RM0385
参考手册

STM32F75xxx 和 STM32F74xxx
基于 ARM 内核的 32 位高级 MCU

前言

本参考手册面向应用开发人员， 提供有关使用 STM32F75xxx 和 STM32F74xxx 微控制器存储器与外设的完整信息。

STM32F75xxx 和 STM32F74xxx 构成一个微控制器系列，各产品具有不同的存储器大小、封装和外设。

有关订购信息以及器件的机械与电气特性，请参见数据手册。

有关带 FPU 的 ARM® Cortex®-M7 内核的信息，请参见带 FPU 的 Cortex®-M7 技术参考手册。

相关文档

意法半导体网站 *www.st.com* 提供以下文档：

- STM32F75xxx 和 STM32F74xxx 数据手册

目录

1 文档约定 58

1.1 寄存器相关缩写词列表 58

1.2 词汇表 58

1.3 外设可用性 59

2 系统和存储器概述 60

2.1 系统架构 60

2.1.1 multi-AHB 总线矩阵 61

2.1.2 AHB/APB 总线桥 (APB) 61

2.1.3 CPU AXIM 总线 62

2.1.4 ITCM 总线 62

2.1.5 DTCM 总线 62

2.1.6 CPU AHBS 总线 62

2.1.7 AHB 外设总线 62

2.1.8 DMA 存储器总线 62

2.1.9 DMA 外设总线 63

2.1.10 以太网 DMA 总线 63

2.1.11 USB OTG HS DMA 总线 63

2.1.12 LCD-TFT 控制器 DMA 总线 63

2.1.13 DMA2D 总线 63

2.2 存储器组织结构 64

2.2.1 前言 64

2.2.2 存储器映射和寄存器边界地址 64

2.3 内部 SRAM 68

2.4 Flash 概述 68

2.5 自举配置 68

3 嵌入式 Flash (FLASH) 70

3.1 前言 70

3.2 Flash 主要特性 70

3.3 Flash 功能说明 71

3.3.1 Flash 构成 71

3.3.2 读访问延迟 72

3.3.3 Flash 编程/擦除操作 74
3.3.4 Flash 控制寄存器解锁 74
3.3.5 编程/擦除并行位数 74
3.3.6 Flash 擦除顺序 74
3.3.7 Flash 编程顺序 75
3.3.8 Flash 中断 76
3.4 FLASH 选项字节 76
3.4.1 选项字节说明 76
3.4.2 选项字节编程 79
3.5 Flash 保护 80
3.5.1 读保护 (RDP) 80
3.5.2 写保护 81
3.6 一次性可编程字节 82
3.7 FLASH 寄存器 83
3.7.1 Flash 访问控制寄存器 (FLASH_ACR) 83
3.7.2 Flash 密钥寄存器 (FLASH_KEYR) 84
3.7.3 Flash 选项密钥寄存器 (FLASH_OPTKEYR) 84
3.7.4 Flash 状态寄存器 (FLASH_SR) 85
3.7.5 Flash 控制寄存器 (FLASH_CR) 86
3.7.6 Flash 选项控制寄存器 (FLASH_OPTCR) 87
3.7.7 Flash 选项控制寄存器 (FLASH_OPTCR1) 89
3.7.8 Flash 接口寄存器映射 90

4 电源控制器 (PWR) 91
4.1 电源 91
4.1.1 独立 A/D 转换器电源和参考电压 92
4.1.2 独立的 USB 收发器电源 93
4.1.3 电池备份域 94
4.1.4 调压器 96
4.2 电源监控器 98
4.2.1 上电复位 (POR)/掉电复位 (PDR) 98
4.2.2 欠压复位 (BOR) 99
4.2.3 可编程电压检测器 (PVD) 100
4.3 低功耗模式 101
4.3.1 调试模式 104
4.3.2 运行模式 104

4.3.3 低功耗模式 105
4.3.4 睡眠模式 106
4.3.5 停止模式 107
4.3.6 待机模式 110
4.3.7 对 RTC 复用功能进行编程以从停止模式和待机模式唤醒器件 111
4.4 电源控制寄存器 113
4.4.1 PWR 电源控制寄存器 (PWR_CR1) 113
4.4.2 PWR 电源控制/状态寄存器 (PWR_CSR1) 115
4.4.3 PWR 电源控制/状态寄存器 2 (PWR_CR2) 117
4.4.4 PWR 电源控制寄存器 2 (PWR_CSR2) 118
4.5 PWR 寄存器映射 120

5 复位和时钟控制 (RCC) 121
5.1 复位 121
5.1.1 系统复位 121
5.1.2 电源复位 121
5.1.3 备份域复位 122
5.2 时钟 122
5.2.1 HSE 时钟 125
5.2.2 HSI 时钟 126
5.2.3 PLL 126
5.2.4 LSE 时钟 127
5.2.5 LSI 时钟 127
5.2.6 系统时钟 (SYSCLK) 选择 127
5.2.7 时钟安全系统 (CSS) 127
5.2.8 RTC/AWU 时钟 128
5.2.9 看门狗时钟 128
5.2.10 时钟输出功能 128
5.2.11 基于 TIM5/TIM11 的内部/外部时钟测量 129
5.2.12 外设时钟使能寄存器（RCC_AHBxENR、 RCC_APBxENRy） 130
5.3 RCC 寄存器 130
5.3.1 RCC 时钟控制寄存器 (RCC_CR) 131
5.3.2 RCC PLL 配置寄存器 (RCC_PLLCFGR) 132
5.3.3 RCC 时钟配置寄存器 (RCC_CFGR) 134
5.3.4 RCC 时钟中断寄存器 (RCC_CIR) 137
5.3.5 RCC AHB1 外设复位寄存器 (RCC_AHB1RSTR) 139
5.3.6 RCC AHB2 外设复位寄存器 (RCC_AHB2RSTR) 141

5.3.7 RCC AHB3 外设复位寄存器 (RCC_AHB3RSTR) 142
5.3.8 RCC APB1 外设复位寄存器 (RCC_APB1RSTR) 143
5.3.9 RCC APB2 外设复位寄存器 (RCC_APB2RSTR) 146
5.3.10 RCC AHB1 外设时钟寄存器 (RCC_AHB1ENR) 148
5.3.11 RCC AHB2 外设时钟使能寄存器 (RCC_AHB2ENR) 150
5.3.12 RCC AHB3 外设时钟使能寄存器 (RCC_AHB3ENR) 151
5.3.13 RCC APB1 外设时钟使能寄存器 (RCC_APB1ENR) 152
5.3.14 RCC APB2 外设时钟使能寄存器 (RCC_APB2ENR) 155
5.3.15 低功耗模式寄存器中的 RCC AHB1 外设时钟使能 (RCC_AHB1LPENR) 157
5.3.16 低功耗模式寄存器中的 RCC AHB2 外设时钟使能 (RCC_AHB2LPENR) 160
5.3.17 低功耗模式寄存器中的 RCC AHB3 外设时钟使能 (RCC_AHB3LPENR) 161
5.3.18 低功耗模式寄存器中的 RCC APB1 外设时钟使能 (RCC_APB1LPENR) 161
5.3.19 低功耗模式寄存器中的 RCC APB2 外设时钟 (RCC_APB2LPENR) . . 165
5.3.20 RCC 备份域控制寄存器 (RCC_BDCR) 167
5.3.21 RCC 时钟控制和状态寄存器 (RCC_CSR) 168
5.3.22 RCC 扩频时钟生成寄存器 (RCC_SSCGR) 170
5.3.23 RCC PLLI2S 配置寄存器 (RCC_PLLI2SCFGR) 171
5.3.24 RCC PLL 配置寄存器 (RCC_PLLSAICFGR) 172
5.3.25 RCC 专用时钟配置寄存器 (RCC_DKCFGR1) 174
5.3.26 RCC 专用时钟配置寄存器 (DCKCFGR) 176
5.3.27 RCC 寄存器映射 178

6 通用 I/O (GPIO) 182

6.1 简介 182
6.2 GPIO 主要特性 182
6.3 GPIO 功能描述 182
6.3.1 通用 I/O (GPIO) 185
6.3.2 I/O 引脚复用功能复用器和映射 185
6.3.3 I/O 端口控制寄存器 186
6.3.4 I/O 端口数据寄存器 186
6.3.5 I/O 数据位操作 186
6.3.6 GPIO 锁定机制 187
6.3.7 I/O 复用功能输入/输出 187
6.3.8 外部中断线/唤醒线 187

6.3.9 输入配置 . . . 187
6.3.10 输出配置 . . . 188
6.3.11 复用功能配置 . . . 189
6.3.12 模拟配置 . . . 190
6.3.13 将 HSE 或 LSE 振荡器引脚用作 GPIO . . . 190
6.3.14 在备份电源域中使用 GPIO 引脚 . . . 191
6.4 GPIO 寄存器 . . . 191
6.4.1 GPIO 端口模式寄存器 (GPIOx_MODER) (x =A..K) . . . 191
6.4.2 GPIO 端口输出类型寄存器 (GPIOx_OTYPER) (x = A..K) . . . 192
6.4.3 GPIO 端口输出速度寄存器 (GPIOx_OSPEEDR) (x = A..K) . . . 192
6.4.4 GPIO 端口上拉/下拉寄存器 (GPIOx_PUPDR) (x = A..K) . . . 193
6.4.5 GPIO 端口输入数据寄存器 (GPIOx_IDR) (x = A..K) . . . 193
6.4.6 GPIO 端口输出数据寄存器 (GPIOx_ODR) (x = A..K) . . . 194
6.4.7 GPIO 端口置位/复位寄存器 (GPIOx_BSRR) (x = A..K) . . . 194
6.4.8 GPIO 端口配置锁定寄存器 (GPIOx_LCKR) (x = A..K) . . . 195
6.4.9 GPIO 复用功能低位寄存器 (GPIOx_AFRL) (x = A..K) . . . 196
6.4.10 GPIO 复用功能高位寄存器 (GPIOx_AFRH) (x = A..J) . . . 196
6.4.11 GPIO 寄存器映射 . . . 197

7 系统配置控制器 (SYSCFG) . . . 199
7.1 I/O 补偿单元 . . . 199
7.2 SYSCFG 寄存器 . . . 199
7.2.1 SYSCFG 存储器重映射寄存器 (SYSCFG_MEMRMP) . . . 199
7.2.2 SYSCFG 外设模式配置寄存器 (SYSCFG_PMC) . . . 200
7.2.3 SYSCFG 外部中断配置寄存器 1 (SYSCFG_EXTICR1) . . . 201
7.2.4 SYSCFG 外部中断配置寄存器 2 (SYSCFG_EXTICR2) . . . 202
7.2.5 SYSCFG 外部中断配置寄存器 3 (SYSCFG_EXTICR3) . . . 202
7.2.6 SYSCFG 外部中断配置寄存器 4 (SYSCFG_EXTICR4) . . . 203
7.2.7 补偿单元控制寄存器 (SYSCFG_CMPCR) . . . 204
7.2.8 SYSCFG 寄存器映射 . . . 205

8 直接存储器访问控制器 (DMA) . . . 206
8.1 DMA 简介 . . . 206
8.2 DMA 主要特性 . . . 206
8.3 DMA 功能说明 . . . 207
8.3.1 一般说明 . . . 207
8.3.2 DMA 传输 . . . 208

8.3.3 通道选择 (Channel selection) . . . 208
8.3.4 仲裁器 . . . 210
8.3.5 DMA 数据流 . . . 210
8.3.6 源、目标和传输模式 . . . 210
8.3.7 指针递增 . . . 213
8.3.8 循环模式 . . . 214
8.3.9 双缓冲区模式 . . . 214
8.3.10 可编程数据宽度、封装/解封、字节序 . . . 215
8.3.11 单次传输和突发传输 . . . 217
8.3.12 FIFO . . . 217
8.3.13 DMA 传输完成 . . . 220
8.3.14 DMA 传输暂停 . . . 221
8.3.15 流控制器 . . . 221
8.3.16 可能的 DMA 配置汇总 . . . 222
8.3.17 流配置过程 . . . 223
8.3.18 错误管理 . . . 224
8.4 DMA 中断 . . . 225
8.5 DMA 寄存器 . . . 225
8.5.1 DMA 低中断状态寄存器 (DMA_LISR) . . . 225
8.5.2 DMA 高中断状态寄存器 (DMA_HISR) . . . 226
8.5.3 DMA 低中断标志清零寄存器 (DMA_LIFCR) . . . 227
8.5.4 DMA 高中断标志清零寄存器 (DMA_HIFCR) . . . 228
8.5.5 DMA 数据流 x 配置寄存器 (DMA_SxCR) (x = 0..7) . . . 229
8.5.6 DMA 数据流 x 数据项数寄存器 (DMA_SxNDTR) (x = 0..7) . . . 232
8.5.7 DMA 数据流 x 外设地址寄存器 (DMA_SxPAR) (x = 0..7) . . . 233
8.5.8 DMA 数据流 x 存储器 0 地址寄存器 (DMA_SxM0AR) (x = 0..7) . . . 233
8.5.9 DMA 数据流 x 存储器 1 地址寄存器 (DMA_SxM1AR) (x = 0..7) . . . 234
8.5.10 DMA 数据流 x FIFO 控制寄存器 (DMA_SxFCR) (x = 0..7) . . . 234
8.5.11 DMA 寄存器映射 . . . 236

9 Chrom-Art Accelerator™ 控制器 (DMA2D) . . . 240
9.1 DMA2D 简介 . . . 240
9.2 DMA2D 的主要特性 . . . 240
9.3 DMA2D 功能说明 . . . 241
9.3.1 概述 . . . 241
9.3.2 DMA2D 控制 . . . 242
9.3.3 DMA2D 前景层 FIFO 和背景层 FIFO . . . 242

9.3.4 DMA2D 前景层和背景层像素格式转换器 (PFC) 242
9.3.5 DMA2D 前景层 FIFO 和背景层 CLUT 接口 244
9.3.6 DMA2D 混合器 245
9.3.7 DMA2D 输出 PFC 245
9.3.8 DMA2D 输出 FIFO 246
9.3.9 DMA2D AHB 主设备端口定时器 246
9.3.10 DMA2D 事务 247
9.3.11 DMA2D 配置 247
9.3.12 DMA2D 传输控制（启动、挂起、中止和完成） 249
9.3.13 水印 249
9.3.14 错误管理 250
9.3.15 AHB 死区 250
9.4 DMA2D 中断 250
9.5 DMA2D 寄存器 251
9.5.1 DMA2D 控制寄存器 (DMA2D_CR) 251
9.5.2 DMA2D 中断状态寄存器 (DMA2D_ISR) 253
9.5.3 DMA2D 中断标志清零寄存器 (DMA2D_IFCR) 254
9.5.4 DMA2D 前景层存储器地址寄存器 (DMA2D_FGMAR) 255
9.5.5 DMA2D 前景层偏移寄存器 (DMA2D_FGOR) 255
9.5.6 DMA2D 背景层存储器地址寄存器 (DMA2D_BGMAR) 256
9.5.7 DMA2D 背景层偏移寄存器 (DMA2D_BGOR) 256
9.5.8 DMA2D 前景层 PFC 控制寄存器 (DMA2D_FGPFCCR) 257
9.5.9 DMA2D 前景层颜色寄存器 (DMA2D_FGCOLR) 258
9.5.10 DMA2D 背景层 PFC 控制寄存器 (DMA2D_BGPFCCR) 259
9.5.11 DMA2D 背景层颜色寄存器 (DMA2D_BGCOLR) 260
9.5.12 DMA2D 前景层 CLUT 存储器地址寄存器 (DMA2D_FGCMAR) 261
9.5.13 DMA2D 背景层 CLUT 存储器地址寄存器 (DMA2D_BGCMAR) 261
9.5.14 DMA2D 输出 PFC 控制寄存器 (DMA2D_OPFCCR) 262
9.5.15 DMA2D 输出颜色寄存器 (DMA2D_OCOLR) 262
9.5.16 DMA2D 输出存储器地址寄存器 (DMA2D_OMAR) 263
9.5.17 DMA2D 输出偏移寄存器 (DMA2D_OOR) 264
9.5.18 DMA2D 行数寄存器 (DMA2D_NLR) 264
9.5.19 DMA2D 行水印寄存器 (DMA2D_LWR) 265
9.5.20 DMA2D AHB 主设备定时器配置寄存器 (DMA2D_AMTCR) 265
9.5.21 DMA2D 寄存器映射 266

10 嵌套向量中断控制器 (NVIC) 268
10.1 NVIC 特性 268
10.1.1 SysTick 校准值寄存器 268
10.1.2 中断和异常向量 268

11 外部中断和事件控制器 (EXTI) 273
11.1 EXTI 主要特性 273
11.2 EXTI 框图 273
11.3 唤醒事件管理 274
11.4 功能描述 274
11.5 硬件中断选择 274
11.6 硬件事件选择 274
11.7 软件中断/事件选择 274
11.8 外部中断/事件线映射 275
11.9 EXTI 寄存器 276
11.9.1 中断屏蔽寄存器 (EXTI_IMR) 276
11.9.2 事件屏蔽寄存器 (EXTI_EMR) 276
11.9.3 上升沿触发选择寄存器 (EXTI_RTSR) 277
11.9.4 下降沿触发选择寄存器 (EXTI_FTSR) 277
11.9.5 软件中断事件寄存器 (EXTI_SWIER) 278
11.9.6 挂起寄存器 (EXTI_PR) 278
11.9.7 EXTI 寄存器映射 279

12 循环冗余校验计算单元（CRC） 280
12.1 简介 280
12.2 CRC 主要特性 280
12.3 CRC 功能说明 280
12.4 CRC 寄存器 282
12.4.1 数据寄存器 (CRC_DR) 282
12.4.2 独立数据寄存器 (CRC_IDR) 282
12.4.3 控制寄存器 (CRC_CR) 283
12.4.4 CRC 初始值 (CRC_INIT) 284
12.4.5 CRC 多项式 (CRC_POL) 284
12.4.6 CRC 寄存器映射 285

13 可变存储控制器 (FMC) 286
13.1 FMC 主要特性 286
13.2 框图 287
13.3 AHB 接口 288
13.3.1 支持的存储器和事务 288
13.4 外部器件地址映射 289
13.4.1 NOR/PSRAM 地址映射 291
13.4.2 NAND Flash 地址映射 291
13.4.3 SDRAM 地址映射 292
13.5 NOR Flash/PSRAM 控制器 295
13.5.1 外部存储器接口信号 296
13.5.2 支持的存储器和事务 298
13.5.3 通用时序规则 300
13.5.4 NOR Flash/PSRAM 控制器异步事务 300
13.5.5 同步事务 317
13.5.6 NOR/PSRAM 控制寄存器 324
13.6 NAND Flash 控制器 330
13.6.1 外部存储器接口信号 331
13.6.2 NAND Flash 支持的存储器和事务 332
13.6.3 NAND Flash 的时序图 332
13.6.4 NAND Flash 操作 333
13.6.5 NAND Flash 预等待功能 334
13.6.6 纠错码 (ECC) 计算 (NAND Flash) 335
13.6.7 NAND Flash 控制寄存器 336
13.7 SDRAM控制器 341
13.7.1 SDRAM 控制器主要特性 341
13.7.2 SDRAM 外部存储器接口信号 341
13.7.3 SDRAM 控制器功能说明 342
13.7.4 低功耗模式 348
13.7.5 SDRAM 控制寄存器 351
13.8 FMC 寄存器映射 358

14 Quad-SPI 接口 (QUADSPI) 360
14.1 前言 360
14.2 QUADSPI 主要特性 360
14.3 QUADSPI 功能说明 360

14.3.1 QUADSPI 框图 360
14.3.2 QUADSPI 命令序列 361
14.3.3 QUADSPI 信号接口协议模式 363
14.3.4 QUADSPI 间接模式 366
14.3.5 QUADSPI 状态标志轮询模式 367
14.3.6 QUADSPI 内存映射模式 367
14.3.7 QUADSPI Flash 配置 368
14.3.8 QUADSPI 延迟数据采样 368
14.3.9 QUADSPI 配置 368
14.3.10 QUADSPI 的用法 369
14.3.11 指令仅发送一次 370
14.3.12 QUADSPI 差错管理 371
14.3.13 QUADSPI 的繁忙位和中止功能 371
14.3.14 nCS 行为 371
14.4 QUADSPI 中断 373
14.5 QUADSPI 寄存器 374
14.5.1 QUADSPI 控制寄存器 (QUADSPI_CR) 374
14.5.2 QUADSPI 器件配置寄存器 (QUADSPI_CCR) 377
14.5.3 QUADSPI 状态寄存器 (QUADSPI_SR) 378
14.5.4 QUADSPI 标志清零寄存器 (QUADSPI_FCR) 379
14.5.5 QUADSPI 数据长度寄存器 (QUADSPI_DLR) 380
14.5.6 QUADSPI 通信配置寄存器 (QUADSPI_CCR) 381
14.5.7 QUADSPI 地址寄存器 (QUADSPI_AR) 383
14.5.8 QUADSPI 交替字节寄存器 (QUADSPI_ABR) 383
14.5.9 QUADSPI 数据寄存器 (QUADSPI_DR) 384
14.5.10 QUADSPI 轮询状态屏蔽寄存器 (QUADSPI _PSMKR) 384
14.5.11 QUADSPI 轮询状态匹配寄存器 (QUADSPI _PSMAR) 385
14.5.12 QUADSPI 轮询间隔寄存器 (QUADSPI _PIR) 385
14.5.13 QUADSPI 低功耗超时寄存器 (QUADSPI_LPTR) 386
14.5.14 QUADSPI 寄存器映射 387

15 模数转换器 (ADC) 388
15.1 ADC 简介 388
15.2 ADC 主要特性 388
15.3 ADC 功能说明 388
15.3.1 ADC 开关控制 390
15.3.2 ADC1/2 和 ADC3 连接功能 390

15.3.3 ADC 时钟 . . . 394
15.3.4 通道选择 (Channel selection) . . . 394
15.3.5 单次转换模式 . . . 394
15.3.6 连续转换模式 . . . 395
15.3.7 时序图 . . . 395
15.3.8 模拟看门狗 . . . 395
15.3.9 扫描模式 (Scan mode) . . . 396
15.3.10 注入通道管理 . . . 396
15.3.11 不连续采样模式 . . . 397
15.4 数据对齐 (Data alignment) . . . 398
15.5 可独立设置各通道采样时间 . . . 399
15.6 外部触发转换和触发极性 . . . 399
15.7 快速转换模式 . . . 401
15.8 数据管理 . . . 401
15.8.1 使用 DMA . . . 401
15.8.2 在不使用 DMA 的情况下管理转换序列 . . . 402
15.8.3 在不使用 DMA 和溢出检测的情况下进行转换 . . . 402
15.9 多重 ADC 模式 . . . 402
15.9.1 注入同步模式 . . . 406
15.9.2 常规同步模式 . . . 407
15.9.3 交替模式 . . . 408
15.9.4 交替触发模式 . . . 409
15.9.5 混合型常规/注入同步模式 . . . 411
15.9.6 常规同步 + 交替触发组合模式 . . . 411
15.10 温度传感器 . . . 412
15.11 电池充电监视 . . . 414
15.12 ADC 中断 . . . 414
15.13 ADC 寄存器 . . . 414
15.13.1 ADC 状态寄存器 (ADC_SR) . . . 414
15.13.2 ADC 控制寄存器 1 (ADC_CR1) . . . 415
15.13.3 ADC 控制寄存器 2 (ADC_CR2) . . . 417
15.13.4 ADC 采样时间寄存器 1 (ADC_SMPR1) . . . 419
15.13.5 ADC 采样时间寄存器 2 (ADC_SMPR2) . . . 420
15.13.6 ADC 注入通道数据偏移寄存器 x (ADC_JOFRx) (x=1..4) . . . 420
15.13.7 ADC 看门狗高阈值寄存器 (ADC_HTR) . . . 421
15.13.8 ADC 看门狗低阈值寄存器 (ADC_LTR) . . . 421

15.13.9 ADC 常规序列寄存器 1 (ADC_SQR1) 422
15.13.10 ADC 常规序列寄存器 2 (ADC_SQR2) 422
15.13.11 ADC 常规序列寄存器 3 (ADC_SQR3) 423
15.13.12 ADC 注入序列寄存器 (ADC_JSQR) 423
15.13.13 ADC 注入数据寄存器 x (ADC_JDRx) (x= 1..4) 424
15.13.14 ADC 常规数据寄存器 (ADC_DR) 424
15.13.15 ADC 通用状态寄存器 (ADC_CSR) 425
15.13.16 ADC 通用控制寄存器 (ADC_CCR) 426
15.13.17 适用于双重和三重模式的 ADC 通用常规数据寄存器 (ADC_CDR) . . . 428
15.13.18 ADC 寄存器映射 429

16 数模转换器 (DAC) 431

16.1 DAC 简介 431
16.2 DAC 主要特性 431
16.3 DAC 功能说明 432
16.3.1 DAC 通道使能 432
16.3.2 DAC 输出缓冲器使能 433
16.3.3 DAC 数据格式 433
16.3.4 DAC 转换 434
16.3.5 DAC 输出电压 434
16.3.6 DAC 触发选择 435
16.3.7 DMA 请求 435
16.3.8 生成噪声 436
16.3.9 生成三角波 437
16.4 DAC 双通道转换 437
16.4.1 独立触发（不产生波形） 438
16.4.2 独立触发（生成单个 LFSR） 438
16.4.3 独立触发（生成不同 LFSR） 438
16.4.4 独立触发（生成单个三角波） 439
16.4.5 独立触发（生成不同三角波） 439
16.4.6 同步软件启动 439
16.4.7 同步触发（不产生波形） 439
16.4.8 同步触发（生成单个 LFSR） 440
16.4.9 同步触发（生成不同 LFSR） 440
16.4.10 同步触发（生成单个三角波） 440
16.4.11 同步触发（生成不同三角波） 441

16.5 DAC 寄存器 441
16.5.1 DAC 控制寄存器 (DAC_CR) 441
16.5.2 DAC 软件触发寄存器 (DAC_SWTRIGR) 444
16.5.3 DAC 1 通道 12 位右对齐数据保持寄存器 (DAC_DHR12R1) 444
16.5.4 DAC 1 通道 12 位左对齐数据保持寄存器 (DAC_DHR12L1) 445
16.5.5 DAC 1 通道 8 位右对齐数据保持寄存器 (DAC_DHR8R1) 445
16.5.6 DAC 2 通道 12 位右对齐数据保持寄存器 (DAC_DHR12R2) 445
16.5.7 DAC 2 通道 12 位左对齐数据保持寄存器 (DAC_DHR12L2) 446
16.5.8 DAC 2 通道 8 位右对齐数据保持寄存器 (DAC_DHR8R2) 446
16.5.9 双 DAC 12 位右对齐数据保持寄存器 (DAC_DHR12RD) 447
16.5.10 双 DAC 12 位左对齐数据保持寄存器 (DAC_DHR12LD) 447
16.5.11 双 DAC 8 位右对齐数据保持寄存器 (DAC_DHR8RD) 448
16.5.12 DAC 1 通道数据输出寄存器 (DAC_DOR1) 448
16.5.13 DAC 2 通道数据输出寄存器 (DAC_DOR2) 448
16.5.14 DAC 状态寄存器 (DAC_SR) 449
16.5.15 DAC 寄存器映射 449

17 数字摄像头接口 (DCMI) 451
17.1 DCMI 简介 451
17.2 DCMI 主要特性 451
17.3 DCMI 引脚 451
17.4 DCMI 时钟 451
17.5 DCMI 功能概述 452
17.5.1 DMA 接口 453
17.5.2 DCMI 物理接口 453
17.5.3 同步 455
17.5.4 捕获模式 457
17.5.5 裁剪功能 458
17.5.6 JPEG 格式 459
17.5.7 FIFO 459
17.6 数据格式说明 459
17.6.1 数据格式 459
17.6.2 单色格式 459
17.6.3 RGB 格式 460
17.6.4 YCbCr 格式 460

17.6.5 YCbCr 格式——仅含 Y 分量 461
17.6.6 半分辨率图像提取 461
17.7 DCMI 中断 461
17.8 DCMI 寄存器说明 461
17.8.1 DCMI 控制寄存器 (DCMI_CR) 462
17.8.2 DCMI 状态寄存器 (DCMI_SR) 464
17.8.3 DCMI 原始中断状态寄存器 (DCMI_SR) 464
17.8.4 DCMI 中断使能寄存器 (DCMI_IER) 465
17.8.5 DCMI 屏蔽中断状态寄存器 (DCMI_MIS) 466
17.8.6 DCMI 中断清零寄存器 (DCMI_ICR) 467
17.8.7 DCMI 内嵌同步码寄存器 (DCMI_ESCR) 468
17.8.8 DCMI 内嵌码同步取消屏蔽寄存器 (DCMI_ESUR) 468
17.8.9 DCMI 裁剪窗口起点 (DCMI_CWSTRT) 469
17.8.10 DCMI 裁剪窗口大小 (DCMI_CWSIZE) 469
17.8.11 DCMI 数据寄存器 (DCMI_DR) 470
17.8.12 DCMI 寄存器映射 470

18 LCD-TFT 控制器 (LTDC) 472
18.1 简介 472
18.2 LTDC 主要特性 472
18.3 LTDC 功能说明 473
18.3.1 LTDC 框图 473
18.3.2 LTDC 复位和时钟 473
18.3.3 LCD-TFT 引脚和信号接口 474
18.4 LTDC 可编程参数 474
18.4.1 LTDC 全局配置参数 474
18.4.2 层可编程参数 477
18.5 LTDC 中断 481
18.6 LTDC 编程步骤 482
18.7 LTDC 寄存器 483
18.7.1 LTDC 同步大小配置寄存器 (LTDC_SSCR) 483
18.7.2 LTDC 后沿配置寄存器 (LTDC_BPCR) 483
18.7.3 LTDC 有效宽度配置寄存器 (LTDC_AWCR) 484
18.7.4 LTDC 总宽度配置寄存器 (LTDC_TWCR) 485
18.7.5 LTDC 全局控制寄存器 (LTDC_GCR) 485
18.7.6 LTDC 影子重载配置寄存器 (LTDC_SRCR) 487

18.7.7 LTDC 背景色配置寄存器 (LTDC_BCCR) 487
18.7.8 LTDC 中断使能寄存器 (LTDC_IER) 488
18.7.9 LTDC 中断状态寄存器 (LTDC_ISR) 489
18.7.10 LTDC 中断清零寄存器 (LTDC_ICR) 489
18.7.11 LTDC 行中断位置配置寄存器 (LTDC_LIPCR) 490
18.7.12 LTDC 当前位置状态寄存器 (LTDC_CPSR) 491
18.7.13 LTDC 当前显示状态寄存器 (LTDC_CDSR) 491
18.7.14 LTDC 第 x 层控制寄存器 (LTDC_LxCR)（其中，x = 1..2） 492
18.7.15 LTDC 第 x 层窗口水平位置配置寄存器 (LTDC_LxWHPCR)（其中 x=1..2） 493
18.7.16 LTDC 第 x 层窗口垂直位置配置寄存器 (LTDC_LxWVPCR)（其中 x=1..2） 494
18.7.17 LTDC 第 x 层色键配置寄存器 (LTDC_LxCKCR)（其中 x=1..2） 495
18.7.18 LTDC 第 x 层像素格式配置寄存器 (LTDC_LxPFCR)（其中 x=1..2） 495
18.7.19 LTDC 第 x 层常数 Alpha 配置寄存器 (LTDC_LxCACR)（其中 x=1..2） 496
18.7.20 LTDC 第 x 层默认颜色配置寄存器 (LTDC_LxDCCR)（其中 x=1..2） 497
18.7.21 LTDC 第 x 层混合系数配置寄存器 (LTDC_LxBFCR)（其中 x=1..2） 497
18.7.22 LTDC 第 x 层颜色帧缓冲区地址寄存器 (LTDC_LxCFBAR)（其中 x=1..2） 499
18.7.23 LTDC 第 x 层颜色帧缓冲区长度寄存器 (LTDC_LxCFBLR)（其中 x=1..2） 499
18.7.24 LTDC 第 x 层颜色帧缓冲区行数寄存器 (LTDC_LxCFBLNR)（其中 x=1..2） 500
18.7.25 LTDC 第 x 层 CLUT 写寄存器 (LTDC_LxCLUTWR)（其中 x=1..2） 500
18.7.26 LTDC 寄存器映射 501

19 随机数发生器 (RNG) 505

19.1 简介 505
19.2 RNG 主要特性 505
19.3 RNG 功能说明 505
19.3.1 操作 506
19.3.2 错误管理 506
19.4 RNG 寄存器 506
19.4.1 RNG 控制寄存器 (RNG_CR) 506
19.4.2 RNG 状态寄存器 (RNG_SR) 507

19.4.3 RNG 数据寄存器 (RNG_DR) 508
19.4.4 RNG 寄存器映射 508

20 加密处理器 (CRYP) 509
20.1 CRYP 简介 509
20.2 CRYP 主要特性 509
20.3 CRYP 功能说明 510
20.3.1 DES/TDES 加密内核 511
20.3.2 AES 加密内核 515
20.3.3 数据类型 525
20.3.4 初始化向量 —— CRYP_IV0...1(L/R) 528
20.3.5 CRYP 忙碌状态 529
20.3.6 加密或解密执行步骤 530
20.3.7 上下文交换 531
20.4 CRYP 中断 532
20.5 CRYP DMA 接口 533
20.6 CRYP 寄存器 533
20.6.1 CRYP 控制寄存器 (CRYP_CR) 534
20.6.2 CRYP 状态寄存器 (CRYP_SR) 536
20.6.3 CRYP 数据输入寄存器 (CRYP_DIN) 537
20.6.4 CRYP 数据输出寄存器 (CRYP_DOUT) 538
20.6.5 CRYP DMA 控制寄存器 (CRYP_DMACR) 539
20.6.6 CRYP 中断屏蔽置位/清零寄存器 (CRYP_IMSCR) 539
20.6.7 CRYP 原始中断状态寄存器 (CRYP_RISR) 540
20.6.8 CRYP 屏蔽中断状态寄存器 (CRYP_MISR) 540
20.6.9 CRYP 密钥寄存器 (CRYP_K0...3(L/R)R) 541
20.6.10 CRYP 初始化向量寄存器 (CRYP_IV0...1(L/R)R) 543
20.6.11 CRYP 上下文交换寄存器（CRYP_CSGCMCCM0..7R 和 CRYP_CSGCM0..7R） 544
20.6.12 CRYP 寄存器映射 545

21 散列处理器 (HASH) 548
21.1 散列简介 548
21.2 散列主要特性 548
21.3 散列功能说明 549

21.3.1 处理的持续时间 550
21.3.2 数据类型 550
21.3.3 消息摘要计算 552
21.3.4 消息填充 553
21.3.5 散列运算 554
21.3.6 HMAC 运算 554
21.3.7 上下文交换 555
21.3.8 散列中断 556
21.4 散列寄存器 556
21.4.1 HASH 控制寄存器 (HASH_CR) 556
21.4.2 散列数据输入寄存器 (HASH_DIN) 558
21.4.3 散列启动寄存器 (HASH_STR) 559
21.4.4 散列摘要寄存器（HASH_HR0 到 4/5/6/7） 560
21.4.5 散列中断使能寄存器 (HASH_IMR) 562
21.4.6 散列状态寄存器 (HASH_SR) 563
21.4.7 散列上下文交换寄存器 (HASH_CSRx) 564
21.4.8 散列寄存器映射 564

22 高级控制定时器 (TIM1/TIM8) 566
22.1 TIM1/TIM8简介 566
22.2 TIM1/TIM8 主要特性 566
22.3 TIM1/TIM8功能说明 568
22.3.1 时基单元 568
22.3.2 计数器模式 570
22.3.3 重复计数器 580
22.3.4 外部触发输入 581
22.3.5 时钟选择 582
22.3.6 捕获/比较通道 585
22.3.7 输入捕获模式 588
22.3.8 PWM 输入模式 589
22.3.9 强制输出模式 590
22.3.10 输出比较模式 590
22.3.11 PWM 模式 592
22.3.12 不对称 PWM 模式 595
22.3.13 组合 PWM 模式 595
22.3.14 组合三相 PWM 模式 596
22.3.15 互补输出和死区插入 597

22.3.16 使用断路功能 599
22.3.17 发生外部事件时清除 OCxREF 信号 604
22.3.18 生成 6 步 PWM 604
22.3.19 单脉冲模式 605
22.3.20 可再触发单脉冲模式 (OPM) 607
22.3.21 编码器接口模式 607
22.3.22 UIF 位重映射 609
22.3.23 定时器输入异或功能 609
22.3.24 连接霍尔传感器 610
22.3.25 定时器同步 612
22.3.26 ADC 同步 615
22.3.27 DMA 连续传送模式 615
22.3.28 调试模式 616
22.4 TIM1/TIM8 寄存器 617
22.4.1 TIM1/TIM8 控制寄存器 1 (TIMx_CR1) 617
22.4.2 TIM1/TIM8 控制寄存器 2 (TIMx_CR2) 618
22.4.3 TIM1/TIM8 从模式控制寄存器 (TIMx_SMCR) 621
22.4.4 TIM1/TIM8 DMA/中断使能寄存器 (TIMx_DIER) 623
22.4.5 TIM1/TIM8 状态寄存器 (TIMx_SR) 625
22.4.6 TIM1/TIM8 事件产生寄存器 (TIMx_EGR) 626
22.4.7 TIM1/TIM8 捕获/比较模式寄存器 1 (TIMx_CCMR1) 628
22.4.8 TIM1/TIM8 捕获/比较模式寄存器 2 (TIMx_CCMR2) 631
22.4.9 TIM1/TIM8 捕获/比较使能寄存器 (TIMx_CCER) 633
22.4.10 TIM1/TIM8 计数器 (TIMx_CNT) 636
22.4.11 TIM1/TIM8 预分频器 (TIMx_PSC) 636
22.4.12 TIM1/TIM8 自动重载寄存器 (TIMx_ARR) 636
22.4.13 TIM1/TIM8 重复计数器寄存器 (TIMx_RCR) 637
22.4.14 TIM1/TIM8 捕获/比较寄存器 1 (TIMx_CCR1) 637
22.4.15 TIM1/TIM8 捕获/比较寄存器 2 (TIMx_CCR2) 637
22.4.16 TIM1/TIM8 捕获/比较寄存器 3 (TIMx_CCR3) 638
22.4.17 TIM1/TIM8 捕获/比较寄存器 4 (TIMx_CCR4) 638
22.4.18 TIM1/TIM8 断路和死区寄存器 (TIMx_BDTR) 639
22.4.19 TIM1/TIM8 DMA 控制寄存器 (TIMx_DCR) 642
22.4.20 TIM1/TIM8 全传输 DMA 地址 (TIMx_DMAR) 642
22.4.21 TIM1/TIM8 捕获/比较模式寄存器 3 (TIMx_CCMR3) 643
22.4.22 TIM1/TIM8 捕获/比较寄存器 5 (TIMx_CCR5) 644
22.4.23 TIM1/TIM8 捕获/比较寄存器 6 (TIMx_CCR6) 645

22.4.24 TIM1 寄存器映射 645
22.4.25 TIM8 寄存器映射 647

23 通用定时器（TIM2/TIM3/TIM4/TIM5） 650

23.1 TIM2/TIM3/TIM4/TIM5 简介 650
23.2 TIM2/TIM3/TIM4/TIM5 的主要特性 650
23.3 TIM2/TIM3/TIM4/TIM5 功能说明 652
23.3.1 时基单元 652
23.3.2 计数器模式 654
23.3.3 时钟选择 664
23.3.4 捕获/比较通道 668
23.3.5 输入捕获模式 670
23.3.6 PWM 输入模式 671
23.3.7 强制输出模式 672
23.3.8 输出比较模式 672
23.3.9 PWM 模式 673
23.3.10 不对称 PWM 模式 676
23.3.11 组合 PWM 模式 677
23.3.12 发生外部事件时清除 OCxREF 信号 678
23.3.13 单脉冲模式 679
23.3.14 编码器接口模式 681
23.3.15 UIF 位重映射 683
23.3.16 定时器输入异或功能 683
23.3.17 定时器与外部触发同步 683
23.3.18 定时器同步 687
23.3.19 DMA 连续传送模式 691
23.3.20 调试模式 692
23.4 TIM2/TIM3/TIM4/TIM5 寄存器 692
23.4.1 TIMx 控制寄存器 1 (TIMx_CR1) 692
23.4.2 TIMx 控制寄存器 2 (TIMx_CR2) 694
23.4.3 TIMx 从模式控制寄存器 (TIMx_SMCR) 695
23.4.4 TIMx DMA/中断使能寄存器 (TIMx_DIER) 697
23.4.5 TIMx 状态寄存器 (TIMx_SR) 699
23.4.6 TIMx 事件生成寄存器 (TIMx_EGR) 700
23.4.7 TIMx 捕获/比较模式寄存器 1 (TIMx_CCMR1) 701
23.4.8 TIMx 捕获/比较模式寄存器 2 (TIMx_CCMR2) 705
23.4.9 TIMx 捕获/比较使能寄存器 (TIMx_CCER) 706

23.4.10 TIMx 计数器 (TIMx_CNT) 708
23.4.11 TIMx 预分频器 (TIMx_PSC) 708
23.4.12 TIMx 自动重载寄存器 (TIMx_ARR) 709
23.4.13 TIMx 捕获/比较寄存器 1 (TIMx_CCR1) 709
23.4.14 TIMx 捕获/比较寄存器 2 (TIMx_CCR2) 710
23.4.15 TIMx 捕获/比较寄存器 3 (TIMx_CCR3) 710
23.4.16 TIMx 捕获/比较寄存器 4 (TIMx_CCR4) 711
23.4.17 TIMx DMA 控制寄存器 (TIMx_DCR) 711
23.4.18 TIMx 全传输 DMA 地址 (TIMx_DMAR) 712
23.4.19 TIM2 选项寄存器 1 (TIM2_OR) 712
23.4.20 TIM2 选项寄存器 1 (TIM5_OR) 713
23.4.21 TIM3 选项寄存器 1 (TIM3_OR1) 713
23.4.22 TIMx 寄存器映射 714

24 通用定时器（TIM9 到 TIM14） 717

24.1 TIM9 到 TIM14 简介 717
24.2 TIM9 到 TIM14 主要特性 717
24.2.1 TIM9/TIM12 主要特性 717
24.2.2 TIM10/TIM11 和 TIM13/TIM14 主要特性 718
24.3 TIM9 到 TIM14 功能说明 719
24.3.1 时基单元 719
24.3.2 计数器模式 721
24.3.3 时钟选择 725
24.3.4 捕获/比较通道 727
24.3.5 输入捕获模式 729
24.3.6 PWM 输入模式（仅适用于 TIM9/12） 729
24.3.7 强制输出模式 730
24.3.8 输出比较模式 731
24.3.9 PWM 模式 732
24.3.10 单脉冲模式 733
24.3.11 TIM9/12 外部触发同步 735
24.3.12 定时器同步 (TIM9/12) 737
24.3.13 调试模式 737
24.4 TIM9 和 TIM12 寄存器 737
24.4.1 TIM9/12 控制寄存器 1 (TIMx_CR1) 738
24.4.2 TIM9/12 从模式控制寄存器 (TIMx_SMCR) 739
24.4.3 TIM9/12 中断使能寄存器 (TIMx_DIER) 740

24.4.4 TIM9/12 状态寄存器 (TIMx_SR) 741
24.4.5 TIM9/12 事件生成寄存器 (TIMx_EGR) 742
24.4.6 TIM9/12 捕获/比较模式寄存器 1 (TIMx_CCMR1) 743
24.4.7 TIM9/12 捕获/比较使能寄存器 (TIMx_CCER) 746
24.4.8 TIM9/12 计数器 (TIMx_CNT) 747
24.4.9 TIM9/12 预分频器 (TIMx_PSC) 747
24.4.10 TIM9/12 自动重载寄存器 (TIMx_ARR) 748
24.4.11 TIM9/12 捕获/比较寄存器 1 (TIMx_CCR1) 748
24.4.12 TIM9/12 捕获/比较寄存器 2 (TIMx_CCR2) 749
24.4.13 TIM9/12 寄存器映射 749
24.5 TIM10/11/13/14 寄存器 751
24.5.1 TIM10/11/13/14 控制寄存器 1 (TIMx_CR1) 751
24.5.2 TIM10/11/13/14 中断使能寄存器 (TIMx_DIER) 752
24.5.3 TIM10/11/13/14 状态寄存器 (TIMx_SR) 752
24.5.4 TIM10/11/13/14 事件生成寄存器 (TIMx_EGR) 753
24.5.5 TIM10/11/13/14 捕获/比较模式寄存器 1 (TIMx_CCMR1) 754
24.5.6 TIM10/11/13/14 捕获/比较使能寄存器 (TIMx_CCER) 756
24.5.7 TIM10/11/13/14 计数器 (TIMx_CNT) 757
24.5.8 TIM10/11/13/14 预分频器 (TIMx_PSC) 757
24.5.9 TIM10/11/13/14 自动重载寄存器 (TIMx_ARR) 757
24.5.10 TIM10/11/13/14 捕获/比较寄存器 1 (TIMx_CCR1) 758
24.5.11 TIM11 选项寄存器 1 (TIM11_OR) 758
24.5.12 TIM10/11/13/14 寄存器映射 759

25 基本定时器 (TIM6/TIM7) 761
25.1 TIM6/TIM7 简介 761
25.2 TIM6/TIM7 主要特性 761
25.3 TIM6/TIM7 功能说明 762
25.3.1 时基单元 762
25.3.2 计数模式 764
25.3.3 UIF 位重映射 767
25.3.4 时钟源 767
25.3.5 调试模式 768
25.4 TIM6/TIM7 寄存器 768
25.4.1 TIM6/TIM7 控制寄存器 1 (TIMx_CR1) 769
25.4.2 TIM6/TIM7 控制寄存器 2 (TIMx_CR2) 770
25.4.3 TIM6/TIM7 DMA/中断使能寄存器 (TIMx_DIER) 770

25.4.4 TIM6/TIM7 状态寄存器 (TIMx_SR) 771
25.4.5 TIM6/TIM7 事件产生寄存器 (TIMx_EGR) 771
25.4.6 TIM6/TIM7 计数器 (TIMx_CNT) 772
25.4.7 TIM6/TIM7 预分频器 (TIMx_PSC) 772
25.4.8 TIM6/TIM7 自动重载寄存器 (TIMx_ARR) 772
25.4.9 TIM6/TIM7 寄存器映射 773

26 低功耗定时器 (LPTIM) 774
26.1 前言 774
26.2 LPTIM 主要特性 774
26.3 LPTIM 实现 774
26.4 LPTIM 功能说明 775
26.4.1 LPTIM 框图 775
26.4.2 LPTIM 复位和时钟 775
26.4.3 干扰滤波器 776
26.4.4 预分频器 776
26.4.5 触发多路复用器 777
26.4.6 工作模式 777
26.4.7 超时功能 779
26.4.8 生成波形 779
26.4.9 寄存器更新 780
26.4.10 计数器模式 781
26.4.11 定时器使能 781
26.4.12 编码器模式 781
26.5 LPTIM 中断 783
26.6 LPTIM 寄存器 783
26.6.1 LPTIM 中断和状态寄存器 (LPTIMx_ISR) 783
26.6.2 LPTIM 中断清零寄存器 (LPTIMx_ICR) 784
26.6.3 LPTIM 中断使能寄存器 (LPTIMx_IER) 785
26.6.4 LPTIM 配置寄存器 (LPTIMx_CFGR) 786
26.6.5 LPTIM 控制寄存器 (LPTIMx_CR) 789
26.6.6 LPTIM 比较寄存器 (LPTIMx_CMP) 790
26.6.7 LPTIM 自动重载寄存器 (LPTIMx_ARR) 790
26.6.8 LPTIM 计数器寄存器 (LPTIMx_CNT) 791
26.6.9 LPTIM1 选项寄存器 (LPTIM1_OR) 791

26.6.10 LPTIM2 选项寄存器 (LPTIM2_OR) 792
26.6.11 LPTIM 寄存器映射 793

27 独立看门狗 (IWDG) 794
27.1 前言 794
27.2 IWDG 主要特性 794
27.3 IWDG 功能说明 794
27.3.1 IWDG 框图 794
27.3.2 窗口选项 795
27.3.3 硬件看门狗 795
27.3.4 低功耗冻结 795
27.3.5 寄存器访问保护 795
27.3.6 调试模式 796
27.4 IWDG 寄存器 796
27.4.1 键寄存器 (IWDG_KR) 796
27.4.2 预分频器寄存器 (IWDG_PR) 797
27.4.3 重载寄存器 (IWDG_RLR) 798
27.4.4 状态寄存器 (IWDG_SR) 798
27.4.5 窗口寄存器 (IWDG_WINR) 799
27.4.6 IWDG 寄存器映射 800

28 系统窗口看门狗 (WWDG) 801
28.1 前言 801
28.2 WWDG 主要特性 801
28.3 WWDG 功能说明 801
28.3.1 使能看门狗 802
28.3.2 控制递减计数器 802
28.3.3 看门狗中断高级特性 802
28.3.4 如何设置看门狗超时 803
28.3.5 调试模式 804
28.4 WWDG 寄存器 804
28.4.1 控制寄存器 (WWDG_CR) 804
28.4.2 配置寄存器 (WWDG_CFR) 805
28.4.3 状态寄存器 (WWDG_SR) 805
28.4.4 WWDG 寄存器映射 806

29 实时时钟 (RTC) 807
29.1 前言 807
29.2 RTC 主要特性 807
29.3 RTC 功能说明 808
29.3.1 RTC 框图 808
29.3.2 RTC 控制的 GPIO 809
29.3.3 时钟和预分频器 811
29.3.4 实时时钟和日历 812
29.3.5 可编程闹钟 812
29.3.6 周期性自动唤醒 813
29.3.7 RTC 初始化和配置 813
29.3.8 读取日历 815
29.3.9 复位 RTC 816
29.3.10 RTC 同步 816
29.3.11 RTC 参考时钟检测 816
29.3.12 RTC 精密数字校准 817
29.3.13 时间戳功能 819
29.3.14 入侵检测 819
29.3.15 校准时钟输出 821
29.3.16 闹钟输出 821
29.4 RTC 低功耗模式 822
29.5 RTC 中断 822
29.6 RTC 寄存器 823
29.6.1 RTC 时间寄存器 (RTC_TR) 823
29.6.2 RTC 日期寄存器 (RTC_DR) 824
29.6.3 RTC 控制寄存器 (RTC_CR) 825
29.6.4 RTC 初始化和状态寄存器 (RTC_ISR) 827
29.6.5 RTC 预分频器寄存器 (RTC_PRER) 829
29.6.6 RTC 唤醒定时器寄存器 (RTC_WUTR) 830
29.6.7 RTC 闹钟 A 寄存器 (RTC_ALRMAR) 831
29.6.8 RTC 闹钟 B 寄存器 (RTC_ALRMBR) 832
29.6.9 RTC 写保护寄存器 (RTC_WPR) 833
29.6.10 RTC 亚秒寄存器 (RTC_SSR) 833
29.6.11 RTC 平移控制寄存器 (RTC_SHIFTR) 834
29.6.12 RTC 时间戳时间寄存器 (RTC_TSTR) 835
29.6.13 RTC 时间戳日期寄存器 (RTC_TSDR) 836

29.6.14 RTC 时间戳亚秒寄存器 (RTC_TSSSR) 836
29.6.15 RTC 校准寄存器 (RTC_CALR) 837
29.6.16 RTC 入侵配置寄存器 (RTC_TAMPCR) 838
29.6.17 RTC 闹钟 A 亚秒寄存器 (RTC_ALRMASSR) 841
29.6.18 RTC 闹钟 B 亚秒寄存器 (RTC_ALRMBSSR) 842
29.6.19 RTC 选项寄存器 (RTC_OR) 843
29.6.20 RTC 备份寄存器 (RTC_BKPxR) 843
29.6.21 RTC 寄存器映射 844

30 内部集成电路 (I2C) 接口 846

30.1 简介 846
30.2 I2C 主要特性 846
30.3 I2C 特性实现 847
30.4 I2C 功能说明 847
30.4.1 I2C 框图 848
30.4.2 I2C 时钟要求 849
30.4.3 模式选择 849
30.4.4 I2C 初始化 850
30.4.5 软件复位 854
30.4.6 数据传输 854
30.4.7 I2C 从模式 856
30.4.8 I2C 主模式 864
30.4.9 I2C_TIMINGR 寄存器配置示例 876
30.4.10 SMBus 特性 876
30.4.11 SMBus 初始化 878
30.4.12 SMBus：I2C_TIMEOUTR 寄存器配置示例 880
30.4.13 SMBus 从模式 881
30.4.14 错误条件 887
30.4.15 DMA 请求 888
30.4.16 调试模式 889
30.5 I2C 低功耗模式 889
30.6 I2C 中断 889
30.7 I2C 寄存器 891
30.7.1 控制寄存器 1 (I2C_CR1) 891
30.7.2 控制寄存器 2 (I2C_CR2) 894
30.7.3 设备自身地址 1 寄存器 (I2C_OAR1) 896

30.7.4 设备自身地址 2 寄存器 (I2C_OAR2) 897
30.7.5 时序寄存器 (I2C_TIMINGR) 898
30.7.6 超时寄存器 (I2C_TIMEOUTR) 899
30.7.7 中断和状态寄存器 (I2C_ISR) 900
30.7.8 中断清零寄存器 (I2C_ICR) 902
30.7.9 PEC 寄存器 (I2C_PECR) 903
30.7.10 接收数据寄存器 (I2C_RXDR) 904
30.7.11 发送数据寄存器 (I2C_TXDR) 904
30.7.12 I2C寄存器映射 905

31 通用同步异步收发器 (USART) 907

31.1 前言 907
31.2 USART 主要特性 907
31.3 USART 扩展特性 908
31.4 USART 实现 908
31.5 USART 功能说明 909
31.5.1 USART 字符说明 911
31.5.2 发送器 912
31.5.3 接收器 915
31.5.4 波特率生成 920
31.5.5 USART 接收器对时钟偏差的容差 923
31.5.6 自动波特率检测 924
31.5.7 多处理器通信 925
31.5.8 Modbus 通信 926
31.5.9 奇偶校验控制 927
31.5.10 LIN（局域互连网络）模式 928
31.5.11 USART 同步模式 930
31.5.12 单线半双工通信 933
31.5.13 智能卡模式 933
31.5.14 IrDA SIR ENDEC 模块 937
31.5.15 使用 DMA 进行连续通信 939
31.5.16 RS232 硬件流控制和 RS485 驱动器使能 941
31.6 USART 低功耗模式 943
31.7 USART 中断 943
31.8 USART 寄存器 944

31.8.1 控制寄存器 1 (USARTx_CR1) 945
31.8.2 控制寄存器 2 (USARTx_CR2) 948
31.8.3 控制寄存器 3 (USARTx_CR3) 951
31.8.4 波特率寄存器 (USARTx_BRR) 954
31.8.5 保护时间和预分频器寄存器 (USARTx_GTPR) 955
31.8.6 接收器超时寄存器 (USARTx_RTOR) 956
31.8.7 请求寄存器 (USARTx_RQR) 957
31.8.8 中断和状态寄存器 (USARTx_ISR) 958
31.8.9 中断标志清零寄存器 (USARTx_ICR) 961
31.8.10 接收数据寄存器 (USARTx_RDR) 962
31.8.11 发送数据寄存器 (USARTx_TDR) 963
31.8.12 USART 寄存器映射 963

32 串行外设接口/集成电路内置音频总线 (SPI/I2S) 965

32.1 简介 965
32.2 SPI 主要特性 965
32.3 I2S 主要特性 966
32.4 SPI/I2S 实现 966
32.5 SPI 功能说明 967
32.5.1 一般说明 967
32.5.2 一个主器件和一个从器件之间的通信 968
32.5.3 标准多从器件通信 970
32.5.4 从器件选择 (NSS) 引脚管理 971
32.5.5 通信格式 972
32.5.6 SPI 配置 974
32.5.7 使能 SPI 的步骤 975
32.5.8 数据发送和接收过程 975
32.5.9 SPI 状态标志 984
32.5.10 SPI 错误标志 985
32.5.11 NSS 脉冲模式 986
32.5.12 TI 模式 986
32.5.13 CRC 计算 987
32.6 SPI 中断 989
32.7 I^2S 功能说明 990
32.7.1 I^2S 一般说明 990
32.7.2 支持的音频协议 991

32.7.3 启动描述 . . . 997
32.7.4 时钟发生器 . . . 999
32.7.5 I2S 主模式 . . . 1001
32.7.6 I2S 从模式 . . . 1002
32.7.7 I2S 状态标志 . . . 1004
32.7.8 I2S 错误标志 . . . 1005
32.7.9 DMA 特性 . . . 1005
32.8 I2S 中断 . . . 1005
32.9 SPI 和 I2S 寄存器 . . . 1006
32.9.1 SPIx 控制寄存器 1 (SPI_CR1) . . . 1006
32.9.2 SPIx 控制寄存器 2 (SPI_CR2) . . . 1008
32.9.3 SPI 状态寄存器 (SPIx_SR) . . . 1010
32.9.4 SPI 数据寄存器 (SPIx_DR) . . . 1012
32.9.5 SPI CRC 多项式寄存器 (SPIx_CRCPR) . . . 1012
32.9.6 SPI 接收 CRC 寄存器 (SPIx_RXCRCR) . . . 1013
32.9.7 SPI 发送 CRC 寄存器 (SPIx_TXCRCR) . . . 1013
32.9.8 SPIx_I2S 配置寄存器 (SPIx_I2SCFGR) . . . 1014
32.9.9 SPIx_I2S 预分频器寄存器 (SPIx_I2SPR) . . . 1016
32.9.10 SPI/I2S 寄存器映射 . . . 1017

33 串行音频接口 (SAI) . . . 1018
33.1 简介 . . . 1018
33.2 SAI 主要特性 . . . 1018
33.3 SAI 功能说明 . . . 1019
33.3.1 SAI 框图 . . . 1019
33.3.2 SAI 的主要模式 . . . 1020
33.3.3 SAI 同步模式 . . . 1021
33.3.4 音频数据大小 . . . 1022
33.3.5 帧同步 . . . 1022
33.3.6 Slot 配置 . . . 1025
33.3.7 SAI 时钟发生器 . . . 1027
33.3.8 内部 FIFO . . . 1028
33.3.9 AC’97 链路控制器 . . . 1030
33.3.10 SPDIF 输出 . . . 1032
33.3.11 特性 . . . 1034
33.3.12 错误标志 . . . 1038

33.3.13 禁止 SAI 1041
33.3.14 SAI DMA 接口 1041
33.4 SAI 中断 1042
33.5 SAI 寄存器 1043
33.5.1 全局配置寄存器 (SAI_GCR) 1043
33.5.2 配置寄存器 1 (SAI_ACR1/SAI_BCR1) 1043
33.5.3 配置寄存器 2 (SAI_ACR2/SAI_BCR2) 1046
33.5.4 帧配置寄存器 (SAI_AFRCR/SAI_BFRCR) 1048
33.5.5 Slot 寄存器 (SAI_ASLOTR/SAI_BSLOTR) 1049
33.5.6 中断屏蔽寄存器 2 (SAI_AIM/SAI_BIM) 1050
33.5.7 状态寄存器 (SAI_ASR/SAI_BSR) 1052
33.5.8 清除标志寄存器 (SAI_ACLRFR/SAI_BCLRFR) 1054
33.5.9 数据寄存器 (SAI_ADR/SAI_BDR) 1055
33.5.10 SAI 寄存器映射 1055

34 SPDIF 接收器接口 (SPDIFRX) 1057
34.1 SPDIFRX 接口简介 1057
34.2 SPDIFRX 主要特性 1057
34.3 SPDIFRX 功能说明 1057
34.3.1 S/PDIF 协议 (IEC-60958) 1058
34.3.2 SPDIFRX 解码器 (SPDIFRX_DC) 1060
34.3.3 SPDIFRX 对时钟偏差的容差 1063
34.3.4 SPDIFRX 同步 1064
34.3.5 SPDIFRX 处理 1066
34.3.6 数据接收管理 1068
34.3.7 专用控制流 1070
34.3.8 接收错误 1070
34.3.9 时钟策略 1073
34.3.10 DMA 接口 1073
34.3.11 中断生成 1074
34.3.12 寄存器保护 1075
34.4 编程步骤 1076
34.4.1 初始化阶段 1076
34.4.2 处理来自 SPDIFRX 的中断 1077
34.4.3 处理来自 DMA 的中断 1077

34.5 SPDIFRX 接口寄存器 1078
34.5.1 控制寄存器 (SPDIFRX_CR) 1078
34.5.2 中断屏蔽寄存器 (SPDIFRX_IMR) 1080
34.5.3 状态寄存器 (SPDIFRX_SR) 1081
34.5.4 中断标志清零寄存器 (SPDIFRX_IFCR) 1083
34.5.5 数据输入寄存器 (SPDIFRX_DR) 1084
34.5.6 数据输入寄存器 (SPDIFRX_DR) 1085
34.5.7 数据输入寄存器 (SPDIFRX_DR) 1086
34.5.8 通道状态寄存器 (SPDIFRX_SR) 1086
34.5.9 调试信息寄存器 (SPDIFRX_DIR) 1087
34.5.10 SPDIFRX 接口寄存器映射 1088

35 SD/SDIO/MMC 卡主机接口 (SDMMC) 1089

35.1 SDMMC主要特性 1089
35.2 SDMMC 总线拓扑 1089
35.3 SDMMC 功能描述 1091
35.3.1 SDMMC 适配器 1092
35.3.2 SDMMC APB2 接口 1103
35.4 卡功能说明 1105
35.4.1 卡识别模式 1105
35.4.2 智能卡复位 1105
35.4.3 工作电压范围验证 1105
35.4.4 卡识别过程 1105
35.4.5 块写入 1106
35.4.6 块读取 1107
35.4.7 流访问、流写入和流读取（仅限多媒体卡） 1107
35.4.8 擦除：组擦除和扇区擦除 1108
35.4.9 宽总线选择或取消选择 1108
35.4.10 保护管理 1109
35.4.11 卡状态寄存器 1111
35.4.12 SD 状态寄存器 1114
35.4.13 SD I/O 模式 1117
35.4.14 命令和响应 1118
35.5 响应格式 1121
35.5.1 R1（正常响应命令） 1122
35.5.2 R1b 1122

35.5.3 R2（CID 和 CSD 寄存器） 1122
35.5.4 R3（OCR 寄存器） 1122
35.5.5 R4（快速 I/O） 1123
35.5.6 R4b 1123
35.5.7 R5（中断请求） 1124
35.5.8 R6 1124
35.6 SDIO I/O 卡专用操作 1125
35.6.1 通过触发 SDMMC_D2 执行的 SDIO I/O 读取等待操作 1125
35.6.2 通过停止 SDMMC_CK 执行的 SDIO 读取等待操作 1125
35.6.3 SDIO 挂起/恢复操作 1126
35.6.4 SDIO 中断 1126
35.7 硬件流控制 1126
35.8 SDMMC 寄存器 1126
35.8.1 SDMMC 电源控制寄存器 (SDMMC_POWER) 1127
35.8.2 SDMMC 时钟控制寄存器 (SDMMC_CLKCR) 1127
35.8.3 SDMMC 参数寄存器 (SDMMC_ARG) 1129
35.8.4 SDMMC 命令寄存器 (SDMMC_CMD) 1129
35.8.5 SDMMC 命令响应寄存器 (SDMMC_RESPCMD) 1130
35.8.6 SDMMC 响应 1..4 寄存器 (SDMMC_RESPx) 1131
35.8.7 SDMMC 数据定时器寄存器 (SDMMC_DTIMER) 1132
35.8.8 SDMMC 数据长度寄存器 (SDMMC_DLEN) 1132
35.8.9 SDMMC 数据控制寄存器 (SDMMC_DCTRL) 1133
35.8.10 SDMMC 数据计数器寄存器 (SDMMC_DCOUNT) 1134
35.8.11 SDMMC 状态寄存器 (SDMMC_STA) 1135
35.8.12 SDMMC 中断清零寄存器 (SDMMC_ICR) 1136
35.8.13 SDMMC 屏蔽寄存器 (SDMMC_MASK) 1138
35.8.14 SDMMC FIFO 计数器寄存器 (SDMMC_FIFOCNT) 1140
35.8.15 SDMMC 数据 FIFO 寄存器 (SDMMC_FIFO) 1141
35.8.16 SDMMC 寄存器映射 1141

36 控制器区域网络 (bxCAN) 1143
36.1 前言 1143
36.2 bxCAN 主要特性 1143
36.3 bxCAN 一般说明 1144
36.3.1 CAN 2.0B 主动内核 1144
36.3.2 控制、状态和配置寄存器 1144

36.3.3 发送邮箱 1144
36.3.4 验收滤波器 1144
36.4 bxCAN 工作模式 1145
36.4.1 初始化模式 1146
36.4.2 正常模式 1146
36.4.3 睡眠模式（低功耗） 1146
36.5 测试模式 1147
36.5.1 静默模式 1147
36.5.2 环回模式 1148
36.5.3 环回与静默组合模式 1148
36.6 调试模式下的行为 1149
36.7 bxCAN 功能说明 1149
36.7.1 发送处理 1149
36.7.2 时间触发通信模式 (Time triggered communication mode) 1150
36.7.3 接收处理 1150
36.7.4 标识符筛选 1152
36.7.5 消息存储 1156
36.7.6 错误管理 1157
36.7.7 位时序 1157
36.8 bxCAN 中断 1160
36.9 CAN 寄存器 1161
36.9.1 寄存器访问保护 1161
36.9.2 CAN 控制和状态寄存器 1161
36.9.3 CAN 邮箱寄存器 1171
36.9.4 CAN 滤波器寄存器 1177
36.9.5 bxCAN 寄存器映射 1181

37 USB on-the-go 全速/高速 (OTG_FS/OTG_HS) 1185
37.1 前言 1185
37.2 USB_OTG 主要特性 1186
37.2.1 通用特性 1186
37.2.2 主机模式特性 1187
37.2.3 从机模式特性 1187
37.3 USB_OTG 实现 1187
37.4 USB OTG 功能说明 1188

37.4.1 USB OTG 框图 1188
37.4.2 OTG 模块 1189
37.4.3 全速 OTG PHY 1190
37.4.4 嵌入式全速 OTG PHY 1190
37.4.5 高速 OTG PHY 1191
37.4.6 使用 I2C 接口的外部全速 OTG PHY 1191
37.5 OTG 双角色设备 (DRD) 1191
37.5.1 ID 线检测 1191
37.5.2 HNP 双角色设备 1192
37.5.3 SRP 双角色设备 1192
37.6 USB 设备 1192
37.6.1 支持 SRP 功能的 USB 设备 1193
37.6.2 USB 设备状态 1193
37.6.3 USB 设备端点 1194
37.7 USB 主机 1196
37.7.1 支持 SRP 功能的 USB 主机 1196
37.7.2 USB 主机状态 1197
37.7.3 主机通道 1198
37.7.4 主机调度器 1199
37.8 SOF 触发 1200
37.8.1 主机 SOF 1200
37.8.2 设备 SOF 1200
37.9 电源选项 1201
37.10 动态更新 OTG_HFIR 寄存器 1202
37.11 USB 数据 FIFO 1202
37.11.1 设备 FIFO 架构 1203
37.11.2 主机 FIFO 架构 1204
37.11.3 FIFO RAM 分配 1205
37.12 OTG_FS 系统性能 1206
37.13 OTG_FS/OTG_HS 中断 1206
37.14 OTG_FS/OTG_HS 控制和状态寄存器 1207
37.14.1 CSR 存储器映射 1208
37.15 OTG_FS/OTG_HS 寄存器 1213
37.15.1 OTG 控制和状态寄存器 (OTG_GOTGCTL) 1214
37.15.2 OTG 中断寄存器 (OTG_GOTGINT) 1217
37.15.3 OTG AHB 配置寄存器 (OTG_GAHBCFG) 1218

37.15.4 OTG USB 配置寄存器 (OTG_GUSBCFG) 1219
37.15.5 OTG 复位寄存器 (OTG_GRSTCTL) 1222
37.15.6 OTG 模块中断寄存器 (OTG_GINTSTS) 1224
37.15.7 OTG 中断屏蔽寄存器 (OTG_GINTMSK) 1229
37.15.8 OTG_FS 接收状态调试读取/OTG 状态读取和出栈寄存器 (OTG_GRXSTSR/OTG_GRXSTSP) 1232
37.15.9 OTG 接收 FIFO 大小寄存器 (OTG_GRXFSIZ) 1234
37.15.10 OTG 主机非周期性发送 FIFO 大小寄存器 (OTG_HNPTXFSIZ)/ 端点 0 发送 FIFO 大小 (OTG_DIEPTXF0) 1234
37.15.11 OTG 非周期性发送 FIFO/队列状态寄存器 (OTG_HNPTXSTS) 1235
37.15.12 OTG I^2C 访问寄存器 (OTG_GI2CCTL) 1236
37.15.13 OTG 通用模块配置寄存器 (OTG_GCCFG) 1237
37.15.14 OTG 模块 ID 寄存器 (OTG_CID) 1238
37.15.15 OTG 模块 LPM 配置寄存器 (OTG_GLPMCFG) 1238
37.15.16 OTG 主机周期性发送 FIFO 大小寄存器 (OTG_HPTXFSIZ) 1242
37.15.17 OTG 设备 IN 端点发送 FIFO 大小寄存器（OTG_DIEPTXFx) （x = 1..5[FS] / 7[HS]，其中 x 为 FIFO 编号） 1242
37.15.18 主机模式寄存器 1243
37.15.19 OTG 主机配置寄存器 (OTG_HCFG) 1243
37.15.20 OTG 主机帧间隔寄存器 (OTG_HFIR) 1244
37.15.21 OTG 主机帧编号/帧剩余时间寄存器 (OTG_HFNUM) 1245
37.15.22 OTG_Host 周期性发送 FIFO/队列状态寄存器 (OTG_HPTXSTS) . . . 1245
37.15.23 OTG 主机全体通道中断寄存器 (OTG_HAINT) 1246
37.15.24 OTG 主机全体通道中断屏蔽寄存器 (OTG_HAINTMSK) 1247
37.15.25 OTG 主机端口控制和状态寄存器 (OTG_HPRT) 1247
37.15.26 OTG 主机通道 x 特性寄存器 (OTG_HCCHARx) （x = 0..15[HS] / 11[FS]，其中 x = 通道编号） 1250
37.15.27 OTG 主机通道 x 分离控制寄存器 (OTG_HCSPLTx) （x = 0..15，其中 x = 通道编号） 1251
37.15.28 OTG 主机通道 x 中断寄存器 (OTG_HCINTx) （x = 0..15[HS] / 11[FS]，其中 x = 通道编号） 1252
37.15.29 OTG 主机通道 x 中断屏蔽寄存器 (OTG_HCINTMSKx) （x = 0..15[HS] / 11[FS]，其中 x = 通道编号） 1253
37.15.30 OTG 主机通道 x 传输大小寄存器 (OTG_HCTSIZx) （x = 0..15[HS] / 11[FS]，其中 x = 通道编号） 1255
37.15.31 OTG 主机通道 x DMA 地址寄存器 (OTG_HCDMAx) （x = 0..15，其中 x = 通道编号） 1256
37.15.32 设备模式寄存器 1256
37.15.33 OTG 设备配置寄存器 (OTG_DCFG) 1256
37.15.34 OTG 设备控制寄存器 (OTG_DCTL) 1258

37.15.35 OTG 设备状态寄存器 (OTG_DSTS) 1260
37.15.36 OTG 设备 IN 端点通用中断屏蔽寄存器 (OTG_DIEPMSK) 1261
37.15.37 OTG 设备 OUT 端点通用中断屏蔽寄存器 (OTG_DOEPMSK) 1262
37.15.38 OTG 设备全体端点中断寄存器 (OTG_HAINT) 1264
37.15.39 OTG 全体端点中断屏蔽寄存器 (OTG_DAINTMSK) 1264
37.15.40 OTG 设备 V_{BUS} 放电时间寄存器 (OTG_DVBUSDIS) 1265
37.15.41 OTG 设备 V_{BUS} 脉冲时间寄存器 (OTG_DVBUSPULSE) 1266
37.15.42 OTG 设备阈值控制寄存器 (OTG_DTHRCTL) 1266
37.15.43 OTG 设备单个端点中断寄存器 (OTG_DEACHINT) 1267
37.15.44 OTG 设备 IN 端点 FIFO 空中断屏蔽寄存器 (OTG_DIEPEMPMSK) 1267
37.15.45 OTG 设备单个端点中断屏蔽寄存器 (OTG_DEACHINTMSK) 1268
37.15.46 OTG 设备控制 IN 端点 0 控制寄存器 (OTG_DIEPCTL0) 1269
37.15.47 OTG 设备端点 x 控制寄存器 (OTG_DIEPCTLx) （x = 1..5[FS] / 0..7[HS]，其中 x = 端点编号） 1270
37.15.48 OTG 设备控制 OUT 端点 0 控制寄存器 (OTG_DOEPCTL0) 1272
37.15.49 OTG 设备端点 x 控制寄存器 (OTG_DOEPCTLx) （x = 1..5[FS] / 7[HS]，其中 x = 端点编号） 1274
37.15.50 OTG 设备端点 x 中断寄存器 (OTG_DIEPINTx) （x = 0..5[FS] / 7[HS]，其中 x = 端点编号） 1276
37.15.51 OTG 设备端点 x 中断寄存器 (OTG_DIEPINTx) （x = 0..5[FS] / 7[HS]，其中 x = 端点编号） 1278
37.15.52 OTG 设备 IN 端点 0 传输大小寄存器 (OTG_DIEPTSIZ0) 1279
37.15.53 OTG 设备 OUT 端点 0 传输大小寄存器 (OTG_DOEPTSIZ0) 1280
37.15.54 OTG 设备 IN 端点 x 传输大小寄存器 (OTG_DIEPTSIZx) （x = 1..5[FS] / 7[HS]，其中 x= 端点编号） 1281
37.15.55 OTG 设备 IN 端点发送 FIFO 状态寄存器 (OTG_DTXFSTSx) （x = 0..5[FS] / 7[HS]，其中 x = 端点编号） 1282
37.15.56 OTG 设备 OUT 端点 x 传输大小寄存器 (OTG_DOEPTSIZx) （x = 1..5[FS] / 7[HS]，其中 x = 端点编号） 1282
37.15.57 OTG 电源和时钟门控控制寄存器 (OTG_PCGCCTL) 1283
37.15.58 OTG_FS/OTG_HS 寄存器映射 1284
37.16 OTG_FS/OTG_HS 编程模型 1295
37.16.1 模块初始化 1295
37.16.2 主机初始化 1295
37.16.3 设备初始化 1296
37.16.4 DMA 模式 1296
37.16.5 主机编程模型 1296
37.16.6 设备编程模型 1324

37.16.7 最坏情况下的响应时间 1342
37.16.8 OTG 编程模型 1343

38 以太网 (ETH)：通过 DMA 控制器进行介质访问控制 (MAC) 1349
38.1 以太网简介 1349
38.2 以太网主要特性 1349
38.2.1 MAC 内核特性 1349
38.2.2 DMA 特性 1350
38.2.3 PTP 特性 1351
38.3 以太网引脚 1351
38.4 以太网功能说明：SMI、MII 和 RMII 1352
38.4.1 站管理接口：SMI 1353
38.4.2 介质独立接口：MII 1356
38.4.3 精简介质独立接口：RMII 1358
38.4.4 MII/RMII 选择 1359
38.5 以太网功能说明：MAC 802.3 1359
38.5.1 MAC 802.3 帧格式 1360
38.5.2 MAC 帧发送 1363
38.5.3 MAC 帧接收 1369
38.5.4 MAC 中断 1374
38.5.5 MAC 过滤 1374
38.5.6 MAC 回送模式 1376
38.5.7 MAC 管理计数器：MMC 1377
38.5.8 电源管理：PMT 1377
38.5.9 精密时间协议 (IEEE1588 PTP) 1380
38.6 以太网功能说明：DMA 控制器操作 1385
38.6.1 使用 DMA 进行传送的初始化 1386
38.6.2 主机总线突发访问 1386
38.6.3 主机数据缓冲区对齐 1387
38.6.4 缓冲区大小计算 1387
38.6.5 DMA 仲裁器 1387
38.6.6 对 DMA 的错误响应 1387
38.6.7 Tx DMA 配置 1388
38.6.8 Rx DMA 配置 1397
38.6.9 DMA 中断 1406
38.7 以太网中断 1407

38.8 以太网寄存器说明 1408
38.8.1 MAC 寄存器说明 1408
38.8.2 MMC 寄存器说明 1424
38.8.3 IEEE 1588 时间戳寄存器 1430
38.8.4 DMA 寄存器说明 1437
38.8.5 以太网寄存器映射 1450

39 HDMI-CEC 控制器 (HDMI-CEC) 1454
39.1 前言 1454
39.2 HDMI-CEC 控制器主要特性 1454
39.3 HDMI-CEC 功能说明 1455
39.3.1 HDMI-CEC 引脚 1455
39.3.2 消息说明 1456
39.3.3 位时序 1456
39.4 仲裁 1457
39.4.1 SFT 选项位 1458
39.5 错误处理 1459
39.5.1 位错误 1459
39.5.2 消息错误 1459
39.5.3 位上升错误 (BRE) 1459
39.5.4 短位周期错误 (SBPE) 1460
39.5.5 长位周期错误 (LBPE) 1460
39.5.6 发送错误检测 (TXERR) 1461
39.6 HDMI-CEC 中断 1463
39.7 HDMI-CEC 寄存器 1463
39.7.1 CEC 控制寄存器 (CEC_CR) 1463
39.7.2 CEC 配置寄存器 (CEC_CFGR) 1465
39.7.3 CEC 发送数据寄存器 (CEC_TXDR) 1467
39.7.4 CEC 接收数据寄存器 (CEC_RXDR) 1467
39.7.5 CEC 中断和状态寄存器 (CEC_ISR) 1468
39.7.6 CEC 中断使能寄存器 (CEC_IER) 1470
39.7.7 HDMI-CEC 寄存器映射 1472

40 调试支持 (DBG) 1473
40.1 概述 1473
40.2 ARM® 参考文档 1474

40.3 SWJ 调试端口（串行接口和 JTAG） 1474
40.3.1 JTAG-DP 或 SW-DP 的切换机制 1475
40.4 引脚排列和调试端口引脚 1475
40.4.1 SWJ 调试端口引脚 1476
40.4.2 灵活的 SWJ-DP 引脚分配 1476
40.4.3 JTAG 引脚上的内部上拉和下拉 1477
40.4.4 使用串行接口以及释放未使用的调试引脚以用作 GPIO 1477
40.5 STM32F75xxx 和 STM32F74xxx JTAG 调试端口连接 1478
40.6 ID 代码和锁定机制 1479
40.6.1 MCU 器件 ID 代码 1479
40.6.2 边界扫描调试端口 1479
40.6.3 Cortex®-M7 带 FPU 的 Cortex®-M7 的调试端口 1479
40.6.4 Cortex®-M7 带 FPU 的 Cortex®-M7 的 JEDEC-106 ID 代码 1480
40.7 JTAG 调试端口 1480
40.8 SW 调试端口 1482
40.8.1 SW 协议简介 1482
40.8.2 SW 协议序列 1482
40.8.3 SW-DP 状态机（复位、空闲状态、ID 代码） 1483
40.8.4 DP 和 AP 读/写访问 1483
40.8.5 SW-DP 寄存器 1484
40.8.6 SW-AP 寄存器 1484
40.9 AHB-AP（AHB 访问端口）——对 JTAG-DP 和 SW-DP 同时有效 . . . 1485
40.10 内核调试 1486
40.11 调试主机在系统复位状态下建立连接的功能 1486
40.12 FPB (Flash patch breakpoint) 1487
40.13 DWT（数据观察点触发） 1487
40.14 ITM（指令跟踪宏单元） 1487
40.14.1 一般说明 1487
40.14.2 时间戳数据包、同步和溢出数据包 1488
40.15 ETM（嵌入式跟踪宏单元） 1489
40.15.1 一般说明 1489
40.15.2 信号协议和数据包类型 1489
40.15.3 主要的 ETM 寄存器 1489
40.15.4 配置示例 1490

40.16 MCU 调试组件 (DBGMCU) 1490
40.16.1 对低功耗模式的调试支持 1490
40.16.2 对定时器、看门狗、bxCAN 和 I^2C 的调试支持 1491
40.16.3 MCU 调试配置寄存器 1491
40.16.4 DBGMCU_CR 寄存器 1491
40.16.5 MCU APB1 调试冻结寄存器 (DBGMCU_APB1_FZ) 1492
40.16.6 MCU APB2 调试冻结寄存器 (DBGMCU_APB2_FZ) 1494
40.17 Pelican TPIU（跟踪端口接口单元） 1495
40.17.1 前言 1495
40.17.2 TRACE 引脚分配 1496
40.17.3 TPIU 格式化器 1498
40.17.4 TPIU 帧同步数据包 1498
40.17.5 发送同步帧数据包 1498
40.17.6 同步模式 1499
40.17.7 异步模式 1499
40.17.8 STM32F75xxx 和 STM32F74xxx 内的 TRACECLKIN 连接 1499
40.17.9 TPIU 寄存器 1500
40.17.10 配置示例 1500
40.18 DBG 寄存器映射 1501

41 设备电子签名 1502
41.1 唯一设备 ID 寄存器（96 位） 1502
41.2 Flash 大小 1503
41.3 封装数据寄存器 1503

42 修订历史 1509

表格索引

表 1. STM32F75xxx 和 STM32F74xxx 寄存器边界地址 64
表 2. 自举模式 69
表 3. Flash 构成 71
表 4. CPU 时钟 (HCLK) 频率对应的等待周期数 72
表 5. 编程/擦除并行位数 74
表 6. Flash 中断请求 76
表 7. 选项字节构成 77
表 8. 不同读保护级别下的访问限制 81
表 9. OTP 区域构成 82
表 10. Flash 寄存器映射与复位值 90
表 11. 调压器配置模式与器件工作模式 97
表 12. 低功耗模式汇总 102
表 13. 所有模式下的特性 103
表 14. 进入和退出立即休眠 106
表 15. 进入和退出退出时休眠 107
表 16. 停止工作模式 108
表 17. 进入和退出停止模式（STM32F75xxx 和 STM32F74xxx） 109
表 18. 进入和退出待机模式 110
表 19. PWR - 寄存器映射和复位值 120
表 20. RCC 寄存器映射和复位值 178
表 21. 端口位配置表 184
表 22. GPIO 寄存器映射和复位值 197
表 23. SYSCFG 寄存器映射和复位值 205
表 24. DMA1 请求映射 209
表 25. DMA2 请求映射 209
表 26. 源和目标地址 210
表 27. 双缓冲区模式下的源和目标地址寄存器 (DBM=1) 215
表 28. 封装/解封和字节序行为（位 PINC = MINC = 1） 216
表 29. PSIZE 与 MSIZE 确定时对 NDT 的限制条件 216
表 30. FIFO 阈值配置 218
表 31. 可能的 DMA 配置 222
表 32. DMA 中断请求 225
表 33. DMA 寄存器映射和复位值 236
表 34. 输入时支持的颜色模式 242
表 35. 存储器中的数据顺序 243
表 36. Alpha 模式配置 244
表 37. 支持的 CLUT 颜色模式 245
表 38. 存储器中的 CLUT 数据顺序 245
表 39. 输出时支持的颜色模式 246
表 40. 存储器中的数据顺序 246
表 41. DMA2D 中断请求 250
表 42. DMA2D 寄存器映射和复位值 266
表 43. STM32F75xxx 和 STM32F74xxx 向量表 268
表 44. 外部中断/事件控制器寄存器映射和复位值 279
表 45. CRC 寄存器映射和复位值 285
表 46. NOR/PSRAM 存储区域选择 291
表 47. NOR/PSRAM 外部存储器地址 291
表 48. NAND 存储映射和时序寄存器 291

表 49. NAND 存储区域选择 . . . 292
表 50. SDRAM 存储区域选择 . . . 292
表 51. SDRAM 地址映射 . . . 292
表 52. 8 位数据总线宽度时的 SDRAM 地址映射 . . . 293
表 53. 16 位数据总线宽度时的 SDRAM 地址映射 . . . 294
表 54. 32 位数据总线宽度时的 SDRAM 地址映射 . . . 294
表 55. NOR/PSRAM 的可编程访问参数 . . . 296
表 56. 非复用 I/O NOR Flash . . . 297
表 57. 16 位复用 I/O NOR Flash . . . 297
表 58. 非复用 I/O PSRAM/SRAM . . . 297
表 59. 16 位复用 I/O PSRAM . . . 298
表 60. NOR Flash/PSRAM：支持的存储器和事务示例 . . . 299
表 61. FMC_BCRx 位字段 . . . 302
表 62. FMC_BTRx 位字段 . . . 302
表 63. FMC_BCRx 位字段 . . . 304
表 64. FMC_BTRx 位字段 . . . 304
表 65. FMC_BWTRx 位字段 . . . 305
表 66. FMC_BCRx 位字段 . . . 307
表 67. FMC_BTRx 位字段 . . . 307
表 68. FMC_BWTRx 位字段 . . . 308
表 69. FMC_BCRx 位字段 . . . 309
表 70. FMC_BTRx 位字段 . . . 310
表 71. FMC_BWTRx 位字段 . . . 310
表 72. FMC_BCRx 位字段 . . . 312
表 73. FMC_BTRx 位字段 . . . 312
表 74. FMC_BWTRx 位字段 . . . 313
表 75. FMC_BCRx 位字段 . . . 314
表 76. FMC_BTRx 位字段 . . . 315
表 77. FMC_BCRx 位字段 . . . 320
表 78. FMC_BTRx 位字段 . . . 321
表 79. FMC_BCRx 位字段 . . . 322
表 80. FMC_BTRx 位字段 . . . 323
表 81. 可编程的 NAND Flash 访问参数 . . . 330
表 82. 8 位 NAND Flash . . . 331
表 83. 16 位 NAND Flash . . . 331
表 84. 支持的存储器和事务 . . . 332
表 85. ECC 结果相关位 . . . 340
表 86. SDRAM 信号 . . . 341
表 87. FMC 寄存器映射 . . . 358
表 88. QUADSPI 中断请求 . . . 373
表 89. QUADSPI 寄存器映射和复位值 . . . 387
表 90. ADC 引脚 . . . 390
表 91. 模拟看门狗通道选择 . . . 396
表 92. 配置触发极性 . . . 399
表 93. 常规通道的外部触发 . . . 400
表 94. 注入通道的外部触发 . . . 400
表 95. ADC 中断 . . . 414
表 96. ADC 全局寄存器映射 . . . 429
表 97. 每个 ADC 的 ADC 寄存器映射和复位值 . . . 429
表 98. ADC 寄存器映射和复位值（通用 DAC 寄存器） . . . 430
表 99. DAC 引脚 . . . 432
表 100. 外部触发器 . . . 435

表 101. DAC 寄存器映射 . . . 449
表 102. DCMI 引脚 . . . 451
表 103. DCMI 信号 . . . 453
表 104. 捕获的数据字节在 32 位字（宽 8 位）中的位置排布 . . . 454
表 105. 捕获的数据字节在 32 位字（宽 10 位）中的位置排布 . . . 454
表 106. 捕获的数据字节在 32 位字（宽 12 位）中的位置排布 . . . 454
表 107. 捕获的数据字节在 32 位字（宽 14 位）中的位置排布 . . . 455
表 108. 单色逐行视频格式的数据存储 . . . 460
表 109. 以 RGB 逐行视频格式存储数据 . . . 460
表 110. YCbCr 逐行视频格式下的数据存储 . . . 460
表 111. YCbCr 逐行视频格式下的数据存储——Y 分量提取模式 . . . 461
表 112. DCMI 中断 . . . 461
表 113. DCMI 寄存器地址映射和复位值 . . . 470
表 114. LCD-TFT 引脚和信号接口 . . . 474
表 115. 像素数据映射与颜色格式的关系 . . . 478
表 116. LTDC 中断请求 . . . 481
表 117. LTDC 寄存器映射和复位值 . . . 501
表 118. RNG 寄存器映射和复位映射 . . . 508
表 119. 处理 128 位块所需的周期数 . . . 509
表 120. 数据类型 . . . 526
表 121. CRYP 寄存器映射和复位值 . . . 545
表 122. HASH 寄存器映射和复位值 . . . 564
表 123. 定时器输出行为与 BRK/BRK2 输入 . . . 603
表 124. 计数方向与编码器信号的关系 . . . 608
表 125. TIMx 内部触发连接 . . . 623
表 126. 具有断路功能的互补通道 OCx 和 OCxN 的输出控制位 . . . 635
表 127. TIM1 寄存器映射和复位值 . . . 645
表 128. TIM8 寄存器映射和复位值 . . . 647
表 129. 计数方向与编码器信号的关系 . . . 681
表 130. TIMx 内部触发连接 . . . 697
表 131. 标准 OCx 通道的输出控制位 . . . 708
表 132. TIM2/TIM3/TIM4/TIM5 寄存器映射和复位值 . . . 714
表 133. TIMx 内部触发连接 . . . 740
表 134. 标准 OCx 通道的输出控制位 . . . 747
表 135. TIM9/12 寄存器映射和复位值 . . . 749
表 136. 标准 OCx 通道的输出控制位 . . . 756
表 137. TIM10/11/13/14 寄存器映射和复位值 . . . 759
表 138. TIM6/TIM7 寄存器映射和复位值 . . . 773
表 139. STM32F75xxx 和 STM32F74xxx LPTIM 特性 . . . 774
表 140. 预分频器的分频比 . . . 776
表 141. 编码器计数方案 . . . 782
表 142. LPTIM 外部触发连接 . . . 788
表 143. LPTIM 寄存器映射和复位值 . . . 793
表 144. IWDG 寄存器映射和复位值 . . . 800
表 145. WWDG 寄存器映射和复位值 . . . 806
表 146. RTC 引脚 PC13 配置 . . . 809
表 147. RTC 引脚 PI8 配置 . . . 810
表 148. RTC 引脚 PC2 配置 . . . 811
表 149. 各个模式下的 RTC 功能 . . . 811
表 150. 低功耗模式对 RTC 的作用 . . . 822
表 151. 中断控制位 . . . 823
表 152. RTC 寄存器映射和复位值 . . . 844

表 153. STM32F75xxx 和 STM32F74xxx I2C 特性实现 . . . 847
表 154. I2C-SMBUS 规范数据建立和保持时间 . . . 852
表 155. I2C 配置表 . . . 856
表 156. I2C-SMBUS 规范时钟时序 . . . 865
表 157. SMBus 超时规范 . . . 877
表 158. 带 PEC 的 SMBUS 配置 . . . 879
表 159. 不同 I2CCLK 频率下的 TIMEOUTA 设置示例（最大 $t_{TIMEOUT}$ = 25 ms） . . . 880
表 160. 不同 I2CCLK 频率下的 TIMEOUTB 设置示例 . . . 880
表 161. 不同 I2CCLK 频率下的 TIMEOUTA 设置示例（最大 t_{IDLE} = 50 µs） . . . 880
表 162. 低功耗模式 . . . 889
表 163. I2C 中断请求 . . . 889
表 164. I2C 寄存器映射和复位值 . . . 905
表 165. STM32F75xxx 和 STM32F74xxx USART 特性 . . . 908
表 166. 通过采样数据进行噪声检测 . . . 919
表 167. 采用 8 倍过采样 (OVER8 = 1) 时，在 fCK = 216 MHz 下编程波特率时的误差计算 . . . 922
表 168. 采用 16 倍过采样 (OVER8 = 0) 时，在 fCK = 216 MHz 下编程波特率时的误差计算 . . . 922
表 169. BRR [3:0] = 0000 时的 USART 接收器容差 . . . 923
表 170. BRR[3:0] 不等于 0000 时的 USART 接收器容差 . . . 923
表 171. 帧格式 . . . 927
表 172. 低功耗模式对 USART 的影响 . . . 943
表 173. USART 中断请求 . . . 943
表 174. USART 寄存器映射和复位值 . . . 963
表 175. STM32F75xxx 和 STM32F74xxx SPI 实现 . . . 966
表 176. SPI 中断请求 . . . 989
表 177. 使用标准 8 MHz HSE 时的音频频率精度 . . . 1000
表 178. I^2S 中断请求 . . . 1005
表 179. SPI 寄存器映射和复位值 . . . 1017
表 180. 外部同步选择 . . . 1021
表 181. 可能的音频采样范围示例 . . . 1027
表 182. SOPD 模式 . . . 1033
表 183. 奇偶校验位计算 . . . 1033
表 184. 音频采样频率与符号率 (SHARK) . . . 1034
表 185. SAI 中断源 . . . 1042
表 186. SAI 寄存器映射和复位值 . . . 1055
表 187. 报头的转换序列 . . . 1063
表 188. SPDIFRX_CLK 的最小频率与音频采样率 . . . 1073
表 189. 位字段属性与 SPDIFRX 状态 . . . 1075
表 190. SPDIFRX 接口寄存器映射和复位值 . . . 1088
表 191. SDMMC I/O 定义 . . . 1092
表 192. 命令格式 . . . 1097
表 193. 短响应格式 . . . 1098
表 194. 长响应格式 . . . 1098
表 195. 命令路径状态标志 . . . 1098
表 196. 数据令牌格式 . . . 1101
表 197. DPSM 标志 . . . 1101
表 198. 传输 FIFO 状态标志 . . . 1102
表 199. 接收 FIFO 状态标志 . . . 1103
表 200. 卡状态 . . . 1112
表 201. SD 状态 . . . 1114
表 202. 速度等级代码字段 . . . 1115
表 203. 移动性能字段 . . . 1115
表 204. AU_SIZE 字段 . . . 1116

表 205. 最大 AU 大小 1116
表 206. 擦除大小字段 1116
表 207. 擦除超时字段 1117
表 208. 擦除偏移字段 1117
表 209. 面向块的写入命令 1119
表 210. 面向块的写保护命令 1120
表 211. 擦除命令 1120
表 212. I/O 模式命令 1120
表 213. 锁定卡 1121
表 214. 应用程序特定的命令 1121
表 215. R1 响应 1122
表 216. R2 响应 1122
表 217. R3 响应 1123
表 218. R4 响应 1123
表 219. R4b 响应 1123
表 220. R5 响应 1124
表 221. R6 响应 1124
表 222. 响应类型和 SDMMC_RESPx 寄存器 1131
表 223. SDMMC 寄存器映射 1141
表 224. 发送邮箱映射 1156
表 225. 接收邮箱映射 1156
表 226. bxCAN 寄存器映射和复位值 1181
表 227. STM32F75xxx 的 USB_OTG 实现 1187
表 228. 模块全局控制和状态寄存器 (CSR) 1208
表 229. 主机模式控制和状态寄存器 (CSR) 1209
表 230. 设备模式控制和状态寄存器 1211
表 231. 数据 FIFO (DFIFO) 访问寄存器地址映射 1213
表 232. 电源和时钟门控控制和状态寄存器 1213
表 233. 软断开的最小时间 1260
表 234. OTG_FS/OTG_HS 寄存器映射和复位值 1284
表 235. 复用功能映射 1351
表 236. 管理帧格式 1353
表 237. 时钟范围 1355
表 238. TX 接口信号编码 1357
表 239. RX 接口信号编码 1357
表 240. 帧状态 1371
表 241. 目标地址过滤 1376
表 242. 源地址过滤 1376
表 243. 接收描述符 0 - 位 7、5 和 0 的编码（仅正常描述符格式，EDFE=0） 1402
表 244. 时间戳快照对寄存器位的相依性 1432
表 245. 以太网寄存器映射和复位值 1450
表 246. HDMI 引脚 1455
表 247. 错误处理时序参数 1461
表 248. TXERR 时序参数 1462
表 249. HDMI-CEC 中断 1463
表 250. HDMI-CEC 寄存器映射和复位值 1472
表 251. SWJ 调试端口引脚 1476
表 252. 灵活的 SWJ-DP 引脚分配 1476
表 253. JTAG 调试端口数据寄存器 1480
表 254. 32 位调试端口寄存器，通过移位值 A[3:2] 进行寻址 1481
表 255. 数据包请求（8 位） 1482
表 256. ACK 响应（3 位） 1482

表 257. DATA 传输（33 位） 1483
表 258. SW-DP 寄存器 1484
表 259. Cortex®-M7 带 FPU 的 Cortex®-M7 的 AHB-AP 寄存器 1485
表 260. 内核调试寄存器 1486
表 261. 主要的 ITM 寄存器 1488
表 262. 异步 TRACE 引脚分配 1496
表 263. 同步 TRACE 引脚分配 1496
表 264. 灵活的 TRACE 引脚分配 1497
表 265. 重要的 TPIU 寄存器 1500
表 266. DBG 寄存器映射和复位值 1501
表 267. 文档版本历史 1509
表. 中文文档版本历史 1510

图片索引

图 1. STM32F75xxx 和 STM32F74xxx 器件的系统架构 60
图 2. 系统架构内的 Flash 接口连接（STM32F75xxx 和 STM32F74xxx） 70
图 3. RDP 级别 81
图 4. 电源概述 92
图 5. 连接到 VDD 电源的 VDDUSB 93
图 6. 连接到外部独立电源的 VDDUSB 94
图 7. 备份域 96
图 8. 上电复位/掉电复位波形 99
图 9. BOR 阈值 100
图 10. PVD 阈值 101
图 11. 复位电路简图 122
图 12. 时钟树 123
图 13. HSE/LSE 时钟源 125
图 14. TIM5 在输入捕获模式下的频率测量 130
图 15. TIM11 在输入捕获模式下的频率测量 130
图 16. I/O 端口位的基本结构 183
图 17. 5 V 容限 I/O 端口位的基本结构 183
图 18. 输入浮空/上拉/下拉配置 188
图 19. 输出配置 189
图 20. 复用功能配置 189
图 21. 高阻态模拟配置 190
图 22. DMA 框图 207
图 23. 通道选择 208
图 24. 外设到存储器模式 211
图 25. 存储器到外设模式 212
图 26. 存储器到存储器模式 213
图 27. FIFO 结构 218
图 28. DMA2D 框图 241
图 29. 外部中断/事件控制器框图 273
图 30. 外部中断/事件 GPIO 映射 275
图 31. CRC 计算单元框图 280
图 32. FMC 框图 287
图 33. FMC 存储区域 290
图 34. 模式 1 读取访问波形 301
图 35. 模式 1 写入访问波形 301
图 36. 模式 A 读取访问波形 303
图 37. 模式 A 读取访问波形 303
图 38. 模式 2 和模式 B 读取访问波形 305
图 39. 模式 2 读取访问波形 306
图 40. 模式 B 读取访问波形 306
图 41. 模式 C 读取访问波形 308
图 42. 模式 C 读取访问波形 309
图 43. ModeD 读取访问波形 311
图 44. 模式 D 读取访问波形 311
图 45. 复用读取访问波形 313
图 46. 复用写入访问波形 314
图 47. 读取访问波形期间的异步等待 316

图 48. 写入访问波形期间的异步等待 . . . 317
图 49. 等待配置波形 . . . 319
图 50. 同步复用读取模式波形 - NOR、PSRAM (CRAM) . . . 320
图 51. 同步复用写入模式波形 - PSRAM (CRAM) . . . 322
图 52. NAND Flash 控制器的通用存储器访问波形 . . . 333
图 53. 访问非“CE 无关”NAND-Flash . . . 334
图 54. 突发写入 SDRAM 访问波形 . . . 343
图 55. 突发读 SDRAM 访问 . . . 344
图 56. RBURST 位置 1 时的读访问逻辑图（CAS=1，RPIPE=0） . . . 345
图 57. 跨行边界的读访问 . . . 346
图 58. 跨行边界的写访问 . . . 347
图 59. 自刷新模式 . . . 349
图 60. 掉电模式 . . . 350
图 61. QUADSPI 功能框图（双闪存模式禁止） . . . 360
图 62. QUADSPI 功能框图（双闪存模式使能） . . . 361
图 63. 四线模式下的读命令示例 . . . 362
图 64. 四线模式下 DDR 命令示例 . . . 365
图 65. CKMODE = 0 时的 nCS（T = CLK 周期） . . . 371
图 66. SDR 模式下 CKMODE = 1 时的 nCS（T = CLK 周期） . . . 372
图 67. DDR 模式下 CKMODE = 1 时的 nCS（T = CLK 周期） . . . 372
图 68. CKMODE = 1 且发生中止时的 nCS（T = CLK 周期） . . . 372
图 69. 单个 ADC 框图 . . . 389
图 70. ADC1连接功能 . . . 391
图 71. ADC2连接功能 . . . 392
图 72. ADC3连接功能 . . . 393
图 73. 时序图 . . . 395
图 74. 模拟看门狗的保护区域 . . . 396
图 75. 注入转换延迟 . . . 397
图 76. 12 位数据的右对齐 . . . 398
图 77. 12 位数据的左对齐 . . . 399
图 78. 6 位数据的左对齐 . . . 399
图 79. 多重 ADC 框图(1) . . . 403
图 80. 4 通道的注入同步模式：双重 ADC 模式 . . . 406
图 81. 4 通道的注入同步模式：三重 ADC 模式 . . . 407
图 82. 16 通道的常规同步模式：双重 ADC 模式 . . . 407
图 83. 16 通道的常规同步模式：三重 ADC 模式 . . . 408
图 84. 连续转换模式下 1 通道的交替模式：双重 ADC 模式 . . . 408
图 85. 连续转换模式下 1 通道的交替模式：三重 ADC 模式 . . . 409
图 86. 交替触发：各个 ADC 的注入组 . . . 410
图 87. 交替触发：不连续采样模式下的 4 个注入通道（各个 ADC） . . . 410
图 88. 交替触发：各个 ADC 的注入组 . . . 411
图 89. 交替 + 常规同步 . . . 412
图 90. 在注入转换期间出现的触发事件 . . . 412
图 91. 温度传感器和 VREFINT 通道框图 . . . 413
图 92. DAC 通道框图 . . . 432
图 93. DAC 输出缓冲器连接 . . . 433
图 94. DAC 单通道模式下的数据寄存器 . . . 433
图 95. DAC 双通道模式下的数据寄存器 . . . 434
图 96. 关闭触发 (TEN = 0) 时的转换时序图 TEN = 0 . . . 434
图 97. DAC LFSR 寄存器计算算法 . . . 436
图 98. LFSR 产生波形的 DAC 转换（使能软件触发） . . . 436
图 99. 生成 DAC 三角波 . . . 437

图 100. 生成三角波波形的 DAC 转换（使能软件触发）. 437
图 101. DCMI 框图 452
图 102. 顶级框图 452
图 103. DCMI 信号波形 453
图 104. 时序图 455
图 105. 快照模式下的帧捕获波形 457
图 106. 连续采集模式下的帧捕获波形 457
图 107. 裁剪后窗口的坐标和大小 458
图 108. 数据捕获波形 458
图 109. 像素光栅扫描顺序 459
图 110. LTDC 框图 473
图 111. LCD-TFT 同步时序 475
图 112. 层窗口可编程参数 477
图 113. 两层与背景混合 480
图 114. 中断事件 481
图 115. 框图 505
图 116. 框图 510
图 117. DES/TDES-ECB 模式加密 512
图 118. DES/TDES-ECB 模式解密 513
图 119. DES/TDES-CBC 模式加密 514
图 120. DES/TDES-CBC 模式解密 515
图 121. AES-ECB 模式加密 516
图 122. AES-ECB 模式解密 517
图 123. AES-CBC 模式加密 518
图 124. AES-CBC 模式解密 519
图 125. AES-CTR 模式加密 520
图 126. AES-CTR 模式解密 521
图 127. 计数器模式下的初始计数器块结构 521
图 128. 根据 DATATYPE 构建 64 位块 527
图 129. TDES-CBC 加密过程中使用的初始化向量 529
图 130. 框图 549
图 131. 位、字节和半字交换 551
图 132. 散列中断映射图 556
图 133. 高级控制定时器框图 567
图 134. 预分频器分频由 1 变为 2 时的计数器时序图 569
图 135. 预分频器分频由 1 变为 4 时的计数器时序图 569
图 136. 计数器时序图，1 分频内部时钟 570
图 137. 计数器时序图，2 分频内部时钟 571
图 138. 计数器时序图，4 分频内部时钟 571
图 139. 计数器时序图，N 分频内部时钟 572
图 140. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载） 572
图 141. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 已预装载） 573
图 142. 计数器时序图，1 分频内部时钟 574
图 143. 计数器时序图，2 分频内部时钟 574
图 144. 计数器时序图，4 分频内部时钟 575
图 145. 计数器时序图，N 分频内部时钟 575
图 146. 计数器时序图，未使用重复计数器时更新事件 576
图 147. 计数器时序图，1 分频内部时钟，TIMx_ARR = 0x6 577
图 148. 计数器时序图，2 分频内部时钟 578
图 149. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36 578
图 150. 计数器时序图，N 分频内部时钟 579
图 151. 计数器时序图，ARPE=1 时的更新事件（计数器下溢） 579

图 152. 计数器时序图，ARPE=1 时的更新事件（计数器上溢）. 580
图 153. 不同模式和 TIMx_RCR 寄存器设置下的更新频率示例. 581
图 154. 外部触发输入模块. 581
图 155. 正常模式下的控制电路，1 分频内部时钟 582
图 156. TI2 外部时钟连接示例. 583
图 157. 外部时钟模式 1 下的控制电路. 584
图 158. 外部触发输入模块. 584
图 159. 外部时钟模式 2 下的控制电路. 585
图 160. 捕获/比较通道（例如：通道 1 输入阶段） 586
图 161. 捕获/比较通道 1 主电路 586
图 162. 捕获/比较通道的输出阶段（通道 1、 通道 2 和通道 3） 587
图 163. 捕获/比较通道的输出阶段（通道 4） 587
图 164. 捕获/比较通道的输出阶段（通道 5 和通道 6） 6) 588
图 165. PWM 输入模式时序 590
图 166. 输出比较模式，翻转 OC1. 591
图 167. 边沿对齐模式的 PWM 波形 (ARR=8) 593
图 168. 中心对齐模式 PWM 波形 (ARR=8). 594
图 169. 50% 占空比时产生的 2 个相移 PWM 信号 595
图 170. 通道 1 和通道 3 上的组合 PWM 模式 596
图 171. 三相组合 PWM 信号（每个周期多个触发脉冲） 597
图 172. 带死区插入的互补输出 598
图 173. 延迟时间大于负脉冲宽度的死区波形 598
图 174. 延迟时间大于正脉冲宽度的死区波形 598
图 175. 断路和断路 2 电路概述 600
图 176. 响应 BRK 上的断路事件的不同输出行为 (OSSI = 1) 602
图 177. BRK 和 BRK2 引脚使能后的 PWM 输出状态 (OSSI=1) 603
图 178. BRK 使能后的 PWM 输出状态 (OSSI=0) 603
图 179. 清除 TIMx 的 OCxREF 604
图 180. COM 事件生成 6 步 PWM 的示例 (OSSR=1). 605
图 181. 单脉冲模式示例. 606
图 182. 可再触发单脉冲模式 607
图 183. 编码器接口模式下的计数器工作示例 608
图 184. TI1FP1 极性反相时的编码器接口模式示例. 609
图 185. 测量 3 个信号上边沿之间的时间间隔 610
图 186. 霍尔传感器接口的示例 611
图 187. 复位模式下的控制电路 612
图 188. 门控模式下的控制电路 613
图 189. 触发模式下的控制电路 614
图 190. 外部时钟模式 2 + 触发模式下的控制电路. 615
图 191. 通用定时器框图. 651
图 192. 预分频器分频由 1 变为 2 时的计数器时序图. 653
图 193. 预分频器分频由 1 变为 4 时的计数器时序图. 653
图 194. 计数器时序图，1 分频内部时钟 654
图 195. 计数器时序图，2 分频内部时钟 655
图 196. 计数器时序图，4 分频内部时钟 655
图 197. 计数器时序图，N 分频内部时钟 656
图 198. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载） 656
图 199. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 已预装载） 657
图 200. 计数器时序图，1 分频内部时钟 658
图 201. 计数器时序图，2 分频内部时钟 658
图 202. 计数器时序图，4 分频内部时钟 659
图 203. 计数器时序图，N 分频内部时钟 659

图 204. 计数器时序图，使用重复计数器时更新事件 660
图 205. 计数器时序图，1 分频内部时钟，TIMx_ARR=0x6 661
图 206. 计数器时序图，2 分频内部时钟 662
图 207. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36 662
图 208. 计数器时序图，N 分频内部时钟 663
图 209. 计数器时序图，ARPE=1 时的更新事件（计数器下溢） 663
图 210. 计数器时序图，ARPE=1 时的更新事件（计数器上溢） 664
图 211. 正常模式下的控制电路，1 分频内部时钟 665
图 212. TI2 外部时钟连接示例 665
图 213. 外部时钟模式 1 下的控制电路 666
图 214. 外部触发输入模块 667
图 215. 外部时钟模式 2 下的控制电路 668
图 216. 捕获/比较通道（例如：通道 1 输入阶段） 669
图 217. 捕获/比较通道 1 主电路 669
图 218. 捕获/比较通道的输出阶段（通道 1） 670
图 219. PWM 输入模式时序 672
图 220. 输出比较模式，翻转 OC1 673
图 221. 边沿对齐模式的 PWM 波形 (ARR=8) 674
图 222. 中心对齐模式 PWM 波形 (ARR=8) 676
图 223. 50% 占空比时产生的 2 个相移 PWM 信号 677
图 224. 通道 1 和通道 3 上的组合 PWM 模式 678
图 225. 清除 TIMx 的 OCxREF 679
图 226. 单脉冲模式示例 680
图 227. 编码器接口模式下的计数器工作示例 682
图 228. TI1FP1 极性反相时的编码器接口模式示例 682
图 229. 复位模式下的控制电路 684
图 230. 门控模式下的控制电路 685
图 231. 触发模式下的控制电路 685
图 232. 外部时钟模式 2 + 触发模式下的控制电路 686
图 233. 主/从定时器示例 687
图 234. 使用 TIM3 的 OC1REF 对 TIM 实施门控控制 688
图 235. 使用 TIM3 的使能信号对 TIM 实施门控控制 689
图 236. 使用 TIM3 的更新事件触发 TIM 689
图 237. 使用 TIM3 的使能信号触发 TIM 690
图 238. 使用 TIM3 的 TI1 输入触发 TIM3 和 TIM2 691
图 239. 通用定时器框图（TIM9 和 TIM12） 718
图 240. 通用定时器框图 (TIM10/11/13/14) 719
图 241. 预分频器分频由 1 变为 2 时的计数器时序图 720
图 242. 预分频器分频由 1 变为 4 时的计数器时序图 721
图 243. 计数器时序图，1 分频内部时钟 722
图 244. 计数器时序图，2 分频内部时钟 722
图 245. 计数器时序图，4 分频内部时钟 723
图 246. 计数器时序图，N 分频内部时钟 723
图 247. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载） 724
图 248. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 未预装载） 724
图 249. 正常模式下的控制电路，1 分频内部时钟 725
图 250. TI2 外部时钟连接示例 726
图 251. 外部时钟模式 1 下的控制电路 727
图 252. 捕获/比较通道（例如：通道 1 输入阶段） 727
图 253. 捕获/比较通道 1 主电路 728
图 254. 捕获/比较通道的输出阶段（通道 1） 728
图 255. PWM 输入模式时序 730

图 256. 输出比较模式，翻转 OC1 . . . 732
图 257. 边沿对齐模式的 PWM 波形 (ARR=8) . . . 733
图 258. 单脉冲模式示例： . . . 734
图 259. 复位模式下的控制电路 . . . 735
图 260. 门控模式下的控制电路 . . . 736
图 261. 触发模式下的控制电路 . . . 737
图 262. 基本定时器框图 . . . 761
图 263. 预分频器分频由 1 变为 2 时的计数器时序图 . . . 763
图 264. 预分频器分频由 1 变为 4 时的计数器时序图 . . . 763
图 265. 计数器时序图，1 分频内部时钟 . . . 764
图 266. 计数器时序图，2 分频内部时钟 . . . 765
图 267. 计数器时序图，4 分频内部时钟 . . . 765
图 268. 计数器时序图，N 分频内部时钟 . . . 766
图 269. 计数器时序图，ARPE = 0 时更新事件（TIMx_ARR 未预装载） . . . 766
图 270. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 未预装载） . . . 767
图 271. 正常模式下的控制电路，1 分频内部时钟 . . . 768
图 272. 低功耗定时器框图 . . . 775
图 273. 干扰滤波器时序图 . . . 776
图 274. 单次计数模式配置时的 LPTIM 输出波形 . . . 777
图 275. 单次计数模式配置及激活置 1 一次模式时的 LPTIM 输出波形（WAVE 位置 1） . . . 778
图 276. 连续计数模式配置时的 LPTIM 输出波形 . . . 778
图 277. 生成波形 . . . 780
图 278. 编码器模式计数序列 . . . 782
图 279. 独立看门狗框图 . . . 794
图 280. 看门狗框图 . . . 802
图 281. 窗口看门狗时序图 . . . 803
图 282. RTC 框图 . . . 808
图 283. I2C 框图 . . . 848
图 284. I2C 总线协议 . . . 850
图 285. 建立和保持时序 . . . 851
图 286. I2C 初始化流程图 . . . 853
图 287. 数据接收 . . . 854
图 288. 数据发送 . . . 855
图 289. 从器件初始化流程图 . . . 858
图 290. I2C 从发送器的传输序列流程图 (NOSTRETCH=0) . . . 859
图 291. I2C 从发送器的传输序列流程图 (NOSTRETCH=1) . . . 860
图 292. I2C 从发送器的传输总线图 . . . 861
图 293. 从接收器的传输序列流程图 (NOSTRETCH=0) . . . 862
图 294. 从接收器的传输序列流程图 (NOSTRETCH=1) . . . 863
图 295. I2C 从接收器的传输总线图 . . . 863
图 296. 主时钟生成 . . . 865
图 297. 主模式初始化流程图 . . . 867
图 298. 10 位地址读访问 (HEAD10R=0) . . . 867
图 299. 10 位地址读访问 (HEAD10R=1) . . . 868
图 300. I2C 主发送器的传输序列流程图（N≤255 字节） . . . 869
图 301. I2C 主发送器的传输序列流程图（N>255 字节） . . . 870
图 302. I2C 主发送器的传输总线图 . . . 871
图 303. I2C 主接收器的传输序列流程图（N≤255 字节） . . . 873
图 304. I2C 主接收器的传输序列流程图（N>255 字节） . . . 874
图 305. I2C 主接收器的传输总线图 . . . 875
图 306. $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的超时间隔 . . . 878

图 307. SMBus 从发送器的传输序列流程图（N 字节 + PEC）. 881
图 308. SMBus 从发送器的传输总线图 (SBC=1) 882
图 309. SMBus 从接收器的传输序列流程图（N 字节 + PEC）. 883
图 310. SMBus 从接收器的总线传输图 (SBC=1) 884
图 311. SMBus 主发送器的总线传输图 885
图 312. SMBus 主接收器的总线传输图 886
图 313. I2C 中断映射图 890
图 314. USART 框图 910
图 315. 字长编程 912
图 316. 可配置的停止位. 913
图 317. 发送时的 TC/TXE 行为 915
图 318. 16 倍或 8 倍过采样时的起始位检测 916
图 319. 16 倍过采样时的数据采样. 919
图 320. 8 倍过采样时的数据采样. 919
图 321. 使用空闲线路检测时的静默模式 925
图 322. 使用地址标记检测时的静默模式 926
图 323. LIN 模式下的断路检测（11 位断路长度 —— LBDL 位置 1）. 929
图 324. LIN 模式下的断路检测与帧错误检测. 930
图 325. USART 同步发送示例. 931
图 326. USART 数据时钟时序图 (M 位 = 00) 931
图 327. USART 数据时钟时序图 (M 位 = 01) 932
图 328. RX 数据建立/保持时间 932
图 329. ISO 7816-3 异步协议 934
图 330. 使用 1.5 个停止位检测奇偶校验错误 935
图 331. IrDA SIR ENDEC —— 框图 938
图 332. IrDA 数据调制 (3/16) —— 正常模式 939
图 333. 使用 DMA 进行发送 940
图 334. 使用 DMA 进行接收 941
图 335. 2 个 USART 间的硬件流控制 941
图 336. RS232 RTS 流控制. 942
图 337. RS232 CTS 流控制. 942
图 338. USART 中断映射图. 944
图 339. SPI 框图 967
图 340. 全双工单个主器件/单个从器件应用 968
图 341. 半双工单个主器件/单个从器件应用 969
图 342. 单工单个主器件/单个从器件应用（主器件为只发送模式/从器件为只接收模式） 969
图 343. 主器件和三个独立的从器件. 970
图 344. 硬件/软件从器件选择管理. 971
图 345. 数据时钟时序图. 973
图 346. 数据长度不等于 8 位或 16 位时的数据对齐 974
图 347. 将数据封装到 FIFO 中以进行发送和接收 977
图 348. 主器件全双工通信. 980
图 349. 从器件全双工通信. 981
图 350. 采用 CRC 的主器件全双工通信 982
图 351. 封装模式下的主器件全双工通信 983
图 352. Motorola SPI 主模式下的 NSSP 脉冲生成 986
图 353. TI 模式传输 987
图 354. I^2S 框图 990
图 355. I^2S Philips 协议波形（16/32 位全精度） 992
图 356. I^2S Philips 标准波形（24 位帧） 992
图 357. 发送 0x8EAA33. 993
图 358. 接收 0x8EAA33. 993

图 359. I^2S Philips 标准波形（16 位扩展到 32 位数据包帧）. 993
图 360. 16 位数据帧扩展到 32 位通道帧的示例 993
图 361. MSB 对齐的 16 位或 32 位全精度长度 994
图 362. MSB 对齐的 24 位帧长度 994
图 363. MSB 对齐的 16 位数据帧扩展为 32 位数据帧. 994
图 364. LSB 对齐的 16 位或 32 位全精度 995
图 365. LSB 对齐的 24 位帧长度. 995
图 366. 发送 0x3478AE 所需的操作 995
图 367. 接收 0x3478AE 时所需的操作 996
图 368. LSB 对齐的 16 位数据帧扩展为 32 位数据帧 996
图 369. 16 位数据帧扩展到 32 位通道帧的示例 996
图 370. PCM 标准波形（16 位）. 997
图 371. PCM 标准波形（16 位扩展到 32 位数据包帧） 997
图 372. 在主模式下启动通信序列 998
图 373. 音频采样频率定义. 999
图 374. I^2S 时钟发生器架构. 999
图 375. 功能框图 1019
图 376. 音频帧. 1022
图 377. FS 的作用是 SOF 信号 + 通道识别信号 (FSDEF = TRIS = 1) 1024
图 378. FS 的作用是 SOF 信号 (FSDEF = 0) 1025
图 379. SAI_xSLOTR 中的 FBOFF = 0 时的 Slot 大小配置 1026
图 380. 第一位偏移 1026
图 381. 音频模块时钟发生器概览 1027
图 382. AC’97 音频帧 1030
图 383. 具有至少 2 个嵌入式 SAI（三个外部 AC’97 解码器）的
器件上的典型 AC’97 配置示例 1031
图 384. SPDIF 格式. 1032
图 385. SAI_xDR 寄存器定序 1033
图 386. SAI 的音频模块中的数据压扩硬件 1036
图 387. 发送无效 Slot 时 SD 输出线上的三态策略 1037
图 388. 采用 I2S 等协议时输出数据线上的三态策略. 1038
图 389. 上溢错误检测 1039
图 390. FIFO 下溢事件 1039
图 391. SPDIFRX 框图 1058
图 392. S/PDIF 子帧格式. 1058
图 393. S/PDIF 块格式 1059
图 394. S/PDIF 报头 1059
图 395. 通道编码示例 1060
图 396. SPDIFRX_DC 解码器. 1061
图 397. 噪声滤波和边沿检测 1061
图 398. 阈值. 1062
图 399. 同步流程图 1065
图 400. 同步过程调度 1066
图 401. SPDIFRX 状态 1067
图 402. SPDIFRX_DR 寄存器格式 1069
图 403. 通道/用户数据格式. 1070
图 404. RXSTEO = 0 时的 S/PDIF 上溢错误. 1072
图 405. RXSTEO = 1 时的 S/PDIF 上溢错误. 1073
图 406. SPDIFRX 接口中断映射图 1074
图 407. “无响应”和“无数据”操作. 1090
图 408. （多个）块读取操作 1090
图 409. （多个）块写入操作 1090

图 410. 连续读取操作 . . . 1091
图 411. 连续写入操作 . . . 1091
图 412. SDMMC 框图 . . . 1091
图 413. SDMMC 适配器 . . . 1093
图 414. 控制单元 . . . 1094
图 415. SDMMC_CK 时钟移相 (BYPASS = 0) . . . 1094
图 416. SDMMC 适配器命令路径 . . . 1095
图 417. 命令路径状态机 (SDMMC) . . . 1096
图 418. SDMMC 命令传输 . . . 1097
图 419. 数据路径 . . . 1099
图 420. 数据路径状态机 (DPSM) . . . 1100
图 421. CAN 网络拓扑结构 . . . 1144
图 422. 双 CAN 框图 . . . 1145
图 423. bxCAN 工作模式 . . . 1147
图 424. 静默模式下的 bxCAN . . . 1148
图 425. 环回模式下的 bxCAN . . . 1148
图 426. 组合模式下的 bxCAN . . . 1149
图 427. 发送邮箱状态 . . . 1150
图 428. 接收 FIFO 状态 . . . 1151
图 429. 滤波器组尺度配置 - 寄存器构成 . . . 1153
图 430. 滤波器编号示例 . . . 1154
图 431. 筛选机制 - 示例 . . . 1155
图 432. CAN 错误状态图 . . . 1156
图 433. 位时序 . . . 1158
图 434. CAN 帧 . . . 1159
图 435. 事件标志与中断产生 . . . 1160
图 436. CAN 邮箱寄存器 . . . 1171
图 437. 全速 OTG 模块框图 . . . 1188
图 438. 高速 OTG 模块框图 . . . 1189
图 439. OTG_FS A-B 设备连接 . . . 1191
图 440. USB_FS 仅做设备的连接 . . . 1193
图 441. USB_FS 仅做主机的连接 . . . 1196
图 442. SOF 连接（SOF 触发输出与 TIM 和 ITR1 的连接） . . . 1200
图 443. 动态更新 OTG_HFIR . . . 1202
图 444. 设备模式下的 FIFO 地址映射和 AHB FIFO 访问映射 . . . 1203
图 445. 主机模式下的 FIFO 地址映射和 AHB FIFO 访问映射 . . . 1204
图 446. 中断层级 . . . 1207
图 447. 发送 FIFO 写任务 . . . 1298
图 448. 接收 FIFO 读任务 . . . 1299
图 449. 正常批量/控制 OUT/SETUP . . . 1300
图 450. 批量/控制 IN 事务 . . . 1303
图 451. 正常中断 OUT . . . 1305
图 452. 正常中断 IN . . . 1309
图 453. 同步 OUT 事务 . . . 1311
图 454. 同步 IN 事务 . . . 1314
图 455. 正常批量/控制 OUT/SETUP 事务 - DMA . . . 1316
图 456. 正常批量/控制 IN 事务 - DMA . . . 1318
图 457. 正常中断 OUT 事务 - DMA 模式 . . . 1319
图 458. 正常中断 IN 事务 - DMA 模式 . . . 1320
图 459. 正常同步 OUT 事务 - DMA 模式 . . . 1321
图 460. 正常同步 IN 事务 - DMA 模式 . . . 1322

图 461. 接收 FIFO 数据包读取 1327
图 462. 处理 SETUP 数据包 1329
图 463. 批量 OUT 事务 1335
图 464. TRDT 最大时序情况 1343
图 465. A 器件 SRP 1344
图 466. B 器件 SRP 1345
图 467. A 器件 HNP 1346
图 468. B 器件 HNP 1347
图 469. ETH 框图 1352
图 470. SMI 接口信号 1353
图 471. MDIO 时序和帧结构 - 写周期 1354
图 472. MDIO 时序和帧结构 - 读周期 1355
图 473. 介质独立接口信号 1356
图 474. MII 时钟源 1357
图 475. 精简介质独立接口信号 1358
图 476. RMII 时钟源 1358
图 477. 时钟方案 1359
图 478. 地址字段格式 1361
图 479. MAC 帧格式 1362
图 480. 带标记的 MAC 帧格式 1362
图 481. 发送位序 1368
图 482. 无冲突发送 1368
图 483. 有冲突发送 1369
图 484. MMI 和 RMII 模式下的帧发送 1369
图 485. 接收位序 1372
图 486. 无错误接收 1373
图 487. 有错误接收 1373
图 488. 出现假载波指示的接收 1373
图 489. MAC 内核中断屏蔽方案 1374
图 490. 唤醒帧过滤寄存器 1378
图 491. 网络时间同步 1380
图 492. 使用精密校准方法更新系统时间 1382
图 493. PTP 触发输出与 TIM2 ITR1 的连接 1384
图 494. PPS 输出 1385
图 495. 描述符环形结构与链接结构 1385
图 496. 默认模式下的 TxDMA 操作 1389
图 497. OSF 模式下的 TxDMA 操作 1390
图 498. 常规发送描述符 1392
图 499. 增强的发送描述符 1396
图 500. DMA 接收操作 1398
图 501. 常规 Rx DMA 描述符结构 1399
图 502. 使能带有 IEEE1588 时间戳的增强的接收描述符字段格式 1404
图 503. 中断方案 1407
图 504. 以太网 MAC 远程唤醒帧过滤寄存器 (ETH_MACRWUFFR) 1416
图 505. 框图 1455
图 506. 消息结构 1456
图 507. 块 1456
图 508. 位时序 1457
图 509. 信号空闲时间 1457
图 510. 仲裁阶段 1458
图 511. 三个标称位周期的 SFT 1458

图 512. 错误位时序 1459
图 513. 错误处理 1460
图 514. TXERR 检测 1462
图 515. STM32 MCU 和带 FPU 的 Cortex®-M7 级调试支持框图 1473
图 516. SWJ 调试端口 1475
图 517. JTAG 调试端口连接 1478
图 518. TPIU 框图 1495

1 文档约定

1.1 寄存器相关缩写词列表

寄存器说明中使用以下缩写词：

读/写 (rw)	软件可以读写这些位。
只读 (r)	软件只能读取这些位。
只写 (w)	软件只能写入该位。读取该位时将返回复位值。
读取/清零 (rc_w1)	软件可以读取该位，也可以通过写入 1 将该位清零。写入“0”对该位的值无影响。
读取/清零 (rc_w0)	软件可以读取该位，也可以通过写入 0 将该位清零。写入“1”对该位的值无影响。
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“0”对该位的值无影响。
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。
保留 (Res.)	保留位，必须保持复位值。

1.2 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：

- CPU 内核集成了两个调试端口：
  - JTAG 调试端口 (**JTAG-DP**) 提供基于联合测试工作组 (JTAG) 协议的 5 引脚标准接口。
  - SWD 调试端口 (**SWD-DP**) 提供基于串行线调试 (SWD) 协议的 2 引脚（时钟和数据）接口。

  有关 JTAG 和 SWD 协议的信息，请参见《Cortex®-M7 技术参考手册》。
- **字**：32 位数据。
- **半字**：16 位数据。
- **字节**：8 位数据。
- **双字**：64 位数据。
- **IAP（在应用中编程）**：IAP 是指可以在用户程序运行期间对微控制器的 Flash 进行重新编程。
- **ICP（在线编程）**：ICP 是指可以在器件安装于用户应用电路板上时使用 JTAG 协议、SWD 协议或自举程序对微控制器的 Flash 进行编程。
- **选项字节**：存储于 Flash 中的产品配置位。
- **AHB**：高级高性能总线。
- **AHBS**：AHB 从总线。
- **AXIM**：AXI 主控总线。
- **ITCM**：指令紧密耦合存储器。
- **DTCM**：数据紧密耦合存储器。
- **CPU**：指带 FPU 的 Cortex®-M7 内核。

1.3 外设可用性

有关各型号产品的外设可用性及数量信息，请参见特殊器件数据手册。

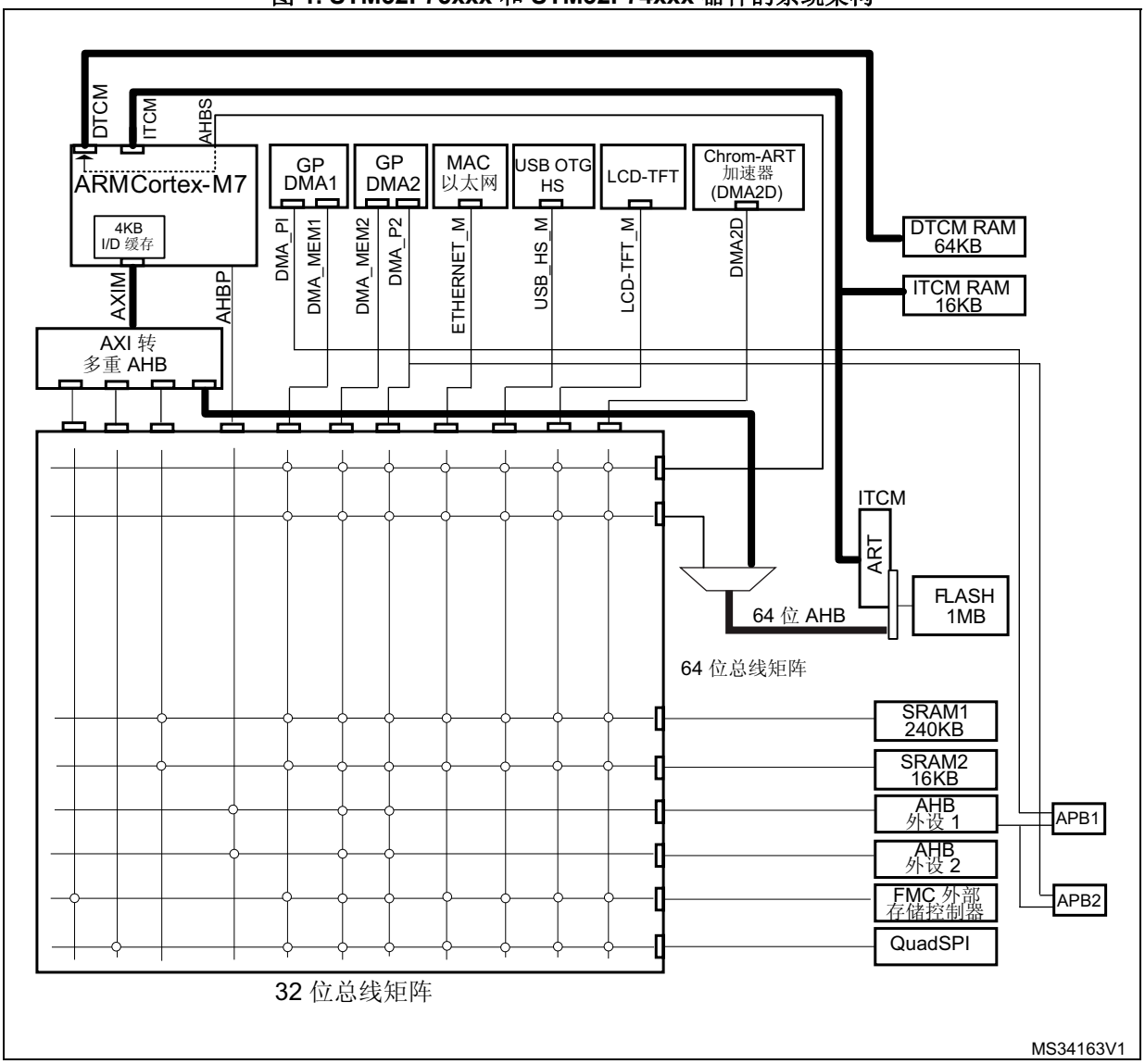
2 系统和存储器概述

2.1 系统架构

主系统架构基于 2 个子系统：

- 一个 AXI 转 multi-AHB 总线桥，用于将 AXI4 协议转换为 AHB-Lite 协议：
  - 一个连接到内嵌 flash 的 AXI 转 64 位 AHB 总线桥
  - 3 个连接到 AHB 总线矩阵的 AXI 转 32 位 AHB 总线桥
- 一个 multi-AHB 总线矩阵

图 1. STM32F75xxx 和 STM32F74xxx 器件的系统架构



multi-AHB 总线矩阵将所有主控总线和被控总线互连，它包括：

- 32 位 multi-AHB 总线矩阵
- 64 位 multi-AHB 总线矩阵：它将来自 CPU 的 64 位 AHB 总线（通过 AXI 转 AHB 总线桥）和来自 GP DMA 与外设 DMA（增至 64 位）的 32 位 AHB 总线连接到内部 flash。

multi-AHB 总线矩阵可连接：

- 十二个总线主控器：
  - 3x32 位 AHB 总线以及 64 位 Cortex®-M7 AXI 主控总线通过 AXI-AHB 总线桥分为 4 个总线主控器
  - 连接到内嵌 flash 的 1x64 位 AHB 总线
  - Cortex® -M7 AHB 外设总线
  - DMA1 存储器总线
  - DMA2 存储器总线
  - DMA2 外设总线
  - 以太网 DMA 总线
  - USB OTG HS DMA 总线
  - LCD 控制器 DMA 总线
  - Chrom-Art 加速器 ™ (DMA2D) 存储器总线
- 八个总线从控制器：
  - AHB 总线上的内嵌 Flash（用于 Flash 读/写访问、代码执行和数据访问）
  - Cortex®-M7 AHBS 从接口（仅用于 DTCM RAM 的 DMA 数据传输）
  - 主 SRAM1 (240 KB)
  - 辅助 SRAM2 (16 KB)
  - AHB1 外设（包括 AHB-APB 总线桥和 APB 外设）
  - AHB2 外设（包括 AHB-APB 总线桥和 APB 外设）
  - FMC
  - Quad SPI

2.1.1 multi-AHB 总线矩阵

multi-AHB 总线矩阵用于主控器之间的访问仲裁管理。仲裁采用循环调度算法。

借助该总线矩阵，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。
DTCM 和 ITCM RAM（TCM - 紧密耦合型存储器）不是总线矩阵的一部分。
数据 TCM RAM 可由 GP-DMA 和 外设 DMA 通过 CPU 的特定 AHB 被控总线访问。
指令 TCM RAM 仅留供 CPU 使用。支持以 CPU 时钟速度，0 个等待周期访问该指令。*图 1* 中给出了相应的架构。

2.1.2 AHB/APB 总线桥 (APB)

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

有关 APB1 和 APB2 最大频率的详细信息，请参见器件数据手册；有关 AHB 和 APB 外设地址映射的信息，请参见*表 1*。

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

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

2.1.3 CPU AXIM 总线

该总线通过 AXI-AHB 总线桥将带 FPU 的 Cortex®-M7 内核的指令总线和数据总线连接到 multi-AHB 总线矩阵。共有 4 种 AXI 总线访问：

- CPU AXI 总线访问 1：该 AXI 总线访问的目标是包含代码或数据的外部存储器（通过 FMC 外设接口）。对于映射到地址 0x8000 0000 到 0x8FFF FFFF 的 NAND 存储区域，该空间的 MPU 存储器特性必须通过软件重新配置到器件中。
- CPU AXI 总线访问 2：该 AXI 总线访问的目标是包含代码或数据的外部存储器（通过 Quad SPI 外设接口）。
- CPU AXI 总线访问 3：该 AXI 总线访问的目标是包含代码或数据的内部 SRAM（SRAM1 和 SRAM2）。
- CPU AXI 总线访问 4：该 AXI 总线访问的目标是映射到 AXI 接口的、包含代码或数据的内部 Flash。

2.1.4 ITCM 总线

Cortex®-M7 使用该总线对映射到 ITCM 接口上的内嵌 flash 进行取指和数据访问，但对于 ITCM RAM，该总线只能进行取指操作。

2.1.5 DTCM 总线

Cortex®-M7 使用该总线对 DTCM RAM 进行数据访问，也可进行取指。

2.1.6 CPU AHBS 总线

该总线将 Cortex®-M7 的 AHB 被控总线连接到总线矩阵。该总线仅用于通用 DMA 和外设 DMA 到 DTCM RAM 上的数据传输。

AHBS 上无法访问 ITCM 总线，因此 RAM 不能通过 ITCM 总线进行 DMA 数据传输。Flash 通过 ITCM 接口进行 DMA 传输时，将强制通过 AHB 总线进行所有传输。

2.1.7 AHB 外设总线

该总线将 Cortex®-M7 的 AHB 外设总线连接到总线矩阵。内核使用该总线执行所有针对外设的数据访问。

该总线的访问目标是 AHB1 总线上的外设（包括 APB 总线上的外设和 AHB2 总线上的外设）。

2.1.8 DMA 存储器总线

此总线用于将 DMA 存储器总线主接口连接到总线矩阵。DMA 通过此总线来执行存储器数据的传入和传出。该总线的访问目标是数据存储器：内部 SRAM1、SRAM2 和 DTCM（通过 Cortex®-M7 的 AHBS 总线）以及内部 Flash 和外部存储器（通过外设接口 FMC 或 Quad SPI）。

2.1.9 DMA 外设总线

此总线用于将 DMA 外设主总线接口连接到总线矩阵。DMA 通过此总线访问 AHB 外设或执行存储器间的数据传输。该总线的访问目标是 AHB 和 APB 总线上的外设以及数据存储器：内部 SRAM1、SRAM2 和 DTCM（通过 Cortex®-M7 的 AHBS 总线）以及内部 Flash 和外部存储器（通过外设接口 FMC 或 Quad SPI）。

2.1.10 以太网 DMA 总线

此总线用于将以太网 DMA 主接口连接到总线矩阵。以太网 DMA 通过此总线向存储器存取数据。该总线的访问目标是数据存储器：内部 SRAM1、SRAM2 和 DTCM（通过 Cortex®-M7 的 AHBS 总线）、内部 Flash 和外部存储器（通过外设接口 FMC 或 Quad SPI）。

2.1.11 USB OTG HS DMA 总线

此总线用于将 USB OTG HS DMA 主接口连接到总线矩阵。USB OTG DMA 通过此总线向存储器加载/存储数据。该总线的访问目标是数据存储器：内部 SRAM1、SRAM2 和 DTCM（通过 Cortex®-M7 的 AHBS 总线）、内部 Flash 和外部存储器（通过外设接口 FMC 或 Quad SPI）。

2.1.12 LCD-TFT 控制器 DMA 总线

此总线用于将 LCD 控制器 DMA 主接口连接到总线矩阵。LCD-TFT DMA 通过此总线向存储器加载/存储数据。该总线的访问目标是数据存储器：内部 SRAM1、SRAM2 和 DTCM（通过 Cortex®-M7 的 AHBS 总线）和外部存储器（通过外设接口 FMC 或 Quad SPI）和内部 Flash。

2.1.13 DMA2D 总线

此总线用于将 DMA2D 主接口连接到总线矩阵。DMA2D 图形加速器通过此总线向存储器加载/存储数据。该总线的访问目标是数据存储器：内部 SRAM1、SRAM2 和 DTCM（通过 Cortex®-M7 的 AHBS 总线）和外部存储器（通过外设接口 FMC 或 Quad SPI）和内部 Flash。

2.2 存储器组织结构

2.2.1 前言

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。字中编号最低的字节被视为该字的最低有效字节，而编号最高的字节被视为最高有效字节。

可寻址的存储空间分为 8 个主要块，每个块为 512 MB。

未分配给片上存储器和外设的所有存储区域均视为“保留区”。有关可用存储器和寄存器区域的详细映射，请参见*存储器映射和寄存器边界地址*一章和外设章节。

2.2.2 存储器映射和寄存器边界地址

有关综合存储器映射图，请参见相应器件的数据手册。

下表给出了器件中可用外设的边界地址。

表 1. STM32F75xxx 和 STM32F74xxx 寄存器边界地址

边界地址	外设	总线	寄存器映射
0xA000 1000 - 0xA0001FFF	QuadSPI 控制寄存器	AHB3	*第 387 页的第 14.5.14 节：QUADSPI 寄存器映射*
0xA000 0000 - 0xA000 0FFF	FMC 控制寄存器		*第 358 页的第 13.8 节：FMC 寄存器映射*
0x5006 0800 - 0x5006 0BFF	RNG	AHB2	*第 508 页的第 19.4.4 节：RNG 寄存器映射*
0x5006 0400 - 0x5006 07FF	HASH		*第 564 页的第 21.4.8 节：散列寄存器映射*
0x5006 0000 - 0x5006 03FF	CRYP		*第 545 页的第 20.6.12 节：CRYP 寄存器映射*
0x5005 0000 - 0x5005 03FF	DCMI		*第 470 页的第 17.8.12 节：DCMI 寄存器映射*
0x5000 0000 - 0x5003 FFFF	USB OTG FS		*第 1284 页的第 37.15.58 节：OTG_FS/OTG_HS 寄存器映射*

表 1. STM32F75xxx 和 STM32F74xxx 寄存器边界地址（续）

边界地址	外设	总线	寄存器映射
0x4004 0000 - 0x4007 FFFF	USB OTG HS	AHB1	*第 1284 页的第 37.15.58 节：OTG_FS/OTG_HS 寄存器映射*
0x4002 B000 - 0x4002 BBFF	Chrom-ART (DMA2D)		*第 266 页的第 9.5.21 节：DMA2D 寄存器映射*
0x4002 8000 - 0x4002 93FF	以太网 MAC		*第 1450 页的第 38.8.5 节：以太网寄存器映射*
0x4002 6400 - 0x4002 67FF	DMA2		*第 236 页的第 8.5.11 节：DMA 寄存器映射*
0x4002 6000 - 0x4002 63FF	DMA1		
0x4002 4000 - 0x4002 4FFF	BKPSRAM		*第 178 页的第 5.3.27 节：RCC 寄存器映射*
0x4002 3C00 - 0x4002 3FFF	Flash 接口寄存器		*第 3.7.8 节：Flash 接口寄存器映射*
0x4002 3800 - 0x4002 3BFF	RCC		*第 178 页的第 5.3.27 节：RCC 寄存器映射*
0x4002 3000 - 0x4002 33FF	CRC		*第 285 页的第 12.4.6 节：CRC 寄存器映射*
0x4002 2800 - 0x4002 2BFF	GPIOK		*第 197 页的第 6.4.11 节：GPIO 寄存器映射*
0x4002 2400 - 0x4002 27FF	GPIOJ		
0x4002 2000 - 0x4002 23FF	GPIOI		*第 197 页的第 6.4.11 节：GPIO 寄存器映射*
0x4002 1C00 - 0x4002 1FFF	GPIOH		
0x4002 1800 - 0x4002 1BFF	GPIOG		
0x4002 1400 - 0x4002 17FF	GPIOF		
0x4002 1000 - 0x4002 13FF	GPIOE		
0x4002 0C00 - 0x4002 0FFF	GPIOD		
0x4002 0800 - 0x4002 0BFF	GPIOC		
0x4002 0400 - 0x4002 07FF	GPIOB		
0x4002 0000 - 0x4002 03FF	GPIOA		

表 1. STM32F75xxx 和 STM32F74xxx 寄存器边界地址（续）

边界地址	外设	总线	寄存器映射
0x4001 6800 - 0x4001 6BFF	LCD-TFT	APB2	*第 501 页的第 18.7.26 节：LTDC 寄存器映射*
0x4001 5C00 - 0x4001 5FFF	SAI2		*第 1055 页的第 33.5.10 节：SAI 寄存器映射*
0x4001 5800 - 0x4001 5BFF	SAI1		*第 1055 页的第 33.5.10 节：SAI 寄存器映射*
0x4001 5400 - 0x4001 57FF	SPI6		*第 1017 页的第 32.9.10 节：SPI/I2S 寄存器映射*
0x4001 5000 - 0x4001 53FF	SPI5		
0x4001 4800 - 0x4001 4BFF	TIM11		*第 759 页的第 24.5.12 节：TIM10/11/13/14 寄存器映射*
0x4001 4400 - 0x4001 47FF	TIM10		
0x4001 4000 - 0x4001 43FF	TIM9		*第 749 页的第 24.4.13 节：TIM9/12 寄存器映射*
0x4001 3C00 - 0x4001 3FFF	EXTI		*第 279 页的第 11.9.7 节：EXTI 寄存器映射*
0x4001 3800 - 0x4001 3BFF	SYSCFG		*第 205 页的第 7.2.8 节：SYSCFG 寄存器映射*
0x4001 3400 - 0x4001 37FF	SPI4		*第 1017 页的第 32.9.10 节：SPI/I2S 寄存器映射*
0x4001 3000 - 0x4001 33FF	SPI1		*第 1017 页的第 32.9.10 节：SPI/I2S 寄存器映射*
0x4001 2C00 - 0x4001 2FFF	SDMMC1		*第 1141 页的第 35.8.16 节：SDMMC 寄存器映射*
0x4001 2000 - 0x4001 23FF	ADC1 - ADC2 - ADC3		*第 429 页的第 15.13.18 节：ADC 寄存器映射*
0x4001 1400 - 0x4001 17FF	USART6		*第 963 页的第 31.8.12 节：USART 寄存器映射*
0x4001 1000 - 0x4001 13FF	USART1		
0x4001 0400 - 0x4001 07FF	TIM8		*第 645 页的第 22.4.24 节：TIM1 寄存器映射*
0x4001 0000 - 0x4001 03FF	TIM1		

表 1. STM32F75xxx 和 STM32F74xxx 寄存器边界地址（续）

边界地址	外设	总线	寄存器映射
0x4000 7C00 - 0x4000 7FFF	UART8	APB1	第 963 页的第 31.8.12 节：USART 寄存器映射
0x4000 7800 - 0x4000 7BFF	UART7		
0x4000 7400 - 0x4000 77FF	DAC		第 449 页的第 16.5.15 节：DAC 寄存器映射
0x4000 7000 - 0x4000 73FF	PWR		第 118 页的第 4.4.4 节：PWR 电源控制寄存器 2 (PWR_CSR2)
0x4000 6C00 - 0x4000 6FFF	HDMI-CEC		第 1472 页的第 39.7.7 节：HDMI-CEC 寄存器映射
0x4000 6800 - 0x4000 6BFF	CAN2		第 1181 页的第 36.9.5 节：bxCAN 寄存器映射
0x4000 6400 - 0x4000 67FF	CAN1		
0x4000 6000 - 0x4000 63FF	I2C4		第 905 页的第 30.7.12 节：I2C 寄存器映射
0x4000 5C00 - 0x4000 5FFF	I2C3		第 905 页的第 30.7.12 节：I2C 寄存器映射
0x4000 5800 - 0x4000 5BFF	I2C2		
0x4000 5400 - 0x4000 57FF	I2C1		
0x4000 5000 - 0x4000 53FF	UART5		第 963 页的第 31.8.12 节：USART 寄存器映射
0x4000 4C00 - 0x4000 4FFF	UART4		
0x4000 4800 - 0x4000 4BFF	USART3		
0x4000 4400 - 0x4000 47FF	USART2		
0x4000 4000 - 0x4000 43FF	SPDIFRX		第 1088 页的第 34.5.10 节：SPDIFRX 接口寄存器映射
0x4000 3C00 - 0x4000 3FFF	SPI3 / I2S3		第 1017 页的第 32.9.10 节：SPI/I2S 寄存器映射
0x4000 3800 - 0x4000 3BFF	SPI2 / I2S2		
0x4000 3000 - 0x4000 33FF	IWDG		第 800 页的第 27.4.6 节：IWDG 寄存器映射
0x4000 2C00 - 0x4000 2FFF	WWDG		第 806 页的第 28.4.4 节：WWDG 寄存器映射
0x4000 2800 - 0x4000 2BFF	RTC 和 BKP 寄存器		第 844 页的第 29.6.21 节：RTC 寄存器映射
0x4000 2400 - 0x4000 27FF	LPTIM1		第 793 页的第 26.6.11 节：LPTIM 寄存器映射
0x4000 2000 - 0x4000 23FF	TIM14		第 759 页的第 24.5.12 节：TIM10/11/13/14 寄存器映射
0x4000 1C00 - 0x4000 1FFF	TIM13		
0x4000 1800 - 0x4000 1BFF	TIM12		第 749 页的第 24.4.13 节：TIM9/12 寄存器映射
0x4000 1400 - 0x4000 17FF	TIM7		第 773 页的第 25.4.9 节：TIM6/TIM7 寄存器映射
0x4000 1000 - 0x4000 13FF	TIM6		
0x4000 0C00 - 0x4000 0FFF	TIM5		第 714 页的第 23.4.22 节：TIMx 寄存器映射
0x4000 0800 - 0x4000 0BFF	TIM4		
0x4000 0400 - 0x4000 07FF	TIM3		
0x4000 0000 - 0x4000 03FF	TIM2		

2.3 内部 SRAM

STM32F75xxx 和 STM32F74xxx 具有以下特性：

- 多达 320 KB 的系统 SRAM，包含 64 KB 的数据 TCM RAM
- 16 KB 的指令 RAM (ITCM-RAM)
- 4 KB 的备份 SRAM（请参见第 5.1.2 节：电池备份域）

内部 SRAM 最多可分为四个块：

- 系统 SRAM：
  - 映射到地址 0x2001 0000 的 SRAM1，可供来自 AHB 总线矩阵的所有 AHB 主控主线访问。
  - 映射到地址 0x2004 C000 的 SRAM2，可供所有来自 AHB 总线矩阵的 AHB 主控主线访问。
  - 映射到地址 0x2000 0000 的 TCM 接口上的 DTCM-RAM（紧密耦合存储器接口），可供所有来自 AHB 总线矩阵的 AHB 主控主线访问，但只能通过 CPU 的特定 AHB 从总线访问。
- 指令 SRAM：
  - 映射到地址 0x0000 0000 的指令 RAM (ITCM-RAM)，仅供 CPU 访问。

SRAM1 和 SRAM2 可按字节、半字（16 位）或全字（32 位）访问。DTCM 和 ITCM RAM 可按字节、半字（16 位）、全字（32 位）或双字（64 位）访问。这些存储器可在最高系统时钟频率下以 0 等待周期寻址。

AHB 主控总线支持并发 SRAM 访问（通过以太网或 USB OTG HS）：例如，当 CPU 对 SRAM1 进行读/写操作时，以太网 MAC 可以同时对 SRAM2 进行读/写操作。

2.4 Flash 概述

Flash 接口可管理 CPU AXI 和 TCM 对 Flash 进行的访问。该接口可针对 Flash 执行擦除和编程操作，并实施读写保护机制。Flash 接口通过 TCM 接口上的 ART 加速器或 AXIM 接口上的 L1 缓存加速代码执行。

Flash 结构如下：

- 主存储器块分为多个扇区。
- 信息块：
  - 系统存储器，器件在系统存储器自举模式下从该存储器启动
  - 1024 OTP（一次性可编程）字节，用于存储用户数据。
  - 选项字节，用于配置读写保护、BOR 级别、软件/硬件看门狗以及器件处于待机或停止模式下的复位。

更多详细信息，请参见*第 3 节：嵌入式 Flash (FLASH)*。

2.5 自举配置

在 STM32F75xxx 和 STM32F74xxx 中，可通过 BOOT 引脚选择两种不同的自举空间并且 BOOT_ADD0 和 BOOT_ADD1 选项字节中编程的自举基址如*表 2* 所示。

表 2. 自举模式

自举模式选择		自举空间
BOOT	自举地址选项字节	
0	BOOT_ADD0[15:0]	由用户选项字节 BOOT_ADD0[15:0] ST 出厂默认的自举地址：位于 0x0020 0000 的 ITCM 上的 Flash
1	BOOT_ADD1[15:0]	由用户选项字节 BOOT_ADD1[15:0] ST 出厂默认的自举地址：位于 0x0010 0000 的系统自举程序

复位后，在 SYSCLK 的第四个上升沿锁存 BOOT 引脚的值。复位后由用户设置 BOOT 引脚。

器件退出待机模式时，还会对 BOOT 引脚重新采样。因此，当器件处于待机模式时，这些引脚必须保持所需的自举模式配置。

启动延迟后（比如从 stop 模式被唤醒），系统会在释放处理器复位前选择自举空间。

BOOT_ADD0 和 BOOT_ADD1 地址选项字节允许将启动地址配置为从 0x0000 0000 到 0x3FFF FFFF 的任意存储器地址，包括：

- 映射到 ITCM 或 AXIM 接口上的所有 Flash 地址空间
- 所有 RAM 地址空间：映射到 AXIM 接口上的 ITCM、DTCM RAM 和 SRAM
- 系统存储器自举程序

可在复位后修改 BOOT_ADD0/BOOT_ADD1 选项字节以在下次复位后从任何其它自举地址自举。

如果编程的自举存储器地址位于存储器映射区域或保留区域之外，则按如下方式编程默认自举获取地址：

  – 自举地址 0：位于 0x0020 0000 的 ITCM-FLASH
  – 自举地址 1：位于 0x0000 0000 的 ITCM-RAM

当 Flash 的保护级别被配置为级别 2 之后，只能从 Flash（位于 ITCM 或 AXIM 接口上）或系统自举程序自举。如果 BOOT_ADD0 和/或 BOOT_ADD1 选项字节中自举地址被配置为位于存储器范围或 RAM 地址（位于 ITCM 或 AXIM 上）之外，则系统只能从位于地址 0x00200000 的 ITCM 接口上的 Flash 开始执行。

内部自举程序

内部自举程序代码位于系统存储器中， 在芯片生产期间由 ST 编程。它用于通过以下串行接口重新编程 Flash：

- 引脚 PA9/PA10 上的 USART1 以及引脚 PB10/PB11 和 PC10/PC11 上的 USART3。
- 引脚 PB5/PB13 上的 CAN2。
- USB OTG FS（引脚 PA11/PA12 上）从设备模式（DFU：器件固件升级）。
- 引脚 PB6/9 上的 I2C1、引脚 PF0/PF1 上的 I2C2 和引脚 PA8/PC9 上的 I2C3。

有关详细信息，请参见应用笔记 (AN2606)。

默认情况下，当选择从系统自举程序自举时，代码从 TCM 接口执行。可通过选项字节重新配置启动的接口。

3 嵌入式 Flash (FLASH)

3.1 前言

Cortex®-M7 AXI 总线和 TCM 总线对 Flash 进行的访问，由 FLASH 内存接口进行管理。该接口可针对 Flash 执行擦除和编程操作，并实施读写保护机制。

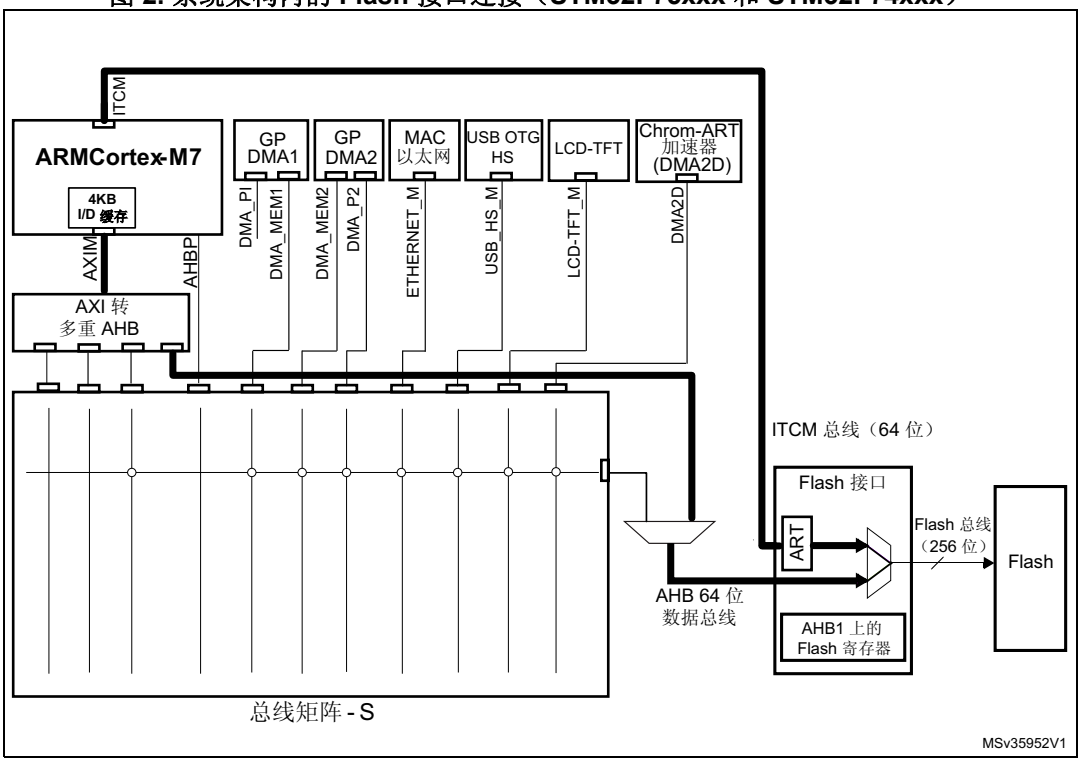
Flash 接口通过 ITCM 接口 (ART Accelerator™) 上的指令预取和缓存机制加速代码执行。

3.2 Flash 主要特性

- Flash 读操作
- Flash 编程/擦除操作
- 读/写保护
- ITCM 接口 (ART Accelerator™) 上的 64 个缓存（256 位宽）
- TCM 指令代码的预取操作

图 2 所示为系统架构内的 Flash 接口连接。

图 2. 系统架构内的 Flash 接口连接（STM32F75xxx 和 STM32F74xxx）



3.3 Flash 功能说明

3.3.1 Flash 构成

Flash 具有以下主要特性：

- 容量高达 1 MB
- 256 位宽数据读取
- 字节、半字、字和双字数据写入
- 扇区擦除与全部擦除

Flash 结构如下：

- 主存储器块，分为 4 个 32 KB 扇区、1 个 128 KB 扇区和 3 个 256 KB 扇区
- 信息块包括：
  - 系统存储器，器件在系统存储器自举模式下从该存储器启动。
  - 1024 字节 OTP（一次性可编程），用于存储用户数据。
  - OTP 区域还有 16 个额外字节，用于锁定对应的 OTP 数据块。
  - 选项字节，用于配置读写保护、BOR 级别、看门狗、自举存储器基址、软件/硬件以及器件进入待机或停止模式是否复位。

嵌入式 Flash 具有三个主接口：

- 64 位 ITCM 接口：
  - 该接口连接 Cortex-M7 的 ITCM 总线，用于指令执行以及数据读访问。
  - ITCM 接口不支持写访问。
  - 支持 64 个标准化缓存行，256 位宽（ART 加速器）。
- 64 位 AHB 接口：
  - 该接口通过 AHB 总线矩阵连接 Cortex-M7 的 AXI 总线，用于代码执行以及读写访问。
  - 无论寻址的 Flash 接口是 TCM 还是 AHB，Flash 上的 DMA 和外设 DMA 数据传输均通过 AHB 接口完成。
- 32 位 AHB 寄存器接口：
  - 用于访问控制寄存器和状态寄存器。

表 7 所示为主存储器块和信息块的构成。

表 3. Flash 构成

块	名称	AXIM 接口上的块基址	ICTM 接口上的块基址	扇区大小
主存储器块	扇区 0	0x0800 0000 - 0x0800 7FFF	0x0020 0000 - 0x0020 7FFF	32 KB
	扇区 1	0x0800 8000 - 0x0800 FFFF	0x0020 8000 - 0x0020 FFFF	32 KB
	扇区 2	0x0801 0000 - 0x0801 7FFF	0x0021 0000 - 0x0021 7FFF	32 KB
	扇区 3	0x0801 8000 - 0x0801 FFFF	0x0021 8000 - 0x0021 FFFF	32 KB
	扇区 4	0x0802 0000 - 0x0803 FFFF	0x0022 0000 - 0x0023 FFFF	128 KB
	扇区 5	0x0804 0000 - 0x0807 FFFF	0x0024 0000 - 0x0027 FFFF	256 KB
	扇区 6	0x0808 0000 - 0x080B FFFF	0x0028 0000 - 0x002B FFFF	256 KB
	扇区 7	0x080C 0000 - 0x080F FFFF	0x002C 0000 - 0x02F FFFF	256 KB

表 3. Flash 构成（续）

块	名称	AXIM 接口上的块基址	ICTM 接口上的块基址	扇区大小
信息块	系统存储器	0x1FF0 0000 - 0x1FF0 EDBF	0x0010 0000 - 0x0010 EDBF	60 KB
	OTP	0x1FF0 F000 - 0x1FF0 F41F	0x0010 F000 - 0x0010 F41F	1024 字节
	选项字节	0x1FFF 0000 - 0x1FFF 001F		32 字节

3.3.2 读访问延迟

为了准确读取 Flash 数据，必须根据 CPU 时钟 (HCLK) 频率和器件电源电压在 Flash 存取控制寄存器 (FLASH_ACR) 中正确地编程等待周期数 (LATENCY)。

表 13 和*表 4* 给出了等待周期和 CPU 时钟频率之间的对应关系。

注：

- 当 VOS[1:0] = “0x01”时，f_{HCLK} 最大值为 144 MHz。

- 当 VOS[1:0] = “0x10”时，f_{HCLK} 最大值为 168 MHz。激活过驱动模式可将其扩展到 180 MHz。

- 当 VOS[1:0] = “0x11”时，f_{HCLK} 最大值为 180 MHz。激活过驱动模式可将其扩展到 216 MHz。

- 当 V_{DD} 介于 1.8 V 到 2.1 V 之间时，不可使用过驱动模式。

有关如何激活过驱动模式的详细信息，请参见第 4.1.4 节：调压器。

表 4. CPU 时钟 (HCLK) 频率对应的等待周期数

等待周期 (WS) (LATENCY)	HCLK (MHz)			
	电压范围 2.7 V - 3.6 V	电压范围 2.4 V - 2.7 V	电压范围 2.1 V - 2.4 V	电压范围 1.8 V - 2.1 V
0 WS（1 个 CPU 周期）	0 < HCLK ≤ 30	0 < HCLK ≤ 24	0 < HCLK ≤ 22	0 < HCLK ≤ 20
1 WS（2 个 CPU 周期）	30 < HCLK ≤ 60	24 < HCLK ≤ 48	22 < HCLK ≤ 44	20 < HCLK ≤ 40
2 WS（3 个 CPU 周期）	60 < HCLK ≤ 90	48 < HCLK ≤ 72	44 < HCLK ≤ 66	40 < HCLK ≤ 60
3 WS（4 个 CPU 周期）	90 < HCLK ≤ 120	72 < HCLK ≤ 96	66 < HCLK ≤ 88	60 < HCLK ≤ 80
4 WS（5 个 CPU 周期）	120 < HCLK ≤ 150	96 < HCLK ≤ 120	88 < HCLK ≤ 110	80 < HCLK ≤ 100
5 WS（6 个 CPU 周期）	150 < HCLK ≤ 180	120 < HCLK ≤ 144	110 < HCLK ≤ 132	100 < HCLK ≤ 120
6 WS（7 个 CPU 周期）	180 < HCLK ≤ 210	144 < HCLK ≤ 168	132 < HCLK ≤ 154	120 < HCLK ≤ 140
7 WS（8 个 CPU 周期）	210 < HCLK ≤ 216	168 < HCLK ≤ 192	154 < HCLK ≤ 176	140 < HCLK ≤ 160
8 WS（9 个 CPU 周期）	-	192 < HCLK ≤ 216	176 < HCLK ≤ 198	160 < HCLK ≤ 180
9 WS（10 个 CPU 周期）	-	-	198 < HCLK ≤ 216	-

复位后，CPU 时钟频率为 16 MHz，FLASH_ACR 寄存器中的等待周期 (WS) 为 0。

强烈建议通过以下软件序列来根据 CPU 频率调整访问 Flash 时的等待周期数。

需要提高 CPU 频率时的操作步骤

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

需要降低 CPU 频率时的操作步骤

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

注： *CPU 时钟配置或等待周期 (WS) 配置的更改不会立即生效。为了确保当前的 CPU 时钟频率即为所配置的频率，用户可检查 AHB 预分频系数和时钟源状态值。为了确保所编程的 WS 数生效，可读取 FLASH_ACR 寄存器的内容来确认。*

指令预取

每个 flash 读操作可读取 256 位，可以是 8 行 32 位指令，也可以是 16 行 16 位指令，具体取决于烧写在 Flash 中的程序。因此对于顺序执行的代码，至少需要 8 个 CPU 周期来执行前一指令行的读取操作。在 CPU 对当前指令行进行操作请求时，可使用 ITCM 总线的预取操作读取 Flash 中的下一个连续存放的指令行。可通过将 FLASH_ACR 寄存器中的 PRFTEN 位置 1 来使能预取功能。当访问 Flash 至少需要一个等待周期时，此功能非常有用。处理非顺序执行的代码（有分支）时，指令可能并不存在于当前使用的指令行以及预取的指令行中。在这种情况下，CPU 延迟周期数至少等于等待周期数。

自适应实时存储器加速器 (ART Accelerator™)

特有的自适应实时 (ART) 存储器加速器对 STM32（符合工业标准，带有 FPU 的 ARM® Cortex®-M7 处理器）进行了优化。该加速器很好地体现了带有 FPU 的 ARM® Cortex®-M7 的固有性能优势，克服了通常条件下，高速的处理器在运行中需要经常等待 FLASH 读取的情况。

为了发挥处理器的全部性能，该加速器使用标准化的指令缓存和分支缓存，从而提高 256 位 Flash 的程序执行速度。根据 CoreMark 基准测试，凭借 ART 加速器所获得的性能相当于 Flash 在 CPU 频率高达 216 MHz 时以 0 个等待周期执行程序。

ART 加速器仅用于 ITCM 接口上的 Flash 访问。

为了减少因指令跳转而产生的时间损耗，可将 64 行 256 位的指令保存到 ART 加速器中。可通过将 FLASH_CR 寄存器的 ARTEN 位置 1 使能此特性。ART Accelerator 为标准化加速器，包含指令及数据缓冲池。每当出现数据缺失（即请求的数据未存在于当前使用的数据行或指令缓存存储器中）时，系统会将新读取的行复制到 ART 的指令缓存存储器中。如果 CPU 请求的数据已存在于指令缓存存储器中，则无需任何延迟即可立即获取。缓存存储器存满后，可采用 LRU（最近最少使用）策略确定缓存存储器中待替换的行。此特性非常适用于包含循环的代码。

注： *用户配置扇区中的数据无法缓存。*

3.3.3 Flash 编程/擦除操作

执行任何 Flash 编程操作（擦除或编程）时，CPU 时钟频率 (HCLK) 不能低于 1 MHz。如果在 Flash 操作期间发生器件复位，无法保证 Flash 中的内容。

若在对 Flash 执行写入或擦除操作期间尝试读取 Flash，会导致总线阻塞。只有在完成编程操作后，才能正确处理读操作。这意味着，写/擦除操作进行期间不能从 Flash 中执行代码或数据获取操作。

3.3.4 Flash 控制寄存器解锁

复位后，Flash 控制寄存器 (FLASH_CR) 不允许执行写操作，以防因电气干扰等原因出现对 Flash 的意外操作。此寄存器的解锁顺序如下：

1. 在 Flash 密钥寄存器 (FLASH_KEYR) 中写入 KEY1 = 0x45670123
2. 在 Flash 密钥寄存器 (FLASH_KEYR) 中写入 KEY2 = 0xCDEF89AB

如果顺序出现错误，将返回总线错误并锁定 FLASH_CR 寄存器，直到下一次复位。

也可通过软件将 FLASH_CR 寄存器中的 LOCK 位置为 1 来锁定 FLASH_CR 寄存器。

注： *当 FLASH_SR 寄存器中的 BSY 位为 1 时，将不能在写模式下访问 FLASH_CR 寄存器。BSY 位为 1 时，对该寄存器尝试进行写操作会导致 AHB 总线阻塞，直到 BSY 位清零。*

3.3.5 编程/擦除并行位数

通过 FLASH_CR 寄存器中的 PSIZE 域配置并行位数。并行位数表示每次对 Flash 进行写操作时将编程的字节数。PSIZE 受限于电源电压以及是否使用外部 V_{PP} 电源。因此，在进行任何编程/擦除操作前，必须在 FLASH_CR 寄存器中对其进行正确配置。

Flash 擦除操作仅针对扇区、块或整个 Flash（批量擦除）执行。擦除时间取决于 PSIZE 编程值。有关擦除时间的详细信息，请参见器件数据手册的电气特性部分。

表 5 提供了正确的 PSIZE 值。

表 5. 编程/擦除并行位数

	电压范围 2.7 - 3.6 V（使用外部 V_{PP}）	电压范围 2.7 - 3.6 V	电压范围 2.4 - 2.7 V	电压范围 2.1 - 2.4 V	电压范围 1.8 V - 2.1 V
并行位数	x64	x32	x16		x8
PSIZE(1:0)	11	10	01		00

注： *如果在编程并行位数/电压范围设置不一致的情况下启动任何编程或擦除操作，可能会导致出现意外结果。即使后续的读操作指示逻辑值已有效写入存储器中，也无法确定写入操作确实成功。*

如果使用 V_{PP}，必须在 V_{PP} 引脚施加一个外部高压电源（电压介于 8 V 到 9 V 之间）。该外部电源必须在直流电耗超过 10 mA 时也能维持该电压范围。建议仅在工厂生产线上进行初始编程时使用 VPP。V_{PP} 电源的供电时间不得超过一小时，否则 Flash 可能会损坏。

3.3.6 Flash 擦除顺序

Flash 擦除操作可针对扇区或整个 Flash（批量擦除）执行。执行批量擦除时，不会影响 OTP 扇区或配置扇区。

扇区擦除

扇区擦除的具体步骤如下：

1. 检查 FLASH_SR 寄存器中的 BSY 位，以确认当前未执行任何 Flash 操作。
2. 在 FLASH_CR 寄存器中将 SER 位置 1 并选择需要擦除的扇区 (SNB)，所选扇区应为主存储器块中的 8 个扇区之一。
3. 将 FLASH_CR 寄存器中的 STRT 位置 1。
4. 等待 BSY 位清零。

批量擦除

要执行批量擦除，建议采用以下步骤：

1. 检查 FLASH_SR 寄存器中的 BSY 位，以确认当前未执行任何 Flash 操作。
2. 将 FLASH_CR 寄存器中的 MER 位置 1。
3. 将 FLASH_CR 寄存器中的 STRT 位置 1。
4. 等待 BSY 位清零。

注： *如果 FLASH_CR 寄存器中的 MERx 位和 SER 位均置为 1，则执行批量擦除。*

如果 MERx 位和 SER 位均复位并且 STRT 位置 1，可能会发生无法预知的行为而不会生成任何错误标志。应禁止此情况发生。

3.3.7 Flash 编程顺序

标准编程

Flash 编程顺序如下：

1. 检查 FLASH_SR 中的 BSY 位，以确认当前未执行任何主要 Flash 操作。
2. 将 FLASH_CR 寄存器中的 PG 位置 1。
3. 针对所需存储器地址（主存储器块或 OTP 区域内）执行数据写入操作：
   - 并行位数为 x8 时按字节写入
   - 并行位数为 x16 时按半字写入
   - 并行位数为 x32 时按字写入
   - 并行位数为 x64 时按双字写入
4. 等待 BSY 位清零。

注： *把 Flash 的单元从“1”写为“0”时，无需执行擦除操作即可进行连续写操作。把 Flash 的单元从“0”写为“1”时，则需要执行 Flash 擦除操作。*

如果同时发出擦除和编程操作请求，首先执行擦除操作。

编程错误

若发生错误，Flash 操作（编程或擦除）会通过以下错误标志位之一中止：

- **PGAERR**：编程对齐错误 (Alignment Programming error)

  不允许针对 Flash 执行跨越 128 位行界限的数据编程操作。如果出现这种情况，写操作将不会执行，并且 FLASH_SR 寄存器中的编程对齐错误标志位 (PGAERR) 将置 1。

- **PGEPRR**：编程并行位数错误 (Programming parallelism error)

  写访问宽度（字节、半字、字或双字）必须与所选并行位数类型（x8、x16、x32 或 x64）相符。否则，写操作将不会执行，并且 FLASH_SR 寄存器中的编程并行位数错误标志位 (PGPERR) 将置 1。

- **ERSERR**：擦除顺序错误 (Erase sequence error)
  如果代码在控制寄存器未正确配置的情况下对 Flash 执行擦除操作，则 ERSERR 错误标志位将置 1。
- **WRPERR**：写保护错误 (Write Protection Error)
  如果发生以下事件，WRPERR 位会置 1：
  – 尝试在写保护区域 (WRP) 执行编程或擦除操作
  – 尝试对系统存储区（bootloader 自举区）执行编程或擦除操作
  – 针对已锁定的 OTP 区域执行写操作
  – 尝试在读保护 (RDP) 设置为级别 2 时修改选项字节
  – Flash 处于读保护状态，但检测到写操作企图

如果 Flash 中的擦除操作还涉及 ART 加速器中的数据，则必须确保在代码执行期间访问这些数据之前将它们重新写入缓存。如果无法可靠执行这一操作，建议将 FLASH_CR 寄存器中的 ARTRST 或 ARTEN 位置 1，以刷新缓存并/或停用 ART 加速器。

注： *ART 缓存只有在 ART 加速器已禁止 (ARTEN = 0) 时才能清空。*

3.3.8 Flash 中断

如果将 FLASH_CR 寄存器中的操作结束中断使能位 (EOPIE) 置 1，则在擦除或编程操作结束时，即 FLASH_SR 寄存器中的繁忙位 (BSY) 清零（操作成功完成或未成功完成）时，将产生中断。此时，FLASH_SR 寄存器中的操作结束 (EOP) 位置 1。

如果在请求编程、擦除或读操作期间出现错误，则 FLASH_SR 寄存器中的以下错误标志位之一将置 1：

- PGAERR、PGPERR、ERSERR（编程错误标志）
- WRPERR（保护错误标志）

这种情况下，FLASH_CR 寄存器中的错误中断使能位 (ERRIE) 置 1，并且如果 FLASH_SR 寄存器中的操作错误位 (OPERR) 置 1，则将产生一个中断。

注： *如果检测到多个连续错误（例如，在对 Flash 进行 DMA 传输期间），则直到连续写操作请求结束，这些错误标志才会清零。*

表 6. Flash 中断请求

中断事件	事件标志	使能控制位
操作结束	EOP	EOPIE
写保护错误	WRPERR	ERRIE
编程错误	PGAERR、PGPERR、ERSERR	ERRIE

3.4 FLASH 选项字节

3.4.1 选项字节说明

选项字节由最终用户根据具体的应用要求进行配置。*表 7* 显示了选项字节在信息块中的构成。

选项字节可从用户配置存储器或选项字节寄存器中读取：

- *Flash 选项控制寄存器 (FLASH_OPTCR)*
- *Flash 选项控制寄存器 (FLASH_OPTCR1)*

表 7. 选项字节构成

AXI 地址	[63:16]	[15:0]
0x1FFF 0000	保留	ROP 和用户选项字节 (**RDP** & **USER**)
0x1FFF 0008	保留	IWDG_STOP、IWDG_STBY 和写保护 nWRP（扇区 0 到 7）和用户选项字节
0x1FFF 0010	保留	BOOT_ADD0
0x1FFF 0018	保留	BOOT_ADD1

用户和读保护选项字节

存储器地址：0x1FFF 0000

ST 编程值：0x5500AAFF

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

位 31:13 未使用。

位 15:8 **RDP**：读保护 (Read Out Protection)

读保护用于用户保护 Flash 中存储的软件代码。

0xAA：级别 0，无保护

0xCC：级别 2，芯片保护（禁止调试和从 RAM 启动）

其他：级别 1，存储器读保护（调试功能受限）

位 7 **nRST_STDBY**

0：进入待机模式时产生复位

1：不产生复位

位 6 **nRST_STOP**

0：进入停止模式时产生复位

1：不产生复位

位 5 **IWDG_SW**：独立看门狗选择 (Independant watchdog selection)

0：硬件独立看门狗

1：软件独立看门狗

位 4 **WWDG_SW**：窗口看门狗选择 (Window watchdog selection)

0：硬件窗口看门狗

1：软件窗口看门狗

位 3:2 **BOR_LEV**：BOR 复位级别 (BOR reset Level)

这些位包含激活/释放复位信号所需达到的供电电压阈值。通过对这些位执行写操作，可将新的 BOR 级别值编程到 Flash。

00：BOR 级别 3 (VBOR3)，欠压阈值级别 3

01：BOR 级别 2 (VBOR2)，欠压阈值级别 2

10：BOR 级别 1 (VBOR1)，欠压阈值级别 1

11：BOR 关闭，施加 POR/PDR 复位阈值电压

注：有关 BOR 特性的完整详情，请参见产品数据手册中的“电气特性”部分。

位 1:0 未使用

用户和写保护选项字节

存储器地址：0x1FFF 0008

ST 编程值：0x0000FFFF

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

位 31:16 未使用。

位 15 **IWDG_STOP**：在停止模式下冻结独立看门狗计数器 (Independent watchdog counter freeze in stop mode)

0：在停止模式下冻结 IWDG 计数器

1：在停止模式下激活 IWDG 计数器

位 14 **IWDG_STDBY**：在待机模式下冻结独立看门狗计数器 (Independent watchdog counter freeze in Standby mode)

1：在待机模式下激活 IWDG 计数器

位 13:8 未使用

位 7:0 **nWRPi**：扇区 i 的低有效写保护 (Non Write Protection of sector i)

0：激活扇区 i 的写保护

1：未激活扇区 i 的写保护

Boot 引脚 = 0 时的自举地址选项字节

存储器地址：0x1FFF 0010

ST 编程值：0xFF7F 0080（ITCM-FLASH 基址）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOT_ADD0[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 未使用。

位 15:0 **BOOT_ADD0[15:0]**：Boot 引脚 = 0 时的自举存储器基址 (Boot base address when Boot pin =0)
BOOT_ADD0[15:0] 对应地址 [29:14]。
自举基址仅支持 0x0000 0000 到 0x2004 FFFF 范围内的地址（地址为 16KB 的整数倍）。
示例：
BOOT_ADD0 = 0x0000：从 ITCM RAM (0x0000 0000) 自举
BOOT_ADD0 = 0x0040：从系统存储器 (0x0010 0000) 自举
BOOT_ADD0 = 0x0080：从 ITCM 接口上的 FLASH (0x0020 0000) 自举
BOOT_ADD0 = 0x2000：从 AXIM 接口上的 FLASH (0x0800 0000) 自举
BOOT_ADD0 = 0x8000：从 DTCM RAM (0x2000 0000) 自举
BOOT_ADD0 = 0x8004：从 SRAM1 (0x2001 0000) 自举
BOOT_ADD0 = 0x8013：从 SRAM2 (0x2004 C000) 自举

Boot 引脚 = 1 时的自举地址选项字节

存储器地址：0x1FFF 0018

ST 编程值：0xFFBF0040（系统存储器自举程序地址）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOT_ADD1[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 未使用

位 15:0 **BOOT_ADD1[15:0]**：Boot 引脚 = 1 时的自举存储器基址 (Boot base address when Boot pin =1)
BOOT_ADD1[15:0] 对应地址 [29:14]。
自举基址仅支持 0x0000 0000 到 0x2004 FFFF 范围内的地址（地址为 16KB 的整数倍）。
示例：
BOOT_ADD1 = 0x0000：从 ITCM RAM (0x0000 0000) 自举
BOOT_ADD1 = 0x0040：从系统存储器 (0x0010 0000) 自举
BOOT_ADD1 = 0x0080：从 ITCM 接口上的 FLASH (0x0020 0000) 自举
BOOT_ADD1 = 0x2000：从 AXIM 接口上的 FLASH (0x0800 0000) 自举
BOOT_ADD1 = 0x8000：从 DTCM RAM (0x2000 0000) 自举
BOOT_ADD1 = 0x8004：从 SRAM1 (0x2001 0000) 自举
BOOT_ADD1 = 0x8013：从 SRAM2 (0x2004 C000) 自举

3.4.2 选项字节编程

要针对此扇区执行任何操作，Flash 选项控制寄存器 (FLASH_OPTCR) 中的选项锁定位 (OPTLOCK) 必须清零。为了能够将该位清零，用户需要顺序执行以下步骤：

1. 在 Flash 选项密钥寄存器 (FLASH_OPTKEYR) 中写入 OPTKEY1 = 0x0819 2A3B
2. 在 Flash 选项密钥寄存器 (FLASH_OPTKEYR) 中写入 OPTKEY2 = 0x4C5D 6E7F

通过软件将 OPTLOCK 位置 1 后，可防止用户选项字节发生意外的擦除/编程操作。

修改用户选项字节

要修改用户选项值，请顺序执行以下步骤：

1. 检查 FLASH_SR 寄存器中的 BSY 位，以确认当前未执行任何 Flash 操作。
2. 在 FLASH_OPTCR 寄存器中写入所需的选项值。
3. 将 FLASH_OPTCR 寄存器中的选项启动位 (OPTSTRT) 置 1。
4. 等待 BSY 位清零。

注： *修改选项字节应该首先擦除信息块，然后以 FLASH_OPTCR 寄存器中包含的值对所有选项字节进行编程，从而修改选项值。*

3.5 Flash 保护

3.5.1 读保护 (RDP)

可对 Flash 中的用户区域实施读保护，以防不受信任的代码读取其中的数据。读保护分三个级别，具体定义如下：

- 级别 0：无读保护

  将 0xAA 写入读保护选项字节 (RDP) 时，读保护级别即设为 0。此时，在所有自举配置（用户 Flash 自举、调试或从 RAM 自举）中，均可执行对 Flash 或备份 SRAM 的读/写操作（如果未设置写保护）。

- 级别 1：使能读保护

  这是擦除选项字节后的默认读保护级别。将任意值（分别用于设置级别 0 和级别 2 的 0xAA 和 0xCC 除外）写入 RDP 选项字节时，即激活读保护级别 1。设置读保护级别 1 后：

  – 在连接调试功能或从 RAM 或系统存储器自举时，不能对 Flash 或备份 SRAM 进行访问（读取、擦除、编程）。读请求将导致总线错误。

  – 从 Flash 自举时，允许通过用户代码对 Flash 和备份 SRAM 进行访问（读取、擦除、编程）。

  激活级别 1 后，如果将保护选项字节 (RDP) 编程为级别 0，则将对 Flash 和备份 SRAM 执行全部擦除。因此，在取消读保护之前，用户代码区域会清零。批量擦除操作仅擦除用户代码区域。包括写保护在内的其它选项字节将不受影响。OTP 区域不受批量擦除操作的影响，同样保持不变。只有在已激活级别 1 并请求级别 0 时，才会执行批量擦除。当提高保护级别 (0->1、1->2、0->2) 时，不会执行批量擦除。

- 级别 2：禁止调试/芯片读保护

  将 0xCC 写入 RDP 选项字节时，可激活读保护级别 2。设置读保护级别 2 后：

  – 级别 1 提供的所有保护均有效。

  – 不再允许从 RAM 或系统存储器自举。

  – JTAG、SWV（单线查看器）、ETM 和边界扫描处于禁止状态。

  – 用户选项字节不能再进行更改。

  – 从 Flash 自举时，允许通过用户代码对 Flash 和备份 SRAM 进行访问（读取、擦除、编程）。

  存储器读保护级别 2 是不可更改的。激活级别 2 后，保护级别不能再降回级别 0 或级别 1。

注： *激活级别 2 后，将永久性禁止 JTAG 端口（相当于 JTAG 熔断）。这样，将无法执行边界扫描。意法半导体无法对设为保护级别 2 的器件做失效分析。*

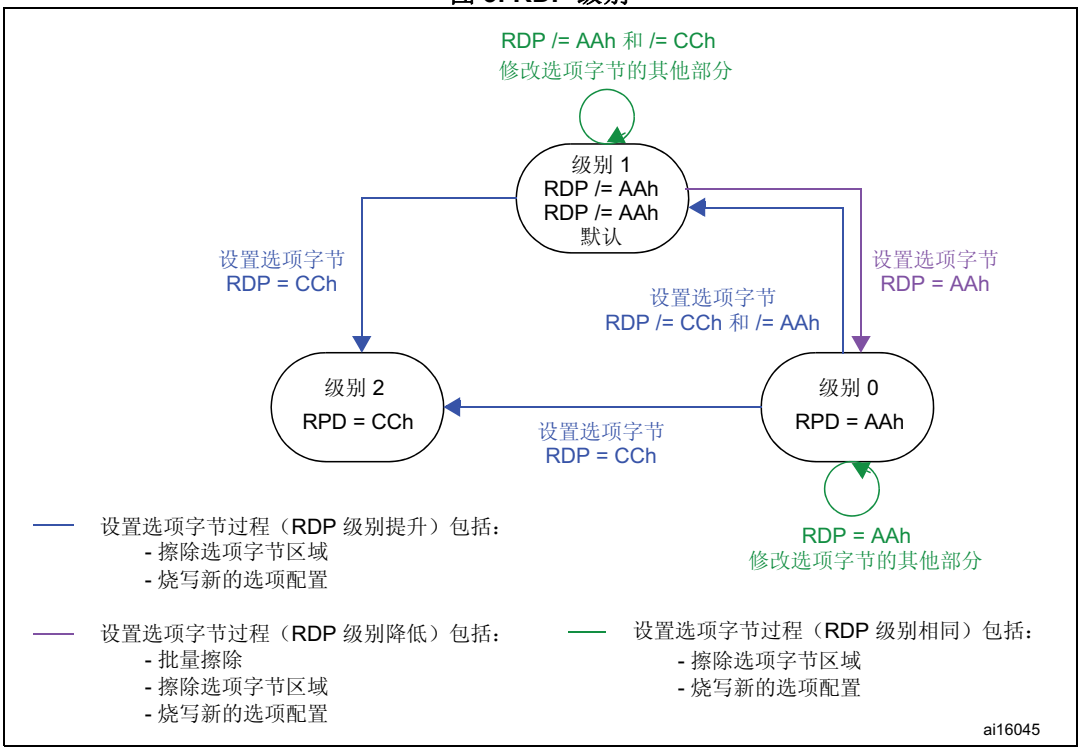
表 8. 不同读保护级别下的访问限制

存储区	保护级别	调试功能，从 RAM 或系统存储器自举			从 Flash 自举		
		读	写	擦除	读	写	擦除
主 Flash 和备份 SRAM	级别 1	否		否(1)	是		
	级别 2	否			是		
选项字节	级别 1	是			是		
	级别 2	否			否		
OTP	级别 1	否		不适用	是		不适用
	级别 2	否		不适用	是		不适用

1. 只有在 RDP 从级别 1 更改为级别 0 时，才会擦除主 Flash 和备份 SRAM。OTP 区域保持不变。

图 3 所示为 RDP 级别切换的过程。

图 3. RDP 级别



3.5.2 写保护

Flash 中有多达 8 个用户扇区具备写保护功能，可防止因程序指针错乱而发生意外的写操作。当 FLASH_OPTCR 或 FLASH_OPTCR1 寄存器中的低有效写保护位 nWRPi (0 ≤ i ≤ 7) 为低电平时，无法对相应的扇区执行擦除或编程操作。因此，如果某个扇区处于写保护状态，则无法对整个器件执行全部擦除。

如果尝试对 Flash 中处于写保护状态的区域执行擦除/编程操作（由写保护位保护的扇区、锁定的 OTP 区域或永远不能执行写操作的 Flash 区域，例如 ICP），则 FLASH_SR 寄存器中的写保护错误标志位 (WRPERR) 将置 1。

注： *选择存储器读保护级别（RDP 级别 = 1）后，如果已连接 CPU 调试功能（JTAG 调试或单线调试）或者正在从 RAM 执行启动代码，则即使 nWRPi = 1，也无法对 Flash 扇区 i 执行编程或擦除操作。*

写保护错误标志

如果对 Flash 的写保护区域执行擦除/编程操作，则 FLASH_SR 寄存器中的写保护错误标志位 (WRPERR) 将置 1。

如果请求执行擦除操作，则以下情况下 WRPERR 位置 1：

- 请求执行扇区擦除但扇区编号 SNB 字段无效。
- 请求执行批量擦除，但至少一个用户扇区通过选项位实施了写保护（FLASH_OPTCR 寄存器中的 MER = 0 并且 nWRi = 0，0 ≤ i ≤ 7 位）。
- 请求针对写保护扇区执行扇区擦除。（FLASH_OPTCR 寄存器中 SER = 1、SNB = i 且 nWRPi = 0，0 ≤ i ≤ 7 位）。
- Flash 处于读保护状态，但检测到擦除操作企图。

如果请求执行编程操作，则以下情况下 WRPERR 位置 1：

- 针对系统存储器或用户特定扇区的保留区域执行写操作。
- 针对信息块执行写操作。
- 针对通过选项位实施写保护的扇区执行写操作。
- 请求针对已锁定的 OTP 区域执行写操作。
- Flash 处于读保护状态，但检测到写操作企图。

3.6 一次性可编程字节

表 9 所示为 OTP 区域中一次性可编程 (OTP) 部分的构成。

表 9. OTP 区域构成

OTP 块	[255:224]	[223:193]	[192:161]	[160:128]	[127:96]	[95:64]	[63:32]	[31:0]	地址 字节 0
0	OTP0	OTP0	OTP0	OTP0	OTP0	OTP0	OTP0	OTP0	0x1FF0 F000
	OTP0	OTP0	OTP0	OTP0	OTP0	OTP0	OTP0	OTP0	0x1FF0 F020
1	OTP1	OTP1	OTP1	OTP1	OTP1	OTP1	OTP1	OTP1	0x1FF0 F040
	OTP1	OTP1	OTP1	OTP1	OTP1	OTP1	OTP1	OTP1	0x1FF0 F060
- - -	-	-	-	-	-	-	-	-	-
14	OPT14	OPT14	OPT14	OPT14	OPT14	OPT14	OPT14	OPT14	0x1FF0 F380
	OPT14	OPT14	OPT14	OPT14	OPT14	OPT14	OPT14	OPT14	0x1FF0 F3A0

表 9. OTP 区域构成（续）

OTP 块	[255:224]	[223:193]	[192:161]	[160:128]	[127:96]	[95:64]	[63:32]	[31:0]	地址 字节 0
15	OPT15	OPT15	OPT15	OPT15	OPT15	OPT15	OPT15	OPT15	0x1FF0 F3C0
	OPT15	OPT15	OPT15	OPT15	OPT15	OPT15	OPT15	OPT15	0x1FF0 F3E0
锁定块	保留	保留	保留	保留	LOCK15... LOCKB12	LOCK11... LOCKB8	LOCK7... LOCKB4	LOCK3... LOCKB0	0x1FF0 F400

OTP 区域划分为 16 个 64 字节的 OTP 数据块和 1 个 16 字节的 OTP 锁定块。OTP 数据块和锁定块均无法擦除。锁定块中包含 16 字节的 LOCKBi (0 ≤ i ≤ 15)，用于锁定相应的 OTP 数据块（块 0 到 15）。每个 OTP 数据块均可编程，除非相应的 OTP 锁定字节编程为 0x00。锁定字节的值只能是 0x00 和 0xFF，否则这些 OTP 字节无法正确使用。

3.7 FLASH 寄存器

3.7.1 Flash 访问控制寄存器 (FLASH_ACR)

Flash access control register

Flash 访问控制寄存器用于使能/关闭加速功能，并且可根据 CPU 频率控制 Flash 访问时间。

偏移地址：0x00
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.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	ARTRST	Res.	ARTEN	PRFTEN	Res.	Res.	Res.	Res.	LATENCY			
				rw		rw	rw					rw	rw	rw	rw

位 31:12 保留，必须保持清零。

位 11 **ARTRST**：ART Accelerator 复位 (ART Accelerator reset)
0：ART Accelerator 不复位
1：ART Accelerator 复位

位 10 保留，必须保持清零。

位 9 **ARTEN**：ART Accelerator 使能 (ART Accelerator Enable)
0：ART Accelerator 禁止
1：ART Accelerator 使能

位 8 **PRFTEN**：预取使能
0：禁止预取
1：使能预取

位 7:4 保留，必须保持清零。

位 3:0 **LATENCY[3:0]**：延迟 (Latency)
这些位表示 CPU 时钟周期与 Flash 访问时间之比。
0000：0 个等待周期
0001：1 个等待周期
0010：2 个等待周期
-
-
-
1110：14 个等待周期
1111：15 个等待周期

3.7.2 Flash 密钥寄存器 (FLASH_KEYR)

Flash key register

借助 Flash 密钥寄存器，可允许对 Flash 控制寄存器的访问，进而允许执行编程和擦除操作。

偏移地址：0x04
复位值：0x0000 0000

访问：无等待周期，按字访问

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

位 31:0 **FKEYR**：FPEC 密钥 (FPEC key)
要将 FLASH_CR 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：
a) KEY1 = 0x45670123
b) KEY2 = 0xCDEF89AB

3.7.3 Flash 选项密钥寄存器 (FLASH_OPTKEYR)

Flash option key register

借助 Flash 选项密钥寄存器，可在信息块中执行编程和擦除操作。

偏移地址：0x08
复位值：0x0000 0000

访问：无等待周期，按字访问

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

位 31:0 **OPTKEYR**：选项字节密钥 (Option byte key)

要将 FLASH_OPTCR 寄存器解锁并允许对其编程，必须顺序编程以下值：

a) OPTKEY1 = 0x08192A3B

b) OPTKEY2 = 0x4C5D6E7F

3.7.4 Flash 状态寄存器 (FLASH_SR)

Flash status register

Flash 状态寄存器提供正在执行的编程和擦除操作的相关信息。

偏移地址：0x0C
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.	Res.	Res.	BSY
															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.	ERSERR	PGPERR	PGAERR	WRPERR	Res.	Res.	OPERR	EOP
								rc_w1	rc_w1	rc_w1	rc_w1			rc_w1	rc_w1

位 31:17 保留，必须保持清零。

位 16 **BSY**：繁忙 (Busy)

该位指示 Flash 操作正在进行。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 Flash 操作
1：当前正在执行 Flash 操作

位 15:8 保留，必须保持清零。

位 7 **ERSERR**：擦除顺序错误 (Erase Sequence Error)

如果代码在控制寄存器未正确配置的情况下对 Flash 执行写访问，将由硬件为该位置 1。
写入 1 即可将该位清零。

位 6 **PGEPRR**：编程并行位数错误 (Programming parallelism error)

如果在编程期间数据访问类型（字节、半字、字和双字）与配置的并行位数 PSIZE（x8、x16、x32、x64）不符，将由硬件为该位置 1。
写入 1 即可将该位清零。

位 5 **PGAERR**：编程对齐错误 (Programming alignment error)
如果要编程的数据不能包含在同一个 128 位 Flash 行中，将由硬件为该位置 1。
写入 1 即可将该位清零。

位 4 **WRPERR**：写保护错误 (Write protection error)
如果要擦除/编程的地址属于 Flash 中处于写保护状态的区域，将由硬件为该位置 1。
写入 1 即可将该位清零。

位 3:2 保留，必须保持清零。

位 1 **OPERR**：操作错误 (Operation error)
如果检测到 Flash 操作（编程/擦除/读取）请求，但由于存在并行位数错误、对齐错误或写保护错误而无法运行，将由硬件对该位置 1。只有在使能错误中断 (ERRIE = 1) 后，该位才会置 1。

位 0 **EOP**：操作结束 (End of operation)
当成功完成一个或多个 Flash 操作（编程/擦除）时，由硬件将该位置 1。只有在使能操作结束中断 (EOPIE = 1) 后，该位才会置 1。
写入 1 即可将该位清零。

3.7.5 Flash 控制寄存器 (FLASH_CR)

Flash control register

Flash 控制寄存器用于配置和启动 Flash 操作。

偏移地址：0x10

复位值：0x8000 0000

访问：当前未执行任何 Flash 操作时无等待周期，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
LOCK	Res.	Res.	Res.	Res.	Res.	ERRIE	EOPIE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	STRT
rs						rw	rw								rs
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	PSIZE[1:0]		Res.	SNB[3:0]				MER	SER	PG
						rw	rw		rw	rw	rw	rw	rw	rw	rw

位 31 **LOCK**：锁定 (Lock)
该位只能写入 1。该位置 1 时，表示 FLASH_CR 寄存器已锁定。当检测到解锁序列时，由硬件将该位清零。
如果解锁操作失败，该位仍保持置 1，直到下一次复位。

位 30:26 保留，必须保持清零。

位 25 **ERRIE**：错误中断使能 (Error interrupt enable)
当 FLASH_SR 寄存器中的 OPERR 位置 1 后，可通过该位使能中断产生功能。
0：禁止产生错误中断
1：使能产生错误中断

位 24 **EOPIE**：操作结束中断使能 (End of operation interrupt enable)
当 FLASH_SR 寄存器中的 EOP 位置 1 后，可通过该位使能中断产生功能。
0：禁止产生中断
1：使能产生中断

位 23:17 保留，必须保持清零。

位 16 **STRT**：启动 (Start)
该位置 1 后可触发擦除操作。该位只能通过软件置 1，并在 BSY 位清零后随之清零。

位 15:10 保留，必须保持清零。

位 9:8 **PSIZE**：编程大小
这些位用于选择编程并行位数。
00 x8 编程
01 x16 编程
10 x32 编程
11 x64 编程

位 7 保留，必须保持清零。

位 6:3 **SNB[3:0]**：扇区编号 (Sector number)
这些位用于选择要擦除的扇区。
0000 扇区 0
0001 扇区 1
...
0111 扇区 7
不允许使用其它值

位 2 **MER**：批量擦除 (Mass Erase)
针对所有用户扇区激活擦除操作。

位 1 **SER**：扇区擦除(Sector Erase)
激活扇区擦除。

位 0 **PG**：编程 (Programming)
激活 Flash 编程。

3.7.6 Flash 选项控制寄存器 (FLASH_OPTCR)

Flash option control register

FLASH_OPTCR 寄存器用于修改用户选项字节。

偏移地址：0x14

复位值：0xC0FFAAFD。复位信号释放时，硬件将 Flash 中的值加载到这些选项字节。

访问：当前未执行任何 Flash 操作时无等待周期，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
IWDG_STOP	IWDG_STDBY	Res.	Res.	Res.	Res.	Res.	Res.	nWRP[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
RDP[7:0]								nRST_STDBY	nRST_STOP	IWDG_SW	WWDG_SW	BOR_LEV[1:0]		OPTSTRT	OPTLOCK
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rs	rs

位 31 **IWDG_STOP**：在停止模式下冻结独立看门狗计数器 (Independent watchdog counter freeze in stop mode)

0：在停止模式下冻结 IWDG 计数器
1：在停止模式下激活 IWDG 计数器

位 30 **IWDG_STDBY**：在待机模式下冻结独立看门狗计数器 (Independent watchdog counter freeze in Standby mode)

0：在待机模式下冻结 IWDG 计数器
1：在待机模式下激活 IWDG 计数器

位 29:24 保留，必须保持清零。

位 23:16 **nWRP[7:0]**：无写保护 (Not write protect)

这些位包含复位后扇区 0 到 7 的写保护选项字节值。通过对这些位执行写操作，可将新的写保护编程到 Flash。
0：激活扇区 i 的写保护
1：未激活扇区 i 的写保护

位 15:8 **RDP[7:0]**：读保护 (Read protect)

这些位包含复位后读保护选项字节的值。通过对这些位执行写操作，可将新的读保护值编程到 Flash。
0xAA：级别 0，未激活读保护
0xCC：级别 2，激活芯片读保护
其他：级别 1，激活存储器读保护

位 7:4 **USER**：用户选项字节 (User option bytes)

这些位包含复位后用户选项字节的值。通过对这些位执行写操作，可将新的用户选项字节值编程到 Flash。
位 7：nRST_STDBY
位 6：nRST_STOP
位 5：IWDG_SW
位 4：WWDG_SW

位 3:2 **BOR_LEV[1:0]**：BOR 复位级别 (BOR reset Level)

这些位包含激活/释放复位信号所需达到的供电电压阈值。可通过对这些位执行写操作，来编程新的 BOR 级别。BOR 默认为关闭。当电源电压 (V_{DD}) 降至所选 BOR 级别以下时，将产生器件复位。
00：BOR 级别 3 (VBOR3)，欠压阈值级别 3
01：BOR 级别 2 (VBOR2)，欠压阈值级别 2
10：BOR 级别 1 (VBOR1)，欠压阈值级别 1
11：BOR 关闭，施加 POR/PDR 复位阈值电压

注： 有关 BOR 特性的完整详情，请参见产品数据手册中的“电气特性”部分。

位 1 **OPTSTRT**：启动选项 (Option start)

该位置 1 后可触发用户选项操作。该位只能通过软件置 1，并在 BSY 位清零后随之清零。

位 0 **OPTLOCK**：锁定选项 (Option lock)

该位只能写入 1。该位置 1 时，表示 FLASH_OPTCR 寄存器已锁定。当检测到解锁序列时，由硬件将该位清零。
如果解锁操作失败，该位仍保持置 1，直到下一次复位。

注： 当修改选项字节 IWDG_SW、IWDG_STOP 或 IWDG_STDBY 时，需要执行系统复位才能使更改生效。

3.7.7 Flash 选项控制寄存器 (FLASH_OPTCR1)

Flash option control register

FLASH_OPTCR1 寄存器用于修改用户选项字节。

偏移地址：0x18

复位值：0xFF7F 0080 (ITCM-FLASH)。复位信号释放时，硬件将 Flash 中的值加载到这些选项字节。

访问：当前未执行任何 Flash 操作时无等待周期，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
BOOT_ADD1[15:0]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOT_ADD0[15:0]															
rw															

位 31:16 **BOOT_ADD1[15:0]**：Boot 引脚 = 0 时的自举基址 (Boot base address when Boot pin =1)

BOOT_ADD1[15:0] 对应地址 [29:14]。

可将自举存储器编程为 0x0000 0000 到 0x2004 FFFF 范围内的地址（地址为 16KB 的整数倍）。

示例：

BOOT_ADD1 = 0x0000：从 ITCM RAM (0x0000 0000) 自举

BOOT_ADD1 = 0x0040：从系统存储器 (0x0010 0000) 自举

BOOT_ADD1 = 0x0080：从 ITCM 接口上的 FLASH (0x0020 0000) 自举

BOOT_ADD1 = 0x2000：从 AXIM 接口上的 FLASH (0x0800 0000) 自举

BOOT_ADD1 = 0x8000：从 DTCM RAM (0x2000 0000) 自举

BOOT_ADD1 = 0x8004：从 SRAM1 (0x2001 0000) 自举

BOOT_ADD1 = 0x8013：从 SRAM2 (0x2004 C000) 自举

位 15:0 **BOOT_ADD0[15:0]**：Boot 引脚 = 0 时的基址 (Boot base address when Boot pin =0)

BOOT_ADD0[15:0] 对应地址 [29:14]。

可将自举基址编程为 0x0000 0000 到 0x2004 FFFF 范围内粒度为 16KB 的任何地址。

示例：

BOOT_ADD0 = 0x0000：从 ITCM RAM (0x0000 0000) 自举

BOOT_ADD0 = 0x0040：从系统存储器 (0x0010 0000) 自举

BOOT_ADD0 = 0x0080：从 ITCM 接口上的 FLASH (0x0020 0000) 自举

BOOT_ADD0 = 0x2000：从 AXIM 接口上的 FLASH (0x0800 0000) 自举

BOOT_ADD0 = 0x8000：从 DTCM RAM (0x2000 0000) 自举

BOOT_ADD0 = 0x8004：从 SRAM1 (0x2001 0000) 自举

BOOT_ADD0 = 0x8013：从 SRAM2 (0x2004 C000) 自举

3.7.8 Flash 接口寄存器映射

表 10. Flash 寄存器映射与复位值

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	**FLASH_ACR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARTRST	Res.	ARTEN	PRFTEN	Res.	Res.	Res.	Res.	LATENCY[3:0]			
	Reset value																					0		0	0					0	0	0	0
0x04	**FLASH_KEYR**	KEY[31:16]																KEY[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
0x08	**FLASH_ OPTKEYR**	OPTKEYR[31:16]																OPTKEYR[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	**FLASH_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BSY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERSERR	PGPERR	PGAERR	WRPERR	Res.	Res.	OPERR	EOP
	Reset value																0									0	0	0	0			0	0
0x10	**FLASH_CR**	LOCK	Res.	Res.	Res.	Res.	Res.	ERRIE	EOPIE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	STRT	Res.	Res.	Res.	Res.	Res.	Res.	PSIZE[1:0]		Res.	SNB[3:0]				MER	SER	PG
	Reset value	1						0	0								0							0	0		0	0	0	0	0	0	0
0x14	**FLASH_OPTCR**	IWDG_STOP	IWDG_STDBY	Res.	Res.	Res.	Res.	Res.	Res.	nWRP[7:0]								RDP[7:0]								nRST_STDBY	nRST_STOP	IWDG_SW	WWDG_SW	BOR_LEV[1:0]		OPTSTRT	OPTLOCK
	Reset value	1	1							1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	1	1	1	1	1	0	1
0x18	**FLASH_ OPTCR1**	BOOT_ADD1[15:0]																BOOT_ADD0[15:0]															
	Reset value	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

4 电源控制器 (PWR)

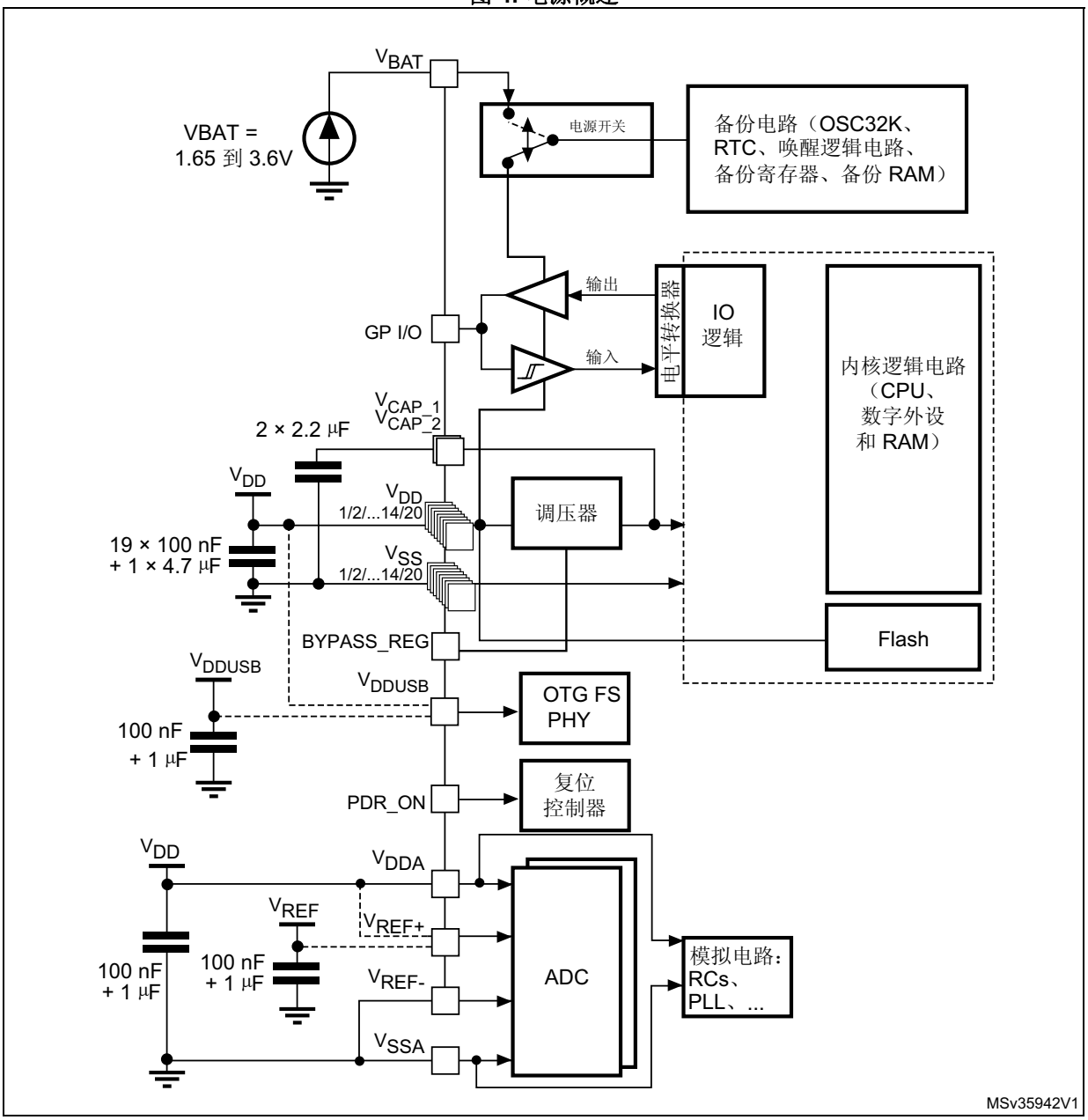
4.1 电源

器件的工作电压 (V_{DD}) 要求介于 1.8 V 到 3.6 V 之间。嵌入式线性调压器用于提供内部 1.2 V 数字电源。

当主电源 V_{DD} 断电时，可通过 V_{BAT} 电压为实时时钟 (RTC)、RTC 备份寄存器和备份 SRAM (BKP SRAM) 供电。

注： *根据工作期间供电电压的不同，某些外设可能只提供有限的功能和性能。有关更多详细信息，请参见 STM32F75xxx 和 STM32F74xxx 数据手册中“通用工作条件”一节。*

图 4. 电源概述



1. V_{DDA} 和 V_{SSA} 必须分别连接到 V_{DD} 和 V_{SS}。

4.1.1 独立 A/D 转换器电源和参考电压

为了提高转换精度，ADC 配有独立电源，可以单独滤波并屏蔽 PCB 上的噪声。

- ADC 电压源从单独的 V_{DDA} 引脚输入。
- V_{SSA} 引脚提供了独立的电源接地连接。

为了提高低压输入的精度，用户可以在 V_{REF} 上连接单独的 ADC 外部参考电压输入。V_{REF} 上的电压的范围为 1.8 V 到 V_{DDA}。

4.1.2 独立的 USB 收发器电源

V_{DDUSB} 是为全速收发器（USB OTG FS 和工作于 FS 模式的 USB OTG HS）供电的独立 USB 电源。它可以连接到 V_{DD} 或 USB 收发器的外部独立电源（3.0 V 到 3.6 V）（请参见 *图 5* 和 *图 6*）。例如，当器件通过 1.8 V 电压供电时，可在 V_{DDUSB} 上连接 3.3 V 独立电源。当 V_{DDUSB} 连接到单独的电源时，它独立于 V_{DD} 或 V_{DDA}，但它必须是最后提供且最先消失的电源。必须考虑以下 V_{DDUSB} 条件：

- 在上电阶段 ($V_{DD} < V_{DD_MIN}$)，V_{DDUSB} 应始终低于 V_{DD}
- 在掉电阶段 ($V_{DD} < V_{DD_MIN}$)，V_{DDUSB} 应始终低于 V_{DD}
- 必须遵循 V_{DDUSB} 上升时间和下降时间的速率规范（请参见表 21 和表 22）
- 在工作模式阶段，V_{DDUSB} 可低于或高于 V_{DD}：
  - 如果使用 USB (USB OTG_HS/OTG_FS)，则由 V_{DDUSB} 供电的相关 GPIO 在 V_{DDUSB_MIN} 到 V_{DDUSB_MAX} 之间工作。
  - V_{DDUSB} 为两个 USB 收发器（USB OTG_HS 和 USB OTG_FS）供电。如果应用项目仅使用一个 USB 收发器，则与另一个 USB 收发器相关的 GPIO 仍由 V_{DDUSB} 供电。
  - 如果未使用 USB (USB OTG_HS/OTG_FS)，则由 V_{DDUSB} 供电的相关 GPIO 在 V_{DD_MIN} 到 V_{DD_MAX} 之间工作。

图 5. 连接到 V_{DD} 电源的 V_{DDUSB}

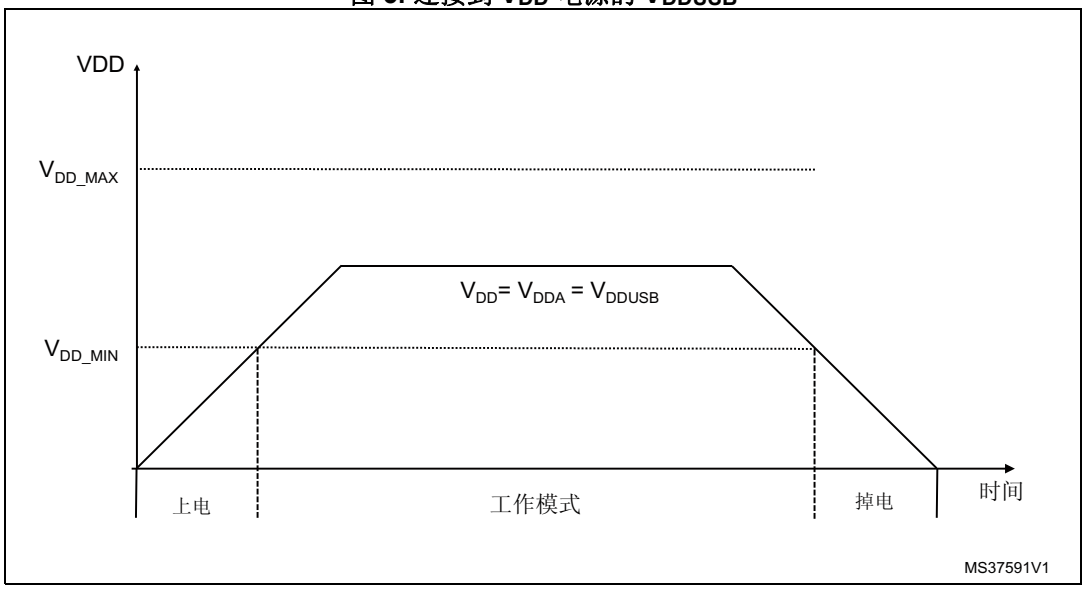
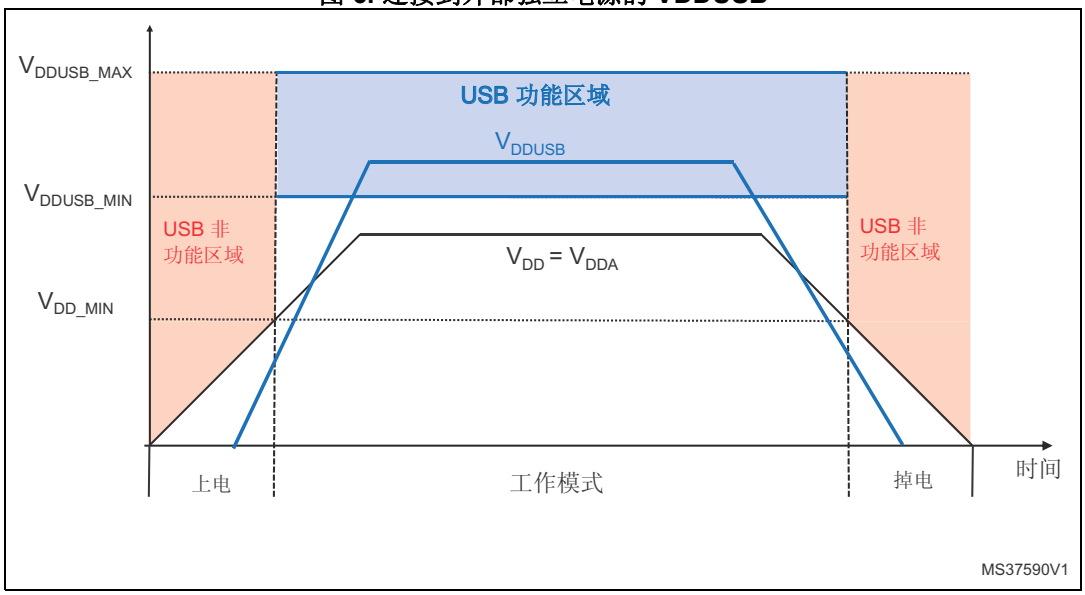


图 6. 连接到外部独立电源的 VDDUSB



4.1.3 电池备份域

备份域说明

要在 V_{DD} 关闭后保留 RTC 备份寄存器和备份 SRAM 的内容并为 RTC 供电，可以将 V_{BAT} 引脚连接到电池或其备用电压。

要使 RTC 即使在主数字电源 (V_{DD}) 关闭后仍然工作，V_{BAT} 引脚需为以下各模块供电：

- RTC
- LSE 振荡器
- 备份 SRAM（使能低功耗备份调压器时）
- PC13 到 PC15 I/O，以及 PI8 I/O（如果封装有该引脚）

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 V_{BAT} 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 V_{BAT} 引脚之间连接一个外部低压降二极管。**

如果应用中未使用任何外部电池，建议将 V_{BAT} 引脚连接到 V_{DD}，V_{DD} 上需并联 100 nF 去耦陶瓷电容。

通过 V_{DD} 对备份域供电时（模拟开关连接到 V_{DD}），可实现以下功能：

- PC14 和 PC15 可用作 GPIO 或 LSE 引脚
- PC13 和 PI8 可用作 GPIO 引脚（有关这些引配置的详细信息，请参见*表 146：RTC 引脚 PC13 配置*和*表 147：RTC 引脚 PI8 配置*）

注： *由于该开关的灌电流能力有限 (3 mA)，因此使用 GPIO PI8 和 PC13 到 PC15 时存在以下限制：每次只能有一个 I/O 用作输出，最大负载为 30 pF 时速率不得超过 2 MHz 并且这些 I/O 不能用作电流源（如用于驱动 LED）。*

通过 V_{BAT} 对备份域供电时（由于不存在 V_{DD}，内部电源开关连接到 V_{BAT}），可实现以下功能：

- PC14 和 PC15 只可用作 LSE 引脚
- PC13 可用作入侵引脚 (TAMP1)
- PI8 可用作入侵引脚 (TAMP2)

备份域访问

复位后，备份域（RTC 寄存器、RTC 备份寄存器和备份 SRAM）将受到保护，以防止意外的写访问。要使能对备份域的访问，请按以下步骤进行操作：

- 访问 RTC 和 RTC 备份寄存器

1. 将 RCC_APB1ENR 寄存器中的 PWREN 位置 1，使能电源接口时钟（请参见*第 5.3.13 节*）
2. 将 *PWR 电源控制寄存器 (PWR_CR1)* 中的 DBP 位置 1，使能对备份域的访问
3. 选择 RTC 时钟源：参见*第 5.2.8 节：RTC/AWU 时钟*
4. 通过对 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 RTCEN [15] 位进行编程，使能 RTC 时钟

- 访问备份 SRAM

1. 将 RCC_APB1ENR 寄存器中的 PWREN 位置 1，使能电源接口时钟（请参见*第 5.3.13 节*）
2. 将 *PWR 电源控制寄存器 (PWR_CR1)* 中的 DBP 位置 1，使能对备份域的访问
3. 通过将 *RCC APB1 外设复位寄存器 (RCC_APB1RSTR)* 中的 BKPSRAMEN 位置 1，使能备份 SRAM 时钟。

RTC 和 RTC 备份寄存器

实时时钟 (RTC) 是一个独立的 BCD 定时器/计数器。RTC 提供一个日历时钟、两个可编程闹钟中断，以及一个具有中断功能的可编程的周期唤醒标志。RTC 包含 32 个备份数据寄存器（128 字节），在检测到入侵事件时将复位。有关详细信息，请参见*第 29 节：实时时钟 (RTC)*。

备份 SRAM

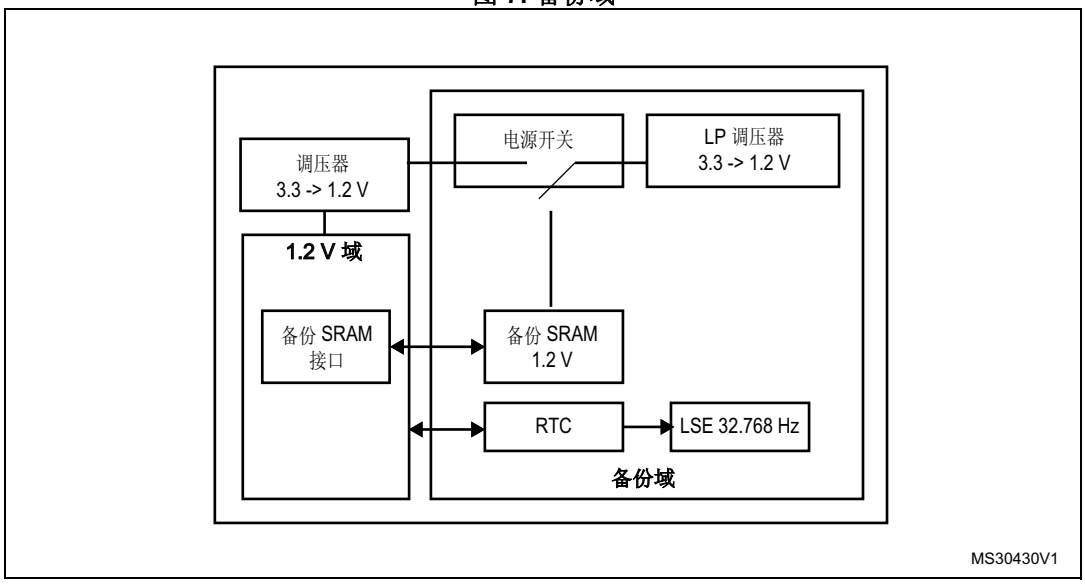
备份域包括可在 32 位、16 位、8 位模式下寻址的 4 KB 备份 SRAM。使能低功耗备份调压器时，即使处于待机或 V_{BAT} 模式，备份 SRAM 的内容也能保留。一直存在 V_{BAT} 时，可以将此备份 SRAM 视为内部 EEPROM。

通过 V_{DD}（模拟开关连接到 V_{DD}）对备份域供电时，备份 SRAM 将由 V_{DD} 而非 V_{BAT} 供电，以此延长电池寿命。

通过 V_{BAT}（由于不存在 V_{DD}，模拟开关连接到 V_{BAT}）对备份域供电时，备份 SRAM 通过专用的低功耗调压器供电。此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 V_{BAT} 模式是否需要备份 SRAM 功能。此调压器的掉电由专用位控制，即 PWR_CSR1 寄存器的 BRE 控制位（参见*第 4.4.2 节：PWR 电源控制/状态寄存器 (PWR_CSR1)*）。

入侵事件不会擦除备份 SRAM。备份 SRAM 设置了读保护，可防止用户对加密私钥等机密数据进行访问。擦除备份 SRAM 的唯一方法是在请求将保护级别从级别 1 更改为级别 0 时通过 Flash 接口实现。请参见读保护 (RDP) 选项字节的说明。

图 7. 备份域



4.1.4 调压器

嵌入式线性调压器为备份域和待机电路以外的所有数字电路供电。调压器输出电压约为 1.2 V。

此调压器需要将两个外部电容连接到两个专用引脚 V_{CAP_1} 和 V_{CAP_2}，所有封装都配有这两个引脚。为激活或停用调压器，必须将特定引脚连接到 V_{SS} 或 V_{DD}。具体引脚与封装有关。

通过软件激活时，调压器在复位后始终处于使能状态。根据应用模式的不同（运行、停止或待机模式），可采用三种不同的模式工作。

- 在**运行模式**下，主调压器为 1.2 V 域（内核、存储器和数字外设）提供全功率。在该模式下，调压器输出电压（约 1.2 V）可通过软件调节为不同的电压值（可通过 PWR_CR1 寄存器的 VOS[1:0] 位配置为级别 1、级别 2 和级别 3）。仅当关闭 PLL 且选择 HSI 或 HSE 时钟源作为系统时钟源时，才能修改级别。仅当 PLL 开启时，编程的新值才有效。若 PLL 关闭，将自动选择电压级别 3。

  器件运行在最高工作频率时，电压调节特性可使功耗得到优化。退出停止模式后，将自动选择电压级别 3。（请参见*第 4.4.1 节：PWR 电源控制寄存器 (PWR_CR1)*）。

  可使用两种工作模式：

  – **正常模式**：CPU 和内核逻辑在给定电压级别（级别 1、级别 2 和级别 3）下以最大频率工作。

  – **过驱动模式**：该模式允许 CPU 和内核逻辑在电压级别 1 和级别 2 下以高于正常模式的频率工作。

- 在**停止模式**下：主调压器或低功耗调压器为 1.2 V 域提供低功率电压，从而保存寄存器和内部 SRAM 的内容。

  还可以将调压器置于主调压器模式 (MR) 或低功耗模式 (LPR)。这两种模式均可通过软件按以下方式配置：

  – **正常模式**：正常泄漏模式下保留 1.2 V 域。当主调压器 (MR) 或低功耗调压器 (LPR) 使能时，该模式为默认模式。
  – **欠驱动模式**：低泄漏模式下保留 1.2 V 域。仅当主调压器或低功耗调压器处于低压模式时，才可使用该模式（请参见*表 11*）。
- 在**待机模式**下：调压器掉电。除待机电路和备份域外，寄存器和 SRAM 的内容都将丢失。

注： *当调压器处于被旁路状态时，不可使用过驱动和欠驱动模式。*

有关详细信息，请参见数据手册的调压器部分。

表 11. 调压器配置模式与器件工作模式(1)

调压器配置	运行模式	睡眠模式	停止模式	待机模式
正常模式	MR	MR	MR 或 LPR	-
过驱动模式(2)	MR	MR	-	-
欠驱动模式	-	-	MR 或 LPR	-
掉电模式	-	-	-	支持

1. “-”表示相应配置不可用。
2. 当 V_{DD} = 1.8 V 到 2.1 V 时，不可使用过驱动模式。

进入过驱动模式

当需要使系统进入过驱动模式时，建议在未运行关键任务并且系统时钟源为 HSI 或 HSE 时执行操作。要优化配置时间，请在 PLL 锁定阶段使能过驱动模式。

要进入过驱动模式，请遵循以下操作顺序：

1. 选择 HSI 或 HSE 作为系统时钟。
2. 配置 RCC_PLLCFGR 寄存器并将 RCC_CR 寄存器的 PLLON 位置 1。
3. 将 PWR_CR1 寄存器的 ODEN 位置 1 以使能过驱动模式，等待 PWR_CSR1 寄存器中的 ODRDY 标志位置 1。
4. 将 PWR_CR1 寄存器中的 ODSW 位置 1，使调压器从正常模式切换为过驱动模式。系统将在模式切换期间停止，但 PLL 时钟系统仍将在锁定期间运行。
5. 等待 PWR_CSR1 中的 ODSWRDY 标志位置 1。
6. 选择所需的 Flash 延迟以及 AHB 和 APB 预分频器。
7. 等待 PLL 锁定。
8. 将系统时钟切换为 PLL。
9. 使能不是由系统 PLL（I2S 时钟、LCD-TFT 时钟、SAI1 时钟、USB_48MHz 时钟....）生成的外设时钟。

注： *PLLI2S 和 PLLSAI 可与系统 PLL 同时配置。*

在过驱动切换激活期间，不应使能任何外设时钟。过驱动模式激活后，必须使能外设时钟。

进入停止模式会禁止过驱动模式以及 PLL。应用软件必须在退出停止模式后再次配置过驱动模式和 PLL。

退出过驱动模式

当需要使系统退出过驱动模式时，建议在未运行关键任务并且系统时钟源为 HSI 或 HSE 时执行操作。可通过以下两种操作顺序退出过驱动模式：

- 同时复位 PWR_CR1 寄存器中的 ODEN 位和 ODSW 位（顺序 1）。
- 首先复位 ODSW 位将调压器切换到正常模式，然后复位 ODEN 位禁止过驱动模式（顺序 2）。

顺序 1 示例：

1. 选择 HSI 或 HSE 作为系统时钟源。
2. 禁止不是由系统 PLL（I2S 时钟、LCD-TFT 时钟、SAI1 时钟、USB_48MHz 时钟....）生成的外设时钟。
3. 同时复位 PWR_CR1 寄存器中的 ODEN 位和 ODSW 位，将调压器切换回正常模式并禁止过驱动模式。
4. 等待 PWR_CSR1 中的 ODWRDY 标志位复位。

顺序 2 示例：

1. 选择 HSI 或 HSE 作为系统时钟源。
2. 禁止不是由系统 PLL（I2S 时钟、LCD-TFT 时钟、SAI1 时钟、USB_48MHz 时钟....）生成的外设时钟。
3. 复位 PWR_CR1 寄存器中的 ODSW 位，将调压器切换回正常模式。系统时钟在电压切换期间停止。
4. 等待 PWR_CSR1 中的 ODWRDY 标志位复位。
5. 将 PWR_CR1 寄存器中的 ODEN 位复位，禁止过驱动模式。

注： *在步骤 3 期间，ODEN 位保持置 1，过驱动模式保持使能，但为无效状态（ODSW 位复位）。但如果 ODEN 位复位，将禁止过驱动模式，且调压器将切换回初始电压。*

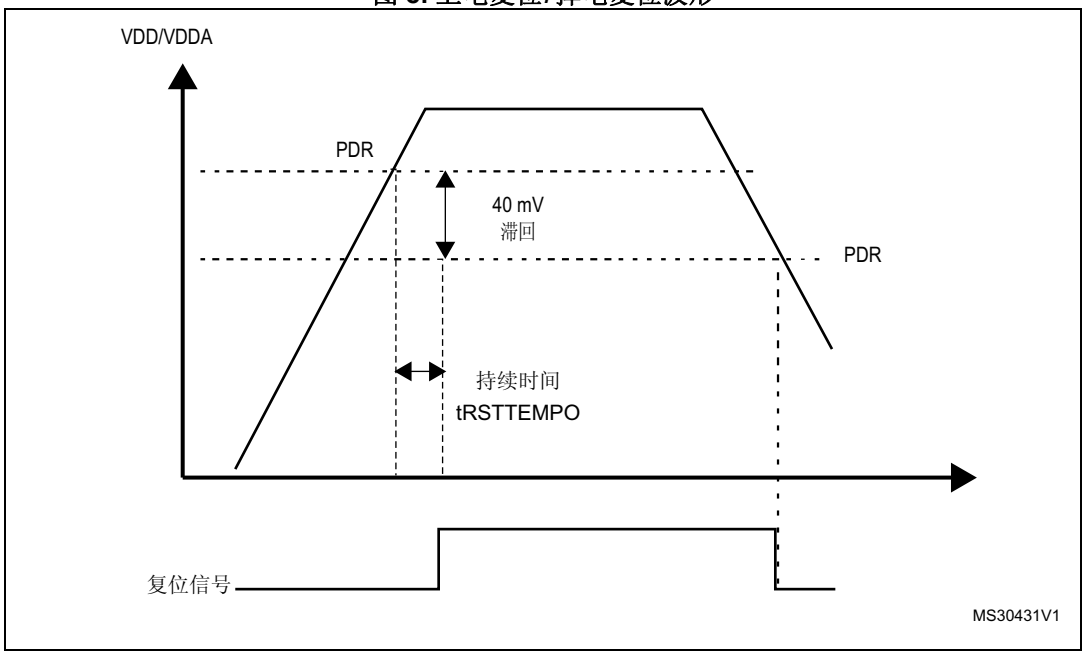
4.2 电源监控器

4.2.1 上电复位 (POR)/掉电复位 (PDR)

本器件内部集成有 POR/PDR 电路，可以从 1.8 V 开始正常工作。

当 V_{DD}/V_{DDA} 低于指定阈值 $V_{POR/PDR}$ 时，器件无需外部复位电路便会保持复位模式。有关上电/掉电复位阈值的相关详细信息，请参见数据手册的电气特性部分。

图 8. 上电复位/掉电复位波形



4.2.2 欠压复位 (BOR)

上电期间，欠压复位 (BOR) 将使器件保持复位状态，直到电源电压达到指定的 V_{BOR} 阈值。

V_{BOR} 通过器件选项字节进行配置。BOR 默认为关闭。可以选择 3 个可编程的 V_{BOR} 阈值级别：

- BOR 级别 3 (VBOR3)。欠压阈值级别 3。
- BOR 级别 2 (VBOR2)。欠压阈值级别 2。
- BOR 级别 1 (VBOR1)。欠压阈值级别 1。

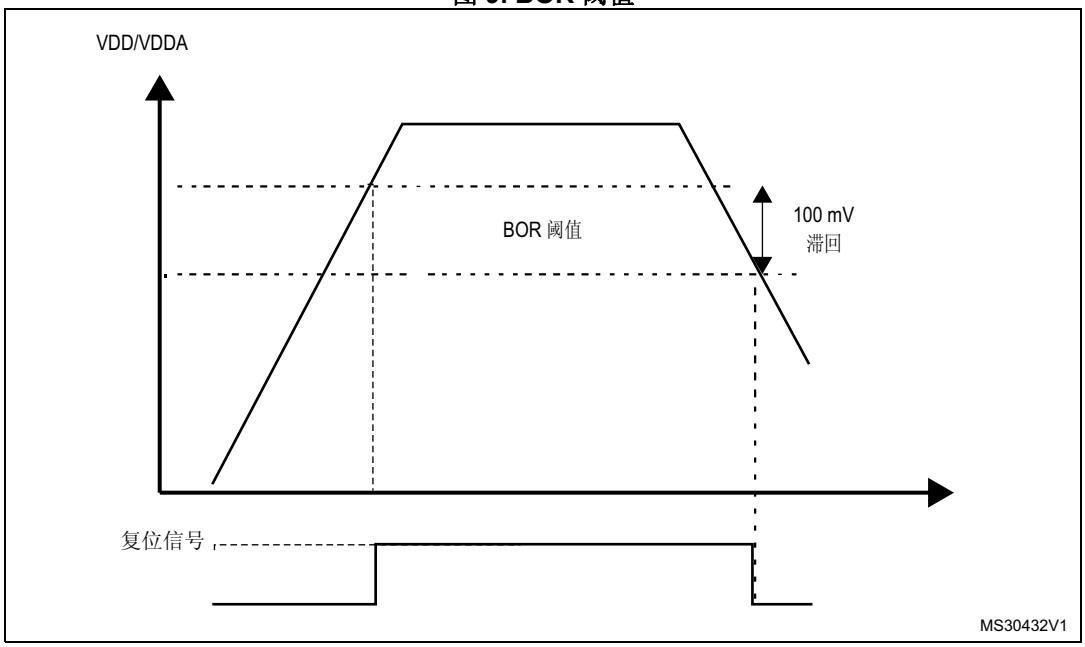
注： *有关 BOR 特性的完整详情，请参见器件数据手册中的“电气特性”部分。*

当电源电压 (V_{DD}) 降至所选 V_{BOR} 阈值以下时，将使器件复位。

通过对器件选项字节进行编程可以禁止 BOR。在这种情况下，POR/PDR 将在随后监视上电和掉电（请参见*第 4.2.1 节：上电复位 (POR)/掉电复位 (PDR)*）。

BOR 阈值滞回电压约为 100 mV（电源电压的上升沿与下降沿之间）。

图 9. BOR 阈值



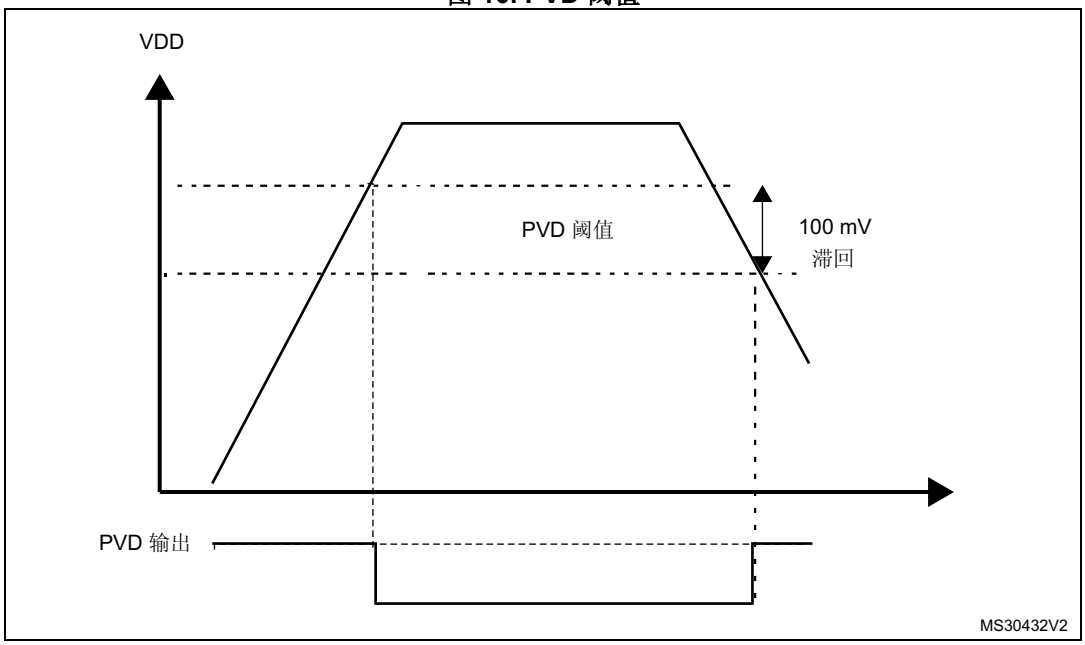
4.2.3 可编程电压检测器 (PVD)

可以使用 PVD 监视 V_{DD} 电源，方法是将其与 *PWR 电源控制寄存器 (PWR_CR1)* 中的 PLS[2:0] 位所选的阈值进行比较。

通过设置 PVDE 位来使能 PVD。

PWR 电源控制/状态寄存器 (PWR_CSR1) 中提供了 PVDO 标志位，用于指示 V_{DD} 是大于还是小于 PVD 阈值。该事件内部连接到 EXTI 线 16，如果通过 EXTI 寄存器使能，则可以产生中断。当 V_{DD} 降至 PVD 阈值以下以及/或者当 V_{DD} 升至 PVD 阈值以上时，可以产生 PVD 输出中断，具体取决于 EXTI 线 16 上升沿/下降沿的配置。该功能的用处之一就是可以在中断服务程序中执行紧急关闭系统的任务。

图 10. PVD 阈值



4.3 低功耗模式

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

器件有三个低功耗模式：

- 睡眠模式（Cortex®-M7 内核停止，外设保持运行）
- 停止模式（所有时钟都停止）
- 待机模式（1.2 V 域断电）

此外，可通过下列方法之一降低运行模式的功耗：

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

进入低功耗模式

当 MCU 执行 WFI（等待中断）或 WFE（等待事件）指令，或者当 Cortex®-M7 系统控制寄存器中的 SLEEPONEXIT 位在从 ISR 恢复期间置 1 时，将进入低功耗模式。

仅当没有中断和事件挂起时，才能通过 WFI 或 WFE 进入低功耗模式。

退出低功耗模式

在睡眠和停止模式下，MCU 根据进入低功耗模式的方式退出低功耗模式：

- 如果使用 WFI 指令或从 ISR 恢复的方式进入低功耗模式，则通过 NVIC 应答的任何外设中断均可唤醒器件。
- 如果使用 WFE 指令进入低功耗模式，则 MCU 会在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®-M7 系统控制寄存器中的 SEVONPEND = 0 时：通过使能外设控制寄存器和 NVIC 中的中断。当 MCU 从 WFE 恢复时，外设中断挂起位和 NVIC 外设 IRQ 通道挂起位（位于 NVIC 中断清除挂起寄存器中）必须清零。只有优先级足够高的 NVIC 中断才能唤醒和中断 MCU。

    当 Cortex®-M7 系统控制寄存器中的 SEVONPEND = 1 时：通过使能外设控制寄存器和 NVIC（可选）中的中断。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（位于 NVIC 中断清除挂起寄存器中）必须清零。所有 NVIC 中断（甚至已禁止的 NVIC 中断）都将唤醒 MCU。只有优先级足够高的已使能 NVIC 中断才能唤醒和中断 MCU。
  - 事件

    这通过在事件模式下配置 EXTI 线来完成。当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置 1，不必清除 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。

MCU 退出待机低功耗模式的方式包括：外部复位（NRST 引脚）、IWDG 复位、已使能 WKUPx 引脚之一的上升沿或者 RTC 事件（请参见*图 282：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已获取等）后的方式重新执行。
只有优先级足够高的已使能 NVIC 中断才能唤醒和中断 MCU。

表 12. 低功耗模式汇总

模式名称	进入	唤醒	对 1.2 V 域时钟的影响	对 V_{DD} 域时钟的影响	调压器
睡眠（立即休眠或退出时休眠）	WFI	任意中断	CPU CLK 关闭对其它时钟或模拟时钟源无影响	无	开启
	WFE	唤醒事件			
停止	SLEEPDEEP 位 + WFI 或 WFE	任意 EXTI 线（在 EXTI 寄存器中配置，内部线和外部线）	所有 1.2 V 域时钟都关闭	HSI 和 HSE 振荡器关闭	主调压器或低功耗调压器（取决于*PWR 电源控制寄存器 (PWR_CR1)*）
待机	PDDS 位 + SLEEPDEEP 位 + WFI 或 WFE	WKUP 引脚上升沿或下降沿、RTC 闹钟（闹钟 A 或闹钟 B）、RTC 唤醒事件、RTC 入侵事件、RTC 时间戳事件、NRST 引脚外部复位、IWDG 复位			关闭

表 13. 所有模式下的特性 (1)

外设	运行	睡眠	停止		待机		VBAT
				唤醒		唤醒	
CPU	Y	-	-		-		-
Flash 访问	Y	Y	-		-		-
DTCM RAM	Y	Y	Y		-		
ITCM RAM	Y	Y	Y		-		
SRAM1	Y	Y	Y		-		-
SRAM2	Y	Y	Y		-		-
FMC	O	O	-		-		-
QUADSPI	O	O	-		-		-
备份寄存器	Y	Y	Y		Y		Y
备份 RAM	Y	Y	Y		Y		Y
欠压复位 (BOR)	Y	Y	Y	Y	Y	Y	
可编程电压检测器 (PVD)	O	O	O	O	-		-
高速内部时钟 (HSI)	O	O	(2)		-		-
高速外部时钟 (HSE)	O	O	-				-
低速内部时钟 (LSI)	O	O	O		O		-
低速外部时钟 (LSE)	O	O	O		O		O
RTC	O	O	O	O	O	O	O
RTC 入侵引脚的数量	3	3	3	3	3	3	2
CRC 计算单元	O	O	-		-		-
GPIO	Y	Y	Y	Y		6 个引脚	2 个入侵
DMA	O	O	-		-		-
Chrom-ART 加速器 (DMA2D)	O	O	-		-		-
LCD-TFT	O	O	-	-	-		-
DCMI	O	O	-		-		-
USARTx (x=1..8)	O	O	-		-		-
I2Cx (x=1,2,3,4)	O	O	-		-		-
SPIx (x=1..6)	O	O	-		-		-
SAIx (x=1,2)	O	O	-		-		-
SPDIFRX	O	O	-		-		-
ADCx (x=1,2,3)	O	O	-		-		-
DACx (x=1,2)	O	O	-		-		-
温度传感器	O	O	-		-		-
定时器 (TIMx)	O	O	-		-		-

表 13. 所有模式下的特性（续）(1)

外设	运行	睡眠	停止		待机		VBAT
				唤醒		唤醒	
低功耗定时器 1 (LPTIM1)	O	O	O	O	-		-
独立看门狗 (IWDG)	O	O	O	O	O	O	-
窗口看门狗 (WWDG)	O	O	-		-		-
Systick 定时器	O	O	-		-		-
随机数发生器 (RNG)	O	O	-		-		-
加密处理器 (CRYP)							
散列处理器 (HASH)	O	O	-		-		-
SDMMC	O	O	-		-		-
CANx (x=1,2)	O	O	-		-		-
USB OTG FS	O	O	-		-		-
USB OTG HS	O	O	-		-		-
以太网	O	O	-		-		-
HDMI-CEC	O	O	-		-		-

1. 图例：Y = 支持（使能）。O = 可选（默认情况下禁止。可通过软件使能）。- = 不提供。
2. 一些能够从停止模式唤醒的外设可请求将 HSI 使能。在这种情况下，HSI 由外设唤醒，并且仅响应请求它的外设。当外设不再需要 HSI 时，HSI 将自动关闭。

4.3.1 调试模式

默认情况下，如果使用调试功能时器件进入停止模式或待机模式，调试连接将中断。这是因为 Cortex®-M7 内核时钟停止了。

不过，通过设置 DBGMCU_CR 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。有关详细信息，请参见*第 40.16.1 节：对低功耗模式的调试支持*。

4.3.2 运行模式

降低系统时钟速度

在运行模式下，可通过对预分频寄存器编程来降低系统时钟（SYSCLK、HCLK、PCLK1 和 PCLK2）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。

有关详细信息，请参见*第 5.3.3 节：RCC 时钟配置寄存器 (RCC_CFGR)*。

外设时钟门控

在运行模式下，可随时停止各外设和存储器的 HCLKx 和 PCLKx 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 AHB1 外设时钟使能寄存器 (RCC_AHB1ENR)、AHB2 外设时钟使能寄存器 (RCC_AHB2ENR) 和 AHB3 外设时钟使能寄存器 (RCC_AHB3ENR) 进行控制（请参见*第 5.3.10 节：RCC AHB1 外设时钟寄存器 (RCC_AHB1ENR)*、*第 5.3.11 节：RCC AHB2 外设时钟使能寄存器 (RCC_AHB2ENR)* 和 *第 5.3.12 节：RCC AHB3 外设时钟使能寄存器 (RCC_AHB3ENR)*）。

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。

4.3.3 低功耗模式

进入低功耗模式

当 MCU 执行 WFI（等待中断）或 WFE（等待事件）指令时，或者当 Cortex®-M7 系统控制寄存器中的 SLEEPONEXIT 位在从 ISR 恢复时置 1 时，将进入低功耗模式。

退出低功耗模式

在睡眠和停止模式下，MCU 根据进入低功耗模式的方式退出低功耗模式：

- 如果使用 WFI 指令或从 ISR 恢复的方式进入低功耗模式，则通过 NVIC 应答的任何外设中断均可唤醒器件
- 如果使用 WFE 指令进入低功耗模式，则 MCU 会在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®-M7 系统控制寄存器中的 SEVEONPEND=0 时。

    通过使能外设控制寄存器和 NVIC 中的中断。当 MCU 从 WFE 恢复时，外设中断挂起位和 NVIC 外设 IRQ 通道挂起位（位于 NVIC 中断清除挂起寄存器中）必须清零。

    只有优先级足够高的 NVIC 中断才能唤醒和中断 MCU。

    - 当 Cortex®-M7 系统控制寄存器中的 SEVEONPEND=1 时。

    通过使能外设控制寄存器和 NVIC（可选）中的中断。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（位于 NVIC 中断清除挂起寄存器中）必须清零。

    所有 NVIC 中断（甚至已禁止的 NVIC 中断）都将唤醒 MCU。

    只有优先级足够高的已使能 NVIC 中断才能唤醒和中断 MCU。

    - 事件

    在事件模式下配置 EXTI 线。当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置 1，不必清除 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。

在待机模式下，MCU 退出低功耗模式的方式包括：外部复位（NRST 引脚）、IWDG 复位、已使能 WKUPx 引脚之一的上升沿或者 RTC 事件（请参见*图 282：RTC 框图*）。

4.3.4 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

执行 WFI（等待中断）或 WFE（等待事件）指令即可进入睡眠模式。根据 Cortex®-M7 系统控制寄存器中 SLEEPONEXIT 位的设置，可以通过两种方案选择睡眠模式进入机制：

- 立即休眠：如果 SLEEPONEXIT 位清零，MCU 将在执行 WFI 或 WFE 指令时立即进入睡眠模式。
- 退出时休眠：如果 SLEEPONEXIT 位置 1，MCU 将在退出优先级最低的 ISR 时立即进入睡眠模式。

有关如何进入睡眠模式的详细信息，请参见*表 14* 和*表 15*。

退出睡眠模式

如果使用 WFI 指令进入睡眠模式，则嵌套向量中断控制器 (NVIC) 确认的任意外设中断都会将器件从睡眠模式唤醒。

如果使用 WFE 指令进入睡眠模式，MCU 将在有事件发生时立即退出睡眠模式。唤醒事件可通过以下方式产生：

- 在外设的控制寄存器使能一个中断，但不在 NVIC 中使能，同时使能 Cortex®-M7 系统控制寄存器中的 SEVONPEND 位。当 MCU 从 WFE 恢复时，需要清除相应外设的中断挂起位和外设 NVIC 中断通道挂起位（在 NVIC 中断清除挂起寄存器中）。
- 配置一个外部或内部 EXTI 线为事件模式。当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置 1，不必清除相应外设的中断挂起位或 NVIC 中断通道挂起位。

由于没有在进入/退出中断时浪费时间，此模式下的唤醒时间最短。

有关如何退出睡眠模式的详细信息，请参见*表 14* 和*表 15*。

表 14. 进入和退出立即休眠

立即休眠模式	说明
进入模式	WFI（等待中断）或 WFE（等待事件），且： – SLEEPDEEP = 0，及 – 没有中断（针对 WFI）和事件（针对 WFE）挂起。 请参见 Cortex®-M7 系统控制寄存器。
	从 ISR 恢复，条件为： – SLEEPDEEP = 0 及 – SLEEPONEXIT = 1， – 没有中断挂起。 请参见 Cortex®-M7 系统控制寄存器。

表 14. 进入和退出立即休眠（续）

立即休眠模式	说明
退出模式	如果使用 WFI 或从 ISR 恢复进入： 中断：请参见*表 43：STM32F75xxx 和 STM32F74xxx 向量表* 如果使用 WFE 进入且 SEVONPEND = 0 唤醒事件：请参见*第 11.3 节：唤醒事件管理* 如果使用 WFE 进入且 SEVONPEND = 1 中断（即使在 NVIC 中被禁止）：请参见*表 43：STM32F75xxx 和 STM32F74xxx 向量表*和唤醒中断（请参见*第 11.3 节：唤醒事件管理*）。
唤醒延迟	无

表 15. 进入和退出退出时休眠

退出时休眠	说明
进入模式	WFI（等待中断）或 WFE（等待事件），且： – SLEEPDEEP = 0，及 – 没有中断（针对 WFI）和事件（针对 WFE）挂起。 请参见 Cortex®-M7 系统控制寄存器。
	从 ISR 恢复，条件是： – SLEEPDEEP = 0，及 – SLEEPONEXIT = 1，及 – 没有中断挂起。 请参见 Cortex®-M7 系统控制寄存器。
退出模式	中断：请参见*表 43：STM32F75xxx 和 STM32F74xxx 向量表*
唤醒延迟	无

4.3.5 停止模式

停止模式基于 Cortex®-M7 深度睡眠模式与外设时钟门控。调压器既可以配置为正常模式，也可以配置为低功耗模式。在停止模式下，1.2 V 域中的所有时钟都会停止，PLL、HSI 和 HSE RC 振荡器也被禁止。内部 SRAM 和寄存器内容将保留。

在停止模式下，可通过使用 PWR_CR1 寄存器中的其它设置进一步降低功耗。但这将导致从停止模式唤醒的启动延迟更长（请参见*表 16*）。

表 16. 停止工作模式

调压器模式		UDEN[1:0] 位	MRUDS 位	LPUDS 位	LPDS 位	FPDS 位	唤醒延迟
正常模式	STOP MR（主调压器）	-	0	-	0	0	HSI RC 启动时间
	STOP MR- FPD	-	0	-	0	1	HSI RC 启动时间 + Flash 从掉电模式唤醒的时间
	STOP LP	-	0	0	1	0	HSI RC 启动时间 + 调压器从 LP 模式唤醒的时间
	STOP LP-FPD	-	-	0	1	1	HSI RC 启动时间 + Flash 从掉电模式唤醒的时间 + 调压器从 LP 模式唤醒的时间
欠驱动模式	STOP UMR-FPD	3	1	-	0	-	HSI RC 启动时间 + Flash 从掉电模式唤醒的时间 + 主调压器从欠驱动模式唤醒的时间 + 内核逻辑切换到正常模式的时间
	STOP ULP-FPD	3	-	1	1	-	HSI RC 启动时间 + Flash 从掉电模式唤醒的时间 + 调压器从 LP 欠驱动模式唤醒的时间 + 内核逻辑切换到正常模式的时间

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®-M7 系统控制寄存器中的 SLEEPDEEP 位置 1 时，将根据*进入低功耗模式的方式*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 17*。

微控制器进入停止模式时，将自动选择电压级别 3。要进一步降低停止模式的功耗，可将内部调压器设置为低功耗模式或低压模式。这可通过 *PWR 电源控制寄存器 (PWR_CR1)* 的 LPDS、MRUDS、LPUDS 和 UDEN 位进行设置。

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

如果正在访问 APB 域，停止模式的进入则延迟到 APB 访问结束后执行。

如果在进入停止模式前使能了过驱动模式，则过驱动模式将在激活停止模式后自动禁止。

在停止模式下，可以通过对各控制位进行编程来选择以下功能：

- 独立的看门狗 (IWDG)：IWDG 通过写入其密钥寄存器或使用硬件选项来启动。而且一旦启动便无法停止，除非复位。请参见*第 27 节：独立看门狗 (IWDG)* 中的*第 27.3 节*。
- 实时时钟 (RTC)：通过 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 RTCEN 位进行配置。
- 内部 RC 振荡器 (LSI RC)：通过 *RCC 时钟控制和状态寄存器 (RCC_CSR)* 中的 LSION 位进行配置。
- 外部 32.768 kHz 振荡器 (LSE OSC)：通过 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 LSEON 位进行配置。

在停止模式下，ADC 或 DAC 也会产生功耗，除非在进入停止模式前将其禁止。要禁止这些转换器，必须将 ADC_CR2 寄存器中的 ADON 位和 DAC_CR 寄存器中的 ENx 位都清零。

注： *在进入停止模式前，建议使能时钟安全系统 (CSS) 功能，以防止外部振荡器 (HSE) 故障影响内部 MCU 行为。*

退出停止模式

有关如何退出停止模式的详细信息，请参见*表 17*。

通过发出中断或唤醒事件退出停止模式时，将选择 HSI RC 振荡器作为系统时钟。

如果欠驱动模式已使能，则将在退出停止模式后自动禁止。

调压器在低功耗模式或低压模式下工作时，从停止模式唤醒的启动延时会更长。在停止模式下一直开启内部调压器虽然可以缩短启动时间，但功耗却增大。

调压器在欠驱动模式下工作时，从停止模式唤醒的启动延时将更长。

表 17. 进入和退出停止模式（STM32F75xxx 和 STM32F74xxx）

停止模式	说明
进入模式	WFI（等待中断）或 WFE（等待事件），且： – 没有中断（针对 WFI）和事件（针对 WFE）挂起。 – Cortex®-M7 系统控制寄存器中的 SLEEPDEEP 位置 1， – 电源控制寄存器 (PWR_CR) 中的 PDDS 位清零， – 配置 PWR_CR 中的 LPDS、MRUDS、LPUDS 和 UDEN 位，选择调压器模式（请参见*表 16：停止工作模式*）。
	从 ISR 恢复，条件为： – 没有中断挂起， – Cortex®-M7 系统控制寄存器中的 SLEEPDEEP 位置 1，及 – SLEEPONEXIT = 1，及 – PWR_CR1 中的 PDDS 清零。
	注： *要进入停止模式，所有 EXTI 线挂起位（在挂起寄存器 (EXTI_PR) 中）、所有外设中断挂起位、RTC 闹钟（闹钟 A 和闹钟 B）、RTC 唤醒、RTC 入侵和 RTC 时间戳标志位必须复位。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 恢复进入： 所有配置为中断模式的 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*第 268 页上的表 43：STM32F75xxx 和 STM32F74xxx 向量表*。 如果使用 WFE 进入且 SEVONPEND = 0： 所有配置为事件模式的 EXTI 线。请参见*第 274 页上的第 11.3 节：唤醒事件管理*。 如果使用 WFE 进入且 SEVONPEND = 1： – 在中断模式下配置的任何 EXTI 线（即使 NVIC 中的相应 EXTI 中断向量被禁止）。中断源可以是外部中断或具有唤醒功能的外设。请参见*第 268 页上的表 43：STM32F75xxx 和 STM32F74xxx 向量表*。 – 唤醒事件：请参见*第 274 页上的第 11.3 节：唤醒事件管理*
唤醒延迟	请参见*表 16：停止工作模式*。

4.3.6 待机模式

待机模式下可达到最低功耗。待机模式基于 Cortex®-M7 深度睡眠模式，其中调压器被禁止。因此 1.2 V 域断电。PLL、HSI 振荡器和 HSE 振荡器也将关闭。除备份域（RTC 寄存器、RTC 备份寄存器和备份 SRAM）和待机电路中的寄存器外，SRAM 和寄存器内容都将丢失（参见*图 9*）。

进入待机模式

当 Cortex®-M7 系统控制寄存器中的 SLEEPDEEP 位置 1 时，将根据进入低功耗模式的方式进入待机模式。

有关如何进入待机模式的详细信息，请参见*表 18*。

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：

- 独立的看门狗 (IWDG)：IWDG 通过写入其密钥寄存器或使用硬件选项来启动。而且一旦启动便无法停止，除非复位。请参见*第 27 节：独立看门狗 (IWDG)*中的*第 27.3 节*。
- 实时时钟 (RTC)：通过备份域控制寄存器 (RCC_BDCR) 中的 RTCEN 位进行配置。
- 内部 RC 振荡器 (LSI RC)：通过控制/状态寄存器 (RCC_CSR) 中的 LSION 位进行配置。
- 外部 32.768 kHz 振荡器 (LSE OSC)：通过备份域控制寄存器 (RCC_BDCR) 中的 LSEON 位进行配置。

退出待机模式

检测到外部复位（NRST 引脚）、IWDG 复位、WKUP 引脚上升沿或下降沿、RTC 闹钟、入侵事件或时间戳事件时，微控制器退出待机模式。从待机模式唤醒后，除 *PWR 电源控制/状态寄存器 (PWR_CSR1)* 外，所有寄存器都将复位。

从待机模式唤醒后，程序将按照复位（启动引脚采样、复位向量已获取等）后的方式重新执行。*PWR 电源控制/状态寄存器 (PWR_CSR1)* 中的 SBF 状态标志指示 MCU 已处于待机模式。

有关如何退出待机模式的详细信息，请参见*表 18*。

表 18. 进入和退出待机模式

待机模式	说明
进入模式	WFI（等待中断）或 WFE（等待事件），且： – Cortex®-M7 系统控制寄存器中的 SLEEPDEEP 置 1， – 电源控制寄存器 (PWR_CR) 中的 PDDS 位置 1， – 没有中断（针对 WFI）和事件（针对 WFE）挂起， – 电源控制寄存器 (PWR_CR) 中的 WUF 位清零， – 将与所选唤醒源（RTC 闹钟 A、RTC 闹钟 B、RTC 唤醒、RTC 入侵或 RTC 时间戳标志）对应的 RTC 标志清零
	从 ISR 恢复，条件为： – Cortex®-M7 系统控制寄存器中的 SLEEPDEEP 位置 1，及 – SLEEPONEXIT = 1，及 – 电源控制寄存器 (PWR_CR) 中的 PDDS 位置 1，及 – 没有中断挂起， – 电源控制/状态寄存器 (PWR_SR) 中的 WUF 位清零， – 将与所选唤醒源（RTC 闹钟 A、RTC 闹钟 B、RTC 唤醒、RTC 入侵或 RTC 时间戳标志）对应的 RTC 标志清零。

表 18. 进入和退出待机模式（续）

待机模式	说明
退出模式	WKUP 引脚上升沿或下降沿、RTC 闹钟（闹钟 A 和闹钟 B）、RTC 唤醒事件、入侵事件、时间戳事件、NRST 引脚外部复位 和 IWDG 复位。
唤醒延迟	复位阶段。

待机模式下的 I/O 状态

在待机模式下，除以下各部分以外，所有 I/O 引脚都处于高阻态：

- 复位引脚（仍可用）
- PC13（如果针对入侵、时间戳、RTC 闹钟输出或 RTC 时钟校准输出进行了配置）
- WKUP 引脚 (PA0/PA2/PC1/PC13/PI8/PI11)（如果使能）

4.3.7 对 RTC 复用功能进行编程以从停止模式和待机模式唤醒器件

RTC 复用功能可以从低功耗模式唤醒 MCU。

RTC 复用功能包括 RTC 闹钟（闹钟 A 和闹钟 B）、RTC 唤醒事件、RTC 入侵事件和 RTC 时间戳事件。

这些 RTC 复用功能可将系统从停止和待机低功耗模式唤醒。

通过使用 RTC 闹钟或 RTC 唤醒事件，无需依赖外部中断即可将系统从低功耗模式唤醒（自动唤醒模式）。

RTC 提供了可编程时基，便于定期从停止或待机模式唤醒器件。

为此，通过对 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 RTCSEL[1:0] 位进行编程，可以选择三个复用 RTC 时钟源中的其中两个：

- 低功耗 32.768 kHz 外部晶振 (LSE OSC)
  此时钟源提供的时基非常精确，功耗也非常低（典型条件下功耗小于 1 µA）
- 低功耗内部 RC 振荡器 (LSI RC)
  此时钟源的优势在于可以节省 32.768 kHz 晶振的成本。此内部 RC 振荡器非常省电。

通过 RTC 复用功能从停止模式唤醒器件

- 要通过 RTC 闹钟事件从停止模式唤醒器件，必须：
  a) 将 EXTI 线 17 配置为检测外部信号的上升沿（中断或事件模式）
  b) 使能 RTC_CR 寄存器中的 RTC 闹钟中断
  c) 配置 RTC 以生成 RTC 闹钟
- 要通过 RTC 入侵事件或时间戳事件从停止模式唤醒器件，必须：
  a) 将 EXTI 线 21 配置为检测外部信号的上升沿（中断或事件模式）
  b) 使能 RTC_CR 寄存器中的 RTC 时间戳中断，或者使能 RTC_TAFCR 寄存器中的 RTC 入侵中断
  c) 配置 RTC 以检测入侵事件或时间戳事件
- 要通过 RTC 唤醒事件从停止模式唤醒器件，必须：
  a) 将 EXTI 线 22 配置为检测外部信号的上升沿（中断或事件模式）
  b) 使能 RTC_CR 寄存器中的 RTC 唤醒中断
  c) 配置 RTC 以生成 RTC 唤醒事件

通过 RTC 复用功能从待机模式唤醒器件

- 要通过 RTC 闹钟事件从待机模式唤醒器件，必须：
  a) 使能 RTC_CR 寄存器中的 RTC 闹钟中断
  b) 配置 RTC 以生成 RTC 闹钟
- 要通过 RTC 入侵事件或时间戳事件从待机模式唤醒器件，必须：
  a) 使能 RTC_CR 寄存器中的 RTC 时间戳中断，或者使能 RTC_TAFCR 寄存器中的 RTC 入侵中断
  b) 配置 RTC 以检测入侵事件或时间戳事件
- 要通过 RTC 唤醒事件从待机模式唤醒器件，必须：
  a) 使能 RTC_CR 寄存器中的 RTC 唤醒中断
  b) 配置 RTC 以生成 RTC 唤醒事件

RTC 复用功能唤醒标志安全清零顺序

为了避免 RTC 复用功能所映射到的引脚发生跳变，并确保器件从停止模式和待机模式正常退出，建议在进入待机模式之前按照以下顺序进行操作：

- 使用 RTC 闹钟从低功耗模式唤醒器件时：
  a) 禁止 RTC 闹钟中断（RTC_CR 寄存器中的 ALRAIE 或 ALRBIE 位）
  b) 将 RTC 闹钟 (ALRAF/ALRBF) 标志清零
  c) 使能 RTC 闹钟中断
  d) 重新进入低功耗模式
- 使用 RTC 唤醒从低功耗模式唤醒器件时：
  a) 禁止 RTC 唤醒中断（RTC_CR 寄存器中的 WUTIE 位）
  b) 使能 RTC 唤醒中断
  c) 重新进入低功耗模式
- 使用 RTC 入侵从低功耗模式唤醒器件时：
  a) 禁止 RTC 入侵中断（RTC_TAFCR 寄存器中的 TAMPIE 位）
  b) 将入侵 (TAMP1F/TSF) 标志清零
  c) 使能 RTC 入侵中断
  d) 重新进入低功耗模式
- 使用 RTC 时间戳从低功耗模式唤醒器件时：
  a) 禁止 RTC 时间戳中断（RTC_CR 寄存器中的 TSIE 位）
  b) 将 RTC 时间戳 (TSF) 标志清零
  c) 使能 RTC 时间戳中断
  d) 重新进入低功耗模式

4.4 电源控制寄存器

4.4.1 PWR 电源控制寄存器 (PWR_CR1)

PWR power control register

偏移地址：0x00

复位值：0x0000 C000（通过从待机模式唤醒进行复位）

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.	UDEN[1:0]		ODSWEN	ODEN
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VOS[1:0]		ADCDC1	Res.	MRUDS	LPUDS	FPDS	DBP	PLS[2:0]			PVDE	CSBF	Res.	PDDS	LPDS
rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rc_w1		rw	rw

位 31:20 保留，必须保持复位值。

位 19:18 **UDEN[1:0]**：停止模式下的欠驱动使能 (Under-drive enable in stop mode)

这些位由软件置 1。这样可以降低停止模式的功耗，但会延长唤醒时间。

置 1 时，如果器件进入停止模式，会降低数字区域的泄漏损耗。

00：禁止欠驱动

01：保留

10：保留

11：使能欠驱动

位 17 **ODSWEN**：使能过驱动切换 (Over-drive switching enabled)

该位由软件置 1， 退出停止模式后或 ODEN 位复位时，该位由硬件自动清零。置 1 时，该位用于切换到过驱动模式。

要将 ODSWEN 位置 1 或复位，HSI 或 HSE 必须被选作系统时钟。

ODSWEN 位必须在 ODRDY 标志位置 1 时置 1，才能切换到过驱动模式。

0：禁止过驱动切换

1：使能过驱动切换

注： 在任一种过驱动切换模式下（使能或禁止），系统时钟将在内部电压设置期间停止。

位 16 **ODEN**：使能过驱动 (Over-drive enable)

该位由软件置 1， 退出停止模式后，该位由硬件自动清零。它用于使能过驱动模式，以达到更高频率。

要将 ODEN 位置 1 或复位，HSI 或 HSE 必须被选作系统时钟。当 ODEN 位置 1 时，应用必须首先等待过驱动就绪标志位 (ODRDY) 置 1，再将 ODSWEN 位置 1。

0：禁止过驱动

1：使能过驱动

位 15:14 **VOS[1:0]**：选择调压器输出电压级别 (Regulator voltage scaling output selection)

这些位用来控制内部主调压器的输出电压，以便在器件未以最大频率工作时使性能与功耗实现平衡（有关详细信息，请参见 STM32F75xxx 和 STM32F74xxx 数据手册）。

仅当 PLL 关闭时，才能修改这些位。只有当 PLL 开启时，编程的新值才有效。若 PLL 关闭，将自动选择电压级别 3。

00：保留（选择级别 3 模式）

01：级别 3 模式

10：级别 2 模式

11：级别 1 模式（复位值）

位 13 **ADCDC1**：
0：无影响。
1：有关如何使用此位的详细信息，请参见 AN4073。
注： 仅当在 2.7 V 到 3.6 V 的电源电压范围内工作时，才可以设置此位。

位 12 保留，必须保持复位值。

位 11 **MRUDS**：深度睡眠欠驱动模式下的主调压器 (Main regulator in deepsleep under-drive mode)
此位由软件置 1 和清零。
0：器件处于停止模式时，主调压器开启
1：器件处于停止欠驱动模式时，主调压器处于欠驱动模式，Flash 处于掉电状态。

位 10 **LPUDS**：深度睡眠欠驱动模式下的低功耗调压器 (Low-power regulator in deepsleep under-drive mode)
此位由软件置 1 和清零。
0：器件处于停止模式时，低功耗调压器开启
1：器件处于停止欠驱动模式时，低功耗调压器处于欠驱动模式（如果 LPDS 位置 1），Flash 处于掉电状态。

位 9 **FPDS**：停止模式下 Flash 掉电 (Flash power-down in Stop mode)
将此位置 1 时，Flash 将在器件进入停止模式后掉电。这样可以降低停止模式的功耗，但会延长重新启动时间。
0：器件进入停止模式后 Flash 不掉电
1：器件进入停止模式后 Flash 掉电

位 8 **DBP**：禁止备份域写保护 (Disable backup domain write protection)
在复位状态下，RCC_BDCR 寄存器、RTC 寄存器（包括备份寄存器）以及 PWR_CSR1 寄存器的 BRE 位均受到写访问保护。必须将此位置 1 才能使能对这些寄存器的写访问。
0：禁止对 RTC、RTC 备份寄存器和备份 SRAM 的访问
1：使能对 RTC、RTC 备份寄存器和备份 SRAM 的访问

位 7:5 **PLS[2:0]**：PVD 电平选择 (PVD level selection)
这些位由软件写入，用于选择电源电压检测器检测的电压阈值
000：2.0 V
001：2.1 V
010：2.3 V
011：2.5 V
100：2.6 V
101：2.7 V
110：2.8 V
111：2.9 V
注： 有关详细信息，请参见数据手册的电气特性。

位 4 **PVDE**：使能电源电压检测器 (Power voltage detector enable)
此位由软件置 1 和清零。
0：禁止 PVD
1：使能 PVD

位 3 **CSBF**：将待机标志清零 (Clear standby flag)
此位始终读为 0。
0：无影响
1：将 SDF 待机标志位清零（写）

位 2 保留，必须保持复位值

位 1 **PDDS**：掉电深度睡眠
此位由软件置 1 和清零。与 LPDS 位结合使用。
0：器件在 CPU 进入深度睡眠时进入停止模式。调压器状态取决于 LPDS 位。
1：器件在 CPU 进入深度睡眠时进入待机模式。

位 0 **LPDS**：低功耗深度睡眠 (Low-power deepsleep)
此位由软件置 1 和清零。与 PDDS 位结合使用。
0：停止模式下主调压器开启
1：停止模式下低功耗调压器开启

4.4.2 PWR 电源控制/状态寄存器 (PWR_CSR1)

PWR power control/status register

偏移地址：0x04

复位值：0x0000 0000（不通过从待机模式唤醒进行复位）

与标准的 APB 读操作相比，读取此寄存器需要更多的 APB 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.	UDRDY[1:0]		ODSWRDY	ODRDY
												rc_w1	rc_w1	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	VOS RDY	Res.	Res.	Res.	Res.	BRE	Res.	Res.	Res.	Res.	Res.	BRR	PVDO	SBF	WUIF
	r					rw						r	r	r	r

位 31:20 保留，必须保持复位值。

位 19:18 **UDRDY[1:0]**：欠驱动就绪标志位 (Under-drive ready flag)
在停止模式下使能欠驱动模式时，这些位由硬件置 1；将这些位编程为 1 可执行清零操作。
00：禁止欠驱动
01：保留
10：保留
11：在停止模式下激活欠驱动模式。

位 17 **ODSWRDY**：过驱动模式切换就绪 (Over-drive mode switching ready)
0：过驱动模式无效
1：过驱动模式在 1.2 V 域上的数字区域有效

位 16 **ODRDY**：过驱动模式就绪 (Over-drive mode ready)
0：过驱动模式未就绪。
1：过驱动模式已就绪。

位 14 **VOSRDY**：调压器输出分级电压选择就绪位 (Regulator voltage scaling output selection ready bit)
0：未就绪
1：已就绪

位[13:10] 保留，必须保持复位值。

位 9 **BRE**：使能备份调压器 (Backup regulator enable)

将此位置 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 SRAM 内容）。如果 BRE 复位，备份调压器关闭。仍可使用备份 SRAM，但在待机模式和 V_{BAT} 模式中其内容将丢失。将此位置 1 后，应用程序必须等待备份调压器就绪标志位 (BRR) 置 1，指示在待机模式和 V_{BAT} 模式下会保持写入 RAM 中的数据。

0：禁止备份调压器

1：使能备份调压器

注： 此位不会在器件从待机模式唤醒时复位，也不会通过系统复位或电源复位进行复位。

位[8:4] 保留，必须保持复位值。

位 3 **BRR**：备份调压器就绪 (Backup regulator ready)

由硬件置 1，用以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

注： 此位不会在器件从待机模式唤醒时复位，也不会通过系统复位或电源复位进行复位。

位 2 **PVDO**：PVD 输出 (PVD output)

此位通过硬件置 1 和清零。仅当通过 PVDE 位使能 PVD 时此位才有效。

0：V_{DD} 高于 PLS[2:0] 位选择的 PVD 阈值。

1：V_{DD} 低于 PLS[2:0] 位选择的 PVD 阈值。

注： PVD 在进入待机模式时停止。因此，进入待机模式或执行复位后，此位等于 0，直到 PVDE 位置 1。

位 1 **SBF**：待机标志 (Standby flag)

此位由硬件置 1，清零则只能通过 POR/PDR（上电复位/掉电复位）或将 *PWR 电源控制寄存器 (PWR_CR1)* 中的 CSBF 位置 1 来实现

0：器件未进入待机模式

1：器件已进入待机模式

位 0 **WUIF**：唤醒内部标志 (Wakeup internal flag)

在待机模式下检测到内部唤醒线上存在唤醒事件时，该位置 1。清除所有内部唤醒源时，该位清零。

0：未发生唤醒内部事件

1：检测到来自 RTC 闹钟（闹钟 A 和闹钟 B）、RTC 入侵事件、RTC 时间戳事件或 RTC 唤醒的唤醒事件。

4.4.3 PWR 电源控制/状态寄存器 2 (PWR_CR2)

PWR power control/status register 2

偏移地址：0x08

复位值：0x0000 0000（不通过从待机模式唤醒进行复位）

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.	WUPP6	WUPP5	WUPP4	WUPP3	WUPP2	WUPP1	Res.	Res.	CWUPF6	CWUPF5	CWUPF4	CWUPF3	CWUPF2	CWUPF1
		rw	rw	rw	rw	rw	rw			r	r	r	r	r	r

位 31:14 保留，始终读为 0。

位 13 **WUPP6**：PI11 唤醒引脚极性位 (Wakeup pin polarity bit for PI11)
这些位定义在外部唤醒引脚 PI11 上进行事件检测所用的极性。
0：在上升沿检测
1：在下降沿检测

位 12 **WUPP5**：PI8 唤醒引脚极性位 (Wakeup pin polarity bit for PI8)
这些位定义在外部唤醒引脚 PI8 上进行事件检测所用的极性。
0：在上升沿检测
1：在下降沿检测

位 11 **WUPP4**：PC13 唤醒引脚极性位 (Wakeup pin polarity bit for PC13)
这些位定义在外部唤醒引脚 PC13 上进行事件检测所用的极性。
0：在上升沿检测
1：在下降沿检测

位 10 **WUPP3**：PC1 唤醒引脚极性位 (Wakeup pin polarity bit for PC1)
这些位定义在外部唤醒引脚 PC1 上进行事件检测所用的极性。
0：在上升沿检测
1：在下降沿检测

位 9 **WUPP2**：PA2 唤醒引脚极性位 (Wakeup pin polarity bit for PA2)
这些位定义在外部唤醒引脚 PA2 上进行事件检测所用的极性。
0：在上升沿检测
1：在下降沿检测

位 8 **WUPP1**：PA0 唤醒引脚极性位 (Wakeup pin polarity bit for PA0)
这些位定义在外部唤醒引脚 PA0 上进行事件检测所用的极性。
0：在上升沿检测
1：在下降沿检测

位 7:6 保留，始终读为 0

位 5 **CWUPF6**：清除 PI11 唤醒引脚标志 (Clear Wakeup Pin flag for PI11)
此位始终读为 0
0：无影响
1：2 个系统时钟周期后将 WUPF 唤醒引脚标志清零

位 4 **CWUPF5**：清除 PI8 唤醒引脚标志 (Clear Wakeup Pin flag for PI8)
此位始终读为 0
0：无影响
1：2 个系统时钟周期后将 WUPF 唤醒引脚标志清零

位 3 **CWUPF4**：清除 PC13 唤醒引脚标志 (Clear Wakeup Pin flag for PC13)
此位始终读为 0
0：无影响
1：2 个系统时钟周期后将 WUPF 唤醒引脚标志清零

位 2 **CWUPF3**：清除 PC1 唤醒引脚标志 (Clear Wakeup Pin flag for PC1)
此位始终读为 0
0：无影响
1：2 个系统时钟周期后将 WUPF 唤醒引脚标志清零

位 1 **CWUPF2**：清除 PA2 唤醒引脚标志 (Clear Wakeup Pin flag for PA2)
此位始终读为 0
0：无影响
1：2 个系统时钟周期后将 WUPF 唤醒引脚标志清零

位 0 **CWUPF1**：清除 PA0 唤醒引脚标志 (Clear Wakeup Pin flag for PA0)
此位始终读为 0
0：无影响
1：2 个系统时钟周期后将 WUPF 唤醒引脚标志清零

4.4.4 PWR 电源控制寄存器 2 (PWR_CSR2)

PWR power control register 2

偏移地址：0x0C

复位值：0x0000 0000（通过从待机模式唤醒进行复位，由 RESET 引脚复位的唤醒标志除外）

与标准的 APB 读操作相比，读取此寄存器需要更多的 APB 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.	EWUP6	EWUP5	EWUP4	EWUP3	EWUP2	EWUP1	Res.	Res.	WUPF6	WUPF5	WUPF4	WUPF3	WUPF2	WUPF1
		rw	rw	rw	rw	rw	rw			r	r	r	r	r	r

位 31:14 保留，始终读为 0。

位 13 **EWUP6**：使能 PI11 唤醒引脚 (Enable Wakeup pin for PI11)
此位由软件置 1 和清零。
0：WKUP 引脚 PI11 上的事件不会将器件从待机模式唤醒。
1：WKUP 引脚 PI11 上的上升沿或下降沿将系统从待机模式唤醒。

位 12 **EWUP5**：使能 PI8 唤醒引脚 (Enable Wakeup pin for PI8)
此位由软件置 1 和清零。
0：WKUP 引脚 PI8 上的事件不会将器件从待机模式唤醒。
1：WKUP 引脚 PI8 上的上升沿或下降沿将系统从待机模式唤醒。

位 11 **EWUP4**：使能 PC13 唤醒引脚 (Enable Wakeup pin for PC13)
此位由软件置 1 和清零。
0：WKUP 引脚 PC13 上的事件不会将器件从待机模式唤醒。
1：WKUP 引脚 PC13 上的上升沿或下降沿将系统从待机模式唤醒。

位 10 **EWUP3**：使能 PC1 唤醒引脚 (Enable Wakeup pin for PC1)
此位由软件置 1 和清零。
0：WKUP 引脚 PC1 上的事件不会将器件从待机模式唤醒。
1：WKUP 引脚 PC1 上的上升沿或下降沿将系统从待机模式唤醒。

位 9 **EWUP2**：使能 PA2 唤醒引脚 (Enable Wakeup pin for PA2)
此位由软件置 1 和清零。
0：WKUP 引脚 PA2 上的事件不会将器件从待机模式唤醒。
1：WKUP 引脚 PA2 上的上升沿或下降沿将系统从待机模式唤醒。

位 8 **EWUP1**：使能 PA0 唤醒引脚 (Enable Wakeup pin for PA0)
此位由软件置 1 和清零。
0：WKUP 引脚 PA0 上的事件不会将器件从待机模式唤醒。
1：WKUP 引脚 PA0 上的上升沿或下降沿将系统从待机模式唤醒。

位 7:6 保留，始终读为 0

位 5 **WUPF6**：PI11 唤醒引脚标志 (Wakeup Pin flag for PI11)
该位通过硬件置 1，并且只能通过复位引脚或通过将 PWR 电源控制寄存器 2 (PWR_CR2) 中的 CWUPF6 位置 1 清零。
0：未发生唤醒事件
1：在 WKUP PI11 上检测到唤醒事件
注：　如果在 WKUP 引脚 PI11 已为高电平时使能 WKUP 引脚（将 EWUP6 位置 1），系统将另外检测到一个唤醒事件。

位 4 **WUPF5**：PI8 唤醒引脚标志 (Wakeup Pin flag for PI8)
该位通过硬件置 1，并且只能通过复位引脚或通过将 PWR 电源控制寄存器 2 (PWR_CR2) 中的 CWUPF5 位置 1 清零。
0：未发生唤醒事件
1：在 WKUP PI8 上检测到唤醒事件
注：　如果在 WKUP 引脚 PI8 已为高电平时使能 WKUP 引脚（将 EWUP5 位置 1），系统将另外检测到一个唤醒事件。

位 3 **WUPF4**：PC13 唤醒引脚标志 (Wakeup Pin flag for PC13)
该位通过硬件置 1，并且只能通过复位引脚或通过将 PWR 电源控制寄存器 2 (PWR_CR2) 中的 CWUPF4 位置 1 清零。
0：未发生唤醒事件
1：在 WKUP PC13 上检测到唤醒事件
注：　如果在 WKUP 引脚 PC13 已为高电平时使能 WKUP 引脚（将 EWUP4 位置 1），系统将另外检测到一个唤醒事件。

位 2 **WUPF3**：PC1 唤醒引脚标志 (Wakeup Pin flag for PC1)
该位通过硬件置 1，并且只能通过复位引脚或通过将 PWR 电源控制寄存器 2 (PWR_CR2) 中的 CWUPF3 位置 1 清零。
0：未发生唤醒事件
1：在 WKUP PC1 上检测到唤醒事件
注：　如果在 WKUP 引脚 PC1 已为高电平时使能 WKUP 引脚（将 EWUP3 位置 1），系统将另外检测到一个唤醒事件。

位 1 **WUPF2**：PA2 唤醒引脚标志 (Wakeup Pin flag for PA2)

该位通过硬件置 1，并且只能通过复位引脚或通过将 PWR 电源控制寄存器 2 (PWR_CR2) 中的 CWUPF2 位置 1 清零。

0：未发生唤醒事件

1：在 WKUP PA2 上检测到唤醒事件

注： 如果在 WKUP 引脚 PA2 已为高电平时使能 WKUP 引脚（将 EWUP2 位置 1），系统将另外检测到一个唤醒事件。

位 0 **WUPF1**：PA0 唤醒引脚标志 (Wakeup Pin flag for PA0)

该位通过硬件置 1，并且只能通过复位引脚或通过将 PWR 电源控制寄存器 2 (PWR_CR2) 中的 CWUPF1 位置 1 清零。

0：未发生唤醒事件

1：在 WKUP PA0 上检测到唤醒事件

注： 如果在 WKUP 引脚 PA0 已为高电平时使能 WKUP 引脚（将 EWUP1 位置 1），系统将另外检测到一个唤醒事件。

4.5 PWR 寄存器映射

下表对 PWR 寄存器进行了汇总。

表 19. 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_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDEN[1:0]		ODSWEN	ODEN	VOS[1:0]		ADCDC1	Res.	MRUDS	LPUDS	FPDS	DBP	PLS[2:0]			PVDE	CSBF	Res.	PDDS	LPDS
	Reset value													1	1	1	1	1	1	0		1	1	0	0	0	0	0	0	0		0	0
0x004	PWR_CSR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDRDY[1:0]		ODSWRDY	ODRDY	Res.	VOSRDY	Res.	Res.	Res.	Res.	BRE	Res.	Res.	Res.	Res.	Res.	BRR	PVDO	SBF	WUIF
	Reset value													0	0	0	0		0					0						0	0	0	0
0x008	PWR_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUPP6	WUPP5	WUPP4	WUPP3	WUPP2	WUPP1	Res.	Res.	CWUPF6	CWUPF5	CWUPF4	CWUPF3	CWUPF2	CWUPF1
	Reset value																			0	0	0	0	0	0			0	0	0	0	0	0
0x00C	PWR_CSR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWUP6	EWUP5	EWUP4	EWUP3	EWUP2	EWUP1	Res.	Res.	WUPF6	WUPF5	WUPF4	WUPF3	WUPF2	WUPF1
	Reset value																			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 64 页上的第 2.2.2 节*。

5 复位和时钟控制 (RCC)

5.1 复位

共有三种类型的复位，分别为系统复位、电源复位和备份域复位。

5.1.1 系统复位

除了时钟控制寄存器 CSR 中的复位标志和备份域中的寄存器外，系统复位会将其它全部寄存器都复位为复位值（请参见*图 11*）。

只要发生以下事件之一，就会产生系统复位：

1. NRST 引脚低电平（外部复位）
2. 窗口看门狗计数结束（WWDG 复位）
3. 独立看门狗计数结束（IWDG 复位）
4. 软件复位（SW 复位）（请参见*软件复位*）
5. 低功耗管理复位（请参见*低功耗管理复位*）

软件复位

可通过查看 *RCC 时钟控制和状态寄存器 (RCC_CSR)* 中的复位标志确定。

要对器件进行软件复位，必须将 Cortex®-M7 应用中断和复位控制寄存器中的 SYSRESETREQ 位置 1。有关详细信息，请参见 Cortex®-M7 技术参考手册。

低功耗管理复位

引发低功耗管理复位的方式有两种：

1. 进入待机模式时产生复位：

   此复位的使能方式是清零用户选项字节中的 nRST_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。
2. 进入停止模式时产生复位：

   此复位的使能方式是清零用户选项字节中的 nRST_STOP 位。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。

5.1.2 电源复位

只要发生以下事件之一，就会产生电源复位：

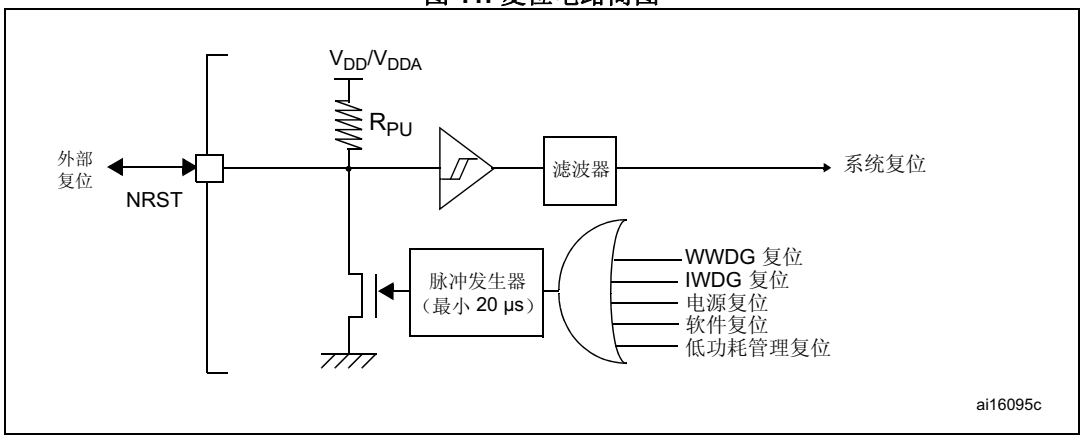
1. 上电/掉电复位（POR/PDR 复位）或欠压 (BOR) 复位。
2. 在退出待机模式时。

除备份域内的寄存器以外，电源复位会将其它全部寄存器设置为复位值（请参见*图 11*）。

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。RESET 复位入口向量在存储器映射中固定在地址 0x0000_0004。

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。

图 11. 复位电路简图



备份域具有两个特定的复位，这两个复位仅作用于备份域本身（请参见*图 11*）。

5.1.3 备份域复位

备份域复位会将所有 RTC 寄存器和 RCC_BDCR 寄存器复位为各自的复位值。BKPSRAM 不受此复位影响。BKPSRAM 的唯一复位方式是通过 Flash 接口将 Flash 保护等级从 1 切换到 0。

只要发生以下事件之一，就会产生备份域复位：

1. 软件复位，通过将 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 BDRST 位置 1 触发。
2. 在电源 V_{DD} 和 V_{BAT} 都已掉电后，其中任何一个又再上电。

5.2 时钟

可以使用三种不同的时钟源来驱动系统时钟 (SYSCLK)：

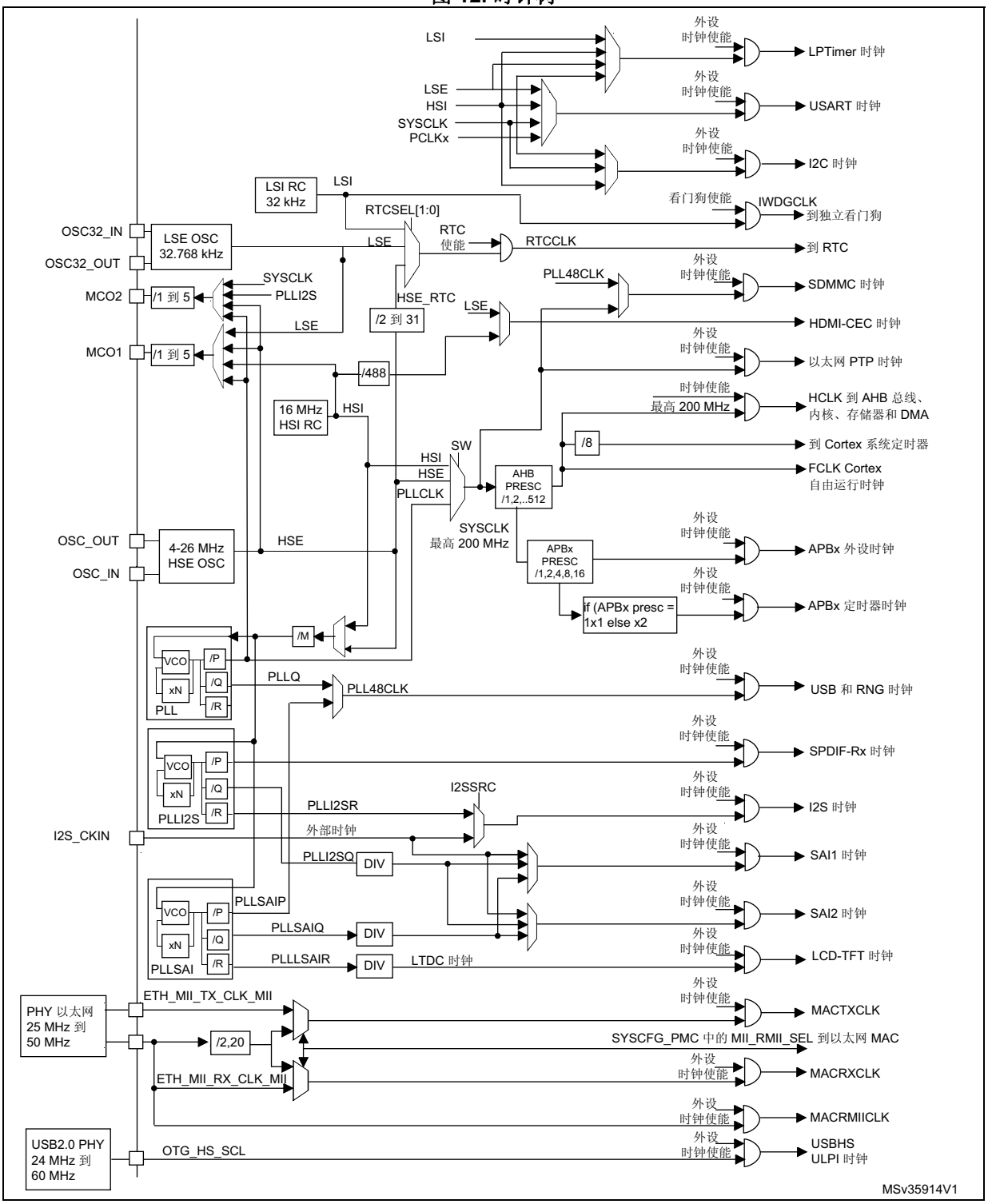
- HSI 振荡器时钟
- HSE 振荡器时钟
- 主 PLL (PLL) 时钟

器件具有以下两个次级时钟源：

- 32 kHz 低速内部 RC (LSI RC)，该 RC 用于驱动独立看门狗，也可选择提供给 RTC 用于停机/待机模式下的自动唤醒
- 32.768 kHz 低速外部晶振（LSE 晶振），用于驱动 RTC 时钟 (RTCCLK)

对于每个时钟源来说，在未使用时都可单独打开或者关闭，以降低功耗。

图 12. 时钟树



1. 有关内部和外部时钟源特性的所有详细信息，请参见器件数据手册的电气特性部分。
2. 当 RCC_DKCFGR1 寄存器的 TIMPRE 位复位时，如果 APBx 预分频器为 1，则 TIMxCLK = PCLKx，否则 TIMxCLK = 2x PCLKx。
3. 当 RCC_DKCFGR1 寄存器的 TIMPRE 位置 1 后，如果 APBx 预分频器为 1、2 或 4，则 TIMxCLK = HCLK，否则 TIMxCLK = 4x PCLKx。

时钟控制器为应用带来了高度的灵活性，用户可灵活选择外部晶振或振荡器，从而以最高频率运行内核和外设。

所有外设时钟均由其总线时钟 (HCLK、PLCK1、PCLK2) 提供，但以下时钟除外：

- 48 MHz 时钟，用于 USB OTG FS、SDMMC 和 RNG。该时钟由以下任一时钟源提供：
  - 主 PLL VCO（PLLQ 时钟）
  - PLLSAI VCO（PLLSAI 时钟）
- 由以下任一时钟源提供的 U(S)ART 时钟：
  - 系统时钟 (SYSCLK)
  - HSI 时钟
  - LSE 时钟
  - APB1 或 APB2 时钟（是 PCLK1 还是 PCLK2 取决于 U(S)ART 映射哪个 APB）
- 由以下任一时钟源提供的 I2C 时钟：
  - 系统时钟 (SYSCLK)
  - HSI 时钟
  - APB1 时钟 (PCLK1)
- I2S 时钟

  要实现高品质的音频性能，可通过特定的 PLL (PLLI2S) 或映射到 I2S_CKIN 引脚的外部时钟提供 I2S 时钟。有关 I2S 时钟频率和精度的详细信息，请参见*第 32.7.4 节：时钟发生器*。
- 由以下任一时钟源提供的 SAI1 和 SAI2 时钟：
  - PLLSAI VCO (PLLSAIQ)
  - PLLI2S VCO (PLLI2SQ)
  - 映射到 I2S_CKIN 引脚的外部时钟
- LTDC 时钟

  LTDC 时钟由特定的 PLL (PLLSAI) 生成。
- 由以下任一时钟源提供的低功耗定时器 (LPTIM1) 时钟：
  - LSI 时钟
  - LSE 时钟
  - HSI 时钟
  - APB1 时钟 (PCLK1)
  - 映射到 LPTIM1_IN1 上的外部时钟
- 由外部 PHY 提供的 USB OTG HS (60 MHz) 时钟。
- 由外部 PHY 提供的以太网 MAC 时钟（TX、RX 和 RMII）。有关以太网配置的详细信息，请参见以太网外设说明中的*第 38.4.4 节：MII/RMII 选择*。当使用以太网时，AHB 时钟频率至少应为 25 MHz。
- SPDIFRX 时钟：由 PLLI2SP VCO 产生。
- 由以下任一时钟源提供的 HDMI-CEC 时钟：
  - LSE 时钟
  - HSI 时钟 488 分频
- 由以下任一时钟源提供的 RTC 时钟：
  - LSE 时钟
  - LSI 时钟
  - HSE 时钟 32 分频

- 始终为 LSI 时钟的 IWDG 时钟。
- 定时器时钟频率由硬件自动设置。根据 RCC_CFGR 寄存器中 TIMPRE 位的取值，共分为两种情况：
  - 如果 RCC_DKCFGR1 寄存器的 TIMPRE 位清 0：

    如果 APB 预分频器分频系数是 1，则定时器时钟频率 (TIMxCLK) 为 PCLKx。否则，定时器时钟频率将为与定时器相连的 APB 域的频率的两倍：TIMxCLK = 2x PCLKx。
  - 如果 RCC_DKCFGR1 寄存器的 TIMPRE 位置 1：

    如果将 APB 预分频器的分频系数配置为1、2 或 4，则定时器时钟频率 (TIMxCLK) 将设置为 HCLK。否则，定时器时钟频率将为与定时器相连的 APB 域的频率的四倍：TIMxCLK = 4xPCLKx。

RCC 向 Cortex 系统定时器 (SysTick) 馈送 8 分频的 AHB 时钟 (HCLK)。SysTick 可使用此时钟作为时钟源，也可使用 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®-M7 的自由运行时钟。有关详细信息，请参见 Cortex®-M7 技术参考手册。

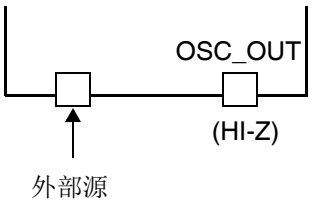
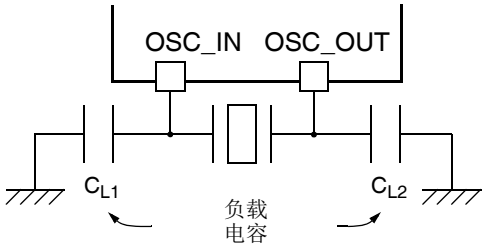
5.2.1 HSE 时钟

高速外部时钟信号 (HSE) 有 2 个时钟源：

- HSE 外部晶振/陶瓷谐振器
- HSE 外部用户时钟

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振荡器的不同做适当调整。

图 13. HSE/LSE 时钟源

	硬件配置
外部时钟	
晶振/陶瓷谐振器	

外部源（HSE 旁路）

在此模式下，必须提供外部时钟源。通过将 *RCC 时钟控制寄存器 (RCC_CR)* 的 HSEBYP 和 HSEON 位置 1 可选择此模式。必须使用占空比约为 50% 的外部时钟信号（方波、正弦波或三角波）来驱动 OSC_IN 引脚，同时 OSC_OUT 引脚应保持为 HI-Z。请参见*图 13*。

外部晶振/陶瓷谐振器（HSE 晶振）

HSE 的特点是精度非常高。

相关的硬件配置如*图 13*所示。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) 中的 HSERDY 标志指示高速外部振荡器是否稳定。在启动时，硬件将此位置 1 后，此时钟才可以使用。如在 *RCC 时钟中断寄存器 (RCC_CIR)* 中使能中断，则可产生中断。

HSE 晶振可通过 *RCC 时钟控制寄存器 (RCC_CR)* 中的 HSEON 位打开或关闭。

5.2.2 HSI 时钟

HSI 时钟信号由内部 16 MHz RC 振荡器生成，可直接用作系统时钟，或者用作 PLL 输入。

HSI RC 振荡器的优点是成本较低（无需使用外部组件）。此外，其启动速度也要比 HSE 晶振块，但即使校准后，其精度也不及外部晶振或陶瓷谐振器。

校准

由于生产工艺不同，不同芯片的 RC 振荡器频率也存在差异，因此 ST 会对每个器件进行出厂校准，使器件在 T_A = 25 °C 时达到 1% 的精度。

复位后，工厂校准值将加载到 *RCC 时钟控制寄存器 (RCC_CR)* 的 HSICAL[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 振荡器的速度。用户可通过 *RCC 时钟控制寄存器 (RCC_CR)* 中的 HSITRIM[4:0] 位对 HSI 频率进行微调。

RCC 时钟控制寄存器 (RCC_CR) 中的 HSIRDY 标志指示 HSI RC 是否稳定。在启动时，硬件将此位置 1 后，HSI 才可以使用。

HSI RC 可通过 *RCC 时钟控制寄存器 (RCC_CR)* 中的 HSION 位打开或关闭。

HSI 信号还可作为备份时钟源（辅助时钟）使用，以防 HSE 晶振发生故障。请参见*第 127 页上的第 5.2.7 节：时钟安全系统 (CSS)*。

5.2.3 PLL

该器件具有三个 PLL：

- 主 PLL (PLL) 由 HSE 或 HSI 振荡器提供时钟信号，并具有两个不同的输出时钟：
  - 第一个输出用于生成高速系统时钟（最高达 216 MHz）
  - 第二个输出用于为 USB OTG FS、SDMMC 和 RNG 生成 48 MHz 时钟。
- PLLI2S 用于生成精确时钟，从而在 I2S、SAI 和 SPDIFRX 接口实现高品质音频性能。
- PLLSAI 用于为 SAI 接口生成时钟，生成 LCD-TFT 时钟以及可供 USB OTG FS、SDMMC 和 RNG 选择的 48 MHz (PLLSAI48CLK) 时钟。

由于在 PLL 使能后主 PLL 配置参数便不可更改，所以建议先对 PLL 进行配置，然后再使能（选择 HSI 或 HSE 振荡器作为 PLL 时钟源，并配置分频系数 M、N、P 和 Q）。

PLLI2S 和 PLLSAI 与 PLL 使用相同的输入时钟（PLLM[5:0] 和 PLLSRC 位为两个 PLL 所共用）。但是，PLLI2S 和 PLLSAI 具有专门的使能/禁止和分频系数（N 和 R）配置位。在 PLLI2S 和 PLLSAI 使能后，配置参数便不能更改。

当进入停机和待机模式后，三个 PLL 将由硬件关闭；如将 HSE 或 PLL（由 HSE 提供时钟信号）用作系统时钟，则在 HSE 发生故障时，三个 PLL 也将由硬件关闭。*RCC PLL 配置寄存器 (RCC_PLLCFGR)*、*RCC 时钟配置寄存器 (RCC_CFGR)* 和 *RCC 专用时钟配置寄存器 (RCC_DKCFGR1)* 可用于配置 PLL、PLLI2S 和 PLLSAI。

5.2.4 LSE 时钟

LSE 时钟由 32.768 kHz 低速外部晶振或陶瓷谐振器产生。可作为实时时钟外设 (RTC) 的时钟源来提供时钟/日历或其它定时功能，具有功耗低且精度高的优点。

LSE 振荡器通过 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 LSEON 位打开和关闭。

RCC 备份域控制寄存器 (RCC_BDCR) 中的 LSERDY 标志指示 LSE 晶振是否稳定。在启动时，硬件将此位置 1 后，LSE 晶振输出时钟信号才可以使用。如在 *RCC 时钟中断寄存器 (RCC_CIR)* 中使能中断，则可产生中断。

外部源（LSE 旁路）

在此模式下，必须提供外部时钟源。最高频率不超过 1 MHz。此模式通过将 *RCC 备份域控制寄存器 (RCC_BDCR)* 中的 LSEBYP 和 LSEON 位置 1 进行选择。必须使用占空比约为 50% 的外部时钟信号（方波、正弦波或三角波）来驱动 OSC32_IN 引脚，同时 OSC32_OUT 引脚应保持为 HI-Z。请参见*图 13*。

5.2.5 LSI 时钟

LSI RC 可作为低功耗时钟源在停机和待机模式下保持运行，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 RC 可通过 *RCC 时钟控制和状态寄存器 (RCC_CSR)* 中的 LSION 位打开或关闭。

RCC 时钟控制和状态寄存器 (RCC_CSR) 中的 LSIRDY 标志指示低速内部振荡器是否稳定。在启动时，硬件将此位置 1 后，此时钟才可以使用。如在 *RCC 时钟中断寄存器 (RCC_CIR)* 中使能中断，则可产生中断。

5.2.6 系统时钟 (SYSCLK) 选择

在系统复位后，默认系统时钟为 HSI。在直接使用 HSI 或者通过 PLL 使用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)* 中的状态位指示哪个（些）时钟已就绪，以及当前哪个时钟正充当系统时钟。

5.2.7 时钟安全系统 (CSS)

时钟安全系统可通过软件激活。激活后，时钟监测器将在 HSE 振荡器启动延迟后使能，并在此振荡器停止时被关闭。

如果 HSE 时钟发生故障，此振荡器将自动禁止，一个时钟故障事件将发送到高级控制定时器 TIM1 和 TIM8 的断路输入，并且同时还将生成一个中断来向软件通知此故障（时钟安全系统中断，CSSI），以使 MCU 能够执行救援操作。CSSI 与 Cortex®-M7 NMI（不可屏蔽中断）异常向量相链接。

注： *当 CSS 使能后，如果 HSE 时钟偶发故障，则 CSS 将生成一个中断，进而促使 NMI 自动生成。NMI 将无限期执行，除非将 CSS 中断挂起位清零。因此，应用程序必须在 NMI ISR 中将 CSS 中断清零，具体方式为在时钟中断寄存器 (RCC_CIR) 中将 CSSC 位置 1。*

如果直接或间接使用 HSE 振荡器作为系统时钟（间接是指该振荡器直接用作 PLL 的输入时钟，并且该 PLL 时钟为系统时钟）并且检出故障，则系统时钟将切换到 HSI 振荡器并且 HSE 振荡器将被禁止。

如果 HSE 振荡器时钟是充当系统时钟的 PPL 的时钟源，则在发生故障时，PLL 也会被禁止。在此情况下，如果 PLLI2S 或 PLLSAI 已使能，则在 HSE 发生故障时也会将其禁止。

5.2.8 RTC/AWU 时钟

一旦选定 RTCCLK 时钟源后，要想修改所做选择，只能复位电源域。

RTCCLK 时钟源可以是 HSE 1 MHz（HSE 由一个可编程的预分频器分频）、LSE 或者 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] 位和 *RCC 时钟配置寄存器 (RCC_CFGR)* 中的 RTCPRE[4:0] 位。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，则系统电源丢失时 RTC 仍将正常工作。如果选择 LSI 作为 AWU 时钟，则在系统电源丢失时将无法保证 AWU 的状态。如果 HSE 振荡器通过一个介于 2 和 31 之间的值进行分频，则在备用或系统电源丢失时将无法保证 RTC 的状态。

LSE 时钟位于备份域中，而 HSE 和 LSI 时钟则不是。因此：

- 如果选择 LSE 作为 RTC 时钟：
  - 只要 V_{BAT} 电源保持工作，即使 V_{DD} 电源关闭，RTC 仍可继续工作。
- 如果选择 LSI 作为自动唤醒单元 (AWU) 时钟：
  - 在 V_{DD} 电源掉电时，AWU 的状态将不能保证。有关 LSI 校准的详细信息，请参见 *第 127 页上的第 5.2.5 节：LSI 时钟*。
- 如果使用 HSE 时钟作为 RTC 时钟：
  - 如果 V_{DD} 电源掉电、内部调压器关闭（切断 1.2 V 域的供电）或进入停机模式时，RTC 的状态将不能保证。

注： *要在 APB1 时钟频率低于 RTC 时钟频率的七倍 ($f_{APB1} < 7xf_{RTCLCK}$) 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取访问。*

5.2.9 看门狗时钟

如果独立看门狗 (IWDG) 已通过硬件选项字节或软件设置的方式启动，则 LSI 振荡器将强制打开且不可禁止。在 LSI 振荡器稳定后，时钟将提供给 IWDG。

5.2.10 时钟输出功能

共有两个微控制器时钟输出 (MCO) 引脚：

- MCO1

  用户可通过可配置的预分配器（从 1 到 5）向 MCO1 引脚 (PA8) 输出四个不同的时钟源：
  - HSI 时钟

  - LSE 时钟
  - HSE 时钟
  - PLL 时钟

  所需的时钟源通过 *RCC 时钟配置寄存器 (RCC_CFGR)* 中的 MCO1PRE[2:0] 和 MCO1[1:0] 位选择。

- MCO2

  用户可通过可配置的预分配器（从 1 到 5）向 MCO2 引脚 (PC9) 输出四个不同的时钟源：
  - HSE 时钟
  - PLL 时钟
  - 系统时钟 (SYSCLK)
  - PLLI2S 时钟

  所需的时钟源通过 *RCC 时钟配置寄存器 (RCC_CFGR)* 中的 MCO2PRE[2:0] 和 MCO2 位选择。

对于不同的 MCO 引脚，必须将相应的 GPIO 端口在复用功能模式下进行设置。

5.2.11 基于 TIM5/TIM11 的内部/外部时钟测量

所有时钟源的频率都可通过对 TIM5 channel4 和 TIM11 channel1 的输入捕获进行间接测量，如*图 14* 和*图 15* 所示。

基于 TIM5 channel4 的内部/外部时钟测量

TIM5 具有一个输入复用器，可选择输入捕获是由 I/O 触发还是由内部时钟触发。此选择通过 TIM5_OR 寄存器的 TI4_RMP [1:0] 位执行。

将 LSE 连接到 channel4 输入捕获的主要目的是为了精确测量 HSI（这需要将 HSI 用作系统时钟源）。借助 LSE 信号连续边沿之间的 HSI 时钟计数数量，即可对内部时钟周期进行测量。利用 LSE 的高精度（通常为几十 ppm），用户能以同一分辨率测定时钟频率，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。

HSI 振荡器设有针对此目的的专用校准位，且支持用户访问。

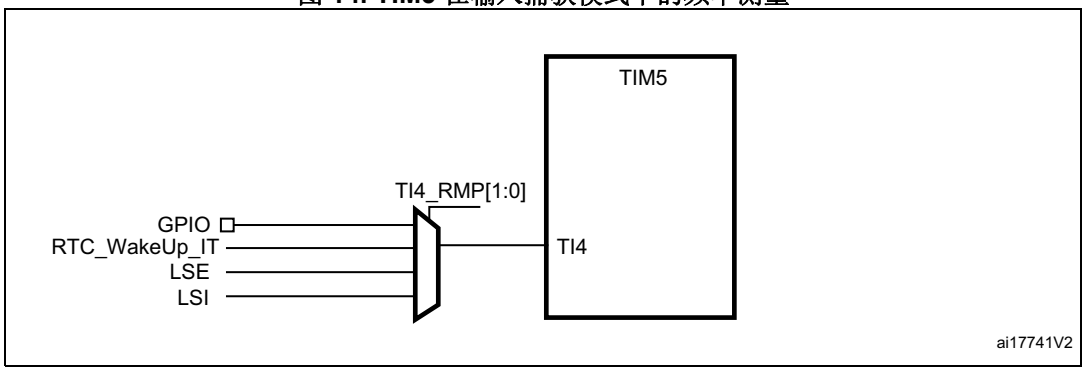
其基本原理是基于相对的测量（例如，HSI/LSE 比）：因此，精度与两个时钟源之比紧密相关。比率越大，测量效果越好。

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：通过测量该振荡器与 HSI 时钟源的比率，可以借助 HSI 精度测定其频率。通过此测量值可实现更加精确的 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。

LSI 频率通过以下步骤测量：

1. 使能 TIM5 定时器并将 channel4 配置为输入捕获模式。
2. 该位将 TIM5_OR 寄存器中的 TI4_RMP 位设置为 0x01，以在内部将 LSI 时钟连接到 TIM5 channel4 输入捕获来实现校准。
3. 通过 TIM5 捕获/比较 4 事件或中断测量 LSI 时钟频率。
4. 使用测得的 LSI 频率来按照所需的时基更新 RTC 的预分频器，并且/或者用其来计算 IWDG 超时。

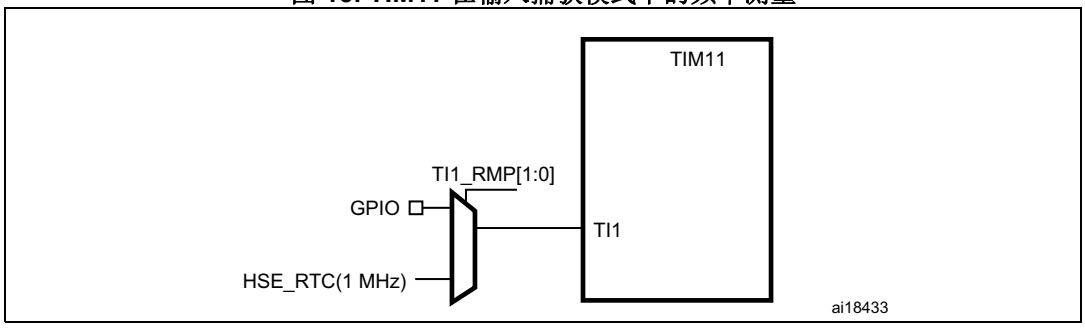
图 14. TIM5 在输入捕获模式下的频率测量



基于 TIM11 channel1 的内部/外部时钟测量

TIM11 具有一个输入复用器，可选择输入捕获是由 I/O 触发还是由内部时钟触发。此选择通过 TIM11_OR 寄存器的 TI1_RMP [1:0] 位执行。HSE_RTC 时钟（由一个可编程预分频器分频的 HSE）连接到通道 1 输入捕获，以粗略指示外部晶振频率。这要求 HSI 为系统时钟源。此功能非常实用，例如可借此测定谐波频率或分谐波频率（–50/+100% 偏差），进而确保符合 IEC 60730/IEC 61335 标准。

图 15. TIM11 在输入捕获模式下的频率测量



5.2.12 外设时钟使能寄存器（RCC_AHBxENR、 RCC_APBxENRy）

每个外设时钟均可由 RCC_AHBxENR 或 RCC_APBxENRy 寄存器的 xxxxEN 位使能。

当外设时钟未激活时，不支持外设寄存器进行读写访问。外设使能位的同步机制可为外设产生无干扰时钟。

使能位置 1 后，在时钟激活前存在时长为 2 个外设时钟周期的延迟。

注意： 使能外设时钟后，软件必须等待 2 个外设时钟周期的延迟才能访问外设寄存器。

5.3 RCC 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.1 节：寄存器相关缩写词列表*。

5.3.1 RCC 时钟控制寄存器 (RCC_CR)

RCC clock control register

偏移地址：0x00

复位值：0x0000 XX83，其中 X 未定义。

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.	PLLSAIRDY	PLLSAION	PLLI2SRDY	PLLI2SON	PLLRDY	PLLON	Res.	Res.	Res.	Res.	CSSON	HSEBYP	HSERDY	HSEON
		r	rw	r	rw	r	rw					rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSICAL[7:0]								HSITRIM[4:0]					Res.	HSIRDY	HSION
r	r	r	r	r	r	r	r	rw	rw	rw	rw	rw		r	rw

位 31:28 保留，必须保持复位值。

位 29 **PLLSAIRDY**：PLLSAI 时钟就绪标志 (PLLSAI clock ready flag)
由硬件置 1，用以指示 PLLSAI 已锁定。
0：PLLSAI 未锁定
1：PLLSAI 已锁定

位 28 **PLLSAION**：PLLSAI 使能 (PLLSAI enable)
由软件置 1 和清零，用于使能 PLLSAI。
当进入停机或待机模式时由硬件清零。
0：PLLSAI 关闭
1：PLLSAI 开启

位 27 **PLLI2SRDY**：PLLI2S 时钟就绪标志 (PLLI2S clock ready flag)
由硬件置 1，用于指示 PLLI2S 已锁定。
0：PLLI2S 未锁定
1：PLLI2S 已锁定

位 26 **PLLI2SON**：PLLI2S 使能 (PLLI2S enable)
由软件置 1 和清零，用于使能 PLLI2S。
当进入停机或待机模式时由硬件清零。
0：PLLI2S 关闭
1：PLLI2S 开启

位 25 **PLLRDY**：主 PLL (PLL) 时钟就绪标志 (Main PLL (PLL) clock ready flag)
由硬件置 1，用以指示 PLL 已锁定。
0：PLL 未锁定
1：PLL 已锁定

位 24 **PLLON**：主 PLL (PLL) 使能 (Main PLL (PLL) enable)
由软件置 1 和清零，用于使能 PLL。
当进入停机或待机模式时由硬件清零。如果 PLL 时钟用作系统时钟，则此位不可清零。
0：PLL 关闭
1：PLL 开启

位 23:20 保留，必须保持复位值。

位 19 **CSSON**：时钟安全系统使能 (Clock security system enable)

由软件置 1 和清零，用于使能时钟安全系统。当 CSSON 置 1 时，时钟监测器将在 HSE 振荡器就绪时由硬件使能，并在检出振荡器故障时由硬件禁止。

0：时钟安全系统关闭（时钟监测器关闭）

1：时钟安全系统开启（如果 HSE 振荡器稳定，则时钟监测器打开；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

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。在将 HSEON 位清零后，HSERDY 将在 6 个 HSE 振荡器时钟周期后转为低电平。

0：HSE 振荡器未就绪

1：HSE 振荡器已就绪

位 16 **HSEON**：HSE 时钟使能 (HSE clock enable)

由软件置 1 和清零。

由硬件清零，用于在进入停机或待机模式时停止 HSE 振荡器。如果 HSE 振荡器直接或间接用于作为系统时钟，则此位不可复位。

0：HSE 振荡器关闭

1：HSE 振荡器开启

位 15:8 **HSICAL[7:0]**：内部高速时钟校准 (Internal high-speed clock calibration)

这些位在启动时自动初始化。

位 7:3 **HSITRIM[4:0]**：内部高速时钟微调 (Internal high-speed clock trimming)

通过这些位，可在 HSICAL[7:0] 位基础上实现可由用户编程的微调值。可通过编程使其适应电压和温度的差异，使内部 HSI RC 的频率更为准确。

位 2 保留，必须保持复位值。

位 1 **HSIRDY**：内部高速时钟就绪标志 (Internal high-speed clock ready flag)

由硬件置 1，用以指示 HSI 振荡器已稳定。在将 HSION 位清零后，HSIRDY 将在 6 个 HSI 时钟周期后转为低电平。

0：HSI 振荡器未就绪

1：HSI 振荡器已就绪

位 0 **HSION**：内部高速时钟使能 (Internal high-speed clock enable)

由软件置 1 和清零。

由硬件置 1，用于在脱离停机或待机模式时或者在直接或间接用作系统时钟的 HSE 振荡器发生故障时强制 HSI 振荡器打开。如果 HSI 直接或间接用于作为系统时钟，则此位不可清零。

0：HSI 振荡器关闭

1：HSI 振荡器开启

5.3.2 RCC PLL 配置寄存器 (RCC_PLLCFGR)

RCC PLL configuration register

偏移地址：0x04

复位值：0x2400 3010

访问：无等待周期，按字、半字和字节访问。

此寄存器用于根据公式配置 PLL 时钟输出：

- $f_{(VCO\ 时钟)} = f_{(PLL\ 时钟输入)} \times (PLLN / PLLM)$
- $f_{(PLL\ 常规时钟输出)} = f_{(VCO\ 时钟)} / PLLP$
- $f_{(USB\ OTG\ FS、SDMMC、RNG\ 时钟输出)} = f_{(VCO\ 时钟)} / PLLQ$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	PLLQ[3:0]				Res.	PLLSRC	Res.	Res.	Res.	Res.	PLLP[1:0]	
				rw	rw	rw	rw		rw					rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PLLN[8:0]									PLLM[5:0]					
	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 27:24 **PLLQ[3:0]**：主 PLL (PLL) 分频系数，适用于 USB OTG FS、SDIO 和随机数发生器时钟 (Main PLL (PLL) division factor for USB OTG FS, SDIO and random number generator clocks)

由软件置 1 或清零，用于控制 USB OTG FS 时钟、随机数发生器时钟和 SDMMC 时钟的频率。仅在 PLL 已禁止时才能写入这些位。

注意： 为使 USB OTG FS 能够正常工作，需要 48 MHz 的时钟。对于 SDMMC 和随即数生成器，频率需要低于或等于 48 MHz 才可正常工作。

USB OTG FS 时钟频率 = VCO 频率 / PLLQ，其中，2 ≤ PLLQ ≤ 15

0000：PLLQ = 0，错误配置

0001：PLLQ = 1，错误配置

0010：PLLQ = 2

0011：PLLQ = 3

0100：PLLQ = 4

...

