以发送）

  如果为低电平，则该信号用于指示 USART 已准备好接收数据。

- RS485 硬件流控制模式

  在 RS485 硬件控制模式下需要 **DE**（驱动器使能）引脚。该信号用于激活外部收发器的发送模式。

有关 LPUART 输入/输出引脚和内部信号的列表，请参见*表 296* 和*表 297*。

表 296. LPUART 输入/输出引脚

引脚名称	信号类型	说明
LPUART_RX	输入	串行数据接收输入。
LPUART_TX	输出	发送数据输出引脚。
LPUART_CTS	输入	清除以发送
LPUART_RTS	输出	请求以发送
LPUART_DE(1)	输出	驱动器使能

1. LPUART_DE 和 LPUART_RTS 共用同一个引脚。

LPUART 输入/输出信号说明

表 297. LPUART 内部输入/输出信号

引脚名称	信号类型	说明
usart_pclk	输入	APB 时钟
lpuart_ker_ck	输入	LPUART 内核时钟
lpuart_wkup	输出	LPUART 提供唤醒中断
lpuart_it	输出	LPUART 全局中断

表 297. LPUART 内部输入/输出信号

引脚名称	信号类型	说明
lpuart_tx_dma	输入/输出	LPUART 发送 DMA 请求
lpuart_rx_dma	输入/输出	LPUART 接收 DMA 请求

37.4.3 LPUART 时钟

图 452 中的简化框图显示的是两个完全独立的时钟域：

- **lpuart_pclk** 时钟域

  lpuart_pclk 时钟信号馈送外设总线接口。需要访问 LPUART 寄存器时，该信号必须有效。

- **lpuart_ker_ck** 内核时钟域

  lpuart_ker_ck 是 LPUART 时钟源。它独立于 **lpuart_pclk**，由 RCC 提供。因此，即使 **lpuart_ker_ck** 停止，也可以对 LPUART 寄存器执行读/写操作。

  不支持双时钟域功能时，**lpuart_ker_ck** 与 lpuart_pclk 时钟相同。

lpuart_pclk 和 **lpuart_ker_ck** 之间无任何限制：**lpuart_ker_ck** 既可快于也可慢于 **lpuart_pclk**，唯一的限制是软件以足够快的速度管理通信的能力。

37.4.4 LPUART 字符说明

可通过对 LPUART_CR1 寄存器中的 M 位（M0：位 12，M1：位 28）进行编程来将字长设置为 7 位、8 位或 9 位（请参见*图 426*）。

- 7 位字符长度：M[1:0] = 10
- 8 位字符长度：M[1:0] = 00
- 9 位字符长度：M[1:0] = 01

在默认情况下，信号（TX 或 RX）在起始位工作期间处于低电平状态。在停止位工作期间处于高电平状态。

通过极性配置控制，可以单独针对每个信号对这些值取反。

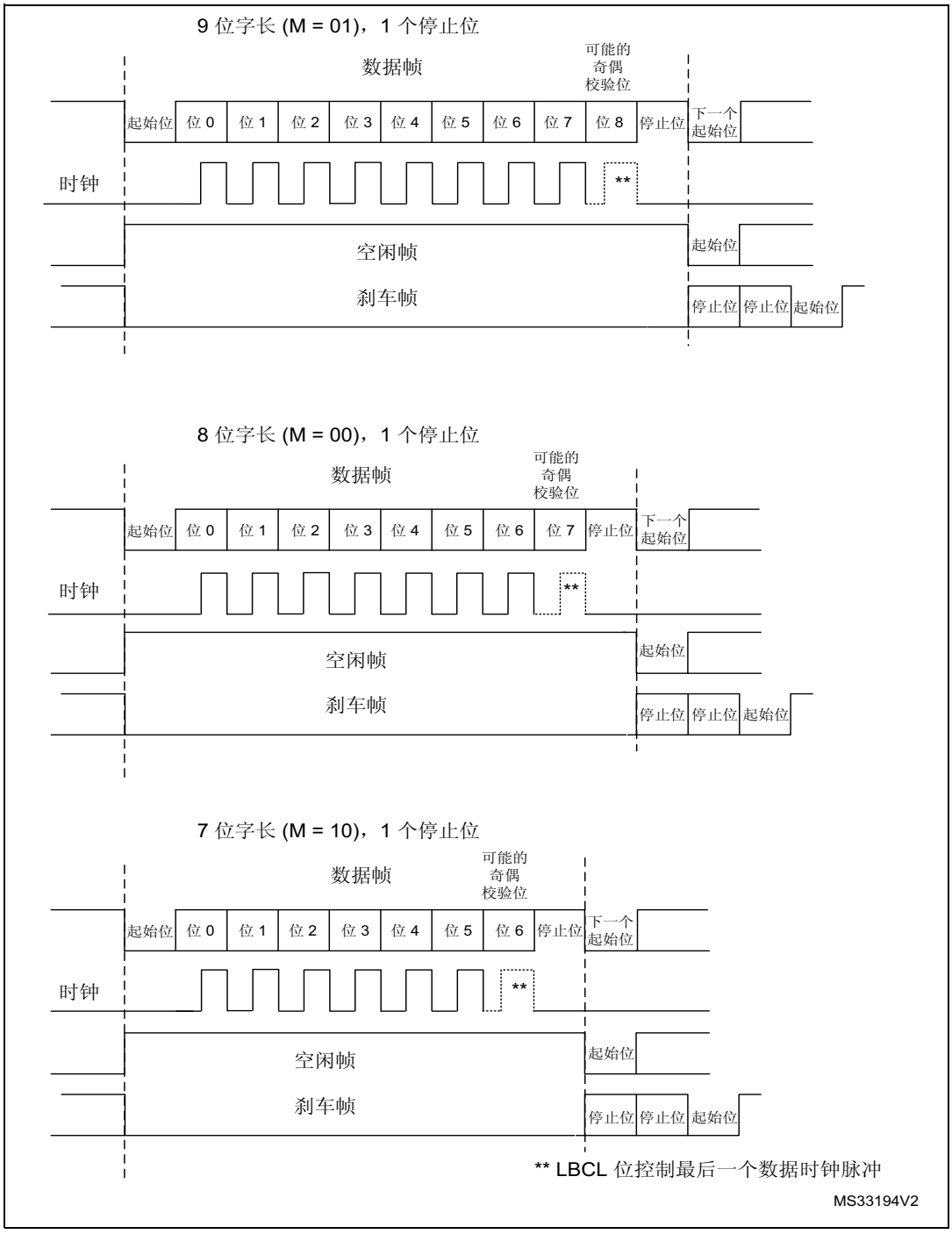
空闲字符可理解为整个帧周期内电平均为“1”（停止位的电平也是“1”）。

刹车字符可理解为在一个帧周期内接收到的电平均为“0”。发送器在刹车帧的末尾插入 2 个停止位。

发送和接收操作由通用波特率发生器驱动。发送器和接收器的使能位置 1 时，将分别生成发送时钟和接收时钟。

下面给出了各个块的详细信息。

图 453. LPUART 字长编程



37.4.5 LPUART FIFO 和阈值

LPUART 可工作在 FIFO 模式下。

LPUART 具有一个发送 FIFO (TXFIFO) 和一个接收 FIFO (RXFIFO)。可通过将 LPUART_CR1 寄存器中的 FIFOEN 位（位 29）置 1 使能 FIFO 模式。

最大数据字长度为 9 位，因此 TXFIFO 为 9 位宽。不过，RXFIFO 的默认宽度为 12 位。这是因为接收器不仅在 FIFO 中存储数据，而且还存储与每个字符相关的错误标志（奇偶校验错误、噪声错误和帧错误标志）。

注意： *接收的数据与相应的标志一起存储在 RXFIFO 中，但读取 RDR 时仅读取数据。*

状态标志位于 LPUART_ISR 寄存器中。

可以定义触发 Tx 和 RX 中断的 TXFIFO 和 RXFIFO 阈值。这些阈值通过 LPUART_CR3 控制寄存器中的 RXFTCFG 和 TXFTCFG 位域进行编程。

在这种情况下：

- 当 RXFIFO 中接收到的数据量达到 RXFTCFG 位域中编程的阈值时，会生成 Rx 中断。此时，LPUART_ISR 寄存器中的 RXFT 标志置 1。这意味着已接收到 RXFTCFG 数据：LPUART_RDR 中有 1 个数据，RXFIFO 中有 (RXFTCFG - 1) 个数据。例如，如果将 RXFTCFG 编程为“101”，则在接收到对应于 FIFO 大小的数据量时，RXFT 标志将置 1：RXFIFO 中有（FIFO 大小 - 1）个数据，LPUART_RDR 中有 1 个数据。因此，下一个接收到的数据不会将上溢标志置 1。
- 当 TXFIFO 中的空位置数达到在 TXFTCFG 位域中编程的阈值时，会生成 Tx 中断。

37.4.6 LPUART 发送器

发送器可发送 7 位、8 位或 9 位的数据字，具体取决于 M 位的状态。要激活发送器功能，必须将发送使能位 (TE) 置 1。发送移位寄存器中的数据输出到 TX 引脚。

字符发送

LPUART 发送期间，首先通过 TX 引脚移出数据的最低有效位（默认配置）。该模式下，LPUART_TDR 寄存器的缓冲区 (TDR) 位于内部总线和发送移位寄存器之间（请参见 *图 452*）。

使能 FIFO 模式时，写入到 LPUART_TDR 寄存器中的数据会在 TXFIFO 中排队。

每个字符前面都有一个起始位，其对应于一个位周期的逻辑低电平。字符由可配置数量的停止位终止。

停止位的数量可为 1 或 2。

注意： *向 LPUART_TDR 中写入要发送的数据前，TE 位必须先置 1。*

数据发送期间不得复位 TE 位。发送期间复位 TE 位会冻结波特率计数器，进而损坏 TX 引脚上的数据。当前发送的数据将会丢失。

使能 TE 位后，将发送空闲帧。

可配置的停止位

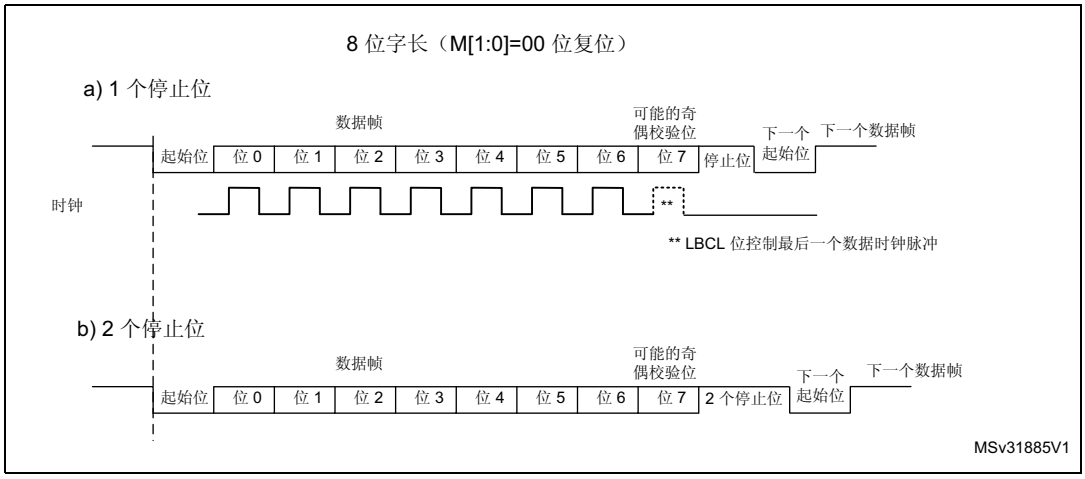
可以在 LPUART_CR2 的位 13 和位 12 中编程将随各个字符发送的停止位的数量。

- ***1 个停止位***：这是停止位数量的默认值。
- ***2 个停止位***：正常 LPUART 模式、单线模式和调制解调器模式支持该值。

空闲帧发送包括停止位。

刹车发送是 10 个低电平位（M[1:0] = 00 时）、11 个低电平位（M[1:0] = 01 时）或 9 个低电平位（M[1:0] = 10 时），然后是 2 个停止位。无法传送长刹车（刹车长度大于 9/10/11 个低电平位）。

图 454. 可配置的停止位



字符发送步骤

要发送字符，需遵循以下步骤：

1. 对 LPUART_CR1 中的 M 位进行编程以定义字长。
2. 使用 LPUART_BRR 寄存器选择所需波特率。
3. 对 LPUART_CR2 中的停止位数量进行编程。
4. 通过向 LPUART_CR1 寄存器中的 UE 位写入 1 使能 LPUART。
5. 如果将进行多缓冲区通信，请选择 LPUART_CR3 中的 DMA 使能 (DMAT)。按照*第 37.4.13 节：使用 DMA 和 LPUART 进行连续通信*中的说明配置 DMA 寄存器。
6. 将 LPUART_CR1 中的 TE 位置 1 以便在首次发送时发送一个空闲帧。
7. 在 LPUART_TDR 寄存器中写入要发送的数据。为每个要在单缓冲区模式下发送的数据重复该操作。
   - 禁止 FIFO 模式时，向 LPUART_TDR 写入数据会将 TXE 标志清零。
   - 使能 FIFO 模式时，向 LPUART_TDR 写入数据会向 TXFIFO 中增添一个数据。当 TXFNF 标志置 1 时，对 LPUART_TDR 寄存器的写操作可以执行。该标志会保持置 1，直到 TXFIFO 已满。
8. 将最后一个数据写入 LPUART_TDR 寄存器后，等待 TC =1。这表明最后一个帧的传送已完成。
   - 禁止 FIFO 模式时，这表示最后一个帧的发送已完成。
   - 使能 FIFO 模式时，这表示 TXFIFO 和移位寄存器均为空。

   当 LPUART 被禁止或进入暂停模式时，需要执行此检查来避免损坏最后一次发送。

单字节通信

- 禁止 FIFO 模式时：

  对发送数据寄存器进行写操作始终会清零 TXE 位。TXE 标志由硬件置 1 以指示：

  – 数据已从 LPUART_TDR 寄存器移到移位寄存器中且数据发送已开始；
  – LPUART_TDR 寄存器为空；
  – LPUART_TDR 寄存器中可写入下一个数据，而不会覆盖前一个数据。

  当 TXEIE 位置 1 时，TXE 标志会生成中断。

  发送时，要传入 LPUART_TDR 寄存器的写指令中存有 TDR 寄存器中的数据，该数据将在当前发送结束时复制到移位寄存器中。

  未发送时，要传入 LPUART_TDR 寄存器的写指令会将数据置于移位寄存器中，数据发送开始时，TXE 位置 1。

- 使能 FIFO 模式时，TXFNF（TXFIFO 未满）标志由硬件置 1，以指示：
  – TXFIFO 未满；
  – LPUART_TDR 寄存器为空；
  – LPUART_TDR 寄存器中可写入下一个数据，而不会覆盖前一个数据。发送时，写 LPUART_TDR 寄存器会将数据存储在 TXFIFO。该数据随后会在当前发送结束时从 TXFIFO 复制到移位寄存器中。

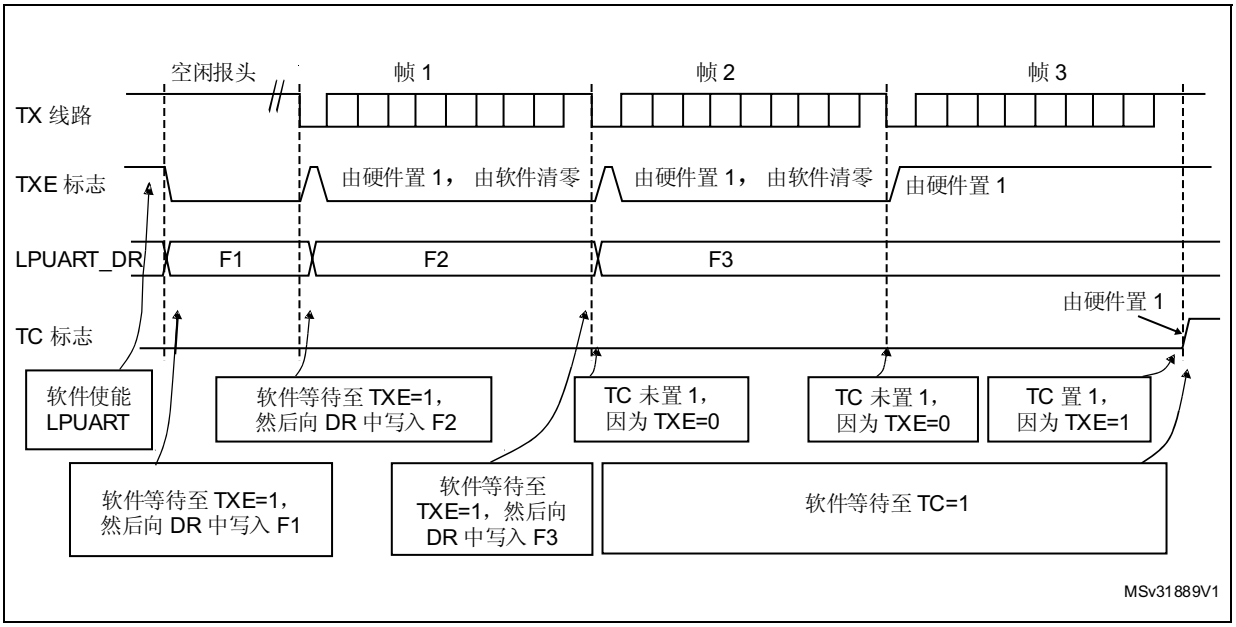
  TXFIFO 未满时，即使在对 LPUART_TDR 寄存器执行完写操作后，TXFNF 标志也保持为 1。该标志在 TXFIFO 已满时清零。TXFNEIE 位置 1 时，该标志会生成中断。

  或者，当达到 TXFIFO 阈值时，会生成中断并将数据写入 TXFIFO。在这种情况下，CPU 可写入由编程的阈值定义的数据块。

  如果帧已发送（停止位后）且 TXE 标志（FIFO 模式下的 TXFE）置 1，TC 位将变为高电平。如果 LPUART_CR1 寄存器中的 TCIE 位置 1，将生成中断。

  向 LPUART_TDR 寄存器中写入最后一个数据后，必须等待至 TC=1，之后才可禁止 LPUART 或使微控制器进入低功率模式（请参见*图 455：发送时的 TC/TXE 行为*）。

图 455. 发送时的 TC/TXE 行为



注意：　使能 FIFO 管理时，TXFNF 标志将用于数据发送。

刹车字符

将 SBKRQ 位置 1 将发送一个刹车字符。刹车帧的长度取决于 M 位（请参见*图 453*）。

如果将 1 写入 SBKRQ 位，则当前字符发送完成后，将在 TX 线路上发送一个刹车字符。通过写操作将 SBKF 位置 1 并在刹车字符发送完成时（发送刹车字符后的停止位期间），该位由硬件复位。LPUART 在刹车帧末尾的两位持续时间内插入一个逻辑“1”信号 (STOP)，以确保识别下个帧的起始位。

如果 SBKRQ 位置 1，则在当前发送结束时，会发送一个刹车字符。

使能 FIFO 模式时，即使 TXFIFO 已满，发送刹车字符的优先级也仍高于发送数据的优先级。

空闲字符

将 TE 位置 1 会驱动 LPUART 在第一个数据帧之前发送一个空闲帧。

37.4.7 LPUART 接收器

LPUART 可接收 7 位、8 位或 9 位的数据字，具体取决于 LPUART_CR1 寄存器中的 M 位。

起始位检测

在 LPUART 中，先在 Rx 线路上出现下降沿时检测起始位，然后在起始位中间采样以确认电平是否仍为 0。如果起始采样为 1，则噪声错误标志 (NE) 置 1，起始位将被丢弃，接收器将等待新的起始位。否则，接收器将继续正常采样所有传入位。

字符接收

LPUART 接收期间，首先通过 RX 引脚移入数据的最低有效位（默认配置）。该模式下，LPUART_RDR 寄存器的缓冲区 (RDR) 位于内部总线和接收移位寄存器之间。

字符接收步骤

要接收字符，需遵循以下步骤：

1. 对 LPUART_CR1 中的 M 位进行编程以定义字长。
2. 使用波特率寄存器 LPUART_BRR 选择所需波特率
3. 对 LPUART_CR2 中的停止位数量进行编程。
4. 通过向 LPUART_CR1 寄存器中的 UE 位写入 1 使能 LPUART。
5. 如果将进行多缓冲区通信，请选择 LPUART_CR3 中的 DMA 使能 (DMAR)。按照*第 37.4.13 节：使用 DMA 和 LPUART 进行连续通信*中的说明配置 DMA 寄存器。
6. 将 RE 位 LPUART_CR1 置 1。这一操作将使能接收器开始搜索起始位。

接收到字符时

- 禁止 FIFO 模式时，RXNE 位置 1。这表明移位寄存器的内容已传送到 RDR。也就是说，已接收到并可读取数据（以及其相应的错误标志）。
- 如果已使能 FIFO 模式，则 RXFNE 位置 1，这表示 RXFIFO 非空。读取 LPUART_RDR 会返回输入到 RXFIFO 中的最早数据。接收到数据时，数据以及相应的错误位将一起被存储在 RXFIFO 中。
- 如果 RXNEIE（FIFO 模式下时为 RXFNEIE）位置 1，则会生成中断。
- 如果接收期间已检测到帧错误、噪声错误或上溢错误，错误标志位可置 1。

- 在多缓冲区通信模式下：
  - 禁止 FIFO 模式时，每接收到一个字节后 RXNE 标志均置 1，然后通过 DMA 对接收数据寄存器执行读操作清零。
  - 如果使能 FIFO 模式，则 RXFNE 标志在 RXFIFO 非空时置 1。每次收到 DMA 请求后，都会从 RXFIFO 取回数据。在 RXFIFO 非空时（即，当 RXFIFO 中存在要读取的数据时），会触发 DMA 请求。
- 在单缓冲区模式下：
  - 如果禁止 FIFO 模式，则通过软件对 LPUART_RDR 寄存器进行读操作来将 RXNE 标志清零。也可以通过将 LPUART_RQR 寄存器中的 RXFRQ 位置 1 将 RXNE 标志位清零。RXNE 位必须在结束接收下一个字符前清零，以避免发生上溢错误。
  - 如果使能 FIFO 模式，则 RXFNE 标志在 RXFIFO 非空时置 1。每次对 LPUART_RDR 寄存器执行完读操作后，都会从 RXFIFO 中取回数据。RXFIFO 为空时，RXFNE 标志将清零。也可以通过将 LPUART_RQR 寄存器中的 RXFRQ 位置 1 将 RXFNE 标志清零。RXFIFO 已满时，必须在结束接收下一个字符前读取 RXFIFO 中的第一个条目，以避免发生上溢错误。当 RXFNEIE 位置 1 时，RXFNE 标志会生成中断。或者，当达到 RXFIFO 阈值时，会生成中断并从 RXFIFO 中读取数据。在这种情况下，CPU 可读取由编程的阈值定义的数据块。

刹车字符

接收到刹车字符时，USART 将会按照帧错误对其进行处理。

空闲字符

检测到空闲帧时，除了在 IDLEIE 位置 1 时会生成中断外，处理步骤与接收到数据字符的情况相同。

上溢错误

- 禁止 FIFO 模式

  如果在 RXNE 未复位时接收到字符，则会发生上溢错误。

  RXNE 位清零前，数据无法从移位寄存器传送到 RDR 寄存器。每接收到一个字节后，RXNE 标志位都将置 1。

  当 RXNE 标志位是 1 时，如果在接收到下一个数据或尚未处理上一个 DMA 请求，则会发生上溢错误。发生上溢错误时：
  - ORE 位置 1。
  - RDR 中的内容不会丢失。对 LPUART_RDR 执行读操作时可使用先前的数据。
  - 移位寄存器会被覆盖。之后，上溢期间接收到的任何数据都将丢失。
  - 如果 RXNEIE 位置 1 或 EIE 位置 1，则会生成中断。
- 使能 FIFO 模式

  移位寄存器准备好传送且接收 FIFO 已满时，会发生上溢错误。

  在 RXFIFO 中出现一个空闲位置之前，数据无法从移位寄存器传送到 LPUART_RDR 寄存器。当 RXFIFO 非空时，RXFNE 标志置 1。

  如果 RXFIFO 已满且移位寄存器已准备好传送，则会发生上溢错误。发生上溢错误时：
  - ORE 位置 1。
  - RXFIFO 中的第一个条目不会丢失。可通过对 LPUART_RDR 执行读操作来获取。
  - 移位寄存器会被覆盖。之后，上溢期间接收到的任何数据都将丢失。
  - 如果 RXFNEIE 位置 1 或 EIE 位置 1，则会生成中断。

通过将 ICR 寄存器中的 ORECF 位置 1 来复位 ORE 位。

注意： *ORE 位置 1 时表示至少 1 个数据丢失。T*

禁止 FIFO 模式时，有以下两种可能

- *如果 RXNE = 1，则最后一个有效数据存储于接收寄存器 (RDR) 中并且可进行读取，*
- *如果 RXNE=0，则最后一个有效数据已被读取，因此 RDR 中没有要读取的数据。接收到新（丢失）数据的同时已读取 RDR 中的最后一个有效数据时，会发生该情况。*

选择时钟源

时钟源的选择通过时钟控制系统完成（请参见*复位和时钟控制器 (RCC) 部分*）。在使能 LPUART 之前，必须通过 UE 位选择时钟源。

必须遵循以下两个条件选择时钟源：

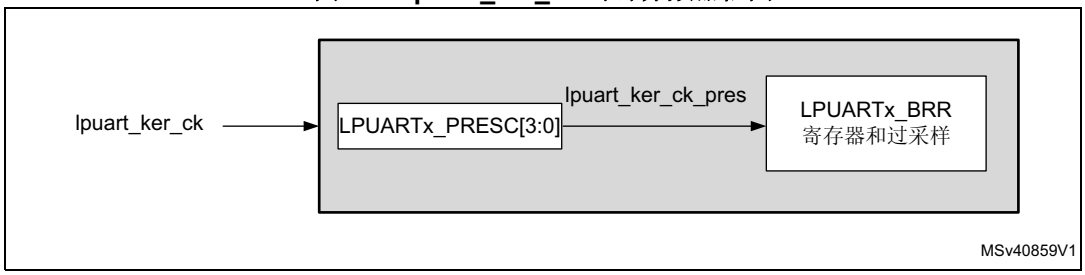
- 可在低功耗模式下使用 LPUART
- 通信速度。

时钟源频率为 lpuart_ker_ck。

如果支持双时钟域和从低功耗模式唤醒功能，则 lpuart_ker_ck 时钟源可在 RCC 中进行配置（请参见*复位和时钟控制器 (RCC)* 部分）。否则，lpuart_ker_ck 与 lpuart_pclk 相同。

lpuart_ker_ck 可根据 LPUART_PRESC 寄存器中的可编程系数进行分频。

图 456. lpuart_ker_ck 时钟分频器框图



某些 lpuart_ker_ck 时钟源允许 LPUART 在 MCU 处于低功耗模式时接收数据。必要时，LPUART 可基于所接收的数据和唤醒模式的选择来唤醒 MCU，以通过用软件读取 LPUART_RDR 寄存器的方式或通过 DMA 的方式传输已接收数据。

对于其他时钟源，系统必须激活才能使能 LPUART 通信。

时钟源还决定通信速度范围（尤其是最大通信速度）。

接收器在尽可能靠近位周期中间的位置采样每个传入位。每个传入位仅进行一次采样。

注意： *不进行数据噪声检测。*

帧错误

如果接收数据时未在预期时间内识别出停止位，从而出现同步失效或过度的噪声，则会检测到帧错误。

检测到帧错误时：

- FE 位由硬件置 1
- 无效数据从移位寄存器传送到 LPUART_RDR 寄存器。
- 单字节通信时无中断产生。然而，在 RXNE 位产生中断时，该位出现上升沿。多缓冲区通信时，LPUART_CR3 寄存器中的 EIE 位置 1 时会发出中断。

通过将 1 写入 LPUART_ICR 寄存器中的 FECF 位来复位 FE 位。

接收期间可配置的停止位

可通过 LPUART_CR2 的控制位配置要接收的停止位的数量：可以是 1 个或 2 个（正常模式下）。

- **1 个停止位**：将在第 8、第 9 和第 10 次采样时对 1 个停止位进行采样。
- **2 个停止位**：将在第二个停止位的中间对 2 个停止位进行采样。RXNE 和 FE 标志在此采样期间（例如，在第二个停止位期间）置 1。发生帧错误时不检测第一个停止位。

37.4.8 LPUART 波特率生成

接收器和发送器（Rx 和 Tx）的波特率均设置为 LPUART_BRR 寄存器中编程的值。

$$\text{Tx/Rx 波特率} = \frac{256 \times \text{lpuart_ker_ck_pres}}{\text{LPUARTDIV}}$$

LPUARTDIV 在 LPUART_BRR 寄存器中定义。

注意： *对 LPUART_BRR 执行写操作后，波特率计数器更新为波特率寄存器中的新值。因此，波特率寄存器的值不得在通信时发生更改。*

禁止在 LPUART_BRR 寄存器中写入小于 0x300 的值。

f_{CK} *必须介于 3 倍波特率到 4096 倍波特率之间。*

LPUART 时钟源为 LSE 时可达到的最大波特率为 9600 波特。当 LPUART 使用其他不同于 LSE 的时钟源时，可以达到更高的波特率。例如，如果 LPUART 时钟源频率为 100 MHz，则可达到的最大波特率约为 33 M 波特。

表 298. lpuart_ker_ck_pres = 32.768 kHz 时编程的波特率的误差计算

波特率		lpuart_ker_ck_pres = 32.768 kHz		
序号	期望值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 期望值) / 所需波特率
1	0.3 k 波特	0.3 k 波特	0x6D3A	0
2	0.6 k 波特	0.6 k 波特	0x369D	0
3	1200 波特	1200.087 波特	0x1B4E	0.007
4	2400 波特	2400.17 波特	0xDA7	0.007
5	4800 波特	4801.72 波特	0x6D3	0.035
6	9600 k 波特	9608.94 波特	0x369	0.093

表 299. f_{CK} = 100 MHz 时编程的波特率的误差计算

波特率		f_{CK} = 100 MHz		
序号	期望值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 期望值) / 所需波特率
1	38400 波特	38400,04 波特	A2C2A	0,0001
2	57600 波特	57600,06 波特	6C81C	0,0001
3	115200 波特	115200,12 波特	3640E	0,0001
4	230400 波特	230400,23 波特	1B207	0,0001
5	460800 波特	460804,61 波特	D903	0,001
6	921600 波特	921625,81 波特	6C81	0,0028
7	4000 k 波特	4000000,00 波特	1900	0
8	10000 k 波特	10000000,00 波特	A00	0
9	20000 k 波特	20000000,00 波特	500	0
10	30000 k 波特	33032258,06 波特	307	0,1

37.4.9 LPUART 接收器对时钟偏差的容差

仅当总时钟系统偏差小于 LPUART 接收器的容差时，LPUART 异步接收器才能正常工作。影响总偏差的因素包括：

- DTRA：发送器误差引起的偏差（其中还包括发送器本地振荡器的偏差）
- DQUANT：接收器的波特率量化引起的误差
- DREC：接收器本地振荡器的偏差
- DTCL：传输线路引起的偏差（通常是由于收发器所引起，它可能会在低电平到高电平转换时序与高电平到低电平转换时序之间引入不对称）

DTRA + DQUANT + DREC + DTCL + DWU < LPUART 接收器的容差

其中

DWU 是使用从低功耗模式唤醒时因采样点偏差而产生的误差。

M[1:0] = 01 时：

$$DWU = \frac{t_{WULPUART}}{11 \times Tbit}$$

M[1:0] = 00 时：

$$DWU = \frac{t_{WULPUART}}{10 \times Tbit}$$

M[1:0] = 10 时：

$$DWU = \frac{t_{WULPUART}}{9 \times Tbit}$$

$t_{WULPUART}$ 是检测到起始位下降沿与时钟（由外设请求）就绪、达到外设且调压器就绪之间的时间。

LPUART 接收器在*表 300* 中指定的最大容许偏差下可正确接收数据：

- 通过 LPUART_CR2 寄存器中的 STOP[1:0] 位定义的停止位数
- LPUART_BRR 寄存器值。

表 300. LPUART 接收器的容差

M 位	768 < BRR < 1024	1024 < BRR < 2048	2048 < BRR < 4096	4096 ≤ BRR
8 位 (M=00)，1 个停止位	1.82%	2.56%	3.90%	4.42%
9 位 (M=01)，1 个停止位	1.69%	2.33%	2.53%	4.14%
7 位 (M=10)，1 个停止位	2.08%	2.86%	4.35%	4.42%
8 位 (M=00)，2 个停止位	2.08%	2.86%	4.35%	4.42%
9 位 (M=01)，2 个停止位	1.82%	2.56%	3.90%	4.42%
7 位 (M=10)，2 个停止位	2.34%	3.23%	4.92%	4.42%

注意：当接收的帧恰好包含 10 个（M 位 = 00）、11 个（M 位 = 01）或 9 个（M 位 = 10）位时间的空闲帧时，表 300 中指定的数据可能与特例中的数据略微不同。

37.4.10 LPUART 多处理器通信

可以执行 LPUART 多处理器通信（多个 LPUART 连接在一个网络中）。例如，其中一个 LPUART 可以是主设备，其 TX 输出与其他 LPUART 的 RX 输入相连接。其他 LPUART 为从设备，其各自的 TX 输出在逻辑上通过与运算连在一起，并与主 LPUART 的 RX 输入相连。

在多处理器配置中，理想情况下通常只有预期的消息接收方主动接收完整的消息内容，从而减少由所有未被寻址的接收器造成的冗余 LPUART 服务开销。

可通过静默功能将未被寻址的器件置于静默模式下。为了使用静默模式功能，必须将 LPUART_CR1 寄存器中的 MME 位置 1。

注意： *使能 FIFO 管理且 MME 已置 1 时，不得清零 MME 位再快速将其置 1（在两个* lpuart_ker_ck *周期内），否则静默模式可能保持有效。*

使能静默模式时：

- 不得将接收状态位置 1。
- 禁止任何接收中断。
- LPUART_ISR 寄存器中的 RWU 位置 1。在某些情况下，RWU 可以由硬件或软件通过 LPUART_RQR 寄存器中的 MMRQ 位自动控制。

根据 LPUART_CR1 寄存器中 WAKE 位的设置，LPUART 可使用以下两种方法进入或退出静默模式：

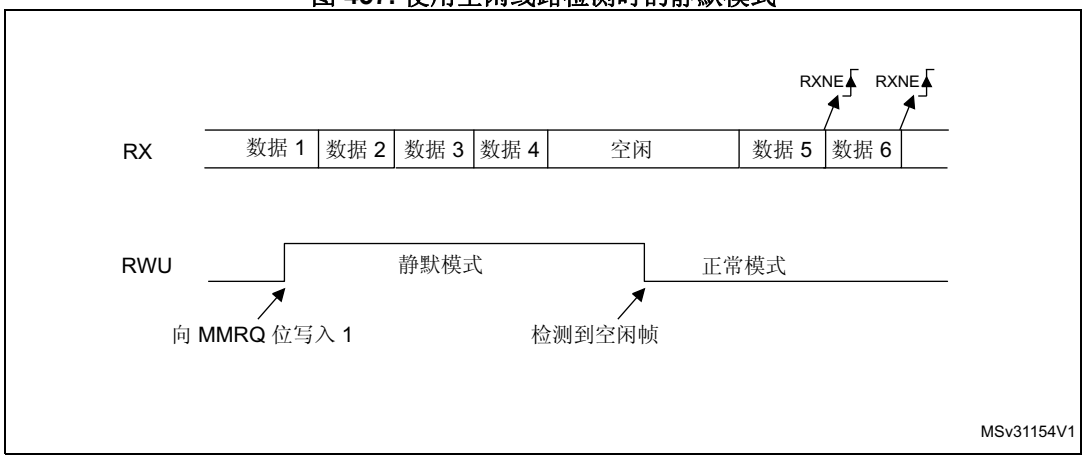
- 如果 WAKE 位被复位，则进行空闲线路检测，
- 如果 WAKE 位置 1，则进行地址标记检测。

空闲线路检测 (WAKE=0)

当向 MMRQ 位写入 1 且 RWU 位自动置 1 时，LPUART 进入静默模式。

检测到空闲帧时，LPUART 将唤醒。此时 RWU 位会由硬件清零，但 LPUART_ISR 寄存器中的 IDLE 位不会置 1。*图 457* 中给出了使用空闲线路检测时静默模式行为的示例。

图 457. 使用空闲线路检测时的静默模式



注意： *如果在 IDLE 字符已经过去时将 MMRQ 位置 1，则不会进入静默模式（RWU 未置 1）。*

如果在线路处于空闲状态时激活 LPUART，在一个 IDLE 帧持续时间后（不只在接收一个字符帧后）会检测到空闲状态。

4 位/7 位地址标记检测 (WAKE=1)

在此模式下，如果字节的 MSB 为 1，则将这些字节识别为地址，否则将其识别为数据。在地址字节中，目标接收器的地址位于 4 个或 7 个 LSB 中。7 位或 4 位地址检测通过 ADDM7 位来选择。接收器会将此 4 位/7 位字与其地址进行比较，该接收器的地址在 LPUART_CR2 寄存器的 ADD 位中进行设置。

注意： *在 7 位和 9 位数据模式下，地址检测分别在 6 位和 8 位地址上完成（ADD[5:0] 和 ADD[7:0]）。*

当接收到与其编程地址不匹配的地址字符时，LPUART 会进入静默模式。此时，RWU 位将由硬件置 1。LPUART 进入静默模式后，RXNE 标志不会针对此地址字节置 1，也不会发出中断或 DMA 请求。

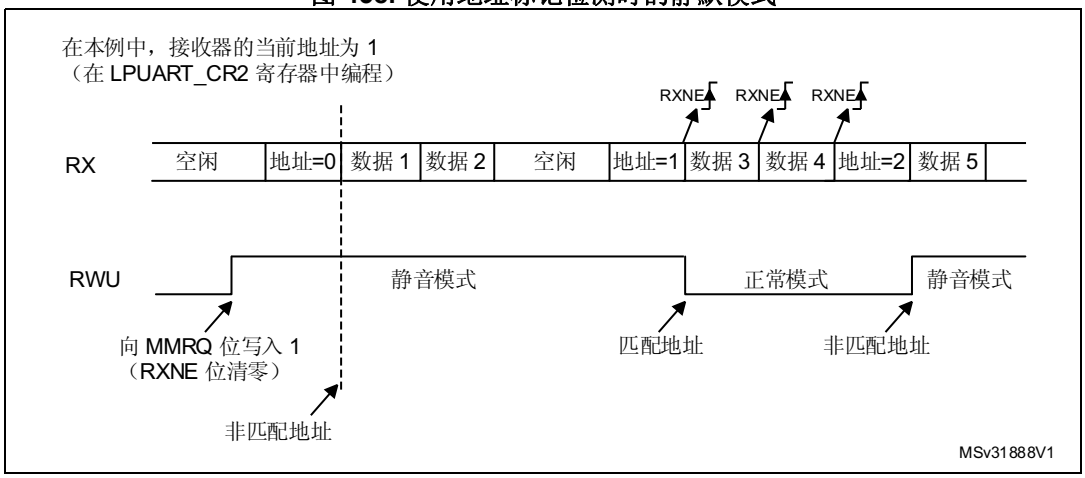
当向 MMRQ 位写入 1 时，LPUART 也会进入静默模式。这种情况下，RWU 位也自动置 1。

当接收到与编程地址匹配的地址字符时，LPUART 会退出静默模式。然后 RWU 位被清零，可以开始正常接收后续字节。由于 RWU 位已清零，RXNE/RXFNE 位会针对地址字符置 1。

注意： *使能 FIFO 管理时，如果在接收器对数据的最后一位进行采样时 MMRQ 位置 1，则可在有效进入静默模式之前接收该数据。*

图 458 中给出了使用地址标记检测时静默模式行为的示例。

图 458. 使用地址标记检测时的静默模式



37.4.11 LPUART 极性控制

将 LPUART_CR1 寄存器中的 PCE 位置 1，可以使能奇偶校验控制（发送时生成奇偶校验位，接收时进行奇偶校验检查）。根据 M 位定义的帧长度，*表 301* 中列出了可能的 LPUART 帧格式。

表 301：LPUART 帧格式

M 位	PCE 位	LPUART 帧(1)
00	0	| SB | 8 位数据 | STB |
00	1	| SB | 7 位数据 | PB | STB |
01	0	| SB | 9 位数据 | STB |
01	1	| SB | 8 位数据 PB | STB |

表 301：LPUART 帧格式（续）

M 位	PCE 位	LPUART 帧(1)
10	0	| SB | 7 位数据 | STB |
10	1	| SB | 6 位数据 | PB | STB |

1. 图注：SB：起始位，STB：停止位，PB：奇偶校验位。
2. 在数据寄存器中，PB 始终位于 MSB 位置（第 8 位或第 7 位，具体取决于 M 位的值）。

偶校验

对奇偶校验位进行计算，使帧和奇偶校验位中“1”的数量为偶数（帧由 6 个、7 个或 8 个 LSB 位组成，具体取决于 M 位的值）。

例如，如果数据 = 00110101 且 4 个位置 1，则在选择偶校验（LPUART_CR1 寄存器中的 PS 位 = 0）时，奇偶校验位等于 0。

奇校验

对奇偶校验位进行计算，使帧和奇偶校验位中“1”的数量为奇数（帧由 6 个、7 个或 8 个 LSB 位组成，具体取决于 M 位的值）。

例如，如果数据 = 00110101 且 4 个位置 1，则在选择奇校验（LPUART_CR1 寄存器中的 PS 位 = 1）时，奇偶校验位等于 1。

接收时进行奇偶校验检查

如果奇偶校验检查失败，则 LPUART_ISR 寄存器中的 PE 标志置 1；如果 LPUART_CR1 寄存器中 PEIE 位置 1，则会生成中断。通过软件将 1 写入 LPUART_ICR 寄存器中的 PECF 位来清零 PE 标志。

发送时的奇偶校验生成

如果 LPUART_CR1 寄存器中的 PCE 位置 1，则在数据寄存器中所写入数据的 MSB 位会进行传送，但是会由奇偶校验位进行更改（如果选择偶校验 (PS=0)，则“1”的数量为偶数；如果选择奇校验 (PS=1)，则“1”的数量为奇数）。

37.4.12 LPUART 单线半双工通信

通过将 LPUART_CR3 寄存器中的 HDSEL 位置 1 来选择单线半双工模式。

LPUART 可以配置为遵循单线半双工协议，其中 TX 和 RX 线路从内部相连接。使用 LPUART_CR3 寄存器中的控制位 HDSEL 可在半双工通信和全双工通信间进行选择。

一旦向 HDSEL 位写入 1：

- TX 和 RX 线路从内部相连接。
- 不能再使用 RX 引脚
- 无数据传输时，TX 引脚始终处于释放状态。因此，它在空闲状态或接收过程中用作标准 I/O。这意味着，必须对 I/O 进行配置，以便将 TX 配置为复用功能开漏并外接上拉电阻。

除此之外，通信协议与正常 LPUART 模式下的通信协议相似。此线路上的任何冲突都必须由软件管理（例如，使用中央仲裁器）。尤其要注意，发送过程永远不会被硬件阻塞，只要数据是在 TE 位置 1 的情况下写入，发送就会持续进行。

注意： *在 LPUART 通信中，当进行 1 个停止位配置时，RXNE 标志在停止位中间置 1。*

37.4.13 使用 DMA 和 LPUART 进行连续通信

LPUART 能够使用 DMA 进行连续通信。接收缓冲区和发送缓冲区的 DMA 请求是独立的。

注意： *要确定是否支持 DMA 模式，请参见第 1253 页的第 36.4 节：USART 实现。如果不支持 DMA，请按照第 36.5.7 节中的说明使用 LPUSRT。当 FIFO 禁止时，可以将 LPUART_ISR 寄存器中的 TXE/RXNE 标志清零，从而实现连续通信。*

使用 DMA 进行发送

将 LPUART_CR3 寄存器中的 DMAT 位置 1 可以使能 DMA 模式进行发送。当 TXE 标志（使能 FIFO 模式时为 TXFNF 标志）置 1 时，可将数据从配置的 SRAM 区（通过 DMA 外设，请参见*直接存储器访问控制器*部分）加载到 LPUART_TDR 寄存器。要映射一个 DMA 通道以进行 LPUART 发送，请按以下步骤操作（x 表示通道编号）：

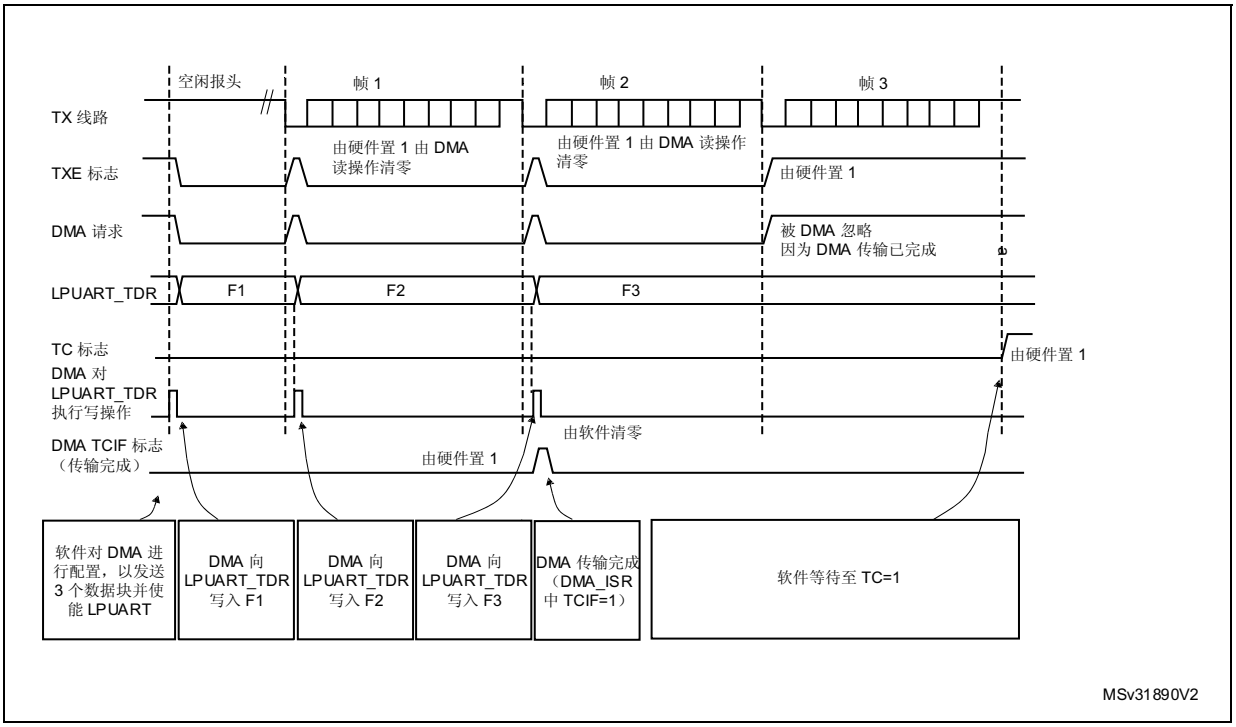
1. 在 DMA 控制寄存器中写入 LPUART_TDR 寄存器地址，将其配置为传输的目标地址。每次发生 TXE（使能 FIFO 模式时为 TXFNF）事件后，数据都从存储器移动到此地址。
2. 在 DMA 控制寄存器中写入存储器地址，将其配置为传输的源地址。每次发生 TXE（使能 FIFO 模式时为 TXFNF）事件后，数据都从该存储区域加载到 LPUART_TDR 寄存器中。
3. 在 DMA 控制寄存器中配置要传输的总字节数。
4. 在 DMA 寄存器中配置通道优先级
5. 根据应用的需求，在完成一半或全部传输后产生 DMA 中断。
6. 通过将 LPUART_ICR 寄存器中的 TCCF 位置 1，将 LPUART_ISR 寄存器中的 TC 标志清零。
7. 在 DMA 寄存器中激活该通道。

当达到在 DMA 控制器中设置的数据传输量时，DMA 控制器会在 DMA 通道的中断向量上产生一个中断。

在发送模式下，DMA 对所有要发送的数据执行了写操作（DMA_ISR 寄存器中的 TCIF 标志置 1）后，可以对 TC 标志进行监视，以确保 LPUART 通信已完成。在禁止 LPUART 或进入低功耗模式前必须执行此步骤，以避免损坏最后一次发送。软件必须等待直到 TC=1。TC 标志在所有数据发送期间都保持清零状态，然后在最后一帧发送结束时由硬件置 1。

注意： *在 DMA 传输结束之前不得将 DMAT 位清零。*

图 459. 使用 DMA 进行发送



注意： *使能 FIFO 管理时，DMA 请求由发送 FIFO 未满（即 TXFNF = 1）触发。*

使用 DMA 进行接收

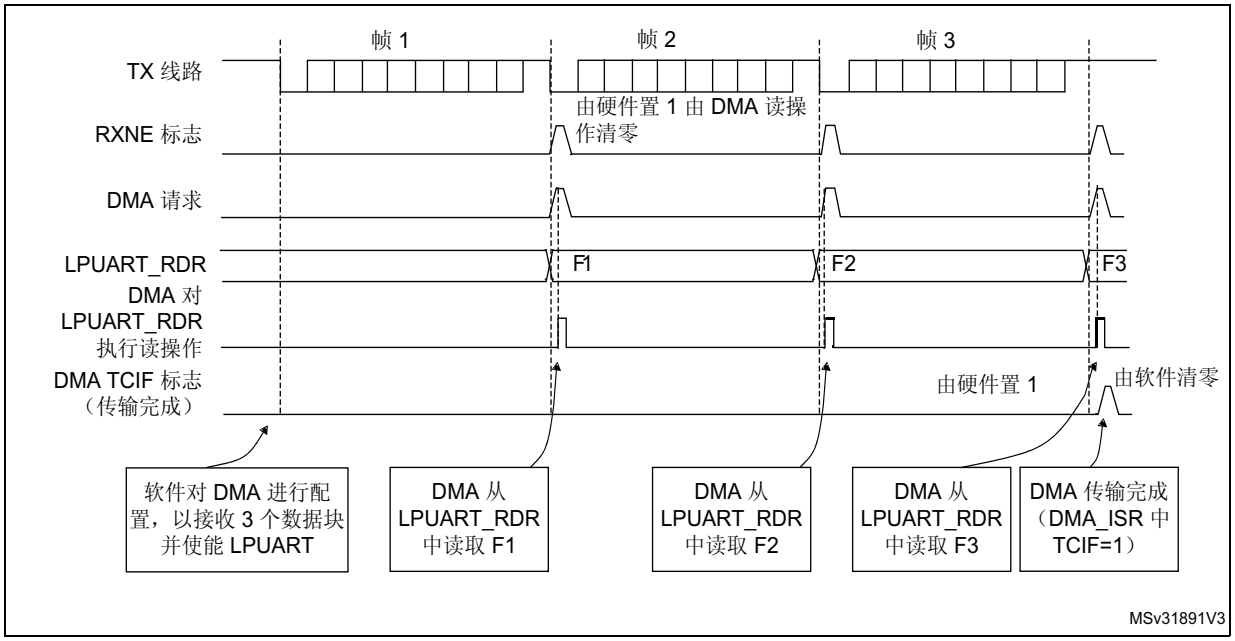
将 LPUART_CR3 寄存器中的 DMAR 位置 1 可以使能 DMA 模式进行接收。接收数据字节时，可使用 DMA 外设（请参见*直接存储器访问控制器 (DMA)* 部分）将数据从 LPUART_RDR 寄存器加载到配置的 SRAM 区域中。要映射一个 DMA 通道以进行 LPUART 接收，请按以下步骤操作：

1. 在 DMA 控制寄存器中写入 LPUART_RDR 寄存器地址，将其配置为传输的源地址。每次发生 RXNE（使能 FIFO 模式时为 RXFNE）事件后，数据都从该地址移动到存储器。
2. 在 DMA 控制寄存器中写入存储器地址，将其配置为传输的目标地址。每次发生 RXNE（使能 FIFO 模式时为 RXFNE）事件后，数据都从 LPUART_RDR 加载到此存储区域。
3. 在 DMA 控制寄存器中配置要传输的总字节数。
4. 在 DMA 控制寄存器中配置通道优先级。
5. 根据应用的需求，在完成一半或全部传输后产生中断。
6. 在 DMA 控制寄存器中激活该通道。

当达到在 DMA 控制器中设置的数据传输量时，DMA 控制器会在 DMA 通道的中断向量上产生一个中断。

注意： *在 DMA 传输结束之前不得将 DMAR 位清零。*

图 460. 使用 DMA 进行接收



注意： *使能 FIFO 管理时，DMA 请求由接收 FIFO 非空（即 RXFNE = 1）触发。*

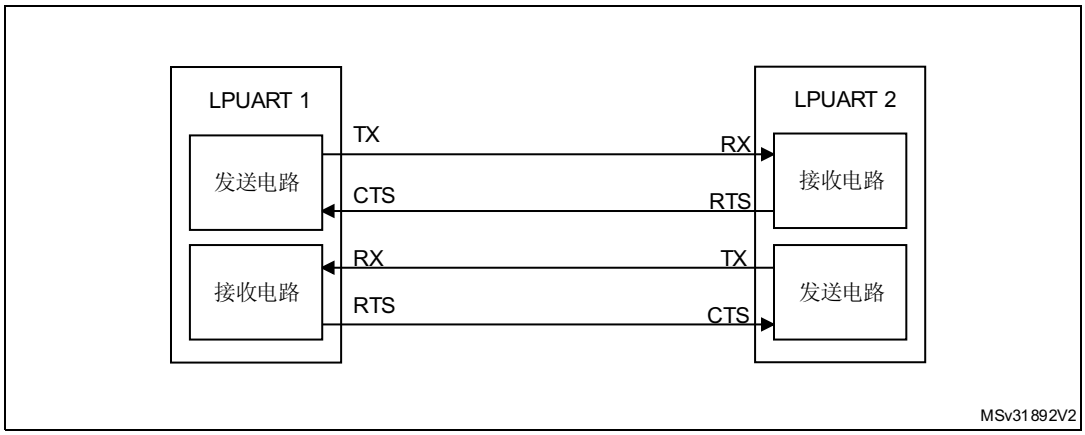
多缓冲区通信中的错误标志和中断生成

在多缓冲区通信模式下，如果事务中发生任何错误，都会在当前字节后将相应错误标志置 1。如果中断使能标志置 1，则会产生中断。在单字节接收过程中，与 RXNE（使能 FIFO 模式时为 RXFNE）一同置位的帧错误、上溢错误和噪声标志具有单独的错误标志中断使能位（LPUART_CR3 寄存器中的 EIE 位）；如果该位置 1，在发生其中任何一个错误时，都会在当前字节后使能中断。

37.4.14 RS232 硬件流控制和 RS485 驱动器使能

使用 CTS 输入和 RTS 输出可以控制 2 个设备间的串行数据流。*图 461* 显示了在这种模式下如何连接 2 个设备：

图 461. 2 个 LPUART 间的硬件流控制

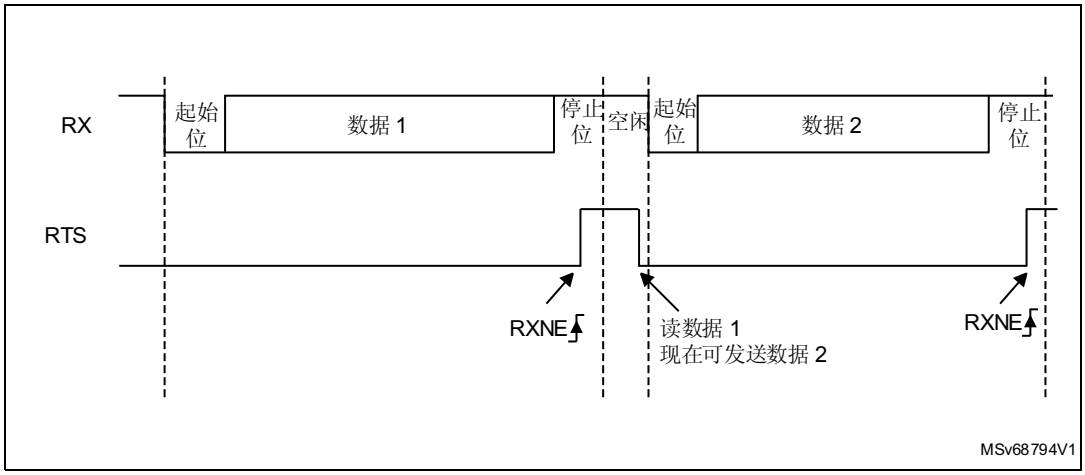


分别向 LPUART_CR3 寄存器中的 RTSE 位和 CTSE 位写入 1，可以分别使能 RS232 RTS 和 CTS 流控制。

RS232 RTS 流控制

如果使能 RTS 流控制 (RTSE=1)，只要 LPUART 接收器准备好接收新数据，便会将 RTS 变为无效（连接到低电平）。当接收寄存器已满时，会将 RTS 变为有效，表明发送过程会在当前帧结束后停止。*图 462* 显示了在使能 RTS 流控制的情况下进行通信的示例。

图 462. RS232 RTS 流控制



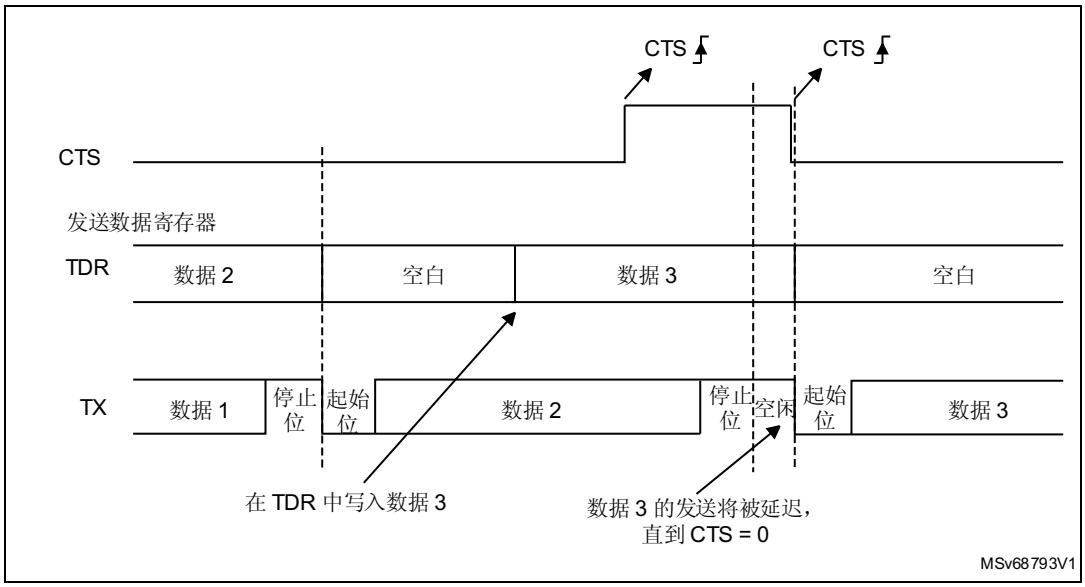
注意： *如果使能 FIFO 模式，则仅当 RXFIFO 已满时，RTS 才会变为有效。*

RS232 CTS 流控制

如果使能 CTS 流控制 (CTSE = 1)，则发送器会在发送下一帧前检查 CTS。如果 CTS 无效（连接到低电平），则会发送下一数据（假设数据已准备好发送，即 TXE/TXFE=0）；否则不会进行发送。如果在发送过程中 CTS 变为有效，则当前发送完成之后，发送器停止。

当 CTSE = 1 时，只要 CTS 发生变化，CTSIF 状态位便会由硬件自动置 1。这指示接收器是否已准备好进行通信。如果 LPUART_CR3 寄存器中的 CTSIE 位置 1，则会产生中断。*图 463* 显示了在使能 CTS 流控制的情况下进行通信的示例。

图 463. RS232 CTS 流控制



注意： *为正常运行，必须在当前字符结束前至少 3 个 LPUART 时钟源周期内将 CTS 拉低。此外还必须注意，当脉冲短于 2 个 PCLK 周期时无法将 CTSCF 标志置 1。*

RS485 驱动器使能

驱动器使能功能可通过将 LPUART_CR3 控制寄存器中的 DEM 位置 1 使能。这样便可通过 DE（驱动器使能）信号激活外部收发器控制。使能时间为激活 DE 信号与 START 位开始间的时间。可以通过 LPUART_CR1 控制寄存器中的 DEAT [4:0] 位域编程禁止时间。禁止时间为发送的消息中最后一个停止位结束与取消激活 DE 信号间的时间。可以通过 LPUART_CR1 控制寄存器中的 DEDT [4:0] 位域编程禁止时间。DE 信号的极性可使用 LPUART_CR3 控制寄存器中的 DEP 位配置。

LPUART 的 DEAT 和 DEDT 用 LPUART 时钟源 (f_{CK}) 周期表示：

- 驱动器使能有效时间等于
  - (1 + (DEAT x P)) x f_{CK}（如果 P # 0）
  - (1 + DEAT) x f_{CK}（如果 P = 0）
- 驱动器使能禁止时间等于
  - (1 + (DEDT x P)) x f_{CK}（如果 P # 0）
  - (1 + DEDT) x f_{CK}（如果 P = 0）

其中 P = BRR[20:11]

37.4.15 LPUART 低功耗管理

LPUART 具有高级低功耗模式功能，即使禁止 lpuart_pclk 时钟，也能正常传输数据。
UESM 位置 1 时，LPUART 能够将 MCU 从低功耗模式唤醒。
当 usart_pclk 被关闭时，如果某些特定操作需要激活 **usart_pclk** 时钟，则 LPUART 会提供唤醒中断 (**usart_wkup**)：

- 禁止 FIFO 模式时

  必须激活 luart_pclk 时钟以清空 LPUART 数据寄存器。

  在这种情况下，lpuart_wkup 中断源为 RXNE 置 1。RXNEIE 位必须在进入低功耗模式之前置 1。

- 使能 FIFO 模式时

  必须激活 luart_pclk 时钟以

  – 填充 TXFIFO
  – 或清空 RXFIFO

  在这种情况下，lpuart_wkup 中断源可以为：

  – RXFIFO 非空。此时，RXFNEIE 位必须在进入低功耗模式之前置 1。
  – RxFIFO 已满。此时，RXFFIE 位必须在进入低功耗模式之前置 1，接收到的数据量对应于 RXFIFO 大小，并且 RXFF 标志未置 1。
  – TXFIFO 为空。此时，TXFEIE 位必须在进入低功耗模式之前置 1。

  这允许在低功耗模式下发送/接收 TXFIFO/RXFIFO 中的数据。

  为了避免发生上溢/下溢错误以及在低功耗模式下发送/接收数据，lpuart_wkup 中断源可以是以下事件之一：

  – 达到 TXFIFO 阈值。此时，TXFTIE 位必须在进入低功耗模式之前置 1。
  – 达到 RXFIFO 阈值。此时，RXFTIE 位必须在进入低功耗模式之前置 1。

  例如，如果唤醒时间短于在线路上接收单个字节的时间，则应用可将阈值设为最大 RXFIFO 大小。

  使用 RXFIFO 已满、TXFIFO 为空、RXFIFO 非空和 RXFIFO/TXFIFO 阈值中断将 MCU 从低功耗模式唤醒时，可在低功耗模式下尽可能多地进行 LPUART 传输，这样有助于优化功耗。

或者，也可通过 WUS 位域选择特定 **lpuart_wkup** 中断。

检测到唤醒事件后，WUF 标志会由硬件置 1 并在 WUFIE 位置 1 时生成一个 **lpuart_wkup** 中断。

注意： *在进入低功耗模式之前，请确保未进行任何 LPUART 传输。检查 BUSY 标志不能确保在进行数据接收时绝不会进入低功耗模式。*

WUF 标志在检测到唤醒事件时置 1，而与 MCU 处于低功耗模式还是工作模式无关。

如果在初始化和使能接收器后立即进入低功耗模式，则必须校验 REACK 位以确保 LPUART 确实已使能。

当 DMA 用于接收时，它必须在进入低功耗模式前禁止，并在退出低功耗模式后重新使能。

如果使能 FIFO，则只有在使能静默模式的情况下才能在地址匹配时从低功耗模式唤醒。

使用静默模式和低功耗模式

如果 LPUART 在进入低功耗模式前处于静默模式：

- 不得使用空闲检测来从静默模式唤醒，因为空闲检测无法在低功耗模式下工作。
- 如果使用地址匹配来从静默模式唤醒，则低功耗模式唤醒源也必须是地址匹配。如果 RXNE 标志在进入低功耗模式时置 1，则接口将在地址匹配时和从低功耗模式唤醒时保持静默模式。

注意： *使能 FIFO 管理时，静默模式与从低功耗模式唤醒搭配使用，而且没有任何限制（即，上述关于静默和低功耗模式的两点仅在 FIFO 管理禁止时有效）。*

LPUART 内核时钟 lpuart_ker_ck 在低功耗模式下关闭时从低功耗模式唤醒

在低功耗模式下，如果在 LPUART 接收线上检测到下降沿时 lpuart_ker_ck 时钟处于关闭状态，则 LPUART 接口会借助 lpuart_ker_ck_req 信号请求开启 lpuart_ker_ck 时钟。lpuart_ker_ck 随后用来接收帧。

如果唤醒事件通过验证，则 MCU 将从低功耗模式唤醒，并会正常进行数据接收。

如果唤醒事件未通过验证，则 usart_ker_ck 会再次关闭，MCU 不会被唤醒并保持在低功耗模式下，且内核时钟请求被释放。

以下示例显示了将唤醒事件编程为“地址匹配检测”并禁止 FIFO 管理的情况。

图 464 所示为唤醒事件通过验证时的行为。

图 464. 唤醒事件通过验证（唤醒事件 = 地址匹配，禁止 FIFO）

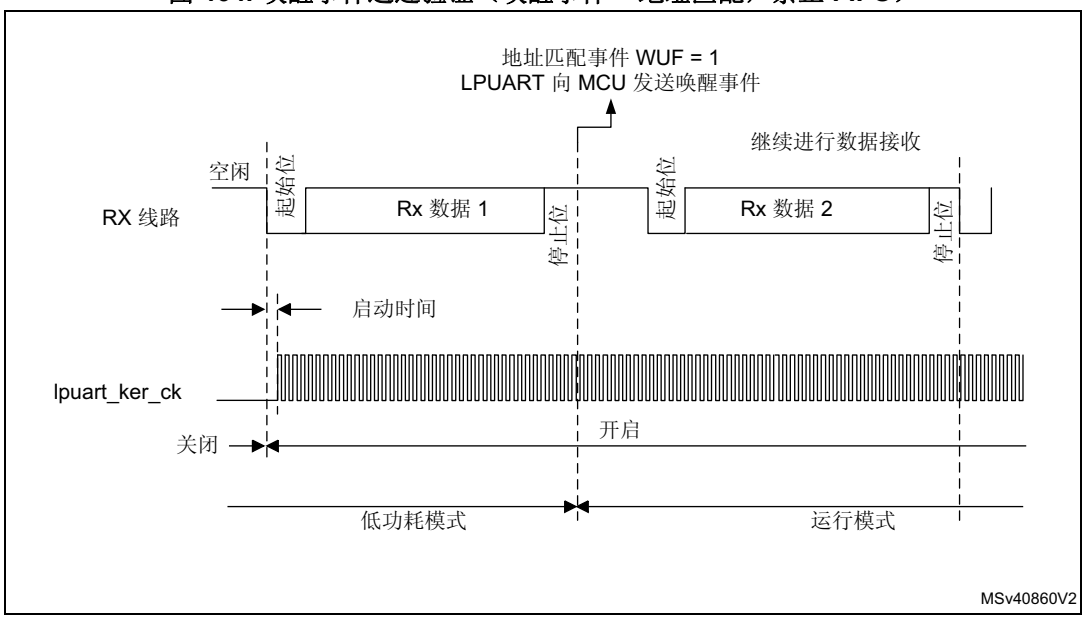
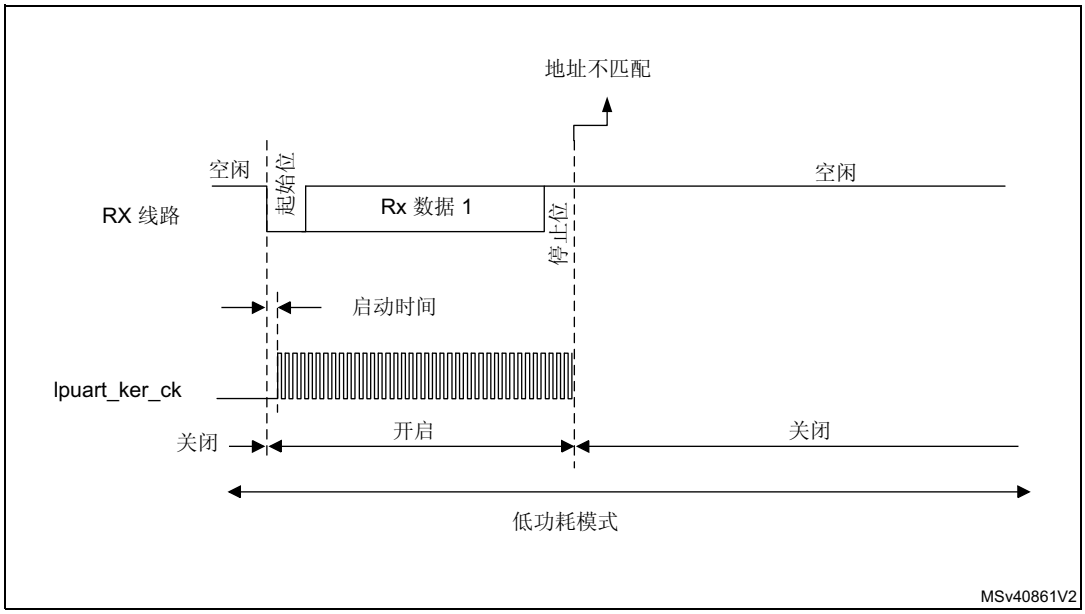


图 465 所示为唤醒事件未通过验证时的行为。

图 465. 唤醒事件未通过验证（唤醒事件 = 地址匹配，禁止 FIFO）



注意： *将地址匹配或任一接收的帧用作唤醒事件时，如上两图适用。如果唤醒事件为起始位检测，则 LPUART 会在起始位结束时向 MCU 发送唤醒事件。*

确定可从低功耗模式正确唤醒 MCU 的最大 LPUART 波特率

允许从低功耗模式正确唤醒 MCU 的最大波特率取决于唤醒时间参数（请参见器件数据手册）和 LPUART 接收器容差（请参见*第 37.4.9 节：LPUART 接收器对时钟偏差的容差*）。

举例来说：OVER8 = 0，M 位 = 01，ONEBIT = 0，BRR [3:0] = 0000。

在这些条件下，根据*表 300：LPUART 接收器的容差*，LPUART 接收器的容差为 3.41%。

DTRA + DQUANT + DREC + DTCL + DWU < LPUART 接收器的容差

$D_{WUmax} = t_{WULPUART} / (11 \times T_{bit\ Min})$

$T_{bit\ Min} = t_{WULPUART} / (11 \times D_{WUmax})$

其中 $t_{WULPUART}$ 为从低功耗模式唤醒的时间。

考虑一种理想情况：参数 DTRA、DQUANT、DREC 和 DTCL 为 0%，则 DWU 的最大值为 3.41%。实际上，我们至少需要考虑 lpuart_ker_ck 不精确的情况。

例如，如果将 HSI 用作 lpuart_ker_ck，HSI 的不精确度为 1%，则可以得到：

$t_{WULPUART}$ = 3 µs（仅提供数值作为参考；有关正确数值，请参见器件数据手册）。

D_{WUmax} = 3.41% - 1% = 2.41%

$T_{bit\ min}$ = 3 µs/ (11 x 2.41%) = 11.32 µs。

因此，允许从低功耗模式正确唤醒的最大波特率为：1/11.32 µs = 88.36 kbaud。

37.5 低功耗模式下的 LPUART

表 302. 低功耗模式对 LPUART 的影响

模式	说明
睡眠	无影响。LPUART 中断可使器件退出睡眠模式。
停止(1)	LPUART 寄存器的内容会保留。 当 LPUART 的时钟由停止模式下可用的振荡器提供时，LPUART 能够将微控制器从停止模式唤醒。
待机	LPUART 外设掉电，退出待机模式后必须重新初始化。

1. 请参见*第 37.3 节：LPUART 实现*以了解给定外设实例是否支持从停止模式唤醒。如果某个实例在给定的停止模式下不起作用，则必须在进入此停止模式之前将其禁止。

37.6 LPUART 中断

有关所有 LPUART 中断请求的详细说明，请参见*表 303*。

表 303. LPUART 中断请求

中断向量	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止(1)模式	退出待机模式
LPUART	发送数据寄存器为空	TXE	TXEIE	写入 TDR	是	否	否
	发送 FIFO 未满	TXFNF	TXFNFIE	TXFIFO 已满		否	
	发送 FIFO 为空	TXFE	TXFEIE	写入 TDR 或向 TXFRQ 中写入 1		是	
	达到发送 FIFO 阈值	TXFT	TXFTIE	写入 TDR		是	
	CTS 中断	CTSIF	CTSIE	向 CTSCF 中写入 1		否	
	发送完成	TC	TCIE	写入 TDR 或向 TCCF 中写入 1		否	
	接收数据寄存器不为空（已准备好读取数据）	RXNE	RXNEIE	读取 RDR 或向 RXFRQ 中写入 1	是	是	
	接收 FIFO 非空	RXFNE	RXFNEIE	读取 RDR 直到 RXFIFO 为空或向 RXFRQ 中写入 1		是	
	接收 FIFO 已满	RXFF(2)	RXFFIE	读取 RDR		是	
	达到接收 FIFO 阈值	RXFT	RXFTIE	读取 RDR		是	
	检测到上溢错误	ORE	RXNEIE/ RXFNEIE	向 ORECF 中写入 1		否	
	检测到空闲线路	IDLE	IDLEIE	向 IDLECF 中写入 1		否	
	奇偶校验错误	PE	PEIE	向 PECF 中写入 1		否	
	多缓冲区通信中的噪声错误。	NE	EIE	向 NFCF 中写入 1		否	
	多缓冲区通信中的上溢错误。	ORE(3)		向 ORECF 中写入 1		否	
	多缓冲区通信中的帧错误。	FE		向 FECF 中写入 1		否	
	字符匹配	CMF	CMIE	向 CMCF 中写入 1		否	
	从低功耗模式唤醒	WUF	WUFIE	向 WUC 中写入 1		是	

1. 仅当外设实例支持从停止模式唤醒功能时，LPUART 才能将器件从停止模式唤醒。关于所支持的停止模式的列表，请参见*第 37.3 节：LPUART 实现*。
2. 如果 LPUART 接收 n+1 个数据（n 表示 RXFIFO 大小，RXFIFO 中有 n 个数据，LPUART_RDR 中有 1 个数据），则 RXFF 标志置 1。在停止模式下，未向 LPUART_RDR 提供时钟。因此，不会对该寄存器进行写操作，一旦有 n 个数据被接收并写入到 RXFIFO，RXFF 中断便会生效（RXFF 标志未置 1）。
3. 当 OVRDIS = 0 时。

37.7 LPUART 寄存器

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节*。

外设寄存器必须按字（32 位）进行访问。

37.7.1 LPUART 控制寄存器 1 (LPUART_CR1)

LPUART control register 1

偏移地址：0x00

复位值：0x0000 0000

同一寄存器可用于使能 FIFO 模式（本节）和禁止 FIFO 模式（下一节）的情况。

使能 FIFO 模式

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXFFIE	TXFEIE	FIFOEN	M1	Res.	Res.	DEAT[4:0]					DEDT[4:0]				
rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXFNFIE	TCIE	RXFNEIE	IDLEIE	TE	RE	UESM	UE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **RXFFIE**：RXFIFO 变满时中断使能 (RXFIFO Full interrupt enable)
该位由软件置 1 和清零。
0：禁止中断
1：当 LPUART_ISR 寄存器中的 RXFF=1 时，生成 LPUART 中断

位 30 **TXFEIE**：TXFIFO 为空时中断使能 (TXFIFO empty interrupt enable)
该位由软件置 1 和清零。
0：禁止中断
1：当 LPUART_ISR 寄存器中的 TXFE=1 时，生成 LPUART 中断

位 29 **FIFOEN**：FIFO 模式使能 (FIFO mode enable)
该位由软件置 1 和清零。
0：禁止 FIFO 模式。
1：使能 FIFO 模式。

位 28 **M1**：字长 (Word length)
该位必须与位 12 (M0) 搭配使用来确定字长度。该位由软件置 1 或清零。
M[1:0] = 00：1 个起始位，8 个数据位，n 个停止位
M[1:0] = 01：1 个起始位，9 个数据位，n 个停止位
M[1:0] = 10：1 个起始位，7 个数据位，n 个停止位
只有在禁止 LPUART (UE=0) 时才能写入该位。
注意：在 7 位数据长度模式下，不支持智能卡模式、LIN 主模式和自动波特率（0x7F 帧和 0x55 帧检测）。

位 27:26 保留，必须保持复位值。

位 25:21 **DEAT[4:0]**：驱动器使能有效时间 (Driver Enable assertion time)

该 5 位值用于定义激活 DE（启动器使能）信号与起始位开始间的时间。以 lpuart_ker_ck 时钟周期数表示。有关详细信息，请参见*第 36.5.21 节：RS232 硬件流控制和 RS485 驱动器使能*。

只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 20:16 **DEDT[4:0]**：驱动器使能无效时间 (Driver Enable deassertion time)

该 5 位值用于定义发送的消息中最后一个停止位结束与取消激活 DE（驱动器使能）信号间的时间。以 lpuart_ker_ck 时钟周期数表示。有关详细信息，请参见*第 37.4.14 节：RS232 硬件流控制和 RS485 驱动器使能*。

如果在 DEDT 时间内对 LPUART_TDR 寄存器执行写操作，则新数据仅在经过 DEDT 和 DEAT 时间后才会发送。

只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 15 保留，必须保持复位值。

位 14 **CMIE**：字符匹配中断使能 (Character match interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：如果 LPUART_ISR 寄存器中的 CMF 位置 1，则生成 LPUART 中断。

位 13 **MME**：静默模式使能 (Mute mode enable)

该位用于激活 LPUART 的静默模式功能。该位置 1 时，LPUART 会按照 WAKE 位的定义在工作模式与静默模式之间切换。该位由软件置 1 和清零。

0：接收器永久处于工作模式

1：接收器可在静默模式和工作模式之间切换。

位 12 **M0**：字长 (Word length)

该位与位 28 (M1) 搭配使用来确定字长度。它由软件置 1 或清零（请参见位 28 (M1) 的说明）。

只有在禁止 LPUART (UE=0) 时才能写入该位。

位 11 **WAKE**：接收器唤醒方法 (Receiver wake-up method)

该位用于确定 LPUART 静默模式的唤醒方法。该位由软件置 1 或清零。

0：空闲线路

1：地址标记

只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 10 **PCE**：奇偶校验控制使能 (Parity control enable)

该位选择硬件奇偶校验控制（生成和检测）。使能奇偶校验控制时，计算出的奇偶校验位被插入到 MSB 位置（如果 M=1，则为第 9 位；如果 M=0，则为第 8 位），并对接收到的数据检查奇偶校验位。该位由软件置 1 和清零。一旦该位置 1，PCE 在当前字节的后面处于活动状态（在接收和发送时）。

0：禁止奇偶校验控制

1：使能奇偶校验控制

只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 9 **PS**：奇偶校验选择 (Parity selection)

该位用于在使能奇偶校验生成/检测（PCE 位置 1）时选择奇校验或偶校验。该位由软件置 1 和清零。在当前字节的后面选择奇偶校验。

0：偶校验

1：奇校验

只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 8 **PEIE**：PE 中断使能 (PE interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 LPUART_ISR 寄存器中的 PE=1 时，生成 LPUART 中断

位 7 **TXFNFIE**：TXFIFO 未满中断使能 (TXFIFO not full interrupt enable)
该位由软件置 1 和清零。
0：禁止中断
1：当 LPUART_ISR 寄存器中的 TXFNF=1 时，生成 LPUART 中断

位 6 **TCIE**：传送完成中断使能 (Transmission complete interrupt enable)
该位由软件置 1 和清零。
0：禁止中断
1：当 LPUART_ISR 寄存器中的 TC=1 时，生成 LPUART 中断

位 5 **RXFNEIE**：RXFIFO 非空中断使能 (RXFIFO not empty interrupt enable)
该位由软件置 1 和清零。
0：禁止中断
1：当 LPUART_ISR 寄存器中的 ORE=1 或 RXFNE=1 时，生成 LPUART 中断

位 4 **IDLEIE**：IDLE 中断使能 (IDLE interrupt enable)
该位由软件置 1 和清零。
0：禁止中断
1：当 LPUART_ISR 寄存器中的 IDLE=1 时，生成 LPUART 中断

位 3 **TE**：发送器使能 (Transmitter enable)
该位使能发送器。该位由软件置 1 和清零。
0：禁止发送器
1：使能发送器
注意：除了在智能卡模式下以外，传送期间 TE 位上的低电平脉冲（0 后紧跟的是 1）会在当前字的后面发送一个报头（空闲线路）。为生成空闲字符，TE 不能立即写入 1。为确保所需的持续时间，软件可轮询 LPUART_ISR 寄存器中的 TEACK 位。
在智能卡模式下，当 TE 置 1 时，在发送开始前存在 1 位的时间延迟。

位 2 **RE**：接收器使能 (Receiver enable)
该位使能接收器。该位由软件置 1 和清零。
0：禁止接收器
1：使能接收器并开始搜索起始位

位 1 **UESM**：低功耗模式下的 LPUART 使能 (LPUART enable in low-power mode)
当该位清零时，LPUART 无法将 MCU 从低功耗模式唤醒。
当该位置 1 时，LPUART 可将 MCU 从低功耗模式唤醒。
该位由软件置 1 和清零。
0：USART 无法将 MCU 从低功耗模式唤醒。
1：USART 能够将 MCU 从低功耗模式唤醒。
注意：建议在进入低功耗模式前将 UESM 位置 1，并在退出低功耗模式时将其清零。

位 0 **UE**：LPUART 使能 (LPUART enable)
该位清零后，LPUART 预分频器和输出将立即停止，并丢弃当前操作。LPUART 的配置保留，但 LPUART_ISR 中的所有状态标志将被复位。该位由软件置 1 和清零。
0：禁止 LPUART 预分频器和输出，低功耗模式
1：使能 LPUART
注意：为进入低功耗模式而不在线路上生成错误，之前必须复位 TE 位，并且软件必须等待 LPUART_ISR 寄存器中的 TC 位置 1 后才能复位 UE 位。
UE = 0 时也会复位 DMA 请求，因此必须在复位 UE 位前禁止 DMA 通道。

37.7.2 LPUART 控制寄存器 1 [备用] (LPUART_CR1)

LPUART control register 1

偏移地址：0x00

复位值：0x0000 0000

同一寄存器可用于使能 FIFO 模式（上一节）和禁止 FIFO 模式（本节）的情况。

禁止 FIFO 模式

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	FIFO EN	M1	Res.	Res.	DEAT[4:0]					DEDT[4:0]				
		rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:30 保留，必须保持复位值。

位 29 **FIFOEN**：FIFO 模式使能 (FIFO mode enable)
该位由软件置 1 和清零。
0：禁止 FIFO 模式。
1：使能 FIFO 模式。

位 28 **M1**：字长 (Word length)
该位必须与位 12 (M0) 搭配使用来确定字长度。该位由软件置 1 或清零。
M[1:0] = 00：1 个起始位，8 个数据位，n 个停止位
M[1:0] = 01：1 个起始位，9 个数据位，n 个停止位
M[1:0] = 10：1 个起始位，7 个数据位，n 个停止位
只有在禁止 LPUART (UE=0) 时才能写入该位。
注意：在 7 位数据长度模式下，不支持智能卡模式、LIN 主模式和自动波特率（0x7F 帧和 0x55 帧检测）。

位 27:26 保留，必须保持复位值。

位 25:21 **DEAT[4:0]**：驱动器使能有效时间 (Driver Enable assertion time)
该 5 位值用于定义激活 DE（启动器使能）信号与起始位开始间的时间。以 lpuart_ker_ck 时钟周期数表示。有关详细信息，请参见*第 36.5.21 节：RS232 硬件流控制和 RS485 驱动器使能*。
只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 20:16 **DEDT[4:0]**：驱动器使能无效时间 (Driver Enable deassertion time)
该 5 位值用于定义发送的消息中最后一个停止位结束与取消激活 DE（驱动器使能）信号间的时间。以 lpuart_ker_ck 时钟周期数表示。有关详细信息，请参见*第 37.4.14 节：RS232 硬件流控制和 RS485 驱动器使能*。
如果在 DEDT 时间内对 LPUART_TDR 寄存器执行写操作，则新数据仅在经过 DEDT 和 DEAT 时间后才会发送。
只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 15 保留，必须保持复位值。

位 14 **CMIE**：字符匹配中断使能 (Character match interrupt enable)
该位由软件置 1 和清零。
0：禁止中断
1：如果 LPUART_ISR 寄存器中的 CMF 位置 1，则生成 LPUART 中断。

位 13 **MME**：静默模式使能 (Mute mode enable)

该位用于激活 LPUART 的静默模式功能。该位置 1 时，LPUART 会按照 WAKE 位的定义在工作模式与静默模式之间切换。该位由软件置 1 和清零。

0：接收器永久处于工作模式

1：接收器可在静默模式和工作模式之间切换。

位 12 **M0**：字长 (Word length)

该位与位 28 (M1) 搭配使用来确定字长度。它由软件置 1 或清零（请参见位 28 (M1) 的说明）。

只有在禁止 LPUART (UE=0) 时才能写入该位。

位 11 **WAKE**：接收器唤醒方法 (Receiver wake-up method)

该位用于确定 LPUART 静默模式的唤醒方法。该位由软件置 1 或清零。

0：空闲线路

1：地址标记

只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 10 **PCE**：奇偶校验控制使能 (Parity control enable)

该位选择硬件奇偶校验控制（生成和检测）。使能奇偶校验控制时，计算出的奇偶校验位被插入到 MSB 位置（如果 M=1，则为第 9 位；如果 M=0，则为第 8 位），并对接收到的数据检查奇偶校验位。该位由软件置 1 和清零。一旦该位置 1，PCE 在当前字节的后面处于活动状态（在接收和发送时）。

0：禁止奇偶校验控制

1：使能奇偶校验控制

只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 9 **PS**：奇偶校验选择 (Parity selection)

该位用于在使能奇偶校验生成/检测（PCE 位置 1）时选择奇校验或偶校验。该位由软件置 1 和清零。在当前字节的后面选择奇偶校验。

0：偶校验

1：奇校验

只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 8 **PEIE**：PE 中断使能 (PE interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 LPUART_ISR 寄存器中的 PE=1 时，生成 LPUART 中断

位 7 **TXEIE**：发送数据寄存器为空 (Transmit data register empty)

该位由软件置 1 和清零。

0：禁止中断

1：当 LPUART_ISR 寄存器中的 TXE=1 时，生成 LPUART 中断

位 6 **TCIE**：传送完成中断使能 (Transmission complete interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 LPUART_ISR 寄存器中的 TC=1 时，生成 LPUART 中断

位 5 **RXNEIE**：接收数据寄存器非空 (Receive data register not empty)

该位由软件置 1 和清零。

0：禁止中断

1：当 LPUART_ISR 寄存器中的 ORE=1 或 RXNE=1 时，生成 LPUART 中断

位 4 **IDLEIE**：IDLE 中断使能 (IDLE interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 LPUART_ISR 寄存器中的 IDLE=1 时，生成 LPUART 中断

位 3 **TE**：发送器使能 (Transmitter enable)

该位使能发送器。该位由软件置 1 和清零。

0：禁止发送器

1：使能发送器

注意：除了在智能卡模式下以外，传送期间 TE 位上的低电平脉冲（0 后紧跟的是 1）会在当前字的后面发送一个报头（空闲线路）。为生成空闲字符，TE 不能立即写入“1”。为确保所需的持续时间，软件可轮询 LPUART_ISR 寄存器中的 TEACK 位。

在智能卡模式下，当 TE 置 1 时，在发送开始前存在 1 位的时间延迟。

位 2 **RE**：接收器使能 (Receiver enable)

该位使能接收器。该位由软件置 1 和清零。

0：禁止接收器

1：使能接收器并开始搜索起始位

位 1 **UESM**：低功耗模式下的 LPUART 使能 (LPUART enable in low-power mode)

当该位清零时，LPUART 无法将 MCU 从低功耗模式唤醒。

当该位置 1 时，LPUART 可将 MCU 从低功耗模式唤醒。

该位由软件置 1 和清零。

0：USART 无法将 MCU 从低功耗模式唤醒。

1：USART 能够将 MCU 从低功耗模式唤醒。

注意：建议在进入低功耗模式前将 UESM 位置 1，并在退出低功耗模式时将其清零。

位 0 **UE**：LPUART 使能 (LPUART enable)

该位清零后，LPUART 预分频器和输出将立即停止，并丢弃当前操作。LPUART 的配置保留，但 LPUART_ISR 中的所有状态标志将被复位。该位由软件置 1 和清零。

0：禁止 LPUART 预分频器和输出，低功耗模式

1：使能 LPUART

注意：为进入低功耗模式而不在线路上生成错误，之前必须复位 TE 位，并且软件必须等待 LPUART_ISR 寄存器中的 TC 位置 1 后才能复位 UE 位。

UE = 0 时也会复位 DMA 请求，因此必须在复位 UE 位前禁止 DMA 通道。

37.7.3 LPUART 控制寄存器 2 (LPUART_CR2)

LPUART control register 2

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD[7:0]								Res.	Res.	Res.	Res.	MSBFIRST	DATAINV	TXINV	RXINV
rw	rw	rw	rw	rw	rw	rw	rw					rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWAP	Res.	STOP[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADDM7	Res.	Res.	Res.	Res.
rw		rw	rw								rw				

位 31:24 **ADD[7:0]**：LPUART 节点的地址 (Address of the LPUART node)

这些位用于指定静默模式下 LPUART 节点的地址或低功耗模式/运行模式下要识别的字符代码。

- 在静默模式下：它们用于在多处理器通信时通过 4 位/7 位地址标记检测从静默模式唤醒。发送器发送字符的 MSB 应为 1。在 4 位地址标记检测中，只使用 ADD[3:0] 位。
- 在低功耗模式下：它们用于在字符匹配时从低功耗模式唤醒。
  如果将 WUS[1:0] 编程为 0b00（WUF 在地址匹配时激活），并且已通过将 WUFIE 位置 1 使能 WUF 中断，则当收到的字符与通过 ADD[6:0] 或 ADD[3:0] 位域（取决于 ADDM7 位）编程的字符匹配时会从低功耗模式唤醒。发送器发送字符的 MSB 应为 1。
- 在静默模式无效时的运行模式下（例如，ModBus 协议中的块结束检测）：接收到的整个字符（8 位）将与 ADD[7:0] 值进行比较，如果匹配，CMF 标志将置 1。如果 CMIE 位置 1，则会生成中断。

仅在禁止接收 (RE = 0) 或禁止 USART (UE = 0) 时才能写入这些位。

位 23:20 保留，必须保持复位值。

位 19 **MSBFIRST**：最高有效位在前 (Most significant bit first)

该位由软件置 1 和清零。
0：发送/接收数据时位 0 在前，后跟起始位。
1：发送/接收数据时 MSB（位 7/8）在前，后跟起始位。
只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 18 **DATAINV**：二进制数据反向 (Binary data inversion)

该位由软件置 1 和清零。
0：按正/正向逻辑发送/接收数据寄存器中的逻辑数据。（1=H，0=L）
1：按负/反向逻辑发送/接收数据寄存器中的逻辑数据。（1=L，0=H）。奇偶校验位也取反。
只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 17 **TXINV**：TX 引脚有效电平反向 (TX pin active level inversion)

该位由软件置 1 和清零。
0：TX 引脚信号使用标准逻辑电平（V_{DD} = 1/空闲，Gnd = 0/标记）工作。
1：对 TX 引脚信号值取反。（V_{DD} = 0/标记，Gnd=1/空闲）。
允许在 TX 线路上使用外部反相器。
只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 16 **RXINV**：RX 引脚有效电平反向 (RX pin active level inversion)

该位由软件置 1 和清零。
0：RX 引脚信号使用标准逻辑电平（V_{DD} = 1/空闲，Gnd = 0/标记）工作。
1：对 RX 引脚信号值取反。（V_{DD} = 0/标记，Gnd=1/空闲）。
允许在 RX 线路上使用外部反相器。
只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 15 **SWAP**：交换 TX/RX 引脚 (Swap TX/RX pins)

该位由软件置 1 和清零。
0：按标准引脚排列定义使用 TX/RX 引脚
1：交换 TX 和 RX 引脚功能。允许在与另一个 UART 的交叉连接时工作。
只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 14 保留，必须保持复位值。

位 13:12 **STOP[1:0]**：停止位 (STOP bit)

这些位用于编程停止位。
00：1个停止位
01：保留
10：2 个停止位
11：保留
只有在禁止 LPUART (UE=0) 时才能写入该位域。

位 11:5 保留，必须保持复位值。

位 4 **ADDM7**：7 位地址检测/4 位地址检测 (7-bit Address Detection/4-bit Address Detection)
该位用于选择 4 位地址检测或 7 位地址检测。
0：4 位地址检测
1：7 位地址检测（在 8 位数据模式下）
只有在禁止 LPUART (UE=0) 时才能写入该位
注意：在 7 位和 9 位数据模式下，地址检测分别在 6 位和 8 位地址上完成（ADD[5:0] 和 ADD[7:0]）。

位 3:0 保留，必须保持复位值。

37.7.4 LPUART 控制寄存器 3 (LPUART_CR3)

LPUART control register 3

偏移地址：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXFTCFG[2:0]			RXFTIE	RXFTCFG[2:0]			Res.	TXFTIE	WUFIE	WUS1	WUS0	Res.	Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEP	DEM	DDRE	OVRDIS	Res.	CTSIE	CTSE	RTSE	DMAT	DMAR	Res.	Res.	HDSEL	Res.	Res.	EIE
rw	rw	rw	rw		rw	rw	rw	rw	rw			rw			rw

位 31:29 **TXFTCFG[2:0]**：TXFIFO 阈值配置 (TXFIFO threshold configuration)
000：TXFIFO 达到其深度的 1/8。
001：TXFIFO 达到其深度的 1/4。
110：TXFIFO 达到其深度的 1/2。
011：TXFIFO 达到其深度的 3/4。
100：TXFIFO 达到其深度的 7/8。
101：TXFIFO 变空。
其余组合：保留。

位 28 **RXFTIE**：RXFIFO 阈值中断使能 (FIFO threshold interrupt enable)
该位由软件置 1 和清零。
0：禁止中断
1：当接收 FIFO 达到 RXFTCFG 中编程的阈值时，生成 LPUART 中断。

位 27:25 **RXFTCFG[2:0]**：接收 FIFO 阈值配置 (Receive FIFO threshold configuration)
000：接收 FIFO 达到其深度的 1/8。
001：接收 FIFO 达到其深度的 1/4。
110：接收 FIFO 达到其深度的 1/2。
011：接收 FIFO 达到其深度的 3/4。
100：接收 FIFO 达到其深度的 7/8。
101：接收 FIFO 已满。
其余组合：保留。

位 24 保留，必须保持复位值。

位 23 **TXFTIE**：TXFIFO 阈值中断使能 (TXFIFO threshold interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 TXFIFO 达到 TXFTCFG 中编程的阈值时，生成 LPUART 中断。

位 22 **WUFIE**：从低功耗模式唤醒中断使能 (Wake-up from low-power mode interrupt enable)

该位由软件置 1 和清零。

0：禁止中断

1：当 LPUART_ISR 寄存器中的 WUF=1 时，生成 USART 中断

注意：WUFIE 必须在进入低功耗模式前置 1。

如果 USART 不支持从停止模式唤醒功能，该位保留且必须保持复位值。请参见第 1333 页的第 37.3 节：LPUART 实现。

位 21:20 **WUS[1:0]**：从低功耗模式唤醒中断标志选择 (Wake-up from low-power mode interrupt flag selection)

该位域用于指定激活 WUF（从低功耗模式唤醒标志）的事件。

00：WUF 在地址匹配时激活（按 ADD[7:0] 和 ADDM7 所定义）

01：保留。

10：WUF 在起始位检测时激活

11：WUF 在 RXNE/RXFNE 时激活。

只有在禁止 LPUART (UE=0) 时才能写入该位域。

注意：如果 USART 不支持从停止模式唤醒功能，该位保留且必须保持复位值。请参见第 1333 页的第 37.3 节：LPUART 实现。

位 19:16 保留，必须保持复位值。

位 15 **DEP**：驱动器使能极性选择 (Driver enable polarity selection)

0：DE 信号高电平有效。

1：DE 信号低电平有效。

只有在禁止 LPUART (UE=0) 时才能写入该位。

位 14 **DEM**：驱动器使能模式 (Driver enable mode)

该位用于通过 DE 信号激活外部收发器控制。

0：禁止 DE 功能。

1：使能 DE 功能。DE 信号在 RTS 引脚上输出。

只有在禁止 LPUART (UE=0) 时才能写入该位。

位 13 **DDRE**：接收出错时的 DMA 禁止 (DMA Disable on Reception Error)

0：接收出错时不禁止 DMA。相应的错误标志置 1，但 RXNE 保持为 0 以防止上溢。因此，将不使能 DMA 请求，从而不会传送错误数据（无 DMA 请求），但会传送接收到的下一个正确数据。

1：接收出错后禁止 DMA。相应的错误标志以及 RXNE 均置 1。屏蔽 DMA 请求，直到错误标志清零。这意味着软件必须首先禁止 DMA 请求 (DMAR = 0) 或者将 RXNE 清零，然后再将错误标志清零。

只有在禁止 LPUART (UE=0) 时才能写入该位。

注意：接收错误包括：奇偶校验错误、帧错误或噪声错误。

位 12 **OVRDIS**：上溢禁止 (Overrun Disable)

该位用于禁止接收上溢检测。

0：接收新数据前未读取已接收的数据时，上溢错误标志 ORE 置 1。

1：禁止上溢功能。如果在 RXNE 标志仍置 1 时接收到新数据，

则 ORE 标志不会置 1，且新接收的数据会覆盖 LPUART_RDR 寄存器之前的内容。

只有在禁止 LPUART (UE=0) 时才能写入该位。

注意：此控制位用于检查通信流而不会读取数据。

位 11 保留，必须保持复位值。

位 10 **CTSIE**：CTS 中断使能 (CTS interrupt enable)

0：禁止中断

1：当 LPUART_ISR 寄存器中 CTSIF=1 时，生成中断

位 9 **CTSE**：CTS 使能 (CTS enable)

0：禁止 CTS 硬件流控制

1：使能 CTS 模式，仅当 CTS 输入无效（连接到 0）时才发送数据。如果在发送数据时使 CTS 输入有效，会在停止之前完成发送。如果使 CTS 有效时数据已写入数据寄存器，则将延迟发送，直到 CTS 无效。

只有在禁止 LPUART (UE=0) 时才能写入该位

位 8 **RTSE**：RTS 使能 (RTS enable)

0：禁止 RTS 硬件流控制

1：使能 RTS 输出，仅当接收缓冲区中有空间时才会请求数据。发送完当前字符后应停止发送数据。可以接收数据时使 RTS 输出无效（拉至 0）。

只有在禁止 LPUART (UE=0) 时才能写入该位。

位 7 **DMAT**：DMA 使能发送器 (DMA enable transmitter)

该位由软件置 1/复位。

1：针对发送使能 DMA 模式

0：针对发送禁止 DMA 模式

位 6 **DMAR**：DMA 使能接收器 (DMA enable receiver)

该位由软件置 1/复位。

1：针对接收使能 DMA 模式

0：针对接收禁止 DMA 模式

位 5:4 保留，必须保持复位值。

位 3 **HDSEL**：半双工选择 (Half-duplex selection)

选择单线半双工模式

0：未选择半双工模式

1：选择半双工模式

只有在禁止 LPUART (UE=0) 时才能写入该位。

位 2:1 保留，必须保持复位值。

位 0 **EIE**：错误中断使能 (Error interrupt enable)

如果发生帧错误、上溢错误或出现噪声标志（LPUART_ISR 寄存器中 FE = 1 或 ORE = 1 或 NE = 1），则需要使用错误中断使能位来使能中断生成。

0：禁止中断

1：LPUART_ISR 寄存器中的 FE=1 或 ORE=1 或 NE=1 时生成中断。

37.7.5 LPUART 波特率寄存器 (LPUART_BRR)

LPUART baud rate register

只有在禁止 LPUART (UE=0) 时才能写入此寄存器。在自动波特率检测模式下，该位由硬件自动更新。

偏移地址：0x0C

复位值：0x0000 0000

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

位 31:20 保留，必须保持复位值。

位 19:0 **BRR[19:0]**：LPUART 波特率分频 (LPUARTDIV) (LPUART baud rate division (LPUARTDIV))

注意： *禁止在 LPUART_BRR 寄存器中写入小于 0x300 的值。*

如果 LPUART_BRR 必须≥0x300 且 LPUART_BRR 为 20 位，则使用高 fck 值生成高波特率时必须十分谨慎。fck 必须在 [3 x 波特率到 4096 x 波特率] 的范围内。

37.7.6 LPUART 请求寄存器 (LPUART_RQR)

LPUART request register

偏移地址：0x18

复位值：0x0000 0000

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

位 31:5 保留，必须保持复位值。

位 4 **TXFRQ**：发送数据刷新请求 (Transmit data flush request)

该位在 FIFO 模式使能时使用。当 TXFRQ 位置 1 时，会清空整个 FIFO。这会将标志 TXFE（TXFIFO 为空，LPUART_ISR 寄存器中的位 23）置 1。

注意：在 FIFO 模式下，TXFNF 标志在清空请求期间复位，直到 TxFIFO 为空，以确保数据寄存器中没有写入数据。

位 3 **RXFRQ**：接收数据刷新请求 (Receive data flush request)

向该位写入 1 时会将 RXNE 标志清零。

这可以丢弃接收的数据而不对其执行读取操作，并避免发生上溢。

位 2 **MMRQ**：静默模式请求 (Mute mode request)
向该位写入 1 可将 LPUART 置于静默模式，并将 RWU 标志复位。

位 1 **SBKRQ**：发送刹车请求 (Send break request)
向该位写入 1 可将 SBKF 标志置 1 并在发送设备可用后立即请求在线路上发送 BREAK。
注意：如果应用需要在之前插入的所有数据（包括尚未发送的数据）后发送刹车字符，软件必须等到 TXE 标志使能后将 SBKRQ 位置 1。

位 0 保留，必须保持复位值。

37.7.7 LPUART 中断和状态寄存器 (LPUART_ISR)

LPUART interrupt and status register

偏移地址：0x1C

复位值：0x0080 00C0

同一寄存器可用于使能 FIFO 模式（本节）和禁止 FIFO 模式（下一节）的情况。

使能 FIFO 模式

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TXFT	RXFT	Res.	RXFF	TXFE	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY
				r	r		r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	CTS	CTSIF	Res.	TXFNF	TC	RXFNE	IDLE	ORE	NE	FE	PE
					r	r		r	r	r	r	r	r	r	r

位 31:28 保留，必须保持复位值。

位 27 **TXFT**：TXFIFO 阈值标志 (FIFO threshold flag)
当 TXFIFO 达到在 LPUART_CR3 寄存器的 TXFTCFG 中编程的阈值时（即 TXFIFO 包含 TXFTCFG 个空位置），该位由硬件置 1。如果 LPUART_CR3 寄存器中的 TXFTIE 位 =1（位 31），则会生成中断。
0：TXFIFO 未达到编程的阈值。
1：TXFIFO 已达到编程的阈值。

位 26 **RXFT**：RXFIFO 阈值标志 (RXFIFO threshold flag)
当 RXFIFO 达到在 LPUART_CR3 寄存器的 RXFTCFG 中编程的阈值时（即接收 FIFO 包含 RXFTCFG 个数据），该位由硬件置 1。如果 LPUART_CR3 寄存器中的 RXFTIE 位 =1（位 27），则会生成中断。
0：接收 FIFO 未达到编程的阈值。
1：接收 FIFO 已达到编程的阈值。

位 25 保留，必须保持复位值。

位 24 **RXFF**：RXFIFO 已满 (RXFIFO Full)
当接收到的数据量对应于 RXFIFO 大小 + 1（RXFIFO 已满 + LPUART_RDR 寄存器中的 1 个数据）时，该位由硬件置 1。
如果 LPUART_CR1 寄存器中 RXFFIE 位 = 1，则会生成中断。
0：RXFIFO 未满。
1：RXFIFO 已满。

位 23 **TXFE**：TXFIFO 为空 (TXFIFO Empty)

当 TXFIFO 为空时，该位由硬件置 1。当 TXFIFO 包含至少一个数据时，该标志被清零。也可以通过向 LPUART_RQR 寄存器中的位 TXFRQ（位 4）写入 1 将 TXFE 标志置 1。

如果 LPUART_CR1 寄存器中的 TXFEIE 位 =1（位 30），则会生成中断。

0：TXFIFO 非空。

1：TXFIFO 为空。

位 22 **REACK**：接收使能确认标志 (Receive enable acknowledge flag)