1111：PLLQ = 15

位 23 保留，必须保持复位值。

位 22 **PLLSRC**：主 PLL (PLL) 和音频 PLL (PLLI2S) 输入时钟源 (Main PLL(PLL) and audio PLL (PLLI2S) entry clock source)

由软件置 1 和清零，用于选择 PLL 和 PLLI2S 时钟源。此位只有在 PLL 和 PLLI2S 已禁止时才可写入。

0：选择 HSI 时钟作为 PLL 和 PLLI2S 时钟输入

1：选择 HSE 振荡器时钟作为 PLL 和 PLLI2S 时钟输入

位 21:18 保留，必须保持复位值。

位 17:16 **PLLP[1:0]**：适用于主系统时钟的主 PLL (PLL) 分频系数 (Main PLL (PLL) division factor for main system clock)

由软件置 1 和清零，用于控制常规 PLL 输出时钟的频率。这些位只能在 PLL 已禁止时写入。

注意： 软件必须正确设置这些位，使其在此域中不超过 180 MHz。

PLL 输出时钟频率 = VCO 频率 / PLLP，其中，PLLP = 2、4、6 或 8

00：PLLP = 2

01：PLLP = 4

10：PLLP = 6

11：PLLP = 8

位 14:6 **PLLN[8:0]**：适用于 VCO 的主 PLL (PLL) 倍频系数 (Main PLL (PLL) division factor for main system clock)

由软件置 1 和清零，用于控制 VCO 的倍频系数。这些位只能在 PLL 已禁止时写入。写入这些位时只允许使用半字和字访问。

注意： 软件必须正确设置这些位，确保 VCO 输出频率介于 192 和 432 MHz 之间。以下是一个 PLL 输入的 PLLN 位域禁用值 F_{PLL_IN} = 1 MHz 的示例：

VCO 输出频率 = VCO 输入频率 × PLLN，其中，192 ≤ PLLN ≤ 432

000000000：PLLN = 0，错误配置

000000001：PLLN = 1，错误配置

...

011000000：PLLN = 192

...

110110000：PLLN = 432

110110001：PLLN = 433，错误配置

...

111111111：PLLN = 511，错误配置

位 5:0 **PLLM[5:0]**：适用于主 PLL（PLL、PLLI2S 和 PLLSAI）输入时钟的分频系数 (Division factor for the main PLLs (PLL, PLLI2S and PLLSAI) input clock)

由软件置 1 和清零，用于在 VCO 之前对 PLL 和 PLLI2S 输入时钟进行分频。这些位只有在 PLL 和 PLLI2S 已禁止时才可写入。

注意： 软件必须正确设置这些位，确保 VCO 输入频率介于 1 和 2 MHz 之间。建议选择 2 MHz 的频率，以便限制 PLL 抖动。

VCO 输入频率 = PLL 输入时钟频率 / PLLM，其中，2 ≤ PLLM ≤ 63

000000：PLLM = 0，错误配置

000001：PLLM = 1，错误配置

000010：PLLM = 2

000011：PLLM = 3

000100：PLLM = 4

...

111110：PLLM = 62

111111：PLLM = 63

5.3.3 RCC 时钟配置寄存器 (RCC_CFGR)

RCC clock configuration register

偏移地址：0x08

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
MCO2		MCO2 PRE[2:0]			MCO1 PRE[2:0]			I2SSCR	MCO1		RTCPRE[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
PPRE2[2:0]			PPRE1[2:0]			Res.	Res.	HPRE[3:0]				SWS1	SWS0	SW1	SW0
rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	r	r	rw	rw

位 31:30 **MCO2[1:0]**：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。

00：选择系统时钟 (SYSCLK)
01：选择 PLLI2S 时钟
10：选择 HSE 振荡器时钟
11：选择 PLL 时钟

位 27:29 **MCO2PRE**：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。

0xx：无分频
100：2 分频
101：3 分频
110：4 分频
111：5 分频

位 24:26 **MCO1PRE**：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。

0xx：无分频
100：2 分频
101：3 分频
110：4 分频
111：5 分频

位 23 **I2SSRC**：I2S 时钟选择 (I2S clock selection)

由软件置 1 和清零。通过此位可在 PLLI2S 时钟和外部时钟之间选择 I2S 时钟源。强烈建议仅在复位之后使能 I2S 模块之前对此位进行更改。

0：PLLI2S 时钟用作 I2S 时钟源
1：映射到 I2S_CKIN 引脚的外部时钟用作 I2S 时钟源

位 22:21 **MCO1**：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且在使能外部振荡器和 PLL 之前来配置这些位。

00：选择 HSI 时钟
01：选择 LSE 振荡器
10：选择 HSE 振荡器时钟
11：选择 PLL 时钟

位 20:16 **RTCPRE**：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 时钟输入时钟进行分频，进而为 RTC 生成 1 MHz 的时钟。

注意： 软件必须正确设置这些位，确保提供给 RTC 的时钟为 1 MHz。在选择 RTC 时钟源之前必须配置这些位。

00000：无时钟
00001：无时钟
00010：HSE/2
00011：HSE/3
00100：HSE/4
...
11110：HSE/30
11111：HSE/31

位 15:13 **PPRE2**：APB 高速预分频器 (APB2) (APB high-speed prescaler (APB2))
由软件 1 和清零，用于控制 APB 高速时钟分频系数。

注意： 软件必须正确设置这些位，使其在此域中不超过 90 MHz。在 PPRE2 写入后，时钟将通过 1 AHB 到 16 AHB 周期新预分频系数进行分频。

0xx：AHB 时钟不分频
100：AHB 时钟 2 分频
101：AHB 时钟 4 分频
110：AHB 时钟 8 分频
111：AHB 时钟 16 分频

位 12:10 **PPRE1**：APB 低速预分频器 (APB1) (APB Low speed prescaler (APB1))
由软件置 1 和清零，用于控制 APB 低速时钟分频系数。

注意： 软件必须正确设置这些位，使其在此域中不超过 45 MHz。在 PPRE1 写入后，时钟将通过 1 AHB 到 16 AHB 周期新预分频系数进行分频。

0xx：AHB 时钟不分频
100：AHB 时钟 2 分频
101：AHB 时钟 4 分频
110：AHB 时钟 8 分频
111：AHB 时钟 16 分频

位 9:8 保留，必须保持复位值。

位 7:4 **HPRE**：AHB 预分频器 (AHB prescaler)
由软件置 1 和清零，用于控制 AHB 时钟分频系数。

注意： 在 HPRE 写入后，时钟将在 1 到 16 个 AHB 周期内通过新的预分频系数进行分频。

注意： 当使用以太网时，AHB 时钟频率必须至少为 25 MHz。

0xxx：系统时钟不分频
1000：系统时钟 2 分频
1001：系统时钟 4 分频
1010：系统时钟 8 分频
1011：系统时钟 16 分频
1100：系统时钟 64 分频
1101：系统时钟 128 分频
1110：系统时钟 256 分频
1111：系统时钟 512 分频

位 3:2 **SWS**：系统时钟切换状态 (System clock switch status)
由硬件置 1 和清零，用于指示用作系统时钟的时钟源。
00：HSI 振荡器用作系统时钟
01：HSE 振荡器用作系统时钟
10：PLL 用作系统时钟
11：不适用

位 1:0 **SW**：系统时钟切换 (System clock switch)
由软件置 1 和清零，用于选择系统时钟源。
由硬件置 1，用于在退出停机或待机模式时或者在直接或间接用作系统时钟的 HSE 振荡器发生故障时强制 HSI 的选择。
00：选择 HSI 振荡器作为系统时钟
01：选择 HSE 振荡器作为系统时钟
10：选择 PLL 作为系统时钟
11：不允许

5.3.4 RCC 时钟中断寄存器 (RCC_CIR)

RCC clock interrupt register

偏移地址：0x0C

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.	CSSC	PLLSAI RDYC	PLLI2S RDYC	PLL RDYC	HSE RDYC	HSI RDYC	LSE RDYC	LSI RDYC
								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.	PLLSAI RDYIE	PLLI2S RDYIE	PLL RDYIE	HSE RDYIE	HSI RDYIE	LSE RDYIE	LSI RDYIE	CSSF	PLLSAI RDYF	PLLI2S RDYF	PLL RDYF	HSE RDYF	HSI RDYF	LSE RDYF	LSI RDYF
	rw	rw	rw	rw	rw	rw	rw	r	r	r	r	r	r	r	r

位 31:24 保留，必须保持复位值。

位 23 **CSSC**：时钟安全系统中断清零 (Clock security system interrupt clear)
此位由软件置 1，用于将 CSSF 标志清零。
0：无影响
1：将 CSSF 标志清零

位 22 **PLLSAIRDYC**：PLLSAI 就绪中断清零 (PLLSAI Ready Interrupt Clear)
此位由软件置 1，用于将 PLLSAIRDYF 标志清零。此位在 PLLSAIRDYF 清零时由硬件复位。
0：未将 PLLSAIRDYF 清零
1：已将 PLLSAIRDYF 清零

位 21 **PLLI2SRDYC**：PLLI2S 就绪中断清零 (PLLI2S ready interrupt clear)
此位由软件置 1，用于将 PLLI2SRDYF 标志清零。
0：无影响
1：已将 PLLI2SRDYF 清零

位 20 **PLLRDYC**：主 PLL(PLL) 就绪中断清零 (Main PLL(PLL) ready interrupt clear)
此位由软件置 1，用于将 PLLRDYF 标志清零。
0：无影响
1：已将 PLLRDYF 清零

位 19 **HSERDYC**：HSE 就绪中断清零 (HSE ready interrupt clear)
此位由软件置 1，用于将 HSERDYF 标志清零。
0：无影响
1：已将 HSERDYF 清零

位 18 **HSIRDYC**：HSI 就绪中断清零 (HSI ready interrupt clear)
此位由软件置 1，用于将 HSIRDYF 标志清零。
0：无影响
1：已将 HSIRDYF 清零

位 17 **LSERDYC**：LSE 就绪中断清零 (LSE ready interrupt clear)
此位由软件置 1，用于将 LSERDYF 标志清零。
0：无影响
1：已将 LSERDYF 清零

位 16 **LSIRDYC**：LSI 就绪中断清零 (LSI ready interrupt clear)
此位由软件置 1，用于将 LSIRDYF 标志清零。
0：无影响
1：已将 LSIRDYF 清零

位 15 保留，必须保持复位值。

位 14 **PLLSAIRDYIE**：PLLSAI 就绪中断使能 (PLLSAI Ready Interrupt Enable)
此位由软件置 1 和复位，用于使能/禁止由 PLLSAI 锁定引起的中断。
0：禁止 PLLSAI 锁定中断
1：使能 PLLSAI 锁定中断

位 13 **PLLI2SRDYIE**：PLLI2S 就绪中断使能 (PLLI2S ready interrupt enable)
此位由软件置 1 和清零，用于使能/禁止由 PLLI2S 锁定引起的中断。
0：禁止 PLLI2S 锁定中断
1：使能 PLLI2S 锁定中断

位 12 **PLLRDYIE**：主 PLL (PLL) 就绪中断使能 (Main PLL (PLL) ready interrupt enable)
此位由软件置 1 和清零，用于使能/禁止由 PLL 锁定引起的中断。
0：禁止 PLL 锁定中断
1：使能 PLL 锁定中断

位 11 **HSERDYIE**：HSE 就绪中断使能 (HSE ready interrupt enable)
此位由软件置 1 和清零，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
1：使能 HSE 就绪中断

位 10 **HSIRDYIE**：HSI 就绪中断使能 (HSI ready interrupt enable)
此位由软件置 1 和清零，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
1：使能 HSI 就绪中断

位 9 **LSERDYIE**：LSE 就绪中断使能 (LSE ready interrupt enable)
此位由软件置 1 和清零，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
1：使能 LSE 就绪中断

位 8 **LSIRDYIE**：LSI 就绪中断使能 (LSI ready interrupt enable)
此位由软件置 1 和清零，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
1：使能 LSI 就绪中断

位 7 **CSSF**：时钟安全系统中断标志 (Clock security system interrupt flag)
当在 HSE 振荡器中检出故障时，此位由硬件置 1。
在 CSSC 位置 1 时，此位由软件清零。
0：当前未因 HSE 时钟故障而引起时钟安全中断
1：因 HSE 时钟故障而引起时钟安全中断

位 6 **PLLSAIRDYF**：PLLSAI 就绪中断标志 (PLLSAI Ready Interrupt flag)
当 PLLSAI 锁定且 PLLSAIRDYDIE 置 1 时，此位由硬件置 1。
在 PLLSAIRDYC 位置 1 时，此位由软件清零。
0：当前未因 PLLSAI 锁定而引起时钟就绪中断
1：因 PLLSAI 锁定而引起时钟就绪中断

位 5 **PLLI2SRDYF**：PLLI2S 就绪中断标志 (PLLI2S ready interrupt flag)
当 PLLI2S 锁定且 PLLI2SRDYDIE 置 1 时，此位由硬件置 1。
在 PLLRI2SDYC 位置 1 时，此位由软件清零。
0：当前未因 PLLI2S 锁定而引起时钟就绪中断
1：因 PLLI2S 锁定而引起时钟就绪中断

位 4 **PLLRDYF**：主 PLL (PLL) 就绪中断标志 (Main PLL (PLL) ready interrupt flag)
当 PLL 锁定且 PLLRDYDIE 置 1 时，此位由硬件置 1。
在 PLLRDYC 位置 1 时，此位由软件清零。
0：当前未因 PLL 锁定而引起时钟就绪中断
1：因 PLL 锁定而引起时钟就绪中断

位 3 **HSERDYF**：HSE 就绪中断标志 (HSE ready interrupt flag)
当外部高速时钟变得稳定且 HSERDYDIE 置 1 时，此位由硬件置 1。
在 HSERDYC 位置 1 时，此位由软件清零。
0：当前未因 HSE 振荡器引起时钟就绪中断
1：因 HSE 振荡器引起时钟就绪中断

位 2 **HSIRDYF**：HSI 就绪中断标志 (HSI ready interrupt flag)
当内部高速时钟变得稳定且 HSIRDYDIE 置 1 时，此位由硬件置 1。
在 HSIRDYC 位置 1 时，此位由软件清零。
0：当前未因 HSI 振荡器引起时钟就绪中断
1：因 HSI 振荡器引起时钟就绪中断

位 1 **LSERDYF**：LSE 就绪中断标志 (LSE ready interrupt flag)
当外部低速时钟变得稳定且 LSERDYDIE 置 1 时，此位由硬件置 1。
在 LSERDYC 位置 1 时，此位由软件清零。
0：当前未因 LSE 振荡器引起时钟就绪中断
1：因 LSE 振荡器引起时钟就绪中断

位 0 **LSIRDYF**：LSI 就绪中断标志 (LSI ready interrupt flag)
当内部低速时钟变得稳定且 LSIRDYDIE 置 1 时，此位由硬件置 1。
在 LSIRDYC 位置 1 时，此位由软件清零。
0：当前未因 LSI 振荡器引起时钟就绪中断
1：因 LSI 振荡器引起时钟就绪中断

5.3.5 RCC AHB1 外设复位寄存器 (RCC_AHB1RSTR)

RCC AHB1 peripheral reset register

偏移地址：0x10

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.	OTGHS RST	Res.	Res.	Res.	ETHMAC RST	Res.	DMA2D RST	DMA2 RST	DMA1 RST	Res.	Res.	Res.	Res.	Res.
		rw				rw		rw	rw	rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRCRST	Res.	GPIOKRST	GPIOJRST	GPIOIRST	GPIOHRST	GPIOGGRST	GPIOFRST	GPIOERST	GPIODRST	GPIOCRST	GPIOBRST	GPIOARST
			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 **OTGHSRST**：USB OTG HS 模块复位 (USB OTG HS module reset)
此位由软件置 1 和清零。
0：不复位 USB OTG HS 模块
1：复位 USB OTG HS 模块

位 28:26 保留，必须保持复位值。

位 25 **ETHMACRST**：以太网 MAC 复位 (Ethernet MAC reset)
此位由软件置 1 和清零。
0：不复位以太网 MAC
1：复位以太网 MAC

位 24 保留，必须保持复位值。

位 23 **DMA2DRST**：DMA2D 复位 (DMA2D reset)
此位由软件置 1 和复位。
0：不复位 DMA2D
1：复位 DMA2D

位 22 **DMA2RST**：DMA2 复位 (DMA2 reset)
此位由软件置 1 和清零。
0：不复位 DMA2
1：复位 DMA2

位 21 **DMA1RST**：DMA2 复位 (DMA2 reset)
此位由软件置 1 和清零。
0：不复位 DMA2
1：复位 DMA2

位 20:13 保留，必须保持复位值。

位 12 **CRCRST**：CRC 复位 (CRC reset)
此位由软件置 1 和清零。
0：不复位 CRC
1：复位 CRC

位 11 保留，必须保持复位值。

位 10 **GPIOKRST**：IO 端口 K 复位 (IO port K reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 K
1：复位 IO 端口 K

位 9 **GPIOJRST**：IO 端口 J 复位 (IO port J reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 J
1：复位 IO 端口 J

位 8 **GPIOIRST**：IO 端口 I 复位 (IO port I reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 I
1：复位 IO 端口 I

位 7 **GPIOHRST**：IO 端口 H 复位 (IO port H reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 H
1：复位 IO 端口 H

位 6 **GPIOGRST**：IO 端口 G 复位 (IO port G reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 G
1：复位 IO 端口 G

位 5 **GPIOFRST**：IO 端口 F 复位 (IO port F reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 F
1：复位 IO 端口 F

位 4 **GPIOERST**：IO 端口 E 复位 (IO port E reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 E
1：复位 IO 端口 E

位 3 **GPIODRST**：IO 端口 D 复位 (IO port D reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 D
1：复位 IO 端口 D

位 2 **GPIOCRST**：IO 端口 C 复位 (IO port C reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 C
1：复位 IO 端口 C

位 1 **GPIOBRST**：IO 端口 B 复位 (IO port B reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 B
1：复位 IO 端口 B

位 0 **GPIOARST**：IO 端口 A 复位 (IO port A reset)
此位由软件置 1 和清零。
0：不复位 IO 端口 A
1：复位 IO 端口 A

5.3.6 RCC AHB2 外设复位寄存器 (RCC_AHB2RSTR)

RCC AHB2 peripheral reset register

偏移地址：0x14

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.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTGFS RST	RNG RST	HASH RST	CRYP RST	Res.	Res.	Res.	DCMI RST
								rw	rw	rw	rw				rw

位 31:8 保留，必须保持复位值。

位 7 **OTGFSRST**：USB OTG FS 模块复位 (USB OTG FS module reset)
由软件置 1 和清零。
0：不复位 USB OTG FS 模块
1：复位 USB OTG FS 模块

位 6 **RNGRST**：随机数发生器模块复位 (Random number generator module reset)
由软件置 1 和清零。
0：不复位随机数发生器模块
1：复位随机数发生器模块

位 5 **HASHRST**：散列模块复位 (Hash module reset)
由软件置 1 和清零。
0：不复位散列模块
1：复位散列模块

位 4 **CRYPRST**：加密模块复位 (Cryptographic module reset)
由软件置 1 和清零。
0：不复位加密模块
1：复位加密模块

位 3:1 保留，必须保持复位值。

位 0 **DCMIRST**：摄像头接口复位 (Camera interface reset)
由软件置 1 和清零。
0：不复位摄像头接口
1：复位摄像头接口

5.3.7 RCC AHB3 外设复位寄存器 (RCC_AHB3RSTR)

RCC AHB3 peripheral reset register

偏移地址：0x18

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.	Res.	Res.	Res.
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.	QSPIRST	FMCRST
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **QSPIRST**：Quad SPI 存储控制器复位 (Quad SPI memory controller reset)
由软件置 1 和清零。
0：不复位 QUADSPI 存储控制器
1：复位 QUADSPI 存储控制器

位 0 **FMCRST**：灵活的存储控制器模块复位 (Flexible memory controller module reset)
由软件置 1 和清零。
0：不复位 FMC 模块
1：复位 FMC 模块

5.3.8 RCC APB1 外设复位寄存器 (RCC_APB1RSTR)

RCC APB1 peripheral reset register

偏移地址：0x20

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
UART8RST	UART7RST	DACRST	PWRRST	CECRST	CAN2RST	CAN1RST	I2C4RST	I2C3RST	I2C2RST	I2C1RST	UART5RST	UART4RST	UART3RST	UART2RST	SPDIFRXRST
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
SPI3RST	SPI2RST	Res.	Res.	WWDGRST	Res.	LPTIM1RST	TIM14RST	TIM13RST	TIM12RST	TIM7RST	TIM6RST	TIM5RST	TIM4RST	TIM3RST	TIM2RST
rw	rw			rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **UART8RST**：UART8 复位 (UART8 reset)
由软件置 1 和清零。
0：不复位 UART8
1：复位 UART8

位 30 **UART7RST**：UART7 复位 (UART7 reset)
由软件置 1 和清零。
0：不复位 UART7
1：复位 UART7

位 29 **DACRST**：DAC 复位 (DAC reset)
由软件置 1 和清零。
0：不复位 DAC 接口
1：复位 DAC 接口

位 28 **PWRRST**：电源接口复位 (Power interface reset)
由软件置 1 和清零。
0：不复位电源接口
1：复位电源接口

位 27 **CECRST**：HDMI-CEC 复位 (HDMI-CEC reset)
由软件置 1 和清零。
0：不复位 HDMI-CEC
1：复位 HDMI-CEC

位 26 **CAN2RST**：CAN2 复位 (CAN2 reset)
由软件置 1 和清零。
0：不复位 CAN2
1：复位 CAN2

位 25 **CAN1RST**：CAN1 复位 (CAN1 reset)
由软件置 1 和清零。
0：不复位 CAN1
1：复位 CAN1

位 24 **I2C4RST**：I2C4 复位 (I2C4 reset)
由软件置 1 和清零。
0：不复位 I2C4
1：复位 I2C4

位 23 **I2C3RST**：I2C3 复位 (I2C3 reset)
由软件置 1 和清零。
0：不复位 I2C3
1：复位 I2C3

位 22 **I2C2RST**：I2C2 复位 (I2C2 reset)
由软件置 1 和清零。
0：不复位 I2C2
1：复位 I2C2

位 21 **I2C1RST**：I2C1 复位 (I2C1 reset)
由软件置 1 和清零。
0：不复位 I2C1
1：复位 I2C1

位 20 **UART5RST**：UART5 复位 (UART5 reset)
由软件置 1 和清零。
0：不复位 UART5
1：复位 UART5

位 19 **UART4RST**：UART4 复位 (UART4 reset)
由软件置 1 和清零。
0：不复位 UART4
1：复位 UART4

位 18 **USART3RST**：USART3 复位 (USART3 reset)
由软件置 1 和清零。
0：不复位 USART3
1：复位 USART3

位 17 **USART2RST**：USART2 复位 (USART2 reset)
由软件置 1 和清零。
0：不复位 USART2
1：复位 USART2

位 16 **SPDIFRXRST**：SPDIFRX 复位 (SPDIFRX reset)
由软件置 1 和清零。
0：不复位 SPDIFRX
1：复位 SPDIFRX

位 15 **SPI3RST**：SPI3 复位 (SPI3 reset)
由软件置 1 和清零。
0：不复位 SPI3
1：复位 SPI3

位 14 **SPI2RST**：SPI2 复位 (SPI2 reset)
由软件置 1 和清零。
0：不复位 SPI2
1：复位 SPI2

位 13:12 保留，必须保持复位值。

位 11 **WWDGRST**：窗口看门狗复位 (Window watchdog reset)
由软件置 1 和清零。
0：不复位窗口看门狗
1：复位窗口看门狗

位 10 保留，必须保持复位值。

位 9 **LPTIM1RST**：低功耗定时器 1 复位 (Low power timer 1 reset)
由软件置 1 和清零。
0：不复位 LPTMI1
1：复位 LPTMI1

位 8 **TIM14RST**：TIM14 复位 (TIM14 reset)
由软件置 1 和清零。
0：不复位 TIM14
1：复位 TIM14

位 7 **TIM13RST**：TIM13 复位 (TIM13 reset)
由软件置 1 和清零。
0：不复位 TIM13
1：复位 TIM13

位 6 **TIM12RST**：TIM12 复位 (TIM12 reset)
由软件置 1 和清零。
0：不复位 TIM12
1：复位 TIM12

位 5 **TIM7RST**：TIM7 复位 (TIM7 reset)
由软件置 1 和清零。
0：不复位 TIM7
1：复位 TIM7

位 4 **TIM6RST**：TIM6 复位 (TIM6 reset)
由软件置 1 和清零。
0：不复位 TIM6
1：复位 TIM6

位 3 **TIM5RST**：TIM5 复位 (TIM5 reset)
由软件置 1 和清零。
0：不复位 TIM5
1：复位 TIM5

位 2 **TIM4RST**：TIM4 复位 (TIM4 reset)
由软件置 1 和清零。
0：不复位 TIM4
1：复位 TIM4

位 1 **TIM3RST**：TIM3 复位 (TIM3 reset)
由软件置 1 和清零。
0：不复位 TIM3
1：复位 TIM3

位 0 **TIM2RST**：TIM2 复位 (TIM2 reset)
由软件置 1 和清零。
0：不复位 TIM2
1：复位 TIM2

5.3.9 RCC APB2 外设复位寄存器 (RCC_APB2RSTR)

RCC APB2 peripheral reset register

偏移地址：0x24

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.	LTDC RST	Res.	Res.	SAI2 RST	SAI1 RST	SPI6 RST	SPI5 RST	Res.	TIM11 RST	TIM10 RST	TIM9 RST
					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.	SYSCFG RST	SPI4 RST	SPI1 RST	SDMMC1 RST	Res.	Res.	ADC RST	Res.	Res.	USART6 RST	USART1 RST	Res.	Res.	TIM8 RST	TIM1 RST
	rw	rw	rw	rw			rw			rw	rw			rw	rw

位 31:27 保留，必须保持复位值。

位 26 **LTDCRST**：LTDC 复位 (LTDC reset)
此位由软件置 1 和复位。
0：不复位 LCD-TFT
1：复位 LCD-TFT

位 27:24 保留，必须保持复位值。

位 23 **SAI2RST**：SAI2 复位 (SAI2 reset)
此位由软件置 1 和清零。
0：不复位 SAI2
1：复位 SAI2

位 22 **SAI1RST**：SAI1 复位 (SAI1 reset)
此位由软件置 1 和复位。
0：不复位 SAI1
1：复位 SAI1

位 21 **SPI6RST**：SPI6 复位 (SPI6 reset)
此位由软件置 1 和清零。
0：不复位 SPI6
1：复位 SPI6

位 20 **SPI5RST**：SPI5 复位 (SPI5 reset)
此位由软件置 1 和清零。
0：不复位 SPI5
1：复位 SPI5

位 19 保留，必须保持复位值。

位 18 **TIM11RST**：TIM11 复位 (TIM11 reset)
此位由软件置 1 和清零。
0：不复位 TIM11
1：复位 TIM14

位 17 **TIM10RST**：TIM10 复位 (TIM10 reset)
此位由软件置 1 和清零。
0：不复位 TIM10
1：复位 TIM10

位 16 **TIM9RST**：TIM9 复位 (TIM9 reset)
此位由软件置 1 和清零。
0：不复位 TIM9
1：复位 TIM9

位 15 保留，必须保持复位值。

位 14 **SYSCFGRST**：系统配置控制器复位 (System configuration controller reset)
此位由软件置 1 和清零。
0：不复位系统配置控制器
1：复位系统配置控制器

位 13 **SPI4RST**：SPI4 复位 (SPI4 reset)
此位由软件置 1 和清零。
0：不复位 SPI4
1：复位 SPI4

位 12 **SPI1RST**：SPI1 复位 (SPI1 reset)
此位由软件置 1 和清零。
0：不复位 SPI1
1：复位 SPI1

位 11 **SDMMC1RST**：SDMMC1 复位 (SDMMC1 reset)
此位由软件置 1 和清零。
0：不复位 SDMMC1 模块
1：复位 SDMMC1 模块

位 10:9 保留，必须保持复位值。

位 8 **ADCRST**：ADC 接口复位（所有 ADC 共用）(ADC interface reset (common to all ADCs))
此位由软件置 1 和清零。
0：不复位 ADC 接口
1：复位 ADC 接口

位 7:6 保留，必须保持复位值。

位 5 **USART6RST**：USART6 复位 (USART6 reset)
此位由软件置 1 和清零。
0：不复位 USART6
1：复位 USART6

位 4 **USART1RST**：USART1 复位 (USART1 reset)
此位由软件置 1 和清零。
0：不复位 USART1
1：复位 USART1

位 3:2 保留，必须保持复位值。

位 1 **TIM8RST**：TIM8 复位 (TIM8 reset)
此位由软件置 1 和清零。
0：不复位 TIM8
1：复位 TIM8

位 0 **TIM1RST**：TIM1 复位 (TIM1 reset)
此位由软件置 1 和清零。
0：不复位 TIM1
1：复位 TIM1

5.3.10 RCC AHB1 外设时钟寄存器 (RCC_AHB1ENR)

RCC AHB1 peripheral clock register

偏移地址：0x30

复位值：0x0010 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.	OTGHS ULPIEN	OTGHS EN	ETHM ACPTP EN	ETHM ACRX EN	ETHM ACTX EN	ETHMA CEN	Res.	DMA2D EN	DMA2 EN	DMA1 EN	DTCMRA MEN	Res.	BKPSR AMEN	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
Res.	Res.	Res.	CRC EN	Res.	GPIOK EN	GPIOJ EN	GPIOI EN	GPIOH EN	GPIOG EN	GPIOF EN	GPIOE EN	GPIOD EN	GPIOC EN	GPIO BEN	GPIO AEN
			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 **OTGHSULPIEN**：USB OTG HSULPI 时钟使能 (USB OTG HSULPI clock enable)
此位由软件置 1 和清零。
0：禁止 USB OTG HS ULPI 时钟
1：使能 USB OTG HS ULPI 时钟

位 29 **OTGHSEN**：USB OTG HS 时钟使能 (USB OTG HS clock enable)
此位由软件置 1 和清零。
0：禁止 USB OTG HS 时钟
1：使能 USB OTG HS 时钟

位 28 **ETHMACPTPEN**：以太网 PTP 时钟使能 (Ethernet PTP clock enable)
此位由软件置 1 和清零。
0：禁止以太网 PTP 时钟
1：使能以太网 PTP 时钟

位 27 **ETHMACRXEN**：以太网接收时钟使能 (Ethernet Reception clock enable)
此位由软件置 1 和清零。
0：禁止以太网接收时钟
1：使能以太网接收时钟

位 26 **ETHMACTXEN**：以太网发送时钟使能 (Ethernet Transmission clock enable)
此位由软件置 1 和清零。
0：禁止以太网发送时钟
1：使能以太网发送时钟

位 25 **ETHMACEN**：以太网 MAC 时钟使能 (Ethernet MAC clock enable)
此位由软件置 1 和清零。
0：禁止以太网 MAC 时钟
1：使能以太网 MAC 时钟

位 24 保留，必须保持复位值。

位 23 **DMA2DEN**：DMA2D 时钟使能 (DMA2D clock enable)
此位由软件置 1 和清零。
0：禁止 DMA2D 时钟
1：使能 DMA2D 时钟

位 22 **DMA2EN**：DMA2 时钟使能 (DMA2 clock enable)
此位由软件置 1 和清零。
0：禁止 DMA2 时钟
1：使能 DMA2 时钟

位 21 **DMA1EN**：DMA1 时钟使能 (DMA1 clock enable)
此位由软件置 1 和清零。
0：禁止 DMA1 时钟
1：使能 DMA1 时钟

位 20 **DTCMRAMEN**：DTCM 数据 RAM 时钟使能 (DTCM data RAM clock enable)
此位由软件置 1 和清零。
0：禁止 DTCM RAM 时钟
1：使能 DTCM RAM 时钟

位 19 保留，必须保持复位值。

位 18 **BKPSRAMEN**：备份 SRAM 接口时钟使能 (Backup SRAM interface clock enable)
此位由软件置 1 和清零。
0：禁止备份 SRAM 接口时钟
1：使能备份 SRAM 接口时钟

位 17:13 保留，必须保持复位值。

位 12 **CRCEN**：CRC 时钟使能 (CRC clock enable)
此位由软件置 1 和清零。
0：禁止 CRC 时钟
1：使能 CRC 时钟

位 11 保留，必须保持复位值。

位 10 **GPIOKEN**：IO 端口 K 时钟使能 (IO port K clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 K 时钟
1：使能 IO 端口 K 时钟

位 9 **GPIOJEN**：IO 端口 J 时钟使能 (IO port J clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 J 时钟
1：使能 IO 端口 J 时钟

位 8 **GPIOIEN**：IO 端口 I 时钟使能 (IO port I clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 I 时钟
1：使能 IO 端口 I 时钟

位 7 **GPIOHEN**：IO 端口 H 时钟使能 (IO port H clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 H 时钟
1：使能 IO 端口 H 时钟

位 6 **GPIOGEN**：IO 端口 G 时钟使能 (IO port G clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 G 时钟
1：使能 IO 端口 G 时钟

位 5 **GPIOFEN**：IO 端口 F 时钟使能 (IO port F clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 F 时钟
1：使能 IO 端口 F 时钟

位 4 **GPIOEEN**：IO 端口 E 时钟使能 (IO port E clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 E 时钟
1：使能 IO 端口 E 时钟

位 3 **GPIODEN**：IO 端口 D 时钟使能 (IO port D clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 D 时钟
1：使能 IO 端口 D 时钟

位 2 **GPIOCEN**：IO 端口 C 时钟使能 (IO port C clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 C 时钟
1：使能 IO 端口 C 时钟

位 1 **GPIOBEN**：IO 端口 B 时钟使能 (IO port B clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 B 时钟
1：使能 IO 端口 B 时钟

位 0 **GPIOAEN**：IO 端口 A 时钟使能 (IO port A clock enable)
此位由软件置 1 和清零。
0：禁止 IO 端口 A 时钟
1：使能 IO 端口 A 时钟

5.3.11 RCC AHB2 外设时钟使能寄存器 (RCC_AHB2ENR)

RCC AHB2 peripheral clock enable register

偏移地址：0x34

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.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTGFSEN	RNGEN	HASHEN	CRYPEN	Res.	Res.	Res.	DCMIEN
								rw	rw	rw	rw				rw

位 31:8 保留，必须保持复位值。

位 7 **OTGFSEN**：USB OTG FS 时钟使能 (USB OTG FS clock enable)
此位由软件置 1 和清零。
0：禁止 USB OTG FS 时钟
1：使能 USB OTG FS 时钟

位 6 **RNGEN**：随机数发生器时钟使能 (Random number generator clock enable)
此位由软件置 1 和清零。
0：禁止随机数发生器时钟
1：使能随机数发生器时钟

位 5 **HASHEN**：散列模块时钟使能 (Hash modules clock enable)
此位由软件置 1 和清零。
0：禁止散列模块时钟
1：使能散列模块时钟

位 4 **CRYPEN**：加密模块时钟使能 (Cryptographic modules clock enable)
此位由软件置 1 和清零。
0：禁止加密模块时钟
1：使能加密模块时钟

位 3:1 保留，必须保持复位值。

位 0 **DCMIEN**：摄像头接口使能 (Camera interface enable)
此位由软件置 1 和清零。
0：禁止摄像头接口时钟
1：使能摄像头接口时钟

5.3.12 RCC AHB3 外设时钟使能寄存器 (RCC_AHB3ENR)

RCC AHB3 peripheral clock enable register

偏移地址：0x38

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.	Res.	Res.	Res.
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.	QSPIEN	FMCEN
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **QSPIEN**：Quad SPI 存储控制器时钟使能 (Quad SPI memory controller clock enable)
此位由软件置 1 和清零。
0：禁止 QUASPI 控制器时钟
1：使能 QUASPI 控制器时钟

位 0 **FMCEN**：灵活的存储控制器时钟使能 (Flexible memory controller clock enable)
此位由软件置 1 和清零。
0：禁止 FMC 时钟
1：使能 FMC 时钟

5.3.13 RCC APB1 外设时钟使能寄存器 (RCC_APB1ENR)

RCC APB1 peripheral clock enable register

偏移地址：0x40

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
UART8 EN	UART7 EN	DAC EN	PWR EN	CEC EN	CAN2 EN	CAN1 EN	I2C4 EN	I2C3 EN	I2C2 EN	I2C1 EN	UART5 EN	UART4 EN	USART 3 EN	USART 2 EN	SPDIFRX EN
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
SPI3 EN	SPI2 EN	Res.	Res.	WWDG EN	Res.	LPTIM1 EN	TIM14 EN	TIM13 EN	TIM12 EN	TIM7 EN	TIM6 EN	TIM5 EN	TIM4 EN	TIM3 EN	TIM2 EN
rw	rw			rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **UART8EN**：UART8 时钟使能 (UART8 clock enable)
此位由软件置 1 和清零。
0：禁止 UART8 时钟
1：使能 UART8 时钟

位 30 **UART7EN**：UART7 时钟使能 (UART7 clock enable)
此位由软件置 1 和清零。
0：禁止 UART7 时钟
1：使能 UART7 时钟

位 29 **DACEN**：DAC 接口时钟使能 (DAC interface clock enable)
此位由软件置 1 和清零。
0：禁止 DAC 接口时钟
1：使能 DAC 接口时钟

位 28 **PWREN**：电源接口时钟使能 (Power interface clock enable)
此位由软件置 1 和清零。
0：禁止电源接口时钟
1：使能电源接口时钟

位 27 **CECEN**：HDMI-CEN 时钟使能 (HDMI-CEN clock enable)
此位由软件置 1 和清零。
0：禁止 HDMI-CEN 时钟
1：使能 HDMI-CEN 时钟

位 26 **CAN2EN**：CAN 2 时钟使能 (CAN 2 clock enable)
此位由软件置 1 和清零。
0：禁止 CAN 2 时钟
1：使能 CAN 2 时钟

位 25 **CAN1EN**：CAN 1 时钟使能 (CAN 1 clock enable)
此位由软件置 1 和清零。
0：禁止 CAN 1 时钟
1：使能 CAN 1 时钟

位 24 **I2C4**：I2C4 时钟使能 (I2C4 clock enable)
此位由软件置 1 和清零。
0：禁止 I2C4 时钟
1：使能 I2C4 时钟

位 23 **I2C3EN**：I2C3 时钟使能 (I2C3 clock enable)
此位由软件置 1 和清零。
0：禁止 I2C3 时钟
1：使能 I2C3 时钟

位 22 **I2C2EN**：I2C2 时钟使能 (I2C2 clock enable)
此位由软件置 1 和清零。
0：禁止 I2C2 时钟
1：使能 I2C2 时钟

位 21 **I2C1EN**：I2C1 时钟使能 (I2C1 clock enable)
此位由软件置 1 和清零。
0：禁止 I2C1 时钟
1：使能 I2C1 时钟

位 20 **UART5EN**：UART5 时钟使能 (UART5 clock enable)
此位由软件置 1 和清零。
0：禁止 UART5 时钟
1：使能 UART5 时钟

位 19 **UART4EN**：UART4 时钟使能 (UART4 clock enable)
此位由软件置 1 和清零。
0：禁止 UART4 时钟
1：使能 UART4 时钟

位 18 **USART3EN**：USART3 时钟使能 (USART3 clock enable)
此位由软件置 1 和清零。
0：禁止 USART3 时钟
1：使能 USART3 时钟

位 17 **USART2EN**：USART2 时钟使能 (USART2 clock enable)
此位由软件置 1 和清零。
0：禁止 USART2 时钟
1：使能 USART2 时钟

位 16 **SPDIFRXEN**：SPDIFRX 时钟使能 (SPDIFRX clock enable)
此位由软件置 1 和清零。
0：禁止 SPDIFRX 时钟
1：使能 SPDIFRX 时钟

位 15 **SPI3EN**：SPI3 时钟使能 (SPI3 clock enable)
此位由软件置 1 和清零。
0：禁止 SPI3 时钟
1：使能 SPI3 时钟

位 14 **SPI2EN**：SPI2 时钟使能 (SPI2 clock enable)
此位由软件置 1 和清零。
0：禁止 SPI2 时钟
1：使能 SPI2 时钟

位 13:12 保留，必须保持复位值。

位 11 **WWDGEN**：窗口看门狗时钟使能 (Window watchdog clock enable)
此位由软件置 1 和清零。
0：禁止窗口看门狗时钟
1：使能窗口看门狗时钟

位 10 保留，必须保持复位值。

位 9 **LPTIM1EN**：低功耗定时器 1 时钟使能
此位由软件置 1 和清零。
0：禁止 LPTIM1 时钟
1：使能 LPTIM1 时钟

位 8 **TIM14EN**：TIM14 时钟使能 (TIM14 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM14 时钟
1：使能 TIM14 时钟

位 7 **TIM13EN**：TIM13 时钟使能 (TIM13 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM13 时钟
1：使能 TIM13 时钟

位 6 **TIM12EN**：TIM12 时钟使能 (TIM12 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM12 时钟
1：使能 TIM12 时钟

位 5 **TIM7EN**：TIM7 时钟使能 (TIM7 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM7 时钟
1：使能 TIM7 时钟

位 4 **TIM6EN**：TIM6 时钟使能 (TIM6 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM6 时钟
1：使能 TIM6 时钟

位 3 **TIM5EN**：TIM5 时钟使能 (TIM5 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM5 时钟
1：使能 TIM5 时钟

位 2 **TIM4EN**：TIM4 时钟使能 (TIM4 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM4 时钟
1：使能 TIM4 时钟

位 1 **TIM3EN**：TIM3 时钟使能 (TIM3 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM3 时钟
1：使能 TIM3 时钟

位 0 **TIM2EN**：TIM2 时钟使能 (TIM2 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM2 时钟
1：使能 TIM2 时钟

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

RCC APB2 peripheral clock enable register

偏移地址：0x44

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.	LTDC EN	Res.	Res.	SAI2EN	SAI1EN	SPI6EN	SPI5EN	Res.	TIM11 EN	TIM10 EN	TIM9 EN
					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.	SYSCFG EN	SPI4 EN	SPI1 EN	SDMMC1 EN	ADC3 EN	ADC2 EN	ADC1 EN	Res.	Res.	USART6 EN	USART1 EN	Res.	Res.	TIM8 EN	TIM1 EN
	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 **LTDCEN**：LTDC 时钟使能 (LTDC clock enable)
此位由软件置 1 和清零。
0：禁止 LTDC 时钟
1：使能 LTDC 时钟

位 27:24 保留，必须保持复位值。

位 23 **SAI2EN**：SAI2 时钟使能 (SAI2 clock enable)
此位由软件置 1 和清零。
0：禁止 SAI2 时钟
1：使能 SAI2 时钟

位 22 **SAI1EN**：SAI1 时钟使能 (SAI1 clock enable)
此位由软件置 1 和清零。
0：禁止 SAI1 时钟
1：使能 SAI1 时钟

位 21 **SPI6EN**：SPI6 时钟使能 (SPI6 clock enable)
此位由软件置 1 和清零。
0：禁止 SPI6 时钟
1：使能 SPI6 时钟

位 20 **SPI5EN**：SPI5 时钟使能 (SPI5 clock enable)
此位由软件置 1 和清零。
0：禁止 SPI5 时钟
1：使能 SPI5 时钟

位 18 **TIM11EN**：TIM11 时钟使能 (TIM11 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM11 时钟
1：使能 TIM11 时钟

位 17 **TIM10EN**：TIM10 时钟使能 (TIM10 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM10 时钟
1：使能 TIM10 时钟

位 16 **TIM9EN**：TIM9 时钟使能 (TIM9 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM9 时钟
1：使能 TIM9 时钟

位 15 保留，必须保持复位值。

位 14 **SYSCFGEN**：系统配置控制器时钟使能 (System configuration controller clock enable)
此位由软件置 1 和清零。
0：禁止系统配置控制器时钟
1：使能系统配置控制器时钟

位 13 **SPI4EN**：SPI4 时钟使能 (SPI4 clock enable)
此位由软件置 1 和清零。
0：禁止 SPI4 时钟
1：使能 SPI4 时钟

位 12 **SPI1EN**：SPI1 时钟使能 (SPI1 clock enable)
此位由软件置 1 和清零。
0：禁止 SPI1 时钟
1：使能 SPI1 时钟

位 11 **SDMMC1EN**：SDMMC1 时钟使能 (SDMMC1 clock enable)
此位由软件置 1 和清零。
0：禁止 SDMMC1 模块时钟
1：使能 SDMMC1 模块时钟

位 10 **ADC3EN**：ADC3 时钟使能 (ADC3 clock enable)
此位由软件置 1 和清零。
0：禁止 ADC3 时钟
1：使能 ADC3 时钟

位 9 **ADC2EN**：ADC2 时钟使能 (ADC2 clock enable)
此位由软件置 1 和清零。
0：禁止 ADC2 时钟
1：使能 ADC2 时钟

位 8 **ADC1EN**：ADC1 时钟使能 (ADC1 clock enable)
此位由软件置 1 和清零。
0：禁止 ADC1 时钟
1：使能 ADC1 时钟

位 7:6 保留，必须保持复位值。

位 5 **USART6EN**：USART6 时钟使能 (USART6 clock enable)
此位由软件置 1 和清零。
0：禁止 USART6 时钟
1：使能 USART6 时钟

位 4 **USART1EN**：USART1 时钟使能 (USART1 clock enable)
此位由软件置 1 和清零。
0：禁止 USART1 时钟
1：使能 USART1 时钟

位 3:2 保留，必须保持复位值。

位 1 **TIM8EN**：TIM8 时钟使能 (TIM8 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM8 时钟
1：使能 TIM8 时钟

位 0 **TIM1EN**：TIM1 时钟使能 (TIM1 clock enable)
此位由软件置 1 和清零。
0：禁止 TIM1 时钟
1：使能 TIM1 时钟

5.3.15 低功耗模式寄存器中的 RCC AHB1 外设时钟使能 (RCC_AHB1LPENR)

RCC AHB1 peripheral clock enable in low power mode register

偏移地址：0x50

复位值：0x7EF7 B7FFh

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.	OTGHS ULPI LPEN	OTGHS LPEN	ETHPTP LPEN	ETHRX LPEN	ETHTX LPEN	ETHMA C LPEN	Res.	DMA2D LPEN	DMA2 LPEN	DMA1 LPEN	DTCM LPEN	Res.	BKPS RAM LPEN	SRAM2 LPEN	SRAM1 LPEN
	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
FLITF LPEN	Res.	AXI LPEN	CRC LPEN	Res.	GPIOK LPEN	GPIOIJ LPEN	GPIOI LPEN	GPIOH LPEN	GPIOGG LPEN	GPIOF LPEN	GPIOE LPEN	GPIOD LPEN	GPIOC LPEN	GPIOB LPEN	GPIOA LPEN
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 **OTGHSULPILPEN**：睡眠模式期间的 USB OTG HS ULPI 时钟使能 (USB OTG HS ULPI clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 USB OTG HS ULPI 时钟
1：睡眠模式期间使能 USB OTG HS ULPI 时钟

位 29 **OTGHSLPEN**：睡眠模式期间的 USB OTG HS 时钟使能 (USB OTG HS clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 USB OTG HS 时钟
1：睡眠模式期间使能 USB OTG HS 时钟

位 28 **ETHMACPTPLPEN**：睡眠模式期间的以太网 PTP 时钟使能 (Ethernet PTP clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止以太网 PTP 时钟
1：睡眠模式期间使能以太网 PTP 时钟

位 27 **ETHMACRXLPEN**：睡眠模式期间的以太网接收时钟使能 (Ethernet reception clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止以太网接收时钟
1：睡眠模式期间使能以太网接收时钟

位 26 **ETHMACTXLPEN**：睡眠模式期间的以太网发送时钟使能 (Ethernet transmission clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止以太网发送时钟

1：睡眠模式期间使能以太网发送时钟

位 25 **ETHMACLPEN**：睡眠模式期间的以太网 MAC 时钟使能 (Ethernet MAC clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止以太网 MAC 时钟

1：睡眠模式期间使能以太网 MAC 时钟

位 24 保留，必须保持复位值。

位 23 **DMA2DLPEN**：睡眠模式期间的 DMA2D 时钟使能 (DMA2D clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 DMA2D 时钟

1：睡眠模式期间使能 DMA2D 时钟

位 22 **DMA2LPEN**：睡眠模式期间的 DMA2 时钟使能 (DMA2 clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 DMA2 时钟

1：睡眠模式期间使能 DMA2 时钟

位 21 **DMA1LPEN**：睡眠模式期间的 DMA1 时钟使能 (DMA1 clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 DMA1 时钟

1：睡眠模式期间使能 DMA1 时钟

位 20 **DTCMLPEN**：睡眠模式期间的 DTCM RAM 接口时钟使能 (DTCM RAM interface clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 DTCM RAM 接口时钟

1：睡眠模式期间使能 DTCM RAM 接口时钟

位 19 保留，必须保持复位值。

位 18 **BKPSRAMLPEN**：睡眠模式期间的备份 SRAM 接口时钟使能 (Backup SRAM interface clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止备份 SRAM 接口时钟

1：睡眠模式期间使能备份 SRAM 接口时钟

位 17 **SRAM2LPEN**：睡眠模式期间的 SRAM2 接口时钟使能 (SRAM2 interface clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 SRAM2 接口时钟

1：睡眠模式期间使能 SRAM2 接口时钟

位 16 **SRAM1LPEN**：睡眠模式期间的 SRAM1 接口时钟使能 (SRAM1 interface clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 SRAM1 接口时钟

1：睡眠模式期间使能 SRAM1 接口时钟

位 15 **FLITFLPEN**：睡眠模式期间的 Flash 接口时钟使能 (Flash interface clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 Flash 接口时钟

1：睡眠模式期间使能 Flash 接口时钟

位 14 保留，必须保持复位值。

位 13 **AXILPEN**：睡眠模式期间的 AXI-AHB 总线桥时钟使能 (AXI to AHB bridge clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 AXI-AHB 总线桥时钟

1：睡眠模式期间使能 AXI-AHB 总线桥时钟

位 12 **CRCLPEN**：睡眠模式期间的 CRC 时钟使能 (CRC clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 CRC 时钟

1：睡眠模式期间使能 CRC 时钟

位 11 保留，必须保持复位值。

位 10 **GPIOKLPEN**：睡眠模式期间的 IO 端口 K 时钟使能 (IO port K clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 IO 端口 K 时钟

1：睡眠模式期间使能 IO 端口 K 时钟

位 9 **GPIOJLPEN**：睡眠模式期间的 IO 端口 J 时钟使能 (IO port J clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 IO 端口 J 时钟

1：睡眠模式期间使能 IO 端口 J 时钟

位 8 **GPIOILPEN**：睡眠模式期间的 IO 端口 I 时钟使能 (IO port I clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 IO 端口 I 时钟

1：睡眠模式期间使能 IO 端口 I 时钟

位 7 **GPIOHLPEN**：睡眠模式期间的 IO 端口 H 时钟使能 (IO port H clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 IO 端口 H 时钟

1：睡眠模式期间使能 IO 端口 H 时钟

位 6 **GPIOGLPEN**：睡眠模式期间的 IO 端口 G 时钟使能 (IO port G clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 IO 端口 G 时钟

1：睡眠模式期间使能 IO 端口 G 时钟

位 5 **GPIOFLPEN**：睡眠模式期间的 IO 端口 F 时钟使能 (IO port F clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 IO 端口 F 时钟

1：睡眠模式期间使能 IO 端口 F 时钟

位 4 **GPIOELPEN**：睡眠模式期间的 IO 端口 E 时钟使能 (IO port E clock enable during Sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 IO 端口 E 时钟

1：睡眠模式期间使能 IO 端口 E 时钟

位 3 **GPIODLPEN**：睡眠模式期间的 IO 端口 D 时钟使能 (IO port D clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 IO 端口 D 时钟
1：睡眠模式期间使能 IO 端口 D 时钟

位 2 **GPIOCLPEN**：睡眠模式期间的 IO 端口 C 时钟使能 (IO port C clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 IO 端口 C 时钟
1：睡眠模式期间使能 IO 端口 C 时钟

位 1 **GPIOBLPEN**：睡眠模式期间的 IO 端口 B 时钟使能 (IO port B clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 IO 端口 B 时钟
1：睡眠模式期间使能 IO 端口 B 时钟

位 0 **GPIOALPEN**：睡眠模式期间的 IO 端口 A 时钟使能 (IO port A clock enable during sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 IO 端口 A 时钟
1：睡眠模式期间使能 IO 端口 A 时钟

5.3.16 低功耗模式寄存器中的 RCC AHB2 外设时钟使能 (RCC_AHB2LPENR)

RCC AHB2 peripheral clock enable in low power mode register

偏移地址：0x54

复位值：0x0000 00F1

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.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTGFS LPEN	RNG LPEN	HASH LPEN	CRYP LPEN	Res.	Res.	Res.	DCMI LPEN
								rw	rw	rw	rw				rw

位 31:8 保留，必须保持复位值。

位 7 **OTGFSLPEN**：睡眠模式期间的 USB OTG FS 时钟使能 (USB OTG FS clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 USB OTG FS 时钟
1：睡眠模式期间使能 USB OTG FS 时钟

位 6 **RNGLPEN**：睡眠模式期间的随机数发生器时钟使能 (Random number generator clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止随机数发生器时钟
1：睡眠模式期间使能随机数发生器时钟

位 5 **HASHLPEN**：睡眠模式期间的散列模块时钟使能 (Hash modules clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止散列模块时钟
1：睡眠模式期间使能散列模块时钟

位 4 **CRYPLPEN**：睡眠模式期间的加密模块时钟使能 (Cryptography modules clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止加密模块时钟

1：睡眠模式期间使能加密模块时钟

位 3:1 保留，必须保持复位值。

位 0 **DCMILPEN**：睡眠模式期间的摄像头接口时钟使能 (Camera interface enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止摄像头接口时钟

1：睡眠模式期间使能摄像头接口时钟

5.3.17 低功耗模式寄存器中的 RCC AHB3 外设时钟使能 (RCC_AHB3LPENR)

RCC AHB3 peripheral clock enable in low power mode register

偏移地址：0x58

复位值：0x0000 0003

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.	Res.	Res.	Res.
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.	QSPI LPEN	FMC LPEN
															rw

位 31:2保留，必须保持复位值。

位 1 **QSPILPEN**：睡眠模式期间的 QUADSPI 存储控制器时钟使能 (QUADSPI memory controller clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 QUADSPI 控制器时钟

1：睡眠模式期间使能 QUADSPI 控制器时钟

位 0 **FMCLPEN**：睡眠模式期间的灵活的存储控制器模块时钟使能 (Flexible memory controller module clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 FMC 模块时钟

1：睡眠模式期间使能 FMC 模块时钟

5.3.18 低功耗模式寄存器中的 RCC APB1 外设时钟使能 (RCC_APB1LPENR)

RCC APB1 peripheral clock enable in low power mode register

偏移地址：0x60

复位值：0xFFFF CBFFh

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
UART8 LPEN	UART7 LPEN	DAC LPEN	PWR LPEN	CEC LPEN	CAN2 LPEN	CAN1 LPEN	I2C4 LPEN	I2C3 LPEN	I2C2 LPEN	I2C1 LPEN	UART5 LPEN	UART4 LPEN	USART3 LPEN	USART2 LPEN	SPDIFRX LPEN
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
SPI3 LPEN	SPI2 LPEN	Res.	Res.	WWDG LPEN	Res.	LPTIM1 LPEN	TIM14 LPEN	TIM13 LPEN	TIM12 LPEN	TIM7 LPEN	TIM6 LPEN	TIM5 LPEN	TIM4 LPEN	TIM3 LPEN	TIM2 LPEN
rw	rw			rw			rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **UART8LPEN**：睡眠模式期间的 UART8 时钟使能 (UART8 clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 UART8 时钟

1：睡眠模式期间使能 UART8 时钟

位 30 **UART7LPEN**：睡眠模式期间的 UART7 时钟使能 (UART7 clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 UART7 时钟

1：睡眠模式期间使能 UART7 时钟

位 29 **DACLPEN**：睡眠模式期间的 DAC 接口时钟使能 (DAC interface clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 DAC 接口时钟

1：睡眠模式期间使能 DAC 接口时钟

位 28 **PWRLPEN**：睡眠模式期间的电源接口时钟使能 (Power interface clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止电源接口时钟

1：睡眠模式期间使能电源接口时钟

位 27 **CECLPEN**：睡眠模式期间的 HDMI-CEN 时钟使能 (HDMI-CEN clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 HDMI-CEN 时钟

1：睡眠模式期间使能 HDMI-CEN 时钟

位 26 **CAN2LPEN**：睡眠模式期间的 CAN 2 时钟使能 (CAN 2 clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 CAN 2 时钟

1：睡眠模式期间使能 CAN 2 时钟

位 25 **CAN1LPEN**：睡眠模式期间的 CAN 1 时钟使能 (CAN 1 clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 CAN 1 时钟

1：睡眠模式期间使能 CAN 1 时钟

位 24 **I2C4LPEN**：睡眠模式期间的 I2C4 时钟使能 (I2C4 clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 I2C4 时钟

1：睡眠模式期间使能 I2C4 时钟

位 23 **I2C3LPEN**：睡眠模式期间的 I2C3 时钟使能 (I2C3 clock enable during Sleep mode)

此位由软件置 1 和清零。

0：睡眠模式期间禁止 I2C3 时钟

1：睡眠模式期间使能 I2C3 时钟

位 22 **I2C2LPEN**：睡眠模式期间的 I2C2 时钟使能 (I2C2 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 I2C2 时钟
1：睡眠模式期间使能 I2C2 时钟

位 21 **I2C1LPEN**：睡眠模式期间的 I2C1 时钟使能 (I2C1 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 I2C1 时钟
1：睡眠模式期间使能 I2C1 时钟

位 20 **UART5LPEN**：睡眠模式期间的 UART5 时钟使能 (UART5 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 UART5 时钟
1：睡眠模式期间使能 UART5 时钟

位 19 **UART4LPEN**：睡眠模式期间的 UART4 时钟使能 (UART4 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 UART4 时钟
1：睡眠模式期间使能 UART4 时钟

位 18 **USART3LPEN**：睡眠模式期间的 USART3 时钟使能 (USART3 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 USART3 时钟
1：睡眠模式期间使能 USART3 时钟

位 17 **USART2LPEN**：睡眠模式期间的 USART2 时钟使能 (USART2 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 USART2 时钟
1：睡眠模式期间使能 USART2 时钟

位 16 **SPDIFRXLPEN**：睡眠模式期间的 SPDIFRX 时钟使能 (SPDIFRX clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SPDIFRX 时钟
1：睡眠模式期间使能 SPDIFRX 时钟

位 15 **SPI3LPEN**：睡眠模式期间的 SPI3 时钟使能 (SPI3 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SPI3 时钟
1：睡眠模式期间使能 SPI3 时钟

位 14 **SPI2LPEN**：睡眠模式期间的 SPI2 时钟使能 (SPI2 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SPI2 时钟
1：睡眠模式期间使能 SPI2 时钟

位 13:12 保留，必须保持复位值。

位 11 **WWDGLPEN**：睡眠模式期间的窗口看门狗时钟使能 (Window watchdog clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止窗口看门狗时钟
1：睡眠模式期间使能窗口看门狗时钟

位 10 保留，必须保持复位值。

位 9 **LPTIM1LPEN**：睡眠模式期间的低功耗定时器 1 时钟使能
此位由软件置 1 和清零。
0：睡眠模式期间禁止 LPTIM1 时钟
1：睡眠模式期间使能 LPTIM1 时钟

位 8 **TIM14LPEN**：睡眠模式期间的 TIM14 时钟使能 (TIM14 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM14 时钟
1：睡眠模式期间使能 TIM14 时钟

位 7 **TIM13LPEN**：睡眠模式期间的 TIM13 时钟使能 (TIM13 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM13 时钟
1：睡眠模式期间使能 TIM13 时钟

位 6 **TIM12LPEN**：睡眠模式期间的 TIM12 时钟使能 (TIM12 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM12 时钟
1：睡眠模式期间使能 TIM12 时钟

位 5 **TIM7LPEN**：睡眠模式期间的 TIM7 时钟使能 (TIM7 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM7 时钟
1：睡眠模式期间使能 TIM7 时钟

位 4 **TIM6LPEN**：睡眠模式期间的 TIM6 时钟使能 (TIM6 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM6 时钟
1：睡眠模式期间使能 TIM6 时钟

位 3 **TIM5LPEN**：睡眠模式期间的 TIM5 时钟使能 (TIM5 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM5 时钟
1：睡眠模式期间使能 TIM5 时钟

位 2 **TIM4LPEN**：睡眠模式期间的 TIM4 时钟使能 (TIM4 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM4 时钟
1：睡眠模式期间使能 TIM4 时钟

位 1 **TIM3LPEN**：睡眠模式期间的 TIM3 时钟使能 (TIM3 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM3 时钟
1：睡眠模式期间使能 TIM3 时钟

位 0 **TIM2LPEN**：睡眠模式期间的 TIM2 时钟使能 (TIM2 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM2 时钟
1：睡眠模式期间使能 TIM2 时钟

5.3.19 低功耗模式寄存器中的 RCC APB2 外设时钟 (RCC_APB2LPENR)

RCC APB2 peripheral clock enabled in low power mode register

偏移地址：0x64

复位值：0x04F7 7F33h

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.	LTDC LPEN	Res.	Res.	SAI2 LPEN	SAI1 LPEN	SPI6 LPEN	SPI5 LPEN	Res.	TIM11 LPEN	TIM10 LPEN	TIM9 LPEN
					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.	SYSCFG LPEN	SPI4 LPEN	SPI1 LPEN	SDMMC1 LPEN	ADC3 LPEN	ADC2 LPEN	ADC1 LPEN	Res.	Res.	USART6 LPEN	USART1 LPEN	Res.	Res.	TIM8 LPEN	TIM1 LPEN
	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 **LTDCLPEN**：睡眠模式期间的 LTDC 时钟使能 (LTDC clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 LTDC 时钟
1：睡眠模式期间使能 LTDC 时钟

位 25:24 保留，必须保持复位值。

位 23 **SAI2LPEN**：睡眠模式期间的 SAI2 时钟使能 (SAI2 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SAI2 时钟
1：睡眠模式期间使能 SAI2 时钟

位 22 **SAI1LPEN**：睡眠模式期间的 SAI1 时钟使能 (SAI1 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SAI1 时钟
1：睡眠模式期间使能 SAI1 时钟

位 21 **SPI6LPEN**：睡眠模式期间的 SPI6 时钟使能 (SPI6 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SPI6 时钟
1：睡眠模式期间使能 SPI6 时钟

位 20 **SPI5LPEN**：睡眠模式期间的 SPI5 时钟使能 (SPI5 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SPI5 时钟
1：睡眠模式期间使能 SPI5 时钟

位 19 保留，必须保持复位值。

位 18 **TIM11LPEN**：睡眠模式期间的 TIM11 时钟使能 (TIM11 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM11 时钟
1：睡眠模式期间使能 TIM11 时钟

位 17 **TIM10LPEN**：睡眠模式期间的 TIM10 时钟使能 (TIM10 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM10 时钟
1：睡眠模式期间使能 TIM10 时钟

位 16 **TIM9LPEN**：睡眠模式期间的 TIM9 时钟使能 (TIM9 clock enable during sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM9 时钟
1：睡眠模式期间使能 TIM9 时钟

位 15 保留，必须保持复位值。

位 14 **SYSCFGLPEN**：睡眠模式期间的系统配置控制器时钟使能 (System configuration controller clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止系统配置控制器时钟
1：睡眠模式期间使能系统配置控制器时钟

位 13 **SPI4LPEN**：睡眠模式期间的 SPI4 时钟使能 (SPI4 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SPI4 时钟
1：睡眠模式期间使能 SPI4 时钟

位 12 **SPI1LPEN**：睡眠模式期间的 SPI1 时钟使能 (SPI1 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SPI1 时钟
1：睡眠模式期间使能 SPI1 时钟

位 11 **SDMMC1LPEN**：睡眠模式期间的 SDMMC1 时钟使能 (SDMMC1 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 SDMMC1 模块时钟
1：睡眠模式期间使能 SDMMC1 模块时钟

位 10 **ADC3LPEN**：睡眠模式期间的 ADC 3 时钟使能 (ADC 3 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 ADC 3 时钟
1：睡眠模式期间使能 ADC 3 时钟

位 9 **ADC2LPEN**：睡眠模式期间的 ADC2 时钟使能 (ADC 2 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 ADC2 时钟
1：睡眠模式期间使能 ADC2 时钟

位 8 **ADC1LPEN**：睡眠模式期间的 ADC1 时钟使能 (ADC1 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 ADC1 时钟
1：睡眠模式期间使能 ADC1 时钟

位 7:6 保留，必须保持复位值。

位 5 **USART6LPEN**：睡眠模式期间的 USART6 时钟使能 (USART6 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 USART6 时钟
1：睡眠模式期间使能 USART6 时钟

位 4 **USART1LPEN**：睡眠模式期间的 USART1 时钟使能 (USART1 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 USART1 时钟
1：睡眠模式期间使能 USART1 时钟

位 3:2 保留，必须保持复位值。

位 1 **TIM8LPEN**：睡眠模式期间的 TIM8 时钟使能 (TIM8 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM8 时钟
1：睡眠模式期间使能 TIM8 时钟

位 0 **TIM1LPEN**：睡眠模式期间的 TIM1 时钟使能 (TIM1 clock enable during Sleep mode)
此位由软件置 1 和清零。
0：睡眠模式期间禁止 TIM1 时钟
1：睡眠模式期间使能 TIM1 时钟

5.3.20 RCC 备份域控制寄存器 (RCC_BDCR)

RCC backup domain control register

偏移地址：0x70

复位值：0x0000 0000，通过备份域复位进行复位。
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。

RCC 备份域控制寄存器 (RCC_BDCR) 中的 LSEON 位、LSEBYP 位、RTCSEL 位 和 RTCEN 位在备份域中。因而复位后，这些位受写保护，必须将 *PWR 电源控制寄存器 (PWR_CR1)* 中的 DBP 位置 1 才能修改这些位。有关详细信息，请参见*第 121 页上的第 5.1.1 节：系统复位*。只有备份域复位后，这些位才能复位（请参见*第 5.1.3 节：备份域复位*）。内部复位和外部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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BDRST
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCEN	Res.	Res.	Res.	Res.	Res.	RTCSEL[1:0]		Res.	Res.	Res.	LSEDRV[1:0]		LSEBYP	LSERDY	LSEON
rw						rw	rw				rw	rw	rw	r	rw

位 31:17 保留，必须保持复位值。

位 16 **BDRST**：备份域软件复位 (Backup domain software reset)
此位由软件置 1 和清零。
0：复位未激活
1：复位整个备份域
注： *BKPSRAM 不受此复位影响，只能在 Flash 保护级别从级别 1 更改为级别 0 时复位 BKPSRAM。*

位 15 **RTCEN**：RTC 时钟使能 (RTC clock enable)
此位由软件置 1 和清零。
0：禁止 RTC 时钟
1：使能 RTC 时钟

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)

这些位由软件置 1，用于选择 RTC 的时钟源。选择 RTC 时钟源后，除非备份域复位，否则不可再将其更改。可使用 BDRST 位对其进行复位。

00：无时钟
01：LSE 振荡器时钟用作 RTC 时钟
10：LSI 振荡器时钟用作 RTC 时钟
11：由可编程预分频器分频的 HSE 振荡器时钟（通过 RCC 时钟配置寄存器 (RCC_CFGR) 中的 RTCPRE[4:0] 位选择）用作 RTC 时钟

位 7:5 保留，必须保持复位值。

位 4:3 **LSEDRV[1:0]**：LSE 振荡器驱动能力 (LSE oscillator drive capability)

由软件置 1，用于调整 LSE 振荡器的驱动能力。

00：低驱动能力
01：中高驱动能力
10：中低驱动能力
11：高驱动能力

位 2 **LSEBYP**：外部低速振荡器旁路 (External low-speed oscillator bypass)

由软件置 1 和清零，用于旁路振荡器。只有在禁止 LSE 时钟后才能写入该位。

0：不旁路 LSE 振荡器
1：旁路 LSE 振荡器

位 1 **LSERDY**：外部低速振荡器就绪 (External low-speed oscillator ready)

此位由硬件置 1 和清零，用于指示外部 32 kHz 振荡器已稳定。在 LSEON 位被清零后，LSERDY 将在 6 个外部低速振荡器时钟周期后转为低电平。

0：LSE 时钟未就绪
1：LSE 时钟就绪

位 0 **LSEON**：外部低速振荡器使能 (External low-speed oscillator enable)

此位由软件置 1 和清零。

0：LSE 时钟关闭
1：LSE 时钟开启

5.3.21 RCC 时钟控制和状态寄存器 (RCC_CSR)

RCC clock control & status register

偏移地址：0x74

复位值：0x0E00 0000，除复位标志只能通过电源复位进行复位以外，其它通过系统复位进行复位。

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
LPWR RSTF	WWDG RSTF	IWDG RSTF	SFT RSTF	POR RSTF	PIN RSTF	BOR RSTF	RMVF	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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSIRDY	LSION
														r	rw

位 31 **LPWRRSTF**：低功耗复位标志 (Low-power reset flag)
发生低功耗管理复位时，由硬件置 1。
通过写入 RMVF 位清零。
0：未发生低功耗管理复位
1：发生低功耗管理复位
有关低功耗管理复位的详细信息，请参见*低功耗管理复位*。

位 30 **WWDGRSTF**：窗口看门狗复位标志 (Window watchdog reset flag)
发生窗口看门狗复位时，此位由硬件置 1。
通过写入 RMVF 位清零。
0：未发生窗口看门狗复位
1：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)
V_{DD} 域发生独立看门狗复位时，此位由硬件置 1。
通过写入 RMVF 位清零。
0：未发生看门狗复位
1：发生看门狗复位

位 28 **SFTRSTF**：软件复位标志 (Software reset flag)
发生软件复位时，此位由硬件置 1。
通过写入 RMVF 位清零。
0：未发生软件复位
1：发生软件复位

位 27 **PORRSTF**：POR/PDR 复位标志 (POR/PDR reset flag)
发生 POR/PDR 复位时，此位由硬件置 1。
通过写入 RMVF 位清零。
0：未发生 POR/PDR 复位
1：发生 POR/PDR 复位

位 26 **PINRSTF**：引脚复位标志 (PIN reset flag)
发生来自 NRST 引脚的复位时，此位由硬件置 1。
通过写入 RMVF 位清零。
0：未发生来自 NRST 引脚的复位
1：发生来自 NRST 引脚的复位

位 25 **BORRSTF**：BOR 复位标志 (BOR reset flag)
通过软件写入 RMVF 位清零。
发生 POR/PDR 复位或 BOR 复位时，此位由硬件置 1。
0：未发生 POR/PDR 复位或 BOR 复位
1：发生 POR/PDR 复位或 BOR 复位

位 24 **RMVF**：清除复位标志 (Remove reset flag)
此位由软件置 1，用于清除复位标志。
0：无影响
1：清除复位标志

位 23:2 保留，必须保持复位值。

位 1 **LSIRDY**：内部低速振荡器就绪 (Internal low-speed oscillator ready)
此位由硬件置 1 和清零，用于指示内部 RC 40 kHz 振荡器已稳定。在 LSION 位被清零后，LSIRDY 将在 3 个 LSI 时钟周期后转为低电平。
0：LSI RC 振荡器未就绪
1：LSI RC 振荡器就绪

位 0 **LSION**：内部低速振荡器使能 (Internal low-speed oscillator enable)
此位由软件置 1 和清零。
0：LSI RC 振荡器关闭
1：LSI RC 振荡器开启

5.3.22 RCC 扩频时钟生成寄存器 (RCC_SSCGR)

RCC spread spectrum clock generation register

偏移地址：0x80

复位值：0x0000 0000

访问：无等待周期，按字、半字和字节访问。

扩频时钟生成仅适用于主 PLL。

RCC_SSCGR 的写操作必须在使能主 PLL 之前或禁止主 PLL 之后进行。

注： *有关 PLL 扩频时钟生成 (SSCG) 特性的所有详细信息，请参见器件数据手册中的“电气特性”部分。*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SSCGEN	SPREADSEL	Res.	Res.	INCSTEP											
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
INCSTEP			MODPER												
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **SSCGEN**：扩频调制使能 (Spread spectrum modulation enable)
此位由软件置 1 和清零。
0：扩频调制禁止。（CR[24]=PLLON 位清零后写入）
1：扩频调制使能。（CR[24]=PLLON 位置 1 前写入）

位 30 **SPREADSEL**：扩频选择 (Spread Select)
此位由软件置 1 和清零。
CR[24]=PLLON 位置 1 前写入。
0：中心扩频
1：向下扩频

位 29:28 保留，必须保持复位值。

位[27:13] **INCSTEP**：增量步长 (Incrementation step)
这些位将由软件置 1 和清零。CR[24]=PLLON 位置 1 前写入。
设置调制曲线振幅输入。

位 12:0 **MODPER**：调制周期 (Modulation period)
这些位将由软件置 1 和清零。CR[24]=PLLON 位置 1 前写入。
设置调制曲线周期输入。

5.3.23 RCC PLLI2S 配置寄存器 (RCC_PLLI2SCFGR)

RCC PLLI2S configuration register

偏移地址：0x84

复位值：0x2400 3000

访问：无等待周期，按字、半字和字节访问。

此寄存器用于根据公式式配置 PLLI2S 时钟输出：

$f_{(VCO\ 时钟)} = f_{(PLLI2S\ 时钟输入)} \times (PLLI2SN / PLLM)$

$f_{(PLLI2S_P)} = f_{(VCO\ 时钟)} / PLLI2SP$

$f_{(PLLI2S_Q)} = f_{(VCO\ 时钟)} / PLLI2SQ$

$f_{(PLLI2S_R)} = f_{(VCO\ 时钟)} / PLLI2SR$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	PLLI2S[2:0]			PLLI2SQ[0:3]				Res.	Res.	Res.	Res.	Res.	Res.	PLLI2SP[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.	PLLI2SN[8:0]									Res.	Res.	Res.	Res.	Res.	Res.
	rw	rw	rw	rw	rw	rw	rw	rw	rw						

位 31 保留，必须保持复位值。

位 30:28 **PLLI2SR[2:0]**：适用于 I2S 时钟的 PLLI2S 分频系数 (PLLI2S division factor for I2S clocks)

这些位由软件置 1 和清零，用于控制 I2S 时钟频率。仅在 PLLI2S 已禁止时才能写入这些位。选择的系数必须与 I2S 外设中的预分频值一致，以达到使用标准晶振时误差不超过 0.3%，使用音频晶振时误差为 0%。有关 I2S 时钟频率和精度的详细信息，请参见 I2S 一章的*第 32.7.3 节：启动描述*。

注意： I2S 的正常工作频率要求低于或等于 192 MHz。

I2S 时钟频率 = VCO 频率 / PLLR，其中，2 ≤ PLLR ≤ 7

000：PLLR = 0，错误配置

001：PLLR = 1，错误配置

010：PLLR = 2

...

111：PLLR = 7

位 27:24 **PLLI2SQ[3:0]**：适用于 SAI 时钟的 PLLI2S 分频系数 (PLLI2S division factor for SAI clocks)

这些位由软件置 1 和清零，用于控制 SAI 时钟频率。

仅在 PLLI2S 已禁止时才能写入这些位。

SAI 时钟频率 = VCO 频率 / PLLI2SQ，其中，2 <= PLLI2SQ <= 15

0000：PLLI2SQ = 0，错误配置

0001：PLLI2SQ = 1，错误配置

0010：PLLI2SQ = 2

0011：PLLI2SQ = 3

0100：PLLI2SQ = 4

0101：PLLI2SQ = 5

...

1111：PLLI2SQ = 15

位 23:18 保留，必须保持复位值。

位 17:16 **PLLI2SP[1:0]**：适用于 SPDIFRX 时钟的 PLLI2S 分频系数 (PLLI2S division factor for SPDIFRX clock)

这些位由软件置 1 和清零，用于控制 SPDIFRX 时钟。这些位只能在 PLLI2S 已禁止时写入。选择的系数必须与 SPDIF 中的预分频值一致，从而根据 SPDIF 模式，使音频时钟频率接近 44 kHz 或 48 kHz。

SPDIF 时钟频率 = VCO 频率 / PLLI2SP，其中，PLLI2SP = 2、4、6 或 8

00：PLLI2SP = 2

01：PLLI2SP = 4

10：PLLI2SP = 6

11：PLLI2SP = 8

位 15 保留，必须保持复位值。

位 14:6 **PLLI2SN[8:0]**：适用于 VCO 的 PLLI2S 倍频系数 (PLLI2S multiplication factor for VCO)

这些位由软件置 1 和清零，用于控制 VCO 的倍频系数。这些位只能在 PLLI2S 已禁止时写入。写入这些位时只允许使用半字和字访问。

注意： 软件必须正确设置这些位，确保 VCO 输出频率介于 49 MHz 和 432 MHz 之间。

VCO 输出频率 = VCO 输入频率 × PLLI2SN，其中，49 ≤ PLLI2SN ≤ 432

000000000：PLLI2SN = 0，错误配置

000000001：PLLI2SN = 1，错误配置

...

000110000：PLLI2SN = 48，错误配置

000110001：PLLI2SN = 49

...

011000000：PLLI2SN = 192

011000001：PLLI2SN = 193

...

110110000：PLLI2SN = 432

110110000：PLLI2SN = 433，错误配置

...

111111111：PLLI2SN = 511，错误配置

位 5:0 保留，必须保持复位值。

5.3.24 RCC PLL 配置寄存器 (RCC_PLLSAICFGR)

RCC PLL configuration register

偏移地址：0x88

复位值：0x2400 3000

访问：无等待周期，按字、半字和字节访问。

此寄存器用于根据公式配置 PLLSAI 时钟输出：

- $f_{(VCO\ 时钟)} = f_{(PLLSAI\ 时钟输入)} \times (PLLSAIN / PLLM)$
- $f_{(PLLISAI_P)} = f_{(VCO\ 时钟)} / PLLSAIP$
- $f_{(PLLISAI_Q)} = f_{(VCO\ 时钟)} / PLLSAIQ$
- $f_{(PLLISAI_R)} = f_{(VCO\ 时钟)} / PLLSAIR$

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

位 31 保留，必须保持复位值。

位 30:28 **PLLSAIR[2:0]**：适用于 LCD 时钟的 PLLSAI 分频系数 (PLLSAI division factor for LCD clock)
由软件置 1 和复位，用于控制 LCD 时钟频率。
这些位应在 PLLSAI 已禁止时写入。
LCD 时钟频率 = VCO 频率 / PLLSAIR，其中，2 ≤ PLLSAIR ≤ 7
000：PLLSAIR = 0，错误配置
001：PLLSAIR = 1，错误配置
010：PLLSAIR = 2
...
111：PLLSAIR = 7

位 27:24 **PLLSAIQ[3:0]**：适用于 SAI 时钟的 PLLSAI 分频系数 (PLLSAI division factor for SAI clock)
由软件置 1 和复位，用于控制 SAI 时钟的频率。
这些位应在 PLLSAI 已禁止时写入。
SAI1 时钟频率 = VCO 频率 / PLLSAIQ，其中，2 ≤ PLLSAIQ ≤ 15
0000：PLLSAIQ = 0，错误配置
0001：PLLSAIQ = 1，错误配置
...
0010：PLLSAIQ = 2
0011：PLLSAIQ = 3
0100：PLLSAIQ = 4
0101：PLLSAIQ = 5
...
1111：PLLSAIQ = 15

位 23:18 保留，必须保持复位值。

位 17:16 **PLLSAIP[1:0]**：适用于 48 MHz 时钟的 PLLSAI 分频系数 (PLLSAI division factor for 48MHz clock)
由软件置 1 和清零，用于控制 PLLSAI 输出时钟 (PLLSAI48CLK) 的频率。可为 USB、RNG、SDMMC（48 MHz 时钟）选择此输出。仅在 PLLSAI 已禁止时才能写入这些位。
写入这些位时只允许使用半字和字访问。

PLLSAI48 输出时钟频率 = VCO 频率 / PLLSAIP，其中，PLLSAIP = 2、4、6 或 8
00：PLLSAIP = 2
01：PLLSAIP = 4
10：PLLSAIP = 6
11：PLLSAIP = 8

位 15 保留，必须保持复位值。

位 14:6 **PLLSAIN[8:0]**：适用于 VCO 的 PLLSAI 倍频系数 (PLLSAI division factor for VCO)
由软件置 1 和复位，用于控制 VCO 的倍频系数。
这些位应在 PLLSAI 已禁止时写入。
写入这些位时只允许使用半字和字访问。
VCO 输出频率 = VCO 输入频率 × PLLSAIN，其中，49 ≤ PLLSAIN ≤ 432
000000000：PLLSAIN = 0，错误配置
000000001：PLLSAIN = 1，错误配置
.....
000110000：PLLSAIN = 48，错误配置
000110001：PLLSAIN = 49
...
011000000：PLLSAIN = 192
011000001：PLLSAIN = 193
...
110110000：PLLSAIN = 432
110110000：PLLSAIN = 433，错误配置
...
111111111：PLLSAIN = 511，错误配置

位 5:0 保留，必须保持复位值

5.3.25 RCC 专用时钟配置寄存器 (RCC_DKCFGR1)

RCC dedicated clocks configuration register

偏移地址：0x8C

复位值：0x0000 0000

访问：无等待周期，按字、半字和字节访问。

此寄存器用于配置定时器时钟预分频器，并可根据以下公式配置 SAI 和 LTDC 外设的 PLLSAI、PLLI2S 输出时钟分频器：

$f_{(PLLSAIDIVQ\ 时钟输出)} = f_{(PLLSAI_Q)} / PLLSAIDIVQ$
$f_{(PLLSAIDIVR\ 时钟输出)} = f_{(PLLSAI_R)} / PLLSAIDIVR$
$f_{(PLLI2SDIVQ\ 时钟输出)} = f_{(PLLI2S_Q)} / PLLI2SDIVQ$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TIMPRE	SAI2SEL[1:0]		SAI1SEL[1:0]		Res.	Res.	PLLSAIDIVR[1:0]	
							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.	PLLSAIDIVQ[4:0]					Res.	Res.	Res.	PLLI2SDIVQ[4:0]				
			rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

位 31:25 保留，必须保持复位值。

位 24 **TIMPRE**：定时器时钟预分频器选择 (Timers clocks prescalers 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 APB 预分频器（RCC_CFGR 寄存器中的 PPRE1、PPRE2）的分频系数配置为 1，则 TIMxCLK = PCLKx。否则，定时器时钟频率将设置为与定时器相连的 APB 域的频率的两倍：TIMxCLK = 2xPCLKx。
1：如果 APB 预分频器（RCC_CFGR 寄存器中的 PPRE1、PPRE2）的分频系数配置为 1、2 或 4，则 TIMxCLK = HCLK。否则，定时器时钟频率将设置为与定时器相连的 APB 域的频率的四倍：
TIMxCLK = 4xPCLKx。

位 23:22 **SAI2SEL[1:0]**：SAI2 时钟源选择 (SAI2 clock source selection)：
这些位由软件置 1 和清零，用于控制 SAI2 时钟频率。
这些位应在 PLLSAI 和 PLLI2S 已禁止时写入。
00：SAI2 时钟频率 = f(PLLSAI_Q) / PLLSAIDIVQ
01：SAI2 时钟频率 = f(PLLI2S_Q) / PLLI2SDIVQ
10：SAI2 时钟频率 = 复用功能输入频率
11：错误配置

位 21:20 **SAI1SEL[1:0]**：SAI1 时钟源选择 (SAI1 clock source selection)：
这些位由软件置 1 和清零，用于控制 SAI1 时钟频率。
这些位应在 PLLSAI 和 PLLI2S 已禁止时写入。
00：SAI1 时钟频率 = f(PLLSAI_Q) / PLLSAIDIVQ
01：SAI1 时钟频率 = f(PLLI2S_Q) / PLLI2SDIVQ
10：SAI1 时钟频率 = 复用功能输入频率
11：错误配置

位 19:18 保留，必须保持复位值。

位 17:16 **PLLSAIDIVR[1:0]**：适用于 LCD_CLK 的分频系数 (division factor for LCD_CLK)
这些位由软件置 1 和清零，用于控制 LCD_CLK 的频率。
仅在 PLLSAI 已禁止时才能写入这些位。
LCD_CLK 频率 = f(PLLSAI_R) / PLLSAIDIVR，其中，2 ≤ PLLSAIDIVR ≤ 16
00：PLLSAIDIVR = 2
01：PLLSAIDIVR = 4
10：PLLSAIDIVR = 8
11：PLLSAIDIVR = 16

位 15:13 保留，必须保持复位值。

位 12:8 **PLLSAIDIVQ[4:0]**：适用于 SAI1 时钟的 PLLSAI 分频系数 (PLLSAI division factor for SAI1 clock)
这些位由软件置 1 和复位，用于控制 SAI1 时钟频率。
仅在 PLLSAI 已禁止时才能写入这些位。
SAI1 时钟频率 = f(PLLSAI_Q) / PLLSAIDIVQ，其中，1 ≤ PLLSAIDIVQ ≤ 32
00000：PLLSAIDIVQ = 1
00001：PLLSAIDIVQ = 2
00010：PLLSAIDIVQ = 3
00011：PLLSAIDIVQ = 4
00100：PLLSAIDIVQ = 5
...
11111：PLLSAIDIVQ = 32

位 7:5 保留，必须保持复位值。

位 4:0 **PLLI2SDIV[4:0]**：适用于 SAI1 时钟的 PLLI2S 分频系数 (PLLI2S division factor for SAI1 clocks)
这些位由软件置 1 和复位，用于控制 SAI1 时钟频率。
仅在 PLLI2S 已禁止时才能写入这些位。
SAI1 时钟频率 = f(PLLI2S_Q) / PLLI2SDIVQ，其中，1 <= PLLI2SDIVQ <= 31
00000：PLLI2SDIVQ = 1
00001：PLLI2SDIVQ = 2
00010：PLLI2SDIVQ = 3
00011：PLLI2SDIVQ = 4
00100：PLLI2SDIVQ = 5
...
11111：PLLI2SDIVQ = 32

5.3.26 RCC 专用时钟配置寄存器 (DCKCFGR)

RCC dedicated clocks configuration register

地址：0x90h

复位值：0x0000 0000h

访问：无等待周期，按字、半字和字节访问

该寄存器用于为 48 MHz、SDMMC、HDMI-CEC、LPTIM1、UART、USART 和 I2C 时钟选择时钟源。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	SDMMCSEL	CK48MSEL	CECSEL	LPTIM1SEL		I2C4SEL		I2C3SEL		I2C2SEL		I2C1SEL	
			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
UART8SEL		UART7SEL		USART6SEL		UART5SEL		UART4SEL		USART3SEL		USART2SEL		USART1SEL	
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 **SDMMCSEL**：SDMMC 时钟源选择 (SDMMC clock source selection)
由软件置 1 和复位。
0：选择 48 MHz 时钟作为 SDMMC 时钟
1：选择系统时钟作为 SDMMC 时钟

位 27 **CK48MSEL**：48 MHz 时钟源选择 (48MHz clock source selection)
由软件置 1 和复位。
0：选择 PLL 的 48 MHz 时钟
1：选择 PLLSAI 的 48 MHz 时钟

位 26 **CECSEL**：HDMI-CEC 时钟源选择 (HDMI-CEC clock source selection)
由软件置 1 和复位。
0：选择 LSE 时钟作为 HDMI-CEC 时钟
1：选择 HSI 时钟 488 分频作为 HDMI-CEC 时钟

位 25:24 **LPTIM1SEL**：低功耗定时器 1 时钟源选择
由软件置 1 和复位。
00：选择 APB1 时钟 (PCLK1) 作为 LPTIM1 时钟
01：选择 LSI 时钟作为 LPTIM1 时钟
10：选择 HSI 时钟作为 LPTIM1 时钟
11：选择 LSE 时钟作为 LPTIM1 时钟

位 23:22 **I2C4SEL**：I2C4 时钟源选择 (I2C4 clock source selection)
由软件置 1 和复位。
00：选择 APB1 时钟 (PCLK1) 作为 I2C4 时钟
01：选择系统时钟作为 I2C4 时钟
10：选择 HSI 时钟作为 I2C4 时钟
11：保留

位 21:20 **I2C3SEL**：I2C3 时钟源选择 (I2C3 clock source selection)
由软件置 1 和复位。
00：选择 APB 时钟作为 I2C3 时钟
01：选择系统时钟作为 I2C3 时钟
10：选择 HSI 时钟作为 I2C3 时钟
11：保留

位 19:18 **I2C2SEL**：I2C2 时钟源选择 (I2C2 clock source selection)
由软件置 1 和复位。
00：选择 APB1 时钟 (PCLK1) 作为 I2C2 时钟
01：选择系统时钟作为 I2C2 时钟
10：选择 HSI 时钟作为 I2C2 时钟
11：保留

位 17:16 **I2C1SEL**：I2C1 时钟源选择 (I2C1 clock source selection)
由软件置 1 和复位。
00：选择 APB 时钟 (PCLK1) 作为 I2C1 时钟
01：选择系统时钟作为 I2C1 时钟
10：选择 HSI 时钟作为 I2C1 时钟
11：保留

位 15:14 **UART8SEL[1:0]**：UART 8 时钟源选择 (UART 8 clock source selection)
由软件置 1 和复位。
00：选择 APB1 时钟 (PCLK1) 作为 UART 8 时钟
01：选择系统时钟作为 UART 8 时钟
10：选择 HSI 时钟作为 UART 8 时钟
11：选择 LSE 时钟作为 UART 8 时钟

位 13:12 **UART7SEL[1:0]**：UART 7 时钟源选择 (UART 7 clock source selection)
由软件置 1 和复位。
00：选择 APB1 时钟 (PCLK1) 作为 UART 7 时钟
01：选择系统时钟作为 UART 7 时钟
10：选择 HSI 时钟作为 UART 7 时钟
11：选择 LSE 时钟作为 UART 7 时钟

位 11:10 **USART6SEL[1:0]**：USART 6 时钟源选择 (USART 6 clock source selection)
由软件置 1 和复位。
00：选择 APB2 时钟 (PCLK2) 作为 USART 6 时钟
01：选择系统时钟作为 USART 6 时钟
10：选择 HSI 时钟作为 USART 6 时钟
11：选择 LSE 时钟作为 USART 6 时钟

位 9:8 **UART5SEL[1:0]**：UART 5 时钟源选择 (UART 5 clock source selection)
由软件置 1 和复位。
00：选择 APB1 时钟 (PCLK1) 作为 UART 5 时钟
01：选择系统时钟作为 UART 5 时钟
10：选择 HSI 时钟作为 UART 5 时钟
11：选择 LSE 时钟作为 UART 5 时钟

位 7:6 **UART4SEL[1:0]**：UART 4 时钟源选择 (UART 4 clock source selection)
由软件置 1 和复位。
00：选择 APB1 时钟 (PCLK1) 作为 UART 4 时钟
01：选择系统时钟作为 UART 4 时钟
10：选择 HSI 时钟作为 UART 4 时钟
11：选择 LSE 时钟作为 UART 4 时钟

位 5:4 **USART3SEL[1:0]**：USART 3 时钟源选择 (USART 3 clock source selection)
由软件置 1 和复位。
00：选择 APB1 时钟 (PCLK1) 作为 USART 3 时钟
01：选择系统时钟作为 USART 3 时钟
10：选择 HSI 时钟作为 USART 3 时钟
11：选择 LSE 时钟作为 USART 3 时钟

位 3:2 **USART2SEL[1:0]**：USART 2 时钟源选择 (USART 2 clock source selection)
由软件置 1 和复位。
00：选择 APB1 时钟 (PCLK1) 作为 USART 2 时钟
01：选择系统时钟作为 USART 2 时钟
10：选择 HSI 时钟作为 USART 2 时钟
11：选择 LSE 时钟作为 USART 2 时钟

位 1:0 **USART1SEL[1:0]**：USART 1 时钟源选择 (USART 1 clock source selection)
由软件置 1 和复位。
00：选择 APB2 时钟 (PCLK2) 作为 USART 1 时钟
01：选择系统时钟作为 USART 1 时钟
10：选择 HSI 时钟作为 USART 1 时钟
11：选择 LSE 时钟作为 USART 1 时钟

5.3.27 RCC 寄存器映射

表 20 给出了寄存器映射和复位值。

表 20. RCC 寄存器映射和复位值

地址偏移	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	RCC_CR	Res.	Res.	PLL SAIRDY	PLL SAION	PLL I2SRDY	PLL I2SON	PLL RDY	PLL ON	Res.	Res.	Res.	Res.	CSSON	HSEBYP	HSERDY	HSEON	HSICAL 7	HSICAL 6	HSICAL 5	HSICAL 4	HSICAL 3	HSICAL 2	HSICAL 1	HSICAL 0	HSITRIM 4	HSITRIM 3	HSITRIM 2	HSITRIM 1	HSITRIM 0	保留	HSIRDY	HSION
0x04	RCC_PLLCFGR	Res.	Res.	Res.	Res.	PLLQ 3	PLLQ 2	PLLQ 1	PLLQ 0	Res.	PLLSRC	Res.	Res.	Res.	Res.	PLLP 1	PLLP 0	Res.	PLLN 8	PLLN 7	PLLN 6	PLLN 5	PLLN 4	PLLN 3	PLLN 2	PLLN 1	PLLN 0	PLLM 5	PLLM 4	PLLM 3	PLLM 2	PLLM 1	PLLM 0

表 20. RCC 寄存器映射和复位值（续）

地址偏移	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
0x08	RCC_CFGR	MCO2 1	MCO2 0	MCO2PRE2	MCO2PRE1	MCO2PRE0	MCO1PRE2	MCO1PRE1	MCO1PRE0	I2SSRC	MCO1 1	MCO1 0	RTCPRE 4	RTCPRE 3	RTCPRE 2	RTCPRE 1	RTCPRE 0	PPRE2 2	PPRE2 1	PPRE2 0	PPRE1 2	PPRE1 1	PPRE1 0	Res.	Res.	HPRE 3	HPRE 2	HPRE 1	HPRE 0	SWS 1	SWS 0	SW 1	SW 0
0x0C	RCC_CIR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSSC	PLLSAIRDYC	PLLI2SRDYC	PLLRDYC	HSERDYC	HSIRDYC	LSERDYC	LSIRDYC	Res.	PLLSAIRDYIE	PLLI2SRDYIE	PLLRDYIE	HSERDYIE	HSIRDYIE	LSERDYIE	LSIRDYIE	CSSF	PLLSAIRDYF	PLLI2SRDYF	PLLRDYF	HSERDYF	HSIRDYF	LSERDYF	LSIRDYF
0x10	RCC_AHB1RSTR	Res.	Res.	Res.	OTGHSRST	Res.	Res.	ETHMACRST	Res.	DMA2DRST	DMA2RST	DMA1RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRCRST	Res.	GPIOKRST	GPIOJRST	GPIOIRST	GPIOHRST	GPIOGRST	GPIOFRST	GPIOERST	GPIODRST	GPIOCRST	GPIOBRST	GPIOARST
0x14	RCC_AHB2RSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTGFSRST	RNGRST	HSAHRST	CRYPRST	Res.	Res.	Res.	DCMIRST
0x18	RCC_AHB3RSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	QSPIRST	FMCRST
0x1C	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x20	RCC_APB1RSTR	UART8RST	UART7RST	DACRST	PWRRST	CECRST	CAN2RST	CAN1RST	I2C4RST	I2C3RST	I2C2RST	I2C1RST	UART5RST	UART4RST	UART3RST	UART2RST	SPDIFRXRST	SPI3RST	SPI2RST	Res.	Res.	WWDGRST	Res.	LPTIM1RST	TIM14RST	TIM13RST	TIM12RST	TIM7RST	TIM6RST	TIM5RST	TIM4RST	TIM3RST	TIM2RST
0x24	RCC_APB2RSTR	Res.	Res.	Res.	Res.	Res.	LTDCRST	Res.	Res.	SAI2RST	SAI1RST	SPI6RST	SPI5RST	Res.	TIM11RST	TIM10RST	TIM9RST	Res.	SYSCFGRST	SP45RST	SPI1RST	SDMMC1RST	Res.	Res.	ADCRST	Res.	Res.	USART6RST	USART1RST	Res.	Res.	TIM8RST	TIM1RST
0x28	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x2C	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x30	RCC_AHB1ENR	Res.	OTGHSULPIEN	OTGHSEN	ETHMACPTPEN	ETHMACRXEN	ETHMACTXEN	ETHMACEN	Res.	DMA2DEN	DMA2EN	DMA1EN	DTCMRAMEN	保留	BKPSRAMEN	Res.	Res.	Res.	Res.	Res.	CRCEN	Res.	GPIOKEN	GPIOJEN	GPIOIEN	GPIOHEN	GPIOGEN	GPIOFEN	GPIOEEN	GPIODEN	GPIOCEN	GPIOBEN	GPIOAEN
0x34	RCC_AHB2ENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTGFSEN	RNGEN	HASHEN	CRYPEN	Res.	Res.	Res.	DCMIEN
0x38	RCC_AHB3ENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	QSPIEN	FMCEN
0x3C	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x40	RCC_APB1ENR	SPI8EN	SPI7EN	DACEN	PWREN	CECEN	CAN2EN	CAN1EN	I2C4EN	I2C3EN	I2C2EN	I2C1EN	UART5EN	UART4EN	USART3EN	USART2EN	SPDIFRXEN	SPI3EN	SPI2EN	Res.	SPI1EN	WWDGEN	Res.	LPTIM1EN	TIM14EN	TIM13EN	TIM12EN	TIM7EN	TIM6EN	TIM5EN	TIM4EN	TIM3EN	TIM2EN
0x44	RCC_APB2ENR	Res.	Res.	Res.	Res.	Res.	LTDCEN	Res.	Res.	SAI2EN	SAI1EN	SPI6EN	SPI5EN	Res.	TIM11EN	TIM10EN	TIM9EN	Res.	SYSCFGEN	SPI4EN	SPI1EN	SDMMC1EN	ADC3EN	ADC2EN	ADC1EN	Res.	Res.	USART6EN	USART1EN	Res.	Res.	TIM8EN	TIM1EN

表 20. RCC 寄存器映射和复位值（续）

地址偏移	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
0x48	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x4C	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x50	RCC_AHB1LPENR	Res.	OTGHSULPILPEN	OTGHSLPEN	ETHMACPTPLPEN	ETHMACRXLPEN	ETHMACTXLPEN	ETHMACLPEN	Res.	DMA2DLPEN	DMA2LPEN	DMA1LPEN	DTCMLPEN	Res.	BKPSRAMLPEN	SRAM2LPEN	SRAM1LPEN	FLITFLPEN	Res.	AXILPEN	CRCLPEN	Res.	GPIOKLPEN	GPIOJLPEN	GPIOILPEN	GPIOHLPEN	GPIOGLPEN	GPIOFLPEN	GPIOELPEN	GPIODLPEN	GPIOCLPEN	GPIOBLPEN	GPIOALPEN
0x54	RCC_AHB2LPENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTGFSLPEN	RNGLPEN	HASHLPEN	CRYPLPEN	Res.	Res.	Res.	DCMILPEN
0x58	RCC_AHB3LPENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	QSPILPEN	FMCLPEN
0x5C	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x60	RCC_APB1LPENR	UART8LPEN	UART7LPEN	DACLPEN	PWRLPEN	CECLPEN	CAN2LPEN	CAN1LPEN	I2C4LPEN	I2C3LPEN	I2C2LPEN	I2C1LPEN	UART5LPEN	UART4LPEN	USART3LPEN	USART2LPEN	SPDIFRXLPEN	SPI3LPEN	SPI2LPEN	Res.	Res.	WWDGLPEN	Res.	LPTIM1LPEN	TIM14LPEN	TIM13LPEN	TIM12LPEN	TIM7LPEN	TIM6LPEN	TIM5LPEN	TIM4LPEN	TIM3LPEN	TIM2LPEN
0x64	RCC_APB2LPENR	Res.	Res.	Res.	Res.	Res.	LTDCLPEN	Res.	Res.	SAI2LPEN	SAI1LPEN	SPI6LPEN	SPI5LPEN	Res.	TIM11LPEN	TIM10LPEN	TIM9LPEN	Res.	SYSCFGLPEN	SPI4LPEN	SPI1LPEN	SDMMC1LPEN	ADC3LPEN	ADC2LPEN	ADC1LPEN	Res.	Res.	USART6LPEN	USART1LPEN	Res.	Res.	TIM8LPEN	TIM1LPEN
0x68	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x6C	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x70	RCC_BDCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BDRST	RTCEN	Res.	Res.	Res.	Res.	Res.	RTCSEL 1	RTCSEL 0	Res.	Res.	Res.	LSEDRV[1:0]	Res.	LSEBYP	LSERDY	LSEON
0x74	RCC_CSR	LPWRRSTF	WWDGRSTF	WDGRSTF	SFTRSTF	PORRSTF	PADRSTF	BORRSTF	RMVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSIRDY	LSION
0x78	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x7C	保留	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x80	RCC_SSCGR	SSCGEN	SPREADSEL	Res.	Res.	INCSTEP															MODPER												
0x84	RCC_PLLI2SCFGR	Res.	PLLI2SR[2:0]			PLLI2SQ[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	PLLI2SP[1:0]		Res.	PLLI2SN[8:0]							Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

表 20. RCC 寄存器映射和复位值（续）

地址偏移	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
0x88	RCC_PLLSAICFGR	Res.	PLLSAIR[2:0]			PLLSAIQ[4:0]				Res.	Res.	Res.	Res.	Res.	Res.	PLLSAIP[1:0]		Res.	PLLSAIN[8:0]									Res.	Res.	Res.	Res.	Res.	Res.
0x8C	RCC_DCKCFGR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TIMPRE	SAI2SEL[1:0]		SAI1SEL[1:0]		Res.	Res.	PLLSAIDIVR[1:0]		Res.	Res.	Res.	PLLSAIDIVQ[4:0]				Res.	Res.	Res.	PLLI2SDIVQ[4:0]					
0x90	RCC_DCKCFGR2	Res.	Res.	Res.	SDMMCSEL	CK48MSEL	CECSEL	LPTIM1SEL		I2C4SEL		I2C3SEL		I2C2SEL		I2C1SEL		UART8SEL		UART7SEL		UART6SEL		UART5SEL		UART4SEL		UART3SEL		UART2SEL		UART1SEL	

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

6 通用 I/O (GPIO)

6.1 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）和 1 个 32 位置位/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) 和 2 个 32 位复用功能选择寄存器（GPIOx_AFRH 和 GPIOx_AFRL）。

6.2 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）
- 置位和复位寄存器 (GPIOx_BSRR)，对 GPIOx_ODR 具有按位写权限
- 锁定机制 (GPIOx_LCKR)，可冻结 I/O 端口配置
- 模拟功能
- 复用功能选择寄存器
- 快速翻转，每次翻转最快只需要两个时钟周期
- 引脚复用非常灵活，允许将 I/O 引脚用作 GPIO 或多种外设功能中的一种

6.3 GPIO 功能描述

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

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

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

图 16 和*图 17* 分别显示了标准和 5 V 容限 I/O 端口位的基本结构。*表 21* 给出了可能的端口位配置方案。

图 16. I/O 端口位的基本结构

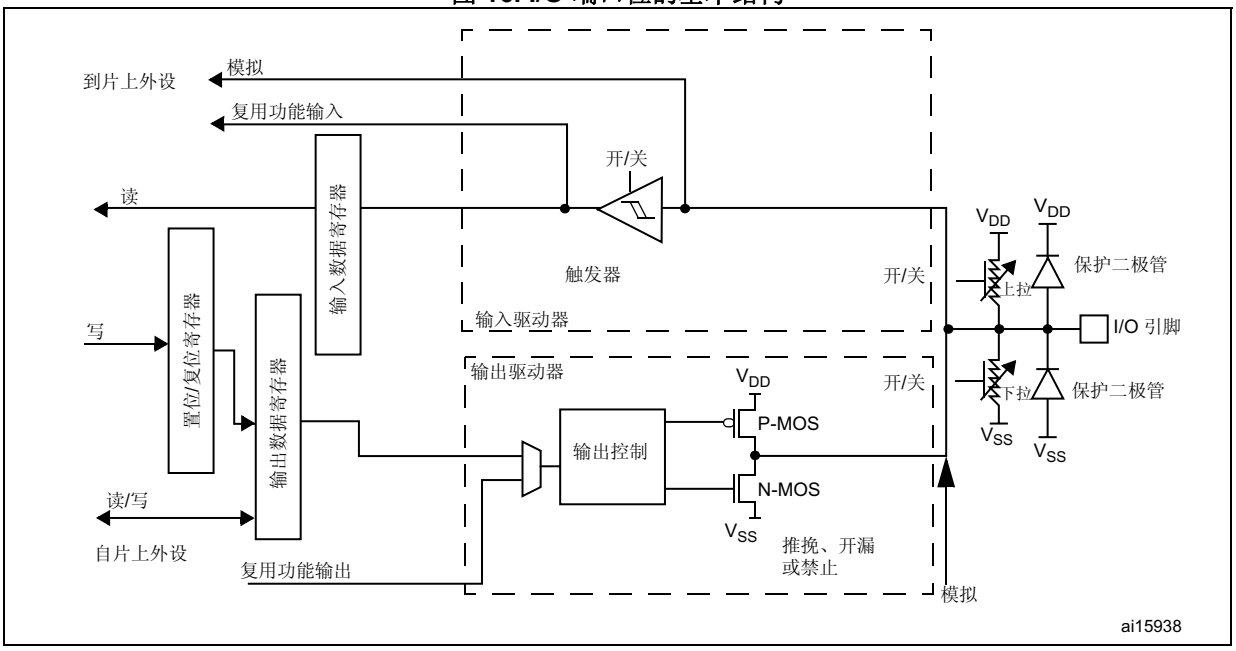
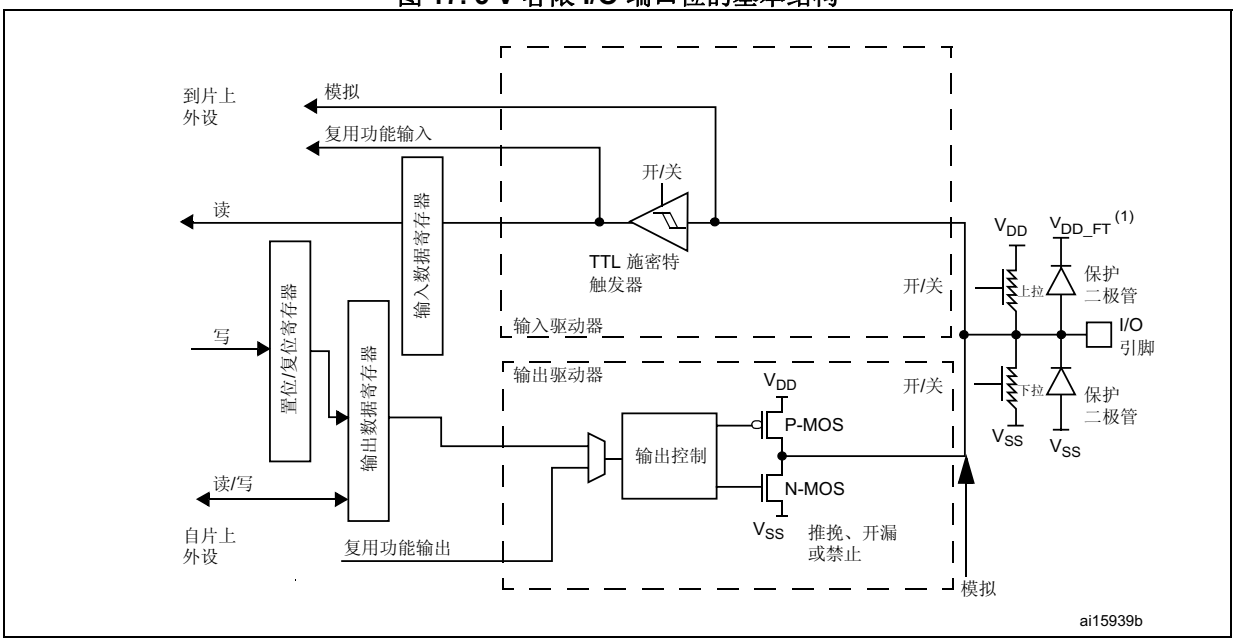


图 17. 5 V 容限 I/O 端口位的基本结构



1. V_{DD_FT} 是和 5 V 容限 I/O 相关的电位，与 V_{DD} 不同。

表 21. 端口位配置表(1)

MODE(i) [1:0]	OTYPER(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）	
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。

6.3.1 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 端口被配置为输入浮空模式。

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/SWDAT 处于上拉状态
- PB4：NJTRST 处于上拉状态
- PB3：JTDO 处于浮空状态

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

输入数据寄存器 (GPIOx_IDR) 每隔 1 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。

6.3.2 I/O 引脚复用功能复用器和映射

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

每个 I/O 引脚都有一个复用器，该复用器采用多达 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。
- Cortex-M7 FPU EVENTOUT 映射到 AF15

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。

要将 I/O 配制成所需功能，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
- **系统功能：**必须将 MCOx 引脚配置为复用功能模式。
- **GPIO：**在 GPIOx_MODER 寄存器中将所需 I/O 配置为输出、输入或模拟通道。
- **外设复用功能：**
  - 在 GPIOx_AFRL 或 GPIOx_AFRH 寄存器中，将 I/O 连接到所需的 AFx。
  - 通过 GPIOx_OTYPER、GPIOx_PUPDR 和 GPIOx_OSPEEDER 寄存器，分别选择类型、上拉/下拉以及输出速度。
  - 在 GPIOx_MODER 寄存器中将所需 I/O 配置为复用功能。

- **其它功能：**
  - 对于 ADC 和 DAC，在 GPIOx_MODER 寄存器中将所需 I/O 配置为模拟模式，并在 ADC 和 DAC 寄存器中配置所需功能。
  - 对于 RTC_OUT、RTC_TS、RTC_TAMPx、WKUPx 和振荡器的其它功能，在相关的 RTC、PWR 和 RCC 寄存器中配置所需功能。这些功能优先于标准 GPIO 寄存器中的配置。有关 RTC 的 I/O 控制的详细信息，请参见*第 808 页上的第 29.3 节：RTC 功能说明*。
- EVENTOUT
  - 配置用于输出 Cortex®-M7 FPU EVENTOUT 信号的 I/O 引脚（通过将其连接到 AF15）。

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。

6.3.3 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。无论采用哪种 I/O 方向，GPIOx_PUPDR 寄存器都用于选择上拉/下拉。

6.3.4 I/O 端口数据寄存器

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

有关寄存器说明的详细信息，请参见*第 6.4.5 节：GPIO 端口输入数据寄存器 (GPIOx_IDR) (x = A..K)*和*第 6.4.6 节：GPIO 端口输出数据寄存器 (GPIOx_ODR) (x = A..K)*。

6.3.5 I/O 数据位操作

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

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

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

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

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

6.3.6 GPIO 锁定机制

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

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

LOCK 序列（参见*第 6.4.8 节：GPIO 端口配置锁定寄存器 (GPIOx_LCKR) (x = A..K)*）只能通过对 GPIOx_LCKR 寄存器进行字（32 位长）访问的方式来执行，因为 GPIOx_LCKR 的第 16 位必须与 [15:0] 位同时置位。

有关详细信息，请参见*第 6.4.8 节：GPIO 端口配置锁定寄存器 (GPIOx_LCKR) (x = A..K)* 中的 LCKR 寄存器说明。

6.3.7 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，可根据应用程序的要求将某个复用功能连接到其它某个引脚。

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

要了解在每个 GPIO 引脚上复用了哪些功能，请参见器件数据手册。

6.3.8 外部中断线/唤醒线

所有端口都具有外部中断功能。要使用外部中断线，必须将端口配置为输入模式。请参见*第 11 节：外部中断和事件控制器 (EXTI)* 和*第 11.3 节：唤醒事件管理*。

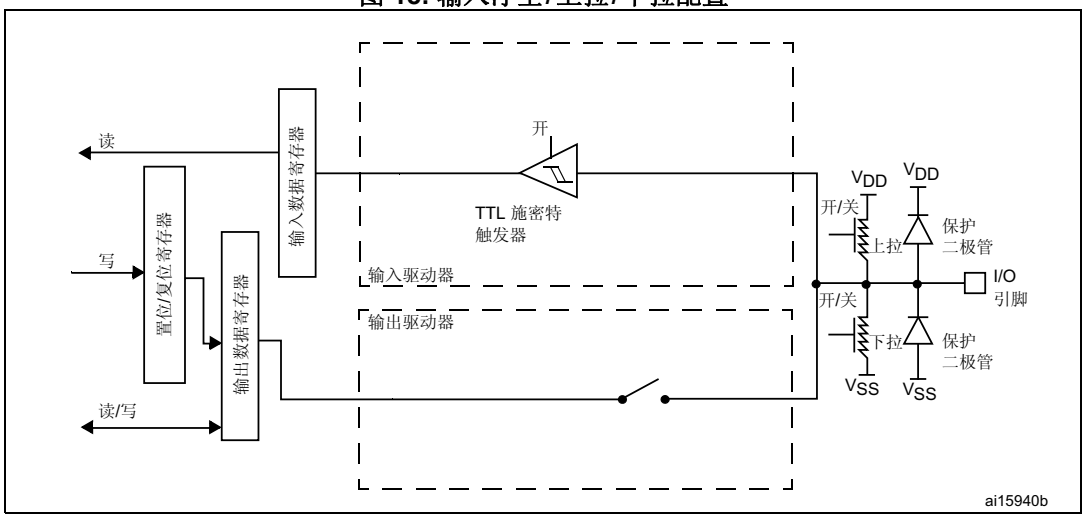
6.3.9 输入配置

对 I/O 端口进行编程作为输入时：

- 输出缓冲器被禁止
- 施密特触发器输入被打开
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开上拉和下拉电阻
- 输入数据寄存器每隔 1 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

图 18 说明了 I/O 端口位的输入配置。

图 18. 输入浮空/上拉/下拉配置



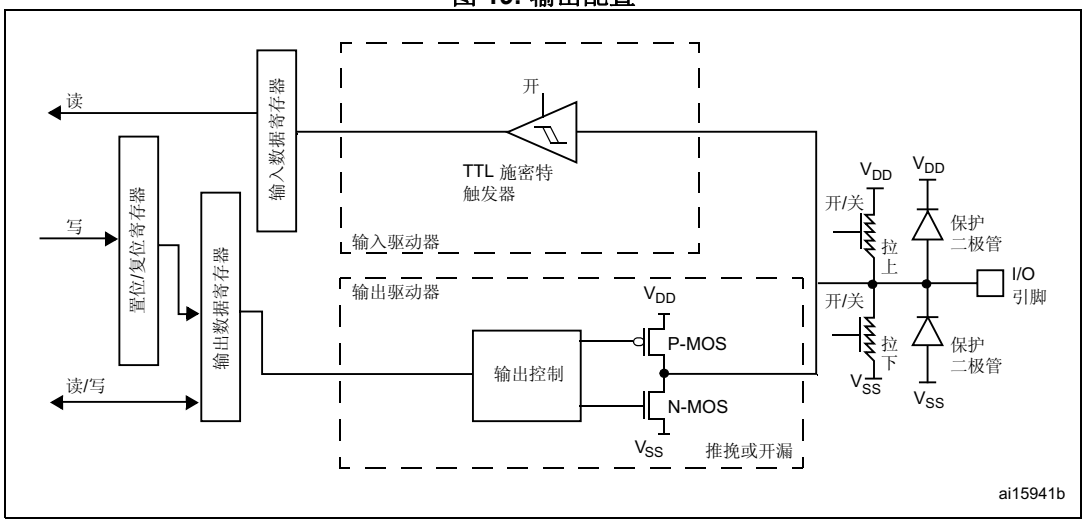
6.3.10 输出配置

对 I/O 端口进行编程作为输出时：

- 输出缓冲器被打开：
  - 开漏模式：输出寄存器中的“0”可激活 N-MOS，而输出寄存器中的“1”会使端口保持高阻态 (Hi-Z)（P-MOS 始终不激活）
  - 推挽模式：输出寄存器中的“0”可激活 N-MOS，而输出寄存器中的“1”可激活 P-MOS
- 施密特触发器输入被打开
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开上拉和下拉电阻
- 输入数据寄存器每隔 1 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

图 19 说明了 I/O 端口位的输出配置。

图 19. 输出配置



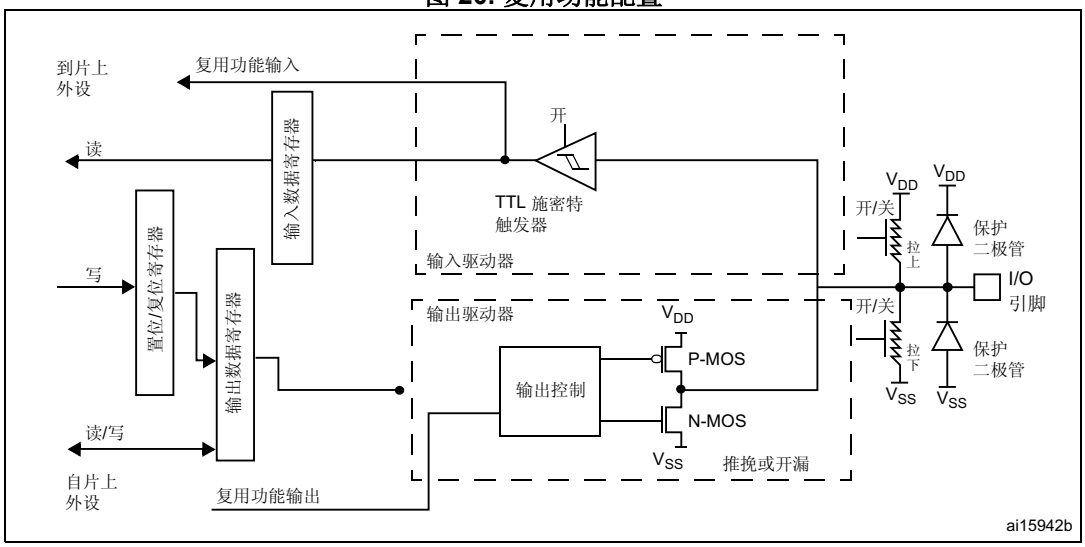
6.3.11 复用功能配置

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）
- 施密特触发器输入被打开
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开弱上拉电阻和下拉电阻
- 输入数据寄存器每隔 1 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

图 20 说明了 I/O 端口位的复用功能配置。

图 20. 复用功能配置



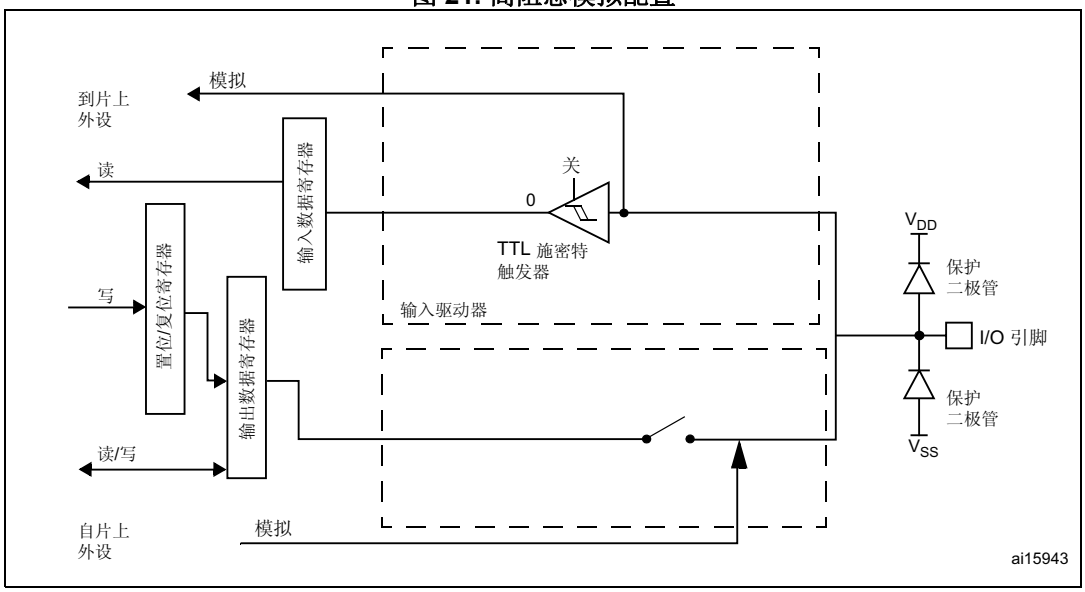
6.3.12 模拟配置

对 I/O 端口进行编程作为模拟配置时：

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

图 21 说明了 I/O 端口位的高阻态模拟输入配置。

图 21. 高阻态模拟配置



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

将振荡器配置为用户外部时钟模式时，仅为时钟输入保留 OSC_IN 或 OSC32_IN 引脚，OSC_OUT 或 OSC32_OUT 引脚仍可用作常规 GPIO。

6.3.14 在备份电源域中使用 GPIO 引脚

当内核电源域掉电时（器件进入待机模式时），PC13/PC14/PC15/PI8 GPIO 功能会丢失。在这种情况下，如果不通过 RTC 配置旁路其 GPIO 配置，这些引脚将被设置为模拟输入模式。

第 808 页上的第 29.3 节：RTC 功能说明。

6.4 GPIO 寄存器

本节对 GPIO 寄存器进行了详细介绍。

有关寄存器位、寄存器偏移地址和复位值的汇总，请参见*表 22*。

可按字、半字或字节模式写入外设寄存器。

6.4.1 GPIO 端口模式寄存器 (GPIOx_MODER) (x =A..K)

GPIO port mode register

偏移地址：0x00

复位值：

- 0xA800 0000（端口 A）
- 0x0000 0280（端口 B）
- 0x0000 0000（其它端口）

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

位 2y+1:2y **MODERy[1:0]**：端口 x 配置位 (Port x configuration bits) (y = 0..15)
这些位通过软件写入，用于配置 I/O 模式。
00：输入模式（复位状态）
01：通用输出模式
10：复用功能模式
11：模拟模式

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

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 配置位 (Port x configuration bits) (y = 0..15)
这些位通过软件写入，用于配置 I/O 输出类型。
0：推挽输出（复位状态）
1：开漏输出

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

GPIO port output speed register

偏移地址：0x08

复位值：

- 0x0C00 0000（端口 A）
- 0x0000 00C0（端口 B）
- 0x0000 0000（其它端口）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSPEEDR15 [1:0]		OSPEEDR14 [1:0]		OSPEEDR13 [1:0]		OSPEEDR12 [1:0]		OSPEEDR11 [1:0]		OSPEEDR10 [1:0]		OSPEEDR9 [1:0]		OSPEEDR8 [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
OSPEEDR7 [1:0]		OSPEEDR6 [1:0]		OSPEEDR5 [1:0]		OSPEEDR4 [1:0]		OSPEEDR3 [1:0]		OSPEEDR2 [1:0]		OSPEEDR1 [1:0]		OSPEEDR0 [1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 2y+1:2y **OSPEEDRy[1:0]**：端口 x 配置位 (Port x configuration bits) (y = 0..15)
这些位通过软件写入，用于配置 I/O 输出速度。
00：低速
01：中速
10：快速
11：高速

注： 有关 OSPEEDRy 位以及 V_{DD} 范围和外部负载的值，请参见产品数据手册。

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

GPIO port pull-up/pull-down register

偏移地址：0x0C

复位值：

- 0x6400 0000（端口 A）
- 0x0000 0100（端口 B）
- 0x0000 0000（其它端口）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]		PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[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
PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]		PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 2y+1:2y **PUPDRy[1:0]**：端口 x 配置位 (Port x configuration bits) (y = 0..15)
这些位通过软件写入，用于配置 I/O 上拉或下拉。
00：无上拉或下拉
01：上拉
10：下拉
11：保留

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

GPIO port input data register

偏移地址：0x10

复位值：0x0000 XXXX（其中 X 表示未定义）

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
IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
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 **IDRy**：端口输入数据 (Port input data) (y = 0..15)
这些位为只读。它们包含相应 I/O 端口的输入值。

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

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
ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
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 **ODRy**：端口输出数据 (Port output data) (y = 0..15)
这些位可通过软件读取和写入。
注： 对于原子置位/复位，通过写入 GPIOx_BSRR 或 GPIOx_BRR 寄存器，可分别置位和/或复位 ODR 位 (x = A..F)。

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

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 复位位 y (Port x reset bit y) (y = 0..15)
这些位为只写。读取这些位可返回值 0x0000。
0：不会对相应的 ODRx 位执行任何操作
1：复位相应的 ODRx 位
注： 如果同时对 BSx 和 BRx 置位，则 BSx 的优先级更高。

位 15:0 **BSy**：端口 x 置位位 y (Port x set bit y) (y= 0..15)
这些位为只写。读取这些位可返回值 0x0000。
0：不会对相应的 ODRx 位执行任何操作
1：置位相应的 ODRx 位

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

GPIO port configuration lock register

当正确的写序列应用到第 16 位 (LCKK) 时，此寄存器将用于锁定端口位的配置。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： *可使用特定的写序列对 GPIOx_LCKR 寄存器执行写操作。在此锁定序列期间只允许使用字访问（32 位长）。*

每个锁定位冻结一个特定的配置寄存器（控制寄存器和复用功能寄存器）。

偏移地址：0x1C

复位值：0x0000 0000

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

可随时读取此位。可使用锁定键写序列对其进行修改。

0：端口配置锁定键未激活

1：端口配置锁定键已激活。在下一次 MCU 复位或外设复位之前，GPIOx_LCKR 寄存器始终处于锁定状态。

锁定键写序列：

WR LCKR[16] = ‘1’ + LCKR[15:0]

WR LCKR[16] = ‘0’ + LCKR[15:0]

WR LCKR[16] = ‘1’ + LCKR[15:0]

RD LCKR

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

注： *在锁定键写序列期间，不能更改 LCK[15:0] 的值。*

锁定序列中的任何错误都将中止锁定操作。

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

位 15:0 **LCKy**：端口 x 锁定位 y (Port x lock bit y) (y= 0..15)

这些位都是读/写位，但只能在 LCKK 位等于“0”时执行写操作。

0：端口配置未锁定

1：端口配置已锁定

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

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
AFR7[3:0]				AFR6[3:0]				AFR5[3:0]				AFR4[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
AFR3[3:0]				AFR2[3:0]				AFR1[3:0]				AFR0[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **AFRy[3:0]**：端口 x 引脚 y 的复用功能选择 (Alternate function selection for port x pin y) (y = 0..7)

这些位通过软件写入，用于配置复用功能 I/O。

AFSELy 选择：

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

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

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
AFR15[3:0]				AFR14[3:0]				AFR13[3:0]				AFR12[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
AFR11[3:0]				AFR10[3:0]				AFR9[3:0]				AFR8[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **AFRy[3:0]**：端口 x 引脚 y 的复用功能选择 (Alternate function selection for port x pin y) (y = 8..15)

这些位通过软件写入，用于配置复用功能 I/O。

AFSELy 选择：

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

6.4.11 GPIO 寄存器映射

下表提供了 GPIO 寄存器映射和复位值。

表 22. 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	**GPIOA_MODER**	MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]		MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
	Reset value	1	0	1	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
0x00	**GPIOB_MODER**	MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]		MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
	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	0	1	0	0	0	0	0	0	0
0x00	**GPIOx_MODER** (where x = **C..K**)	MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]		MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
	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
0x04	**GPIOx_OTYPER** (where x = **A..K**)	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
0x08	**GPIOA_OSPEEDR**	OSPEEDR15[1:0]		OSPEEDR14[1:0]		OSPEEDR13[1:0]		OSPEEDR12[1:0]		OSPEEDR11[1:0]		OSPEEDR10[1:0]		OSPEEDR9[1:0]		OSPEEDR8[1:0]		OSPEEDR7[1:0]		OSPEEDR6[1:0]		OSPEEDR5[1:0]		OSPEEDR4[1:0]		OSPEEDR3[1:0]		OSPEEDR2[1:0]		OSPEEDR1[1:0]		OSPEEDR0[1:0]	
	Reset value	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
0x08	**GPIOB_OSPEEDR**	OSPEEDR15[1:0]		OSPEEDR14[1:0]		OSPEEDR13[1:0]		OSPEEDR12[1:0]		OSPEEDR11[1:0]		OSPEEDR10[1:0]		OSPEEDR9[1:0]		OSPEEDR8[1:0]		OSPEEDR7[1:0]		OSPEEDR6[1:0]		OSPEEDR5[1:0]		OSPEEDR4[1:0]		OSPEEDR3[1:0]		OSPEEDR2[1:0]		OSPEEDR1[1:0]		OSPEEDR0[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	1	1	0	0	0	0	0	0

表 22. 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** (where x = **C..K**)	OSPEEDR15[1:0]		OSPEEDR14[1:0]		OSPEEDR13[1:0]		OSPEEDR12[1:0]		OSPEEDR11[1:0]		OSPEEDR10[1:0]		OSPEEDR9[1:0]		OSPEEDR8[1:0]		OSPEEDR7[1:0]		OSPEEDR6[1:0]		OSPEEDR5[1:0]		OSPEEDR4[1:0]		OSPEEDR3[1:0]		OSPEEDR2[1:0]		OSPEEDR1[1:0]		OSPEEDR0[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
0x0C	**GPIOA_PUPDR**	PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]		PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[1:0]		PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]		PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[1:0]	
	Reset value	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
0x0C	**GPIOB_PUPDR**	PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]		PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[1:0]		PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]		PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[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	1	0	0	0	0	0	0	0	0
0x10	**GPIOx_IDR** (where x = A..I/J/K)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
	Reset value																	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x14	**GPIOx_ODR** (where x = A..K**)**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	**GPIOx_BSRR** (where x = A..I/J/K)	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** (where x = A..K)	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** (where x = **A..K**)	AFR7[3:0]				AFR6[3:0]				AFR5[3:0]				AFR4[3:0]				AFR3[3:0]				AFR2[3:0]				AFR1[3:0]				AFR0[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** (where x = **A..J**)	AFR15[3:0]				AFR14[3:0]				AFR13[3:0]				AFR12[3:0]				AFR11[3:0]				AFR10[3:0]				AFR9[3:0]				AFR8[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

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

7 系统配置控制器 (SYSCFG)

系统配置控制器主要用于：

- 重新映射存储区域
- 选择以太网 PHY 接口
- 管理 GPIO 的外部中断线连接。

7.1 I/O 补偿单元

默认情况下不使用 I/O 补偿单元。但是，当以 50 MHz 或 100 MHz 模式配置 I/O 输出缓冲区速度时，建议使用补偿单元对 I/O $t_{f(IO)out}/t_{r(IO)out}$ 进行斜率控制，从而降低 I/O 端口噪声对电源的影响。

补偿单元使能后，会设置一个“就绪”标志，指示补偿单元已就绪，可供使用。只有电源电压范围为 2.4 到 3.6 V 时，才可以使用 I/O 补偿单元。

7.2 SYSCFG 寄存器

7.2.1 SYSCFG 存储器重映射寄存器 (SYSCFG_MEMRMP)

SYSCFG memory remap register

此寄存器用于对存储器映射进行特定配置：

- 使用 1 个位来指示使用 BOOT_ADD0 还是 BOOT_ADD1 选项字节定义自举存储器基址。
- 其它位用于交换 FMC SDRAM 存储区域与 FMC NOR/PSRAM 存储区域。

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.	SWP_FMC[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM_BOOT
				rw	rw										r

位 31:12 保留，必须保持复位值。

位 11:10 **SWP_FMC[1:0]**：FMC 存储器映射交换 (FMC memory mapping swap)

由软件置 1 和清零。这些位用于交换 FMC SDRAM 存储区域和 FMC NOR/PSRAM 存储区域，以便在不修改默认 MPU 特性的情况下，允许从 SDRAM 存储区域执行代码

00：不交换 FMC 存储区映射

将 SDRAM 存储区域 1 和存储区域 2 分别映射到 0xC000 0000 和 0xD000 0000（默认映射）

可在 0x60000000 处访问 NOR/RAM（默认映射）

01：交换 NOR/RAM 和 SDRAM 存储区映射，

将 SDRAM 存储区域 1 和存储区域 2 分别映射到 0x6000 0000 和 0x7000 0000

在 0xC000 0000 处映射 NOR/PSRAM 存储区域

10：保留

11：保留

位 9:1 保留，必须保持复位值。

位 0 **MEM_BOOT**：存储器自举映射 (Memory boot mapping)

该位指示使用 BOOT_ADD0 还是 BOOT_ADD1 选项字节定义自举存储器基址。

0：通过 BOOT_ADD0 选项字节来定义自举存储器基址

（出厂复位值：将 TCM-FLASH 映射到 0x00200000）。

1：通过 BOOT_ADD1 选项字节来定义自举存储器基址

（出厂复位值：将系统存储区映射到 0x001 0000）。

注： 有关自举存储器基址选择的详细信息，请参见第 2.3 节：存储器映射。

7.2.2 SYSCFG 外设模式配置寄存器 (SYSCFG_PMC)

SYSCFG peripheral mode configuration 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.	MII_RMII_SEL	Res.	Res.	Res.	Res.	ADCxDC2		
								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 31:24 保留，必须保持复位值。

位 23 **MII_RMII_SEL**：以太网 PHY 接口选择 (Ethernet PHY interface selection)

由软件置 1 和清零。这些位控制以太网 MAC 的 PHY 接口。

0：选择 MII 接口

1：选择 RMII PHY 接口

注： 必须在 MAC 处于复位状态且在使能 MAC 时钟之前完成此配置。

位 22:19 保留，必须保持复位值。

位 18:16 **ADCxDC2**：

0：无影响。

1：请参见 AN4073 了解如何使用此位。

注： 只有满足以下条件时才能将这些位置 1：

- ADC 时钟大于或等于 30 MHz。

- 如果多个 ADC 的转换启动时间不同且采样时间不同，则必须仅选择一个 ADCxDC2 位。

- 当 PWR_CR 寄存器中的 ADCDC1 位置 1 时，这些位则不能置 1。

位 15:0 保留，必须保持复位值。

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

SYSCFG external interrupt configuration register 1

偏移地址：0x08

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15:0 **EXTIx[3:0]**：EXTI x 配置（x = 0 到 3）(EXTI x configuration (x = 0 to 3))

这些位通过软件写入，以选择 EXTIx 外部中断的源输入。

0000：PA[x] 引脚
0001：PB[x] 引脚
0010：PC[x] 引脚
0011：PD[x] 引脚
0100：PE[x] 引脚
0101：PF[x] 引脚
0110：PG[x] 引脚
0111：PH[x] 引脚
1000：PI[x] 引脚
1001：PJ[x] 引脚
1010：PK[x] 引脚

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

SYSCFG external interrupt configuration register 2

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
EXTI7[3:0]				EXTI6[3:0]				EXTI5[3:0]				EXTI4[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:0 **EXTIx[3:0]**：EXTI x 配置（x = 4 到 7）(EXTI x configuration (x = 0 to 3))
这些位通过软件写入，以选择 EXTIx 外部中断的源输入。
0000：PA[x] 引脚
0001：PB[x] 引脚
0010：PC[x] 引脚
0011：PD[x] 引脚
0100：PE[x] 引脚
0101：PF[x] 引脚
0110：PG[x] 引脚
0111：PH[x] 引脚
1000：PI[x] 引脚
1001：PJ[x] 引脚
1010：PK[x] 引脚

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

SYSCFG external interrupt configuration register 3

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
EXTI11[3:0]				EXTI10[3:0]				EXTI9[3:0]				EXTI8[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:0 **EXTIx[3:0]**：EXTI x 配置（x = 8 到 11）(EXTI x configuration (x = 0 to 3))
这些位通过软件写入，以选择 EXTIx 外部中断的源输入。
0000：PA[x] 引脚
0001：PB[x] 引脚
0010：PC[x] 引脚
0011：PD[x] 引脚
0100：PE[x] 引脚
0101：PF[x] 引脚
0110：PG[x] 引脚
0111：PH[x] 引脚
1000：PI[x] 引脚
1001：PJ[x] 引脚
1010：PK[x] 引脚
注： PK[11:8] 未使用

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

SYSCFG external interrupt configuration register 4

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
EXTI15[3:0]				EXTI14[3:0]				EXTI13[3:0]				EXTI12[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15:0 **EXTIx[3:0]**：EXTI x 配置（x = 12 到 15）(EXTI x configuration (x = 0 to 3))
这些位通过软件写入，以选择 EXTIx 外部中断的源输入。
0000：PA[x] 引脚
0001：PB[x] 引脚
0010：PC[x] 引脚
0011：PD[x] 引脚
0100：PE[x] 引脚
0101：PF[x] 引脚
0110：PG[x] 引脚
0111：PH[x] 引脚
1001：PJ[x] 引脚
1010：PK[x] 引脚
注： PK[15:12] 未使用

7.2.7 补偿单元控制寄存器 (SYSCFG_CMPCR)

Compensation cell control 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.	READY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMP_PD
							r								rw

位 31:9 保留，必须保持复位值。

位 8 **READY**：补偿单元就绪标志 (Compensation cell ready flag)
0：I/O 补偿单元未就绪
1：I/O 补偿单元就绪

位 7:2 保留，必须保持复位值。

位 0 **CMP_PD**：补偿单元掉电 (Compensation cell power-down)
0：I/O 补偿单元掉电（关闭）
1：I/O 补偿单元上电（使能）

7.2.8 SYSCFG 寄存器映射

下表列出了 SYSCFG 寄存器映射和复位值。

表 23. SYSCFG 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	SYSCFG_ MEMRMP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWP_FMC[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM_BOOT
	Reset value																					0	0										0
0x04	SYSCFG_PMC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MII_RMII_SEL	Res.	Res.	Res.	Res.	ADC3DC2	ADC2DC2	ADC1DC2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value									0					0	0	0																
0x08	SYSCFG_EXTICR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EXTI3[3:0]				EXTI2[3:0]				EXTI1[3:0]				EXTI0[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	SYSCFG_EXTICR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EXTI7[3:0]				EXTI6[3:0]				EXTI5[3:0]				EXTI4[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	SYSCFG_EXTICR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EXTI11[3:0]				EXTI10[3:0]				EXTI9[3:0]				EXTI8[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	SYSCFG_EXTICR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EXTI15[3:0]				EXTI14[3:0]				EXTI13[3:0]				EXTI12[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	SYSCFG_CMPCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	READY	Res.	Res.	Res.	Res.	Res.	Res.	CMP_PD
	Reset value																									0							0

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

8 直接存储器访问控制器 (DMA)

8.1 DMA 简介

直接存储器访问 (DMA) 用于在外设与存储器之间以及存储器与存储器之间提供高速数据传输。可以在无需任何 CPU 操作的情况下通过 DMA 快速移动数据。这样节省的 CPU 资源可供其它操作使用。

DMA 控制器基于复杂的总线矩阵架构，将功能强大的双 AHB 主总线架构与独立的 FIFO 结合在一起，优化了系统带宽。

两个 DMA 控制器总共有 16 个数据流（每个控制器 8 个），每一个 DMA 控制器都用于管理一个或多个外设的存储器访问请求。每个数据流总共可以有多达 8 个通道（或称请求）。每个通道都有一个仲裁器，用于处理 DMA 请求间的优先级。

8.2 DMA 主要特性

DMA 主要特性是：

- 双 AHB 主总线架构，一个用于存储器访问，另一个用于外设访问
- 仅支持 32 位访问的 AHB 从编程接口
- 每个 DMA 控制器有 8 个数据流，每个数据流有多达 8 个通道（或称请求）
- 每个数据流有四个字深的 32 位先进先出存储器缓冲区 (FIFO)，可用于 FIFO 模式或直接模式：
  - FIFO 模式：可通过软件将阈值级别选取为 FIFO 大小的 1/4、1/2 或 3/4
  - 直接模式

    每个 DMA 请求会立即启动对存储器的传输。当在直接模式（禁止 FIFO）下将 DMA 请求配置为以存储器到外设模式传输数据时，DMA 仅会将一个数据从存储器预加载到内部 FIFO，从而确保一旦外设触发 DMA 请求时则立即传输数据。
- 通过硬件可以将每个数据流配置为：
  - 支持外设到存储器、存储器到外设和存储器到存储器传输的常规通道
  - 在存储器端支持双缓冲的双缓冲区通道
- 8 个数据流中的每一个都连接到专用硬件 DMA 通道（请求）
- DMA 数据流请求之间的优先级可用软件编程（4 个级别：非常高、高、中、低），在软件优先级相同的情况下可以通过硬件决定优先级（例如，请求 0 的优先级高于请求 1）
- 每个数据流也支持存储器到存储器传输的软件触发（仅限 DMA2 控制器）
- 可供每个数据流选择的通道请求多达 8 个。此选择可由软件配置，允许几个外设发起 DMA 请求
- 要传输的数据项的数目可以由 DMA 控制器或外设管理：
  - DMA 流控制器：要传输的数据项的数目可用软件编程，从 1 至 65535
  - 外设流控制器：要传输的数据项的数目未知并由源或目标外设控制，这些外设通过硬件发出传输结束的信号

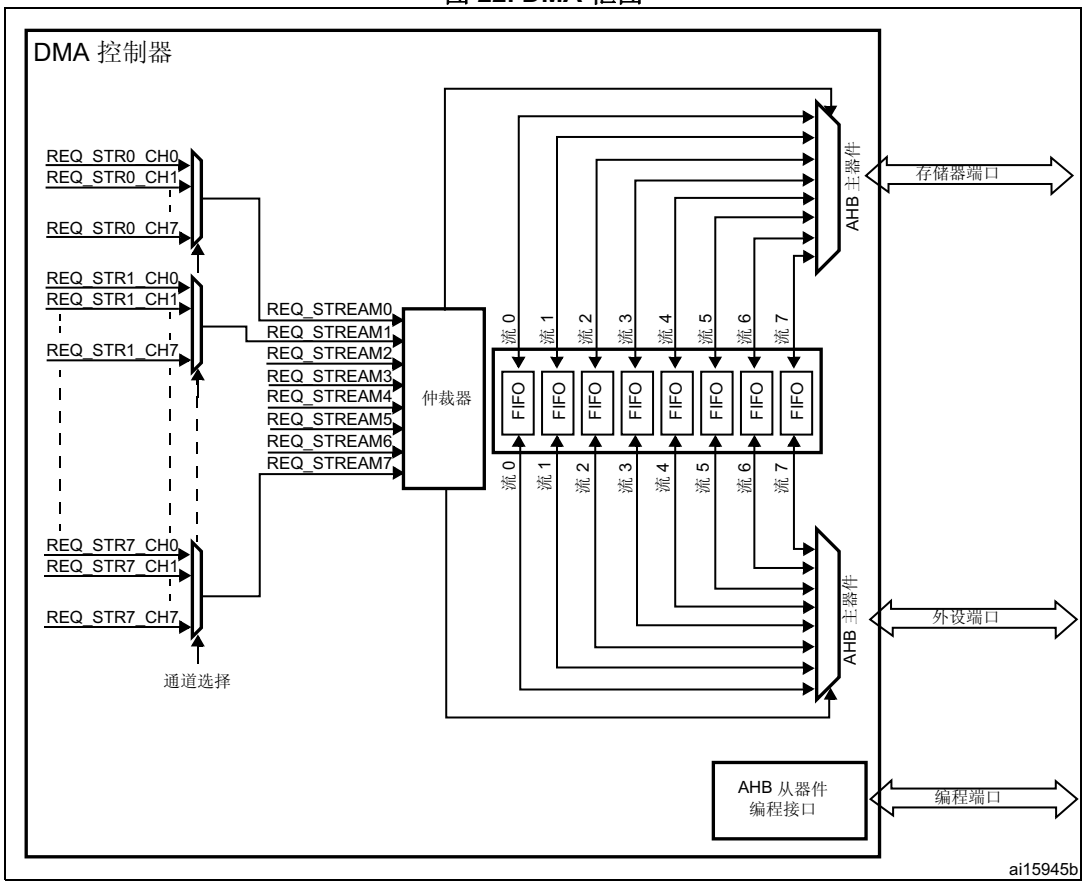
- 独立的源和目标传输宽度（字节、半字、字）：源和目标的数据宽度不相等时，DMA 自动封装/解封必要的传输数据来优化带宽。这个特性仅在 FIFO 模式下可用
- 对源和目标的增量或非增量寻址
- 支持 4 个、8 个和 16 个节拍的增量突发传输。突发增量的大小可由软件配置，通常等于外设 FIFO 大小的一半
- 每个数据流都支持循环缓冲区管理
- 5 个事件标志（DMA 半传输、DMA 传输完成、DMA 传输错误、DMA FIFO 错误、直接模式错误），进行逻辑或运算，从而产生每个数据流的单个中断请求

8.3 DMA 功能说明

8.3.1 一般说明

图 22 显示了 DMA 的框图。

图 22. DMA 框图



DMA 控制器执行直接存储器传输：做为一个 AHB 主设备，它可以控制 AHB 总线矩阵来发起 AHB 传输。

它可以执行下列传输：

- 外设到存储器的传输
- 存储器到外设的传输
- 存储器到存储器的传输

DMA 控制器提供两个 AHB 主端口：AHB 存储器端口（用于连接存储器）和 AHB 外设端口（用于连接外设）。但是，要执行存储器到存储器的传输，AHB 外设端口必须也能访问存储器。

AHB 从端口用于对 DMA 控制器进行编程（它仅支持 32 位访问）。

注： *DMA1 控制器 AHB 外设端口连接到总线矩阵的方式与 DMA2 控制器不同，因此，仅 DMA2 数据流能够执行存储器到存储器的传输。*

有关两个 DMA 控制器的系统实现，请参见*图 1*。

8.3.2 DMA 传输

DMA 传输由给定数目的数据传输序列组成。要传输的数据项的数目及其宽度（8 位、16 位或 32 位）可用软件编程。

每个 DMA 传输包含三项操作：

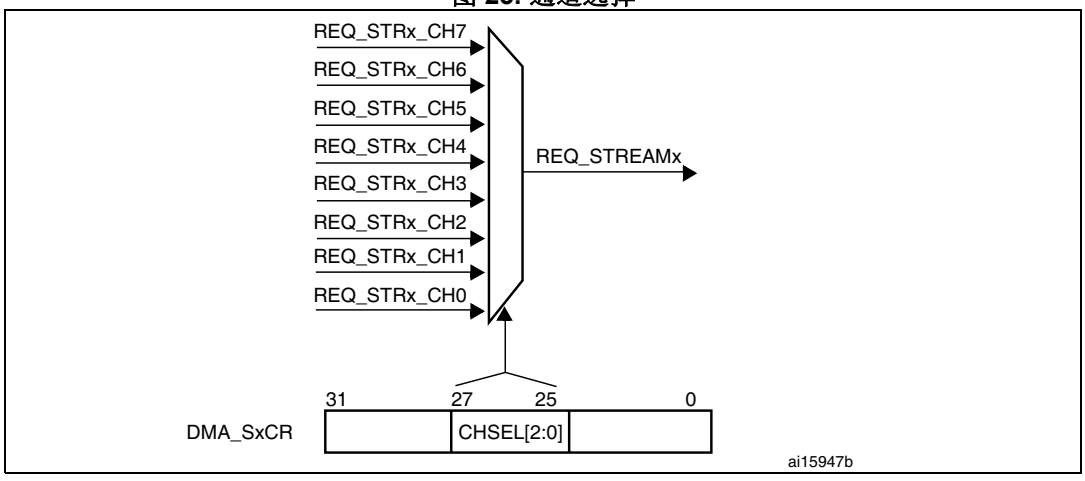
- 通过 DMA_SxPAR 或 DMA_SxM0AR 寄存器寻址，从外设数据寄存器或存储器单元中加载数据
- 通过 DMA_SxPAR 或 DMA_SxM0AR 寄存器寻址，将加载的数据存储到外设数据寄存器或存储器单元
- DMA_SxNDTR 计数器在数据存储结束后递减，该计数器中包含仍需执行的事务数

在产生事件后，外设会向 DMA 控制器发送请求信号。DMA 控制器根据通道优先级处理该请求。只要 DMA 控制器访问外设，DMA 控制器就会向外设发送确认信号。外设获得 DMA 控制器的确认信号后，便会立即释放其请求。一旦外设使请求失效，DMA 控制器就会释放确认信号。如果有更多请求，外设可以启动下一个事务。

8.3.3 通道选择 (Channel selection)

每个数据流都与一个 DMA 请求相关联，此 DMA 请求可以从 8 个可能的通道请求中选出。此选择由 DMA_SxCR 寄存器中的 CHSEL[2:0] 位控制。

图 23. 通道选择



来自外设的 8 个请求（TIM、ADC、SPI、I2C 等）独立连接到每个通道，具体的连接取决于产品实现情况。

表 24 和*表 25* 给出了 DMA 请求映射的示例。

表 24. DMA1 请求映射

外设请求	数据流 0	数据流 1	数据流 2	数据流 3	数据流 4	数据流 5	数据流 6	数据流 7
通道 0	SPI3_RX	SPDIF-RX_DT	SPI3_RX	SPI2_RX	SPI2_TX	SPI3_TX	SPDIF-RX_CS	SPI3_TX
通道 1	I2C1_RX	I2C3_RX	TIM7_UP		TIM7_UP	I2C1_RX	I2C1_TX	I2C1_TX
通道 2	TIM4_CH1	-	I2C4 _RX	TIM4_CH2	-	I2C4 _TX	TIM4_UP	TIM4_CH3
通道 3	-	TIM2_UP TIM2_CH3	I2C3_RX	-	I2C3_TX	TIM2_CH1	TIM2_CH2 TIM2_CH4	TIM2_UP TIM2_CH4
通道 4	UART5_RX	USART3_RX	UART4_RX	USART3_TX	UART4_TX	USART2_RX	USART2_TX	UART5_TX
通道 5	UART8_TX	UART7_TX	TIM3_CH4 TIM3_UP	UART7_RX	TIM3_CH1 TIM3_TRIG	TIM3_CH2	UART8_RX	TIM3_CH3
通道 6	TIM5_CH3 TIM5_UP	TIM5_CH4 TIM5_TRIG	TIM5_CH1	TIM5_CH4 TIM5_TRIG	TIM5_CH2	-	TIM5_UP	-
通道 7	-	TIM6_UP	I2C2_RX	I2C2_RX	USART3_TX	DAC1	DAC2	I2C2_TX

表 25. DMA2 请求映射

外设请求	数据流 0	数据流 1	数据流 2	数据流 3	数据流 4	数据流 5	数据流 6	数据流 7
通道 0	ADC1	SAI1_A	TIM8_CH1 TIM8_CH2 TIM8_CH3	SAI1_A	ADC1	SAI1_B	TIM1_CH1 TIM1_CH2 TIM1_CH3	SAI2_B
通道 1	-	DCMI	ADC2	ADC2	SAI1_B	SPI6_TX	SPI6_RX	DCMI
通道 2	ADC3	ADC3	-	SPI5_RX	SPI5_TX	CRYP_OUT	CRYP_IN	HASH_IN
通道 3	SPI1_RX	-	SPI1_RX	SPI1_TX	SAI2_A	SPI1_TX	SAI2_B	QUADSPI
通道 4	SPI4_RX	SPI4_TX	USART1_RX	SDMMC1	-	USART1_RX	SDMMC1	USART1_TX
通道 5	-	USART6_RX	USART6_RX	SPI4_RX	SPI4_TX	-	USART6_TX	USART6_TX
通道 6	TIM1_TRIG	TIM1_CH1	TIM1_CH2	TIM1_CH1	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP	TIM1_CH3	-
通道 7	-	TIM8_UP	TIM8_CH1	TIM8_CH2	TIM8_CH3	SPI5_RX	SPI5_TX	TIM8_CH4 TIM8_TRIG TIM8_COM

8.3.4 仲裁器

仲裁器为两个 AHB 主端口（存储器和外设端口）提供基于请求优先级的 8 个 DMA 数据流请求管理，并启动外设/存储器访问序列。

优先级管理分为两个阶段：

- 软件：每个数据流优先级都可以在 DMA_SxCR 寄存器中配置。分为四个级别：
  - 非常高优先级
  - 高优先级
  - 中优先级
  - 低优先级
- 硬件：如果两个请求具有相同的软件优先级，则编号低的数据流优先于编号高的数据流。例如，数据流 2 的优先级高于数据流 4。

8.3.5 DMA 数据流

8 个 DMA 控制器数据流都能够提供源和目标之间的单向传输链路。

每个数据流配置后都可以执行：

- 常规类型事务：存储器到外设、外设到存储器或存储器到存储器的传输
- 双缓冲区类型事务：使用两个存储器指针的双缓冲区传输（当 DMA 正在进行自/至缓冲区的读/写操作时，应用程序可以进行至/自其它缓冲区的写/读操作）。

要传输的数据量（多达 65535）可以编程，并与连接到外设 AHB 端口的外设（请求 DMA 传输）的源宽度相关。每个事务完成后，包含要传输的数据项总量的寄存器都会递减。

8.3.6 源、目标和传输模式

源传输和目标传输在整个 4 GB 区域（地址在 0x0000 0000 和 0xFFFF FFFF 之间）都可以寻址外设和存储器。

传输方向使用 DMA_SxCR 寄存器中的 DIR[1:0] 位进行配置，有三种可能的传输方向：存储器到外设、外设到存储器或存储器到存储器。*表 26* 介绍了相应的源和目标地址。

表 26. 源和目标地址

DMA_SxCR 寄存器的位 DIR[1:0]	方向	源地址	目标地址
00	外设到存储器	DMA_SxPAR	DMA_SxM0AR
01	存储器到外设	DMA_SxM0AR	DMA_SxPAR
10	存储器到存储器	DMA_SxPAR	DMA_SxM0AR
11	保留	-	-

当数据宽度（在 DMA_SxCR 寄存器的 PSIZE 或 MSIZE 位中编程）分别是半字或字时，写入 DMA_SxPAR 或 DMA_SxM0AR/M1AR 寄存器的外设或存储器地址必须分别在字或半字地址的边界对齐。

外设到存储器模式

图 24 介绍了这种模式。

使能这种模式（将 DMA_SxCR 寄存器中的位 EN 置 1）时，每次产生外设请求，数据流都会启动数据源到 FIFO 的传输。

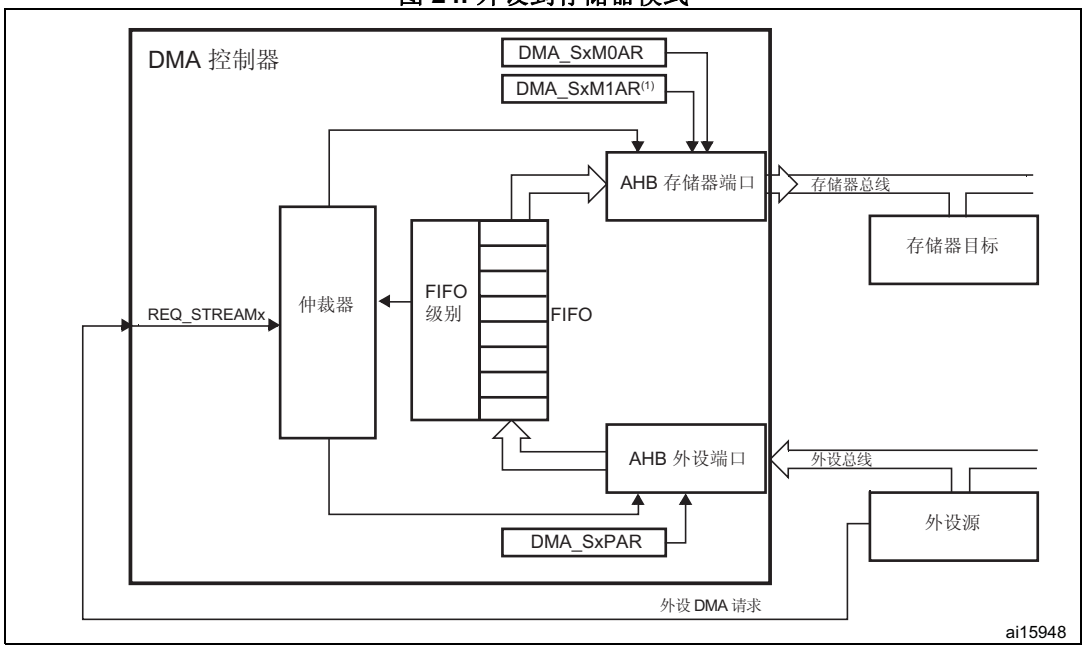
达到 FIFO 的阈值级别时，FIFO 的内容移出并存储到目标中。

如果 DMA_SxNDTR 寄存器达到零、外设请求传输终止（在使用外设流控制器的情况下）或 DMA_SxCR 寄存器中的 EN 位由软件清零，传输即会停止。

在直接模式下（当 DMA_SxFCR 寄存器中的 DMDIS 值为“0”时），不使用 FIFO 的阈值级别控制：每完成一次从外设到 FIFO 的数据传输后，相应的数据立即就会移出并存储到目标中。

只有赢得了数据流的仲裁后，相应数据流才有权访问 AHB 源或目标端口。系统使用在 DMA_SxCR 寄存器 PL[1:0] 位中为每个数据流定义的优先级执行仲裁。

图 24. 外设到存储器模式



1. 用于双缓冲区模式。

存储器到外设模式

图 25 介绍了这种模式。

使能这种模式（将 DMA_SxCR 寄存器中的 EN 位置 1）时，数据流会立即启动传输，从源完全填充 FIFO。

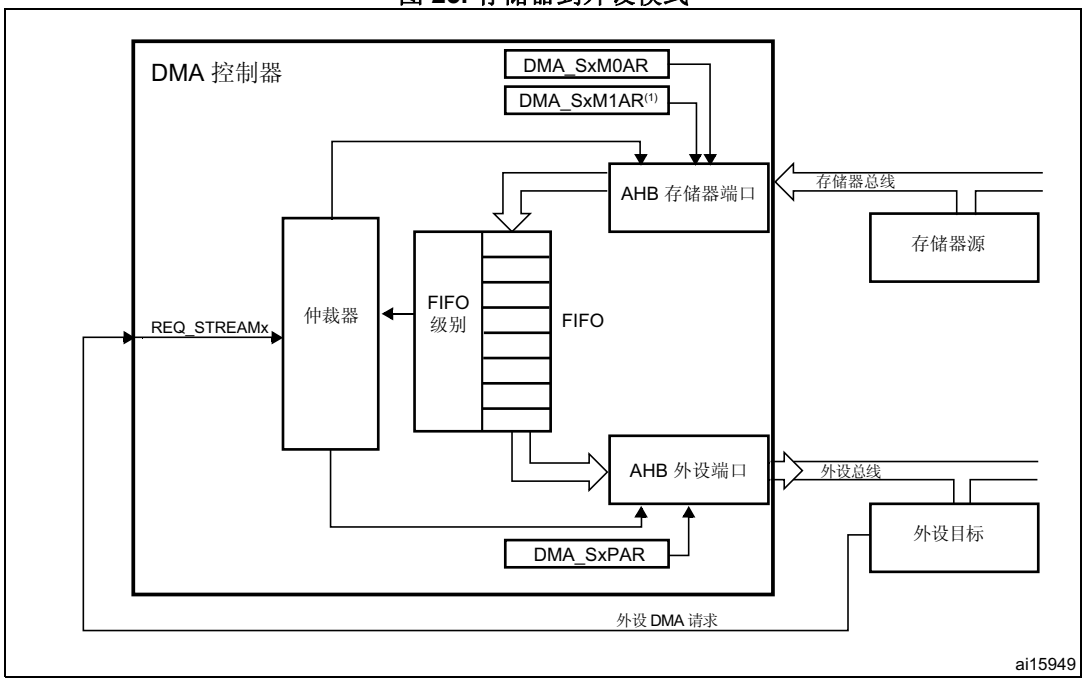
每次发生外设请求，FIFO 的内容都会移出并存储到目标中。当 FIFO 的级别小于或等于预定义的阈值级别时，将使用存储器中的数据完全重载 FIFO。

如果 DMA_SxNDTR 寄存器达到零、外设请求传输终止（在使用外设流控制器的情况下）或 DMA_SxCR 寄存器中的 EN 位由软件清零，传输即会停止。

在直接模式下（当 DMA_SxFCR 寄存器中的 DMDIS 值为“0”时），不使用 FIFO 的阈值级别。一旦使能了数据流，DMA 便会预装载第一个数据，将其传输到内部 FIFO。一旦外设请求数据传输，DMA 便会将预装载的值传输到配置的目标。然后，它会使用要传输的下一个数据再次重载内部空 FIFO。预装载的数据大小为 DMA_SxCR 寄存器中 PSIZE 位字段的值。

只有赢得了数据流的仲裁后，相应数据流才有权访问 AHB 源或目标端口。系统使用在 DMA_SxCR 寄存器 PL[1:0] 位中为每个数据流定义的优先级执行仲裁。

图 25. 存储器到外设模式



1. 用于双缓冲区模式。

存储器到存储器模式

DMA 通道在没有外设请求触发的情况下同样可以工作。此为*图 26* 中介绍的存储器到存储器模式。

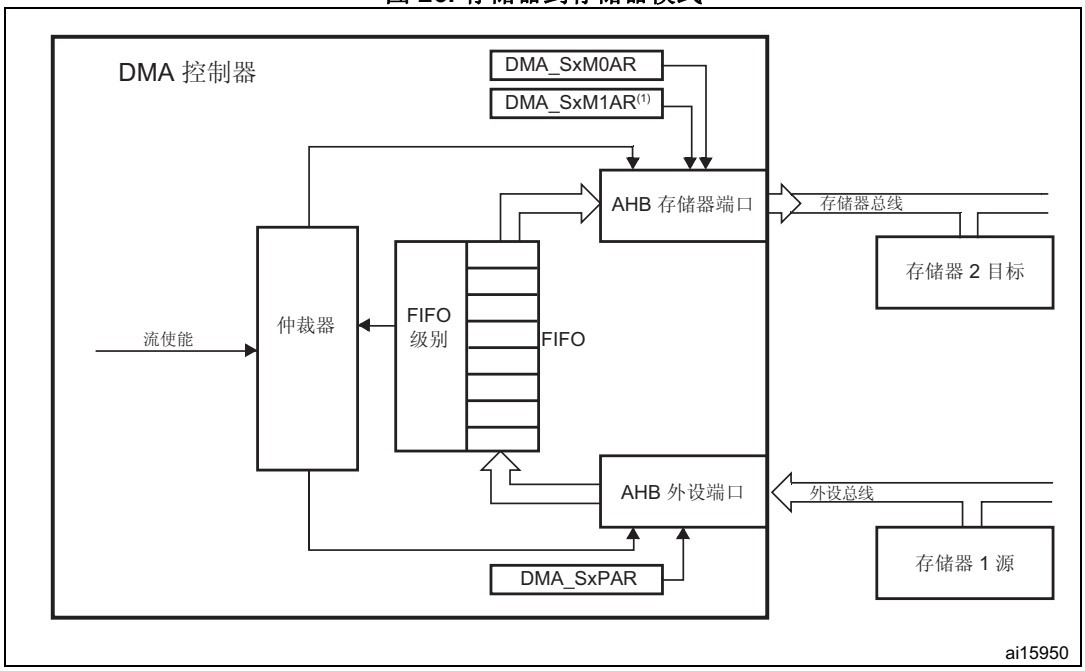
通过将 DMA_SxCR 寄存器中的使能位 (EN) 置 1 来使能数据流时，数据流会立即开始填充 FIFO，直至达到阈值级别。达到阈值级别后，FIFO 的内容便会移出，并存储到目标中。

如果 DMA_SxNDTR 寄存器达到零或 DMA_SxCR 寄存器中的 EN 位由软件清零，传输即会停止。

只有赢得了数据流的仲裁后，相应数据流才有权访问 AHB 源或目标端口。系统使用在 DMA_SxCR 寄存器 PL[1:0] 位中为每个数据流定义的优先级执行仲裁。

注： *使用存储器到存储器模式时，不允许循环模式和直接模式。*
只有 DMA2 控制器能够执行存储器到存储器的传输。

图 26. 存储器到存储器模式



1. 用于双缓冲区模式。

8.3.7 指针递增

根据 DMA_SxCR 寄存器中 PINC 和 MINC 位的状态，外设和存储器指针在每次传输后可以自动向后递增或保持常量。

通过单个寄存器访问外设源或目标数据时，禁止递增模式十分有用。

如果使能了递增模式，则根据在 DMA_SxCR 寄存器 PSIZE 或 MSIZE 位中编程的数据宽度，下一次传输的地址将是前一次传输的地址递增 1（对于字节）、2（对于半字）或 4（对于字）。

为了优化封装操作，可以不管 AHB 外设端口上传输的数据的大小，将外设地址的增量偏移大小固定下来。DMA_SxCR 寄存器中的 PINCOS 位用于将增量偏移大小与外设 AHB 端口或 32 位地址（此时地址递增 4）上的数据大小对齐。PINCOS 位仅对 AHB 外设端口有影响。

如果将 PINCOS 位置 1，则不论 PSIZE 值是多少，下一次传输的地址总是前一次传输的地址递增 4（自动与 32 位地址对齐）。但是，AHB 存储器端口不受此操作影响。

8.3.8 循环模式

循环模式可用于处理循环缓冲区和连续数据流（例如 ADC 扫描模式）。可以使用 DMA_SxCR 寄存器中的 CIRC 位使能此特性。

当激活循环模式时，要传输的数据项的数目在数据流配置阶段自动用设置的初始值进行加载，并继续响应 DMA 请求。

注： *在循环模式下，如果为存储器配置了突发模式，必须遵循下列规则：*

DMA_SxNDTR = ((Mburst 节拍) × (Msize)/(Psize)) 的倍数，其中：

- *(Mburst 节拍) = 4、8 或 16（取决于 DMA_SxCR 寄存器中的 MBURST 位）*
- *((Msize)/(Psize)) = 1、2、4、1/2 或 1/4（Msize 和 Psize 表示 DMA_SxCR 寄存器中的 MSIZE 和 PSIZE 位。它们与字节相关）*
- *DMA_SxNDTR = AHB 外设端口上要传输的数据项的数目*

Mburst 节拍 = 8 (INCR8)，MSIZE = “00”（字节）和 PSIZE = “01”（半字），在此例中：DMA_SxNDTR 必须是 (8 × 1/2 = 4) 的倍数。

如果不遵循此公式，则 DMA 行为和数据完整性得不到保证。

NDTR 还必须是外设突发大小与外设数据大小乘积的倍数，否则会导致错误的 DMA 行为。

8.3.9 双缓冲区模式

此模式可用于所有 DMA1 和 DMA2 数据流。

通过将 DMA_SxCR 寄存器中的 DBM 位置 1，即可使能双缓冲区模式。

除了有两个存储器指针之外，双缓冲区数据流的工作方式与常规（单缓冲区）数据流的一样。使能双缓冲区模式时，将自动使能循环模式（DMA_SxCR 中的 CIRC 位的状态是“无关”），并在每次事务结束时交换存储器指针。

在此模式下，每次事务结束时，DMA 控制器都从一个存储器目标交换为另一个存储器目标。这样，软件在处理一个存储器区域的同时，DMA 传输还可以填充/使用第二个存储器区域。如 *表 27：双缓冲区模式下的源和目标地址寄存器 (DBM=1)* 所述，双缓冲区数据流可以双向工作（存储器既可以是源也可以是目标）。

注： *在双缓冲区模式下使能数据流时，可遵循下列条件，实时更新 AHB 存储器的基址（DMA_SxM0AR 或 DMA_SxM1AR）：*

- *当 DMA_SxCR 寄存器中的 CT 位为“0”时，可以写入 DMA_SxM1AR 寄存器。当 CT = “1”时，试图写入此寄存器会将错误标志位 (TEIF) 置 1，并自动禁止数据流。*
- *当 DMA_SxCR 寄存器中的 CT 位为“1”时，可以写入 DMA_SxM0AR 寄存器。当 CT = “0”时，试图写入此寄存器会将错误标志位 (TEIF) 置 1，并自动禁止数据流。*

为避免出现任何错误状态，建议在 TCIF 标志位置位时立即更改基址。因为此时根据上述两个条件之一，目标存储器依据 DMA_SxCR 寄存器中 CT 值的情况，一定已从存储器 0 更改为存储器 1（或从存储器 1 更改为存储器 0）。

对于所有其它模式（双缓冲区模式除外），一旦使能数据流，存储器地址寄存器即被写保护。

表 27. 双缓冲区模式下的源和目标地址寄存器 (DBM=1)

DMA_SxCR 寄存器的位 DIR[1:0]	方向	源地址	目标地址
00	外设到存储器	DMA_SxPAR	DMA_SxM0AR/DMA_SxM1AR
01	存储器到外设	DMA_SxM0AR/DMA_SxM1AR	DMA_SxPAR
10	不允许(1)		
11	保留	-	-

1. 使能双缓冲区模式时，自动使能循环模式。由于存储器到存储器模式与循环模式不兼容，所以当使能双缓冲区模式时，不允许配置存储器到存储器模式。

8.3.10 可编程数据宽度、封装/解封、字节序

要传输的数据项数目必须在使能数据流之前编程到 DMA_SxNDTR（要传输数据项数目位，NDT）中，当流控制器是外设且 DMA_SxCR 中的 PFCTRL 位置为 1 时除外。

当使用内部 FIFO 时，源和目标数据的数据宽度可以通过 DMA_SxCR 寄存器的 PSIZE 和 MSIZE 位（可以是 8、16 或32 位）编程。

当 PSIZE 和 MSIZE 不相等时：

- 在 DMA_SxNDTR 寄存器中配置的要传输的数据项数目的数据宽度等于外设总线的宽度（由 DMA_SxCR 寄存器中的 PSIZE 位配置）。例如，在外设到存储器、存储器到外设或存储器到存储器传输的情况下，如果将 PSIZE[1:0] 位配置为半字，则要传输的字节数等于 2 × NDT。
- DMA 控制器仅按小字节序寻址源和目标。相关内容在*表28：封装/解封和字节序行为（位 PINC = MINC = 1）*中介绍。

在封装/解封数据的过程中，如果在数据完全封装/解封前中断操作，则有数据损坏的危险。因此，为了确保数据一致性，可将数据流配置成生成突发传输：在这种情况下，属于一个突发的每组传输不可分割（请参见*第 8.3.11 节：单次传输和突发传输*）。

在直接模式下（DMA_SxFCR 寄存器中的 DMDIS = 0），不能进行数据封装/解封。这种情况下，不允许源与目标的传输数据宽度不同，二者必须相等，并由 DMA_SxCR 中的 PSIZE 位定义，MSIZE 位的状态是“无关”。

表 28. 封装/解封和字节序行为（位 PINC = MINC = 1）

AHB 存储器端口宽度	AHB 外设端口宽度	要传输的数据项的数目 (NDT)	存储器传输数目	存储器端口地址/字节通道	外设传输数目	外设端口地址/字节通道	
						PINCOS = 1	PINCOS = 0
8	8	4	1 2 3 4	0x0/B0[7:0] 0x1/B1[7:0] 0x2/B2[7:0] 0x3/B3[7:0]	1 2 3 4	0x0/B0[7:0] 0x4/B1[7:0] 0x8/B2[7:0] 0xC/B3[7:0]	0x0/B0[7:0] 0x1/B1[7:0] 0x2/B2[7:0] 0x3/B3[7:0]
8	16	2	1 2 3 4	0x0/B0[7:0] 0x1/B1[7:0] 0x2/B2[7:0] 0x3/B3[7:0]	1 2	0x0/B1|B0[15:0] 0x4/B3|B2[15:0]	0x0/B1|B0[15:0] 0x2/B3|B2[15:0]
8	32	1	1 2 3 4	0x0/B0[7:0] 0x1/B1[7:0] 0x2/B2[7:0] 0x3/B3[7:0]	1	0x0/B3|B2|B1|B0[31:0]	0x0/B3|B2|B1|B0[31:0]
16	8	4	1 2	0x0/B1|B0[15:0] 0x2/B3|B2[15:0]	1 2 3 4	0x0/B0[7:0] 0x4/B1[7:0] 0x8/B2[7:0] 0xC/B3[7:0]	0x0/B0[7:0] 0x1/B1[7:0] 0x2/B2[7:0] 0x3/B3[7:0]
16	16	2	1 2	0x0/B1|B0[15:0] 0x2/B1|B0[15:0]	1 2	0x0/B1|B0[15:0] 0x4/B3|B2[15:0]	0x0/B1|B0[15:0] 0x2/B3|B2[15:0]
16	32	1	1 2	0x0/B1|B0[15:0] 0x2/B3|B2[15:0]	1	0x0/B3|B2|B1|B0[31:0]	0x0/B3|B2|B1|B0[31:0]
32	8	4	1	0x0/B3|B2|B1|B0[31:0]	1 2 3 4	0x0/B0[7:0] 0x4/B1[7:0] 0x8/B2[7:0] 0xC/B3[7:0]	0x0/B0[7:0] 0x1/B1[7:0] 0x2/B2[7:0] 0x3/B3[7:0]
32	16	2	1	0x0/B3|B2|B1|B0[31:0]	1 2	0x0/B1|B0[15:0] 0x4/B3|B2[15:0]	0x0/B1|B0[15:0] 0x2/B3|B2[15:0]
32	32	1	1	0x0/B3|B2|B1|B0 [31:0]	1	0x0/B3|B2|B1|B0 [31:0]	0x0/B3|B2|B1|B0[31:0]

注： *外设端口可以是源或目标（在存储器到存储器传输的情况下，也可能是存储器源）。*

必须配置 PSIZE、MSIZE 和 NDT[15:0]，以确保最后一次传输的完整性。当外设端口的数据宽度（PSIZE 位）小于存储器端口的数据宽度（MSIZE 位）时，可能会发生数据传输不完整的情况。此限制条件汇总在*表 29* 中。

表 29. PSIZE 与 MSIZE 确定时对 NDT 的限制条件

DMA_SxCR 的 PSIZE[1:0]	DMA_SxCR 的 MSIZE[1:0]	DMA_SxNDTR 的 NDT[15:0]
00（8 位）	01（16 位）	必须是 2 的倍数
00（8 位）	10（32 位）	必须是 4 的倍数
01（16 位）	10（32 位）	必须是 2 的倍数

8.3.11 单次传输和突发传输

DMA 控制器可以产生单次传输或 4 个、8 个和 16 个节拍的增量突发传输。

突发大小通过软件针对两个 AHB 端口独立配置，配置时使用 DMA_SxCR 寄存器中的 MBURST[1:0] 和 PBURST[1:0] 位。

突发大小指示突发中的节拍数，而不是传输的字节数。

为确保数据一致性，形成突发的每一组传输都不可分割：在突发传输序列期间，AHB 传输会锁定，并且 AHB 总线矩阵的仲裁器不解除对 DMA 主总线的授权。

根据单次或突发配置的情况，每个 DMA 请求在 AHB 外设端口上相应地启动不同数量的传输。

- 当 AHB 外设端口被配置为单次传输时，根据 DMA_SxCR 寄存器 PSIZE[1:0] 位的值，每个 DMA 请求产生一次字节、半字或字的数据传输。
- 当 AHB 外设端口被配置为突发传输时，根据 DMA_SxCR 寄存器 PBURST[1:0] 和 PSIZE[1:0] 位的值，每个 DMA 请求相应地生成 4 个、8 个或 16 个节拍的字节、半字或字的传输。

对于需要配置 MBURST 和 MSIZE 位的 AHB 存储器端口，必须考虑与上述相同的内容。

在直接模式下，数据流只能生成单次传输，而 MBURST[1:0] 和 PBURST[1:0] 位由硬件强制配置。

必须选择地址指针（DMA_SxPAR 或 DMA_SxM0AR 寄存器），以确保一个突发块内的所有传输在等于传输大小的地址边界对齐。

选择突发配置必须要遵守 AHB 协议，即突发传输 *不得*越过 1 KB 地址边界，因为可以分配给单个从设备的最小地址空间是 1 KB。这意味着突发块传输不应越过 1 KB 地址边界，否则就会产生一个 AHB 错误，并且 DMA 寄存器不会报告这个错误。

8.3.12 FIFO

FIFO 结构

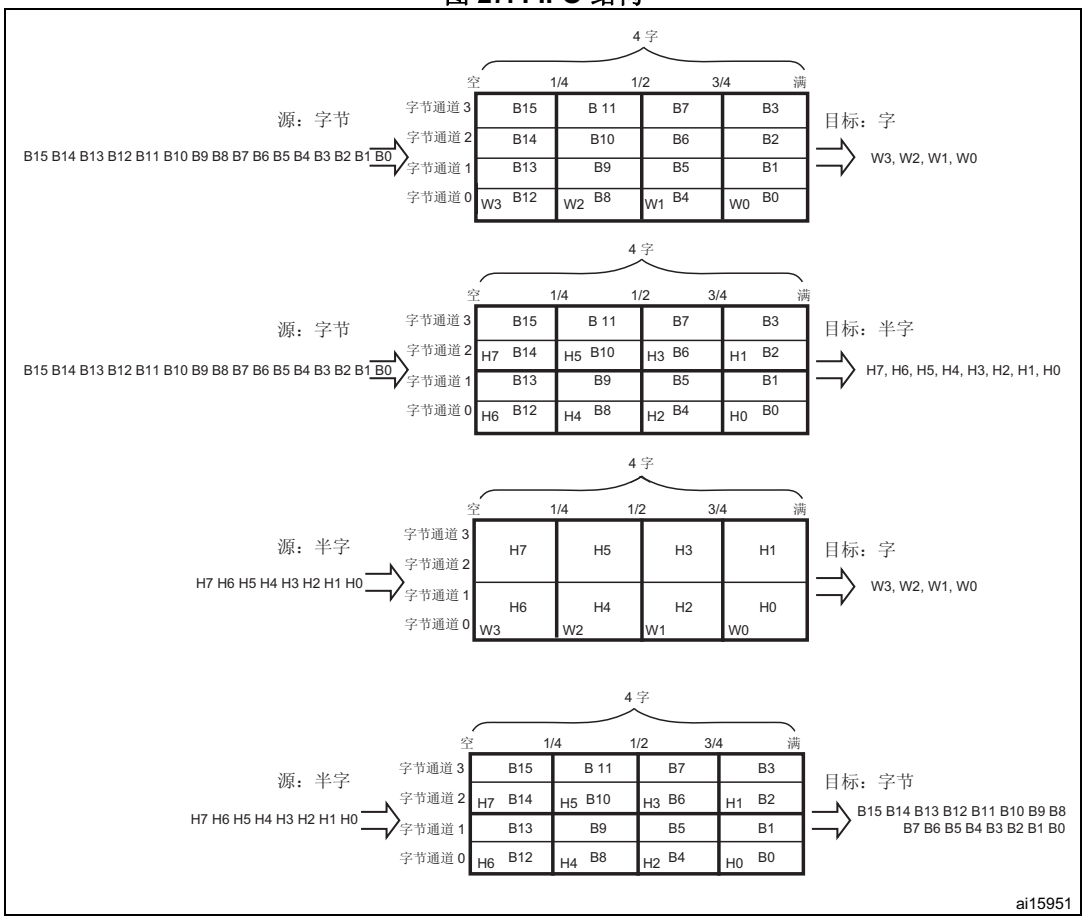
FIFO 用于在源数据传输到目标之前临时存储这些数据。

每个数据流都有一个独立的 4 字 FIFO，阈值级别可由软件配置为 1/4、1/2、3/4 或满。

为了使能 FIFO 阈值级别，必须通过将 DMA_SxFCR 寄存器中的 DMDIS 位置 1 来禁止直接模式。

FIFO 的结构随源与目标数据宽度而不同，相关内容在*图 27：FIFO 结构*中介绍。

图 27. FIFO 结构



FIFO 阈值与突发配置

选择 FIFO 阈值（DMA_SxFCR 寄存器的位 FTH[1:0]）和存储器突发大小（DMA_SxCR 寄存器的 MBURST[1:0] 位）时需要小心：FIFO 阈值指向的内容必须与整数个存储器突发传输完全匹配。如果不是这样，当使能数据流时将生成一个 FIFO 错误（DMA_HISR 或 DMA_LISR 寄存器的标志 FEIFx），然后将自动禁止数据流。允许的和禁止的配置在*表 30* 中介绍。禁止的配置在表中以灰色突出显示。

表 30. FIFO 阈值配置

MSIZE	FIFO 级别	MBURST = INCR4	MBURST = INCR8	MBURST = INCR16
字节	1/4	4 个节拍的 1 次突发	禁止	禁止
	1/2	4 个节拍的 2 次突发	8 个节拍的 1 次突发	
	3/4	4 个节拍的 3 次突发	禁止	
	满	4 个节拍的 4 次突发	8 个节拍的 2 次突发	16 个节拍的 1 次突发

表 30. FIFO 阈值配置（续）

MSIZE	FIFO 级别	MBURST = INCR4	MBURST = INCR8	MBURST = INCR16
半字	1/4	禁止	禁止	禁止
	1/2	4 个节拍的 1 次突发		
	3/4	禁止		
	满	4 个节拍的 2 次突发	8 个节拍的 1 次突发	
字	1/4	禁止	禁止	
	1/2			
	3/4			
	满	4 个节拍的 1 次突发		

所有这些情况下，突发大小与数据大小的乘积不得超过 FIFO 大小（数据大小可以为：1（字节）、2（半字）或 4（字））。

如果发生下列一种情况，会导致 DMA 传输结束时出现不完整的突发传输：

- 对于 AHB 外设端口配置：数据项总数（在 DMA_SxNDTR 寄存器中设置）不是突发大小与数据大小乘积的倍数。
- 对于 AHB 存储器端口配置：要传输到存储器的 FIFO 中的剩余数据项的数目不是突发大小与数据大小乘积的倍数。

在这些情况下，即使在 DMA 数据流配置期间请求突发事务，要传输的剩余数据也将由 DMA 在单独模式下管理。

注： *当在外设 AHB 端口上请求突发传输并且使用 FIFO（DMA_SxCR 寄存器中 DMDIS = 1）时，必须根据 DMA 数据流方向遵守下列规则来避免出现上溢或下溢情况：*

如果 (PBURST × PSIZE) = FIFO_SIZE（4 字），则当 PSIZE = 1、2 或 4，PBURST = 4、8 或 16 时，禁止 FIFO_Threshold = 3/4。

此规则将确保一次释放足够的 FIFO 空间来处理外设的请求。

FIFO 刷新

当复位 DMA_SxCR 寄存器中的 EN 位来禁止数据流，以及配置数据流来管理外设到存储器或存储器到存储器的传输时，可以刷新 FIFO。因为如果禁止数据流时仍有某些数据存留在 FIFO 中，DMA 控制器会将剩余的数据继续传输到目标（即使已经有效禁止了数据流）。刷新完成时，会将 DMA_LISR 或 DMA_HISR 寄存器中的传输完成状态位 (TCIFx) 置 1。

在这种情况下，剩余数据计数器 DMA_SxNDTR 保持的值指示在目标存储器现有多少可用数据项。

请注意在 FIFO 刷新操作期间，如果 FIFO 中要传输到存储器的剩余数据项的数目（以字节为单位）小于存储器数据宽度（例如在 MSIZE 配置为字时 FIFO 中为 2 个字节），则会使用在 DMA_SxCR 寄存器中的 MSIZE 位设置的数据宽度发送数据。这意味着将使用非预期值写入存储器。软件可以读取 DMA_SxNDTR 寄存器来确定包含良好数据的存储器区域（起始地址和最后地址）。

如果 FIFO 中剩余数据项的数目小于突发大小，并且数据流通过将 DMA_SxCR 寄存器 MBURST 位置 1 进行配置来管理在 AHB 存储器端口上的突发传输，则使用单次传输来完成 FIFO 的刷新。

直接模式

默认情况下，FIFO 以直接模式操作（将 DMA_SxFCR 中的 DMDIS 位置 1），不使用 FIFO 阈值级别。如果在每次 DMA 请求之后，系统需要至/自存储器的立即和单独传输，这种模式非常有用。

当在直接模式（禁止 FIFO）下将 DMA 配置为以存储器到外设模式传输数据时，DMA 会将一个数据从存储器预加载到内部 FIFO，从而确保一旦外设触发 DMA 请求时则立即传输数据。

为了避免 FIFO 饱和，建议使用高优先级配置相应的数据流。

该模式仅限以下方式的传输：

- 源和目标传输宽度相等，并均由 DMA_SxCR 中的 PSIZE[1:0] 位定义（MSIZE[1:0] 位的状态是“无关”）
- 不可能进行突发传输（DMA_SxCR 中的 PBURST[1:0] 和 MBURST[1:0] 位的状态是“无关”）

当实现存储器到存储器传输时不得使用直接模式。

8.3.13 DMA 传输完成

以下各种事件均可以结束传输过程，并将 DMA_LISR 或 DMA_HISR 状态寄存器中的 TCIFx 位置 1：

- 在 DMA 流控制器模式下：
  - 在存储器到外设模式下，DMA_SxNDTR 计数器已达到零
  - 传输结束前禁止了数据流（通过将 DMA_SxCR 寄存器中的 EN 位清零），并且（当传输是外设到存储器或存储器到存储器）所有的剩余数据均已从 FIFO 刷新到存储器
- 在外设流控制器模式下：
  - 已从外设生成最后的外部突发请求或单独请求，并当 DMA 在外设到存储器模式下工作时，剩余数据已从 FIFO 传输到存储器
  - 数据流由软件禁止，并当 DMA 在外设到存储器模式下工作时，剩余数据已从 FIFO 传输到存储器

注： *仅在外设到存储器模式下，传输的完成取决于 FIFO 中要传输到存储器的剩余数据。这种情况不适用于存储器到外设模式。*

如果是在非循环模式下配置数据流，传输结束后（即要传输的数据数目达到零），除非软件重新对数据流编程并重新使能数据流（通过将 DMA_SxCR 寄存器中的 EN 位置 1），否则 DMA 即会停止传输（通过硬件将 DMA_SxCR 寄存器中的 EN 位清零）并且不再响应任何 DMA 请求。

8.3.14 DMA 传输暂停

可以随时暂停 DMA 传输以供稍后重新开始；也可以在 DMA 传输结束前明确禁止传输。

分为两种情况：

- 数据流禁止传输，以后不从停止点重新开始。这种情况下，只需将 DMA_SxCR 寄存器中的 EN 位清零来禁止数据流，除此之外不需要任何其他操作。禁止数据流可能要花费一些时间（需要首先完成正在进行的传输）。需要将传输完成中断标志（DMA_LISR 或 DMA_HISR 寄存器中的 TCIF）置 1 来指示传输结束。当前 DMA_SxCR 中的 EN 位的值是“0”，借此确认数据流已经终止传输。DMA_SxNDTR 寄存器包含数据流停止时剩余数据项的数目，这样软件便可以确定数据流中断前已传输了多少数据项。
- 数据流在 DMA_SxNDTR 寄存器中要传输的剩余数据项数目达到 0 之前暂停传输。目的是以后通过重新使能数据流重新开始传输。为了在传输停止点重新开始传输，软件必须在通过写入 DMA_SxCR 寄存器中的 EN 位（然后检查确认该位为 ‘0’）禁止数据流之后，首先读取 DMA_SxNDTR 寄存器来了解已经收集的数据项的数目。然后：
  - 必须更新外设和/或存储器地址以调整地址指针
  - 必须使用要传输的剩余数据项的数目（禁止数据流时读取的值）更新 SxNDTR 寄存器
  - 然后可以重新使能数据流，从停止点重新开始传输

注： *请注意，传输完成中断标志（DMA_LISR 或 DMA_HISR 中的 TCIF）置 1 将指示因数据流中断而结束传输。*

8.3.15 流控制器

控制要传输的数据数目的实体称为流控制器。此流控制器使用 DMA_SxCR 寄存器中的 PFCTRL 位针对每个数据流独立配置。

流控制器可以是：

- DMA 控制器：在这种情况下，要传输的数据项的数目在使能 DMA 数据流之前由软件编程到 DMA_SxNDTR 寄存器。
- 外设源或目标：当要传输的数据项的数目未知时属于这种情况。当所传输的是最后的数据时，外设通过硬件向 DMA 控制器发出指示。仅限能够发出传输结束信号的外设支持此功能，也就是：
  - SDMMC1

当外设流控制器用于给定数据流时，写入 DMA_SxNDTR 的值对 DMA 传输没有作用。实际上，不论写入什么值，一旦使能数据流，硬件即会将该值强制置为 0xFFFF 来执行下列方案：

- 预期的数据流中断：DMA_SxCR 寄存器中的 EN 位由软件重置为 0，以在外设发送最后的数据硬件信号（单独或突发）之前停止数据流。这样，在外设到存储器 DMA 传输的情况下，数据流即会关闭并触发 FIFO 刷新。状态寄存器中相应数据流的 TCIFx 标志置 1 以指示 DMA 完成传输。要了解 DMA 传输期间传输的数据项的数目，请读取 DMA_SxNDTR 寄存器并应用下列公式：
  - 传输的数据数目 = 0xFFFF – DMA_SxNDTR

- 因接收到最后的数据硬件信号而引起的正常数据流中断：当外设请求最后的传输（单独或突发）并当此传输完成时，自动中断数据流。相应流的 TCIFx 标志在状态寄存器中置 1 以指示 DMA 传输完成。要了解传输的数据项的数目，请读取 DMA_SxNDTR 寄存器并应用与上面相同的公式。
- DMA_SxNDTR 寄存器达到 0：状态寄存器中相应数据流的 TCIFx 标志置 1 以指示强制的 DMA 传输完成。即使尚未置位最后的数据硬件信号（单独或突发），也会自动关闭数据流。已传输的数据不会丢失。这意味着即使在外设流控制模式下，DMA 在单独的事务中最多处理 65535 个数据项。

注： *当在存储器到存储器模式下配置时，DMA 始终是流控制器，而 PFCTRL 位由硬件强制置为 0。*

在外设流控制器模式下禁止循环模式。

8.3.16 可能的 DMA 配置汇总

表 31 汇总了各种可能的 DMA 配置。禁止的配置在表中以灰色突出显示。

表 31. 可能的 DMA 配置

DMA 传输模式	源	目标	流控制器	循环模式	传输类型	直接模式	双缓冲区模式
外设到存储器	AHB 外设端口	AHB 存储器端口	DMA	允许	单独	允许	允许
					突发	禁止	
			外设	禁止	单独	允许	禁止
					突发	禁止	
存储器到外设	AHB 存储器端口	AHB 外设端口	DMA	允许	单独	允许	允许
					突发	禁止	
			外设	禁止	单独	允许	禁止
					突发	禁止	
存储器到存储器	AHB 外设端口	AHB 存储器端口	仅 DMA	禁止	单独	禁止	禁止
					突发		

8.3.17 流配置过程

配置 DMA 数据流 x（其中 x 是数据流编号）时应遵守下面的顺序：

1. 如果使能了数据流，通过重置 DMA_SxCR 寄存器中的 EN 位将其禁止，然后读取此位以确认没有正在进行的数据流操作。将此位写为 0 不会立即生效，因为实际上只有所有当前传输都已完成时才会将其写为 0。当所读取 EN 位的值为 0 时，才表示可以配置数据流。因此在开始任何数据流配置之前，需要等待 EN 位置 0。应将先前的数据块 DMA 传输中在状态寄存器（DMA_LISR 和 DMA_HISR）中置 1 的所有数据流专用的位置 0，然后才可重新使能数据流。
2. 在 DMA_SxPAR 寄存器中设置外设端口寄存器地址。外设事件发生后，数据会从此地址移动到外设端口或从外设端口移动到此地址。
3. 在 DMA_SxMA0R 寄存器（在双缓冲区模式的情况下还有 DMA_SxMA1R 寄存器）中设置存储器地址。外设事件发生后，将从此存储器读取数据或将数据写入此存储器。
4. 在 DMA_SxNDTR 寄存器中配置要传输的数据项的总数。每出现一次外设事件或每出现一个节拍的突发传输，该值都会递减。
5. 使用 DMA_SxCR 寄存器中的 CHSEL[2:0] 选择 DMA 通道（请求）。
6. 如果外设用作流控制器而且支持此功能，请将 DMA_SxCR 寄存器中的 PFCTRL 位置 1。
7. 使用 DMA_SxCR 寄存器中的 PL[1:0] 位配置数据流优先级。
8. 配置 FIFO 的使用情况（使能或禁止，发送和接收阈值）
9. 配置数据传输方向、外设和存储器增量 / 固定模式、单独或突发事务、外设和存储器数据宽度、循环模式、双缓冲区模式和传输完成一半和/或全部完成，和/或 DMA_SxCR 寄存器中错误的中断。
10. 通过将 DMA_SxCR 寄存器中的 EN 位置 1 激活数据流。

一旦使能了流，即可响应连接到数据流的外设发出的任何 DMA 请求。

一旦在 AHB 目标端口上传输了一半数据，传输一半标志 (HTIF) 便会置 1，如果传输一半中断使能位 (HTIE) 置 1，还会生成中断。传输结束时，传输完成标志 (TCIF) 便会置 1，如果传输完成中断使能位 (TCIE) 置 1，还会生成中断。

警告： **要关闭连接到 DMA 数据流请求的外设，必须首先关闭外设连接的 DMA 数据流，然后等待 EN 位 = 0。只有这样才能安全地禁止外设。**

8.3.18 错误管理

DMA 控制器可以检测到以下错误：

- **传输错误**：当发生下列情况时，传输错误中断标志 (TEIFx) 将置 1：
  - DMA 读或写访问期间发生总线错误
  - 软件请求在双缓冲区模式下写访问存储器地址寄存器，但是，已使能数据流，并且当前目标存储器是受写入存储器地址寄存器操作影响的存储器（请参见*第 8.3.9 节：双缓冲区模式*）
- **FIFO 错误**：如果发生下列情况，FIFO 错误中断标志 (FEIFx) 将置 1：
  - 检测到 FIFO 下溢情况
  - 检测到 FIFO 上溢情况（在存储器到存储器模式下由 DMA 内部管理请求和传输，所以在此模式下不检测溢出情况）
  - 当 FIFO 阈值级别与存储器突发大小不兼容时使能流（请参见*表 30：FIFO 阈值配置*）
- **直接模式错误**：只有当在直接模式下工作并且已将 DMA_SxCR 寄存器中的 MINC 位清零时，才能在外设到存储器模式下将直接模式错误中断标志 (DMEIFx) 置 1。当在先前数据未完全传输到存储器（因为存储器总线未得到授权）的情况下发生 DMA 请求时，该标志将置 1。在这种情况下，该标志指示有两个数据项相继传输到相同的目标地址，如果目标不能管理这种情况，会发生问题。

在直接模式下，如果出现下列条件，FIFO 错误标志也会置 1：

- 在外设到存储器模式下，如果未对存储器总线授权支持多个外设请求，FIFO 可能饱和（上溢）
- 在存储器到外设模式下，如果在外设请求发生前存储器总线未得到授权，则可能发生下溢的情况

如果由于突发大小与 FIFO 阈值级别之间的不兼容而引起 TEIFx 或 FEIFx 标志置 1，则硬件自动将相应数据流配置寄存器 (DMA_SxCR) 中的 EN 位清零，从而禁止错误的数据流。

如果由于上溢或下溢情况引起 DMEIFx 或 FEIFx 标志置 1，则不会自动禁止错误的数据流，而是由软件决定是否通过重置 DMA_SxCR 寄存器中的 EN 位禁止数据流。这是因为当发生这种错误时没有数据丢失。

当 DMA_LISR 或 DMA_HISR 寄存器中数据流的错误中断标志（TEIF、FEIF、DMEIF）置 1 时，如果 DMA_SxCR 或 DMA_SxFCR 寄存器中相应的中断使能位（TEIE、FEIE、DMIE）也置 1，则会生成一个中断。

注： *当 FIFO 上溢或下溢的情况发生时，因为直到上溢或下溢的情况被清除，数据流才会确认外设请求，所以数据不会丢失。如果此确认过程花费过多时间，则外设本身会检测到其内部缓冲器上溢或下溢的情况，数据可能丢失。*

8.4 DMA 中断

对于每个 DMA 数据流，可在发生以下事件时产生中断：

- 达到半传输
- 传输完成
- 传输错误
- FIFO 错误（上溢、下溢或 FIFO 级别错误）
- 直接模式错误

可以使用单独的中断使能位以实现灵活性，如*表 32* 所示。

表 32. DMA 中断请求

中断事件	事件标志	使能控制位
半传输	HTIF	HTIE
传输完成	TCIF	TCIE
传输错误	TEIF	TEIE
FIFO 上溢/下溢	FEIF	FEIE
直接模式错误	DMEIF	DMEIE

注： *在将使能控制位置 '1' 前，应将相应的事件标志清零，否则会立即产生中断。*

8.5 DMA 寄存器

DMA 寄存器必须按字（32 位）进行访问。

8.5.1 DMA 低中断状态寄存器 (DMA_LISR)

DMA low interrupt 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.	TCIF3	HTIF3	TEIF3	DMEIF3	Res.	FEIF3	TCIF2	HTIF2	TEIF2	DMEIF2	Res.	FEIF2
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.	TCIF1	HTIF1	TEIF1	DMEIF1	Res.	FEIF1	TCIF0	HTIF0	TEIF0	DMEIF0	Res.	FEIF0
r	r	r	r	r	r	r	r		r	r	r	r	r		r

位 31:28、15:12 保留，必须保持复位值。

位 27、21、11、5 **TCIFx**：数据流 x 传输完成中断标志 (Stream x transfer complete interrupt flag) (x = 3..0)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_LIFCR 寄存器的相应位。
0：数据流 x 上无传输完成事件
1：数据流 x 上发生传输完成事件

位 26、20、10、4 **HTIFx**：数据流 x 半传输中断标志 (Stream x half transfer interrupt flag) (x=3..0)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_LIFCR 寄存器的相应位。
0：数据流 x 上无半传输事件
1：数据流 x 上发生半传输事件

位 25、19、9、3 **TEIFx**：数据流 x 传输错误中断标志 (Stream x transfer error interrupt flag) (x=3..0)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_LIFCR 寄存器的相应位。
0：数据流 x 上无传输错误
1：数据流 x 上发生传输错误

位 24、18、8、2 **DMEIFx**：数据流 x 直接模式错误中断标志 (Stream x direct mode error interrupt flag) (x=3..0)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_LIFCR 寄存器的相应位。
0：数据流 x 上无直接模式错误
1：数据流 x 上发生直接模式错误

位 23、17、7、1 保留，必须保持复位值。

位 22、16、6、0 **FEIFx**：数据流 x FIFO 错误中断标志 (Stream x FIFO error interrupt flag) (x=3..0)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_LIFCR 寄存器的相应位。
0：数据流 x 上无 FIFO 错误事件
1：数据流 x 上发生 FIFO 错误事件

8.5.2 DMA 高中断状态寄存器 (DMA_HISR)

DMA high interrupt status 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.	TCIF7	HTIF7	TEIF7	DMEIF7	Res.	FEIF7	TCIF6	HTIF6	TEIF6	DMEIF6	Res.	FEIF6
				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.	TCIF5	HTIF5	TEIF5	DMEIF5	Res.	FEIF5	TCIF4	HTIF4	TEIF4	DMEIF4	Res.	FEIF4
				r	r	r	r		r	r	r	r	r		r

位 31:28、15:12 保留，必须保持复位值。

位 27、21、11、5 **TCIFx**：数据流 x 传输完成中断标志 (Stream x transfer complete interrupt flag) (x=7..4)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_HIFCR 寄存器的相应位。
0：数据流 x 上无传输完成事件
1：数据流 x 上发生传输完成事件

位 26、20、10、4 **HTIFx**：数据流 x 半传输中断标志 (Stream x half transfer interrupt flag) (x=7..4)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_HIFCR 寄存器的相应位。
0：数据流 x 上无半传输事件
1：数据流 x 上发生半传输事件

位 25、19、9、3 **TEIFx**：数据流 x 传输错误中断标志 (Stream x transfer error interrupt flag) (x=7..4)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_HIFCR 寄存器的相应位。
0：数据流 x 上无传输错误
1：数据流 x 上发生传输错误

位 24、18、8、2 **DMEIFx**：数据流 x 直接模式错误中断标志 (Stream x direct mode error interrupt flag) (x=7..4)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_HIFCR 寄存器的相应位。
0：数据流 x 上无直接模式错误
1：数据流 x 上发生直接模式错误

位 23、17、7、1 保留，必须保持复位值。

位 22、16、6、0 **FEIFx**：数据流 x FIFO 错误中断标志 (Stream x FIFO error interrupt flag) (x=7..4)
此位将由硬件置 1， 由软件清零，软件只需将 1 写入 DMA_HIFCR 寄存器的相应位。
0：数据流 x 上无 FIFO 错误事件
1：数据流 x 上发生 FIFO 错误事件

8.5.3 DMA 低中断标志清零寄存器 (DMA_LIFCR)

DMA low interrupt flag clear 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.	CTCIF3	CHTIF3	CTEIF3	CDMEIF3	Res.	CFEIF3	CTCIF2	CHTIF2	CTEIF2	CDMEIF2	Res.	CFEIF2
				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.	CTCIF1	CHTIF1	CTEIF1	CDMEIF1	Res.	CFEIF1	CTCIF0	CHTIF0	CTEIF0	CDMEIF0	Res.	CFEIF0
				w	w	w	w		w	w	w	w	w		w

位 31:28、15:12 保留，必须保持复位值。

位 27、21、11、5 **CTCIFx**：数据流 x 传输完成中断标志清零 (Stream x clear transfer complete interrupt flag) (x = 3..0)
将 1 写入此位时，DMA_LISR 寄存器中相应的 TCIFx 标志将清零

位 26、20、10、4 **CHTIFx**：数据流 x 半传输中断标志清零 (Stream x clear half transfer interrupt flag) (x = 3..0)
将 1 写入此位时，DMA_LISR 寄存器中相应的 HTIFx 标志将清零

位 25、19、9、3 **CTEIFx**：数据流 x FIFO 传输错误中断标志清零 (Stream x clear transfer error interrupt flag) (x = 3..0)
将 1 写入此位时，DMA_LISR 寄存器中相应的 TEIFx 标志将清零

位 24、18、8、2 **CDMEIFx**：数据流 x 直接模式错误中断标志清零 (Stream x clear direct mode error interrupt flag) (x = 3..0)
将 1 写入此位时，DMA_LISR 寄存器中相应的 DMEIFx 标志将清零

位 23、17、7、1 保留，必须保持复位值。

位 22、16、6、0 **CFEIFx**：数据流 x FIFO 错误中断标志清零 (Stream x clear FIFO error interrupt flag) (x = 3..0)
将 1 写入此位时，DMA_LISR 寄存器中相应的 CFEIFx 标志将清零

8.5.4 DMA 高中断标志清零寄存器 (DMA_HIFCR)

DMA high 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.	CTCIF7	CHTIF7	CTEIF7	CDMEIF7	Res.	CFEIF7	CTCIF6	CHTIF6	CTEIF6	CDMEIF6	Res.	CFEIF6
				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.	CTCIF5	CHTIF5	CTEIF5	CDMEIF5	Res.	CFEIF5	CTCIF4	CHTIF4	CTEIF4	CDMEIF4	Res.	CFEIF4
				w	w	w	w		w	w	w	w	w		w

位 31:28、15:12 保留，必须保持复位值。

位 27、21、11、5 **CTCIFx**：数据流 x 传输完成中断标志清零 (Stream x clear transfer complete interrupt flag) (x = 7..4)

将 1 写入此位时，DMA_HISR 寄存器中相应的 TCIFx 标志将清零

位 26、20、10、4 **CHTIFx**：数据流 x 半传输中断标志清零 (Stream x clear half transfer interrupt flag) (x = 7..4)

将 1 写入此位时，DMA_HISR 寄存器中相应的 HTIFx 标志将清零

位 25、19、9、3 **CTEIFx**：数据流 x FIFO 传输错误中断标志清零 (Stream x clear transfer error interrupt flag) (x = 7..4)

将 1 写入此位时，DMA_HISR 寄存器中相应的 TEIFx 标志将清零

位 24、18、8、2 **CDMEIFx**：数据流 x 直接模式错误中断标志清零 (Stream x clear direct mode error interrupt flag) (x = 7..4)

将 1 写入此位时，DMA_HISR 寄存器中相应的 DMEIFx 标志将清零

位 23、17、7、1 保留，必须保持复位值。

位 22、16、6、0 **CFEIFx**：数据流 x FIFO 错误中断标志清零 (Stream x clear FIFO error interrupt flag) (x = 7..4)

将 1 写入此位时，DMA_HISR 寄存器中相应的 CFEIFx 标志将清零

8.5.5 DMA 数据流 x 配置寄存器 (DMA_SxCR) (x = 0..7)

DMA stream x configuration register

此寄存器用于配置相关数据流。

偏移地址：0x10 + 0x18 × *数据流编号*

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	CHSEL[3:0]			MBURST [1:0]		PBURST[1:0]		Res.	CT	DBM or reserved	PL[1:0]	
				rw	rw	rw	rw	rw	rw	rw		rw	rw or r	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR[1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
rw	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 27:25 **CHSEL[2:0]**：通道选择 (Channel selection)

这些位将由软件置 1 和清零。

000：选择通道 0
001：选择通道 1
010：选择通道 2
011：选择通道 3
100：选择通道 4
101：选择通道 5
110：选择通道 6
111：选择通道 7

这些位受到保护，只有 EN 为“0”时才可以写入

位 24:23 **MBURST**：存储器突发传输配置 (Memory burst transfer configuration)

这些位将由软件置 1 和清零。

00：单次传输
01：INCR4（4 个节拍的增量突发传输）
10：INCR8（8 个节拍的增量突发传输）
11：INCR16（16 个节拍的增量突发传输）

这些位受到保护，只有 EN 为“0”时才可以写入

在直接模式中，当位 EN =“1”时，这些位由硬件强制置为 0x0。

位 22:21 **PBURST[1:0]**：外设突发传输配置 (Peripheral burst transfer configuration)

这些位将由软件置 1 和清零。

00：单次传输
01：INCR4（4 个节拍的增量突发传输）
10：INCR8（8 个节拍的增量突发传输）
11：INCR16（16 个节拍的增量突发传输）

这些位受到保护，只有 EN 为“0”时才可以写入

在直接模式下，这些位由硬件强制置为 0x0。

位 20 保留，必须保持复位值。

位 19 **CT**：当前目标（仅在双缓冲区模式下）(Current target (only in double buffer mode))
此位由硬件置 1 和清零， 也可由软件写入。
0：当前目标存储器为存储器 0（使用 DMA_SxM0AR 指针寻址）
1：当前目标存储器为存储器 1（使用 DMA_SxM1AR 指针寻址）
只有 EN 为“0”时，此位才可以写入，以指示第一次传输的目标存储区。在使能数据流后，此位相当于一个状态标志，用于指示作为当前目标的存储区。

位 18 **DBM**：双缓冲区模式 (Double buffer mode)
此位由软件置 1 和清零。
0：传输结束时不切换缓冲区
1：DMA 传输结束时切换目标存储区
此位受到保护，只有 EN 为“0”时才可以写入。

位 17:16 **PL[1:0]**：优先级 (Priority level)
这些位将由软件置 1 和清零。
00：低
01：中
10：高
11：非常高
这些位受到保护，只有 EN 为“0”时才可以写入。

位 15 **PINCOS**：外设增量偏移量 (Peripheral increment offset size)
此位由软件置 1 和清零
0：用于计算外设地址的偏移量与 PSIZE 相关
1：用于计算外设地址的偏移量固定为 4（32 位对齐）。
如果位 PINC =“0”，则此位没有意义。
此位受到保护，只有 EN 为“0”时才可以写入。
如果选择直接模式或者 PBURST 不等于“00”，则当使能数据流（位 EN =“1”）时，此位由硬件强制置为低电平。

位 14:13 **MSIZE[1:0]**：存储器数据大小 (Memory data size)
这些位将由软件置 1 和清零。
00：字节（8 位）
01：半字（16 位）
10：字（32 位）
11：保留
这些位受到保护，只有 EN 为“0”时才可以写入。
在直接模式下，当位 EN =“1”时，MSIZE 位由硬件强制置为与 PSIZE 相同的值。

位 12:11 **PSIZE[1:0]**：外设数据大小 (Peripheral data size)
这些位将由软件置 1 和清零。
00：字节（8 位）
01：半字（16 位）
10：字（32 位）
11：保留
这些位受到保护，只有 EN 为“0”时才可以写入

位 10 **MINC**：存储器递增模式 (Memory increment mode)
此位由软件置 1 和清零。
0：存储器地址指针固定
1：每次数据传输后，存储器地址指针递增（增量为 MSIZE 值）
此位受到保护，只有 EN 为“0”时才可以写入。

位 9 **PINC**：外设递增模式 (Peripheral increment mode)
此位由软件置 1 和清零。
0：外设地址指针固定
1：每次数据传输后，外设地址指针递增（增量为 PSIZE 值）
此位受到保护，只有 EN 为“0”时才可以写入。

位 8 **CIRC**：循环模式 (Circular mode)
此位由软件置 1 和清零，并可由硬件清零。
0：禁止循环模式
1：使能循环模式
如果外设为流控制器（位 PFCTRL=1）且使能数据流（位 EN=1），此位由硬件自动强制清零。
如果 DBM 位置 1，当使能数据流（位 EN =“1”）时，此位由硬件自动强制置 1。

位 7:6 **DIR[1:0]**：数据传输方向 (Data transfer direction)
这些位将由软件置 1 和清零。
00：外设到存储器
01：存储器到外设
10：存储器到存储器
11：保留
这些位受到保护，只有 EN 为“0”时才可以写入。

位 5 **PFCTRL**：外设流控制器 (Peripheral flow controller)
此位由软件置 1 和清零。
0：DMA 是流控制器
1：外设是流控制器
此位受到保护，只有 EN 为“0”时才可以写入。
选择存储器到存储器模式（位 DIR[1:0]=10）后，此位由硬件自动强制清零。

位 4 **TCIE**：传输完成中断使能 (Transfer complete interrupt enable)
此位由软件置 1 和清零。
0：禁止 TC 中断
1：使能 TC 中断

位 3 **HTIE**：半传输中断使能 (Half transfer interrupt enable)
此位由软件置 1 和清零。
0：禁止 HT 中断
1：使能 HT 中断

位 2 **TEIE**：传输错误中断使能 (Transfer error interrupt enable)
此位由软件置 1 和清零。
0：禁止 TE 中断
1：使能 TE 中断

位 1 **DMEIE**：直接模式错误中断使能 (Direct mode error interrupt enable)
此位由软件置 1 和清零。
0：禁止 DME 中断
1：使能 DME 中断

位 0 **EN**：数据流使能 / 读作低电平时数据流就绪标志 (Stream enable / flag stream ready when read low)

此位由软件置 1 和清零。

0：禁止数据流

1：使能数据流

以下情况下，此位可由硬件清零：

– DMA 传输结束时（准备好配置数据流）

– AHB 主总线出现传输错误时

– 存储器 AHB 端口上的 FIFO 阈值与突发大小不兼容时

此位读作 0 时，软件可以对配置和 FIFO 位寄存器编程。EN 位读作 1 时，禁止向这些寄存器执行写操作。

注： 将 EN 位置“1”以启动新传输之前，DMA_LISR 或 DMA_HISR 寄存器中与数据流相对应的事件标志必须清零。

8.5.6 DMA 数据流 x 数据项数寄存器 (DMA_SxNDTR) (x = 0..7)

DMA stream x number of data register

偏移地址：0x14 + 0x18 × *数据流编号*

复位值：0x0000 0000

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

位 31:16 保留，必须保持复位值。

位 15:0 **NDT[15:0]**：要传输的数据项数目 (Number of data items to transfer)

要传输的数据项数目（0 到 65535）。只有在禁止数据流时，才能向此寄存器执行写操作。使能数据流后，此寄存器为只读，用于指示要传输的剩余数据项数。每次 DMA 传输后，此寄存器将递减。

传输完成后，此寄存器保持为零（数据流处于正常模式时），或者在以下情况下自动以先前编程的值重载：

– 以循环模式配置数据流时。

– 通过将 EN 位置“1”来重新使能数据流时。

如果该寄存器的值为零，则即使使能数据流，也无法完成任何事务。

8.5.7 DMA 数据流 x 外设地址寄存器 (DMA_SxPAR) (x = 0..7)

DMA stream x peripheral address register

偏移地址：0x18 + 0x18 × *数据流编号*

复位值：0x0000 0000

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

位 31:0 **PAR[31:0]**：外设地址 (Peripheral address)

读/写数据的外设数据寄存器的基址。

这些位受到写保护，只有 DMA_SxCR 寄存器中的 EN 为“0”时才可以写入。

8.5.8 DMA 数据流 x 存储器 0 地址寄存器 (DMA_SxM0AR) (x = 0..7)

DMA stream x memory 0 address register

偏移地址：0x1C + 0x18 × *数据流编号*

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
M0A[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
M0A[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **M0A[31:0]**：存储器 0 地址 (Memory 0 address)

读/写数据的存储区 0 的基址。

这些位受到写保护， 只有在以下情况下才可以写入：

– 禁止数据流（DMA_SxCR 寄存器中的位 EN=“0”）或

– 使能数据流（DMA_SxCR 寄存器中的 EN=“1”）并且 DMA_SxCR 寄存器中的位 CT =“1”（在双缓冲区模式下）。

8.5.9 DMA 数据流 x 存储器 1 地址寄存器 (DMA_SxM1AR) (x = 0..7)

DMA stream x memory 1 address register

偏移地址：0x20 + 0x18 × *数据流编号*

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
M1A[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
M1A[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **M1A[31:0]**：存储器 1 地址（用于双缓冲区模式）(Memory 1 address (used in case of Double buffer mode))

读/写数据的存储区 1 的基址。

此寄存器仅用于双缓冲区模式。

这些位受到写保护， 只有在以下情况下才可以写入：

- 禁止数据流（DMA_SxCR 寄存器中的位 EN=“0”）或
- 使能数据流（DMA_SxCR 寄存器中的 EN=“1”）并且 DMA_SxCR 寄存器中的位 CT =“0”。

8.5.10 DMA 数据流 x FIFO 控制寄存器 (DMA_SxFCR) (x = 0..7)

DMA stream x FIFO control register

偏移地址：0x24 + 0x24 × *数据流编号*

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.	FEIE	Res.	FS[2:0]			DMDIS	FTH[1:0]	
								rw		r	r	r	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7 **FEIE**：FIFO 错误中断使能 (FIFO error interrupt enable)

此位由软件置 1 和清零。

0：禁止 FE 中断

1：使能 FE 中断

位 6 保留，必须保持复位值。

位 5:3 **FS[2:0]**：FIFO 状态 (FIFO status)

这些位为只读。

000：0 < fifo_level < 1/4

001：1/4 ≤ fifo_level < 1/2

010：1/2 ≤ fifo_level < 3/4

011：3/4 ≤ fifo_level < 满

100：FIFO 为空

101：FIFO 已满

其它：无意义

在直接模式（DMDIS 位为零）下，这些位无意义。

位 2 **DMDIS**：直接模式禁止 (Direct mode disable)

此位由软件置 1 和清零。它可由硬件置 1。

0：使能直接模式

1：禁止直接模式

此位受到保护，只有 EN 为“0”时才可以写入。

如果选择存储器到存储器模式（DMA_SxCR 中的 DIR 位为“10”），并且 DMA_SxCR 寄存器中的 EN 位为“1”，则此位由硬件置 1，因为在存储器到存储器配置不能使用直接模式。

位 1:0 **FTH[1:0]**：FIFO 阈值选择 (FIFO threshold selection)

这些位将由软件置 1 和清零。

00：FIFO 容量的 1/4

01：FIFO 容量的 1/2

10：FIFO 容量的 3/4

11：FIFO 完整容量

在直接模式（DMDIS 值为零）下，不使用这些位。

这些位受到保护，只有 EN 为“0”时才可以写入。

8.5.11 DMA 寄存器映射

表 33 汇总了 DMA 寄存器。

表 33. DMA 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
0x0000	**DMA_LISR**	Res.	Res.	Res.	Res.	TCIF3	HTIF3	TEIF3	DMEIF3	Res.	FEIF3	TCIF2	HTIF2	TEIF2	DMEIF2	Res.	FEIF2	Res.	Res.	Res.	Res.	TCIF1	HTIF1	TEIF1	DMEIF1	Res.	FEIF1	TCIF0	HTIF0	TEIF0	DMEIF0	Reserved	FEIF0
	Reset value					0	0	0	0		0	0	0	0	0		0					0	0	0	0		0	0	0	0	0		0
0x0004	**DMA_HISR**	Res	Res	Res	Res	TCIF7	HTIF7	TEIF7	DMEIF7	Res	FEIF7	TCIF6	HTIF6	TEIF6	DMEIF6	Res	FEIF6	Res	Res	Res	Res	TCIF5	HTIF5	TEIF5	DMEIF5	Res	FEIF5	TCIF4	HTIF4	TEIF4	DMEIF4	Reserved	FEIF4
	Reset value					0	0	0	0		0	0	0	0	0		0					0	0	0	0		0	0	0	0	0		0
0x0008	**DMA_LIFCR**	Res	Res	Res	Res	CTCIF3	CHTIF3	TEIF3	CDMEIF3	Reserved	CFEIF3	CTCIF2	CHTIF2	CTEIF2	CDMEIF2	Res	CFEIF2	Res	Res	Res	Res	CTCIF1	CHTIF1	CTEIF1	CDMEIF1	Res	CFEIF1	CTCIF0	CHTIF0	CTEIF0	CDMEIF0	Reserved	CFEIF0
	Reset value					0	0	0	0		0	0	0	0	0		0					0	0	0	0		0	0	0	0	0		0
0x000C	**DMA_HIFCR**	Res	Res	Res	Res	CTCIF7	CHTIF7	CTEIF7	CDMEIF7	Reserved	CFEIF7	CTCIF6	CHTIF6	CTEIF6	CDMEIF6	Res	CFEIF6	Res	Res	Res	Res	CTCIF5	CHTIF5	CTEIF5	CDMEIF5	Res	CFEIF5	CTCIF4	CHTIF4	CTEIF4	CDMEIF4	Reserved	CFEIF4
	Reset value	Res	Res	Res	Res	0	0	0	0		0	0	0	0	0	Res	0					0	0	0	0		0	0	0	0	0		0
0x0010	**DMA_S0CR**	Res	Res	Res	Res	CHSEL[2:0]			MBURST[1:0]		PBURST[1:0]		Res	CT	DBM	PL[1:0]		PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR[1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
	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
0x0014	**DMA_S0NDTR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:.]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0018	**DMA_S0PAR**	PA[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
0x001C	**DMA_S0M0AR**	M0A[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
0x0020	**DMA_S0M1AR**	M1A[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
0x0024	**DMA_S0FCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FEIE	Res	FS[2:0]			DMDIS	FTH [1:0]	
	Reset value																									0		1	0	0	0	0	1
0x0028	**DMA_S1CR**	Res	Res	Res	Res	CHSEL [2:0]			MBURST[1:]		PBURST[1:0]		Res	CT	DBM	PL[1:0]		PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR[1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
	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
0x002C	**DMA_S1NDTR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:.]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 33. DMA 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
0x0030	**DMA_S1PAR**	PA[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
0x0034	**DMA_S1M0A**R	M0A[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
0x0038	**DMA_S1M1AR**	M1A[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
0x003C	**DMA_S1FCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FEIE	Res	FS[2:0]			DMDIS	FTH [1:0]	
	Reset value																									0		1	0	0	0	0	1
0x0040	**DMA_S2CR**	Res	Res	Res	Res	CHSEL [2:0]			MBURST[1:0]		PBURST[1:0]		Res	CT	DBM	PL[1:0]		PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR [1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
	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
0x0044	**DMA_S2NDTR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:.]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0048	**DMA_S2PAR**	PA[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
0x004C	**DMA_S2M0AR**	M0A[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
0x0050	**DMA_S2M1AR**	M1A[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
0x0054	**DMA_S2FCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FEIE	Res	FS[2:0]			DMDIS	FTH [1:0]	
	Reset value																									0		1	0	0	0	0	1
0x0058	**DMA_S3CR**	Res	Res	Res	Res	CHSEL[2:0]			MBURST[1:0]		PBURST[1:0]		Res	CT	DBM	PL[1:0]		PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR[1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
	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
0x005C	**DMA_S3NDTR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:.]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0060	**DMA_S3PAR**	PA[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
0x0064	**DMA_S3M0AR**	M0A[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 33. DMA 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
0x0068	**DMA_S3M1AR**	M1A[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
0x006C	**DMA_S3FCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FEIE	Res	FS[2:0]			DMDIS	FTH [1:0]	
	Reset value																									0		1	0	0	0	0	1
0x0070	**DMA_S4CR**	Res	Res	Res	Res	CHSEL[2:0]			MBURST[1:0]		PBURST[1:0]		Res	CT	DBM	PL[1:0]		PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR [1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
	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
0x0074	**DMA_S4NDTR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:.]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0078	**DMA_S4PAR**	PA[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
0x007C	**DMA_S4M0AR**	M0A[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
0x0080	**DMA_S4M1AR**	M1A[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
0x0084	**DMA_S4FCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FEIE	Res	FS[2:0]			DMDIS	FTH [1:0]	
	Reset value																									0		1	0	0	0	0	1
0x0088	**DMA_S5CR**	Res	Res	Res	Res	CHSEL[2:0]			MBURST[1:0]		PBURST[1:0]		Res	CT	DBM	PL[1:0]		PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR[1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
	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
0x008C	**DMA_S5NDTR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:.]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0090	**DMA_S5PAR**	PA[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
0x0094	**DMA_S5M0AR**	M0A[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
0x0098	**DMA_S5M1AR**	M1A[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
0x009C	**DMA_S5FCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FEIE	Res	FS[2:0]			DMDIS	FTH [1:0]	
	Reset value																									0		1	0	0	0	0	1
0x00A0	**DMA_S6CR**	Res	Res	Res	Res	CHSEL[2:0]			MBURST[1:0]		PBURST[1:0]		Res	CT	DBM	PL[1:0]		PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR[1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
	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
0x00A4	**DMA_S6NDTR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:.]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00A8	**DMA_S6PAR**	PA[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 33. DMA 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
0x00AC	**DMA_S6M0AR**	M0A[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
0x00B0	**DMA_S6M1AR**	M1A[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
0x00B4	**DMA_S6FCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FEIE	Res	FS[2:0]			DMDIS	FTH [1:0]	
	Reset value																									0		1	0	0	0	0	1
0x00B8	**DMA_S7CR**	Res	Res	Res	Res	CHSEL[2:0]			MBURST[1:0]		PBURST[1:0]		Res	CT	DBM	PL[1:0]		PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR[1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
	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
0x00BC	**DMA_S7NDTR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:.]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00C0	**DMA_S7PAR**	PA[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
0x00C4	**DMA_S7M0AR**	M0A[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
0x00C8	**DMA_S7M1AR**	M1A[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
0x00CC	**DMA_S7FCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FEIE	Res	FS[2:0]			DMDIS	FTH [1:0]	
	Reset value																									0		1	0	0	0	0	1

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

9 Chrom-Art Accelerator™ 控制器 (DMA2D)

9.1 DMA2D 简介

Chrom-Art Accelerator™ (DMA2D) 是专用于图像处理的专业 DMA。它可以执行下列操作：

- 用特定颜色填充目标图像的一部分或全部
- 将源图像的一部分（或全部）复制到目标图像的一部分（或全部）中
- 通过像素格式转换将源图像的一部分或全部复制到目标图像的一部分或全部中
- 将像素格式不同的两个源图像部分和 / 或全部混合，再将结果复制到颜色格式不同的部分或整个目标图像中。

在索引颜色模式或直接颜色模式下，支持所有经典颜色编码方案，并支持每像素 4 位到最高 32 位。DMA2D 自身具有专门的 CLUT（颜色查找表）存储器。

9.2 DMA2D 的主要特性

DMA2D 的主要特性有：

- 采用单 AHB 主设备总线架构
- AHB 从设备编程接口支持 8/16/32 位访问 （32 位的 CLUT 访问除外）
- 用户可编程工作区大小
- 用户可编程源区域和目标区域的偏移
- 用户可编程整个存储空间的源地址和目标地址
- 最多支持 2 个源的混合操作
- Alpha 值可修改（源值、固定值或调制的值）
- 用户可编程源颜色格式和目标的颜色格式
- 采用间接或直接颜色编码时，支持多达 11 种颜色格式，且支持每像素 4 位到最高 32 位
- 间接颜色模式下使用 2 个内部存储器存储 CLUT
- 通过 CPU 自动加载 CLUT 或对 CLUT 进行编程
- 用户可编程 CLUT 大小
- 使用内部定时器控制 AHB 带宽
- 支持 4 种工作模式：寄存器到存储器、存储器到存储器、存储器到存储器且支持像素格式转换和存储器到存储器且支持像素格式转换和混合
- 可使用固定颜色进行区域填充
- 可从一个区域复制到另一个区域
- 在源图像和目标图像之间进行复制时进行像素格式转换
- 支持从颜色格式不同的两幅源图像复制并混合
- 可中止并挂起 DMA2D 操作
- 支持在传输用户可编程的目标行时生成水印中断
- 支持发生总线错误或访问冲突时生成中断
- 支持处理完成时生成中断

9.3 DMA2D 功能说明

9.3.1 概述

DMA2D 控制器执行直接存储器传输。作为一个 AHB 主设备，它可以控制 AHB 总线矩阵来启动 AHB 事务。

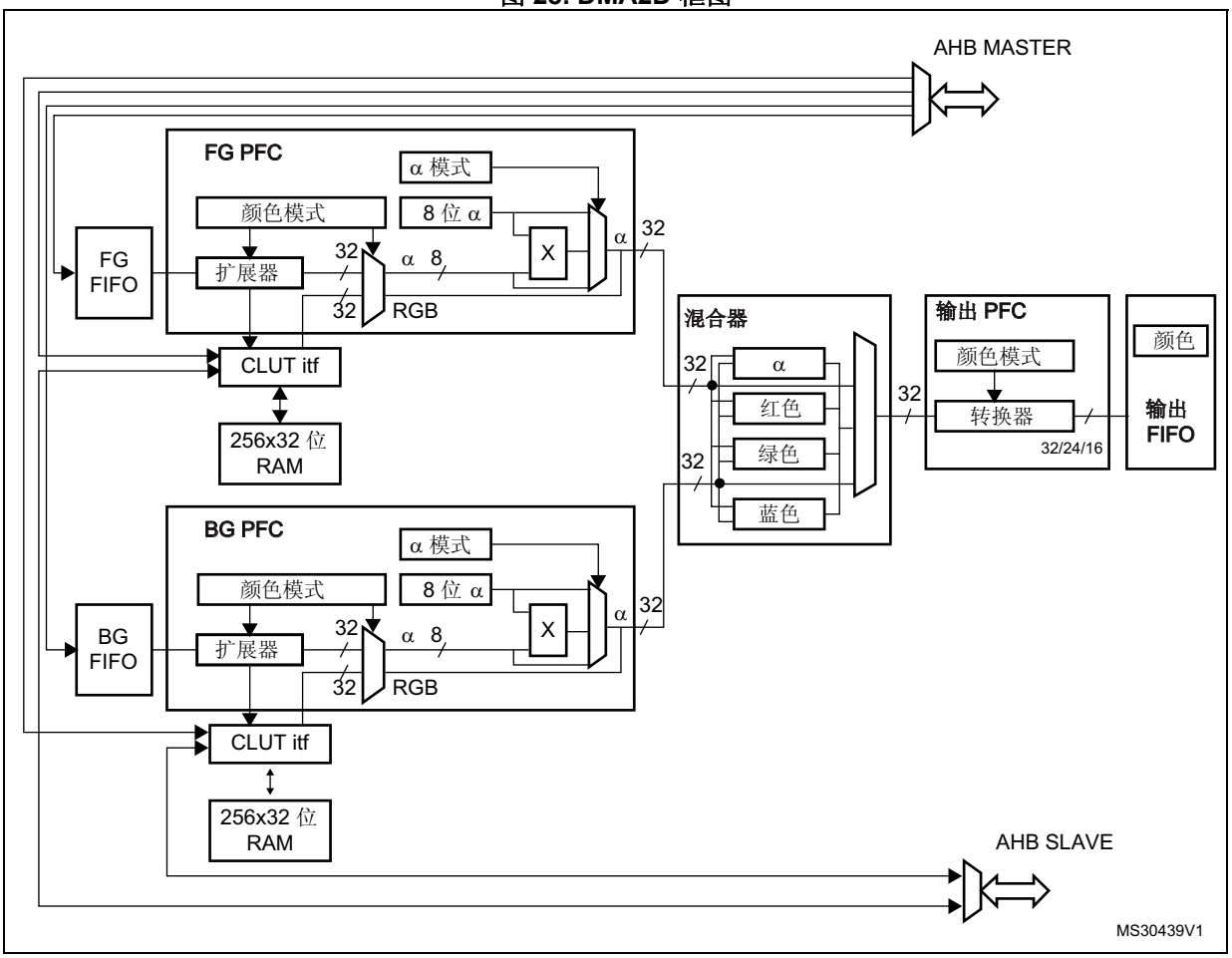
DMA2D 可在以下四种模式下工作：

- 寄存器到存储器
- 存储器到存储器
- 存储器到存储器并执行像素格式转换
- 存储器到存储器并执行像素格式转换和混合

AHB 从设备端口用于编程 DMA2D 控制器。

DMA2D 的框图如*图 28：DMA2D 框图*所示。

图 28. DMA2D 框图



9.3.2 DMA2D 控制

通过 DMA2D 控制寄存器 (DMA2D_CR) 配置 DMA2D 控制器，允许选择：

用户应用可以执行下列操作：

- 选择工作模式
- 使能/禁止 DMA2D 中断
- 启动/挂起/中止进行中的数据传输

9.3.3 DMA2D 前景层 FIFO 和背景层 FIFO

DMA2D 前景层 (FG) FG FIFO 和背景层 (BG) FIFO 获取要复制和/或处理的输入数据。

这些 FIFO 根据相应像素格式转换器 (PFC) 中定义的颜色格式获取像素。

通过如下一组寄存器对它们进行编程：

- DMA2D 前景层存储器地址寄存器 (DMA2D_FGMAR)
- DMA2D 前景层偏移寄存器 (DMA2D_FGOR)
- DMA2D 背景层存储器地址寄存器 (DMA2D_BGMAR)
- DMA2D 背景层偏移寄存器 (DMA2D_BGBOR)
- DMA2D 行数寄存器（行数和每行像素数）(DMA2D_NLR)

DMA2D 在寄存器到存储器模式下工作时，不激活任何 FIFO。

DMA2D 在存储器到存储器模式下工作时（无像素格式转换和混合操作），仅激活 FG FIFO，并将其用作缓冲区。

DMA2D 在存储器到存储器模式下工作时并支持像素格式转换时（无混合操作），不会激活 BG FIFO。

9.3.4 DMA2D 前景层和背景层像素格式转换器 (PFC)

DMA2D 前景层和背景层像素格式转换器 (PFC) 执行像素格式转换，以生成每像素 32 位的值。PFC 还能够修改 alpha 通道。

转换器在第一阶段转换颜色格式。前景层像素和背景层像素的原始颜色格式分别通过 DMA2D_FGPFCCR 和 DMA2D_BGPFCCR 的 CM[3:0] 位来配置。

*表 34：输入时支持的颜色模式*给出了支持的输入格式。

表 34. 输入时支持的颜色模式

CM[3:0]	颜色模式
0000	ARGB8888
0001	RGB888
0010	RGB565
0011	ARGB1555
0100	ARGB4444
0101	L8
0110	AL44
0111	AL88

表 34. 输入时支持的颜色模式（续）

CM[3:0]	颜色模式
1000	L4
1001	A8
1010	A4

颜色格式的编码方式如下：

- Alpha 值字段：透明
  0xFF 值对应不透明像素，0x00 对应透明像素。
- R 字段代表红色
- G 字段代表绿色
- B 字段代表蓝色
- L 字段：亮度
  该字段是 CLUT 的索引，用于检索三个/四个 RGB/ARGB 分量。

如果原始格式为直接颜色模式，则通过将 MSB 复制到 LSB 扩展为每通道 8 位。这可以确保转换具有良好的线性。

如果原始格式不包括 alpha 通道，则会自动将 alpha 值设为 0xFF（不透明）。

如果原始格式为间接颜色模式，则需要使用 CLUT，并且每个像素格式转换器与一个 256 个 32 位 条目的 CLUT 相关联。

对于特定的 alpha 模式 A4 和 A8，既不存储颜色信息，也不编制索引。用于生成图像的颜色是固定的，并且在 DMA2D_FGCOLR 寄存器中定义前景层像素的颜色，在 DMA2D_BGCOLR 寄存器中定义背景层像素的颜色。

系统存储器中的字段顺序如*表 35：存储器中的数据顺序*所示。

表 35. 存储器中的数据顺序

颜色模式	@ + 3	@ + 2	@ + 1	@ + 0
ARGB8888	A_0[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
RGB888	B_1[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
	G_2[7:0]	B_2[7:0]	R_1[7:0]	G_1[7:0]
	R_3[7:0]	G_3[7:0]	B_3[7:0]	R_2[7:0]
RGB565	R_1[4:0]G_1[5:3]	G_1[2:0]B_1[4:0]	R_0[4:0]G_0[5:3]	G_0[2:0]B_0[4:0]
ARGB1555	A_1[0]R_1[4:0]G_1[4:3]	G_1[2:0]B_1[4:0]	A_0[0]R_0[4:0]G_0[4:3]	G_0[2:0]B_0[4:0]
ARGB4444	A_1[3:0]R_1[3:0]	G_1[3:0]B_1[3:0]	A_0[3:0]R_0[3:0]	G_0[3:0]B_0[3:0]
L8	L_3[7:0]	L_2[7:0]	L_1[7:0]	L_0[7:0]
AL44	A_3[3:0]L_3[3:0]	A_2[3:0]L_2[3:0]	A_1[3:0]L_1[3:0]	A_0[3:0]L_0[3:0]
AL88	A_1[7:0]	L_1[7:0]	A_0[7:0]	L_0[7:0]
L4	L_7[3:0]L_6[3:0]	L_5[3:0]L_4[3:0]	L_3[3:0]L_2[3:0]	L_1[3:0]L_0[3:0]
A8	A_3[7:0]	A_2[7:0]	A_1[7:0]	A_0[7:0]
A4	A_7[3:0]A_6[3:0]	A_5[3:0]A_4[3:0]	A_3[3:0]A_2[3:0]	A_1[3:0]A_0[3:0]

通过 ARGB8888 模式支持按 32 位对齐 24 位 RGB888。

生成 32 位值后，即可根据 DMA2D_FGPFCCR/DMA2D_BGPFCCR 寄存器的 AM[1:0] 字段修改 alpha 通道，如*表 36：Alpha 模式配置*所示。

Alpha 通道可以：

- 保持不变（不做修改），
- 替换为 DMA2D_FGPFCCR/DMA2D_BGPFCCR 的 ALPHA[7:0] 值。
- 或替换为原始 alpha 值与 DMA2D_FGPFCCR/DMA2D_BGPFCCR 的 ALPHA [7:0] 值除以 255 所得商的乘积。

表 36. Alpha 模式配置

AM[1:0]	Alpha 模式
00	不做修改
01	替换为 DMA2D_xxPFCCR 中的值
10	替换为原始值与 DMA2D_xxPFCCR 中的值/255 所得商的乘积
11	保留

9.3.5 DMA2D 前景层 FIFO 和背景层 CLUT 接口

CLUT 接口可管理对 CLUT 存储器的访问以及 CLUT 的自动加载。

支持如下三种访问：

- PFC 在像素格式转换期间读取 CLUT
- CPU 对 CLUT 进行数据的读取或写入时，通过 AHB 从设备端口访问 CLUT
- 执行自动加载 CLUT 时，通过 AHB 主设备端口进行 CLUT 写入

可通过两种不同方法执行 CLUT 存储器加载：

- 自动加载

  加载 CLUT 时应遵守以下顺序：

  a) 将 CLUT 地址编程到 DMA2D_FGCMAR 寄存器（前景层 CLUT）或 DMA2D_BGCMAR 寄存器（背景层 CLUT）。

  b) 将 CLUT 大小编程到 DMA2D_FGPFCCR 寄存器（前景层 CLUT）或 MA2D_BGPFCCR 寄存器（背景层 CLUT）的 S[7:0] 字段。

  c) 将 MA2D_FGPFCCR 寄存器（前景层 CLUT）或 DMA2D_BGPFCCR 寄存器（背景层 CLUT）的 START 位置 1 以启动传输。自动加载过程期间，不可通过 CPU 访问 CLUT。如果出现冲突，若 DMA2D_CR 中 CAEIE 被置 1，则发生 CLUT 访问错误中断。

- 手动加载

  应用程序必须通过 DMA2D AHB 从设备端口手动编程本地 CLUT 存储器映射到的 CLUT。前景层 CLUT 从偏移地址 0x0400 开始，背景层 CLUT 从偏移地址 0x0800 开始。

CLUT 格式可以是 24 位或 32 位。通过 DMA2D_FGPFCCR 寄存器（前景层 CLUT）或 DMA2D_BGPFCCR 寄存器（背景层 CLUT）的 CCM 位配置格式，如*表 37：支持的 CLUT 颜色模式*所示。

表 37. 支持的 CLUT 颜色模式

CCM	CLUT 颜色模式
0	32 位 ARGB8888
1	24 位 RGB888

*表 38：存储器中的 CLUT 数据顺序*给出了系统存储器中 CLUT 数据的组织方式。

表 38. 存储器中的 CLUT 数据顺序

CLUT 颜色模式	@ + 3	@ + 2	@ + 1	@ + 0
ARGB8888	A_0[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
RGB888	B_1[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
	G_2[7:0]	B_2[7:0]	R_1[7:0]	G_1[7:0]
	R_3[7:0]	G_3[7:0]	B_3[7:0]	R_2[7:0]

9.3.6 DMA2D 混合器

DMA2D 混合器成对混合源像素以计算结果像素。

混合将按以下公式执行：

$$\text{其中 } \alpha_{Mult} = \frac{\alpha_{FG} \cdot \alpha_{BG}}{255}$$

$$\alpha_{OUT} = \alpha_{FG} + \alpha_{BG} - \alpha_{Mult}$$

$$C_{OUT} = \frac{C_{FG} \cdot \alpha_{FG} + C_{BG} \cdot \alpha_{BG} - C_{BG} \cdot \alpha_{Mult}}{\alpha_{OUT}} \quad \text{其中 C = R 或 G 或 B}$$

商将向下取整

混合器不需要任何配置寄存器。是否使用混合器取决于 DMA2D_CR 寄存器的 MODE[1:0] 字段中定义的 DMA2D 工作模式。

9.3.7 DMA2D 输出 PFC

输出 PFC 将像素格式从 32 位转换为指定的输出格式，输出格式在 DMA2D 输出像素格式转换器配置寄存器 (DMA2D_OPFCCR) 的 CM[2:0] 字段中定义。

*表 39：输出时支持的颜色模式*给出了支持的输出格式。

表 39. 输出时支持的颜色模式

CM[2:0]	颜色模式
000	ARGB8888
001	RGB888
010	RGB565
011	ARGB1555
100	ARGB4444

9.3.8 DMA2D 输出 FIFO

输出 FIFO 根据输出 PFC 中定义的颜色格式对像素进行编程。

通过如下一组寄存器定义目标区域：

- DMA2D 输出存储器地址寄存器 (DMA2D_OMAR)
- DMA2D 输出偏移寄存器 (DMA2D_OOR)
- DMA2D 行数寄存器（行数和每行像素数）(DMA2D_NLR)

如果 DMA2D 在寄存器到存储器模式下工作，则配置的输出矩形将以 DMA2D 输出颜色寄存器 (DMA2D_OCOLR) 中指定的颜色填充，该寄存器中包含固定的 32 位、24 位或 16 位值。通过 DMA2D_OPFCCR 寄存器的 CM[2:0] 字段选择格式。

将按照*表 40：存储器中的数据顺序*中定义的顺序在存储器中存储数据。

表 40. 存储器中的数据顺序

颜色模式	@ + 3	@ + 2	@ + 1	@ + 0
ARGB8888	A_0[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
RGB888	B_1[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
	G_2[7:0]	B_2[7:0]	R_1[7:0]	G_1[7:0]
	R_3[7:0]	G_3[7:0]	B_3[7:0]	R_2[7:0]
RGB565	R_1[4:0]G_1[5:3]	G_1[2:0]B_1[4:0]	R_0[4:0]G_0[5:3]	G_0[2:0]B_0[4:0]
ARGB1555	A_1[0]R_1[4:0]G_1[4:3]	G_1[2:0]B_1[4:0]	A_0[0]R_0[4:0]G_0[4:3]	G_0[2:0]B_0[4:0]
ARGB4444	A_1[3:0]R_1[3:0]	G_1[3:0]B_1[3:0]	A_0[3:0]R_0[3:0]	G_0[3:0]B_0[3:0]

通过 ARGB8888 模式支持按 32 位对齐 RGB888。

9.3.9 DMA2D AHB 主设备端口定时器

AHB 主设备端口内嵌一个 8 位定时器，以便可选择限制交叉开关矩阵的带宽。

此定时器由 AHB 时钟驱动，对两个连续访问之间的死区进行计数。这样可限制带宽的使用。

通过 AHB 主设备端口定时器配置寄存器 (DMA2D_AMPTCR) 配置定时器使能和死区值。

9.3.10 DMA2D 事务

DMA2D 事务由给定数目的数据传输序列组成。可通过软件对数据数目和宽度进行编程。

每个 DMA2D 数据传输最多需要 4 个步骤：

1. 从 DMA2D_FGMAR 寄存器寻址的存储单元加载数据并按照 DMA2D_FGCR 中的定义进行像素格式转换。
2. 从 DMA2D_BGMAR 寄存器寻址的存储单元加载数据并按照 DMA2D_BGCR 中的定义进行像素格式转换。
3. 根据对 alpha 值进行 PFC 操作所得到的 alpha 通道，将所有检索到的像素混合。
4. 根据 DMA2D_OCR 寄存器对合成像素进行像素格式转换，然后将数据编程到通过 DMA2D_OMAR 寄存器寻址的存储单元。

9.3.11 DMA2D 配置

源和目标数据传输在整个 4 GB 区域（地址范围在 0x0000 0000 和 0xFFFF FFFF 之间）都可以寻址外设和存储器。

DMA2D 可在以下四种模式下工作，通过 DMA2D_CR 寄存器的 MODE[1:0] 位选择工作模式：

- 寄存器到存储器
- 存储器到存储器
- 存储器到存储器并执行 PFC
- 存储器到存储器并执行 PFC 和混合

寄存器到存储器

寄存器到存储器模式用于以预定义颜色填充用户自定义区域。

颜色格式在 DMA2D_OPFCCR 中设置。

DMA2D 不从任何源获取数据。它只将 DMA2D_OCOLR 寄存器中定义的颜色写入通过 DMA2D_OMAR 寻址以及 DMA2D_NLR 和 DMA2D_OOR 定义的区域。

存储器到存储器

在存储器到存储器模式下，DMA2D 不执行任何图形数据转换。前景层输入 FIFO 充当缓冲区，数据从 DMA2D_FGMAR 中定义的源存储单元传输到 DMA2D_OMAR 寻址的目标存储单元。

DMA2D_FGPFCCR 寄存器的 CM[3:0] 位中编程的颜色模式决定输入和输出的每像素位数。

对于要传输的区域大小，源区域大小由 DMA2D_NLR 和 DMA2D_FGOR 寄存器定义，目标区域大小则由 DMA2D_NLR 和 DMA2D_OOR 寄存器定义。

存储器到存储器并执行 PFC

此模式下，DMA2D 对源数据执行像素格式转换并将结果存储在目标存储单元。

对于要传输的区域大小，源区域大小由 DMA2D_NLR 和 DMA2D_FGOR 寄存器定义，目标区域大小则由 DMA2D_NLR 和 DMA2D_OOR 寄存器定义。

从 DMA2D_FGMAR 寄存器定义的位置获取数据，并由前景层 PFC 进行处理。原始像素格式通过 DMA2D_FGPFCCR 寄存器配置。

如果原始像素格式是直接颜色模式，则所有颜色通道都扩展到 8 位。

如果像素格式是间接颜色模式，则必须将相关 CLUT 加载到 CLUT 存储器中。

CLUT 加载可按如下顺序自动完成：

1. 在 DMA2D_FGCMAR 中设置 CLUT 地址。
2. 在 DMA2D_FGPFCCR 寄存器的 CS[7:0] 位设置 CLUT 大小。
3. 在 DMA2D_FGPFCCR 寄存器的 CCM 位设置 CLUT 格式（24 或 32 位）。
4. 将 DMA2D_FGPFCCR 寄存器的 START 位置 1 启动 CLUT 加载。

CLUT 加载完成时，DMA2D_ISR 寄存器将置位 CTCIF 标志；如果 DMA2D_CR 中的 CTCIE 位置 1，还将产生中断。CLUT 自动加载过程无法与传统的 DMA2D 传输同时进行。

CLUT 还可由 CPU 填充，或由任意其它主设备通过 APB 端口填充。在 DMA2D 传输进行期间和使用 CLUT（间接颜色格式）时无法访问 CLUT。

在颜色转换执行期间，可根据 DMA2D_FGPFCCR 寄存器中编程的值添加或更改 alpha 值。如果原始图像没有 alpha 通道，则会自动添加一个默认的 alpha 值 0xFF 以获得完全不透明的像素。可根据 DMA2D_FGPFCCR 寄存器的 AM[1:0] 位修改 alpha 值：

- 保持不变。
- 替换为 DMA2D_FGPFCCR 寄存器的 ALPHA[7:0] 值中定义的值。
- 替换为原始值与 DMA2D_FGPFCCR 寄存器的 ALPHA[7:0] 值除以 255 所得商的乘积。

结果得到的 32 位数据由 OUT PFC 编码成 DMA2D_OPFCCR 寄存器的 CM[2:0] 字段所指定的格式。输出像素格式不可是间接模式，原因是输出时不支持 CLUT 生成过程。

数据经处理后，将写入 DMA2D_OMAR 寻址的目标存储单元。

存储器到存储器并执行 PFC 和混合

此模式下，将在前景层 FIFO 和背景层 FIFO（分别在 DMA2D_FGMAR 和 DMA2D_BGMAR 中定义）获取 2 个源图像。

必须按存储器到存储器模式中所述配置两个像素格式转换器。由于这两个像素格式转换器各自独立且自身具有 CLUT 存储器，因此其配置可以不同。

在每个像素都通过相应的 PFC 转换为 32 位后，将根据以下公式进行混合：

$$\text{其中 } \alpha_{Mult} = \frac{\alpha_{FG} \cdot \alpha_{BG}}{255}$$

$$\alpha_{OUT} = \alpha_{FG} + \alpha_{BG} - \alpha_{Mult}$$

$$C_{OUT} = \frac{C_{FG} \cdot \alpha_{FG} + C_{BG} \cdot \alpha_{BG} - C_{BG} \cdot \alpha_{Mult}}{\alpha_{OUT}} \quad \text{其中 C = R 或 G 或 B}$$

商将向下取整

输出 PFC 将根据指定的输出格式对得到的 32 位像素值进行编码，并且编码数据将写入 DMA2D_OMAR 寻址的目标存储单元。

配置错误检测

DMA2D 将在每次执行传输前检查配置是否正确。开始新的传输/自动加载时，如果检测到配置错误，硬件将设置配置错误中断标志。如果 DMA2D_CR 寄存器的 CEIE 位置 1，还将产生中断。

可检测到的错误配置如下：

- 前景层 CLUT 自动加载：DMA2D_FGCMAR 的 MA 位与 DMA2D_FGPFCCR 的 CCM 位不匹配
- 背景层 CLUT 自动加载：DMA2D_BGCMAR 的 MA 位与 DMA2D_BGPFCCR 的 CCM 位不匹配
- 存储器传输（寄存器到存储器模式除外）：DMA2D_FGMAR 的 MA 位与 DMA2D_FGPFCCR 的 CM 位不匹配
- 存储器传输（寄存器到存储器模式除外）：DMA2D_FGPFCCR 中的 CM 位无效
- 存储器传输（寄存器到存储器模式除外）：DMA2D_NLR 的 PL 位为奇，而 DMA2D_FGPFCCR 的 CM 位为 A4 或 L4
- 存储器传输（寄存器到存储器模式除外）：DMA2D_FGOR 中的 LO 位为奇，而 DMA2D_FGPFCCR 的 CM 位为 A4 或 L4
- 存储器传输（仅限混合模式）：DMA2D_BGMAR 中的 MA 位与 DMA2D_BGPFCCR 的 CM 位不一致
- 存储器传输：DMA2D_BGPFCCR 的 CM 无效（仅限混合模式）
- 存储器传输（仅限混合模式）：DMA2D_NLR 的 PL 位为奇，而 DMA2D_BGPFCCR 的 CM 位为 A4 或 L4
- 存储器传输（仅限混合模式）：DMA2D_BGOR 的 LO 位为奇，而 DMA2D_BGPFCCR 的 CM 位为 A4 或 L4
- 存储器传输（存储器到存储器模式除外）：DMA2D_OMAR 的 MA 位与 DMA2D_OPFCCR 的 CM 位不匹配
- 存储器传输（存储器到存储器模式除外）：DMA2D_OPFCCR 中的 CM 位无效
- 存储器传输：DMA2D_NLR 中的 NL 位 = 0
- 存储器传输：DMA2D_NLR 中的 PL 位 = 0

9.3.12 DMA2D 传输控制（启动、挂起、中止和完成）

配置 DMA2D 后，通过将 DMA2D_CR 寄存器的 START 位置 1 可启动传输。传输完成时，START 位自动复位且 DMA2D_ISR 寄存器置位 TCIF 标志。如果 DMA2D_CR 寄存器中的 TCIE 位置 1，还将产生中断。

用户应用程序随时都可通过将 DMA2D_CR 寄存器的 SUSP 位置 1 来挂起 DMA2D。随即可通过将 DMA2D_CR 寄存器的 ABORT 位置 1 中止事务，或通过将 DMA2D_CR 寄存器的 SUSP 位复位重新启动事务。

用户应用程序随时都可通过将 DMA2D_CR 寄存器的 ABORT 位置 1 来中止处理中的事务。在这种情况下，不会置位 TCIF 标志。

还可通过 DMA2D_FGPFCCR 和 DDM12D1_BGPFCCR 寄存器自身的 START 位中止或挂起 CLUT 自动传输过程。

9.3.13 水印

可对水印编程，以在指定行的最后一个像素写入目标存储区域时产生中断。

行号在 DMA2D_LWR 寄存器的 LW[15:0] 字段中定义。

在该行的最后一个像素传输完成时，DMA2D_ISR 将置位 TWIF 标志；在 DMA2D_CR 的 TWIE 位置 1 的情况下，还将产生中断。

9.3.14 错误管理

可触发两种错误：

- AHB 主设备端口错误，通过 DMA2D_ISR 寄存器的 TEIF 标志指示。
- CLUT 访问引发的冲突（CPU 在 CLUT 加载或 DMA2D 传输执行期间尝试访问 CLUT），通过 DMA2D_ISR 寄存器的 CAEIF 标志指示。

这两个标志都与其在 DMA2D_CR 寄存器中的中断使能标志（控制在需要时产生中断的 TEIE 和 CAEIE）相关。

9.3.15 AHB 死区

要限制 AHB 带宽的使用，可在两个连续的 AHB 访问之间编程一个死区。

可通过将 DMA2D_AMTCR 寄存器中的 EN 位置 1 使能此特性。

死区值存储在 DMA2D_AMTCR 寄存器的 DT[7:0] 字段中。该值表示 AHB 总线上两个连续事务之间允许占用的最少周期数。

在 DMA2D 运行过程中对死区值所做的更新将在下一次 AHB 传输时生效。

9.4 DMA2D 中断

发生如下事件时可生成中断：

- 配置错误
- CLUT 传输完成
- CLUT 访问错误
- 到达传输水印
- 传输完成
- 传输错误

可以使用单独的中断使能位以提高灵活性。

表 41. DMA2D 中断请求

中断事件	事件标志	使能控制位
配置错误	CEIF	CEIE
CLUT 传输完成	CTCIF	CTCIE
CLUT 访问错误	CAEIF	CAEIE
传输水印	TWF	TWIE
传输完成	TCIF	TCIE
传输错误	TEIF	TEIE

9.5 DMA2D 寄存器

9.5.1 DMA2D 控制寄存器 (DMA2D_CR)

DMA2D control register

偏移地址：0x0000

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	CEIE	CTCIE	CAEIE	TWIE	TCIE	TEIE	Res.	Res.	Res.	Res.	Res.	ABORT	SUSP	START
		rw	rw	rw	rw	rw	rw						rs	rw	rs

位 31:18 保留，必须保持复位值

位 17:16 **MODE**：DMA2D 模式 (DMA2D mode)
此位由软件置 1 和清零。无法在传输进行时修改此位。
00：存储器到存储器（仅限 FG 获取）
01：存储器到存储器并执行 PFC（仅限 FG PFC 激活时的 FG 获取）
10：存储器到存储器并执行混合（执行 PFC 和混合时的 FG 和 BG 获取）
11：寄存器到存储器（无 FG 和 BG，仅输出阶段激活）

位 15:14 保留，必须保持复位值

位 13 **CEIE**：配置错误中断使能 (Configuration Error Interrupt Enable)
此位由软件置 1 和清零。
0：CE 中断禁止
1：CE 中断使能

位 12 **CTCIE**：CLUT 传输完成中断使能 (CLUT transfer complete interrupt enable)
此位由软件置 1 和清零。
0：CTC 中断禁止
1：CTC 中断使能

位 11 **CAEIE**：CLUT 访问错误中断使能 (CLUT access error interrupt enable)
此位由软件置 1 和清零。
0：CAE 中断禁止
1：CAE 中断使能

位 10 **TWIE**：传输水印中断使能 (Transfer watermark interrupt enable)
此位由软件置 1 和清零。
0：TW 中断禁止
1：TW 中断使能

位 9 **TCIE**：传输完成中断使能 (Transfer complete interrupt enable)

此位由软件置 1 和清零。

0：TC 中断禁止

1：TC 中断使能

位 8 **TEIE**：传输错误中断使能 (Transfer error interrupt enable)

此位由软件置 1 和清零。

0：TE 中断禁止

1：TE 中断使能

位 7:3 保留，必须保持复位值

位 2 **ABORT**：中止 (Abort)

此位可用于中止当前传输。此位可通过软件置 1 并在 START 位复位时由硬件自动复位。

0：不请求传输中止

1：请求传输中止

位 1 **SUSP**：挂起 (Suspend)

此位可用于挂起当前传输。此位由软件置 1 和复位。此位在 START 位复位时由硬件自动复位。

0：传输不挂起

1：传输挂起

位 0 **START**：启动 (Start)

此位可用于根据各种配置寄存器中加载的参数启动 DMA2D。在下列情况下将自动复位此位：

- 传输结束时
- 通过用户应用程序将 DMA2D_CR 中的 ABORT 位置 1 中止数据传输时
- 数据传输出错时
- 因配置错误或已经在进行其它传输操作（CLUT 自动加载）导致数据传输未启动时。

9.5.2 DMA2D 中断状态寄存器 (DMA2D_ISR)

DMA2D Interrupt Status Register

偏移地址：0x0004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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.	CEIF	CTCIF	CAEIF	TWIF	TCIF	TEIF
										r	r	r	r	r	r

位 31:6 保留，必须保持复位值

位 5 **CEIF**：配置错误中断标志 (Configuration error interrupt flag)
DMA2D_CR、DMA2DFGPFCCR 或 DMA2D_BGPFCCR 的 START 位置 1 以及编程了错误的配置时，此位置 1。

位 4 **CTCIF**：CLUT 传输完成中断标志 (CLUT transfer complete interrupt flag)
完成将 CLUT 从系统存储区复制到 DMA2D 内部存储器时，此位置 1。

位 3 **CAEIF**：CLUT 访问错误中断标志 (CLUT access error interrupt flag)
在从系统存储器自动将 CLUT 复制到 DMA2D 内部存储器期间，CPU 若访问 CLUT，此位将置 1。

位 2 **TWIF**：传输水印中断标志 (Transfer watermark interrupt flag)
带水印行的最后一个像素完成传输时，此位置 1。

位 1 **TCIF**：传输完成中断标志 (Transfer complete interrupt flag)
DMA2D 传输操作完成（仅限数据传输）时此位置 1。

位 0 **TEIF**：传输错误中断标志 (Transfer error interrupt flag)
DMA 传输期间（数据传输或 CLUT 自动加载）出错时此位置 1。

9.5.3 DMA2D 中断标志清零寄存器 (DMA2D_IFCR)

DMA2D interrupt flag clear register

偏移地址：0x0008

复位值：0x0000 0000

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

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.	CCEIF	CCTCIF	CAECIF	CTWIF	CTCIF	CTEIF
										rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

位 31:6 保留，必须保持复位值

位 5 **CCEIF**：清除配置错误中断标志 (Clear configuration error interrupt flag)
将此位编程为 1 可清除 DMA2D_ISR 寄存器中的 CEIF 标志

位 4 **CCTCIF**：清除 CLUT 传输完成中断标志 (Clear CLUT transfer complete interrupt flag)
将此位编程为 1 可清除 DMA2D_ISR 寄存器中的 CTCIF 标志

位 3 **CAECIF**：清除 CLUT 访问错误中断标志 (Clear CLUT access error interrupt flag)
将此位编程为 1 可清除 DMA2D_ISR 寄存器中的 CAEIF 标志

位 2 **CTWIF**：清除传输水印中断标志 (Clear transfer watermark interrupt flag)
将此位编程为 1 可清除 DMA2D_ISR 寄存器中的 TWIF 标志

位 1 **CTCIF**：清除传输完成中断标志 (Clear transfer complete interrupt flag)
将此位编程为 1 可清除 DMA2D_ISR 寄存器中的 TCIF 标志

位 0 **CTEIF**：清除传输错误中断标志 (Clear Transfer error interrupt flag)
将此位编程为 1 可清除 DMA2D_ISR 寄存器中的 TEIF 标志

9.5.4 DMA2D 前景层存储器地址寄存器 (DMA2D_FGMAR)

DMA2D foreground memory address register

偏移地址：0x000C

复位值：0x0000 0000

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

位 31:0 **MA[31: 0]**：存储器地址 (Memory address)

前景层图像所用数据的地址。只有在禁止数据传输的情况下才能写入该寄存器。数据传输一旦启动，此寄存器即变为只读。

地址对齐必须与所选图像格式相匹配，例如每像素 32 位格式必须为 32 位对齐，每像素 16 位格式必须为 16 位对齐，而每像素 4 位格式必须为 8 位对齐。

9.5.5 DMA2D 前景层偏移寄存器 (DMA2D_FGOR)

DMA2D foreground offset register

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.	LO[13:0]													
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:14 保留，必须保持复位值

位 13:0 **LO[13: 0]**：行偏移 (Line offset)

用于前景层图像的行偏移（以像素表示）。此值用于生成地址。行偏移将添加到各行末尾，用于确定下一行的起始地址。

只有在禁止数据传输的情况下才能写入这些位。数据传输一旦启动，这些位将变为只读。

如果图像格式为每像素 4 位，则行偏移值必须为偶数。

9.5.6 DMA2D 背景层存储器地址寄存器 (DMA2D_BGMAR)

DMA2D background memory address register

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
MA[31:16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MA[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **MA[31: 0]**：存储器地址 (Memory address)

背景层图像所用数据的地址。只有在禁止数据传输的情况下才能写入该寄存器。数据传输一旦启动，此寄存器即变为只读。

地址对齐必须与所选图像格式相匹配，例如每像素 32 位格式必须为 32 位对齐，每像素 16 位格式必须为 16 位对齐，而每像素 4 位格式必须为 8 位对齐。

9.5.7 DMA2D 背景层偏移寄存器 (DMA2D_BGOR)

DMA2D background offset register

偏移地址：0x0018

复位值：0x0000 0000

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

位 31:14 保留，必须保持复位值

位 13:0 **LO[13: 0]**：行偏移 (Line offset)

用于背景层图像的行偏移（以像素表示）。此值用于生成地址。行偏移将添加到各行末尾，用于确定下一行的起始地址。

只有在禁止数据传输的情况下才能写入这些位。数据传输一旦启动，这些位将变为只读。

如果图像格式为每像素 4 位，则行偏移值必须为偶数。

9.5.8 DMA2D 前景层 PFC 控制寄存器 (DMA2D_FGPFCCR)

DMA2D foreground PFC control register

偏移地址：0x001C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALPHA[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	AM[1:0]	
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
CS[7:0]								Res.	Res.	START	CCM	CM[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw			rc_w1	rw	rw	rw	rw	rw

位 31:24 **ALPHA[7: 0]**：Alpha 值 (Alpha value)

这些位定义固定的 alpha 通道值，该值可替代原始的 alpha 值或与原始的 alpha 值相乘，具体取决于通过 AM[1:0] 位选择的 alpha 模式。

只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

位 23:18 保留，必须保持复位值

位 17:16 **AM[1: 0]**：Alpha 模式 (Alpha mode)

这些位用于选择将用于前景层图像的 alpha 通道值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

00：不修改前景层图像的 alpha 通道值

01：原始前景层图像的 alpha 通道值替换为 ALPHA[7: 0]

10：原始前景层图像的 alpha 通道值替换为 ALPHA[7: 0] 与原始 alpha 通道值的乘积

其它配置无意义

位 15:8 **CS[7: 0]**：CLUT 大小 (CLUT size)

这些位定义前景层图像所用的 CLUT 的大小。CLUT 传输一旦启动，此字段将变为只读。CLUT 条目数等于 CS[7:0] + 1。

位 7:6 保留，必须保持复位值

位 5 **START**：启动 (Start)

可将此位置 1 以启动 CLUT 的自动加载过程。该位在以下情况下自动复位：

- 传输结束时
- 通过用户应用程序将 DMA2D_CR 中的 ABORT 位置 1 中止传输时
- 传输出错时
- 因配置错误或已经在进行其它传输操作（数据传输或自动背景层 CLUT 传输）导致传输未启动时

位 4 **CCM**：CLUT 颜色模式 (CLUT color mode)

此位定义 CLUT 的颜色格式。只有在禁止数据传输的情况下才能写入该位。CLUT 传输一旦启动，此位将变为只读。

0：ARGB8888

1：RGB888

其它：无意义

位 3:0 **CM[3: 0]**：颜色模式 (Color mode)

这些位定义前景层图像的颜色格式。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

0000：ARGB8888
0001：RGB888
0010：RGB565
0011：ARGB1555
0100：ARGB4444
0101：L8
0110：AL44
0111：AL88
1000：L4
1001：A8
1010：A4
其它：无意义

9.5.9 DMA2D 前景层颜色寄存器 (DMA2D_FGCOLR)

DMA2D foreground color 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.	RED[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
GREEN[7:0]								BLUE[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 **RED[7: 0]**：红色值 (Red Value)

这些位定义前景层图像的 A4 或 A8 模式的红色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

位 15:8 **GREEN[7: 0]**：绿色值 (Green Value)

这些位定义前景层图像的 A4 或 A8 模式的绿色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

位 7:0 **BLUE[7: 0]**：蓝色值 (Blue Value)

这些位定义前景层图像的 A4 或 A8 模式的蓝色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

9.5.10 DMA2D 背景层 PFC 控制寄存器 (DMA2D_BGPFCCR)

DMA2D background PFC control 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
ALPHA[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	AM[1:0]	
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
CS[7:0]								Res.	Res.	START	CCM	CM[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw			rc_w1	rw	rw	rw	rw	rw

位 31:24 **ALPHA[7: 0]**：Alpha 值 (Alpha value)

这些位定义固定的 alpha 通道值，该值可替代原始的 alpha 值或与原始的 alpha 值相乘，具体取决于通过 AM[1:0] 位选择的 alpha 模式。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

位 23:18 保留，必须保持复位值

位 17:16 **AM[1: 0]**：Alpha 模式 (Alpha mode)

这些位定义将用于背景层图像的 alpha 通道值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

00：不修改前景层图像的 alpha 通道值

01：原始背景层图像的 alpha 通道值替换为 ALPHA[7: 0]

10：原始背景层图像的 alpha 通道值替换为 ALPHA[7: 0] 与原始 alpha 通道值的乘积

其它：无意义

位 15:8 **CS[7: 0]**：CLUT 大小 (CLUT size)

这些位定义背景层图像所用的 CLUT 的大小。CLUT 传输一旦启动，此字段将变为只读。CLUT 条目数等于 CS[7:0] + 1。

位 7:6 保留，必须保持复位值

位 5 **START**：启动 (Start)

可将此位置 1 以启动 CLUT 的自动加载过程。该位在以下情况下自动复位：

- 传输结束时
- 通过用户应用程序将 DMA2D_CR 中的 ABORT 位置 1 中止传输时
- 传输出错时
- 因配置错误或已经在进行其它传输操作（数据传输或自动背景层 CLUT 传输）导致传输未启动时

位 4 **CCM**：CLUT 颜色模式 (CLUT Color mode)

这些位定义 CLUT 的颜色格式。只有在禁止传输的情况下才能写入该寄存器。CLUT 传输一旦启动，此位将变为只读。

0：ARGB8888

1：RGB888

其它：无意义

位 3:0 **CM[3: 0]**：颜色模式 (Color mode)

这些位定义前景层图像的颜色格式。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

0000：ARGB8888
0001：RGB888
0010：RGB565
0011：ARGB1555
0100：ARGB4444
0101：L8
0110：AL44
0111：AL88
1000：L4
1001：A8
1010：A4
其它：无意义

9.5.11 DMA2D 背景层颜色寄存器 (DMA2D_BGCOLR)

DMA2D background color register

偏移地址：0x0028

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RED[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
GREEN[7:0]								BLUE[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 **RED[7: 0]**：红色值 (Red Value)

这些位定义背景层图像的 A4 或 A8 模式的红色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

位 15:8 **GREEN[7: 0]**：绿色值 (Green Value)

这些位定义背景层图像的 A4 或 A8 模式的绿色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

位 7:0 **BLUE[7: 0]**：蓝色值 (Blue Value)

这些位定义背景层图像的 A4 或 A8 模式的蓝色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

9.5.12 DMA2D 前景层 CLUT 存储器地址寄存器 (DMA2D_FGCMAR)

DMA2D foreground CLUT memory address register

偏移地址：0x002C

复位值：0x0000 0000

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

位 31:0 **MA[31: 0]**：存储器地址 (Memory Address)

专用于前景层图像的 CLUT 地址所使用的数据地址。只有不存在进行中传输的情况下才能写入该寄存器。CLUT 传输一旦启动，此寄存器将变为只读。

如果前景层 CLUT 格式是 32 位，则地址必须是 32 位对齐。

9.5.13 DMA2D 背景层 CLUT 存储器地址寄存器 (DMA2D_BGCMAR)

DMA2D background CLUT memory address register

偏移地址：0x0030

复位值：0x0000 0000

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

位 31:0 **MA[31: 0]**：存储器地址 (Memory address)

专用于背景层图像的 CLUT 地址所使用的数据地址。只有不存在进行中传输的情况下才能写入该寄存器。CLUT 传输一旦启动，此寄存器将变为只读。

如果背景层 CLUT 格式是 32 位，则地址必须是 32 位对齐。

9.5.14 DMA2D 输出 PFC 控制寄存器 (DMA2D_OPFCCR)

DMA2D output PFC control register

偏移地址：0x0034

复位值：0x0000 0000

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

位 31:3 保留，必须保持复位值

位 2:0 **CM[2: 0]**：颜色模式 (Color mode)

这些位定义背景层图像的颜色格式。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

000：ARGB8888
001：RGB888
010：RGB565
011：ARGB1555
100：ARGB4444
其它：无意义

9.5.15 DMA2D 输出颜色寄存器 (DMA2D_OCOLR)

DMA2D output color register

偏移地址：0x0038

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALPHA[7:0]								RED[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
GREEN[7:0]								BLUE[7:0]							
RED[4:0]					GREEN[5:0]					BLUE[4:0]					
A	RED[4:0]				GREEN[4:0]					BLUE[4:0]					
ALPHA[3:0]				RED[3:0]				GREEN[3:0]				BLUE[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **ALPHA[7: 0]**：Alpha 通道值 (Alpha Channel Value)

这些位定义输出颜色的 alpha 通道。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

位 23:16 **RED[7: 0]**：红色值 (Red Value)

这些位定义输出图像的红色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

位 15:8 **GREEN[7: 0]**：绿色值 (Green Value)

这些位定义输出图像的绿色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

位 7:0 **BLUE[7: 0]**：蓝色值 (Blue Value)

这些位定义输出图像的蓝色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

9.5.16 DMA2D 输出存储器地址寄存器 (DMA2D_OMAR)

DMA2D output memory address register

偏移地址：0x003C

复位值：0x0000 0000

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

位 31:0 **MA[31: 0]**：存储器地址 (Memory Address)

输出 FIFO 所用数据的地址。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

地址对齐必须与所选图像格式相匹配，例如每像素 32 位格式必须为 32 位对齐，每像素 16 位格式必须为 16 位对齐。

9.5.17 DMA2D 输出偏移寄存器 (DMA2D_OOR)

DMA2D output offset 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	LO[13:0]													
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:14 保留，必须保持复位值

位 13:0 **LO[13: 0]**：行偏移 (Line Offset)

用于输出的行偏移（以像素表示）。此值用于生成地址。行偏移将添加到各行末尾，用于确定下一行的起始地址。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

9.5.18 DMA2D 行数寄存器 (DMA2D_NLR)

DMA2D number of line register

偏移地址：0x0044

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	PL[13:0]													
		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
NL[15:0]															
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 29:16 **PL[13: 0]**：每行像素数 (Pixel per lines)

待传输区域的每行像素数。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

如果任何一个输入图像格式为每像素 4 位，则每行像素数必须为偶数。

位 15:0 **NL[15: 0]**：行数 (Number of lines)

待传输区域的行数。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

9.5.19 DMA2D 行水印寄存器 (DMA2D_LWR)

DMA2D line watermark 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
LW[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 **LW[15:0]**：行水印 (Line watermark)
这些位可用以配置可产生中断的行水印。
在带水印行的最后一个像素传输完成时产生中断。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

9.5.20 DMA2D AHB 主设备定时器配置寄存器 (DMA2D_AMTCR)

DMA2D AHB master timer configuration register

偏移地址：0x004C

复位值：0x0000 0000

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

位 31:16 保留

位 15:8 **DT[7: 0]**：死区 (Dead Time)
在 AHB 主设备端口上两个连续访问之间所插入的死区值，以 AHB 时钟周期数表示。这些位表示两个连续 AHB 访问之间允许占用的最少周期数。

位 7:1 保留

位 0 **EN**：使能 (Enable)
使能死区功能。

9.5.21 DMA2D 寄存器映射

下表对 DMA2D 寄存器进行了汇总。有关 DMA2D 寄存器基地址的信息，请参见*第 64 页上的第 2.2.2 节*。

表 42. DMA2D 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
0x0000	DMA2D_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MODE[1:0]		Res.	Res.	Res.	CEIE	CTCIE	CAEIE	TWIE	TCIE	TEIE	Res.	Res.	Res.	Res.	ABORT	SUSP	START
	Reset value															0	0				0	0	0	0	0	0					0	0	0
0x0004	DMA2D_ISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CEIF	CTCIF	CAEIF	TWIF	TCIF	TEIF
	Reset value																											0	0	0	0	0	0
0x0008	DMA2D_IFCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCEIF	CCTCIF	CAECIF	CTWIF	CTCIF	CTEIF
	Reset value																											0	0	0	0	0	0
0x000C	DMA2D_FGMAR	MA[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
0x0010	DMA2D_FGOR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LO[13:0]													
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0014	DMA2D_BGMAR	MA[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
0x0018	DMA2D_BGOR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LO[13:0]													
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x001C	DMA2D_FGPFCCR	ALPHA[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	AM[1:0]		CS[7:0]								Res.	Res.	START	CCM	CM[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
0x0020	DMA2D_FGCOLR	APLHA[7:0]								RED[7:0]								GREEN[7:0]								BLUE[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
0x0024	DMA2D_BGPFCCR	ALPHA[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	AM[1:0]		CS[7:0]								Res.	Res.	START	CCM	CM[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
0x0028	DMA2D_BGCOLR	APLHA[7:0]								RED[7:0]								GREEN[7:0]								BLUE[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
0x002C	DMA2D_FGCMAR	MA[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
0x0030	DMA2D_BGCMAR	MA[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
0x0034	DMA2D_OPFCCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CM[2:0]		
	Reset value																														0	0	0

表 42. DMA2D 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
0x0038	DMA2D_OCOLR	APLHA[7:0]								RED[7:0]								GREEN[7:0]								BLUE[7:0]							
		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RED[4:0]					GREEN[6:0]						BLUE[4:0]				
		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	A	RED[4:0]				GREEN[4:0]						BLUE[4:0]				
		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ALPHA[3:0]				RED[3:0]				GREEN[3:0]				BLUE[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
0x003C	DMA2D_OMAR	MA[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
0x0040	DMA2D_OOR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LO[13:0]													
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0044	DMA2D_NLR	Res.	Res.	PL[13:0]														NL[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
0x0048	DMA2D_LWR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LW[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x004C	DMA2D_AMTCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DT[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	EN
	Reset value																	0	0	0	0	0	0	0	0								0
0x0050-Ox03FF		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x0400-0x07FF	DMA2D_FGCLUT	APLHA[7:0][255:0]								RED[7:0][255:0]								GREEN[7:0][255:0]								BLUE[7:0][255:0]							
	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
0x0800-0x0BFF	DMA2D_BGCLUT	APLHA[7:0][255:0]								RED[7:0][255:0]								GREEN[7:0][255:0]								BLUE[7:0][255:0]							
	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

10 嵌套向量中断控制器 (NVIC)

10.1 NVIC 特性

嵌套向量中断控制器 NVIC 包含以下特性：

- STM32F75xxx 和 STM32F74xxx 具有多达 98 个可屏蔽中断通道（不包括带 FPU 的 Cortex®-M7 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

嵌套向量中断控制器 (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。更多关于异常和 NVIC 编程的说明，请参见编程手册 PMxxxx。

10.1.1 SysTick 校准值寄存器

SysTick 校准值设置为 18750。当 SysTick 时钟设置为 18.75 MHz（HCLK/8，HCLK 设为 150 MHz），会产生 1 ms 时间基准。

10.1.2 中断和异常向量

请参见*表 43*，了解 STM32F75xxx 和 STM32F74xxx 器件的向量表。

表 43. STM32F75xxx 和 STM32F74xxx 向量表

位置	优先级	优先级类型	缩略语	说明	地址
-	-	-	-	保留	0x0000 0000
-	-3	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	所有类型的错误	0x0000 000C
-	0	可设置	MemManage	存储器管理	0x0000 0010
-	1	可设置	BusFault	预取指失败，存储器访问失败	0x0000 0014
-	2	可设置	UsageFault	未定义的指令或非法状态	0x0000 0018
-	-	-	-	保留	0x0000 001C - 0x0000 002B
-	3	可设置	SVCall	通过 SWI 指令调用的系统服务	0x0000 002C

表 43. STM32F75xxx 和 STM32F74xxx 向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
-	4	可设置	调试监控器	调试监控器	0x0000 0030
-	-	-	-	保留	0x0000 0034
-	5	可设置	PendSV	可挂起的系统服务	0x0000 0038
-	6	可设置	SysTick	系统节拍定时器	0x0000 003C
0	7	可设置	WWDG	窗口看门狗中断	0x0000 0040
1	8	可设置	PVD	连接到 EXTI 线的可编程电压检测 (PVD) 中断	0x0000 0044
2	9	可设置	TAMP_STAMP	连接到 EXTI 线的入侵和时间戳中断	0x0000 0048
3	10	可设置	RTC_WKUP	连接到 EXTI 线的 RTC 唤醒中断	0x0000 004C
4	11	可设置	FLASH	Flash 全局中断	0x0000 0050
5	12	可设置	RCC	RCC 全局中断	0x0000 0054
6	13	可设置	EXTI0	EXTI 线 0 中断	0x0000 0058
7	14	可设置	EXTI1	EXTI 线 1 中断	0x0000 005C
8	15	可设置	EXTI2	EXTI 线 2 中断	0x0000 0060
9	16	可设置	EXTI3	EXTI 线 3 中断	0x0000 0064
10	17	可设置	EXTI4	EXTI 线 4 中断	0x0000 0068
11	18	可设置	DMA1_Stream0	DMA1 流 0 全局中断	0x0000 006C
12	19	可设置	DMA1_Stream1	DMA1 流 1 全局中断	0x0000 0070
13	20	可设置	DMA1_Stream2	DMA1 流 2 全局中断	0x0000 0074
14	21	可设置	DMA1_Stream3	DMA1 流 3 全局中断	0x0000 0078
15	22	可设置	DMA1_Stream4	DMA1 流 4 全局中断	0x0000 007C
16	23	可设置	DMA1_Stream5	DMA1 流 5 全局中断	0x0000 0080
17	24	可设置	DMA1_Stream6	DMA1 流 6 全局中断	0x0000 0084
18	25	可设置	ADC	ADC1、ADC2 和 ADC3 全局中断	0x0000 0088
19	26	可设置	CAN1_TX	CAN1 TX 中断	0x0000 008C
20	27	可设置	CAN1_RX0	CAN1 RX0 中断	0x0000 0090
21	28	可设置	CAN1_RX1	CAN1 RX1 中断	0x0000 0094
22	29	可设置	CAN1_SCE	CAN1 SCE 中断	0x0000 0098
23	30	可设置	EXTI9_5	EXTI 线 [9:5] 中断	0x0000 009C
24	31	可设置	TIM1_BRK_TIM9	TIM1 刹车中断和 TIM9 全局中断	0x0000 00A0
25	32	可设置	TIM1_UP_TIM10	TIM1 更新中断和 TIM10 全局中断	0x0000 00A4

表 43. STM32F75xxx 和 STM32F74xxx 向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
26	33	可设置	TIM1_TRG_COM_TIM11	TIM1 触发和换相中断与 TIM11 全局中断	0x0000 00A8
27	34	可设置	TIM1_CC	TIM1 捕获比较中断	0x0000 00AC
28	35	可设置	TIM2	TIM2 全局中断	0x0000 00B0
29	36	可设置	TIM3	TIM3 全局中断	0x0000 00B4
30	37	可设置	TIM4	TIM4 全局中断	0x0000 00B8
31	38	可设置	I2C1_EV	I^2C1 事件中断	0x0000 00BC
32	39	可设置	I2C1_ER	I^2C1 错误中断	0x0000 00C0
33	40	可设置	I2C2_EV	I^2C2 事件中断	0x0000 00C4
34	41	可设置	I2C2_ER	I^2C2 错误中断	0x0000 00C8
35	42	可设置	SPI1	SPI1 全局中断	0x0000 00CC
36	43	可设置	SPI2	SPI2 全局中断	0x0000 00D0
37	44	可设置	USART1	USART1 全局中断	0x0000 00D4
38	45	可设置	USART2	USART2 全局中断	0x0000 00D8
39	46	可设置	USART3	USART3 全局中断	0x0000 00DC
40	47	可设置	EXTI15_10	EXTI 线 [15:10] 中断	0x0000 00E0
41	48	可设置	RTC_Alarm	连接到 EXTI 线的 RTC 闹钟（A 和 B）中断	0x0000 00E4
42	49	可设置	OTG_FS WKUP	连接到 EXTI 线的 USB On-The-Go FS 唤醒中断	0x0000 00E8
43	50	可设置	TIM8_BRK_TIM12	TIM8 刹车中断和 TIM12 全局中断	0x0000 00EC
44	51	可设置	TIM8_UP_TIM13	TIM8 更新中断和 TIM13 全局中断	0x0000 00F0
45	52	可设置	TIM8_TRG_COM_TIM14	TIM8 触发和换相中断与 TIM14 全局中断	0x0000 00F4
46	53	可设置	TIM8_CC	TIM8 捕获比较中断	0x0000 00F8
47	54	可设置	DMA1_Stream7	DMA1 流 7 全局中断	0x0000 00FC
48	55	可设置	FSMC	FSMC 全局中断	0x0000 0100
49	56	可设置	SDMMC1	SDMMC1 全局中断	0x0000 0104
50	57	可设置	TIM5	TIM5 全局中断	0x0000 0108
51	58	可设置	SPI3	SPI3 全局中断	0x0000 010C
52	59	可设置	UART4	UART4 全局中断	0x0000 0110
53	60	可设置	UART5	UART5 全局中断	0x0000 0114

表 43. STM32F75xxx 和 STM32F74xxx 向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
54	61	可设置	TIM6_DAC	TIM6 全局中断， DAC1 和 DAC2 下溢错误中断	0x0000 0118
55	62	可设置	TIM7	TIM7 全局中断	0x0000 011C
56	63	可设置	DMA2_Stream0	DMA2 流 0 全局中断	0x0000 0120
57	64	可设置	DMA2_Stream1	DMA2 流 1 全局中断	0x0000 0124
58	65	可设置	DMA2_Stream2	DMA2 流 2 全局中断	0x0000 0128
59	66	可设置	DMA2_Stream3	DMA2 流 3 全局中断	0x0000 012C
60	67	可设置	DMA2_Stream4	DMA2 流 4 全局中断	0x0000 0130
61	68	可设置	ETH	以太网全局中断	0x0000 0134
62	69	可设置	ETH_WKUP	连接到 EXTI 线的以太网唤醒中断	0x0000 0138
63	70	可设置	CAN2_TX	CAN2 TX 中断	0x0000 013C
64	71	可设置	CAN2_RX0	CAN2 RX0 中断	0x0000 0140
65	72	可设置	CAN2_RX1	CAN2 RX1 中断	0x0000 0144
66	73	可设置	CAN2_SCE	CAN2 SCE 中断	0x0000 0148
67	74	可设置	OTG_FS	USB On The Go FS 全局中断	0x0000 014C
68	75	可设置	DMA2_Stream5	DMA2 流 5 全局中断	0x0000 0150
69	76	可设置	DMA2_Stream6	DMA2 流 6 全局中断	0x0000 0154
70	77	可设置	DMA2_Stream7	DMA2 流 7 全局中断	0x0000 0158
71	78	可设置	USART6	USART6 全局中断	0x0000 015C
72	79	可设置	I2C3_EV	I^2C3 事件中断	0x0000 0160
73	80	可设置	I2C3_ER	I^2C3 错误中断	0x0000 0164
74	81	可设置	OTG_HS_EP1_OUT	USB On The Go HS 端点 1 输出全局中断	0x0000 0168
75	82	可设置	OTG_HS_EP1_IN	USB On The Go HS 端点 1 输入全局中断	0x0000 016C
76	83	可设置	OTG_HS_WKUP	连接到 EXTI 的 USB On The Go HS 唤醒中断	0x0000 0170
77	84	可设置	OTG_HS	USB On The Go HS 全局中断	0x0000 0174
78	85	可设置	DCMI	DCMI 全局中断	0x0000 0178
79	86	可设置	CRYP	CRYP 加密全局中断	0x0000 017C
80	87	可设置	HASH_RNG	哈希和随机数发生器全局中断	0x0000 0180
81	88	可设置	FPU	FPU 全局中断	0x0000 0184

表 43. STM32F75xxx 和 STM32F74xxx 向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
82	89	可设置	UART7	UART7 全局中断	0x0000 0188
83	90	可设置	UART8	UART8 全局中断	0x0000 018C
84	91	可设置	SPI4	SPI4 全局中断	0x0000 0190
85	92	可设置	SPI5	SPI5 全局中断	0x0000 0194
86	93	可设置	SPI6	SPI6 全局中断	0x0000 0198
87	94	可设置	SAI1	SAI1 全局中断	0x0000 019C
88	95	可设置	LCD-TFT	LCD-TFT 全局中断	0x0000 01A0
89	96	可设置	LCD-TFT	LCD-TFT 全局错误中断	0x0000 01A4
90	97	可设置	DMA2D	DMA2D 全局中断	0x0000 01A8
91	98	可设置	SAI2	SAI2 全局中断	0x0000 01AC
92	99	可设置	QuadSPI	QuadSPI 全局中断	0x0000 01B0
93	100	可设置	LP Timer1	LP Timer1 全局中断	0x0000 01B4
94	101	可设置	HDMI-CEC	HDMI-CEC 全局中断	0x0000 01B8
95	102	可设置	I2C4_EV	I2C4 事件中断	0x0000 01BC
96	103	可设置	I2C4_ER	I2C4 错误中断	0x0000 01C0
97	104	可设置	SPDIFRX	SPDIFRX 全局中断	0x0000 01C4

11 外部中断和事件控制器 (EXTI)

外部中断/事件控制器包含多达 24 个用于产生事件/中断请求的边沿检测器。每根输入线都可单独进行配置，以选择类型（中断或事件）和相应的触发事件（上升沿触发、下降沿触发或边沿触发）。每根输入线还可单独屏蔽。挂起寄存器用于保持中断请求的状态线。

11.1 EXTI 主要特性

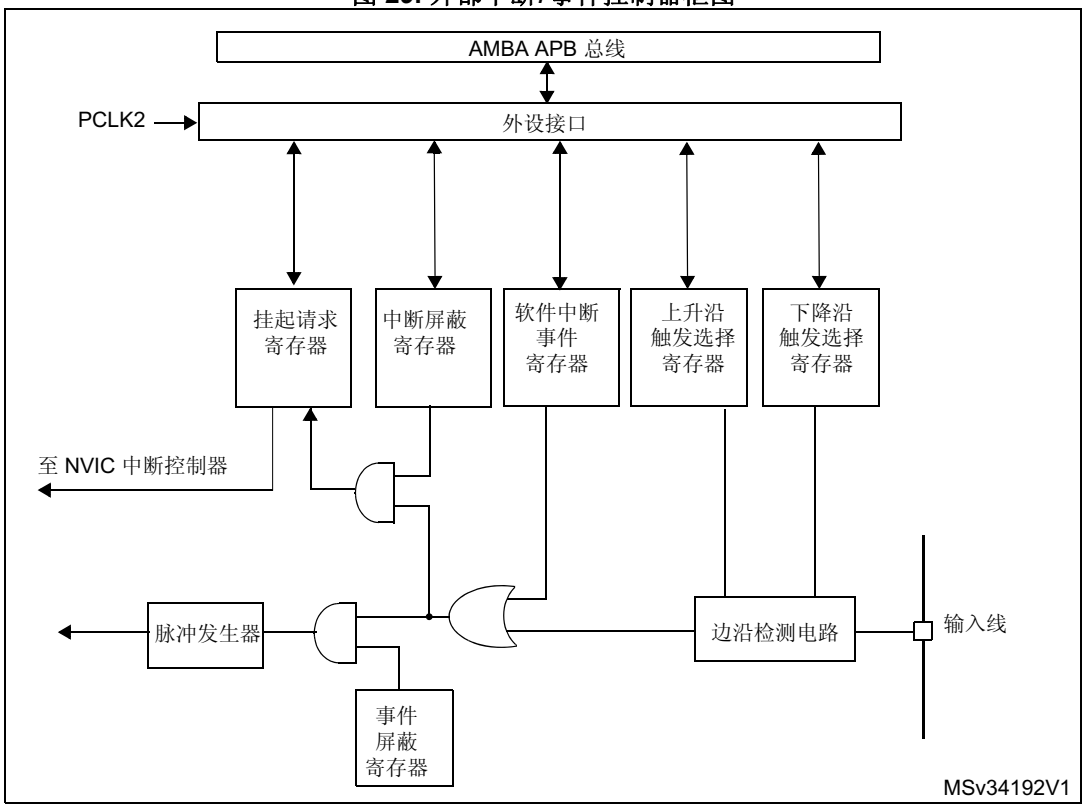
EXTI 控制器的主要特性如下：

- 每个中断/事件线上都具有独立的触发和屏蔽
- 每个中断线都具有专用的状态位
- 支持多达 24 个软件事件/中断请求
- 检测脉冲宽度低于 APB2 时钟宽度的外部信号。有关此参数的详细信息，请参见 STM32F75xxx 和 STM32F74xxx 数据手册的电气特性部分。

11.2 EXTI 框图

图 29 显示了框图。

图 29. 外部中断/事件控制器框图



11.3 唤醒事件管理

STM32F75xxx 和 STM32F74xxx 器件能够处理外部或内部事件来唤醒内核 (WFE)。唤醒事件可通过以下方式产生：

- 在外设的控制寄存器使能一个中断，但不在 NVIC 中使能，同时使能 Cortex®-M7 系统控制寄存器中的 SEVONPEND 位。当 MCU 从 WFE 恢复时，需要清除相应外设的中断挂起位和外设 NVIC 中断通道挂起位（在 NVIC 中断清除挂起寄存器中）。
- 配置一个外部或内部 EXTI 线为事件模式。当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置位，不必清除相应外设的中断挂起位或 NVIC 中断通道挂起位。

使用外部线作为唤醒事件，请参见*第 11.4 节：功能描述*。

11.4 功能描述

要产生中断，必须先配置好并使能中断线。根据需要的边沿检测设置 2 个触发寄存器，同时在中断屏蔽寄存器的相应位写“1”使能中断请求。当外部中断线上出现选定信号沿时，便会产生中断请求， 对应的挂起位也会置 1。在挂起寄存器的对应位写“1”，将清除该中断请求。

要产生事件，必须先配置好并使能事件线。根据需要的边沿检测设置 2 个触发寄存器，同时在事件屏蔽寄存器的相应位写“1”允许事件请求。当事件线上出现选定信号沿时，便会产生事件脉冲， 对应的挂起位不会置 1。

通过在软件中对软件中断/事件寄存器写“1”，也可以产生中断/事件请求。

11.5 硬件中断选择

要将一根线配置为中断源，请执行以下步骤：

1. 配置相应的屏蔽位 (EXTI_IMR)
2. 配置中断线的触发选择位（EXTI_RTSR 和 EXTI_FTSR）
3. 配置对应到外部中断控制器 (EXTI) 的 NVIC 中断通道的使能和屏蔽位，使得 24 个中断线中的请求可以被正确地响应。

11.6 硬件事件选择

要将一根线配置为事件源，请执行以下步骤：

1. 配置相应的屏蔽位 (EXTI_EMR)
2. 配置事件线的触发选择位（EXTI_RTSR 和 EXTI_FTSR）

11.7 软件中断/事件选择

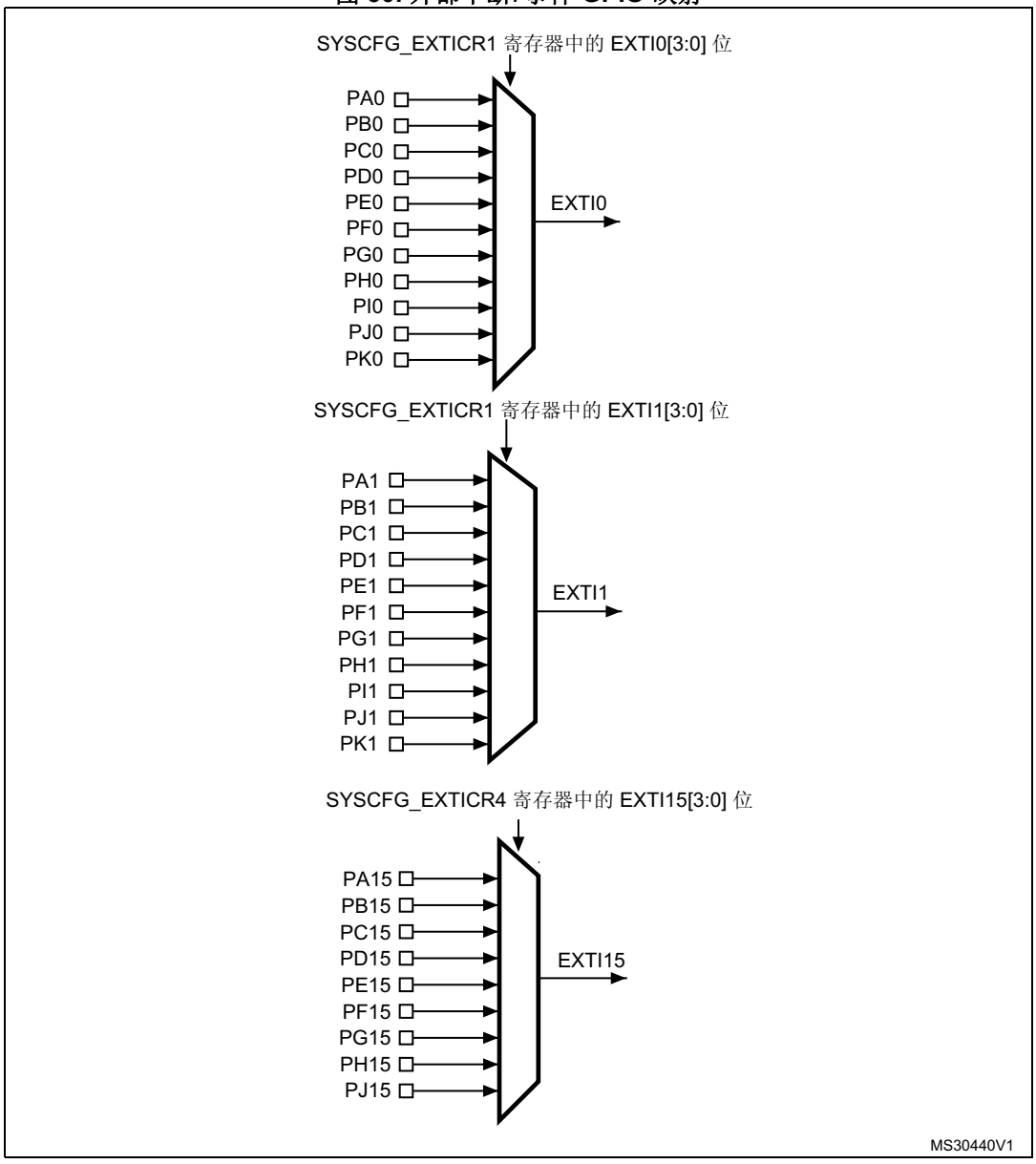
可将这根线配置为软件中断/事件线。以下为产生软件中断的步骤。

1. 配置相应的屏蔽位（EXTI_IMR、EXTI_EMR）
2. 在软件中断寄存器设置相应的请求位 (EXTI_SWIER)

11.8 外部中断/事件线映射

多达 168 个 GPIO 通过以下方式连接到 16 个外部中断/事件线：

图 30. 外部中断/事件 GPIO 映射



另外八根 EXTI 线连接方式如下：

- EXTI 线 16 连接到 PVD 输出
- EXTI 线 17 连接到 RTC 闹钟事件
- EXTI 线 18 连接到 USB OTG FS 唤醒事件
- EXTI 线 19 连接到以太网唤醒事件
- EXTI 线 20 连接到 USB OTG HS（在 FS 中配置）唤醒事件
- EXTI 线 21 连接到 RTC 入侵和时间戳事件

- EXTI 线 22 连接到 RTC 唤醒事件
- EXTI 线 23 连接到 LPTIM1 异步事件

11.9 EXTI 寄存器

有关寄存器说明中使用的缩写，请参见*第 58 页上的第 1.1 节*。

11.9.1 中断屏蔽寄存器 (EXTI_IMR)

Interrupt mask 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.	MR23	MR22	MR21	MR20	MR19	MR18	MR17	MR16
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MR15	MR14	MR13	MR12	MR11	MR10	MR9	MR8	MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23:0 **MRx**：x 线上的中断屏蔽 (Interrupt mask on line x)

0：屏蔽来自 x 线的中断请求
1：开放来自 x 线的中断请求

11.9.2 事件屏蔽寄存器 (EXTI_EMR)

Event mask 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.	MR23	MR22	MR21	MR20	MR19	MR18	MR17	MR16
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MR15	MR14	MR13	MR12	MR11	MR10	MR9	MR8	MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23:0 **MRx**：x 线上的事件屏蔽 (Event mask on line x)

0：屏蔽来自 x 线的事件请求
1：开放来自 x 线的事件请求

11.9.3 上升沿触发选择寄存器 (EXTI_RTSR)

Rising trigger selection 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.	TR23	TR22	TR21	TR20	TR19	TR18	TR17	TR16
								rw	rw	rw	rw	rw	rw	rw	rw

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

位 31:24 保留，必须保持复位值。

位 23:0 **TRx**：线 x 的上升沿触发事件配置位 (Rising trigger event configuration bit of line x)
0：禁止输入线上升沿触发（事件和中断）
1：允许输入线上升沿触发（事件和中断）

注： *外部唤醒线配置为边沿触发时，在这些线上不能出现毛刺信号。*
如果在向 EXTI_RTSR 寄存器写入值的同时外部中断线上产生上升沿，挂起位将被置位。
在同一中断线上，可以同时设置上升沿和下降沿触发。即任一边沿都可触发中断。

11.9.4 下降沿触发选择寄存器 (EXTI_FTSR)

Falling trigger selection 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.	TR23	TR22	TR21	TR20	TR19	TR18	TR17	TR16
								rw	rw	rw	rw	rw	rw	rw	rw

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

位 31:24 保留，必须保持复位值。

位 23:0 **TRx**：线 x 的下降沿触发事件配置位 (Falling trigger event configuration bit of line x)
0：禁止输入线下降沿触发（事件和中断）
1：允许输入线下降沿触发（事件和中断）。

注： *外部唤醒线配置为边沿触发时，在这些线上不能出现毛刺信号。*
如果在向 EXTI_FTSR 寄存器写入值的同时外部中断线上产生下降沿，挂起位不会被置位。
在同一中断线上，可以同时设置上升沿和下降沿触发。即任一边沿都可触发中断。

11.9.5 软件中断事件寄存器 (EXTI_SWIER)

Software interrupt event 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.	SWIER 23	SWIER 22	SWIER 21	SWIER 20	SWIER 19	SWIER 18	SWIER 17	SWIER 16
								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIER 15	SWIER 14	SWIER 13	SWIER 12	SWIER 11	SWIER 10	SWIER 9	SWIER 8	SWIER 7	SWIER 6	SWIER 5	SWIER 4	SWIER 3	SWIER 2	SWIER 1	SWIER 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 23:0 **SWIERx**：线 x 上的软件中断 (Software Interrupt on line x)

当 SWIERx 位设置为“0”时，将“1”写入该位会将 EXTI_PR 寄存器中相应挂起位置 1。
如果在 EXTI_IMR 寄存器中允许在 x 线上产生该中断，则产生中断请求。
通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为“0”。

11.9.6 挂起寄存器 (EXTI_PR)

Pending register

偏移地址：0x14
复位值：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.	PR23	PR22	PR21	PR20	PR19	PR18	PR17	PR16
								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
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

位 31:24 保留，必须保持复位值。

位 23:0 **PRx**：挂起位 (Pending bit)

0：未发生触发请求
1：发生了选择的触发请求
当在外部中断线上发生了选择的边沿事件，该位被置“1”。
将此位编程为“1”可清除此位。

11.9.7 EXTI 寄存器映射

表 44 给出了 EXTI 寄存器映射和复位值。

表 44. 外部中断/事件控制器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	**EXTI_IMR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MR[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
0x04	**EXTI_EMR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MR[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
0x08	**EXTI_RTSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TR[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
0x0C	**EXTI_FTSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TR[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
0x10	**EXTI_SWIER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWIER[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
0x14	**EXTI_PR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR[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

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

12 循环冗余校验计算单元（CRC）

12.1 简介

CRC（循环冗余校验）计算单元使用一个多项式发生器从 8 位/16 位/32 位的数据字中产生 CRC 码。

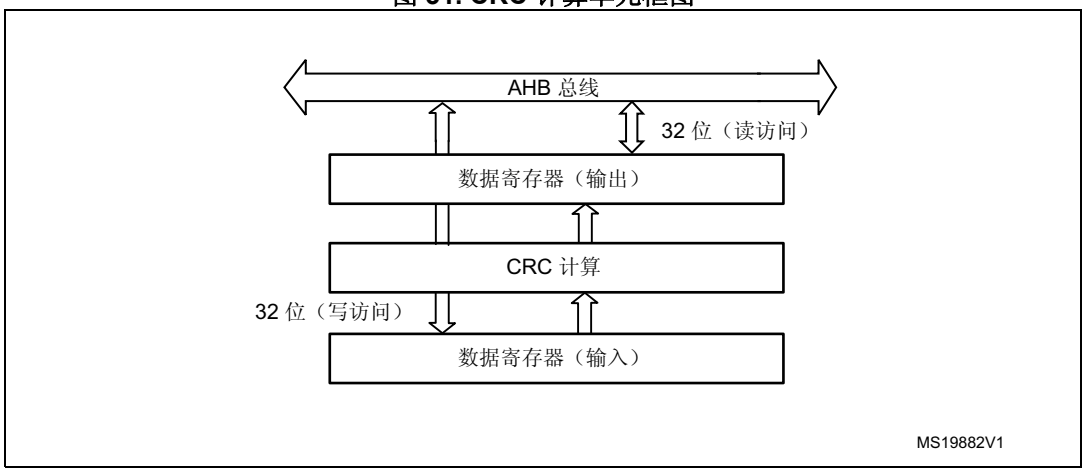
在众多的应用中，基于 CRC 的技术还常用来验证数据传输或存储的完整性。根据功能安全标准的规定，这些技术提供了验证 Flash 完整性的方法。CRC 计算单元有助于在运行期间计算软件的签名，并将该签名与链接时生成并存储在指定存储单元的参考签名加以比较。

12.2 CRC 主要特性

- 使用位数可预设定的（7 位、8 位、16 位和 32 位）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 数据的可逆性选项

12.3 CRC 功能说明

图 31. CRC 计算单元框图



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。对于其它寄存器，只允许进行 32 位访问。

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

输入缓冲器中可立即写入第二个数据，无需因之前的 CRC 计算而等待任何等待状态。

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。

输入数据的顺序可反转，以管理各种数据存放方式（双字/单字/字节，大端/小端等等）。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，输入数据 0x1A2B3C4D 在 CRC 计算中用作：

按字节执行位反转的 0x58D43CB2

按半字执行位反转的 0xD458B23C

按全字执行位反转的 0xB23CD458

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。

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。

12.4 CRC 寄存器

12.4.1 数据寄存器 (CRC_DR)

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

12.4.2 独立数据寄存器 (CRC_IDR)

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

位 31:8 保留，必须保持清零。

位 7:0 **IDR[7:0]**：通用的 8 位数据寄存器位
这些位可用作一个字节的临时存储单元。
此寄存器不受 CRC_CR 寄存器中 RESET 位产生的 CRC 复位影响

12.4.3 控制寄存器 (CRC_CR)

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_输出	REV_IN[1:0]		POLYSIZE[1:0]		Res.	Res.	RESET
								rw	rw	rw	rw	rw			rs

位 31:8 保留，必须保持清零。

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：不影响位顺序
1：位反转输出格式

位 6:5 **REV_IN[1:0]**：反转输入数据 (Reverse input data)
这些位用于控制输入数据位顺序的反转。
00：不影响位顺序
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，必须保持清零。

位 0 **RESET**：RESET 位
此位由软件置 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。此位只能置 1，将由硬件自动进行清零

12.4.4 CRC 初始值 (CRC_INIT)

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

位 31:0 **CRC_INIT**：*可编程 CRC 初始值 (Programmable initial CRC value)*
此寄存器用于写入 CRC 初始值。

12.4.5 CRC 多项式 (CRC_POL)

CRC polynomial

偏移地址：0x14

复位值：0x04C11DB7

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

位 31:0 **POL[31:0]**：可编程多项式 (Programmable polynomial)
此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。

12.4.6 CRC 寄存器映射

表 45. CRC 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	**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**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDR[7:0]							
	Reset value																									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**	Polynomial coefficients																															
	Reset value	0x04C11DB7																															

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

13 可变存储控制器 (FMC)

可变存储控制器 (FMC) 包括以下 3 个存储控制器：

- NOR/PSRAM 存储控制器
- NAND 存储控制器
- 同步 DRAM (SDRAM/Mobile LPSDR SDRAM) 控制器

13.1 FMC 主要特性

FMC 功能块可连接：同步/异步静态存储器、SDRAM 存储器和 NAND Flash。其主要用途有：

- 将 AHB 数据通信事务转换为适当的外部器件协议
- 满足外部存储器器件的访问时间要求

所有外部存储器共享地址、数据和控制信号， 但有各自的片选信号。FMC 一次只能访问一个外部器件。

FMC 控制器的主要特性如下：

- 连接静态存储器映射的器件：
  - 静态随机访问存储器 (SRAM)
  - NOR Flash/OneNAND Flash
  - PSRAM（4 个存储区域）
  - 带有硬件 ECC 的 NAND Flash 存储器，可检查多达 8 KB 的数据
- 连接同步 DRAM (SDRAM/Mobile LPSDR SDRAM) 存储器
- 支持突发模式，能够更快速地访问同步器件（如 NOR Flash、PSRAM 和 SDRAM）
- 可编程连续时钟输出以支持异步和同步访问
- 具有 8 位、16 位或 32 位宽的数据总线
- 每个存储区域有独立的片选控制
- 每个存储区域可独立配置
- 写使能和字节通道选择输出，可配合 PSRAM、SRAM 和 SDRAM 器件使用
- 外部异步等待控制
- 16 x 32 位深度写 FIFO
- SDRAM 控制器具有可缓存的 6 x 32 位深度读 FIFO（6 x 14 位地址标记）。

写 FIFO 由所有存储控制器所共用，包括：

- 写数据 FIFO，用于存储要写入存储器的 AHB 数据（最多 32 位）以及 AHB 传输的一个控制位（突发或非连续模式）。
- 写地址 FIFO，用于存储 AHB 地址（最多 28 位）以及 AHB 数据大小（最多 2 位）。在突发模式下工作时，将仅存储起始地址，但越过页边界时除外（适用于 PSRAM 和 SDRAM）。在此情况下，AHB 突发传输将分成两个 FIFO 条目。

通过将 FMC_BCR1 寄存器中的 WFDIS 位置 1 可禁止写 FIFO。

启动时，必须通过用户应用程序对 FMC 引脚进行配置。应用程序未使用的 FMC I/O 引脚可用于其它用途。

定义外部器件类型和其特性的 FMC 寄存器通常在启动时进行设置，并且在下次上电或复位前保持不变。但是，可随时更改设置。

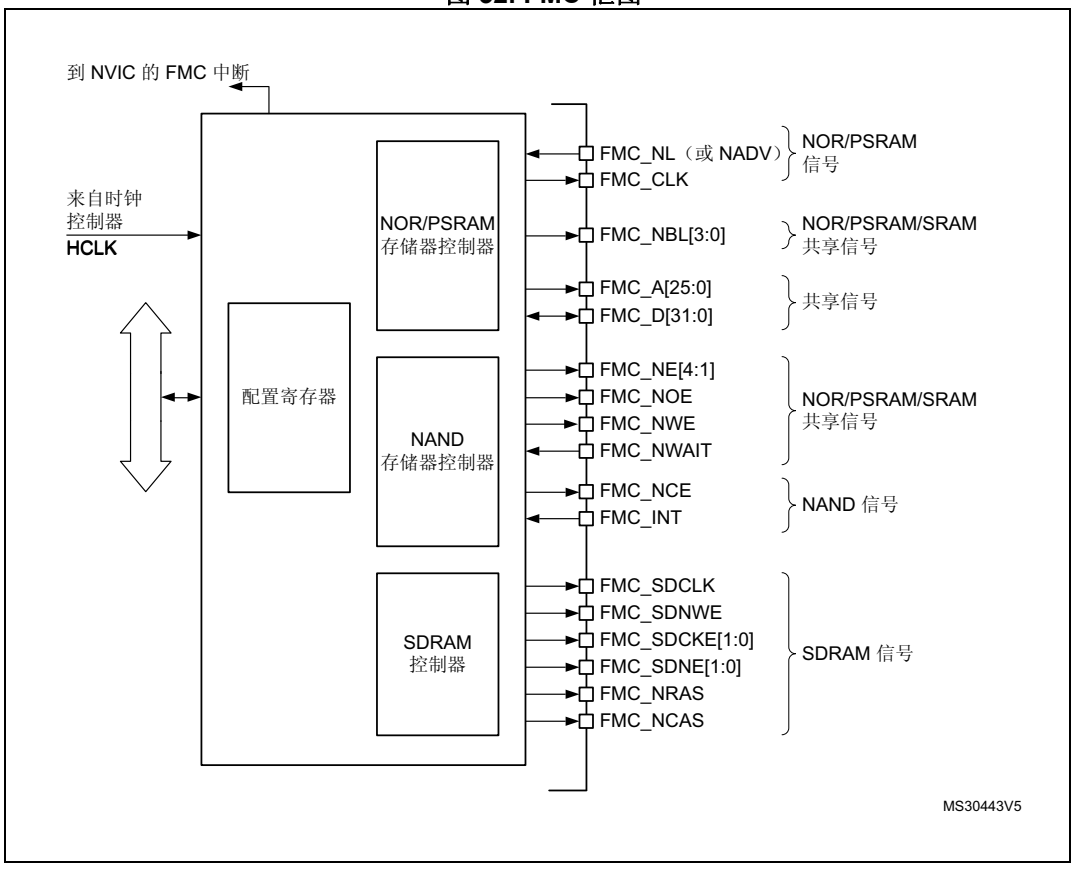
13.2 框图

FMC 包含以下主要模块：

- AHB 接口（包括 FMC 配置寄存器）
- NOR Flash/PSRAM/SRAM 控制器
- SDRAM 控制器
- NAND 控制器

框图如下图所示。

图 32. FMC 框图



13.3 AHB 接口

CPU 和其它 AHB 总线主设备可通过该 AHB 从设备接口访问外部存储器。

AHB 事务会转换为外部器件协议。尤其是当所选外部存储器的宽度为 16 位或 8 位时，AHB 中的 32 位宽事务将被划分成多个连续的 16 或 8 位访问。连续访问之间，FMC 片选 (FMC_NEx) 不会翻转。

出现以下条件时，FMC 将产生 AHB 错误：

- 读取或写入未使能的 FMC 存储区域。
- 在 FMC_BCRx 寄存器中的 FACCEN 位复位时读取或写入 NOR Flash 存储区域。
- 向写保护的 SDRAM 存储区域(SDRAM_SDCRx 寄存器中 WP 位置 1）写入时。
- 违反 SDRAM 地址范围（访问保留的地址范围）时。

此 AHB 错误的影响具体取决于尝试进行读写访问的 AHB 主设备：

- 如果为带 FPU CPU 的 Cortex®-M7，则会生成硬性故障 (Hard fault) 中断。
- 如果为 DMA 控制器，则会生成 DMA 传输错误，并会自动禁止相应的 DMA 通道。

AHB 时钟 (HCLK) 是 FMC 的参考时钟。

13.3.1 支持的存储器和事务

通用事务规则

所请求的 AHB 事务数据宽度可以是 8、16 或 32 位，但访问的外部器件具有固定的数据宽度。这可能会导致不一致的数据宽度。

因此，必须遵循一些简单的事务规则：

- AHB 事务数据宽度和存储器数据宽度相等：
  在此情况下没有任何问题。
- AHB 事务数据宽度大于存储器宽度：
  在此情况下，FMC 会将 AHB 事务分为多个较小的连续存储器访问，以符合外部数据宽度。连续访问之间，FMC 片选 (FMC_NEx) 不会翻转。
- AHB 事务数据宽度小于存储器宽度：
  传送可能一致，也可能不一致，具体取决于外部器件的类型：
  - 访问具有字节选择功能的器件（SRAM、ROM、PSRAM 和 SDRAM）
    FMC 允许读/写事务并通过其字节选择通道 NBL[3:0] 访问恰当的数据。
    通过 NBL[3:0] 寻址要写入的字节。
    读取所有存储器字节（NBL[3:0] 在读取事务期间保持为低电平），并丢弃无用的字节。
  - 访问不具有字节选择功能的器件（NOR 和 NAND Flash）
    当请求对 16 位宽的 Flash 存储器进行字节访问时会发生此情形。由于无法在字节模式下访问器件（只能向 Flash 读取或写入 16 位字），因此允许读事务和写事务（控制器会读取全部 16 位存储器字，但只使用所需字节）。

NOR Flash/PSRAM 和 SDRAM 的回卷支持

由于并非所有主器件都能发出回卷事务，因此同步存储器必须按未定义长度的线性突发模式进行配置。

如果主器件生成 AHB 回卷事务：

- 读操作将分为两个线性突发事务。
- 如果使能写 FIFO，写操作将分为两个线性突发事务；如果禁止写 FIFO，写操作将分为多个线性突发事务。

配置寄存器

可通过一组寄存器配置 FMC。有关 NOR Flash/PSRAM 控制寄存器的详细说明，请参见*第 13.5.6 节*。有关 NAND Flash 寄存器的详细说明，请参见*第 13.6.7 节*，有关 SDRAM 控制寄存器的详细说明，请参见*第 13.7.5 节*。

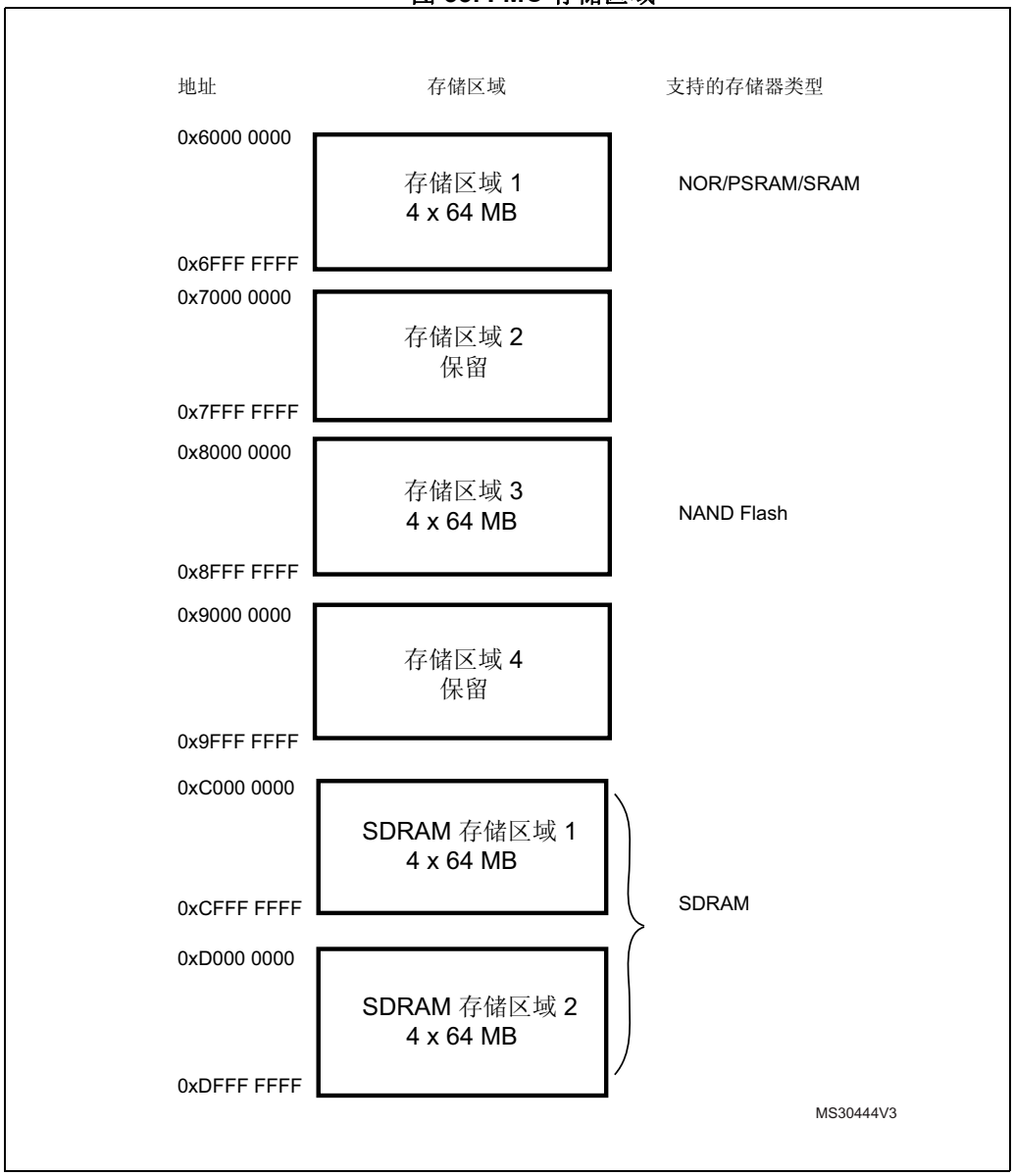
13.4 外部器件地址映射

从 FMC 的角度，外部存储器被划分为固定大小的存储区域，每个存储区域的大小为 256 MB（请参见*图 33*）：

- 存储区域 1 可连接多达 4 个 NOR Flash 或 PSRAM 器件。此存储区域被划分为如下 4 个 NOR/PSRAM 子区域，带 4 个专用片选信号：
  - 存储区域 1 NOR/PSRAM 1
  - 存储区域 1 NOR/PSRAM 2
  - 存储区域 1 NOR/PSRAM 3
  - 存储区域 1 NOR/PSRAM 4
- 存储区域 3 用于连接 NAND Flash 器件。此空间的 MPU 存储器特性必须通过软件重新配置到器件中。
- 存储区域 4 和 5 用于连接 SDRAM 器件（每个存储区域一个器件）。

对于每个存储区域，所要使用的存储器类型可由用户应用程序通过配置寄存器配置。

图 33. FMC 存储区域



13.4.1 NOR/PSRAM 地址映射

HADDR[27:26] 位用于从*表 46* 中所示的四个存储区域之中选择其中一个存储区域。

表 46. NOR/PSRAM 存储区域选择

HADDR[27:26](1)	选择的存储区域
00	存储区域 1 NOR/PSRAM 1
01	存储区域 1 NOR/PSRAM 2
10	存储区域 1 NOR/PSRAM 3
11	存储区域 1 NOR/PSRAM 4

1. HADDR 是 AHB 内部地址线，但也会参与对外部存储器的寻址。

HADDR[25:0] 位包含外部存储器地址。由于 HADDR 为字节地址，而存储器按字寻址，所以根据存储器数据宽度不同，实际向存储器发送的地址也将有所不同，如下表所示。

表 47. NOR/PSRAM 外部存储器地址

存储器宽度(1)	向存储器发出的数据地址	最大存储器容量（位）
8 位	HADDR[25:0]	64 MB x 8 = 512 Mb
16 位	HADDR[25:1] >> 1	64 MB/2 x 16 = 512 Mb
32 位	HADDR[25:2] >> 2	64 MB/4 x 32 = 512 Mb

1. 如果外部存储器的宽度为 16 位，FMC 将使用内部的 HADDR[25:1] 地址来作为对外部存储器的寻址地址 FMC_A[24:0]。如果存储器宽度为 32 位，FMC 将使用内部的 HADDR[25:2] 地址进行外部寻址。
无论外部存储器的宽度是多少，FMC_A[0] 都应连接到外部存储器地址 A[0]。

13.4.2 NAND Flash 地址映射

NAND 存储区域分为几个存储器区域，如*表 48* 所示。

表 48. NAND 存储映射和时序寄存器

起始地址	结束地址	FMC 存储区域	存储空间	时序寄存器
0x8800 0000	0x8BFF FFFF	存储区域 3 - NAND Flash	特性区	FMC_PATT (0x8C)
0x8000 0000	0x83FF FFFF		通用区	FMC_PMEM (0x88)

对于 NAND Flash 存储器，通用区和特性区存储空间分为三个部分，均位于低位 256 KB 中（见下面的*表 49*）：

- 数据区域（通用/特性存储空间中的第一个 64 KB）
- 命令区域（通用/特性存储空间中的第二个 64 KB）
- 地址区域（通用/特性存储空间中的下一个 128 KB）

表 49. NAND 存储区域选择

部分名称	HADDR[17:16]	地址范围
地址区域	1X	0x020000-0x03FFFF
命令区域	01	0x010000-0x01FFFF
数据区域	00	0x000000-0x0FFFF

应用程序软件使用这 3 个区域来访问 NAND Flash 存储器：

- **向 NAND Flash 发送命令**，软件必须向命令区域中的任意存储单元写入命令值。
- **指定读取或写入的 NAND Flash 地址**，软件必须向地址区域中的任意存储单元写入地址值。由于地址的长度可以是 4 或 5 个字节（具体取决于实际存储器大小），要指定完整的地址，需要对地址区域执行多个连续写入操作。
- **读取或写入数据**，软件将从数据区域中的任意存储单元读取数据，或者向其中写入数据。

由于 NAND Flash 存储器会自动递增地址，所以在访问连续存储器位置时，无需递增数据区域的地址。

13.4.3 SDRAM 地址映射

HADDR[28] 位（内部 AHB 地址线 28）用于选择两个存储区域之一，如*表 50* 所示。

表 50. SDRAM 存储区域选择

HADDR[28]	选择的存储区域	控制寄存器	时序寄存器
0	SDRAM Bank1	FMC_SDCR1	FMC_SDTR1
1	SDRAM Bank2	FMC_SDCR2	FMC_SDTR2

下表显示了 13 位行和 11 位列配置的 SDRAM 映射。

表 51. SDRAM 地址映射

存储器宽度(1)	内部存储区域	行地址	列地址(2)	最大存储器容量（MB）
8 位	HADDR[25:24]	HADDR[23:11]	HADDR[10:0]	64 MB： 4 x 8K x 2K
16 位	HADDR[26:25]	HADDR[24:12]	HADDR[11:1]	128 MB： 4 x 8K x 2K x 2
32 位	HADDR[27:26]	HADDR[25:13]	HADDR[12:2]	256 MB： 4 x 8K x 2K x 4

1. 连接 16 位存储器时，FMC 内部使用 HADDR[11:1] 内部 AHB 地址线进行外部寻址。连接 32 位存储器时，FMC 内部使用 HADDR[12:2] 地址线进行外部寻址。无论外部存储器的宽度是多少，FMC_A[0] 都必须连接到外部存储器地址 A[0]。
2. 不支持 AutoPrecharge。FMC_A[10] 必须连接到外部存储器地址 A[10]，但始终为“低电平”。

HADDR[27:0] 位将转换为外部 SDRAM 地址，具体取决于 SDRAM 控制器配置：

- 数据大小：8、16 或 32 位
- 行大小：11、12 或 13 位
- 列大小：8、9、10 或 11 位
- 内部存储区域数量：两个或四个内部存储区域

下表显示了不同 SDRAM 控制器配置的 SDRAM 地址映射。

表 52. 8 位数据总线宽度时的 SDRAM 地址映射(1)(2)

行大小配置	HADDR（AHB 内部地址线）																										
	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
11 位行大小配置	Res.							Bank [1:0]		Row[10:0]											Column[7:0]							
	Res.						Bank [1:0]			Row[10:0]											Column[8:0]							
	Res.					Bank [1:0]				Row[10:0]											Column[9:0]							
	Res.				Bank [1:0]					Row[10:0]											Column[10:0]							
12 位行大小配置	Res.						Bank [1:0]			Row[11:0]											Column[7:0]							
	Res.					Bank [1:0]				Row[11:0]											Column[8:0]							
	Res.				Bank [1:0]					Row[11:0]											Column[9:0]							
	Res.			Bank [1:0]						Row[11:0]											Column[10:0]							
13 位行大小配置	Res.					Bank [1:0]				Row[12:0]											Column[7:0]							
	Res.				Bank [1:0]					Row[12:0]											Column[8:0]							
	Res.			Bank [1:0]						Row[12:0]											Column[9:0]							
	Res.		Bank [1:0]							Row[12:0]											Column[10:0]							

1. BANK[1:0] 为存储区域地址 BA[1:0]。当仅使用 2 个内部存储区域时，BA1 必须始终设置为“0”。
2. 访问保留的 (Res.) 地址范围会生成 AHB 错误。

表 53. 16 位数据总线宽度时的 SDRAM 地址映射(1)(2)

行大小配置	HADDR（AHB 地址线）																										
	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
11 位行大小配置	Res.						Bank [1:0]		Row[10:0]											Column[7:0]								BM0(3)
	Res.						Bank [1:0]		Row[10:0]											Column[8:0]								BM0
	Res.						Bank [1:0]		Row[10:0]											Column[9:0]								BM0
	Res.						Bank [1:0]		Row[10:0]											Column[10:0]								BM0
12 位行大小配置	Res.						Bank [1:0]		Row[11:0]											Column[7:0]								BM0
	Res.						Bank [1:0]		Row[11:0]											Column[8:0]								BM0
	Res.						Bank [1:0]		Row[11:0]											Column[9:0]								BM0
	Res.						Bank [1:0]		Row[11:0]											Column[10:0]								BM0
13 位行大小配置	Res.						Bank [1:0]		Row[12:0]											Column[7:0]								BM0
	Res.						Bank [1:0]		Row[12:0]											Column[8:0]								BM0
	Res.						Bank [1:0]		Row[12:0]											Column[9:0]								BM0
	Res.						Bank [1:0]		Row[12:0]											Column[10:0]								BM0

1. BANK[1:0] 为存储区域地址 BA[1:0]。当仅使用 2 个内部存储区域时，BA1 必须始终设置为“0”。
2. 访问保留的空间 (Res.) 会生成 AHB 错误。
3. BM0：是 16 位访问的字节屏蔽。

表 54. 32 位数据总线宽度时的 SDRAM 地址映射(1)(2)

行大小配置	HADDR（AHB 地址线）																										
	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
11 位行大小配置	Res.						Bank [1:0]		Row[10:0]											Column[7:0]							BM[1:0](3)	
	Res.						Bank [1:0]		Row[10:0]											Column[8:0]							BM[1:0]	
	Res.						Bank [1:0]		Row[10:0]											Column[9:0]							BM[1:0]	
	Res.						Bank [1:0]		Row[10:0]											Column[10:0]							BM[1:0]	

表 54. 32 位数据总线宽度时的 SDRAM 地址映射(1)(2)（续）

行大小配置	HADDR（AHB 地址线）																											
	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
12 位行大小配置	Res.				Bank [1:0]		Row[11:0]												Column[7:0]								BM[1:0]	
	Res.				Bank [1:0]														Column[8:0]								BM[1:0]	
	Res.				Bank [1:0]														Column[9:0]								BM[1:0]	
	Res.				Bank [1:0]														Column[10:0]								BM[1:0]	
13 位行大小配置	Res.				Bank [1:0]		Row[12:0]												Column[7:0]								BM[1:0]	
	Res.				Bank [1:0]														Column[8:0]								BM[1:0]	
	Res.				Bank [1:0]														Column[9:0]								BM[1:0]	
	Bank [1:0]																		Column[10:0]								BM[1:0]	

1. BANK[1:0] 为存储区域地址 BA[1:0]。当仅使用 2 个内部存储区域时，BA1 必须始终设置为“0”。
2. 访问保留的空间 (Res.) 会生成 AHB 错误。
3. BM[1:0]是 32 位访问的字节屏蔽。

13.5 NOR Flash/PSRAM 控制器

FMC 会生成适当的信号时序，以驱动以下类型的存储器：

- 异步 SRAM 和 ROM
  - 8 位
  - 16 位
  - 32 位
- PSRAM (Cellular RAM)
  - 异步模式
  - 同步访问的突发模式
  - 复用或非复用
- NOR Flash
  - 异步模式
  - 同步访问的突发模式
  - 复用或非复用

FMC 会为每个存储区域输出唯一的片选信号 NE[4:1]。所有其它信号（地址、数据和控制）均为共享信号。

FMC 通过可编程时序支持多种器件，其中：

- 等待周期可编程（最多 15 个时钟周期）
- 总线周转周期可编程（最多 15 个时钟周期）
- 输出使能和写入使能延迟可编程（最多 15 个时钟周期）
- 独立的读和写时序和协议，以支持各种存储器和时序
- 可编程连续时钟 (FMC_CLK) 输出

FMC 时钟 (FMC_CLK) 是 HCLK 时钟的约数。该时钟可传送到选定的外部器件，根据 FMC_BCR1 寄存器中 CCKEN 位的配置，可决定只在同步访问期间传送还是同步异步访问期间都传送：

- 如果 CCLKEN 位置 0，则 FMC 仅在同步访问（读/写事务）期间生成时钟 (CLK)。
- 如果 CCLKEN 位置 1，则 FMC 将在异步和同步访问期间生成连续时钟。要生成 FMC_CLK 连续时钟，存储区域 1 还必须配置为支持同步模式（请参见*第 13.5.6 节：NOR/PSRAM 控制寄存器*）。由于所有同步存储器均使用同一个时钟，因此在生成连续输出时钟和执行同步访问时，AHB 数据大小必须与存储器数据宽度 (MWID) 相同，否则 FMC_CLK 频率将根据 AHB 数据事务发生变化（有关 FMC_CLK 分频比公式的信息请参见*第 13.5.5 节: 同步事务*）。

每个存储区域的大小固定，均为 64 MB。每个存储区域都通过专用的寄存器配置（请参见*第 13.5.6 节：NOR/PSRAM 控制寄存器*）。

存储器的可编程参数包括访问时间（请参见*表 55*）和对等待管理的支持（用于在突发模式下访问 NOR Flash 和 PSRAM）。

表 55. NOR/PSRAM 的可编程访问参数

参数	功能	访问模式	单位	最小值	最大值
地址建立	地址建立阶段的持续时间	异步	AHB 时钟周期 (HCLK)	0	15
地址保持	地址保持阶段的持续时间	异步，复用 I/O	AHB 时钟周期 (HCLK)	1	15
数据建立	数据建立阶段的持续时间	异步	AHB 时钟周期 (HCLK)	1	256
总线周转	总线周转阶段的持续时间	异步和同步读/写	AHB 时钟周期 (HCLK)	0	15
时钟分频比	构建一个存储器时钟周期 (CLK) 所需的 AHB 时钟周期 (HCLK) 数量	同步	AHB 时钟周期 (HCLK)	2	16
数据延迟	在发出突发的第一个数据前向存储器发出的时钟周期数量	同步	存储器时钟周期 (CLK)	2	17

13.5.1 外部存储器接口信号

表 56、*表 57* 和*表 58* 列出了通常用于连接 NOR Flash、SRAM 和 PSRAM 的信号。

注： *前缀“N”标识低电平有效的信号。*

NOR Flash，非复用 I/O

表 56. 非复用 I/O NOR Flash

FMC 信号名称	I/O	功能
CLK	O	时钟（用于同步访问）
A[25:0]	O	地址总线
D[31:0]	I/O	双向数据总线
NE[x]	O	片选，x = 1..4
NOE	O	输出使能
NWE	O	写入使能
NL(= NADV)	O	锁存使能（对于部分 NOR Flash 器件，此信号也称为地址有效 (NADV)）
NWAIT	I	FMC 的 NOR Flash 等待输入信号

最大容量为 512 Mb（26 个地址线）。

NOR Flash，16 位复用 I/O

表 57. 16 位复用 I/O NOR Flash

FMC 信号名称	I/O	功能
CLK	O	时钟（用于同步访问）
A[25:16]	O	地址总线
AD[15:0]	I/O	16 位复用，双向地址/数据总线 （16 位地址 A[15:0] 和数据 D[15:0] 在数据总线上复用）
NE[x]	O	片选，x = 1..4
NOE	O	输出使能
NWE	O	写入使能
NL(= NADV)	O	锁存使能 （对于部分 NOR Flash 器件，此信号也称为地址有效 (NADV)）
NWAIT	I	FMC 的 NOR Flash 等待输入信号

最大容量为 512 Mb。

PSRAM/SRAM，非复用 I/O

表 58. 非复用 I/O PSRAM/SRAM

FMC 信号名称	I/O	功能
CLK	O	时钟（仅用于 PSRAM 同步访问）
A[25:0]	O	地址总线
D[31:0]	I/O	数据双向总线

表 58. 非复用 I/O PSRAM/SRAM（续）

FMC 信号名称	I/O	功能
NE[x]	O	片选，x = 1..4 （在 PSRAM 应用中被称作 NCE（Cellular RAM，即 CRAM））
NOE	O	输出使能
NWE	O	写入使能
NL(= NADV)	O	仅用于 PSRAM 输入的地址有效信号（存储器信号名称：NADV）
NWAIT	I	PSRAM 发送给 FMC 的等待输入信号
NBL[3:0]	O	字节通道输出。字节 0 到字节 3 控制（高字节和低字节使能）

最大容量为 512 Mb。

PSRAM，16 位复用 I/O

表 59. 16 位复用 I/O PSRAM

FMC 信号名称	I/O	功能
CLK	O	时钟（用于同步访问）
A[25:16]	O	地址总线
AD[15:0]	I/O	16 位复用，双向地址/数据总线 （16 位地址 A[15:0] 和数据 D[15:0] 在数据总线上复用）
NE[x]	O	片选，x = 1..4 （在 PSRAM 应用中被称作 NCE（Cellular RAM，即 CRAM））
NOE	O	输出使能
NWE	O	写入使能
NL(= NADV)	O	用于 PSRAM 输入的地址有效信号（存储器信号名称：NADV）
NWAIT	I	PSRAM 发送给 FMC 的等待输入信号
NBL[1:0]	O	字节通道输出。字节 0 和字节 1 控制（高字节和低字节使能）

最大容量为 512 Mb（26 个地址线）。

13.5.2 支持的存储器和事务

下面的*表 60* 显示的是当 NOR Flash、PSRAM 和 SRAM 的存储器数据总线宽度为 16 位时所支持的设备、访问模式和事务的示例。本示例中 FMC 不允许（或不支持）的事务以灰色显示。

表 60. NOR Flash/PSRAM：支持的存储器和事务示例

设备	模式	R/W	AHB 数据大小	存储器数据大小	是否允许	注释
NOR Flash（复用 I/O 和非复用 I/O）	异步	R	8	16	是	-
	异步	W	8	16	N	-
	异步	R	16	16	是	-
	异步	W	16	16	是	-
	异步	R	32	16	是	分为 2 次 FMC 访问
	异步	W	32	16	是	分为 2 次 FMC 访问
	异步页	R	-	16	N	不支持该模式
	同步	R	8	16	N	-
	同步	R	16	16	是	-
	同步	R	32	16	是	-
PSRAM（复用 I/O 和非复用 I/O）	异步	R	8	16	是	-
	异步	W	8	16	是	使用字节通道 NBL[1:0]
	异步	R	16	16	是	-
	异步	W	16	16	是	-
	异步	R	32	16	是	分为 2 次 FMC 访问
	异步	W	32	16	是	分为 2 次 FMC 访问
	异步页	R	-	16	N	不支持该模式
	同步	R	8	16	N	-
	同步	R	16	16	是	-
	同步	R	32	16	是	-
	同步	W	8	16	是	使用字节通道 NBL[1:0]
	同步	W	16/32	16	是	-
SRAM 和 ROM	异步	R	8 / 16	16	是	-
	异步	W	8 / 16	16	是	使用字节通道 NBL[1:0]
	异步	R	32	16	是	分为 2 次 FMC 访问
	异步	W	32	16	是	分为 2 次 FMC 访问，使用字节通道 NBL[1:0]

13.5.3 通用时序规则

信号同步

- 所有的控制器输出信号在内部时钟 (HCLK) 的上升沿变化
- 在同步模式（读取或写入）下，所有输出信号在 HCLK 的上升沿变化。无论 CLKDIV 值为何，所有输出均会按以下方式变化：
  - NOEL/NWEL/NEL/NADVL/NADVH/NBLL/ 地址有效输出在 FMC_CLK 时钟的下降沿变化。
  - NOEH/NWEH/NEH/NOEH/NBLH/ 地址有效输出在 FMC_CLK 时钟的上升沿变化。

13.5.4 NOR Flash/PSRAM 控制器异步事务

异步静态存储器（NOR Flash、PSRAM、SRAM）

- 信号通过内部时钟 HCLK 进行同步。此时钟不会发送到存储器
- FMC 总是会先对数据进行采样，而后再禁止片选信号 NE。这样可以确保符合存储器数据保持时序的要求（片选使能为高到数据转换最短时间通常为 0 ns）
- 如果使能扩展模式（FMC_BCRx 寄存器中的 EXTMOD 位置 1），则最多可提供四种扩展模式（A、B、C 和 D）。可以混合使用 A、B、C 和 D 模式来进行读取和写入操作。例如，可以在模式 A 下执行读取操作，而在模式 B 下执行写入操作。
- 如果禁用扩展模式（FMC_BCRx 寄存器中的 EXTMOD 位复位），则 FMC 可以在模式 1 或模式 2 下运行，如下所述：
  - 当选择 SRAM/PSRAM 存储器类型时，模式 1 为默认模式（FMC_BCRx 寄存器中 MTYP = 0x0 或 0x01）
  - 当选择 NOR 存储器类型时，模式 2 为默认模式（FMC_BCRx 寄存器中 MTYP = 0x10）。

模式 1 - SRAM/PSRAM (CRAM)

下图显示了所支持模式的读取和写入事务以及所需的 FMC _BCRx 和 FMC_BTRx/FMC_BWTRx 寄存器配置。

图 34. 模式 1 读取访问波形

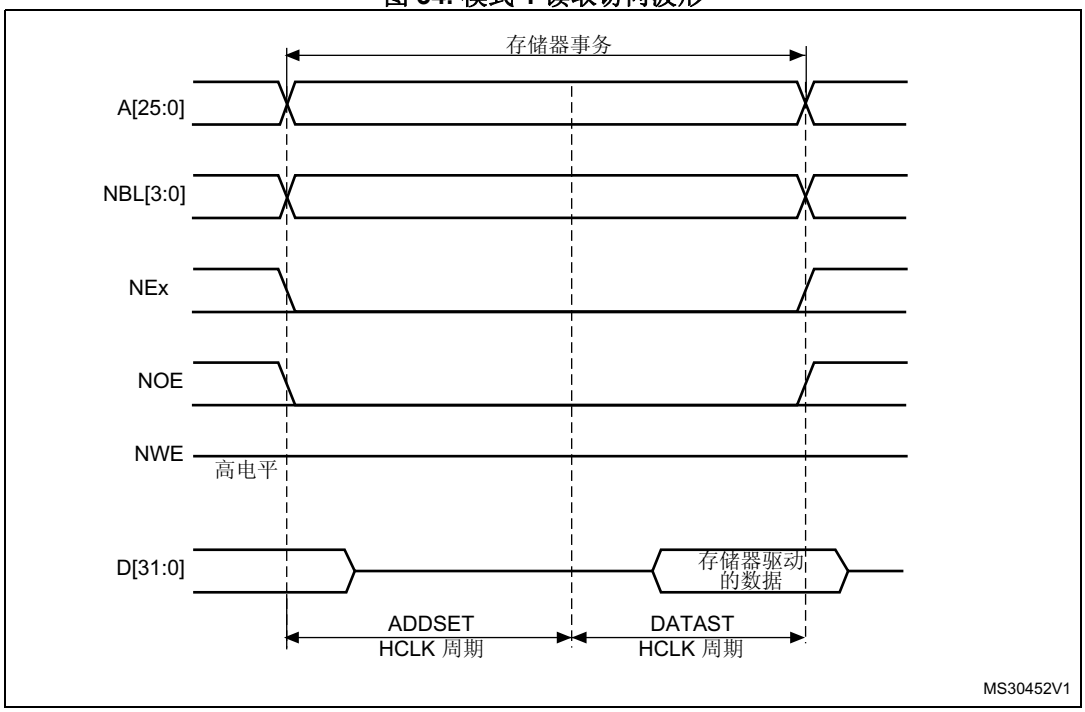
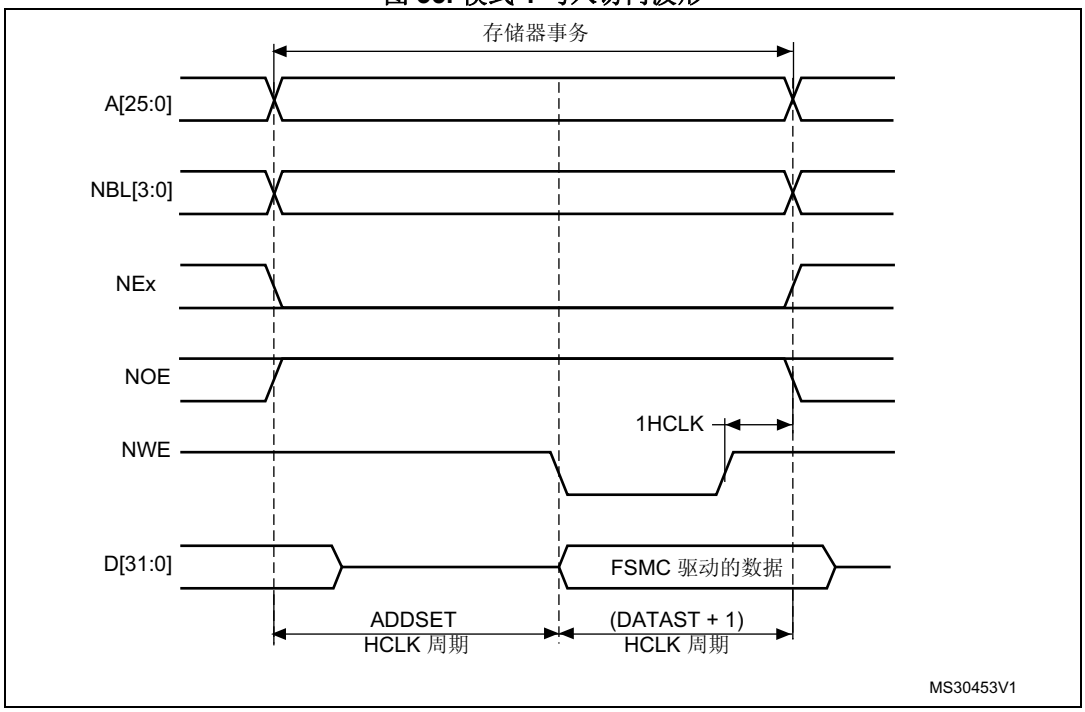


图 35. 模式 1 写入访问波形



位于写入事务末尾的一个 HCLK 周期有助于确保 NWE 上升沿之后的地址和数据保持时间。由于存在此 HCLK 周期，DATAST 值必须大于零 (DATAST > 0)。

表 61. FMC_BCRx 位字段

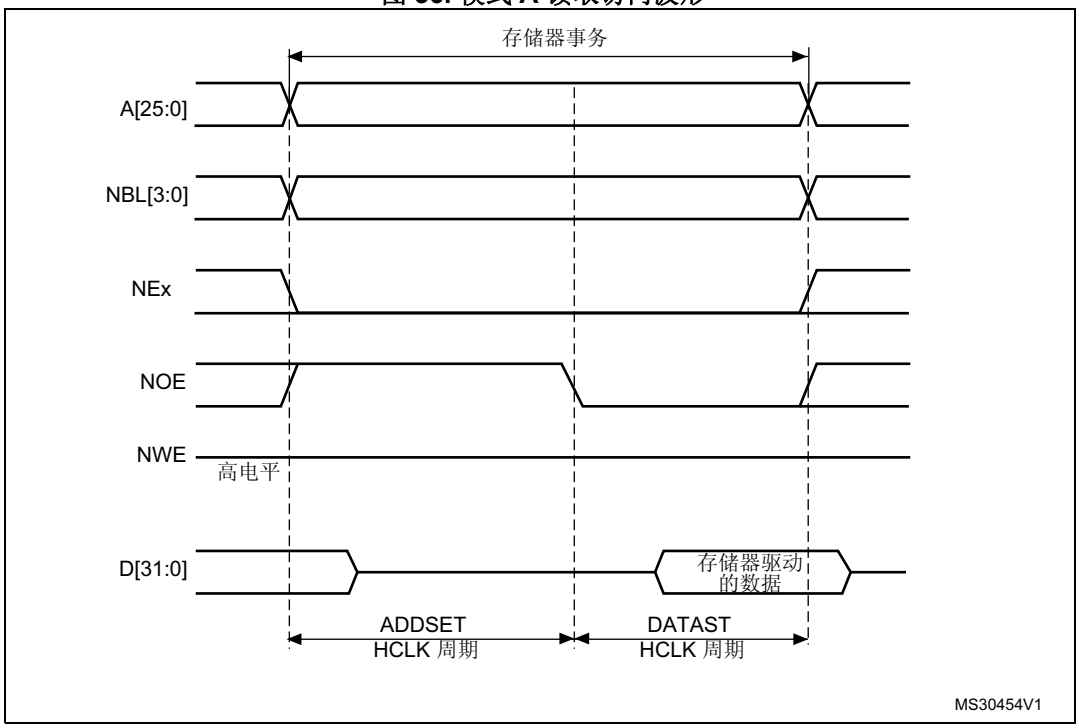
位号	位名	要设置的值
31-22	保留	0x000
21	WFDIS	根据需要进行设置
20	CCLKEN	根据需要进行设置
19	CBURSTRW	0x0（对异步模式没有影响）
18:16	CPSIZE	0x0（对异步模式没有影响）
15	ASYNCWAIT	如果存储器支持该特性，则置为 1。否则，保持为 0。
14	EXTMOD	0x0
13	WAITEN	0x0（对异步模式没有影响）
12	WREN	根据需要进行设置
11	保留	0x0
10	WRAPMOD	0x0
9	WAITPOL	仅当位 15 为 1 时才有意义
8	BURSTEN	0x0
7	保留	0x1
6	FACCEN	无关
5-4	MWID	根据需要进行设置
3-2	MTYP	根据需要进行设置，0x2 除外 (NOR Flash)
1	MUXE	0x0
0	MBKEN	0x1

表 62. FMC_BTRx 位字段

位号	位名	要设置的值
31:30	保留	0x0
29-28	ACCMOD	无关
27-24	DATLAT	无关
23-20	CLKDIV	无关
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	第二个访问阶段的持续时间（写入访问为 DATAST+1 个 HCLK 周期，读取访问为 DATAST 个 HCLK 周期）。
7-4	ADDHLD	无关
3-0	ADDSET	第一个访问阶段的持续时间（ADDSET 个 HCLK 周期）。 ADDSET 最小值为 0。

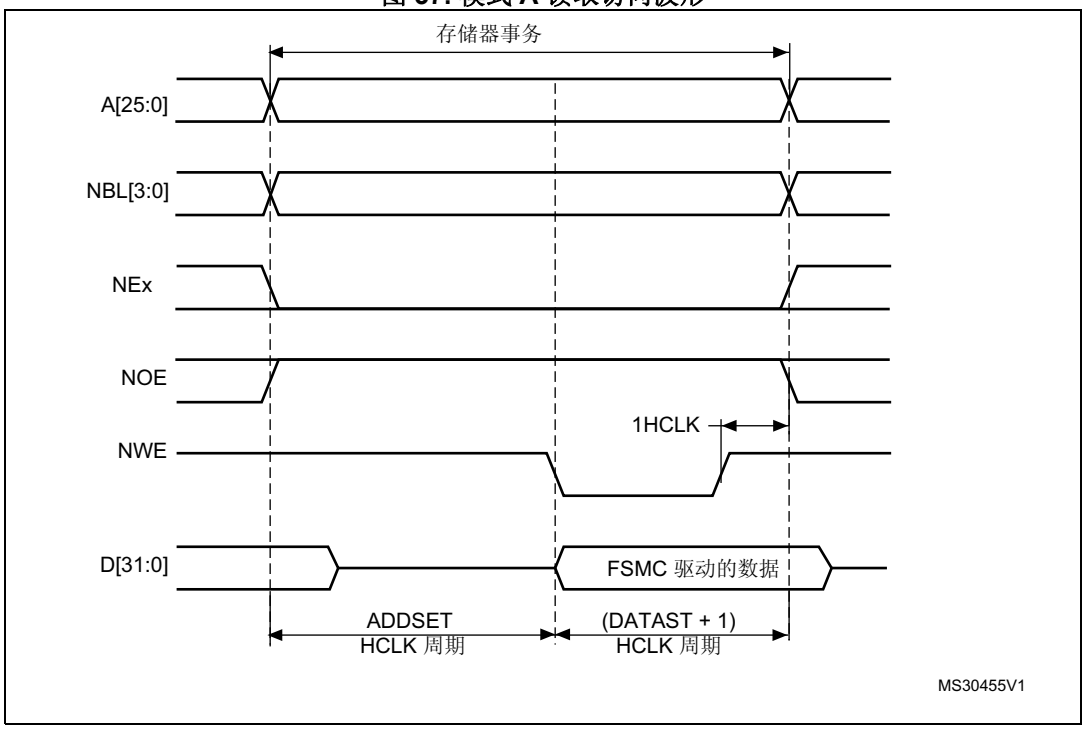
模式 A - SRAM/PSRAM (CRAM) OE 切换

图 36. 模式 A 读取访问波形



1. NBL[3:0] 在进行读取访问时为低电平

图 37. 模式 A 读取访问波形



与模式 1 的不同之处在于 NOE 的切换与独立的读取和写入时序。

表 63. FMC_BCRx 位字段

位号	位名	要设置的值
31-22	保留	0x000
21	WFDIS	根据需要进行设置
20	CCLKEN	根据需要进行设置
19	CBURSTRW	0x0（对异步模式没有影响）
18:16	CPSIZE	0x0（对异步模式没有影响）
15	ASYNCWAIT	如果存储器支持该特性，则置为 1。否则，保持为 0。
14	EXTMOD	0x1
13	WAITEN	0x0（对异步模式没有影响）
12	WREN	根据需要进行设置
11	WAITCFG	无关
11	保留	0x0
9	WAITPOL	仅当位 15 为 1 时才有意义
8	BURSTEN	0x0
7	保留	0x1
6	FACCEN	无关
5-4	MWID	根据需要进行设置
3-2	MTYP	根据需要进行设置，0x2 除外 (NOR Flash)
1	MUXEN	0x0
0	MBKEN	0x1

表 64. FMC_BTRx 位字段

位号	位名	要设置的值
31:30	保留	0x0
29-28	ACCMOD	0x0
27-24	DATLAT	无关
23-20	CLKDIV	无关
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	写入访问第二个阶段的持续时间（DATAST+1 个 HCLK 周期）
7-4	ADDHLD	无关
3-0	ADDSET	写入访问第一个阶段的持续时间（ADDSET+1 个 HCLK 周期）。 ADDSET 最小值为 0。

表 65. FMC_BWTRx 位字段

位号	位名	要设置的值
31:30	保留	0x0
29-28	ACCMOD	0x0
27-24	DATLAT	无关
23-20	CLKDIV	无关
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	写入访问第二个阶段的持续时间（DATAST 个 HCLK 周期）。
7-4	ADDHLD	无关
3-0	ADDSET	写入访问第一个阶段的持续时间（ADDSET 个 HCLK 周期）。 ADDSET 最小值为 0。

模式 2/B - NOR Flash

图 38. 模式 2 和模式 B 读取访问波形

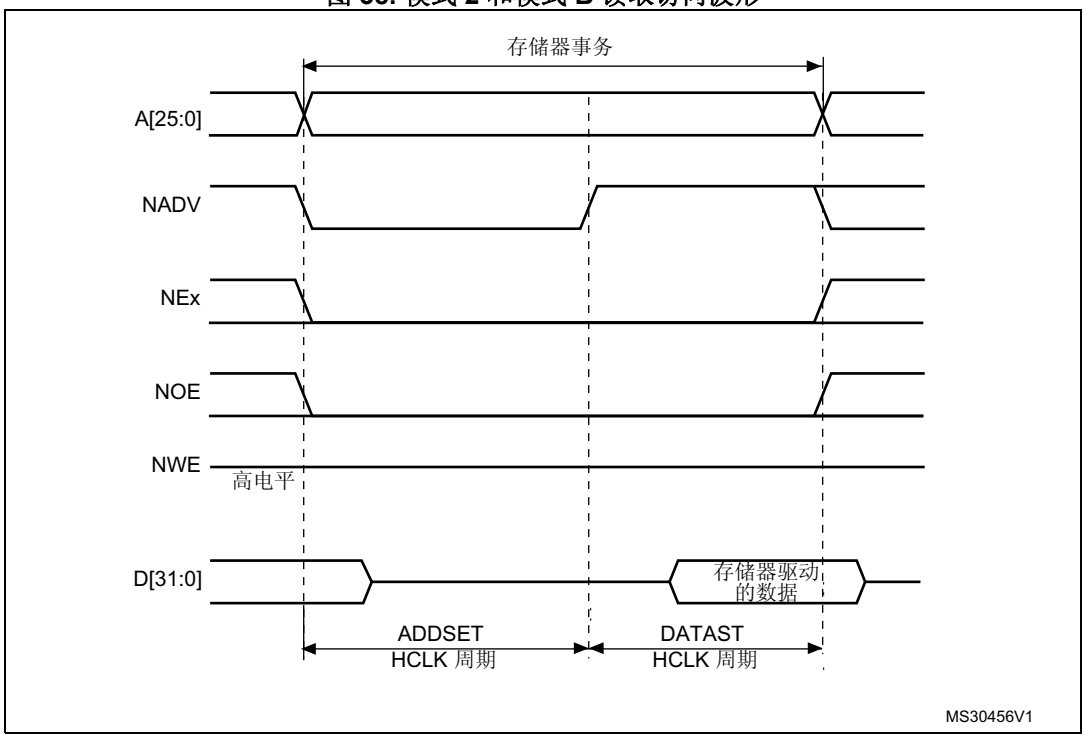


图 39. 模式 2 读取访问波形

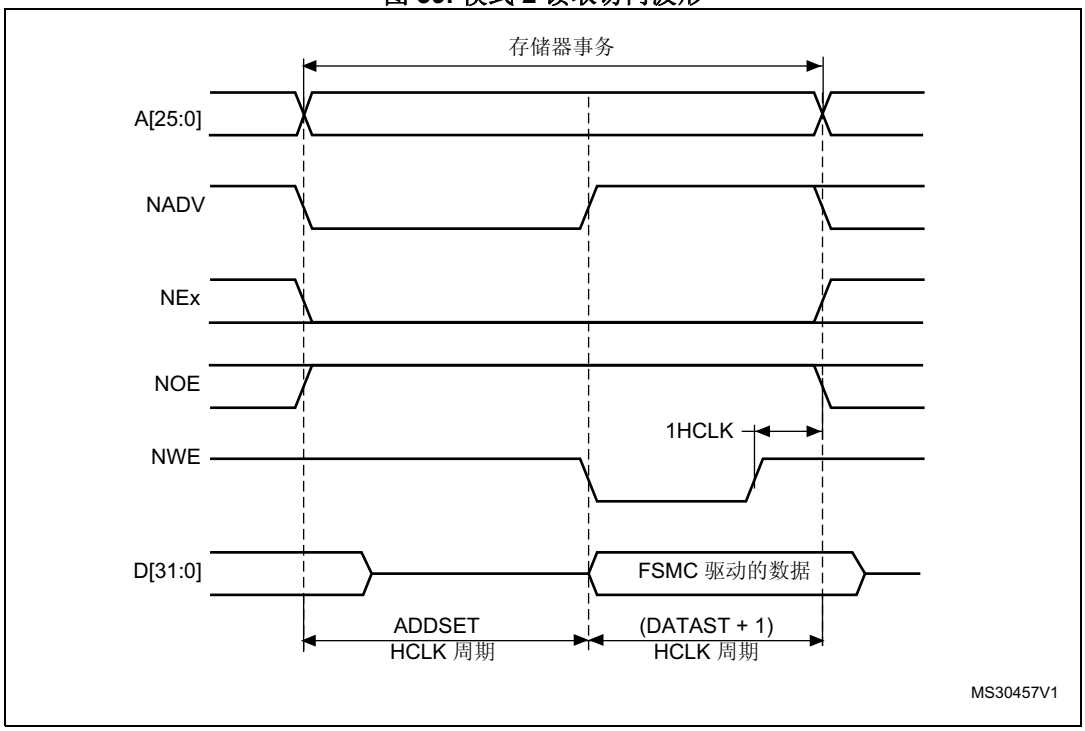
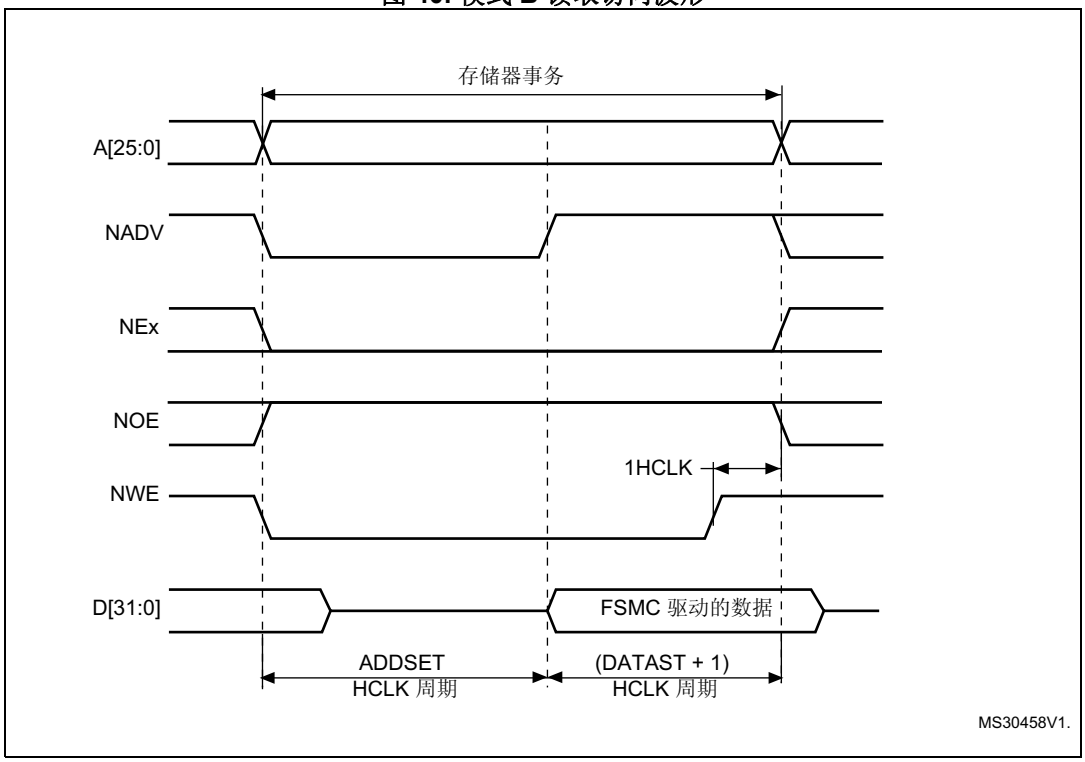


图 40. 模式 B 读取访问波形



与模式 1 的不同之处在于设置扩展模式（模式 B）时的 NWE 切换与独立的读取和写入时序。

表 66. FMC_BCRx 位字段

位号	位名	要设置的值
31-22	保留	0x000
21	WFDIS	根据需要进行设置
20	CCLKEN	根据需要进行设置
19	CBURSTRW	0x0（对异步模式没有影响）
18:16	CPSIZE	0x0（对异步模式没有影响）
15	ASYNCWAIT	如果存储器支持该特性，则置为 1。否则，保持为 0。
14	EXTMOD	模式 B 为 0x1，模式 2 为 0x0
13	WAITEN	0x0（对异步模式没有影响）
12	WREN	根据需要进行设置
11	WAITCFG	无关
10	保留	0x0
9	WAITPOL	仅当位 15 为 1 时才有意义
8	BURSTEN	0x0
7	保留	0x1
6	FACCEN	0x1
5-4	MWID	根据需要进行设置
3-2	MTYP	0x2 (NOR Flash)
1	MUXEN	0x0
0	MBKEN	0x1

表 67. FMC_BTRx 位字段

位号	位名	要设置的值
31-30	保留	0x0
29-28	ACCMOD	0x1 if extended mode is set
27-24	DATLAT	无关
23-20	CLKDIV	无关
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	读取访问第二个阶段的持续时间（DATAST 个 HCLK 周期）。
7-4	ADDHLD	无关
3-0	ADDSET	读取访问第一个阶段的持续时间（ADDSET 个 HCLK 周期）。ADDSET 最小值为 0。

表 68. FMC_BWTRx 位字段

位号	位名	要设置的值
31-30	保留	0x0
29-28	ACCMOD	0x1 if extended mode is set
27-24	DATLAT	无关
23-20	CLKDIV	无关
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	写入访问第二个阶段的持续时间（DATAST 个 HCLK 周期）。
7-4	ADDHLD	无关
3-0	ADDSET	写入访问第一个阶段的持续时间（ADDSET 个 HCLK 周期）。ADDSET 最小值为 0。

注：　仅当设置了扩展模式（模式B）时，FMC_BWTRx 寄存器才有效，否则其内容均为“无关”。

模式 C - NOR Flash - OE 切换

图 41. 模式 C 读取访问波形

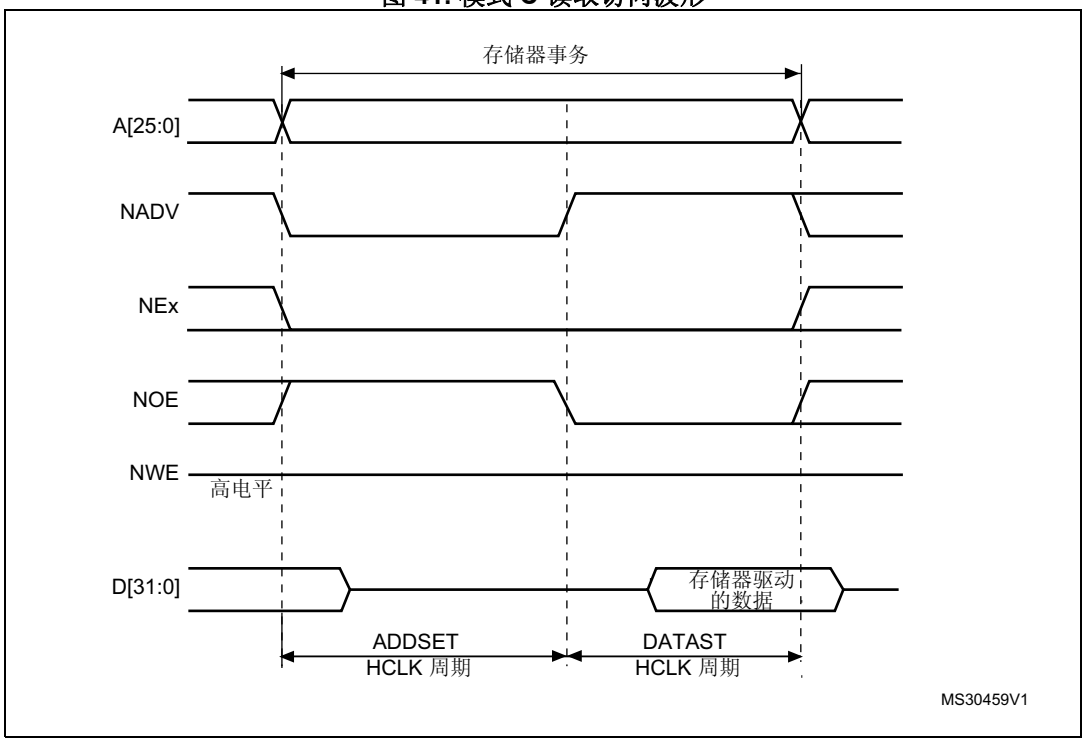
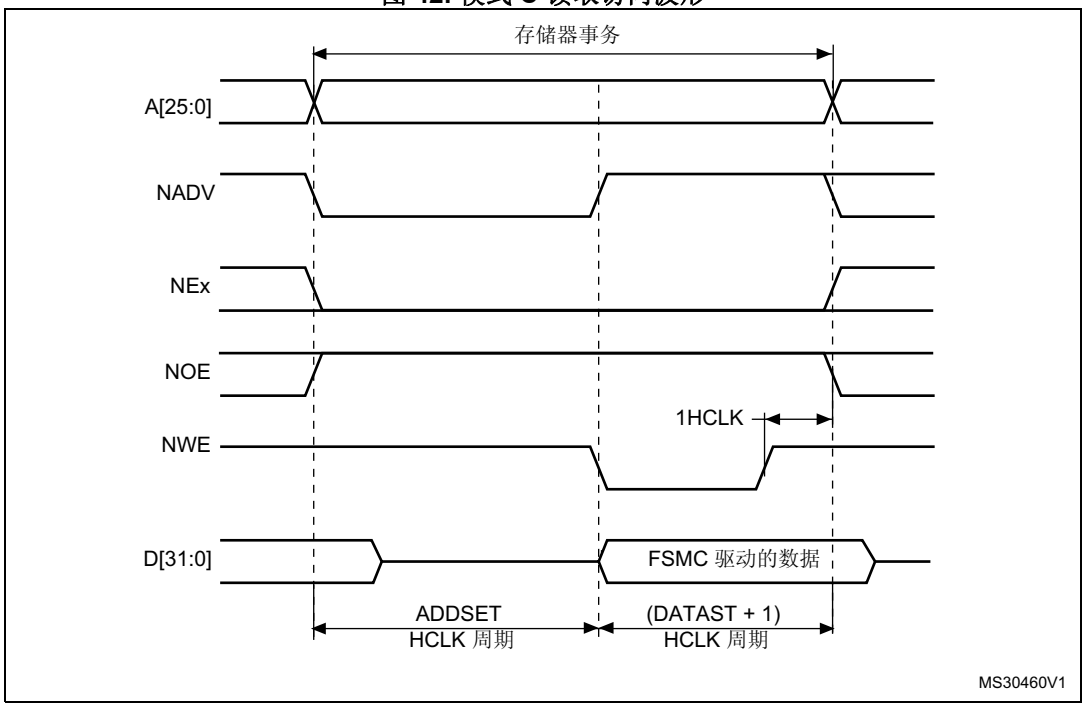


图 42. 模式 C 读取访问波形



与模式 1 的不同之处在于 NOE 的切换与独立的读取和写入时序。

表 69. FMC_BCRx 位字段

位号	位名	要设置的值
31-22	保留	0x000
21	WFDIS	根据需要进行设置
20	CCLKEN	根据需要进行设置
19	CBURSTRW	0x0（对异步模式没有影响）
18:16	CPSIZE	0x0（对异步模式没有影响）
15	ASYNCWAIT	如果存储器支持该特性，则置为 1。否则，保持为 0。
14	EXTMOD	0x1
13	WAITEN	0x0（对异步模式没有影响）
12	WREN	根据需要进行设置
11	WAITCFG	无关
10	保留	0x0
9	WAITPOL	仅当位 15 为 1 时才有意义
8	BURSTEN	0x0
7	保留	0x1
6	FACCEN	0x1

表 69. FMC_BCRx 位字段（续）

位号	位名	要设置的值
5-4	MWID	根据需要进行设置
3-2	MTYP	0x02 (NOR Flash)
1	MUXEN	0x0
0	MBKEN	0x1

表 70. FMC_BTRx 位字段

位号	位名	要设置的值
31:30	保留	0x0
29-28	ACCMOD	0x2
27-24	DATLAT	0x0
23-20	CLKDIV	0x0
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	写入访问第二个阶段的持续时间（DATAST+1 个 HCLK 周期）
7-4	ADDHLD	无关
3-0	ADDSET	写入访问第一个阶段的持续时间（ADDSET+1 个 HCLK 周期）。ADDSET 最小值为 0。

表 71. FMC_BWTRx 位字段

位号	位名	要设置的值
31:30	保留	0x0
29-28	ACCMOD	0x2
27-24	DATLAT	无关
23-20	CLKDIV	无关
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	写入访问第二个阶段的持续时间（DATAST 个 HCLK 周期）。
7-4	ADDHLD	无关
3-0	ADDSET	写入访问第一个阶段的持续时间（ADDSET 个 HCLK 周期）。ADDSET 最小值为 0。

模式 D - 扩展地址异步访问

图 43. ModeD 读取访问波形

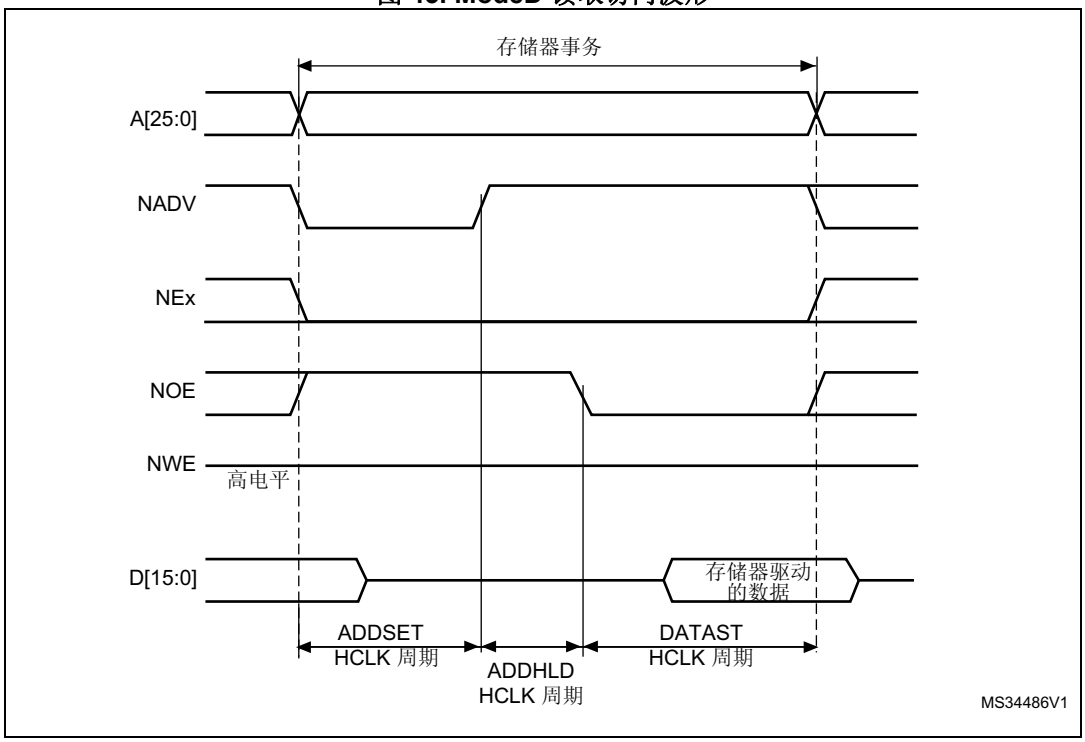
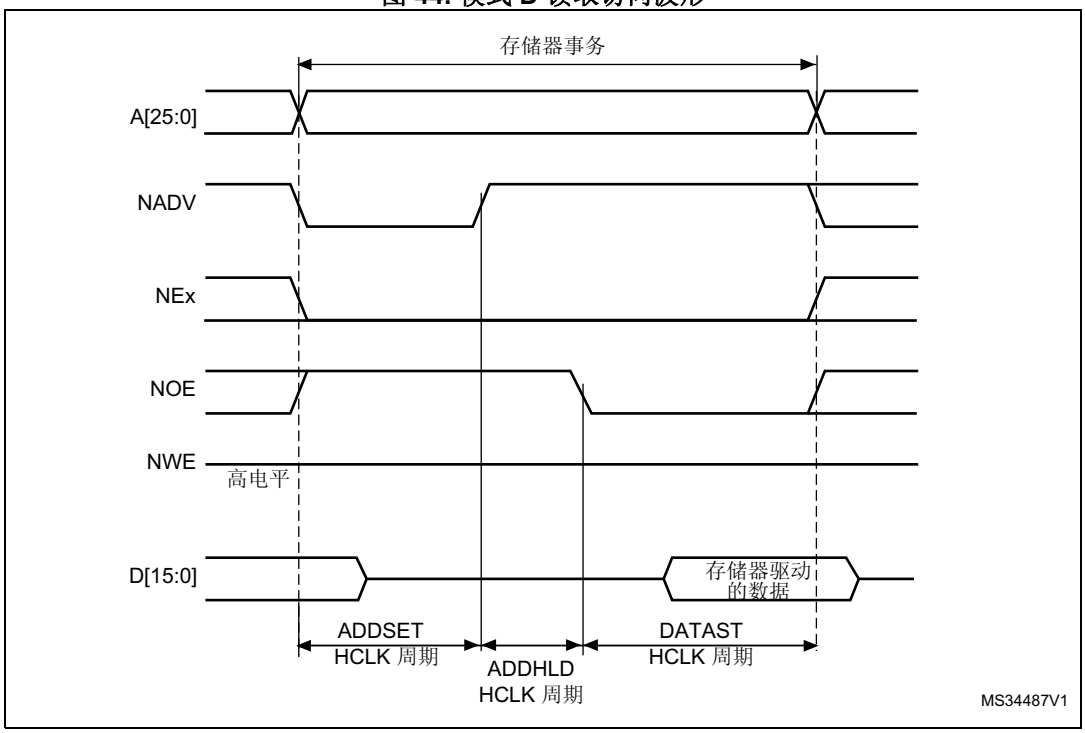


图 44. 模式 D 读取访问波形



与模式 1 的不同之处在于 NADV 变化后 NOE 的切换与独立的读取和写入时序。

表 72. FMC_BCRx 位字段

位号	位名	要设置的值
31-22	保留	0x000
21	WFDIS	根据需要进行设置
20	CCLKEN	根据需要进行设置
19	CBURSTRW	0x0（对异步模式没有影响）
18:16	CPSIZE	0x0（对异步模式没有影响）
15	ASYNCWAIT	如果存储器支持该特性，则置为 1。否则，保持为 0。
14	EXTMOD	0x1
13	WAITEN	0x0（对异步模式没有影响）
12	WREN	根据需要进行设置
11	WAITCFG	无关
10	保留	0x0
9	WAITPOL	仅当位 15 为 1 时才有意义
8	BURSTEN	0x0
7	保留	0x1
6	FACCEN	根据存储器支持情况进行设置
5-4	MWID	根据需要进行设置
3-2	MTYP	根据需要进行设置
1	MUXEN	0x0
0	MBKEN	0x1

表 73. FMC_BTRx 位字段

位号	位名	要设置的值
31:30	保留	0x0
29-28	ACCMOD	0x3
27-24	DATLAT	无关
23-20	CLKDIV	无关
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	写入访问第二个阶段的持续时间（DATAST+1 个 HCLK 周期）
7-4	ADDHLD	读取访问中间阶段的持续时间（ADDHLD 个 HCLK 周期）。
3-0	ADDSET	写入访问第一个阶段的持续时间（ADDSET+1 个 HCLK 周期）。ADDSET 的最小值为 1。

表 74. FMC_BWTRx 位字段

位号	位名	要设置的值
31:30	保留	0x0
29-28	ACCMOD	0x3
27-24	DATLAT	无关
23-20	CLKDIV	无关
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	第二个访问阶段的持续时间（写入访问为 DATAST+1 个 HCLK 周期）
7-4	ADDHLD	写入访问中间阶段的持续时间（ADDHLD 个 HCLK 周期）
3-0	ADDSET	写入访问第一个阶段的持续时间（ADDSET 个 HCLK 周期）。ADDSET 的最小值为 1。

复用模式 - 复用异步访问 NOR Flash

图 45. 复用读取访问波形

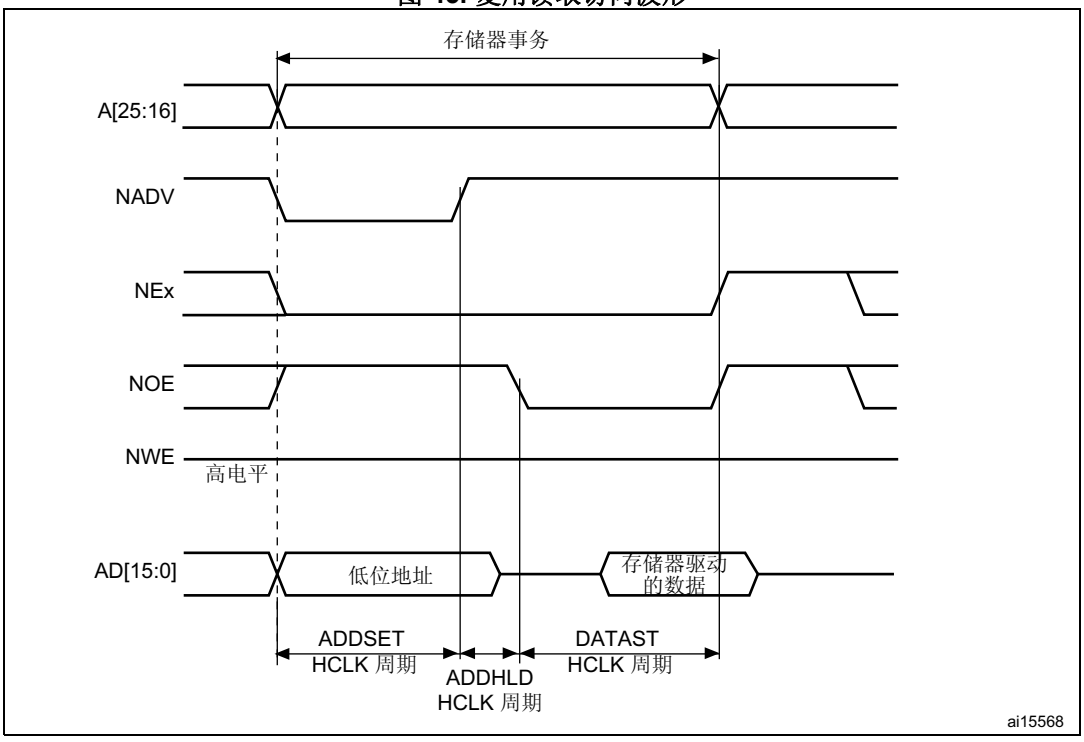
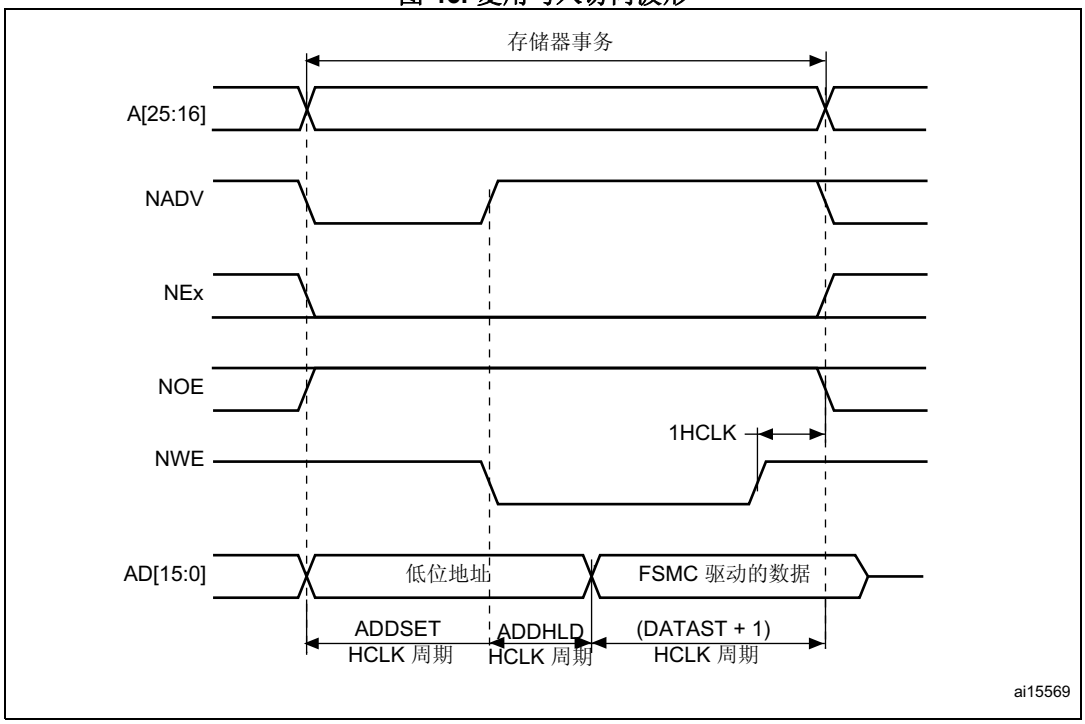


图 46. 复用写入访问波形



与模式 D 的不同之处在于在数据总线上驱动低地址字节。

表 75. FMC_BCRx 位字段

位号	位名	要设置的值
31-22	保留	0x000
21	WFDIS	根据需要进行设置
20	CCLKEN	根据需要进行设置
19	CBURSTRW	0x0（对异步模式没有影响）
18:16	CPSIZE	0x0（对异步模式没有影响）
15	ASYNCWAIT	如果存储器支持该特性，则置为 1。否则，保持为 0。
14	EXTMOD	0x0
13	WAITEN	0x0（对异步模式没有影响）
12	WREN	根据需要进行设置
11	WAITCFG	无关
10	保留	0x0
9	WAITPOL	仅当位 15 为 1 时才有意义
8	BURSTEN	0x0
7	保留	0x1
6	FACCEN	0x1

表 75. FMC_BCRx 位字段（续）

位号	位名	要设置的值
5-4	MWID	根据需要进行设置
3-2	MTYP	0x2 (NOR Flash)
1	MUXEN	0x1
0	MBKEN	0x1

表 76. FMC_BTRx 位字段

位号	位名	要设置的值
31:30	保留	0x0
29-28	ACCMOD	0x0
27-24	DATLAT	无关
23-20	CLKDIV	无关
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	第二个访问阶段的持续时间（读取访问为 DATAST 个 HCLK 周期，写入访问为 DATAST+1 个 HCLK 周期）。
7-4	ADDHLD	访问中间阶段的持续时间（ADDHLD 个 HCLK 周期）。
3-0	ADDSET	第一个访问阶段的持续时间（ADDSET 个 HCLK 周期）。ADDSET 的最小值为 1。

异步访问中的 WAIT 管理

如果异步存储器发出 WAIT 信号，指示尚未准备好接受或提供数据，则 FMC_BCRx 寄存器中的 ASYNCWAIT 位必须置 1。

如果 WAIT 信号处于有效状态（电平高低取决于 WAITPOL 位），则由 DATAST 位控制的第二个访问阶段（数据建立阶段）将延长，直到 WAIT 变为无效状态。与数据建立阶段不同，由 ADDSET 和 ADDHLD 位控制的第一个访问阶段（地址建立和地址保持阶段）对 WAIT 不敏感，因此第一个访问阶段不会延长。

必须配置数据建立阶段，以便在存储器事务结束前 4 个 HCLK 周期检测到 WAIT。必须考虑以下情况：

1. 存储器发出的 WAIT 信号和 NOE/NWE 信号对齐：

$$DATAST \geq (4 \times HCLK) + max_wait_assertion_time$$

2. 存储器发出的 WAIT 信号和 NEx 对齐（或者 NOE/NWE 信号不翻转）：

   如果

$$max_wait_assertion_time > address_phase + hold_phase$$

   那么：

$$DATAST \geq (4 \times HCLK) + (max_wait_assertion_time - address_phase - hold_phase)$$

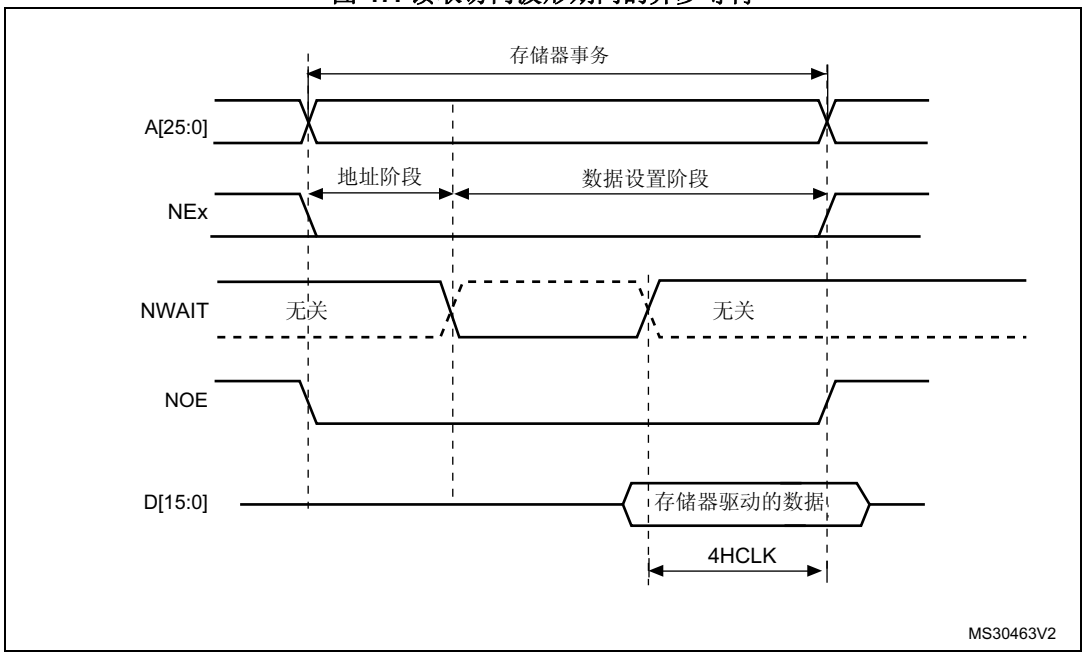
   否则

$$DATAST \geq 4 \times HCLK$$

   其中，max_wait_assertion_time 是在 NEx/NOE/NWE 变为低电平后存储器使能 WAIT 信号所花费的最长时间。

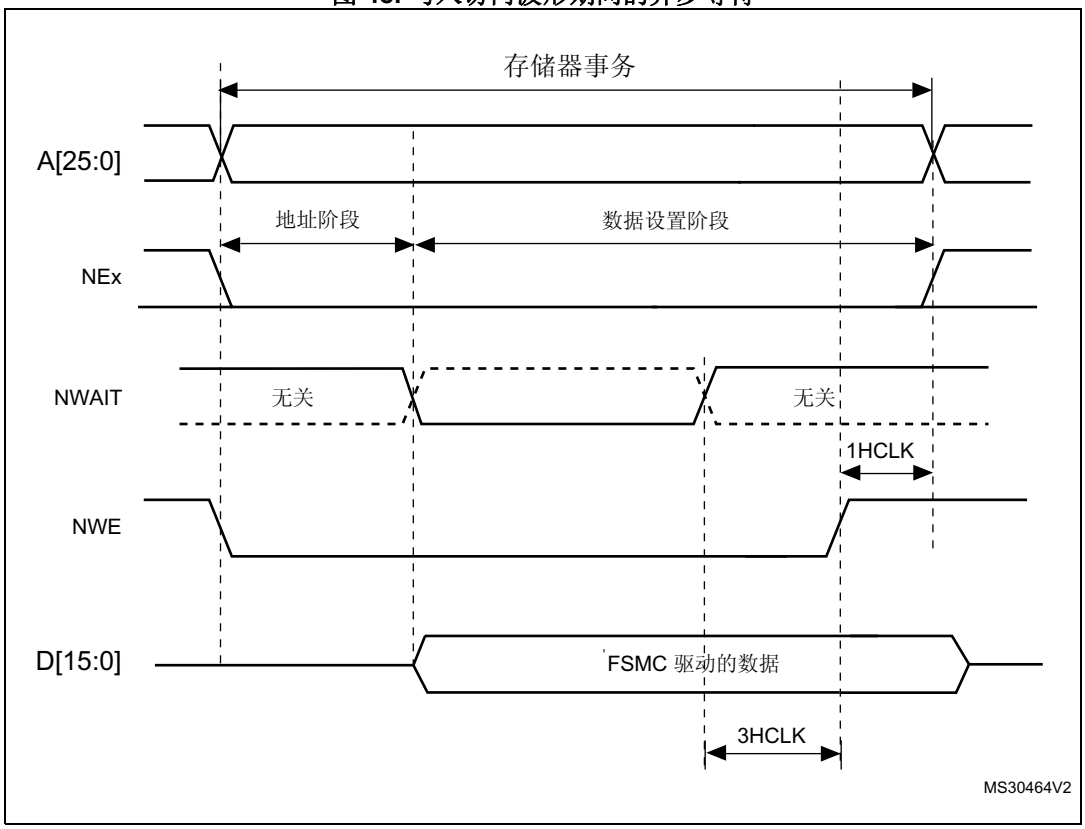
图 47 和*图 48* 显示了异步存储器释放 WAIT 之后，在存储器访问阶段增加的 HCLK 时钟周期的个数（与上述情况无关）。

图 47. 读取访问波形期间的异步等待



1. NWAIT 极性取决于 FMC_BCRx 寄存器中的 WAITPOL 位设置。

图 48. 写入访问波形期间的异步等待



1. NWAIT 极性取决于 FMC_BCRx 寄存器中的 WAITPOL 位设置。

13.5.5 同步事务

存储器时钟 FMC_CLK 是 HCLK 的约数。它取决于 CLKDIV 的值和 MWID/AHB 数据大小，如以下公式所示：

$$\text{FMC_CLK 分频比} = \max(\text{CLKDIV} + 1, \text{MWID(AHB 数据大小)})$$

MWID 为 16 位或 8 位时，FMC_CLK 分频比始终由 CLKDIV 的设定值定义。

MWID 为 32 位时，FMC_CLK 分频比还取决于 AHB 数据大小。

示例：

- CLKDIV = 1，MWID = 32 位，AHB 数据大小 = 8 位时，FMC_CLK = HCLK/4。
- CLKDIV = 1，MWID = 16 位，AHB 数据大小 = 8 位时，FMC_CLK = HCLK/2。

NOR Flash 指定了从 NADV 使能到 CLK 高电平的最短时间。为了符合这一限制，FMC 不会在同步访问的第一个内部时钟周期内（NADV 使能之前）将时钟发到存储器。这样可以确保存储器时钟的上升沿出现在 NADV 低脉冲的中间。

数据延迟与 NOR 延迟

数据延迟是对数据进行采样之前需要等待的周期数。DATLAT 的值必须与 NOR FLASH 配置寄存器中指定的延迟值一致。当数据延迟计数中 NADV 为低电平时，FMC 不会计入时钟周期。

警告： 一些 NOR Flash 将 NADV 低电平周期计入数据延迟计数，这样 NOR Flash 延迟和 FMC DATLAT 参数之间的确切关系可以是以下任一种：

- NOR Flash 延迟 = (DATLAT + 2) 个 CLK 时钟周期
- NOR Flash 延迟 = (DATLAT + 3) 个 CLK 时钟周期

近来有一些存储器会在延迟阶段使能 NWAIT。在这种情况下，可以将 DATLAT 设置为最小值。然后，FMC 会对数据进行采样，并且等待足够长的时间来评估数据是否有效。这样，FMC 就能检测到存储器存在延迟的时间，从而处理真实数据。

其他存储器不会在延迟期间使能 NWAIT。在这种情况下，必须正确设置 FMC 和存储器的延迟，否则可能会将无效数据误用为有效数据，或者在存储器访问初始阶段丢失有效数据。

单次突发传输

当所选存储区域配置为同步突发模式时，例如，如果向 16 位存储器请求了一个 AHB 单次突发事务，则 FMC 会执行长度为 1 的突发事务（如果 AHB 传输为 16 位）或者长度为 2 的突发事务（如果 AHB 传输为 32 位），然后在最后一个数据选通时禁止片选信号。

与异步读取操作相比，就周期而言这并不是最有效的传输方法。但是，随机异步读取需要先重新编程存储器访问模式，这样总时间会更长。

越过 Cellular RAM 1.5 的边界页

Cellular RAM 1.5 不允许突发访问越过页边界。根据存储器页大小配置 FMC_BCR1 寄存器中的 CPSIZE 位来达到存储器页大小时，FMC 控制器会自动分离突发访问。

等待管理

对于同步 NOR Flash，会在配置的延迟周期（相当于 (DATLAT+2) 个 CLK 时钟周期）之后对 NWAIT 进行评估。

如果 NWAIT 有效（WAITPOL = 0 时为低电平，WAITPOL = 1 时为高电平），会插入等待状态，直到 NWAIT 无效（WAITPOL = 0 时为高电平，WAITPOL = 1 时为低电平）。

当 NWAIT 无效时，数据将立即（位 WAITCFG = 1）或在下一个时钟边沿（位 WAITCFG = 0）被视为有效。

通过 NWAIT 信号插入等待周期期间，控制器会继续将时钟脉冲发送到存储器，保持片选和输出使能信号有效。但不将数据视为有效。

突发模式下，NOR Flash NWAIT 信号有两种时序配置：

- Flash 在等待周期之前一个数据周期发出 NWAIT 信号（复位后的默认值）。
- Flash 在等待周期期间发出 NWAIT 信号。

FMC 支持这两种 NOR Flash 等待周期配置，通过 FMC_BCRx 寄存器的 WAITCFG 位 (x = 0..3) 针对每个片选进行配置。

图 49. 等待配置波形

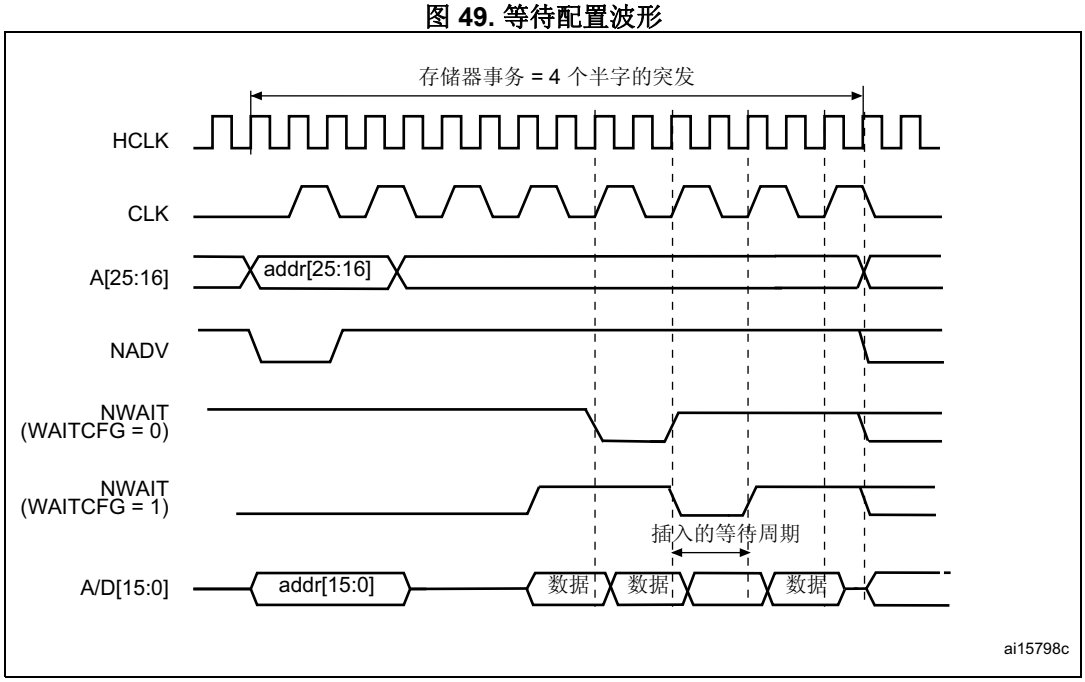
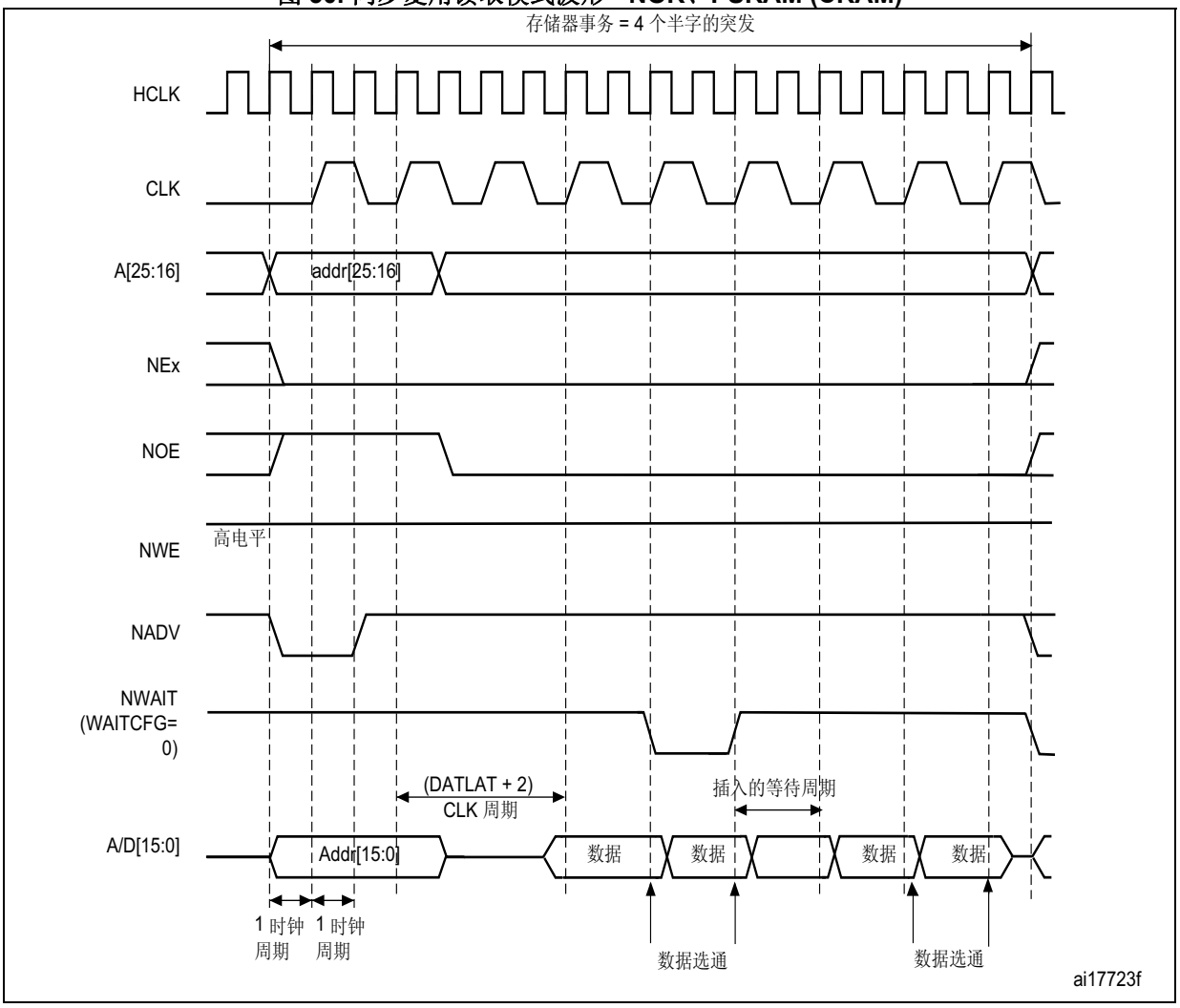


图 50. 同步复用读取模式波形 - NOR、PSRAM (CRAM)



1. 字节通道输出 NBL 未显示，它们对于 NOR 访问保持高电平，对于 PSRAM (CRAM) 访问则保持低电平。

表 77. FMC_BCRx 位字段

位号	位名	要设置的值
31-21	保留	0x000
21	WFDIS	根据需要进行设置
20	CCLKEN	根据需要进行设置
19	CBURSTRW	对同步读取没有影响
18:16	CPSIZE	0x0（对异步模式没有影响）
15	ASYNCWAIT	0x0
14	EXTMOD	0x0
13	WAITEN	在存储器支持该特性的情况下置位为 1，否则保持为 0
12	WREN	对同步读取没有影响
11	WAITCFG	是否置位视存储器情况而定

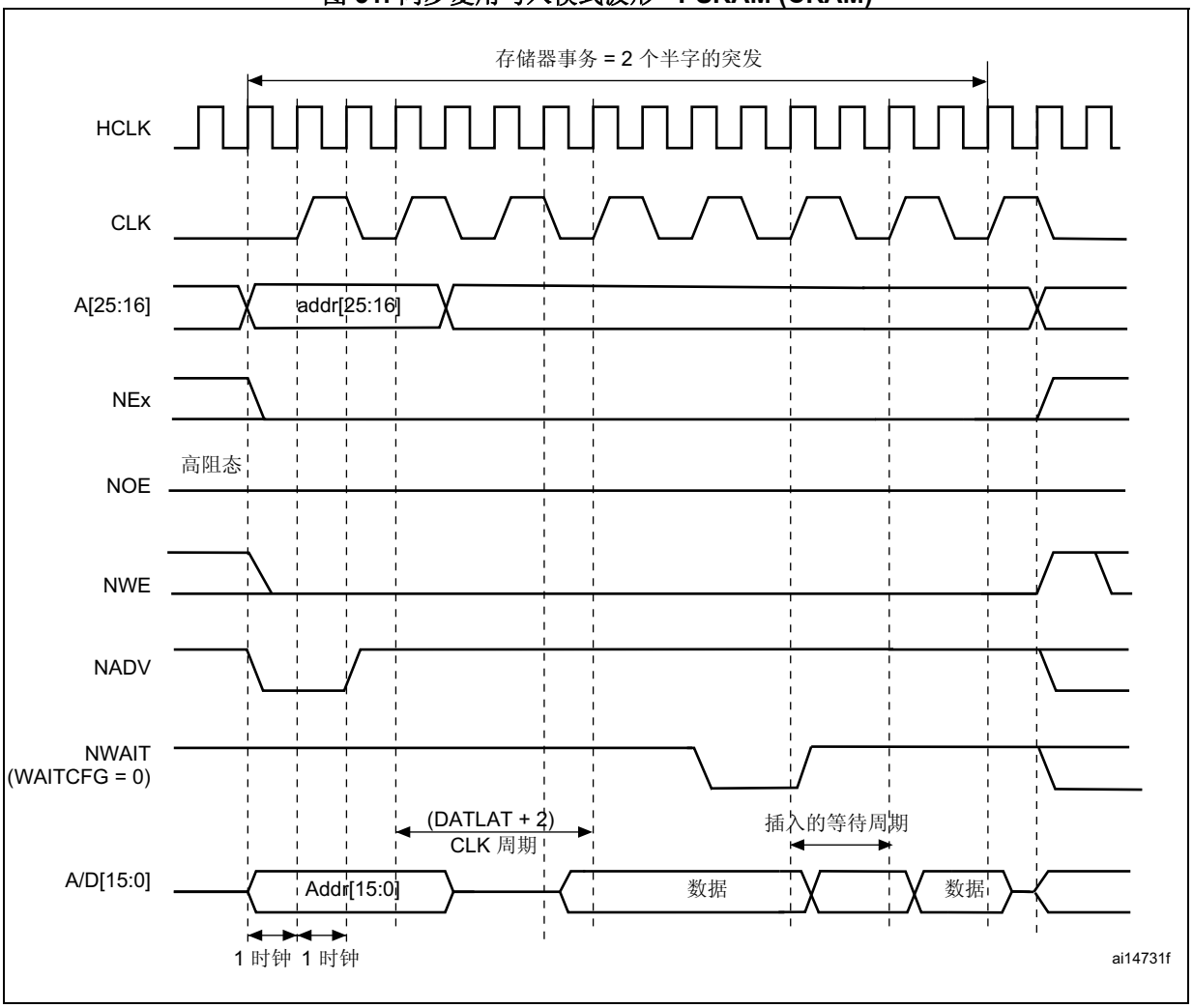
表 77. FMC_BCRx 位字段（续）

位号	位名	要设置的值
10	保留	0x0
9	WAITPOL	是否置位视存储器情况而定
8	BURSTEN	0x1
7	保留	0x1
6	FACCEN	在存储器支持的情况下置位 (NOR Flash)
5-4	MWID	根据需要进行设置
3-2	MTYP	0x1 或 0x2
1	MUXEN	根据需要进行设置
0	MBKEN	0x1

表 78. FMC_BTRx 位字段

位号	位名	要设置的值
31:30	保留	0x0
29:28	ACCMOD	0x0
27-24	DATLAT	数据延迟
27-24	DATLAT	数据延迟
23-20	CLKDIV	0x0，使 CLK = HCLK（不支持） 0x1，使 CLK = 2 × HCLK ..
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	无关
7-4	ADDHLD	无关
3-0	ADDSET	无关

图 51. 同步复用写入模式波形 - PSRAM (CRAM)



1. 存储器必须提前一个周期发出 NWAIT 信号，相应地 WAITCFG 必须编程为 0。
2. 字节通道 (NBL) 输出未显示，当 NEx 有效时它们保持低电平。

表 79. FMC_BCRx 位字段

位号	位名	要设置的值
31-20	保留	0x000
21	WFDIS	根据需要进行设置
20	CCLKEN	根据需要进行设置
19	CBURSTRW	0x1
18:16	CPSIZE	根据需要进行设置（CRAM 1.5 为 0x1）
15	ASYNCWAIT	0x0
14	EXTMOD	0x0
13	WAITEN	在存储器支持该特性的情况下置位为 1，否则保持为 0。
12	WREN	0x1

表 79. FMC_BCRx 位字段（续）

位号	位名	要设置的值
11	WAITCFG	0x0
10	保留	0x0
9	WAITPOL	是否置位视存储器情况而定
8	BURSTEN	对同步写入没有影响
7	保留	0x1
6	FACCEN	根据存储器支持情况进行设置
5-4	MWID	根据需要进行设置
3-2	MTYP	0x1
1	MUXEN	根据需要进行设置
0	MBKEN	0x1

表 80. FMC_BTRx 位字段

位号	位名	要设置的值
31-30	保留	0x0
29:28	ACCMOD	0x0
27-24	DATLAT	数据延迟
23-20	CLKDIV	0x0，使 CLK = HCLK（不支持） 0x1，使 CLK = 2 × HCLK
19-16	BUSTURN	NEx 变为高电平到 NEx 变为低电平之间的时间 (BUSTURN HCLK)
15-8	DATAST	无关
7-4	ADDHLD	无关
3-0	ADDSET	无关

13.5.6 NOR/PSRAM 控制寄存器

SRAM/NOR-Flash 片选控制寄存器 1..4 (FMC_BCR1..4)

SRAM/NOR-Flash chip-select control registers 1..4

偏移地址：8 * (x – 1)，x = 1...4

复位值：对于存储区域 1 为 0x0000 30DB，对于存储区域 2 到 4 为 0x0000 30D2

该寄存器包含每个存储区域的控制信息，用于 SRAM、PSRAM 和 NOR Flash。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WFDIS	CCLK EN	CBURST RW	CPSIZE[2:0]		
										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASYNC WAIT	EXT MOD	WAIT EN	WREN	WAIT CFG	Res.	WAIT POL	BURST EN	Res.	FACC EN	MWID		MTYP		MUX EN	MBK EN
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 **WFDIS**：写 FIFO 禁止 (Write FIFO Disable)

此位可禁止 FMC 控制器使用的写 FIFO。

0：使能写 FIFO（复位后的默认值）

1：禁止写 FIFO

注： FMC_BCR2..4 寄存器的 WFDIS 位为“无关”位。只能通过 FMC_BCR1 寄存器使能。

位 20 **CCLKEN**：连续时钟使能 (Continuous Clock Enable)。

该位可使能向外部存储器器件输出 FMC_CLK 时钟。

0：仅在同步存储器访问（读/写事务）期间生成 FMC_CLK。FMC_CLK 时钟分频比由 FMC_BCRx 寄存器中配置的 CLKDIV 值指定（复位后的默认值）。

1：异步和同步访问期间连续生成 FMC_CLK。CCLKEN 置 1 将激活 FMC_CLK 时钟。

注： FMC_BCR2..4 寄存器的 CCLKEN 位为“无关”位。只能通过 FMC_BCR1 寄存器使能。存储区域 1 必须配置为支持同步模式才能生成 FMC_CLK 连续时钟。

注： 如果 CCLKEN 位置 1，则 FMC_CLK 时钟分频比将由 FMC_BTR1 寄存器的 CLKDIV 值指定。FMC_BWTR1 寄存器中的 CLKDIV 为无关位。

注： 如果采用同步模式且 CCLKEN 位置 1，则与存储区域 1 之外的其它存储区域相连的同步存储器将共用同一个时钟源（FMC_BTR2..4 寄存器和 FMC_BWTR2..4 寄存器中的 CLKDIV 值对其它存储区域不起作用。）

位 19 **CBURSTRW**：突发写使能 (Write burst enable)。

PSRAM (CRAM) 在突发模式下工作时，该位可使能同步写访问。同步读取访问的使能位为 FMC_BCRx 寄存器中的 BURSTEN 位。

0：始终在异步模式下写入

1：在同步模式下写入。

位 18:16 **CPSIZE[2:0]**：CRAM 页大小 (CRAM page size)。

这些位用于 Cellular RAM 1.5，Cellular RAM 1.5 不允许突发访问越过页面之间的地址边界。如果配置这些位，FMC 控制器会在达到存储器页大小后自动分离突发访问（请参见存储器数据手册了解页大小）。

000：越过页边界后不进行突发分离（复位后的默认值）
001：128 字节
010：256 字节
011：512 字节
100：1024 字节
其它：保留

位 15 **ASYNCWAIT**：异步传输期间的等待信号 (Wait signal during asynchronous transfers)。

该位可使能/禁止 FMC 使用等待信号，即使在异步协议期间也有效。

0：运行异步协议时不考虑 NWAIT 信号（复位后的默认值）
1：运行异步协议时考虑 NWAIT 信号

位 14 **EXTMOD**：扩展模式使能 (Extended mode enable)。

FMC 可对 FMC_BWTR 寄存器中非复用异步访问的写入时序进行配置，此配置由 EXTMOD 位使能，进而使读取和写入操作采用不同时序。

0：不考虑 FMC_BWTR 寄存器中的值（复位后的默认值）
1：考虑 FMC_BWTR 寄存器中的值

注： 如果禁用扩展模式，FMC 可以在模式 1 或模式 2 下运行，如下所述：

- *当选择 SRAM/PSRAM 存储器类型时，模式 1 为默认模式（MTYP = 0x0 或 0x01）*
- *当选择 NOR 存储器类型时，模式 2 为默认模式（MTYP = 0x10）。*

位 13 **WAITEN**：等待使能位 (Wait enable bit)。

该位可使能/禁止在同步模式下访问存储器时通过 NWAIT 信号插入等待周期。

0：禁止 NWAIT 信号（不考虑其电平，不在配置过的 Flash 延迟周期后插入等待周期）
1：使能 NWAIT 信号（考虑其电平，如果使能，在配置过的延迟周期后插入等待周期）（复位后的默认值）

位 12 **WREN**：写入使能位 (Write enable bit)。

该位指示 FMC 是否使能/禁止在存储区域内写入：

0：FMC 禁止在存储区域内写入，如果进行写操作将报告 AHB 错误，
1：FMC 使能在存储区域内写入（复位后的默认值）。

位 11 **WAITCFG**：等待时序配置 (Wait timing configuration)。

NWAIT 信号指示存储器中的数据是否有效，或者在同步模式下访问存储器时是否必须插入等待周期。该配置位决定存储器是在等待周期之前的一个时钟周期还是等待周期期间使能 NWAIT：

0：NWAIT 信号在等待周期之前的一个数据周期有效（复位后的默认值），
1：NWAIT 信号在等待周期期间有效（不适用于 PSRAM）。

位 10 保留，必须保持复位值

位 9 **WAITPOL**：等待信号极性位 (Wait signal polarity bit)。

定义同步或异步模式下使用的存储器的等待信号极性：

0：NWAIT 低电平有效（复位后的默认值），
1：NWAIT 高电平有效。

位 8 **BURSTEN**：突发使能位 (Burst enable bit)。

该位可使能/禁止同步读取访问。该位仅对突发模式下工作的同步存储器有效：

0：禁止突发模式（复位后的默认值）。在异步模式下进行读取访问。

1：使能突发模式。在同步模式下进行读取访问。

位 7 保留，必须保持复位值

位 6 **FACCEN**：Flash 访问使能 (Flash access enable)。

使能 NOR Flash 访问操作。

0：禁止相应的 NOR Flash 访问

1：使能相应的 NOR Flash 访问（复位后的默认值）

位 5:4 **MWID[1:0]**：存储器数据总线宽度 (Memory data bus width)。

定义外部存储器器件宽度，对所有类型的存储器均有效。

00：8 位

01：16 位（复位后的默认值）

10：32 位

11：保留

位 3:2 **MTYP[1:0]**：存储器类型 (Memory type)。

定义与相应存储区域相连的外部存储器类型：

00：SRAM（对于存储区域 2...4，复位后的默认值）

01：PSRAM (CRAM)

10：NOR Flash/OneNAND Flash（对于存储区域 1，复位后的默认值）

11：保留

位 1 **MUXEN**：地址/数据复用使能位 (Address/data multiplexing enable bit)。

该位置 1 时，地址和数据值在数据总线上复用，仅对 NOR 和 PSRAM 存储器有效：

0：地址/数据非复用

1：地址/数据在数据总线上复用（复位后的默认值）

位 0 **MBKEN**：存储区域使能位 (Memory bank enable bit)。

使能存储区域。复位后使能存储区域 1，其它存储区域均禁止。访问禁止的存储区域会引起 AHB 总线上的错误。

0：禁止相应的存储区域

1：使能相应的存储区域

SRAM/NOR-Flash 片选时序寄存器 1..4 (FMC_BTR1..4)

SRAM/NOR-Flash chip-select timing registers 1..4

偏移地址：0x04 + 8 * (x – 1)，x = 1..4

复位值：0x0FFF FFFF

该寄存器包含每个存储区域的控制信息，用于 SRAM、PSRAM 和 NOR Flash。如果 FMC_BCRx 寄存器中的 EXTMOD 位置 1，该寄存器将和另外一个寄存器配合来配置写入和读取访问，也就是说有 2 个寄存器可用：一个用于配置读取访问 （此寄存器），另一个用于配置写入访问（FMC_BWTRx 寄存器）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	ACCMOD		DATLAT				CLKDIV				BUSTURN			
		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
DATAST								ADDHLD				ADDSET			
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 29:28 **ACCMOD[1:0]**：访问模式 (Access mode)

指定异步访问模式，如时序图所示。仅当 FMC_BCRx 寄存器中的 EXTMOD 位置 1 时，这些位才有效。

00：访问模式 A
01：访问模式 B
10：访问模式 C
11：访问模式 D

位 27:24 **DATLAT[3:0]**：同步存储器的数据延迟 (Data latency for synchronous memory)（请参见位说明下的注释）

对于使能了读/写突发模式（BURSTEN/CBURSTRW 位置 1）的同步访问，该字段定义读写首个数据前要发送给存储器的存储器时钟周期数 (+2)：

该时序参数以 FMC_CLK 周期而非 HCLK 周期表示。

该值与异步访问模式无关。

0000：首次突发访问时，2 个 CLK 时钟周期的数据延迟
1111：首次突发访问时，17 个 CLK 时钟周期的数据延迟（复位后的默认值）

位 23:20 **CLKDIV[3:0]**：FMC_CLK 信号的时钟分频比 (Clock divide ratio (for FMC_CLK signal))

定义 FMC_CLK 时钟输出信号的周期，以 HCLK 周期数表示：

0000：保留
0001：FMC_CLK 周期 = 2 × HCLK 周期
0010：FMC_CLK 周期 = 3 × HCLK 周期
1111：FMC_CLK 周期 = 16 × HCLK 周期（复位后的默认值）

在异步 NOR Flash、SRAM 或 PSRAM 访问模式下，该值为无关值。

注： 有关 FMC_CLK 分频比公式的信息，请参见第 13.5.5 节：同步事务。

位 19:16 **BUSTURN[3:0]**：总线周转阶段的持续时间 (Bus turnaround phase duration)

通过软件写入这些位可在写-读（和读-写）事务结束时添加延迟。

总线周转延迟还会插入对不同存储区域进行的两个连续的读/写事务之间。当两个连续的读事务指向同一个 FMC 存储区域时，无论所配置的 BUSTURN 时序为何，均只会插入 1 个 HCLK 时钟周期。当两个连续的写事务指向同一个 FMC 存储区域时，不会插入总线周转延迟。

总线周转延迟可以匹配连续事务之间的最短时间（t_{EHEL} 由 NEx 高电平变为 NEx 低电平）以及存储器在读取访问后释放数据总线所需的最长时间 (t_{EHQZ})：

(BUSTRUN + 1)HCLK 周期 ≥ $t_{EHELmin}$ 和 (BUSTRUN + 2)HCLK 周期 ≥ $t_{EHQZmax}$（如果 EXTMOD = “0”）

(BUSTRUN + 2)HCLK 周期 ≥ max（$t_{EHELmin}$，$t_{EHQZmax}$）（如果 EXTMOD = “1”）。

0000：BUSTURN 阶段的持续时间 = 增加 0 个 HCLK 时钟周期

...

1111：BUSTURN 阶段的持续时间 = 15 × HCLK 时钟周期（复位后的默认值）

位 15:8 **DATAST[7:0]**：数据阶段的持续时间 (Data-phase duration)

通过软件写入这些位可定义数据阶段的持续时间（请参见*图 34* 到*图 46*），适用于异步访问模式：

0000 0000：保留

0000 0001：DATAST 阶段的持续时间 = 1 × HCLK 时钟周期

0000 0010：DATAST 阶段的持续时间 = 2 × HCLK 时钟周期

...

1111 1111：DATAST 阶段的持续时间 = 255 × HCLK 时钟周期（复位后的默认值）

有关每种存储器类型和访问模式数据阶段持续时间的信息，请参见相应图片（*图 34* 到*图 46*）。

例如：在模式 1、写入访问以及 DATAST=1 条件下，数据阶段的持续时间 = DATAST+1 = 2 个 HCLK 时钟周期。

注：　在同步访问模式下，该值为无关值。

位 7:4 **ADDHLD[3:0]**：地址保持阶段的持续时间 (Address-hold phase duration)

通过软件写入这些位可定义*地址保持*阶段的持续时间（请参见*图 34* 到*图 46*），适用于模式 D 或复用访问：

0000：保留

0001：ADDHLD 阶段的持续时间 = 1 × HCLK 时钟周期

0010：ADDHLD 阶段的持续时间 = 2 × HCLK 时钟周期

...

1111：ADDHLD 阶段的持续时间 = 15 × HCLK 时钟周期（复位后的默认值）

有关每种访问模式地址保持阶段持续时间的信息，请参见相应图片（*图 34* 到*图 46*）。

注：　在同步访问模式下，该值不使用，因为地址保持阶段的持续时间始终是 1 个存储器时钟周期。

位 3:0 **ADDSET[3:0]**：地址设置阶段的持续时间 (Address setup phase duration)

通过软件写入这些位可定义*地址设置*阶段的持续时间（请参见*图 34* 到*图 46*），适用于 SRAM、ROM、异步 NOR Flash 和 PSRAM：

0000：ADDSET 阶段的持续时间 = 0 × HCLK 时钟周期

...

1111：ADDSET 阶段的持续时间 = 15 × HCLK 时钟周期（复位后的默认值）

有关每种访问模式地址设置阶段持续时间的信息，请参见相应图片（请参见 *图 34* 到*图 46*）。

注：　在同步访问模式下，该值为无关值。

复用模式或模式 D 下，ADDSET 的最小值为 1。

注： *PSRAM (CRAM) 由于内部刷新而导致数据延时时间长度不确定。因此，这些存储器会在整个延迟阶段发送 NWAIT 信号，以便按照需要延长延迟。*
对于 PSRAM (CRAM)，字段 DATLAT 必须设置为 0，这样 FMC 会立即退出延迟阶段，开始对存储器中的 NWAIT 采样，然后在存储器准备就绪后开始读取或写入。
此方法也适用于最新一代的同步 Flash，同早期 Flash 不同的是，此类 Flash 会发送 NWAIT 信号（检查所用的具体 Flash 数据表）。

SRAM/NOR-Flash 写入时序寄存器 1..4 (FMC_BWTR1..4)

SRAM/NOR-Flash write timing registers 1..4

偏移地址：0x104 + 8 * (x – 1)，x = 1...4

复位值：0x0FFF FFFF

此寄存器包含每个存储区域的控制信息， 用于 SRAM、PSRAM 和 NOR Flash。当 FMC_BCRx 寄存器中的 EXTMOD 位置 1 时，该寄存器将处于有效状态，可以进行写入访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	ACCMOD		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUSTURN			
		rw	rw									rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAST								ADDHLD				ADDSET			
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 29:28 **ACCMOD[1:0]**：访问模式 (Access mode)。
指定异步访问模式，如下一个时序图所示。仅当 FMC_BCRx 寄存器中的 EXTMOD 位置 1 时，这些位才有效。
00：访问模式 A
01：访问模式 B
10：访问模式 C
11：访问模式 D

位 27:20 保留，必须保持复位值

位 19:16 **BUSTURN[3:0]**：总线周转阶段的持续时间 (Bus turnaround phase duration)。
通过软件写入这些位可在写-读事务结束时增加一个延迟，进而匹配两个连续事务之间的最小时间（t_{EHEL} 由 ENx 高电平变为 ENx 低电平）：
(BUSTRUN + 1) HCLK 周期 ≥ $t_{EHELmin}$。
0000：BUSTURN 阶段的持续时间 = 增加 0 个 HCLK 时钟周期
...
1111：BUSTURN 阶段的持续时间 = 增加 15 个 HCLK 时钟周期（复位后的默认值）

位 15:8 **DATAST[7:0]**：数据阶段的持续时间 (Data-phase duration)。
通过软件写入这些位可定义数据阶段的持续时间（请参见*图 34* 到*图 46*），适用于异步 SRAM、PSRAM 和 NOR Flash 访问模式：
0000 0000：保留
0000 0001：DATAST 阶段的持续时间 = 1 × HCLK 时钟周期
0000 0010：DATAST 阶段的持续时间 = 2 × HCLK 时钟周期
...
1111 1111：DATAST 阶段的持续时间 = 255 × HCLK 时钟周期（复位后的默认值）

位 7:4 **ADDHLD[3:0]**：地址保持阶段的持续时间 (Address-hold phase duration)。
通过软件写入这些位可定义*地址保持*阶段的持续时间（请参见*图 43* 到*图 46*），适用于异步复用访问：
0000：保留
0001：ADDHLD 阶段的持续时间 = 1 × HCLK 时钟周期
0010：ADDHLD 阶段的持续时间 = 2 × HCLK 时钟周期
...
1111：ADDHLD 阶段的持续时间 = 15 × HCLK 时钟周期（复位后的默认值）
注：　在同步 NOR Flash 访问模式下，该值不使用，因为地址保持阶段的持续时间始终是 1 个 Flash 时钟周期。

位 3:0 **ADDSET[3:0]**：地址建立阶段的持续时间 (Address setup phase duration)。
通过软件写入这些位可定义以 HCLK 周期表示的*地址建立*阶段持续时间（请参见*图 34* 到*图 46*），适用于异步访问模式：
0000：ADDSET 阶段的持续时间 = 0 × HCLK 时钟周期
...
1111：ADDSET 阶段的持续时间 = 15 × HCLK 时钟周期（复位后的默认值）
注：　在同步访问模式下，该值不使用，因为地址建立阶段的持续时间始终是 1 个 Flash 时钟周期。
复用模式下，ADDSET 的最小值为 1。

13.6 NAND Flash 控制器

FMC 会生成相应的信号时序，用于驱动以下类型的设备：

- 8 位和 16 位 NAND Flash

NAND 存储区域通过专用的寄存器配置（*第 13.6.7 节*）。可编程的存储器参数包括访问时序（如*表 81* 所示）和 ECC 配置。

表 81. 可编程的 NAND Flash 访问参数

参数	功能	访问模式	单位	最小值	最大值
存储器建立时间	命令使能前地址建立所需时钟周期 (HCLK) 数	读/写	AHB 时钟周期 (HCLK)	1	255
存储器等待	命令使能的最小持续时间（按 HCLK 时钟周期计）	读/写	AHB 时钟周期 (HCLK)	2	255
存储器保持	命令禁止后，必须保持地址（如果进行了写访问还需保持数据）的时钟周期 (HCLK) 数	读/写	AHB 时钟周期 (HCLK)	1	254
存储器数据总线高阻态	开始进行写访问后，数据总线保持高阻状态期间的时钟周期 (HCLK) 数	写	AHB 时钟周期 (HCLK)	1	255

13.6.1 外部存储器接口信号

下表列出了通常用于连接 NAND Flash 的信号。

注： *前缀“N”标识低电平有效的信号。*

8 位 NAND Flash

表 82. 8 位 NAND Flash

FMC 信号名称	I/O	功能
A[17]	O	NAND Flash 地址锁存使能 (ALE) 信号
A[16]	O	NAND Flash 命令锁存使能 (CLE) 信号
D[7:0]	I/O	8 位复用双向地址/数据总线
NCE	O	片选
NOE(= NRE)	O	输出使能（存储器信号名称：读取使能，NRE）
NWE	O	写入使能
NWAIT/INT	I	输入 FMC 的 NAND Flash 就绪/繁忙信号

由于 FMC 能够管理足够多的地址周期，因此理论上不存在容量限制。

16 位 NAND Flash

表 83. 16 位 NAND Flash

FMC 信号名称	I/O	功能
A[17]	O	NAND Flash 地址锁存使能 (ALE) 信号
A[16]	O	NAND Flash 命令锁存使能 (CLE) 信号
D[15:0]	I/O	16 位复用双向地址/数据总线
NCE	O	片选
NOE(= NRE)	O	输出使能（存储器信号名称：读取使能，NRE）
NWE	O	写入使能
NWAIT/INT	I	输入 FMC 的 NAND Flash 就绪/繁忙信号

由于 FMC 能够管理足够多的地址周期，因此理论上不存在容量限制。

13.6.2 NAND Flash 支持的存储器和事务

表 84 介绍了所支持的设备、访问模式和事务。NAND Flash 控制器不允许（或不支持）的事务以灰色显示。

表 84. 支持的存储器和事务

设备	模式	R/W	AHB 数据大小	存储器数据大小	允许/不允许	注释
8 位 NAND	异步	R	8	8	是	-
	异步	W	8	8	是	-
	异步	R	16	8	是	分为 2 次 FMC 访问
	异步	W	16	8	是	分为 2 次 FMC 访问
	异步	R	32	8	是	分为 4 次 FMC 访问
	异步	W	32	8	是	分为 4 次 FMC 访问
16 位 NAND	异步	R	8	16	是	-
	异步	W	8	16	N	-
	异步	R	16	16	是	-
	异步	W	16	16	是	-
	异步	R	32	16	是	分为 2 次 FMC 访问
	异步	W	32	16	是	分为 2 次 FMC 访问

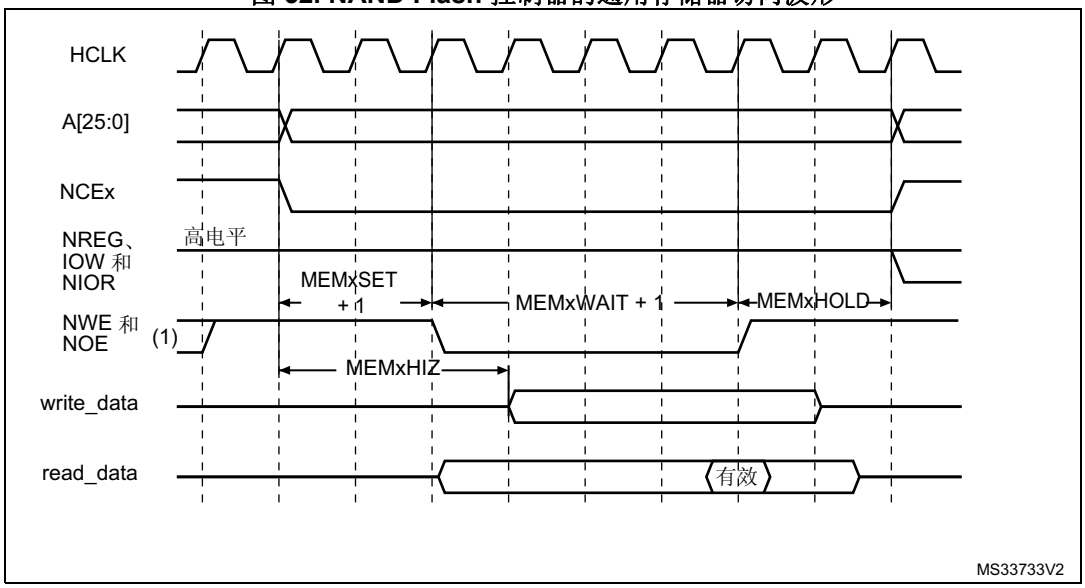
13.6.3 NAND Flash 的时序图

NAND Flash 存储区域通过以下一组寄存器进行管理：

- 控制寄存器：FMC_PCR
- 中断状态寄存器：FMC_SR
- ECC 寄存器：FMC_ECCR
- 通用存储器空间的时序寄存器：FMC_PMEM
- 特性存储器空间的时序寄存器：FMC_PATT

每个时序配置寄存器包含四个参数，其中三个参数用于定义 NAND Flash 的三个访问阶段的 HCLK 周期数，另一个参数用于定义进行写访问时开始驱动数据总线的时序。*图 52* 介绍了通用存储器访问的时序参数定义，特性存储器空间的访问时序与此类似。

图 52. NAND Flash 控制器的通用存储器访问波形



1. 进行写访问期间 NOE 保持高电平（无效）。进行读访问期间 NWE 保持高电平（无效）。

13.6.4 NAND Flash 操作

NAND Flash 设备的命令锁存使能 (CLE) 信号和地址锁存使能 (ALE) 信号由 FMC 控制器的地址信号驱动。这意味着要向 NAND Flash 发送命令或地址，CPU 必须对其存储器空间中的特定地址执行写操作。

从 NAND Flash 设备进行的典型页读取操作需要执行以下步骤：

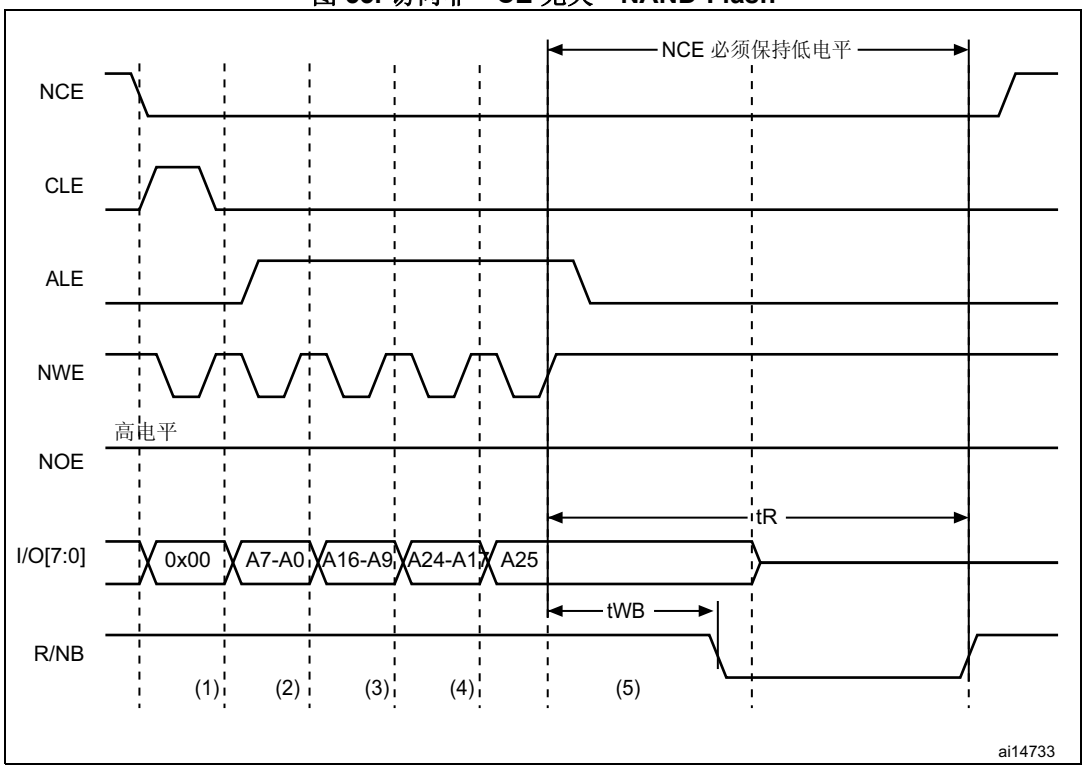
1. 根据 NAND Flash 的特性（PWID 位指示 NAND Flash 数据总线宽度，PTYP = 1，根据需要 PWAITEN = 0 或 1；有关时序配置，请参见*第 13.4.2 节：NAND Flash 地址映射*），配置 FMC_PCR 和 FMC_PMEM（对于某些设备，配置 FMC_PATT，请参见*第 13.6.5 节：NAND Flash 预等待功能*）寄存器，进而配置和使能相应的存储区域。
2. CPU 向通用存储器空间执行字节写操作，此时数据字节等于一个 Flash 命令字节（例如 Samsung NAND Flash 设备的字节为 0x00）。在写入选通期间（NWE 上为低电平脉冲），NAND Flash 的 LE 输入有效，因此将已写入的字节视为 NAND Flash 的一个命令。该命令由存储器设备锁存后，随后进行页读取操作时，不需要再次写入该命令。
3. 通过在通用存储器空间或特性空间写入 STARTAD[7:0]、STARTAD[16:9]、STARTAD[24:17] 以及 STARTAD[25]（针对 64 Mb x 8 位的 NAND Flash）这 4 个字节（对于容量较小的设备写入 3 个字节），CPU 能够发送读操作的起始地址 (STARTAD)。在写入选通期间（NWE 上为低电平脉冲），NAND Flash 设备的 ALE 输入有效，因此将已写入的字节视为读操作的起始地址。借助特性存储器空间，可使用 FMC 的另一种不同时序配置，实现某些 NAND Flash 所需的预等待功能（有关详细信息，请参见*第 13.6.5 节：NAND Flash 预等待功能*）。
4. 控制器在对同一个或另一个存储区域进行新访问之前，等待 NAND Flash 准备好（R/NB 信号处于高电平）。等待期间，控制器保持 NCE 信号有效（低电平）。
5. 然后 CPU 能够从通用存储器空间执行字节读操作，进而按字节读取 NAND Flash 页（数据字段 + 备用字段）。

6. 可读取下一个 NAND Flash 页，而无需任何 CPU 命令或地址写操作。可采用以下三种不同的方式实现：
   – 执行步骤 5 中介绍的操作
   – 通过重新开始步骤 3 中的操作随机访问一个新地址
   – 通过重新开始步骤 2 向 NAND Flash 设备发送新命令

13.6.5 NAND Flash 预等待功能

一些 NAND Flash 设备需要在写入地址的最后一部分后，控制器等待 R/NB 信号变为低电平。（见*图 53*）。

图 53. 访问非“CE 无关”NAND-Flash



1. CPU 在地址 0x7001 0000 处写入字节 0x00。
2. CPU 在地址 0x7002 0000 处写入字节 A7~A0。
3. CPU 在地址 0x7002 0000 处写入字节 A16~A9。
4. CPU 在地址 0x7002 0000 处写入字节 A24~A17。
5. CPU 在地址 0x7802 0000 处写入字节 A25：FMC 通过 FMC_PATT2 时序定义执行写访问，其中 ATTHOLD ≥ 7（假设 (7+1) × HCLK = 112 ns > t_{WB} 最大值）。这可确保 NCE 保持低电平，直到 R/NB 再次变为低电平后变为高电平（只有 NCE 信号对之有作用的 NAND Flash 才需要此功能）。

当需要此功能时，可通过编程 MEMHOLD 值确保满足 t_{WB} 时序。然而 CPU 对 NAND Flash 进行的所有读或写访问均会经过 (MEMHOLD + 1) 个 HCLK 周期的保持延迟（该延迟插入在 NWE 信号的上升沿与下一访问之间）。

要克服该时序限制，可使用特性存储器空间，将其时序寄存器编程为满足 t_{WB} 时序的 ATTHOLD 值并将 MEMHOLD 保持为其最小值。然后， CPU 必须使用通用存储器空间进行所有的 NAND Flash 读和写访问，向 NAND Flash 设备写入最后一个地址字节时除外，此时 CPU 必须对特性存储器空间执行写操作。

13.6.6 纠错码 (ECC) 计算 (NAND Flash)

FMC NAND 卡控制器包括两个纠错码计算硬件模块，每个存储区域各有一个。这些模块可在软件处理 ECC 时减少主机 CPU 工作负载。

这两个 ECC 模块相同，分别与存储区域 2 和存储区域 3 相关联。因此，硬件 ECC 计算均不适用于与存储区域 4 相连的存储器。

对 NAND Flash 执行读取或写入操作时，相应每 256、512、1 024、2 048、4 096 或 8 192 个字节，FMC 中使用的 ECC 算法可修正 1 位错误并且检测出 2 位错误。该操作基于 Hamming 编码算法，并且包括计算行和列奇偶校验。

每当 NAND Flash 存储区域处于激活状态时，ECC 模块均会监视 NAND Flash 数据总线和读/写信号（NCE 和 NWE）。

ECC 按如下说明操作：

- 当访问 NAND Flash 存储区域 2 或存储区域 3 时，将锁存 D[15:0] 总线上出现的数据并将其用于 ECC 计算。
- 当访问 NAND Flash 中的任何其它地址时，ECC 逻辑会进入空闲状态，不执行任何操作。因此，进行 ECC 计算时，用于定义 NAND Flash 命令或地址的写操作无效。

主机 CPU 对 NAND Flash 完成所需字节数的读取/写入操作后，必须读取 FMC_ECCR 寄存器，才能检索计算出的值。读取后，应通过将 ECCEN 位复位为零来将这些寄存器清零。要计算新的数据块，必须将 FMC_PCR 寄存器中的 ECCEN 位置 1。

要进行 ECC 计算：

1. 使能 FMC_PCR 寄存器中的 ECCEN 位。
2. 将数据写入 NAND Flash 页。在写入 NAND 页期间，ECC 模块将计算 ECC 值。
3. 读取 FMC_ECCR 寄存器中所提供的 ECC 值，并将其存储到变量。
4. 将 FMC_PCR 寄存器中的 ECCEN 位清零后使能，然后从 NAND 页回读写入的数据。在读取 NAND 页期间，ECC 模块将计算 ECC 值。
5. 读取 FMC_ECCR 寄存器中所提供的新 ECC 值。
6. 如果两次读取的 ECC 值相同，则无需校正，否则说明存在 ECC 错误，并且软件校正例程将返回有关该错误是否能够得到校正的信息。

13.6.7 NAND Flash 控制寄存器

NAND Flash 控制寄存器 (FMC_PCR)

NAND Flash control registers

偏移地址：0x80

复位值：0x0000 0018

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAR			TCLR				Res.	Res.	ECCEN	PWID		PTYP	PBKEN	PWAITEN	Res.
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 19:17 **ECCPS[2:0]**：ECC 页大小 (ECC page size)。
定义扩展 ECC 的页大小：
000：256 字节
001：512 字节
010：1024 字节
011：2048 字节
100：4096 字节
101：8192 字节

位 16:13 **TAR[3:0]**：ALE 到 RE 的延迟 (ALE to RE delay)。
以 AHB 时钟周期数 (HCLK) 设置从 ALE 低电平到 RE 低电平的时间。
时间是 t_ar = (TAR + SET + 2) × THCLK，其中 THCLK 是 HCLK 时钟周期
0000：1 个 HCLK 周期（默认）
1111：16 个 HCLK 周期
注： 根据寻址空间，SET 为 MEMSET 或 ATTSET。

位 12:9 **TCLR[3:0]**：CLE 到 RE 的延迟 (CLE to RE delay)。
以 AHB 时钟周期数 (HCLK) 设置从 CLE 低电平到 RE 低电平的时间。
时间是 t_clr = (TCLR + SET + 2) × THCLK，其中 THCLK 是 HCLK 时钟周期
0000：1 个 HCLK 周期（默认）
1111：16 个 HCLK 周期
注： 根据寻址空间，SET 为 MEMSET 或 ATTSET。

位 8:7 保留，必须保持复位值

位 6 **ECCEN**：ECC 计算逻辑使能位 (ECC computation logic enable bit)。
0：禁止和复位 ECC 逻辑（复位后为默认值），
1：使能 ECC 逻辑。

位 5:4 **PWID[1:0]**：数据总线宽度 (Data bus width)。
定义外部存储器设备宽度。
00：8 位
01：16 位（复位后的默认值）。
10：保留。
11：保留。

位 3 **PTYP**：存储器类型 (Memory type)。
定义附加到相应存储区域的设备的类型：
0：保留，必须保持复位值
1：NAND Flash（复位后为默认值）

位 2 **PBKEN**：NAND Flash 存储区域使能位 (NAND Flash memory bank enable bit)。
使能存储区域。访问禁止的存储区域会引起 AHB 总线上的错误
0：禁止相应的存储区域（复位后为默认值）
1：使能相应的存储区域

位 1 **PWAITEN**：等待特性使能位 (Wait feature enable bit)。
使能 NAND Flash 存储区域的等待特性：
0：禁用
1：使能

位 0 保留，必须保持复位值

FIFO 状态和中断寄存器 (FMC_SR)

FIFO status and interrupt register

偏移地址：0x84

复位值：0x0000 0040

该寄存器包含有关 FIFO 状态和中断的信息。FMC 具有一个 FIFO，当向存储器执行写入操作以传输来自 AHB 的多达 16 字的数据时，使用该 FIFO。

当 FMC 将其 FIFO 的内容移入存储器时，此寄存器用于快速写入 FIFO，然后释放 AHB 供 FMC 以外的外设的事务使用。这些寄存器位中有一位用来指示 FIFO 的状态，供 ECC 使用。

在将数据写入存储器时计算 ECC。因此，为了读取正确的 ECC，软件必须等到 FIFO 为空。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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.	FEMPT	IFEN	ILEN	IREN	IFS	ILS	IRS
									r	rw	rw	rw	rw	rw	rw

位 31:7 保留，必须保持复位值

位 6 **FEMPT**：FIFO 为空 (FIFO empty)
该位是提供 FIFO 状态的只读位
0：FIFO 非空
1：FIFO 为空

位 5 **IFEN**：中断下降沿检测使能位 (Interrupt falling edge detection enable bit)
0：中断下降沿检测请求禁止
1：中断下降沿检测请求使能

位 4 **ILEN**：中断高电平检测使能位 (Interrupt high-level detection enable bit)
0：中断高电平检测请求禁止
1：中断高电平检测请求使能

位 3 **IREN**：中断上升沿检测使能位 (Interrupt rising edge detection enable bit)
0：中断上升沿检测请求禁止
1：中断上升沿检测请求使能

位 2 **IFS**：中断下降沿状态 (Interrupt falling edge status)
此标志由硬件置 1，由软件复位。
0：未出现中断下降沿
1：出现中断下降沿

位 1 **ILS**：中断高电平状态 (Interrupt high-level status)
此标志由硬件置 1，由软件复位。
0：未出现中断高电平
1：出现中断高电平

位 0 **IRS**：中断上升沿状态 (Interrupt rising edge status)
此标志由硬件置 1，由软件复位。
0：未出现中断上升沿
1：出现中断上升沿

通用存储器空间时序寄存器 2..4 (FMC_PMEM)

Common memory space timing register 2..4

偏移地址：地址：0x88

复位值：0xFCFC FCFC

FMC_PMEM 读/写寄存器包含 NAND Flash 存储区域的时序信息。该信息用于访问 NAND Flash 的通用存储空间来实现命令、地址写访问和数据读/写访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEMHIZx								MEMHOLDx							
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
MEMWAITx								MEMSETx							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **MEMHIZ[7:0]**：通用存储器 x 数据总线高阻态时间 (Common memory x databus Hi-Z time)
定义在存储区上对 NAND Flash 的通用存储空间开始执行写访问之后，数据总线保持高阻态所持续的 HCLK 时钟周期数。仅对写入事务有效：
0000 0000：1 个 HCLK 周期
1111 1110：255 个 HCLK 周期
1111 1111：保留。

位 23:16 **MEMHOLD[7:0]**：通用存储器保持时间 (Common memory hold time)
针对在存储区 x 上对 NAND Flash 的通用存储空间执行的读或写访问，定义禁止命令（NWE、NOE）之后保持地址（和写访问数据）的 HCLK 时钟周期数：
0000 0000：保留。
0000 0001：1 个 HCLK 周期
1111 1110：254 个 HCLK 周期
1111 1111：保留。

位 15:8 **MEMWAIT[7:0]**：通用存储器等待时间 (Common memory wait time)

针对在存储区上对 NAND Flash 的通用存储空间执行的读或写访问，定义使能命令（NWE、NOE）所需的 HCLK (+1) 时钟周期数最小值。如果等待信号 (NWAIT) 在编程的 HCLK 值末尾处有效（低电平），则命令使能的持续时间将延长。

0000 0000：保留
0000 0001：2 个 HCLK 周期（+ 禁止 NWAIT 时引入的等待周期）
1111 1110：255 个 HCLK 周期（+ 禁止 NWAIT 时引入的等待周期）
1111 1111：保留。

位 7:0 **MEMSET[7:0]**：通用存储器 x 建立时间 (Common memory x setup time)

针对在存储区 x 上对 NAND Flash 的通用存储空间执行的读或写访问，定义使能命令（NWE，NOE）前建立地址所需的 HCLK (+1) 时钟周期数：

0000 0000：1 个 HCLK 周期
1111 1110：255 个 HCLK 周期
1111 1111：保留。

特性存储器空间时序寄存器 2..4 (FMC_PATT)

Attribute memory space timing registers 2..4

偏移地址：0x8C

复位值：0xFCFC FCFC

FMC_PATT 读/写寄存器包含 NAND Flash 存储区域的时序信息。当对 NAND Flash 的特性存储空间进行 8 位访问时，如果最后一次地址写访问的时序必须与先前访问的时序不同，则使用该寄存器（有关就绪/繁忙管理的信息，请参见*第 13.6.5 节：NAND Flash 预等待功能*）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ATTHIZx								ATTHOLDx							
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
ATTWAITx								ATTSETx							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **ATTHIZ[7:0]**：特性存储器数据总线高阻态时间 (Attribute memory data bus Hi-Z time)

定义在存储区上对 NAND Flash 的特性存储空间开始执行写访问之后，数据总线保持高阻态所持续的 HCLK 时钟周期数。仅对写入事务有效：

0000 0000：0 个 HCLK 周期
1111 1110：255 个 HCLK 周期
1111 1111：保留。

位 23:16 **ATTHOLD[7:0]**：特性存储器保持时间 (Attribute memory hold time)

针对在存储区上对 NAND Flash 的特性存储空间执行的读或写访问，定义禁止命令（NWE、NOE）之后保持地址（和写访问数据）的 HCLK 时钟周期数：

0000 0000：保留
0000 0001：1 个 HCLK 周期
1111 1110：254 个 HCLK 周期
1111 1111：保留。

位 15:8 **ATTWAIT[7:0]**：特性存储器等待时间 (Attribute memory wait time)

针对在存储区 x 上对 NAND Flash 的特性存储空间执行的读或写访问，定义使能命令（NWE、NOE）所需的 HCLK (+1) 时钟周期数最小值。如果等待信号 (NWAIT) 在编程的 HCLK 值末尾处有效（低电平），则命令使能的持续时间将延长。

0000 0000：保留

0000 0001：2 个 HCLK 周期（+ 禁止 NWAIT 时引入的等待周期）

1111 1110：255 个 HCLK 周期（+ 禁止 NWAIT 时引入的等待周期）

1111 1111：保留。

位 7:0 **ATTSET[7:0]**：特性存储器建立时间 (Attribute memory setup time)

针对在存储区上对 NAND Flash 的特性存储空间执行的读或写访问，定义使能命令（NWE，NOE）前建立地址所需的 HCLK (+1) 时钟周期数：

0000 0000：1 个 HCLK 周期

1111 1110：255 个 HCLK 周期

1111 1111：保留。

ECC 结果寄存器 (FMC_ECCR)

ECC result registers

偏移地址：0x94

复位值：0x0000 0000

该寄存器包含由 FMC NAND 控制器的 ECC 计算模块计算所得的当前错误校正代码值。当 CPU 从 NAND Flash 页的正确地址读取数据时（请参见*第 13.6.6 节：纠错码 (ECC) 计算 (NAND Flash)*），ECC 计算模块会自动处理读取自/写入 NAND Flash 的数据。读取 X 个字节后（依据 FMC_PCR 寄存器的 ECCPS 字段），CPU 必须从 FMC_ECC 寄存器读取计算所得的 ECC 值。然后验证这些计算的奇偶校验数据是否与备用区记录的奇偶校验数据相同，从而确定该页是否有效，如不同则进行校正。FMC_ECCR 寄存器被读取后应通过将 ECCEN 位置 0 进行清零。如要计算新的数据块，则必须将 ECCEN 位置 1。

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

位 31:0 **ECC**：ECC 结果 (ECC result)

该字段包含由 ECC 计算逻辑计算所得的值。*表 85* 介绍了这些位字段的内容。

表 85. ECC 结果相关位

ECCPS[2:0]	以字节为单位的页大小	ECC 位
000	256	ECC[21:0]
001	512	ECC[23:0]
010	1024	ECC[25:0]
011	2048	ECC[27:0]
100	4096	ECC[29:0]
101	8192	ECC[31:0]

13.7 SDRAM控制器

13.7.1 SDRAM 控制器主要特性

SDRAM 控制器的主要特性如下：

- 两个 SDRAM 存储区域，可独立配置
- 8 位、16 位和 32 位数据总线宽度
- 13 位地址行，11 位地址列，4 个内部存储区域：4x16Mx32bit (256 MB)、4x16Mx16bit (128 MB)、4x16Mx8bit (64 MB)
- 支持字、半字和字节访问
- SDRAM 时钟可以是 HCLK/2 或 HCLK/3
- 自动进行行和存储区域边界管理
- 多存储区域乒乓访问
- 可编程时序参数
- 支持自动刷新操作，可编程刷新速率
- 自刷新模式
- 掉电模式
- 通过软件进行 SDRAM 上电初始化
- CAS 延迟 1,2,3
- 读 FIFO 可缓存，支持 6 行 x 32 位深度（6 x14 位地址标记）

13.7.2 SDRAM 外部存储器接口信号

启动时，必须通过用户应用程序对用于连接 FMC SDRAM 控制器与外部 SDRAM 设备的 SDRAM I/O 引脚进行配置。应用程序未使用的 SDRAM 控制器 I/O 引脚可用于其它用途。

表 86. SDRAM 信号

SDRAM 信号	I/O 类型	说明	复用功能
SDCLK	O	SDRAM 时钟	-
SDCKE[1:0]	O	SDCKE0：SDRAM 存储区域 1 时钟使能 SDCKE1：SDRAM 存储区域 2 时钟使能	-
SDNE[1:0]	O	SDNE0：SDRAM 存储区域 1 芯片使能 SDNE1：SDRAM 存储区域 2 芯片使能	-
A[12:0]	O	地址	FMC_A[12:0]
D[31:0]	I/O	双向数据总线	FMC_D[31:0]
BA[1:0]	O	存储区域地址	FMC_A[15:14]
NRAS	O	行地址选通	-
NCAS	O	列地址选通	-
SDNWE	O	写入使能	-
NBL[3:0]	O	写访问的输出字节屏蔽 （存储器信号名称：DQM[3:0]）	FMC_NBL[3:0]

13.7.3 SDRAM 控制器功能说明

所有 SDRAM 控制器输出（信号、地址和数据）在存储器时钟 (FMC_SDCLK) 的下降沿上变化。

SDRAM 初始化

初始化序列通过软件进行管理。如果使用了两个存储区域，则必须将 FMC_SDCMR 寄存器中的目标存储区域位 CTB1 和 CTB2 置 1，同时为存储区域 1 和存储区域 2 生成初始化序列：

1. 将存储器设备的特性编程到 FMC_SDCRx 寄存器中。SDRAM 时钟频率、RBURST 和 RPIPE 特性必须编程到 FMC_SDCR1 寄存器中。
2. 将存储器设备的时序编程到 FMC_SDTRx 寄存器中。TRP 和 TRC 时序必须编程到 FMC_SDTR1 寄存器中。
3. 将 MODE 位置为“001”并配置 FMC_SDCMR 寄存器中的目标存储区域位（CTB1 和/或 CTB2）以开始为存储器提供时钟信号（SDCKE 驱动为高电平）。
4. 等待指定延迟周期。典型延迟为 100 µs（有关上电后所需延迟的信息，请参见 SDRAM 数据手册）。
5. 将 MODE 位置为“010”并配置 FMC_SDCMR 寄存器中的目标存储区域位（CTB1 和/或 CTB2）以发送“全部预充电”命令。
6. 将 MODE 位置为“011”并配置 FMC_SDCMR 寄存器中的目标存储区位（CTB1 和/或 CTB2）和连续自动刷新命令 (NRFS) 的数量。请参见 SDRAM 数据手册了解应发出的自动刷新命令个数。通常为 8 个。
7. 根据 SDRAM 设备配置 MRD 字段，将 MODE 位置为“100”并配置 FMC_SDCMR 寄存器中的目标存储区域位（CTB1 和/或 CTB2）以发送“加载模式寄存器”命令并对 SDRAM 设备进行编程。
   尤其是：
   a) 必须根据 FMC_SDCR12 中的值来选择 CAS 延迟
   b) 必须通过将模式寄存器中的 M[2:0] 位设置为 000 来将突发长度 (BL) 设置为 1。请参见 SDRAM 设备数据手册。

   如果两个 SDRAM 存储区域的模式寄存器不同，则此步骤必须重复两次，每个存储区域各一次且目标存储区域位相应置 1。
8. 编程 FMC_SDRTR 寄存器中的刷新速率。
   刷新速率对应于刷新周期之间的延迟。其值必须与 SDRAM 设备相适应。
9. 对于移动 SDRAM 器件，要编程扩展模式寄存器，应在 SDRAM 器件初始化后执行：首先，应在 BA1=1 且 BA=0 时执行一次空的读访问（有关 BA[1:0] 地址映射的信息，请参见 SDRAM 地址映射部分），以便选择扩展模式寄存器而非加载模式寄存器，然后编程所需的值。

在这一阶段，SDRAM 设备已做好接受命令的准备。如果进行 SDRAM 访问期间发生系统复位，则数据总线仍可能由 SDRAM 设备驱动。因此，必须在复位后重新初始化 SDRAM 设备，NOR Flash/PSRAM/SRAM 或 NAND Flash 控制器才能发送新的访问命令。

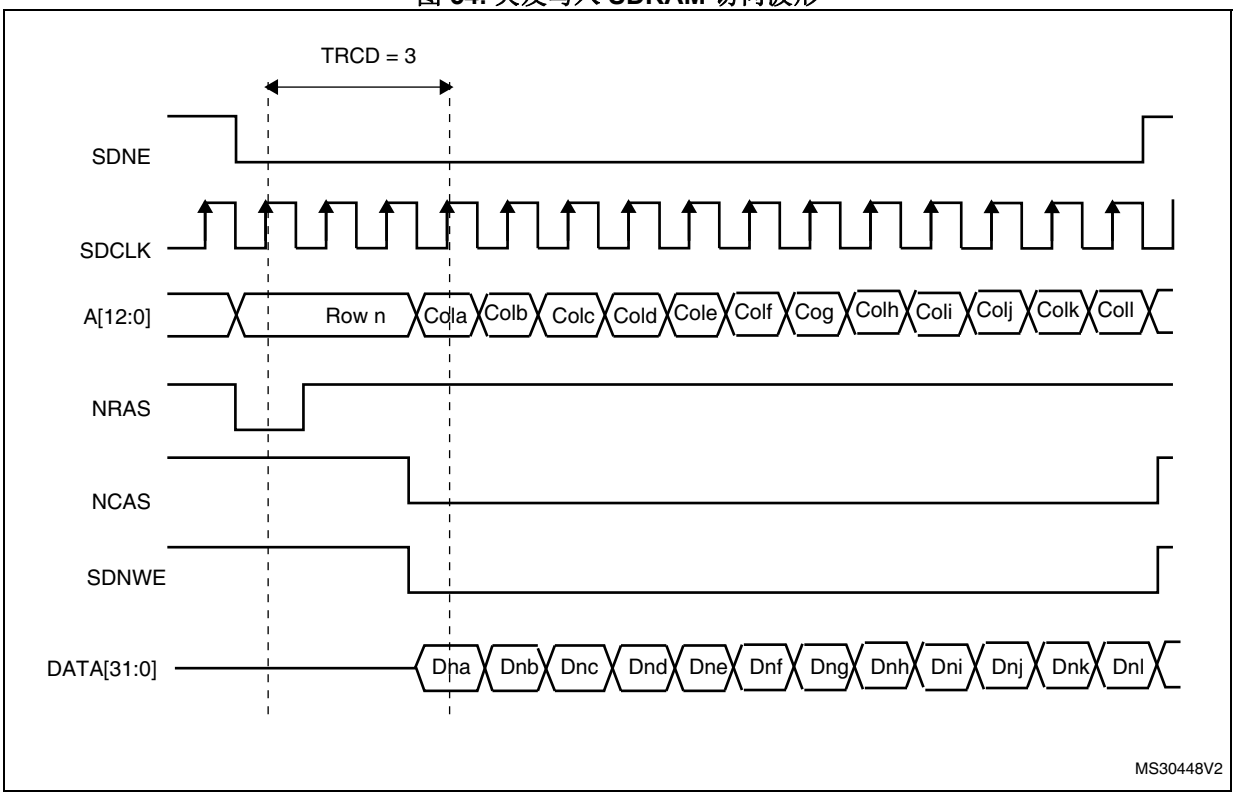
注： *如果有两个 SDRAM 设备连接到 FMC，对命令模式寄存器同时访问这两个器件（加载模式寄存器命令）的情况，将按 FMC_SDTR1 寄存器中为 SDRAM 存储区域 1 配置的时序参数（TMRD 和 TRAS 时序）发出访问命令。*

SDRAM 控制器写周期

SDRAM 控制器可接收单次的和突发的写请求，并将其视为单次存储器访问。在这两种情况下，SDRAM 控制器都会跟踪各存储区域的有效行，以能够对不同的存储区域进行连续的写访问（多存储区域乒乓访问）。

执行任何写访问前，必须将 FMC_SDCRx 寄存器中的 WP 位清零，禁止 SDRAM 存储区域的写保护。

图 54. 突发写入 SDRAM 访问波形



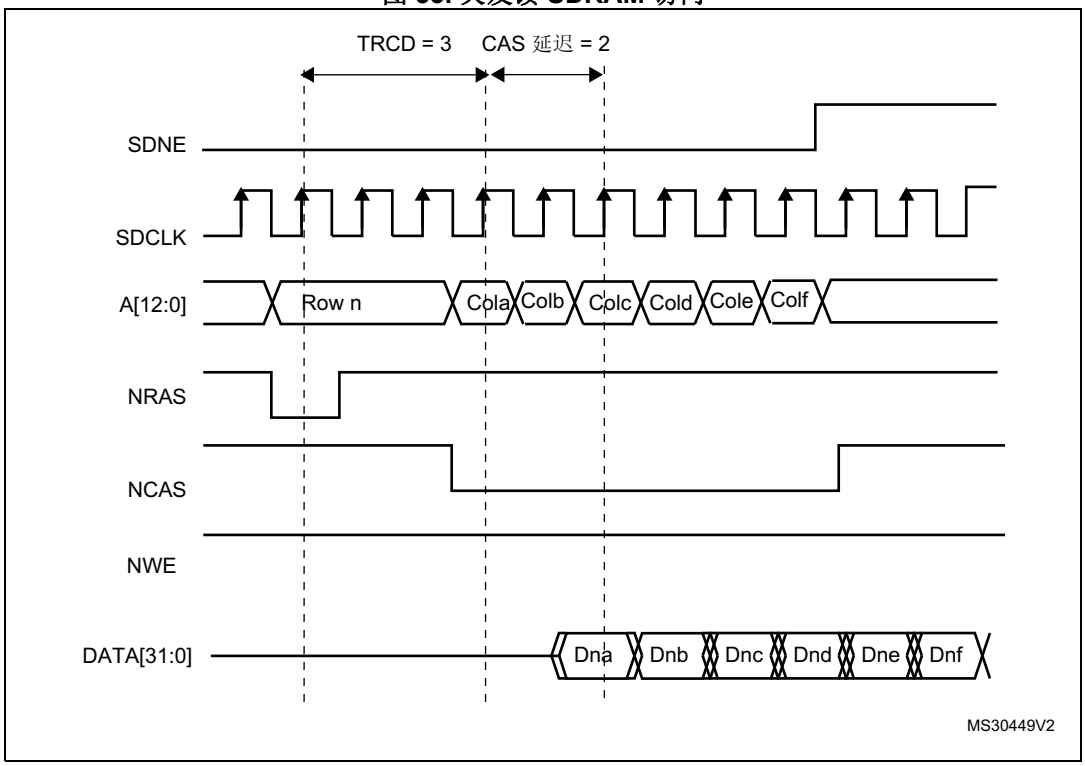
SDRAM 控制器始终会检查下一个访问。

- 如果下一个访问发生在同一行或在其它有效行，则直接执行写操作。
- 如果下一个访问指向一个无效行，则 SDRAM 控制器将生成预充电命令、激活该新行并初始化写命令。

SDRAM 控制器读周期

SDRAM 控制器可接收单次的和突发的读请求，并将其视为单次存储器访问。在这两种情况下，SDRAM 控制器都会跟踪各存储区域的有效行，以能够对不同的存储区域进行连续的读访问（多存储区域乒乓访问）。

图 55. 突发读 SDRAM 访问



FMC SDRAM 控制器具有可缓存的读 FIFO（6 行 x 32 位），它用于根据以下公式在 CAS 延迟周期和 RPIPE 延迟期间提前存储读取的数据。必须将 FMC_SDCR1 寄存器中的 RBURST 位置 1 才能接受下一个读访问。

预期的数据数量 = CAS 延迟 + 1 + (RPIPE 延迟)/2

示例：

- CAS 延迟 = 3，RPIPE 延迟 = 0：四个数据（未提交）存储在 FIFO 中。
- CAS 延迟 = 3, RPIPE 延迟 = 0：五个数据（未提交）存储在 FIFO 中。

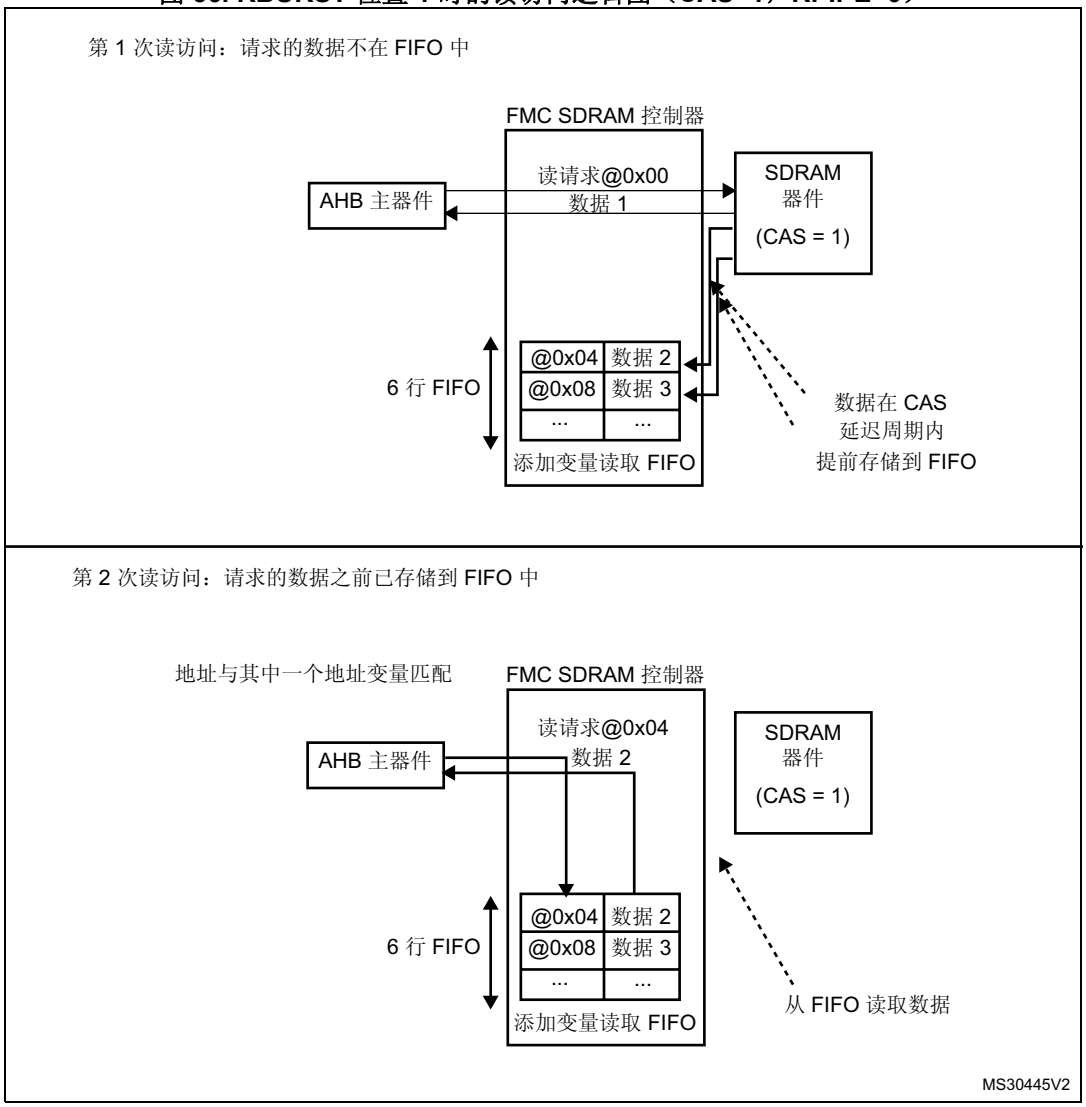
读 FIFO 的每行都具有 14 位地址标记用于标识自身内容：11 位用于表示列地址，2 位用于表示选择内部存储区域和有效行，1 位用于选择 SDRAM 设备

在 AHB 突发读取期间，如果提前到达行末尾，则提前读取的数据（未提交）不存储到读 FIFO。对于单次读访问，数据将正确存储到读 FIFO。

每当出现读请求时，SDRAM 控制器将检查：

- 地址与地址标记之一是否匹配，如果找到匹配，则直接从 FIFO 读取数据并清空相应地址标记/行内容，同时压缩 FIFO 中的剩余数据以避免空行。
- 否则，向存储器发送新的读命令并用新数据更新 FIFO。如果 FIFO 已满，则较早的数据将丢失。

图 56. RBURST 位置 1 时的读访问逻辑图（CAS=1，RPIPE=0）



读访问或预充电命令期间，读 FIFO 将刷新并做好填充新数据的准备。

接到第一个读请求后，如果当前访问还未进行到行边界，则 SDRAM 控制器将在 CAS 延迟周期和 RPIPE 延迟（如果已配置）期间接受下一个读访问。这将通过递增存储器地址来实现。必须满足以下条件：

- FMC_SDCR1 寄存器中的 RBURST 控制位必须置“1”。

地址管理取决于下一个 AHB 请求：

- 下一个 AHB 请求是连续的（AHB 突发）
  这种情况下，SDRAM 控制器将递增地址。

- 下一个 AHB 请求不连续
  - 如果新的读请求指向与上一请求相同的行或另一个有效行，则新地址将传送给存储器，同时主设备在 CAS 延迟周期停止工作，等待从存储器获取新数据。
  - 如果新的读请求指向无效行，则 SDRAM 控制器生成预充电命令、激活该新行并初始化读命令。

如果 RURST 位置 0，则不使用读 FIFO。

行和存储区域边界管理

当读/写访问跨越了行边界时，如果下一个读/写访问是连续的并且当前访问已执行到行边界，则 SDRAM 控制器将执行以下操作：

1. 对有效行进行预充电，
2. 激活新行。
3. 启动读/写命令。

对于各种列和数据总线宽度配置，都支持在行边界自动激活下一行。

SDRAM 控制器可根据需要在以下命令之间插入附加时钟周期：

- 在预充电和激活命令之间插入以匹配 TRP 参数（仅当下一个访问指向同一存储区域中的其它行时），
- 在激活和读命令之间插入以匹配 TRCD 参数。

这些参数在 FMC_SDTRx 寄存器中定义。

有关跨越行边界读取和突发写访问的信息，请参见*图 57* 和*图 58*。

图 57. 跨行边界的读访问

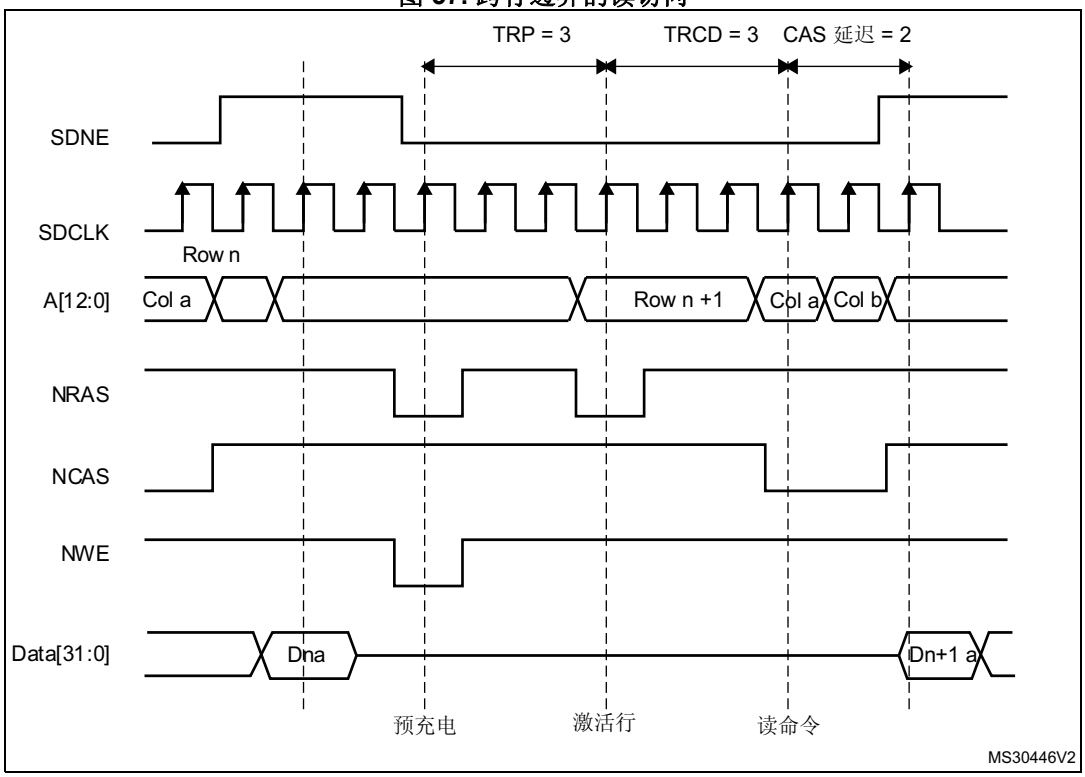
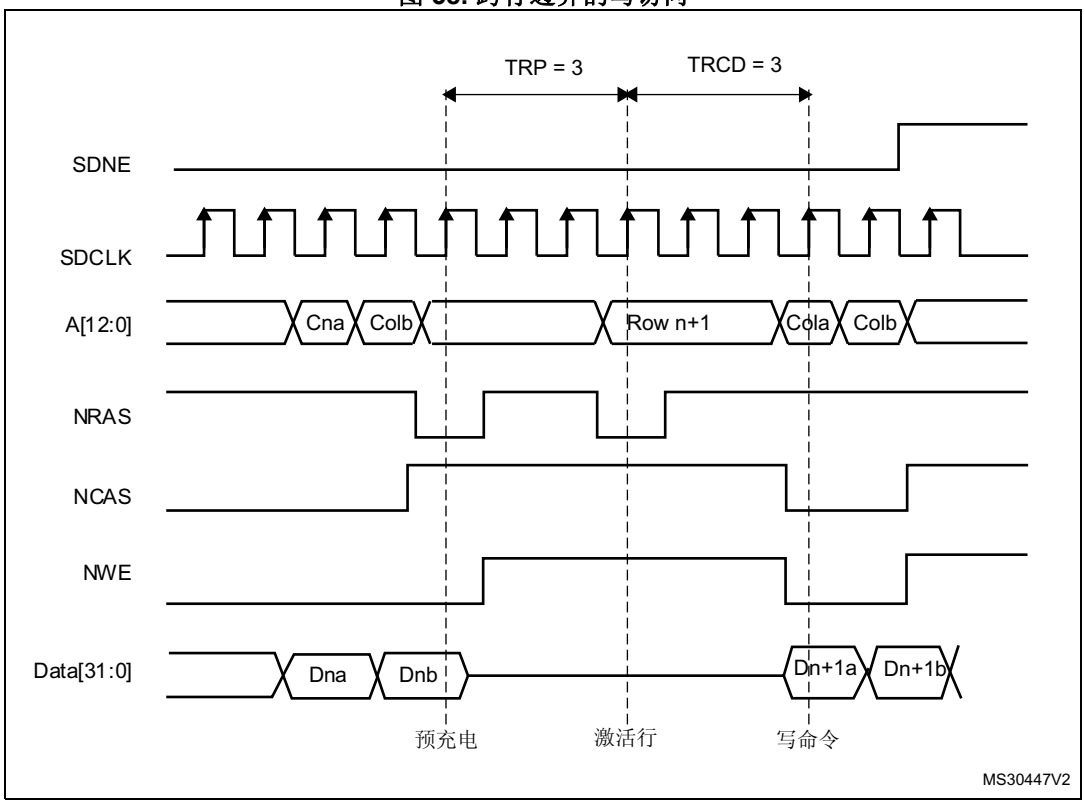


图 58. 跨行边界的写访问



如果下一个访问是连续的并且当前访问跨越了存储区域边界，则 SDRAM 控制器将激活下一个存储区域的第一个行并发出新的读/写命令。可能存在以下两种情况：

- 如果当前存储区域不是最后一个存储区域，则必须对新存储区域中的激活行进行预充电。对于各种行/列和数据总线宽度配置，都支持在存储区域边界自动激活下一行。
- 如果当前存储区域是最后一个存储区域，则仅在对 13 位行、11 位列、4 个内部存储区域和 32 位数据总线的 SDRAM 设备寻址时，支持自动激活下一行。否则，将违反 SDRAM 地址范围并生成 AHB 错误。
- 对于 13 位行地址、11 位列地址、4 个内部存储区域和总线宽度为 32 位的 SDRAM 存储器，SDRAM 控制器将通过第二个 SDRAM 设备继续进行读/写操作（假设第二个 SDRAM 设备已初始化）：
  a) SDRAM 控制器将激活第一行（在对有效行预充电之后，假设第一个内部存储区域中存在有效行）并发出新的读/写命令。
  b) 如果第一行已激活，则 SDRAM 控制器将仅发出读/写命令。

SDRAM 控制器刷新周期

自动刷新命令用于刷新 SDRAM 设备的内容。SDRAM 控制器会定期发送自动刷新命令。它使用一个内部计数器装载 FMC_SDRTR 寄存器中的 COUNT 值。该值定义刷新周期之间的存储器时钟周期个数（刷新速率）。该计数器的值达到零时将生成一个内部脉冲。

如果存在进行中的存储器访问，则会延迟自动刷新请求。不过，在存储器访问和自动刷新请求同时出现时，则优先处理自动刷新请求。

如果在自动刷新期间访问存储器，则会缓存访问请求并在自动刷新完成后进行处理。

如果在上一个自动刷新请求尚未完成的情况下又出现了新的自动刷新请求，则状态寄存器中的 RE（刷新错误）位将置 1。该位如果已使能（REIE = “1”），将生成中断。

如果 SDRAM 的行不是空闲状态（并非所有行都已关闭），则 SDRAM 控制器将生成 PALL（全部预充电）命令，然后再进行自动刷新。

如果由 FMC_SDCMR 命令模式寄存器（模式位 = “011”）生成自动刷新命令，则必须先发出 PALL 命令（模式位 = “010”）。

13.7.4 低功耗模式

可使用两种低功耗模式：

- 自刷新模式
  由 SDRAM 设备自身执行自动刷新循环以保留数据，无需外部时钟。
- 掉电模式
  由 SDRAM 控制器执行自动刷新循环。

自刷新模式

通过将 MODE 位置为 “101” 并配置 FMC_SDCMR 寄存器中的目标存储区域位（CTB1 和/或 CTB2）来选择该模式。

SDRAM 时钟在 TRAS 延迟后停止运行，而内部刷新定时器只有在满足以下条件之一时才停止计数：

- 向两个设备都发出了自刷新命令
- 其中一个设备未激活（SDRAM 存储区域未初始化）。

进入自刷新模式前，SDRAM 控制器会自动发送 PALL 命令。

如果写数据 FIFO 非空，则所有数据都将在自刷新模式激活前发送到存储器并且 BUSY 状态标志保持置 1。

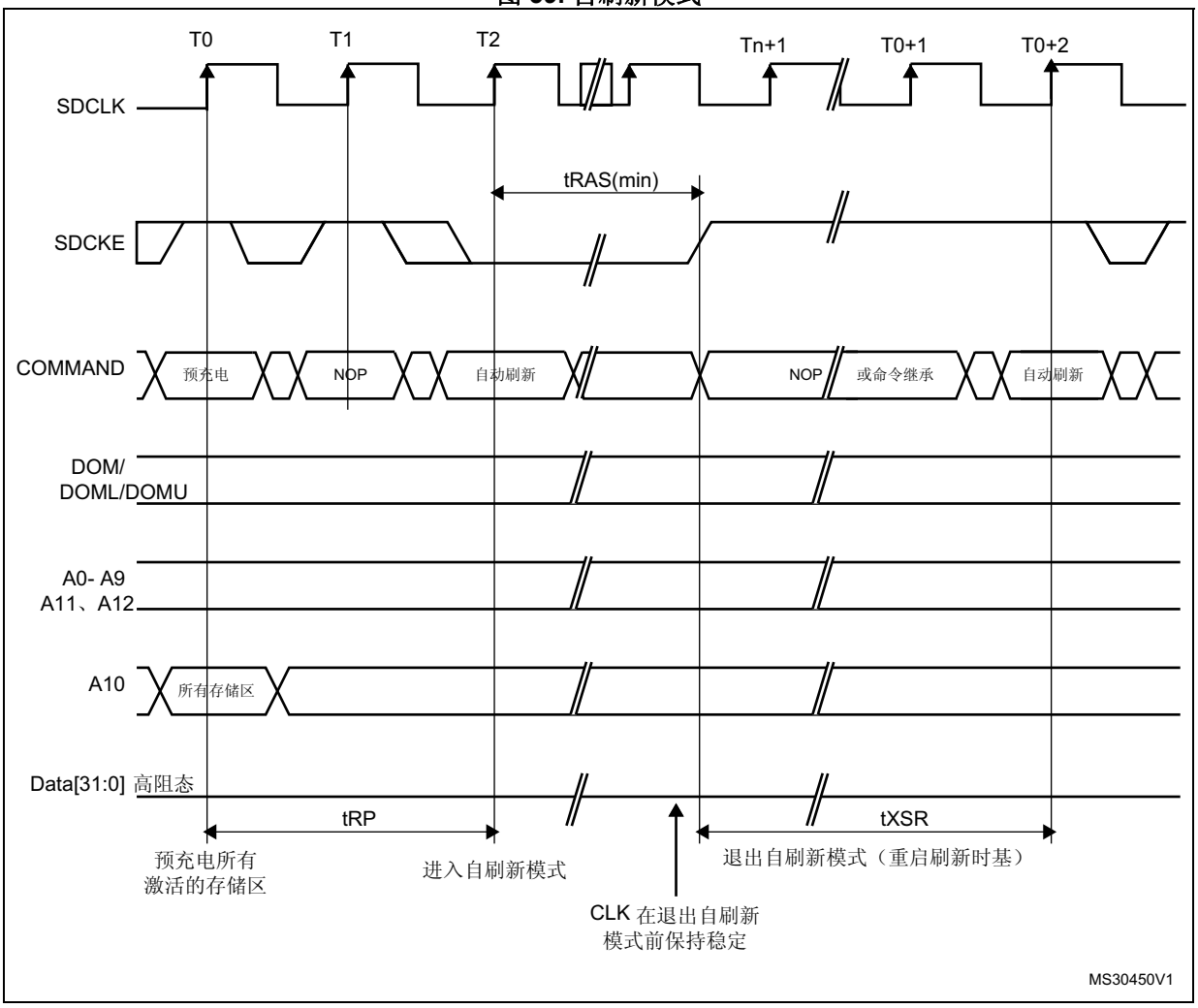
在自刷新模式下，除保持低电平的 SDCKE 外，SDRAM 设备的所有输入都无效。

SDRAM 设备必须处于自刷新模式最短为 TRAS 时间，且能够在更长的时间内始终处于自刷新模式。为保证这一最短时长，在自刷新激活后的 TRAS 延迟期间，BUSY 状态标志将保持高电平。

SDRAM 控制器会在有 SDRAM 设备被选定后立即生成一个命令序列以退出自刷新模式。存储器访问完成后，选定的设备将保持正常模式。

要退出自刷新模式，必须将 MODE 位置为 “000”（正常模式）并配置 FMC_SDCMR 寄存器中的目标存储区域位（CTB1 和/或 CTB2）。

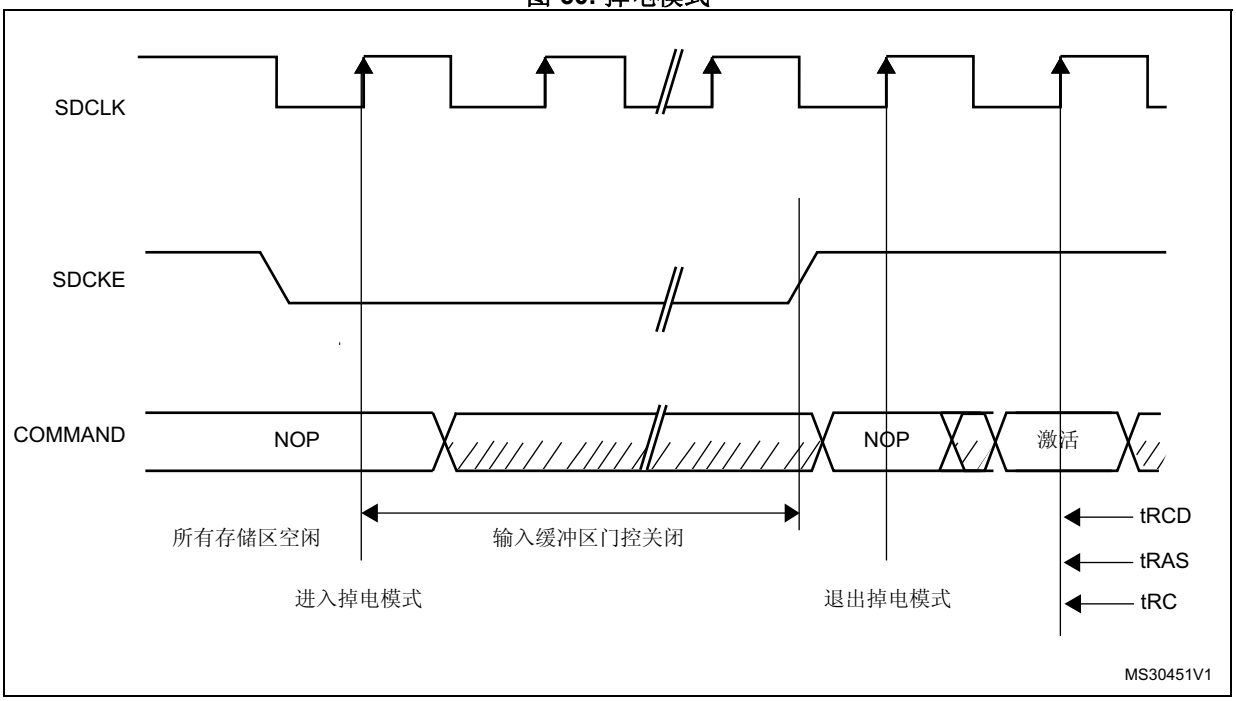
图 59. 自刷新模式


T0
T1
T2
Tn+1
T0+1
T0+2
SDCLK
tRAS(min)
SDCKE
COMMAND
预充电
NOP
自动刷新
NOP
或命令继承
自动刷新
DOM/
DOML/DOMU
A0- A9
A11、A12
A10
所有存储区
Data[31:0] 高阻态
tRP
tXSR
预充电所有
激活的存储区
进入自刷新模式
退出自刷新模式（重启刷新时基）
CLK 在退出自刷新
模式前保持稳定
MS30450V1

掉电模式

通过将 MODE 位置为“110”并配置 FMC_SDCMR 寄存器中的目标存储区域位（CTB1 和/或 CTB2）来选择该模式。

图 60. 掉电模式



如果写数据 FIFO 非空，则所有数据都将在掉电模式激活前发送到存储器。

SDRAM 控制器会在有 SDRAM 设备被选定后立即退出掉电模式。存储器访问完成后，选定的 SDRAM 设备将保持正常模式。

在掉电模式期间，将禁用 SDRAM 设备的所有输入/输出缓冲区，只有保持低电平的 SDCKE 除外。

SDRAM 设备保持掉电模式的时间不会长于刷新周期，并且自身无法执行自刷新循环。因此，SDRAM 控制器通过以下操作执行刷新：

1. 退出掉电模式并将 SDCKE 驱动为高电平
2. 生成 PALL 命令（前提是在掉电模式下存在激活行）
3. 生成自动刷新命令
4. 再次将 SDCKE 驱动为低电平以返回掉电模式。

要退出掉电模式，必须将 MODE 位置为“000”（正常模式）并配置 FMC_SDCMR 寄存器中的目标存储区域位（CTB1 和/或 CTB2）。

13.7.5 SDRAM 控制寄存器

SDRAM 控制寄存器 1，2（FMC_SDCR1，FMC_SDCR2）

SDRAM Control registers 1,2

偏移地址：0x140+ 4* (x – 1)，x = 1,2

复位值：0x0000 02D0

此寄存器包含每个 SDRAM 存储区域的控制参数

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	RPIPE		RBURST	SDCLK		WP	CAS		NB	MWID		NR		NC	
	rw	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 14:13 **RPIPE[1:0]**：读管道 (Read pipe)

这些位可定义在 CAS 延迟后延后多少个 HCLK 时钟周期读取数据。

00：0 个 HCLK 时钟周期延迟
01：1 个 HCLK 时钟周期延迟
10：2 个 HCLK 时钟周期延迟
11：保留。

注： FMC_SDCR2 寄存器中的相应位为无关位。

位 12 **RBURST**：突发读 (Burst read)

此位可使能突发读模式。SDRAM 控制器预期在 CAS 延迟期间接受下一个读命令并将数据存储在读 FIFO 中。

0：不将单次读请求作为突发请求管理
1：始终将单次读请求作为突发请求管理

注： FMC_SDCR2 寄存器中的相应位为无关位。

位 11:10 **SDCLK[1:0]**：SDRAM 时钟配置 (SDRAM clock configuration)

这些位用于定义两个 SDRAM 存储区域的 SDRAM 时钟周期以及在更改频率前禁止时钟。此时，必须重新初始化 SDRAM。

00：禁止 SDCLK 时钟
01：保留
10：SDCLK 周期 = 2 x HCLK 周期
11：SDCLK 周期 = 3 x HCLK 周期

注： FMC_SDCR2 寄存器中的相应位为无关位。

位 9 **WP**：写保护 (Write protection)

该位可使能对 SDRAM 存储区域的写模式访问。

0：允许写访问
1：忽略写访问

位 8:7 **CAS[1:0]**：CAS 延迟 (CAS Latency)

该位可设置 SDRAM CAS 延迟，按存储器时钟周期计

00：保留。

01：1 个周期

10：2 个周期

11：3 个周期

位 6 **NB**：内部存储区域数量 (Number of internal banks)

该位可设置内部存储区域数量。

0：2 个内部存储区域

1：4 个内部存储区域

位 5:4 **MWID[1:0]**：存储器数据总线宽度 (Memory data bus width)

这些位定义存储器件宽度。

00：8 位

01：16 位

10：32 位

11：保留。

位 3:2 **NR[1:0]**：行地址位数 (Number of row address bits)

这些位定义行地址的位数。

00：11 位

01：12 位

10：13 位

11：保留。

位 1:0 **NC[1:0]**：列地址位数 (Number of column address bits)

这些位定义列地址的位数。

00：8 位

01：9 位

10：10 位

11：11 位

注： *修改 RBURST 或 RPIPE 设置或者禁止 SDCLK 时钟之前，用户必须先发送 PALL 命令以确保先完成正在进行的操作。*

SDRAM 时序寄存器 1，2（FMC_SDTR1，FMC_SDTR2）

SDRAM Timing registers 1,2

偏移地址：0x148 + 4 * (x – 1)，x = 1，2

复位值：0x0FFF FFFF

此寄存器包含每个 SDRAM 存储区域的时序参数

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TRCD				TRP				TWR			
				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
TRC				TRAS				TXSR				TMRD			
rw	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 27:24 **TRCD[3:0]**：行到列延迟 (Row to column delay)

这些位定义激活命令与读/写命令之间的延迟，按存储器时钟周期数计。

0000：1 个周期。
0001：2 个周期
....
1111：16 个周期

位 23:20 **TRP[3:0]**：行预充电延迟 (Row precharge delay)

这些位定义预充电命令与其它命令之间的延迟，按存储器时钟周期数计。仅在 FMC_SDTR1 寄存器中配置 TRP 时序。如果使用了两个 SDRAM 设备，则必须使用最慢设备的时序配置 TRP。

0000：1 个周期
0001：2 个周期
....
1111：16 个周期

注： FMC_SDTR2 寄存器中的相应位为无关位。

位 19:16 **TWR[3:0]**：恢复延迟 (Recovery delay)

这些位定义写命令和预充电命令之间的延迟，按存储器时钟周期数计。

0000：1 个周期
0001：2 个周期
....
1111：16 个周期

注： TWR 必须设置成与 SDRAM 数据手册中定义的写恢复时间 (t_{WR}) 相匹配，保证：
TWR ≥ TRAS - TRCD，并且 TWR ≥ TRC - TRCD - TRP
例如：TRAS= 4 个周期，TRCD= 2 个周期。因此，TWR >= 2 个周期。TWR 必须设置为 0x1。
如果使用了两个 SDRAM 设备，则必须为 FMC_SDTR1 和 FMC_SDTR2 配置相同的 TWR 时序（对应于较慢的 SDRAM 设备）。

位 15:12 **TRC[3:0]**：行循环延迟 (Row cycle delay)

这些位定义刷新命令和激活命令之间的延迟，以及两个相邻刷新命令之间的延迟， 以存储器时钟周期数表示。仅在 FMC_SDTR1 寄存器中配置 TRC 时序。如果使用了两个 SDRAM 设备，则必须使用最慢设备的时序配置 TRC。

0000：1 个周期
0001：2 个周期
....
1111：16 个周期

注： TRC 必须与 SDRAM 设备数据手册中定义的 TRC 和 TRFC（自动刷新周期）时序相匹配。

注： FMC_SDTR2 寄存器中的相应位为无关位。

位 11:8 **TRAS[3:0]**：自刷新时间 (Self refresh time)

这些位定义最短的自刷新周期，按存储器时钟周期数计。

0000：1 个周期
0001：2 个周期
....
1111：16 个周期

位 7:4 **TXSR[3:0]**：退出自刷新延迟 (Exit Self-refresh delay)

这些位定义从发出自刷新命令到发出激活命令之间的延迟，按存储器时钟周期数计。

0000：1 个周期
0001：2 个周期
....
1111：16 个周期

注： 如果使用了两个 SDRAM 设备，则必须为 FMC_SDTR1 和 FMC_SDTR2 配置相同的 TXSR 时序（对应于较慢的 SDRAM 设备）。

位 3:0 **TMRD[3:0]**：加载模式寄存器到激活 (Load Mode Register to Active)

这些位定义加载模式寄存器命令和激活或刷新命令之间的延迟，按存储器时钟周期计。

0000：1 个周期
0001：2 个周期
....
1111：16 个周期

注： 如果连接了两个 SDRAM 设备，对于命令模式寄存器同时访问这两个设备（加载模式寄存器命令）的情况，将按 FMC_SDTR1 寄存器中为存储区域 1 配置的时序参数（TMRD 和 TRAS 时序）发出访问命令。

仅在 FMC_SDTR1 寄存器中配置 TRP 和 TRC 时序。如果使用了两个 SDRAM 设备，则必须使用最慢设备的时序配置 TRP 和 TRC 时序。

SDRAM 命令模式寄存器 (FMC_SDCMR)

SDRAM Command Mode register

地址偏移：0x150

复位值：0x0000 0000

该寄存器包含访问 SDRAM 设备时所发出的命令。该寄存器用于初始化 SDRAM 设备、激活自刷新模式和掉电模式。写入 MODE 字段后，将根据 CTB1 和 CTB2 命令位向单个或全部两个 SDRAM 存储区域发送命令。该寄存器为两个 SDRAM 存储区域所共用。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MRD							NRFS				CTB1	CTB2	MODE		
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 21:9 **MRD[12:0]**：模式寄存器定义 (Mode Register definition)

此 13 位字段定义 SDRAM 模式寄存器内容。可通过加载模式寄存器命令对模式寄存器进行编程。

位 8:5 **NRFS[3:0]**：自刷新次数 (Number of Auto-refresh)

这些位定义 MODE =“011”时所发出的连续自刷新命令个数。

0000：1 个自刷新周期
0001：2 个自刷新周期
....
1110：15 个自刷新周期
1111：16 个自刷新周期

位 4 **CTB1**：命令目标存储区域 1 (Command Target Bank 1)

该位指示是否向 SDRAM 存储器 1 发送命令。

0：命令未发送到 SDRAM 存储区域 1
1：命令已发送到 SDRAM 存储区域 1

位 3 **CTB2**：命令目标存储区域 2 (Command Target Bank 2)

该位指示是否向 SDRAM 存储器 2 发送命令。

0：命令未发送到 SDRAM 存储区域 2
1：命令已发送到 SDRAM 存储区域 2

位 2:0 **MODE[2:0]**：命令模式 (Command mode)

这些位定义发送到 SDRAM 设备的命令。

000：正常模式
001：时钟配置使能
010：PALL（“预充电所有存储区域”）命令
011：自刷新命令
100：加载模式寄存器
101：自刷新命令
110：掉电命令
111：保留

注：　命令发出后，至少一个命令目标存储区域位（CTB1 或 CTB2）必须置 1，否则该命令将被忽略。

注：　如果使用两个 SDRAM 存储区域，则必须通过将 CTB1 和 CTB2 位置 1 来向两个器件同时发送自刷新和 PALL 命令，否则该命令将被忽略。

注：　如果只使用一个 SDRAM 存储区域，并通过将其关联的 CTB 位置 1 来发出命令，则未使用存储区域的另一个 CTB 位必须保持为 0。

SDRAM 刷新定时器寄存器 (FMC_SDRTR)

SDRAM Refresh Timer register

偏移地址：0x154

复位值：0x0000 0000

该寄存器通过配置刷新定时器计数值来设置刷新循环之间的刷新速率，按 SDCLK 时钟周期数计。

$$\text{刷新率} = (\text{COUNT} + 1) \times \text{SDRAM 时钟频率}$$

$$\text{COUNT} = ((\text{SDRAM 刷新周期})/\text{行数}) - 20$$

示例

$$\text{刷新率} = 64\ \text{ms}/(8196\ \text{行}) = 7.81\mu\text{s}$$

其中 64 ms 是 SDRAM 的刷新周期。

$$7.81\mu\text{s} \times 60\text{MHz} = 468.6$$

如果在接受读请求后出现内部刷新请求，则必须将刷新速率增加 20 个 SDRAM 时钟周期（如上所示）以获得重充足的裕量。其对应的 COUNT 值为“0000111000000”(448)。

该 13 位字段将加载到使用 SDRAM 时钟递减的定时器。此定时器在计数到零时生成刷新脉冲。COUNT 值必须设置为至少 41 个 SDRAM 时钟周期。

一旦完成对 FMC_SDRTR 寄存器的编程，定时器就开始计数。如果寄存器中编程的值为“0”，则不会执行刷新。切不可在初始化后重新编程该寄存器以避免刷新速率被修改。

每当生成刷新脉冲时，都会重新将该 13 位 COUNT 字段加载到计数器中。

如果存在进行中的存储器访问，则会延迟自动刷新请求。不过，在存储器访问和自动刷新请求同时出现时，则优先处理自动刷新请求。如果在刷新期间访问存储器，则会缓存访问请求并在刷新完成后进行处理。

此寄存器为 SDRAM 存储区域 1 和存储区域 2 所共用。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	REIE	COUNT													CRE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	w

位 31:15 保留，必须保持复位值

位 14 **REIE**：RES 中断使能 (RES Interrupt Enable)

0：禁止中断

1：RE = 1 时生成中断

位 13:1 **COUNT[12:0]**：刷新定时器计数 (Refresh Timer Count)

该 13 位字段定义 SDRAM 设备的刷新速率， 以存储器时钟周期数表示。该字段必须设置为至少 41 个 SDRAM 时钟周期。

刷新速率 = (COUNT + 1) x SDRAM 频率时钟

COUNT =（SDRAM 刷新周期/行数) - 20

位 0 **CRE**：清除刷新错误标志 (Clear Refresh error flag)

该位用于清除状态寄存器中的刷新错误标志 (RE)。

0：无操作

1：清除刷新错误标志

注： *所编程的 COUNT 值不可等于以下时序之和：TWR+TRP+TRC+TRCD+4 个存储器时钟周期。*

SDRAM 状态寄存器 (FMC_SDSR)

SDRAM Status register

偏移地址：0x158

复位值：0x0000 0000

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

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.	BUSY	MODES2		MODES1		RE
										r	r	r	r	r	r

位 31:5 保留，必须保持复位值

位 5 **BUSY**：繁忙状态 (Busy status)

该位定义接到命令模式请求后 SDRAM 控制器的状态

0：SDRAM 控制器已准备好接受新请求

1：SDRAM 控制器未准备好接受新请求

位 4:3 **MODES2[1:0]**：存储区域 2 的状态模式 (Status Mode for Bank 2)

该位定义 SDRAM 存储区域 2 的状态模式。

00：正常模式

01：自刷新模式

10：掉电模式

位 2:1 **MODES1[1:0]**：存储区域 1 的状态模式 (Status Mode for Bank 1)

该位定义 SDRAM 存储区域 1 的状态模式。

00：正常模式

01：自刷新模式

10：掉电模式

位 0 **RE**：清除错误标志 (Refresh error flag)

0：未检测到刷新错误

1：检测到刷新错误

在 REIE = 1 且 RE = 1 时会生成中断

13.8 FMC 寄存器映射

下表对 FMC 寄存器进行了汇总。

表 87. FMC 寄存器映射

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	**FMC_BCR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WFDIS	CCLKEN	CBURSTRW	CPSIZE [2:0]			ASYNCWAIT	EXTMOD	WAITEN	WREN	WAITCFG	Res.	WAITPOL	BURSTEN	Res.	FACCEN	MWID [1:0]		MTYP [1:0]		MUXEN	MBKEN
	Reset value											0	0	0	0	0	0	0	0	1	1	0		0	0		1	0	1	1	0	1	1
0x08	**FMC_BCR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CBURSTRW	CPSIZE [2:0]			ASYNCWAIT	EXTMOD	WAITEN	WREN	WAITCFG	Res.	WAITPOL	BURSTEN	Res.	FACCEN	MWID [1:0]		MTYP [1:0]		MUXEN	MBKEN
	Reset value													0	0	0	0	0	0	1	1	0		0	0		1	0	1	0	0	1	0
0x10	**FMC_BCR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CBURSTRW	CPSIZE [2:0]			ASYNCWAIT	EXTMOD	WAITEN	WREN	WAITCFG	Res.	WAITPOL	BURSTEN	Res.	FACCEN	MWID [1:0]		MTYP [1:0]		MUXEN	MBKEN
	Reset value													0	0	0	0	0	0	1	1	0		0	0		1	0	1	0	0	1	0
0x18	**FMC_BCR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CBURSTRW	CPSIZE [2:0]			ASYNCWAIT	EXTMOD	WAITEN	WREN	WAITCFG	Res.	WAITPOL	BURSTEN	Res.	FACCEN	MWID [1:0]		MTYP [1:0]		MUXEN	MBKEN
	Reset value													0	0	0	0	0	0	1	1	0		0	0		1	0	1	0	0	1	0
0x04	**FMC_BTR1**	Res.	Res.	ACCMOD[1:0]		DATLAT[3:0]				CLKDIV[3:0]				BUSTURN[3:0]				DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
	Reset value			0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x0C	**FMC_BTR2**	Res.	Res.	ACCMOD[1:0]		DATLAT[3:0]				CLKDIV[3:0]				BUSTURN[3:0]				DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
	Reset value			0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x14	**FMC_BTR3**	Res.	Res.	ACCMOD[1:0]		DATLAT[3:0]				CLKDIV[3:0]				BUSTURN[3:0]				DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
	Reset value			0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x1C	**FMC_BTR4**	Res.	Res.	ACCMOD[1:0]		DATLAT[3:0]				CLKDIV[3:0]				BUSTURN[3:0]				DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
	Reset value			0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x104	**FMC_BWTR1**	Res.	Res.	ACCMOD[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUSTURN[3:0]				DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
	Reset value			0	0									1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

表 87. FMC 寄存器映射（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
0x10C	**FMC_BWTR2**	Res.	Res.	ACCMOD[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUSTURN[3:0]				DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
	Reset value			0	0									1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x114	**FMC_BWTR3**	Res.	Res.	ACCMOD[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUSTURN[3:0]				DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
	Reset value			0	0									1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x11C	**FMC_BWTR4**	Res.	Res.	ACCMOD[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUSTURN[3:0]				DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
	Reset value			0	0									1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x80	**FMC_PCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCPS[2:0]				TAR[3:0]				TCLR[3:0]				Res.	Res.	ECCEN	PWID[1:0]		PTYP	PBKEN	PWAITEN	Res.
	Reset value													0	0	0	0	0	0	0	0	0	0	0			0	0	1	1	0	0	
0x84	**FMC_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.	FEMPT	IFEN	ILEN	IREN	IFS	ILS	IRS
	Reset value																										1	0	0	0	0	0	0
0x88	**FMC_PMEM**	MEMHIZx[7:0]								MEMHOLDx[7:0]								MEMWAITx[7:0]								MEMSETx[7:0]							
	Reset value	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0	0
0x8C	**FMC_PATT**	ATTHIZ[7:0]								ATTHOLD[7:0]								ATTWAIT[7:0]								ATTSET[7:0]							
	Reset value	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0	0
0x94	**FMC_ECCR**	ECCx[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
0x140	**FMC_SDCR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RPIPE[1:0]		RBURST	SDCLK[1:0]		WP	CAS[1:0]		NB	MWID[1:0]		NR[1:0]		NC	
	Reset value																		0	0	0	1	1	0	1	0	0	1	0	0	0	0	0
0x144	**FMC_SDCR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RBURST	SDCLK[1:0]		WP	CAS[1:0]		NB	MWID[1:0]		NR[1:0]		NC	
	Reset value																				0	1	1	0	1	0	0	1	0	0	0	0	0
0x148	**FMC_SDTR1**	Res.	Res.	Res.	Res.	TRCD[3:0]				TRP[3:0]				TWR[3:0]				TRC[3:0]				TRAS[3:0]				TXSR[3:0]				TMRD[3:0]			
	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
0x14C	**FMC_SDTR2**	Res.	Res.	Res.	Res.	TRCD[3:0]				TRP[3:0]				TWR[3:0]				TRC[3:0]				TRAS[3:0]				TXSR[3:0]				TMRD[3:0]			
	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
0x150	**FMC_SDCMR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MRD[12:0]														NRFS[3:0]				CTB1	CTB2	MODE[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
0x154	**FMC_SDRTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REIE	COUNT[12:0]													CRE
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x158	**FMC_SDSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUSY	MODES2[1:0]		MODES1[1:0]		Res.
	Reset value																											0	0	0	0	0	

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

14 Quad-SPI 接口 (QUADSPI)

14.1 前言

QUADSPI 是一种专用的通信接口，连接单、双或四（条数据线）SPI Flash 存储介质。该接口可以在以下三种模式下工作：

- 间接模式：使用 QUADSPI 寄存器执行全部操作
- 状态轮询模式：周期性读取外部 Flash 状态寄存器，而且标志位置 1 时会产生中断（如擦除或烧写完成，会产生中断）
- 内存映射模式：外部 Flash 映射到微控制器地址空间，从而系统将其视作内部存储器

采用双闪存模式时，将同时访问两个 Quad-SPI Flash，吞吐量和容量均可提高二倍。

14.2 QUADSPI 主要特性

- 三种功能模式：间接模式、状态轮询模式和内存映射模式
- 双闪存模式，通过并行访问两个 Flash，可同时发送/接收 8 位数据
- 支持 SDR 和 DDR 模式
- 针对间接模式和内存映射模式，完全可编程操作码
- 针对间接模式和内存映射模式，完全可编程帧格式
- 集成 FIFO，用于发送和接收
- 允许 8、16 和 32 位数据访问
- 具有适用于间接模式操作的 DMA 通道
- 在达到 FIFO 阈值、超时、操作完成以及发生访问错误时产生中断

14.3 QUADSPI 功能说明

14.3.1 QUADSPI 框图

图 61. QUADSPI 功能框图（双闪存模式禁止）

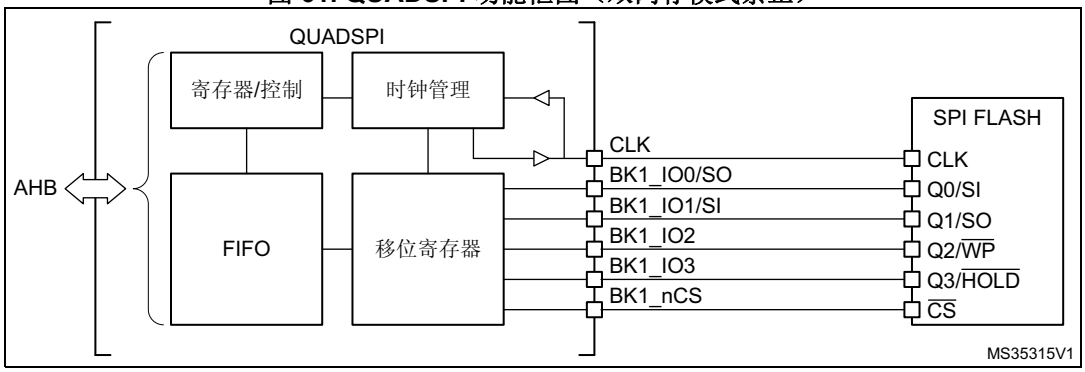
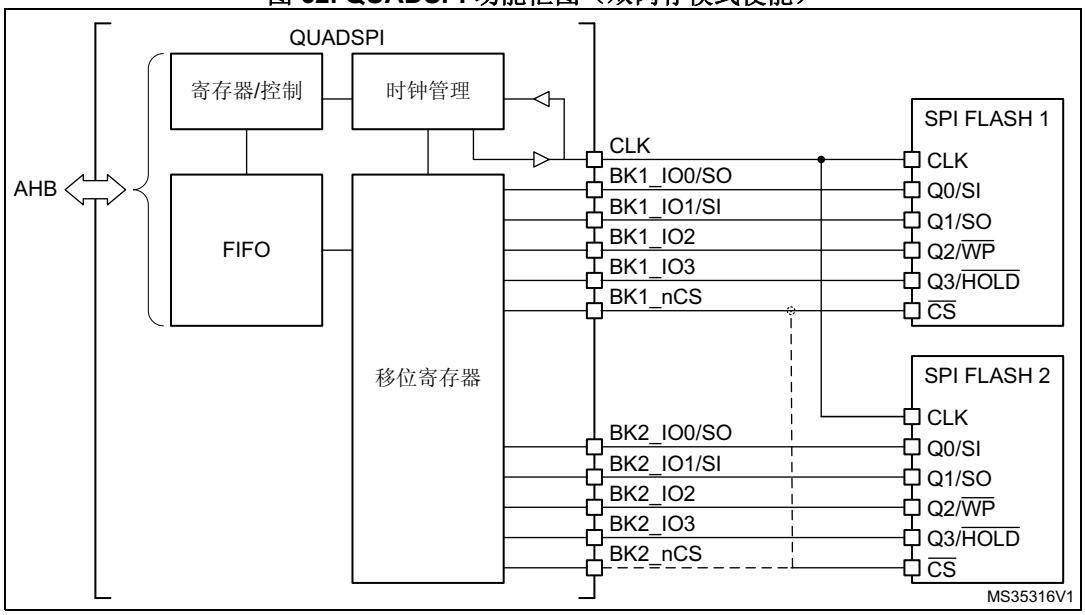


图 62. QUADSPI 功能框图（双闪存模式使能）



在双闪存模式中，QUADSPI 使用 6 个信号连接单个 Flash，使用 10 到 11 个信号连接两个 Flash（FLASH 1 和 FLASH 2）：

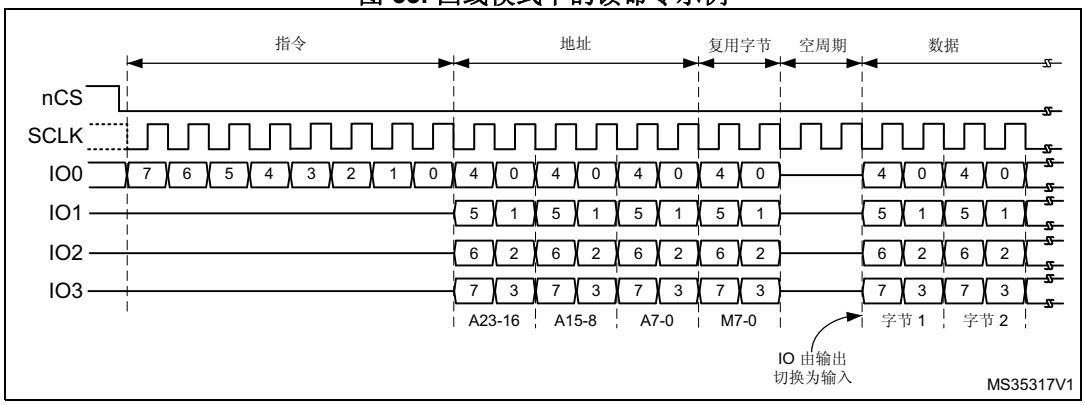
- CLK - 时钟输出，适用于两个存储器
- BK1_IO0/SO - 在双线/四线模式中为双向 IO，单线模式中为串行输出，适用于 FLASH 1
- BK1_IO1/SI - 在双线/四线模式中为双向 IO，单线模式中为串行输入，适用于 FLASH 1
- BK1_IO2 - 在四线模式中为双向 IO，适用于 FLASH 1
- BK1_IO3 - 在四线模式中为双向 IO，适用于 FLASH 1
- BK2_IO0/SO - 在双线/四线模式中为双向 IO，单线模式中为串行输出，适用于 FLASH 2
- BK2_IO1/SI - 在双线/四线模式中为双向 IO，单线模式中为串行输入，适用于 FLASH 2
- BK2_IO2 - 在四线模式中为双向 IO，适用于 FLASH 2
- BK2_IO3 - 在四线模式中为双向 IO，适用于 FLASH 2
- BK1_nCS - 片选输出（低电平有效），适用于 FLASH 1。如果 QUADSPI 始终在双闪存模式下工作，则其也可用于 FLASH 2
- BK2_nCS - 片选输出（低电平有效），适用于 FLASH 2。如果 QUADSPI 始终在双闪存模式下工作，则其也可用于 FLASH 1

14.3.2 QUADSPI 命令序列

QUADSPI 通过命令与 Flash 通信 每条命令包括指令、地址、交替字节、空指令和数据这五个阶段 任一阶段均可跳过，但至少要包含指令、地址、交替字节或数据阶段之一。

nCS 在每条指令开始前下降，在每条指令完成后再次上升。

图 63. 四线模式下的读命令示例



指令阶段

这一阶段，将在 QUADSPI_CCR[7:0] 寄存器的 INSTRUCTION 字段中配置的一条 8 位指令发送到 Flash，指定待执行操作的类型。

尽管大多数 Flash 从 IO0/SO 信号（单线 SPI 模式）只能以一次 1 位的方式接收指令，但指令阶段可选择一次发送 2 位（在双线 SPI 模式中通过 IO0/IO1）或一次发送 4 位（在四线 SPI 模式中通过 IO0/IO1/IO2/IO3）。这可通过 QUADSPI_CCR[9:8] 寄存器中的 IMODE[1:0] 字段进行配置。

若 IMODE = 00，则跳过指令阶段，命令序列从地址阶段（如果存在）开始。

地址阶段

在地址阶段，将 1-4 字节发送到 Flash，指示操作地址。待发送的地址字节数在 QUADSPI_CCR[13:12] 寄存器的 ADSIZE[1:0] 字段中进行配置。在间接模式和自动轮询模式下，待发送的地址字节在 QUADSPI_AR 寄存器的 ADDRESS[31:0] 中指定。在内存映射模式下，则通过 AHB（来自于 Cortex® 或 DMA）直接给出地址。

地址阶段可一次发送 1 位（在单线 SPI 模式中通过 SO）、2 位（在双线 SPI 模式中通过 IO0/IO1）或 4 位（在四线 SPI 模式中通过 IO0/IO1/IO2/IO3）。这可通过 QUADSPI_CCR[11:10] 寄存器中的 ADMODE[1:0] 字段进行配置。

若 ADMODE = 00，则跳过地址阶段，命令序列直接进入下一阶段（如果存在）。

交替字节阶段

在交替字节阶段，将 1-4 字节发送到 Flash，一般用于控制操作模式。待发送的交替字节数在 QUADSPI_CCR[17:16] 寄存器的 ABSIZE[1:0] 字段中进行配置。待发送的字节在 QUADSPI_ABR 寄存器中指定。

交替字节阶段可一次发送 1 位（在单线 SPI 模式中通过 SO）、2 位（在双线 SPI 模式中通过 IO0/IO1）或 4 位（在四线 SPI 模式中通过 IO0/IO1/IO2/IO3）。这可通过 QUADSPI_CCR[15:14] 寄存器中的 ABMODE[1:0] 字段进行配置。

若 ABMODE = 00，则跳过交替字节阶段，命令序列直接进入下一阶段（如果存在）。

交替字节阶段存在仅需发送单个半字节而不是一个全字节的情况，比如采用双线模式并且仅使用两个周期发送交替字节时。在这种情况下，固件可采用四线模式 (ABMODE = 11) 并发送一个字节，方法是 ALTERNATE 的位 7 和 3 置“1”（IO3 保持高电平）且位 6 和 2 置“0”（IO2 线保持低电平）。此时，半字节的高 2 位存放在 ALTERNATE 的位 4:3，低 2 位存放在位 1 和 0 中。例如，如果半字节 2 (0010) 通过 IO0/IO1 发送，则 ALTERNATE 应设置为 0x8A (1000_1010)。

空指令周期阶段

在空指令周期阶段，给定的 1-31 个周期内不发送或接收任何数据，目的是当采用更高的时钟频率时，给 Flash 留出准备数据阶段的时间。这一阶段中给定的周期数在 QUADSPI_CCR[22:18] 寄存器的 DCYC[4:0] 字段中指定。在 SDR 和 DDR 模式下，持续时间被指定为一定个数的全时钟周期。

若 DCYC 为零，则跳过空指令周期阶段，命令序列直接进入数据阶段（如果存在）。

空指令周期阶段的操作模式由 DMODE 确定。

为确保数据信号从输出模式转变为输入模式有足够的“周转”时间，使用双线和四线模式从 Flash 接收数据时，至少需要指定一个空指令周期。

数据阶段

在数据阶段，可从 Flash 接收或向其发送任意数量的字节。

在间接模式和自动轮询模式下，待发送/接收的字节数在 QUADSPI_DLR 寄存器中指定。

在间接写入模式下，发送到 Flash 的数据必须写入 QUADSPI_DR 寄存器。在间接读取模式下，通过读取 QUADSPI_DR 寄存器获得从 Flash 接收的数据。

在内存映射模式下，读取的数据通过 AHB 直接发送回 Cortex 或 DMA。

数据阶段可一次发送/接收 1 位（在单线 SPI 模式中通过 SO）、2 位（在双线 SPI 模式中通过 IO0/IO1）或 4 位（在四线 SPI 模式中通过 IO0/IO1/IO2/IO3）。这可通过 QUADSPI_CCR[15:14] 寄存器中的 ABMODE[1:0] 字段进行配置。

若 DMODE = 00，则跳过数据阶段，命令序列在拉高 nCS 时立即完成。这一配置仅可用于仅间接写入模式。

14.3.3 QUADSPI 信号接口协议模式

单线 SPI 模式

传统 SPI 模式允许串行发送/接收单独的 1 位。在此模式下，数据通过 SO 信号（其 I/O 与 IO0 共享）发送到 Flash。从 Flash 接收到的数据通过 SI（其 I/O 与 IO1 共享）送达。

通过将（QUADSPI_CCR 中的）IMODE/ADMODE/ABMODE/DMODE 字段设置为 01，可对不同的命令阶段分别进行配置，以使用此单个位模式。

在每个已配置为单线模式的阶段中：

- IO0 (SO) 处于输出模式
- IO1 (SI) 处于输入模式（高阻抗）
- IO2 处于输出模式并强制置“0”（以禁止“写保护”功能）
- IO3 处于输出模式并强制置“1”（以禁止“保持”功能）

若 DMODE = 01，这对于空指令阶段也同样如此。

双线 SPI 模式

在双线模式下，通过 IO0/IO1 信号同时发送/接收两位。

通过将 QUADSPI_CCR 寄存器的 IMODE/ADMODE/ABMODE/DMODE 字段设置为 10，可对不同的命令阶段分别进行配置，以使用双线 SPI 模式。

在每个已配置为双线模式的阶段中：

- IO0/IO1 在数据阶段进行读取操作时处于高阻态（输入），在其他情况下为输出
- IO2 处于输出模式并强制置“0”
- IO3 处于输出模式并强制置“1”

在空指令阶段，若 DMODE = 01，则 IO0/IO1 始终保持高阻态。

四线 SPI 模式

在四线模式下，通过 IO0/IO1/IO2/IO3 信号同时发送/接收四位。

通过将 QUADSPI_CCR 寄存器的 IMODE/ADMODE/ABMODE/DMODE 字段设置为 11，可对不同的命令阶段分别进行配置，以使用四线 SPI 模式。

在每个已配置为四线模式的阶段中，IO0/IO1/IO2/IO3 在数据阶段进行读取操作时均处于高阻态（输入），在其他情况下为输出。

在空指令阶段中，若 DMODE = 11，则 IO0/IO1/IO2/IO3 均为高阻态。

IO2 和 IO3 仅用于 Quad SPI 模式 如果未配置任何阶段使用四线 SPI 模式，即使 QUADSPI 激活，对应 IO2 和 IO3 的引脚也可用于其他功能。

SDR 模式

默认情况下，DDRM 位 (QUADSPI_CCR[31]) 为 0，QUADSPI 在单倍数据速率 (SDR) 模式下工作。

在 SDR 模式下，当 QUADSPI 驱动 IO0/SO、IO1、IO2、IO3 信号时，这些信号仅在 CLK 的下降沿发生转变。

在 SDR 模式下接收数据时，QUADSPI 假定 Flash 也通过 CLK 的下降沿发送数据。默认情况下 (SSHIFT = 0 时)，将使用 CLK 后续的边沿（上升沿）对信号进行采样。

DDR 模式

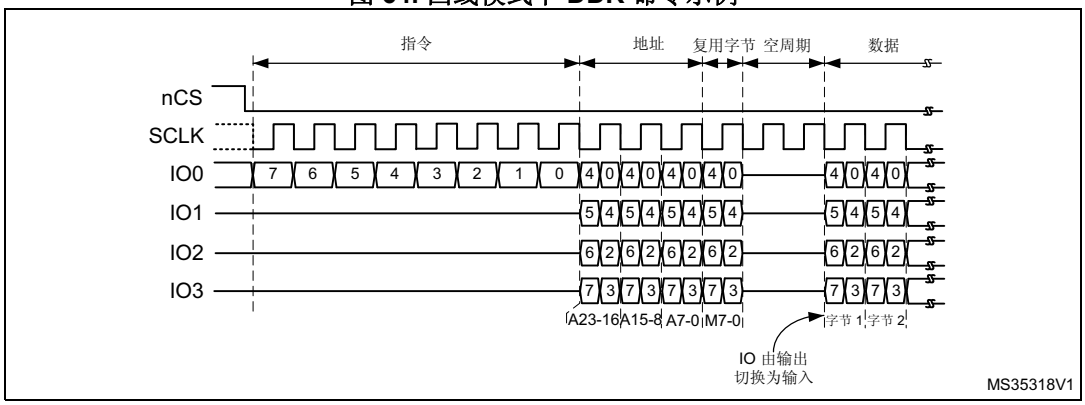
若 DDRM 位 (QUADSPI_CCR[31]) 置 1，则 QUADSPI 在双倍数据速率 (DDR) 模式下工作。

在 DDR 模式下，当 QUADSPI 在地址/交替字节/数据阶段驱动 IO0/SO、IO1、IO2、IO3 信号时，将在 CLK 的每个上升沿和下降沿发送 1 位。

指令阶段不受 DDRM 的影响。始终通过 CLK 的下降沿发送指令。

在 DDR 模式下接收数据时，QUADSPI 假定 Flash 通过 CLK 的上升沿和下降沿均发送数据。若 DDRM = 1，固件必须清零 SSHIFT 位 (QUADSPI_CR[4])。因此，在半个 CLK 周期后（下一个反向边沿）对信号采样。

图 64. 四线模式下 DDR 命令示例



双闪存模式

若 DFM 位 (QUADSPI_CR[6]) 为 1，QUADSPI 处于双闪存模式。QUADSPI 使用两个外部四线 SPI Flash（FLASH 1 和 FLASH 2），在每个周期中发送/接收 8 位（在 DDR 模式下为 16 位），能够有效地将吞吐量和容量扩大一倍。

每个 Flash 使用同一个 CLK 并可选择使用同一个 nCS 信号，但其 IO0、IO1、IO2 和 IO3 信号是各自独立的。

双闪存模式可与单比特模式、双比特模式以及四比特模式结合使用，也可与 SDR 或 DDR 模式相结合。

Flash 的大小在 FSIZE[4:0] (QUADSPI_DCR[20:16]) 中指定，指定的值应能够反映 Flash 的总容量，即单个组件容量的 2 倍。

如果地址 X 为偶数，QUADSPI 赋给地址 X 的字节是存放于 FLASH 1 的地址 X/2 中的字节，QUADSPI 赋给地址 X+1 的字节是存放于 FLASH 2 的地址 X/2 中的字节。也就是说，偶地址中的字节存储于 FLASH 1，奇地址中的字节存储于 FLASH 2。

在双闪存模式下读取 Flash 状态寄存器时，需要读取的字节数是单闪存模式下的 2 倍。这意味着在状态寄存器获取指令到达后，如果每个 Flash 给出 8 个有效位，则 QUADSPI 必须配置为 2 个字节（16 位）的数据长度，它将从每个 Flash 接收 1 个字节。如果每个 Flash 给出一个 16 位的状态，则 QUADSPI 必须配置为读取 4 字节，以在双闪存模式下可获取两个 Flash 的所有状态位。结果（在数据寄存器中）的最低有效字节是 FLASH 1 状态寄存器的最低有效字节，而下一个字节是 FLASH 2 状态寄存器的最低有效字节。数据寄存器的第三个字节是 FLASH 1 的第二个字节，第四个字节是 FLASH 2 的第二个字节（Flash 具有 16 位状态寄存器时）。

偶数个字节必须始终在双闪存模式下访问。因此，若 DRM = 1，则数据长度字段 (QUADSPI_DLR[0]) 的位 0 始终保持为 1。

在双闪存模式下，FLASH 1 接口信号的行为基本上与正常模式下相同。在指令、地址、交替字节以及空指令周期阶段，FLASH 2 接口信号具有与 FLASH 1 接口信号完全相同的波形。也就是说，每个 Flash 总是接收相同的指令与地址。然后，在数据阶段，BK1_IOx 和 BK2_IOx 总线并行传输数据，但发送到 FLASH 1（或从其接收）的数据与 FLASH 2 中的不同。

14.3.4 QUADSPI 间接模式

在间接模式下，通过写入 QUADSPI 寄存器来触发命令；并通过读写数据寄存器来传输数据，就如同对待其他通信外设那样。

若 FMODE = 00 (QUADSPI_CCR[27:26])，则 QUADSPI 处于间接写入模式，字节在数据阶段中发送到 Flash。通过写入数据寄存器 (QUADSPI_DR) 的方式提供数据。

若 FMODE = 01，则 QUADSPI 处于间接读取模式，在数据阶段中从 Flash 接收字节。通过读取 QUADSPI_DR 来获取数据。

读取/写入的字节数在数据长度寄存器 (QUADSPI_DLR) 中指定。如果 QUADSPI_DLR = 0xFFFF_FFFF（全为“1”），则数据长度视为未定义，QUADSPI 将继续传输数据，直到到达（由 FSIZE 定义的）Flash 的结尾。如果不传输任何字节，DMODE (QUADSPI_CCR[25:24]) 应设置为 00。

如果 QUADSPI_DLR = 0xFFFF_FFFF 并且 FSIZE = 0x1F（最大值指示一个 4GB 的 Flash），在此特殊情况下，传输将无限继续下去，仅在出现终止请求或 QUADSPI 被禁止后停止。在读取最后一个存储器地址后（地址为 0xFFFF_FFFF），将从地址 = 0x0000_0000 开始继续读取。

当发送或接收的字节数达到编程设定值时，如果 TCIE = 1，则 TCF 置 1 并产生中断。在数据数量不确定的情况下，将根据 QUADSPI_CR 中定义的 Flash 大小，在达到外部 SPI 的限制时，TCF 置 1。

触发命令启动

从本质上讲，在固件给出命令所需的最后一点信息时，命令即会启动。根据 QUADSPI 的配置，在间接模式下有三种触发命令启动的方式。在出现以下情形时，命令立即启动：

1. 对 INSTRUCTION[7:0] (QUADSPI_CCR) 执行写入操作，如果没有地址是必需的（当 ADMODE = 00）并且不需要固件提供数据（当 FMODE = 01 或 DMODE = 00）
2. 对 ADDRESS[31:0] (QUADSPI_AR) 执行写入操作，如果地址是必需的（当 ADMODE = 00）并且不需要固件提供数据 （当 FMODE = 01 或 DMODE = 00）
3. 对 DATA[31:0] (QUADSPI_DR) 执行写入操作，如果地址是必需的（当 ADMODE != 00）并且需要固件提供数据（当 FMODE = 00 并且 DMODE != 00）

写入交替字节寄存器 (QUADSPI_ABR) 始终不会触发命令启动。如果需要交替字节，必须预先进行编程。

如果命令启动，BUSY 位（QUADSPI_SR 的位 5）将自动置 1。

FIFO 和数据管理

在间接模式中，数据将通过 QUADSPI 内部的一个 32 字节 FIFO。FLEVEL[5:0] (QUADSPI_SR[13:8]) 指示 FIFO 目前保存了多少字节。

在间接写入模式下 (FMODE = 00)，固件写入 QUADSPI_DR 时，将在 FIFO 中加入数据。字写入将在 FIFO 中增加 4 个字节，半字写入增加 2 个字节，而字节写入仅增加 1 个字节。如果固件在 FIFO 中加入的数据过多（超过 DL[31:0] 指示的值），将在写入操作结束（TCF 置 1）时从 FIFO 中清除超出的字节。

对 QUADSPI_DR 的字节/半字访问必须仅针对该 32 位寄存器的最低有效字节/半字。

FTHRES[3:0] 用于定义 FIFO 的阈值 如果达到阈值，FTF（FIFO 阈值标志）置 1 在间接读取模式下，从 FIFO 中读取的有效字节数超过阈值时，FTF 置 1。从 Flash 中读取最后一个字节后，如果 FIFO 中依然有数据，则无论 FTHRES 的设置为何，FTF 也都会置 1。在间接写入模式下，当 FIFO 中的空字节数超过阈值时，FTF 置 1。

如果 FTIE = 1，则 FTF 置 1 时产生中断。如果 DMAEN = 1，则 FTF 置 1 时启动数据传送。如果阈值条件不再为“真”（CPU 或 DMA 传输了足够的数据后），则 FTF 由 HW 清零。

在间接模式下，当 FIFO 已满，QUADSPI 将暂时停止从 Flash 读取字节以避免上溢。请注意，只有在 FIFO 中的 4 个字节为空 (FLEVEL ≤ 11) 时才会重新开始读取 Flash。因此，若 FTHRES ≥ 13，应用程序必须读取足够的字节以确保 QUADSPI 再次从 Flash 检索数据。否则，只要 11 < FLEVEL < FTHRES，FTF 标志将保持为“0”。

14.3.5 QUADSPI 状态标志轮询模式

在自动轮询模式下，QUADSPI 周期性启动命令以读取一定数量的状态字节（最多 4 个）。可屏蔽接收的字节以隔离一些状态位，从而在所选的位具有定义的值时可产生中断。

对 Flash 的访问最初与在间接读取模式下相同：如果不需要地址 (AMODE = 00)，则在写入 QUADSPI_CCR 时即开始访问。否则，如果需要地址，则在写入 QUADSPI_AR 时开始第一次访问。BUSY 在此时变为高电平，即使在周期性访问期间也保持不变。

在自动轮询模式下，MASK[31:0] (QUADSPI_PSMAR) 的内容用于屏蔽来自 Flash 的数据。如果 MASK[n] = 0，则屏蔽结果的位 n，从而不考虑该位。如果 MASK[n] = 1 并且位 [n] 的内容与 MATCH[n] (QUADSPI_PSMAR) 相同，说明存在位 n 匹配。

如果轮询匹配模式位 (PMM, QUADSPI_CR[23]) 为 0，将激活“AND”匹配模式。这意味着状态匹配标志 (SMF) 仅在全部未屏蔽位均存在匹配时置 1。

如果 PMM = 1，则激活“OR”匹配模式。这意味着 SMF 在任意未屏蔽位存在匹配时置 1。

如果 SMIE = 1，则在 SMF 置 1 时调用一个中断。

如果自动轮询模式停止 (APMS) 位置 1，则操作停止并且 BUSY 位在检测到匹配时清零。否则，BUSY 位保持为“1”，在发生中止或禁止 QUADSPI (EN = 0) 前继续进行周期性访问。

数据寄存器 (QUADSPI_DR) 包含最新接收的状态字节（FIFO 停用）。数据寄存器的内容不受匹配逻辑所用屏蔽方法的影响。FTF 状态位在新一次状态读取完成后置 1，并且 FTF 在数据读取后清零。

14.3.6 QUADSPI 内存映射模式

在配置为内存映射模式时，外部 SPI 器件被视为是内部存储器。

QUADSPI 外设若没有正确配置并使能，禁止访问 QUADSPI Flash 的存储区域。

即使 Flash 容量更大，寻址空间也无法超过 256MB。

如果访问的地址超出 FSIZE 定义的范围但仍在 256MB 范围内，则生成 AHB 错误。此错误的影响具体取决于尝试进行访问的 AHB 主设备：

- 如果为 Cortex® CPU，则生成硬性故障 (Hard fault) 中断
- 如果为 DMA，则生成 DMA 传输错误，并自动禁用相应的 DMA 通道。

支持字节、半字和字访问类型。

支持芯片内执行 (XIP) 操作，QUADSPI 接受下一个微控制器访问并提前加载后面地址中的字节。如果之后访问的是连续地址，由于值已经预取，访问将更快完成。

默认情况下，即便在很长时间内不访问 Flash，QUADSPI 也不会停止预取操作，之前的读取操作将保持激活状态并且 nCS 保持低电平。由于 nCS 保持低电平时，Flash 功耗增加，应用程序可能会激活超时计数器 (TCEN = 1, QUADSPI_CR[3])。从而在 FIFO 中写满预取的数据后，若在 TIMEOUT[15:0] (QUADSPI_LPTR) 个周期的时长内没有访问，则释放 nCS。

BUSY 在第一个存储器映射访问发生时变为高电平。由于进行预取操作，BUSY 在发生超时、中止或外设禁止前不会下降。

14.3.7 QUADSPI Flash 配置

设备配置寄存器 (QUADSPI_DCR) 可用于指定外部 SPI Flash 的特性。

FSIZE[4:0] 字段使用下面的公式定义外部存储器的大小：

Flash 中的字节数 = $2^{[FSIZE+1]}$

FSIZE+1 是对 Flash 寻址所需的地址位数。在间接模式下，Flash 容量最高可达 4GB（使用 32 位进行寻址），但在内存映射模式下的可寻址空间限制为 256MB。

如果 DFM = 1，FSIZE 表示两个 Flash 容量的总和。

QUADSPI 连续执行两条命令时，它在两条命令之间将片选信号 (nCS) 置为高电平默认仅一个 CLK 周期时长。如果 Flash 需要命令之间的时间更长，可使用片选高电平时间 (CSHT) 字段指定 nCS 必须保持高电平的最少 CLK 周期数（最大为 8）。

时钟模式 (CKMODE) 位指示命令之间的 CLK 信号逻辑电平（nCS = 1 时）。

14.3.8 QUADSPI 延迟数据采样

默认情况下，QUADSPI 在 Flash 驱动信号后过半个 CLK 周期才对 Flash 驱动的数据采样。

在外部信号延迟时，这有利于推迟数据采样。使用 SSHIFT 位 (QUADSPI_CR[4])，可将数据采样移位半个 CLK 周期。

DDR 模式下不支持时钟移位：若 DDRM 位置 1，SSHIFT 位必须清零。

14.3.9 QUADSPI 配置

QUADSPI 配置分两个阶段

- QUADSPI IP 配置
- QUADSPI Flash 配置

QUADSPI 在配置完毕并使能后，即可在间接模式、状态轮询模式和内存映射模式这三种操作模式之一下工作。

QUADSPI IP 配置

通过 QUADSPI_CR 配置 QUADSPI IP。用户应配置传入数据的时钟预分频器的分频系数以及采样移位设置。

DDR 模式可通过 DDRM 位进行设置。使能该模式后，在每个时钟的上升沿和下降沿都会发送地址和交替字节以及发送/接收数据。无论 DDRM 位如何设置，都将在 SDR 模式下发送指令。

DMA 请求通过 DMAEN 位置 1 使能。若是用于中断，则相关使能位也可在该阶段置 1。

生成 DMA 请求或生成中断的 FIFO 电平在 FTHRES 位中进行编程。

如果需要超时计数器，则可将 TCEN 位置 1 并在 QUADSPI_LPTR 寄存器中编程超时值。

双闪存模式可通过将 DFM 置 1 来激活。

QUADSPI Flash 配置

与外部目标 Flash 相关的参数通过 QUADSPI_DCR 寄存器进行配置。用户应在 FSIZE 位中编程 Flash 的大小、在 CSHT 位中编程片选保持高电平的最短时间以及在 MODE 位中编程功能模式（模式 0 或模式 3）。

14.3.10 QUADSPI 的用法

使用 FMODE[1:0] (QUADSPI_CCR[27:26]) 选择操作模式。

间接模式的操作步骤

FMODE 编程为 00 可选择间接写入模式，将数据发送到 Flash。FMODE 编程为 01 可选择间接读取模式，读取 Flash 中的数据。

QUADSPI 用于间接模式时，采用以下方式构建帧：

1. 在 QUADSPI_DLR 中指定待读取或写入的字节数
2. 在 QUADSPI_CCR 中指定帧格式、模式和指令代码
3. 在 QUADSPI_ABR 中指定要在地址阶段后立即发送的可选交替字节
4. 在 QUADSPI_CR 中指定工作模式。若 FMODE = 00（间接写入模式）并且 DMAEN = 1，则应在 QUADSPI_CR 前指定 QUADSPI_AR。否则在 QUADSPI_AR 更新前（如果已经使能 DMA 控制器），DMA 便可能写入 QUADSPI_DR。
5. 在 QUADSPI_AR 中指定目标地址
6. 通过 QUADSPI_DR 从 FIFO 读取数据/向 FIFO 写入数据

在写入控制寄存器 (QUADSPI_CR) 时，用户可指定以下设置：

- 使能位 (EN) 设置为“1”
- DMA 使能位 (DMAEN)，用于向/从 RAM 传输/接收数据
- 超时计数器使能位 (TCEN)
- 采样移位设置 (SSHIFT)
- FIFO 阈值 (FTRHES)，以指示 FTF 标志在何时置 1
- 中断使能
- 自动轮询模式参数：匹配模式和停止模式 （在 FMODE = 11 时有效）
- 时钟预分频器

在写入通信配置寄存器 (QUADSPI_CCR) 时，用户指定以下参数：

- 通过 INSTRUCTION 位指定指令字节
- 通过 IMODE 位指定指令发送方式（1/2/4 线）
- 通过 ADMODE 位指定地址发送方式（无/1/2/4 线）
- 通过 ADSIZE 位指定地址长度（8/16/24/32 位）
- 通过 ABMODE 位指定交替字节发送方式（无/1/2/4 线）
- 通过 ABSIZE 位指定交替字节数 (1/2/3/4)
- 通过 DBMODE 位指定是否存在空指令字节
- 通过 DCYC 位指定空指令字节数
- 通过 DMODE 位指定数据发送/接收方式（无/1/2/4 线）

如果无需为某个命令更新地址寄存器 (QUADSPI_AR) 与数据寄存器 (QUADSPI_DR)，则在写入 QUADSPI_CCR 时，该命令序列便立即启动。在 ADMODE 和 DMODE 均为 00 时，或在间接读取模式 (FMODE = 01) 下仅 ADMODE = 00 时，便属于此情况。

在需要地址（ADMODE 不为 00），但无需写入数据寄存器（FMODE = 01 或 DMODE = 00）时，通过写入 QUADSPI_AR 更新地址后，命令序列便立即启动。

在数据传输（FMODE = 00 并且 DMODE ! = 00）中，通过 QUADSPI_DR 写入 FIFO 触发通信启动。

状态标志轮询模式

将 FMODE 字段 (QUADSPI_CCR[27:26]) 设置为 10，使能状态标志轮询模式 在此模式下，将发送编程的帧并周期性检索数据。

每帧中读取的最大数据量为 4 字节。如果 QUADSPI_DLR 请求更多的数据，则忽略多余的部分并仅读取 4 个字节。

在 QUADSPI_PISR 寄存器中指定周期性。

在检索到状态数据后，可在内部进行处理，以达到以下目的：

- 将状态匹配标志位置 1，如果使能，还将产生中断
- 自动停止周期性检索状态字节

接收到的值可通过存储于 QUADSPI_PSMKR 中的值进行屏蔽，并与存储在 QUADSPI_PSMAR 中的值进行或运算或与运算。

若是存在匹配，则状态匹配标志置 1，并且在使能了中断的情况下还将产生中断；如果 AMPS 位置 1，则 QUADSPI 自动停止。

在任何情况下，最新的检索值都在 QUADSPI_DR 中可用。

内存映射模式

在内存映射模式下，外部 Flash 被视为内部存储器，只是存在访问延迟。在该模式下，仅允许对外部 Flash 执行读取操作。

将 QUADSPI_CCR 寄存器中的 FMODE 设置为 11 可进入内存映射模式。

当 AHB 主器件访问存储器映射空间时，将发送已编程的指令和帧。

FIFO 用作预取缓冲区以接受线性读取。在此模式中，对于 QUADSPI_DR 的任何访问均返回零。

数据长度寄存器 (QUADSPI_DLR) 在内存映射模式中无意义。

14.3.11 指令仅发送一次

一些 Flash（例如 Winbound）能够提供一种模式，指令在该模式中仅通过第一个命令序列进行发送，后续的命令根据地址直接启动。用户通过使用 SIOO 位 (QUADSPI_CCR[28]) 可利用此功能的优势。

SIOO 对于所有功能模式（间接模式、状态轮询模式和内存映射模式）均有效。如果 SIOO 位置 1，仅第一条命令发送指令，接着对 QUADSPI_CCR 执行写入操作。后续命令序列都将跳过指令阶段，直到 QUADSPI_CCR 被写入为止。

在 IMODE = 00（无指令）时，SIOO 不起作用。

14.3.12 QUADSPI 差错管理

在以下情况下可能产生错误：

- 在间接模式或状态标志轮询模式下，如果在 QUADSPI_AR 中编程了错误的地址（根据 QUADSPI_DCR 中 FSIZE[4:0] 定义的 Flash 大小）：TEF 将置 1，如果使能，还将产生中断。
- 另外，在间接模式下，如果地址加数据的长度超过 Flash 的大小，TEF 将在访问被触发时置 1。
- 在内存映射模式下，当 AHB 主器件执行的访问超出范围或 QUADSPI 被禁止时：将产生 AHB 错误，以响应故障 AHB 请求。
- 当 AHB 主机访问存储器映射空间，但内存映射模式被禁止时：将产生 AHB 错误，以响应故障 AHB 请求。

14.3.13 QUADSPI 的繁忙位和中止功能

在 QUADSPI 启动对 Flash 的操作时，QUADSPI_SR 中的 BUSY 位自动置 1。

在间接模式下，在 QUADSPI 完成了请求的命令序列并且 FIFO 为空时，BUSY 位复位。

在自动轮询模式下，仅当最后一次周期性访问完成时（因 APMS = 1 时发生匹配，或因中止），BUSY 位才变为低电平。

在内存映射模式下进行第一次访问后，仅在发生超时事件或中止时 BUSY 位变为低电平。

任何操作都可通过将 QUADSPI_CR 中的 ABORT 位置 1 来中止。在完成中止时，BUSY 位和 ABORT 位自动复位，FIFO 清空。

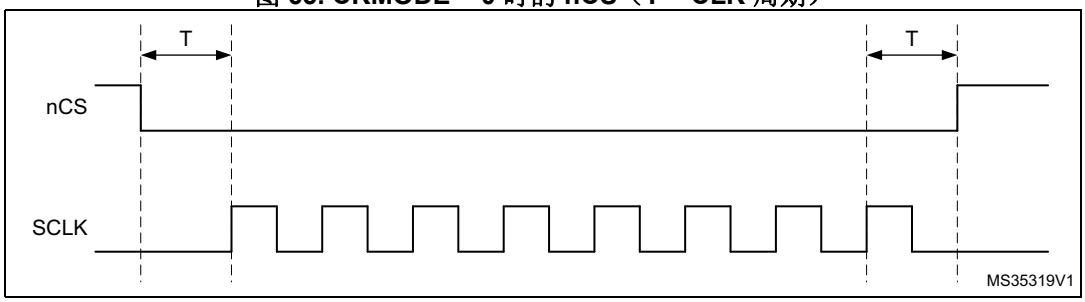
注： *如果中止对状态寄存器的写入操作，有些 Flash 可能发生错误行为。*

14.3.14 nCS 行为

默认情况下，nCS 为高电平，取消选择外部 Flash。nCS 在操作开始前下降，在操作完成时立即上升。

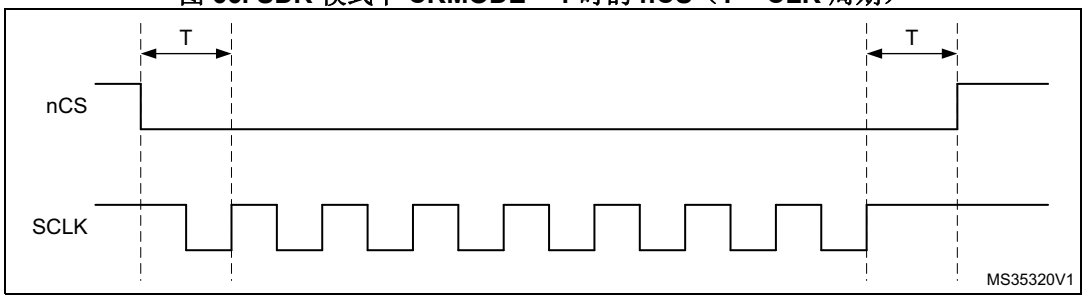
当 CKMODE = 0（“模式 0”，在未进行任何操作时 CLK 保持低电平）时，nCS 在操作首次升高 CLK 边沿时的一个 CLK 周期前降至低电平，在操作最后一次升高 CLK 边沿时的一个 CLK 周期后升至高电平，如*图 65* 所示。

图 65. CKMODE = 0 时的 nCS（T = CLK 周期）



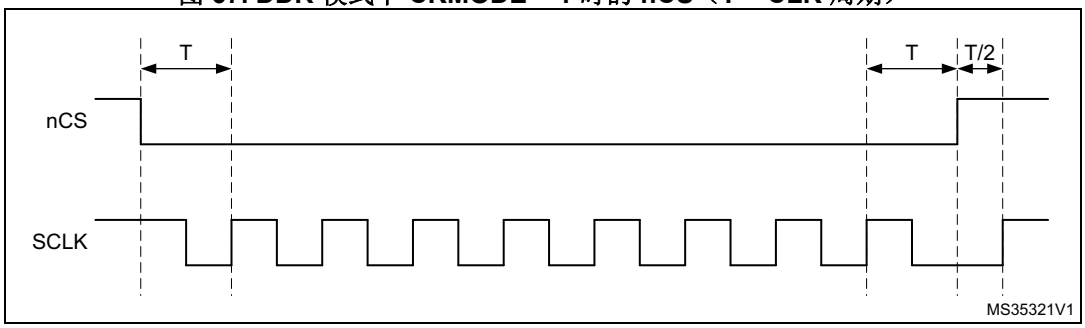
当 CKMODE = 1（“模式 3”，在未进行任何操作时 CLK 升至高电平）且 DDRM = 0（SDR 模式）时，nCS 仍在操作首次升高 CLK 边沿时的一个 CLK 周期前降至低电平，在操作最后一次升高 CLK 边沿时的一个 CLK 周期后升至高电平，如*图 66* 所示。

图 66. SDR 模式下 CKMODE = 1 时的 nCS（T = CLK 周期）



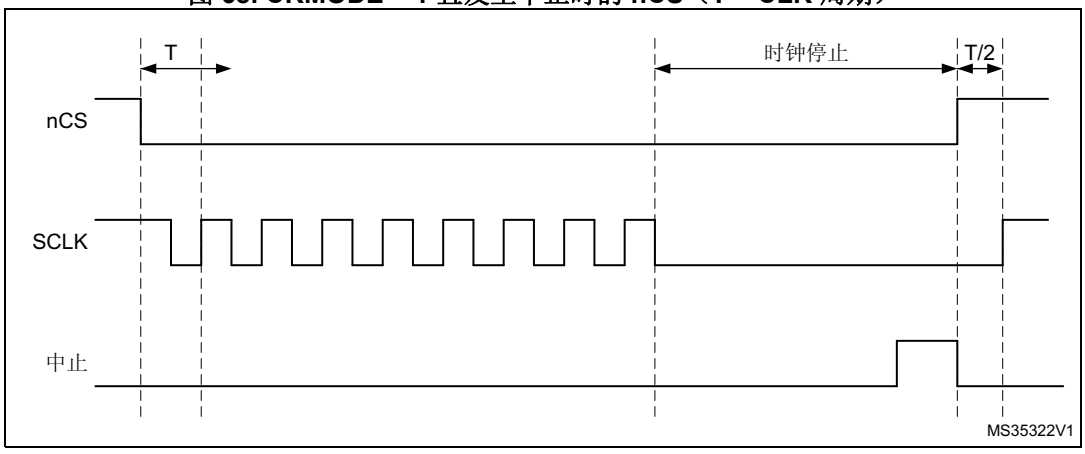
当 CKMODE = 1（模式 3）且 DDRM = 1（DDR 模式）时，nCS 在操作首次升高 CLK 边沿时的一个 CLK 周期前降至低电平，在操作最后一次升高 CLK 边沿时的一个 CLK 周期后升至高电平，如*图 67* 所示。由于 DDR 操作必须伴随下降沿完成，当 nCS 变为高电平时，CLK 为低电平并在经过半个周期后恢复高电平。

图 67. DDR 模式下 CKMODE = 1 时的 nCS（T = CLK 周期）



若 FIFO 在读取操作中保持写满状态或在写入操作中保持为空，则在固件干预 FIFO 前，操作停止并且 CLK 保持低电平。若操作停止时发生中止，则 nCS 在请求中止后即升至高电平，CLK 则在半个周期后升至高电平，如*图 68* 所示。

图 68. CKMODE = 1 且发生中止时的 nCS（T = CLK 周期）



不处于双闪存模式 (DFM = 0) 时，仅访问 FLASH 1，因此 BK2_nCS 保持高电平。在双闪存模式下，BK2_nCS 与 BK1_nCS 的行为完全相同。因此，如果存在 FLASH 2 并且应用程序始终处于双闪存模式，则 FLASH 2 可使用 BK1_nCS，而 BK2_nCS 引脚输出可用于其他功能。

14.4 QUADSPI 中断

发生如下事件时可生成中断：

- 超时
- 状态匹配
- FIFO 阈值
- 传输完成
- 传输错误

可以使用单独的中断使能位以提高灵活性。

表 88. QUADSPI 中断请求

中断事件	事件标志	使能控制位
超时	TOF	TOIE
状态匹配	SMF	SMIE
FIFO 阈值	FTF	FTIE
传输完成	TCF	TCIE
传输错误	TEF	TEIE

14.5 QUADSPI 寄存器

14.5.1 QUADSPI 控制寄存器 (QUADSPI_CR)

QUADSPI control register

偏移地址：0x0000

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRESCALER								PMM	APMS	Res.	TOIE	SMIE	FTIE	TCIE	TEIE
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.	FTHRES					FSEL	DFM	Res.	SSHIFT	TCEN	DMAEN	ABORT	EN
			rw	rw	rw	rw	rw	rw	rw		rw	rw	w1s	rw	w1s

位 31:24 **PRESCALER[7:0]**：时钟预分频器 (Clock prescaler)

该字段定义基于 AHB 时钟生成 CLK 所用的分频系数（值 + 1）。

0：$F_{CLK} = F_{AHB}$，AHB 直接用作 QUADSPI CLK（预分频器被旁路）

1：$F_{CLK} = F_{AHB}/2$

2：$F_{CLK} = F_{AHB}/3$

...