LPUART 采用接收使能值时，通过硬件将该位置 1/ 复位。

该位可用于验证 LPUART 是否准备好在进入低功耗模式前接收数据。

注意：如果 LPUART 不支持从停止模式唤醒功能，该位保留且保持复位值。

位 21 **TEACK**：发送使能确认标志 (Transmit enable acknowledge flag)

LPUART 采用发送使能值时，通过硬件将该位置 1/ 复位。

通过写入 TE=0 生成空闲帧请求，然后在 LPUART_CR1 寄存器中写入 TE=1 以遵循 TE=0 最短周期时，可使用该位。

位 20 **WUF**：从低功耗模式唤醒标志 (Wake-up from low-power mode flag)

当检测到唤醒事件时，该位由硬件置 1。事件通过 WUS 位域定义。通过向 LPUART_ICR 寄存器中的 WUCF 写入 1，该位由软件清零。

如果 LPUART_CR3 寄存器中 WUFIE=1，则会生成中断。

注意：当 UESM 清零时，WUF 标志也清零。

如果 USART 不支持从停止模式唤醒功能，该位保留且保持复位值。请参见第 1333 页的第 37.3 节：LPUART 实现。

位 19 **RWU**：*接收器从静默模式唤醒 (Receiver wake-up from Mute mode)*

该位指示 LPUART 是否处于静默模式。当识别出唤醒/静默序列时，该位由硬件清零/置 1。静默模式控制序列（地址或 IDLE）通过 LPUART_CR1 寄存器中的 WAKE 位选择。

当选择 IDLE 模式下唤醒时，该位只能通过用软件向 LPUART_RQR 寄存器中的 MMRQ 位写 1 的方式置 1。

0：接收器处于工作模式

1：接收器处于静默模式

注意：如果 LPUART 不支持从停止模式唤醒功能，该位保留且保持复位值。

位 18 **SBKF**：发送刹车标志 (Send break flag)

该位指示已请求发送刹车字符。通过将 1 写入 LPUART_CR3 寄存器中的 SBKRQ 位，该位由软件置 1。该位在刹车发送的停止位期间由硬件自动复位。

0：不发送刹车字符

1：刹车字符已发送

位 17 **CMF**：字符匹配标志 (Character match flag)

接收到由 ADD[7:0] 定义的字符后由硬件将该位置 1。通过向 LPUART_ICR 寄存器中的 CMCF 写入 1，该位由软件清零。

如果 LPUART_CR1 寄存器中 CMIE=1，则会生成中断。

0：未检测到字符匹配

1：检测到字符匹配

位 16 **BUSY**：忙标志 (Busy flag)

该位由硬件置 1 和复位。当 RX 线路上正在进行通信（成功检测到起始位）时有效。在接收结束（成功或失败）时复位。

0：LPUART 处于空闲状态（无接收）

1：正在接收

位 15:11 保留，必须保持复位值。

位 10 **CTS**：CTS 标志 (CTS flag)

该位由硬件置 1/复位。该位是对 CTS 输入引脚的状态取反。

0：CTS 线置 1

1：CTS 线复位

注意：如果不支持硬件流控制功能，该位保留且保持复位值。

位 9 **CTSIF**：CTS 中断标志 (CTS interrupt flag)

如果 CTSE 位置 1，当 CTS 输入切换时，该位由硬件置 1。通过将 1 写入 LPUART_ICR 寄存器中的 CTSCF 位，该位由软件清零。

如果 LPUART_CR3 寄存器中 CTSIE=1，则会生成中断。

0：CTS 状态线上未发生变化

1：CTS 状态线上发生变化

注意：如果不支持硬件流控制功能，该位保留且保持复位值。

位 8 保留，必须保持复位值。

位 7 **TXFNF**：TXFIFO 未满 (TXFIFO not full)

TXFNF 会在 TXFIFO 未满时由硬件置 1，因此可向 LPUART_TDR 中写入数据。每次对 LPUART_TDR 进行写操作都会将数据置于 TXFIFO 中。该标志会保持置 1，直到 TXFIFO 已满。当 TXFIFO 已满时，该标志清零，表示不能向 LPUART_TDR 写入数据。

在清空请求期间，TXFNF 保持复位，直到 TXFIFO 为空。发送清空请求（通过将 TXFRQ 位置 1）后，必须先检查 TXFNF 标志，然后再写入 TXFIFO（TXFNF 和 TXFE 会同时置 1）。

如果 LPUART_CR1 寄存器中 TXFNFIE 位 = 1，则会生成中断。

0：数据寄存器已满/发送 FIFO 已满。

1：数据寄存器/发送 FIFO 未满。

注意：单缓冲区发送期间使用该位。

位 6 **TC**：发送完成 (Transmission complete)

该位指示写入到 LPUART_TDR 中的最后一个数据已从移位寄存器中发出。TC 标志的行为如下：

- 当 TDN = 0 时，如果已完成对包含数据的帧的发送并且 TXFE 置 1，TC 标志将会置 1。
- 当 TDN 等于 TXFIFO 中的数据量时，如果 TXFIFO 为空并且达到 TDN 时，TC 标志将会置 1。
- 当 TDN 大于 TXFIFO 中的数据量时，TC 将保持清零状态，直到 TXFIFO 再次填满以达到设定要传输的数据量。
- 当 TDN 小于 TXFIFO 中的数据量时，即使 TXFIFO 不为空，当达到 TDN 时 TC 也会置 1。

如果 LPUART_CR1 寄存器中 TCIE = 1，则会生成中断。

TC 位由软件清零，方法是向 LPUART_ICR 寄存器中的 TCCF 写入 1 或向 LPUART_TDR 寄存器执行写操作。

位 5 **RXFNE**：RXFIFO 非空 (RXFIFO not empty)

RXFIFO 非空时，RXFNE 位由硬件置 1，并因此可以从 LPUART_RDR 寄存器读取数据。每次对 LPUART_RDR 进行读操作都会在 RXFIFO 中释放一个位置。RXFNE 在 RXFIFO 为空时清零。

也可以通过向 LPUART_RQR 寄存器中的 RXFRQ 位写入 1 将 RXFNE 标志位清零。

如果 LPUART_CR1 寄存器中 RXFNEIE=1，则会生成中断。

0：未接收到数据

1：已准备好读取接收到的数据

位 4 **IDLE**：检测到空闲线路 (Idle line detected)

检测到空闲线路时，该位由硬件置 1。如果 LPUART_CR1 寄存器中 IDLEIE=1，则会生成中断。通过向 LPUART_ICR 寄存器中的 IDLECF 写入 1，该位由软件清零。

0：未检测到空闲线路

1：检测到空闲线路

注意：直到 RXFNE 位已置 1 时（即，当出现新的空闲线路时）IDLE 位才会被再次置 1。

使能静默模式 (MME=1) 后，如果 LPUART 未静默 (RWU=0)，则 IDLE 置 1，无论是否通过 WAKE 位选择了静默模式。如果 RWU=1，IDLE 不置 1。

位 3 **ORE**：上溢错误 (Overrun error)

在 RXFF = 1 的情况下，当移位寄存器中当前正在接收的数据

准备好传输到 LPUART_RDR 寄存器时，该位由硬件置 1。通过向 LPUART_ICR 寄存器中的 ORECF 写入 1，该位由软件清零。

如果 LPUART_CR1 寄存器中 RXFNEIE=1 或 LPUART_CR3 寄存器中 EIE = 1，则会生成中断。

1：检测到上溢错误

注意：当该位置 1 时，LPUART_RDR 寄存器的内容不会丢失，但移位寄存器会被覆盖。EIE 位置 1 后，如果在多缓冲区通信中 ORE 标志置 1，则会生成中断。

LPUART_CR3 寄存器中的 OVRDIS 位置 1 时，该位将被永久强制清零（无上溢检测）。

位 2 **NE**：起始位噪声检测标志 (Start bit noise detection flag)

当在接收的帧的起始位上检测到噪声时，该位由硬件置 1。通过向 LPUART_ICR 寄存器中的 NFCF 写入 1，该位由软件清零。

0：未检测到噪声

1：检测到噪声

注意：该位不会生成中断，因为该位出现的时间与本身生成中断的 RXFNE 位出现的时间相同。EIE 位置 1 后，如果在多缓冲区通信中 NE 标志置 1，则会生成中断。

此错误与 LPUART_RDR 中的字符相关联。

位 1 **FE**：帧错误 (Framing error)

当检测到去同步化、过度的噪声或刹车字符时，该位由硬件置 1。通过向 LPUART_ICR 寄存器中的 FECF 写入 1，该位由软件清零。

在智能卡模式下发送数据时，如果在达到最大发送尝试次数后仍未成功（智能卡向数据帧发送 NACK 信号），则该位置 1。

如果 LPUART_CR3 寄存器中 EIE = 1，则会生成中断。

0：未检测到帧错误

1：检测到帧错误或刹车字符

注意：此错误与 LPUART_RDR 中的字符相关联。

位 0 **PE**：奇偶校验错误 (Parity error)

当在接收模式下发生奇偶校验错误时，该位由硬件置 1。通过向 LPUART_ICR 寄存器中的 PECF 写入 1，该位由软件清零。

如果 LPUART_CR1 寄存器中 PEIE = 1，则会生成中断。

0：无奇偶校验错误

1：奇偶校验错误

注意：此错误与 LPUART_RDR 中的字符相关联。

37.7.8 LPUART 中断和状态寄存器 [备用] (LPUART_ISR)

LPUART interrupt and status register [alternate]

偏移地址：0x1C

复位值：0x0000 00C0

同一寄存器可用于使能 FIFO 模式（上一节）和禁止 FIFO 模式（本节）的情况。

禁止 FIFO 模式

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY
									r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	CTS	CTSIF	Res.	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
					r	r		r	r	r	r	r	r	r	r

位 31:23 保留，必须保持复位值。

位 22 **REACK**：接收使能确认标志 (Receive enable acknowledge flag)

LPUART 采用接收使能值时，通过硬件将该位置 1/复位。

该位可用于验证 LPUART 是否准备好在进入低功耗模式前接收数据。

注意：如果 LPUART 不支持从停止模式唤醒功能，该位保留且保持复位值。

位 21 **TEACK**：发送使能确认标志 (Transmit enable acknowledge flag)

LPUART 采用发送使能值时，通过硬件将该位置 1/复位。

通过写入 TE=0 生成空闲帧请求，然后在 LPUART_CR1 寄存器中写入 TE=1 以遵循 TE=0 最短周期时，可使用该位。

位 20 **WUF**：从低功耗模式唤醒标志 (Wake-up from low-power mode flag)

当检测到唤醒事件时，该位由硬件置 1。事件通过 WUS 位域定义。通过向 LPUART_ICR 寄存器中的 WUCF 写入 1，该位由软件清零。

如果 LPUART_CR3 寄存器中 WUFIE=1，则会生成中断。

注意：当 UESM 清零时，WUF 标志也清零。

如果 USART 不支持从停止模式唤醒功能，该位保留且保持复位值。请参见第 1333 页的第 37.3 节：LPUART 实现。

位 19 **RWU**：*接收器从静默模式唤醒 (Receiver wake-up from Mute mode)*

该位指示 LPUART 是否处于静默模式。当识别出唤醒/静默序列时，该位由硬件清零/置 1。静默模式控制序列（地址或 IDLE）通过 LPUART_CR1 寄存器中的 WAKE 位选择。

当选择 IDLE 模式下唤醒时，该位只能通过用软件向 LPUART_RQR 寄存器中的 MMRQ 位写 1 的方式置 1。

0：接收器处于工作模式

1：接收器处于静默模式

注意：如果 LPUART 不支持从停止模式唤醒功能，该位保留且保持复位值。

位 18 **SBKF**：发送刹车标志 (Send break flag)

该位指示已请求发送刹车字符。通过将 1 写入 LPUART_CR3 寄存器中的 SBKRQ 位，该位由软件置 1。该位在刹车发送的停止位期间由硬件自动复位。

0：不发送刹车字符

1：刹车字符已发送

位 17 **CMF**：字符匹配标志 (Character match flag)

接收到由 ADD[7:0] 定义的字符后由硬件将该位置 1。通过向 LPUART_ICR 寄存器中的 CMCF 写入 1，该位由软件清零。

如果 LPUART_CR1 寄存器中 CMIE=1，则会生成中断。

0：未检测到字符匹配

1：检测到字符匹配

位 16 **BUSY**：忙标志 (Busy flag)

该位由硬件置 1 和复位。当 RX 线路上正在进行通信（成功检测到起始位）时有效。在接收结束（成功或失败）时复位。

0：LPUART 处于空闲状态（无接收）

1：正在接收

位 15:11 保留，必须保持复位值。

位 10 **CTS**：CTS 标志 (CTS flag)

该位由硬件置 1/复位。该位是对 CTS 输入引脚的状态取反。

0：CTS 线置 1

1：CTS 线复位

注意：如果不支持硬件流控制功能，该位保留且保持复位值。

位 9 **CTSIF**：CTS 中断标志 (CTS interrupt flag)

如果 CTSE 位置 1，当 CTS 输入切换时，该位由硬件置 1。通过将 1 写入 LPUART_ICR 寄存器中的 CTSCF 位，该位由软件清零。

如果 LPUART_CR3 寄存器中 CTSIE=1，则会生成中断。

0：CTS 状态线上未发生变化

1：CTS 状态线上发生变化

注意：如果不支持硬件流控制功能，该位保留且保持复位值。

位 8 保留，必须保持复位值。

位 7 **TXE**：发送数据寄存器为空 (Transmit data register empty)

当 LPUART_TDR 寄存器的内容已传输到移位寄存器时，TXE 由硬件置 1。通过对 LPUART_TDR 寄存器执行写入操作将该位清零。

如果 LPUART_CR1 寄存器中 TXEIE 位 = 1，则会生成中断。

0：数据寄存器已满/发送 FIFO 已满。

1：数据寄存器/发送 FIFO 未满。

注意：单缓冲区发送期间使用该位。

位 6 **TC**：发送完成 (Transmission complete)

该位指示写入到 USART_TDR 中的最后一个数据已从移位寄存器中发出。如果已完成对包含数据的帧的发送并且 TXE 置 1，TC 标志将会置 1。

如果 LPUART_CR1 寄存器中 TCIE = 1，则会生成中断。

TC 位由软件清零，方法是向 USART_ICR 寄存器中的 TCCF 写入 1 或向 USART_TDR 寄存器执行写操作。

位 5 **RXNE**：读取数据寄存器不为空 (Read data register not empty)

当 LPUART_RDR 移位寄存器的内容已传输到 LPUART_RDR 寄存器时，RXNE 位由硬件置 1。通过对 LPUART_RDR 寄存器执行读入操作将该位清零。

也可以通过向 LPUART_RQR 寄存器中的 RXFRQ 写入 1 来将 RXNE 标志清零。

也可以通过向 LPUART_RQR 寄存器中的 RXFRQ 位写入 1 将 RXFNE 标志位清零。

如果 LPUART_CR1 寄存器中 RXNEIE = 1，则会生成中断。

0：未接收到数据

1：已准备好读取接收到的数据

位 4 **IDLE**：检测到空闲线路 (Idle line detected)

检测到空闲线路时，该位由硬件置 1。如果 LPUART_CR1 寄存器中 IDLEIE=1，则会生成中断。通过向 LPUART_ICR 寄存器中的 IDLECF 写入 1，该位由软件清零。

0：未检测到空闲线路

1：检测到空闲线路

注意：直到 RXNE 位已置 1 时（即，当出现新的空闲线路时）IDLE 位才会被再次置 1。

使能静默模式 (MME=1) 后，如果 LPUART 未静默 (RWU=0)，则 IDLE 置 1，无论是否通过 WAKE 位选择了静默模式。如果 RWU=1，IDLE 不置 1。

位 3 **ORE**：上溢错误 (Overrun error)

在 RXNE = 1 的情况下，当移位寄存器中当前正在接收的数据

准备好传输到 LPUART_RDR 寄存器且 RXNE=1 时（如果使能 FIFO 模式则为 RXFF = 1），该位由硬件置 1。通过向 LPUART_ICR 寄存器中的 ORECF 写入 1，该位由软件清零。

如果 LPUART_CR1 寄存器中 RXNEIE=1 或 LPUART_CR3 寄存器中 EIE = 1，则会生成中断。

1：检测到上溢错误

注意：当该位置 1 时，LPUART_RDR 寄存器的内容不会丢失，但移位寄存器会被覆盖。EIE 位置 1 后，如果在多缓冲区通信中 ORE 标志置 1，则会生成中断。

LPUART_CR3 寄存器中的 OVRDIS 位置 1 时，该位将被永久强制清零（无上溢检测）。

位 2 **NE**：起始位噪声检测标志 (Start bit noise detection flag)

当在接收的帧的起始位上检测到噪声时，该位由硬件置 1。通过向 LPUART_ICR 寄存器中的 NFCF 写入 1，该位由软件清零。

0：未检测到噪声

1：检测到噪声

注意：该位不会生成中断，因为该位出现的时间与本身生成中断的 RXNE/RXFNE 位出现的时间相同。EIE 位置 1 后，如果在多缓冲区通信中 NE 标志置 1，则会生成中断。

在 FIFO 模式下，此错误与 LPUART_RDR 中的字符相关联。

位 1 **FE**：帧错误 (Framing error)

当检测到去同步化、过度的噪声或刹车字符时，该位由硬件置 1。通过向 LPUART_ICR 寄存器中的 FECF 写入 1，该位由软件清零。

在智能卡模式下发送数据时，如果在达到最大发送尝试次数后仍未成功（智能卡向数据帧发送 NACK 信号），则该位置 1。

如果 LPUART_CR3 寄存器中 EIE = 1，则会生成中断。

0：未检测到帧错误

1：检测到帧错误或刹车字符

注意：在 FIFO 模式下，此错误与 LPUART_RDR 中的字符相关联。

位 0 **PE**：奇偶校验错误 (Parity error)

当在接收模式下发生奇偶校验错误时，该位由硬件置 1。通过向 LPUART_ICR 寄存器中的 PECF 写入 1，该位由软件清零。

如果 LPUART_CR1 寄存器中 PEIE = 1，则会生成中断。

0：无奇偶校验错误

1：奇偶校验错误

注意：在 FIFO 模式下，此错误与 LPUART_RDR 中的字符相关联。

37.7.9 LPUART 中断标志清零寄存器 (LPUART_ICR)

LPUART interrupt flag clear register

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUCF	Res.	Res.	CMCF	Res.
											w			w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CTSCF	Res.	Res.	TCCF	Res.	IDLECF	ORECF	NECF	FECF	PECF
						w			w		w	w	w	w	w

位 31:21 保留，必须保持复位值。

位 20 **WUCF**：从低功耗模式唤醒清零标志 (Wake-up from low-power mode clear flag)
将 1 写入该位时，USART_ISR 寄存器中 WUF 标志将清零。
注意：如果 USART 不支持从停止模式唤醒功能，该位保留且必须保持复位值。请参见第 1333 页的第 37.3 节：LPUART 实现。

位 19:18 保留，必须保持复位值。

位 17 **CMCF**：字符匹配清零标志 (Character match clear flag)
将 1 写入该位时，LPUART_ISR 寄存器中的 CMF 标志将清零。

位 16:10 保留，必须保持复位值。

位 9 **CTSCF**：CTS 清零标志 (CTS clear flag)
将 1 写入该位时，LPUART_ISR 寄存器中的 CTSIF 标志将清零。

位 8 保留，必须保持复位值。

位 7 保留，必须保持复位值。

位 6 **TCCF**：发送完成清零标志 (Transmission complete clear flag)
将 1 写入该位时，LPUART_ISR 寄存器中的 TC 标志将清零。

位 5 保留，必须保持复位值。

位 4 **IDLECF**：检测到空闲线路清零标志 (Idle line detected clear flag)
将 1 写入该位时，LPUART_ISR 寄存器中 IDLE 标志将清零。

位 3 **ORECF**：上溢错误清零标志 (Overrun error clear flag)
将 1 写入该位时，LPUART_ISR 寄存器中 ORE 标志将清零。

位 2 **NECF**：检测到噪声清零标志 (Noise detected clear flag)
将 1 写入该位时，LPUART_ISR 寄存器中 NE 标志将清零。

位 1 **FECF**：帧错误清零标志 (Framing error clear flag)
将 1 写入该位时，LPUART_ISR 寄存器中 FE 标志将清零。

位 0 **PECF**：奇偶校验错误清零标志 (Parity error clear flag)
将 1 写入该位时，LPUART_ISR 寄存器中 PE 标志将清零。

37.7.10 LPUART 接收数据寄存器 (LPUART_RDR)

LPUART receive data register

偏移地址：0x24

复位值：0x0000 0000

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

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

位 31:9 保留，必须保持复位值。

位 8:0 **RDR[8:0]**：接收数据值 (Receive data value)

包含接收到的数据字符。

RDR 寄存器在输入移位寄存器和内部总线之间提供了并行接口（请参见*图 452*）。

在使能奇偶校验位的情况下进行接收时，从 MSB 位中读取的值为接收到的奇偶校验位。

37.7.11 LPUART 发送数据寄存器 (LPUART_TDR)

LPUART transmit data register

偏移地址：0x28

复位值：0x0000 0000

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

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

位 31:9 保留，必须保持复位值。

位 8:0 **TDR[8:0]**：发送数据值 (Transmit data value)

包含要发送的数据字符。

TDR 寄存器在内部总线和输出移位寄存器之间提供了并行接口（请参见*图 452*）。

在使能奇偶校验位的情况下（LPUART_CR1 寄存器中的 PCE 位被置 1）进行发送时，由于 MSB 的写入值（位 7 或位 8，具体取决于数据长度）会被奇偶校验位所取代，因此该值不起任何作用。

注意：只能在 TXE/TXFNF=1 时写入此寄存器。

37.7.12 LPUART 预分频器寄存器 (LPUART_PRESC)

LPUART prescaler register

只有在禁止 LPUART (UE=0) 时才能写入此寄存器。

偏移地址：0x2C

复位值：0x0000 0000

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

位 31:4 保留，必须保持复位值。

位 3:0 **PRESCALER[3:0]**：时钟预分频器 (Clock prescaler)

LPUART 输入时钟可通过预分频器进行分频：

0000：输入时钟未分频
0001：输入时钟 2 分频
0010：输入时钟 4 分频
0011：输入时钟 6 分频
0100：输入时钟 8 分频
0101：输入时钟 10 分频
0110：输入时钟 12 分频
0111：输入时钟 16 分频
1000：输入时钟 32 分频
1001：输入时钟 64 分频
1010：输入时钟 128 分频
1011：输入时钟 256 分频
其余组合：保留。

注意：为 PRESCALER 编程不允许的值时，编程的预分频值等于 1011，即输入时钟除以 256。

37.7.13 LPUART 寄存器映射

表 304. LPUART 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	LPUART_CR1 FIFO mode enabled	RXFFIE	TXFEIE	FIFOEN	M1	Res.	Res.	DEAT[4:0]					DEDT[4:0]					Res.	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXFNFIE	TCIE	RXFNEIE	IDLEIE	TE	RE	UESM	UE
	Reset value	0	0	0	0			0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00	LPUART_CR1 FIFO mode disabled	Res.	Res.	FIFOEN	M1	Res.	Res.	DEAT[4:0]					DEDT[4:0]					Res.	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
	Reset value			0	0			0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 304. LPUART 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x04	LPUART_CR2	ADD[7:0]								Res.	Res.	Res.	Res.	MSBFIRST	DATAINV	TXINV	RXINV	SWAP	Res.	STOP [1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADDM7	Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0	0	0					0	0	0	0	0		0	0								0				
0x08	LPUART_CR3	TXFTCFG[2:0]			RXFTIE	RXFTCFG[2:0]			Res.	TXFTIE	WUFIE	WUS 1	WUS 0	Res.	Res.	Res.	Res.	DEP	DEM	DDRE	OVRDIS	Res.	CTSIE	CTSE	RTSE	DMAT	DMAR	Res.	Res.	HDSEL	Res.	Res.	Res.	EIE
	Reset value	0	0	0	0	0	0	0		0	0	0	0					0	0	0	0		0	0	0	0	0			0			0
0x0C	LPUART_BRR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BRR[19:0]																			
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10-0x14	Reserved																																
0x18	LPUART_RQR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFRQ	RXFRQ	MMRQ	SBKRQ	Res.
	Reset value																												0	0	0	0	
0x1C	LPUART_ISR FIFO mode enabled	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RXFF	TXFF	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY	Res.	Res.	Res.	Res.	Res.	CTS	CTSIF	Res.	TXFNF	TC	RXFNE	IDLE	ORE	NE	FE	PE
	Reset value								0	1	0	0	0	0	0	0	0						0	0		1	0	0	0	0	0	0	0
0x1C	LPUART_ISR FIFO mode disabled	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY	Res.	Res.	Res.	Res.	Res.	CTS	CTSIF	Res.	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
	Reset value										0	0	0	0	0	0	0						0	0		1	0	0	0	0	0	0	0
0x20	LPUART_ICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUCF	Res.	Res.	CMCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CTSCF	Res.	Res.	TCCF	Res.	IDLECF	ORECF	NECF	FECF	PECF
	Reset value												0			0								0			0		0	0	0	0	0
0x24	LPUART_RDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDR[8:0]								
	Reset value																								0	0	0	0	0	0	0	0	0
0x28	LPUART_TDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDR[8:0]								
	Reset value																								0	0	0	0	0	0	0	0	0
0x2C	LPUART_ PRESC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESCALER[3:0]			
	Reset value																													0	0	0	0

有关寄存器边界地址的信息，请参见*第 2.2 节*。

38 串行外设接口 (SPI)

38.1 简介

串行外设接口 (SPI) 可使用特定同步协议与外部器件进行通信。SPI 协议支持与外部器件进行半双工、全双工和单工同步串行通信。该接口可配置为主模式或从模式，并且能够在多从或多主配置中运行。配置为主器件的器件为从器件提供通信时钟 (SCK)。可以选择性地应用从器件选择 (SS) 和就绪 (RDY) 信号，仅用于与具体从器件建立通信，并确保其正确处理数据流。默认情况下使用 Motorola 数据格式，但也支持某些其他特定模式。

38.2 SPI 主要特性

- 基于三条线的全双工同步传输
- 基于双线的半双工同步传输，其中一条可作为双向数据线
- 基于双线的单工同步传输，其中一条可作为单向数据线
- 可选择 4 位到最多 32 位的数据大小，或 8 位倍数的固定大小
- 多主或多从模式功能
- 双时钟域，外设内核时钟独立于 APB 总线时钟
- 波特率预分频器高达内核频率的一半，或者，在主模式下，可从 RCC 旁路
- 保护配置和设置
- 对于主模式和从模式都可以通过硬件或者软件进行 SS 管理
- 数据之间的最小延时，以及 SS 与数据流之间的最小延时均可调
- 可配置的 SS 信号极性和时序，MISO x MOSI 交换功能
- 可编程的时钟极性和相位
- 可编程的数据顺序，最先移位 MSB 或 LSB
- 可以对传输的数据量进行编程，以控制 SS 和 CRC
- 可触发中断的专用发送和接收标志
- SPI Motorola 和 TI 格式支持
- 硬件 CRC 功能可在事务结束时通过以下方式验证通信完整性：
  - 在发送模式下添加 CRC 值
  - 在接收模式下自动进行 CRC 错误校验
- 在数据上溢、CRC 错误、数据下溢、模式故障和帧错误（具体取决于操作模式）的情况下可触发中断的错误检测
- 两个嵌入式 8 位倍数接收和发送 FIFO（FIFO 大小取决于实例）
- 可配置的 FIFO 阈值（数据封装）
- 可由系统 DMA 控制器处理数据流
- 从模式下，下溢条件可配置（支持级联循环缓冲区）
- 可选的状态引脚 RDY 信号指示从器件已准备就绪，可处理数据流

38.3 SPI 实现

表 305 描述 SPI 实现。这些实例以全面的功能集或有限功能集应用。

表 305. SPI 特性

SPI 特性	SPI2S1、SPI2S2、SPI2S3 （全面的功能集实例）
数据和 CRC 大小	可配置为 4 至 32 位
CRC 计算	CRC 多项式长度可配置为 5 至 33 位
FIFO 的大小	16x8 位
数据控制数量 (TSIZE)	多达 65536
I2S 功能	是
在具有唤醒功能的停止模式下具有自主功能	否
在具有唤醒功能的 LP-Stop 和 待机模式下具有自主功能	否

注意： *有关退出停止和待机模式的实例功能的详细信息，请参见表 309：SPI 唤醒和中断请求。*

38.4 SPI 功能描述

38.4.1 SPI 框图

SPI 支持在 MCU 与外部器件之间进行同步串行通信。应用软件可通过轮询状态标志或使用专用 SPI 中断对通信进行管理。主要 SPI 元件及其交互如*图 466* 所示。

图 466. SPI/I2S 框图

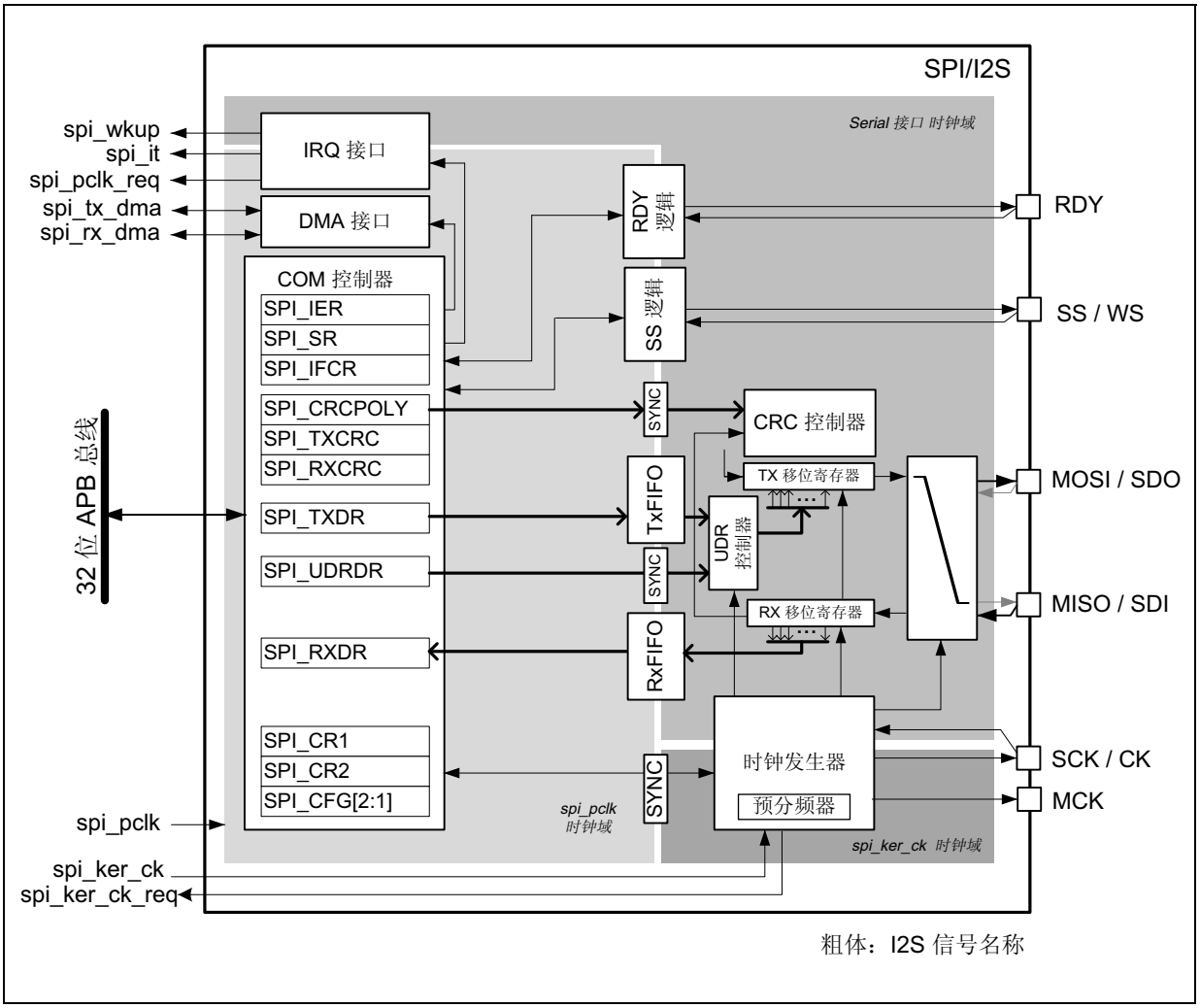


图 466 的简图显示了三个完全独立的时钟域：

- **spi_pclk** 时钟域
- **spi_ker_ck** 内核时钟域
- 串行接口时钟域

这些域之间的所有控制和状态信号均严格同步。这些时钟信号之间的频率比没有特定限制。用户必须考虑使用与数据流速度兼容的比率，以避免数据下溢或上溢。

spi_pclk 时钟信号为外设总线（APB 总线）提供时钟。需要访问 SPI 寄存器时，该信号必须有效。

SPI 在从器件模式下工作时，会使用源自外部 SCK 信号（由外部主 SPI 器件提供）的串行接口时钟来处理数据流。在这种情况下，即使 **spi_pclk** 和 **spi_ker_ck** 时钟信号无效，SPI 从器件也能接收和发送数据。因此，工作在从模式下的 SPI，如果需要设置寄存器的标志位，那么需要提供外设总线时钟（例如：出现上溢或下溢时，有关详细信息，请参见*第 38.5.2 节*）。总线进入空闲状态时，不能进行上述过程。在某些情况下，从器件甚至需要时钟发生器工作（请参见*第 38.5.1 节*）。

当 SPI 作为主接口时，来自 RCC 的 **spi_ker_ck** 内核时钟在通信期间需要处于活动状态，以便通过时钟发生器提供串行接口时钟，可以在此通过预分频器对其进行分频，也可以将其旁路（可选）。然后，该信号会通过 SCK 引脚提供给从接口，并在内部提供给主接口的串行接口域。

38.4.2 SAI 引脚和内部信号

有多达五个 I/O 引脚专用于与外部器件进行 SPI 通信。

- **MISO**：主输入 / 从输出数据。通常情况下，此引脚用于在从模式下发送数据和在主模式下接收数据。
- **MOSI**：主输出 / 从输入数据。通常情况下，此引脚用于在主模式下发送数据和在从模式下接收数据。
- **SCK**：SPI 主器件的串行时钟输出引脚以及 SPI 从器件的串行时钟输入引脚。
- **SS**：从器件选择引脚。根据 SPI 和 SS 设置，该引脚可用于：
  - 选择单个从器件以进行通信
  - 同步数据帧，或
  - 检测多个主器件之间是否存在冲突

  详细信息，请参见*第 38.4.7 节*。
- **RDY**：状态引脚（可选）发出信号以指示从器件 FIFO 的占用情况，以便从器件能继续处理事务，而不会出现任何数据流损坏的风险。主器件可对其进行检查，从而对进行中通信的临时挂起加以控制。

所有这些引脚（RDY 除外）在 I2S 模式下均可共用。该模式具有额外的 I2S 特定 MCK 信号。有关 I2S 信号的详细信息，请参见*第 38.9.2 节*。

SPI 总线支持一个主器件与一个或多个从器件之间进行通信。该总线至少由两条线构成：一条用于时钟信号，另一条用于同步数据传输。其他信号为可选信号，可以根据 SPI 节点间的数据交换及其通信控制管理进行添加。

有关 SPI 输入/输出引脚和内部信号的列表，请参见*表 306* 和*表 307*。

表 306. SPI/I2S 输入/输出引脚(1)

引脚名称	I/O 类型	说明
MISO/SDI(2)	输入/输出	主数据输入/从数据输出
MOSI/SDO(2)	输入/输出	主数据输出/从数据输入
SCK/CK	输入/输出	主时钟输出/从时钟输入
SS/WS	输入/输出	主输出/从选择输入

表 306. SPI/I2S 输入/输出引脚(1)

引脚名称	I/O 类型	说明
RDY	输入/输出	SPI 主输入/从 FIFO 状态占用输出
MCK	输出	I2S 主模式频率输出

1. 有关详细信息，请参见*第 38.9.2 节：与 SPI 功能共享的引脚*部分。
2. MOSI/SDO 与 MISO/SDI 引脚的功能可以互换。在 SPI 双向半双工模式下，它们的方向可能会有所不同。

SPI 输入/输出信号说明

表 307. SPI 内部输入/输出信号

信号名称	信号类型	说明
spi_pclk	输入	SPI 时钟信号为外设总线接口提供时钟。
spi_ker_ck	输入	SPI 内核时钟
spi_ker_ck_req	输出	SPI 内核时钟请求
spi_pclk_req	输出	SPI 时钟请求
spi_wkup	输出	SPI 提供唤醒中断
spi_it	输出	SPI 全局中断
spi_tx_dma	输入/输出	SPI 发送 DMA 请求
spi_rx_dma	输入/输出	SPI 接收 DMA 请求

SPI 互连说明

38.4.3 SPI 通信一般情况

SPI 支持 MCU 基于目标器件和应用要求使用不同的配置进行通信。这些配置使用 2 条或 3 条线（通过软件 SS 管理），也可以使用 3 条或 4 条线（通过硬件 SS 管理）。通信始终由主器件发起和控制。主器件基于 SCK 线提供时钟信号，并在由硬件进行管理时通过 SS 线选择或同步从器件以实现通信。主从器件之间通过 MOSI 和/或 MISO 线实现数据流动。

38.4.4 一个主器件和一个从器件之间的通信

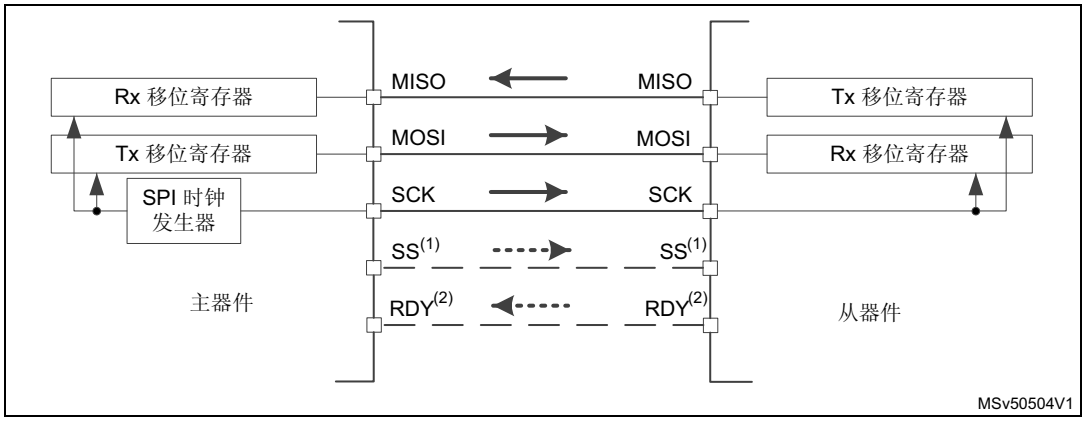
通信流可使用以下 3 种模式之一：全双工（3 线）模式、半双工（2 线）模式或单工（2 线）模式。SS 信号在单一主从器件配置下是可选信号，并且一般不使用。不过，SS 信号在此配置下可有助于同步数据流，并在默认情况下用于某些特定 SPI 模式（例如，TI 模式）。

下一个可选的 RDY 信号可以帮助确保在从器件侧正确管理所有处理的数据。

全双工通信

默认情况下，SPI 配置为全双工通信（SPI_CFG2 寄存器中的位 COMM[1:0] = 00）。在这种配置下，主器件和从器件的移位寄存器通过 MOSI 和 MISO 引脚之间的两条单向线连接。在 SPI 通信过程中，数据随主器件提供的 SCK 时钟边沿同步移位。主器件通过 MOSI 线将待发送的数据发送给从器件，同时通过 MISO 线从从器件接收数据。当数据帧传输完成时（所有位均移出），主器件和从器件之间即完成信息交换。

图 467. 全双工单个主器件/单个从器件应用

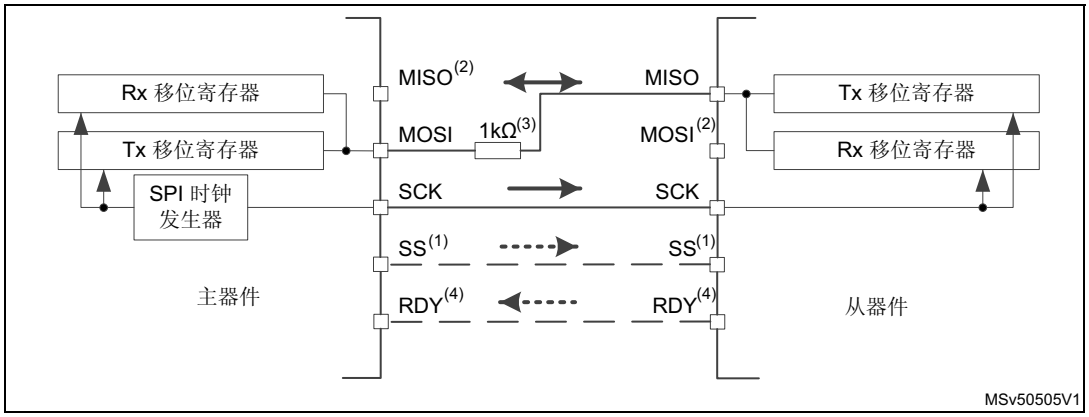


1. 要使 SPI 接口工作，并非一定要应用 SS 引脚互连（有关详细信息，请参见*第 38.4.7 节*）。
2. 从器件提供的 RDY 信号可以由主器件选择性地读取。

半双工通信

通过设置 SPI_CFG2 寄存器中的 COMM[1:0] = 11，SPI 可采用半双工模式进行通信。在这种配置下，使用一条交叉连接线将主器件和从器件的移位寄存器连接起来。在此通信过程中，数据随 SCK 时钟边沿在移位寄存器之间进行移位，传输方向由主器件和从器件通过各自 SPI_CR1 寄存器中的 HDDIR 位进行选择。请注意，更改通信方向时，必须禁止 SPI。在该配置下，主器件的 MISO 引脚和从器件的 MOSI 引脚空闲，可在其他应用中用作 GPIO。

图 468. 半双工单个主器件/单个从器件应用



1. 要使 SPI 接口工作，并非一定要应用 SS 引脚互连（有关详细信息，请参见*第 38.4.7 节*）。
2. 在该配置下，主器件的 MISO 引脚和从器件的 MOSI 引脚可用作 GPIO。
3. 如果在双向模式下工作的两个节点之间的通信方向未同步更改，可能会发生严重情况。新发送器会访问公共数据线，而前一个发送器仍然在线路上保留相反值（该值取决于 SPI 配置和通信数据）。节点可能会暂时与线路上的相反输出电平发生冲突，直到前一个发送器更改其数据方向设置。建议此模式下在 MISO 和 MOSI 引脚之间插入串行电阻以在这种情况下保护冲突 输出并限制电流在二者之间流过。
4. 从器件提供的 RDY 信号可以由主器件选择性地读取。

单工通信

通过 SPI_CFG2 寄存器中的 COMM[1:0] 位域将 SPI 设置为只发送模式或只接收模式，可使 SPI 以单工模式进行通信。在这种配置下，仅使用一条线在主器件和从器件的移位寄存器之间进行传输。其余 MISO 或 MOSI 引脚对不用于通信，可用作标准 GPIO。

- **只发送模式：COMM[1:0] = 01**

  只要 TxFIFO 中存在可用数据，并且主器件传输正在进行，只发送模式下的主器件就会生成时钟。

  如果只发送模式下的从器件接收到 SCK 引脚上的时钟并且 SS 引脚（或软件管理的内部信号）有效，则该从器件会发送数据（请参见*第 38.4.7 节*）。

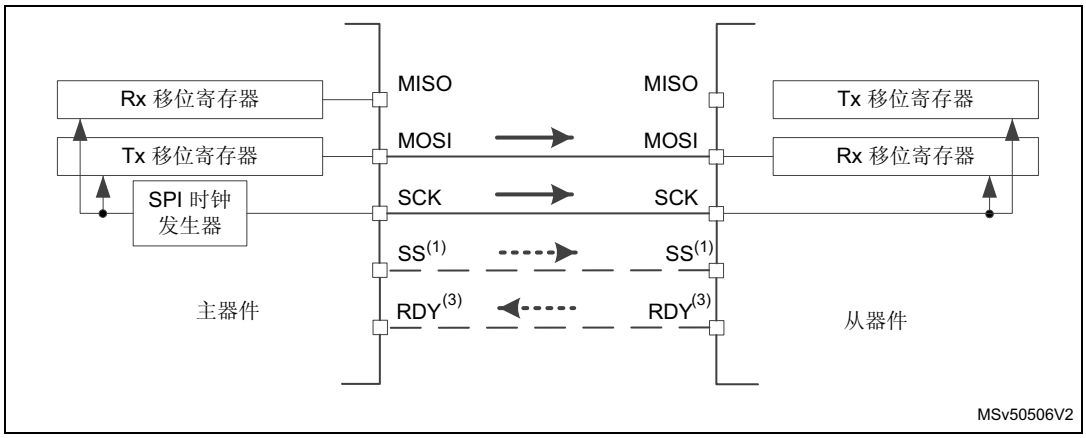
- **只接收模式：COMM[1:0] = 10**

  在主模式下，MOSI 输出被禁止，并可用作 GPIO。只要 SPI 使能并且 SPI_CR1 寄存器中的 CSTART 位置 1，就会持续生成时钟信号。时钟既可以由软件通过将 SPI_CR1 寄存器的 CSUSP 位置 1 进行明确请求来停止，也可以在 RxFIFO 已满时（SPI_CR1 的 MASRX 位置 1 的情况下）自动停止。

  在从器件配置下，MISO 输出被禁止，该引脚可用作 GPIO。当从器件选择信号有效时，从器件继续从 MOSI 引脚接收数据（请参见*第 38.4.7 节*）。

注意： *无论在何种主模式和从模式下，通过更改 SPI_CFG2 寄存器中的 IOSWP 位值，可将专用于发送的数据引脚替换为专用于接收的数据引脚，反之亦然（该位只能在 SPI 禁止时进行修改）。*
任何单工通信均可以通过半双工通信的一种变型来替换，该变型中设置的数据传输方向不变（在 HDDIR 位保持不变的同时，双向模式处于使能状态），在忽略未使用的数据线和对应数据流时，也可以由全双工控制来替换。

图 469. 单工单个主器件/单个从器件应用（主器件处于只发送模式/从器件处于只接收模式）



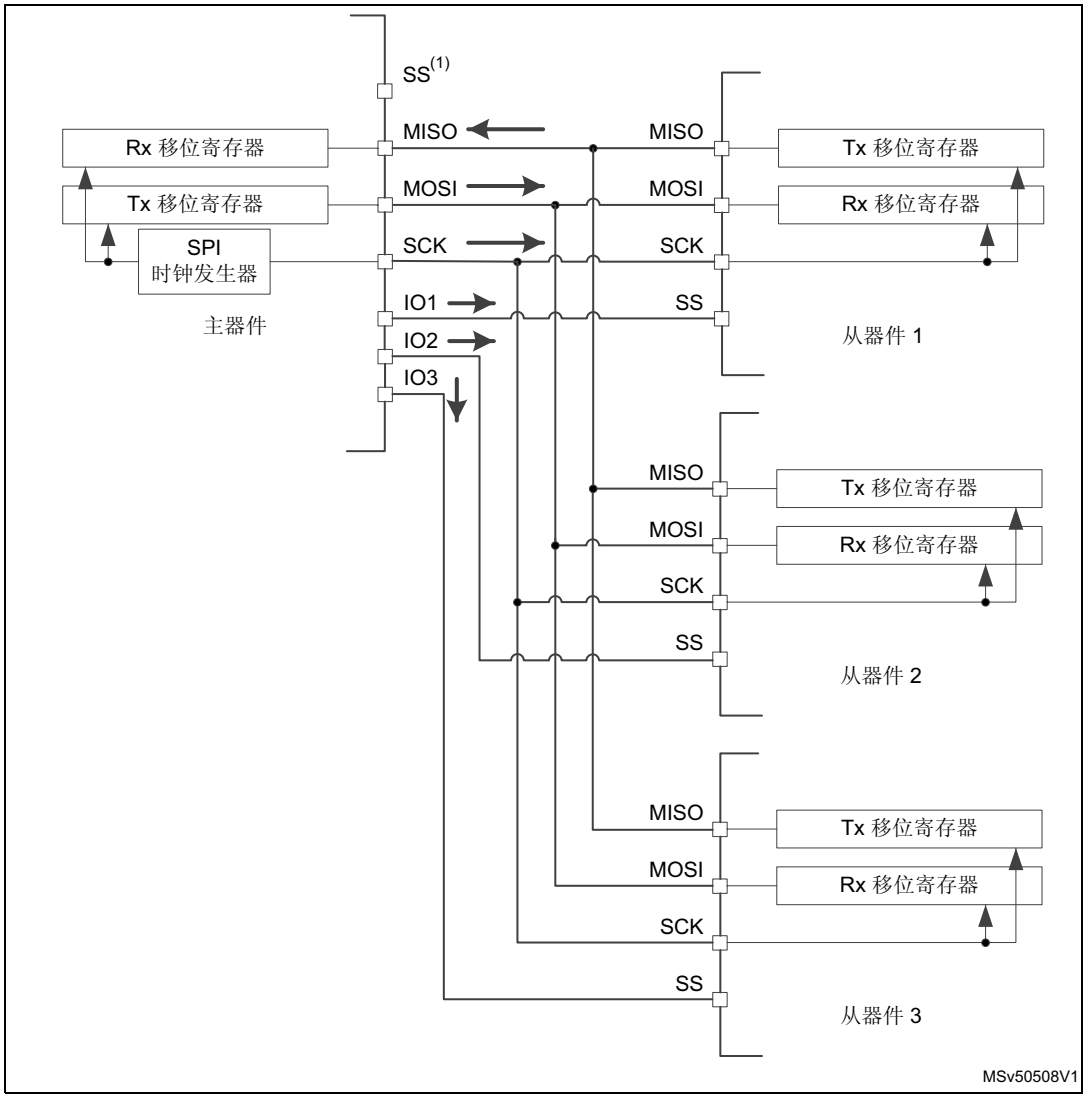
1. 要使 SPI 接口工作，并非一定要应用 SS 引脚互连（请参见*第 38.4.7 节*）。
2. 在这种配置下，两个 MISO 引脚均可用作 GPIO。
3. 从器件提供的 RDY 信号可以由主器件选择性地读取。

38.4.5 标准多从器件通信

在具有两个或多个独立从器件的配置下，主器件使用星型拓扑和专用的 GPIO 引脚来分别管理每个从器件的片选线（请参见*图 470.*）。主器件必须通过拉低与从器件 SS 输入相连的 GPIO 的电平来单独选择一个从器件（在指定时间，只有一个从器件可以控制共用 MISO 线上的数据）。完成上述操作后，随即会在主器件和所选从器件之间建立通信。这种拓扑除了

具有简便性优势外，还支持对各个从器件应用特定 SPI 配置，因为所有通信会话均在一个主器件-从器件对内单独进行。此外，如果无需从从器件中读取任何信息，主器件可以将相同的信息发送到多个从器件。

图 470. 使用星形拓扑连接的主器件和三个独立从器件



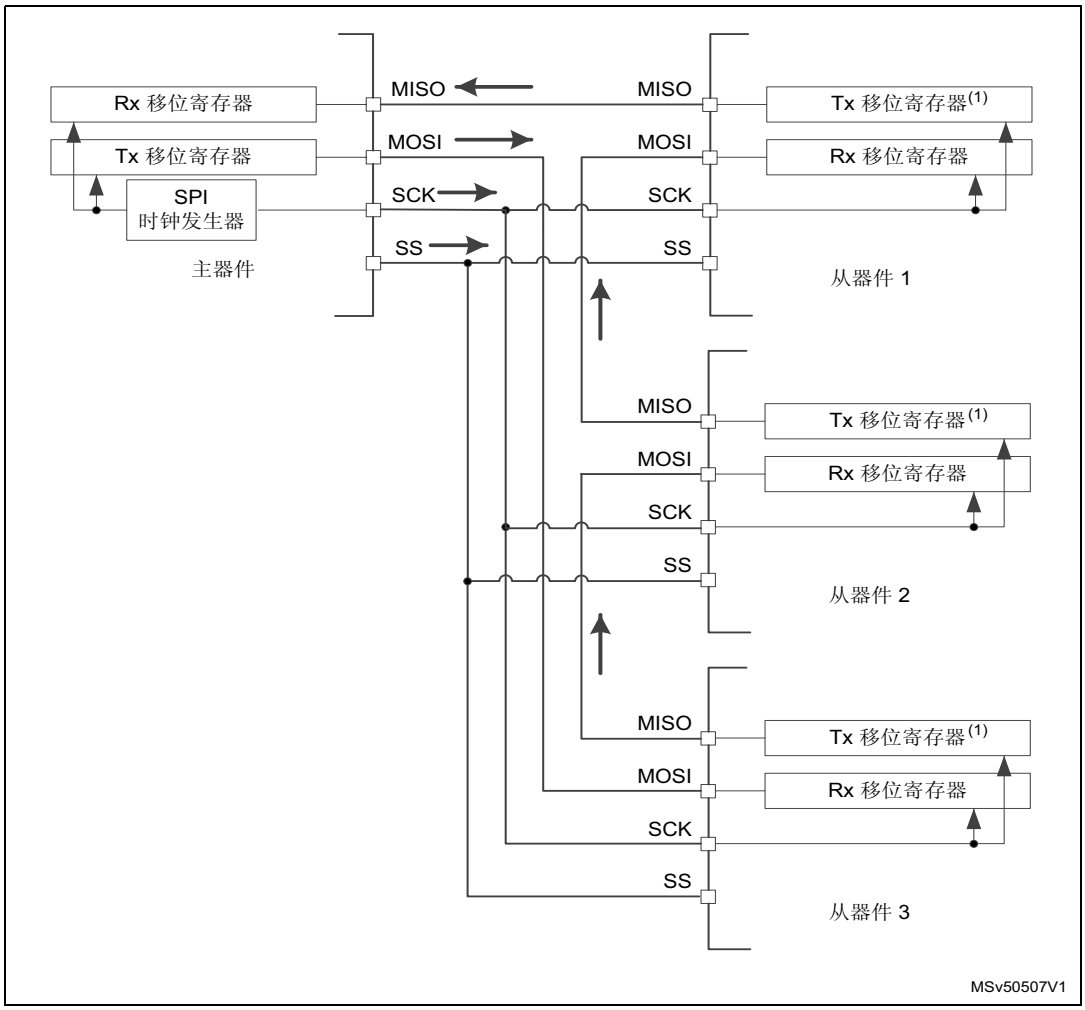
1. 主器件单个 SS 引脚硬件输出功能无法支持此拓扑（由软件控制下的一组 GPIO 代替），用户应避免在引脚处进行 SPI AF 设置（有关详细信息，请参见*第 38.4.7 节*）。
2. 如果应用程序无法确保主器件在给定时间提供不超过一个 SS 有效信号，则最好将 MISO 引脚配置为开漏配置，并在 MISO 线上外接上拉电阻，以防止互连的从器件输出之间发生冲突。另外，可应用推挽配置而无需额外电阻（请参见 I/O 复用功能输入/输出 (GPIO) 部分）。
3. 主器件可以选择性地从从器件读取 RDY 信号。

当采用环形拓扑时，主器件可以与所有从器件进行 SPI 通信（请参见*图 471*）。所有从器件类似于串行菊花链上的简单移位寄存器，通过共用的从器件选择 (SS) 信号和时钟 (SCK) 信号进行控制。所有信息在环上同时移位，然后返回到主器件。会话的长度是固定的，主器件与从器件数处理的数据帧数相同。

在菊花链中处理第一个数据帧时，主器件只通过第一个从节点输入向菊花链中最后一个从节点发送信息，主器件接收的第一条信息来自最后一个节点输出。相应地，用来结束会话的最

后处理的数据发送给第一个从节点，同时其最先传出的数据在会话期间通过菊花链上所有其他从器件正好到达主器件输入。
采用该拓扑时，菊花链中所有节点的数据格式配置和时钟设置都必须相同。由于接收和发送移位寄存器在内部是彼此分开的，因此当在接收器和发射器之间通过硬件交互信息时，必须使用 TxFIFO 从器件的下溢功能来完成。
在这种情况下，传输下溢功能被配置为重复最后接收到的数据帧的模式 (UDRCFG=1)。在会话开始之前，每个从器件把单个数据写入 TxFIFO，然后发起会话传输（通常用到从器件状态信息）。在这种情况下，下溢实际上发生在处理完第一个数据帧之后。为了能够立即清除内部下溢条件，并再次通过 TxFIFO 内容重新启动会话，用户必须先禁止再使能会话之间的 SPI，且必须通过新的单个数据模式填充 TxFIFO（以克服清除时造成的传播延时，通过 UDRC 位以标准方式清除下溢时会引起这种延时）。

图 471. 一主三从环形连接（菊花链）拓扑



1. 在该配置下，从器件使用下溢特性，一旦从器件的 TxFIFO 变空，从器件可以将先前接收的数据发送到 Rx 移位寄存器中。
2. RDY 信号可以由主器件选择性地单独读取，也可以配置为开漏输出（当 RDIOP=0 时），并以共用链就绪状态与由最慢器件过度驱动的上拉电阻连接。

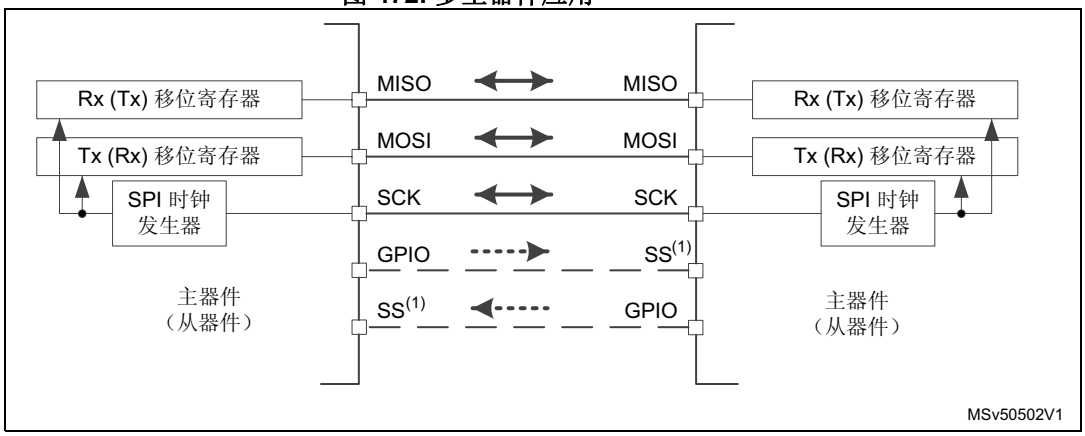
38.4.6 多主器件通信

如果 SPI 总线并非主要用于多主功能，可使用内置功能来检测尝试同时控制总线的两个节点间是否存在潜在冲突。对于该检测，SS 引脚配置为硬件输入模式。由于在指定时间只有一个结点可将其输出施加到公用数据线上，因此无法连接超过两个以此模式工作的 SPI 节点。

当节点无效时，默认情况下均保持从模式。如果一个节点要接管对总线的控制，它会将自身切换到主模式，然后通过专用 GPIO 引脚向其他节点的从器件选择输入施加有效电平。会话完成后，有效的从器件选择信号将被释放，控制总线的节点会短暂切换回被动从模式，等待下一个会话开始。

如果两个节点同时发出各自的控制请求，则会出现总线冲突（请参见模式故障 MODF 事件）。用户可应用某个简单的仲裁过程（例如，在两个节点中施加不同的预定义超时来推迟下一个尝试）。

图 472. 多主器件应用



1. 在两个节点上，SS 引脚都配置为硬件输入模式。当被动节点配置为从器件时，其有效电平将使能 MISO 线输出控制。
2. 该通信中不使用 RDY 信号。

38.4.7 从器件选择 (SS) 引脚管理

在从模式下，SS 用作标准的“片选”输入，使从器件与主器件进行通信。在主模式下，SS 可用作输出或输入。用作输入时，可防止多主模式总线冲突；用作输出时，可驱动单个从器件的从器件选择信号。SS 信号可在内部进行管理（SS 输入信号的软件管理），或在 SS 输入和输出均与 SS 引脚相关联时在外部进行管理（硬件 SS 管理）。用户可以通过 SSIOP 位设置来配置（SS 引脚上）输入/输出外部信号的有效电平。如果 SS 电平等于 SSIOP，则将其视为处于活动状态。

可以使用 SPI_CFG2 寄存器中的 SSM 位设置硬件或软件从器件选择管理：

- **软件 SS 管理 (SSM = 1)**：在这种配置下，由 SPI_CR1 寄存器中的 SSI 位的值内部驱动从器件选择信息。外部 SS 引脚空闲，可供其他应用使用（用作 GPIO 或其他复用功能）。
- **硬件 SS 管理 (SSM = 0)**：在这种情况下，可行的配置有两种：所用配置取决于 SS 输出配置（SPI_CFG2 寄存器中的 SSOE 位）。
  - **SS 输出使能 (SSOE = 1)**：仅在将 MCU 设置为主器件时才使用该配置。SS 引脚由硬件管理。该功能与 CSTART 和 EOT 控制相关。因此，主器件必须应用适当的 TSIZE>0 设置来正确控制 SS 输出。即使 SPI AF 未应用于 SS 引脚（则可用作标准 GPIO），仍要保持 SSOE = 1，以确保默认 SS 输入电平，并防止在主 SS 内部逻辑（仅适用于多主拓扑）的输入处进行任何模式故障评估。

  a) 如果 SSOM = 0 且 SP = 000，则在主器件传输开始 (CSTART = 1) 时，SS 信号会被驱动至有效电平，并始终保持有效，直到其 EOT 标志置 1 或传输进入挂起状态。

  b) 当 SP = 001 时，根据 TI 模式定义生成脉冲。

  c) 当 SSOM = 1、SP = 000 且 MIDI>1 时，SS 在各个数据帧之间的脉冲无效，并在 MIDI 值减 1 得到的值所定义的 SPI 时钟周期（1 到 14）内保持无效。

  d) SS 输入在主器件内部被强制为非活动状态，以防止其任何模式故障。

  – **SS 输出禁止（SSM = 0，SSOE = 0）：**

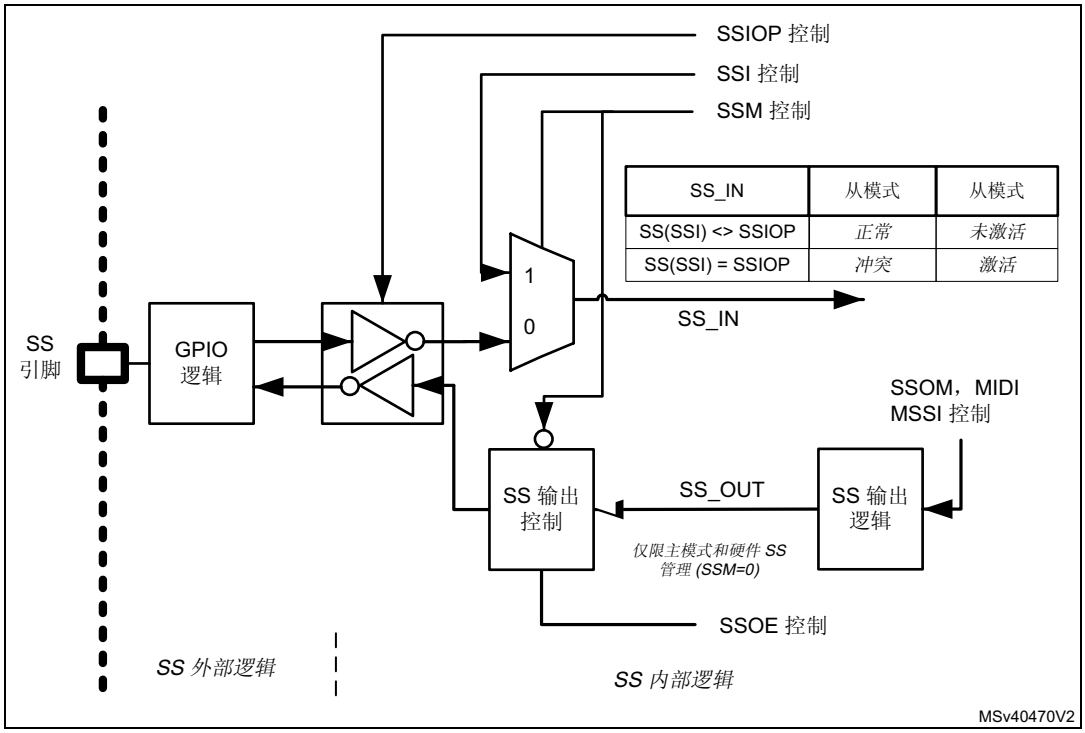
  a) 如果微控制器在总线上用作主器件，则此配置可实现多主模式功能。如果在该模式下将 SS 引脚拉至有效电平，SPI 将进入主模式故障状态，SPI 器件将在从模式 (MASTER = 0) 下自动进行重新配置。

  b) 在从模式下，SS 引脚用作标准的“片选”输入，当 SS 线为有效电平时将选择此从器件。

注意： *在模式故障条件下自动切换到从模式可避免数据和时钟线上发生潜在冲突。由于 SPE 在这种情况下自动复位，接收和发送 FIFO 均会刷新，并且当前数据会丢失。*

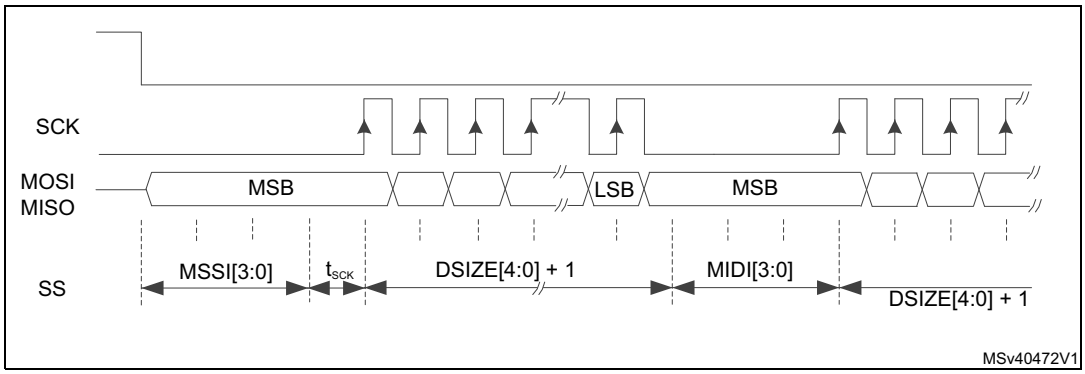
当 SPI 从器件在硬件 SS 管理模式下使能时，即使发现 SS 处于有效电平，所有数据传输也会被忽略。数据传输将被忽略，直到从器件检测到 SS 信号的开始（从非有效电平转换为有效电平），使从器件与主器件同步。因此，外部 SS 引脚固定时不能使用硬件管理模式。SS 软件管理不提供此类保护功能。总线上无通信时，必须更改 SSI 位，在传输之间 SCK 信号处于空闲状态电平（仅限这种情况）。

图 473. SS 控制逻辑原理图



应用硬件输出 SS 控制（SSM = 0，SSOE = 1）时，用户可通过配置 MIDI[3:0] 和 MSSI[3:0] 位域来控制数据帧之间的 SS 信号时序，以及插入每次事务开始时的额外延时（以分隔 SS 和时钟启动）。从器件需要减慢数据流速以获得充足的空间来正确处理数据时，这十分有用（请参见*图 474*）。

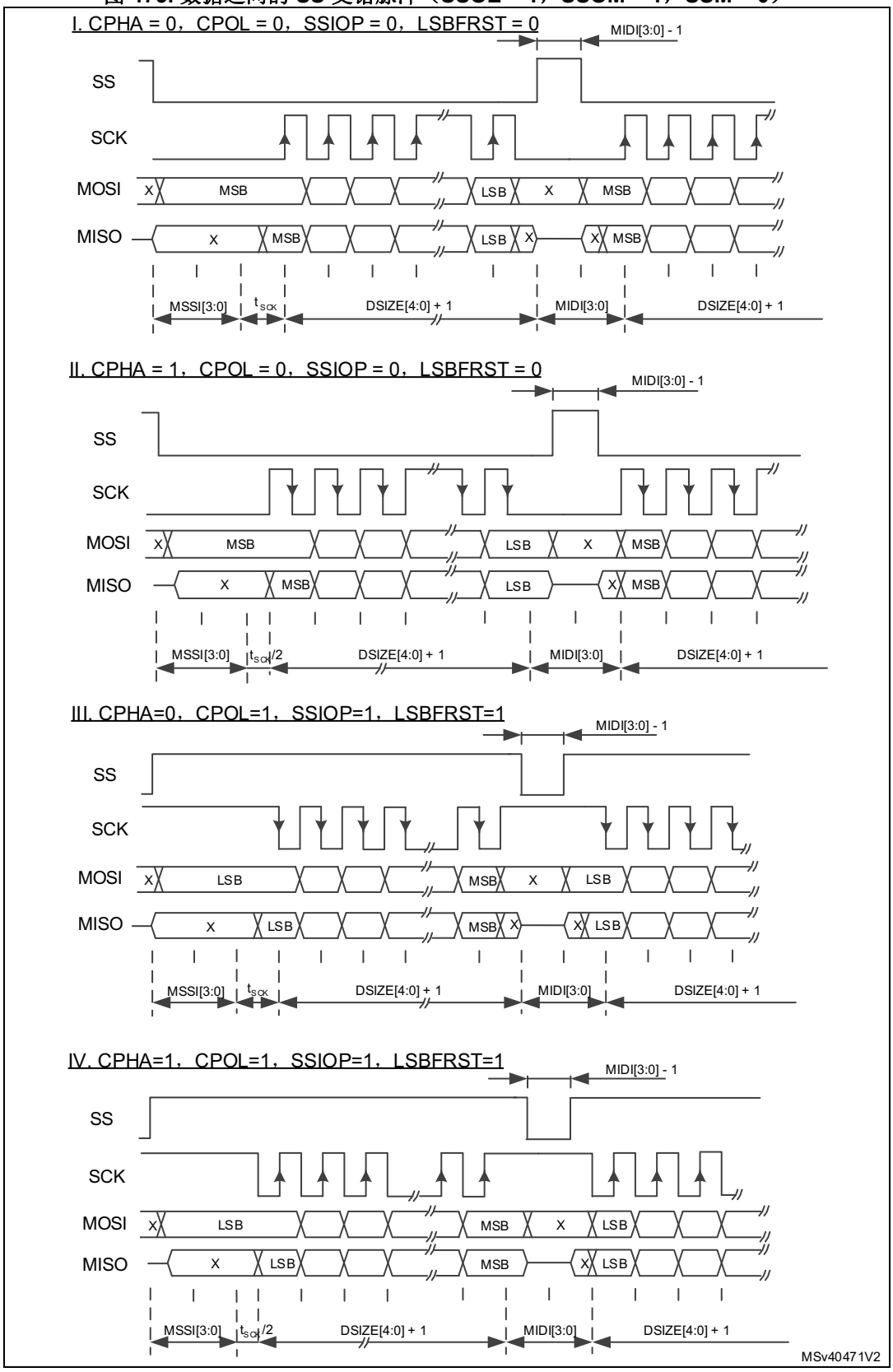
图 474. 数据流时序控制（SSOE = 1，SSOM = 0，SSM = 0）



1. MSSI[3:0] = 0011，MIDI[3:0] = 0011（MIDI[3:0] = 0 时，SCK 流连续）。
2. CPHA = 0，CPOL = 0，SSIOP = 0，LSBFRST = 0。

此外，位 SSOM =1 设置会调用特定模式，在该模式下，如果有足够的空间可用则会在数据帧之间加入交错脉冲（必须将 MIDI[3:0] 设置为大于一个 SPI 周期的值）。*图 475* 给出了一些配置示例。

图 475. 数据之间的 SS 交错脉冲（SSOE = 1，SSOM = 1，SSM = 0）



1. MSSI[3:0] = 0010，MIDI[3:0] = 0010。
2. 当 MIDI[3:0] > 1 个交错脉冲宽度时，数据之间的 SS 交错脉冲始终比提供的帧间隙（由 MIDI 参数定义）少一个 SCK 周期。如果设置了 MIDI，则帧由单独一个 SCK 周期分隔，但 SS 上不会出现交错脉冲。

38.4.8 就绪引脚 (RDY) 管理

从器件数据处理能力的状态可以通过 RDY 引脚进行检查。默认情况下，低电平表示从器件尚未准备好进行事务处理。原因可能是从器件的 TxFIFO 为空、RxFIFO 已满或 SPI 被禁止。信号的有效电平可以通过 RDIOP 位来选择。如果主器件在从器件指示未就绪状态时继续或开始与之通信，则事务极有可能会失败。

控制 RDY 输出的逻辑相当复杂，与 TSIZE 和 DSIZE 设置密切相关。RDY 反应更悲观，在帧事务期间，对 TxFIFO 几乎变空和/或 RxFIFO 几乎变满更敏感。尽管 TxFIFO 完全为空和/或 RxFIFO 完全被占用，但仅当 RDY 保持有效时，这种悲观逻辑才会在事务结束时受到抑制，以防止任何数据损坏，并及时通知主器件需要暂时挂起事务，直到可以再次安全地处理下一个已处理数据为止。当在主器件端使能 RDY 信号输入时，一旦从器件指示未就绪状态，主器件便会将通信挂起。这样做可以防止主器件完成进行中帧的事务，该事务仅清空从器件的 TxFIFO 或将其 RxFIFO 填满，直到在其中写入和/或读取下一个数据为止（尽管该帧仍然可以不受约束地完成）。如果在从器件上应用 TSIZE = 0 配置，则可能会出现问题，因为从器件始终不会评估事务结束（这会在发送最后一个数据时抑制未就绪状态）。这样，用户就必须通过软件在从器件侧释放 RxFIFO 并/或向 TxFIFO 写入额外（甚至是空的）数据，以释放非 RDY 信号，解锁 ST 主器件，使其能够在通信偶尔在帧中间挂起时继续。

当主器件不使用 RDY 时，必须将其禁止 (RDIOM = 0)。然后主器件的内部逻辑会模拟从器件的始终就绪状态。在这种情况下，RDIOP 位设置没有任何意义。

由于时钟域之间的同步以及主从端的 RDY 逻辑评估，RDY 引脚功能不再可靠，并且当数据帧的大小配置为不足 8 位时无法使用。

38.4.9 通信格式

SPI 通信过程中，将同时执行接收和发送操作。串行时钟 (SCK) 对数据线上的信息的移位和采样进行同步。通信格式取决于时钟相位、时钟极性和数据帧格式。为了能够在彼此间进行通信，主器件和从器件必须遵循相同的通信格式并且必须正确同步。

时钟相位和极性控制

通过 SPI_CFG2 寄存器中的 CPOL 和 CPHA 位，可以用软件选择四种可能的时序关系。CPOL（时钟极性）位控制不传输任何数据时时钟的空闲状态值。该位对主器件和从器件都有作用。如果复位 CPOL，SCK 引脚在空闲状态处于低电平。如果将 CPOL 置 1，SCK 引脚在空闲状态处于高电平。

如果将 CPHA 位置 1，则会在 SCK 引脚的第二个边沿捕获传输的第一个数据位（如果复位 CPOL 位，则为下降沿；如果将 CPOL 位置 1，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。如果将 CPHA 位复位，则会在 SCK 引脚的第一个边沿捕获传输的第一个数据位（如果将 CPOL 位置 1，则为下降沿；如果将 CPOL 位复位，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。

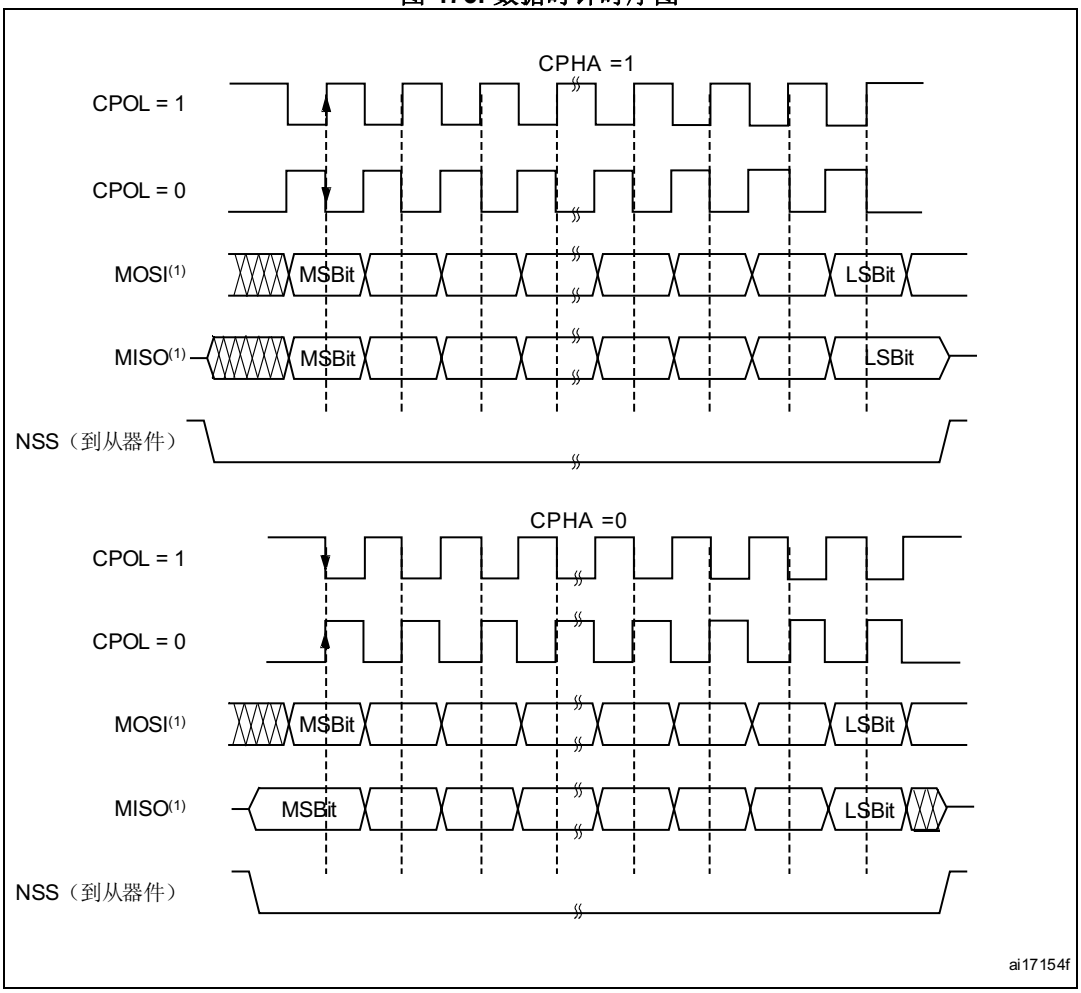
CPOL（时钟极性）和 CPHA（时钟相位）位的组合用于选择数据捕获时钟边沿（*图 476* 中的虚线）。

图 476 给出了在 CPHA 和 CPOL 位的四种组合下的 SPI 全双工传输。

注意： *在切换 CPOL/CPHA 位之前，必须通过复位 SPE 位来禁止 SPI。*

SCK 的空闲状态必须与 SPI_CFG2 寄存器中选择的极性相对应（如果 CPOL = 1，则上拉 SCK 引脚；如果 CPOL = 0，则下拉 SCK 引脚）。

图 476. 数据时钟时序图



1. 数据位的顺序取决于 LSBFRST 位的设置。

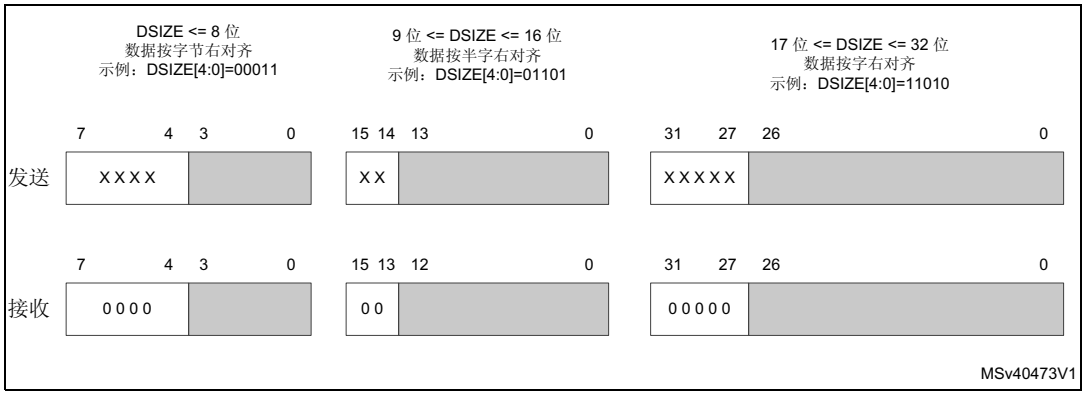
数据帧格式

SPI 移位寄存器可设置为以 MSB 在前或 LSB 在前的方式移出数据，具体取决于 SPI_CFG2 寄存器的 LSBFRST 位的值。

在具有完整功能集的实例中，使用 SPI_CFG1 寄存器中的 DSIZE[4:0] 位来选择数据帧大小。数据帧的长度可设置为 4 位到 32 位，且此设置对于发送和接收均适用。访问 SPI_TXDR/SPI_RXDR 寄存器时，数据帧始终按字节（数据不超过一个字节时）、半字或字进行右对齐（请参见*图 477*）。

如果访问量是配置的数据大小的倍数，则会自动进行数据封装。在通信过程中，只会为数据帧内的位提供时钟并传输这些位。

图 477. 数据大小不等于 8 位、16 位或 32 位时的数据对齐



注意： *数据长度至少为 4 位。如果所选的数据长度不足 4 位，则会将数据帧长度强制为 4 位。*

在具有有限功能集的实例中，数据大小固定为 8 位倍数到最大数据长度（基于 DSIZE[4:3] 位的值，因具体实例而异）。如果对 SPI_TXDR 或 SPI_RXDR 进行更大数据长度（已配置数据大小的倍数）的访问，则会自动应用数据封装。

38.4.10 配置 SPI

主器件和从器件的配置步骤几乎相同。对于具体的模式设置，请遵从相应章节的内容。如果必须对标准通信进行初始化，请执行以下步骤：

1. 对正确的 GPIO 寄存器进行写操作：在 MOSI、MISO、SCK、SS 和 RDY 引脚（如果适用）上配置 GPIO 复用功能。
2. 在使能 SPI 之前，对 SPI_CFG1 和 SPI_CFG2 寄存器进行写操作，以便为所有“非保留”位以及相应的位域设置合适的值，但以下内容除外：
   a) 仅在主模式下会考虑 SSOM、MASRX、SSOE、RDIOM、MBR[2:0]、BPASS、MIDI[3:0]、MSSI[3:0] 位，在主模式下，如果 SSOE 位置 1，则 MSSI[3:0] 位有效，如果 RDIOM 位未置 1，则 RDIOP 位无效。当从器件配置为 TI 模式时，还会考虑 MBR[2:0] 设置。
   b) 仅在从模式下会考虑 UDRCFG。
   c) CRCEN 置 1 时需要配置 CRCSIZE[4:0]。
   d) TI 模式下不需要 CPOL、CPHA、LSBFRST、SSOM、SSOE、SSIOP、SSM、RDIOP、RDIOM、MSSI 和 MIDI。
   e) 一旦 SPI_CFG2 寄存器中的 AFCNTR 位置 1，所有 SPI 输出都会开始传播到相关的 GPIO 引脚上，无论外设是否使能。因此，SPI_CFG1 和 SPI_CFG2 寄存器的任何后续配置更改都会影响这些引脚上的信号电平。

3. 对 SPI_CR2 寄存器进行写操作以选择传输的长度，如果值未知，则必须将 TSIZE 编程为零。
4. 对 SPI_CRCPOLY 进行写操作，并对 SPI2S_CR1 寄存器的 TCRCINI、RCRCINI 和 CRC33_17 位进行写操作，以根据需要配置 CRC 多项式和 CRC 计算。
5. 如果使用 DMA 数据流，请在 DMA 寄存器中配置 SPI Tx 和 Rx 专用的 DMA 数据流（请参见“使用 DMA 进行通信”一章）。
6. 如果需要，可配置 SPI_CR1 寄存器中的 SSI、HDDIR 和 MASRX。
7. 如果需要配置保护，请对 SPI_CFG1 寄存器的 IOLOCK 位进行编程（安全起见）。

38.4.11 使能 SPI

建议在主器件发送时钟前配置和使能 SPI 从器件。但是，如果在总线上正在进行传输时完成配置和启用过程，则不会有任何影响，前提是 SS 信号由硬件在从器件上进行管理，或者对 SS 信号进行软件管理时由从器件一端的软件保持非有效状态（请参见*第 38.4.7 节*）。从器件发送器的数据寄存器应包含待发送的数据，这样主器件才能开始提供时钟。在由 SS 选择 SPI 从器件之前，SCK 信号必须稳定到所选极性所对应的空闲状态电平，否则传输可能会不同步。

当 SPI 从器件在硬件 SS 管理模式下使能时，即使发现 SS 处于有效电平，所有数据传输也会被忽略。数据传输将被忽略，直到从器件检测到 SS 信号的开始（从非有效电平转换为有效电平），使从器件与主器件同步。因此，外部 SS 引脚固定时不能使用硬件管理模式。SS 软件管理不提供此类保护功能。在这种情况下，当总线上无通信时，SSI 位必须更改，在传输之间 SCK 信号处于空闲状态电平。

当 SPI 处于使能状态，CSTART 位置 1 且 TxFIFO 非空时，或者对 TxFIFO 执行下一次写操作时，全双工模式（或任何只发送模式）下的主器件开始通信。

在任何主器件只接收模式下，在 SPI 使能并且 CSTART 位置 1 后，主器件开始通信，时钟开始运行。

有关处理 DMA 的信息，请参见*第 38.4.15 节*。

38.4.12 SPI 数据发送和接收过程

数据通信格式的设置遵循的基本原则是，必须在一个会话（事务）内传输一定数量的大小灵活的数据，同时，数据处理可以有效地累积到 SPI 数据寄存器的单次访问中（数据封装），如果以一致的大数据包收集数据，则甚至可以将其分组为一系列数据处理服务。数据处理服务基于 FIFO 数据包占用事件。因此，必须根据专用数据包标志来专门提供完整的数据包。

为了更好地理解本节后面的复杂内容，用户应首先了解以下各项的配置影响和含义：

数据大小 (DSIZE) – 定义数据帧（设置单个数据帧的位数）。

FIFO 阈值 (FTHLV) – 定义数据包，设置单个数据包的数据帧数量，以及数据包占用事件的发生情况，以通过软件或 DMA 处理 SPI 数据寄存器。

数据访问 – 发生数据包事件时，在应用程序与 SPI FIFO 之间传输数据时，处理 SPI 数据寄存器内容的一种方法。其取决于数据包大小配置。可以选择性地通过单次访问寄存器（通过数据封装）或通过此类访问序列（当提供更大的数据包时）来有效地进行多数据处理。

FIFO 大小 – 容纳可用数据的容量或空间。数据在此空间内的压缩和排列方式取决于数据大小和内部硬件效率。FTHLV 设置必须基于 FIFO 容量，以至少存储两个数据包。

事务大小 (TSIZE) – 定义总体上可能由多个数据包服务覆盖的一个事务会话中所涉及的数据帧总数。无需将此数值与数据包大小对齐（如果 TSIZE 编程正确，则支持处理最后一个未对齐的数据包）。

通过 RxFIFO 和 TxFIFO 进行数据处理

所有 SPI 数据传输均通过按字节组织的嵌入式 FIFO（N x 8 位）。FIFO 的大小 (N) 与产品和外设实例相关。这使得 SPI 能够以连续流工作，并能防止在数据帧长度较短或中断/DMA 延时过长时发生上溢。每个方向都有其自身的 FIFO，分别被称为 TxFIFO 和 RxFIFO。

FIFO 内容的处理基于完全由专用 FIFO 数据包占用标志（TXP、RXP 或 DXP）引发的数据包提供事件。这些标志的出现与否取决于数据交换模式（双工、单工）、数据帧大小（帧中的位数）以及数据在数据包中的组织排列方式。当以定义 FIFO 阈值的方式将数据编排到数据包中时，数据包事件的频率会显著降低。这样，就可以通过单次 SPI 数据寄存器访问或访问序列，基于单个 FIFO 占用数据包事件，有效地处理按数据包分组的多个数据帧，从而消耗更少的系统性能。用户可以通过转换数据寄存器地址来控制访问类型，以强制应用具体的 CPU 指令来读取或写入寄存器。访问可以是 8 位、16 位或 32 位，但必须至少访问单数据帧。务必使数据包大小 (FTHVL) 和数据大小 (DSIZE) 的设置与所应用的数据寄存器访问（无论是否应用单次访问或访问序列）始终保持平衡，以便在发生相关事件时应用和完成单个数据包服务。原则上，无论是否应用 DMA、中断或轮询，FIFO 占用标志的出现和清除能力都是通用的。

对 SPI_RXDR 寄存器执行读访问时，会返回尚未读取的 RxFIFO 中存储的最早的值。对 SPI_TXDR 执行写访问时，会在发送队列结束时将写入的数据存储到 TxFIFO 中。

SPI_RXDR 寄存器的读访问必须通过 RXP 事件进行管理。接收处于激活状态时，如果接收 FIFO 中至少有一个完整数据包（通过 SPI_CFG1 寄存器的 FTHVL[3:0] 位定义为接收器阈值）可用，则该标志由硬件置 1。通过软件或 DMA 读取 SPI_RXDR 时，只要 RXFIFO 中的可用数据少于完整的单个数据包，就会清零 RXP。

RXP 会在 RXPIE 位置 1 时触发中断，或在 RXDMAEN 位置 1 时触发 DMA 请求。

RXP 标志置 1 时，应用程序会执行适当数量的 SPI 数据寄存器读取操作，以下载一个数据包的内容。下载完整数据包后，应用软件或 DMA 会检查 RXP 值，以查看接收 FIFO 中是否存在待决的其他数据包，如果存在，则逐个数据包对其进行下载，直到 RXP 读为 0。RxFIFO 最多可存储 N 个数据帧（帧大小 ≤ 8 位时）、N/2 个数据帧（8 位 < 帧大小 ≤ 16 位时）、N/3 个数据帧（16 位 < 帧大小 ≤ 24 位时）或 N/4 个数据帧（数据帧大小 > 24 位时），其中 N 表示 FIFO 的大小（以字节计）。

接收结束时，可能会发生 RxFIFO 中仍然有某些数据可用，而不会达到 FTHVL 级别的情况，因此 RXP 不会置 1。在这种情况下，FIFO 中的剩余 RX 数据帧数将由 SPI_SR 寄存器中的 RXWNE 和 RXPLVL 位域指示。传输大小和数据包大小未对齐时，如果在传输中接收到的最后数据量不能完全达到配置的数据包大小，则会发生上述情况。不过，应用软件仍可从 RxFIFO 中进行用于先前完整数据包的标准次数的读取操作，而不会产生不利影响：只会从 RxFIFO 中弹出一致数据（完整的数据帧），而冗余读取次数（或任何不完整的数据）将为读数 0。通过这样的方式，应用软件可以使用相同的方式处理传输中的所有数据，并且可减轻负担来预知传输中的最后一个数据的接收，可减轻对要从 RxFIFO 中弹出的数据的适当读取次数进行计算的负担。

同样地，待发送数据帧的写访问通过 TXP 事件进行管理。发送处于激活状态时，如果应用程序有足够的空间来将至少一个完整的数据包（通过 SPI_CFG1 寄存器的 FTHVL[3:0] 位进行定义）推入到发送 FIFO 中，则该标志由硬件置 1。TxFIFO 由软件和/或 DMA 进行填充后，TXP 会被清零。当前可用于下一完整数据包的空间将丢失。逐帧存储新数据包时，如果从 TxFIFO 中释放数据，则这会导致 TXP 信号的振荡。TXTF 置 1 或禁止 SPI 时，如果没有足够的空间来存储至少一个数据帧（未遵循 TXP 事件），则对 TxFIFO 进行的任何写操作都将被忽略。

TXP 会在 TXPIE 位置 1 时触发中断，或在 TXDMAEN 位置 1 时触发 DMA 请求。TXPIE 屏蔽在 TXTF 标志置 1 时由硬件清零。

TXP 标志置 1 时，应用会执行适当数量的 SPI 数据寄存器写入操作，以上传一整个数据包的内容。上传完新的完整数据包后，应用软件或 DMA 会检查 TXP 值，以查看是否可将其他数据包推入 TxFIFO 中，如果可以，则逐个数据包对其进行上传，直到 TXP 读为 0。

传输大小和数据包大小未对齐时，传输中的最后一个数据量可短于配置的数据包大小。不过，应用软件仍可进行用于先前数据包的标准次数的数据寄存器写入操作，而不会产生不利影响：只会将一致数据推入 TxFIFO 中，而冗余写入次数将被忽略。通过这样的方式，应用软件可以使用相同的方式处理传输中的所有数据，并且可减轻负担来预知传输中的最后一个数据的发送，可减轻对要推入 RxFIFO 中的最后一个数据的适当写入次数进行计算的负担。仅在最后一个数据情况下，如果 TxFIFO 中有足够的空间来存储剩余数据进而完成当前传输，TXP 事件将由 SPI 置为有效。

TXP 和 RXP 事件均可通过中断进行轮询或处理。在全双工模式下，DXP 位可作为通用 TXP 和 RXP 事件进行监视。

将 DXP 标志置 1 时，应用软件会向 SPI 数据寄存器执行适当数量的写入操作，来上传一个用于发送的完整数据包的内容，然后从 SPI 数据寄存器进行相同数量的读取操作来下载一个数据包的内容。上传并下载一个数据包后，应用软件或 DMA 会检查 DXP 值，以查看是否可按顺序推入和弹出其他数据包，如果可以，则逐个数据包对其进行上传/下载，直到 DXP 读为 0。

DXPIE 位置 1 时 DXP 会触发中断。DXPIE 屏蔽在 TXTF 标志置 1 时由硬件清零。

DXP 对于全双工通信十分有用，可优化数据上传/下载性能，并减少支持 SPI 传输所需的中断数或 DMA 序列，从而最大限度地减少对 CPU 带宽和系统电源的需求，特别是在 SPI 工作在停止模式时，尤为如此。

只中继 DXP 中断时，如果 TXP 和 RXP 事件由通用程序处理，则用户必须考虑这种简化的缺点，因为在这种情况下 TXP 服务会被故意延迟。这是因为 TXP 事件通常发生在接收 RXP 事件之前，以允许 TXP 处理最后帧的上一事务，将 TxFIFO 完全清空，否则主器件无法提供连续的 SCK 时钟流，从器件甚至会出现下溢。可能的解决方法是，在会话开始之前预先填充几个数据包，并在事务结束时通过 EOT 专门处理在 TXTF 事件后接收到的所有数据（因为 TXTF 会在事务结束时抑制 DXP 中断）。如果使能 CRC 计算，则在事务结束时，若仅依靠 EOT，用户必须计算出额外的空间，以容纳 RxFIFO 中的 CRC 帧。

另一种管理数据交换的方式是使用 DMA（请参见*第 38.4.15 节*）。

如果在 RxFIFO 已满时收到下一个数据，将发生上溢事件（请参见*第 38.5.2 节*中的 OVR 标志说明）。上溢事件可通过中断来轮询或处理。

当 MASRX = 0 时，在从模式下或主接收模式下会发生这种情况。如果 MASRX 位在主接收器上置 1，则当 RxFIFO 填满时，生成的时钟会自动停止，从而防止发生上溢。

禁止 SPI (SPE = 0) 时，RxFIFO 和 TxFIFO 内容保持清空状态，无法访问。

事务处理

可通过一个序列传送一些数据帧从而完成一条消息。用户可通过存储在 TSIZE 中的值，处理一条消息中的多个数据。原则上，消息的传输在通过将 CSTART 位置 1 来使能 SPI 时开始，并在处理完所需数据总量后结束。如适用，事务的结束可控制 CRC 和硬件 SS 管理。强烈建议在下一个事务开始之前禁止然后重新使能 SPI（尽管设置未更改），以正确重启内部状态机。

如果在 CSTART 置 1 时 TSIZE 保持为零，则会初始化无限事务（不控制传输大小）。在无限事务期间，仅支持与 FIFO 阈值相符的已处理数据量。如果数据量（或其分组数据包数量）不可预测，则用户必须将 FIFO 阈值设置（数据包大小）保持为单个数据 (FTHLV = 0)，以确保每个数据帧引发其自己的数据包事件并由应用程序或 DMA 进行处理。凭借设置

可清零 CSTART 位的 CSUSP 位，可随时挂起事务。在此类软件挂起后必须始终禁止 SPI，然后在下一个事务开始前重新使能。

使能发送后，序列即开始，只要主器件的 TxFIFO 中存在数据便一直继续。时钟信号由主器件永久性提供，直至 TxFIFO 变为空，之后时钟信号停止，等待其他数据。

在只接收模式、半双工（COMM[1:0] = 11，HDDIR = 0）或单工 (COMM[1:0] = 10) 模式下，主器件在 SPI 使能时启动序列并通过将 CSTART 位置 1 来启动传输。时钟信号由主器件提供，且直至主器件禁止/挂起 SPI 或仅接收模式才会停止。在此之前，主器件会永久性接收数据帧。可通过软件控制以及通过向 SPI_CR1 寄存器的 CSUSP 位写入 1 将接收挂起，或者可在 MASRX = 1 且 RxFIFO 已满时自动将接收挂起，还可以在 RDY 状态下（如果应用该信号）将接收挂起（请参见*第 38.4.8 节*）。完成 TSIZE 中编程的帧数时，接收也会自动停止。

为禁止主器件只接收模式，必须先挂起 SPI。SPI 被挂起时，会在更改配置前完成当前帧。

小心： 如果在主模式下将 SPE 位清零，而接收正在进行且不会挂起，则时钟将停止，且不会完成当前帧，并会清空 RxFIFO。

当主器件能够以连续模式（SCK 信号连续）提供所有交互时，任何时候都必须根据从器件功能来处理数据流及其内容。如果从器件提供 RDY 信号选项，则主器件可以监视从器件发出的 RDY 信号，以控制通信流。如果未使用 RDY 引脚，则会将从器件视为始终准备就绪可以与主器件进行通信。

必要时，主器件必须降低通信速度，提供较慢的时钟或带有足够延时的单独帧或数据会话（通过将 MIDI[3:0] 位置 1），或者，提供初始延时（通过将 MSSI[1:0] 置 1），这会延迟传输的开始，从而为从器件提供充分的空间来准备数据。请注意，来自从器件的数据始终由主器件传输和处理，即使从器件无法及时正确地准备数据也是如此。从器件最好使用 DMA，尤其是数据帧较短，按字节访问 FIFO 且 SPI 总线速率较高时。

为了对来自从器件发送器节点的 SPI 通信流添加软件控制功能，可使用写入到 SPI_UDRDR（SPI 下溢数据寄存器）中的特定值。在从器件侧，当 TxFIFO 变为空时，该值将作为下一个数据自动送出，并且在主器件接收器侧可通过软件进行解析（丢弃或解析为类似 XOFF 的命令，以便通过软件挂起主器件接收器）。

在多从星形拓扑中，在指定时间只能使能一个从器件用于输出数据。选定与主器件通信的从器件需要在与主器件开始通信之前检测其 SS 输入是否变为有效电平。在单个从器件系统中，无需通过 SS 来控制从器件，但此时提供此脉冲通常会更好，以在每个数据序列开始时同步从器件。SS 可通过软件和硬件进行管理（请参见*第 38.4.7 节*）。

38.4.13 关闭 SPI

要禁止 SPI，必须按照本段中介绍的禁止步骤进行操作。

在主模式下，在外设时钟停止，系统进入低功耗模式之前执行该操作十分重要，否则会损坏进行中的事务。

在从模式下，SPI 通信可在 **spi_pclk** 和 **spi_ker_ck** 时钟停止时继续不间断进行，直至达到结束通信或数据服务请求条件。通常，可通过将系统置于停止模式来停止 **spi_pclk**。更多相关信息，请参见 RCC 部分。

当处于全双工或仅发送模式下的主器件停止提供待发送的数据时，可结束任何事务。在这种情况下，时钟在最后一个数据传输后停止。可轮询 TXC 标志（或通过 EOTIE = 1 使能中断），以等待发送最后一个数据帧。

为停止外设，当主器件处于仅接收模式下时，必须首先通过将 CSUSP 置 1 挂起 SPI 通信。

当 SPI 被挂起时，已接收但未读取的数据始终存储在 RxFIFO 中。

在软件挂起之后，必须始终禁止 SPI，才能正确重新启动内部状态机。

禁止 SPI 时，RxFIFO 会被清空。为了防止丢失未读取的数据，用户必须通过读取所有剩余的数据（如 SPI_SR 寄存器的 RXP、RXWNE 和 RXPLVL 位域所示），来确保在禁止 SPI 时 RxFIFO 为空。

标准禁止步骤通过轮询 EOT 和/或 TXC 状态来检查发送会话是否（完全）结束。还可以在必须识别正在处理的传输是否结束的特定情况下完成这种检查，例如：

- 当主器件通过与 SPI 无关的 GPIO 处理 SS 信号（例如在多从机星形拓扑的情况下），且必须为从器件提供正确的 SS 脉冲结束时，或者
- 最后一个数据帧或 CRC 帧传输仍在外设总线中处理时，来自 DMA 或 FIFO 的传输数据流完成。

当 TSIZE>0 时，EOT 和 TXC 信号相等，因此无论采用何种 SPI 通信模式，EOT 轮询均可靠，以检查总线活动的结束。当 TSIZE = 0 时，用户必须根据所应用的 SPI 模式和数据流终止方式，来检查 TXC、SUSP 或 FIFO 占用标志。

主模式下的正确禁止步骤如下（使用仅接收模式时除外）：

1. 等待 TXC = 1 和/或 EOT = 1（不再有待发送的数据和发送最后一个数据帧）。使用 CRC 时，会在块中的最后一个数据被处理后自动发送。在这种情况下，TXC/EOT 会在 CRC 帧完成时置 1。当发送挂起时，软件必须等到 CSTART 位清零。
2. 读取所有 RxFIFO 数据（直到 RXWNE = 0 且 RXPLVL = 00）
3. 禁止 SPI (SPE = 0)。

主器件仅接收模式的正确禁止步骤如下：

1. 等待 EOT 或通过挂起 SPI (CSUSP = 1) 中断接收流
2. 如果接收流已挂起，等待至 SUSP = 1（最后一个数据帧已处理完）。
3. 读取所有 RxFIFO 数据（直到 RXWNE = 0 和 RXPLVL = 00）
4. 禁止 SPI (SPE = 0)。

在从模式下，禁止 SPI 时，任何正在进行的数据都将丢失。

控制 I/O

SPI 禁止后，使能的相关 AF 输出仍可由器件驱动，具体取决于 AFCNTR 设置。如果应用了有效输出控制 (AFCNTR = 1) 并且 SPI 刚被禁止 (SPE = 0)，则与 SPI 控制信号相关的使能输出（例如主器件上的 SS 和 SCK 以及从器件上的 RDY）可以立即切换到无效电平（分别根据主器件的 SSIOP 和 CPOL 设置以及从器件的 RDIOP 设置）。而数据线输出（主器件的 MOSI 和从器件的 MISO）可以根据实际 TxFIFO 内容立即更改其电平，从而可能导致总线上最后一个处理位的值无效并且得不到保证。如有必要，用户须注意数据线上正确的数据保留时间，避免在最后一个数据事务刚完成就快速将 SPI 禁止。

注意： *尽管在设计上保证了时钟沿采样期间最后位的稳定性，但对于某些器件，可能需要在采样期间延长该数据位保持稳定的时间间隔。例如，可以通过在 EOT 事件发生与禁止 SPI 的操作之间插入短暂的软件延迟来完成。*

38.4.14 数据封装

从用户的角度来看，数据封装方式有两种，这两种方式可彼此覆盖：

- 向 TxFIFO 中写入数据或从 RxFIFO 中读取数据时的访问类型。
  如果数据大小成倍地小于对 SPI_TXDR 或 SPI_RXDR 寄存器执行的访问大小，则可通过单次访问有效地推入或提取多个数据。
- 在单一软件服务期间要处理的数据量
  可方便地将数据分组到数据包中，并将所有数据包内容累积到 FIFO 服务，而不是分别逐帧处理数据。用户可通过 FIFO 阈值设置定义数据包。之后，所有 FIFO 占用事件均与该阈值相关，而所需服务通过具有中断和/或唤醒功能的适当标志来反映。

若数据帧大小不足一个字节（小于或等于 8 位），则在 SPI_RXDR/SPI_TXDR 寄存器上执行任何 16 位或 32 位读写访问时都将自动使用数据封装。在这种情况下将并行处理多数据帧类型。最初，SPI 以所访问字的 LSB 中存储的模式工作，然后以 MSB 中存储的另一种模式来工作。

图 478 提供了全面功能集实例中数据封装模式序列处理的示例。此时 DSIZE[4:0] 被配置为 4 位，在对发送器的 SPI_TXDR 寄存器进行单次 16 位或 32 位访问后，会向 TxFIFO 中写入两个或四个数据帧。如果数据帧大小介于 9 位和 16 位之间，则在完成一次 32 位访问后，会自动使用数据封装。将首先使用最低有效半字。（与 LSBFRST 值无关）