255：$F_{CLK} = F_{AHB}/256$

对于奇数时钟分频系数，CLK 的占空比并非 50%。时钟信号的高电平持续时间比低电平持续时间多一个周期

仅可在 BUSY = 0 时修改该字段。

位 23 **PMM**：轮询匹配模式 (Polling match mode)

该位指示在自动轮询模式期间用来确定是否“匹配”的方法。

0：AND 匹配模式。如果从 Flash 接收的所有未屏蔽位均与匹配寄存器中的对应位相匹配，则 SMF 置 1。

1：OR 匹配模式。如果从 Flash 接收的任意一个未屏蔽位与匹配寄存器中的对应位相匹配，则 SMF 置 1。

仅可在 BUSY = 0 时修改该位。

位 22 **APMS**：自动轮询模式停止 (Automatic poll mode stop)

该位确定在匹配后自动轮询是否停止。

0：仅通过中止或禁用 QUADSPI 停止自动轮询模式。

1：发生匹配时，自动轮询模式停止。

仅可在 BUSY = 0 时修改该位。

位 21 保留，必须保持复位值。

位 20 **TOIE**：TimeOut 中断使能 (TimeOut interrupt enable)

该位使能 TimeOut 中断。

0：禁止中断

1：使能中断

位 19 **SMIE**：状态匹配中断使能 (Status match interrupt enable)
该位使能状态匹配中断。
0：禁止中断
1：使能中断

位 18 **FTIE**：FIFO 阈值中断使能 (FIFO threshold interrupt enable)。
该位使能 FIFO 阈值中断。
0：禁止中断
1：使能中断

位 17 **TCIE**：传输完成中断使能 (Transfer complete interrupt enable)
该位使能传输完成中断。
0：禁止中断
1：使能中断

位 16 **TEIE**：传输错误中断使能 (Transfer error interrupt enable)
该位使能传输错误中断。
0：禁止中断
1：使能中断

位 15:13 保留，必须保持复位值。

位 12:8 **FTHRES[4:0]**：FIFO 阈值级别 (FIFO threshold level)
定义在间接模式下 FIFO 中将导致 FIFO 阈值标志（FTF，QUADSPI_SR[2]）置 1 的字节数阈值。
在间接写入模式下 (FMODE = 00)：
0：如果 FIFO 中存在 1 个或更多空闲字节可供写入，则 FTF 置 1
1：如果 FIFO 中存在 2 个或更多空闲字节可供写入，则 FTF 置 1
...
31：如果 FIFO 中存在 32 个空闲字节可供写入，则 FTF 置 1
在间接读取模式下 (FMODE = 01)：
0：如果 FIFO 中存在 1 个或更多有效字节可供读取，则 FTF 置 1
1：如果 FIFO 中存在 2 个或更多有效字节可供读取，则 FTF 置 1
...
31：如果 FIFO 中存在 32 个有效字节可供读取，则 FTF 置 1
如果 DMAEN = 1，则在更改 FTHRES 值前，必须禁止相应通道的 DMA 控制器。

位 7 **FSEL**：Flash 选择 (Flash memory selection)
该位选择单闪存模式 (DFM = 0) 下要寻址的 Flash。
0：选择 FLASH 1
1：选择 FLASH 2
仅可在 BUSY = 0 时修改该位。
在 DFM = 1 时忽略该位。

位 6 **DFM**：双闪存模式 (Dual-flash mode)
该位激活双闪存模式，同时使用两个外部 Flash 以将吞吐量和容量扩大一倍。
0：禁止双闪存模式
1：使能双闪存模式。
仅可在 BUSY = 0 时修改该位。

位 5 保留，必须保持复位值。

位 4 **SSHIFT**：采样移位 (Sample shift)

默认情况下，QUADSPI 在 Flash 驱动数据后过半个 CLK 周期开始采集数据。使用该位，可考虑外部信号延迟，推迟数据采集。

0：不发生移位

1：移位半个周期

在 DDR 模式下 (DDRM = 1)，固件必须确保 SSHIFT = 0。

仅可在 BUSY = 0 时修改该字段。

位 3 **TOIE**：超时计数器使能 (Timeout counter enable)

该位仅在内存映射模式 (FMODE = 11) 下有效。激活该位后，如果在一段时间（通过 TIMEOUT[15:0] (QUADSPI_LPTR) 定义）内一直没有进行访问，将释放片选 (nCS)（从而降低功耗）。

使能超时计数器。

默认情况下，即便在很长时间内不访问 Flash，QUADSPI 也不会停止预取操作，之前的读取操作将保持激活状态并且 nCS 保持低电平。由于 nCS 保持低电平时，Flash 功耗增加，应用程序可能会激活超时计数器 (TCEN = 1, QUADSPI_CR[3])。从而在 FIFO 中写满预取的数据后，若在 TIMEOUT[15:0] (QUADSPI_LPTR) 个周期的时长内没有访问，则释放 nCS。

0：禁止超时计数器，在内存映射模式中进行访问后，片选 (nCS) 保持激活。

1：使能超时计数器，在内存映射模式下，Flash 持续不活动 TIMEOUT[15:0] 个周期后释放片选 (nCS)。

仅可在 BUSY = 0 时修改该位。

位 2 **DMAEN**：DMA 使能 (DMA enable)

在间接模式下，通过 QUADSPI_DR 寄存器可使用 DMA 输入或输出数据。FIFO 阈值标志 FTF 置 1 时，将触发 DMA 传输。

0：间接模式下禁止 DMA

1：间接模式下使能 DMA

位 1 **ABORT**：中止请求 (Abort request)

该位中止执行中的命令序列。在中止完成时自动复位。

该位可停止当前的传输。

在轮询模式或内存映射模式下，该位也用以复位 APM 位或 DM 位。

0：不请求中止

1：请求中止

位 0 **EN**：使能 (Enable)

使能 QUADSPI。

0：禁止 QUADSPI

1：使能 QUADSPI

14.5.2 QUADSPI 器件配置寄存器 (QUADSPI_CCR)

QUADSPI device configuration register

偏移地址：0x0004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FSIZE				
											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.	CSHT			Res.	Res.	Res.	Res.	Res.	Res.	Res.	CK-MODE
					rw	rw	rw								rw

位 31:21 保留，必须保持复位值。

位 20:16 **FSIZE[4:0]**：Flash 大小 (Flash memory size)

该字段使用下面的公式定义了外部存储器的大小：

Flash 中的字节数 = $2^{[FSIZE+1]}$

FSIZE+1 是对 Flash 寻址所需的地址位数。在间接模式下，Flash 容量最高可达 4GB（使用 32 位进行寻址），但在内存映射模式下的可寻址空间限制为 256MB。

如果 DFM = 1，FSIZE 表示两个 Flash 容量的总和。

仅可在 BUSY = 0 时修改该字段。

位 15:11 保留，必须保持复位值。

位 10:8 **CSHT[2:0]**：片选高电平时间 (Chip select high time)

CSHT+1 定义片选 (nCS) 在发送至 Flash 的命令之间必须保持高电平的最少 CLK 周期数。

0：nCS 在 Flash 命令之间保持高电平至少 1 个周期

1：nCS 在 Flash 命令之间保持高电平至少 2 个周期

...

7：nCS 在 Flash 命令之间保持高电平至少 8 个周期

仅可在 BUSY = 0 时修改该字段。

位 7:1 保留，必须保持复位值。

位 0 **CKMODE**：模式 0/模式 3 (Mode 0 / mode 3)

该位指示 CLK 在命令之间（nCS = 1 时）的电平。

0：nCS 为高电平（片选释放）时，CLK 必须保持低电平。这称为模式 0。

1：nCS 为高电平（片选释放）时，CLK 必须保持高电平。这称为模式 3。

仅可在 BUSY = 0 时修改该字段。

14.5.3 QUADSPI 状态寄存器 (QUADSPI_SR)

QUADSPI status register

偏移地址：0x0008

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	FLEVEL[5:0]						Res.	Res.	BUSY	TOF	SMF	FTF	TCF	TEF
		r	r	r	r	r	r			r	r	r	r	r	r

位 31:14 保留，必须保持复位值。

位 13:8 **FLEVEL[5:0]**：FIFO 级别 (FIFO level)

该字段给出 FIFO 中的有效字节数。FIFO 为空时 FLEVEL = 0，写满时 FLEVEL = 32。在内存映射模式和自动状态轮询模式下，FLEVEL 为零。

位 7:6 保留，必须保持复位值。

位 5 **BUSY**：忙 (Busy)

操作进行时，该位置 1。在对 Flash 的操作完成并且 FIFO 为空时，该位自动清零。

位 4 **TOF**：超时标志 (Timeout flag)

发生超时时该位置 1。向 CTOF 写入 1 可将该位清零。

位 3 **SMF**：状态匹配标志 (Status match flag)

在自动轮询模式下，若未屏蔽的接收数据与匹配寄存器 (QUADSPI_PSMAR) 中的对应位相匹配，则该位置 1。向 CSMF 写入 1 可将该位清零。

位 2 **FTF**：FIFO 阈值标志 (FIFO threshold flag)

在间接模式下，若达到 FIFO 阈值，或从 Flash 读取完成后，FIFO 中留有数据时，该位置 1。只要阈值条件不再为“真”，该位就自动清零。

在自动轮询模式下，每次读取状态寄存器时，该位即置 1；读取数据寄存器时，该位清零。

位 1 **TCF**：传输完成标志 (Transfer complete flag)

在间接模式下，当传输的数据数量达到编程设定值，或在任何模式下传输中止时，该位置 1。向 CTCF 写入 1 时，该位清零。

位 0 **TEF**：传输错误标志 (Transfer error flag)

在间接模式下访问无效地址时，该位置 1。向 CTEF 写入 1 可将该位清零。

14.5.4 QUADSPI 标志清零寄存器 (QUADSPI_FCR)

QUADSPI flag clear register

偏移地址：0x000C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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.	CTOF	CSMF	Res.	CTCF	CTEF
											w1o	w1o		w1o	w1o

位 31:4 保留，必须保持复位值。

位 4 **CTOF**：清除超时标志 (Clear timeout flag)
写入 1 可将 QUADSPI_SR 寄存器中的 TOF 标志清零

位 3 **CSMF**：清除状态匹配标志 (Clear status match flag)
写入 1 可将 QUADSPI_SR 寄存器中的 SMF 标志清零

位 2 保留，必须保持复位值。

位 1 **CTCF**：清除传输完成标志 (Clear transfer complete flag)
写入 1 可将 QUADSPI_SR 寄存器中的 TCF 标志清零

位 0 **CTEF**：清除传输错误标志 (Clear Transfer error flag)
写入 1 可将 QUADSPI_SR 寄存器中的 TEF 标志清零

14.5.5 QUADSPI 数据长度寄存器 (QUADSPI_DLR)

QUADSPI data length register

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
DL[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
DL[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **DL[31: 0]**：数据长度 (Data length)

在间接模式和状态轮询模式下待检索的数据数量（值 + 1）。对状态轮询模式应使用不大于 3 的值（表示 4 字节）。

在间接模式下，所有位置 1 表示未定义长度，QUADSPI 将继续传输数据直到到达由 FSIZE 定义的存储器末尾。

0x0000_0000：传输 1 个字节

0x0000_0001：传输 2 个字节

0x0000_0002：传输 3 个字节

0x0000_0003：传输 4 个字节

...

0xFFFF_FFFD：传输 4,294,967,294 (4G-2) 个字节

0xFFFF_FFFE：传输 4,294,967,295 (4G-1) 个字节

0xFFFF_FFFF：未定义长度 -- 传输所有字节直到到达由 FSIZE 定义的 Flash 的结尾。如果 FSIZE = 0x1F，则读取无限继续下去。

在双闪存模式 (DFM = 1) 下，即使该位写入“0”，DL[0] 也始终保持为“1”，因此保证了每次访问均传输偶数个字节。

该字段在内存映射模式 (FMODE = 10) 下不起作用

仅可在 BUSY = 0 时写入该字段。

14.5.6 QUADSPI 通信配置寄存器 (QUADSPI_CCR)

QUADSPI communication configuration register

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
DDRM	DHHC	Res.	SIOO	FMODE[1:0]		DMODE		Res.	DCYC[4:0]					ABSIZE	
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
ABMODE		ADSIZE		ADMODE		IMODE		INSTRUCTION[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **DDRM**：双倍数据速率模式 (Double data rate mode)
该位为地址、交替字节和数据阶段设置 DDR 模式：
0：禁止 DDR 模式
1：使能 DDR 模式
仅可在 BUSY = 0 时写入该字段。

位 30 **DHHC**：DDR 保持 (DDR hold)
DDR 模式下数据输出延迟 1/4 个 QUADSPI 输出时钟周期：
0：使用模拟延迟来延迟数据输出
1：数据输出延迟 1/4 个 QUADSPI 输出时钟周期。
该特性仅在 DDR 模式下激活。
仅可在 BUSY = 0 时写入该字段。

位 29 保留，必须保持复位值。

位 28 **SIOO**：仅发送指令一次模式
请参见*第 370 页上的第 14.3.11 节：指令仅发送一次*。IMODE = 00 时，该位不起作用。
0：在每个事务中发送指令
1：仅为第一条命令发送指令
仅可在 BUSY = 0 时写入该字段。

位[27:26] **FMODE[1:0]**：功能模式 (Functional mode)
该字段定义 QUADSPI 操作的功能模式：
00：间接写入模式
01：间接读取模式
10：自动轮询模式
11：内存映射模式
如果 DMAEN = 1，则在更改 FMODE 值前，必须禁止相应通道的 DMA 控制器。
仅可在 BUSY = 0 时写入该字段。

位 25:24 **DMODE[1:0]**：数据模式 (Data mode)
该字段定义数据阶段的操作模式：
00：无数据
01：单线传输数据
10：双线传输数据
11：四线传输数据
该字段还定义空指令阶段的操作模式。
仅可在 BUSY = 0 时写入该字段。

位 23 保留，必须保持复位值。

位[22:18] **DCYC[4:0]**：空指令周期数 (Number of dummy cycles)
该字段定义空指令阶段的持续时间。在 SDR 和 DDR 模式下，它指定 CLK 周期数 (0-31)。
仅可在 BUSY = 0 时写入该字段。

位 17:16 **ABSIZE[1:0]**：交替字节长度 (Alternate bytes size)
该位定义交替字节长度：
00：8 位交替字节
01：16 位交替字节
10：24 位交替字节
11：32 位交替字节
仅可在 BUSY = 0 时写入该字段。

位 15:14 **ABMODE[1:0]**：交替字节模式 (Alternate bytes mode)
该字段定义交替字节阶段的操作模式：
00：无交替字节
01：单线传输交替字节
10：双线传输交替字节
11：四线传输交替字节
仅可在 BUSY = 0 时写入该字段。

位 13:12 **ADSIZE[1:0]**：地址长度 (Address size)
该位定义地址长度：
00：8 位地址
01：16 位地址
10：24 位地址
11：32 位地址
仅可在 BUSY = 0 时写入该字段。

位 11:10 **ADDRESS[1:0]**：地址模式 (Address mode)
该字段定义地址阶段的操作模式：
00：无地址
01：单线传输地址
10：双线传输地址
11：四线传输地址
仅可在 BUSY = 0 时写入该字段。

位 9:8 **IMODE[1:0]**：指令模式 (Instruction mode)
该字段定义指令阶段的操作模式：
00：无指令
01：单线传输指令
10：双线传输指令
11：四线传输指令
仅可在 BUSY = 0 时写入该字段。

位 7:0 **INSTRUCTION[7: 0]**：指令 (Instruction)
指定要发送到外部 SPI 设备的指令。
仅可在 BUSY = 0 时写入该字段。

14.5.7 QUADSPI 地址寄存器 (QUADSPI_AR)

QUADSPI address register

偏移地址：0x0018

复位值：0x0000 0000

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

位 31:0 **ADDRESS[31:0]**：地址 (Address)

指定要发送到外部 Flash 的地址。

BUSY = 0 或 FMODE = 11（内存映射模式）时，将忽略写入该字段。

在双闪存模式下，由于地址始终为偶地址，ADDRESS[0] 自动保持为“0”。

14.5.8 QUADSPI 交替字节寄存器 (QUADSPI_ABR)

QUADSPI alternate bytes registers

偏移地址：0x001C

复位值：0x0000 0000

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

位 31:0 **ALTERNATE[31: 0]**：交替字节 (Alternate Bytes)

指定要在地址后立即发送到外部 SPI 设备的可选数据。

仅可在 BUSY = 0 时写入该字段。

14.5.9 QUADSPI 数据寄存器 (QUADSPI_DR)

QUADSPI data 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
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]**：数据 (Data)

指定要与外部 SPI 设备交换的数据。

在间接写入模式下，写入该寄存器的数据在数据阶段发送到 Flash，在此之前则存储于 FIFO。如果 FIFO 太满，将暂停写入，直到 FIFO 具有足够的空间接受要写入的数据才继续。

在间接模式下，读取该寄存器可获得（通过 FIFO）已从 Flash 接收的数据。如果 FIFO 所含字节数比读取操作要求的字节数少并且 BUSY=1，将暂停读取，直到足够的数据出现或传输完成（不分先后）才继续。

在自动轮询模式下，该寄存器包含最后从 Flash 读取的数据（未进行屏蔽）。

支持对该寄存器进行字、半字以及字节访问。在间接写入模式下，字节写入将在 FIFO 中增加 1 个字节，半字写入增加 2 个，而字写入则增加 4 个。类似地，在间接读取模式下，字节读取将擦除 FIFO 中的 1 个字节，半字读取擦除 2 个，而字读取则擦除 4 个。间接模式下的访问必须与此寄存器的最低位对齐：字节读取必须读取 DATA[7:0] 而半字读取必须读取 DATA[15:0]。

14.5.10 QUADSPI 轮询状态屏蔽寄存器 (QUADSPI _PSMKR)

QUADSPI polling status mask 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
MASK[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
MASK[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **MASK[31: 0]**：状态屏蔽 (Status mask)

对在轮询模式下接收的状态字节进行屏蔽

对于位 n：

0：屏蔽在自动轮询模式下所接收数据的位 n，在匹配逻辑中不考虑其值

1：不屏蔽在自动轮询模式下所接收数据的位 n，在匹配逻辑中考虑其值

仅可在 BUSY = 0 时写入该字段。

14.5.11 QUADSPI 轮询状态匹配寄存器 (QUADSPI _PSMAR)

QUADSPI polling status match register

偏移地址：0x0028

复位值：0x0000 0000

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

位 31:0 **MATCH[31: 0]**：状态匹配 (Status match)
该值将与屏蔽状态寄存器比较以进行匹配。
仅可在 BUSY = 0 时写入该字段。

14.5.12 QUADSPI 轮询间隔寄存器 (QUADSPI _PIR)

QUADSPI polling interval register

偏移地址：0x002C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTERVAL[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 **INTERVAL[15: 0]**：轮询间隔 (Polling interval)
自动轮询阶段读取操作之间的 CLK 周期数。
仅可在 BUSY = 0 时写入该字段。

14.5.13 QUADSPI 低功耗超时寄存器 (QUADSPI_LPTR)

QUADSPI low-power timeout register

偏移地址：0x0030

复位值：0x0000 0000

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

在内存映射模式下，每次访问结束后，QUADSPI 将预取后续字节并将其存放在 FIFO 中。该字段指示在 FIFO 写满后，QUADSPI 等待多少个 CLK 时钟周期才让 nCS 升至高电平将 Flash 置为低功耗状态。

仅可在 BUSY = 0 时写入该字段。

14.5.14 QUADSPI 寄存器映射

表 89. QUADSPI 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
0x0000	**QUADSPI_CR**	PRESCALER[7:0]								PMM	APMS	Res.	TOIE	SMIE	FTIE	TCIE	TEIE	Res.	Res.	Res.	FTHRES [4:0]					FSEL	DFM	Res.	SSHIFT	TCEN	DMAEN	ABORT	EN
	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
0x0004	**QUADSPI_DCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FSIZE[4:0]					Res.	Res.	Res.	Res.	Res.	CSHT			Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKMODE
	Reset value												0	0	0	0	0						0	0	0								0
0x0008	**QUADSPI_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLEVEL[6:0]							Res.	Res.	BUSY	TOF	SMF	FTF	TCF	TEF
	Reset value																		0	0	0	0	0	0	0			0	0	0	0	0	0
0x000C	**QUADSPI_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.	CTOF	CSMF	Res.	CTCF	CTEF
	Reset value																												0	0		0	0
0x0010	**QUADSPI_DLR**	DL[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
0x0014	**QUADSPI_CCR**	DDRM	DHHC	Res.	SIOO	FMODE[1:0]		DMODE[1:0]		Res.	DCYC[4:0]					ABSIZE[1:0]		ABMODE[1:0]		ADSIZE[1:0]		ADMODE[1:0]		IMODE[1:0]		INSTRUCTION[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
0x0018	**QUADSPI_AR**	ADDRESS[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
0x001C	**QUADSPI_ABR**	ALTERNATE[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
0x0020	**QUADSPI_DR**	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
0x0024	**QUADSPI_ PSMKR**	MASK[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
0x0028	**QUADSPI_ PSMAR**	MATCH[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
0x002C	**QUADSPI_PIR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INTERVAL[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0030	**QUADSPI_ LPTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TIMEOUT[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 2.2.2 节*。

15 模数转换器 (ADC)

15.1 ADC 简介

12 位 ADC 是逐次趋近型模数转换器。它具有多达 19 个复用通道，可测量来自 16 个外部源、两个内部源和 V_{BAT} 通道的信号。这些通道的 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 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。

15.2 ADC 主要特性

- 可配置 12 位、10 位、8 位或 6 位分辨率
- 在转换结束、注入转换结束以及发生模拟看门狗或溢出事件时产生中断
- 单次和连续转换模式
- 扫描模式，自动转化通道 0 到通道 n 数据
- 数据对齐以保持内置数据一致性
- 可独立设置各通道采样时间
- 外部触发器选项，可为常规转换和注入转换配置极性
- 不连续采样模式
- 双重/三重模式（具有 2 个或更多 ADC 的器件提供）
- 双重/三重 ADC 模式下可配置的 DMA 数据存储
- 双重/三重交替模式下可配置的转换间延迟
- ADC 转换类型（参见数据手册）
- ADC 电源要求：供电在 2.4V 到 3.6V 下可全速运行，供电低至 1.8V 时为慢速运行
- ADC 输入范围：$V_{REF-} \leq V_{IN} \leq V_{REF+}$
- 常规通道转换期间可产生 DMA 请求

图 69 显示了 ADC 的框图。

注： *V_{REF-} 如果可用（取决于封装），则必须将其连接到 V_{SSA}。*

15.3 ADC 功能说明

图 69 给出了单个 ADC 的框图，*表 90* 列出了 ADC 的引脚说明。

图 69. 单个 ADC 框图

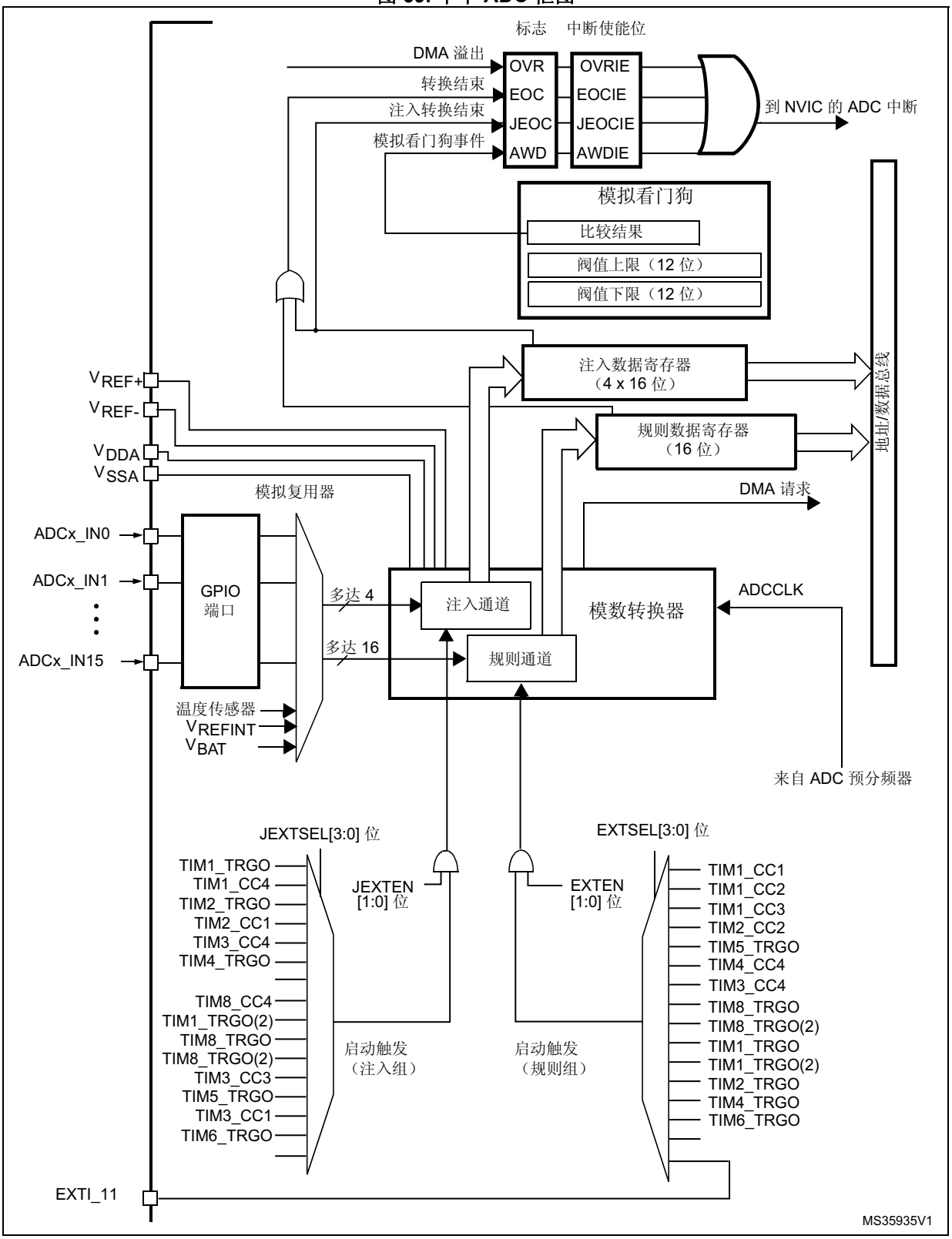

标志
中断使能位
DMA 溢出
转换结束
注入转换结束
模拟看门狗事件
OVR
EOC
JEOC
AWD
OVRIE
EOCIE
JEOCIE
AWDIE
到 NVIC 的 ADC 中断
模拟看门狗
比较结果
阈值上限（12 位）
阈值下限（12 位）
注入数据寄存器
（4 x 16 位）
规则数据寄存器
（16 位）
地址/数据总线
DMA 请求
VREF+
VREF-
VDDA
VSSA
模拟复用器
ADCx_IN0
ADCx_IN1
ADCx_IN15
GPIO
端口
多达 4
多达 16
注入通道
规则通道
模数转换器
ADCCLK
温度传感器
VREFINT
VBAT
来自 ADC 预分频器
JEXTSEL[3:0] 位
EXTSEL[3:0] 位
TIM1_TRGO
TIM1_CC4
TIM2_TRGO
TIM2_CC1
TIM3_CC4
TIM4_TRGO
TIM8_CC4
TIM1_TRGO(2)
TIM8_TRGO
TIM8_TRGO(2)
TIM3_CC3
TIM5_TRGO
TIM3_CC1
TIM6_TRGO
JEXTEN
[1:0] 位
EXTEN
[1:0] 位
启动触发
（注入组）
启动触发
（规则组）
TIM1_CC1
TIM1_CC2
TIM1_CC3
TIM2_CC2
TIM5_TRGO
TIM4_CC4
TIM3_CC4
TIM8_TRGO
TIM8_TRGO(2)
TIM1_TRGO
TIM1_TRGO(2)
TIM2_TRGO
TIM4_TRGO
TIM6_TRGO
EXTI_11
MS35935V1

表 90. ADC 引脚

名称	信号类型	备注
V_{REF+}	正模拟参考电压输入	ADC 高/正参考电压，1.8 V ≤ V_{REF+} ≤ V_{DDA}
V_{DDA}	模拟电源输入	模拟电源电压等于 V_{DD}， 全速运行时，2.4 V ≤ V_{DDA} ≤ V_{DD} (3.6 V) 低速运行时，1.8 V ≤ V_{DDA} ≤ V_{DD} (3.6 V)
V_{REF-}	负模拟参考电压输入	ADC 低/负参考电压， V_{REF-} = V_{SSA}
V_{SSA}	模拟电源地输入	模拟电源地电压等于 V_{SS}
ADCx_IN[15:0]	模拟输入信号	16 个模拟输入通道

15.3.1 ADC 开关控制

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

SWSTART 或 JSWSTART 位置 1 时，启动 AD 转换。

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

15.3.2 ADC1/2 和 ADC3 连接功能

ADC1、ADC2 和 ADC3 紧密耦合并共用某些外部通道，

如*图 70*、*图 71* 和*图 72* 所示。

图 70. ADC1连接功能

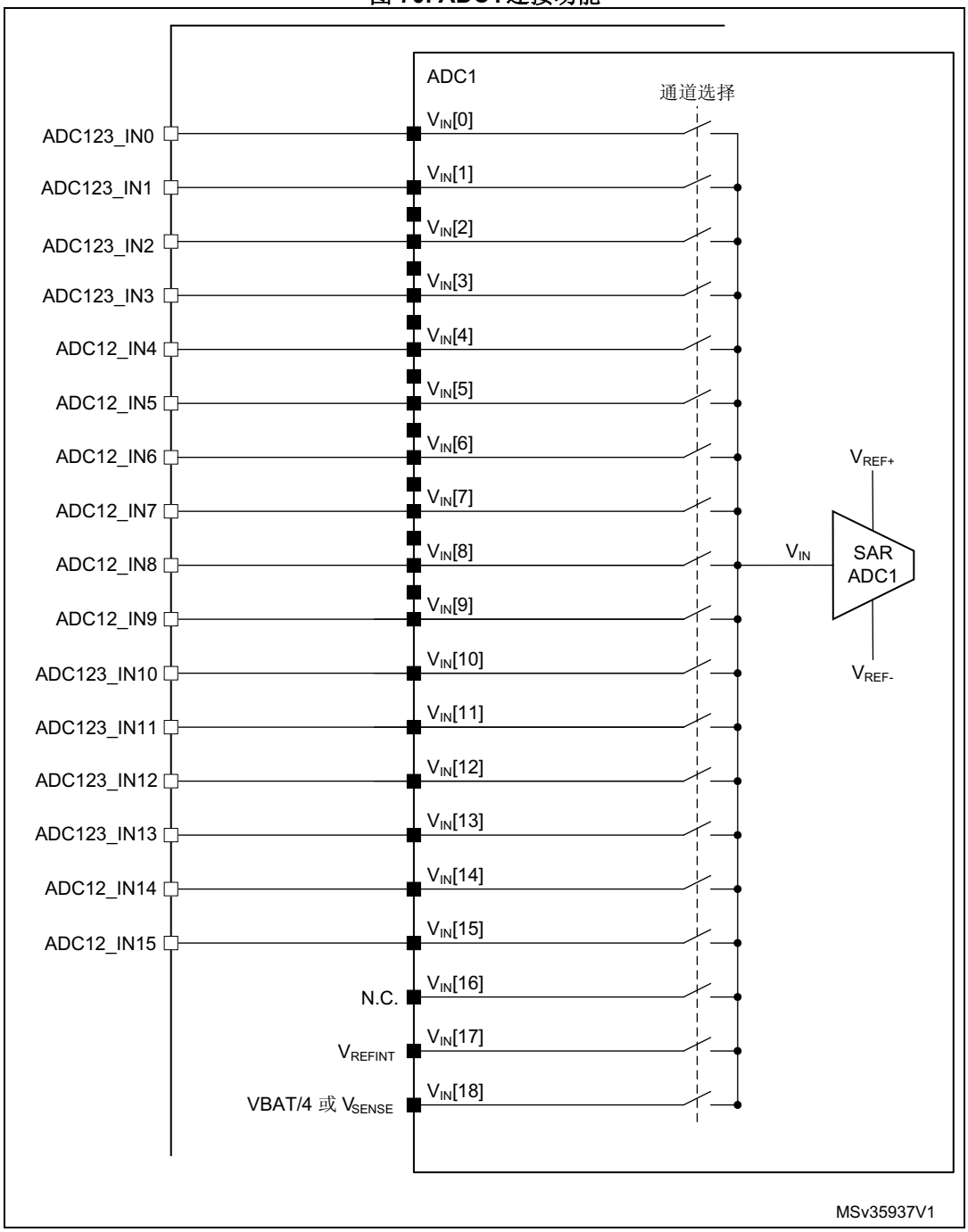


图 71. ADC2连接功能

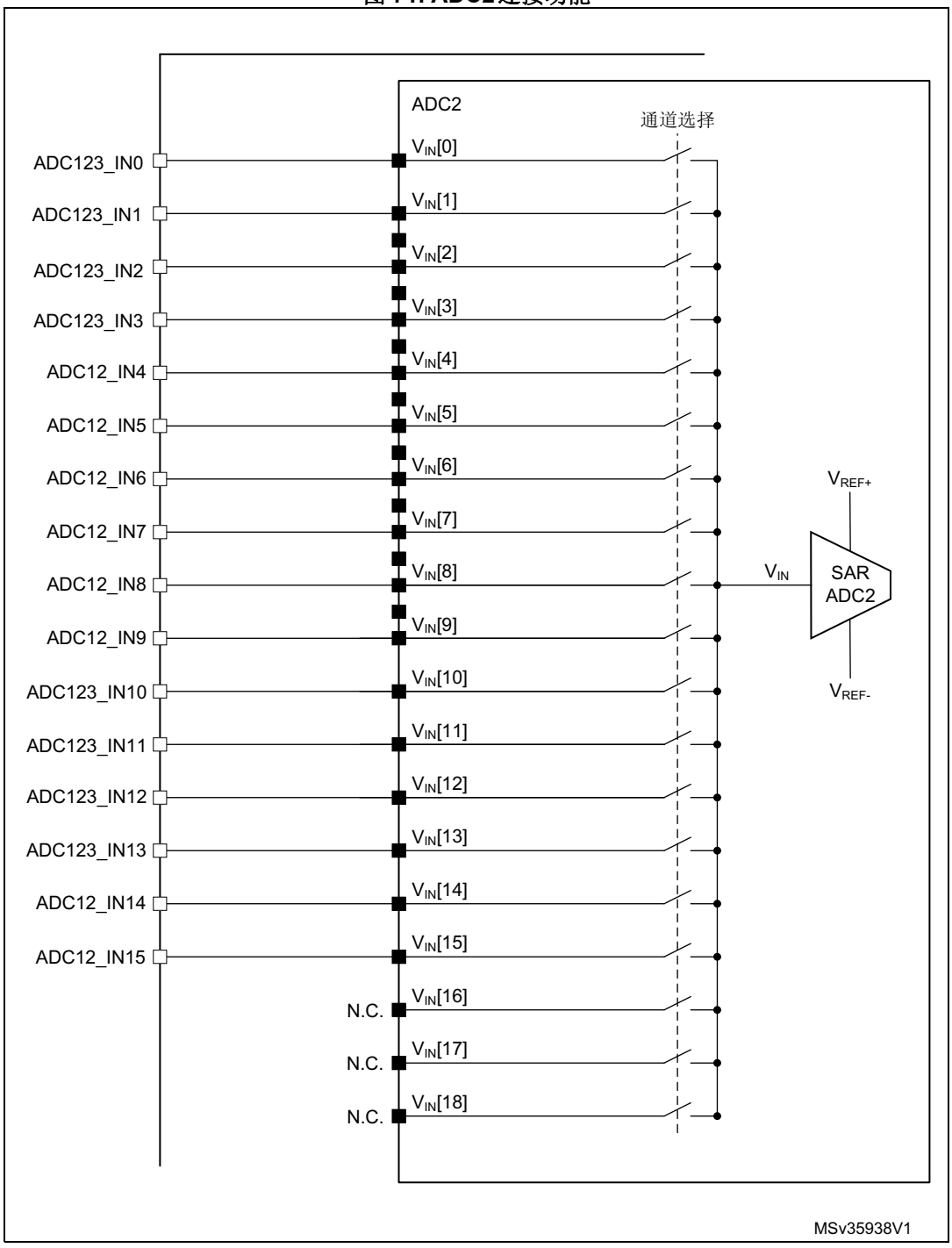
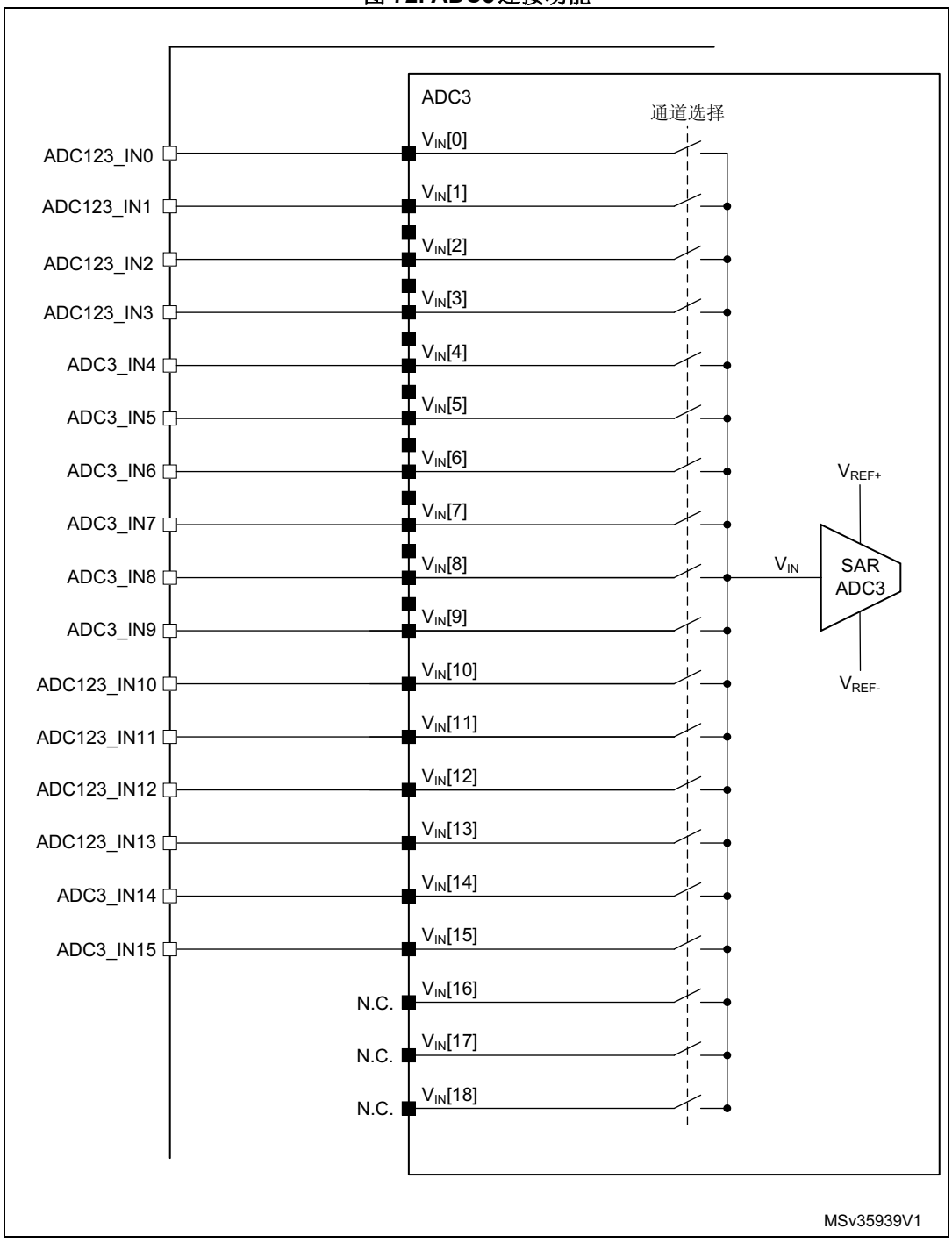


图 72. ADC3连接功能



15.3.3 ADC 时钟

ADC 具有两个时钟方案：

- 模拟电路时钟：ADCCLK，对于所有 ADC 均通用

  此时钟来自于经可编程预分频器分频的 APB2 时钟，该预分频器允许 ADC 在 f_{PCLK2}/2、/4、/6 或 /8 下工作。有关 ADCCLK 的最大值，请参见数据手册。
- 用于数字接口的时钟（用于寄存器读/写访问）

  此时钟等效于 APB2 时钟。可以通过 RCC APB2 外设时钟使能寄存器 (RCC_APB2ENR) 分别为每个 ADC 使能/禁止数字接口时钟。

15.3.4 通道选择 (Channel selection)

有 16 条复用通道。可以将转换分为两组：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：ADC_IN3、ADC_IN8、ADC_IN2、ADC_IN2、ADC_IN0、ADC_IN2、ADC_IN2、ADC_IN15。

- 一个**常规转换组**最多由 16 个转换构成。必须在 ADC_SQRx 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。

如果在转换期间修改 ADC_SQRx 或 ADC_JSQR 寄存器，将复位当前转换并向 ADC 发送一个新的启动脉冲，以转换新选择的组。

温度传感器、V_{REFINT} 和 V_{BAT} 内部通道

- 温度传感器在内部连接到与 VBAT 共用的通道 ADC1_IN18。一次只能选择一个转换（温度传感器或 VBAT）。同时设置了温度传感器和 VBAT 转换时，将只进行 VBAT 转换。

  内部参考电压 VREFINT 连接到 ADC1_IN17。

V_{BAT} 通道连接到 ADC1_IN18 通道。该通道也可转换为注入通道或常规通道。

注： *温度传感器、V_{REFINT} 和 V_{BAT} 通道只在主 ADC1 外设上可用。*

15.3.5 单次转换模式

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

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

完成所选通道的转换之后：

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

然后，ADC 停止。

15.3.6 连续转换模式

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

每次转换之后：

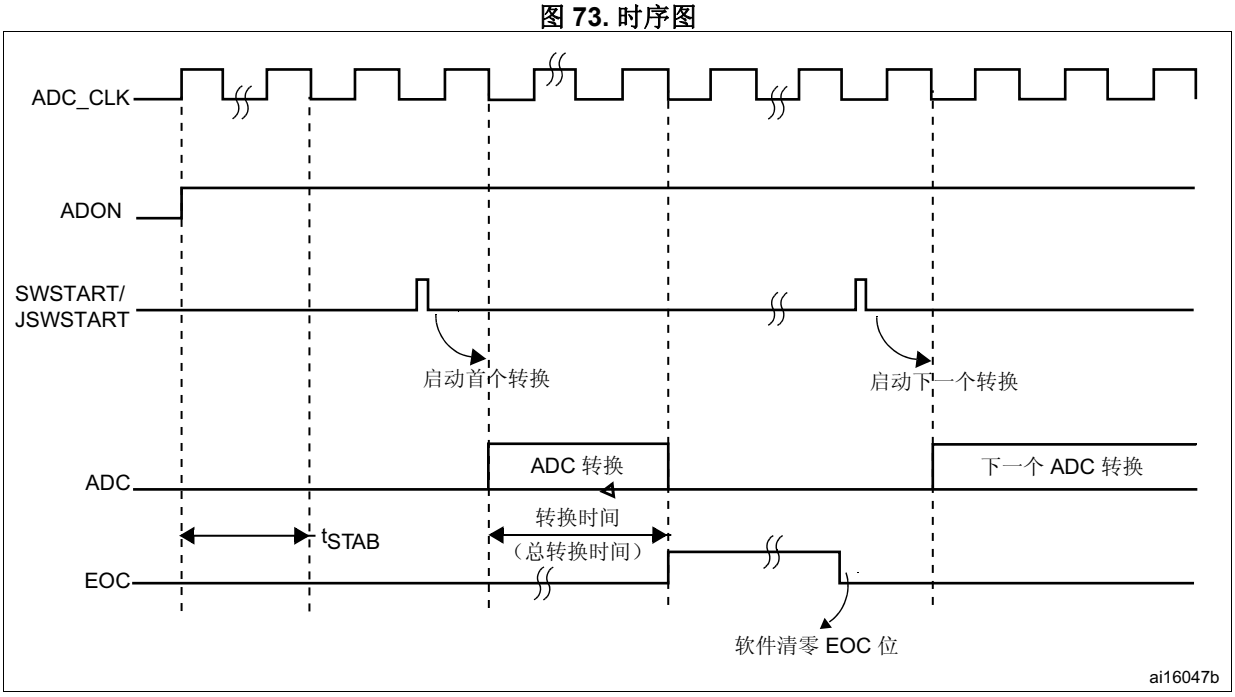
- 如果转换了常规通道组：
  - 上次转换的数据存储在 16 位 ADC_DR 寄存器中
  - EOC（转换结束）标志置 1
  - EOCIE 位置 1 时将产生中断

注： *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，请参见自动注入一节。*

15.3.7 时序图

如*图 73* 所示，ADC 在开始精确转换之前需要一段稳定时间 t_{STAB}。ADC 开始转换并经过 15 个时钟周期后，EOC 标志置 1，转换结果存放在 16 位 ADC 数据寄存器中。

图 73. 时序图



15.3.8 模拟看门狗

如果 ADC 转换的模拟电压低于阈值下限或高于阈值上限，则 AWD 模拟看门狗状态位会置 1。这些阈值在 ADC_HTR 和 ADC_LTR 16 位寄存器的 12 个最低有效位中进行编程。可以使用 ADC_CR1 寄存器中的 AWDIE 位使能中断。

阈值与 ADC_CR2 寄存器中的 ALIGN 位的所选对齐方式无关。在对齐之前，会将模拟电压与阈值上限和下限进行比较。

表 91 介绍了应如何配置 ADC_CR1 寄存器才能在一个或多个通道上使能模拟看门狗。

图 74. 模拟看门狗的保护区域



表 91. 模拟看门狗通道选择

模拟看门狗保护的通道	ADC_CR1 寄存器控制位（x = 无关）		
	AWDSGL 位	AWDEN 位	JAWDEN 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 AWDCH[4:0] 位选择

15.3.9 扫描模式 (Scan mode)

此模式用于扫描一组模拟通道。

通过将 ADC_CR1 寄存器中的 SCAN 位置 1 来选择扫描模式。将此位置 1 后，ADC 会扫描在 ADC_SQRx 寄存器（对于常规通道）或 ADC_JSQR 寄存器（对于注入通道）中选择的所有通道。组中的每个通道都执行一次转换。每次转换结束后，会自动转换该组中的下一个通道。如果将 CONT 位置 1，常规通道转换不会在组中最后一个所选通道处停止，而是再次从第一个所选通道继续转换。

如果将 DMA 位置 1，则在每次常规通道转换之后，均使用直接存储器访问 (DMA) 控制器将转换自常规通道组的数据（存储在 ADC_DR 寄存器中）传输到 SRAM。

在以下情况下，ADC_SR 寄存器中的 EOC 位置 1：

- 如果 EOCS 位清零，在每个常规组序列转换结束时
- 如果 EOCS 位置 1，在每个常规通道转换结束时

从注入通道转换的数据始终存储在 ADC_JDRx 寄存器中。

15.3.10 注入通道管理

触发注入

要使用触发注入，必须将 ADC_CR1 寄存器中的 JAUTO 位清零。

1. 通过外部触发或将 ADC_CR2 寄存器中的 SWSTART 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 JSWSTART 位置 1，则当前的转换会复位，并且注入通道序列会切换为单次扫描模式。
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。
   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 75* 显示了相应的时序图。

注： *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 3 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

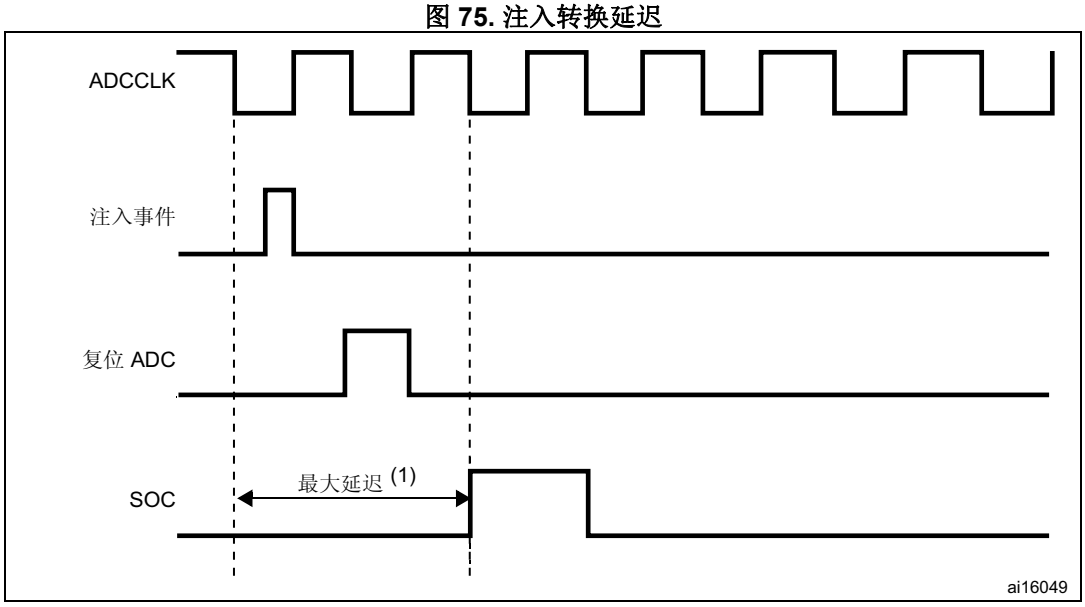
如果将 JAUTO 位置 1，则注入组中的通道会在常规组通道之后自动转换。这可用于转换最多由 20 个转换构成的序列，这些转换在 ADC_SQRx 和 ADC_JSQR 寄存器中编程。

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： *不能同时使用自动注入和不连续采样模式。*

图 75. 注入转换延迟



1. 有关延迟的最大值，请参见 STM32F75xxx 和 STM32F74xxx 数据手册中的电气特性部分。

15.3.11 不连续采样模式

常规组

可将 ADC_CR1 寄存器中的 DISCEN 位置 1 来使能此模式。该模式可用于转换含有 n ($n \leq 8$) 个转换的短序列，该短序列是在 ADC_SQRx 寄存器中选择的转换序列的一部分。可通过写入 ADC_CR1 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_SQRx 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：

- n = 3，要转换的通道 = 0、1、2、3、6、7、9、10
- 第一次触发：转换序列 0、1、2。每次转换时都生成 EOC 事件
- 第二次触发：转换序列 3、6、7。每次转换时都生成 EOC 事件
- 第三次触发：转换序列 9、10。每次转换时都生成 EOC 事件
- 第四次触发：转换序列为 0、1、2。每次转换时都生成 EOC 事件

注： *在不连续采样模式下转换常规组时，不会出现翻转。*

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 4 次触发重新转换了第 1 个子组中的通道 0、1 和 2。

注入组

可将 ADC_CR1 寄存器中的 JDISCEN 位置 1 来使能此模式。在出现外部触发事件之后，可使用该模式逐通道转换在 ADC_JSQR 寄存器中选择的序列。

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：

n = 1，要转换的通道 = 1、2、3
第一次触发：转换通道 1
第二次触发：转换通道 2
第三次触发：转换通道 3 并生成 JEOC 事件
第四次触发：通道 1

注： *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 4 次触发重新转换了第 1 个注入通道。*

不能同时使用自动注入和不连续采样模式。

不得同时为常规组和注入组设置不连续采样模式。只能针对一个组使能不连续采样模式。

15.4 数据对齐 (Data alignment)

ADC_CR2 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 76* 和*图 77* 所示。

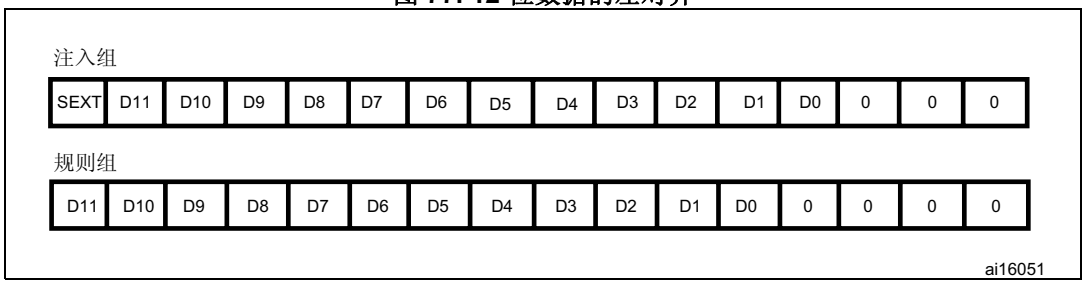
注入通道组的转换数据将减去 ADC_JOFRx 寄存器中写入的用户自定义偏移量，因此结果可以是一个负值。SEXT 位表示扩展的符号值。

对于常规组中的通道，不会减去任何偏移量，因此只有十二个位有效。

图 76. 12 位数据的右对齐



图 77. 12 位数据的左对齐



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 78* 所示。

图 78. 6 位数据的左对齐



15.5 可独立设置各通道采样时间

ADC 会在数个 ADCCLK 周期内对输入电压进行采样，可使用 ADC_SMPR1 和 ADC_SMPR2 寄存器中的 SMP[2:0] 位修改周期数。每个通道均可以使用不同的采样时间进行采样。

总转换时间的计算公式如下：

T_{conv} = 采样时间 + 12 个周期

示例：

ADCCLK = 30 MHz 且采样时间 = 3 个周期时：

T_{conv} = 3 + 12 = 15 个周期 = 0.5 µs（APB2 为 60 MHz 时）

15.6 外部触发转换和触发极性

可以通过外部事件（例如，定时器捕获、EXTI 中断线）触发转换。如果 EXTEN[1:0] 控制位（对于行规转换）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。*表 92* 提供了 EXTEN[1:0] 和 JEXTEN[1:0] 值与触发极性之间的对应关系。

表 92. 配置触发极性

源	EXTEN[1:0]/JEXTEN[1:0]
禁止触发检测	00
在上升沿时检测	01
在下降沿时检测	10
在上升沿和下降沿均检测	11

注： *可以实时更改外部触发的极性。*

EXTSEL[3:0] 和 JEXTSEL[3:0] 控制位用于从 16 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。

表 93 给出了可用于常规转换的外部触发。

表 93. 常规通道的外部触发

源	类型	EXTSEL[3:0]
TIM1_CC1 事件	片上定时器的内部信号	0000
TIM1_CC2 事件		0001
TIM1_CC3 事件		0010
TIM2_CC2 事件		0011
TIM5_TRGO 事件		0100
TIM4_CC4 事件		0101
TIM3_CC4		0110
TIM8_TRGO 事件		0111
TIM8_TRGO(2) 事件		1000
TIM1_TRGO 事件		1001
TIM1_TRGO(2) 事件		1010
TIM2_TRGO 事件		1011
TIM4_TRGO 事件		1100
TIM6_TRGO 事件		1101
EXTI 线 11	外部引脚	1111

表 94 给出了可用于注入转换的外部触发。

表 94. 注入通道的外部触发

源	连接类型	JEXTSEL[3:0]
TIM1_TRGO 事件	片上定时器的内部信号	0000
TIM1_CC4 事件		0001
TIM2_TRGO 事件		0010
TIM2_CC1 事件		0011
TIM3_CC4 事件		0100
TIM4_TRGO 事件		0101

表 94. 注入通道的外部触发（续）

源	连接类型	JEXTSEL[3:0]
TIM8_CC4 事件	片上定时器的内部信号	0111
TIM1_TRGO(2) 事件		1000
TIM8_TRGO 事件		1001
TIM8_TRGO(2) 事件		1010
TIM3_CC3 事件		1011
TIM5_TRGO 事件		1100
TIM3_CC1 事件		1101
TIM6_TRGO 事件		1110

可通过将 ADC_CR2 寄存器中的 SWSTART（对于常规转换）或 JSWSTART（对于注入转换）位置 1 来产生软件源触发事件。

可通过注入触发中断常规组转换。

注： *可以实时更改触发选择。不过，当更改触发选择时，会在 1 个 APB 时钟周期的时间范围内禁止触发检测。这是为了避免在转换期间出现意外检测。*

15.7 快速转换模式

可通过降低 ADC 分辨率来执行快速转换。RES 位用于选择数据寄存器中可用的位数。每种分辨率的最小转换时间如下：

- 12 位：3 + 12 = 15 ADCCLK 周期
- 10 位：3 + 10 = 13 ADCCLK 周期
- 8 位：3 + 8 = 11 ADCCLK 周期
- 6 位：3 + 6 = 9 ADCCLK 周期

15.8 数据管理

15.8.1 使用 DMA

由于常规通道组只有一个数据寄存器，因此，对于多个常规通道的转换，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。

在使能 DMA 模式的情况下（ADC_CR2 寄存器中的 DMA 位置 1），每完成常规通道组中的一个通道转换后，都会生成一个 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，如果数据丢失（溢出），则会将 ADC_SR 寄存器中的 OVR 位置 1 并生成一个中断（如果 OVRIE 使能位已置 1）。随后会禁止 DMA 传输并且不再接受 DMA 请求。在这种情况下，如果生成 DMA 请求，则会中止正在进行的常规转换并忽略之后的常规触发。随后需要将所使用的 DMA 流中的 OVR 标志和 DMAEN 位清零，并重新初始化 DMA 和 ADC，以将需要的转换通道数据传输到正确的存储器单元。只有这样，才能恢复转换并再次使能数据传输。注入通道转换不会受到溢出错误的影响。

在 DMA 模式下，当 OVR = 1 时，传送完最后一个有效数据后会阻止 DMA 请求，这意味着传输到 RAM 的所有数据均被视为有效。

在最后一次 DMA 传输（DMA 控制器的 DMA_SxNTR 寄存器中配置的传输次数）结束时：

- 如果将 ADC_CR2 寄存器中的 DDS 位清零，则不会向 DMA 控制器发出新的 DMA 请求（这可避免产生溢出错误）。不过，硬件不会将 DMA 位清零。必须将该位写入 0 然后写入 1 才能启动新的传输。
- 如果将 DDS 位置 1，则可继续生成请求。从而允许在双缓冲区循环模式下配置 DMA。

要在使用 DMA 时将 ADC 从 OVR 状态中恢复，请按以下步骤操作：

1. 重新初始化 DMA（调整目标地址和 NDTR 计数器）
2. 将 ADC_SR 寄存器中的 ADC OVR 位清零
3. 触发 ADC 以开始转换。

15.8.2 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，必须将 ADC_CR2 寄存器中的 EOCS 位置 1，才能使 EOC 状态位在每次转换结束时置 1，而不仅是在序列结束时置 1。当 EOCS = 1 时，会自动使能溢出检测。因此，每当转换结束时，EOC 都会置 1，并且可以读取 ADC_DR 寄存器。溢出管理与使用 DMA 时的管理相同。

要在 EOCS 位置 1 时将 ADC 从 OVR 状态中恢复，请按以下步骤操作：

1. 将 ADC_SR 寄存器中的 ADC OVR 位清零
2. 触发 ADC 以开始转换。

15.8.3 在不使用 DMA 和溢出检测的情况下进行转换

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）。为此，必须禁止 DMA (DMA = 0) 并且仅在序列结束 (EOCS = 0) 时才将 EOC 位置 1。在此配置中，溢出检测已禁止。

15.9 多重 ADC 模式

在具有两个或更多 ADC 的器件中，可使用双重（具有两个 ADC）和三重（具有三个 ADC）ADC 模式（参见*图 79*）。

在多重 ADC 模式下，通过 ADC1 主器件到 ADC2 和 ADC3 从器件的交替触发或同时触发来启动转换，具体取决于 ADC_CCR 寄存器中的 MULTI[4:0] 位所选的模式。

注： *在多重 ADC 模式下，配置外部事件触发转换时，应用必须设置为仅主器件触发而禁止从器件触发，以防止出现意外触发而启动不需要的从转换。*

可实现以下四种模式：

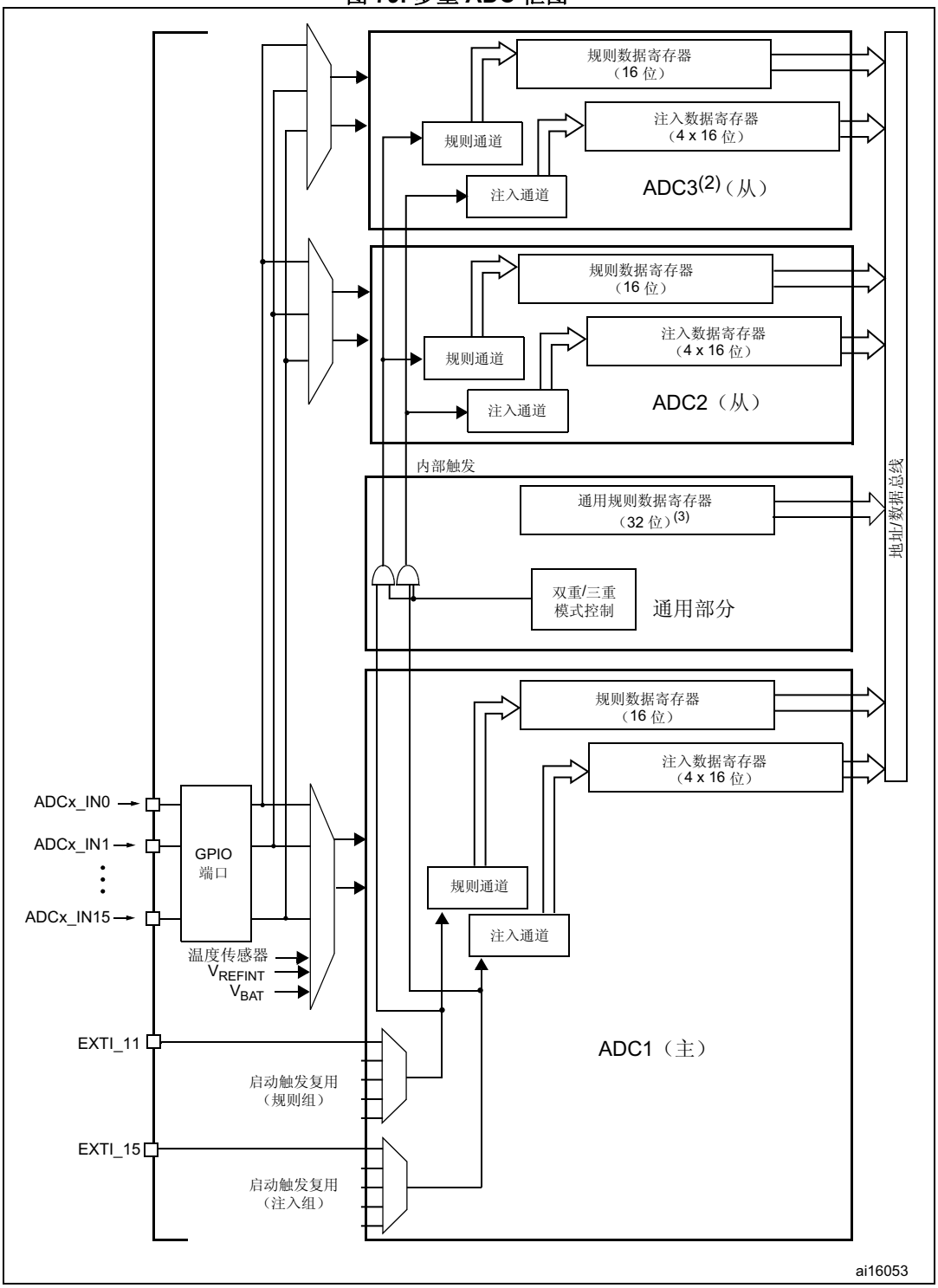
- 注入同步模式
- 常规同步模式
- 交替模式
- 交替触发模式

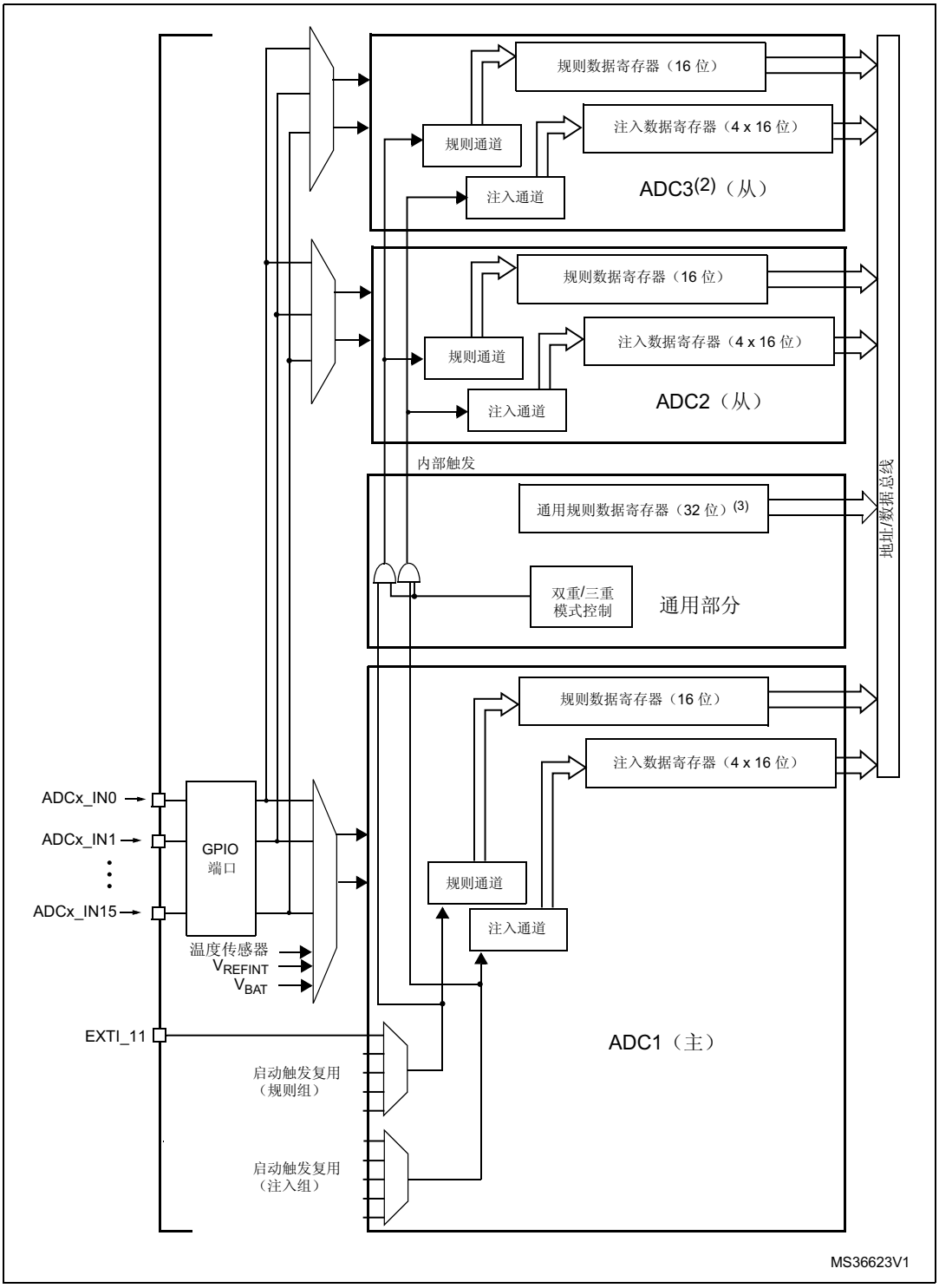
也可按以下方式组合使用上述模式：

- 注入同步模式 + 常规同步模式
- 常规同步模式 + 交替触发模式

注： *在多重 ADC 模式下，可在多模式数据寄存器 (ADC_CDR) 中读取转换的数据。可在多模式状态寄存器 (ADC_CSR) 中读取状态位。*

图 79. 多重 ADC 框图(1)





1. 尽管 ADC2 和 ADC3 上存在外部触发，但它们并未显示在此图中。
2. 在双重 ADC 模式下，不存在 ADC3 从器件部分。
3. 在三重 ADC 模式下，ADC 通用数据寄存器 (ADC_CDR) 包含 ADC1、ADC2 和 ADC3 的常规转换数据。按照所选的存储顺序使用全部 32 个寄存器位。
   在双重 ADC 模式下，ADC 通用数据寄存器 (ADC_CDR) 包含 ADC1 和 ADC2 的常规转换数据。使用全部 32 个寄存器位。

- 多重 ADC 模式下的 DMA 请求：

  在多重 ADC 模式下，可将 DMA 配置为使用三种不同的模式来传输转换的数据。在所有情况下，要使用的 DMA 流均连接到 ADC：

  – **DMA 模式 1**：每发出一个 DMA 请求（一个数据项可用），一个半字代表一个 ADC 转换数据项完成。

    在双重 ADC 模式下，发出第一个请求时传输 ADC1 的数据，发出第二个请求时传输 ADC2 的数据，依次类推。

    在三重 ADC 模式下，发出第一个请求时传输 ADC1 的数据，发出第二个请求时传输 ADC2 的数据，发出第三个请求时传输 ADC3 的数据；重复此序列。因此 DMA 首先传输 ADC1 的数据，随后传输 ADC2 的数据，再传输 ADC3 的数据，依次类推。

    DMA 模式 1 用于三重常规同步模式。

    示例：

    三重常规同步模式：生成 3 个连续的 DMA 请求（每个请求对应一个转换数据项）

    第 1 个请求：ADC_CDR[31:0] = ADC1_DR[15:0]

    第 2 个请求：ADC_CDR[31:0] = ADC2_DR[15:0]

    第 3 个请求：ADC_CDR[31:0] = ADC3_DR[15:0]

    第 4 个请求：ADC_CDR[31:0] = ADC1_DR[15:0]

  – **DMA 模式 2**：每发送一个 DMA 请求（两个数据项可用），就会以字的形式传输表示两个 ADC 转换数据项的两个半字。

    在双重 ADC 模式下，发出第一个请求时会传输 ADC2 和 ADC1 的数据（ADC2 数据占用高位半字，ADC1 数据占用低位半字），依此类推。

    在三重 ADC 模式下，将生成三个 DMA 请求：发出第一个请求时，会传输 ADC2 和 ADC1 的数据（ADC2 数据占用高位半字，ADC1 数据占用低位半字）。发出第二个请求时，会传输 ADC1 和 ADC3 的数据（ADC1 数据占用高位半字，ADC3 数据占用地位半字）。发出第三个请求时，会传输 ADC3 和 ADC2 的数据（ADC3 占用高位半字，ADC2 数据占用地位半字），依此类推。

    DMA 模式 2 用于交替模式和常规同步模式（仅适用于双重 ADC 模式）。

    示例：

    a) 双重交替模式：每当有 2 个数据项可用时，就会生成一个 DMA 请求：

    第 1 个请求：ADC_CDR[31:0] = ADC2_DR[15:0] | ADC1_DR[15:0]

    第 2 个请求：ADC_CDR[31:0] = ADC2_DR[15:0] | ADC1_DR[15:0]

    b) 三重交替模式：每当有 2 个数据项可用时，就会生成一个 DMA 请求

    第 1 个请求：ADC_CDR[31:0] = ADC2_DR[15:0] | ADC1_DR[15:0]

    第 2 个请求：ADC_CDR[31:0] = ADC1_DR[15:0] | ADC3_DR[15:0]

    第 3 个请求：ADC_CDR[31:0] = ADC3_DR[15:0] | ADC2_DR[15:0]

    第 4 个请求：ADC_CDR[31:0] = ADC2_DR[15:0] | ADC1_DR[15:0]

  – **DMA** 模式 3：此模式与 DMA 模式 2 相似。唯一的区别是：在这种模式下，每发送一个 DMA 请求（两个数据项可用），就会以半字的形式传输表示两个 ADC 转换数据项的两个字节。此模式下的数据传输顺序与 DMA 模式 2 相似。

    DMA 模式 3 用于分辨率为 6 位和 8 位时的交替模式。

    示例：

    a) 双重交替模式：每当有 2 个数据项可用时，就会生成一个 DMA 请求

    第一个请求：ADC_CDR[15:0] = ADC2_DR[7:0] | ADC1_DR[7:0]

第二个请求：ADC_CDR[15:0] = ADC2_DR[7:0] | ADC1_DR[7:0]

b) 三重交替模式：每当有 2 个数据项可用时，就会生成一个 DMA 请求

第一个请求：ADC_CDR[15:0] = ADC2_DR[7:0] | ADC1_DR[7:0]

第二个请求：ADC_CDR[15:0] = ADC1_DR[7:0] | ADC3_DR[15:0]

第三个请求：ADC_CDR[15:0] = ADC3_DR[7:0] | ADC2_DR[7:0]

第四个请求：ADC_CDR[15:0] = ADC2_DR[7:0] | ADC1_DR[7:0]

溢出检测：如果在其中一个相关的 ADC（双重和三重模式下的 ADC1 和 ADC2，仅有三重模式时的 ADC3）上检测到溢出，则不再发出 DMA 请求，以确保传输到 RAM 的所有数据都有效。对于与某个 ADC 对应的 EOC 位，有时可能会因为此 ADC 的数据寄存器包含有效数据而保持置 1。

15.9.1 注入同步模式

此模式可转换注入通道组。外部触发源来自 ADC1 的注入组多路复用器（通过 ADC1_CR2 寄存器中的 JEXTSEL[3:0] 位选择）。同时触发可用于 ADC2 和 ADC3。

注： *不要在两个/三个 ADC 上转换同一通道（转换同一通道时，不允许两个/三个 ADC 采样时间重叠）。*

在同步模式下，必须使用同一长度来转换序列，或必须确保触发之间的间隔长于 2 个序列（双重 ADC 模式）/3 个序列（三重 ADC 模式）中的较长时间。否则，当序列较长的 ADC 完成上一次转换时，序列较短的 ADC 可能重新开始转换。

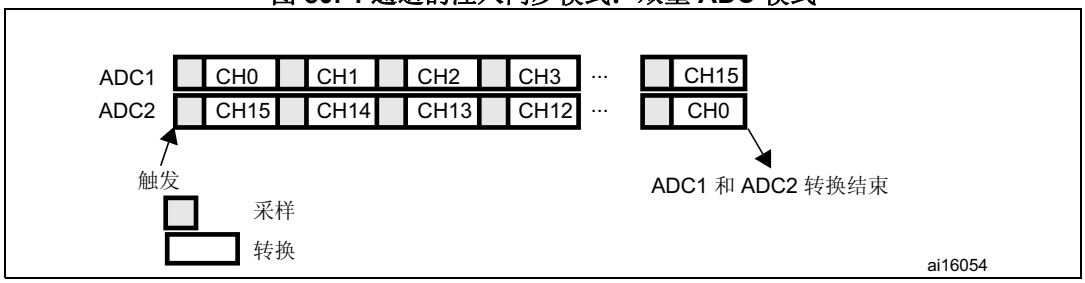
常规转换可在一个或所有 ADC 上执行。这种情况下，它们彼此之间都是独立的，而且会在出现注入事件时中断。它们会在注入转换组结束时恢复转换。

双重 ADC 模式

在 ADC1 或 ADC2 转换事件结束时：

- 转换的数据会存储在各个 ADC 接口的 ADC_JDRx 寄存器中。
- 当 ADC1/ADC2 的注入通道全部完成转换后，会生成一个 JEOC 中断（如果已在两个 ADC 接口中的一个接口上使能）。

图 80. 4 通道的注入同步模式：双重 ADC 模式

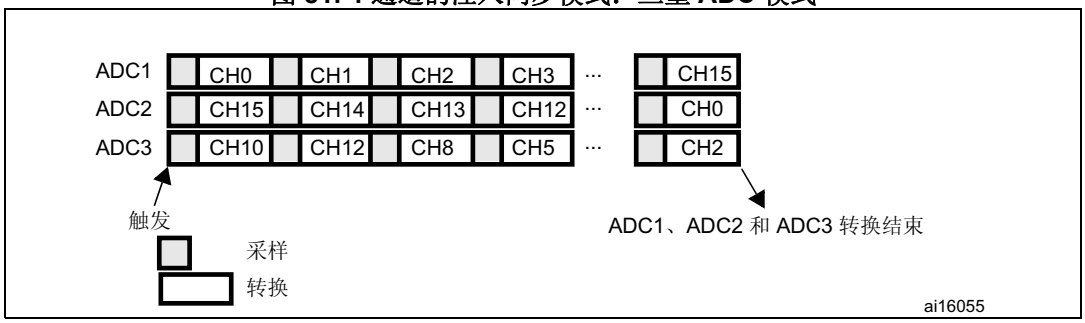


三重 ADC 模式

在 ADC1、ADC2 或 ADC3 转换事件结束时：

- 转换的数据会存储在各个 ADC 接口的 ADC_JDRx 寄存器中。
- 当 ADC1/ADC2/ADC3 的注入通道全部完成转换后，会生成一个 JEOC 中断（如果已在三个 ADC 接口中的一个接口上使能）。

图 81. 4 通道的注入同步模式：三重 ADC 模式



15.9.2 常规同步模式

此模式可用于常规通道组。外部触发源来自 ADC1 的常规组多路复用器（通过 ADC1_CR2 寄存器中的 EXTSEL[3:0] 位进行选择）。同时触发可用于 ADC2 和 ADC3。

注： *不要在两个/三个 ADC 上转换同一通道（转换同一通道时，不允许两个/三个 ADC 采样时间重叠）。*

在常规同步模式下，必须使用同一长度来转换序列，或必须确保触发之间的间隔长于 2 个序列（双重 ADC 模式）/3 个序列（三重 ADC 模式）中的较长转换时间。否则，当序列较长的 ADC 完成上一次转换时，序列较短的 ADC 可能重新开始转换。

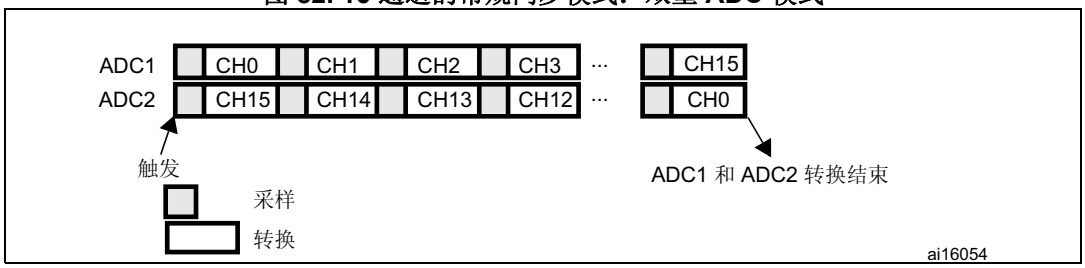
必须禁止注入转换。

双重 ADC 模式

在 ADC1 或 ADC2 转换事件结束时：

- 会生成一个 32 位 DMA 传输请求（如果 ADC_CCR 寄存器中的 DMA[1:0] 位等于 0b10）。此请求会将存储在 ADC_CDR 32 位寄存器高位半字中的 ADC2 转换数据传输到 SRAM，然后将存储在 ADC_CCR 低位半字中的 ADC1 转换数据传输到 SRAM。
- 当 ADC1/ADC2 的常规通道全部完成转换后，会生成一个 EOC 中断（如果已在两个 ADC 接口中的一个接口上使能）。

图 82. 16 通道的常规同步模式：双重 ADC 模式

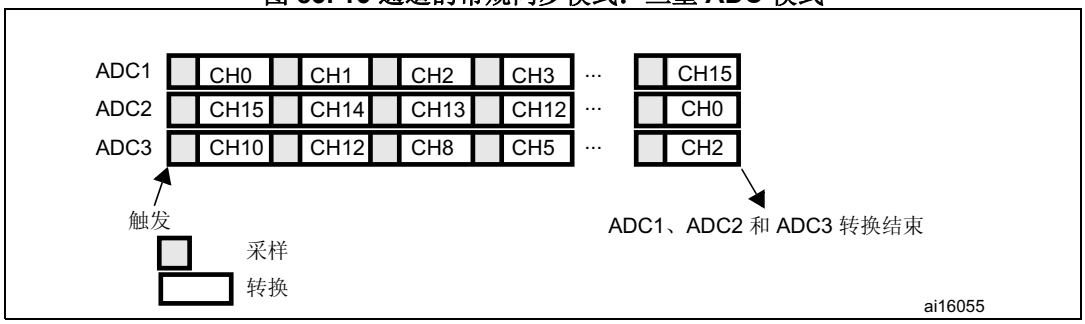


三重 ADC 模式

在 ADC1、ADC2 或 ADC3 转换事件结束时：

- 会生成三个 32 位 DMA 传输请求（如果 ADC_CCR 寄存器中的 DMA[1:0] 位等于 0b01）。之后会发生三次从 ADC_CDR 32 位寄存器到 SRAM 的传输：首先传输 ADC1 转换数据，然后传输 ADC2 转换数据，最后是 ADC3 转换数据。每次出现新的三阶段转换时都会重复这一过程。
- 当 ADC1/ADC2/ADC3 的常规通道全部完成转换后，会生成一个 EOC 中断（如果已在三个 ADC 接口中的一个接口上使能）。

图 83. 16 通道的常规同步模式：三重 ADC 模式



15.9.3 交替模式

此模式只能用于常规组（通常为一个通道）。外部触发源来自 ADC1 的常规通道多路复用器。

双重 ADC 模式

出现外部触发之后：

- ADC1 立即启动
- 经过几个-ADC 时钟周期延迟后 ADC2 启动

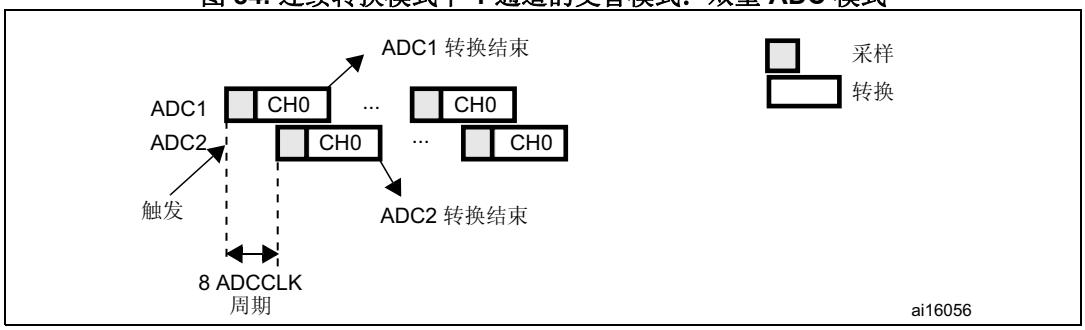
交替模式下 2 个转换之间的最小延迟通过 ADC_CCR 寄存器中的 DELAY 位进行配置。但是，如果某个 ADC 的互补 ADC 仍在对其输入进行采样，则该 ADC 无法启动转换（在给定时间内，只有一个 ADC 能够对输入信号采样）。在这种情况下，延迟时间为采样时间 + 2 个 ADC 时钟周期。例如，如果两个 ADC 的 DELAY = 5 个时钟周期，且采样时间为 15 个时钟周期，则 ADC1 和 ADC2 之间的转换延迟为 17 个时钟周期。

如果 ADC1 和 ADC2 上的 CONT 位均置 1，则这两个 ADC 所选常规通道会连续进行转换。

注： *如果转换序列中断（例如 DMA 传输结束时），则必须首先通过在独立模式下进行配置来将多重 ADC 定序器复位（位 DUAL[4:0] = 00000），然后才可以对交替模式进行编程。*

ADC2 生成一个 EOC 中断之后（如果已通过 EOCIE 位使能），会生成一个 32 位 DMA 传输请求（如果 ADC_CCR 寄存器中的 DMA[1:0] 位等于 0b10）。此请求首先会将存储在 ADC_CDR 32 位寄存器高位半字中的 ADC2 转换数据传输到 SRAM，然后将存储在寄存器低位半字中的 ADC1 转换数据传输到 SRAM。

图 84. 连续转换模式下 1 通道的交替模式：双重 ADC 模式



三重 ADC 模式

出现外部触发之后：

- ADC1 立即启动
- 经过几个 ADC 时钟周期延迟后 ADC2 启动
- 在 ADC2 转换经过几个 ADC 时钟周期的延迟后 ADC3 启动

交替模式下 2 个转换之间的最小延迟通过 ADC_CCR 寄存器中的 DELAY 位进行配置。但是，如果某个 ADC 的互补 ADC 仍在对其输入进行采样，则该 ADC 无法启动转换（在给定时间内，只有一个 ADC 能够对输入信号采样）。在这种情况下，延迟时间为采样时间 + 2 个 ADC 时钟周期。例如，如果这三个 ADC 的 DELAY = 5 个时钟周期，且采样时间为 15 个时钟周期，则 ADC1、ADC2 和 ADC3 之间的转换延迟为 17 个时钟周期。

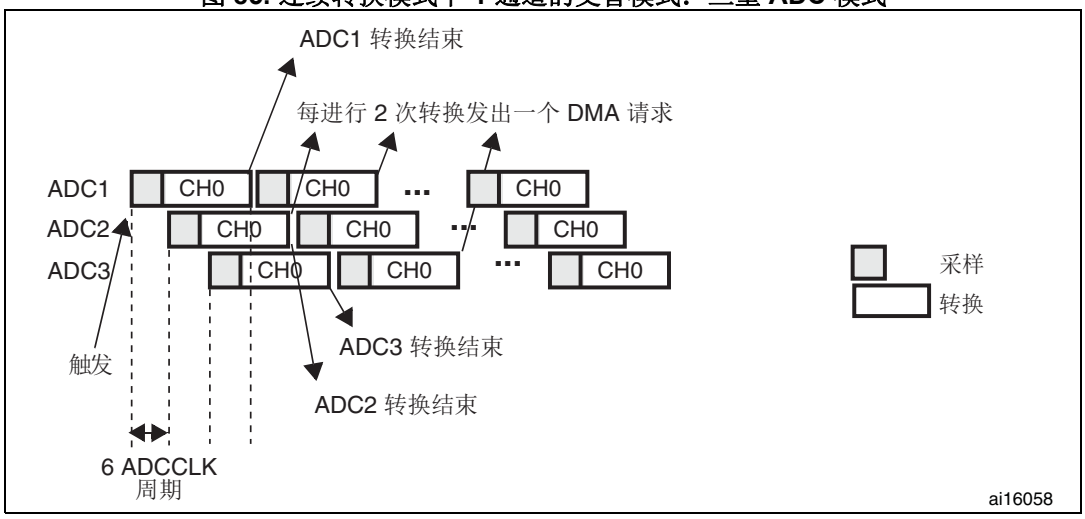
如果 ADC1、ADC2 和 ADC3 上的 CONT 位均置 1，则这些 ADC 所选常规通道会连续进行转换。

注： *如果转换序列中断（例如 DMA 传输结束时），则必须首先通过在独立模式下进行配置来将多重 ADC 定序器复位（位 DUAL[4:0] = 00000），然后才可以对交替模式进行编程。*

在此模式下，每当出现 2 个可用数据项时，就会生成一个 DMA 传输请求（如果 ADC_CCR 寄存器中的 DMA[1:0] 位等于 0b10）。此请求首先会将存储在 ADC_CDR 32 位寄存器低位半字中的第一批转换数据传输到 SRAM，然后将存储在 ADC_CDR 高位半字中的第二批转换数据传输到 SRAM。具体顺序如下：

- 第 1 个请求：ADC_CDR[31:0] = ADC2_DR[15:0] | ADC1_DR[15:0]
- 第 2 个请求：ADC_CDR[31:0] = ADC1_DR[15:0] | ADC3_DR[15:0]
- 第 3 个请求：ADC_CDR[31:0] = ADC3_DR[15:0] | ADC2_DR[15:0]
- 第 4 个请求：ADC_CDR[31:0] = ADC2_DR[15:0] | ADC1_DR[15:0]，...

图 85. 连续转换模式下 1 通道的交替模式：三重 ADC 模式



15.9.4 交替触发模式

此模式只能用于注入组。外部触发源来自 ADC1 的注入组多路复用器。

注： *常规转换可在一个或所有 ADC 上使能。在这种情况下，常规转换彼此之间是独立的。当 ADC 必须执行注入转换时，会中断常规转换。它会在注入转换完成后恢复。*

如果转换序列中断（例如DMA 传输结束时），则必须首先通过在独立模式下进行配置来将多重 ADC 定序器复位（位 DUAL[4:0] = 00000），然后才可以对交替模式进行编程。

两次触发事件之间的时间间隔必须大于或等于 1 个 ADC 时钟周期。同一个 ADC 上可启动转换的两个触发事件之间的最小时间间隔与单个 ADC 模式下相同。

双重 ADC 模式

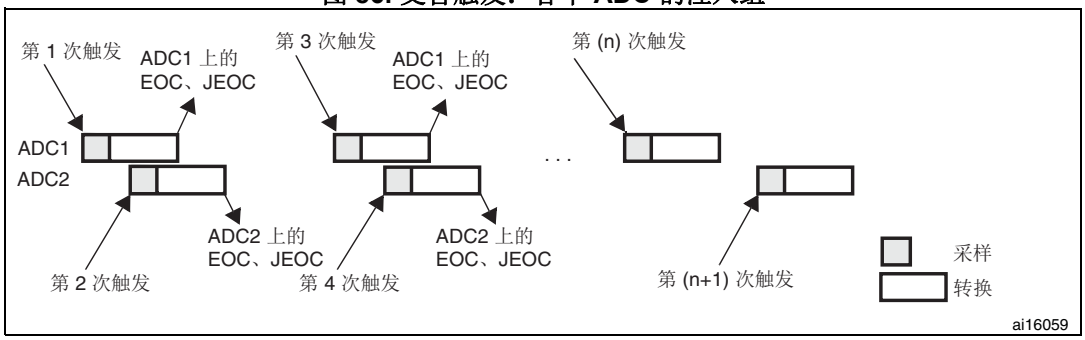
- 发生第一次触发时，将转换 ADC1 中注入组的所有通道
- 发生第二次触发时，将转换 ADC2 中注入组的所有通道
- 以此类推

当组中的所有注入 ADC1 通道都转换完成后，会生成一个 JEOC 中断（如果已使能）。

当组中的所有注入 ADC2 通道都转换完成后，会生成一个 JEOC 中断（如果已使能）。

如果在组中的所有注入通道都完成转换后出现另一个外部触发，则可通过转换组中的注入 ADC1 通道来重新启动交替触发过程。

图 86. 交替触发：各个 ADC 的注入组



如果使能 ADC1 和 ADC2 的注入不连续采样模式：

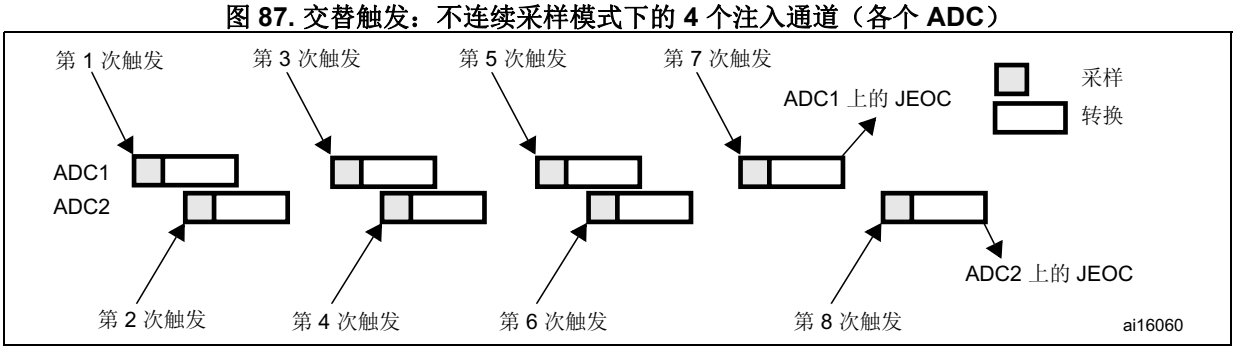
- 发生第一次触发时，将转换第一个注入 ADC1 通道
- 发生第二次触发时，将转换第一个注入 ADC2 通道
- 以此类推

当组中的所有注入 ADC1 通道都转换完成后，会生成一个 JEOC 中断（如果已使能）。

当组中的所有注入 ADC2 通道都转换完成后，会生成一个 JEOC 中断（如果已使能）。

如果注入组中的所有通道都完成转换后出现另一个外部触发，则会重新启动交替触发过程。

图 87. 交替触发：不连续采样模式下的 4 个注入通道（各个 ADC）



三重 ADC 模式

- 发生第一次触发时，将转换 ADC1 中注入组的所有通道。
- 发生第二次触发时，将转换 ADC2 中注入组的所有通道。
- 发生第三次触发时，将转换 ADC3 中注入组的所有通道。
- 以此类推。

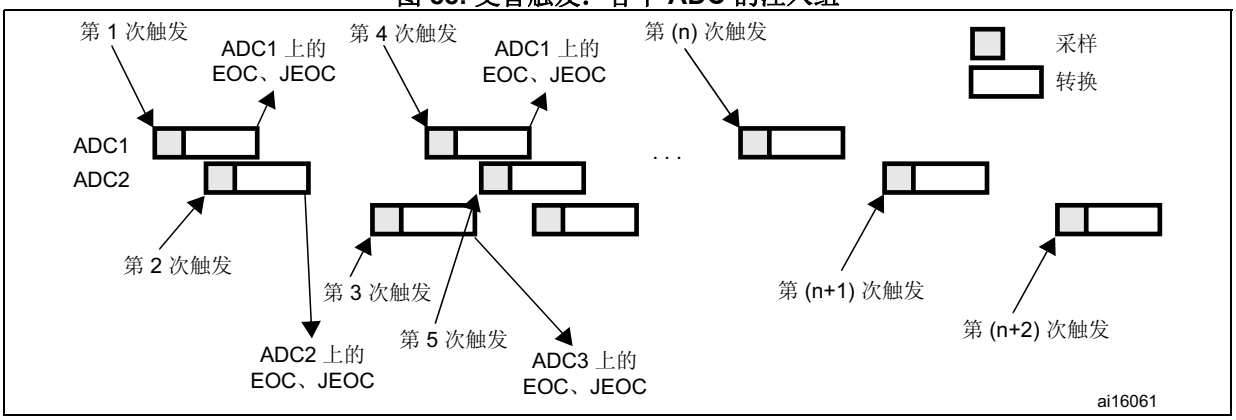
当组中的所有注入 ADC1 通道都转换完成后，会生成一个 JEOC 中断（如果已使能）。

当组中的所有注入 ADC2 通道都转换完成后，会生成一个 JEOC 中断（如果已使能）。

当组中的所有注入 ADC3 通道都转换完成后，会生成一个 JEOC 中断（如果已使能）。

如果在组中的所有注入通道都完成转换后出现另一个外部触发，则可通过转换组中的注入 ADC1 通道来重新启动交替触发过程。

图 88. 交替触发：各个 ADC 的注入组



15.9.5 混合型常规/注入同步模式

可以中断常规组的同步转换，然后开始注入组的同步转换。

注： *在混合型常规/注入同步模式下，必须使用同一长度来转换序列，或必须确保触发之间的间隔长于 2 个序列（双重 ADC 模式）/3 个序列（三重 ADC 模式）中的较长转换时间。否则，当序列较长的 ADC 完成上一次转换时，序列较短的 ADC 可能重新开始转换。*

15.9.6 常规同步 + 交替触发组合模式

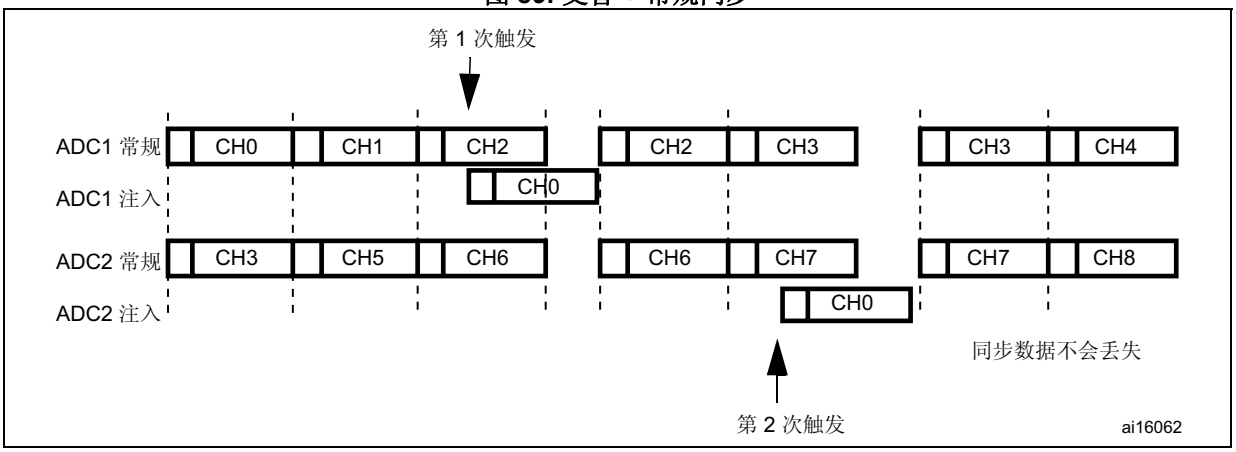
可以中断常规组的同步转换，然后开始注入组的交替触发转换。*图 89* 说明了交替触发模式中断同步常规转换的行为。

注入事件后立即开始注入交替转换。当常规转换处于运行状态时，为确保在注入转换后实现同步，所有的（主/从）ADC 常规转换均将停止，并会在注入转换结束时得以恢复运行。

注： *在常规同步 + 交替触发组合模式下，必须使用同一长度来转换序列，或必须确保触发之间的间隔长于 2 个序列（双重 ADC 模式）/3 个序列（三重 ADC 模式）的长转换时间。否则，当序列较长的 ADC 完成上一次转换时，序列较短的 ADC 可能重新开始转换。*

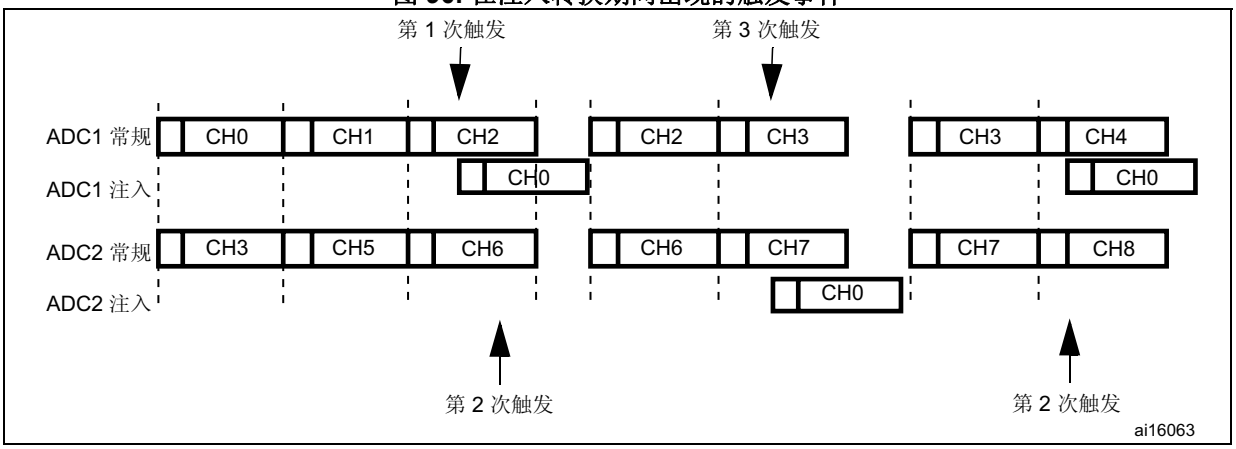
如果转换序列中断（例如 DMA 传输结束时），则必须首先通过在独立模式下进行配置来将多重 ADC 定序器复位（位 DUAL[4:0] = 00000），然后才可以对交替模式进行编程。

图 89. 交替 + 常规同步



在已导致常规转换中断的注入转换期间出现的触发将被忽略。*图 90* 说明了这种情况下的行为（第 2 次触发被忽略）。

图 90. 在注入转换期间出现的触发事件



15.10 温度传感器

温度传感器可用于测量器件的环境温度 (T_A)。

- 对于 STM32F75xxx 和 STM32F74xxx 器件，温度传感器内部连接到与 VBAT 共用的输入通道 ADC1_IN18：ADC1_IN18 用于将传感器输出电压或 VBAT 转换为数字值。一次只能选择一个转换（温度传感器或 VBAT）。同时设置了温度传感器和 VBAT 转换时，将只进行 VBAT 转换。

图 91 显示了温度传感器框图。

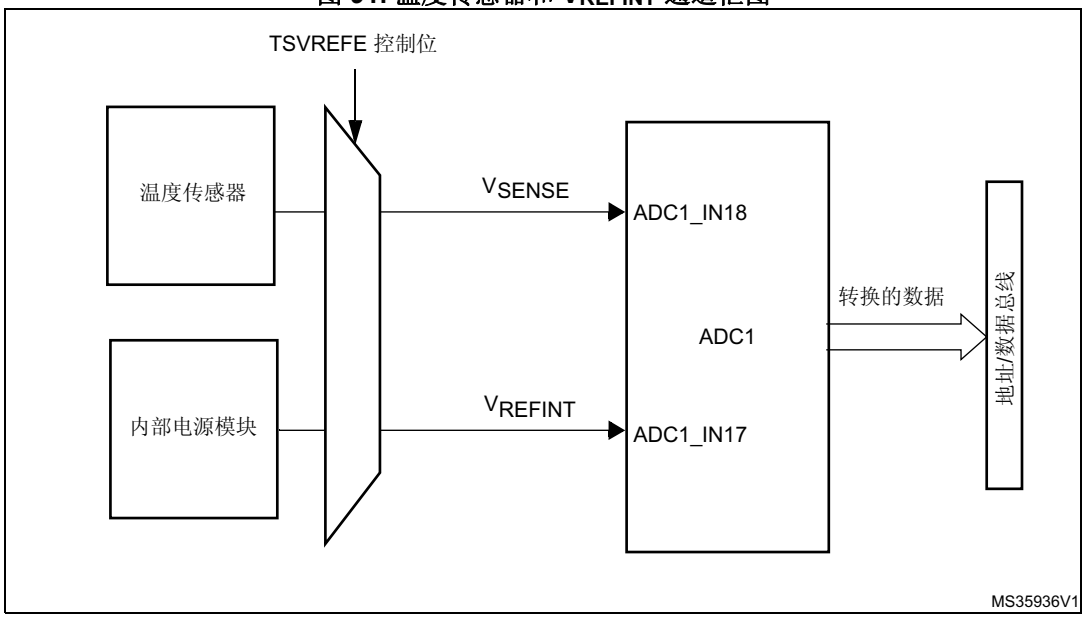
不使用时可将传感器置于掉电模式。

注： *必须将 TSVREFE 位置 1 才能同时对两个内部通道进行转换：ADC1_IN18（温度传感器）和 ADC1_IN17 (VREFINT)。*

主要特性

- 支持的温度范围：-40 °C 到 125 °C
- 精度：±1.5 °C

图 91. 温度传感器和 V_{REFINT} 通道框图



1. V_{SENSE} 输入到 ADC1_IN18。

读取温度

要使用传感器，请执行以下操作：

1. 选择 ADC1_IN18 输入通道。
2. 选择一个采样时间，该采样时间要大于数据手册中所指定的最低采样时间。
3. 在 ADC_CCR 寄存器中将 TSVREFE 位置 1，以便将温度传感器从掉电模式中唤醒。
4. 通过将 SWSTART 位置 1（或通过外部触发）开始 ADC 转换
5. 读取 ADC 数据寄存器中生成的 V_{SENSE} 数据
6. 使用以下公式计算温度：

   温度（单位为 °C）= {(V_{SENSE} – V_{25}) / Avg_Slope} + 25

   其中：

   – V_{25} = 25 °C 时的 V_{SENSE} 值
   – Avg_Slope = 温度与 V_{SENSE} 曲线的平均斜率（以 mV/°C 或 µV/°C 表示）

   有关 V_{25} 和 Avg_Slope 实际值的相关信息，请参见数据手册中的电气特性一节。

注： *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，为尽可能减少延迟间，应同时将 ADON 和 TSVREFE 位置 1。*

温度传感器的输出电压随温度线性变化。由于工艺不同，该线性函数的偏移量取决于各个芯片（芯片之间的温度变化可达 45 °C）。

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情况。如果需要读取精确温度，则应使用外部温度传感器。

15.11 电池充电监视

ADC_CCR 寄存器中的 VBATE 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} 引脚需要内部连接到桥接分配器，以确保 ADC 正确运行。

设置 VBATE 后，桥接器会自动使能以进行以下连接：

- 将 VBAT/4 连接到 ADC1_IN18 输入通道

注： *VBAT 和温度传感器连接到同一 ADC 内部通道 (ADC1_IN18)。一次只能选择一个转换（温度传感器或 VBAT）。同时使能两个转换时，将只进行 VBAT 转换。*

15.12 ADC 中断

当模拟看门狗状态位和溢出状态位分别置 1 时，常规组和注入组在转换结束时可能会产生中断。可以使用单独的中断使能位以提高灵活性。

ADC_SR 寄存器中存在另外两个标志，但这两个标志不存在中断相关性：

- JSTRT（开始转换注入组的通道）
- STRT（开始转换常规组的通道）

表 95. ADC 中断

中断事件	事件标志	使能控制位
结束常规组的转换	EOC	EOCIE
结束注入组的转换	JEOC	JEOCIE
模拟看门狗状态位置 1	AWD	AWDIE
上溢	OVR	OVRIE

15.13 ADC 寄存器

有关寄存器说明中使用的缩写，请参见*第 58 页上的第 1.1 节*。

必须在字级别（32 位）对外设寄存器执行写入操作。而读访问可支持字节（8 位）、半字（16 位）或字（32 位）。

15.13.1 ADC 状态寄存器 (ADC_SR)

ADC 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.	Res.	Res.	Res.	Res.	Res.	OVR	STRT	JSTRT	JEOC	EOC	AWD
										rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

位 31:6 保留，必须保持复位值。

位 5 **OVR**：溢出 (Overrun)

数据丢失时，硬件将该位置 1（在单一模式或双重/三重模式下）。但需要通过软件清零。溢出检测仅在 DMA = 1 或 EOCS = 1 时使能。

0：未发生溢出

1：发生溢出

位 4 **STRT**：常规通道开始标志 (Regular channel start flag)

常规通道转换开始时，硬件将该位置 1。但需要通过软件清零。

0：未开始常规通道转换

1：已开始常规通道转换

位 3 **JSTRT**：注入通道开始标志 (Injected channel start flag)

注入组转换开始时，硬件将该位置 1。但需要通过软件清零。

0：未开始注入组转换

1：已开始注入组转换

位 2 **JEOC**：注入通道转换结束 (Injected channel end of conversion)

组内所有注入通道转换结束时，硬件将该位置 1。但需要通过软件清零。

0：转换未完成

1：转换已完成

位 1 **EOC**：常规通道转换结束 (Regular channel end of conversion)

常规组通道转换结束后，硬件将该位置 1。通过软件或通过读取 ADC_DR 寄存器将该位清零。

0：转换未完成 (EOCS=0) 或转换序列未完成 (EOCS=1)

1：转换已完成 (EOCS=0) 或转换序列已完成 (EOCS=1)

位 0 **AWD**：模拟看门狗标志 (Analog watchdog flag)

当转换电压超过在 ADC_LTR 和 ADC_HTR 寄存器中编程的值时，硬件将该位置 1。但需要通过软件清零。

0：未发生模拟看门狗事件

1：发生模拟看门狗事件

15.13.2 ADC 控制寄存器 1 (ADC_CR1)

ADC control register 1

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	OVRIE	RES		AWDEN	JAWDEN	Res.	Res.	Res.	Res.	Res.	Res.
					rw	rw	rw	rw	rw						

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

位 31:27 保留，必须保持复位值。

位 26 **OVRIE**：溢出中断使能 (overrun interrupt enable)

通过软件将该位置 1 和清零可使能/禁止溢出中断。

0：禁止溢出中断

1：使能溢出中断。OVR 位置 1 时产生中断。

位 25:24 **RES[1:0]**：分辨率 (Resolution)
通过软件写入这些位可选择转换的分辨率。
00：12 位（15 ADCCLK 周期）
01：10 位（13 ADCCLK 周期）
10：8 位（11 ADCCLK 周期）
11：6 位（9 ADCCLK 周期）

位 23 **AWDEN**：常规通道上的模拟看门狗使能 (Analog watchdog enable on regular channels)
此位由软件置 1 和清零。
0：在常规通道上禁止模拟看门狗
1：在常规通道上使能模拟看门狗

位 22 **JAWDEN**：注入通道上的模拟看门狗使能 (Analog watchdog enable on injected channels)
此位由软件置 1 和清零。
0：在注入通道上禁止模拟看门狗
1：在注入通道上使能模拟看门狗

位 21:16 保留，必须保持复位值。

位 15:13 **DISCNUM[2:0]**：不连续采样模式通道计数 (Discontinuous mode channel count)
软件将写入这些位，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

位 12 **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

位 11 **DISCEN**：常规通道的不连续采样模式 (Discontinuous mode on regular channels)
通过软件将该位置 1 和清零可使能/禁止常规通道的不连续采样模式。
0：禁止常规通道的不连续采样模式
1：使能常规通道的不连续采样模式

位 10 **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

位 9 **AWDSGL**：在扫描模式下使能单一通道上的看门狗 (Enable the watchdog on a single channel in scan mode)
通过软件将该位置 1 和清零可分别使能/禁止通过 AWDCH[4:0] 位确定的通道上的模拟看门狗。
0：在所有通道上使能模拟看门狗
1：在单一通道上使能模拟看门狗

位 8 **SCAN**：扫描模式 (Scan mode)
通过软件将该位置 1 和清零可使能/禁止扫描模式。在扫描模式下，转换通过 ADC_SQRx 或 ADC_JSQRx 寄存器选择的输入。
0：禁止扫描模式
1：使能扫描模式
注： EOCIE 位置 1 时将生成 EOC 中断：
– *如果 EOCS 位清零，在每个常规组序列转换结束时*
– *如果 EOCS 位置 1，在每个常规通道转换结束时*
注： JEOCIE 位置 1 时，JEOC 中断仅在最后一个通道转换结束时生成。

位 7 **JEOCIE**：注入通道的中断使能 (Interrupt enable for injected channels)
通过软件将该位置 1 和清零可使能/禁止注入通道的转换结束中断。
0：禁止 JEOC 中断
1：使能 JEOC 中断 JEOC 位置 1 时产生中断。

位 6 **AWDIE**：模拟看门狗中断使能 (Analog watchdog interrupt enable)
通过软件将该位置 1 和清零可使能/禁止模拟看门狗中断。
0：禁止模拟看门狗中断
1：使能模拟看门狗中断

位 5 **EOCIE**：EOC 中断使能 (Interrupt enable for EOC)
通过软件将该位置 1 和清零可使能/禁止转换结束中断。
0：禁止 EOC 中断
1：使能 EOC 中断 EOC 位置 1 时产生中断。

位 4:0 **AWDCH[4:0]**：模拟看门狗通道选择位 (Analog watchdog channel select bits)
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。
注： 00000：ADC 模拟输入通道 0
00001：ADC 模拟输入通道 1
...
01111：ADC 模拟输入通道 15
10000：ADC 模拟输入通道 16
10001：ADC 模拟输入通道 17
10010：ADC 模拟输入通道 18
保留其它值

15.13.3 ADC 控制寄存器 2 (ADC_CR2)

ADC control register 2

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.	SWSTART	EXTEN		EXTSEL[3:0]				Res.	JSWSTART	JEXTEN		JEXTSEL[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
Res.	Res.	Res.	Res.	ALIGN	EOCS	DDS	DMA	Res.	Res.	Res.	Res.	Res.	Res.	CONT	ADON
				rw	rw	rw	rw							rw	rw

位 31 保留，必须保持复位值。

位 30 **SWSTART**：开始转换常规通道 (Start conversion of regular channels)
通过软件将该位置 1 可开始转换，而硬件会在转换开始后将该位清零。
0：复位状态
1：开始转换常规通道
注： 该位只能在 ADON = 1 时置 1，否则不会启动转换。

位 29:28 **EXTEN**：常规通道的外部触发使能 (External trigger enable 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：禁止触发检测
01：上升沿上的触发检测
10：下降沿上的触发检测
11：上升沿和下降沿上的触发检测

位 27:24 **EXTSEL[3:0]**：为常规组选择外部事件 (External event select for regular group)
这些位可选择用于触发常规组转换的外部事件。
0000：定时器 1 CC1 事件
0001：定时器 1 CC2 事件
0010：定时器 1 CC3 事件
0011：定时器 2 CC2 事件
0100：定时器 5 TRGO 事件
0101：定时器 4 CC4 事件
0110：定时器 3 CC4 事件
0111：定时器 8 TRGO 事件
1000：定时器 8 TRGO(2) 事件
1001：定时器 1 TRGO 事件
1010：定时器 1 TRGO(2) 事件
1011：定时器 2 TRGO 事件
1100：定时器 4 TRGO 事件
1101：定时器 6 TRGO 事件
1110：保留
1111：EXTI 线 11

位 23 保留，必须保持复位值。

位 22 **JSWSTART**：开始转换注入通道 (Start conversion of injected channels)
转换开始后，软件将该位置 1，而硬件将该位清零。
0：复位状态
1：开始转换注入通道
注：　该位只能在 ADON = 1 时置 1，否则不会启动转换。

位 21:20 **JEXTEN**：注入通道的外部触发使能 (External trigger enable 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：禁止触发检测
01：上升沿上的触发检测
10：下降沿上的触发检测
11：上升沿和下降沿上的触发检测

位 19:16 **JEXTSEL[3:0]**：为注入组选择外部事件 (External event select for injected group)
这些位可选择用于触发注入组转换的外部事件。
0000：定时器 1 TRGO 事件
0001：定时器 1 CC4 事件
0010：定时器 2 TRGO 事件
0011：定时器 2 CC1 事件
0100：定时器 3 CC4 事件
0101：定时器 4 TRGO 事件
0110：保留
0111：定时器 8 CC4 事件
1000：定时器 1 TRGO(2) 事件
1001：定时器 8 TRGO 事件
1010：定时器 8 TRGO(2) 事件
1011：定时器 3 CC3 事件
1100：定时器 5 TRGO 事件
1101：定时器 3 CC1 事件
1110：定时器 6 TRGO 事件
1111：保留

位 15:12 保留，必须保持复位值。

位 11 **ALIGN**：数据对齐 (Data alignment)
此位由软件置 1 和清零。请参见*图 76*和*图 77*。
0：右对齐
1：左对齐

位 10 **EOCS**：结束转换选择 (End of conversion selection)
此位由软件置 1 和清零。
0：在每个常规转换序列结束时将 EOC 位置 1。溢出检测仅在 DMA=1 时使能。
1：在每个常规转换结束时将 EOC 位置 1。使能溢出检测。

位 9 **DDS**：DMA 禁止选择（对于单一 ADC 模式）(DMA disable selection (for single ADC mode))
此位由软件置 1 和清零。
0：最后一次传输后不发出新的 DMA 请求（在 DMA 控制器中进行配置）
1：只要发生数据转换且 DMA = 1，便会发出 DAM 请求

位 8 **DMA**：直接存储器访问模式（对于单一 ADC 模式）(Direct memory access mode (for single ADC mode))
此位由软件置 1 和清零。有关详细信息，请参见 DMA 控制器一章。
0：禁止 DMA 模式
1：使能 DMA 模式

位 7:2 保留，必须保持复位值。

位 1 **CONT**：连续转换 (Continuous conversion)
此位由软件置 1 和清零。该位置 1 时，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

位 0 **ADON**：A/D 转换器开启 / 关闭 (A/D Converter ON / OFF)
此位由软件置 1 和清零。
注： 0：禁止 ADC 转换并转至掉电模式
1：使能 ADC

15.13.4 ADC 采样时间寄存器 1 (ADC_SMPR1)

ADC sample time register 1

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:1]	
					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 31:27 保留，必须保持复位值。

位 26:0 **SMPx[2:0]**：通道 x 采样时间选择 (Channel x sampling time selection)

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。

注： 000：3 个周期
001：15 个周期
010：28 个周期
011：56 个周期
100：84 个周期
101：112 个周期
110：144 个周期
111：480 个周期

15.13.5 ADC 采样时间寄存器 2 (ADC_SMPR2)

ADC sample time 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.	SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:1]	
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP5_0	SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

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)

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。

注： 000：3 个周期
001：15 个周期
010：28 个周期
011：56 个周期
100：84 个周期
101：112 个周期
110：144 个周期
111：480 个周期

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

ADC injected channel data offset register x

偏移地址：0x14-0x20

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.	JOFFSETx[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 **JOFFSETx[11:0]**：注入通道 x 的数据偏移 (Data offset for injected channel x)
通过软件写入这些位可定义在转换注入通道时从原始转换数据中减去的偏移量。可从 ADC_JDRx 寄存器中读取转换结果。

15.13.7 ADC 看门狗高阈值寄存器 (ADC_HTR)

ADC watchdog higher threshold register

偏移地址：0x24

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.	HT[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 **HT[11:0]**：模拟看门狗高阈值 (Analog watchdog higher threshold)
通过软件写入这些位可为模拟看门狗定义高阈值。

15.13.8 ADC 看门狗低阈值寄存器 (ADC_LTR)

ADC watchdog lower threshold 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.	LT[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 **LT[11:0]**：模拟看门狗低阈值 (Analog watchdog lower threshold)
通过软件写入这些位可为模拟看门狗定义低阈值。

15.13.9 ADC 常规序列寄存器 1 (ADC_SQR1)

ADC regular sequence register 1

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.	L[3:0]				SQ16[4:1]			
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ16_0	SQ15[4:0]					SQ14[4:0]					SQ13[4:0]				
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 23:20 **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

位 19:15 **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..18) 分配为转换序列中的第十六次转换。

位 14:10 **SQ15[4:0]**：常规序列中的第十五次转换 (15th conversion in regular sequence)

位 9:5 **SQ14[4:0]**：常规序列中的第十四次转换 (14th conversion in regular sequence)

位 4:0 **SQ13[4:0]**：常规序列中的第十三次转换 (13th conversion in regular sequence)

15.13.10 ADC 常规序列寄存器 2 (ADC_SQR2)

ADC regular sequence register 2

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.	SQ12[4:0]					SQ11[4:0]					SQ10[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
SQ10_0	SQ9[4:0]					SQ8[4:0]					SQ7[4:0]				
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 29:26 **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..18) 分配为序列中的第十二次转换。

位 24:20 **SQ11[4:0]**：常规序列中的第十一次转换 (11th conversion in regular sequence)

位 19:15 **SQ10[4:0]**：常规序列中的第十次转换 (10th conversion in regular sequence)

位 14:10 **SQ9[4:0]**：常规序列中的第九次转换 (9th conversion in regular sequence)

位 9:5 **SQ8[4:0]**：常规序列中的第八次转换 (8th conversion in regular sequence)

位 4:0 **SQ7[4:0]**：常规序列中的第七次转换 (7th conversion in regular sequence)

15.13.11 ADC 常规序列寄存器 3 (ADC_SQR3)

ADC regular sequence register 3

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.	SQ6[4:0]					SQ5[4:0]					SQ4[4:1]			
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ4_0	SQ3[4:0]					SQ2[4:0]					SQ1[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:30 保留，必须保持复位值。

位 29:25 **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..18) 分配为序列中的第六次转换。

位 24:20 **SQ5[4:0]**：常规序列中的第五次转换 (5th conversion in regular sequence)

位 19:15 **SQ4[4:0]**：常规序列中的第四次转换 (4th conversion in regular sequence)

位 14:10 **SQ3[4:0]**：常规序列中的第三次转换 (3th conversion in regular sequence)

位 9:5 **SQ2[4:0]**：常规序列中的第二次转换 (2th conversion in regular sequence)

位 4:0 **SQ1[4:0]**：常规序列中的第一次转换 (1th conversion in regular sequence)

15.13.12 ADC 注入序列寄存器 (ADC_JSQR)

ADC injected sequence register

偏移地址：0x38

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JL[1:0]		JSQ4[4:1]			
										rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JSQ4[0]	JSQ3[4:0]					JSQ2[4:0]					JSQ1[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:22 保留，必须保持复位值。

位 21:20 **JL[1:0]**：注入序列长度 (Injected 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

位 19:15 **JSQ4[4:0]**：注入序列中的第四次转换 (4th conversion in injected sequence)（当 JL[1:0] = 3 时，请参见下方的注释）

通过软件写入这些位，并将通道编号 (0..18) 分配为序列中的第四次转换。

位 14:10 **JSQ3[4:0]**：注入序列中的第三次转换 (3rd conversion in injected sequence)（当 JL[1:0] = 3 时，请参见下方的注释）

位 9:5 **JSQ2[4:0]**：注入序列中的第二次转换 (2nd conversion in injected sequence)（当 JL[1:0] = 3 时，请参见下方的注释）

位 4:0 **JSQ1[4:0]**：注入序列中的第一次转换 (1st conversion in injected sequence)（当 JL[1:0] = 3 时，请参见下方的注释）

注： *当 JL[1:0]=3（定序器中有 4 次注入转换）时，ADC 转换通道的顺序为：JSQ1[4:0]、JSQ2[4:0]、JSQ3[4:0] 和 JSQ4[4:0]。*

当 JL=2（定序器中有 3 次注入转换）时，ADC 转换通道的顺序为：JSQ2[4:0]、JSQ3[4:0] 和 JSQ4[4:0]。

当 JL = 1（定序器中有 2 次注入转换）时，ADC 转换通道的顺序为：先是 JSQ3[4:0]，而后是 JSQ4[4:0]。

当 JL = 0（定序器中有 1 次注入转换）时，ADC 将仅转换 JSQ4[4:0] 通道。

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

ADC injected data register x

偏移地址：0x3C - 0x48

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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 X 的转换结果。数据有左对齐和右对齐两种方式，如*图 76* 和*图 77* 所示。

15.13.14 ADC 常规数据寄存器 (ADC_DR)

ADC regular data 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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

位 31:16 保留，必须保持复位值。

位 15:0 **DATA[15:0]**：常规数据 (Regular data)

这些位为只读。它们包括来自常规通道的转换结果。数据有左对齐和右对齐两种方式，如*图 76* 和*图 77* 所示。

15.13.15 ADC 通用状态寄存器 (ADC_CSR)

ADC Common status register

偏移地址：0x00（该偏移地址与 ADC1 基地址 + 0x300 相关）

复位值：0x0000 0000

该寄存器可提供不同 ADC 的状态位图像。但是，它为只读形式且不允许将不同的状态位清零。必须在对应的 ADC_SR 寄存器中将其写为 0，才能将各个状态位清零。

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.	OVR3	STRT3	JSTRT3	JEOC 3	EOC3	AWD3
										ADC3					
										r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	OVR2	STRT2	JSTRT 2	JEOC2	EOC2	AWD2	Res.	Res.	OVR1	STRT1	JSTRT1	JEOC 1	EOC1	AWD1
		ADC2								ADC1					
		r	r	r	r	r	r			r	r	r	r	r	r

位 31:22 保留，必须保持复位值。

位 21 **OVR3**：ADC3 的溢出标志 (Overrun flag of ADC3)
该位是 ADC3_SR 寄存器中 OVR 位的副本。

位 20 **STRT3**：ADC3 的常规通道开始标志 (Regular channel Start flag of ADC3)
该位是 ADC3_SR 寄存器中 STRT 位的副本。

位 19 **JSTRT3**：ADC3 的注入通道开始标志 (Injected channel Start flag of ADC3)
该位是 ADC3_SR 寄存器中 JSTRT 位的副本。

位 18 **JEOC3**：ADC3 的注入通道转换结束 (Injected channel end of conversion of ADC3)
该位是 ADC3_SR 寄存器中 JEOC 位的副本。

位 17 **EOC3**：ADC3 的转换结束 (End of conversion of ADC3)
该位是 ADC3_SR 寄存器中 EOC 位的副本。

位 16 **AWD3**：ADC3 的模拟看门狗标志 (Analog watchdog flag of ADC3)
该位是 ADC3_SR 寄存器中 AWD 位的副本。

位 15:14 保留，必须保持复位值。

位 13 **OVR2**：ADC2 的溢出标志 (Overrun flag of ADC2)
该位是 ADC2_SR 寄存器中 OVR 位的副本。

位 12 **STRT2**：ADC2 的常规通道开始标志 (Regular channel Start flag of ADC2)
该位是 ADC2_SR 寄存器中 STRT 位的副本。

位 11 **JSTRT2**：ADC2 的注入通道开始标志 (Injected channel Start flag of ADC2)
该位是 ADC2_SR 寄存器中 JSTRT 位的副本。

位 10 **JEOC2**：ADC2 的注入通道转换结束 (Injected channel end of conversion of ADC2)
该位是 ADC2_SR 寄存器中 JEOC 位的副本。

位 9 **EOC2**：ADC2 转换结束 (End of conversion of ADC2)
该位是 ADC2_SR 寄存器中 EOC 位的副本。

位 8 **AWD2**：ADC2 的模拟看门狗标志 (Analog watchdog flag of ADC2)
该位是 ADC2_SR 寄存器中 AWD 位的副本。

位 7:6 保留，必须保持复位值。

位 5 **OVR1**：ADC1 的溢出标志 (Overrun flag of ADC1)
该位是 ADC1_SR 寄存器中 OVR 位的副本。

位 4 **STRT1**：ADC1 的常规通道开始标志 (Regular channel Start flag of ADC1)
该位是 ADC1_SR 寄存器中 STRT 位的副本。

位 3 **JSTRT1**：ADC1 的注入通道开始标志 (Injected channel Start flag of ADC1)
该位是 ADC1_SR 寄存器中 JSTRT 位的副本。

位 2 **JEOC1**：ADC1 的注入通道转换结束 (Injected channel end of conversion of ADC1)
该位是 ADC1_SR 寄存器中 JEOC 位的副本。

位 1 **EOC1**：ADC1 的转换结束 (End of conversion of ADC1)
该位是 ADC1_SR 寄存器中 EOC 位的副本。

位 0 **AWD1**：ADC1 的模拟看门狗标志 (Analog watchdog flag of ADC1)
该位是 ADC1_SR 寄存器中 AWD 位的副本。

15.13.16 ADC 通用控制寄存器 (ADC_CCR)

ADC common control register

偏移地址：0x04（该偏移地址与 ADC1 基地址 + 0x300 相关）

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA[1:0]		DDS	Res.	DELAY[3:0]				Res.	Res.	Res.	MULTI[4:0]				
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 **TSVREFE**：温度传感器和 V_{REFINT} 使能 (Temperature sensor and V_{REFINT} enable)
通过软件将该位置 1 和清零可使能/禁止温度传感器和 V_{REFINT} 通道。
0：禁止温度传感器和 V_{REFINT} 通道
1：使能温度传感器和 V_{REFINT} 通道
注： 当 TSVREFE 位置 1 时，必须禁止 VBATE。两个位同时置 1 时，仅进行 VBAT 转换。

位 22 **VBATE**：V_{BAT} 使能 (V_{BAT} enable)
通过软件将该位置 1 和清零可使能/禁止 V_{BAT} 通道。
0：禁止 V_{BAT} 通道
1：使能 V_{BAT} 通道

位 21:18 保留，必须保持复位值。

位 17:16 **ADCPRE**：ADC 预分频器 (ADC prescaler)
由软件置 1 和清零，以选择 ADC 的时钟频率。该时钟为所有 ADC 所共用。
注： 00：PCLK2 2 分频
01：PCLK2 4 分频
10：PCLK2 6 分频
11：PCLK2 8 分频

位 15:14 **DMA**：直接存储器访问模式（对于多个 ADC 模式）(Direct memory access mode for multi ADC mode)
此位由软件置 1 和清零。有关详细信息，请参见 DMA 控制器一节。
00：禁止 DMA 模式
01：使能 DMA 模式 1（依次 2/3 半字 - 1、2、3 依次进行）
10：使能 DMA 模式 2（成对 2/3 半字 - 2 和 1、1 和 3、3 和 2 依次进行）
11：使能 DMA 模式 3（成对 2/3 字节 - 2 和 1、1 和 3、3 和 2 依次进行）

位 13 **DDS**：DMA 禁止选择（对于多个 ADC 模式）(DMA disable selection (for multi-ADC mode))
此位由软件置 1 和清零。
0：最后一次传输后不发出新的 DMA 请求（在 DMA 控制器中进行配置）。DMA 位不通过硬件清零，但必须在生成新的 DMA 请求前，通过软件清零并设置为需要的模式。
1：只要数据发生转换且 DMA = 01、10 或 11，便会发出 DMA 请求。

位 12 保留，必须保持复位值。

位 11:8 **DELAY**：2 个采样阶段之间的延迟 (Delay between 2 sampling phases)
由软件置 1 和清零。这些位在双重或三重交错模式下使用。
0000：5 * T_{ADCCLK}
0001：6 * T_{ADCCLK}
0010：7 * T_{ADCCLK}
...
1111：20 * T_{ADCCLK}

位 7:5 保留，必须保持复位值。

位 4:0 **MULTI[4:0]**：多重 ADC 模式选择 (Multi ADC mode selection)

通过软件写入这些位可选择操作模式。

– 所有 ADC 均独立：
00000：独立模式

– 00001 到 01001：双重模式，ADC1 和 ADC2 一起工作，ADC3 独立
00001：常规同步 + 注入同步组合模式
00010：常规同步 + 交替触发组合模式
00011：保留
00101：仅注入同步模式
00110：仅常规同步模式
00111：仅交错模式
01001：仅交替触发模式

– 10001 到 11001：三重模式：ADC1、ADC2 和 ADC3 一起工作
10001：常规同步 + 注入同步组合模式
10010：常规同步 + 交替触发组合模式
10011：保留
10101：仅注入同步模式
10110：仅常规同步模式
10111：仅交错模式
11001：仅交替触发模式

其它所有组合均需保留且不允许编程

注：　在多重模式下，更改通道配置会生成中止，进而导致同步丢失。建议在更改配置前禁用多重 ADC 模式。

15.13.17 适用于双重和三重模式的 ADC 通用常规数据寄存器 (ADC_CDR)

ADC common regular data register for dual and triple modes

偏移地址：0x08（该偏移地址与 ADC1 基地址 + 0x300 相关）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA2[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
DATA1[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 **DATA2[15:0]**：常规转换对中的第二个数据项 (2nd data item of a pair of regular conversions)

– 在双重模式下，这些位包含 ADC2 的常规数据。请参见*双重 ADC 模式*。

– 在三重模式下，这些位还可以包含 ADC2、ADC1 和 ADC3 的常规数据。请参见*三重 ADC 模式*。

位 15:0 **DATA1[15:0]**：常规转换对中的第一个数据项 (1st data item of a pair of regular conversions)

– 在双重模式下，这些位包含 ADC1 的常规数据。请参见*双重 ADC 模式*。

– 在三重模式下，这些位还可以包含 ADC1、ADC3 和 ADC2 的常规数据。请参见*三重 ADC 模式*。

15.13.18 ADC 寄存器映射

下表对 ADC 寄存器进行了汇总。

表 96. ADC 全局寄存器映射

偏移	寄存器
0x000 - 0x04C	ADC1
0x050 - 0x0FC	Reserved
0x100 - 0x14C	ADC2
0x118 - 0x1FC	Reserved
0x200 - 0x24C	ADC3
0x250 - 0x2FC	Reserved
0x300 - 0x308	通用寄存器

表 97. 每个 ADC 的 ADC 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	**ADC_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.	OVR	STRT	JSTRT	JEOC	EOC	AWD
	Reset value																											0	0	0	0	0	0
0x04	**ADC_CR1**	Res.	Res.	Res.	Res.	Res.	OVRIE	RES[1:0]		AWDEN	JAWDEN	Res.	Res.	Res.	Res.	Res.	Res.	DISCNUM [2:0]			JDISCEN	DISCEN	JAUTO	AWD SGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[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
0x08	**ADC_CR2**	Res.	SWSTART	EXTEN[1:0]		EXTSEL [3:0]				Res.	JSWSTART	JEXTEN[1:0]		JEXTSEL [3:0]				Res.	Res.	Res.	Res.	ALIGN	EOCS	DDS	DMA	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CONT	ADON
	Reset value		0	0	0	0	0	0	0		0	0	0	0	0	0	0					0	0		0							0	0
0x0C	**ADC_SMPR1**	Sample time bits SMPx_x																															
	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	**ADC_SMPR2**	Sample time bits SMPx_x																															
	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	**ADC_JOFR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JOFFSET1[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x18	**ADC_JOFR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JOFFSET2[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x1C	**ADC_JOFR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JOFFSET3[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x20	**ADC_JOFR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JOFFSET4[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x24	**ADC_HTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HT[11:0]											
	Reset value																					1	1	1	1	1	1	1	1	1	1	1	1
0x28	**ADC_LTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LT[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0

表 97. 每个 ADC 的 ADC 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
0x2C	**ADC_SQR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	L[3:0]				Regular channel sequence SQx_x bits																			
	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	**ADC_SQR2**	Res.	Res.	Regular channel sequence SQx_x bits																													
	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
0x34	**ADC_SQR3**	Res.	Res.	Regular channel sequence SQx_x bits																													
	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
0x38	**ADC_JSQR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JL[1:0]		Injected channel sequence JSQx_x bits																			
	Reset value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	**ADC_JDR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	**ADC_JDR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	**ADC_JDR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	**ADC_JDR4**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	**ADC_DR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Regular DATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 98. ADC 寄存器映射和复位值（通用 DAC 寄存器）

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	**ADC_CSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OVR	STRT	JSTRT	JEOC	EOC	AWD	Res.	Res.	OVR	STRT	JSTRT	JEOC	EOC	AWD	Res.	Res.	OVR	STRT	JSTRT	JEOC	EOC	AWD
	Reset value											0	0	0	0	0	0			0	0	0	0	0	0			0	0	0	0	0	0
												ADC3								ADC2								ADC1					
0x04	**ADC_CCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSVREFE	VBATE	Res.	Res.	Res.	Res.	ADCPRE[1:0]		DMA[1:0]		DDS	Res.	DELAY [3:0]				Res.	Res.	Res.	MULTI [4:0]				
	Reset value									0	0					0	0	0	0	0		0	0	0	0				0	0	0	0	0
0x08	**ADC_CDR**	Regular DATA2[15:0]																Regular DATA1[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

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

16 数模转换器 (DAC)

16.1 DAC 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；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。可通过一个输入参考电压引脚 V_{REF+}（与 ADC 共享）来提高精度。

16.2 DAC 主要特性

- 两个 DAC 转换器：各对应一个输出通道
- 12 位模式下数据采用左对齐或右对齐
- 同步更新功能
- 生成噪声波
- 生成三角波
- DAC 双通道单独或同时转换
- 每个通道都具有 DMA 功能
- DMA 下溢错误检测
- 通过外部触发信号进行转换
- 输入参考电压 V_{REF+}

图 92 所示为 DAC 通道的框图，*表 99* 则给出了引脚说明。

图 92. DAC 通道框图

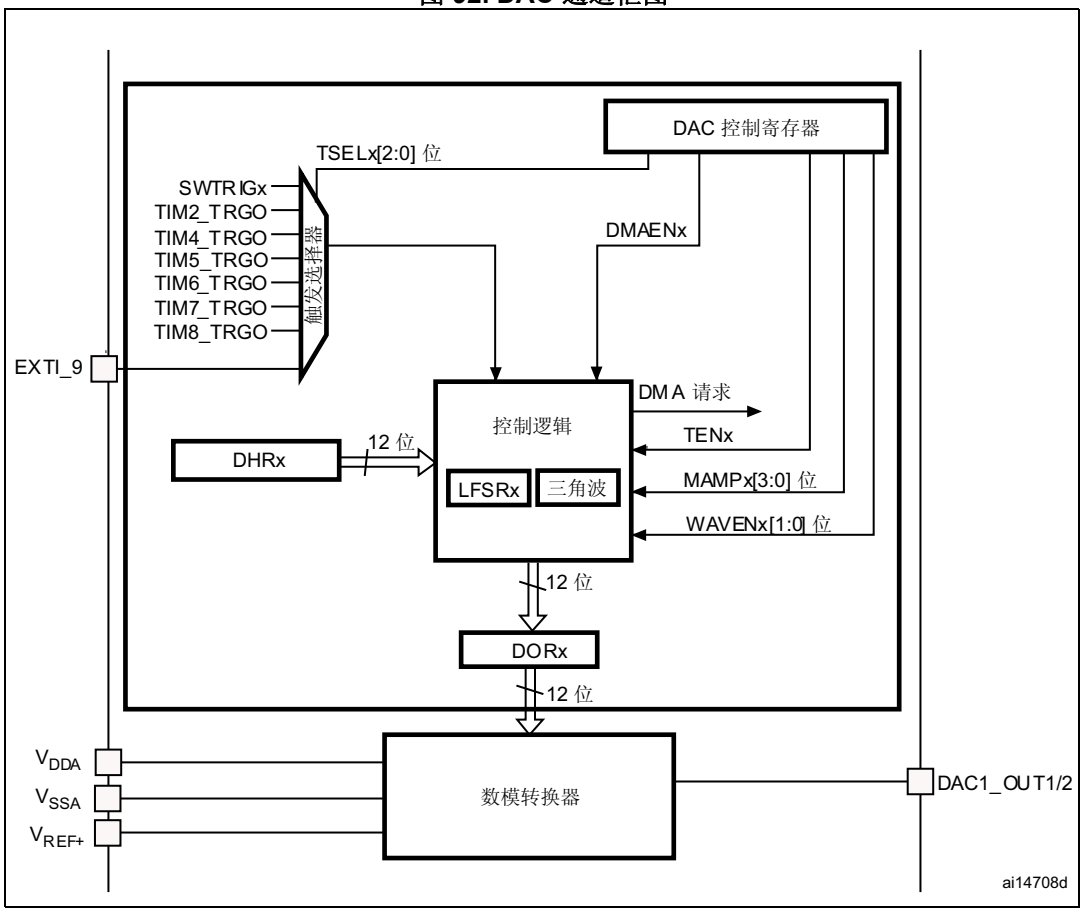


表 99. DAC 引脚

名称	信号类型	备注
V_{REF+}	正模拟参考电压输入	DAC 高/正参考电压，1.8 V ≤ V_{REF+} ≤ V_{DDA}
V_{DDA}	模拟电源输入	模拟电源
V_{SSA}	模拟电源地输入	模拟电源地
DAC_OUTx	模拟输出信号	DAC 通道 x 模拟输出

注： *使能 DAC 通道 x 后，相应 GPIO 引脚（PA4 或 PA5）将自动连接到模拟转换器输出 (DAC_OUTx)。为了避免寄生电流消耗，应首先将 PA4 或 PA5 引脚配置为模拟模式 (AIN)。*

16.3 DAC 功能说明

16.3.1 DAC 通道使能

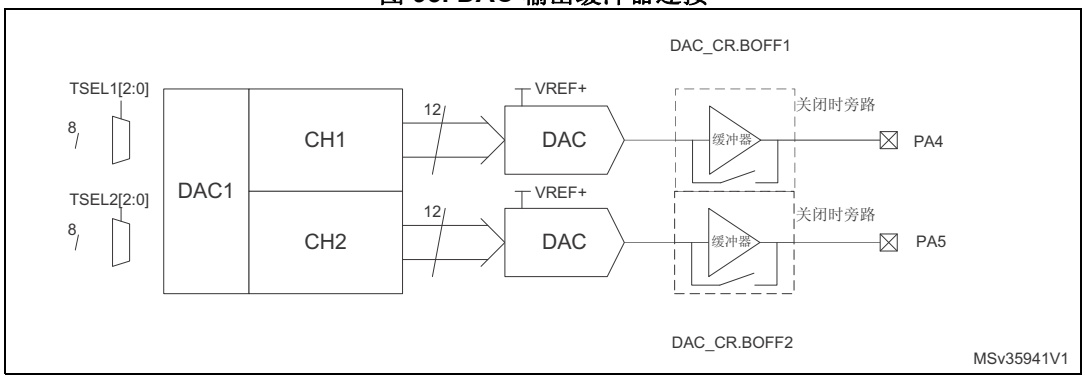
将 DAC_CR 寄存器中的相应 ENx 位置 1，即可使能对应 DAC 通道。经过一段启动时间 t_{WAKEUP} 后，DAC 通道被真正使能。

注： *ENx 位只会使能模拟 DAC Channelx 宏单元。即使 ENx 位复位，DAC Channelx 数字接口仍处于使能状态。*

16.3.2 DAC 输出缓冲器使能

DAC 集成了两个输出缓冲器，可用来降低输出阻抗并在不增加外部运算放大器的情况下直接驱动外部负载。通过 DAC_CR 寄存器中的相应 BOFFx 位，可使能或禁止各 DAC 通道输出缓冲器。

图 93. DAC 输出缓冲器连接



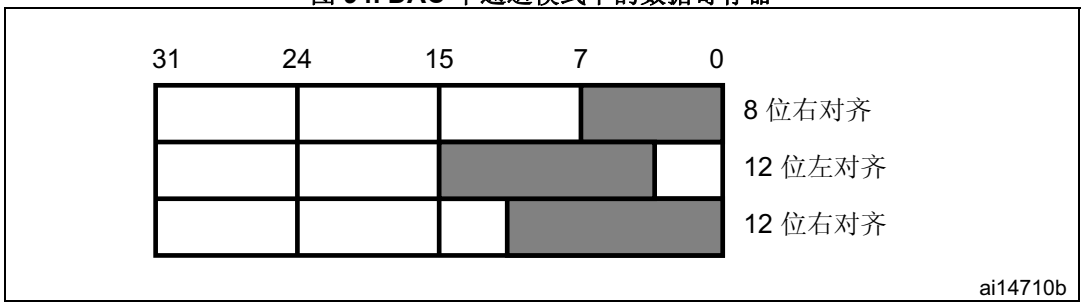
16.3.3 DAC 数据格式

根据所选配置模式，数据必须按如下方式写入指定寄存器：

- 对于 DAC 单通道 x，有三种可能的方式：
  - 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 94. DAC 单通道模式下的数据寄存器

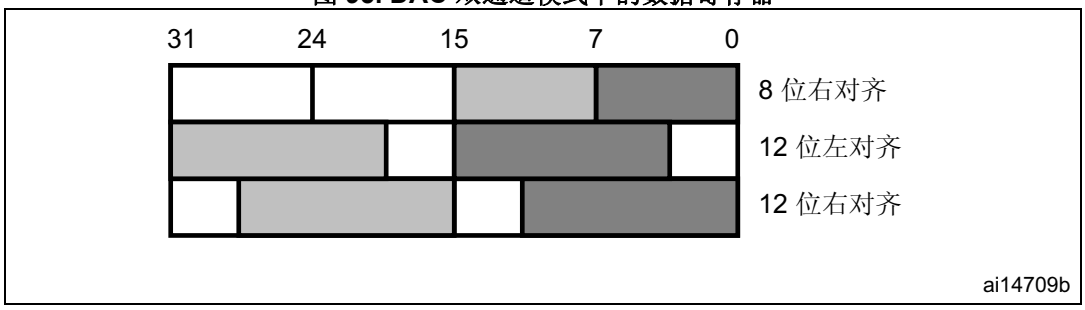


- 对于 DAC 双通道，有三种可能的方式：
  - 8 位右对齐：将 DAC 1 通道的数据加载到 DAC_DHR8RD [7:0] 位（存储到 DHR1[11:4] 位），将 DAC 2 通道的数据加载到 DAC_DHR8RD [15:8] 位（存储到 DHR2[11:4] 位）

- 12 位左对齐：将 DAC 1 通道的数据加载到 DAC_DHR12RD [15:4] 位（存储到 DHR1[11:0] 位），将 DAC 2 通道的数据加载到 DAC_DHR12RD [31:20] 位（存储到 DHR2[11:0] 位）
- 12 位右对齐：将 DAC 1 通道的数据加载到 DAC_DHR12RD [11:0] 位（存储到 DHR1[11:0] 位），将 DAC 2 通道的数据加载到 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 DOR1 和 DOR2 寄存器。

图 95. DAC 双通道模式下的数据寄存器



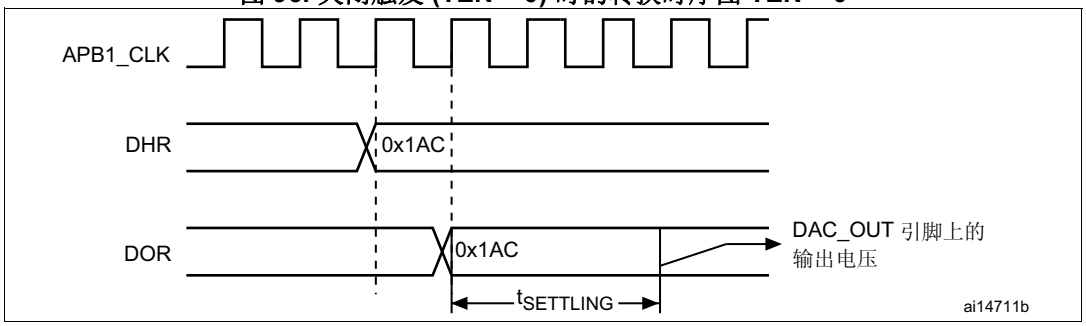
16.3.4 DAC 转换

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 APB1 时钟周期后，DAC_DHRx 寄存器中存储的数据将自动转移到 DAC_DORx 寄存器。但是，如果选择硬件触发（置位 DAC_CR 寄存器中的 TENx 位）且触发条件到来，将在三个 APB1 时钟周期后进行转移。

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。

图 96. 关闭触发 (TEN = 0) 时的转换时序图 TEN = 0



16.3.5 DAC 输出电压

经过线性转换后，数字输入会转换为 0 到 V_{REF+} 之间的输出电压。

各 DAC 通道引脚的模拟输出电压通过以下公式确定：

$$DACoutput = V_{REF} \times \frac{DOR}{4096}$$

16.3.6 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[2:0] 控制位将决定通过 8 个可能事件中的哪一个来触发转换，如*表 100* 所示。

表 100. 外部触发器

源	类型	TSEL[2:0]
定时器 6 TRGO 事件	片上定时器的内部信号	000
定时器 8 TRGO 事件		001
定时器 7 TRGO 事件		010
定时器 5 TRGO 事件		011
定时器 2 TRGO 事件		100
定时器 4 TRGO 事件		101
EXTI 线 9	外部引脚	110
SWTRIG	软件控制位	111

每当 DAC 接口在所选定时器 TRGO 输出或所选外部中断线 9 上检测到上升沿时，DAC_DHRx 寄存器中存储的最后一个数据即会转移到 DAC_DORx 寄存器中。发生触发后再经过三个 APB1 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： *ENx 位置 1 时，无法更改 TSELx[2:0] 位。*

如果选择软件触发，DAC_DHRx 寄存器的内容只需一个 APB1 时钟周期即可转移到 DAC_DORx 寄存器。

16.3.7 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），则将产生 DAC DMA 请求。DAC_DHRx 寄存器的值随后转移到 DAC_DORx 寄存器。
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，应仅将相应 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。

DMA 下溢

DAC DMA 请求没有缓冲队列。这样，如果第二个外部触发到达时尚未收到第一个外部触发的应答，将不会发出新的请求，并且 DAC_SR 寄存器中的 DAM 通道下溢标志 DMAUDRx 将置 1，以报告这一错误状况。DMA 数据传输随即禁止，并且不再处理其他 DMA 请求。DAC 通道仍将继续转换旧有数据。

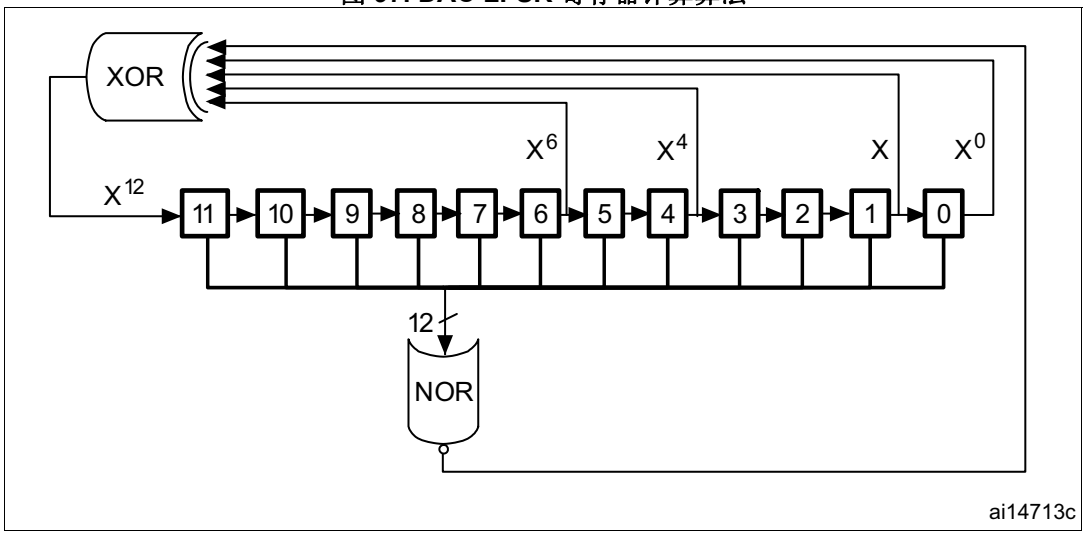
软件应通过写入“1”来将 DMAUDRx 标志清零，将所用 DMA 数据流的 DMAEN 位清零，并重新初始化 DMA 和 DAC 通道，以便正确地重新开始 DMA 传输。软件应修改 DAC 触发转换频率或减轻 DMA 工作负载，以避免再次发生 DMA 下溢。最后，可通过使能 DMA 数据传输和转换触发来继续完成 DAC 转换。

对于各 DAC 通道，如果使能 DAC_CR 寄存器中相应的 DMAUDRIEx 位，还将产生中断。

16.3.8 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 APB1 时钟周期，该寄存器会依照特定的计算算法完成更新。

图 97. DAC LFSR 寄存器计算算法

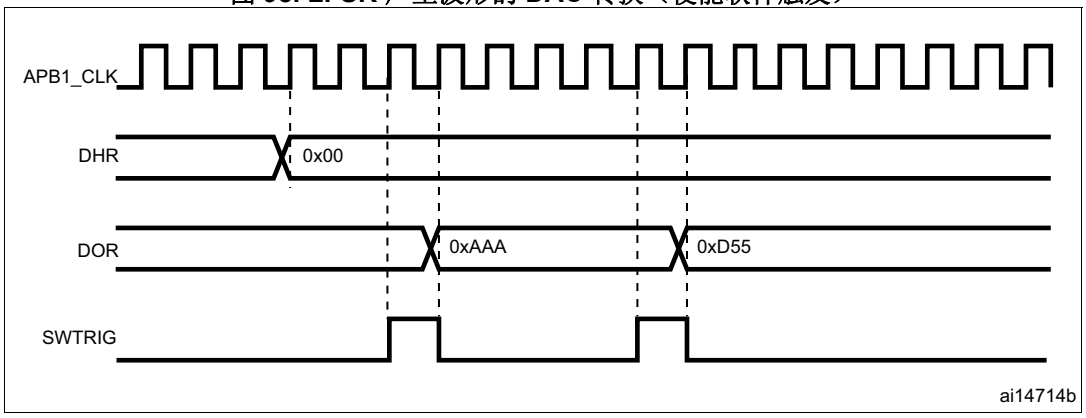


LFSR 值可以通过 DAC_CR 寄存器中的 MAMPx[3:0] 位来部分或完全屏蔽，在不发生溢出的情况下，该值将与 DAC_DHRx 的内容相加，然后存储到 DAC_DORx 寄存器中。

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 波形产生功能关闭。

图 98. LFSR 产生波形的 DAC 转换（使能软件触发）



注： *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。*

16.3.9 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 DAC 生成三角波。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 APB1 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，所得总和将存储到 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] 位来将三角波产生功能关闭。

图 99. 生成 DAC 三角波

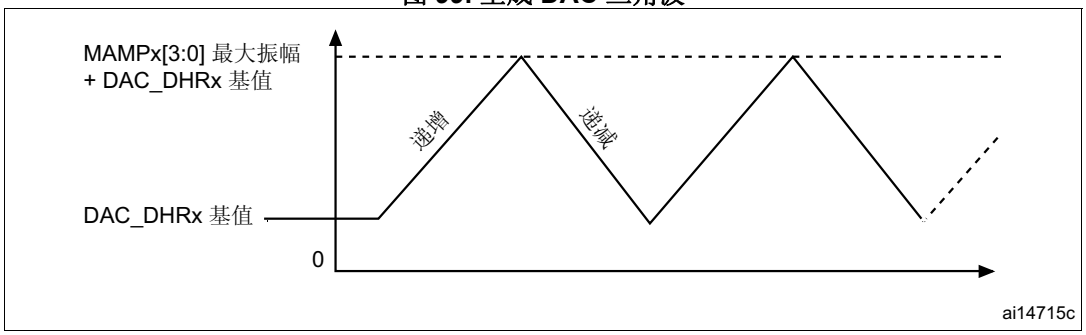
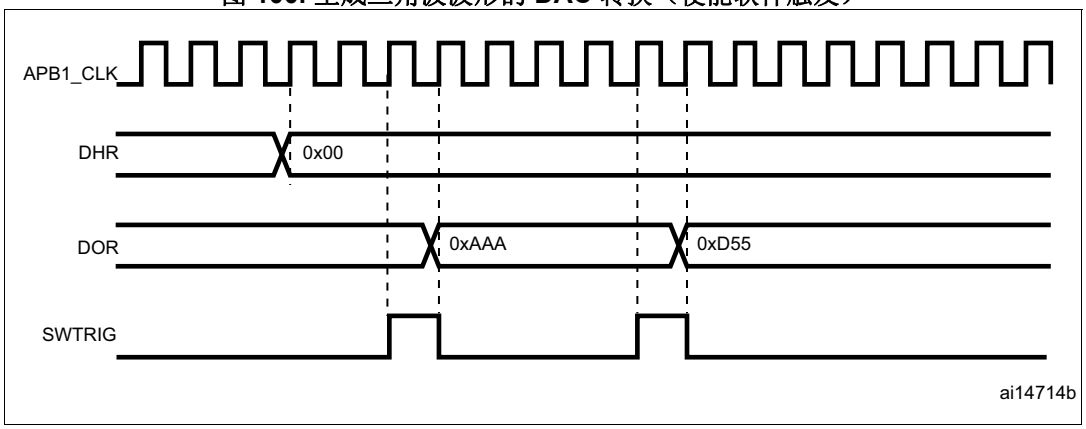


图 100. 生成三角波波形的 DAC 转换（使能软件触发）



注： *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。*

16.4 DAC 双通道转换

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 模块有三个双寄存器可供操作：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。

通过两个 DAC 通道和这三个双寄存器可以实现 11 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。

16.4.1 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：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源
- 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）

DAC 1 通道触发信号到达时，DHR1 寄存器的内容转移到 DAC_DOR1（三个 APB1 时钟周期之后）。

DAC 2 通道触发信号到达时，DHR2 寄存器的内容转移到 DAC_DOR2（三个 APB1 时钟周期之后）。

16.4.2 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：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源
- 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
- 将 DAC 双通道数据加载到所需 DHR 寄存器（DHR12RD、DHR12LD 或 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，所得总和转移到 DAC_DOR1 中（三个 APB1 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，所得总和转移到 DAC_DOR2 中（三个 APB1 时钟周期之后）。LFSR2 计数器随即更新。

16.4.3 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：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源
- 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
- 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，所得总和转移到 DAC_DOR1 中（三个 APB1 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，所得总和转移到 DAC_DOR2 中（三个 APB1 时钟周期之后）。LFSR2 计数器随即更新。

16.4.4 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：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源
- 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
- 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 1 通道 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，所得总和转移到 DAC_DOR1 中（三个 APB1 时钟周期之后）。DAC 1 通道 三角波计数器随即更新。

DAC 2 通道 触发信号到达时，DAC 2 通道 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，所得总和转移到 DAC_DOR2 中（三个 APB1 时钟周期之后）。DAC 2 通道 三角波计数器随即更新。

16.4.5 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：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源
- 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
- 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）

DAC 1 通道 触发信号到达时，DAC 1 通道 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，所得总和转移到 DAC_DOR1 中（三个 APB1 时钟周期之后）。DAC 1 通道 三角波计数器随即更新。

DAC 2 通道 触发信号到达时，DAC 2 通道 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，所得总和转移到 DAC_DOR2 中（三个 APB1 时钟周期之后）。DAC 2 通道 三角波计数器随即更新。

16.4.6 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 APB1 时钟周期后分别转移到 DAC_DOR1 和 DAC_DOR2 中。

16.4.7 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：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源
- 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 寄存器内容将分别转移到 DAC_DOR1 和 DAC_DOR2 中（三个 APB1 时钟周期之后）。

16.4.8 同步触发（生成单个 LFSR）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源
- 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
- 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，LFS1 R计数器内容（使用相同的掩码）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个 APB1 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，所得总和转移到 DAC_DOR2 中（三个 APB1 时钟周期之后）。LFSR2 计数器随即更新。

16.4.9 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：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源
- 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
- 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，所得总和转移到 DAC_DOR1 中（三个 APB1 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，所得总和转移到 DAC_DOR2 中（三个 APB1 时钟周期之后）。LFSR2 计数器随即更新。

16.4.10 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：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源
- 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
- 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 1 通道 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，所得总和转移到 DAC_DOR1 中（三个 APB1 时钟周期之后）。DAC 1 通道三角波计数器随即更新。
同时，DAC 2 通道 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，所得总和转移到 DAC_DOR2 中（三个 APB1 时钟周期之后）。DAC 2 通道三角波计数器随即更新。

16.4.11 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：

- 将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1
- 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源
- 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
- 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，所得总和转移到 DAC_DOR1 中（三个 APB1 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，所得总和转移到 DAC_DOR2 中（三个 APB1 时钟周期之后）。DAC 通道 2 三角波计数器随即更新。

16.5 DAC 寄存器

有关寄存器说明中使用的缩写，请参见*第 58 页上的第 1 节*。

外设寄存器必须按字（32 位）进行访问。

16.5.1 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.	Res.	DMAUDRIE2	DMAEN2	MAMP2[3:0]				WAVE2[1:0]		TSEL2[2:0]			TEN2	BOFF2	EN2
		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.	DMAUDRIE1	DMAEN1	MAMP1[3:0]				WAVE1[1:0]		TSEL1[2:0]			TEN1	BOFF1	EN1
		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 **DMAUDRIE2**：DAC 2 通道 DMA 下溢中断使能 (DAC channel2 DMA underrun interrupt enable)
此位由软件置 1 和清零。
0：禁止 DAC 2 通道 DMA 下溢中断
1：使能 DAC 2 通道 DMA 下溢中断

位 28 **DMAEN2**：DAC 2 通道 DMA 使能 (DAC channel2 DMA enable)
此位由软件置 1 和清零。
0：禁止 DAC 2 通道 DMA 模式
1：使能 DAC 2 通道 DMA 模式

位 27:24 **MAMP2[3:0]**：DAC 2 通道掩码/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

位 23:22 **WAVE2[1:0]**：DAC 2 通道噪声/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： 只在位 TEN2 = 1（使能 DAC 2 通道触发）时使用

位 21:19 **TSEL2[2:0]**：DAC 2 通道触发器选择 (DAC channel2 trigger selection)

这些位用于选择 DAC 2 通道的外部触发事件

000：定时器 6 TRGO 事件

001：定时器 8 TRGO 事件

010：定时器 7 TRGO 事件

011：定时器 5 TRGO 事件

100：定时器 2 TRGO 事件

101：定时器 4 TRGO 事件

110：外部中断线 9

111：软件触发

注： 只在位 TEN2 = 1（使能 DAC 2 通道触发）时使用。

位 18 **TEN2**：DAC 2 通道触发使能 (DAC channel2 trigger enable)

此位由软件置 1 和清零，以使能/禁止 DAC 2 通道触发

0：禁止 DAC 2 通道触发，写入 DAC_DHRx 寄存器的数据在一个 APB1 时钟周期之后转移到 DAC_DOR2 寄存器

1：使能 DAC 2 通道触发，DAC_DHRx 寄存器的数据在三个 APB1 时钟周期之后转移到 DAC_DOR2 寄存器

注： 如果选择软件触发，DAC_DHRx 寄存器的内容只需一个 APB1 时钟周期即可转移到 DAC_DOR2 寄存器。

位 17 **BOFF2**：DAC 2 通道输出缓冲器禁止 (DAC channel2 output buffer disable)

此位由软件置 1 和清零，以使能/禁止 DAC 2 通道输出缓冲器。

0：使能 DAC 2 通道输出缓冲器

1：禁止 DAC 2 通道输出缓冲器

位 16 **EN2**：DAC 2 通道使能 (DAC channel2 enable)

此位由软件置 1 和清零，以使能/禁止 DAC 2 通道。

0：禁止 DAC 2 通道

1：使能 DAC 2 通道

位 15:14 保留，必须保持复位值。

位 13 **DMAUDRIE1**：DAC 1 通道 DMA 下溢中断使能 (DAC channel1 DMA Underrun Interrupt enable)

此位由软件置 1 和清零。

0：禁止 DAC 1 通道 DMA 下溢中断

1：使能 DAC 1 通道 DMA 下溢中断

位 12 **DMAEN1**：DAC 1 通道 DMA 使能 (DAC channel1 DMA enable)

此位由软件置 1 和清零。

0：禁止 DAC 1 通道 DMA 模式

1：使能 DAC 1 通道 DMA 模式

位 11:8 **MAMP1[3:0]**：DAC 1 通道掩码/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 1 通道噪声/三角波生成使能 (DAC channel1 noise/triangle wave generation enable)

这些位将由软件置 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

注： 只在位 TEN1 = 1（使能 DAC 1 通道触发）时使用。

位 5:3 **TSEL1[2:0]**：DAC 1 通道触发器选择 (DAC channel1 trigger selection)

这些位用于选择 DAC 1 通道的外部触发事件。

000：定时器 6 TRGO 事件

001：定时器 8 TRGO 事件

010：定时器 7 TRGO 事件

011：定时器 5 TRGO 事件

100：定时器 2 TRGO 事件

101：定时器 4 TRGO 事件

110：外部中断线 9

111：软件触发

注： 只在位 TEN1 = 1（使能 DAC 1 通道触发）时使用。

位 2 **TEN1**：DAC 1 通道触发使能 (DAC channel1 trigger enable)

此位由软件置 1 和清零，以使能/禁止 DAC 1 通道触发。

0：禁止 DAC 1 通道触发，写入 DAC_DHRx 寄存器的数据在一个 APB1 时钟周期之后转移到 DAC_DOR1 寄存器

1：使能 DAC 1 通道触发，DAC_DHRx 寄存器的数据在三个 APB1 时钟周期之后转移到 DAC_DOR1 寄存器

注： 如果选择软件触发，DAC_DHRx 寄存器的内容只需一个 APB1 时钟周期即可转移到 DAC_DOR1 寄存器。

位 1 **BOFF1**：DAC 1 通道输出缓冲器禁止 (DAC channel1 output buffer disable)
此位由软件置 1 和清零，以使能/禁止 DAC 1 通道输出缓冲器。
0：使能 DAC 1 通道输出缓冲器
1：禁止 DAC 1 通道输出缓冲器

位 0 **EN1**：DAC 1 通道使能 (DAC channel1 enable)
此位由软件置 1 和清零，以使能/禁止 DAC 1 通道。
0：禁止 DAC 1 通道
1：使能 DAC 1 通道

16.5.2 DAC 软件触发寄存器 (DAC_SWTRIGR)

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 2 通道软件触发 (DAC channel2 software trigger)
此位由软件置 1 和清零，以使能/禁止软件触发。
0：禁止软件触发
1：使能软件触发
注：　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 APB1 时钟周期之后）。

位 0 **SWTRIG1**：DAC 1 通道软件触发 (DAC channel1 software trigger)
此位由软件置 1 和清零，以使能/禁止软件触发。
0：禁止软件触发
1：使能软件触发
注：　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 APB1 时钟周期之后）。

16.5.3 DAC 1 通道 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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 31:12 保留，必须保持复位值。

位 11:0 **DACC1DHR[11:0]**：DAC 1 通道 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 1 通道指定 12 位数据。

16.5.4 DAC 1 通道 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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 31:16 保留，必须保持复位值。

位 15:4 **DACC1DHR[11:0]**：DAC 1 通道 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 1 通道指定 12 位数据。

位 3:0 保留，必须保持复位值。

16.5.5 DAC 1 通道 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC1DHR[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 **DACC1DHR[7:0]**：DAC 1 通道 8 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 1 通道指定 8 位数据。

16.5.6 DAC 2 通道 12 位右对齐数据保持寄存器 (DAC_DHR12R2)

DAC channel2 12-bit right aligned data holding 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
Res.	Res.	Res.	Res.	DACC2DHR[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 **DACC2DHR[11:0]**：DAC 2 通道 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 2 通道指定 12 位数据。

16.5.7 DAC 2 通道 12 位左对齐数据保持寄存器 (DAC_DHR12L2)

DAC channel2 12-bit left aligned data holding 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
DACC2DHR[11:0]												Res.	Res.	Res.	Res.
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 15:4 **DACC2DHR[11:0]**：DAC 2 通道 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 2 通道指定 12 位数据。

位 3:0 保留，必须保持复位值。

16.5.8 DAC 2 通道 8 位右对齐数据保持寄存器 (DAC_DHR8R2)

DAC channel2 8-bit right-aligned data holding register

偏移地址：0x1C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DHR[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 **DACC2DHR[7:0]**：DAC 2 通道 8 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 2 通道指定 8 位数据。

16.5.9 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 2 通道 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 2 通道指定 12 位数据。

位 15:12 保留，必须保持复位值。

位 11:0 **DACC1DHR[11:0]**：DAC 1 通道 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 1 通道指定 12 位数据。

16.5.10 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 2 通道 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 2 通道指定 12 位数据。

位 19:16 保留，必须保持复位值。

位 15:4 **DACC1DHR[11:0]**：DAC 1 通道 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 1 通道指定 12 位数据。

位 3:0 保留，必须保持复位值。

16.5.11 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 2 通道 8 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 2 通道指定 8 位数据。

位 7:0 **DACC1DHR[7:0]**：DAC 1 通道 8 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 1 通道指定 8 位数据。

16.5.12 DAC 1 通道数据输出寄存器 (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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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 31:12 保留，必须保持复位值。

位 11:0 **DACC1DOR[11:0]**：DAC 1 通道数据输出 (DAC channel1 data output)
这些位为只读，其中包含 DAC 1 通道的数据输出。

16.5.13 DAC 2 通道数据输出寄存器 (DAC_DOR2)

DAC channel2 data output 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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 31:12 保留，必须保持复位值。

位 11:0 **DACC2DOR[11:0]**：DAC 2 通道数据输出 (DAC channel2 data output)
这些位为只读，其中包含 DAC 2 通道的数据输出。

16.5.14 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
Res.	Res.	DMAUDR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
		rc_w1													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	DMAUDR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
		rc_w1													

位 31:30 保留，必须保持复位值。

位 29 **DMAUDR2**：DAC 2 通道 DMA 下溢标志 (DAC channel2 DMA underrun flag)
此位由硬件置 1，由软件清零（写入 1）。
0：DAC 2 通道未发生 DMA 下溢错误状况
1：DAC 2 通道发生 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 2 通道转换）

位 28:14 保留，必须保持复位值。

位 13 **DMAUDR1**：DAC 1 通道 DMA 下溢标志 (DAC channel1 DMA underrun flag)
此位由硬件置 1，由软件清零（写入 1）。
0：DAC 1 通道未发生 DMA 下溢错误状况
1：DAC 1 通道发生 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 1 通道转换）

位 12:0 保留，必须保持复位值。

16.5.15 DAC 寄存器映射

表 101 汇总了 DAC 寄存器。

表 101. DAC 寄存器映射

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	DAC_CR	Res.	Res.	DMAUDRIE2	DMAEN2	MAMP2[3:0]				WAVE2[2:0]		TSEL2[2:0]			TEN2	BOFF2	EN2	Res.	Res.	DMAUDRIE1	DMAEN1	MAMP1[3:0]				WAVE1[2:0]		TSEL1[2:0]			TEN1	BOFF1	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

表 101. DAC 寄存器映射（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	DAC_SWTRIGR	Res.	Res.	Res.	Res.	Res.	Res.	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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC1DHR[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x0C	DAC_DHR12L1	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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				
0x10	DAC_DHR8R1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC1DHR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x14	DAC_DHR12R2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DHR[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x18	DAC_DHR12L2	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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				
0x1C	DAC_DHR8R2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DHR[7:0]							
	Reset value																									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]												Reserved				DACC1DHR[11:0]												Reserved			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0				
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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC1DOR[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x30	DAC_DOR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DOR[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x34	DAC_SR	Res.	Res.	DMAUDR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAUDR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value			0																0													

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

17 数字摄像头接口 (DCMI)

17.1 DCMI 简介

数字摄像头接口是一个同步并行接口，能够接收外部 8 位、10 位、12 位或 14 位 CMOS 摄像头模块发出的高速数据流。可支持不同的数据格式：YCbCr4:2:2/RGB565 逐行视频和压缩数据 (JPEG)。

此接口适用于黑白摄像头、X24 和 X5 摄像头，并假定所有预处理（如调整大小）都在摄像头模块中执行。

17.2 DCMI 主要特性

- 8 位、10 位、12 位或 14 位并行接口
- 内嵌码/外部行同步和帧同步
- 连续模式或快照模式
- 裁剪功能
- 支持以下数据格式：
  - 8/10/12/14 位逐行视频：单色或原始拜尔格式
  - YCbCr 4:2:2 逐行视频
  - RGB 565 逐行视频
  - 压缩数据：JPEG

17.3 DCMI 引脚

表 102 显示了 DCMI 引脚。

表 102. DCMI 引脚

名称	信号类型
D[0:13]	数据输入
HSYNC	水平同步（行同步）输入
VSYNC	垂直同步（场同步）输入
PIXCLX	像素时钟输入

17.4 DCMI 时钟

数字摄像头接口使用 PIXCLK 和 HCLK 这两个时钟域。伴随 PIXCLK 产生的信号稳定之后，将在 HCLK 上升沿时对这些信号进行采样。HCLK 域中的一个使能信号用于指示摄像头发出的数据已稳定，可进行采样。PIXCLK 的最大周期必须大于 2.5 个 HCLK 周期。

17.5 DCMI 功能概述

数字摄像头接口是一个同步并行接口，可接收高速（可达 54 MB/s）数据流。该接口包含多达 14 条数据线 (D13-D0) 和一条像素时钟线 (PIXCLK)。像素时钟的极性可以编程，因此可以在像素时钟的上升沿或下降沿捕获数据。

这些数据被放到 32 位数据寄存器 (DCMI_DR) 中，然后通过通用 DMA 进行传输。图像缓冲区由 DMA 管理，而不是由摄像头接口管理。

从摄像头接收的数据可以按行/帧来组织（原始 YUB/RGB/ 拜尔模式），也可以是一系列 JPEG 图像。要使能 JPEG 图像接收，必须将 JPEG 位（DCMI_CR 寄存器的位 3）置 1。

数据流可由可选的 HSYNC（水平同步）信号和 VSYNC（垂直同步）信号硬件同步，或者通过数据流中嵌入的同步码同步。

图 101 显示了 DCMI 框图。

图 101. DCMI 框图

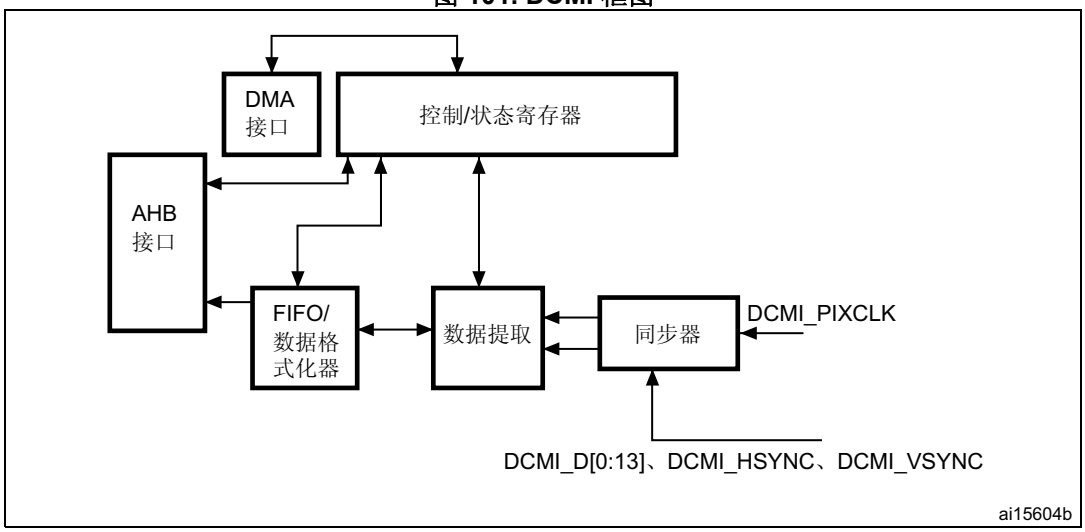
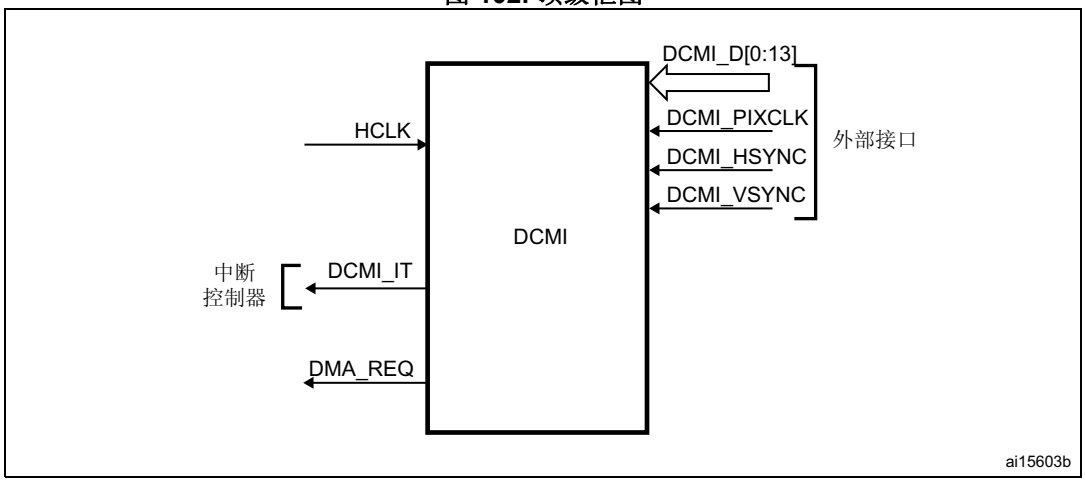


图 102. 顶级框图



17.5.1 DMA 接口

当 DCMI_CR 寄存器中的 CAPTURE 位置 1 时，激活 DMA 接口。摄像头接口每次在其寄存器中收到一个完整的 32 位数据块时，都将触发一个 DMA 请求。

17.5.2 DCMI 物理接口

该接口由 11/13/15/17 个输入信号组成。仅支持从模式。

根据 DCMI_CR 寄存器中 EDM[1:0] 位的设置，摄像头接口可以捕获 8 位、10 位、12 位或 14 位数据。如果使用的位数少于 14，则必须将未使用的输入引脚接地。

表 103. DCMI 信号

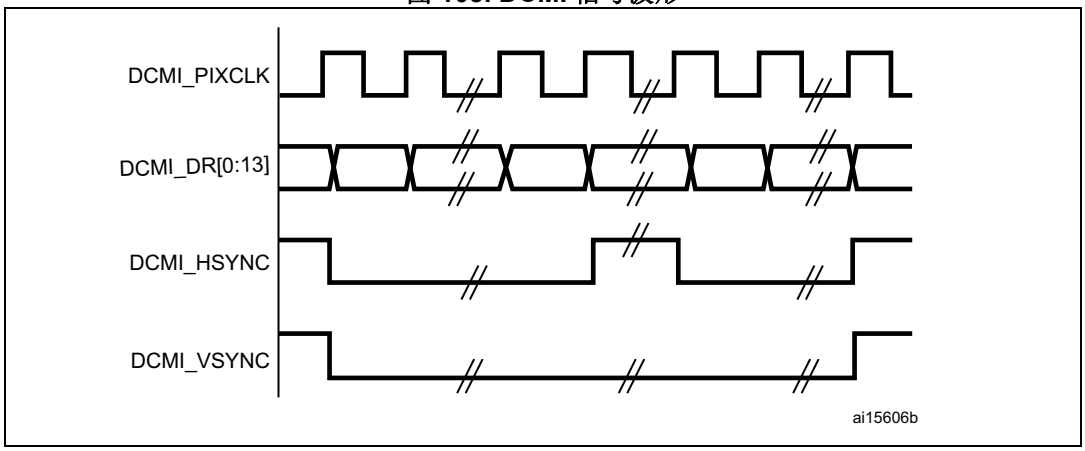
信号名称		信号说明
8 位 10 位 12 位 14 位	D[0..7] D[0..9] D[0..11] D[0..13]	Data
PIXCLK		像素时钟
HSYNC		水平同步/数据有效
VSYNC		垂直同步

数据与 PIXCLK 保持同步，并根据像素时钟的极性在像素时钟上升沿/下降沿发生变化。

HSYNC 信号指示行的开始/结束。

VSYNC 信号指示帧的开始/结束。

图 103. DCMI 信号波形



1. DCMI_PIXCLK 的捕获沿为下降沿，DCMI_HSYNC 和 DCMI_VSYNC 的有效状态为 1。
2. DCMI_HSYNC 和 DCMI_VSYNC 的状态可同时发生更改。

8 位数据

当 DCMI_CR 中的 EDM[1:0] 编程为“00”时，接口将捕获其输入 (D[0:7]) 的 8 个 LSB，并将其存储为 8 位数据。D[13:8] 输入则忽略。在此情况下，要捕获 32 位字，摄像头接口需要花费四个像素时钟周期。

捕获的第一个数据字节放置在 32 位字的 LSB 位置，捕获的第四个数据字节放置在 32 位字的 MSB 位置。*表 104* 列举了捕获到的数据字节在两个 32 位字中的位置排布。

表 104. 捕获的数据字节在 32 位字（宽 8 位）中的位置排布

字节地址	31:24	23:16	15:8	7:0
0	$D_{n+3}[7:0]$	$D_{n+2}[7:0]$	$D_{n+1}[7:0]$	$D_n[7:0]$
4	$D_{n+7}[7:0]$	$D_{n+6}[7:0]$	$D_{n+5}[7:0]$	$D_{n+4}[7:0]$

10 位数据

当 DCMI_CR 中的 EDM[1:0] 编程为“01”时，摄像头接口将捕获其输入 D[0..9] 的 10 位数据，并将其存储为 16 位字的 10 个最低有效位。DCMI_DR 寄存器中的其余最高有效位（位 11 到 15）将清零。因此，在此情况下，每两个像素时钟周期会生成一个 32 位数据字。

捕获的第一个数据放置在 32 位字的 LSB 位置，捕获的第二个数据放置在 32 位字的 MSB 位置，如*表 105* 中所示。

表 105. 捕获的数据字节在 32 位字（宽 10 位）中的位置排布

字节地址	31:26	25:16	15:10	9:0
0	0	$D_{n+1[9:0]}$	0	$D_n[9:0]$
4	0	$D_{n+3}[9:0]$	0	$D_{n+2}[9:0]$

12 位数据

当 DCMI_CR 中的 EDM[1:0] 编程为“10”时，摄像头接口将捕获其输入 D[0..11] 的 12 位数据，并将其存储为 16 位字的 12 个最低有效位。其余最高有效位将清零。因此，在此情况下，每两个像素时钟周期会生成一个 32 位数据字。

捕获的第一个数据放置在 32 位字的 LSB 位置，捕获的第二个数据放置在 32 位字的 MSB 位置，如*表 106* 中所示。

表 106. 捕获的数据字节在 32 位字（宽 12 位）中的位置排布

字节地址	31:28	27:16	15:12	11:0
0	0	$D_{n+1}[11:0]$	0	$D_n[11:0]$
4	0	$D_{n+3}[11:0]$	0	$D_{n+2}[11:0]$

14 位数据

当 DCMI_CR 中的 EDM[1:0] 编程为“11”时，摄像头接口将捕获其输入 D[0..13] 的 14 位数据，并将其存储为 16 位字的 14 个最低有效位。其余最高有效位将清零。因此，在此情况下，每两个像素时钟周期会生成一个 32 位数据字。

捕获的第一个数据放置在 32 位字的 LSB 位置，捕获的第二个数据放置在 32 位字的 MSB 位置，如*表 107* 中所示。

表 107. 捕获的数据字节在 32 位字（宽 14 位）中的位置排布

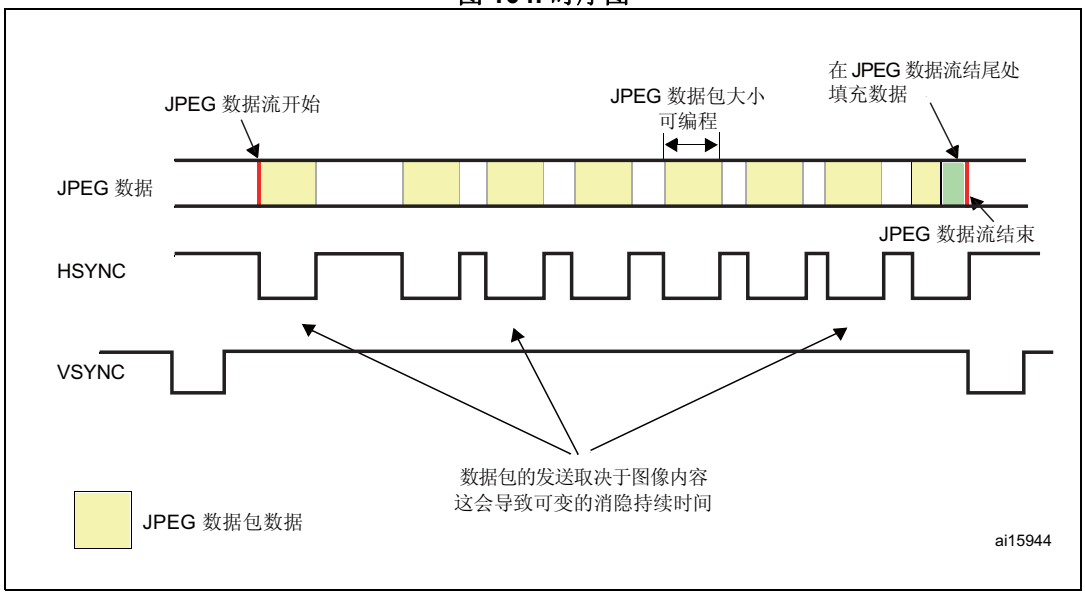
字节地址	31:30	29:16	15:14	13:0
0	0	D_{n+1}[13:0]	0	D_n[13:0]
4	0	D_{n+3}[13:0]	0	D_{n+2}[13:0]

17.5.3 同步

数字摄像头接口支持内嵌码同步或硬件（HSYNC 和 VSYNC）同步。使用内嵌码同步时，由数字摄像头模块确保 0x00 和 0xFF 值仅用于同步（不用于数据中）。只有 8 位并行数据接口宽度支持内嵌码同步码（即，DCMI_CR 寄存器中的 EDM[1:0] 位应为“00”）。

对于压缩数据，DCMI 仅支持硬件同步模式。这种情况下，VSYNC 指示图像的开始/结束，HSYNC 则用作“数据有效”信号。*图 104* 显示了相应的时序图。

图 104. 时序图



硬件同步模式

硬件同步模式下将使用两个同步信号 (HSYNC/VSYNC)。

根据摄像头模块/模式的不同，可能在水平/垂直同步期间内发送数据。由于系统会忽略 HSYNC/VSYNC 信号有效电平期间内接收的所有数据，HSYNC/VSYNC 信号相当于消隐信号。

为了正确地将图像传输到 DMA/RAM 缓冲区，数据传输将与 VSYNC 信号同步。选择硬件同步模式并启用捕获（DCMI_CR 中的 CAPTURE 位置 1）时，数据传输将与 VSYNC 信号的无效电平同步（开始下一帧时）。

之后传输便可以连续执行，由 DMA 将连续帧传输到多个连续的缓冲区或一个具有循环特性的缓冲区。为了允许 DMA 管理连续帧，每一帧结束时都将激活 VSIF（垂直同步中断标志）。

内嵌码同步模式

在此同步模式下，将使用数据流中嵌入的 32 位码来同步数据流。这些码使用数据中*不*再使用的值 0x00/0xFF。共有 4 种同步码类型，均采用 0xFF0000XY 格式。只有 8 位并行数据接口支持内嵌码同步（DCMI_CR 寄存器中的 EDM[1:0] 位应编程为“00”）。对于其它数据宽度，此模式将造成无法预知的结果，因此不得使用。

注： *（在隔行扫描模式下）摄像头模块具有 8 个此类代码。因此，摄像头接口不支持隔行扫描模式（否则每帧数据的一半都会被丢弃）。*

- 模式 2

  四个内嵌码可表示以下事件

  – 帧开始 (FS)
  – 帧结束 (FE)
  – 行开始 (LS)
  – 行结束 (LE)

  4 个同步码采用的格式 0xFF0000XY 中的 XY 值可编程（参见*第 17.8.7 节：DCMI 内嵌同步码寄存器 (DCMI_ESCR)*）。

  将 0xFF 编程为“帧结束”意味着所有除此之外的同步码都视为有效的帧结束同步码。

  在此模式下，一旦使能摄像头接口，将在首次出现帧结束 (FE) 同步码并且后接帧开始 (FS) 同步码之后开始捕获帧。

- 模式 1

  摄像头模式 1 是另一种编码。此模式与 ITU656 兼容。

  这些同步码表示另一组事件：

  – SAV（有效行）——行开始
  – EAV（有效行）——行结束
  – SAV（消隐）——帧间消隐期内的行开始
  – EAV（消隐）——帧间消隐期内的行结束

可通过对同步码进行如下编程来支持此模式：

- FS ≤ 0xFF
- FE ≤ 0xFF
- LS ≤ SAV（有效）
- LE ≤ EAV（有效）

此外还针对帧/行开始和帧/行结束同步码实现了非屏蔽位功能。这样可以仅使用同步码中未被屏蔽的位进行比较。因此，可以选择一个位用于同步码的比较，来检测帧/行起始和帧/行结束。这意味着可以多个同步码表示帧/行的起始和结束，它们仅在未被屏蔽的位相同即可。

示例

FS = 0xA5

FS 的非屏蔽码 = 0x10

这种情况下，只需要比较数据码的第 4 位来检测是否是 FS 信号。

17.5.4 捕获模式

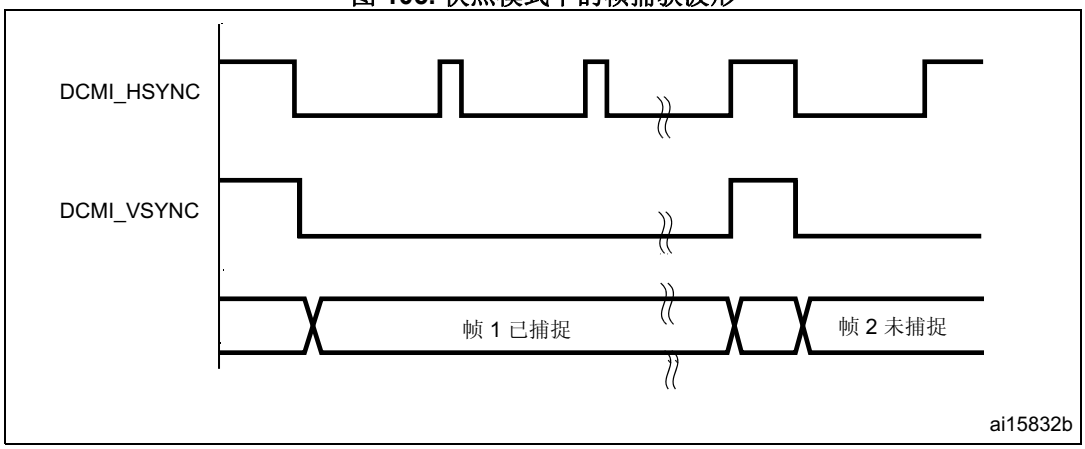
此接口支持两种类型的捕获：快照（单帧）和连续采集。

快照模式（单帧）

此模式下只捕获单帧（DCMI_CR 寄存器中的 CM = “1”）。在 DCMI_CR 中的 CAPTURE 位置 1 后，该接口将等待系统检测帧开始，然后再对数据进行采样。收到完整的第一帧后，将自动禁止摄像头接口（DCMI_CR 中的 CAPTURE 位清零）。如果使能相应中断，将生成中断 (IT_FRAME)。

如果发生溢出，这帧将丢失并且 CAPTURE 位清零。

图 105. 快照模式下的帧捕获波形

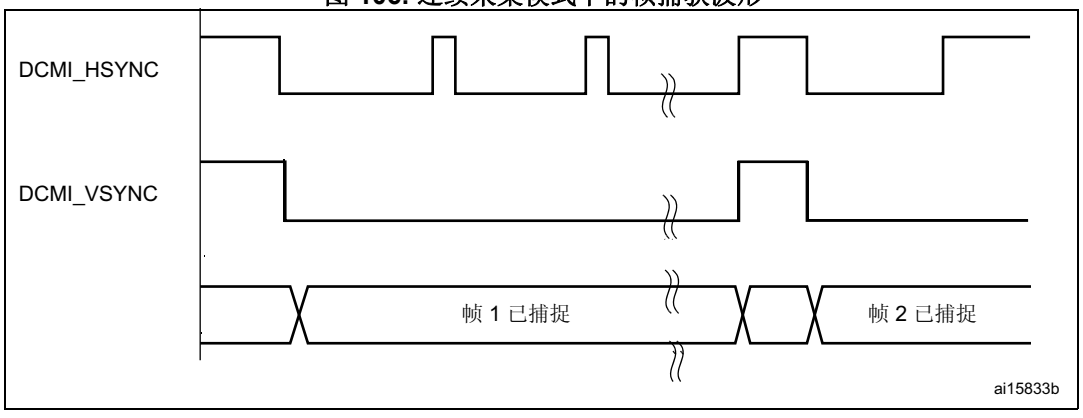


1. 此例中，DCMI_HSYNC 和 DCMI_VSYNC 的有效状态为 1。
2. DCMI_HSYNC 和 DCMI_VSYNC 的状态可同时发生更改。

连续采集模式

在此模式下（DCMI_CR 中的 CM 位 = “0”），一旦 DCMI_CR 中的 CAPTURE 位置 1，将在下一个 VSYNC 或内嵌同步码帧起始同步码时启动采集过程，具体取决于同步模式。该过程一直持续到 DCMI_CR 中的 CAPTURE 位清零。CAPTURE 位清零后，采集过程将持续到当前帧结束。

图 106. 连续采集模式下的帧捕获波形



1. 此例中，DCMI_HSYNC 和 DCMI_VSYNC 的有效状态为 1。
2. DCMI_HSYNC 和 DCMI_VSYNC 的状态可同时发生更改。

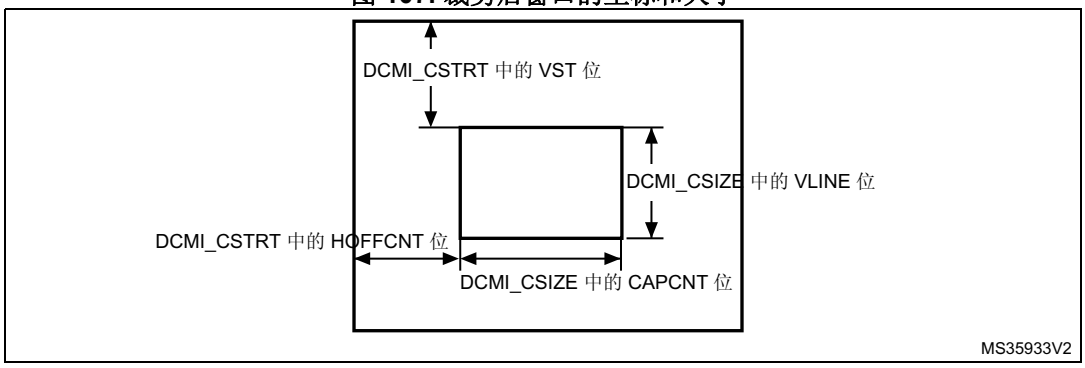
在连续采集模式下，可以通过配置 DCMI_CR 中的 FCRC 位来选择采集所有图片，或每两帧采集一次图片，或每四帧采集一张图片，以此降低帧捕获率。

注： *在硬件同步模式下（DCMI_CR 中的 ESS = “0”），即使 DCMI_CR 中的 CAPTURE = “0”，也将生成 IT_VSYNC 中断（如果使能）。因此为进一步降低帧捕获率，IT_VSYNC 中断可与快照模式结合使用，用来统计 2 次捕获之间的帧数。这并不适用于内嵌码同步模式。*

17.5.5 裁剪功能

摄像头接口可以使用裁剪功能从收到的图像中选择一个矩形窗口。起始（左上角）坐标和窗口大小（用像素时钟数表示的水平尺寸以及用行数表示的垂直尺寸）由两个 32 位寄存器（DCMI_CWSTRT 和 DCMI_CWSIZE）指定。窗口大小用像素时钟数（水平尺寸）和行数（垂直尺寸）表示。

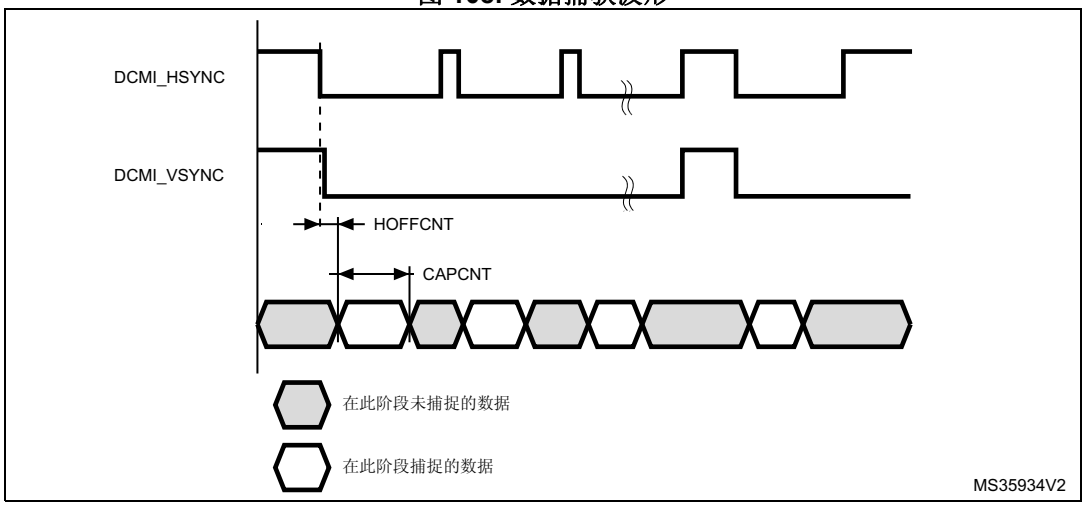
图 107. 裁剪后窗口的坐标和大小



这些寄存器将捕获窗口的起点坐标指定为某一行号（从 0 开始）和像素时钟数（从 0 开始），窗口大小则指定为行数和像素时钟数。CAPCNT 值只能是 4 的倍数（两个最低有效位强制为 0），才能通过 DMA 正确传输数据。

如果在捕获 DCMI_CWSIZE 寄存器中指定行数完成之前，VSYNC 信号已有效，那么捕获将停止，并且在中断使能时生成 IT_FRAME 中断。

图 108. 数据捕获波形



1. 此例中，DCMI_HSYNC 和 DCMI_VSYNC 的有效状态为 1。
2. DCMI_HSYNC 和 DCMI_VSYNC 的状态可同时发生更改。

17.5.6 JPEG 格式

要允许接收 JPEG 图像，必须将 DCMI_CR 寄存器中的 JPEG 位置 1。JPEG 图像不按行和帧存储，因此 VSYNC 信号用于启动捕获过程，而 HSYNC 则用作数据使能信号。行中包含的字节数可能不是 4 的倍数，因此处理此类情况时应十分谨慎，因为需要每次从捕获的数据形成一个完整的 32 位字时，才生成一个 DMA 请求。检测到帧结束并且尚未凑成 32 位字时，将使用“0”进行填充，并触发一个 DMA 请求。

裁剪功能和内嵌码同步不适用于 JPEG 格式。

17.5.7 FIFO

输入模式

为了对 AHB 上的数据传输率加以管理，在摄像头接口上实现了 4 个字深度的 FIFO。DCMI 配有一个简单的 FIFO 控制器，每次摄像头接口从 AHB 读取数据时读指针递增，每次摄像头接口向 FIFO 写入数据时写指针递增。因为没有溢出保护，如果数据传输率超过了 AHB 接口能够承受的速率，FIFO 中的数据就会被覆盖。

如果同步信号出错，或者 FIFO 发生溢出，FIFO 将复位，DCMI 接口将等待新的数据帧开始。

17.6 数据格式说明

17.6.1 数据格式

支持三种类型的数据：

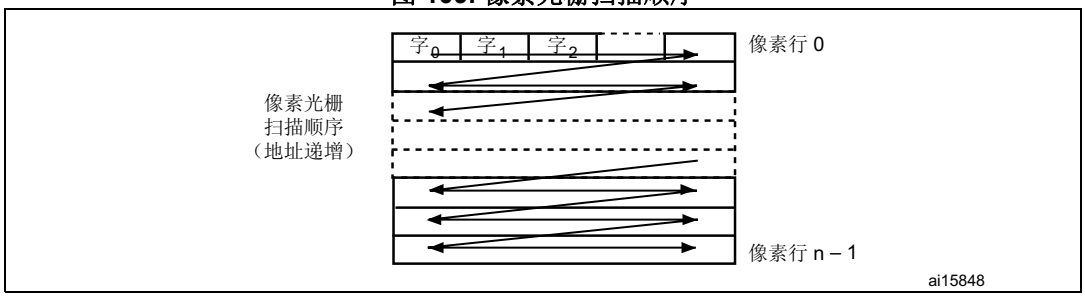
- 8 位逐行视频：单色或原始拜尔格式
- YCbCr 4:2:2 逐行视频
- RGB565 逐行视频。采用 16 位（5 位表示蓝色，5 位表示红色，6 位表示绿色）编码的像素需要两个时钟周期传输。

压缩数据：JPEG

对于 B&W、YCbCr 或 RGB 数据，最大输入大小为 2048 × 2048 像素。JPEG 压缩模式无限制。

对于单色、RGB 和 YCbCr，帧缓冲区以光栅模式存储。使用 32 位字。仅支持小端对齐格式。

图 109. 像素光栅扫描顺序



17.6.2 单色格式

特性：

- 光栅格式
- 每个像素 8 位

表 108 显示了数据的存储方式。

表 108. 单色逐行视频格式的数据存储

字节地址	31:24	23:16	15:8	7:0
0	n + 3	n + 2	n + 1	n
4	n + 7	n + 6	n + 5	n + 4

17.6.3 RGB 格式

特性：

- 光栅格式
- RGB
- 一个缓冲区交替存储 RGB 信号：BRGBRGBRG 等
- 对显示输出的优化

RGB 平面格式与标准的 OS 帧缓冲显示格式兼容。
仅支持 16 BPP（每个像素 16 位）：RGB565（每 32 位字表示 2 个像素）。
不支持 24 BPP（托盘化格式）和灰度格式。像素按照光栅扫描顺序进行存储，即从顶部像素行到底部像素行，从像素行的左侧到右侧。像素分量包括 R（红色）、G（绿色）和 B（蓝色）。所有分量的空间分辨率都相同（格式 4:4:4）。一帧数据中各个分量交替间隔存储。

表 109 显示了数据的存储方式。

表 109. 以 RGB 逐行视频格式存储数据

字节地址	31:27	26:21	20:16	15:11	10:5	4:0
0	红色 n + 1	绿色 n + 1	蓝色 n + 1	红色 n	绿色 n	蓝色 n
4	红色 n + 4	绿色 n + 3	蓝色 n + 3	红色 n + 2	绿色 n + 2	蓝色 n + 2

17.6.4 YCbCr 格式

特性：

- 光栅格式
- YCbCr 4:2:2
- 一个缓冲区交替存储 Y、Cb 和 Cr：CbYCrYCbYCr 等

像素分量包括 Y（亮度）、Cb 和 Cr（蓝色色度和红色色度）。每个分量都采用 8 位进行编码。亮度和色度（交替）存储在一起，如*表 110* 中所示。

表 110. YCbCr 逐行视频格式下的数据存储

字节地址	31:24	23:16	15:8	7:0
0	Y n + 1	Cr n	Y n	Cb n
4	Y n + 3	Cr n + 2	Y n + 2	Cb n + 2

17.6.5 YCbCr 格式——仅含 Y 分量

特性：

- 光栅格式
- YCbCr 4:2:2
- 缓冲区仅包含 Y 分量信息——单色图像

像素分量包括 Y（亮度）、Cb 和 Cr（蓝色色度和红色色度）。在此模式下，将丢弃色度信息。仅存储每个像素采用 8 位进行编码的亮度分量，如*表 111* 所示。

结果为单色图像，其分辨率与原始 YCbCr 数据的分辨率相同。

表 111. YCbCr 逐行视频格式下的数据存储——Y 分量提取模式

字节地址	31:24	23:16	15:8	7:0
0	Y n + 3	Y n + 2	Y n + 1	Y n
4	Y n + 7	Y n + 6	Y n + 5	Y n + 4

17.6.6 半分辨率图像提取

这是对先前接收模式的修改，适用于单色、RGB 或 Y 分量提取模式。

此模式仅允许存储半分辨率图像。可通过 OELS 和 LSM 控制位进行选择。

17.7 DCMI 中断

有五种中断源。所有中断都可通过软件屏蔽。全局中断 (IT_DCMI) 是所有单个中断的逻辑或运算所得结果。*表 112* 列出了所有中断。

表 112. DCMI 中断

中断名称	中断事件
IT_LINE	表示行结束
IT_FRAME	表示帧捕获结束
IT_OVR	表示接收的数据发生溢出错误
IT_VSYNC	表示同步帧
IT_ERR	表示内嵌码同步帧检测期间检测到错误
IT_DCMI	以上中断的逻辑或结果

17.8 DCMI 寄存器说明

所有 DCMI 寄存器都必须作为 32 位字访问，否则将发生总线错误。

17.8.1 DCMI 控制寄存器 (DCMI_CR)

DCMI control register

偏移地址：0x00

复位值：0x0000 0x0000

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OELS	LSM	OEBS	BSM		Res.	ENABLE	Res.	Res.	EDM		FCRC		VSPOL	HSPOL	PCKPOL	ESS	JPEG	CROP	CM	CAPTURE
											rw	rw	rw	rw	rw		rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:21 保留，必须保持复位值。

位 20 **OELS**：奇数/偶数行选择（行选择开始）(Odd/Even Line Select (Line Select Start))

此位与 LSM 字段 (LSM = 1) 一起使用

0：接口在帧开始后捕获第一行，同时丢弃第二行

1：接口在帧开始后捕获第二行，同时丢弃第一行

位 19 **LSM**：行选择模式 (Line Select mode)

0：接口捕获所有接收到的行

1：接口每两行捕获一行

位 18 **OEBS**：奇数/偶数字节选择（字节选择开始）(Odd/Even Byte Select (Byte Select Start))

此位与 BSM 字段 (BSM <> 00) 一起使用

0：接口从帧/行开始捕获第一个数据（字节或双字节），同时丢弃第二个字节

1：接口从帧/行开始捕获第二个数据（字节或双字节），同时丢弃第一个字节

位 17:16 **BSM[1:0]**：字节选择模式 (Byte Select mode)

00：接口捕获所有接收到的数据

01：接口对接收到的数据每隔一个字节进行捕获

10：接口每四个字节捕获一个字节

11：接口每四个字节捕获两个字节

注： 此模式仅在 EDM[1:0]=00 时有效。对于所有其它 EDM 值，必须将此位字段编程为复位值。

位 15 保留，必须保持复位值。

位 14 **ENABLE**：DCMI 使能 (DCMI enable)

0：禁止 DCMI

1：使能 DCMI

注： 使能此位之前，应对 DCMI 配置寄存器进行适当的设置

位 13:12 保留，必须保持复位值。

位 11:10 **EDM[1:0]**：扩展数据模式 (Extended data mode)

00：接口每个像素时钟捕获 8 位数据

01：接口每个像素时钟捕获 10 位数据

10：接口每个像素时钟捕获 12 位数据

11：接口每个像素时钟捕获 14 位数据

位 9:8 **FCRC[1:0]**：帧捕获率控制 (Frame capture rate control)
这些位定义了帧捕获频率。仅在连续采集模式下有效。快照模式下将被忽略。
00：捕获所有帧
01：每隔一帧捕获一次（带宽降低 50%）
10：每隔三帧捕获一次（带宽降低 75%）
11：保留

位 7 **VSPOL**：垂直同步极性 (Vertical synchronization polarity)
此位指示数据在并行接口上无效时 VSYNC 引脚的电平。
0：VSYNC 低电平有效
1：VSYNC 高电平有效

位 6 **HSPOL**：水平同步极性 (Horizontal synchronization polarity)
此位指示数据在并行接口上无效时 HSYNC 引脚的电平。
0：HSYNC 低电平有效
1：HSYNC 高电平有效

位 5 **PCKPOL**：像素时钟极性 (Pixel clock polarity)
此位用来配置像素时钟的捕获沿。
0：下降沿有效。
1：上升沿有效。

位 4 **ESS**：内嵌码同步选择 (Embedded synchronization select)
0：硬件同步，数据捕获（帧/行开始/停止）由 HSYNC/VSYNC 信号同步。
1：内嵌码同步，数据捕获由数据流中嵌入的同步码同步。
注：　仅对 8 位并行数据有效。ESS 位置 1 时，将忽略 HSPOL/VSPOL。
JPEG 模式下会禁止此位。

位 3 **JPEG**：JPEG 格式 (JPEG format)
0：未经压缩的视频格式
1：此位用于 JPEG 数据传输。HSYNC 信号用作数据使能信号。此模式下无法使用裁剪和内嵌码同步功能（ESS 位）。

位 2 **CROP**：裁剪功能 (Crop feature)
0：捕获完整图像。这种情况下，图像帧包含的字节总数应该为 4 的倍数
1：仅捕获剪裁寄存器所指定的窗口中的数据。如果窗口大小超出图片大小，则仅捕获图片大小。

位 1 **CM**：捕获模式 (Capture mode)
0：连续采集模式——收到的数据将通过 DMA 传输到目标存储区。缓冲区位置和模式（线性或循环缓冲区）由系统 DMA 控制。
1：快照模式（单帧）——一旦激活，接口将等待帧开始，然后通过 DMA 传输单帧。帧结束时将自动复位 CAPTURE 位。

位 0 **CAPTURE**：使能捕获 (Capture enable)
0：禁止捕获。
1：使能捕获。
摄像头接口等待第一帧开始，然后生成一个 DMA 请求以将收到的数据传输到目标存储器中。
在快照模式下，收到的第一帧结束时将自动使 CAPTURE 位清零。
在连续采集模式下，如果在执行捕获操作时通过软件将此位清零，则帧结束后此位的清零才生效。
注：　使能此位之前，应对 DMA 控制器和所有 DCMI 配置寄存器进行适当的编程。

17.8.2 DCMI 状态寄存器 (DCMI_SR)

DCMI status register

偏移地址：0x04

复位值：0x0000 0x0000

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FNEF	VSYNC	HSYNC
																													r	r	r

位 31:3 保留，必须保持复位值。

位 2 **FNE**：FIFO 非空 (FIFO not empty)

此位指示 FIFO 的状态

1：FIFO 包含有效数据

0：FIFO 为空

位 1 **VSYNC**

此位指示编程了适当极性的 VSYNC 引脚的状态。

使用内嵌码同步时，此位的含义如下：

0：有效帧

1：在帧之间同步

如果使用内嵌码同步，则仅当 DCMI_CR 中的 CAPTURE 位置 1 时，此位才有意义。

位 0 **HSYNC**

此位指示编程了适当极性的 HSYNC 引脚的状态。

使用内嵌码同步时，此位的含义如下：

0：有效行

1：在行之间同步

如果使用内嵌码同步，则仅当 DCMI_CR 中的 CAPTURE 位置 1 时，此位才有意义。

17.8.3 DCMI 原始中断状态寄存器 (DCMI_SR)

DCMI raw interrupt status register

偏移地址：0x08

复位值：0x0000 0x0000

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LINE_RIS	VSYNC_RIS	ERR_RIS	OVR_RIS	FRAME_RIS
																											r	r	r	r	r

DCMI_RIS 用来指示原始中断状态，以只读方式访问。执行读操作时，此寄存器返回的是对应中断未被 DCMI_IER 寄存器屏蔽的原始状态。

位 31:5 保留，必须保持复位值。

位 4 **LINE_RIS**：行原始中断状态 (Line raw interrupt status)
当 HSYNC 信号从无效状态更改为有效状态时，此位置 1。即使行无效也会变成高电平。
如果使用内嵌码同步，则仅当 DCMI_CR 中的 CAPTURE 位置 1 时，此位才置 1。
向 DCMI_ICR 中的 LINE_ISC 位写入“1”即可将此位清零。

位 3 **VSYNC_RIS**：VSYNC 原始中断状态 (VSYNC raw interrupt status)
当 VSYNC 信号从无效状态更改为有效状态时，此位置 1。
如果使用内嵌码同步，则仅当 DCMI_CR 中的 CAPTURE 位置 1 时，此位才置 1。
向 DCMI_ICR 中的 VSYNC_ISC 位写入“1”即可将此位清零。

位 2 **ERR_RIS**：同步错误原始中断状态 (Synchronization error raw interrupt status)
0：未检测到同步错误
1：未按正确顺序接收内嵌码同步字符。
此位仅在内嵌码同步模式下有效。向 DCMI_ICR 中的 ERR_ISC 位写入“1”即可将此位清零。
注： 此位仅适用于内嵌码同步模式。

位 1 **OVR_RIS**：溢出原始中断状态 (Overrun raw interrupt status)
0：未发生数据缓冲区溢出
1：发生数据缓冲区溢出，数据 FIFO 损坏。
向 DCMI_ICR 中的 OVR_ISC 位写入“1”即可将此位清零。

位 0 **FRAME_RIS**：捕获完成原始中断状态 (Capture complete raw interrupt status)
0：没有新的捕获数据
1：已捕获一帧。
对一帧数据或裁剪窗口内的数据捕获完毕后，此位置 1。
如果捕获裁剪窗口，则将在裁剪窗口最后一行的行结束时将此位置 1。即使捕获的帧为空（例如，窗口超出帧范围），此位也将置 1。
向 DCMI_ICR 中的 FRAME_ISC 位写入“1”即可将此位清零。

17.8.4 DCMI 中断使能寄存器 (DCMI_IER)

DCMI interrupt enable register

偏移地址：0x0C

复位值：0x0000 0x0000

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LINE_IE	VSYNC_IE	ERR_IE	OVR_IE	FRAME_IE
																											rw	rw	rw	rw	rw

DCMI_IER 寄存器用于使能中断。当其中任一位置 1 时，将使能相应中断。此寄存器可读写。

位 31:5 保留，必须保持复位值。

位 4 **LINE_IE**：使能行中断 (Line interrupt enable)
0：接收到行时不生成中断
1：完全接收到行时生成一个中断。

位 3 **VSYNC_IE**：VSYNC 中断使能 (VSYNC interrupt enable)
0：不生成中断
1：VSYNC 每次从无效电平变为有效电平时都生成一个中断。
VSYNC 信号的有效电平由 VSPOL 位定义。

位 2 **ERR_IE**：同步错误中断使能 (Synchronization error interrupt enable)
0：不生成中断
1：如果未按正确顺序接收内嵌码同步代码，则生成一个中断。
注： 此位仅适用于内嵌码同步模式。

位 1 **OVR_IE**：溢出中断使能 (Overrun interrupt enable)
0：不生成中断
1：如果 DMA 无法在收到新数据（32 位）之前传输上一个数据，则生成一个中断。

位 0 **FRAME_IE**：捕获完成中断使能 (Capture complete interrupt enable)
0：不生成中断
1：接收完成一帧数据或裁剪窗口内的数据（在裁剪模式下），生成一个中断。

17.8.5 DCMI 屏蔽中断状态寄存器 (DCMI_MIS)

DCMI masked interrupt status register

此 DCMI_MIS 寄存器是一个只读寄存器。执行读操作时，此寄存器将返回相应中断的当前屏蔽状态（取决于 DCMI_IER 中的值）。如果 DCMI_IER 中的相应使能位和 DCMI_RIS 中的相应位都置 1，则此寄存器中的对应位将置 1。

偏移地址：0x10

复位值：0x0000

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LINE_MIS	VSYNC_MIS	ERR_MIS	OVR_MIS	FRAME_MIS
																											r	r	r	r	r

位 31:5 保留，必须保持复位值。

位 4 **LINE_MIS**：行屏蔽中断状态 (Line masked interrupt status)
此位指示屏蔽行中断的状态
0：接收到行时不生成中断
1：收到完整一行数据且 DCMI_IER 中的 LINE_IE 位置 1 时生成一个中断。

位 3 **VSYNC_MIS**：VSYNC 屏蔽中断状态 (VSYNC masked interrupt status)
此位指示屏蔽 VSYNC 中断的状态
0：VSYNC 电平跳变时不生成中断
1：VSYNC 每次从无效电平变为有效电平且 DCMI_IER 中的 VSYNC_IE 位置 1 时都生成一个中断。
VSYNC 信号的有效电平由 VSPOL 位定义。

位 2 **ERR_MIS**：同步错误屏蔽中断状态 (Synchronization error masked interrupt status)
此位指示屏蔽同步错误中断
0：发生同步错误时不生成中断
1：如果未按正确顺序接收内嵌同步码且 DCMI_IER 中的 ERR_IE 位置 1，则生成一个中断。
注： 此位仅适用于内嵌码同步模式。

位 1 **OVR_MIS**：溢出屏蔽中断状态 (Overrun masked interrupt status)
此位指示屏蔽溢出中断的状态
0：发生溢出时不生成中断
1：如果 DMA 无法在收到新数据（32 位）之前传输上一个数据且 DCMI_IER 中的 OVR_IE 位置 1，则生成一个中断。

位 0 **FRAME_MIS**：捕获完成屏蔽中断状态 (Capture complete masked interrupt status)
此位指示屏蔽捕获完成中断的状态
0：完成捕获后不生成中断
1：接收完成一帧数据或裁剪窗口内的数据（在裁剪模式下），且 DCMI_IER 中的 FRAME_IE 位置 1 时生成一个中断。

17.8.6 DCMI 中断清零寄存器 (DCMI_ICR)

DCMI interrupt clear register

偏移地址：0x14

复位值：0x0000 0x0000

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LINE_ISC	VSYNC_ISC	ERR_ISC	OVR_ISC	FRAME_ISC
																											w	w	w	w	w

DCMI_ICR 寄存器为只写寄存器。向此寄存器的某一位写入“1”可将 DCMI_RIS 和 DCMI_MIS 寄存器中的相应位清零。写入“0”则不会带来任何影响。

位 15:5 保留，必须保持复位值。

位 4 **LINE_ISC**：线中断状态清零 (line interrupt status clear)
在此位中写入“1”可将 DCMI_RIS 寄存器中的 LINE_RIS 清零

位 3 **VSYNC_ISC**：垂直同步中断状态清零 (Vertical synch interrupt status clear)
在此位中写入“1”可将 DCMI_RIS 中的 VSYNC_RIS 位清零

位 2 **ERR_ISC**：同步错误中断状态清零 (Synchronization error interrupt status clear)
在此位中写入“1”可将 DCMI_RIS 中的 ERR_RIS 位清零
注： 此位仅适用于内嵌码同步模式。

位 1 **OVR_ISC**：溢出中断状态清零 (Overrun interrupt status clear)
在此位中写入“1”可将 DCMI_RIS 中的 OVR_RIS 位清零

位 0 **FRAME_ISC**：捕获完成中断状态清零 (Capture complete interrupt status clear)
在此位中写入“1”可将 DCMI_RIS 中的 FRAME_RIS 位清零

17.8.7 DCMI 内嵌同步码寄存器 (DCMI_ESCR)

DCMI embedded synchronization code register

偏移地址：0x18

复位值：0x0000 0x0000

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
FEC								LEC								LSC								FSC							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **FEC**：帧结束分隔码 (Frame end delimiter code)

此字节指定帧结束分隔码。由 4 个字节组成，格式为 0xFF 0x00 0x00 FEC。

如果将 FEC 编程为 0xFF，所有未使用的其它代码 (0xFF0000XY) 都将视为帧结束分隔符。

位 23:16 **LEC**：行结束分隔码 (Line end delimiter code)

此字节指定行结束分隔码。由 4 个字节组成，格式为 0xFF 0x00 0x00 LEC。

位 15:8 **LSC**：行开始分隔码 (Line start delimiter code)

此字节指定行开始分隔码。由 4 个字节组成，格式为 0xFF 0x00 0x00 LSC。

位 7:0 **FSC**：帧开始分隔码 (Frame start delimiter code)

此字节指定帧开始分隔码。由 4 个字节组成，格式为 0xFF 0x00 0x00 FSC。

如果将 FSC 编程为 0xFF，将检测不到任何帧开始分隔符。但在 FEC 代码后第一次出现 LSC 时将视为帧分隔符的开始。

17.8.8 DCMI 内嵌码同步取消屏蔽寄存器 (DCMI_ESUR)

DCMI embedded synchronization unmask register

偏移地址：0x1C

复位值：0x0000 0x0000

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
FEU								LEU								LSU								FSU							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **FEU**：帧结束分隔符取消屏蔽 (Frame end delimiter unmask)

此字节指定对帧结束分隔码的屏蔽。

0：将帧结束分隔符与收到的数据进行比较时，将屏蔽 DCMI_ESCR 中 FEC 字节中的相应位

1：将帧结束分隔符与收到的数据进行比较时，将比较 DCMI_ESCR 中 FEC 字节中的相应位

位 23:16 **LEU**：行结束分隔符取消屏蔽 (Line end delimiter unmask)

此字节指定对行结束分隔码的屏蔽。

0：将行结束分隔符与收到的数据进行比较时，将屏蔽 DCMI_ESCR 中 LEC 字节中的相应位

1：将行结束分隔符与收到的数据进行比较时，将比较 DCMI_ESCR 中 LEC 字节中的相应位

位 15:8 **LSU**：行开始分隔符取消屏蔽 (Line start delimiter unmask)

此字节指定对行开始分隔码的屏蔽。

0：将行开始分隔符与收到的数据进行比较时，将屏蔽 DCMI_ESCR 中 LSC 字节中的相应位

1：将行开始分隔符与收到的数据进行比较时，将比较 DCMI_ESCR 中 LSC 字节中的相应位

位 7:0 **FSU**：帧开始分隔符取消屏蔽 (Frame start delimiter unmask)

此字节指定对帧开始分隔码的屏蔽。

0：将帧开始分隔符与收到的数据进行比较时，将屏蔽 DCMI_ESCR 中 FSC 字节中的相应位

1：将帧开始分隔符与收到的数据进行比较时，将比较 DCMI_ESCR 中 FSC 字节中的相应位

17.8.9 DCMI 裁剪窗口起点 (DCMI_CWSTRT)

DCMI crop window start

偏移地址：0x20

复位值：0x0000 0x0000

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
Res.	Res.	Res.	VST[12:0]													Res.	Res.	HOFFCNT[13:0]													
			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:29 保留，必须保持复位值。

位 28:16 **VST[12:0]**：窗口开始的行数 (Vertical start line count)

图像捕获从此行开始。对之前行的数据不予捕获。

0x0000 => 行 1

0x0001 => 行 2

0x0002 => 行 3

....

位 15:14 保留，必须保持复位值。

位 13:0 **HOFFCNT[13:0]**：窗口开始的水平方向位移 (Horizontal offset count)

窗口行内，每行在捕获数据前需空出的像素时钟个数。

17.8.10 DCMI 裁剪窗口大小 (DCMI_CWSIZE)

CMI crop window size

偏移地址：0x24

复位值：0x0000 0x0000

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
Res.	Res.	VLINE13:0]														Res.	Res.	CAPCNT[13:0]													
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:30 保留，必须保持复位值。

位 29:16 **VLINE[13:0]**：垂直行计数 (Vertical line count)

窗口内包含的行数。

0x0000 => 1 行

0x0001 => 2 行

0x0002 => 3 行

....

位 15:14 保留，必须保持复位值。

位 13:0 **CAPCNT[13:0]**：捕获计数 (Capture count)

窗口内要捕获的像素时钟数。其值应与表示不同并行接口宽度的字对齐数据相对应。

0x0000 => 1 个像素

0x0001 => 2 个像素

0x0002 => 3 个像素

....

17.8.11 DCMI 数据寄存器 (DCMI_DR)

DCMI data register

偏移地址：0x28

复位值：0x0000 0x0000

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
Byte3								Byte2								Byte1								Byte0							
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:24 数据字节 3

位 23:16 数据字节 2

位 15:8 数据字节 1

位 7:0 数据字节 0

数字摄像头接口每收到 32 位数据，才触发一次 DMA 请求。4 字深度的 FIFO 可为 DMA 传输留出足够时间并避免出现 DMA 溢出情况。

17.8.12 DCMI 寄存器映射

表 113 汇总了 DCMI 寄存器。

表 113. DCMI 寄存器地址映射和复位值

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	DCMI_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OELS	LSM	OEBS	BSM		Res.	ENABLE	Res.	Res.	EDM		FCRC		VSPOL	HSPOL	PCKPOL	ESS	JPEG	CROP	CM	CAPTURE
	Reset value												0	0	0	0	0		0			0	0	0	0	0	0	0	0	0	0	0	0
0x04	DCMI_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.	FNE	VSYNC	HSYNC
	Reset value																														0	0	0

表 113. DCMI 寄存器地址映射和复位值（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	DCMI_RIS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LINE_RIS	VSYNC_RIS	ERR_RIS	OVR_RIS	FRAME_RIS
	Reset value																												0	0	0	0	0
0x0C	DCMI_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.	LINE_IE	VSYNC_IE	ERR_IE	OVR_IE	FRAME_IE
	Reset value																												0	0	0	0	0
0x10	DCMI_MIS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LINE_MIS	VSYNC_MIS	ERR_MIS	OVR_MIS	FRAME_MIS
	Reset value																												0	0	0	0	0
0x14	DCMI_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.	LINE_ISC	VSYNC_ISC	ERR_ISC	OVR_ISC	FRAME_ISC
	Reset value																												0	0	0	0	0
0x18	DCMI_ESCR	FEC								LEC								LSC								FSC							
	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	DCMI_ESUR	FEU								LEU								LSU								FSU							
	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	DCMI_CWSTRT	Res.	Res.	Res.	VST[12:0]													Res.	Res.	HOFFCNT[13:0]													
	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
0x24	DCMI_CWSIZE	Res.	Res.	VLINE13:0]														Res.	Res.	CAPCNT[13:0]													
	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
0x28	DCMI_DR	Byte3								Byte2								Byte1								Byte0							
	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 64 页上的第 2.2.2 节*。

18 LCD-TFT 控制器 (LTDC)

18.1 简介

LCD-TFT（液晶显示器——薄膜晶体管）显示器控制器提供并行数字 RGB（红色、绿色、蓝色）以及水平同步、垂直同步、像素时钟和数据使能信号，这些信号直接输出到不同 LCD 和 TFT 面板的接口。

18.2 LTDC 主要特性

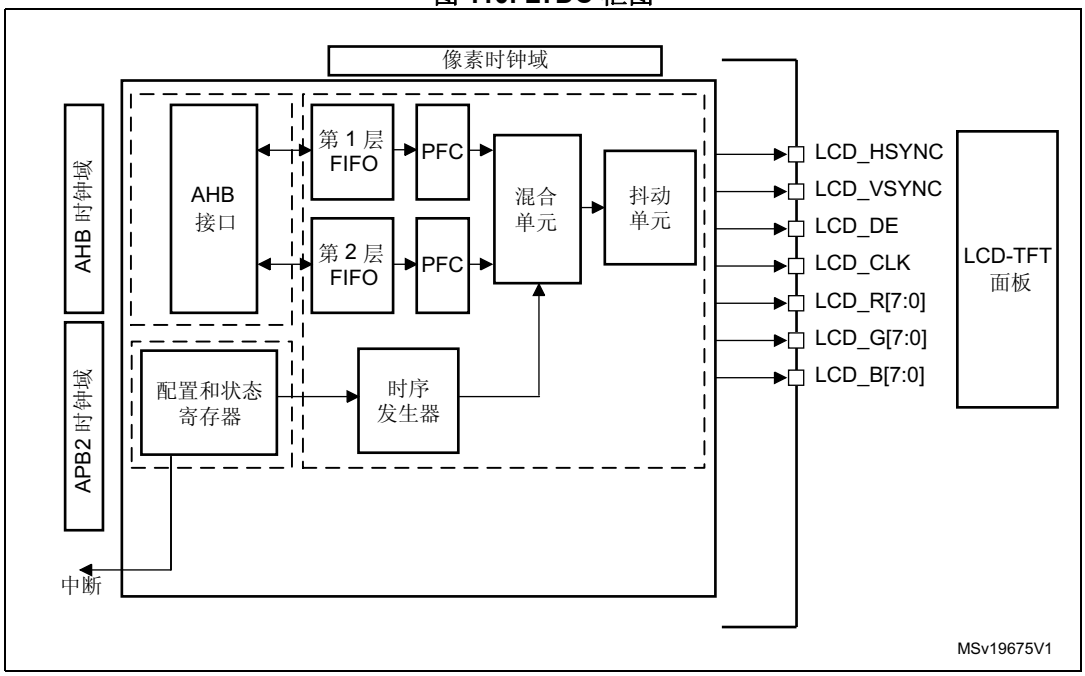
- 24 位 RGB 并行像素输出；每像素 8 位 (RGB888)
- 2 个带有专用 FIFO 的显示层（FIFO 深度 64x32 位）
- 查色表 (CLUT)，每层高达 256 种颜色（256x24 位）
- 支持高达 XGA (1024x768) 的分辨率
- 可针对不同显示面板编程时序
- 可编程背景色
- 可编程 HSync、VSync 和数据使能信号的极性
- 每层有多达 8 个输入颜色格式可供选择
  - ARGB8888
  - RGB888
  - RGB565
  - ARGB1555
  - ARGB4444
  - L8（8 位 Luminance 或 CLUT）
  - AL44（4 位 alpha + 4 位 luminance）
  - AL88（8 位 alpha + 8 位 luminance）
- 每通道的低位采用伪随机抖动输出
  - 红色、绿色、蓝色的抖动宽度为 2 位
- 使用 alpha 值（每像素或常数）在两层之间灵活混合
- 色键（透明颜色）
- 可编程窗口位置和大小
- 支持薄膜晶体管 (TFT) 彩色显示器
- AHB 主接口支持 16 个字的突发
- 高达 4 个可编程中断事件

18.3 LTDC 功能说明

18.3.1 LTDC 框图

LTDC 的框图如*图 110：LTDC 框图*所示。

图 110. LTDC 框图



层 FIFO：每层一个 64x32 位 FIFO。

PFC：执行像素格式转换的像素格式转换器，从层的所选输入像素格式转换为字。

AHB 接口：用于将数据从存储器传输到 FIFO。

有关混合单元、抖动单元和时序发生器的信息：请参见*第 18.4.1 节*和*第 18.4.2 节*。

18.3.2 LTDC 复位和时钟

LCD-TFT 控制器外设使用 3 个时钟域：

- AHB 时钟域 (HCLK)：用于将数据从存储器传输到 FIFO 层
- APB2 时钟域 (PCLK2)：用于配置寄存器
- 像素时钟域 (LCD_CLK)：用于生成 LCD-TFT 接口信号。LCD_CLK 输出应按照面板要求配置。LCD_CLK 通过 PLLSAI 进行配置（见 RCC 部分）

通过将 RCC_APB2RSTR 寄存器中的相应位置 1 可将 LCD 控制器复位。这将复位三个时钟域。

18.3.3 LCD-TFT 引脚和信号接口

下表汇总了 LTDC 信号接口：

表 114. LCD-TFT 引脚和信号接口

LCD-TFT 信号	I/O	说明
LCD_CLK	O	时钟输出
LCD_HSYNC	O	水平同步
LCD_VSYNC	O	垂直同步
LCD_DE	O	数据使能
LCD_R[7:0]	O	数据：8 位红色数据
LCD_G[7:0]	O	数据：8 位绿色数据
LCD_B[7:0]	O	数据：8 位蓝色数据

必须通过用户程序配置 LCD-TFT 控制器引脚。未使用的引脚可用于其他功能。

对于高达 24 位 (RGB888) 的 LTDC 输出，如果使用低于 8bpp 的像素深度将 RGB565 或 RGB666 输出到 16 位或 18 位显示器，则 RGB 显示数据线必须连接到 LCD-TFT 控制器 RGB 数据线的 MSB。例如，当 LCD-TFT 控制器与 RGB565 16 位显示器相连时，LCD 显示器的 R[4:0]、G[5:0] 和 B[4:0] 数据线引脚必须连接至 LCD-TFT 控制器的 LCD_R[7:3]、LCD_G[7:2] 和 LCD_B[7:3]。

18.4 LTDC 可编程参数

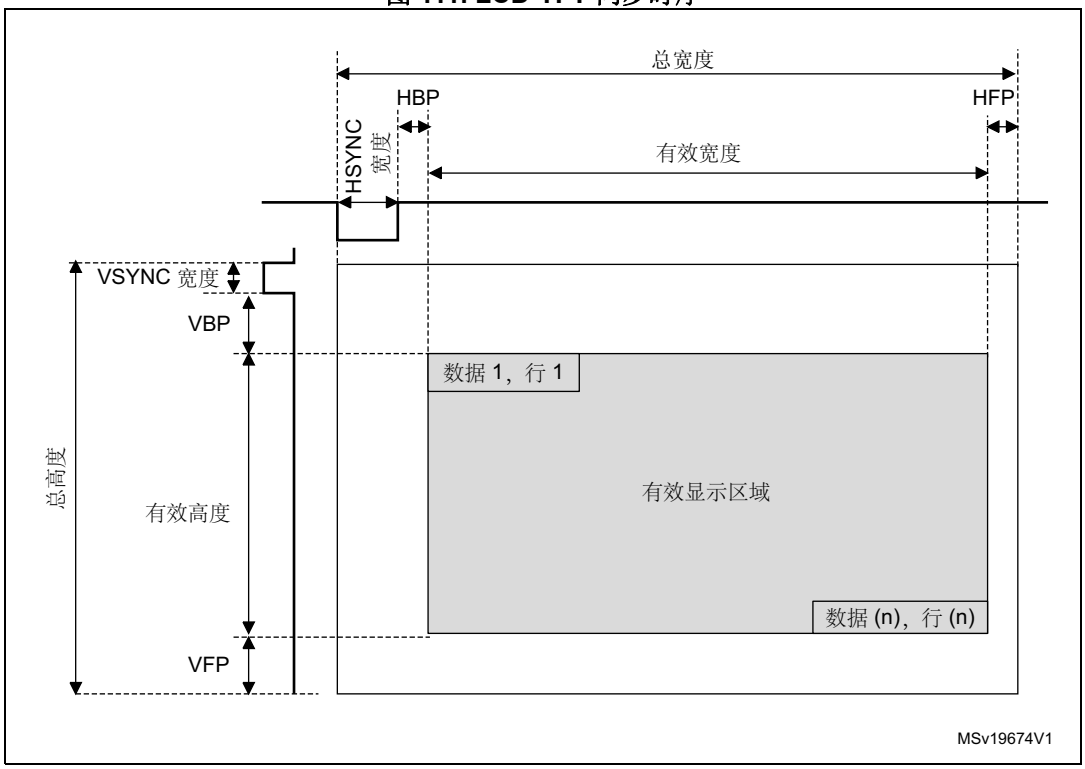
LCD-TFT 控制器提供灵活的可配置参数。其可通过 **LTDC_GCR** 寄存器使能或禁止。

18.4.1 LTDC 全局配置参数

同步时序：

图 111 显示了框图*图 110* 中所示的同步时序发生器模块生成的可配置时序参数。该模块生成水平和垂直同步时序面板信号、像素时钟和数据使能信号。

图 111. LCD-TFT 同步时序



注： *HBP 和 HFP 分别为水平后沿周期和水平前沿周期。*

VBP 和 VFP 分别为垂直后沿周期和垂直前沿周期。

LCD-TFT 可编程同步时序包括：

- HSYNC 和 VSYNC 宽度：水平和垂直同步宽度，通过编程 **LTDC_SSCR** 寄存器中的 ***HSYNC Width - 1*** 和 ***VSYNC Width - 1*** 的值进行配置。
- HBP 和 VBP：水平和垂直同步后沿宽度，通过编程 **LTDC_BPCR** 寄存器中的累加值 ***HSYNC Width + HBP - 1*** 和累加值 ***VSYNC Width + VBP - 1*** 进行配置。
- 有效宽度和有效高度：有效宽度和有效高度通过编程 **LTDC_AWCR** 寄存器中的累加值 ***HSYNC Width + HBP + Active Width - 1*** 和累加值 ***VSYNC Width + VBP + Active Height - 1*** 进行配置（仅支持最高 1024x768）。
- 总宽度：总宽度通过编程 **LTDC_TWCR** 寄存器中的累加值 ***HSYNC Width + HBP + Active Width + HFP - 1*** 进行配置。HFP 为水平前沿周期。
- 总高度：总高度通过编程 **LTDC_TWCR** 寄存器中的累加值 ***VSYNC Height + VBP + Active Height + VFP - 1*** 进行配置。VFP 为垂直前沿周期。

注： *使能 LTDC 时，产生的时序以 X/Y=0/0 位置作为垂直同步区域中的第一个水平同步像素，随后是后沿、有效数据显示区域和前沿。*

禁止 LTDC 时，时序发生器模块复位为 X= *总宽度 - 1*、Y= *总高度- 1*，并在垂直同步阶段和 FIFO 刷新前保持上一个像素。因此，仅连续输出消隐数据。

同步时序配置示例：

TFT-LCD 时序（应从面板数据手册中提取）：

- 水平和垂直同步宽度：0x8 像素，0x4 行
- 水平和垂直后沿：0x7 像素，0x2 行
- 有效宽度和有效高度：0x280 像素，0x1E0 行 (640x480)
- 水平前沿：0x6 像素
- 垂直前沿：0x2 行

LTDC 时序寄存器中编程的值将为：

- **LTDC_SSCR** 寄存器：将编程为 0x00070001。（HSW[11:0] 为 0x7 且 VSH[10:0] 为 0x3）
- **LTDC_BPCR** 寄存器：将编程为 0x000E0005。（AHBP[11:0] 为 0xE(0x8 + 0x6) 且 AVBP[10:0] 为 0x5(0x4 + 0x1)）
- **LTDC_AWCR** 寄存器：将编程为 0x028E01E5。（AAW[11:0] 为 0x28E(0x8 + 0x7 + 0x27F) 且 AAH[10:0] 为 0x1E5(0x4 + 0x2 + 0x1DF)）
- **LTDC_TWCR** 寄存器：将编程为 0x00000294。（TOTALW[11:0] 为 0x294(0x8 + 0x7 + 0x280 + 0x5)）
- **LTDC_THCR** 寄存器：将编程为 0x000001E7。（TOTALH[10:0] 为 0x1E7(0x4 + 0x2 + 0x1E0 + 1)）

可编程极性

水平和垂直同步、数据使能和像素时钟输出信号的极性可通过 **LTDC_GCR** 寄存器编程为高电平有效或低电平有效。

背景色

恒定的背景色 (RGB888) 可通过 **LTDC_BCCR** 寄存器编程。它用于与底层混合。

抖动

使用 LFSR 的伪随机抖动技术用于向各个像素颜色通道值（R、G或B）添加小的随机值（阈值），从而当 18 位显示器上显示 24 位数据时，可在某些情况下对 MSB 进行舍入操作。因此，抖动技术用于对各帧中不同的数据进行舍入操作。

伪随机抖动技术的过程是将 LSB 与阈值比较，并在 LSB 部分 >= 阈值时，仅向 MSB 部分加 1。一旦应用抖动技术，通常会减少 LSB。

添加的伪随机值的宽度为每个颜色通道 2 位；红色 2 位、绿色 2 位及蓝色 2 位。

使能 LCD-TFT 控制器后，LFSR 以第一个有效像素开始运行，并且即使在消隐周期内和抖动关闭时也保持运行状态。如果禁止 LTDC，LFSR 将复位。

可通过 **LTDC_GCR** 寄存器实时开启和关闭抖动。

重载影子寄存器

一些配置寄存器执行影子操作。对活动寄存器执行写操作时，或在 **LTDC_SRCR** 寄存器配置阶段之后的垂直消隐周期开始时，可将影子寄存器值立即重载到活动寄存器中。如果选择了立即重载配置，则只应在所有新寄存器完成写操作后激活重载。

不应在重载完成前再次修改影子寄存器。读取影子寄存器将返回实际有效值。新写入的值只能在重载发生后读取。

如果在 **LTDC_IER** 寄存器中相应使能，则可产生寄存器重载中断。

影子寄存器均为第 1 层和第 2 层寄存器，但 **LTDC_LxCLUTWR** 寄存器除外。

中断产生事件

有关中断配置，请参见*第 18.5 节：LTDC 中断*。

18.4.2 层可编程参数

最多可单独使能、禁止和配置两个层。层显示顺序固定，即自下而上。如果使能两个层，则层 2 为顶部显示窗口。

窗口

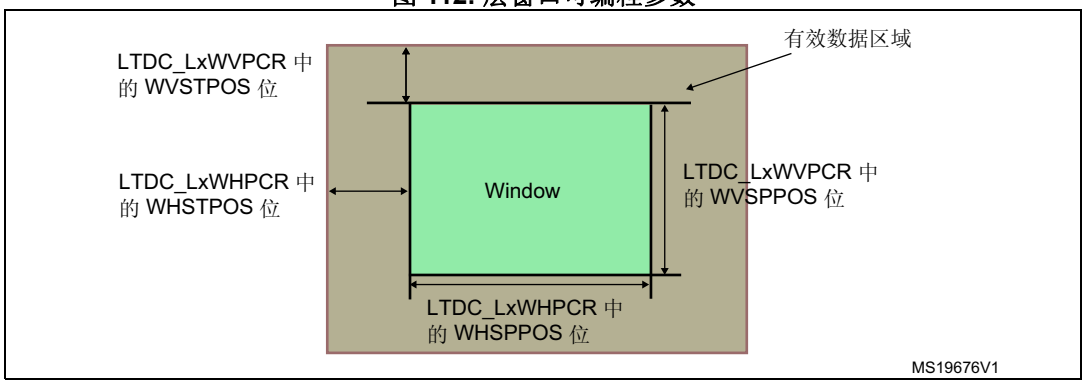
可为每个层定位和调整大小，各个层必须位于有效显示区域内。

窗口位置和大小通过左上和右下的 X/Y 位置以及包含同步、后沿大小和有效数据区域的内部时序发生器配置。请参见 **LTDC_LxWHPCR** 和 **LTDC_WVPCR** 寄存器。

可编程层位置和大小定义了一行中的第一个/最后一个可见像素和窗口中的第一个/最后一个可见行。它允许显示完整的图像帧，也允许只显示图像帧的一部分。请参见*图 112*。

- 层中的第一个和最后一个可见像素通过配置 **LTDC_LxWHPCR** 寄存器中的 WHSTPOS[11:0] 和 WHSPPOS[11:0] 进行设置。
- 层中的第一个和最后一个可见行通过配置 **LTDC_LxWVPCR** 寄存器中的 WVSTPOS[10:0] 和 WVSPPOS[10:0] 进行设置。

图 112. 层窗口可编程参数



像素输入格式

可编程像素格式用于层的帧缓冲区中存储的数据。

可通过 **LTDC_LxPFCR** 寄存器为每个层配置多达 8 个输入像素格式。

像素数据从帧缓冲区中读取，随后按照以下方式转换为内部 8888 (ARGB) 格式：

- 宽度低于 8 位的分量通过位重复扩展到 8 位。所选位范围多次拼接，直至其超过 8 位。在得到的向量中，选择高 8 位。例如：5 位 RGB565 红色通道将变为（位位置）：43210432（低 3 位由 5 位中的高 3 位填充）

下图说明了像素数据映射与所选格式的关系。

表 115. 像素数据映射与颜色格式的关系

ARGB8888			
@+3 A_x[7:0]	@+2 R_x[7:0]	@+1 G_x[7:0]	@ B_x[7:0]
@+7 A_{x+1}[7:0]	@+6 R_{x+1}[7:0]	@+5 G_{x+1}[7:0]	@+4 B_{x+1}[7:0]
RGB888			
@+3 B_{x+1}[7:0]	@+2 R_x[7:0]	@+1 G_x[7:0]	@ B_x[7:0]
@+7 G_{x+2}[7:0]	@+6 B_{x+2}[7:0]	@+5 R_{x+1}[7:0]	@+4 G_{x+1}[7:0]
RGB565			
@+3 R_{x+1}[4:0] G_{x+1}[5:3]	@+2 G_{x+1}[2:0] B_{x+1}[4:0]	@+1 R_x[4:0] G_x[5:3]	@ G_x[2:0] B_x[4:0]
@+7 R_{x+3}[4:0] G_{x+3}[5:3]	@+6 G_{x+3}[2:0] B_{x+3}[4:0]	@+5 R_{x+2}[4:0] G_{x+2}[5:3]	@+4 G_{x+2}[2:0] B_{x+2}[4:0]
ARGB1555			
@+3 A_{x+1}[0]R_{x+1}[4:0] G_{x+1}[4:3]	@+2 G_{x+1}[2:0] B_{x+1}[4:0]	@+1 A_x[0] R_x[4:0] G_x[4:3]	@ G_x[2:0] B_x[4:0]
@+7 A_{x+3}[0]R_{x+3}[4:0] G_{x+3}[4:3]	@+6 G_{x+3}[2:0] B_{x+3}[4:0]	@+5 A_{x+2}[0]R_{x+2}[4:0]G_{x+2}[4:3]	@+4 G_{x+2}[2:0] B_{x+2}[4:0]
ARGB4444			
@+3 A_{x+1}[3:0]R_{x+1}[3:0]	@+2 G_{x+1}[3:0] B_{x+1}[3:0]	@+1 A_x[3:0] R_x[3:0]	@ G_x[3:0] B_x[3:0]
@+7 A_{x+3}[3:0]R_{x+3}[3:0]	@+6 G_{x+3}[3:0] B_{x+3}[3:0]	@+5 A_{x+2}[3:0]R_{x+2}[3:0]	@+4 G_{x+2}[3:0] B_{x+2}[3:0]

表 115. 像素数据映射与颜色格式的关系（续）

ARGB8888			
L8			
@+3 L_{x+3}[7:0]	@+2 L_{x+2}[7:0]	@+1 L_{x+1}[7:0]	@ L_x[7:0]
@+7 L_{x+7}[7:0]	@+6 L_{x+6}[7:0]	@+5 L_{x+5}[7:0]	@+4 L_{x+4}[7:0]
AL44			
@+3 A_{x+3}[3:0] L_{x+3}[3:0]	@+2 A_{x+2}[3:0] L_{x+2}[3:0]	@+1 A_{x+1}[3:0] L_{x+1}[3:0]	@ A_x[3:0] L_x[3:0]
@+7 A_{x+7}[3:0] L_{x+7}[3:0]	@+6 A_{x+6}[3:0] L_{x+6}[3:0]	@+5 A_{x+5}[3:0] L_{x+5}[3:0]	@+4 A_{x+4}[3:0] L_{x+4}[3:0]
AL88			
@+3 A_{x+1}[7:0]	@+2 L_{x+1}[7:0]	@+1 A_x[7:0]	@ L_x[7:0]
@+7 A_{x+3}[7:0]	@+6 L_{x+3}[7:0]	@+5 A_{x+2}[7:0]	@+4 L_{x+2}[7:0]

查色表 (CLUT)

可在运行时通过 **LTDC_LxCR** 寄存器为每个层使能 CLUT，CLUT 仅在使用 L8、AL44 和 AL88 输入像素格式时适用于索引色。

首先，CLUT 必须加载用于替换相应像素（索引色）的原始 R、G 和 B 值的 R、G 和 B 值。每个颜色（RGB 值）在 CLUT 内都有自己对应的地址。

R、G 和 B 值及其各自的地址均通过 **LTDC_LxCLUTWR** 寄存器编程。

- 在使用 L8 和 AL88 输入像素格式时，CLUT 必须加载 256 个颜色。各颜色的地址在 **LTDC_LxCLUTWR** 寄存器的 CLUTADD 位中配置。
- 在使用 AL44 输入像素格式时，CLUT 必须仅加载 16 个颜色。各颜色的地址必须通过将 4 位 L 通道重复为 8 位进行填充，具体如下：
  - L0（索引色 0），地址 0x00 处
  - L1，地址 0x11 处
  - L2，地址 0x22 处
  - .....
  - L15，地址 0xFF 处

颜色帧缓冲区地址

每个层的颜色帧缓冲区均有一个起始地址，该地址通过 **LTDC_LxCFBAR** 寄存器进行配置。

当使能某个层时，将从颜色帧缓冲区中获取该数据。

颜色帧缓冲区长度

每层均设置颜色帧缓冲区的总行长（单位为字节）和行数，二者可分别通过 **LTDC_LxCFBLR** 和 **LTDC_LxCFBLNR** 寄存器进行配置。

行长和行数的设置用于阻止超出帧缓存结尾的数据被预取到层对应的 FIFO 中。

- 如果设置为低于所需字节，则会产生 FIFO 下溢中断（如果之前已使能）。
- 如果设置为高于实际所需字节，则将丢弃从 FIFO 中读取的无用数据。无用数据不会显示。

颜色帧缓冲区间距

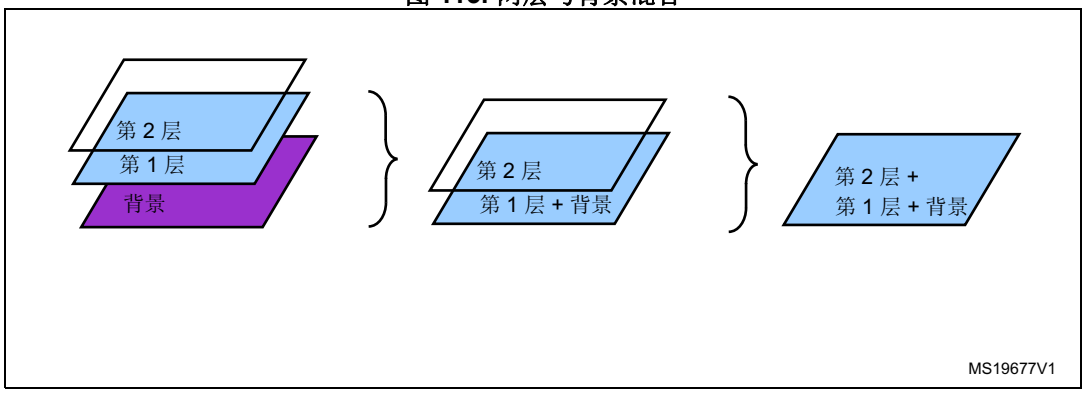
每层的颜色帧缓冲区均具有可配置间距，此间距是一行的开始与下一行开始的距离（以字节为单位）。它通过 **LTDC_LxCFBLR** 寄存器配置。

层混合

混合操作始终有效，两层可按照 **LTDC_LxBFCR** 寄存器中配置的混合系数进行混合。

混合顺序固定，即由下至上。如果使能了两层，首先第 1 层将与背景色混合，随后第 2 层与第 1 层和背景的混合颜色结果再次混合。请参见*图 113*。

图 113. 两层与背景混合



默认颜色

每层可具有 ARGB 格式的默认颜色，该颜色在定义的层窗口外使用或在层禁止时使用。

默认颜色通过 **LTDC_LxDCCR** 寄存器配置。

始终在两层间执行混合操作，即便其中一层禁止也是如此。要避免层禁止时显示默认颜色，需将 LTDC_LxBFCR 寄存器中此层的混合系数设置为其复位值。

色键

色键 (RGB) 可配置为代表透明像素。

使能色键后，当前像素（格式转换后、混合前的像素）将与色键进行比较。如果当前像素与编程的 RGB 值相匹配，则该像素的所有通道 (ARGB) 均设置为 0。

运行时，可配置色键值并用其替换像素 RGB 值。

色键通过 **LTDC_LxCKCR** 寄存器配置。

18.5 LTDC 中断

LTDC 提供四个可屏蔽中断，这些中断经逻辑或运算后产生两个中断向量。

中断源可通过 **LRDC_IER** 寄存器单独使能或禁止。将相应的屏蔽位置 1 可使能相应中断。

发生如下事件时会产生两个中断：

- 行中断：达到编程的行时产生。行中断的位置在 LTDC_LIPCR 寄存器中编程
- 寄存器重载中断：在垂直消隐周期内执行影子寄存器重载时产生
- FIFO 下溢中断：从空层 FIFO 中请求像素时产生
- 传输错误中断：数据传输期间出现 AHB 总线错误时产生

这些中断事件与 NVIC 控制器相连，如下图所示。

图 114. 中断事件

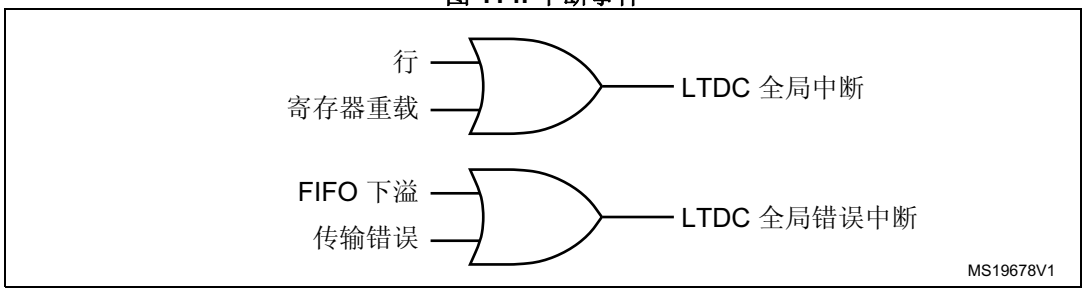


表 116. LTDC 中断请求

中断事件	事件标志	使能控制位
行	LIF	LIE
寄存器重载	RRIF	RRIEN
FIFO 下溢	FUDERRIF	FUDERRIE
传输错误	TERRIF	TERRIE

18.6 LTDC 编程步骤

- 在 RCC 寄存器中使能 LTDC 时钟。
- 按照面板数据表配置所需像素时钟。
- 按照*第 18.4.1 节：LTDC 全局配置参数*中所述的面板数据表配置同步时序：VSYNC、HSYNC、垂直和水平后沿、有效数据区域以及前沿时序。
- 配置 **LTDC_GCR** 寄存器中的同步信号和时钟极性。
- 必要时，配置 **LTDC_BCCR** 寄存器中的背景色。
- 配置 **LTDC_IER** 和 **LTDC_LIPCR** 寄存器中的所需中断。
- 通过执行以下编程操作配置第 1/2 层的参数：
  - 编程 **LTDC_LxWHPCR** 和 **LTDC_WVPCR** 寄存器中的层窗口的水平和垂直位置。层窗口必须位于有效数据区域
  - 编程 **LTDC_LxPFCR** 寄存器中的像素输入格式
  - 编程 **LTDC_LxCFBAR** 寄存器中的颜色帧起始地址
  - 编程 **LTDC_LxCFBLR** 寄存器中的颜色帧缓冲区的行长和间距
  - 编程 **LTDC_LxCFBLNR** 寄存器中的颜色帧缓冲区的行数
  - 必要时，在 **LTDC_LxCLUTWR** 寄存器中为 CLUT 加载 RGB 值及其地址
  - 必要时，分别在 **LTDC_LxDCCR** 和 **LTDC_LxBFCR** 寄存器中配置默认颜色和混合系数
- 使能 **LTDC_LxCR** 寄存器中的第 1/2 层，必要时使能 CLUT。
- 必要时，可分别在 **LTDC_GCR 和 LTDC_LxCKCR** 寄存器中使能抖动和色键。也可以实时使能这两个功能。
- 通过 **LTDC_SRCR** 寄存器将影子寄存器重载到活动寄存器中。
- 使能 **LTDC_GCR** 寄存器中的 LCD-TFT 控制器。
- 除 CLUT 外，所有层参数均可实时修改。新配置必须通过配置 **LTDC_SRCR** 寄存器立即重载或在垂直消隐周期内重载。

注： *所有层的寄存器均执行影子操作。一旦对某个寄存器执行写操作，便不应在重载完成前再次进行修改。因此，如果在尚未重载时对同一寄存器执行新的写操作，则将覆盖之前的配置。*

18.7 LTDC 寄存器

18.7.1 LTDC 同步大小配置寄存器 (LTDC_SSCR)

LTDC Synchronization Size Configuration Register

此寄存器定义水平同步像素数减 1 以及垂直同步行数减 1。有关配置的示例，请参见*图 111*和*第 18.4 节：LTDC 可编程参数*。

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.	HSW											
				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.	VSH										
					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 **HSW[11:0]**：水平同步宽度 (Horizontal Synchronization Width)（以像素时钟周期为单位）
这些位定义水平同步像素数减 1。

位 15:11 保留，必须保持复位值

位 10:0 **VSH[10:0]**：垂直同步高度 (Vertical Synchronization Height)（以水平扫描行为单位）
这些位定义垂直同步高度减 1。它代表水平同步行的数量。

18.7.2 LTDC 后沿配置寄存器 (LTDC_BPCR)

LTDC Back Porch Configuration Register

此寄存器定义水平同步像素加水平后沿像素的累加数减 1（**HSYNC 宽度 + HBP- 1**）以及垂直同步行加垂直后沿行的累加数减 1（**VSYNC 高度 + VBP - 1**）。有关配置的示例，请参见*图 111*和*第 18.4 节：LTDC 可编程参数*。

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.	AHBP											
				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.	AVBP										
					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 **AHBP[11:0]**：累加水平后沿 (Accumulated Horizontal back porch)（以像素时钟周期为单位）
这些位定义累加水平后沿宽度（水平同步像素加水平后沿像素减 1）。
水平后沿是水平同步信号变为无效到下一扫描行的有效显示开始之间的间隔。

位 15:11 保留，必须保持复位值

位 10:0 **AVBP[10:0]**：累加垂直后沿 (Accumulated Vertical back porch)（以水平扫描行为单位）
这些位定义累加垂直后沿宽度（垂直同步行加垂直后沿行减 1）。
垂直后沿是帧开始到下一帧的首个有效扫描行开始所包含的水平扫描行的数量。

18.7.3 LTDC 有效宽度配置寄存器 (LTDC_AWCR)

LTDC Active Width Configuration Register

此寄存器定义水平同步像素加水平后沿像素加有效像素的累加数减 1（**HSYNC 宽度 + HBP + 有效宽度 - 1**）以及垂直同步行加垂直后沿行加有效行的累加数减 1（**VSYNC 高度+ BVBP + 有效高度 - 1**）。有关配置的示例，请参见*图 111* 和*第 18.4 节：LTDC 可编程参数*。

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.	AAW											
				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.	AAH										
					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 **AAW[11:0]**：累加有效宽度 (Accumulated Active Width)（以像素时钟周期为单位）
这些位定义累加有效宽度（水平同步像素加水平后沿像素加有效像素减 1）。
有效宽度是面板扫描行的有效显示区中的像素数。支持的最大有效宽度为 0x400。

位 15:11 保留，必须保持复位值

位 10:0 **AAH[10:0]**：累加有效高度 (Accumulated Active Height)（以水平扫描行为单位）
这些位定义累加高度（垂直同步行加垂直后沿行加有效高度行减 1）。有效高度是面板中的有效行数。支持的最大有效高度为 0x300。

18.7.4 LTDC 总宽度配置寄存器 (LTDC_TWCR)

LTDC Total Width Configuration Register

此寄存器定义水平同步像素加水平后沿像素加有效像素加水平前沿像素的累加数减 1（**HSYNC 宽度 + HBP + 有效宽度 + HFP - 1**）以及垂直同步行加垂直后沿行加有效行加垂直前沿行的累加数减 1（**VSYNC 高度 + BVBP + 有效高度 + VFP - 1**）。有关配置的示例，请参见*图 111*和*第 18.4 节：LTDC 可编程参数*。

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.	TOTALW											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
16	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	TOTALH										
					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 **TOTALW[11:0]**：总宽度 (Total Width)（以像素时钟周期为单位）
这些位定义累加总宽度（水平同步像素加水平后沿像素加有效宽度加水平前沿像素减 1）。

位 15:11 保留，必须保持复位值

位 10:0 **TOTALH[10:0]**：总高度 (Total Height)（以水平扫描行为单位）
这些位定义累加高度（垂直同步行加垂直后沿行加有效高度加垂直前沿行减 1）。

18.7.5 LTDC 全局控制寄存器 (LTDC_GCR)

LTDC Global Control Register

此寄存器定义 LCD-TFT 控制器的全局配置。

偏移地址：0x18

复位值：0x0000 2220

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HSPOL	VSPOL	DEPOL	PCPOL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DEN
rw	rw	rw	rw												rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	DRW			Res.	DGW			Res.	DBW			Res.	Res.	Res.	LTDCEN
	r	r	r		r	r	r		r	r	r				rw

位 31 **HSPOL**：水平同步极性 (Horizontal Synchronization Polarity)
此位由软件置 1 和清零。
0：水平同步极性低电平有效
1：水平同步极性高电平有效

位 30 **VSPOL**：垂直同步极性 (Vertical Synchronization Polarity)
此位由软件置 1 和清零。
0：垂直同步低电平有效
1：垂直同步高电平有效

位 29 **DEPOL**：数据使能极性 (Data Enable Polarity)
此位由软件置 1 和清零。
0：数据使能极性低电平有效
1：数据使能极性高电平有效

位 28 **PCPOL**：像素时钟极性 (Pixel Clock Polarity)
此位由软件置 1 和清零。
0：输入像素时钟
1：反相输入像素时钟

位 27:17 保留，必须保持复位值

位 16 **DEN**：抖动使能 (Dither Enable)
此位由软件置 1 和清零。
0：禁止抖动
1：使能抖动

位 15 保留，必须保持复位值

位 14:12 **DRW[2:0]**：抖动红色宽度 (Dither Red Width)
这些位返回抖动红色位

位 11 保留，必须保持复位值

位 10:8 **DGW[2:0]**：抖动绿色宽度 (Dither Green Width)
这些位返回抖动绿色位

位 7 保留，必须保持复位值

位 6:4 **DBW[2:0]**：抖动蓝色宽度 (Dither Blue Width)
这些位返回抖动蓝色位

位 3:1 保留，必须保持复位值

位 0 **LTDCEN**：LCD-TFT 控制器使能位 (LCD-TFT controller enable bit)
此位由软件置 1 和清零。
0：禁止 LTDC
1：使能 LTDC

18.7.6 LTDC 影子重载配置寄存器 (LTDC_SRCR)

LTDC Shadow Reload Configuration Register

此寄存器允许立即或在垂直消隐周期内将影子寄存器的值重载到活动寄存器中。影子寄存器均为第 1 层和第 2 层寄存器，但 LTDC_L1CLUTWR 和 LTDC_L2CLUTWR 除外。

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.	Res.	Res.	Res.	Res.	Res.	VBR	IMR
														rw	rw

位 31:2 保留，必须保持复位值

位 1 **VBR**：垂直消隐重载 (Vertical Blanking Reload)

此位由软件置 1，只有重载后才由硬件清零。（一旦置 1，便无法通过寄存器写操作清零）

0：无影响

1：影子寄存器在垂直消隐周期（有效显示区后的第一行开始时）内重载

位 0 **IMR**：立即重载 (Immediate Reload)

此位由软件置 1，只有重载后才由硬件清零。

0：无影响

1：影子寄存器立即重载

注： *影子寄存器回读有效值。直至重载完成，才会读取“旧”值。*

18.7.7 LTDC 背景色配置寄存器 (LTDC_BCCR)

LTDC Background Color Configuration Register

此寄存器定义背景色 (RGB888)。

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.	BCRED							
								rw	rw	rw	rw	rw	rw	rw	rw

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

位 31:24 保留，必须保持复位值

位 23:16 **BCRED[7:0]**：背景红色值 (Background Color Red value)
这些位配置背景红色值

位 15:8 **BCGREEN[7:0]**：背景绿色值 (Background Color Green value)
这些位配置背景绿色值

位 7:0 **BCBLUE[7:0]**：背景蓝色值 (Background Color Blue value)
这些位配置背景蓝色值

18.7.8 LTDC 中断使能寄存器 (LTDC_IER)

LTDC Interrupt Enable Register

此寄存器通过将对应的位置 1 来确定哪一个状态标志位产生中断请求。

偏移地址：0x34

复位值：0x0000 0000

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

位 31:4 保留，必须保持复位值

位 3 **RRIE**：寄存器重载中断使能 (Register Reload interrupt enable)
此位由软件置 1 和清零
0：禁止寄存器重载中断
1：使能寄存器重载中断

位 2 **TERRIE**：传输错误中断使能 (Transfer Error Interrupt Enable)
此位由软件置 1 和清零
0：禁止传输错误中断
1：使能传输错误中断

位 1 **FUIE**：FIFO 下溢中断使能 (FIFO Underrun Interrupt Enable)
此位由软件置 1 和清零
0：禁止 FIFO 下溢中断
1：使能 FIFO 下溢中断

位 0 **LIE**：行中断使能 (Line Interrupt Enable)
此位由软件置 1 和清零
0：禁止行中断
1：使能行中断

18.7.9 LTDC 中断状态寄存器 (LTDC_ISR)

LTDC Interrupt Status Register

此寄存器返回中断状态标志

偏移地址：0x38

复位值：0x0000 0000

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

位 31:24 保留，必须保持复位值

位 3 **RRIF**：寄存器重载中断标志 (Register Reload Interrupt Flag)
- 0：未产生寄存器重载中断
- 1：发生垂直消隐重载时（以及到达有效区域后的第一行时）产生寄存器重载中断

位 2 **TERRIF**：传输错误中断标志 (Transfer Error interrupt flag)
- 0：未产生传输错误中断
- 1：出现总线错误时产生传输错误中断

位 1 **FUIF**：FIFO 下溢中断标志 (FIFO Underrun Interrupt flag)
- 0：未产生 FIFO 下溢中断
- 1：当其中一个层 FIFO 为空并从 FIFO 读取像素数据时，将产生 FIFO 下溢中断

位 0 **LIF**：行中断标志 (Line Interrupt flag)
- 0：未产生行中断
- 1：到达编程的行时产生行中断

18.7.10 LTDC 中断清零寄存器 (LTDC_ICR)

LTDC Interrupt Clear 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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.	CRRIF	CTERRIF	CFUIF	CLIF
												w	w	w	w

位 31:24 保留，必须保持复位值

位 3 **CRRIF**：寄存器重载中断清零标志 (Clears Register Reload Interrupt Flag)
0：无影响
1：将 LTDC_ISR 寄存器中的 RRIF 标志清零

位 2 **CTERRIF**：传输错误中断清零标志 (Clears the Transfer Error Interrupt Flag)
0：无影响
1：将 LTDC_ISR 寄存器中的 TERRIF 标志清零

位 1 **CFUIF**：FIFO 下溢中断清零标志 (Clears the FIFO Underrun Interrupt flag)
0：无影响
1：将 LTDC_ISR 寄存器中的 FUDERRIF 标志清零

位 0 **CLIF**：行中断清零标志 (Clears the Line Interrupt Flag)
0：无影响
1：将 LTDC_ISR 寄存器中的 LIF 标志清零

18.7.11 LTDC 行中断位置配置寄存器 (LTDC_LIPCR)

LTDC Line Interrupt Position Configuration Register

此寄存器定义行中断的位置。要编程的行值取决于时序参数。请参见*图 111*。

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.	LIPOS										
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:11 保留，必须保持复位值

位 10:0 **LIPOS[10:0]**：行中断位置 (Line Interrupt Position)
这些位配置行中断位置

18.7.12 LTDC 当前位置状态寄存器 (LTDC_CPSR)

LTDC Current Position Status 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
CXPOS															
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
CYPOS															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16： **CXPOS[15:0]**：当前 X 位置 (Current X Position)
这些位返回当前 X 位置

位 15:0 **CYPOS[15:0]**：当前 Y 位置 (Current Y Position)
这些位返回当前 Y 位置

18.7.13 LTDC 当前显示状态寄存器 (LTDC_CDSR)

LTDC Current Display Status Register

此寄存器返回由 HSYNC、VSYNC 和水平/垂直 DE 信号控制的当前显示阶段的状态。

示例：如果当前显示阶段为垂直同步阶段，则 VSYNCS 位置 1（高电平有效）。示例：如果当前显示阶段为水平同步阶段，则 HSYNCS 位高电平有效。

偏移地址：0x48

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.	HSYNCS	VSYNCS	HDES	VDES
												r	r	r	r

位 31:24 保留，必须保持复位值

位 3 **HSYNCS**：水平同步显示状态 (Horizontal Synchronization display Status)
0：低电平有效
1：高电平有效

位 2 **VSYNCS**：垂直同步显示状态 (Vertical Synchronization display Status)
0：低电平有效
1：高电平有效

位 1 **HDES**：水平数据使能显示状态 (Horizontal Data Enable display Status)
0：低电平有效
1：高电平有效

位 0 **VDES**：垂直数据使能显示状态 (Vertical Data Enable display Status)
0：低电平有效
1：高电平有效

注： *返回的状态不取决于* ***LTDC_GCR*** *寄存器中配置的极性，而是返回当前的有效显示阶段。*

18.7.14 LTDC 第 x 层控制寄存器 (LTDC_LxCR)（其中，x = 1..2）

LTDC Layerx Control Register

偏移地址：0x84 + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0000

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

位 31:5 保留，必须保持复位值

位 4 **CLUTEN**：查色表使能 (Color Look-Up Table Enable)
此位由软件置 1 和清零。
0：禁止查色表
1：使能查色表
CLUT 仅对 L8、AL44 和 AL88 像素格式有意义。请参见*第 479 页上的查色表 (CLUT)*。

位 3 保留，必须保持复位值

位 2 保留，必须保持复位值

位 1 **COLKEN**：色键使能 (Color Keying Enable)
此位由软件置 1 和清零。
0：禁止色键
1：使能色键

位 0 **LEN**：层使能 (Layer Enable)
此位由软件置 1 和清零。
0：禁止层
1：使能层

18.7.15 LTDC 第 x 层窗口水平位置配置寄存器 (LTDC_LxWHPCR)（其中 x=1..2）

LTDC Layerx Window Horizontal Position Configuration Register

此寄存器定义第 1 层或第 2 层窗口的水平位置（第一个和最后一个像素）。

一行的第一个可见像素是在 **LTDC_BPCR** 寄存器中编程的 ***AHBP[10:0] bits + 1*** 的值。

一行的最后一个可见像素是在 **LTDC_AWCR** 寄存器中编程的 ***AAW[10:0] bits*** 的值。此范围内的所有值均为允许值。

偏移地址：0x88 + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	WHSPPOS											
				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.	WHSTPOS											
				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 **WHSPPOS[11:0]**：窗口水平结束位置 (Window Horizontal Stop Position)
这些位配置层窗口的一行的最后一个可见像素。

位 15:12 保留，必须保持复位值

位 11:0 **WHSTPOS[11:0]**：窗口水平起始位置 (Window Horizontal Start Position)
这些位配置层窗口的一行的第一个可见像素。

示例：

LTDC_BPCR 寄存器配置为 0x000E0005（AHBP[11:0] 为 0xE），LTDC_AWCR 寄存器配置为 0x028E01E5（AAW[11:0] 为 0x28E）。要配置大小为 630x460 的窗口的水平位置（有效数据区域中的水平起始偏移为 5 个像素）。

1. 层窗口的第一个像素：WHSTPOS[11:0] 应编程为 0x14 (0xE+1+0x5)
2. 层窗口的最后一个像素：WHSPPOS[11:0] 应编程为 0x28A

18.7.16 LTDC 第 x 层窗口垂直位置配置寄存器 (LTDC_LxWVPCR)（其中 x=1..2）

LTDC Layerx Window Vertical Position Configuration Register

此寄存器定义第 1 层或第 2 层窗口的垂直位置（第一行或最后一行）。

一个帧的第一个可见行是在 **LTDC_BPCR** 寄存器中编程的 ***AVBP[10:0] bits + 1*** 的值。

一个帧的最后一个可见行是在 **LTDC_AWCR** 寄存器中编程的 ***AAH[10:0] bits*** 的值。此范围内的所有值均为允许值。

偏移地址：0x8C + 0x80 x (*第 x 层*-1)，*x* = 1 或 2

复位值：0x0000 0000

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

位 31:27 保留，必须保持复位值

位 26:16 **WVSPPOS[10:0]**：窗口垂直结束位置 (Window Vertical Stop Position)
这些位配置层窗口的最后一个可见行。

位 15:11 保留，必须保持复位值

位 10:0 **WVSTPOS[10:0]**：窗口垂直起始位置 (Window Vertical Start Position)
这些位配置层窗口的第一个可见行。

示例：

LTDC_BPCR 寄存器配置为 0x000E0005（AVBP[10:0] 为 0x5），LTDC_AWCR 寄存器配置为 0x028E01E5（AAH[10:0] 为 0x1E5）。要配置大小为 630x460 的窗口的垂直位置（有效数据区域中的垂直起始偏移为 8 行）：

1. 层窗口的第一行：WVSTPOS[10:0] 应编程为 0xE (0x5 + 1 + 0x8)
2. 层窗口的最后一行：WVSPPOS[10:0] 应编程为 0x1DA

18.7.17 LTDC 第 x 层色键配置寄存器 (LTDC_LxCKCR)（其中 x=1..2）

LTDC Layerx Color Keying Configuration Register

此寄存器定义色键使用的色键值 (RGB)。

偏移地址：0x90 + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0000

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

位 31:24 保留，必须保持复位值

位 23:16 **CKRED[7:0]**：色键红色值 (Color Key Red value)

位 15:8 **CKGREEN[7:0]**：色键绿色值 (Color Key Green value)

位 7:0 **CKBLUE[7:0]**：色键蓝色值 (Color Key Blue value)

18.7.18 LTDC 第 x 层像素格式配置寄存器 (LTDC_LxPFCR)（其中 x=1..2）

LTDC Layerx Pixel Format Configuration Register

此寄存器定义层的帧缓冲区中存储的数据所使用的像素格式。像素数据从帧缓冲区读取，随后转换为内部格式 8888 (ARGB)。

偏移地址：0x94 + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0000

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

位 31:3 保留，必须保持复位值

位 2:0 **PF[2:0]**：像素格式 (Pixel Format)
这些位配置像素格式
000：ARGB8888
001：RGB888
010：RGB565
011：ARGB1555
100：ARGB4444
101：L8（8 位 Luminance）
110：AL44（4 位 Alpha，4 位 Luminance）
111：AL88（8 位 Alpha，8 位 Luminance）

18.7.19 LTDC 第 x 层常数 Alpha 配置寄存器 (LTDC_LxCACR)（其中 x=1..2）

LTDC Layerx Constant Alpha Configuration Register

此寄存器定义在 alpha 混合中使用的常数 alpha 值（由硬件实现 255 等分）。请参见 LTDC_LxBFCR 寄存器。

偏移地址：0x98 + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：(第 x 层 - 1) 0x0000 00FF

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

位 31:8 保留，必须保持复位值

位 7:0 **CONSTA[7:0]**：常数 Alpha (Constant Alpha)
这些位配置混合时使用的常数 Alpha。常数 Alpha 由硬件实现 255 等分。
示例：如果编程的常数 Alpha 为 0xFF，则常数 Alpha 值为 255/255=1

18.7.20 LTDC 第 x 层默认颜色配置寄存器 (LTDC_LxDCCR)（其中 x=1..2）

LTDC Layerx Default Color Configuration Register

此寄存器定义采用 ARGB 格式的层的默认颜色。默认颜色在定义的层窗口外使用或在层禁止时使用。复位值 0x00000000 定义了透明黑色。

偏移地址：0x9C + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DCALPHA								DCRED							
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
DCGREEN								DCBLUE							
rw								rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **DCALPHA[7:0]**：默认 Alpha 值 (Default Color Alpha)
这些位配置默认 alpha 值

位 23:16 **DCRED[7:0]**：默认颜色红色 (Default Color Red)
这些位配置默认红色值

位 15:8 **DCGREEN[7:0]**：默认颜色绿色 (Default Color Green)
这些位配置默认绿色值

位 7:0 **DCBLUE[7:0]**：默认颜色蓝色 (Default Color Blue)
这些位配置默认蓝色值

18.7.21 LTDC 第 x 层混合系数配置寄存器 (LTDC_LxBFCR)（其中 x=1..2）

LTDC Layerx Blending Factors Configuration Register

此寄存器定义混合系数 F1 和 F2。

通用混合公式为：BC = BF1 x C + BF2 x Cs

- BC = 混合后的颜色
- BF1 = 混合系数 1
- C = 当前层颜色
- BF2 = 混合系数 2
- Cs = 底层混合后的颜色

偏移地址：0xA0 + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0607

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	BF1			Res.	Res.	Res.	Res.	Res.	BF2		
					rw	rw	rw						rw	rw	rw

位 31:11 保留，必须保持复位值

位 10:8 **BF1[2:0]**：混合系数 1 (Blending Factor 1)
这些位选择混合系数 F1
000：保留
001：保留
010：保留
011：保留
100：常数 Alpha
101：保留
110：像素 Alpha x 常数 Alpha
111：保留

位 7:3 保留，必须保持复位值

位 2:0 **BF2[2:0]**：混合系数 2 (Blending Factor 2)
这些位选择混合系数 F2
000：保留
001：保留
010：保留
011：保留
100：保留
101：1——常数 Alpha
110：保留
111：1——（像素 Alpha x 常数 Alpha）

注： *常数 Alpha 值是在寄存器中编程的值，由硬件实现 255 等分。*

示例：仅使能第 1 层，BF1 配置为常数 Alpha

BF2 配置为 1—— 常数 Alpha

常数 Alpha：在 LxCACR 寄存器中编程的常数 Alpha 为 240 (0xF0)。因此，常数 Alpha 值为 240/255 = 0.94

C：当前层颜色为 128

Cs：背景色为 48

第 1 层与背景色混合。

BC = 常数 Alpha x C + (1 - 常数 Alpha) x Cs = *0.94 x 128 + (1- 0.94) x 48 = 123。*

18.7.22 LTDC 第 x 层颜色帧缓冲区地址寄存器 (LTDC_LxCFBAR)（其中 x=1..2）

LTDC Layerx Color Frame Buffer Address Register

此寄存器定义颜色帧缓冲区的起始地址，该地址必须指向帧缓冲区中存储的层的左上角像素的像素数据地址。

偏移地址：0xAC + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0000

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

位 31:0 **CFBADD[31:0]**：颜色帧缓冲区起始地址 (Color Frame Buffer Start Address)
这些位定义颜色帧缓冲区的起始地址。

18.7.23 LTDC 第 x 层颜色帧缓冲区长度寄存器 (LTDC_LxCFBLR)（其中 x=1..2）

LTDC Layerx Color Frame Buffer Length Register

此寄存器定义颜色帧缓冲区的行长和行间距。

偏移地址：0xB0 + 0x80 (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	CFBP												
			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.	CFBLL												
			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 **CFBP[17:0]**：颜色帧缓冲区间距（以字节为单位）(Color Frame Buffer Pitch in bytes)
这些位定义从像素某行的起始处到下一行的起始处的增量（以字节为单位）。

位 15:13 保留，必须保持复位值

位 12:0 **CFBLL[12:0]**：颜色帧缓冲区行长 (Color Frame Buffer Line Length)
这些位定义一行像素的长度（以字节为单位）+ 3。
行长的计算方法为：有效宽度 x 每像素的字节数 + 3。

示例：

- 采用 RGB565（每像素 2 个字节）格式且宽度为 256 像素的帧缓冲区（每行总字节数为 256x2=512）需要向此寄存器写入值 0x02000203（其中，间距 = 行长）。
- 采用 RGB888（每像素 3 个字节）格式且宽度为 320 像素的帧缓冲区（每行总字节数为 320x3=960）需要向此寄存器写入值 0x03C003C3（其中，间距 = 行长）。

18.7.24 LTDC 第 x 层颜色帧缓冲区行数寄存器 (LTDC_LxCFBLNR)（其中 x=1..2）

LTDC Layerx ColorFrame Buffer Line Number Register

此寄存器定义颜色帧缓冲区中的行数。

偏移地址：0xB4 + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0000

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

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

位 31:11 保留，必须保持复位值

位 10:0 **CFBLNBR[10:0]**：帧缓冲区行数 (Frame Buffer Line Number)
这些位用来定义缓冲区中的行数，行数跟有效高度对应。

注： *行数和行长设置定义针对每层从每个帧中获取的数据量。如果配置为低于所需字节，则会产生 FIFO 下溢中断（如果使能）。*
另一方面，起始地址和间距设置定义存储器中每行的正确起始位置。

18.7.25 LTDC 第 x 层 CLUT 写寄存器 (LTDC_LxCLUTWR) （其中 x=1..2）

LTDC Layerx CLUT Write Register

此寄存器定义 CLUT 地址和 RGB 值。

偏移地址：0xC4 + 0x80 x (*第 x 层* - 1)，*x* = 1 或 2

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLUTADD								RED							
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
GREEN								BLUE							
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位 31:24 **CLUTADD[7:0]**：CLUT 地址 (CLUT Address)
这些位配置每个 RGB 值的 CLUT 地址（CLUT 内的颜色位置）

位 23:16 **RED[7:0]**：红色值 (Red value)
这些位配置红色值

位 15:8 **GREEN[7:0]**：绿色值 (Green value)
这些位配置绿色值

位 7:0 **BLUE[7:0]**：蓝色值 (Blue value)
这些位配置蓝色值

注： *CLUT 写寄存器只应在消隐周期内或在层禁止时配置。CLUT 可通过 **LTDC_LxCR** 寄存器使能和禁止。*

CLUT 仅对 L8、AL44 和 AL88 像素格式有意义。

18.7.26 LTDC 寄存器映射

下表对 LTDC 寄存器进行了汇总。有关 LTDC 寄存器的基本地址，请参见寄存器边界地址表。

表 117. LTDC 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
0x0008	LTDC_SSCR	Res.	Res.	Res.	Res.	Res.	Res.	HSW[11:0]										Res.	Res.	Res.	Res.	Res.	VSH[10:0]										
	Reset value							0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0
0x000C	LTDC_BPCR	Res.	Res.	Res.	Res.	Res.	Res.	AHBP[11:0]										Res.	Res.	Res.	Res.	Res.	AVBP[10:0]										
	Reset value							0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0
0x0010	LTDC_AWCR	Res.	Res.	Res.	Res.	Res.	Res.	AAV[11:0]										Res.	Res.	Res.	Res.	Res.	AAH[10:0]										
	Reset value							0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0
0x0014	LTDC_TWCR	Res.	Res.	Res.	Res.	TOTALW[11:0]												Res.	Res.	Res.	Res.	Res.	TOTALH[10:0]										
	Reset value							0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0
0x0018	LTDC_GCR	HSPOL	VSPOL	DEPOL	PCPOL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DEN	Res.	Res.	DRW[2:0]		Res.	DGW[2:0]			Res.	Res.	DBW[2:0]			Res.	Res.	Res.	LTDCEN
	Reset value	0	0	0	0												0		0	1	0		0	1	0			0	1	0				0
0x0024	LTDC_SRCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VBR	IMR
	Reset value																															0	0
0x002C	LTDC_BCCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BCRED[7:0]								Res.	Res.	BCGREEN[7:0]								BCBLUE[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

表 117. LTDC 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
0x0034	LTDC_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.	RRIE	TERRIE	FUIE	LIE
	Reset value																													0	0	0	0
0x0038	LTDC_ISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RRIF	TERRIF	FUIF	LIF
	Reset value																													0	0	0	0
0x003C	LTDC_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.	CRRIF	CTERRIF	CFUIF	CLIF
	Reset value																													0	0	0	0
0x0040	LTDC_LIPCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BCRED[10:0]										
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0x0044	LTDC_CPSR	CXPOS[15:0]																CYPOS[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
0x0048	LTDC_CDSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSYNCS	VSYNCS	HDES	VDES
	Reset value																													1	1	1	1
0x0084	LTDC_L1CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLUTEN	Res.	Res.	COLKEN	LEN
	Reset value																												0			0	0
0x0088	LTDC_L1WHPCR	Res.	Res.	Res.	Res.	WHSPPOS[11:0]												Res.	Res.	Res.	Res.	WHSTPOS[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
0x008C	LTDC_L1WVPCR	Res.	Res.	Res.	Res.	Res.	WVSPPOS[10:0]											Res.	Res.	Res.	Res.	Res.	WVSTPOS[10:0]										
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0
0x0090	LTDC_L1CKCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKRED[7:0]								CKGREEN[7:0]								CKBLUE[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
0x0094	LTDC_L1PFCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PF[2:0]		
	Reset value																														0	0	0
0x0098	LTDC_L1CACR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CONSTA[7:0]							
	Reset value																									1	1	1	1	1	1	1	1

表 117. LTDC 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
0x009C	LTDC_L1DCCR	DCALPHA[7:0]								DCRED[7:0]								DCGREEN[7:0]								DCBLUE[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
0x00A0	LTDC_L1BFCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BF1[2:0]			Res.	Res.	Res.	Res.	Res.	BF2[2:0]		
	Reset value																						1	1	0						1	1	1
0x00AC	LTDC_L1CFBAR	CFBADD[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
0x00B0	LTDC_L1CFBLR	Res.	Res.	Res.	CFBP[17:0]													Res.	Res.	Res.	CFBLL[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
0x00B4	LTDC_L1CFBLNR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CFBLNBR[10:0]										
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0x00C4	LTDC_L1CLUTWR	CLUTADD[7:0]								RED[7:0]								GREEN[7:0]								BLUE[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
0x0104	LTDC_L2CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLUTEN	Res.	Res.	COLKEN	LEN
	Reset value																												0			0	0
0x0108	LTDC_L2WHPCR	Res.	Res.	Res.	Res.	WHSPPOS[11:0]												Res.	Res.	Res.	Res.	WHSTPOS[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
0x010C	LTDC_L2WVPCR	Res.	Res.	Res.	Res.	Res.	WVSPPOS[10:0]											Res.	Res.	Res.	Res.	Res.	WVSTPOS[10:0]										
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0
0x0110	LTDC_L2CKCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKRED[7:0]								CKGREEN[7:0]								CKBLUE[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
0x0114	LTDC_L2PFCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PF[2:0]		
	Reset value																														0	0	0

表 117. LTDC 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
0x0118	LTDC_L2CACR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CONSTA[7:0]							
	Reset value																									1	1	1	1	1	1	1	1
0x011C	LTDC_L2DCCR	DCALPHA[7:0]								DCRED[7:0]								DCGREEN[7:0]								DCBLUE[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
0x0120	LTDC_L2BFCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BF1[2:0]			Res.	Res.	Res.	Res.	Res.	BF2[2:0]		
	Reset value																						1	1	0						1	1	1
0x012C	LTDC_L2CFBAR	CFBADD[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
0x0130	LTDC_L2CFBLR	Res.	Res.	Res.	CFBP[12:0]													Res.	Res.	Res.	CFBLL[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
0x0134	LTDC_L2CFBLNR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CFBLNBR[10:0]										
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0x0144	LTDC_L2CLUTWR	CLUTADD[7:0]								RED[7:0]								GREEN[7:0]								BLUE[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

19 随机数发生器 (RNG)

19.1 简介

RNG 处理器是一个以连续模拟噪声为基础的随机数发生器，在主机读数时提供一个 32 位的随机数。

RNG 已通过 FIPS PUB 140-2（2001 年 10 月 10 日）测试，成功率达 99%。

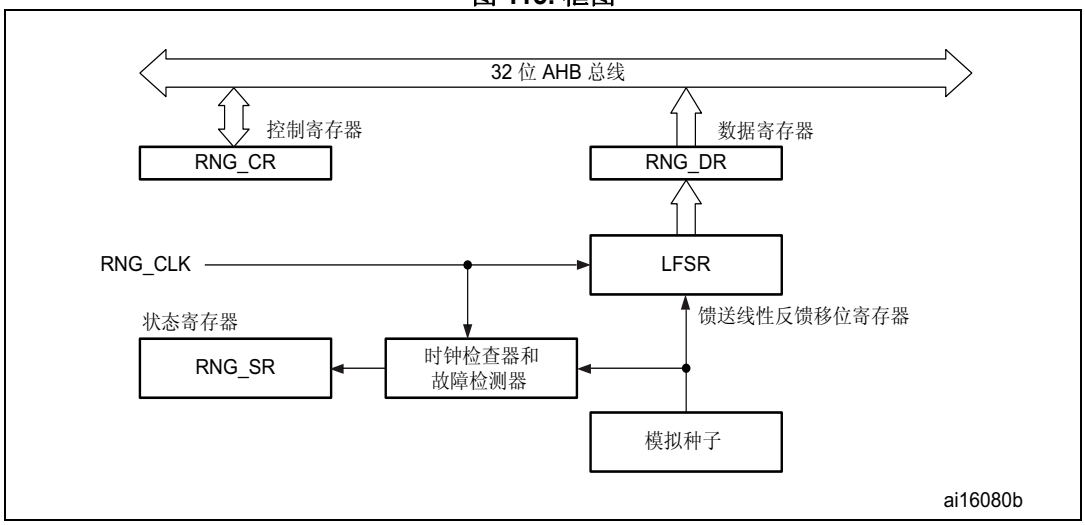
19.2 RNG 主要特性

- 提供由模拟量发生器产生的 32 位随机数
- 两个连续随机数的间隔为 40 个 RNG_CLK 时钟信号周期
- 通过监视 RNG 熵来标识异常行为（产生稳定值，或产生稳定的值序列）
- 可被禁止以降低功耗

19.3 RNG 功能说明

图 115 显示了 RNG 框图。

图 115. 框图



1. 有关 RNG时钟 (RNG_CLK) 源的更多详细信息，请参见*第 5 节：复位和时钟控制 (RCC)*。

随机数发生器采用模拟电路实现。此电路产生馈入线性反馈移位寄存器 (RNG_LFSR) 的种子，用于生成 32 位随机数。

该模拟电路由几个环形振荡器组成，振荡器的输出进行异或运算以产生种子。RNG_LFSR 由专用时钟 (RNG_CLK) 按恒定频率提供时钟信息，因此随机数质量与 HCLK 频率无关。当将大量种子引入 RNG_LFSR 后，RNG_LFSR 的内容会传入数据寄存器 (RNG_DR)。

同时，系统会监视模拟种子和专用时钟 RNG_CLK。状态位（RNG_SR 寄存器中）指示何时在种子上出现异常序列，或指示何时 RNG_CLK 时钟频率过低。检测到错误时生成中断。

19.3.1 操作

要运行 RNG，请按以下步骤操作：

1. 如果需要，使能中断（为此，将 RNG_CR 寄存器中的 IE 位置 1）。准备好随机数时或出现错误时生成中断。
2. 通过将 RNG_CR 寄存器中的 RNGEN 位置 1 使能随机数产生。这会激活模拟部分、RNG_LFSR 和错误检测器。
3. 每次中断时，检查确认未出现错误（RNG_SR 寄存器中的 SEIS 和 CEIS 位应为 0），并且随机数已准备就绪（RNG_SR 寄存器中的 DRDY 位为 1）。然后即可读取 RNG_DR 寄存器中的内容。

按照 FIPS PUB（联邦信息处理标准出版物）140-2 的要求，将 RNGEN 位置 1 后产生的第一个随机数不应使用，但应保存起来，与产生的下一个随机数进行比较。随后产生的每个随机数都需要与产生的上一个随机数进行比较。如果任何一对进行比较的数字相等，则测试失败（连续随机数发生器测试）。

19.3.2 错误管理

如果 CEIS 位的值为 1（时钟错误）

出现时钟错误时，RNG 无法再产生随机数，因为 RNG_CLK 时钟不正确。检查时钟控制器是否正确配置，是否可提供 RNG 时钟，然后将 CEIS 位清零。当 CECS 位为 0 时，RNG 可正常工作。时钟错误对产生的上一个随机数没有影响，因此 RNG_DR 寄存器内容可以使用。

如果 SEIS 位的值为 1（种子错误）

出现种子错误时，只要 SECS 位为 1，就会中断随机数产生。如果 RNG_DR 寄存器中有可用随机数，不能使用该随机数，因为它可能没有足够的熵。

应执行以下操作：将 SEIS 位清零，然后将 RNGEN 位清零并置 1，以便重新初始化和重新启动 RNG。

19.4 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。必须通过字（32 位）进行访问。

19.4.1 RNG 控制寄存器 (RNG_CR)

RNG 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.	Res.	Res.	Res.
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.	IE	RNGEN	Res.	Res.
												rw	rw		

位 31:4 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 DRDY=1 或 SEIS=1 或 CEIS=1，就会挂起中断。

位 2 **RNGEN**：随机数发生器使能 (Random number generator enable)
0：禁止随机数发生器
1：使能随机数发生器。

位 1:0 保留，必须保持复位值

19.4.2 RNG 状态寄存器 (RNG_SR)

RNG status 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.	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)
此位与 SECS 同时设置，通过向其写入 0 来清零。
0：未检测到错误序列
1：检测到以下错误序列之一：
– 超过 64 个连续位具有相同值（0 或 1）
– 超过 32 个连续交替的 0 和 1 (0101010101...01)
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)
此位与 CECS 同时设置，通过向其写入 0 来清零。
0：正确检测到 RNG_CLK 时钟
1：未正确检测到 RNG_CLK 时钟 ($f_{RNG_CLK} < f_{HCLK}/16$)
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：检测到以下错误序列之一：
– 超过 64 个连续位具有相同值（0 或 1）
– 超过 32 个连续交替的 0 和 1 (0101010101...01)

位 1 **CECS**：时钟错误当前状态 (Clock error current status)
0：正确检测到 RNG_CLK 时钟。如果 CEIS 位置 1，则意味着已检测到时钟错误并已恢复正常。
1：未正确检测到 RNG_CLK 时钟 ($f_{RNG_CLK} < f_{HCLK}/16$)。

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

注： 如果 RNG_CR 寄存器中 IE = 1，则会挂起中断。

读取 RNG_DR 寄存器后，此位恢复到 0，直到计算出新的有效值。

19.4.3 RNG 数据寄存器 (RNG_DR)

RNG data register

偏移地址：0x08

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。读取后，此寄存器在最多 40 个 RNG_CLK 时钟周期后，提供新的随机数值。在读取 RNDATA 值之前，软件必须检查 DRDY 位是否已置 1。

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

位 31:0 **RNDATA**：随机数据 (Random data)

32 位随机数据。

19.4.4 RNG 寄存器映射

表 118 给出了 RNG 寄存器映射和复位值。

表 118. RNG 寄存器映射和复位映射

偏移	寄存器名称 复位值	寄存器大小																														
		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	**RNG_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.	Res.	IE	RNGEN	Res.	Res.
	Reset value																													0	0		
0x04	**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	Res.	Res.
	Reset value																										0	0			0	0	0
0x08	**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

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

20 加密处理器 (CRYP)

除非特别说明，否则本部分适用于整个 STM32F756xx 系列器件。

20.1 CRYP 简介

借助加密处理器，可使用 DES、三重 DES 或 AES（128、192 或 256）算法对数据进行加密或解密。加密处理器完全兼容下列标准：

- 联邦信息处理标准出版物“FIPS PUB 46-3，1999 年 10 月 25 日”规定的数据加密标准 (DES) 和三重 DES (TDES)。它遵循美国国家标准协会 (ANSI) X9.52 标准。
- 联邦信息处理标准出版物（FIPS PUB 197，2001 年 11 月 26 日）规定的高级加密标准 (AES)

CRYP 处理器可在电子密码本 (ECB) 模式或加密分组链接 (CBC) 模式下使用 DES 和 TDES 算法执行数据加密和解密。

CRYP 外设为 32 位 AHB2 外设。它支持传入数据和已处理数据的 DMA 传输，并具有输入和输出 FIFO（分别为 8 个字的深度）。

20.2 CRYP 主要特性

- 适用于 AES、DES 和 TDES 加密和解密操作
- AES
  - 支持 ECB、CBC、CTR、CCM 和 GCM 链接算法
  - 支持 128 位、192 位和 256 位密钥
  - 支持在 CBC、CTR、CCM 和 GCM 模式下使用的 4 × 32 位初始化向量 (IV)

表 119. 处理 128 位块所需的周期数

算法/密钥大小	ECB	CBC	CTR	GCM				CCM			
				初始化	标头	有效负载	变量	初始化	标头	有效负载	变量
128b	14	14	14	24	10	14	14	12	14	25	14
192b	16	16	16	28	10	16	16	14	16	29	16
256b	18	18	18	32	10	18	18	16	18	33	18

- DES/TDES
  - 直接执行简单 DES 算法（使用单一密钥 K1）
  - 支持 ECB 和 CBC 链接算法
  - 支持 64 位、128 位和 192 位密钥（包括奇偶校验）
  - 支持在 CBC 模式下使用的 2 × 32 位初始化向量 (IV)
  - 使用 DES 处理一个 64 位块需要 16 个 HCLK 周期
  - 使用 TDES 处理一个 64 位块需要 48 个 HCLK 周期

- DES/TDES 和 AES 的共同点
  - 采用 IN 和 OUT FIFO（各具有 8 个字深和 32 位宽，与 4 个 DES 块或 2 个 AES 块相对应）
  - 采用自动数据流控制，支持直接存储器访问 (DMA)（使用 2 个通道，分别用于传入数据和已处理数据）
  - 采用数据交换逻辑，支持 1 位、8 位、16 位或 32 位数据

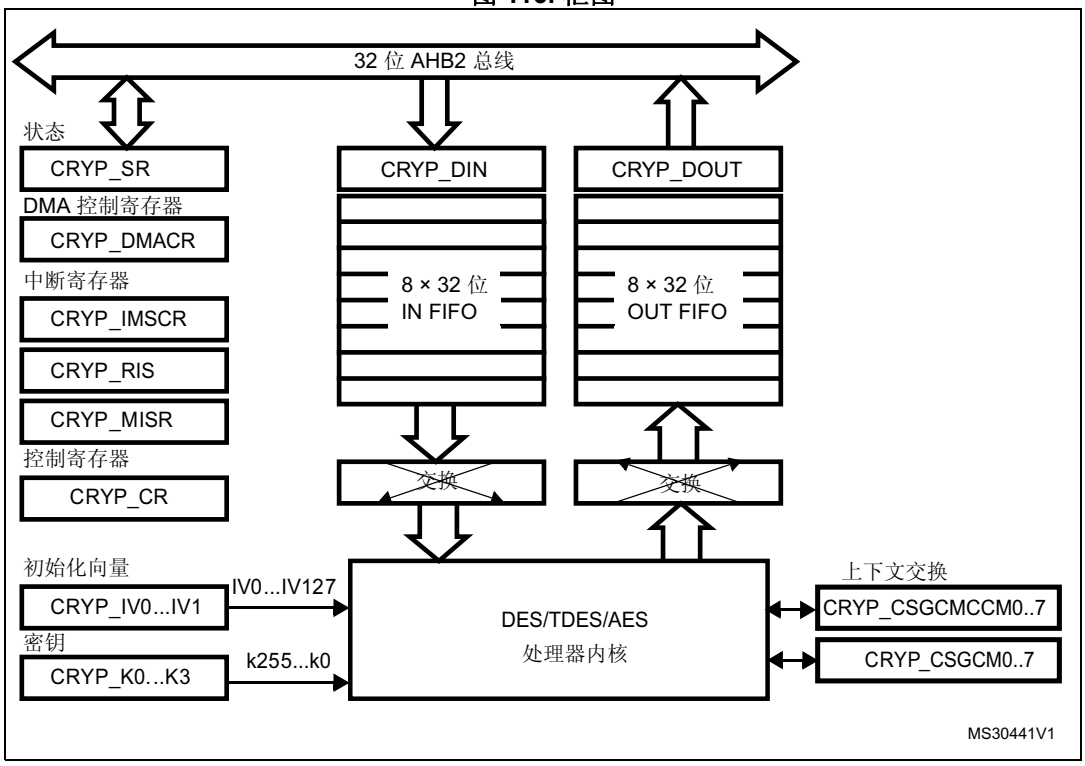
20.3 CRYP 功能说明

加密处理器可实现三重 DES（即 TDES，同样支持 DES）内核和 AES 加密内核。*第 20.3.1 节*和*第 20.3.2 节*提供有关内核的详细信息。

由于 TDES 和 AES 算法使用块密码，因此，加密前需对不完整的输入数据块进行填充（应将额外的位附加到数据串的尾端）。解密后需要丢弃填充项。硬件不处理填充操作，需要通过软件进行处理。

图 116 显示了加密处理器的框图。

图 116. 框图



20.3.1 DES/TDES 加密内核

DES/三重 DES 加密内核由三部分组成：

- DES 算法 (DEA)
- 多个密钥（DES 算法使用 1 个密钥，TDES 算法使用 1 到 3 个密钥）
- 初始化向量（在 CBC 模式中使用）

TDES 中涉及的基本处理如下：输入块通过 DEA 读取，并使用第一个密钥 K1 加密（TDES 模式中不使用 K0）。然后，输出使用第二个密钥 K2 解密，再使用第三个密钥 K3 加密。密钥由使用的算法决定：

- DES 模式：密钥 = [K1]
- TDES 模式：密钥 = [K3 K2 K1]

其中 Kx = [KxR KxL]，R = 右，L = 左

生成的输出块用于计算密文，具体取决于使用的模式。

注意，中间 DEA 阶段的输出始终不会在加密边界外显示。

TDES 允许使用三种不同的密钥选项：

- 三个独立的密钥

  第一个选项指定所有的密钥均是独立的，即 K1、K2 和 K3 均是独立的。FIPS PUB 46-3 – 1999（以及 ANSI X9.52 – 1998）将该选项称为“密钥选项 1”，并将这种 TDES 称为“3 密钥 TDES”。
- 两个独立的密钥

  第二个选项指定 K1 和 K2 是独立的，而 K3 等于 K1，即 K1 和 K2 独立，而 K3 = K1。FIPS PUB 46-3 – 1999（以及 ANSI X9.52 – 1998）将该选项称为“密钥选项 2”，并将这种 TDES 称为“2 密钥 TDES”。
- 三个相同的密钥

  第三个选项指定 K1、K2 和 K3 是相同的，即 K1 = K2 = K3。FIPS PUB 46-3 – 1999（以及 ANSI X9.52 – 1998）将第三个选项称为“密钥选项 3”。这种“1 密钥”TDES 与单独 DES 相同。

FIPS PUB 46-3 – 1999（以及 ANSI X9.52-1998）对 TDEA（TDES 算法）所提供的四种操作模式中涉及的处理过程进行了详尽的解释，这四种模式分别为：TDES-ECB 加密、TDES-ECB 解密、TDES-CBC 加密和 TDES-CBC 解密。
本参考手册仅对每种模式进行了简要解释。

DES 和 TDES 电子密码本 (DES/TDES-ECB) 模式

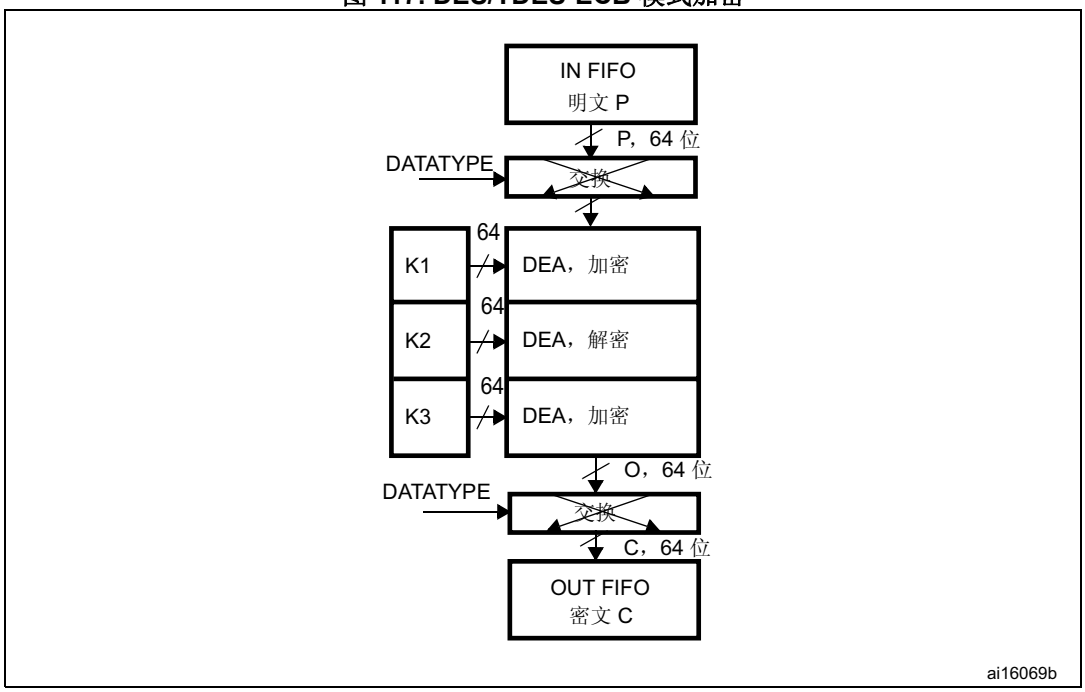
- DES/TDES-ECB 模式加密

  图 117 介绍了 DES 和 TDES 电子密码本 (DES/TDES-ECB) 模式中的加密。64 位明文数据块 (P) 经过位/字节/半字交换（请参见*第 525 页上的第 20.3.3 节：数据类型*）后作为输入块 (I)。输入块通过 DEA 在加密状态下使用 K1 进行加密处理。上述处理过程的输出会直接反馈到 DEA 的输入，在解密状态下使用 K2 执行 DES。上述处理过程的输出会直接反馈到 DEA 的输入，在加密状态下使用 K3 执行 DES。生成的 64 位输出块 (O) 在执行位/字节/半字交换之后，以密文 (C) 形式推入 OUT FIFO。

- DES/TDES-ECB 模式解密

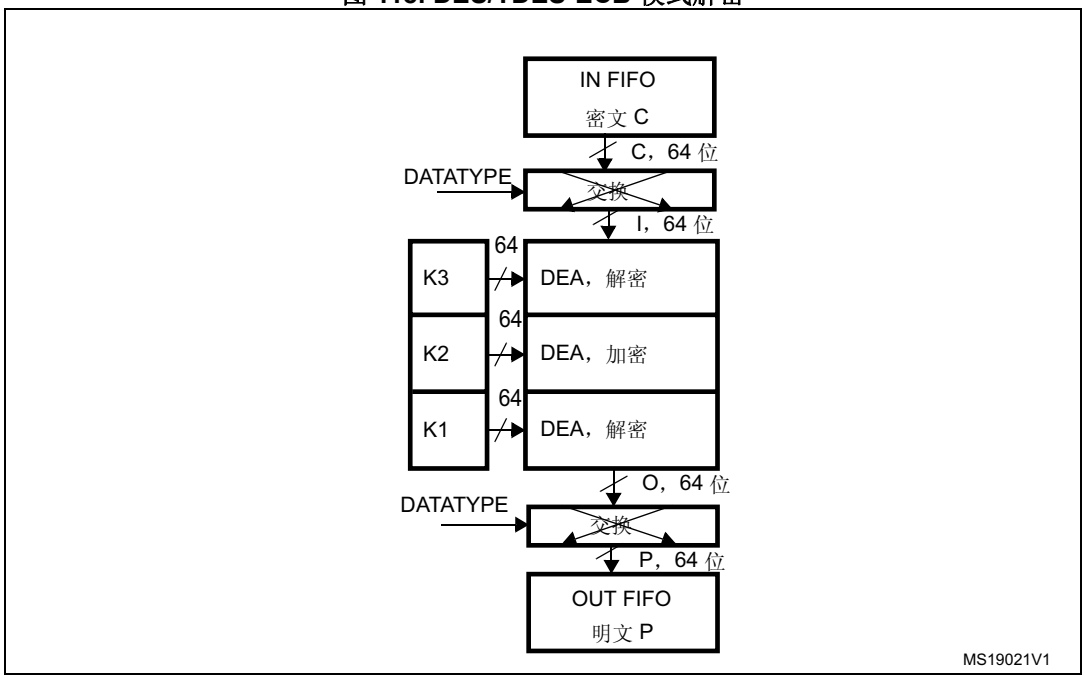
  图 118 介绍了 DES/TDES-ECB 解密。64 位官方块 (C) 经过位/字节/半字交换后，作为输入块 (I)。该密钥序列与加密过程中使用的密钥序列相反。输入块通过 DEA 在解密状态下使用 K3 进行解密处理。上述处理过程的输出会直接反馈到 DEA 的输入，在加密状态下使用 K2 执行 DES。新结果会直接反馈到 DEA 的输入，在解密状态下使用 K1 执行 DES。生成的 64 位输出块 (O) 在进行位/字节/半字交换后，产生明文 (P)。

图 117. DES/TDES-ECB 模式加密



1. K：密钥；C：密文；I：输入块；O：输出块；P：明文。

图 118. DES/TDES-ECB 模式解密



1. K：密钥；C：密文；I：输入块；O：输出块；P：明文。

DES 和 TDES 密码块链接 (DES/TDES-CBC) 模式

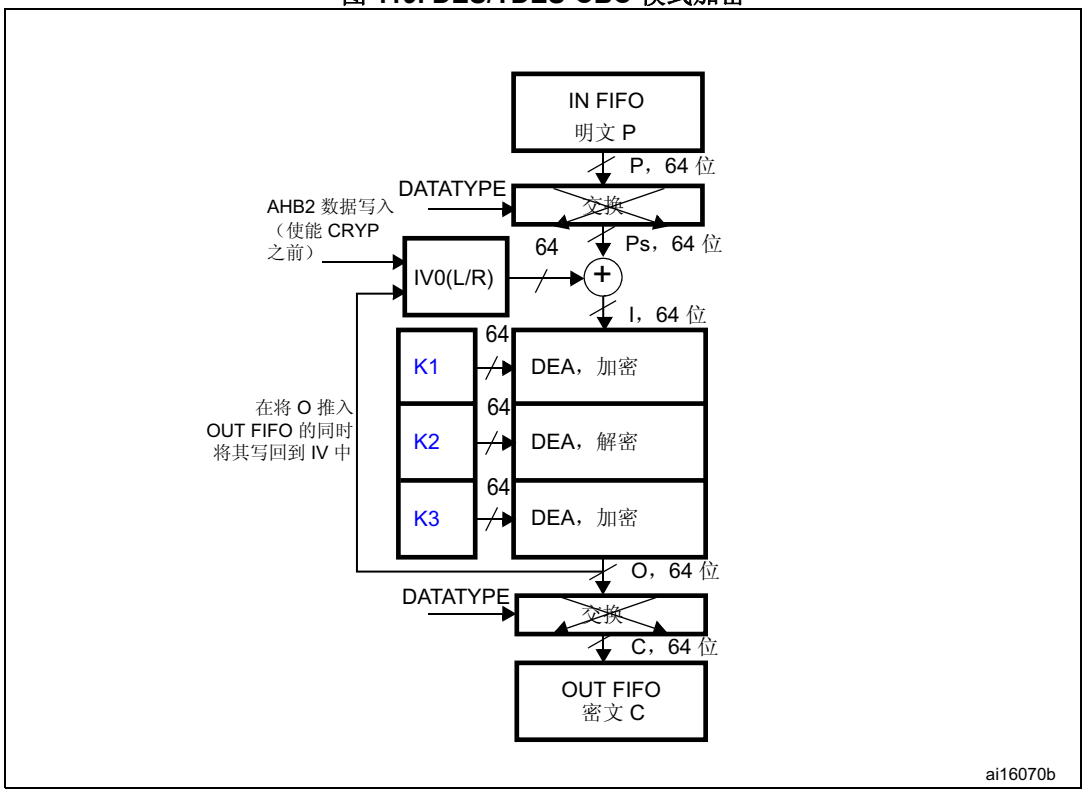
- DES/TDES-CBC 模式加密

  图 119 介绍了 DES 和三重 TDES 密码分组链接 (DES/TDES-CBC) 模式加密。该模式首先将明文消息分成多个 64 位数据块。在 TCBC 加密中，执行位/字节/半字交换（请参见 *第 525 页上的第 20.3.3 节：数据类型*）后获得的第一个输入块 (I_1) 是通过一个 64 位初始化向量 IV 与第一个明文数据块 (P_1) 进行异或运算形成的 ($I_1 = IV \oplus P_1$)。输入块通过 DEA 在加密状态下使用 K1 进行加密处理。上述处理过程的输出会直接反馈到 DEA 的输入，在解密状态下使用 K2 执行 DES。上述处理过程的输出会直接反馈到 DEA 的输入，在加密状态下使用 K3 执行 DES。生成的 64 位输出块 (O_1) 将直接用作密文 (C_1)，即 $C_1 = O_1$。然后，第一个密文块与第二个明文数据块进行异或运算，从而生成第二个输入块 (I_2) = ($C_1 \oplus P_2$)。注意，此时的 I_2 和 P_2 指的是第二个块。第二个输入块通过 TDEA 处理而生成第二个密文块。此加密处理会不断将后续密文块和明文块链接到一起，直到消息中最后一个明文块得到加密为止。如果消息中包含的数据块数不是整数，则应按照应用程序指定的方式对最后的不完整数据块进行加密。

- DES/TDES-CBC 模式解密

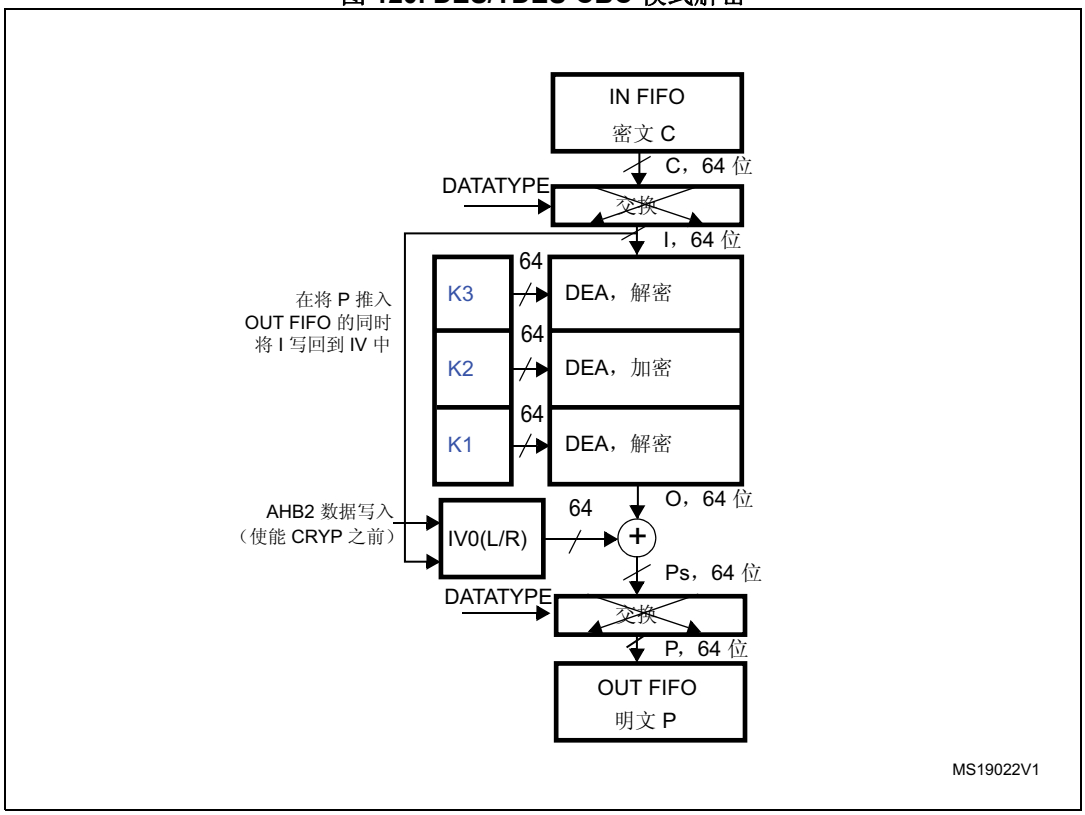
  在 DES/TDES-CBC 解密（请参见 *图 120*）中，第一个密文块 (C_1) 将直接用作输入块 (I_1)。该密钥序列与加密过程中使用的密钥序列相反。输入块通过 DEA 在解密状态下使用 K3 进行解密处理。上述处理过程的输出会直接反馈到 DEA 的输入，在加密状态下使用 K2 执行 DES。生成值会直接反馈到 DEA 的输入，在解密状态下使用 K1 处理 DES。生成的输出块与 IV（必须与加密期间使用的相同）进行异或运算，从而生成第一个明文块 ($P_1 = O_1 \oplus IV$)。然后，第二个密文块将用作下一个输入块，并由 TDEA 进行处理。生成的输出块与第一个密文块进行异或运算，从而生成第二个明文数据块 ($P_2 = O_2 \oplus C_1$)。（注意，P_2 和 O_2 指的是第二个数据块。）TCBC 解密过程将以此方式继续进行，直到最后一个完整密文块得到解密为止。必须按照应用程序指定的方式对不完整数据块密文进行解密。

图 119. DES/TDES-CBC 模式加密



1. K：密钥；C：密文；I：输入块；O：输出块；Ps：交换前（解码时）或交换后（编码时）的明文；P：明文；IV：初始化向量。

图 120. DES/TDES-CBC 模式解密



1. K：密钥；C：密文；I：输入块；O：输出块；Ps：交换前（解码时）或交换后（编码时）的明文；P：明文；IV：初始化向量。

20.3.2 AES 加密内核

AES 加密内核由以下三部分组成：

- AES 算法（AEA：高级加密算法）
- 多个密钥
- 初始化向量或随机值

AES 可使用以下三种长度的密钥：128 位密钥、192 位密钥或 256 位密钥，并且根据使用的操作模式，不使用初始化向量 (IV) 或使用一个 128 位的初始化向量 (IV)。

AES 中涉及的基本处理如下：128 位的输入块通过输入 FIFO 读取，然后发送至 AEA 使用密钥 (K0...3) 进行加密。密钥格式取决于密钥大小：

- 如果密钥大小 = 128：则密钥 = [K3 K2]
- 如果密钥大小 = 192：则密钥 = [K3 K2 K1]
- 如果密钥大小 = 256：则密钥 = [K3 K2 K1 K0]

其中 Kx = [KxR KxL]，R = 右，L = 左

生成的输出块用于计算密文，具体取决于使用的模式。

FIPS PUB 197（2001 年 11 月 26 日）对 AES 内核提供的四种操作模式中涉及的处理过程进行了详尽的解释，这四种模式为：AES-ECB 加密、AES-ECB 解密、AES-CBC 加密和 AES-CBC 解密。本参考手册仅对每种模式进行了简要解释。

AES 电子密码本 (AES-ECB) 模式

- AES-ECB 模式加密

  图 121 介绍了 AES 电子密码本 (AES-ECB) 模式加密。

  在 AES-ECB 加密中，执行位/字节/半字交换（请参见*第 525 页上的第 20.3.3 节：数据类型*）后，128 位明文数据块 (P) 将用作输入块 (I)。输入块通过 AEA 在加密状态下使用 128 位、192 位或 256 位密钥进行处理。执行位/字节/半字交换后，生成的 128 位输出块 (O) 将用作密文 (C)。该密文随后推入 OUT FIFO。

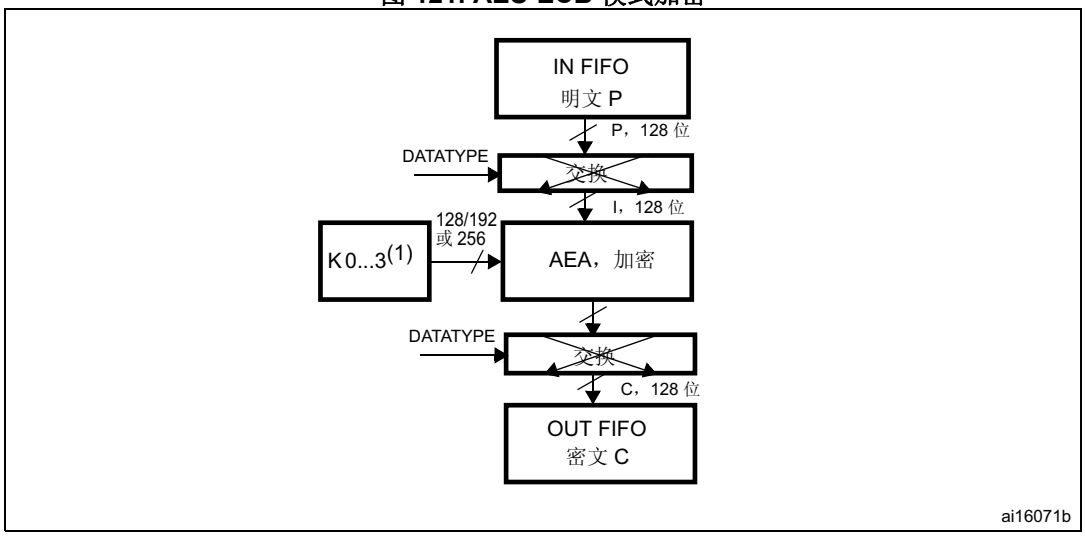
- AES-ECB 模式解密

  图 122 介绍了 AES 电子密码本 (AES-ECB) 模式加密。

  要在 ECB 模式下执行 AES 解密，需要准备密钥（需要针对加密执行完整的密钥计划），方法为：收集最后一个轮密钥并将其用作解密密文的第一个轮密钥。该准备过程通过 AES 内核计算完成。有关如何准备密钥的详细信息，请参见*第 20.3.6 节：加密或解密执行步骤*。

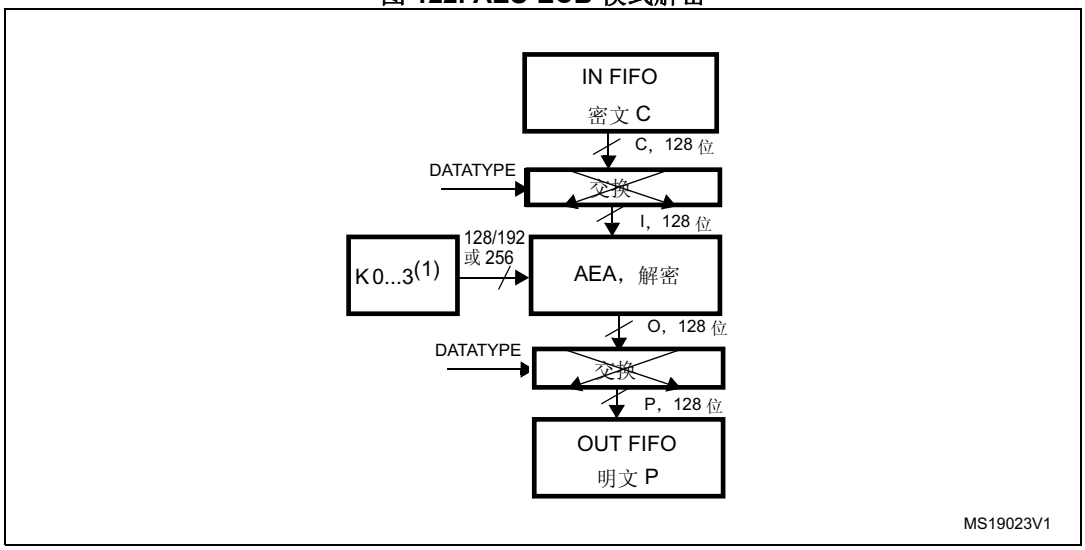
  在 AES-ECB 解密中，执行位/字节/半字交换后，128 位密文块 (C) 将用作输入块 (I)。该密钥序列与加密处理中的密钥序列相反。执行位/字节/半字交换，生成的 128 位输出块 (O) 将产生明文 (P)。

图 121. AES-ECB 模式加密



1. K：密钥；C：密文；I：输入块；O：输出块；P：明文。
2. 如果密钥大小 = 128：则密钥 = [K3 K2]。
   如果密钥大小 = 192：则密钥 = [K3 K2 K1]。
   如果密钥大小 = 256：则密钥 = [K3 K2 K1 K0]。

图 122. AES-ECB 模式解密



1. K：密钥；C：密文；I：输入块；O：输出块；P：明文。
2. 如果密钥大小 = 128 => 密钥 = [K3 K2]。
   如果密钥大小 = 192 => 密钥 = [K3 K2 K1]。
   如果密钥大小 = 256 => 密钥 = [K3 K2 K1 K0]。

AES 加密分组链接 (AES-CBC) 模式

- AES-CBC 模式加密

  AES 加密分组链接 (AES-CBC) 模式解密如*图 123* 所示。

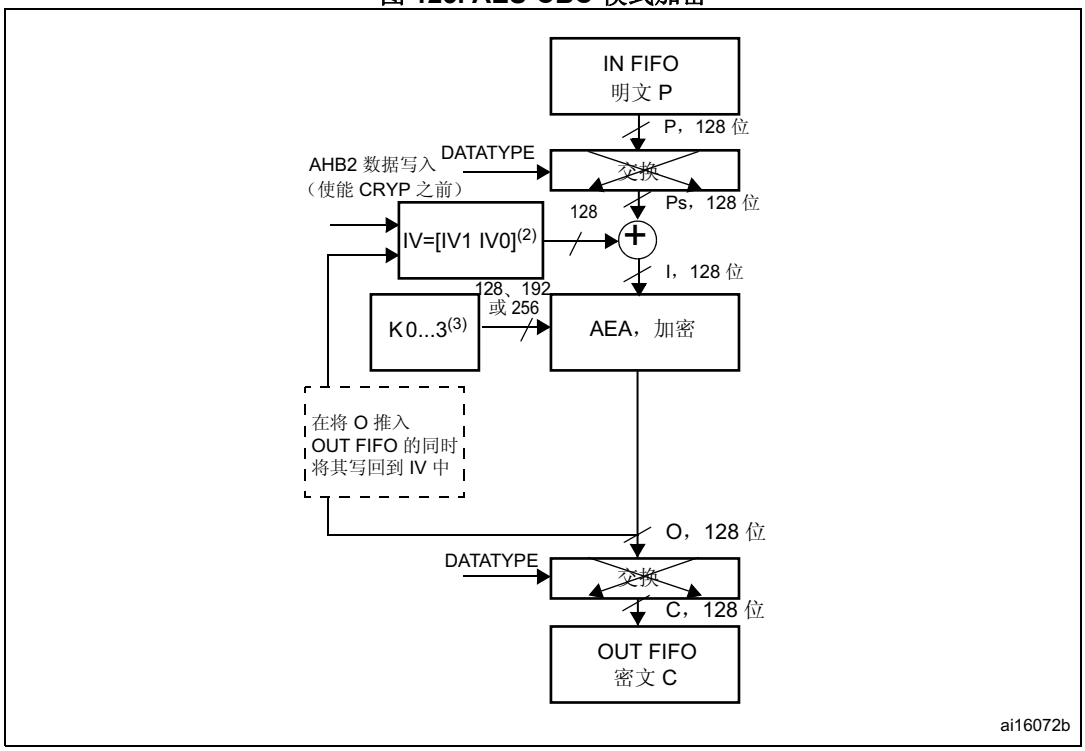
  在 AES-CBC 加密中，执行位/字节/半字交换（请参见*第 525 页上的第 20.3.3 节：数据类型*）后所获得的第一个输入块 (I_1) 是通过一个 128 位初始化向量 IV 与第一个明文数据块 (P_1) 进行异或运算形成的 ($I_1 = IV \oplus P_1$)。输入块通过 AEA 在加密状态下使用 128 位、192 位或 256 位密钥 (K0...K3) 进行处理。生成的 128 位输出块 (O_1) 将直接用作密文 (C_1)，即 $C_1 = O_1$。然后，第一个密文块与第二个明文数据块进行异或运算，从而生成第二个输入块 (I_2) = ($C_1 \oplus P_2$)。注意，此时的 I_2 和 P_2 指的是第二个块。第二个输入块通过 AEA 处理而生成第二个密文块。此加密处理会不断将后续密文块和明文块链接到一起，直到消息中最后一个明文块得到加密为止。如果消息中包含的数据块数不是整数，则应按照应用程序指定的方式对最后的不完整数据块进行加密。

  在 CBC 模式下（如 ECB 模式），必须准备密钥才可以执行 AES 解密。有关如何准备密钥的详细信息，请参见*第 530 页上的第 20.3.6 节：加密或解密执行步骤*。

- AES-CBC 模式解密

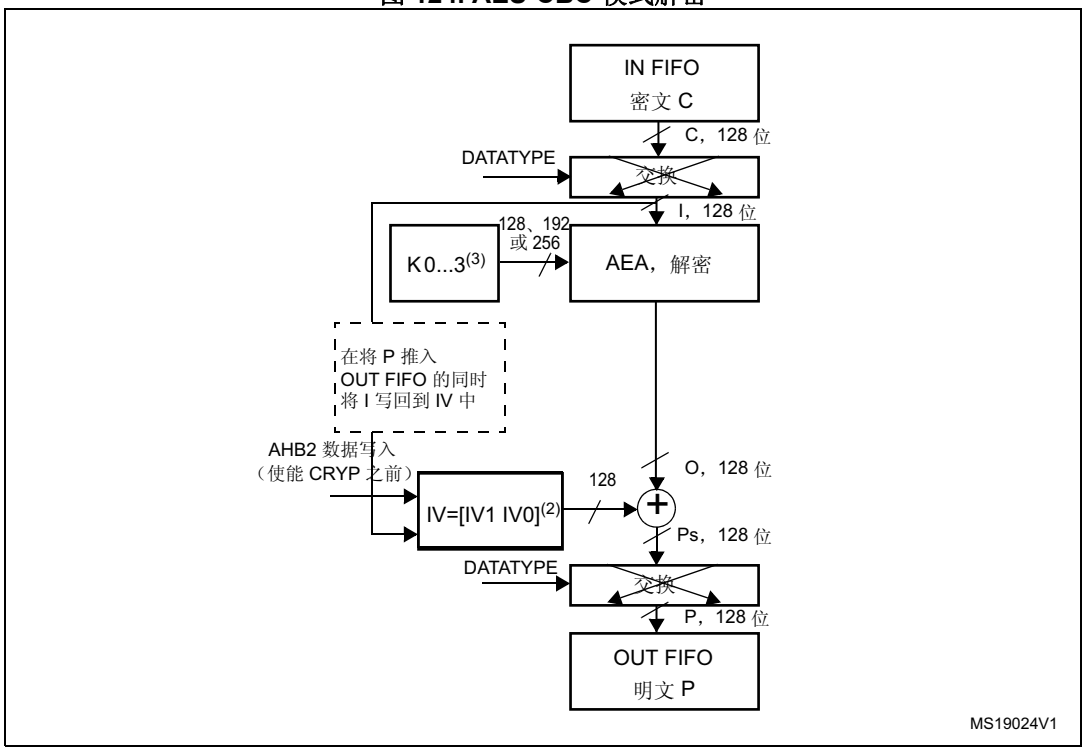
  在 AES-CBC 解密（请参见*图 124*）中，第一个 128 位密文块 (C_1) 将直接用作输入块 (I_1)。输入块通过 AEA 在解密状态下使用 128 位、192 位或 256 位密钥进行处理。生成的输出块与 128 位初始化向量 IV（必须与加密期间使用的相同）进行异或运算，从而生成第一个明文块 ($P_1 = O_1 \oplus IV$)。然后，第二个密文块将用作下一个输入块，并由 AEA 进行处理。生成的输出块与第一个密文块进行异或运算，从而生成第二个明文数据块 ($P_2 = O_2 \oplus C_1$)。（注意，P_2 和 O_2 指的是第二个数据块。）AES-CBC 解密过程将以此方式继续进行，直到最后一个完整密文块得到解密为止。必须按照应用程序指定的方式对不完整数据块密文进行解密。

图 123. AES-CBC 模式加密



1. K：密钥；C：密文；I：输入块；O：输出块；Ps：交换前（解码时）或交换后（编码时）的明文；P：明文；IV：初始化向量。
2. IVx = [IVxR IVxL]，R = 右，L = 左。
3. 如果密钥大小 = 128 => 密钥 = [K3 K2]。
   如果密钥大小 = 192 => 密钥 = [K3 K2 K1]。
   如果密钥大小 = 256 => 密钥 = [K3 K2 K1 K0]。

图 124. AES-CBC 模式解密



1. K：密钥；C：密文；I：输入块；O：输出块；Ps：交换前（解码时）或交换后（编码时）的明文；P：明文；IV：初始化向量。
2. IVx = [IVxR IVxL]，R = 右，L = 左。
3. 如果密钥大小 = 128 => 密钥 = [K3 K2]。
   如果密钥大小 = 192 => 密钥 = [K3 K2 K1]。
   如果密钥大小 = 256 => 密钥 = [K3 K2 K1 K0]。

AES 计数器模式 (AES-CTR) 模式

AES 计数器模式使用 AES 块作为密钥流生成器。然后，生成的密钥与明文进行异或运算后获得密文。因此，认为 CTR 加密和解密有差异是不合理的，因为这两种操作完全相同。

事实上，给定：

- 明文：P[0]、P[1]、...、P[n]（每个均为 128 位）
- 要使用的密钥 K（大小不重要）
- 初始计数器块（称为 ICB，但功能与 CBC 的 IV 相同）

密文将按如下公式计算：

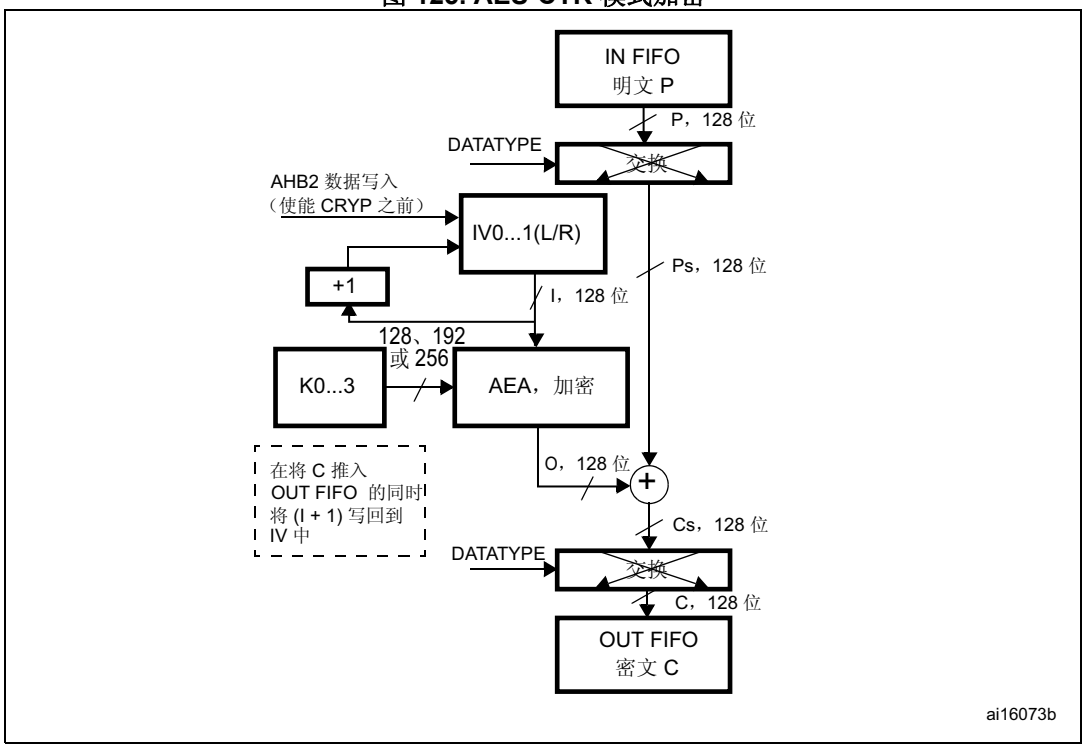
`C[i] = enck(iv[i]) xor P[i]`，其中：

`iv[0]` = ICB，而 `iv[i+1] = func(iv[i])`，其中 `func` 是应用于先前 iv 块的更新函数；`func` 从本质上而言是组成 iv 块的字段之一的增量。

假定解密所用的 ICB 与加密所用的 ICB 相同，则在解密期间生成的密钥流便与加密期间生成的密钥流相同。然后，密文会与密钥流进行异或运算，从而得到原始明文。因此，解密操作与加密操作的操作方式完全相同。

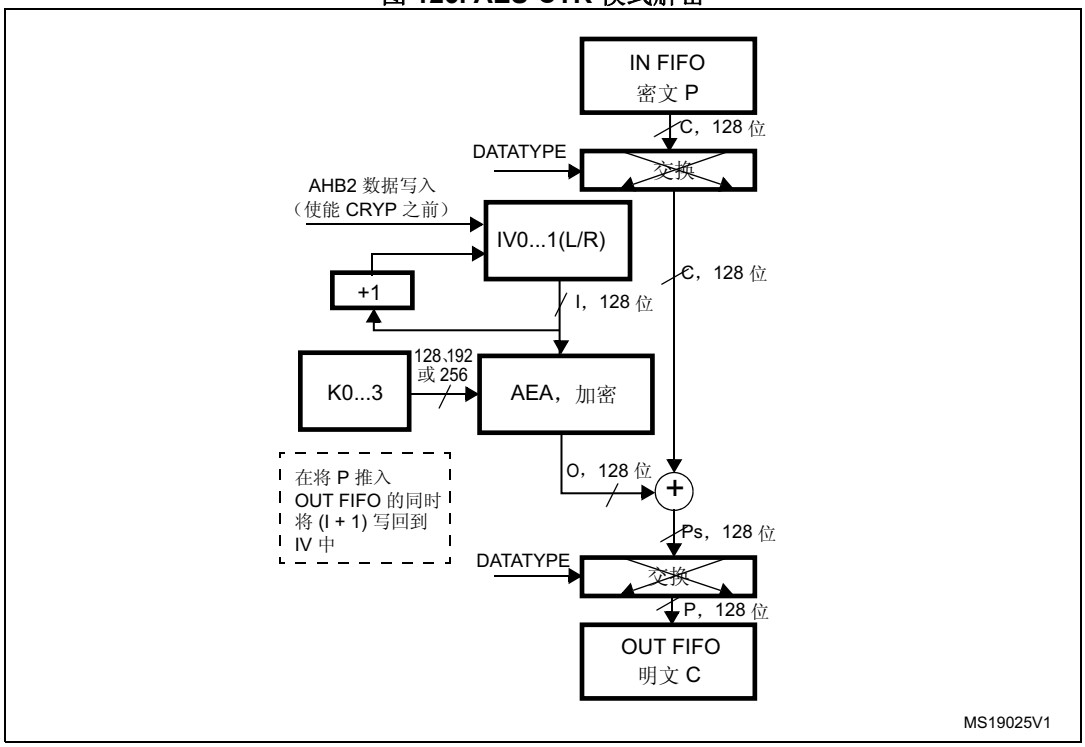
图 125 和*图 126* 分别介绍了 AES-CTR 加密和解密。

图 125. AES-CTR 模式加密



1. K：密钥；C：密文；I：输入块；O：输出块；Ps：交换前（解码时）或交换后（编码时）的明文；Cs：交换后（解码时）或交换前（编码时）的密文；P：明文；IV：初始化向量。

图 126. AES-CTR 模式解密



1. K：密钥；C：密文；I：输入块；O：输出块；Ps：交换前（解码时）或交换后（编码时）的明文；Cs：交换后（解码时）或交换前（编码时）的密文；P：明文；IV：初始化向量。

图 127 显示了由标准 [2] 定义的 IV 块结构。它由三个不同字段组成。

图 127. 计数器模式下的初始计数器块结构



- 随机值为 32 位一次性值。应将新的随机值分配至每个不同的通信。
- 初始化向量 (IV) 是一个 64 位值，而标准指定加密器必须选择 IV，以确保一个给定值只用于一个给定的密钥。
- 计数器是一个 32 位的大端模式的整数，随着块的加密次数而递增。应将计数器的初始值设置为“1”。

块将使用最低有效的 32 位进行增量计数，而保持其它最高有效的 96 位不变。

AES Galois/计数器模式 (GCM)

AES Galois/计数器模式 (GCM) 允许加密和验证明文，以及生成对应的密文和标记（也称为消息验证码或消息完整性检查）。该算法基于 AES 计数器模式，可保证保密性。它针对固定有限的字段使用乘法器来生成标记。开始执行算法时需要使用初始化向量。

要处理的消息分为 2 个部分：

- 标头（又称附加认证数据）：需要验证但不受保护的数据（如用于路由数据包的信息）
- 有效负载（又称明文或密文）：经验证和加密的消息本身。

注： *标头必须在有效负载的前面，并且两部分不能混合。*

根据 GCM 标准，在消息结束时，必须传递完由标头大小（64 位）和有效负载大小（64 位）构成的特定 128 位块。在计算过程中，必须将标头块与有效负载块区分开。

在 GCM 模式下，执行加密/解密需要以下四个步骤：

1. GCM 初始化阶段

   在此步骤中，将在内部计算和保存 HASH 密钥，供处理所有块使用。建议遵守下列顺序：

   a) 确保通过将 CRYP_CR 寄存器中的 CRYPEN 位清零来禁止加密处理器。

   b) 通过将 CRYP_CR 中的 ALGOMODE 位编程为“01000”来选择 GCM 链接模式。

   c) 将 CRYP_CR 中的 GCM_CCMPH 位配置为“00”来开始 GCM 初始化阶段。

   d) 初始化 CRYP_KEYRx 中的密钥寄存器（128 位、192 位和 256 位）以及初始化向量 (IV)。

   e) 将 CRYPEN 位置 1 以开始计算 HASH 密钥。

   f) 等待 CRYPEN 位清零后进入下一个阶段。

   g) 将 CRYPEN 位置“1”。

2. GCM 标头阶段

   必须在 GCM 初始化阶段后方能执行该步骤：

   h) 将 CRYP_CR 中的 GCM_CCMPH 位置“01”来指示标头阶段已开始。

   i) 写入标头数据。可以使用以下三种方法：

   – 按 32 位块将数据写入 CRYP_DIN 寄存器，并使用 IFNF 标志来确定输入 FIFO 是否能够接收数据。标头大小必须为 128 位（4 个字）的倍数。

   – 按 8 个字块将数据写入 CRYP_DIN 寄存器，并使用 IFEM 标志来确定输入 FIFO 是否能够接收数据（IFEM = “1”）。标头大小必须为 128 位（4 个字）的倍数。

   – 使用 DMA。

   j) 提供完所有的标头数据后，等待 CRYP_SR 寄存器中的 BUSY 位清零。

3. GCM 有效负载阶段（加密/解密）

   必须在 GCM 标头阶段后方能执行该步骤：

   k) 将 CRYP_CR 寄存器中的 GCM_CCMPH 配置为“10”。

   l) 通过使用 CRYP_CR 中的 ALGODIR 位来选择算法方向（加密或解密）。

   m) 将有效负载消息写入 CRYP_DIN 寄存器，并使用 IFNF 标志来确定输入 FIFO 是否能够接收数据。也可以按 8 个字块将数据写入 CRYP_DIN 寄存器，并使用 IFEM 标志来确定输入 FIFO 是否能够接收数据（IFEM = “1”）。同时，可对 CRYP_DOUT 寄存器的 OFNE/OFFU 标志进行监控，以检查输出 FIFO 是否为空。

   n) 重复之前的步骤，直到所有的有效负载块均得到加密或解密。也可以使用 DMA。

4. GCM 最后阶段

   该步骤将生成验证标记：

   o) 将 CRYP_CR 中的 GCM_CCMPH[1:0] 配置为“11”。

   p) 将输入写入 CRYP_DIN 寄存器 4 次。输入中必须包含标头中的位数（64 位），并连接着有效负载中的位数（64 位）。

   q) 等待 CRYP_SR 寄存器中的 OFNE 标志（FIFO 输出非空）置“1”。

   r) 读取 CRYP_DOUT 寄存器 4 次：此输出即为验证标记。

   s) 禁止加密处理器（CRYP_CR 中的 CRYPEN 位 =“0”）

注： *执行解密时，开始时无需计算密钥。解密结束时，应将生成的标记与通过消息传递的预期标记进行比较。此外，必须将 ALGODIR 位（算法方向）置“1”。*

从标头阶段到标记阶段时，无需禁止/使能 CRYP 处理器。

AES Galois 消息认证代码 (GMAC)

加密处理器还支持使用 GMAC 算法对明文进行验证。它针对固定有限的字段使用 GCM 算法和乘法器来生成相应标记。

开始执行算法时需要使用初始化向量。

实际上，GMAC 算法相当于应用在仅由标头构成的消息上的 GCM 算法。因此，不需要有效负载阶段。

AES 组合式密码机 (CCM)

CCM 算法允许加密和验证明文，以及生成对应的密文和标记（又称消息验证码或消息完整性检查）。该算法基于 AES 计数器模式，可保证保密性。它使用 AES CBC 模式生成 128 位标记。

该 CCM 标准（RFC 3610 Counter with CBC-MAC (CCM) 标准，2003 年 9 月发布）可为首个验证块（在此标准中称为 B0）定义特定的编码规则。具体而言，首个块包括标志、随机值和有效负载长度（字节数）。CCM 标准可为加密/解密指定另外的格式（称为 A 或计数器）。计数器在有效负载阶段会递增计数，其 32 个低有效位在生成标记期间会初始化为“1”（在 CCM 标准中称为 A0 数据包）。

注： *硬件不执行 B0 数据包的格式化操作，该操作由软件处理。*

对于 GCM 算法，要处理的消息会分为 2 个部分：

- 标头（又称附加认证数据）：需要验证但不受保护的数据（如用于路由数据包的信息）
- 有效负载（又称明文或密文）：经验证和加密的消息本身。

注： *标头必须在有效负载的前面，并且两部分不能混合。*

在 CCM 模式下，执行加密或解密需要以下四个步骤：

1. CCM 初始化阶段

   第一步，将 CCM 消息的 B0 数据包（第一个数据包）会写入 CRYP_DIN 寄存器。在此阶段期间，CRYP_DOUT 寄存器不包含任何输出数据。

   必须遵守以下顺序：

   a) 确保通过将 CRYP_CR 寄存器中的 CRYPEN 位清零来禁止加密处理器。

   b) 通过将 CRYP_CR 寄存器中的 ALGOMODE 位置为“01001”来选择 CCM 链接模式。

   c) 将 CRYP_CR 中的 GCM_CCMPH 位配置为“00”来开始 CCM 初始化阶段。

   d) 初始化 CRYP_KEYRx 中的密钥寄存器（128 位、192 位和 256 位）以及初始化向量 (IV)。

   e) 将 CRYP_CR 中的 CRYPEN 位置“1”。

   f) 将 B0 数据包写入输入数据寄存器。

   g) 等待 CRYPEN 位清零后进入下一个阶段。

   h) 将 CRYPEN 位置“1”。

2. CCM 标头阶段

   必须在 CCM 初始化阶段后方能执行该步骤。加密与解密的顺序完全相同。

   在此阶段期间，CRYP_DOUT 寄存器不包含任何输出数据。

   如无附加验证数据，则可以跳过该阶段。

   必须遵守以下顺序：

   i) 将 CRYP_CR 中的 GCM_CCMPH 位置“01”来指示标头阶段已开始。

   j) 可以使用以下三种方法：

   – 按 32 位块将标头数据写入 CRYP_DIN 寄存器，并使用 IFNF 标志来确定输入 FIFO 是否能够接收到数据。标头大小必须为 128 位（4 个字）的倍数。

   – 按 8 个字块将标头数据写入 CRYP_DIN 寄存器，并使用 IFEM 标志来确定输入 FIFO 是否能够接收到数据（IFEM = “1”）。标头大小必须为 128 位（4 个字）的倍数。

   – 使用 DMA。

注： *必须使用标头长度格式化首个块 B1。应通过软件执行该任务。*

   k) 提供完所有的标头数据后，等待 BUSY 标志清零。

3. CCM 有效负载阶段（加密/解密）

   必须在 CCM 标头阶段后方能执行该步骤。在此阶段期间，加密/解密的有效负载将存储在 CRYP_DOUT 寄存器中。

   必须遵守以下顺序：

   l) 将 CRYP_CR 中的 GCM_CCMPH 位配置为“10”。

   m) 通过使用 CRYP_CR 中的 ALGODIR 位来选择算法方向（加密或解密）。

   n) 将有效负载消息写入 CRYP_DIN 寄存器，并使用 IFNF 标志来确定输入 FIFO 是否能够接收数据。也可以按 8 个字块将数据写入 CRYP_DIN 寄存器，并使用 IFEM 标志来确定输入 FIFO 是否能够接收数据（IFEM = “1”）。同时，可对 CRYP_DOUT 寄存器的 OFNE/OFFU 标志进行监控，以检查输出 FIFO 是否为空。

   o) 重复之前的步骤，直到所有的有效负载块均得到加密或解密。也可以使用 DMA。

4. CCM 最后阶段

   该步骤将生成认证标记。在此阶段期间，将生成消息的认证标记，该标记存储在 CRYP_DOUT 寄存器中。

   p) 将 CRYP_CR 中的 GCM_CCMPH[1:0] 位配置为“11”。

   q) 加载 A0 初始化计数器，并通过将 32 位数据写入 CRYP_DIN 寄存器 4 次来编程 128 位 A0 值。

   r) 等待 CRYP_SR 寄存器中的 OFNE 标志（FIFO 输出非空）置“1”。

   s) 读取 CRYP_DOUT 寄存器 4 次：此输出即为加密的验证标记。

   t) 禁止加密处理器（CRYP_CR 中的 CRYPEN 位 =“0”）。

注： *硬件不执行原始 B0 和 B1 数据包的格式化操作，也不执行加密与解密之间的标记比较操作。这些操作通过软件处理。*

从标头阶段到标记阶段，无需禁止/使能加密处理器。

AES 密文消息验证代码 (CMAC)

CMAC 算法允许验证明文并生成相应的标记。CMAC 顺序与 CCM 顺序相同，但其跳过有效负载阶段。

20.3.3 数据类型

将数据写入 CRYP_DIN 寄存器时，一次会向 CRYP 处理器输入 32 位（字）数据。DES 的原则是每隔 64 位对数据流进行处理。针对每个 64 位块，将从 M1 到 M64 对位进行编号，其中 M1 为块最左侧的位，M64 为块最右侧的位。AES 使用相同的原则，但其块大小为 128 位。

系统存储器结构采用小端模式：无论使用何种数据类型（位、字节、16 位半字、32 位字），最低有效数据均占用最低地址位置。因此，对于从 IN FIFO 中读取的数据，在其进入 CRYP 处理器之前，必须对这些数据执行位、字节或半字交换操作（取决于要加密的数据类型）。在 CRYP 数据写入 OUT FIFO 之前，需要对其执行同样的交换操作。例如，对 ASCII 文本流执行字节交换操作。

要处理的数据类型通过 CRYP 控制寄存器 (CRYP_CR) 中的 DATATYPE 位字段进行配置。表 120 的彩色图例如下：绿色、蓝色、红色和白色突出显示系统存储器中表示的 TDES 块值。

表 120. 数据类型

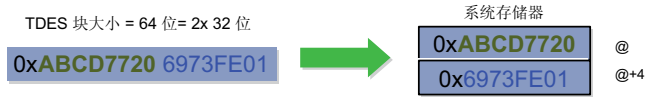
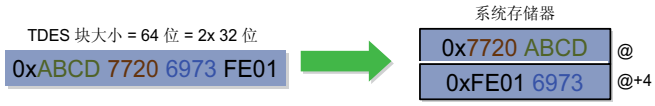
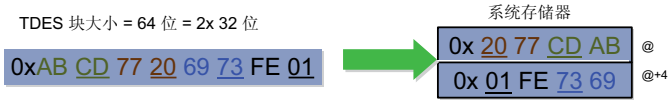
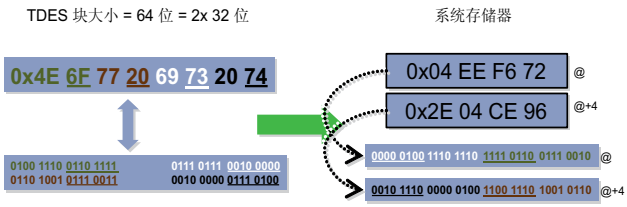
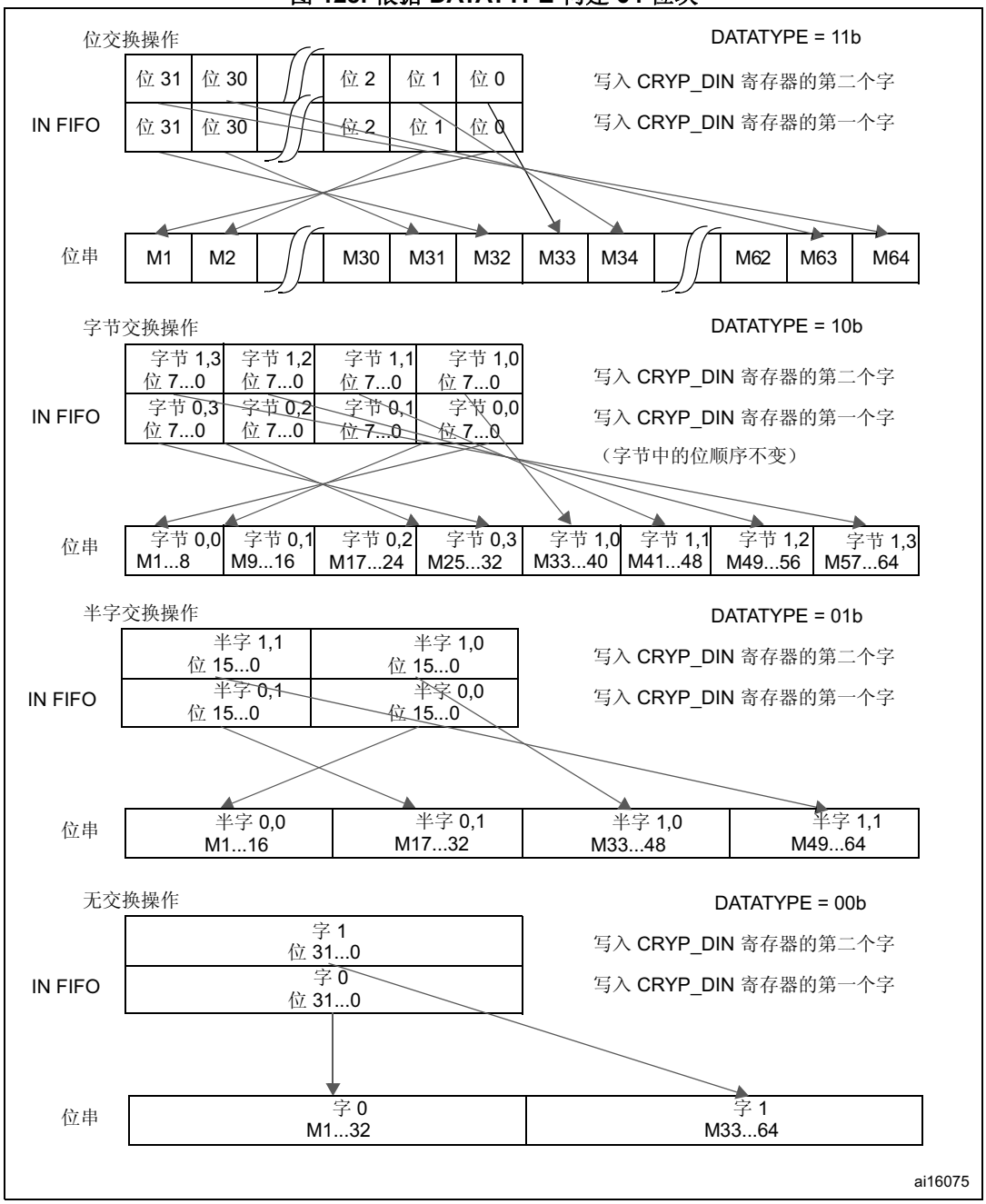
CRYP_CR 中的 DATATYPE	要执行的交换	系统存储器数据（明文或密码）
00b	无交换	示例：TDES 块值 **0xABCD77206973FE01** 在系统存储器中表示为：
01b	半字（16 位）交换	示例：TDES 块值 **0xABCD77206973FE01** 在系统存储器中表示为：
10b	字节（8位）交换	示例：TDES 块值 **0xABCD77206973FE01** 在系统存储器中表示为：
11b	位交换	TDES 块值 **0x4E6F772069732074** 在系统存储器中表示为：

图 128 展示了根据不同的 DATATYPE 值，64 位数据块 M1...64 是如何由 CRYP 处理器从 IN FIFO 中弹出的两个连续的 32 位字来构建的。相同的原理，可以轻松扩展构建用于 AES 加密算法的 128 位数据块（对于 AES，块长度为四个 32 位字，但由于交换仅发生在字级别，因此它与此处描述的 TDES 的交换过程相同）。

注： *IN FIFO 和 CRYP 数据块之间，以及 CRYP 数据块和 OUT FIFO 之间执行相同的交换操作。*

图 128. 根据 DATATYPE 构建 64 位块



20.3.4 初始化向量 —— CRYP_IV0...1(L/R)

可将初始化向量视为两个 64 位数据项。因此，初始化向量在系统存储器中的数据格式和表示形式均不同于明文或密码数据，而且它们不受 DATATYPE 值的影响。

初始化向量采用两个连续的 32 位字进行定义，即 CRYP_IVL（左部分，记为位 IV1...32）和 CRYP_IVR（右部分，记为位 IV33...64）。

在 DES 或 TDES CBC 加密期间，CRYP_IV0(L/R) 位会与一个 64 位数据块（即，数据块的 M1...64 位）进行异或运算，这里的 64 位数据块是根据 DATATYPE 值交换后从 IN FIFO 弹出的。当 DEA3 块的输出可用时，该输出值会复制到 CRYP_IV0(L/R) 向量，之后，这个新值与 IN FIFO 弹出的下一个 64 位数据块进行异或运算，以此类推。

在 DES 或 TDES CBC 解密期间，CRYP_IV0(L/R) 位首先会与 TDEA1 块输出的 64 位数据块（即，M1...64 位）进行异或运算，然后异或运算后的结果根据 DATATYPE 值进行交换并压入 OUT FIFO。异或运算后的结果进行交换并压入 OUT FIFO 后，IN FIFO 的输出会取代 CRYP_IV0(L/R) 值，然后 IN FIFO 执行弹出操作，随后可对新的 64 位数据块进行处理。

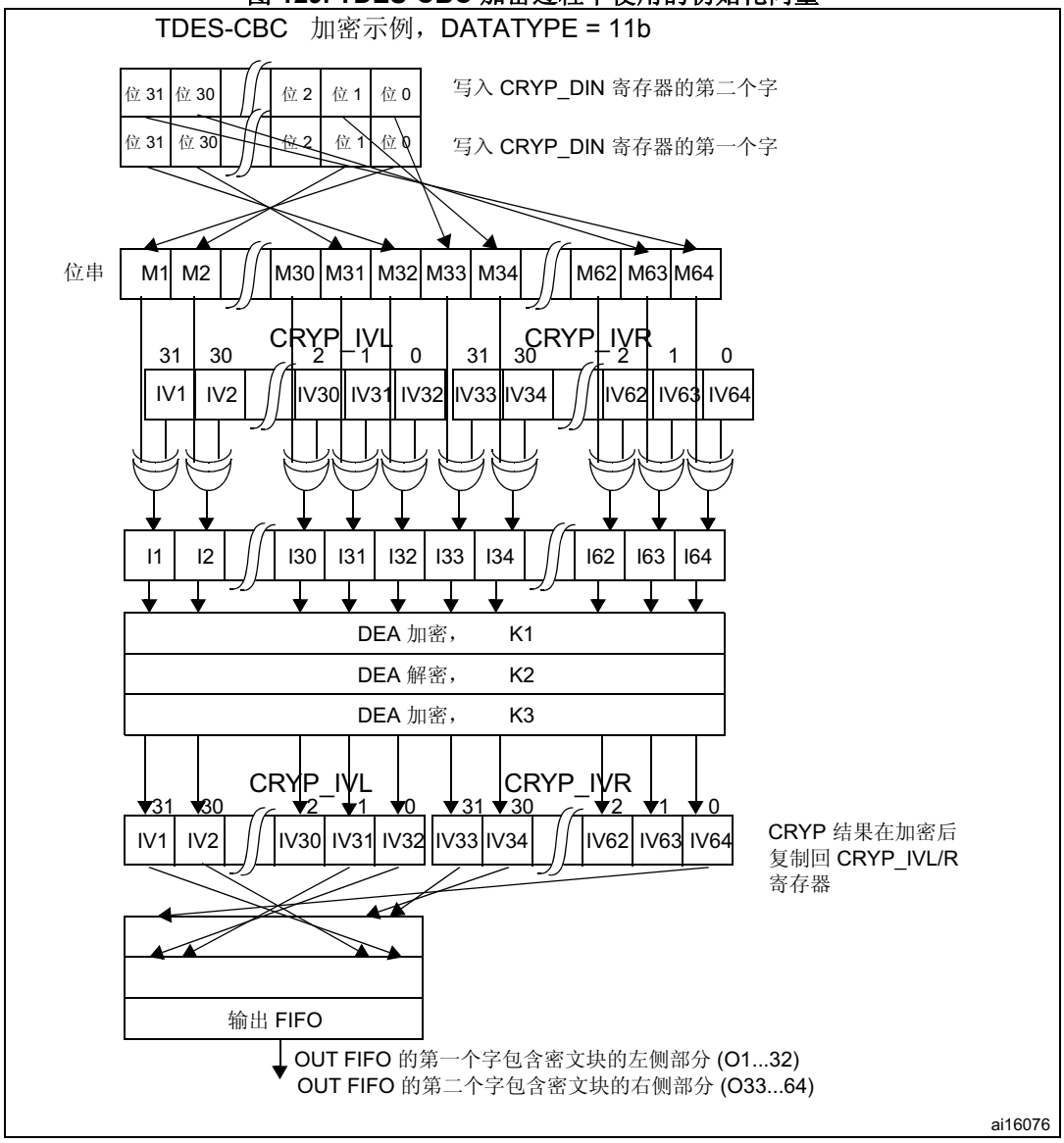
在 AES CBC 加密期间，CRYP_IV0...1(L/R) 位会与一个 128 位数据块进行异或运算，这里的 128 位数据块是根据 DATATYPE 值交换后从 IN FIFO 中弹出的。当 AES 内核的输出可用时，该输出值会复制到 CRYP_IV0...1(L/R) 向量，之后，这个新值与 IN FIFO 弹出的下一个 128 位数据块进行异或运算，以此类推。

在 AES CBC 解密期间，CRYP_IV0...1(L/R) 位首先会与 AES 内核输出的 128 位数据块进行异或运算，然后异或运算后的结果根据 DATATYPE 值进行交换并压入 OUT FIFO。异或运算后的结果进行交换并压入 OUT FIFO 后，IN FIFO 的输出会取代 CRYP_IV0...1(L/R) 值，然后 IN FIFO 执行弹出操作，随后可对新的 128 位数据块进行处理。

在 AES CTR 加密或解密期间，AES 内核会将 CRYP_IV0...1(L/R) 位加密。之后，加密的结果会与一个 128 位数据块进行异或运算，这里的 128 位数据块是根据 DATATYPE 值交换后从 IN FIFO 中弹出的。异或运算后的结果进行交换并压入 OUT FIFO 后，CRYP_IV0...1(L/R) 值 (32 LSB) 的计数器部分会递增。

当 CRYP_SR 寄存器的 BUSY 位 = 1b 时，针对 CRYP_IV0...1(L/R) 寄存器的任何写操作都会被忽略（CRYP_IV0...1(L/R) 寄存器内容不会被修改）。因此，在修改初始化向量之前，必须检查 BUSY 位 = 0b。

图 129. TDES-CBC 加密过程中使用的初始化向量



20.3.5 CRYP 忙碌状态

当输入 FIFO 中有充足的数据（至少有 2 个字可用于 DES 或 TDES 算法模式，至少有 4 个字可用于 AES 算法模式）、输出 FIFO 中有充足的自由空间（至少有 2 个 (DES/TDES) 或 4 个 (AES) 字位置），以及 CRYP_CR 寄存器中的位 CRYPEN = 1 时，加密处理器会自动开始加密或解密过程（根据 CRYP_CR 寄存器中 ALGODIR 位的值）。

在此过程中，执行三重 DES 算法需要 48 个 AHB2 时钟周期，执行简易 DES 算法需 16 个 AHB2 时钟周期，而执行密钥长度为 128、192 或 256 位的 AES 算法则分别需要 14、16 或 18 个 AHB2 时钟周期。在整个过程中，CRYP_SR 寄存器的 BUSY 位始终置“1”。完成此过程后，CRYP 内核会将两个 (DES/TDES) 或四个 (AES) 字写入输出 FIFO，并将 BUSY 位清零。在 CBC、CTR 模式下，还会更新初始化向量 CRYP_IVx(L/R)R (x = 0..3)。

当加密处理器繁忙时（CRYP_SR 寄存器中的位 BUSY = 1b），会忽略针对密钥寄存器（CRYP_Kx(L/R)R，x = 0..3）、初始化寄存器（CRYP_IVx(L/R)R, x = 0..3）或 CRYP_CR 寄存器的位 [9:2] 的写操作，且不会修改这些寄存器。因此，不能在加密处理器处理数据块时修改其配置。不过，可以在 BUSY = 1 时将 CRYPEN 位清零，这种情况下，只有完成正在进行的 DES、TDES 或 AES 处理过程并将两个/四个字的结果写入输出 FIFO 后才能将 BUSY 位清零。

注： *在 DES 或 TDES 模式下处理某个块时，如果输出 FIFO 已满并且输入 FIFO 至少含一个新块，则输入 FIFO 会弹出新块且 BUSY 位保持置 1，直到有足够的空间可将这个新块存储到输出 FIFO。*

20.3.6 加密或解密执行步骤

初始化

1. 初始化外设（操作顺序并不重要，除非密钥准备用于 AES-ECB 或 AES-CBC 解密。在准备密钥之前必须输入密钥大小和密钥值，准备好密钥后，必须立即配置相应算法）：
   a) 使用 CRYP_CR 寄存器中的 KEYSIZE 位配置密钥大小（129 位、192 位或 256 位，仅限 AES）
   b) 将对称密钥写入 CRYP_KxL/R 寄存器（需写入 2 到 8 个寄存器，具体取决于算法）
   c) 使用 CRYP_CR 寄存器中的 DATATYPE 位配置数据类型（1 位、8 位、16 位或 32 位）
   d) 在 AES-ECB 或 AES-CBC 解密的情况下，必须准备以下密钥：将 CRYP_CR 寄存器中的 ALGOMODE 位置为“111”来配置密钥准备模式。随后向 CRYPEN 位写入”1“：BUSY 位随即置 1。等待 BUSY 位返回 0（CRYPEN 位也会自动清零）：已准备好密钥供解密使用
   e) 使用 CRYP_CR 寄存器中的 ALGOMODE 位配置算法和链接（在 ECB/CBC 中为 DES/TDES，在 ECB/CBC/CTR/GCM/CCM 中为 AES）
   f) 使用 CRYP_CR 寄存器中的 ALGODIR 位配置方向（加密/解密）
   g) 将初始化向量写入 CRYP_IVxL/R 寄存器（仅在 CBC 或 CTR 模式下）
2. 向 CRYP_CR 寄存器中的 FFLUSH 位写入 1，刷新 IN 和 OUT FIFO

DMA 用于存储器数据传入和传出时的处理过程

1. 将 DMA 控制器配置为传输存储器中的输入数据。传输长度为消息的长度。当消息填充并非由外设进行管理时，消息长度必须为整个数量的数据块。数据传输均在突发模式下进行。AES 中的突发长度为 4 个字，而 DES/TDES 中的突发长度为 2 个字或 4 个字。应将 DMA 配置为在完成输出数据传输时设置一个中断，以指示处理过程已结束。
2. 通过向 CRYPEN 位写入 1 来使能加密处理器。将 CRYP_DMACR 寄存器中的 DIEN 和 DOEN 位置 1，以使能 DMA 请求。
3. 所有传输和处理过程均由 DMA 和加密处理器管理。DMA 中断表示处理过程已完成。两个 FIFO 通常均为空，且 BUSY = 0。

在中断期间通过 CPU 传输数据的处理过程

1. 将 CRYP_IMSCR 寄存器中的 INIM 和 OUTIM 位置 1，以使能中断。
2. 将 CRYP_CR 寄存器中的 CRYPEN 位置 1，以使能加密处理器。
3. 在中断中管理输入数据：将输入消息加载到 IN FIFO。一次性可加载 2 个字或 4 个字，或者加载数据直到 FIFO 已满。当消息的最后一个字进入 FIFO 时，可通过将 INIM 位清零来禁止中断。
4. 在中断中管理输出数据：读取 OUT FIFO 中的输出消息。一次可读取 1 个块（2 个字或 4 个字），或者读取数据直到 FIFO 为空。读取最后一个字后，INIM=0、BUSY=0 且两个 FIFO 均为空（IFEM=1 且 OFNE=0）。将 OUTIM 位清零可禁止中断，而将 CRYPEN 位清零可禁止外设。

不使用 DMA 也不使用中断时的处理过程

1. 将 CRYP_CR 寄存器中的 CRYPEN 位置 1，以使能加密处理器。
2. 将首个块写入输入 FIFO（2 到 8 个字）。
3. 重复以下步骤，直到处理完整个信息：
   a) 等待 OFNE=1，然后读取输出 FIFO（读取 1 个块，或 FIFO 为空为止）
   b) 等待 IFNF=1，然后写入 IN FIFO（写入 1 个块，或 FIFO 已满为止）
4. 处理过程结束时，BUSY=0 且两个 FIFO 均为空（IFEM=1 且 OFNE=0）。将 CRYPEN 位清零可禁止外设。

20.3.7 上下文交换

如果因 OS 发起的一项新任务需要上下文资源而需要进行上下文交换，则须执行以下任务来恢复完整的上下文（以使用 DMA 为例）：

在 AES 和 DES 的情况下

1. 保存上下文。
   a) 将 CRYP_DMACR 寄存器中的 DIEN 位清零，以停止 DMA 对 IN FIFO 的数据传输。
   b) 等待 IN 和 OUT FIFO 均为空（CRYP_SR 寄存器中的 IFEM=1 且 OFNE=0），并且 BUSY 位清零。
   c) 将 CRYP_DMACR 寄存器中的 DOEN 位清零并将 CRYPEN 位清零，以停止 DMA 对 OUT FIFO 的数据传输。
   d) 保存当前配置（CRYP_CR 寄存器中的位 [9:2] 和位 19）和初始化向量（如果不是在 ECB 模式下）。密钥的值必须已经存在于存储器中。根据需要保存 DMA 状态（IN 和 OUT 消息的指针、保留字节数等）。

   使用 GCM/GMAC 或 CCM/CMAC 算法时应保存其它位：

   – CRYP_CR 寄存器中的位 [17:16]
   – 上下文交换寄存器：

     使用 GCM/GMAC 或 CCM/CMAC 算法时为 CRYP_CSGCMCCM0..7

     使用 GCM/GMAC 算法时为 CRYP_CSGCM0..7。
2. 配置和执行其它处理过程。

3. 上下文恢复。
   a) 使用保存的配置按*第 530 页上的第 20.3.6 节：加密或解密执行步骤*的*初始化*中所述配置处理器。针对 AES-ECB 或 AES-CBC 解密，必须再次准备密钥。
   b) 根据需要重新配置 DMA 控制器，以便传输余下的消息。
   c) 将 CRYPEN 位置 1 以使能处理器，将 DIEN 和 DOEN 位置 1，以使能 DMA 请求。

在 TDES 的情况下

可采用与 AES 相同的方式进行 TDES 下的上下文交换。不过，由于这种情况下的输入 FIFO 可包含多达 4 个未处理的块，而且每个块的处理时间很长，因此，在某些情况下，不用等待 IN FIFO 为空就可中断处理过程，这样能够节省时间。

1. 保存上下文。
   a) 将 CRYP_DMACR 寄存器中的 DIEN 位清零，以停止 DMA 对 IN FIFO 的数据传输。
   b) 将 CRYPEN 位清零，以禁止处理器（处理过程会在当前块的末尾停止）。
   c) 等待 OUT FIFO 为空（CRYP_SR 寄存器中的 OFNE=0），并且 BUSY 位清零。
   d) 向 CRYP_DMACR 寄存器中的 DOEN 位写入 0，以停止 DMA 对 OUT FIFO 的数据传输。
   e) 保存当前配置（CRYP_CR 寄存器中的位 [9:2] 和位 19）和初始化向量（如果不是在 ECB 模式下）。密钥的值必须已经存在于存储器中。根据需要保存 DMA 状态（IN 和 OUT 消息的指针、保留字节数等）。读回 IN FIFO 中加载的尚未处理的数据，并将其保存到存储器中，直到 FIFO 为空。

注： *在 GCM/GMAC 或 CCM/CMAC 模式下，还应保存 CRYP_CR 寄存器的位 [17:16]。*

2. 配置和执行其它处理过程。
3. 上下文恢复。
   a) 使用保存的配置按*第 530 页上的第 20.3.6 节：加密或解密执行步骤*的*初始化*中所述配置处理器。针对 AES-ECB 或 AES-CBC 解密，必须再次准备密钥。
   b) 将保存上下文期间保存的数据写入 IN FIFO。
   c) 根据需要重新配置 DMA 控制器，以便传输余下的消息。
   d) 将 CRYPEN 位置 1 以使能处理器，将 DIEN 和 DOEN 位置 1，以使能 DMA 请求。

20.4 CRYP 中断

CRYP 可产生两个可单独屏蔽的中断源。这两个中断源合为同一个中断信号，而该中断信号是 CRYP 发出的唯一中断信号，用于驱动 NVIC（嵌套向量中断控制器）。这一组合中断是两个单独屏蔽的中断源的或运算结果。如果下面列出的各个中断中的任何一个中断产生，则此组合中断即会产生。

通过更改 CRYP_IMSCR 寄存器中的屏蔽位，可单独使能或禁止各个中断源。将相应的屏蔽位置“1”以使能中断。

关于各个中断源的状态，通过 CRYP_RISR 寄存器可以读取原始中断状态，通过 CRYP_MISR 寄存器可以读取屏蔽中断状态。

输出 FIFO 服务中断 - OUTMIS

当输出 FIFO 中存在一个或多个（32 位字）数据项时，即会产生输出 FIFO 服务中断。通过读取输出 FIFO 的数据，直到读完所有有效（32 位）字，即可将此中断清除（即，该中断的状态与 OFNE（输出 FIFO 非空）标志一致）。

输出 FIFO 服务中断 OUTMIS 不是通过 CRYP 使能位使能。因此，如果输出 FIFO 非空，即使禁止 CRYP 之后也不会强制 OUTMIS 信号为低电平。

输入 FIFO 服务中断 - INMIS

当输入 FIFO 中少于四个字时，会产生输入 FIFO 服务中断。对输入 FIFO 执行写操作直到其中所含字不小于四字，这样即可将此中断清除。

输入 FIFO 服务中断 INMIS 通过 CRYP 使能位使能。因此，禁止 CRYP 之后，即使输入 FIFO 为空，INMIS 信号也为低（无效）。

20.5 CRYP DMA 接口

加密处理器可使用一个接口连接 DMA 控制器。DMA 操作通过 CRYP DMA 控制寄存器 CRYP_DMACR 进行控制。

突发传输请求信号和单次传输请求信号并不相互排斥。这两种信号可同时产生。例如，当 OUT FIFO 中存在 6 个字时，会产生突发传输请求和单次传输请求。突发传输 4 个字之后，将只产生单次传输请求来传输余下的 2 个字。当数据流中待接收的剩余字数少于突发传输字数时，这一特性非常有用。

在产生相关的 DMA 清除信号之前，仍会产生各个请求信号。禁止请求清除信号之后，可再次激活某个请求信号，具体取决于上述条件。如果已禁止 CRYP 外设并且已将 DMA 使能位清零（CRYP_DMACR 寄存器中的 DIEN 位用于 IN FIFO，DOEN 位用于 OUT FIFO），则会禁止所有请求信号。

注： *DMA 控制器必须配置为执行不多余 4 字的突发传输。否则可能会丢失一些数据。*

为了在装满 IN FIFO 之前让 DMA 控制器清空 OUT FIFO，OUTDMA 通道的优先级应高于 INDMA 通道。

20.6 CRYP 寄存器

加密内核连接若干控制和状态寄存器、八个密钥寄存器和四个初始化向量寄存器。

20.6.1 CRYP 控制寄存器 (CRYP_CR)

CRYP 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.	ALGO MODE [3]	Res.	GCM_CCMPH	
												rw		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRYPEN	FFLUSH	Res.	Res.	Res.	Res.	KEYSIZE		DATATYPE		ALGOMODE[2:0]			ALGODIR	Res.	Res.
rw	w					rw	rw	rw	rw	rw	rw	rw	rw		

位 31:20 保留，由硬件强制清零。

位 18 保留，由硬件强制清零。

位 17:16 GCM_CCMPH[1:0]：如果未设置“GCM 或 CCM 算法”则该位无影响

00：GCM_CCM 初始化阶段
01：GCM_CCM 标头阶段
10：GCM_CCM 有效负载阶段
11：GCM_CCM 最后阶段

位 15 **CRYPEN**：加密处理器使能 (Cryptographic processor enable)

0：CRYP 处理器禁止
1：CRYP 处理器使能

注： 当密钥准备过程结束 (ALGOMODE=111b) 或执行 GCM_CCM 初始化阶段，硬件会自动将 CRYPEN 位清零

位 14 **FFLUSH**：FIFO 刷新 (FIFO flush)

CRYPEN = 0 时，在此位中写入 1 可刷新 IN 和 OUT FIFO（即，将复位 FIFO 的读指针和写指针）。若在此位中写入 0 则不会产生影响。
CRYPEN = 1 时，在该位中写入 0 或 1 都不会产生影响。
读取该位时，始终返回 0。

位 13:10 保留，由硬件强制清零。

位 9:8 **KEYSIZE[1:0]**：密钥大小选择 (Key size selection)（仅限 AES 模式）

此位字段定义了 AES 加密内核使用的密钥位长度。此位字段在 DES 或 TDES 模式中可忽略。

00：128 位密钥长度
01：192 位密钥长度
10：256 位密钥长度
11：保留，不使用该值

位 7:6 **DATATYPE[1:0]**：数据类型选择 (Data type selection)

此位字段定义了在 CRYP_DIN 寄存器中输入的数据格式（请参见*第 20.3.3 节：数据类型*）。

00：32 位数据。不交换各个字。压入 IN FIFO（或 OUT FIFO 弹出）的首个字形成数据块的位 1...32，第二个字形成数据块的位 33...64。

01：16 位数据或半字。可将压入 IN FIFO（或 OUT FIFO 弹出）的每个字视为 2 个已相互交换的半字。

10：8 位数据或字节。可将压入 IN FIFO（或 OUT FIFO 弹出）的每个字视为 4 个已相互交换的字。

11：位数据或位串。可将压入 IN FIFO（或 OUT FIFO 弹出）的每个字视为 32 个已相互交换的位（字符串的首个位在位置 0）。

位 19 和 5:3 **ALGOMODE[3:0]**：算法模式 (Algorithm mode)

0000：TDES-ECB（三重 DES 电子密码本）：数据块之间无反馈。不使用初始化向量 (CRYP_IV0(L/R))，而是使用三个密钥向量（K1、K2 和 K3）（不使用 K0）。

0001：TDES-CBC（三重 DES 加密分组链接）：输出块会与后续输入块先进行异或运算，然后再执行相应算法。必须对初始化向量 (CRYP_IV0L/R) 进行初始化，使用三个密钥向量（K1、K2 和 K3）（不使用 K0）。

0010：DES-ECB（简易 DES 电子密码本）：数据块之间无反馈。不使用初始化向量 (CRYP_IV0L/R)，仅使用一个密钥向量 (K1)（不使用 K0、K2 和 K3）。

0011：DES-CBC（简易 DES 加密分组链接）：输出块会与后续输入块先进行异或运算，然后再执行相应算法。必须对初始化向量 (CRYP_IV0L/R) 进行初始化。仅使用一个密钥向量 (K1)（不使用 K0、K2 和 K3）。

0100：AES-ECB（AES 电子密码本）：数据块之间无反馈。不使用初始化向量 (CRYP_IV0L/R...1L/R)。使用所有四个密钥向量 (K0...K3)。

0101：AES-CBC（AES 加密分组链接）：输出块会与后续输入块先进行异或运算，然后再执行相应算法。必须对初始化向量 (CRYP_IV0L/R...1L/R) 进行初始化。使用所有四个密钥向量 (K0...K3)。

0110：AES-CTR（AES 计数器模式）：输出块会与后续输入块先进行异或运算，然后再执行相应算法。必须对初始化向量 (CRYP_IV0L/R...1L/R) 进行初始化。使用所有四个密钥向量 (K0...K3)。CTR 解密与 CTR 加密并无不同，因为内核始终会对当前的计数器块加密以产生密钥数据流，该密钥数据流与输入中的明文或密码进行异或运算。因此，ALGOMODE = 110b 时 ALGODIR 可忽略，而且不得提供（准备）解密密钥。

0111：针对解密模式准备 AES 密钥。CRYPEN = 1 时写入该值会立即启动 AES 轮来准备密钥。秘密密钥必须已预先加载到 K0...K3 寄存器。准备密钥期间会将 CRYP_SR 寄存器的 BUSY 位置 1。完成密钥处理之后，得到的密钥会复制到 K0...K3 寄存器，并且 BUSY 位会清零。

1000：Galois 计数器模式 (GCM)。此算法模式同样适用于 GMAC 算法。

1001：CBC-MAC 计数器模式 (CCM)。此算法模式同样适用于 CMAC 算法。

位 2 **ALGODIR**：算法方向 (Algorithm direction)

0：加密

1：解密

位 1:0 保留，必须保持为零。

注： *当 BUSY=1 时写入 KEYSIZE、DATATYPE、ALGOMODE 和 ALGODIR 位无效。这些位只能在 BUSY=0 时进行配置。*

FFLUSH 位必须在 BUSY=0 时才能置 1。如果在其他情况下将 FFLUSH 位置 1，虽然 FIFO 会进行刷新，但在刷新操作后可能会将所处理的数据块压入输出 FIFO，从而产生 FIFO 非空的情况。

20.6.2 CRYP 状态寄存器 (CRYP_SR)

CRYP status register

偏移地址：0x04

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.	BUSY	OFFU	OFNE	IFNF	IFEM
											r	r	r	r	r

位 31:5 保留，必须保持复位值

位 4 **BUSY**：忙碌位 (Busy bit)

0：CRYP 内核当前没有处理任何数据。具体原因是：

– CRYP 内核已禁止（CRYP_CR 寄存器的 CRYPEN=0）且最后的处理过程已完成，或者

– CRYP 内核正在等待输入 FIFO 的数据足够多或者输出 FIFO 的自由空间足够大（即，在这两种情况下，DES 中至少有 2 个字，AES 中至少有 4 个字）。

1：CRYP 内核目前正在处理一个数据块或准备密钥（用于 AES 解密）。

位 3 **OFFU**：输出 FIFO 已满 (Output FIFO full)

0：输出 FIFO 未满

1：输出 FIFO 已满

位 2 **OFNE**：输出 FIFO 非空 (Output FIFO not empty)

0：输出 FIFO 为空

1：输出 FIFO 非空

位 1 **IFNF**：输入 FIFO 未满 (Input FIFO not full)

0：输入 FIFO 已满

1：输入 FIFO 未满

位 0 **IFEM**：输入 FIFO 为空 (Input FIFO empty)

0：输入 FIFO 非空

1：输入 FIFO 为空

20.6.3 CRYP 数据输入寄存器 (CRYP_DIN)

CRYP data input register

偏移地址：0x08

复位值：0x0000 0000

CRYP_DIN 寄存器为数据输入寄存器。其宽度为 32 位。最多可将四个 64 位 (TDES) 或两个 128 位 (AES) 明文（加密时）或密文（解密时）块输入到输入 FIFO，一次输入一个 32 位字。

写入 FIFO 的首个字为输入块的 MSB。最后写入的是输入块的 LSB。不论数据交换如何：

- 在 DES/TDES 模式下：块是指从位 1（最左位）到位 64（最右位）的位序列。位 1 对应输入 FIFO 中的首个字的 MSB（位 31），位 64 对应输入 FIFO 中的第二个字的 LSB（位 0）。
- 在 AES 模式下：数据块是指从位 0（最左位）到位 127（最右位）的位序列。位 0 对应写入 FIFO 中的首个字的 MSB（位 31），位 127 对应写入 FIFO 中的第四个字的 LSB（位 0）。

为了满足不同的数据大小，通过配置 CRYP_CR 寄存器中的 DATATYPE 位，写入 CRYP_DIN 寄存器中的数据可在处理之前进行交换。更多详细信息，请参见*第 525 页上的第 20.3.3 节：数据类型*。

写入 CRYP_DIN 寄存器时，数据会压入输入 FIFO。如果 DES/TDES 模式下至少有两个 32 位字（或者 AES 模式下至少有四个 32 位字）已压入输入 FIFO，并且输出 FIFO 中至少有 2 个字的自由空间，则 CRYP 引擎会启动加密或解密过程。在 DES/TDES 模式下，此过程会从输入 FIFO 取出两个 32 位字（在 AES 模式下则取出四个 32 位字），并且每一轮处理过程都会将两个 32 位字（在 AES 模式则为 4 个字）传输到输出 FIFO。

读取 CRYP_DIN 寄存器时：

- 如果 CRYPEN = 0，则 FIFO 会弹出数据，之后，返回输入 FIFO 中的数据（从最早的数据（首先读取）到最新的数据（最后读取）。在执行每次读取操作前必须检查 IFEM 标志，以确保 FIFO 非空。
- 如果 CRYPEN = 1，则会返回一个未定义值。

读取 CRYP_DIN 寄存器一次或多次之后，在处理新数据之前必须通过将 FFLUSH 位置 1 来刷新 FIFO。

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

位 31:0 **DATAIN**：数据输入 (Data input)

读取 = 在 CRYPEN = 0 时返回输入 FIFO 内容，否则返回不确定值。

写入 = 写入输入 FIFO。

20.6.4 CRYP 数据输出寄存器 (CRYP_DOUT)

CRYP data output register

偏移地址：0x0C

复位值：0x0000 0000

CRYP_DOUT 寄存器为数据输出寄存器。该寄存器为 32 位宽度的只读寄存器。最多可接收来自输出 FIFO 的四个 64 位（TDES 模式）或两个 128 位（AES 模式）密文（加密时）或明文（解密时）块，一次接收一个 32 位字。

与输入数据相似，输出块的 MSB 是从输出 FIFO 读取的首个字。最后读取的是输出块的 LSB。不论数据交换如何：

- 在 DES/TDES 模式下：位 1（最左位）对应从输出 FIFO 中读出的首个字的 MSB（位 31），位 64（最右位）对应从输出 FIFO 中读出的第二个字的 LSB（位 0）。
- 在 AES 模式下：位 0（最左位）对应从输出 FIFO 中读出的首个字的 MSB（位 31），位 127（最右位）对应从输出 FIFO 中读出的第四个字的 LSB（位 0）。

为了满足不同的数据大小，通过配置 CRYP_CR 寄存器中的 DATATYPE 位，数据可在处理之后进行交换。更多详细信息，请参见*第 525 页上的第 20.3.3 节：数据类型*。

读取 CRYP_DOUT 寄存器时，会返回最后输入输出 FIFO 的数据（由读指针指定）。

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

位 31:0 **DATAOUT**：数据输出 (Data output)

读取 = 返回输出 FIFO 中的内容。

写入 = 无操作。

20.6.5 CRYP DMA 控制寄存器 (CRYP_DMACR)

CRYP DMA control 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOEN	DIEN
														rw	rw

位 31:2 保留，必须保持复位值

位 1 **DOEN**：DMA 输出使能 (DMA output enable)
0：禁止用于传出数据传输的 DMA
1：使能用于传出数据传输的 DMA

位 0 **DIEN**：DMA 输入使能 (DMA input enable)
0：禁止用于传入数据传输的 DMA
1：使能用于传入数据传输的 DMA

20.6.6 CRYP 中断屏蔽置位/清零寄存器 (CRYP_IMSCR)

CRYP interrupt mask set/clear register

偏移地址：0x14

复位值：0x0000 0000

CRYP_IMSCR 寄存器为中断屏蔽置位/清零寄存器。该寄存器为读/写寄存器。执行读取操作时，此寄存器会给出相关中断屏蔽的当前值。向该特定位写入 1 时会设置屏蔽，从而使能要读取的中断。向该位写入 0 时会清除相应的屏蔽。外设复位后，所有位均清零。

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

位 31:2 保留，必须保持复位值

位 1 **OUTIM**：输出 FIFO 服务中断屏蔽 (Output FIFO service interrupt mask)
0：屏蔽输出 FIFO 服务中断
1：不屏蔽输出 FIFO 服务中断

位 0 **INIM**：输入 FIFO 服务中断屏蔽 (Input FIFO service interrupt mask)
0：屏蔽输入 FIFO 服务中断
1：不屏蔽输入 FIFO 服务中断

20.6.7 CRYP 原始中断状态寄存器 (CRYP_RISR)

CRYP raw interrupt status register

偏移地址：0x18

复位值：0x0000 0001

CRYP_RISR 寄存器为原始中断状态寄存器。该寄存器为只读寄存器。执行读取操作时，此寄存器会在屏蔽前给出相关中断的当前原始状态。执行写入操作不起作用。

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

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.	OUTRIS	INRIS
														r	r

位 31:2 保留，必须保持复位值

位 1 **OUTRIS**：输出 FIFO 服务原始中断状态 (Output FIFO service raw interrupt status)
给出屏蔽输出 FIFO 服务中断之前的原始中断状态。
0：原始中断未挂起
1：原始中断挂起

位 0 **INRIS**：输入 FIFO 服务原始中断状态 (Input FIFO service raw interrupt status)
给出屏蔽输入 FIFO 服务中断之前的原始中断状态。
0：原始中断未挂起
1：原始中断挂起

20.6.8 CRYP 屏蔽中断状态寄存器 (CRYP_MISR)

CRYP masked interrupt status register

偏移地址：0x1C

复位值：0x0000 0000

CRYP_RISR 寄存器为屏蔽中断状态寄存器。该寄存器为只读寄存器。执行读取操作时，此寄存器会在屏蔽前给出相关中断的当前屏蔽状态。执行写入操作不起作用。

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

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.	OUTMIS	INMIS
														r	r

位 31:2 保留，必须保持复位值

位 1 **OUTMIS**：输出 FIFO 服务屏蔽中断状态 (Output FIFO service masked interrupt status)
给出屏蔽输出 FIFO 服务中断之后的中断状态。
0：中断未挂起
1：中断挂起

位 0 **INMIS**：输入 FIFO 服务屏蔽中断状态 (Input FIFO service masked interrupt status)
给出屏蔽输入 FIFO 服务中断之后的中断状态。
0：中断未挂起
1：CRYPEN = 1 时中断挂起

20.6.9 CRYP 密钥寄存器 (CRYP_K0...3(L/R)R)

CRYP key registers

偏移地址：0x20 到 0x3C

复位值：0x0000 0000

这些寄存器中含有加密密钥。

在 TDES 模式下，密钥为 64 位二进制值（按从左到右编号，最左位为位 1），分别命名为 K1、K2 和 K3（不使用 K0），各个密钥由 56 个信息位和 8 个奇偶校验位组成。奇偶校验位保留用于错误检测，当前块不使用这些位。因此，不使用各个 64 位密钥值 Kx[1:64] 的位 8、16、24、32、40、48、56 和 64。

在 AES 模式下，可将密钥视为一个 128 位、192 位或 256 位长的位序列，$k_0k_1k_2...k_{127/191/255}$（$k_0$ 为最左位）。AES 密钥按如下方式输入寄存器中：

- 对于 AES-128：$k_0..k_{127}$ 对应 $b_{127}..b_0$（不使用 $b_{255}..b_{128}$），
- 对于 AES-192：$k_0..k_{191}$ 对应 $b_{191}..b_0$（不使用 $b_{255}..b_{192}$），
- 对于 AES-256：$k_0..k_{255}$ 对应 $b_{255}..b_0$。

在任何情况下，b_0 为最右位。

CRYP_K0LR（偏移地址：0x20）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
b255	b254	b253	b252	b251	b250	b249	b248	b247	b246	b245	b244	b243	b242	b241	b240
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
b239	b238	b237	b236	b235	b234	b233	b232	b231	b230	b229	b228	b227	b226	b225	b224
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

CRYP_K0RR（偏移地址：0x24）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
b223	b222	b221	b220	b219	b218	b217	b216	b215	b214	b213	b212	b211	b210	b209	b208
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
b207	b206	b205	b204	b203	b202	b201	b200	b199	b198	b197	b196	b195	b194	b193	b192
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

CRYP_K1LR（偏移地址：0x28）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
k1.1 b191	k1.2 b190	k1.3 b189	k1.4 b188	k1.5 b187	k1.6 b186	k1.7 b185	k1.8 b184	k1.9 b183	k1.10 b182	k1.11 b181	k1.12 b180	k1.13 b179	k1.14 b178	k1.15 b177	k1.16 b176
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
k1.17 b175	k1.18 b174	k1.19 b173	k1.20 b172	k1.21 b171	k1.22 b170	k1.23 b169	k1.24 b168	k1.25 b167	k1.26 b166	k1.27 b165	k1.28 b164	k1.29 b163	k1.30 b162	k1.31 b161	k1.32 b160
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

CRYP_K1RR（偏移地址：0x2C）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
k1.33 b159	k1.34 b158	k1.35 b157	k1.36 b156	k1.37 b155	k1.38 b154	k1.39 b153	k1.40 b152	k1.41 b151	k1.42 b150	k1.43 b149	k1.44 b148	k1.45 b147	k1.46 b146	k1.47 b145	k1.48 b144
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
k1.49 b143	k1.50 b142	k1.51 b141	k1.52 b140	k1.53 b139	k1.54 b138	k1.55 b137	k1.56 b136	k1.57 b135	k1.58 b134	k1.59 b133	k1.60 b132	k1.61 b131	k1.62 b130	k1.63 b129	k1.64 b128
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

CRYP_K2LR（偏移地址：0x30）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
k2.1 b127	k2.2 b126	k2.3 b125	k2.4 b124	k2.5 b123	k2.6 b122	k2.7 b121	k2.8 b120	k2.9 b119	k2.10 b118	k2.11 b117	k2.12 b116	k2.13 b115	k2.14 b114	k2.15 b113	k2.16 b112
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
k2.17 b111	k2.18 b110	k2.19 b109	k2.20 b108	k2.21 b107	k2.22 b106	k2.23 b105	k2.24 b104	k2.25 b103	k2.26 b102	k2.27 b101	k2.28 b100	k2.29 b99	k2.30 b98	k2.31 b97	k2.32 b96
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

CRYP_K2RR（偏移地址：0x34）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
k2.33 b95	k2.34 b94	k2.35 b93	k2.36 b92	k2.37 b91	k2.38 b90	k2.39 b89	k2.40 b88	k2.41 b87	k2.42 b86	k2.43 b85	k2.44 b84	k2.45 b83	k2.46 b82	k2.47 b81	k2.48 b80
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
k2.49 b79	k2.50 b78	k2.51 b77	k2.52 b76	k2.53 b75	k2.54 b74	k2.55 b73	k2.56 b72	k2.57 b71	k2.58 b70	k2.59 b69	k2.60 b68	k2.61 b67	k2.62 b66	k2.63 b65	k2.64 b64
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

CRYP_K3LR（偏移地址：0x38）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
k3.1 b63	k3.2 b62	k3.3 b61	k3.4 b60	k3.5 b59	k3.6 b58	k3.7 b57	k3.8 b56	k3.9 b55	k3.10 b54	k3.11 b53	k3.12 b52	k3.13 b51	k3.14 b50	k3.15 b49	k3.16 b48
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
k3.17 b47	k3.18 b46	k3.19 b45	k3.20 b44	k3.21 b43	k3.22 b42	k3.23 b41	k3.24 b40	k3.25 b39	k3.26 b38	k3.27 b37	k3.28 b36	k3.29 b35	k3.30 b34	k3.31 b33	k3.32 b32
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

CRYP_K3RR（偏移地址：0x3C）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
k3.33 b31	k3.34 b30	k3.35 b29	k3.36 b28	k3.37 b27	k3.38 b26	k3.39 b25	k3.40 b24	k3.41 b23	k3.42 b22	k3.43 b21	k3.44 b20	k3.45 b19	k3.46 b18	k3.47 b17	k3.48 b16
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
k3.49 b15	k3.50 b14	k3.51 b13	k3.52 b12	k3.53 b11	k3.54 b10	k3.55 b9	k3.56 b8	k3.57 b7	k3.58 b6	k3.59 b5	k3.60 b4	k3.61 b3	k3.62 b2	k3.63 b1	k3.64 b0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

注：　当加密处理器繁忙时（CRYP_SR 寄存器中的位 BUSY = 1），会忽略对这些寄存器的写访问。

20.6.10 CRYP 初始化向量寄存器 (CRYP_IV0...1(L/R)R)

CRYP initialization vector registers

偏移地址：0x40 到 0x4C

复位值：0x0000 0000

CRYP_IV0...1(L/R)R 是初始化向量专用的左字和右字寄存器（DES/TDES 模式下为 64 位，AES 模式下为 128 位），可在 CBC（加密分组链接）和计数器 (CTR) 模式下使用。完成每一轮的 TDES 或 AES 内核计算后，CRYP_IV0...1(L/R)R 寄存器会按*第 513 页上的“DES 和 TDES 密码块链接 (DES/TDES-CBC) 模式”*一节、*第 517 页上的“AES 加密分组链接 (AES-CBC) 模式”*一节和*第 519 页上的“AES 计数器模式 (AES-CTR) 模式”*一节所述进行更新。

IV0 是初始化向量的最左位，而 IV63（DES，TDES）或 IV127 (AES) 是初始化向量的最右位。IV1(L/R)R 只能在 AES 模式中使用。

CRYP_IV0LR（偏移地址：0x40）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV0	IV1	IV2	IV3	IV4	IV5	IV6	IV7	IV8	IV9	IV10	IV11	IV12	IV13	IV14	IV15
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
IV16	IV17	IV18	IV19	IV20	IV21	IV22	IV23	IV24	IV25	IV26	IV27	IV28	IV29	IV30	IV31
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

CRYP_IV0RR（偏移地址：0x44）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV32	IV33	IV34	IV35	IV36	IV37	IV38	IV39	IV40	IV41	IV42	IV43	IV44	IV45	IV46	IV47
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
IV48	IV49	IV50	IV51	IV52	IV53	IV54	IV55	IV56	IV57	IV58	IV59	IV60	IV61	IV62	IV63
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

CRYP_IV1LR（偏移地址：0x48）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV64	IV65	IV66	IV67	IV68	IV69	IV70	IV71	IV72	IV73	IV74	IV75	IV76	IV77	IV78	IV79
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
IV80	IV81	IV82	IV83	IV84	IV85	IV86	IV87	IV88	IV89	IV90	IV91	IV92	IV93	IV94	IV95
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

CRYP_IV1RR（偏移地址：0x4C）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV96	IV97	IV98	IV99	IV100	IV101	IV102	IV103	IV104	IV105	IV106	IV107	IV108	IV109	IV110	IV111
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
IV112	IV113	IV114	IV115	IV116	IV117	IV118	IV119	IV120	IV121	IV122	IV123	IV124	IV125	IV126	IV127
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

注： *在 DES/3DES 模式下，仅使用 CRYP_IV0(L/R)。*

当加密处理器繁忙时（CRYP_SR 寄存器中的位 BUSY = 1），会忽略对这些寄存器的写访问。

20.6.11 CRYP 上下文交换寄存器（CRYP_CSGCMCCM0..7R 和CRYP_CSGCM0..7R）

CRYP context swap registers

偏移地址：

- CRYP_CSGCMCCM0..7：0x050 到 0x06C：仅用于 GCM/GMAC 或 CCM/CMAC 算法
- CRYP_CSGCM0..7：0x070 到 0x08C：仅用于 GCM/GMAC 算法

复位值：0x0000 0000

在选择 GCM/GMAC 或 CCM/CMAC 算法后，这些寄存器含有 CRYP 处理器的全部内部寄存器状态。如果高优先级的任务需要使用加密处理器，而该处理器正在执行其他任务，则需要使用上述寄存器执行上下文交换。

出现此类事件时，需要对 CRYP_CSGCMCCM0..7R 和 CRYP_CSGCM0..7R（GCM/GMAC 模式）或 CRYP_CSGCMCCM0..7R（CCM/CMAC 模式）寄存器进行读操作，并且需要将读出的值保存到系统存储器空间。之后，可使用加密处理器执行高优先级的任务，完成加密计算之后，可从存储器中读出保存的上下文并将其写入相应的上下文交换寄存器。

注： *这些寄存器仅在选择了 GCM/GMAC 或 CCM/CMAC 算法模式时才可使用。*

CRYP_CSGCMCCMxR：其中 x=[7:0]

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

CRYP_CSGCMxR：其中 x=[7:0]

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

20.6.12 CRYP 寄存器映射

表 121. CRYP 寄存器映射和复位值

偏移	寄存器名称 复位值	寄存器大小 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 0x00	CRYP_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ALGOMODE[3]	Reserved	GCM_CCMPH		CRYPEN	FFLUSH	Res.	Res.	Res.	Res.	KEYSIZE		DATATYPE		ALOMODE[2:0]			ALGODIR	Res.	Res.
	Reset value															0	0	0	0					0	0	0	0	0	0	0	0		
0x04	CRYP_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.	BUSY	OFFU	OFNE	IFNF	IFEM
	Reset value																												0	0	0	1	1
0x08	CRYP_DIN	DATAIN																															
	Reset value																				0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	CRYP_DOUT	DATAOUT																															
	Reset value																				0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	CRYP_DMACR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOEN	DIEN
	Reset value																															0	0
0x14	CRYP_IMSCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OUTIM	INIM
	Reset value																															0	0
0x18	CRYP_RISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OUTRIS	INRIS
	Reset value																															0	1
0x1C	CRYP_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.	Res.	Res.	Res.	Res.	Res.	OUTMIS	IN%IS
	Reset value																															0	0
0x20	CRYP_K0LR	CRYP_K0LR																															
	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	CRYP_K0RR	CRYP_K0RR																															
	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
...																																	
0x38	CRYP_K3LR	CRYP_K3LR																															
	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 121. CRYP 寄存器映射和复位值（续）

偏移	寄存器名称 复位值	寄存器大小 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
0x3C	CRYP_K3RR	CRYP_K3RR																															
	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	CRYP_IV0LR	CRYP_IV0LR																															
	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	CRYP_IV0RR	CRYP_IV0RR																															
	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	CRYP_IV1LR	CRYP_IV1LR																															
	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	CRYP_IV1RR	CRYP_IV1RR																															
	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	CRYP_ CSGCMCCMR	CRYP_CSGCMCCM0R																															
	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
0x54	CRYP_ CSGCMCCM1R	CRYP_CSGCMCCM1R																															
	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
0x58	CRYP_ CSGCMCCM2R	CRYP_CSGCMCCM2R																															
	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
0x5C	CRYP_ CSGCMCCM3R	CRYP_CSGCMCCM3R																															
	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
0x60	CRYP_ CSGCMCCM4R	CRYP_CSGCMCCM4R																															
	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
0x64	CRYP_ CSGCMCCM5R	CRYP_CSGCMCCM5R																															
	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
0x68	CRYP_ CSGCMCCM6R	CRYP_CSGCMCCM6R																															
	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
0x6C	CRYP_ CSGCMCCM7R	CRYP_CSGCMCCM7R																															
	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
0x70	CRYP_CSGCM0R	CRYP_CSGCM0R																															
	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	CRYP_CSGCM1R	CRYP_CSGCM1R																															
	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
0x78	CRYP_CSGCM2R	CRYP_CSGCM2R																															
	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
0x7C	CRYP_CSGCM3R	CRYP_CSGCM3R																															
	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
0x80	CRYP_CSGCM4R	CRYP_CSGCM4R																															
	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
0x84	CRYP_CSGCM5R	CRYP_CSGCM5R																															
	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 121. CRYP 寄存器映射和复位值（续）

偏移	寄存器名称 复位值	寄存器大小																														
		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	CRYP_CSGCM6R	CRYP_CSGCM6R																														
	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
0x8C	CRYP_CSGCM7R	CRYP_CSGCM7R																														
	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 64 页上的第 2.2.2 节*。

21 散列处理器 (HASH)

除非特别说明，否则本部分适用于整个 STM32F756xx 系列器件。

21.1 散列简介

散列处理器完全兼容安全散列算法（SHA-1、SHA-224 和 SHA-256）、MD5（消息摘要算法 5）散列算法和适合多种应用的 HMAC（密钥散列消息认证码）散列算法。对长达 (2^{64} – 1) 位的消息，散列处理器计算消息摘要（SHA-1 算法为 160 位，SHA-256 算法为 256 位，SHA-224 算法为 224 位，MD5 算法为 128 位），而 HMAC 算法则通过散列函数来对消息进行验证。HMAC 算法存在于两次调用 SHA-1、SHA-224、SHA-256 或 MD5 散列函数。

21.2 散列主要特性

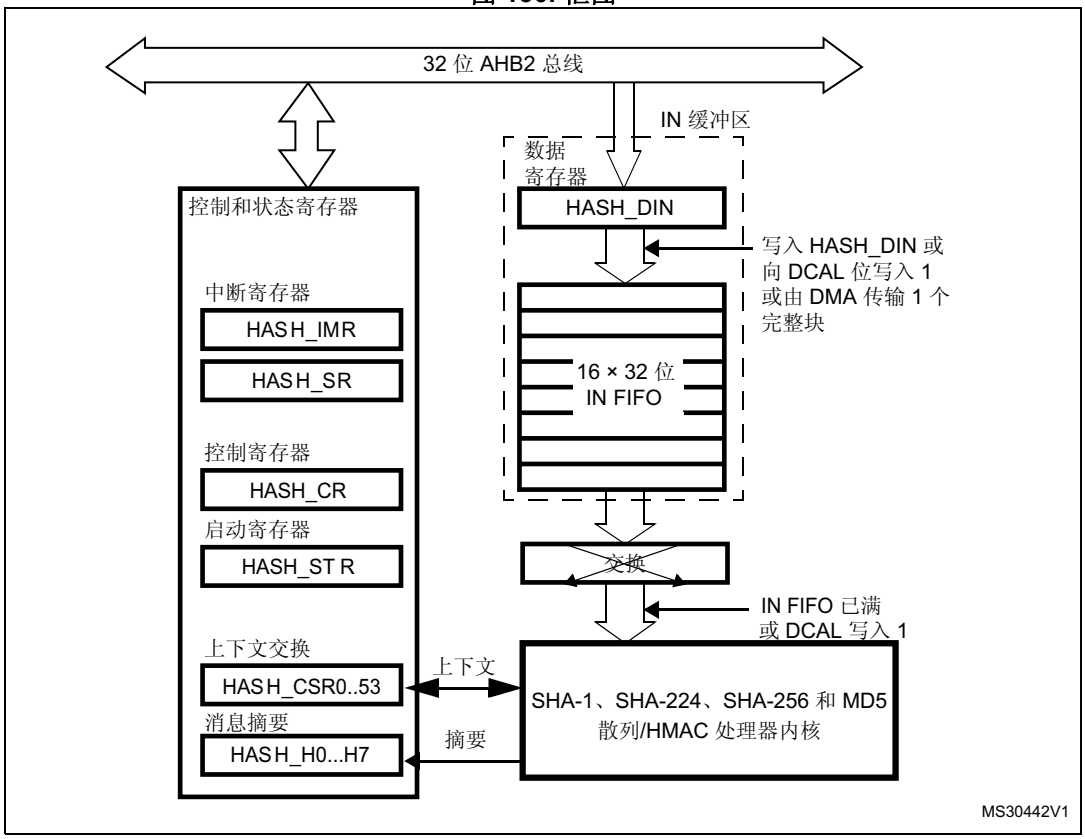
- 适合于数据验证应用，符合以下标准：
  - FIPS PUB 180-2（联邦信息处理标准出版物 180-2）
  - 安全散列标准规范（SHA-1、SHA-224 和 SHA-256）
  - IETF RFC 1321（互联网工程任务组征求意见文档编号 1321）规范 (MD5)
- 快速计算 SHA-1、SHA-224、SHA-256 以及 MD5
- AHB 从外设
- 32 位数据字用于输入数据，支持字、半字、字节和位位串表示法（仅采用小端模式数据表示法）
- 可自动交换，以兼容大端模式 SHA1、SHA-224 和 SHA-256 计算标准（采用小端模式输入位串表示法）
- 可自动填充来完成输入位串，从而适应模数为 512（16 × 32 位）消息摘要计算
- 8 x 32 位字（H0 到 H7）用于输出消息摘要，重载可继续被打断的消息摘要计算
- 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
- 数据流自动控制，支持直接存储器访问 (DMA)

注： *SHA-1、SHA-224 和 SHA-256 算法中定义的填充是，指在 bx1 中添加一个位，然后在 bx0 中添加 N 位，以使总长度与 448%512 同余。然后，将使用 64 位整数（也就是原始消息长度的二进制表示）来补全消息。*
对于此散列处理器，用于输入消息的量是 32 位字，因此必须在消息的末尾处附加信息，也就是最后输入的 32 位字中有效位的数量。

21.3 散列功能说明

图 130 显示了散列处理器的框图。

图 130. 框图



针对计算消息或数据文件的压缩表示，FIPS PUB 180-2 标准和 IETF RFC 1321 出版物分别详细说明了 SHA-1、SHA-224 和 SHA-256 以及 MD5 安全散列算法。输入中提供的任何消息长度低于 2^{64} 位时，SHA-1、SHA-224 和 SHA-256 以及 MD5 将分别生成一个 160 位、224 位、256 位和 128 位的输出位串，称为消息摘要。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散列算法。

SHA-1、SHA-224 和 SHA-256 以及 MD5 安全可靠，因为要找出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，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。有关 SHA-1 或 SHA-224 和 SHA-256 算法的详细信息，请参见 2002 年 8 月 1 日颁布的 FIPS PUB 180-2（联邦信息处理标准出版物 180-2）。

目前该标准的实施采用小端模式输入数据约定。例如，C 位串“abc”在存储器中必须表示为 24 位十六进制值 0x434241。

要由散列处理器处理的消息或数据文件应视为位串。消息长度为消息中的位数（空消息的长度为 0）。可以将该位串的 32 位视为构成了一个 32 位字。请注意，FIPS PUB 180-1 标准使用如下约定：位串从左到右增长，位可以分组为字节（8 位）或字（32 位）（但某些实施还使用半字（16 位），并使用大端模式字节（半字）排序）。此约定主要是对于填充很重要（请参见*第 12 页的第 1.3.4 节：消息填充*）。

21.3.1 处理的持续时间

计算消息的中间块 所花费的时间是：

- 在 SHA-1 中，为 66 个 HCLK 时钟周期
- 在 SHA-224 中，为 50 个 HCLK 时钟周期
- 在 SHA-256 中，为 50 个 HCLK 时钟周期
- 在 MD5 中，为 50 个 HCLK 时钟周期

必须将上述时间与将块的 16 字装入到处理器所需要的时间（对于 512 位块, 至少为 16 个时钟周期）相加。

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 模式中）。与处理中间块相比，此时间可能按如下某个系数增加：

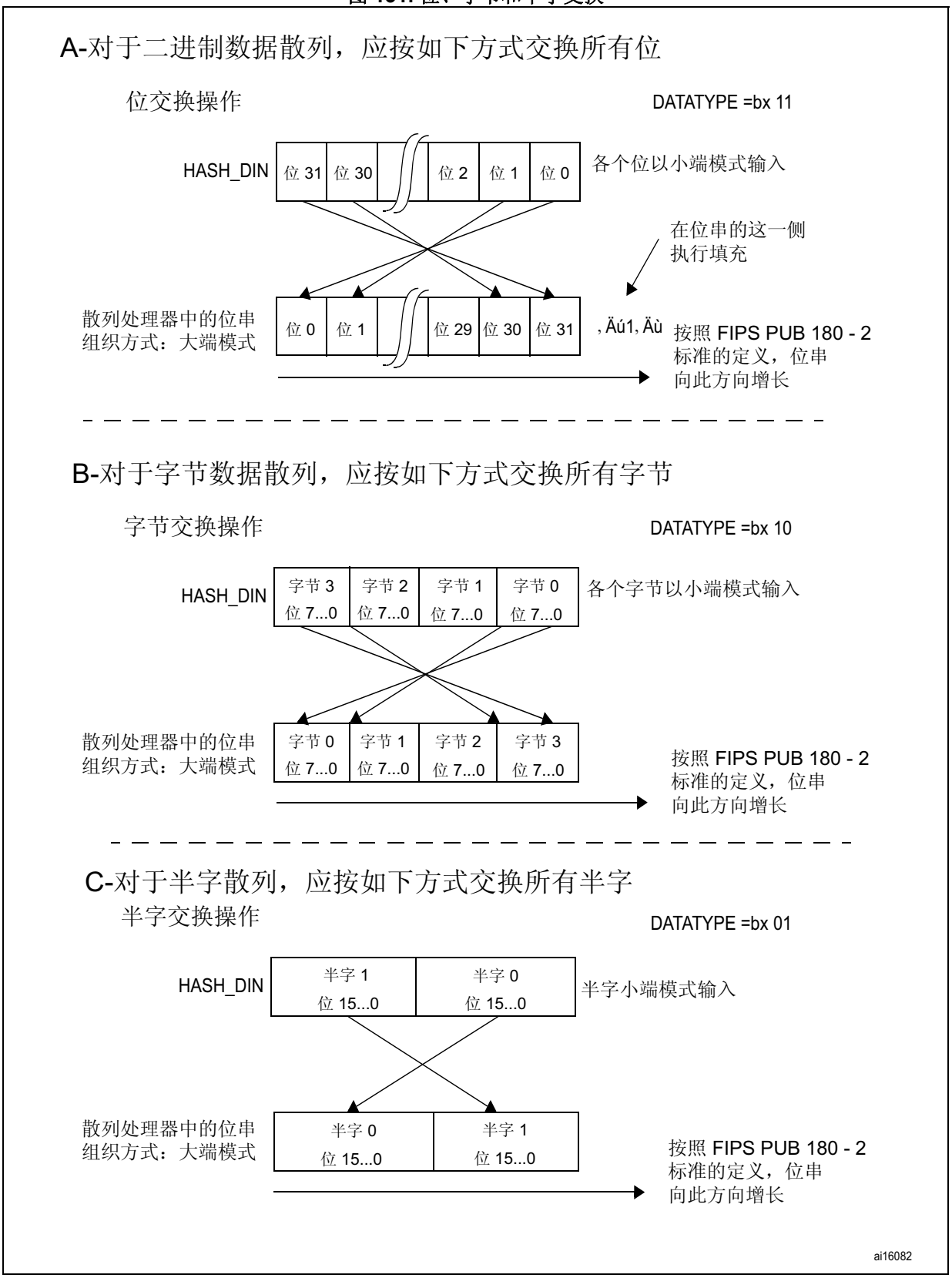
- 对于散列消息，该系数为 1 到 2.5
- 对于 HMAC 输入密钥，该系数在 2.5 左右
- 对于 HMAC 消息，该系数为 1 到 2.5
- 对于 HMAC 输出密钥，如果是短密钥，则该系数在 2.5 左右
- 对于 HMAC 输出密钥，如果是长密钥，则该系数为 3.5 到 5

21.3.2 数据类型

通过将数据写入到 HASH_DIN 寄存器，数据将一次性输入到散列处理器 32 位（字）中。但原始位串可能是按照字节、半字或字来组织，甚至可能表示为位。由于系统存储器组织方式为小端模式，而 SHA1、SHA-224 和 SHA-256 计算方式为大端模式，根据原始位串的分组方式，散列处理器将自动执行位、字节或半字交换操作。

使用散列控制寄存器 (HASH_CR) 中的 DATATYPE 位字段可配置要处理的数据类型。

图 131. 位、字节和半字交换



消息的最低有效位必须在输入到散列处理器的第一个字中的位置 0（右侧），位串的第 32 位必须在输入到散列处理器的第二个字中的位置 0，依此类推。

21.3.3 消息摘要计算

散列在计算消息摘要时，将按顺序处理 512 位的块。这样，DMA 或 CPU 每次将 16 x 32 位字（= 512 位）写入到散列处理器时，散列将自动开始计算消息摘要。此操作称为“部分摘要计算”。

要处理的消息以写入到 HASH_DIN 寄存器的 32 位字来输入到外设。每次在寄存器中写入新数据时，HASH_DIN 寄存器的当前内容将传输到输入 FIFO (IN FIFO)。HASH_DIN 和输入 FIFO 构成 17 字长度的 FIFO（名为 IN 缓冲器）。

只有在块的最后一个值已进入 IN FIFO 之后，才能开始处理块。外设必须获取有关 HASH_DIN 寄存器是否包含消息的最后一位的信息。可能发生两种情况：

- 未使用 DMA 时：
  - 如果是部分摘要计算，则方式是将额外的一个字（实际上是下一个块的第一个字）写入到 HASH_DIN 寄存器。然后，软件必须等到处理器再次就绪（当 DINIS=1 时），然后才能在 HASH_DIN 中写入新数据。
  - 如果是最终摘要计算（输入的最后一个块），则方式是将 DCAL 位写入到 1。
- 使用 DMA 时：

  将使用 DMA 控制器发送的信息来自动解释 HASH_DIN 寄存器的存储信息。
  - 如果是单个 DMA 传输：应将多个 DMA 传输 (MDMAT) 位清零。如果最后一个块已通过 DMA 通道传输到 HASH_DIN 寄存器，则 HASH_STR 寄存器中的 DCAL 位将自动置 1，以启动最终摘要计算。
  - 如果是多个 DMA 传输：应该由软件将多个 DMA 传输 (MDMAT) 位置为 1，所以硬件无法将 DCAL 位自动置 1，在这种情况下，散列的最终摘要计算以及 HMAC 的每个阶段的最终摘要计算在 DMA 传输请求结束时将不会启动，因此处理器便可以接收新的 DMA 传输（有关 HMAC 阶段的详细信息，请参见“HMAC 运算”部分）。在最后 DMA 传输期间，应由软件将多个 DMA 传输 (MDMAT) 位清零，以便在最后一个块结束时将 DCAL 位自动置 1 并启动最终摘要。
  - 将使用 DMA 控制器发送的信息来自动解释 HASH_DIN 寄存器的存储信息。

这一过程（数据输入和部分摘要计算）将继续，直至原始消息的最后一位写入到 HASH_DIN 寄存器。因为消息的长度（位数）可以是任何整数值，因此写入到散列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 突寄存器的 NBLW 位段中，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。

此操作完成后，将 HASH_STR 的 DCAL 位置 1 将使散列处理器开始处理消息的最后一个输入块。此处理包括：

- 自动执行消息填充操作：此操作的目的是使填充的消息的总长度变为 512 的倍数。散列处理器在计算消息摘要时，将按顺序处理 512 位的块
- 计算最终消息摘要

如果使能 DMA，DMA 会在传输最后一个数据字时向散列处理器提供信息。然后将自动执行填充和摘要计算，就如同 DCAL 已写入到 1 一样。

21.3.4 消息填充

消息填充的方法是：在原始消息的结尾添加一个“1”，后跟 m 个“0”以及一个 64 位整数，目的是生成一个长度为 512 的填充消息块。“1”将添加到 HASH_DIN 寄存器中写入的最后一个字，位位置由 NBLW 位字段定义，而其余更高的位将被清零（“0”）。

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”，现在为 448。这将生成（十六进制，大端模式）：

```
61626380 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000
```

将追加双字表示法格式的 L 值，即 00000000 00000018。因此，十六进制格式的最终填充消息为：

```
61626380 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000028
```

如果散列编程为使用小端模式字节输入格式，则以上消息必须通过以下步骤来输入：

1. 0xUU636261 写入到 HASH_DIN 寄存器（其中的“U”意味着无关）。
2. 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. 0x10 写入到 HASH_STR 寄存器以启动消息填充和摘要计算。当 NBLW ≠ 0x00 时，消息填充会将“1”放入到 HASH_DIN 寄存器中（位的位置由 NBLW 值定义），并且在位位置 [31:(NBLW+1)] 处插入“0”。当 NBLW == 0x00 时，消息填充使用值 0x0000 0001 插入一个新字。然后将添加一个全零字 (0x0000 0000) 以及双字表示法格式的消息长度，从而形成 16 x 32 位字的块。
4. 将执行散列计算，并且随后在 HASH_Hx 寄存器 (x = 0...4) 中消息摘要可供 SHA-1 算法使用。例如：

```
H0 = 0xA9993E36
H1 = 0x4706816A
H2 = 0xBA3E2571
H3 = 0x7850C26C
H4 = 0x9CD0D89D
```

21.3.5 散列运算

当 HASH_CR 的 MODE 位为“0”时，将 HASH_CR 寄存器的 INIT 置“1”将选择散列函数（SHA-1、SHA-224、SHA-256 和 MD5）。同时（即 INIT 位置 1 时），使用 ALGO 位来选择算法（SHA-1、SHA-224、SHA-256 或 MD5）。

随后通过逐字将消息写入到 HASH_DIN 寄存器来发送消息。写入了 512 位的块（即 16 个字）后，部分摘要计算将在写入下一个块的第一个数据之后启动。对于 SHA-1 算法，散列处理器在 66 个周期内保持忙碌，对于 MD5 算法、SHA-224 算法和 SHA-256 算法，为 50 个周期。

随后可以重复此过程，直至消息的最后一个字。如果使用了 DMA 传输，请参见 *DMA 加载数据的过程* 一节。否则，如果消息长度不是 512 位的整倍数，则必须写入 HASH_STR 寄存器以启动最终摘要的计算。

计算完成之后，可以从 HASH_H0 到 HASH_H7 寄存器中读取摘要（对于 MD5 算法 HASH_H4 无关），其中：

选择了 MD5 算法时，HASH_H4 到 HASH_H7 无关。

选择了 SHA-1 算法时， HASH_H5 到 HASH_H7 无关。

选择了 SHA-224 算法时，HASH_H7 无关。

21.3.6 HMAC 运算

HMAC 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。有关 HMAC 规范，请参见 H. Krawczyk、M. Bellare 和 R. Canetti 于 1997 年 2 月发表的文章“HMAC：密钥散列消息认证，H. Krawczyk，M. Bellare, R. Canetti，1997 年 2 月”。

基本而言，该算法由两个嵌套的散列运算组成：

```
HMAC(message) = Hash[((key | pad) XOR 0x5C)
          | Hash(((key | pad) XOR 0x36) | message)]
```

其中：

- pad 是将密钥扩展到底层散列函数数据块的长度所需要的零序列（对于 SHA-1、SHA224、SHA-256 和 MD5 散列算法是 512 位）
- | 表示串联运算符

要计算 HMAC，需要四个不同阶段：

1. 当 MODE 位为 1 且 ALGO 位段已根据所需算法设好对应的值，通过将 INIT 位置 1 对块进行初始化。如果使用的密钥长度超过 64 位，则在此阶段中还必须将 LKEY 位置 1（在此情况下，HMAC 规范规定密钥的散列应代替真实密钥）。
2. 随后将用于内部散列函数的密钥提供给内核。此操作采用的机制即是用于在散列运算中发送消息的机制（也就是通过写入到 HASH_DIN 并最终写入到 HASH_STR）。
3. 输入了最后一个字并且计算启动后，散列处理器便开始生成密钥。然后便可以使用与在散列运算中发送消息相同的机制接受消息文本。
4. 在第一轮散列之后，散列处理器将返回“就绪”以表明其可以接收用于外部散列函数的密钥（通常，此密钥与用于内部散列函数的密钥相同）。在输入了密钥的最后一个字并且计算开始后，HMAC 结果存储于 HASH_H0...HASH_H7 寄存器。

注： 1 HMAC 本原的计算延迟取决于密钥和消息的长度。可以将 HMAC 视为密钥长度相同（长或短）的两个嵌套底层散列函数。

21.3.7 上下文交换

可以中断散列/HMAC 进程以执行优先级更高的其它处理，并可在稍后当优先级更高的任务完成后再执行被中断的进程。要这样做，必须将被中断的任务的上下文从散列寄存器保存到存储器，然后从存储器恢复到散列寄存器。

以下说明由软件或 DMA 控制数据流的过程。

软件加载数据的过程

仅当目前未在处理任何块时，才能保存上下文。即，必须等待 DINIS = 1（最后的块已处理并且输入 FIFO 为空）或 NBW ≠ 0（FIFO 未满，并且未在进行任何处理）。

- 保存上下文：

  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_CSR53。
- 恢复上下文：

  在高优先级任务完成后，可以恢复上下文。请遵循以下顺序。

  a) 将存储器中保存的值写入以下寄存器中：HASH_IMR、HASH_STR 和 HASH_CR

  b) 通过将 HASH_CR 寄存器中的 INIT 位置 1 来初始化散列处理器

  c) 将存储器中保存的值写入 HASH_CSR0 到 HASH_CSR53 寄存器中

  现在便可以从之前的中断点重新开始处理。

DMA 加载数据的过程

在此情况下，无法预测 DMA 传输是否在进行，也无法预测处理是否在进行。因此，必须停止 DMA 传输，然后等到散列就绪才能中断消息处理。

- 中断一个处理：
  - 将 DMAE 位清零以禁止 DMA 接口
  - 等待直至当前 DMA 传输完成（等待 HASH_SR 寄存器中的 DMAES = 0）。请注意，块不一定已完全传输到散列。
  - 在 DMA 控制器中禁止对应的通道
  - 等到散列处理器就绪（未在处理任何块），也就是等待 DINIS = 1
- 上下文保存阶段和上下文恢复阶段与以上相同（请参见*软件加载数据的过程*）。

重新配置 DMA 控制器以使其传输消息的结尾。现在，通过将 DMAE 位置 1 便可从之前的中断点重新开始处理。

注： *如果上下文交换不涉及 HMAC 操作，则不必保存和恢复 HASH_CSR38 到 HASH_CSR53 寄存器。*

如果在两个块之间进行上下文交换（最后一个块已完全处理并且下一个块尚未推入到 IN FIFO，HASH_CR 寄存器中 NBW = 000），则不必保存和恢复 HASH_CSR22 到 HASH_CSR37 寄存器。

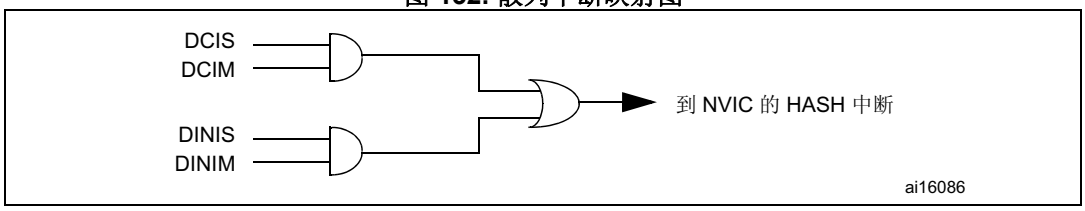
21.3.8 散列中断

散列处理器可生成两个独立的可屏蔽中断源。这两个源连接到同一个中断向量。

通过更改 HASH_IMR 寄存器中的屏蔽位，可单独使能或禁止各个中断源。将相应的屏蔽位置 1 以使能中断。

可以从 HASH_SR 寄存器中读取各个中断源的状态。

图 132. 散列中断映射图



21.4 散列寄存器

散列内核与多个控制和状态寄存器以及五个消息摘要寄存器相关联。

所有这些寄存器都只能通过字来访问，否则将生成 AHB 错误。

21.4.1 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]	Res.	LKEY
													rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	MDMAT	DINNE	NBW				ALGO[0]	MODE	DATATYPE		DMAE	INIT	Res.	Res.
		rw	r	r	r	r	r	rw	rw	rw	rw	rw	w		

位 31:19 保留，由硬件强制清零。

位 17 保留，由硬件强制清零。

位 16 **LKEY**：长密钥选择 (Long key selection)

在 HMAC 模式中，该位在短密钥（≤ 64 字节）或长密钥（> 64 字节）之间进行选择

0：短密钥（≤ 64 字节）

1：长密钥（> 64 字节）

注： 仅当 INIT 位置 1 并且 MODE = 1 时此选择才有效。在计算期间更改该位不起作用。

位 15:14 保留，由硬件强制清零。

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散列运算，则将此位置 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)

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）写入到 1 时，会将此位清零。

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 11:8 **NBW**：已推入的字数 (Number of words already pushed)

该位字段反映了消息中已推入到 IN FIFO 的字数。

如果对 HASH_DIN 寄存器执行了写访问且 DINNE = 1，则 NBW 递增 (+1)。

当 INIT 位写入到 1 或者当摘要计算启动时（DCAL 写入到 1 或者 DMA 传输结束），NBW 将变为 0000。

″ 如果未使用 DMA：

0000 且 DINNE=0：没有任何字推入到 DIN 缓冲器（缓冲器为空，HASH_DIN 寄存器和 IN FIFO 均为空）

0000 且 DINNE=1：1 个字已推入到 DIN 缓冲器（HASH_DIN 寄存器包含 1 个字，IN FIFO 为空）

0001：2 个字已推入到 DIN 缓冲器（HASH_DIN 寄存器和 IN FIFO 分别包含 1 个字）

...

1111：16 个字已推入到 DIN 缓冲器

″ 如果使用了 DMA，则 NBW 刚好是已推入到 IN FIFO 的字数。

位 18 和位 7 **ALGO[1:0]**：算法选择 (Algorithm selection)

这些位用于选择 SHA-1、SHA-224、SHA256 或 MD5 算法：

00：选择 SHA-1 算法

01：选择 MD5 算法

10：选择 SHA224 算法

11：选择 SHA256 算法

注： 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 6 **MODE**：模式选择 (Mode selection)

该位针对所选算法选择散列或 HMAC 模式：

0：选择散列模式

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 LKEY 置 1。

注： 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**：数据类型选择 (Data type selection)

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 的数据将由散列处理直接使用，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 2 个半字，并且在交换之后由散列处理使用。

10：8 位数据或字节。写入到 HASH_DIN 的数据被视为 4 个字节，并且在交换之后由散列处理使用。

11：位数据或位串。写入到 HASH_DIN 的数据被视为 32 位（位串的第一位处于位置 0），并且在交换之后由散列处理使用（位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 散列内核可以接收数据时，便发送 DMA 请求。

注： 1：当 DMA 发出 DMA 端子计数信号时（传输消息的最后一个数据时），将由硬件将该位清零。当 INIT 位写入到 1 时，不会将该位清零。

2：如果该位写入到 0，而已经向 DMA 请求了 DMA 传输，则会将 DMAE 清零但是不会中止当前传输。DMA 接口继续在内部使能，直到传输完成或 INIT 写入到 1。

位 2 **INIT**：初始化消息摘要计算 (Initialize message digest calculation)

将该位写入到 1 将会复位散列处理器内核，以使散列可以计算新消息的消息摘要。

若在此位中写入 0 则不会产生影响。

读取该位将始终返回 0。

位 1:0 保留，必须保持清零。

21.4.2 散列数据输入寄存器 (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。它用于以 512 位的块来输入消息。写入 HASH_DIN 寄存器时，在 AHB 数据总线上呈现的值将被“推入”散列内核，并且寄存器获取 AHB 数据总线上呈现的新值。必须之前已在 HASH_CR 寄存器中配置了 DATATYPE 位才能获取正确的消息表示法。

16 字的块写入到 HASH_DIN 寄存器后，将启动中间摘要计算。有两种启动方式：

- 如果未使用 DMA，将新数据写入到 HASH_DIN 寄存器来启动，即写入下一个块的第一个字（中间摘要计算）
- 如果使用了 DMA，则自动启动

最后一个块写入到 HASH_DIN 寄存器之后，将启动最终摘要计算（包括填充）。有两种启动方式：

- 通过在 HASH_STR 寄存器中将 DCAL 位写入 1 来启动（最终摘要计算）
- 如果使用了 DMA 并且 MDMAT 位设置为“0”，则自动启动。

当摘要计算（中间或最终）正在进行时，将扩展对 HASH_DIN 寄存器的所有新的写访问（通过在 AHB 总线上插入等待状态），直至散列计算完成。

读取 HASH_DIN 寄存器时，将访问该位置中写入的最后一个字（复位后为零）。

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

位 31:0 **DATAIN**：数据输入 (Data input)

读取 = 返回寄存器的当前内容。

写入 = 当前寄存器内容推入到 IN FIFO，并且寄存器获取在 AHB 数据总线上呈现的新值。

21.4.3 散列启动寄存器 (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：

- 用于定义散列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）
- 通过将 DCAL 位写入到 1，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				
							w				rw	rw	rw	rw	rw

位 31:9 保留，由硬件强制清零。

位 8 **DCAL**：摘要计算 (Digest calculation)

将该位写 1 将开始使用先前写入的 NBLW 值来填充消息，并且，由于 INIT 位已被写 1，因此将使用写入到 IN 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，由硬件强制清零。

位 4:0 **NBLW**：在散列处理器的位串结构中，消息的最后一个字中的有效位数

如果写入这些位并且 DCAL 为“0”，则获取 AHB 数据总线上的值：

0x00：对于在散列处理器的位串结构中写入的最后一个数据，所有 32 位（在数据交换后）均有效。

0x01：对于在散列处理器的位串结构中写入的最后一个数据，仅位 [31]（在数据交换后）有效

0x02：对于在散列处理器的位串结构中写入的最后一个数据，仅位 [31:30]（在数据交换后）有效

0x03：对于在散列处理器的位串结构中写入的最后一个数据，仅位 [31:29]（在数据交换后）有效

...

0x1F：对于在散列处理器的位串结构中写入的最后一个数据，仅位 [0]（在数据交换后）有效

如果当 DCAL 为“1”时写入这些位，则位字段不会更改。

读取这些位将返回写入到 NBLW 的最后一个值。

注： 必须在设置 DCAL 位之前配置这些位，否则这些位不会有效。特别请注意，配置 NBLW 和设置 DCAL 不能同时进行。

21.4.4 散列摘要寄存器（HASH_HR0 到 4/5/6/7）

HASH digest registers

偏移地址：0x0C 到 0x32C

复位值：0x0000 0000

这些寄存器包含消息摘要结果，其命名方式为：

1. 在 SHA1 算法说明中分别为 H0、H1、H2、H3 和 H4。
   请注意，这里不使用 HASH_H5 到 HASH_H7 并且读取值为零。
2. 在 MD5 算法说明中分别为 A、B、C 和 D。
   请注意，这里不使用 HASH_H4 到 HASH_H7 寄存器并且读取值为零。
3. 在 SHA224 算法说明中分别为 H0 到 H6。
   请注意，这里不使用 HASH_H7 并且读取值为零。
4. 在 SHA256 算法说明中分别为 H0 到 H7。

如果当散列内核在计算中间摘要或最终消息摘要时（也就是在 DCAL 位已经写入到 1 时），对这些寄存器之一进行读访问，则读取将被拖延直到散列计算完成。

注： *H0、H1、H2、H3 和 H4 映射到两个区域。*

HASH_HR0

偏移地址：0x310

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
H0															
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
H0															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

HASH_HR1

偏移地址：0x314

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
H1															
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
H1															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

HASH_HR2

偏移地址：0x318

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
H2															
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
H2															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

HASH_HR3

偏移地址：0x31C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
H3															
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
H3															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

HASH_HR4

偏移地址：0x320

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
H4															
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
H4															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

HASH_HR5

偏移地址：0x324

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
H5															
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
H5															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

HASH_HR6

偏移地址：0x328

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
H6															
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
H6															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

HASH_HR7

偏移地址：0x32C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
H7															
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
H7															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

注：　当启动新位流的摘要计算时（通过将 INIT 位写入到 1），这些寄存器将呈现其复位值。

21.4.5 散列中断使能寄存器 (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，由硬件强制清零。

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

21.4.6 散列状态寄存器 (HASH_SR)

HASH status register

偏移地址：0x24

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.	BUSY	DMAS	DCIS	DINIS
												r	r	rc_w0	rc_w0

位 31:4 保留，由硬件强制清零。

位 3 **BUSY**：忙碌位 (Busy bit)

0：当前未在处理任何块

1：散列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 一起设置，并且在 DMAE=0 且未在进行任何 DMA 传输时将被清零。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，或者向 HASH_CR 寄存器中的 INIT 位写入 1。

0：HASH_Hx 寄存器中无任何摘要

1：摘要计算已完成，摘要存储于 HASH_Hx 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)

该位是在输入缓冲器准备好获取新块时（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。

21.4.7 散列上下文交换寄存器 (HASH_CSRx)

HASH context swap registers

偏移地址：0x0F8 到 0x1CC

- 对于 HASH_CSR0 寄存器：复位值为 0x0000 0002。
- 对于其它寄存器：复位值是 0x0000 0000，但是其中的 HASH_CSR2 寄存器复位值为 0x2000 0000 的器件除外

这些寄存器包含散列处理器的完整内部寄存器状态。当高优先级任务必须使用散列处理器，而散列处理器已由其它任务使用时，必须执行上下文交换，这时这些寄存器很有用。

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某个位置。然后有优先权的任务便可以使用散列处理器，在散列计算完成时，可以从存储器中读取保存的上下文并回写到这些 HASH_CSRx 寄存器中。

HASH_CSRx

偏移地址：0x0F8 到 0x1CC

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

21.4.8 散列寄存器映射

表 122 汇总了散列寄存器映射和复位值。

表 122. HASH 寄存器映射和复位值

偏移	寄存器名称 复位值	寄存器大小																															
		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.	Res.	ALGO[1]	Reserved	LKEY	Res.	Res.	MDMAT	DINNE	NBW				ALGO[0]	MODE	DATATYPE	DMAE	INIT	Res.	Res.
	Reset value														0		0			0	0	0	0	0	0	0	0	0	0	0		
0x04	HASH_DIN	DATAIN																															
	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				
	Reset value																								0				0	0	0	0	0
0x0C	HASH_HR0	H0																															
	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																															
	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																															
	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 122. HASH 寄存器映射和复位值（续）

偏移	寄存器名称 复位值	寄存器大小																															
		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	HASH_HR3	H3																															
	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																															
	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	0	0	0	0	0	0	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	HASH_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.	BUSY	DMAS	DCIS	DINIS
	Reset value																													0	0	0	1
0xF8	HASH_CSR0	CSR0																															
	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	0
...																																	
...																																	
0x1CC	HASH_CSR53	CSR53																															
	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																															
	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																															
	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																															
	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																															
	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																															
	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																															
	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																															
	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																															
	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

22 高级控制定时器 (TIM1/TIM8)

22.1 TIM1/TIM8 简介

高级控制定时器 (TIM1//TIM8) 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。

此类定时器可用于各种用途，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。

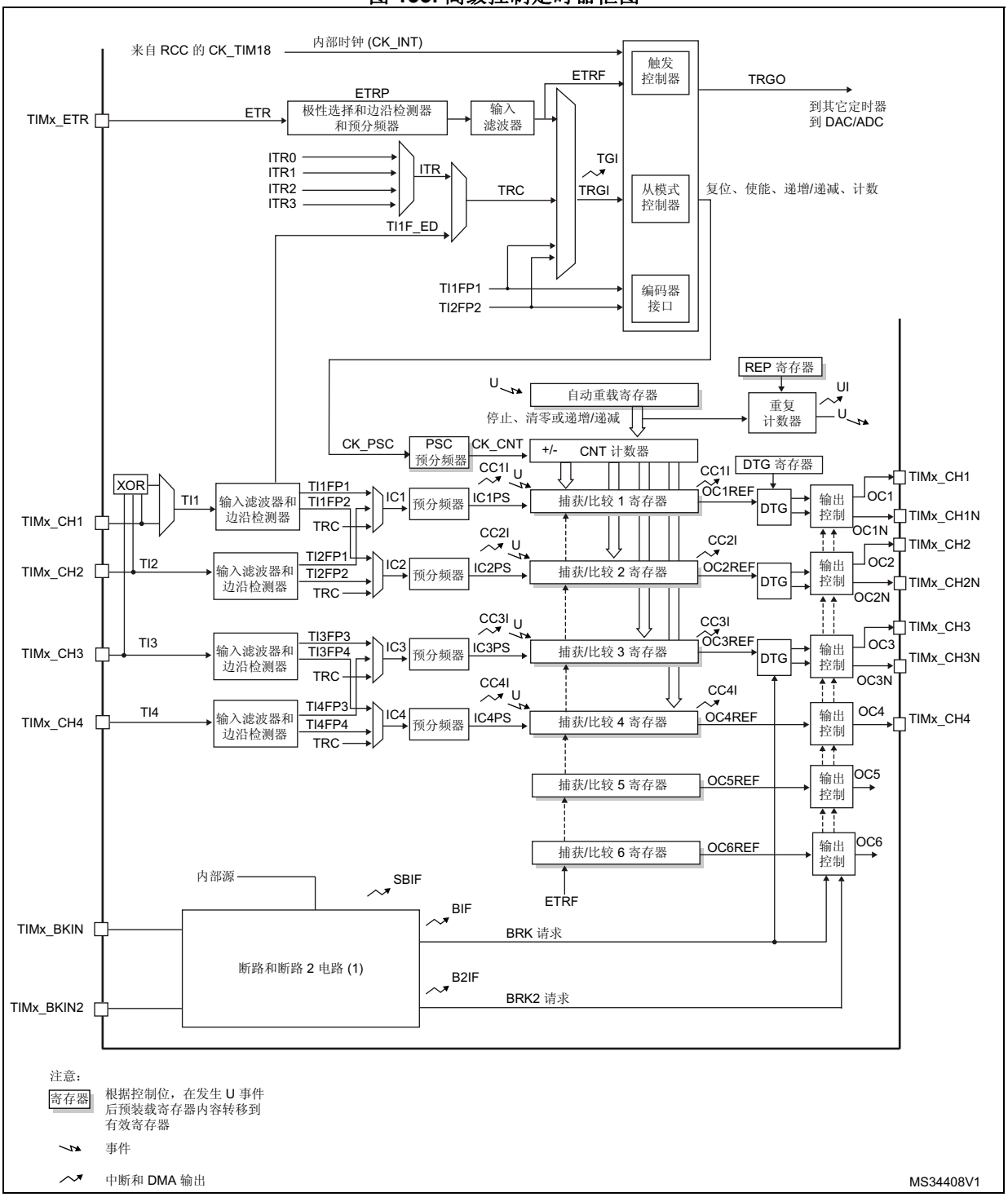
高级控制定时器（TIM1/TIM8）和通用 (TIMx) 定时器彼此完全独立，不共享任何资源。如*第 22.3.26 节*中所述，它们可以同步操作。

22.2 TIM1/TIM8 主要特性

TIM1/TIM8 定时器具有以下特性：

- 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。
- 使用外部信号控制定时器且可实现多个定时器互连的同步电路。
- 重复计数器，用于仅在给定数目的计数器周期后更新定时器寄存器。
- 2 个断路输入，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。
- 触发输入用作外部时钟或逐周期电流管理。

图 133. 高级控制定时器框图



1. 详细信息，请参见*图 175：断路和断路 2 电路概述*。

22.3 TIM1/TIM8 功能说明

22.3.1 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、递减计数或交替进行递增和递减计数。计数器的时钟可通过预分频器进行分频。

计数器、自动重载寄存器和预分频器寄存器可通过软件进行读写。即使在计数器运行时也可执行读写操作。

时基单元包括：

- 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直接传送到影子寄存器，也可以在每次发生更新事件 (UEV) 时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位 (ARPE)。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。下文将针对各配置的更新事件的产生进行详细介绍。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位 (CEN) 置 1 时，才会启动计数器（有关计数器使能的更多详细信息，另请参见从模式控制器的相关说明）。

注意，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频率进行分频，分频系数介于 1 和 65536 之间。该预分频器基于 TIMx_PSC 寄存器中的 16 位寄存器所控制的 16 位计数器。由于该控制寄存器具有缓冲功能，因此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 134 和 *图 135* 以一些示例说明在预分频比实时变化时计数器的行为。

图 134. 预分频器分频由 1 变为 2 时的计数器时序图

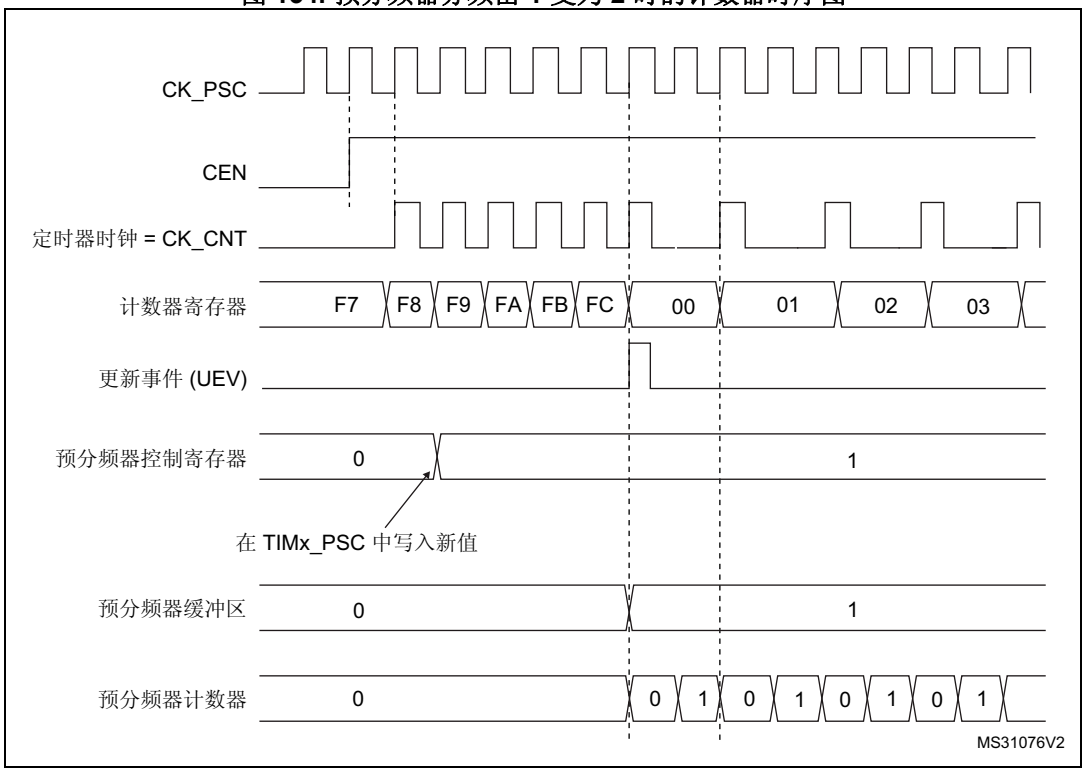
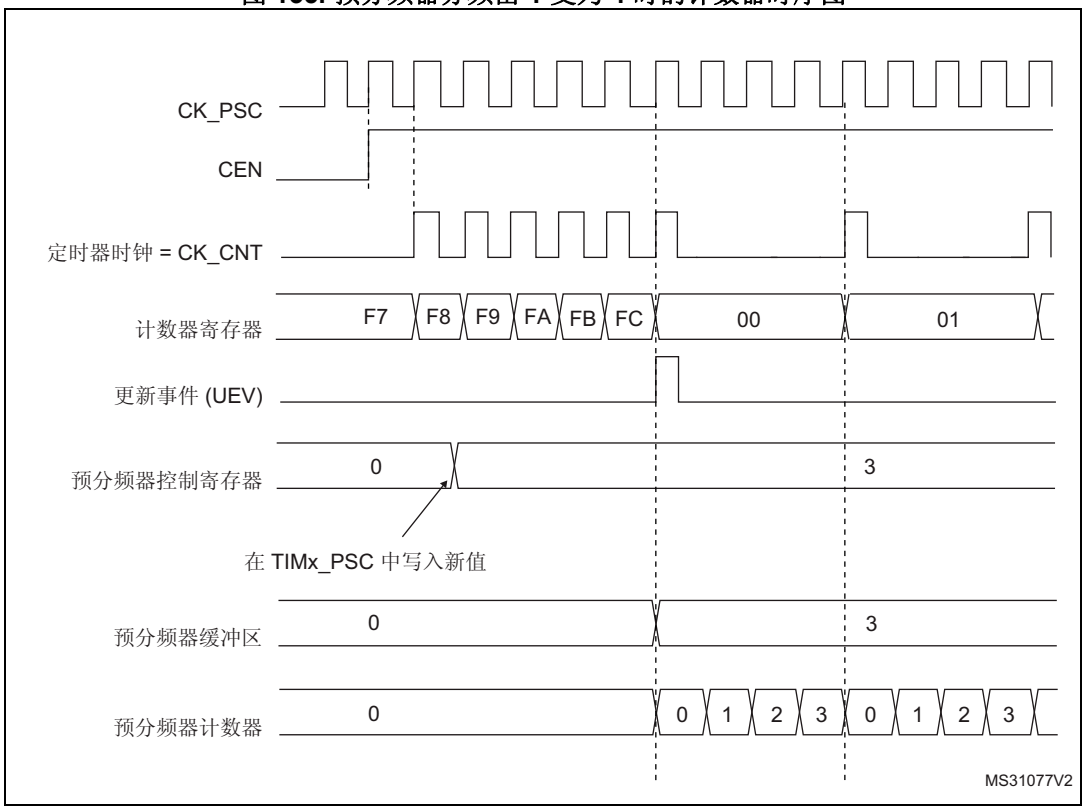


图 135. 预分频器分频由 1 变为 4 时的计数器时序图



22.3.2 计数器模式

递增计数模式

在递增计数模式下，计数器从 0 计数到自动重载值（TIMx_ARR 寄存器的内容），然后重新从 0 开始计数并生成计数器上溢事件。

如果使用重复计数器，则当递增计数的重复次数达到重复计数器寄存器中编程的次数加一次（(TIMx_RCR) + 1）后，将生成更新事件 (UEV)。否则，将在每次计数器上溢时产生更新事件。

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）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容，
- 自动重载影子寄存器将以预装载值 (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 136. 计数器时序图，1 分频内部时钟

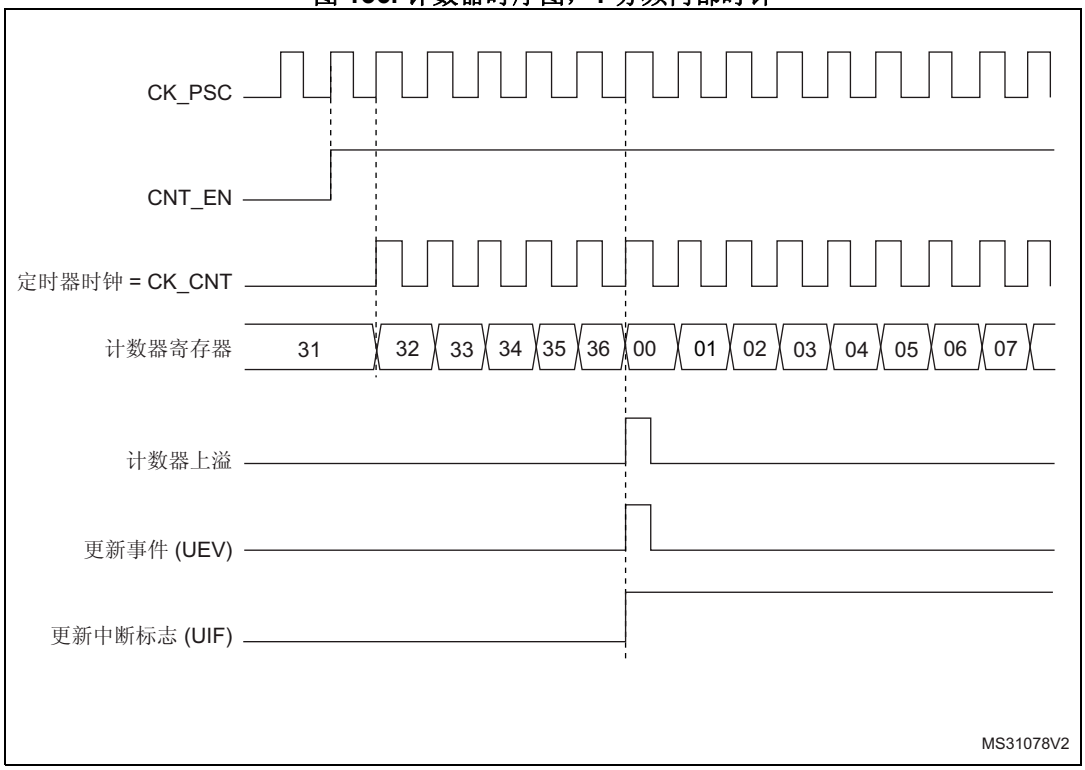


图 137. 计数器时序图，2 分频内部时钟

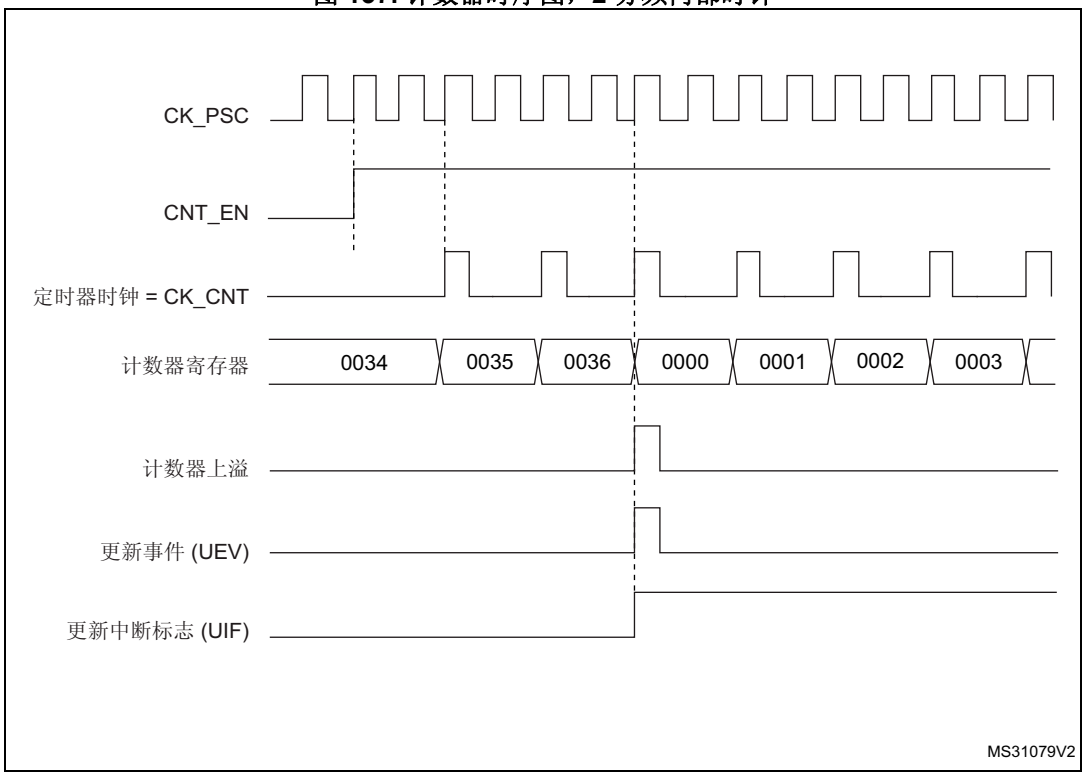


图 138. 计数器时序图，4 分频内部时钟

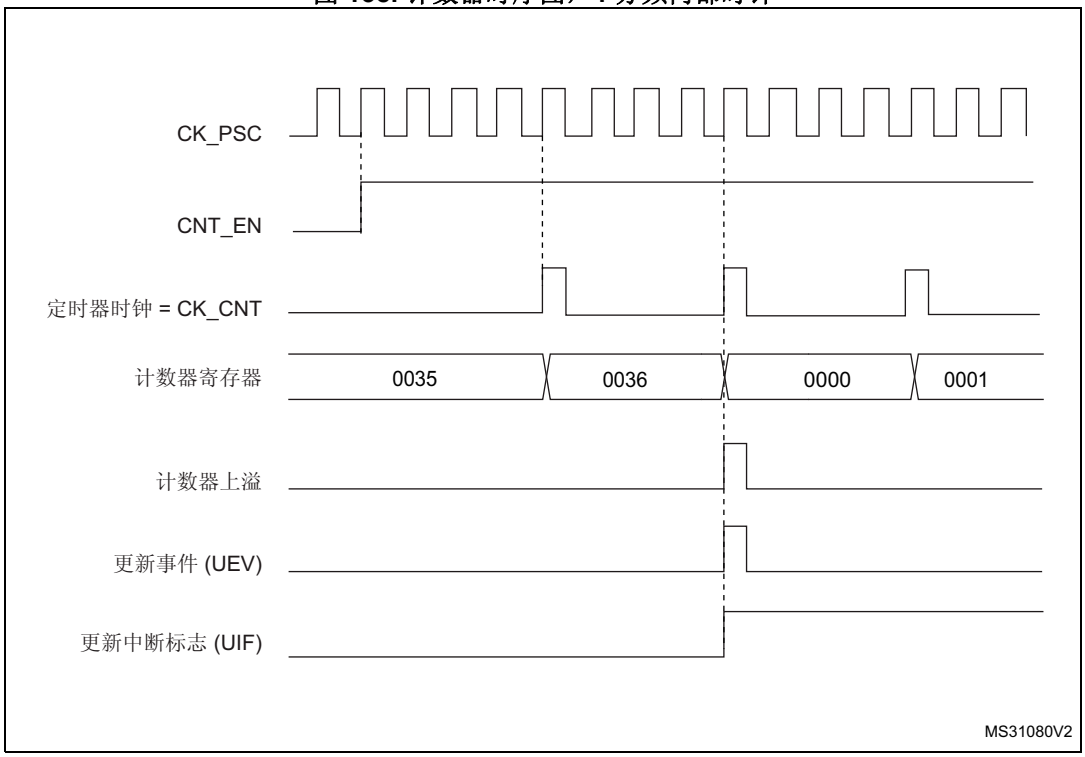


图 139. 计数器时序图，N 分频内部时钟

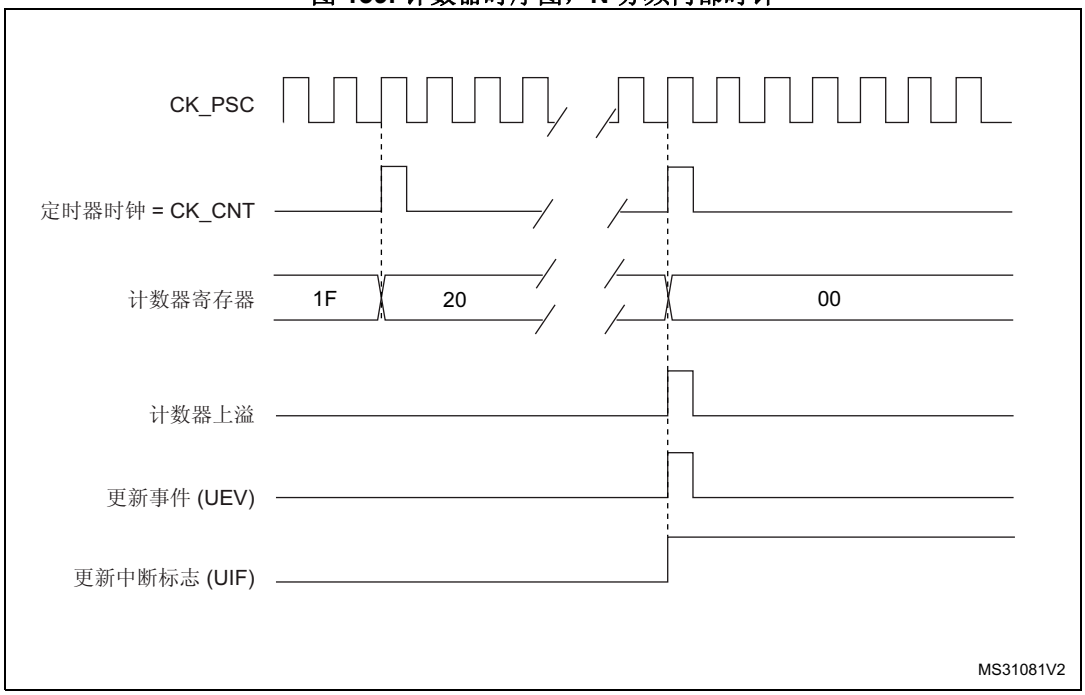


图 140. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载）

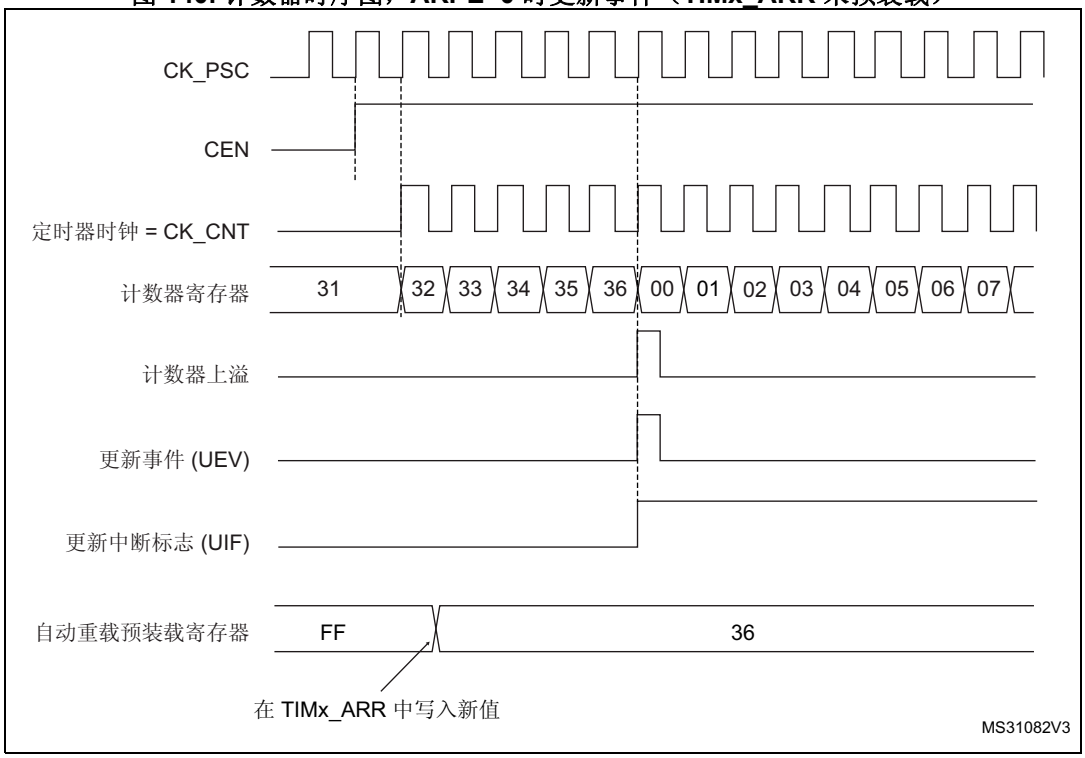
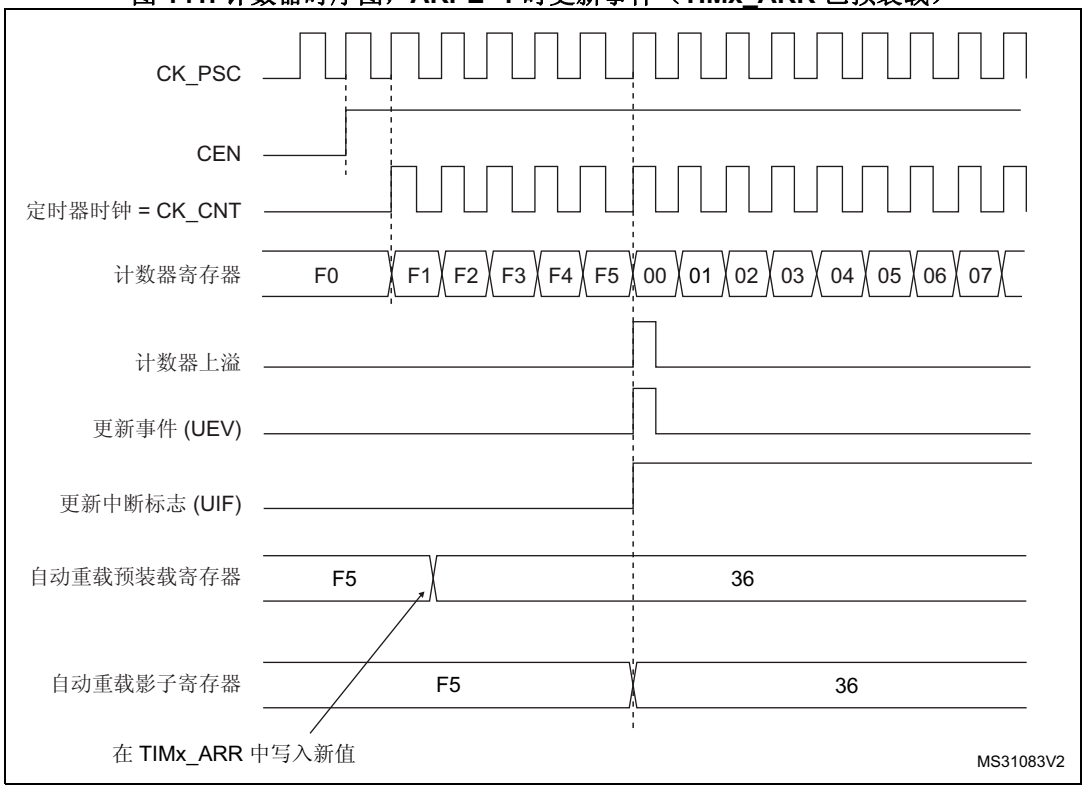


图 141. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 已预装载）



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。

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，计数器会重新从当前自动重载值开始计数，而预分频器计数器则重新从 0 开始计数（但预分频比保持不变）。

此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容。
- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，ARR 寄存器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 142. 计数器时序图，1 分频内部时钟

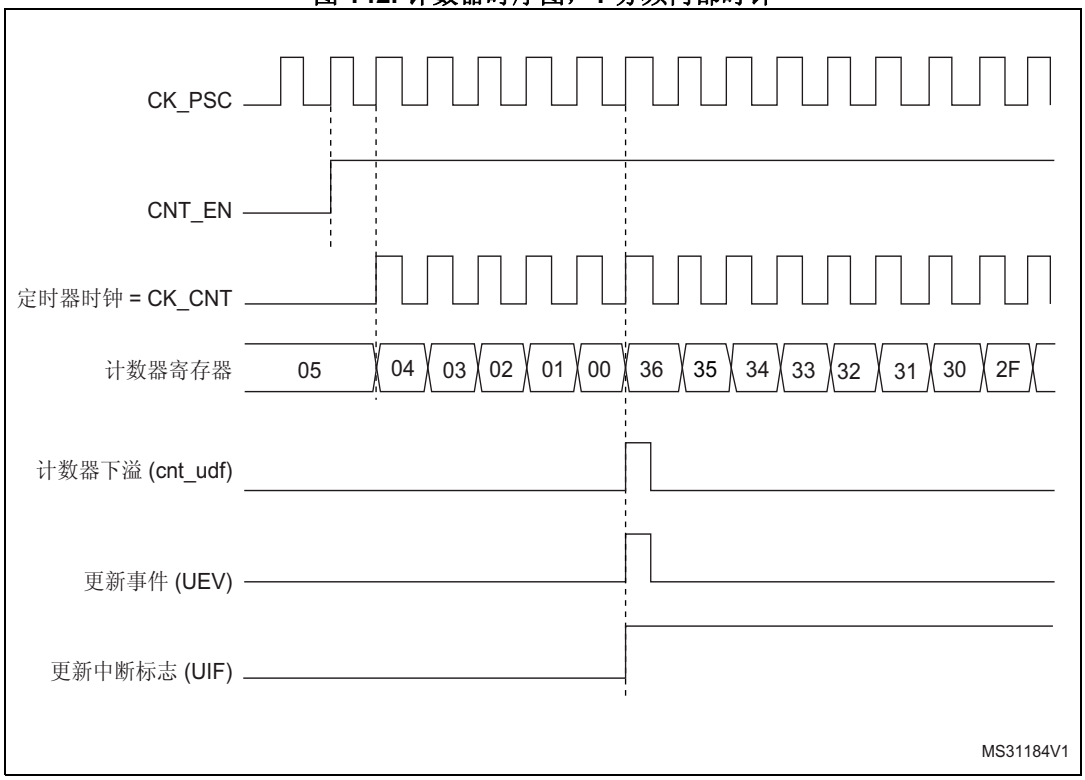


图 143. 计数器时序图，2 分频内部时钟

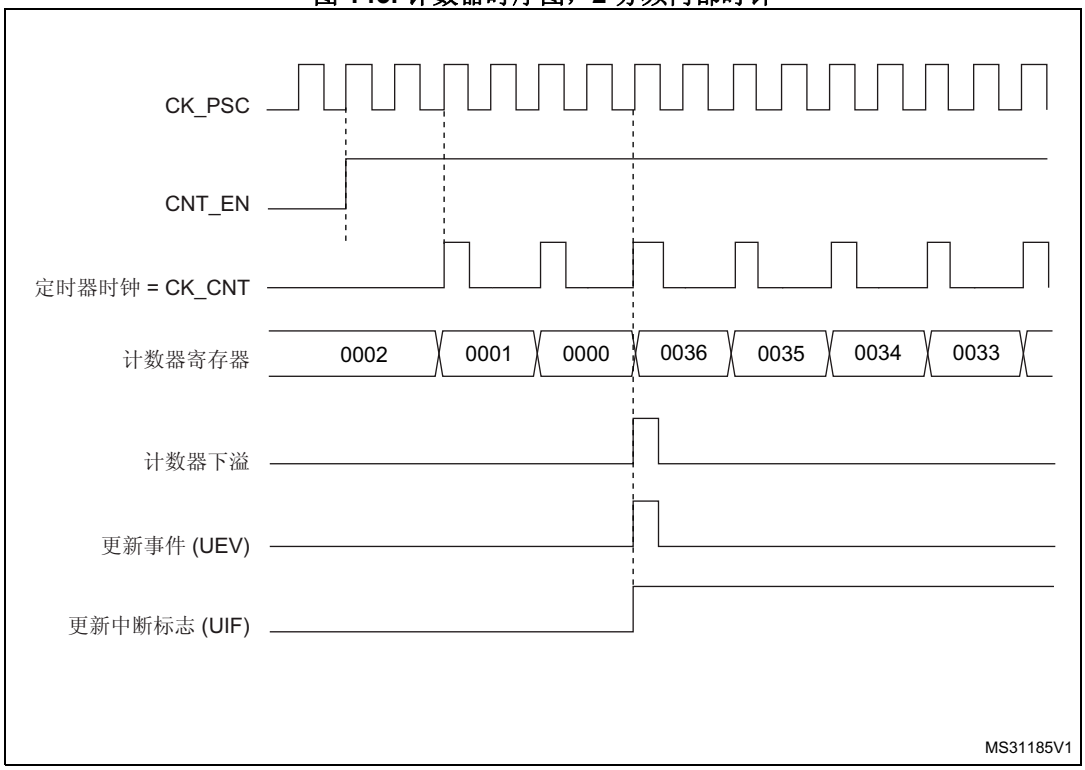


图 144. 计数器时序图，4 分频内部时钟

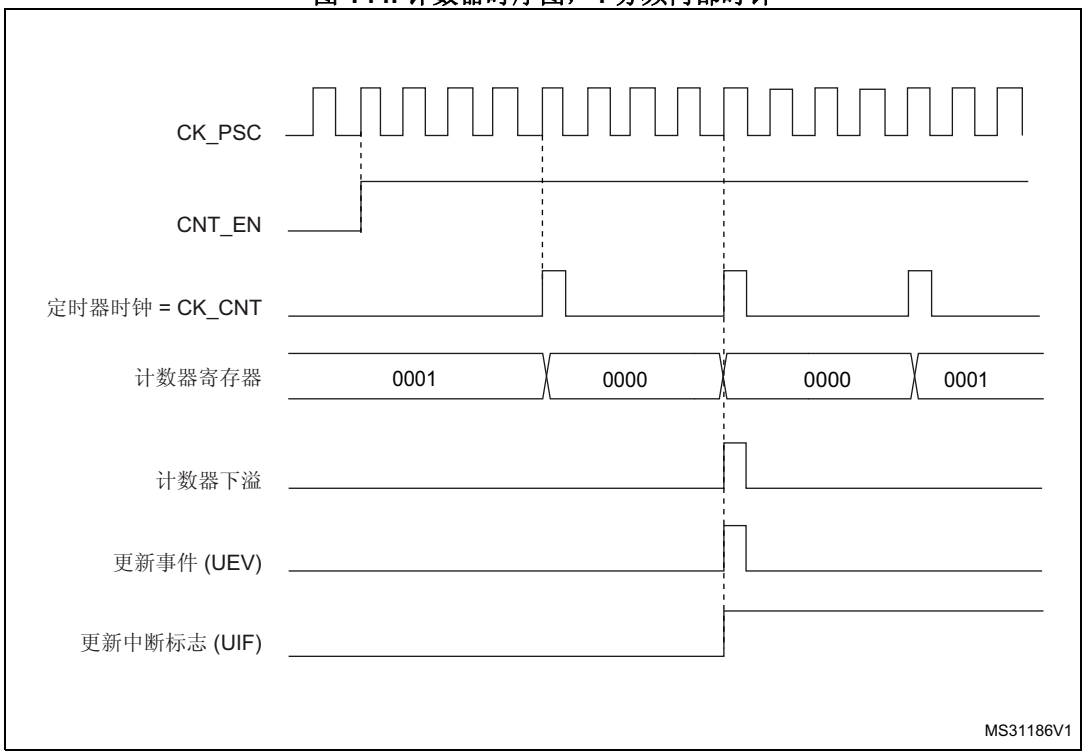


图 145. 计数器时序图，N 分频内部时钟

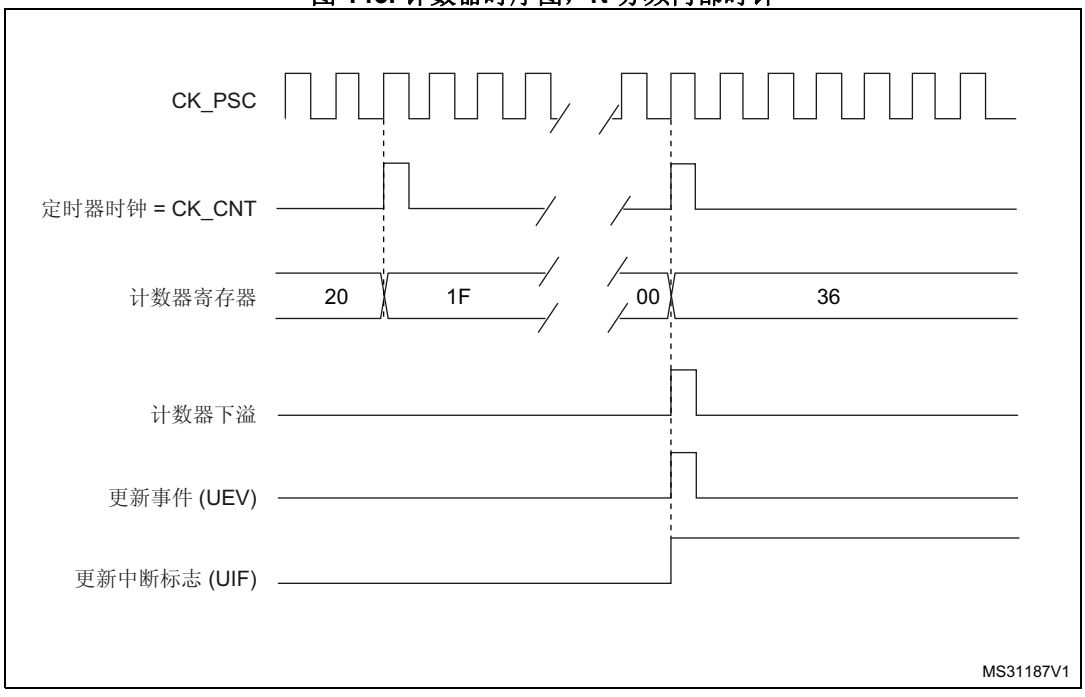
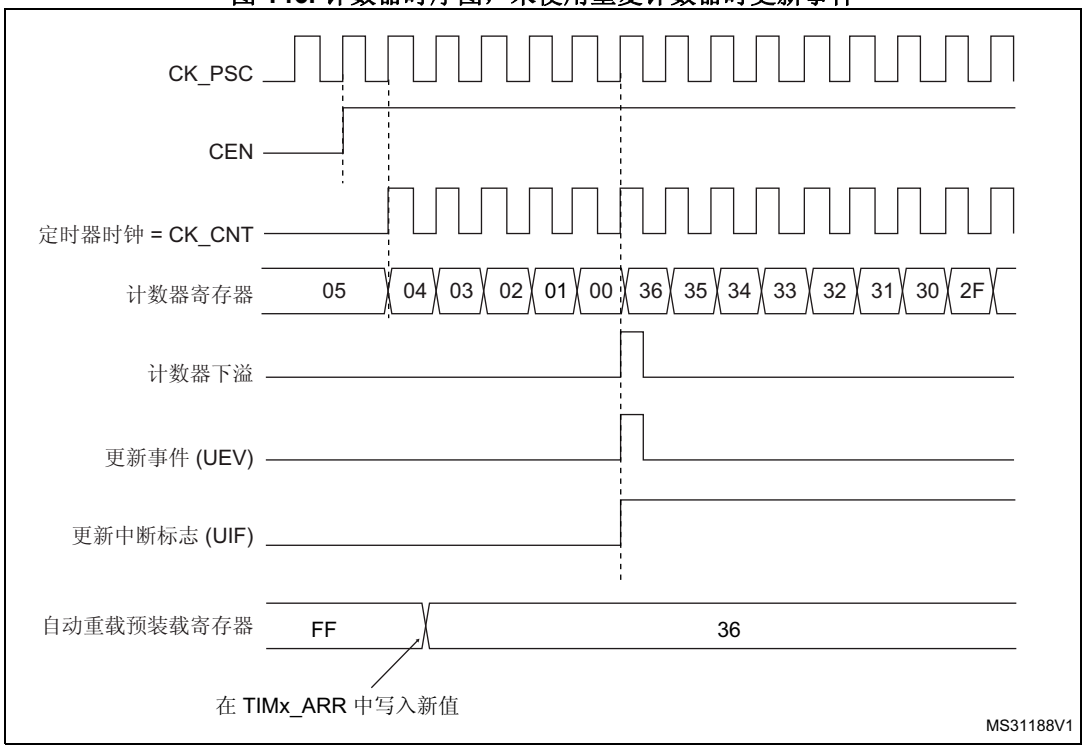


图 146. 计数器时序图，未使用重复计数器时更新事件



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。

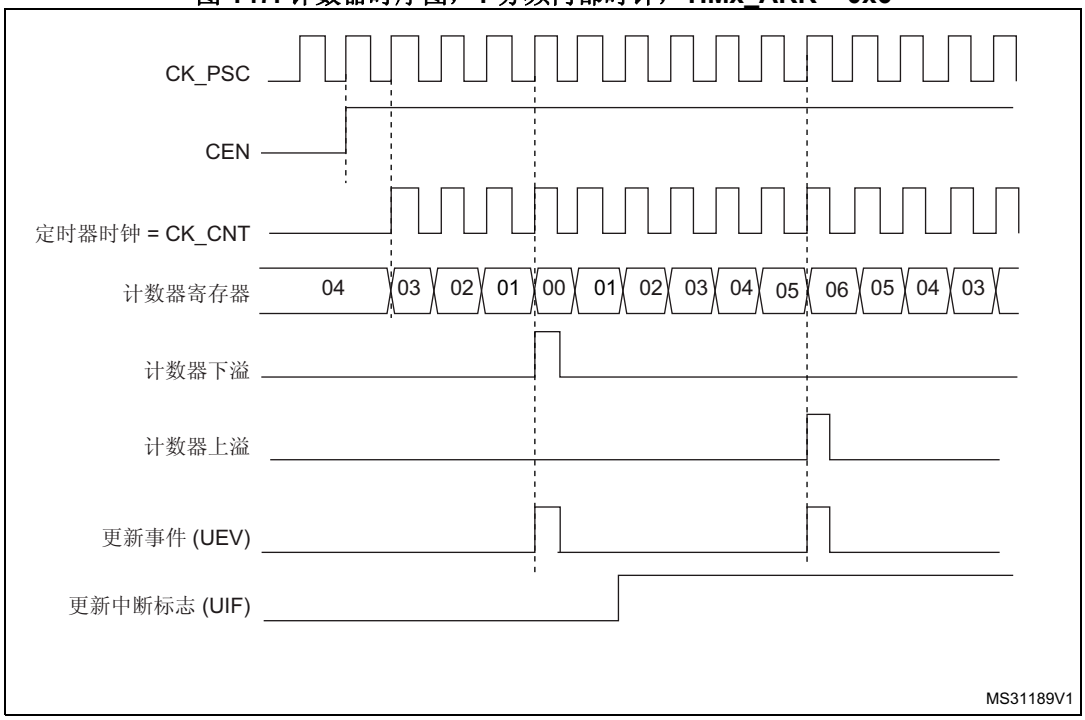
此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会产生 UEV 更新事件，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容
- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）
- 自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，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 147. 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 22.4 节：TIM1/TIM8 寄存器*）。

图 148. 计数器时序图，2 分频内部时钟

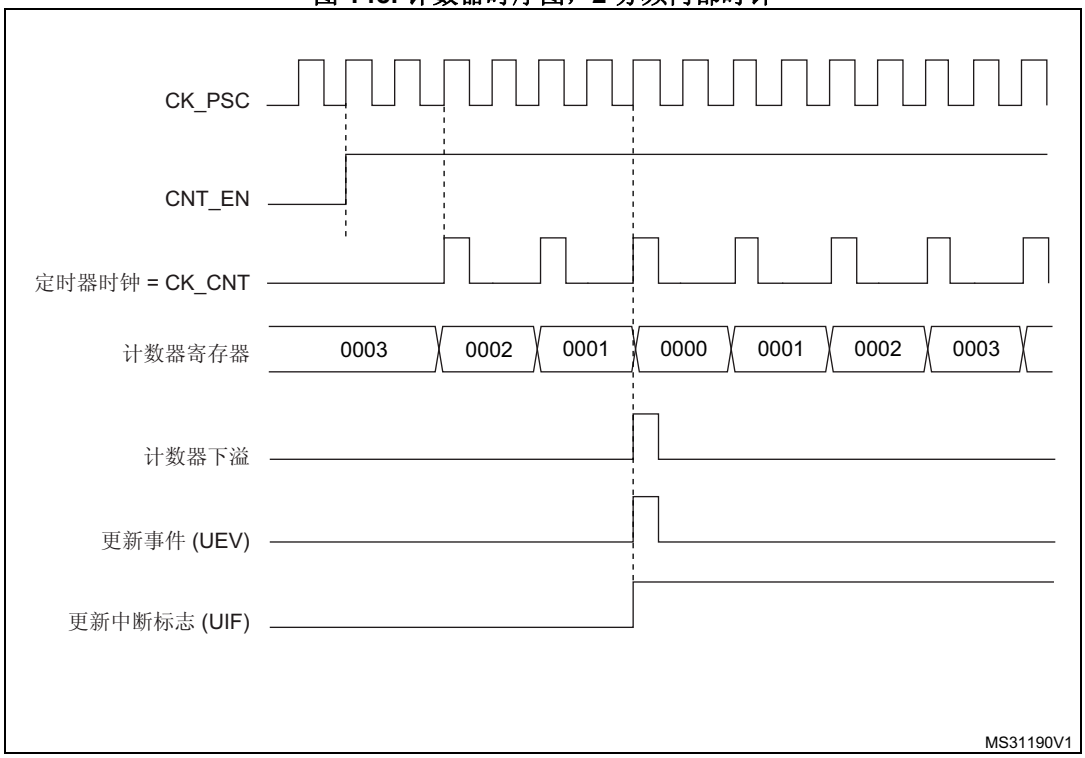


图 149. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36

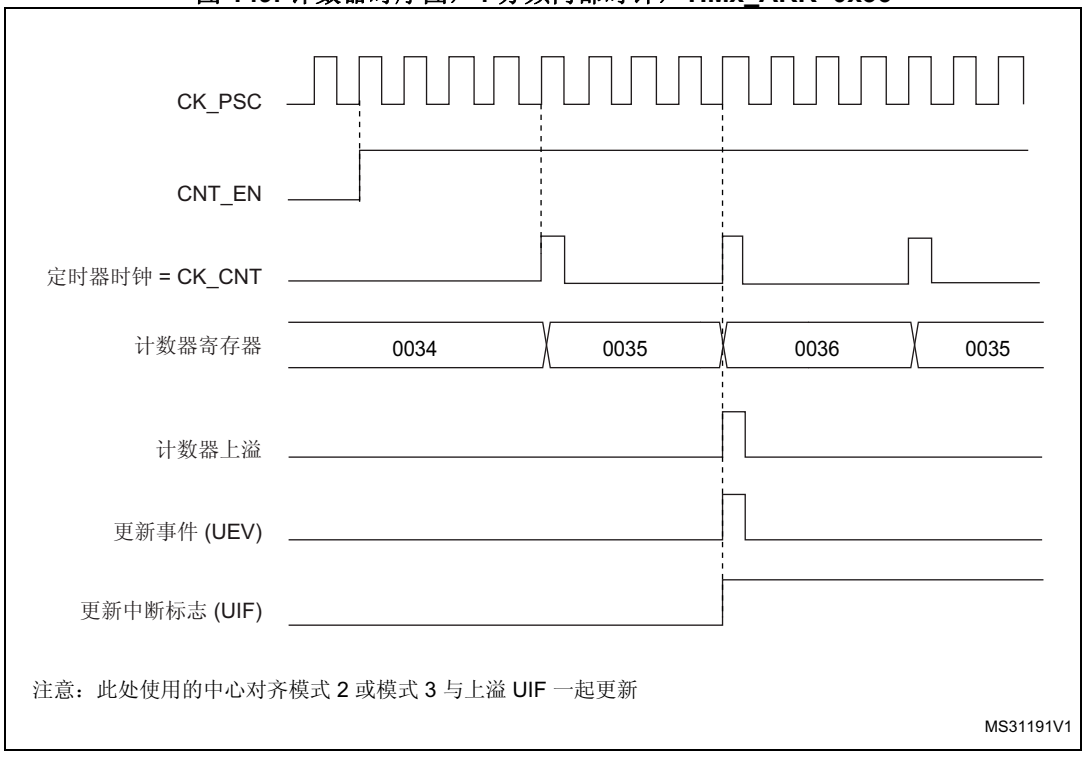


图 150. 计数器时序图，N 分频内部时钟

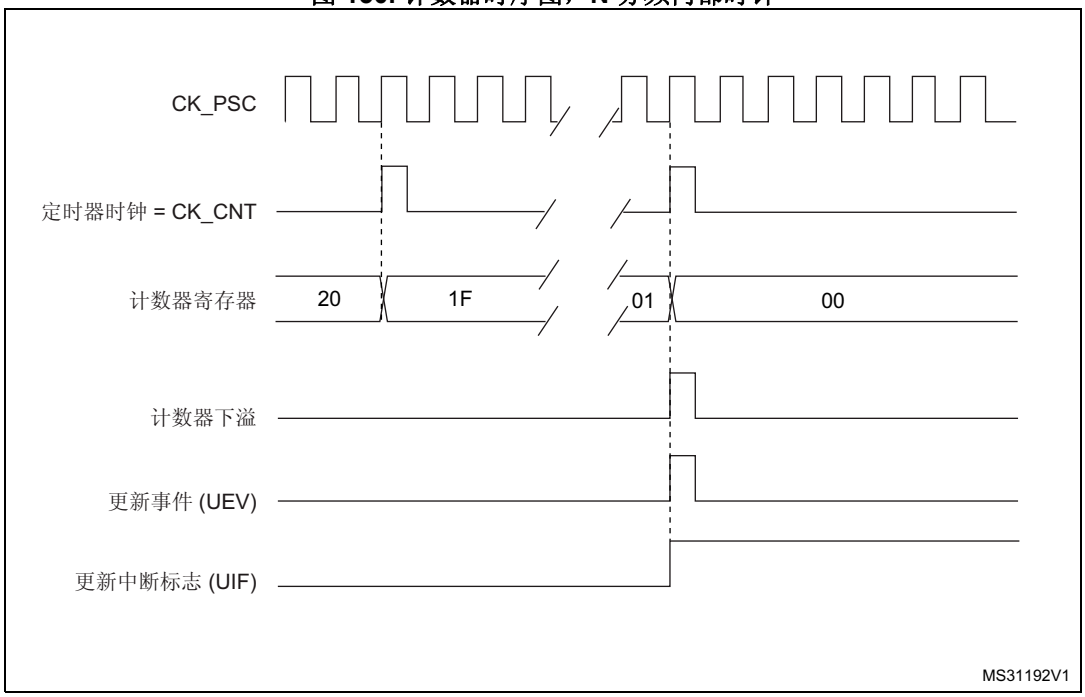


图 151. 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

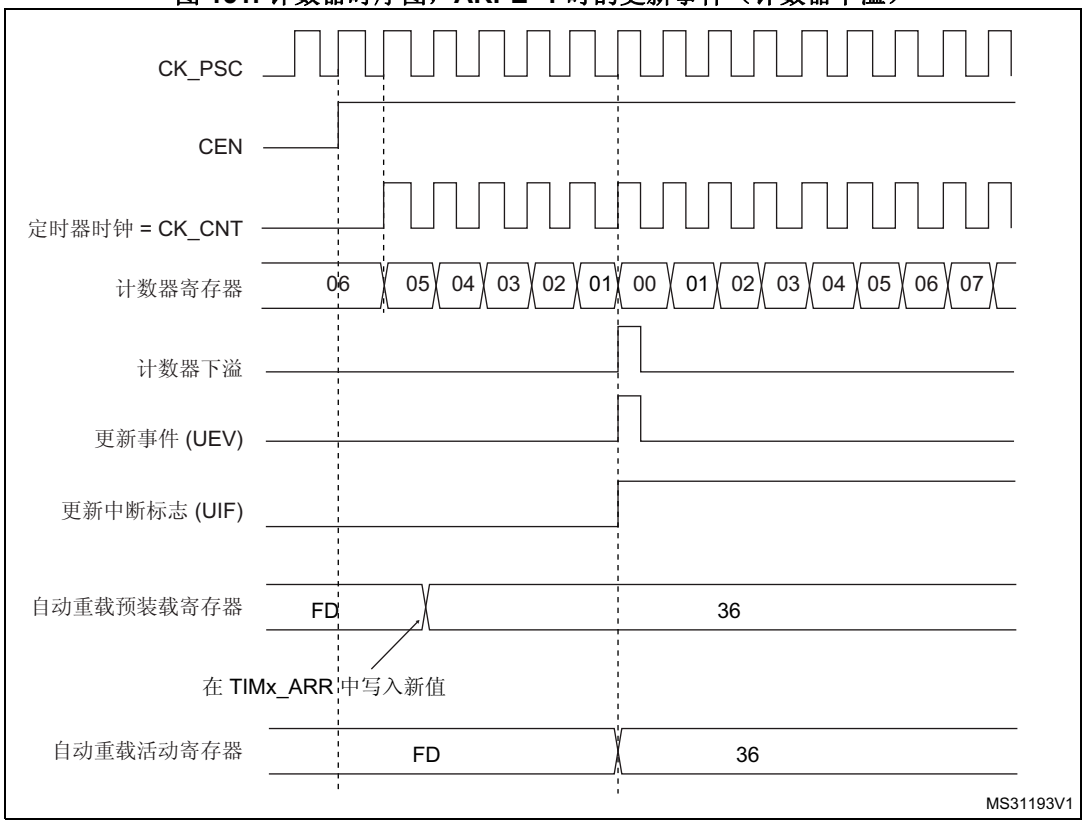
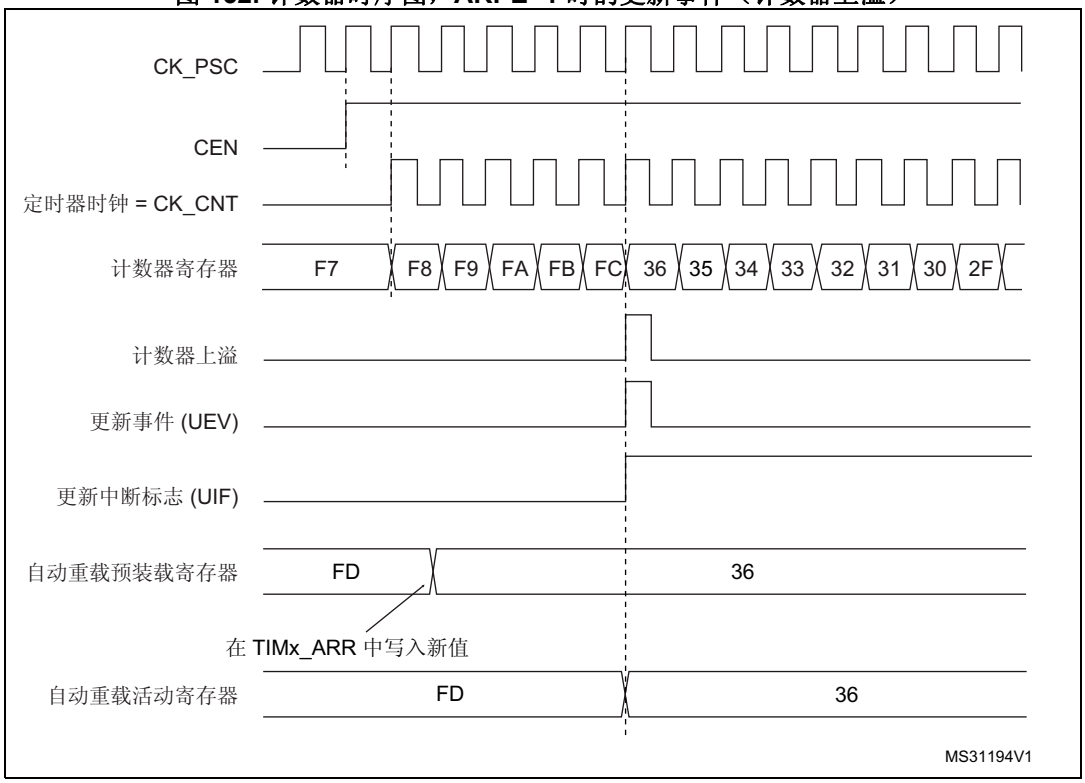


图 152. 计数器时序图，ARPE=1 时的更新事件（计数器上溢）



22.3.3 重复计数器

*第 22.3.1 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：

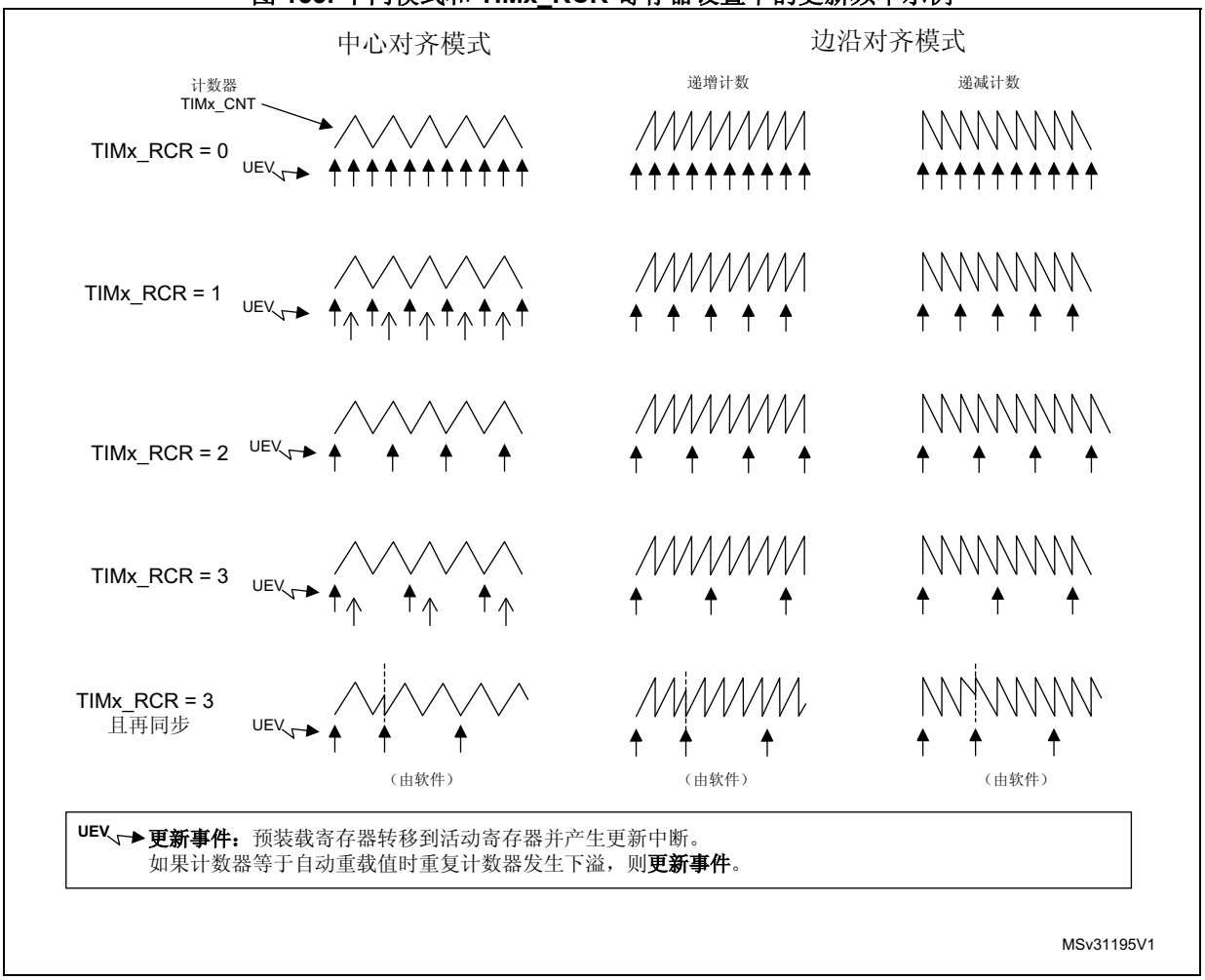
- 递增计数模式下的每个计数器上溢，
- 递减计数模式下的每个计数器下溢，
- 中心对齐模式下每个计数器上溢和计数器下溢。尽管这使得最大重复次数不超过 32768 个 PWM 周期，但在每个 PWM 周期内可更新占空比两次。当在中心对齐模式下，每个 PWM 周期仅刷新一次比较寄存器时，由于模式的对称性，最大分辨率为 $2xT_{ck}$。

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 153*）。当更新事件由软件（通过将 TIMx_EGR 寄存器的 UG 位置 1）或硬件（通过从模式控制器）生成时，无论重复计数器的值为多少，更新事件都将立即发生，并且在重复计数器中重新装载 TIMx_RCR 寄存器的内容。

在中心对齐模式下，如果 RCR 值为奇数，更新事件将在上溢或下溢时发生，这取决于何时写入 RCR 寄存器以及何时启动计数器：如果在启动计数器前写入 RCR，则 UEV 在上溢时发生。如果在启动计数器后写入 RCR，则 UEV 在下溢时发生。

例如，如果 RCR = 3，UEV 将在每个周期的第四个上溢或下溢事件时产生（取决于何时写入 RCR）。

图 153. 不同模式和 TIMx_RCR 寄存器设置下的更新频率示例



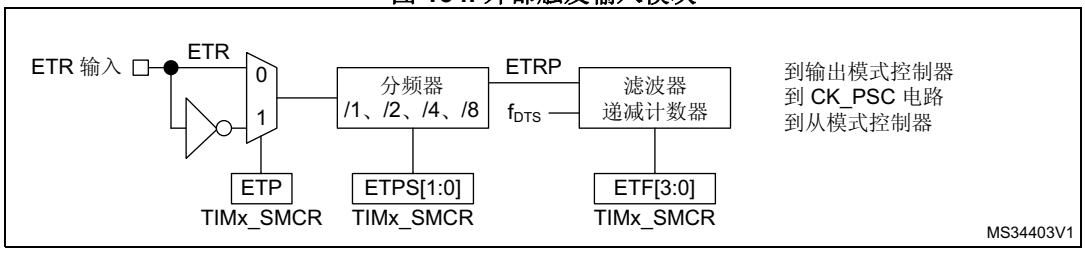
22.3.4 外部触发输入

定时器具有一个外部触发输入 ETR，它可用作：

- 外部时钟（外部时钟模式 2，请参见*第 22.3.5 节*）
- 用于从模式的触发信号（请参见*第 22.3.25 节*）
- 用于逐周期电流调节的 PWM 复位输入（请参见*第 22.3.17 节*）

下面的*图 154* 介绍了 ETR 输入的调节过程。输入极性通过 TIMxSMCR 寄存器中的 ETP 位定义。触发信号可通过 ETPS[1:0] 位字段编程的分频比进行预分频，然后通过 ETF[3:0] 位字段进行数字滤波。

图 154. 外部触发输入模块



22.3.5 时钟选择

计数器时钟可由下列时钟源提供：

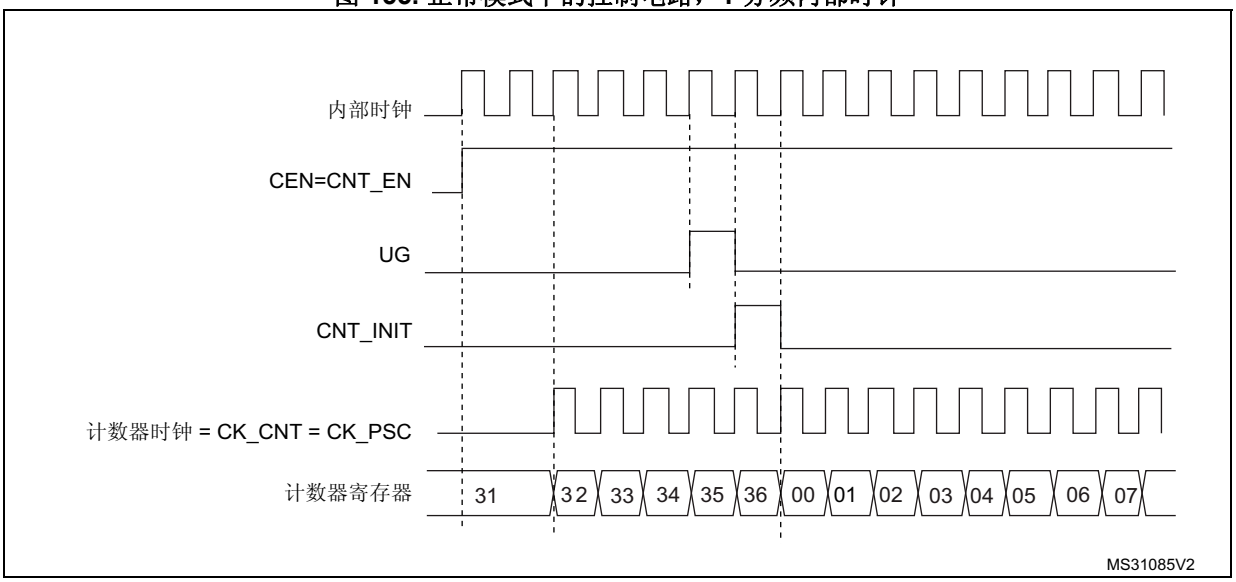
- 内部时钟 (CK_INT)
- 外部时钟模式 1：外部输入引脚
- 外部时钟模式 2：外部触发输入 ETR
- 编码器模式

内部时钟源 (CK_INT)

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 CK_INT 提供。

图 155 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）。

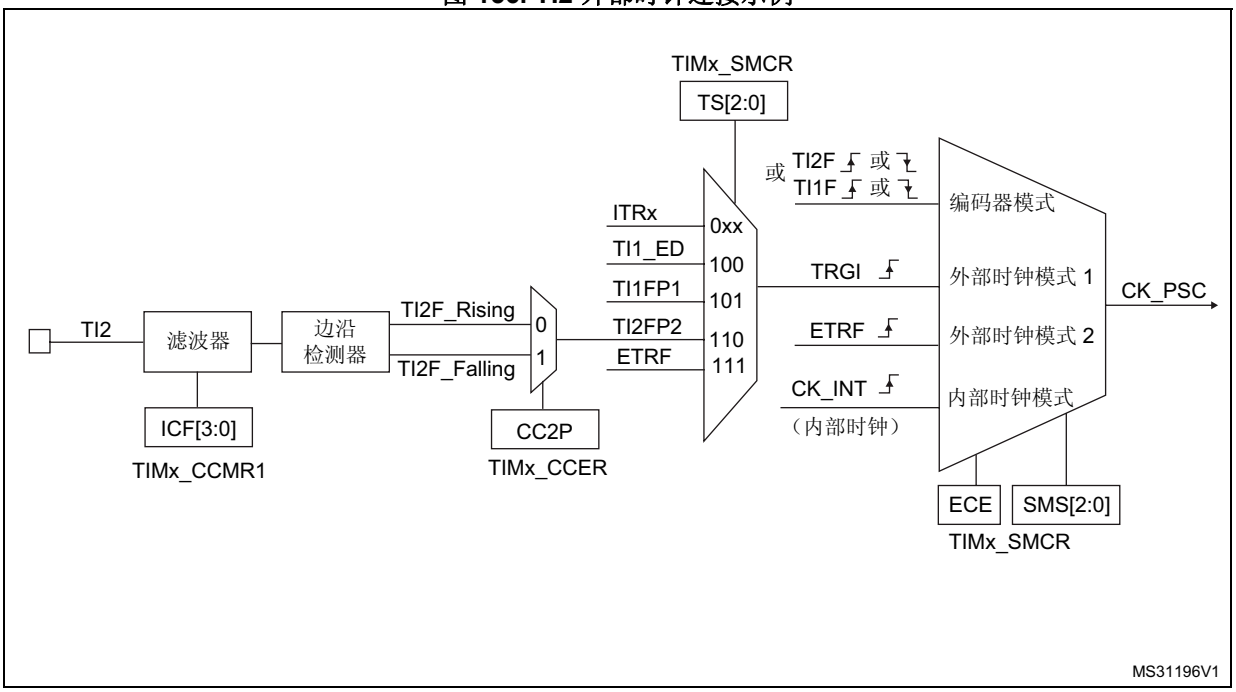
图 155. 正常模式下的控制电路，1 分频内部时钟



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 156. TI2 外部时钟连接示例



例如，要使递增计数器在 TI2 输入出现上升沿时计数，请执行以下步骤：

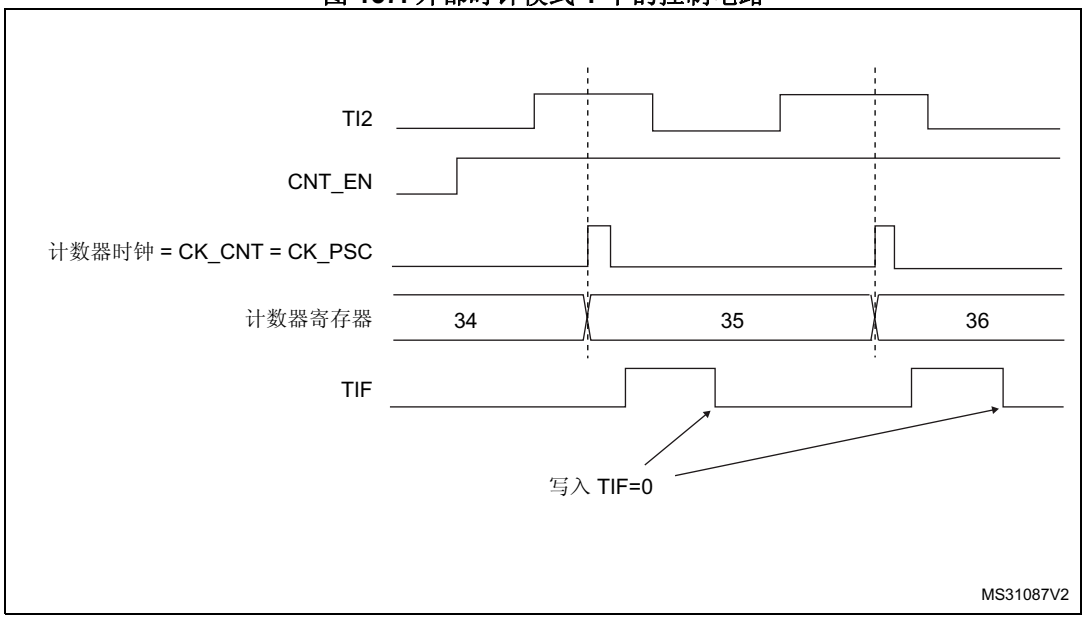
1. 通过在 TIMx_CCMR1 寄存器中写入 CC2S = “01” 来配置通道 2，使其能够检测 TI2 输入的上升沿。
2. 通过在 TIMx_CCMR1 寄存器中写入 IC2F[3:0] 位来配置输入滤波带宽（如果不需要任何滤波器，请保持 IC2F=0000）。
3. 通过在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=0 来选择上升沿极性。
4. 通过在 TIMx_SMCR 寄存器中写入 SMS=111，使定时器在外部时钟模式 1 下工作。
5. 通过在 TIMx_SMCR 寄存器中写入 TS=110 来选择 TI2 作为触发输入源。
6. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

注： *由于捕获预分频器不用于触发操作，因此无需对其进行配置。*

当 TI2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

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

图 157. 外部时钟模式 1 下的控制电路



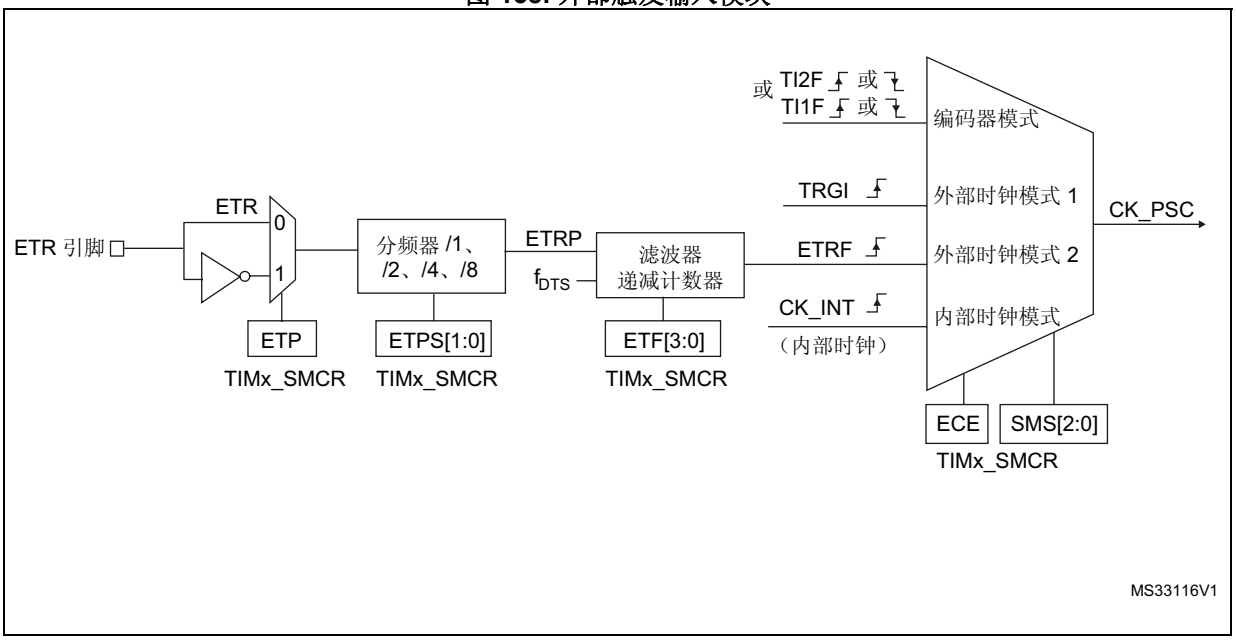
外部时钟源模式 2

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 ETR 出现上升沿或下降沿时计数。

图 158 简要介绍了外部触发输入模块。

图 158. 外部触发输入模块



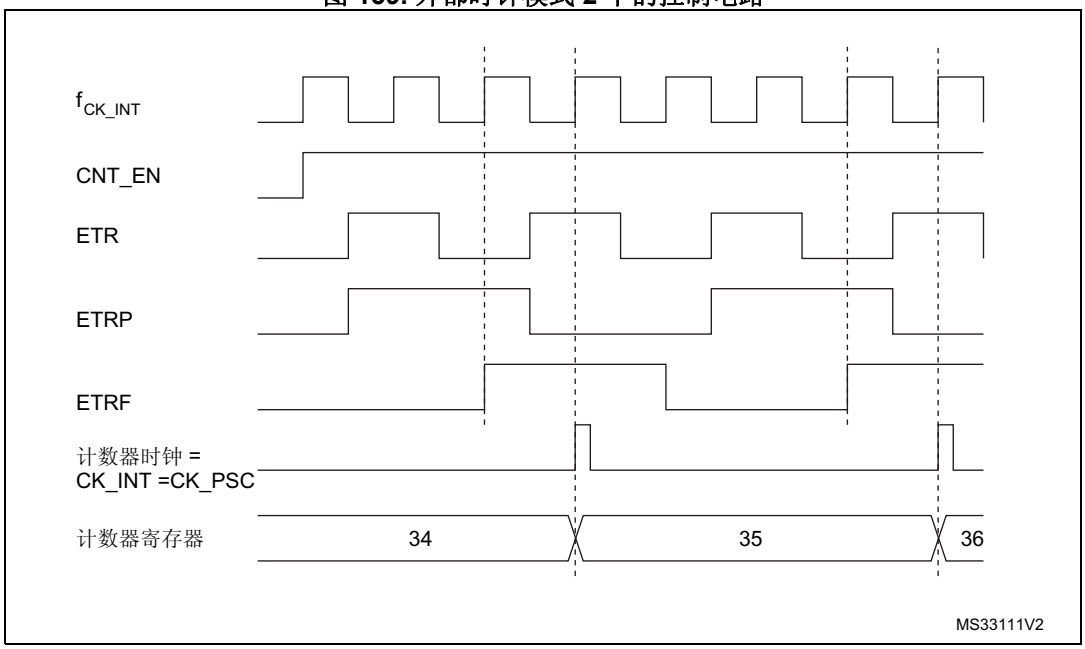
例如，要使递增计数器在 ETR 每出现 2 个上升沿时计数，请执行以下步骤：

1. 由于此例中不需滤波器，因此在 TIMx_SMCR 寄存器中写入 ETF[3:0]=0000。
2. 通过在 TIMx_SMCR 寄存器中写入 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 ETR 引脚的上升沿检测。
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。

ETR 每出现 2 个上升沿，计数器计数一次。

ETR 的上升沿与实际计数器时钟之间的延迟是由于 ETRP 信号的重新同步电路引起的。

图 159. 外部时钟模式 2 下的控制电路



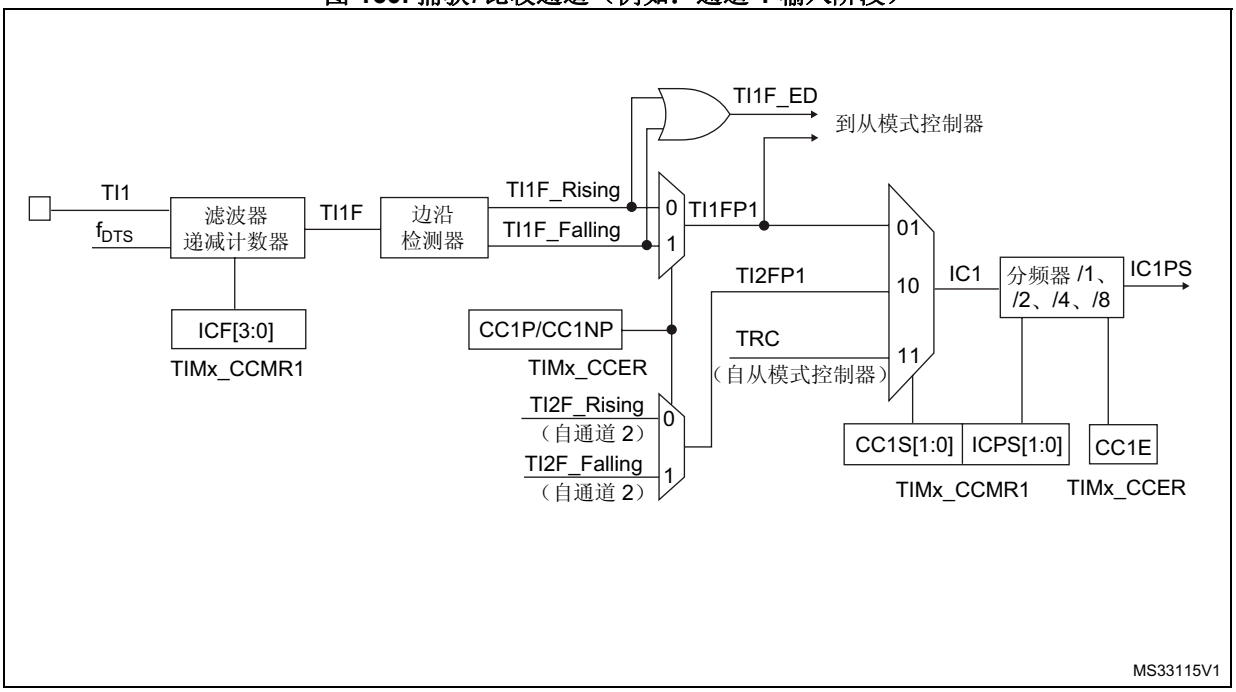
22.3.6 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入阶段（数字滤波、多路复用和预分频器，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 160 到*图 163* 概括介绍了一个捕获/比较通道。

输入阶段对相应的 TIx 输入进行采样，生成一个滤波后的信号 TIxF。然后，带有极性选择功能的边沿检测器生成一个信号 (TIxFPx)，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 160. 捕获/比较通道（例如：通道 1 输入阶段）



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 161. 捕获/比较通道 1 主电路

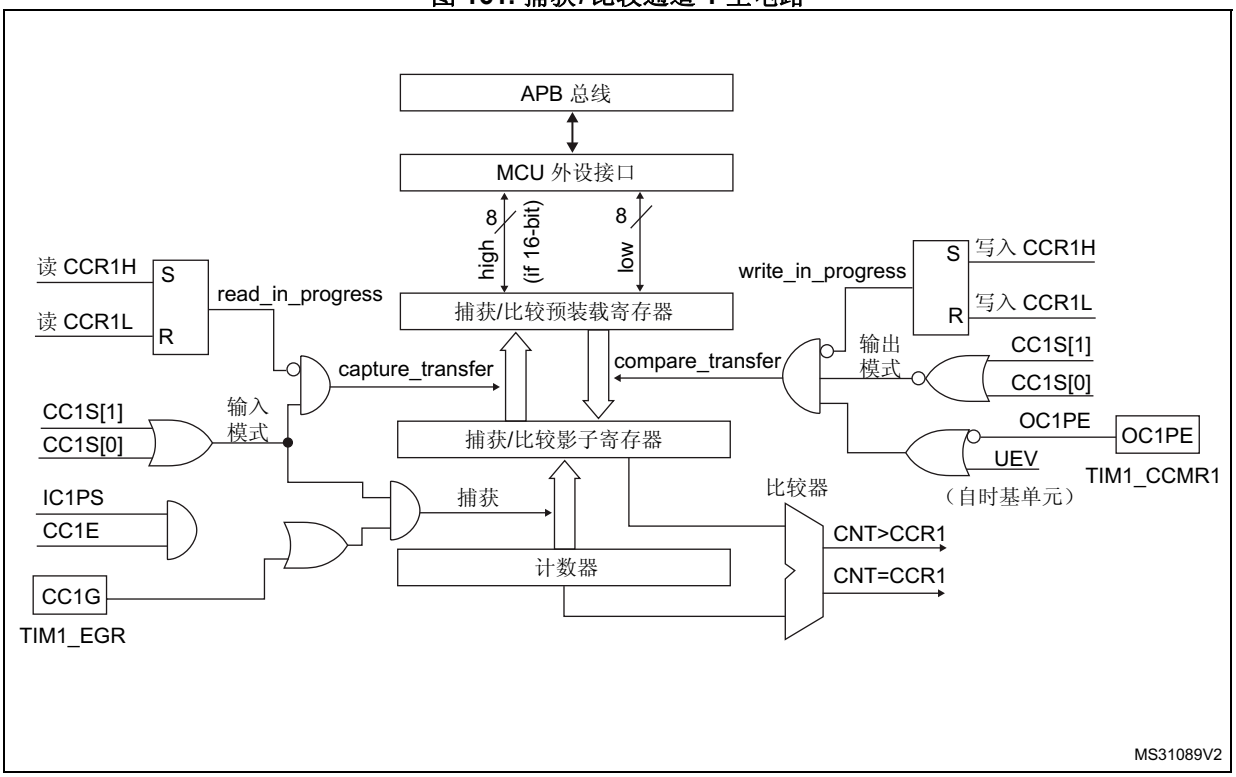
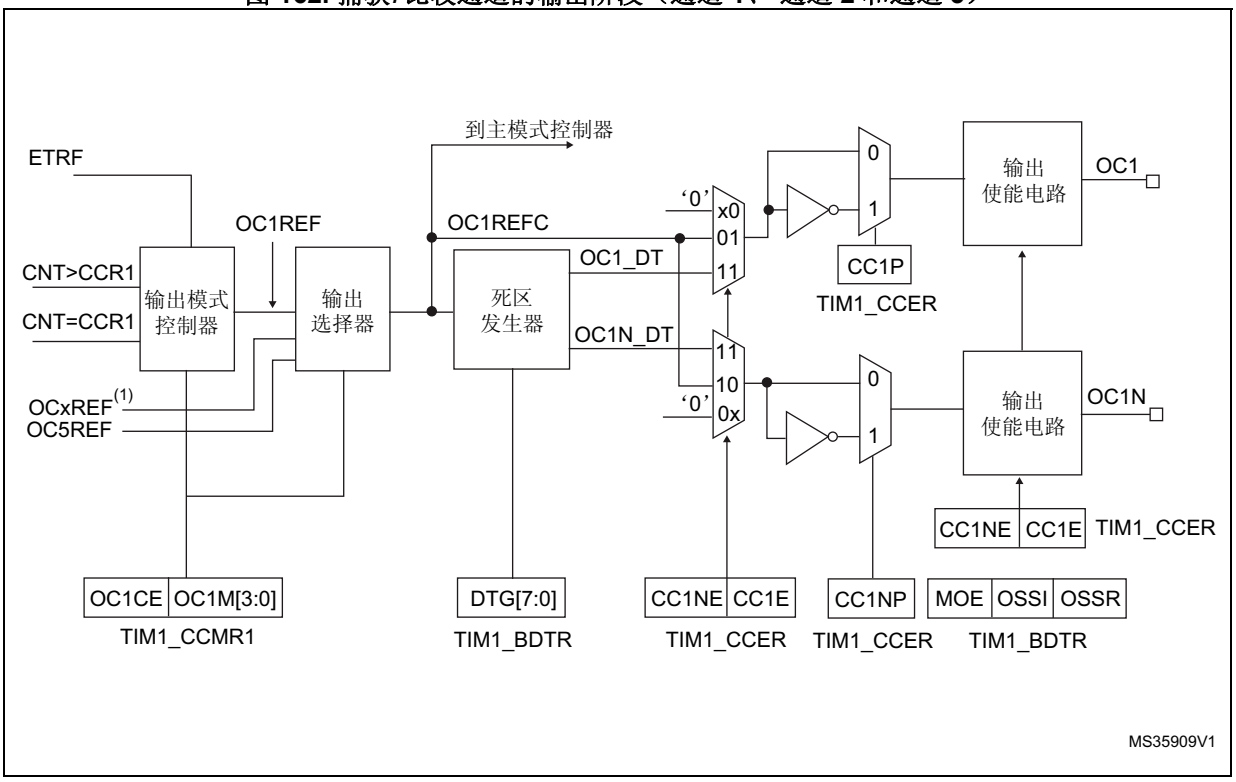


图 162. 捕获/比较通道的输出阶段（通道 1、 通道 2 和通道 3）



1. OCxREF，其中 x 为互补通道的序号

图 163. 捕获/比较通道的输出阶段（通道 4）

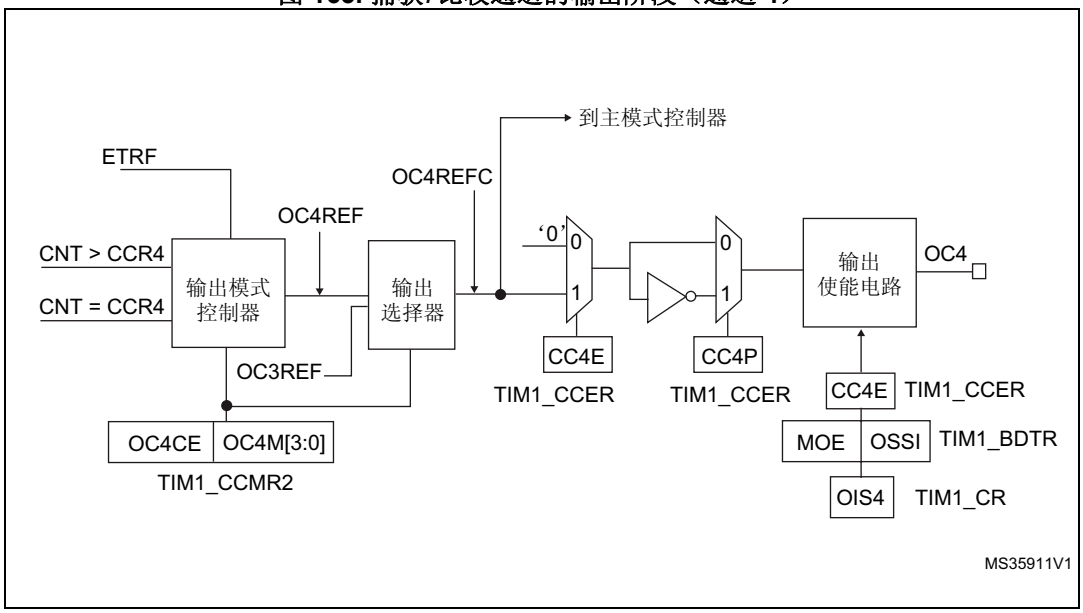
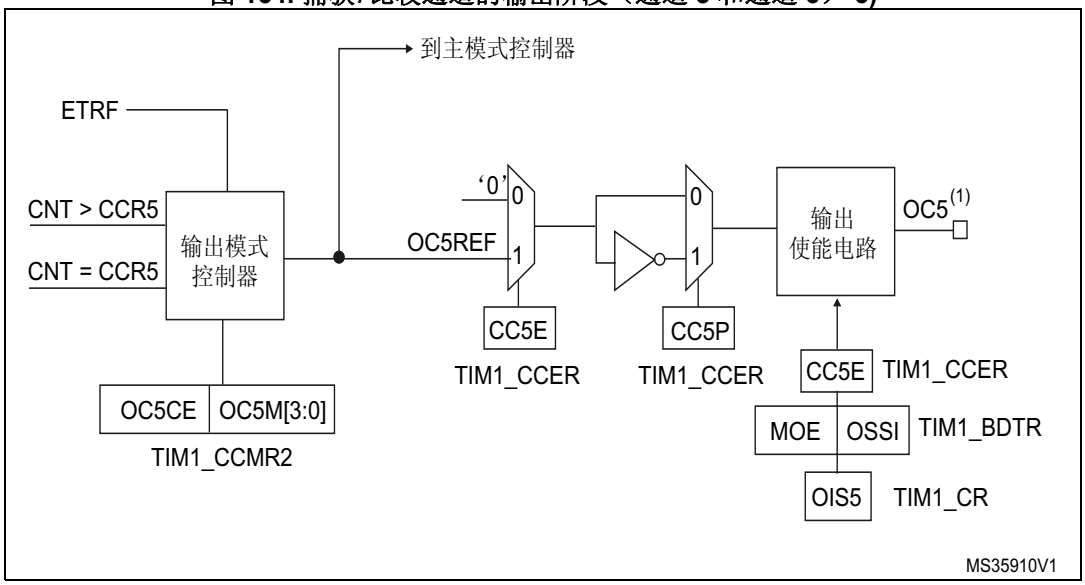


图 164. 捕获/比较通道的输出阶段（通道 5 和通道 6） 6)



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。始终可通过读写操作访问预装载寄存器。

在捕获模式下，捕获实际发生在影子寄存器中，然后将影子寄存器的内容复制到预装载寄存器中。

在比较模式下，预装载寄存器的内容将复制到影子寄存器中，然后将影子寄存器的内容与计数器进行比较。

22.3.7 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，则会将重复捕获标志 CCxOF（TIMx_SR 寄存器）置 1。可通过软件将 CCxIF 清零，方法是：向 CCxIF 写入“0”，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 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 必须连接到 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。
- 根据连接到定时器的信号，对所需的输入滤波带宽进行编程（如果输入为 TIx 之一，则对 TIMx_CCMRx 寄存器中的 ICxF 位进行编程）。假设信号边沿变化时，输入信号最多在 5 个内部时钟周期内发生抖动。因此，我们必须将滤波带宽设置为大于 5 个内部时钟周期。在检测到 8 个具有新电平的连续采样（以 f_{DTS} 频率采样）后，可以确认 TI1 上的跳变沿。然后向 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 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。

要处理重复捕获，建议在读出捕获溢出标志之前读取数据。这样可避免丢失在读取捕获溢出标志之后与读取数据之前可能出现的重复捕获信息。

注： *通过软件将 TIMx_EGR 寄存器中的相应 CCxG 位置 1 可生成 IC 中断和/或 DMA 请求。*

22.3.8 PWM 输入模式

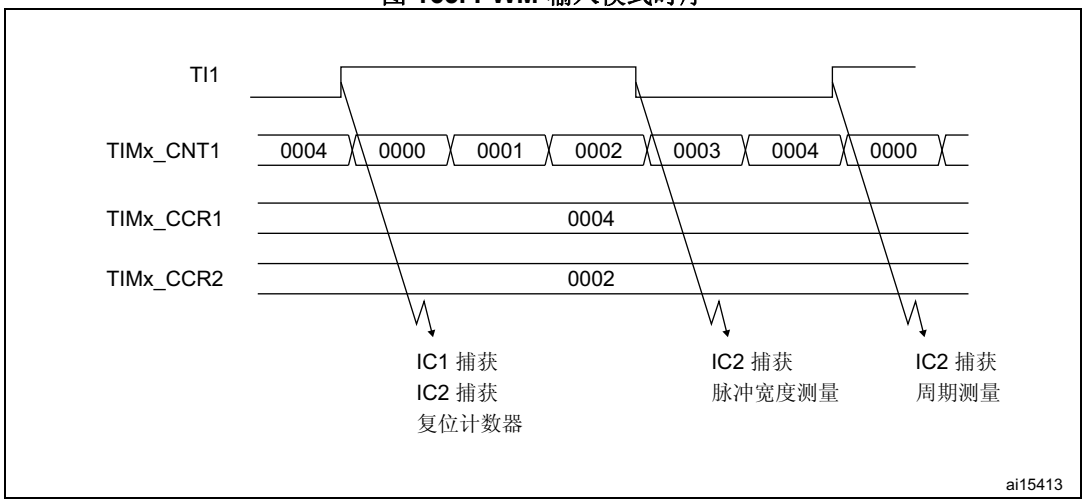
此模式是输入捕获模式的一个特例。其实现步骤与输入捕获模式基本相同，仅存在以下不同之处：

- 两个 ICx 信号被映射至同一个 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。

例如，可通过以下步骤对应用于 TI1 的 PWM 的周期（位于 TIMx_CCR1 寄存器中）和占空比（位于 TIMx_CCR2 寄存器中）进行测量（取决于 CK_INT 频率和预分频器的值）：

- 选择 TIMx_CCR1 的有效输入：向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01（选择 TI1）。
- 选择 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 TI1）。
- 选择 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 101（选择 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 165. PWM 输入模式时序



22.3.9 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（OCxREF 和 OCx/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 (OCXREF/OCx) 强制设置为有效电平，只需向相应 TIMx_CCMRx 寄存器中的 OCxM 位写入 0101。OCXREF 进而强制设置为高电平（OCxREF 始终为高电平有效），同时 OCx 获取 CCxP 极性位的相反值。

例如：CCxP=0（OCx 高电平有效）=> 将 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 OCxREF 信号强制设置为低电平。

无论如何，TIMx_CCRx 影子寄存器与计数器之间的比较仍会执行，而且允许将标志置 1。因此可发送相应的中断和 DMA 请求。下面的输出比较模式一节对此进行了介绍。

22.3.10 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。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载寄存器。

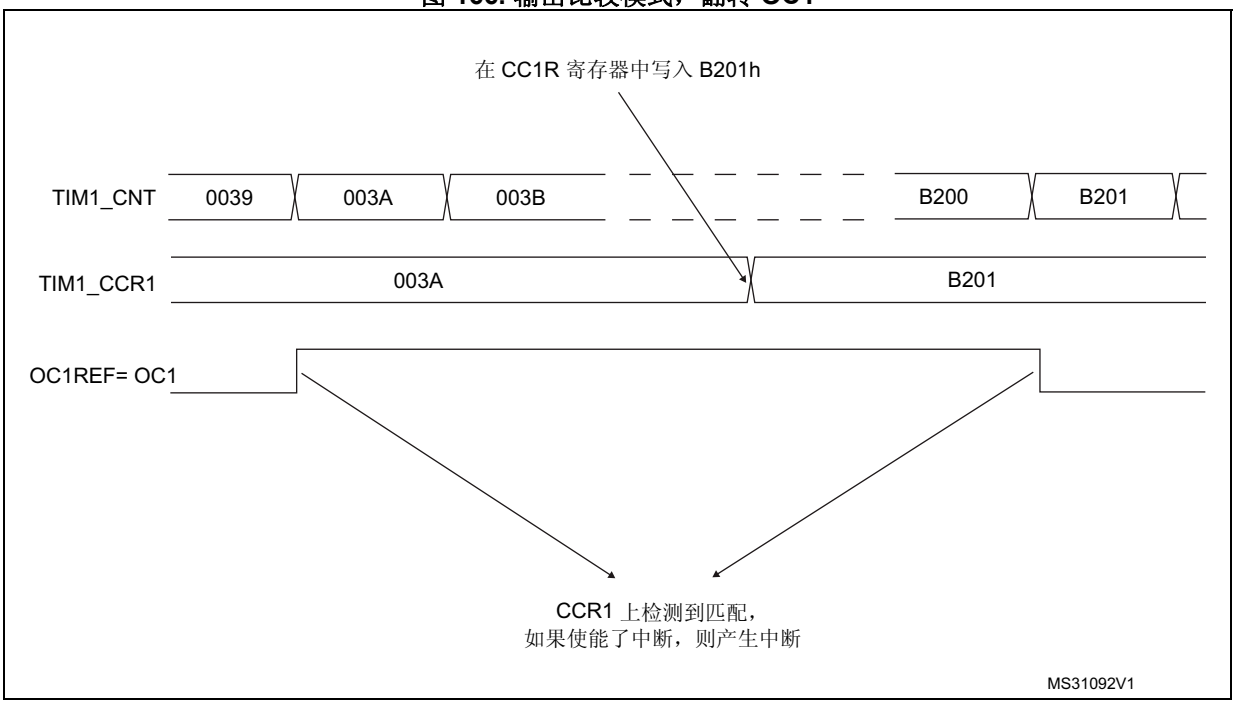
在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出毫无影响。同步的精度可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

步骤

1. 选择计数器时钟（内部、外部、预分频器）。
2. 在 TIMx_ARR 和 TIMx_CCRx 寄存器中写入所需数据。
3. 如果要生成中断请求，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 OCx 输出引脚
   - 写入 OCxPE = 0 以禁止预装载寄存器
   - 写入 CCxP = 0 以选择高电平有效极性
   - 写入 CCxE = 1 以使能输出
5. 通过将 TIMx_CR1 寄存器中的 CEN 位置 1 来使能计数器。

可通过软件随时更新 TIMx_CCRx 寄存器以控制输出波形，前提是未使能预加载寄存器（OCxPE=“0”，否则 TIMx_CCRx 影子寄存器仅在下一更新事件 UEV 发生时进行更新）。*图 166* 给出了一个示例。

图 166. 输出比较模式，翻转 OC1



22.3.11 PWM 模式

脉冲宽度调制模式可以生成一个信号，该信号频率由 TIMx_ARR 寄存器值决定，其占空比则由 TIMx_CCRx 寄存器值决定。

各通道可以独立选择 PWM 模式（每个 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。

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 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。

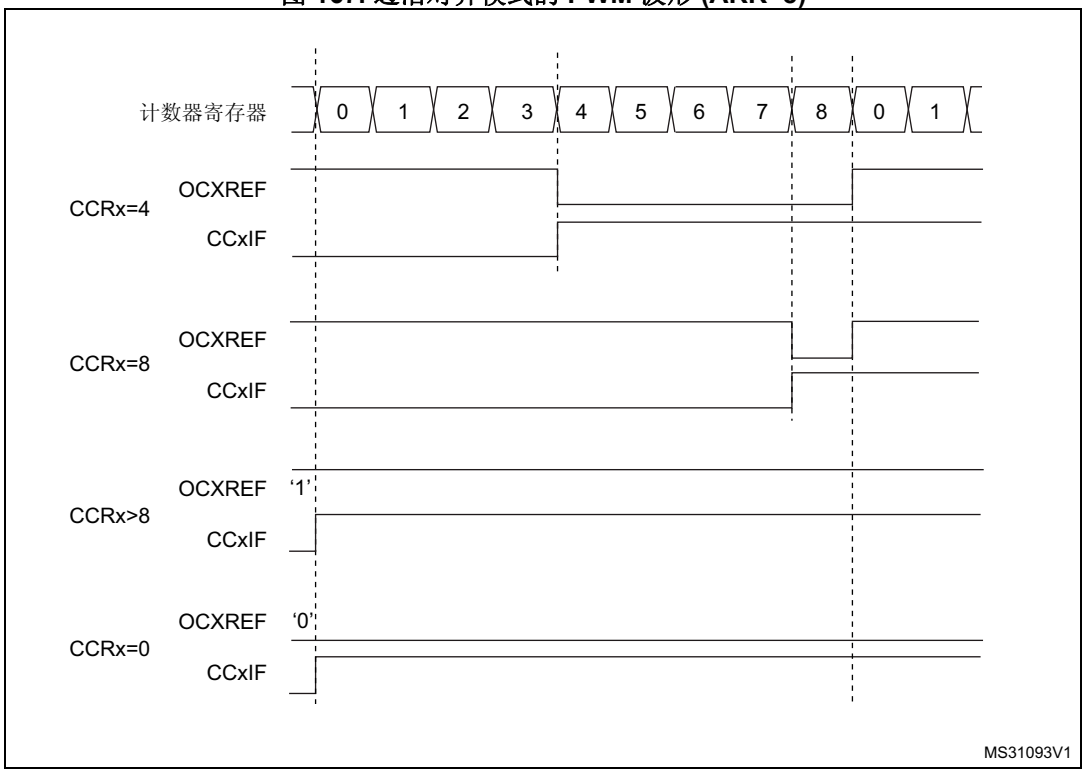
PWM 边沿对齐模式

- 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 570 页上的递增计数模式*。

  以下以 PWM 模式 1 为例。只要 TIMx_CNT < TIMx_CCRx，PWM 参考信号 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 OCxREF 保持为“1”。如果比较值为 0，则 OCxRef 保持为“0”。*图 167* 举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

图 167. 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 573 页上的递减计数模式*。

  在 PWM 模式 1 下，只要 TIMx_CNT > TIMx_CCRx，参考信号 OCxRef 即为低电平，否则其为高电平。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重载值，则 OCxREF 保持为“1”。此模式下不可能产生 0% 的 PWM 波形。

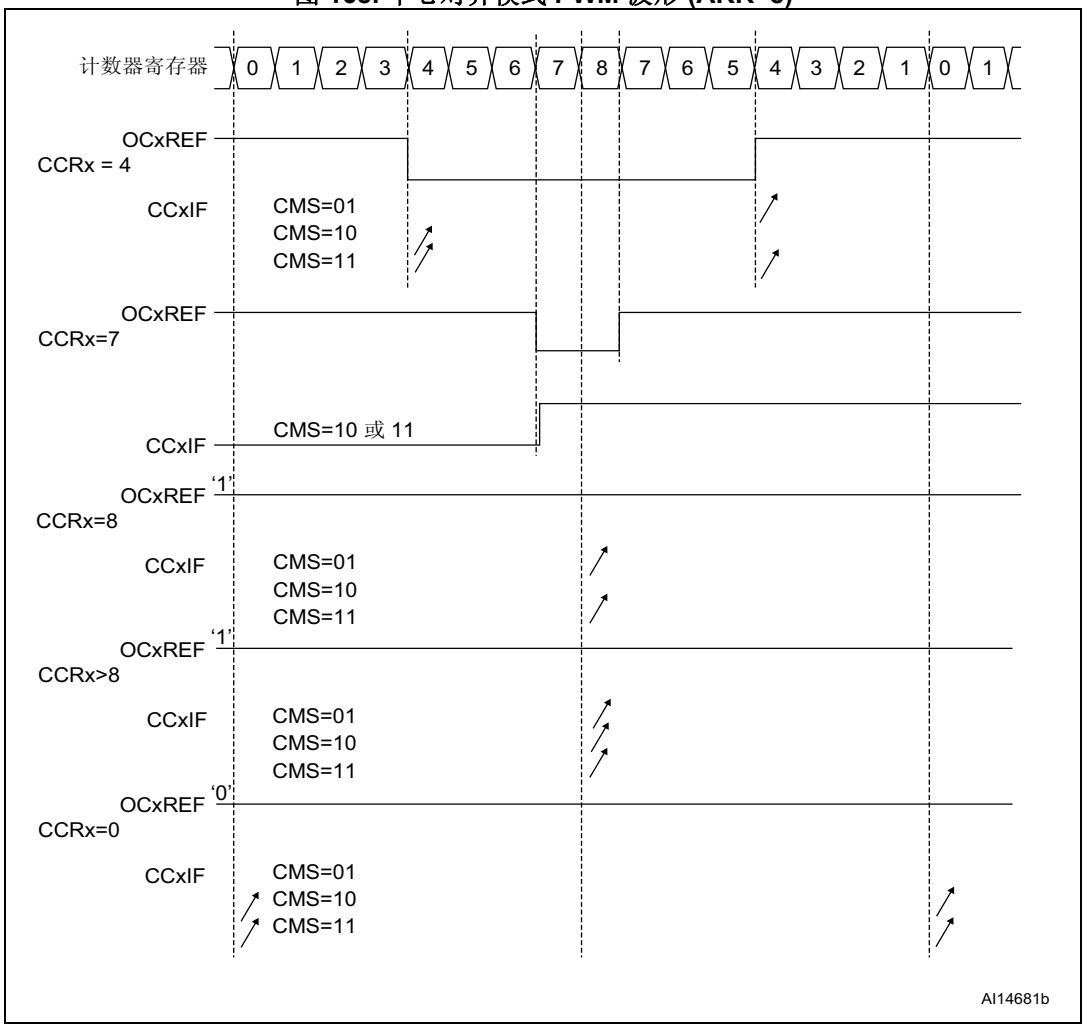
PWM 中心对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为“00”（其余所有配置对 OCxRef/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 576 页上的中心对齐模式（递增/递减计数）*。

图 168 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 168. 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，不得同时通过软件修改 DIR 和 CMS 位。
- 不建议在运行中心对齐模式时对计数器执行写操作，否则将发生意想不到的结果。尤其是：
  - 如果写入计数器中的值大于自动重载值 (TIMx_CNT>TIMx_ARR)，计数方向不会更新。例如，如果计数器之前递增计数，则继续递增计数。
  - 如果向计数器写入 0 或 TIMx_ARR 的值，计数方向会更新，但不生成更新事件 UEV。
- 使用中心对齐模式最为保险的方法是：在启动计数器前通过软件生成更新（将 TIMx_EGR 寄存器中的 UG 位置 1），并且不要在计数器运行过程中对其执行写操作。

22.3.12 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：

- OC1REFC（或 OC2REFC）由 TIMx_CCR1 和 TIMx_CCR2 控制
- OC3REFC（或 OC4REFC）由 TIMx_CCR3 和 TIMx_CCR4 控制

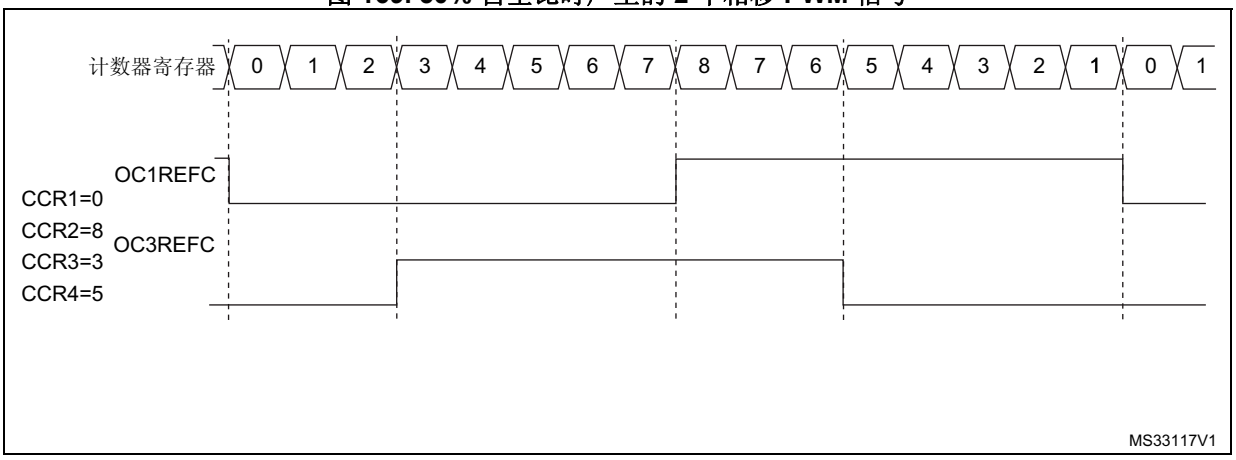
两个通道可以独立选择不对称 PWM 模式（每对 CCR 寄存器一个 OCx 输出），只需向 TIMx_CCMRx 寄存器的 OCxM 位写入“1110”（不对称 PWM 模式 1）或“1111”（不对称 PWM 模式 2）。

注： *出于兼容性原因，OCxM[3:0] 位字段分为两部分，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 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 OC2REF 信号或 OC2REFC 信号。

图 169 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 在不对称 PWM 模式 1 下配置）。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 169. 50% 占空比时产生的 2 个相移 PWM 信号



22.3.13 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 OCxREFC 由两个参考 PWM 的逻辑或运算或者逻辑与运算组合组成。

- OC1REFC（或 OC2REFC）由 TIMx_CCR1 和 TIMx_CCR2 控制
- OC3REFC（或 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 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）。

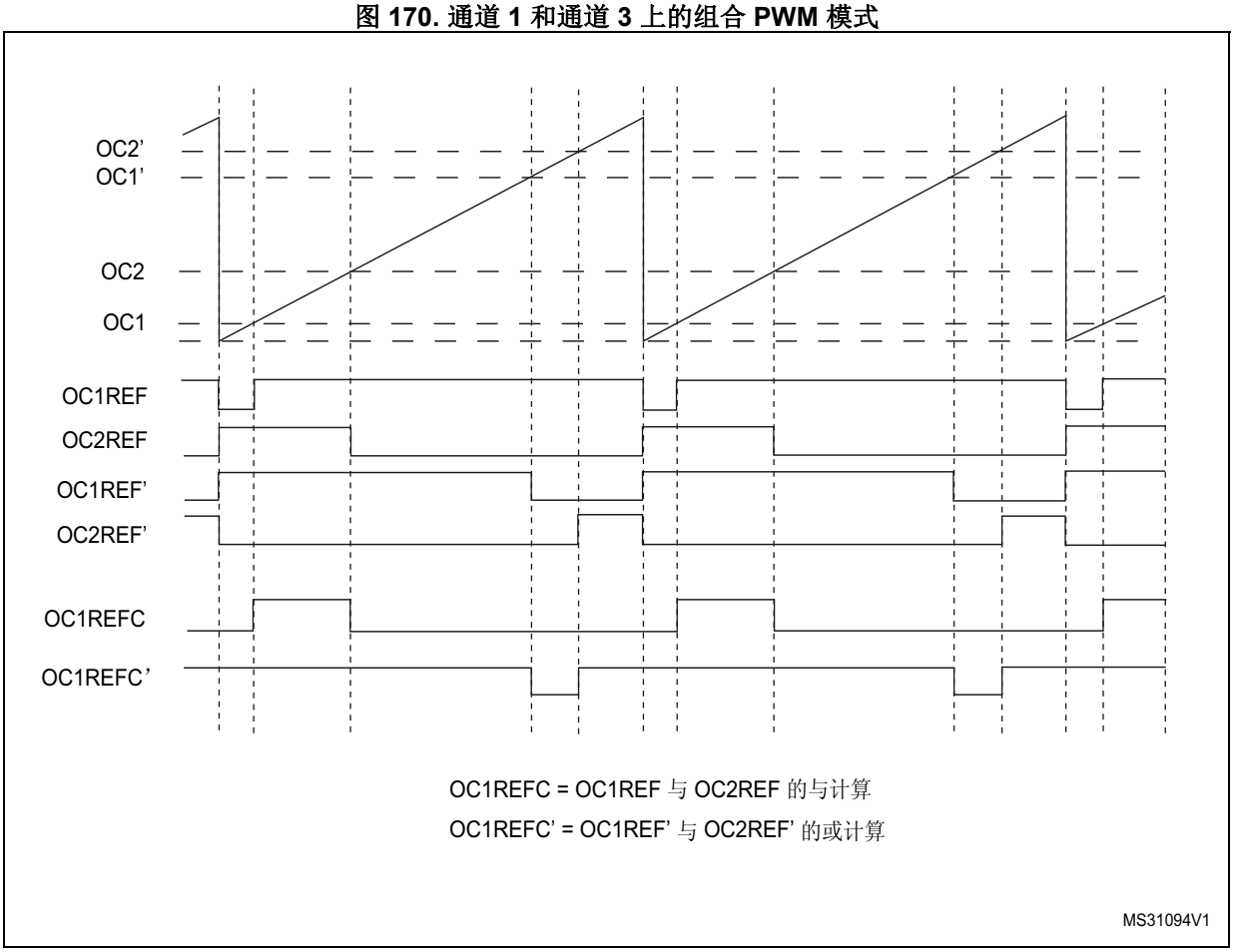
当给定通道用作组合 PWM 通道时，其互补通道必须在相反的 PWM 模式下配置（例如，一个通道在组合 PWM 模式 1 下配置，另一个通道在组合 PWM 模式 2 下配置）。

注： *出于兼容性原因，OCxM[3:0] 位字段分为两部分，最高有效位与最低有效的 3 位不相邻。*

图 170 显示了不对称 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 在组合 PWM 模式 2 下配置，
- 通道 2 在 PWM 模式 1 下配置，
- 通道 3 在组合 PWM 模式 2 下配置，
- 通道 4 在 PWM 模式 1 下配置。

图 170. 通道 1 和通道 3 上的组合 PWM 模式



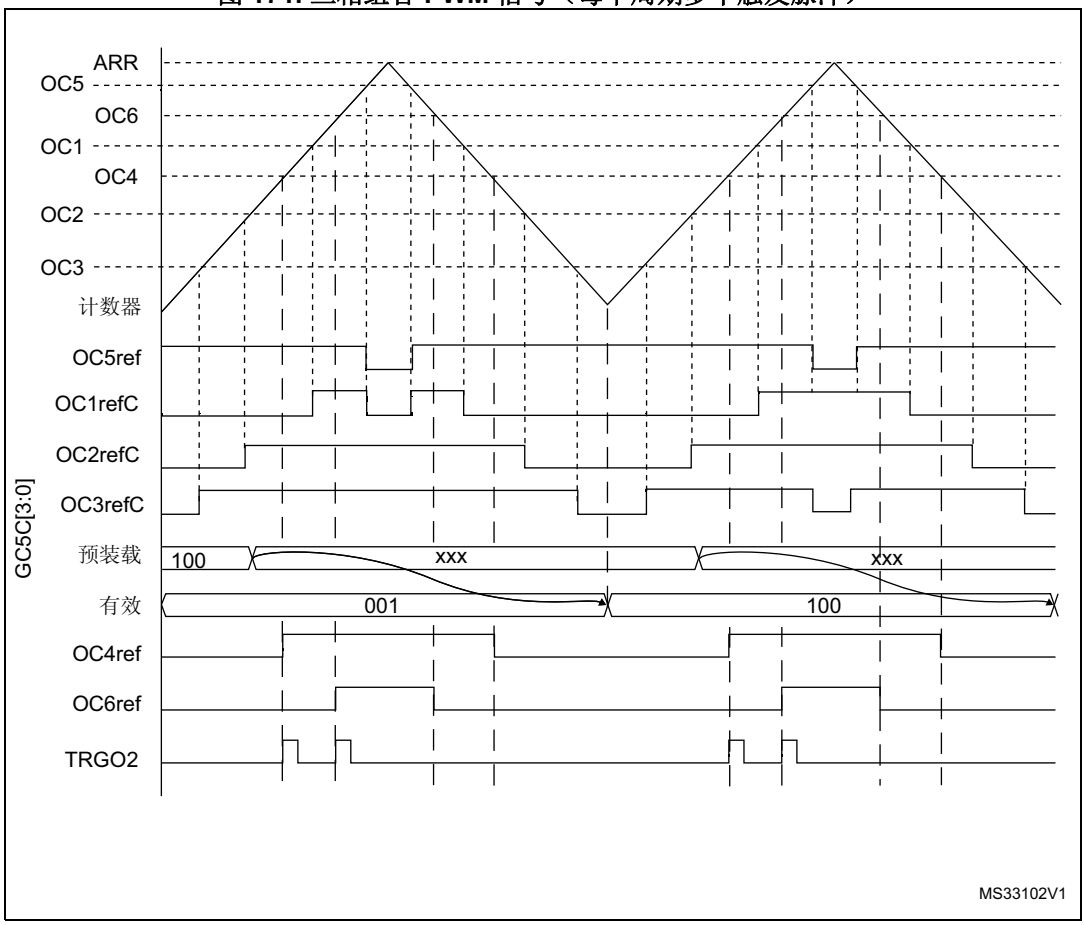
22.3.14 组合三相 PWM 模式

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。OC5REF 信号用于定义产生的组合信号。凭借 TIMx_CCR5 中的 3 位 GC5C[3:1]，可以选择 OC5REF 与哪个参考信号组合。产生的信号 OCxREFC 由两个参考 PWM 的逻辑与运算组合组成。

- 如果 GC5C1 置 1，则 OC1REFC 由 TIMx_CCR1 和 TIMx_CCR5 控制
- 如果 GC5C2 置 1，则 OC2REFC 由 TIMx_CCR2 和 TIMx_CCR5 控制
- 如果 GC5C3 置 1，则 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 171. 三相组合 PWM 信号（每个周期多个触发脉冲）



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 22.3.26 节：ADC 同步*。

22.3.15 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 OCx 或互补输出 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 OCx 和 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 635 页上的表 126：具有断路功能的互补通道 OCx 和 OCxN 的输出控制位*。应当注意，切换至空闲状态（MOE 下降到 0）的时刻，死区仍然有效。

CCxE 和 CCxNE 位同时置 1 并且 MOE 位置 1（如果存在断路）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 OCxREF 生成 2 个输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高电平有效：

- 输出信号 OCx 与参考信号相同，只是其上升沿相对参考上升沿存在延迟。
- 输出信号 OCxN 与参考信号相反，并且其上升沿相对参考下降沿存在延迟。

如果延迟时间大于有效输出（OCx 或 OCxN）的宽度，则不会产生相应的脉冲。

下图所示为死区发生器的输出信号与参考信号 OCxREF 之间的关系。（在这些示例中，假定 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1）

图 172. 带死区插入的互补输出

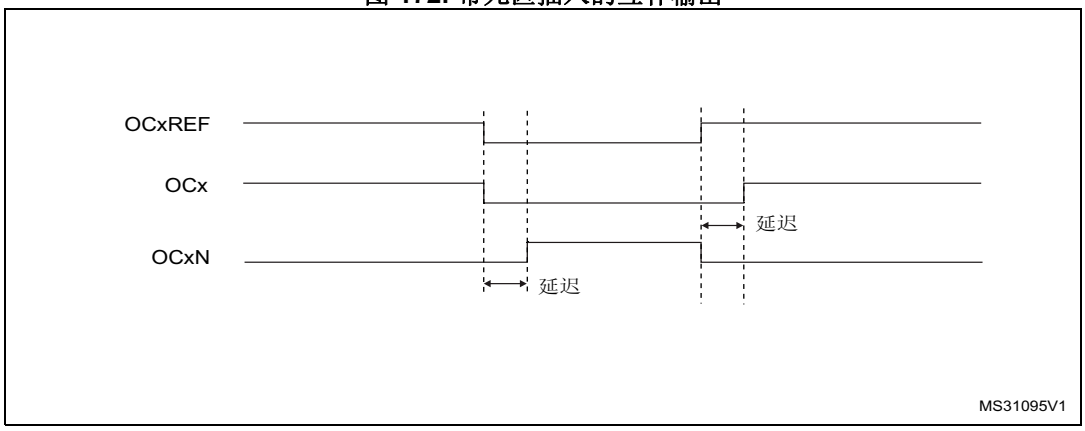


图 173. 延迟时间大于负脉冲宽度的死区波形

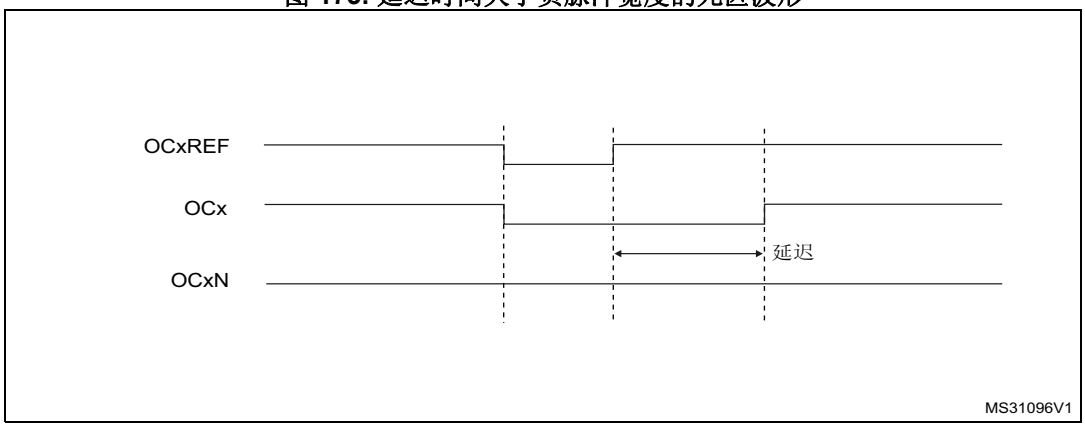
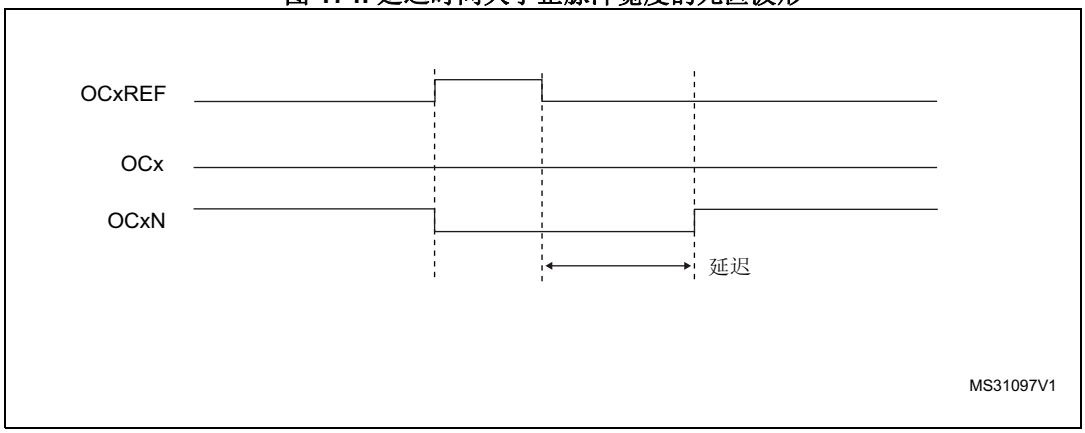


图 174. 延迟时间大于正脉冲宽度的死区波形



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 639 页上的第 22.4.18 节：TIM1/TIM8 断路和死区寄存器 (TIMx_BDTR)*。

将 OCxREF 重定向到 OCx 或 OCxN

在输出模式（强制输出模式、输出比较模式或 PWM 模式）下，通过配置 TIMx_CCER 寄存器中的 CCxE 和 CCxNE 位，可将 OCxREF 重定向到 OCx 输出或 OCxN 输出。

通过此功能，可以在一个输出上发送特定波形（如 PWM 或静态有效电平），而同时使互补输出保持其无效电平。或者，使两个输出同时保持无效电平，或者两个输出同时处于有效电平，两者互补并且带死区。

注： *如果仅使能 OCxN (CCxE=0, CCxNE=1)，两者不互补，一旦 OCxREF 为高电平，OCxN 即变为有效。例如，如果 CCxNP=0，则 OCxN=OCxRef。另一方面，如果同时使能 OCx 和 OCxN (CCxE=CCxNE=1)，OCx 在 OCxREF 为高电平时变为有效，而 OCxN 则与之互补，在 OCxREF 为低电平时变为有效。*

22.3.16 使用断路功能

断路功能的目的是保护由 TIM1 和 TIM8 定时器产生的 PWM 信号所驱动的电源开关。两个断路输入通常连接到功率级和三相逆变器的故障输出。激活时，断路电路会关闭 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。

有两个断路通道。一个断路通道收集系统级故障（时钟失效和奇偶校验错误等）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）。断路 2 通道只包括应用故障，能够将输出强制为无效状态。

断路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，允许通过软件使能/禁止输出，在发生断路和断路 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，还是释放对 GPIO 控制器的控制（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 OCx 和 OCxN 输出同时设置为有效电平。更多详细信息，请参见*第 635 页上的表 126：具有断路功能的互补通道 OCx 和 OCxN 的输出控制位*。

退出复位状态后，断路功能处于禁止状态，MOE 位处于低电平。将 TIMx_BDTR 寄存器中的 BKE 位和 BKE2 位置 1，可使能断路功能。可通过配置同一寄存器中的 BKP 位和 BKP2 位来选择断路输入的极性。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 处于低电平时向其写入 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。

如果 BIRDIE 条件标记为真，则必须按如下所述替换此部分：

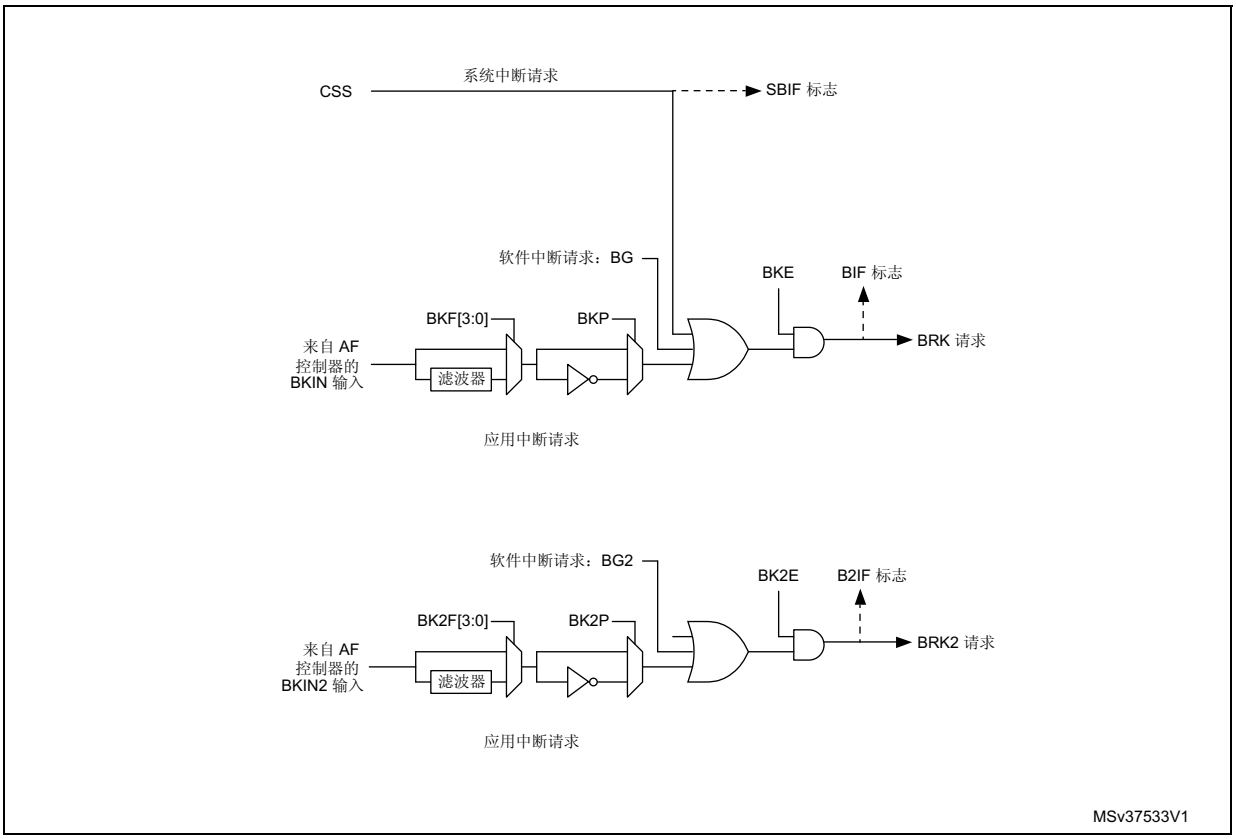
以下两个事件源经过或运算后可产生断路 (BRK) 事件：

- 连接到 BKIN 引脚的外部源（由 AFIO 控制器设定）
- 内部源：CSS 检测器产生时钟故障事件

连接到 BKIN2 引脚的外部源可产生断路 2 (BRK2)（由 AFIO 控制器设定）。

也可由软件通过 TIMx_EGR 寄存器中的 BG 和 B2G 位产生断路事件。

图 175. 断路和断路 2 电路概述



注： *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）来保证能够处理断路事件。*

发生断路之一（其中一个断路输入上出现所选电平）时：

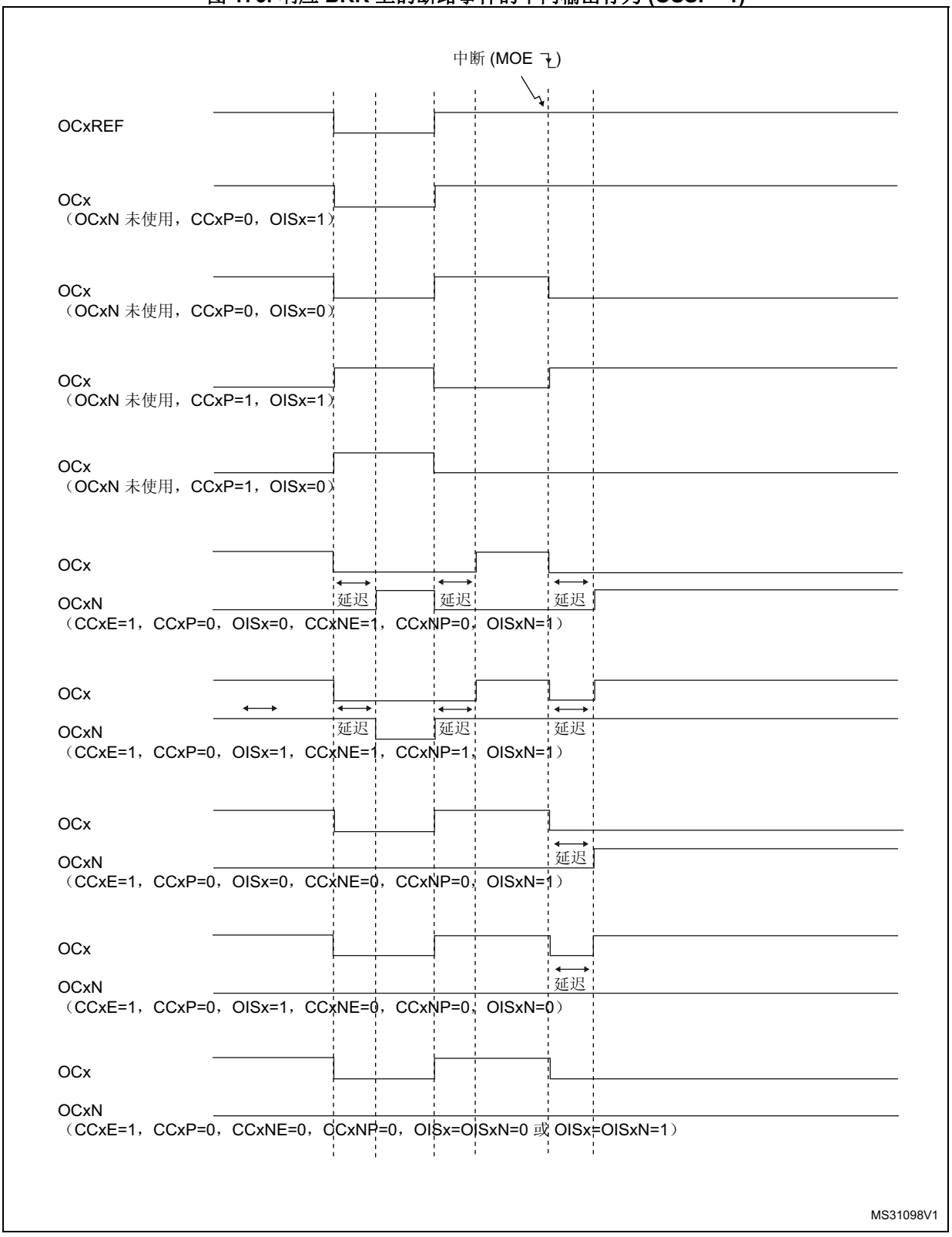
- MOE 位异步清零，使输出处于无效状态、空闲状态甚至释放对 GPIO 控制器的控制（通过 OSSI 位进行选择）。即使 MCU 振荡器关闭，该功能仍然使能。
- MOE=0 时，将以 TIMx_CR2 寄存器 OISx 位中编程的电平驱动每个输出通道。如果 OSSI=0，定时器将释放输出控制（由 GPIO 控制器接管），否则使能输出保持高电平。
- 使用互补输出时：
  - 输出首先置于无效状态（取决于极性）。这是异步操作，因此即使没有为定时器提供时钟，该操作仍有效。
  - 如果定时器时钟仍存在，则将重新激活死区发生器，进而在死区后以 OISx 和 OISxN 位中编程的电平驱动输出。即使在这种情况下，也不能同时将 OCx 和 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 ck_tim 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。
- 将断路状态标志（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。可以将断路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： *断路输入为电平有效。因此，当断路输入有效电平时，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。*

除断路输入和输出管理外，断路电路内部还实施了写保护，用以保护应用的安全。通过该功能，用户可冻结多个参数配置（死区持续时间、OCx/OCxN 极性和禁止时的状态、OCxM 配置、断路使能和极性）。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 639 页上的第 22.4.18 节：TIM1/TIM8 断路和死区寄存器 (TIMx_BDTR)*。MCU 复位后只能对 LOCK 位执行一次写操作。

图 176 所示为输出对断路响应行为的示例。

图 176. 响应 BRK 上的断路事件的不同输出行为 (OSSI = 1)



两个断路输入针对定时器输出具有不同的行为：

- 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。
- BRK2 只能禁止（无效状态）PWM 输出。

BRK 输入的优先级高于 BRK2 输入，如*表 123* 所示。

注： *BRK2 必须只在 OSSR = OSSI = 1 时使用。*

表 123. 定时器输出行为与 BRK/BRK2 输入

BRK	BRK2	定时器输出状态	典型用例	
			OCxN输出（下桥臂开关）	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 177 给出了 BRK 和 BRK2 输入上出现有效信号时 OCx 和 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 177. BRK 和 BRK2 引脚使能后的 PWM 输出状态 (OSSI=1)