如果将 RxFIFO 阈值设为 1 个帧（且以帧为单位读取数据，即解封形式），则该序列可在接收器中生成两个或四个 RXP 事件，或者，如果 SPI_CFG1 寄存器中的 FTHLV[3:0] 位域被配置为以封装模式读取的多个帧（16 位或 32 位读访问），则可生成一个 RXP 事件。

数据根据*图 477* 进行对齐。有效位仅在总线上执行。未使用的位在发送器侧无关紧要，而在接收器侧则用零填充。

将短数据帧（<8 位或 < 16 位）与大型数据访问模式（16 位或 32 位）搭配使用时，FTHVL 值必须被配置为帧/数据访问数的倍数（如果将 32 位访问用于最高 8 位的帧则为 4 的倍数，或者，如果将 16 位访问用于最高为 8 位的帧或者将 32 位访问用于最高 16 位的帧，则为 2 的倍数）。

RxFIFO 阈值设置必须始终高于或至少等于后续读访问大小，否则将读取额外的伪数据。

不允许小于配置的数据大小的 FIFO 数据访问。始终须确保至少访问一个完整的数据帧。

如果 FIFO 中存在不完整的数据包，则会出现特定问题：小于在 FTHLV 位设置的阈值。

解决该问题的方法有两种：

A. 不使用 TSIZE 位域

在发送器侧，只需将任意奇序列的最后一个数据帧以 8 位/16 位访问的方式写入 SPI_TXDR 即可。

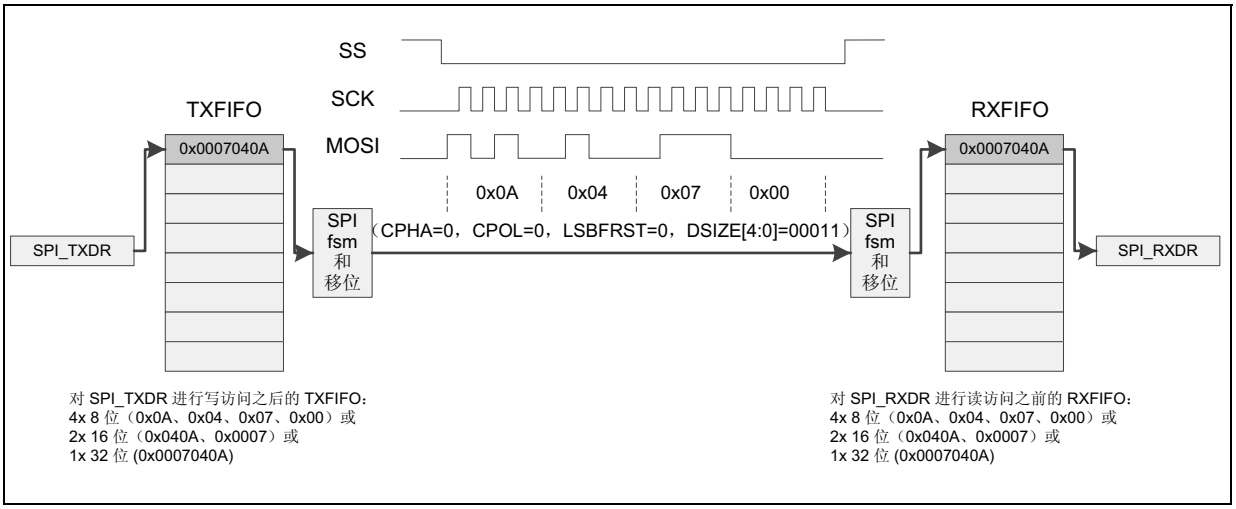
在接收器侧，可通过任意访问读取剩余数据。任何额外的数据读取都会用零填充。轮询 RXWNE 和 RXPLVL 可用于检测 RX 数据何时在 RxFIFO 中可用。（可以在系统级利用超时来检测轮询）

B. 使用 TSIZE 位域

在发送器侧，事务在遇到 EOT 事件时由主器件停止。

在接收过程中，RXP 标志不会在 EOT 置 1 时置 1。要接收的数据数 (TSIZE) 不是数据包大小的倍数时，剩余数据的数目通过 SPI_SR 寄存器的 RXWNE 和 RXPLVL 位域指示。可通过任意访问读取剩余数据。任何额外的读取都会用零填充。

图 478. 将数据封装在 FIFO 中，以便以全面功能集实例进行发送和接收



1. DSIZE[4:0] 配置为 4 位，数据右对齐，有效位仅在总线上执行，其顺序取决于 LSBFRST，如果置 1，所有数据帧的顺序都将反向。

38.4.15 使用 DMA（直接存储器寻址）进行通信

为了以最大速度工作并且为了促进避免上溢所需的数据寄存器读/写过程，SPI 提供了 DMA 功能，该功能采用了简单的请求/应答协议。

将 SPI_CFG1 寄存器的使能位 TXDMAEN 或 RXDMAEN 置 1 时，将请求 DMA 访问。必须向发送缓冲区和接收缓冲区发出单独的请求，以完成所定义数据包的服务。

- 在发送过程中，每次 TXP 置 1 都会触发一系列 DMA 请求。然后，DMA 将对 SPI_TXDR 寄存器执行一系列写操作。
- 在接收过程中，每次 RXP 置 1 都会触发一系列 DMA 请求。然后，DMA 将对 SPI_RXDR 寄存器执行一系列读操作。如果 EOT 在事务结束时置 1，且最后一个数据包不完整，则会根据 RXWNE 和 RXPLVL[1:0] 设置自动激活 DMA 请求来读取剩余数据。

如果将 SPI 编程为仅接收模式，则 UDR 将永不置 1。

如果将 SPI 编程为发送模式，如果发送数据未准备好，则 TXP 和 UDR 最终可在从器件侧置 1。在这种情况下，将根据 UDR 管理选择在 TX 线上发送一些数据。

在单工模式下使用 SPI 时，用户必须仅使能足够的 DMA 通道，同时禁止未使用的互补通道。

如果将 SPI 编程为仅发送模式，则 RXP 和 OVR 将永不置 1。

如果将 SPI 编程为全双工模式，若接收数据未读取，则 RXP 和 OVR 最终会置 1。

在发送模式下，DMA 或用户写入所有要发送的数据（SPI_SR 寄存器中的 TXTF 标志置 1）后，可以对 EOT（或 TXC（TSIZE = 0 时））标志进行监视，以确保 SPI 通信已完成。在禁止 SPI 之前或在主模式下禁止 **spi_pclk** 之前必须执行此步骤，以避免损坏最后一次发送。软件必须首先等待至 EOT = 1 和/或 TXC = 1。

通过 DMA 开始通信时，为防止 DMA 通道管理引发错误事件，必须按顺序执行以下步骤：

1. 如果使用 DMA Rx，通过 SPI_CFG1 寄存器中的 RXDMAEN 位来使能 DMA 接收缓冲区。
2. 如果使用 DMA，则通过 DMA 寄存器来使能 Tx 和 Rx 的 DMA 请求。
3. 如果使用 DMA Tx，通过 SPI_CFG1 寄存器中的 TXDMAEN 位来使能 DMA 发送缓冲区。
4. 通过将 SPE 位置 1 使能 SPI。

要关闭通信，必须按顺序执行以下步骤：

1. 如果使用 DMA，则通过 DMA 寄存器来禁止 Tx 和 Rx 的 DMA 请求。
2. 通过后续 SPI 禁止步骤来禁止 SPI。
3. 如果使用 DMA Tx 和/或 DMA Rx，通过将 SPI_CFG1 寄存器中的 TXDMAEN 和 RXDMAEN 位清零来禁止 DMA 发送缓冲区和接收缓冲区。

使用 DMA 时的数据封装

如果由 DMA 管理传输（SPI_CFG1 寄存器中的 TXDMAEN 和 RXDMAEN 位置 1），则将根据为 SPI TX 和 SPI RX 的 DMA 通道配置的 PSIZE 值自动使能/禁止封装模式。

如果 DMA 通道的 PSIZE 值等于 16 位，且 SPI 数据大小小于或等于 8 位，则将使能封装模式。同理，如果 DMA 通道的 PSIZE 值等于 32 位，且 SPI 数据大小小于或等于 16 位，则将使能封装模式。然后，DMA 将自动管理对 SPI_TXDR 寄存器的写操作。

无论使用何种数据封装模式，也无论要传输的数据量是否为 DMA 数据大小（16 位或 32 位）的倍数，如果帧大小较小，则 DMA 会根据 TSIZE 位域设置自动完成传输。

此外，最后几个数据帧还可由软件以单一/解封模式写入。

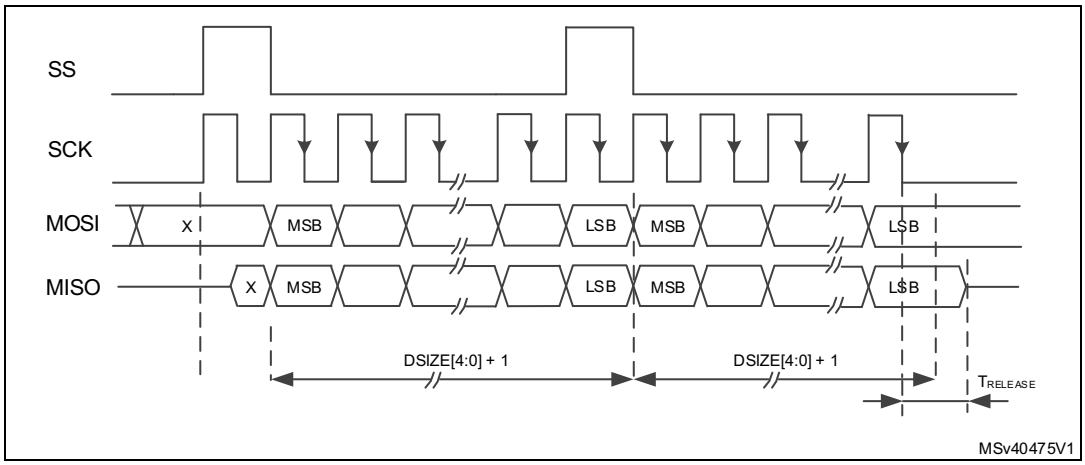
要配置 DMA，禁止小于配置的数据大小的 DMA 数据访问。始终须确保至少访问一个完整的数据帧。

38.5 SPI 特定模式和控制

38.5.1 TI 模式

通过 SPI_CFG2 寄存器的特定 SP[2:0] 位域设置，可以将 SPI 配置为符合 TI 协议。SCK 和 SS 信号极性、相位和流以及位顺序均固定，因此在 SPI 处于 TI 模式配置下时，无需 CPOL、CPHA、LSBFRST、SSOM、SSOE、SSIOP、SSM、RDIOP、RDIOM、MSSI 和 MIDI 的设置。SS 信号通过 LSB 数据位的脉冲同步协议，如*图 479* 所示。

图 479. TI 模式传输



在从模式下，时钟发生器用于定义当前事务完成时，MISO 引脚的从器件输出变为高阻态的时间。主波特率设置（SPI_CFG1 中的 MBR[2:0]）适用，并且可以使用任意波特率，可以非常灵活地确定此时刻。MISO 信号变为 HiZ 的延迟 (TRELEASE) 也取决于内部重新同步，这需要另外 2-4 个时钟信号周期来馈送到发生器。具体公式如下：

$$\frac{Tbaud}{2} + 2 \times Tspi_ker_ck \leq Trelease \leq \frac{Tbaud}{2} + 4 \times Tspi_ker_ck$$

如果从器件在进行数据事务时检测到错位的 SS 脉冲，TIFRE 标志将置 1。

38.5.2 SPI 错误标志

如果以下其中一个错误标志置 1 且已通过将相应的中断使能位置 1 使能了中断，则将生成 SPI 中断。

上溢标志 (OVR)

当主器件或从器件接收了数据但 RxFIFO 没有足够的空间来存储接收的数据时，将出现上溢的情况。如果软件或 DMA 没有足够的时间来读取之前接收到的数据（存储在 RxFIFO 中），则可能会发生这种情况。

当出现上溢的情况时，OVR 标志置 1，新接收的值不会覆盖 RxFIFO 中之前的值。新接收的值将被丢弃，之后发送的所有数据都将丢失。OVRIE 位置 1 时 OVR 标志会触发中断。向 SPI_IFCR 的 OVRC 位写入 1 可以将 OVR 位清零。为了防止接下来发生溢出事件，应在通过软件读取清空 RxFIFO 后进行清零。建议尽可能释放 RxFIFO 空间，这意味着根据 RXP 标志的指示读出所有可用数据包。

在主模式下，用户可通过自动通信挂起（MASRX 位）来防止出现 RxFIFO 上溢。

下溢标志 (UDR)

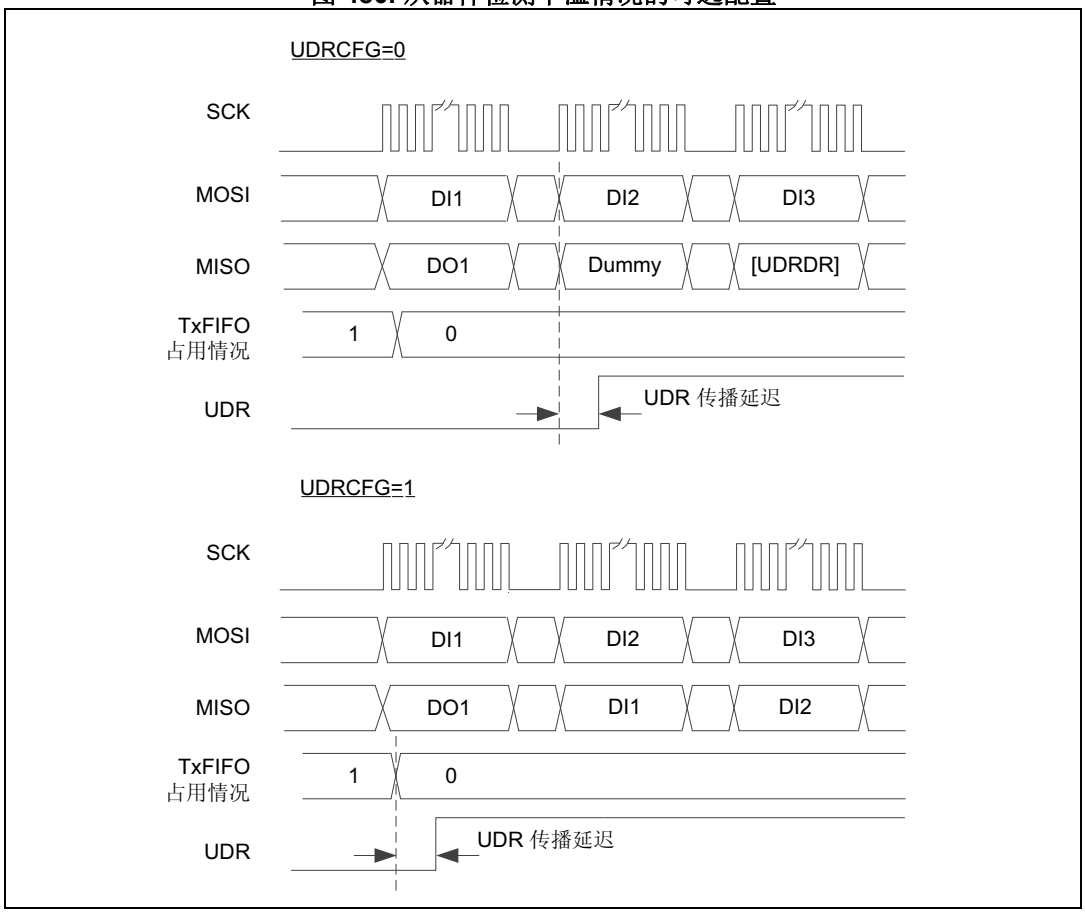
在从器件发送模式下，如果通常情况下从器件 TxFIFO 中没有数据可用于发送，则会由硬件在内部捕获下溢情况。UDR 标志设置随后通过硬件传播到状态寄存器（请参见下面的注）。UDRIE 位置 1 时 UDR 会触发中断。

下溢检测逻辑和系统行为取决于 UDRCFG 位。当从器件检测到下溢时，它可以提供用户在 UDRDR 寄存器中存储的恒定模式，或提供之前从主器件接收的数据。当应用第一个配置 (UDRCFG = 0) 时，只要 TxFIFO 为空时主器件开始传送新数据帧，就会对下溢状况进行评估。然后，始终会在最后一个有效数据与 UDRDR 寄存器中定义的常量模式之间插入一个附加的空（意外）数据（请参见 *图 480*）。第二种配置 (UDRCFG=1) 可用于环形拓扑结构（请参见 *图 471*）。假设主器件开始通信时 TxFIFO 不为空，则一旦 FIFO 在下一个数据流期间变空，就会对下溢情况进行评估。然后，TxFIFO 中的有效数据立即会被最后接收的数据顶替。

软件将 UDR 标志清零后会重新使能标准发送，此清零操作会由硬件传播到 SPI 逻辑。用户应向 TxFIFO 中写入一些数据，然后再清零 UDR 标志，以防捕获到下一个下溢情况。

从器件处理的数据不可预测，尤其是在 TxFIFO 为空，且尚未捕获或刚清除完下溢条件时，开始或继续事务的情况下，更为如此。通常，在刚刚使能 SPI 时，或者在刚刚开始已规定大小的事务时，会出现这种情况。在这种情况下，前几个位可能会损坏，此外，在数据事务开始之前过于紧密地向空 TxFIFO 中写入第一个数据（到 TxFIFO 的数据传播会占用几个 APB 时钟周期）时，也会出现这种情况。

图 480. 从器件检测下溢情况的可选配置



注意： *要对 UDR 事件进行硬件传播，总线上需要有额外流量。传播始终需要事件发生后的几个额外的 SPI 时钟周期（硬件捕获下溢并由软件清零）。如果在数据帧事务接近结束时通过软件将 UDR 标志清零，或者 SCK 线在帧之间处于空闲状态，则在再次处理来自 TxFIFO 的有效数据之前，会先由从器件在内部发送下一个额外下溢模式。用户可以通过在会话之间禁止/使能 SPI 来防止出现这种情况，以重新启动下溢逻辑，从而通过有效数据启动下一个会话。*

模式故障 (MODF)

当主器件的内部 SS 信号（SS 硬件模式下为 SS 引脚，SS 软件模式下为 SSI 位）被拉低时，将发生模式故障。这将以以下方式自动影响 SPI 接口：

- 如果 MODFIE 位置 1，MODF 位将置 1，并触发 SPI 中断。
- SPE 位强制为零，直到 MODF 位置 1。这会禁止 SPI，并阻止除 MODF 中断请求（如果使能）之外的所有外设输出。
- MASTER 位清零，从而强制器件进入从模式。

向 SPI_IFCR 的 MODFC 位写入 1 可以将 MODF 清零。

为避免在包含多个 MCU 的系统中发生多从冲突，在重新使能 SPI 之前，必须将 SPE 位置 1，以将 SS 引脚拉至无效电平。

为安全起见，硬件不允许在 MODF 位置 1 时将 SPE 位置 1。在从器件中，MODF 位不可置 1，但由前一次多主模式冲突引起时除外。

在多主系统中，当主器件接管总线时，正确的软件程序应如下所示：

- 当 SSOE = 0 时切换到主模式（如果另一个主器件占用总线，可能会发生冲突。在这种情况下会引发 MODF，从而阻止任何下一个节点切换到主模式）
- 将专用于另一个主器件 SS 控制的 GPIO 引脚置于有效电平
- 执行数据事务
- 将专用于另一个主器件 SS 控制的 GPIO 引脚置于非有效电平
- 切换回从模式

CRC 错误 (CRCE)

当 SPI_CFG1 寄存器中的 CRCEN 位置 1 时，此标志用于验证接收数据的有效性。在接收了最后一个数据后（由 TSIZE 定义），如果移位寄存器中接收的值与接收器 SPI_RXCRC 的值不匹配，则 SPI_SR 寄存器中的 CRCE 标志将置 1。CRCEIE 位置 1 时 CRCE 标志会触发中断。向 SPI_IFCR 的 CRCEC 位写入 1 可以将 CRCE 位清零。

TI 模式帧格式错误 (TIFRE)

如果 SPI 在从模式下工作，并配置为符合 TI 模式协议，则在通信进行期间出现 SS 脉冲时，将检测到 TI 模式帧格式错误。出现此错误时，SPI_SR 寄存器中的 TIFRE 标志将置 1。发生错误时不会禁止 SPI，但会忽略 SS 脉冲，并且 SPI 会等待下一个 SS 脉冲，然后再开始新的传输。由于错误检测可能导致丢失几个数据帧，因此数据可能会损坏。

向 SPI_IFCR 的 TIFREC 位写入 1 可以将 TIFRE 标志清零。如果 TIFREIE 位置 1，则检测到 SS 错误时将生成中断。由于无法再保证数据一致性，因此应由软件重新启动主从器件之间的通信。

38.5.3 CRC 计算

为检查发送数据和接收数据的可靠性，使用两个独立的 33 位或 17 位 CRC 计算器。对于完整功能集实例，最大数据大小为 32 位的情况，SPI 可提供 5 到 33 位的 CRC 多项式长度；对于最大数据大小被限制为 16 位的外设实例，则可提供 9 到 17 位的多项式长度。对于有限功能集实例，仅当数据大小限制为 16 位时，CRC 多项式长度才可以设置为 9 或 17 位，当数据大小扩展到 32 位时，可以选择设置为 33 位。多项式的长度由存储在 CRCPOLY 寄存器中的值的最高有效位定义。必须将其设置为大于 DSIZE 位域中定义的数据帧长度的值。应用最大数据大小时，必须额外将 CRC33_17 位置 1，以定义多项式字符串的最高有效位，同时保持其大小始终大于数据大小。SPI_CFG1 中的 CRCSIZE 位域随后会定义 CRC 计算寄存器中被处理并与 CRC 帧进行比较的最高有效位数。其定义与数据帧长度无关，但必须等于数据帧大小或是数据帧大小的整数倍，且不能超过实例的最大数据大小。

为了充分获得 CRC 计算功能的好处，多项式长度设置必须与 CRC 模式大小相对应，否则会对计算中未使用的位进行处理，并且如果其大小设置为大于多项式长度，则 CRC 模式末尾应全部为零。

CRC 原理

在使能 SPI (SPE = 1) 前，通过将 SPI_CFG1 寄存器中的 CRCEN 位置 1 来使能 CRC 计算。然后使用由 CRCPOLY 寄存器和 CRC33_17 位定义的 CRC 多项式计算 CRC 值。如果使能了 SPI，则可以更改 CRC 多项式，但只有在总线上无通信时才可进行更改。

按照由 SPI_CR1 寄存器的 CPHA 位和 CPOL 位定义的采样时钟边沿，逐位进行 CRC 计算。会在 SPI_CR2 寄存器专门定义的数据块末尾，自动检查计算得到的 CRC 值。

如果在接收到的数据内部计算得出的 CRC 与从器件发送器接收到的 CRC 之间检测到不匹配，则 CRCE 标志将置 1，以指示存在数据损坏错误。CRC 的正确处理步骤取决于 SPI 配置和所选的传输管理。

CRC 传输管理

通信开始后将一直持续到必须发送完或接收完 SPI_DR 寄存器中的最后一个数据帧。

传输的长度必须由 TSIZE 定义。处理了所需数量的数据后，将发送 TXCRC，并将线路上接收到的数据与 RXCRC 值进行比较。

无论 CRCSIZE 的配置如何，如果使能了 CRC，则 TSIZE 不能在完整功能集实例中设置为 0xFFFF，也不能在有限功能集实例中设置为 0x3FF 值。

在发送过程中，CRC 计算在 CRC 事务期间被冻结，TXCRC 将以帧的形式进行发送，帧的长度等于 CRCSIZE 位域的值。

在接收过程中，处理完所需的数据量后，RXCRC 也会被冻结。随后以帧的形式接收要与 RXCRC 寄存器内容进行比较的信息，帧长等于 CRCSIZE 值。

完成 CRC 帧后，将执行自动校验，将接收的 CRC 值与 SPI_RXCRC 寄存器中计算的值进行比较。软件必须校验 SPI_SR 寄存器中的 CRCE 标志，以确定数据传输是否损坏。软件通过向 CRCEC 写入 1 将 CRCE 标志清零。

用户无需考虑刷新冗余 CRC 信息，该过程将自动完成。

复位 SPI_TXCRC 和 SPI_RXCRC 值

在 CRC 阶段后对新数据进行采样时，SPI_TXCRC 和 SPI_RXCRC 值将自动初始化。借此，可使用 DMA 循环模式来不间断地传输数据（中间 CRC 校验阶段会涉及一些数据块）。接收器和发送器的初始化模式可配置为零或全 1，具体取决于对 SPI_CR1 寄存器的位 TCRCINI 和 RCRCINI 进行的设置。

禁止 SPI 时，CRC 值复位。

38.6 SPI 低功耗模式

表 308. 低功耗模式对 SPI 的影响

模式	说明
睡眠	无影响。SPI 中断可使器件退出睡眠模式。
停止(1)	SPI 寄存器内容保留。
待机(1)	该模式期间断电时，SPI 实例不起作用，必须在退出待机模式后重新初始化。

1. 有关各实例从停止模式唤醒支持以及待机模式可用性的信息，请参见*第 38.3 节：SPI 实现*。如果某个实例在停止模式下不起作用，则必须在进入此停止模式之前将其禁用。

38.7 SPI 中断

表 309 概述了能够生成中断的 SPI 事件（如果使能）。其中的某些事件还具有低功耗模式唤醒功能。大多数事件均可在使用特定中断使能控制位时单独使能和禁止。与事件相关的标志通过特定方法清零。有关事件标志的更多详细信息，请参见 SPI 寄存器的说明。SPI 被禁止时，除 MODF 中断请求外，所有挂起的中断请求都将被阻止，以防止它们传播到中断服务。

表 309. SPI 唤醒和中断请求

中断向量	中断事件	事件标志	使能控制位	事件清除方法	从停止和待机模式退出(1)(2)
SPI	TxFIFO 准备就绪，可装载数据（存在可供一个数据包使用的空间——FIFO 阈值）	TXP	TXPIE	TxFIFO 包含的空位置数少于 FTHVL 个时，TXP 由硬件清零	是
	在 RxFIFO 中接收到数据（一个数据包可用——FIFO 阈值）	RXP	RXPIE	RxFIFO 包含的采样数少于 FTHVL 个时，RXP 由硬件清零	是
	TXP 和 RXP 均有效	DXP	DXPIE	TXP 或 RXP 被清零时	是
	发送传输已填充	TXTF	TXTFIE	向 TXTFC 写入 1	否
	下溢	UDR	UDRIE	向 UDRC 写入 1	是
	上溢	OVR	OVRIE	向 OVRC 写入 1	是
	CRC 错误	CRCE	CRCEIE	向 CRCEC 写入 1	是
	TI 帧格式错误	TIFRE	TIFREIE	向 TIFREC 写入 1	否
	模式故障	MODF	MODFIE	向 MODFC 写入 1	否
	传输结束（完整传输序列完成——基于 TSIZE 值）	EOT	EOTIE	向 EOTC 写入 1	是
	主模式处于挂起状态	SUSP		向 SUSPC 写入 1	是
	TxFIFO 发送完成（TxFIFO 为空）	TXC(3)		总线上开始进行发送活动时，TXC 由硬件清零	否

1. 在各实例中，所有中断事件都能将系统从睡眠模式唤醒。有关从具体停止和待机模式退出的实例功能的详细信息，请参见“功能取决于工作模式”表。
2. 有关待机模式可用性的信息，请参见*第 38.3 节：SPI 实现*。
3. TXC 标志的行为取决于 TSIZE 设置。当 TSIZE>0 时，该标志完全遵循 EOT 标志，包括由 EOTC 清零。

38.8 I2S 主要特性

- 全双工通信
- 单工通信（仅作为发送器或接收器）
- 主模式或从模式操作
- 8 位可编程线性预分频器
- 数据长度可以是 16 位、24 位或 32 位(a)
- 在主模式下通道长度可以是 16 位或 32 位，在从模式下则可为任一值
- 可编程时钟极性
- 错误标志指示，以提高可靠性：下溢、上溢和帧错误
- 嵌入式 Rx 和 TxFIFO
- 支持的 I^2S 协议：
  - I^2S Philips 标准
  - MSB 对齐标准（左对齐）
  - LSB 对齐标准（右对齐）
  - PCM 标准（支持短帧同步和长帧同步）
- 数据排序可编程（LSb 在前或 MSb 在前）
- 用于发送和接收的 DMA 功能
- 可输出主时钟以驱动外部音频元件：
  - 对于所有 I2S 模式，$F_{MCK} = 256 \times F_{WS}$
  - 对于所有 PCM 模式，$F_{MCK} = 128 \times F_{WS}$

注意： *F_{MCK} 是主时钟频率，F_{WS} 是音频采样频率。*

38.9 I2S 功能描述

38.9.1 I2S 一般描述

图 466 所示的框图同样适用于 I2S 模式。

I2SMOD 位置 1 时，SPI/I2S 块可工作在 I2S/PCM 模式下。专用寄存器 (SPI_I2SCFGR) 可用于配置专用 I2S 参数，包括时钟发生器和串行链路接口。

将 SPI/I2S 置于主模式下时，I2S/PCM 功能使用时钟发生器生成通信时钟。此时钟发生器也是主时钟输出 (MCK) 的源。

诸如 RxFIFO、TxFIFO、DMA 等资源和部分中断信号与 SPI 功能共享。低功耗模式功能也可用于 I2S 模式，请参见*第 38.6 节：SPI 低功耗模式*和*第 38.10 节：I2S 中断*。

a. 并非始终可用，请参见*第 38.3 节：SPI 实现*以确认是否支持 24 位和 32 位数据宽度。

38.9.2 与 SPI 功能共享的引脚

I2S 与 SPI 共享以下四个共用引脚：

- SDO：串行数据输出（映射到 MOSI 引脚上），用于在主模式下发送音频采样以及在从模式下接收音频采样。请参见*第 1421 页的串行数据线交换*一节。
- SDI：串行数据输入（映射到 MISO 引脚上），用于在主模式下接收音频采样以及在从模式下发送音频采样。请参见*第 1421 页的串行数据线交换*一节。
- WS：字选择（映射到 SS 引脚上），表示帧同步。它在主模式下被配置为输出，在从模式下被配置为输入。
- CK：串行时钟（映射到 SCK 引脚上），表示串行位时钟。它在主模式下被配置为输出，在从模式下被配置为输入。

当某些外部音频设备需要使用主时钟输出时，可以使用其他引脚：

- MCK：主时钟（单独映射），适用于 I2S 配置为主模式的情况。

38.9.3 I2S/PCM 模式下可用的位域

选择 I2S/PCM 模式（I2SMOD = “1”）时，一些位域不再可用，必须将其强制设为特定值，以便保证 I2S/PCM 功能正常工作。*表 310* 列出了 I2S/PCM 模式下可用的位和位域，并指示了其中哪些必须强制设为特定值。

表 310. PCM/I2S 模式下可用的位域

寄存器名称	PCM/I2S 模式下可用的位域	关于其他位域的限制
SPI/I2S 控制寄存器 1 (SPI_CR1)	IOLOCK、CSUSP、CSTART、SPE	其他位域设为其各自的复位值
SPI/I2S 控制寄存器 2 (SPI_CR2)	-	设为复位值
SPI/I2S 配置寄存器 1 (SPI_CFG1)	TXDMAEN、RXDMAEN、FTHLV	其他位域设为其各自的复位值
SPI/I2S 配置寄存器 2 (SPI_CFG2)	AFCNTR、LSBFRST、IOSWP	其他位域设为其各自的复位值
SPI/I2S 中断使能寄存器 (SPI_IER)	TIFREIE、OVRIE、UDRIE、TXPIE、RXPIE	
SPI/I2S 状态寄存器 (SPI_SR)	SUSP、TIFRE、OVR、UDR、TXP、RXP	其他标志不相关
SPI/I2S 中断/状态标志清零寄存器 (SPI_IFCR)	SUSPC、TIFREC、OVRC、UDRC	其他位域设为其各自的复位值
SPI/I2S 接收数据寄存器 (SPI_RXDR)	整个寄存器	-
SPI/I2S 多项式寄存器 (SPI_CRCPOLY)	-	设为复位值
SPI/I2S 发送器 CRC 寄存器 (SPI_TXCRC)	-	
SPI/I2S 接收器 CRC 寄存器 (SPI_RXCRC)	-	
SPI/I2S 下溢数据寄存器 (SPI_UDRDR)	-	
SPI/I2S 配置寄存器 (SPI_I2SCFGR)	整个寄存器	-

38.9.4 从模式和主模式

SPI/I2S 块都支持采用 I2S 或 PCM 协议的主模式和从模式。
在主模式下，CK、WS 和 MCK 信号均被设置为输出。

在从模式下，CK 和 WS 信号被设置为输入。从模式下无法使用信号 MCK。

为了改善从模式下 SPI/I2S 块的稳健性，外设根据 WS 信号在每次接收和发送时再同步。这意味着：

- 使用 I2S Philips 标准时，在每个 WS 沿跳变之后，每个数据的移入或移出都会被触发一个时钟周期。
- 使用 I2S MSB 对齐标准时，如果检测到 WS 沿跳变，则会触发各个数据的移入或移出。
- 使用支持短帧同步的 PCM 标准时，在 WS 上升沿之后，每个数据的移入或移出都会被触发一个时钟周期。
- 使用支持长帧同步的 PCM 标准时，只要检测到 WS 上升沿，就会触发各个数据的移入或移出。

注意： *这种再同步机制不适用于 I2S LSB 对齐标准。*

另请注意，将 SPI/I2S 配置为从模式时，无需提供内核时钟。

38.9.5 支持的音频协议

I2S/PCM 接口支持四种音频标准，可使用 SPI_I2SCFGR 寄存器中的 I2SSTD[1:0] 和 PCMSYNC 位对其进行配置。

使用 I2S 协议时，在左右两个通道上时分复用音频数据。WS 信号用于指示哪个通道必须视为左通道，以及哪个通道必须视为右通道。

在 I2S 主模式下，支持以下四种帧格式：

- 将 16 位数据封装在 16 位通道中
- 将 16 位数据封装在 32 位通道中
- 将 24 位数据封装在 32 位通道中(a)
- 将 32 位数据封装在 32 位通道中(a)

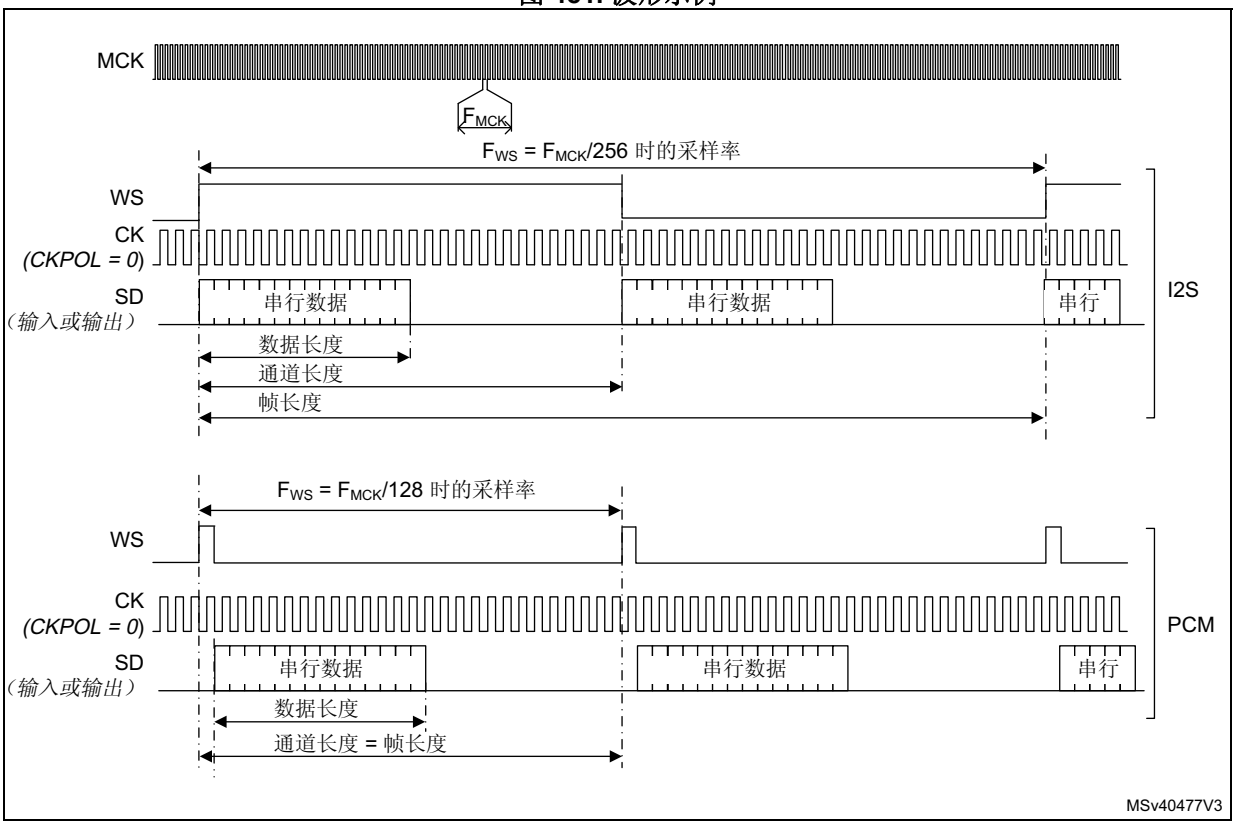
在 PCM 主模式下，支持以下三种帧格式：

- 将 16 位数据封装在 16 位通道中
- 将 16 位数据封装在 32 位通道中
- 将 24 位数据封装在 32 位通道中(a)

下图显示了本节中使用的主要定义：数据长度、通道长度和帧长度。

a. 并非始终可用，请参见*第 38.3 节：SPI 实现*以确认是否支持 24 位和 32 位数据宽度。

图 481. 波形示例



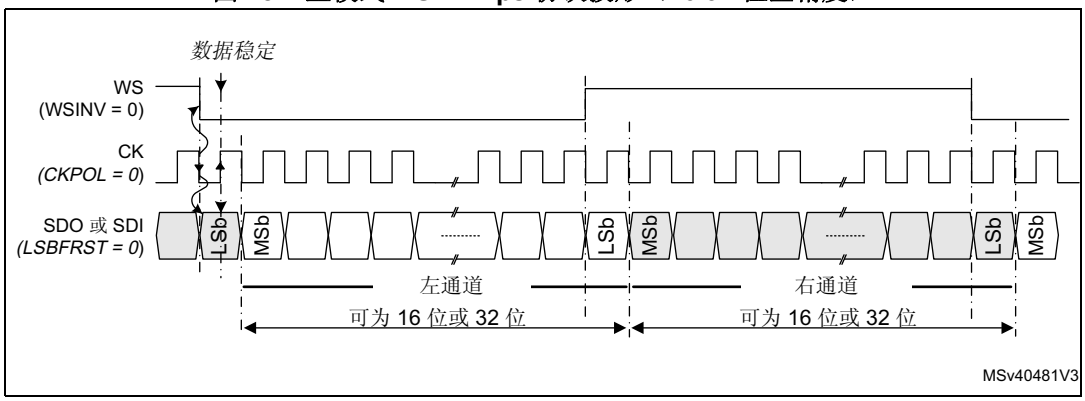
为简单起见，在下文的图中，SDI 代表串行数据输入，SDO 代表串行数据输出。有关串行数据线方向控制的详细信息，请参见*串行数据线交换*一节。

I²S Philips 标准

要选择 I2S Philips 标准，将 I2SSTD 设为 0b00 即可。主模式和从模式均支持该标准。

对于该标准，在第一个位（对于 I2S Philips 标准，为 MSb）可用之前 WS 信号将翻转一个 CK 时钟周期。WS 的下降沿跳变表示下一个传输的数据将用于左通道，上升沿跳变则表示下一个传输的数据用于右通道。

图 482. 主模式 I2S Philips 协议波形（16/32 位全精度）



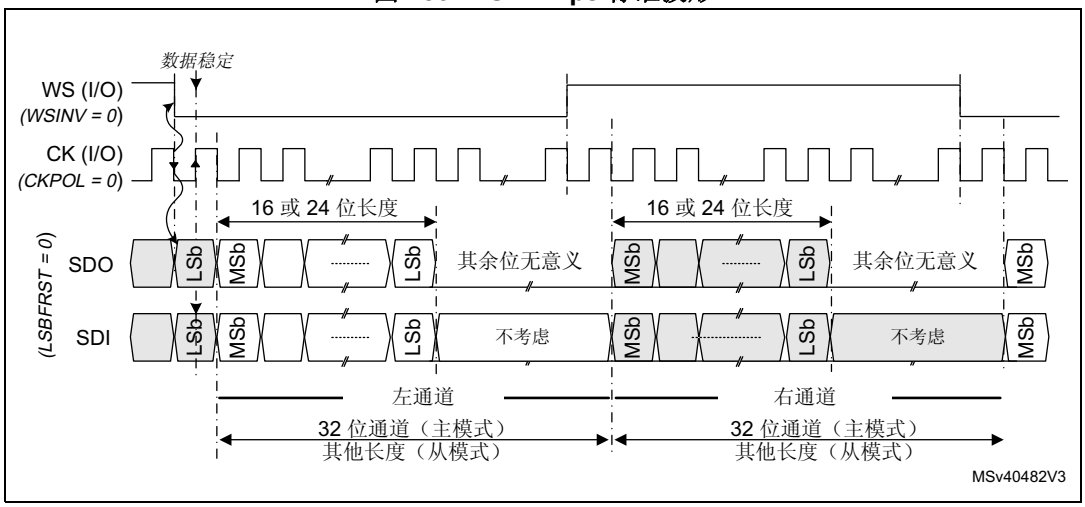
1. 并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），有关支持的数据大小，请参见*第 38.3 节：SPI 实现*。

为与 I2S Philips 协议匹配，要将 CKPOL 清零。有关 WS 信号处理的信息，请参见 *CK 采样边沿的选择*。

图 482 所示为通道长度等于数据长度时 SPI/I2S 所生成的波形示例。更确切地说，这适用于 CHLEN = 0 且 DATLEN = 0b00 或 CHLEN = 1 且 DATLEN = 0b10 的情况。

有关 WS 信号处理的信息，请参见 *WS 反转控制*。

图 483. I2S Philips 标准波形



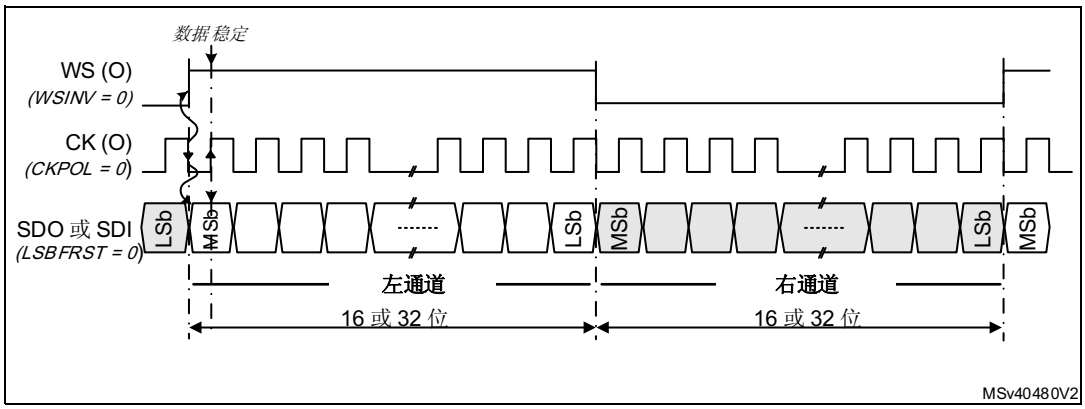
1. 并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），有关支持的数据大小，请参见*第 38.3 节：SPI 实现*。

如果通道长度大于数据长度，则将 SPI/I2S 配置为发送模式时，其余位没有意义。这同时适用于主模式和从模式。

MSB 对齐标准

对于该标准，提供第一个数据位时，WS 信号翻转。如果 WS 为高电平，则传输的数据表示左通道；如果 WS 为低电平则表示右通道。

图 484. 主模式 MSB 对齐的 16 位或 32 位全精度长度

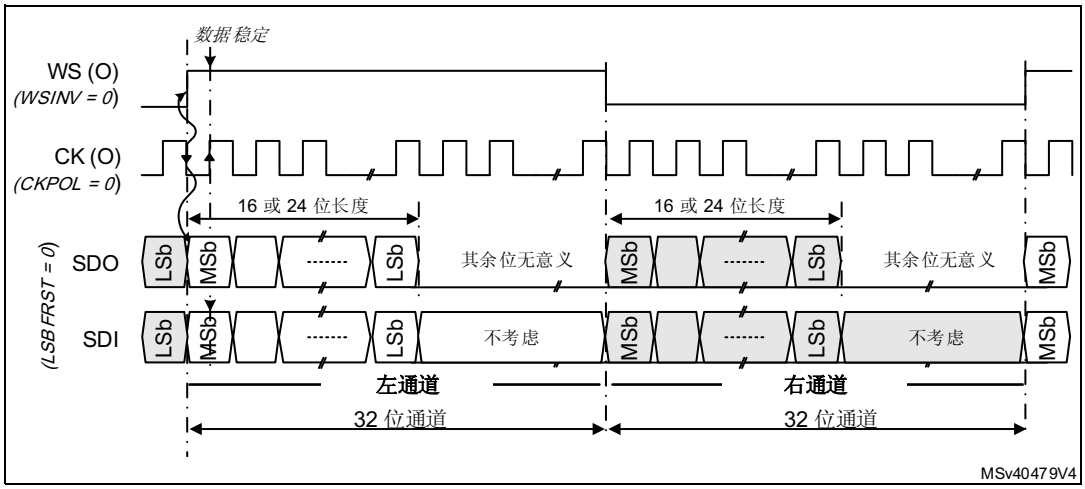


1. 并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），请参见*第 38.3 节：SPI 实现*确认支持的数据大小。

为与 I2S MSB 对齐协议匹配，要将 CKPOL 清零。有关 WS 信号处理的信息，请参见 *CK 采样边沿的选择*。

有关 WS 信号处理的信息，请参见 *WS 反转控制*。

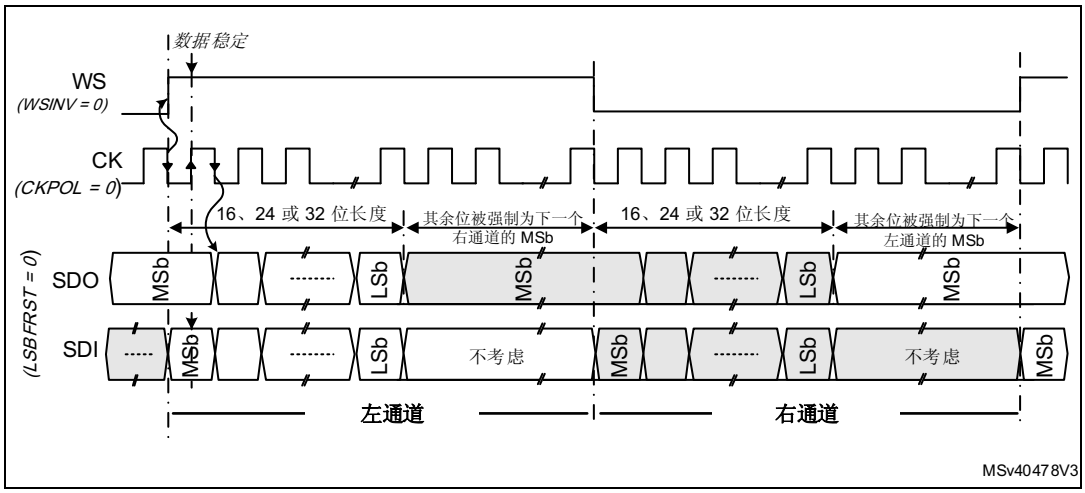
图 485. 主模式 MSB 对齐的 16 位或 24 位数据长度



1. 并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），请参见*第 38.3 节：SPI 实现*确认支持的数据大小。

如果通道长度大于数据长度，则将 SPI/I2S 配置为主发送模式时，其余位没有意义。在从器件发送模式下，其余位被强制为要生成的下一个数据的第一位的值，以避免出现时序问题（请参见*图 486*）。

图 486. 从模式 MSB 对齐



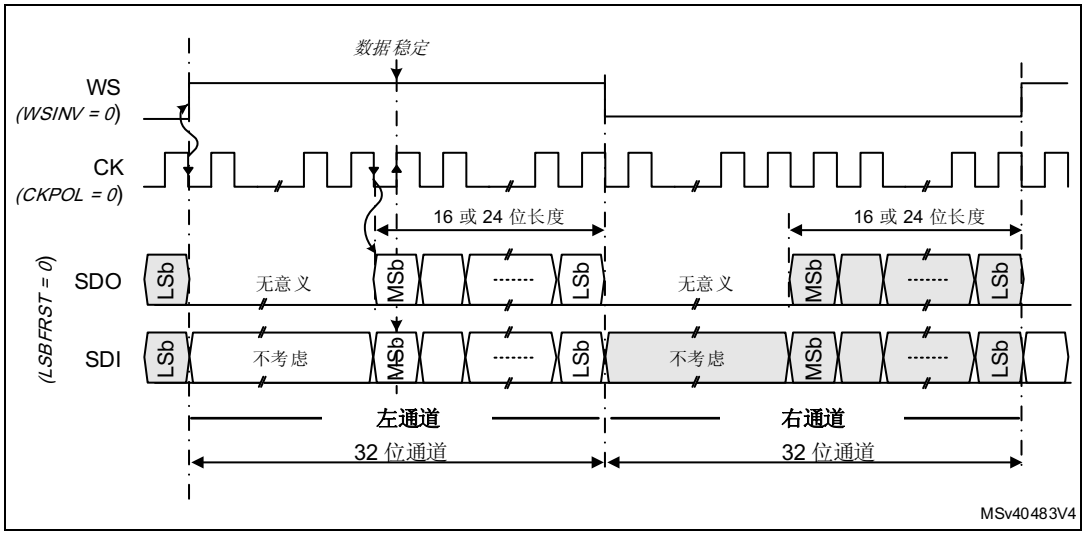
1. 并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），请参见*第 38.3 节：SPI 实现*确认支持的数据大小。

LSB 对齐标准

该标准与主模式下的 MSB 对齐标准类似（对于 16 位和 32 位全精度帧格式，没有任何不同）。LSB 对齐的 16 位或 32 位全精度格式的波形与 MSB 对齐模式下的波形类似（请参见*图 484*），因为通道和数据的长度相同。

注意： *在 LSB 对齐格式下，主模式和从模式仅支持 16 位和 32 位通道长度。这是因为，如果发送器和接收器侧不清楚通道长度，则无法正确传输数据。*

图 487. LSB 对齐的 16 位或 24 位数据长度



1. 并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），请参见*第 38.3 节：SPI 实现*确认支持的数据大小。

为与 I2S LSB 对齐协议匹配，要将 CKPOL 清零。有关 WS 信号处理的信息，请参见 *CK 采样边沿的选择*。

有关 WS 信号处理的信息，请参见 *WS 反转控制*。

PCM 标准

对于 PCM 标准，无需使用通道信息。可使用两种 PCM 模式（短帧和长帧），并且可使用 SPI_I2SCFGR 寄存器中的 PCMSYNC 位来进行选择模式。

在 PCM 长帧模式下：

- 主模式下，WS 信号置为有效的时间固定为 13 个 CK 周期，
- WS 信号被置为有效后，会立即接收或发送第一个数据位。

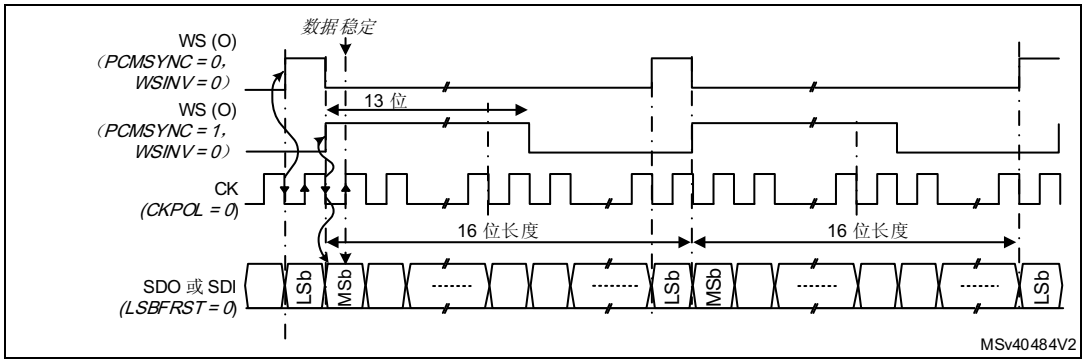
在 PCM 短帧模式下：

- 主模式下，WS 信号置为有效的时间固定为一个 CK 周期，
- WS 被置为有效后的一个 CK 周期内，会接收或发送第一个数据位。

对于两种 PCM 模式：

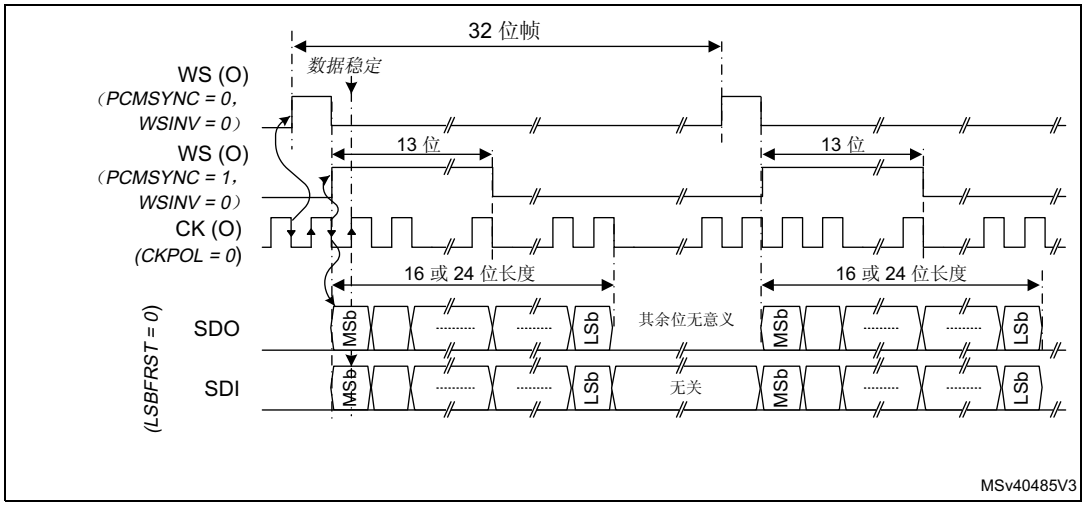
- 第一个数据位是 MSb 或 LSb，具体取决于 LSBFIRST 位的值。
- 可通过 CKPOL 位选择 CK 采样边沿。
- 可通过 WSINV 位反转 WS 信号。有关 WS 信号处理的信息，请参见 *WS 反转控制*。

图 488. 帧长度等于数据长度时的主模式 PCM



将通道长度设置为 32 位时，可以使用 16 位或 24 位数据大小。

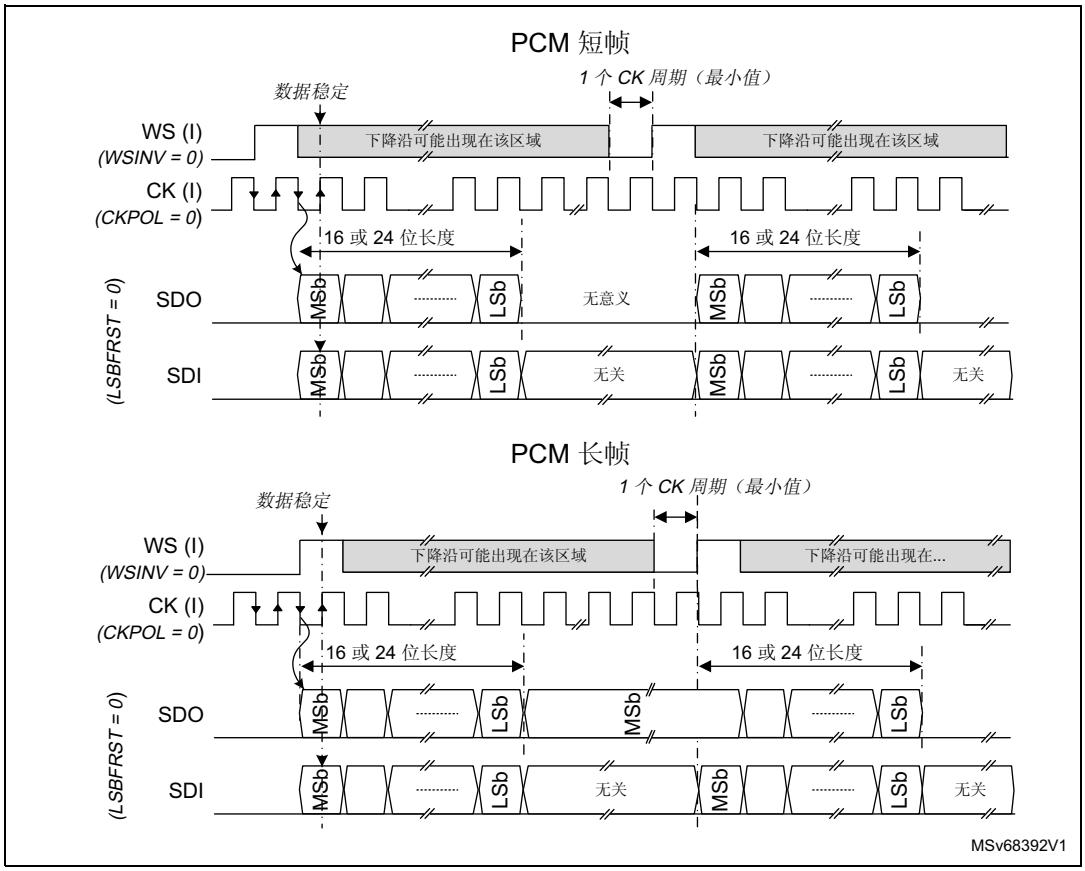
图 489. 主模式 PCM 标准波形（16 位或 24 位数据长度）



1. 并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），请参见*第 38.3 节：SPI 实现*确认支持的数据大小。

如果将 PCM 协议用于从模式，则帧长度可以不是 16 位或 32 位。如*图 490* 所示，在从模式下，由于 WS 的上升沿检测到帧开始，因此可以接受 WS 的各种脉冲宽度。唯一的限制是 WS 必须在至少一个 CK 周期内恢复其非有效状态。

图 490. 从模式 PCM 波形



1. 并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），请参见*第 38.3 节：SPI 实现*确认支持的数据大小。

注意： *通道长度大于数据长度时，在从器件 PCM 长帧模式下，如果 TXFIFO 包含下一个要发送的数据，则剩余位的发送将强制为要生成的下一个数据的第一位的值。*

注意： *在从模式下，必须始终正确编程 CHLEN（无论 FIXCH 值如何）。例如，如果 CHLEN 清零（16 位长度），则即使 DATLEN 不为 0，数据传输也会被截断为 16 位。为了避免这种情况，如果 DATLEN = 1 或 2，则必须将 CHLEN 置 1。*

38.9.6 更灵活的串行接口

从模式下的帧长度可变

在从模式下，只要通道长度大于数据长度，便可接受非 16 位或 32 位的通道长度。这适用于除 I2S LSB 对齐协议外的所有协议。

数据排序

对于所有数据格式和通信标准，可通过 *SPI/I2S 配置寄存器 2 (SPI_CFG2)* 的 LSBFRST 位来选择数据排序（MSb 在前或 LSb 在前）。

CK 采样边沿的选择

SPI/I2S 配置寄存器 (SPI_I2SCFGR) 的位 CKPOL 允许用户为从模式和主模式选择 CK 的采样边沿极性，这适用于所有协议。

- CKPOL = 0 时，在 CK 的下降沿更改串行数据 SDO 和 WS（主模式下），在上升沿读取串行数据 SDI 和 WS（从模式下）。
- CKPOL = 1 时，在 CK 的上升沿更改串行数据 SDO 和 WS（主模式下），在下降沿读取串行数据 SDI 和 WS（从模式下）。

WS 反转控制

可通过将 WSINV 置 1，为主模式和从模式反转默认 WS 信号极性，这适用于所有协议。默认情况下，WS 极性如下：

- 使用 I2S Philips 标准时，WS 对于左通道为低电平，对于右通道为高电平
- 在 MSB/LSB 对齐模式下，WS 对于左通道为高电平，对于右通道为低电平
- 在 PCM 模式下，帧起始通过 WS 的上升沿表示。

WSINV 置 1 时，WS 极性将反转，因此：

- 使用 I2S Philips 标准时，WS 对于左通道为高电平，对于右通道为低电平
- 在 MSB/LSB 对齐模式下，WS 对于左通道为低电平，对于右通道为高电平
- 在 PCM 模式下，帧起始通过 WS 的下降沿表示。

WSINV 位于 *SPI/I2S 配置寄存器 (SPI_I2SCFGR)* 中。

IO 的控制

SPI/I2S 块借助 *SPI/I2S 配置寄存器 2 (SPI_CFG2)* 的 AFCNTR 位，允许在使能 SPI/I2S 之前将 WS 和 CK 信号稳定在无效状态。

可通过使用以下序列编程 CKPOL 和 WSINV 来完成上述过程：

假设 AFCNTR 最初被清零：

- 将 I2SMOD 置 1（以通知硬件通过 CKPOL 和 WSINV 控制 CK 和 WS 极性）。
- 将位 CKPOL 和 WSINV 设为所需值。
- 将 AFCNTR 置 1。
  随即会根据 CKPOL 和 WSINV 值设置 CK 和 WS I/O 的无效电平，即使尚未使能 SPI/I2S，亦如此。
- 然后执行 I2S/PCM 的激活序列

表 311 所示为将 AFCNTR 位置 1 时以及在使能 SPI/I2S 块之前的 WS 和 CK 信号电平（即，无效电平）。请注意，WS 电平还取决于所选协议。

表 311. AFCNTR = 1 时在使能 SPI/I2S 之前的 WS 和 CK 电平

WSINV	I2SSTD		使能 SPI/I2S 之前的 WS 电平
0	I2S 标准 (00)	→	高
	其他	→	低
1	I2S 标准 (00)	→	低
	其他	→	高

CKPOL		使能 SPI/I2S 之前的 CK 电平
0	→	低
1	→	高

注意： *SPI2S 处于从模式下时，位 AFCNTR 不得置 1。*

串行数据线交换

SDI 和 SDO 的方向取决于 *SPI/I2S 配置寄存器 2 (SPI_CFG2)* 的 IOSWP 位，以及从/主模式。*表 312* 给出了该功能的详细信息。

表 312. 串行数据线交换

方向	IOSWP	主模式		从模式	
		输入线	输出线	输入线	输出线
RX	0	SDI	-	SDO	-
	1	SDO	-	SDI	-
TX	0	-	SDO	-	SDI
	1	-	SDI	-	SDO
全双工	0	SDI	SDO	SDO	SDI
	1	SDO	SDI	SDI	SDO

38.9.7 启动序列

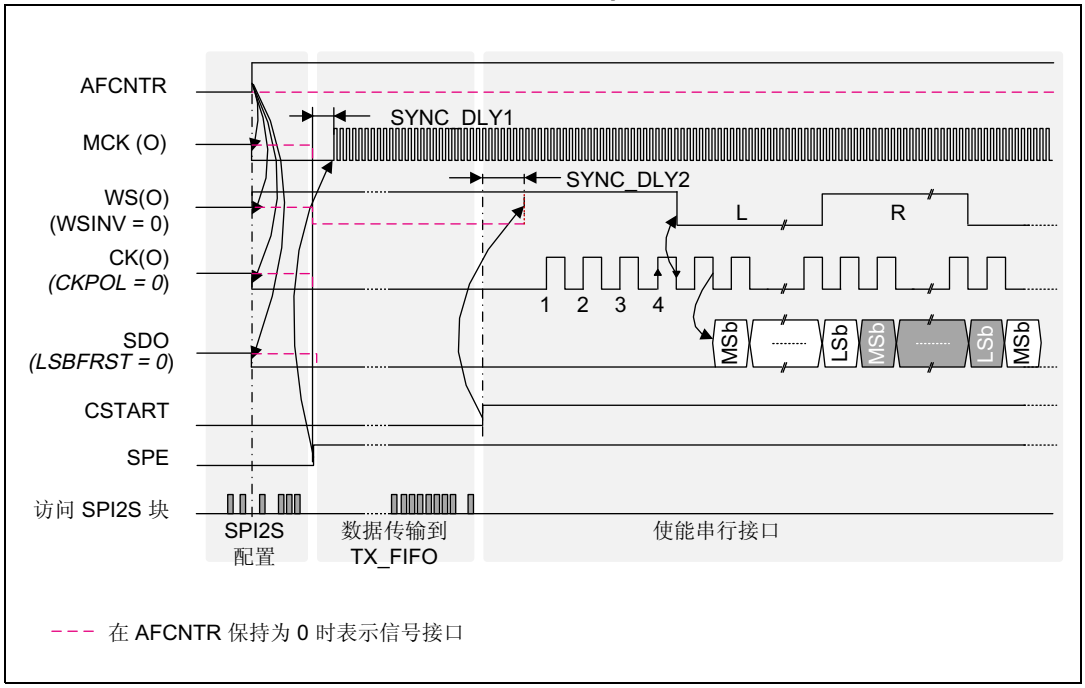
将 SPE 位清零时，不允许用户对 SPI_RXDR 和 SPI_TXDR 寄存器进行读操作和写操作，但允许访问其他寄存器。

如果应用要使用 SPI/I2S 块，则用户必须按以下步骤进行操作：

1. 确保将 SPE 清零，否则向 SPE 写入 0。
2. 根据所需配置去编程所有配置和控制寄存器。有关详细的编程示例，请参见 *第 38.9.16 节*。
3. 通过将 SPI_IFCR 寄存器的 USPC、TIFREC、OVRC 和 UDRC 位置 1 来清零所有状态标志。请注意，如果未将标志 SUSP 清零（通过 SUSPC 位），则 CSTART 控制位无效。
4. 将 SPE 位置 1，以激活 SPI/I2S 块。该位置 1 时，串行接口仍然被禁止，但 DMA 和中断服务处于工作状态，因而可将数据传输到 TxFIFO。当 SPE 变为 1 时，也会开始生成 MCK。
5. 将 CSTART 位置 1，以激活串行接口。

如*图 491* 所示，对于符合 I2S Philips 标准的主 TX，一旦 CSTART 位置 1 且 TxFIFO 非空，就会在一个重新同步延迟 (SYNC_DLY2) 后开始生成 WS 和 CK 信号。请注意，在 WS 下降沿之前的 4 个上升沿，位时钟 CK 处于激活状态，以确保外部从器件可正确检测 WS 沿跳变。其他标准的相应行为与此类似。

图 491. 启动序列，I2S Philips 标准，主模式



1. 如图所示，一旦 SPE 位置 1，就可以使能 MCK。它在同步延迟 (SYNC_DLY1) 之后生成。有关详细信息，请参见*第 38.9.9 节：时钟发生器*中的 MCK 生成。
2. 请注意，如果将 AFCNTR 位置 1，则在配置阶段，WS 和 CK 信号的电平将由 SPI/I2S 块进行控制

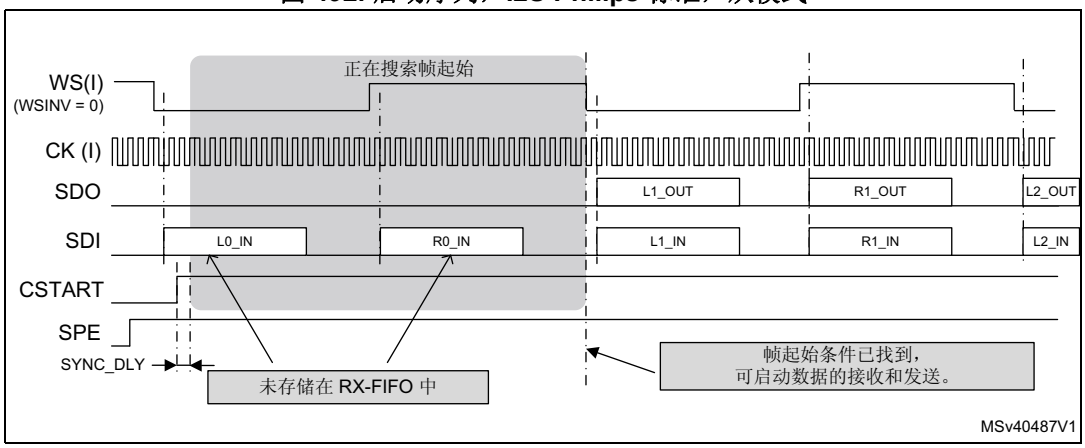
注意： *由于时钟域再同步，因此在约 3 个 CK 时钟周期 (SYNC_DLY2) 后硬件会考虑 CSTART 位。*

在从模式下，如果将 CSTART 位置 1，则在满足帧起始条件时将开始数据传输：

- 对于 I2S Philips 标准，帧起始条件为 WS 信号下降沿。将在延迟一个时钟周期后开始发送/接收。
  如果 WSINV = 1，则帧起始条件为上升沿。
- 对于其他协议，帧起始条件为 WS 信号上升沿。对于 MSB 对齐协议，在 WS 上升沿开始发送/接收。对于 PCM 协议，将在延迟一个时钟周期后开始发送/接收。
  如果 WSINV = 1，则帧起始条件为下降沿。

图 492 所示为采用 I2S Philips 标准的从模式的启动序列示例。

图 492. 启动序列，I2S Philips 标准，从模式



注意： *由于时钟域再同步，因此在 2 个 CK 时钟周期 (SYNC_DLY) 后硬件会考虑 CSTART 位。*

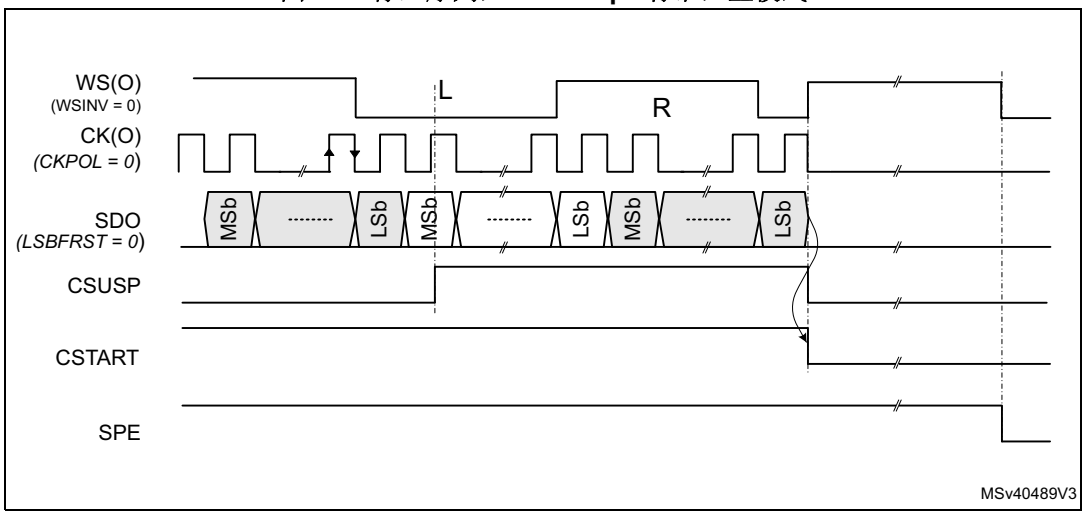
38.9.8 停止序列

应用可通过将 SPE 位清零来停止 I2S/PCM 传输。在这种情况下，通信将立即停止，而不会等到当前帧结束。

在主模式下，还可以在当前帧结束时停止 I2S/PCM 传输。为此，用户必须将 CSUSP 位置 1，并轮询 CSTART 位，直到其值为 0。当前立体声（如果选择了 I2S 模式）或单声道采样完全移入或移出时，CSTART 位将变为 0。之后可将 SPE 位清零。

图 493 所示为主模式下的停止序列示例。CSUSP 位置 1，在传送左采样期间，传输继续进行，直到传输右采样的最后一位。随后 CSTART 和 CSUSP 返回至 0，CK 和 WS 信号返回到无效状态，用户可以将 SPE 位清零。

图 493. 停止序列，I2S Philips 标准，主模式

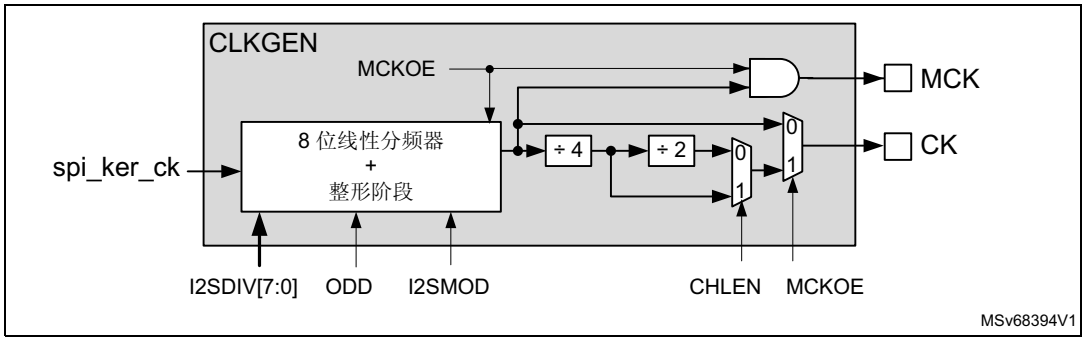


注意： *在从模式下，停止序列仅由 SPE 位控制。*

38.9.9 时钟发生器

将 I2S 或 PCM 配置为主模式时，用户需要对时钟发生器进行编程，以便基于所需频率产生帧同步 (WS)、位时钟 (CK) 和主时钟 (MCK)。
如果将 I2S 或 PCM 用于从模式下，则无需配置时钟发生器。

图 494. I^2S 时钟发生器架构



MCK、CK 和 WS 上生成的频率主要取决于 I2SDIV、ODD、CHLEN 和 MCKOE。MCKOE 位指示是否需要生成主时钟。主时钟的频率是帧同步频率的 256 或 128 倍。如果要为外部音频编解码器提供参考时钟，则通常需要该主时钟。

注意： *在主模式下，总线时钟频率 (F_{pclk}) 和位时钟频率 (F_{CK}) 之比无特定限制。总线时钟频率必须足够高，这样才能支持数据吞吐量。*

在 I2S 模式下，生成主时钟 (MCKOE = 1) 时，可通过以下公式得出帧同步的频率：

$$F_{WS} = \frac{F_{i2s_clk}}{256 \times \{(2 \times I2SDIV) + ODD\}}$$

在 PCM 模式下通过以下公式得出频率：

$$F_{WS} = \frac{F_{i2s_clk}}{128 \times \{(2 \times I2SDIV) + ODD\}}$$

此外，可通过下列公式得出 MCK 频率 (F_{MCK})：

$$F_{MCK} = \frac{F_{i2s_clk}}{\{(2 \times I2SDIV) + ODD\}}$$

在 I2S 模式下，禁止主时钟 (MCKOE = 0) 时，可通过以下公式得出帧同步的频率：

$$F_{WS} = \frac{F_{i2s_clk}}{32 \times (CHLEN + 1) \times \{(2 ¥ I2SDIV) + ODD\}}$$

在 PCM 模式下通过以下公式得出频率：

$$F_{WS} = \frac{F_{i2s_clk}}{16 \times (CHLEN + 1) \times \{(2 ¥ I2SDIV) + ODD\}}$$

其中，F_{WS} 表示帧同步的频率，F_{i2s_clk} 表示提供给 SPI/I2S 块的内核时钟的频率。

注意： *CHLEN 和 ODD 既可以为 0 也可以为 1。*

ODD = 0 时，I2SDIV 可以为 0 到 255 之间的任一值；但当 ODD = 1 时，不允许值 I2SDIV = 1。

I2SDIV = 0 时，{(2 x I2SDIV) + ODD} 强制为 1。

注意： *{(2 x I2SDIV) + ODD} 为奇数时，MCK 或 CK 信号的占空比不会为 50%。必须谨慎使用奇数比：因为它会影响建立和保持时间的裕量。例如，如果 {(2 x I2SDIV) + ODD} = 5，则占空比可为 40%。*

表 313 所示为适用于 I2S 模式的时钟发生器编程示例。

MCK 生成

当满足以下条件时，会生成主时钟 MCK：

- I2SMOD 必须等于 1，
- I2SCFG 必须选择主模式，
- MCKOE 必须置 1。
- SPE 必须置 1

表 313. 适合常见 I2S 频率的 CLKGEN 编程示例

i2s_clk (MHz)	通道长度（位）	I2SDIV	ODD	MCK	采样率：F_{WS} (kHz)
12.288	16	12	0	否	16
12.288	32	6	0		16
12.288	16	6	0		32
12.288	32	3	0		32
49.152	16	16	0		48
49.152	32	8	0		48
49.152	16	8	0		96
49.152	32	4	0		96
49.152	16	4	0		192
49.152	32	2	0		192
4.096	16 或 32	0	-	是	16
24.576	16 或 32	3	0		
49.152	16 或 32	3	0		32
12.288	16 或 32	0	-		48
49.152	16 或 32	2	0		
61.44	16 或 32	2	1		
98.304	16 或 32	2	0		96
196.608	16 或 32	2	0		192

38.9.10 内部 FIFO

I2S 接口可将专用 FIFO 用于 RX 和 TX 路径。可通过 SPI_TXDR 寄存器将要发送的采样写入 TxFIFO。通过 SPI_RXDR 寄存器读取 RxFIFO。

数据对齐和排序

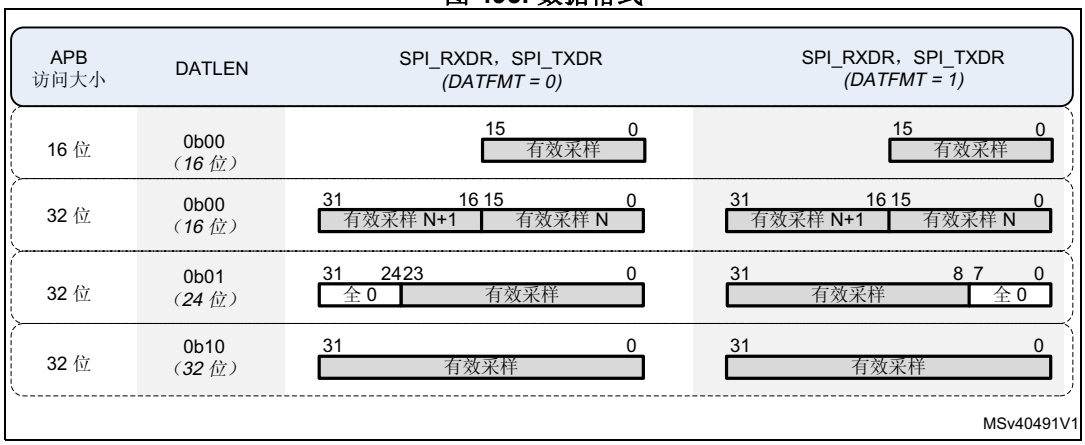
可通过 DATFMT 位选择对齐到 SPI_RXDR 和 SPI_TXDR 寄存器的数据。

另请注意，SPI_RXDR 或 SPI_TXDR 中数据的格式还取决于通过 APB 总线对这些寄存器进行访问的方式。

图 495 显示了 APB 访问大小、DATFMT 和 DATLEN 之间的允许设置。

注意： *APB 访问大小为 32 位且 DATLEN = 0 时必须格外注意。对于读操作，RxFIFO 必须至少包含两个数据，否则读取的数据将无效。同样，对于写操作，TxFIFO 必须至少有两个空位置，否则数据会丢失。*

图 495. 数据格式



1. 在 I2S 模式下，采样 N 表示左采样，采样 N+1 表示右采样。
2. 并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），请参见*第 38.3 节：SPI 实现*确认支持的数据大小。

可根据可编程的 FIFO 阈值生成中断或 DMA 请求。FIFO 阈值对 RX 通用，TxFIFO 可通过 FTHVL 进行调整。

在 I2S 模式下，左音频采样和右音频采样在 FIFO 中交错。这表示，对于发送操作，用户必须开始先用左采样然后再用右采样来填充 TxFIFO，依此类推。对于接收模式，从 RxFIFO 中读取的第一个数据用于表示左通道，下一个数据用于表示右通道，依此类推。

请注意，SPE 位清零时，FIFO 的读指针和写指针将复位。

更多详细信息，请参见*第 38.9.11 节*和*第 38.9.15 节*。

FIFO 大小优化

字节是 FIFO 的基本元素。这样可以优化 FIFO 位置数。例如，数据大小固定为 24 位时，每个音频采样都会占用 3 个基本 FIFO 元素。

举例来说，具有 16 个基本元素的 FIFO 的深度可为：

- 8 个采样，适合 DATLEN = 0 的情况（16 位）
- 5 个采样，适合 DATLEN = 1 的情况（24 位）(a)
- 4 个采样，适合 DATLEN = 2 的情况（32 位）(a)

38.9.11 FIFO 状态标志

应用程序可通过两个状态标志来全面监视 I2S 接口的状态。两个标志均可生成一个中断请求。如果使能 RXPIE 位则会生成接收中断，如果使能 TXPIE 位则会生成发送中断。这些位均位于 SPI_IER 寄存器中。

达到了 TxFIFO 阈值 (TXP)

该标志置 1 时，表示 TxFIFO 至少包含 FTHLV 个空位置。这样，要发送的 FTHLV 个新数据就可以写入 SPI_TXDR 中。当空位置数量少于 FTHLV 个时，TXP 标志将复位。请注意，如果禁止 I2S（SPE 位复位），则 TXP = 1。

a. 并非始终可用，请参见*第 38.3 节：SPI 实现*以确认是否支持 24 位和 32 位数据宽度。

达到了 RxFIFO 阈值 (RXP)

该标志置 1 时，表示 RxFIFO 中至少有 FTHLV 个有效数据，因此用户可以通过 SPI_RXDR 读取这些数据。RxFIFO 包含的数据数量少于 FTHLV 个时，该标志会复位。

有关 I2S 模式下中断功能的更多信息，请参见*第 38.10 节*。

38.9.12 下溢情况的处理

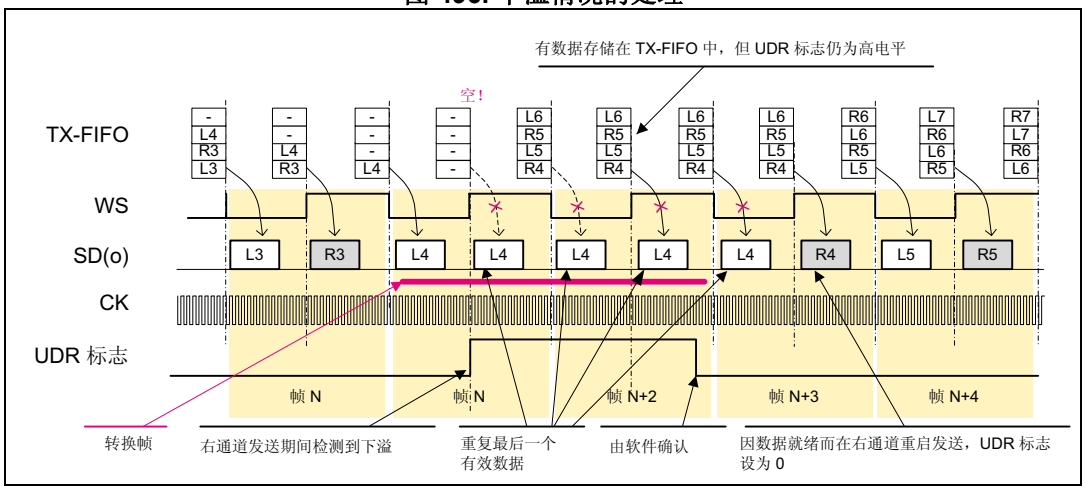
在发送模式下，需要将新数据载入移位寄存器中，而 TxFIFO 已空时，会检测到下溢情况。在这种情况下，UDR 标志将置 1，并且至少一个音频帧包含异常数据。

对于 I2S 模式，存在一种硬件机制，可防止出现偏离情况（左右通道互换）。检测到下溢情况时，如果实际通道长度与 CHLEN 位选择的长度相符，则将重复移位寄存器中存在的最后一个有效数据。在其他情况下，将重复未定义的数据。通常，如果块编程为从 TX 模式，并且外部主音频设备使用的通道长度不是 16 位或 32 位，则在下溢时重复的数据值不会被定义。

如下图所示，发生下溢时，只要不满足重新开始发送的条件，外设就会重复左右通道上的最后一个有效数据。重新开始发送：

- TxFIFO 中存在足够的数据时，
- UDR 标志由软件清零时，
- 满足重新开始条件时：
  - 如果在需要发送右通道数据时发生下溢，则当需要发送右通道时重新开始发送，或者
  - 如果在需要发送左通道数据时发生下溢，则在需要发送左通道时重新开始发送。

图 496. 下溢情况的处理



当模块配置为其中一种 PCM 模式时，如果 TxFIFO 中有足够的数据，并且 UDR 标志清零，发送将在下一帧开始时重新开始。

如果 SPI_IER 寄存器的 UDRIE 位置 1，则 UDR 标志可触发中断。通过向 SPI_IFCR 寄存器的 UDRC 位写入 1 来将 UDR 位清零。

注意： *主模式或从模式下都可能发生下溢情况。在主模式下，发生下溢时，不会对 WS、CK 和 MCK 信号进行门控。*

出于再同步的原因，在经过至少 2 个 CK 时钟周期后，硬件将考虑 UDR 标志的变化。

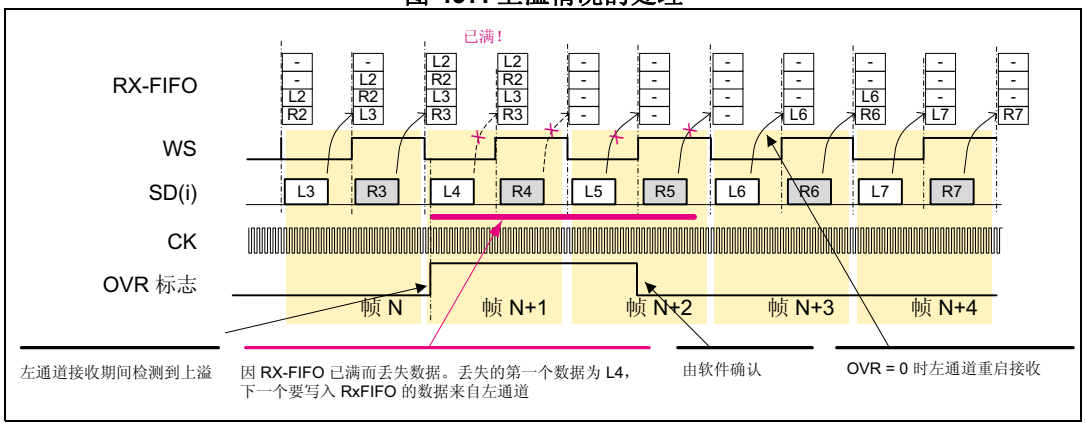
38.9.13 上溢情况的处理

在接收模式下，当移位寄存器需要将新数据存储到 RxFIFO 中，而 RxFIFO 已满时，就会检测到溢出情况。在这种情况下，OVR 标志会置 1，并且传入的数据会丢失。

在 I2S 模式下，存在一种硬件机制，可防止出现偏离情况（左右通道互换）。如下图所示，发生上溢时，只要不满足重新启动条件，外设就会停止向 RxFIFO 中写入数据。

RxFIFO 中有足够的空间，且 OVR 标志清零时，如果在接收右通道数据时发生上溢，则块会通过向 RxFIFO 中写入下一个右通道开始；或者，如果在接收左通道数据时发生上溢，则块会通过写入下一个左通道开始。

图 497. 上溢情况的处理



当该模块配置为 PCM 模式时，发生上溢时，只要不满足重新启动的条件，该模块就会停止向 RxFIFO 中写入数据。RxFIFO 中有足够的空间，且 OVR 标志清零时，下一个接收的数据将写入 RxFIFO 中。

如果 SPI_IER 寄存器中的 OVRIE 位置 1，可产生中断。通过向 SPI_IFCR 寄存器的 OVRC 位写入 1 来将 OVR 位清零。

注意： *主模式或从模式下都可能发生上溢情况。在主模式下，发生上溢时，不会对 WS、CK 和 MCK 信号进行门控。*

38.9.14 帧错误检测

配置为从模式时，SPI/I2S 块会检测以下两种帧错误：

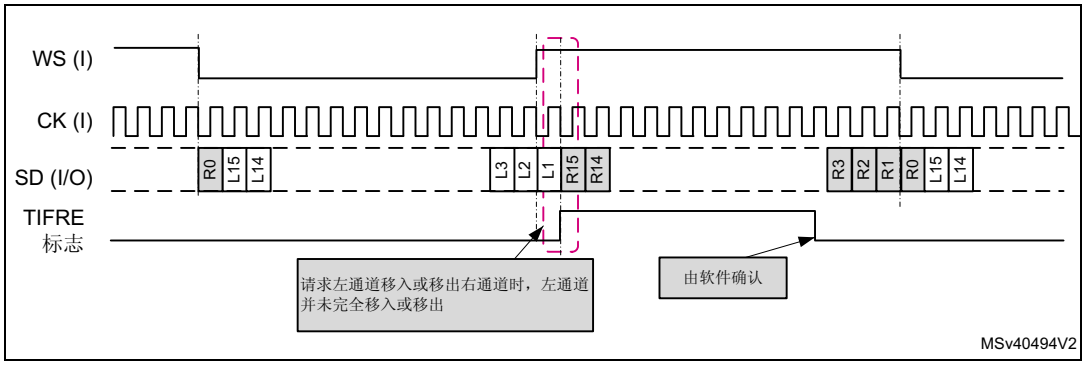
- 在先前数据的移入或移出未完成时接收到帧同步（早期帧错误）。通过 FIXCH = 0 选择此模式。
- 在意外位置发生帧同步。通过 FIXCH = 1 选择此模式。

在从模式下，如果由外部主器件提供的帧长度不是 32 位或 64 位，则用户必须将 FIXCH 清零。由于 SPI/I2S 会将每次传输均与 WS 同步，因此不存在偏离风险，但是在噪声环境下，如果 CK 信号出现毛刺，则采样可能会受到影响，而应用不会了解这一情况。

如果外部主器件提供的帧长度等于 32 位或 64 位，则用户可以将 FIXCH 置 1 并相应调整 CHLEN。由于 SPI/I2S 会将每次传输均与 WS 同步，因此仍然不存在偏离风险，并且如果每个通道边界之间的位时钟量不同于 CHLEN，则帧错误标志 (TIFRE) 将置 1。

图 498 所示为帧错误检测示例。SPI/I2S 块处于从模式下，左通道的位时钟周期数不足以用于移入或移出数据。该图显示的是，正在进行的传输被中断，下一次传输已启动，以保持与 WS 信号对齐。

图 498. 帧错误检测，FIXCH = 0

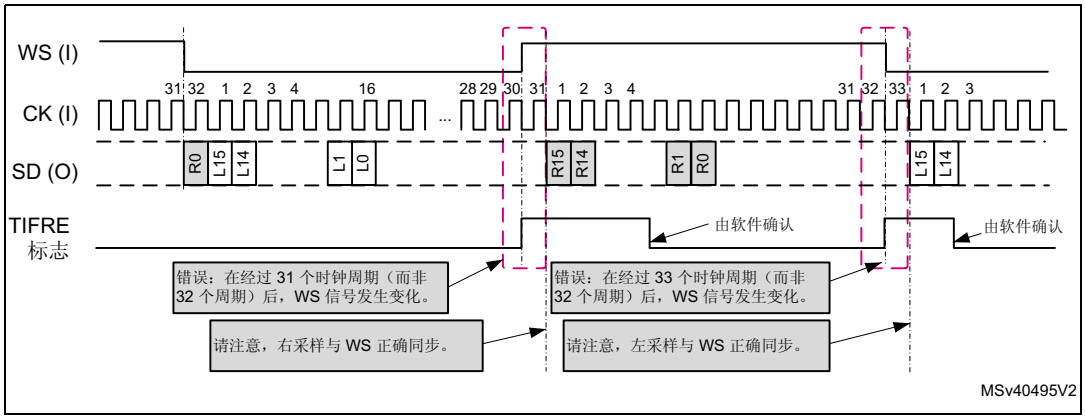


如果 TIFREIE 位置 1，则可生成中断。通过向 SPI_IFCR 寄存器的 TIFREC 位写入 1 来将帧错误标志 (TIFRE) 清零。

可以通过将位 FIXCH 置 1 来扩大帧错误标志的覆盖范围。该位置 1 时，SPI/I2S 在从模式下需要固定通道长度。这意味着根据 CHLEN，预期通道长度可以是 16 位或 32 位。如*图 499* 所示，在该模式下，SPI/I2S 模块能够检测 WS 信号是否在预期时间（过早或过晚）发生变化。

注意： *图 498* 和*图 499* 所示为适用于从器件发送模式的机制，但该机制还适用于从器件接收和从全双工模式。

图 499. 帧错误检测，FIXCH = 1



通常会因噪声环境干扰了 WS 或 CK 信号的良好接收而进行帧错误检测。

注意： *如果在同一帧内出现上溢和早期帧，则 SPI/I2S 无法正常恢复。在这种情况下，用户必须禁止并重新使能 SPI/I2S。*

38.9.15 DMA 接口

I2S/PCM 模式与 SPI 功能共用相同的 DMA 请求线。TX 和 RX 路径有单独的 DMA 通道。每个 DMA 通道均可通过 SPI_CFG1 寄存器的 RXDMAEN 和 TXDMAEN 位使能。

在接收模式下，DMA 接口的工作如下：

1. 硬件评估 RxFIFO 级别。
2. 如果 RxFIFO 至少包含 FTHLV 个采样，则会生成 FTHLV DMA 请求。
   – FTHLV DMA 请求完成时，硬件返回到第 1 步，如此循环往复。
3. 如果 RxFIFO 包含的采样数少于 FTHLV 个，则不会生成 DMA 请求，硬件返回到第 1 步，如此循环往复。

在发送模式下，DMA 接口的工作如下：

1. 硬件评估 TxFIFO 级别。
2. 如果 TxFIFO 至少包含 FTHLV 个空位置，则会生成 FTHLV DMA 请求。
   – FTHLV DMA 请求完成时，硬件返回到第 1 步，如此循环往复。
3. 如果 TxFIFO 包含的空位置数少于 FTHLV 个，则不会生成 DMA 请求，硬件返回到第 1 步，如此循环往复。

38.9.16 编程示例

主模式 I2S Philips 标准，全双工

该示例显示如何在使用主时钟的情况下，以 48 kHz 采样率对接口进行编程，以便支持主全双工模式下的 I2S Philips 标准协议。假定 SPI/I2S 正在从电路的时钟控制器接收 61.44 MHz 的内核时钟 (i2s_clk)。在上例中，我们假设需要对外部音频编解码器进行编程，例如在开始传输之前通过 I2C 接口进行编程。此外，假设该外部音频编解码器需要 MCK 以接受 I2C 命令。

步骤

1. 通过 RCC 模块使能总线接口和内核时钟，根据需要将复位信号置为有效并加以释放。
2. 对 AFMUX 进行编程以选择所需的 I/O。在当前示例中，需要 MCK、CK、WS、SDO、SDI。
3. 对时钟发生器进行编程，以提供 MCK 时钟，以及获得恰好为 48 kHz 的帧同步率。将 I2SDIV 设置为 2，将 ODD 置 1，MCKOE 置 1。
4. 编程串行接口协议：CKPOL = 0，WSINV = 0，LSBFRST = 0，CHLEN = 1（每通道 32 位），DATLEN = 1（24 位），I2SSTD = 0（I2S Philips 标准），I2SCFG = 5（主全双工模式），I2SMOD = 1（针对 I2S/PCM 模式）。
5. 通过将所需值设置为 FTHLV 来调整 FIFO 阈值。例如，如果需要 2 个音频采样的阈值，则 FTHLV = 1。
6. 如果应用希望通过 DMA 执行数据传输，则将 TXDMAEN 和 RXDMAEN 位置 1。
7. 将 SPI_IER 寄存器中的所有中断使能位域清零。
8. 通过将 SPI_IFCR 寄存器的 SUSPC、TIFREC、OVRC、UDRC 置 1 将所有状态标志清零。
9. 将 SPE 位置 1，会生成 MCK。在此处的示例中，SPE = 1 后，不会立即填充 TxFIFO。这样，即使要发送的音频采样尚不可用，应用程序也可以通过 I2C 接口配置外部音频编解码器。下文提供了一个 *替代序列*。

10. 当应用程序想要开始数据传输时：
    – 如果数据传输使用 DMA，则要：
      ——编程 DMA 外设：两个通道，一个通道用于 RX，另一个通道用于 TX
      ——针对 TX 路径使用有效的音频采样初始化存储器缓冲区——使能 DMA 通道，
      ——如果需要，使能 UDRIE 和 OVRIE 等中断事件以检测传输错误。
      ——使能 DMA 通道。如果 TXDMAEN 已置 1，则 TxFIFO 会填满。
    – 如果数据传输通过中断完成：
      ——使能中断事件 UDRIE、OVRIE、TXPIE 和 RXPIE（通过写 1）。中断请求会立即激活，允许中断处理程序填充 TxFIFO。
11. 最后，可通过将 CSTART 位置 1 来使能 SPI/I2S 串行接口。CSTART 位位于 SPI_CR1 寄存器中。

替代序列

- 步骤 1 到 8 与上一个序列类似。
- 如果数据传输使用 DMA，则要：
  – 编程 DMA 外设：两个通道，一个通道用于 RX，另一个通道用于 TX
  – 针对 TX 路径使用有效的音频采样初始化存储器缓冲区
  – 如果需要，使能 UDRIE 和 OVRIE 等中断事件以检测传输错误。
  – 使能 DMA 通道
- 如果数据传输通过中断完成：
  – 使能中断事件 UDRIE、OVRIE、TXPIE 和 RXPIE（通过写 1）。中断请求会立即激活，允许中断处理程序将 TxFIFO 填满。
- 将 SPE 置 1，该位置 1 后，可能会发生以下操作：
  – 如果使能中断生成，SPI/I2S 将生成中断请求，允许中断处理程序填充 TxFIFO。
  – 如果使能 DMA 传输，SPI/I2S 将生成 DMA 请求来填充 TxFIFO
  – MCK 已生成。然后，应用程序可以通过 I2C 接口配置外部音频编解码器。
- 最后，可通过将 CSTART 位置 1 来使能 SPI/I2S 串行接口。CSTART 位位于 SPI_CR1 寄存器中。

在主模式下的停止步骤

1. 将 CSUSP 位置 1，以停止正在进行的传输
2. 检查 CSTART 位的值，直至其变为 0
3. 通过将 SUSPC 置 1 清零 SUSP 标志
4. 停止 DMA 外设、总线时钟...
5. 清零 SPE 位以禁止 SPI/I2S 模块

从模式 I2S Philips 标准，接收

该示例显示如何以 48 kHz 采样率对接口进行编程，以便支持从器件接收器模式下的 I2S Philips 标准协议。请注意，在从模式下，SPI/I2S 块不能控制接收采样的采样率。在本示例中，我们假定外部主器件具有通道长度为 24 位的 I2S 帧结构。因此，FIXCH 置 1 时，我们无法使用为帧错误检测提供的功能。

步骤

1. 通过 RCC 模块使能总线接口和内核时钟，根据需要将复位信号置为有效并加以释放，
2. 对 AFMUX 进行编程以选择所需的 I/O。在当前示例中为 CK、WS、SDI，
3. 编程串行接口协议：CKPOL = 0，WSINV = 0，LSBFRST = 0，FIXCH = 0（因为通道长度不为 16 位和 32 位），DATLEN = 0（16 位），I2SSTD = 0（Philips 协议），I2SCFG = 1（从器件接收模式），I2SMOD = 1（针对 I2S 模式）。
4. 通过将所需值设置为 FTHLV 来调整 FIFO 阈值。例如，如果需要 2 个音频采样的阈值，则 FTHLV = 1。
5. 清除所有状态标志寄存器。
6. 使能会生成中断的标志，例如 OVRIE 和 TIFRE。
7. 如果数据传输使用 DMA，则要：
   – 编程 DMA 外设：一个 RX 通道
   – 使能 DMA 通道
   – 在 SPI/I2S 块中，通过将 RXDMAEN 位置 1 来使能 DMA。
8. 如果通过中断进行数据传输，则用户必须通过将 RXPIE 位置 1 来使能中断。
9. 将 SPE 置 1。
10. 最后，用户可将 CSTART 位置 1 来使能串行接口。SPI/I2S 在下次出现外部主器件发送的左通道数据时，开始将数据存储到 RxFIFO 中。

在从模式下的停止步骤

1. 清零 SPE 位以禁止 SPI/I2S 模块
2. 停止 DMA 外设、总线时钟 ...