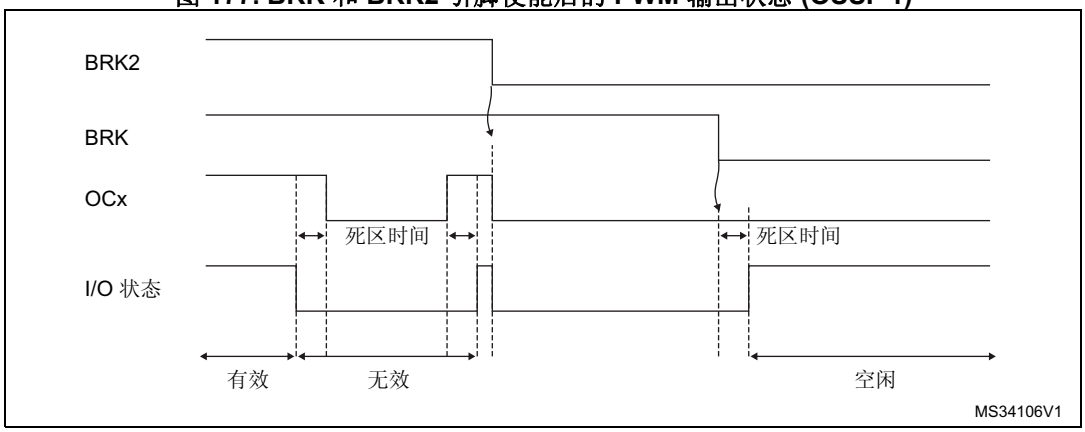
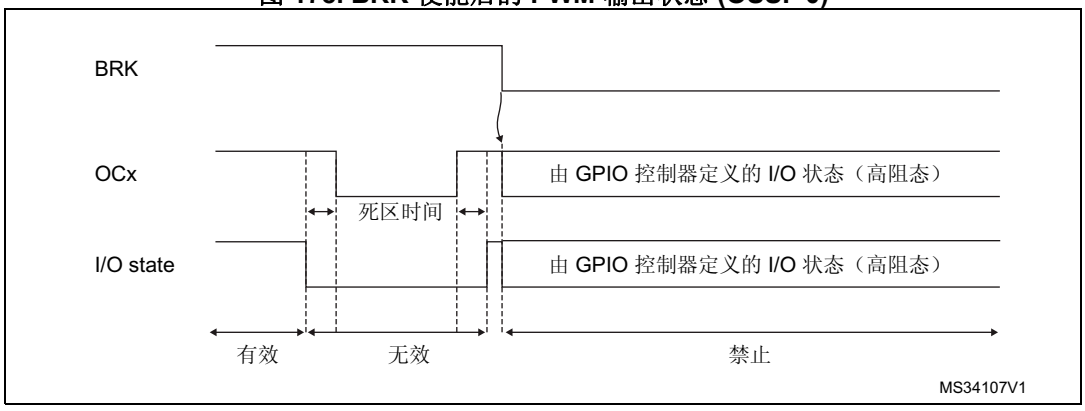


图 178. BRK 使能后的 PWM 输出状态 (OSSI=0)



22.3.17 发生外部事件时清除 OCxREF 信号

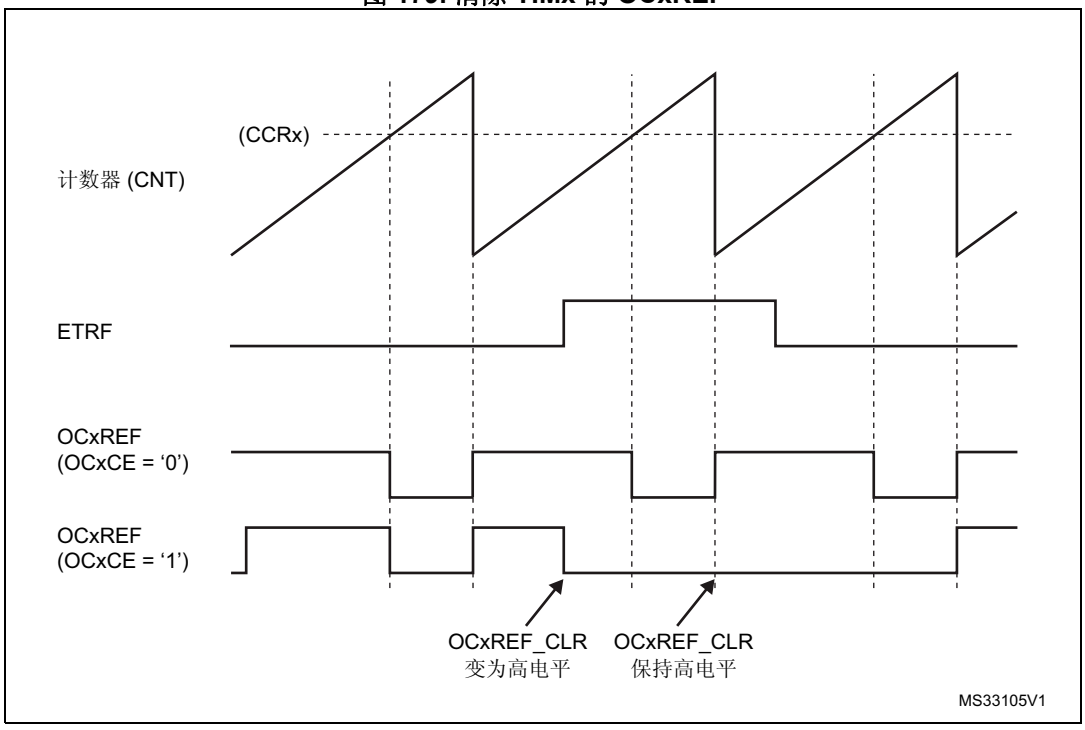
对于给定通道，在 ETRF 输入施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置“1”），可将 OCxREF 信号驱动为低电平。OCxREF 信号将保持低电平，直到发生下一更新事件 UEV。

选择 ETRF 时，ETR 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) 可根据用户需要进行配置。

图 179 对比了使能位 OCxCE 在不同值下的情况，显示了当 ETRF 输入变为高电平时 OCxREF 信号的行为。在本例中，定时器 TIMx 编程为 PWM 模式。

图 179. 清除 TIMx 的 OCxREF



注： *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 OCxREF。*

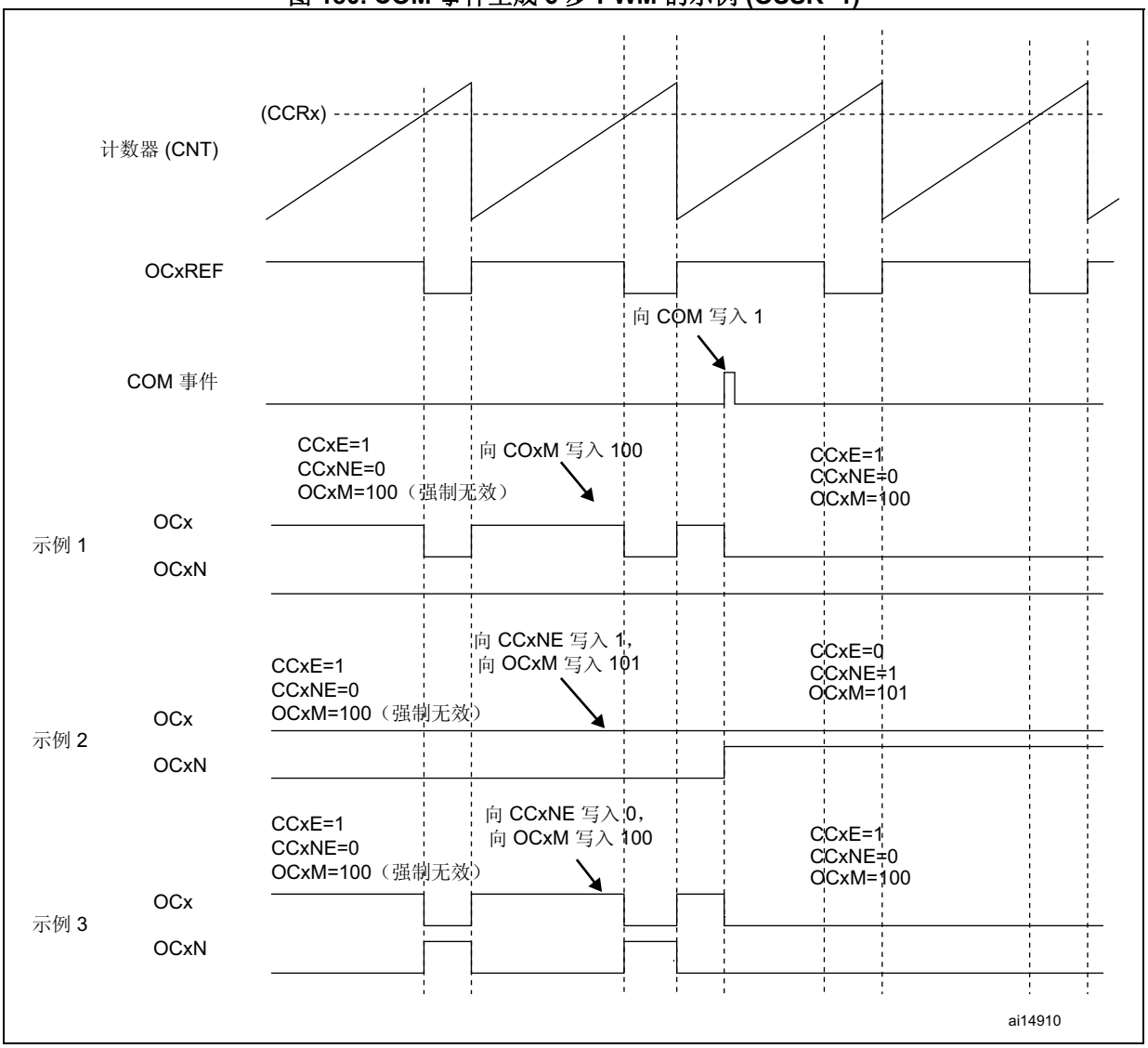
22.3.18 生成 6 步 PWM

当通道使用互补输出时，OCxM、CCxE 和 CCxNE 位上提供预装载位。发生 COM 换向事件时，这些预装载位将传输到影子位。因此，用户可以预先编程下一步骤的配置，并同时更改所有通道的配置。COM 可由软件通过将 TIMx_EGR 寄存器中的 COM 位置 1 而生成，也可以由硬件在 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 180 以 3 种不同的编程配置为例，显示了发生 COM 事件时 OCx 和 OCxN 输出的行为。

图 180. COM 事件生成 6 步 PWM 的示例 (OSSR=1)



22.3.19 单脉冲模式

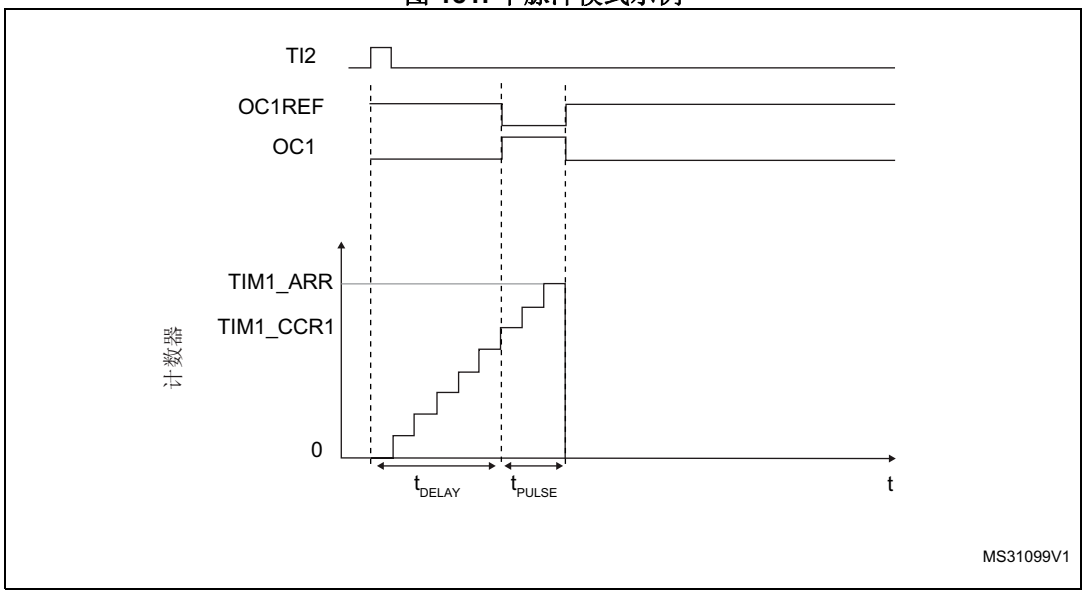
单脉冲模式 (OPM) 是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。将 TIMx_CR1 寄存器中的 OPM 位置 1，即可选择单脉冲模式。这样，发生下一更新事件 UEV 时，计数器将自动停止。

只有当比较值与计数器初始值不同时，才能正确产生一个脉冲。启动前（定时器等待触发时），必须进行如下配置：

- 递增计数时：CNT < CCRx ≤ ARR（特别注意，0 < CCRx）
- 递减计数时：CNT > CCRx

图 181. 单脉冲模式示例



例如，用户希望达到这样的效果：在 TI2 输入引脚检测到上升沿时，经过 t_{DELAY} 的延迟，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 TI2FP2 作为触发 1：

- 在 TIMx_CCMR1 寄存器中写入 CC2S=“01”，以将 TI2FP2 映射到 TI2。
- 在 TIMx_CCER 寄存器中写入 CC2P=“0”和 CC2NP=“0”，使 TI2FP2 能够检测上升沿。
- 在 TIMx_SMCR 寄存器中写入 TS=“110”，以将 TI2FP2 配置为从模式控制器的触发 (TRGI)。
- 在 TIMx_SMCR 寄存器中写入 SMS=“110”（触发模式），以使用 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，应在 TIMx_CCMR1 寄存器中写入 OC1M=111，以使能 PWM 模式 2。如果需要，可选择在 TIMx_CCMR1 寄存器的 OC1PE 和 TIMx_CR1 寄存器的 ARPE 中写入“1”，以使能预装载寄存器。这种情况下，必须在 TIMx_CCR1 寄存器中写入比较值并在 TIMx_ARR 寄存器中写入自动重载值，通过将 UG 位置 1 来产生更新，然后等待 TI2 上的外部触发事件。本例中，CC1P 的值为“0”。

在本例中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应为低。

由于仅需要 1 个脉冲（单脉冲模式），因此应向 TIMx_CR1 寄存器的 OPM 位写入“1”，以便在发生下一更新事件（计数器从自动重载值返回到 0）时使计数器停止计数。TIMx_CR1 寄存器中的 OPM 位置“0”时，即选择重复模式。

特殊情况：OCx 快速使能：

在单脉冲模式下，TIx 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时，将切换输出。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟（t_{DELAY} 最小值）。

如果要输出延迟时间最短的波形，可以将 TIMx_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 OCxRef（和 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。

22.3.20 可再触发单脉冲模式 (OPM)

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 22.3.19 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

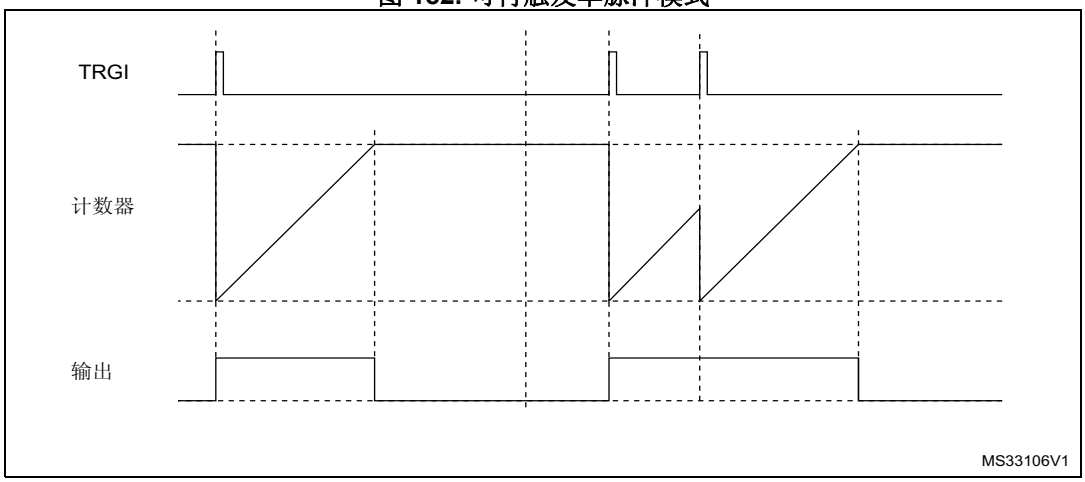
定时器必须处于从模式，TIMx_SMCR 寄存器中的位 SMS[3:0] = “1000”（组合复位 + 触发模式），针对可再触发 OPM 模式 1 或模式 2 将 OCxM[3:0] 位设置为“1000”或“1001”。

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）。定时器配置为递减计数模式时，ARR 必须置 0（CCRx 寄存器设置脉冲长度）。

注： *出于兼容性原因，OCxM[3:0] 和 SMS[3:0] 位字段分为两部分，最高有效位与最低有效的 3 位不相邻。*

在可再触发单脉冲模式下，CCxIF 标志没有意义。

图 182. 可再触发单脉冲模式



22.3.21 编码器接口模式

选择编码器接口模式时，如果计数器仅在 TI2 边沿处计数，在 TIMx_SMCR 寄存器中写入 SMS=“001”；如果计数器仅在 TI1 边沿处计数，写入 SMS=“010”；如果计数器在 TI1 和 TI2 边沿处均计数，则写入 SMS=“011”。

通过编程 TIMx_CCER 寄存器的 CC1P 和 CC2P 位，选择 TI1 和 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。

TI1 和 TI2 两个输入用于连接正交编码器。请参见*表 124*。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 TI1FP1 或 TI2FP2 上的每次有效信号转换提供。TI1FP1 和 TI2FP2 是进行输入滤波器和极性选择后 TI1 和 TI2 的信号，如果不进行滤波和反相，则 TI1FP1=TI1，TI2FP2=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（TI1 或 TI2）发生信号转换时，都会计算 DIR 位，无论计数器是仅在 TI1 或 TI2 边沿处计数，还是同时在 TI1 和 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 TI1 和 TI2 不同时切换）。

表 124. 计数方向与编码器信号的关系

有效边沿	相反信号的电平（TI1FP1 对应 TI2，TI2FP2 对应 TI1）	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 处计数	高	递减	递增	不计数	不计数
	低	递增	递减	不计数	不计数
仅在 TI2 处计数	高	不计数	不计数	递增	递减
	低	不计数	不计数	递减	递增
在 TI1 和 TI2 处均计数	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中断输入相连，用以触发计数器复位。

图 183 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，TI1FP1 映射到 TI1 上）。
- CC2S=“01”（TIMx_CCMR2 寄存器，TI1FP2 映射到 TI2 上）。
- CC1P=“0”，CC1NP=“0”（TIMx_CCER 寄存器，TI1FP1 未反相，TI1FP1=TI1）。
- CC2P=“0”，CC2NP=“0”（TIMx_CCER 寄存器，TI1FP2 未反相，TI1FP2= TI2）。
- SMS=“011”（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 183. 编码器接口模式下的计数器工作示例

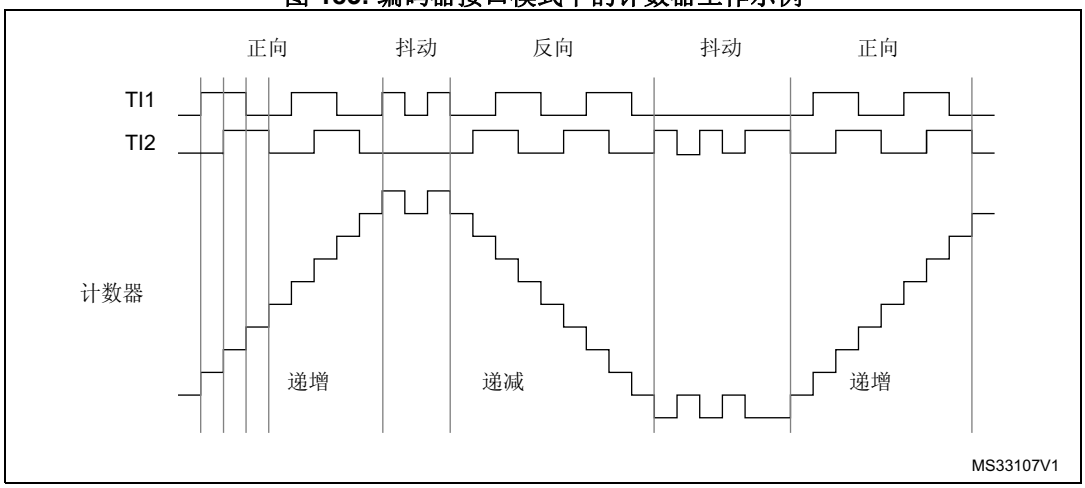
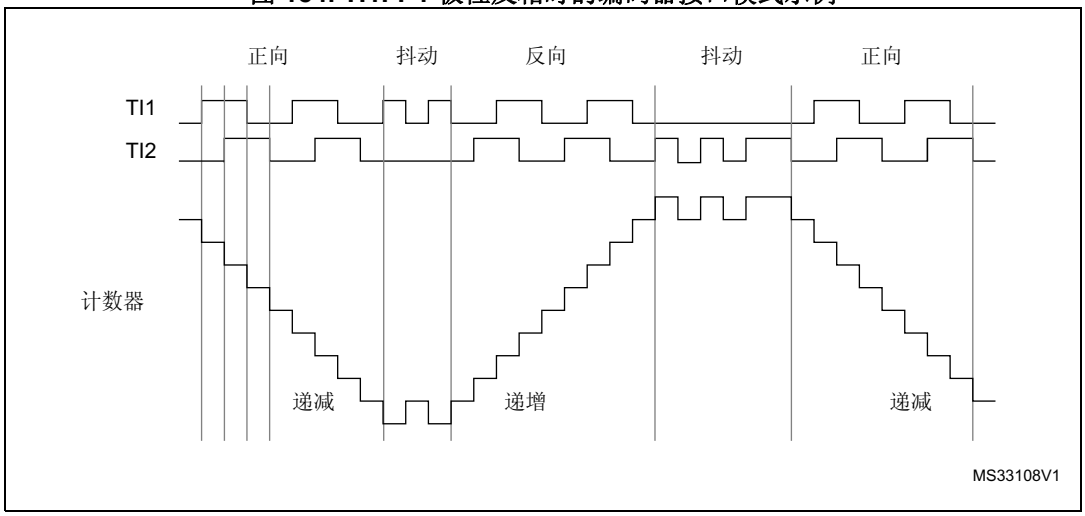


图 184 举例说明 TI1FP1 极性反相时计数器的行为（除 CC1P=“1” 外，其它配置与上例相同）。

图 184. TI1FP1 极性反相时的编码器接口模式示例



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 标志使能之间没有延迟。

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）。

22.3.22 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。

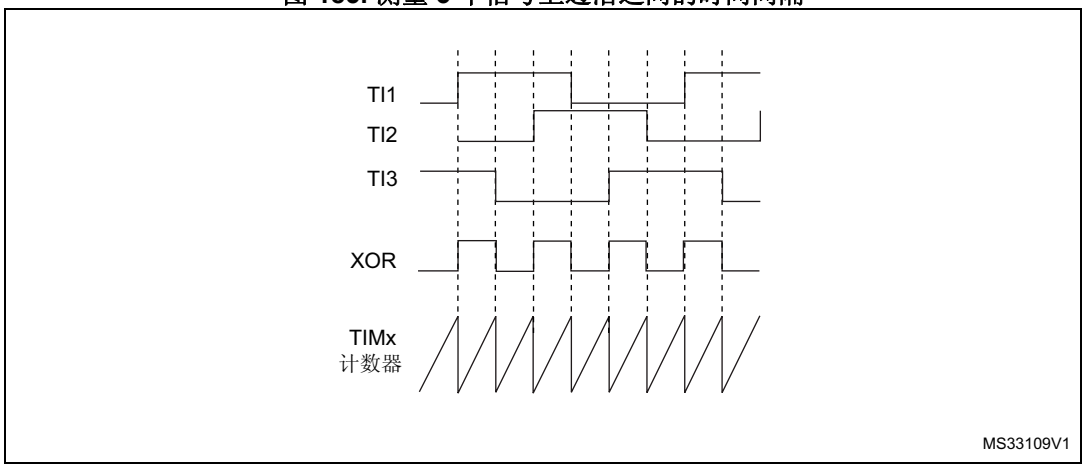
UIF 和 UIFCPY 标志使能之间没有延迟。

22.3.23 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 TIMx_CH1 和 TIMx_CH2 这两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 185* 所示）。

图 185. 测量 3 个信号上边沿之间的时间间隔



22.3.24 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（TIM1 或 TIM8）以及*图 186* 中称为“接口定时器”的另一个定时器 TIMx（TIM2、TIM3 和 TIM4），实现与霍尔传感器的连接。3 个定时器输入引脚（CC1、CC2 和 CC3）通过异或门连接到 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 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 TRC（请参见*第 586 页上的图 160：捕获/比较通道（例如：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（TIM1 或 TIM8）各个通道的配置。TIM1 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 TRGO 输出发送到高级控制定时器（TIM1 或 TIM8）。

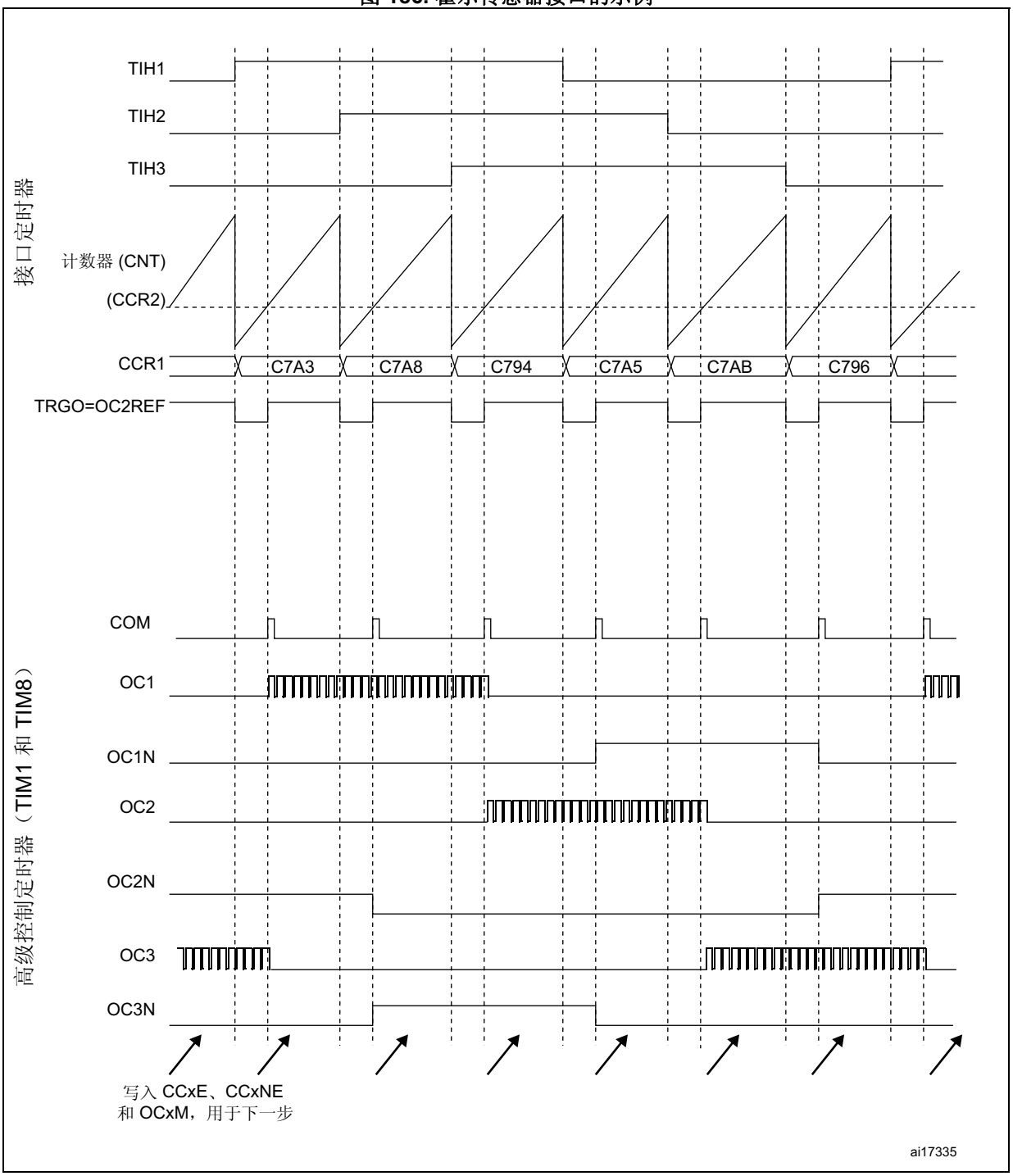
示例：霍尔输入与一个 TIMx 定时器相连接，每当霍尔输入发生更改，需要在所编程的延迟过后更改高级控制定时器 TIM1 的 PWM 配置。

- 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 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 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 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 OC2REF 作为 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 TIM1 中，必须选择正确的 ITR 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 OC2REF 上升沿产生的中断子程序中完成。

图 186 为本示例的示意图。

图 186. 霍尔传感器接口的示例



22.3.25 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。它们可在以下几种模式下实现同步：复位模式、门控模式和触发模式。

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。

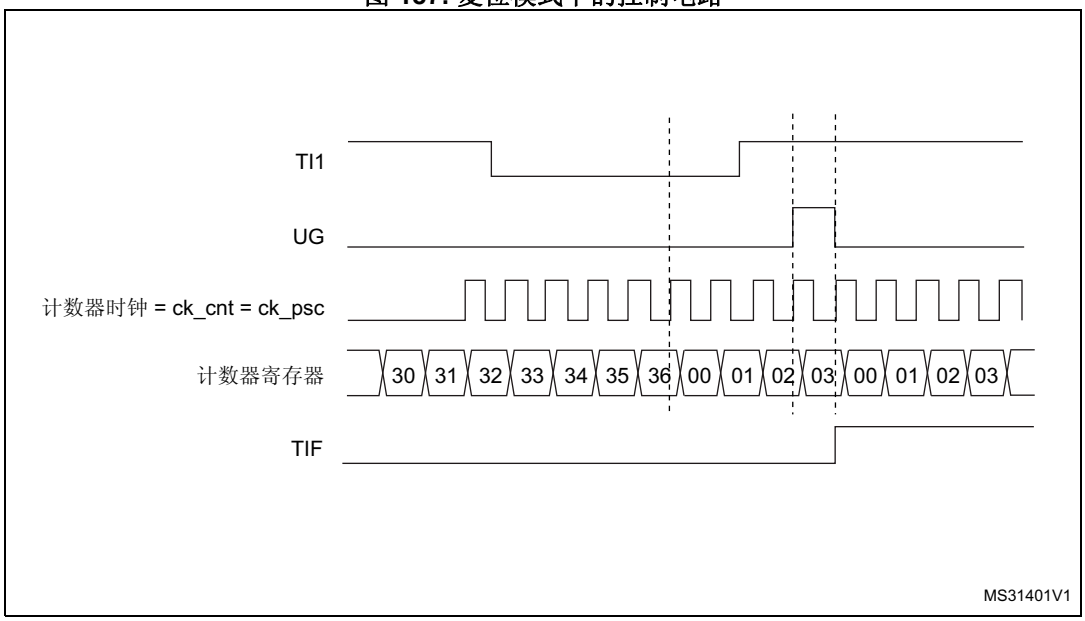
在以下示例中，TI1 输入上出现上升沿时，递增计数器清零：

- 将通道 1 配置为检测 TI1 的上升沿。配置输入滤波带宽（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=101，选择 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 TI1 上升沿。当 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。TI1 的上升沿与实际计数器复位之间的延迟是由于 TI1 输入的重新同步电路引起的。

图 187. 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。

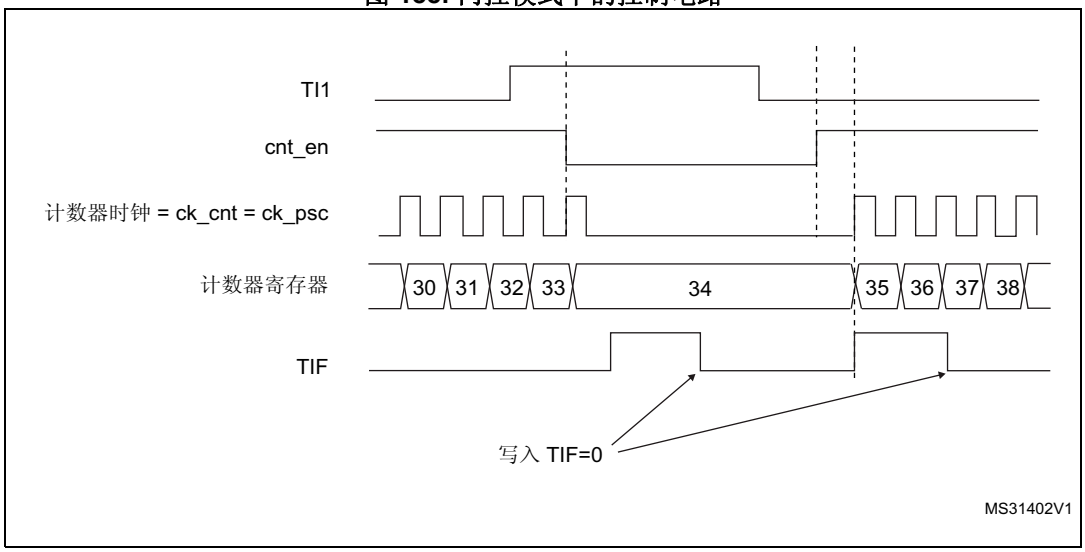
在以下示例中，递增计数器仅在 TI1 输入为低电平时计数：

- 将通道 1 配置为检测 TI1 上的低电平。配置输入滤波带宽（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=101，选择 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 TI1 为低电平，计数器就开始根据内部时钟计数，直到 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。

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

图 188. 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。

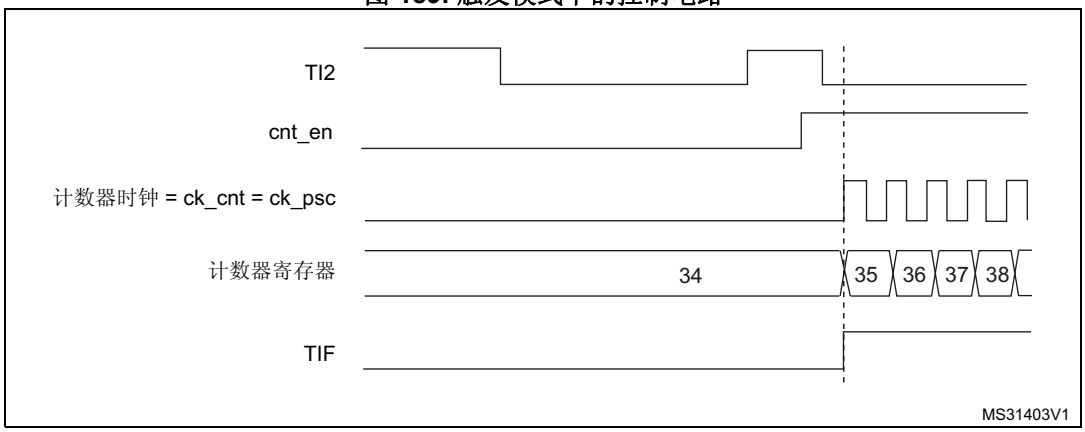
在以下示例中，TI2 输入上出现上升沿时，递增计数器启动：

- 将通道 2 配置为检测 TI2 上的上升沿。配置输入滤波带宽（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=110，选择 TI2 作为输入源。

当 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。

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

图 189. 触发模式下的控制电路



从模式：组合复位 + 触发模式

在这种情况下，在出现所选触发输入 (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：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，ETR 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 ETR 作为 TRGI。

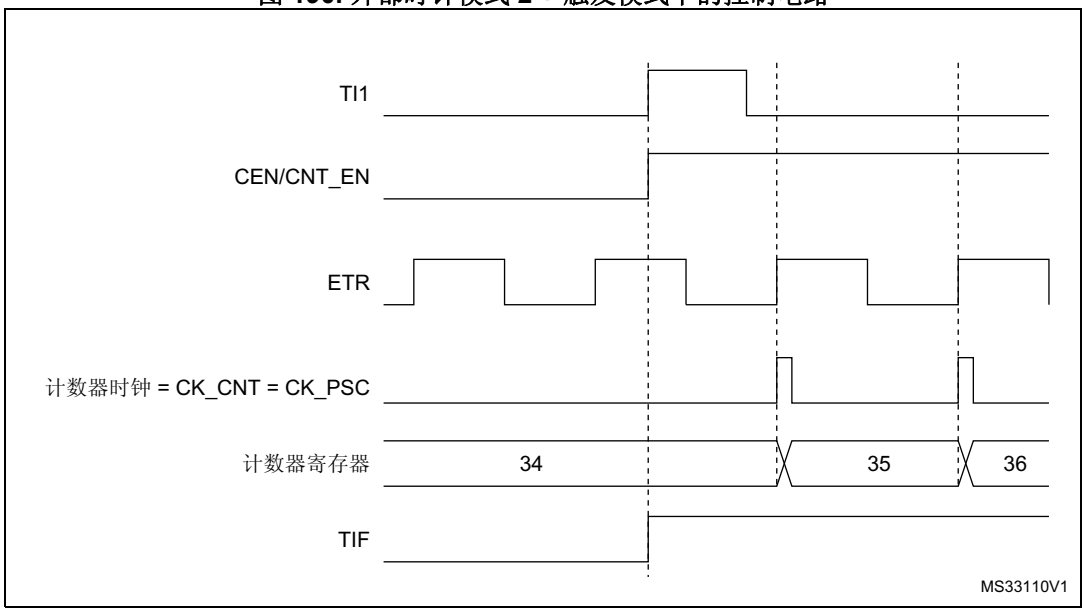
在以下示例中，只要 TI1 出现上升沿，递增计数器即会在 ETR 信号的每个上升沿处递增：

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 ETR 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=101，选择 TI1 作为输入源。

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 ETR 出现上升沿时计数。

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

图 190. 外部时钟模式 2 + 触发模式下的控制电路



注： *从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。*

22.3.26 ADC 同步

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 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 597 页上的图 171 给出了三相电机驱动的应用示例。

注： *从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。*

注： *从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。*

22.3.27 DMA 连续传送模式

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，但也可用于定期读取一行中的多个寄存器。

DMA 控制器目标唯一，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）。每次写入 TIMx_DMAR 寄存器都会重定向到其中一个定时器寄存器。

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，定时器进行一次连续传送，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

例如，定时器 DMA 连续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 位字段配置如下来配置 DCR 寄存器：
   DBL = 3 次传输，DBA = 0xE。
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，则要传输的数据量应为 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。

22.3.28 调试模式

当微控制器进入调试模式（Cortex®-M7 内核停止）时，TIMx 计数器会根据 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。有关详细信息，请参见*第 40.16.2 节：对定时器、看门狗、bxCAN 和 I^2C 的调试支持*。

22.4 TIM1/TIM8 寄存器

有关寄存器说明中使用的缩写，请参见相应列表。

22.4.1 TIM1/TIM8 控制寄存器 1 (TIMx_CR1)

TIM1/TIM8 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.	Res.	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

位 15:12 保留，必须保持复位值。

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

此位字段指示定时器时钟 (CK_INT) 频率与死区发生器以及数字滤波器（ETR、TIx）所使用的死区及采样时钟 (t_{DTS}) 之间的分频比，

00：$t_{DTS}=t_{CK_INT}$
01：$t_{DTS}=2*t_{CK_INT}$
10：$t_{DTS}=4*t_{CK_INT}$
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 CxS=00）的输出比较中断标志才置 1。
10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志才置 1。
11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志都会置 1。

注： 只要计数器处于使能状态 (CEN=1)，就不得从边沿对齐模式切换为中心对齐模式。

位 4 **DIR**：方向 (Direction)

0：计数器递增计数
1：计数器递减计数

注： 当定时器配置为中心对齐模式或编码器模式时，该位为只读状态。

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)

此位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能时，所有以下事件都会产生更新中断或 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/下溢会产生更新中断或 DMA 请求。

位 1 **UDIS**：更新禁止 (Update disable)

此位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新 (UEV) 事件可通过以下事件之一生成：

– 计数器上溢/下溢

– 将 UG 位置 1

– 通过从模式控制器生成的更新事件

然后更新影子寄存器的值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

位 0 **CEN**：计数器使能 (Counter enable)

0：禁止计数器

1：使能计数器

注： 只有事先通过软件将 CEN 位置 1，才可以使用外部时钟、门控模式和编码器模式。而触发模式可通过硬件自动将 CEN 位置 1。

22.4.2 TIM1/TIM8 控制寄存器 2 (TIMx_CR2)

TIM1/TIM8 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.	MMS2[3:0]				Res.	OIS6	Res.	OIS5
								rw	rw	rw	rw		rw		rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Res.	CCPC
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw

位 31:24 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 (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 (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 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 (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或运算组合而成。当计数器使能信号由触发输入控制时，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 (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（即使已为高电平），只要发生捕获或比较匹配，触发输出 (TRGO2) 都会发送一个正脉冲。
0100：**比较**——OC1REF 信号用作触发输出 (TRGO2)
0101：**比较**——OC2REF 信号用作触发输出 (TRGO2)
0110：**比较**——OC3REF 信号用作触发输出 (TRGO2)
0111：**比较**——OC4REF 信号用作触发输出 (TRGO2)
1000：**比较**——OC5REF 信号用作触发输出 (TRGO2)
1001：**比较**——OC6REF 信号用作触发输出 (TRGO2)
1010：**比较脉冲**——OC4REF 上升沿或下降沿时，TRGO2 上生成脉冲
1011：**比较脉冲**——OC6REF 上升沿或下降沿时，TRGO2 上生成脉冲
1100：**比较脉冲**——OC4REF 或 OC6REF 上升沿时，TRGO2 上生成脉冲
1101：**比较脉冲**——OC4REF 上升沿或 OC6REF 下降沿时，TRGO2 上生成脉冲
1110：**比较脉冲**——OC5REF 或 OC6REF 上升沿时，TRGO2 上生成脉冲
1111：**比较脉冲**——OC5REF 上升沿或 OC6REF 下降沿时，TRGO2 上生成脉冲

注：　从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。

位 19 保留，必须保持复位值。

位 18 **OIS6**：输出空闲状态 6（OC6 输出）(Output Idle state 6 (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（OC5 输出）(Output Idle state 5 (OC5 output))
请参见 OIS1 位

位 15 保留，必须保持复位值。

位 14 **OIS4**：输出空闲状态 4（OC4 输出）(Output Idle state 4 (OC4 output))
请参见 OIS1 位

位 13 **OIS3N**：输出空闲状态 3（OC3N 输出）(Output Idle state 3 (OC3N output))
请参见 OIS1N 位

位 12 **OIS3**：输出空闲状态 3（OC3 输出）(Output Idle state 3 (OC3 output))
请参见 OIS1 位

位 11 **OIS2N**：输出空闲状态 2（OC2N 输出）(Output Idle state 2 (OC2N output))
请参见 OIS1N 位

位 10 **OIS2**：输出空闲状态 2（OC2 输出）(Output Idle state 2 (OC2 output))
请参见 OIS1 位

位 9 **OIS1N**：输出空闲状态 1（OC1N 输出）(Output Idle state 1 (OC1N output))
0：当 MOE=0 时，经过死区时间后 OC1N=0
1：当 MOE=0 时，经过死区时间后 OC1N=1
注：　只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，此位即无法修改。

位 8 **OIS1**：输出空闲状态 1（OC1 输出）(Output Idle state 1 (OC1 output))
0：当 MOE=0 时，（如果 OC1N 有效，则经过死区时间之后）OC1=0
1：当 MOE=1 时，（如果 OC1N 有效，则经过死区时间之后）OC1=0
注：　只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，此位即无法修改。

位 7 **TI1S**：TI1 选择 (TI1 selection)
0：TIMx_CH1 引脚连接到 TI1 输入
1：TIMx_CH1、CH2 和 CH3 引脚连接到 TI1 输入（异或组合）

位 6:4 **MMS[1:0]**：主模式选择 (Master mode selection)
这些位可选择主模式下将要发送到从定时器以实现同步的信息 (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 (TRGO)。如果复位由触发输入生成（从模式控制器配置为复位模式），则 TRGO 上的信号相比实际复位会有延迟。
001：**使能**——计数器使能信号 CNT_EN 用作触发输出 (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或运算组合而成。当计数器使能信号由触发输入控制时，TRGO 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中 MSM 位的说明）。
010：**更新**——选择更新事件作为触发输出 (TRGO)。例如，主定时器可用作从定时器的预分频器。
011：**比较脉冲**——一旦发生输入捕获或比较匹配事件，当 CC1IF 标志被置 1 时（即使已为高电平），触发输出都会发送一个正脉冲。(TRGO)。
100：**比较**——OC1REF 信号用作触发输出 (TRGO)
101：**比较**——OC2REF 信号用作触发输出 (TRGO)
110：**比较**——OC3REF 信号用作触发输出 (TRGO)
111：**比较**——OC4REF 信号用作触发输出 (TRGO)
注：　从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。

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/比较控制位进行预装载 (CCPC=1)，仅通过将 COMG 位置 1 来对这些位进行更新
1：如果捕获/比较控制位进行预装载 (CCPC=1)，可通过将 COMG 位置 1 或 TRGI 的上升沿对这些位进行更新。
注：　此位仅对具有互补输出的通道有效。

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 TRGI 上检测到上升沿，取决于 CCUS 位）时才会对这些位进行更新。
注：　此位仅对具有互补输出的通道有效。

22.4.3 TIM1/TIM8 从模式控制寄存器 (TIMx_SMCR)

TIM1/TIM8 slave mode 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.	SMS[3]
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			Res	SMS[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw

位 31:17 保留，必须保持复位值。

位 16 **SMS[3]**：从模式选择——位 3 (Slave mode selection - bit 3)
请参见 SMS 说明——位 2:0

位 15 **ETP**：外部触发极性 (External trigger polarity)
此位可选择将 ETR 还是 $\overline{ETR}$ 用于触发操作
0：ETR 未反相，高电平或上升沿有效。
1：ETR 反相，低电平或下降沿有效。

位 14 **ECE**：外部时钟使能 (External clock enable)
此位可使能外部时钟模式 2。
0：禁止外部时钟模式 2
1：使能外部时钟模式 2。计数器时钟由 ETRF 信号的任意有效边沿提供。

注： ***1：****将 ECE 位置 1 与选择外部时钟模式 1 并将 TRGI 连接到 ETRF（SMS=111 且 TS=111）具有相同效果。*

2：*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 TRGI 不得连接 ETRF（TS 位不得为 111）。*

3：*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 ETRF。*

位 13:12 **ETPS[1:0]**：外部触发预分频器 (External trigger prescaler)
外部触发信号 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 ETRP 频率。这种方法在输入快速外部时钟时非常有用。
00：预分频器关闭
01：2 分频 ETRP 频率
10：4 分频 ETRP 频率
11：8 分频 ETRP 频率

位 11:8 **ETF[3:0]**：外部触发滤波器 (External trigger filter)

此位字段可定义 ETRP 信号的采样频率和适用于 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：$f_{SAMPLING}=f_{CK_INT}$，N=2
0010：$f_{SAMPLING}=f_{CK_INT}$，N=4
0011：$f_{SAMPLING}=f_{CK_INT}$，N=8
0100：$f_{SAMPLING}=f_{DTS}/2$，N=6
0101：$f_{SAMPLING}=f_{DTS}/2$，N=8
0110：$f_{SAMPLING}=f_{DTS}/4$，N=6
0111：$f_{SAMPLING}=f_{DTS}/4$，N=8
1000：$f_{SAMPLING}=f_{DTS}/8$，N=6
1001：$f_{SAMPLING}=f_{DTS}/8$，N=8
1010：$f_{SAMPLING}=f_{DTS}/16$，N=5
1011：$f_{SAMPLING}=f_{DTS}/16$，N=6
1100：$f_{SAMPLING}=f_{DTS}/16$，N=8
1101：$f_{SAMPLING}=f_{DTS}/32$，N=5
1110：$f_{SAMPLING}=f_{DTS}/32$，N=6
1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 7 **MSM**：主/从模式 (Master/slave mode)

0：不执行任何操作
1：当前定时器的触发输入事件 (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 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)

此位字段可选择将要用于同步计数器的触发输入。

000：内部触发 0 (ITR0)
001：内部触发 1 (ITR1)
010：内部触发 2 (ITR2)
011：内部触发 3 (ITR3)
100：TI1 边沿检测器 (TI1F_ED)
101：滤波后的定时器输入 1 (TI1FP1)
110：滤波后的定时器输入 2 (TI2FP2)
111：外部触发输入 (ETRF)

有关各定时器 ITRx 含义的详细信息，请参见*第 623 页上的表 125：TIMx 内部触发连接*。

注：　这些位只能在未使用的情况下（例如，SMS=000 时）进行更改，以避免转换时出现错误的边沿检测。

位 3　保留，必须保持复位值。

位 2:0 **SMS**：从模式选择 (Slave mode selection)

选择外部信号时，触发信号 (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 TI1FP2 电平在 TI2FP1 边沿递增/递减计数。
0010：编码器模式 2——计数器根据 TI2FP1 电平在 TI1FP2 边沿递增/递减计数。
0011：编码器模式 3——计数器在 TI1FP1 和 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 (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 (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 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 (TRGI) 的上升沿提供计数器时钟。
1000：组合复位 + 触发模式——在出现所选触发输入 (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。
1000 以上的代码：保留。

注：　如果将 TI1F_ED 选作触发输入（TS=“100”），则不得使用门控模式。实际上，TI1F 每次转换时，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：　从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。

表 125. TIMx 内部触发连接

从 TIM	ITR0 (TS = 000)	ITR1 (TS = 001)	ITR2 (TS = 010)	ITR3 (TS = 011)
TIM1	TIM5	TIM2	TIM3	TIM4
TIM8	TIM1	TIM2	TIM4	TIM5

22.4.4 TIM1/TIM8 DMA/中断使能寄存器 (TIMx_DIER)

TIM1/TIM8 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.	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 15 保留，必须保持复位值。

位 14 **TDE**：触发 DMA 请求使能 (Trigger DMA request enable)
0：禁止触发 DMA 请求
1：使能触发 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**：断路中断使能 (Break interrupt enable)
0：禁止断路中断
1：使能断路中断

位 6 **TIE**：触发中断使能 (Trigger interrupt enable)
0：禁止触发中断
1：使能触发中断

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

22.4.5 TIM1/TIM8 状态寄存器 (TIMx_SR)

TIM1/TIM8 status 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.	C6IF	C5IF
														rc_w0	rc_w0

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

位 31:18 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： 保留，必须保持复位值。

位 12 **CC4OF**：捕获/比较 4 重复捕获标志 (Capture/Compare 4 overcapture flag)
请参见 CC1OF 说明

位 11 **CC3OF**：捕获/比较 3 重复捕获标志 (Capture/Compare 3 overcapture flag)
请参见 CC1OF 说明

位 10 **CC2OF**：捕获/比较 2 重复捕获标志 (Capture/compare 2 overcapture flag)
请参见 CC1OF 说明

位 9 **CC1OF**：捕获/比较 1 重复捕获标志 (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：未检测到重复捕获。
1：TIMx_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。

位 8 **B2IF**：断路 2 中断标志 (Break 2 interrupt flag)
只要断路 2 输入变为有效状态，此标志便由硬件置 1。断路 2 输入无效后可通过软件对其清零。
0：未发生断路事件。
1：在断路 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**：断路中断标志 (Break interrupt flag)
只要断路输入变为有效状态，此标志便由硬件置 1。断路输入无效后可通过软件对其清零。
0：未发生断路事件。
1：在断路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 TRGI 输入上检测到有效边沿时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动或停止时置 1。但需要通过软件清零。
0：未发生触发事件。
1：触发中断挂起。

位 5 **COMIF**：COM 中断标志 (COM interrupt flag)
此标志在发生 COM 事件时（捕获/比较控制位 CCxE、CCxNE 和 OCxM 已更新时）由硬件置 1。但需要通过软件清零。
0：未发生 COM 事件。
1：COM 中断挂起。

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)
如果通道 CC1 配置为输出：当计数器与比较值匹配时，此标志由硬件置 1，中心对齐模式下除外（请参见 TIMx_CR1 寄存器中的 CMS 位说明）。但需要通过软件清零。
0：不匹配。
1：TIMx_CNT 计数器的值与 TIMx_CCR1 寄存器的值匹配。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。
如果通道 CC1 配置为输入：此位将在发生捕获事件时由硬件置 1。通过软件或读取 TIMx_CCR1 寄存器将该位清零。
0：未发生输入捕获事件。
1：TIMx_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）。

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 22.4.3 节：TIM1/TIM8 从模式控制寄存器 (TIMx_SMCR)*）。

22.4.6 TIM1/TIM8 事件产生寄存器 (TIMx_EGR)

TIM1/TIM8 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.	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**：断路 2 生成 (Break 2 generation)
此位由软件置 1 以生成事件，并由硬件自动清零。
0：不执行任何操作
1：生成断路 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**：断路生成 (Break generation)
此位由软件置 1 以生成事件，并由硬件自动清零。
0：不执行任何操作
1：生成断路事件。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)
此位由软件置 1 以生成事件，并由硬件自动清零。
0：不执行任何操作
1：TIMx_SR 寄存器中的 TIF 标志置 1。使能后可发生相关中断或 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：不执行任何操作
1：CCPC 位置 1 时，可更新 CCxE、CCxNE 和 OCxM 位
注： 此位仅对具有互补输出的通道有效。

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)
此位由软件置 1 以生成事件，并由硬件自动清零。
0：不执行任何操作
1：通道 1 上生成捕获/比较事件：
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：不执行任何操作
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)。

22.4.7 TIM1/TIM8 捕获/比较模式寄存器 1 (TIMx_CCMR1)

TIM1/TIM8 capture/compare mode register 1

偏移地址：0x18

复位值：0x0000 0000

这些通道可用于输入（捕获模式）或输出（比较模式）模式。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其它位在输入模式和输出模式下的功能均不同。对于任一给定位，OCxx 用于说明通道配置为输出时该位对应的功能，ICxx 则用于说明通道配置为输入时该位对应的功能。因此，必须注意同一个位在输入阶段和输出阶段具有不同的含义。

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]
							Res.								Res.

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]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[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 24 **OC2M[3]**：输出比较 2 模式——位 3 (Output Compare 2 mode - bit 3)

位 23:17 保留，必须保持复位值。

位 16 **OC1M[3]**：输出比较 1 模式——位 3 (Output Compare 1 mode - bit 3)
请参见 OC1M 说明——位 6:4

位 15 **OC2CE**：输出比较 2 清零使能 (Output Compare 2 clear enable)

位 14:12 **OC2M[2:0]**：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)
此位字段定义通道方向（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，IC2 映射到 TI2 上
10：CC2 通道配置为输入，IC2 映射到 TI1 上
11：CC2 通道配置为输入，IC2 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注： 仅当通道关闭时（TIMx_CCER 中的 CC2E = “0”），才可向 CC2S 位写入数据。

位 7 **OC1CE**：输出比较 1 清零使能 (Output Compare 1 clear enable)
OC1CE：输出比较 1 清零使能 (Output Compare 1 Clear Enable)
0：OC1Ref 不受 ETRF 输入影响
1：ETRF 输入上检测到高电平时，OC1Ref 立即清零

位 6:4 **OC1M**：输出比较 1 模式 (Output Compare 1 mode)

这些位定义提供 OC1 和 OC1N 的输出参考信号 OC1REF 的行为。OC1REF 为高电平有效，而 OC1 和 OC1N 的有效电平则取决于 CC1P 位和 CC1NP 位。

0000：冻结——输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。

0001：将通道 1 设置为匹配时输出有效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1 (TIMx_CCR1) 匹配时，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，OC1REF 信号强制变为低电平。

0011：翻转——TIMx_CNT=TIMx_CCR1 时，OC1REF 发生翻转。

0100：强制变为无效电平——OC1REF 强制变为低电平。

0101：强制变为有效电平——OC1REF 强制变为高电平。

0110：PWM 模式 1——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为无效状态（OC1REF=“0”），否则为有效状态（OC1REF=“1”）。

0111：PWM 模式 2——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为无效状态，否则为有效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。

1000：可再触发 OPM 模式 1——在递增计数模式下，通道为有效状态，直至（在 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 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 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 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——OC1REF 与在 PWM 模式 1 下的行为相同。OC1REFC 是 OC1REF 和 OC2REF 的逻辑或运算结果。

1101：组合 PWM 模式 2——OC1REF 与在 PWM 模式 2 下的行为相同。OC1REFC 是 OC1REF 和 OC2REF 的逻辑与运算结果。

1110：不对称 PWM 模式 1——OC1REF 与在 PWM 模式 1 下的行为相同。计数器递增计数时，OC1REFC 输出 OC1REF；计数器递减计数时，OC1REFC 输出 OC2REF。

1111：不对称 PWM 模式 2——OC1REF 与在 PWM 模式 2 下的行为相同。计数器递增计数时，OC1REFC 输出 OC1REF；计数器递减计数时，OC1REFC 输出 OC2REF。

注：　只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 3 且 CC1S=“00”（通道配置为输出），这些位即无法修改。

注：　在 PWM 模式下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“PWM”模式时，OCREF 电平才会发生更改。

注：　此位字段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活动寄存器中。

*注：　**1：**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。*

*　　**2：**只有单脉冲模式下才可在未验证预装载寄存器的情况下使用 PWM 模式（TIMx_CR1 寄存器中的 OPM 位置 1）。其它情况下则无法保证该行为。*

位 2 **OC1FE**：输出比较 1 快速使能 (Output Compare 1 fast enable)

此位用于加快触发输入事件对 CC 输出的影响。

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**：捕获/比较 1 选择 (Capture/Compare 1 Selection)

此位字段定义通道方向（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，IC1 映射到 TI1 上

10：CC1 通道配置为输入，IC1 映射到 TI2 上

11：CC1 通道配置为输入，IC1 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注： 仅当通道关闭时（TIMx_CCER 中的 CC1E = “0”），才可向 CC1S 位写入数据。

输入捕获模式

位 31:16 保留，必须保持复位值。

位 15:12 **IC2F**：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**：捕获/比较 2 选择 (Capture/Compare 2 selection)

此位字段定义通道方向（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，IC2 映射到 TI2 上

10：CC2 通道配置为输入，IC2 映射到 TI1 上

11：CC2 通道配置为输入，IC2 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注： 仅当通道关闭时（TIMx_CCER 中的 CC2E = “0”），才可向 CC2S 位写入数据。

位 7:4 **IC1F[3:0]**：输入捕获 1 滤波器 (Input capture 1 filter)

此位字段可定义 TI1 输入的采样频率和适用于 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：$f_{SAMPLING}=f_{CK_INT}$，N=2

0010：$f_{SAMPLING}=f_{CK_INT}$，N=4

0011：$f_{SAMPLING}=f_{CK_INT}$，N=8

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8

1000：$f_{SAMPLING}=f_{DTS}/8$，N=6

1001：$f_{SAMPLING}=f_{DTS}/8$，N=8

1010：$f_{SAMPLING}=f_{DTS}/16$，N=5

1011：$f_{SAMPLING}=f_{DTS}/16$，N=6

1100：$f_{SAMPLING}=f_{DTS}/16$，N=8

1101：$f_{SAMPLING}=f_{DTS}/32$，N=5

1110：$f_{SAMPLING}=f_{DTS}/32$，N=6

1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 3:2 **IC1PSC**：输入捕获 1 预分频器 (Input capture 1 prescaler)

此位字段定义 CC1 输入 (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**：捕获/比较 1 选择 (Capture/Compare 1 Selection)

此位字段定义通道方向（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，IC1 映射到 TI1 上
10：CC1 通道配置为输入，IC1 映射到 TI2 上
11：CC1 通道配置为输入，IC1 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注： 仅当通道关闭时（TIMx_CCER 中的 CC1E = “0”），才可向 CC1S 位写入数据。

22.4.8 TIM1/TIM8 捕获/比较模式寄存器 2 (TIMx_CCMR2)

TIM1/TIM8 capture/compare mode register 2

偏移地址：0x1C

复位值：0x0000 0000

请参见上述 CCMR1 寄存器说明。

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]
							Res.								Res.
							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]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

位 31:25 保留，必须保持复位值。

位 24 **OC4M[3]**：输出比较 4 模式——位 3 (Output Compare 4 mode - bit 3)

位 23:17 保留，必须保持复位值。

位 16 **OC3M[3]**：输出比较 3 模式——位 3 (Output Compare 3 mode - bit 3)

位 15 OC4CE：输出比较 4 清零使能 (Output compare 4 clear enable)

位 14:12 **OC4M**：输出比较 4 模式 (Output compare 4 mode)

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**：捕获/比较 4 选择 (Capture/Compare 4 selection)
此位字段定义通道方向（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，IC4 映射到 TI4 上
10：CC4 通道配置为输入，IC4 映射到 TI3 上
11：CC4 通道配置为输入，IC4 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注： 仅当通道关闭时（TIMx_CCER 中的 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 6:4 **OC3M**：输出比较 3 模式 (Output compare 3 mode)

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**：捕获/比较 3 选择 (Capture/compare 3 selection)
此位字段定义通道方向（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，IC3 映射到 TI3 上
10：CC3 通道配置为输入，IC3 映射到 TI4 上
11：CC3 通道配置为输入，IC3 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注： 仅当通道关闭时（TIMx_CCER 中的 CC3E = "0"），才可向 CC3S 位写入数据。

输入捕获模式

位 31:16 保留，必须保持复位值。

位 15:12 **IC4F**：输入捕获 4 滤波器 (Input capture 4 filter)

位 11:10 **IC4PSC**：输入捕获 4 预分频器 (Input capture 4 prescaler)

位 9:8 **CC4S**：捕获/比较 4 选择 (Capture/Compare 4 selection)
此位字段定义通道方向（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，IC4 映射到 TI4 上
10：CC4 通道配置为输入，IC4 映射到 TI3 上
11：CC4 通道配置为输入，IC4 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注： 仅当通道关闭时（TIMx_CCER 中的 CC4E = "0"），才可向 CC4S 位写入数据。

位 7:4 **IC3F**：输入捕获 3 滤波器 (Input capture 3 filter)

位 3:2 **IC3PSC**：输入捕获 3 预分频器 (Input capture 3 prescaler)

位 1:0 **CC3S**：捕获/比较 3 选择 (Capture/compare 3 selection)
此位字段定义通道方向（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，IC3 映射到 TI3 上
10：CC3 通道配置为输入，IC3 映射到 TI4 上
11：CC3 通道配置为输入，IC3 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注： 仅当通道关闭时（TIMx_CCER 中的 CC3E = "0"），才可向 CC3S 位写入数据。

22.4.9 TIM1/TIM8 捕获/比较使能寄存器 (TIMx_CCER)

TIM1/TIM8 capture/compare 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.	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	Res.	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	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 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：OC1N 高电平有效。
1：OC1N 低电平有效。
CC1 通道配置为输入：
此位与 CC1P 配合使用，用以定义 TI1FP1 和 TI2FP1 的极性。请参见 CC1P 说明。

注： 只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 2 或 3 且 CC1S=“00”（通道配置为输出），此位立即变为不可写状态。

注： 此位将在具有互补输出的通道上进行预装载。如果 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——OC1N 未激活。OC1N 电平是 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的函数。
1：开启——在相应输出引脚上输出 OC1N 信号，具体取决于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位。

注： 此位将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1NE 有效位才会从预装载位获取新值。

位 1 **CC1P**：捕获/比较 1 输出极性 (Capture/Compare 1 output polarity)
CC1 通道配置为输出：
0：OC1 高电平有效
1：OC1 低电平有效

CC1 通道配置为输入：CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的有效极性。
00：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）。
01：反相/下降沿触发。电路对 TIxFP1 下升沿敏感（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 反相（在门控模式或编码器模式下执行触发操作）。
10：保留，不使用此配置。
11：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。

注： 只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 2 或 3，此位立即变为不可写状态。

注： 此位将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1P 有效位才会从预装载位获取新值。

位 0 **CC1E**：捕获/比较 1 输出使能 (Capture/Compare 1 output enable)

CC1 通道配置为输出：

0：关闭——OC1 未激活。OC1 电平是 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的函数。

1：开启——OC1 信号输出到相应的输出引脚上，具体取决于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位。

CC1 通道配置为输入：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 (TIMx_CCR1) 中。

0：禁止捕获。

1：使能捕获。

注：　此位将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1E 有效位才会从预装载位获取新值。

表 126. 具有断路功能的互补通道 OCx 和 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	OCx 输出状态	OCxN 输出状态
1	X	X	0	0	禁止输出（不由定时器驱动：高阻态） OCx=0、OCxN=0	
		0	0	1	禁止输出（不由定时器驱动：高阻态） OCx=0	OCxREF + 极性 OCxN = OCxREF 异或 CCxNP
		0	1	0	OCxREF + 极性 OCx=OCxREF 异或 CCxP	禁止输出（不由定时器驱动：高阻态） 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） OCx=CCxP	OCxREF + 极性 OCxN = OCxREF 异或 CCxNP
		1	1	0	OCxREF + 极性 OCx=OCxREF 异或 CCxP	关闭状态（输出使能为无效状态） OCxN=CCxNP
0	0	X	X	X	禁止输出（不由定时器驱动：高阻态） OCx=CCxP、OCxN=CCxNP	
	1		0	0		
			0	1	关闭状态（输出使能为无效状态） 异步：OCx=CCxP、OCxN=CCxNP（如果触发 BRK 或 BRK2）。 随后（仅当触发 BRK 时才有效），如果存在时钟：在死区后 OCx=OISx 且 OCxN=OISxN，假定 OISx 和 OISxN 并没有都设置成 OCx 及 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：**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：　与互补通道 OCx 和 OCxN 相连的外部 I/O 引脚的状态取决于通道 OCx 和 OCxN 的状态以及 GPIO 寄存器。

22.4.10 TIM1/TIM8 计数器 (TIMx_CNT)

TIM1/TIM8 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
UIFCPY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Re s.	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)

22.4.11 TIM1/TIM8 预分频器 (TIMx_PSC)

TIM1/TIM8 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 (CK_CNT) 等于 f_{CK_PSC} / (PSC[15:0] + 1)。
PSC 包含每次发生更新事件（包括计数器通过 TIMx_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）时要装载到活动预分频器寄存器的值。

22.4.12 TIM1/TIM8 自动重载寄存器 (TIMx_ARR)

TIM1/TIM8 auto-reload register

偏移地址：0x2C

复位值：0x0000

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 15:0 **ARR[15:0]**：预分频器值 (Prescaler 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 568 页上的第 22.3.1 节：时基单元*。
当自动重载值为空时，计数器不工作。

22.4.13 TIM1/TIM8 重复计数器寄存器 (TIMx_RCR)

TIM1/TIM8 repetition counter register

偏移地址：0x30

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]**：重复计数器值 (Repetition Counter value)

使能预装载寄存器时，用户可通过这些位设置比较寄存器的更新频率（即，从预装载寄存器向活动寄存器周期性传输数据）；使能更新中断时，也可设置更新中断的生成速率。

与 REP_CNT 相关的减计数器每次计数到 0 时，都将生成一个更新事件并且计数器从 REP 值重新开始计数。由于只有生成重复更新事件 U_RC 时，REP_CNT 才会重载 REP 值，因此在生成下一重复更新事件之前，无论向 TIMx_RCR 寄存器写入何值都无影响。

这意味着 PWM 模式下 (REP+1) 相当于：

边沿对齐模式下的 PWM 周期数

中心对齐模式下的 PWM 半周期数。

22.4.14 TIM1/TIM8 捕获/比较寄存器 1 (TIMx_CCR1)

TIM1/TIM8 capture/compare register 1

偏移地址：0x34

复位值：0x0000

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 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 为要装载到实际捕获/比较 1 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR1 寄存器中的 OC1PE 位来使能预装载功能，则该值立刻生效； 否则只在发生更新事件时生效（拷贝到实际起作用的捕获/比较寄存器 1）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC1 输出上发出信号的值。

如果通道 CC1 配置为输入：CR1 为上一个输入捕获 1 事件 (IC1) 发生时的计数器值。

22.4.15 TIM1/TIM8 捕获/比较寄存器 2 (TIMx_CCR2)

TIM1/TIM8 capture/compare register 2

偏移地址：0x38

复位值：0x0000

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 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 为要装载到实际捕获/比较 2 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR1 寄存器中的 OC2PE 位来使能预装载功能，则该值立刻生效； 否则只在发生更新事件时生效（拷贝到实际起作用的捕获/比较寄存器 2）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC2 输出上发出信号的值。

如果通道 CC2 配置为输入：CCR2 为上一个输入捕获 2 事件 (IC2) 发生时的计数器值。

22.4.16 TIM1/TIM8 捕获/比较寄存器 3 (TIMx_CCR3)

TIM1/TIM8 capture/compare register 3

偏移地址：0x3C

复位值：0x0000

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 15:0 **CCR3[15:0]**：捕获/比较值 (Capture/Compare value)

如果通道 CC3 配置为输出：CCR3 为要装载到实际捕获/比较 3 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR2 寄存器中的 OC3PE 位来使能预装载功能，则该值立刻生效； 否则只在发生更新事件时生效（拷贝到实际起作用的捕获/比较寄存器 3）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC3 输出上发出信号的值。

如果通道 CC3 配置为输入：CCR3 为上一个输入捕获 3 事件 (IC3) 发生时的计数器值。

22.4.17 TIM1/TIM8 捕获/比较寄存器 4 (TIMx_CCR4)

TIM1/TIM8 capture/compare register 4

偏移地址：0x40

复位值：0x0000

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 15:0 **CCR4[15:0]**：捕获/比较值 (Capture/Compare value)

如果通道 CC4 配置为输出：CCR4 为要装载到实际捕获/比较 4 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR2 寄存器中的 OC4PE 位来使能预装载功能，则该值立刻生效； 否则只在发生更新事件时生效（拷贝到实际起作用的捕获/比较寄存器 4）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC4 输出上发出信号的值。

如果通道 CC4 配置为输入：CCR4 为上一个输入捕获 4 事件 (IC4) 发生时的计数器值。

22.4.18 TIM1/TIM8 断路和死区寄存器 (TIMx_BDTR)

TIM1/TIM8 break and dead-time 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
Res.	Res.	Res.	Res.	Res.	Res.	BK2P	BK2E	BK2F[3:0]				BKF[3:0]			
						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
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

注： *由于可以根据 LOCK 配置锁定位 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 31:26 保留。

位 25 **BK2P**：断路 2 极性 (Break 2 polarity)

0：断路输入 BRK2 为低电平有效
1：断路输入 BRK2 为高电平有效

注： 只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。

注： 对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 24 **BK2E**：断路 2 使能 (Break 2 enable)

该位可使能完整的断路 2 保护（包括连接到 bk_acth 的所有源和相应的 BKIN 源）。

0：禁止断路 2 功能
1：使能断路 2 功能

注： BRKIN2 必须只在 OSSR = OSSI = 1 时使用。

注： 编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1 后，此位即无法修改。

注： 对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 23:20 **BK2F[3:0]**：断路 2 滤波器 (Break 2 filter)

此位字段可定义 BRK2 输入的采样频率和适用于 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，BRK2 异步工作
0001：$f_{SAMPLING}=f_{CK_INT}$，N=2
0010：$f_{SAMPLING}=f_{CK_INT}$，N=4
0011：$f_{SAMPLING}=f_{CK_INT}$，N=8
0100：$f_{SAMPLING}=f_{DTS}/2$，N=6
0101：$f_{SAMPLING}=f_{DTS}/2$，N=8
0110：$f_{SAMPLING}=f_{DTS}/4$，N=6
0111：$f_{SAMPLING}=f_{DTS}/4$，N=8
1000：$f_{SAMPLING}=f_{DTS}/8$，N=6
1001：$f_{SAMPLING}=f_{DTS}/8$，N=8
1010：$f_{SAMPLING}=f_{DTS}/16$，N=5
1011：$f_{SAMPLING}=f_{DTS}/16$，N=6
1100：$f_{SAMPLING}=f_{DTS}/16$，N=8
1101：$f_{SAMPLING}=f_{DTS}/32$，N=5
1110：$f_{SAMPLING}=f_{DTS}/32$，N=6
1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

注： 编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1 后，此位即无法修改。

位 19:16 **BKF[3:0]**：断路滤波器 (Break filter)

此位字段可定义 BRK 输入的采样频率和适用于 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，BRK 异步工作
0001：$f_{SAMPLING}=f_{CK_INT}$，N=2
0010：$f_{SAMPLING}=f_{CK_INT}$，N=4
0011：$f_{SAMPLING}=f_{CK_INT}$，N=8
0100：$f_{SAMPLING}=f_{DTS}/2$，N=6
0101：$f_{SAMPLING}=f_{DTS}/2$，N=8
0110：$f_{SAMPLING}=f_{DTS}/4$，N=6
0111：$f_{SAMPLING}=f_{DTS}/4$，N=8
1000：$f_{SAMPLING}=f_{DTS}/8$，N=6
1001：$f_{SAMPLING}=f_{DTS}/8$，N=8
1010：$f_{SAMPLING}=f_{DTS}/16$，N=5
1011：$f_{SAMPLING}=f_{DTS}/16$，N=6
1100：$f_{SAMPLING}=f_{DTS}/16$，N=8
1101：$f_{SAMPLING}=f_{DTS}/32$，N=5
1110：$f_{SAMPLING}=f_{DTS}/32$，N=6
1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

注： 编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1 后，此位即无法修改。

位 15 **MOE**：主输出使能 (Main output enable)

只要断路输入（BRK 或 BRK2）为有效状态，此位便由硬件异步清零。此位由软件置 1，也可根据 AOE 位状态自动置 1。此位仅对配置为输出的通道有效。

0：响应断路事件（2 个）。禁止 OC 和 OCN 输出
响应断路事件或向 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 633 页上的第 22.4.9 节：TIM1/TIM8 捕获/比较使能寄存器 (TIMx_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（如果断路输入 BRK 和 BRK2 均无效）

注： 只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。

位 13 **BKP**：断路极性 (Break polarity)

0：断路输入 BRK 为低电平有效
1：断路输入 BRK 为高电平有效

注： 只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。
注： 对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 12 **BKE**：断路使能 (Break enable)

该位可使能完整的断路保护（包括连接到 bk_acth 的所有源和相应的 BKIN 源）。

0：禁止断路功能
1：使能断路功能

注： 编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1 后，此位即无法修改。
注： 对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。

位 11 **OSSR**：运行模式下的关闭状态选择 (Off-state selection for Run mode)

此位在 MOE=1 时作用于配置为输出模式且具有互补输出的通道。如果定时器中没有互补输出，则不存在 OSSR。

有关详细信息，请参见 OC/OCN 使能说明（*第 633 页上的第 22.4.9 节：TIM1/TIM8 捕获/比较使能寄存器 (TIMx_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，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：　编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 2 后，此位即无法修改。

位 10 **OSSI**：空闲模式下的关闭状态选择 (Off-state selection for Idle mode)

当由于断路事件或软件写操作而使 MOE=0 时，此位作用于配置为输出的通道。

有关详细信息，请参见 OC/OCN 使能说明（*第 633 页上的第 22.4.9 节：TIM1/TIM8 捕获/比较使能寄存器 (TIMx_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：　编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 2 后，此位即无法修改。

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 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：　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)

此位字段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：　只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，此位字段即无法修改。

22.4.19 TIM1/TIM8 DMA 控制寄存器 (TIMx_DCR)

TIM1/TIM8 DMA control register

偏移地址：0x48

复位值：0x0000

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 15:13 保留，必须保持复位值。

位 12:8 **DBL[4:0]**：DMA 连续传送长度 (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，定时器进行一次连续传送），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 次传送
00001：2 次传送
00010：3 次传送
...
10001：18 次传送

示例：以下面的传送为例：DBL = 7 字节且 DBA = TIM2_CR1。

– 如果 DBL = 7 字节且 DBA = TIM2_CR1 表示待传送字节的地址，应通过以下公式给出传送的地址：

(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。在这种情况下，将向自以下地址开始的 7 个寄存器传送数据：(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，
...

22.4.20 TIM1/TIM8 全传输 DMA 地址 (TIMx_DMAR)

TIM1/TIM8 DMA address for full transfer

偏移地址：0x4C

复位值：0x0000

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 15:0 **DMAB[15:0]**：DMA 连续传送寄存器 (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。

22.4.21 TIM1/TIM8 捕获/比较模式寄存器 3 (TIMx_CCMR3)

TIM1/TIM8 capture/compare mode register 3

偏移地址：0x54

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		

输出比较模式

位 31:25 保留，必须保持复位值。

位 24 **OC6M[3]**：输出比较 6 模式——位 3 (Output Compare 6 mode - bit 3)

位 23:17 保留，必须保持复位值。

位 16 **OC5M[3]**：输出比较 5 模式——位 3 (Output Compare 5 mode - bit 3)

位 15 **OC6CE**：输出比较 6 清零使能 (Output compare 6 clear enable)

位 14:12 **OC6M**：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 6:4 **OC5M**：输出比较 5 模式 (Output compare 5 mode)

位 3 **OC2PE**：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。

22.4.22 TIM1/TIM8 捕获/比较寄存器 5 (TIMx_CCR5)

TIM1/TIM8 capture/compare register 5

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
GC5C3	GC5C2	GC5C1	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
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 (Group Channel 5 and Channel 3)

通道 3 输出上失真：

0：OC5REF 对 OC3REFC 无影响

1：OC3REFC 是 OC3REFC 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 TIMxCCMR2 中选择了预装载功能）。

注： 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 (Group Channel 5 and Channel 2)

通道 2 输出上失真：

0：OC5REF 对 OC2REFC 无影响

1：OC2REFC 是 OC2REFC 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： 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 (Group Channel 5 and Channel 1)

通道 1 输出上失真：

0：OC5REF 对 OC1REFC5 无影响

1：OC1REFC 是 OC1REFC 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： 也可在组合 PWM 信号上应用此失真。

位 28:16 保留，必须保持复位值。

位 15:0 **CCR5[15:0]**：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（拷贝到实际起作用的捕获/比较寄存器 5）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC5 输出上发出信号的值。

22.4.23 TIM1/TIM8 捕获/比较寄存器 6 (TIMx_CCR6)

TIM1/TIM8 capture/compare register 6

偏移地址：0x5C

复位值：0x0000

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 15:0 **CCR6[15:0]**：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（拷贝到实际起作用的捕获/比较寄存器 6）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC6 输出上发出信号的值。

22.4.24 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 127. 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
0x00	**TIM1_CR1**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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
0x04	**TIM1_CR2**	Res	Res	Res	Res	Res	Res	Res	Res	MMS2[3:0]				Res	OIS6	Res	OIS5	Res	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
0x08	**TIM1_SMCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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
0x0C	**TIM1_DIER**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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
0x10	**TIM1_SR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	C6IF	C5IF	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
0x14	**TIM1_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	COM	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																								0	0	0	0	0	0	0	0	0

表 127. 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
0x18	**TIM1_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	0
	TIM1_CCMR1 Input Capture mode	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	IC2F[3:0]				IC2PSC [1:0]		CC2S [1:0]		IC1F[3:0]				IC1PSC [1:0]		CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	**TIM1_CCMR2** Output Compare mode	Res	Res	Res	Res	Res	Res	Res	OC4M[3]	Res	Res	Res	Res	Res	Res	Res	OC3M[3]	OC4CE	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	0
	TIM1_CCMR2 Input Capture mode	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	IC4F[3:0]				IC4PSC [1:0]		CC4S [1:0]		IC3F[3:0]				IC3PSC [1:0]		CC3S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	**TIM1_CCER**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC6P	CC6E	Res	Res	CC5P	CC5E	Res	Res	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
0x24	**TIM1_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
0x28	**TIM1_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	**TIM1_ARR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ARR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	**TIM1_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
0x34	**TIM1_CCR1**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	**TIM1_CCR2**	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
0x3C	**TIM1_CCR3**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR3[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	**TIM1_CCR4**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR4[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 127. 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
0x44	**TIM1_BDTR**	Res	Res	Res	Res	Res	Res	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
0x48	**TIM1_DCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DBL[4:0]					Res	Res	Res	DBA[4:0]				
	Reset value																				0	0	0	0	0				0	0	0	0	0
0x4C	**TIM1_DMAR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DMAB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x54	**TIM1_CCMR3** Output Compare mode	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		
0x58	**TIM1_CCR5**	GC5C3	GC5C2	GC5C1	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR5[15:0]															
	Reset value	0	0	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x5C	**TIM1_CCR6**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR6[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 64 页上的第 2.2.2 节*。

22.4.25 TIM8 寄存器映射

TIM8 寄存器可映射为 16 位可寻址寄存器，如下表所述：

表 128. TIM8 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	**TIM8_CR1**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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
0x04	**TIM8_CR2**	Res	Res	Res	Res	Res	Res	Res	Res	MMS2[3:0]				Res	OIS6	Res	OIS5	Res	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

表 128. TIM8 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	**TIM8_SMCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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
0x0C	**TIM8_DIER**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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
0x10	**TIM8_SR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	C6IF	C5IF	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
0x14	**TIM8_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	COM	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																								0	0	0	0	0	0	0	0	0
0x18	**TIM8_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
	TIM8_CCMR1 Input Capture mode	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	IC2F[3:0]				IC2PSC [1:0]		CC2S [1:0]		IC1F[3:0]				IC1PSC [1:0]		CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	**TIM8_CCMR2** Output Compare mode	Res	Res	Res	Res	Res	Res	Res	OC4M[3]	Res	Res	Res	Res	Res	Res	Res	OC3M[3]	OC4CE	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
	TIM8_CCMR2 Input Capture mode	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	IC4F[3:0]				IC4PSC [1:0]		CC4S [1:0]		IC3F[3:0]				IC3PSC [1:0]		CC3S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	**TIM8_CCER**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC6P	CC6E	Res	Res	CC5P	CC5E	Res	Res	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
0x24	**TIM8_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
0x28	**TIM8_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	**TIM8_ARR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ARR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 128. TIM8 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
0x30	**TIM8_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
0x34	**TIMx_CCR1**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	**TIM8_CCR2**	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
0x3C	**TIM8_CCR3**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR3[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	**TIM8_CCR4**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR4[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	**TIM8_BDTR**	Res	Res	Res	Res	Res	Res	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
0x48	**TIM8_DCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DBL[4:0]					Res	Res	Res	DBA[4:0]				
	Reset value																				0	0	0	0	0				0	0	0	0	0
0x4C	**TIM8_DMAR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DMAB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x54	**TIM8_CCMR3** Output Compare mode	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		
0x58	**TIM8_CCR5**	GC5C3	GC5C2	GC5C1	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR5[15:0]															
	Reset value	0	0	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x5C	**TIM8_CCR6**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CCR6[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 64 页上的第 2.2.2 节*。

23 通用定时器（TIM2/TIM3/TIM4/TIM5）

23.1 TIM2/TIM3/TIM4/TIM5 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。

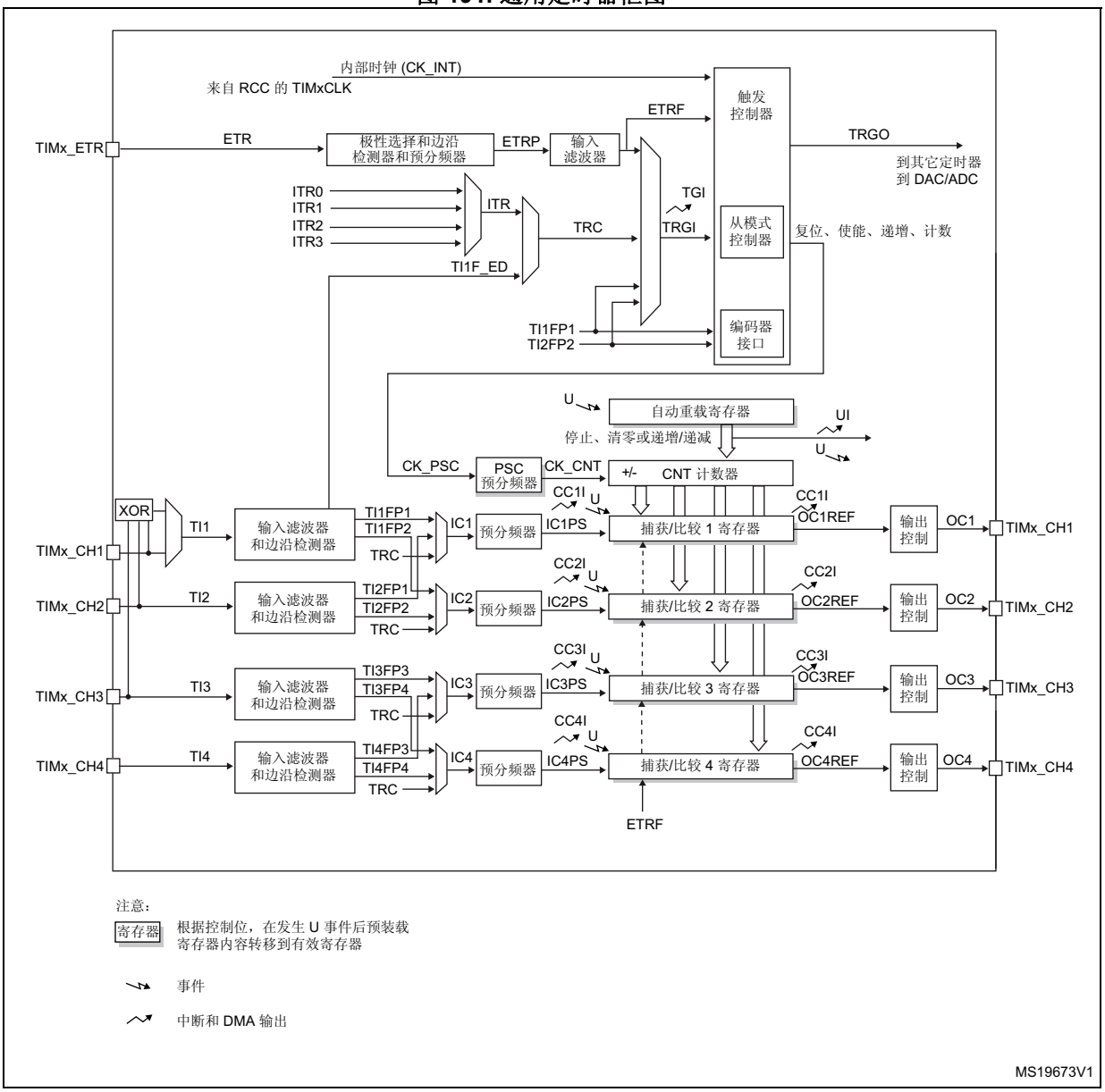
这些定时器彼此完全独立，不共享任何资源。如*第 23.3.18 节*中所述，它们可以同步操作。

23.2 TIM2/TIM3/TIM4/TIM5 的主要特性

通用 TIMx 定时器具有以下特性：

- 16 位（TIM3 和 TIM4）或 32 位（TIM2 和 TIM5）递增、递减和递增/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 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（边沿和中心对齐模式）
  - 单脉冲模式输出
- 使用外部信号控制定时器且可实现多个定时器互连的同步电路。
- 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
- 触发输入作为外部时钟或者逐周期电流管理

图 191. 通用定时器框图



23.3 TIM2/TIM3/TIM4/TIM5 功能说明

23.3.1 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、递减计数或同时递增和递减计数。计数器的时钟可通过预分频器进行分频。

计数器、自动重载寄存器和预分频器寄存器可通过软件进行读写。即使在计数器运行时也可执行读写操作。

时基单元包括：

- 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直接传送到影子寄存器，也可以在每次发生更新事件 (UEV) 时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位 (ARPE)。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。下文将针对各配置的更新事件的产生进行详细介绍。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位 (CEN) 置 1 时，才会启动计数器（有关计数器使能的更多详细信息，另请参见从模式控制器的相关说明）。

请注意，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频率进行分频，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 192 和*图 193* 以一些示例说明在预分频比实时变化时计数器的行为：

图 192. 预分频器分频由 1 变为 2 时的计数器时序图

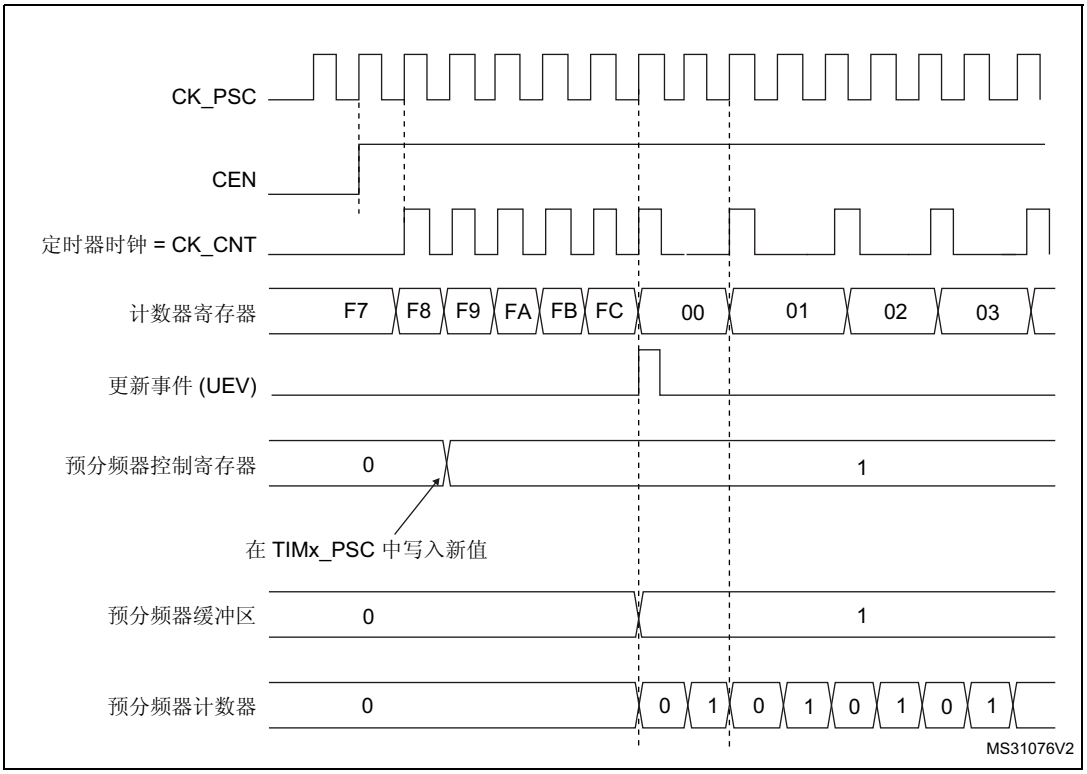
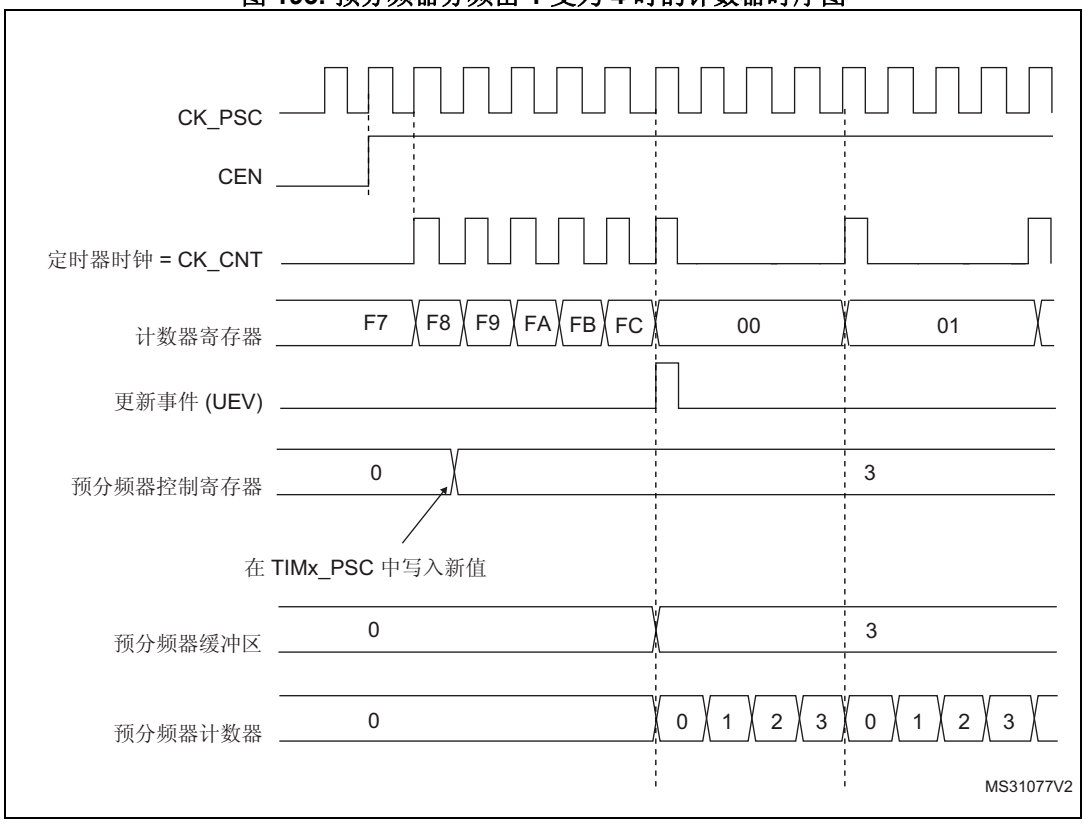


图 193. 预分频器分频由 1 变为 4 时的计数器时序图



23.3.2 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）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）
- 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 194. 计数器时序图，1 分频内部时钟

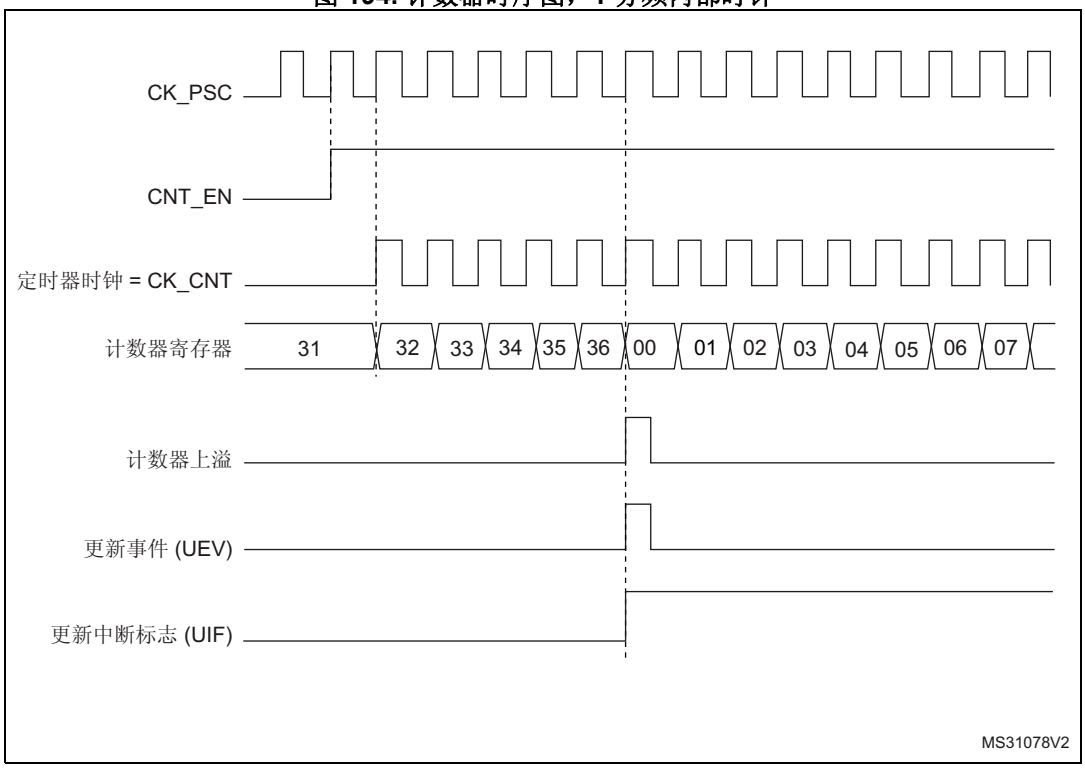


图 195. 计数器时序图，2 分频内部时钟

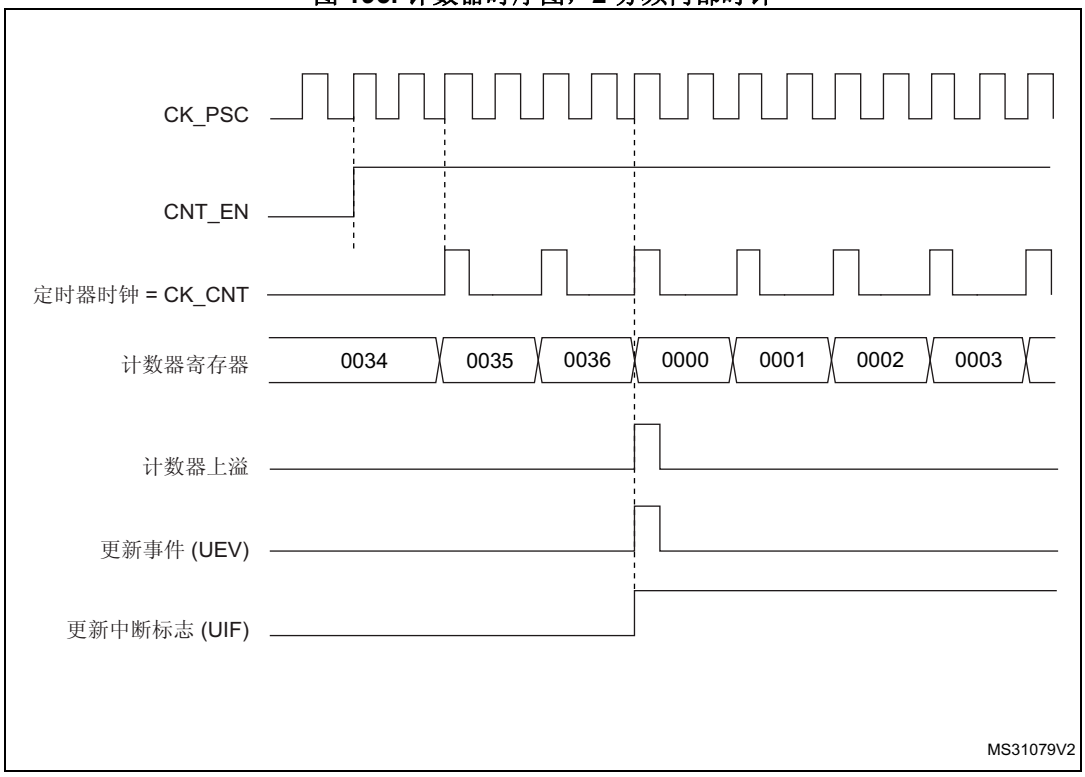


图 196. 计数器时序图，4 分频内部时钟

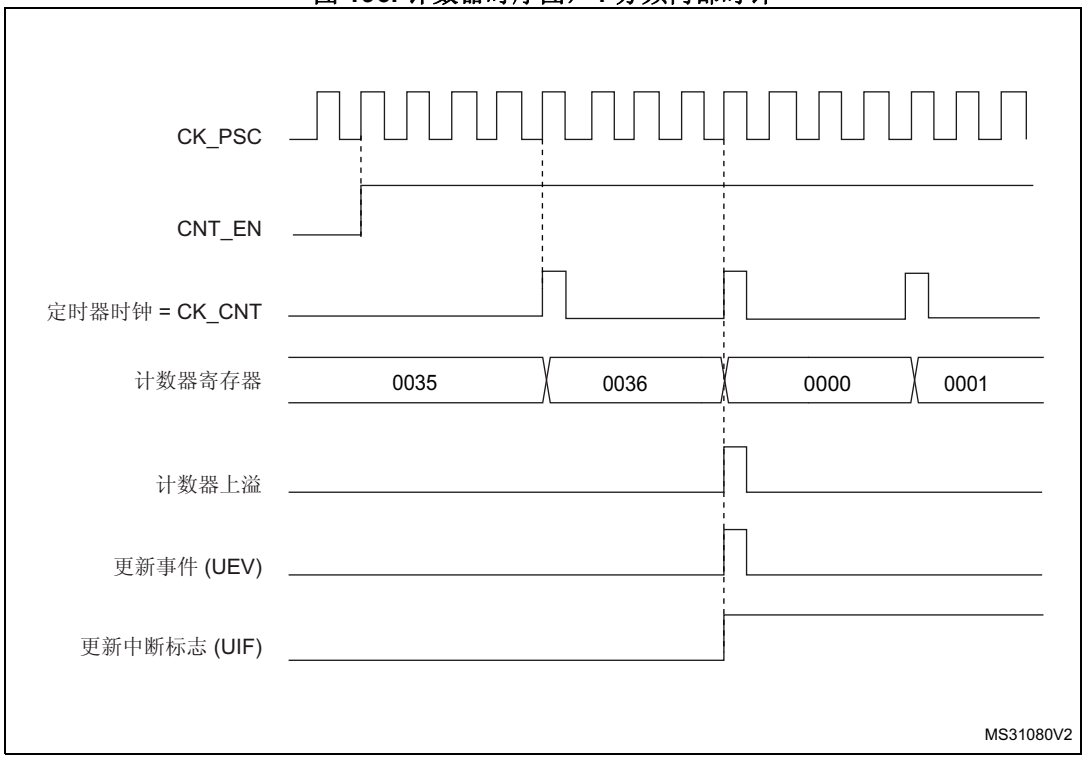


图 197. 计数器时序图，N 分频内部时钟

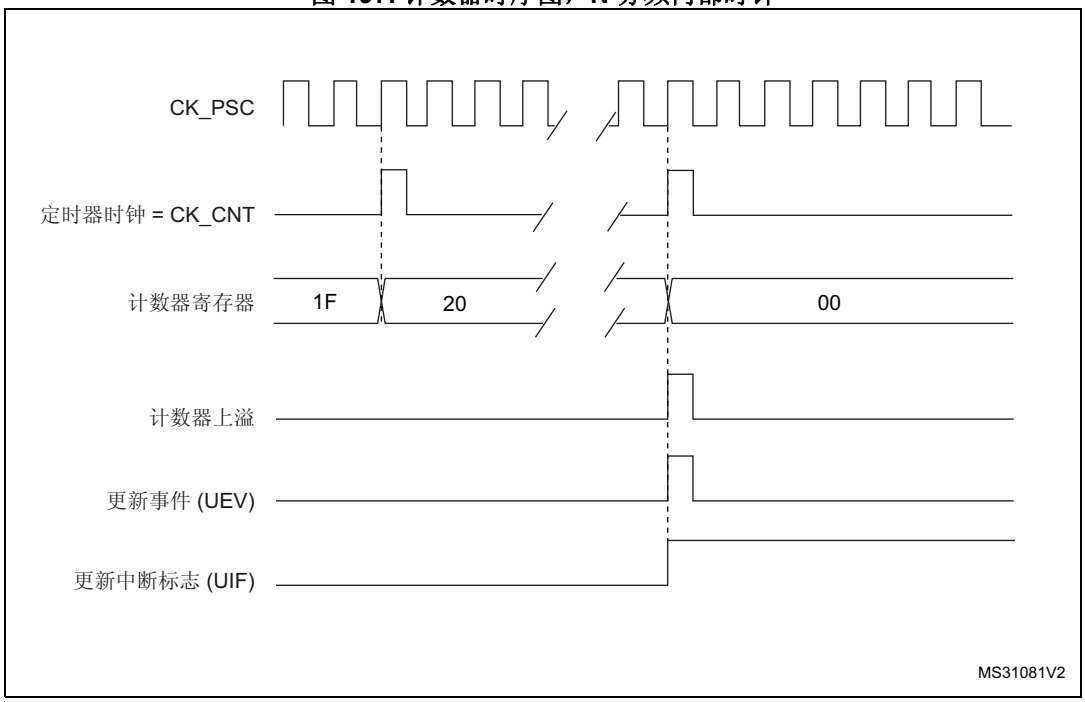


图 198. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载）

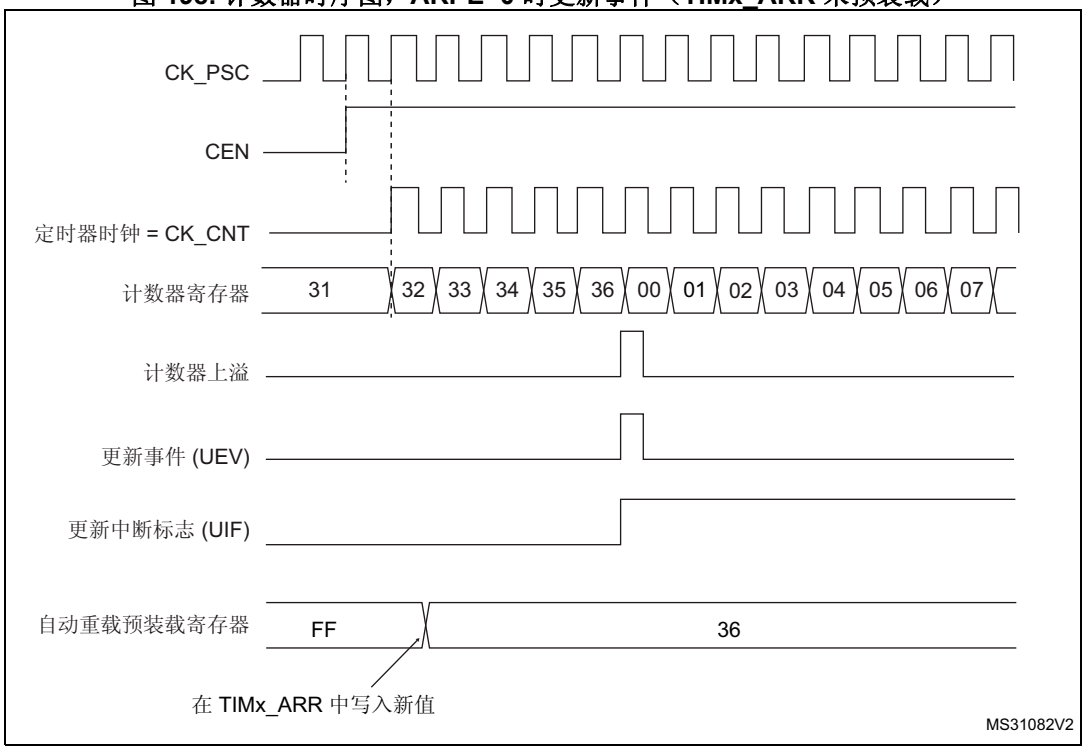
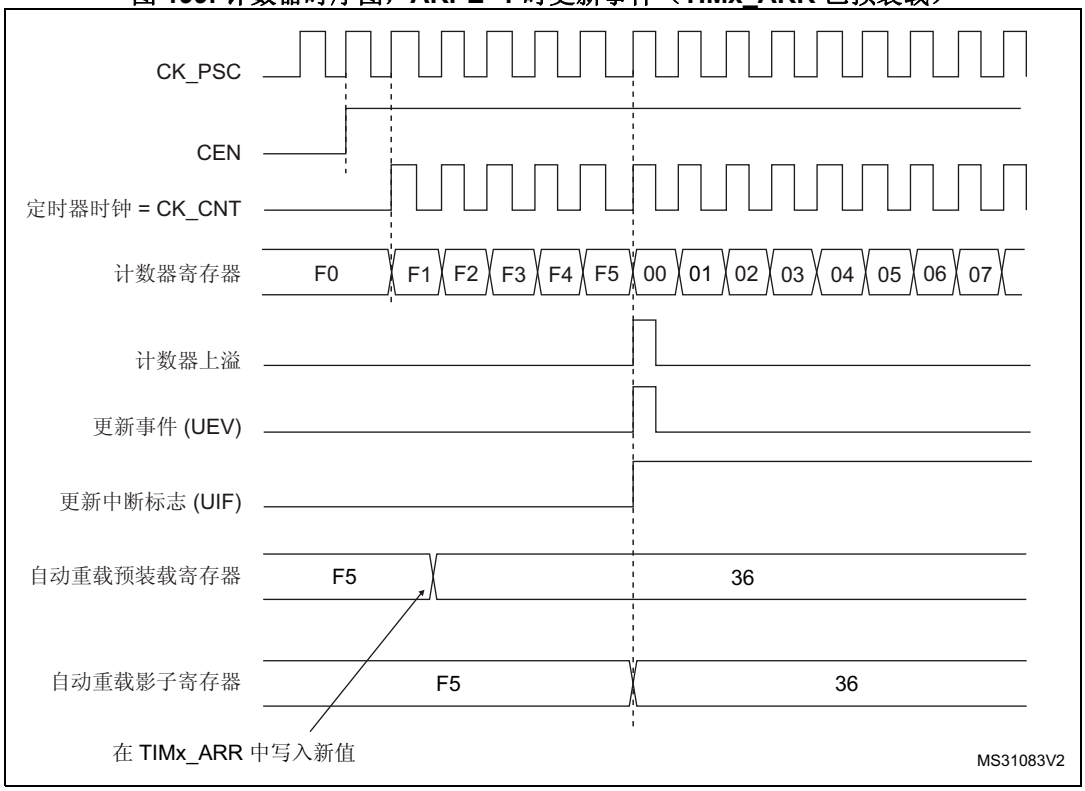


图 199. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 已预装载）



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，计数器会重新从当前自动重载值开始计数，而预分频器计数器则重新从 0 开始计数（但预分频比保持不变）。

此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，ARR 寄存器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 200. 计数器时序图，1 分频内部时钟

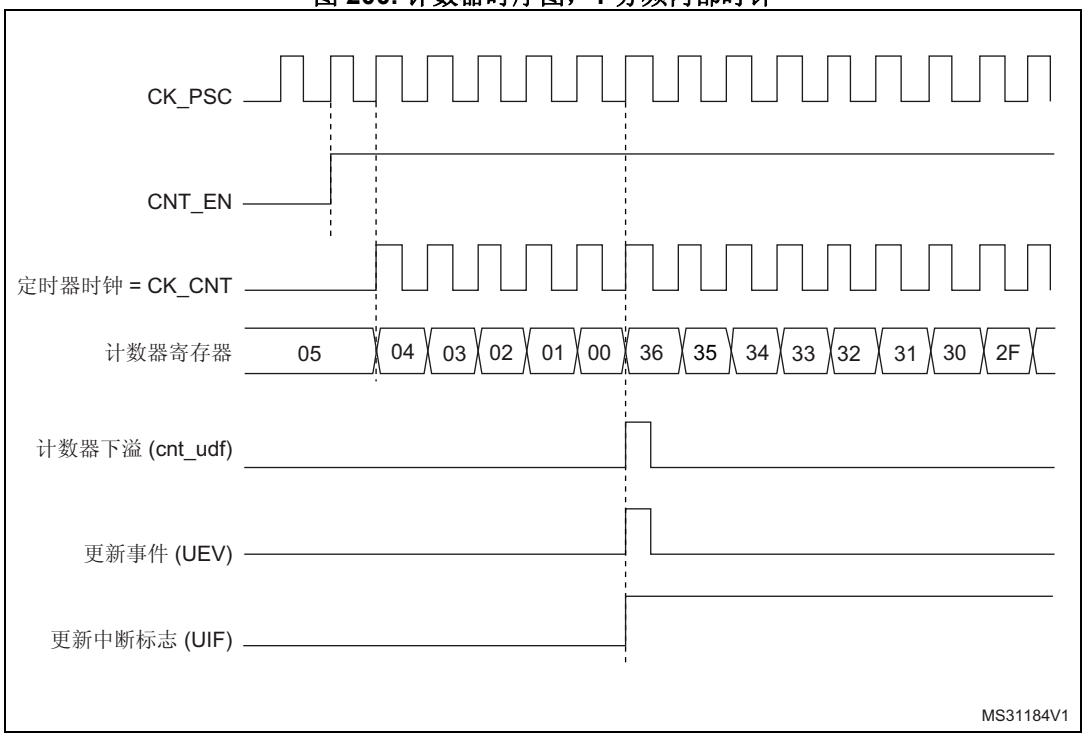


图 201. 计数器时序图，2 分频内部时钟

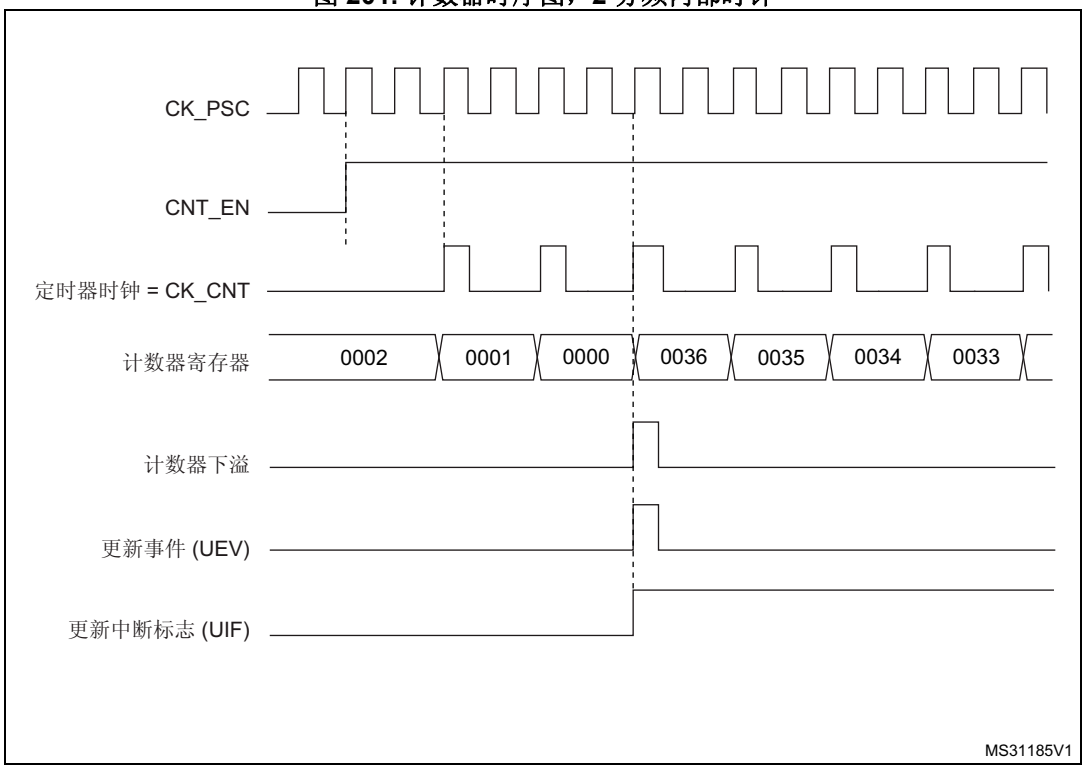


图 202. 计数器时序图，4 分频内部时钟

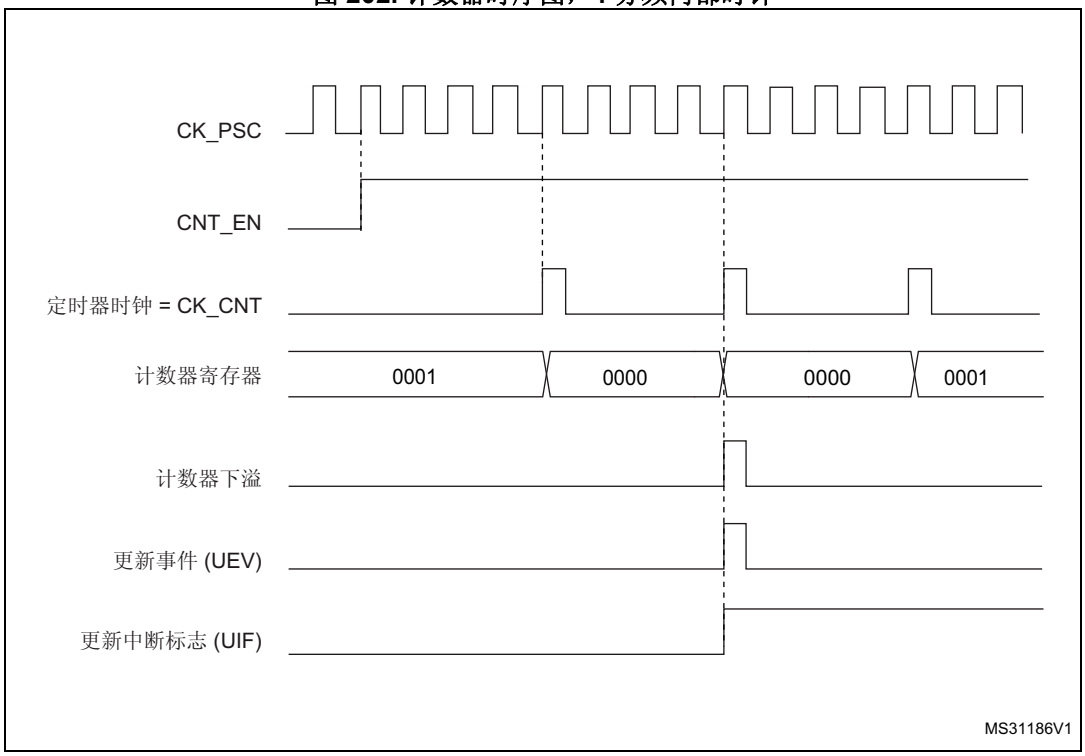


图 203. 计数器时序图，N 分频内部时钟

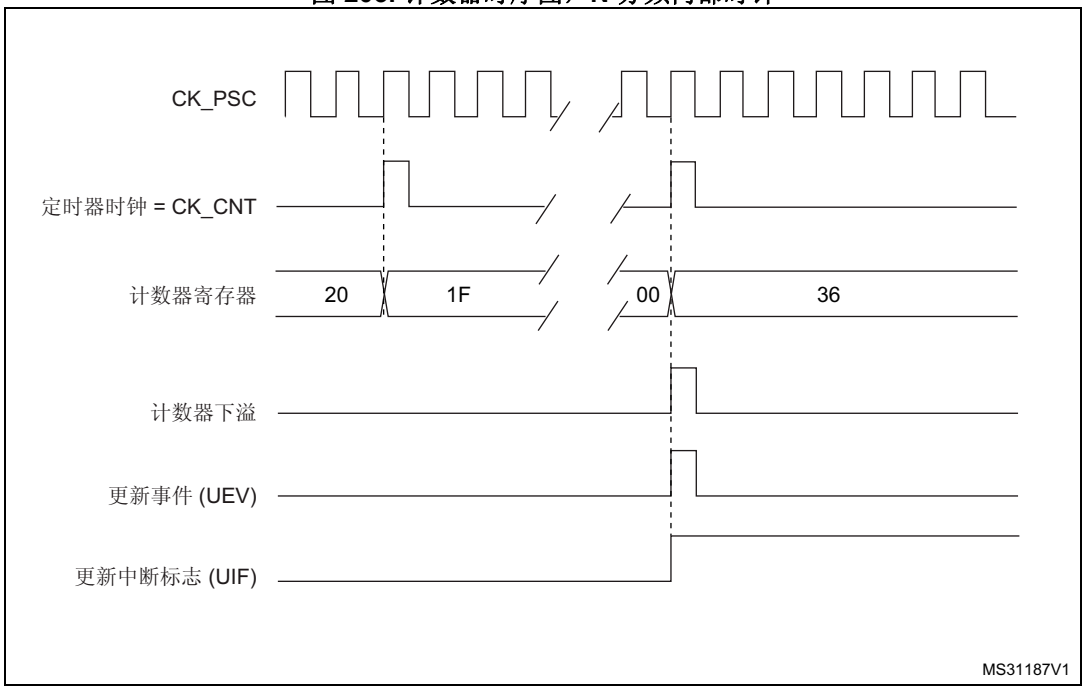
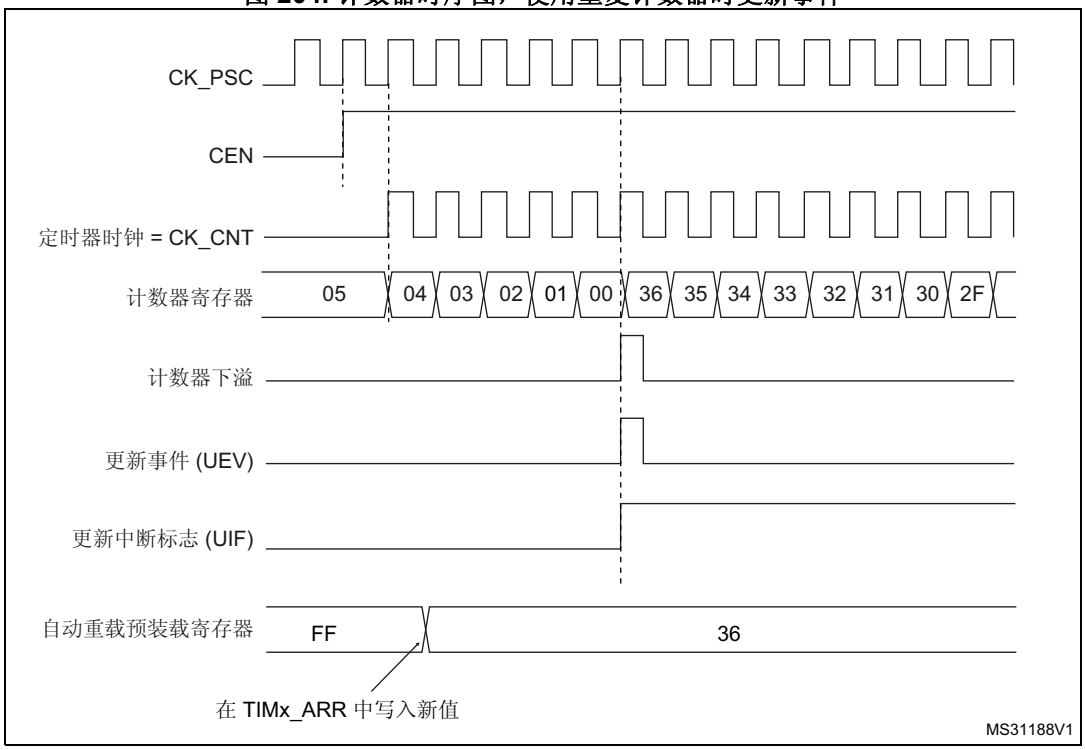


图 204. 计数器时序图，使用重复计数器时更新事件



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”）。

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。

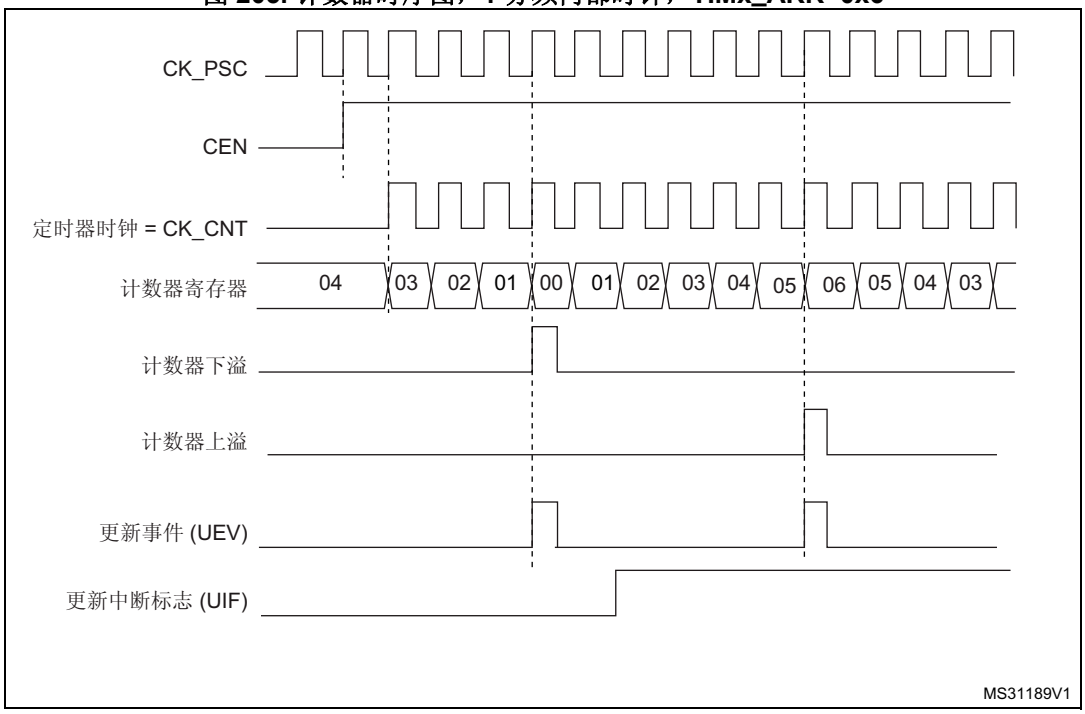
此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，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 205. 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 692 页上的第 23.4.1 节：TIMx 控制寄存器 1 (TIMx_CR1)*）。

图 206. 计数器时序图，2 分频内部时钟

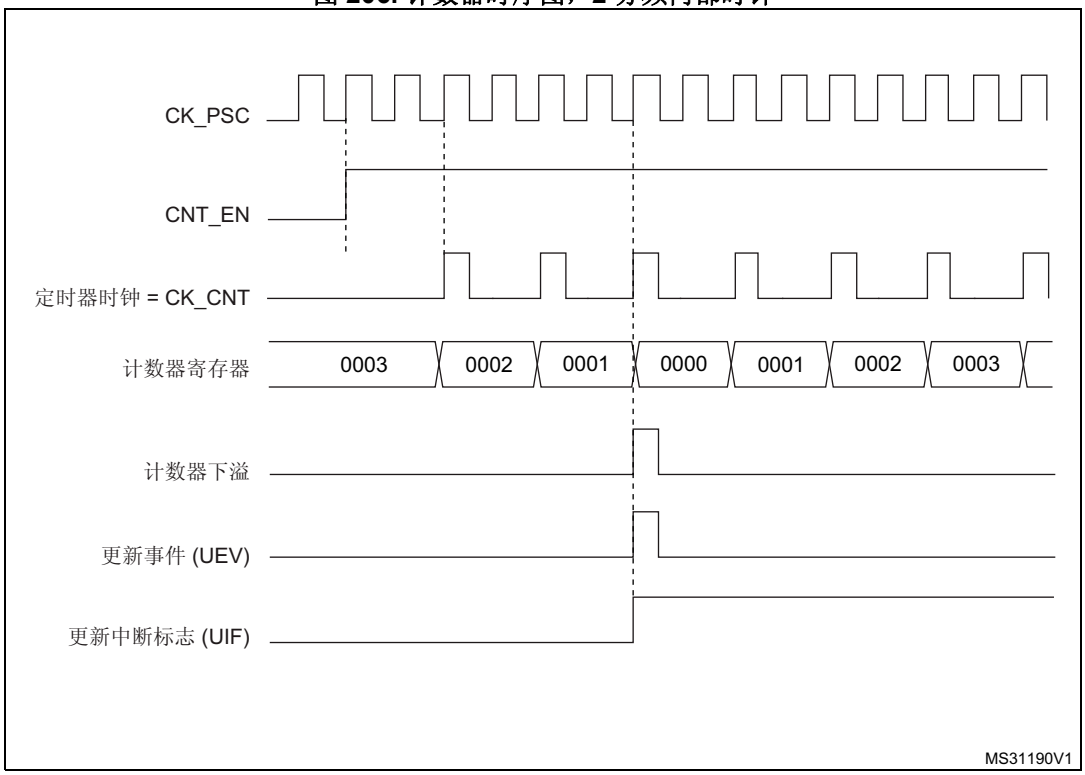
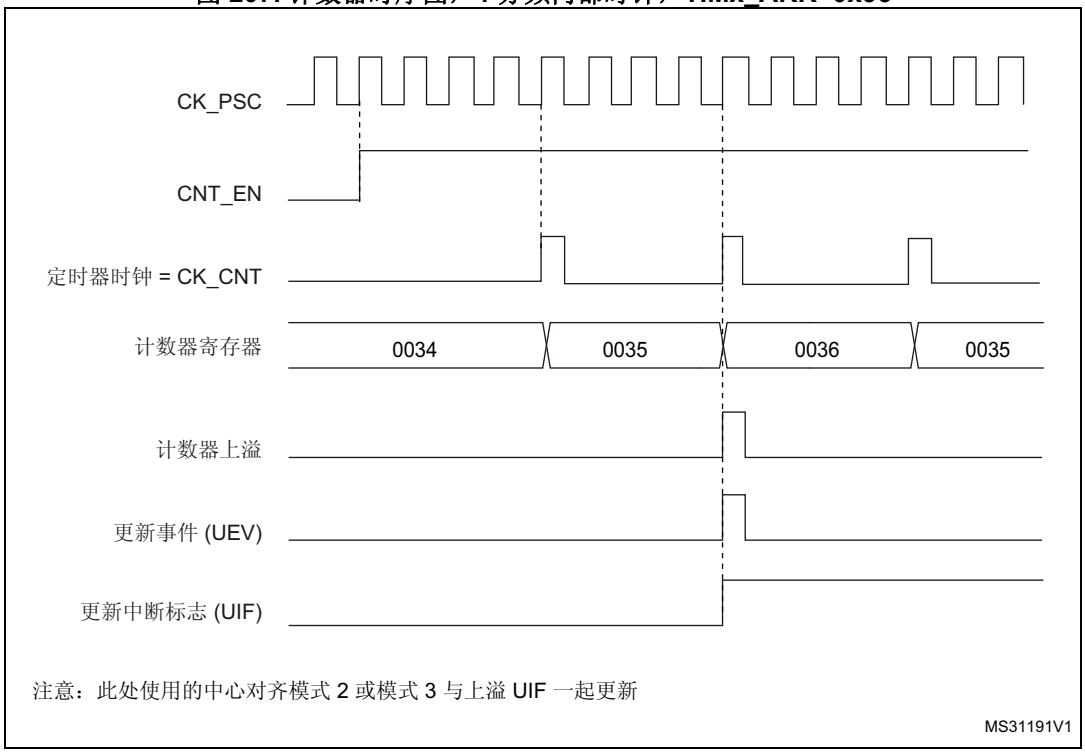


图 207. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36



1. 中心对齐模式 2 或模式 3 与上溢 UIF 结合使用。

图 208. 计数器时序图，N 分频内部时钟

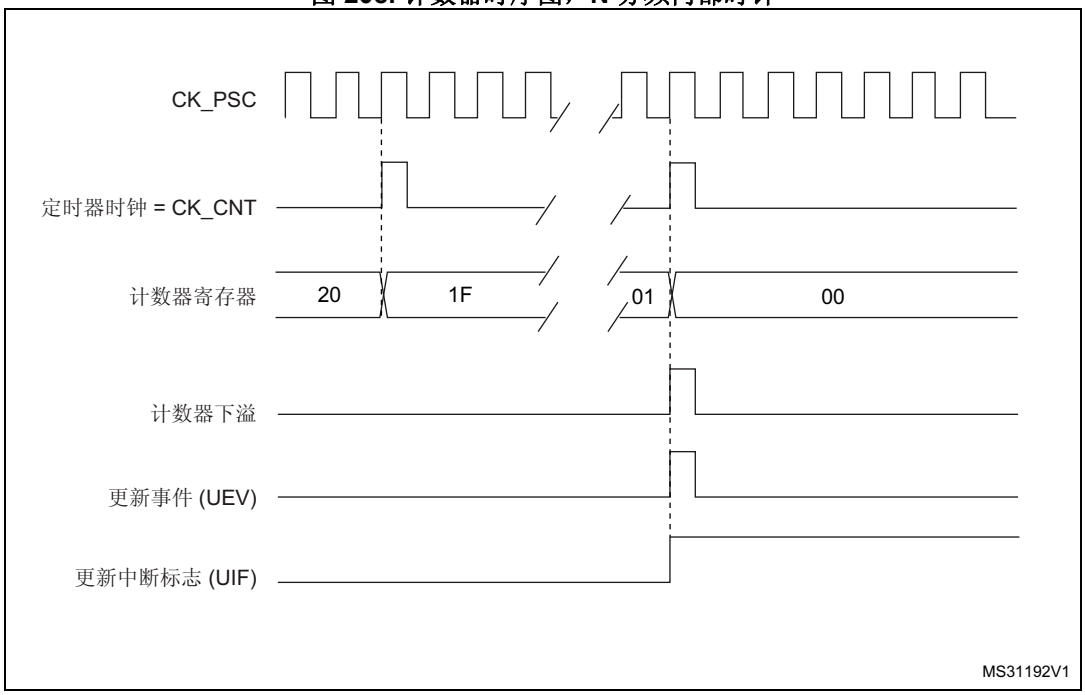


图 209. 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

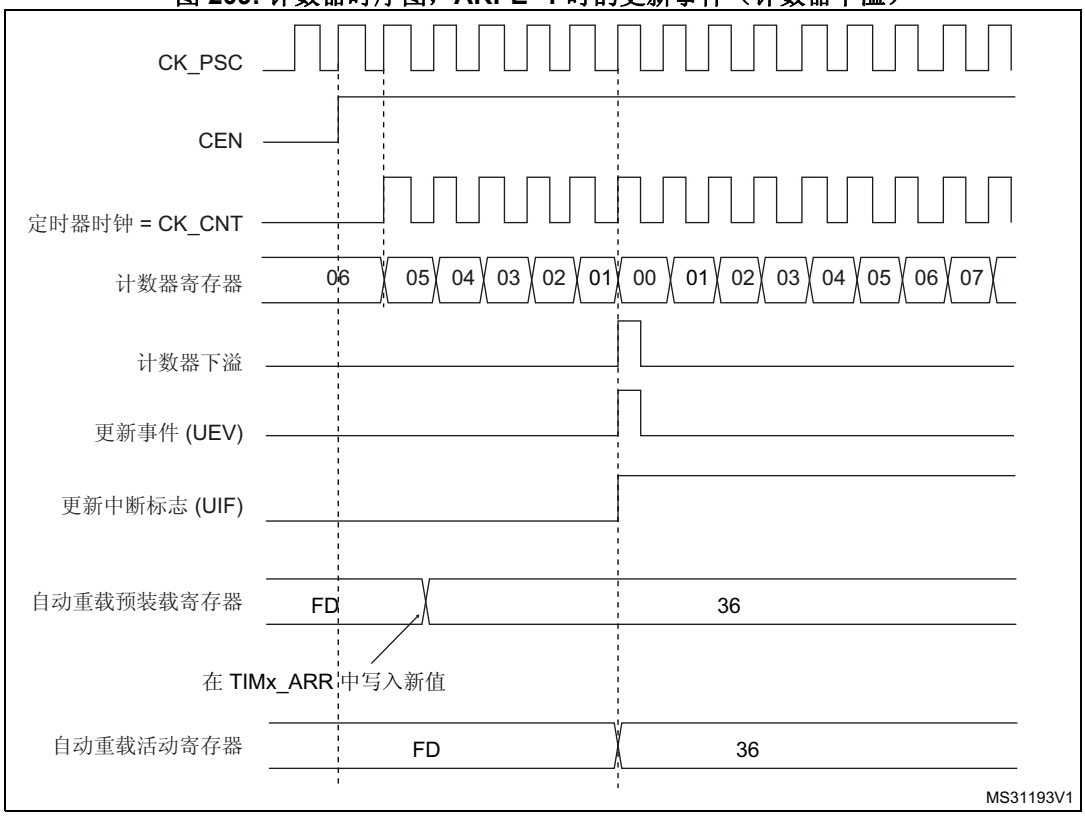
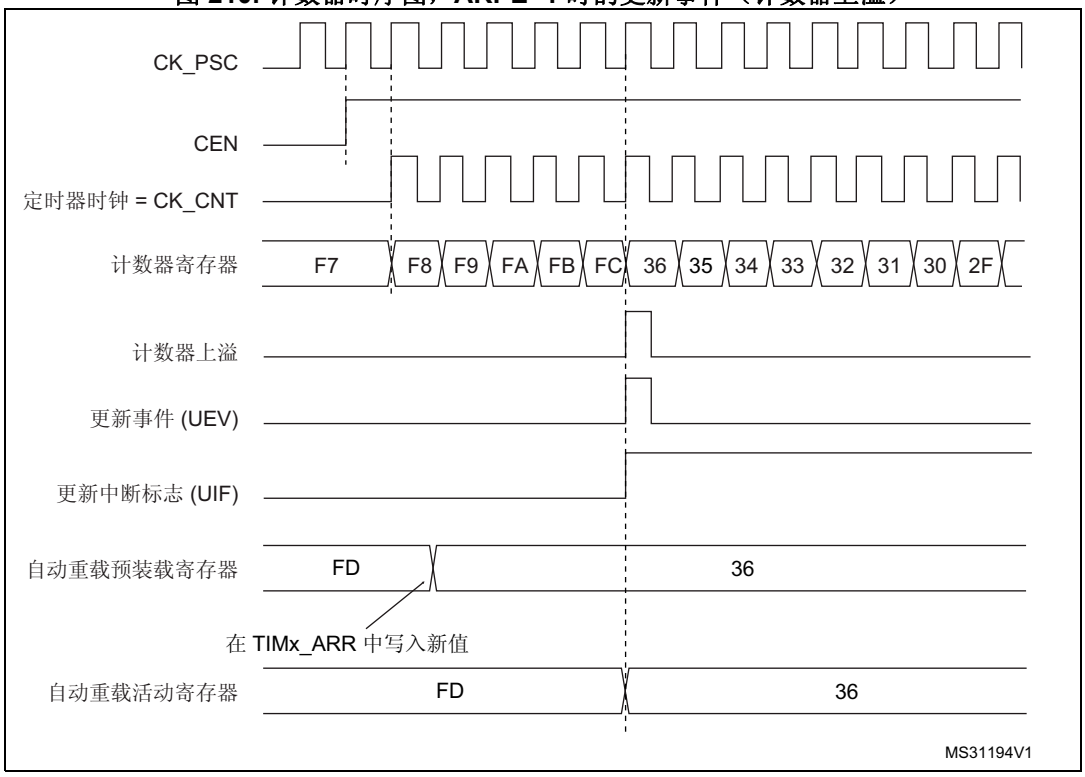


图 210. 计数器时序图，ARPE=1 时的更新事件（计数器上溢）



23.3.3 时钟选择

计数器时钟可由下列时钟源提供：

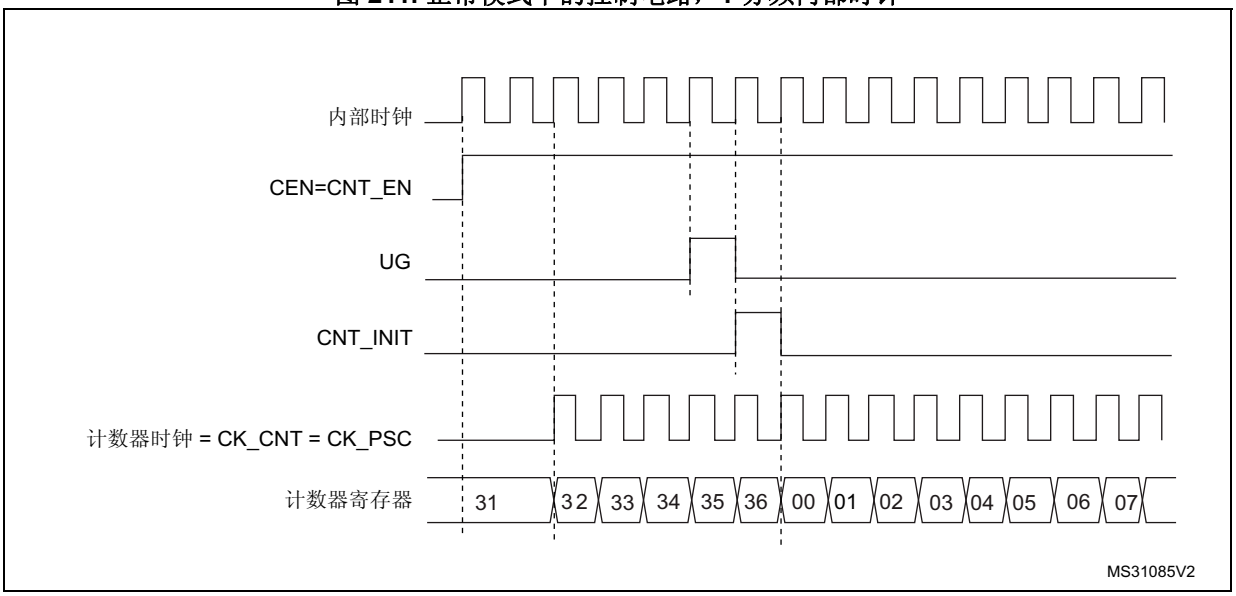
- 内部时钟 (CK_INT)
- 外部时钟模式 1：外部输入引脚 (TIx)
- 外部时钟模式 2：外部触发输入 (ETR)
- 外部触发输入 (ITRx)：使用一个定时器作为另一定时器的预分频器，例如，可将定时器 13 配置为定时器 2 的预分频器。更多详细信息，请参见：*第 687 页上的将一个定时器用作另一个定时器的预分频器*。

内部时钟源 (CK_INT)

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 CK_INT 提供。

图 211 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）。

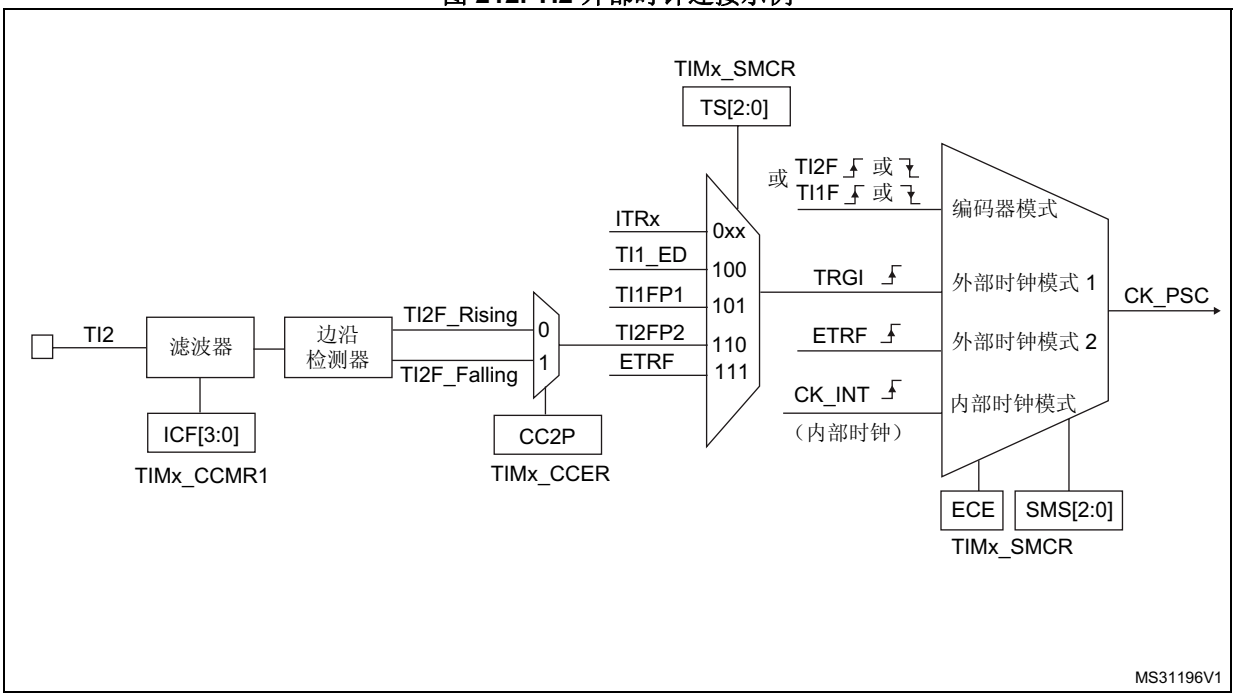
图 211. 正常模式下的控制电路，1 分频内部时钟



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 212. TI2 外部时钟连接示例



例如，要使递增计数器在 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 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）。

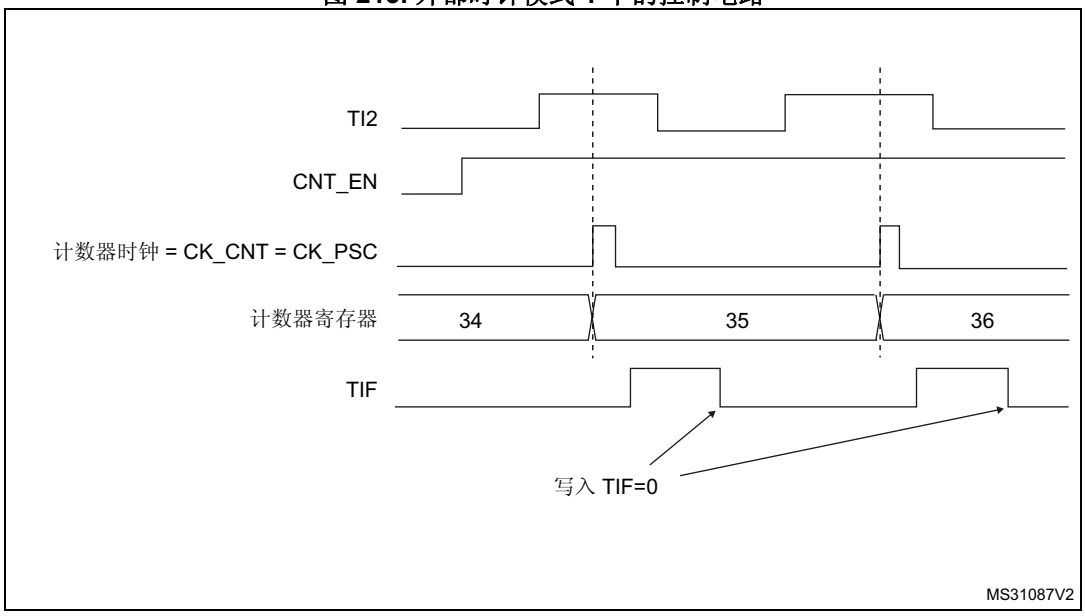
注： *由于捕获预分频器不用于触发操作，因此无需对其进行配置。*

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=110 来选择 TI2 作为输入源。
6. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

当 TI2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

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

图 213. 外部时钟模式 1 下的控制电路



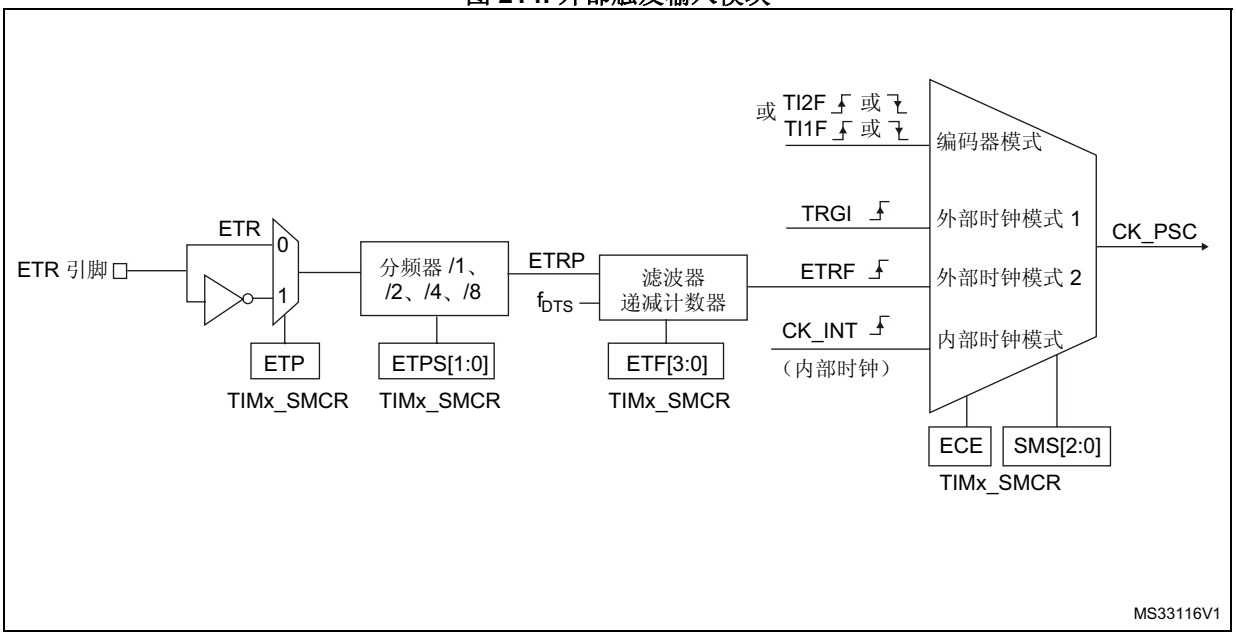
外部时钟源模式 2

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 ETR 出现上升沿或下降沿时计数。

图 214 简要介绍了外部触发输入模块。

图 214. 外部触发输入模块



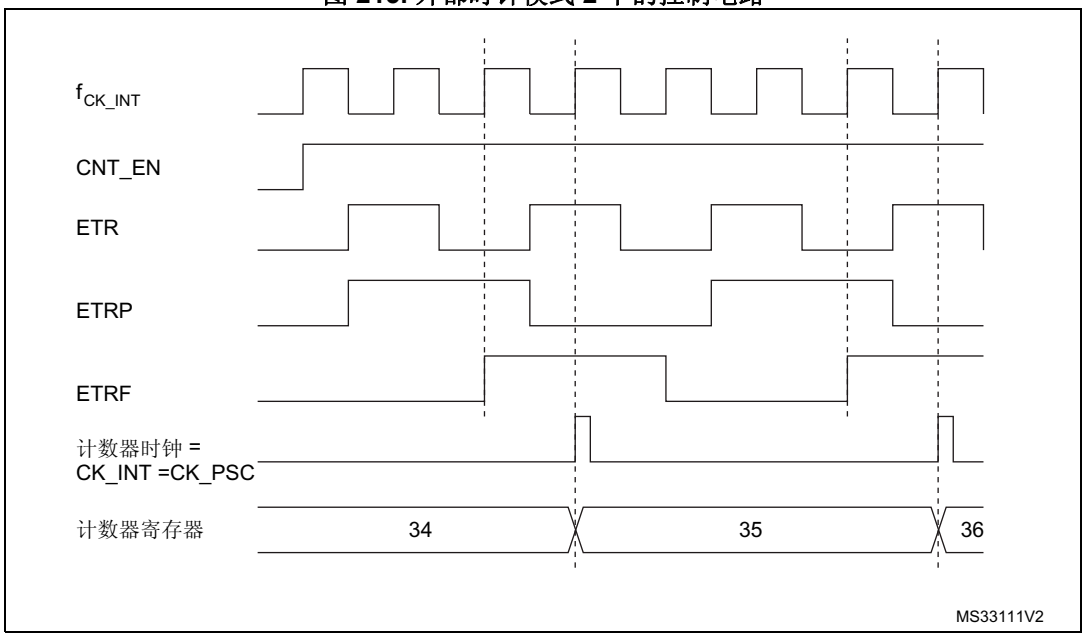
例如，要使递增计数器在 ETR 每出现 2 个上升沿时计数，请执行以下步骤：

1. 由于此例中不需滤波器，因此在 TIMx_SMCR 寄存器中写入 ETF[3:0]=0000。
2. 通过在 TIMx_SMCR 寄存器中写入 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 ETR 引脚的上升沿检测。
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。

ETR 每出现 2 个上升沿，计数器计数一次。

ETR 的上升沿与实际计数器时钟之间的延迟是由于 ETRP 信号的重新同步电路引起的。

图 215. 外部时钟模式 2 下的控制电路



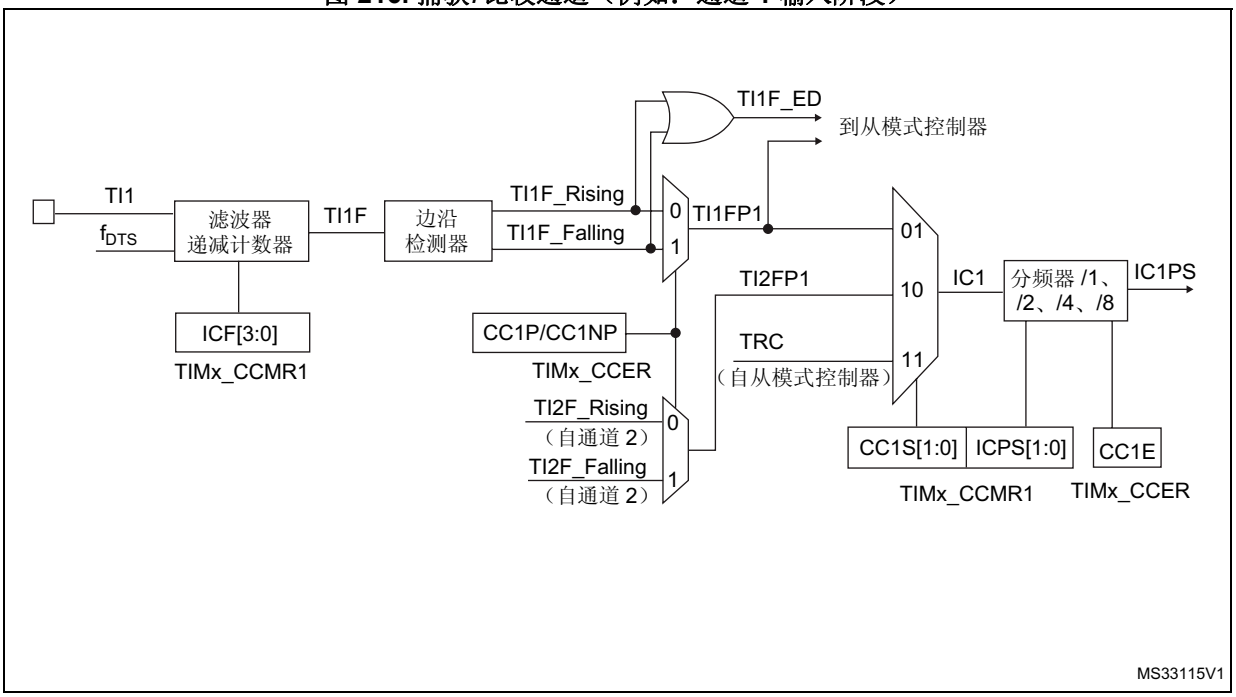
23.3.4 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入阶段（数字滤波、多路复用和预分频器）和一个输出阶段（比较器和输出控制）构建而成。

下图简要介绍了一路捕获/比较通道。

输入阶段对相应的 TIx 输入进行采样，生成一个滤波后的信号 TIxF。然后，带有极性选择功能的边沿检测器生成一个信号 (TIxFPx)，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 216. 捕获/比较通道（例如：通道 1 输入阶段）



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 217. 捕获/比较通道 1 主电路

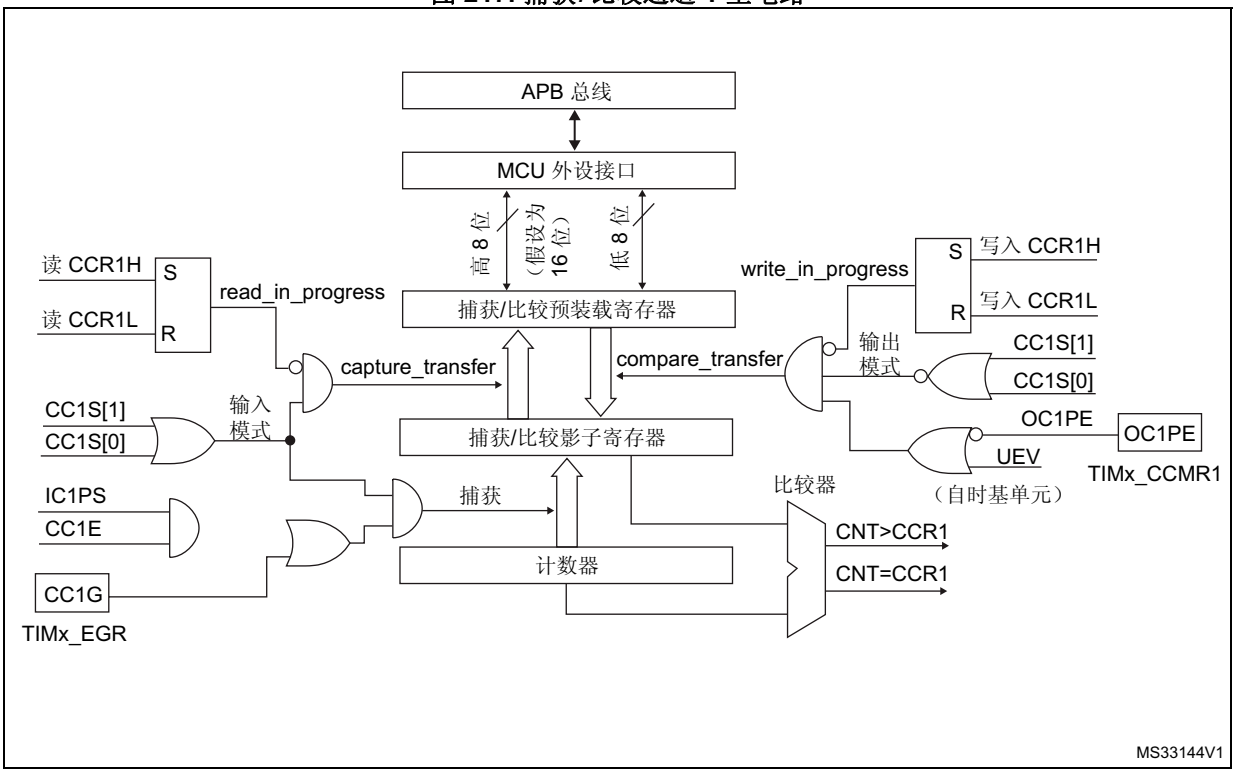
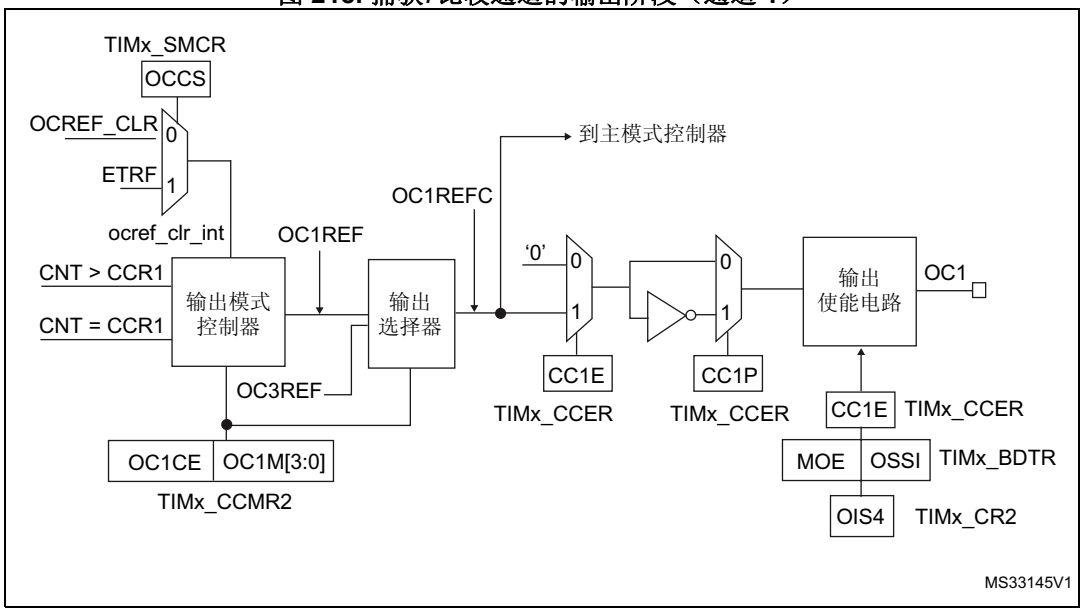


图 218. 捕获/比较通道的输出阶段（通道 1）



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。始终可通过读写操作访问预装载寄存器。

在捕获模式下，捕获实际发生在影子寄存器中，然后将影子寄存器的内容复制到预装载寄存器中。

在比较模式下，预装载寄存器的内容将复制到影子寄存器中，然后将影子寄存器的内容与计数器进行比较。

23.3.5 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，则会将重复捕获标志 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 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. 选择有效输入：TIMx_CCR1 必须连接到 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。
2. 根据连接到定时器的信号，对所需的输入滤波带宽进行编程（如果输入为 TIx 之一，则对 TIMx_CCMRx 寄存器中的 ICxF 位进行编程）。假设信号边沿变化时，输入信号最多在 5 个内部时钟周期内发生抖动。因此，我们必须将滤波带宽设置为大于 5 个内部时钟周期。在检测到 8 个具有新电平的连续采样（以 f_{DTS} 频率采样）后，可以确认 TI1 上的跳变沿。然后向 TIMx_CCMR1 寄存器中的 IC1F 位写入 0011。
3. 通过向 TIMx_CCER 寄存器中的 CC1P 位和 CC1NP 位写入 000，选择 TI1 通道的有效跳变沿（本例中为上升沿）。

4. 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）。
5. 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。
6. 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。

要处理重复捕获，建议在读出捕获溢出标志之前读取数据。这样可避免丢失在读取捕获溢出标志之后与读取数据之前可能出现的重复捕获信息。

注： *通过软件将 TIMx_EGR 寄存器中的相应 CCxG 位置 1 可生成 IC 中断和/或 DMA 请求。*

23.3.6 PWM 输入模式

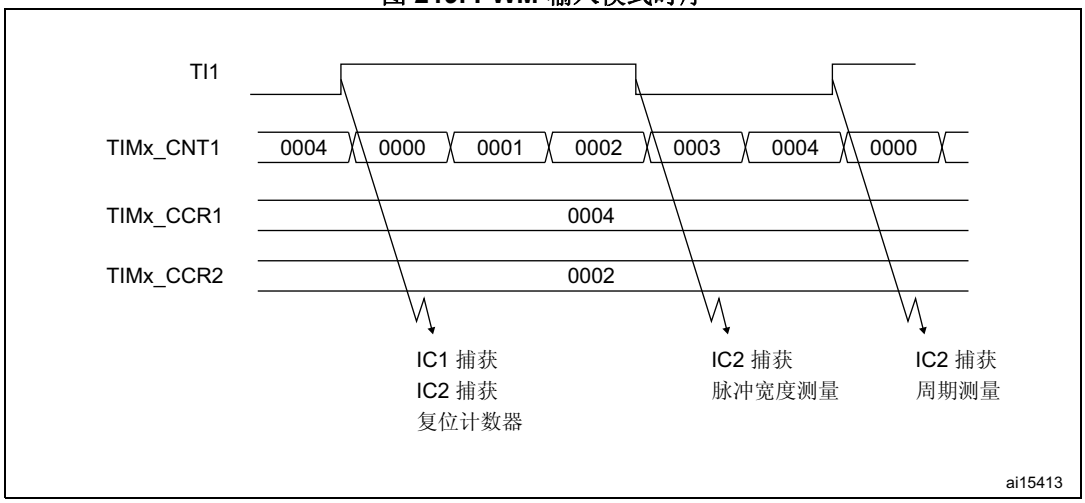
此模式是输入捕获模式的一个特例。其实现步骤与输入捕获模式基本相同，仅存在以下不同之处：

- 两个 ICx 信号被映射至同一个 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。

例如，可通过以下步骤对应用于 TI1 的 PWM 的周期（位于 TIMx_CCR1 寄存器中）和占空比（位于 TIMx_CCR2 寄存器中）进行测量（取决于 CK_INT 频率和预分频器的值）：

1. 选择 TIMx_CCR1 的有效输入：向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01（选择 TI1）。
2. 选择 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）。
3. 选择 TIMx_CCR2 的有效输入：向 TIMx_CCMR1 寄存器中的 CC2S 写入 10（选择 TI1）。
4. 选择 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）。
5. 选择有效触发输入：向 TIMx_SMCR 寄存器中的 TS 位写入 101（选择 TI1FP1）。
6. 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。
7. 使能捕获：向 TIMx_CCER 寄存器中的 CC1E 位和 CC2E 位写入“1”。

图 219. 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 TI1FP1 和 TI2FP2 与从模式控制器相连。

23.3.7 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（OCxREF 和 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 (ocxref/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。ocxref 进而强制设置为高电平（OCxREF 始终为高电平有效），同时 OCx 获取 CCxP 极性位的相反值。

例如：CCxP=0（OCx 高电平有效）=> 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 ocxref 信号强制设置为低电平。

无论如何，TIMx_CCRx 影子寄存器与计数器之间的比较仍会执行，而且允许将标志置 1。因此可发送相应的中断和 DMA 请求。输出比较模式一节对此进行了介绍。

23.3.8 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载寄存器。

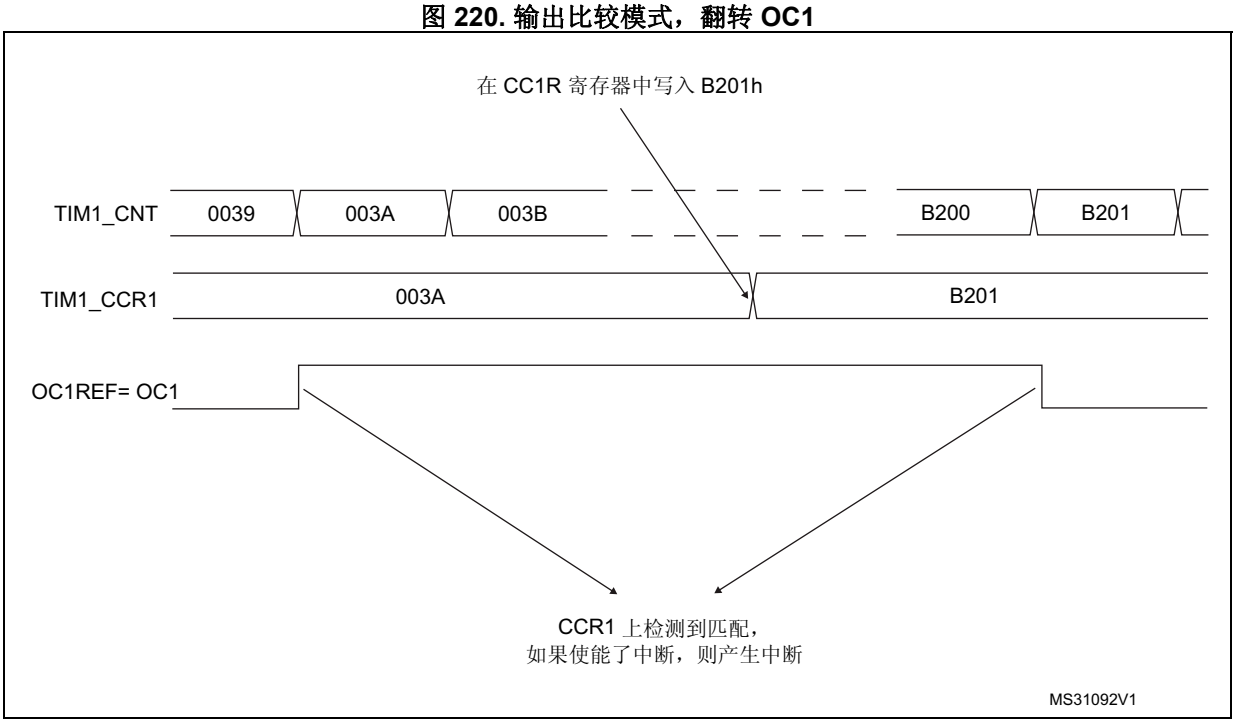
在输出比较模式下，更新事件 UEV 对 ocxref 和 OCx 输出毫无影响。同步的精度可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

步骤

1. 选择计数器时钟（内部、外部、预分频器）。
2. 在 TIMx_ARR 和 TIMx_CCRx 寄存器中写入所需数据。
3. 如果要生成中断和/或 DMA 请求，将 CCxIE 位和/或 CCxDE 位置 1。
4. 选择输出模式。例如，当 CNT 与 CCRx 匹配、未使用预装载 CCRx 并且 OCx 使能且为高电平有效时，必须写入 OCxM=011、OCxPE=0、CCxP=0 和 CCxE=1 来翻转 OCx 输出引脚。
5. 通过将 TIMx_CR1 寄存器中的 CEN 位置 1 来使能计数器。

可随时通过软件更新 TIMx_CCRx 寄存器以控制输出波形，前提是未使能预装载寄存器（OCxPE=0，否则仅当发生下一个更新事件 UEV 时，才会更新 TIMx_CCRx 影子寄存器）。*图 220* 给出了一个示例。

图 220. 输出比较模式，翻转 OC1



23.3.9 PWM 模式

脉冲宽度调制模式可以生成一个信号，该信号频率由 TIMx_ARR 寄存器值决定，其占空比则由 TIMx_CCRx 寄存器值决定。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），可以独立选择各通道（每个 OCx 输出对应一个 PWM）的 PWM 模式。必须通过将 TIMx_CCMRx 寄存器中的 OCxPE 位置 1 使能相应预装载寄存器，最后通过将 TIMx_CR1 寄存器中的 ARPE 位置 1 使能自动重载预装载寄存器（在递增计数或中心对齐模式下）。

由于只有在发生更新事件时预装载寄存器才会传送到影子寄存器，因此启动计数器之前，必须通过将 TIMx_EGR 寄存器中的 UG 位置 1 来初始化所有寄存器。

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。OCx 输出通过将 TIMx_CCER 寄存器中的 CCxE 位置 1 来使能。有关详细信息，请参见 TIMx_CCERx 寄存器说明。

在 PWM 模式（1 或 2）下，TIMx_CNT 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）。不过，为符合 OCREF_CLR 功能（在下一个 PWM 周期之前，ETR 信号上的一个外部事件能够清除 OCREF），OCREF 信号仅在以下情况下变为有效状态：

- 比较结果发生改变，或
- 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。

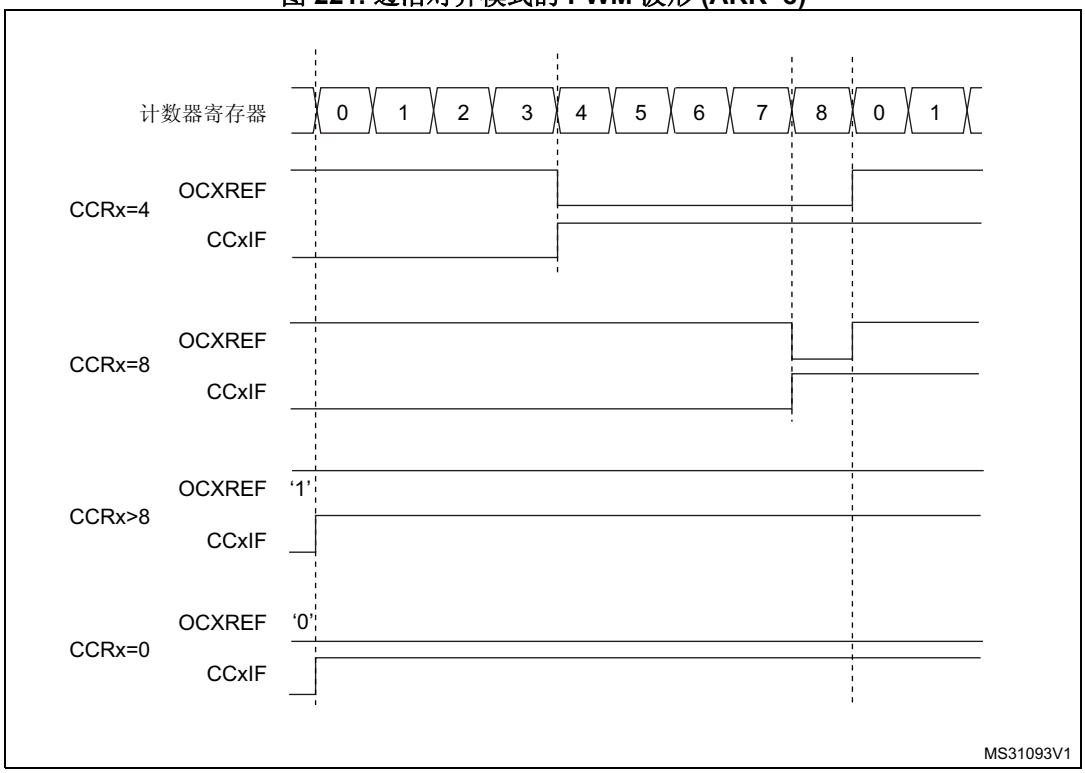
PWM 边沿对齐模式

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 654 页上的递增计数模式*。

以下以 PWM 模式 1 为例。只要 TIMx_CNT < TIMx_CCRx，PWM 参考信号 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 OCxREF 保持为“1”。如果比较值为 0，则 OCxREF 保持为“0”。*图 221* 举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

图 221. 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 657 页上的递减计数模式*。

在 PWM 模式 1 下，只要 TIMx_CNT>TIMx_CCRx，参考信号 ocxref 便为低电平，否则为高电平。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重载值，则 ocxref 保持为“1”。此模式下不可能产生 0% 的 PWM 波形。

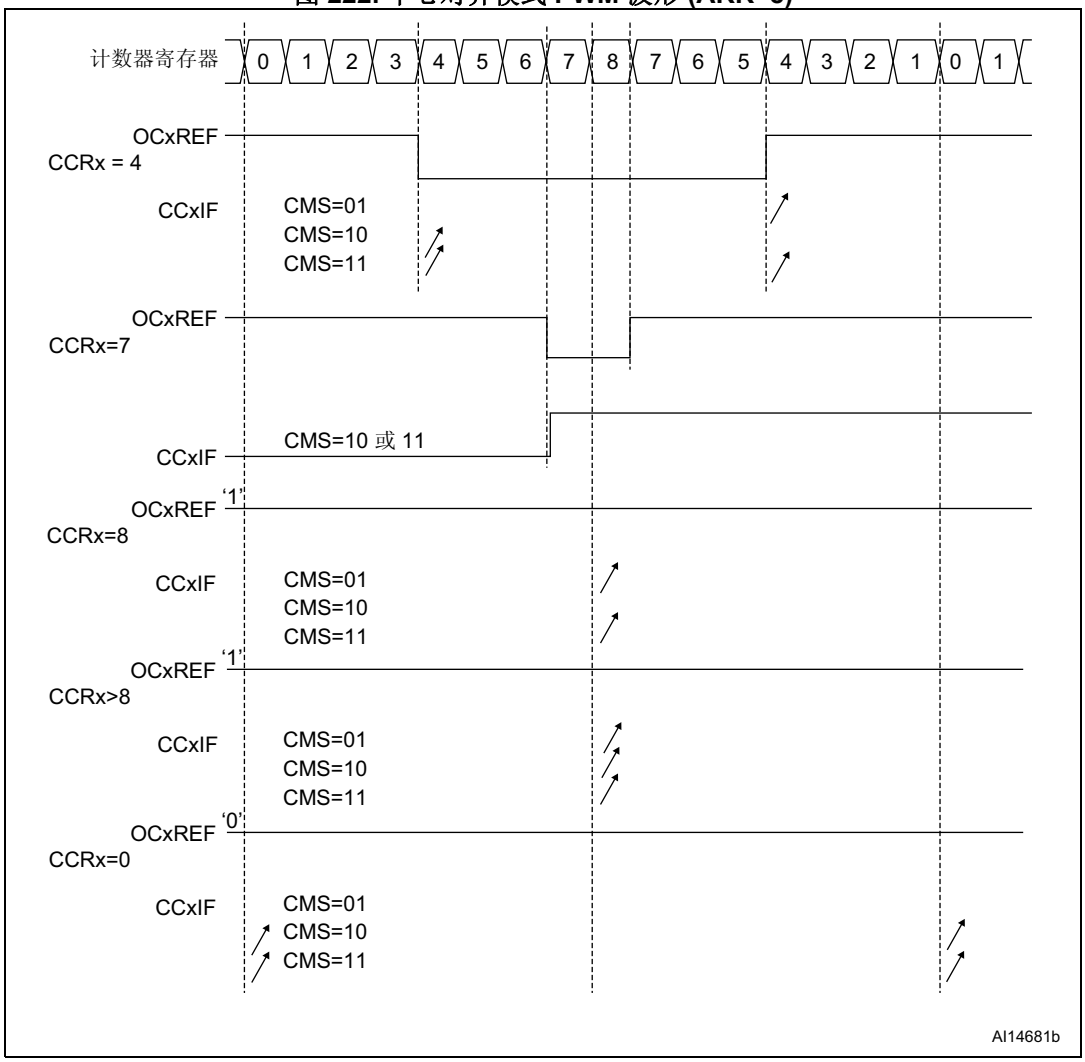
PWM 中心对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时（其余所有配置对 ocxref/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 660 页上的中心对齐模式（递增/递减计数）*。

图 222 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 222. 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，不得同时通过软件修改 DIR 和 CMS 位。
- 不建议在运行中心对齐模式时对计数器执行写操作，否则将发生意想不到的结果。尤其是：
  - 如果写入计数器中的值大于自动重载值 (TIMx_CNT>TIMx_ARR)，计数方向不会更新。例如，如果计数器之前递增计数，则继续递增计数。
  - 如果向计数器写入 0 或 TIMx_ARR 的值，计数方向会更新，但不生成更新事件 UEV。
- 使用中心对齐模式最为保险的方法是：在启动计数器前通过软件生成更新（将 TIMx_EGR 寄存器中的 UG 位置 1），并且不要在计数器运行过程中对其执行写操作。

23.3.10 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：

- OC1REFC（或 OC2REFC）由 TIMx_CCR1 和 TIMx_CCR2 控制
- OC3REFC（或 OC4REFC）由 TIMx_CCR3 和 TIMx_CCR4 控制

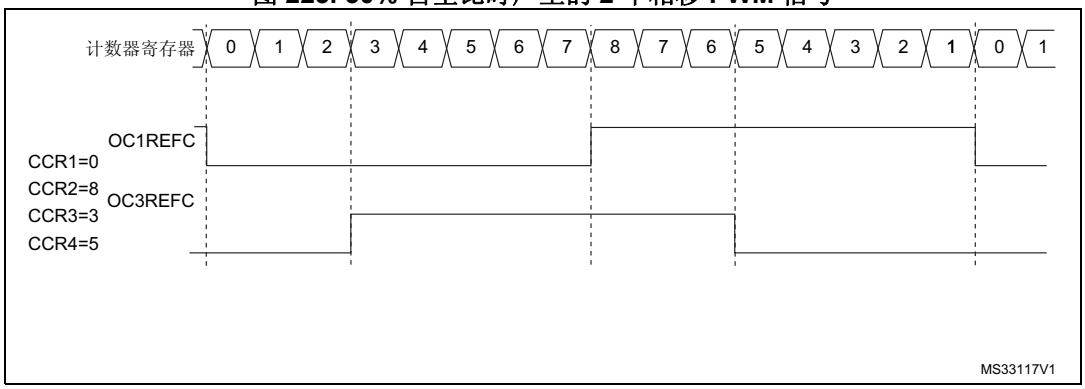
两个通道可以独立选择不对称 PWM 模式（每对 CCR 寄存器一个 OCx 输出），只需向 TIMx_CCMRx 寄存器的 OCxM 位写入“1110”（不对称 PWM 模式 1）或“1111”（不对称 PWM 模式 2）。

注： *出于兼容性原因，OCxM[3:0] 位字段分为两部分，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 OC1REFC 信号（不对称 PWM 模式 1），则由于不对称 PWM 模式 2 的原因，通道 2 上可输出 OC2REF 信号或 OC2REFC 信号。

图 223 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 在不对称 PWM 模式 1 下配置）。

图 223. 50% 占空比时产生的 2 个相移 PWM 信号



23.3.11 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 OCxREFC 由两个参考 PWM 的逻辑或运算或者逻辑与运算组合组成。

——OC1REFC（或 OC2REFC）由 TIMx_CCR1 和 TIMx_CCR2 控制

——OC3REFC（或 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 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）。

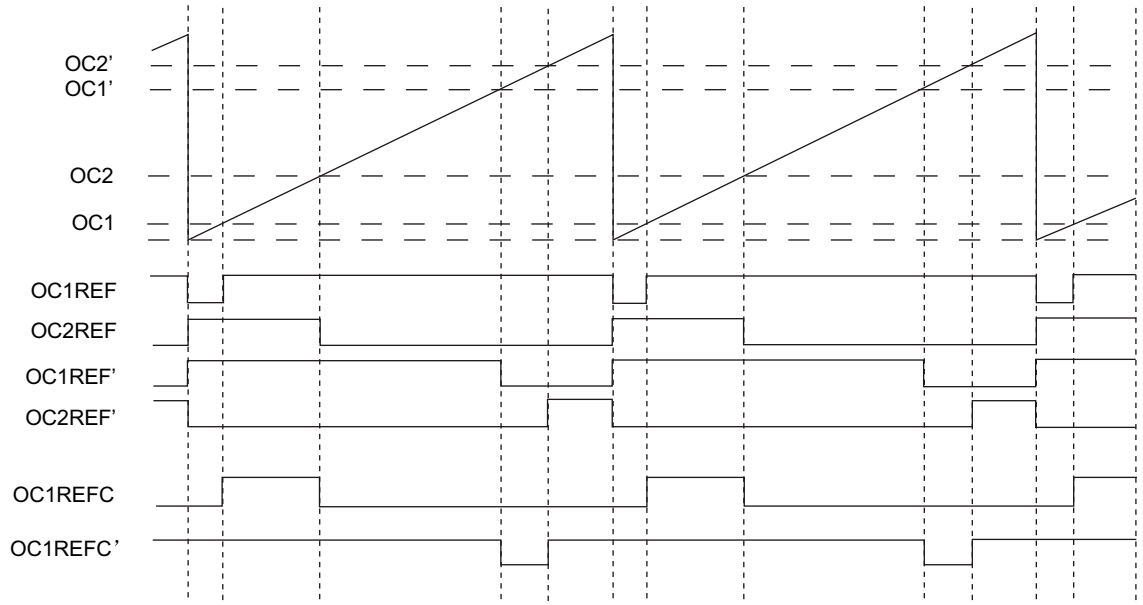
当给定通道用作组合 PWM 通道时，其辅助通道必须在相反的 PWM 模式下配置（例如，一个通道在组合 PWM 模式 1 下配置，另一个通道在组合 PWM 模式 2 下配置）。

注： *出于兼容性原因，OCxM[3:0] 位字段分为两部分，最高有效位与最低有效的 3 位不相邻。*

图 224 显示了不对称 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 在组合 PWM 模式 2 下配置，
- 通道 2 在 PWM 模式 1 下配置，
- 通道 3 在组合 PWM 模式 2 下配置，
- 通道 4 在 PWM 模式 1 下配置

图 224. 通道 1 和通道 3 上的组合 PWM 模式



23.3.12 发生外部事件时清除 OCxREF 信号

对于给定通道，在 OCREF_CLR_INPUT 施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置 1），可将 OCxREF 信号清零。OCxREF 信号将保持低电平，直到发生下一更新事件 (UEV)。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 OCREF_CLR 输入和 ETRF（滤波器后的 ETR）之间选择 OCREF_CLR_INPUT。

选择 ETRF 时，ETR 必须配置如下：

对于给定通道，在 ETRF 输入施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置 1），可将 OCxREF 信号复位。OCxREF 信号将保持低电平，直到发生下一更新事件 (UEV)。

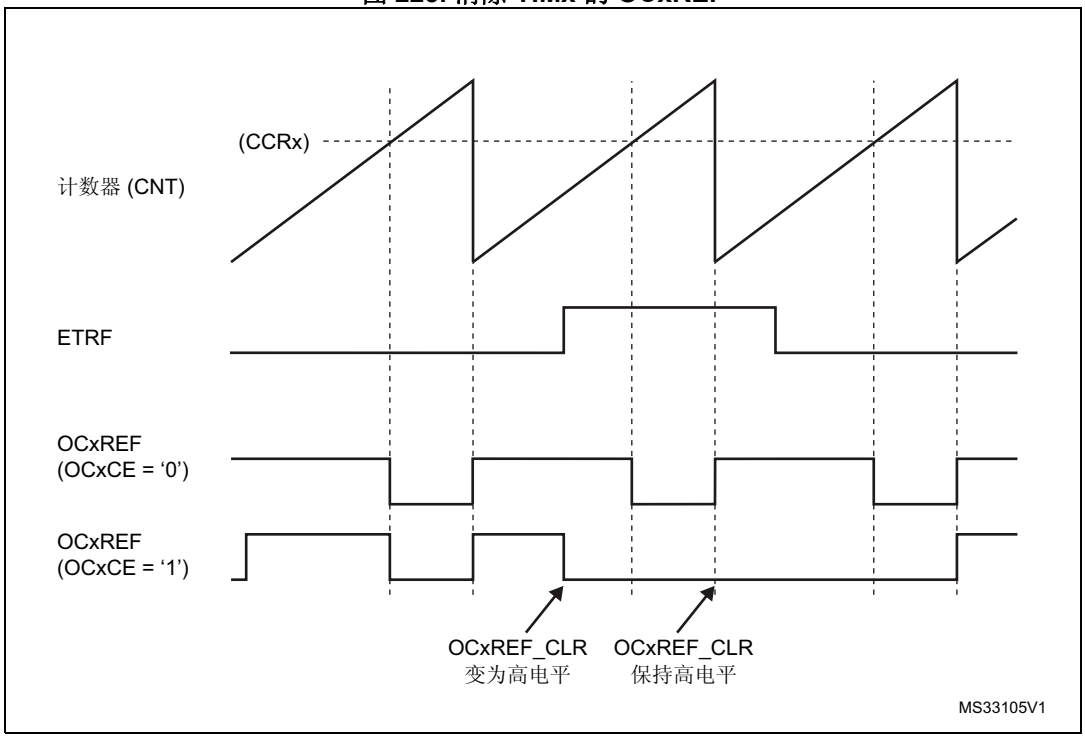
该功能只能在输出比较模式和 PWM 模式下使用。在强制模式下不起作用。

例如，OCxREF 信号可以连接到比较器的输出，用于控制电流。此时，ETR 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 225 对比了不同使能位 OCxCE 值的情况，显示了当 ETRF 输入变为高电平时 OCxREF 信号的行为。在本例中，定时器 TIMx 编程为 PWM 模式。

图 225. 清除 TIMx 的 OCxREF



注： *如果 PWM 的占空比为 100% (CCRx>ARR)，则下次计数器上溢时会再次使能 OCxREF。*

23.3.13 单脉冲模式

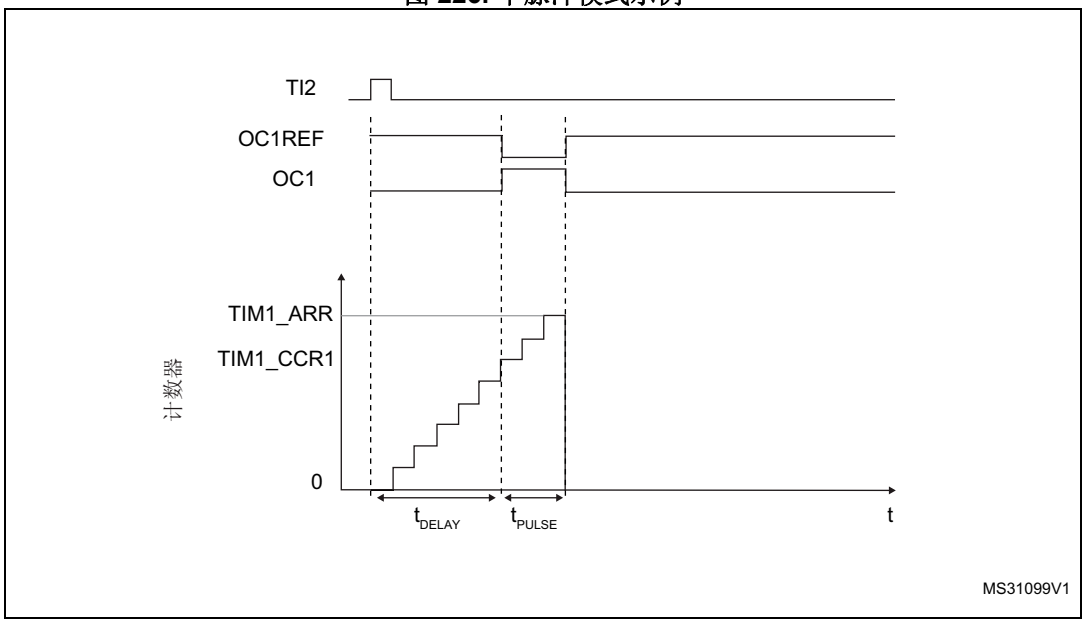
单脉冲模式 (OPM) 是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。将 TIMx_CR1 寄存器中的 OPM 位置 1，即可选择单脉冲模式。这样，发生下一更新事件 UEV 时，计数器将自动停止。

只有当比较值与计数器初始值不同时，才能正确产生一个脉冲。启动前（定时器等待触发时），必须进行如下配置：

- CNT<CCRx≤ARR（特别注意，0<CCRx），

图 226. 单脉冲模式示例



例如，用户希望达到这样的效果：在 TI2 输入引脚检测到上升沿时，经过 t_{DELAY} 的延迟，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 TI2FP2 作为触发 1：

- 在 TIMx_CCMR1 寄存器中写入 CC2S=01，将 TI2FP2 映射到 TI2。
- 在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=“0”，使 TI2FP2 能够检测上升沿。
- 在 TIMx_SMCR 寄存器中写入 TS=110，将 TI2FP2 配置为从模式控制器的触发 (TRGI)。
- 在 TIMx_SMCR 寄存器中写入 SMS=“110”（触发模式），使用 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，应在 TIMx_CCMR1 寄存器中写入 OC1M=111，以使能 PWM 模式 2。如果需要，可选择在 TIMx_CCMR1 寄存器的 OC1PE 和 TIMx_CR1 寄存器的 ARPE 中写入 1，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 TI2 上的外部触发事件。本例中，CC1P 的值为“0”。

在本例中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应为低。

由于仅需要 1 个脉冲（单脉冲模式），因此应向 TIMx_CR1 寄存器的 OPM 位写入“1”，以便在发生下一更新事件（计数器从自动重载值返回到 0）时使计数器停止计数。TIMx_CR1 寄存器中的 OPM 位置“0”时，即选择重复模式。

特殊情况：OCx 快速使能：

在单脉冲模式下，TIx 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时，将切换输出。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟（t_{DELAY} 最小值）。

如果要输出延迟时间最短的波形，可以将 TIMx_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 OCxRef（和 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。

23.3.14 编码器接口模式

选择编码器接口模式时，如果计数器仅在 TI2 边沿处计数，在 TIMx_SMCR 寄存器中写入 SMS=001；如果计数器仅在 TI1 边沿处计数，写入 SMS=010；如果计数器在 TI1 和 TI2 边沿处均计数，则写入 SMS=011。

通过编程 TIMx_CCER 寄存器的 CC1P 和 CC2P 位，选择 TI1 和 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。CC1NP 和 CC2NP 必须保持低电平。

TI1 和 TI2 两个输入用于连接增量编码器。请参见 *表 129*。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 TI1FP1 或 TI2FP2 上的每次有效信号转换提供。TI1FP1 和 TI2FP2 是进行输入滤波器和极性选择后 TI1 和 TI2 的信号，如果不进行滤波和反相，则 TI1FP1=TI1，TI2FP2=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（TI1 或 TI2）发生信号转换时，都会计算 DIR 位，无论计数器是仅在 TI1 或 TI2 边沿处计数，还是同时在 TI1 和 TI2 处计数。

编码器接口模式就相当于带有方向选择的外部时钟。这意味着，计数器仅在 0 到 TIMx_ARR 寄存器中的自动重载值之间进行连续计数（根据具体方向，从 0 递增计数到 ARR，或从 ARR 递减计数到 0）。因此，在启动前必须先配置 TIMx_ARR。同样，捕获、比较、预分频器、触发输出功能继续正常工作。

在此模式下，计数器会根据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 TI1 和 TI2 不同时切换）。

表 129. 计数方向与编码器信号的关系

有效边沿	相反信号的电平（TI1FP1 对应 TI2，TI2FP2 对应 TI1）	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 处计数	高	递减	递增	不计数	不计数
	低	递增	递减	不计数	不计数
仅在 TI2 处计数	高	不计数	不计数	递增	递减
	低	不计数	不计数	递减	递增
在 TI1 和 TI2 处均计数	高	递减	递增	递增	递减
	低	递增	递减	递减	递增

外部增量编码器可直接与 MCU 相连，无需外部接口逻辑。不过，通常使用比较器将编码器的差分输出转换为数字信号。这样大幅提高了抗噪声性能。用于指示机械零位的第三个编码器输出可与外部中断输入相连，用以触发计数器复位。

图 227 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，TI1FP1 映射到 TI1 上）
- CC2S=01（TIMx_CCMR2 寄存器，TI2FP2 映射到 TI2 上）
- CC1P 和 CC1NP =“0”（TIMx_CCER 寄存器，TI1FP1 未反相，TI1FP1=TI1）
- CC2P 和 CC2NP =“0”（TIMx_CCER 寄存器，TI2FP2 未反相，TI2FP2= TI2）
- SMS=011（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 227. 编码器接口模式下的计数器工作示例

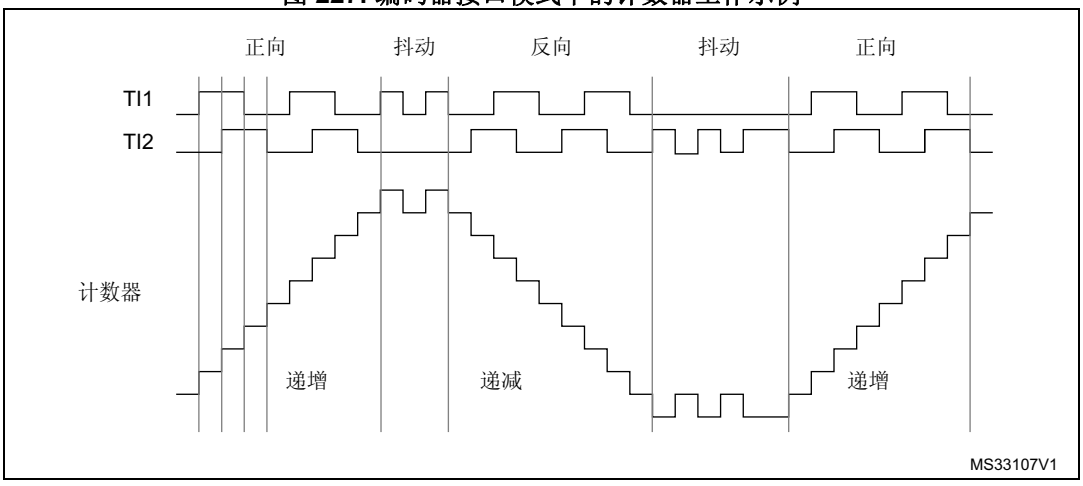
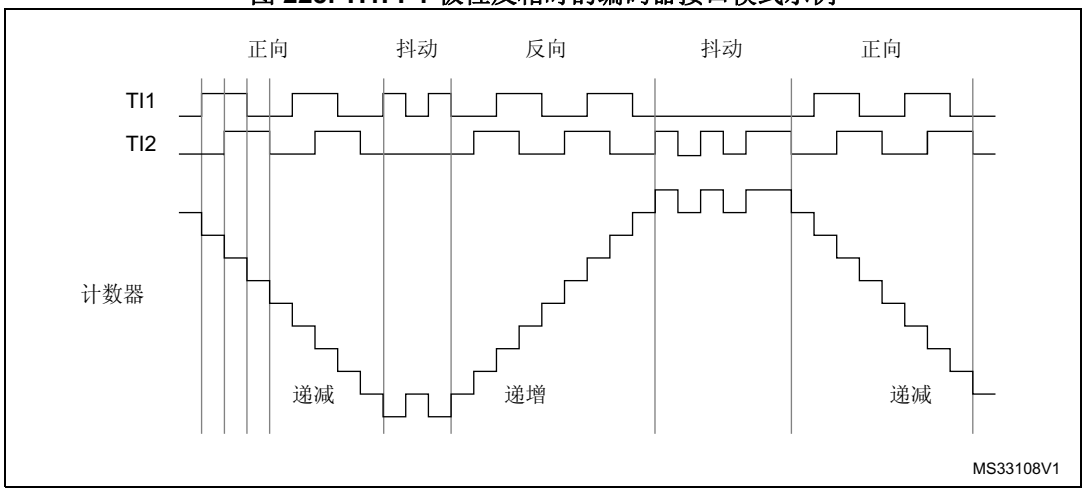


图 228 说明了 TI1FP1 极性反相时的计数器行为示例（除 CC1P=1 外，其它配置与上例相同）。

图 228. TI1FP1 极性反相时的编码器接口模式示例



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）； 此外，还可以通过实时时钟产生的 DMA 请求读取计数器值。

23.3.15 UIF 位重映射

TIMx_CR1 寄存器中的 IUFREMAP 位强制将更新中断标志 (UIF) 连续复制到定时计数器寄存器的位 31 (TIMxCNT[31]) 中。这样便可自动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 标志使能之间没有延迟。

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）。

23.3.16 定时器输入异或功能

借助 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 TIMx_CH1 到 TIMx_CH3 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 610 页上的第 22.3.24 节：连接霍尔传感器*以连接霍尔传感器为例介绍了此功能。

23.3.17 定时器与外部触发同步

TIMx 定时器可与外部触发以下列模式实现同步：复位模式、门控模式和触发模式。

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。

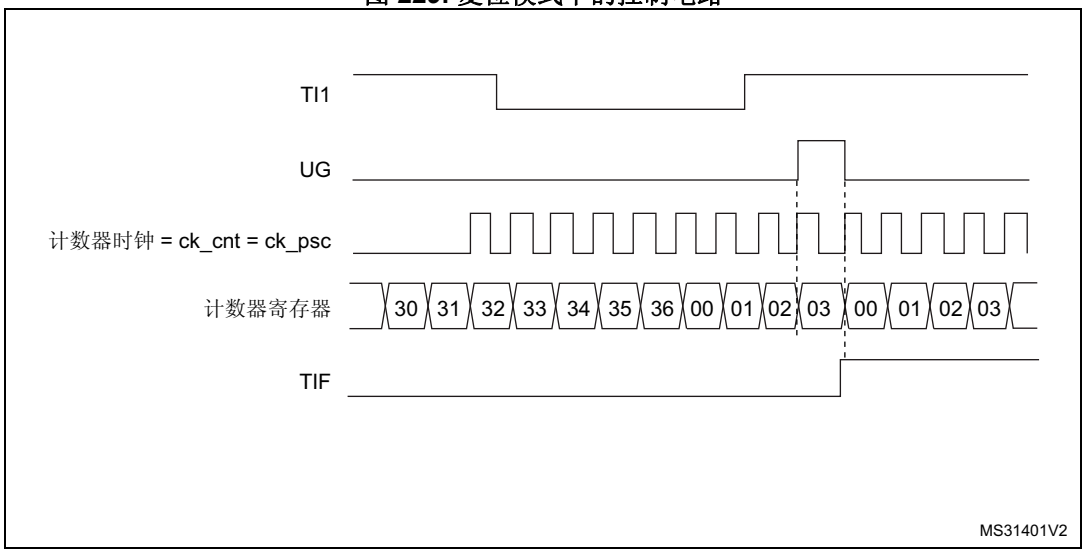
在以下示例中，TI1 输入上出现上升沿时，递增计数器清零：

1. 将通道 1 配置为检测 TI1 的上升沿。配置输入滤波带宽（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=101，选择 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 TI1 上升沿。当 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。TI1 的上升沿与实际计数器复位之间的延迟是由于 TI1 输入的重新同步电路引起的。

图 229. 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。

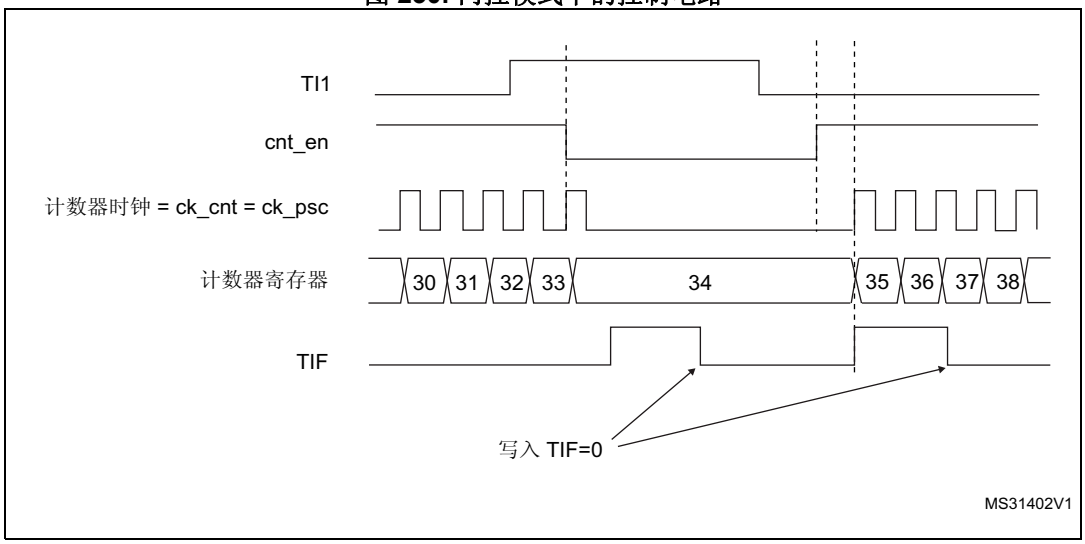
在以下示例中，递增计数器仅在 TI1 输入为低电平时计数：

1. 将通道 1 配置为检测 TI1 上的低电平。配置输入滤波带宽（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=101，选择 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 TI1 为低电平，计数器就开始根据内部时钟计数，直到 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。

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

图 230. 门控模式下的控制电路



1. 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。

注： *由于门控模式作用于电平而非边沿，因此在门控模式下，“CCxP=CCxNP=1”（同时检测上升沿和下降沿）不发挥任何作用。*

从模式：触发模式

所选输入上发生某一事件时可以启动计数器。

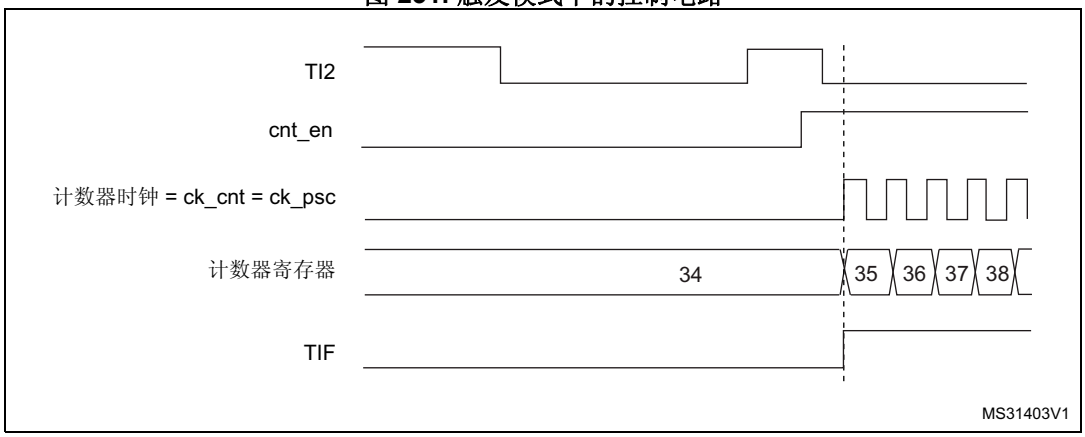
在以下示例中，TI2 输入上出现上升沿时，递增计数器启动：

1. 将通道 2 配置为检测 TI2 上的上升沿。配置输入滤波带宽（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）。
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=110，选择 TI2 作为输入源。

当 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。

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

图 231. 触发模式下的控制电路



从模式：外部时钟模式 2 + 触发模式

外部时钟模式 2 可与另一种从模式（外部时钟模式 1 和编码器模式除外）结合使用。这种情况下，ETR 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 ETR 作为 TRGI。

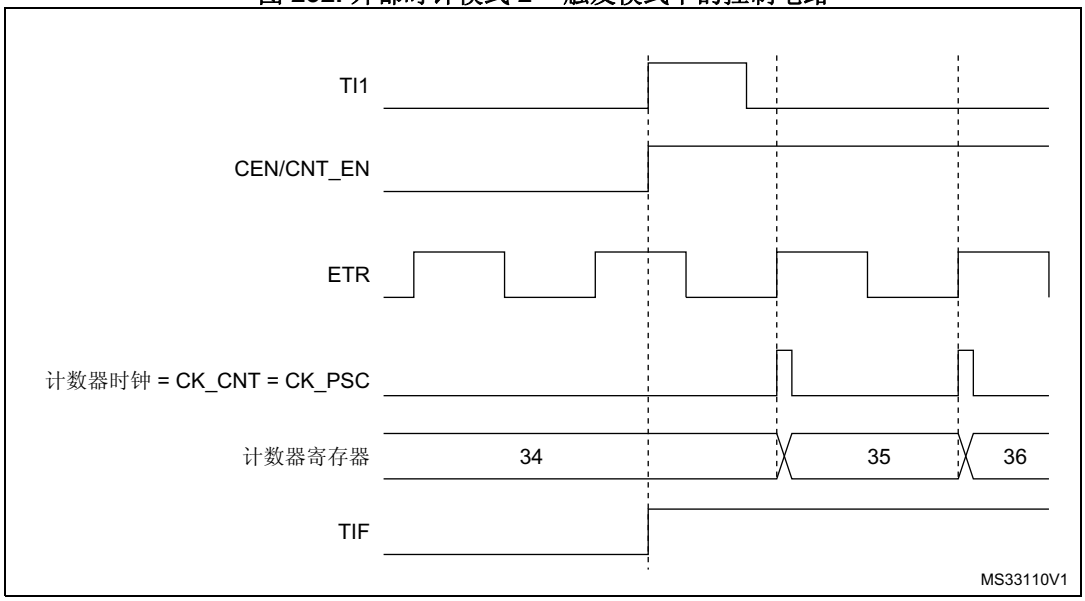
在以下示例中，只要 TI1 出现上升沿，递增计数器即会在 ETR 信号的每个上升沿处递增：

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 ETR 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=101，选择 TI1 作为输入源。

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 ETR 出现上升沿时计数。

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

图 232. 外部时钟模式 2 + 触发模式下的控制电路



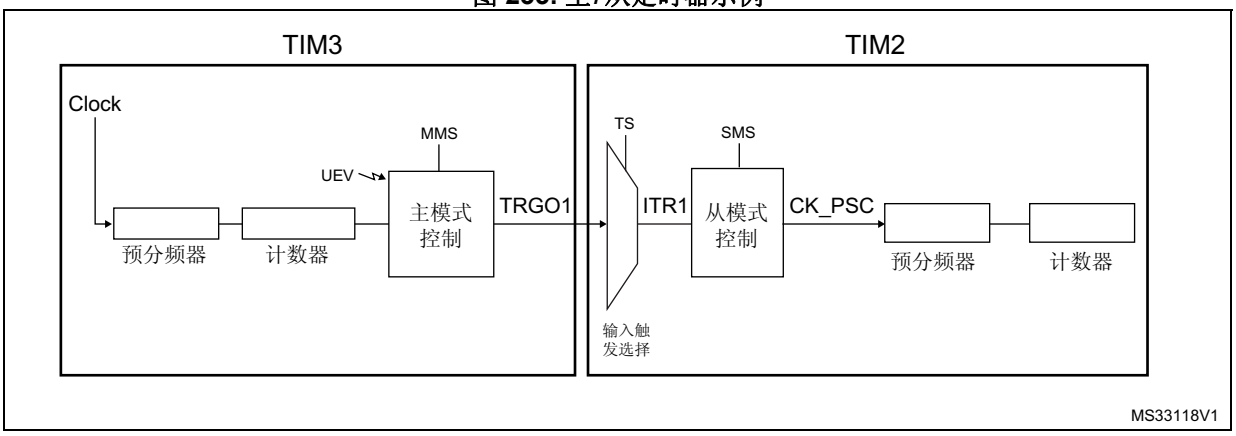
23.3.18 定时器同步

TIMx 定时器从内部连接在一起，以实现定时器同步或链接。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 233：主/从定时器示例*简要介绍了触发选择和主模式选择框图。

将一个定时器用作另一个定时器的预分频器

图 233. 主/从定时器示例



例如，可以将 TIM3 配置为 TIM 的预分频器。请参见*图 233*。为此：

1. 将 TIM3 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 TIM3_CR2 寄存器中写入 MMS=010，则每次生成更新事件时，TRGO1 都会输出一个上升沿。
2. 要将 TIM3 的 TRGO1 输出连接到 TIM，必须将 TIM 配置为从模式，使用 ITR2 作为内部触发。通过 TIM_SMCR 寄存器中的 TS 位（写入 TS=010）可对此进行选择。
3. 然后将从模式控制器设为外部时钟模式 1（在 TIM_SMCR 寄存器中写入 SMS=111）。这样一来，TIM 的时钟将由 TIM3 周期性触发信号的上升沿（与 TIM3 的计数器上溢对应）提供。
4. 最后必须通过将这两个定时器的相应 CEN 位（TIMx_CR1 寄存器）置 1 同时使能二者。

注： *如果选择 TIM3 的 OCx 信号作为触发输出 (MMS=1xx)，该信号的上升沿将用于驱动 TIM 的计数器。*

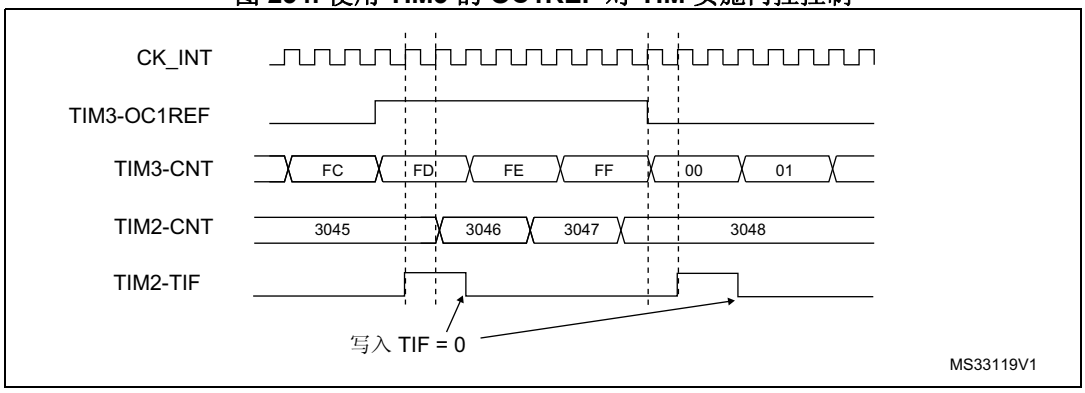
使用一个定时器使能另一个定时器

本例中通过定时器 3 的输出比较 1 来控制 TIM 的使能。相关连接图，请参见*图 233*。仅当 TIM3 的 OC1REF 为高电平时，TIM 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 CK_INT 通过预分频器执行 3 分频 ($f_{CK_CNT} = f_{CK_INT}/3$)。

1. 将 TIM3 配置为主模式，发送其输出比较 1 参考信号 (OC1REF) 作为触发输出（TIM3_CR2 寄存器中的 MMS=100）。
2. 配置 TIM3 的 OC1REF 波形（TIM3_CCMR1 寄存器）。
3. 配置 TIM 以接收来自 TIM3 的输入触发（TIM_SMCR 寄存器中的 TS=010）。
4. 将 TIM 配置为门控模式（TIM_SMCR 寄存器中的 SMS=101）。
5. 通过向 CEN 位（TIM_CR1 寄存器）写入“1”使能 TIM。
6. 通过向 CEN 位（TIM3_CR1 寄存器）写入“1”启动 TIM3。

注： *计数器的时钟与计数器 1 不同步，此模式仅影响 TIM 的计数器使能信号。*

图 234. 使用 TIM3 的 OC1REF 对 TIM 实施门控控制

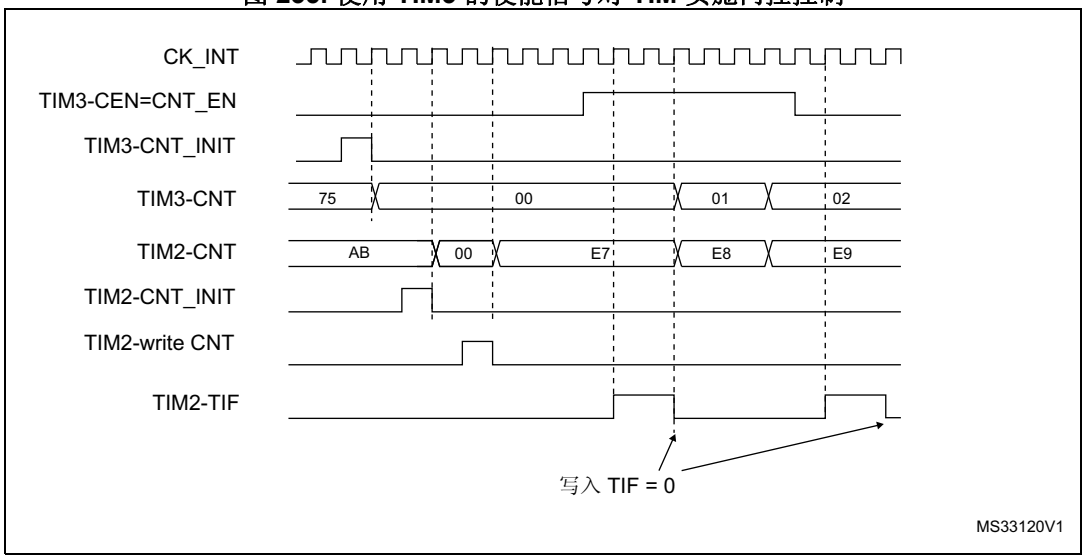


在*图 234* 的示例中，TIM 的计数器和预分频器在启动前未进行初始化。因此从各自的当前值开始计数。启动定时器 TIM3 之前，通过复位这两个定时器可以从指定值开始计数。这样便可以在定时器计数器中写入所需的任意值。两个定时器都可通过软件使用 TIMx_EGR 寄存器中的 UG 位轻松复位。

在下一示例中，TIM3 与 TIM 同步。TIM3 为主模式，从 0 开始计数。TIM 为从模式，从 0xE7 开始计数。两个定时器的预分频比相同。通过向 TIM3_CR1 寄存器中的 CEN 位写入“0”来禁止 TIM3 时，TIM 将停止：

1. 将 TIM3 配置为主模式，发送其输出比较 1 参考信号 (OC1REF) 作为触发输出（TIM3_CR2 寄存器中的 MMS=100）。
2. 配置 TIM3 的 OC1REF 波形（TIM3_CCMR1 寄存器）。
3. 配置 TIM 以接收来自 TIM3 的输入触发（TIM_SMCR 寄存器中的 TS=010）。
4. 将 TIM 配置为门控模式（TIM_SMCR 寄存器中的 SMS=101）。
5. 通过向 UG 位（TIM3_EGR 寄存器）写入“1”复位 TIM3。
6. 通过向 UG 位（TIM_EGR 寄存器）写入“1”复位 TIM。
7. 通过在 TIM 的计数器 (TIM_CNTL) 中写入“0xE7”使 TIM 初始化为 0xE7。
8. 通过向 CEN 位（TIM_CR1 寄存器）写入“1”使能 TIM。
9. 通过向 CEN 位（TIM3_CR1 寄存器）写入“1”启动 TIM3。
10. 通过向 CEN 位（TIM3_CR1 寄存器）写入“0”停止 TIM3。

图 235. 使用 TIM3 的使能信号对 TIM 实施门控控制

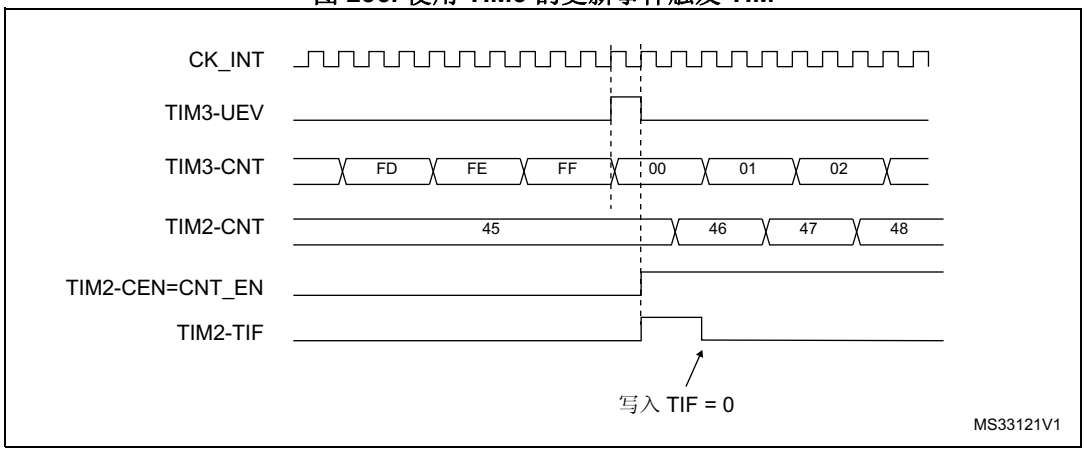


使用一个定时器启动另一个定时器

本例中使用定时器定时器 3 的更新事件使能定时器。相关连接图，请参见*图 233*。只要定时器 1 生成更新事件，定时器便根据分频后的内部时钟从当前值（可以不为 0）开始计数。定时器收到触发信号时，其 CEN 位自动置 1，并且计数器开始计数，直到向 TIM_CR1 寄存器的 CEN 位写入“0”后停止计数。两个计数器的时钟频率都基于 CK_INT 通过预分频器执行 3 分频 ($f_{CK_CNT} = f_{CK_INT}/3$)。

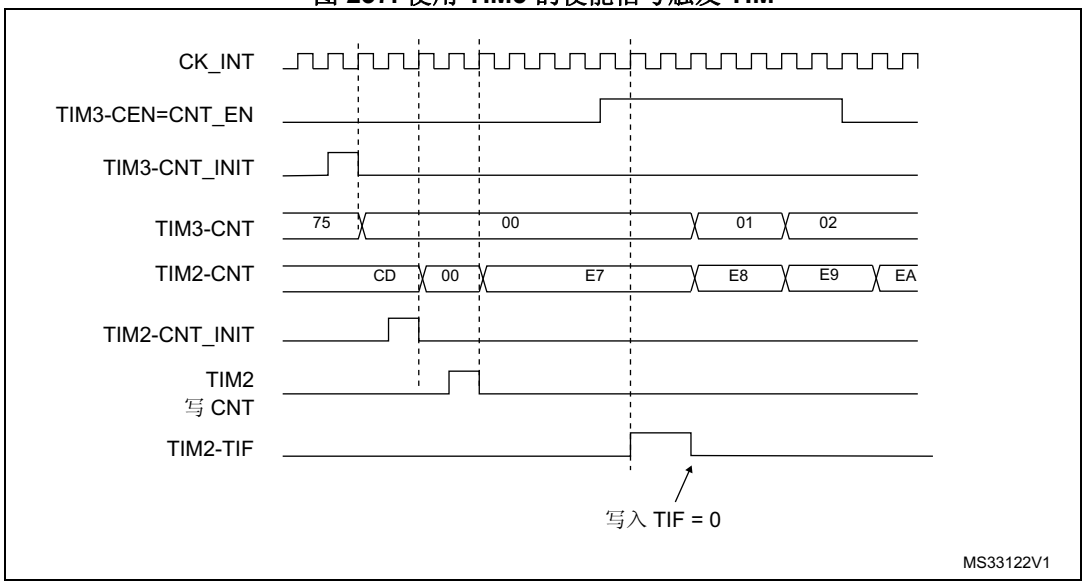
1. 将 TIM3 配置为主模式，发送其更新事件 (UEV) 作为触发输出（TIM3_CR2 寄存器中的 MMS=010）。
2. 配置 TIM3 的周期（TIM3_ARR 寄存器）。
3. 配置 TIM 以接收来自 TIM3 的输入触发（TIM_SMCR 寄存器中的 TS=010）。
4. 将 TIM 配置为触发模式（TIM_SMCR 寄存器中的 SMS=110）。
5. 通过向 CEN 位（TIM3_CR1 寄存器）写入“1”启动 TIM3。

图 236. 使用 TIM3 的更新事件触发 TIM



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 237* 显示了与*图 236* 具有相同配置，只不过处于触发模式（TIM_SMCR 寄存器中的 SMS=110）而非门控模式的计数行为。

图 237. 使用 TIM3 的使能信号触发 TIM



使用一个外部触发同步的启动 2 个定时器

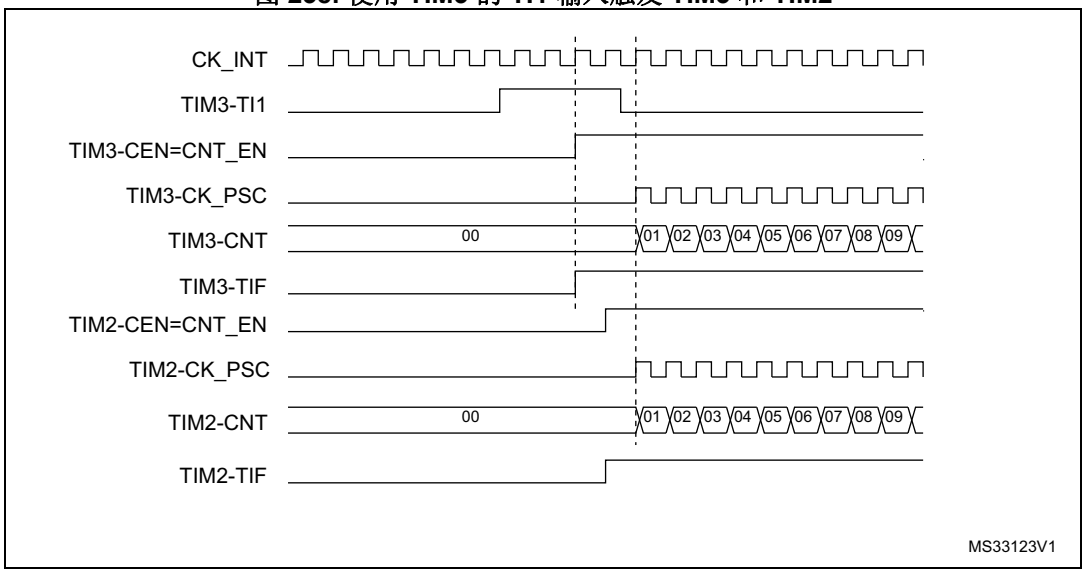
本例中，TIM3 的 TI1 输入出现上升沿时使能 TIM3，使能 TIM3 的同时使能 TIM2。相关连接图，请参见*图 233*。要确保计数器对齐，TIM3 必须配置为主/从模式（对应的 TI1 为从，对应 TIM2 为主）：

1. 将 TIM3 配置为主模式，发送其使能信号作为触发输出（TIM3_CR2 寄存器中的 MMS=001）。
2. 将 TIM3 配置为从模式以接收来自 TI1 的输入触发（TIM3_SMCR 寄存器中的 TS=100）。
3. 将 TIM3 配置为触发模式（TIM3_SMCR 寄存器中的 SMS=110）。
4. 通过写入 MSM=1（TIM3_SMCR 寄存器）将 TIM3 配置为主/从模式。
5. 配置 TIM2 以接收来自 TIM3 的输入触发（TIM2_SMCR 寄存器中的 TS=000）。
6. 将 TIM2 配置为触发模式（TIM2_SMCR 寄存器中的 SMS=110）。

当 TI1 (TIM3) 出现上升沿时，两个计数器开始根据内部时钟同步计数，并且两个 TIF 标志都置 1。

注： *本例中，两个定时器都在启动之前进行了初始化（通过将各自的 UG 位置 1）。两个计数器都从 0 开始计数，但可以通过对任意一个计数器寄存器 (TIMx_CNT) 进行写操作，在二者之间轻松插入一个偏移量。可注意到主/从模式在 TIM3 的 CNT_EN 与 CK_PSC 之间产生了延迟。*

图 238. 使用 TIM3 的 TI1 输入触发 TIM3 和 TIM2



注： *从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。*

23.3.19 DMA 连续传送模式

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，但也可用于定期读取一行中的多个寄存器。

DMA 控制器目标唯一，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）。每次写入 TIMx_DMAR 寄存器都会重定向到其中一个定时器寄存器。

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，定时器进行一次连续传送，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

例如，定时器 DMA 连续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 位字段配置如下来配置 DCR 寄存器：
   DBL = 3 次传输，DBA = 0xE。
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，则要传输的数据量应为 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。

23.3.20 调试模式

当微控制器进入调试模式时（Cortex®-M7 内核停止），TIMx 计数器会根据 DBGMCU 模块中的 DBG_TIMx_STOP 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*第 40.16.2 节：对定时器、看门狗、bxCAN 和 I^2C 的调试支持*。

23.4 TIM2/TIM3/TIM4/TIM5 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.1 节*。

外设寄存器可支持半字（16 位）或字（32 位）访问。

23.4.1 TIMx 控制寄存器 1 (TIMx_CR1)

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.	Res.	UIF RE-MAP	Res.	CKD[1:0]		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
				rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15:12 保留，必须保持复位值。

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**：时钟分频 (Clock division)

此位字段指示定时器时钟 (CK_INT) 频率与数字滤波器所使用的采样时钟（ETR、TIx）之间的分频比，

00：$t_{DTS} = t_{CK_INT}$
01：$t_{DTS} = 2 \times t_{CK_INT}$
10：$t_{DTS} = 4 \times t_{CK_INT}$
11：保留

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**：中心对齐模式选择 (Center-aligned mode selection)

00：边沿对齐模式。计数器根据方向位 (DIR) 递增计数或递减计数。
01：中心对齐模式 1。计数器交替进行递增计数和递减计数。仅当计数器递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志才置 1。
10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志才置 1。
11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志都会置 1。

注：　只要计数器处于使能状态 (CEN=1)，就不得从边沿对齐模式切换为中心对齐模式。

位 4 **DIR**：方向 (Direction)

0：计数器递增计数
1：计数器递减计数

注：　当定时器配置为中心对齐模式或编码器模式时，该位为只读状态。

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)

此位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能时，所有以下事件都会产生更新中断或 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)

此位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新 (UEV) 事件可通过以下事件之一产生：
- 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

然后更新影子寄存器的值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

位 0 **CEN**：计数器使能 (Counter enable)

0：禁止计数器
1：使能计数器

注：　只有事先通过软件将 CEN 位置 1，才可以使用外部时钟、门控模式和编码器模式。而触发模式可通过硬件自动将 CEN 位置 1。

在单脉冲模式下，当发生更新事件时会自动将 CEN 位清零。

23.4.2 TIMx 控制寄存器 2 (TIMx_CR2)

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.	TI1S	MMS[2:0]			CCDS	Res.	Res.	Res.
								rw	rw	rw	rw	rw			

位 15:8 保留，必须保持复位值。

位 7 **TI1S**：TI1 选择 (TI1 selection)

0：TIMx_CH1 引脚连接到 TI1 输入

1：TIMx_CH1、CH2 和 CH3 引脚连接到 TI1 输入（异或组合）。另请参见*第 610 页上的第 22.3.24 节：连接霍尔传感器*

位 6:4 **MMS**：主模式选择 (Master mode selection)

这些位可选择主模式下将要发送到从定时器以实现同步的信息 (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 (TRGO)。如果复位由触发输入生成（从模式控制器配置为复位模式），则 TRGO 上的信号相比实际复位会有延迟。

001：**使能**——计数器使能信号 CNT_EN 用作触发输出 (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或运算组合而成。

当计数器使能信号由触发输入控制时，TRGO 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中 MSM 位的说明）。

010：**更新**——选择更新事件作为触发输出 (TRGO)。例如，主定时器可用作从定时器的预分频器。

011：**比较脉冲**——一旦发生输入捕获或比较匹配事件，当 CC1IF 标志被置 1 时（即使已为高电平），触发输出都会发送一个正脉冲。(TRGO)

100：**比较**——OC1REF 信号用作触发输出 (TRGO)

101：**比较**——OC2REF 信号用作触发输出 (TRGO)

110：**比较**——OC3REF 信号用作触发输出 (TRGO)

111：**比较**——OC4REF 信号用作触发输出 (TRGO)

注： 从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。

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 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。

23.4.3 TIMx 从模式控制寄存器 (TIMx_SMCR)

TIMx slave mode control register

偏移地址：0x08

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMS[3]
															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 31:17 保留，必须保持复位值。

位 16 **SMS[3]**：从模式选择——位 3 (Slave mode selection - bit 3)
请参见 SMS 说明——位 2:0

位 15 **ETP**：外部触发极性 (External trigger polarity)
此位可选择将 ETR 还是 $\overline{ETR}$ 用于触发操作。
0：ETR 未反相，高电平或上升沿有效。
1：ETR 反相，低电平或下降沿有效。

位 14 **ECE**：外部时钟使能 (External clock enable)
此位可使能外部时钟模式 2。
0：禁止外部时钟模式 2。
1：使能外部时钟模式 2。计数器时钟由 ETRF 信号的任意有效边沿提供。
1：将 ECE 位置 1 与选择外部时钟模式 1 并将 TRGI 连接到 ETRF（SMS=111 且 TS=111）具有相同效果。
2：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 TRGI 不得连接 ETRF（TS 位不得为 111）。
3：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 ETRF。

位 13:12 **ETPS[1:0]**：外部触发预分频器 (External trigger prescaler)
外部触发信号 ETRP 频率不得超过 CK_INT 频率的 1/4。可通过使能预分频器来降低 ETRP 频率。这种方法在输入快速外部时钟时非常有用。
00：预分频器关闭。
01：2 分频 ETRP 频率。
10：4 分频 ETRP 频率。
11：8 分频 ETRP 频率。

位 11:8 **ETF[3:0]**：外部触发滤波器 (External trigger filter)

此位字段可定义 ETRP 信号的采样频率和适用于 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：$f_{SAMPLING}=f_{CK_INT}$，N=2
0010：$f_{SAMPLING}=f_{CK_INT}$，N=4
0011：$f_{SAMPLING}=f_{CK_INT}$，N=8
0100：$f_{SAMPLING}=f_{DTS}/2$，N=6
0101：$f_{SAMPLING}=f_{DTS}/2$，N=8
0110：$f_{SAMPLING}=f_{DTS}/4$，N=6
0111：$f_{SAMPLING}=f_{DTS}/4$，N=8
1000：$f_{SAMPLING}=f_{DTS}/8$，N=6
1001：$f_{SAMPLING}=f_{DTS}/8$，N=8
1010：$f_{SAMPLING}=f_{DTS}/16$，N=5
1011：$f_{SAMPLING}=f_{DTS}/16$，N=6
1100：$f_{SAMPLING}=f_{DTS}/16$，N=8
1101：$f_{SAMPLING}=f_{DTS}/32$，N=5
1110：$f_{SAMPLING}=f_{DTS}/32$，N=6
1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 7 **MSM**：主/从模式 (Master/slave mode)

0：不执行任何操作
1：当前定时器的触发输入事件 (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 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**：触发选择 (Trigger selection)

此位字段可选择将要用于同步计数器的触发输入。

000：内部触发 0 (ITR0)。保留
001：内部触发 1 (ITR1)
010：内部触发 2 (ITR2)
011：内部触发 3 (ITR3)。保留
100：TI1 边沿检测器 (TI1F_ED)
101：滤波后的定时器输入 1 (TI1FP1)
110：滤波后的定时器输入 2 (TI2FP2)
111：外部触发输入 (ETRF)

有关各定时器 ITRx 含义的详细信息，请参见*第 697 页上的表 130：TIMx 内部触发连接*。

注： 这些位只能在未使用的情况下（例如，SMS=000 时）进行更改，以避免转换时出现错误的边沿检测。

位 3 **OCCS**：OCREF 清零选择 (OCREF clear selection)

该位用于选择 OCREF 清零源

0：OCREF_CLR_INT 连接到 OCREF_CLR 输入
1：OCREF_CLR_INT 连接到 ETRF

位 2:0 **SMS**：从模式选择 (Slave mode selection)

选择外部信号时，触发信号 (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 TI1FP2 电平在 TI2FP1 边沿递增/递减计数。
0010：编码器模式 2——计数器根据 TI2FP1 电平在 TI1FP2 边沿递增/递减计数。
0011：编码器模式 3——计数器在 TI1FP1 和 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 (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 (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 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 (TRGI) 的上升沿提供计数器时钟。
1000：组合复位 + 触发模式——在出现所选触发输入 (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。

注：　如果将 TI1F_ED 选作触发输入 (TS=100)，则不得使用门控模式。实际上，TI1F 每次转换时，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：　从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。

表 130. TIMx 内部触发连接

从 TIM	ITR0 (TS = 000)	ITR1 (TS = 001)	ITR2 (TS = 010)	ITR3 (TS = 011)
TIM2	TIM1	TIM8/OTG_FS SOF(1)	TIM3	TIM4
TIM3	TIM1	TIM2	TIM15	TIM4
TIM4	TIM1	TIM2	TIM3	TIM8
TIM5	TIM2	TIM3	TIM4	TIM8

1. 取决于 TIM2_OR1 寄存器中的 ITR1_RMP 位。

23.4.4 TIMx DMA/中断使能寄存器 (TIMx_DIER)

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.	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 15 保留，必须保持复位值。

位 14 **TDE**：触发 DMA 请求使能 (Trigger DMA request enable)
0：禁止触发 DMA 请求。
1：使能触发 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**：触发中断使能 (Trigger interrupt enable)
0：禁止触发中断。
1：使能触发中断。

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。

23.4.5 TIMx 状态寄存器 (TIMx_SR)

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.	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 15:13 保留，必须保持复位值。

位 12 **CC4OF**：捕获/比较 4 重复捕获标志 (Capture/Compare 4 overcapture flag)
请参见 CC1OF 说明。

位 11 **CC3OF**：捕获/比较 3 重复捕获标志 (Capture/Compare 3 overcapture flag)
请参见 CC1OF 说明。

位 10 **CC2OF**：捕获/比较 2 重复捕获标志 (Capture/compare 2 overcapture flag)
请参见 CC1OF 说明。

位 9 **CC1OF**：捕获/比较 1 重复捕获标志 (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：未检测到重复捕获。
1：TIMx_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。

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 TRGI 输入上检测到有效边沿时，该标志将由硬件置 1。选择门控模式时，
该标志将在计数器启动或停止时置 1。但需要通过软件清零。
0：未发生触发事件。
1：触发中断挂起。

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)
如果通道 CC1 配置为输出：当计数器与比较值匹配时，此标志由硬件置 1，中心对齐模式（请参见 TIMx_CR1 寄存器中的 CMS 位说明）和可再触发单脉冲模式下除外。但需要通过软件清零。
0：不匹配。
1：TIMx_CNT 计数器的内容与 TIMx_CCR1 寄存器的内容匹配。
如果通道 CC1 配置为输入：此位将在发生捕获事件时由硬件置 1。通过软件或读取 TIMx_CCR1 寄存器将该位清零。
0：未发生输入捕获事件。
1：TIMx_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）。

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（对于 TIM2 到 TIM4）以及当 TIMx_CR1 寄存器中的 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）。

23.4.6 TIMx 事件生成寄存器 (TIMx_EGR)

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.	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)
此位由软件置 1 以生成事件，并由硬件自动清零。
0：不执行任何操作
1：TIMx_SR 寄存器中的 TIF 标志置 1。使能后可发生相关中断或 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)
此位由软件置 1 以生成事件，并由硬件自动清零。
0：不执行任何操作
1：通道 1 上生成捕获/比较事件：
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：不执行任何操作
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)。

23.4.7 TIMx 捕获/比较模式寄存器 1 (TIMx_CCMR1)

TIMx capture/compare mode register 1

偏移地址：0x18

复位值：0x0000

这些通道可用于输入（捕获模式）或输出（比较模式）模式。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其它位在输入模式和输出模式下的功能均不同。对于任一给定位，OCxx 用于说明通道配置为输出时该位对应的功能，ICxx 则用于说明通道配置为输入时该位对应的功能。因此，必须注意同一个位在输入阶段和输出阶段具有不同的含义。

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]
							Res.								Res.
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[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，始终读为 0。

位 24 **OC2M[3]**：输出比较 2 模式——位 3 (Output Compare 2 mode - bit 3)

位 23:17 保留，始终读为 0。

位 16 **OC1M[3]**：输出比较 1 模式——位 3 (Output Compare 1 mode - bit 3)

位 15 **OC2CE**：输出比较 2 清零使能 (Output Compare 2 clear enable)

位 14:12 **OC2M[2:0]**：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)
此位字段定义通道方向（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，IC2 映射到 TI2 上
10：CC2 通道配置为输入，IC2 映射到 TI1 上
11：CC2 通道配置为输入，IC2 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注：　仅当通道关闭时（TIMx_CCER 中的 CC2E = 0），才可向 CC2S 位写入数据。

位 7 **OC1CE**：输出比较 1 清零使能 (Output Compare 1 clear enable)
OC1CE：输出比较 1 清零使能 (Output Compare 1 Clear Enable)
0：OC1Ref 不受 ETRF 输入影响
1：ETRF 输入上检测到高电平时，OC1Ref 立即清零

位 6:4 **OC1M**：输出比较 1 模式 (Output Compare 1 mode)

这些位定义提供 OC1 和 OC1N 的输出参考信号 OC1REF 的行为。OC1REF 为高电平有效，而 OC1 和 OC1N 的有效电平则取决于 CC1P 位和 CC1NP 位。

0000：冻结——输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。

0001：将通道 1 设置为匹配时输出有效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1 (TIMx_CCR1) 匹配时，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，OC1REF 信号强制变为低电平。

0011：翻转——TIMx_CNT=TIMx_CCR1 时，OC1REF 发生翻转。

0100：强制变为无效电平——OC1REF 强制变为低电平。

0101：强制变为有效电平——OC1REF 强制变为高电平。

0110：PWM 模式 1——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为无效状态 (OC1REF=0)，否则为有效状态 (OC1REF=1)。

0111：PWM 模式 2——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为无效状态，否则为有效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。

1000：可再触发 OPM 模式 1——在递增计数模式下，通道为有效状态，直至（在 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 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 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 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——OC1REF 与在 PWM 模式 1 下的行为相同。OC1REFC 是 OC1REF 和 OC2REF 的逻辑或运算结果。

1101：组合 PWM 模式 2——OC1REF 与在 PWM 模式 2 下的行为相同。OC1REFC 是 OC1REF 和 OC2REF 的逻辑与运算结果。

1110：不对称 PWM 模式 1——OC1REF 与在 PWM 模式 1 下的行为相同。计数器递增计数时，OC1REFC 输出 OC1REF；计数器递减计数时，OC1REFC 输出 OC2REF。

1111：不对称 PWM 模式 2——OC1REF 与在 PWM 模式 2 下的行为相同。计数器递增计数时，OC1REFC 输出 OC1REF；计数器递减计数时，OC1REFC 输出 OC2REF。

注： ***1****：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。*

2*：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。*

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活动寄存器中。

注： ***1****：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。*

2*：只有单脉冲模式下才可在未验证预装载寄存器的情况下使用 PWM 模式（TIMx_CR1 寄存器中的 OPM 位置 1）。其它情况下则无法保证该行为。*

位 2 **OC1FE**：输出比较 1 快速使能 (Output Compare 1 fast enable)

此位用于加快触发输入事件对 CC 输出的影响。

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**：捕获/比较 1 选择 (Capture/Compare 1 Selection)

此位字段定义通道方向（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，IC1 映射到 TI1 上。
10：CC1 通道配置为输入，IC1 映射到 TI2 上。
11：CC1 通道配置为输入，IC1 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注：　仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

输入捕获模式

位 31:16 保留，始终读为 0。

位 15:12 **IC2F**：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**：捕获/比较 2 选择 (Capture/Compare 2 selection)

此位字段定义通道方向（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，IC2 映射到 TI2 上。
10：CC2 通道配置为输入，IC2 映射到 TI1 上。
11：CC2 通道配置为输入，IC2 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效。

注：　仅当通道关闭时（TIMx_CCER 中的 CC2E = 0），才可向 CC2S 位写入数据。

位 7:4 **IC1F**：输入捕获 1 滤波器 (Input capture 1 filter)

此位字段可定义 TI1 输入的采样频率和适用于 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：$f_{SAMPLING}=f_{CK_INT}$，N=2
0010：$f_{SAMPLING}=f_{CK_INT}$，N=4
0011：$f_{SAMPLING}=f_{CK_INT}$，N=8
0100：$f_{SAMPLING}=f_{DTS}/2$，N=6
0101：$f_{SAMPLING}=f_{DTS}/2$，N=8
0110：$f_{SAMPLING}=f_{DTS}/4$，N=6
0111：$f_{SAMPLING}=f_{DTS}/4$，N=8
1000：$f_{SAMPLING}=f_{DTS}/8$，N=6
1001：$f_{SAMPLING}=f_{DTS}/8$，N=8
1010：$f_{SAMPLING}=f_{DTS}/16$，N=5
1011：$f_{SAMPLING}=f_{DTS}/16$，N=6
1100：$f_{SAMPLING}=f_{DTS}/16$，N=8
1101：$f_{SAMPLING}=f_{DTS}/32$，N=5
1110：$f_{SAMPLING}=f_{DTS}/32$，N=6
1111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 3:2 **IC1PSC**：输入捕获 1 预分频器 (Input capture 1 prescaler)

此位字段定义 CC1 输入 (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**：捕获/比较 1 选择 (Capture/Compare 1 Selection)

此位字段定义通道方向（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，IC1 映射到 TI1 上
10：CC1 通道配置为输入，IC1 映射到 TI2 上
11：CC1 通道配置为输入，IC1 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注： 仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

23.4.8 TIMx 捕获/比较模式寄存器 2 (TIMx_CCMR2)

TIMx capture/compare mode register 2

偏移地址：0x1C

复位值：0x0000

请参见上述 CCMR1 寄存器说明。

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]
							Res.								Res.
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

位 31:25 保留，始终读为 0。

位 24 **OC4M[3]**：输出比较 2 模式——位 3 (Output Compare 4 mode - bit 3)

位 23:17 保留，始终读为 0。

位 16 **OC3M[3]**：输出比较 1 模式——位 3 (Output Compare 3 mode - bit 3)

位 15 OC4CE：输出比较 4 清零使能 (Output compare 4 clear enable)

位 14:12 **OC4M**：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**：捕获/比较 4 选择 (Capture/Compare 4 selection)
此位字段定义通道方向（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，IC4 映射到 TI4 上
10：CC4 通道配置为输入，IC4 映射到 TI3 上
11：CC4 通道配置为输入，IC4 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注： 仅当通道关闭时（TIMx_CCER 中的 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 6:4 **OC3M**：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**：捕获/比较 3 选择 (Capture/compare 3 selection)
此位字段定义通道方向（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，IC3 映射到 TI3 上
10：CC3 通道配置为输入，IC3 映射到 TI4 上
11：CC3 通道配置为输入，IC3 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注： 仅当通道关闭时（TIMx_CCER 中的 CC3E = 0），才可向 CC3S 位写入数据。

输入捕获模式

位 31:16 保留，始终读为 0。

位 15:12 **IC4F**：输入捕获 4 滤波器 (Input capture 4 filter)

位 11:10 **IC4PSC**：输入捕获 4 预分频器 (Input capture 4 prescaler)

位 9:8 **CC4S**：捕获/比较 4 选择 (Capture/Compare 4 selection)
此位字段定义通道方向（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，IC4 映射到 TI4 上
10：CC4 通道配置为输入，IC4 映射到 TI3 上
11：CC4 通道配置为输入，IC4 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注： 仅当通道关闭时（TIMx_CCER 中的 CC4E = 0），才可向 CC4S 位写入数据。

位 7:4 **IC3F**：输入捕获 3 滤波器 (Input capture 3 filter)

位 3:2 **IC3PSC**：输入捕获 3 预分频器 (Input capture 3 prescaler)

位 1:0 **CC3S**：捕获/比较 3 选择 (Capture/compare 3 selection)
此位字段定义通道方向（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，IC3 映射到 TI3 上
10：CC3 通道配置为输入，IC3 映射到 TI4 上
11：CC3 通道配置为输入，IC3 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效
注： 仅当通道关闭时（TIMx_CCER 中的 CC3E = 0），才可向 CC3S 位写入数据。

23.4.9 TIMx 捕获/比较使能寄存器 (TIMx_CCER)

TIMx capture/compare enable register

偏移地址：0x20

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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 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 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 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 配合使用可定义 TI1FP1/TI2FP1 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)*
CC1 通道配置为输出：
0：OC1 高电平有效
1：OC1 低电平有效
CC1 通道配置为输入：CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的极性。
00：未反相/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）。
01：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）。
10：保留，不使用此配置
11：未反相/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。

位 0 **CC1E**：*捕获/比较 1 输出使能 (Capture/Compare 1 output enable)*。
CC1 通道配置为输出：
0：关闭——OC1 未激活
1：开启——在相应输出引脚上输出 OC1 信号
CC1 通道配置为输入：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 (TIMx_CCR1) 中。
0：禁止捕获
1：使能捕获

表 131. 标准 OCx 通道的输出控制位

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

注： *与标准 OCx 通道相连的外部 IO 引脚的状态取决于 OCx 通道的状态以及 GPIO 和 AFIO 寄存器。*

23.4.10 TIMx 计数器 (TIMx_CNT)

TIMx counter

偏移地址：0x24

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT[31]或UIFCPY	CNT[30:16]（取决于定时器）														
rw or r	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 值取决于 TIMx_CR1 中的 IUFREMAP。

如果 UIFREMAP = 0

CNT[31]：计数器值的最高有效位（对于 TIM2 和 TIM5 ）

其它定时器上保留

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 **CNT[30:16]**：计数器值的最高有效部分（对于 TIM2 和 TIM5 ）

位 15:0 **CNT[15:0]**：计数器值的最低有效部分

23.4.11 TIMx 预分频器 (TIMx_PSC)

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 CK_CNT 等于 f_{CK_PSC} / (PSC[15:0] + 1)。

PSC 包含在每次发生更新事件时要装载到实际预分频器寄存器的值。

23.4.12 TIMx 自动重载寄存器 (TIMx_ARR)

TIMx auto-reload register

偏移地址：0x2C

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR[31:16]（取决于定时器）															
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 31:16 **ARR[31:16]**：自动重载值的高 16 位（对于 TIM2 和 TIM5 ）

位 15:0 **ARR[15:0]**：自动重载值的低 16 位
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 652 页上的第 23.3.1 节：时基单元*。
当自动重载值为空时，计数器不工作。

23.4.13 TIMx 捕获/比较寄存器 1 (TIMx_CCR1)

TIMx capture/compare register 1

偏移地址：0x34

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1[31:16]（取决于定时器）															
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 31:16 **CCR1[31:16]**：捕获/比较 1 值的高 16 位（对于 TIM2 和 TIM5）

位 15:0 **CCR1[15:0]**：捕获/比较 1 值的低 16 位 (Low 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（拷贝到实际起作用的捕获/比较寄存器 1）。
实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC1 输出上发出信号的值。

如果通道 CC1 配置为输入：
CCR1 为上一个输入捕获 1 事件 (IC1) 发生时的计数器值。

23.4.14 TIMx 捕获/比较寄存器 2 (TIMx_CCR2)

TIMx capture/compare register 2

偏移地址：0x38

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2[31:16]（取决于定时器）															
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 31:16 **CCR2[31:16]**：捕获/比较 2 值的高 16 位（对于 TIM2 和 TIM5）。

位 15:0 **CCR2[15:0]**：捕获/比较 2 值的低 16 位 (Low 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（拷贝到实际起作用的捕获/比较寄存器 2）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC2 输出上发出信号的值。

如果通道 CC2 配置为输入：

CCR2 为上一个输入捕获 2 事件 (IC2) 发生时的计数器值。

23.4.15 TIMx 捕获/比较寄存器 3 (TIMx_CCR3)

TIMx capture/compare register 3

偏移地址：0x3C

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3[31:16]（取决于定时器）															
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 31:16 **CCR3[31:16]**：捕获/比较 3 值的高 16 位（对于 TIM2 和 TIM5）

位 15:0 **CCR3[15:0]**：捕获/比较 3 值的低 16 位 (Low 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（拷贝到实际起作用的捕获/比较寄存器 3）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC3 输出上发出信号的值。

如果通道 CC3 配置为输入：

CCR3 为上一个输入捕获 3 事件 (IC3) 发生时的计数器值。

23.4.16 TIMx 捕获/比较寄存器 4 (TIMx_CCR4)

TIMx capture/compare register 4

偏移地址：0x40

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4[31:16]（取决于定时器）															
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 31:16 **CCR4[31:16]**：捕获/比较 4 值的高 16 位（对于 TIM2 和 TIM5）

位 15:0 **CCR4[15:0]**：捕获/比较 4 值的低 16 位 (Low Capture/Compare value)

1. 如果 CC4 通道配置为输出（CC4S 位）：
   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（拷贝到实际起作用的捕获/比较寄存器 4）。
   实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC4 输出上发出信号的值。
2. 如果 CC4 通道配置为输入（TIMx_CCMR4 寄存器中的 CC4S 位）：
   CCR4 为上一个输入捕获 4 事件 (IC4) 发生时的计数器值。

23.4.17 TIMx DMA 控制寄存器 (TIMx_DCR)

TIMx DMA control register

偏移地址：0x48

复位值：0x0000

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 15:13 保留，必须保持复位值。

位 12:8 **DBL[4:0]**：DMA 连续传送长度 (DMA burst length)

该 5 位向量定义了 DMA 的传送次数（当对 TIMx_DMAR 寄存器进行读或写时，定时器进行一次连续传送）。

00000：1 次传送，
00001：2 次传送，
00010：3 次传送，
...
10001：18 次传送。

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
...

示例：以下面的传送为例：DBL = 7 次传送且 DBA = TIMx_CR1。这种情况下将向/从自 TIMx_CR1 地址开始的 7 个寄存器传输数据。

23.4.18 TIMx 全传输 DMA 地址 (TIMx_DMAR)

TIMx DMA address for full transfer

偏移地址：0x4C

复位值：0x0000

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 15:0 **DMAB[15:0]**：DMA 连续传送寄存器 (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。

23.4.19 TIM2 选项寄存器 1 (TIM2_OR)

TIM2 option register 1

偏移地址：0x50

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	ITR1_RMP[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
				rw	rw										

位 15:12 保留，必须保持复位值。

位 11:10 **ITR1_RMP[1:0]**：内部触发 1 重映射 (Internal trigger 1 remap)

由软件置 1 和清零。
00：TIM8_TRGOUT
01：PTP 触发输出连接到 TIM2_ITR1
10：OTG FS SOF 连接到 TIM2_ITR1 输入
11：OTG HS SOF 连接到 TIM2_ITR1 输入

位 9:0 保留，必须保持复位值。

23.4.20 TIM2 选项寄存器 1 (TIM5_OR)

TIM2 option register 1

偏移地址：0x50

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.	TI4_RMP[1:0]		Res.	Res.	Res.	Res.	Res.	Res.
								rw	rw						

位 15:8 保留，必须保持复位值。

位 7:6 **TI4_RMP[1:0]**：定时器输入 4 重映射 (Timer Input 4 remap)
由软件置 1 和清零。
00：TIM5 通道 4 连接到 GPIO：请参见数据手册中的复用功能映射表。
01：为进行校准，LSI 内部时钟连接到 TIM5_CH4 输入。
10：为进行校准，LSE 内部时钟连接到 TIM5_CH4 输入。
11：为进行校准，RTC 唤醒中断连接到 TIM5_CH4 输入。
应使能唤醒中断。

位 5:0 保留，必须保持复位值。

23.4.21 TIM3 选项寄存器 1 (TIM3_OR1)

TIM3 option register 1

偏移地址：0x50

复位值：0x0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1_RMP[1:0]	
														rw	rw

位 31:2 保留，必须保持复位值。

位 1:0 **TI1_RMP[1:0]**：输入捕获 1 重映射 (Input Capture 1 remap)
00：TIM3 输入捕获 1 连接到 I/O
01：TIM3 输入捕获 1 连接到 COMP1_OUT
10：TIM3 输入捕获 1 连接到 COMP2_OUT
11：TIM3 输入捕获 1 连接到 COMP1_OUT 和 COMP2_OUT 的逻辑或运算结果

23.4.22 TIMx 寄存器映射

TIMx 寄存器按照下表所述方式映射：

表 132. TIM2/TIM3/TIM4/TIM5 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	**TIMx_CR1**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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
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	TI1S	MMS[2:0]			CCDS	Res	Res	Res
	Reset value																									0	0	0	0	0			
0x08	**TIMx_SMCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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
0x0C	**TIMx_DIER**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	TDE	COMDE	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
0x10	**TIMx_SR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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
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	TG	Res	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																										0		0	0	0	0	0
0x18	**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	0
	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	0
0x1C	**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	0
	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	0
0x20	**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 132. TIM2/TIM3/TIM4/TIM5 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
0x24	**TIMx_CNT**	CNT[31] or UIFCPY	CNT[30:16] (TIM2 and TIM5 only, reserved on the other timers)															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
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**	ARR[31:16] (TIM2 and TIM5 only, reserved on the other timers)																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	0	0
0x30	Reserved																																
0x34	**TIMx_CCR1**	CCR1[31:16] (TIM2 and TIM5 only, reserved on the other timers)																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
0x38	**TIMx_CCR2**	CCR2[31:16] (TIM2 and TIM5 only, reserved on the other timers)																CCR2[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
0x3C	**TIMx_CCR3**	CCR3[31:16] (TIM2 and TIM5 only, reserved on the other timers)																CCR3[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
0x40	**TIMx_CCR4**	CCR4[31:16] (TIM2 and TIM5 only, reserved on the other timers)																CCR4[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
0x44	Reserved																																
0x48	**TIMx_DCR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DBL[4:0]			Res	Res	Res	DBA[4:0]						
	Reset value																				0	0	0	0	0				0	0	0	0	0
0x4C	**TIMx_DMAR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DMAB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	**TIM2_OR**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ITR1_RMP[1:0]		Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
	Reset value																					0	0										

表 132. TIM2/TIM3/TIM4/TIM5 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
0x50	**TIM5_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	TI4_RMP[1:0]		Res	Res	Res	Res	Res	Res
	Reset value																									0	0						
0x50	**TIM3_OR1**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	TI1_RMP[1:0]
	Reset value																																0

有关寄存器边界地址的信息，请参见*第 2.2.2 节：存储器映射和寄存器边界地址*。

24 通用定时器（TIM9 到 TIM14）

24.1 TIM9 到 TIM14 简介

TIM9 到 TIM14 通用定时器包含一个 16 位自动重载计数器，该计数器由可编程预分频器驱动。

它们可用于多种用途，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）。

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。

TIM9 到 TIM14 定时器彼此完全独立，不共享任何资源。如*第 24.3.12 节*中所述，它们可以同步操作。

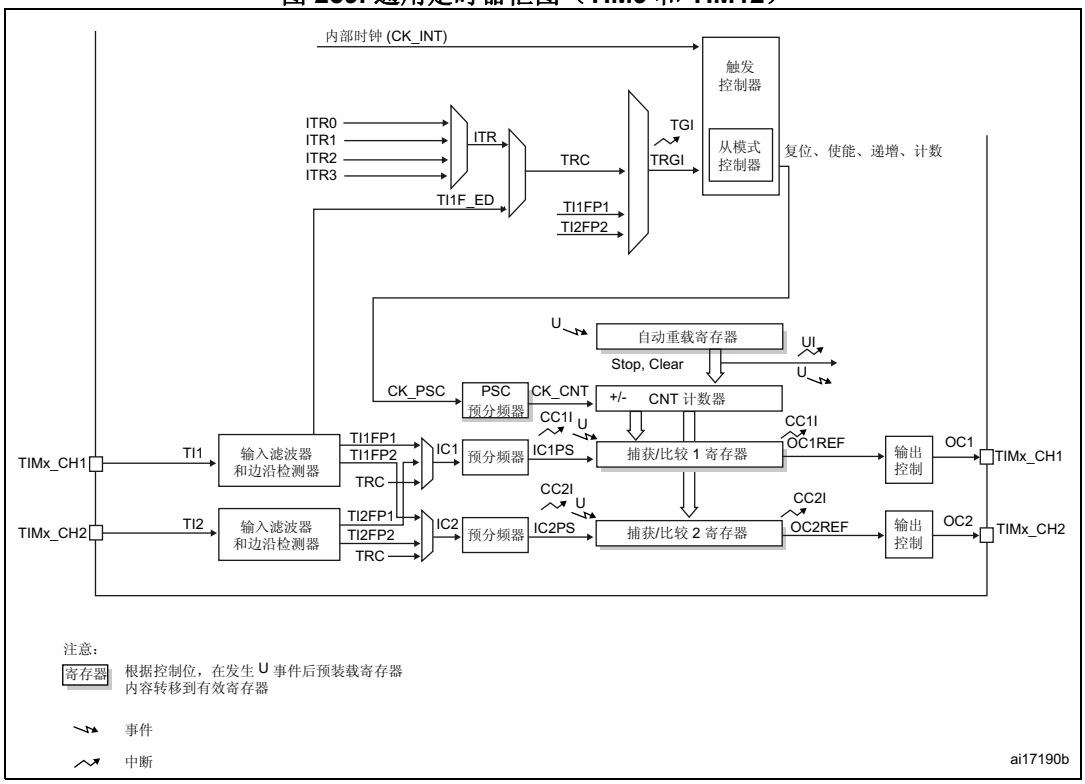
24.2 TIM9 到 TIM14 主要特性

24.2.1 TIM9/TIM12 主要特性

TIM9 到 TIM14 通用定时器具有以下特性：

- 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 65536 之间
- 多达 2 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）
  - 单脉冲模式输出
- 使用外部信号控制定时器且可实现多个定时器互连的同步电路
- 发生如下事件时生成中断：
  - 更新：计数器上溢、计数器初始化（通过软件或内部触发）
  - 触发事件（计数器启动、停止、初始化或者由内部触发计数）
  - 输入捕获
  - 输出比较

图 239. 通用定时器框图（TIM9 和 TIM12）

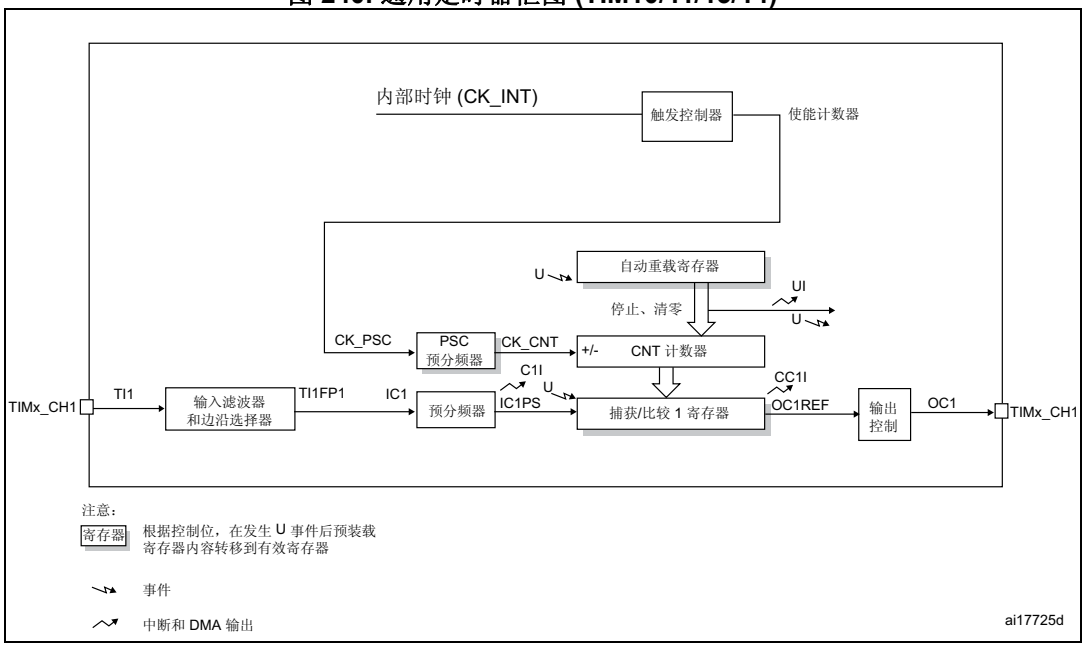


24.2.2 TIM10/TIM11 和 TIM13/TIM14 主要特性

通用定时器 TIM10/TIM11 和 TIM13/TIM14 具有以下特性：

- 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 65536 之间
- 独立通道，可用于：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿对齐模式）
  - 单脉冲模式输出
- 发生如下事件时生成中断：
  - 更新：计数器上溢、计数器初始化（通过软件）
  - 输入捕获
  - 输出比较

图 240. 通用定时器框图 (TIM10/11/13/14)



24.3 TIM9 到 TIM14 功能说明

24.3.1 时基单元

定时器的主要模块由一个 16 位计数器及其相关的自动重载寄存器组成。计数器采用递增方式计数。

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直接传送到影子寄存器，也可以在每次发生更新事件 (UEV) 时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位 (ARPE)。当计数器达到上溢值并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。下文将针对各配置的更新事件的产生进行详细介绍。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位 (CEN) 置 1 时，才会启动计数器（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频率进行分频，分频系数介于 1 和 65536 之间。该预分频器基于 TIMx_PSC 寄存器中的 16 位寄存器所控制的 16 位计数器。由于该控制寄存器具有缓冲功能，因此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 241 和*图 242* 以一些示例说明在预分频比实时变化时计数器的行为。

图 241. 预分频器分频由 1 变为 2 时的计数器时序图

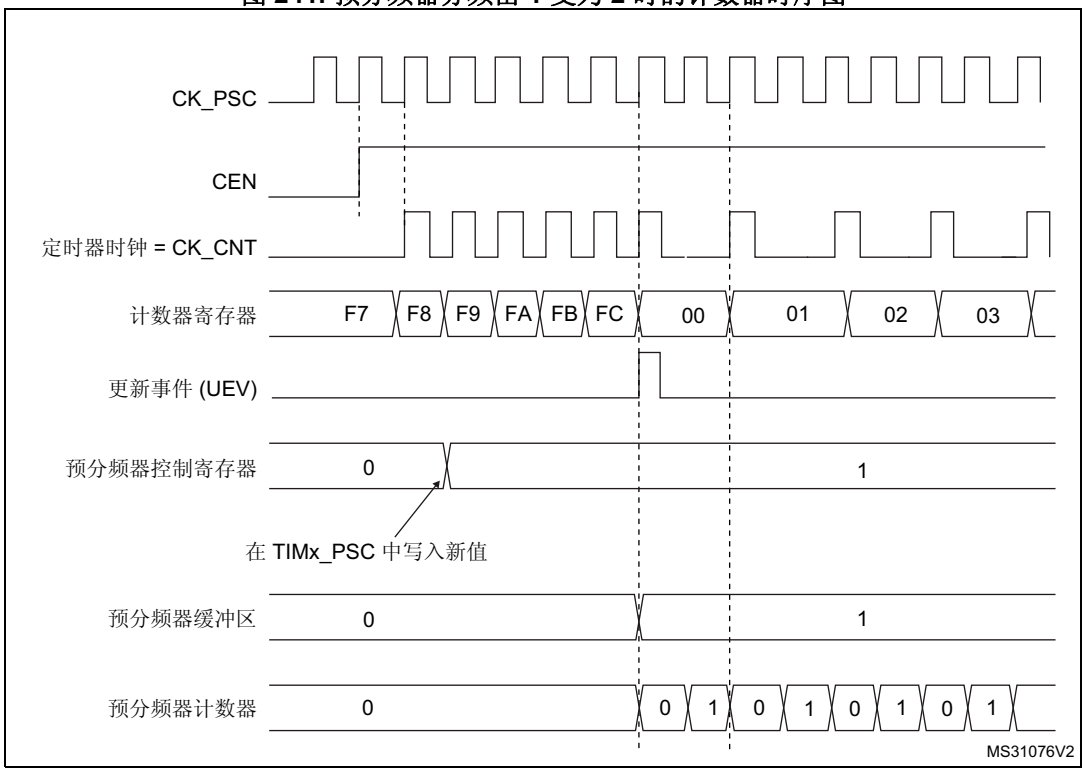
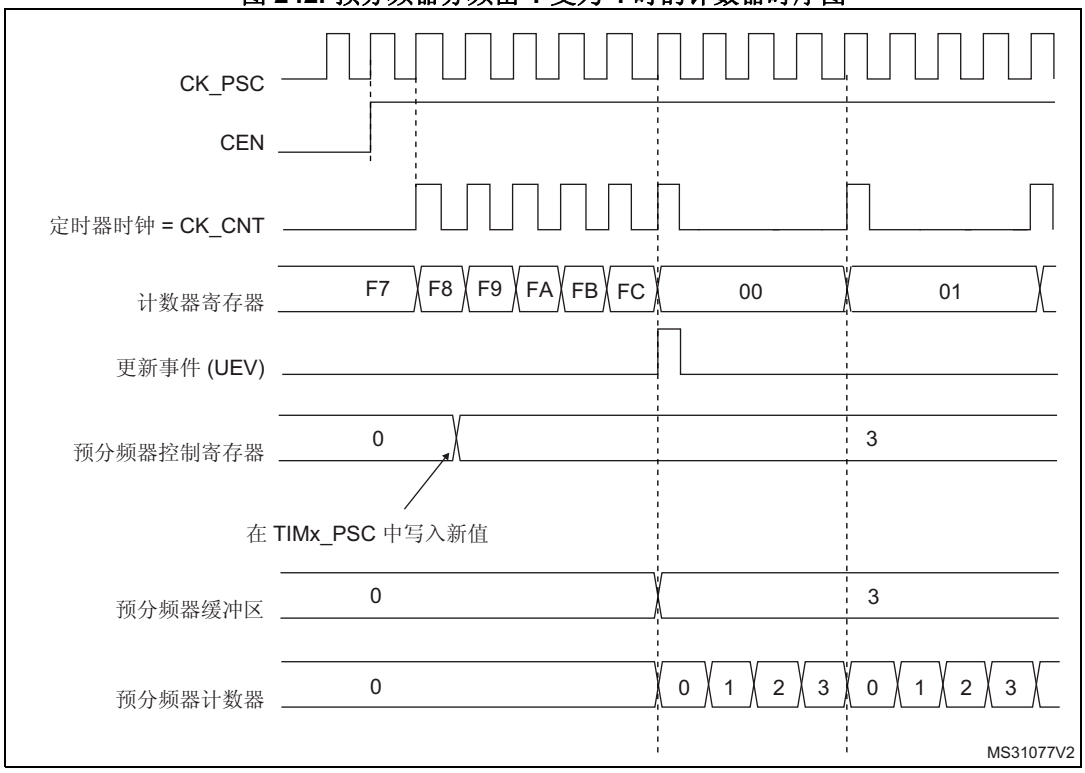


图 242. 预分频器分频由 1 变为 4 时的计数器时序图



24.3.2 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。

将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用 TIM9 和 TIM12 上的从模式控制器）也会生成更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变）。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 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 243. 计数器时序图，1 分频内部时钟

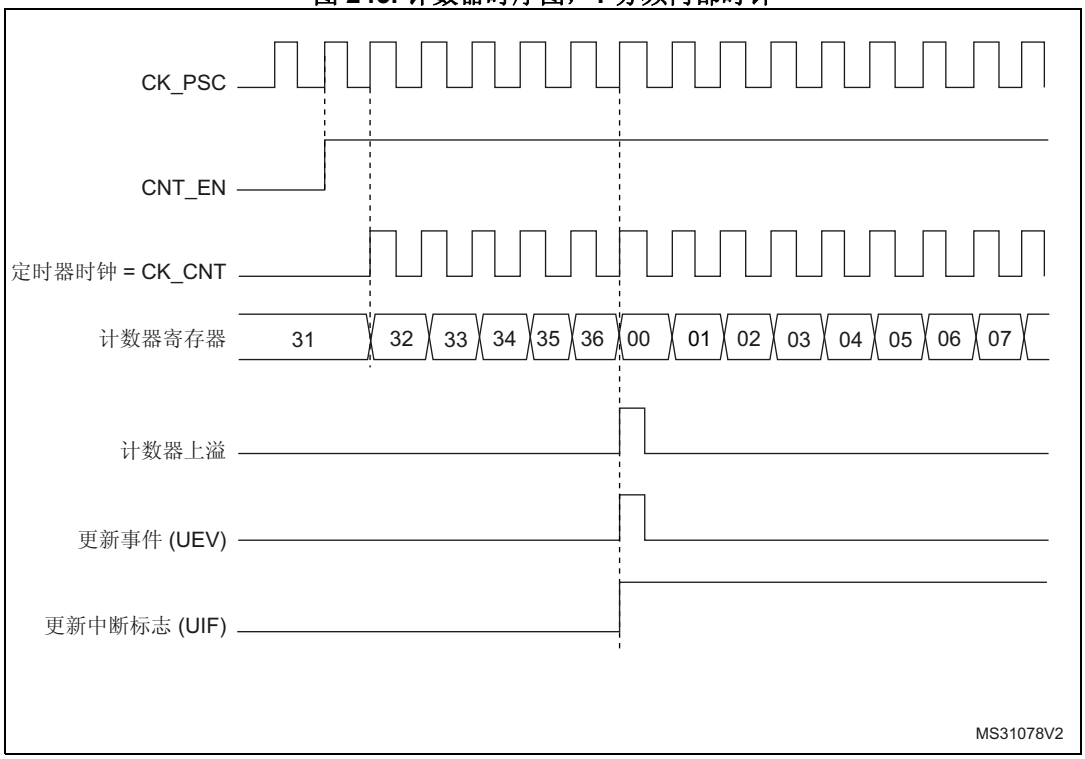


图 244. 计数器时序图，2 分频内部时钟

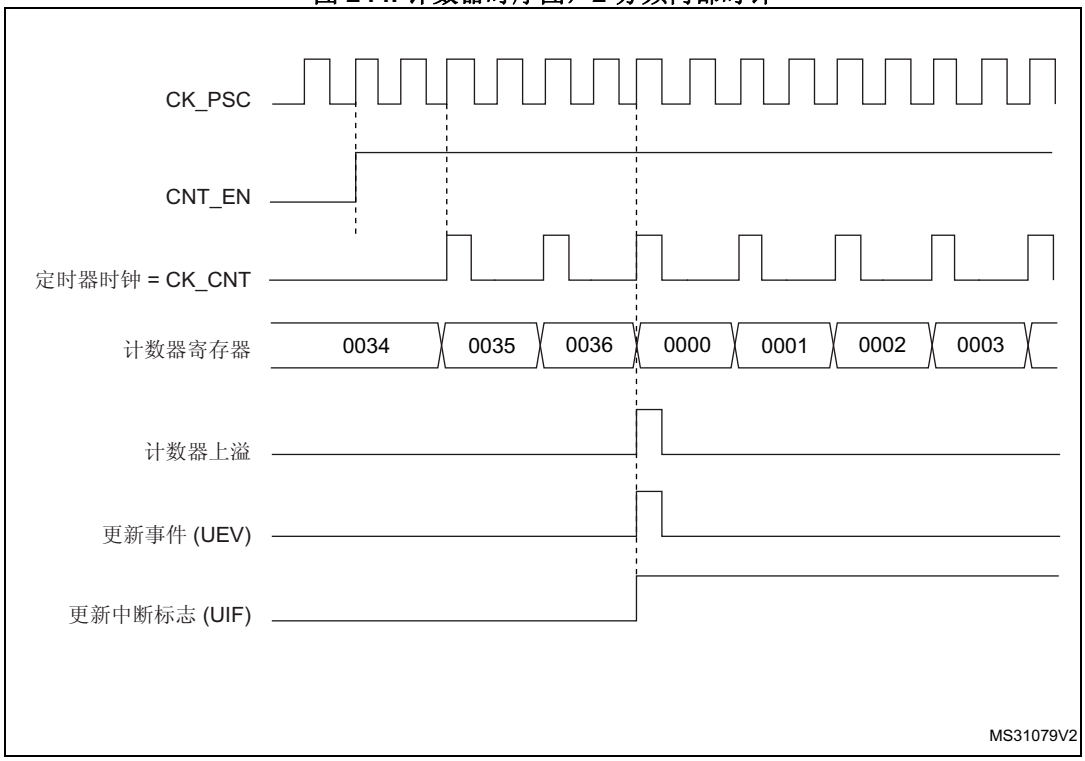


图 245. 计数器时序图，4 分频内部时钟

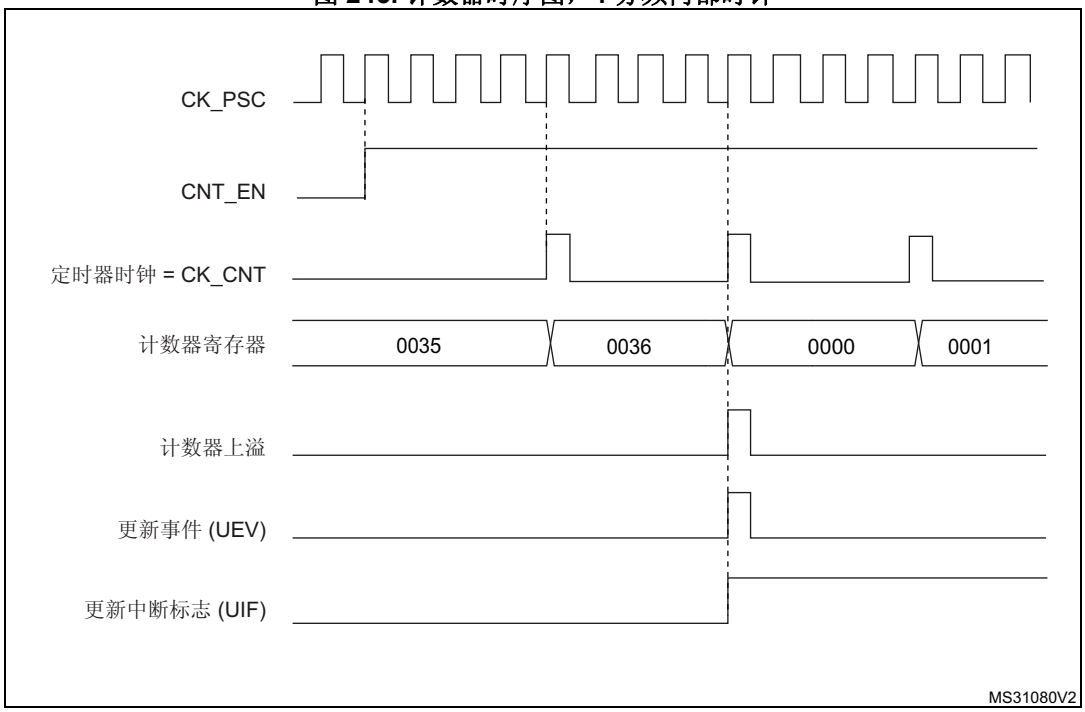


图 246. 计数器时序图，N 分频内部时钟

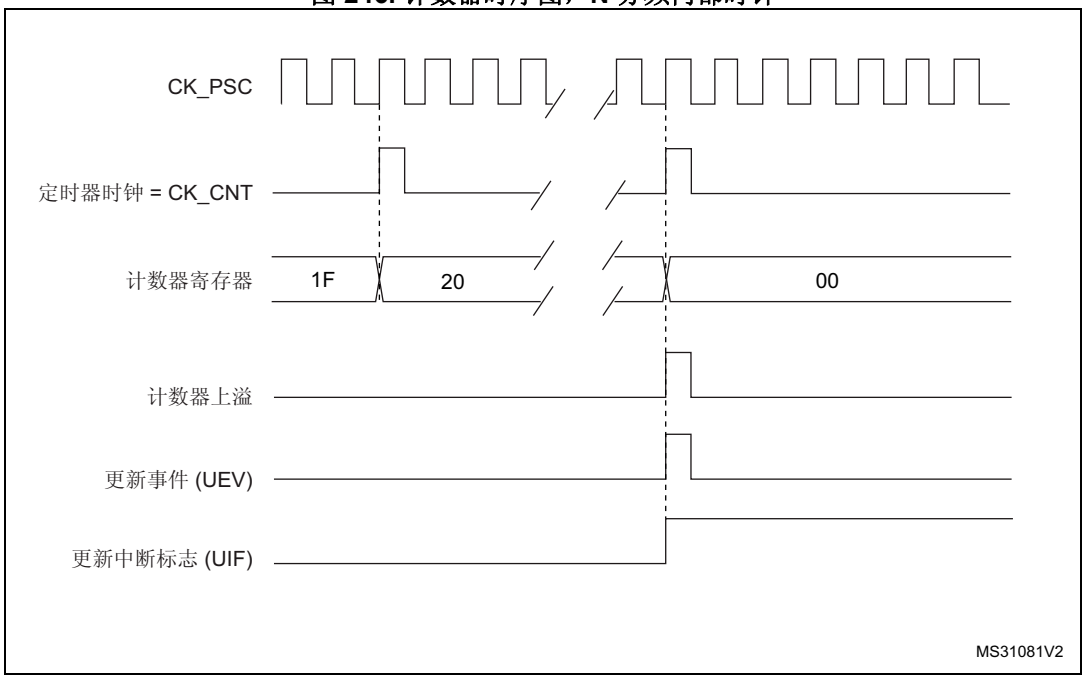


图 247. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载）

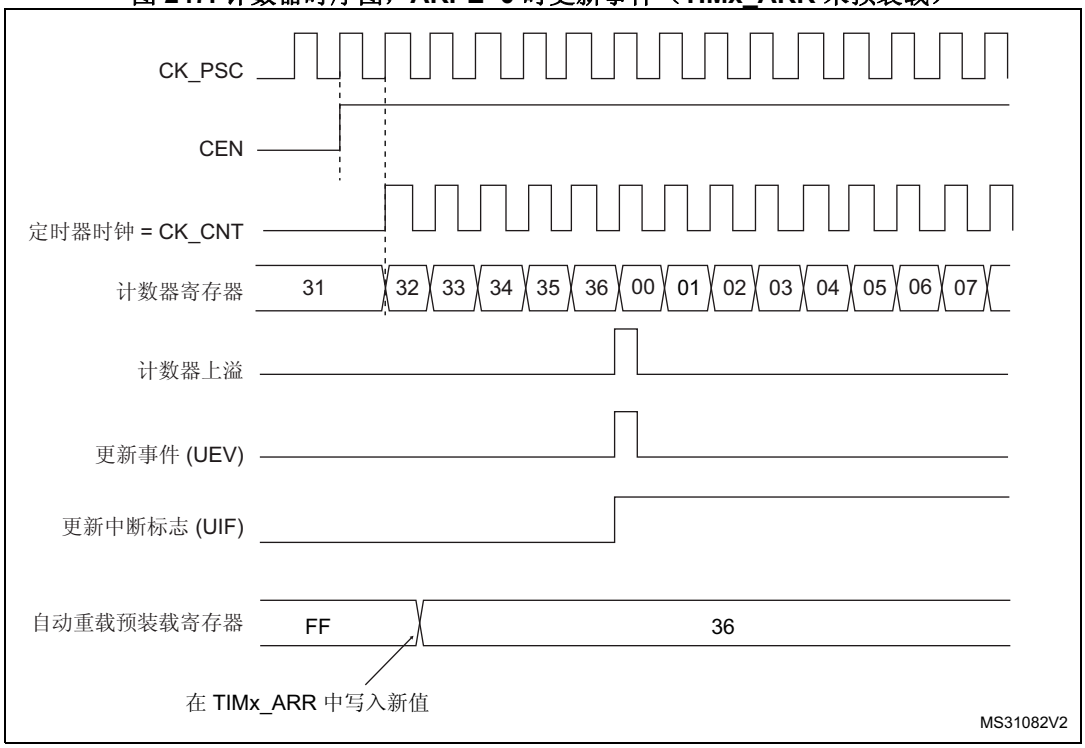
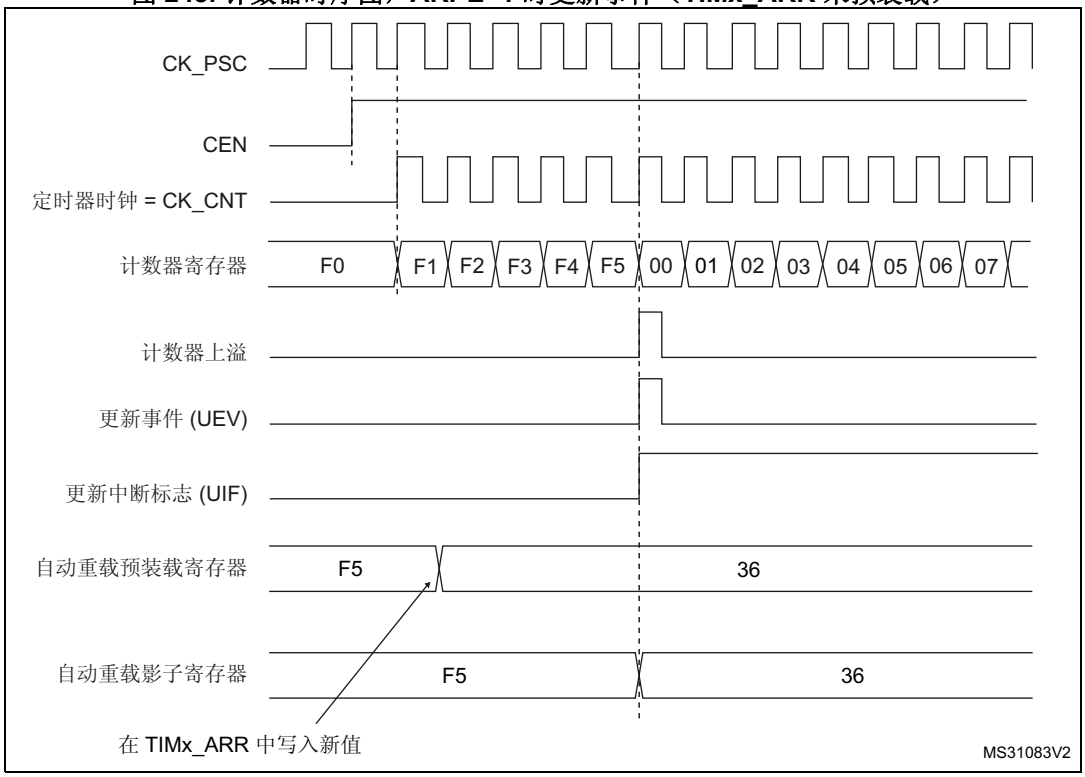


图 248. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 未预装载）



24.3.3 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟 (CK_INT)
- 外部时钟模式 1（针对 **TIM9 和 TIM12**）：外部输入引脚 (TIx)
- 内部触发输入 (ITRx)（针对 **TIM9 和 TIM12**）：连接来自其它计数器的触发输出。更多详细信息，请参见*将一个定时器用作另一个定时器的预分频器*一节。

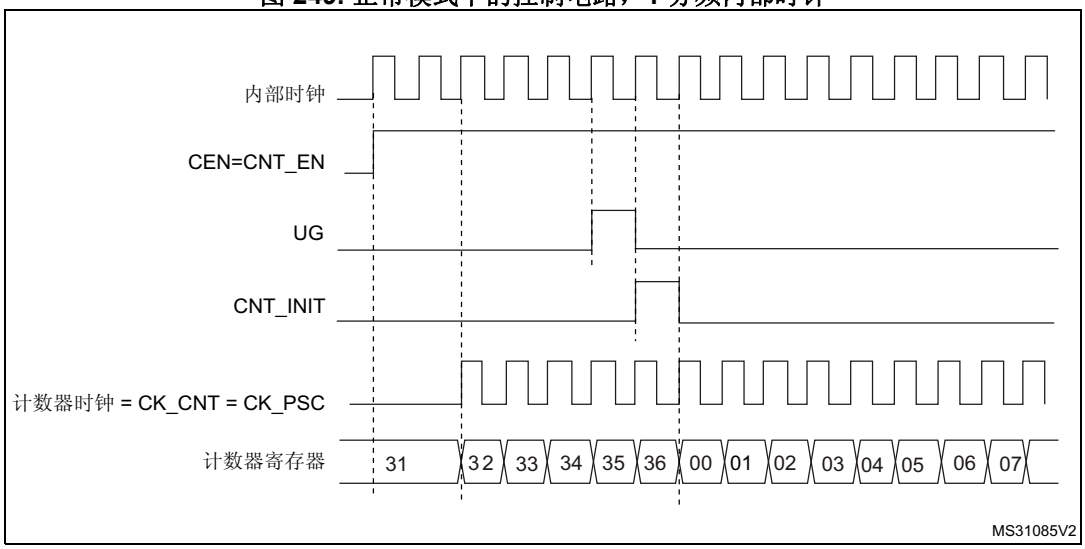
内部时钟源 (CK_INT)

内部时钟源是 TIM10/TIM11 和 TIM13/TIM14 的默认时钟源。

对于 TIM9 和 TIM12，在禁止从模式控制器后选择内部时钟源（SMS=“000”）。然后，将 TIMx_CR1 寄存器中的 CEN 位和 TIMx_EGR 寄存器中的 UG 位用作控制位，并且只能通过软件对其进行更改（保持清零状态的 UG 除外）。当对 CEN 位写入 1 时，预分频器的时钟就由内部时钟 CK_INT 提供。

图 249 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）。

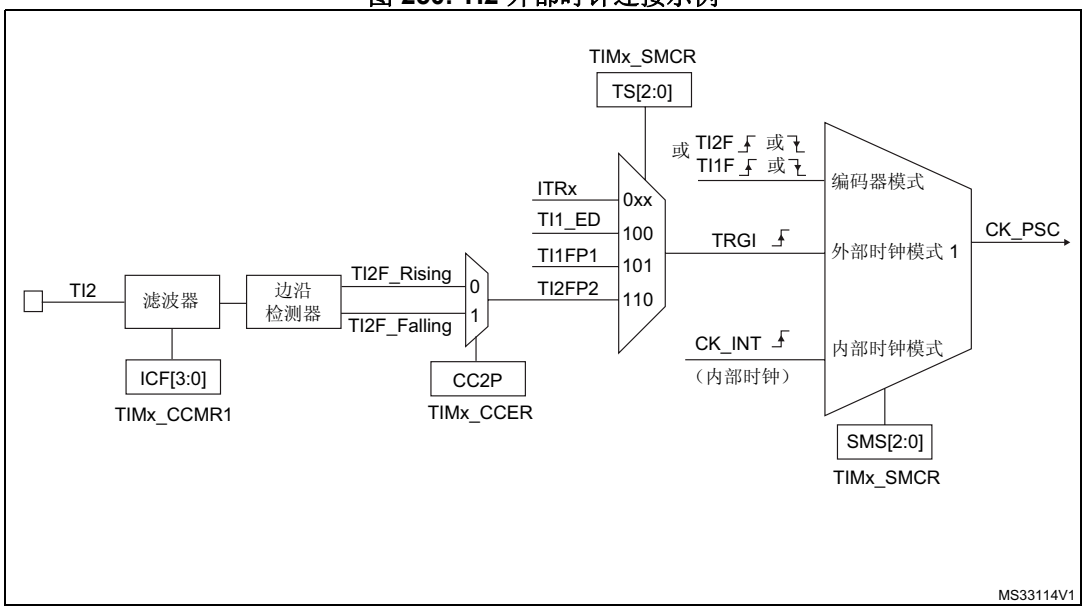
图 249. 正常模式下的控制电路，1 分频内部时钟



外部时钟源模式 1（TIM9 和 TIM12）

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 250. TI2 外部时钟连接示例



例如，要使递增计数器在 TI2 输入出现上升沿时计数，请执行以下步骤：

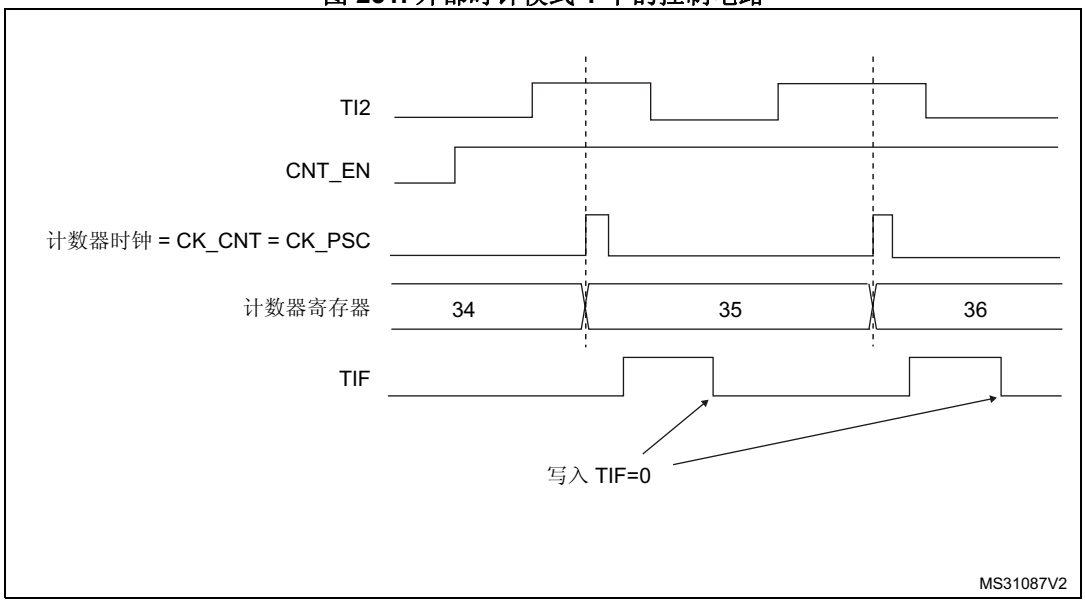
1. 通过在 TIMx_CCMR1 寄存器中写入 CC2S = “01”来配置通道 2，使其能够检测 TI2 输入的上升沿。
2. 通过在 TIMx_CCMR1 寄存器中写入 IC2F[3:0] 位来配置输入滤波带宽（如果不需要任何滤波器，请保持 IC2F=“0000”）。
3. 通过在 TIMx_CCER 寄存器中写入 CC2P=“0”和 CC2NP=“0”来选择上升沿极性。
4. 通过在 TIMx_SMCR 寄存器中写入 SMS=“111”，使定时器在外部时钟模式 1 下工作。
5. 通过在 TIMx_SMCR 寄存器中写入 TS=“110”来选择 TI2 作为触发输入源。
6. 通过在 TIMx_CR1 寄存器中写入 CEN=“1”来使能计数器。

注： *由于捕获预分频器不用于触发操作，因此无需对其进行配置。*

当 TI2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

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

图 251. 外部时钟模式 1 下的控制电路



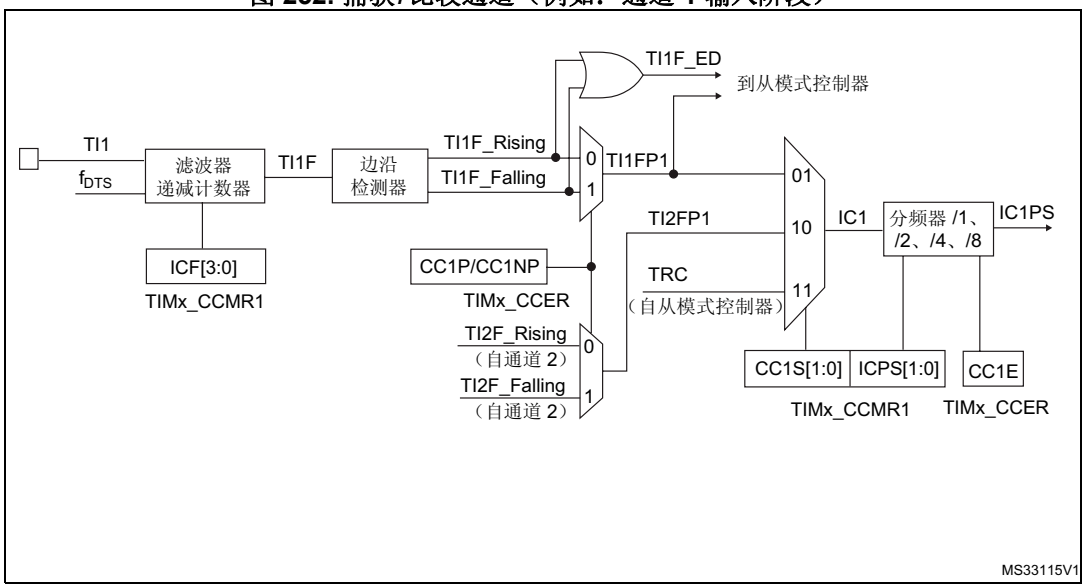
24.3.4 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入阶段（数字滤波、多路复用和预分频器）和一个输出阶段（比较器和输出控制）构建而成。

图 252 到*图 254* 概括介绍了一个捕获/比较通道。

输入阶段对相应的 TIx 输入进行采样，生成一个滤波后的信号 TIxF。然后，带有极性选择功能的边沿检测器生成一个信号 (TIxFPx)，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 252. 捕获/比较通道（例如：通道 1 输入阶段）



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 253. 捕获/比较通道 1 主电路

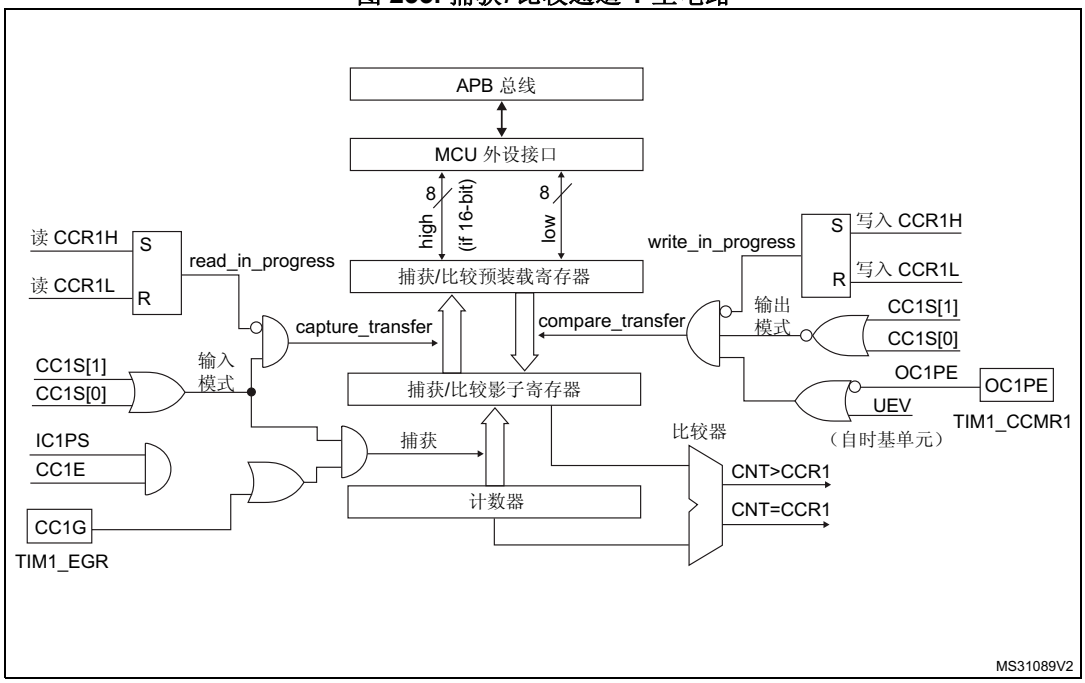
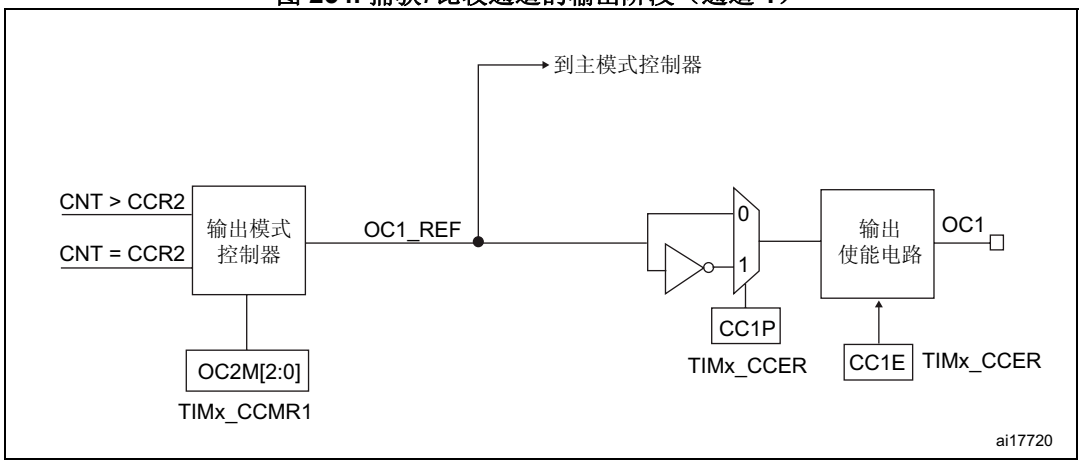


图 254. 捕获/比较通道的输出阶段（通道 1）



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。始终可通过读写操作访问预装载寄存器。

在捕获模式下，捕获实际发生在影子寄存器中，然后将影子寄存器的内容复制到预装载寄存器中。

在比较模式下，预装载寄存器的内容将复制到影子寄存器中，然后将影子寄存器的内容与计数器进行比较。

24.3.5 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，则会将重复捕获标志 CCxOF（TIMx_SR 寄存器）置 1。可通过软件将 CCxIF 清零，方法是：向 CCxIF 写入“0”，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 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. 选择有效输入：TIMx_CCR1 必须连接到 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。
2. 根据连接到定时器的信号，对所需的输入滤波带宽进行编程（如果输入为 TIx 输入之一，则对 TIMx_CCMRx 寄存器中的 ICxF 位进行编程）。假设信号边沿变化时，输入信号最多在 5 个内部时钟周期内发生抖动。因此，我们必须将滤波带宽设置为大于 5 个内部时钟周期。在检测到 8 个具有新电平的连续采样（以 f_{DTS} 频率采样）后，可以确认 TI1 上的跳变沿。然后向 TIMx_CCMR1 寄存器中的 IC1F 位写入“0011”。
3. 通过在 TIMx_CCER 寄存器中将 CC1P 位和 CC1NP 位编程为“00”，选择 TI1 上的有效转换边沿（本例中为上升沿）。
4. 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”）。
5. 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。
6. 如果需要，可通过将 TIMx_DIER 寄存器中的 CC1IE 位置 1 来使能相关中断请求。

发生输入捕获时：

- 发生有效跳变沿时，TIMx_CCR1 寄存器会获取计数器的值。
- 将 CC1IF 标志置 1（中断标志）。如果至少发生了两次连续捕获，但 CC1IF 标志未被清零，这样 CC1OF 捕获溢出标志会被置 1。
- 根据 CC1IE 位生成中断。

要处理重复捕获，建议在读出捕获溢出标志之前读取数据。这样可避免丢失在读取捕获溢出标志之后与读取数据之前可能出现的重复捕获信息。

注： *通过软件将 TIMx_EGR 寄存器中的相应 CCxG 位置 1 可生成 IC 中断请求。*

24.3.6 PWM 输入模式（仅适用于 TIM9/12）

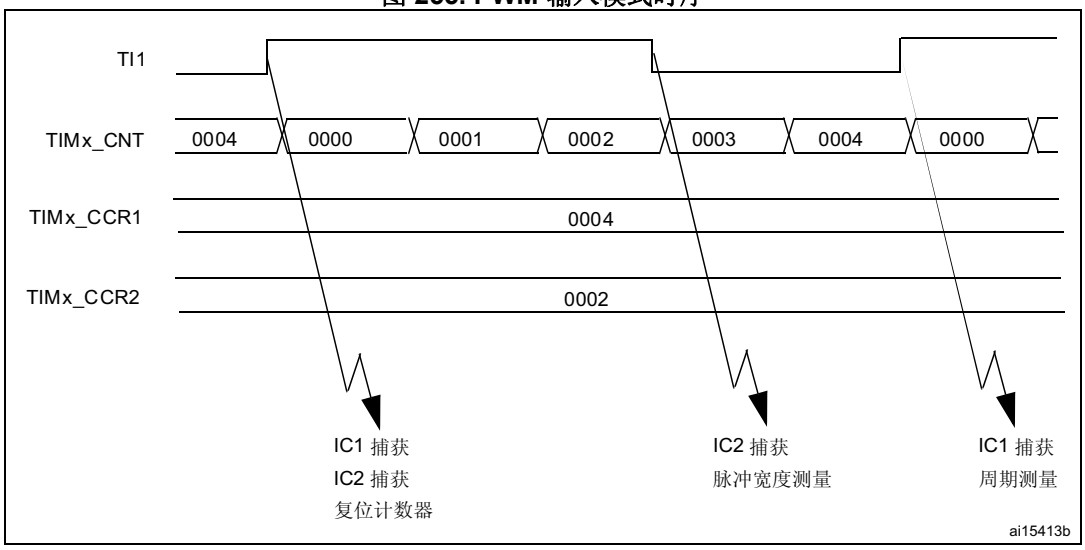
此模式是输入捕获模式的一个特例。其实现步骤与输入捕获模式基本相同，仅存在以下不同之处：

- 两个 ICx 信号被映射至同一个 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。

例如，可通过以下步骤对应用于 TI1 的 PWM 的周期（位于 TIMx_CCR1 寄存器中）和占空比（位于 TIMx_CCR2 寄存器中）进行测量（取决于 CK_INT 频率和预分频器的值）：

1. 选择 TIMx_CCR1 的有效输入：向 TIMx_CCMR1 寄存器中的 CC1S 位写入“01”（选择 TI1）。
2. 选择 TI1FP1 的有效极性（同时用于 TIMx_CCR1 中的捕获和计数器清零）：将 CC1P 位和 CC1NP 位编程为“00”（上升沿有效）。
3. 选择 TIMx_CCR2 的有效输入：向 TIMx_CCMR1 寄存器中的 CC2S 位写入“10”（选择 TI1）。
4. 选择 TI1FP2 的有效极性（用于 TIMx_CCR2 中的捕获）：将 CC2P 位和 CC2NP 位编程为“11”（下降沿有效）。
5. 选择有效触发输入：向 TIMx_SMCR 寄存器中的 TS 位写入“101”（选择 TI1FP1）。
6. 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”。
7. 使能捕获：向 TIMx_CCER 寄存器中的 CC1E 位和 CC2E 位写入“1”。

图 255. 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 TI1FP1 和 TI2FP2 与从模式控制器相连。

24.3.7 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（OCxREF 和 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 (OCXREF/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”。OCXREF 进而强制设置为高电平（OCxREF 始终为高电平有效），同时 OCx 获取 CCxP 极性位的相反值。

例如：CCxP=“0”（OCx 高电平有效）=> 将 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 OCxREF 信号强制设置为低电平。

无论如何，TIMx_CCRx 影子寄存器与计数器之间的比较仍会执行，而且允许将标志置 1。因此可相应发送中断请求。下面的输出比较模式一节对此进行了介绍。

24.3.8 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：

1. 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”）。
2. 将中断状态寄存器中的标志置 1（TIMx_SR 寄存器中的 CCxIF 位）。
3. 如果相应中断使能位（TIMx_DIER 寄存器中的 CCXIE 位）置 1，将生成中断。

使用 TIMx_CCMRx 寄存器中的 OCxPE 位，可将 TIMx_CCRx 寄存器配置为带或不带预装载寄存器。

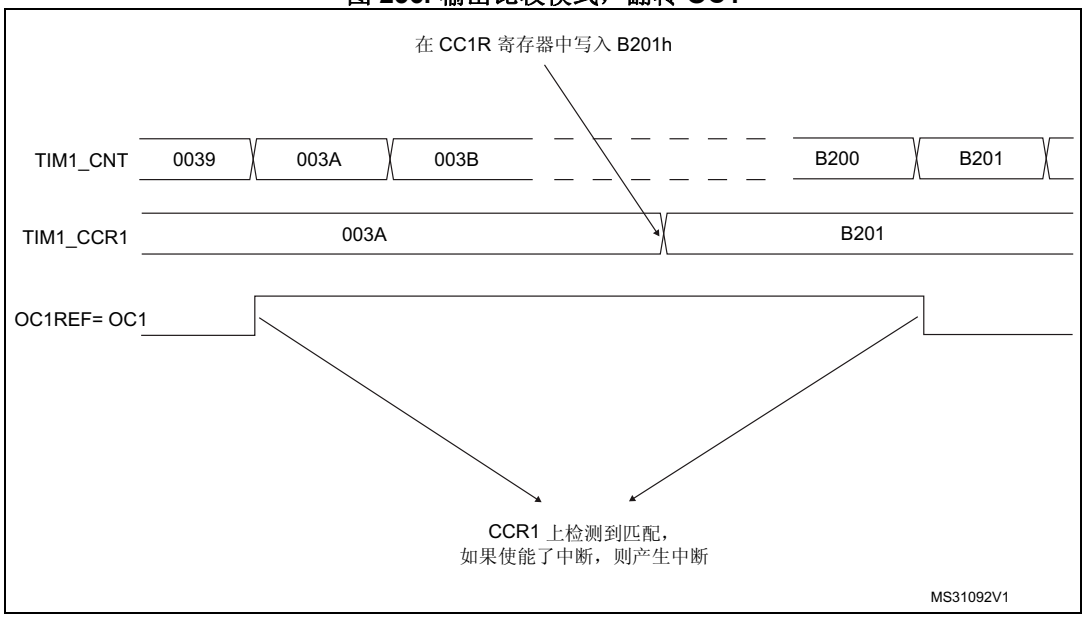
在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出毫无影响。同步的精度可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

步骤：

1. 选择计数器时钟（内部、外部、预分频器）。
2. 在 TIMx_ARR 和 TIMx_CCRx 寄存器中写入所需数据。
3. 如果要生成中断请求，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 =“011”以翻转 OCx 输出引脚。
   - 写入 OCxPE =“0”以禁止预装载寄存器
   - 写入 CCxP =“0”以选择高电平有效极性
   - 写入 CCxE =“1”以使能输出
5. 通过将 TIMx_CR1 寄存器中的 CEN 位置 1 来使能计数器。

可通过软件随时更新 TIMx_CCRx 寄存器以控制输出波形，前提是未使能预加载寄存器（OCxPE=“0”，否则 TIMx_CCRx 影子寄存器仅在下一更新事件 UEV 发生时进行更新）。*图 256* 给出了一个示例。

图 256. 输出比较模式，翻转 OC1



24.3.9 PWM 模式

脉冲宽度调制模式可以生成一个信号，该信号频率由 TIMx_ARR 寄存器值决定，其占空比则由 TIMx_CCRx 寄存器值决定。

各通道可以独立选择 PWM 模式（每个 OCx 输出对应一个 PWM），只需向 TIMx_CCMRx 寄存器的 OCxM 位写入“110”（PWM 模式 1）或“111”（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。

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。OCx 输出通过将 TIMx_CCER 寄存器中的 CCxE 位置 1 来使能。有关详细信息，请参见 TIMx_CCERx 寄存器说明。

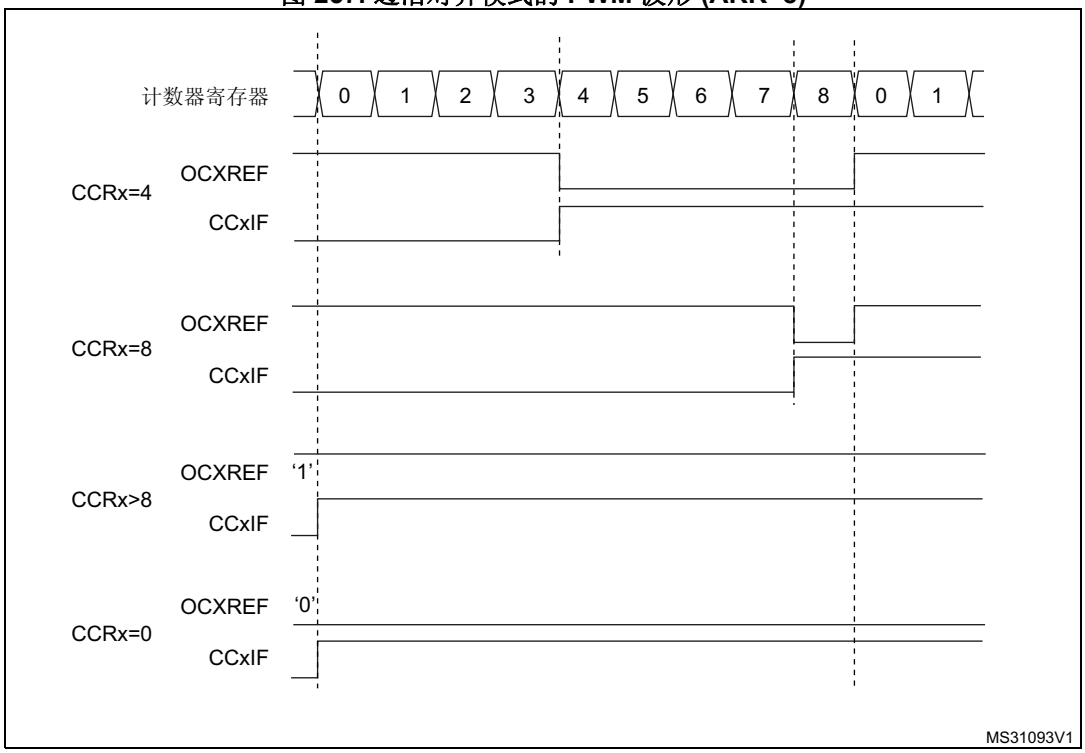
在 PWM 模式（1 或 2）下，TIMx_CNT 始终与 TIMx_CCRx 进行比较，以确定是 TIMx_CNT ≤ TIMx_CCRx。

因为计数器采用递增方式计数，所以定时器能够在边沿对齐模式下生成 PWM。

PWM 边沿对齐模式

以下以 PWM 模式 1 为例。只要 TIMx_CNT < TIMx_CCRx，PWM 参考信号 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 OCxREF 保持为“1”。如果比较值为 0，则 OCxRef 保持为“0”。*图 257* 举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

图 257. 边沿对齐模式的 PWM 波形 (ARR=8)



24.3.10 单脉冲模式

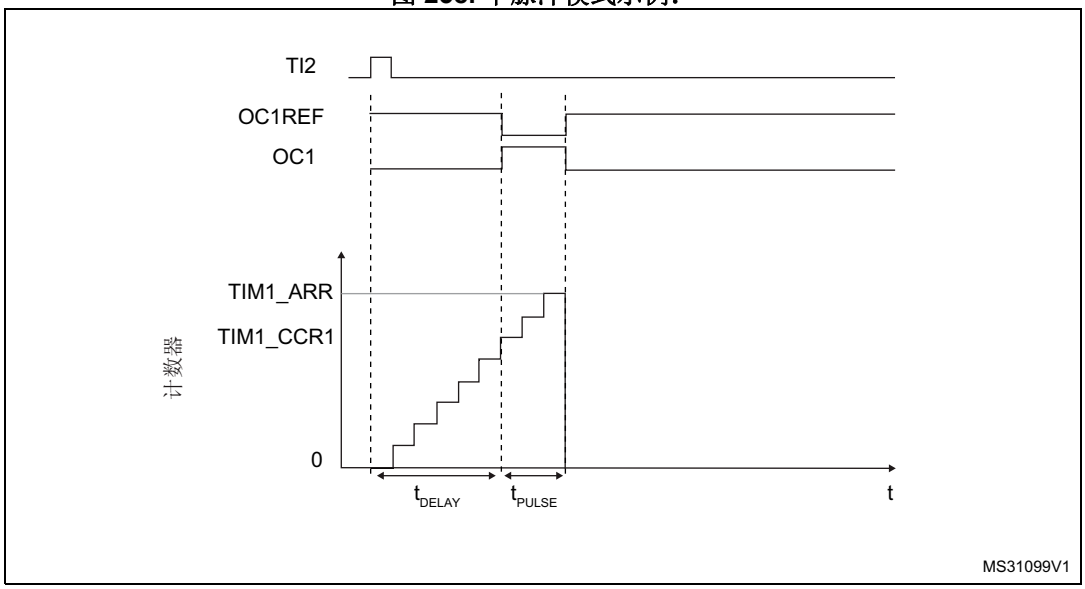
单脉冲模式 (OPM) 是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。将 TIMx_CR1 寄存器中的 OPM 位置 1，即可选择单脉冲模式。这样，发生下一更新事件 UEV 时，计数器将自动停止。

只有当比较值与计数器初始值不同时，才能正确产生一个脉冲。启动前（定时器等待触发时），必须进行如下配置：

CNT < CCRx ≤ ARR（特别注意，0 < CCRx）

图 258. 单脉冲模式示例：



例如，用户希望达到这样的效果：在 TI2 输入引脚检测到上升沿时，经过 t_{DELAY} 的延迟，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 TI2FP2 作为触发 1：

1. 在 TIMx_CCMR1 寄存器中写入 CC2S=“01”，以将 TI2FP2 映射到 TI2。
2. 在 TIMx_CCER 寄存器中写入 CC2P=“0”和 CC2NP=“0”，使 TI2FP2 能够检测上升沿。
3. 在 TIMx_SMCR 寄存器中写入 TS=“110”，以将 TI2FP2 配置为从模式控制器的触发 (TRGI)。
4. 在 TIMx_SMCR 寄存器中写入 SMS=“110”（触发模式），以使用 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，应在 TIMx_CCMR1 寄存器中写入 OC1M=“111”，以使能 PWM 模式 2。如果需要，可选择在 TIMx_CCMR1 寄存器的 OC1PE 和 TIMx_CR1 寄存器的 ARPE 中写入“1”，以使能预装载寄存器。这种情况下，必须在 TIMx_CCR1 寄存器中写入比较值并在 TIMx_ARR 寄存器中写入自动重载值，通过将 UG 位置 1 来产生更新，然后等待 TI2 上的外部触发事件。本例中，CC1P 的值为“0”。

由于仅需要 1 个脉冲（单脉冲模式），因此应向 TIMx_CR1 寄存器的 OPM 位写入“1”，以便在发生下一更新事件（计数器从自动重载值返回到 0）时使计数器停止计数。TIMx_CR1 寄存器中的 OPM 位置“0”时，即选择重复模式。

特殊情况：OCx 快速使能

在单脉冲模式下，TIx 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时，将切换输出。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟（t_{DELAY} 最小值）。

如果要输出延迟时间最短的波形，可以将 TIMx_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 OCxRef（和 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。

24.3.11 TIM9/12 外部触发同步

TIM9/12 定时器可与外部触发以下列模式实现同步：复位模式、门控模式和触发模式。

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。

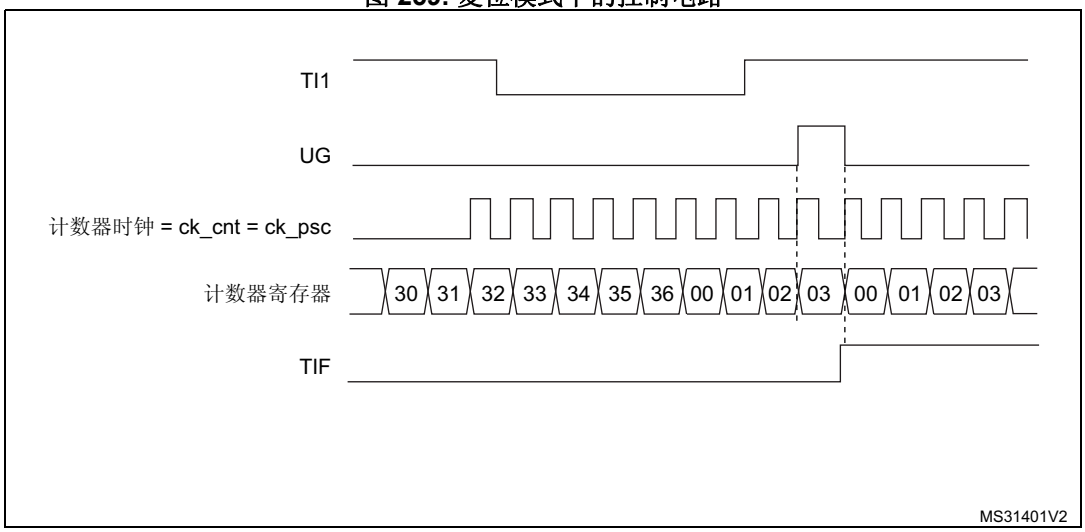
在以下示例中，TI1 输入上出现上升沿时，递增计数器清零：

1. 将通道 1 配置为检测 TI1 的上升沿。配置输入滤波带宽（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”。将 TIMx_CCER 寄存器中的 CC1P 和 CC1NP 编程为“00”，以验证极性（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=“101”，选择 TI1 作为输入源。
3. 通过在 TIMx_CR1 寄存器中写入 CEN=“1”来启动动计数器。

计数器开始根据内部时钟计数，然后正常运转，直到出现 TI1 上升沿。当 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，还可发送中断请求（取决于 TIMx_DIER 寄存器中的 TIE 位）。

下图显示了自动重载寄存器 TIMx_ARR=0x36 时的相关行为。TI1 的上升沿与实际计数器复位之间的延迟是由于 TI1 输入的重新同步电路引起的。

图 259. 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。

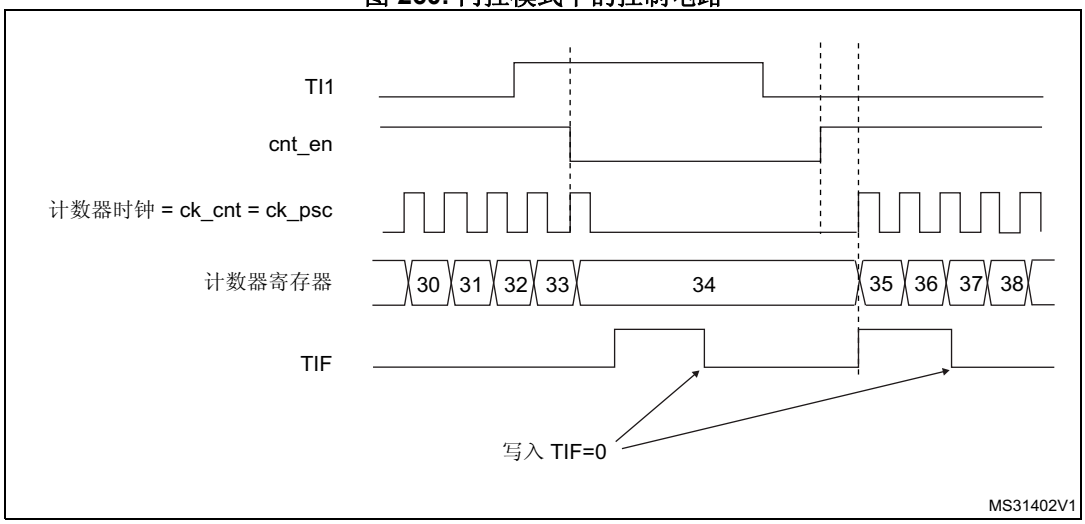
在以下示例中，递增计数器仅在 TI1 输入为低电平时计数：

1. 将通道 1 配置为检测 TI1 上的低电平。配置输入滤波带宽（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”。将 TIMx_CCER 寄存器中的 CC1P 和 CC1NP 分别编程为“1”和“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=“101”，选择 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”，则不论触发输入电平如何，计数器都不启动）。

只要 TI1 为低电平，计数器就开始根据内部时钟计数，直到 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。

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

图 260. 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。

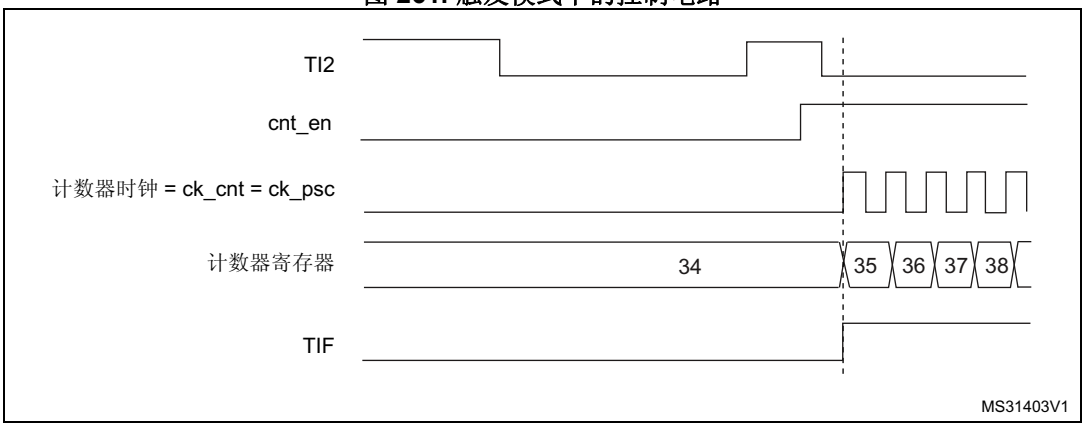
在以下示例中，TI2 输入上出现上升沿时，递增计数器启动：

1. 将通道 2 配置为检测 TI2 上的上升沿。配置输入滤波带宽（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”。将 TIMx_CCER 寄存器中的 CC2P 和 CC2NP 分别编程为“1”和“0”，以确定极性（仅检测低电平）。
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=“110”，选择 TI2 作为输入源。

当 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。

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

图 261. 触发模式下的控制电路



24.3.12 定时器同步 (TIM9/12)

TIM 定时器从内部链接在一起，以实现定时器同步或级联。有关详细信息，请参见*第 612 页上的第 22.3.25 节：定时器同步*。

注： *从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。*

24.3.13 调试模式

当微控制器进入调试模式（Cortex®-M7 内核停止）时，TIMx 计数器会根据 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。有关详细信息，请参见*第 40.16.2 节：对定时器、看门狗、bxCAN 和 I^2C 的调试支持*。

24.4 TIM9 和 TIM12 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.1 节*。

外设寄存器的写访问仅支持半字（16 位）或字（32 位）。而读访问可支持字节（8 位）、半字（16 位）或字（32 位）。

24.4.1 TIM9/12 控制寄存器 1 (TIMx_CR1)

TIM9/12 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.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
						rw	rw	rw				rw	rw	rw	rw

位 15:10 保留，必须保持复位值。

位 9:8 **CKD**：时钟分频 (Clock division)

此位字段指示定时器时钟 (CK_INT) 频率与数字滤波器所使用的采样时钟 (TIx) 之间的分频比

00：$t_{DTS} = t_{CK_INT}$

01：$t_{DTS} = 2 \times t_{CK_INT}$

10：$t_{DTS} = 4 \times t_{CK_INT}$

11：保留

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)

此位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能后，所有以下事件都会生成更新中断：

– 计数器上溢

– 将 UG 位置 1

1：使能后，只有计数器上溢会生成更新中断。

位 1 **UDIS**：更新禁止 (Update disable)

此位由软件置 1 和清零，用以使能/禁止更新事件 (UEV) 生成。

0：使能 UEV。UEV 可通过以下事件之一生成：

– 计数器上溢

– 将 UG 位置 1

然后更新影子寄存器的值。

1：禁止 UEV。不会生成 UEV，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。如果 UG 位置 1，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。

24.4.2 TIM9/12 从模式控制寄存器 (TIMx_SMCR)

TIM9/12 slave mode control register

偏移地址：0x08

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.	MSM	TS[2:0]			Res.	SMS[2:0]		
								rw	rw	rw	rw		rw	rw	rw

位 15:8 保留，必须保持复位值。

位 7 **MSM**：主/从模式 (Master/Slave mode)

0：不执行任何操作

1：当前定时器的触发输入事件 (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 TRGO）。此设置适用于要在发生单个外部事件时对多个定时器进行同步的情况。

位 6:4 **TS**：触发选择 (Trigger selection)

此位字段可选择将要用于同步计数器的触发输入。

000：内部触发 0 (ITR0)
001：内部触发 1 (ITR1)
010：内部触发 2 (ITR2)
011：内部触发 3 (ITR3)
100：TI1 边沿检测器 (TI1F_ED)
101：滤波后的定时器输入 1 (TI1FP1)
110：滤波后的定时器输入 2 (TI2FP2)
111：保留。

有关各定时器 ITRx 含义的详细信息，请参见*第 740 页上的表 133：TIMx 内部触发连接*。

注：　这些位只能在未使用的情况下（例如，SMS= “000”时）进行更改，以避免转换时出现错误的边沿检测。

位 3 保留，必须保持复位值。

位 2:0 **SMS**：从模式选择 (Slave mode selection)

选择外部信号时，触发信号 (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）。

000：禁止从模式——如果 CEN = “1”，预分频器时钟直接由内部时钟提供
001：保留
010：保留
011：保留
100：复位模式——在出现所选触发输入 (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
101：门控模式——触发输入 (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是受控的。
110：触发模式——触发信号 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
111：外部时钟模式 1——在所选触发 (TRGI) 的上升沿驱动计数器

注：　如果将 TI1F_ED 选作触发输入（TS= “100”），则不得使用门控模式。实际上，TI1F 每次转换时，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：　从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。

表 133. TIMx 内部触发连接

从 TIM	ITR0（TS = “000”）	ITR1（TS = “001”）	ITR2（TS = “010”）	ITR3（TS = “011”）
TIM9	TIM2	TIM3	TIM10_OC	TIM11_OC
TIM12	TIM4	TIM5	TIM13_OC	TIM14_OC

24.4.3 TIM9/12 中断使能寄存器 (TIMx_DIER)

TIM9/12 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.	Res.	Res.	TIE	Res.	Res.	Res.	CC2IE	CC1IE	UIE
									rw				rw	rw	rw

位 15:7 保留，必须保持复位值。

位 6 **TIE**：触发中断使能 (Trigger interrupt enable)
0：禁止触发 (TGRI) 中断。
1：使能触发 (TGRI) 中断。

位 5:3 保留，必须保持复位值。

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。

24.4.4 TIM9/12 状态寄存器 (TIMx_SR)

TIM9/12 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.	CC2OF	CC1OF	Res.	Res.	TIF	Res.	Res.	Res.	CC2IF	CC1IF	UIF
					rc_w0	rc_w0			rc_w0				rc_w0	rc_w0	rc_w0

位 15:11 保留，必须保持复位值。

位 10 **CC2OF**：捕获/比较 2 重复捕获标志 (Capture/compare 2 overcapture flag)
请参见 CC1OF 说明。

位 9 **CC1OF**：捕获/比较 1 重复捕获标志 (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：未检测到重复捕获。
1：TIMx_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。

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 TRGI 输入上检测到有效边沿时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动或停止时置 1。但需要通过软件清零。
0：未发生触发事件。
1：触发中断挂起。

位 5:3 保留，必须保持复位值。

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)
如果通道 CC1 配置为输出：
当计数器与比较值匹配时，此标志由硬件置 1。但需要通过软件清零。
0：不匹配。
1：TIMx_CNT 计数器的值与 TIMx_CCR1 寄存器的值匹配。当 TIMx_CCR1 的值大于 TIMx_ARR 的值时，CC1IF 位将在计数器发生上溢时变为高电平。
如果通道 CC1 配置为输入：
此位将在发生捕获事件时由硬件置 1。通过软件或读取 TIMx_CCR1 寄存器将该位清零。
0：未发生输入捕获事件。
1：TIMx_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）。

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：
– 上溢且当 TIMx_CR1 寄存器中 UDIS =“0”时。
– 当由于 TIMx_CR1 寄存器中 URS =“0”且 UDIS =“0”而通过软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。
– 当由于 TIMx_CR1 寄存器中 URS =“0”且 UDIS =“0”而通过触发事件（请参见同步控制寄存器说明）重新初始化 CNT 时。

24.4.5 TIM9/12 事件生成寄存器 (TIMx_EGR)

TIM9/12 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.	TG	Res.	Res.	Res.	CC2G	CC1G	UG
									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)
此位由软件置 1 以生成事件，并由硬件自动清零。
0：不执行任何操作。
1：TIMx_SR 寄存器中的 TIF 标志置 1。使能后可发生相关中断。

位 5:3 保留，必须保持复位值。

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)
此位由软件置 1 以生成事件，并由硬件自动清零。
0：不执行任何操作。
1：通道 1 上生成捕获/比较事件：
如果通道 CC1 配置为输出：
使能后，CC1IF 标志置 1 并发送相应中断。
如果通道 CC1 配置为输入：
TIMx_CCR1 寄存器中将捕获到计数器的当前值。使能后，CC1IF 标志置 1 并发送相应中断。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：不执行任何操作。
1：重新初始化计数器并生成寄存器更新事件。预分频器计数器也将清零，但预分频比不受影响。计数器清零。

24.4.6 TIM9/12 捕获/比较模式寄存器 1 (TIMx_CCMR1)

TIM9/12 capture/compare mode register 1

偏移地址：0x18

复位值：0x0000

这些通道可用于输入（捕获模式）或输出（比较模式）模式。通道方向通过配置相应的 CCxS 位进行定义。该寄存器的所有其它位在输入模式和输出模式下的功能不同。对于任一给定位，OCxx 用于说明通道配置为输出模式时该位对应的功能，ICxx 则用于说明通道配置为输入模式时该位对应的功能。因此，必须注意同一个位在输入阶段和输出阶段具有不同的含义。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		Res.	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[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 15 保留，必须保持复位值。

位 14:12 **OC2M[2:0]**：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)

此位字段定义通道方向（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，IC2 映射到 TI2 上

10：CC2 通道配置为输入，IC2 映射到 TI1 上

11：CC2 通道配置为输入，IC2 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注： 仅当通道关闭时（TIMx_CCER 中的 CC2E = 0），才可向 CC2S 位写入数据。

位 7 保留，必须保持复位值。

位 6:4 **OC1M**：输出比较 1 模式 (Output Compare 1 mode)

这些位定义提供 OC1 和 OC1N 的输出参考信号 OC1REF 的行为。OC1REF 为高电平有效，而 OC1 和 OC1N 的有效电平则分别取决于 CC1P 位和 CC1NP 位。

000：冻结——输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。

001：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，OC1REF 信号强制变为高电平。

010：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，OC1REF 信号强制变为低电平。

011：翻转——TIMx_CNT=TIMx_CCR1 时，OC1REF 发生翻转

100：强制变为无效电平——OC1REF 强制变为低电平

101：强制变为有效电平——OC1REF 强制变为高电平

110：PWM 模式 1——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为无效状态（OC1REF=“0”），否则为有效状态（OC1REF=“1”）

111：PWM 模式 2——在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为无效状态，否则为有效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。

注：　在 PWM 模式 1 或 PWM 模式 2 下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“PWM”模式时，OCREF 电平才会发生更改。

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活动寄存器中

注：　只有单脉冲模式下才可在未验证预装载寄存器的情况下使用 PWM 模式（TIMx_CR1 寄存器中的 OPM 位置 1）。其它情况下则无法保证该行为。

位 2 **OC1FE**：输出比较 1 快速使能 (Output Compare 1 fast enable)

此位用于加快触发输入事件对 CC 输出的影响。

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，OC1FE 才会起作用。

位 1:0 **CC1S**：捕获/比较 1 选择 (Capture/Compare 1 Selection)

此位字段定义通道方向（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，IC1 映射到 TI1 上

10：CC1 通道配置为输入，IC1 映射到 TI2 上

11：CC1 通道配置为输入，IC1 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注：　仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

输入捕获模式

位 15:12 **IC2F**：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**：捕获/比较 2 选择 (Capture/compare 2 selection)

此位字段定义通道方向（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，IC2 映射到 TI2 上
10：CC2 通道配置为输入，IC2 映射到 TI1 上
11：CC2 通道配置为输入，IC2 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注：　仅当通道关闭时（TIMx_CCER 中的 CC2E = 0），才可向 CC2S 位写入数据。

位 7:4 **IC1F**：输入捕获 1 滤波器 (Input capture 1 filter)

此位字段可定义 TI1 输入的采样频率和适用于 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1000：$f_{SAMPLING}=f_{DTS}/8$，N=6
0001：$f_{SAMPLING}=f_{CK_INT}$，N=21001：$f_{SAMPLING}=f_{DTS}/8$，N=8
0010：$f_{SAMPLING}=f_{CK_INT}$，N=41010：$f_{SAMPLING}=f_{DTS}/16$，N=5
0011：$f_{SAMPLING}=f_{CK_INT}$，N=81011：$f_{SAMPLING}=f_{DTS}/16$，N=6
0100：$f_{SAMPLING}=f_{DTS}/2$，N=61100：$f_{SAMPLING}=f_{DTS}/16$，N=8
0101：$f_{SAMPLING}=f_{DTS}/2$，N=81101：$f_{SAMPLING}=f_{DTS}/32$，N=5
0110：$f_{SAMPLING}=f_{DTS}/4$，N=61110：$f_{SAMPLING}=f_{DTS}/32$，N=6
0111：$f_{SAMPLING}=f_{DTS}/4$，N=81111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 3:2 **IC1PSC**：输入捕获 1 预分频器 (Input capture 1 prescaler)

此位字段定义 CC1 输入 (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**：捕获/比较 1 选择 (Capture/Compare 1 Selection)

此位字段定义通道方向（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，IC1 映射到 TI1 上
10：CC1 通道配置为输入，IC1 映射到 TI2 上
11：CC1 通道配置为输入，IC1 映射到 TRC 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注：　仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

24.4.7 TIM9/12 捕获/比较使能寄存器 (TIMx_CCER)

TIM9/12 capture/compare enable register

偏移地址：0x20

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
								rw		rw	rw	rw		rw	rw

位 15:8 保留，必须保持复位值。

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：CC1NP 必须保持清零
CC1 通道配置为输入：CC1NP 与 CC1P 配合使用可定义 TI1FP1/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)。
CC1 通道配置为输出：
0：OC1 高电平有效。
1：OC1 低电平有效。
CC1 通道配置为输入：
CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的极性。
00：未反相/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）。
01：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）。
10：保留，不使用此配置。

注： 11：未反相/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。

位 0 **CC1E**：捕获/比较 1 输出使能 (Capture/Compare 1 output enable)。
CC1 通道配置为输出：
0：关闭——OC1 未激活。
1：开启——在相应输出引脚上输出 OC1 信号。
CC1 通道配置为输入：
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 (TIMx_CCR1) 中。
0：禁止捕获。
1：使能捕获。

表 134. 标准 OCx 通道的输出控制位

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

注： *与标准 OCx 通道相连的外部 I/O 引脚的状态取决于通道 OCx 的状态以及 GPIO 寄存器。*

24.4.8 TIM9/12 计数器 (TIMx_CNT)

TIM9/12 counter

偏移地址：0x24

复位值：0x0000 0000

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 15:0 **CNT[15:0]**：计数器值 (Counter value)

24.4.9 TIM9/12 预分频器 (TIMx_PSC)

TIM9/12 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 CK_CNT 等于 f_{CK_PSC} / (PSC[15:0] + 1)。

PSC 包含在每次发生更新事件时要装载到实际预分频器寄存器的值。

24.4.10 TIM9/12 自动重载寄存器 (TIMx_ARR)

TIM9/12 auto-reload register

偏移地址：0x2C

复位值：0x0000

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 15:0 **ARR[15:0]**：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 719 页上的第 24.3.1 节：时基单元*。

当自动重载值为空时，计数器不工作。

24.4.11 TIM9/12 捕获/比较寄存器 1 (TIMx_CCR1)

TIM9/12 capture/compare register 1

偏移地址：0x34

复位值：0x0000

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 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（拷贝到实际起作用的捕获/比较寄存器 1）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC1 输出上发出信号的值。

如果通道 CC1 配置为输入：

CCR1 为上一个输入捕获 1 事件 (IC1) 发生时的计数器值。

24.4.12 TIM9/12 捕获/比较寄存器 2 (TIMx_CCR2)

TIM9/12 capture/compare register 2

偏移地址：0x38

复位值：0x0000

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 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（拷贝到实际起作用的捕获/比较寄存器 2）

活动捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC2 输出上发出信号的值。

如果通道 CC2 配置为输入：

CCR2 为上一个输入捕获 2 事件 (IC2) 发生时的计数器值。

24.4.13 TIM9/12 寄存器映射

TIM9/12 寄存器可映射为 16 位可寻址寄存器，如下表所述：

表 135. TIM9/12 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	**TIMx_CR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD [1:0]		ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
	Reset value																							0	0	0				0	0	0	0
0x08	**TIMx_SMCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MSM	TS[2:0]			Res.	SMS[2:0]		
	Reset value																									0	0	0	0		0	0	0
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.	Res.	Res.	TIE	Res.	Res.	Res.	CC2IE	CC1IE	UIE
	Reset value																										0				0	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.	CC2OF	CC1OF	Res.	Res.	TIF	Res.	Res.	Res.	CC2IF	CC1IF	UIF
	Reset value																						0	0			0				0	0	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.	TG	Res.	Res.	Res.	CC2G	CC1G	UG
	Reset value																										0				0	0	0

表 135. TIM9/12 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
0x18	**TIMx_CCMR1** Output Compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M [2:0]			OC2PE	OC2FE	CC2S [1:0]		Res.	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
	Reset value																		0	0	0	0	0	0	0		0	0	0	0	0	0	0
	TIMx_CCMR1 Input Capture mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2F[3:0]				IC2PSC [1:0]		CC2S [1:0]		IC1F[3:0]				IC1PSC [1:0]		CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	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.
0x20	**TIMx_CCER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
	Reset value																									0		0	0	0		0	0
0x24	**TIMx_CNT**	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
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**	Reserved																ARR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	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.
0x34	**TIMx_CCR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	**TIMx_CCR2**	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
0x3C to 0x4C	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.

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节：存储器映射和寄存器边界地址*。

24.5 TIM10/11/13/14 寄存器

外设寄存器的写访问仅支持半字（16 位）或字（32 位）。而读访问可支持字节（8 位）、半字（16 位）或字（32 位）。

24.5.1 TIM10/11/13/14 控制寄存器 1 (TIMx_CR1)

TIM10/11/13/14 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.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.	Res.	Res.	Res.	URS	UDIS	CEN
						rw	rw	rw					rw	rw	rw

位 15:10 保留，必须保持复位值。

位 9:8 **CKD**：时钟分频 (Clock division)

此位字段指示定时器时钟 (CK_INT) 频率与数字滤波器所使用的采样时钟 (TIx) 之间的分频比

00：$t_{DTS} = t_{CK_INT}$
01：$t_{DTS} = 2 \times t_{CK_INT}$
10：$t_{DTS} = 4 \times t_{CK_INT}$
11：保留

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 6:3 保留，必须保持复位值。

位 2 **URS**：更新请求源 (Update request source)

此位由软件置 1 和清零，用以选择更新中断 (UEV) 源。
0：使能后，所有以下事件都会生成 UEV：
- 计数器上溢
- 将 UG 位置 1

1：使能后，只有计数器上溢会生成 UEV。

位 1 **UDIS**：更新禁止 (Update disable)

此位由软件置 1 和清零，用以使能/禁止更新中断 (UEV) 事件生成。
0：使能 UEV。UEV 可通过以下事件之一生成：
- 计数器上溢
- 将 UG 位置 1

然后更新影子寄存器的值。
1：禁止 UEV。不会生成 UEV，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。如果 UG 位置 1，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

24.5.2 TIM10/11/13/14 中断使能寄存器 (TIMx_DIER)

TIM10/11/13/14 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IE	UIE
														rw	rw

位 15:2 保留，必须保持复位值。

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

24.5.3 TIM10/11/13/14 状态寄存器 (TIMx_SR)

TIM10/11/13/14 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.	CC1OF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IF	UIF
						rc_w0								rc_w0	rc_w0

位 15:10 保留，必须保持复位值。

位 9 **CC1OF**：捕获/比较 1 重复捕获标志 (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：未检测到重复捕获。
1：TIMx_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。

位 8:2 保留，必须保持复位值。

位 1 **CC1IF**：捕获/比较 1 中断标志 (Capture/compare 1 interrupt flag)

如果通道 CC1 配置为输出：

当计数器与比较值匹配时，此标志由硬件置 1。但需要通过软件清零。

0：不匹配。

1：TIMx_CNT 计数器的值与 TIMx_CCR1 寄存器的值匹配。当 TIMx_CCR1 的值大于 TIMx_ARR 的值时，CC1IF 位将在计数器发生上溢时变为高电平。

如果通道 CC1 配置为输入：

此位将在发生捕获事件时由硬件置 1。通过软件或读取 TIMx_CCR1 寄存器将该位清零。

0：未发生输入捕获事件。

1：TIMx_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）。

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：

– 上溢且当 TIMx_CR1 寄存器中 UDIS = “0”时。

– 当由于 TIMx_CR1 寄存器中 URS = “0”且 UDIS = “0”而通过软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。

24.5.4 TIM10/11/13/14 事件生成寄存器 (TIMx_EGR)

TIM10/11/13/14 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.	CC1G	UG
														w	w

位 15:2 保留，必须保持复位值。

位 1 **CC1G**：捕获/比较 1 生成 (Capture/Compare 1 generation)

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作

1：通道 1 上生成捕获/比较事件：

如果通道 CC1 配置为输出：

使能后，CC1IF 标志置 1 并发送相应的中断。

如果通道 CC1 配置为输入：

TIMx_CCR1 寄存器中将捕获到计数器当前值。使能后，CC1IF 标志置 1 并发送相应中断。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：不执行任何操作

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）。计数器清零。

24.5.5 TIM10/11/13/14 捕获/比较模式寄存器 1 (TIMx_CCMR1)

TIM10/11/13/14 capture/compare mode register 1

偏移地址：0x18

复位值：0x0000

这些通道可用于输入（捕获模式）或输出（比较模式）模式。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其它位在输入模式和输出模式下的功能均不同。对于任一给定位，OCxx 用于说明通道配置为输出时该位对应的功能，ICxx 则用于说明通道配置为输入时该位对应的功能。因此，必须注意同一个位在输入阶段和输出阶段具有不同的含义。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1F[3:0]				IC1PSC[1:0]			
								rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

位 15:7 保留，必须保持复位值。

位 6:4 **OC1M**：输出比较 1 模式 (Output Compare 1 mode)

这些位定义提供 OC1 的输出参考信号 OC1REF 的行为。OC1REF 为高电平有效，而 OC1 的有效电平则取决于 CC1P 位。

000：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。

001：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，OC1REF 信号强制变为高电平。

010：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，OC1REF 信号强制变为低电平。

011：翻转——TIMx_CNT = TIMx_CCR1 时，OC1REF 发生翻转。

100：强制变为无效电平——OC1REF 强制变为低电平。

101：强制变为有效电平——OC1REF 强制变为高电平。

110：PWM 模式 1——只要 TIMx_CNT < TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。

111：PWM 模式 2——只要 TIMx_CNT < TIMx_CCR1，通道 1 便为无效状态，否则为有效状态。

注： 在 PWM 模式 1 或 PWM 模式 2 下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“PWM”模式时，OCREF 电平才会发生更改。

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活动寄存器中。

注： 只有单脉冲模式下才可在未验证预装载寄存器的情况下使用 PWM 模式（TIMx_CR1 寄存器中的 OPM 位置 1）。其它情况下则无法保证该行为。

位 2 **OC1FE**：输出比较 1 快速使能 (Output Compare 1 fast enable)

此位用于加快触发输入事件对 CC 输出的影响。

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，OC1FE 才会起作用。

位 1:0 **CC1S**：捕获/比较 1 选择 (Capture/Compare 1 Selection)

此位字段定义通道方向（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，IC1 映射到 TI1 上。

10:

11:

注： 仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

输入捕获模式

位 15:8 保留，必须保持复位值。

位 7:4 **IC1F**：输入捕获 1 滤波器 (Input capture 1 filter)

此位字段可定义 TI1 输入的采样频率和适用于 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1000：$f_{SAMPLING}=f_{DTS}/8$，N=6

0001：$f_{SAMPLING}=f_{CK_INT}$，N=21001：$f_{SAMPLING}=f_{DTS}/8$，N=8

0010：$f_{SAMPLING}=f_{CK_INT}$，N=41010：$f_{SAMPLING}=f_{DTS}/16$，N=5

0011：$f_{SAMPLING}=f_{CK_INT}$，N=81011：$f_{SAMPLING}=f_{DTS}/16$，N=6

0100：$f_{SAMPLING}=f_{DTS}/2$，N=61100：$f_{SAMPLING}=f_{DTS}/16$，N=8

0101：$f_{SAMPLING}=f_{DTS}/2$，N=81101：$f_{SAMPLING}=f_{DTS}/32$，N=5

0110：$f_{SAMPLING}=f_{DTS}/4$，N=61110：$f_{SAMPLING}=f_{DTS}/32$，N=6

0111：$f_{SAMPLING}=f_{DTS}/4$，N=81111：$f_{SAMPLING}=f_{DTS}/32$，N=8

位 3:2 **IC1PSC**：输入捕获 1 预分频器 (Input capture 1 prescaler)

此位字段定义 CC1 输入 (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**：捕获/比较 1 选择 (Capture/Compare 1 Selection)

此位字段定义通道方向（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，IC1 映射到 TI1 上

10：保留

11：保留

注： 仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

24.5.6 TIM10/11/13/14 捕获/比较使能寄存器 (TIMx_CCER)

TIM10/11/13/14 capture/compare enable register

偏移地址：0x20

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1NP	Res.	CC1P	CC1E
												rw		rw	rw

位 15:4 保留，必须保持复位值。

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：CC1NP 必须保持清零。
CC1 通道配置为输入：CC1NP 与 CC1P 配合使用可定义 TI1FP1 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)。
CC1 通道配置为输出：
0：OC1 高电平有效
1：OC1 低电平有效
CC1 通道配置为输入：
CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的极性。
00：未反相/上升沿触发
电路对 TI1FP1 上升沿敏感（捕获模式），TI1FP1 未反相。
01：反相/下降沿触发
电路对 TI1FP1 下降沿敏感（捕获模式），TI1FP1 反相。
10：保留，不使用此配置。
11：未反相/上升沿和下降沿均触发
电路对 TI1FP1 上升沿和下降沿均敏感（捕获模式），TI1FP1 未反相。

位 0 **CC1E**：捕获/比较 1 输出使能 (Capture/Compare 1 output enable)。
CC1 通道配置为输出：
0：关闭——OC1 未激活
1：开启——在相应输出引脚上输出 OC1 信号
CC1 通道配置为输入：
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 (TIMx_CCR1) 中。
0：禁止捕获
1：使能捕获

表 136. 标准 OCx 通道的输出控制位

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

注： *与标准 OCx 通道相连的外部 I/O 引脚的状态取决于通道 OCx 的状态以及 GPIO 寄存器。*

24.5.7 TIM10/11/13/14 计数器 (TIMx_CNT)

TIM10/11/13/14 counter

偏移地址：0x24

复位值：0x0000

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 15:0 **CNT[15:0]**：计数器值 (Counter value)

24.5.8 TIM10/11/13/14 预分频器 (TIMx_PSC)

TIM10/11/13/14 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 (CK_CNT) 等于 f_{CK_PSC} / (PSC[15:0] + 1)。
PSC 包含在每次发生更新事件时要装载到实际预分频器寄存器的值。

24.5.9 TIM10/11/13/14 自动重载寄存器 (TIMx_ARR)

TIM10/11/13/14 auto-reload register

偏移地址：0x2C

复位值：0x0000

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 15:0 **ARR[15:0]**：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 更新和行为的详细信息，请参见*第 719 页上的第 24.3.1 节：时基单元*。
当自动重载值为空时，计数器不工作。

24.5.10 TIM10/11/13/14 捕获/比较寄存器 1 (TIMx_CCR1)

TIM10/11/13/14 capture/compare register 1

偏移地址：0x34

复位值：0x0000

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 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（拷贝到实际起作用的捕获/比较寄存器 1）。

实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 OC1 输出上发出信号的值。

如果通道 CC1 配置为输入：

CCR1 为上一个输入捕获 1 事件 (IC1) 发生时的计数器值。

24.5.11 TIM11 选项寄存器 1 (TIM11_OR)

TIM11 option register 1

偏移地址：0x50

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.	TI1_RMP[1:0]	
														rw	

位 15:2 保留，必须保持复位值。

位 1:0 **TI1_RMP[1:0]**：TIM11 输入 1 重映射功能 (TIM11 Input 1 remapping capability)
由软件置 1 和清零。
00：TIM11 通道 1 连接到 GPIO（请参见复用功能映射）
01：SPDIFRX 帧同步
10：MSI 内部时钟（RTC 为 1 MHz）连接到 TIM11_CH1 输入，用于测量
11：MCO1 连接到 TIM11_CH1 输入

24.5.12 TIM10/11/13/14 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 137. TIM10/11/13/14 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	**TIMx_CR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD [1:0]		ARPE	Res.	Res.	Res.	Res.	URS	UDIS	CEN
	Reset value																							0	0	0					0	0	0
0x08	**TIMx_SMCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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																																
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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IE	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.	CC1OF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IF	UIF
	Reset value																							0								0	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.	CC1G	UG
	Reset value																															0	0
0x18	**TIMx_CCMR1** Output compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
	Reset value																										0	0	0	0	0	0	0
	TIMx_CCMR1 Input capture mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1F[3:0]				IC1 PSC [1:0]		CC1S [1:0]	
	Reset value																									0	0	0	0	0	0	0	0
0x1C	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.
0x20	**TIMx_CCER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1NP	Res.	CC1P	CC1E
	Reset value																													0		0	0
0x24	**TIMx_CNT**	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
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.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 137. TIM10/11/13/14 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
0x30	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.
0x34	**TIMx_CCR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38 to 0x4C	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.
0x50	TIMx_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.	TI1_RMP	
	Reset value																															0	0

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

25 基本定时器 (TIM6/TIM7)

25.1 TIM6/TIM7 简介

基本定时器 TIM6 和 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。

此类定时器不仅可用作通用定时器以生成时基，还可以专门用于驱动数模转换器 (DAC)。实际上，此类定时器内部连接到 DAC 并能够通过其触发输出驱动 DAC。

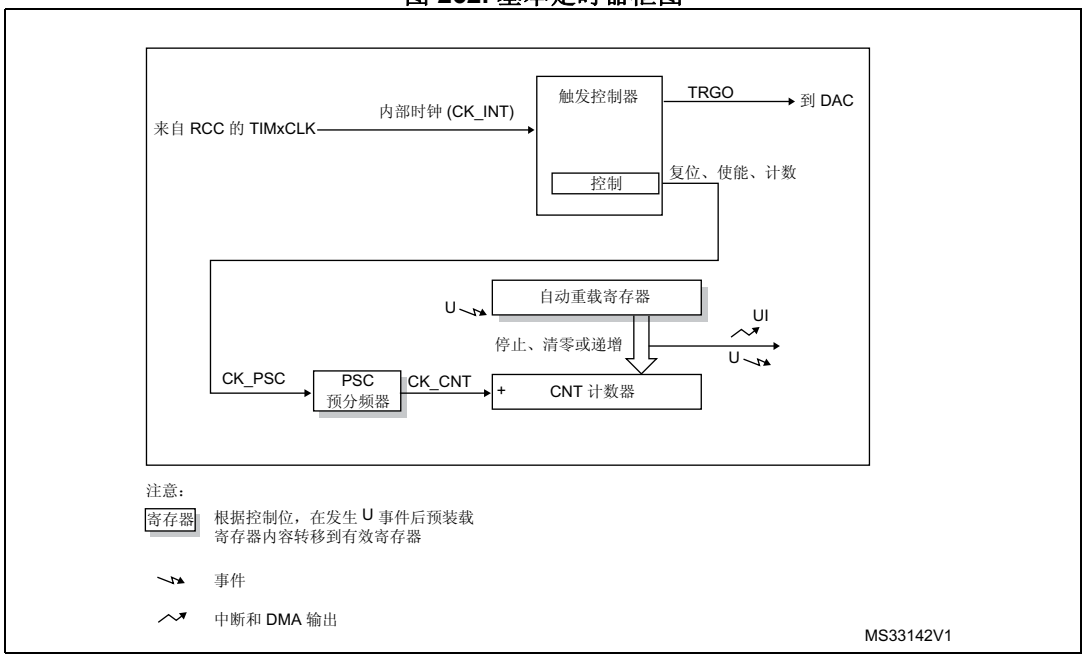
这些定时器彼此完全独立，不共享任何资源。

25.2 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

图 262. 基本定时器框图



25.3 TIM6/TIM7 功能说明

25.3.1 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。下文将针对各配置的更新事件的产生进行详细介绍。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位 (CEN) 置 1 时，才会启动计数器。

请注意，实际的计数器使能信号 CNT_EN 在 CEN 置 1 的一个时钟周期后被置 1。

预分频器说明

预分频器可对计数器时钟频率进行分频，分频系数介于 1 和 65536 之间。该预分频器基于 TIMx_PSC 寄存器中的 16 位寄存器所控制的 16 位计数器。由于 TIMx_PSC 控制寄存器有缓冲，因此可对预分频器进行实时更改。而新的预分频比将在下一更新事件发生时被采用。

图 263 和 *图 264* 以一些示例说明在预分频比实时变化时计数器的行为。

图 263. 预分频器分频由 1 变为 2 时的计数器时序图

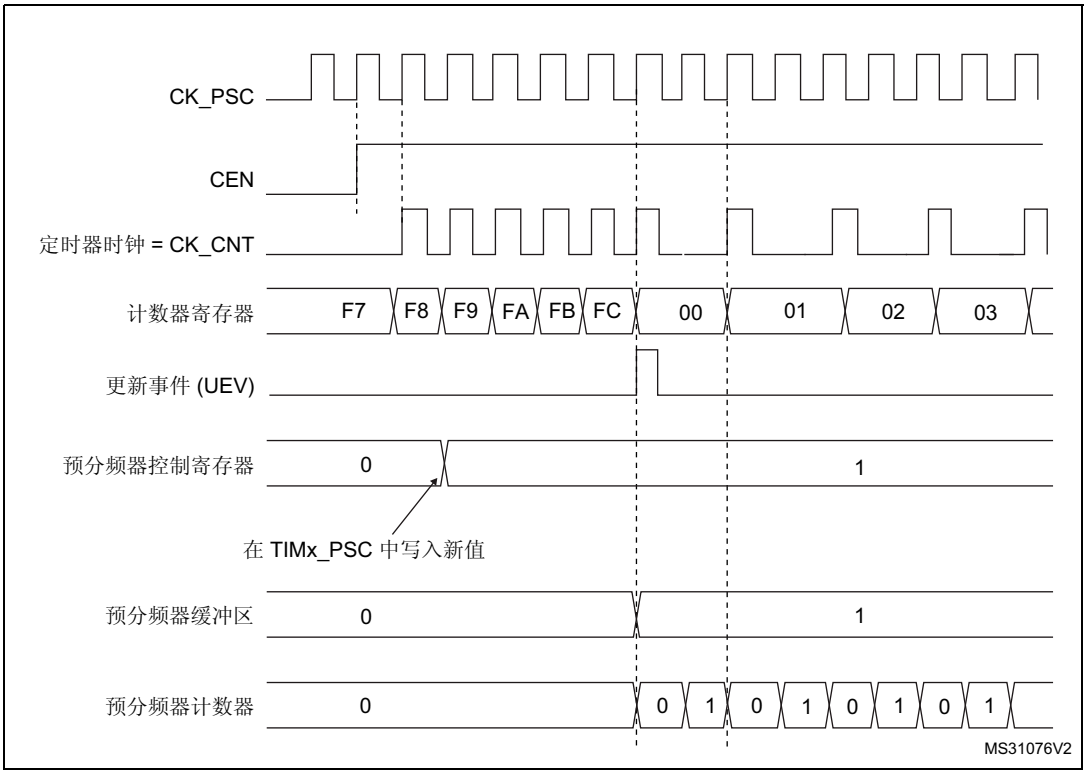
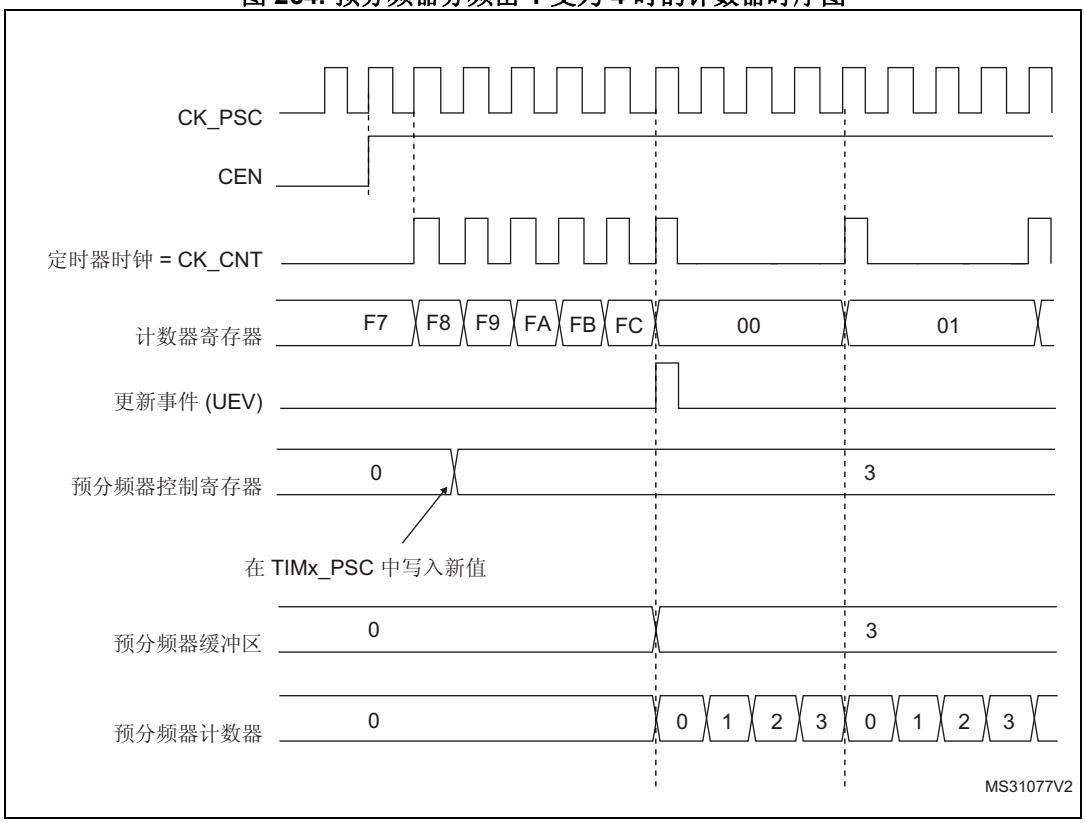


图 264. 预分频器分频由 1 变为 4 时的计数器时序图



25.3.2 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 265. 计数器时序图，1 分频内部时钟

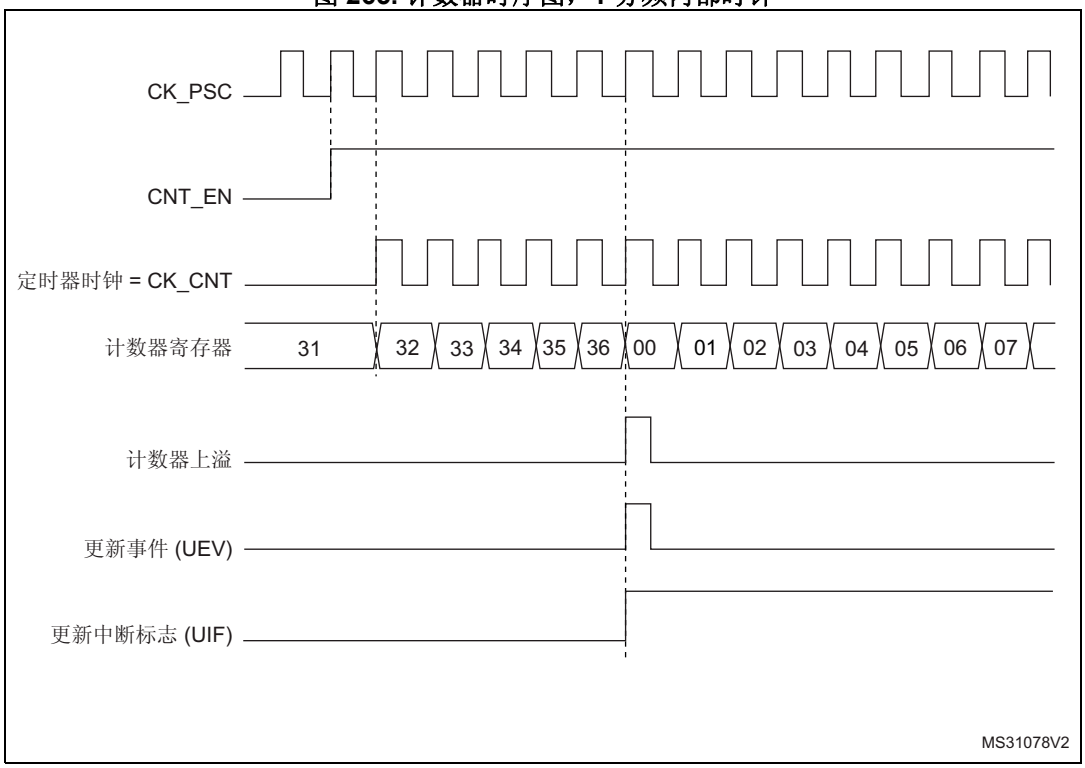


图 266. 计数器时序图，2 分频内部时钟

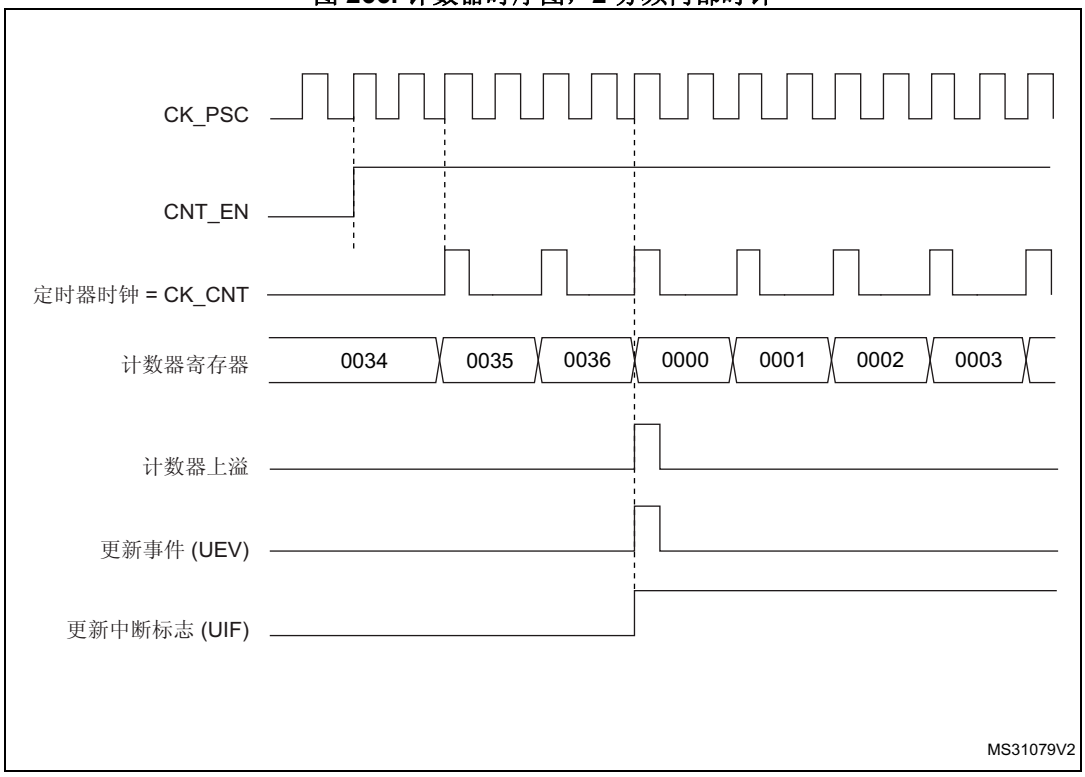


图 267. 计数器时序图，4 分频内部时钟

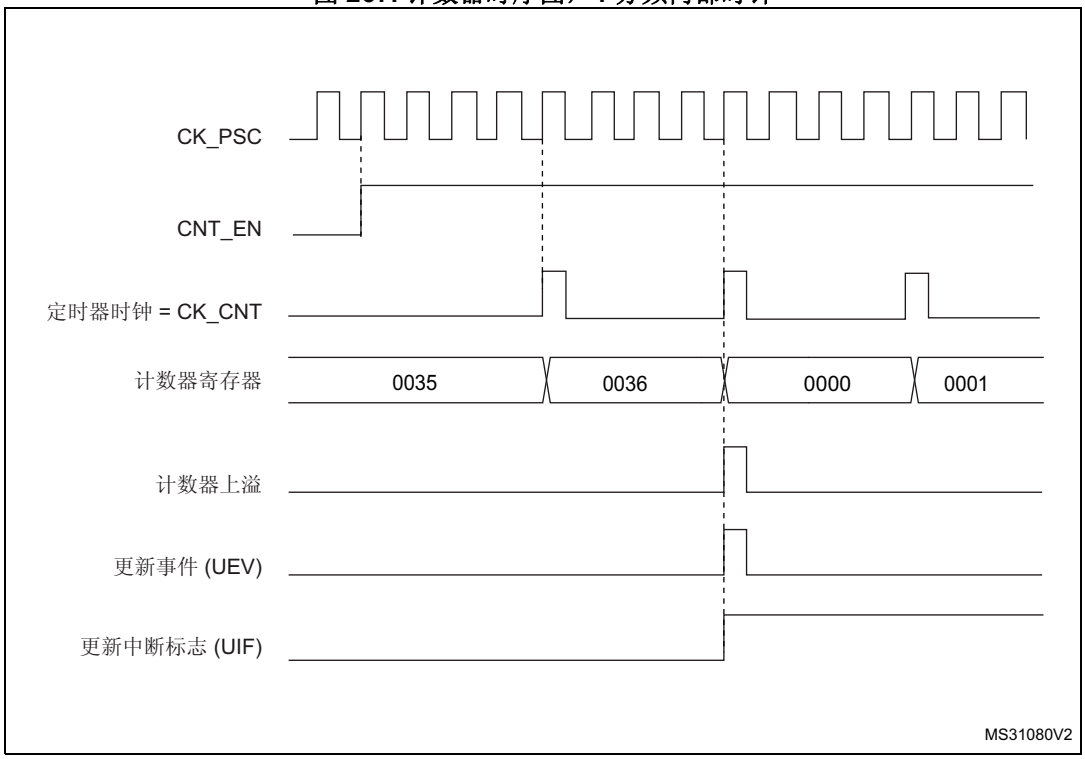


图 268. 计数器时序图，N 分频内部时钟

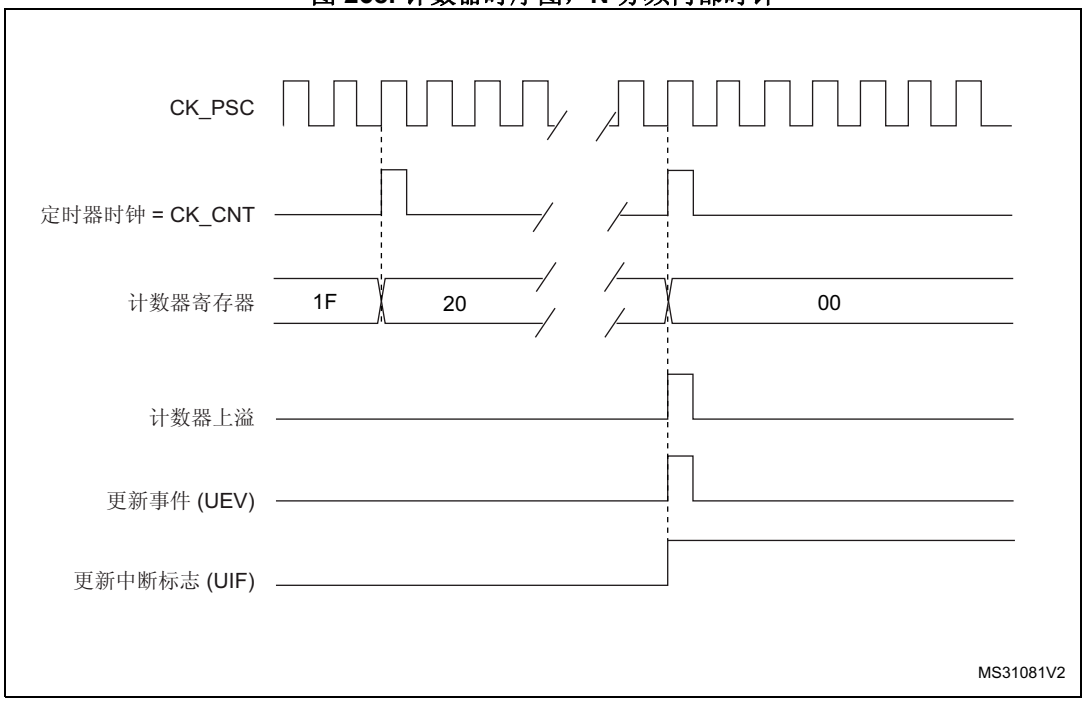


图 269. 计数器时序图，ARPE = 0 时更新事件（TIMx_ARR 未预装载）

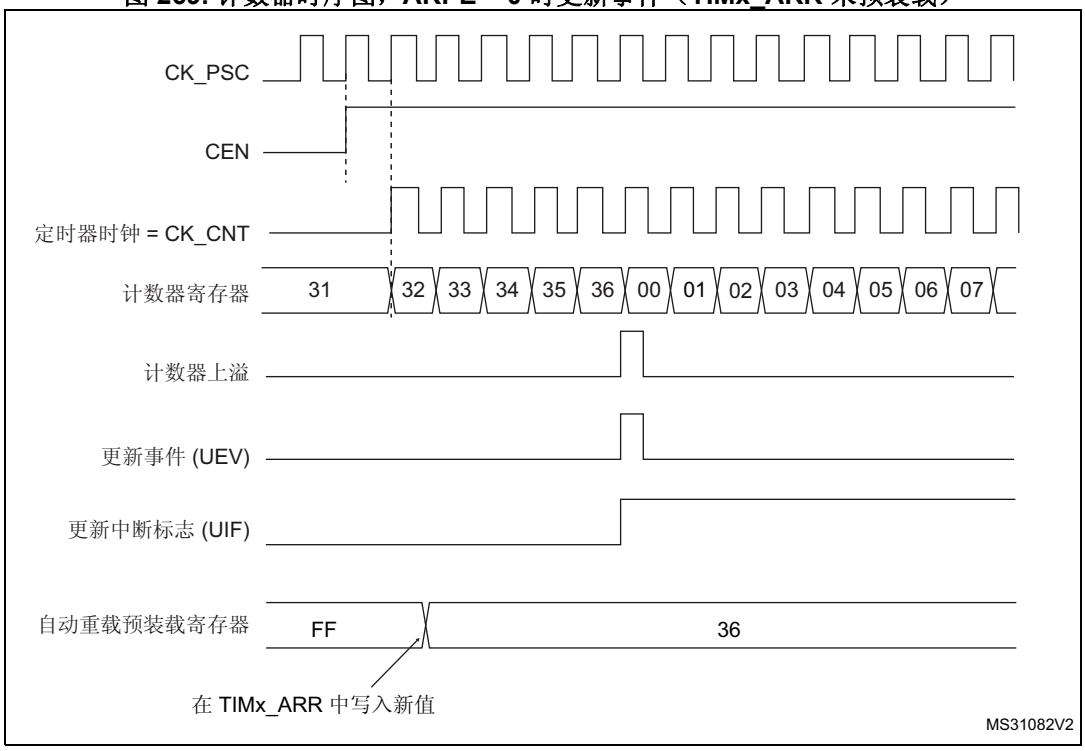
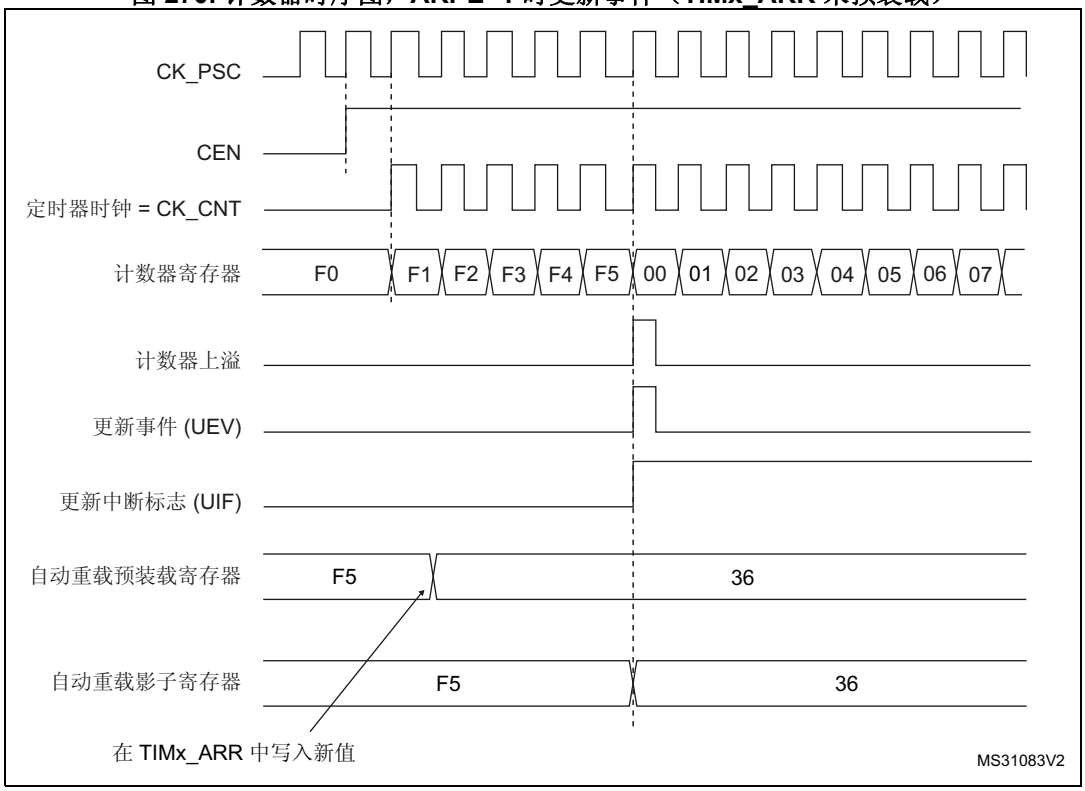


图 270. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 未预装载）



25.3.3 UIF 位重映射

TIMx_CR1 寄存器中的 IUFREMAP 位强制将更新中断标志 UIF 连续复制到定时计数器寄存器的位 31 (TIMxCNT[31]) 中。这样便可自动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 标志使能之间没有延迟。

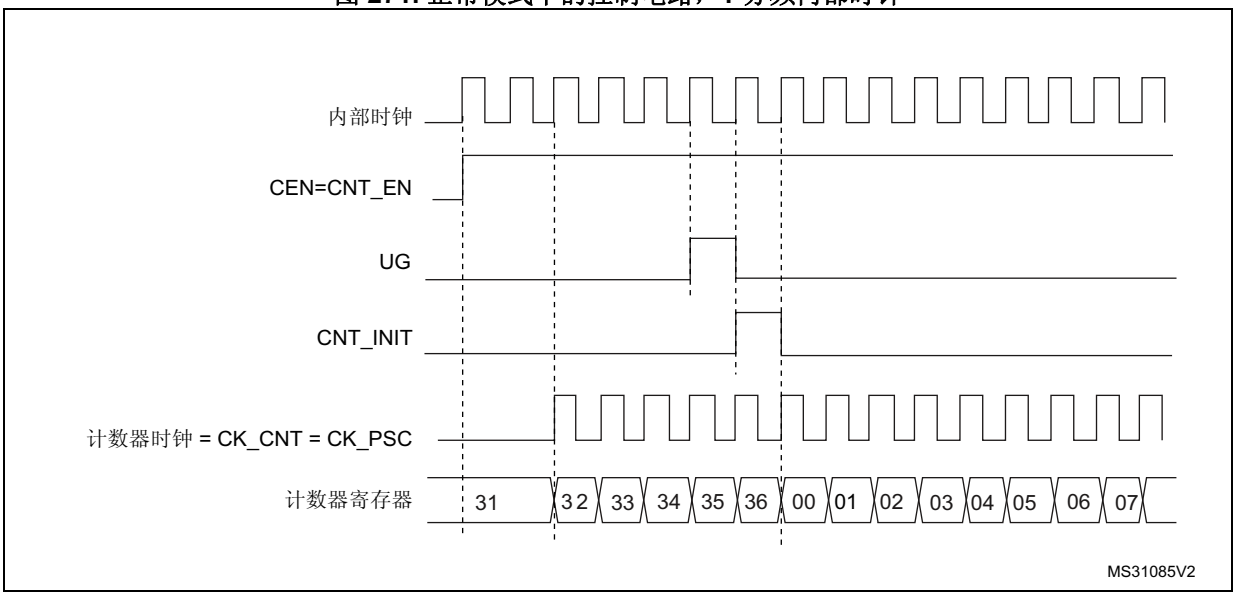
25.3.4 时钟源

计数器时钟由内部时钟 (CK_INT) 源提供。

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 CK_INT 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



25.3.5 调试模式

当微控制器进入调试模式时（Cortex®-M7 内核停止），TIMx 计数器会根据 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。有关详细信息，请参见*第 40.16.2 节：对定时器、看门狗、bxCAN 和 I²C 的调试支持*。

25.4 TIM6/TIM7 寄存器

有关寄存器说明中使用的缩写，请参见*第 58 页上的第 1.1 节*。

外设寄存器可支持半字（16 位）或字（32 位）访问。

25.4.1 TIM6/TIM7 控制寄存器 1 (TIMx_CR1)

TIM6/TIM7 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	Res	UIF RE-MAP	Res	Res	Res	ARPE	Res	Res	Res	OPM	URS	UDIS	CEN
				rw				rw				rw	rw	rw	rw

位 15:12 保留，必须保持复位值。

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)
此位由软件置 1 和清零，用以选择 UEV 事件源。
0：使能时，所有以下事件都会产生更新中断或 DMA 请求。此类事件包括：
– 计数器上溢/下溢
– 将 UG 位置 1
– 通过从模式控制器产生的更新事件
1：使能时，只有计数器上溢/下溢会产生更新中断或 DMA 请求。

位 1 **UDIS**：更新禁止 (Update disable)
此位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。
0：使能 UEV。更新 (UEV) 事件可通过以下事件之一生成：
– 计数器上溢/下溢
– 将 UG 位置 1
– 通过从模式控制器生成的更新事件
然后更新影子寄存器的值。
1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR 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
注： *只有事先通过软件将 CEN 位置 1，才可以使用门控模式。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。

25.4.2 TIM6/TIM7 控制寄存器 2 (TIMx_CR2)

TIM6/TIM7 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**：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 (TRGO)。如果复位由触发输入产生（从模式控制器配置为复位模式），则 TRGO 上的信号相比实际复位会有延迟。

001：**使能**——计数器使能信号 CNT_EN 用作触发输出 (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或运算组合而成。

当计数器使能信号由触发输入控制时，TRGO 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中对 MSM 位的说明）。

010：**更新**——选择更新事件作为触发输出 (TRGO)。例如，主定时器可用作从定时器的预分频器。

注： 从定时器时钟必须在主定时器接收事件之前使能，并且在接收主定时器触发信号时不得实时更改。

位 3:0 保留，必须保持复位值。

25.4.3 TIM6/TIM7 DMA/中断使能寄存器 (TIMx_DIER)

TIM6/TIM7 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。

25.4.4 TIM6/TIM7 状态寄存器 (TIMx_SR)

TIM6/TIM7 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：

– 重复计数器值上溢或下溢并且 TIMx_CR1 寄存器中的 UDIS=0 时。

– 当由于 TIMx_CR1 寄存器中 URS = 0 且 UDIS = 0 而通过软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。

25.4.5 TIM6/TIM7 事件产生寄存器 (TIMx_EGR)

TIM6/TIM7 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）。

25.4.6 TIM6/TIM7 计数器 (TIMx_CNT)

TIM6/TIM7 counter

偏移地址：0x24

复位值：0x0000

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)

25.4.7 TIM6/TIM7 预分频器 (TIMx_PSC)

TIM6/TIM7 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 CK_CNT 等于 f_{CK_PSC} / (PSC[15:0] + 1)。
PSC 包含在每次发生更新事件时要装载到实际预分频器寄存器的值。

25.4.8 TIM6/TIM7 自动重载寄存器 (TIMx_ARR)

TIM6/TIM7 auto-reload register

偏移地址：0x2C

复位值：0x0000

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 15:0 **ARR[15:0]**：预分频器值 (Prescaler value)
ARR 为要装载到实际自动重载寄存器的值。
有关 ARR 更新和行为的详细信息，请参见*第 762 页上的第 25.3.1 节：时基单元*。
当自动重载值为空时，计数器不工作。

25.4.9 TIM6/TIM7 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 138. TIM6/TIM7 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	**TIMx_CR1**	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	UIFREMAP	Res	Res	Res	ARPE	Res	Res	Res	OPM	URS	UDIS	CEN
	Reset value																					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	Res	Res	Res	Res	ARR[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 2.2.2 节：存储器映射和寄存器边界地址*。

26 低功耗定时器 (LPTIM)

26.1 前言

LPTIM 是一个 16 位定时器，可从降低功耗的最终发展中受益。由于时钟源的多样性，LPTIM 能够在除待机模式以外的所有电源模式下保持运行。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。

26.2 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：LSE、LSI、HSI 或 APB 时钟
  - ULPTIM 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
- 可编程数字干扰滤波器
- 可配置输出：脉冲和 PWM
- 可配置 I/O 极性
- 编码器模式

26.3 LPTIM 实现

表 139 介绍了 STM32F75xxx 和 STM32F74xxx 器件上的 LPTIM 实现：LPTIM1 支持所有特性。LPTIM2 支持的特性较少，但在其余方面与 LPTIM1 完全相同。

表 139. STM32F75xxx 和 STM32F74xxx LPTIM 特性

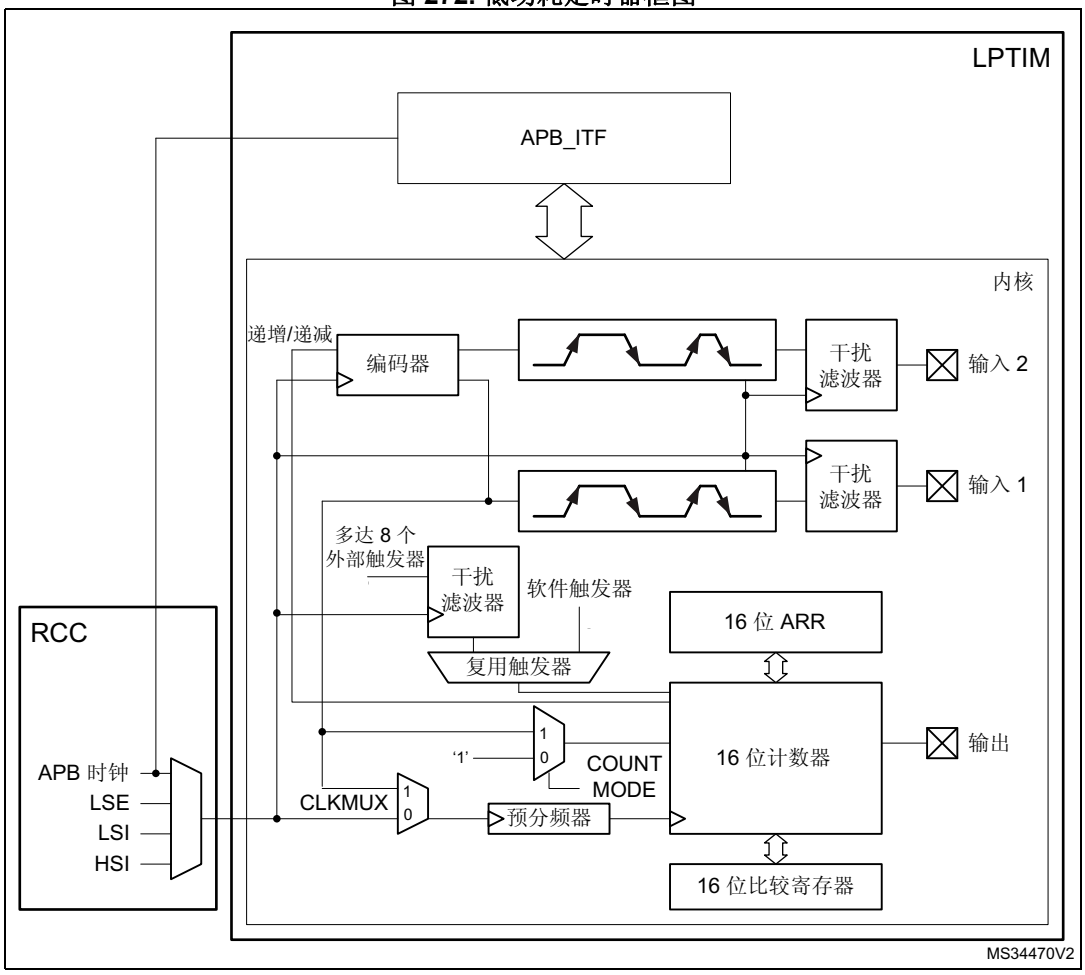
LPTIM 模式/特性(1)	LPTIM1	LPTIM2
编码器模式	X	-

1. X = 支持。

26.4 LPTIM 功能说明

26.4.1 LPTIM 框图

图 272. 低功耗定时器框图



26.4.2 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时钟树控制器 (RCC) 在 APB、LSI、LSE 或 HSI 时钟源中进行选择。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 APB 或 LSE、LSI 和 HSI 等任何其他内置振荡器为 LPTIM 提供内部时钟信号。
- 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，则还应提供内部时钟信号（第一种配置）。在这种情况下，内部时钟信号频率应至少为外部时钟信号频率的五倍。

26.4.3 干扰滤波器

LPTIM 输入（外部或内部）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，首先应向 LPTIM 提供内部时钟源，这是保证滤波器正常工作的必要条件。

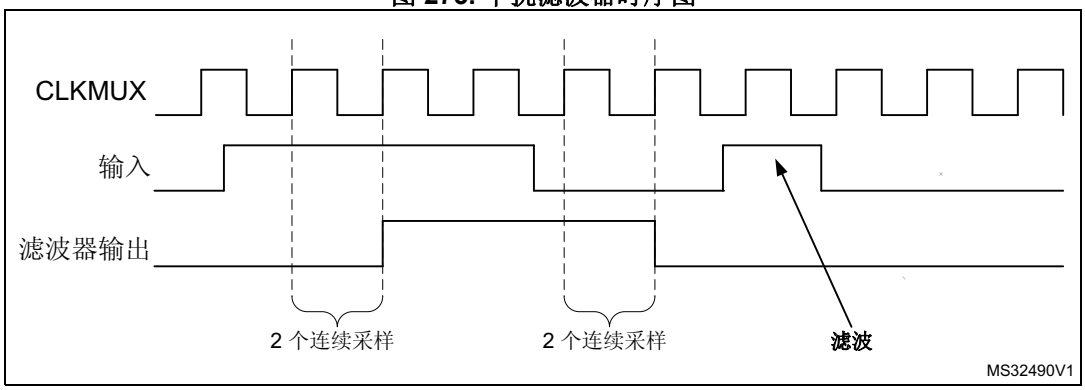
数字滤波器分为两组：

- 第一组数字滤波器保护 LPTIM 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 内部触发输入。数字滤波器的敏感性由 TRGFLT 位控制。

注： *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切换。*图 273* 给出了编程 2 个连续采样时，干扰滤波器行为的示例。

图 273. 干扰滤波器时序图



注： *不提供内部时钟信号时，必须通过将 CKFLT 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。*

26.4.4 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 140. 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

26.4.5 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： *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。*

26.4.6 工作模式

LPTIM 支持以下两种工作模式：

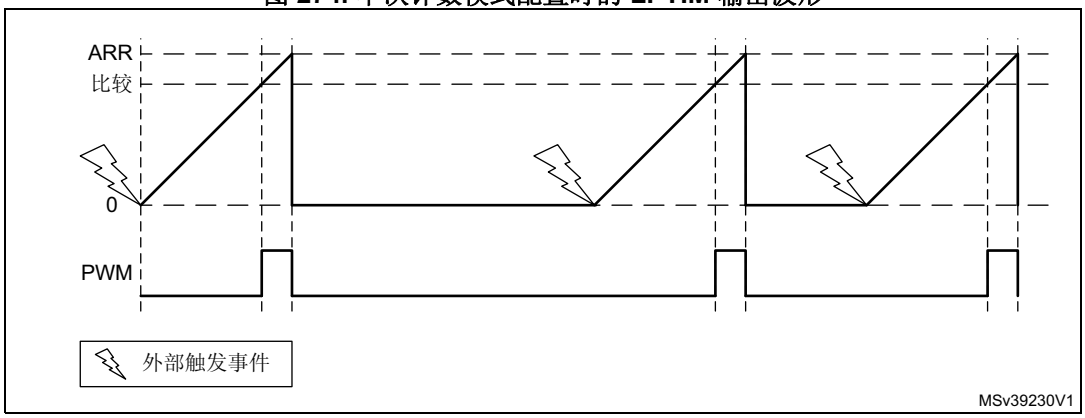
- 连续模式：定时器自由运行，由触发事件启动并且直到被禁止才会停止。
- 单触发模式：定时器由触发事件启动，当达到 ARR 值时停止。

新的触发事件将重新启动定时器。从计数器启动到计数器达到 ARR，这段时间内发生的任何触发事件均将被丢弃。

要使能单触发计数，必须将 SNGSTRT 位置 1。

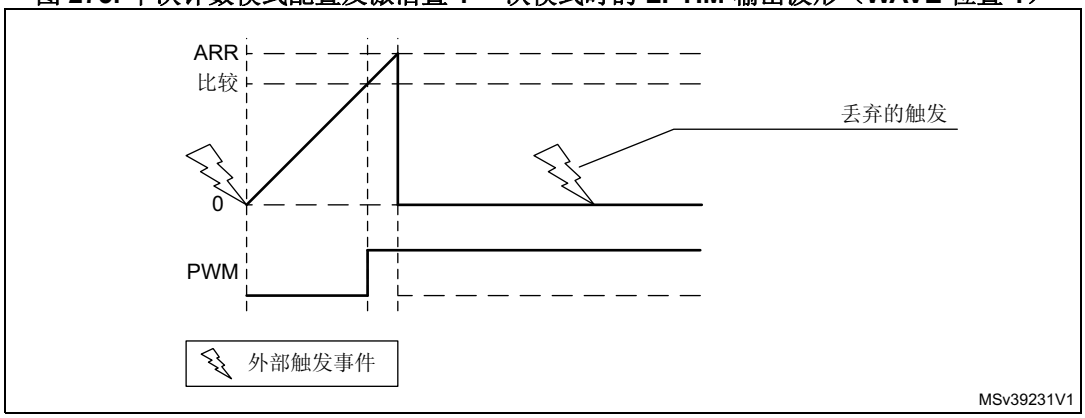
当选择外部触发时，SNGSTRT 位置 1 后及计数器寄存器停止后（包含 0 值）到达的每个外部触发事件将使计数器开始新的单次计数周期，如*图 274* 所示。

图 274. 单次计数模式配置时的 LPTIM 输出波形



应注意，当 LPTIMx_CFGR 寄存器中的 WAVE 位字段置 1 时，将激活置 1 一次模式。在这种情况下，计数器仅在第一个触发后启动一次，任何后续的触发事件均被丢弃，如*图 274* 所示。

图 275. 单次计数模式配置及激活置 1 一次模式时的 LPTIM 输出波形（WAVE 位置 1）



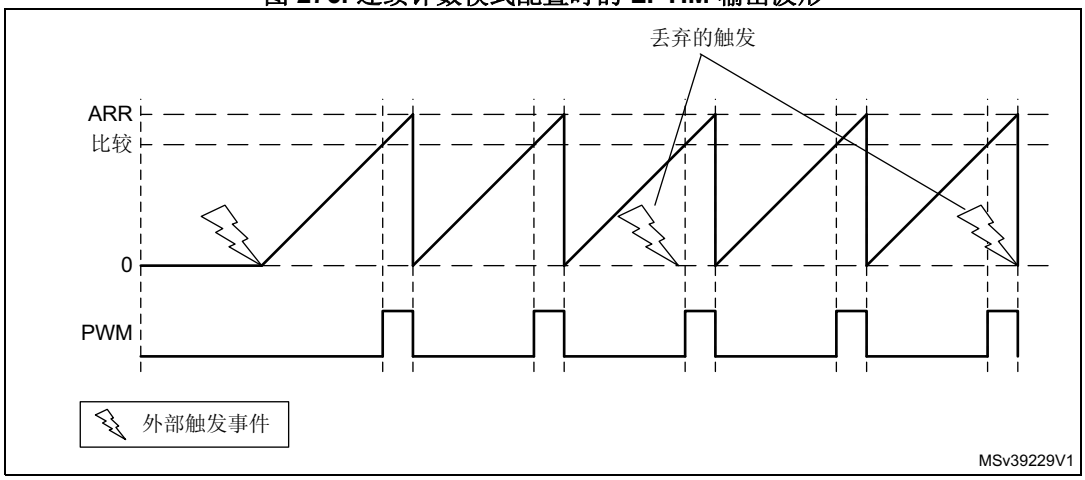
若通过软件启动（TRIGEN[1:0] = “00”），将 SNGSTRT 置 1 会使计数器进行单触发计数。

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均将被丢弃，如*图 276* 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 276. 连续计数模式配置时的 LPTIM 输出波形



SNGSTRT 和 CNTSTRT 位只能在定时器使能时（ENABLE 位置 1）置 1。可以将“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）将在达到 ARR 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）将在达到 ARR 后立即重新启动。

26.4.7 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。

第一个触发事件将启动定时器，任何后续的触发事件将复位计数器，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。

26.4.8 生成波形

两个 16 位寄存器，LPTIMx_ARR（自动重载寄存器）和 LPTIMx_CMP（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：若 LPTIMx_CMP 寄存器与 LPTIMx_CNT 寄存器匹配，则会立即将 LPTIM 输出置 1。若 LPTIMx_ARR 寄存器与 LPTIMx_CNT 寄存器匹配，则会立即将 LPTIM 输出复位。
- 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 LPTIMx_ARR 寄存器的值严格大于 LPTIMx_CMP 寄存器的值。

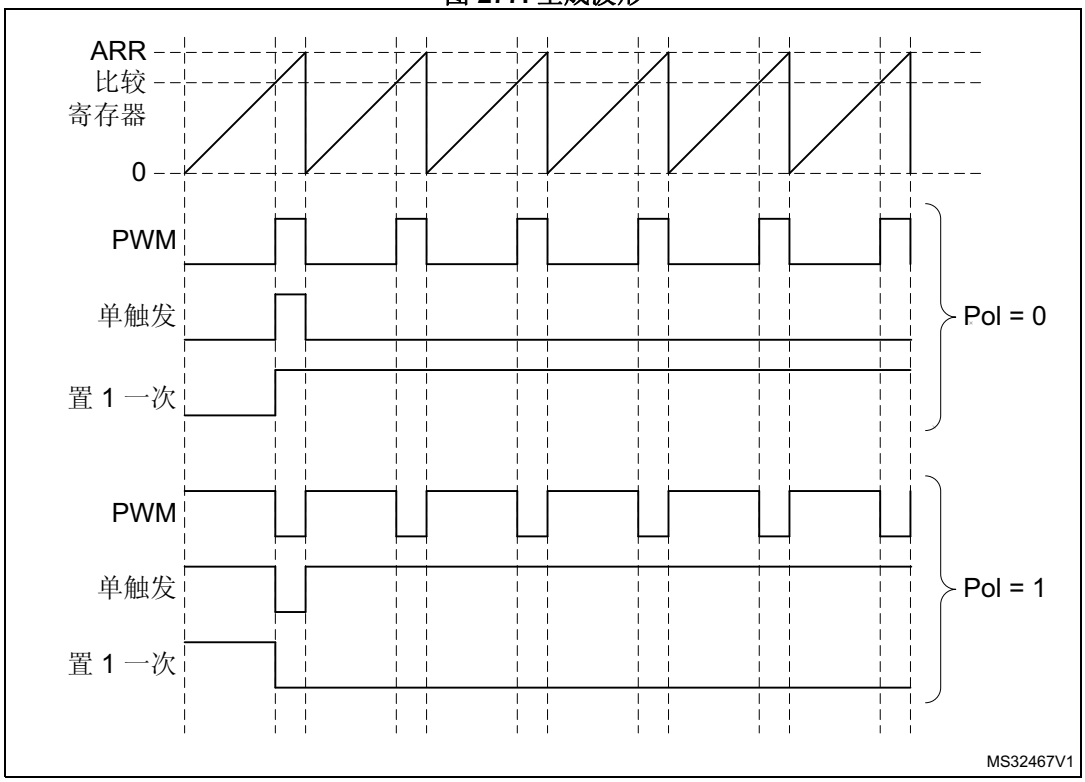
LPTIM 输出波形可通过 WAVE 位配置，具体如下：

- 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 一次波形。

WAVPOL 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 277* 给出了可能在 LPTIM 输出上生成的三种波形。此外，此图还显示了通过 WAVPOL 位更改极性所产生的效果。

图 277. 生成波形



26.4.9 寄存器更新

LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器在 APB 总线写操作后会立即更新，若定时器已启动，也可在当前周期结束时更新。

PRELOAD 位控制 LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器的更新方式：

- 当 PRELOAD 位复位为 0 时，LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器会在写访问后立即更新。
- 当 PRELOAD 位置 1 时，若定时器已启动，LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器会在当前周期结束时更新。

APB 总线和 LPTIM 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。

LPTIMx_ISR 寄存器中的 ARROK 标志和 CMPOK 标志分别指示 LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器的写操作已完成。

向 LPTIMx_ARR 寄存器和 LPTIMx_CMP 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 标志或 CMPOK 标志置 1 前执行连续的写操作将造成无法预知的结果。

26.4.10 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 位用于控制将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

    当 LPTIM 由内部时钟源提供时钟，且 LPTIM 计数器根据在 LPTIM 外部 Input1 上检测到的有效边沿进行更新时，不得对提供给 LPTIM 的内部时钟进行预分频（PRESC[2:0] = “000”）。
  - COUNTMODE = 1

    LPTIM 外部 Input1 通过提供给 LPTIM 的内部时钟采样。因此，为了不丢失任何事件，外部 Input1 信号变化的频率决不应超过提供给 LPTIM 的内部时钟的频率。
- 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。此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 LPTIM 的时钟，计数器递增计数前存在一些初始延时（使能 PTIM 后）。更确切地说，LPTIM 外部 Input1 的前五个有效边沿将丢失（使能 PTIM 后）。

26.4.11 定时器使能

LPTIMx_CR 寄存器的 ENABLE 位用于使能/禁止 LPTIM。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。

LPTIMx_CFGR 和 LPTIMx_IER 寄存器必须在禁止 LPTIM 后才能修改。

26.4.12 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 LPTIMx_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，在启动前必须先配置 LPTIMx_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 LPTIMx_ISR 寄存器中的两个递减和递增标志指示。此外，如果通过 LPTIMx_IER 寄存器使能，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，对应于所连传感器的旋转方向。

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）。

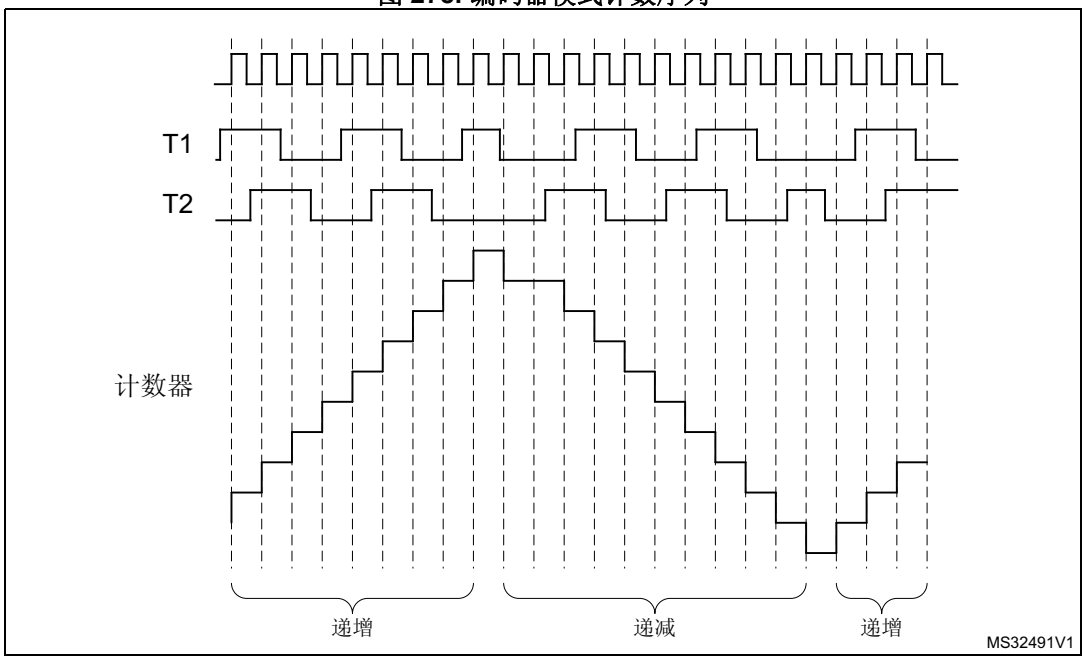
表 141. 编码器计数方案

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 278. 编码器模式计数序列



26.5 LPTIM 中断

若以下事件通过 LPTIMx_IER 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）。

注： *只要 LPTIMx_IER 寄存器（中断使能寄存器）中的位在 LPTIMx_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*

26.6 LPTIM 寄存器

26.6.1 LPTIM 中断和状态寄存器 (LPTIMx_ISR)

LPTIM 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.	Res.	Res.	Res.	Res.	DOWN	UP	ARROK	CMPOK	EXTTRIG	ARRM	CMPM
									r	r	r	r	r	r	r

位 31:7 保留，必须保持复位值。

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。

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。

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 LPTIMx_ARR 寄存器的 APB 总线写操作已成功完成。如果这样，便可启动新的写操作。

位 3 **CMPOK**：比较寄存器更新成功 (Compare register update OK)
由硬件将 CMPOK 置 1 时，会通知应用 LPTIMx_CMP 寄存器的 APB 总线写操作已成功完成。如果这样，便可启动新的写操作。

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。

位 1 **ARRM**：自动重载匹配 (Autoreload match)
由硬件将 ARRM 置 1 时，会通知应用 LPTIMx_CNT 寄存器的值已达到 LPTIMx_ARR 寄存器的值。

位 0 **CMPM**：比较匹配 (Compare match)
由硬件将 CMPM 位置 1 时，会通知应用 LPTIMx_CNT 寄存器的值已达到 LPTIMx_CMP 寄存器的值。

26.6.2 LPTIM 中断清零寄存器 (LPTIMx_ICR)

LPTIM interrupt clear 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.	DOWNCF	UPCF	ARROKCF	CMPOKCF	EXTTRIGCF	ARRMCF	CMPMCF
									w	w	w	w	w	w	w

位 31:7 保留，必须保持复位值。

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，LPT_ISR 寄存器中的 DOWN 标志将清零

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，LPT_ISR 寄存器中的 UP 标志将清零

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，LPT_ISR 寄存器中的 ARROK 标志将清零

位 3 **CMPOKCF**：比较寄存器更新成功清零标志 (Compare register update OK Clear Flag)
将 1 写入此位时，LPT_ISR 寄存器中的 CMPOK 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，LPT_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，LPT_ISR 寄存器中的 ARRM 标志将清零

位 0 **CMPMCF**：比较匹配清零标志 (compare match Clear Flag)
将 1 写入此位时，LPT_ISR 寄存器中的 CMP 标志将清零

26.6.3 LPTIM 中断使能寄存器 (LPTIMx_IER)

LPTIM interrupt enable 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.	Res.	DOWNIE	UPIE	ARROKIE	CMPOKIE	EXTTRIGIE	ARRMIE	CMPMIE
									rw	rw	rw	rw	rw	rw	rw

位 31:7 保留，必须保持复位值。

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

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

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 **CMPOKIE**：比较寄存器更新成功中断使能 (Compare register update OK Interrupt Enable)
0：禁止 CMPOK 中断
1：使能 CMPOK 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 **CMPMIE**：比较匹配中断使能 (Compare match Interrupt Enable)
0：禁止 CMPM 中断
1：使能 CMPM 中断

小心： 必须在 LPTIM 禁止已时（ENABLE 位复位为 0）才能修改 LPTIMx_IER 寄存器。

26.6.4 LPTIM 配置寄存器 (LPTIMx_CFGR)

LPTIM configuration 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.	ENC	COUNT MODE	PRELOAD	WAVPOL	WAVE	TIMOUT	TRIGEN		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
TRIGSEL			Res.	PRESC			Res.	TRGFLT		Res.	CKFLT		CKPOL		CKSEL
rw	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 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

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 LPTIMx_ARR 寄存器和 LPTIMx_CMP 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 **WAVPOL**：波形极性 (Waveform shape polarity)
WAVEPOL 位控制输出极性
0：LPTIM 输出反映 LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器之间的比较结果。
1：LPTIM 输出反映与 LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器之间的比较结果相反的值

位 20 **WAVE**：波形 (Waveform shape)
WAVE 位控制输出波形
0：禁止置 1 一次模式，PWM/ 单脉冲波形（取决于 OPMODE 位）
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计数器

位 18:17 **TRIGEN**：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**：触发源选择器 (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: ext_trig0
001: ext_trig1
010: ext_trig2
011: ext_trig3
100: ext_trig4
101: ext_trig5
110: ext_trig6
111: ext_trig7

有关各定时器的 ITRx 含义的更多详细信息，请参见*表 142：LPTIM 外部触发连接*。

位 12 保留，必须保持复位值。

位[11:9] **PRESC**：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**：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**：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**：时钟极性 (Clock Polarity)

如果 LPTIM 由外部时钟源提供时钟：

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

01：下降沿为用于计数的有效边沿

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。

11：不允许

如果将 LPTIM 配置为编码器模式（ENC 位置 1）：

00：编码器子模式 1 激活

01：编码器子模式 2 激活

10：编码器子模式 3 激活

有关编码器模式子模式的更多详细信息，请参见*第 26.4.12 节：编码器模式*。

位 0 **CKSEL**：时钟选择器 (Clock selector)

CKSEL 位选择 LPTIM 将使用的时钟源：

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 LPTIMx_CFGR 寄存器

表 142. LPTIM 外部触发连接

TRIGSEL	外部触发信号
ext_trig0	GPIO
ext_trig1	RTC_ALARMA
ext_trig2	RTC_ALARMB
ext_trig3	RTC_TAMP1_OUT
ext_trig4	RTC_TAMP2_OUT
ext_trig5	RTC_TAMP3_OUT
ext_trig6	保留
ext_trig7	保留

26.6.5 LPTIM 控制寄存器 (LPTIMx_CR)

LPTIM control 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT STRT	SNG STRT	ENA BLE
													rw	rw	rw

位 31:3 保留，必须保持复位值。

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 LPTIMx_ARR 寄存器和 LPTIMx_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 LPTIMx_ARR 寄存器和 LPTIMx_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

1：使能 LPTIM

26.6.6 LPTIM 比较寄存器 (LPTIMx_CMP)

LPTIM compare 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
CMP[15:0]															
rw															

位 31:16 保留，必须保持复位值。

位 15:0 **CMP**：比较值 (Compare value)。
CMP 为 LPTIM 所使用的比较值。
必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIMx_CMP 寄存器的内容。

26.6.7 LPTIM 自动重载寄存器 (LPTIMx_ARR)

LPTIM autoreload register

偏移地址：0x18

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															

位 31:16 保留，必须保持复位值。

位 15:0 **ARR**：自动重载值 (Auto reload value)。
ARR 为 LPTIM 的自动重载值。
此值必须严格大于 CMP[15:0] 的值。
必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIMx_ARR 寄存器的内容。

26.6.8 LPTIM 计数器寄存器 (LPTIMx_CNT)

LPTIM counter register

偏移地址：0x1C

复位值：0x0000 0000

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

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

位 31:16 保留，必须保持复位值。

位 15:0 **CNT**：计数器值 (Counter value)。

当 LPTIM 通过异步时钟运行时，读取 LPTIMx_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。

应注意，要实现可靠的 LPTIM_CNT 寄存器读访问，必须执行两个连续的读访问并加以比较。当两个连续的读访问的值相等时，可认为读访问可靠。

26.6.9 LPTIM1 选项寄存器 (LPTIM1_OR)

LPTIM1 option 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.	OR_1	OR_0
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **OR_1**：选项寄存器位 1 (Option register bit 1)

0：LPTIM1 输入 2 连接到 I/O

1：LPTIM1 输入 2 连接到 COMP2_OUT

位 0 **OR_0**：选项寄存器位 0 (Option register bit 0)

0：LPTIM1 输入 1 连接到 I/O

1：LPTIM1 输入 1 连接到 COMP1_OUT

26.6.10 LPTIM2 选项寄存器 (LPTIM2_OR)

LPTIM2 option 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.	OR_1	OR_0
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **OR_1**：选项寄存器位 1 (Option register bit 1)
0：LPTIM2 输入 1 连接到 I/O
1：LPTIM2 输入 1 连接到 COMP2_OUT

位 0 **OR_0**：选项寄存器位 0 (Option register bit 0)
0：LPTIM2 输入 1 连接到 I/O
1：LPTIM2 输入 1 连接到 COMP1_OUT

注： *当 OR_1 和 OR_0 均置 1 时，LPTIM2 输入 1 连接到（COMP1_OUT 或 COMP2_OUT）。*

26.6.11 LPTIM 寄存器映射

下表对 LPTIM 寄存器进行了汇总。

表 143. LPTIM 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	LPTIMx_ISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWN	UP	ARROK	CMPOK	EXTTRIG	ARRM	CMPM
	Reset value																										0	0	0	0	0	0	0
0x04	LPTIMx_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.	DOWNCF	UPCF	ARROKCF	CMPOKCF	EXTTRIGCF	ARRMCF	CMPMCF
	Reset value																										0	0	0	0	0	0	0
0x08	LPTIMx_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.	DOWNIE	UPIE	ARROKIE	CMPOKIE	EXTTRIGIE	ARRMIE	CMPMIE
	Reset value																										0	0	0	0	0	0	0
0x0C	LPTIMx_CFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENC	COUNTMODE	PRELOAD	WAVPOL	WAVE	TIMOUT	TRIGEN		Res.	TRIGSEL			Res.	PRESC			Res.	TRGFLT		Res.	CKFLT		CKPOL		CKSEL
	Reset value								0	0	0	0	0	0	0	0		0	0	0		0	0	0		0	0		0	0	0	0	0
0x10	LPTIMx_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.	Res.	Res.	CNTSTRT	SNGSTRT	ENABLE
	Reset value																														0	0	0
0x14	LPTIMx_CMP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMP[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	LPTIMx_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
0x1C	LPTIMx_CNT	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
0x20	LPTIMx_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.	OR_1	OR_0
	Reset Value																															0	0

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

27 独立看门狗 (IWDG)

27.1 前言

此器件具有一个嵌入式看门狗外设，具有安全性高、定时准确及使用灵活的优点。此独立看门狗外设可用于检测并解决由软件错误导致的故障，并在计数器达到给定的超时值时触发系统复位。

独立看门狗 (IWDG) 由其专用低速时钟 (LSI) 驱动，因此即便在主时钟发生故障时仍然保持工作状态。

IWDG 最适合应用于那些需要看门狗作为一个在主程序之外，能够完全独立工作，并且对时间精度要求较低的场合。有关窗口看门狗的详细信息，请参见*第 801 页上的第 28 节*。

27.2 IWDG 主要特性

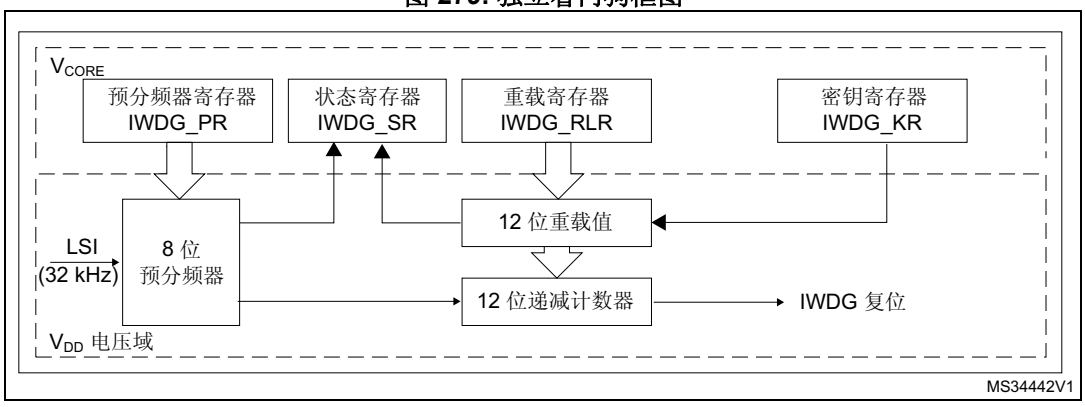
- 自由运行递减计数器
- 时钟由独立 RC 振荡器提供（可在待机和停止模式下运行）
- 复位条件
  - 当递减计数器值小于 0x000 时复位（如果看门狗已激活）
  - 在窗口之外重载递减计数器时复位（如果看门狗已激活）

27.3 IWDG 功能说明

27.3.1 IWDG 框图

图 279 给出了独立看门狗模块的功能框图。

图 279. 独立看门狗框图



注： *看门狗功能由 V_{CORE} 电压域供电，在停止模式和待机模式下仍能工作。*

通过向键寄存器 (IWDG_KR) 中写入值 0x0000 CCCC 来启动独立看门狗时，计数器开始从复位值 0xFFF 递减计数。当计数器计数到终值 (0x000) 时会产生一个复位信号（IWDG 复位）。

任何时候将键值 0x0000 AAAA 写到 IWDG_KR 寄存器中，IWDG_RLR 的值就会被重载到计数器，从而避免产生看门狗复位。

27.3.2 窗口选项

通过在 IWDG_WINR 寄存器中设置合适的窗口，IWDG 也可以用作窗口看门狗。

当计数器值大于窗口寄存器 (IWDG_WINR) 中存储的值时，如果执行重载操作，则会产生复位。

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。

窗口值一经更改，便执行重载操作，以便将递减计数器复位为 IWDG_RLR 值，并方便计算周期数以生成下一次重载。

使能窗口选项时配置 IWDG

1. 通过在 IWDG_KR 寄存器中写入 0x0000 CCCC 来使能 IWDG。
2. 通过在 IWDG_KR 寄存器中写入 0x0000 5555 来使能寄存器访问。
3. 通过将 IWDG_PR 编程为 0~7 中的数值来配置 IWDG 预分频器。
4. 写重载寄存器 (IWDG_RLR)。
5. 等待寄存器被更新 (IWDG_SR = 0x0000 0000)。
6. 写窗口寄存器 IWDG_WINR，将自动刷新计数器值为 IWDG_RLR 的值。

注： *当 IWDG_SR 设置为 0x0000 0000 时，写入窗口值允许刷新计数器值为 RLR 的值。*

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_KR 寄存器中写入 0x0000 CCCC 来使能 IWDG。
2. 通过在 IWDG_KR 寄存器中写入 0x0000 5555 来使能寄存器访问。
3. 通过将 IWDG_PR 编程为 0~7 中的数值来配置 IWDG 预分频器。
4. 写重载寄存器 (IWDG_RLR)。
5. 等待寄存器更新 (IWDG_SR = 0x0000 0000)。
6. 刷新计数器值为 IWDG_RLR 的值 (IWDG_KR = 0x0000 AAAA)。

27.3.3 硬件看门狗

如果通过器件选项位使能“硬件看门狗”功能，上电时将自动使能看门狗；如果在计数器计数结束前，若软件没有向键寄存器写入相应的值，或者在窗口内部重载了递减计数器，则系统会产生复位。

27.3.4 低功耗冻结

根据 IWDG_STOP 和 IWDG_STBY 选项配置，IWDG 可分别在停止模式和待机模式期间继续计数或停止计数。如果停止模式或待机模式期间 IWDG 保持运行，它可从此模式唤醒器件。更多详细信息，请参见*用户和读保护选项字节*一节。

27.3.5 寄存器访问保护

IWDG_PR、IWDG_RLR 和 IWDG_WINR 寄存器具有写访问保护。若要修改寄存器，必须首先对 IWDG_KR 寄存器写入代码 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。这意味着重装载操作（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。

27.3.6 调试模式

当微控制器进入调试模式时（内核停止），IWDG 计数器会根据 DBG 模块中的 DBG_IWDG_STOP 配置位选择继续正常工作或者停止工作。

27.4 IWDG 寄存器

有关寄存器说明中使用的缩写，请参见*第 58 页上的第 1.1 节*。

外设寄存器可支持半字（16 位）或字（32 位）访问。

27.4.1 键寄存器 (IWDG_KR)

Key register

偏移地址：0x00

复位值：0x0000 0000（待机模式时复位）

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
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）

必须每隔一段时间便通过软件对这些位写入键值 0xAAAA，否则当计数器计数到 0 时，看门狗会产生复位。

写入键值 0x5555 可使能对 IWDG_PR、IWDG_RLR 和 IWDG_WINR 寄存器的访问（请参见*第 27.3.5 节：寄存器访问保护*）

写入键值 CCCCh 可启动看门狗（选中硬件看门狗选项的情况除外）

27.4.2 预分频器寄存器 (IWDG_PR)

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.	Res.	PR[2:0]		
													rw	rw	rw

位 31:3 保留，必须保持复位值。

位 2:0 **PR[2:0]**：预分频系数 (Prescaler divider)

这些位受写访问保护，请参见*第 27.3.5 节：寄存器访问保护*。通过软件设置这些位来选择计数器时钟的预分频因子。若要更改预分频器的分频系数，IWDG_SR 的 PVU 位必须为 0。

000：4 分频
001：8 分频
010：16 分频
011：32 分频
100：64 分频
101：128 分频
110：256 分频
111：256 分频

注： *读取该寄存器会返回 VDD 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_SR 寄存器中的 PVU 位为 0 时，从寄存器读取的值才有效。*

27.4.3 重载寄存器 (IWDG_RLR)

Reload register

偏移地址：0x08

复位值：0x0000 0FFF（待机模式时复位）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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 27.3.5 节*。这个值由软件设置，每次对 IWDG_KR 寄存器写入值 0xAAAA 时，这个值就会重装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。超时周期由该值和时钟预分频器共同决定。有关超时信息，请参见数据手册。

若要更改重载值，IWDG_SR 中的 RVU 位必须为 0。

注： *读取该寄存器会返回 VDD 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_SR 寄存器中的 RVU 位为 0 时，从寄存器读取的值才有效。*

27.4.4 状态寄存器 (IWDG_SR)

Status register

偏移地址：0x0C

复位值：0x0000 0000（待机模式时不复位）

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

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.	WVU	RVU	PVU
													r	r	r

位 31:3 保留，必须保持复位值。

位 2 **WVU**：看门狗计数器窗口值更新 (Watchdog counter window value update)

可通过硬件将该位置 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（需要多达 5 个 RC 40 kHz 周期），会通过硬件将该位复位。

窗口值只有在 WVU 位为 0 时才可更新。

此位只有在通用“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)
可通过硬件将该位置 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（需要多达 5 个 RC 40 kHz 周期），会通过硬件将该位复位。
重载值只有在 RVU 位为 0 时才可更新。

位 0 **PVU**：看门狗预分频器值更新 (Watchdog prescaler value update)
可通过硬件将该位置 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值更新操作后（需要多达 5 个 RC 40 kHz 周期），会通过硬件将该位复位。
预分频器值只有在 PVU 位为 0 时才可更新。

注： *如果应用使用多个重载值、预分频器值或窗口值，则必须等到 RVU 位被复位后才能更改重载值，等到 PVU 位被复位后才能更改预分频器值，而且必须等到 WVU 位被复位后才能更改窗口值。但是，在更新预分频器和/或重载/窗口值之后，则无需等到 RVU、PVU 或 WVU 复位后再继续执行代码（进入低功耗模式时除外）。*

27.4.5 窗口寄存器 (IWDG_WINR)

Window register

偏移地址：0x10

复位值：0x0000 0FFF（待机模式时复位）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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 27.3.5 节*。这些位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，当递减计数器的值低于窗口寄存器值
且大于 0x0 时必须重载
若要更改重载值，IWDG_SR 中的 WVU 位必须为 0。

注： *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_SR 寄存器中的 WVU 位为 0 时，从寄存器读取的值才有效。*

27.4.6 IWDG 寄存器映射

下表提供了 IWDG 寄存器映射和复位值。

表 144. IWDG 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	**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	Res	PR[2:0]		
	Reset value																														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	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	WVU	RVU	PVU
	Reset value																														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

有关寄存器边界地址的信息，请参见*第 2.2.2 节：存储器映射和寄存器边界地址*。

28 系统窗口看门狗 (WWDG)

28.1 前言

系统窗口看门狗 (WWDG) 通常被用来监测，由外部干扰或不可预见的逻辑条件造成的应用程序背离正常的运行序列而产生的软件故障。除非递减计数器的值在 T6 位变成 0 前被刷新，看门狗电路在达到预置的时间周期时，会产生一个 MCU 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，也会产生 MCU 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程序。

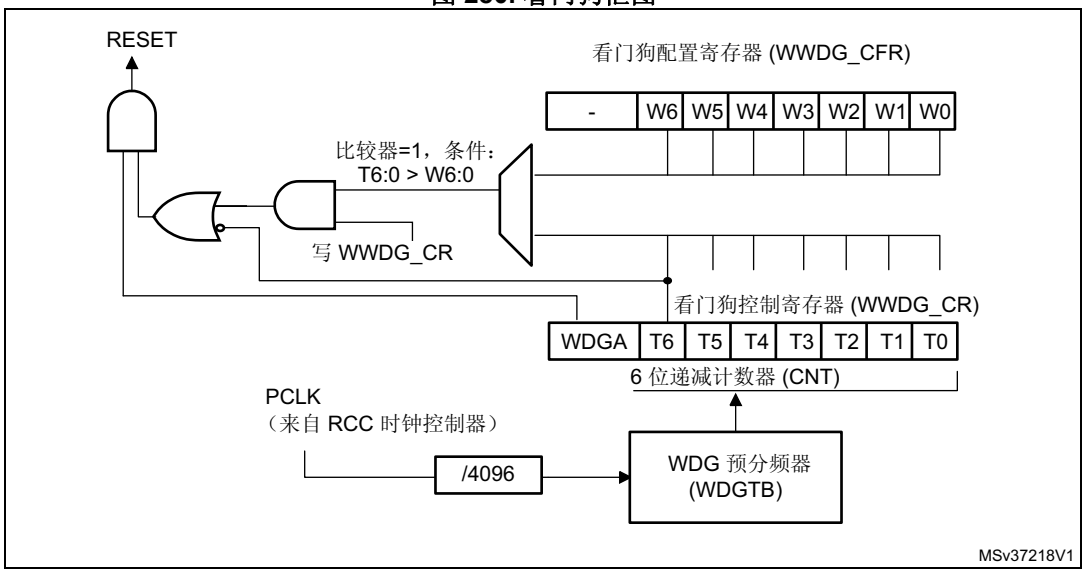
28.2 WWDG 主要特性

- 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 281*）
- 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）

28.3 WWDG 功能说明

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 滚动到 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。

图 280. 看门狗框图



应用程序在正常运行过程中必须定期地写入 WWDG_CR 寄存器以防止 MCU 发生复位。只有当计数器值低于窗口寄存器值时，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。

28.3.1 使能看门狗

在系统复位后，看门狗总是处于关闭状态。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。

28.3.2 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 281*）。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窗口寄存器值且大于 0x3F 时必须重载。*图 281* 介绍了窗口看门狗的工作过程。

注： *可使用 T6 位产生软件复位（将 WDGA 位置 1 并将 T6 位清零）。*

28.3.3 看门狗中断高级特性

如果在产生实际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)。通过设置 WWDG_CFR 寄存器中的 EWI 位使能 EWI 中断。当递减计数器的值为 0x40 时，将生成 EWI 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，相应的中断服务程序 (ISR) 可用来重载 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 EWI 中断。

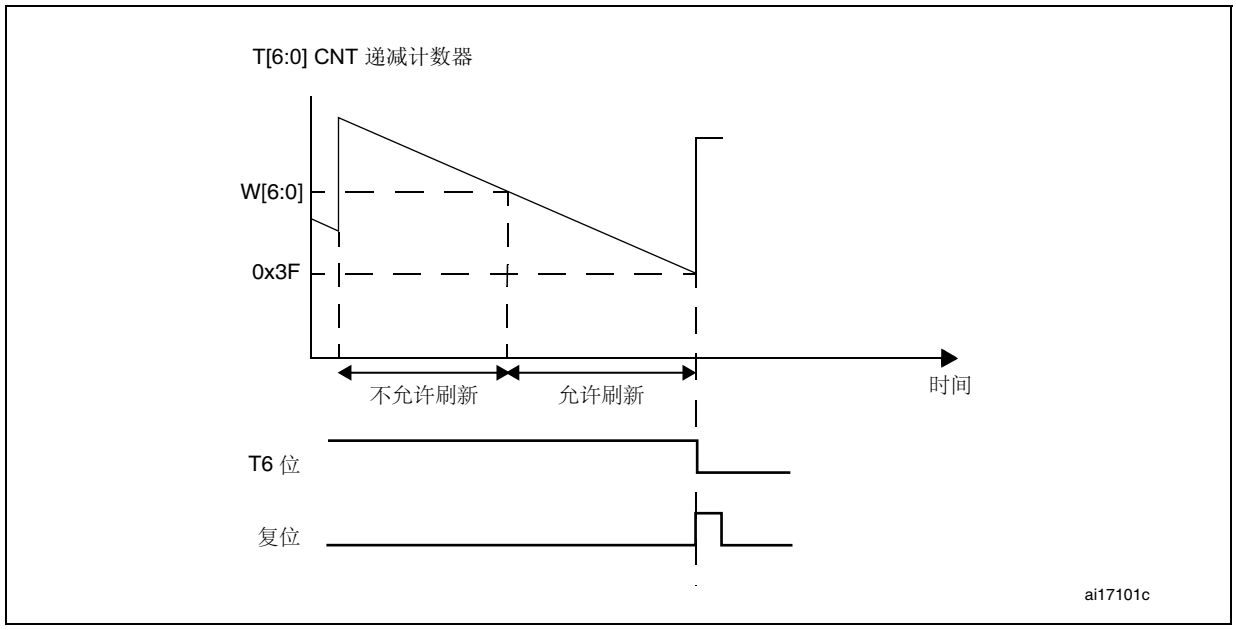
注： *当由于在更高优先级任务中有系统锁定而无法使用 EWI 中断时，最终会产生 WWDG 复位。*

28.3.4 如何设置看门狗超时

可以使用*图 281* 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 281. 窗口看门狗时序图



超时值的计算公式如下：

$$t_{WWDG} = t_{PCLK} \times 4096 \times 2^{WDGTB[1:0]} \times (T[5:0] + 1) \quad (ms)$$

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，假设 APB 频率等于 48 MHz，将 WDGTB[1:0] 设置为 3 并将 T[5:0] 设置为 63：

$$t_{WWDG} = 1/48000 \times 4096 \times 2^{3} \times (63 + 1) = 43.69 \text{ ms}$$

有关 t_{WWDG} 的最小值和最大值，请参见数据手册。

28.3.5 调试模式

当微控制器进入调试模式时（Cortex®-M7 内核停止），WWDG 计数器会根据 DBG 模块中的 DBG_WWDG_STOP 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*第 40.16.2 节：对定时器、看门狗、bxCAN 和 I^2C 的调试支持*。

28.4 WWDG 寄存器

有关寄存器说明中使用的缩写，请参见*第 58 页上的第 1.1 节*。

外设寄存器可支持半字（16 位）或字（32 位）访问。

28.4.1 控制寄存器 (WWDG_CR)

Control register

偏移地址：0x00

复位值：0x0000 007F

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

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

位 31:8 保留，必须保持复位值。

位 7 **WDGA**：激活位 (Activation bit)
此位由软件置 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。它每隔 (4096 x $2^{WDGTB[1:0]}$) PCLK 个周期递减一次。当它从 0x40 滚动到 0x3F（T6 清零）时会产生复位。

28.4.2 配置寄存器 (WWDG_CFR)

Configuration register

偏移地址：0x04

复位值：0x0000 007F

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

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

位 31:10 保留，必须保持复位值。

位 9 **EWI**：提前唤醒中断 (Early wakeup interrupt)
置 1 后，只要计数器值达到 0x40 就会产生中断。此中断只有在复位后才由硬件清零。

位 8:7 **WDGTB[1:0]**：定时器时基 (Timer base)
可按如下方式修改预分频器的时基：
00：CK 计数器时钟 (PCLK div 4096) 分频器 1
01：CK 计数器时钟 (PCLK div 4096) 分频器 2
10：CK 计数器时钟 (PCLK div 4096) 分频器 4
11：CK 计数器时钟 (PCLK div 4096) 分频器 8

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。

28.4.3 状态寄存器 (WWDG_SR)

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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 wakeup interrupt flag)
当计数器值达到 0x40 时此位由硬件置 1。它必须由软件通过写入 0 来清零。写入 1 不起作用。
如果不使能中断，此位也会被置 1。

28.4.4 WWDG 寄存器映射

下表提供了 WWDG 寄存器映射和复位值。

表 145. WWDG 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	**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
0x04	**WWDG_CFR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWI	WDGTB1	WDGTB0	W[6:0]						
	Reset value																							0	0	0	1	1	1	1	1	1	1
0x08	**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 2.2.2 节：存储器映射和寄存器边界地址*。

29 实时时钟 (RTC)

29.1 前言

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。

RTC 还包含具有中断功能的周期性可编程唤醒标志。

两个 32 位寄存器包含二进码十进数格式 (BCD) 的秒、分钟、小时（12 或 24 小时制）、星期几、日期、月份和年份。此外，还可提供二进制格式的亚秒值。

系统可以自动将月份的天数补偿为 28、29（闰年）、30 和 31 天。并且还可以进行夏令时补偿。

其它 32 位寄存器还包含可编程的闹钟亚秒、秒、分钟、小时、星期几和日期。

此外，还可以使用数字校准功能对晶振精度的偏差进行补偿。

备份域复位后，所有 RTC 寄存器都会受到保护，以防止可能的非正常写访问。

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。

29.2 RTC 主要特性

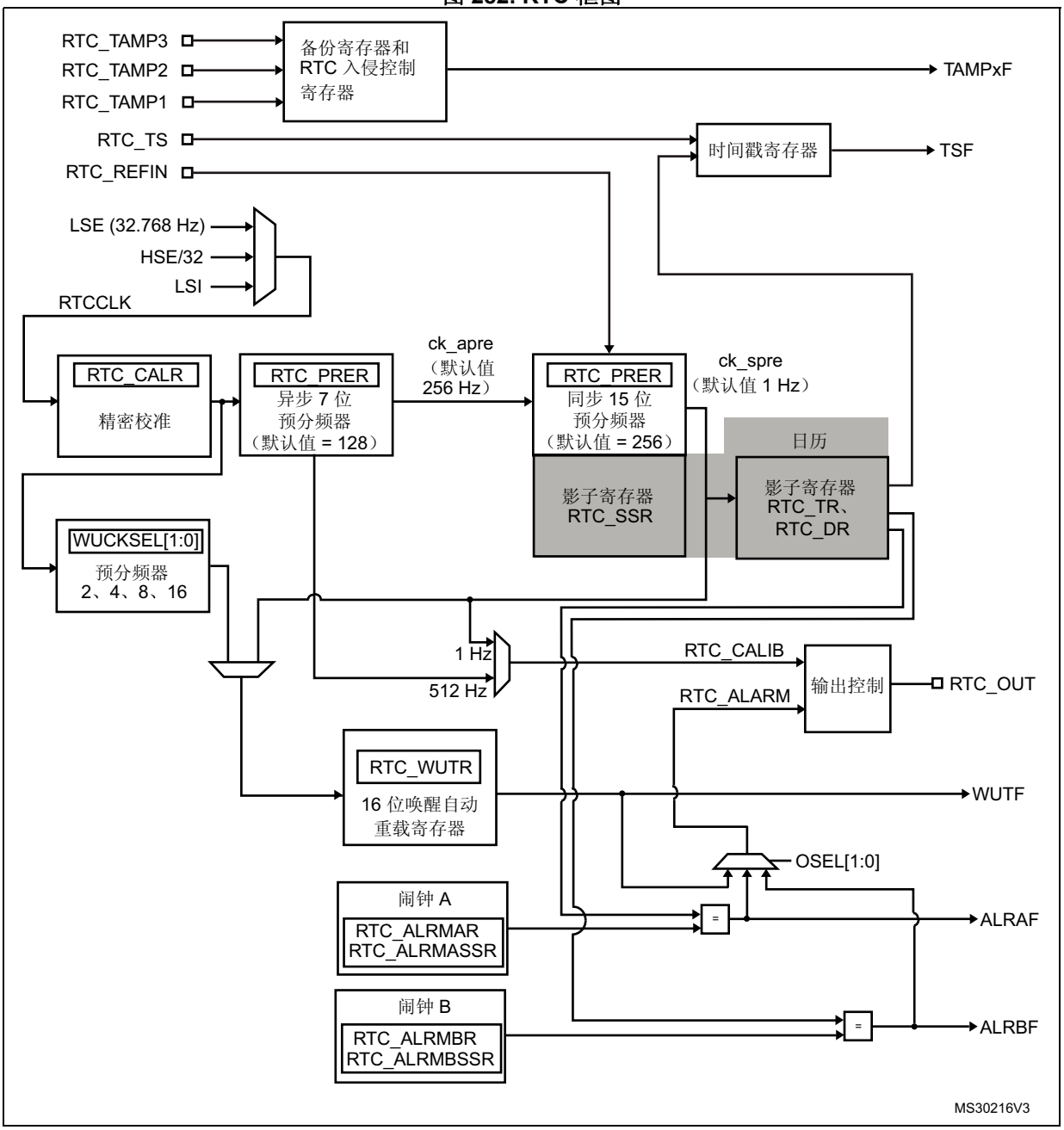
RTC 单元的主要特性如下（参见*图 282：RTC 框图*）：

- 包含亚秒、秒、分钟、小时（12/24 小时制）、星期几、日期、月份和年份的日历。
- 软件可编程的夏令时补偿。
- 具有中断功能的可编程闹钟。可通过任意日历字段的组合触发闹钟。
- 自动唤醒单元，可周期性地生成标志以触发自动唤醒中断。
- 参考时钟检测：可使用更加精确的第二时钟源（50 Hz 或 60 Hz）来提高日历的精确度。
- 利用亚秒级移位特性与外部时钟实现精确同步。
- 数字校准电路（周期性计数器调整）：精度为 0.95 ppm，在数秒钟的校准窗口中获得。
- 用于事件保存的时间戳功能。
- 带可配置过滤器和内部上拉的入侵检测事件。
- 可屏蔽中断/事件：
  - 闹钟 A
  - 闹钟 B
  - 唤醒中断
  - 时间戳
  - 入侵检测
- 32 备份寄存器。

29.3 RTC 功能说明

29.3.1 RTC 框图

图 282. RTC 框图



RTC 包含：

- 两个闹钟
- 三个入侵事件
- 32 个 32 位备份寄存器
  - 备份寄存器 (RTC_BKPxR) 在 RTC 域中实现，可在 VDD 电源关闭时通过 VBAT 保持上电状态。
- 复用功能输出 RTC_OUT，可选择以下两个输出之一：
  - RTC_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。
  - RTC_ALARM：可通过配置 RTC_CR 寄存器中的 OSEL[1:0] 位使能此输出，可选择闹钟 A、闹钟 B 或唤醒输出。
- 复用功能输入：
  - RTC_TS：时间戳事件
  - RTC_TAMP1：tamper1 事件检测
  - RTC_TAMP2：tamper2 事件检测
  - RTC_TAMP3：tamper3 事件检测
  - RTC_REFIN：50 Hz 或 60 Hz 参考时钟输入

29.3.2 RTC 控制的 GPIO

RTC_OUT、RTC_TS 和 RTC_TAMP1 映射到同一引脚 (PC13)。无论 PC13 的 GPIO 配置如何，PC13 引脚的配置都由 RTC 控制。映射在 PC13 上的 RTC 功能可用于所有低功耗模式和 VBAT 模式。

输出机制遵循*表 146* 中所示的优先级顺序。

表 146. RTC 引脚 PC13 配置(1)

PC13 引脚配置和函数	OSEL[1:0] 位（RTC_ALARM 输出使能）	COE 位（RTC_CALIB 输出使能）	RTC_ALARM_TYPE 位	TAMP1E 位（RTC_TAMP1 输入使能）	TSE 位（RTC_TS 输入使能）	TSINSEL 位
RTC_ALARM 输出 OD	01 或 10 或 11	无关	0	无关	无关	无关
RTC_ALARM 输出 PP	01 或 10 或 11	无关	1	无关	无关	无关
RTC_CALIB 输出 PP	00	1	无关	无关	无关	无关
RTC_TAMP1 输入浮空	00	0	无关	1	0	无关
	00	1				
	01 或 10 或 11	0				

表 146. RTC 引脚 PC13 配置(1)（续）

PC13 引脚配置和函数	OSEL[1:0] 位（RTC_ALARM 输出使能）	COE 位（RTC_CALIB 输出使能）	RTC_ALARM_TYPE 位	TAMP1E 位（RTC_TAMP1 输入使能）	TSE位（RTC_TS 输入使能）	TSINSEL 位
RTC_TS 和 RTC_TAMP1 输入浮空	00	0	无关	1	1	00
	00	1				
	01 或 10 或 11	0				
RTC_TS 输入浮空	00	0	无关	0	1	00
	00	1				
	01 或 10 或 11	0				
唤醒引脚或标准 GPIO	00	0	无关	0	0	无关

1. OD：开漏；PP：推挽。

RTC_TAMP2 和 RTC_TS 映射到同一引脚 (PI8)。无论 PI8 的 GPIO 配置如何，PI8 引脚的配置都由 RTC 控制。映射在 PI8 上的 RTC 功能可用于所有低功耗模式和 VBAT 模式。

输出机制遵循*表 147* 中所示的优先级顺序。

表 147. RTC 引脚 PI8 配置

PI8 引脚配置和函数	TAMP2E 位（RTC_TAMP2 输入使能）	TSE位（RTC_TS 输入使能）	TSINSEL位（时间戳引脚选择）
RTC_TAMP2 输入浮空	1	0	无关
RTC_TS 和 RTC_TAMP2 输入浮空	1	1	01
RTC_TS 输入浮空	0	1	01
唤醒引脚或标准 GPIO	0	0	无关

RTC_TAMP3 和 RTC_TS 映射到同一引脚 (PC1)。无论 PC1 的 GPIO 配置如何，PC1 引脚的配置都由 RTC 控制。映射在 PC1 上的 RTC 功能可用于所有低功耗模式，但不能用于 VBAT 模式。

表 148. RTC 引脚 PC2 配置

PC2 引脚配置和函数	TAMP2E 位（RTC_TAMP3 输入使能）	TSE 位（RTC_TS 输入使能）	TSINSEL 位（时间戳引脚选择）
RTC_TAMP3 输入浮空	1	0	无关
RTC_TS 和 RTC_TAMP3 输入浮空	1	1	10 或 11
RTC_TS 输入浮空	0	1	01
唤醒引脚或标准 GPIO	0	0	无关

RTC_REFIN 映射到 PB15。要使用 RTC_REFIN 功能，必须将 PB15 配置为复用功能模式。RTC_REFIN 不可用于 VBAT 模式和待机模式。

下表汇总了所有模式下的 RTC 引脚和功能。

表 149. 各个模式下的 RTC 功能

引脚	RTC 功能	待机模式以外的所有低功耗模式下的功能	待机模式下的功能性	VBAT 模式下的功能性
PC13	RTC_TAMP1 RTC_TS RTC_OUT	是	是	是
PI8	RTC_TAMP2 RTC_TS	是	是	是
PC1	RTC_TAMP3 RTC_TS	是	是	否
PB15	RTC_REFIN	是	否	否

29.3.3 时钟和预分频器

RTC 时钟源 (RTCCLK) 通过时钟控制器从 LSE 时钟、LSI 振荡器时钟以及 HSE 时钟三者中选择。有关 RTC 时钟源配置的更多信息，请参见*第 5 节：复位和时钟控制 (RCC)*。

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 282：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： *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 29.3.6 节：周期性自动唤醒*）。

29.3.4 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，这样可避免等待同步的持续时间。

- RTC_SSR 对应于亚秒
- RTC_TR 对应于时间
- RTC_DR 对应于日期

每隔两个 RTCCLK 周期，便将当前日历值复制到影子寄存器，并将 RTC_ISR 寄存器的 RSF 位置 1（请参见*第 29.6.4 节：RTC 初始化和状态寄存器 (RTC_ISR)*）。在停机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 2 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。

29.3.5 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。可通过 RTC_CR 寄存器中的 ALRAIE 位来使能闹钟中断。

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 RTC_ALARM 输出。可通过 RTC_CR 寄存器的 POL 位配置 RTC_ALARM 输出极性。

29.3.6 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。

唤醒定时器的时钟输入可以是：

- 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。
- ck_spre（通常为 1 Hz 内部时钟）。
  当 ck_spre 频率为 1 Hz 时，可得到的唤醒时间为 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} 添加到 16 位计数器当前值。完成初始化序列后（请参见 *第 814 页上的编程唤醒定时器*），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_ISR 寄存器的 WUTF 标志会置 1，并且唤醒寄存器会使用其重载值（RTC_WUTR 寄存器值）动重载。

之后必须用软件清零 WUTF 标志。

通过将 RTC_CR2 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 RTC_ALARM 输出。可通过 RTC_CR 寄存器的 POL 位配置 RTC_ALARM 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。

29.3.7 RTC 初始化和配置

RTC 寄存器访问

RTC 寄存器为 32 位寄存器。除了当 BYPSHAD=0 时对日历影子寄存器执行的读访问之外，APB 接口会在访问 RTC 寄存器时引入 2 个等待周期。

RTC 寄存器写保护

系统复位后，可通过对 PWR_CR1 寄存器中的 DBP 位清零来保护 RTC 寄存器以防止非正常的写访问。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，所有 RTC 寄存器均受到写保护。通过向写保护寄存器 (RTC_WPR) 写入一个密钥来使能对 RTC 寄存器的写操作。

要解锁所有 RTC 寄存器（RTC_TAMPCR、RTC_BKPxR、RTC_OR 和 RTC_ISR[13:8] 除外）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。

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_ISR 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_ISR 寄存器中的 INITF 位。当 INITF 置 1 时进入初始化阶段模式。大约需要 2 个 RTCCLK 时钟周期（由于时钟同步）。
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。
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，在 4 个 RTCCLK 时钟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。

注： *系统复位后，应用可读取 RTC_ISR 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，表明日历尚未初始化，因为年份字段设置为其备份域复位时的默认值 (0x00)。*

要在初始化之后读取日历，必须首先用软件检查 RTC_ISR 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)。
3. 将 RTC_CR 寄存器中的 ALRAE 位置 1 以再次使能闹钟 A。

注： *约 2 个 RTCCLK 时钟周期（由于时钟同步）后，将执行对 RTC_CR 寄存器的更改。*

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. 清零 RTC_CR 中的 WUTE 以禁止唤醒定时器。
2. 轮询 RTC_ISR 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。大约需要 2 个 RTCCLK 时钟周期（由于时钟同步）。
3. 编程唤醒自动重载值 WUT[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。由于时钟同步的原因，WUTWF 位在 WUTE 清零后的 2 个 RTCCLK 时钟周期后清零。

29.3.8 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行为的安全性。

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_ISR 寄存器中的 RSF 位都会置 1。每两个 RTCCLK 周期执行一次复制。为确保这 3 个值来自同一时刻点，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 2 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（停机模式或待机模式）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 814 页上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 29.3.10 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（停机模式或待机模式）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： *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。*

29.3.9 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_ISR) 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。

相反，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）、RTC 入侵和复用功能配置寄存器 RTC_TAMPCR)、RTC 备份寄存器 (RTC_BKPxR)、唤醒定时器寄存器 (RTC_WUTR)、闹钟 A 和闹钟 B 寄存器（RTC_ALRMASSR/RTC_ALRMAR 和 RTC_ALRMBSSR/RTC_ALRMBR）以及选项寄存器 (RTC_OR)。

此外，如果复位源不是备份域复位源，则发生系统复位时，RTC 会继续工作。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。

29.3.10 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。

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。

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/(PREDIV_S + 1) 秒的分辨率对 RTC_SHIFTR 进行写操作，将时钟平移（延迟或提前）最长 1 秒。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：这将使时钟产生延迟。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，因此将使时钟提前。

小心： 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。

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，硬件会将将该位清零。

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。

29.3.11 RTC 参考时钟检测

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，都会行强制输出 ck_apre 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： *RTC_REFIN 时钟检测在待机模式下不可用。*

29.3.12 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 RTCCLK 脉冲）进行频率校正。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。

当输入频率为 32768 Hz 时，精密数字校准的周期约为 2^{20} 个 RTCCLK 脉冲或 32 秒。此周期由一个通过 RTCCLK 提供时钟信号的 20 位计数器 cal_cnt[19:0] 维持。

精密数字校准寄存器 (RTC_CALR) 可指定 32 秒周期内要减少的 RTCCLK 时钟周期数：

- 将位 CALM[0] 置 1 时，32 秒周期内将只

减少 1 个脉冲。

- 将 CALM[1] 置 1 时，将减少两个周期。
- 将 CALM[2] 置 1 时，将减少 4 个周期。
- 依此类推，将 CALM[8] 置 1 时，将减少 256 个时钟。

注： *CALM[8:0] (RTC_CALR) 可指定 32 秒周期内要减少的 RTCCLK 脉冲数。将位 CALM[0] 置 1 时，32 秒周期内将只减少 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 RTCCLK 周期有效插入一个额外的 RTCCLK 脉冲，这意味着每 32 秒周期可增加 512 个时钟。

与 CALM 和 CALP 配合使用时，可在 32 秒周期内增加一个范围为 -511 到 +512 RTCCLK 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})：

$$F_{CAL} = F_{RTCCLK} \times [1 + (CALP \times 512 - CALM) / (2^{20} + CALM - CALP \times 512)]$$

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。

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 RTCCLK 时钟周期，这意味着每 32 秒可增加 256 个时钟周期。因此，仅使用 CALM 位，可在每 32 秒内有效增加 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）。唯一相关的其它情况是，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

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_ISR/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_ISR/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。

29.3.13 时间戳功能

将 RTC_CR 寄存器的 TSE 位或 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 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。切换至 VBAT 电源可生成内部时间戳事件。

由于内部事件或外部事件而发生时间戳事件时，RTC_ISR 寄存器中的时间戳标志位 (TSF) 将置 1。如果是内部事件，RTC_ISR 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，可在发生入侵检测事件时生成中断。

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： *在发生由同步过程引发的时间戳事件后，TSF 在 2 个 ck_apre 周期置为 1。*

TSOVF 的产生中不存在延迟。也就是说如果两个时间戳事件接连发生，TSOVF 可能为“1”而 TSF 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。

此外，入侵事件可能导致时间戳被记录。有关 TAMPTS 控制位的说明，请参见*第 29.6.14 节：RTC 时间戳亚秒寄存器 (RTC_TSSSR)*。

29.3.14 入侵检测

RTC_TAMPx 输入事件既可配置为边沿检测，也可配置为带过滤的电平检测。

入侵检测可配置用于以下目的：

- 擦除 RTC 备份寄存器（默认配置）
- 生成中断，能够从停止模式和待机模式唤醒
- 为低功耗定时器生成硬件触发

RTC 备份寄存器

备份寄存器 (RTC_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。

备份寄存器在发生入侵检测事件（请参见*第 29.6.20 节：RTC 备份寄存器 (RTC_BKPxR)* 和*第 820 页上的入侵检测初始化*）或者在将 Flash 的读保护从级别 1 更改为级别 0 时复位，TAMPxNOERASE 位置 1 时或者 RTC_TAMPCR 寄存器中的 TAMPxMF 置 1 时除外。

入侵检测初始化

可通过将 RTC_TAMPCR 寄存器中相应的 TAMPxE 位置 1 来使能各输入。

每个 RTC_TAMPx 入侵检测输入分别与 RTC_ISR 寄存器中的标志 TAMPxF 相关联。

将 TAMPxMF 清零时：

TAMPxF 标志将在引脚上出现入侵事件后使能，并存在下述延迟：

- 当 TAMPFLT 不为 0x0 时（带过滤的电平检测），延迟为 3 个 ck_apre 周期
- 当 TAMPTS=1 时（入侵事件的时间戳），延迟为 3 个 ck_apre 周期
- 当 TAMPFLT=0x0（边沿检测）且 TAMPTS=0 时，无延迟

在此期间，只要 TAMPxF 置 1，就无法检测到同一引脚上出现的新入侵。

将 TAMPxMF 置 1 时：

在上述延迟期间以及在 2.5 个 ck_rtc 附加周期内，无法检测到同一引脚上出现的新入侵。

通过将 RTC_TAMPCR 寄存器中的 TAMPIE 位置 1，可在发生入侵检测事件时（当 TAMPxF 置 1 时）生成中断。当一个或多个 TAMPxMF 置 1 时，不允许将 TAMPIE 置 1。

将 TAMPIE 清零时，可通过将 RTC_TAMPCR 寄存器中相应的 TAMPxIE 位置 1 来分别使能各个入侵引脚事件中断。当相应的 TAMPxMF 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 RTC_TAMPCR 寄存器中的 TAMPxMF 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 TAMPxMF 位置 1 时，TAMPxF 标志会被屏蔽，并在 RTC_ISR 寄存器中保持清零。此配置允许在停止模式下自动触发低功耗定时器，无需通过系统唤醒来执行 TAMPxF 清零。在这种情况下，备份寄存器不清零。

入侵事件的时间戳

当 TAMPTS 置“1”时，任何入侵事件都会导致时间戳事件发生。在这种情况下，如同发生正常时间戳事件一样，RTC_ISR 中的 TSF 位或 TSOVF 位会置 1。在 TSF 或 TSOVF 置 1 的同时，受影响的入侵标志寄存器 TAMPxF 也会随之置 1。

对入侵输入的边沿检测

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，RTC_TAMPx 引脚将生成入侵检测事件。选择边沿检测时，会禁止 RTC_TAMPx 输入上的内部上拉电阻。

小心： 使用边沿检测时，建议在使能入侵检测后（通过读取 GPIO 寄存器）以及向备份寄存器中写入敏感值前立即通过软件检查入侵引脚电平，以确保使能入侵事件检测后才出现有效边沿。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 (RTC_BKPxR) 之前禁止 RTC_TAMPx 复用功能，然后再重新使能（TAMPxE 置 1）。这可防止应用在 RTC_TAMPx 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 RTC_TAMPx 复用功能输入的电平检测。

注： *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 RTC_TAMPx 复用功能映射到的引脚从外部连接到正确的电平。*

对 RTC_TAMPx 输入的带过滤电平检测

通过将 TAMPFLT 设置为非零值可执行带过滤的电平检测。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 RTC_TAMPx 输入预充电。预充电的持续时间由 TAMPPRCH 位确定，允许增大 RTC_TAMPx 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： *有关上拉电阻的电气特性，请参见数据手册。*

29.3.15 校准时钟输出

将 RTC_CR 寄存器中的 COE 位置 1 时，会在 RTC_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 RTC_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。RTC_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。

如果 COSEL 置 1 且“PREDIV_S+1”为 256 的非零整数倍（比如：PREDIV_S[7:0] = 0xFF），则 RTC_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： *选择 RTC_CALIB 或 RTC_ALARM 输出时，RTC_OUT 引脚会自动配置为输出复用功能。*

29.3.16 闹钟输出

RTC_CR 寄存器中的 OSEL[1:0] 控制位用于激活闹钟复用功能输出 RTC_ALARM，以及选择输出的功能。这些功能可反映 RTC_ISR 寄存器中相应标志的内容。

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。

闹钟复用功能输出

使用 RTC_OR 寄存器中的控制位 RTC_ALARM_TYPE 可将 RTC_ALARM 引脚配置为输出开漏或输出上拉。

注： *使能 RTC_ALARM 输出之后，其优先级高于 RTC_CALIB（与 COE 位无关，该为必须保持清零）。*

选择 RTC_CALIB 或 RTC_ALARM 输出时，RTC_OUT 引脚会自动配置为输出复用功能。

29.4 RTC 低功耗模式

表 150. 低功耗模式对 RTC 的作用

模式	说明
睡眠	无操作 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 闹钟、RTC 入侵事件、RTC 时间戳事件和 RTC 唤醒会使器件退出停机模式。
待机	当 RTC 时钟源为 LSE 或 LSI 时，RTC 保持工作状态。RTC 闹钟、RTC 入侵事件、RTC 时间戳事件和 RTC 唤醒会使器件退出待机模式。

29.5 RTC 中断

所有 RTC 中断均与 EXTI 控制器相连。请参见*第 11 节：外部中断和事件控制器 (EXTI)*。

要使能 RTC 闹钟中断，需按照以下顺序操作：

1. 将对应于 RTC 闹钟事件的 EXTI 线配置为中断模式并将其使能，然后选择上升沿有效。
2. 配置 NVIC 中的 RTC_ALARM IRQ 通道并将其使能。
3. 配置 RTC 以生成 RTC 闹钟。

要使能 RTC 入侵中断，需按照以下顺序操作：

1. 将对应于 RTC 入侵事件的 EXTI 线配置为中断模式并将其使能，然后选择上升沿有效。
2. 配置 NVIC 中的 RTC_TAMP_STAMP IRQ 通道并将其使能。
3. 配置 RTC 以检测 RTC 入侵事件。

要使能 RTC 时间戳中断，需按照以下顺序操作：

1. 将对应于 RTC 时间戳事件的 EXTI 线配置为中断模式并将其使能，然后选择上升沿有效。
2. 配置 NVIC 中的 RTC_TAMP_STAMP IRQ 通道并将其使能。
3. 配置 RTC 以检测 RTC 时间戳事件。

要使能唤醒定时器中断，需按照以下顺序操作：

1. 将对应于唤醒定时器事件的 EXTI 线配置为中断模式并将其使能，然后选择上升沿有效。
2. 配置 NVIC 中的 RTC_WKUP IRQ 通道并将其使能。
3. 配置 RTC 以检测 RTC 唤醒定时器事件。

表 151. 中断控制位

中断事件	事件标志	启用控制位	退出睡眠模式	退出停止模式	退出待机模式
闹钟 A	ALRAF	ALRAIE	是	是(1)	是(1)
闹钟 B	ALRBF	ALRBE	是	是(1)	是(1)
RTC_TS 输入（时间戳）	TSF	TSIE	是	是(1)	是(1)
RTC_TAMP1 输入检测	TAMP1F	TAMPIE	是	是(1)	是(1)
RTC_TAMP2 输入检测	TAMP2F	TAMPIE	是	是(1)	是(1)
RTC_TAMP3 输入检测	TAMP3F	TAMPIE	是	是(1)	是(1)
唤醒定时器中断	WUTF	WUTIE	是	是(1)	是(1)

1. 仅当 RTC 时钟源为 LSE 或 LSI 时，才能从停机和待机模式唤醒。

29.6 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 58 页上的第 1.1 节*。

外设寄存器可按字（32 位）进行访问。

29.6.1 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 814 页上的日历初始化和配置*和*第 815 页上的读取日历*。

此寄存器受写保护。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

偏移地址：0x00

备份域复位值：0x0000 0000

系统复位：当 BYPSHAD = 0 时为 0x0000 0000。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)

29.6.2 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 814 页上的日历初始化和配置*和*第 815 页上的读取日历*。

此寄存器受写保护。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

偏移地址：0x04

备份域复位值：0x0000 2101

系统复位：当 BYPSHAD = 0 时为 0x0000 2101。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**：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)

29.6.3 RTC 控制寄存器 (RTC_CR)

RTC control register

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.	ITSE	COE	OSEL[1:0]		POL	COSEL	BKP	SUB1H	ADD1H
							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	ALRBE	ALRAIE	TSE	WUTE	ALRBE	ALRAE	Res.	FMT	BYPSHAD	REFCKON	TSEDGE	WUCKSEL[2:0]		
rw	rw	rw	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 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 RTC_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 RTC_ALARM 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 RTC_ALARM 输出的极性
0：当 ALRAF/ALRBF/WUTF 置 1 时（取决于 OSEL[1:0]），该引脚为高电平
1：当 ALRAF/ALRBF/WUTF 置 1 时（取决于 OSEL[1:0]），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 RTC_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 29.3.15 节：校准时钟输出*。

位 18 **BKP**：备份 (Backup)
用户可对此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。此位始终读为 0。
当前小时为 0 时，将此位置 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。此位始终读为 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 (Time-stamp 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 (Wakeup 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 (Wakeup timer enable)
0：禁止唤醒定时器
1：使能唤醒定时器

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 保留，必须保持复位值。

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： 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： 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]**：唤醒时钟选择 (Wakeup 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：选择 ck_spre 时钟（通常为 1 Hz）
11x：选择 ck_spre 时钟（通常为 1 Hz）并将 WUT 计数器值增加 2^{16}（见下面的注释）

注： *只能在初始化模式下 (RTC_ISR/INITF = 1) 对该寄存器的位 7、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 to 0xFFFF) + 0x10000（增加的值）。

只能在 RTC_CR WUTE 位 = 0 且 RTC_ISR 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，因为这样做会屏蔽日历小时的增量。

ADD1H 和 SUB1H 的更改在下一秒生效。

此寄存器受写保护。第 813 页上的RTC 寄存器写保护中介绍了写访问的过程。

29.6.4 RTC 初始化和状态寄存器 (RTC_ISR)

RTC initialization and status register

此寄存器受写保护（RTC_ISR[13:8] 位除外）。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

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.	ITSF	RECALPF
														rc_w0	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAMP3F	TAMP2F	TAMP1F	TSOVF	TSF	WUTF	ALRBF	ALRAF	INIT	INITF	RSF	INITS	SHPF	WUTWF	ALRBWF	ALRAWF
rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rw	r	rc_w0	r	r	r	r	r

位 31:18 保留，必须保持复位值

位 17 **ITSF**：内部时间戳标志 (Internal time-stamp flag)

发生内部时间戳事件时，由硬件将此标志置 1。
该标志由软件写零清零，且必须通过向两个位写零将此标志与 TSF 位一起清零。

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 **TAMP3F**：RTC_TAMP3 检测标志 (RTC_TAMP3 detection flag)

在 RTC_TAMP3 输入上检测到入侵检测事件时，由硬件将此标志置 1。
该标志由软件写零清除

位 14 **TAMP2F**：RTC_TAMP2 检测标志 (RTC_TAMP2 detection flag)
在 RTC_TAMP2 输入上检测到入侵检测事件时，由硬件将此标志置 1。
该标志由软件写零清除

位 13 **TAMP1F**：RTC_TAMP1 检测标志 (RTC_TAMP1 detection flag)
在 RTC_TAMP1 输入上检测到入侵检测事件时，由硬件将此标志置 1。
该标志由软件写零清除

位 12 **TSOVF**：时间戳溢出标志 (Time-stamp 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。
该标志由软件写零清除。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，则溢出事件可能会被漏掉。

位 11 **TSF**：时间戳标志 (Time-stamp flag)
发生时间戳事件时，由硬件将此标志置 1。
该标志由软件写零清除。如果 ITSF 标志已置 1，必须通过向两个位写零将 TSF 与 ITSF 一起清零。

位 10 **WUTF**：唤醒定时器标志 (Wakeup timer flag)
当唤醒自动重载计数器计数到 0 时，由硬件将此标志置 1。
该标志由软件写零清除。
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 9 **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。
该标志由软件写零清除。

位 8 **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。
该标志由软件写零清除。

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）以及预分频器寄存器 (RTC_PRER)。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)
当此位置 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_SSRx、RTC_TRx 和 RTC_DRx）时，都会由硬件将此位置 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)

0：没有平移操作挂起

1：某个平移操作挂起

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。

位 2 **WUTWF**：唤醒定时器写标志 (Wakeup timer write flag)

该位在 RTC_CR 中的 WUTE 位被清零后的最长 2 个 RTCCLK 周期后由硬件置 1；在 WUTE 位被置 1 后的 2 个 RTCCLK 周期后清零。当 WUTE 位清零且 WUTWF 位置 1 时，可更改唤醒定时器的值。

0：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 **ALRBWF**：闹钟 B 写标志 (Alarm B write flag)

在 RTC_CR 寄存器中的 ALRBE 位置 0 之后，当闹钟 B 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：不允许更新闹钟 B

1：允许更新闹钟 B

位 0 **ALRAWF**：闹钟 A 写标志 (Alarm A write flag)

在 RTC_CR 寄存器中的 ALRAE 位置 0 后，当闹钟 A 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：不允许更新闹钟 A

1：允许更新闹钟 A

注： *将 ALRAF、ALRBF、WUTF 和 TSF 位编程为 0 之后 2 个 APB 时钟周期，清零生效。*

29.6.5 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 814 页上的日历初始化和配置*。

此寄存器受写保护。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

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)

29.6.6 RTC 唤醒定时器寄存器 (RTC_WUTR)

RTC wakeup timer register

仅当 RTC_ISR 中的 WUTWF 置 1 时才可对该寄存器执行写操作。

此寄存器受写保护。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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 保留，必须保持复位值

位 15:0 **WUT[15:0]**：唤醒自动重载值位 (Wakeup 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 之后 (WUT+1) 个 ck_wut 周期。禁止在 WUCKSEL[2:0]=011(RTCCLK/2) 时将 WUT[15:0] 设置为 0x0000。

29.6.7 RTC 闹钟 A 寄存器 (RTC_ALRMAR)

RTC alarm A register

仅当 RTC_ISR 中的 ALRAWF 置 1 时或在初始化模式下，才可以对该寄存器执行写操作。

此寄存器受写保护。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

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
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]**：日期的个位或日（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)

29.6.8 RTC 闹钟 B 寄存器 (RTC_ALRMBR)

RTC alarm B register

仅当 RTC_ISR 中的 ALRBWF 置 1 时或在初始化模式下，才可以对该寄存器执行写操作。

此寄存器受写保护。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

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
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 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)

29.6.9 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							
								w	w	w	w	w	w	w	w

位 31:8 保留，必须保持复位值。

位 7:0 **KEY**：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*。

29.6.10 RTC 亚秒寄存器 (RTC_SSR)

RTC sub second register

偏移地址：0x28

备份域复位值：0x0000 0000

系统复位：当 BYPSHAD = 0 时为 0x0000 0000。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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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 31:16 保留，必须保持复位值

位 15:0 **SS**：亚秒值 (Sub seconds value)
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)
注： 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。

29.6.11 RTC 平移控制寄存器 (RTC_SHIFTR)

RTC shift control register

此寄存器受写保护。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

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

此位为只写位且始终读为 0。当平移操作挂起（RTC_ISR 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**：减少亚秒值 (Subtract a fraction of a second)

此位为只写位且始终读为 0。当平移操作挂起（RTC_ISR 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)))。

注： 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。

29.6.12 RTC 时间戳时间寄存器 (RTC_TSTR)

RTC timestamp time register

仅当 RTC_ISR 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。

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)。

29.6.13 RTC 时间戳日期寄存器 (RTC_TSDR)

RTC timestamp date register

仅当 RTC_ISR 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。

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[1:0]			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[1:0]**：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)

29.6.14 RTC 时间戳亚秒寄存器 (RTC_TSSSR)

RTC time-stamp sub second register

仅当 RTC_ISR/TSF 置 1 时，该寄存器的内容才有效。当 RTC_ISR/TSF 位复位时，清零该寄存器。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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 31:16 保留，必须保持复位值

位 15:0 **SS**：亚秒值 (Sub seconds value)

当发生时间戳事件时，SS[15:0] 是同步预分频器计数器的值。

29.6.15 RTC 校准寄存器 (RTC_CALR)

RTC calibration register

此寄存器受写保护。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

偏移地址：0x3C

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	CALW16	Res.	Res.	Res.	Res.	CALM[8:0]								
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 **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 29.3.12 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： *CALW8=“1”时，CALM[1:0] 将始终保持为“00”。请参见第 29.3.12 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，则不能将该位置“1”。
注： *当 CALW16=“1”时，CALM[0] 将始终保持为“0”。请参见第 29.3.12 节：RTC 精密数字校准。*

位 12:9 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 817 页上的第 29.3.12 节：RTC 精密数字校准*。

29.6.16 RTC 入侵配置寄存器 (RTC_TAMPCR)

RTC tamper configuration register

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP3MF	TAMP3NOERASE	TAMP3IE	TAMP2MF	TAMP2NOERASE	TAMP2IE	TAMP1MF	TAMP1NOERASE	TAMP1IE
							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
TAMPPUDIS	TAMPPRCH[1:0]		TAMPFLT[1:0]		TAMPFREQ[2:0]			TAMPTS	TAMP3TRG	TAMP3E	TAMP2TRG	TAMP2E	TAMPIE	TAMP1TRG	TAMP1E
rw	rw	rw	rw	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 24 **TAMP3MF**：入侵 3 屏蔽标志 (Tamper 3 mask flag)

0：入侵 3 事件生成触发事件，必须通过软件将 TAMP3F 清零来允许下一次入侵事件检测。

1：入侵 3 事件生成触发事件。屏蔽 TAMP3F 并由硬件在内部将其清零。不擦除备份寄存器。

注： TAMP3MF 置 1 时，不得使能入侵 3 中断。

位 23 **TAMP3NOERASE**：入侵 3 不擦除 (Tamper 3 no erase)

0：入侵 3 事件擦除备份寄存器。

1：入侵 3 事件不擦除备份寄存器。

位 22 **TAMP3IE**：入侵 3 中断使能 (Tamper 3 interrupt enable)

0：TAMPIE = 0 时禁止入侵 3 中断。

1：使能入侵 3 中断。

位 21 **TAMP2MF**：入侵 2 屏蔽标志 (Tamper 2 mask flag)

0：入侵 2 事件生成触发事件，如果要允许下一次入侵事件检测，必须通过软件将 TAMP2F 清零。

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。

注： TAMP2MF 置 1 时，不得使能入侵 2 中断。

位 20 **TAMP2NOERASE**：入侵 2 不擦除 (Tamper 2 no erase)

0：入侵 2 事件擦除备份寄存器。

1：入侵 2 事件不擦除备份寄存器。

位 19 **TAMP2IE**：入侵 2 中断使能 (Tamper 2 interrupt enable)

0：TAMPIE = 0 时禁止入侵 2 中断。

1：使能入侵 2 中断。

位 18 **TAMP1MF**：入侵 1 屏蔽标志 (Tamper 1 mask flag)

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。

注： TAMP1MF 置 1 时，不得使能入侵 1 中断。

位 17 **TAMP1NOERASE**：入侵 1 不擦除 (Tamper 1 no erase)

0：入侵 1 事件擦除备份寄存器。

1：入侵 1 事件不擦除备份寄存器。

位 16 **TAMP1IE**：入侵 1 中断使能 (Tamper 1 interrupt enable)

0：TAMPIE = 0 时禁止入侵 1 中断。

1：使能入侵 1 中断。

位 15 **TAMPPUDIS**：RTC_TAMPx 上拉禁止 (RTC_TAMPx pull-up disable)

该位决定在每次采样之前是否对每个 RTC_TAMPx 引脚都进行预充电。

0：采样之前对 RTC_TAMPx 引脚进行预充电（使能内部上拉）

1：禁止对 RTC_TAMPx 引脚进行预充电。

位 14:13 **TAMPPRCH[1:0]**：RTC_TAMPx 预充电持续时间 (RTC_TAMPx 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 RTC_TAMPx 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 12:11 **TAMPFLT[1:0]**：RTC_TAMPx 过滤器计数 (RTC_TAMPx 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 RTC_TAMPx 输入都有效。

0x0：在 RTC_TAMPx 输入转变为有效电平的边沿激活入侵事件（RTC_TAMPx 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 10:8 **TAMPFREQ[2:0]**：入侵采样频率 (Tamper sampling frequency)

决定对每个 RTC_TAMPx 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）

位 7 **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时间戳

1：发生入侵检测事件时保存时间戳

即便 RTC_CR 寄存器中的 TSE=0，TAMPTS 仍有效。

位 6 **TAMP3TRG**：RTC_TAMP3 输入的电平有效 (Active level for RTC_TAMP3 input)

如果 TAMPFLT ≠ 00：

0：RTC_TAMP3 输入保持低电平会触发入侵检测事件。

1：RTC_TAMP3 输入保持高电平会触发入侵检测事件。

如果 TAMPFLT = 00：

0：RTC_TAMP3 输入上升沿会触发入侵检测事件。

1：RTC_TAMP3 输入下降沿会触发入侵检测事件。

注： 使用 VBAT 的开关时不允许进行入侵 3 下降沿检测，否则进入 Vbat 模式后会一直进行检测。

位 5 **TAMP3E**：RTC_TAMP3 检测使能 (RTC_TAMP3 detection enable)

0：禁止 RTC_TAMP3 输入检测

1：使能 RTC_TAMP3 输入检测

位 4 **TAMP2TRG**：RTC_TAMP2 输入的有效电平 (Active level for RTC_TAMP2 input)

如果 TAMPFLT != 00：

0：RTC_TAMP2 输入保持低电平会触发入侵检测事件。

1：RTC_TAMP2 输入保持高电平会触发入侵检测事件。

如果 TAMPFLT = 00：

0：RTC_TAMP2 输入上升沿会触发入侵检测事件。

1：RTC_TAMP2 输入下降沿会触发入侵检测事件。

位 3 **TAMP2E**：RTC_TAMP2 输入检测使能 (RTC_TAMP2 input detection enable)

0：禁止 RTC_TAMP2 检测

1：使能 RTC_TAMP2 检测

位 2 **TAMPIE**：入侵中断使能 (Tamper interrupt enable)

0：禁止入侵中断

1：使能入侵中断

注： 该位使能所有入侵引脚事件的中断，无论 TAMPxIE 电平如何。如果将该位清零，可以通过将 TAMPxIE 置 1 分别使能每个入侵事件中断。

位 1 **TAMP1TRG**：RTC_TAMP1 输入的有效电平 (Active level for RTC_TAMP1 input)

如果 TAMPFLT != 00

0：RTC_TAMP1 输入保持低电平会触发入侵检测事件。

1：RTC_TAMP1 输入保持高电平会触发入侵检测事件。

如果 TAMPFLT = 00：

0：RTC_TAMP1 输入上升沿会触发入侵检测事件。

1：RTC_TAMP1 输入下降沿会触发入侵检测事件。

位 0 **TAMP1E**：RTC_TAMP1 输入检测使能 (RTC_TAMP1 input detection enable)

0：禁止 RTC_TAMP1 检测

1：使能 RTC_TAMP1 检测

小心： 如果 TAMPFLT = 0，则更改 TAMP1TRG 时必须复位 TAMP1E，以避免将 TAMP1F 意外置 1。

29.6.17 RTC 闹钟 A 亚秒寄存器 (RTC_ALRMASSR)

RTC alarm A sub second 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。

此寄存器受写保护。*第 813 页上的RTC 寄存器写保护*中介绍了写访问的过程。

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

位 31:28 保留，必须保持复位值。

位 27:24 MASKSS[3:0]：屏蔽从此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（假定其余字段均匹配）。
1：在闹钟 A 比较中，SS[14:1] 为无关位。仅比较 SS[0]。
2：在闹钟 A 比较中，SS[14:2] 为无关位。仅比较 SS[1:0]。
3：在闹钟 A 比较中，SS[14:3] 为无关位。仅比较 SS[2:0]。
...
12：在闹钟 A 比较中，SS[14:12] 为无关位。比较 SS[11:0]。
13：在闹钟 A 比较中，SS[14:13] 为无关位。比较 SS[12:0]。
14：在闹钟 A 比较中，SS[14] 为无关位。比较 SS[13:0]。
15：所有 15 个 SS 位均进行比较，并且必须全部匹配才能激活闹钟。
同步计数器的溢出位（位 15）从不进行比较。仅当执行平移操作之后，此位才不为 0。

位 23:15 保留，必须保持复位值。

位 14:0 SS[14:0]：亚秒值 (Sub seconds 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。

29.6.18 RTC 闹钟 B 亚秒寄存器 (RTC_ALRMBSSR)

RTC alarm B sub second 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。

该寄存器受写保护。*RTC 寄存器写保护*一节中介绍了写访问的过程。

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

位 31:28 保留，必须保持复位值。

位 27:24 **MASKSS[3:0]**：屏蔽从此位开始的最高有效位 (Mask the most-significant bits starting at this bit)

0x0：不对闹钟 B 的亚秒进行比较。当秒单元递增时设置闹钟（假定其余字段均匹配）。
0x1：在闹钟 B 比较中，SS[14:1] 为无关位。仅比较 SS[0]。
0x2：在闹钟 B 比较中，SS[14:2] 为无关位。仅比较 SS[1:0]。
0x3：在闹钟 B 比较中，SS[14:3] 为无关位。仅比较 SS[2:0]。
...
0xC：在闹钟 B 比较中，SS[14:12] 为无关位。比较 SS[11:0]。
0xD：在闹钟 B 比较中，SS[14:13] 为无关位。比较 SS[12:0]。
0xE：在闹钟 B 比较中，SS[14] 为无关位。比较 SS[13:0]。
0xF：所有 15 个 SS 位均进行比较，并且必须全部匹配才能激活闹钟。
同步计数器的溢出位（位 15）从不进行比较。仅当执行平移操作之后，此位才不为 0。

位 23:15 保留，必须保持复位值。

位 14:0 **SS[14:0]**：亚秒值 (Sub seconds 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。

29.6.19 RTC 选项寄存器 (RTC_OR)

RTC option register

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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.	RTC_ALARM_TYPE	TSINSEL[1:0]		Res.
												rw	rw	rw	

位 31:4 保留，必须保持复位值。

位 3 **RTC_ALARM_TYPE**：RTC_ALARM 在 PC13 上的输出类型 (RTC_ALARM on PC13 output type)

0：RTC_ALARM 在映射到 PC13 上时为开漏输出

1：RTC_ALARM 在映射到 PC13 上时为推挽输出

位 2:1 **TSINSEL[1:0]**：TIMESTAMP 映射 (TIMESTAMP mapping)

00：TIMESTAMP 映射到 PC13 上

01：TIMESTAMP 映射到 PI8 上

10：TIMESTAMP 映射到 PC1 上

11：TIMESTAMP 映射到 PC1 上

位 0 保留，必须保持复位值。

29.6.20 RTC 备份寄存器 (RTC_BKPxR)

RTC backup registers

偏移地址：0x50 到 0xCC

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。

当 V_{DD} 关闭时，这些寄存器由 V_{BAT} 供电，因而系统复位时，这些寄存器不会复位，并且当器件在低功耗模式下工作时，寄存器的内容仍然有效。

发生入侵检测事件时该寄存器会被复位，并且只要 TAMPxF=1，该寄存器就一直保持复位。或者在禁止闪存读出保护时复位。

29.6.21 RTC 寄存器映射

表 152. RTC 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	**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_CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITSE	COE	OSEL [1:0]		POL	COSEL	BKP	SUB1H	ADD1H	TSIE	WUTIE	ALRBE	ALRAIE	TSE	WUTE	ALRBE	ALRAE	Res.	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
0x0C	**RTC_ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITSF	RECALPF	TAMP3F	TAMP2F	TAMP1F	TSOVF	TSF	WUTF	ALRBF	ALRAF	INIT	INITF	RSF	INITS	SHPF	WUTWF	ALRBWF	ALRAWF
	Reset value															0	0		0	0	0	0	0	0	0	0	0	0	0	0	1	1	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**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUT[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x1C	**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
0x20	**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]				MSK2	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
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							
	Reset value																									0	0	0	0	0	0	0	0
0x28	**RTC_SSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SS[15:0]															
	Reset value																	0	0	0	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
0x34	**RTC_TSDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDU[1:0]			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

表 152. RTC 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
0x38	**RTC_TSSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SS[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	**RTC_ CALR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CALP	CALW8	CALW16	Res.	Res.	Res.	Res.	CALM[8:0]								
	Reset value																	0	0	0					0	0	0	0	0	0	0	0	0
0x40	**RTC_TAMPCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP3MF	TAMP3NOERASE	TAMP3IE	TAMP2MF	TAMP2NOERASE	TAMP2IE	TAMP1MF	TAMP1NOERASE	TAMP1IE	TAMPPUDIS	TAMPPRCH[1:0]		TAMPFLT[1:0]		TAMPFREQ[2:0]			TAMPTS	TAMP3-TRG	TAMP3E	TAMP2-TRG	TAMP2E	TAMPIE	TAMP1TRG	TAMP1E
	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
0x44	**RTC_ ALRMASSR**	Res.	Res.	Res.	Res.	MASKSS [3:0]				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
0x48	**RTC_ ALRMBSSR**	Res.	Res.	Res.	Res.	MASKSS [3:0]				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
0x4C	**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.	RTC_ALARM_TYPE	TSINSEL[1:0]		Res.
	Reset value																													0	0	0	
0x50 to 0xCC	**RTC_BKP0R**	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
	to **RTC_BKP31R**	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 64 页上的第 2.2.2 节*。

30 内部集成电路 (I2C) 接口

30.1 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+)。

它还与 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。

30.2 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

还可额外提供以下特性，具体取决于产品实现（请参见*第 30.3 节：I2C 特性实现*）：

- 兼容 SMBus 规范第 2.0 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 1.1 版标准
- 独立时钟：选择独立时钟源可使 I2C 通信速度不受 PCLK 时钟频率更改的影响

30.3 I2C 特性实现

本手册介绍了 I2C1、I2C2、I2C3 和 I2C4 中实现的所有特性。I2C1、I2C2、I2C3 和 I2C4 完全相同，均实现了下表中列出的全部特性。

表 153. STM32F75xxx 和 STM32F74xxx I2C 特性实现

I2C 特性(1)	I2C1	I2C2	I2C3	I2C4
7 位寻址模式	X	X	X	X
10 位寻址模式	X	X	X	X
标准速度模式（高达 100 kbps）	X	X	X	X
快速模式（高达 400 kbps）	X	X	X	X
超快速模式（高达 1 Mbps）	X	X	X	X
独立时钟	X	X	X	X
SMBus	X	X	X	X

1. X = 支持。

30.4 I2C 功能说明

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。

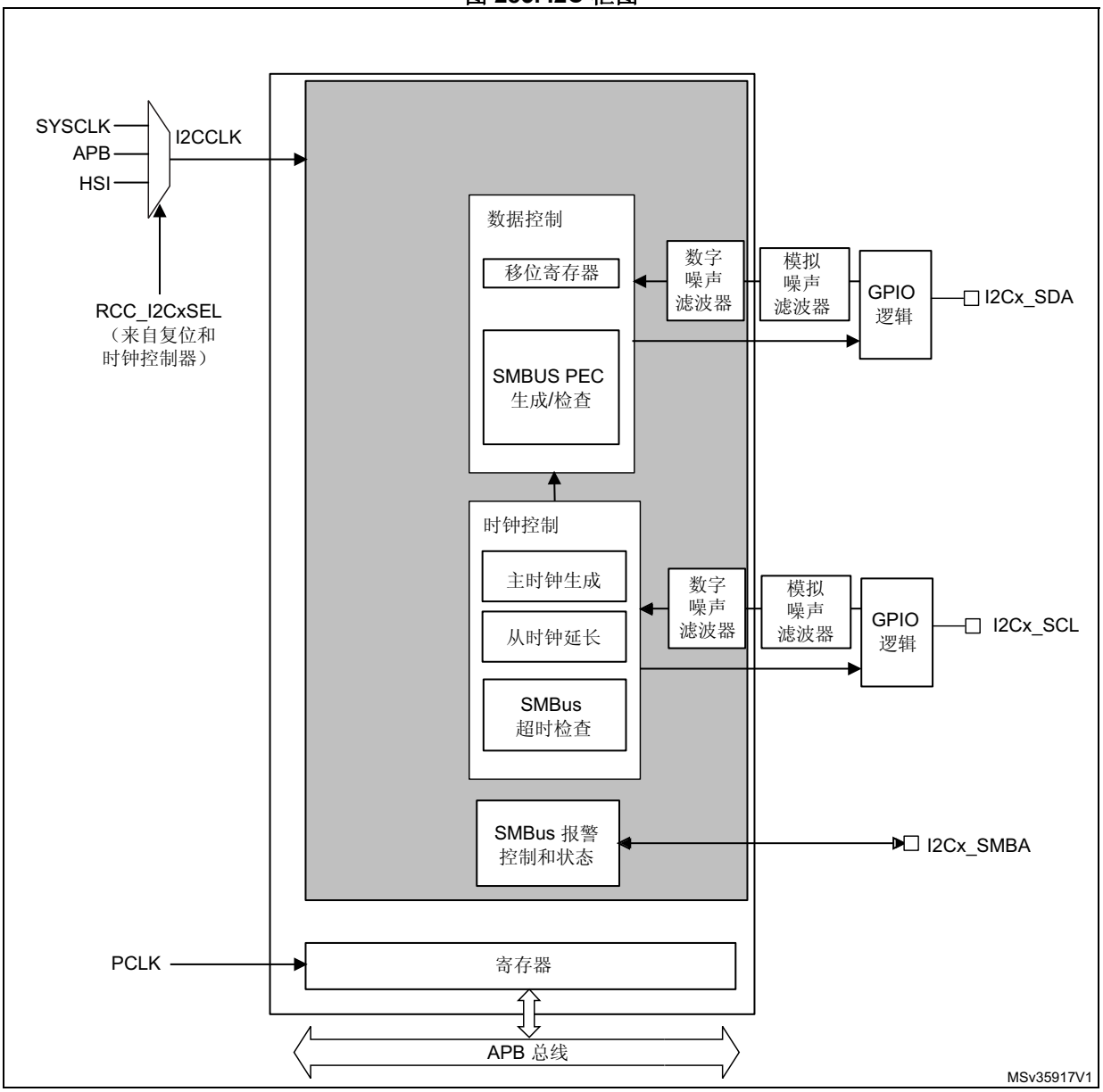
该接口也可通过数据引脚 (SDA) 和时钟引脚 (SCL) 连接到 SMBus。

如果支持 SMBus 功能：还可使用额外的可选 SMBus 报警引脚 (SMBA)。

30.4.1 I2C 框图

I2C 接口如*图 283* 所示。

图 283. I2C 框图



I2C 的时钟由独立时钟源提供，这使得 I2C 能够独立于 PCLK 频率工作。

该独立时钟源可从以下三种时钟源中任选其一：

- PCLK1：APB1 时钟（默认值）
- HSI：高速内部振荡器
- SYSCLK：系统时钟

更多详细信息，请参见*第 5 节：复位和时钟控制 (RCC)*。

30.4.2 I2C 时钟要求

I2C 内核的时钟由 I2CCLK 提供。

I2CCLK 周期 t_{I2CCLK} 必须遵循以下条件：

$t_{I2CCLK} < (t_{LOW} - t_{filters}) / 4$ 且 $t_{I2CCLK} < t_{HIGH}$

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。

模拟滤波器延时最大值为 260 ns。数字滤波器延时为 DNF x t_{I2CCLK}。

PCLK 时钟周期 t_{PCLK} 必须遵循以下条件：

$t_{PCLK} < 4/3\ t_{SCL}$

其中，t_{SCL}：SCL 周期

小心： 当 I2C 内核的时钟由 PCLK 提供时，PCLK 必须遵循 t_{I2CCLK} 的条件。

30.4.3 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

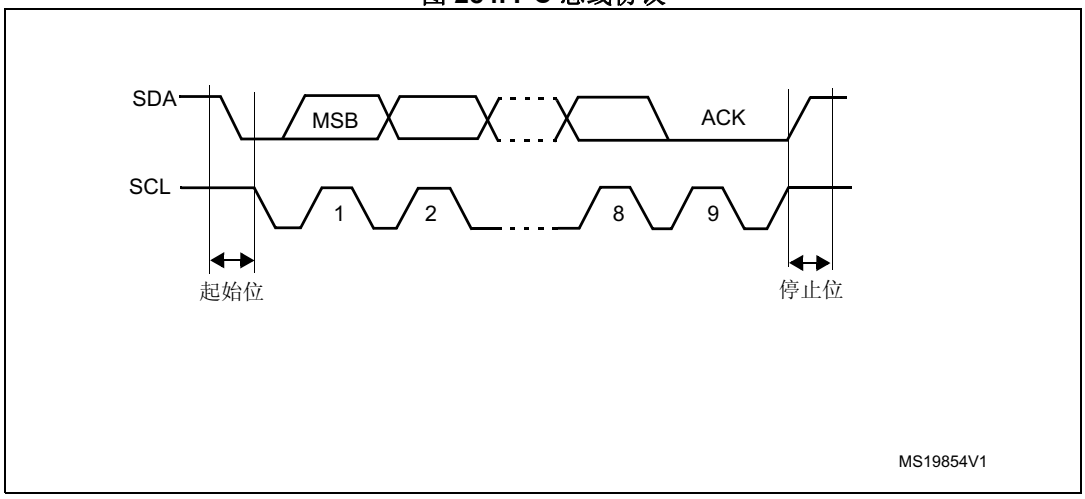
在主模式下，I2C 接口会启动数据传输并生成时钟信号。串行数据传输始终是在出现起始位时开始，在出现停止位时结束。起始位和停止位均在主模式下由软件生成。

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传送。

在字节传输 8 个时钟周期后是第 9 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。请参见下图。

图 284. 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。

30.4.4 I2C 初始化

使能和禁止外设

I2C 外设时钟必须在时钟控制器中进行配置和使能 （请参见*第 5 节：复位和时钟控制 (RCC)*）。

然后可通过将 I2C_CR1 寄存器中的 PE 位置 1 使能 I2C。

当禁止 I2C (PE=0) 时，I²C 将执行软件复位。更多详细信息，请参见*第 30.4.5 节：软件复位*。

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，必须配置噪声滤波器。默认情况下，SDA 和 SCL 输入上集成了模拟噪声滤波器。该模拟滤波器符合 I²C 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。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。

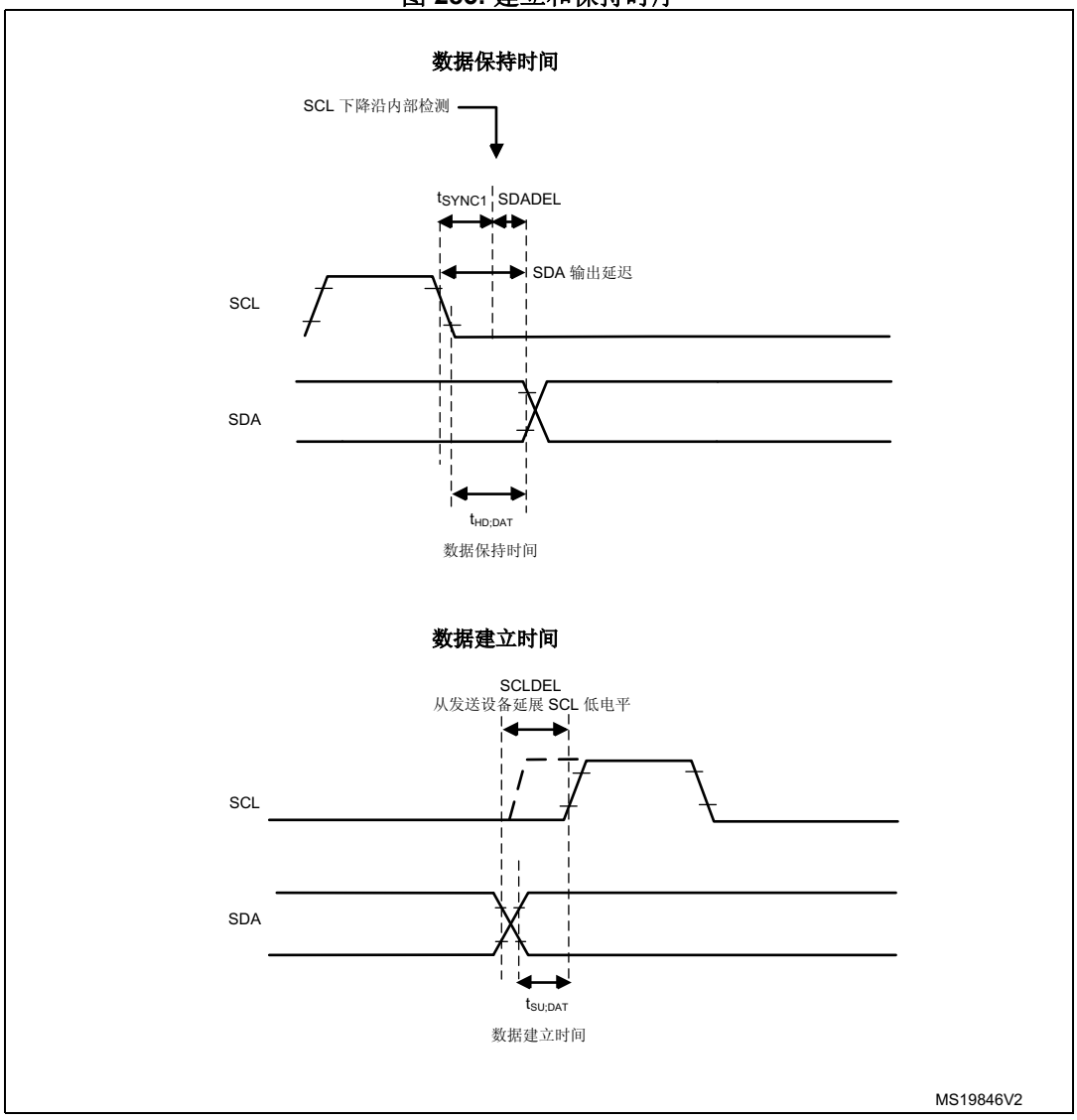
使能数字滤波器时，SCL 或 SDA 线的电平只有在电平稳定时间超过 DNF x I2CCLK 个周期后才会发生内部变化， 从而可抑制的尖峰脉宽在 1 到 15 个 I2CCLK 周期可编程。

小心： 使能 I2C 时，不允许更改滤波器配置。

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。

图 285. 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。该延时为 t_{SDADEL} = SDADEL x t_{PRESC} + t_{I2CCLK}，其中 t_{PRESC} = (PRESC+1) x t_{I2CCLK}。
  T_{SDADEL} 会影响保持时间 $t_{HD;DAT}$。

SDA 总输出延时为：

t_{SYNC1} + {[SDADEL x (PRESC+1) + 1] x t_{I2CCLK} }

t_{SYNC1} 持续时间取决于以下参数：

- SCL 下降斜率
- 模拟滤波器（使能时）引入的输入延时：$t_{AF(min)} < t_{AF} < t_{AF(max)}$ ns
- 数字滤波器（使能时）引入的输入延时：$t_{DNF} = DNF \times t_{I2CCLK}$
- SCL 与 I2CCLK 时钟建立同步而产生的延时（2 到 3 个 I2CCLK 周期）

为了桥接 SCL 下降沿的未定义区域，编程 SDADEL 时必须遵循以下条件：

$$\{t_{f\ (max)} + t_{HD;DAT\ (min)} - t_{AF(min)} - [(DNF + 3) \times t_{I2CCLK}]\} / \{(PRESC + 1) \times t_{I2CCLK}\} \leq SDADEL$$

$$SDADEL \leq \{t_{HD;DAT\ (max)} - t_{AF(max)} - [(DNF+4) \times t_{I2CCLK}]\} / \{(PRESC + 1) \times t_{I2CCLK}\}$$

注： *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。*

标准模式、快速模式和超快速模式下的 $t_{HD;DAT}$ 最大值分别可达 3.45 µs、0.9 µs 和 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，因此在这种情况下，上述公式变成如下形式：

$$SDADEL \leq \{t_{VD;DAT\ (max)} - t_{r\ (max)} - 260\ ns - [(DNF+4) \times t_{I2CCLK}]\} / \{(PRESC + 1) \times t_{I2CCLK}\}$$。

注： *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 154：I2C-SMBUS 规范数据建立和保持时间*。

- 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 $t_{SU;DAT}$。

为了桥接 SDA 跳变沿（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：

$$\{[t_{r\ (max)} + t_{SU;DAT\ (min)}] / [(PRESC+1)] \times t_{I2CCLK}]\} - 1 <= SCLDEL$$

有关 t_r 和 $t_{SU;DAT}$ 标准值的信息，请参见*表 154：I2C-SMBUS 规范数据建立和保持时间*。

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。

表 154. 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 信号的上升时间	-	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}。

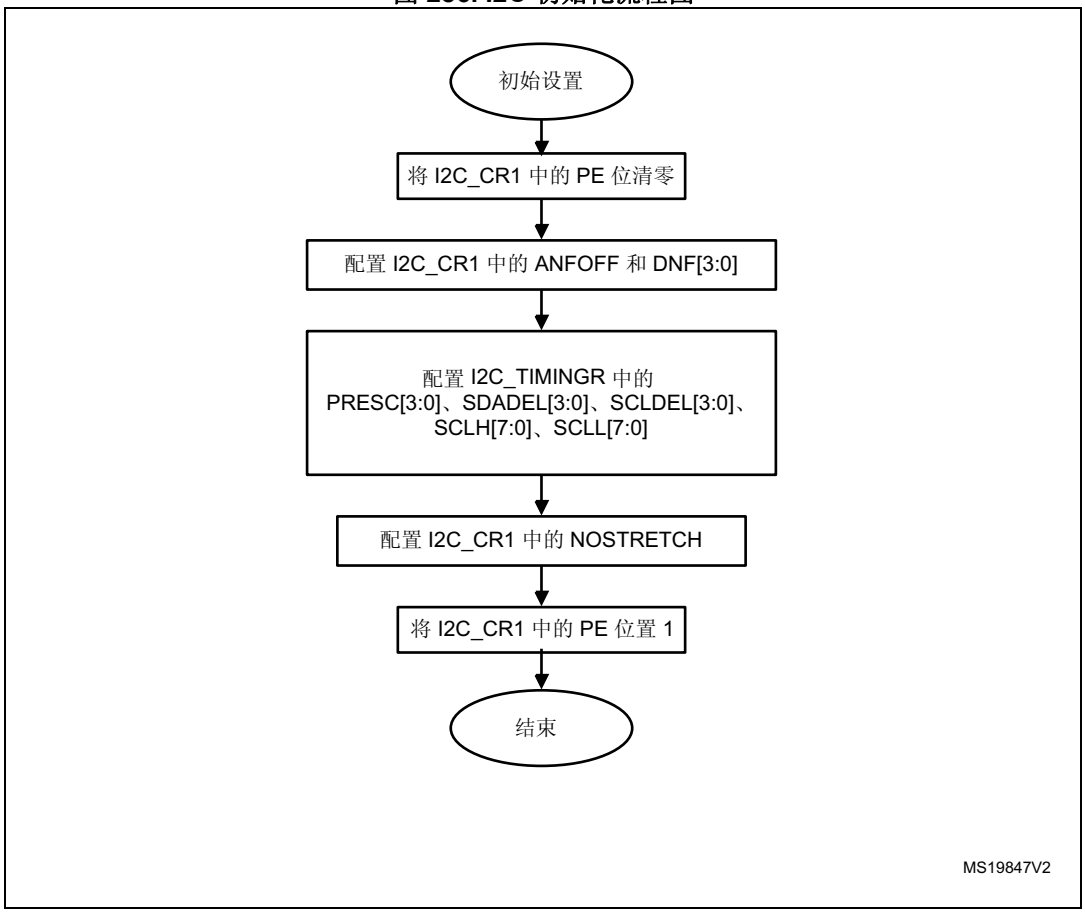
更多详细信息，请参见 *I2C 主模式初始化*一节。

小心： 使能 I2C 后，不允许更改时序配置。

此外，还必须在使能从设备前，对 NOSTRETCH 进行配置。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 286. I2C 初始化流程图



30.4.5 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。

下面列出了受影响的寄存器位：

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 3 个 APB 时钟周期，才能成功执行软件复位。使用以下软件写序列可确保这一点：- 写入 PE=0 - 检查 PE=0 - 写入 PE=1

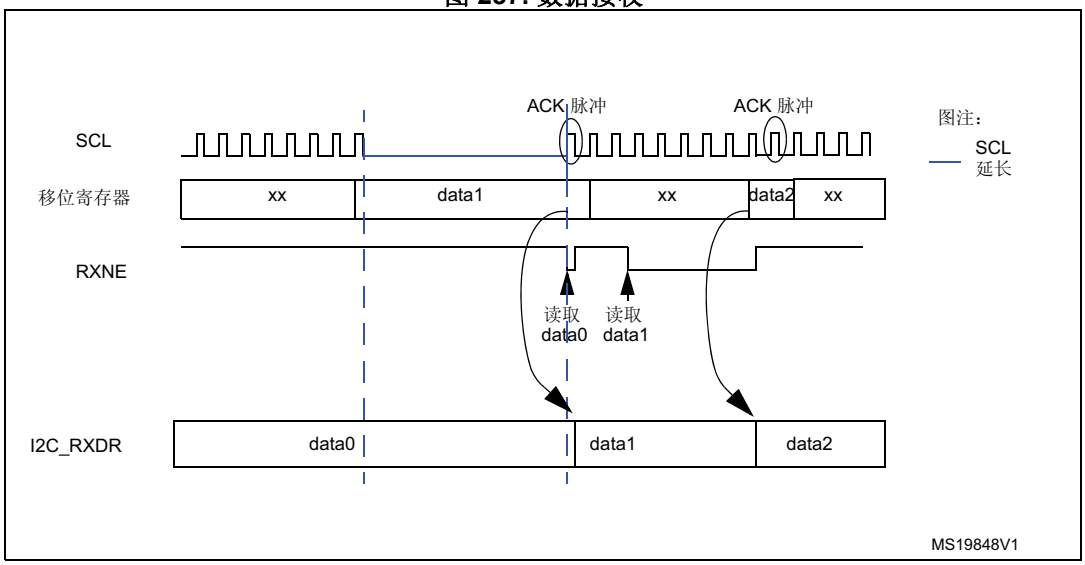
30.4.6 数据传输

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 8 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 8 个和第 9 个 SCL 脉冲之间（应答脉冲之前）插入一段延长的时间。

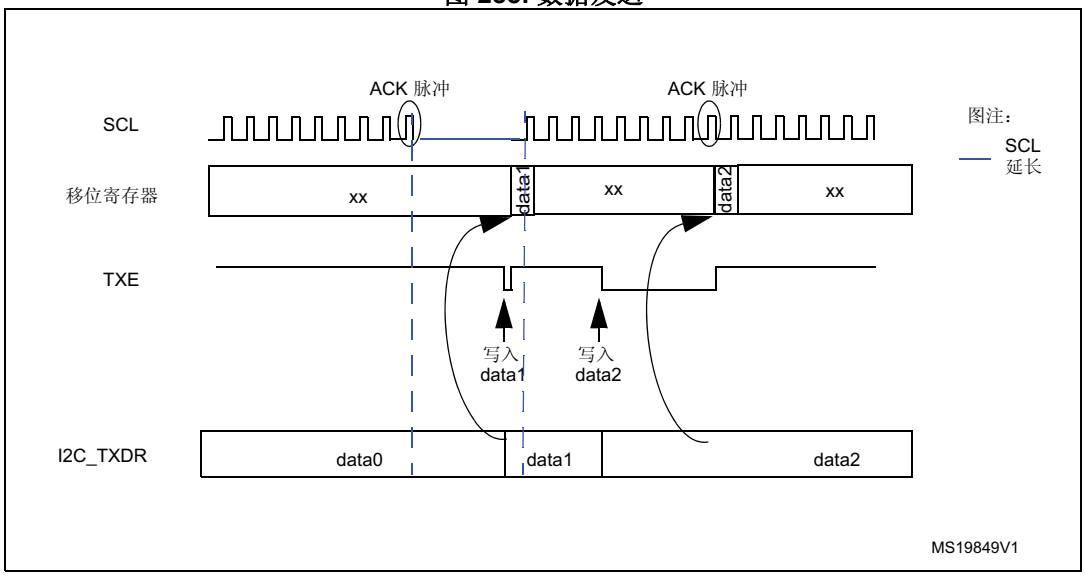
图 287. 数据接收



发送

如果 I2C_TXDR 寄存器不为空 (TXE=0)，则其内容会在第 9 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 9 个 SCL 脉冲后进行延长。

图 288. 数据发送



硬件传输管理

I2C 在硬件中内置了字节计数器，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，但可以通过软件来使能，方法是将 I2C_CR2 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] 位字段中进行编程。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。当往 NBYTES 写入一个非零值时，TCR 由软件清零。

在往 NBYTE 中设置最后一次传输的字节数前，必须把 RELOAD 位清零。

当主模式下 RELOAD=0 时，可在以下 2 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] 位字段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] 位字段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。当软件把 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 155. I2C 配置表

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

30.4.7 I2C 从模式

I2C 从模式初始化

要在从模式下工作，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 2 个从地址 OA2。将 I2C_OAR2 寄存器的 OA2MSK[2:0] 位置 1 最多可屏蔽 7 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，如果从器件使能多个地址，则必须读取 I2C_ISR 寄存器中的 ADDCODE[6:0] 位，以检查哪个地址匹配。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 ADRR 标志时，将释放该时钟延展。
- 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。往 I2C_TXDR 寄存器中写入数据时，将释放该时钟延展。
- 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，将释放该时钟延展。
- 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] 字段写入一个非零值以将 TCR 清零时，将释放该时钟延展。
- 在 SCL 下降沿检测之后，I2C 会延长 SCL 的低电平时间（不超过 $[(SDADEL+SCLDEL+1) \times (PRESC+1) + 1] \times t_{I2CCLK}$）。

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，则应检查 OVR 状态，甚至对于待发送的第一个数据。
- 接收时，必须在下一个数据字节的第 9 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 NBYTE 重载为 0x1。接收到字节后，TCR 位将置 1，从而延长 SCL 信号的第 8 个和第 9 个脉冲之间的低电平时间。可以从 I2C_RXDR 寄存器中读取数据，然后决定是否配置 I2C_CR2 寄存器中的 ACK 位以进行应答。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。

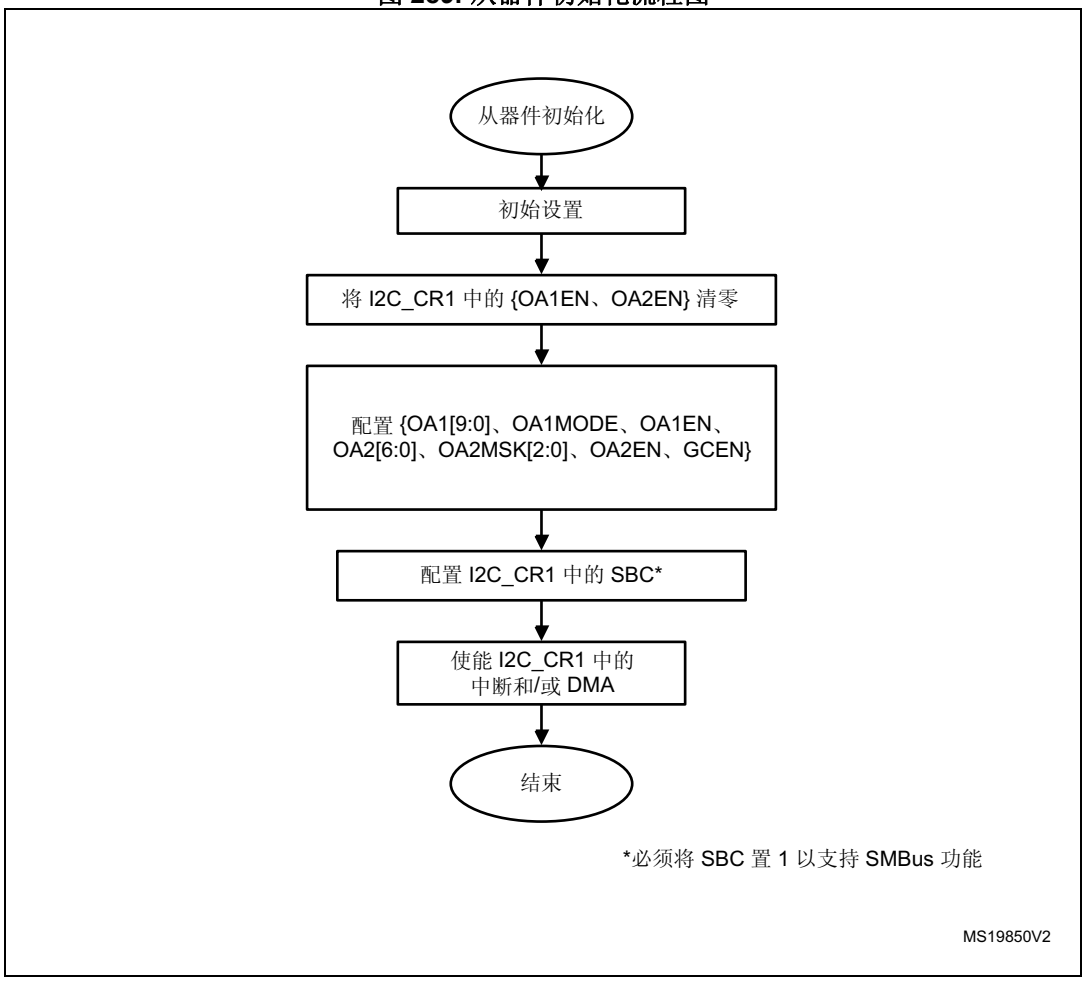
NBYTES 可加载大于 0x1 的值，在这种情况下，接收流在 NBYTES 个数据接收期间是连续的。

注： *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 289. 从器件初始化流程图



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，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 NBYTE 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，因此无法通过在 ADDR 子程序中刷新 I2C_TXDR 寄存器的内容来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，以确保在第一次数据传输之后地址应答之前执行这些操作。

  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），则必须将 TXE 位和 TXIS 位均置 1，以便生成 TXIS 事件。

图 290. I2C 从发送器的传输序列流程图 (NOSTRETCH=0)

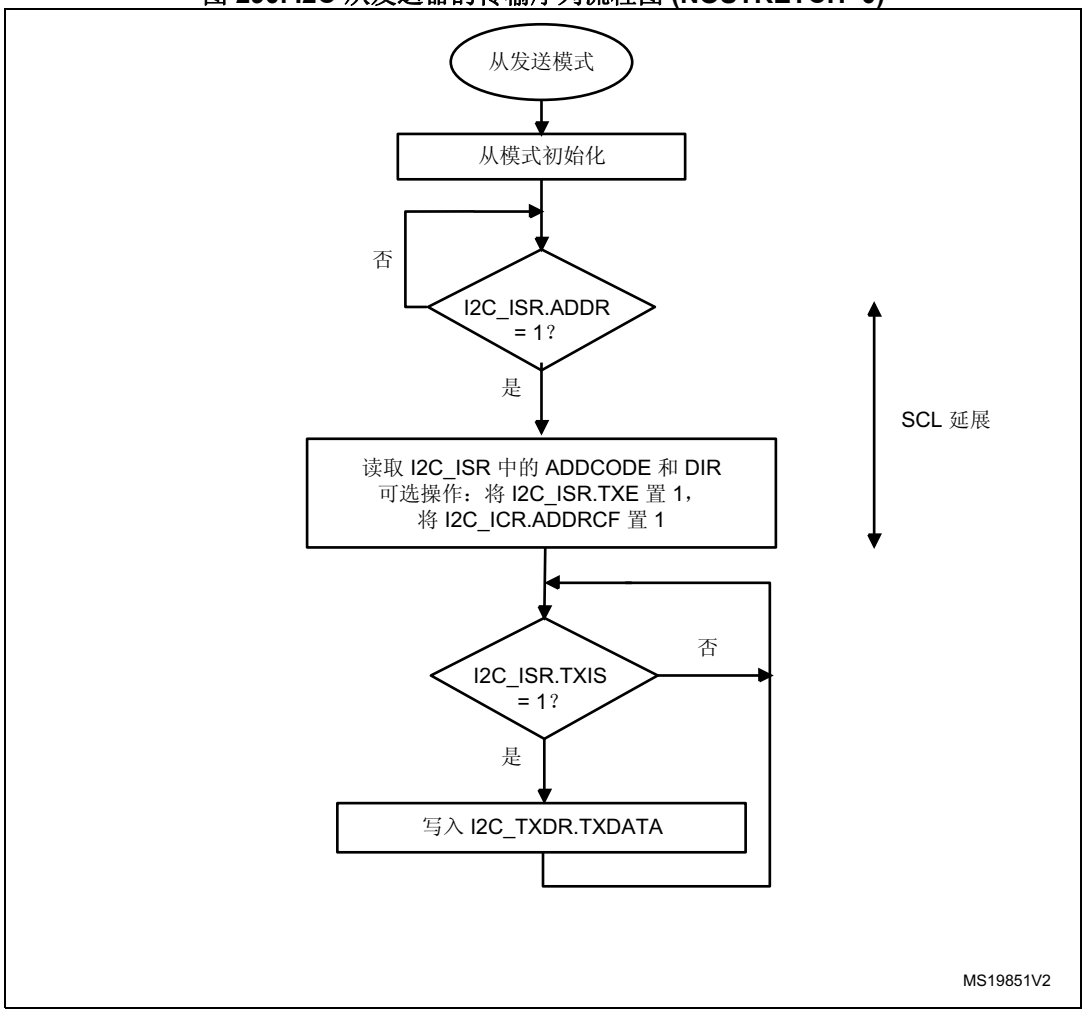


图 291. I2C 从发送器的传输序列流程图 (NOSTRETCH=1)

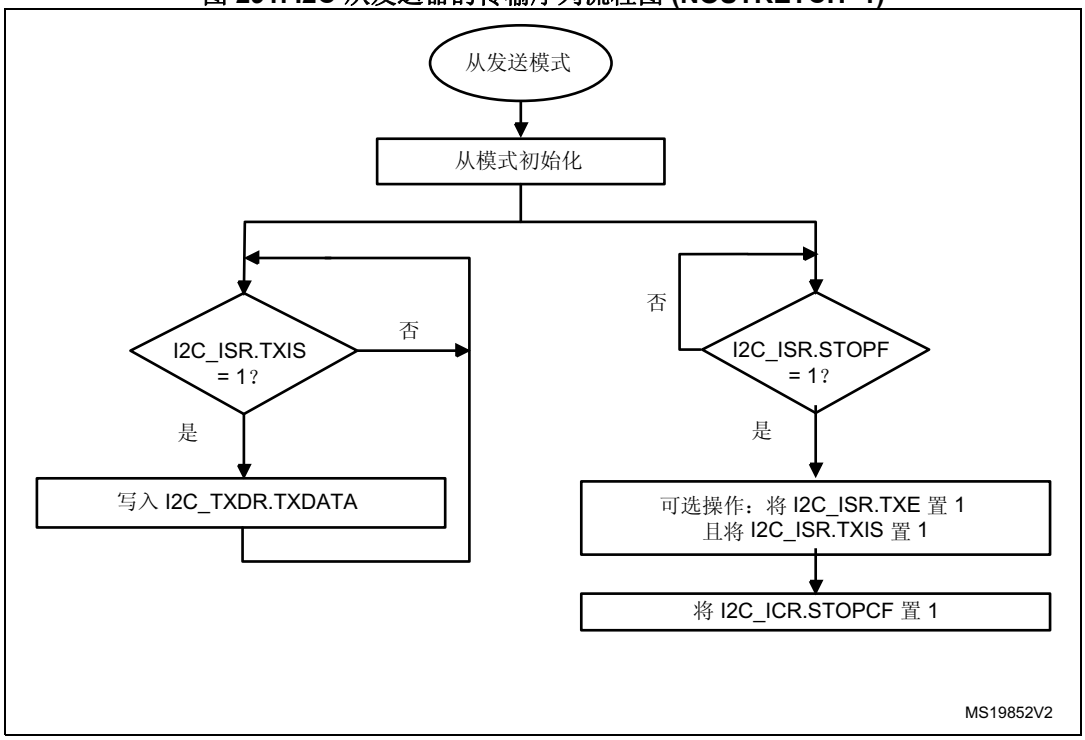
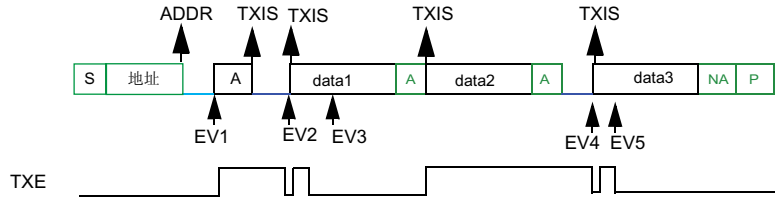


图 292. I2C 从发送器的传输总线图

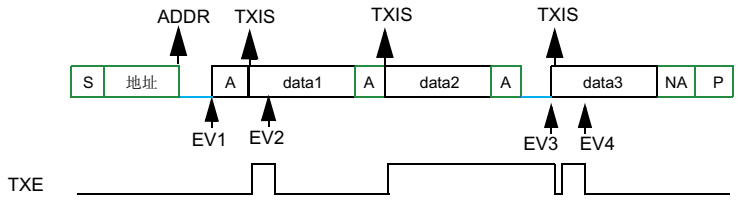
示例：I2C 从器件发送 3 个字节，刷新第 1 个数据，NOSTRETCH=0：



图注：
发送
接收
SCL 延长

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）

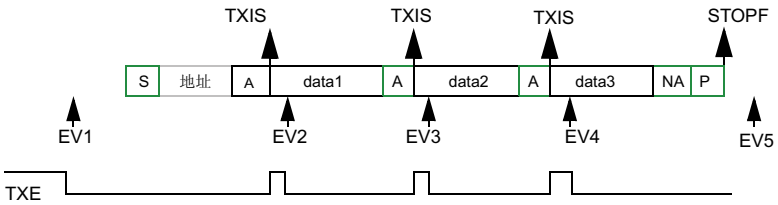
示例：I2C 从器件发送 3 个字节，不刷新第 1 个数据，NOSTRETCH=0：



图注：
发送
接收
SCL 延长

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 从器件发送 3 个字节，NOSTRETCH=1：



图注：
发送
接收
SCL 延长

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：（可选操作：将 TXE 和 TXIS 置 1），将 STOPCF 置 1

MS19853V1

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 293. 从接收器的传输序列流程图 (NOSTRETCH=0)

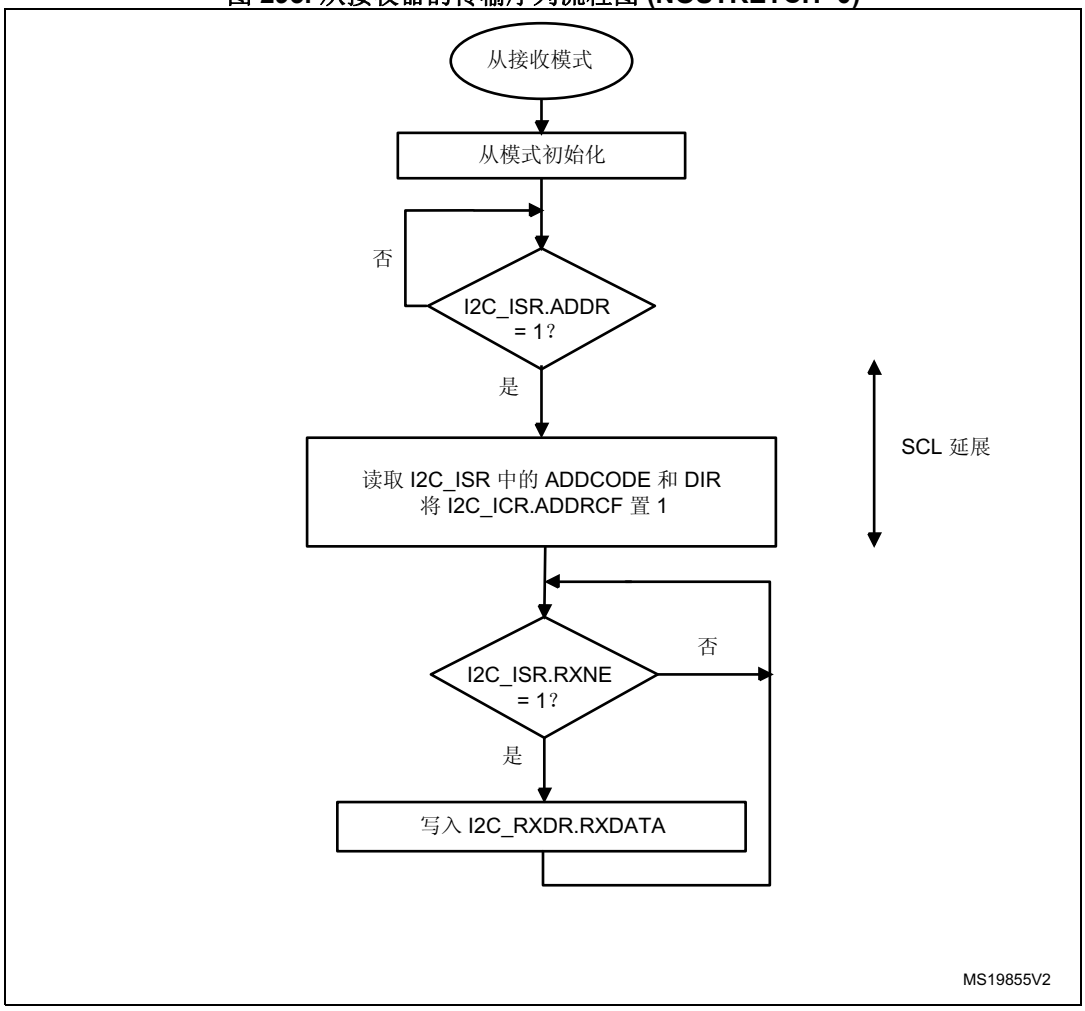


图 294. 从接收器的传输序列流程图 (NOSTRETCH=1)

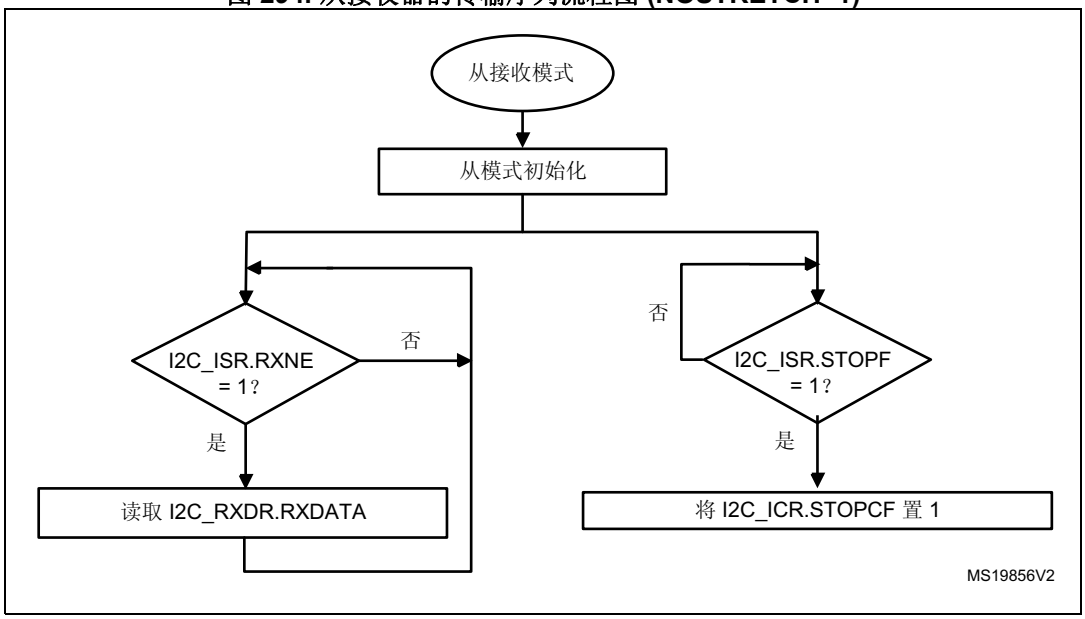
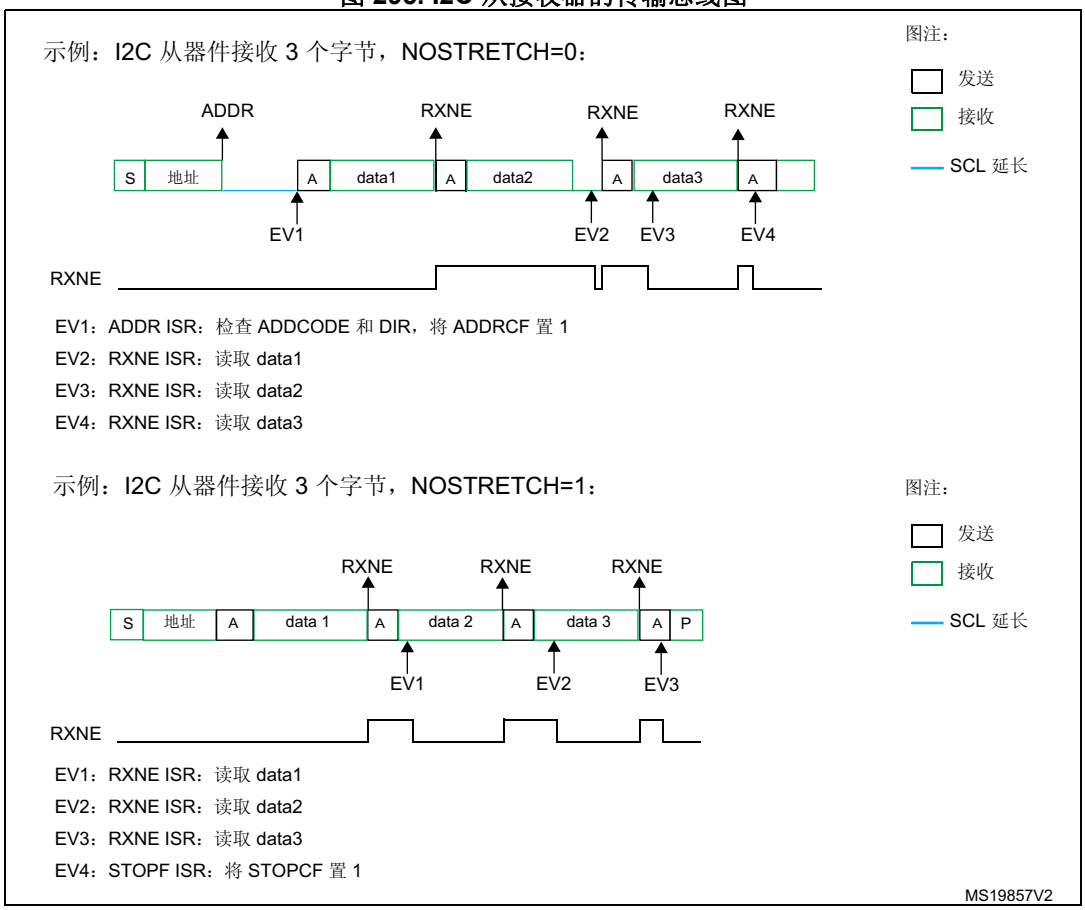


图 295. I2C 从接收器的传输总线图



30.4.8 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。

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：

$$t_{SCL} = t_{SYNC1} + t_{SYNC2} + \{[(SCLH+1) + (SCLL+1)] \times (PRESC+1) \times t_{I2CCLK}\}$$

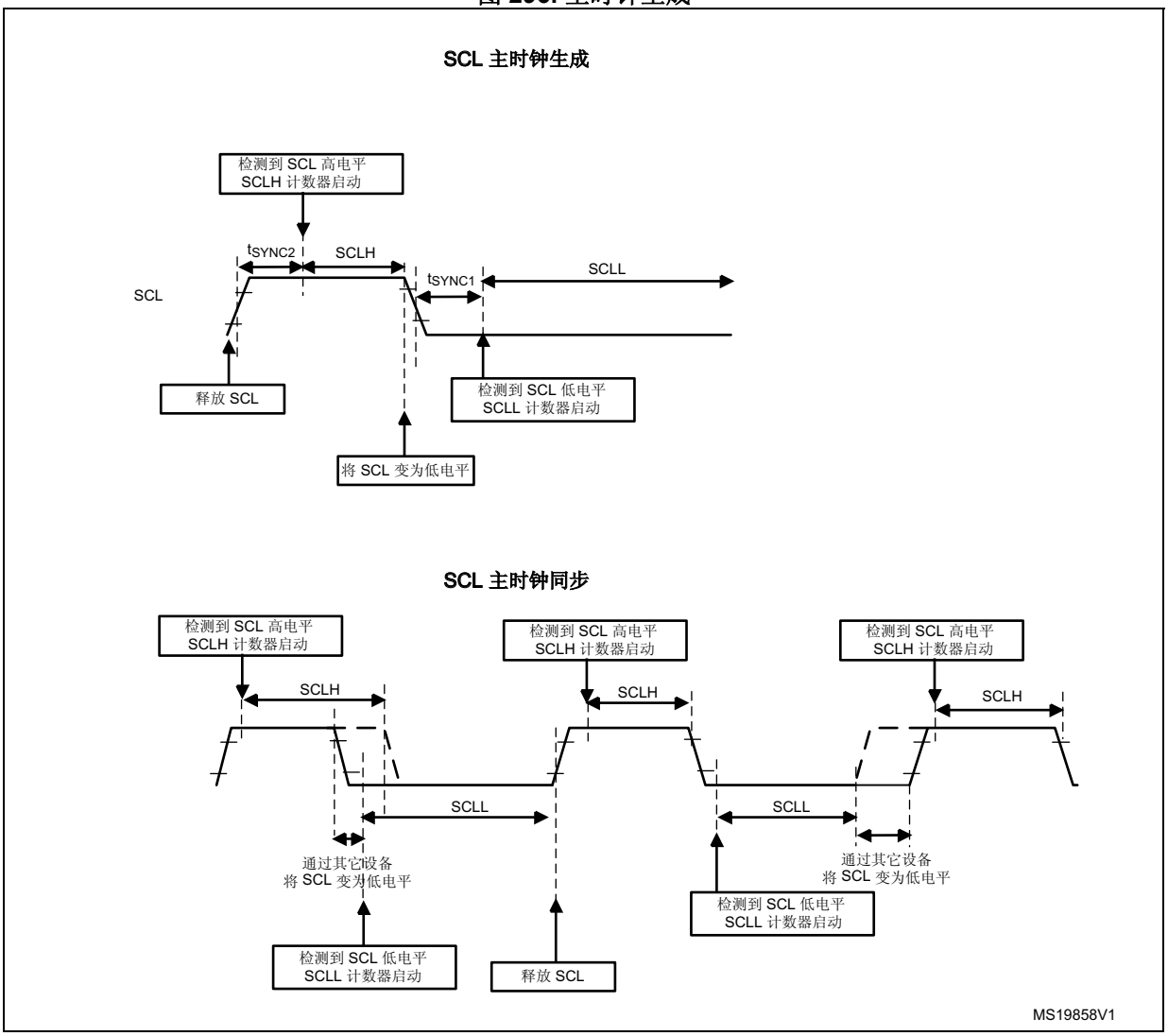
t_{SYNC1} 的持续时间取决于以下参数：

- 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 I2CCLK 时钟建立同步而产生的延时（2 到 3 个 I2CCLK 周期）

t_{SYNC2} 的持续时间取决于以下参数：

- 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 I2CCLK 时钟建立同步而产生的延时（2 到 3 个 I2CCLK 周期）

图 296. 主时钟生成



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 156. 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	-	µs
$t_{SU:STO}$	停止条件的建立时间	4.0	-	0.6		0.26	-	4.0	-	µs

表 156. I^2C-SMBUS 规范时钟时序（续）

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_{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	-	µs
t_{LOW}	SCL 时钟的低电平周期	4.7	-	1.3		0.5	-	4.7	-	µs
t_{HIGH}	SCL 时钟的高电平周期	4.0	-	0.6		0.26	-	4.0	50	µs
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	ns

注： *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 I2CCLK 频率的示例，请参见*第 30.4.9 节：I2C_TIMINGR 寄存器配置示例*。

主模式通信初始化（地址阶段）

要发起通信，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，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。

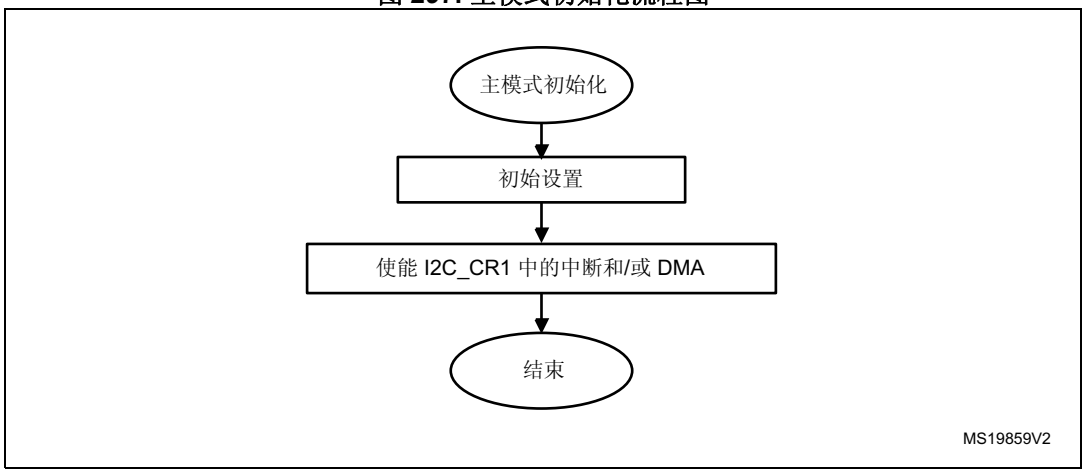
之后，当主器件检测到总线空闲 (BUSY = 0) 时，它会在经过 t_{BUF} 的延时后自动发送起始位，随后发出从器件地址。

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： *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。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 位将在 ADDRCF 位置 1 时清零。*

注： *该步骤同样适用于重复起始位。在这种情况下，BUSY=1。*

图 297. 主模式初始化流程图



主接收器寻址 10 位地址从器件的初始化过程

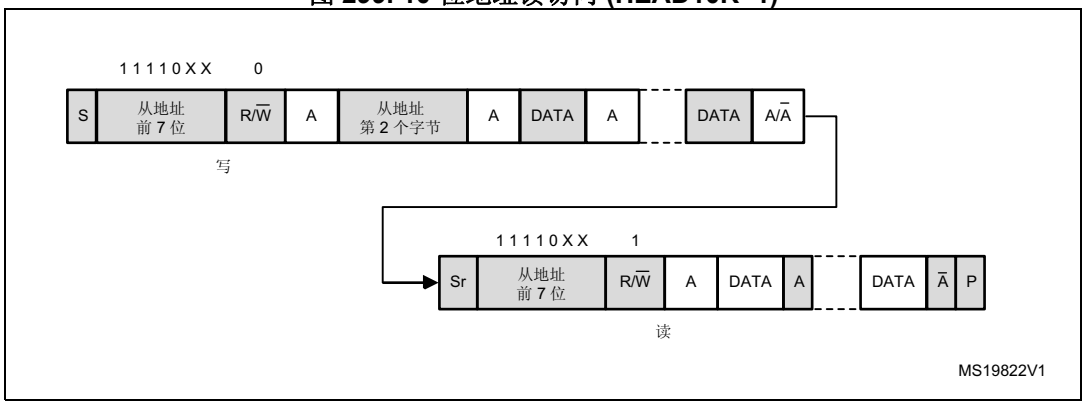
- 如果从地址为 10 位格式，可选择将 I2C_CR2 寄存器中的 HEAD10R 位清零来发送完整的读序列。在这种情况下，主器件会在 START 位置 1 后自动发送以下完整序列：（重复）起始位 + 带写方向的从器件 10 位地址头字节 + 从器件地址第 2 个字节 + 重复起始位 + 带读方向的从器件 10 位地址头字节。

图 298. 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 + 从地址 10 位头读取。

图 299. 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 9 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 线的低电平将被延展，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 线的低电平将被延展，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 300. I2C 主发送器的传输序列流程图（N≤255 字节）

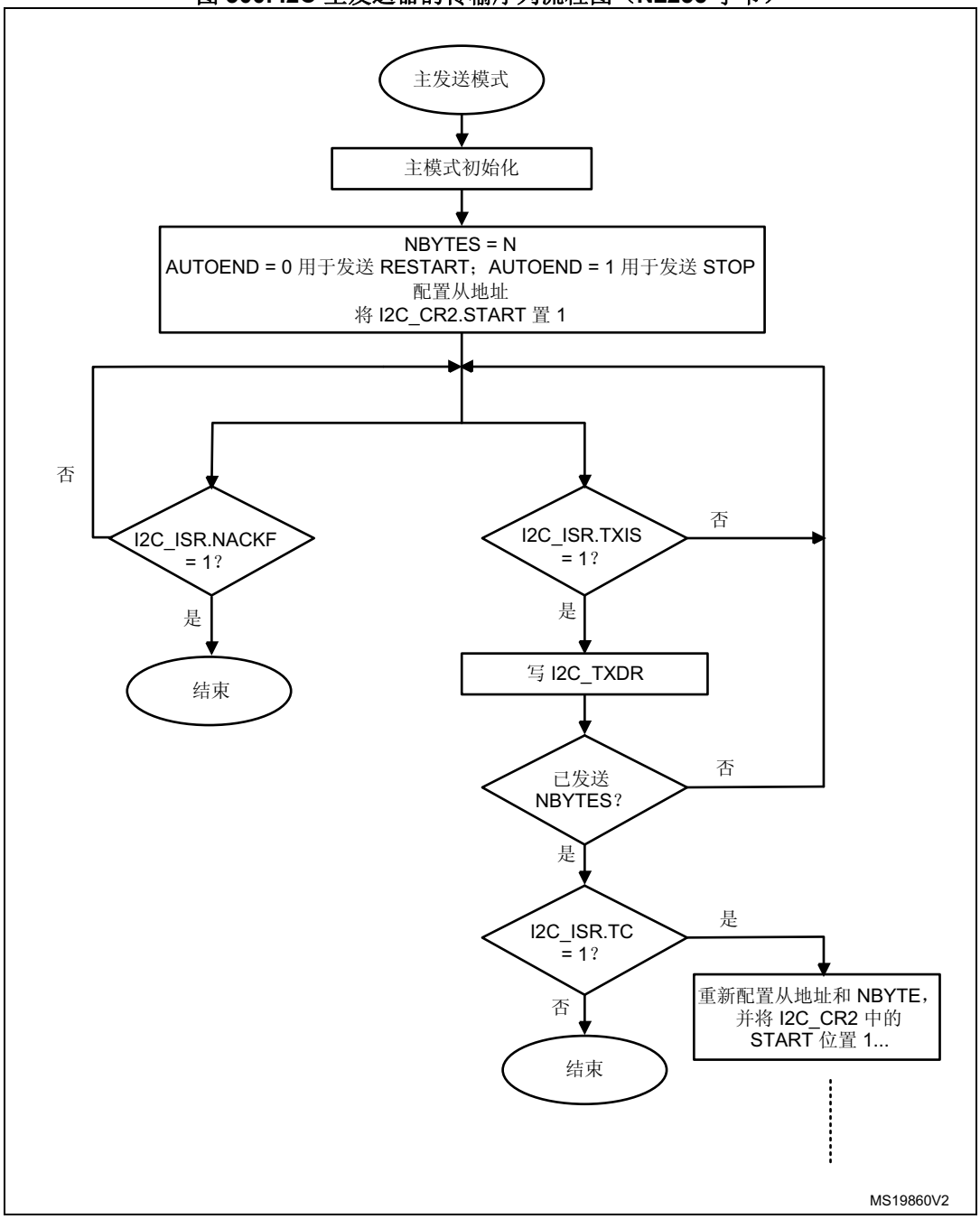


图 301. I2C 主发送器的传输序列流程图（N>255 字节）

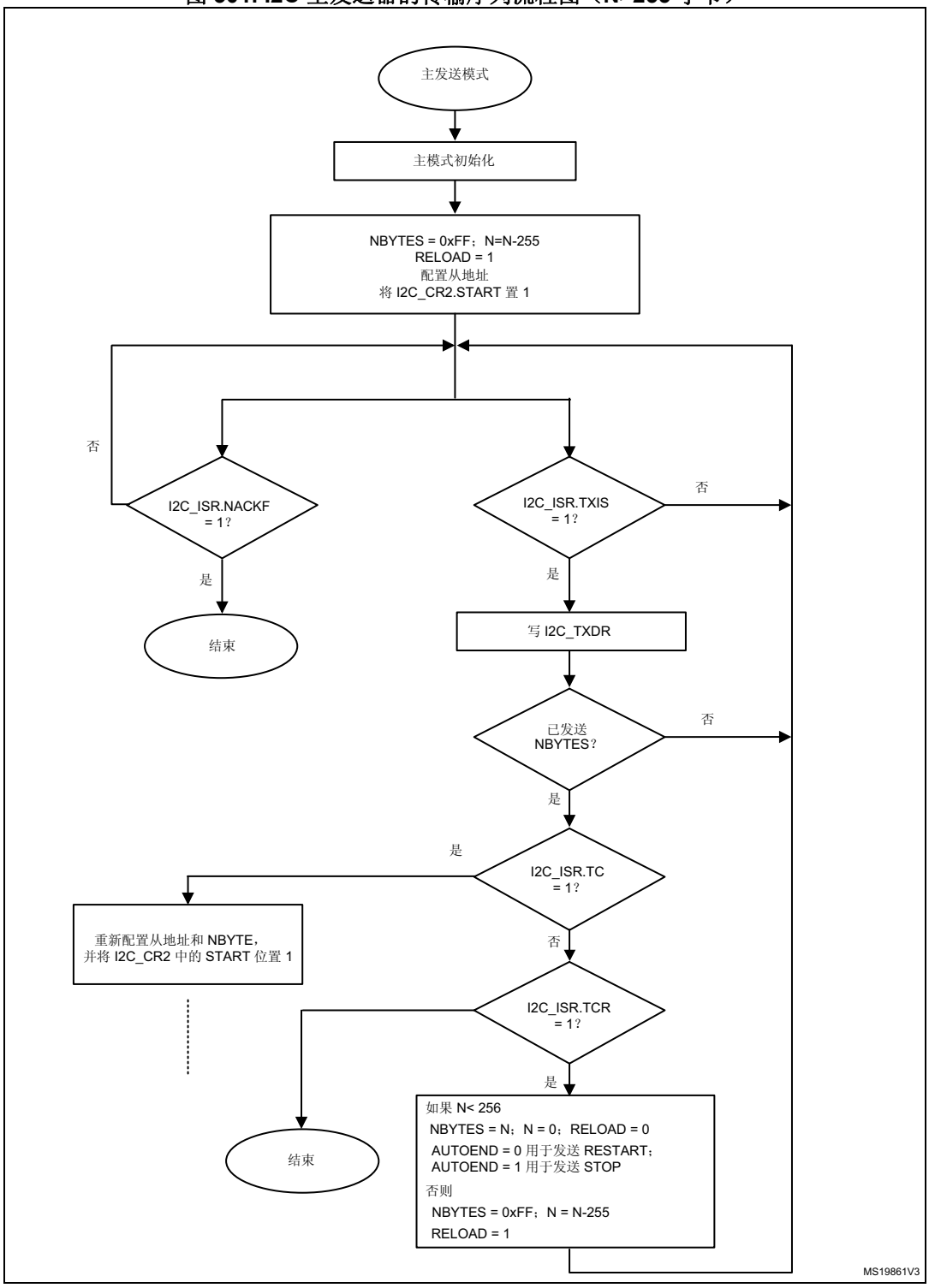
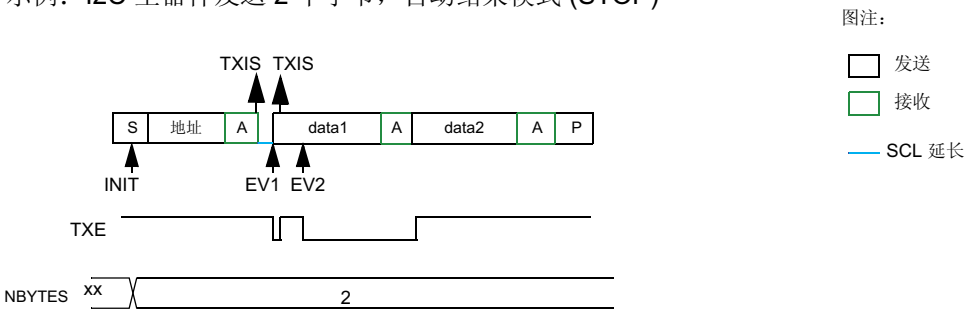


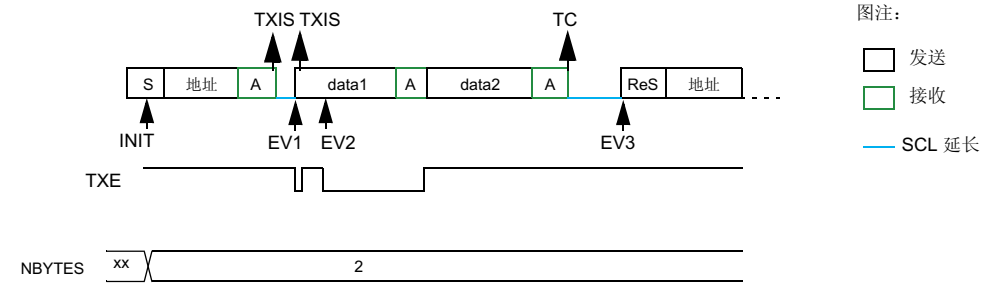
图 302. I2C 主发送器的传输总线图

示例：I2C 主器件发送 2 个字节，自动结束模式 (STOP)



INIT：设置从地址，设置 NBYTES = 2，AUTOEND=1，将 START 置 1
EV1：TXIS ISR：写入 data1
EV2：TXIS ISR：写入 data2

示例：I2C 主器件发送 2 个字节，软件结束模式 (RESTART)



INIT：设置从地址，设置 NBYTES = 2，AUTOEND=0，将 START 置 1
EV1：TXIS ISR：写入 data1
EV2：TXIS ISR：写入 data2
EV3：TC ISR：设置从地址，设置 NBYTES = N，将 START 置 1

MS19862V1

主接收器

读传输时，在接收到每个字节（即第 8 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 线的低电平将被延展，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 线的低电平将被延展，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 303. I2C 主接收器的传输序列流程图（N≤255 字节）

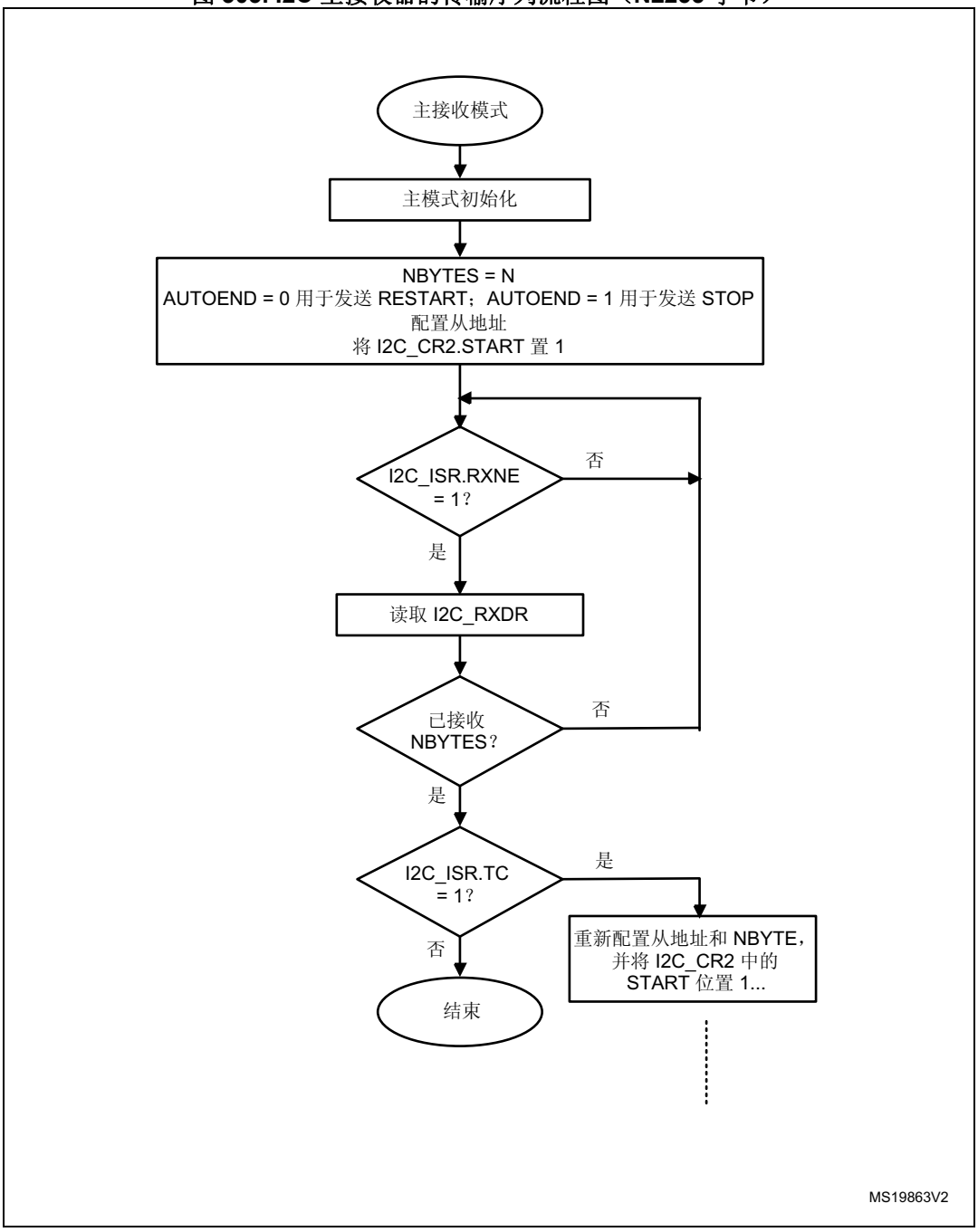


图 304. I2C 主接收器的传输序列流程图（N>255 字节）

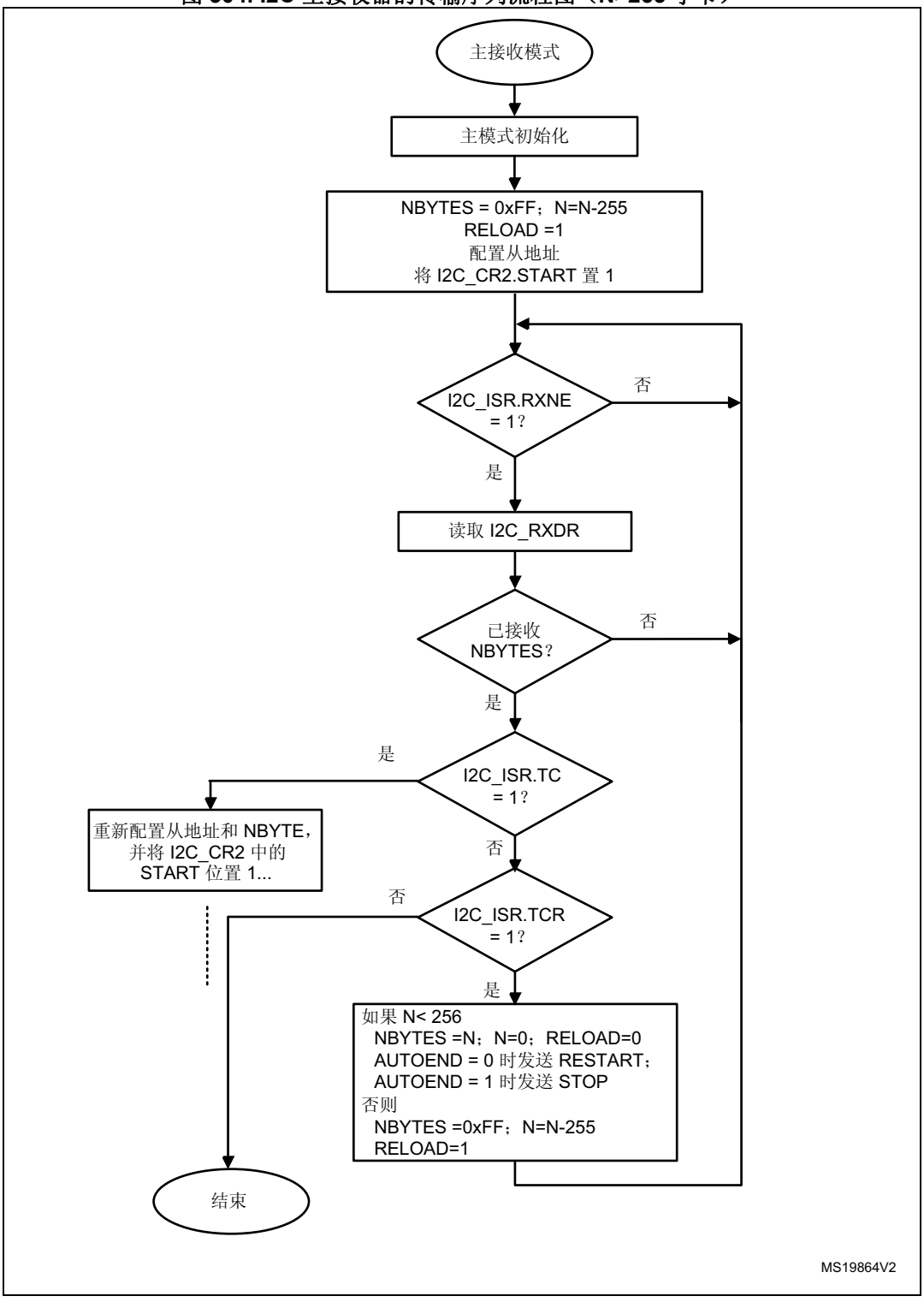
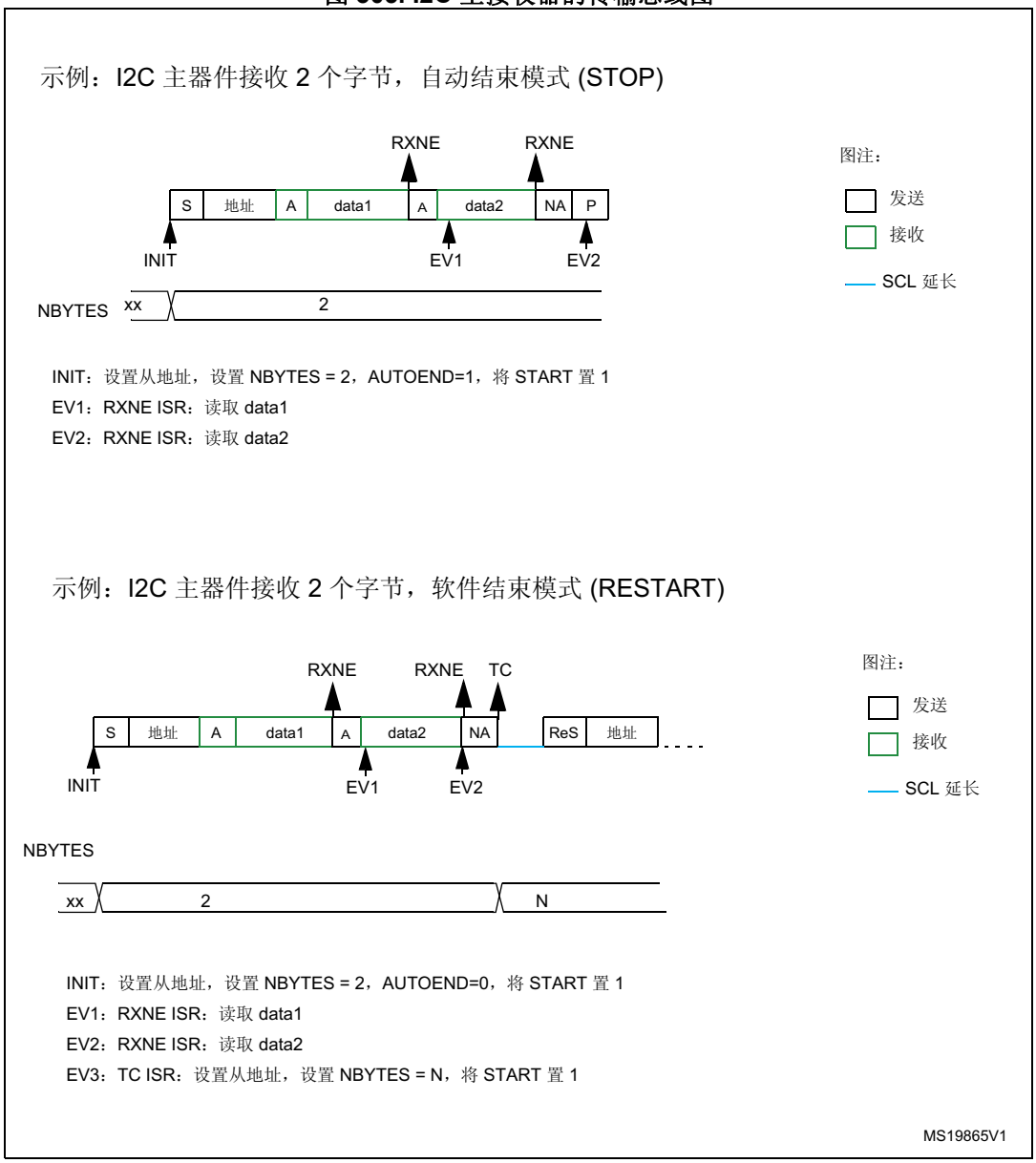


图 305. I2C 主接收器的传输总线图



30.4.9 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 I²C 规范的时序。要获取更准确的配置值，请参见应用笔记：*《I²C 时序配置工具》* (AN4235) 及相应软件 STSW-STM32126。

30.4.10 SMBus 特性

仅当支持 SMBus 功能时，才涉及本节内容。请参见*第 30.3 节：I2C 特性实现*。

前言

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 I²C 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 规范第 2.0 版兼容 (*http://smbus.org*)。

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 - 从器件功能，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。

SMBUS 以 I²C 规范第 2.1 版为基础。

总线协议

任何给定器件都有十一种可用命令协议。器件既可以在这十一种协议中任选其一，也可以使用全部十一种协议进行通信。这十一种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。这些协议应通过用户软件实施。

有关这些协议的详细信息，请参见 SMBus 规范第 2.0 版 (*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 命令应通过用户软件实现。

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 规范第 2.0 版 (*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否定应答。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*第 857 页上的从器件字节控制模式*一节。

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 规范第 2.0 版中定义的 3 个超时。

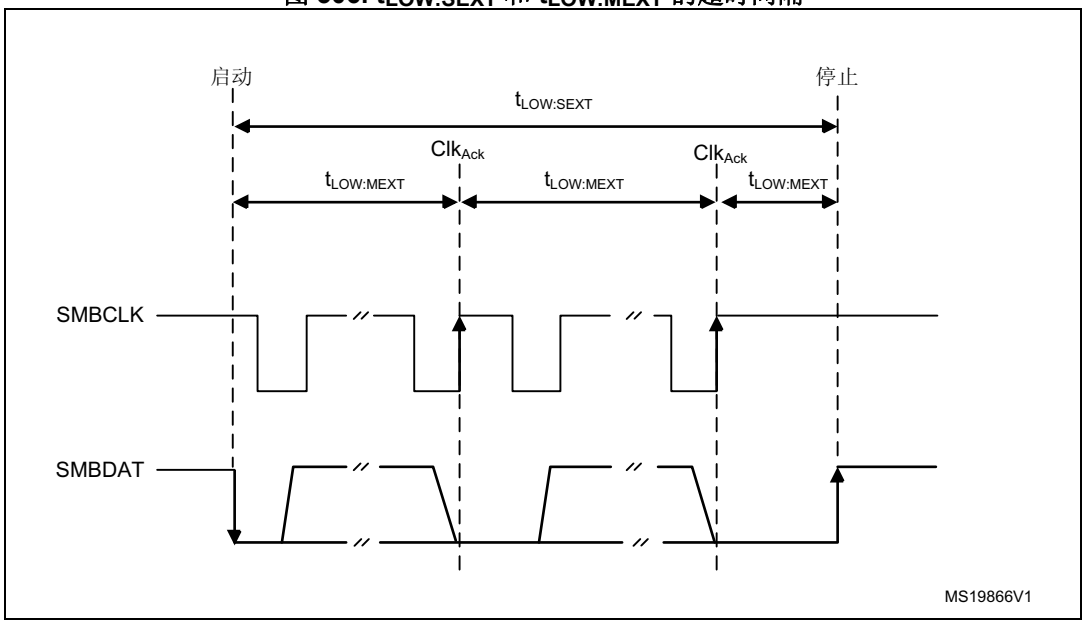
表 157. SMBus 超时规范

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	ms
$t_{LOW:MEXT}$(2)	累积时钟低电平延长时间（主器件）	-	10	ms

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展的时间。其它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，测量该参数时该器件应该是全速主器件寻址的唯一器件。

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）内时钟信号可延展的时间。从器件或其它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 306. $t_{LOW:SEXT}$ 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已达 t_{IDLE}（超过 $t_{HIGH,MAX}$），则认为总线空闲（请参见*表 156：I2C-SMBUS 规范时钟时序*）。

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。

30.4.11 SMBus 初始化

仅当支持 SMBus 功能时，才涉及本节内容。请参见*第 30.3 节：I2C 特性实现*。

除了 I2C 初始化之外，还必须进行一些其它的特定初始化，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否定应答。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*第 857 页上的从器件字节控制模式*。

特定地址（从模式）

必要时应使能特定的 SMBus 地址。更多详细信息，请参见*第 878 页上的总线空闲检测*。

- 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 158. 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 规范第 2.0 版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 159：不同 I2CCLK 频率下的 TIMEOUTA 设置示例（最大 $t_{TIMEOUT}$ = 25 ms）*。

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 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，因此可以为这两个参数选择相同的值。

  然后，通过将 I2C_TIMEOUTR 寄存器中的 TEXTEN 位置 1 来使能定时器。

  如果 SMBus 外设延展 SCL 的累积时间超过 (TIMEOUTB+1) x 2048 x t_{I2CCLK}，并且达到*第 878 页上的总线空闲检测*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 160：不同 I2CCLK 频率下的 TIMEOUTB 设置示例*。

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] 字段编程为定时器重载值，以获取 t_{IDLE} 参数。必须将 TIDLE 位配置为“1”，以检测 SCL 和 SDA 高电平超时。

然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 位置 1 来使能定时器。

如果 SCL 和 SDA 线的高电平持续时间超过 (TIMEOUTA+1) x 4 x t_{I2CCLK}，I2C_ISR 寄存器中的 TIMEOUT 标志将置 1。

请参见*表 161：不同 I2CCLK 频率下的 TIMEOUTA 设置示例（最大 t_{IDLE} = 50 µs）*。

小心： TIMEOUTEN 置 1 时，不允许更改 TIMEOUTA 和 TIDLE 配置。

30.4.12 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 30.3 节：I2C 特性实现*。

- 将 $t_{TIMEOUT}$ 的最大持续时间配置为 25 ms：

表 159. 不同 I2CCLK 频率下的 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 160. 不同 I2CCLK 频率下的 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 161. 不同 I2CCLK 频率下的 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

30.4.13 SMBus 从模式

仅当支持 SMBus 功能时，才涉及本节内容。请参见*第 30.3 节：I2C 特性实现*。

除了 I2C 从模式传输管理（请参见*第 30.4.7 节：I2C 从模式*）之外，还提供了一些额外的软件流程图来支持 SMBus。

SMBus 从发送器

在 SMBus 模式下使用 IP 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 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 307. SMBus 从发送器的传输序列流程图（N 字节 + PEC）

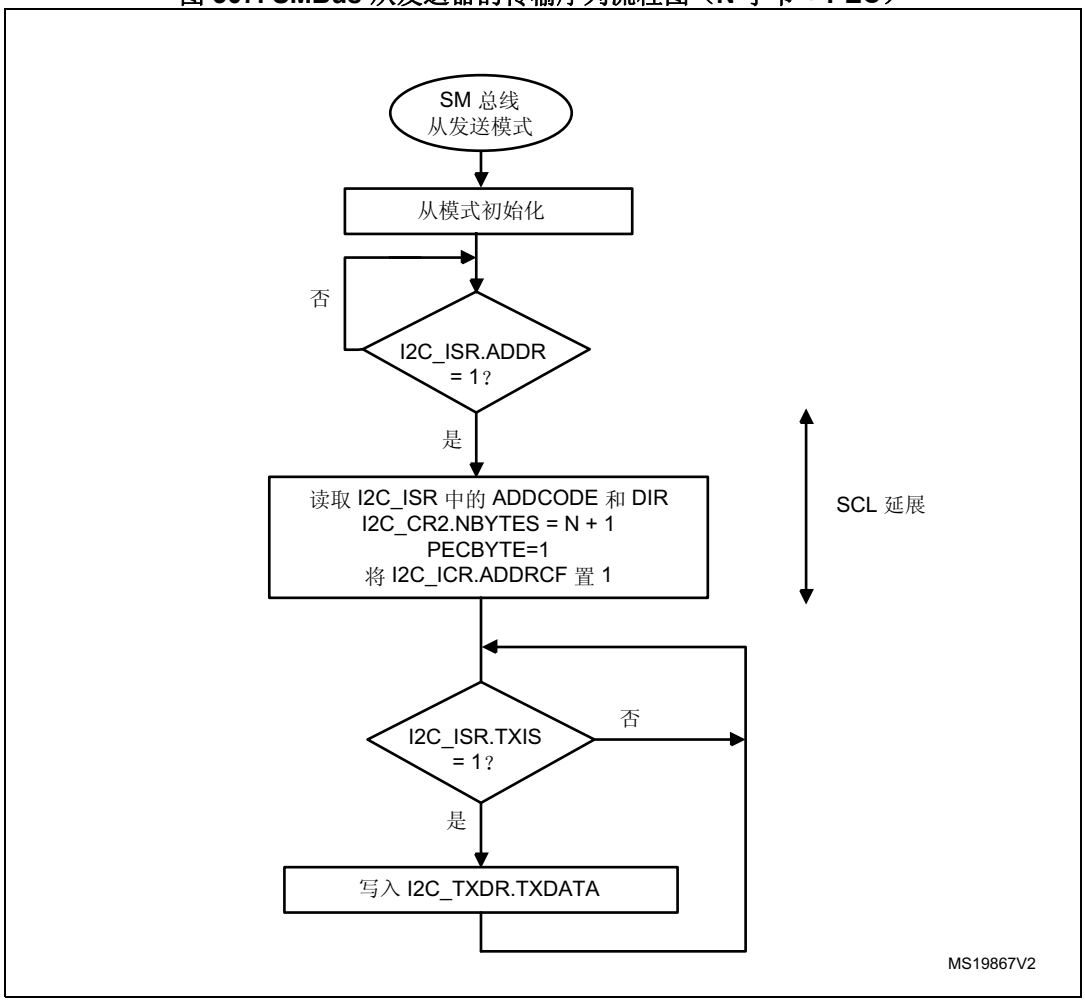
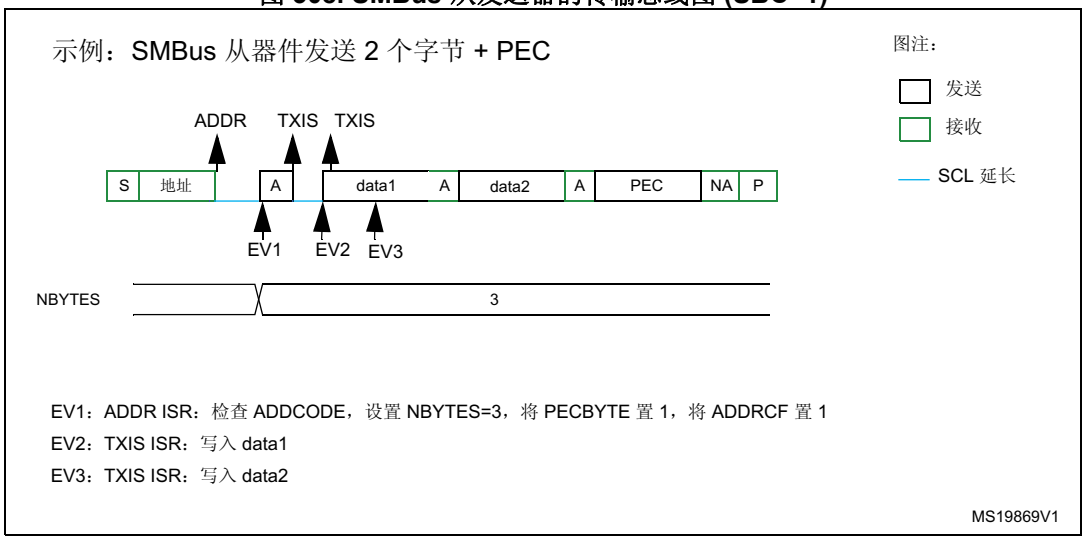


图 308. SMBus 从发送器的传输总线图 (SBC=1)



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*第 857 页上的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，可设置 PECBYTE=1，在同样的写操作下，将 NBYTES 设置为将连续接收的字节数。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 309. SMBus 从接收器的传输序列流程图（N 字节 + PEC）

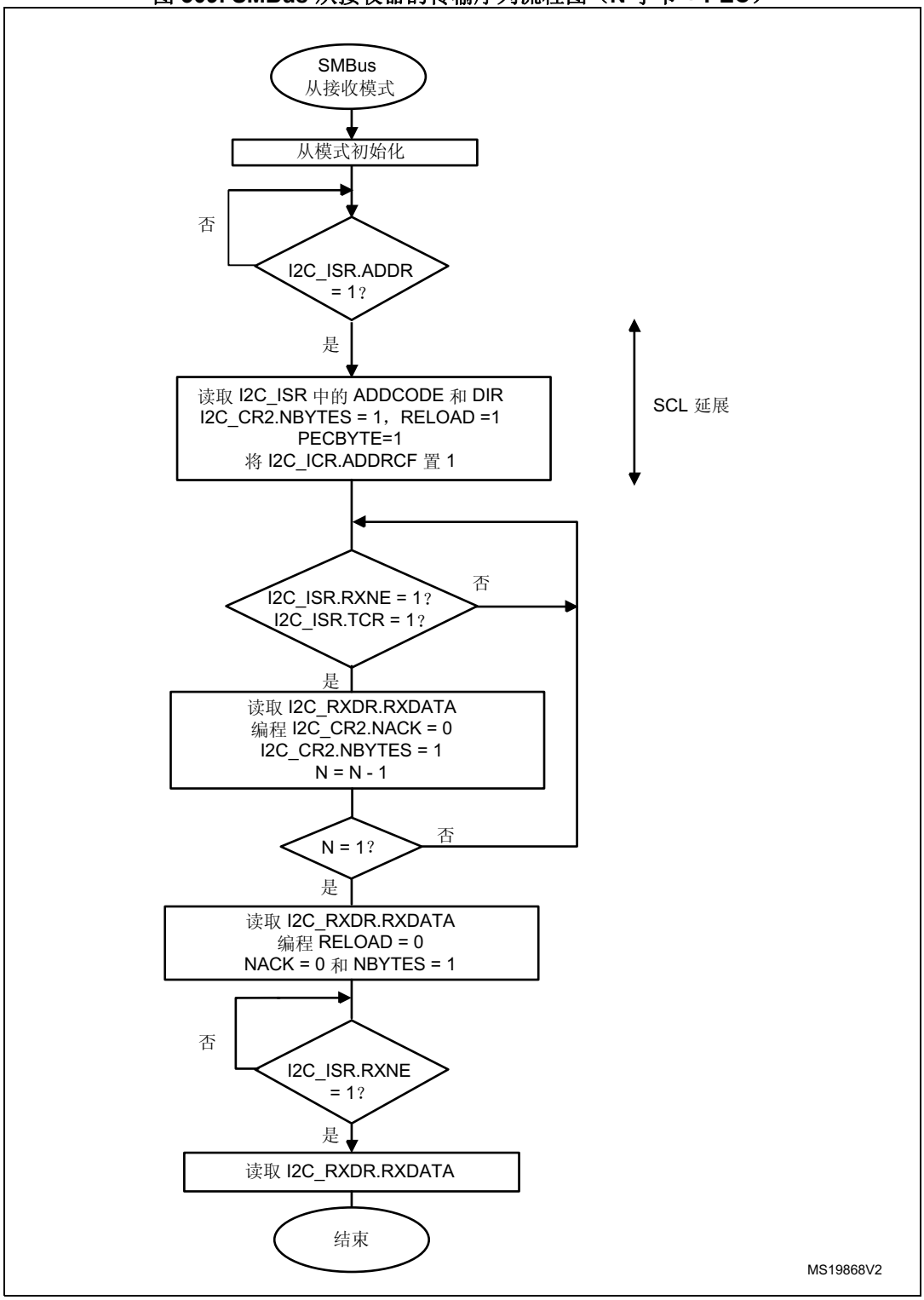
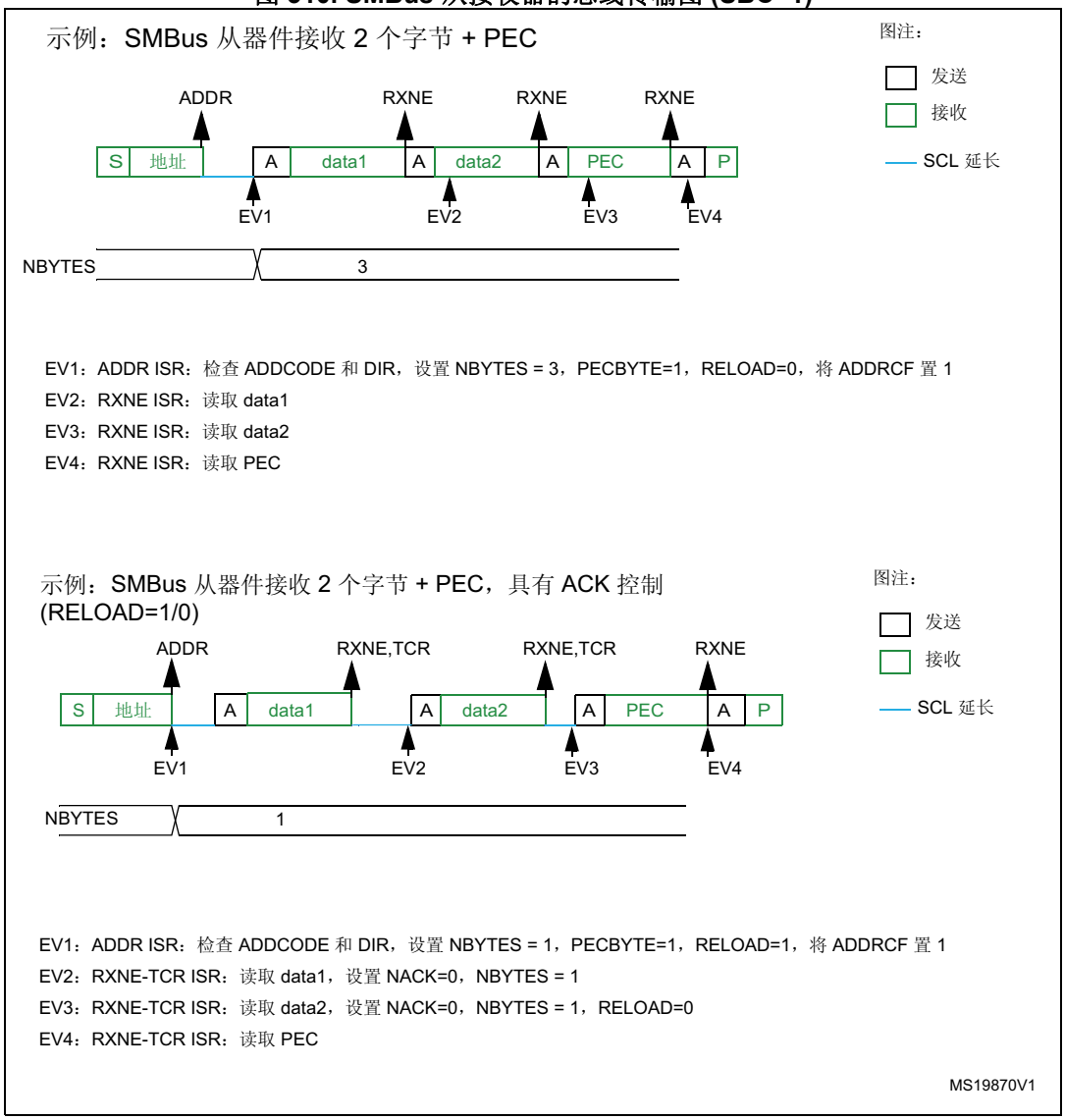


图 310. SMBus 从接收器的总线传输图 (SBC=1)



仅当支持 SMBus 功能时，才涉及本节内容。请参见*第 30.3 节：I2C 特性实现*。

除了 I2C 主模式传输管理（请参见*第 30.4.8 节：I2C 主模式*）之外，还提供了一些额外的软件流程图来支持 SMBus。

SMBus 主发送器

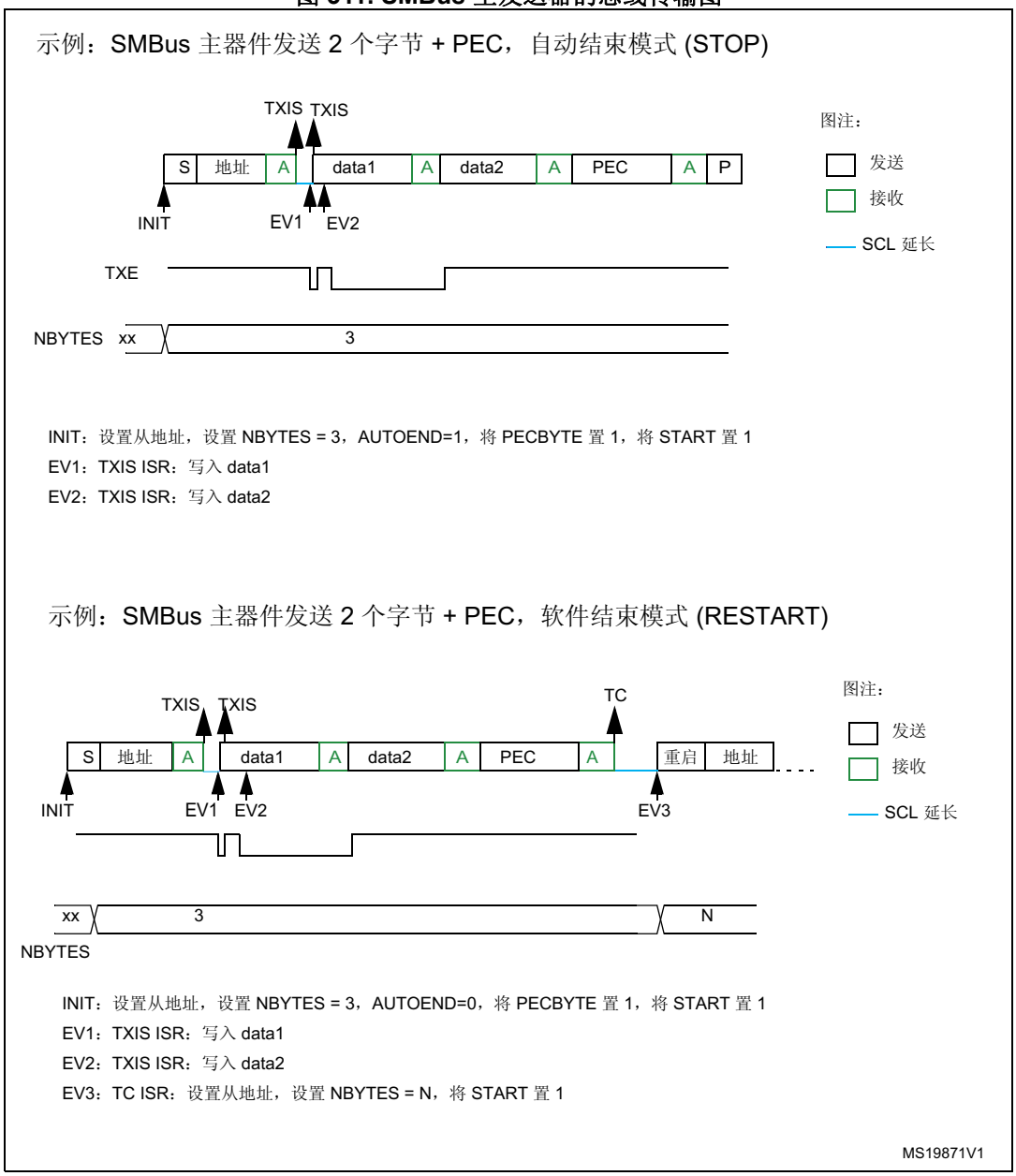
当 SMBus 主器件想要发送 PEC 时，必须在 START 位置 1 前，将 PECBYTE 位置 1 并在 NBYTES[7:0] 字段中设置字节数。在这种情况下，总 TXIS 中断数为 NBYTES-1。因此，如果 PECBYTE 位在 NBYTES=0x1 时置 1，则将自动发送 I2C_PECR 寄存器的内容。