38.10 I2S 中断

在 PCM/I2S 模式下，可根据*表 314* 中描述的事件生成中断 (**spi_it**) 或唤醒事件信号 (**spi_wkup**)。

中断事件可分别进行使能和禁止。

表 314. I2S 中断请求

中断向量	中断事件	事件标志	事件 / 中断清除方法	退出睡眠模式	退出停止模式	退出待机模式
SPI	达到 TxFIFO 阈值	TXP	TxFIFO 包含的空位置数少于 FTHLV 个时	是	是	否
	达到 RxFIFO 阈值	RXP	RxFIFO 包含的采样数少于 FTHLV 个时			
	上溢错误	OVR	向 OVRC 写入 1			
	下溢错误	UDR	向 UDRC 写入 1			
	帧错误	TIFRE	向 TIFREC 写入 1		否	

38.11 SPI/I2S 寄存器

38.11.1 SPI/I2S 控制寄存器 1 (SPI_CR1)

SPI/I2S control register 1

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IOLOCK
															rs

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCRCINI	RCRCINI	CRC33_17	SSI	HDDIR	CSUSP	CSTART	MASRX	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPE
rw	rw	rw	rw	rw	w	rs	rw								rw

位 31:17 保留，必须保持复位值。

位 16 **IOLOCK**：锁定相关 I/O 的 AF 配置 (Locking the AF configuration of associated I/Os)

仅当 SPI 被禁止 (SPE = 0) 时，才可以通过软件更改该位。如果发生 MODF 事件，将由硬件清零

0：AF 配置未锁定

1：AF 配置已锁定

该位置 1 时，不能再修改 SPI_CFG2 寄存器内容。SPI 使能 (SPE = 1) 时，该位受到写保护。

位 15 **TCRCINI**：发送器 CRC 计算初始化模式控制 (CRC calculation initialization pattern control for transmitter)

0：应用全零模式

1：应用全 1 模式

位 14 **RCRCINI**：接收器 CRC 计算初始化模式控制 (CRC calculation initialization pattern control for receiver)

0：应用全零模式

1：应用全 1 模式

位 13 **CRC33_17**：32 位 CRC 多项式配置 (32-bit CRC polynomial configuration)

0：不使用全尺寸（33 位或 17 位）CRC 多项式

1：使用全尺寸（33 位或 17 位）CRC 多项式

位 12 **SSI**：内部 SS 信号输入电平 (Internal SS signal input level)

仅当 SSM 位置 1 时，该位才有效。此位的值将在内部作用到外设 SS 输入上，并忽略 SS 引脚的 I/O 值。

位 11 **HDDIR**：半双工模式下的 Rx/Tx 方向 (Rx/Tx direction at Half-duplex mode)

在半双工配置下，HDDIR 位确立了数据传输的 Rx/Tx 方向。在全双工或任何单工配置下，忽略该位。

0：SPI 为接收器

1：SPI 为发送器

位 10 **CSUSP**：主挂起请求 (Master suspend request)

该位读为零。

在主模式下，该位由软件置 1 时，CSTART 位将在当前帧结束时复位，通信将被挂起。用户必须检查 SUSP 标志以检查帧事务的结束。

在进入低功耗模式之前，主模式通信必须挂起（通过该位或将 TXDR 保持为空）。

软件挂起后，必须清零 SUSP 标志并在下一个事务开始之前禁止然后重新使能 SPI。

位 9 **CSTART**：主传输开始 (Master transfer start)

如果仅使能 SPI 来启动 SPI 或 I2S/PCM 通信，则可以由软件将该位置 1。在 SPI 模式下，传输结束 (EOT) 标志置 1 或接受事务挂起请求时，该位由硬件清零。在 I2S/PCM 模式下，它也由硬件清零，如*第 38.9.8 节：停止序列*所述。

0：主传输处于空闲状态

1：主传输正在进行，或通过自动挂起被临时挂起

在 SPI 模式下，该位仅在主模式下使用。如果使能了发送，则只有当发送 FIFO 中存在数据时，才能启动或继续通信。

位 8 **MASRX**：接收模式下主器件自动挂起 (Master automatic SUSP in Receive mode)

该位由软件置 1 和清零，用于控制主器件接收器模式下的连续 SPI 传输以及自动管理，以避免出现上溢情况。

0：无论是否出现上溢情况，SPI 流/时钟生成都会连续。（数据会丢失）

1：在达到上溢情况之前，SPI 流在 RxFIFO 已满状态下暂时挂起。SPI 通信挂起时，SUSP 标志会置 1。

SPI 通信由硬件自动挂起时，可能会因内部同步延迟而导致同步输出下一帧的几个位。

因此，数据大小低于 8 位时，自动挂起并不十分可靠。在这种情况下，可通过结合在 MIDI 参数保持非零值时所应用的数据帧之间插入的延迟，来实现安全挂起；数据大小和交错 SPI 周期之和应始终至少产生长度为 8 个 SPI 时钟周期的间隔。软件清零 SUSP 位后，通信将恢复并继续进行后续位传输，不受任何限制。在 SUSP 位清零之前，用户必须根据 RXP 标志指示读出 RxFIFO 中所有可用的数据包，尽可能地释放 RxFIFO 空间，以防止任何后续挂起。

位 7:1 保留，必须保持复位值。

位 0 **SPE**：串行外设使能 (Serial peripheral enable)

该位由软件置 1 和清零。

0：禁止串行外设。

1：使能串行外设。

当 SPE = 1时，SPI 数据传输使能，SPI_CFG1 和 SPI_CFG2 配置寄存器，SPI_CR1 寄存器中的 CRCPOLY、UDRDR、IOLOCK 位受到写保护。只有在 SPE = 0 时，才能更改。

当 SPE = 0 时，任何 SPI 操作都会停止并被禁止，除 MODF 中断请求外的所有已使能中断的事件的挂起请求都将被阻止（但它们的挂起仍会传播 spi_plck 时钟的请求），SS 输出在主器件上停用，RDY 信号在从器件上保持未就绪状态，内部状态机复位，所有 FIFO 内容都将刷新，CRC 计算会进行初始化，接收数据寄存器读为零。

MODF 错误标志有效时，SPE 清零且不能置 1。

38.11.2 SPI/I2S 控制寄存器 2 (SPI_CR2)

SPI/I2S control register 2

偏移地址：0x04

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15:0 **TSIZE[15:0]**：当前传输的数据量 (Number of data at current transfer)

当这些位由软件更改时，必须禁止 SPI。

CSTART 置 1 且 TSIZE 存储零值时，将初始化无限事务。如果使能了 CRC，则不能将 TSIZE 设置为值 0xFFFF 或 0x3FFF。

注意：TSIZE[15:10] 位在有限功能集实例中保留，并且必须保持复位值。

38.11.3 SPI/I2S 配置寄存器 1 (SPI_CFG1)

SPI/I2S configuration register 1

偏移地址：0x08

复位值：0x0007 0007

使能 SPI 时，该寄存器的内容受到写保护。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BPASS	MBR[2:0]			Res.	Res.	Res.	Res.	Res.	CRCEN	Res.	CRCSIZE[4:0]				
rw	rw	rw	rw						rw		rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXDMAEN	RXDMAEN	Res.	Res.	Res.	Res.	UDRCFG	FTHLV[3:0]				DSIZE[4:0]				
rw	rw					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **BPASS**：主波特率时钟发生器的预分频器旁路

0：禁止旁路

1：使能旁路

位 30:28 **MBR[2:0]**：主波特率预分频器设置
000：SPI 主时钟/2
001：SPI 主时钟/4
010：SPI 主时钟/8
011：SPI 主时钟/16
100：SPI 主时钟/32
101：SPI 主时钟/64
110：SPI 主时钟/128
111：SPI 主时钟/256
注意：在 TI 模式下工作的从器件也考虑 MBR 设置（请参见第 38.5.1 节：TI 模式）。

位 27:23 保留，必须保持复位值。

位 22 **CRCEN**：硬件 CRC 计算使能 (hardware CRC computation enable)
0：禁止 CRC 计算
1：使能 CRC 计算

位 21 保留，必须保持复位值。

位 20:16 **CRCSIZE [4:0]**：要处理和比较的 CRC 帧的长度 (length of CRC frame to be transacted and compared)
处理或比较 CRC 结果时，多项式计算会考虑最高有效位。多项式的长度不受该设置的影响。
00000：保留
00001：保留
00010：保留
00011：4 位
00100：5 位
00101：6 位
00110：7 位
00111：8 位
.....
11101：30 位
11110：31 位
11111：32 位
该值必须设置与数据大小 (DSIZE[4:0]) 相等或者是数据大小的倍数。其最大大小对应于实例的 DSIZE 最大值。
注意：对于数据大小被限制为 16 位的外设实例，CRCSIZE 位域的最高有效位将保留。

位 15 **TXDMAEN**：Tx DMA 数据流使能 (Tx DMA stream enable)
0：禁止 Tx DMA
1：使能 Tx DMA

位 14 **RXDMAEN**：Rx DMA 数据流使能 (Rx DMA stream enable)
0：禁止 Rx-DMA
1：使能 Rx-DMA

位 13:10 保留，必须保持复位值。

位 9 **UDRCFG**：下溢条件时从器件发送器行为 (behavior of slave transmitter at underrun condition)
0：从器件发送一个常数，该常数由用户在 SPI_UDRDR 寄存器中定义
1：从器件重复上一次从主器件接收到的数据。当从器件配置为仅发送模式 (COMM[1:0] = 01) 时，将重复全零模式。
有关更多详细信息，请参见*图 480：从器件检测下溢情况的可选配置*。

位 8:5 **FTHLV[3:0]**：FIFO 阈值级别 (FIFO threshold level)

定义单个数据包的数据帧数。数据包的大小不应超过 FIFO 空间的 1/2。

0000：1 个数据
0001：2 个数据
0010：3 个数据
0011：4 个数据
0100：5 个数据
0101：6 个数据
0110：7 个数据
0111：8 个数据
1000：9 个数据
1001：10 个数据
1010：11 个数据
1011：12 个数据
1100：13 个数据
1101：14 个数据
1110：15 个数据
1111：16 个数据

如果配置的数据包大小与数据寄存器访问并行位数对齐，则 SPI 接口的效率更高：

– 如果将 SPI 数据寄存器作为 16 位寄存器进行访问，且 DSIZE ≤ 8 位，则最好选择 FTHLV = 2、4、6。
– 如果将 SPI 数据寄存器作为 32 位寄存器进行访问，且 DSIZE > 8 位,则最好选择 FTHLV = 2、4、6，而如果 DSIZE ≤ 8 位，则最好选择 FTHLV = 4、8、12。

注意：FTHLV[3:2] 位在有限功能集实例中保留

位 4:0 **DSIZE [4:0]**：单个 SPI 数据帧的位数 (number of bits in at single SPI data frame)

00000：未使用
00001：未使用
00010：未使用
00011：4 位
00100：5 位
00101：6 位
00110：7 位
00111：8 位
.....
11101：30位
11110：31 位
11111：32 位

注意：在某些情况下，最大数据大小最多可限制为 16 位。在有限功能集实例中，DSIZE[2:0] 位保留，并且必须保持复位状态。然后 DSIZE[4:3] 位会控制数据大小的下一个设置：
00xxx：8 位
01xxx：16 位
10xxx：24 位
11xxx：32 位

38.11.4 SPI/I2S 配置寄存器 2 (SPI_CFG2)

SPI/I2S configuration register 2

偏移地址：0x0C

复位值：0x0000 0000

SPI 使能或 SPI_CR1 寄存器中的 IOLOCK 位置 1 时，该寄存器的内容受到写保护。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFCNTR	SSOM	SSOE	SSIOP	Res.	SSM	CPOL	CPHA	LSBFRST	MASTER	SP[2:0]			COMM[1:0]		Res.
rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IOSWP	RDIOP	RDIOM	Res.	Res.	Res.	Res.	Res.	MIDI[3:0]				MSSI[3:0]			
rw	rw	rw						rw	rw	rw	rw	rw	rw	rw	rw

位 31 **AFCNTR**：复用功能 GPIO 控制 **(alternate function GPIOs control)**

仅当 SPE = 0 时，才考虑该位。

0：外设在禁止时不对 GPIO 进行控制

1：外设始终控制所有相关 GPIO

出于特定配置原因（例如 CRC 复位、CPHA 或 HDDIR 更改）而必须暂时禁止 SPI 时，将该位置 1 可将配置为复用功能模式的相关输出强制为当前 SPI 配置所对应的状态，以防止出现毛刺。

注意：该位也可用于 PCM 和 I2S 模式。

注意：时钟处于从模式下时，位 AFCNTR 不得置 1。

位 30 **SSOM**：主模式 SS 输出管理 (SS output management in Master mode)

SSOE 使能时，在主模式下考虑该位。其允许在两次连续数据传输之间配置 SS 输出。

0：SS 保持为有效电平，直到数据传输完成，随即会通过 EOT 标志变为无效状态

1：当 MIDI[3:0] > 1 时，SPI 数据帧与 SS 非有效脉冲交错

位 29 **SSOE**：SS 输出使能 (SS output enable)

该位仅在主模式下使用

0：禁止 SS 输出，SPI 可在多主配置下工作

1：使能 SS 输出。SPI 不能在多主模式环境下工作。在传输完成后或者 SPI 被禁止后，它会根据 SSOM、MIDI、MSSI、SSIOP 位设置将 SS 引脚强制为无效电平

位 28 **SSIOP**：SS 输入/输出极性 (SS input/output polarity)

0：SS 信号低电平有效

1：SS 信号高电平有效

位 27 保留，必须保持复位值。

位 26 **SSM**：SS 信号输入的软件管理 (software management of SS signal input)

0：SS 输入值由 SS PAD 决定

1：SS 输入值由 SSI 位决定

当主器件使用硬件 SS 输出时（SSM = 0 且 SSOE = 1），SS 信号输入在内部强制为非有效状态，以防止出现主模式故障错误。

位 25 **CPOL**：时钟极性 (clock polarity)

0：空闲时 SCK 信号为低电平

1：空闲时 SCK 信号为高电平

位 24 **CPHA**：时钟相位 (clock phase)
0：从第一个时钟边沿开始采样数据
1：从第二个时钟边沿开始采样数据

位 23 **LSBFRST**：数据帧格式 (data frame format)
0：先发送 MSB
1：先发送 LSB
注意：该位也可用于 PCM 和 I2S 模式。

位 22 **MASTER**：SPI 主模式 (SPI Master)
0：SPI 从模式
1：SPI 主模式

位 21:19 **SP[2:0]**：串行协议 (serial protocol)
000：SPI Motorola
001：SPI TI
其他：保留，不得使用

位 18:17 **COMM[1:0]**：SPI 通信模式 (SPI Communication Mode)
00：全双工
01：单工发送器
10：单工接收器
11：半双工

位 16 保留，必须保持复位值。

位 15 **IOSWP**：交换 MISO 和 MOSI 引脚的功能 (swap functionality of MISO and MOSI pins)
0：不交换
1：交换 MOSI 和 MISO 引脚
该位置 1 时，MISO 和 MOSI 引脚复用功能互换。
原 MISO 引脚变为 MOSI 引脚，原 MOSI 引脚变为 MISO 引脚。
注意：该位还可用于 PCM 和 I2S 模式以交换 SDO 和 SDI 引脚。

位 14 **RDIOP**：RDY 信号输入/输出极性 (RDY signal input/output polarity)
0：信号高电平表示从器件准备通信
1：信号低电平表示从器件准备通信

位 13 **RDIOM**：RDY 信号输入/输出管理 (RDY signal input/output management)
0：RDY 信号在内部定义为永久有效（RDIOP 设置无效）
1：RDY 信号由专用引脚的复用功能输入（主模式）或输出（从模式）取代（RDIOP 设置有效）
注意：当 SPI_CFG1 寄存器中的 DSIZE 配置为小于 8 位时，RDIOM 位必须保持为零。

位 12:8 保留，必须保持复位值。

位 7:4 **MIDI [3:0]**：主模式数据间空闲 (master Inter-Data Idleness)
指定主模式下在两个连续数据帧之间插入的最小时间延迟（以 SPI 时钟周期数表示）。
0000：无延迟
0001：1 个时钟周期延迟
...
1111：15 个时钟周期延迟
注意：TI 模式不支持该功能。

位 3:0 **MSSI[3:0]**：主模式 SS 空闲

指定在使能 SSOE 时，在主模式下，开启会话的 SS 有效边沿与会话的第一个数据帧开始之间插入的额外延迟（以 SPI 时钟周期数表示）。

0000：无额外延迟

0001：增加了 1 个时钟周期的延迟

...

1111：增加了 15 个时钟周期的延迟

注意：TI 模式不支持该功能。

为了加入延迟，必须在会话之间禁止然后重新使能 SPI。

38.11.5 SPI/I2S 中断使能寄存器 (SPI_IER)

SPI/I2S interrupt enable register

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	MODFIE	TIFREIE	CRCEIE	OVRIE	UDRIE	TXTFIE	EOTIE	DXPIE	TXPIE	RXPIE
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:10 保留，必须保持复位值。

位 9 **MODFIE**：模式故障中断使能 (mode Fault interrupt enable)

0：禁止 MODF 中断

1：使能 MODF 中断

位 8 **TIFREIE**：使能 TIFRE 中断

0：禁止 TIFRE 中断

1：使能 TIFRE 中断

位 7 **CRCEIE**：CRC 错误中断使能 (CRC error interrupt enable)

0：禁止 CRC 中断

1：使能 CRC 中断

位 6 **OVRIE**：使能 OVR 中断

0：禁止 OVR 中断

1：使能 OVR 中断

位 5 **UDRIE**：UDR 中断使能 (UDR interrupt enable)

0：禁止 UDR 中断

1：使能 UDR 中断

位 4 **TXTFIE**：TXTFIE 中断使能 (TXTFIE interrupt enable)

0：禁止 TXTF 中断

1：使能 TXTF 中断

位 3 **EOTIE**：EOT、SUSP 和 TXC 中断使能 (EOT, SUSP and TXC interrupt enable)

0：禁止 EOT/SUSP/TXC 中断

1：使能 EOT/SUSP/TXC 中断

位 2 **DXPIE**：使能 DXP 中断
DXPIE 由软件置 1，并由 TXTF 标志置 1 事件清零。
0：禁止 DXP 中断
1：使能 DXP 中断

位 1 **TXPIE**：TXP 中断使能 (TXP Interrupt enable)
TXPIE 由软件置 1，并由 TXTF 标志置 1 事件清零。
0：禁止 TXP 中断
1：使能 TXP 中断

位 0 **RXPIE**：RXP 中断使能 (RXP interrupt enable)
0：禁止 RXP 中断
1：使能 RXP 中断

38.11.6 SPI/I2S 状态寄存器 (SPI_SR)

SPI/I2S status register

偏移地址：0x14

复位值：0x0000 1002

当 SPI 重新使能时，该寄存器的所有标志不会自动清除。它们需要专门通过标志清零寄存器进行特定的清零访问，如下面各个位的说明中所述。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CTSIZE[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXWNE	RXPLVL[1:0]		TXC	SUSP	Res.	MODF	TIFRE	CRCE	OVR	UDR	TXTF	EOT	DXP	TXP	RXP
r	r	r	r	r		r	r	r	r	r	r	r	r	r	r

位 31:16 **CTSIZE[15:0]**：当前 TSIZE 会话中剩余的数据帧数 (number of data frames remaining in current TSIZE session)
当总线上有数据传输时，该值并不十分可靠。
注意：CTSIZE[15:0] 位在有限功能集实例中不可用。

位 15 **RXWNE**：RxFIFO 字非空 (RxFIFO word not empty)
0：数据占用的 RxFIFO 空间少于 4 个字节
1：至少有四个字节的 RxFIFO 空间被数据占用
注意：该位值不依赖于 DSIZE 设置，并随残留数据所占用的 RxFIFO 的相关 RXPLVL[1:0] 信息一起保留。

位 14:13 **RXPLVL[1:0]**：RxFIFO 封装级别

当 RXWNE = 0 且数据大小设置为 16 位时，该值指定 RxFIFO 中保留的剩余数据帧数。

00：RxFIFO 中没有下一个可用的帧

01：1 个帧可用

10：2 个帧可用*

11：3 个帧可用*

注意：(*)：仅当数据大小设置为 8 位时可能的值。

当数据大小大于 16 位时，这些位始终读为 00。因此，如果数据大小设置为 17 位到 24 位，则 RWNE 和 RXPLVL 位均无法检测到在 FIFO 接收到的单个数据帧。然后，用户必须应用其他方法来检测接收到的数据量，例如当 TSIZE > 0 时监视 EOT 事件，或当 FTHLV = 0 时监视 RXP 事件。

位 12 **TXC**：TxFIFO 发送完成 (TxFIFO transmission complete)

该标志的行为取决于 TSIZE 设置。

如果 TSIZE = 0，则 TXC 专门由硬件更改，只要 TxFIFO 变为空就会置 1，总线上无任何活动。

如果 TSIZE ≠ 0，则没有特定理由监视 TXC，因为它只是复制 EOT 标志值（包括其软件清零）。EOTIE 置 1 时，TXC 会生成中断。

SPI 复位或禁止时，该标志置 1。

0：当前数据事务仍在进行中、TxFIFO 中存在数据或者正在进行最后一个帧发送。

1：最后一个 TxFIFO 或 CRC 帧发送已完成

位 11 **SUSP**：暂停状态 (suspension status)

在主模式下，只要在 CSUSP 请求执行后当前帧已完成，或者在 RxFIFO 已满后进入主器件自动挂起接收模式（SPI_CR1 寄存器的 MASRX 位置 1），SUSP 即会由硬件置 1。

当 EOTIE 置 1 时，SUSP 会生成中断。

该位必须在 SPI 禁止之前清零，这可以通过向 SPI_IFCR 的 SUSPC 位专门写入 1 来完成。

0：SPI 未挂起（主模式有效或其他模式）。

1：主模式挂起（当前帧已完成）。

位 10 保留，必须保持复位值。

位 9 **MODF**：模式故障 (mode fault)

0：无模式故障

1：检测到模式故障。

当 MODF 置 1 时，SPI_CR1 寄存器的 SPE 和 IOLOCK 位将复位，并且在 MODF 清零之前，会阻止再次将 SPE 置 1。

向 SPI_IFCR 的 MODFC 位写入 1 可将该位清零。

位 8 **TIFRE**：TI 帧格式错误 (TI frame format error)

0：无 TI 帧错误

1：检测到 TI 帧错误

向 SPI_IFCR 的 TIFREC 位写入 1 可将该位清零。

位 7 **CRCE**：CRC 错误 (CRC error)

0：无 CRC 错误

1：检测到 CRC 错误

当 SPI 重新使能或通过向 SPI_IFCR 的 CRCEC 位写入 1 时（可选），该位被清零。

位 6 **OVR**：上溢 (Overrun)

0：无上溢

1：检测到上溢

当 SPI 重新使能或通过向 SPI_IFCR 的 OVRC 位写入 1 时，该位被清除。

位 5 **UDR**：下溢

0：无下溢

1：检测到下溢

当 SPI 重新使能时该位清零，也可以通过向 SPI_IFCR 的 UDRC 位写入 1 将该位清零。

注意：在 SPI 模式下，UDR 标志仅适用于从模式。在 I2S/PCM 模式下，该标志（如果可用）适用于主模式和从模式。

位 4 **TXTF**：发送传输已填充 (transmission transfer filled)

0：TxFIFO 上传正在进行或者未启动

1：TxFIFO 上传已完成

某次传输中的所有数据包均由应用软件或 DMA 提交以用于发送时，即，TSIZE 数据量已推入到 TxFIFO 中时，TXTF 将由硬件置 1。

该位可由软件通过向 SPI_IFCR 的 TXTFC 位写入 1 来清零。

TXTFIE 位置 1 时 TXTF 标志会触发中断。

TXTF 设置可清除 TXPIE 和 DXPIE 屏蔽，因此可在计算禁止 TXP 和 DXP 中断的时间时减轻应用软件的负担。

位 3 **EOT**：传输结束 (end of transfer)

一旦完整传输完成，即当 SPI 重新使能或在 SPI 上发送和/或接收 TSIZE 数量的数据时，EOT 即会由硬件置 1。当 SPI 重新使能时该位清零，也可以通过向 SPI_IFCR 的 EOTC 位写入 1 将该位清零。

EOTIE 位置 1 时 EOT 标志会触发中断。

如果在 TXTF 标志置 1 以及 DXPIE 清零之前始终使用 DXP 标志，则可使用 EOT 一次性下载 RxFIFO 中包含的最后数据包。

0：传输正在执行或未启动

1：传输完成

在主模式下，EOT 事件会终止数据事务，并且可选择性处理 SS 输出。应用 CRC 时，EOT 事件将延伸至 CRC 帧事务。

强烈建议在下一个事务开始之前禁止然后重新使能 SPI（尽管设置未更改），以正确重启内部状态机。

位 2 **DXP**：双工数据包 (duplex packet)

0：TxFIFO 已满且/或 RxFIFO 为空

1：TxFIFO 具有用于写操作的空间，且 RxFIFO 至少包含一个用于读操作的数据包

TXP 和 RXP 标志均置 1 时（无论 SPI 为何模式），DXP 标志都将置 1。

位 1 **TXP**：可用的 Tx 数据包空间 (Tx-packet space available)

0：TxFIFO 没有足够的可用空间来容纳下一个数据包

1：TxFIFO 有足够的可用空间来容纳至少一个数据包

TXP 标志只能由硬件更改。其值取决于 FIFO 的物理大小及其阈值 (FTHLV[3:0])、数据帧大小（SPI 模式下的 DSIZE[4:0] 以及 I2S/PCM 模式下相应的 DATLEN[1:0]）以及实际的通信流。如果通过对 SPI_TXDR 执行连续写操作来存储数据包，则在 TxFIFO 中存储完整的数据包后，必须立即再次检查 TXP 标志。尽管 SPI 复位或禁止时 SPI TxFIFO 变得不可访问，但 TXP 仍将置 1。

位 0 **RXP**：接收数据包可用 (Rx-packet available)

0：RxFIFO 为空或接收到的数据包不完整

1：RxFIFO 至少包含 1 个数据包

该标志由硬件进行更改。如果使能 SPI，它将监视 RxFIFO 中当前可用的总数据量。RXP 值取决于 FIFO 阈值 (FTHLV[3:0])、数据帧大小（SPI 模式下的 DSIZE[4:0] 以及 I2S/PCM 模式下相应的 DATLEN[1:0]）以及实际的通信流。如果通过从 SPI_RXDR 执行连续读操作来读取数据包，则在从 RxFIFO 中读取完整的数据包后，必须立即再次检查 RXP 标志。

38.11.7 SPI/I2S 中断/状态标志清零寄存器 (SPI_IFCR)

SPI/I2S interrupt/status flags clear register

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	SUSPC	Res.	MODFC	TIFREC	CRCEC	OVRC	UDRC	TXTFC	EOTC	Res.	Res.	Res.
				w		w	w	w	w	w	w	w			

位 31:12 保留，必须保持复位值。

位 11 **SUSPC**：挂起标志清零 (Suspend flag clear)
向该位写入 1 可将 SPI_SR 寄存器的 SUSP 标志清零

位 10 保留，必须保持复位值。

位 9 **MODFC**：模式故障标志清零 (mode fault flag clear)
向该位写入 1 可将 SPI_SR 寄存器的 MODF 标志清零

位 8 **TIFREC**：TI 帧格式错误标志清零 (TI frame format error flag clear)
向该位写入 1 可将 SPI_SR 寄存器中的 TIFRE 标志清零

位 7 **CRCEC**：CRC 错误标志清零 (CRC error flag clear)
向该位写入 1 可将 SPI_SR 寄存器的 CRCE 标志清零

位 6 **OVRC**：上溢标志清零 (overrun flag clear)
向该位写入 1 可将 SPI_SR 寄存器中的 OVR 标志清零

位 5 **UDRC**：下溢标志清零 (underrun flag clear)
向该位写入 1 可将 SPI_SR 寄存器中的 UDR 标志清零

位 4 **TXTFC**：发送传输填充标志清零 (transmission transfer filled flag clear)
向该位写入 1 可将 SPI_SR 寄存器中的 TXTF 标志清零

位 3 **EOTC**：传输结束标志清零 (end of transfer flag clear)
向该位写入 1 可将 SPI_SR 寄存器中的 EOT 标志清零

位 2:0 保留，必须保持复位值。

38.11.8 SPI/I2S 发送数据寄存器 (SPI_TXDR)

SPI/I2S transmit data register

偏移地址：0x20

复位值：0x0000 0000

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

位 31:0 **TXDR[31:0]**：发送数据寄存器 (transmit data register)

该寄存器用作连接 TxFIFO 的接口。对其进行写入操作可访问 TxFIFO。

注意：在 SPI 模式下，数据始终右对齐。I2S 模式下的数据对齐取决于 DATLEN 和 DATFMT 设置。写入寄存器时将忽略未使用位，读取寄存器时会将未使用位读为 0。

注意：DR 可按字节进行访问（8 位访问）：在这种情况下，通过单次访问只能写入一个数据字节。

按半字进行访问（16 位访问）：在这种情况下，通过单次访问可写入 2 个数据字节或 1 个半字数据。

逐字进行访问（32 位访问）：在这种情况下，通过单次访问可写入 4 个数据字节或者 2 个半字数据或字数据。

禁止对该寄存器进行小于所配置数据大小的写访问。

38.11.9 SPI/I2S 接收数据寄存器 (SPI_RXDR)

SPI/I2S receive data register

偏移地址：0x30

复位值：0x0000 0000

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

位 31:0 **RXDR[31:0]**：接收数据寄存器 (receive data register)

该寄存器用作连接 RxFIFO 的接口。读取该寄存器即使对 RxFIFO 进行访问。

注意：在 SPI 模式下，数据始终右对齐。I2S 模式下的数据对齐取决于 DATLEN 和 DATFMT 设置。读取该寄存器时，未使用的位将读为 0。对该寄存器进行的写操作将被忽略。

注意：DR 可按字节进行访问（8 位访问）：在这种情况下，通过单次访问只能读取一个数据字节

按半字进行访问（16 位访问）：在这种情况下，通过单次访问可读取 2 个数据字节或 1 个半字数据

逐字进行访问（32 位访问）：在这种情况下，通过单次访问可读取 4 个数据字节或者 2 个半字数据或字数据。

禁止对该寄存器进行小于所配置数据大小的读访问。

38.11.10 SPI/I2S 多项式寄存器 (SPI_CRCPOLY)

SPI/I2S polynomial register

偏移地址：0x40

复位值：0x0000 0107

使能 SPI 时，该寄存器的内容受到写保护。

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

位 31:0 **CRCPOLY[31:0]**：CRC 多项式寄存器 (CRC polynomial register)

此寄存器包含用于 CRC 计算的多项式。

默认 9 位多项式设置 0x107 对应于 DSIZE 的默认 8 位设置。它与多项式字符串（字符串的最高有效位始终被隐藏）长度固定的其他 ST 产品使用的设置 0x07 兼容。

多项式的长度由存储在该寄存器中的值的最高有效位给出。必须将其设置为大于 DSIZE 的值。此外，将 DSIZE 配置为最大 32 位或 16 位大小并使能 CRC 时，还必须通过 CRCPOLY 寄存器将 CRC33_17 位置 1（以使多项式长度大于数据大小）。

注意：对于数据大小被限制为 16 位的实例，CRCPOLY[31:16] 位保留。对这些地址执行 32 位访问时，不受限制。保留的位 31-16 始终读为零，并忽略所有写操作。

38.11.11 SPI/I2S 发送器 CRC 寄存器 (SPI_TXCRC)

SPI/I2S transmitter CRC register

偏移地址：0x44

复位值：0x0000 0000

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

位 31:0 **TXCRC[31:0]**：发送器的 CRC 寄存器

使能 CRC 计算后，TXCRC[31:0] 位将包含后续发送字节在计算后所得到的 CRC 值。将 SPI_CR1 的 CRCEN 位置 1 或完整处理数据块时，将初始化 CRC 计算。CRC 通过 SPI_CRCPOLY 寄存器中编程的多项式连续计算。

计算时考虑的位数取决于 SPI_CRCPOLY 寄存器和 SPI_CFG1 寄存器的 CRCSIZE 位设置。

注意：进行通信时读取该寄存器会返回一个错误值。

注意：该位域不适用于 I2S 模式。

注意：对于数据大小限制为 16 位的实例，TXCRC[31-16] 位保留。对这些地址执行 32 位访问时，不受限制。保留的位 31-16 始终读为零，并忽略所有写操作。

注意：当软件读取该寄存器的内容时，CRCSIZE 位域的配置不予考虑。在这种情况下，不会对未使用的位应用屏蔽。

38.11.12 SPI/I2S 接收器 CRC 寄存器 (SPI_RXCRC)

SPI/I2S receiver CRC register

偏移地址：0x48

复位值：0x0000 0000

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

位 31:0 **RXCRC[31:0]**：接收器的 CRC 寄存器

使能 CRC 计算后，RXCRC[31:0] 位将包含后续接收字节在计算后所得到的 CRC 值。将 SPI_CR1 的 CRCEN 位置 1 或完整处理数据块时，将初始化 CRC 计算。CRC 通过 SPI_CRCPOLY 寄存器中编程的多项式连续计算。

计算时考虑的位数取决于 SPI_CRCPOLY 寄存器和 SPI_CFG1 寄存器的 CRCSIZE 位设置。

注意：进行通信时读取该寄存器会返回一个错误值。

该位域不适用于 I2S 模式。

对于数据大小限制为 16 位的外设实例，RXCRC[31-16] 位保留。对这些地址执行 32 位访问时，不受限制。保留的位 31-16 始终读为零，并忽略所有写操作。

注意：当软件读取该寄存器的内容时，CRCSIZE 位域的配置不予考虑。在这种情况下，不会对未使用的位应用屏蔽。

38.11.13 SPI/I2S 下溢数据寄存器 (SPI_UDRDR)

SPI/I2S underrun data register

偏移地址：0x4C

复位值：0x0000 0000

使能 SPI 时，该寄存器的内容受到写保护。

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

位 31:0 **UDRDR[31:0]**：从模式下溢时的数据 (data at slave underrun condition)

仅在从模式下和下溢条件下考虑该寄存器。所考虑的位数取决于 SPI_CFG1 寄存器的 DSIZE 位设置。下溢条件的处理取决于 SPI_CFG1 寄存器的 UDRCFG 位是否置 1。

注意：对于数据大小限制为 16 位的外设实例，UDRDR[31-16] 位保留。对这些地址执行 32 位访问时，不受限制。保留的位 31-16 始终读为零，并忽略所有写操作。

38.11.14 SPI/I2S 配置寄存器 (SPI_I2SCFGR)

SPI/I2S configuration register

偏移地址：0x50

复位值：0x0000 0000

该寄存器必须在 I2S 禁止时 (SPE = 0) 进行配置。除了 I2SMOD 位需要保持为 0 之外，在 SPI 模式下，不考虑该寄存器的内容。

该寄存器在不支持 I2S 模式的实例中保留。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV[7:0]							
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	DATFMT	WSINV	FIXCH	CKPOL	CHLEN	DATLEN[1:0]		PCMSYNC	Res.	I2SSTD[1:0]		I2SCFG[2:0]			I2SMOD
	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw

位 31:26 保留，必须保持复位值。

位 25 **MCKOE**：主时钟输出使能 (master clock output enable)

0：禁止主时钟输出

1：使能主时钟输出

位 24 **ODD**：预分频器的奇数因子 (odd factor for the prescaler)

0：实际分频值为 = I2SDIV *2

1：实际分频值为 = (I2SDIV * 2) + 1

有关详细信息，请参见*第 38.9.9 节：时钟发生器*。

位 23:16 **I2SDIV[7:0]**：I^2S 线性预分频器

当 ODD 也等于 1 时，I2SDIV 可以取除了值 1 之外的任何值。

有关详细信息，请参见*第 38.9.9 节：时钟发生器*。

位 15 保留，必须保持复位值。

位 14 **DATFMT**：数据格式 (data format)

0：SPI_RXDR 或 SPI_TXDR 内的数据右对齐

1：SPI_RXDR 或 SPI_TXDR 内的数据左对齐。

位 13 **WSINV**：字选择反转 (word select inversion)

该位用于反转 WS 信号的默认极性。

0：在 I2S Philips 标准中，左通道传输在 WS 下降沿后一个 CK 周期开始，右通道传输在 WS 上升沿后一个 CK 周期开始。

在 MSB 或 LSB 对齐模式下，WS 为高电平时传输左通道，WS 为低电平时传输右通道。

在 PCM 短帧模式下，数据传输从 WS 的下降沿开始，而在 PCM 长帧模式下，数据传输从 WS 的上升沿开始。

1：在 I2S Philips 标准中，左通道传输在 WS 上升沿后一个 CK 周期开始，右通道传输在 WS 下降沿后一个 CK 周期开始。

在 MSB 或 LSB 对齐模式下，WS 为低电平时传输左通道，WS 为高电平时传输右通道。

在 PCM 短帧模式下，数据传输从 WS 的上升沿开始，而在 PCM 长帧模式下，数据传输从 WS 的下降沿开始。

位 12 **FIXCH**：从模式下的固定通道长度 (fixed channel length in slave)

0：从模式下的通道长度不为 16 位或 32 位（CHLEN 必须置 1）

1：从模式下的通道长度应为 16 位或 32 位（取决于 CHLEN）

位 11 **CKPOL**：串行音频时钟极性 (serial audio clock polarity)

0：由 SPI/I2S 生成的信号（即，SDO 和 WS）在 CK 的下降沿发生更改，由 SPI/I2S 接收的信号（即，SDI 和 WS）在 CK 的上升沿进行读取。

1：由 SPI/I2S 生成的信号（即，SDO 和 WS）在 CK 的上升沿发生更改，由 SPI/I2S 接收的信号（即，SDI 和 WS）在 CK 的下降沿进行读取。

位 10 **CHLEN**：通道长度（每个音频通道的位数）(channel length (number of bits per audio channel))

0：16 位

1：32 位

位 9:8 **DATLEN[1:0]**：要传输的数据长度 (data length to be transferred)

00：16 位数据长度

01：24 位数据长度

10：32 位数据长度

11：不允许

注意：并非始终支持 24 位和 32 位数据宽度（DATLEN = 01 或 10），请参见第 38.3 节：SPI 实现确认支持的数据大小。

位 7 **PCMSYNC**：PCM 帧同步 (PCM frame synchronization)

0：短帧同步

1：长帧同步

位 6 保留，必须保持复位值。

位 5:4 **I2SSTD[1:0]**：I^2S 标准选择

00：I^2S Philips 标准

01：MSB 对齐标准（左对齐）

10：LSB 对齐标准（右对齐）

11：PCM 标准

有关 I^2S 标准的详细信息，请参见*第 38.9.5 节：支持的音频协议*

位 3:1 **I2SCFG[2:0]**：I2S 配置模式 (I2S configuration mode)

000：从模式——发送

001：从模式——接收

010：主模式——发送

011：主模式——接收

100：从模式——全双工

101：主模式——全双工

其他，未使用

位 0 **I2SMOD**：I2S 模式选择 (I2S mode selection)

0：选择 SPI 模式

1：选择 I2S/PCM 模式

38.11.15 SPI/I2S 寄存器映射

表 315. SPI 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	SPI_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IOLOCK	TCRCINI	RCRCINI	CRC33_17	SSI	HDDIR	CSUSP	CSTART	MASRX	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPE
	Reset value																0	0	0	0	0	0	0	0	0								0

表 315. SPI 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x04	SPI_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSIZE[15:0]															
	Reset value																	0(1)	0(1)	0(1)	0(1)	0(1)	0(1)	0	0	0	0	0	0	0	0	0	0
0x08	SPI_CFG1	BPASS	MBR[2:0]			Res.	Res.	Res.	Res.	Res.	CRCEN	Res.	CRCSIZE[4:0]					TXDMAEN	RXDMAEN	Res.	Res.	Res.	Res.	UDRCFG	FTHLV[3:0]				DSIZE[4:0]				
	Reset value	0	0	0	0						0		1	1	1	1	1	0	0					0	0(1)	0(1)	0	0	0	0(1)	1(1)	1(1)	1(1)
0x0C	SPI_CFG2	AFCNTR	SSOM	SSOE	SSIOP	Res.	SSM	CPOL	CPHA	LSBFRST	MASTER	SP[2:0]			COMM [1:0]		Res.	IOSWP	RDIOP	RDIOM	Res.	Res.	Res.	Res.	Res.	MIDI[3:0]				MSSI[3:0]			
	Reset value	0	0	0	0		0	0	0	0	0	0	0	0	0	0		0	0	0						0	0	0	0	0	0	0	0
0x10	SPI_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MODFIE	TIFREIE	CRCEIE	OVRIE	UDRIE	TXTFIE	EOTIE	DXPIE	TXPIE	RXPIE
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x14	SPI_SR	CTSIZE[15:0](1)																RXWNE	RXPLVL [1:0]		TXC	SUSP	Res.	MODF	TIFRE	CRCE	OVR	UDR	TXTF	EOT	DPXP	TXP	RXP
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0		0	0	0	0	0	0	0	0	1	0
0x18	SPI_IFCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUSPC	Res.	MODFC	TIFREC	CRCEC	OVRC	UDRC	TXTFC	EOTC	Res.	Res.	Res.
	Reset value																					0		0	0	0	0	0	0	0			
0x1C	Reserved	Reserved																															
0x20	SPI_TXDR	TXDR[31:16]																TXDR[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24-0x2C	Reserved	Reserved																															
0x30	SPI_RXDR	RXDR[31:16]																RXDR[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34 - 0x3C	Reserved	Reserved																															
0x40	SPI_CRCPOLY	CRCPOLY[31:16](2)																CRCPOLY[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1
0x44	SPI_TXCRC	TXCRC[31:16](2)																TXCRC[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	SPI_RXCRC	RXCRC[31:16](2)																RXCRC[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	SPI_UDRDR	UDRDR[31:16](2)																UDRDR[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	SPI_I2SCFGR	Res.	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV[7:0]								Res.	Res.	WSINV	FIXCH	CKPOL	CHLEN	DATLEN[1:0]		PCMSYNC	Res.	I2SSTD[1:0]		I2SCFG[2:0]			I2SMOD
	Reset value							0	0	0	0	0	0	0	0	0	0			0	0	0	0	0	0	0		0	0	0	0	0	0

1. 对于有限功能集实例，该位域保留，并且必须保持复位值。有关详细信息，请参见*第 38.11 节：SPI/I2S 寄存器*中的具体寄存器说明。
2. 对于数据大小被限制为 16 位的外设实例，位 31-16 保留。对这些地址执行 32 位访问时，不受限制。位 31-16 在保留的情况下始终读为零，并忽略对这些位进行的所有写操作。

有关寄存器边界地址的信息，请参见*第 2.2 节*。

39 FD 控制器局域网络 (FDCAN)

39.1 简介

控制器局域网络 (CAN) 子系统（请参见*图 500*）由一个 CAN 模块、一个共享消息 RAM 和一个配置模块组成。关于这些组成部分的基地址，请参见存储器映射。

FDCAN 模块符合 ISO 11898-1: 2015（CAN 协议规范第 2.0 版 A、B 部分）和 CAN FD 协议规范第 1.0 版。

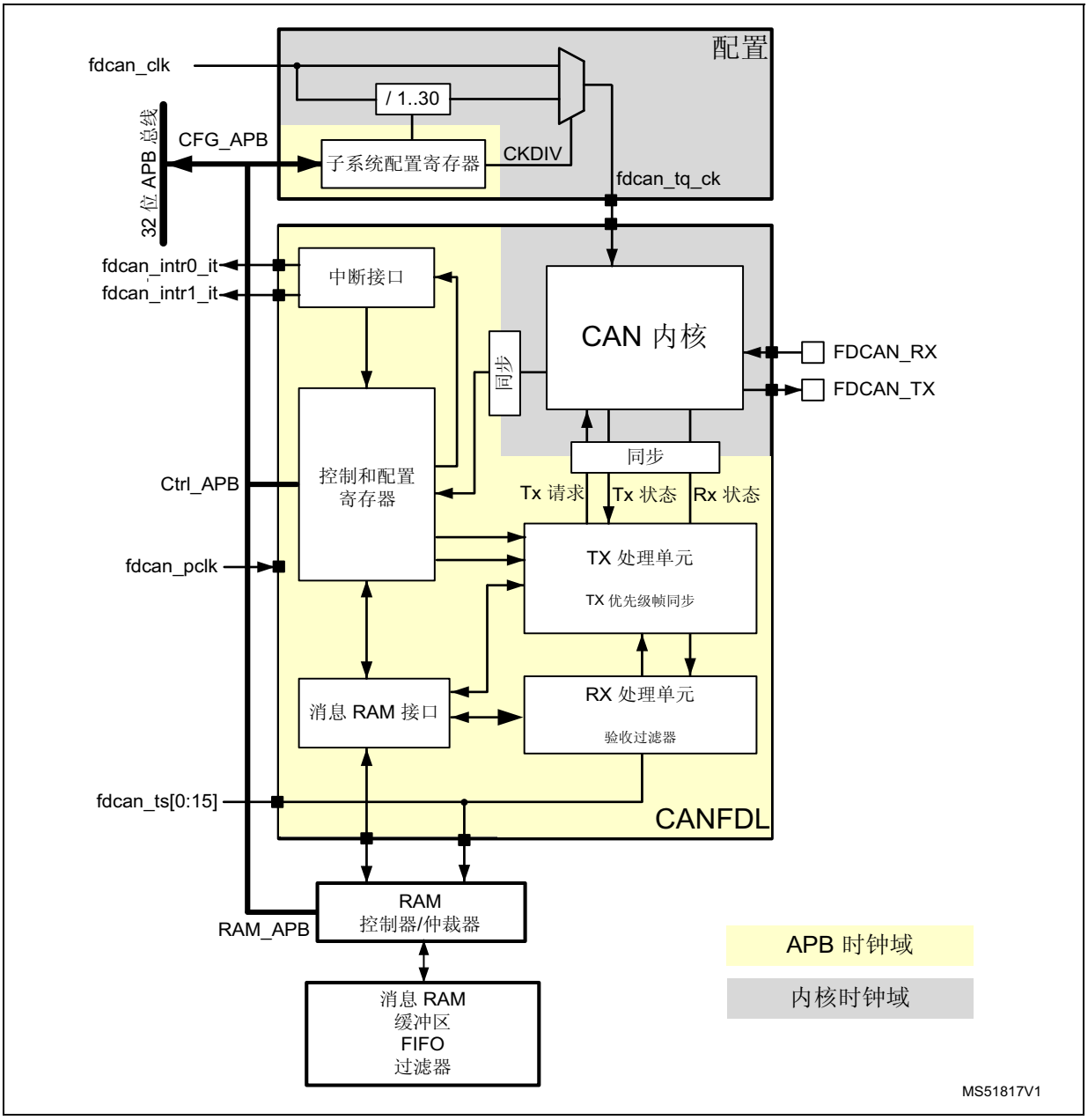
每个 FDCAN 实例有 0.8 KB 的消息 RAM，可实现过滤器、接收 FIFO、发送事件 FIFO 和发送 FIFO。

表 316 和*图 500* 分别对 CAN 子系统 I/O 信号和引脚进行了详细介绍。

表 316. CAN 子系统 I/O 信号

名称	类型	说明
fdcan_ck	数字输入	CAN 子系统内核时钟输入
fdcan_pclk		CAN 子系统 APB 接口时钟输入
fdcan_intr0_it	数字输出	FDCAN 中断 0
fdcan_intr1_it		FDCAN 中断 1
fdcan_ts[0:15]	-	外部时间戳向量
FDCAN_RX	数字输入	FDCAN 接收引脚
FDCAN_TX	数字输出	FDCAN 发送引脚
APB 接口	数字输入/输出	一个 APB 具有多个 psel，分别用于配置、控制和 RAM 访问

图 500. CAN 子系统

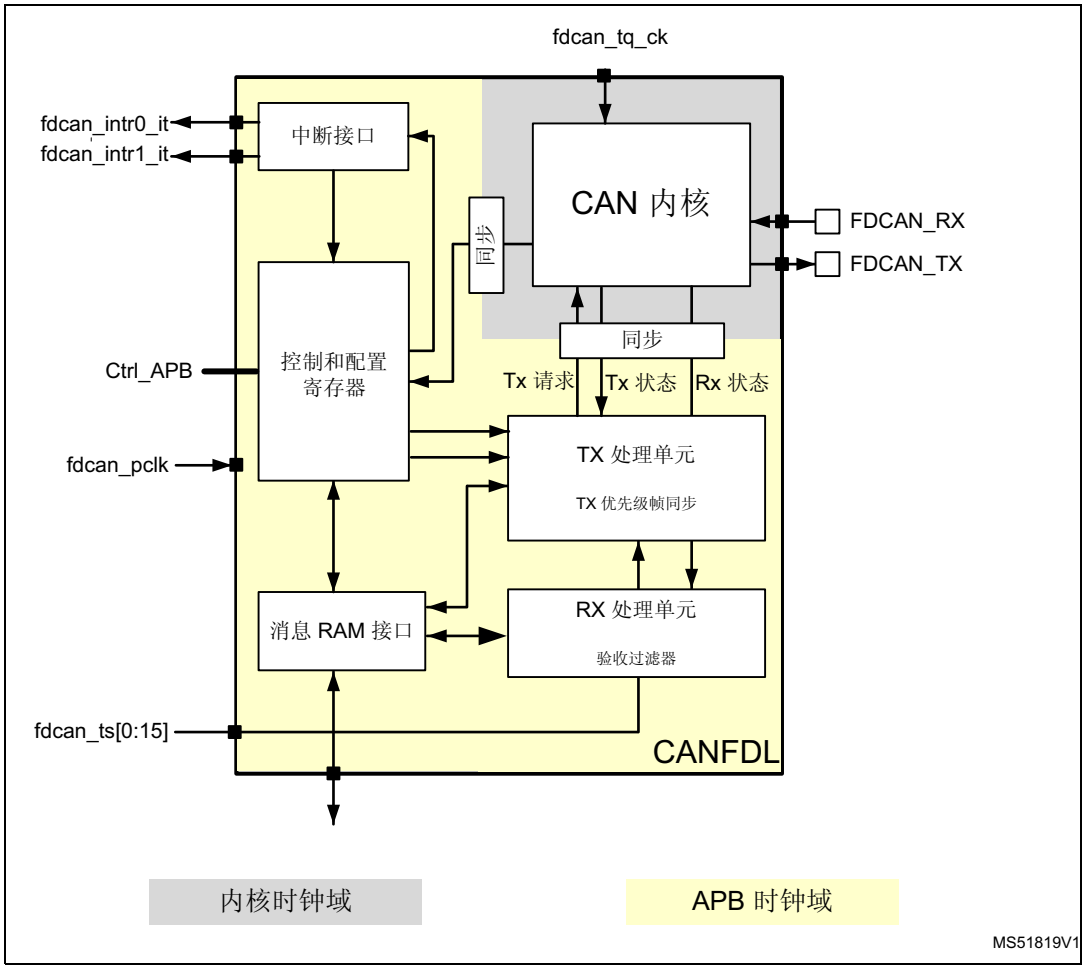

配置
fdcan_clk
/ 1..30
32 位 APB 总线
CFG_APB
子系统配置寄存器
CKDIV
fdcan_tq_ck
fdcan_intr0_it
fdcan_intr1_it
中断接口
CAN 内核
同步
FDCAN_RX
FDCAN_TX
控制和配置
寄存器
同步
Ctrl_APB
Tx 请求
Tx 状态
Rx 状态
fdcan_pclk
TX 处理单元
TX 优先级帧同步
消息 RAM 接口
RX 处理单元
验收过滤器
fdcan_ts[0:15]
CANFDL
RAM
控制器/仲裁器
RAM_APB
消息 RAM
缓冲区
FIFO
过滤器
APB 时钟域
内核时钟域
MS51817V1

39.2 FDCAN 主要特性

- 符合 CAN 协议第 2.0 版 A、B 部分和 ISO 11898-1: 2015, -4
- CAN FD 最大支持 64 个数据字节
- CAN 错误记录
- 支持 AUTOSAR 和 J1939
- 改进了接收过滤
- 两个接收 FIFO，分别可容纳三个有效负载（每个有效负载最多 64 字节）
- 接收到高优先级消息时单独发出信号指示
- 可配置的发送 FIFO/队列，可容纳三个有效负载（每个有效负载最多 64 字节）
- 发送事件 FIFO
- 可编程环回测试模式
- 可屏蔽模块中断
- 两个时钟域：APB 总线接口和 CAN 内核时钟
- 支持掉电

39.3 FDCAN 功能说明

图 501. FDCAN 框图



双中断线

FDCAN 外设提供 fdcan_intr0_it 和 fdcan_intr1_it 两条中断线。

通过对 FDCAN_ILE 寄存器中的 EINT0 和 EINT1 位进行编程，可分别使能或禁止这两条中断线。

CAN 内核

CAN 内核包含协议控制器和接收/发送移位寄存器。它可处理所有 ISO 11898-1: 2015 协议功能，并支持 11 位和 29 位标识符。

同步

同步模块将来自 APB 时钟域的信号同步到 CAN 内核时钟域，以及将来自 CAN 时钟域的信号同步到 APB 内核时钟域。

发送处理单元

该处理单元控制从消息 RAM 到 CAN 内核的消息传输。最多可提供三个发送缓冲区进行发送。发送缓冲区可用作发送 FIFO 或发送队列。发送事件 FIFO 会将发送时间戳与相应的消息 ID 存储在一起，另外还支持取消发送。

接收处理单元

该处理单元控制着从 CAN 内核接收的消息到外部消息 RAM 的传输。接收处理单元支持两个接收 FIFO，用于存储所有通过验收过滤的消息。每条消息均与其接收时间戳存储在一起。对于 11 位 ID，最多可定义 28 个过滤器；对于 29 位 ID，最多可定义 8 个过滤器。

APB 接口

该接口可将 FDCAN 连接到 APB 总线，用于配置寄存器、控制器配置和 RAM 访问。

消息 RAM 接口

通过 RAM 控制器/仲裁器将 FDCAN 访问连接到外部 1 KB 消息 RAM。

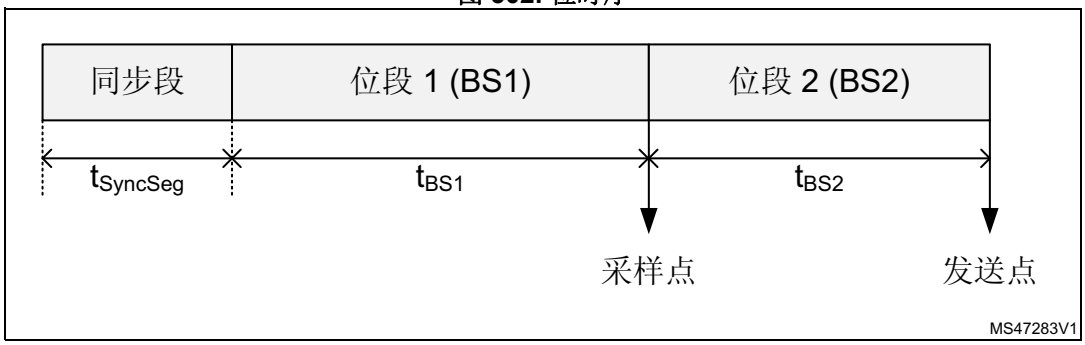
39.3.1 位时序

位时序逻辑将监视串行总线，执行采样并调整采样点，在调整采样点时，需要在起始位边沿进行同步并在后续的边沿进行再同步。

如*图 502* 中所示，通过将位时间划分为以下三段，即可解释其工作过程：

- 同步段（SYNC_SEG）：位变化应该在此时间段内发生，具体为一个时间片的固定长度 (1 x tq)。
- 位段 1 (BS1)：定义采样点的位置。它包括 CAN 标准的 PROP_SEG 和 PHASE_SEG1。其持续长度可以在 1 到 16 个时间片之间调整，但也可以自动加长，以补偿不同网络节点的频率差异所导致的正相位漂移。
- 位段 2 (BS2)：定义发送点的位置。它代表 CAN 标准的 PHASE_SEG2，其持续长度可以在 1 到 8 个时间片之间调整，但也可以自动缩短，以补偿负相位漂移。

图 502. 位时序



波特率是位时间的倒数（波特率 = 1 / 位时间），而位时间又是三个分量的总和。如*图 502* 中所示，位时间 = $t_{SyncSeg}$ + t_{BS1} + t_{BS2}，其中：

- 对于标称位时间
  - tq = (FDCAN_NBTP.NBRP[8:0] + 1) * $t_{fdcan_tq_clk}$
  - $t_{SyncSeg}$ = 1 tq
  - t_{BS1} = tq * (FDCAN_NBTP.NTSEG1[7:0] + 1)
  - t_{BS2} = tq * (FDCAN_NBTP.NTSEG2[6:0] + 1)
- 对于数据位时间
  - tq = (FDCAN_DBTP.DBRP[4:0] + 1) * $t_{fdcan_tq_clk}$
  - $t_{SyncSeg}$ = 1 tq
  - t_{BS1} = tq * (FDCAN_DBTP.DTSEG1[4:0] + 1)
  - t_{BS2} = tq * (FDCAN_DBTP.DTSEG2[3:0] + 1)

（再）同步跳转宽度 (SJW) 定义位段加长或缩短的上限。它可以在 1 到 4 个时间片之间调整。

有效边沿是指一个位时间内总线电平从显性到隐性的第一次转换（前提是控制器本身不发送隐性位）。

如果在 BS1 而不是 SYNC_SEG 中检测到有效边沿，则 BS1 会延长最多 SJW，以便延迟采样点。

相反地，如果在 BS2 而不是 SYNC_SEG 中检测到有效边沿，则 BS2 会缩短最多 SJW，以便提前发送点。

为了避免编程错误，位时序寄存器只能在器件处于待机模式时进行配置。寄存器 FDCAN_DBTP 和 FDCAN_NBTP（分别专用于数据位时序和标称位时序）只能在 CCCR.CCE 和 CCCR.INIT 置 1 时进行访问。

注意： *有关 CAN 位时序和再同步机制的详细说明，请参见 ISO 11898-1 标准。*

39.3.2 工作模式

配置

访问 IP 版本、硬件和输入时钟分频器配置。当时钟分频器设置为 0 时，按原样使用主输入时钟。

软件初始化

要进行软件初始化，首先应通过软件或硬件复位、或者进入 Bus_Off 状态将 FDCAN_CCCR 寄存器中的 INIT 位置 1。当 FDCAN_CCCR 寄存器中的 INIT 位置 1 时，从 CAN 总线传入和传出的消息都将停止，并且 CAN 总线输出 FDCAN_TX 的状态为隐性（高）。EML（错误管理逻辑）计数器保持不变。将 FDCAN_CCCR 中的 INIT 位置 1 不会更改任何配置寄存器。将 FDCAN_CCCR 中的 INIT 位清零会结束软件初始化。随后，位流处理单元 (BSP) 会等待出现 11 个连续隐性位 (Bus_Idle) 序列，以此将其自身与 CAN 总线上的数据传输进行同步，然后才能参与总线活动并开始消息传输。

仅当 FDCAN_CCCR 寄存器中的 INIT 位以及 FDCAN_CCCR 寄存器中的 CCE 位均置 1 时，才能访问 FDCAN 配置寄存器。

FDCAN_CCCR 寄存器中的 CCE 位只能在 FDCAN_CCCR 中的 INIT 位置 1 时才能置 1/清零。FDCAN_CCCR 寄存器中的 CCE 位会在 FDCAN_CCCR 中的 INIT 位清零时自动清零。

FDCAN_CCCR 寄存器中的 CCE 位置 1 后，以下寄存器会复位：

- FDCAN_HPMS——高优先级消息状态
- FDCAN_RXF0S——接收 FIFO 0 状态
- FDCAN_RXF1S——接收 FIFO 1 状态
- FDCAN_TXFQS——发送 FIFO/队列状态
- FDCAN_TXBRP——发送缓冲区请求挂起
- FDCAN_TXBTO——发送缓冲区发送已发生
- FDCAN_TXBCF——发送缓冲区取消完成
- FDCAN_TXEFS——发送事件 FIFO 状态

FDCAN_CCCR 中的 CCE 位置 1 后，FDCAN_TOCV 寄存器中的超时计数器值 TOC 位会预设为由 FDCAN_TOCC 寄存器中的 TOP 位配置的值。

此外，FDCAN_CCCR 中的 CCE 位置 1 时，发送处理单元和接收处理单元中的状态机会保持空闲状态。

仅当 FDCAN_CCCR 寄存器中的 CCE 位清零时，才能写入以下寄存器：

- TXBAR——发送缓冲区添加请求
- TXBCR——发送缓冲区取消请求

仅当 CCCR 寄存器中的 INIT 位和 CCE 位均已置 1 时，才能通过软件将 FDCAN_CCCR 中的 TEST 位和 MON 位置 1。这两个位可随时复位。仅当 FDCAN_CCCR 中的 INIT 位和 CCE 位均已置 1 时，才能将 FDCAN_CCCR 中的 DAR 位置 1/清零。

正常工作

FDCAN 在硬件复位后的默认工作模式是事件驱动 CAN 通信。不支持 TT 工作模式。

FDCAN 完成初始化且 FDCAN_CCCR 寄存器中的 INIT 位清零后，FDCAN 会将其自身与 CAN 同步并准备好进行通信。

通过接收滤波后，接收到的包含消息 ID 和 DLC 的消息会存储到接收 FIFO 0 或接收 FIFO 1 中。

要发送消息，可初始化或更新发送 FIFO 或发送队列。不支持在接收到远程帧后自动发送。

CAN FD 操作

FDCAN 协议有两种变体：

1. 长帧模式 (LFM)，在该模式下，CAN 帧的数据字段可以超过八个字节
2. 快速帧模式 (FFM)，在该模式下，CAN 帧的控制域、数据域和 CRC 域的发送比特率要高于帧开始和结束的发送比特率

快速帧模式可与长帧模式结合使用。

具有 11 位标识符的 CAN 帧中之前保留的位和具有 29 位标识符的 CAN 帧中之前保留的第一个位解码为 FDF 位：FDF 隐性表示 CAN FD 帧，而 FDF 显性则表示经典 CAN 帧。

在 CAN FD 帧中，FDF 之后的两个位 res 和 BRS 决定是否切换这一 CAN FD 帧中的比特率。CAN FD 比特率切换通过 res 显性和 BRS 隐性指示。res 隐性的编码保留，供将来协议扩展使用。如果 FDCAN接收到的帧为 FDF 隐性和 res 隐性，则会通过将 PSR.PXE 位置 1 的方式发出协议异常事件。如果使能协议异常处理 (CCCR.PXHD = 0)，下一采样点的工作状态会从接收器 (PSR.ACT = 10) 变为同步中 (PSR.ACT = 00)。如果禁止协议异常处理 (CCCR.PXHD = 1)，FDCAN 会将隐性 res 位当作格式错误处理，并将以错误帧进行响应。

CAN FD 操作通过编程 CCCR.FDOE 来使能。如果 CCCR.FDOE = 1，则会使能 CAN FD 帧的发送和接收。始终可进行经典 CAN 帧的发送和接收。是否可发送 FD CAN 帧或经典 CAN 帧可通过相应发送缓冲区元素中的 FDF 位配置。如果 CCCR.FDOE = 0，接收到的帧会被当作经典 CAN 帧，从而会在接收到 CAN FD 帧时发送错误帧。如果 CAN FD 操作已禁止，即使发送缓冲区元素的 FDF 位置 1，也不会发送 CAN FD 帧。仅当 CCCR.INIT 和 CCCR.CCE 均置 1 时，才能更改 CCCR.FDOE 和 CCCR.BRSE。

如果 CCCR.FDOE = 0，则会忽略 FDF 和 BRS 位的设置，并会以经典 CAN 格式发送帧。如果 CCCR.FDOE = 1 且 CCCR.BRSE = 0，则仅会评估发送缓冲区元素的 FDF 位。如果 CCCR.FDOE = 1 且 CCCR.BRSE = 1，则会使能比特率切换发送 CAN FD 帧。所有 FDF 和 BRS 位置 1 的发送缓冲区元素均会通过比特率切换以 CAN FD 格式发送。

建议仅当满足以下条件时在 CAN 操作期间切换模式：

- CAN FD 数据阶段的故障率显著高于 CAN FD 仲裁阶段的故障率。在这种情况下，应禁止发送的 CAN FD 比特率切换选项。
- 系统启动期间，所有节点在被验证能够以 CAN FD 格式进行通信之前，都将发送经典 CAN 消息。如果通过验证，所有节点会切换为 CAN FD 操作。
- CAN 局部网络中的唤醒消息必须以经典 CAN 格式进行发送。
- 并非所有节点均能进行 CAN FD 操作时采用行结束编程。非 CAN FD 节点在编程完成之前会一直保持静默模式。随后，所有节点会切换回经典 CAN 通信。

在 FDCAN 格式中，DLC 的编码与其中一种标准 CAN 格式不同。DLC 代码 0 到 8 的编码与标准 CAN 相同，代码 9 到 15（在标准 CAN 中，所有代码的数据字段均为 8 个字节）是按照*表 317* 进行编码的。

表 317. FDCAN 中的 DLC 编码

DLC	9	10	11	12	13	14	15
数据字节数	12	16	20	24	32	48	64

在 CAN FD 快速帧中，如果该位为隐性，则在 BRS（比特率切换）位后，位时长将在帧内切换。在 BRS 位之前，在 FDCAN 仲裁阶段，标准 CAN 位时长按照位时长和预分频器寄存器 BTP 定义的方式使用。在接下来的 FDCAN 数据阶段，快速 CAN 位时长按照快速位时长和预分频器寄存器 FBTP 定义的方式使用。位时长会在出现 CRC 分隔符或检测到错误时从快速时长切换回来，以先发生的事件为准。

CAN FD 数据阶段的最大可配置比特率取决于 FDCAN 内核时钟频率。举例来说，如果 FDCAN 内核时钟频率为 20 MHz，最短可配置位时间为四个时间片 (tq)，则数据阶段的比特率为 5 Mb/s。

在 CAN FD 长帧和 CAN FD 快速帧这两种数据帧格式中，位 ESI（错误状态指示符）的值是由发送开始时的发送器错误状态决定的。如果发送器为错误被动状态，则 ESI 会隐性传送，否则会显性传送。在 CAN FD 远程帧中，ESI 位始终显性传送，与发送器错误状态无关。CAN FD 远程帧的数据长度代码以 0 的形式发送。

当 DLC > 8 的 FDCAN 发送配置 FDCAN 发送缓冲区时，前 8 个字节会按照发送缓冲区中的配置发送，而数据字段的剩余部分会用 0xCC 填充。当 FDCAN 接收到 DLC > 8 的 FDCAN 帧时，该帧的前 8 个字节会存储到匹配的接收 FIFO 中。剩余字节会被丢弃。

收发器延迟补偿

在 FDCAN 发送的数据阶段，只有一个节点会进行发送，所有其他节点都是接收器。总线的长度没有影响。通过引脚 FDCAN_TX 进行发送时，协议控制器会通过引脚 FDCAN_RX 接收到其本地 CAN 收发器发送的数据。接收数据的延迟为 CAN 收发器环路延迟。如果该延迟

大于 TSEG1（采样点之前的时间段），则会检测到位错误。如果不进行收发器延迟补偿，FDCAN 帧数据阶段的比特率会受到收发器环路延迟的限制。

FDCAN 会实施延迟补偿机制来补偿 CAN 收发器环路延迟，从而可在 FDCAN 数据阶段以较高的比特率进行发送，而不受特定 CAN 收发器延迟的影响。

为了检查发送节点的数据阶段是否存在位错误，会将延迟发送数据与在第二采样点 (SSP) 接收到的数据进行比较。如果检测到位错误，发送器将在下一常规采样点对这一位错误作出响应。在仲裁阶段，始终会禁止延迟补偿。

发送器延迟补偿会使能数据位时间短于发送器延迟的配置，新版 ISO11898-1 中进行了详细介绍。发送器延迟补偿是通过将 DBTP.TDC 位置 1 使能的。

接收到的位会与在 SSP 发送的位进行比较。SSP 位置定义为从 FDCAN 发送输出引脚 FDCAN_TX 通过收发器到接收输入引脚 FDCAN_RX 测得的延迟与由 TDCR.TDCO 配置的发送器延迟补偿偏移之和。发送器延迟补偿偏移用于调整已接收位中 SSP 的位置（例如数据阶段的半个位时间）。第二采样点的位置会向下舍入到下一 mtq 整数（最小时间片，即一个 fdcan_tq_ck 时钟周期）。

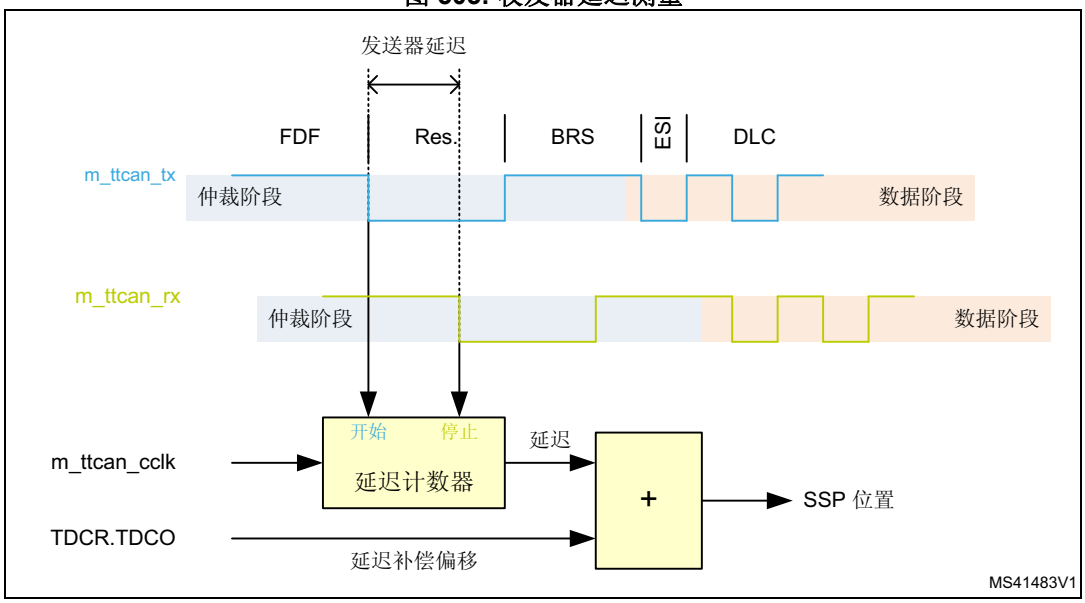
PSR.TDCV 显示了实际发送器延迟补偿值。CCCR.INIT 置 1 时，PSR.TDCV 会清零；并且如果 DBTP.TDC 置 1，每次发送 FD 帧时，PSR.TDCV 都会更新。

对于在 FDCAN 中实施的发送器延迟补偿，必须考虑以下边界条件：

- 测得的 FDCAN_Tx 到 FDCAN_RX 的延迟与配置的发送器延迟补偿偏移 TDCR.TDCO 之和必须小于数据阶段的 6 个位时间。
- 测得的 FDCAN_TX 到 FDCAN_RX 的延迟与配置的发送器延迟补偿偏移 TDCR.TDCO 之和必须小于或等于 127 mtq。如果二者之和超过该值，则为发送器延迟补偿使用最大值 (127 mtq)。
- 数据阶段在 CRC 分隔符的采样点结束，此时会停止检查在 SSP 接收到的位。

如果通过编程 DBTP.TDC = 1 使能发送器延迟补偿，则会在每个 FDF 位到 res 位的下降沿处发送的 CAN FD 帧中开始测量。在发送器的接收输入引脚 FDCAN_RX 上观察到该边沿时，会停止测量。该测量的分辨率为一个 mtq。

图 503. 收发器延迟测量



为了避免接收到的 FDF 位中的显性毛刺信号导致在接收到的 res 位下降沿之前结束延迟补偿测量（造成 SSP 位置过早），可通过编程 TDCR.TDCF 来使能发送器延迟补偿过滤器窗口，从而定义最小 SSP 位置值。进行发送器延迟测量时，会忽略可能导致 SSP 位置更加提前的 FDCAN_RX 上的显性边沿。当 SSP 位置至少为 TDCR.TDCF 设定值且 FDCAN_RX 为低电平时，会停止测量。

受限工作模式

在受限工作模式下，节点能够接收数据帧和远程帧，并能对有效帧进行确认，但不会发送数据帧、远程帧、有效错误帧或过载帧。如果存在错误条件或过载条件，则不会发送显性位，而会等待出现总线空闲条件，以便使自身与 CAN 通信重新同步。错误记录 (ECR.CEL) 激活时，错误计数器 (ECR.REC, ECR.TEC) 会冻结。软件可通过将 CCCR.ASM 位置 1 的方式将 FDCAN 设置为进入受限工作模式。仅当 CCCR.CCE 和 CCCR.INIT 均设为 1 时，才能通过软件将该位置 1。该位可随时通过软件清零。

当发送处理单元无法及时从消息 RAM 中读取数据时，会自动进入受限工作模式。要退出受限工作模式，软件必须复位 CCCR.ASM。

受限工作模式可用于根据不同 CAN 比特率对自身进行调整的应用中。在这种情况下，应用会测试不同比特率，并在收到有效帧后退出受限工作模式。

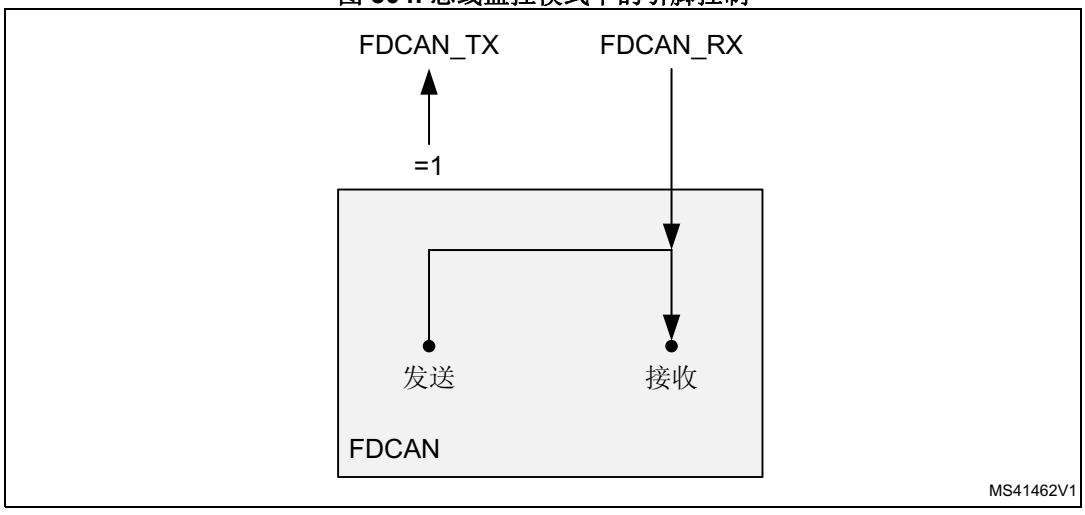
注意： *受限工作模式不得与环回模式（内部或外部）结合使用。*

总线监控模式

将 CCCR.MON 位置 1 可以将 FDCAN 设置为总线监控模式。在总线监控模式下（更多详细信息，请参见 ISO11898-1，10.12 总线监控），FDCAN 能够接收有效数据帧和有效远程帧，但不能启动发送过程。在该模式下，FDCAN 仅会在 CAN 总线上发送隐性位，如果 FDCAN 必须发送一个显性位（ACK 位、过载标志、活动错误标志），该位将在内部被改道发送，以便 FDCAN 可以监视该位，即使 CAN 总线保持隐性状态。在总线监控模式下，TXBRP 寄存器会保持复位状态。

总线监控模式可用于分析 CAN 总线上的流量，同时又不会因发送显性位对其造成影响。*图 504* 显示了总线监控模式下 FDCAN_TX 和 FDCAN_RX 信号与 FDCAN 的连接。

图 504. 总线监控模式下的引脚控制



禁止自动重发送 (DAR) 模式

根据 CAN 规范（请参见 ISO11898-1，6.3.3 恢复管理），FDCAN 提供的方法可自动重发已丢失仲裁的帧或在发送期间受到错误干扰的帧。默认使能自动重发送。

DAR 模式下的帧发送

在 DAR 模式下，所有 CAN 总线启动的发送都会自动取消。发送缓冲区发送请求挂起位 TXBRP.TRPx 会在成功发送后复位，如果取消时发送尚未开始、发送已因仲裁丢失而中止、或者帧发送期间出错，该位也会复位。

- 成功发送
  - 相应的发送缓冲区发送已发生位 TXBTO[TOx] 置 1
  - 相应的发送缓冲区取消完成位 TXBCF[CFx] 不置 1
- 即使已取消发送也会成功进行发送
  - 相应的发送缓冲区发送已发生位 TXBTO[TOx] 置 1
  - 相应的发送缓冲区取消完成位 TXBCF[CFx] 置 1
- 仲裁丢失或帧发送受到干扰
  - 相应的发送缓冲区发送已发生位 TXBTO[TOx] 不置 1
  - 相应的发送缓冲区取消完成位 TXBCF[CFx] 置 1

对于帧发送成功的情况，如果使能了发送事件存储，则会写入发送事件 FIFO 元素，写入的事件类型 ET = 10（即使已取消发送也会进行发送）。

掉电（休眠模式）

FDCAN 可通过 CC 控制寄存器 CCCR[CSR] 设为由时钟停止请求输入控制的掉电模式。只要时钟停止请求生效，CCCR[CSR] 位就读为 1。

所有挂起发送请求完成后，FDCAN 会一直等待至检测到总线空闲状态。检测到总线空闲状态之后，FDCAN 会置 1，随后，CCCR[INIT] 会置 1，以免继续进行 CAN 传输。此时，FDCAN 会通过将 CCCR[CSA] 置 1 确认其准备好进入掉电状态。在该模式下，时钟关闭之前，可继续访问寄存器。对 CCCR[INIT] 的写访问不起作用。此时，可以关闭模块时钟输入。

退出掉电模式之前，应用必须在复位 CC 控制寄存器标志 CCCR.CSR 之前开启模块时钟。FDCAN 将通过复位 CCCR[CSA] 确认时钟开启。之后，应用可通过复位 CCCR[INIT] 位重新开始 CAN 通信。

测试模式

要使能对 *FDCAN 测试寄存器 (FDCAN_TEST)* 的写访问，必须将 CCCR.TEST 位置 1，从而能够配置测试模式和功能。

可通过编程 TEST.TX 将四种输出功能用于 CAN 发送引脚 FDCAN_TX。除了默认功能（串行数据输出）之外，还可驱动 CAN 采样点信号来监控 FDCAN 位时长，并可驱动常量显性或隐性值。引脚 FDCAN_RX 的实际值可从 TEST.RX 读取。这两种功能均可用于检查 CAN 总线物理层。

由于 CAN 内核时钟与 APB 时钟域之间的同步机制，在 FDCAN_TX 输出引脚上显示新配置之前，对 TEST.TX 的写操作之间可能有几个 APB 时钟周期的延迟。这一点还适用于通过 TEST.RX 读取 FDCAN_RX 输入引脚的情况。

注意： *测试模式必须仅限用于生产测试和自检。对 FDCAN_TX 引脚进行软件控制会干扰所有 CAN 协议功能。不建议为应用使用测试模式。*

外部环回模式

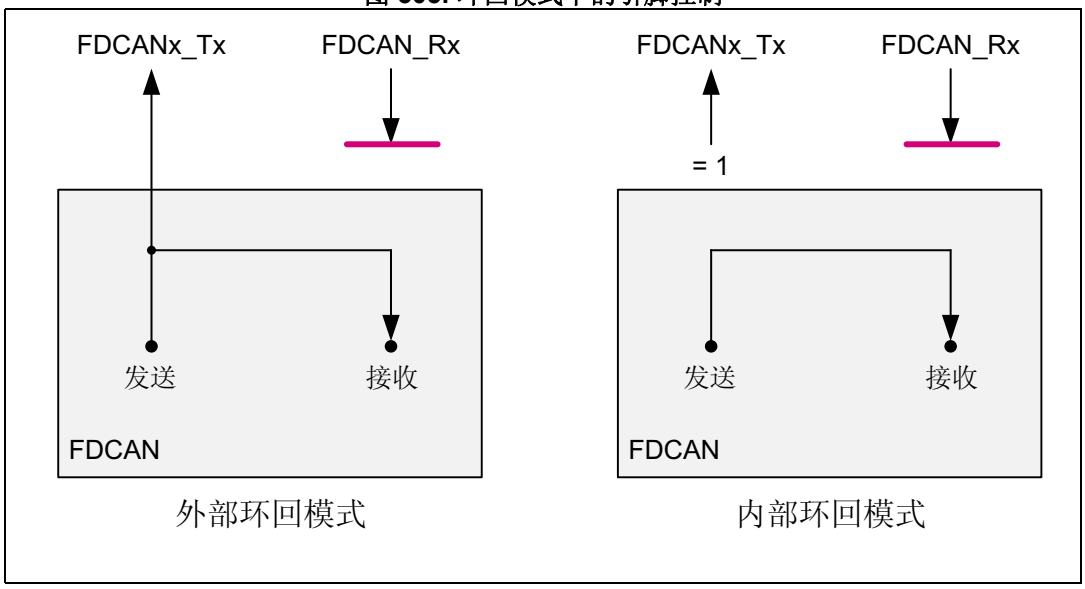
在外部环回模式下，可通过将 TEST.LBCK 编程为 1 来将 FDCAN 置 1。在环回模式下，FDCAN 将其自身发送的消息作为接收的消息来处理，并将消息存储（如果这些消息通过了验收过滤）在接收 FIFO 中。*图 505* 显示了外部环回模式下发送和接收信号 FDCAN_TX 和 FDCAN_RX 与 FDCAN 的连接。

该模式为硬件自检提供。为了不受外部仿真的影响，FDCAN 在环回模式下将忽略确认错误（在数据/远程帧的确认时隙对隐性位采样）。在此模式下，FDCAN 将执行从发送输出到其接收输入的内部反馈。FDCAN 将忽略 FDCAN_RX 输入引脚的实际值。从 FDCAN_TX 发送引脚可以监视发送的消息。

内部环回模式

将 TEST.LBCK 和 CCCR.MON 位编程为 1 会进入内部环回模式。该模式可用于“热自检”，也就是说，FDCAN 可以进行检测，同时又不会影响与 FDCAN_TX 和 FDCAN_RX 引脚相连接的运行中的 CAN 系统。在此模式下，FDCAN_RX 引脚与 FDCAN 断开连接，FDCAN_TX 引脚则保持隐性。*图 505* 显示了内部环回模式下 FDCAN_TX 和 FDCAN_RX 引脚与 FDCAN 的连接。

图 505. 环回模式下的引脚控制



时间戳生成

FDCAN 提供一个 16 位回卷计数器来生成时间戳。预分频器 TSCC.TCP 可配置为以 CAN 位时间的倍数 (1…16) 为计数器提供时钟。计数器值可通过 TSCV[TCV] 读取。对寄存器 TSCV 进行写访问会将计数器复位为 0。时间戳计数器回卷时，中断标志 IR[TSW] 会置 1。

开始接收/发送帧时，会捕获计数器值，并将该值存储在接收 FIFO (RXTS[15:0]) 或发送事件 FIFO (TXTS[15:0]) 元素的时间戳部分。

通过编程 TSCC.TSS 位，可以使用 16 位时间戳。

调试模式行为

在调试模式下，进行读访问时，置 1/复位功能在调试器寄存器访问期间自动禁止，在正常 MCU 操作期间自动使能

超时计数器

FDCAN 提供 16 位超时计数器来指示接收 FIFO 0、接收 FIFO 1 和发送事件 FIFO 的超时条件。它作为递减计数器工作，并且与时间戳计数器一样，使用由 TSCC[TCP] 控制的预分频器。超时计数器通过寄存器 TOCC 配置。实际计数器值可从 TOCV[TOC] 读取。仅当 CCCR[INIT] = 0 时，才会启动超时计数器。当 CCCR[INIT] = 1 时，超时计数器停止计数，例如 FDCAN 进入 Bus_Off 状态。

工作模式是通过 TOCC[TOS] 选择的。在连续模式下工作时，计数器会在 CCCR[INIT] 复位时开始计数。对 TOCV 进行写操作会将计数器预设为由 TOCC[TOP] 配置的值并继续递减计数。

超时计数器由其中一个 FIFO 控制时，空 FIFO 会将计数器预设为由 TOCC[TOP] 配置的值。存储了第一个 FIFO 元素后，会开始递减计数。对 TOCV 执行写入操作不起作用。

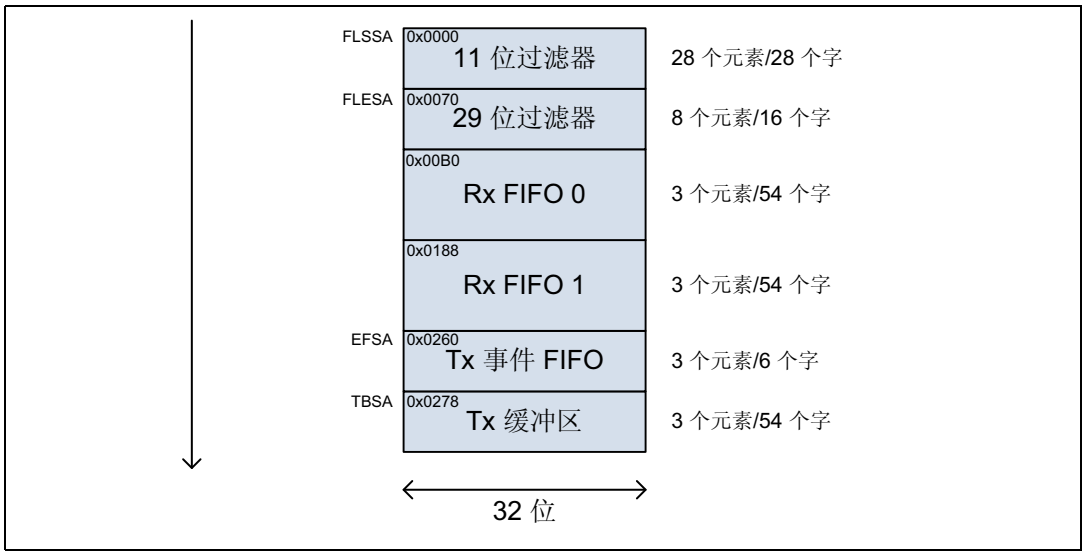
计数器达到 0 时，中断标志 IR[TOO] 会置 1。在连续模式下，计数器会在达到 TOCC[TOP] 后立即重启。

注意： *超时计数器的时钟信号来自 CAN 内核采样点信号。因此，由于 CAN 内核同步/重新同步机制的原因，超时计数器递减的时间点可能有所不同。如果使用 FDCAN 中的波特率切换功能，则超时计数器在仲裁字段和数据字段中使用不同的时钟。*

39.3.3 消息 RAM

消息 RAM 的宽度为 32 位，FDCAN 模块最多可在其中分配 212 个字。不需要一一配置*图 506* 中所示的每个部分。

图 506. 消息 RAM 配置



当 FDCAN 对消息 RAM 进行寻址时，会寻址 32 位字（对齐），而不是单独字节。RAM 地址为 32 位字，也就是说，仅会评估位 15 到位 2，两个最低有效位会被忽略。

在具有多个模块的情况下，FDCANn 的 RAM 起始地址通过 FDCANn-1 的结束地址 + 4 进行计算，FDCANn 结束地址通过 FDCANn 起始地址 + 0x0350 - 4 进行计算。

以具有两个模块的情况为例：

- FDCAN1：
  - 起始地址 0x0000
  - 结束地址 0x034C（如*图 506* 中所示）
- FDCAN2：
  - 起始地址 = 0x034C（FDCAN1 结束地址）+ 4 = 0x0350
  - 结束地址 = 0x0350（FDCAN2 起始地址）+ 0x0350 - 4 = 0x069C。

接收处理

接收处理单元控制着验收过滤、已接收消息到接收缓冲区或其中一个接收 FIFO（共两个）的传输以及接收 FIFO 放入和获取索引。

验收过滤器

FDCAN 能够配置两组验收过滤器，一组供标准标识符使用，另一组供扩展标识符使用。这两组过滤器可分配给接收 FIFO 0 或接收 FIFO 1。进行验收过滤时，会从第 0 个元素开始执行每个过滤器列表，直至过滤出第一个匹配元素。过滤出第一个匹配元素后，验收过滤会停止。不会为该消息评估以下过滤器元素。

主要特性有：

- 每个过滤器元素可配置为
  - 范围过滤器（起止范围）
  - 用于一个或两个专用 ID 的过滤器
  - 典型位屏蔽过滤器
- 每个过滤器元素均可配置为实现验收过滤或拒绝过滤
- 每个过滤器元素均可单独使能/禁止
- 会按顺序检查过滤器，过滤出第一个匹配的过滤器元素后会停止执行

相关配置寄存器位：

- 全局过滤器配置 (RXGFC)
- 扩展 ID 与掩码 (XIDAM)

根据过滤器元素的配置 (SFEC/EFEC)，发生匹配时会触发下列操作之一：

- 将接收到的帧存储在 FIFO 0 或 FIFO 1 中
- 拒绝接收到的帧
- 将高优先级消息中断标志 IR[HPM] 置 1
- 将高优先级消息中断标志 IR[HPM] 置 1，并将接收到的帧存储在 FIFO 0 或 FIFO 1 中

接收到完整的标识符后，开始验收过滤。验收过滤完成后，如果已找到匹配的接收 FIFO，消息处理单元便会开始将接收到的消息数据以 32 位的形式写入匹配的接收 FIFO 中。如果 CAN 协议控制器已检测到错误条件（例如 CRC 错误），则会丢弃此消息，丢弃后的影响如下：

- **接收 FIFO**
  匹配接收 FIFO 的放入索引不会更新，但相关接收 FIFO 元素（部分）会被接收到的数据覆盖。有关错误类型，请参见 PSR.LEC 和 PSR.DLEC。如果匹配的接收 FIFO 在覆盖模式下工作，必须考虑 *接收 FIFO 覆盖模式*中介绍的边界条件。

注意： *当接受的消息写入两个接收 FIFO 之一时，未修改的已接收标识符会独立于所用过滤器进行存储。验收过滤过程的结果很大程度上取决于所配置的过滤器元素的顺序。*

范围过滤器

该过滤器会将所有接收到的帧与由 SF1ID/SF2ID 和 EF1ID/EF2ID 定义的范围内的消息 ID 进行匹配检查。

范围过滤与扩展帧一起使用时，有两种可能的情况：

- EFT = 00：应用范围过滤器之前，会将已接收帧的消息 ID 与上扩展 ID 与掩码 (XIDAM) 进行与运算
- EFT = 11：不会为范围过滤使用扩展 ID 与上掩码 (XIDAM)

专用 ID 过滤器

可将过滤器元素配置为过滤一个或两个特定的消息 ID。要过滤一个特定的消息 ID，过滤器元素必须配置为 SF1ID = SF2ID 和 EF1ID = EF2ID。

典型位屏蔽过滤器

典型位屏蔽过滤用于通过屏蔽已接收消息 ID 的单个位来过滤消息 ID 组。进行典型位屏蔽过滤时，SF1ID/EF1ID 用作消息 ID 过滤器，SF2ID/EF2ID 则用作过滤器屏蔽位。

过滤器屏蔽的 0 位将屏蔽掉已配置 ID 过滤器的相应位位置，例如，已接收消息 ID 在该位位置的值与验收过滤无关。只有相应屏蔽位为 1 的已接收消息 ID 的位才与验收过滤有关。

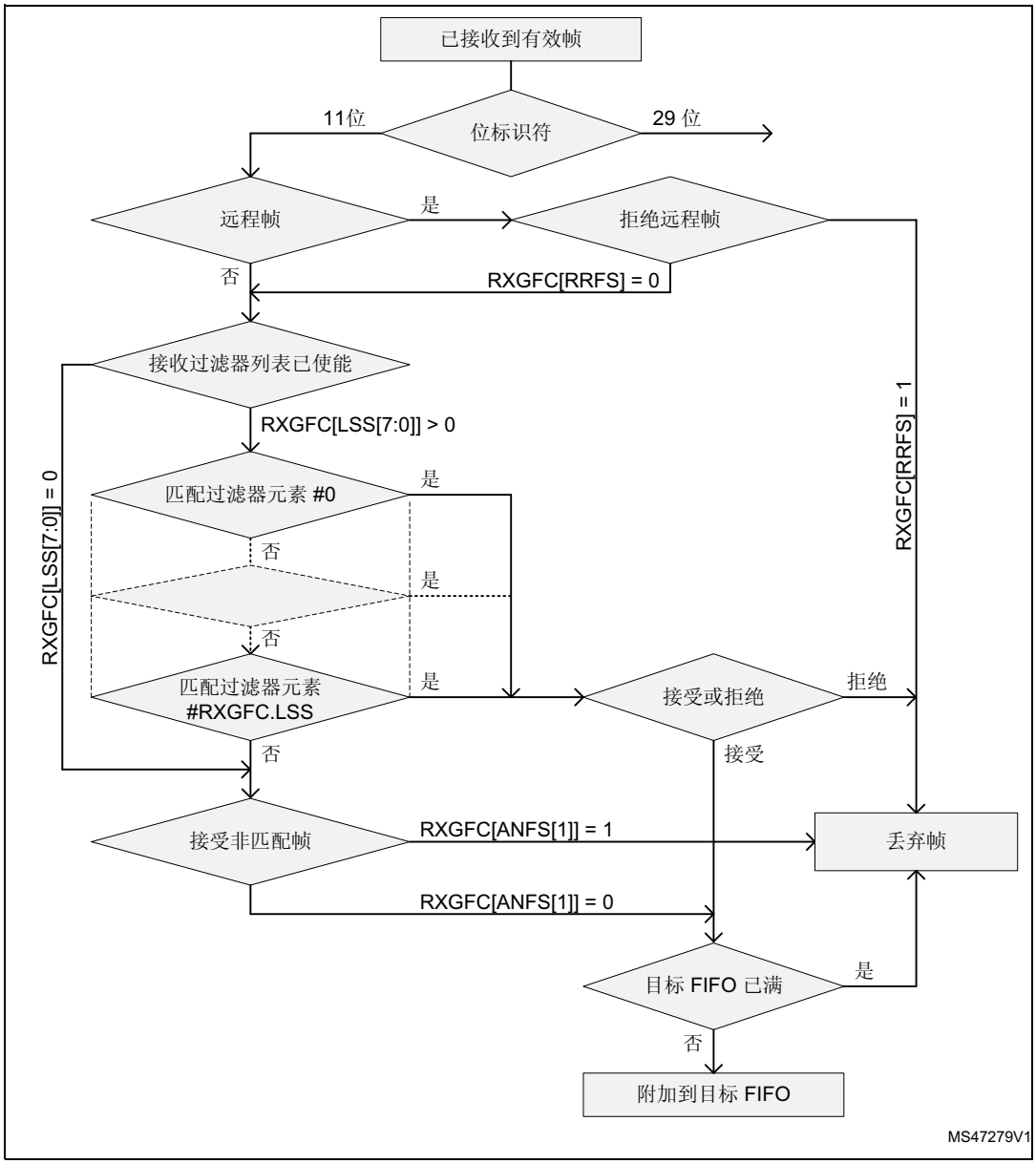
如果所有屏蔽位均为 1，则仅当接收到的消息 ID 和消息 ID 过滤器完全相同时，才会出现匹配。如果所有屏蔽位均为 0，则所有消息 ID 都匹配。

标准消息 ID 过滤

图 507 介绍了标准消息 ID（11 位标识符）的过滤流程。*第 39.3.8 节*中介绍了标准消息 ID 过滤器元素。

已接收帧的远程发送请求位 (RTR) 和标识符扩展位 (IDE) 会在全局过滤器配置 (RXGFC) 消息 ID 的控制下与已配置过滤器元素列表进行比较。

图 507. 标准消息 ID 过滤器路径

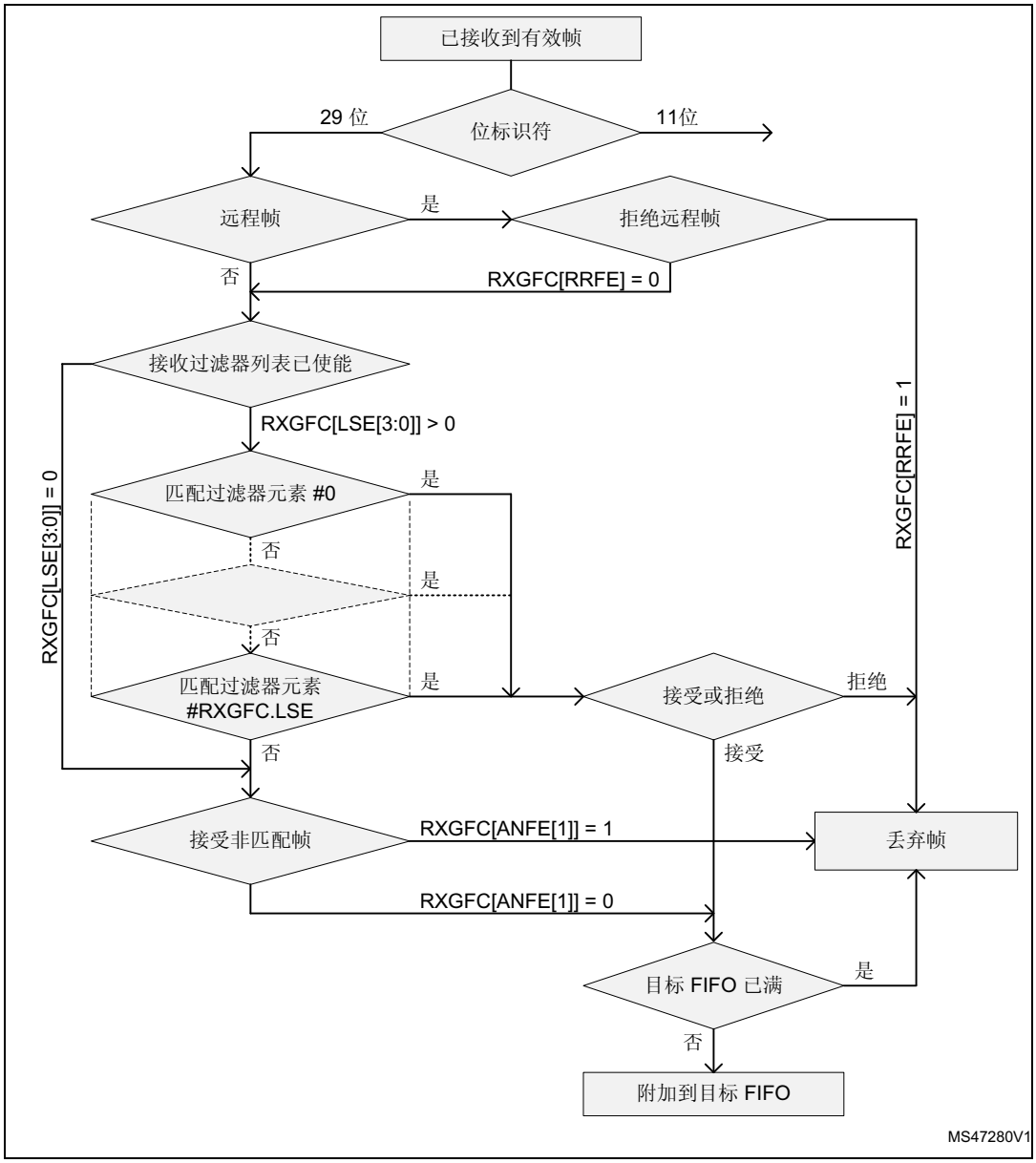


扩展消息 ID 过滤

图 508 介绍了扩展消息 ID（29 位标识符）的过滤流程。*第 39.3.9 节*中介绍了扩展消息 ID 过滤器元素。

已接收帧的远程发送请求位 (RTR) 和标识符扩展位 (IDE) 会在全局过滤器配置 RXGFC 和扩展 ID 过滤器配置 RXGFC 消息 ID 的控制下与已配置过滤器元素列表进行比较。

图 508. 扩展消息 ID 过滤器路径



扩展 ID 与掩码 (XIDAM) 会在执行过滤器列表之前与接收到的标识符进行与运算。

接收 FIFO

接收 FIFO 0 和接收 FIFO 1 最多可分别保存三个元素。

通过验收过滤的已接收消息会传输到由匹配过滤器元素配置的接收 FIFO 中。有关可用于接收 FIFO 0 和接收 FIFO 1 的过滤器机制说明，请参见*验收过滤器。第 39.3.5 节*中介绍了接收 FIFO 元素。

当 IR[RFnF] 指示接收 FIFO 已满条件时，至少在已读取一条消息且接收 FIFO 获取索引递增之前，不会继续向相应的接收 FIFO 写入消息。如果在相应的接收 FIFO 已满时收到消息，此消息会被丢弃，中断标志 IR[RFnL] 会置 1。

从接收 FIFO 中读取消息时，接收 FIFO 获取索引 RXFnS[FnGI] + FIFO 元素大小必须与相应的接收 FIFO 起始地址 [FnSA] 相加。

接收 FIFO 阻止模式

接收 FIFO 阻止模式是通过 RXGFC.FnOM = 0 配置的。该模式是接收 FIFO 的默认工作模式。

达到接收 FIFO 已满条件时 (RXFnS.FnPI = RXFnS.FnGI)，至少在已读取一条消息且接收 FIFO 获取索引递增之前，不会继续向相应的接收 FIFO 写入消息。接收 FIFO 已满条件是通过 RXFnS.FnF = 1 指示的。此外，中断标志 IR.RFnF 也会置 1。

如果在相应的接收 FIFO 已满时收到消息，此消息会被丢弃，并会通过 RXFnS.RFnL = 1 指示消息丢失条件。此外，中断标志 IR.RFnL 也会置 1。

接收 FIFO 覆盖模式

接收 FIFO 覆盖模式是通过 RXGFC.FnOM = 1 配置的。

当 RXFnS.FnF = 1 指示接收 FIFO 已满条件时 (RXFnS.FnPI = RXFnS.FnGI)，FIFO 接收的下一条消息覆盖最原始的 FIFO 消息。放入索引和获取索引都会加 1。

如果接收 FIFO 在覆盖模式下工作，并且指示了接收 FIFO 已满条件，则必须至少从获取索引 + 1 处开始读取接收 FIFO 元素。这是因为 CPU 从消息 RAM（获取索引）读取消息时可能会将接收到的消息写入消息 RAM（放入索引）。在这种情况下，从相应的接收 FIFO 元素中读取的数据可能不一致。从接收 FIFO 读取数据时向获取索引中添加偏移可避免此问题。偏移取决于 CPU 访问 Rx FIFO 的速度。

从接收 FIFO 读取数据后，读取的最后一个元素的编号必须写入接收 FIFO 确认索引 RXFnA.FnA 中。这样会使获取索引增加到该元素编号。如果放入索引未增加到该接收 FIFO 元素，接收 FIFO 已满条件会复位（RXFnS.FnF = 0）。

发送处理

发送处理单元处理发送 FIFO 和发送队列的发送请求。它控制着发送消息到 CAN 内核的传输、放入和获取索引以及发送事件 FIFO，最多可为消息发送设置 3 个发送缓冲区。CAN 消息数据字段配置为 64 字节，Tx FIFO 分配十八个 32 位字用于存储发送元素。

表 318. 帧发送的可能配置

CCCR		发送缓冲区元素		帧发送
BRSE	FDOE	FDF	BRS	
忽略	0	忽略	忽略	经典 CAN
0	1	0	忽略	经典 CAN
0	1	1	忽略	无比特率切换的 FD

表 318. 帧发送的可能配置（续）

CCCR		发送缓冲区元素		帧发送
BRSE	FDOE	FDF	BRS	
1	1	0	忽略	经典 CAN
1	1	1	0	无比特率切换的 FD
1	1	1	1	有比特率切换的 FD

注意： *AUTOSAR 至少需要三个发送队列缓冲区，并支持发送取消。*

当发送缓冲区请求挂起寄存器 TXBRP 更新时，或者发送已启动时，发送处理单元会启动发送扫描来检查优先级最高的挂起发送请求（消息 ID 最小的发送缓冲区）。

发送暂停

发送暂停功能用于 CAN 消息标识符（永久性地）被指定为特定值、并且无法轻易更改的 CAN 系统中。这些消息标识符的 CAN 仲裁优先级可能高于其他已定义的消息，而在特定的应用中，其相对仲裁优先级必须反转。这样可能会导致如下情况：一个 ECU 发送的 CAN 消息突发造成 CAN 仲裁优先级较低的另一个 ECU CAN 消息被延迟。

举例来说，如果 CAN ECU-1 使能了此功能，并且其应用软件请求发送四条消息，成功发送第一条消息后，将等待两个总线空闲的 CAN 位时间，然后才允许开始发送下一条请求的消息。如果其他 ECU 有挂起的消息，则会在空闲时间开始发送这些消息，不需要与 ECU-1 的下一条消息进行仲裁。接收到消息后，ECU-1 可在接收到的消息释放 CAN 总线后立即开始进行下一次发送。

此功能是通过 CCCR 寄存器中的 TXP 位控制的。如果该位置 1，则每次成功发送消息后，FDCAN 都暂停两个 CAN 位时间，然后再开始进行下一次发送。这样一来，即使网络中其他节点的优先级标识符较低，也能发送消息。默认禁止此功能（CCCR.TXP = 0）。

此功能可使来自单个节点的突发发送变得松散一些，避免出现应用程序错误地请求过多传输的“胡乱指示”情况。

发送 FIFO

发送 FIFO 操作是通过将 TXBC[TFQM] 编程为 0 配置的。发送 FIFO 中存储的消息是先从获取索引 TXFQS[TFGI] 引用的消息开始发送的。每次发送后，获取索引会循环递增，直至发送 FIFO 为空。发送 FIFO 可按消息写入发送 FIFO 的顺序发送来自不同发送缓冲区但消息 ID 相同的消息。FDCAN 会计算获取索引和放入索引之差作为发送 FIFO 空闲级别 TXFQS[TFFL]，用于指示可用（空闲）的发送 FIFO 元素数。

新的发送消息必须写入以放入索引 TXFQS[TFQPI] 引用的发送缓冲区开始的发送 FIFO 中。添加请求会将放入索引增加到下一空闲发送 FIFO 元素。放入索引达到获取索引后，会指示发送 FIFO 已满（TXFQS[TFQF]= 1）。在这种情况下，下一条消息已发送且获取索引已递增之前，不得继续向发送 FIFO 写入消息。

如果有一条消息添加到发送 FIFO，则会通过向与发送 FIFO 放入索引引用的发送缓冲区相关的 TXBAR 位写入 1 来请求发送消息。

如果有多条 (n) 消息添加到发送 FIFO，则会写入以放入索引开始的 n 个连续发送缓冲区中。随后会通过 TXBAR 请求发送。放入索引随后会循环递增 n。请求的发送缓冲区数不得超过发送 FIFO 空闲级别指示的空闲发送缓冲区数。

如果获取索引引用的发送缓冲区的发送请求取消，获取索引会增加到下一个具有挂起发送请求的发送缓冲区，并会重新计算发送 FIFO 空闲级别。如果取消对其他任何发送缓冲区的发送，获取索引和 FIFO 空闲级别保持不变。

发送 FIFO 元素会在消息 RAM 中分配十八个 32 位字。因此，下一可用（空闲）发送 FIFO 缓冲区的起始地址是通过将放入索引 TXFQS[TFQPI] (0 ... 2) 乘以四再与发送缓冲区起始地址 TBSA 相加计算得出的。

发送队列

发送队列操作是通过将 TXBC[TFQM] 编程为 1 配置的。发送队列中存储的消息是先从消息 ID 最小（优先级最高）的消息开始发送的。

在标准消息 ID 与扩展消息 ID 混合的情况下，将标准消息 ID 与扩展消息 ID 的位 [28:18] 进行比较。

如果多个队列缓冲区配置为使用同一消息 ID，则会先发送缓冲区编号最小的队列缓冲区。

新消息必须写入放入索引 TXFQS[TFQPI] 引用的发送缓冲区中。添加请求会将放入索引循环增加到下一空闲发送缓冲区。如果发送队列已满（TXFQS[TFQF]= 1），则放入索引无效，并且至少在有一个请求的消息已发出或挂起的发送请求已取消之前，不得继续向发送队列写入消息。

应用可使用寄存器 TXBRP 来代替放入索引，并可将消息放入任何没有挂起传输请求的发送缓冲区中。

发送队列缓冲区会在消息 RAM 中分配十八个 32 位字。因此，下一可用（空闲）发送队列缓冲区的起始地址是通过将发送队列放入索引 TXFQS[TFQPI] (0...2) 乘以四再与发送缓冲区起始地址 TBSA 相加计算得出的。

发送取消

FDCAN 支持发送取消功能。要取消专发送队列缓冲区请求的发送，主机必须向寄存器 TXBCR 中相应的位位置（= 发送缓冲区的数量）写入 1。发送取消不适用于发送 FIFO 操作。

成功取消后，寄存器 TXBCF 的相应位会设为 1。

如果已经在从发送缓冲区进行发送时请求发送取消，则只要发送正在进行中，相应的 TXBRP 位就会保持置 1 状态。如果发送成功，相应的 TXBTO 和 TXBCF 位会置 1。如果发送不成功，则不会重复发送，只会将相应的 TXBCF 位置 1。

注意： *如果在挂起的发送尚未开始之前立即将其取消，即使该节点中有另一消息挂起，也会有一个短暂的时间窗，在此期间不会开始任何发送。这样一来，另一节点便可以发送优先级可能比该节点中第二条消息低的消息。*

发送事件处理

为了支持发送事件处理，FDCAN 实现了发送事件 FIFO。FDCAN 在 CAN 总线上发送消息后，消息 ID 和时间戳会存储在发送事件 FIFO 元素中。为了将发送事件关联到发送事件 FIFO 元素，已发送的发送缓冲区中的消息标志会被复制到发送事件 FIFO 元素中。

发送事件 FIFO 配置为三个元素。*发送 FIFO* 中介绍了发送事件 FIFO 元素。

发送事件 FIFO 的用途是将处理发送状态信息与处理发送消息分开，也就是让发送缓冲区仅保存要发送的消息，而将发送状态单独存储在发送事件 FIFO 中。这样做有很大的优势，尤其是在处理动态管理的发送队列时，发送缓冲区可在发送成功后立即用于新消息。覆盖发送缓冲区之前，不需要保存该发送缓冲区的发送状态信息。

当 IR[TEFF] 指示发送事件 FIFO 已满条件时，在至少已读取一个元素且发送事件 FIFO 获取索引递增之前，不会继续向发送事件 FIFO 写入元素。如果在发送事件 FIFO 已满时发生发送事件，此事件会被丢弃，中断标志 IR[TEFL] 会置 1。

从发送事件 FIFO 读取数据时，必须将发送事件 FIFO 获取索引 TXEFS[EFGI] 乘以二再与发送事件 FIFO 起始地址 EFSA 相加。

39.3.4 FIFO 确认处理

接收 FIFO 0、接收 FIFO 1 和发送事件 FIFO 是通过向相应的 FIFO 确认索引进行写操作控制的，请参见*第 39.4.23 节*和*第 39.4.25 节*。对 FIFO 确认索引进行写操作会将 FIFO 获取索引设置为 FIFO 确认索引加 1，进而会更新 FIFO 填充级别。有两种用例：

1. 如果只从 FIFO 读取了一个元素（获取索引指向的元素），则该获取索引值会写入 FIFO 确认索引中。
2. 如果已从 FIFO 中读取一系列元素，则仅在该读取序列结束时写入一次 FIFO 确认索引（数值：读取的最后一个元素的索引）即可更新 FIFO 获取索引。

由于 CPU 可自由访问 FDCAN 消息 RAM，以任意顺序（不考虑获取索引）读取 FIFO 元素时需要特别留意。这一点在从两个接收 FIFO 之一读取高优先级消息时会很有用。在这种情况下，不得写入 FIFO 确认索引，否则会将获取索引设为错误的位置，还会改变 FIFO 填充级别。在这种情况下，一些时间较久的 FIFO 元素会丢失。

注意： *应用必须确保 FIFO 确认索引中已写入有效值。FDCAN 不会检查数值是否正确。*

39.3.5 FDCAN 接收 FIFO 元素

消息 RAM 中配置两个接收 FIFO。每个接收 FIFO 部分均可配置为最多存储三个已接收消息。接收 FIFO 元素的结构如*表 319* 所示，相关说明请参见*表 320*。

表 319. 接收 FIFO 元素

位	31			24	23				16	15	8	7	0
R0	ESI	XTD	RTR		ID[28:0]								
R1	ANMF	FIDX[6:0]			Res.	FDF	BRS	DLC[3:0]		RXTS[15:0]			
R2	DB3[7:0]				DB2[7:0]					DB1[7:0]		D[7:0]	
R3	DB7[7:0]				DB6[7:0]					DB5[7:0]		DB4[7:0]	
⋮	⋮				⋮					⋮			
Rn	DBm[7:0]				DBm-1[7:0]					DBm-2[7:0]		DBm-3[7:0]	

可对元素大小进行配置，以存储数据字段为 64 字节的 CAN FD 消息。

表 320. 接收 FIFO 元素说明

位域	说明
R0 位 31 ESI	错误状态指示符 (Error state indicator) – 0：发送节点为错误主动状态 – 1：发送节点为错误被动状态
R0 位 30 XTD	扩展标识符 (Extended identifier) 向主机指示接收到的帧具有标准标识符还是扩展标识符。 – 0：11 位标准标识符 – 1：29 位扩展标识符
R0 位 29 RTR	远程发送请求 (Remote transmission request) 向主机指示接收到的帧属于数据帧还是远程帧。 – 0：接收到的帧为数据帧 – 1：接收到的帧为远程帧
R0 位 28:0 ID[28:0]	标识符 (Identifier) 标准标识符或扩展标识符取决于 XTD 位。标准标识符存储在 ID[28:18] 中。
R1 位 31 ANMF	接受非匹配帧 (Accepted non-matching frame) 可通过 RXGFC[ANFS] 和 RXGFC[ANFE] 使能接受非匹配帧。 – 0：接收到的帧与过滤器索引 FIDX 匹配 – 1：接收到的帧与任何接收过滤器元素都不匹配
R1 位 30:24 FIDX[6:0]	过滤器索引 (Filter index) 0-27 = 匹配接收验收过滤器元素的索引（ANMF = 1 时无效）。 范围为 0 到 RXGFC[LSS] - 1 或 RXGFC[LSE] - 1。
R1 位 21 FDF	FD 格式 (FD format) – 0：标准帧格式 – 1：FDCAN 帧格式（新 DLC 编码和 CRC）
R1 位 20 BRS	比特率切换 (Bit rate switch) – 0：接收帧时不切换比特率 – 1：接收帧时切换比特率
R1 位 19:16 DLC[3:0]	数据长度代码 (Data length code) – 0-8：经典 CAN + CAN FD：接收到的帧包含 0-8 个数据字节 – 9-15：经典 CAN：接收到的帧包含 8 个数据字节 – 9-15：CAN FD：接收到的帧包含 12/16/20/24/32/48/64 个数据字节
R1 位 15:0 RXTS[15:0]	接收时间戳 (Rx timestamp) 开始接收帧时捕获的时间戳计数器值。分辨率取决于时间戳计数器预分频器 TSCC[TCP] 的配置。
R2 位 31:24 DB3[7:0]	数据字节 3 (Data byte 3)
R2 位 23:16 DB2[7:0]	数据字节 2 (Data byte 2)
R2 位 15:8 DB1[7:0]	数据字节 1 (Data byte 1)
R2 位 7:0 D[7:0]	数据字节 0 (Data byte 0)

表 320. 接收 FIFO 元素说明（续）

位域	说明
R3 位 31:24 DB7[7:0]	数据字节 7 (Data byte 7)
R3 位 23:16 DB6[7:0]	数据字节 6 (Data byte 6)
R3 位 15:8 DB5[7:0]	数据字节 5 (Data byte 5)
R3 位 7:0 DB4[7:0]	数据字节 4 (Data byte 4)
⋮	⋮
Rn 位 31:24 DBm[7:0]	数据字节 m (Data byte m)
Rn 位 23:16 DBm-1[7:0]	数据字节 m-1 (Data byte m-1)
Rn 位 15:8 DBm-2[7:0]	数据字节 m-2 (Data byte m-2)
Rn 位 7:0 DBm-3[7:0]	数据字节 m-3 (Data byte m-3)

39.3.6 FDCAN 发送缓冲区元素

发送缓冲区部分（三个元素）可配置为保存发送 FIFO 或发送队列。发送处理单元使用发送缓冲区配置 FDCAN_TXBC.TFQM 来区分发送 FIFO 与发送队列。可对元素大小进行配置，以存储数据多达 64 字节的 CAN FD 消息。

表 321. 发送缓冲区和 FIFO 元素

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T0	ESI	XTD	RTR	ID[28:0]																												
T1	MM[7:0]								EFC	Res.	FDF	BPS	DLC[3:0]				Res.															
T2	DB3[7:0]								DB2[7:0]								DB1[7:0]								D[7:0]							
T3	DB7[7:0]								DB6[7:0]								DB5[7:0]								DB4[7:0]							
⋮	⋮								⋮								⋮															
Tn	DBm[7:0]								DBm-1[7:0]								DBm-2[7:0]								DBm-3[7:0]							

表 322. 发送缓冲区元素说明

位域	说明
T0 位 31 ESI(1)	错误状态指示符 (Error state indicator) – 0：CAN FD 格式的 ESI 位仅取决于错误被动标志 – 1：CAN FD 格式的 ESI 位会进行隐性发送
T0 位 30 XTD	扩展标识符 (Extended identifier) – 0：11 位标准标识符 – 1：29 位扩展标识符
T0 位 29 RTR(2)	远程发送请求 (Remote transmission request) – 0：发送数据帧 – 1：发送远程帧
T0 位 28:0 ID[28:0]	标识符 (Identifier) 标准标识符或扩展标识符取决于 XTD 位。必须将标准标识符写入 ID[28:18]。
T1 位 31:24 MM[7:0]	消息标记 (Message marker) 在发送缓冲区配置期间由 CPU 写入。复制到发送事件 FIFO 元素中，用于标识发送消息状态。
T1 位 23 EFC	事件 FIFO 控制 (Event FIFO control) – 0：不存储发送事件 – 1：存储发送事件
T1 位 21 FDF	FD 格式 (FD format) – 0：以经典 CAN 格式发送帧 – 1：以 CAN FD 格式发送帧
T1 位 20 BRS(3)	比特率切换 (Bit rate switching) – 0：发送 CAN FD 帧时不切换比特率 – 1：发送 CAN FD 帧时切换比特率
T1 位 19:16 DLC[3:0]	数据长度代码 (Data length code) – 0 - 8：经典 CAN + CAN FD：接收到的帧包含 0-8 个数据字节 – 9-15：经典 CAN：接收到的帧包含 8 个数据字节 – 9-15：CAN FD：接收到的帧包含 12/16/20/24/32/48/64 个数据字节
T2 位 31:24 DB3[7:0]	数据字节 3 (Data byte 3)
T2 位 23:16 DB2[7:0]	数据字节 2 (Data byte 2)
T2 位 15:8 DB1[7:0]	数据字节 1 (Data byte 1)
T2 位 7:0 D[7:0]	数据字节 0 (Data byte 0)
T3 位 31:24 DB7[7:0]	数据字节 7 (Data byte 7)
T3 位 23:16 DB6[7:0]	数据字节 6 (Data byte 6)
T3 位 15:8 DB5[7:0]	数据字节 5 (Data byte 5)

表 322. 发送缓冲区元素说明（续）

位域	说明
T3 位 7:0 DB4[7:0]	数据字节 4 (Data byte 4)
⋮	⋮
Tn 位 31:24 DBm[7:0]	数据字节 m (Data byte m)
Tn 位 23:16 DBm-1[7:0]	数据字节 m-1 (Data byte m-1)
Tn 位 15:8 DBm-2[7:0]	数据字节 m-2 (Data byte m-2)
Tn 位 7:0 DBm-3[7:0]	数据字节 m-3 (Data byte m-3)

1. 发送缓冲区的 ESI 位会与错误被动标志进行或运算，以确定已发送 FD 帧中 ESI 位的值。按照 CAN FD 协议规范的要求，错误主动节点可选择隐性发送 ESI 位，但错误被动节点始终隐性发送 ESI 位。
2. 当 RTR = 1 时，即使 CCCR.FDOE 使能以 CAN FD 格式进行发送，FDCAN 也会按照 ISO11898-1 的规定发送远程帧。
3. 仅当 CAN FD 操作使能时 (CCCR.FDOE = 1)，才会评估 ESI、FDF 和 BRS 位。仅当额外设置 CCCR.BRSE = 1 时，才会评估 BRS 位。

39.3.7 FDCAN 发送事件 FIFO 元素

每个元素均存储与已发送消息相关的消息。通过读取发送事件 FIFO，主机 CPU 会按消息的发送顺序获取此类信息。关于发送事件 FIFO 的状态信息，可从寄存器 TXEFS 中获取。

表 323. 发送事件 FIFO 元素

位	31			24	23					16	15	8	7	0
E0	ESI	XTD	RTR	ID[28:0]										
E1	MM[7:0]				ET[1:0]	EDL	BRS	DLC[3:0]	TXTS[15:0]					

表 324. 发送事件 FIFO 元素说明

位域	说明
E0 位 31 ESI	错误状态指示符 (Error state indicator) – 0：发送节点为错误主动状态 – 1：发送节点为错误被动状态
E0 位 30 XTD	扩展标识符 (Extended identifier) – 0：11 位标准标识符 – 1：29 位扩展标识符
E0 位 29 RTR	远程发送请求 (Remote transmission request) – 0：发送数据帧 – 1：发送远程帧

表 324. 发送事件 FIFO 元素说明（续）

位域	说明
E0 位 28:0 ID[28:0]	标识符 (Identifier) 标准标识符或扩展标识符取决于 XTD 位。必须将标准标识符写入 ID[28:18]。
E1 位 31:24 MM[7:0]	消息标记 (Message marker) 从发送缓冲区复制到发送事件 FIFO 元素中，用于标识发送消息状态。
E1 位 23:22 EFC	事件类型 (Event type) – 00：保留 – 01：发送事件 – 10：取消后仍发送（在 DAR 模式下发送时始终设为该值） – 11：保留
E1 位 21 EDL	扩展数据长度 (Extended data length) – 0：标准帧格式 – 1：FDCAN 帧格式（新 DLC 编码和 CRC）
E1 位 20 BRS	比特率切换 (Bit rate switching) – 0：发送帧时不切换比特率 – 1：发送帧时切换比特率
T1 位 19:16 DLC[3:0]	数据长度代码 (Data length code) 0 - 8：发送包含 0-8 个数据字节的帧 9-15：发送包含 8 个数据字节的帧
E1 位 15:0 TXTS[15:0]	发送时间戳 (Tx Timestamp) 开始发送帧时捕获的时间戳计数器值。分辨率取决于时间戳计数器预分频器 TSCC[TCP] 的配置。

39.3.8 FDCAN 标准消息 ID 过滤器元素

最多可为 11 位标准 ID 配置 28 个过滤器元素。访问标准消息 ID 过滤器元素时，其地址为过滤器列表标准起始地址 FLSSA 加上过滤器元素的索引 (0 ... 27)。

表 325. 标准消息 ID 过滤器元素

位	31		24	23	16	15	8	7	0
S0	SFT[1:0]	SFEC[2:0]		SFID1[10:0]		Res.		SFID2[10:0]	

表 326. 标准消息 ID 过滤器元素位域说明

位域	说明
位 31:30 SFT[1:0](1)	标准过滤器类型 (Standard filter type) – 00：从 SFID1 到 SFID2 的范围过滤器 – 01：用于 SFID1 或 SFID2 的双 ID 过滤器 – 10：典型过滤器：SFID1 = 过滤器，SFID2 = 掩码 – 11：禁止过滤器元素
位 29:27 SFEC[2:0]	标准过滤器元素配置 (Standard filter element configuration) 所有使能的过滤器元素都用于对标准帧进行验收过滤。当发现第一个匹配的已使能过滤器元素时，或者到达过滤器列表结尾处时，会停止验收过滤。如果 SFEC = 100、101 或 110，则出现匹配时，会将中断标志 IR.HPM 置 1，并会生成中断（若使能）。在这种情况下，寄存器 HPMS 会更新为优先级匹配的状态。 – 000：禁止过滤器元素 – 001：如果过滤器匹配，则存储在接收 FIFO 0 中 – 010：如果过滤器匹配，则存储在接收 FIFO 1 中 – 011：如果过滤器匹配，则拒绝 ID – 100：如果过滤器匹配，则设置优先级 – 101：如果过滤器匹配，则设置优先级并存储在 FIFO 0 中 – 110：如果过滤器匹配，则设置优先级并存储在 FIFO 1 中 – 111：未使用
位 26:16 SFID1[10:0]	标准过滤器 ID 1 (Standard filter ID 1) 标准 ID 过滤器元素的第一个 ID。
位 10:0 SFID2[10:0]	标准过滤器 ID 2 (Standard filter ID 2) 标准 ID 过滤器元素的第二个 ID。

1. 当 SFT = 11 时，会禁止过滤器元素，验收过滤会继续进行（与 SFEC =“000”时的行为相同）。

注意： *如果配置了保留值，则会将过滤器元素视为已禁止。*

39.3.9 FDCAN 扩展消息 ID 过滤器元素

最多可为 29 位扩展 ID 配置 8 个过滤器元素。访问扩展消息 ID 过滤器元素时，其地址为过滤器列表扩展起始地址 FLESA 加上过滤器元素的索引 (0 ... 7) 的二倍。

表 327. 扩展消息 ID 过滤器元素

位	31		24	23 16	15 8	7 0
F0	EFEC[2:0]		EFID1[28:0]			
F1	EFTI[1:0]	Res.	EFID2[28:0]			

表 328. 扩展消息 ID 过滤器元素位域说明

位域	说明
F0 位 31:29 EFEC[2:0]	扩展过滤器元素配置 (Extended filter element configuration) 所有使能的过滤器元素都用于对扩展帧进行验收过滤。当发现第一个匹配的已使能过滤器元素时，或者到达过滤器列表结尾处时，会停止验收过滤。如果 EFEC = 100、101 或 110，则出现匹配时，会将中断标志 IR[HPM] 置 1，并会生成中断（若使能）。在这种情况下，寄存器 HPMS 会更新为优先级匹配的状态。 – 000：禁止过滤器元素 – 001：如果过滤器匹配，则存储在接收 FIFO 0 中 – 010：如果过滤器匹配，则存储在接收 FIFO 1 中 – 011：如果过滤器匹配，则拒绝 ID – 100：如果过滤器匹配，则设置优先级 – 101：如果过滤器匹配，则设置优先级并存储在 FIFO 0 中 – 110：如果过滤器匹配，则设置优先级并存储在 FIFO 1 中 – 111：未使用
F0 位 28:0 EFID1[28:0]	扩展过滤器 ID 1 (Extended filter ID 1) 扩展 ID 过滤器元素的第一个 ID。 过滤接收 FIFO 时，该位域定义要存储的扩展消息的 ID。接收的标识符必须完全匹配，仅使用 XIDAM 屏蔽机制。
F1 位 31:30 EFT[1:0]	扩展过滤器类型 (Extended filter type) – 00：从 EF1ID 到 EF2ID 的范围过滤器 (EF2ID >= EF1ID) – 01：用于 EF1ID 或 EF2ID 的双 ID 过滤器 – 10：典型过滤器：EF1ID = 过滤器，EF2ID = 掩码 – 11：从 EF1ID 到 EF2ID 的范围过滤器 (EF2ID >= EF1ID)，未应用 XIDAM 掩码
F1 位 29	未使用
F1 位 28:0 EFID2[28:0]	扩展过滤器 ID 2 (Extended filter ID 2) 扩展 ID 过滤器元素的第二个 ID。

39.4 FDCAN 寄存器

39.4.1 FDCAN 内核释放寄存器 (FDCAN_CREL)

FDCAN core release register

偏移地址：0x0000

复位值：0x3214 1218

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

位 31:28 **REL[3:0]**：3

位 27:24 **STEP[3:0]**：2

位 23:20 **SUBSTEP[3:0]**：1

位 19:16 **YEAR[3:0]**：4

位 15:8 **MON[7:0]**：12

位 7:0 **DAY[7:0]**：18

39.4.2 FDCAN 字节序寄存器 (FDCAN_ENDN)

FDCAN endian register

偏移地址：0x0004

复位值：0x8765 4321

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

位 31:0 **ETV[31:0]**：字节序测试值 (Endianness test value)
字节序测试值为 0x8765 4321。

注意： *寄存器读操作必须给出复位值，以确保没有字节序问题。*

39.4.3 FDCAN 数据位定时和预分频器寄存器 (FDCAN_DBTP)

FDCAN data bit timing and prescaler register

偏移地址：0x000C

复位值：0x0000 0A33

仅当 CCCR.CCE 和 CCCR.INIT 位置 1 时，才可以向此寄存器写入数据。CAN 时间片的可编程范围为 1 到 32 个 FDCAN 时钟周期。tq = (DBRP + 1) 个 FDCAN 时钟周期。

DTSEG1 为 Prop_Seg 与 Phase_Seg1 之和。DTSEG2 为 Phase_Seg2。因此，位时间的长度为（编程值）[DTSEG1 + DTSEG2 + 3] tq 或（函数值）[Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq。

信息处理时间 (IPT) 为 0，这意味着下一位的数据在采样点后的第一个时钟边沿可用。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDC	Res.	Res.	DBRP[4:0]				
								rw			rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DTSEG1[4:0]					DTSEG2[3:0]				DSJW[3:0]			
			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23 **TDC**：收发器延迟补偿 (Transceiver delay compensation)
0：禁止收发器延迟补偿
1：使能收发器延迟补偿

位 22:21 保留，必须保持复位值。

位 20:16 **DBRP[4:0]**：数据比特率预分频器 (Data bit rate prescaler)
生成位时间片时将振荡器频率除以的值。位时间为该时间片的倍数。比特率预分频器的有效值为 0 到 31。硬件将该值解析为编程值加 1。

位 15:13 保留，必须保持复位值。

位 12:8 **DTSEG1[4:0]**：采样点之前的数据时间段 (Data time segment before sample point)
有效值为 0 到 31。硬件使用的值为编程值加 1，即 t_{BS1} = (DTSEG1 + 1) x tq。

位 7:4 **DTSEG2[3:0]**：采样点之后的数据时间段 (Data time segment after sample point)
有效值为 0 到 15。硬件使用的值为编程值加 1，即 t_{BS2} = (DTSEG2 + 1) x tq。

位 3:0 **DSJW[3:0]**：同步跳转宽度 (Synchronization jump width)
有效值为 0 到 15。硬件使用的值为编程值加 1：t_{SJW} = (DSJW + 1) x tq。

注意： *如果 FDCAN 时钟为 8 MHz，复位值 0x00000A33 会将 FDCAN 的比特率配置为较快的 500 kb/s。*

注意： *数据阶段比特率必须高于或等于标称比特率。*

39.4.4 FDCAN 测试寄存器 (FDCAN_TEST)

FDCAN test register

必须通过将 CCCR[TEST] 位置 1 的方式使能对该寄存器的写访问。CCCR[TEST] 位复位时，所有寄存器功能都会设为其复位值。

接收引脚 FDCANx_TX 的环回模式和软件控制属于硬件测试模式。将 TX 编程为 00 以外的值可能会干扰 CAN 总线上的消息传输。

偏移地址：0x0010

复位值：0x0000 0000

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

位 31:8 保留，必须保持复位值。

位 7 **RX**：接收引脚 (Receive pin)

监视 FDCANx_RX 引脚的实际值

0：CAN 总线为显性 (FDCANx_RX = 0)

1：CAN 总线为隐性 (FDCANx_RX = 1)

位 6:5 **TX[1:0]**：发送引脚控制 (Control of transmit pin)

00：复位值，FDCANx_TX 由 CAN 内核控制，会在 CAN 位时间结束时更新

01：可在引脚 FDCANx_TX 监控采样点

10：引脚 FDCANx_TX 上为显性 (0) 电平

11：引脚 FDCANx_TX 上为隐性 (1) 电平

位 4 **LBCK**：环回模式 (Loop back mode)

0：复位值，禁止环回模式

1：使能环回模式（请参见*掉电（休眠模式）*）

位 3:0 保留，必须保持复位值。

39.4.5 FDCAN RAM 看门狗寄存器 (FDCAN_RWD)

FDCAN RAM watchdog register

RAM 看门狗监视消息 RAM 的 READY 输出。对消息 RAM 进行访问会启动消息 RAM 看门狗计数器，计数器的起始值由 RWD[WDC] 位配置。

当消息 RAM 通过激活其 READY 输出指示访问成功完成时，计数器会重新载入 RWD[WDC] 位的值。如果计数器递减为 0 之前消息 RAM 没有发出响应，则计数器会停止计数，中断标志 IR[WDI] 位置 1。RAM 看门狗计数器由 fdcan_pclk 时钟驱动。

偏移地址：0x0014

复位值：0x0000 0000

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

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

位 31:16 保留，必须保持复位值。

位 15:8 **WDV[7:0]**：看门狗值 (Watchdog value)
实际消息 RAM 看门狗计数器值。

位 7:0 **WDC[7:0]**：看门狗配置 (Watchdog configuration)
消息 RAM 看门狗计数器的起始值。如果使用复位值 00，计数器会禁止。
这些位受写保护 (P)，仅当 FDCAN_CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能进行写访问。

39.4.6 FDCAN CC 控制寄存器 (FDCAN_CCCR)

FDCAN CC control register

偏移地址：0x0018

复位值：0x0000 0001

有关各个位置 1 和复位的详细信息，请参见*软件初始化*。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NISO	TXP	EFBI	PXHD	Res.	Res.	BRSE	FDOE	TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	r	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15 **NISO**：非 ISO 操作 (Non ISO operation)
如果该位置 1，FDCAN 会使用 Bosch CAN FD 规范 V1.0 规定的 CAN FD 帧格式。
0：符合 ISO11898-1 规定的 CAN FD 帧格式
1：符合 Bosch CAN FD 规范 V1.0 规定的 CAN FD 帧格式

位 14 **TXP**：
如果该位置 1，成功发送帧后，FDCAN 会先暂停两个 CAN 位时间，然后再开始进行下一次发送。
0：禁止
1：使能

位 13 **EFBI**：总线同步期间的边沿过滤 (Edge filtering during bus integration)
0：禁止边沿过滤
1：需要两个连续显性 tq 才能检测硬同步边沿

位 12 **PXHD**：协议异常处理禁止 (Protocol exception handling disable)
0：使能协议异常处理
1：禁止协议异常处理

位 11:10 保留，必须保持复位值。

位 9 **BRSE**：FDCAN 比特率切换 (FDCAN bit rate switching)
0：禁止发送时进行比特率切换
1：使能发送时进行比特率切换

位 8 **FDOE**：FD 操作使能 (FD operation enable)
0：禁止 FD 操作
1：使能 FD 操作

位 7 **TEST**：测试模式使能 (Test mode enable)
0：正常操作，寄存器 TEST 保存复位值
1：测试模式，使能对寄存器 TEST 的写访问

位 6 **DAR**：禁止自动重发送 (Disable automatic retransmission)
0：使能自动重发送未成功发送的消息
1：禁止自动重发送

位 5 **MON**：总线监控模式 (Bus monitoring mode)
仅当 CCE 和 INIT 均置 1 时，才能通过软件将 MON 位置 1。该位可随时通过主机复位。
0：禁止总线监控模式
1：使能总线监控模式

位 4 **CSR**：时钟停止请求 (Clock stop request)
0：未请求时钟停止
1：已请求时钟停止。如果请求时钟停止，则在所有挂起的传输请求均已完成且 CAN 总线达到空闲状态之后，INIT 会先置 1，然后 CSA 会置 1。

位 3 **CSA**：时钟停止确认 (Clock stop acknowledge)
0：未确认时钟停止
1：可通过停止 APB 时钟和内核时钟将 FDCAN 设为掉电状态。

位 2 **ASM**：ASM 受限工作模式 (ASM restricted operation mode)
受限工作模式可用于根据不同 CAN 比特率对自身进行调整的应用中。应用会测试不同比特率，并在收到有效帧后退出受限工作模式。在可选的受限工作模式下，节点能够发送和接收数据帧和远程帧，并能对有效帧进行确认，但不会发送有效错误帧或过载帧。如果存在错误条件或过载条件，则不会发送显性位，而会等待出现总线空闲条件，以便使自身与 CAN 通信重新同步。错误计数器不会递增。仅当 CCE 和 INIT 均置 1 时，才能通过软件将 ASM 位置 1。该位可随时通过软件复位。
0：正常 CAN 操作
1：受限工作模式激活

位 1 **CCE**：配置更改使能 (Configuration change enable)
0：CPU 对受保护的配置寄存器没有写访问权限。
1：CPU 对受保护的配置寄存器有写访问权限（当 CCCR.INIT = 1 时）。

位 0 **INIT**：初始化
0：正常工作
1：启动初始化

注意： *由于两个时钟域之间存在同步机制，可能会经过一定的延迟之后才能读取写入 INIT 的值。因此，编程人员必须确保之前写入 INIT 中的值已通过读取 INIT 的方式被接受，然后再将 INIT 设为新值。*

39.4.7 FDCAN 标称位定时和预分频器寄存器 (FDCAN_NBTP)

FDCAN nominal bit timing and prescaler register

偏移地址：0x001C

复位值：0x0600 0A03

仅当 CCCR[CCE] 和 CCCR[INIT] 位置 1 时，才可以向此寄存器写入数据。CAN 位时间的可编程范围为 4 到 81 个 tq。CAN 时间片的可编程范围为 1 到 1024 个 FDCAN 内核时钟周期。

tq = (BRP + 1) 个 FDCAN 时钟周期 fdcan_clk

NTSEG1 为 Prop_Seg 与 Phase_Seg1 之和。NTSEG2 为 Phase_Seg2。因此，位时间的长度为（编程值）[NTSEG1 + NTSEG2 + 3] tq 或（函数值）[Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq。

信息处理时间 (IPT) 为 0，这意味着下一位的数据在采样点后的第一个时钟边沿可用。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NSJW[6:0]							NBRP[8:0]								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NTSEG1[7:0]								Res.	NTSEG2[6:0]						
rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

位 31:25 **NSJW[6:0]**：标称（重新）同步跳转宽度 (Nominal (re)synchronization jump width)
有效值为 0 到 127。实际上，硬件会将该值解析为编程值加 1。
这些位受写保护 (P)，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能进行写访问。

位 24:16 **NBRP[8:0]**：比特率预分频器 (Bit rate prescaler)
生成位时间片时将振荡器频率除以的值。位时间为该时间片的倍数。有效值为 0 到 511。实际上，硬件会将该值解析为编程值加 1。
这些位受写保护 (P)，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能进行写访问。

位 15:8 **NTSEG1[7:0]**：采样点之前的标称时间段 (Nominal time segment before sample point)
有效值为 0 到 255。实际上，硬件会将该值解析为编程值加 1。
这些位受写保护 (P)，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能进行写访问。

位 7 保留，必须保持复位值。

位 6:0 **NTSEG2[6:0]**：采样点之后的标称时间段 (Nominal time segment after sample point)
有效值为 0 到 127。实际上，硬件会将该值解析为编程值加 1。

注意： *如果 CAN 内核时钟为 48 MHz，复位值 0x06000A03 会将 FDCAN 的比特率配置为 3 Mb/s。*

39.4.8 FDCAN 时间戳计数器配置寄存器 (FDCAN_TSCC)

FDCAN timestamp counter configuration register

偏移地址：0x0020

复位值：0x0000 0000

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

位 31:20 保留，必须保持复位值。

位 19:16 **TCP[3:0]**：时间戳计数器预分频器 (Timestamp counter prescaler)
将时间戳和超时计数器时间单位配置为 CAN 位时间的倍数
[1 ... 16]。
实际上，硬件会将该值解析为编程值加 1。
在 CAN FD 模式下，由于仲裁阶段与数据阶段之间的 CAN 位时间不同，内部时间戳计数器 TCP 不会提供恒定的时基。因此，CAN FD 需要使用外部计数器生成时间戳 (TSS = 10)。
这些位受写保护 (P)，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能进行写访问。

位 15:2 保留，必须保持复位值。

位 1:0 **TSS[1:0]**：时间戳选择 (Timestamp select)
00：时间戳计数器值始终为 0x0000
01：时间戳计数器值按 TCP 递增
10：TIM3 提供的外部时间戳计数器值 (tim3_cnt[0:15])
11：与 00 相同。
这些位受写保护 (P)，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能进行写访问。

39.4.9 FDCAN 时间戳计数器值寄存器 (FDCAN_TSCV)

FDCAN timestamp counter value register

偏移地址：0x0024

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15:0 **TSC[15:0]**：时间戳计数器 (Timestamp counter)

内部/外部时间戳计数器值是在（接收和发送）帧开始时捕获的。当 TSCC[TSS] = 01 时，时间戳计数器会以 CAN 为时间的倍数 [1 ... 16] 递增，具体取决于 TSCC[TCP] 的配置。回卷操作会将中断标志 IR[TSW] 置 1。写访问会将计数器复位为 0。

当 TSCC.TSS = 10 时，TSC 会反映外部时间戳计数器值。写访问没有影响。

注意： *“回卷”是指时间戳计数器值不是由于对 TSCV 进行写访问的原因而由非零值变为 0。*

39.4.10 FDCAN 超时计数器配置寄存器 (FDCAN_TOCC)

FDCAN timeout counter configuration register

偏移地址：0x0028

复位值：0xFFFF 0000

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

位 31:16 **TOP[15:0]**：超时时长 (Timeout period)

超时计数器（递减计数器）的起始值。配置超时时长。

位 15:3 保留，必须保持复位值。

位 2:1 **TOS[1:0]**：超时选择 (Timeout select)

在连续模式下工作时，对 TOCV 进行写访问会将计数器预设为由 TOCC[TOP] 配置的值并继续递减计数。超时计数器由其中一个 FIFO 控制时，空 FIFO 会将计数器预设为由 TOCC[TOP] 配置的值。存储了第一个 FIFO 元素后，会开始递减计数。

00：连续工作

01：超时由发送事件 FIFO 控制

10：超时由接收 FIFO 0 控制

11：超时由接收 FIFO 1 控制

这些位受写保护 (P)，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能进行写访问。

位 0 **ETOC**：超时计数器使能 (Timeout counter enable)

0：禁止超时计数器

1：使能超时计数器

该位受写保护 (P)，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能进行写访问。

有关更多详细信息，请参见*超时计数器*。

39.4.11 FDCAN 超时计数器值寄存器 (FDCAN_TOCV)

FDCAN timeout counter value register

偏移地址：0x002C

复位值：0x0000 FFFF

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

位 31:16 保留，必须保持复位值。

位 15:0 **TOC[15:0]**：超时计数器 (Timeout counter)

超时计数器会以 CAN 位时间的倍数 [1 ... 16] 递减，具体取决于 TSCC.TCP 的配置。当计数器递减至 0 时，中断标志 IR.TOO 会置 1，超时计数器会停止计数。开始和复位/重启条件是通过 TOCC.TOS 配置的。

39.4.12 FDCAN 错误计数器寄存器 (FDCAN_ECR)

FDCAN error counter register

偏移地址：0x0040

复位值：0x0000 0000

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

位 31:24 保留，必须保持复位值。

位 23:16 **CEL[7:0]**：CAN 错误记录 (CAN error logging)

每当 CAN 协议错误导致发送错误计数器或接收错误计数器递增时，计数器都会递增。对 CEL 进行读访问时，计数器会复位。计数器值达到 0xFF 时，会停止计数；TEC 或 REC 下一次递增时，中断标志 IR[ELO] 会置 1。

访问类型为 RX：进行读访问时复位。

位 15 **RP**：接收错误被动 (Receive error passive)

0：接收错误计数器低于错误被动级别 128。

1：接收错误计数器已达到错误被动级别 128。

位 14:8 **REC[6:0]**：接收错误计数器 (Receive error counter)

接收错误计数器的实际状态，取值范围为 0 到 127。

位 7:0 **TEC[7:0]**：发送错误计数器 (Transmit error counter)
发送错误计数器的实际状态，取值范围为 0 到 255。
如果 CCCR.ASM 置 1，则检测到 CAN 协议错误时，CAN 协议控制器不会使 TEC 和 REC 递增，但 CEL 仍会递增。

39.4.13 FDCAN 协议状态寄存器 (FDCAN_PSR)

FDCAN protocol status register

偏移地址：0x0044

复位值：0x0000 0707

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDCV[6:0]						
									r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PXE	REDL	RBRS	RESI	DLEC[2:0]			BO	EW	EP	ACT[1:0]		LEC[2:0]		
	rc_r	rc_r	rc_r	rc_r	rs	rs	rs	r	r	r	r	r	rs	rs	rs

位 31:23 保留，必须保持复位值。

位 22:16 **TDCV[6:0]**：发送器延迟补偿值 (Transmitter delay compensation value)
第二采样点的位置，由从 FDCAN_TX 到 FDCAN_RX 之间测得的延迟与 TDCR.TDCO 之和定义。在数据阶段，SSP 位置为已发送位起点与第二采样点之间的最小时间片 (mtq) 数。有效值为 0 到 127 个 mtq。

位 15 保留，必须保持复位值。

位 14 **PXE**：协议异常事件 (Protocol exception event)
0：自上次读访问起未发生协议异常事件
1：已发生协议异常事件

位 13 **REDL**：已接收 FDCAN 消息 (Received FDCAN message)
该位的设置与验收过滤无关。
0：由于该位是通过 CPU 复位的，因此尚未接收到 FDCAN 消息。
1：已接收到 EDL 标志置 1 的 FDCAN 格式的消息。
访问类型为 RX：进行读访问时复位。

位 12 **RBRS**：上次接收的 FDCAN 消息的 BRS 标志 (BRS flag of last received FDCAN message)
该位与 REDL 一起设置，与验收过滤无关。
0：上次接收的 FDCAN 消息的 BRS 标志未置 1。
1：上次接收的 FDCAN 消息的 BRS 标志已置 1。
访问类型为 RX：进行读访问时复位。

位 11 **RESI**：上次接收的 FDCAN 消息的 ESI 标志 (ESI flag of last received FDCAN message)
该位与 REDL 一起设置，与验收过滤无关。
0：上次接收的 FDCAN 消息的 ESI 标志未置 1。
1：上次接收的 FDCAN 消息的 ESI 标志已置 1。
访问类型为 RX：进行读访问时复位。

位 10:8 **DLEC[2:0]**：数据上一次错误代码 (Data last error code)
在 BRS 标志置 1 的 FDCAN 格式帧数据阶段发生的上一次错误类型。编码与 LEC 相同。当 BRS 标志置 1 的 FDCAN 格式帧已无错传输（接收或发送）时，该位域将被清零。
访问类型为 RS：进行读访问时置 1。

位 7 **BO**：Bus_Off 状态 (Bus_Off status)

0：FDCAN 未处于 Bus_Off 状态。

1：FDCAN 处于 Bus_Off 状态。

位 6 **EW**：警告状态 (Warning status)

0：两个错误计数器的值均小于 Error_Warning 限值 96。

1：至少有一个错误计数器已达到 Error_Warning 限值 96。

位 5 **EP**：错误被动 (Error passive)

0：FDCAN 处于 Error_Active 状态。该位通常会参与总线通信，并会在检测到错误后发送主动错误标志。

1：FDCAN 处于 Error_Passive 状态。

位 4:3 **ACT[1:0]**：活动 (Activity)

监控模块的 CAN 通信状态。

00：同步中：节点在 CAN 通信时同步。

01：空闲：节点既不是接收器，也不是发送器。

10：接收器：节点作为接收器工作。

11：发送器：节点作为发送器工作。

位 2:0 **LEC[2:0]**：上一个错误代码 (Last error code)

LEC 指示 CAN 总线上发生的上一个错误的类型。消息已无错传输后（接收或发送），该位域将被清零。

000：无错：自消息成功接收或发送而将 LEC 复位后，未发生任何错误。

001：内容错误：在已接收的消息中，有一部分连续出现 5 个以上的相等位，这种情况是不允许发生的。

010：格式错误：已接收帧的固定格式部分的格式不正确。

011：AckError：由 FDCAN 发送的消息未被另一节点确认。

100：Bit1Error：在消息发送过程中（仲裁字段例外），器件希望发送隐性电平（位逻辑值为 1），但受监控的总线值为显性。

101：Bit0Error：在消息发送过程中（或者确认位，或者主动错误标志，或者过载标志），器件希望发送显性电平（数据或标识符位逻辑值 0），但受监控的总线值为隐性。在 Bus_Off 恢复期间，每次监控到 11 个隐性位组成的序列时，此状态都会置 1。这样，CPU 便可监控 Bus_Off 恢复序列的进行（指示总线并未保持显性或持续受到干扰）。

110：CRCError：已接收消息的 CRC 校验和不正确。传入消息的 CRC 与通过已接收数据计算出的 CRC 不匹配。

111：NoChange：如果对协议状态寄存器进行读访问，则会将 LEC 重新初始化为“7”。如果 LEC 显示的值为“7”，则自 CPU 上一次对协议状态寄存器进行读访问起，未检测到任何 CAN 总线事件。

访问类型为 RS：进行读访问时置 1。

注意： *如果采用 FDCAN 格式的帧已到达 BRS 标志置 1 的数据阶段，下一 CAN 事件（错误或有效帧）将显示在 FLEC 中，而不显示在 LEC 中。FDCAN CRC 序列中的固定内容位中的错误将显示为格式错误，而不是内容错误。*

注意： *无法通过将 CCCR[INIT] 置 1 或复位来缩短 Bus_Off 恢复序列（请参见 CAN 规范第 2.0 版或 ISO11898-1）。如果器件进入 Bus_Off 状态，则会将其自身的 CCCR.INIT 置 1，从而会停止所有总线活动。一旦 CCCR[INIT] 由 CPU 清零，器件随后将等待总线空闲状态出现 129 次（129 × 11 个连续隐性位），然后才会恢复正常操作。Bus_Off 恢复序列结束时，错误管理计数器将复位。在 CCCR[INIT] 复位后的等待时间内，每次监控到由 11 个隐性位组成的序列时，Bit0 错误代码都会写入 PSR[LEC] 中，从而使 CPU 能够准备好检查 CAN 总线保持显性还是持续受到干扰，并能够监控 Bus_Off 恢复序列。ECR[REC] 用于对这些序列进行计数。*

39.4.14 FDCAN 发送器延迟补偿寄存器 (FDCAN_TDCR)

FDCAN transmitter delay compensation register

偏移地址：0x0048

复位值：0x0000 0000

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

位 31:15 保留，必须保持复位值。

位 14:8 **TDCO[6:0]**：发送器延迟补偿偏移 (Transmitter delay compensation offset)

定义从 FDCAN_TX 到 FDCAN_RX 测得的延迟与第二采样点之间距离的偏移值。有效值为 0 到 127 个 mtq。

这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

位 7 保留，必须保持复位值。

位 6:0 **TDCF[6:0]**：发送器延迟补偿过滤器窗口长度 (Transmitter delay compensation filter window length)

定义 SSP 位置的最小值，要测量发送器延迟，会忽略 FDCAN_RX 上会导致 SSP 位置提前的显性边沿。

这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

39.4.15 FDCAN 中断寄存器 (FDCAN_IR)

FDCAN interrupt register

如果检测到其中一个列出的条件（边沿有效），标志会置 1。在主机将标志清零之前，标志保持置 1 状态。通过向对应位位置写入 1 将标志清零。

写入 0 不会起任何作用。硬复位会将寄存器清零。IE 的配置控制着是否生成中断。ILS 的配置控制着是哪条中断线发出中断。

偏移地址：0x0050

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARA	PED	PEA	WDI	BO	EW	EP	ELO
								rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOO	MRAF	TSW	TEFL	TEFF	TEFN	TFE	TCF	TC	HPM	RF1L	RF1F	RF1N	RF0L	RF0F	RF0N
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

位 31:24 保留，必须保持复位值。

位 23 **ARA**：访问保留地址 (Access to reserved address)

0：未对保留地址进行访问

1：已对保留地址进行访问

位 22 **PED**：数据阶段的协议错误（使用数据位时间）(Protocol error in data phase (data bit time is used))

0：数据阶段没有协议错误

1：检测到数据阶段有协议错误（PSR.DLEC 不是 0、7）

位 21 **PEA**：仲裁阶段中的协议错误（使用标称位时间）(Protocol error in arbitration phase (nominal bit time is used))

0：仲裁阶段中没有协议错误

1：检测到仲裁阶段有协议错误（PSR.LEC 不是 0、7）

位 20 **WDI**：看门狗中断 (Watchdog interrupt)

0：未发生消息 RAM 看门狗事件

1：因 READY 缺失而发生消息 RAM 看门狗事件

位 19 **BO**：Bus_Off 状态 (Bus_Off status)

0：Bus_Off 状态未更改

1：Bus_Off 状态已更改

位 18 **EW**：警告状态 (Warning status)

0：Error_Warning 状态未更改

1：Error_Warning 状态已更改

位 17 **EP**：错误被动 (Error passive)

0：Error_Passive 状态未更改

1：Error_Passive 状态已更改

位 16 **ELO**：错误记录溢出 (Error logging overflow)

0：CAN 错误记录计数器未溢出

1：CAN 错误记录计数器已溢出

位 15 **TOO**：发生超时 (Timeout occurred)

0：无超时

1：达到超时

位 14 **MRAF**：消息 RAM 访问失败 (Message RAM access failure)

如果接收处理单元未在已接收以下消息的仲裁字段之前完成对已接收消息的验收过滤或存储，

- 此标志会置 1。在这种情况下，验收过滤或消息存储会中止，接收处理单元会开始处理以下消息。
- 无法向消息 RAM 写入消息。在这种情况下，会中止消息存储。

在这两种情况下，FIFO 放入索引不会更新。下一条消息存储到该位置时，部分存储的消息会被覆盖。

当发送处理单元无法及时从消息 RAM 中读取数据时，此标志也会置 1。在这种情况下，消息发送会中止。如果发送处理单元访问失败，FDCAN 会切换到受限工作模式（请参见*受限工作模式*）。要退出受限工作模式，主机 CPU 必须复位 CCCR.ASM。

0：未发生消息 RAM 访问失败

1：发生消息 RAM 访问失败

位 13 **TSW**：时间戳回卷 (Timestamp wraparound)

0：时间戳未回卷

1：时间戳已回卷

位 12 **TEFL**：发送事件 FIFO 元素丢失 (Tx event FIFO element lost)
0：发送事件 FIFO 元素未丢失
1：发送事件 FIFO 元素已丢失

位 11 **TEFF**：发送事件 FIFO 已满 (Tx event FIFO full)
0：发送事件 FIFO 未满
1：发送事件 FIFO 已满

位 10 **TEFN**：发送事件 FIFO 新条目 (Tx event FIFO New Entry)
0：发送事件 FIFO 未更改
1：发送处理单元写入发送事件 FIFO 元素

位 9 **TFE**：发送 FIFO 为空 (Tx FIFO empty)
0：发送 FIFO 非空
1：发送 FIFO 为空

位 8 **TCF**：发送取消完成 (Transmission cancellation finished)
0：发送取消未完成
1：发送取消已完成

位 7 **TC**：发送完成 (Transmission completed)
0：发送未完成
1：发送已完成

位 6 **HPM**：高优先级消息 (High-priority message)
0：未接收到任何高优先级消息
1：接收到高优先级消息

位 5 **RF1L**：接收 FIFO 1 消息丢失 (Rx FIFO 1 message lost)
0：接收 FIFO 1 消息未丢失
1：接收 FIFO 1 消息已丢失

位 4 **RF1F**：接收 FIFO 1 已满 (Rx FIFO 1 full)
0：接收 FIFO 1 未满
1：接收 FIFO 1 已满

位 3 **RF1N**：接收 FIFO 1 新消息 (Rx FIFO 1 new message)
0：未向接收 FIFO 1 写入新消息
1：已向接收 FIFO 1 写入新消息

位 2 **RF0L**：接收 FIFO 0 消息丢失 (Rx FIFO 0 message lost)
0：接收 FIFO 0 消息未丢失
1：接收 FIFO 0 消息已丢失

位 1 **RF0F**：接收 FIFO 0 已满 (Rx FIFO 0 full)
0：接收 FIFO 0 未满
1：接收 FIFO 0 已满

位 0 **RF0N**：接收 FIFO 0 新消息 (Rx FIFO 0 new message)
0：未向接收 FIFO 0 写入新消息
1：已向接收 FIFO 0 写入新消息

39.4.16 FDCAN 中断使能寄存器 (FDCAN_IE)

FDCAN interrupt enable register

中断使能寄存器中的设置决定了将在中断线上指示中断寄存器中的哪些状态更改。

偏移地址：0x0054

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARAE	PEDE	PEAE	WDIE	BOE	EWE	EPE	ELOE
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOOE	MRAFE	TSWE	TEFLE	TEFFE	TEFNE	TFEE	TCFE	TCE	HPME	RF1LE	RF1FE	RF1NE	RF0LE	RF0FE	RF0NE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23 **ARAE**：访问保留地址使能 (Access to reserved address enable)

位 22 **PEDE**：数据阶段的协议错误使能 (Protocol error in data phase enable)

位 21 **PEAE**：仲裁阶段中的协议错误使能 (Protocol error in arbitration phase enable)

位 20 **WDIE**：看门狗中断使能 (Watchdog interrupt enable)

0：禁止中断
1：使能中断

位 19 **BOE**：Bus_Off 状态 (Bus_Off status)

0：禁止中断
1：使能中断

位 18 **EWE**：警告状态中断使能 (Warning status interrupt enable)

0：禁止中断
1：使能中断

位 17 **EPE**：错误被动中断使能 (Error passive interrupt enable)

0：禁止中断
1：使能中断

位 16 **ELOE**：错误记录溢出中断使能 (Error logging overflow interrupt enable)

0：禁止中断
1：使能中断

位 15 **TOOE**：超时已发生中断使能 (Timeout occurred interrupt enable)

0：禁止中断
1：使能中断

位 14 **MRAFE**：消息 RAM 访问失败中断使能 (Message RAM access failure interrupt enable)

0：禁止中断
1：使能中断

位 13 **TSWE**：时间戳回卷中断使能 (Timestamp wraparound interrupt enable)

0：禁止中断
1：使能中断

位 12 **TEFLE**：发送事件 FIFO 元素丢失中断使能 (Tx event FIFO element lost interrupt enable)
0：禁止中断
1：使能中断

位 11 **TEFFE**：发送事件 FIFO 已满中断使能 (Tx event FIFO full interrupt enable)
0：禁止中断
1：使能中断

位 10 **TEFNE**：发送事件 FIFO 新条目中断使能 (Tx event FIFO new entry interrupt enable)
0：禁止中断
1：使能中断

位 9 **TFEE**：Tx FIFO 为空时中断使能 (Tx FIFO empty interrupt enable)
0：禁止中断
1：使能中断

位 8 **TCFE**：发送取消完成中断使能 (Transmission cancellation finished interrupt enable)
0：禁止中断
1：使能中断

位 7 **TCE**：发送完成中断使能 (Transmission completed interrupt enable)
0：禁止中断
1：使能中断

位 6 **HPME**：高优先级消息中断使能 (High-priority message interrupt enable)
0：禁止中断
1：使能中断

位 5 **RF1LE**：接收 FIFO 1 消息丢失中断使能 (Rx FIFO 1 message lost interrupt enable)
0：禁止中断
1：使能中断

位 4 **RF1FE**：接收 FIFO 1 已满中断使能 (Rx FIFO 1 full interrupt enable)
0：禁止中断
1：使能中断

位 3 **RF1NE**：接收 FIFO 1 新消息中断使能 (Rx FIFO 1 new message interrupt enable)
0：禁止中断
1：使能中断

位 2 **RF0LE**：接收 FIFO 0 消息丢失中断使能 (Rx FIFO 0 message lost interrupt enable)
0：禁止中断
1：使能中断

位 1 **RF0FE**：接收 FIFO 0 已满中断使能 (Rx FIFO 0 full interrupt enable)
0：禁止中断
1：使能中断

位 0 **RF0NE**：接收 FIFO 0 新消息中断使能 (Rx FIFO 0 new message interrupt enable)
0：禁止中断
1：使能中断

39.4.17 FDCAN 中断线选择寄存器 (FDCAN_ILS)

FDCAN interrupt line select register

该寄存器将特定组中断标志生成的中断从中断寄存器分配到两个模块中断线之一。要生成中断，必须通过 ILE[EINT0] 和 ILE[EINT1] 使能相应的中断线。

偏移地址：0x0058

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERR	BERR	MISC	TFERR	SMSG	RXFIFO1	RXFIFO0
									rw	rw	rw	rw	rw	rw	rw

位 31:7 保留，必须保持复位值。

位 6 **PERR**：将以下中断分组给协议错误 (Protocol error grouping the following interruption)
- ARAL：访问保留地址线
- PEDL：数据阶段的协议错误线
- PEAL：仲裁阶段的协议错误线
- WDIL：看门狗中断线
- BOL：Bus_Off 状态
- EWL：警告状态中断线

位 5 **BERR**：将以下中断分组给位和线错误 (Bit and line error grouping the following interruption)
- EPL：错误被动中断线
- ELOL：错误记录溢出中断线

位 4 **MISC**：将以下中断重新分组给下面的中断 (Interrupt regrouping the following interruption)
- TOOL：超时已发生中断线
- MRAFL：消息 RAM 访问失败中断线
- TSWL：时间戳回卷中断线

位 3 **TFERR**：将以下中断分组给发送 FIFO 错误 (Tx FIFO ERROR grouping the following interruption)
- TEFLL：发送事件 FIFO 元素丢失中断线
- TEFFL：发送事件 FIFO 已满中断线
- TEFNL：发送事件 FIFO 新条目中断线
- TFEL：发送 FIFO 为空中断线

位 2 **SMSG**：将以下中断分组给状态消息位 (Status message bit grouping the following interruption)
- TCFL：发送取消完成中断线
- TCL：发送完成中断线
- HPML：高优先级消息中断线

位 1 **RXFIFO1**：将以下中断分组给接收 FIFO 位 (RX FIFO bit grouping the following interruption)
- RF1LL：接收 FIFO 1 消息丢失中断线
- RF1FL：接收 FIFO 1 已满中断线
- RF1NL：接收 FIFO 1 新消息中断线

位 0 **RXFIFO0**：将以下中断分组给接收 FIFO 位 (RX FIFO bit grouping the following interruption)
RF0LL：接收 FIFO 0 消息丢失中断线
RF0FL：接收 FIFO 0 已满中断线
RF0NL：接收 FIFO 0 新消息中断线

39.4.18 FDCAN 中断线使能寄存器 (FDCAN_ILE)

FDCAN interrupt line enable register

通过对位 EINT0 和 EINT1 进行编程，可分别使能/禁止两条连接至 CPU 的中断线。

偏移地址：0x005C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EINT1	EINT0
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **EINT1**：使能中断线 1 (Enable interrupt line 1)
0：禁止中断线 fdcan_intr0_it
1：使能中断线 fdcan_intr0_it

位 0 **EINT0**：使能中断线 0 (Enable interrupt line 0)
0：禁止中断线 fdcan_intr1_it
1：使能中断线 fdcan_intr1_it

39.4.19 FDCAN 全局过滤器配置寄存器 (FDCAN_RXGFC)

FDCAN global filter configuration register

用于消息 ID 过滤的全局设置。全局过滤器配置控制着标准和扩展消息的过滤器路径，请参见*图 507* 和*图 508*。

偏移地址：0x0080

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	LSE[3:0]				Res.	Res.	Res.	LSS[4:0]				
				rw	rw	rw	rw				rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	F0OM	F1OM	Res.	Res.	ANFS[1:0]		ANFE[1:0]		RRFS	RRFE
						rw	rw			rw	rw	rw	rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27:24 **LSE[3:0]**：扩展列表大小 (List size extended)
0：无扩展消息 ID 过滤器
1 到 8：扩展消息 ID 过滤器元素数量
>8：大于 8 的值会被解析为 8。
这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

位 23:21 保留，必须保持复位值。

位 20:16 **LSS[4:0]**：标准列表大小 (List size standard)
0：无标准消息 ID 过滤器
1 到 28：标准消息 ID 过滤器元素数量
>28：大于 28 的值会被解析为 28。
这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

位 15:10 保留，必须保持复位值。

位 9 **F0OM**：FIFO 0 工作模式（覆盖或阻止）(FIFO 0 operation mode (overwrite or blocking))
该位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

位 8 **F1OM**：FIFO 1 工作模式（覆盖或阻止）(FIFO 1 operation mode (overwrite or blocking))
该位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

位 7:6 保留，必须保持复位值。

位 5:4 **ANFS[1:0]**：接受非匹配标准帧 (Accept Non-matching frames standard)
定义了如何处理与过滤器列表的任何元素都不匹配的 ID 为 11 位的已接收消息。
00：在接收 FIFO 0 中接受
01：在接收 FIFO 1 中接受
10：拒绝
11：拒绝
这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

位 3:2 **ANFE[1:0]**：接受非匹配扩展帧 (Accept non-matching frames extended)
定义了如何处理与过滤器列表的任何元素都不匹配的 ID 为 29 位的已接收消息。
00：在接收 FIFO 0 中接受
01：在接收 FIFO 1 中接受
10：拒绝
11：拒绝
这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

位 1 **RRFS**：拒绝标准远程帧 (Reject remote frames standard)
0：过滤采用 11 位标准 ID 的远程帧
1：拒绝所有采用 11 位标准 ID 的远程帧
这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

位 0 **RRFE**：拒绝扩展远程帧 (Reject remote frames extended)
0：过滤采用 29 位标准 ID 的远程帧
1：拒绝所有采用 29 位标准 ID 的远程帧
这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

39.4.20 FDCAN 扩展 ID 和掩码寄存器 (FDCAN_XIDAM)

FDCAN extended ID and mask register

偏移地址：0x0084

复位值：0x1FFF FFFF

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

位 31:29 保留，必须保持复位值。

位 28:0 **EIDM[28:0]**：扩展 ID 掩码 (Extended ID mask)

要对扩展帧进行验收过滤，会将扩展 ID 与掩码与已接收帧的消息 ID 进行与运算。用于屏蔽 SAE J1939 中的 29 位 ID。如果所有位的复位值均设为 1，则掩码无效。

这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

39.4.21 FDCAN 高优先级消息状态寄存器 (FDCAN_HPMS)

FDCAN high-priority message status register

每次消息 ID 过滤器元素配置为生成优先级事件匹配时，都会更新此寄存器。此寄存器可用于监控传入高优先级消息的状态，并可使能对这些消息的快速访问。

偏移地址：0x0088

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15 **FLST**：过滤器列表 (Filter list)

指示匹配过滤器元素的过滤器列表。

0：标准过滤器列表

1：扩展过滤器列表

位 14:13 保留，必须保持复位值。

位 12:8 **FIDX[4:0]**：过滤器索引 (Filter index)

匹配过滤器元素的索引。范围为 0 到 RXGFC[LSS] - 1 或 RXGFC[LSE] - 1。

位 7:6 **MSI[1:0]**：消息存储指示符 (Message storage indicator)
00：未选择 FIFO
01：FIFO 上溢
10：消息存储在 FIFO 0 中
11：消息存储在 FIFO 1 中

位 5:3 保留，必须保持复位值。

位 2:0 **BIDX[2:0]**：缓冲区索引
消息存储到的接收 FIFO 元素的索引。仅当 MSI[1] = 1 时有效。

39.4.22 FDCAN 接收 FIFO 0 状态寄存器 (FDCAN_RXF0S)

FDCAN Rx FIFO 0 status register

偏移地址：0x0090

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	RF0L	F0F	Res.	Res.	Res.	Res.	Res.	Res.	F0PI[1:0]	
						r	r							r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	F0GI[1:0]		Res.	Res.	Res.	Res.	F0FL[3:0]			
						r	r					r	r	r	r

位 31:26 保留，必须保持复位值。

位 25 **RF0L**：接收 FIFO 0 消息丢失 (Rx FIFO 0 message lost)
该位是中断标志 IR[RF0L] 的副本。当 IR[RF0L] 复位时，该位也会复位。
0：接收 FIFO 0 消息未丢失
1：接收 FIFO 0 消息已丢失，尝试向大小为 0 的接收 FIFO 0 进行写入后也会置 1

位 24 **F0F**：接收 FIFO 0 已满 (Rx FIFO 0 full)
0：接收 FIFO 0 未满
1：接收 FIFO 0 已满

位 23:18 保留，必须保持复位值。

位 17:16 **F0PI[1:0]**：接收 FIFO 0 放入索引 (Rx FIFO 0 put index)
接收 FIFO 0 写入索引指针，范围为 0 到 2。

位 15:10 保留，必须保持复位值。

位 9:8 **F0GI[1:0]**：接收 FIFO 0 获取索引 (Rx FIFO 0 get index)
接收 FIFO 0 读取索引指针，范围为 0 到 2。

位 7:4 保留，必须保持复位值。

位 3:0 **F0FL[3:0]**：Rx FIFO 0 填充级别 (Rx FIFO 0 fill level)
接收 FIFO 0 中存储的元素数，范围为 0 到 3。

39.4.23 CAN 接收 FIFO 0 确认寄存器 (FDCAN_RXF0A)

CAN Rx FIFO 0 acknowledge register

偏移地址：0x0094

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	F0AI[2:0]		
													rw	rw	rw

位 31:3 保留，必须保持复位值。

位 2:0 **F0AI[2:0]**：接收 FIFO 0 确认索引 (Rx FIFO 0 acknowledge index)

主机从接收 FIFO 0 读取消息或消息序列后，必须将从接收 FIFO 0 读取的最后一个元素的缓冲区索引写入 F0AI 中。此操作会将接收 FIFO 0 获取索引 RXF0S[F0GI] 设为 F0AI + 1，并会更新 FIFO 0 填充级别 RXF0S[F0FL]。

39.4.24 FDCAN 接收 FIFO 1 状态寄存器 (FDCAN_RXF1S)

FDCAN Rx FIFO 1 status register

偏移地址：0x0098

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	RF1L	F1F	Res.	Res.	Res.	Res.	Res.	Res.	F1PI[1:0]	
						r	r							r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	F1GI[1:0]		Res.	Res.	Res.	Res.	F1FL[3:0]			
						r	r					r	r	r	r

位 31:26 保留，必须保持复位值。

位 25 **RF1L**：接收 FIFO 1 消息丢失 (Rx FIFO 1 message lost)

该位是中断标志 IR[RF1L] 的副本。当 IR[RF1L] 复位时，该位也会复位。

0：接收 FIFO 1 消息未丢失

1：接收 FIFO 1 消息已丢失，尝试向大小为 0 的接收 FIFO 1 进行写入后也会置 1

位 24 **F1F**：接收 FIFO 1 已满 (Rx FIFO 1 full)

0：接收 FIFO 1 未满

1：接收 FIFO 1 已满

位 23:18 保留，必须保持复位值。

位 17:16 **F1PI[1:0]**：接收 FIFO 1 放入索引 (Rx FIFO 1 put index)

接收 FIFO 1 写入索引指针，范围为 0 到 2。

位 15:10 保留，必须保持复位值。

位 9:8 **F1GI[1:0]**：接收 FIFO 1 获取索引 (Rx FIFO 1 get index)
接收 FIFO 1 读取索引指针，范围为 0 到 2。

位 7:4 保留，必须保持复位值。

位 3:0 **F1FL[3:0]**：Rx FIFO 1 填充级别 (Rx FIFO 1 fill level)
接收 FIFO 1 中存储的元素数，范围为 0 到 3。

39.4.25 FDCAN 接收 FIFO 1 确认寄存器 (FDCAN_RXF1A)

FDCAN Rx FIFO 1 acknowledge register

偏移地址：0x009C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	F1AI[2:0]		
													rw	rw	rw

位 31:3 保留，必须保持复位值。

位 2:0 **F1AI[2:0]**：接收 FIFO 1 确认索引 (Rx FIFO 1 acknowledge index)
主机从接收 FIFO 1 读取消息或消息序列后，必须将从接收 FIFO 1 读取的最后一个元素的缓冲区索引写入 F1AI 中。此操作会将接收 FIFO 1 获取索引 RXF1S[F1GI] 设为 F1AI + 1，并会更新 FIFO 1 填充级别 RXF1S[F1FL]。

39.4.26 FDCAN 发送缓冲区配置寄存器 (FDCAN_TXBC)

FDCAN Tx buffer configuration register

偏移地址：0x00C0

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TFQM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
							rw								

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

位 31:25 保留，必须保持复位值。

位 24 **TFQM**：发送 FIFO/队列模式 (Tx FIFO/queue mode)
0：发送 FIFO 操作
1：发送队列操作
该位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

位 23:0 保留，必须保持复位值。

39.4.27 FDCAN 发送 FIFO/队列状态寄存器 (FDCAN_TXFQS)

FDCAN Tx FIFO/queue status register

Tx FIFO/队列状态与寄存器 TXBRP 中列出的挂起发送请求相关。因此，添加/取消请求的作用可能因正在运行发送扫描而延迟（TXBRP 尚未更新）。

偏移地址：0x00C4

复位值：0x0000 0003

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TFQF	Res.	Res.	Res.	TFQPI[1:0]	
										r				r	r

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

位 31:22 保留，必须保持复位值。

位 21 **TFQF**：发送 FIFO/队列已满 (Tx FIFO/queue full)
0：发送 FIFO/队列未满
1：发送 FIFO/队列已满

位 20:18 保留，必须保持复位值。

位 17:16 **TFQPI[1:0]**：发送 FIFO/队列放入索引 (Tx FIFO/queue put index)
发送 FIFO/队列写入索引指针，范围为 0 到 3

位 15:10 保留，必须保持复位值。

位 9:8 **TFGI[1:0]**：发送 FIFO 获取索引 (Tx FIFO get index)
发送 FIFO 读取索引指针，范围为 0 到 3如果已配置发送队列操作（TXBC.TFQM = 1），则读为 0

位 7:3 保留，必须保持复位值。

位 2:0 **TFFL[2:0]**：发送 FIFO 空闲级别 (Tx FIFO free level)
从 TFGI 开始的连续空闲发送 FIFO 元素数，范围为 0 到 3。如果已配置发送队列操作（TXBC[TFQM] = 1），则读为 0。

39.4.28 FDCAN 发送缓冲区请求挂起寄存器 (FDCAN_TXBRP)

FDCAN Tx buffer request pending register

偏移地址：0x00C8

复位值：0x0000 0000

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

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

位 31:3 保留，必须保持复位值。

位 2:0 **TRP[2:0]**：发送请求挂起 (Transmission request pending)

每个发送缓冲区都有自己的发送请求挂起位。这些位通过寄存器 TXBAR 置 1。请求的发送已完成或已通过寄存器 TXBCR 取消后，这些位会复位。

TXBRP 位置 1 后，会启动发送扫描以检查优先级最高（消息 ID 最小的发送缓冲区）的挂起发送请求。

取消请求会使寄存器 TXBRP 的相应发送请求挂起位复位。如果请求取消时发送已开始进行，则无论发送是否成功，都会在发送结束时执行复位操作。相应的 TXBRP 位复位后，取消请求位也会立即复位。

在发出取消请求后，成功传输后通过 TXBCF 信号完成取消，并与相应的 TXBTO 位一起传输。

取消时发送尚未开始

发送因仲裁丢失而中止

帧发送过程中出错

在 DAR 模式下，所有发送在发送失败后都会自动取消。对于所有未成功的发送，相应的 TXBCF 位会置 1。

0：无发送请求挂起

1：发送请求挂起

注意： *如果 TXBRP 位在正在进行发送扫描时置 1，则进行该特殊发送扫描期间不会考虑该位。如果请求取消此类发送缓冲区，则该添加请求会立即取消，相应的 TXBRP 位会复位。*

39.4.29 FDCAN 发送缓冲区添加请求寄存器 (FDCAN_TXBAR)

FDCAN Tx buffer add request register

偏移地址：0x00CC

复位值：0x0000 0000

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

位 31:3 保留，必须保持复位值。

位 2:0 **AR[2:0]**：添加请求 (Add request)

每个发送缓冲区都有自己的添加请求位。写入 1 会将相应的添加请求位置 1；写入 0 没有影响。这样，主机通过对 TXBAR 进行一次写操作便可为多个发送缓冲区设置发送请求。如果没有运行发送扫描，这些位会立即复位，否则在发送扫描过程完成之前，这些位保持置 1 状态。

0：未添加发送请求

1：已添加发送请求。

注意： *如果添加请求应用到具有挂起发送请求的发送缓冲区（相应的 TXBRP 位已置 1），则会忽略请求。*

39.4.30 FDCAN 发送缓冲区取消请求寄存器 (FDCAN_TXBCR)

FDCAN Tx buffer cancellation request register

偏移地址：0x00D0

复位值：0x0000 0000

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

位 31:3 保留，必须保持复位值。

位 2:0 **CR[2:0]**：取消请求 (Cancellation request)

每个发送缓冲区都有自己的取消请求位。写入 1 会将相应的 CR 位置 1；写入 0 没有影响。这样，主机通过对 TXBCR 进行一次写操作便可为多个发送缓冲区设置取消请求。在相应的 TXBRP 位复位之前，这些位保持置 1 状态。

0：无取消挂起

1：取消挂起

39.4.31 FDCAN 发送缓冲区发送已发生寄存器 (FDCAN_TXBTO)

FDCAN Tx buffer transmission occurred register

偏移地址：0x00D4

复位值：0x0000 0000

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

位 31:3 保留，必须保持复位值。

位 2:0 **TO[2:0]**：发送已发生 (Transmission occurred)

每个发送缓冲区都有自己的 TO 位。当相应的 TXBRP 位在发送成功后清零时，这些位会置 1。通过向寄存器 TXBAR 的相应位写入 1 请求进行新发送时，这些位会复位。

0：未进行发送

1：已进行发送

39.4.32 FDCAN 发送缓冲区取消完成寄存器 (FDCAN_TXBCF)

FDCAN Tx buffer cancellation finished register

偏移地址：0x00D8

复位值：0x0000 0000

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

位 31:3 保留，必须保持复位值。

位 2:0 **CF[2:0]**：取消完成 (Cancellation finished)

每个发送缓冲区都有自己的 CF 位。当相应的 TXBRP 位在通过 TXBCR 发出请求取消后清零时，这些位会置 1。如果相应的 TXBRP 位在取消时未置 1，CF 会立即置 1。通过向寄存器 TXBAR 的相应位写入 1 请求进行新发送时，这些位会复位。

0：无发送缓冲区取消

1：发送缓冲区取消完成

39.4.33 FDCAN 发送缓冲区发送中断使能寄存器 (FDCAN_TXBTIE)

FDCAN Tx buffer transmission interrupt enable register

偏移地址：0x00DC

复位值：0x0000 0000

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

位 31:3 保留，必须保持复位值。

位 2:0 **TIE[2:0]**：发送中断使能 (Transmission interrupt enable)

每个发送缓冲区都有自己的 TIE 位。

0：禁止发送中断

1：使能发送中断

39.4.34 FDCAN 发送缓冲区取消完成中断使能寄存器 (FDCAN_TXBCIE)

FDCAN Tx buffer cancellation finished interrupt enable register

偏移地址：0x00E0

复位值：0x0000 0000

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

位 31:3 保留，必须保持复位值。

位 2:0 **CFIE[2:0]**：取消完成中断使能 (Cancellation finished interrupt enable)
每个发送缓冲区都有自己的 CFIE 位。
0：禁止取消完成中断
1：使能取消完成中断

39.4.35 FDCAN 发送事件 FIFO 状态寄存器 (FDCAN_TXEFS)

FDCAN Tx event FIFO status register

偏移地址：0x00E4

复位值：0x0000 0000

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

位 31:26 保留，必须保持复位值。

位 25 **TEFL**：发送事件 FIFO 元素丢失 (Tx event FIFO element lost)
该位是中断标志 IR[TEFL] 的副本。当 IR[TEFL] 复位时，该位也会复位。
0：发送事件 FIFO 元素未丢失
1：发送事件 FIFO 元素已丢失，尝试向大小为 0 的发送事件 FIFO 进行写入时也会置 1。

位 24 **EFF**：事件 FIFO 已满 (Event FIFO full)
0：发送事件 FIFO 未满
1：发送事件 FIFO 已满

位 23:18 保留，必须保持复位值。

位 17:16 **EFPI[1:0]**：事件 FIFO 放入索引 (Event FIFO put index)
发送事件 FIFO 写入索引指针，范围为 0 到 3。

位 15:10 保留，必须保持复位值。

位 9:8 **EFGI[1:0]**：事件 FIFO 获取索引 (Event FIFO get index)
发送事件 FIFO 读取索引指针，范围为 0 到 3。

位 7:3 保留，必须保持复位值。

位 2:0 **EFFL[2:0]**：事件 FIFO 填充级别 (Event FIFO fill level)
发送事件 FIFO 中存储的元素数，范围为 0 到 3。

39.4.36 FDCAN 发送事件 FIFO 确认寄存器 (FDCAN_TXEFA)

FDCAN Tx event FIFO acknowledge register

偏移地址：0x00E8

复位值：0x0000 0000

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

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

位 31:2 保留，必须保持复位值。

位 1:0 **EFAI[1:0]**：事件 FIFO 确认索引 (Event FIFO acknowledge index)
主机从发送事件 FIFO 读取元素或元素序列后，必须将从发送事件 FIFO 读取的最后一个元素的索引写入 EFAI 中。此操作会将发送事件 FIFO 获取索引 TXEFS[EFGI] 设为 EFAI + 1，并会更新 FIFO 0 填充级别 TXEFS[EFFL]。

39.4.37 FDCAN CFG 时钟分频寄存器（FDCAN_CKDIV)

FDCAN CFG clock divider register

偏移地址：0x0100

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **PDIV[3:0]**：输入时钟分频器 (input clock divider)

APB 时钟可以先进行分频，然后再提供给 CAN 子系统使用。必须使用分频器输出时钟来计算速率。

0000：1 分频
0001：2 分频
0010：4 分频
0011：6 分频
0100：8 分频
0101：10 分频
0110：12 分频
0111：14 分频
1000：16 分频
1001：18 分频
1010：20 分频
1011：22 分频
1100：24 分频
1101：26 分频
1110：28 分频
1111：30 分频

这些位受写保护 (P)，也就是说，仅当 CCCR 寄存器的位 1 [CCE] 和位 0 [INIT] 置 1 时，才能通过这些位进行写访问。

39.4.38 FDCAN 寄存器映射

表 329. FDCAN 寄存器映射和复位值(1)

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0000	FDCAN_CREL	REL[3:0]				STEP[3:0]				SUBSTEP [3:0]				YEAR[3:0]				MON[7:0]								DAY[7:0]							
	Reset value	0	0	0	0	0	0	0	1	1	1	1	0	1	0	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0
0x0004	FDCAN_ENDN	ETV[31:0]																															
	Reset value	1	0	0	0	0	1	1	1	0	1	1	0	0	1	0	1	0	1	0	0	0	0	1	1	0	0	1	0	0	0	0	1
0x0008	Reserved	Reserved																															
0x000C	FDCAN_DBTP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDC	Res.	Res.	DBRP[4:0]					Res.	Res.	Res.	DTSEG1[4:0]					DTSEG2 [3:0]				DSJW[3:0]			
	Reset value									0	0	0	0	0	0	0	0				0	1	0	1	0	0	0	1	1	0	0	1	1
0x0010	FDCAN_TEST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RX	TX [1:0]		LBCK	Res.	Res.	Res.	Res.
	Reset value																									0	0	0	0				
0x0014	FDCAN_RWD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDV[7:0]								WDC[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 329. FDCAN 寄存器映射和复位值(1)（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0018	FDCAN_CCCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NISO	TXP	EFBI	PXHD	Res.	Res.	BRSE	FDOE	TEST	DAR	MON	CSR	CSA	ASM	CCE	INT
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x001C	FDCAN_NBTP	NSJW[6:0]							NBRP[8:0]									NTSEG1[7:0]								Res.	NTSEG2[6:0]						
	Reset value	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	1
0x0020	FDCAN_TSCC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TCP[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSS [1:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0024	FDCAN_TSCV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSC[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0028	FDCAN_TOCC	TOP[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TOS [1:0]		ETOC
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x002C	FDCAN_TOCV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TOC[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x0030-0x003C	Reserved	Reserved																															
0x0040	FDCAN_ECR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CEL[7:0]								RP	REC[6:0]							TEC[7:0]							
	Reset value								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0044	FDCAN_PSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDCV[6:0]							Res.	PXE	REDL	RBRSRESI1	RESI	DLEC[2:0]			BO	EW	EP	ACT[1:0]		LEC[2:0]		
	Reset value									0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
0x0048	FDCAN_TDCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDCO[6:0]							Res.	TDCF[6:0]						
	Reset value																		0	0	0	0	0	0		0	0	0	0	0	0	0	0
0x004C	Reserved	Reserved																															
0x0050	FDCAN_IR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARA	PED	PEA	WDI	BO	EW	EP	ELO	TOO	MRAF	TSW	TEFL	TEFF	TEFN	TFE	TCF	TC	HPM	RF1L	RF1F	RF1N	RF0L	RF0F	RF0N
	Reset value								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0054	FDCAN_IE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARAE	PEDE	PEAE	WDIE	BOE	EWE	EPE	ELOE	TOOE	MRAFE	TSWE	TEFLE	TEFFE	TEFNE	TFEE	TCFE	TCE	HPME	RF1LE	RF1FE	RF1NE	RF0LE	RF0FE	RF0NE
	Reset value								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0058	FDCAN_ILS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERR	BERR	MISC	TFERR	SMSG	RXFIFO1	RXFIFO0
	Reset value	0																									0	0	0	0	0	0	0

表 329. FDCAN 寄存器映射和复位值(1)（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x005C	FDCAN_ILE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EINT1	EINT0
	Reset value																															0	0
0x0060-0x007C	Reserved	Reserved																															
0x0080	FDCAN_RXGFC	Res.	Res.	Res.	Res.	LSE[3:0].				Res.	Res.	Res.	LSS[4:0]					Res.	Res.	Res.	Res.	Res.	Res.	F0OM	F1OM	Res.	Res.	ANFS[1:0]		ANFE[1:0]		RRFS	RRFE
	Reset value					0	0	0	0				0	0	0	0	0							0	0			0	0	0	0	0	0
0x0084	FDCAN_XIDAM	Res.	Res.	Res.	EIDM[28:0]																												
	Reset value				1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x0088	FDCAN_HPMS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLST	Res.	Res.	FIDX[4:0]					MSI [1:0]		Res.	Res.	Res.	BIDX [2:0]		
	Reset value																	0			0	0	0	0	0	0	0				0	0	0
0x0090	FDCAN_RXF0S	Res.	Res.	Res.	Res.	Res.	Res.	RF0L	F0F	Res.	Res.	Res.	Res.	Res.	Res.	F0PI [1:0]		Res.	Res.	Res.	Res.	Res.	Res.	F0GI [1:0]		Res.	Res.	Res.	Res.	F0FL[3:0]			
	Reset value							0	0							0	0							0	0					0	0	0	0
0x0094	FDCAN_RXF0A	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	F0AI[2:0]		
	Reset value																														0	0	0
0x0098	FDCAN_RXF1S	Res.	Res.	Res.	Res.	Res.	Res.	RF1L	F1F	Res.	Res.	Res.	Res.	Res.	Res.	F1PI [1:0]		Res.	Res.	Res.	Res.	Res.	Res.	F1GI [1:0]		Res.	Res.	Res.	Res.	F1FL[3:0]			
	Reset value							0	0							0	0							0	0					0	0	0	0
0x009C	FDCAN_RXF1A	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	F1AI[2:0]		
	Reset value																														0	0	0
0x00A0-0x00BC	Reserved	Reserved																															
0x00C0	FDCAN_TXBC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TFQM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value								0	0																							
0x00C4	FDCAN_TXFQS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TFQF	Res.	Res.	Res.	TFQPI[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	TFGI[1:0]		Res.	Res.	Res.	Res.	Res.	TFFL[2:0]		
	Reset value											0				0	0							0	0						0	1	1
0x00C8	FDCAN_TXBRP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRP2	TRP1	TRP0
	Reset value																														0	0	0
0x00CC	FDCAN_TXBAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AR2	AR1	AR0
	Reset value																														0	0	0

表 329. FDCAN 寄存器映射和复位值(1)（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00D0	FDCAN_TXBCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CR2	CR1	CR0
	Reset value																														0	0	0
0x00D4	FDCAN_TXBTO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TO2	TO1	TO0
	Reset value																														0	0	0
0x00D8	FDCAN_TXBCF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CF2	CF1	CF0
	Reset value																														0	0	0
0x00DC	FDCAN_TXBTIE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TIE2	TIE1	TIE0
	Reset value																														0	0	0
0x00E0	FDCAN_TXBCIE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CFIE2	CFIE1	CFIE0
	Reset value																														0	0	0
0x00E4	FDCAN_TXEFS	Res.	Res.	Res.	Res.	Res.	Res.	TEFL	EFF	Res.	Res.	Res.	Res.	Res.	Res.	EFPI[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EFG[1:0]		Res.	Res.	Res.	Res.	Res.	EFFL[2:0]		
	Reset value							0	0							0	0							0	0						0	0	0
0x00E8	FDCAN_TXEFA	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EFAI[1:0]	
	Reset value																															0	0
0x0100	FDCAN_CKDIV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PDIV[3:0]			
	Reset value																												0	0	0	0	0

1. R = 读访问，S = 进行读访问时置 1，X = 进行读访问时复位，W = 写访问，P = 受写保护，p = 受保护置 1，C = 进行写访问时清零/预设。

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

40 通用串行总线全速主机/设备接口 (USB)

40.1 简介

USB 外设在全速 USB 2.0 总线和 APB2 总线之间实现了一个接口。

此外设支持 USB 挂起/恢复，这样可以停止设备时钟以降低功耗。

40.2 USB 主要特性

- 符合 USB 2.0 全速规范
- 支持主机和设备模式
- 可配置端点数为 1 个到 8 个
- 2048 字节的专用数据包缓冲区存储器 (SRAM)
- 循环冗余校验 (CRC) 生成/校验、非归零反相 (NRZI) 编码/解码以及位填充
- 同步传输支持
- 双缓冲批量/同步端点/通道支持
- USB 挂起/恢复操作
- 帧锁定时钟脉冲生成
- USB 2.0 链路层电源管理支持（仅设备模式）
- 电池充电规范第 1.2 版支持（仅设备模式）
- USB 连接/断开功能（USB_DP 线上可控制的内置上拉电阻）

40.3 USB 实现

表 330 列出了器件中的 USB 实现。

表 330. STM32H503xxUSB 实现

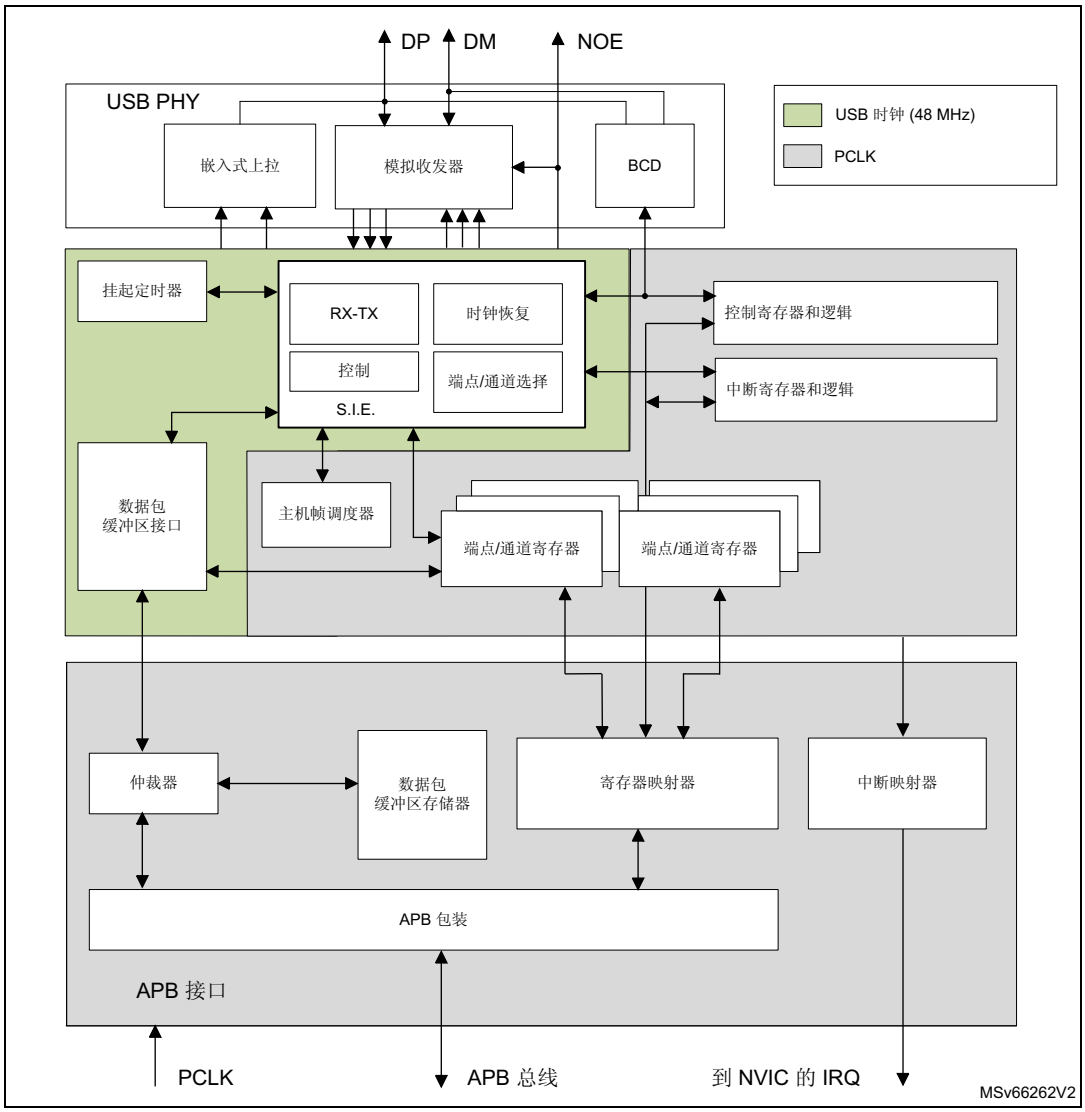
USB 特性(1)	USB
主机模式	X
端点数	8
专用数据包缓冲区存储器 SRAM 的大小	2048 字节
专用数据包缓冲区存储器 SRAM 访问模式	32 位
设备支持 USB 2.0 链路层电源管理 (LPM)	X
设备支持电池充电检测 (BCD)	X
USB_DP 线上的内置上拉电阻	X

1. X = 支持

40.4 USB 功能描述

图 509 显示了 USB 外设的框图。

图 509. USB 外设框图



一般描述和设备模式功能

USB 外设在微控制器实现的功能与外部 USB 功能之间提供 USB 兼容连接，USB 外设可以是主机 PC，也可以是 USB 设备。外部 USB 主机或设备与系统存储器之间通过专用数据包缓冲区存储器进行数据传输，此数据包缓冲区存储器通过 USB 外设直接访问。此专用存储器大小为 2048 字节，最多可使用 16 个单向端点或 8 个双向端点。USB 外设与外部 USB 主机或设备相连，按 USB 标准的要求检测令牌数据包、处理数据发送/接收以及处理握手包。事务格式化通过硬件来执行，其中包括 CRC 生成和校验。

每个端点/通道都与一个缓冲区描述块关联，此描述块指示与端点/通道相关的存储区的位置、大小或必须传送的字节数。当 USB 外设识别到有效功能/端点对的令牌时，将根据需要进行相关的数据传输（如果已配置端点/通道）。USB 外设缓冲的数据装载到内部 16 位寄存

器中，并且会对专用缓冲区执行存储器访问。当所有数据传输完成后，必要时，可根据传输方向生成或等待适当的 USB 握手包。

事务结束时将生成端点/通道相关中断，以读取状态寄存器和/或使用不同的中断响应程序。微控制器可确定：

- 必须处理哪个端点/通道
- 出现错误时发生了哪类事务（位填充、格式化、CRC、协议、ACK 丢失和上溢/下溢等）。

为同步传输和高吞吐量批量传输提供特殊支持，实现双缓冲区用法，从而确保当微控制器使用其中一个缓冲区时，始终有另一个缓冲区可供 USB 外设使用。

寄存器 USB_ISTR 中的特殊位 THR512 允许通知缓冲区要接收（或发送）512 字节。该位必须用于长 ISO 数据包（512 到 1023 字节），因为它有利于数据的早期启动或读/写。这样，前 512 个字节可以由软件处理，同时避免使用双缓冲模式。仅配置一个 ISO 端点时，该位会起作用。

必要时，可通过写入控制寄存器将单元置于低功耗模式（挂起模式）。此时，可避免产生所有静态功耗，USB 时钟会减速或停止。如果在低功耗模式下检测到 USB 输入上存在活动，则会异步唤醒设备。可以将特殊中断源直接连接到唤醒线，从而使系统立即重启正常的时钟生成操作和/或支持直接时钟启动/停止。

主机模式和具体功能

寄存器 USB_CNTR 中的单个位 HOST 可用来激活主机模式。主机模式功能允许 USB 与远程外设通信。支持的功能与设备模式一致，并且使用相同的寄存器结构来管理缓冲区。在主机模式下，可支持相同数量的端点，但在该模式下首选使用术语“通道”，因为每个通道实际上是连接的设备和该设备上的端点的组合。数据包收发的基本机制与设备模式中支持的机制相同。

当在主机模式下运行时，USB 负责总线，为此必须发出与有效的周期性和非周期性端点相对应的事务请求。主机帧调度程序确保有效使用帧。支持连接到集线器。支持通过直连方式及通过集线器连接低速设备。

在主机模式下，在批量通道和同步通道中，还支持双缓冲模式（如前文设备模式中所述）。THR512 功能也同样受到支持（与设备模式相同），但仅限 ISO 通信数据。

注意： *在设备模式下会检测电池充电能力（以便快速充电），而主机模式与之不同，主机模式并未集成相应支持来呈现电池充电能力（标准中的 CDP 或 DCP 情形），主机端口始终以默认的标准数据端口 (SDP) 的形式提供。*

注意： *对于 LPM（链路层电源管理），主机模式不支持该功能。*

40.4.1 设备和主机模式下所用 USB 模块的说明

USB 外设实现了与 USB 连接相关的所有功能，具体包括以下模块：

- USB 物理接口 (USB PHY)：此模块维持与外部 USB 主机间的电气连接。它包含差分模拟收发器本身、可控制的内置上拉电阻（连接到 USB_DP 线）以及电池充电检测 (BCD) 支持，在相同 USB_DP 和 USB_DM 线上复用。模拟收发器的输出使能控制信号（低电平有效）由 USB_NOE 在外部提供。它可用于驱动某些活动 LED，或向某些其他电路提供有关实际通信方向的信息。
- 串行接口引擎 (SIE)：此模块的功能包括：同步模式识别、位填充、CRC 生成和校验、PID 验证/生成以及握手评估。它必须与 USB 收发器连接，并使用数据包缓冲区接口提供的虚拟缓冲区来存储本地数据。该单元还会根据 USB 外设事件（例如，帧起始 (SOF)、USB_Reset 和数据错误等）和端点相关事件（例如，数据包发送结束或接收成功）生成相应信号，这些信号随后用于生成中断。

- 定时器：此模块生成帧起始锁定时钟脉冲，并在已有 3 ms 未接收到通信数据时检测全局挂起（来自主机）。
- 数据包缓冲区接口：此模块管理本地存储器，能够以灵活的方式实现一组用于发送和接收的缓冲区。它可根据来自 SIE 的请求选择合适的缓冲区，并将这些请求置于端点/通道寄存器指向的存储器地址。在数据包交换过程中，每交换一个字节，地址便会递增一次，这样便可跟踪已交换的字节数，避免超出缓冲区的最大容量。
- 端点/通道相关寄存器：每个端点/通道有一个相关的寄存器，其中包含端点类型及其当前状态。对于单向/单缓冲区端点，可使用单个寄存器实现两个不同的端点。这类寄存器有 8 个，最多支持 16 个单向/单缓冲区端点或 7 个双缓冲区端点，并且支持这两种端点的任意组合。例如，可以将 USB 外设编程为具有 4 个双缓冲区端点和 8 个单缓冲区/单向端点。
- 控制寄存器：这类寄存器包含与整个 USB 外设的状态相关的信息，用于强制执行某些 USB 事件（例如，恢复和掉电）。
- 中断寄存器：这类寄存器包含中断屏蔽和事件记录。它们可用于查询中断原因、中断状态或清除挂起中断的状态。

注意： ** 端点/通道 0 始终用于单缓冲区模式下控制传输。*

USB 外设通过 APB2 接口连接到 APB2 总线，此接口包含以下模块：

- 数据包存储器：此存储器是在物理上包含数据包缓冲区的本地存储器。可通过数据包缓冲区接口使用此存储器，以创建数据结构并且可通过应用软件直接访问。数据包存储器的大小为 2048 字节，以 512 个 32 位字的结构存在。
- 仲裁器：此模块接受来自 APB2 总线和 USB 接口的存储器请求。它通过为 APB2 访问分配优先级来解决冲突，同时始终保留一半存储器带宽来完成所有 USB 传输。该时分双工方案实现了虚拟双端口 SRAM，允许在发生 USB 事务时进行存储器访问。此方案还支持任意长度的多字 APB2 传输。
- 寄存器映射器：此模块收集 USB 外设不同字节宽和位宽的寄存器，这些寄存器采用 32 位宽字组的形式，通过 APB2 寻址。
- APB2 封装器：此模块为存储器和寄存器提供连接 APB2 的接口。它还将整个 USB 外设映射到 APB2 地址空间。
- 中断映射器：此模块用于选择可能的 USB 事件如何生成中断以及将中断映射到 NVIC。

40.4.2 特定于主机模式的主机帧调度器 (HFS) 的说明

主机帧调度器属于硬件机器，负责根据 USB 优先顺序和带宽访问规则在总线上提交主机通道请求。

主机通道分为两类：

- 周期性通道：等时和中断通信数据类型。带宽访问有保证。
- 非周期性通道：批量和控制通信数据类型。提供尽力服务。

主机帧调度器将全速帧编排在 3 个连续窗口中

- 周期性服务窗口
- 非周期性服务窗口
- 黑色安全窗口

在新帧开始时，主机调度器：

1. 首先考虑帧开始时处于活动状态的所有周期性通道（STAT 位有效）
2. 按照硬件优先级顺序从 CH#1 到 CH#8 执行单轮周期性通道服务（周期性服务窗口）。对于双向通道，它首先执行 OUT 方向的服务
3. 当周期性循环结束时，HFS 会关闭周期性服务窗口并停止处理周期性数据通信，即使在 SOF 之后重新使能了某些周期性通道或使能了一些新通道也是如此。
4. 按照硬件优先级顺序从 CH#1 到 CH#8 开始处理当前处于活动状态（STAT 位有效）的所有非周期性通道。对于双向通道，它首先执行 OUT 方向的服务。
5. 执行非周期性通道的多个循环调度服务周期，直到帧快结束为止
6. 可以随时请求非周期性数据通信，并由 HFS 以尽力服务延迟进行处理，但帧末尾的黑色安全窗口除外，其中新注入的请求将直接推迟到下一帧，以避免出现混乱。对于在安全窗口之前尚未得到处理的挂起事务也是如此。

40.5 设备模式和主机模式下的编程注意事项

以下各节将介绍 USB 外设与应用软件之间的预期交互，帮助简化应用软件开发。

40.5.1 通用 USB 设备编程

本部分将介绍应用软件需要执行哪些主要任务才能获得符合 USB 规范的行为。与大多数常规 USB 事件相关的操作均在考虑范围之内，并且有相关段落专门说明双缓冲端点和同步传输这两个特例。除系统复位外，操作始终在下述其中一个 USB 事件的驱动下由 USB 外设发起。

40.5.2 系统复位和上电复位

系统复位和上电复位后，应用软件必须执行的第一个操作是为 USB 外设提供所需的全部时钟信号，随后使其复位信号无效，以便能够访问其寄存器。下文将介绍整个初始化序列。

首先，应用软件需要使用设备时钟管理逻辑所提供的相关控制位激活寄存器宏单元时钟并使宏单元特定的复位信号无效。

随后，必须使用 CNTR 寄存器的 PDWN 位接通设备中与 USB 收发器相关的模拟部分，这一步需要特殊处理。该位旨在接通为端口收发器供电的内部参考电压。此电路有一段定义的启动时间（数据手册中指定的 $t_{STARTUP}$），这段时间内未定义 USB 收发器的行为。因此，在将 CNTR 寄存器中的 PDWN 位置 1 后，必须等待这段时间结束时才能消除 USB 部分的复位条件（通过将 CNTR 寄存器中的 USBRST 位清零）。清除 ISTR 寄存器会在任何其他宏单元操作使能之前，消除所有意外的挂起中断。

系统复位时，微控制器必须初始化所需的全部寄存器和数据包缓冲区描述表，以便 USB 外设能够正常地产生中断和进行数据传输。对于非特定于任何端点/通道的所有寄存器，必须按照应用软件的需求（使能中断的选择，所选的数据包缓冲区地址等）进行初始化。对于 USB 复位情况，过程将继续进行（请参见下一段）。

设备模式下的 USB 总线复位（RST_DCON 中断）

发生该事件时，USB 外设的状态与系统复位后的初始化状态（如上一段所述）相同：所有端点寄存器中均禁止通信（USB 外设不会响应任何数据包）。作为对 USB 复位事件的响应，必须使能 USB 功能（USB 地址为 0），仅实现默认控制端点（端点地址也为 0）。具体实现方法是：将 USB_DADDR 寄存器的使能功能 (EF) 位置 1 并相应地初始化 CHEP0R 寄存器及其相关的数据包缓冲区。在 USB 枚举过程中，主机会为此设备分配一个唯一地址（该地址必须写入 USB_DADDR 寄存器的 ADD[6:0] 位）并配置任何其他必要的端点。

当接收到 RST_DCON 中断时，应用软件负责在触发中断的复位序列结束后的 10 ms 内再次使能 USB 功能 0 的默认端点。

主机模式下 USB 总线复位

在主机模式下，总线复位通过设置 USB_CNTR 寄存器的 USBRST 位来激活。达到标准中的最短有效复位时间后，必须随后由软件将其清零。

数据包缓冲区的结构和用法

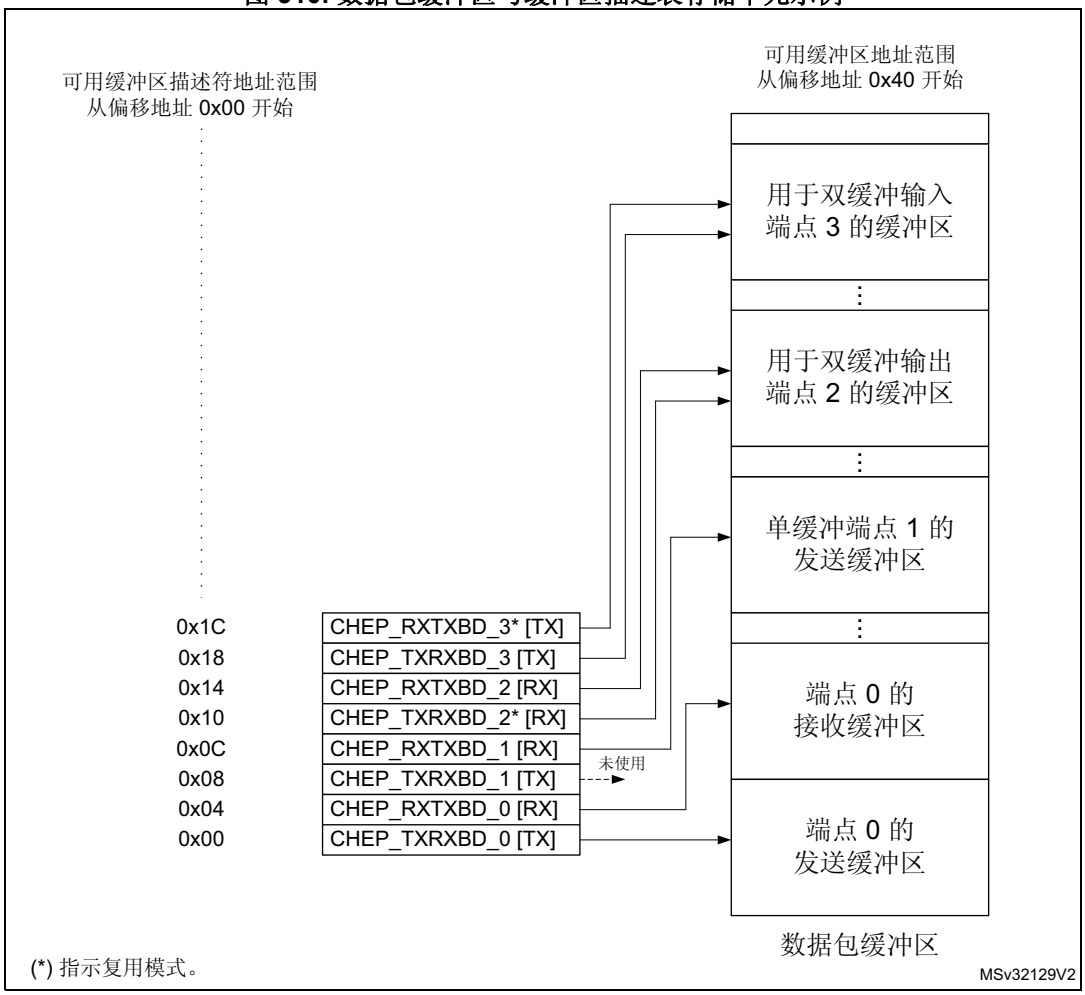
每个双向端点均可通过总线接收或发送数据。端点接收的数据存储在为该端点保留的专用存储器缓冲区内，而另一个存储器缓冲区则包含该端点要发送的数据。对该存储器的访问通过数据包缓冲区接口模块来执行，该模块会发出存储器访问请求并等待应答。由于数据包缓冲区存储器还必须供微处理器访问，因此需通过仲裁逻辑来处理访问冲突：半个 APB2 周期供微控制器访问，剩下的半个周期供 USB 外设访问。通过这种方式，微控制器和 USB 外设可将数据包存储器视为双端口 SRAM 来执行操作，而无需考虑任何冲突，即使微控制器正在执行背靠背访问也如此。USB 外设逻辑使用专用时钟。该专用时钟的频率按照 USB 标准的要求固定为 48 Mhz，这可能与用于连接 APB2 总线的时钟不同。时钟配置可以不同，其中 APB2 时钟频率可以高于或低于 USB 外设的时钟频率。

注意： *根据 USB 数据速率和数据包存储器接口要求，必须确保 APB2 时钟的频率至少为 12 MHz，以避免发生数据上溢/下溢问题。*

每个端点与两个数据包缓冲区关联（通常，一个用于发送，另一个用于接收）。缓冲区可置于数据包存储器内的任何位置，因为其位置和大小在缓冲区描述表中指定，此描述表也位于数据包存储器中。每个表条目与一个端点寄存器关联并且由两个 32 字组成，因此表起始地址必须始终与 8 字节边界对齐。*第 40.6.2 节：缓冲区描述符表*介绍了缓冲区描述符表条目。对于一个既非同步端点也非双缓冲批量端点的单向端点，只需要一个数据包缓冲区（与所支持的传输方向相关的数据包缓冲区）。与不支持的传输方向或未使用端点相关的其他表存储单元可供用户使用。同步端点和双缓冲批量端点的数据包缓冲区处理方式比较特殊（请分别参见*第 40.5.5 节：设备模式下的同步传输*和*第 40.5.3 节：设备模式下双缓冲端点和使用*）。*图 510* 给出了缓冲区描述表条目和数据包缓冲区之间的关系。

对于主机模式，多个章节特别是*第 40.5.6 节：主机模式下的同步传输*和*第 40.5.4 节：双缓冲通道：主机模式下的使用*对缓冲区用法模型进行了阐述。

图 510. 数据包缓冲区与缓冲区描述表存储单元示例



在接收或发送期间，每个数据包缓冲区从底部开始使用。USB 外设始终不会更改与所分配存储器缓冲区相邻的存储单元的内容；如果接收的数据包长度大于所分配缓冲区的长度（缓冲区上溢条件），则到达最后一个可用的存储单元后便停止将数据复制到存储器。

端点初始化

初始化端点的第一步是向 CHEP_TXBD_n 和 CHEP_RXBD_n 寄存器（在 SRAM 中）的 ADDRn_TX/ADDRn_RX 位域写入相应值，以使 USB 外设获悉要发送的数据已准备就绪，要接收的数据可进行缓冲。必须根据端点类型设置 USB_CHEPnR 寄存器中的 UTYPE 位，最终使用 EPKIND 位使能所需的任何特殊功能。在发送端，必须通过 USB_CHEPnR 寄存器中的 STATTX 位使能端点，并且必须初始化 COUNTn_TX。接收时，必须将 STATRX 位置 1 以使能接收，并且必须通过 BLSIZE 和 NUM_BLOCK 位域向 COUNTn_RX 写入所分配的缓冲区大小。单向端点（同步端点和双缓冲批量端点除外）只需初始化与所支持方向相关的位和寄存器。使能发送与/或接收后，不得通过应用软件修改寄存器 USB_CHEPnR 以及存储单元 ADDRn_TX/ADDRn_RX、COUNTn_TX/COUNTn_RX（分别地），因为硬件会实时更改其值。数据传输操作完成时（通过 CTR 中断事件通知），可再次访问这些存储单元以重新使能新操作。

设备模式下的数据发送（IN 数据包）

接收 IN 令牌数据包时，如果所接收的地址与已配置的有效端点匹配，则 USB 外设会访问与所寻址端点相关的缓冲区描述符表条目内 CHEP_TXBD_n（ADDRn_TX 和 COUNTn_TX 位域）的内容。这些存储单元的内容存储在其内部 16 位寄存器 ADDR 和 COUNT 中（无法通过软件访问）。将再次访问数据包存储器以读取要发送的第一个字节（请参见*第 1518 页的数据包缓冲区的结构和用法*），USB 外设会根据 USB_CHEPnR 寄存器的 DTOGTX 位开始发送 DATA0 或 DATA1 PID。PID 完成时，从缓冲区存储器中读取的第一个字节将加载到输出移位寄存器中，以便在 USB 总线上发送。发送最后一个数据字节后，将发送计算得出的 CRC。如果寻址的端点无效，将根据 USB_CHEPnR 寄存器中的 STATTX 位发送 NAK 或 STALL 握手包而非数据包。

内部寄存器 CHEP_TXBD_n 中的 ADDRn_TX 位域用作当前缓冲区存储单元的指针，而 COUNT 用于对要发送的剩余字节计数。从数据包缓冲区存储器读取的每个半字均通过 USB 总线从最低有效字节开始发送。在发送缓冲区存储器中从 ADDRn_TX 指向的地址开始读取 COUNTn_TX/4 个字。如果发送的数据包由奇数个字节组成，则仅会使用所访问的最后一个半字的低半部分。

主机接收到 ACK 后，USB_CHEPnR 寄存器按以下方式更新：DTOGTX 位翻转，通过设置 STATTX = 10 (NAK) 将端点置为无效，将 VTTX 位置 1。应用软件必须首先识别端点，即请求微控制器关注以检查 USB_ISTR 寄存器中的 IDN 和 DIR 位。处理 VTTX 事件会开始清零中断位；应用软件随后会准备另一个装满数据的发送缓冲区，用下一次传输期间要发送的字节数更新 COUNTn_TX 表存储单元，最后将 STATTX 设置为 11 (VALID) 以重新使能发送。当 STATTX 位等于 10 (NAK) 时，寻址到相应端点的任何 IN 请求都得不到应答，并且指示流控制状态：USB 主机将重试事务，直到成功。必须按上述顺序执行操作序列，以避免与触发 CTR 中断的 IN 事务寻址到同一端点且紧随其后的第二个 IN 事务的通知丢失。

主机模式下的数据发送（OUT 数据包）

主机模式下的数据发送所遵循的一般原则与设备模式相同。主要差异归因于协议。例如，主机发起传送，而设备则响应传入的令牌。

ADDRn_TX 必须设置为数据包存储器中为发送数据包所保留的单元。然后，OUT 数据包的内容将写入数据包存储器中的这一地址，并且 COUNTn_TX 必须更新（如有必要），以指示数据包中的字节数。

必须为正确的端点写入 DEVADDR，之后必须将 STATTX 设置为 11（有效）才能触发发送。然后由 HFS 安排发送。

成功发送后，将触发 CTR（正确传输）中断。通过检查 IDN 和 DIR 位，可以获悉相应的通道和方向。在所指示的通道上，STATTX 位域现已转换为 DISABLE。如果是接收 NAK（外设未就绪时），STATTX 现在处于 NAK 状态。如果是 STALL 响应，则 STATTX 处于 STALL 状态。在最后一种情况下，总线必须复位。

器件收到 ACK 回执后，USB_CHEPnR 寄存器会以下列方式更新：DTOGTX 位翻转。

在以下情况下，将通过 VTTX 和 ERR_TX 位指示错误状态：

- 未及时收到握手信号
- 错误的 EOP
- 位填充错误
- 握手 PID 无效

设备模式下的数据接收（OUT 和 SETUP 数据包）

USB 外设对这两种令牌的处理方式大致相同，SETUP 数据包不同于 OUT 数据包的处理将在下文有关控制传输的段落中详细说明。接收 OUT/SETUP PID 时，如果地址与有效端点匹配，则 USB 外设会访问与所寻址端点相关的缓冲区描述符表条目内的 ADDRn_RX 和 COUNTn_RX 位域的内容。ADDRn_RX 位域的内容直接存储在其内部寄存器 ADDR 中。内部寄存器 COUNT 现在复位，BLSIZE 和 NUM_BLOCK 位域的值（从 USB_CHEP_RXBD_n 内容中读取）用于初始化内部 16 位计数器 BUF_COUNT，该计数器用于检查缓冲区上溢条件（上述所有内部寄存器均无法通过软件访问）。USB 外设后续接收的数据字节均以半字为单位打包（接收的第一个字节作为最低有效字节存储），随后传输到从内部 ADDR 寄存器包含的地址开始的数据包缓冲区，每次传输字节时 BUF_COUNT 递减、COUNT 递增。当检测到 DATA 数据包结束时，将检验所接收的 CRC 是否正确，只有当接收期间没有错误时，才会将 ACK 握手包发送回发送主机。

当出现 CRC 错误或其他类型的错误（位填充超限、帧错误等）时，数据字节仍将在数据包存储器缓冲区中进行复制，至少到错误检测点为止，但不会发送 ACK 数据包且 USB_ISTR 寄存器中的 ERR 位置 1。不过，这种情况下通常不需要软件操作：USB 外设会从接收错误中恢复，并保持就绪状态等待下一个事务的到来。如果寻址的端点无效，将根据 USB_CHEPnR 寄存器中的 STATRX 位发送 NAK 或 STALL 握手包而非 ACK，并且不会向接收存储器缓冲区中写入任何数据。

接收存储器缓冲区存储单元从 ADDRn_RX 中包含的地址开始写入，写入的字节数与接收的数据包长度相对应，或者一直写到分配的最后一个存储单元（由 BLSIZE 和 NUM_BLOCK 定义），以先结束者为准。凭借这种方法，USB 外设写入的数据始终不会超出所分配接收存储器缓冲区的容量。如果数据包有效负载的长度（应用软件实际使用的字节数）大于所分配的缓冲区，则 USB 外设会检测到缓冲区上溢条件。在这种情况下，将发送 STALL 握手而非常规的 ACK 来通知主机出现问题，此时不会产生中断，事务将被视作失败。

当事务正确完成后（通过发送 ACK 握手包），内部 COUNT 寄存器将复制回缓冲区描述表条目内的 COUNTn_RX 存储单元。BLSIZE 和 NUM_BLOCK 位域不受影响，这两个位域通常无需重写，USB_CHEPnR 寄存器通过以下方式更新：DTOGRX 位翻转，通过设置 STATRX = 10 (NAK) 将端点置为无效，将 VTRX 位置 1。如果事务因错误或缓冲区上溢条件而失败，则上面列出的所有操作均不会发生。应用软件必须首先识别端点，即请求微控制器关注以检查 USB_ISTR 寄存器中的 IDN 和 DIR 位。首先确定事务类型（USB_CHEPnR 寄存器中的 SETUP 位）以处理 VTRX 事件；应用软件必须清零中断标志位，并通过读取与正在处理的端点相关的缓冲区描述表条目内的 COUNTn_RX 存储单元获得所接收字节的数量。处理完接收的数据后，应用软件必须将 USB_CHEPnR 中的 STATRX 位置为 11（有效），从而允许其他事务。当 STATRX 位等于 10 (NAK) 时，寻址到相应端点的任何 OUT 请求都得不到应答，并且指示流控制状态：USB 主机将重试事务，直到成功。必须按上述顺序执行操作序列，以避免丢失与触发 CTR 中断的 OUT 事务寻址到同一端点且紧随其后的第二个 OUT 事务的通知。

主机模式下的数据接收（IN 数据包）

主机模式下的数据接收所遵循的一般原则与设备模式相同。主要差异同样归因于协议。在设备中，是否可以接收数据取决于先前操作后的准备情况，而主机仅在准备就绪并能存储数据时才请求接收数据。

ADDRn_TX 必须设置为数据包存储器中为发送数据包所保留的单元。然后，在数据阶段收到的、对 IN 令牌数据包响应的内容中，将写入到数据包存储器中的上述地址中，并且在此过程中 COUNTn_TX 由硬件更新以指示数据包中的字节数。

必须为正确的端点写入 DEVADDR，之后必须将 STATRX 置为有效才能触发接收。然后由 HFS 安排接收。

成功接收后，将触发 CTR（正确传输）中断。通过检查 IDN 和 DIR 位，可以获悉相应的通道和方向。在所指示的通道上，STATRX 位域现已转换为 DISABLE。如果是接收 NAK（外设未就绪时），STATRX 现在处于 NAK 状态。如果是 STALL 响应，则 STATRX 处于 STALL 状态。在最后一种情况下，总线必须复位。在 IN 数据包期间，如果发生以下情况，将通过 VTRX 和 ERR_RX 位指示错误状态：

- 错误的 EOP
- 位填充错误
- CRC 错误

设备模式下的控制传输

控制传输包含一个 SETUP 事务，其后跟有零个或多个数据阶段（全部数据阶段均采用相同的方向），再后面是一个状态阶段（一个零字节反向传输）。SETUP 事务仅由控制端点来处理，并且与 OUT 事务十分相似（数据接收），不同之处在于所寻址端点寄存器的 DTOGTX 和 DTOGRX 位的值分别设置为 1 和 0 以初始化控制传输，STATTX 和 STATRX 均设置为 10 (NAK) 以允许软件决定后续事务必须为 IN 还是 OUT，具体取决于 SETUP 的内容。控制端点必须在每次发生 VTRX 事件时检查 USB_CHEPnR 寄存器中的 SETUP 位，以区分正常的 OUT 事务与 SETUP 事务。USB 设备可通过解析 SETUP 阶段传输的数据来确定数据阶段的数量和方向，并用于在出现错误时停止事务。为此，在最后一个数据阶段前的所有数据阶段中，必须将未使用的方向设置为 STALL，以便在主机过于迅速地改变传输方向时获取 STALL 作为状态阶段。

使能最后一个数据阶段时，必须将相反方向设置为 NAK，以便在主机立即改变传输方向（以执行状态阶段）时一直等待控制操作完成。如果控制操作成功完成，软件会将 NAK 更改为 VALID，否则将更改为 STALL。同时，如果状态阶段为 OUT，则必须将 STATUS_OUT（USB_CHEPnR 寄存器中的 EPKIND）位置 1，这样，如果执行数据非零的状态事务，就会生成错误。当处理状态事务时，应用软件会将 STATUS_OUT 位清零，并将 STATRX 设置为 VALID（以接受新命令）、将 STATTX 设置为 NAK（以在下一次 SETUP 后立即延迟一段可能的状态阶段）。

由于 USB 规范规定 SETUP 数据包不可通过 ACK 以外的握手包来应答（这最终将中止之前发出的命令而启动新命令），USB 逻辑不允许控制端点通过 NAK 或 STALL 握手包来应答从主机接收的 SETUP 令牌。

当 STATRX 位设置为 01 (STALL) 或 10 (NAK) 且接收到 SETUP 令牌时，USB 将接受数据（以执行所需的数据传输）并发回 ACK 握手包。如果该端点之前发出的 VTRX 请求尚未得到应用软件的确认（例如，VTRX 位仍因之前完成的接收操作而置 1），USB 将丢弃 SETUP 事务且不会通过任何握手包进行应答（无论其状态如何均如此），从而模拟接收错误并强制主机再次发送 SETUP 令牌。这样做的目的在于避免与触发 VTRX 中断的 SETUP 事务寻址到同一端点且紧随其后的第二个 SETUP 事务的通知丢失。

主机模式下的控制传输

控制传输包含一个 SETUP 事务，其后跟有零个或多个数据阶段（全部数据阶段均采用相同的方向），再后面是一个状态阶段（一个零字节反向传输）。SETUP 事务仅由控制端点处理。控制端点必须将 USB_CHEPnR 寄存器中的 SETUP 位置 1。已寻址的端点寄存器的 DTOGTX 和 DTOGRX 位的值设为 0。根据是控制写入还是控制读取，STATTX 或 STATRX 设为 11（ACTIVE），以便通过主机帧调度器触发控制传输。

接收到 CTR 中断后，可以通过检查 IDN 和 DIR 位来确定通道（设备地址和端点）。设备应对每个控制发送 NAK，除非数据包已损坏，在这种情况下它们不会进行确认。这种情况通过 STATTX 的值反映。

如果出现错误情况，ERR 位会置 1。设备上可能会出现 CRC 错误，在这种情况下，不会向主机返回 ACK。主机看不到 ACK，并且在适当延迟之后，会生成一个超时错误且 ERR_TX 置 1（可生成中断）。

40.5.3 设备模式下双缓冲端点和使用

USB 标准定义的所有不同端点类型均代表不同的通信模型，并且说明了各类数据传输操作的典型要求。当主机 PC 和 USB 功能之间要传输大量数据时，批量端点类型是最适合的模型。这是因为主机会调度批量事务以填充帧中的所有可用带宽，因此只要 USB 功能就绪处理寻址到它的批量事务，便可最大程度地提高实际传输速率。如果在下一个事务到达时，USB 功能仍忙于上一个事务，则它将通过 NAK 握手进行应答。主机 PC 将再次发出同一事务，直到 USB 功能准备好处理该事务。由于重复发送占用带宽，因此实际传输速率会降低。为此，可为批量端点搭配使用称为“双缓冲”的专用功能。

激活“双缓冲”功能时，数据翻转序列用于选择 USB 外设将使用哪个缓冲区来执行所需的数据传输，使用“发送”和“接收”数据包存储区在每个事务成功完成时管理缓冲区交换，以便在 USB 外设填充一个缓冲区时，始终有另一个完整的缓冲区可供应用软件使用。例如，在指向“接收”双缓冲批量端点的 OUT 事务期间，当一个缓冲区用来自 USB 主机的新数据填充时，另一个缓冲区可供微控制器软件使用（同样适用于“发送”双缓冲批量端点和 IN 事务）。

由于交换缓冲区管理需要使用全部 4 个缓冲区描述表存储单元来管理地址指针和已分配存储区的长度，因此用于实现双缓冲批量端点的 USB_CHEPnR 寄存器被强制用作单向寄存器。因此，只有一对 STAT 位必须设置为非 00（DISABLED）的值：如果针对接收使能了双缓冲批量端点，则为 STATRX；如果针对发送使能了双缓冲批量端点，则为 STATTX。如果需要将双缓冲批量端点使能同时用于发送和接收，则必须使用两个 USB_CHEPnR 寄存器。

要利用双缓冲功能达到最高传输速率，必须修改前几章中介绍的端点流控制结构，以便只在 USB 外设和应用软件之间发生缓冲区冲突时（而非每次事务成功结束时），将端点状态切换为 NAK。USB 外设当前使用的存储器缓冲区由与端点方向相关的 DTOG 位定义：DTOGRX（USB_CHEPnR 寄存器的位 14）用于“接收”双缓冲批量端点，而 DTOGTX（USB_CHEPnR 寄存器的位 6）用于“发送”双缓冲批量端点。为实现新的流控制方案，USB 外设必须知道应用软件当前使用哪个数据包缓冲区，以便获悉是否存在任何冲突。由于 USB_CHEPnR 寄存器中有两个 DTOG 位，但只有一个位可供 USB 外设用于数据和缓冲区排序（由于双缓冲功能所要求的单向限制），因此应用软件可使用另一个位显示当前使用哪个缓冲区。这种新的缓冲区标志称为 SW_BUF。下表给出了对于“发送”和“接收”双缓冲批量端点，USB_CHEPnR 寄存器位与 DTOG/SW_BUF 定义之间的对应关系。

表 331. 双缓冲缓冲区标志定义

缓冲区标志	“发送”端点	“接收”端点
DTOG	DTOGTX（USB_CHEPnR 位 6）	DTOGRX（USB_CHEPnR 位 14）
SW_BUF	USB_CHEPnR 位 14	USB_CHEPnR 位 6

USB 外设当前使用的存储器缓冲区由 DTOG 缓冲区标志定义，而应用软件当前使用的缓冲区由 SW_BUF 缓冲区标志定义。对于“发送”和“接收”这两种情况，缓冲区标志值与使用的数据包缓冲区之间的关系是相同的，如下表所列。

表 332. 批量双缓冲存储器缓冲区用法

端点类型	DTOG	SW_BUF	USB 外设使用的数据包缓冲区	应用软件使用的数据包缓冲区
发送 (IN)	0	1	USB_CHEP_TXRXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。	USB_CHEP_RXTXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
	1	0	USB_CHEP_RXTXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。	USB_CHEP_TXRXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
	0	0	无 (1)	USB_CHEP_TXRXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
	1	1	无(1)	USB_CHEP_RXTXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
接收 (OUT)	0	1	USB_CHEP_RXTXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。	USB_CHEP_TXRXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。
	1	0	USB_CHEP_TXRXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。	USB_CHEP_RXTXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。
	0	0	无(1)	USB_CHEP_RXTXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。
	1	1	无(1)	USB_CHEP_TXRXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。

1. 端点处于 NAK 状态。

批量端点的双缓冲功能通过以下方式激活：

- 向 USB_CHEPnR 寄存器的 UTYPE 位域写入 00，以将端点定义为批量，
- 将同一寄存器中的 EPKIND 位置 1 (DBL_BUF)。

应用软件负责根据使用的第一个缓冲区初始化 DTOG 和 SW_BUF 位；考虑到这两位具有特殊的仅-位翻转属性，必须执行该操作。如果第一个事务在 DBL_BUF 置 1 后结束，则将触发双缓冲批量端点的特殊流控制，该特殊流控制用于寻址到该端点的所有其他事务，直到 DBL_BUF 保持置 1。每个事务结束时，所寻址端点 USB_CHEPnR 寄存器的 VTRX 或 VTTX 位将置 1，具体取决于使能的方向。同时，USB_CHEPnR 寄存器中受影响的 DTOG 位由硬件翻转，从而使 USB 外设缓冲区在完全不依赖软件的情况下实现交换。与普通事务以及 DBL_BUF 置 1 后的第一个事务不同，STAT 位对不受事务终止的影响，其值保持为 11 (VALID)。不过，当接收到新事务的令牌数据包时，如果检测到 USB 外设和应用软件之间存在缓冲区冲突（此时，DTOG 和 SW_BUF 的值相同，请参见 *第 1524 页的表 332*），则实际的端点状态将被屏蔽为 10 (NAK)。应用软件通过清零中断标志并开始对完成的事务进行任何所需处理来响应 CTR 事件通知。当应用软件数据包缓冲区使用完毕后，软件将翻转 SW_BUF 位，向该位写入 1 以通知 USB 外设该缓冲区当前可用。通过这种方式，不应答事务的数量仅受应用软件准备事务数据的时间限制：如果准备时间短于在 USB 总线上完成一

个事务所需的时间，则将不会发生因流控制引起的重新发送，实际传输速率将仅受主机 PC 的限制。

应用软件可始终覆盖为双缓冲批量端点实现的特殊流控制，向相关 USB_CHEPnR 寄存器的 STAT 位对写入 11 (VALID) 以外的明确状态。在这种情况下，USB 外设将始终使用编程的端点状态，而与缓冲区使用情况无关。

40.5.4 双缓冲通道：主机模式下的使用

在主机模式下，双缓冲通道的底层发送和接收方法与设备模式下相同。

与设备模式表类似，在下面的新的*表 333：批量双缓冲存储器缓冲区用法（主机模式）* 中，列出了 OUT 和 IN 令牌的编程设置。

表 333. 批量双缓冲存储器缓冲区用法（主机模式）

端点类型	DTOG	SW_BUF	USB 外设使用的数据包缓冲区	应用软件使用的数据包缓冲区
发送 (OUT)	0	1	USB_CHEP_TXRXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。	USB_CHEP_RXTXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
	1	0	USB_CHEP_RXTXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。	USB_CHEP_TXRXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
	0	0	无 (1)	USB_CHEP_TXRXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
	1	1	无 (1)	USB_CHEP_RXTXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
接收 (IN)	0	1	USB_CHEP_RXTXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。	USB_CHEP_TXRXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。
	1	0	USB_CHEP_TXRXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。	USB_CHEP_RXTXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。
	0	0	无 (1)	USB_CHEP_RXTXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。
	1	1	无 (1)	USB_CHEP_TXRXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。

1. 端点处于 NAK 状态。

40.5.5 设备模式下的同步传输

USB 标准支持需要固定且精确的数据生产/消费频率的全速外设，这种通信定义为“同步”。该数据的典型示例包括：音频采样、压缩视频流以及通常对传输频率精度有严苛要求的任何一种采样数据。当端点在枚举阶段定义为“同步”时，主机将在帧中分配所需带宽并在每个帧中严密地传输一个 IN 或 OUT 帧，具体取决于端点方向。为限制带宽需求，同步通信中不允许重新发送失败的事务，这将导致以下情况：同步事务没有握手阶段，数据包之后不等待或发送任何 ACK 包。出于同样的原因，同步传输不支持数据位翻转，始终使用 DATA0 PID 来启动任何数据包。

端点的同步行为通过将其 USB_CHEPnR 寄存器中的 UTYPE 位设置为 10 来选择；由于没有握手阶段，STATRX/STATTX 位对的合法值只有 00 (DISABLED) 和 11 (VALID)，任何其他值都将产生不符合 USB 标准的结果。同步端点通过实现双缓冲来简化应用软件的开发，使用“发送”和“接收”数据包存储区在每个事务成功时管理缓冲区交换，以便在 USB 外设填充一个缓冲区时，始终有另一个完整的缓冲区可供应用软件使用。

根据*表 334*，USB 外设当前使用的存储器缓冲区由与端点方向相关的 DTOG 位定义（DTOGRX 用于“接收”同步端点，DTOGTX 用于“发送”同步端点，二者均位于相关的 USB_CHEPnR 寄存器中）。

表 334. 同步存储器缓冲区的用法

端点类型	DTOG 位值	应用软件使用的数据包缓冲区	应用软件使用的数据包缓冲区
发送 (IN)	0	USB_CHEP_TXRXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。	USB_CHEP_RXTXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
	1	USB_CHEP_RXTXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。	USB_CHEP_TXRXBD_0 (ADDR_TX / COUNT_TX) 缓冲区描述表存储单元。
接收 (OUT)	0	USB_CHEP_RXTXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。	USB_CHEP_TXRXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。
	1	USB_CHEP_TXRXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。	USB_CHEP_RXTXBD_0 (ADDR_RX / COUNT_RX) 缓冲区描述表存储单元。

与双缓冲批量端点一样，用于实现同步端点的 USB_CHEPnR 寄存器被强制用作单向寄存器。如果需要将同步端点使能同时用于发送和接收，则必须使用两个 USB_CHEPnR 寄存器。

应用软件负责根据使用的第一个缓冲区初始化 DTOG 位；考虑到这两位具有特殊的仅-位翻转属性，必须执行该操作。每个事务结束时，所寻址端点 USB_CHEPnR 寄存器的 VTRX 或 VTTX 位将置 1，具体取决于使能的方向。同时，USB_CHEPnR 寄存器中受影响的 DTOG 位由硬件翻转，从而使缓冲区在完全不依赖软件的情况下实现交换。STAT 位对不受事务完成的影响，因为同步传输缺少握手阶段，没有可用的流控制，端点始终保持 11 (VALID)。同步 OUT 传输期间出现的 CRC 错误或缓冲区上溢条件仍被视为正确事务，它们始终会触发 VTRX 事件。但是，CRC 错误仍会将 USB_ISTR 寄存器中的 ERR 位置 1 以向软件通知可能的数据损坏。

40.5.6 主机模式下的同步传输

从主机的角度而言，同步数据包是由主机帧调度器逐帧发出或请求的。没有 NAK/ACK 协议，也不重新发送数据或令牌。

该机制基于与设备模式非常相似的表。参见下面的*表 335* 以了解 DTOG 位缓冲区与缓冲区使用之间的关系。

表 335. 同步存储器缓冲区的用法

端点类型	DTOG 位值	应用软件使用的数据包缓冲区	应用软件使用的数据包缓冲区
发送 (OUT)	0	USB_CHEP_TXRXBD_0 (ADDRn_TX / COUNTn_TX) 缓冲区描述表存储单元。	USB_CHEP_RXTXBD_0 (ADDRn_TX / COUNTn_TX) 缓冲区描述表存储单元。
	1	USB_CHEP_RXTXBD_0 (ADDRn_TX / COUNTn_TX) 缓冲区描述表存储单元。	USB_CHEP_TXRXBD_0 (ADDRn_TX / COUNTn_TX) 缓冲区描述表存储单元。
接收 (IN)	0	USB_CHEP_RXTXBD_0 (ADDRn_RX / COUNTn_RX) 缓冲区描述表存储单元。	USB_CHEP_TXRXBD_0 (ADDRn_RX / COUNTn_RX) 缓冲区描述表存储单元。
	1	USB_CHEP_TXRXBD_0 (ADDRn_RX / COUNTn_RX) 缓冲区描述表存储单元。	USB_CHEP_RXTXBD_0 (ADDRn_RX / COUNTn_RX) 缓冲区描述表存储单元。

端点的同步行为通过将其 USB_CHEPnR 寄存器中的 UTYPE 位设置为 10 来选择；由于没有握手阶段，STATRX/STATTX 位对的合法值只有 00 (DISABLED) 和 11 (VALID)，

如同在设备模式下一样，该机制允许自动翻转 DTOG 位。请注意，在主机模式下，在翻转的同时，已完成缓冲区的 STATTX 或 STATRX 会自动设置为 DISABLED，这样，在以后通过将 STATTX 或 STATRX 设置为 11 (VALID) 而重新使能缓冲区之前，就可以访问缓冲区。

40.5.7 挂起/恢复事件

USB 标准定义了一种特殊的外设状态，称为 SUSPEND，在该状态下，USB 总线的平均电流消耗不得超过 2.5 mA。该要求对于总线供电设备至关重要，自供电设备则无需遵守这一严格的功耗限制。在挂起模式下，主机 PC 不在 USB 总线上发送任何通信数据并持续 3 ms 以上的时间，以此发送通知：因为在正常操作期间必须每隔 1 ms 发送一次 SOF 数据包，如果 USB 外设检测到 3 个连续的 SOF 数据包缺失，则会将这种情况视为主机 PC 的挂起请求，然后将 USB_ISTR 寄存器中的 SUSP 位设置为 1，从而引起中断（如果使能）。设备挂起后，可通过所谓的 RESUME 序列恢复其正常操作，该序列可从主机 PC 或直接从外设本身启动，但始终通过主机 PC 终止。挂起的 USB 外设必须仍能够检测到 RESET 序列，将此事件视为正常的 USB 复位事件并对其做出响应。

挂起 USB 外设所使用的实际步骤取决于设备，因为对于不同的设备构成，降低总功耗所需的操作也会有所不同。

下面给出了典型挂起步骤的简单说明，重点关注与 USB 外设的 SUSP 通知对应的应用软件程序中与 USB 相关的方面：

1. 将 USB_CNTR 寄存器中的 SUSPEN 位置 1。此操作将激活 USB 外设中的挂起模式。激活挂起模式后，将立即禁止“接收到 SOF 时检查”功能，以避免在 USB 挂起时发出任何其他 SUSP 中断。

2. 消除或降低 USB 外设以外的模块中的任何静态功耗。
3. 将 USB_CNTR 寄存器中的 SUSPRDY 位置 1，以消除模拟 USB 收发器中的静态功耗，但使这些收发器能够检测恢复活动。
4. 可选择关闭外部振荡器或器件 PLL 以停止设备内的任何活动。

如果在设备处于 SUSPEND 模式时发生 USB 事件，则必须调用 RESUME 程序来恢复标称时钟和正常的 USB 行为。当唤醒事件是 USB 复位序列时，必须特别注意确保此过程不超过 10 ms（更多详细信息，请参见“通用串行总线规范”）。在 USB 外设挂起时启动恢复或复位序列会将 USB_CNTR 寄存器中的 SUSPRDY 位异步清零。即使此事件能够触发 WKUP 中断（如果使能），也必须仔细评估是否使用中断响应程序，因为系统时钟重启会引入较长的延时；为缩短重新激活标称时钟前的延时，建议紧接着挂起程序末尾放置恢复程序，以便在系统时钟重启时，立即执行相应代码。为防止 ESD 放电或任何其他类型的噪声唤醒系统（退出挂起模式是一个异步事件），挂起期间需在数据线上激活合适的模拟滤波器；滤波器的宽度约为 70 ns。

以下是恢复程序必须解决的一系列操作：

1. 可选择接通外部振荡器和/或器件 PLL。
2. 将 USB_CNTR 寄存器中的 SUSPEN 位清零。
3. 如果必须识别恢复触发事件，则可根据*表 336* 使用 USB_FNR 寄存器中的位 RXDP 和 RXDM，表中还列出了所有情况下的目标软件操作。如果需要，可通过监视上述位的状态（即检查它们何时达到 10 配置，该配置表示空闲总线状态）来检测恢复或复位序列是否结束；此外，当复位序列结束时，USB_ISTR 寄存器中的 RST_DCON 位将置 1，从而发出中断（如果使能），该中断必须照常处理。

表 336. 恢复事件检测

[RXDP,RXDM] 状态	唤醒事件	所需的软件恢复操作
“00”	根复位	无
“10”	无（总线上的噪声）	返回挂起模式
“01”	根恢复	无
“11”	不允许（总线上的噪声）	返回挂起模式

设备可能需要退出挂起模式来响应与 USB 协议不是直接相关的特定事件（例如，通过移动鼠标唤醒整个系统）。在这种情况下，可通过如下方式启动恢复序列：将 USB_CNTR 寄存器中的 L2RES 位置 1，然后在一段介于 1 ms 和 15 ms 之间的时间间隔后将其复位为 0（此时间间隔可使用 ESOF 中断来定时，当系统时钟以标称频率运行时，周期为 1 ms）。L2RES 位清零后，恢复序列将由主机 PC 完成，其结束可再次使用 USB_FNR 寄存器中的 RXDP 和 RXDM 位来监视。

注意： *L2RES 位仍必须仅在 USB 外设置于挂起模式后（即 USB_CNTR 寄存器中的 SUSPEN 位置 1）使用。*

主机模式下挂起和恢复

上一节中已介绍了挂起和恢复机制的基本内容。

对主机而言，是通过写入 USB_CNTR 中的 SUSPEN 位来进入挂起状态的。当确认挂起条目时，SUSPRDY（也在 USB_CNTR 中）即置 1。

处于挂起状态后，当应用程序想要恢复总线时，可以通过将 USB_CNTR 中的 L2RES 位置 1 来完成。

下面的*表 337* 指出了在唤醒事件后建议的不同操作。根据唤醒事件后不同的线路状态，列出相应事件的解释和建议的行为。请注意，与之前所示的器件表相比，上表有所扩充，因为主机可能会遇到全速和低速器件，对于挂起和恢复，这两类器件使用不同的线路状态。

表 337. 针对主机恢复事件检测

[RXDP,RXDM] 状态	唤醒事件	所需的软件恢复操作
“00”	不允许（总线上的噪声）	返回挂起模式
“10”	全速器件：不允许（总线上的噪声） 低速器件器件远程唤醒恢复	无
“01”	全速器件：器件远程唤醒恢复 低速器件 不允许（总线上的噪声）	无
“11”	不允许（总线上的噪声）	返回挂起模式

40.6 USB 和 USB SRAM 寄存器

USB 外设寄存器可分为以下几组：

- 通用寄存器：中断和控制寄存器
- 端点/通道寄存器：端点/通道配置和状态

USB SRAM 寄存器包括：

- 缓冲区描述符表：数据缓冲区所在的数据包存储器的存储单元（有关 USB SRAM 基址，请参见*第 2.2 节：存储器结构*）。

所有寄存器地址均以相对于 USB 外设寄存器基址的偏移量表示，但缓冲区描述符表的存储单元除外，其起始地址位于 USB SRAM 基址。

有关寄存器说明中使用的缩写，请参见*第 63 页的第 1.2 节*。

外设寄存器可按字（32 位）进行访问。

40.6.1 通用寄存器

这类寄存器会影响 USB 外设的一般行为，包括定义工作模式、中断处理、设备地址以及访问主机 PC 所更新的当前帧数。

USB 控制寄存器 (USB_CNTR)

USB control register

偏移地址：0x40

复位值：0x0000 0003

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HOST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DDISCM	THR512M
rw														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTRM	PMAOVRM	ERRM	WKUPM	SUSPM	RST_DCONM	SOFM	ESOFM	L1REQM	Res.	L1RES	L2RES	SUSPEN	SUSPRDY	PDWN	USBRST
rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	r	rw	rw

位 31 **HOST**：HOST 模式 (HOST mode)

HOST 位在主机或设备 USB 操作模式之间进行选择。必须在通过功能使能位使能 USB 外设之前将其置 1。

0：USB 设备功能

1：USB 主机功能

位 30:18 保留，必须保持复位值。

位 17 **DDISCM**：设备断开屏蔽 (Device disconnection mask)

– 主机模式

0：禁止设备断开中断

1：使能设备断开中断

位 16 **THR512M**：512 字节阈值中断屏蔽 (512 byte threshold interrupt mask)

0：禁止 512 字节阈值中断

1：使能 512 字节阈值中断

位 15 **CTRM**：正确传输中断屏蔽 (Correct transfer interrupt mask)

0：禁止正确传输 (CTR) 中断。

1：使能 CTR 中断，当 USB_ISTR 寄存器中的相应位置 1 时，将生成中断请求。

位 14 **PMAOVRM**：数据包存储区上溢/下溢中断屏蔽 (Packet memory area over / underrun interrupt mask)

0：禁止 PMAOVR 中断。

1：使能 PMAOVR 中断，当 USB_ISTR 寄存器中的相应位置 1 时，将生成中断请求。

位 13 **ERRM**：错误中断屏蔽 (Error interrupt mask)

0：禁止 ERR 中断。

1：使能 ERR 中断，当 USB_ISTR 寄存器中的相应位置 1 时，将生成中断请求。

位 12 **WKUPM**：唤醒中断屏蔽 (Wake-up interrupt mask)

0：禁止 WKUP 中断。

1：使能 WKUP 中断，当 USB_ISTR 寄存器中的相应位置 1 时，将生成中断请求。

位 11 **SUSPM**：挂起模式中断屏蔽 (Suspend mode interrupt mask)

0：禁止挂起模式请求 (SUSP) 中断。

1：使能 SUSP 中断，当 USB_ISTR 寄存器中的相应位置 1 时，将生成中断请求。

位 10 **RST_DCONM**：USB 复位请求（设备模式）或设备连接/断开（主机模式）中断屏蔽 (USB reset request (Device mode) or device connect/disconnect (Host mode) interrupt mask)

0：禁止 RESET 中断。

1：使能 RESET 中断，当 USB_ISTR 寄存器中的相应位置 1 时，将生成中断请求。

位 9 **SOFM**：帧起始中断屏蔽 (Start of frame interrupt mask)

0：禁止 SOF 中断。

1：使能 SOF 中断，当 USB_ISTR 寄存器中的相应位置 1 时，将生成中断请求。

位 8 **ESOFM**：预期帧起始中断屏蔽 (Expected start of frame interrupt mask)

0：禁止预期帧起始 (ESOF) 中断。

1：使能 ESOF 中断，当 USB_ISTR 寄存器中的相应位置 1 时，将生成中断请求。

位 7 **L1REQM**：LPM L1 状态请求中断屏蔽 (LPM L1 state request interrupt mask)

0：禁止 LPM L1 状态请求 (L1REQ) 中断。

1：使能 L1REQ 中断，当 USB_ISTR 寄存器中的相应位置 1 时，将生成中断请求。

位 6 保留，必须保持复位值。

位 5 **L1RES**：L1 远程唤醒/恢复驱动器 (L1 remote wake-up / resume driver)

– 设备模式

软件将该位置 1 以向主机发送 LPM L1 50 µs 远程唤醒信号。信号发送结束后，该位通过硬件清零。

0：无影响

1：向主机发送 50 µs 远程唤醒信号

位 4 **L2RES**：L2 远程唤醒/恢复驱动器 (L2 remote wake-up / resume driver)

– 设备模式

微控制器可将该位置 1 以向主机发送远程唤醒信号。它必须根据 USB 规范激活并持续 1 ms 到 15 ms 的时间，经过这段时间后，主机 PC 便可驱动恢复序列直到结束。

– 主机模式

软件将该位置 1 以向设备发送恢复信号。

软件清零该位以将恢复结束发送至设备并重新启动 SOF 生成。

在远程唤醒的情况下，该位将在 WAKEUP 中断之后置 1。

0：无影响

1：向设备发送 L2 恢复信号

位 3 **SUSPEN**：挂起状态使能 (Suspend state enable)

– 条件：设备模式

软件可在接收到 SUSP 中断时将该位置 1，当持续 3 ms 未从 USB 外设接收到任何通信数据时，将发出 SUSP 中断。当接收到 L1REQ 中断并发送肯定应答时，软件也可以将该位置 1。

挂起状态在内部传播后，所有设备活动会立即停止，USB 时钟被门控，USB 收发器置为低功耗模式，SUSPRDY 位由硬件置 1。如果设备应用想要通过停止 USB 时钟源并将微控制器置为停止模式来获得更多节能（如总线供电设备应用的情况），它必须先等待几个周期，才能看到 SUSPRDY = 1 确认挂起请求。

该位在 WAKEUP 标志置 1 的同时由硬件清零。

0：无影响

1：进入 L1/L2 挂起状态

– 条件：主机模式

当主机应用程序没有对下一帧进行任何安排并且想要进入长期节能状态时，软件可以将该位置 1。置 1 后，它会立即停止 SOF 生成和任何其他主机活动，门控 USB 时钟并将收发器置为低功耗模式。如果在 SUSPEN 置 1 时有任何 USB 事务正在进行，则在当前事务结束时进入挂起状态。

挂起状态在内部传播并生效后，SUSPRDY 位会立即置 1。如果主机应用想要通过停止 USB 时钟源并将微控制器置为 STOP 模式来获得更多节能，它必须先等待几个周期，才能看到 SUSPRDY=1 确认挂起请求。

该位在 WAKEUP 标志置 1 的同时由硬件清零。

0：无影响

1：进入 L1/L2 挂起状态

位 2 **SUSPRDY**：挂起状态生效 (Suspend state effective)

只要通过 SUSPEN 控制进入的挂起状态在内部生效，该位就会立即由硬件置 1。在该状态下，USB 活动挂起，USB 时钟受到门控，通过禁用差分接收器将收发器置为低功耗模式。仅异步唤醒逻辑和单端接收器保持活动状态，以检测远程唤醒或恢复事件。

软件必须轮询该位以确认在进入任何 STOP 模式之前将其置 1。

该位在 WAKEUP 标志置 1 的同时由硬件清零。

0：正常工作

1：挂起状态

位 1 **PDWN**：掉电 (Power down)

当出于某种原因而需要完全禁止 USB 外设时，该位用于完全关闭所有与 USB 相关的模拟部分。该位置 1 时，USB 外设将与收发器断开，并且无法使用。

0：退出掉电模式

1：进入掉电模式

位 0 **USBRST**：USB 复位 (USB Reset)

– 条件：设备模式

软件可以将该位置 1 以复位 USB 内核，就像在 USB 上接收 RESET 信号时一样。USB 外设响应 RESET，将其内部协议状态机复位。在 RST_DCON 位清零之前，将始终禁止发送和接收。所有配置寄存器均不复位：微控制器必须明确清零这些寄存器（这是为了确保能够安全提供 RST_DCON中断，以及能够完成 RESET 后紧跟的任何事务）。功能地址和端点寄存器由 USB 复位事件复位。

0：无影响

1：USB 内核正在重置

– 条件：主机模式

软件将该位置 1 以驱动总线上的 USB 复位状态并初始化设备。一旦该位由软件清零，USB 复位就会终止。

0：无影响

1：已驱动 USB 复位

USB 中断状态寄存器 (USB_ISTR)

USB interrupt status register

偏移地址：0x44

复位值：0x0000 0000

该寄存器包含所有中断源的状态，允许应用软件确定哪些事件引起了中断请求。

此寄存器的高半部分包含多个单独位，每个位表示一个特定事件。这些位会在发生相关事件时由硬件置 1；如果 USB_CNTR 寄存器中的相应位置 1，将生成通用中断请求。中断程序用于检查每个位，它将执行所有必要的操作并在最终将已处理的位清零。如果其中的任一位未清零，则中断仍被视为处于挂起状态，中断线将再次保持高电平。如果多个位同时置 1，则仅生成一个中断。

端点/通道事务完成可通过一种不同的方式进行处理，以缩短中断响应延时。端点/通道成功完成一个事务后，硬件会立即将 CTR 位置 1，从而在 USB_CNTR 中的相应位置 1 时生成通用中断请求。端点/通道专用中断状态的激活与 USB_CNTR 寄存器中的 CTRM 位无关。中断状态均保持激活，直到软件将相应 USB_CHEPnR 寄存器中的挂起位清零为止（CTR 位实际上是一个只读位）。对于与端点/通道相关的中断，软件可使用事务方向 (DIR) 和 IDN 只读位来确定哪个端点/通道发出最后一个中断请求以及调用相应的中断服务程序。

用户可通过指定软件在中断服务程序中检查 USB_ISTR 位的顺序来选择同时挂起的多个 USB_ISTR 事件的相对优先级。仅清零与事件相关且已被处理的位。在服务程序结束时，将请求另一个中断来处理剩余中断状态。

为避免意外清零某些位，建议通过加载指令清零相应位，即向所有不得改变的位写入 1，向所有待清零的位写入 0（这些位只能通过软件清零）。由于在读操作和写操作之间，另一个位会由硬件置 1，并将在设备有时间处理事件之前由下一次写操作清零，因此必须避免读-修改-写周期。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	LS_DCON	DCON_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DDISC	THR512
	r	r												rc_w0	rc_w0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTR	PMAOVR	ERR	WKUP	SUSP	RST_DCON	SOF	ESOF	L1REQ	Res.	Res.	DIR	IDN[3:0]			
r	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0			r	r	r	r	r

位 31 保留，必须保持复位值。

位 30 **LS_DCON**：已连接低速设备 (Low speed device connected)

– 主机模式：

检测到 LS 设备连接时，该位由硬件置 1。从未连接状态开始，连续 22 个 USB 时钟 (48 Mhz) 周期对 LS J 状态进行采样后，会发出设备连接信号。

位 29 **DCON_STAT**：设备的连接状态 (Device connection status)

– 主机模式：

该位包含有关设备连接状态的信息。当 LS/FS 设备连接到主机时，它由硬件置 1，而当设备断开连接时，它会被复位。

0：未连接设备

1：FS 或 LS 设备连接到主机

位 28:18 保留，必须保持复位值。

位 17 **DDISC**：设备连接 (Device connection)

– 主机模式

检测到设备连接时，该位置 1。该位为读/写位，但只能写入 0，写入 1 不起作用。

位 16 **THR512**：512 字节阈值中断 (512 byte threshold interrupt)

当同步传输期间发送或接收了 512 个字节时，该位由硬件置 1。该位为读/写位，但只能写入 0，写入 1 不起作用。请注意，没有信息用来指示关联的通道/端点，但实际上只能支持一个具有如此大数据包的 ISO 端点/通道，因此就是该通道。

位 15 **CTR**：在主机模式下传输已完成 (Completed transfer in host mode)

该位由硬件置 1，用于指示端点/通道已成功完成事务；软件可通过 DIR 和 IDN 位确定哪个端点/通道请求中断。该位为只读。

位 14 **PMAOVR**：数据包存储区上溢/下溢 (Packet memory area over / underrun)

当微控制器无法及时响应 USB 存储器请求时，该位将置 1。USB 外设按以下方式处理该事件：接收期间不发送 ACK 握手包，发送期间在发送的流中强制位填充错误；在这两种情况下，主机都将重试事务。正常操作期间，绝对不能出现 PMAOVR 中断。由于主机重试失败的事务，应用软件将有机会在此中断处理期间加快设备操作，以为下一次事务重试做准备；但这在同步传输期间不会发生（无论如何也不会重试任何同步事务），因此会导致数据丢失。该位为读/写位，但只能写入 0，写入 1 不起作用。

位 13 **ERR**：错误 (Error)

只要出现下列其中一个错误，此标志便会置 1。

NANS： 无应答。主机响应已超时。

CRC： 循环冗余校验错误。接收的 CRC（令牌或数据中）之一有错误。

BST： 位填充错误。在 PID、数据和/或 CRC 中的任一位置上检测到了位填充错误。

FVIO： 帧格式违例。接收到了非标准帧（EOP 未处于正确位置、令牌序列错误等）。

由于 USB 外设和 PC 主机在出现错误时采用全透明方式管理重新发送，因此 USB 软件通常可忽略错误。该中断在软件开发阶段会很有用，可用于监视 USB 总线上的传输质量以及向用户指示可能的问题（例如，连接器松动、环境过于嘈杂、USB 电缆中的导线断裂等）。该位为读/写位，但只能写入 0，写入 1 不起作用。

位 12 **WKUP**：唤醒 (Wake-up)

当在挂起模式期间检测到用于唤醒 USB 外设的活动时，该位将由硬件置 1。该事件将异步清零 CTLR 寄存器中的 SUSPRDY 位并激活 USB_WAKEUP 线，这可用于通知设备的其余部分（例如，唤醒单元）恢复过程即将开始。该位为读/写位，但只能写入 0，写入 1 不起作用。

位 11 **SUSP**：挂起模式请求 (Suspend mode request)

– 设备模式

当持续 3 ms 未接收到通信数据时，该位将由硬件置 1，用于指示来自 USB 总线的挂起模式请求。挂起状态检查将在任何 USB 复位后立即使能，并在挂起模式激活 (SUSPEN=1) 时由硬件禁用，直到恢复序列结束。该位为读/写位，但只能写入 0，写入 1 不起作用。

位 10 **RST_DCON**：USB 复位请求（设备模式）或设备连接/断开连接（主机模式）(USB reset request (Device mode) or device connect/disconnect (Host mode))

– 设备模式

当主机释放 USB 复位且总线返回空闲状态时，该位由硬件置 1。在连续 60 个 SE0 周期的采样后，会在内部检测到 USB 复位状态。

– 主机模式

当检测到设备连接或设备断开连接时，该位由硬件置 1。从未连接状态开始，连续 22 个周期对 J 状态采样后，将发出设备连接信号。从连接状态开始连续 22 个位时间看到 SE0 状态后，将发出设备断开连接信号。

位 9 **SOF**：帧起始 (Start of frame)

该位指示新 USB 帧的开始，并在 SOF 数据包通过 USB 总线到达时置 1。中断服务程序可监视 SOF 事件，以获得 USB 主机的 1 ms 同步事件以及安全读取在接收到 SOF 数据包时更新的 USB_FNR 寄存器（这对于同步应用非常有用）。该位为读/写位，但只能写入 0，写入 1 不起作用。

位 8 **ESOF**：预期帧起始 (Expected start of frame)

– 设备模式

当未接收到预期的 SOF 数据包时，该位由硬件置 1。主机每隔 1 ms 发送一个 SOF 数据包，但如果器件未按预期接收到数据包，则挂起定时器将发出此中断。如果连续生成三个 ESOF 中断（例如三个 SOF 数据包丢失）且两两中断之间没有任何通信数据，则将生成 SUSP 中断。即使在挂起定时器尚未锁定时丢失 SOF 数据包，该位也会置 1。该位为读/写位，但只能写入 0，写入 1 不起作用。

位 7 **L1REQ**：LPM L1 状态请求 (LPM L1 state request)

– 设备模式

当成功接收并应答进入 L1 状态的 LPM 命令时，该位由硬件置 1。该位为读/写位，但只能写入 0，写入 1 不起作用。

位 6:5 保留，必须保持复位值。

位 4 **DIR**：事务方向 (Direction of transaction)

该位将由硬件根据已生成中断请求的成功事务的方向写入。
如果 DIR 位为 0，则与中断端点相关的 USB_CHEPnR 寄存器中的 VTTX 位置 1。中断事务的类型为 IN（USB 外设将数据发送给主机 PC）。
如果 DIR 位为 1，则与中断端点相关的 USB_CHEPnR 寄存器中的 VTRX 位置 1，或者 VTTX/VTRX 位均置 1。中断事务的类型为 OUT（USB 外设从主机 PC 接收数据），或者两个挂起事务均等待处理。
此信息可供应用软件用于访问与触发事务相关的 USB_CHEPnR 寄存器的位，因为它表示中断挂起的方向。该位为只读。

位 3:0 **IDN[3:0]**：设备端点/主机通道标识号 (Device Endpoint / host channel identification number)

这些位将由硬件根据已生成中断请求的主机通道或设备端点编号写入。如果有多个端点/通道事务挂起，则硬件将写入与优先级最高的端点/通道相关的标识符，优先级的定义方式如下：定义两个级别，同步和双缓冲批量通道/端点的优先级最高，其他次之。如果来自同一组的多个端点/通道请求一个中断，则 USB_ISTR 寄存器中的 IDN 位将按照请求寄存器的编号升序分配，即 CHEP0R 的优先级最高，CHEP1R 次之，依此类推。应用软件可按照此优先级方案为每个端点/通道分配一个寄存器，以便对同时发生的多个端点/通道请求进行适当排序。这些位为只读。

USB 帧数寄存器 (USB_FNR)

USB frame number register

偏移地址：0x48

复位值：0x0000 0XXX（其中 X 未定义）

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDP	RXDM	LCK	LSOF[1:0]		FN[10:0]										
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 保留，必须保持复位值。

位 15 **RXDP**：接收数据 + 线状态 (Receive data + line status)

该位可用于观察所接收的数据正上行端口数据线的状态。可以在挂起结束程序期间使用该位来帮助确定唤醒事件。

位 14 **RXDM**：接收数据 - 线状态 (Receive data - line status)

该位可用于观察所接收的数据负上行端口数据线的状态。可以在挂起结束程序期间使用该位来帮助确定唤醒事件。

位 13 **LCK**：锁定 (Locked)

– 设备模式

当在 USB 复位状态结束后或 USB 恢复序列结束后接收到至少两个连续的 SOF 数据包时，该位将由硬件置 1。锁定后，帧定时器将保持此状态，直到发生 USB 复位或 USB 挂起事件。

位 12:11 **LSOF[1:0]**：SOF 丢失 (Lost SOF)

– 设备模式

这些位在生成 ESOF 中断时由硬件写入，用于对丢失的连续 SOF 数据包进行计数。接收到 SOF 数据包时，这些位将被清零。

位 10:0 **FN[10:0]**：帧编号 (Frame number)

该位域包含最后接收的 SOF 数据包中的 11 位帧数。主机每发送一个帧，帧数便会递增，帧数对于同步传输十分有用。该位域在生成 SOF 中断时更新。

USB 设备地址 (USB_DADDR)

USB Device address

偏移地址：0x4C

复位值：0x0000 0000

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

位 31:8 保留，必须保持复位值。

位 7 **EF**：使能功能 (Enable function)

该位由软件置 1，用于使能 USB 设备。此设备的地址包含在下面的 ADD[6:0] 位中。如果该位为 0，则无论 USB_CHEPnR 寄存器的设置为何，都不会处理任何事务。

位 6:0 **ADD[6:0]**：设备地址 (Device address)

– 设备模式

这些位包含枚举过程中由主机 PC 分配的 USB 功能地址。该位域和相关 USB_CHEPnR 寄存器中的端点/通道地址 (EA) 位域均必须与 USB 令牌中包含的信息匹配，才能处理所需端点的事务。

– 主机模式

这些位包含随 LPM 事务发送的地址

LPM 控制和状态寄存器 (USB_LPMCSR)

LPM control and status register

偏移地址：0x54

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BESL[3:0]				REM WAKE	Res.	LPM ACK	LPM EN
								r	r	r	r	r		rw	rw

位 31:8 保留，必须保持复位值。

位 7:4 **BESL[3:0]**：BESL 值 (BESL value)

– 设备模式

这些位包含通过最后应答的 LPM 令牌接收的 BESL 值

位 3 **REMWAKE**：bRemoteWake 值 (bRemoteWake value)

– 设备模式

该位包含通过最后应答的 LPM 令牌接收的 bRemoteWake 值

位 2 保留，必须保持复位值。

位 1 **LPMACK**：LPM 令牌确认使能 (LPM token acknowledge enable)

– 设备模式：

0：有效 LPM 令牌将为 NYET。
1：有效 LPM 令牌将为 ACK。
只有成功完成 LPM 事务后才会返回 NYET/ACK：
EXT 令牌和 LPM 令牌中均无错误（否则返回 ERROR）
接收到有效 bLinkState = 0001B (L1)（否则返回 STALL）

位 0 **LPMEN**：LPM 支持使能 (LPM support enable)

– 设备模式

该位由软件置 1，用于在 USB 设备内使能 LPM 支持。如果该位为 0，则不会处理任何 LPM 事务。

电池充电检测器 (USB_BCDR)

Battery charging detector

偏移地址：0x58

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPPU_DPD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PS2 DET	SDET	PDET	DC DET	SDEN	PDEN	DCD EN	BCD EN
rw								r	r	r	r	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15 **DPPU_DPD**：DP 上拉/DPDM 下拉 (DP pull-up / DPDM pull-down)

– 设备模式

该位由软件置 1，用于使能 DP 线上的内置上拉。必要时，可通过用户软件将该位清零以向主机发出断开信号。

– 主机模式

该位由软件置 1，用于使能 DP 和 DM 线上的内置上拉。

位 14:8 保留，必须保持复位值。

位 7 **PS2DET**：DM 上拉检测状态 (DM pull-up detection status)

– 设备模式

该位只在 PD 期间有效，用于指示 DM 电压电平与 V_{LGC} 阈值之间的比较结果。正常情况下，DM 电平必须低于此阈值。如果 DM 电平高于此阈值，则意味着 DM 已外部拉为高电平。原因可能是连接到 PS2 端口（同时上拉 DP 和 DM 线）或某些不遵循 BCD 规范的专有充电器。

0：检测到正常端口（连接到 SDP、ACA、CDP 或 DCP）。

1：检测到 PS2 端口或专有充电器。

位 6 **SDET**：二次检测 (SD) 状态 (Secondary detection (SD) status)

– 设备模式

该位指示 SD 的结果。

0：检测到 CDP。

1：检测到 DCP。

位 5 **PDET**：初次检测 (PD) 状态 (Primary detection (PD) status)

– 设备模式

该位指示 PD 的结果。

0：未检测到 BCD 支持（连接到 SDP 或专有设备）。

1：检测到 BCD 支持（连接到 ACA、CDP 或 DCP）。

位 4 **DCDET**：数据接触点检测 (DCD) 状态 (Data contact detection (DCD) status)

– 设备模式

该位指示 DCD 的结果。

0：未检测到数据线接触。

1：检测到数据线接触。

位 3 **SDEN**：二次检测 (SD) 模式使能 (Secondary detection (SD) mode enable)

– 设备模式

该位由软件置 1，用于将 BCD 置于 SD 模式。若要正确工作，必须只选择一个检测模式（DCD、PD、SD 或 OFF）。

位 2 **PDEN**：初次检测 (PD) 模式使能 (Primary detection (PD) mode enable)

– 设备模式

该位由软件置 1，用于将 BCD 置于 PD 模式。若要正确工作，必须只选择一个检测模式（DCD、PD、SD 或 OFF）。

位 1 **DCDEN**：数据接触点检测 (DCD) 模式使能 (Data contact detection (DCD) mode enable)

– 设备模式

该位由软件置 1，用于将 BCD 置于 DCD 模式。若要正确工作，必须只选择一个检测模式（DCD、PD、SD 或 OFF）。

位 0 **BCDEN**：电池充电器检测 (BCD) 使能 (Battery charging detector (BCD) enable)

– 设备模式

该位由软件置 1，用于在 USB 设备内使能 BCD 支持。使能后，USB PHY 完全由 BCD 控制，无法用于正常通信。BCD 检测完成后，必须通过将该位清零使 BCD 置于 OFF 模式，以实现正常的 USB 操作。

主机通道/设备端点相关寄存器

这类寄存器的数量因 USB 外设设计要处理的端点数或主机通道数而异。USB 外设最多支持 8 个双向端点或主机通道。每个 USB 设备必须支持一个控制端点/通道，其地址（EA 位）必须设为 0。当使能多个端点/通道编号值相同的端点时，USB 外设将以一种未定义的方式工作。对于每个端点而言，USB_CHEPnR 寄存器可用于存储端点/通道特定信息。

USB 端点/通道 n 寄存器 (USB_CHEPnR)

USB endpoint/channel n register

偏移地址：0x00 + 0x4 * n，（n = 0 到 7）

复位值：0x0000 0000

当从 USB 总线接收到 USB 复位或通过 CTLR 寄存器中的 USBRST 位强制 USB 复位时，这些位也会复位；但 VTRX 和 VTTX 位除外，它们将保持不变，以避免丢失紧跟在 USB 复位事件后的数据包通知。每个端点/通道都有自己的 USB_CHEPnR 寄存器，其中 *n* 是端点/通道标识符。

由于在读操作和写操作之间，一些位会被硬件置 1，并将在 CPU 有时间检测变化之前由下一次写操作修改，因此必须避免这类寄存器上的读-修改-写周期。为此，受该问题影响的所有位都有一个“不变量”值，无需修改时必须使用该值。建议通过加载指令来修改这类寄存器，即向只能通过硬件修改的所有位写入其“不变量”值。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	THREE_ERR_RX[1:0]		THREE_ERR_TX[1:0]		ERR_RX	ERR_TX	LS_EP	NAK	DEVADDR[6:0]						
	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rw	rc_w0	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VTRX	DTOG RX	STATRX[1:0]		SETUP	UTYPE[1:0]		EP KIND	VTTX	DTOG TX	STATTX[1:0]		EA[3:0]			
rc_w0	t	t	t	r	rw	rw	rw	rc_w0	t	t	t	rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30:29 **THREE_ERR_RX[1:0]**：IN 事务的三个错误 (Three errors for an IN transaction)

– 主机模式

当 IN 事务的 USB 总线上连续出现 3 个事务错误时，该位由硬件置 1。同步事务不会生成 THREE_ERR_RX。软件只能将该位清零。

接收到的错误的编码：

00：接收到的错误少于 3 个。

01：接收到的错误超过 3 个，最后一个错误是超时错误。

10：接收到的错误超过 3 个，最后一个错误是数据错误（CRC 错误）。

11：接收到的错误超过 3 个，最后一个错误是协议错误（无效 PID、假 EOP、位填充错误、SYNC 错误）。

位 28:27 **THREE_ERR_TX[1:0]**：OUT 或 SETUP 事务的三个错误 (Three errors for an OUT or SETUP transaction)

– 主机模式

当 OUT 事务的 USB 总线上连续出现 3 个事务错误时，该位由硬件置 1。同步事务不会生成 THREE_ERR_TX。软件只能将该位清零。

接收到的错误的编码：

00：接收到的错误少于 3 个。

01：接收到的错误超过 3 个，最后一个错误是超时错误。

10：接收到的错误超过 3 个，最后一个错误是数据错误（CRC 错误）。

11：接收到的错误超过 3 个，最后一个错误是协议错误（无效 PID、假 EOP、位填充错误、SYNC 错误）。

位 26 **ERR_RX**：接收到 IN 事务的错误 (Received error for an IN transaction)

– 主机模式

在此通道上 IN 事务期间发生错误（例如设备无应答、CRC 错误、位填充错误、帧格式违规等）时，该位由硬件置 1。软件只能将该位清零。如果 USB_CNTR 寄存器中的 ERRM 位置 1，则会生成通用中断条件和通道相关标志，后者始终有效。

位 25 **ERR_TX**：接收到 OUT/SETUP 事务的错误 (Received error for an OUT/SETUP transaction)

– 主机模式

在此通道上 OUT 或 SETUP 事务期间发生错误（例如设备无应答、CRC 错误、位填充错误、帧格式违规等）时，该位由硬件置 1。软件只能将该位清零。如果 USB_CNTR 寄存器中的 ERRM 位置 1，则会生成通用中断条件和通道相关标志，后者始终有效。

位 24 **LS_EP**：低速端点 (Low speed endpoint) – 仅带 HUB 的主机

– 主机模式

该位由软件置 1，以将 LS 事务发送到相应的端点。

0：全速端点

1：低速端点

位 23 **NAK**：

– 主机模式

当设备以 NAK 响应时，该位由硬件置 1。软件可以使用该位来监视从设备接收到的 NAK 数量。

位 22:16 **DEVADDR[6:0]**：

– 主机模式

在枚举过程中分配给端点的设备地址。

位 15 **VTRX**：接收到 USB 有效事务 (USB valid transaction received)

– 设备模式

当此端点上成功完成 OUT/SETUP 事务时，该位将由硬件置 1；软件只能将该位清零。如果 USB_CNTR 寄存器中的 CTRM 位相应地置 1，则会生成通用中断条件和端点相关中断条件，后者始终有效。已发生事务的类型（OUT 或 SETUP）可通过下述的 SETUP 位来确定。

以 NAK 或 STALL 握手结束的事务不会将该位置 1，因为在出现协议错误或数据翻转不匹配时，实际上并未传输任何数据。

该位为读/写位，但只能写入 0，写入 1 不起作用。

– 主机模式

当 IN 事务在此通道上成功完成时，该位由硬件置 1。软件只能将该位清零。如果 USB_CNTR 寄存器中的 CTRM 位置 1，则会生成通用中断条件和通道相关标志，后者始终有效。

- 以 NAK 结束的事务会将该位置 1，并将 NAK 应答报告给从该寄存器的 STATRX 位域读取 NAK 状态的应用。一个已确认事务保持挂起状态，并由主机在下一帧自动重试，或者主机可以通过将 STATRX 状态复位为 VALID 来立即重试。

- 以 STALL 握手结束的事务会将该位置 1，并将 STALL 应答报告给从该寄存器的 STATRX 位域读取 STALL 状态的应用。因此，主机应用必须禁止该通道并重新枚举。

- 以 ACK 握手结束的事务将该位置 1

如果禁止双缓冲，ACK 应答将由从该寄存器的 STATRX 位域读取 DISABLE 状态的应用来报告。主机应用必须从 USBRAM 读取接收到的数据，并通过将 VALID 写入该寄存器的 STATRX 位域来重新启动通道。

如果使能双缓冲，ACK 应答将由从该寄存器的 STATRX 位域读取 VALID 状态的应用来报告。主机应用必须从 USBRAM 读取接收到的数据并翻转该寄存器的 DTOGTX 位。

- 结束但出错的事务将该位置 1。

可以通过 ERR_RX 位查看错误（仅限主机模式）。

该位为读/写位，但只能写入 0，写入 1 不起作用。

位 14 **DTOGRX**：数据翻转，用于接收传输 (Data Toggle, for reception transfers)

如果端点/通道不是同步，则该位包含要接收的下一个数据包的数据翻转位的预期值（0 = DATA0，1 = DATA1）。当发送 ACK 握手时，硬件会将该位翻转，接着将接收到具有匹配数据 PID 值的数据包；如果端点被定义为控制端点，则硬件会在接收到来自主机的 SETUP PID（设备模式下）时将该位清零，而在设备确认 SETUP 事务（主机模式下）时将该位置 1。

如果端点/通道正在使用双缓冲功能，则该位还用于支持数据包缓冲区交换（请参见*第 40.5.3 节：设备模式下双缓冲端点和使用*）。

如果端点/通道是同步端点/通道，则该位只用于支持数据发送的数据包缓冲区交换，因为此类通道/端点不使用数据翻转，仅发送 DATA0 数据包（请参见*第 40.5.5 节：设备模式下的同步传输*）。硬件在数据包接收结束后立即翻转该位，因为同步传输不使用握手。

该位还可通过软件同步以初始化其值（当端点不是控制端点时是强制的）或强制使用特定的数据翻转/数据包缓冲区。当应用软件写入 0 时，DTOGRX 的值保持不变，而写入 1 时，该位的值将进行翻转。该位为读/写位，但只能通过写入 1 来翻转。

位 13:12 **STATRX[1:0]**：状态位，表示接收传输 (Status bits, for reception transfers)

– 设备模式

这些位包含有关端点状态的信息，如*第 1544 页的表 338：接收状态编码*所列。这些位可通过软件翻转以初始化其值。当应用软件写入 0 时，该位的值保持不变，而写入 1 时，该位的值将进行翻转。当寻址到该端点的 OUT 或 SETUP（仅控制）事务已正确传输后 (VTRX = 1)，硬件会将 STATRX 位设置为 NAK，因此软件有时间在应答新事务之前准备接收的数据。

双缓冲批量端点实现了一种特殊的事务流控制，可基于缓冲区可用性条件来控制状态（请参见*第 40.5.3 节：设备模式下双缓冲端点和使用*）。

如果端点定义为同步端点，则其状态只能是“VALID”或“DISABLED”，在这种情况下，硬件无法在事务成功完成后更改端点的状态。如果软件为同步端点将 STATRX 位设置为“STALL”或“NAK”，则 USB 外设的行为是不明确的。这些位为读/写位，但只能通过写入 1 来翻转。

– 主机模式

这些位是主机应用控制，用于启动、重试或中止由通道驱动的主机事务。

这些位还包含有关设备对最后一个 IN 通道事务的应答的信息，并根据以下 STATRX 状态表报告通道的当前状态：

- DISABLE

如果在单缓冲区通道上收到 ACK 应答，则会报告 DISABLE 值。当处于 DISABLE 状态时，通道未使用或未激活，等待应用通过写入 VALID 来重新启动通道。应用软件可以将 VALID 通道复位为 DISABLE 以中止事务。在这种情况下，该事务将立即从主机执行列表中删除。如果中止的事务已在执行中，则会定期在 USB 上终止，但不会生成相关的 VTRX 中断。

- VALID

仅当处于 VALID 状态时，主机通道才会主动尝试向设备提交 USB 事务。VALID 状态可以由软件设置，也可以在新帧开始时在 NAKED 通道上由硬件自动设置。当设置为 VALID 时，主机通道进入主机执行队列，并等待主机帧调度程序的许可，以提交其配置的事务。

如果在双缓冲通道上收到 ACK 应答，也会报告 VALID 值。在这种情况下，通道在备用缓冲区上保持有效状态，同时应用软件需要读取当前缓冲区并翻转 DTOGTX。如果软件读取较晚，并且备用缓冲区未准备就绪，主机通道会以对应用透明的方式自动挂起。延迟恢复并且 DTOGTX 翻转后，挂起的双缓冲通道就会立即重新激活。

- NAK

如果收到 NAK 应答，则报告 NAK 值。当处于 NAK 状态时，通道被挂起并且不会尝试发送。NAK 状态在下一帧开始时由硬件置为 VALID，或者软件可以通过将其写入 VALID 来将其更改为立即重试发送，或者可以通过写入 DISABLE 将其禁止并中止事务

- STALL

如果收到 STALL 应答，则报告 STALL 值。当处于 STALL 状态时，通道表现为禁止状态下的行为。应用软件不得重试发送，而是复位 USB 并重新枚举。

位 11 **SETUP**：建立事务已完成 (Setup transaction completed)

– 设备模式

该位为只读位，当最后完成的事务为 SETUP 时，它由硬件置 1。该位仅针对控制端点更改其值。它必须在接收事务成功完成（VTRX 事件）时进行检查，以确定所发生事务的类型。为防止中断服务程序因接下来传入的令牌而更改 SETUP 位，该位将在 VTRX 位为 1 时保持冻结；其状态将在 VTRX 为 0 时更改。该位为只读。

– 主机模式

该位由软件置 1，以在控制端点上发送 SETUP 事务。该位仅针对控制端点更改其值。当 SETUP 事务被确认并生成 VTTX 中断时，该位由硬件清零。

位 10:9 **UTYPE[1:0]**：USB 事务类型 (USB type of transaction)

这些位配置该端点/通道的行为，如*表 339：端点/通道类型编码*所述。端点 0/通道 0 必须始终为控制端点/通道，每个 USB 功能必须至少有一个地址为 0 的控制端点/通道，但必要时也可以有其他控制通道/端点。只有控制通道/端点会处理 SETUP 事务，其他类型的端点会忽略 SETUP 事务。SETUP 事务不能通过 NAK 或 STALL 应答。SETUP 事务不能通过 NAK 或 STALL 应答。如果控制端点/通道定义为 NAK，则 USB 外设将不会应答，而会在接收到 SETUP 事务时在接收方向上模拟接收错误。如果控制端点/通道在接收方向上定义为 STALL，则 SETUP 数据包仍会被接受，从而传输数据并发出 CTR 中断。OUT 事务的接收以常规方式处理，即使端点/通道为控制端点/通道也如此。

批量端点和中断端点的行为十分相似，唯一的区别在于使用 EPKIND 配置位可实现的特殊功能。

*第 40.5.5 节：设备模式下的同步传输*中介绍了同步端点/通道的用法。

位 8 **EPKIND**：端点/通道类型 (endpoint/channel kind)

该位的含义取决于 UTYPE 位所配置的端点/通道类型。*表 340* 汇总了不同的含义。

DBL_BUF：该位由软件置 1，用于为此批量端点使能双缓冲功能。*第 40.5.3 节：设备模式下双缓冲端点和使用*中介绍了双缓冲批量端点的用法。

STATUS_OUT：该位由软件置 1，用于指示将出现状态输出事务：在这种情况下，所有包含非零字节数据的 OUT 事务均通过“STALL”而非“ACK”来应答。该位可用于提高应用软件针对控制传输期间的协议错误的鲁棒性（仅限控制端点）。当 STATUS_OUT 复位时，OUT 事务可以有任何数量的字节（根据需要）。

位 7 **VTTX**：已发送的有效 USB 事务 (Valid USB transaction transmitted)

– 设备模式

当此端点上成功完成 IN 事务时，该位将由硬件置 1；软件只能将该位清零。如果 USB_CNTR 寄存器中的 CTRM 位相应地置 1，则会生成通用中断条件和端点相关中断条件，后者始终有效。

以 NAK 或 STALL 握手结束的事务不会将该位置 1，因为在出现协议错误或数据翻转不匹配时，实际上并未传输任何数据。

该位为读/写位，但只能写入“0”。

– 主机模式

与 VTRX 行为相同，但适用于 USB OUT 和 SETUP 事务。

位 6 **DTOGTX**：数据翻转，用于发送传输 (Data Toggle, for transmission transfers)

如果端点/通道不是同步端点/通道，则该位包含要发送的下一个数据包的数据翻转位的预期值（0 = DATA0，1 = DATA1）。当从 USB 主机接收到 ACK 握手时，硬件会将该位翻转，接着将发送一个数据包。如果端点/通道定义为控制端点/通道，则硬件会在接收到寻址到该端点的 SETUP PID时将该位置 1（设备模式下），或者当设备确认 SETUP 事务时将该位置 1（主机模式下）。

如果端点/通道正在使用双缓冲功能，则该位还用于支持数据包缓冲区交换（请参见*第 40.5.3 节：设备模式下双缓冲端点和使用*）。

如果端点/通道是同步端点/通道，则该位用于支持数据包缓冲区交换，因为此类端点不使用数据同步，仅传输 DATA0 数据包（请参见*第 40.5.5 节：设备模式下的同步传输*）。硬件在数据包接收结束后立即翻转该位，因为同步传输不使用握手。

该位还可通过软件翻转以初始化其值（当端点/通道不是控制端点/通道时十分必要）或强制使用特定的数据翻转/数据包缓冲区。当应用软件写入 0 时，DTOGTX 的值保持不变，而写入 1 时，该位的值将进行翻转。该位为读/写位，但只能通过写入 1 来翻转。

位 5:4 **STATTX[1:0]**：状态位，表示发送传输 (Status bits, for transmission transfers)

– 设备模式

这些位包含有关端点状态的信息，如*表 341* 所列。这些位可通过软件翻转以初始化其值。当应用软件写入 0 时，该位的值保持不变，而写入 1 时，该位的值将进行翻转。当寻址到该端点/通道的 IN 或 SETUP（仅控制）事务已正确传输后 (VTTX = 1)，硬件会将 STATTX 位设置为 NAK。然后，它将等待软件准备下一组要发送的数据。

双缓冲批量端点实现了一种特殊的事务流控制，可基于缓冲区可用性来条件控制状态（请参见*第 40.5.3 节：设备模式下双缓冲端点和使用*）。

如果端点定义为同步端点，则其状态只能是“VALID”或“DISABLED”。因此，硬件无法在事务成功完成后更改端点/通道的状态。如果软件为同步通道/端点将 STATTX 位设置为“STALL”或“NAK”，则 USB 外设的行为是不确定的。这些位为读/写位，但只能通过写入 1 来翻转。

– 主机模式

STATTX 位包含有关通道状态的信息。有关完整说明（“主机模式”说明），请参见*表 341*。在设备模式下，这些位包含后续事务发出的状态，而在主机模式下，它们捕获最后从设备接收到的状态。如果接收到 NAK，则 STATTX 包含指示 NAK 的值。

位 3:0 **EA[3:0]**：端点/通道地址 (endpoint/channel address)

– 设备模式

软件必须在该位域中写入 4 位地址，用于确定定向到此端点的事务。必须在使能相应端点前写入一个值。

– 主机模式

软件必须在该位域中写入 4 位地址，用于确定主机事务寻址的通道。

表 338. 接收状态编码

STATRX[1:0]	意义
00	**DISABLED**：忽略寻址到该端点/通道的所有接收请求。
01	**STALL**： 设备模式：端点停止运行，所有接收请求导致 STALL 握手。 主机模式：表明设备已使通道停止运行。
10	**NAK**： 设备模式：不应答端点，所有接收请求导致 NAK 握手。 主机模式：表明设备未应答接收请求。
11	**VALID**：将该端点/通道使能用于接收。

表 339. 端点/通道类型编码

UTYPE[1:0]	意义
00	BULK
01	CONTROL
10	ISO
11	INTERRUPT

表 340. 端点/通道种类的意义

UTYPE[1:0]		EPKIND 的意义
00	BULK	DBL_BUF
01	CONTROL	STATUS_OUT
10	ISO	SBUF_ISO：该位由软件置 1，用于为同步端点使能单缓冲功能。
11	INTERRUPT	未使用

表 341. 发送状态编码

STATTX[1:0]	意义
00	**DISABLED**：忽略寻址到该端点/通道的所有发送请求。
01	**STALL**： 设备模式：端点停止运行，所有发送请求导致 STALL 握手。 主机模式：表明设备已使通道停止运行。
10	**NAK**： 设备模式：不应答端点，所有发送请求导致 NAK 握手。 主机模式：表明设备未应答发送请求。
11	**VALID**：将该端点/通道使能用于发送。

40.6.2 缓冲区描述符表

注意： *缓冲区描述符表位于数据包缓冲区存储器内的独立“USB SRAM”地址空间。*

尽管缓冲区描述符表位于数据包缓冲区存储器（“USB SRAM”区）内，但其条目可视为额外的寄存器，用于配置在 USB 宏单元和设备之间交换数据所使用的数据包缓冲区的位置和大小。

第一个数据包存储单元位于 USB SRAM 基址。下文介绍了与 USB_CHEPnR 寄存器相关的缓冲区描述符表条目。必须使用字（32 位）访问对存储器进行寻址。

有关数据包缓冲区和缓冲区描述符表用法的详细说明，请参见*第 1518 页的数据包缓冲区的结构和用法*。

通道/端点发送缓冲区描述符 n (USB_CHEP_TXRXBD_n)

Channel/endpoint transmit buffer descriptor n

偏移地址：n*8

如果相应的 CHEPnR 寄存器未对在接收模式下工作的双缓冲使用进行编程，则该寄存器描述适用（否则请参见后文的寄存器描述）

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

位 31:26 保留，必须保持复位值。

位 25:16 **COUNT_TX[9:0]**：发送字节数 (Transmission byte count)

这些位包含与 USB_CHEPnR 寄存器相关的端点/通道将在寻址到它的下一个 IN 令牌中发送的字节数。

位 15:0 **ADDR_TX[15:0]**：发送缓冲区地址 (Transmission buffer address)

这些位指向将包含如下内容的数据包缓冲区的起始地址：与 USB_CHEPnR 寄存器相关的端点/通道将在寻址到它的下一个 IN 令牌中发送的数据。位 1 和 位 0 必须始终写为“00”，因为数据包存储器为字宽，所有数据包缓冲区必须按字对齐。

通道/端点接收缓冲区描述符 n [复用] (USB_CHEP_TXRXBD_n)

Channel/endpoint receive buffer descriptor n

偏移地址：n*8

如果相应的 CHEPnR 寄存器对双缓冲使用进行了编程并激活接收缓冲区，则该寄存器描述适用（否则请参见上一个寄存器描述）

此表存储单元用于存储数据包接收期间需要的两个不同的值。最高有效位包含已分配缓冲区大小的定义，以便进行缓冲区溢出检测；此存储单元的最低有效位由 USB 外设在接收结束时回写，以给出已接收字节的实际数量。由于可用位数受到限制，缓冲区大小使用已分配存储块的数量来表示，其中可通过选择块大小在精细粒度/小缓冲区与粗略粒度/大缓冲区之间进行折衷。已分配缓冲区的大小是端点/通道描述符的一部分，通常在枚举过程中根据其 maxPacketSize 参数值定义（请参见“通用串行总线规范”）。

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

位 31 **BLSIZE**：块大小 (Block size)

该位选择用于定义已分配缓冲区的存储器块的大小。

– 如果 BLSIZE = 0，则存储器块为 2 字节，这是半字宽存储器中允许的最小块。使用此块大小时，所分配缓冲区的大小在 2 字节到 62 字节范围内。

– 如果 BLSIZE = 1，则存储器块为 32 字节，可达到 USB 规范定义的最大数据包长度。使用此块大小时，理论上所分配缓冲区的大小在 32 字节到 1024 字节范围内，这是 USB 标准规范允许的最大数据包长度。不过，可用的缓冲区大小受可用缓冲区存储器的限制。

位 30:26 **NUM_BLOCK[4:0]**：区块数 (Number of blocks)

这些位定义分配到此数据包缓冲区的存储器块的数量。已分配存储器的实际数量取决于*表 342* 中给出的 BLSIZE 值。

位 25:16 **COUNT_RX[9:0]**：接收字节数 (Reception byte count)

这些位包含与 USB_CHEPnR 寄存器相关的端点/通道在寻址到它的最后一个 OUT/SETUP 事务期间接收的字节数。

位 15:0 **ADDR_RX[15:0]**：接收缓冲区地址 (Reception buffer address)

这些位指向将包含如下内容的数据包缓冲区的起始地址：与 USB_CHEPnR 寄存器相关的端点/通道在寻址到它的下一个 OUT/SETUP 令牌中接收的数据。位 1 和 位 0 必须始终写为“00”，因为数据包存储器为字宽，所有数据包缓冲区必须按字对齐。

表 342. 已分配缓冲区存储器的定义

NUM_BLOCK[4:0] 的值	已分配的存储器 BLSIZE=0 时	已分配的存储器 BLSIZE=1 时
0 (00000)	不允许	32 字节
1 (00001)	2 字节	64 字节
2 (00010)	4 字节	96 字节
3 (00011)	6 字节	128 字节
...	...	...
14 (01110)	28 字节	480 字节
15 (01111)	30 字节	
16 (10000)	32 字节	
...	...	...
29 (11101)	58 字节	...
30 (11110)	60 字节	992 字节
31 (11111)	62 字节	1023 字节

通道/端点接收缓冲区描述符 n (USB_CHEP_RXTXBD_n)

Channel/endpoint receive buffer descriptor n

偏移地址：n*8 + 4

如果相应的 CHEPnR 寄存器未对在发送模式下工作的双缓冲使用进行编程，则该寄存器描述适用（否则请参见后文的寄存器描述）

此表存储单元用于存储数据包接收期间需要的两个不同的值。最高有效位包含已分配缓冲区大小的定义，以便进行缓冲区溢出检测；此存储单元的最低有效位由 USB 外设在接收结束时回写，以给出已接收字节的实际数量。由于可用位数受到限制，缓冲区大小使用已分配存储块的数量来表示，其中可通过选择块大小在精细粒度/小缓冲区与粗略粒度/大缓冲区之间进行折衷。已分配缓冲区的大小是端点/通道描述符的一部分，通常在枚举过程中根据其 maxPacketSize 参数值定义（请参见“通用串行总线规范”）。

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

位 31 **BLSIZE**：块大小 (Block size)

该位选择用于定义已分配缓冲区的存储器块的大小。

- 如果 BLSIZE = 0，则存储器块为 2 字节，这是半字宽存储器中允许的最小块。使用此块大小时，所分配缓冲区的大小在 2 字节到 62 字节范围内。
- 如果 BLSIZE = 1，则存储器块为 32 字节，可达到 USB 规范定义的最大数据包长度。使用此块大小时，理论上所分配缓冲区的大小在 32 字节到 1024 字节范围内，这是 USB 标准规范允许的最大数据包长度。不过，可用的缓冲区大小受可用缓冲区存储器的限制。

位 30:26 **NUM_BLOCK[4:0]**：区块数

这些位定义分配到此数据包缓冲区的存储器块的数量。已分配存储器的实际数量取决于*表 342* 中给出的 BLSIZE 值。

位 25:16 **COUNT_RX[9:0]**：接收字节数 (Reception byte count)

这些位包含与 USB_CHEPnR 寄存器相关的端点/通道在寻址到它的最后一个 OUT/SETUP 事务期间接收的字节数。

位 15:0 **ADDR_RX[15:0]**：接收缓冲区地址 (Reception buffer address)

这些位指向将包含如下内容的数据包缓冲区的起始地址：与 USB_CHEPnR 寄存器相关的端点/通道在寻址到它的下一个 OUT/SETUP 令牌中接收的数据。位 1 和 位 0 必须始终写为“00”，因为数据包存储器为字宽，所有数据包缓冲区必须按字对齐。

通道/端点发送缓冲区描述符 n [复用] (USB_CHEP_RXTXBD_n)

Channel/endpoint transmit buffer descriptor n

偏移地址：n*8 + 4

如果相应的 CHEPnR 寄存器对双缓冲使用进行了编程并激活发送缓冲区，则该寄存器描述适用（否则请参见上一个寄存器描述）

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

位 31:26 保留，必须保持复位值。

位 25:16 **COUNT_TX[9:0]**：发送字节数 (Transmission byte count)

这些位包含与 USB_CHEPnR 寄存器相关的端点/通道将在寻址到它的下一个 IN 令牌中发送的字节数。

位 15:0 **ADDR_TX[15:0]**：发送缓冲区地址 (Transmission buffer address)

这些位指向将包含如下内容的数据包缓冲区的起始地址：与 USB_CHEPnR 寄存器相关的端点/通道将在寻址到它的下一个 IN 令牌中发送的数据。位 1 和 位 0 必须始终写为“00”，因为数据包存储器为字宽，所有数据包缓冲区必须按字对齐。

40.6.3 USB 寄存器映射

下表提供了 USB 寄存器映射和复位值。

表 343. USB 寄存器映射和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**USB_CHEP0R**	Res.	THREE_ERR_RX [1:0]		THREE_ERR_TX [1:0]		ERR_RX	ERR_TX	LS_EP	NAK	DEVADDR[6:0]							VTRX	DTOGRX	STATRX [1:0]		SETUP	UTYPE [1:0]		EPKIND	VTTX	DTOGTX	STATTX [1:0]		EA[3:0]			
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	**USB_CHEP1R**	Res.	THREE_ERR_RX [1:0]		THREE_ERR_TX [1:0]		ERR_RX	ERR_TX	LS_EP	NAK	DEVADDR[6:0]							VTRX	DTOGRX	STATRX [1:0]		SETUP	UTYPE [1:0]		EPKIND	VTTX	DTOGTX	STATTX [1:0]		EA[3:0]			
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	**USB_CHEP2R**	Res.	THREE_ERR_RX [1:0]		THREE_ERR_TX [1:0]		ERR_RX	ERR_TX	LS_EP	NAK	DEVADDR[6:0]							VTRX	DTOGRX	STATRX [1:0]		SETUP	UTYPE [1:0]		EPKIND	VTTX	DTOGTX	STATTX [1:0]		EA[3:0]			
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	**USB_CHEP3R**	Res.	THREE_ERR_RX [1:0]		THREE_ERR_TX [1:0]		ERR_RX	ERR_TX	LS_EP	NAK	DEVADDR[6:0]							VTRX	DTOGRX	STATRX [1:0]		SETUP	UTYPE [1:0]		EPKIND	VTTX	DTOGTX	STATTX [1:0]		EA[3:0]			
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	**USB_CHEP4R**	Res.	THREE_ERR_RX [1:0]		THREE_ERR_TX [1:0]		ERR_RX	ERR_TX	LS_EP	NAK	DEVADDR[6:0]							VTRX	DTOGRX	STATRX [1:0]		SETUP	UTYPE [1:0]		EPKIND	VTTX	DTOGTX	STATTX [1:0]		EA[3:0]			
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	**USB_CHEP5R**	Res.	THREE_ERR_RX [1:0]		THREE_ERR_TX [1:0]		ERR_RX	ERR_TX	LS_EP	NAK	DEVADDR[6:0]							VTRX	DTOGRX	STATRX [1:0]		SETUP	UTYPE [1:0]		EPKIND	VTTX	DTOGTX	STATTX [1:0]		EA[3:0]			
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	**USB_CHEP6R**	Res.	THREE_ERR_RX [1:0]		THREE_ERR_TX [1:0]		ERR_RX	ERR_TX	LS_EP	NAK	DEVADDR[6:0]							VTRX	DTOGRX	STATRX [1:0]		SETUP	UTYPE [1:0]		EPKIND	VTTX	DTOGTX	STATTX [1:0]		EA[3:0]			
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	**USB_CHEP7R**	Res.	THREE_ERR_RX [1:0]		THREE_ERR_TX [1:0]		ERR_RX	ERR_TX	LS_EP	NAK	DEVADDR[6:0]							VTRX	DTOGRX	STATRX [1:0]		SETUP	UTYPE [1:0]		EPKIND	VTTX	DTOGTX	STATTX [1:0]		EA[3:0]			
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 343. USB 寄存器映射和复位值（续）

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x20-0x3F	Reserved																																
0x40	**USB_CNTR**	HOST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DDISCM	THR512M	CTRM	PMAOVRM	ERRM	WKUPM	SUSPM	RST_DCONM	SOFM	ESOFM	L1REQM	Res.	L1RES	L2RES	SUSPEN	SUSPRDY	PDWN	USBRST
	Reset value	0														0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	1	1
0x44	**USB_ISTR**	Res.	LS_DCON	DCON_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DDISC	THR512	CTR	PMAOVR	ERR	WKUP	SUSP	RST_DCON	SOF	ESOF	L1REQ	Res.	Res.	DIR	IDN[3:0]			
	Reset value		0	0												0	0	0	0	0	0	0	0	0	0	0			0	0	0	0	0
0x48	**USB_FNR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RXDP	RXDM	LCK	LSOF [1:0]		FN[10:0]										
	Reset value																	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x
0x4C	**USB_DADDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EF	ADD[6:0]						
	Reset value																									0	0	0	0	0	0	0	0
0x54	**USB_LPMCSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BESL[3:0]				REMWAKE	Res.	LPMACK	LPMEN
	Reset value																									0	0	0	0	0		0	0
0x58	**USB_BCDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DPPU_DPD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PS2DET	SDET	PDET	DCDET	SDEN	PDEN	DCDEN	BCDEN
	Reset value																	0								0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 68 页的第 2.2 节*。

41 调试支持 (DBG)

41.1 简介

为支持软件开发和系统集成，提供了以下一整套调试功能：

- CPU 内核断点调试
- 代码执行跟踪
- 软件指令
- 交叉触发

调试功能可以通过 JTAG/串行线调试访问端口，使用行业标准调试工具进行控制。跟踪端口允许捕获数据以进行记录和分析。

调试功能基于 Arm® CoreSight™ 组件。

- SWJ-DP：JTAG/串行线调试端口
- AHB-AP：AHB 访问端口
- ROM 表
- 系统控制空间 (SCS)
- 断点单元 (BPU)
- 数据观察点和跟踪单元 (DWT)
- 指令跟踪宏单元 (ITM)
- 嵌入式跟踪宏单元™ (ETM)
- 交叉触发接口 (CTI)
- 跟踪端口接口单元 (TPU)

调试器可通过 AHB-AP 访问调试功能。

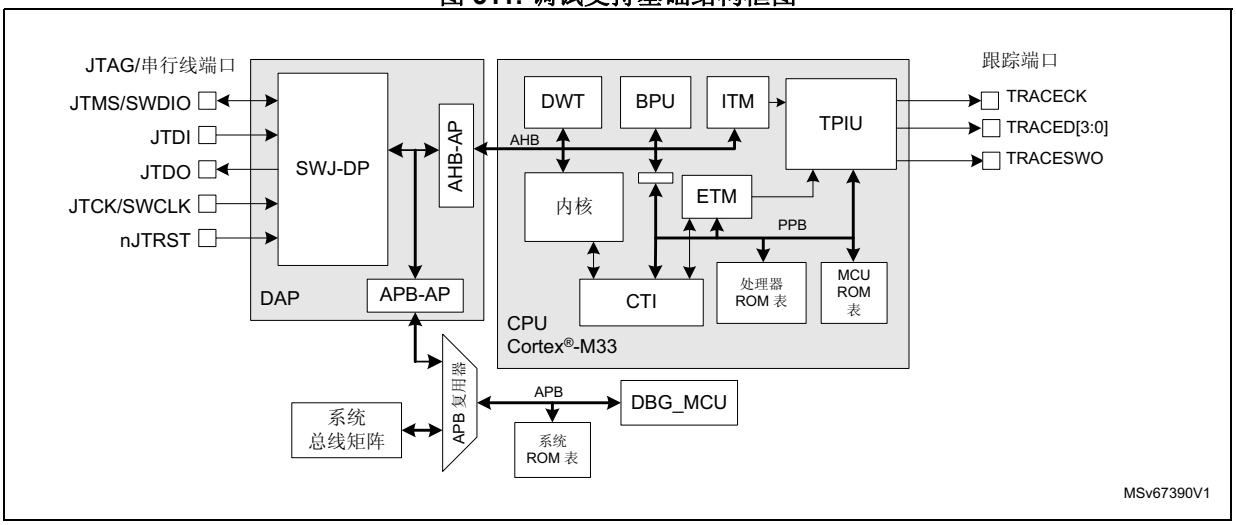
更多相关信息，请参见*第 41.13 节*中引用的 Arm® 文档。

注意： *本产品仅支持非安全存储器访问。*

41.2 DBG 功能描述

41.2.1 DBG 框图

图 511. 调试支持基础结构框图



41.2.2 DBG 引脚和内部信号

表 345. JTAG/串行线调试端口引脚

引脚名称	JTAG 调试端口		SW 调试端口		引脚分配
	类型	说明	类型	说明	
JTMS/SWDIO	I	JTAG 测试模式选择	IO	串行线数据输入/输出	PA13
JTCK/SWCLK	I	JTAG 测试时钟	I	串行线时钟	PA14
JTDI	I	JTAG 测试数据输入	-	-	PA15
JTDO	O	JTAG 测试数据输出	-	-	PB3
nJTRST	I	JTAG 测试复位	-	-	PB4

表 346. 跟踪端口引脚

引脚名称	类型	说明	引脚分配
TRACED0	O	跟踪同步数据输出 0	请参见数据手册
TRACED1		跟踪同步数据输出 1	
TRACED2		跟踪同步数据输出 2	
TRACED3		跟踪同步数据输出 3	
TRACECK		TRACE 时钟	

表 347. 单线跟踪端口引脚

引脚名称	类型	说明	引脚分配
TRACESWO	O	单线跟踪同步数据输出	PB3(1)

1. TRACESWO 与 JTDO 多路复用。这意味着单线跟踪仅在使用串行线调试接口时可用，而在使用 JTAG 时不可用。

41.2.3 DBG 复位和时钟

调试端口 (SWJ-DP) 通过上电复位进行复位，以及从待机模式唤醒时复位。

调试器通过调试接口引脚 JTCK/SWCLK 为调试端口提供时钟。该时钟用于在串行线和 JTAG 模式下寄存串行输入数据，同时操作状态机和调试端口的内部逻辑。因此，为确保调试端口返回到空闲状态，在访问结束之后，该时钟还必须持续翻转几个周期。

SWJ-DP 包含一个异步接口，用来连接到 DCLK 域，该域覆盖 SWJ-DP 的余部和访问端口。

DCLK 是通过对系统时钟进行门控而得到的。

DCLK 域由调试器使用 *DP 控制和状态寄存器 (DP_CTRLSTATR)* 中的 CDBGPWRUPREQ 位来使能。必须先使能时钟，然后调试器才能访问器件上的所有调试功能。时钟的可用性由 DP_CTRL/STATR 中的 CDBGPWRUPACK 位指示。DCLK 在上电时禁止，调试器断开时必须禁止 DCLK，以避免浪费能源。

处理器中包含的调试和跟踪组件由处理器时钟提供时钟。

41.2.4 DBG 电源域

调试组件位于内核电源域中。这意味着，在待机低功耗模式下无法连接调试器。为避免器件进入待机模式时失去连接，可以通过将 *DBGMCU 配置寄存器 (DBGMCU_CR)* 中的一个位置位来保持为内核供电。该操作还可以使处理器时钟保持活动状态并阻止复位，从而维持调试会话。

41.2.5 调试和低功耗模式

器件具有节能特性，允许在不需要的时候关闭或停止内核电源域。如果电源关闭或者内核没有时钟信号，则调试器将无法访问所有调试组件。为了避免这种情况，设计了节能模式仿真这一解决方案。如果对该域使能了仿真功能，则该域仍会进入节能模式，但其时钟和电源保持不变。换句话说，该域的行为就像处于节能模式，但调试器仍然可以对其进行访问。

仿真模式在微控制器调试 (DBGMCU) 单元中被编程。有关详细信息，请参见*第 41.12 节：微控制器调试单元 (DBGMCU)*。

41.2.6 安全性

跟踪和调试组件允许在产品开发期间对处理器和系统进行深度访问。为了保护用户代码，并确保无法使用调试功能来改变或损害成品的正常操作，可以禁止这些功能或限制其范围。

当处理器从系统 Flash 启动时，将禁止调试器访问。

系统使用以下验证信号来确定使能或禁止哪些调试功能：

- **dbgen**：全局使能，用于所有调试功能
  0：所有调试功能均禁止。
  1：使能调试功能。
- **Niden**：使能跟踪和性能监控（非侵入式调试）。
  0：禁止跟踪生成。
  1：使能非安全状态下的跟踪生成。

有关各个组件根据验证信号状态的行为的详细信息，请参见相关组件的章节或相关 Arm® 技术文档。

信号状态根据调试状态设置，如*表 348* 中所示。

表 348. 验证信号状态

调试状态	验证信号状态	说明
OPEN	dbgen = 1 spiden = 1 niden = 1 spniden = 1	无论处理器的状态如何，都会使能调试和跟踪。调试器可访问所有存储器和资源。
CLOSED	dbgen = 0 spiden = 0 niden = 0 spniden = 0	禁止调试和跟踪。

验证信号的状态可从 Cortex®-M33 的系统控制空间 (SCS) 中的 DAUTHSTATUS 寄存器读取。

调试状态取决于产品生命周期状态（请参见*第 3.5 节：产品生命周期*）。

表 349. 生命周期状态和调试状态

产品生命周期状态	调试状态
OPEN	OPEN
CLOSED	CLOSED
LOCKED	CLOSED

41.2.7 调试身份验证

图 512. 产品生命周期状态和调试身份验证

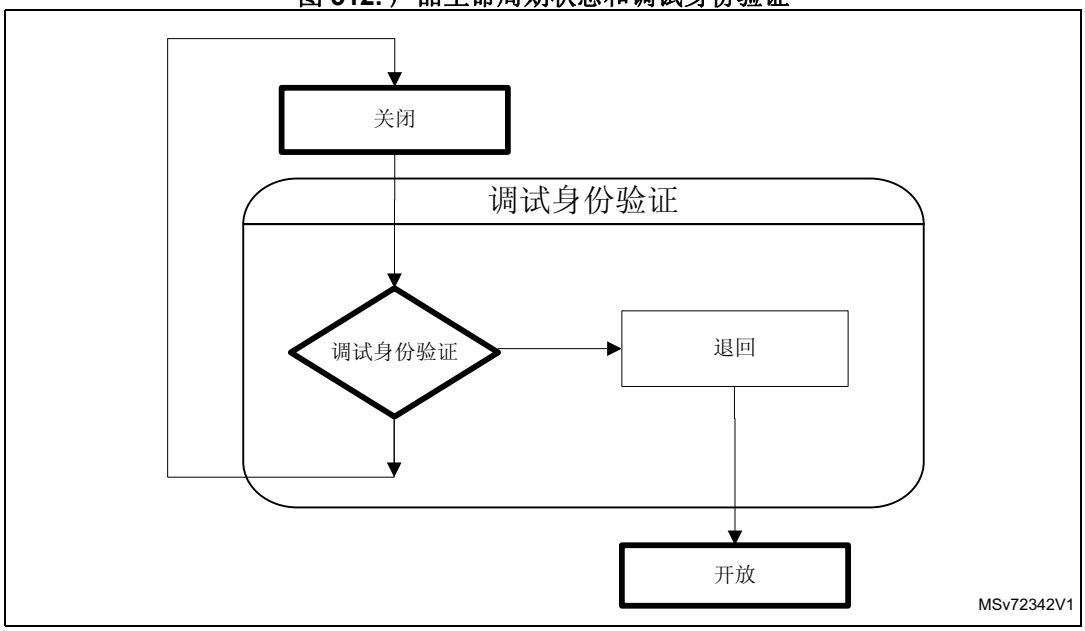


图 512 显示产品生命周期和调试身份验证状态。如果设备的 PRODUCT_STATE 为“CLOSED”，则调试状态为 CLOSED。调试身份验证过程允许受信任的调试器通过执行完全回退来开放访问。

完全回退相当于释放所有代码和资产。允许完全回退管理的中间状态称为回退。

调试身份验证控制原理

用户可以借助调试器访问系统的大部分内容，鉴于此，调试身份验证是系统最关键的安全功能之一。

为了控制调试的重新开启，器件实施了调试身份验证协议。

该协议基于密码来触发完全回退操作。

该协议通过位于 DBGMCU 中的邮箱接口实现主机与设备之间的双向通信。

主机可以通过 JTAG/SWD 接口写入邮箱，并预期通过同一邮箱获得来自设备的响应和消息。

当主机发出“开启请求”消息时，调试身份验证协议即会在设备上电复位时启动。

该协议基于：

- 初始消息：由主机发布，结合复位以在设备上启动调试身份验证过程。
- 密码消息：主机发送一条包含用户密码的消息。

该设计方案通过系统 Flash 中嵌入的代码来保证。如果主机已在邮箱中发布初始消息，则复位后会自动调用此代码。

意法半导体设计方案基于 Arm® PSA-ADAC 解决方案进行调试身份验证。

调试身份验证可以使用专有或开源协议来实现。

由于此功能对于安全性至关重要，因此 STM32H5 器件在系统 Flash 中提供了调试身份验证功能。意法半导体提供与一些调试环境 (IDE) 集成的主机工具。

调试身份验证预置

STM32H5 平台本身支持调试身份验证。这意味着，ST 调试身份验证 (ST-DA) 使用的数据必须配置为在 OTP Flash 区域中的指定位置（在 Flash 中定义 OTP）。

当 PRODUCT_STATE 为“Provisioning”时，必须完成调试身份验证配置。当 PRODUCT_STATE 为“Open”时，不得执行此操作。

密码的 SHA256 算法必须配置为在 OTP 区域的最开头（请参见*第 7.6.2 节：RSS 用户函数*）。

小心： 在转换为与“Open”或“Provisioning”不同的状态之前，用户必须仔细按照上文说明配置调试身份验证。

41.3 串行线和 JTAG 调试端口 (SWJ-DP)

SWJ-DP 是一个 Coresight™ 组件，提供用来连接调试设备的外部访问端口。

可配置两种类型的接口：

- 一个 5 针标准 JTAG 接口 (JTAG-DP)
- 一个 2 针（时钟 + 数据）串行线调试端口 (SW-DP)

这两种模式共用相同的 IO 引脚，因此两者互斥。

默认情况下，系统复位或上电复位后选择 JTAG-DP 模式。五个 IO 引脚由硬件在调试备用功能模式下配置。在 SWJ-DP 模式下，在 JTDI、JTMS/SWDIO 和 nJTRST 上集成了上拉电阻，在 JTCK/SWCLK 上集成了下拉电阻。

调试器可以通过在 JTMS/SWDIO 上传送以下串行数据序列来选择 SW-DP 模式：

...（50 个或更多）..., 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, ...（50 个或更多）...

对于每一数据位都必须循环选通 JTCK/SWCLK。

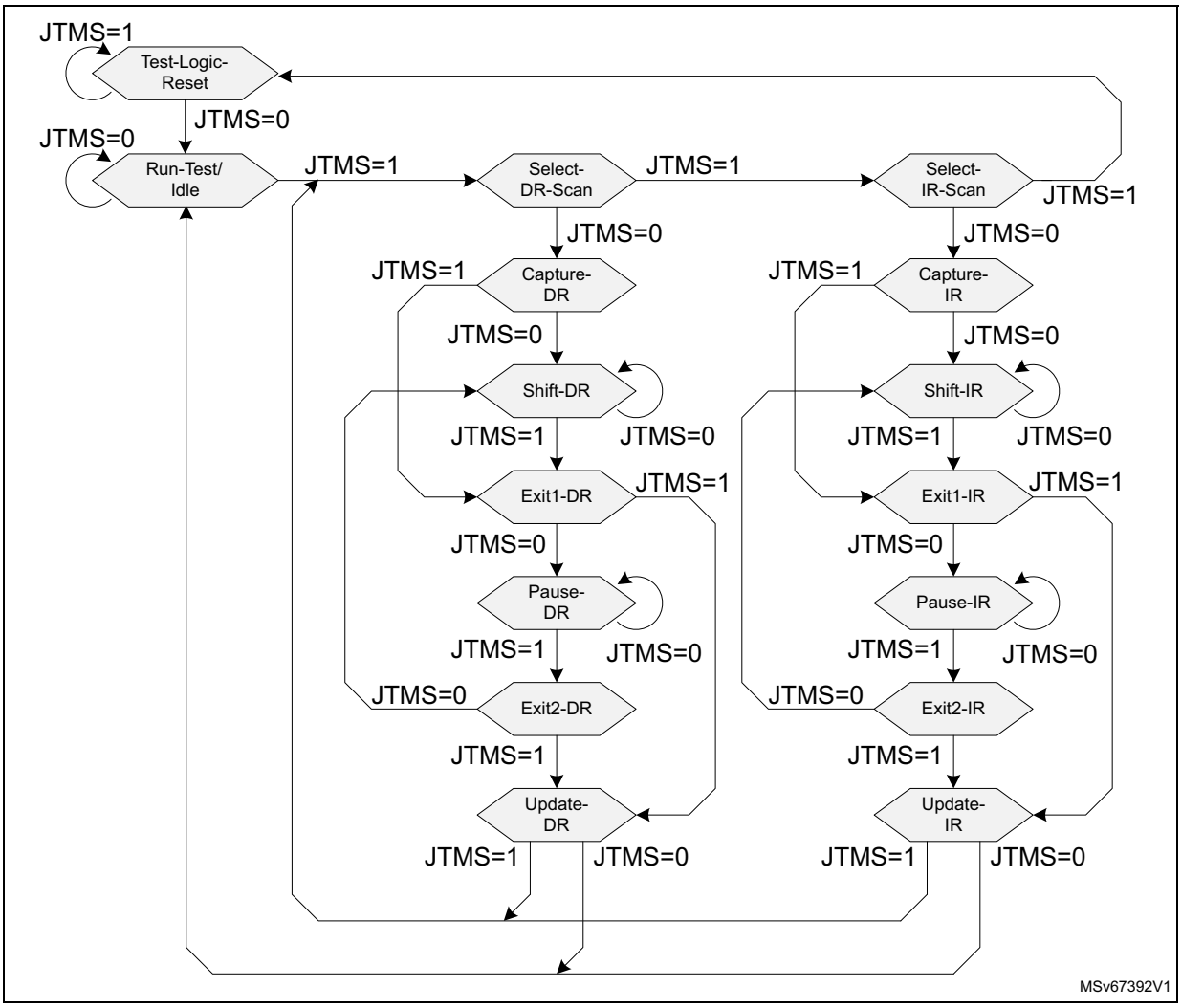
在 SW-DP 模式下，未使用的 JTAG 引脚 JTDI、JTDO 和 nJTRST 可用于其他功能。

注意： *所有 SWJ 端口 IO 都可以通过软件重新配置为其他功能，但不能再进行调试。*

41.3.1 JTAG 调试端口

JTAG-DP 设计了一个基于 IEEE 标准 1149.1-1990 的 TAP 状态机 (TAPSM)，如下图所示。该状态机控制两个扫描通道，一个与指令寄存器 (IR) 相关，一个与多个数据寄存器 (DR) 相关。

图 513. JTAG TAP 状态机



JTAG-DP 的工作原理如下：

1. 当 TAPSM 通过 Capture-IR 状态时，0b0001 被传送到指令寄存器 (IR) 扫描通道。IR 扫描通道连接在 JTDI 和 JTDO 之间。
2. 当 TAPSM 处于 Shift-IR 状态时，IR 扫描通道在 JTCK 的每个上升沿移位一位。这意味，在第一个节拍上：
   – IR 扫描通道的 LSB 是 JTDO 上的输出。
   – IR 扫描通道的位 [n] 被传送到位 [n-1]。
   – JTDI 上的值会传送到 IR 扫描通道的 MSB 中。
3. 当 TAPSM 通过 Update-IR 状态时，扫描进 IR 扫描通道中的值被传送到指令寄存器。
4. 当 TAPSM 通过 Capture-DR 状态时，值从其中一个数据寄存器传送到其中一个 DR 扫描通道，DR 扫描通道连接在 JTDI 和 JTDO 之间。
5. 指令寄存器中保存的值决定了数据寄存器和相关的 DR 扫描通道的选择。
6. 当 TAPSM 处于 Shift-DR 状态时，数据进行移位，与处于 IR 状态时的 IR 移位情形相同。
7. 当 TAPSM 通过 Update-DR 状态时，扫描进 DR 扫描通道中的值被传送到所选择的数据寄存器中。
8. 当 TAPSM 处于 Run-Test/Idle 状态时，不会发生特殊操作。IDCODE 指令加载在 IR 中。

nJTRST 信号在激活后将状态机异步复位为 test-logic-reset 状态。

下表列出与 4 位 IR 指令对应的数据寄存器。

表 350. JTAG-DP 数据寄存器

IR 指令	DR 寄存器	扫描通道长度	说明
0000 到 0111	(BYPASS)	1	未设计：选择 BYPASS
1000	ABORT	35	ABORT 寄存器 – 位 31:1 = 保留 – 位 0 = APABORT：写入 1 以生成 AP 中止。
1001	(BYPASS)	1	保留：选择 BYPASS
1010	DPACC	35	调试端口访问寄存器 初始化调试接口，并允许访问调试接口寄存器。 – 传输 IN 数据时： 位 34:3 = DATA[31:0] = 为写请求传输的 32 位数据 位 2:1 = A[3:2] = 调试端口寄存器的 2 位地址。 位 0 = RnW = 读请求 (1) 或写请求 (0)。 – 传输 OUT 数据时： 位 34:3 = DATA[31:0] = 读请求后读取的 32 位数据 位 2:0 = ACK[2:0] = 3 位确认： – 010 = OK/FAULT – 001 = WAIT – 其他值 = 保留

表 350. JTAG-DP 数据寄存器（续）

IR 指令	DR 寄存器	扫描通道长度	说明
1011	APACC	35	存取接口寄存器 初始化存取接口并允许访问存取接口寄存器。 – 传输 IN 数据时： 位 34:3 = DATA[31:0] = 为写请求移入的 32 位数据 位 2:1 = A[3:2] = 访问端口寄存器的 2 位子地址。 位 0 = RnW = 读请求 (1) 或写请求 (0)。 – 传输 OUT 数据时： 位 34:3 = DATA[31:0] = 读请求后读取的 32 位数据 位 2:0 = ACK[2:0] = 3 位确认： – 010 = OK/FAULT – 001 = WAIT – 其他值：保留
1100	(BYPASS)	1	保留：选择 BYPASS
1101	(BYPASS)	1	保留：选择 BYPASS
1110	IDCODE	32	标识码 0x6BA0 0477：Cortex®-M33 JTAG 调试端口 ID 代码
1111	(BYPASS)	1	旁路 在 JTDI 和 JTDO 之间插入单个 JTCK 周期延迟

有关 DR 寄存器的详细说明，请参见《ARM® 调试接口架构规范》（请参见*第 41.13 节*）。

41.3.2 串行线调试端口

串行线调试协议使用以下引脚：

- SWCLK：从主机到目标的时钟
- SWDIO：双向串行数据

串行数据传输与时钟同步，首先传输 LSB。

传输过程包括三个阶段：

1. 主机发送的数据包请求（8 位）（请参见*表 351*）。
2. 目标发送的确认响应（3 位）（请参见*表 352*）。
3. 主机（写操作时）或目标（读操作时）发送的传输数据（33 位）（请参见*表 353*）。

只有在确认响应为“OK”时才进行数据传输。

在每一个传输阶段之间，如果数据传输方向发生改变，则插入一个单个时钟周期的周转时间。

表 351. 数据包请求

位域	名称	说明
0	Start	必须为 1
1	APnDP	– 0：DP 寄存器访问——请参见*第 41.3.3 节：调试端口寄存器* – 1：AP 寄存器访问——请参见*第 41.4 节：访问端口*

表 351. 数据包请求（续）

位域	名称	说明
2	RnW	– 0：写请求 – 1：读请求
4:3	A(3:2)	DP 或 AP 寄存器的地址位域（请参见*表 355* 或*表 357*）
5	奇偶校验	前面几位的单位奇偶校验
6	停止	0
7	驻留	不受主机驱动，必须由目标读为 1

表 352. ACK 响应

位域	名称	说明
2:0	ACK	– 000：FAULT – 010：WAIT – 100：OK

表 353. 数据传输

位域	名称	说明
31:0	WDATA 或 RDATA	写入或读取数据
32	奇偶校验	32 个数据位的单独位奇偶校验

如果目标响应为 FAULT 或 WAIT ACK，则数据传输过程将取消，除非使能了上溢检测，在这种情况下，数据会被目标忽略（写操作时）或不接受指令控制（读操作时）。

当引线第一次接通时，必须由主机生成引线复位信号，否则会出现协议错误。引线复位信号由 50 或以上个 SWDIO 为高电平的 SWCLK 周期和紧随的两个 SWDIO 为低电平的 SWCLK 周期组成。

有关串行线调试协议的更多详细信息，请参见《ARM® 调试接口架构规范》*[1]*。

注意： *SWJ-DP 适用 SWD 协议版本 2。*

41.3.3 调试端口寄存器

SW-DP 和 JTAG-DP 都可以访问*表 355* 中列出的调试端口 (DP) 寄存器。

调试器可访问 DP 寄存器，如下所示：

1. 如果使用 JTAG，则使用存储区中的寄存器地址对 DPACC 寄存器中的 A(3:2) 位域编程。对 RnW 位编程，选择读操作或写操作。如果选择写操作，则使用写入数据对数据位域进行编程。如果使用 SWD，则 A(3:2) 和 RnW 位域包含在发送到 SW-DP 的数据包请求字中，数据包请求字的 APnDP 位复位（请参见*表 351*）。写入数据在数据阶段发送。
2. 要访问地址 0x4 处的某个分区 DP 寄存器，必须先将寄存器编号写入地址 0x8 处的 DP_SELECTR 寄存器。对地址 0x4 的任何后续读取操作或写入操作都将访问与 DP_SELECTR 寄存器的内容相对应的寄存器。

表 354. 调试端口寄存器

地址	A(3:2) 值	R/W	说明
0x0	00	R	*DP 调试端口标识寄存器 (DP_DPIDR)* 包含调试端口的 IDCODE。
		W	*DP 中止寄存器 (DP_ABORTR)*(1) 中止当前的 AP 事务。该寄存器还用于清除 DP_CTRLSTATR 寄存器中的错误标志。
0x4	01	R/W	如果 DP_SELECTR.DPBANKSEL[3:0] = 0x0，*DP 控制和状态寄存器 (DP_CTRLSTATR)* 控制 DP 并提供状态信息。
			如果 DP_SELECTR.DPBANKSEL[3:0] = 0x1，*DP 数据链路控制寄存器 (DP_DLCR)*(2) 控制 SWD 数据链路的工作模式。
			如果 DP_SELECTR.DPBANKSEL[3:0] = 0x2，*DP 目标标识寄存器 (DP_TARGETIDR)* 提供目标标识信息。
			如果 DP_SELECTR.DPBANKSEL[3:0] = 0x3，*DP 数据链路协议标识寄存器 (DP_DLPIDR)*(2) 提供 SWD 协议版本。
0x8	10	R	*DP 事件状态寄存器 (DP_RESENDR)*(2) 返回由最后一个 AP 读操作或 DP_RDBUFF 读操作返回的值。在读传输损坏的情况下使用。
		W	*DP 访问端口选择寄存器 (DP_SELECTR)* 选择访问端口、访问端口寄存器存储区域和地址为 0x4 的 DP 寄存器。
0xC	11	R	*DP 读缓冲区寄存器 (DP_RDBUFFR)* – 通过 JTAG-DP，该寄存器允许调试器在执行一系列操作后获取最后结果（无需请求新的 JTAG-DP 操作）。 – 通过 SW-DP，该寄存器包含以前的 AP 读访问的结果，从而避免了重新进行 AP 访问。

1. 使用 ABORT 指令实现从 JTAG-DP 访问 AP ABORT 寄存器。
2. 只能通过 SW-DP 访问。寄存器通过 JTAG-DP “保留”。

DP 调试端口标识寄存器 (DP_DPIDR)

DP debug port identification register

偏移地址：0x0

复位值：0x6BA0 2477

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REVISION[3:0]				PARTNO[7:0]								Res.	Res.	Res.	MIN
r	r	r	r	r	r	r	r	r	r	r	r				r

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

位 31:28 **REVISION[3:0]**：版本代码 (revision code)
0x6：第 6 版

位 27:20 **PARTNO[7:0]**：调试端口的产品编号 (part number for the debug port)
0xBA

位 19:17 保留，必须保持复位值。

位 16 **MIN**：最少调试端口 (MINDP) 设计 (minimal debug port (MINDP) implementation)
0x0：MIDP 未设计

位 15:12 **VERSION[3:0]**：调试端口架构版本 (debug port architecture version)
0x2：DPv2

位 11:1 **DESIGNER[10:0]**：JEDEC 设计人员标识代码 (JEDEC designer identity code)
0x23B：Arm® JEDEC 代码

位 0 保留，必须保持复位值。

DP 中止寄存器 (DP_ABORTR)

DP abort register

偏移地址：0x0

复位值：0x0000 0000

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

位 31:5 保留，必须保持复位值。

位 4 **ORUNERRCLR**：上溢错误清零 (overrun error clear)
0：无影响
1：DP_CTRL/STATR 寄存器中的 STICKYORUN 位清零

位 3 **WDERRCLR**：写入数据错误清零 (write data error clear)
0：无影响
1：DP_CTRL/STATR 寄存器中的 WDATAERR 位清零

位 2 **STKERRCLR**：粘滞错误清零 (sticky error clear)
0：无影响
1：DP_CTRL/STATR 寄存器中的 STICKYERR 位清零

位 1 保留，必须保持复位值。

位 0 **DAPABORT**：如果返回的 WAIT 响应数量过多，则当前 AP 事务中止 (current AP transaction aborted if excessive number of WAIT responses returned)

该位指示事务已停止。

0：无影响

1：事务中止

DP 控制和状态寄存器 (DP_CTRLSTATR)

DP control and status register

偏移地址：0x4

复位值：0x0000 0000

当 DP_SELECTR.DPBANKSEL[3:0] = 0x0 时可访问该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	CDBGPWRUPACK	CDBGPWRUPREQ	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
		r	r												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATAERR	READOK	STICKYERR	Res.	Res.	Res.	STICKYORUN	ORUNDETECT
								r	r	r				r	r

位 31:30 保留，必须保持复位值。

位 29 **CDBGPWRUPACK**：调试上电确认 (debug power-up acknowledge)

请参见*第 41.2.5 节：调试和低功耗模式*中的说明。

0：门控 DCLK

1：使能 DCLK

位 28 **CDBGPWRUPREQ**：调试上电请求 (debug power-up request)

该位控制 DCLK 使能请求信号。

0：请求 DCLK 门控

1：请求 DCLK 使能

位 27:8 保留，必须保持复位值。

位 7 **WDATAERR**：SW-DP 模式下的写入数据错误（只读）(write data error (read only) in SW-DP)

该位指示写入的数据阶段存在奇偶校验或组帧错误，或者已被 DP 接受的写入操作被丢弃，而不是提交给 AP。

向 ABORT.WDERRCLR 位写入 1 即可将此位复位。

0：无错误

1：发生错误

注意：在 JTAG-DP 模式下，该位保留。

位 6 **READOK**：SW-DP 下的 AP 读取响应（只读）(AP read response (read-only) in SW-DP)

该位表示对最后一个 AP 读访问的响应。

0：读操作不正常

1：读操作正常

注意：在 JTAG-DP 模式下，该位保留。

位 5 **STICKYERR**：事务错误 (Transaction error)（在 SW-DP 模式下，该位只读；在 JTAG-DP 模式下，该位可读/写）

该位指示 AP 事务中发生错误。该位通过向 DP_ABORTR.STKERRCLR 位写入 1 来复位（在 SW-DP 和 JTAG-DP 下）

0：无错误

1：发生错误

位 4:2 保留，必须保持复位值。

位 1 **STICKYORUN**：上溢 (Overrun)（在 SW-DP 模式下，该位只读；在 JTAG-DP 模式下，该位可读/写）

该位表示发生了上溢（在完成上一个事务之前接收到新的事务）。该位仅在 ORUNDETECT 位置 1 时才置 1。该位通过向 DP_ABORTR.ORUNERRCLR 位写入 1 来复位（在 SW-DP 和 JTAG-DP 下）。

0：无上溢

1：发生上溢

位 0 **ORUNDETECT**：上溢检测模式使能 (overrun detection mode enable)

0：禁止

1：使能。如果发生上溢，则 STICKYORUN 位将置 1，同时后续事务被阻止，直到 STICKYORUN 位清零。

DP 数据链路控制寄存器 (DP_DLCR)

DP data link control register

偏移地址：0x4

复位值：0x0000 0000

当 DP_SELECTR.DPBANKSEL[3:0] = 0x1 时可访问该寄存器。

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

位 31:10 保留，必须保持复位值。

位 9:8 **TURNROUND[1:0]**: SWDIO 的三态周期 (tristate period for SWDIO)

0x0：1 个数据位周期

位 7:6 **WIREMODE[1:0]**：SW-DP 模式 (SW-DP mode)
0x0：同步模式

位 5:0 保留，必须保持复位值。

DP 目标标识寄存器 (DP_TARGETIDR)

DP target identification register

偏移地址：0x4

复位值：0xXXXX 0041

当 DP_SELECTR.DPBANKSEL[3:0] = 0x2 时可访问该寄存器。

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

位 31:28 **TREVISION[3:0]**：目标版本 (target revision)

位 27:12 **TPARTNO[15:0]**：目标产品编号 (target part number)
0x4740：STM32H503xx

位 11:1 **TDESIGNER[10:0]**：目标设计人员 JEDEC 代码 (target designer JEDEC code)
0x020：意法半导体

位 0 保留，必须保持复位值。

DP 数据链路协议标识寄存器 (DP_DLPIDR)

DP data link protocol identification register

偏移地址：0x4

复位值：0x0000 0001

当 DP_SELECTR.DPBANKSEL[3:0] = 0x3 时可访问该寄存器。

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

位 31:28 **TINSTANCE[3:0]**：目标实例编号 (target instance number)
在多点系统中，该位域定义器件的实例编号。
0x0：实例编号 0

位 27:4 保留，必须保持复位值。

位 3:0 **PROTSVN[3:0]**：串行线调试协议版本 (Serial-wire debug protocol version)
0x1：版本 2

DP 事件状态寄存器 (DP_RESENDR)

DP event status register

偏移地址：0x8

复位值：0x0000 0000

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

位 31:0 **RESEND[31:0]**：最后一次 AP 读取或 DP_RDBUFF 读取返回的值 (value returned by the last AP read or DP_RDBUFF read)
在读传输损坏的情况下使用该寄存器。

DP 访问端口选择寄存器 (DP_SELECTR)

DP access port select register

偏移地址：0x8

复位值：0xXXXX XXXX

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

位 31:24 **APSEL[7:0]**：访问端口选择 (access port select)
该位域选择下一个事务的访问端口。
0x00：AP0——系统调试访问端口 (APB-AP)
0x01：AP1——Cortex®-M33 调试访问端口 (AHB-AP)
其他值：保留

位 23:8 保留，必须保持复位值。

位 7:4 **APBANKSEL[3:0]**：AP 寄存器存储区域选择 (AP register bank select)
该位域选择活动 AP 上的 4 字寄存器存储区域，以用于下一个事务。

位 3:0 **DPBANKSEL[3:0]**：DP 寄存器存储区域选择 (DP register bank select)

该位域选择调试端口上地址为 0x4 的寄存器。

0x0：DP_CTRLSTAT 寄存器

0x1：DP_DLCR 寄存器

0x2：DP_TARGETID 寄存器

0x3：DP_DLPIDR 寄存器

其他值：保留

DP 读缓冲区寄存器 (DP_RDBUFFR)

DP read buffer register

偏移地址：0xC

复位值：0x0000 0000

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

位 31:0 **RDBUFF[31:0]**：最后一次 AP 读取访问返回的值 (value returned by the last AP read access)

AP 读访问返回的值可通过对同一地址进行第二次读访问来获得，这将在相应的总线上启动新的事务，或者还可以从该寄存器读取，在这种情况下不会发生新的 AP 事务。

41.3.4 调试端口寄存器映射和复位值

这些寄存器未在 CPU 存储器总线上.只能通过 SW-DP 和 JTAG-DP 调试接口访问。

调试端口地址偏移为 4 位宽，在 JTAG-DP 寄存器 DPACC 或 SW-DP 数据包请求 A[3:2] 位域中定义两个最高有效位。两个最低有效位是 00。

表 355. 调试端口寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0	**DP_DPIDR**	REVISION [3:0]				PARTNO[7:0]								Res.	Res.	Res.	MIN	VERSION [3:0]				DESIGNER[10:0]											Res.
	Reset value	0	1	1	0	1	0	1	1	1	0	1	0				0	0	0	1	0	0	1	0	0	0	1	1	1	0	1	1	1
0x0	**DP_ABORTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ORUNERRCLR	WDERRCLR	STKERRCLR	Res.	DAPABORT
	Reset value																												0	0	0		0

表 355. 调试端口寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x4(1)	**DP_CTRLSTATR**	Res.	Res.	CDBGPWRUPACK	CDBGPWRUPREQ	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATAERR	READOK	STICKYERR	Res.	Res.	Res.	STICKYORUN	ORUNDETECT
	Reset value			0	0																					0	0	0				0	0
0x4(2)	**DP_DLCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TURNROUND[1:0]		WIREMODE[1:0]		Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																							0	0	0	0						
0x4(3)	**DP_TARGETIDR**	TREVISION [3:0]				TPARTNO[15:0]																TDESIGNER[10:0]											Res.
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
0x4(4)	**DP_DLPIDR**		TINSTANCE[3:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PROTSVN[3:0]			
	Reset value	0	0	0	0																									0	0	0	1
0x8	**DP_RESENDR**	RESEND[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x8	**DP_SELECTR**	APSEL[7:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	APBANKSEL[3:0]				DPBANKSEL[3:0]			
	Reset value	x	x	x	x	x	x	x	x																	x	x	x	x	x	x	x	x
0xC	**DP_RDBUFFR**	RDBUFF[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. DP_SELECTR.DPBANKSEL[3:0] = 0x0。
2. DP_SELECTR.DPBANKSEL[3:0] = 0x1。
3. DP_SELECTR.DPBANKSEL[3:0] = 0x2。
4. DP_SELECTR.DPBANKSEL[3:0] = 0x3。

41.4 访问端口

DP 上附有两个访问端口 (AP)。

- 系统调试访问端口 (AP0)：允许通过 APB 总线访问 DBGMCU 和系统 ROM 表。
- Cortex®-M33 调试访问端口 (AP1)：允许通过内部 AHB 总线访问 Cortex®-M33 处理器内核中集成的调试和跟踪功能。

41.4.1 访问端口寄存器

访问端口均为 MEM-AP 类型：调试和跟踪组件寄存器均映射在 AHB 的地址空间中。调试器将 AP 视为一组 32 位寄存器，每个存储区域有四个寄存器。其中一些寄存器用于配置或监视 AP 本身，而其他寄存器则用于执行总线传输。*表 357* 列出了 AP 寄存器。

AP 寄存器的地址包括以下位域：

- 位[7:4]：*DP 访问端口选择寄存器 (DP_SELECTR)* 中 APBANKSEL[3:0] 位域的内容
- 位 [3:2]：JTAG-DP 模式下 APACC 数据寄存器（请参见*表 350*）或 SW-DP 数据包请求（请参见*表 351*）中的 A(3:2) 位域的内容，具体取决于使用的调试接口
- 位 [1:0]：始终设置为 0

DP_SELECTR.APSEL[3:0] 位域的内容定义被访问的 MEM-AP。

表 356. MEM-AP寄存器

地址	APBANKSEL	A(3:2)	名称	说明
0x00	0x0	0	CSWR	控制/状态字寄存器
0x04	0x0	1	TAR	传输地址寄存器 总线事务的目标地址。
0x08	-	-	-	保留
0x0C	0x0	3	DRWR	数据读/写寄存器 访问该寄存器可触发调试总线上的相应事务，其目标地址为 TAR[31:0] 中的地址。
0x10	0x1	0	BD0R	分区数据 0 寄存器 访问该寄存器可触发调试总线上的相应事务，其目标地址为 TAR[31:4] 中的地址 + 0x0。
0x14	0x1	1	BD1R	分组数据 1 寄存器 访问该寄存器可触发调试总线上的相应事务，其目标地址为 TAR[31:4] 中的地址 + 0x4。
0x18	0x1	2	BD2R	分组数据 2 寄存器 访问该寄存器可触发调试总线上的相应事务，其目标地址为 TAR[31:4] 中的地址 + 0x8。
0x1C	0x1	3	BD3R	分组数据 3 寄存器 访问该寄存器可触发调试总线上的相应事务，其目标地址为 TAR[31:4] 中的地址 + 0xC。
0x20	-	-	-	保留
0x24 至 0xEC	-	-	-	保留
0xF0	-	-	-	保留
0xF4	0xF	1	CFGR	配置寄存器（只读）
0xF8	0xF	2	BASER	调试基址寄存器（只读） ROM 表的基址
0xFC	0xF	3	IDR	标识寄存器（只读）

调试器可访问 AP 寄存器，如下所示：

1. 对 *DP 访问端口选择寄存器 (DP_SELECTR)* 中的 APSEL[3:0] 位域编程以选择 AP，对 DP_SELECTR 中的 APBANKSEL[3:0] 位域编程以选择要访问的寄存器存储区域。
2. 如果使用 JTAG，则使用存储区中的寄存器地址对 APACC 数据寄存器中的 A(3:2) 位域编程。对 RnW 位编程，选择读操作或写操作。如果选择写操作，则使用写入数据对 DATA 位域进行编程。如果使用 SWD，则 A(3:2) 和 RnW 位域包含在发送到 SW-DP 的数据包请求字当中，数据包请求字的 APnDP 位置 1（请参见*表 351*）。写入数据在数据阶段发送。

调试器可通过 AP 寄存器访问存储器映射的调试组件寄存器（使用上述 AP 寄存器访问过程），如下所示：

1. 对 *APx 传输地址寄存器 (APx_TAR) (x = 0, 1)* 中的事务目标地址编程。
2. 对 *AP1 控制/状态字寄存器 (AP1_CSWR)* 编程，必要时，请使用传输参数（例如 AddrInc）。
3. 针对 AP_TA 中保存的地址，写入或读取 *Apx 数据读/写寄存器 (APx_DRWR) (x = 0, 1)*，以启动总线事务。或者，读取或写入 *Apx 分区数据 n 寄存器 (APx_BDnR) (x = 0, 1)*，以触发访问地址 TAR[31:4] + n，从而无需更改 AP_TAR 寄存器中的地址即可访问多达四个连续的地址。

有关 MEM-AP 的更多详细信息，请参见《Arm® 调试接口架构规范》*[1]*。

AP0 控制/状态字寄存器 (AP0_CSWR)

AP0 control/status word register

偏移地址：0x0

复位值：0x8000 0042

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DBGSWEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	MODE[3:0]				TRINPROG	DEVICEEN	ADDRINC[1:0]		Res.	SIZE[2:0]		
				rw	rw	rw	rw	r	r	rw	rw		r	r	r

位 31 **DBGSWEN**：软件访问使能 (software access enable)
使能或禁止对 APB 总线进行软件访问
0：禁止软件访问
1：使能软件访问

位 30:12 保留，必须保持复位值。

位 11:8 **MODE[3:0]**：工作模式 (mode of operation)
0b0000：正常下载或上传
其他值：保留

位 7 **TRINPROG**：传输正在进行 (transfer in progress)（只读）
该位域指示 APB 主端口当前是否正在进行传输
0：无正在进行的 APB 传输。
1：正在进行 APB 传输

位 6 **DEVICEEN**：器件使能状态 (device enable status)（只读）
1：APB 传输始终使能

位 5:4 **ADDRINC[1:0]**：自动递增模式 (auto-increment mode)
定义了在事务结束之后 TAR 地址是否会自动递增。
0x0：不自动递增
0x1：地址按事务字节（SIZE 位域）的大小递增。
其他值：保留

位 3 保留，必须保持复位值。

位 2:0 **SIZE[2:0]**：下一个存储器访问事务的大小 (size of next memory access transaction)
0x2：字（32 位）

AP1 控制/状态字寄存器 (AP1_CSWR)

AP1 control/status word register

偏移地址：0x0

复位值：0x43X0 00X2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	SPROT	Res.	Res.	PROT[3:0]				SPISTATUS	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	rw			rw	rw	rw	rw	r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRINPROG	DBGSTATUS	ADDRINC[1:0]		Res.	SIZE[2:0]		
								r	r	rw	rw		rw	rw	rw

位 31 保留，必须保持复位值。

位 30 **SPROT**：安全传输请求 (secure transfer request)
该位域为总线传输设置保护属性 HPROT[6]。
0：保留
1：非安全传输，HPROT[6] = 高电平
如果 SPIDEN = 0 且 SPROT = 0，则不会发生总线传输

位 29:28 保留，必须保持复位值。

位 27:24 **PROT[3:0]**：总线传输保护 (bus transfer protection)
该位域为总线传输设置保护属性 HPROT[3:0]。
0bXXX0：指令访问
0bXXX1：数据访问
0bXX0X：用户模式
0bXX1X：特权模式
0bX0XX：不可缓冲
0bX1XX：可缓冲
0b0XXX：不可共享、不可查找、不可修改
0b1XXX：可共享、查找、可修改

位 23 **SPISTATUS**：安全调试身份验证状态 (secure debug authentication status)
该位域指示 SPIDEN 信号的状态
0：不允许安全 AHB 传输

位 22:8 保留，必须保持复位值。

位 7 **TRINPROG**：传输正在进行 (transfer in progress)（只读）
该位域指示 APB 主端口当前是否正在进行传输
0：无正在进行的 AHB 传输。
1：正在进行 AHB 传输

位 6 **DBGSTATUS**：调试使能 (DBGEN) 状态 (debug enable (DBGEN) status)
0：阻止 AHB 传输
1：使能 AHB 传输

位 5:4 **ADDRINC[1:0]**：自动递增模式 (auto-increment mode)
定义了在事务结束之后 TAR 地址是否会自动递增。
0x0：不自动递增
0x1：地址按事务字节（SIZE 位域）的大小递增。单次传输
0x2：地址按事务字节（SIZE 位域）的大小递增。。将四个 8 位传输或两个 16 位传输压缩为一个 32 位 DAP 传输。多个事务在 AHB 接口上执行。
其他值：保留

位 3 保留，必须保持复位值。

位 2:0 **SIZESIZE[2:0]**：下一个存储器访问事务的大小 (size of next memory access transaction)
0x0：字节（8 位）
0x1：半字（16 位）
0x2：字（32 位）
其他值：保留

APx 传输地址寄存器 (APx_TAR) (x = 0, 1)

APx transfer address register

偏移地址：0x04

复位值：0xXXXX XXXX

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

位 31:0 **TA[31:0]**：当前传输的地址 (address of current transfer)。在 AP0 中，TA[1:0] 固定为 0。

Apx 数据读/写寄存器 (APx_DRWR) (x = 0, 1)

APx data read/write register

偏移地址：0x0C

复位值：0xXXXX XXXX

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

位 31:0 **TD[31:0]**：当前传输的数据 (data of current transfer)

Apx 分区数据 n 寄存器 (APx_BDnR) (x = 0, 1)

APx banked data n register

偏移地址：0x10 + 4 * n，（n = 0 到 3）

复位值：0x0000 0000

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

位 31:0 **DATA[31:0]**：当前传输到地址 TA [31:4] + 4 * n 的分区数据。
APx_BD0-3R 上不执行自动地址递增。只有字传输支持分区传输。

APx 基址寄存器 (APx_BASER) (x = 0, 1)

APx base address register

偏移地址：0xF8

复位值：AP 0：0xE00E 0003

复位值：AP 1：0xE00F E003

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

位 31:12 **BASEADDR[31:12]**：第一个 ROM 表的基址（位 31 到 12）(base address (bits 31 to 12) of the first ROM table)

由于 ROM 表必须与 4 KB 边界对齐，因此 12 个 LSB 都为零。

0xE00E0：AP0

0xE00FE：AP1

位 11:2 保留，必须保持复位值。

位 1 **FORMAT**：基址寄存器格式 (base-address register format)

1：Arm® 调试接口 v5

位 0 **ENTRYPRESENT**：调试组件存在 (debug components presence)

指示访问端口总线上有无调试组件。

1：有调试组件

Apx 标识寄存器 (APx_IDR) (x = 0, 1)

APx identification register

偏移地址：0xFC

复位值：AP 0：0x5477 0002

复位值：AP 1：0x8477 0001

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

位 31:28 **REVISION[3:0]**：版本号 (revision number)
0x5：r1p0
0x8：r0p9

位 27:24 **JEDECBANK[3:0]**：JEDEC 存储区域 (JEDEC bank)
0x4：Arm®

位 23:17 **JEDECCODE[6:0]**：JEDEC 代码
0x3B：Arm®

位 16:13 **CLASS[3:0]**：
0x1：MEM-AP

位 12:8 保留，必须保持复位值。

位 7:0 **IDENTITY[7:0]**：
0x1：AHB-AP
0x2：APB-AP

41.4.2 访问端口寄存器映射

这些寄存器未在 CPU 存储器总线上，只能通过 SW-DP 和 JTAG-DP 调试接口访问。

访问端口地址为 8 位宽，由 DP_SELECTR.APBANKSEL[3:0] 位域和 JTAG-DP 寄存器 DPACC 或 SW-DP 数据包请求 A[3:2] 位域定义。

表 357. 访问端口寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**AP0_CSWR**	DBGSWEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MODE[3:0]				TRINPROG	DEVICEEN	ADDRINC[1:0]		Res.	SIZE[2:0]		
	Reset value	1																				0	0	0	0	0	1	0	0		0	1	0
0x00	**AP1_CSWR**	Res.	SPROT	Res.	Res.	PROT[3:0]				SPISTATUS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MODE[3:0]				TRINPROG	DBGSTATUS	ADDRINC[1:0]		Res.	SIZE[2:0]		
	Reset value		1			0	0	1	1	X												0	0	0	0	0	X	0	0		0	1	0
0x04	**APx_TAR**	TA[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x08	Reserved	Reserved																															
0x0C	**APx_DRWR**	TD[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x10	**APx_BD0R**	DATA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	**APx_BD1R**	DATA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	**APx_BD2R**	DATA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	**APx_BD3R**	DATA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 357. 访问端口寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x20 to 0xF4	Reserved	Reserved																															
0xF8	**AP0_BASER**	BASEADDR[31:12]																				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FORMAT	ENTRYPRESENT
	Reset value	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0											1	1
0xF8	**AP1_BASER**	BASEADDR[31:12]																				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FORMAT	ENTRYPRESENT
	Reset value	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0											1	1
0xFC	**AP0_IDR**	REVISION[3:0]				JEDECBANK[3:0]				JEDECCODE[6:0]							CLASS[3:0]				Res.	Res.	Res.	Res.	Res.	IDENTITY[7:0]							
	Reset value	0	1	0	1	0	1	0	0	0	1	1	1	0	1	1	1	0	0	0						0	0	0	0	0	0	1	0
0xFC	**AP0_IDR**	REVISION[3:0]				JEDECBANK[3:0]				JEDECCODE[6:0]							CLASS[3:0]				Res.	Res.	Res.	Res.	Res.	IDENTITY[7:0]							
	Reset value	1	0	0	0	0	1	0	0	0	1	1	1	0	1	1	1	0	0	0						0	0	0	0	0	0	0	1

41.5 ROM 表

ROM 表是一个 CoreSight™ 组件，其中包含可通过其所连接的访问端口进行访问的 CoreSight 调试组件的基址。通过这些表，调试器可以自动发现 CoreSight 系统的拓扑。

每个访问端口 APn 后面都有一个顶层 ROM 表。该 ROM 表的基址可通过读取访问端口的 APn_BASER 寄存器来获得。顶层 ROM 表也可以指向其他 ROM 表。

AP0 基址寄存器 AP0_BASER 指向系统 ROM 表。它包含 DBGMCU 的基址指针。

由调试器访问时，系统 ROM 表占用 4 KB、32 位宽的地址空间块（从 0xE00E 0000 到 0xE00E 0FFC）。CPU 可以在 0x4402 0000 到 0x4402 0FFC 的地址范围内对其进行访问。

表 358. 系统 ROM 表

偏移地址（ROM 表中）	组件名称	组件基址	组件偏移地址	大小 (KB)	入口
0x000	DBGMCU	0xE00E 4000（调试器） 0x4402 4000 (CPU)	0x0000 4000	4	0x0000 4003
0x004	表顶部	-	-	-	0x0000 0000

表 358. 系统 ROM 表（续）

偏移地址（ROM 表中）	组件名称	组件基址	组件偏移地址	大小 (KB)	入口
0x008 至 0xFC8	保留	-	-	-	0x0000 0000
0xFCC 到 0xFFC	ROM 表寄存器	-	-	-	请参见*表 361*

CPU 子系统中有两个 ROM 表。AP1 基址寄存器 AP1_BASER 指向 MCU ROM 表。它包含处理器 ROM 表和 TPIU 寄存器的基址指针。

MCU ROM 表（参见下表）占用 4 KB、32 位宽的地址空间块（从 0xE00F E000 到 0xE00F EFFC）。

表 359. MCU ROM 表

偏移地址（ROM 表中）	组件名称	组件基址	组件偏移地址	大小 (KB)	入口
0x000	处理器 ROM 表	0xE00F F000	0x0000 1000	4	0x0000 1003
0x004	TPIU	0xE004 0000	0xFFF4 2000	4	0xFFF4 2003
0x008	保留	-	-	-	0x1FF0 2002
0x00C	保留	-	-	-	0x1FF0 2002
0x010	表顶部	-	-	-	0x0000 0000
0x014 到 0xFC8	保留	-	-	-	0x0000 0000
0xFCC 到 0xFFC	ROM 表寄存器	-	-	-	请参见*表 362*

处理器 ROM 表包含系统控制空间 (SCS) 寄存器的基址指针（允许调试器识别 CPU 内核），以及 BPU、DWT、ITM、ETM 和 CTI 的基址指针。

处理器 ROM 表（参见下表）占用 4 KB、32 位宽的地址空间块（从 0xE00F F000 到 0xE00F FFFC）。

表 360. 处理器 ROM 表

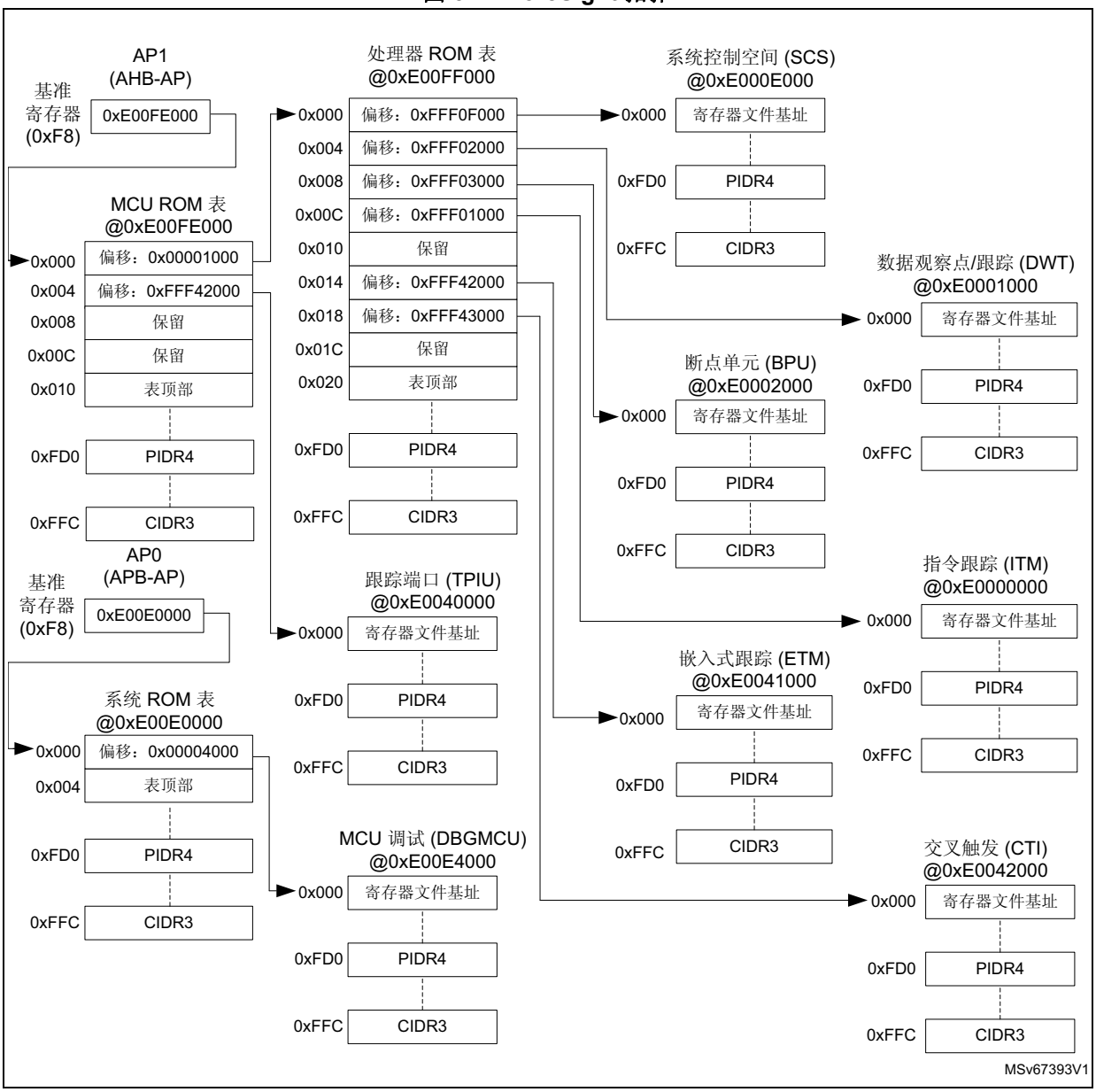
地址（ROM 表中）	组件名称	组件基址	组件偏移地址	大小 (KB)	入口
0xE00F F000	SCS	0xE000 E000	0xFFF0 F000	4	0xFFF0 F003
0xE00F F004	DWT	0xE000 1000	0xFFF0 2000	4	0xFFF0 2003
0xE00F F008	BPU	0xE000 2000	0xFFF0 3000	4	0xFFF0 3003
0xE00F F00C	ITM	0xE000 0000	0xFFF0 1000	4	0xFFF0 1003
0xE00F F010	保留	-	-	-	0xFFF4 1002
0xE00F F014	ETM	0xE004 1000	0xFFF4 2000	4	0xFFF4 2003
0xE00F F018	CTI	0xE004 2000	0xFFF4 3000	4	0xFFF4 3003
0xE00F F01C	保留	-	-	-	0xFFF4 4002
0xE00F F020	表顶部	-	-	-	0x0000 0000

表 360. 处理器 ROM 表（续）

地址（ROM 表中）	组件名称	组件基址	组件偏移地址	大小 (KB)	入口
0xE00F F024 至 0xE00F FFC8	保留	-	-	-	0x0000 0000
0xE00F FFCC 至 0xE00F FFFC	ROM 表寄存器	-	-	-	请参见*表 363*

CoreSight™ 组件的拓扑如下图所示。

图 514. CoreSight 拓扑



41.5.1 系统 ROM 表寄存器

系统 ROM 存储器类型寄存器 (SYSROM_MEMTYPER)

System ROM memory type register

偏移地址：0xFCC

复位值：0x0000 0001

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

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

位 31:1 保留，必须保持复位值。

位 0 **SYSMEM**：系统存储器 (system memory)
0x1：该总线上有系统存储器

系统 ROM CoreSight 外设标识寄存器 4 (SYSROM_PIDR4)

System ROM CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)
0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)
0x0：意法半导体 JEDEC 连续代码

系统 ROM CoreSight 外设标识寄存器 0 (SYSROM_PIDR0)

System ROM CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 00XX

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (part number bits [7:0])

0x74：STM32H503xx

系统 ROM CoreSight 外设标识寄存器 1 (SYSROM_PIDR1)

System ROM CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 000X

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])

0x0：意法半导体 JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])

0x4：STM32H503xx

系统 ROM CoreSight 外设标识寄存器 2 (SYSROM_PIDR2)

System ROM CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 000A

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x0：rev r0p0

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
1：JEDEC 指定的设计人员标识

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x2：意法半导体 JEDEC 代码

系统 ROM CoreSight 外设标识寄存器 3 (SYSROM_PIDR3)

System ROM CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)
0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)
0x0：无客户修改

系统 ROM CoreSight 组件标识寄存器 0 (SYSROM_CIDR0)

System ROM CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]

0x0D：公用标识值

系统 ROM CoreSight 外设标识寄存器 1 (SYSROM_CIDR1)

System ROM CoreSight peripheral identity register 1

偏移地址：0xFF4

复位值：0x0000 0010

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件标识位 [15:12] - 组件类 (component class)

0x1：ROM 表组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]

0x0：公用标识值

系统 ROM CoreSight 组件标识寄存器 2 (SYSROM_CIDR2)

System ROM CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]
0x05：公用标识值

系统 ROM CoreSight 组件标识寄存器 3 (SYSROM_CIDR3)

System ROM CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位 [31:24]
0xB1：公用标识值

41.5.2 系统 ROM 表寄存器映射

表 361. 系统 ROM 表寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFCC	SYSROM_MEMTYPER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSMEM
	Reset value																																1

表 361. 系统 ROM 表寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFD0	**SYSROM_PIDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE [3:0]				JEP106CON [3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFD4 to FDC	Reserved	Reserved																															
0xFE0	**SYSROM_PIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PARTNUM[7:0]							
	Reset value																									X	X	X	X	X	X	X	X
0xFE4	**SYSROM_PIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID [3:0]				PARTNUM [11:8]			
	Reset value																									0	0	0	0	X	X	X	X
0xFE8	**SYSROM_PIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION [3:0]				JEDEC	JEP106ID [6:4]		
	Reset value																									0	0	0	0	1	0	1	0
0xFEC	**SYSROM_PIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFF0	**SYSROM_CIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**SYSROM_CIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLASS[3:0]				PREAMBLE [11:8]			
	Reset value																									0	0	0	1	0	0	0	0
0xFF8	**SYSROM_CIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1
0xFFC	**SYSROM_CIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*表 358：系统 ROM 表*。

41.5.3 MCU ROM 表寄存器

MCU ROM 存储器类型寄存器 (MCUROM_MEMTYPER)

MCU ROM memory type register

偏移地址：0xFCC

复位值：0x0000 0001

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

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

位 31:1 保留，必须保持复位值。

位 0 **SYSMEM**：系统存储器 (system memory)
0x1：该总线上有系统存储器

MCU ROM CoreSight 外设标识寄存器 4 (MCUROM_PIDR4)

MCU ROM CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)
0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)
0x0：意法半导体 JEDEC 连续代码

MCU ROM CoreSight 外设标识寄存器 0 (MCUROM_PIDR0)

MCU ROM CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 00XX

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (part number bits [7:0])
0x74：STM32H503xx

MCU ROM CoreSight 外设标识寄存器 1 (MCUROM_PIDR1)

MCU ROM CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 000X

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])
0x0：意法半导体 JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])
0x4：STM32H503xx

MCU ROM CoreSight 外设标识寄存器 2 (MCUROM_PIDR2)

MCU ROM CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 000A

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x0：rev r0p0

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
1：JEDEC 指定的设计人员标识

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x2：意法半导体 JEDEC 代码

MCU ROM CoreSight 外设标识寄存器 3 (MCUROM_PIDR3)

MCU ROM CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)

0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)

0x0：无客户修改

MCU ROM CoreSight 组件标识寄存器 0 (MCUROM_CIDR0)

MCU ROM CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]

0x0D：公用标识值

MCU ROM CoreSight 外设标识寄存器 1 (MCUROM_CIDR1)

MCU ROM CoreSight peripheral identity register 1

偏移地址：0xFF4

复位值：0x0000 0010

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件标识位 [15:12] - 组件类 (component class)
0x1：ROM 表组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]
0x0：公用标识值

MCU ROM CoreSight 组件标识寄存器 2 (MCUROM_CIDR2)

MCU ROM CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]
0x05：公用标识值

MCU ROM CoreSight 组件标识寄存器 3 (MCUROM_CIDR3)

MCU ROM CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位[31:24]
0xB1：公用标识值

41.5.4 MCU ROM 表寄存器映射

表 362. MCU ROM 表寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFCC	**MCUROM_ MEMTYPER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSMEM
	Reset value																																1
0xFD0	**MCUROM_PIDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE [3:0]				JEP106CON [3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFD4 to FDC	Reserved	Reserved																															
0xFE0	**MCUROM_PIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PARTNUM[7:0]							
	Reset value																									X	X	X	X	X	X	X	X
0xFE4	**MCUROM_PIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID [3:0]				PARTNUM [11:8]			
	Reset value																									0	0	0	0	X	X	X	X
0xFE8	**MCUROM_PIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION [3:0]				JEDEC	JEP106ID [6:4]		
	Reset value																									0	0	0	0	1	0	1	0
0xFEC	**MCUROM_PIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFF0	**MCUROM_CIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**MCUROM_CIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLASS[3:0]				PREAMBLE [11:8]			
	Reset value																									0	0	0	1	0	0	0	0

表 362. MCU ROM 表寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFF8	MCUROM_CIDR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1
0xFFC	MCUROM_CIDR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*表 359：MCU ROM 表*。

41.5.5 处理器 ROM 表寄存器

CPU ROM 存储器类型寄存器 (CPUROM_MEMTYPER)

CPU ROM memory type register

偏移地址：0xFCC

复位值：0x0000 0001

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

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

位 31:1 保留，必须保持复位值。

位 0 **SYSMEM**：系统存储器 (system memory)
1：该总线上有系统存储器

CPU ROM CoreSight 外设标识寄存器 4 (CPUROM_PIDR4)

CPU ROM CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0004

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)
0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)
0x4：ARM® JEDEC 连续代码

CPU ROM CoreSight 外设标识寄存器 0 (CPUROM_PIDR0)

CPU ROM CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 00C9

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (Part number bits [7:0])
0xC9：Cortex®-M33

CPU ROM CoreSight 外设标识寄存器 1 (CPUROM_PIDR1)

CPU ROM CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 00B4

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])
0xB：ARM® JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])
0x4：Cortex®-M33

CPU ROM CoreSight 外设标识寄存器 2 (CPUROM_PIDR2)

CPU ROM CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 000B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x0：rev r0p0

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
1：JEDEC 指定的设计人员 ID

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x3：Arm® JEDEC 代码

CPU ROM CoreSight 外设标识寄存器 3 (CPUROM_PIDR3)

CPU ROM CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)
0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)
0x0：无客户修改

CPU ROM CoreSight 组件标识寄存器 0 (CPUROM_CIDR0)

CPU ROM CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]
0x0D：公用标识值

CPU ROM CoreSight 外设标识寄存器 1 (CPUROM_CIDR1)

CPU ROM CoreSight peripheral identity register 1

偏移地址：0xFF4

复位值：0x0000 0010

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件标识位 [15:12] - 组件类 (component class)
0x1：ROM 表组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]
0x0：公用标识值

CPU ROM CoreSight 组件标识寄存器 2 (CPUROM_CIDR2)

CPU ROM CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]

0x05：公用标识值

CPU ROM CoreSight 组件标识寄存器 3 (CPUROM_CIDR3)

CPU ROM CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位 [31:24]

0xB1：公用标识值

41.5.6 处理器 ROM 表寄存器映射

表 363. CPU ROM 表寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFCC	**CPUROM_ MEMTYPER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSMEM
	Reset value																																1
0xFD4 to FDC	Reserved	Reserved																															
0xFD0	**CPUROM_PIDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE [3:0]				JEP106CON [3:0]			
	Reset value																									0	0	0	0	0	1	0	0
0xFE0	**CPUROM_PIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PARTNUM[7:0]							
	Reset value																									1	1	0	0	1	0	0	1
0xFE4	**CPUROM_PIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID [3:0]				PARTNUM [11:8]			
	Reset value																									1	0	1	1	0	1	0	0
0xFE8	**CPUROM_PIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION [3:0]				JEDEC	JEP106ID [6:4]		
	Reset value																									0	0	0	0	1	0	1	1
0xFEC	**CPUROM_PIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFF0	**CPUROM_CIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**CPUROM_CIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLASS[3:0]				PREAMBLE [11:8]			
	Reset value																									0	0	0	1	0	0	0	0
0xFF8	**CPUROM_CIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1
0xFFC	**CPUROM_CIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*表 360：处理器 ROM 表*。

41.6 数据观察点和跟踪单元 (DWT)

DWT 提供了四个比较器，可用作以下其中一项：

- 观察点
- ETM 触发器
- PC 采样触发器
- 数据地址采样触发器
- 数据比较器（仅适用于 COMP 1）
- 时钟周期计数比较器（仅适用于 COMP 0）

还包含以下计数器：

- 时钟周期计数器
- 分支指令
- 加载存储单元 (LSU) 操作
- 睡眠周期
- 每条指令周期数计数器
- 中断开销

DWT 比较器将其 *DWT 比较器 x 寄存器 (DWT_COMPxR)* 中保存的值与下列其中一项进行比较：

- 数据地址
- 指令地址
- 数据值
- 周期计数值，仅适用于 COMP 0

比较器可以使用掩码进行地址匹配，因此可匹配一系列地址。

一旦匹配成功时，比较器会生成以下其中一项：

- 一个或多个 DWT 数据跟踪数据包，包括一个或多个：
  - 引发数据访问的指令的地址
  - 偏移地址，数据访问地址的位 [15:0]
  - 匹配的数据值
- 观察点调试事件（在 PC 值上或者在被访问的数据地址上）
- 指示匹配超出 DWT 单元的 CMPMATCH[N] 事件

观察点调试事件会生成 DebugMonitor 异常，或者导致处理器停止执行并进入调试状态。

有关如何使用 DWT 的更多详细信息，请参见《Arm®v8-M 架构参考手册》*[4]*。

41.6.1 DWT 寄存器

DWT 寄存器的地址范围为 0xE000 1000 到 0xE000 1FFC。

DWT 控制寄存器 (DWT_CTRLR)

DWT control register

偏移地址：0x000

复位值：0x4000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUMCOMP[3:0]				NOTRCPKT	NOEXTTRIG	NOCYCCNT	NOPRFCNT	CYCDISS	CYCEVTENA	FOLDEVTENA	LSUEVTENA	SLEEPEVTENA	EXCEVTENA	CPIEVTENA	EXCTRCENA
r	r	r	r	r	r	r	r	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	PCSAMPLENA	SYNCTAP[1:0]		CYCTAP	POSTINIT[3:0]				POSTRESET[3:0]				CYCCNTENA
			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:28 **NUMCOMP[3:0]**：设计的比较器数量 (number of comparators implemented)，只读值
0x4：四个比较器

位 27 **NOTRCPKT**：支持跟踪采样和异常跟踪 (trace sampling and exception tracing support)，只读值
0：支持

位 26 **NOEXTTRIG**：外部信号匹配 CMPMATCH 支持 (external match signal, CMPMATCH support)，只读值
0：支持

位 25 **NOCYCCNT**：循环计数器支持 (cycle counter support)，只读值
0：支持

位 24 **NOPRFCNT**：性能分析计数器支持 (profiling counter support)，只读值
0：支持

位 23 **CYCDISS**：安全模式禁止循环计数器 (cycle counter disabled secure)。
控制在安全模式下是否禁止循环计数器。
0：无影响
1：保留

位 22 **CYCEVTENA**：使能 POSTCNT 下溢事件计数器数据包生成 (enable for POSTCNT underflow event counter packet generation)
0：禁止
1：使能

位 21 **FOLDEVTENA**：使能折叠指令计数器上溢事件生成 (enable for folded instruction counter overflow event generation)
0：禁止
1：使能

位 20 **LSUEVTENA**：使能 LSU 计数器上溢事件生成 (enable for LSU counter overflow event generation)

0：禁止
1：使能

位 19 **SLEEPEVTENA**：使能睡眠计数器上溢事件生成 (enable for sleep counter overflow event generation)

0：禁止
1：使能

位 18 **EXCEVTENA**：使能异常开销计数器上溢事件生成 (enable for exception overhead counter overflow event generation)

0：禁止
1：使能

位 17 **CPIEVTENA**：使能 CPI 计数器上溢事件生成 (enable for CPI counter overflow event generation)

0：禁止
1：使能

位 16 **EXCTRCENA**：使能异常跟踪生成 (enable for exception trace generation)

0：禁止
1：使能

位 15:13 保留，必须保持复位值。

位 12 **PCSAMPLENA**：使能将 POSTCNT 计数器用作定时器，用于定期生成 PC 样本数据包 (enable for POSTCNT counter to be used as a timer for periodic PC sample packet generation)

0：禁止
1：使能

位 11:10 **SYNCTAP[1:0]**：同步数据包计数器点击 CYCCNT 计数器的位置 (position of synchronization packet counter tap on CYCCNT counter)

该位域确定同步数据包速率。
00：禁止，无同步数据包
01：点击位置为 CYCCNT[24]
10：点击位置为 CYCCNT[26]
11：点击位置为 CYCCNT[28]

位 9 **CYCTAP**：选择 POSTCNT 点击 CYCCNT 计数器的位置 (Selects the position of the POSTCNT tap on the CYCCNT counter)。

0：点击位置为 CYCCNT[6]
1：点击位置为 CYCCNT[10]

位 8:5 **POSTINIT[3:0]**：POSTCNT 计数器初始值 (initial value of the POSTCNT counter)

如果使能了 POSTCNT 计数器，则忽略对此位域的写操作。在写入 POSTINIT 之前，必须先将 CYCEVTENA 或 PCSAMPLENA 位复位。

位 4:1 **POSTRESET[3:0]**：POSTCNT 计数器的重载值 (reload value of the POSTCNT counter)

位 0 **CYCCNTENA**：使能 CYCCNT 计数器

0：禁止
1：使能

DWT 周期计数寄存器 (DWT_CYCCNTR)

DWT cycle count register

偏移地址：0x004

复位值：0x0000 0000

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

位 31:0 **CYCCNT[31:0]**：处理器时钟周期计数器 (processor clock cycle counter)

DWT CPI 计数寄存器 (DWT_CPICNTR)

DWT CPI count register

偏移地址：0x008

复位值：0xXXXX XXXX

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

位 31:8 保留，必须保持复位值。

位 7:0 **CPICNT[7:0]**：CPI 计数器 (CPI counter)

对执行多周期指令（由 DWT_LSUCNTR 记录的多周期指令除外）进行计数，并对任何指令读取停止 (Stall) 进行计数。

DWT 异常计数寄存器 (DWT_EXCCNTR)

DWT exception count register

偏移地址：0x00C

复位值：0xXXXX XXXX

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **EXCCNT[7:0]**：异常开销周期计数器 (exception overhead cycle counter)
对在异常处理中花费的周期数进行计数。

DWT 睡眠计数寄存器 (DWT_SLPCNTR)

DWT sleep count register

偏移地址：0x010

复位值：0xXXXX XXXX

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **SLEEPCNT[7:0]**：睡眠周期计数器 (sleep cycle counter)
对在睡眠模式（WFI、WFE、退出时睡眠）下消耗的周期数进行计数。

DWT LSU 计数寄存器 (DWT_LSUCNTR)

DWT LSU count register

偏移地址：0x014

复位值：0xXXXX XXXX

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

位 31:8 保留，必须保持复位值。

位 7:0 **LSUCNT[7:0]**：加载存储计数器 (load store counter)
对执行加载和存储指令所需的其他周期进行计数。

DWT 折叠计数寄存器 (DWT_FOLDCNTR)

DWT fold count register

偏移地址：0x018

复位值：0xXXXX XXXX

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

位 31:8 保留，必须保持复位值。

位 7:0 **FOLDCNT[7:0]**：折叠指令计数器 (folded instruction counter)
每一条只花费 0 周期的指令就递增计数。

DWT 程序计数器采样寄存器 (DWT_PCSR)

DWT program counter sample register

偏移地址：0x01C

复位值：0xXXXX XXXX

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

位 31:0 **EIASAMPLE[31:0]**：执行指令地址采样值 (executed instruction address sample value)
采样程序计数器的当前值。

DWT 比较器 x 寄存器 (DWT_COMPxR)

DWT comparator x register

偏移地址：0x020 + 0x010 * x，（x = 0 到 3）

复位值：0xXXXX XXXX

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

位 31:0 **COMP[31:0]**：用于比较的参考值 (reference value for comparison)

DWT 功能寄存器 0 (DWT_FUNCTR0)

DWT function register 0

偏移地址：0x028

复位值：0x5800 0000

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

位 31:27 **ID[4:0]**：能力标识 (capability identification)
标识比较器 0 的匹配能力。
0b01011：周期计数器、指令地址、数据地址和有值数据地址

位 26:25 保留，必须保持复位值。

位 24 **MATCHED**：比较器匹配 (comparator match)
指示自上次读取寄存器以来是否发生了比较器匹配。
0：不匹配
1：发生了匹配

位 23:12 保留，必须保持复位值。

位 11:10 **DATAVSIZE[1:0]**：数据值大小 (data value size)
定义通过数据值和数据地址比较器监视的对象的大小。
0x0：1 字节
0x1：2 字节
0x2：4 字节
0x3：保留

位 9:6 保留，必须保持复位值。

位 5:4 **ACTION[1:0]**：匹配时的操作 (action on match)
0x0：仅触发
0x1：生成调试事件
0x2：对于周期计数器、指令地址、数据地址、数据值或链接数据值比较器，生成数据跟踪匹配数据包。对于有值数据地址比较器，生成数据跟踪数据值数据包。
0x3：对于数据地址限制比较器，生成数据跟踪数据地址数据包。对于周期计数器、指令地址限制或数据地址比较器，生成数据跟踪 PC 值数据包。对于有值数据地址比较器，生成数据跟踪 PC 值数据包和数据跟踪数据值数据包。

位 3:0 **MATCH[3:0]**：匹配类型 (match type)
控制比较器 0 生成的匹配的类型。
有关该位域的可能值，请参见*[4]*。

DWT 功能寄存器 1 (DWT_FUNCTR1)

DWT function register 1

偏移地址：0x038

复位值：0xD000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ID[4:0]					Res.	Res.	MATCHED	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r	r	r	r			r								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DATAVSIZE[1:0]		Res.	Res.	Res.	Res.	ACTION[1:0]		MATCH[3:0]			
				rw	rw					rw	rw	rw	rw	rw	rw

位 31:27 **ID[4:0]**：能力标识 (capability identification)
标识比较器 1 的匹配能力。
0b11010：指令地址、指令地址限制、数据地址、数据地址限制和有值数据地址

位 26:25 保留，必须保持复位值。

位 24 **MATCHED**：比较器匹配 (Comparator match)
指示自上次读取寄存器以来是否发生了比较器匹配。
0：不匹配
1：发生了匹配

位 23:12 保留，必须保持复位值。

位 11:10 **DATAVSIZE[1:0]**：数据值大小 (data value size)
定义通过数据值和数据地址比较器监视的对象的大小。
0x0：1 字节
0x1：2 字节
0x2：4 字节
0x3：保留

位 9:6 保留，必须保持复位值。

位 5:4 **ACTION[1:0]**：匹配时的操作 (action on match)
0x0：仅触发
0x1：生成调试事件
0x2：对于周期计数器、指令地址、数据地址、数据值或链接数据值比较器，生成数据跟踪匹配数据包。对于有值数据地址比较器，生成数据跟踪数据值数据包。
0x3：对于数据地址限制比较器，生成数据跟踪数据地址数据包。对于周期计数器、指令地址限制或数据地址比较器，生成数据跟踪 PC 值数据包。对于有值数据地址比较器，生成数据跟踪 PC 值数据包和数据跟踪数据值数据包。

位 3:0 **MATCH[3:0]**：匹配类型 (match type)
控制比较器 1 生成的匹配的类型。
有关该位域的可能值，请参见*[4]*。

DWT 功能寄存器 2 (DWT_FUNCTR2)

DWT function register 2

偏移地址：0x048

复位值：0x5000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ID[4:0]					Res.	Res.	MATCHED	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r	r	r	r			r								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DATAVSIZE[1:0]		Res.	Res.	Res.	Res.	ACTION[1:0]		MATCH[3:0]			
				rw	rw					rw	rw	rw	rw	rw	rw

位 31:27 **ID[4:0]**：能力标识 (capability identification)
标识比较器 2 的匹配能力。
0b01010：指令地址、数据地址和有值数据地址

位 26:25 保留，必须保持复位值。

位 24 **MATCHED**：比较器匹配 (comparator match)
指示自上次读取寄存器以来是否发生了比较器匹配。
0：不匹配
1：发生了匹配

位 23:12 保留，必须保持复位值。

位 11:10 **DATAVSIZE[1:0]**：数据值大小 (Data value size)：
定义通过数据值和数据地址比较器监视的对象的大小。
0x0：1 字节
0x1：2 字节
0x2：4 字节
0x3：保留

位 9:6 保留，必须保持复位值。

位 5:4 **ACTION[1:0]**：匹配时的操作 (action on match)
0x0：仅触发
0x1：生成调试事件
0x2：对于周期计数器、指令地址、数据地址、数据值或链接数据值比较器，生成数据跟踪匹配数据包。对于有值数据地址比较器，生成数据跟踪数据值数据包。
0x3：对于数据地址限制比较器，生成数据跟踪数据地址数据包。对于周期计数器、指令地址限制或数据地址比较器，生成数据跟踪 PC 值数据包。对于有值数据地址比较器，生成数据跟踪 PC 值数据包和数据跟踪数据值数据包。

位 3:0 **MATCH[3:0]**：匹配类型 (match type)
控制比较器 2 生成的匹配的类型。
有关该位域的可能值，请参见*[4]*

DWT 功能寄存器 3 (DWT_FUNCTR3)

DWT function register 3

偏移地址：0x058

复位值：0xF000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ID[4:0]					Res.	Res.	MATCHED	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r	r	r	r			r								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DATAVSIZE[1:0]		Res.	Res.	Res.	Res.	ACTION[1:0]		MATCH[3:0]			
				rw	rw					rw	rw	rw	rw	rw	rw

位 31:27 **ID[4:0]**：能力标识 (capability identification)
标识比较器 2 的匹配能力。
0b11110：指令地址、指令地址限制、数据地址、数据地址限制、数据值、链接数据值和有值数据地址

位 26:25 保留，必须保持复位值。

位 24 **MATCHED**：比较器匹配 (comparator match)
指示自上次读取寄存器以来是否发生了比较器匹配。
0：不匹配
1：发生了匹配

位 23:12 保留，必须保持复位值。

位 11:10 **DATAVSIZE[1:0]**：数据值大小 (data value size)
定义通过数据值和数据地址比较器监视的对象的大小。
0x0：1 字节
0x1：2 字节
0x2：4 字节
0x3：保留

位 9:6 保留，必须保持复位值。

位 5:4 **ACTION[1:0]**：匹配时的操作 (action on match)
0x0：仅触发
0x1：生成调试事件
0x2：对于周期计数器、指令地址、数据地址、数据值或链接数据值比较器，生成数据跟踪匹配数据包。对于有值数据地址比较器，生成数据跟踪数据值数据包。
0x3：对于数据地址限制比较器，生成数据跟踪数据地址数据包。对于周期计数器、指令地址限制或数据地址比较器，生成数据跟踪 PC 值数据包。对于有值数据地址比较器，生成数据跟踪 PC 值数据包和数据跟踪数据值数据包。

位 3:0 **MATCH[3:0]**：匹配类型 (match type)
控制比较器 2 生成的匹配的类型。
有关该位域的可能值，请参见*[4]*

DWT 器件类型架构寄存器 (DWT_DEVARCHR)

DWT device type architecture register

偏移地址：0xFC8

复位值：0x4770 1A02

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARCHITECT[10:0]											PRESENT	REVISION[3:0]			
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

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

位 31:21 **ARCHITECT[10:0]**：架构 JEP106 代码 (architect JEP106 code)
0x23B：JEP106 连续代码 0x4，JEP106 ID 代码 0x3B。限定 Arm®。

位 20 **PRESENT**：DWT_DEVARCH 寄存器存在 (DWT_DEVARCH register present)
0x1：存在

位 19:16 **REVISION[3:0]**：架构版本 (architecture revision)
0x0：DWT 架构 v2.0

位 15:12 **ARCHVER[3:0]**：架构版本 (architecture version)
0x1：DWT 架构 v2.0

位 11:0 **ARCHPART[11:0]**：架构部分 (architecture part)
0xA02：DWT 架构

DWT 器件类型寄存器 (DWT_DEVTYPER)

DWT device type register

偏移地址：0xFCC

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **SUB[3:0]**：子类型 (sub-type)
0x0：其他

位 3:0 **MAJOR[3:0]**：主类型 (major type)
0x0：其他

DWT CoreSight 外设标识寄存器 4 (DWT_PIDR4)

DWT CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0004

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)
0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)
0x4：Arm® JEDEC 代码

DWT CoreSight 外设标识寄存器 0 (DWT_PIDR0)

DWT CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 0021

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (part number bits [7:0])
0x21：Cortex®-M33 DWT 产品编号

DWT CoreSight 外设标识寄存器 1 (DWT_PIDR1)

DWT CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 00BD

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])
0xB：Arm® JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])
0xD：Cortex®-M33 DWT 产品编号

DWT CoreSight 外设标识寄存器 2 (DWT_PIDR2)

DWT CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 000B

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x0：r0p0

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
0x1：JEDEC 指定的设计人员标识

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x3：Arm® JEDEC 代码

DWT CoreSight 外设标识寄存器 3 (DWT_PIDR3)

DWT CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)
0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)
0x0：无客户修改

DWT CoreSight 组件标识寄存器 0 (DWT_CIDR0)

DWT CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]
0x0D：公用标识值

DWT CoreSight 外设标识寄存器 1 (DWT_CIDR1)

DWT CoreSight peripheral identity register 1

偏移地址：0xFF4

复位值：0x0000 0090

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件标识位 [15:12] - 组件类 (component class)
0x9：调试组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]
0x0：公共标识值

DWT CoreSight 组件标识寄存器 2 (DWT_CIDR2)

DWT CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]

0x05：公用标识值

DWT CoreSight 组件标识寄存器 3 (DWT_CIDR3)

DWT CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位 [31:24]

0xB1：公用标识值

41.6.2 DWT 寄存器映射

DWT 寄存器的地址范围为 0xE000 1000 到 0xE000 1FFC。

表 364. DWT 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**DWT_CTRLR**	NUMCOMP[3:0]				NOTRCPKT	NOEXTTRIG	NOCYCCNT	NOPRFCNT	CYCDISS	CYCEVTENA	FOLDEVTENA	LSUEVTENA	SLEEPEVTENA	EXCEVTENA	CPIEVTENA	EXCTRCENA	Res.	Res.	Res.	PCSAMPLENA	SYNCTAP[1:0]		CYCTAP		POSTINIT[3:0]				POSTPRESET[3:0]				CYCCNTENA
	Reset value	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0	0	0	0	0	0	0	0	0	0	0	0	0
0x004	**DWT_CYCCNTR**	CYCCNT[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x008	**DWT_CPICNTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CPICNT[7:0]							
	Reset value																									X	X	X	X	X	X	X	X
0x00C	**DWT_EXCCNTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EXCCNT[7:0]							
	Reset value																									X	X	X	X	X	X	X	X
0x010	**DWT_SLPCNTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SLEEPCNT[7:0]							
	Reset value																									X	X	X	X	X	X	X	X
0x014	**DWT_LSUCNTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSUCNT[7:0]							
	Reset value																									X	X	X	X	X	X	X	X
0x018	**DWT_FOLDCNTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FOLDCNT[7:0]							
	Reset value																									X	X	X	X	X	X	X	X
0x01C	**DWT_PCSR**	EIASAMPLE[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x020	**DWT_COMP0R**	COMP[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x024	Reserved	Reserved																															
0x028	**DWT_FUNCTR0**	ID[4:0]				Res.	Res.	MATCHED	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATAVSIZE[1:0]		Res.	Res.	Res.	Res.	ACTION[1:0]		MATCH[3:0]			
	Reset value	0	1	0	1	1			0													0	0					0	0	0	0	0	0
0x02C	Reserved	Reserved																															
0x030	**DWT_COMP1R**	COMP[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x034	Reserved	Reserved																															
0x038	**DWT_FUNCTR1**	ID[4:0]				Res.	Res.	MATCHED	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATAVSIZE[1:0]		Res.	Res.	Res.	Res.	ACTION[1:0]		MATCH[3:0]			
	Reset value	1	1	0	1	0			0													0	0					0	0	0	0	0	0
0x03C	Reserved	Reserved																															

表 364. DWT 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x040	**DWT_COMP2R**	COMP[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x044	Reserved	Reserved																															
0x048	**DWT_FUNCTR2**	ID[4:0]					Res.	Res.	MATCHED	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATAVSIZE[1:0]		Res.	Res.	Res.	Res.	ACTION[1:0]		MATCH[3:0]			
	Reset value	0	1	0	1	0			0													0	0					0	0	0	0	0	0
0x04C	Reserved	Reserved																															
0x050	**DWT_COMP3R**	COMP[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x054	Reserved	Reserved																															
0x058	**DWT_FUNCTR3**	ID[4:0]					Res.	Res.	MATCHED	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATAVSIZE[1:0]		Res.	Res.	Res.	Res.	ACTION[1:0]		MATCH[3:0]			
	Reset value	1	1	1	1	0			0													0	0					0	0	0	0	0	0
0x05C to 0xFC4	Reserved	Reserved																															
0xFC8	**DWT_DEVARCHR**	ARCHITECT[10:0]											PRESENT	REVISION[3:0]				ARCHVER[3:0]				ARCHPART[11:0]												
	Reset value	0	1	0	0	0	1	1	1	0	1	1	1	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	1	0
0xFCC	**DWT_DEVTYPER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUB[3:0]				MAJOR[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFD0	**DWT_PIDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
	Reset value																									0	0	0	0	0	1	0	0
0xFD4 to 0xFDC	Reserved	Reserved																															
0xFE0	**DWT_PIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PARTNUM[7:0]							
	Reset value																									0	0	1	0	0	0	0	1
0xFE4	**DWT_PIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
	Reset value																									1	0	1	1	1	1	0	1

表 364. DWT 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFE8	**DWT_PIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC		JEP106ID[6:4]	
	Reset value																									0	0	0	0	1	0	1	1
0xFEC	**DWT_PIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFF0	**DWT_CIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**DWT_CIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLASS[3:0]				PREAMBLE[11:8]			
	Reset value																									1	0	0	1	0	0	0	0
0xFF8	**DWT_CIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1
0xFFC	**DWT_CIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*表 360：处理器 ROM 表*。

41.7 指令跟踪宏单元 (ITM)

ITM 将跟踪信息生成为数据包。三个源可生成数据包。如果多个源同时生成数据包，则 ITM 仲裁数据包输出的顺序。按照优先级降序排列，三个源依次为：

- 软件跟踪

  软件可直接写入任意 32 x 32 位 ITM 激励寄存器以生成数据包。可以对每个端口的权限级别进行编程。当软件写入使能的激励端口时，ITM 将端口标识、写访问的大小和被写入的数据合并成其写入 FIFO 的数据包。ITM 将 FIFO 中的数据包输出到跟踪总线上。读取激励端口寄存器会返回激励寄存器的状态（空或挂起），通过位 0 指示。

- 硬件跟踪

  DWT 生成跟踪数据包以响应数据跟踪事件、PC 采样或性能分析计数器回卷。ITM 将这些数据包输出在跟踪总线上。

- 本地时间戳

  ITM 包含一个 21 位计数器，由（预分频的）处理器时钟提供时钟信号。计数器值以时间戳数据包的形式输出到跟踪总线上。每生成一个时间戳数据包，计数器就复位为零。因此，时间戳指示自从上一个时间戳数据包以来经过的时间。

有关 ITM 及其使用方法的更多信息，请参见*[4]*。

41.7.1 ITM 寄存器

ITM 寄存器的地址范围为 0xE000 0000 到 0xE000 0FFC。

ITM 激励寄存器 x (ITM_STIMRx)

ITM stimulus register x

偏移地址：0x000 + 0x004 * x，（x = 0 到 31）

复位值：0xXXXX XXXX

条件：写操作时

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

位 31:0 **STIMULUS[31:0]**：跟踪输出数据 (trace output data)
写数据以软件事件数据包的形式输出在跟踪总线上。

ITM 激励寄存器 x [复用] (ITM_STIMRx)

ITM stimulus register x

偏移地址：0x000 + 0x004 * x，（x = 0 到 31）

复位值：0xXXXX XXXX

条件：读取时

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

位 31:2 保留，必须保持复位值。

位 1 **DISABLE**：禁止标志 (Disable flag)
0：使能激励端口和 ITM
1：禁止激励端口和 ITM

位 0 **FIFO_READY**：FIFO 就绪指示符 (FIFO ready indicator)
0：激励端口缓冲区已满（或端口被禁止）
1：激励端口可接受新的写数据

ITM 跟踪使能寄存器 (ITM_TER)

ITM trace enable register

偏移地址：0xE00

复位值：0x0000 0000

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

位 31:0 **STIMENA[31:0]**：激励端口使能 (stimulus port enable)

每一个位 x（0 到 31）均使能与 ITM_STIMRx 寄存器关联的激励端口。

0：禁止端口

1：使能端口

ITM 跟踪特权寄存器 (ITM_TPR)

ITM trace privilege register

偏移地址：0xE40

复位值：0x0000 0000

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

位 31:4 保留，必须保持复位值。

位 3:0 **PRIVMASK[3:0]**：禁止非特权访问 ITM 激励端口 (Disable unprivileged access to ITM stimulus ports)

每一位控制八个激励端口。

XXX0：允许非特权访问端口 0 到 7

XXX1：仅允许特权访问端口 0 到 7

XX0X：允许非特权访问端口 8 到 15

XX1X：仅允许特权访问端口 8 到 15

X0XX：允许非特权访问端口 16 到 23

X1XX：仅允许特权访问端口 16 到 23

0XXX：允许非特权访问端口 24 到 31

1XXX：仅允许特权访问端口 24 到 31

ITM 跟踪控制寄存器 (ITM_TCR)

ITM trace control register

偏移地址：0xE80

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUSY	TRACEBUSID[6:0]						
								r	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	TSPRESCALE[1:0]		Res.	Res.	STALLENA	SWOENA	TXENA	SYNCENA	TSENA	ITMENA
						rw	rw			rw	r	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23 **BUSY**：指示 ITM 当前是否正在处理事件
0：不忙
1：忙

位 22:16 **TRACEBUSID[6:0]**：多源跟踪流格式化标识符 (identifier for multi-source trace stream formatting)
如果使用了多源跟踪，则调试器必须将一个非零值写入该位域。
注意：系统中每个跟踪源必须使用不同的标识符。

位 15:10 保留，必须保持复位值。

位 9:8 **TSPRESCALE[1:0]**：局部时间戳预分频器，与跟踪数据包参考时钟一起使用 (local timestamp prescaler, used with the trace packet reference clock)
0x0：无预分频
0x1：4 分频。
0x2：16 分频。
0x3：64 分频。

位 7:6 保留，必须保持复位值。

位 5 **STALLENA**：停止使能 (stall enable)
0：如果 ITM 输出停止，则丢弃硬件源数据包并生成溢出。
1：停止处理器以保证传送数据跟踪数据包。

位 4 **SWOENA**：SWO 使能 (SWO enable)
使能时间戳计数器的异步时钟，只读值。
0：时间戳计数器使用处理器时钟。

位 3 **TXENA**：发送使能 (transmit enable)
使能将硬件数据包从 DWT 单元转发到跟踪端口。
0：禁止
1：使能

位 2 **SYNCENA**：同步数据包传输使能 (synchronization packet transmission enable)
如果调试器将该位置 1，则还必须配置 DWT_CTRLR.SYNCTAP 位域，以获得正确的同步速度。
0：禁止
1：使能

位 1 **TSENA**：局部时间戳生成使能 (local timestamp generation enable)
0：禁止
1：使能

位 0 **ITMENA**：ITM 使能 (ITM enable)
0：禁止
1：使能

ITM 器件类型架构寄存器 (ITM_DEVARCHR)

ITM device type architecture register

偏移地址：0xFBC

复位值：0x4770 1A01

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

位 31:21 **ARCHITECT[10:0]**：架构 JEP106 代码 (architect JEP106 code)
0x23B：JEP106 连续代码 0x4，JEP106 ID 代码 0x3B。限定 Arm®。

位 20 **PRESENT**：存在 DEVARCH 寄存器 (DEVARCH register presence)
0x1：存在

位 19:16 **REVISION[3:0]**：架构版本 (architecture revision)
0x0：ITM 架构 v2.0

位 15:12 **ARCHVER[3:0]**：架构版本 (architecture version)
0x1：ITM 架构 v2.0

位 11:0 **ARCHPART[11:0]**：架构部分 (architecture part)
0xA01：ITM 架构

ITM 器件类型寄存器 (ITM_DEVTYPER)

ITM device type register

偏移地址：0xFCC

复位值：0x0000 0043

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **SUB[3:0]**：子类型 (sub-type)
0x4：与总线相关，激励源自总线活动

位 3:0 **MAJOR[3:0]**：主类型 (major type)
0x3：跟踪源

ITM CoreSight 外设标识寄存器 4 (ITM_PIDR4)

ITM CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0004

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)
0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)
0x4：Arm® JEDEC 代码

ITM CoreSight 外设标识寄存器 0 (ITM_PIDR0)

ITM CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 0021

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (part number bits [7:0])
0x21：ITM 产品编号

ITM CoreSight 外设标识寄存器 1 (ITM_PIDR1)

ITM CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 00BD

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])
0xB：Arm® JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])
0xD：ITM 产品编号

ITM CoreSight 外设标识寄存器 2 (ITM_PIDR2)

ITM CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 000B

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x0：r0p0

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
0x1：JEDEC 指定的设计人员标识

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x3：Arm® JEDEC 代码

ITM CoreSight 外设标识寄存器 3 (ITM_PIDR3)

ITM CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)
0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)
0x0：无客户修改

ITM CoreSight 组件标识寄存器 0 (ITM_CIDR0)

ITM CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]
0x0D：公用标识值

ITM CoreSight 外设标识寄存器 1 (ITM_CIDR1)

ITM CoreSight peripheral identity register 1

偏移地址：0xFF4

复位值：0x0000 00E0

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件标识位 [15:12] - 组件类 (component class)
0xE：跟踪发生器组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]
0x0：公用标识值

ITM CoreSight 组件标识寄存器 2 (ITM_CIDR2)

ITM CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]
0x05：公用标识值

ITM CoreSight 组件标识寄存器 3 (ITM_CIDR3)

ITM CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位 [31:24]
0xB1：公用标识值

41.7.2 ITM 寄存器映射

ITM 寄存器的地址范围为 0xE000 0000 到 0xE000 0FFC。

表 365. ITM 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000 to 0x07C	**ITM_STIMR0 to ITM_STIMR31**	STIMULUS[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x07C to 0xDFC	Reserved	Reserved																															

表 365. ITM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xE00	**ITM_TER**	STIMENA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xE04 to 0xE3C	Reserved	Reserved																															
0xE40	**ITM_TPR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRIVMASK [3:0]			
	Reset value																													0	0	0	0
0xE44 to 0xE7C	Reserved	Reserved																															
0xE80	**ITM_TCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUSY	TRACEBUSID[6:0]							Res.	Res.	Res.	Res.	Res.	Res.	TSPRESCALE [1:0]		Res.	Res.	STALLENA	SWOENA	TXENA	SYNCENA	TSENA	ITMENA
	Reset value									0	0	0	0	0	0	0	0							0	0			0	0	0	0	0	0
0xE84 to 0xFB8	Reserved	Reserved																															
0xFBC	**ITM_DEVARCHR**	ARCHITECT[10:0]											PRESENT	REVISION [3:0]				ARCHVER [3:0]				ARCHPART[3:0]											
	Reset value	0	1	0	0	0	1	1	1	0	1	1	1	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1
0xFC0 to 0xFC8	Reserved	Reserved																															
0xFCC	**ITM_DEVTYPER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUB[3:0]				MAJOR[3:0]			
	Reset value																									0	1	0	0	0	0	1	1
0xFD0	**ITM_PIDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE [3:0]				JEP106CON [3:0]			
	Reset value																									0	0	0	0	0	1	0	0
0xFD4 to 0xFDC	Reserved	Reserved																															
0xFE0	**ITM_PIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PARTNUM[7:0]							
	Reset value																									0	0	1	0	0	0	0	1
0xFE4	**ITM_PIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID [3:0]				PARTNUM [11:8]			
	Reset value																									1	0	1	1	1	1	0	1
0xFE8	**ITM_PIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION [3:0]				JEDEC	JEP106ID [6:4]		
	Reset value																									0	0	0	0	1	0	1	1
0xFEC	**ITM_PIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFF0	**ITM_CIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**ITM_CIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLASS[3:0]				PREAMBLE [11:8]			
	Reset value																									1	1	1	0	0	0	0	0
0xFF8	**ITM_CIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1

表 365. ITM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFFC	ITM_CIDR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*表 360：处理器 ROM 表*。

41.8 断点单元 (BPU)

BPU 允许用户设置硬件断点。它包含八个比较器，用于监视指令获取地址。如果发生匹配，可将指令比较器配置为生成断点指令。

有关断点单元及其使用方法的更多信息，请参见*[4]*。

41.8.1 BPU 寄存器

BPU 寄存器的地址范围为 0xE0002000 到 0xE0002FFC。

BPU 控制寄存器 (BPU_CTRLR)

BPU control register

偏移地址：0x000

复位值：0x1000 0080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REV[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r	r	r												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	NUM_CODE[6:4]			Res.	Res.	Res.	Res.	NUM_CODE[3:0]				Res.	Res.	KEY	ENABLE
	r	r	r					r	r	r	r			rw	rw

位 31:28 **REV[3:0]**：版本号 (revision number)
0x1：BPU 版本 2

位 27:15 保留，必须保持复位值。

位 14:12、7:4 **NUM_CODE[6:0]**：支持的指令地址比较器数量 (number of instruction address comparators supported)
0x08：支持 8 个指令比较器

位 11:8、3:2 保留，必须保持复位值。

位 1 **KEY**：写保护关键字 (Write protect key)
如果该位未置 1，则忽略对 FPB_CTRLR 寄存器的写操作。

位 0 **ENABLE**：FPB 使能 (FPB enable)
0：禁止
1：使能

BPU 比较器 x 寄存器 (BPU_COMPxR)

BPU comparator x register

偏移地址：0x008 + 0x004 * x，（x = 0 到 7）

复位值：0x0000 0000

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

位 31:1 **BPADDR[31:1]**：断点地址 (breakpoint address)

位 0 **BE**：断点使能 (breakpoint enable)

0：禁止

1：使能

BPU 器件类型架构寄存器 (BPU_DEVARCHR)

BPU device type architecture register

偏移地址：0xFBC

复位值：0x4770 1A03

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

位 31:21 **ARCHITECT[10:0]**：架构 JEP106 代码 (architect JEP106 code)

0x23B：JEP106 连续代码 0x4，JEP106 ID 代码 0x3B。限定 Arm®。

位 20 **PRESENT**：DEVARCH 寄存器存在 (DEVARCH register present)

0x1：存在

位 19:16 **REVISION[3:0]**：架构版本 (architecture revision)

0x0：BPU 架构 v2.0

位 15:12 **ARCHVER[3:0]**：架构版本 (architecture version)

0x1：BPU 架构 v2.0

位 11:0 **ARCHPART[11:0]**：架构部分 (architecture part)

0xA03：BPU 架构

BPU 器件类型寄存器 (BPU_DEVTYPER)

BPU device type register

偏移地址：0xFCC

复位值：0x0000 0000

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

位 31:8 保留，必须保持复位值。

位 7:4 **SUB[3:0]**：子类型 (sub-type)
0x0：其他

位 3:0 **MAJOR[3:0]**：主类型 (major type)
0x0：其他

BPU CoreSight 外设标识寄存器 4 (BPU_PIDR4)

BPU CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)
0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)
0x4：Arm® JEDEC 代码

BPU CoreSight 外设标识寄存器 0 (BPU_PIDR0)

BPU CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 0021

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (part number bits [7:0])
0x21：BPU 产品编号

BPU CoreSight 外设标识寄存器 1 (BPU_PIDR1)

BPU CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 00BD

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])
0xB：Arm® JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])
0xD：BPU 产品编号

BPU CoreSight 外设标识寄存器 2 (BPU_PIDR2)

BPU CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 000B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x0：r0p0

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
0x1：JEDEC 指定的设计人员标识

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x3：Arm® JEDEC 代码

BPU CoreSight 外设标识寄存器 3 (BPU_PIDR3)

BPU CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)
0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)
0x0：无客户修改

BPU CoreSight 组件标识寄存器 0 (BPU_CIDR0)

BPU CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]
0x0D：公用标识值

BPU CoreSight 外设标识寄存器 1 (BPU_CIDR1)

BPU CoreSight peripheral identity register 1

偏移地址：0xFF4

复位值：0x0000 0090

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件标识位 [15:12] - 组件类 (component class)
0x9：调试组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]
0x0：公共标识值

BPU CoreSight 组件标识寄存器 2 (BPU_CIDR2)

BPU CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]
0x05：公用标识值

BPU CoreSight 组件标识寄存器 3 (BPU_CIDR3)

BPU CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位 [31:24]
0xB1：公用标识值

41.8.2 BPU 寄存器映射

BPU 寄存器的地址范围为 0xE000 2000 到 0xE000 2FFC。

表 366. BPU 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**BPU_CTRLR**	REV[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NUM_CODE [6:4]			Res.	Res.	Res.	Res.	NUM_CODE [3:0]				Res.	Res.	KEY	ENABLE
	Reset value	0	0	0	1														0	0	0					1	0	0	0			0	0

表 366. BPU 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x004	Reserved	Reserved																															
0x008 to 0x024	**BPU_COMP0R to BPU_COMP7R**	BPADDR[31:1]																															BE
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x028 to 0xFB8	Reserved	Reserved																															
0xFBC	**BPU_DEVARCHR**	ARCHITECT[10:0]											PRESENT	REVISION [3:0]				ARCHVER [3:0]				ARCHPART[11:0]											
	Reset value	0	1	0	0	0	1	1	1	0	1	1	1	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	1	1
0xFC0 to 0xFC8	Reserved	Reserved																															
0xFCC	**BPU_DEVTYPER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUB[3:0]				MAJOR[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFD0	**BPU_PIDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE [3:0]				JEP106CON [3:0]			
	Reset value																									0	0	0	0	0	1	0	0
0xFD4 to 0xFDC	Reserved	Reserved																															
0xFE0	**BPU_PIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PARTNUM[7:0]							
	Reset value																									0	0	1	0	0	0	0	1
0xFE4	**BPU_PIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID [3:0]				PARTNUM [11:8]			
	Reset value																									1	0	1	1	1	1	0	1
0xFE8	**BPU_PIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION [3:0]				JEDEC	JEP106ID [6:4]		
	Reset value																									0	0	0	0	1	0	1	1
0xFEC	**BPU_PIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFF0	**BPU_CIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**BPU_CIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLASS[3:0]				PREAMBLE [11:8]			
	Reset value																									1	0	0	1	0	0	0	0
0xFF8	**BPU_CIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1
0xFFC	**BPU_CIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*表 360：处理器 ROM 表*。

41.9 嵌入式跟踪宏单元(ETM)

ETM 是与 CPU 紧密耦合的 CoreSight™ 组件。ETM 生成跟踪数据包，可用于对 Cortex®-M33 内核的执行进行跟踪。在 STM32H503xx 中，ETM 配置为仅用于指令跟踪。数据访问不在跟踪信息中提供。

ETM 通过处理器跟踪接口从 CPU 接收信息，包括：

- 在同一周期内执行的指令数量
- 程序流的变化
- 当前处理器指令状态
- 由加载和存储指令访问的存储单元的地址
- 传输的类型、方向和大小
- 条件代码信息
- 异常信息
- 等待中断状态信息

有关更多信息，请参见《ARM® CoreSight™ ETM-M33 技术参考手册》*[6]*。

41.9.1 ETM 寄存器

ETM 寄存器的地址范围为 0xE004 1000 到 0xE004 1FFC。

ETM 编程控制寄存器 (ETM_PRGCTLR)

ETM programming control register

偏移地址：0x004

复位值：0x0000 0000

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

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

位 31:1 保留，必须保持复位值。

位 0 **EN**：跟踪单元使能 (trace unit enable)
0：禁止
1：使能

ETM 状态寄存器 (ETM_STATR)

ETM status register

偏移地址：0x00C

复位值：0xXXXX XXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PMSTABLE	IDLE
														r	r

位 31:2 保留，必须保持复位值。

位 1 **PMSTABLE**：稳定状态 (stability status)

指示 ETM-M33 寄存器是否处于稳定状态并可读取。

0：不稳定

1：稳定

位 0 **IDLE**：跟踪单元状态

指示跟踪单元处于非活动状态。

0：未空闲

1：空闲

ETM 配置寄存器 (ETM_CONFIGR)

ETM configuration register

偏移地址：0x010

复位值：0xXXXX XXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	RS	Res.	COND[5:0]						CCI	BB	Res.	Res.	Res.
			rw		rw	rw	rw	rw	rw	rw	rw	rw			

位 31:13 保留，必须保持复位值。

位 12 **RS**：返回堆栈使能 (return stack enable)

0：禁止

1：使能

位 11 保留，必须保持复位值。

位 10:5 **COND[5:0]**：条件指令跟踪 (conditional instruction tracing)
0x0：禁止条件指令跟踪
0x1：跟踪条件加载指令
0x2：跟踪条件存储指令
0x3：跟踪条件加载和存储指令
0x7：跟踪所有条件指令

位 4 **CCI**：在指令跟踪中循环计数 (cycle counting in instruction trace)
0：禁止
1：使能

位 3 **BB**：分支广播模式 (branch broadcast mode)
0：禁止
1：使能

位 2:0 保留，必须保持复位值。

ETM 事件控制 0 寄存器 (ETM_EVENTCTL0R)

ETM event control 0 register

偏移地址：0x020

复位值：0xXXXX XXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYPE1	Res.	Res.	Res.	SEL1[3:0]				TYPE0	Res.	Res.	Res.	SEL0[3:0]			
rw				rw	rw	rw	rw	rw				rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15 **TYPE1**：事件 1 资源类型 (resource type for event1)
0：单选资源
1：布尔组合资源对

位 14:12 保留，必须保持复位值。

位 11:8 **SEL1[3:0]**：基于 TYPE1 的资源编号
根据 TYPE1 的值选择资源编号。
当 TYPE1 = 0 时，从 SEL1[3:0] 定义的 0-15 中选择一个单选资源。
当 TYPE1 = 1 时，选择一个 SEL1[2:0] 定义的布尔组合资源对。

位 7 **TYPE0**：事件 0 的资源类型 (resource type for event0)
0：单选资源
1：布尔组合资源对

位 6:4 保留，必须保持复位值。

位 3:0 **SEL0[3:0]**：基于 TYPE0 的资源编号
根据 TYPE0 的值选择资源编号。
当 TYPE0 = 0 时，从 SEL0[3:0] 定义的 0-15 中选择一个单选资源。
当 TYPE0 = 1 时，选择一个 SEL0[2:0] 定义的布尔组合资源对。

ETM 事件控制 1 寄存器 (ETM_EVENTCTL1R)

ETM event control 1 register

偏移地址：0x024

复位值：0xXXXX XXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	LPOVERRIDE	ATB	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INSTEN[1:0]	
			rw	rw										rw	rw

位 31:13 保留，必须保持复位值。

位 12 **LPOVERRIDE**：低功耗状态行为覆盖 (low power state behavior override)
0：正常低功耗状态行为
1：进入低功耗状态不影响资源和事件跟踪生成。

位 11 **ATB**：ATB 触发使能 (ATB trigger enable)
0：禁止
1：使能

位 10:2 保留，必须保持复位值。

位 1:0 **INSTEN[1:0]**：指令事件生成 (instruction event generation)
使能在指令流中生成事件元素。
0bX0：事件 0 不引起一个事件元素。
0bX1：事件 0 发生时引起一个事件元素。
0b0X: 事件 1 不引起一个事件元素。
0b1X：事件 1 发生时引起一个事件元素。

ETM 停止控制寄存器 (ETM_STALLCTLR)

ETM stall control register

偏移地址：0x02C

复位值：0xXXXX XXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	INSTPRIORITY	Res.	ISTALL	Res.	Res.	Res.	Res.	LEVEL[3:0]			
					rw		rw					rw	rw	rw	rw

位 31:11 保留，必须保持复位值。

位 10 **INSTPRIORITY**：指令跟踪优先级 (instruction trace priority)
如果指令跟踪缓冲区空间小于 LEVEL[3:0]，则优先进行指令跟踪。
0：ETM 不得优先进行指令跟踪。
1：ETM 可以优先进行指令跟踪。

位 9 保留，必须保持复位值。

位 8 **ISTALL**：处理器停止 (processor stalling)
基于指令跟踪缓冲区空间停止处理器 (Stalls processor based on instruction trace buffer space)。
0：ETM 不得停止处理器。
1：ETM 可停止处理器。

位 7:4 保留，必须保持复位值。

位 3:0 **LEVEL[3:0]**：从停止状态变为活动状态的阈值 (Threshold at which stalling becomes active)
该位域提供四个级别。可以改变级别以优化因停止导致的入侵级别，并平衡 FIFO 溢出的风险。
0x0：零入侵，但 FIFO 溢出风险较大
...
0xF：入侵级别最高，但 FIFO 溢出风险较小

ETM 同步周期寄存器 (ETM_SYNCPR)

ETM synchronization period register

偏移地址：0x034

复位值：0x0000 000A

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

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

位 31:5 保留，必须保持复位值。

位 4:0 **PERIOD[4:0]**：同步周期 (synchronization period)
将跟踪同步请求之间的跟踪字节数定义为指令流生成的字节总数。
0xA：1024 字节

ETM 循环计数控制寄存器 (ETM_CCCTLR)

ETM cycle count control register

偏移地址：0x038

复位值：0xXXXX XXXX

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

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

位 31:12 保留，必须保持复位值。

位 11:0 **THRESHOLD[11:0]**：指令跟踪循环计数阈值 (instruction trace cycle count threshold)
设置指令跟踪循环计数的阈值。该阈值表示循环计数跟踪数据包之间的最小间隔。

ETM 跟踪标识寄存器 (ETM_TRACEIDR)

ETM trace identification register

偏移地址：0x040

复位值：0xXXXX XXXX

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

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

位 31:7 保留，必须保持复位值。

位 6:0 **TRACEID[6:0]**：输出到跟踪总线的跟踪标识 (Trace identification to output onto the trace bus)
该位域必须编程为唯一值，以将其与系统中的其他跟踪源区分。
值 0x00 和 0x70-0x7F 保留。

ETM ViewInst 主控制寄存器 (ETM_VICTLR)

ETM ViewInst main control register

偏移地址：0x080

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EXLEVEL_S[3:0]			
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	TRCERR	TRCRESET	SSSTATUS	Res.	EVENT[7:0]							
				rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw

位 31:20 保留，必须保持复位值。

位 19:16 **EXLEVEL_S[3:0]**：安全状态下的异常级别 (exception level in secure state)
控制在安全状态下是否为相应的异常级别使能指令跟踪。
0bXXX0：在安全状态下，对于异常级别 0，不生成指令跟踪
0bXXX1：在安全状态下，对于异常级别 0，生成指令跟踪
0b0XXX：在安全状态下，对于异常级别 3，不生成指令跟踪
0b1XXX：在安全状态下，对于异常级别 3，生成指令跟踪

位 15:12 保留，必须保持复位值。

位 11 **TRCERR**：跟踪系统错误异常 (trace system error exception)
0：仅当对恰好在系统错误异常之前的指令或异常进行跟踪时，才会跟踪系统错误异常。
1：始终跟踪系统错误异常。

位 10 **TRCRESET**：跟踪复位异常 (trace reset exception)
0：仅当对恰好在复位异常之前的指令或异常进行跟踪时，才会跟踪复位异常。
1：始终跟踪复位异常。

位 9 **SSSTATUS**：启动/停止逻辑状态 (start/stop logic status)
0：停止
1：启动

位 8 保留，必须保持复位值。

位 7:0 **EVENT[7:0]**：事件选择器 (event selector)

ETM 计数器重载值寄存器 0 (ETM_CNTRLDVR0)

ETM counter reload value register 0

偏移地址：0x140

复位值：0xXXXX XXXX

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

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

位 31:16 保留，必须保持复位值。

位 15:0 **VALUE[15:0]**：计数器重载值 (counter reload value)
每当发生重载事件时，该值就被加载到计数器中。

ETM 标识寄存器 8 (ETM_IDR8)

ETM identification register 8

偏移地址：0x180

复位值：0x0000 0000

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

位 31:0 **MAXSPEC[31:0]**：最大推测深度 (maximum speculation depth)

指示指令跟踪流的最大推测深度。即在任一时间未提交到跟踪流中的 P0 元素的最大数量。

0x0：最大跟踪推测深度为零。

ETM 标识寄存器 9 (ETM_IDR9)

ETM identification register 9

偏移地址：0x184

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUMP0KEY[31:16]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUMP0KEY[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **NUMP0KEY[31:0]**：使用的 P0 右手键数量 (number of P0 right-hand keys used)

0x0：指令跟踪中未使用 P0 右手键

ETM 标识寄存器 10 (ETM_IDR10)

ETM identification register 10

偏移地址：0x188

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUMP1KEY[31:16]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUMP1KEY[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **NUMP1KEY[31:0]**：使用的 P1 右手键数量 (number of P1 right-hand keys used)，包括正常键和特殊键

0x0：指令跟踪中未使用 P1 右手键

ETM 标识寄存器 11 (ETM_IDR11)

ETM identification register 11

偏移地址：0x18C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUMP1SPC[31:16]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUMP1SPC[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **NUMP1SPC[31:0]**：使用的特殊 P1 右手键数量 (number of special P1 right-hand keys used)

0x0：任何配置中都未使用特殊 P1 右手键

ETM 标识寄存器 12 (ETM_IDR12)

ETM identification register 12

偏移地址：0x190

复位值：0x0000 0001

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

位 31:0 **NUMCONDKEY[31:0]**：使用的条件指令右手键数量 (number of conditional instruction right-hand keys used)，包括正常键和特殊键

0x1：设计了一个条件指令右手键

ETM 标识寄存器 13 (ETM_IDR13)

ETM identification register 13

偏移地址：0x194

复位值：0x0000 0000

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

位 31:0 **NUMCONDSPC[31:0]**：使用的特殊条件指令右手键数量 (number of special conditional instruction right-hand keys used)

0x0：没有设计特殊条件指令右手键

ETM 针对特定设计方案的寄存器 0 (ETM_IMSPECR0)

ETM implementation specific register 0

偏移地址：0x1C0

复位值：0x0000 0000

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

位 31:4 保留，必须保持复位值。

位 3:0 **SUPPORT[3:0]**：针对特定设计方案的扩展支持
0x0：不支持针对特定设计方案的扩展

ETM 标识寄存器 0 (ETM_IDR0)

ETM identification register 0

偏移地址：0x1E0

复位值：0x2800 06E1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	COMMOPT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRCEXDATA	QSUPP[1]
		r												r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QSUPP[0]	Res.	CONDTYPE[1:0]		NUMEVENT[1:0]		RETSTACK	Res.	TRCCCI	TRCCOND	TRCBB	TRCDATA[1:0]		INSTP0[1:0]		Res.
r		r	r	r	r	r		r	r	r	r	r	r	r	

位 31:30 保留，必须保持复位值。

位 29 **COMMOPT**：提交位域的含义 (commit field meaning)
指示某些数据包中提交位域的含义。
1：提交模式 1

位 28:18 保留，必须保持复位值。

位 17 **TRCEXDATA**：跟踪数据传输异常 (trace data transfers for exceptions)
指示支持跟踪数据传输异常和异常返回。
0：未设计

位 16:15 **QSUPP[1:0]**：Q 元素支持 (Q element support)
0：不支持

位 14 保留，必须保持复位值。

位 13:12 **CONDTYPE[1:0]**：条件结果跟踪 (conditional results tracing)
指示如何跟踪条件结果。
0：跟踪单元仅指示条件指令是否通过其条件代码检查

位 11:10 **NUMEVENT[1:0]**：支持的事件数量 (Number of events supported)
0x1：两个事件

位 9 **RETSTACK**：返回堆栈支持 (return stack support)
1：两条目返回堆栈

位 8 保留，必须保持复位值。

位 7 **TRCCCI**：循环计数支持 (cycle counting support)
1：设计了循环计数

位 6 **TRCCOND**：条件指令支持 (conditional instruction support)
1：设计了条件指令跟踪

位 5 **TRCBB**：分支广播支持 (branch broadcast support)
1：设计了分支广播跟踪

位 4:3 **TRCDATA[1:0]**：数据跟踪支持 (data tracing support)
0x0：不支持数据跟踪

位 2:1 **INSTP0[1:0]**：支持将加载和存储指令作为 P0 元素跟踪
0x0：不支持

位 0 保留，必须保持复位值。

ETM 标识寄存器 1 (ETM_IDR1)

ETM identification register 1

偏移地址：0x1E4

复位值：0x4100 F421

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DESIGNER[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r	r	r	r	r	r	r								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	TRCARCHMAJ[3:0]				TRCARCHMIN[3:0]				REVISION[3:0]			
				r	r	r	r	r	r	r	r	r	r	r	r

位 31:24 **DESIGNER[7:0]**：跟踪单元设计人员 (trace unit designer)
0x41：Arm®

位 23:12 保留，必须保持复位值。

位 11:8 **TRCARCHMAJ[3:0]**：主跟踪单元架构版本号 (major trace unit architecture version number)
0x4：ETMv4

位 7:4 **TRCARCHMIN[3:0]**：子跟踪单元架构版本号 (minor trace unit architecture version number)
0x2：次版本 2

位 3:0 **REVISION[3:0]**：设计版本号 (implementation revision number)
0x1：设计版本 1

ETM 标识寄存器 2 (ETM_IDR2)

ETM identification register 2

偏移地址：0x1E8

复位值：0x0000 0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	CCSIZE[3:0]				DVSIZE[4:0]					DASIZE[4:1]			
			r	r	r	r	r	r	r	r	r	r	r	r	r

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

位 31:29 保留，必须保持复位值。

位 28:25 **CCSIZE[3:0]**：循环计数器大小 (cycle counter size)
0x0：12 位

位 24:20 **DVSIZE[4:0]**：数据值大小 (data value size)
0x0：不支持数据值大小

位 19:15 **DASIZE[4:0]**：数据地址大小 (data address size)
0x0：不支持数据地址大小

位 14:10 **VMIDSIZE[4:0]**：虚拟机 ID 大小 (virtual machine ID size)
0x0：未设计虚拟机 ID 跟踪

位 9:5 **CIDSIZE[4:0]**：上下文 ID 大小 (context ID size)
0x0：未设计上下文 ID 跟踪

位 4:0 **IASIZE[4:0]**：指令地址大小 (instruction address size)
0x4：32 位最大地址大小

ETM 标识寄存器 3 (ETM_IDR3)

ETM identification register 3

偏移地址：0x1EC

复位值：0x0F09 0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NOOVERFLOW	NUMPROC[2:0]			SYSSTALL	STALLCTL	SYNCPR	TRCERR	Res.	Res.	Res.	Res.	EXLEVEL_S[3:0]			
r	r	r	r	r	r	r	r					r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	CCITMIN[11:0]											
				r	r	r	r	r	r	r	r	r	r	r	r

位 31 **NOOVERFLOW**：ETM_STALLCTLR.NOOVERFLOW 设计 (ETM_STALLCTLR.NOOVERFLOW implementation)

0：未设计

位 30:28 **NUMPROC[2:0]**：可用于跟踪的处理器数量 (number of processors available for tracing)

0x0：一个处理器

位 27 **SYSSTALL**：系统支持处理器的停止控制 (system support for stall control of the processor)

1：系统支持停止控制

位 26 **STALLCTL**：停止控制支持 (stall control support)

1：ETM_STALLCTLR 已设计

位 25 **SYNCPR**：跟踪同步周期支持 (trace synchronization period support)

1：对于仅限指令跟踪的配置，ETM_SYNCPR 为只读。跟踪同步周期是固定的。

位 24 **TRCERR**：ETM_VICTLR.TRCERR 设计 (ETM_VICTLR.TRCERR implementation)

0x1：已设计

位 23:20 保留，必须保持复位值。

位 19:16 **EXLEVEL_S[3:0]**：特权级别设计 (privilege levels implementation)

0x9：设计了特权级别线程和处理程序

位 15:12 保留，必须保持复位值。

位 11:0 **CCITMIN[11:0]**：可以对 TRCCCCTLR.THRESHOLD 编程设定的最小值 (minimum value that can be programmed to TRCCCCTLR.THRESHOLD)

定义最小循环计数阈值。

0x4：四指令跟踪周期数的最小值

ETM 标识寄存器 4 (ETM_IDR4)

ETM identification register 4

偏移地址：0x1F0

复位值：0x0011 4000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUMVMIDC[3:0]				NUMCIDC[3:0]				NUMSSCC[3:0]				NUMRSPAIR[3:0]			
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUMPC[3:0]				Res.	Res.	Res.	SUPPDAC	NUMDVC[3:0]				NUMACPAIRS[3:0]			
r	r	r	r				r	r	r	r	r	r	r	r	r

位 31:28 **NUMVMIDC[3:0]**：虚拟机 ID (VMID) 比较器数量 (number of virtual machine ID (VMID) comparators)

0x0：未设计 VMID 比较器

位 27:24 **NUMCIDC[3:0]**：上下文 ID 比较器数量 (number of Context ID comparators)

0x0：不支持上下文 ID 比较器

位 23:20 **NUMSSCC[3:0]**：单发比较器控件数量 (number of single-shot comparator controls)

0x1：设计了一个单发比较器控件

位 19:16 **NUMRSPAIR[3:0]**：资源选择对数量 (number of resource selection pairs)

0x1：设计了两个资源选择对

位 15:12 **NUMPC[3:0]**：DWT 的处理器比较器输入的数量 (number of processor comparator inputs for the DWT)

0x4：设计了四个处理器比较器输入

位 11:9 保留，必须保持复位值。

位 8 **SUPPDAC**：数据地址比较 (data address comparison)

0：不支持数据地址比较

位 7:4 **NUMDVC[3:0]**：数据值比较器数量 (number of data value comparators)

0x0：未设计数据值比较器

位 3:0 **NUMACPAIRS[3:0]**：地址比较器对数量 (number of address comparator pairs)

0x0：未设计地址比较器对

ETM 标识寄存器 5 (ETM_IDR5)

ETM identification register 5

偏移地址：0x1F4

复位值：0x90C7 0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REDFUNCNTR	NUMCNTR[2:0]			NUMSEQSTATE[2:0]			Res.	LPOVERRIDE	ATBTRIG	TRACEIDSIZE[5:0]					
r	r	r	r	r	r	r		r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	NUMEXTINSEL[2:0]			NUMEXTIN[8:0]								
				r	r	r	r	r	r	r	r	r	r	r	r

位 31 **REDFUNCNTR**：简化功能计数器 (reduced function counter)
1：计数器 0 设计为简化功能计数器。

位 30:28 **NUMCNTR[2:0]**：计数器数量 (number of counters)
0x1：设计了一个计数器。

位 27:25 **NUMSEQSTATE[2:0]**：定序器状态数量 (number of sequencer states)
0x0：未设计定序器状态。

位 24 保留，必须保持复位值。

位 23 **LPOVERRIDE**：低功耗状态覆盖支持 (low-power state override support)
1：设计了低功耗状态覆盖支持

位 22 **ATBTRIG**：ATB 触发支持 (ATB trigger support)
1：设计了 ATB 触发支持

位 21:16 **TRACEIDSIZE[5:0]**：跟踪标识的位数 (number of bits of trace identification)
0x7：设计了 7 位跟踪标识

位 15:12 保留，必须保持复位值。

位 11:9 **NUMEXTINSEL[2:0]**：外部输入选择器数量 (number of external input selectors)
0x0：未设计外部输入选择器

位 8:0 **NUMEXTIN[8:0]**：外部输入数量 (number of external inputs)
0x004：设计了四个外部输入。

ETM 资源寄存器 2 (ETM_RSCTLR2)

ETM resource register 2

偏移地址：0x208

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PAIRINV	INV	Res.	GROUP[2:0]		
										rw	rw		rw	rw	rw

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

位 31:22 保留，必须保持复位值。

位 21 **PAIRINV**：组合资源对结果反转 (result of a combined pair of resources inversion)
0：未反转
1：已反转

位 20 **INV**：所选资源反转 (selected resources inversion)
0：未反转
1：已反转

位 19 保留，必须保持复位值。

位 18:16 **GROUP[2:0]**：资源组选择 (group of resources selection)
0X0：外部输入选择器（选择 0-3）
0x1：来自处理器 DWT 比较器元件的输入（选择 0-3）
0x2：计数器为零（选择 0）
0X3：单发比较器（选择 0）
其他值：保留

位 15:8 保留，必须保持复位值。

位 7:0 **SELECT[7:0]**：更多资源选择 (more resources selection)
从在 GROUP[2:0] 中选择的组中选择一个或多个资源。

ETM 资源寄存器 3 (ETM_RSCTLR3)

ETM resource register 3

偏移地址：0x20C

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INV	Res.	GROUP[2:0]		
											rw		rw	rw	rw

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

位 31:21 保留，必须保持复位值。

位 20 **INV**：所选资源反转 (selected resources inversion)
0：未反转
1：已反转

位 19 保留，必须保持复位值。

位 18:16 **GROUP[2:0]**：资源组选择 (group of resources selection)
0x0：外部输入选择器（选择 0-3）
0x1：来自处理器 DWT 比较器元件的输入（选择 0-3）
0x2：计数器为零（选择 0）
0X3：单发比较器（选择 0）
其他值：保留

位 15:8 保留，必须保持复位值。

位 7:0 **SELECT[7:0]**：更多资源选择 (more resources selection)
从在 GROUP[2:0] 中选择的组中选择一个或多个资源。

ETM 单发比较器控制寄存器 0 (ETM_SSCCR0)

ETM single-shot comparator control register 0

偏移地址：0x280

复位值：0xXXXX XXXX

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

位 31:25 保留，必须保持复位值。

位 24 **RST**：单发比较器复位 (single-shot comparator reset)
使能在发生比较器匹配时复位单发比较器资源，以使能检测另一个比较器匹配。
1：使能复位

位 23:0 保留，必须保持复位值。

ETM 单发比较器状态寄存器 0 (ETM_SSCSR0)

ETM single-shot comparator status register 0

偏移地址：0x2A0

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATUS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PC	DV	DA	INST
												r	r	r	r

位 31 **STATUS**：单发比较器状态 (single-shot comparator status)
指示是否有任何所选比较器发生了匹配。
0：没有发生匹配
1：至少发生一个匹配

位 30:4 保留，必须保持复位值。

位 3 **PC**：处理器比较器输入灵敏度 (processor comparator input sensitivity)
1：单发比较器对处理器比较器输入灵敏

位 2 **DV**：数据值比较器支持 (data value comparator support)
0：不支持单发数据值比较

位 1 **DA**：数据地址比较器支持 (data address comparator support)
0：不支持单发数据地址比较

位 0 **INST**：指令地址比较器支持 (instruction address comparator support)
0：不支持单发指令地址比较

ETM 单发处理器比较器输入控制寄存器 0 (ETM_SSPCICR0)

ETM single-shot processor comparator input control register 0

偏移地址：0x2C0

复位值：0xXXXX XXXX

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **PC[3:0]**：针对单发控制的处理器比较器输入选择 (processor comparator inputs selection for single-shot control)

0xXXX0：未选择处理器比较器输入 0
0xXXX1：已选择处理器比较器输入 0
0xXX0X：未选择处理器比较器输入 1
0xXX1X：已选择处理器比较器输入 1
0xX0XX：未选择处理器比较器输入 2
0xX1XX：已选择处理器比较器输入 2
0x0XXX：未选择处理器比较器输入 3
0x1XXX：已选择处理器比较器输入 3

ETM 掉电控制寄存器 (ETM_PDCR)

ETM power-down control register

偏移地址：0x310

复位值：0x0000 0000

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

位 31:4 保留，必须保持复位值。

位 3 **PU**：上电请求 (power-up request)
0：未请求上电
1：已请求上电

位 2:0 保留，必须保持复位值。

ETM 掉电状态寄存器 (ETM_PDSR)

ETM power-down status register

偏移地址：0x314

复位值：0x0000 0003

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

位 31:2 保留，必须保持复位值。

位 1 **STICKYPD**：粘滞掉电状态 (sticky power-down state)

0：自上次读取 ETM_PDSR 以来，跟踪寄存器电源尚未移除。

1：自上次读取 ETM_PDSR 后，跟踪寄存器电源已移除。

位 0 **POWER**：ETM 上电状态 (ETM power-up status)

1：ETM 上电

ETM 声明标记设置寄存器 (ETM_CLAIMSETR)

ETM claim tag set register

偏移地址：0xFA0

复位值：0x0000 000F

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **CLAIMSET[3:0]**：声明标记位设置 (claim tag bits setting)

写：

0000：无影响

xxx1：将位 0 置 1。

xx1x：将位 1 置 1。

x1xx：将位 2 置 1。

1xxx：将位 3 置 1。

读：

0xF：表示声明标记使用四个位。

ETM 声明标记清零寄存器 (ETM_CLAIMCLR)

ETM claim tag clear register

偏移地址：0xFA4

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **CLAIMCLR[3:0]**：声明标记位复位 (claim tag bits reset)
写：
0000：无影响
xxx1：将位 0 清零
xx1x：将位 1 清零
x1xx：将位 2 清零
1xxx：将位 3 清零
读：返回声明标记的当前值。

ETM 认证状态寄存器 (ETM_AUTHSTATR)

ETM authentication status register

偏移地址：0xFB8

复位值：0xXXXX XXXX

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

位 31:8 保留，必须保持复位值。

位 7:6 **SNID[1:0]**：安全非侵入式调试的安全等级
0x2：禁止安全非侵入式调试

位 5:4 **SID[1:0]**：安全侵入式调试的安全等级 (security level for secure invasive debug)
0x0：未设计

位 3:2 **NSNID[1:0]**：非安全非侵入式调试的安全等级
0x2：禁止非安全非侵入式调试
0x3：使能非安全非侵入式调试

位 1:0 **NSID[1:0]**：非安全侵入式调试的安全等级 (security level for nonsecure invasive debug)
0x0：未设计

ETM 器件类型架构寄存器 (ETM_DEVARCHR)

ETM device type architecture register

偏移地址：0xFBC

复位值：0x4772 4A13

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARCHITECT[10:0]											PRESENT	REVISION[3:0]			
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

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

位 31:21 **ARCHITECT[10:0]**：架构 JEP106 代码 (architect JEP106 code)
0x23B：JEP106 连续代码 0x4，JEP106 ID 代码 0x3B。限定 Arm®。

位 20 **PRESENT**：存在 DEVARCH 寄存器 (DEVARCH register presence)
0x1：存在

位 19:16 **REVISION[3:0]**：架构版本 (architecture revision)
0x2：ETM 架构 v4.2

位 15:12 **ARCHVER[3:0]**：架构版本 (architecture version)
0x4：ETM 架构 v4.2

位 11:0 **ARCHPART[11:0]**：架构部分 (architecture part)
0xA13：ETM 架构

ETM CoreSight 器件类型寄存器 (ETM_DEVTYPER)

ETM CoreSight device type register

偏移地址：0xFCC

复位值：0x0000 0013

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **SUBTYPE[3:0]**：器件子类型标识符 (device sub-type identifier)
0x1：处理器跟踪

位 3:0 **MAJORTYPE[3:0]**：器件主类型标识符 (device main type identifier)
0x3：跟踪源

ETM CoreSight 外设标识寄存器 4 (ETM_PIDR4)

ETM CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0004

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)
0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)
0x4：Arm® JEDEC 代码

ETM CoreSight 外设标识寄存器 0 (ETM_PIDR0)

ETM CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 0021

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (part number bits [7:0])
0x21：ETM 产品编号

ETM CoreSight 外设标识寄存器 1 (ETM_PIDR1)

ETM CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 00BD

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])
0xB：Arm® JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])
0xD：ETM 产品编号

ETM CoreSight 外设标识寄存器 2 (ETM_PIDR2)

ETM CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 001B

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x1：r0p1

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
0x1：JEDEC 指定的设计人员标识

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x3：Arm® JEDEC 代码

ETM CoreSight 外设标识寄存器 3 (ETM_PIDR3)

ETM CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)
0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)
0x0：无客户修改

ETM CoreSight 组件标识寄存器 0 (ETM_CIDR0)

ETM CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]
0x0D：公用标识值

ETM CoreSight 外设标识寄存器 1 (ETM_CIDR1)

ETM CoreSight peripheral identity register 1

偏移地址：0xFF4

复位值：0x0000 0090

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件标识位 [15:12] - 组件类 (component class)
0x9：跟踪发生器组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]
0x0：公共标识值

ETM CoreSight 组件标识寄存器 2 (ETM_CIDR2)

ETM CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]
0x05：公用标识值

ETM CoreSight 组件标识寄存器 3 (ETM_CIDR3)

ETM CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位 [31:24]
0xB1：公用标识值

41.9.2 ETM 寄存器映射

ETM 寄存器可由调试器在地址范围 0xE0041000 到 0xE0041FFC 内进行访问。

表 367. ETM 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x004	**ETM_PRGCTLR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EN
	Reset value																																0
0x008	Reserved	Reserved																															
0x00C	**ETM_STATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PMSTABLE	IDLE
	Reset value																															X	X
0x010	**ETM_CONFIGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RS	Res.	COND[5:0]						CCI	BB	Res.	Res.	Res.
	Reset value																				X		X	X	X	X	X	X	X	X			
0x014 to 0x01C	Reserved	Reserved																															
0x020	**ETM_ EVENTCTL0R**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TYPE1	Res.	Res.	Res.	SEL1[3:0]				TYPE0	Res.	Res.	Res.	SEL0[3:0]			
	Reset value																	X				X	X	X	X	X				X	X	X	X
0x024	**ETM_ EVENTCTL1R**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPOVERRIDE	ATB	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INSTEN[1:0]	
	Reset value																				X	X										X	X
0x028	Reserved	Reserved																															

表 367. ETM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x02C	**ETM_STALLCTLR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INSTPRIORITY	Res.	ISTALL	Res.	Res.	Res.	Res.	LEVEL[3:0]			
	Reset value																						X		X					X	X	X	X
0x030	Reserved	Reserved																															
0x034	**ETM_SYNCPR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERIOD[4:0]			
	Reset value																												0	1	0	1	0
0x038	**ETM_CCCTLR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	THRESHOLD[11:0]											
	Reset value																					X	X	X	X	X	X	X	X	X	X	X	X
0x03C	Reserved	Reserved																															
0x040	**ETM_TRACEIDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRACEID[6:0]						
	Reset value																										X	X	X	X	X	X	X
0x044 to 0x07C	Reserved	Reserved																															
0x080	**ETM_VICTLR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EXLEVEL_S [3:0]				Res.	Res.	Res.	Res.	TRCERR	TRCRESET	SSSTATUS	Res.	EVENT[7:0]							
	Reset value													X	X	X	X					X	X	X		X	X	X	X	X	X	X	X
0x084 to 0x13C	Reserved	Reserved																															
0x140	**ETM_CNTRLDVR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VALUE[15:0]															
	Reset value																	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x144 to 0x17C	Reserved	Reserved																															
0x180	**ETM_IDR8**	MAXSPEC[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x184	**ETM_IDR9**	NUMP0KEY[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x188	**ETM_IDR10**	NUMP1KEY[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18C	**ETM_IDR11**	NUMP1SPC[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x190	**ETM_IDR12**	NUMCONDKEY[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x194	**ETM_IDR13**	NUMCONDSPC[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x198 to 0x1BC	Reserved	Reserved																															
0x1C0	**ETM_IMSPECR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUPPORT [3:0]			
	Reset value																													0	0	0	0
0x1C4 to 0x1DC	Reserved	Reserved																															

表 367. ETM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1E0	**ETM_IDR0**	Res.	Res.	COMMOPT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRCEXDATA	QSUPP[1:0]		Res.	CONDTYPE [1:0]		NUMEVENT [1:0]		RETSTACK	Res.	TRCCCI	TRCCOND	TRCBB	TRCDATA[1:0]		INSTP0[1:0]		Res.
	Reset value			1												0	0	0		0	0	0	1	1		1	1	1	0	0	0	0	
0x1E4	**ETM_IDR1**	DESIGNER[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRCARCHMAJ [3:0]				TRCARCHMIN [3:0]			REVISION [3:0]				
	Reset value	0	1	0	0	0	0	0	1													0	1	0	0	0	0	1	0	0	0	0	1
0x1E8	**ETM_IDR2**	Res.	Res.	Res.	CCSIZE[3:0]				DVSIZE[4:0]					DASIZE[4:0]					VMIDSIZE[4:0]			CIDSIZE[4:0]				IASIZE[4:0]							
	Reset value				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0x1EC	**ETM_IDR3**	NOOVERFLOW	NUMPROC[2:0]			SYSSTALL	STALLCTL	SYNCPR	TRCERR	Res.	Res.	Res.	Res.	EXLEVEL_S [3:0]				Res.	Res.	Res.	Res.	CCITMIN[11:0]											
	Reset value	0	0	0	0	1	1	1	1					1	0	0	1					0	0	0	0	0	0	0	0	0	1	0	0
0x1F0	**ETM_IDR4**	NUMVMIDC [3:0]				NUMCIDC [3:0]				NUMSSCC [3:0]				NUMRSPAIR [3:0]				NUMPC [3:0]				Res.	Res.	Res.	SUPPDAC	NUMDVC [3:0]				NUMACPAIRS [3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0				0	0	0	0	0	0	0	0	0
0x1F4	**ETM_IDR5**	REDFUNCNTR	NUMCNTR[2:0]			NUMSEQSTATE [2:0]			Res.	LPOVERRIDE	ATBTRIG	TRACEIDSIZE [5:0]						Res.	Res.	Res.	Res.	NUMEXTINSEL [2:0]			NUMEXTIN[8:0]								
	Reset value	1	0	0	1	0	0	0		1	1	0	0	0	1	1	1					0	0	0	0	0	0	0	0	0	1	0	0
0x1F8 to 0x204	Reserved	Reserved																															
0x208	**ETM_RSCTLR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PAIRINV	INV	Res.	GROUP [2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SELECT[7:0]							
	Reset value											X	X		X	X	X									X	X	X	X	X	X	X	X
0x20C	**ETM_RSCTLR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INV	Res.	GROUP [2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SELECT[7:0]							
	Reset value												X		X	X	X									X	X	X	X	X	X	X	X
0x210 to 0x27C	Reserved	Reserved																															
0x280	**ETM_SSCCR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value								X																								
0x284 to 0x29C	Reserved	Reserved																															
0x2A0	**ETM_SSCSR0**	STATUS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PC	DV	DA	INST
	Reset value	X																											X	X	X	X
0x2A4 to 0x2BC	Reserved	Reserved																															

表 367. ETM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2C0	**ETM_SSPCICR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PC[3:0]			
	Reset value																													X	X	X	X
0x2C4 to 0x30C	Reserved	Reserved																															
0x310	**ETM_PDCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PU	Res.	Res.	Res.
	Reset value																													0			
0x314	**ETM_PDSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	STICKYPD	POWER
	Reset value																															1	1
0x318 to 0xF9C	Reserved	Reserved																															
0xFA0	**ETM_CLAIMSETR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLAIMSET [3:0]			
	Reset value																													1	1	1	1
0xFA4	**ETM_CLAIMCLR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLAIMCLR [3:0]			
	Reset value																													0	0	0	0
0xFA8 to 0xFB4	Reserved	Reserved																															
0xFB8	**ETM_AUTHSTATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SNID [1:0]		SID [1:0]		NSNID [1:0]		NSID [1:0]	
	Reset value																									X	X	X	X	X	X	X	X
0xFBC	**ETM_DEVARCHR**	ARCHITECT[10:0]											PRESENT	REVISION [3:0]				ARCHVER [3:0]				ARCHPART[11:0]											
	Reset value	0	1	0	0	0	1	1	1	0	1	1	1	0	0	1	0	0	1	0	0	1	0	1	0	0	0	0	1	0	0	1	1
0xFC0to 0xFC8	Reserved	Reserved																															
0xFCC	**ETM_DEVTYPER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUBTYPE [3:0]				MAJORTYP [3:0]			
	Reset value																									0	0	0	1	0	0	1	1
0xFD0	**ETM_PIDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE [3:0]				JEP106CON [3:0]			
	Reset value																									0	0	0	0	0	1	0	0
0xFD4to 0xFDC	Reserved	Reserved																															
0xFE0	**ETM_PIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PARTNUM[7:0]							
	Reset value																									0	0	1	0	0	0	0	1
0xFE4	**ETM_PIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID [3:0]				PARTNUM [11:8]			
	Reset value																									1	0	1	1	1	1	0	1
0xFE8	**ETM_PIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION [3:0]				JEDEC	JEP106ID [6:4]		
	Reset value																									0	0	0	1	1	0	1	1
0xFEC	**ETM_PIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0

表 367. ETM 寄存器映射和复位值（续）

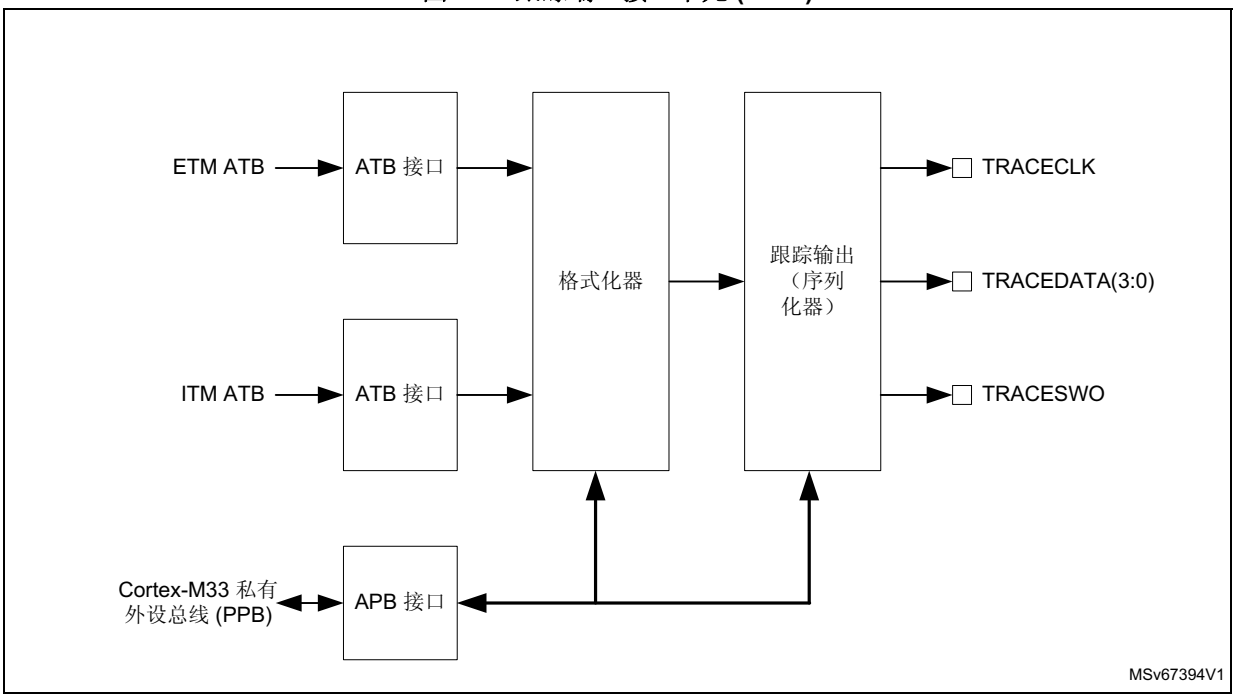
偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFF0	**ETM_CIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**ETM_CIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLASS[3:0]				PREAMBLE [11:8]			
	Reset value																									1	0	0	1	0	0	0	0
0xFF8	**ETM_CIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1
0xFFC	**ETM_CIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*表 360：处理器 ROM 表*。

41.10 跟踪端口接口单元 (TPIU)

TPIU 用来对跟踪流进行格式化处理并将其输出到外部跟踪端口信号上。如下图所示，TPIU 有两个 ATB 从端口，分别用于从 ETM 和 ITM 传入的跟踪数据。跟踪端口为同步并行端口，包括一个时钟输出 TRACECLK 和四个数据输出 TRACEDATA(3:0)。跟踪端口宽度的可编程范围为 1 到 4。使用较小的端口宽度可减少所需测试点/连接器引脚的数量，可将腾出的 IO 用于其他用途，但代价是限制了跟踪端口的带宽，从而限制了可以实时输出的跟踪信息的数量。

图 515. 跟踪端口接口单元 (TPIU)



跟踪数据也可以输出到串行线输出 TRACESWO 上。

有关 Cortex®-M33 中跟踪端口接口的更多信息，请参见《Arm® Cortex®-M33 技术参考手册》*[5]*。

41.10.1 TPIU 寄存器

TPIU 支持的端口大小寄存器 (TPIU_SSPSR)

TPIU supported port size register

偏移地址：0x000

复位值：0x0000 000F

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

位 31:0 **PORTSIZE[31:0]**：跟踪端口大小，从 1 到 32 个引脚 (trace port sizes, from 1 to 32 pins)。
位 n-1 置 1 指示支持端口大小 n。
0x0000 000F：支持端口大小 1 到 4

TPIU 当前端口大小寄存器 (TPIU_CSPSR)

TPIU current port size register

偏移地址：0x004

复位值：0x0000 0001

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

位 31:0 **PORTSIZE[31:0]**：当前跟踪端口大小 (current trace port size)
位 n-1 置 1 指示当前端口大小为 n 个引脚。n 的值必须在支持的端口大小范围 (1-4) 内。只能将一个位置 1，否则可能导致不可预测的行为。
只能在格式化器停止时修改该寄存器。

TPIU 异步时钟预分频器寄存器 (TPIU_ACPR)

TPIU asynchronous clock prescaler register

偏移地址：0x010

复位值：0x0000 0000

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

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

位 31:13 保留，必须保持复位值。

位 12:0 **PRESCALER[12:0]**：异步输出 TRACESWO 的波特率 (baud rate for the asynchronous output, TRACESWO)

波特率为 TRACELKIN 频率除以 (PRESCALER +1)。

TPIU 所选引脚协议寄存器 (TPIU_SPPR)

TPIU selected pin protocol register

偏移地址：0x0F0

复位值：0x0000 0001

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

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

位 31:2 保留，必须保持复位值。

位 1:0 **TXMODE[1:0]**：用于跟踪输出的协议 (protocol used for trace output)

0x0：并行跟踪端口模式
0x1：使用曼彻斯特编码的异步 SWO
0x2：使用 NRZ 编码的异步 SWO
0x3：保留

TPIU 格式化器和刷新状态寄存器 (TPIU_FFSR)

TPIU formatter and flush status register

偏移地址：0x300

复位值：0x0000 0008

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

位 31:4 保留，必须保持复位值。

位 3 **FTNONSTOP**：格式化器停止 (formatter stop)

指示是否可以将格式化器停止。

1：不能停止格式化器。

位 2 **TCPRESENT**：TRACECTL 输出引脚可用性 (TRACECTL output pin availability)

指示可选的 TRACECTL 输出引脚是否可用。

0：该器件中不存在 TRACECTL 引脚。

位 1 **FTSTOPPED**：格式化器停止 (formatter stop)

格式化器收到了停止请求信号和所有跟踪数据，并发送了后置码。ATB 接口上的任何其他跟踪数据都被忽略。

0：格式化器未停止。

位 0 **FLINPROG**：刷新正在进行 (flush in progress)

指示 ATB 从端口上是否有正在进行的刷新。该位反映 AFVALIDS 输出的状态。刷新可由 TPIU_FFCR 寄存器中的刷新控制位启动。

0：无正在进行的刷新

1：正在进行刷新

TPIU 格式化器和刷新控制寄存器 (TPIU_FFCR)

TPIU formatter and flush control register

偏移地址：0x304

复位值：0x0000 0102

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIGIN	Res.	FONMAN	Res.	Res.	Res.	Res.	ENFCONT	Res.
							r		rw					rw	

位 31:9 保留，必须保持复位值。

位 8 **TRIGIN**：通过触发输入的触发指示 (trigger on trigger in)
1：指示当 TRIGIN 输入置为有效时，跟踪流中存在触发。

位 7 保留，必须保持复位值。

位 6 **FONMAN**：手动刷新 (flush on manual)
0：刷新已完成
1：产生刷新。

位 5:2 保留，必须保持复位值。

位 1 **ENFCONT**：使能连续格式化 (continuous formatting enable)
在 SWO 模式下将此位清 0 会忽略格式化器，并且仅输出 ITM/DWT 跟踪，ETM 跟踪将被丢弃。
0：禁止连续格式化
1：使能连续格式化

位 0 保留，必须保持复位值。

TPIU 定期同步计数器寄存器 (TPIU_PSCR)

TPIU periodic synchronization counter register

偏移地址：0x308

复位值：0x0000 0000

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

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

位 31:13 保留，必须保持复位值。

位 12:0 **PSCOUNT[12:0]**：格式化器帧计数器 (formatter frames counter)

使能有效使用不同大小的 TPA，而不会浪费捕获器件的大量存储容量。该计数器包含自最上一个 128 位同步数据包以来的格式化帧数。它是一个 12 位计数器，最大计数器值为 4096。相当于每 65536 个字节（即 4096 个数据包 x 16 字节/数据包）同步一次。默认设置为每 1024 个字节（即每 64 个格式化帧）传输一个同步数据包。如果格式化器配置为连续模式，则在正常运行情况下插入全字和半字同步帧。在这些情况下，计数器值为完全同步数据包间的最大完整帧数。

TPIU 声明标记设置寄存器 (TPIU_CLAIMSETR)

TPIU claim tag set register

偏移地址：0xFA0

复位值：0x0000 000F

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **CLAIMSET[3:0]**：声明标记位设置 (claim tag bits setting)

写：

0000：无影响

xxx1：将位 0 置 1。

xx1x：将位 1 置 1。

x1xx：将位 2 置 1。

1xxx：将位 3 置 1。

读：

0xF：表示声明标记使用四个位。

TPIU 声明标记清零寄存器 (TPIU_CLAIMCLR)

TPIU claim tag clear register

偏移地址：0xFA4

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **CLAIMCLR[3:0]**：声明标记位复位 (claim tag bits reset)

写：

0000：无影响

xxx1：将位 0 清零。

xx1x：将位 1 清零。

x1xx：将位 2 清零。

1xxx：将位 3 清零。

读：返回声明标记的当前值。

TPIU 器件配置寄存器 (TPIU_DEVIDR)

TPIU device configuration register

偏移地址：0xFC8

复位值：0x0000 0CA1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	SWOUARTNRZ	SWOMAN	TCLKDATA	FIFOSIZE[2:0]			CLKRELAT	MAXNUM[4:0]				
				r	r	r	r	r	r	r	r	r	r	r	r

位 31:12 保留，必须保持复位值。

位 11 **SWOUARTNRZ**：串行线输出，NRZ 支持 (Serial-wire output, NRZ support)

0x1：支持

位 10 **SWOMAN**：支持曼彻斯特编码格式的串行线输出 (Serial-wire output, Manchester encoded format, support)

0x1：支持

位 9 **TCLKDATA**：支持跟踪时钟加数据 (trace clock plus data support)

0x0：支持

位 8:6 **FIFOSIZE[2:0]**：以 2 的幂表示的 FIFO 大小 (FIFO size in powers of 2)

0x2：FIFO 大小 = 4 字节

位 5 **CLKRELAT**：ATB 时钟与 TRACECLKIN 关系 (ATB clock and TRACECLKIN relationship)，同步或异步

0x1：异步

位 4:0 **MAXNUM[4:0]**：ATB 输入端口多路复用的数量/类型 (number/type of ATB input port multiplexing)

0x1：两个输入端口

TPIU 器件类型标识寄存器 (TPIU_DEVTYPER)

TPIU device type identifier register

偏移地址：0xFCC

复位值：0x0000 0011

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **SUBTYPE[3:0]**：子分类 (sub-classification)

0x1：跟踪端口组件

位 3:0 **MAJORTYPE[3:0]**：主分类 (major classification)

0x1：跟踪 Sink 组件

TPIU CoreSight 外设标识寄存器 4 (TPIU_PIDR4)

TPIU CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0004

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)

0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)

0x4：Arm® JEDEC 代码

TPIU CoreSight 外设标识寄存器 0 (TPIU_PIDR0)

TPIU CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 0021

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (part number bits [7:0])
0x21：TPIU 产品编号

TPIU CoreSight 外设标识寄存器 1 (TPIU_PIDR1)

TPIU CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 00BD

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])
0xB：Arm® JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])
0xD：TPIU 产品编号

TPIU CoreSight 外设标识寄存器 2 (TPIU_PIDR2)

TPIU CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 000B

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x0：r0p0

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
0x1：JEDEC 指定的设计人员标识

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x3：Arm® JEDEC 代码

TPIU CoreSight 外设标识寄存器 3 (TPIU_PIDR3)

TPIU CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)
0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)
0x0：无客户修改

TPIU CoreSight 组件标识寄存器 0 (TPIU_CIDR0)

TPIU CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]

0x0D：公用标识值

TPIU CoreSight 外设标识寄存器 1 (TPIU_CIDR1)

TPIU CoreSight peripheral identity register 1

偏移地址：0xFF4

复位值：0x0000 0090

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件 ID 位 [15:12]——组件类 (component class)

0x9：CoreSight™ 组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]

0x0：公共标识值

TPIU CoreSight 组件标识寄存器 2 (TPIU_CIDR2)

TPIU CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]
0x05：公用标识值

TPIU CoreSight 组件标识寄存器 3 (TPIU_CIDR3)

TPIU CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位 [31:24]
0xB1：公用标识值

41.10.2 TPIU 寄存器映射

表 368. TPIU 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**TPIU_SSPSR**	PORTSIZE[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
0x004	**TPIU_CSPSR**	PORTSIZE[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x008	Reserved	Reserved																															

表 368. TPIU 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x010	**TPIU_ACPR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESCALER[12:0]												
	Reset value																				0	0	0	0	0	0	0	0	0	0	0	0	0
0x014 to 0x0EC	Reserved	Reserved																															
0x0F0	**TPIU_SPPR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXMODE [1:0]	
	Reset value																															0	1
0x0F4 to 0x2FC	Reserved	Reserved																															
0x300	**TPIU_FFSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FTNONSTOP	TCPRESENT	FTSTOPPED	FLINPROG
	Reset value																													1	0	0	0
0x304	**TPIU_FFCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIGIN	Res.	FONMAN	Res.	Res.	Res.	Res.	ENFCONT	Res.
	Reset value																								1		0					1	
0x308	**TPIU_PSCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSCOUNT[12:0]												
	Reset value																				0	0	0	0	0	0	0	0	0	0	0	0	0
030C to 0xF9C	Reserved	Reserved																															
0xFA0	**TPIU_CLAIMSETR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLAIMSET [3:0]			
	Reset value																													1	1	1	1
0xFA4	**TPIU_CLAIMCLR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLAIMCLR [3:0]			
	Reset value																													0	0	0	0
0FA8 to 0xFC4	Reserved	Reserved																															
0xFC8	**TPIU_DEVIDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWOUARTNRZ	SWOMAN	TCLKDATA	FIFOSIZE[2:0]			CLKRELAT	MaXNUM[4:0]				
	Reset value																					1	1	0	0	1	0	1	0	0	0	0	1
0xFCC	**TPIU_DEVTYPER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUBTYPE [3:0]				MAJORTYPE [3:0]			
	Reset value																									0	0	0	1	0	0	0	1
0xFD0	**TPIU_PIDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE [3:0]				JEP106CON [3:0]			
	Reset value																									0	0	0	0	0	1	0	0
0xFE0	**TPIU_PIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PARTNUM[7:0]							
	Reset value																									0	0	1	0	0	0	0	1
0xFE4	**TPIU_PIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID [3:0]				PARTNUM [11:8]			
	Reset value																									1	0	1	1	1	1	0	1

表 368. TPIU 寄存器映射和复位值

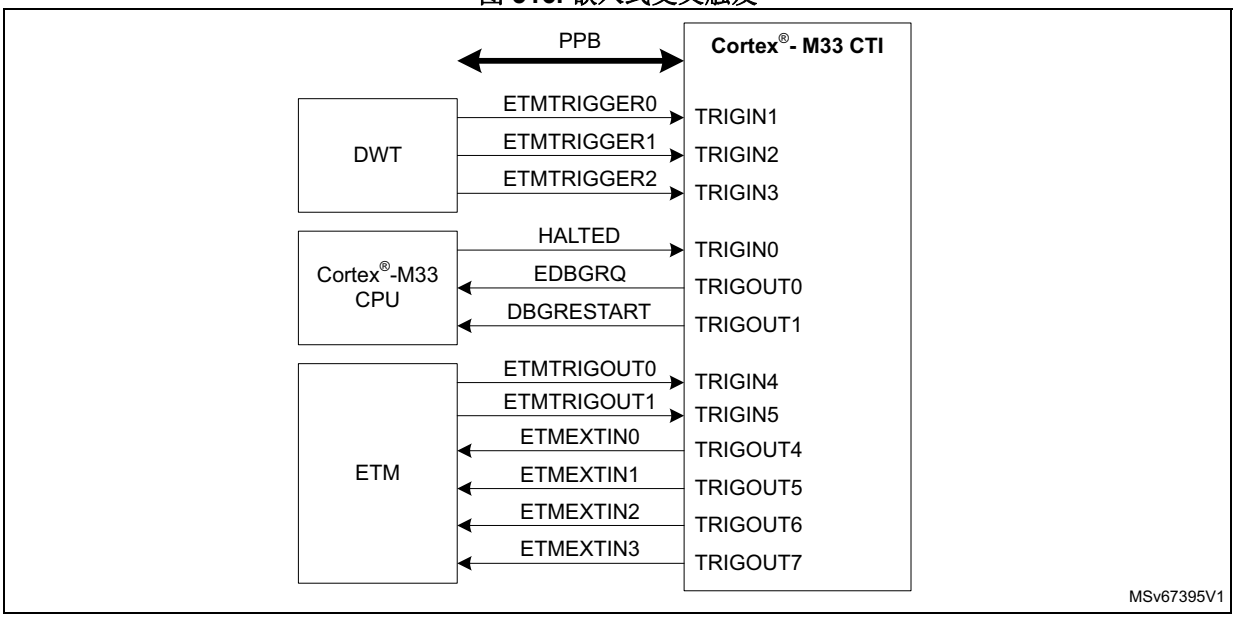
偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFE8	**TPIU_PIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION [3:0]				JEDEC	JEP106ID [6:4]		
	Reset value																									0	0	0	0	1	0	1	1
0xFEC	**TPIU_PIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFF0	**TPIU_CIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**TPIU_CIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLASS[3:0]				PREAMBLE [11:8]			
	Reset value																									1	0	0	1	0	0	0	0
0xFF8	**TPIU_CIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1
0xFFC	**TPIU_CIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*表 359：MCU ROM 表*。

41.11 交叉触发接口 (CTI)

CTI 允许在处理器与 ETM 之间交叉触发（参见下图）。

图 516. 嵌入式交叉触发



CTI 允许来自各种源的事件触发调试和/或跟踪活动。例如，处理器中到达的观察点可以启动或停止代码跟踪，或者跟踪比较器可以停止处理器。

下表列出了 CTI 的触发输入和输出信号。

表 369. CTI 输入

编号	源信号	源组件	备注
0	HALTED	CPU	处理器停止——CPU 处于调试模式
1	ETMTRIGGER0	DWT	DWT 比较器输出 0
2	ETMTRIGGER1	DWT	DWT 比较器输出 1
3	ETMTRIGGER2	DWT	DWT 比较器输出 2
4	ETMTRIGOUT0	ETM	ETM 事件输出 0
5	ETMTRIGOUT1	ETM	ETM 事件输出 1
6	-	-	未使用
7	-	-	未使用

表 370. CTI 输出

编号	源信号	目标组件	备注
0	EDBGRQ	CPU	CPU 停止请求——将 CPU 置于调试模式
1	DBGRESTART	CPU	CPU 重启请求——CPU 退出调试模式
2	ETMEXTIN0	ETM	ETM 事件输入 0
3	ETMEXTIN1	ETM	ETM 事件输入 1
4	ETMEXTIN2	ETM	ETM 事件输入 2
5	ETMEXTIN3	ETM	ETM 事件输入 3
6	-	-	未使用
7	-	-	未使用

有关交叉触发接口 CoreSight™ 组件的更多信息，请参见《ARM® CoreSight™ SoC-400 技术参考手册》*[2]*。

41.11.1 CTI 寄存器

CTI 的寄存器文件基址为 0xE004 2000。

CTI 控制寄存器 (CTI_CONTROLR)

CTI control register

偏移地址：0x000

复位值：0x0000 0000

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

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

位 31:1 保留，必须保持复位值。

位 0 **GLBEN**：全局 CTI 使能 (global CTI enable)。

0：禁止

1：使能

CTI 触发确认寄存器 (CTI_INTACKR)

CTI trigger acknowledge register

偏移地址：0x010

复位值：0xXXXX XXXX

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **INTACK[7:0]**：触发确认 (trigger acknowledge)

每个 CTITRIGOUT 输出都使用一个寄存器位。当向该寄存器的某位写入 1 时，则确认相应的 CTITRIGOUT 输出，使其被清零。

CTI 应用程序触发设置寄存器 (CTI_APPSETR)

CTI application trigger set register

偏移地址：0x014

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **APPSET[3:0]**：通道事件设置 (channel event setting)

读：

XXX0：通道 0 事件处于未激活状态

XXX0：通道 0 事件处于激活状态

XX0X：通道 1 事件处于未激活状态

XX1X：通道 1 事件处于激活状态

X0XX：通道 2 事件处于未激活状态

X1XX：通道 2 事件处于激活状态

0XXX：通道 3 事件处于未激活状态

1XXX：通道 3 事件处于激活状态

写：

XXX0:：无影响

XXX0：设置通道 0 事件。

XX0X：无影响

XX1X：设置通道 1 事件。

X0XX：无影响

X1XX：设置通道 2 事件。

0XXX：无影响

1XXX：设置通道 3 事件。

CTI 应用程序触发清零寄存器 (CTI_APPCLEAR)

CTI application trigger clear register

偏移地址：0x018

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **APPCLEAR[3:0]**：清除通道事件 (channel event clear)

0000：无影响
XXX1：清除通道 0 事件。
XX1X：清除通道 1 事件。
X1XX：清除通道 2 事件。
1XXX：清除通道 3 事件。

CTI 应用程序脉冲寄存器 (CTI_APPPULSER)

CTI application pulse register

偏移地址：0x01C

复位值：0xXXXX XXXX

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

位 31:4 保留，必须保持复位值。

位 3:0 **APPPULSE[3:0]**：脉冲通道事件 (pulse channel event)

该寄存器立即将自己清零。
0000：无影响
XXX1：生成通道 0 脉冲。
XX1X：生成通道 1 脉冲。
X1XX：生成通道 2 脉冲。
1XXX：生成通道 3 脉冲。

CTI 触发输入 x 使能寄存器 (CTI_INENxR)

CTI trigger input x enable register

偏移地址：0x020 + 0x004 * x，（x = 0 到 7）

复位值：0x0000 0000

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

位 31:4 保留，必须保持复位值。

位 3:0 **TRIGINEN[3:0]**：触发输入事件使能 (trigger input event enable)

当 CTITRIGINx（x = 0 到 7）处于激活状态时，使能或禁止四个通道中的每个通道上的交叉触发事件。

0000：触发不生成通道事件。

XXX1：触发 x 生成通道 0 事件。

XX1X：触发 x 生成通道 1 事件。

X1XX：触发 x 生成通道 2 事件。

1XXX：触发 x 生成通道 3 事件。

CTI 触发输出 x 使能寄存器 (CTI_OUTENxR)

CTI trigger output x enable register

偏移地址：0x0A0 + 0x004 * x，（x = 0 到 7）

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **TRIGOUTEN[3:0]**：触发输出事件使能 (trigger output event enable)

对于每个通道，定义该通道上的事件是否生成 CTITRIGOUTx（x = 0 到 7）触发。

0000：通道事件在触发输出时不生成触发。

XXX1：通道 0 事件在触发输出 x 时生成触发。

XX1X：通道 1 事件在触发输出 x 时生成触发。

X1XX：通道 2 事件在触发输出 x 时生成触发。

1XXX：通道 3 事件在触发输出 x 时生成触发。

CTI 触发输入状态寄存器 (CTI_TRGISTSR)

CTI trigger input status register

偏移地址：0x130

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **TRIGINSTATUS[7:0]**：触发输入状态 (trigger input status)

每个 CTITRIGINx 输入都使用一个寄存器位。如果某个位置 1，则表示相应触发输入处于激活状态。如果某位设置为 0，则表示相应触发输入处于未激活状态。

CTI 触发输出状态寄存器 (CTI_TRGOSTSR)

CTI trigger output status register

偏移地址：0x134

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **TRIGOUTSTATUS[7:0]**：触发输出状态 (trigger output status)

每个 CTITRIGOUT 输出都使用一个寄存器位。如果某个位置 1，则表示相应触发输出处于激活状态。如果某个位设置为 0，则表示相应触发输出处于未激活状态。

CTI 通道输入状态寄存器 (CTI_CHINSTSR)

CTI channel input status register

偏移地址：0x138

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **CHINSTATUS[3:0]**：通道输入状态 (channel input status)

每个通道输入都使用一个寄存器位。如果某位设置为 1，则表示相应通道输入处于激活状态。如果某位设置为 0，则表示相应通道输入处于未激活状态。

CTI 通道输出状态寄存器 (CTI_CHOUTSTSR)

CTI channel output status register

偏移地址：0x13C

复位值：0x0000 0000

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **CHOUTSTATUS[3:0]**：通道输出状态 (channel output status)

每个通道输出都使用一个寄存器位。如果某位设置为 1，则表示相应通道输出处于激活状态。如果某位设置为 0，则表示相应通道输出处于未激活状态。

CTI 通道门控寄存器 (CTI_GATER)

CTI channel gate register

偏移地址：0x140

复位值：0x0000 000F

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

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

位 31:4 保留，必须保持复位值。

位 3:0 **GATEEN[3:0]**：通道输出使能 (channel output enable)

对于每个通道，定义了该通道事件能否通过 CTM 传播到其他 CTI。

0000：通道事件不能够传播。

XXX1：通道 0 事件能够传播。

XX1X：通道 1 事件能够传播。

X1XX：通道 2 事件能够传播。

1XXX：通道 3 事件能够传播。

CTI 器件配置寄存器 (CTI_DEVIDR)

CTI device configuration register

偏移地址：0xFC8

复位值：0x0004 0800

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NUMCH[3:0]			
												r	r	r	r

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

位 31:20 保留，必须保持复位值。

位 19:16 **NUMCH[3:0]**：可用 ECT 通道数量 (number of ECT channels available)
0x4：4 个通道

位 15:8 **NUMTRIG[7:0]**：可用 ECT 触发数量 (number of ECT triggers available)
0x8：8 个触发输入和 8 个触发输出

位 7:5 保留，必须保持复位值。

位 4:0 **EXTMUXNUM[4:0]**：触发输入/输出多路复用器数量 (number of trigger input/output multiplexers)
0x0：无

CTI 器件类型标识寄存器 (CTI_DEVTYPER)

CTI device type identifier register

偏移地址：0xFCC

复位值：0x0000 0014

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **SUBTYPE[3:0]**：子分类 (sub-classification)
0x1：交叉触发组件。

位 3:0 **MAJORTYPE[3:0]**：主分类 (major classification)
0x4：表示该组件允许调试器控制 CoreSight™ SoC-400 系统中的其他组件。

CTI CoreSight 外设标识寄存器 4 (CTI_PIDR4)

CTI CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0004

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)
0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)
0x4：Arm® JEDEC 代码

CTI CoreSight 外设标识寄存器 0 (CTI_PIDR0)

CTI CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 0021

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (part number bits [7:0])
0x21：CTI 产品编号

CTI CoreSight 外设标识寄存器 1 (CTI_PIDR1)

CTI CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 00BD

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])
0xB：Arm® JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])
0xD：CTI 产品编号

CTI CoreSight 外设标识寄存器 2 (CTI_PIDR2)

CTI CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 000B

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x0：r0p0

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
0x1：JEDEC 指定的设计人员标识

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x3：Arm® JEDEC 代码

CTI CoreSight 外设标识寄存器 3 (CTI_PIDR3)

CTI CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)
0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)
0x0：无客户修改

CTI CoreSight 组件标识寄存器 0 (CTI_CIDR0)

CTI CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]
0x0D：公用标识值

CTI CoreSight 外设标识寄存器 1 (CTI_CIDR1)

CTI CoreSight peripheral identity register 1

偏移地址：0xFF4

复位值：0x0000 0090

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件标识位 [15:12] - 组件类 (component class)
0x9：CoreSight™ 组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]
0x0：公共标识值

CTI CoreSight 组件标识寄存器 2 (CTI_CIDR2)

CTI CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]
0x05：公用标识值

CTI CoreSight 组件标识寄存器 3 (CTI_CIDR3)

CTI CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位 [31:24]
0xB1：公用标识值

41.11.2 CTI 寄存器映射

表 371. CTI 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**CTI_CONTROLR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GLBEN
	Reset value																																0
0x004 to 0x00C	Reserved	Reserved																															
0x010	**CTI_INTACKR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INTACK[7:0]							
	Reset value																									X	X	X	X	X	X	X	X
0x014	**CTI_APPSETR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	APPSET[3:0]			
	Reset value																													0	0	0	0
0x018	**CTI_APPCLEAR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	APPCLEAR [3:0]			
	Reset value																													0	0	0	0
0x01C	**CTI_APPPULSER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	APPPULSE [3:0]			
	Reset value																													X	X	X	X
0x020 to 0x03C	**CTI_INEN0R to CTI_INEN7R**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIGINEN [3:0]			
	Reset value																													0	0	0	0
0x040 to 0x09C	Reserved	Reserved																															
0x0A0 to 0x0BC	**CTI_OUTEN0R to CTI_OUTEN7R**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIGOUTEN [3:0]			
	Reset value																													0	0	0	0
0x0C0 to 0x12C	Reserved	Reserved																															

表 371. CTI 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x130	**CTI_TRGISTSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIGINSTATUS[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x134	**CTI_TRGOSTSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIGOUTSTATUS[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x138	**CTI_CHINSTSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CHINSTATUS[3:0]			
	Reset value																													0	0	0	0
0x13C	**CTI_CHOUTSTSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CHOUTSTATUS[3:0]			
	Reset value																													0	0	0	0
0x140	**CTI_GATER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GATEEN[3:0]			
	Reset value																													1	1	1	1
0x144 to 0xFC4	Reserved	Reserved																															
0xFC8	**CTI_DEVIDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NUMCH[3:0]				NUMTRIG[7:0]								Res.	Res.	Res.	EXTMUXNUM[4:0]				
	Reset value													0	1	0	0	0	0	0	0	1	0	0	0				0	0	0	0	0
0xFCC	**CTI_DEVTYPER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUB[3:0]				MAJOR[3:0]			
	Reset value																									0	0	0	1	0	1	0	0
0xFD0	**CTI_PIDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
	Reset value																									0	0	0	0	0	1	0	0
0xFD4 to 0xFDC	Reserved	Reserved																															
0xFE0	**CTI_PIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PARTNUM[7:0]							
	Reset value																									0	0	0	1	0	0	0	1
0xFE4	**CTI_PIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
	Reset value																									1	0	1	1	1	1	0	1
0xFE8	**CTI_PIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
	Reset value																									0	0	0	0	1	0	1	1
0xFEC	**CTI_PIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFF0	**CTI_CIDR0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**CTI_CIDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLASS[3:0]				PREAMBLE[11:8]			
	Reset value																									1	0	0	1	0	0	0	0

表 371. CTI 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFF8	**CTI_CIDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1
0xFFC	**CTI_CIDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*表 360：处理器 ROM 表*。

41.12 微控制器调试单元 (DBGMCU)

DBGMCU 组件包含多个寄存器，用来在调试模式下控制电源和时钟行为。它允许调试器（或软件）：

- 在低功耗模式（睡眠、停止或待机）下，保持处理器内核的时钟和电源
- 在低功耗模式下，保持系统调试和跟踪组件的时钟和电源
- 当处理器内核在调试模式下停止时，停止某些外设（SMBUS 超时、看门狗、定时器、RTC）的时钟

41.12.1 器件 ID

DBGMCU 包含一个标识代码寄存器 DBGMCU_IDCODE。该寄存器包含器件的 ID 代码。调试工具可以通过*第 41.5 节：ROM 表*所述的 CoreSight™ 发现流程找到该寄存器。

41.12.2 低功耗模式仿真

当器件进入停止模式（时钟停止）或待机模式（内核电源关闭）时，调试器无法再访问调试访问端口，与器件的连接会断开。为避免这种情况，调试器（或软件）可以将 *DBGMCU 配置寄存器 (DBGMCU_CR)* 中的 DBG_STANDBY 和/或 DBG_STOP 位置 1。这些位置 1 后，可在器件处于相应的低功耗模式时保持处理器的时钟和电源。处理器保持睡眠模式，并以正常方式退出低功耗模式。但外设会继续运行，因此其行为可能与实际低功耗模式下的行为不同。

41.12.3 外设时钟冻结

DBGMCU 外设时钟冻结寄存器允许在调试模式下将某些运行中的外设挂起。下表列出了支持该功能的外设。

表 372. 外设时钟冻结控制位

总线	控制寄存器	外设	说明
APB1L	DBGMCU_APB1LFZR	I3C1	I3C1 SCL 停止超时计数器
		I2C2	I2C2 SMBUS 超时
		I2C1	I2C1 SMBUS 超时
		IWDG	独立看门狗
		WWDG	窗口看门狗
		TIM7	通用定时器 7
		TIM6	通用定时器 6
		TIM3	通用定时器 3
		TIM2	通用定时器 2
APB1H	DGBMCU_APB1HFZR	LPTIM2	低功耗定时器 2
APB2	DBGMCU_APB2FZR	TIM1	通用定时器 1
APB3	DBGMCU_APB3FZR	RTC	实时时钟
		LPTIM1	低功耗定时器 1
		I3C2	I3C2 SCL 停止超时计数器
AHB1	DBGMCU_AHB1FZR	GPDMA2 0 到 7	通用 DMA2 通道 0 到 7
		GPDMA1 0 到 7	通用 DMA1 通道 0 到 7

每个外设单元或 DMA 通道都有一个相应的控制位 DBG_xxx_STOP，其中 xxx 是外设（或 DMA 通道）的缩写。控制位在 DBGMCU_zzzFZR 寄存器中排列，其中 zzz 对应于总线名称（AHB 或 APB）。例如，DBGMCU_APB1LFZR 包含 APB1L 总线上的外设的控制位。

当该控制位置 1 时，如果 CPU 在调试模式下停止 (HALTED = 1)，相应的外设运行将挂起，如下表所示：

表 373. 调试模式下的外设行为

HALTED	DBG_xxx_STOP	外设行为
0	X	继续运行。
1	0	继续运行。
1	1	运行挂起。

41.12.4 DBGMCU 寄存器

DBGMCU 寄存器不通过系统复位而复位，只能通过上电复位而复位。可以通过位于基址 0xE004 4000 的 AHB 访问端口访问调试器，也可以通过位于基址 0x4402 4000 的软件来访问调试器。

DBGMCU 标识代码寄存器 (DBGMCU_IDCODE)

DBGMCU identity code register

偏移地址：0x00

复位值：0xXXXX 6XXX

该寄存器始终可访问。

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

位 31:16 **REV_ID[15:0]**：版本 (revision)
该位域指示器件的版本。
0x1000：版本 A
0x1001：版本 Z
0x1002：版本 Y

位 15:12 保留，必须保持复位值。

位 11:0 **DEV_ID[11:0]**：器件标识 (device identification)
器件 ID 为 0x474：STM32H503

DBGMCU 配置寄存器 (DBGMCU_CR)

DBGMCU configuration register

偏移地址：0x04

复位值：0x0000 0000

身份验证成功后，调试器可以访问该寄存器。在此之前，调试器访问将被忽略。软件始终可以访问该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DCRT
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRACE_MODE[1:0]		TRACE_EN	TRACE_IOEN	Res.	DBG_STANDBY	DBG_STOP	Res.
								rw	rw	rw	rw		rw	rw	

位 31:17 保留，必须保持复位值。

位 16 **DCRT**：调试凭据复位类型 (Debug credentials reset type)
该位用于选择使用哪种复位类型来撤销调试身份验证凭据
0：系统复位
1：电源复位

位 15:8 保留，必须保持复位值。

位 7:6 **TRACE_MODE[1:0]**：跟踪引脚分配 (trace pin assignment)
0x0：分配用于异步模式的跟踪引脚 (TRACESWO)
0x1：分配用于同步模式的跟踪引脚，端口宽度为 1（TRACECK，TRACED0）
0x2：分配用于同步模式的跟踪引脚，端口宽度为 2（TRACECK，TRACED0-1）
0x3：分配用于同步模式的跟踪引脚，端口宽度为 4（TRACECK，TRACED0-3）

位 5 **TRACE_EN**：跟踪端口和时钟使能 (trace port and clock enable)。
该位使能跟踪端口时钟 TRACECK。
0：禁止
1：使能

位 4 **TRACE_IOEN**：跟踪引脚使能 (trace pin enable)
0：禁止——未分配跟踪引脚
1：使能——根据 TRACE_MODE 位域的值分配跟踪引脚

位 3 保留，必须保持复位值。

位 2 **DBG_STANDBY**：允许在待机模式下进行调试 (Allows debug in standby mode)
0：正常工作
在待机模式下，将禁止所有时钟，内核将自动掉电。
1：禁止自动停止时钟/掉电
所有活动的时钟和振荡器将在待机模式下继续运行，并将保持内核电源，因此支持全面的调试功能。在退出待机模式时，将执行系统复位。

位 1 **DBG_STOP**：允许在停止模式下进行调试 (Allows debug in stop mode)

0：正常工作

在停止模式下，将自动禁止所有时钟。

1：禁止自动停止时钟

所有活动的时钟和振荡器将在停止模式下继续运行，因此支持全面的调试功能。从停止模式退出时，时钟设置被设为停止模式退出状态。

位 0 保留，必须保持复位值。

DBGMCU APB1L 外设冻结寄存器 (DBGMCU_APB1LFZR)

DBGMCU APB1L peripheral freeze register

偏移地址：0x08

复位值：0x0000 0000

身份验证成功后，调试器可以访问该寄存器。在此之前，调试器访问将被忽略。软件始终可以访问该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_I3C1_STOP	DBG_I2C2_STOP	DBG_I2C1_STOP	Res.	Res.	Res.	Res.	Res.
								rw	rw	rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DBG_IWDG_STOP	DBG_WWDG_STOP	Res.	Res.	Res.	Res.	Res.	DBG_TIM7_STOP	DBG_TIM6_STOP	Res.	Res.	DBG_TIM3_STOP	DBG_TIM2_STOP
			rw	rw						rw	rw			rw	rw

位 31:24 保留，必须保持复位值。

位 23 **DBG_I3C1_STOP**：I3C1 SCL 停止计数器在调试模式下停止 (I3C1 SCL stall counter stop in debug)

0：正常工作。当 CPU 处于调试模式时，I3C1 SCL 停止超时计数器继续运行。

1：在调试模式下停止。当 CPU 处于调试模式时，I3C1 SCL 停止超时计数器被冻结。

位 22 **DBG_I2C2_STOP**：在调试模式下，I2C2 SMBUS 超时停止 (I2C2 SMBUS timeout stop in debug)

0：正常工作。当 CPU 处于调试模式时，I2C2 SMBUS 超时继续运行。

1：在调试模式下停止。当 CPU 处于调试模式时，I2C2 SMBUS 超时被冻结。

位 21 **DBG_I2C1_STOP**：在调试模式下，I2C1 SMBUS 超时停止 (I2C1 SMBUS timeout stop in debug)

0：正常工作。当 CPU 处于调试模式时，I2C1 SMBUS 超时继续运行。

1：在调试模式下停止。当 CPU 处于调试模式时，I2C1 SMBUS 超时被冻结。

位 20:13 保留，必须保持复位值。

位 12 **DBG_IWDG_STOP**：IWDG 在调试模式下停止

0：正常工作。当 CPU 处于调试模式时，IWDG 继续运行。

1：在调试模式下停止。当 CPU 处于调试模式时，IWDG 被冻结。

位 11 **DBG_WWDG_STOP**：WWDG 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，WWDG 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，WWDG 被冻结。

位 10:6 保留，必须保持复位值。

位 5 **DBG_TIM7_STOP**：TIM7 在调试模式下停止 (TIM7 stop in debug)
0：正常工作。当 CPU 处于调试模式时，TIM7 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，TIM7 被冻结。

位 4 **DBG_TIM6_STOP**：TIM6 在调试模式下停止 (TIM6 stop in debug)
0：正常工作。当 CPU 处于调试模式时，TIM6 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，TIM6 被冻结。

位 3:2 保留，必须保持复位值。

位 1 **DBG_TIM3_STOP**：TIM3 在调试模式下停止 (TIM3 stop in debug)
0：正常工作。当 CPU 处于调试模式时，TIM3 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，TIM3 被冻结。

位 0 **DBG_TIM2_STOP**：TIM2 在调试模式下停止 (TIM2 stop in debug)
0：正常工作。当 CPU 处于调试模式时，TIM2 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，TIM2 被冻结。

DBGMCU APB1H 外设冻结寄存器 (DBGMCU_APB1HFZR)

DBGMCU APB1H peripheral freeze register

偏移地址：0x0C

复位值：0x0000 0000

身份验证成功后，调试器可以访问该寄存器。在此之前，调试器访问将被忽略。软件始终可以访问该寄存器。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_LPTIM2_STOP	Res.	Res.	Res.	Res.	Res.
										rw					

位 31:6 保留，必须保持复位值。

位 5 **DBG_LPTIM2_STOP**：LPTIM2 在调试模式下停止 (LPTIM2 stop in debug)
0：正常工作。当 CPU 处于调试模式时，LPTIM2 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，LPTIM2 被冻结。

位 4:0 保留，必须保持复位值。

DBGMCU APB2 外设冻结寄存器 (DBGMCU_APB2FZR)

DBGMCU APB2 peripheral freeze register

偏移地址：0x10

复位值：0x0000 0000

身份验证成功后，调试器可以访问该寄存器。在此之前，调试器访问将被忽略。软件始终可以访问该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	DBG_TIM1_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
				rw											

位 31:12 保留，必须保持复位值。

位 11 **DBG_TIM1_STOP**：TIM1 在调试模式下停止 (TIM1 stop in debug)

0：正常工作。当 CPU 处于调试模式时，TIM1 继续运行。

1：在调试模式下停止。当 CPU 处于调试模式时，TIM1 被冻结。

位 10:0 保留，必须保持复位值。

DBGMCU APB3 外设冻结寄存器 (DBGMCU_APB3FZR)

DBGMCU APB3 peripheral freeze register

偏移地址：0x14

复位值：0x0000 0000

身份验证成功后，调试器可以访问该寄存器。在此之前，调试器访问将被忽略。软件始终可以访问该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	DBG_RTC_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_LPTIM1_STOP	Res.
	rw													rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DBG_I3C2_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
			rw												

位 31 保留，必须保持复位值。

位 30 **DBG_RTC_STOP**：RTC 在调试模式下停止 (RTC stop in debug)

0：正常工作。当 CPU 处于调试模式时，RTC 继续运行。

1：在调试模式下停止。当 CPU 处于调试模式时，RTC 被冻结。

位 29:28 保留，必须保持复位值。

位 17 **DBG_LPTIM1_STOP**：LPTIM1 在调试模式下停止 (LPTIM1 stop in debug)

0：正常工作。当 CPU 处于调试模式时，LPTIM1 继续运行。

1：在调试模式下停止。当 CPU 处于调试模式时，LPTIM1 被冻结。

位 16:13 保留，必须保持复位值。

位 12 **DBG_I3C2_STOP**：I3C2 SCL 停止计数器在调试模式下停止 (I3C2 SCL stall counter stop in debug)

0：正常工作。当 CPU 处于调试模式时，I3C2 SCL 停止超时计数器继续运行。

1：在调试模式下停止。当 CPU 处于调试模式时，I3C2 SCL 停止超时计数器被冻结。

位 11:0 保留，必须保持复位值。

DBGMCU AHB1 外设冻结寄存器 (DBGMCU_AHB1FZR)

DBGMCU AHB1 peripheral freeze register

偏移地址：0x20

复位值：0x0000 0000

身份验证成功后，调试器可以访问该寄存器。在此之前，调试器访问将被忽略。软件始终可以访问该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_GPDMA2_7_STOP	DBG_GPDMA2_6_STOP	DBG_GPDMA2_5_STOP	DBG_GPDMA2_4_STOP	DBG_GPDMA2_3_STOP	DBG_GPDMA2_2_STOP	DBG_GPDMA2_1_STOP	DBG_GPDMA2_0_STOP
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_GPDMA1_7_STOP	DBG_GPDMA1_6_STOP	DBG_GPDMA1_5_STOP	DBG_GPDMA1_4_STOP	DBG_GPDMA1_3_STOP	DBG_GPDMA1_2_STOP	DBG_GPDMA1_1_STOP	DBG_GPDMA1_0_STOP
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23 **DBG_GPDMA2_7_STOP**：GPDMA2 通道 7 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA2 通道 7 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA2 通道 7 被冻结。

位 22 **DBG_GPDMA2_6_STOP**：GPDMA2 通道 6 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA2 通道 6 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA2 通道 6 被冻结。

位 21 **DBG_GPDMA2_5_STOP**：GPDMA2 通道 5 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA2 通道 5 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA2 通道 5 被冻结。

位 20 **DBG_GPDMA2_4_STOP**：GPDMA2 通道 4 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA2 通道 4 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA2 通道 4 被冻结。

位 19 **DBG_GPDMA2_3_STOP**：GPDMA2 通道 3 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA2 通道 3 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA2 通道 3 被冻结。

位 18 **DBG_GPDMA2_2_STOP**：GPDMA2 通道 2 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA2 通道 2 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA2 通道 2 被冻结。

位 17 **DBG_GPDMA2_1_STOP**：GPDMA2 通道 1 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA2 通道 1 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA2 通道 1 被冻结。

位 16 **DBG_GPDMA2_0_STOP**：GPDMA2 通道 0 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA2 通道 0 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA2 通道 0 被冻结。

位 15:8 保留，必须保持复位值。

位 7 **DBG_GPDMA1_7_STOP**：GPDMA1 通道 7 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA1 通道 7 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA1 通道 7 被冻结。

位 6 **DBG_GPDMA1_6_STOP**：GPDMA1 通道 6 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA1 通道 6 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA1 通道 6 被冻结。

位 5 **DBG_GPDMA1_5_STOP**：GPDMA1 通道 5 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA1 通道 5 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA1 通道 5 被冻结。

位 4 **DBG_GPDMA1_4_STOP**：GPDMA1 通道 4 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA1 通道 4 继续运行
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA1 通道 4 被冻结。

位 3 **DBG_GPDMA1_3_STOP**：GPDMA1 通道 3 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA1 通道 3 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA1 通道 3 被冻结。

位 2 **DBG_GPDMA1_2_STOP**：GPDMA1 通道 2 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA1 通道 2 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA1 通道 2 被冻结。

位 1 **DBG_GPDMA1_1_STOP**：GPDMA1 通道 1 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA1 通道 1 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA1 通道 1 被冻结。

位 0 **DBG_GPDMA1_0_STOP**：GPDMA1 通道 0 在调试模式下停止
0：正常工作。当 CPU 处于调试模式时，GPDMA1 通道 0 继续运行。
1：在调试模式下停止。当 CPU 处于调试模式时，GPDMA1 通道 0 被冻结。

DBGMCU 状态寄存器 (DBGMCU_SR)

DBGMCU status register

偏移地址：0xFC

复位值：0x0001 XX03

该寄存器始终可访问。

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

位 31:16 **AP_ENABLED[15:0]**：位 n 标识访问端口 AP n 是打开（可通过调试端口访问）还是锁定（阻止对 AP 的调试访问）
位 n = 0：APn 已锁定
位 n = 1：Apn 已使能

位 15:0 **AP_PRESENT[15:0]**：位 n 标识器件中是否有访问端口 AP n
位 n = 0：没有 APn
位 n = 1：有 APn

DBGMCU 调试身份验证邮箱主机寄存器 (DBGMCU_DBG_AUTH_HOST)

DBGMCU debug authentication mailbox host register

偏移地址：0x100

复位值：0xXXXX XXXX

该寄存器在由 CPU 访问时为只读，写入无效。

该寄存器可由外部调试器写入和读取。

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

位 31:0 **MESSAGE[31:0]**：调试主机与器件之间的邮箱消息 (Debug host to device mailbox message)。
在调试身份验证期间，调试主机通过该寄存器与器件通信。

DBGMCU 调试身份验证邮箱器件寄存器 (DBGMCU_DBG_AUTH_DEVICE)

DBGMCU debug authentication mailbox device register

偏移地址：0x104

复位值：0xXXXX XXXX

该寄存器在通过调试端口进行访问时为只读，写入无效。

该寄存器可由 CPU 进行写入和读取。

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

位 31:0 **MESSAGE[31:0]**：器件与调试主机间的邮箱消息 (Device to debug host mailbox message)。
在调试身份验证期间，器件通过该寄存器与调试主机通信。

DBGMCU 调试身份验证邮箱确认寄存器 (DBGMCU_DBG_AUTH_ACK)

DBGMCU debug authentication mailbox acknowledge register

偏移地址：0x108

复位值：0x0000 0000

该寄存器始终可访问。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DEV_ACK	HOST_ACK
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **DEV_ACK**：器件对主机确认 (Device to host acknowledge)。
主机将该位置 1 以指示它在 DBGMCU_DBG_AUTH_HOST 寄存器中放入一条消息。器件在读取消息后将其复位

位 0 **HOST_ACK**：主机对器件确认 (Host to device acknowledge)。
器件将该位置 1 以指示它在 DBGMCU_DBG_AUTH_DEVICE 寄存器中放入一条消息。主机在读取消息后应将其复位

DBGMCU CoreSight 外设标识寄存器 4 (DBGMCU_PIDR4)

DBGMCU CoreSight peripheral identity register 4

偏移地址：0xFD0

复位值：0x0000 0000

该寄存器始终可访问。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIZE[3:0]				JEP106CON[3:0]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **SIZE[3:0]**：寄存器文件大小 (register file size)
0x0：寄存器文件占用单个 4 KB 区域

位 3:0 **JEP106CON[3:0]**：JEP106 连续代码 (JEP106 continuation code)
0x0：意法半导体 JEDEC 代码

DBGMCU CoreSight 外设标识寄存器 0 (DBGMCU_PIDR0)

DBGMCU CoreSight peripheral identity register 0

偏移地址：0xFE0

复位值：0x0000 0000

该寄存器始终可访问。

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PARTNUM[7:0]**：产品编号位 [7:0] (part number bits [7:0])
0x00：DBGMCU 产品编号

DBGMCU CoreSight 外设标识寄存器 1 (DBGMCU_PIDR1)

DBGMCU CoreSight peripheral identity register 1

偏移地址：0xFE4

复位值：0x0000 0000

该寄存器始终可访问。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JEP106ID[3:0]				PARTNUM[11:8]			
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **JEP106ID[3:0]**：JEP106 标识代码位 [3:0] (JEP106 identity code bits [3:0])
0x0：意法半导体 JEDEC 代码

位 3:0 **PARTNUM[11:8]**：产品编号位 [11:8] (part number bits [11:8])
0x0：DBGMCU 产品编号

DBGMCU CoreSight 外设标识寄存器 2 (DBGMCU_PIDR2)

DBGMCU CoreSight peripheral identity register 2

偏移地址：0xFE8

复位值：0x0000 000A

该寄存器始终可访问。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REVISION[3:0]				JEDEC	JEP106ID[6:4]		
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:4 **REVISION[3:0]**：组件版本号 (component revision number)
0x0：r0p0

位 3 **JEDEC**：JEDEC 分配的值 (JEDEC assigned value)
0x1：JEDEC 指定的设计人员标识

位 2:0 **JEP106ID[6:4]**：JEP106 标识代码位 [6:4] (JEP106 identity code bits [6:4])
0x2：意法半导体 JEDEC 代码

DBGMCU CoreSight 外设标识寄存器 3 (DBGMCU_PIDR3)

DBGMCU CoreSight peripheral identity register 3

偏移地址：0xFEC

复位值：0x0000 0000

该寄存器始终可访问。

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **REVAND[3:0]**：金属修复版本 (Metal fix version)
0x0：无金属修复

位 3:0 **CMOD[3:0]**：客户修改 (customer modified)
0x0：无客户修改

DBGMCU CoreSight 组件标识寄存器 0 (DBGMCU_CIDR0)

DBGMCU CoreSight component identity register 0

偏移地址：0xFF0

复位值：0x0000 000D

该寄存器始终可访问。

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

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

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[7:0]**：组件标识位 [7:0]
0x0D：公用标识值

DBGMCU CoreSight 组件标识寄存器 1 (DBGMCU_CIDR1)

DBGMCU CoreSight component identity register 1

偏移地址：0xFF4

复位值：0x0000 00F0

该寄存器始终可访问。

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

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

位 31:8 保留，必须保持复位值。

位 7:4 **CLASS[3:0]**：组件标识位 [15:12] - 组件类 (component class)
0xF：非 CoreSight 组件

位 3:0 **PREAMBLE[11:8]**：组件标识位 [11:8]
0x0：公共标识值

DBGMCU CoreSight 组件标识寄存器 2 (DBGMCU_CIDR2)

DBGMCU CoreSight component identity register 2

偏移地址：0xFF8

复位值：0x0000 0005

该寄存器始终可访问。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[19:12]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[19:12]**：组件标识位 [23:16]
0x05：公用标识值

DBGMCU CoreSight 组件标识寄存器 3 (DBGMCU_CIDR3)

DBGMCU CoreSight component identity register 3

偏移地址：0xFFC

复位值：0x0000 00B1

该寄存器始终可访问。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREAMBLE[27:20]							
								r	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PREAMBLE[27:20]**：组件标识位 [31:24]
0xB1：公用标识值

41.12.5 DBGMCU 寄存器映射

表 374. DBGMCU 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**DBGMCU_IDCODE**	REV_ID[15:0]																Res.	Res.	Res.	Res.	DEV_ID[11:0]											
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X					X	X	X	X	X	X	X	X	X	X	X	X
0x004	**DBGMCU_CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DCRT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRACE_MODE [1:0]		TRACE_EN	TRACE_IOEN	Res.	DBG_STANDBY	DBG_STOP	Res.
	Reset value																0									0	0	0	0		0	0	
0x008	**DBGMCU_ APB1LFZR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_I3C1_STOP	DBG_I2C2_STOP	DBG_I2C1_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_IWDG_STOP	DBG_WWDG_STOP	Res.	Res.	Res.	Res.	Res.	DBG_TIM7_STOP	DBG_TIM6_STOP	Res.	Res.	DBG_TIM3_STOP	DBG_TIM2_STOP
	Reset value									0	0	0									0	0						0	0			0	0
0x0C0	**DBGMCU_ APB1HFZR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_LPTIM2_STOP	Res.	Res.	Res.	Res.	Res.
	Reset value																											0					
0x010	**DBGMCU_ APB2FZR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_TIM1_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																					0											
0x04	**DBGMCU_ APB3FZR**	Res.	DBG_RTC_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_LPTIM1_STOP	Res.	Res.	Res.	Res.	DBG_I3C2_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value		0													0					0												
0x018 to 0x01C	Reserved	Reserved																															
0x020	**DBGMCU_ AHB1FZR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_GPDMA2_7_STOP	DBG_GPDMA2_6_STOP	DBG_GPDMA2_5_STOP	DBG_GPDMA2_4_STOP	DBG_GPDMA2_3_STOP	DBG_GPDMA2_2_STOP	DBG_GPDMA2_1_STOP	DBG_GPDMA2_0_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_GPDMA1_7_STOP	DBG_GPDMA1_6_STOP	DBG_GPDMA1_5_STOP	DBG_GPDMA1_4_STOP	DBG_GPDMA1_3_STOP	DBG_GPDMA1_2_STOP	DBG_GPDMA1_1_STOP	DBG_GPDMA1_0_STOP
	Reset value									0	0	0	0	0	0	0	0									0	0	0	0	0	0	0	0
0x024 to 0x0F8	Reserved	Reserved																															

表 374. DBGMCU 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0FC	**DBGMCU_SR**	AP_ENABLED[15:0]																AP_PRESENT[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
0x100	**DBGMCU_DBG_AUTH_HOST**	MESSAGE[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x104	**DBGMCU_DBG_AUTH_DEVICE**	MESSAGE[31:0]																															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x108	**DBGMCU_DBG_AUTH_ACK**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DEV_ACK	HOST_ACK
	Reset value																															0	0
0x10C to 0xFBC	Reserved	Reserved																															
0xFD0	**DBGMCU_PIDR4**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SIZE [3:0]				JEP106CON [3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFD4 to 0xFDC	Reserved	Reserved																															
0xFE0	**DBGMCU_PIDR0**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PARTNUM[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0xFE4	**DBGMCU_PIDR1**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	JEP106ID [3:0]				PARTNUM [11:8]			
	Reset value																									0	0	0	0	0	0	0	0
0xFE8	**DBGMCU_PIDR2**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	REVISION [3:0]				JEDEC	JEP106ID [6:4]		
	Reset value																									0	0	0	0	1	0	1	0
0xFEC	**DBGMCU_PIDR3**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	REVAND[3:0]				CMOD[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0xFF0	**DBGMCU_CIDR0**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PREAMBLE[7:0]							
	Reset value																									0	0	0	0	1	1	0	1
0xFF4	**DBGMCU_CIDR1**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CLASS[3:0]				PREAMBLE [11:8]			
	Reset value																									1	1	1	1	0	0	0	0
0xFF8	**DBGMCU_CIDR2**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PREAMBLE[19:12]							
	Reset value																									0	0	0	0	0	1	0	1
0xFFC	**DBGMCU_CIDR3**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PREAMBLE[27:20]							
	Reset value																									1	0	1	1	0	0	0	1

有关寄存器边界地址的信息，请参见*第 2.2 节*。

41.13 参考

1. IHI 0031C (ID080813)——《Arm® 调试接口架构规范 ADIv5.0 到 ADIv5.2》，发布版本 C，2013 年 8 月 8 日
2. DDI 0480F (ID100313)——《Arm® CoreSight™ SoC-400 r3p2 技术参考手册》，发布版本 G，2015 年 3 月 16 日
3. DDI 0314H——《Arm® CoreSight™ 组件技术参考手册》，发布版本 H，2009 年 7 月 10 日
4. DDI 0553A (ID092917)——《Arm®v8-M 架构参考手册》，发布版本 A.f，2017 年 9 月 29 日
5. 100230_0002_00_en——《Arm® Cortex®-M33 处理器 r0p2 技术参考手册》，发布版本 0002-00，2017 年 5 月 10 日
6. 100232_0001_00_en——《Arm® CoreSight™ ETM-M33 r0p1 技术参考手册》，发布版本 0001-00，2017 年 2 月 3 日

42 器件电子签名

器件电子签名存储在 Flash 模块的系统存储区中，可以使用调试接口或 CPU 对其进行读取。它包含出厂前编程的标识和校准数据，这些数据允许用户固件或其他外部器件与器件的特性自动匹配。

42.1 唯一器件 ID 寄存器（96 位）

Unique device ID register

唯一器件标识符最适合：

- 用作序列号（例如 USB 字符串序列号或其他终端应用程序）
- 在对内部 Flash 进行编程前将唯一 ID 与软件加密原语和协议结合使用时用作安全密钥的一部分以提高 Flash 中代码的安全性
- 激活安全启动进程等

96 位的唯一器件标识符提供了一个对于任何器件和任何上下文都唯一的参考号码。用户不能更改这些位。

基址：0x08FF F800

偏移地址：0x00

只读 = 0xXXXX XXXX，其中 X 是出厂前编程的

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

位 31:0 **UID[31:0]**：晶圆上的 X 和 Y 坐标

偏移地址：0x04

只读 = 0xXXXX XXXX，其中 X 是出厂前编程的

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

位 31:8 **UID[63:40]**：LOT_NUM[23:0]
批号（ASCII 编码）

位 7:0 **UID[39:32]**：WAF_NUM[7:0]
晶圆编号（8 位无符号数）

偏移地址：0x08

只读 = 0xXXXX XXXX，其中 X 是出厂前编程的

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UID[95:80]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UID[79:64]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **UID[95:64]**：LOT_NUM[55:24]
批号（ASCII 编码）

42.2 Flash 大小数据寄存器

Flash size data register

基址：0x08FF F80C

偏移地址：0x00

只读 = 0xXXXX，其中 X 是出厂前编程的

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

位 15:0 **FLASH_SIZE[15:0]**：Flash 大小 (Flash memory size)
此处的位域指示以 KB 表示的器件 Flash 大小。
例如，0x080 对应于 128 KB。

42.3 封装数据寄存器

Package data register

基址：0x08FF F80E

偏移地址：0x00

只读 = 0xXXXX，其中 X 是出厂前编程的

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

位 15:5 保留，必须保持复位值。

位 4:0 **PKG[4:0]**：封装类型

00000：LQFP64
00101：LQFP48
01001：UFQFPN32
01111：WLCSP25
10000：UFQFPN48
其他值：保留

43 重要安全说明

意法半导体集团公司 (ST) 高度重视产品安全性，因此，本文档中所述的 ST 产品可能已通过各种安全认证机构的认证和/或可能已实施我们内部的安全措施（如本文所述）。但是，任何级别的安全认证和/或 ST 内部的安全措施都不能保证 ST 产品能够抵御所有形式的攻击。因此，ST 的每位客户都需自行负责确定 ST 产品中提供的安全级别是否满足自身的需求，而且除了 ST 产品本身的安全级别之外，还需要考虑在最终产品或应用中与其他组件和/或软件结合使用时的情况。特别要注意以下几点：

- ST 产品可能已通过一家或多家安全认证机构的认证，例如平台安全架构 (www.psacertified.org) 和/或物联网平台安全评估标准 (www.trustcb.com)。有关此处引用的 ST 产品是否已获得此类安全认证以及认证级别和当前认证状态的详细信息，请访问相关认证标准网站或访问 *www.st.com* 上的相关产品页以获取最新信息。由于 ST 产品的安全认证状态和/或级别可能会不时发生变化，客户应根据需要随时检查 ST 产品的安全认证状态/级别。如果未显示 ST 产品已获得特定安全标准认证，客户不应主观认为 ST 产品已通过认证。
- 认证机构有权评估、授予和撤销与 ST 产品相关的安全认证。因此，这些认证机构独立负责授予或撤销 ST 产品的安全认证，针对认证机构对任何 ST 产品所做的误判、评估、测试或其他活动，ST 不承担任何责任。
- 行业通用的加密算法（如 AES、DES 或 MD5）以及其他可与 ST 产品结合使用的开放标准技术均基于非 ST 开发的标准。对于此类加密算法或开放技术所存在的任何缺陷，或者可能甚至已经开发出的能够绕过、解密或破解此类算法或技术的任何方法，ST 不承担任何责任。
- 尽管 ST 产品已完成可靠的安全测试，但任何级别的认证都无法绝对保证抵御所有形式的攻击，其中包括未经测试的高级攻击、新的或未识别形式的攻击、在规范或预期用途之外使用 ST 产品时存在的任何形式攻击，或者与客户用来创建最终产品或应用的其他组件或软件结合使用时存在的攻击等。对于此类攻击，ST 不承担任何责任。因此，无论集成的安全功能和/或 ST 可能提供的任何信息或支持如何，每位客户都需独自负责确定测试的攻击抵御级别是否满足自身的需求，而且除了 ST 产品本身的安全级别之外，还需要考虑将 ST 产品整合到最终产品或应用中的情况。
- ST 产品（包括任何硬件、软件、文档等）的所有安全功能，包括但不限于 ST 添加的任何增强安全功能，均按“原样”提供。因此，除非适用的书面和签署的合同条款另有明确规定，否则在适用法律允许的范围内，ST 不提供任何明示或暗示的担保，包括但不限于对适销性或特定用途适用性的暗示担保。

44 版本历史

表 375. 文档版本历史

日期	版本	变更
2023 年 3 月 1 日	1	首次发布。
2023 年 3 月 16 日	2	标题更新。

索引

A

ADC_AWD2CR 565
ADC_AWD3CR 566
ADC_CALFACT 567
ADC_CCR 568
ADC_CFGR 549
ADC_CFGR2 553
ADC_CR 546
ADC_DIFSEL 566
ADC_DR 562
ADC_HWCFGR0 569
ADC_IER 544
ADC_IPDR 571
ADC_ISR 543
ADC_JDRy 565
ADC_JSQR 562
ADC_OFRy 564
ADC_OR 567
ADC_SIDR 571
ADC_SMPR1 555
ADC_SMPR2 555
ADC_SQR1 558
ADC_SQR2 559
ADC_SQR3 560
ADC_SQR4 561
ADC_TR1 556
ADC_TR2 557
ADC_TR3 558
ADC_VERR 570
AP0_CSWR 1570
AP1_CSWR 1571
APx_BASER 1574
APx_BDnR 1573
APx_DRWR 1573
APx_IDR 1574
APx_TAR 1572

B

BPU_CIDR0 1630
BPU_CIDR1 1630
BPU_CIDR2 1631
BPU_CIDR3 1631
BPU_COMPxR 1626
BPU_CTRLR 1625
BPU_DEVARCHR 1626
BPU_DEVTYPER 1627
BPU_PIDR0 1628
BPU_PIDR1 1628
BPU_PIDR2 1629
BPU_PIDR3 1629
BPU_PIDR4 1627

C

C1ROM_CIDR3 . . . . . .1583, 1589, 1595
COMP_CFGR1 636
COMP_CFGR2 638
COMP_ICFR 635
COMP_SR 635
CPUROM_CIDR0 1593
CPUROM_CIDR1 1593
CPUROM_CIDR2 1594
CPUROM_CIDR3 1594
CPUROM_MEMTYPER 1590
CPUROM_PIDR0 1591
CPUROM_PIDR1 1591
CPUROM_PIDR2 1592
CPUROM_PIDR3 1592
CPUROM_PIDR4 1590
CRC_CR 482
CRC_DR 481
CRC_IDR 481
CRC_INIT 483
CRC_POL 483
CRS_CFGR 324
CRS_CR 323
CRS_ICR 327
CRS_ISR 325
CTI_APPCLEAR 1680
CTI_APPPULSER 1681
CTI_APPSETR 1680
CTI_CHINSTSR 1683
CTI_CHOUTSTSR 1684
CTI_CIDR0 1688
CTI_CIDR1 1689
CTI_CIDR2 1689
CTI_CIDR3 1690
CTI_CONTROLR 1679
CTI_DEVIDR 1685
CTI_DEVTYPER 1685
CTI_GATER 1684
CTI_INENxR 1681
CTI_INTACKR 1679
CTI_OUTENxR 1682
CTI_PIDR0 1686
CTI_PIDR1 1687

CTI_PIDR2 1687
CTI_PIDR3 1688
CTI_PIDR4 1686
CTI_TRGISTSR 1682
CTI_TRGOSTSR 1683

D

DAC_CCR 621
DAC_CR 610
DAC_DHR12L1 614
DAC_DHR12L2 616
DAC_DHR12LD 617
DAC_DHR12R1 614
DAC_DHR12R2 615
DAC_DHR12RD 617
DAC_DHR8R1 615
DAC_DHR8R2 616
DAC_DHR8RD 618
DAC_DOR1 618
DAC_DOR2 619
DAC_MCR 621
DAC_SHHR 624
DAC_SHRR 625
DAC_SHSR1 623
DAC_SHSR2 623
DAC_SR 619
DAC_SWTRGR 613
DBGMCU_AHB1FZR 1699
DBGMCU_APB1HFZR 1697
DBGMCU_APB1LFZR 1696
DBGMCU_APB2FZR 1698
DBGMCU_APB3FZR 1698
DBGMCU_CIDR0 1706
DBGMCU_CIDR1 1706
DBGMCU_CIDR2 1707
DBGMCU_CIDR3 1707
DBGMCU_CR 1695
DBGMCU_DBG_AUTH_ACK 1703
DBGMCU_DBG_AUTH_DEVICE 1702
DBGMCU_DBG_AUTH_HOST 1702
DBGMCU_IDCODE 1694
DBGMCU_PIDR0 1704
DBGMCU_PIDR1 1704
DBGMCU_PIDR2 1705
DBGMCU_PIDR3 1705
DBGMCU_PIDR4 1703
DBGMCU_SR 1701
DP_ABORTR 1562
DP_CTRLSTATR 1563
DP_DLCR 1564
DP_DLPIDR 1565
DP_DPIDR 1561
DP_RDBUFFR 1567
DP_RESENDR 1566
DP_SELECTR 1566
DP_TARGETIDR 1565
DTS_CFGR1 583
DTS_DR 586
DTS_ICIFR 589
DTS_ITENR 588
DTS_ITR1 585
DTS_OR 590
DTS_RAMPVALR 585
DTS_SR 586
DTS_T0VALR1 584
DWT_CIDR0 1610
DWT_CIDR1 1610
DWT_CIDR2 1611
DWT_CIDR3 1611
DWT_COMPxR 1602
DWT_CPICNTR 1599
DWT_CTRLR 1597
DWT_CYCCNTR 1599
DWT_DEVARCHR 1606
DWT_DEVTYPER 1607
DWT_EXCCNTR 1600
DWT_FOLDCNTR 1601
DWT_FUNCTR0 1602
DWT_FUNCTR1 1603
DWT_FUNCTR2 1604
DWT_FUNCTR3 1605
DWT_LSUCNTR 1601
DWT_PCSR 1602
DWT_PIDR0 1608
DWT_PIDR1 1608
DWT_PIDR2 1609
DWT_PIDR3 1609
DWT_PIDR4 1607
DWT_SLPCNTR 1600

E

ETM_AUTHSTATR 1654
ETM_CCCTLR 1638
ETM_CIDR0 1658
ETM_CIDR1 1659
ETM_CIDR2 1659
ETM_CIDR3 1660
ETM_CLAIMCLR 1653
ETM_CLAIMSETR 1653
ETM_CNTRLDVR0 1639
ETM_CONFIGR 1634
ETM_DEVARCHR 1655

ETM_DEVTYPER 1655
ETM_EVENTCTL0R 1635
ETM_EVENTCTL1R 1636
ETM_IDR0 1643
ETM_IDR1 1644
ETM_IDR10 1641
ETM_IDR11 1641
ETM_IDR12 1642
ETM_IDR13 1642
ETM_IDR2 1645
ETM_IDR3 1646
ETM_IDR4 1647
ETM_IDR5 1648
ETM_IDR8 1640
ETM_IDR9 1640
ETM_IMSPECR0 1643
ETM_PDCR 1652
ETM_PDSR 1652
ETM_PIDR0 1656
ETM_PIDR1 1657
ETM_PIDR2 1657
ETM_PIDR3 1658
ETM_PIDR4 1656
ETM_PRGCTLR 1633
ETM_RSCTLR2 1649
ETM_RSCTLR3 1649
ETM_SSCCR0 1650
ETM_SSCSR0 1651
ETM_SSPCICR0 1651
ETM_STALLCTLR 1636
ETM_STATR 1634
ETM_SYNCPR 1637
ETM_TRACEIDR 1638
ETM_VICTLR 1638
EXTI_EMR1 472
EXTI_EMR2 474
EXTI_EXTICR1 467
EXTI_EXTICR2 468
EXTI_EXTICR3 469
EXTI_EXTICR4 470
EXTI_FPR1 461
EXTI_FPR2 466
EXTI_FTSR1 459
EXTI_FTSR2 463
EXTI_IMR1 471
EXTI_IMR2 473
EXTI_PRIVCFGR1 462
EXTI_PRIVCFGR2 466
EXTI_RPR1 460
EXTI_RPR2 465
EXTI_RTSR1 459
EXTI_RTSR2 463
EXTI_SWIER1 460
EXTI_SWIER2 464

F

FDCAN_CCCR 1483
FDCAN_CKDIV 1508
FDCAN_CREL 1480
FDCAN_DBTP 1481
FDCAN_ECR 1488
FDCAN_ENDN 1480
FDCAN_HPMS 1499
FDCAN_IE 1494
FDCAN_ILE 1497
FDCAN_ILS 1496
FDCAN_IR 1491
FDCAN_NBTP 1485
FDCAN_PSR 1489
FDCAN_RWD 1483
FDCAN_RXF0A 1501
FDCAN_RXF0S 1500
FDCAN_RXF1A 1502
FDCAN_RXF1S 1501
FDCAN_RXGFC 1497
FDCAN_TDCR 1491
FDCAN_TEST 1482
FDCAN_TOCC 1487
FDCAN_TOCV 1488
FDCAN_TSCC 1486
FDCAN_TSCV 1486
FDCAN_TXBAR 1504
FDCAN_TXBC 1502
FDCAN_TXBCF 1506
FDCAN_TXBCIE 1507
FDCAN_TXBCR 1505
FDCAN_TXBRP 1503
FDCAN_TXBTIE 1506
FDCAN_TXBTO 1505
FDCAN_TXEFA 1508
FDCAN_TXEFS 1507
FDCAN_TXFQS 1503
FDCAN_XIDAM 1499
FLASH_ACR 154
FLASH_ECCCORR 174
FLASH_ECCDETR 175
FLASH_ECCDR 176
FLASH_HDP1R_CUR 173
FLASH_HDP1R_PRG 174
FLASH_HDP2R_CUR 178
FLASH_HDP2R_PRG 178
FLASH_HDPEXTR 164
FLASH_NSBOOTR_CUR 170

FLASH_NSBOOTR_PRG 170
FLASH_NSCCR 162
FLASH_NSCR 160
FLASH_NSKEYR 155
FLASH_NSSR 158
FLASH_OPSR 157
FLASH_OPTCR 157
FLASH_OPTKEYR 156
FLASH_OPTSR_CUR 164
FLASH_OPTSR_PRG 166
FLASH_OPTSR2_CUR 168
FLASH_OPTSR2_PRG 169
FLASH_OTPBLR_CUR 171
FLASH_OTPBLR_PRG 171
FLASH_PRIVBB1R1 172
FLASH_PRIVBB2R1 176
FLASH_PRIVCFGR 163
FLASH_WRP1R_CUR 172
FLASH_WRP1R_PRG 173
FLASH_WRP2R_CUR 177
FLASH_WRP2R_PRG 177

G

GPDMA_CxBR1 435-436
GPDMA_CxBR2 441
GPDMA_CxCR 427
GPDMA_CxDAR 439
GPDMA_CxFCR 424
GPDMA_CxLBAR 424
GPDMA_CxLLR 442-443
GPDMA_CxSAR 438
GPDMA_CxSR 425
GPDMA_CxTR1 429
GPDMA_CxTR2 432
GPDMA_CxTR3 440
GPDMA_MISR 423
GPDMA_PRIVCFGR 422
GPIOx_AFRH 342
GPIOx_AFRL 342
GPIOx_BRR 343
GPIOx_BSRR 340
GPIOx_HSLVR 344
GPIOx_IDR 339
GPIOx_LCKR 341
GPIOx_MODER 337
GPIOx_ODR 340
GPIOx_OSPEEDR 338
GPIOx_OTYPER 338
GPIOx_PUPDR 339
GTZC1_MPCBBz_PRIVCFGRx 113
GTZC1_TZSC_MPCWM4ACFGR 110
GTZC1_TZSC_MPCWM4AR 111
GTZC1_TZSC_PRIVCFGR1 106
GTZC1_TZSC_PRIVCFGR2 108
GTZC1_TZSC_PRIVCFGR3 109

H

HASH_CR 680
HASH_CSRx 687
HASH_DIN 682
HASH_HRAx 684
HASH_HRx 684-685
HASH_IMR 685
HASH_SR 686
HASH_STR 683

I

I2C_CR1 1131
I2C_CR2 1134
I2C_ICR 1142
I2C_ISR 1140
I2C_OAR1 1136
I2C_OAR2 1137
I2C_PECR 1143
I2C_RXDR 1144
I2C_TIMEOUTR 1139
I2C_TIMINGR 1138
I2C_TXDR 1144
I3C_BCR 1242
I3C_CEVR 1230
I3C_CFGR 1210
I3C_CR 1206, 1208
I3C_CRCAPR 1245
I3C_DCR 1243
I3C_DEVR0 1232
I3C_DEVRx 1234
I3C_EPIDR 1248
I3C_EVR 1223
I3C_GETCAPR 1244
I3C_GETMXDSR 1246
I3C_IBIDR 1218
I3C_IER 1227
I3C_MAXRLR 1236
I3C_MAXWLR 1237
I3C_RDR 1214
I3C_RDWR 1215
I3C_RMR 1223
I3C_SER 1221
I3C_SR 1220
I3C_TDR 1216
I3C_TDWR 1217
I3C_TGTTDR 1219

I3C_TIMINGR0 1237
I3C_TIMINGR1 1238
I3C_TIMINGR2 1240
ICACHE_CR 191
ICACHE_FCR 193
ICACHE_HMONR 194
ICACHE_IER 192
ICACHE_MMONR 194
ICACHE_SR 192
ITM_CIDR0 1622
ITM_CIDR1 1622
ITM_CIDR2 1623
ITM_CIDR3 1623
ITM_DEVARCHR 1618
ITM_DEVTYPER 1619
ITM_PIDR0 1620
ITM_PIDR1 1620
ITM_PIDR2 1621
ITM_PIDR3 1621
ITM_PIDR4 1619
ITM_STIMRx 1615
ITM_TCR 1617
ITM_TER 1616
ITM_TPR 1616
IWDG_EWCR 996
IWDG_KR 993
IWDG_PR 993
IWDG_RLR 994
IWDG_SR 995
IWDG_WINR 996

L

LPTIM_ARR 977
LPTIM_CCMR1 980
LPTIM_CCR1 977
LPTIM_CCR2 982
LPTIM_CFGR 973
LPTIM_CFGR2 978
LPTIM_CNT 978
LPTIM_CR 976
LPTIM_RCR 979
LPTIMx_DIER 970, 972
LPTIMx_ICR 968-969
LPTIMx_ISR 964, 966
LPUART_BRR 1370
LPUART_CR1 . . . . . 1360, 1363
LPUART_CR2 1365
LPUART_CR3 1367
LPUART_ICR 1378
LPUART_ISR 1371, 1375
LPUART_PRESC 1380
LPUART_RDR 1379
LPUART_RQR 1370
LPUART_TDR 1379

M

MCUROM_CIDR0 1587
MCUROM_CIDR1 1588
MCUROM_CIDR2 1588
MCUROM_CIDR3 1589
MCUROM_MEMTYPER 1584
MCUROM_PIDR0 1585
MCUROM_PIDR1 1586
MCUROM_PIDR2 1586
MCUROM_PIDR3 1587
MCUROM_PIDR4 1585

O

OPAMP1_CSR 650
OPAMP1_HSOTR 652
OPAMP1_OTR 652

P

PWR_BDCR 225
PWR_BDSR 226
PWR_DBPCR 226
PWR_IORETR 232
PWR_PMCR 221
PWR_PMSR 223
PWR_PRIVCFGR 232
PWR_SCCR 227
PWR_VMCR 228
PWR_VMSR 229
PWR_VOSCR 223
PWR_VOSSR 224
PWR_WUCR 231
PWR_WUSCR 230
PWR_WUSR 230

R

RAMCFG_M2WPR1 97
RAMCFG_MxCR 93
RAMCFG_MxDEAR 96
RAMCFG_MxECCKEYR 97
RAMCFG_MxERKEYR 98
RAMCFG_MxICR 96
RAMCFG_MxIER 94
RAMCFG_MxISR 95
RAMCFG_MxSEAR 95
RCC_AHB1ENR 287

RCC_AHB1LPENR 295
RCC_AHB1RSTR 280
RCC_AHB2ENR 288
RCC_AHB2LPENR 296
RCC_AHB2RSTR 281
RCC_APB1HENR 292
RCC_APB1HLPENR 300
RCC_APB1HRSTR 284
RCC_APB1LENR 290
RCC_APB1LLPENR 298
RCC_APB1LRSTR 282
RCC_APB2ENR 293
RCC_APB2LPENR 301
RCC_APB2RSTR 285
RCC_APB3ENR 294
RCC_APB3LPENR 302
RCC_APB3RSTR 286
RCC_BDCR 308
RCC_CCIPR1 303
RCC_CCIPR2 304
RCC_CCIPR3 305
RCC_CCIPR4 306
RCC_CCIPR5 307
RCC_CFGR1 263
RCC_CFGR2 266
RCC_CICR 278
RCC_CIER 276
RCC_CIFR 277
RCC_CR 259
RCC_CRRCR 262
RCC_CSICFGR 263
RCC_HSICFGR 262
RCC_PLL1CFGR 268
RCC_PLL1DIVR 272
RCC_PLL1FRACR 273
RCC_PLL2CFGR 270
RCC_PLL2DIVR 274
RCC_PLL2FRACR 275
RCC_PRIVCFGR 312
RCC_RSR 310
RNG_CR 664
RNG_DR 667
RNG_HTCR 667
RNG_SR 666
RTC_ALRABINR 1045
RTC_ALRBBINR 1045
RTC_ALRMAR 1038
RTC_ALRMASSR 1039
RTC_ALRMBR 1040
RTC_ALRMBSSR 1041
RTC_CALR 1034
RTC_CR 1029
RTC_DR 1024
RTC_ICSR 1026
RTC_MISR 1043
RTC_PRER 1027
RTC_PRIVCFGR 1032
RTC_SCR 1044
RTC_SHIFTR 1035
RTC_SR 1042
RTC_SSR 1025
RTC_TR 1023
RTC_TSDR 1037
RTC_TSSSR 1037
RTC_TSTR 1036
RTC_WPR 1033
RTC_WUTR 1028

S

SBS_CCCSR 358
SBS_CCSWCR 360
SBS_CCVALR 359
SBS_CFGR2 361
SBS_CNSLCKR 362
SBS_DBGCR 355
SBS_DBGLOCKR 356
SBS_ECCNMIR 362
SBS_FPUIMR 357
SBS_HDPLCR 354
SBS_HDPLSR 354
SBS_MESR 357
SBS_PMCR 356
SPI_CFG1 1435
SPI_CFG2 1438
SPI_CR1 1433
SPI_CR2 1435
SPI_CRCPOLY 1446
SPI_I2SCFGR 1449
SPI_IER 1440
SPI_IFCR 1444
SPI_RXCRC 1447
SPI_RXDR 1445
SPI_SR 1441
SPI_TXCRC 1447
SPI_TXDR 1445
SPI_UDRDR 1448
SYSROM_CIDR0 1582
SYSROM_CIDR1 1582
SYSROM_CIDR2 1583
SYSROM_CIDR3 1583
SYSROM_MEMTYPER 1579
SYSROM_PIDR0 1580
SYSROM_PIDR1 1580

SYSROM_PIDR2 1581
SYSROM_PIDR3 1581
SYSROM_PIDR4 1579

T

TAMP_ATCR1 1066
TAMP_ATCR2 1069
TAMP_ATOR 1068
TAMP_ATSEEDR 1068
TAMP_BKPxR 1081
TAMP_CFGR 1070
TAMP_COUNT1R 1079
TAMP_CR1 1060
TAMP_CR2 1062
TAMP_CR3 1063
TAMP_FLTCR 1065
TAMP_IER 1073
TAMP_MISR 1076
TAMP_PRIVCFGR 1072
TAMP_RPCFGR 1080
TAMP_SCR 1078
TAMP_SR 1075
TIM1_AF1 809
TIM1_AF2 812
TIM1_ARR 797
TIM1_BDTR 801
TIM1_CCER 793
TIM1_CCMR1 785-786
TIM1_CCMR2 789-790
TIM1_CCMR3 806
TIM1_CCR1 798
TIM1_CCR2 799
TIM1_CCR3 799
TIM1_CCR4 800
TIM1_CCR5 804
TIM1_CCR6 805
TIM1_CNT 796
TIM1_CR1 772
TIM1_CR2 773
TIM1_DCR 814
TIM1_DIER 779
TIM1_DMAR 815
TIM1_DTR2 806
TIM1_ECR 807
TIM1_EGR 783
TIM1_PSC 797
TIM1_RCR 797
TIM1_SMCR 776
TIM1_SR 781
TIM1_TISEL 808
TIM2_ARR 910
TIM2_CCR1 911
TIM2_CCR2 912
TIM2_CCR3 914
TIM2_CCR4 915
TIM2_CNT 908
TIM3_ARR 909
TIM3_CCR1 910
TIM3_CCR2 912
TIM3_CCR3 913
TIM3_CCR4 914
TIM3_CNT 908
TIMx_AF1 918
TIMx_AF2 918
TIMx_ARR 941
TIMx_CCER 906
TIMx_CCMR1 . . . . . 900, 902
TIMx_CCMR2 904-905
TIMx_CNT 940
TIMx_CR1 . . . . . 891, 937
TIMx_CR2 . . . . . 892, 938
TIMx_DCR 919
TIMx_DIER 896, 939
TIMx_DMAR 920
TIMx_ECR 916
TIMx_EGR 900, 940
TIMx_PSC 909, 941
TIMx_SMCR 894
TIMx_SR 898, 939
TIMx_TISEL 917
TPIU_ACPR 1666
TPIU_CIDR0 1674
TPIU_CIDR1 1674
TPIU_CIDR2 1675
TPIU_CIDR3 1675
TPIU_CLAIMCLR 1669
TPIU_CLAIMSETR 1669
TPIU_CSPSR 1665
TPIU_DEVIDR 1670
TPIU_DEVTYPER 1671
TPIU_FFCR 1668
TPIU_FFSR 1667
TPIU_PIDR0 1672
TPIU_PIDR1 1672
TPIU_PIDR2 1673
TPIU_PIDR3 1673
TPIU_PIDR4 1671
TPIU_PSCR 1668
TPIU_SPPR 1666
TPIU_SSPSR 1665

U

USART_BRR 1314
USART_CR1 . . . . . .1300, 1303
USART_CR2 1306
USART_CR3 1310
USART_GTPR 1314
USART_ICR 1327
USART_ISR 1317, 1322
USART_PRESC 1329
USART_RDR 1328
USART_RQR 1316
USART_RTOR 1315
USART_TDR 1329
USB_BCDR 1537
USB_CHEP_RXTXBD_n 1548
USB_CHEP_TXRXBD_n 1545-1547
USB_CHEPnR 1539
USB_CNTR 1530
USB_DADDR 1536
USB_FNR 1535
USB_ISTR 1533
USB_LPMCSR 1537

W

WWDG_CFR 1003
WWDG_CR 1002
WWDG_SR 1004

重要通知——请仔细阅读

意法半导体公司及其子公司（“ST”）保留随时对 ST 产品和/或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于 ST 产品的最新信息。ST 产品的销售依照订单确认时的相关 ST 销售条款。

买方自行负责对ST产品的选择和使用，ST概不承担与应用协助或买方产品设计相关的任何责任。

ST不对任何知识产权进行任何明示或默示的授权或许可。

转售的ST产品如有不同于此处提供的信息的规定，将导致ST针对该产品授予的任何保证失效。

ST 和 ST 徽标是 ST 的商标。关于 ST 商标的其他信息，请访问 *www.st.com/trademarks*。其他所有产品或服务名称是其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。

© 2024 STMicroelectronics - 保留所有权利