如果 SMBus 主器件想要在 PEC 后发送停止位，则应选择自动结束模式 (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 311. SMBus 主发送器的总线传输图



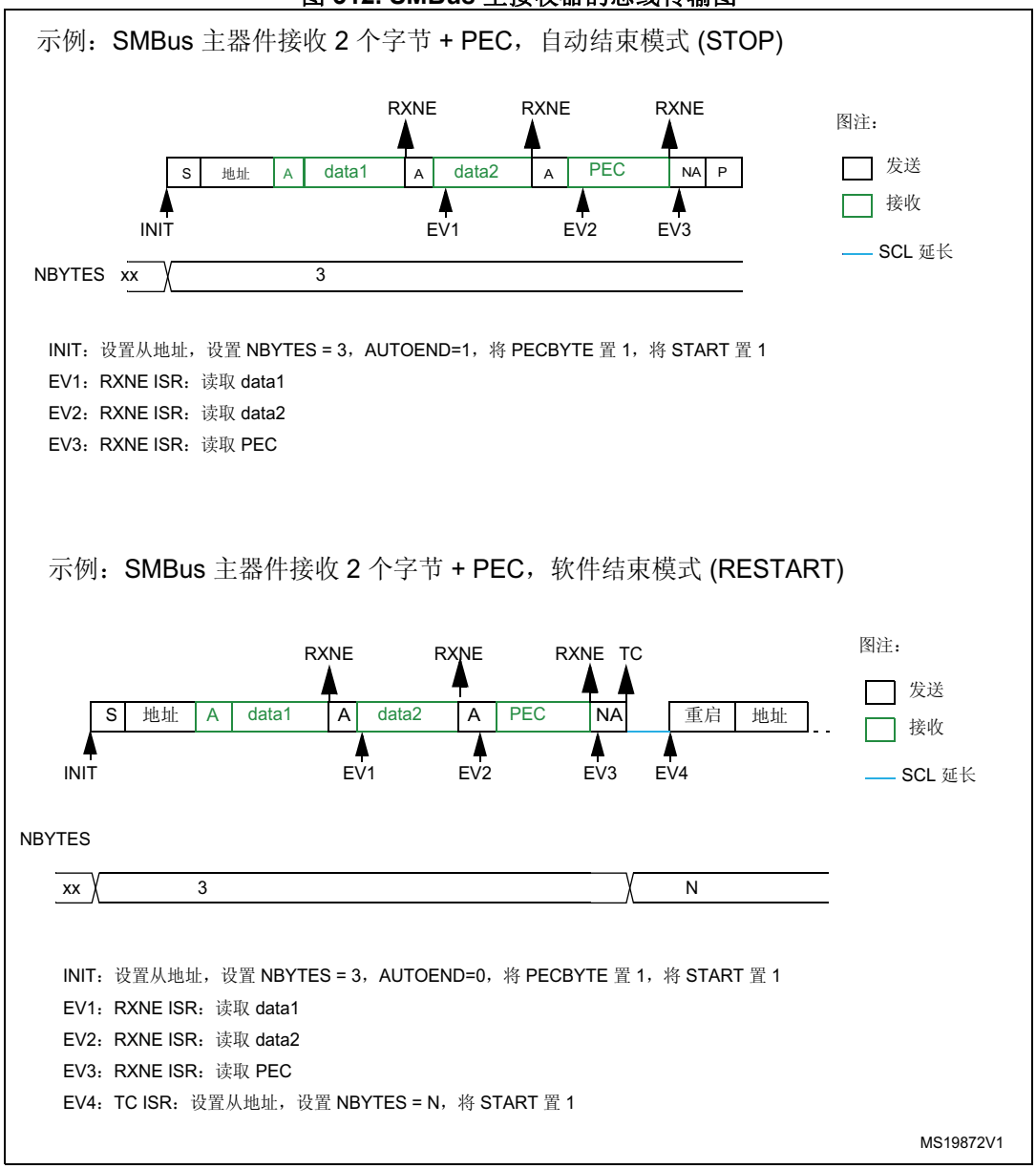
SMBus 主接收器

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 312. SMBus 主接收器的总线传输图



30.4.14 错误条件

以下错误条件可能导致通信失败。

总线错误 (BERR)

当检测到起始位或停止位但不位于第 9N 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：

- 接收过程中接收到一个新字节但尚未读取 RXDR 寄存器。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 且应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。
  - 应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 功能时，才涉及本节内容。请参见*第 30.3 节：I2C 特性实现*。

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 功能时，才涉及本节内容。请参见*第 30.3 节：I2C 特性实现*。

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 功能时，才涉及本节内容。请参见*第 30.3 节：I2C 特性实现*。

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。

30.4.15 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*第 206 页上的第 8 节：直接存储器访问控制器 (DMA)*）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*第 868 页上的主发送器*。
- 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*第 881 页上的SMBus 从发送器*和*第 884 页上的SMBus 主发送器*。

注： *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*第 206 页上的第 8 节：直接存储器访问控制器 (DMA)*）。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 30.3 节：I2C 特性实现*）：PEC 传输由 NBYTES 计数器管理。请参见*第 882 页上的SMBus 从接收器*和*第 886 页上的SMBus 主接收器*。

注： *如果使用 DMA 进行接收，则无需使能 RXIE 位。*

30.4.16 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_STOP 配置位选择继续正常工作或者停止工作。

30.5 I2C 低功耗模式

表 162. 低功耗模式

模式	说明
睡眠	对 I2C 通信没有任何影响 I2C 中断可使器件退出睡眠模式。
停止	I2C 模块的寄存器内容仍被保持。
待机	I2C 外设掉电，退出待机模式后必须重新初始化。

30.6 I2C 中断

下表给出了 I2C 中断请求列表。

表 163. I2C 中断请求

中断事件	事件标志	事件标志/中断清除方法	中断使能控制位
接收缓冲区非空	RXNE	读取 I2C_RXDR 寄存器	RXIE
发送缓冲区中断状态	TXIS	写入 I2C_TXDR 寄存器	TXIE
停止位检测中断标志	STOPF	写入 STOPCF=1	STOPIE
传输完成等待重载	TCR	写入 I2C_CR2（NBYTES[7:0] ≠ 0）	TCIE
传输完成	TC	写入 START=1 或 STOP=1	
地址匹配	ADDR	写入 ADDRCF=1	ADDRIE

表 163. I2C 中断请求（续）

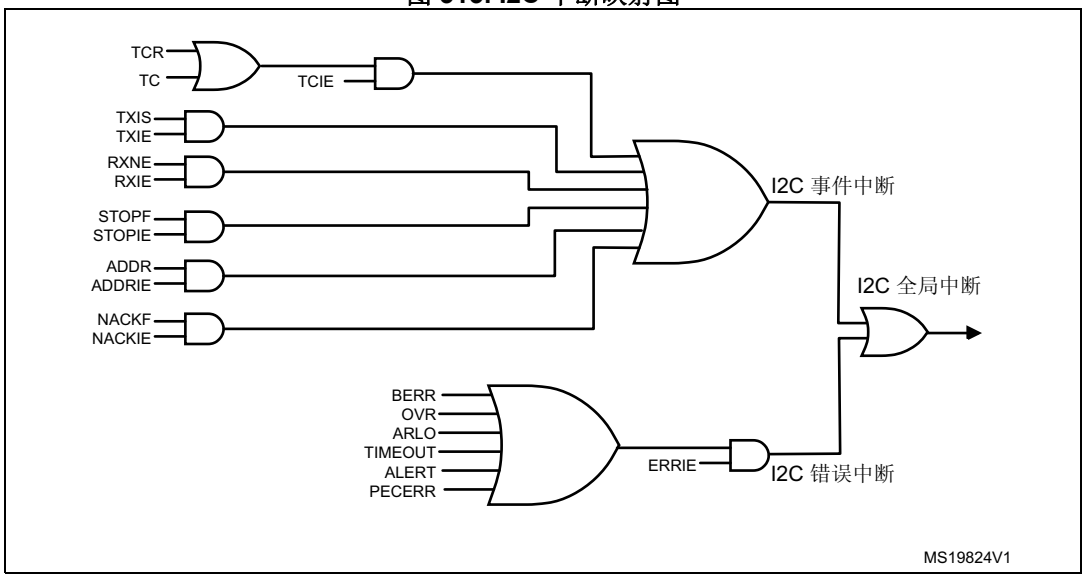
中断事件	事件标志	事件标志/中断清除方法	中断使能控制位
接收到 NACK 应答	NACKF	写入 NACKCF=1	NACKIE
总线错误	BERR	写入 BERRCF=1	ERRIE
仲裁丢失	ARLO	写入 ARLOCF=1	
上溢/下溢	OVR	写入 OVRCF=1	
PEC 错误	PECERR	写入 PECERRCF=1	
超时/t_{LOW} 错误	TIMEOUT	写入 TIMEOUTCF=1	
SMBus 报警	ALERT	写入 ALERTCF=1	

根据产品实现的不同，上述所有中断既可以共享同一个中断向量（I2C 全局中断），也可以分配到 2 个不同的中断向量上（I2C 事件中断和 I2C 错误中断）。有关详细信息，请参见。

要使能 I2C 中断，需按照以下顺序操作：

1. 配置 NVIC 中的 I2C IRQ 通道并将其使能。
2. 配置 I2C 以生成中断。

图 313. I2C 中断映射图



30.7 I2C 寄存器

有关寄存器说明中使用的缩写，请参见*第 58 页上的第 1.1 节*。

外设寄存器按字（32 位）进行访问。

30.7.1 控制寄存器 1 (I2C_CR1)

Control register 1

偏移地址：0x00

复位值：0x0000 0000

访问：无等待周期，正在对该寄存器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 PCLK1 + 6 x I2CCLK。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PECEN	ALERT EN	SMBD EN	SMBH EN	GCEN	Res.	NOSTR ETCH	SBC
								rw	rw	rw	rw	rw		rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDMA EN	TXDMA EN	Res.	ANF OFF	DNF				ERRIE	TCIE	STOP IE	NACK IE	ADDR IE	RXIE	TXIE	PE
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 **PECEN**：PEC 使能 (PEC enable)

0：禁止 PEC 计算

1：使能 PEC 计算

注：　如果不支持 SMBus 功能，该位保留，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

位 22 **ALERTEN**：SMBus 报警使能 (SMBus alert enable)

从机模式 (SMBHEN=0)：

0：将 SMBA 引脚释放为高电平并禁止报警响应地址头：0001100x 后跟 NACK。

1：将 SMBA 引脚驱动为低电平并使能报警响应地址头：0001100x 后跟 ACK。

主机模式 (SMBHEN=1)：

0：不支持 SMBus 报警引脚 (SMBA)。

1：支持 SMBus 报警引脚 (SMBA)。

注：　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，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

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：　如果不支持 SMBus 功能，该位保留，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

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： 如果不支持 SMBus 功能，该位保留，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

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 保留，必须保持复位值。

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： 该位只能在 I2C 禁止时 (PE = 0) 编程。

位 16 **SBC**：从设备模式下的字节控制 (Slave byte control)

该位用于在从设备模式下使能硬件字节控制。

0：禁止从设备模式下的字节控制

1：使能从设备模式下的字节控制

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： 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： 如果模拟滤波器也已使能，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 interrupts enable)
0：禁止错误检测中断
1：使能错误检测中断
注： *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： *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： *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 3 个 APB 时钟周期。*

30.7.2 控制寄存器 2 (I2C_CR2)

Control register 2

偏移地址：0x04

复位值：0x0000 0000

访问：无等待周期，正在对该寄存器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 PCLK1 + 6 x I2CCLK。

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							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NACK	STOP	START	HEAD 10R	ADD10	RD_W RN	SADD[9:0]									
rs	rs	rs	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)

此位由软件置 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： *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，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

位 25 **AUTOEND**：自动结束模式（主模式）(Automatic end mode (master mode))

此位由软件置 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： *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)

此位由软件置 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，该字段为无关字段。

注： *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))

此位由软件置 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： *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： *向该位写入“0”不起作用。*

位 13 **START**：起始位生成 (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、出现超时错误时、或者 PE = 0 时由硬件清零。它也可由软件清零，方法是向 I2C_ICR 寄存器中的 ADDRCF 位写入“1”。

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： *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： *START 位置 1 时，不允许更改此位。*

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： *START 位置 1 时，不允许更改此位。*

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： *START 位置 1 时，不允许更改此位。*

位 9:8 **SADD[9:8]**：从设备地址的第 8 到第 9 位（主模式）(Slave address bit 9:8 (master mode))
在 7 位寻址模式 (ADD10 = 0) 下：
这些位无意义
在 10 位寻址模式 (ADD10 = 1) 下：
这些位应写入待发送从地址的第 8 和第 9 位
注： START 位置 1 时，不允许更改这些位。

位 7:1 **SADD[7:1]**：从地址位 7:1（主模式）(Slave address bit 7:1 (master mode))
在 7 位寻址模式 (ADD10 = 0) 下：
这些位应写入待发送的 7 位从地址
在 10 位寻址模式 (ADD10 = 1) 下：
这些位应写入待发送从地址的第 1 到第 7 位。
注： START 位置 1 时，不允许更改这些位。

位 0 **SADD0**：从地址位 0（主模式）(Slave address bit 0 (master mode))
在 7 位寻址模式 (ADD10 = 0) 下：
该位无意义
在 10 位寻址模式 (ADD10 = 1) 下：
该位应写入待发送从地址的第 0 位
注： START 位置 1 时，不允许更改这些位。

30.7.3 设备自身地址 1 寄存器 (I2C_OAR1)

Own address 1 register

偏移地址：0x08

复位值：0x0000 0000

访问：无等待周期，正在对该寄存器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 PCLK1 + 6 x I2CCLK。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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[9:8]		OA1[7:1]							OA1[0]
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： 仅可在 OA1EN=0 时写入该位。

位 9:8 **OA1[9:8]**：接口地址 (Interface address)
7 位寻址模式：无关
10 位寻址模式：地址位 9:8
注： 仅可在 OA1EN=0 时写入这些位。

位 7:1 **OA1[7:1]**：接口地址 (Interface address)
地址位 7:1
注： 仅可在 OA1EN=0 时写入这些位。

位 0 **OA1[0]**：接口地址 (Interface address)
7 位寻址模式：无关
10 位寻址模式：地址位 0
注： 仅可在 OA1EN=0 时写入该位。

30.7.4 设备自身地址 2 寄存器 (I2C_OAR2)

Own address 2 register

偏移地址：0x0C

复位值：0x0000 0000

访问：无等待周期，正在对该寄存器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 PCLK1 + 6 x I2CCLK。

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

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： 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)
地址位 7:1
注： 仅可在 OA2EN=0 时写入这些位。

位 0 保留，必须保持复位值。

30.7.5 时序寄存器 (I2C_TIMINGR)

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			

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

位 31:28 **PRESC[3:0]**：时序预分频因子 (Timing prescaler)
该字段用于对 I2CCLK 进行预分频，以生成用于数据建立和保持计数器（请参见*第 851 页上的 I2C 时序*）以及 SCL 高电平和低电平计数器（请参见*第 864 页上的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)
在发送模式下，该字段用于在 SDA 边沿和 SCL 上升沿之间生成延时 t_{SCLDEL}。
$t_{SCLDEL} = (SCLDEL+1) \times t_{PRESC}$
注： 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)
在发送模式下，该字段用于在 SCL 下降沿和 SDA 边沿之间生成延时 t_{SDADEL}。
$t_{SDADEL} = SDADEL \times t_{PRESC}$
注： 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，该字段用于生成 SCL 高电平周期。
$t_{SCLH} = (SCLH+1) \times t_{PRESC}$
注： SCLH 还用于生成 $t_{SU:STO}$ 和 $t_{HD:STA}$ 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，该字段用于生成 SCL 低电平周期。
$t_{SCLL} = (SCLL+1) \times t_{PRESC}$
注： SCLL 还用于生成 t_{BUF} 和 $t_{SU:STA}$ 时序。

注： 该寄存器必须在 I2C 禁止时 (PE = 0) 进行配置。

30.7.6 超时寄存器 (I2C_TIMEOUTR)

Timeout register

偏移地址：0x14

复位值：0x0000 0000

访问：无等待周期，正在对该寄存器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 PCLK1 + 6 x I2CCLK。

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											
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											

位 31 **TEXTEN**：时钟信号延展超时使能 (Extended clock timeout enable)

0：禁止时钟信号延展超时检测

1：使能时钟信号延展超时检测 当 I2C 接口执行 SCL 延展的累积时间超过 $t_{LOW:EXT}$ 时，将检测到超时错误 (TIMEOUT=1)。

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)

该字段用于配置累积时钟延展超时：

在主模式下，将检测主器件的累积时钟低电平延展时间 ($t_{LOW:MEXT}$)

在从模式下，将检测从器件的累积时钟低电平延展时间 ($t_{LOW:SEXT}$)

$t_{LOW:EXT}$= (TIMEOUTB+1) x 2048 x t_{I2CCLK}

注： 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： 仅可在 TIMOUTEN=0 时写入该位。

位 11:0 **TIMEOUTA[11:0]**：总线超时 A (Bus Timeout A)

该字段用于配置：

– SCL 低电平超时条件 $t_{TIMEOUT}$（当 TIDLE=0 时）

$t_{TIMEOUT}$= (TIMEOUTA+1) x 2048 x t_{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： 仅可在 TIMOUTEN=0 时写入这些位。

注： 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 30.3 节：I2C 特性实现。

30.7.7 中断和状态寄存器 (I2C_ISR)

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
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 2 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**：总线繁忙 (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： *当 PE=0 时，该位由硬件清零。*
如果不支持 SMBus 功能，该位保留，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

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： *当 PE=0 时，该位由硬件清零。*
如果不支持 SMBus 功能，该位保留，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

位 11 **PECERR**：接收期间的 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：　当 PE=0 时，该位由硬件清零。

如果不支持 SMBus 功能，该位保留，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

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：　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：　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：　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：　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：　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，该位置位的前提是外设已经已经发出停止位。

– 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：　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：　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：　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：　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：　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：　当 PE=0 时，该位由硬件置 1。

30.7.8 中断清零寄存器 (I2C_ICR)

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.	ALERT CF	TIM OUTCF	PECCF	OVRCF	ARLO CF	BERR CF	Res.	Res.	STOP CF	NACK CF	ADDR CF	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 写入此位时，I2C_ISR 寄存器中的 ALERT 标志将清零。
注：　如果不支持 SMBus 功能，该位保留，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

位 12 **TIMOUTCF**：超时检测标志清零 (Timeout detection flag clear)
将 1 写入此位时，I2C_ISR 寄存器中的 TIMEOUT 标志将清零。
注：　如果不支持 SMBus 功能，该位保留，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

位 11 **PECCF**：PEC 错误标志清零 (PEC Error flag clear)
将 1 写入此位时，I2C_ISR 寄存器中的 PECERR 标志将清零。
注：　如果不支持 SMBus 功能，该位保留，并由硬件强制为“0”。请参见第 30.3 节：I2C 特性实现。

位 10 **OVRCF**：上溢/下溢标志清零 (Overrun/Underrun flag clear)
将 1 写入此位时，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 写入此位时，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 写入此位时，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 写入此位时，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)
将 1 写入此位时，I2C_ISR 寄存器中的 ACKF 标志将清零。

位 3 **ADDRCF**：地址匹配标志清零 (Address Matched flag clear)
将 1 写入此位时，I2C_ISR 寄存器中的 ADDR 标志将清零。将 1 写入此位时，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。

30.7.9 PEC 寄存器 (I2C_PECR)

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							

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，此字段包含内部 PEC。
当 PE=0 时，PEC 由硬件清零。

注： *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 30.3 节：I2C 特性实现。*

30.7.10 接收数据寄存器 (I2C_RXDR)

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							

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。

30.7.11 发送数据寄存器 (I2C_TXDR)

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							

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： 仅可在 TXE=1 时写入这些位。

30.7.12 I2C寄存器映射

下表提供了 I2C 寄存器映射和复位值。

表 164. I2C 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
0x0	**I2C_CR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PECEN	ALERTEN	SMBDEN	SMBHEN	GCEN	Res.	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
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.		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 164. I2C 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
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 64 页上的第 2.2.2 节*。

31 通用同步异步收发器 (USART)

31.1 前言

通用同步异步收发器 (USART) 能够灵活地与外部设备进行全双工数据交换，满足外部设备对工业标准 NRZ 异步串行数据格式的要求。USART 通过可编程波特率发生器提供了多种波特率。

它支持同步单向通信、半双工单线通信以及多处理器通信； 还支持 LIN（局域互连网络）、智能卡协议与 IrDA（红外线数据协会）SIR ENDEC 规范，以及调制解调器操作 (CTS/RTS)。

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。

31.2 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，因而为速度容差与时钟容差的灵活配置提供了可能
- 当 USART 时钟源为系统时钟频率（最大值为 216 MHz）且采用 8 倍过采样时，通用可编程收发波特率最高为 27 Mbps
- 双时钟域允许：
  - 方便的波特率编程，独立于 PCLK 重新编程
- 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模式和时钟输出
- 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 引脚配置可交换
- 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
- 十四个具有标志位的中断源
- 多处理器通信
  如果地址不匹配，USART 将进入静默模式
- 从静默模式唤醒（通过空闲线检测或地址标记检测）

31.3 USART 扩展特性

- LIN 主模式同步断路发送功能和 LIN 从模式断路检测功能
  - 当 USART 被配置为 LIN 使用时，可生成 13 位断路和检测 10/11 位断路
- 正常模式下支持 3/16 位持续时间的 IrDA SIR 编解码器
- 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 或 1.5 个停止位
- 支持 ModBus 通信
  - 超时功能
  - CR/LF 字符识别

31.4 USART 实现

表 165. STM32F75xxx 和 STM32F74xxx USART 特性

USART 模式/特性(1)	USART1/USART2/ USART3/USART6	UART4/UART5/ UART7/UART8
调制解调器的硬件流控	X	X
使用 DMA 进行连续通信	X	X
多处理器通信	X	X
同步模式	X	-
智能卡模式	X	-
单线半双工通信	X	X
IrDA SIR ENDEC 模块	X	X
LIN 模式	X	X
双时钟域	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
USART 数据长度	7 位、8 位和 9 位	

1. X = 支持。

31.5 USART 功能说明

任何 USART 双向通信均需要至少两个引脚：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。

  串行数据输入引脚。过采样技术可区分有效输入数据和噪声，从而用于恢复数据。
- **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。在单线和智能卡模式下，该 I/O 用于发送和接收数据。

正常 USART 模式下，通过这些引脚发送和接收串行数据。这些帧包括：

- 发送或接收前保持空闲线路
- 起始位
- 数据字（字长 7 位、8 位或 9 位），最低有效位在前
- 用于指示帧传输已完成的 0.5 个、1 个、1.5 个、2 个停止位
- USART 接口使用波特率发生器
- 状态寄存器 (USARTx_ISR)
- 接收和发送数据寄存器（USARTx_RDR、USARTx_TDR）
- 波特率寄存器 (USARTx_BRR)
- 智能卡模式下的保护时间寄存器 (USARTx_GTPR)。

有关各个位的定义，请参见*第 944 页上的第 31.8 节：USART 寄存器*。

在同步模式和智能卡模式下连接时需要以下引脚：

- **SCLK**：时钟输出。该引脚用于输出发送器数据时钟，以便按照 SPI 主模式进行同步发送（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）。RX 上可同步接收并行数据。这一点可用于控制带移位寄存器的外设。时钟相位和极性可通过软件编程。在智能卡模式下，SCLK 输出可向智能卡提供时钟。

在 RS232 硬件流控制模式下需要以下引脚：

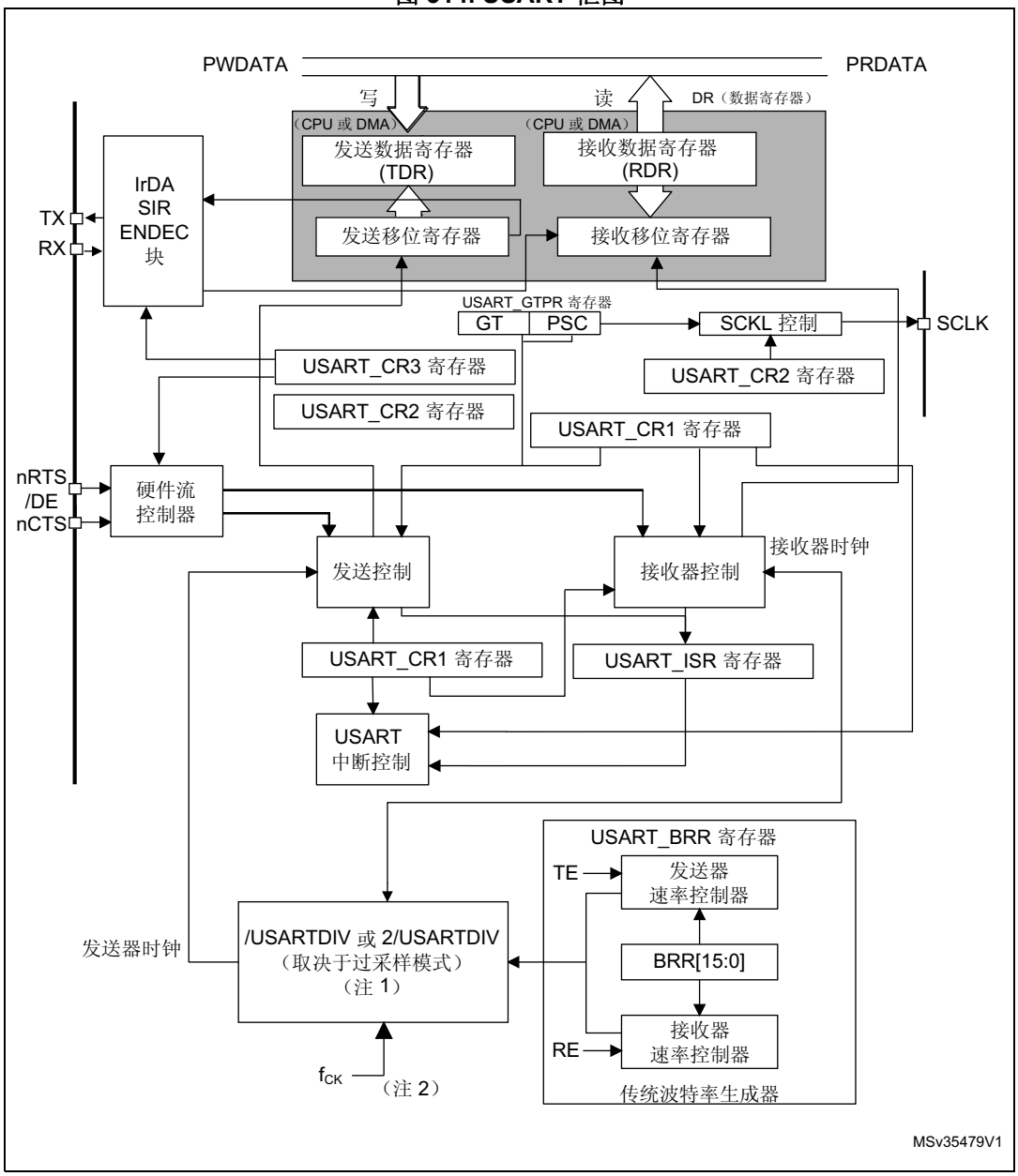
- **nCTS**：“清除以发送”用于在当前传输结束时阻止数据发送（高电平时）。
- **nRTS**：“请求以发送”用于指示 USART 已准备好接收数据（低电平时）。

在 RS485 硬件控制模式下需要以下引脚：

- **DE**：“驱动器使能”用于激活外部收发器的发送模式。

注： *DE 和 nRTS 共用同一个引脚。*

图 314. USART 框图



1. 有关在 USARTx_BRR 寄存器中编码 USARTDIV 的详细信息，请参见*第 31.5.4 节：波特率生成*。
2. f_{CK} 可以是 f_{LSE}、f_{HSI}、f_{PCLK}、f_{SYS}。

31.5.1 USART 字符说明

可通过对 USARTx_CR1 寄存器中的 M[1:0] 位进行编程来选择 7 位、8 位或 9 位的字长（请参见*图 315*）。

- 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： *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）。只有某些 USART 支持 7 位模式。*

在默认配置下，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。

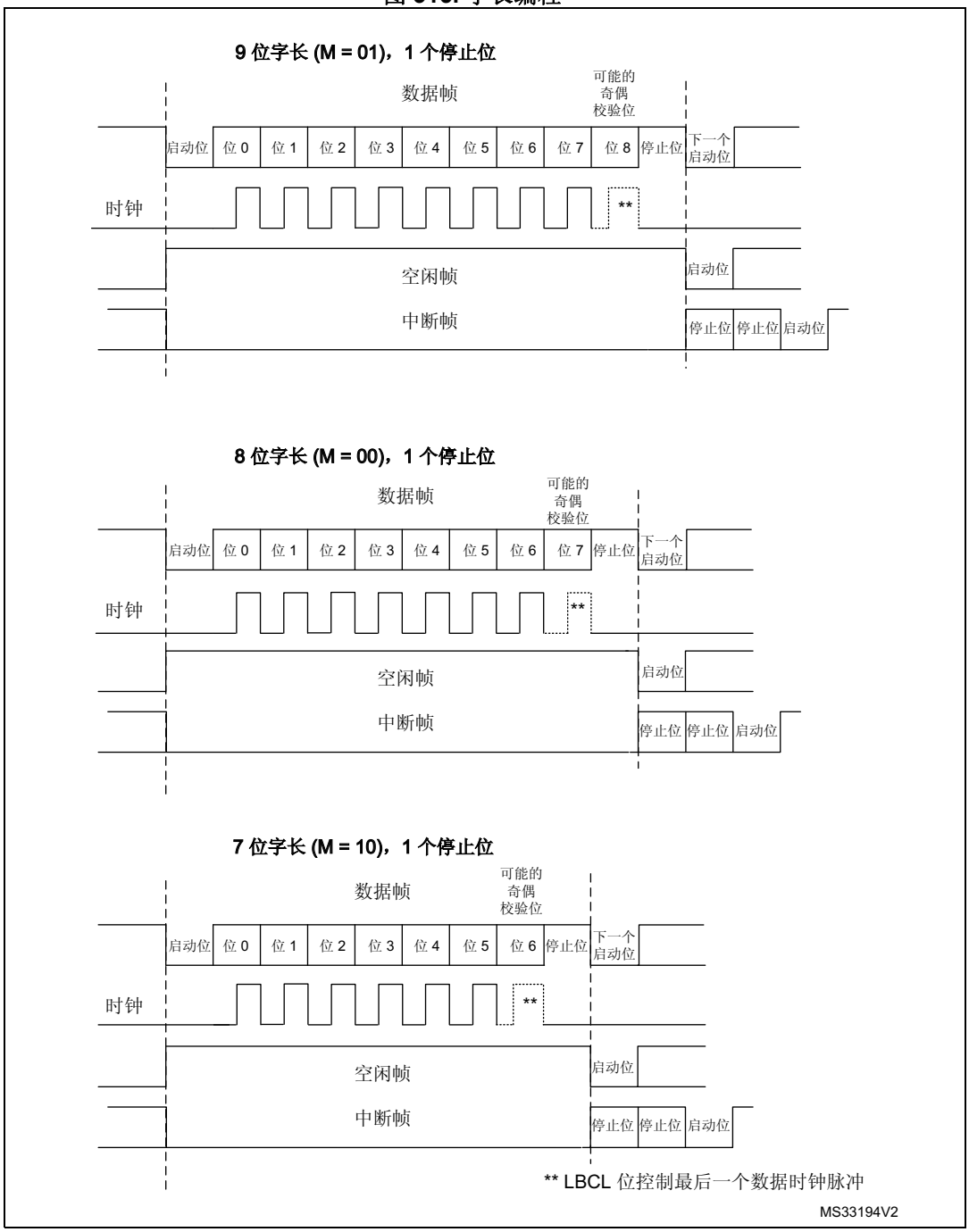
空闲字符可理解为整个帧周期内电平均为“1”。（停止位的电平也是“1”）。

停止字符可理解为在一个帧周期内接收到的电平均为“0”。发送器在中断帧的末尾插入 2 个停止位。

发送和接收由通用波特率发生器驱动，发送器和接收器的使能位分别置 1 时将生成相应的发送时钟和接收时钟。

下面给出了各个块的详细信息。

图 315. 字长编程



31.5.2 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 SCLK 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）。该模式下，USARTx_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 314*）。

每个字符前面都有一个起始位，其逻辑电平在一个位周期内为低电平。字符由可配置数量的停止位终止。

USART 支持以下停止位：0.5、1、1.5 和 2 个停止位。

注： *向 USARTx_TDR 中写入要发送的数据前，TE 位必须先置 1。*

数据发送期间不应复位 TE 位。发送期间复位 TE 位会冻结波特率计数器，从而将损坏 TX 引脚上的数据。当前传输的数据将会丢失。

使能 TE 位后，将会发送空闲帧。

可配置的停止位

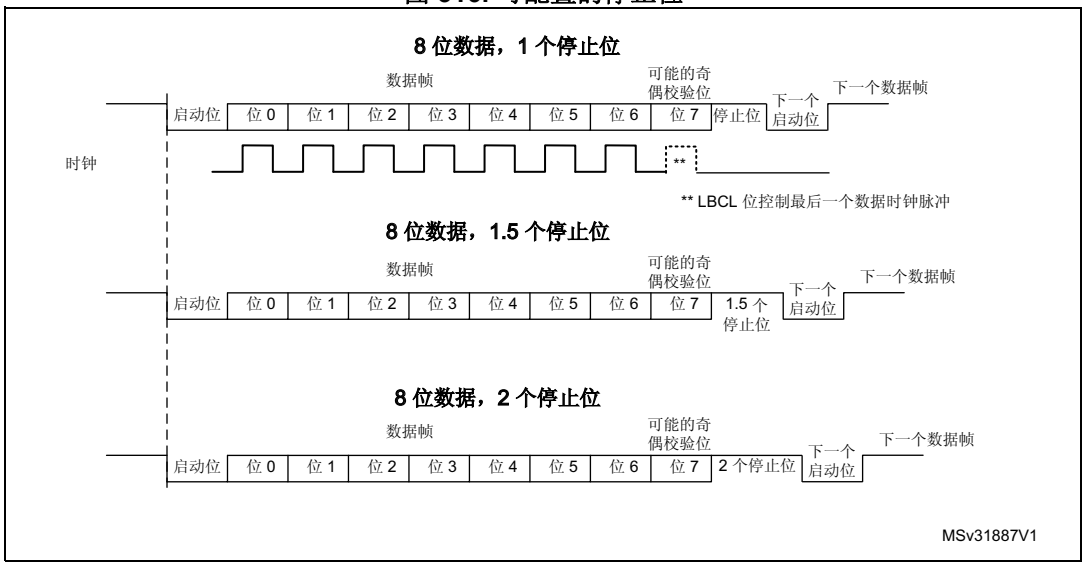
可以在控制寄存器 2 的位 13 和位 12 中编程将随各个字符发送的停止位的数量。

- ***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。
- ***0.5 个停止位***：在智能卡模式下接收数据时使用。

空闲帧发送将包括停止位。

中断发送是 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（请参见*图 316*）。无法传送长中断（中断长度大于 9/10/11 个低电平位）。

图 316. 可配置的停止位



字符发送步骤

1. 对 USARTx_CR1 中的 M 位进行编程以定义字长。
2. 使用 USARTx_BRR 寄存器选择所需波特率。
3. 对 USARTx_CR2 中的停止位数量进行编程。
4. 通过向 USARTx_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 USARTx_CR3 中的 DMA 使能 (DMAT)。按照多缓冲区通信中的解释说明配置 DMA 寄存器。
6. 将 USARTx_CR1 中的 TE 位置 1 以便在首次发送时发送一个空闲帧。
7. 在 USARTx_TDR 寄存器中写入要发送的数据（该操作将清零 TXE 位）。为每个要在单缓冲区模式下发送的数据重复这一步骤。
8. 向 USARTx_TDR 寄存器写入最后一个数据后，等待至 TC=1。这表明最后一个帧的传送已完成。禁止 USART 或进入暂停模式时需要此步骤，以避免损坏最后一次发送。

单字节通信

始终通过向发送数据寄存器写入数据将 TXE 位清零。

TXE 位由硬件置 1，它表示：

- 数据已从 USARTx_TDR 寄存器移到移位寄存器中且数据发送已开始。
- USARTx_TDR 寄存器为空。
- USARTx_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。

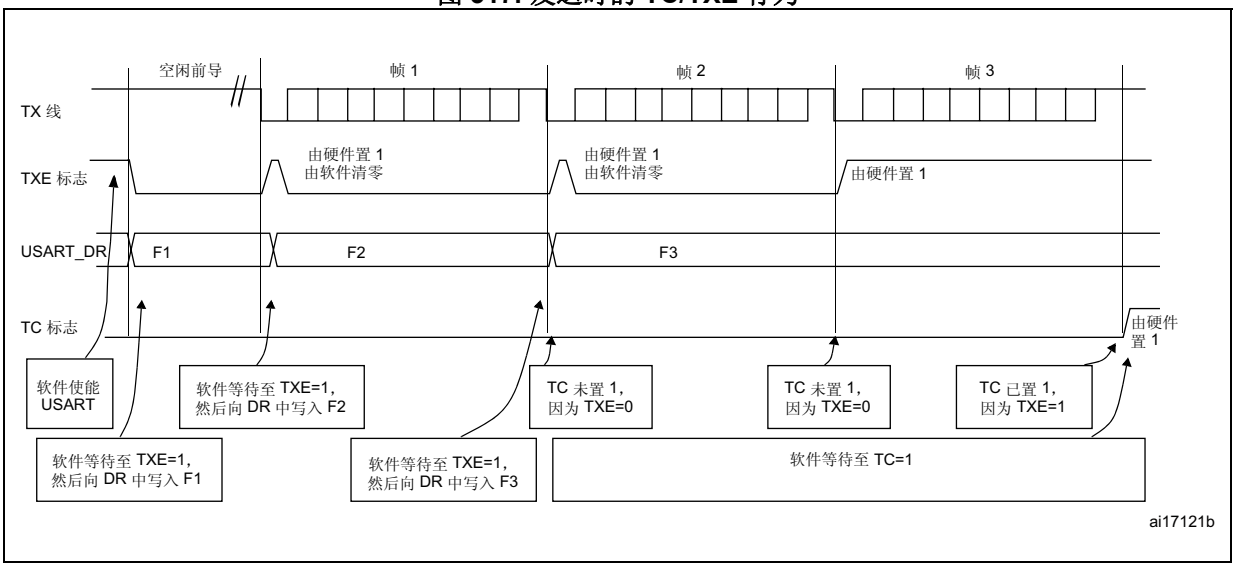
发送时，要传入 USARTx_TDR 寄存器的写指令中存有 TDR 寄存器中的数据，接下来，该数据将在当前进行的发送结束时复制到移位寄存器中。

未发送时，要传入 USARTx_TDR 寄存器的写指令将数据置于移位寄存器中，数据发送开始时，TXE 位置 1。

如果帧已发送（停止位后）且 TXE 位置 1，TC 位将变为高电平。如果 USARTx_CR1 寄存器中的 TCIE 位置 1，将生成中断。

向 USARTx_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 USART 或使微控制器进入低功率模式（请参见*图 317：发送时的 TC/TXE 行为*）。

图 317. 发送时的 TC/TXE 行为



中断字符

将 SBKRQ 位置 1 将发送一个中断字符。中断帧的长度取决于 M 位（请参见*图 315*）。

如果将“1”写入 SBKRQ 位，则当前字符发送完成后，将在 TX 线路上发送一个中断字符。通过写操作将 SBKF 位置 1 并在中断字符发送完成时（发送中断字符后的停止位期间），该位由硬件复位。USART 在中断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。

这种情况下，应用需要在之前插入的所有数据（包括尚未发送的数据）后发送中断字符，软件应等到 TXE 标志使能后将 SBKRQ 位置 1。

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。

31.5.3 接收器

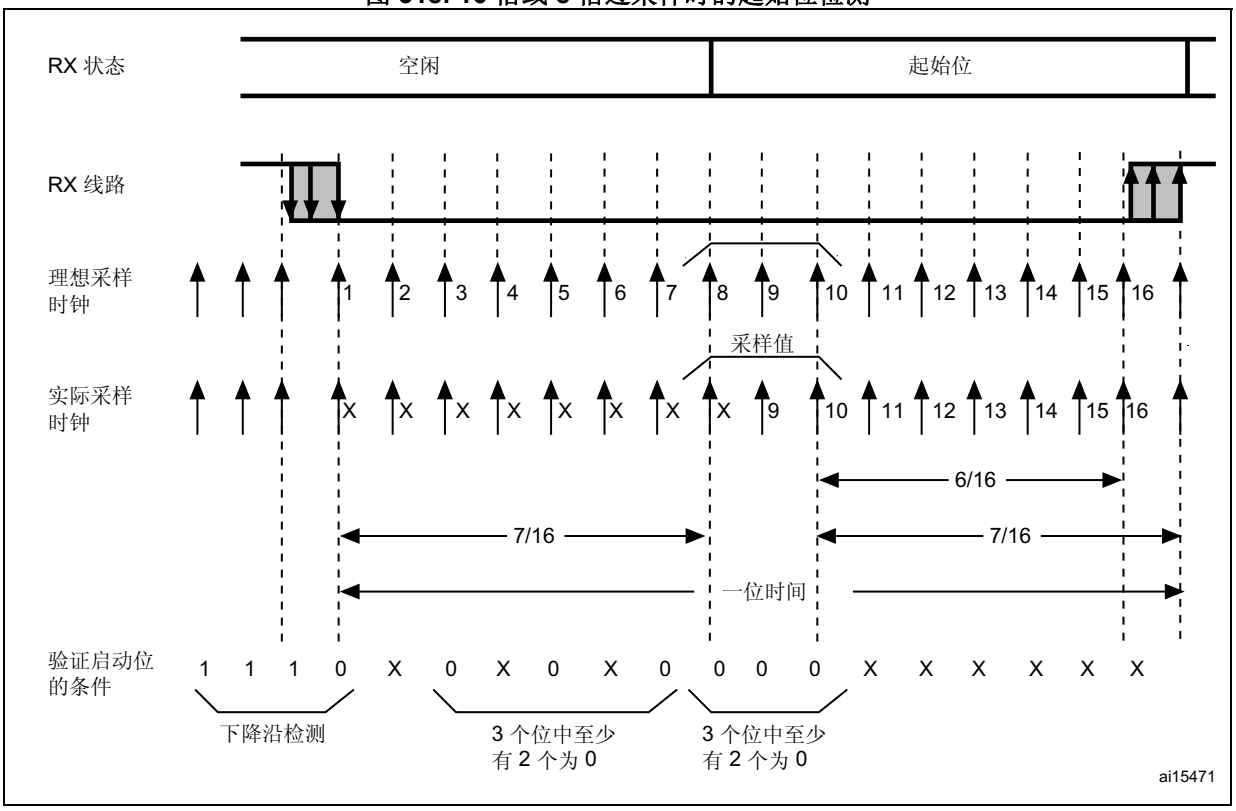
USARTx 可接收 7 位、8 位或 9 位的数据字，具体取决于 USART_CR1 寄存器中的 M 位。

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起始位。
此序列为：1 1 1 0 X 0 X 0X 0X 0 X 0X 0。

图 318. 16 倍或 8 倍过采样时的起始位检测



注： *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检测到这 3 位均为 0），可确认起始位（RXNE 标志位置 1，RXNEIE=1 时生成中断）。

满足以下条件时，可验证起始位（RXNE 标志位置 1，如果 RXNEIE=1 则生成中断）但 NF 噪声标志位置 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。

如果条件 a 或条件 b 都不满足，则启动检测中止，接收器返回空闲状态（无标志位置 1）。

字符接收

USART 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，USARTx_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

1. 对 USARTx_CR1 中的 M 位进行编程以定义字长。
2. 使用波特率寄存器 USARTx_BRR 选择所需波特率。
3. 对 USARTx_CR2 中的停止位数量进行编程。
4. 通过向 USARTx_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 USARTx_CR3 中的 DMA 使能 (DMAR)。按照多缓冲区通信中的解释说明配置 DMA 寄存器。
6. 将 RE 位 USARTx_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：

- 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（及其相应的错误标志）。
- 如果 RXNEIE 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。也可使用 RXNE 位将 PE 标志置 1。
- 在多缓冲区模式下，每接收到一个字节后 RXNE 均置 1，然后通过 DMA 对接收数据寄存器执行读操作清零。
- 在单缓冲区模式下，通过软件对 USARTx_RDR 寄存器执行读操作将 RXNE 位清零。也可以通过将 USARTx_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。

中断字符

接收到中断字符时，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，处理步骤与接收到数据的情况相同；如果 IDLEIE 位置 1，则会产生中断。

上溢错误

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 请求时，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 USARTx_RDR 执行读操作时可使用先前的数据。
- 移位寄存器将被覆盖。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。
- 通过设置 ICR 寄存器中的 ORECF 位复位 ORE 位。

注： *ORE 位置 1 时表示至少 1 个数据丢失。存在两种可能：*

- 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，则表示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（和丢失）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。

选择时钟源和合适的过采样方法

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 (RCC) 部分）。时钟源必须在使能 USART（通过将 UE 位置 1）前选择。

时钟源频率为 f_{CK}。

如果支持双时钟域， 时钟源可以是下列源之一：PCLK（默认）、LSE、HSI 或 SYSCLK。

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，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平衡。

可通过编程 USARTx_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 319* 和*图 320*）。

根据应用：

- 选择 8 倍过采样 (OVER8=1) 以获得更高的速度（高达 $f_{CK}/8$）。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 923 页上的第 31.5.5 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。这种情况下，最大速度限制为最高 $f_{CK}/16$，其中 f_{CK} 为时钟源频率。

可通过编程 USARTx_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：

- 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，NF 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*图 166*），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 923 页上的第 31.5.5 节：USART 接收器对时钟偏差的容差*）。这种情况下 NF 位始终不会置 1。

帧中检测到噪声时：

- 在 RXNE 位的上升沿时 NF 位置 1。
- 无效数据从移位寄存器传送到 USARTx_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，USARTx_CR3 寄存器中的 EIE 位置 1 时将发出中断。

通过设置 ICR 寄存器中的 NFCF 位复位 NF 位。

注： *LIN、智能卡和 IrDA 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 319. 16 倍过采样时的数据采样

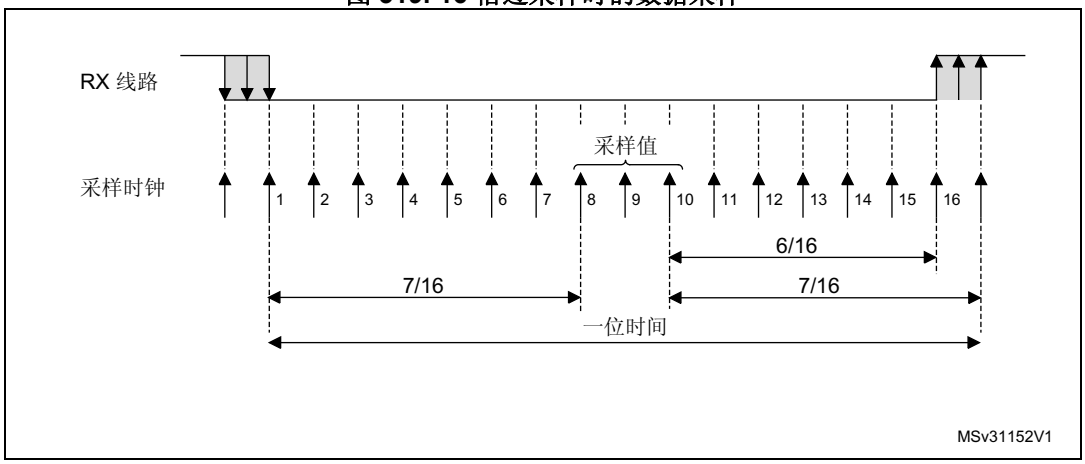


图 320. 8 倍过采样时的数据采样

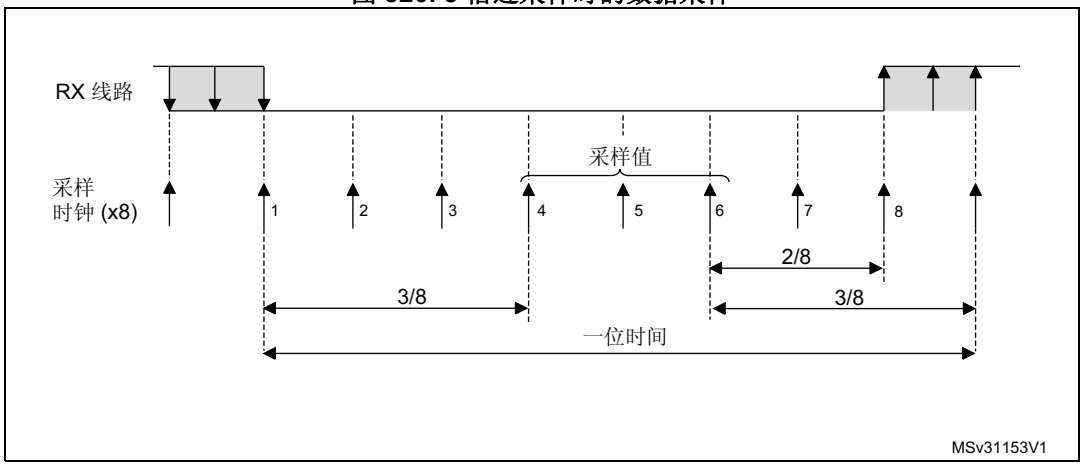


表 166. 通过采样数据进行噪声检测

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

以下情况下将检测到帧错误：

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。

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 USARTx_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，USARTx_CR3 寄存器中的 EIE 位置 1 时将发出中断。

通过将 1 写入 USARTx_ICR 寄存器中的 FECF 位来复位 FE 位。

接收期间可配置的停止位

可通过控制寄存器 2 中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），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中断帧。*
- ***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（智能卡模式）***：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（USARTx_CR1 寄存器中的 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 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 933 页上的第 31.5.13 节：智能卡模式*。
- ***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，则帧错误标志位将会置 1。发生帧错误时不检测第 2 个停止位。RXNE 标志将在第一个停止位末尾时置 1。

31.5.4 波特率生成

接收器和发送器（Rx 和 Tx）的波特率设置为 USARTx_BRR 寄存器中编程的相同值。

公式 1：适用于标准 USART（包括 SPI 模式）的波特率（OVER8 = 0 或 1）

16 倍过采样时的公式为：

$$\text{Tx/Rx baud} = \frac{f_{CK}}{\text{USARTDIV}}$$

8 倍过采样时的公式为：

$$\text{Tx/Rx baud} = \frac{2 \times f_{CK}}{\text{USARTDIV}}$$

公式 2：智能卡、LIN 和 IrDA 模式 (OVER8 = 0) 下的波特率

智能卡、LIN 和 IrDA 模式下只支持 16 倍过采样：

$$\text{Tx/Rx baud} = \frac{f_{CK}}{\text{USARTDIV}}$$

USARTDIV 是一个存放在 USARTx_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： *对 USARTx_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应在通信时发生更改。*

16 倍或 8 倍过采样时，USARTDIV 必须大于或等于 16d。

如何从 USARTx_BRR 寄存器中获取 USARTDIV

示例 1

要通过 f_{CK} = 8 MHz 获得 9600 波特率。

- 16 倍过采样时：
  USARTDIV = 8 000 000/9600
  BRR = USARTDIV = 833d = 0341h
- 8 倍过采样时：
  USARTDIV = 2 * 8 000 000/9600
  USARTDIV = 1666,66 (1667d = 683h)
  BRR[3:0] = 3h >> 1 = 1h
  BRR = 0x681

示例 2

要通过 f_{CK} = 48 MHz 获得 921.6 K 波特率。

- 16 倍过采样时：
  USARTDIV = 48 000 000/921 600
  BRR = USARTDIV = 52d = 34h
- 8 倍过采样时：
  USARTDIV = 2 * 48 000 000/921 600
  USARTDIV = 104 (104d = 68h)
  BRR[3:0] = USARTDIV[3:0] >> 1 = 8h >> 1 = 4h
  BRR = 0x64

表 167. 采用 8 倍过采样 (OVER8 = 1) 时，在 fCK = 216 MHz 下编程波特率时的误差计算(1)

所需波特率 (Bps)	实际波特率 (Bps)	BRR	误差
9600	9600.853	AFC4	0.00888968
19200	19201.707	57E2	0.00888968
38400	38403.414	2BF1	0.00888968
57600	57646.117	1D46	0.08006405
115200	115292.234	EA3	0.08006405
230400	230646.022	751	0.10678057
460800	463519.313	3A4	0.59012876
921600	927038.627	1D2	0.59012876
13500000	13500000.000	20	0.00000000
27000000	27000000.000	10	0.00000000

1. CPU 时钟越低，特定波特率的精度就越低。可达到的波特率的上限可使用这些数据进行确定。

表 168. 采用 16 倍过采样 (OVER8 = 0) 时，在 fCK = 216 MHz 下编程波特率时的误差计算(1)

所需波特率 (Bps)	实际波特率 (Bps)	BRR	误差
9600	9600.000	57E4	0.00000000
19200	19200.000	2BF2	0.00000000
38400	38400.000	15F9	0.00000000
57600	57600.000	EA6	0.00000000
115200	115200.000	753	0.00000000
230400	230522.946	3A9	0.05336179
460800	461538.462	1D4	0.16025641
921600	923076.923	EA	0.16025641
4000000	4000000.000	36	0.00000000
6000000	6000000.000	24	0.00000000
10000000	10285714.286	15	2.85714286
13500000	13500000.000	10	0.00000000

1. CPU 时钟越低，特定波特率的精度就越低。可达到的波特率的上限可使用这些数据进行确定。

31.5.5 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 < USART 接收器的容差

USART 接收器在*表 169* 和*表 170* 中指定的最大容许偏差下可正确接收数据，具体取决于以下选择：

- 由 USARTx_CR1 寄存器中的 M 位定义的 9 位、10 位或 11 位字符长度
- 由 USARTx_CR1 寄存器中的 OVER8 位定义的 8 倍或 16 倍过采样
- USARTx_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 USARTx_CR3 寄存器中 ONEBIT 位的值

表 169. 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 170. 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： *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）位持续时间的空闲帧时，*表 169 和表 170 中指定的数据可能与特例中的数据略微不同。

31.5.6 自动波特率检测

USART 能够根据接收一个字符检测并自动设置 USARTx_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匹配（16 倍过采样时，波特率介于 f_{CK}/65535 与 f_{CK}/16 之间； 8 倍过采样时，波特率介于 f_{CK}/65535 与 f_{CK}/8 之间）。

激活自动波特率检测前，必须先选择自动波特率检测模式。根据不同的字符模式，存在各种检测模式。

这些模式可通过 USARTx_CR2 寄存器中的 ABRMOD[1:0] 字段选择。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。

这些模式包括：

- **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 位模式开头的任意字符。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 对字符的其它位进行采样。
- **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 对字符的其它位进行采样。

  同时，对 RX 线路的每个中间转换执行其它检查。如果 RX 上的转换与接收器（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前，必须先通过向 USARTx_BRR 寄存器写入非零的波特率值来初始化该寄存器。

通过将 USARTx_CR2 寄存器中的 ABREN 位置 1 来激活自动波特率检测。之后 USART 将等待 RX 线路上的第一个字符。通过将 USARTx_ISR 寄存器中的 ABRF 标志置 1 来指示自动波特率操作完成。如果线路繁忙，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 错误标志位将置 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。

RXNE 中断将指示操作结束。

稍后，可随时通过复位 ABRF 标志（通过写入 0）重新启动自动波特率检测。

注： *如果在自动波特率操作期间禁止 USART (UE=0)，则可能损坏 BRR 值。*

31.5.7 多处理器通信

在多处理器通信中，以下位要保持清零：

- USART_CR2 寄存器中的 LINEN 位，
- USART_CR3 寄存器中的 HDSEL、IREN 和 SCEN 位。

可以通过 USART 进行多处理器通信（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 输出与其它 USART 的 RX 输入相连接。其它 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 输入相连接。

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 USARTx_CR1 寄存器中的 MME 位置 1。

在静默模式下：

- 不得将接收状态位置 1。
- 禁止任何接收中断。
- USARTx_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 USARTx_RQR 寄存器中的 MMRQ 位自动控制。

根据 USARTx_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：

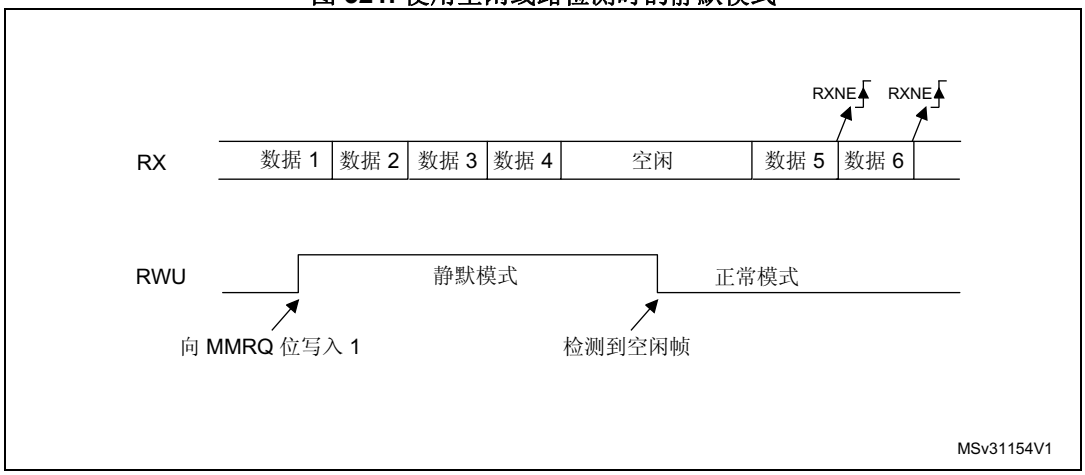
- 如果 WAKE 位被复位，则进行空闲线路检测，
- 如果 WAKE 位置 1，则进行地址标记检测。

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。

当检测到空闲帧时，它会被唤醒。此时 RWU 位会由硬件清零，但 USARTx_ISR 寄存器中的 IDLE 位不会置 1。*图 321* 中给出了使用空闲线路检测时静默模式行为的示例。

图 321. 使用空闲线路检测时的静默模式



注： *如果在 IDLE 字符已经过去时将 MMRQ 位置 1，将不会进入静默模式（RWU 未置 1）。*

如果在线路处于空闲状态时激活 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 USARTx_CR2 寄存器的 ADD 位中进行设置。

注： *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]）。*

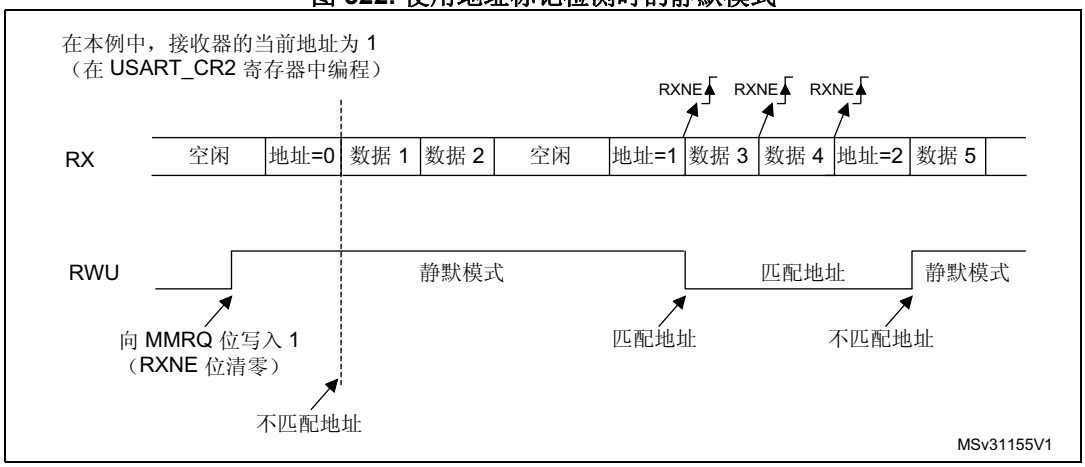
当接收到与其编程地址不匹配的地址字符时，USART 会进入静默模式。此时，RWU 位将由硬件置 1。USART 进入静默模式后，RXNE 标志不会针对此地址字节置 1，也不会发出中断或 DMA 请求。

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 位会针对地址字符置 1。

图 322 中给出了使用地址标记检测时静默模式行为的示例。

图 322. 使用地址标记检测时的静默模式



31.5.8 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，无需软件开销或其它资源。

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 USARTx_CR2 寄存器中的 RTOEN 位和 USARTx_CR1 寄存器中的 RTOIE 位激活。与 2 个字符时间（例如 22 x 位持续时间）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] 字段中编程 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。

31.5.9 奇偶校验控制

将 USARTx_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 171* 中列出了可能的 USART 帧格式。

表 171. 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，P：奇偶校验位。在数据寄存器中，PB 始终位于 MSB 位置（第 9 位、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（USARTx_CR1 寄存器中的 PS 位 = 0）时，校验位为 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（USARTx_CR1 寄存器中的 PS 位 = 1）时，校验位为 1。

接收时进行奇偶校验检查

如果奇偶校验检查失败，则 USARTx_ISR 寄存器中的 PE 标志置 1；如果 USARTx_CR1 寄存器中 PEIE 位置 1，则会生成中断。通过软件将 1 写入 USARTx_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 USARTx_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）。

31.5.10 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 908 页上的第 31.4 节：USART 实现*。

通过将 USARTx_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：

- USARTx_CR2 寄存器中的 CLKEN 位，
- USARTx_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 主器件中发送时必须采用*第 31.5.2 节：发送器*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”位作为断路字符。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，将激活断路检测电路。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，只要发生断路即可检测出来。

接收器（USARTx_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断路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（USARTx_CR2 中 LBDL = 0 时）或 11 个（USARTx_CR2 中 LBDL=1 时）连续位均检测为“0”，且其后跟随分隔符，则 USARTx_ISR 寄存器中的 LBDF 标志将置 1。如果 LBDIE 位=1，则会生成中断。在验证断路前，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 次采样前已对“1”采样，则断路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，不会再进行断路检测。

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断路帧中），接收器即会停止，直到断路检测电路接收到“1”（断路字不完整时）或接收到分隔符（检测到断路时）为止。

*第 929 页上的图 323：LIN 模式下的断路检测（11 位断路长度 —— LBDL 位置 1）*中显示了断路检测器状态机和断路标志的行为。

*第 930 页上的图 324：LIN 模式下的断路检测与帧错误检测*中列出了断路帧的示例。

图 323. LIN 模式下的断路检测（11 位断路长度 —— LBDL 位置 1）

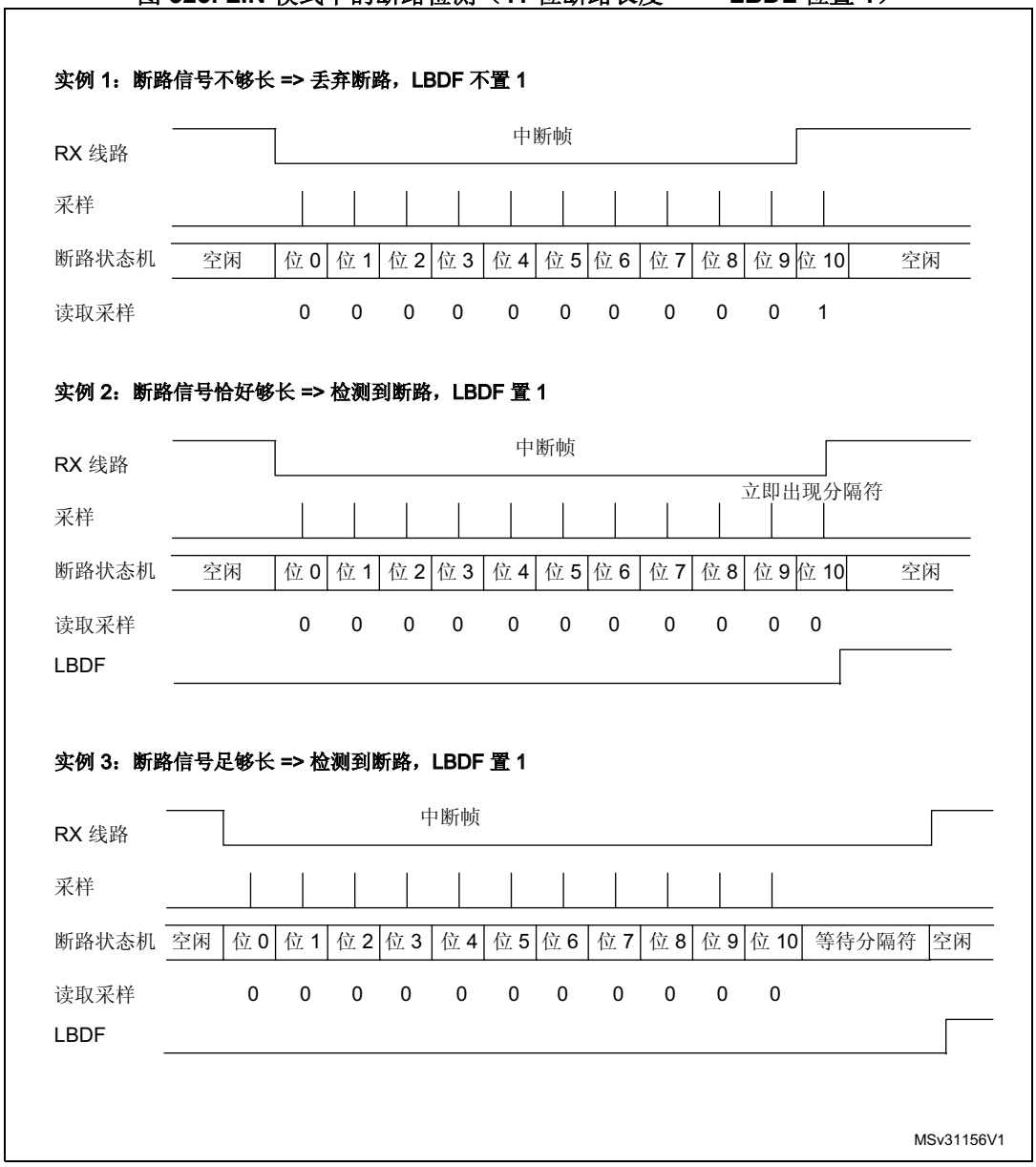
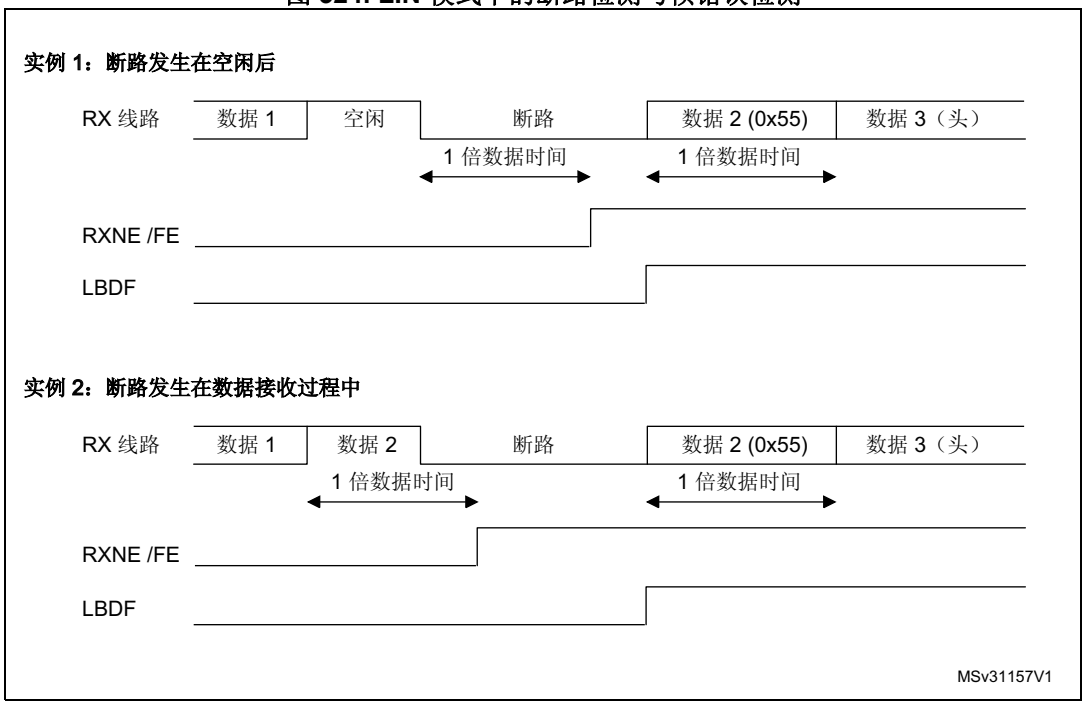


图 324. LIN 模式下的断路检测与帧错误检测



31.5.11 USART 同步模式

通过将 USARTx_CR2 寄存器中的 CLKEN 位写入 1 来选择同步模式。在同步模式下，必须将以下位清零：

- USARTx_CR2 寄存器中的 LINEN 位，
- USARTx_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。SCLK 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 SCLK 引脚发送时钟脉冲。在最后一个有效数据位（地址标记）期间，会（也可能不会）生成时钟脉冲，这取决于 USARTx_CR2 寄存器中 LBCL 位的状态。USARTx_CR2 寄存器中的 CPOL 位用于选择时钟极性；USARTx_CR2 寄存器中的 CPHA 位用于选择外部时钟的相位（请参见*图 325*、*图 326* 和*图 327*）。

在空闲状态、报头模式和发送断路期间，外部 SCLK 时钟处于未激活状态。

USART 发送器在同步模式下的工作方式与异步模式下完全相同。但是由于 SCLK 与 TX 同步（根据 CPOL 和 CPHA），因此 TX 上的数据是同步的。

在此模式下，USART 接收器的工作方式与异步模式下不同。如果 RE=1，则数据在 SCLK 上采样（上升或下降沿，取决于 CPOL 和 CPHA），而不会进行任何过采样。此时必须确保建立时间和保持时间（取决于波特率：1/16 位持续时间）。

注： *SCLK 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（数据寄存器 USARTx_TDR 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。*

当 USART 被禁止时 (UE=0)，必须选择 LBCL、CPOL 和 CPHA 位以确保时钟脉冲正常工作。

图 325. USART 同步发送示例

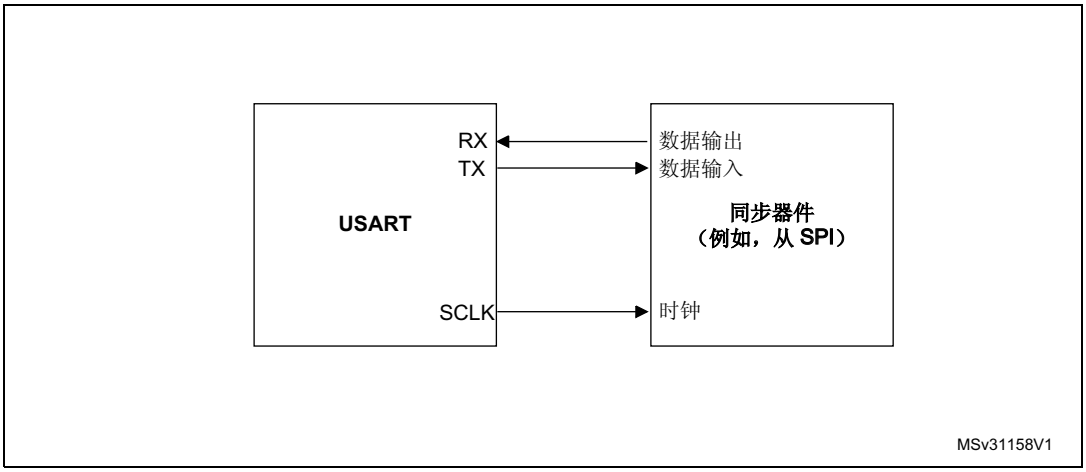


图 326. USART 数据时钟时序图 (M 位 = 00)

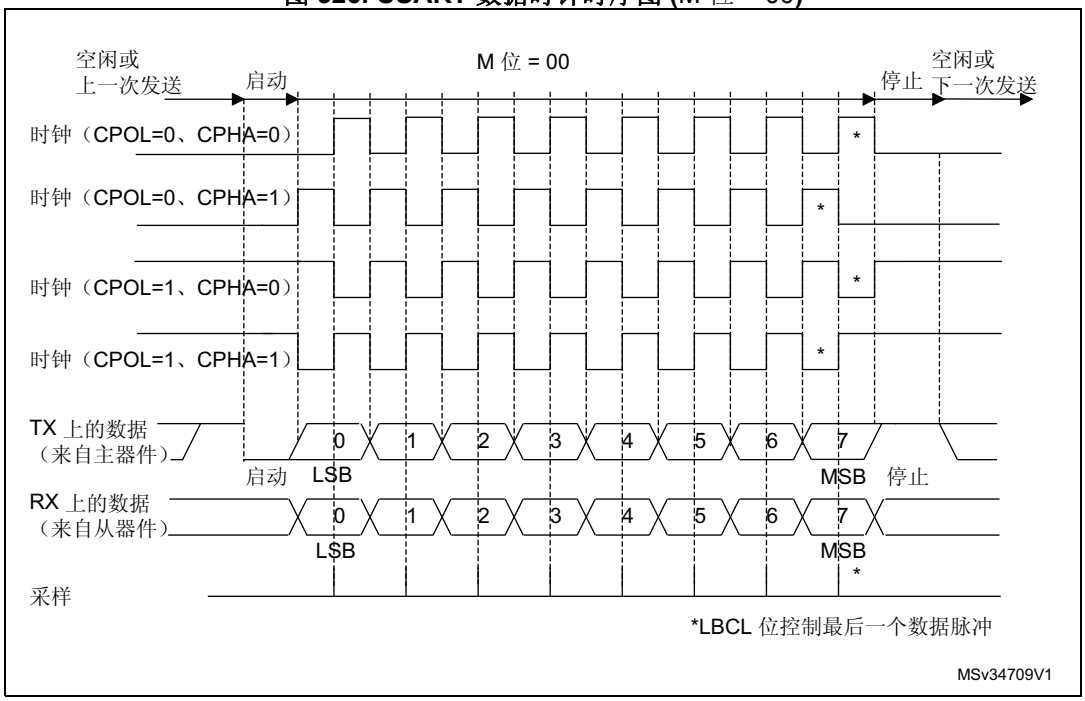


图 327. USART 数据时钟时序图 (M 位 = 01)

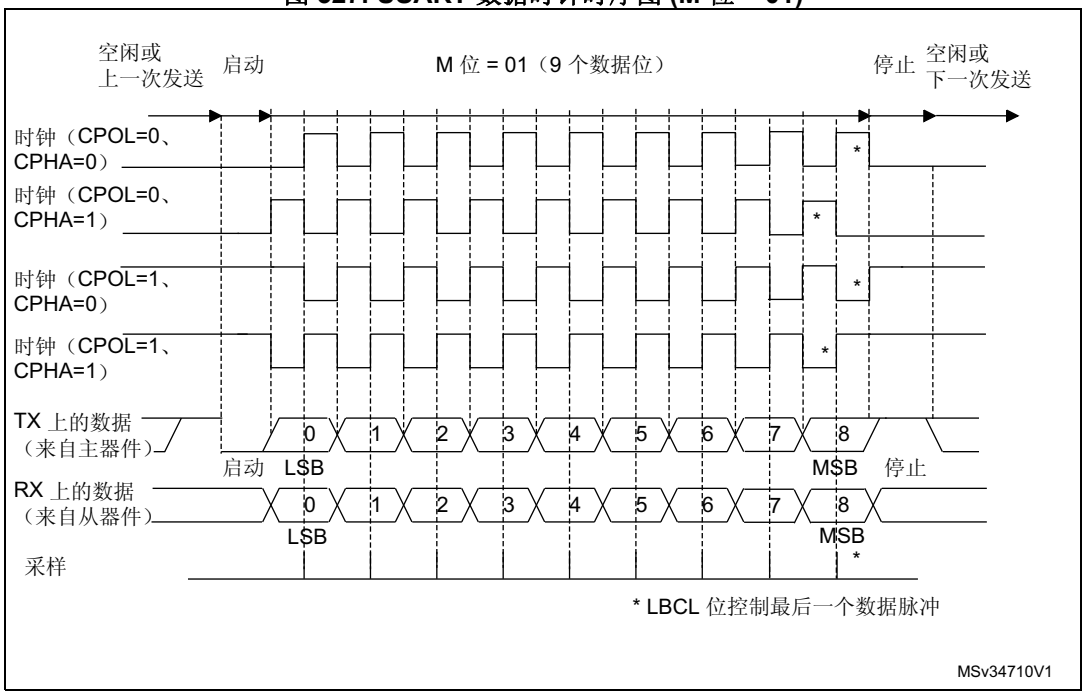
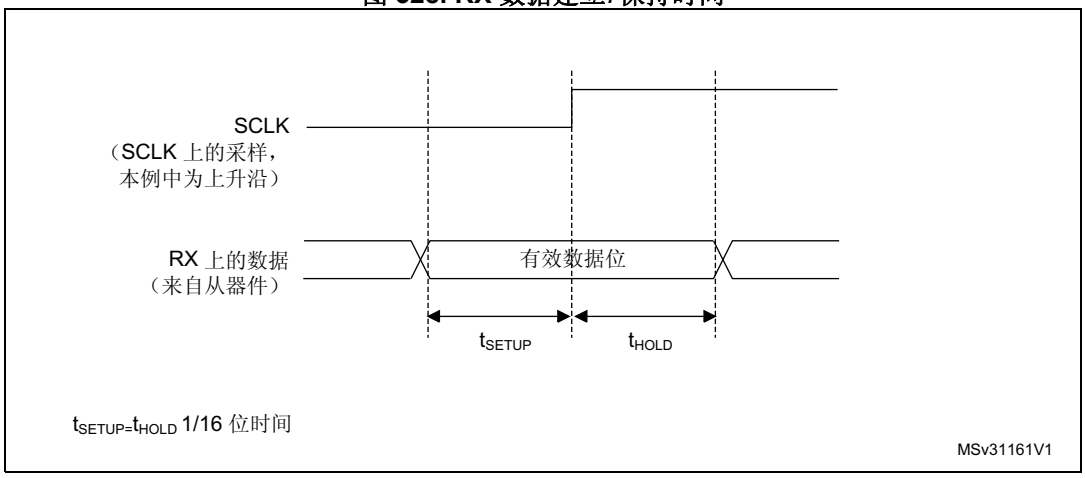


图 328. RX 数据建立/保持时间



注： *在智能卡模式下，SCLK 的功能有所不同。更多详细信息，请参见第 31.5.13 节：智能卡模式。*

31.5.12 单线半双工通信

通过将 USARTx_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：

- USARTx_CR2 寄存器中的 LINEN 位和 CLKEN 位，
- USARTx_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 USARTx_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 USART 模式下的通信协议相似。此线路上的任何冲突必须由软件管理（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封锁，只要数据是在 TE 位置 1 的情况下写入，发送就会持续进行。

31.5.13 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 908 页上的第 31.4 节：USART 实现*。

通过将 USARTx_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：

- USARTx_CR2 寄存器中的 LINEN 位，
- USARTx_CR3 寄存器中的 HDSEL 和 IREN 位。

此外，可能需要将 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 标准的异步协议智能卡。同时支持 T=0（字符模式）和 T=1（块模式）。

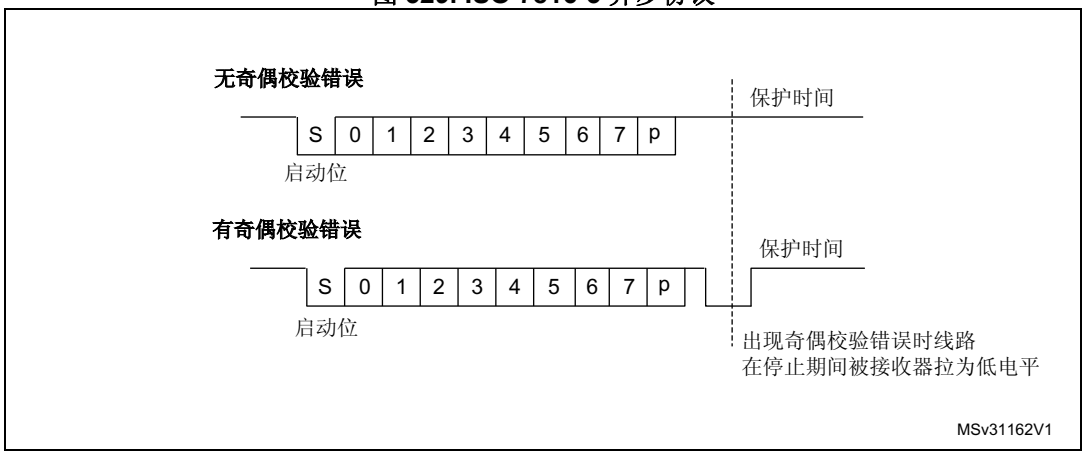
USART 应如下所示进行配置：

- 8 个位加奇偶校验：其中字长度设置为 8 位，USARTx_CR1 寄存器中 PCE=1
- 1.5 个停止位：其中 USARTx_CR2 寄存器中 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 329 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 329. 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 位字段中编程。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。可以使用 USARTx_RQR 寄存器中的 TXFRQ 位将 TXE 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/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 位字段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。

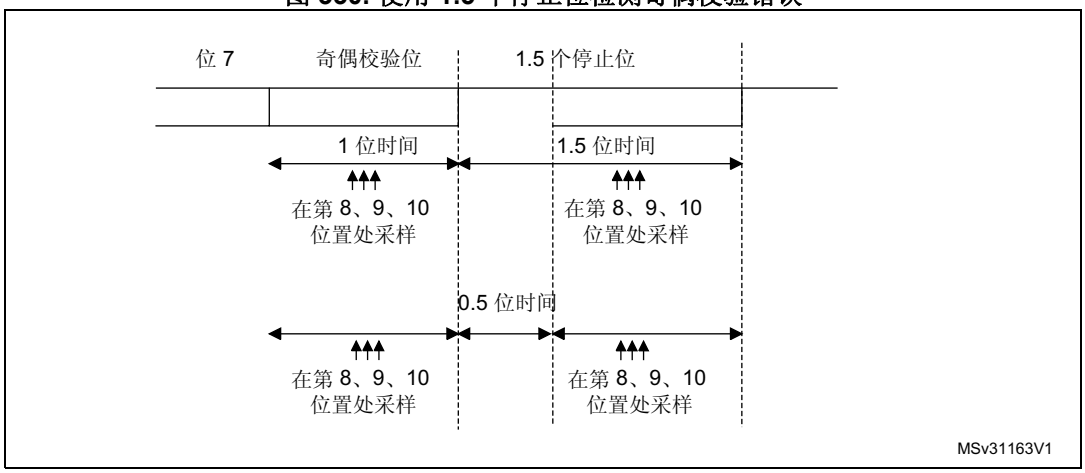
- TC 标志的释放不受智能卡模式的影响。
- 如果在发送端检测到帧错误（由来自接收器的 NACK 信号引起），则发送端的接收块不会将 NACK 作为起始位进行检测。根据 ISO 协议，接收到的 NACK 信号的持续时间可以是 1 或 2 个时钟周期。
- 在接收端，如果检测到奇偶校验错误并发送了 NACK 信号，则接收端不会将 NACK 作为起始位进行检测。

注： *中断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，而非中断。*

当翻转 TE 位时，不会发送空闲帧。空闲帧（在其它配置中进行了定义）在 ISO 协议中未进行定义。

图 330 详细介绍了 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 330. 使用 1.5 个停止位检测奇偶校验错误



USART 可以通过 SCLK 输出为智能卡提供时钟。在智能卡模式下，SCLK 仅通过一个 5 位预分频器由内部外设输入时钟提供。分频比在预分频器寄存器 USARTx_GTPR 中进行配置。SCLK 频率可在 $f_{CK}/2$ 到 $f_{CK}/62$ 之间进行编程，其中 f_{CK} 为外设输入时钟。

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。

在块模式下从智能卡请求读操作时，软件必须通过将 USART_CR2 寄存器中的 RTOEN 位置 1 使能接收器超时功能，并将 RTOR 寄存器中的 RTO 位字段编程为 BWT（块等待时间）- 11 值。如果此时间段后未从智能卡接收到应答，RTOF 标志将置 1，并生成超时中断（如果 USART_CR1 寄存器中的 RTOIE 位置 1）。如果该时间段后接收到第一个字符，则通过 RXNE 中断发出信号指示。

注： *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 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 中断）后，为允许自动检查两个连续字符间的最大等待时间，必须将 RTOR 寄存器中的 RTO 位字段编程为 CWT（字符等待时间）- 11 值。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： *RTO 计数器开始计数：*

- STOP = 00 时，从停止位结束时开始。

- STOP = 10 时，从第二个停止位结束时开始。

- STOP = 11 时，在 STOP 位开始后开始计数 1 个位的持续时间。

按照智能卡协议定义，BWT/CWT 的值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 (TXE=0) 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）中传达。必须将此值编程到 USARTx_RTOR 寄存器中的 BLEN 字段。使用 DMA 模式时，在块开始之前，必须将此寄存器字段编程为最小值 (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： *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： *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，时间分量 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，时间分量 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 的九个时间分量中，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 接收到的字节将为“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 LLL 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。

31.5.14 IrDA SIR ENDEC 模块

仅在支持 IrDA 模式时才涉及本节内容。请参见*第 908 页上的第 31.4 节：USART 实现*。

通过将 USARTx_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：

- USARTx_CR2 寄存器中的 LINEN、STOP 和 CLKEN 位，
- USARTx_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 331*）。

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 Kbps。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 IrDA 解码器的解码数据时），则 IrDA 不会对 USART 发送到 IrDA 的 TX 上的数据进行编码。接收数据时，应避免同时进行发送，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（参见*图 332*）。
- 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。

- IIrDA 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 USARTx_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，USARTx_CR2 寄存器中的 STOP 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 应丢弃持续时间短于 1 个 PSC 周期的脉冲。只有当持续时间大于 2 个 IrDA 低功耗波特时钟周期（USARTx_GTPR 寄存器中的 PSC 值）时，才是有效低电平。

注： *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。*

接收器的建立时间应由软件进行管理。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 331. IrDA SIR ENDEC —— 框图

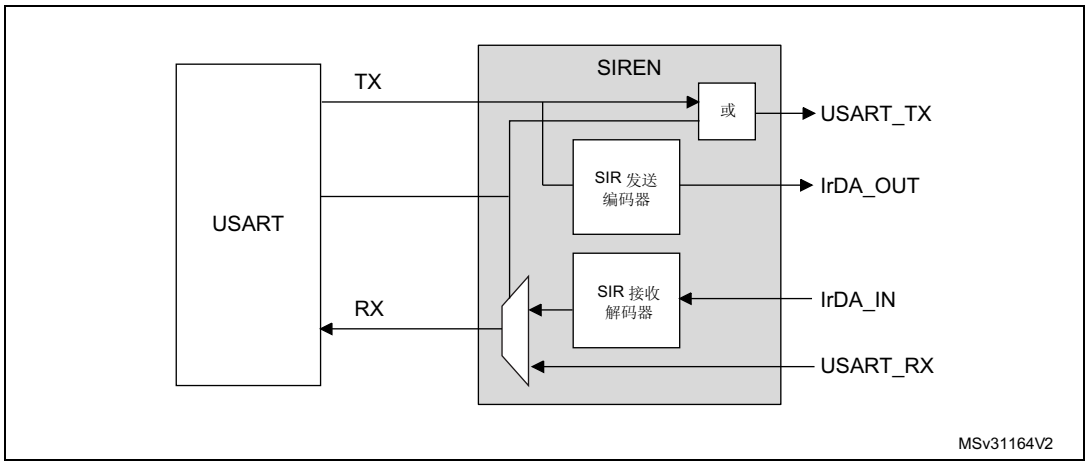
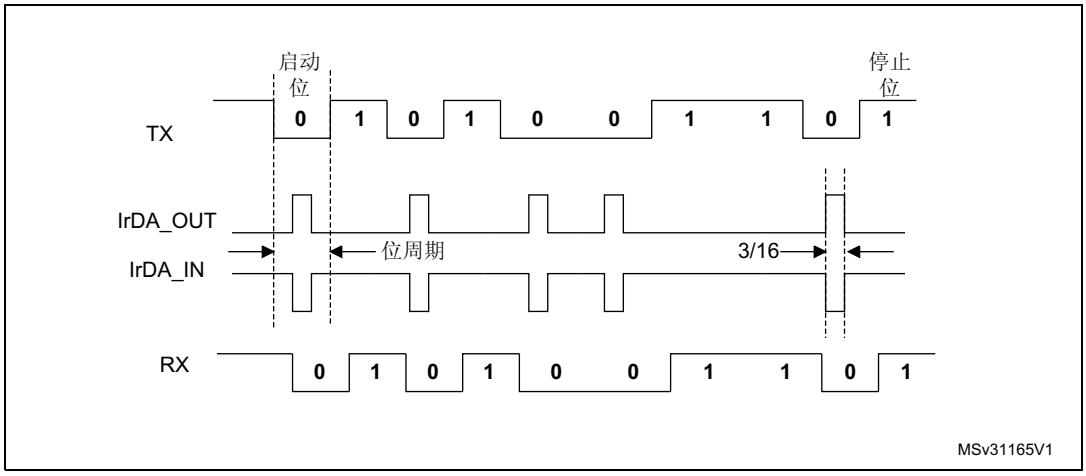


图 332. IrDA 数据调制 (3/16) —— 正常模式



31.5.15 使用 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 请求是独立生成的。

注： *要确定是否支持 DMA 模式，请参见第 908 页上的第 31.4 节：USART 实现。*如果不支持 DMA 模式，请按照*第 31.5.2 节：发送器*或*第 31.5.3 节：接收器*中的*解释说明使用 USART。可以将 USARTx_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

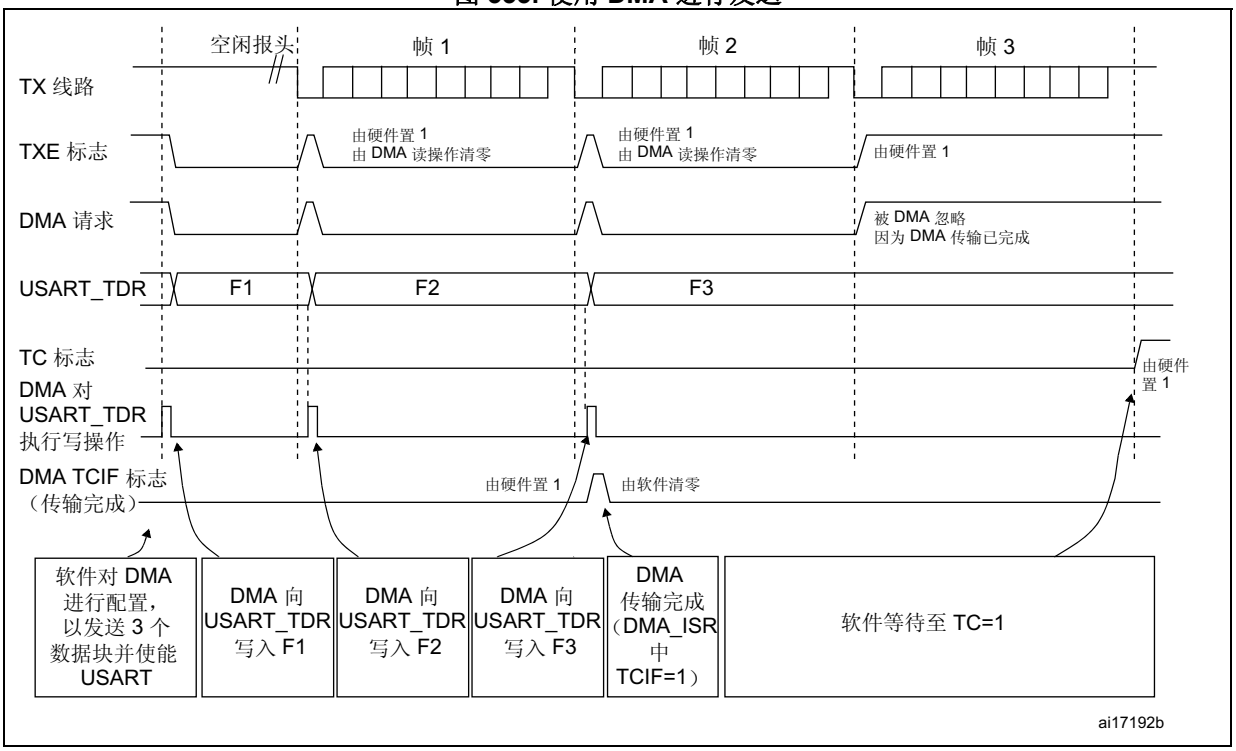
将 USARTx_CR3 寄存器中的 DMAT 位置 1 可以使能 DMA 模式进行发送。当 TXE 位置 1 时，可将数据从 SRAM 区（通过 DMA 外设，请参见*第 206 页上的第 8 节：直接存储器访问控制器 (DMA)*）加载到 USARTx_TDR 寄存器。要映射一个 DMA 通道以进行 USART 发送，请按以下步骤操作（x 表示通道编号）：

1. 在 DMA 控制寄存器中写入 USARTx_TDR 寄存器地址，将其配置为传输的目标地址。每次发生 TXE 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 事件后，数据都从这个存储区域加载到 USARTx_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 USARTx_ICR 寄存器中的 TCCF 位置 1，将 USARTx_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 或进入停止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。

图 333. 使用 DMA 进行发送



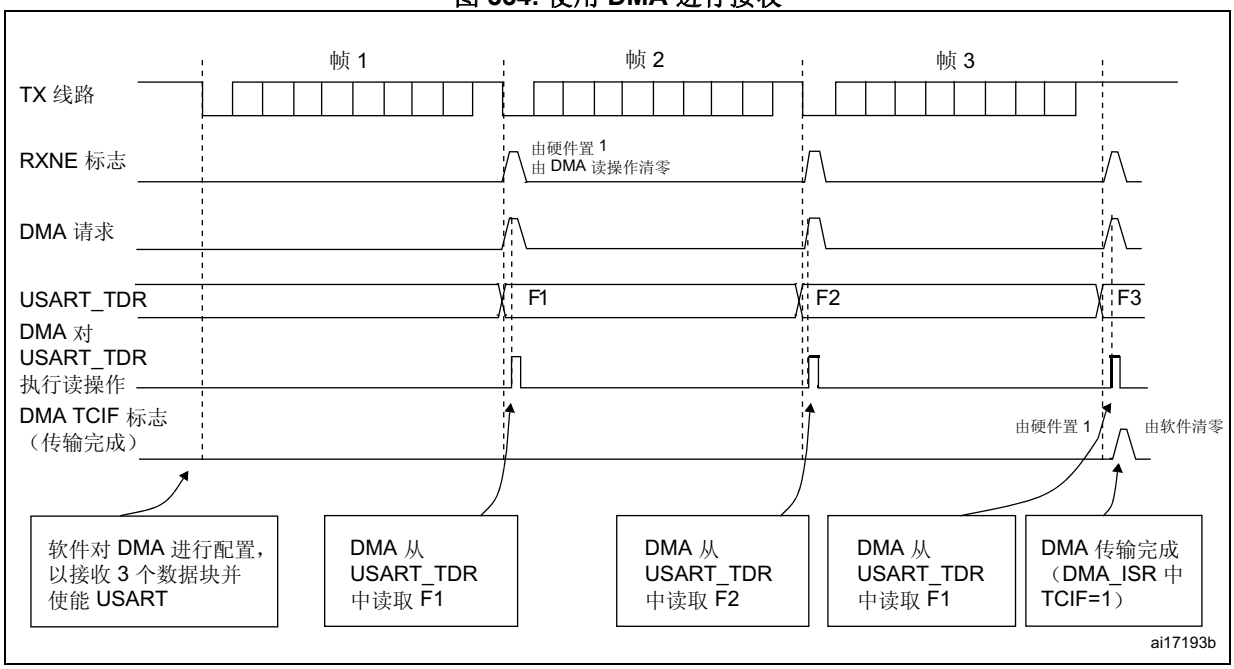
使用 DMA 进行接收

将 USARTx_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，数据会从 USARTx_RDR 寄存器加载到 SRAM 区域中（通过 DMA 外设，请参见*第 8 节：直接存储器访问控制器 (DMA)*）。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 USARTx_RDR 寄存器地址，将其配置为传输的源地址。每次发生 RXNE 事件后，数据都从此地址移动到存储器。
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 事件后，数据都从 USARTx_RDR 寄存器加载到此存储区。
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。

图 334. 使用 DMA 进行接收



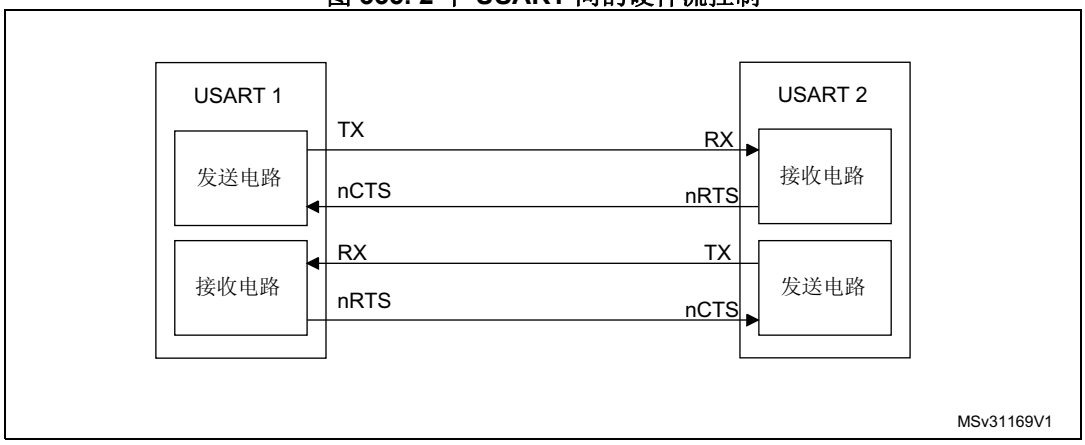
多缓冲区通信中的错误标志和中断生成

在多缓冲区通信中，如果事务中发生任何错误，都会在当前字节后放置错误标志。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 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（USARTx_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，在当前字节后使能中断。

31.5.16 RS232 硬件流控制和 RS485 驱动器使能

使用 nCTS 输入和 nRTS 输出可以控制 2 个器件间的串行数据流。*图 335* 显示了在这种模式下如何连接 2 个器件：

图 335. 2 个 USART 间的硬件流控制

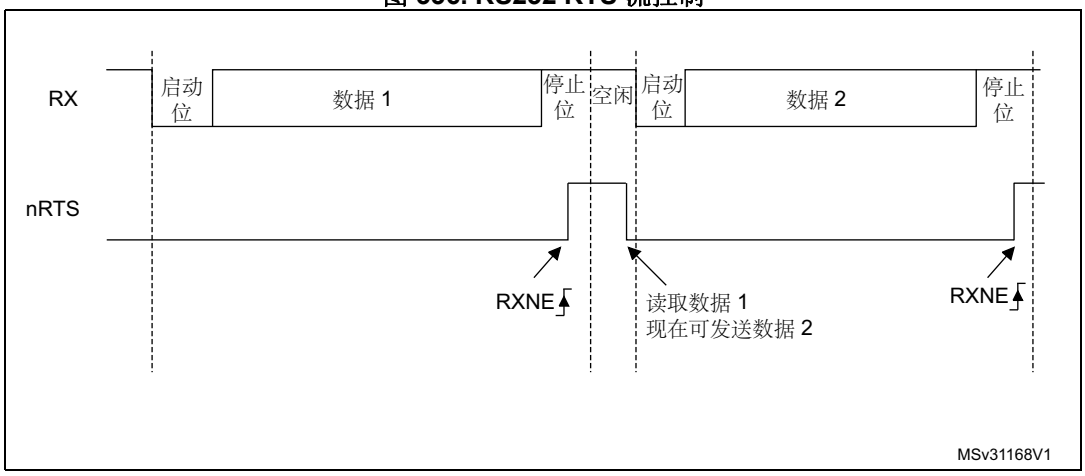


分别向 USARTx_CR3 寄存器中的 RTSE 位和 CTSE 位写入 1，可以分别使能 RS232 RTS 和 CTS 流控制。

RS232 RTS 流控制

如果使能 RTS 流控制 (RTSE=1)，只要 USART 接收器准备好接收新数据，便会将 nRTS 变为有效（连接到低电平）。当接收寄存器已满时，会将 nRTS 变为无效，表明发送过程会在当前帧结束后停止。*图 336* 显示了在使能 RTS 流控制的情况下进行通信的示例。

图 336. RS232 RTS 流控制

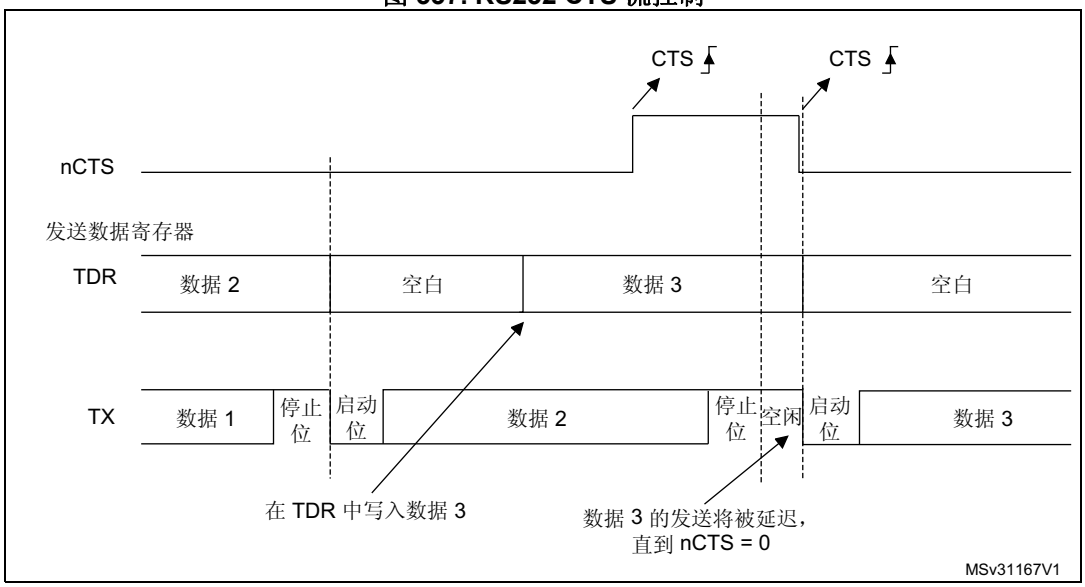


RS232 CTS 流控制

如果使能 CTS 流控制 (CTSE=1)，则发送器会在发送下一帧前检查 nCTS。如果 nCTS 有效（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=0）；否则不会进行发送。如果在发送过程中 nCTS 变为无效，则当前发送完成之后，发送器停止。

当 CTSE=1 时，只要 nCTS 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 USARTx_CR3 寄存器中的 CTSIE 位置 1，则会产生中断。*图 337* 显示了在使能 CTS 流控制的情况下进行通信的示例。

图 337. RS232 CTS 流控制



注： *为正常运行，必须在当前字符结束前至少 3 个 USART 时钟源周期内使能 nCTS。此外还应注意，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 USARTx_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 USARTx_CR1 控制寄存器中的 DEAT [4:0] 位字段编程使能时间。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 USARTx_CR1 控制寄存器中的 DEDT [4:0] 位字段编程禁止时间。DE 信号的极性可使用 USARTx_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 位持续时间，具体取决于过采样速率）。

31.6 USART 低功耗模式

表 172. 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	无影响。
待机	USART 掉电，当器件退出待机模式后必须重新初始化 USART。

31.7 USART 中断

表 173. USART 中断请求

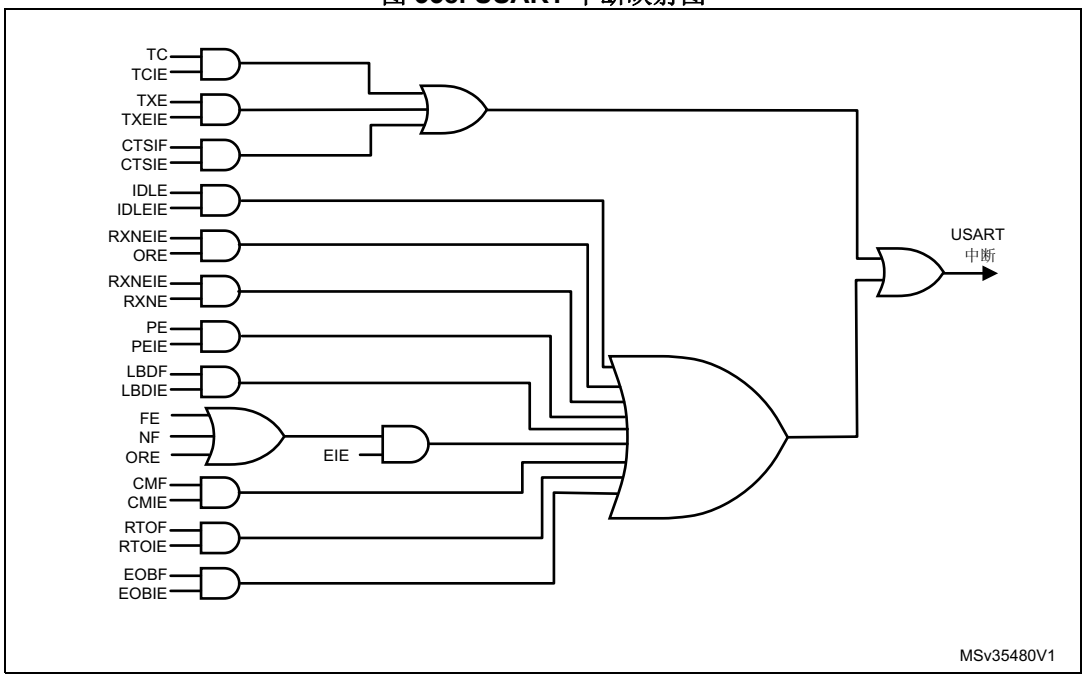
中断事件	事件标志	使能控制位
发送数据寄存器为空	TXE	TXEIE
CTS 中断	CTSIF	CTSIE
发送完成	TC	TCIE
接收数据寄存器不为空（已准备好读取数据）	RXNE	RXNEIE
检测到溢出错误	ORE	
检测到空闲线路	IDLE	IDLEIE
奇偶校验错误	PE	PEIE
LIN 断路	LBDF	LBDIE
多缓冲区通信中的噪声标志、溢出错误和帧错误。	NF 或 ORE 或 FE	EIE
字符匹配	CMF	CMIE
接收器超时错误	RTOF	RTOIE
块结束	EOBF	EOBIE

USART 中断事件被连接到相同的中断向量（请参见*图 338*）。

- 发送期间：发送完成、清除以发送、发送数据寄存器为空或帧错误（智能卡模式下）中断。
- 接收期间：空闲线路检测、上溢错误、接收数据寄存器不为空、奇偶校验错误、LIN 断路检测、噪声标志、帧错误、字符匹配等。

如果相应的使能控制位置 1，则这些事件会生成中断。

图 338. USART 中断映射图



31.8 USART 寄存器

有关寄存器说明中使用的缩写，请参见*第 58 页上的第 1.1 节*。

31.8.1 控制寄存器 1 (USARTx_CR1)

Control register 1

偏移地址：0x00

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	EOBIE	RTOIE	DEAT[4:0]					DEDT[4:0]				
			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
OVER8	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	Res.	UE
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 **M1**：字长 (Word length)

此位和位 12 (M0) 用于确定字长度。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) 时才能写入此位。

注： 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)

此位由软件置 1 和清零。

0：禁止中断

1：USARTx_ISR 寄存器中的 EOBF 标志置 1 时生成 USART 中断

注： 如果 USART 不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 26 **RTOIE**：接收器超时中断使能 (Receiver timeout interrupt enable)

此位由软件置 1 和清零。

0：禁止中断

1：USARTx_ISR 寄存器中的 RTOF 位置 1 时生成 USART 中断。

注： 如果 USART 不支持接收器超时功能，该位保留并由硬件强制清零。第 908 页上的第 31.4 节：USART 实现。

位 25:21 **DEAT[4:0]**：驱动器使能使能时间 (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 位持续时间，具体取决于过采样速率）。

只有在禁止 USART (UE=0) 后才能写入此位字段。

注： 如果不支持驱动器使能功能，该位保留并且必须保持清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 20:16 **DEDT[4:0]**：驱动器使能禁止时间 (Driver Enable de-assertion 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 位持续时间，具体取决于过采样速率）。

如果在 DEDT 时间内对 USARTx_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) 后才能写入此位字段。

注： 如果不支持驱动器使能功能，该位保留并且必须保持清零。请参见第 908 页上的第 31.4 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) 时才能写入此位。

注： 在 LIN 和 IrDA 模式下，此位必须保持清零。

位 14 **CMIE**：字符匹配中断使能 (Character match interrupt enable)

此位由软件置 1 和清零。

0：禁止中断

1：如果 USARTx_ISR 寄存器中的 CMF 位置 1，则生成 USART 中断。

位 13 **MME**：静默模式使能 (Mute mode enable)

此位用于激活 USART 的静默模式功能，此位置 1 时，USART 可按 WAKE 位定义的方式在活动模式与静默模式之间切换。该位由软件置 1 和清零。

0：接收器永久处于活动模式

1：接收器可在静默模式和活动模式之间切换。

位 12 **M0**：字长 (Word length)

此位和位 28 (M1) 用于确定字长度。此位由软件置 1 或清零。请参见位 28 (M1) 的说明。

只有在禁止 USART (UE=0) 时才能写入此位。

位 11 **WAKE**：接收器唤醒方法 (Receiver wakeup method)

此位用于确定 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) 后才能写入此位字段。

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。此位由软件置 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) 后才能写入此位字段。

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。将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) 后才能写入此位字段。

位 8 **PEIE**：PE 中断使能 (PE interrupt enable)

此位由软件置 1 和清零。

0：禁止中断

1：当 USARTx_ISR 寄存器中 PE=1 时，生成 USART 中断

位 7 **TXEIE**：中断使能 (interrupt enable)

此位由软件置 1 和清零。

0：禁止中断

1：当 USARTx_ISR 寄存器中 TXE=1 时，生成 USART 中断

位 6 **TCIE**：传输完成中断使能 (Transfer complete interrupt enable)

此位由软件置 1 和清零。

0：禁止中断

1：当 USARTx_ISR 寄存器中的 TC=1 时，生成 USART 中断

位 5 **RXNEIE**：RXNE 中断使能 (RXNE interrupt enable)

此位由软件置 1 和清零。

0：禁止中断

1：当 USARTx_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)

此位由软件置 1 和清零。

0：禁止中断

1：当 USARTx_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： *除了在智能卡模式下以外，传送期间 TE 位上的“0”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 USARTx_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 保留，必须保持复位值。

位 0 **UE**：USART 使能 (USART enable)

此位清零后，USART 预分频器和输出将立即停止，并丢弃当前操作。USART 的配置保留，但 USARTx_ISR 寄存器中的所有状态标志将设置为其默认值。此位由软件置 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： *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 USARTx_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，因必须在复位 UE 位前禁止 DMA 通道。

31.8.2 控制寄存器 2 (USARTx_CR2)

Control register 2

偏移地址：0x04

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD[7:4]				ADD[3:0]				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	Res.	Res.	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw				

位 31:28 **ADD[7:4]**：USART 节点的地址 (Address of the USART node)

此位字段用于指定 USART 节点的地址或要识别的字符代码。

此位字段在多处理器通信时于静默模式下使用，以通过 7 位地址标记检测进行唤醒。发送器发送字符的 MSB 应为 1。此位字段还可用于正常接收和静默模式无效时的字符检测（例如，ModBus 协议中的块结束检测）。这种情况下，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。

仅在禁止接收 (RE = 0) 或禁止 USART (UE=0) 时才能写入该位字段

位 27:24 **ADD[3:0]**：USART 节点的地址 (Address of the USART node)

此位字段用于指定 USART 节点的地址或要识别的字符代码。

此位字段在多处理器通信时于静默模式下使用，以通过地址标记检测进行唤醒。

仅在禁止接收 (RE = 0) 或禁止 USART (UE=0) 时才能写入该位字段

位 23 **RTOEN**：接收器超时使能 (Receiver timeout enable)

此位由软件置 1 和清零。

0：禁止接收器超时功能。

1：使能接收器超时功能。

使能此功能后，如果 RX 线路在 RTOR（接收器超时寄存器）中编程的持续时间内处于空闲状态（无接收），则 USARTx_ISR 寄存器中的 RTOF 标志置 1。

注：　如果 USART 不支持接收器超时功能，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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) 时才能写入该位字段。

注：　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，则此位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 20 **ABREN**：自动波特率使能 (Auto baud rate enable)

此位由软件置 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：　如果 USART 不支持自动波特率功能，则此位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 19 **MSBFIRST**：最高有效位在前 (Most significant bit first)

此位由软件置 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 7/8/9）在前，后跟起始位。

只有在禁止 USART (UE=0) 后才能写入此位字段。

位 18 **DATAINV**：二进制数据反向 (Binary data inversion)

此位由软件置 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) 后才能写入此位字段。

位 17 **TXINV**：TX 引脚有效电平反向 (TX pin active level inversion)

此位由软件置 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) 后才能写入此位字段。

位 16 **RXINV**：RX 引脚有效电平反向 (RX pin active level inversion)

此位由软件置 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) 后才能写入此位字段。

位 15 **SWAP**：交换 TX/RX 引脚 (Swap TX/RX pins)

此位由软件置 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 USART 的交叉连接时工作。

只有在禁止 USART (UE=0) 后才能写入此位字段。

位 14 **LINEN**：LIN 模式使能 (LIN mode enable)

此位由软件置 1 和清零。

0：禁止 LIN 模式

1：使能 LIN 模式

LIN 模式可以使用 USARTx_RQR 寄存器中的 SBKRQ 位发送 LIN 同步断路（13 个低位），并可检测 LIN 同步断路。

只有在禁止 USART (UE=0) 后才能写入此位字段。

注：　如果 USART 不支持 LIN 模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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) 后才能写入此位字段。

位 11 **CLKEN**：时钟使能 (Clock enable)

该位允许用户使能 SCLK 引脚。

0：禁止 SCLK 引脚

1：使能 SCLK 引脚

只有在禁止 USART (UE=0) 时才能写入此位。

注：　如果既不支持同步模式，也不支持智能卡模式，则该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

注：　为了向智能卡正确提供 SCLK 时钟，必须按以下步骤操作：

- UE = 0

- SCEN = 1

- GTPR 配置（如果需要配置 PSC，建议通过单次访问 USARTx_ GTPR 寄存器来同时配置 PSC 和 GT）

- 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 SCLK 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 SCLK 引脚为低电平

1：空闲时 SCLK 引脚为高电平

只有在禁止 USART (UE=0) 时才能写入此位。

注：　如果不支持同步模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 9 **CPHA**：时钟相位 (Clock phase)

此位用于在同步模式下选择 SCLK 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 326 和图 327）

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) 时才能写入此位。

注：　如果不支持同步模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 8 **LBCL**：最后一个位时钟脉冲 (Last bit clock pulse)

此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 SCLK 引脚上输出。

0：最后一个数据位的时钟脉冲不在 SCLK 引脚上输出

1：最后一个数据位的时钟脉冲在 SCLK 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 USARTx_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) 时才能写入此位。

注：　如果不支持同步模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 7 保留，必须保持复位值。

位 6 **LBDIE**：LIN 断路检测中断使能 (LIN break detection interrupt enable)

断路中断屏蔽（使用断路分隔符进行断路检测）

0：禁止中断

1：当 USARTx_ISR 寄存器中 LBDF = 1 时，生成中断

注： 如果 LIN 不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 5 **LBDL**：LIN 断路检测长度 (LIN break detection length)

该位用于选择 11 位断路检测或 10 位断路检测。

0：10 位断路检测

1：11 位断路检测

只有在禁止 USART (UE=0) 时才能写入此位。

注： 如果 LIN 不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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)

此位用于选择 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： 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。

注： 使能发送器时不应对这 3 个位（CPOL、CPHA、LBCL）进行写操作。

31.8.3 控制寄存器 3 (USARTx_CR3)

Control register 3

偏移地址：0x08

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		SCARCNT2:0]			Res.
												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	HDSEL	IRLP	IREN	EIE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	V	V	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

位 23:20 保留，必须保持复位值。

位 19:17 **SCARCNT[2:0]**：智能卡自动重试计数 (Smartcard auto-retry count)

此位字段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，此位字段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，此位字段用于指定生成接收错误（RXNE 位和 PE 位置 1）前错误接收尝试的次数。

只有在禁止 USART (UE=0) 时才能编程此位字段。

使能 USART (UE=1) 时，此位字段只能写入 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： 如果不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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) 时才能写入此位。

注： 如果不支持驱动器使能功能，该位保留并且必须保持清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 14 **DEM**：驱动器使能模式 (Driver enable mode)

此位用于通过 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) 时才能写入此位。

注： 如果不支持驱动器使能功能，该位保留并且必须保持清零。第 908 页上的第 31.4 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 清零，然后再将错误标志清零。

只有在禁止 USART (UE=0) 时才能写入此位。

注： 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)

此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 USARTx_RDR 寄存器之前的内容。

只有在禁止 USART (UE=0) 时才能写入此位。

注： 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 (NF)。

0：三个采样位方法

1：一个采样位方法

只有在禁止 USART (UE=0) 时才能写入此位。

位 10 **CTSIE**：CTS 中断使能 (CTS interrupt enable)

0：禁止中断

1：当 USARTx_ISR 寄存器中的 CTSIF=1 时，生成中断

注：　如果不支持硬件流控制功能，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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 nCTS 输入有效（连接到 0）时才发送数据。如果在发送数据时使 nCTS 输入无效，会在停止之前完成发送。如果使 nCTS 无效时数据已写入数据寄存器，则将延迟发送，直到 nCTS 有效。

只有在禁止 USART (UE=0) 时才能写入该位

注：　如果不支持硬件流控制功能，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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 nRTS 输出有效（拉至 0）。

只有在禁止 USART (UE=0) 时才能写入此位。

注：　如果不支持硬件流控制功能，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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) 后才能写入此位字段。

注：　如果 USART 不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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) 后才能写入此位字段。

注：　如果 USART 不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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) 时才能写入此位。

位 2 **IRLP**：IrDA 低功耗模式 (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) 时才能写入此位。

注： 如果不支持 IrDA 模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 1 **IREN**：IrDA 模式使能 (IrDA mode enable)

此位由软件置 1 和清零。

0：禁止 IrDA

1：使能 IrDA

只有在禁止 USART (UE=0) 时才能写入此位。

注： 如果不支持 IrDA 模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

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（USARTx_ISR 寄存器中 FE = 1 或 ORE = 1 或 NF = 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

USARTx_ISR 寄存器中的 FE=1 或 ORE=1 或 NF=1 时生成中断。

31.8.4 波特率寄存器 (USARTx_BRR)

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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 15:4 **BRR[15:4]**

BRR[15:4] = USARTDIV[15:4]

位 3:0 **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。

31.8.5 保护时间和预分频器寄存器 (USARTx_GTPR)

Guard time and prescaler register

偏移地址：0x10

复位值：0x0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GT[7:0]								PSC[7:0]							
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)

此位字段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) 后才能写入此位字段。

注： 如果不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 7:0 **PSC[7:0]**：预分频器值 (Prescaler value)

在 IrDA 低功耗和正常的 IrDA 模式下：

PSC[7:0] = IrDA 正常和低功耗波特率

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

...

在智能卡模式下：

PSC[4:0]：预分频器值 (Prescaler value)

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) 后才能写入此位字段。

注： 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，该位字段保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

31.8.6 接收器超时寄存器 (USARTx_RTOR)

Receiver timeout register

偏移地址：0x14

复位值：0x0000

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

此位字段用于提供智能卡 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 时复位。

此位字段也可用于其它模式。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： 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)

此位字段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。

这种情况下，从接收到的最后一个字符的起始位开始执行超时测量。

注： 每个接收到的字符只能对此值编程一次。

注： 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 908 页上的第 31.4 节：USART 实现。

31.8.7 请求寄存器 (USARTx_RQR)

Request register

偏移地址：0x18

复位值：0x0000

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

向该位写入 1 时会将 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 USARTx_ISR 寄存器中的 FE 标志有效时，只能在智能卡模式下使用此位。

如果 USART 不支持智能卡模式，该位保留并由硬件强制清零。请参见*第 908 页上的第 31.4 节：USART 实现*。

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)

向此位写入 1 可将 USART 置于静默模式，并将 RWU 标志置 1。

位 1 **SBKRQ**：发送断路请求 (Send break request)

向此位写入 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： *这种情况下，应用需要在之前插入的所有数据（包括尚未发送的数据）后发送中断字符，软件应等到 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)

向此位写入 1 可复位 USARTx_ISR 寄存器中的 ABRF 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： *如果 USART 不支持自动波特率功能，则此位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。*

31.8.8 中断和状态寄存器 (USARTx_ISR)

Interrupt and status register

偏移地址：0x1C

复位值：0x0200 00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TEACK	Res.	Res.	SBKF	CMF	BUSY
										r			r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABRF	ABRE	Res.	EOBF	RTOF	CTS	CTSIF	LBDF	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
r	r		r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:25 保留，必须保持复位值。

位 24:22 保留，必须保持复位值。

位 21 **TEACK**：发送使能确认标志 (Transmit enable acknowledge flag)

USART 采用发送使能值时，通过硬件将此位置 1/复位。

通过写入 TE=0 生成空闲帧请求，然后在 USARTx_CR1 寄存器中写入 TE=1 以遵循 TE=0 最短周期时，可使用此位。

位 20:19 保留，必须保持复位值。

位 18 **SBKF**：发送断路标志 (Send break flag)

此位指示已请求发送断路字符。通过将 1 写入 USARTx_RQR 寄存器中的 SBKRQ 位，此位由软件置 1。此位在断路发送的停止位期间由硬件自动复位。

0：不发送断路字符

1：将发送断路字符

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此位置 1。通过向 USARTx_ICR 寄存器中的 CMCF 写入 1，此位由软件清零。

如果 USARTx_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)

此位由硬件置 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 也将置 1，并在 RXNEIE = 1 时生成中断），或者自动波特率操作未成功完成时，此位由硬件置 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 USARTx_RQR 寄存器中的 ABRRQ 写入 1，此位由软件清零。

注： 如果 USART 不支持自动波特率功能，则此位保留并由硬件强制清零。

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），此位由硬件置 1。

通过将 1 写入 USARTx_CR3 寄存器中的 ABRRQ 位，此位由软件清零。

注： 如果 USART 不支持自动波特率功能，则此位保留并由硬件强制清零。

位 13 保留，必须保持复位值。

位 12 **EOBF**：块结束标志 (End of block flag)

接收到完整块后，此位由硬件置 1（例如 T=1 智能卡模式）。接收到的字节数（从块的起始处，包括起始字段）等于或大于 BLEN + 4 时执行检测。

如果 USARTx_CR2 寄存器中 EOBIE = 1，则会生成中断。

通过向 USARTx_ICR 寄存器中的 EOBCF 写入 1，此位由软件清零。

0：未达到块结束

1：已达到块结束（字符数）

注：　如果不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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，此位由硬件置 1。通过向 USARTx_ICR 寄存器中的 RTOCF 写入 1，此位由软件清零。

如果 USARTx_CR2 寄存器中 RTOIE=1，则会生成中断。

在智能卡模式下，该超时对应于 CWT 或 BWT 时间。

0：未达到超值值

1：已达到超时值，未接收到任何数据

注：　如果 RTOR 寄存器中编程的时间值将 2 个字符隔开，则 RTOF 不置 1。如果此时间大于该值 + 2 个采样时间（2/16 或 2/8，具体取决于过采样方法），则 RTOF 标志置 1。

即使 RE = 0，计数器仍会计数，但 RTOF 仅在 RE = 1 时置 1。如果 RE 置 1 时已经超时，则 RTOF 将置 1。

如果 USART 不支持接收器超时功能，该位保留并由硬件强制清零。

位 10 **CTS**：CTS 标志 (CTS flag)

此位由硬件置 1/复位。此位是对 nCTS 输入引脚的状态取反。

0：nCTS 线置 1

1：nCTS 线复位

注：　如果不支持硬件流控制功能，该位保留并由硬件强制清零。

位 9 **CTSIF**：CTS 中断标志 (CTS interrupt flag)

如果 CTSE 位置 1，当 nCTS 输入切换时，此位由硬件置 1。通过向 USARTx_ICR 寄存器中的 CTSCF 位写入 1，此位由软件清零。

如果 USARTx_CR3 寄存器中 CTSIE=1，则会生成中断。

0：nCTS 状态线上未发生变化

1：nCTS 状态线上发生变化

注：　如果不支持硬件流控制功能，该位保留并由硬件强制清零。

位 8 **LBDF**：LIN 断路检测标志 (LIN break detection flag)

检测到 LIN 断路时，该位由硬件置 1。通过向 USARTx_ICR 寄存器中的 LBDCF 写入 1，此位由软件清零。

如果 USARTx_CR2 寄存器中 LBDIE = 1，则会生成中断。

0：未检测到 LIN 断路

1：检测到 LIN 断路

注：　如果 USART 不支持 LIN 模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 7 **TXE**：发送数据寄存器为空 (Transmit data register empty)

当 USARTx_TDR 寄存器的内容已传输到移位寄存器时，该位由硬件置 1。通过对 USARTx_TDR 寄存器执行写入操作将该位清零。

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 USARTx_RQR 寄存器中的 TXFRQ 写入 1 来将 TXE 标志清零。

如果 USARTx_CR1 寄存器中 TXEIE 位 = 1，则会生成中断。

0：数据未传输到移位寄存器

1：数据传输到移位寄存器

注： 单缓冲区发送期间使用该位。

位 6 **TC**：发送完成 (Transmission complete)

如果已完成对包含数据的帧的发送并且 TXE 置 1，则该位由硬件置 1。如果 USARTx_CR1 寄存器中 TCIE=1，则会生成中断。通过向 USARTx_ICR 寄存器中的 TCCF 写入 1 或向 USARTx_TDR 寄存器执行写操作，此位由软件清零。

如果 USARTx_CR1 寄存器中 TCIE=1，则会生成中断。

0：传送未完成

1：传送已完成

注： 如果 TE 位复位且无任何发送正在进行，TC 位会立即置 1。

位 5 **RXNE**：读取数据寄存器不为空 (Read data register not empty)

当 RDR 移位寄存器的内容已传输到 USARTx_RDR 寄存器时，该位由硬件置 1。通过对 USARTx_RDR 寄存器执行读入操作将该位清零。也可以通过将 USARTx_RQR 寄存器中的 RXFRQ 位置 1 将 RXNE 标志位清零。

如果 USARTx_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 USARTx_CR1 寄存器中 IDLEIE=1，则会生成中断。通过向 USARTx_ICR 寄存器中的 IDLECF 写入 1，此位由软件清零。

0：未检测到空闲线路

1：检测到空闲线路

注： 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**：溢出错误 (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 RDR 寄存器时，该位由硬件置 1。通过向 USARTx_ICR 寄存器中的 ORECF 写入 1，此位由软件清零。

如果 USARTx_CR1 寄存器中 RXNEIE=1 或 EIE = 1，则会生成中断。

0：无溢出错误

1：检测到溢出错误

注： 当此位置 1 时，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。

USARTx_CR3 寄存器中的 OVRDIS 位置 1 时，此位将被永久强制清零（无上溢检测）。

位 2 **NF**：START 位噪声检测标志 (START bit 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 USARTx_ICR 寄存器中的 NFCF 写入 1，此位由软件清零。

0：未检测到噪声

1：检测到噪声

注： 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 NF 标志置 1，则会生成中断。

注： 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 NF 标志（请参见第 923 页上的第 31.5.5 节：USART 接收器对时钟偏差的容差）。

位 1 **FE**：帧错误 (Framing error)

当检测到去同步化、过度的噪声或中断字符时，该位由硬件置 1。通过向 USARTx_ICR 寄存器中的 FECF 写入 1，此位由软件清零。

在智能卡模式下发送期间，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），则此位置 1。

如果 USARTx_CR1 寄存器中 EIE = 1，则会生成中断。

0：未检测到帧错误

1：检测到帧错误或中断字符

位 0 **PE**：奇偶校验错误 (Parity error)

当在接收器模式下发生奇偶校验错误时，该位由硬件置 1。通过向 USARTx_ICR 寄存器中的 PECF 写入 1，此位由软件清零。

如果 USARTx_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

31.8.9 中断标志清零寄存器 (USARTx_ICR)

Interrupt flag clear register

偏移地址：0x20

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMCF	Res.
														w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	EOBCF	RTOCF	Res.	CTSCF	LBDCF	Res.	TCCF	Res.	IDLECF	ORECF	NCF	FECF	PECF
			w	w		w	w		w		w	w	w	w	w

位 31:20 保留，必须保持复位值。

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 写入此位时，USARTx_ISR 寄存器中 CMF 标志将清零。

位 16:13 保留，必须保持复位值。

位 12 **EOBCF**：块结束清零标志 (End of block clear flag)
将 1 写入此位时，USARTx_ISR 寄存器中 EOBF 标志将清零。
注：　如果 USART 不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 11 **RTOCF**：接收器超时清零标志 (Receiver timeout clear flag)
将 1 写入此位时，USARTx_ISR 寄存器中 RTOF 标志将清零。
注：　如果 USART 不支持接收器超时功能，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 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 写入此位时，USARTx_ISR 寄存器中 CTSIF 标志将清零。
注：　如果不支持硬件流控制功能，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

位 8 **LBDCF**：LIN 中断检测清零标志 (LIN break detection clear flag)
将 1 写入此位时，USARTx_ISR 寄存器中 LBDF 标志将清零。
注：　如果 LIN 不支持智能卡模式，该位保留并由硬件强制清零。请参见第 908 页上的第 31.4 节：USART 实现。

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 写入此位时，USARTx_ISR 寄存器中 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 写入此位时，USARTx_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 写入此位时，USARTx_ISR 寄存器中 ORE 标志将清零。

位 2 **NCF**：检测到噪声清零标志 (Noise detected clear flag)
将 1 写入此位时，USARTx_ISR 寄存器中 NF 标志将清零。

位 1 **FECF**：帧错误清零标志 (Framing error clear flag)
将 1 写入此位时，USARTx_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 写入此位时，USARTx_ISR 寄存器中 PE 标志将清零。

31.8.10 接收数据寄存器 (USARTx_RDR)

Receive data register

偏移地址：0x24

复位值：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
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 314*）。
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。

31.8.11 发送数据寄存器 (USARTx_TDR)

Transmit data register

偏移地址：0x28

复位值：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
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 314*）。
在使能奇偶校验位的情况下（USARTx_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： 只能在 TXE=1 时写入此寄存器。

31.8.12 USART 寄存器映射

下表提供了 USART 寄存器映射和复位值。

表 174. USART 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	**USARTx_CR1**	Res.	Res.	Res.	M1	EOBIE	RTOIE	DEAT4	DEAT3	DEAT2	DEAT1	DEAT0	DEDT4	DEDT3	DEDT2	DEDT1	DEDT0	OVER8	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	Res.	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
0x04	**USARTx_CR2**	ADD[7:4]				ADD[3:0]				RTOEN	ABRMOD1	ABRMOD0	ABREN	MSBFIRST	DATAINV	TXINV	RXINV	SWAP	LINEN	STOP [1:0]		CLKEN	CPOL	CPHA	LBCL	Res.	LBDIE	LBDL	ADDM7	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		0	0	0				

表 174. USART 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	**USARTx_CR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		SCARCNT2:0]			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
0x0C	**USARTx_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	**USARTx_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	**USARTx_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
0x18	**USARTx_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	**USARTx_ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TEACK	Res.	Res.	SBKF	CMF	BUSY	ABRF	ABRE	Res.	EOBF	RTOF	CTS	CTSIF	LBDF	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
	Reset value											0			0	0	0	0	0		0	0	0	0	0	1	1	0	0	0	0	0	0
0x20	**USARTx_ICR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMCF	Res.	Res.	Res.	Res.	EOBCF	RTOCF	Res.	CTSCF	LBDCF	Res.	TCCF	Res.	IDLECF	ORECF	NCF	FECF	PECF
	Reset value															0					0	0		0	0		0		0	0	0	0	0
0x24	**USARTx_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																								X	X	X	X	X	X	X	X	X
0x28	**USARTx_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																								X	X	X	X	X	X	X	X	X

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2 节*。

32 串行外设接口/集成电路内置音频总线 (SPI/I2S)

32.1 简介

SPI/I²S 接口可用于使用 SPI 协议或 I^2S 音频协议与外部器件进行通信。SPI 或 I^2S 模式可通过软件进行选择。器件复位后默认选择 SPI Motorola 模式。

串行外设接口 (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，在这种情况下，它可为外部从器件提供通信时钟 (SCK)。该接口还能够在多主模式配置下工作。

集成电路内置音频总线（Inter-IC sound，I^2S）也是同步串行通信接口。它可以在从模式或主模式下进行半双工通信。它可满足四种不同音频标准的要求，包括 Philips I^2S 标准、MSB 和 LSB 对齐标准以及 PCM 标准。

32.2 SPI 主要特性

- 主模式或从模式操作
- 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
- 数据大小可选择 4 位到 16 位
- 多主模式功能
- 8 个主模式波特率预分频器，可达 $f_{PCLK}/2$
- 从模式频率可达 $f_{PCLK}/2$
- 对于主模式和从模式都可通过硬件或软件进行 NSS 管理：动态切换主/从操作
- 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
- 可触发中断的专用发送和接收标志
- SPI 总线忙状态标志
- 支持 SPI Motorola 模式
- 用于确保可靠通信的硬件 CRC 功能：
  - 在发送模式下可将 CRC 值作为最后一个字节发送
  - 根据收到的最后一个字节自动进行 CRC 错误校验
- 可触发中断的主模式故障和上溢标志
- CRC 错误标志
- 具有 DMA 功能的两个 32 位内置 Rx 和 Tx FIFO
- 支持 SPI TI 模式

32.3 I2S 主要特性

- 半双工通信（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，可实现精确的音频采样频率（从 8 kHz 到 192 kHz）
- 数据格式可以是 16 位、24 位或 32 位
- 数据包帧由音频通道固定为 16 位（可容纳 16 位数据帧）或 32 位（可容纳 16 位、24 位、32 位数据帧）
- 可编程的时钟极性（就绪时的电平状态）
- 从发送模式下的下溢标志、接收模式下的上溢标志（主模式和从模式），以及接收和发送模式下的帧错误标志（仅从模式）
- 发送和接收使用同一个 16 位数据寄存器
- 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（在 16 位通道帧或扩展为 32 位通道帧的 16 位数据帧上进行短帧和长帧同步）
- 数据方向始终为 MSB 在前
- 用于发送和接收的 DMA 功能（16 位宽）
- 可输出主时钟以驱动外部音频元件。比率固定为 256 × F_S（其中 F_S 为音频采样频率）

32.4 SPI/I2S 实现

本手册介绍了 SPI/I2S 在器件 STM32F75xxx 和 STM32F74xxx 中的实现方案。

表 175. STM32F75xxx 和 STM32F74xxx SPI 实现

SPI 特性(1)	SPI1	SPI2	SPI3	SPI4	SPI5	SPI6
硬件 CRC 计算	X	X	X	X	X	X
Rx/Tx FIFO	X	X	X	X	X	X
NSS脉冲模式	X	X	X	X	X	X
I2S 模式	X	X	X	-	-	-
TI 模式	X	X	X	X	X	X

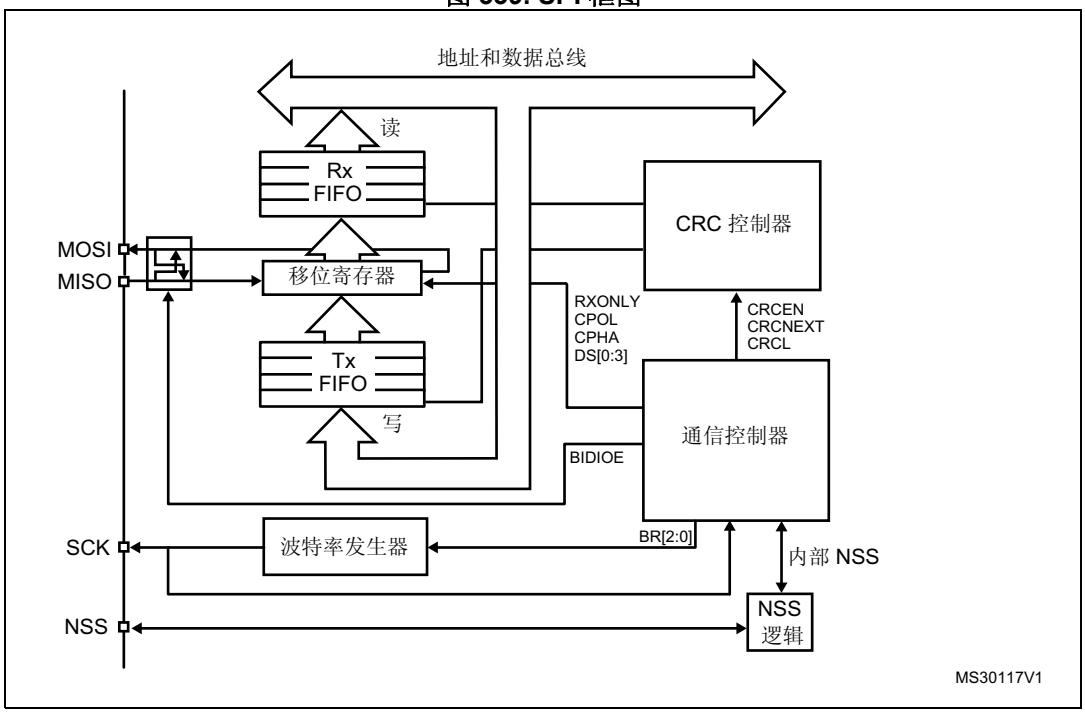
1. X = 支持。

32.5 SPI 功能说明

32.5.1 一般说明

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。SPI 的主要组件及其交互方式如以下框图所示*图 339*。

图 339. SPI 框图



四个 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。
- **NSS**：从器件选择引脚。根据 SPI 和 NSS 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 32.5.4 节：从器件选择 (NSS) 引脚管理*。

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。其它信号可以根据 SPI 节点间的数据交换及其从器件选择信号管理进行添加。

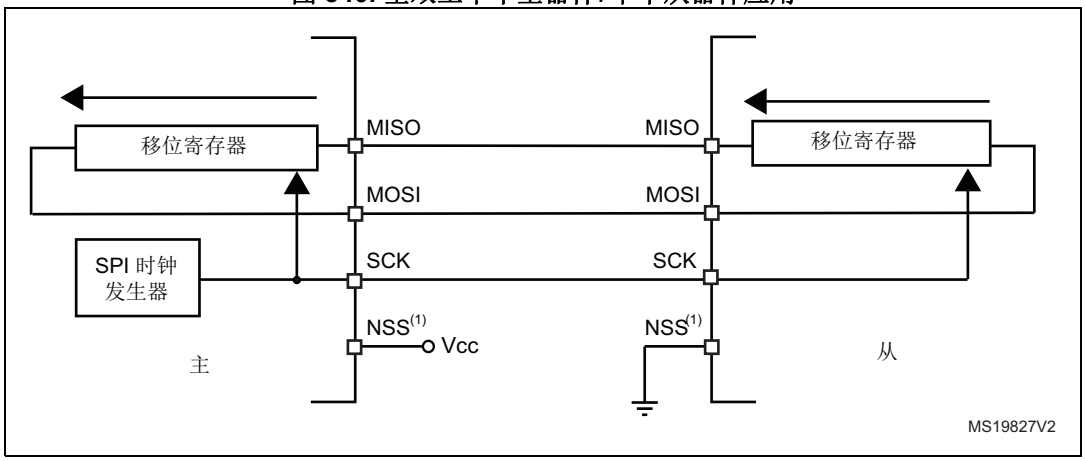
32.5.2 一个主器件和一个从器件之间的通信

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 NSS 管理），也可以使用 3 条或 4 条线（通过硬件 NSS 管理）。通信始终由主器件发起。

全双工通信

默认情况下，SPI 配置为进行全双工通信。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，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 340. 全双工单个主器件/单个从器件应用

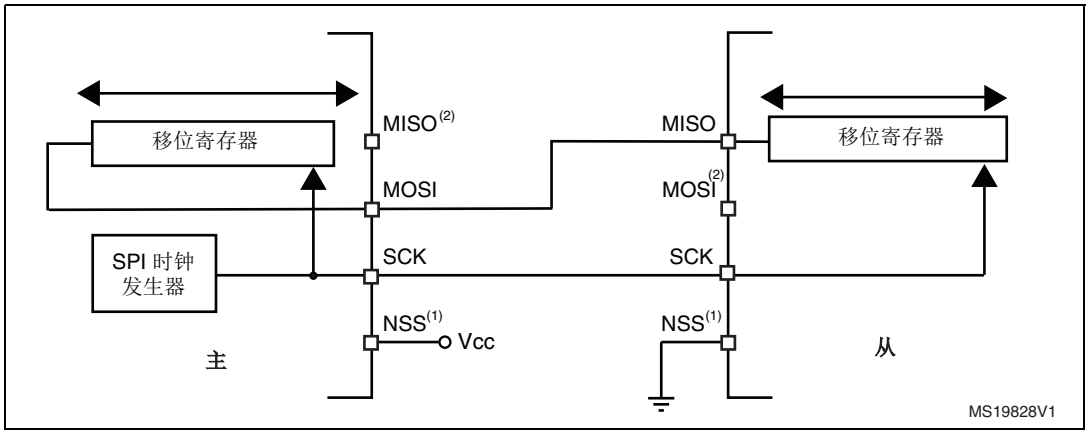


1. 在这种情况下，NSS 引脚配置为输入。

半双工通信

通过将 SPIx_CR1 寄存器的 BIDIMODE 位置 1，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 SPIx_CR1 寄存器中的 BDIOE 位进行选择。在这种配置下，主器件的 MISO 引脚和从器件的 MOSI 引脚空闲，可在其它应用中用作 GPIO。

图 341. 半双工单个主器件/单个从器件应用



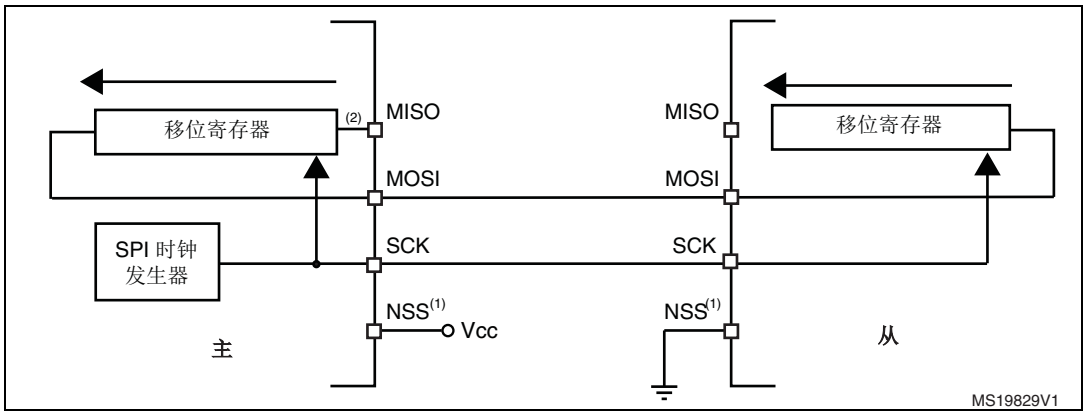
1. 在这种情况下，NSS 引脚配置为输入。
2. 在这种配置下，主器件的 MISO 引脚和从器件的 MOSI 引脚可用作 GPIO。

单工通信

通过 SPIx_CR2 寄存器中的 RXONLY 位将 SPI 设置为只发送模式或只接收模式，可使 SPI 以单工模式进行通信。在这种配置下，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 和 MOSI 引脚对不用于通信，可用作标准 GPIO。

- **只发送模式 (RXONLY=0)**：配置设置与全双工设置相同。应用必须忽略在未使用的输入引脚上捕获的信息。该引脚可以用作标准 GPIO。
- **只接收模式 (RXONLY=1)**：应用可通过将 RXONLY 位置 1 来禁止 SPI 输出功能。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 *32.5.4：从器件选择 (NSS) 引脚管理*）。基于数据缓冲区的配置产生接收数据事件。在主器件配置下，MOSI 输出被禁止，该引脚可用作 GPIO。只要 SPI 处于使能状态，便不断生成时钟信号。停止时钟的唯一方式是将 RXONLY 位或 SPE 位清零，直至来自 MISO 引脚的传入模式结束，然后基于相应配置填充数据缓冲区结构。

图 342. 单工单个主器件/单个从器件应用（主器件为只发送模式/从器件为只接收模式）



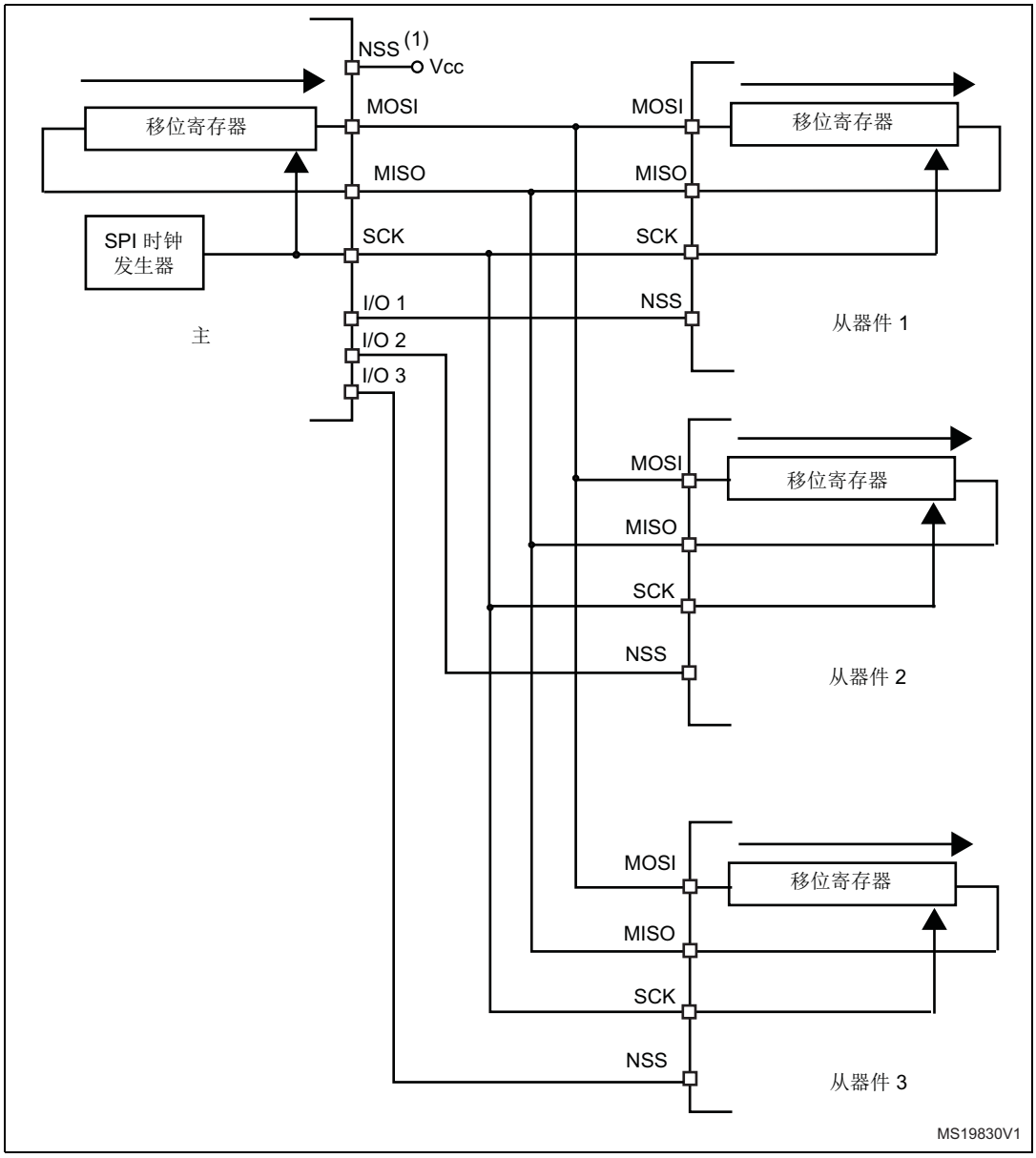
1. 在这种情况下，NSS 引脚配置为输入。
2. 输入信息在移位寄存器中捕获，在标准的只发送模式下必须忽略此信息（例如，OVF 标志）。
3. 在这种配置下，两个 MISO 引脚均可用作 GPIO。

注： *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 BDIO 位保持不变的同时，双向模式处于使能状态）。*

32.5.3 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 GPIO 引脚来管理每个从器件的片选线（请参见*图 343.*）。主器件必须通过拉低与从器件 NSS 输入相连的 GPIO 的电平来单独选择一个从器件。执行该操作后，便建立了标准主器件与专用从器件之间的通信。

图 343. 主器件和三个独立的从器件



1. 由于从器件的 MISO 引脚连在一起，所有从器件 MISO 引脚的 GPIO 配置必须设置为复用功能开漏（请参见*第 187 页上的第 6.3.7 节：I/O 复用功能输入/输出*）。

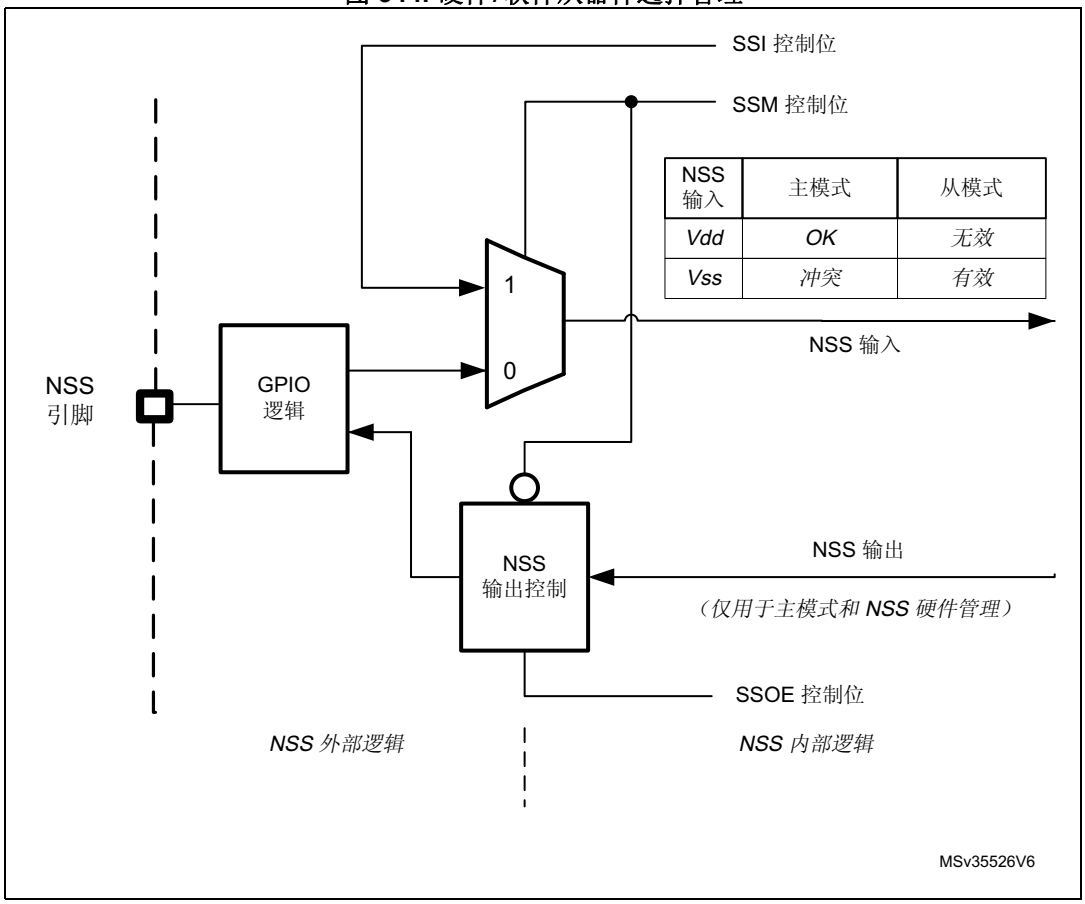
32.5.4 从器件选择 (NSS) 引脚管理

在从模式下，NSS 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，NSS 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。

可以使用 SPIx_CR1 寄存器中的 SSM 位设置硬件或软件从器件选择管理：

- **软件 NSS 管理 (SSM = 1)**：在这种配置下，由 SPIx_CR1 寄存器中的 SSI 位的值内部驱动从器件选择信息。外部 NSS 引脚空闲，可供其它应用使用。
- **硬件 NSS 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 NSS 输出配置（SPIx_CR1 寄存器中的 SSOE 位）。
  - **NSS 输出使能（SSM=0 且 SSOE = 1）**：仅在将 MCU 设置为主器件时才使用该配置。NSS 引脚由硬件管理。只要在主模式下使能 SPI (SPE=1)，NSS 信号便会被驱动为低电平，并且会一直保持低电平状态，直至禁止 SPI (SPE =0)。如果激活 NSS 脉冲模式 (NSSP=1)，连续通信间会生成脉冲。SPI 无法在采用此 NSS 设置的多主模式配置下工作。
  - **NSS 输出禁止（SSM=0 且 SSOE = 0）**：如果微控制器在总线上用作主器件，此配置可实现多主模式功能。如果在该模式下将 NSS 引脚拉至低电平，SPI 将进入主模式故障状态，器件将在从模式下自动进行重新配置。在从模式下，NSS 引脚用作标准的“片选”输入，当 NSS 线为低电平时将选择从器件。

图 344. 硬件/软件从器件选择管理



32.5.5 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。

时钟相位和极性控制

通过 SPIx_CR1 寄存器中的 CPOL 和 CPHA 位，可以用软件选择四种可能的时序关系。CPOL（时钟极性）位控制不传输任何数据时时钟的空闲状态值。此位对主器件和从 器件都有作用。如果复位 CPOL，SCK 引脚在空闲状态处于低电平。如果将 CPOL 置 1，SCK 引脚在空闲状态处于高电平。

如果将 CPHA 位置 1，则会在 SCK 引脚的第二个边沿捕获传输的第一个数据位（如果复位 CPOL 位，则为下降沿；如果将 CPOL 位置 1，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。如果将 CPHA 位复位，则会在 SCK 引脚的第一个边沿捕获传输的第一个数据位（如果将 CPOL 位置 1，则为下降沿；如果将 CPOL 位复位，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。

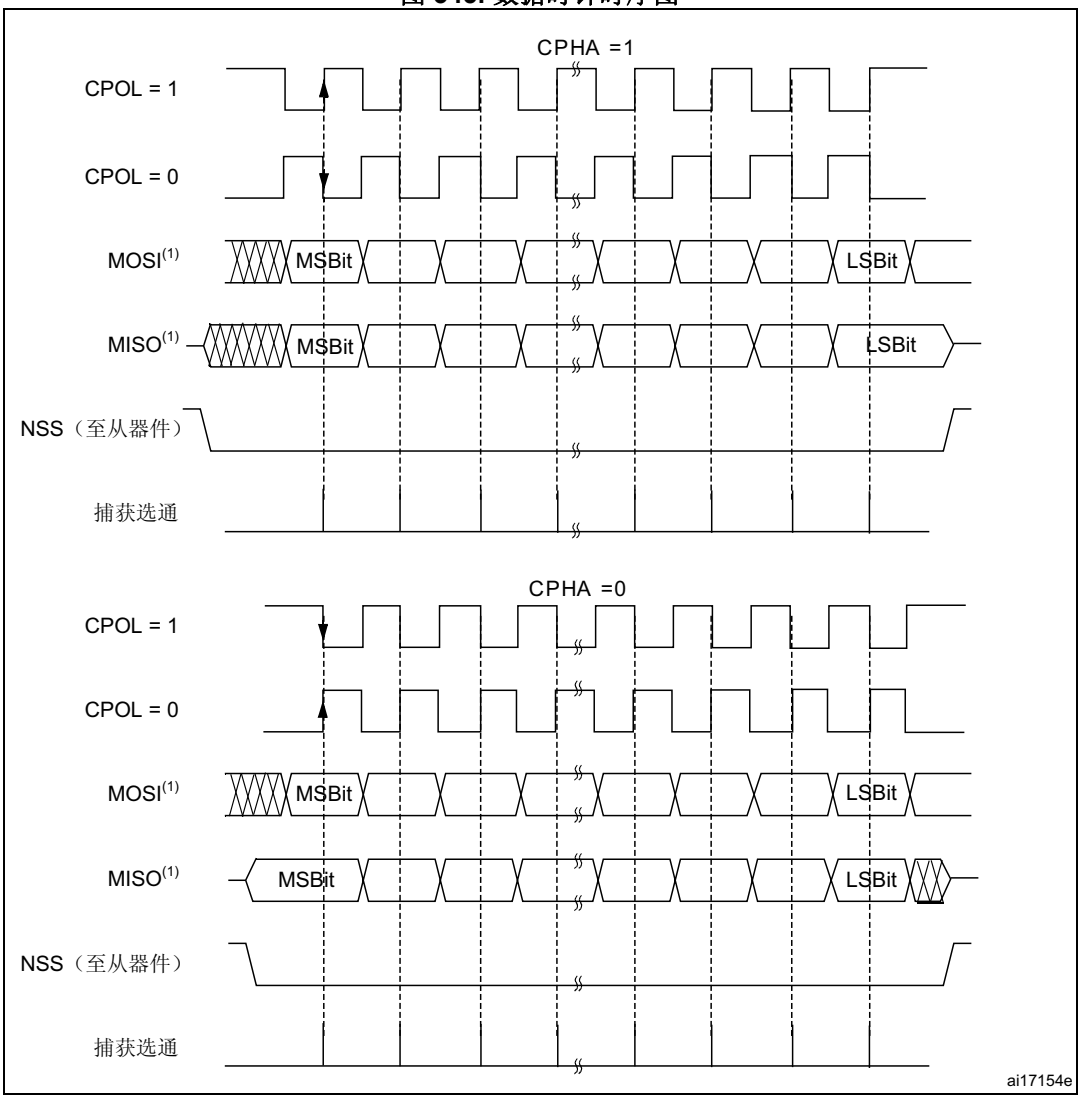
CPOL（时钟极性）和 CPHA（时钟相位）位的组合用于选择数据捕获时钟边沿。

图 345 给出了在 CPHA 和 CPOL 位的四种组合下的 SPI 全双工传输。

注： *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 SPIx_CR1 寄存器中选择的极性相对应（如果 CPOL=1，则上拉 SCK；如果 CPOL=0，则下拉 SCK）。

图 345. 数据时钟时序图

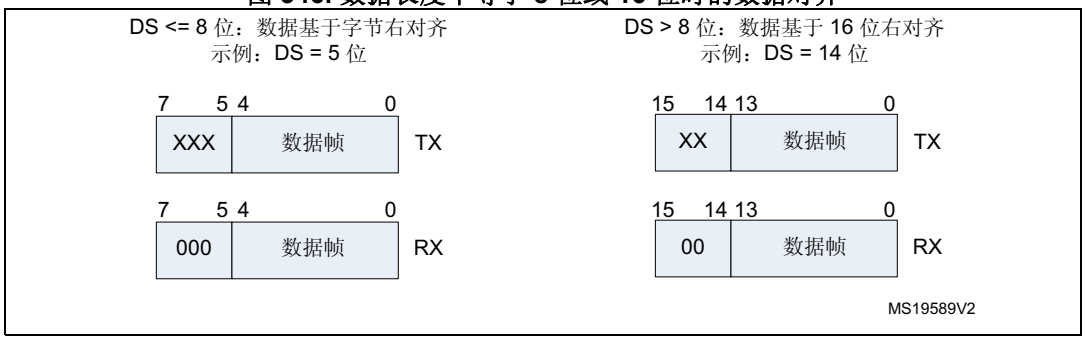


1. 数据位的顺序取决于 LSBFIRST 位的设置。

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 LSBFIRST 位的值。数据帧的长度通过 DS 位进行选择。数据帧的长度可设置为 4 位到 16 位，且此设置对于发送和接收均适用。无论所选的数据帧长度为何，必须依照 FRXTH 电平对 FIFO 进行读访问。访问 SPIx_DR 寄存器时，数据帧始终按字节（数据不超过一个字节时）或半字进行右对齐（请参见*图 346*）。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 346. 数据长度不等于 8 位或 16 位时的数据对齐



注： *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 8 位。*

32.5.6 SPI 配置

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。若要对标准通信进行初始化，请执行以下步骤：

1. 对适当的 GPIO 寄存器执行写操作：将 MOSI、MISO 和 SCK 引脚配置为 GPIO。
2. 对 SPI_CR1 寄存器执行写操作：
   a) 通过 BR[2:0] 位配置串行时钟波特率（*注：* 4）。
   b) 配置 CPOL 和 CPHA 位的组合，定义数据传输和串行时钟之间的关系（四种关系中的一种）（NSSP 模式下必须将 CPHA 清零）。（*注：* 2）。
   c) 通过配置 RXONLY 或 BIDIMODE 和 BIDIOE 来选择单工或半双工模式（RXONLY 和 BIDIMODE 不可同时置 1）。
   d) 配置 LSBFIRST 位以定义帧格式（*注：* 2）。
   e) 如果需要 CRC，请配置 CRCL 和 CRCEN 位（SCK 时钟信号处于空闲状态时）。
   f) 配置 SSM 和 SSI（*注：* 2 和 3）。
   g) 配置 MSTR 位（在多主模式 NSS 配置下，如果主器件配置为防止发生 MODF 错误，则应避免 NSS 上出现状态冲突）。
3. 对 SPI_CR2 寄存器执行写操作：
   a) 配置 DS[3:0] 位，选择传输的数据长度。
   b) 配置 SSOE（*注：* 1、2 和 3）。
   c) 如果需要使用 TI 协议，请将 FRF 位置 1（TI 模式下将 NSSP 位保持清零状态）。
   d) 如果在两个数据单元之间需要 NSS 脉冲模式，请将 NSSP 位置 1（NSSP 模式下将 CHPA 和 TI 位保持清零状态）。
   e) 配置 FRXTH 位。RXFIFO 阈值必须与 SPIx_DR 寄存器的读访问大小相符。
   f) 如果封装模式下使用 DMA，请初始化 LDMA_TX 和 LDMA_RX 位。
4. 对 SPI_CRCPR 寄存器执行写操作：需要时配置 CRC 多项式。
5. 对适当的 DMA 寄存器执行写操作：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。

注： *(1) 从模式下无需此步骤。*

(2) TI 模式下无需此步骤。

(3) NSSP 模式下无需此步骤。

(4) 从模式下无需此步骤，但从器件在 TI 模式下工作时除外。

32.5.7 使能 SPI 的步骤

建议在主器件发送时钟前使能 SPI 从器件。否则，数据传输可能会不正常。从器件的数据寄存器必须包含待发送的数据才能开始与主器件通信（在通信时钟的第一个边沿；如果时钟信号连续，则是在正在进行的通信结束前）。使能 SPI 从器件前，SCK 信号必须稳定为所选极性对应的空闲状态电平。

当 SPI 处于使能状态，且 TXFIFO 不为空或者对 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（RXONLY=1 或 BIDIMODE=1 且 BIDIOE=0）下，使能 SPI 后，主器件立即开始通信且时钟立即开始运行。

要处理 DMA，请遵从相应章节的内容。

32.5.8 数据发送和接收过程

RXFIFO 和 TXFIFO

所有 SPI 数据交互均经由 32 位内置 FIFO。这使得 SPI 能够以连续流工作，并能防止在数据帧长度较短时发生上溢。每个方向都有其自身的 FIFO，称为 TXFIFO 和 RXFIFO。这些 FIFO 可在所有 SPI 模式下使用，但已使能 CRC 计算的只接收模式（从器件或主器件）除外（请参见*第 32.5.13 节：CRC 计算*）。

FIFO 的处理取决于数据交换模式（双工和单工）、数据帧格式（帧中的位数）、FIFO 数据寄存器中的访问大小（8 位或 16 位）以及访问 FIFO 时是否使用数据封装（请参见*第 32.5.12 节：TI 模式*）。

对 SPIx_DR 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 SPIx_DR 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。读访问必须始终符合由 SPIx_CR2 寄存器中的 FRXTH 位配置的 RXFIFO 阈值。FTLVL[1:0] 和 FRLVL[1:0] 位用于指示两个 FIFO 当前的占用水平。

SPIx_DR 寄存器的读访问必须通过 RXNE 事件进行管理。当数据存储到 RXFIFO 中且达到阈值（由 FRXTH 位定义）时会触发该事件。当 RXNE 清零时，RXFIFO 被视为空。同样地，待发送数据帧的写访问通过 TXE 事件进行管理。当 TXFIFO 占用水平小于或等于其容量的一半时会触发该事件。否则，TXE 清零，TXFIFO 被视为已满。通过这种方式，当数据帧格式不超过 8 位时，RXFIFO 最多可存储四个数据帧，而 TXFIFO 最多只能存储三个数据帧。当软件尝试向 TXFIFO 中写入更多 16 位模式的数据时，这种差异能够防止 TXFIFO 中已存储的 3 个 8 位数据帧出现损坏的情况。TXE 和 RXNE 事件可通过中断来轮询或处理。请参见*图 348* 到*图 351*。

另一种管理数据交换的方式是使用 DMA（请参见*第 8.2 节：DMA 主要特性*）。

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 32.5.9 节：SPI 状态标志*中的 OVR 标志说明）。上溢事件可通过中断来轮询或处理。

BSY 位被置 1 表示正在处理当前数据帧交互。当时钟信号连续运行时，在主器件中，BSY 标志在数据帧之间保持置 1 状态，但在从器件中，BSY 标志在数据帧传输之间变为低电平并持续最短的一段时间（一个 SPI 时钟）。

序列处理

可通过一个序列传送一些数据帧从而完成一条消息。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。时钟信号由主器件持续提供，直至 TXFIFO 变为空，之后时钟信号停止，等待其它数据。

在只接收模式、半双工模式（BIDIMODE=1 且 BIDIOE=0）或单工模式（BIDIMODE=0 且 RXONLY=1）下，当使能 SPI 并激活只接收模式后，主器件将立即启动序列。时钟信号由主器件提供，且直至主器件禁止 SPI 或只接收模式才会停止。在此之前，主器件会连续接收数据帧。

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。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段。请注意，SPI 模式下不存在主器件或从器件的下溢错误信号，来自从器件的数据始终由主器件处理，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，尤其是数据帧较短而总线速率较高时。

在多从器件系统中，每个序列必须通过 NSS 脉冲进行控制，从而只选择其中一个从器件进行通信。在单个从器件系统中，无需通过 NSS 来控制从器件，但此时提供此脉冲通常会更好，以在每个数据序列开始时同步从器件。NSS 可通过软件和硬件进行管理（请参见*第 32.5.4 节：从器件选择 (NSS) 引脚管理*）。

当 BSY 位置 1 时，表示正在处理数据帧事务。当所进行的帧交互完成时，RXNE 标志将置 1。最后一位采样后，整个数据帧会存储到 RXFIFO 中。

禁止 SPI 的步骤

当禁止 SPI 时，必须按照本段中介绍的禁止步骤进行操作。当外设时钟停止时，在系统进入低功耗模式前做到这一点是十分重要的。否则会损坏正在进行的交互。在某些模式下，禁止步骤是停止所进行的连续通信的唯一方式。

当处于全双工或只发送模式下的主器件停止提供待发送的数据时，可结束任何交互。在这种情况下，时钟在最后一个数据传输后停止。当处理奇数数量的数据帧时，必须特别注意封装模式以防止交换一些空字节（请参见*数据封装*部分）。在这些模式下禁止 SPI 之前，用户必须按照标准的禁止步骤进行操作。在主发送器上禁止 SPI，而同时正在处理帧交互或已将下一个数据帧存储到 TXFIFO 中时，SPI 行为无法保证。

只要主器件处于只接收模式，停止连续时钟的唯一方式就是通过 SPE=0 来禁止外设。这必须在最后一个数据帧传输内的特定时间段，即第一位采样与最后一位传输开始之间完成（以便接收全部数量的预期数据帧并防止在最后一个有效数据帧后读取任何其它的“空”数据）。在该模式下禁止 SPI 时必须遵循特定步骤。

禁止 SPI 后，已接收但未读取的数据始终存储在 RXFIFO 中，这些数据必须在下次使能 SPI 后进行处理，然后才能启动新序列。为防止存在未读取的数据，需确保禁止 SPI 时 RXFIFO 为空，可通过正确的禁止步骤来禁止 SPI，也可以通过控制外设复位专用的特定寄存器以软件复位的方式来初始化所有 SPI 寄存器从而禁止 SPI（请参见 RCC_APBiRSTR 寄存器中的 SPIiRST 位）。

标准禁止步骤通过轮询 BSY 状态以及 FTLVL[1:0] 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 NSS 信号由软件管理且主器件必须为从器件提供 NSS 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。

正确的禁止步骤如下（使用只接收模式时除外）：

1. 等待至 FTLVL[1:0] = 00（无需发送更多数据）。
2. 等待至 BSY=0（最后一个数据帧已处理完）。
3. 禁止 SPI (SPE = 0)。
4. 读取数据，直至 FRLVL[1:0] = 00（读取接收的所有数据）。

某些只接收模式的正确禁止步骤如下：

1. 当最后一个数据帧正在处理时，通过在特定时间窗口内禁止 SPI (SPE=0) 来中断接收流。
2. 等待至 BSY=0（最后一个数据帧已处理完）。
3. 读取数据，直至 FRLVL[1:0] = 00（读取接收的所有数据）。

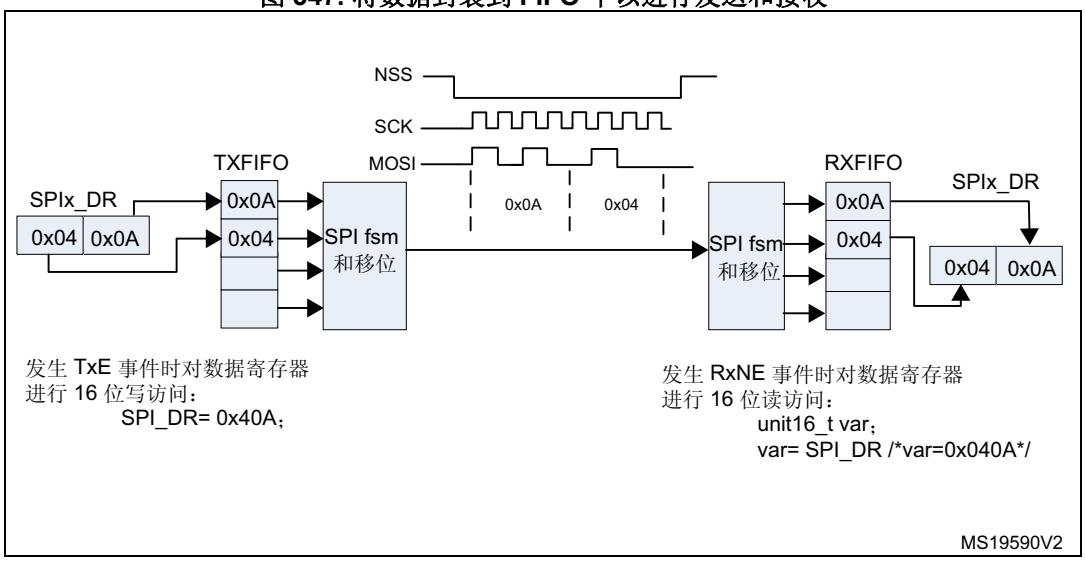
注： *如果使用封装模式并且必须接收奇数数量的数据帧且数据帧格式为小于或等于 8 位（不超过一个字节），则 FRXTH 必须在 FRLVL[1:0] = 01 时置 1，以便生成 RXNE 事件从而读取最后的奇数编号数据帧并且使 FIFO 指针保持正确对齐。*

数据封装

若数据帧长度不足一个字节（小于或等于 8 位），当 SPIx_DR 寄存器上执行任何 16 位读写访问时将自动使用数据封装。在这种情况下将并行处理双数据帧模式。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 347* 给出了数据封装模式序列处理的示例。在对发送器的 SPIx_DR 寄存器执行单次 16 位访问后发送两个数据帧。如果 RXFIFO 阈值设置为 16 位 (FRXTH=0)，该序列只会在接收器中生成一个 RXNE 事件。接收器随后必须通过对 SPIx_DR 执行单次 16 位读访问来访问这两个数据帧，从而响应该单个 RXNE 事件。RxFIFO 阈值设置和后续读访问必须始终与接收器侧保持一致，因为若不一致，则会丢失数据。

如果必须处理奇数数量的此类“不超过一个字节”的数据帧，则会出现特定问题。在发送器侧，只需将任意奇序列的最后一个数据帧以 8 位访问的方式写入 SPIx_DR 即可。接收器必须针对所接收的奇序列中的最后一个数据帧更改 Rx_FIFO 阈值大小，以生成 RXNE 事件。

图 347. 将数据封装到 FIFO 中以进行发送和接收



1. 在此示例中：数据大小 DS[3:0] 配置为 4 位、CPOL=0、CPHA=1 且 LSBFIRST =0。数据存储始终采用右对齐方式，同时有效位仅在总线上执行，总线上 LSB 字节的内容在前，未使用的位在发送器侧无需考虑，在接收器侧则用零进行填充。

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 SPIx_CR2 寄存器中的使能位 TXE 或 RXNE 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。

- 在发送过程中，每次 TXE 位置 1 都会发出 DMA 请求。然后，DMA 将对 SPIx_DR 寄存器执行写操作。
- 在接收过程中，每次 RXNE 位置 1 都会发出 DMA 请求。然后，DMA 将对 SPIx_DR 寄存器执行读操作。

请参见*图 348* 到*图 351*。

当 SPI 仅用于发送数据时，可以只使能 SPI Tx DMA 通道。在这种情况下，OVR 标志会置 1，因为未读取接收的数据。当 SPI 仅用于接收数据时，可以只使能 SPI Rx DMA 通道。

在发送模式下，DMA 写入所有要发送的数据（DMA_ISR 寄存器中的 TCIF 标志置 1）后，可以对 BSY 标志进行监视，以确保 SPI 通信已完成。在禁止 SPI 或进入停止模式前必须执行此步骤，以避免损坏最后一次发送。软件必须首先等待 FTLVL[1:0]=00，再等待 BSY=0。

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_CR2 寄存器中的 RXDMAEN 位来使能 DMA 接收缓冲区。
2. 如果使用数据流，通过 DMA 寄存器来使能 Tx 和 Rx 的 DMA 数据流。
3. 如果使用 DMA Tx，通过 SPI_CR2 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. 如果使用数据流，通过 DMA 寄存器来禁止 Tx 和 Rx 的 DMA 数据流。
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_CR2 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（SPIx_CR2 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。然后，DMA 将自动管理对 SPIx_DR 寄存器的写操作。

如果使用数据封装模式，且要传输的数据数量不是 2 的倍数，则 LDMA_TX/LDMA_RX 位必须置 1。然后，SPI 会认为最后一次 DMA 传输时只发送或接收一个数据（有关详细信息，请参见*第 977 页上的数据封装*）。

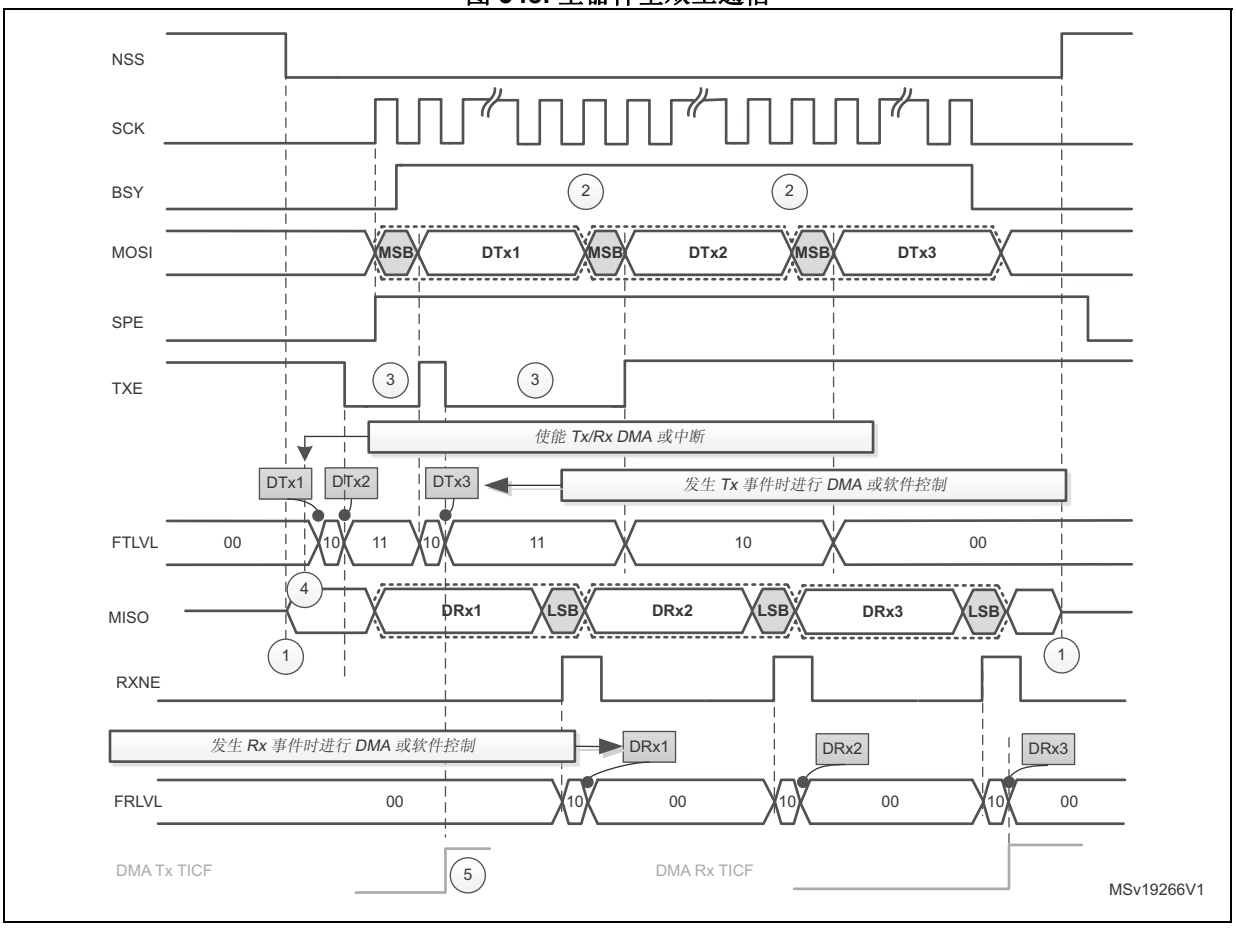
通信图

本部分将介绍一些典型的时序图。无论 SPI 事件是通过轮询、中断还是 DMA 进行处理，这些时序图均有效。为简单起见，此处均假设 LSBFIRST=0、CPOL=0 和 CPHA=1。不提供 DMA 数据流的完整配置。

以下带编号的注释对*第 980 页上的图 348* 到*第 983 页上的图 351* 均适用。

1. 激活 NSS 并使能 SPI 后，从器件开始控制 MISO 线，而当其中一个条件不成立时，从器件将与 MISO 线断开。必须为从器件提供充足的时间，以便在传输开始前准备好主器件专用的数据。
   在主器件上，只有使能 SPI 后，SPI 外设才会控制 MOSI 和 SCK 信号（偶尔还会控制 NSS 信号）。如果禁止 SPI，SPI 外设会与 GPIO 逻辑断开，因此这些线上的电平只取决于 GPIO 设置。
2. 在主器件上，如果通信（时钟信号）连续，BSY 在数据帧之间保持有效。在从器件上，BSY 信号在数据帧之间始终会保持至少一个时钟周期的低电平状态。
3. 只有 TXFIFO 已满时，TXE 信号才会清零。
4. DMA 仲裁过程在 TXDMAEN 位置 1 后立即开始。TXE 中断在 TXEIE 置 1 后立即生成。TXE 信号处于有效电平时，开始向 TxFIFO 传输数据，直至 TxFIFO 已满或 DMA 传输完成。
5. 如果要发送的所有数据可装入 TxFIFO，则 DMA Tx TCIF 标志甚至会在 SPI 总线上的通信开始前置 1。SPI 传输完成前，该标志始终为高电平状态。
6. 封装的 CRC 值在 SPIx_TxCRCR 和 SPIx_RxCRCR 寄存器中逐帧连续进行计算。完成整个数据封装后，CRC 信息可通过 DMA 自动处理（Tx 通道必须设置为要处理的数据帧数），也可由软件处理（用户必须在处理最后一个数据帧的过程中处理 CRCNEXT 位）。
   SPIx_TxCRCR 中计算的 CRC 值仅由发送器发出时，接收的 CRC 信息将加载到 RxFIFO 中，然后与 SPIx_RxCRCR 寄存器的内容进行比较（如果存在任何差异，CRC 错误标志会置 1）。因此用户必须注意刷新 FIFO 中的相关信息，可以通过软件读出 RxFIFO 中存储的所有内容的方式来实现，若已针对 Rx 通道预设置了适当数量的数据帧（数据帧数 + CRC 帧数），也可通过 DMA 的方式来实现（请参见示例假设中的设置）。
7. 在数据封装模式下，TxE 和 RxNE 事件成对出现，对 FIFO 的每次读/写访问都为 16 位宽，直至数据帧数为偶数。如果 TxFIFO 处于 ¾ 满状态，则 FTLVL 将保持 FIFO 满时对应的状态。因此在 TxFIFO 变为 ½ 满状态前，无法存储最后一个奇数编号的数据帧。该帧可通过软件或自动由 DMA（LDMA_TX 控制置 1 时）对其进行 8 位访问的方式存储在 TxFIFO 中。
8. 要在封装模式下接收最后一个奇数编号的数据帧，必须在处理最后一个数据帧后，将 Rx 阈值更改为 8 位，这可通过软件（设置为 FRXTH=1）实现或由 DMA 内部信号在 LDMA_RX 置 1 时自动实现。

图 348. 主器件全双工通信



主器件全双工通信示例的假设条件如下：

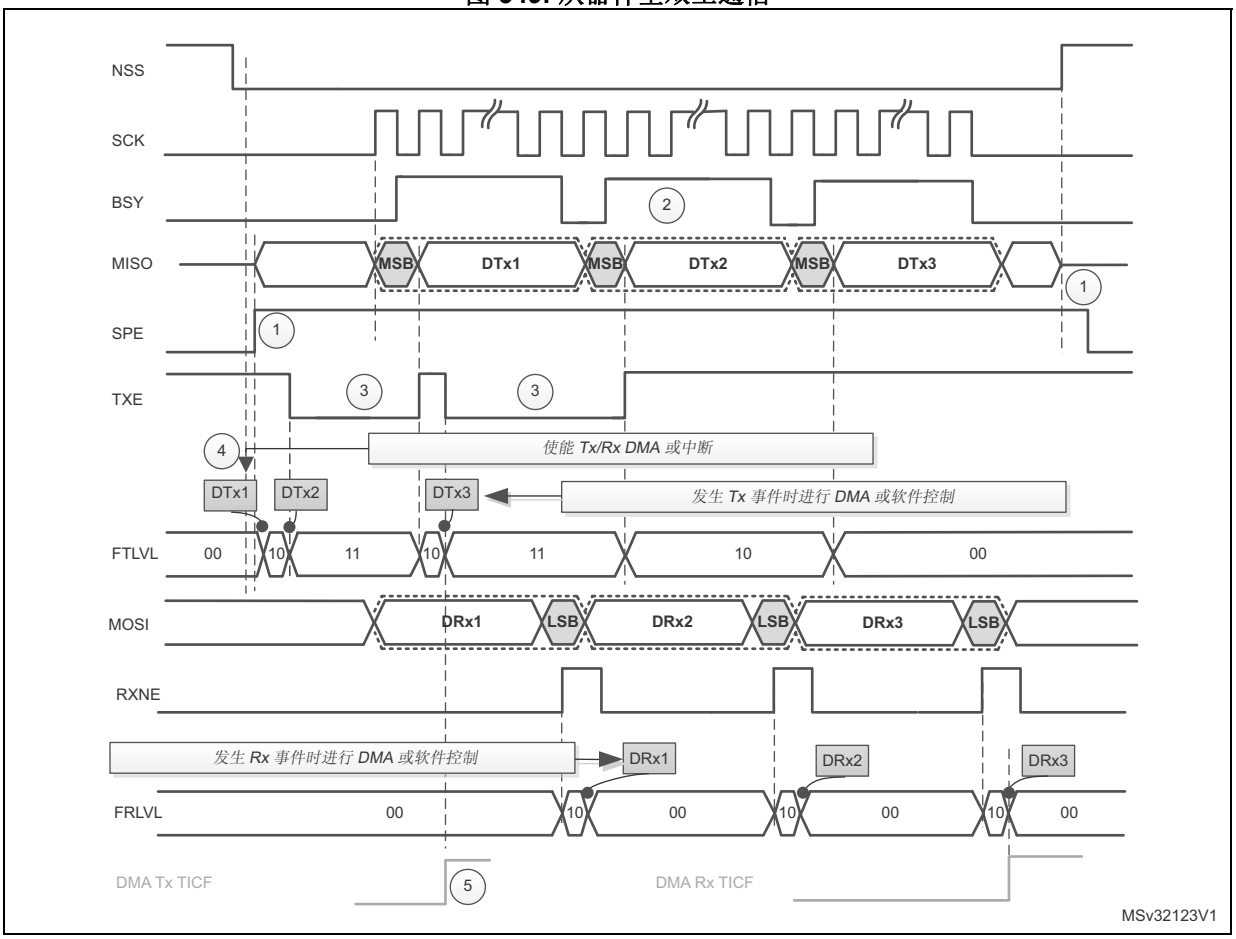
- 数据大小 > 8 位

如果使用 DMA：

- 由 DMA 处理的 Tx 帧的数量设置为 3
- 由 DMA 处理的 Rx 帧的数量设置为 3

有关通用假设条件和注释的详细信息，另请参见*第 979 页上的通信图*一节。

图 349. 从器件全双工通信



从器件全双工通信示例的假设条件如下：

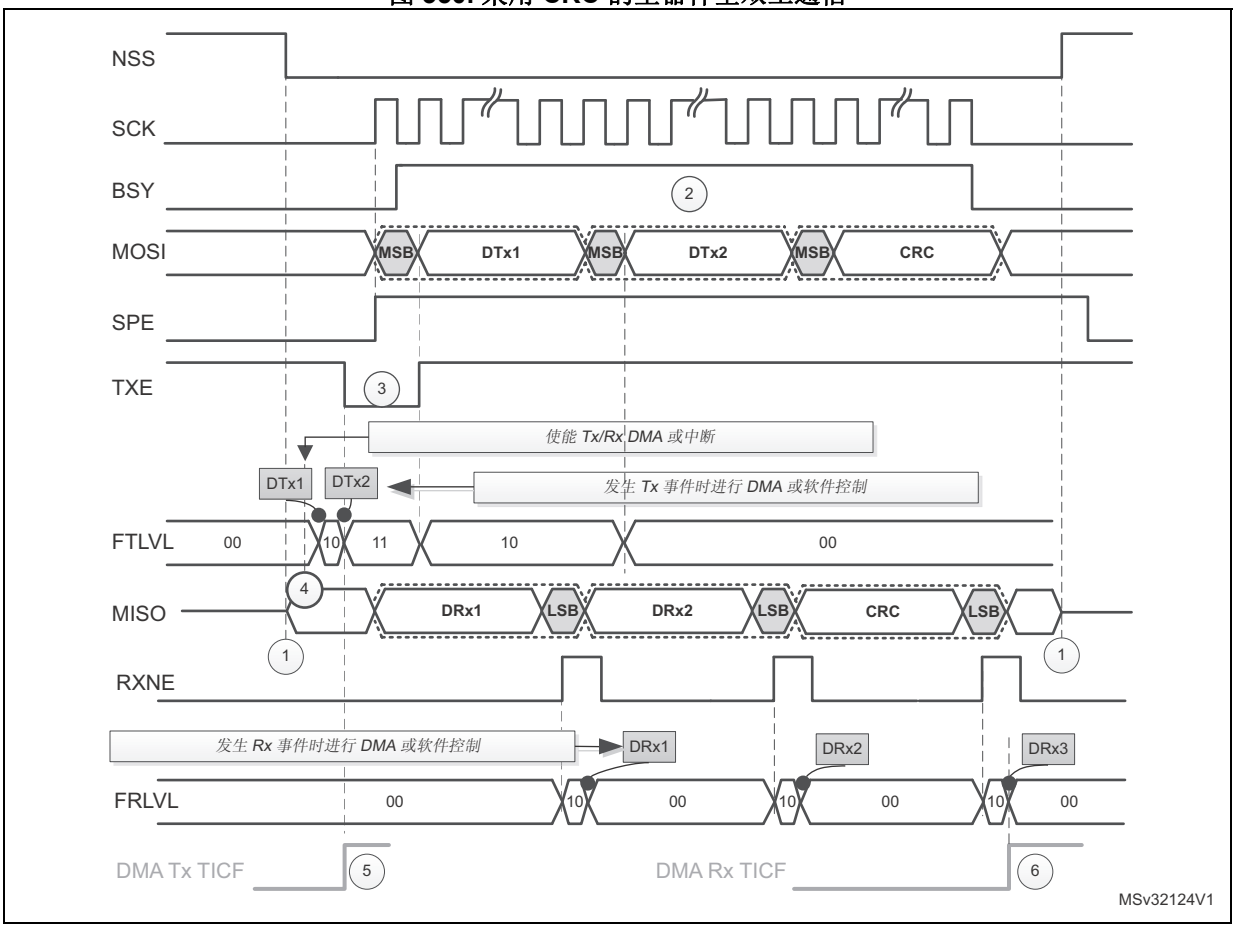
- 数据大小 > 8 位

如果使用 DMA：

- 由 DMA 处理的 Tx 帧的数量设置为 3
- 由 DMA 处理的 Rx 帧的数量设置为 3

有关通用假设条件和注释的详细信息，另请参见*第 979 页上的通信图*一节。

图 350. 采用 CRC 的主器件全双工通信



采用 CRC 的主器件全双工通信的假设条件如下：

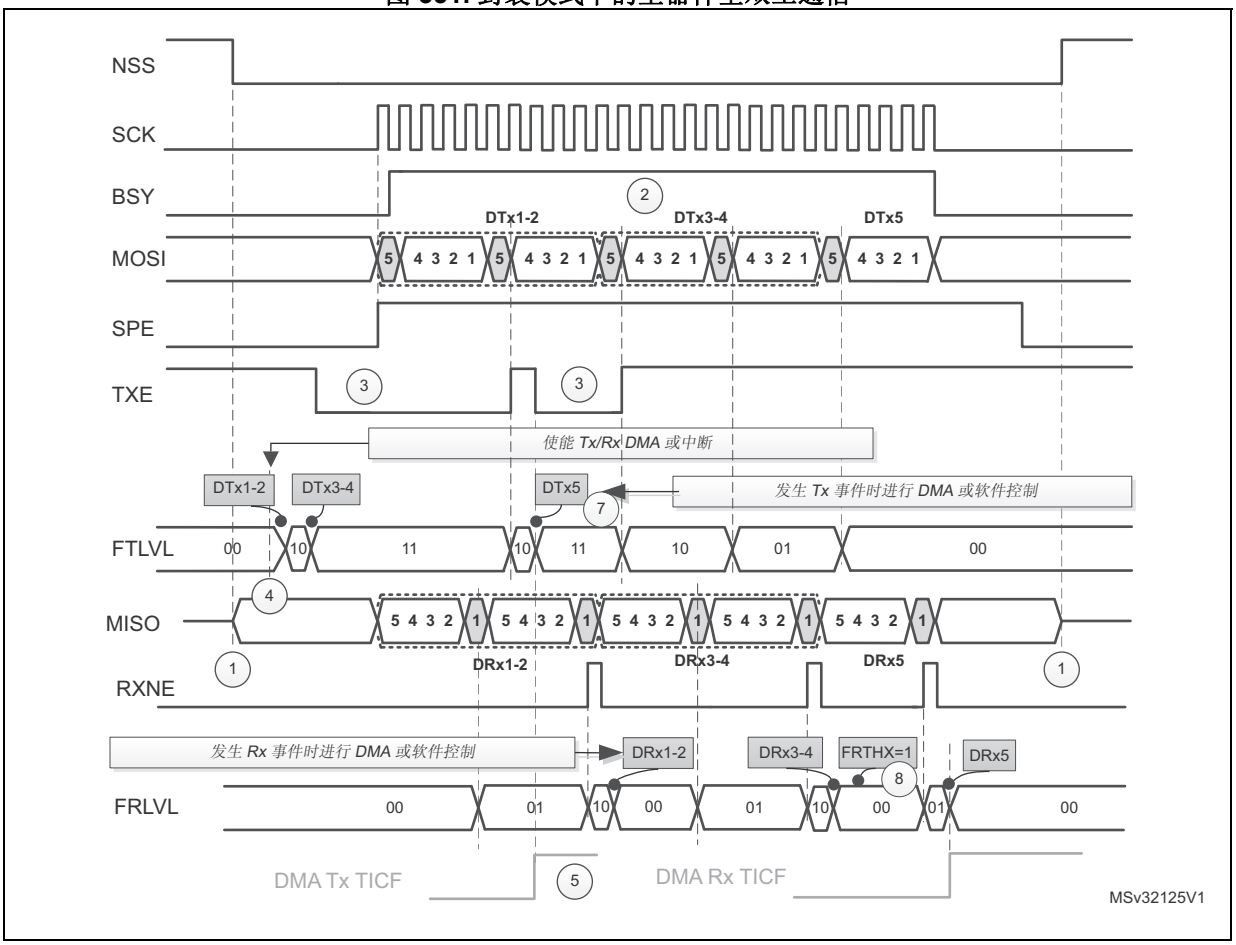
- 数据大小 = 16 位
- CRC 已使能

如果使用 DMA：

- 由 DMA 处理的 Tx 帧的数量设置为 2
- 由 DMA 处理的 Rx 帧的数量设置为 3

有关通用假设条件和注释的详细信息，另请参见*第 979 页上的通信图*一节。

图 351. 封装模式下的主器件全双工通信



封装模式下的主器件全双工通信示例的假设条件如下：

- 数据大小 = 5 位
- 主要通过 16 位访问的方式来读/写 FIFO
- FRXTH=0

如果使用 DMA：

- 要由 DMA 处理的 Tx 帧的数量设置为 3
- 要由 DMA 处理的 Rx 帧的数量设置为 3
- Tx 和 Rx 的 DMA 通道 的 PSIZE 均设置为 16 位
- LDMA_TX=1 且 LDMA_RX=1

有关通用假设条件和注释的详细信息，另请参见*第 979 页上的通信图*一节。

32.5.9 SPI 状态标志

应用可通过三种状态标志监视 SPI 总线的状态。

发送缓冲区为空 (TXE)

当发送 TXFIFO 有足够的空间来存储要发送的数据时，TXE 标志将置 1。TXE 标志与 TXFIFO 占用水平相关。该标志变为高电平后将一直保持高电平状态，直至 TXFIFO 占用水平小于或等于 FIFO 深度的 1/2。如果 SPIx_CR2 寄存器中的 TXEIE 位置 1，可产生中断。当 TXFIFO 占用水平超过 1/2 时，该位将自动清零。

接收缓冲区非空 (RXNE)

RXNE 标志根据 SPIx_CR2 寄存器中 FRXTH 位的值进行设置：

- 如果 FRXTH 置 1，RXNE 将变为高电平并一直保持高电平状态，直至 RXFIFO 占用水平大于或等于 1/4（8 位）。
- 如果 FRXTH 清零，RXNE 将变为高电平并一直保持高电平状态，直至 RXFIFO 占用水平大于或等于 1/2（16 位）。

如果 SPIx_CR2 寄存器中的 RXNEIE 位置 1，可产生中断。

当上述条件不再为真时，RXNE 将由硬件自动清零。

忙标志 (BSY)

BSY 标志由硬件置 1 和清零（写入此标志没有任何作用）。

当 BSY 置 1 时，表示 SPI 上正在进行数据传输（SPI 总线繁忙）。

某些模式下可以使用 BSY 标志来检测传输是否结束，以便软件在进入低功耗模式（该模式下不提供外设时钟）前禁止 SPI 或其外设时钟。这可避免破坏最后一个数据的传输。

BSY 标志还可用于避免在多主模式系统中发生写冲突。

在以下任意一种条件下，BSY 标志将清零：

- 正确禁止 SPI 时
- 在主模式下检测到故障时（MODF 位置 1）
- 在主模式下，完成了数据发送并且不准备发送任何新数据时
- 在从模式下，BSY 标志在各传输之间的至少一个 SPI 时钟周期内置为“0”时。

注： *当主器件可以立即处理下一次发送时（例如，如果主器件处于只接收模式或其发送 FIFO 不为空），在主器件侧的传输之间，通信连续且 BSY 标志始终置“1”。尽管从器件并非如此，但建议始终使用 TXE 和 RXNE 标志（而非 BSY 标志）来处理数据发送或接收操作。*

32.5.10 SPI 错误标志

如果以下其中一个错误标志置 1 且已通过将 ERRIE 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的数据（存储在 RXFIFO 中)或数据存储空间受限（例如在只接收模式下使能 CRC 时 RXFIFO 不可用，在这种情况下，接收缓冲区便限制为一个数据帧缓冲区），会发生这种情况（请参见*第 32.5.13 节：CRC 计算*）。

当出现上溢的情况时，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。要将 OVR 位清零，应首先对 SPI_DR 寄存器执行读访问，然后再对 SPI_SR 寄存器执行读访问。

模式故障 (MODF)

当主器件的内部 NSS 信号（NSS 硬件模式下为 NSS 引脚，NSS 软件模式下为 SSI 位）被拉低时，将发生模式故障。这会自动将 MODF 位置 1。主模式故障会在以下几方面影响 SPI 接口：

- 如果 ERRIE 位置 1，MODF 位将置 1，并生成 SPI 中断。
- SPE 位清零。这将关闭器件的所有输出，并禁止 SPI 接口。
- MSTR 位清零，从而强制器件进入从模式。

使用以下软件序列将 MODF 位清零：

1. 在 MODF 位置 1 时，对 SPIx_SR 寄存器执行读或写访问。
2. 然后，对 SPIx_CR1 寄存器执行写操作。

为避免包含多个 MCU 的系统中发生多从模式冲突，必须在 MODF 位清零序列期间将 NSS 引脚拉高。在该清零序列后，可以将 SPE 和 MSTR 位恢复到原始状态。安全起见，硬件不允许在 MODF 位置 1 时将 SPE 和 MSTR 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。

CRC 错误 (CRCERR)

当 SPIx_CR1 寄存器中的 CRCEN 位置 1 时，此标志用于验证接收数据的有效性。如果移位寄存器中接收的值与 SPIx_RXCRCR 的值不匹配，SPIx_SR 寄存器中的 CRCERR 标志将置 1。该标志由软件清零。

TI 模式帧格式错误 (FRE)

如果 SPI 在从模式下工作，并配置为符合 TI 模式协议，则在通信进行期间出现 NSS 脉冲时，将检测到 TI 模式帧格式错误。出现此错误时，SPIx_SR 寄存器中的 FRE 标志将置 1。发生错误时不会禁止 SPI，但会忽略 NSS 脉冲，并且 SPI 会等待下一个 NSS 脉冲，然后再开始新的传输。由于错误检测可能导致丢失两个数据字节，因此数据可能会损坏。

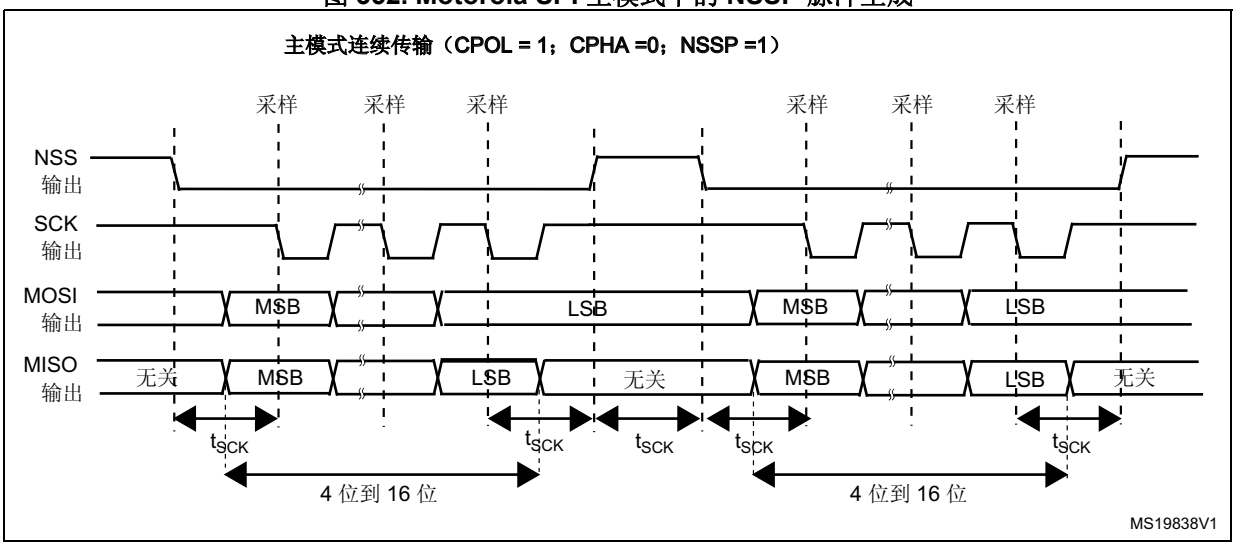
读取 SPIx_SR 寄存器时，将清零 FRE 标志。如果 ERRIE 位置 1，则检测到 NSS 错误时将生成中断。在这种情况下，由于无法保证数据的一致性，应禁止 SPI，并在重新使能从 SPI 后，由主器件重新发起通信。

32.5.11 NSS 脉冲模式

该模式通过 SPIx_CR2 寄存器中的 NSSP 位来激活，只有将 SPI 接口配置为 Motorola SPI 主模式 (FRF=0) 且在第一个边沿捕捉时，该模式才起作用（SPIx_CR1 CPHA = 0，CPOL 设置忽略）。激活后，当 NSS 至少保持一个时钟周期的高电平状态时，两个连续的数据帧传输间将生成 NSS 脉冲。该模式下，从器件可以锁存数据。NSSP 脉冲模式旨在用于具有一个主器件-从器件对的应用。

图 352 给出了使能 NSSP 脉冲模式后的 NSS 引脚管理情况。

图 352. Motorola SPI 主模式下的 NSSP 脉冲生成



注： 当 CPOL = 0 时会出现类似行为。在这种情况下，采样边沿为 SCK 的*上升*沿，NSS 的使能和禁止均参考该采样边沿。

32.5.12 TI 模式

主模式下的 TI 协议

SPI 接口与 TI 协议兼容。可以使用 SPIx_CR2 寄存器的 FRF 位来配置 SPI，以兼容此协议。

时钟极性和相位都被强制为遵循 TI 协议，和 SPIx_CR1 中的设置无关。NSS 管理也特定于 TI 协议，在这种情况下，无法通过 SPIx_CR1 和 SPIx_CR2 寄存器（SSM、SSI 和 SSOE）来对 NSS 管理进行配置。

在从模式下，SPI 波特率预分频器用于控制在当前传输完成时 MISO 引脚切换为高阻态的时刻（请参见*图 353*）。可以使用任意波特率，因此可以非常灵活地确定此时刻。但是，波特率通常设置为外部主时钟波特率。MISO 信号变为高阻态的延时 ($t_{release}$) 取决于内部重新同步以及通过 SPIx_CR1 寄存器的 BR[2:0] 位设置的波特率值。具体公式如下：

$$\frac{t_{baud_rate}}{2} + 4 \times t_{pclk} < t_{release} < \frac{t_{baud_rate}}{2} + 6 \times t_{pclk}$$

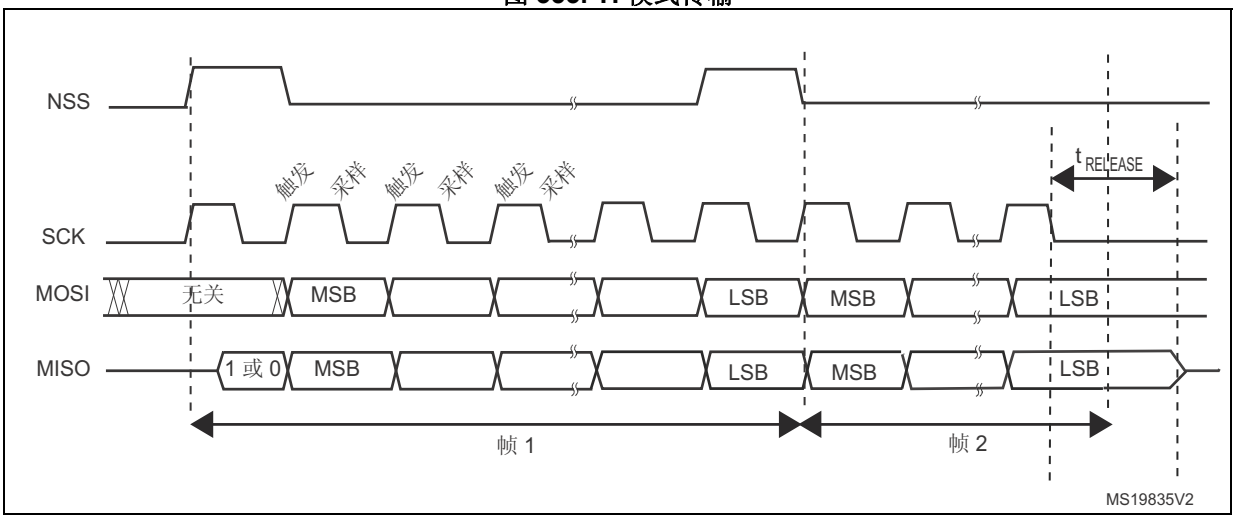
如果从器件在数据帧传输期间检测到错位的 NSS 脉冲，TIFRE 标志将置 1。

如果数据大小等于 4 位或 5 位，处于全双工模式或只发送模式的主器件将使用在 LSB 后再添加一个空数据位的协议。每个周期内，将在该空位（而非 LSB）时钟周期生成 TI NSS 脉冲。

此特性不适用于 Motorola SPI 通信（FRF 位为 0）。

*图 353：TI 模式传输*给出了选择 TI 模式时的 SPI 通信波形。

图 353. TI 模式传输



32.5.13 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 CRC 计算器。SPI 可提供 CRC8 或 CRC16 计算，而与固定为 8 位或 16 位的数据帧长度无关。对于所有其它的数据帧长度，CRC 均不适用。

CRC 原理

在使能 SPI (SPE = 1) 前，通过将 SPIx_CR1 寄存器中的 CRCEN 位置 1 来使能 CRC 计算。使用值为奇数的可编程多项式对每个位计算 CRC 值。在由 SPIx_CR1 寄存器中的 CPHA 位和 CPOL 位定义的采样时钟边沿进行计算。所计算的 CRC 值在数据块末尾自动进行校验，以及针对由 CPU 或 DMA 管理的传输进行校验。当检测到所接收数据内部计算的 CRC 与发送器发送的 CRC 不匹配时，CRCERR 标志将置 1 以指示数据损坏错误。CRC 计算的正确处理步骤取决于 SPI 配置和所选的传输管理。

注： *多项式值只应为奇数。不支持任何偶数值。*

CPU 管理的 CRC 传输

通信开始后将一直持续到必须发送或接收 SPIx_DR 寄存器中的最后一个数据帧时。之后，SPIx_CR1 寄存器中的 CRCNEXT 位必须置 1，以指示当前处理的数据帧传输后将处理 CRC 帧传输。CRCNEXT 位必须在最后一个数据帧传输结束前置 1。在 CRC 传输期间，CRC 计算将冻结。

所接收的 CRC 以数据字节或数据字的形式存储在 RXFIFO 中。因此仅在 CRC 模式下，接收缓冲区才必须被视为一个 16 位缓冲区且一次仅接收一个数据帧。

CRC 帧的传输通常在数据序列结束时再传输一个数据帧。然而，在设置通过 16 位 CRC 校验的 8 位数据帧时，还需要另外两个帧才能发送完整的 CRC。

接收最后一个 CRC 数据后，将执行自动校验，将接收的值与 SPIx_RXCRC 寄存器中的值进行比较。软件必须校验 SPIx_SR 寄存器中的 CRCERR 标志，以确定数据传输是否损坏。软件通过向 CRCERR 标志写入“0”来将其清零。

接收 CRC 后，CRC 值存储到 RXFIFO 中，且必须在 SPIx_DR 寄存器中进行读取，以将 RXNE 标志清零。

DMA 管理的 CRC 传输

当使能的 SPI 通信支持 CRC 通信和 DMA 模式时，在通信结束时会自动发送和接收 CRC（在只接收模式下读取 CRC 数据时除外）。CRCNEXT 位不是一定要通过软件来处理。SPI 发送 DMA 通道计数器必须设置为要发送的数据帧数，其中不包括 CRC 帧。在接收器侧，接收的 CRC 值在传输结束时通过 DMA 自动处理，但用户必须注意刷新 RXFIFO 中接收的 CRC 信息（因为该信息始终加载到其中）。在全双工模式下，接收 DMA 通道计数器可设置为要接收的数据帧数，其中包括 CRC，这意味着在通过 16 位 CRC 校验的 8 位数据帧等特殊情况下：

DMA_RX = Numb_of_data + 2

在只接收模式下，DMA 接收通道计数器应仅包含已传输的数据量，而不包括 CRC 计算。之后，基于通过 DMA 实现的完整传输，必须通过软件从 FIFO 中读回所有 CRC 值，因为 FIFO 在该模式下用作单个缓冲区。

如果传输过程中出现损坏，则在数据和 CRC 传输结束时，SPIx_SR 寄存器中的 CRCERR 标志将置 1。

如果使用封装模式，则 LDMA_RX 位需要管理数据数是否为奇数。

复位 SPIx_TXCRC 和 SPIx_RXCRC 值

在 CRC 阶段后对新数据进行采样时，SPIx_TXCRC 和 SPIx_RXCRC 值将自动清零。借此，可使用 DMA 循环模式（只接收模式下不适用）从而不间断地传输数据（中间 CRC 校验阶段会涉及一些数据块）。

如果在通信过程中禁止 SPI，必须按以下顺序进行操作：

1. 禁止 SPI
2. 将 CRCEN 位清零
3. 使能 CRCEN 位
4. 使能 SPI

注： *当 SPI 处于从模式时，只要 CRCEN 位置 1，无论 SPE 位的值如何，只要存在 SCK 从输入时钟，CRC 计算器就开始工作。为了避免 CRC 计算错误，软件必须仅在时钟稳定（处于稳态）时才能使能 CRC 计算。当 SPI 接口配置为从模式时，NSS 内部信号需要在数据阶段和 CRC 阶段之间保持低电平。*

32.6 SPI 中断

在 SPI 通信过程中，中断可由以下事件产生：

- 发送 TXFIFO 准备就绪，可以装载数据
- 接收 RXFIFO 中接收了数据
- 主模式故障
- 上溢错误
- TI 帧格式错误
- CRC 协议错误

中断可分别进行使能和禁止。

表 176. SPI 中断请求

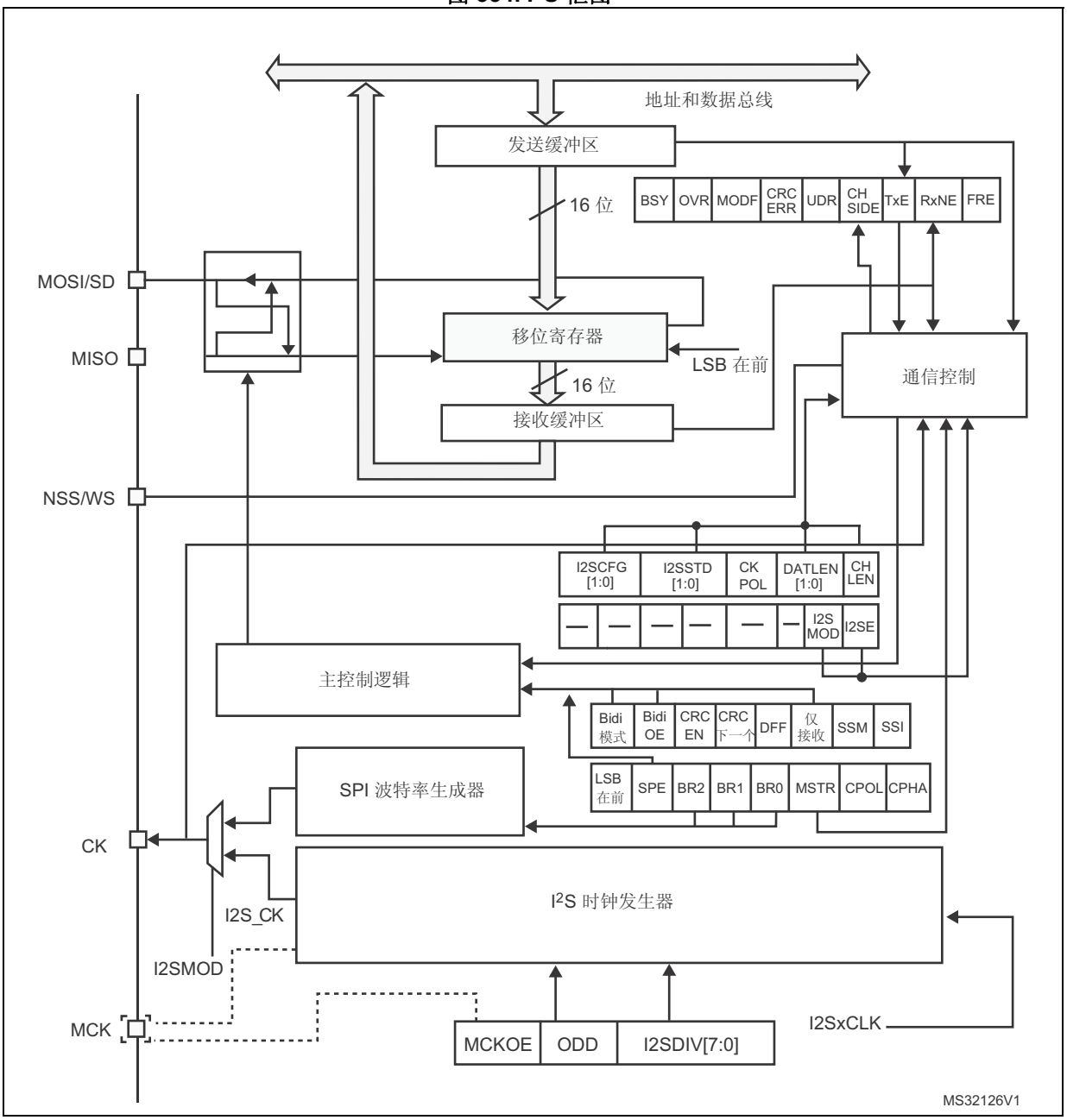
中断事件	事件标志	使能控制位
发送 TXFIFO 准备就绪，可以装载数据	TXE	TXEIE
RXFIFO 中接收了数据	RXNE	RXNEIE
主模式故障	MODF	ERRIE
上溢错误	OVR	
TI 帧格式错误	FRE	
CRC 协议错误	CRCERR	

32.7 I²S 功能说明

32.7.1 I²S 一般说明

I²S 的框图如*图 354* 所示。

图 354. I²S 框图



1. MCK 映射到 MISO 引脚上。

当使能 I²S 功能（将 SPIx_I2SCFGR 寄存器中的 I2SMOD 位置 1）后，SPI 可用作音频 I²S 接口。此接口使用几乎与 SPI 相同的引脚、标志和中断。

I^2S 与 SPI 共用以下三个引脚：

- SD：串行数据（映射到 MOSI 引脚），用于发送或接收两个时分复用的数据通道上的数据（仅半双工模式）。
- WS：字选择（映射到 NSS 引脚），是主模式下的数据控制信号输出以及从模式下的数据控制信号输入。
- CK：串行时钟（映射到 SCK 引脚），是主模式下的串行时钟输出以及从模式下的串行时钟输入。

当某些外部音频设备需要使用主时钟输出时，可以使用其它引脚：

- MCK：当 I^2S 配置为主模式（并且 SPIx_I2SPR 寄存器中的 MCKOE 位置 1）时，使用主时钟（单独映射）输出此附加时钟，该时钟以 256 × f_S 的预配置频率生成，其中 f_S 为音频信号采样频率。

I^2S 在主模式下使用自身的时钟发生器生成通信时钟。此时钟发生器也是主时钟输出的源。在 I^2S 模式下可以使用两个额外的寄存器。一个是时钟发生器配置寄存器 SPIx_I2SPR，另一个是通用 I^2S 配置寄存器 SPIx_I2SCFGR（音频标准、从/主模式、数据格式、数据包帧、时钟极性等）。

在 I^2S 模式下不使用 SPIx_CR1 寄存器和所有 CRC 寄存器。同样，也不使用 SPIx_CR2 寄存器中的 SSOE 位以及 SPIx_SR 中的 MODF 和 CRCERR 位。

I^2S 使用相同的 SPI 寄存器 (SPIx_DR) 进行 16 位数据传输。

32.7.2 支持的音频协议

三线总线仅需要处理通常在左右两个通道上时分复用的音频数据。但是，只有一个 16 位寄存器进行发送和接收。所以，需由软件将与每个通道对应的值写入数据寄存器，以及从数据寄存器中读取数据，并通过检查 SPIx_SR 寄存器中的 CHSIDE 位来识别对应的通道。始终先发送左通道数据，而后再发送右通道数据（对于 PCM 协议来说，CHSIDE 没有意义）。

数据和帧格式组合有四种， 可采用下列格式发送数据：

- 将 16 位数据封装在 16 位帧中
- 将 16 位数据封装在 32 位帧中
- 将 24 位数据封装在 32 位帧中
- 将 32 位数据封装在 32 位帧中

当使用 32 位数据包中的 16 位数据时，前 16 位 (MSB) 为有效位，16 位 LSB 被强制清零，无需任何软件操作或 DMA 请求（只需一个读/写操作）。

如果应用程序首选 DMA，则 24 位和 32 位数据帧需要对 SPIx_DR 寄存器执行两次 CPU 读取或写入操作，或者需要两次 DMA 操作。对于 24 位数据帧，硬件会在低 8 位填充 8 个 0 扩展到 32 位。

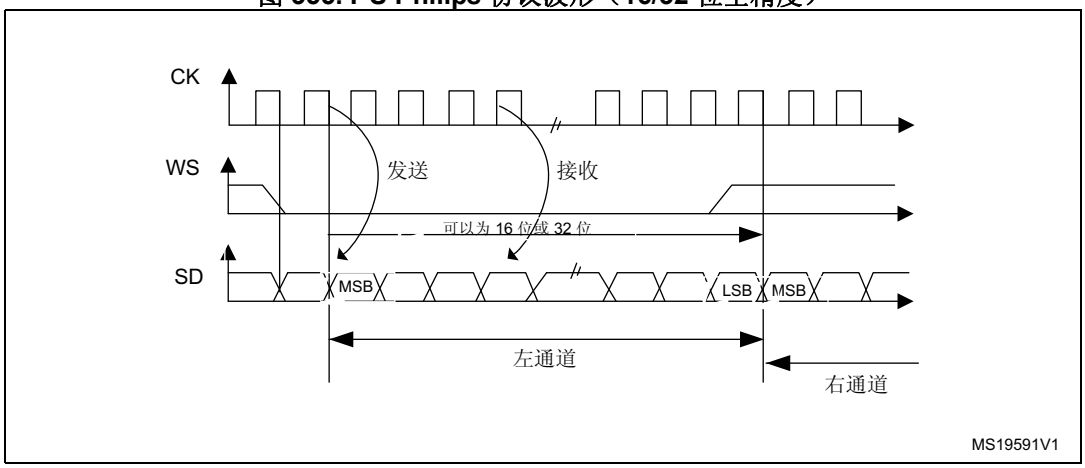
对于所有数据格式和通信标准而言，始终会先发送最高有效位（MSB 优先）。

I^2S 接口支持四种音频标准，可使用 SPIx_I2SCFGR 寄存器中的 I2SSTD[1:0] 和 PCMSYNC 位对其进行配置。

I²S Philips 标准

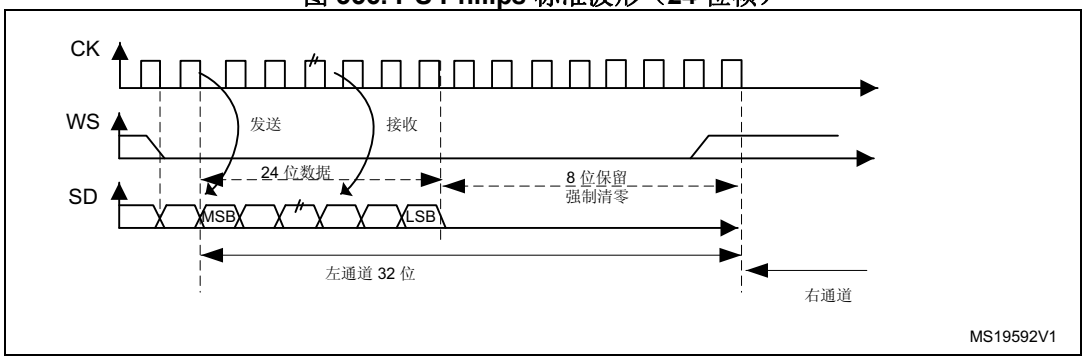
使用 WS 信号来指示当前正在发送的数据所属的通道。该信号从当前通道数据的第一个位 (MSB) 之前的一个时钟开始有效。

图 355. I²S Philips 协议波形（16/32 位全精度）



发送方在时钟信号 (CK) 的下降沿改变数据，接收方在上升沿读取数据。WS 信号也在 CK 的下降沿变化。

图 356. I²S Philips 标准波形（24 位帧）

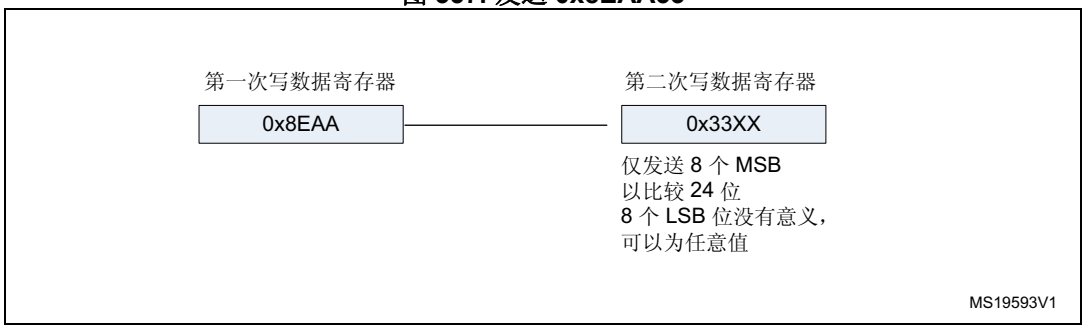


该模式需要对 SPIx_DR 寄存器执行两次写入或读取操作。

- 在发送模式下：

  如果需要发送 0x8EAA33（24 位）：

图 357. 发送 0x8EAA33



- 在接收模式下：
  如果接收数据 0x8EAA33：

图 358. 接收 0x8EAA33

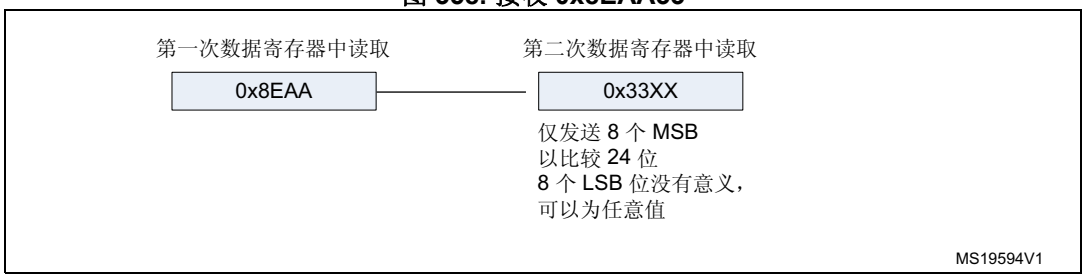
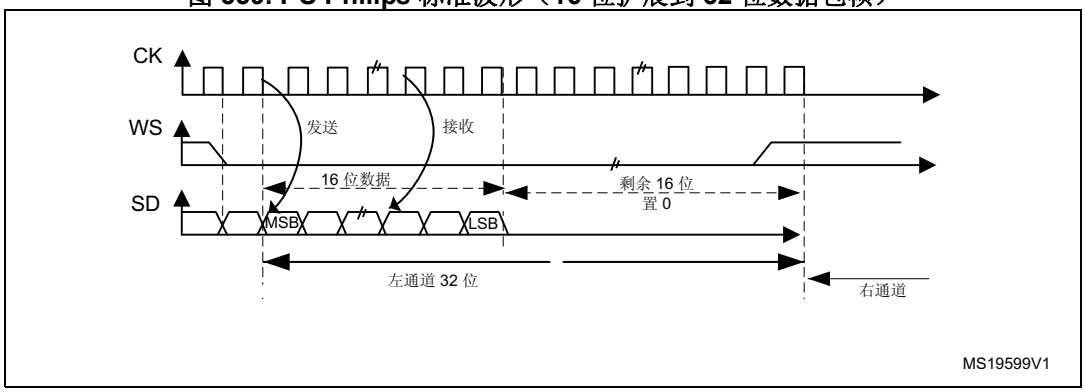


图 359. I2S Philips 标准波形（16 位扩展到 32 位数据包帧）



如果在 I2S 配置阶段选择将 16 位数据帧扩展到 32 位通道帧，则只需要访问一次 SPIx_DR 寄存器。剩余 16 位由硬件置 0 以扩展至 32 位数据格式。

如果要发送的数据或已接收的数据为 0x76A3（0x76A30000 扩展为 32 位），则需要执行*图 360* 中显示的操作。

图 360. 16 位数据帧扩展到 32 位通道帧的示例



发送时，每次将 MSB 写入 SPIx_DR，TXE 标志就会置 1，并在中断使能的情况下触发中断，以将要发送的新数据加载到 SPIx_DR 寄存器。即使硬件填充的低 16 位 0x0000 还未发送，也会如此，因为低 16 位是由硬件发送。

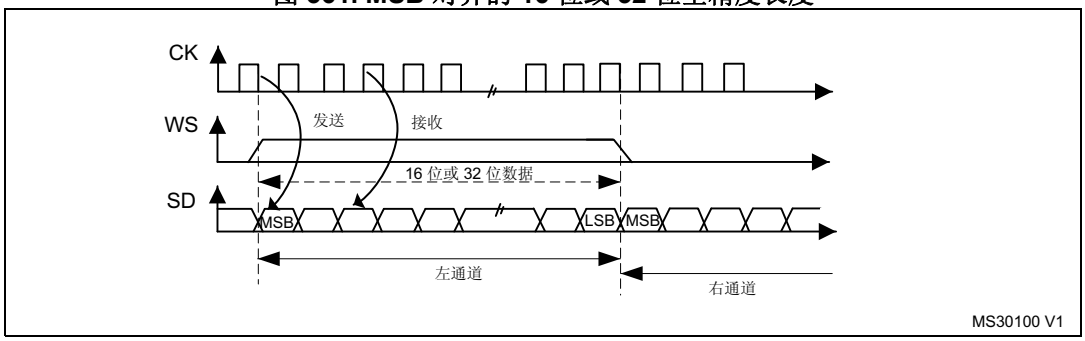
接收时，接收到第一个半字（高 16 位），则硬件将 RXNE 标志置 1，并在中断使能的情况下触发中断。

这样，就延长了两个写入或读取操作之间的时间间隔，从而可防止出现下溢或上溢情况（具体取决于数据传输方向）。

MSB 对齐标准

此标准同时生成 WS 信号和第一个数据位（即 MSBit）。

图 361. MSB 对齐的 16 位或 32 位全精度长度



发送方在时钟信号的下降沿改变数据；接收方在上升沿读取数据。

图 362. MSB 对齐的 24 位帧长度

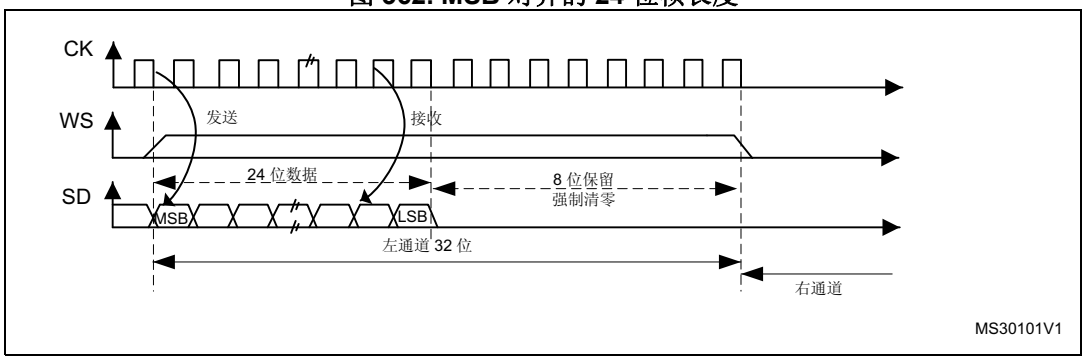
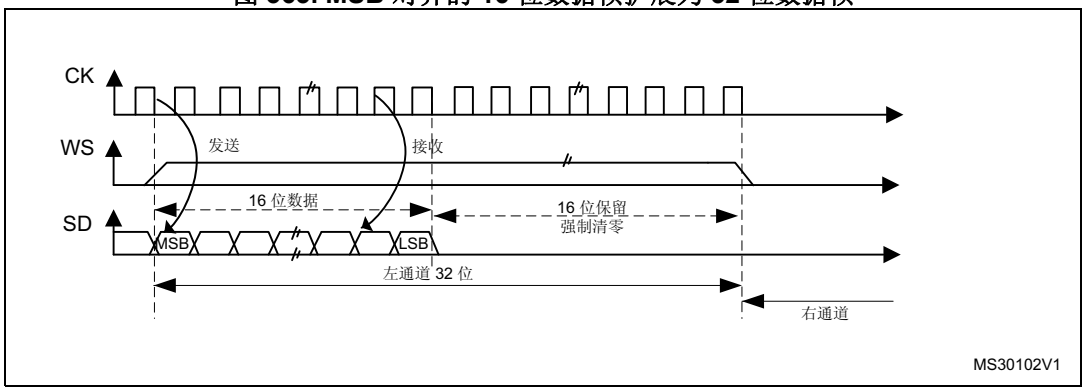


图 363. MSB 对齐的 16 位数据帧扩展为 32 位数据帧



LSB 对齐标准

该标准与 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）。

输入和输出信号的采样与 I2S Philips 标准的采样相同。

图 364. LSB 对齐的 16 位或 32 位全精度

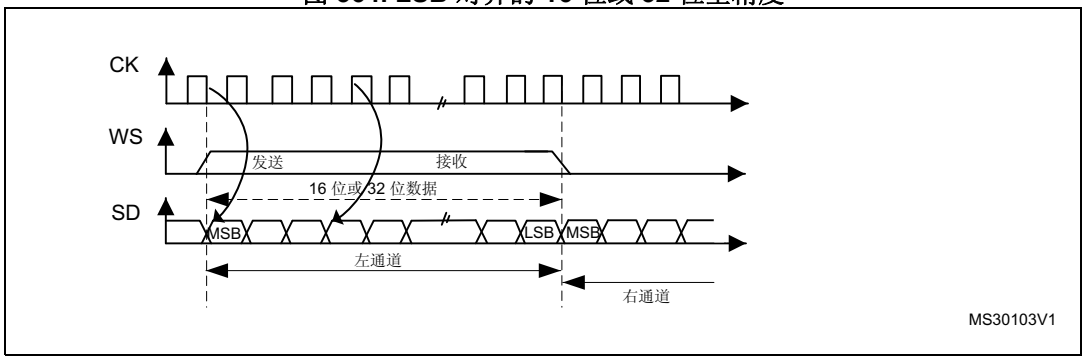
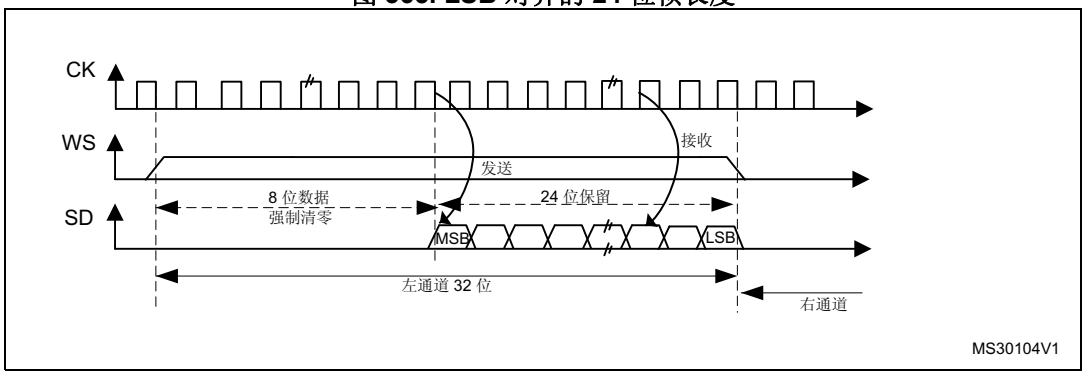


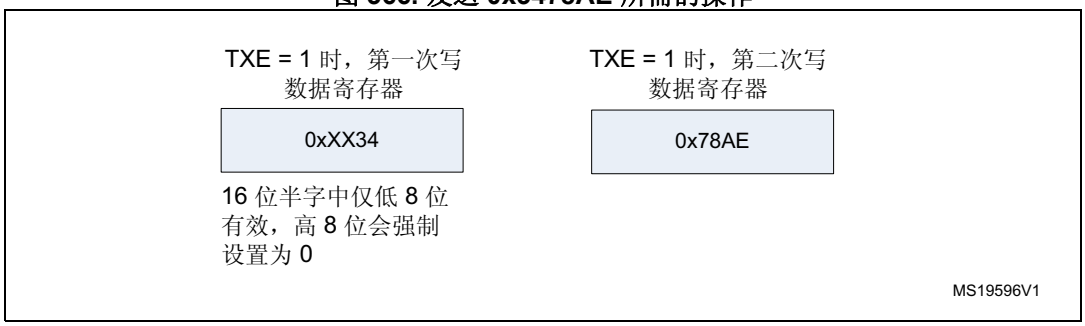
图 365. LSB 对齐的 24 位帧长度



- 在发送模式下：

  如果需要发送数据 0x3478AE，则需要通过软件或 DMA 对 SPIx_DR 寄存器执行两次写入操作。下面给出了这些操作。

图 366. 发送 0x3478AE 所需的操作



- 在接收模式下：
  如果接收到数据 0x3478AE，则在每个 RXNE 事件时需要对 SPIx_DR 寄存器执行两次连续的读取操作。

图 367. 接收 0x3478AE 时所需的操作

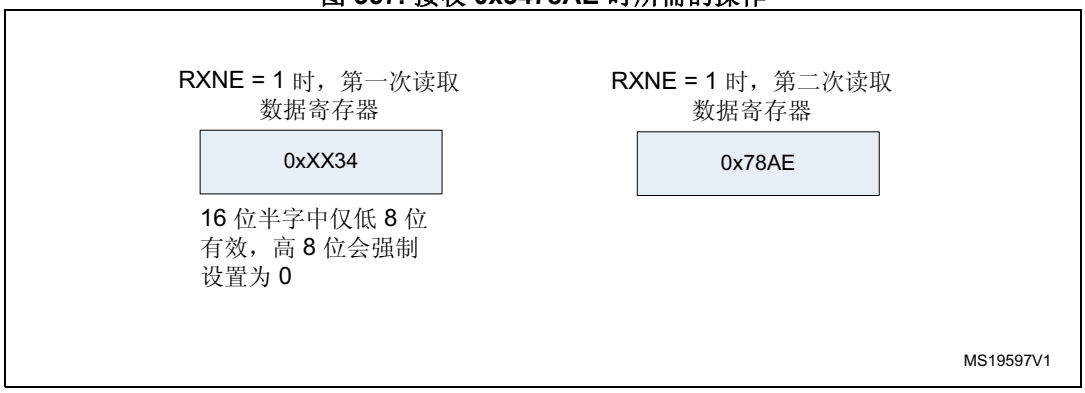
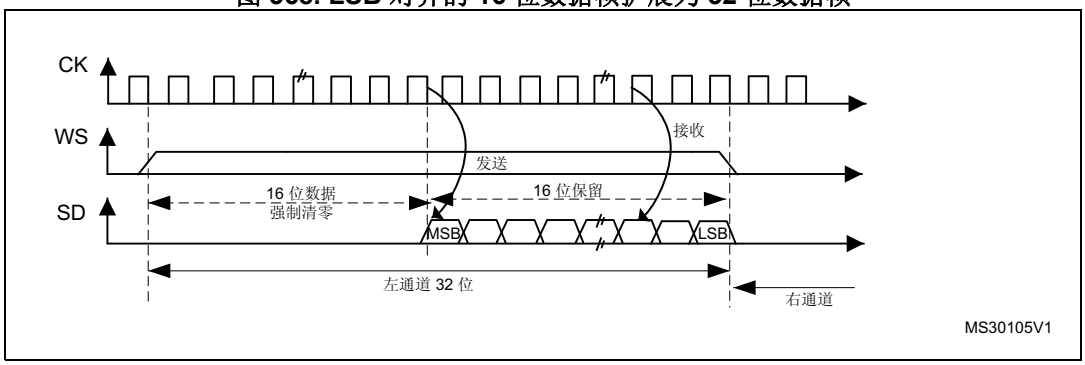


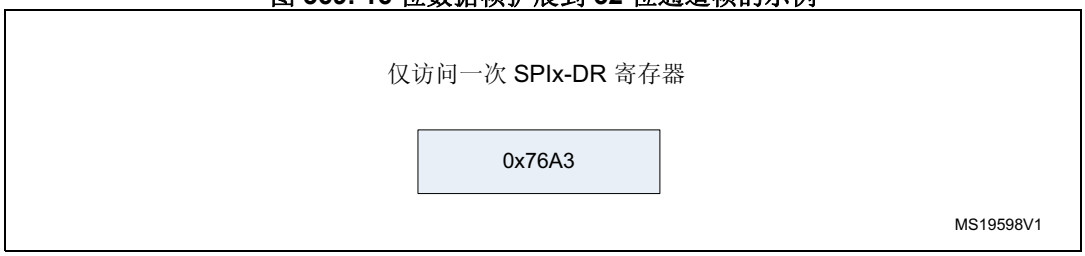
图 368. LSB 对齐的 16 位数据帧扩展为 32 位数据帧



如果在 I^2S 配置阶段选择将 16 位数据帧扩展到 32 位通道帧，则只需要访问一次 SPIx_DR 寄存器。扩展到 32 位中高半字（16 位 MSB）被硬件置为 0x0000。在这种情况下，其对应于半字 MSB。

如果要发送的数据或已接收的数据为 0x76A3（0x0000 76A3 扩展为 32 位），则需要执行*图 369* 中显示的操作。

图 369. 16 位数据帧扩展到 32 位通道帧的示例



在发送模式下，发生 TXE 事件时，应用程序需要写入要发送的数据（此例中，为 0x76A3）。首先发送 0x000 字段（扩展到 32 位）。有效数据 (0x76A3) 发送到 SD 后，TXE 标志会被再次置 1。

在接收模式下，当接收到有效半字后（而非 0x0000 字段），即会置位 RXNE。

这样，就延长了两个写入或读取操作之间的时间间隔，以防止出现下溢或上溢情况。

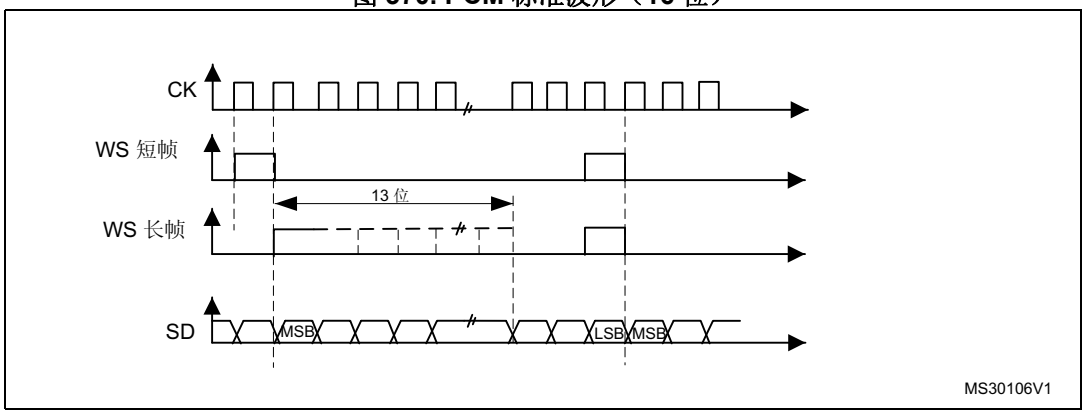
PCM 标准

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 SPIx_I2SCFGR 寄存器中的 PCMSYNC 位来配置。

在 PCM 模式下，输出信号（WS、SD）在 CK 信号的上升沿采样。输入信号（WS、SD）在 CK 的下降沿捕获。

请注意，CK 和 WS 在主模式下配置为输出。

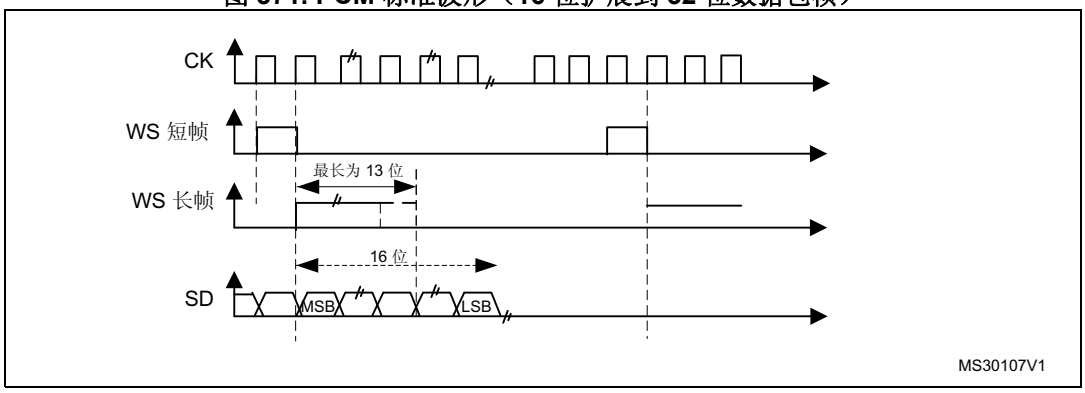
图 370. PCM 标准波形（16 位）



对于长帧同步，在主模式下会将 WS 信号持续 13 个周期。

对于短帧同步，WS 同步信号的持续时间仅为一个周期。

图 371. PCM 标准波形（16 位扩展到 32 位数据包帧）

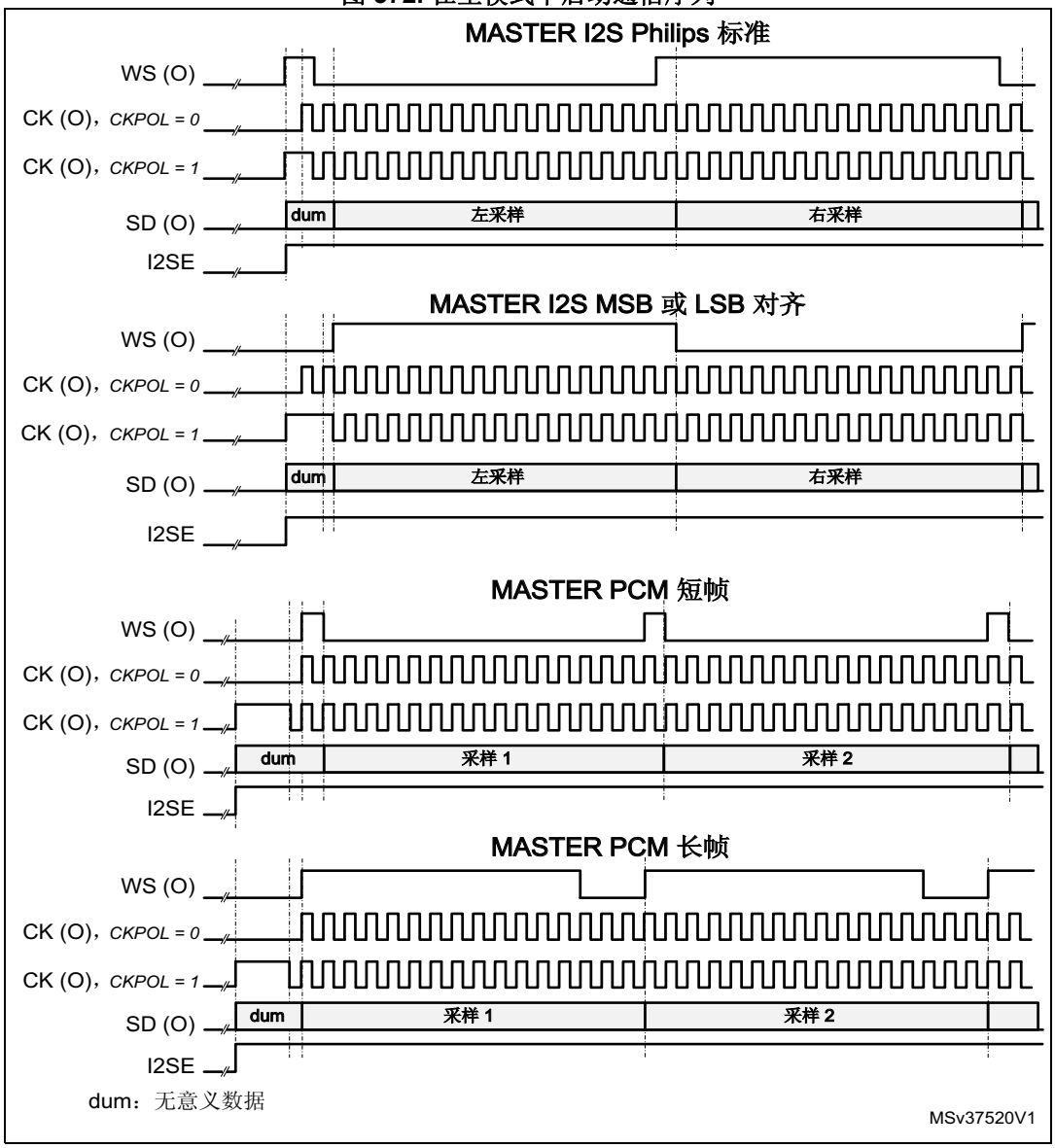


注： *对于两种模式（主/从模式）和两种同步（短/长同步），即使在从模式下，也需要指定两组连续数据（以及两个同步信号）之间位的个数（SPIx_I2SCFGR 寄存器中的 DATLEN 位和 CHLEN 位）。*

32.7.3 启动描述

图 372 显示了通过 I2SE 位使能 SPI/I2S 时，串行接口在主模式下的处理方式。它还显示了 CKPOL 对产生的信号的影响。

图 372. 在主模式下启动通信序列



在从模式下，帧同步检测的方式，取决于 ASTRTEN 位的值

如果 ASTRTEN = 0，当使能音频接口 (I2SE = 1) 时，硬件使用 CK 信号检测输入信号 WS 的相应的边沿。

使用 I2S Philips 标准时，对应的边沿为 WS 信号的下降沿；使用其它标准时，则为上升沿。先采样到 WS 电平为 1，再采样到 WS 电平为 0，检测到下降沿，反之则检测到上升沿。

如果 ASTRTEN = 1，则用户必须在 WS 变为有效前使能音频接口。这意味着在 I2S Philips 标准下 WS = 1 时，或在其它标准下 WS = 0 时，I2SE 位必须置 1。

32.7.4 时钟发生器

I^2S 比特率用来确定 I^2S 数据线上的数据流和 I^2S 时钟信号频率。

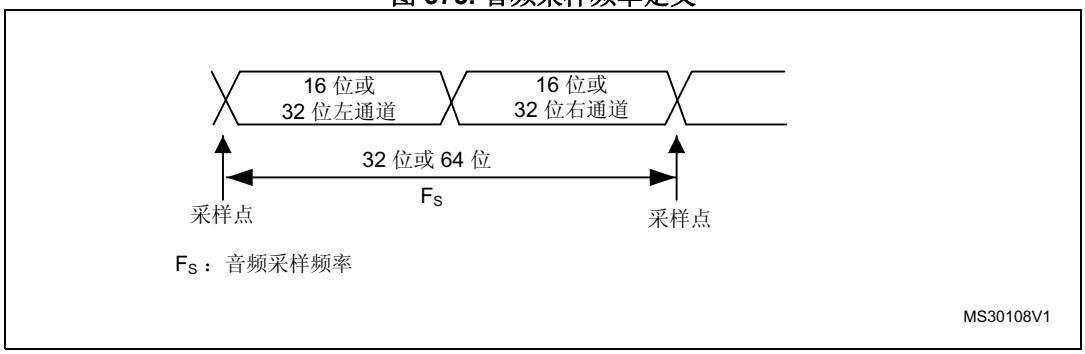
I^2S 比特率 = 每个通道的位数 × 通道数 × 音频采样频率

对于 16 位双通道音频，I^2S 比特率的计算公式如下：

I^2S 比特率 = 16 × 2 × f_S

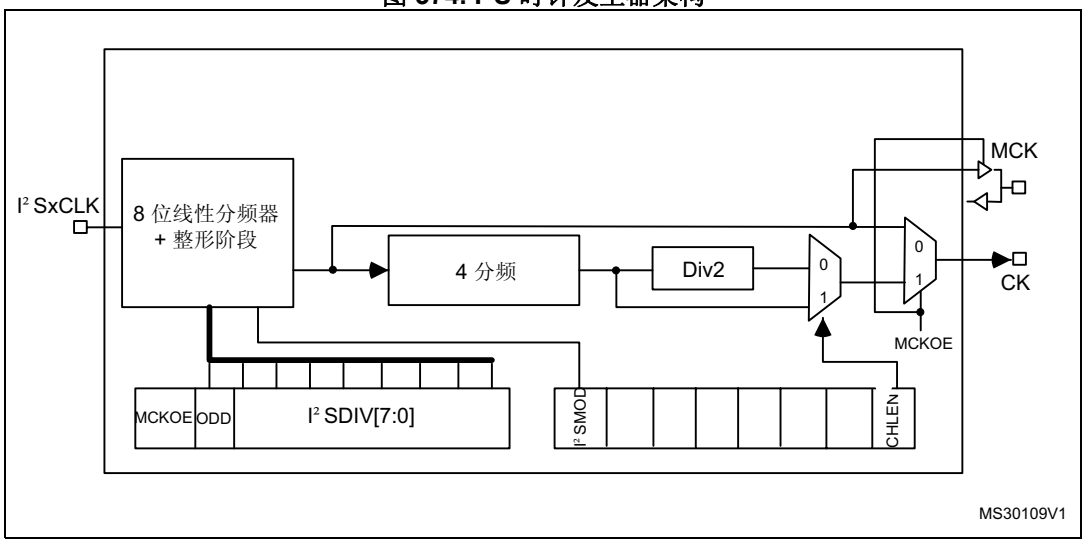
如果数据包为 32 位宽，则 I^2S 比特率 = 32 x 2 x f_S。

图 373. 音频采样频率定义



配置主模式时，需要正确地对线性分频器进行设置，以便采用所需的音频频率进行通信。

图 374. I^2S 时钟发生器架构



1. 其中 X 可以是 2 或 3。

图 374 展示了通信时钟架构。I2Sx 时钟始终为系统时钟。

音频采样频率可以是 192 kHz、96 kHz、48 kHz、44.1 kHz、32 kHz、22.05 kHz、16 kHz、11.025 kHz 或 8 kHz（或此范围内的任何其它值）。为达到所需频率，需要根据以下公式对线性分频器进行编程：

输出主时钟（SPIx_I2SPR 寄存器中的 MCKOE 置 1）时：

f_S = I2SxCLK / [(16*2)*((2*I2SDIV)+ODD)*8)]（通道帧宽度为 16 位时）

f_S = I2SxCLK / [(32*2)*((2*I2SDIV)+ODD)*4)]（通道帧宽度为 32 位时）

禁止主时钟输出（MCKOE 位清零）时：

f_S = I2SxCLK / [(16*2)*((2*I2SDIV)+ODD)]（通道帧宽度为 16 位时）

f_S = I2SxCLK / [(32*2)*((2*I2SDIV)+ODD)]（通道帧宽度为 32 位时）

表 177 提供了针对不同时钟配置的示例精度值。

注： *还可以采用其它配置以达到更好的时钟精度。*

表 177. 使用标准 8 MHz HSE 时的音频频率精度(1)

SYSCLK (MHz)	数据长度	I2SDIV	I2SODD	MCLK	目标 fs (Hz)	实际 fs (kHz)	误差
48	16	8	0	无	96000	93750	2.3438%
48	32	4	0	无	96000	93750	2.3438%
48	16	15	1	无	48000	48387.0968	0.8065%
48	32	8	0	无	48000	46875	2.3438%
48	16	17	0	无	44100	44117.647	0.0400%
48	32	8	1	无	44100	44117.647	0.0400%
48	16	23	1	无	32000	31914.8936	0.2660%
48	32	11	1	无	32000	32608.696	1.9022%
48	16	34	0	无	22050	22058.8235	0.0400%
48	32	17	0	无	22050	22058.8235	0.0400%
48	16	47	0	无	16000	15957.4468	0.2660%
48	32	23	1	无	16000	15957.447	0.2660%
48	16	68	0	无	11025	11029.4118	0.0400%
48	32	34	0	无	11025	11029.412	0.0400%
48	16	94	0	无	8000	7978.7234	0.2660%
48	32	47	0	无	8000	7978.7234	0.2660%
48	16	2	0	有	48000	46875	2.3430%
48	32	2	0	有	48000	46875	2.3430%
48	16	2	0	有	44100	46875	6.2925%
48	32	2	0	有	44100	46875	6.2925%
48	16	3	0	有	32000	31250	2.3438%
48	32	3	0	有	32000	31250	2.3438%
48	16	4	1	有	22050	20833.333	5.5178%
48	32	4	1	有	22050	20833.333	5.5178%
48	16	6	0	有	16000	15625	2.3438%
48	32	6	0	有	16000	15625	2.3438%

表 177. 使用标准 8 MHz HSE 时的音频频率精度(1)（续）

SYSCLK (MHz)	数据长度	I2SDIV	I2SODD	MCLK	目标 fs (Hz)	实际 fs (kHz)	误差
48	16	8	1	有	11025	11029.4118	0.0400%
48	32	8	1	有	11025	11029.4118	0.0400%
48	16	11	1	有	8000	8152.17391	1.9022%
48	32	11	1	有	8000	8152.17391	1.9022%

1. 该表格仅给出了不同时钟配置的示例值。还可以采用其他配置以达到更好的时钟精度。

32.7.5 I²S 主模式

I²S 可配置为主模式。这意味着将在 CK 引脚输出串行时钟，在 WS 引脚生成字选信号。主时钟 (MCK) 可以输出，也可以不输出，具体由 SPIx_I2SPR 寄存器中的 MCKOE 位控制。

步骤

1. 设置 SPIx_I2SPR 寄存器的 I2SDIV[7:0] 位，以定义串行时钟波特率，从而达到适当的音频采样频率。SPIx_I2SPR 寄存器的 ODD 位也需要设置。
2. 设置 CKPOL 位，定义时钟在空闲时的电平状态。如果需要为外部 DAC/ADC 音频组件提供主时钟 MCK，则将 SPIx_I2SPR 寄存器的 MCKOE 位置 1（I2SDIV 和 ODD 值应根据 MCK 输出的状态进行计算。有关详细信息，请参见*第 32.7.4 节：时钟发生器*）。
3. 将 SPIx_I2SCFGR 寄存器中的 I2SMOD 位置 1 以激活 I²S 功能，通过 I2SSTD[1:0] 和 PCMSYNC 位选择 I²S 标准，通过 DATLEN[1:0] 位选择数据长度并通过配置 CHLEN 位选择每个通道的位数。此外，通过 SPIx_I2SCFGR 寄存器的 I2SCFG[1:0] 位选择 I²S 主模式和方向（发送器或接收器）。
4. 如果需要，通过对 SPIx_CR2 寄存器执行写操作来选择所有可能的中断源和 DMA 功能。
5. SPIx_I2SCFGR 寄存器的 I2SE 位必须置 1。

WS 和 CK 配置为输出模式。如果 SPIx_I2SPR 的 MCKOE 位置 1，则 MCK 也是输出。

发送序列

将半字写入发送缓冲区后，发送序列随即开始。

假设写入发送缓冲区的第一个数据对应于左通道数据。数据从发送缓冲区传输到移位寄存器时，TXE 置 1，并且必须将对应于右通道的数据写入发送缓冲区。CHSIDE 标志指示将发送的数据对应的通道。TXE 标志置 1 时，CHSIDE 标志有意义，因为该标志在 TXE 变为高电平时进行更新。

一个完整帧表示先进行左通道数据发送再进行右通道数据发送。不存在仅发送左通道的部分帧。

首位发送期间，数据按半字并行加载到 16 位移位寄存器中，然后以串行方式移位并输出到 MOSI/SD 引脚（MSB 在前）。每次数据从发送缓冲区传输到移位寄存器后，TXE 标志都将置 1，如果 SPIx_CR2 寄存器的 TXEIE 位置 1，将产生中断。

有关各种 I²S 标准模式的写操作的更多详细信息，请参见*第 32.7.2 节：支持的音频协议*。

为确保连续进行音频数据发送，必须在当前数据发送结束前将下一个要发送的数据写入 SPIx_DR 寄存器。

要通过将 I2SE 清零来关闭 I²S，必须等待 TXE = 1 且 BSY = 0。

接收序列

此工作模式与发送模式相同，只有第 3 点存在不同（请参见*第 32.7.5 节：I²S 主模式*所述的步骤），即通过 I2SCFG[1:0] 位设置主器件接收模式。

无论数据或通道长度如何，音频数据始终按 16 位数据包进行接收。这意味着，每当接收缓冲区满时，RXNE 标志即置 1，并且如果 SPIx_CR2 寄存器的 RXNEIE 位置 1，还将产生中断。所接收的右通道或左通道的音频值可通过一次或两次接收操作进入接收缓冲区，具体取决于数据和通道长度配置。

读取 SPIx_DR 寄存器即会使 RXNE 位清零。

CHSIDE 在每次接收后进行更新。它由 I²S 单元所产生的 WS 信号触发。

有关各种 I²S 标准模式中读操作的更多详细信息，请参见*第 32.7.2 节：支持的音频协议*。

如果在先前收到的数据尚未读取时又接收到新数据，将产生上溢错误并将 OVR 标志置 1。如果 SPIx_CR2 寄存器的 ERRIE 位置 1，将产生中断以指示该错误。

要关闭 I²S，需要执行特定操作来确保 I²S 正确完成传输周期而不启动新的数据传输。该序列取决于数据和通道长度的配置，以及所选的音频协议模式。在以下情况下：

- 32 位通道长度上扩展的 16 位数据长度（DATLEN = 00 且 CHLEN = 1），使用 LSB 对齐模式 (I2SSTD = 10)
  a) 等待倒数第二个 RXNE = 1 (n – 1)
  b) 然后等待 17 个 I²S 时钟周期（使用软件循环）
  c) 禁止 I²S (I2SE = 0)
- 32 位通道长度上扩展的 16 位数据长度（DATLEN = 00 且 CHLEN = 1），使用 MSB 对齐、I²S 或 PCM 模式（分别为 I2SSTD = 00、I2SSTD = 01 或 I2SSTD = 11）
  a) 等待最后一个 RXNE
  b) 然后等待 1 个 I²S 时钟周期（使用软件循环）
  c) 禁止 I²S (I2SE = 0)
- 对于 DATLEN 和 CHLEN 的所有其它组合，无论通过 I2SSTD 位选择何种音频模式，都将执行以下序列来关闭 I²S：
  a) 等待倒数第二个 RXNE = 1 (n – 1)
  b) 然后等待一个 I²S 时钟周期（使用软件循环）
  c) 禁止 I²S (I2SE = 0)

注： *传输期间，BSY 标志保持低电平。*

32.7.6 I²S 从模式

对于从配置而言，I²S 可配置为发送模式或接收模式。此工作模式所遵循的规则与 I²S 主模式配置基本相同。在从模式下，I²S 接口不产生时钟。时钟和 WS 信号从 I²S 接口所连接的外部主器件输入。这样，用户便不需要配置时钟。

应遵循如下配置步骤：

1. 将 SPIx_I2SCFGR 寄存器的 I2SMOD 位置 1 以选择 I²S 模式，通过 I2SSTD[1:0] 位选择 I²S 标准，通过 DATLEN[1:0] 位选择数据长度并通过配置 CHLEN 位选择帧中每个通道的位数。此外，通过 SPIx_I2SCFGR 寄存器的 I2SCFG[1:0] 位选择从器件的模式（发送或接收）。
2. 如果需要，通过对 SPIx_CR2 寄存器执行写操作来选择所有可能的中断源和 DMA 功能。
3. SPIx_I2SCFGR 寄存器的 I2SE 位必须置 1。

发送序列

当外部主器件发送时钟并且通过 NSS_WS 信号请求传输数据时，发送序列开始。必须首先使能从器件，然后外部主器件才能开始通信。主器件开始通信前，还必须加载 I²S 数据寄存器。

对于 I²S、MSB 对齐和 LSB 对齐模式，要写入数据寄存器的第一个数据项对应于左通道的数据。通信开始时，数据从发送缓冲区传输到移位寄存器。TXE 标志随即置 1，以请求将右通道的数据写入 I²S 数据寄存器。

CHSIDE 标志指示将发送的数据对应的通道。与主发送模式相比，在从模式下，CHSIDE 由来自外部主器件的 WS 信号触发。这意味着，从器件需要首先为发送第一个数据做好准备，然后主器件才能产生时钟。WS 置位意味着首先发送左通道数据。

注： *必须要在主器件发出的第一个时钟出现在 CK 线上至少 2 个 PCLK 周期之前置位 I2SE。*

首位发送期间，数据按半字从内部总线并行加载到 16 位移位寄存器中，然后以串行方式移位并输出到 MOSI/SD 引脚（MSB 在前）。每次数据从发送缓冲区传输到移位寄存器后，TXE 标志都将置 1，如果 SPIx_CR2 寄存器的 TXEIE 位置 1，将产生中断。

请注意，仅当 TXE 标志为 1 时，才可以尝试向发送缓冲区写入数据。

有关各种 I²S 标准模式中写操作的更多详细信息，请参见*第 32.7.2 节：支持的音频协议*。

为确保连续进行音频数据发送，必须在当前数据发送结束前将下一个要发送数据写入 SPIx_DR 寄存器。如果在数据尚未写入 SPIx_DR 寄存器时下一个数据通信的首个时钟边沿到来，下溢标志将置 1 并可能产生中断。通过这种方式，软件可以获知所传输的数据不正确。如果 SPIx_CR2 寄存器的 ERRIE 位置 1，则当 SPIx_SR 寄存器中的 UDR 标志变为 1 时，将产生中断。这种情况下，必须关闭 I²S 并从左通道开始重新启动数据传输。

要通过将 I2SE 位清零来关闭 I²S，必须等待 TXE = 1 且 BSY = 0。

接收序列

此工作模式与发送模式相同，只有第 1 点存在不同（请参见*第 32.7.6 节：I²S 从模式*所述的步骤），即通过 SPIx_I2SCFGR 寄存器的 I2SCFG[1:0] 位设置从器件接收模式。

无论数据长度或通道长度如何，音频数据始终按 16 位数据包进行接收。这意味着，每当接收缓冲区填满时，SPIx_SR 寄存器中的 RXNE 标志即置 1，并且如果 SPIx_CR2 寄存器的 RXNEIE 位置 1，还将产生中断。所接收的右通道或左通道的音频值可能通过一次或两次接收操作进入接收缓冲区，具体取决于数据长度和通道长度配置。

每次接收要从 SPIx_DR 寄存器读取的数据时，CHSIDE 标志都将更新。该标志由外部主器件所管理的外部 WS 线路触发。

读取 SPIx_DR 寄存器即会使 RXNE 位清零。

有关各种 I²S 标准模式的读操作的更多详细信息，请参见*第 32.7.2 节：支持的音频协议*。

如果在先前收到的数据尚未读取时又接收到新数据，将产生上溢错误，OVR 标志将置 1。如果 SPIx_CR2 寄存器的 ERRIE 位置 1，将产生中断以指示该错误。

要在接收模式下关闭 I²S，必须在接收到最后一个 RXNE = 1 后立即将 I2SE 清零。

注： *外部主器件应能够通过音频通道以 16 位或 32 位数据包发送/接收数据。*

32.7.7 I^2S 状态标志

应用程序可通过三个状态标志来全面监视 I^2S 总线的状态。

忙标志 (BSY)

BSY 标志由硬件置 1 和清零（写入此标志没有任何作用）。该标志表示 I^2S 通信层的状态。

BSY 置 1 时，表示 I^2S 正在忙于通信。在主接收模式 (I2SCFG = 11) 中，BSY 标志的情况例外，该标志在接收期间仍保持低电平。

如果软件需要禁止 I^2S，可使用 BSY 标志检测传输是否结束。这可以避免损坏最后传输的数据。为此，必须严格遵循下述步骤。

在传输开始时（I^2S 处于主接收器模式时除外），将 BSY 标志置 1。

出现以下情况时，BSY 标志被硬件清零：

- 传输完成时（主发送模式除外，在该模式下通信是连续的）
- 禁止 I^2S 时

当通信连续时：

- 在主发送模式下，BSY 标志在所有传输期间均保持高电平
- 在从模式下，BSY 标志在每次传输之间变为低电平并持续一个 I^2S 时钟周期

注： *请勿使用 BSY 标志处理每次数据发送或接收，最好改用 TXE 标志和 RXNE 标志。*

发送缓冲区为空 (TXE)

如果此标志置 1，表示发送缓冲区为空，可将要发送的下一个数据加载到其中。发送缓冲区已包含要发送的数据时，TXE 标志复位。禁止 I^2S（I2SE 位复位）时，该标志也会复位。

接收缓冲区非空 (RXNE)

此标志置 1 时，表示接收缓冲区中存在有效的已接收数据。读取 SPIx_DR 寄存器时，该标志复位。

通道方向 (CHSIDE)

在发送模式下，此标志将在 TXE 变为高电平时进行刷新。此标志指示 SD 上要传输的数据所属的通道。如果在从发送模式下发生下溢错误事件，此标志将不可靠，在恢复通信前需要关闭并重新开启 I^2S。

在接收模式下，该标志将在 SPIx_DR 中接收数据时进行刷新。它表示接收的数据所属的通道。请注意，如果发生错误（例如 OVR），此标志将失去意义，应通过禁止并重新使能 I^2S（根据需要更改配置）来复位。

此标志在 PCM 标准中没有意义（短帧和长帧模式）。

当 SPIx_SR 中的 OVR 或 UDR 标志置 1，并且 SPIx_CR2 中的 ERRIE 位也置 1 时，将产生中断。中断源被清除后，可通过读取 SPIx_SR 状态寄存器来将此中断清除。

32.7.8 I²S 错误标志

I²S 单元共有三个错误标志。

下溢标志 (UDR)

在从发送模式下，如果在软件尚未将任何值加载到 SPIx_DR 之前出现第一个数据发送时钟，此标志将置 1。SPIx_I2SCFGR 寄存器中的 I2SMOD 位置 1 时，可以使用此标志。如果 SPIx_CR2 寄存器中的 ERRIE 位置 1，可产生中断。
UDR 位通过 SPIx_SR 寄存器上的读操作进行清零。

上溢标志 (OVR)

如果在尚未从 SPIx_DR 寄存器读取上一个数据时又接收到新数据，此标志将置 1。因此，传入的数据将丢失。如果 SPIx_CR2 寄存器中的 ERRIE 位置 1，可产生中断。

这种情况下，将不会用接收到的新数据更新接收缓冲区的内容。对 SPIx_DR 寄存器执行的读操作将返回先前正确接收的数据。主器件后续发送的所有其它半字都将丢失。

要将 OVR 位清零，应首先对 SPIx_DR 寄存器执行读操作，然后再对 SPIx_SR 寄存器进行读访问。

帧错误标志 (FRE)

仅当 I²S 配置为从模式时，此标志才可由硬件置 1。如果外部主器件没有按照从器件期望的那样改变 WS 线，则此标志将置 1。如果失去同步，要从此状态中恢复并将外部主器件与 I²S 从器件重新同步，需执行下列步骤：

1. 禁止 I²S。
2. 重新使能 I2S 接口（保持 ASTRTEN=0）。

主器件与从器件之间的同步失效可能是由于 CK 通信时钟或 WS 帧同步信号线上存在噪音干扰。如果 ERRIE 位置 1，可产生错误中断。读取状态寄存器时，同步失效标志 (FRE) 由软件清零。

32.7.9 DMA 特性

在 I²S 模式下，DMA 的工作方式与在 SPI 模式下完全相同。除了由于不存在数据传输保护机制，I²S 模式下没有 CRC 功能外，没有其它差别。

32.8 I²S 中断

表 178 为 I²S 中断的列表。

表 178. I²S 中断请求

中断事件	事件标志	使能控制位
发送缓冲区为空	TXE	TXEIE
接收缓冲区非空	RXNE	RXNEIE
上溢错误	OVR	ERRIE
下溢错误	UDR	
帧错误	FRE	

32.9 SPI 和 I^2S 寄存器

外设寄存器可支持半字（16 位）或字（32 位）访问。此外，SPI_DR 可支持 8 位访问。

32.9.1 SPIx 控制寄存器 1 (SPI_CR1)

SPI 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
BIDI MODE	BIDI OE	CRC EN	CRC NEXT	CRCL	RX ONLY	SSM	SSI	LSB FIRST	SPE	BR [2:0]			MSTR	CPOL	CPHA
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15 **BIDIMODE**：双向通信数据模式使能 (Bidirectional data mode enable) 该位通过一条共用的双向数据线来支持半双工通信。双向模式激活时，使 RXONLY 位保持清零状态。

0：选择双线单向通信数据模式

1：选择单线双向数据模式

注： 该位不适用于 I^2S 模式。

位 14 **BIDIOE**：双向模式下的输出使能 (Output enable in bidirectional mode)

该位结合 BIDIMODE 位，用于选择双向模式下的传输方向

0：禁止输出（只接收模式）

1：使能输出（只发送模式）

注： 在主模式下，使用 MOSI 引脚；在从模式下，使用 MISO 引脚。

该位不适用于 I^2S 模式。

位 13 **CRCEN**：硬件 CRC 计算使能 (Hardware CRC calculation enable)

0：禁止 CRC 计算

1：使能 CRC 计算

注： 为确保正确操作，只应在禁止 SPI（SPE = “0”）时对此位执行写操作。

该位不适用于 I^2S 模式。

位 12 **CRCNEXT**：发送下一个 CRC (Transmit CRC next)

0：下一个发送值来自发送缓冲区

1：下一个发送值来自发送 CRC 寄存器

注： 向 SPIx_DR 寄存器写入最后一个数据后，必须立即对该位执行写操作。

该位不适用于 I^2S 模式。

位 11 **CRCL**：CRC 长度 (CRC length)

此位由软件置 1 和清零，用以选择 CRC 长度。

0：8 位 CRC 长度

1：16 位 CRC 长度

注： 为确保正确操作，只应在禁止 SPI（SPE = “0”）时对此位执行写操作。

该位不适用于 I^2S 模式。

位 10 **RXONLY**：只接收模式使能 (Receive only mode enabled)。
该位用于使能通过一条双向线专门接收数据的单工通信。当只接收模式激活时，将 BIDIMODE 位保持清零状态。此位也适用于多从模式系统，在此类系统中，不会访问特定从器件，也不会损坏访问的从器件的输出。
0：全双工（发送和接收）
1：禁止输出（只接收模式）
注：　该位不适用于 I^2S 模式。

位 9 **SSM**：软件从器件管理 (Software slave management)
当 SSM 位置 1 时，NSS 引脚输入替换为 SSI 位的值。
0：禁止软件从器件管理
1：使能软件从器件管理
注：　该位不适用于 I^2S 模式 和 SPI TI 模式。

位 8 **SSI**：内部从器件选择 (Internal slave select)
仅当 SSM 位置 1 时，此位才有效。此位的值将作用到 NSS 引脚上，并忽略 NSS 引脚的 I/O 值。
注：　该位不适用于 I^2S 模式 和 SPI TI 模式。

位 7 **LSBFIRST**：帧格式 (Frame format)
0：发送/接收数据时 MSB 在前
1：发送/接收数据时 LSB 在前
注：　1. 正在通信时不应更改此位。
2. 该位不适用于 I^2S 模式 和 SPI TI 模式。

位 6 **SPE**：SPI 使能 (SPI enable)
0：禁止外设
1：使能外设
注：　禁止 SPI 时，请按照第 976 页上的禁止 SPI 的步骤中所述的步骤操作。
该位不适用于 I^2S 模式。

位 [5:3] **BR[2:0]**：波特率控制 (Baud rate control)
000：$f_{PCLK}/2$
001：$f_{PCLK}/4$
010：$f_{PCLK}/8$
011：$f_{PCLK}/16$
100：$f_{PCLK}/32$
101：$f_{PCLK}/64$
110：$f_{PCLK}/128$
111：$f_{PCLK}/256$
注：　正在通信时不应更改这些位。
该位不适用于 I^2S 模式。

位 2 **MSTR**：主模式选择 (Master selection)
0：从配置
1：主配置
注：　正在通信时不应更改此位。
该位不适用于 I^2S 模式。

位 1 **CPOL**：时钟极性 (Clock Polarity)

0：空闲状态时，CK 为 0

1：空闲状态时，CK 为 1

注：　正在通信时不应更改此位。

该位不适用于 SPI TI 模式。

位 0 **CPHA**：时钟相位 (Clock phase)

0：从第一个时钟边沿开始捕获数据

1：从第二个时钟边沿开始捕获数据

注：　正在通信时不应更改此位。

该位不适用于 SPI TI 模式。

32.9.2 SPIx 控制寄存器 2 (SPI_CR2)

SPI control register 2

偏移地址：0x04

复位值：0x0700

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	LDMA_TX	LDMA_RX	FRXTH	DS [3:0]				TXEIE	RXNEIE	ERRIE	FRF	NSSP	SSOE	TXDMAEN	RXDMAEN
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15 保留，必须保持复位值。

位 14 **LDMA_TX**：发送的最后一次 DMA 传输 (Last DMA transfer for transmission)

此位用于数据封装模式，用于定义通过 DMA 发送的数据总数为奇数还是偶数。只有 SPIx_CR2 寄存器中的 TXDMAEN 位置 1 且使用封装模式（数据长度 =< 8 位，对 SPIx_DR 的写访问为 16 位宽）时，该位才有意义。当 SPI 禁止时（SPIx_CR1 寄存器中的 SPE = 0），必须对其执行写操作。

0：待传输数据数量为偶数

1：待传输数据数量为奇数

注：　如果 CRCEN 置 1，请参见第 976 页上的禁止 SPI 的步骤。

该位不适用于 I²S 模式。

位 13 **LDMA_RX**：接收的最后一次 DMA 传输 (Last DMA transfer for reception)

此位用于数据封装模式下，用于定义通过 DMA 接收的数据总数为奇数还是偶数。只有 SPIx_CR2 寄存器中的 RXDMAEN 位置 1 且使用封装模式（数据长度 =< 8 位，对 SPIx_DR 的写访问为 16 位宽）时，该位才有意义。当 SPI 禁止时（SPIx_CR1 寄存器中的 SPE = 0），必须对其执行写操作。

0：待传输数据数量为偶数

1：待传输数据数量为奇数

注：　如果 CRCEN 位置 1，请参见第 976 页上的禁止 SPI 的步骤。

该位不适用于 I²S 模式。

位 12 **FRXTH**：FIFO 接收阈值 (FIFO reception threshold)

该位用于设置触发 RXNE 事件的 RXFIFO 阈值

0：如果 FIFO 占用水平大于或等于 1/2（16 位），将生成 RXNE 事件。

1：如果 FIFO 占用水平大于或等于 1/4（8 位），将生成 RXNE 事件。

注：　该位不适用于 I²S 模式。

位 11:8 **DS [3:0]**：数据大小 (Data size)

这些位用于配置 SPI 传输的数据长度：

0000：未使用
0001：未使用
0010：未使用
0011：4 位
0100：5 位
0101：6 位
0110：7 位
0111：8 位
1000：9 位
1001：10 位
1010：11 位
1011：12 位
1100：13 位
1101：14 位
1110：15 位
1111：16 位

如果软件尝试写入其中一个“未使用”值，这些位将被强制设为“0111”（8 位）。

注： 该位不适用于 I²S 模式。

位 7 **TXEIE**：发送缓冲区空中断使能 (Tx buffer empty interrupt enable)

0：屏蔽 TXE 中断
1：使能 TXE 中断。TXE 标志置 1 时产生中断请求。

位 6 **RXNEIE**：接收缓冲区非空中断使能 (RX buffer not empty interrupt enable)

0：屏蔽 RXNE 中断
1：使能 RXNE 中断 RXNE 标志置 1 时产生中断请求。

位 5 **ERRIE**：错误中断使能 (Error interrupt enable)

此位用于在出现错误条件（SPI 模式下的 CRCERR、OVR 和 MODF；TI 模式下的 FRE；以及 I^2S 模式下的 UDR、OVR 和 FRE）时控制中断的生成。

0：屏蔽错误中断
1：使能错误中断

位 4 **FRF**：帧格式 (Frame format)

0：SPI Motorola 模式
1：SPI TI 模式

注： 只有在禁止 SPI (SPE=0) 后才能对此位执行写操作。
该位不适用于 I^2S 模式。

位 3 **NSSP**：NSS 脉冲管理 (NSS pulse management)

此位仅用于主模式。连续传输时，该位允许 SPI 在两个连续数据间生成 NSS 脉冲。单次数据传输时，该位强制 NSS 引脚在传输后变为高电平。

如果 CPHA =“1”或 FRF =“1”，该位无意义。

0：未生成 NSS 脉冲
1：生成 NSS 脉冲

注： 1. 只有在禁止 SPI (SPE=0) 后才能对此位执行写操作。
2. 该位不适用于 I^2S 模式 和 SPI TI 模式。

位 2 **SSOE**：SS 输出使能 (SS output enable)
0：在主模式下禁止 SS 输出，SPI 接口可在多主模式配置下工作
1：在主模式下以及使能 SPI 接口后使能 SS 输出。SPI 接口不能在多主模式环境下工作。
注： 该位不适用于 I²S 模式 和 SPI TI 模式。

位 1 **TXDMAEN**：发送缓冲区 DMA 使能 (Tx buffer DMA enable)
当此位置 1 时，每当 TXE 标志置 1，即产生 DMA 请求。
0：禁止发送缓冲区 DMA
1：使能发送缓冲区 DMA

位 0 **RXDMAEN**：接收缓冲区 DMA 使能 (Rx buffer DMA enable)
当此位置 1 时，每当 RXNE 标志置 1，即产生 DMA 请求。
0：禁止接收缓冲区 DMA
1：使能接收缓冲区 DMA

32.9.3 SPI 状态寄存器 (SPIx_SR)

SPI status register

偏移地址：0x08

复位值：0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	FTLVL[1:0]		FRLVL[2:0]		FRE	BSY	OVR	MODF	CRC ERR	UDR	CHSIDE	TXE	RXNE
			r	r	r	r	r	r	r	r	rc_w0			r	r

位 15:13 保留，必须保持复位值。

位[12:11] **FTLVL[1:0]**：FIFP 发送级别 (FIFO Transmission Level)
这些位将由硬件置 1 和清零。
00：FIFO 为空
01：1/4 FIFO
10：1/2 FIFO
11：FIFO 为满（当 FIFO 阈值大于 1/2 时即视为满）
注： 这些位不适用于 I²S 模式。

位[10:9] **FRLVL[1:0]**：FIFO 接收级别 (FIFO reception level)
这些位将由硬件置 1 和清零。
00：FIFO 为空
01：1/4 FIFO
10：1/2 FIFO
11：FIFO 已满
注： 使能 CRC 计算时，这些位不适用于 I²S 模式和 SPI 仅接收模式。

位 8 **FRE**：帧格式错误 (Frame format error)
该标志在 TI 从模式和 I²S 从模式下用于 SPI。请参见*第 32.5.10 节：SPI 错误标志*和*第 32.7.8 节：I²S 错误标志*。
此标志由硬件置 1，在读取 SPIx_SR 时由软件复位。
0：未发生帧格式错误
1：发生帧格式错误

位 7 **BSY**：忙标志 (Busy flag)

0：SPI（或 I2S）不繁忙

1：SPI（或 I2S）忙于通信或者发送缓冲区不为空

此标志由硬件置 1 和清零。

注：　BSY 标志必须谨慎使用：请参见第 32.5.9 节：SPI 状态标志和第 976 页上的禁止 SPI 的步骤。

位 6 **OVR**：上溢标志 (Overrun flag)

0：未发生上溢

1：发生上溢

此标志由硬件置 1，由软件序列复位。有关软件序列，请参见*第 1005 页上的 I^2S 错误标志*。

位 5 **MODF**：模式故障 (Mode fault)

0：未发生模式故障

1：发生模式故障

此标志由硬件置 1，由软件序列复位。有关软件序列，请参见*第 985 页上的模式故障 (MODF)*一节。

注：　该位不适用于 I^2S 模式。

位 4 **CRCERR**：CRC 错误标志 (CRC error flag)

0：接收到的 CRC 值与 SPIx_RXCRCR 值匹配

1：接收到的 CRC 值与 SPIx_RXCRCR 值不匹配

此标志由硬件置 1，通过软件写入 0 来清零。

注：　该位不适用于 I^2S 模式。

位 3 **UDR**：下溢标志 (Underrun flag)

0：未发生下溢

1：发生下溢

此标志由硬件置 1，由软件序列复位。有关软件序列，请参见*第 1005 页上的 I^2S 错误标志*。

注：　该位不适用于 SPI 模式。

位 2 **CHSIDE**：通道 (Channel side)

0：发送或接收左通道信息

1：发送或接收右通道信息

注：　该位不适用于 SPI 模式。该位在 PCM 模式下无意义。

位 1 **TXE**：发送缓冲区为空 (Transmit buffer empty)

0：发送缓冲区非空

1：发送缓冲区为空

位 0 **RXNE**：接收缓冲区非空 (Receive buffer not empty)

0：发送缓冲区为空

1：接收缓冲区非空

32.9.4 SPI 数据寄存器 (SPIx_DR)

SPI data 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
DR[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15:0 **DR[15:0]**：数据寄存器 (Data register)
已接收或者要发送的数据。fuzzyend)
数据寄存器用于连接 Rx 和 Tx FIFO。读取数据寄存器时，将访问 RxFIFO；而写入数据寄存器时，将访问 TxFIFO（请参见*第 32.5.8 节：数据发送和接收过程*）。
注： 数据始终右对齐。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。Rx 阈值设置必须始终与当前使用的读访问相符。

32.9.5 SPI CRC 多项式寄存器 (SPIx_CRCPR)

SPI CRC polynomial register

偏移地址：0x10

复位值：0x0007

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

位 15:0 **CRCPOLY[15:0]**：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。
CRC 多项式 (0007h) 是此寄存器的复位值。可根据需要配置另一个多项式。

注： 多项式值只应为奇数。不支持任何偶数值。

32.9.6 SPI 接收 CRC 寄存器 (SPIx_RXCRCR)

SPI Rx CRC 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
RxCRC															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 15:0 **RXCRC[15:0]**：接收 CRC 寄存器 (Rx CRC register)

使能 CRC 计算后，RxCRC[15:0] 位将包含后续接收字节在计算后所得到的 CRC 值。当 SPIx_CR1 寄存器中的 CRCEN 位写入 1 时，此寄存器复位。CRC 通过 SPIx_CRCPR 寄存器中编程的多项式连续计算。

数据帧格式设置为 8 位数据（SPIx_CR1 的 CRCL 位清零）时，仅考虑 8 个 LSB 位。CRC 计算依据任意 CRC8 标准进行。

选择 16 位数据帧格式（SPIx_CR1 寄存器的 CRCL 位置 1）时，考虑此寄存器的全部 16 个位。CRC 计算依据任意 CRC16 标准进行。

注：　当 BSY 标志置 1 时，读取此寄存器可能返回一个不正确的值。

这些位不适用于 I^2S 模式。

32.9.7 SPI 发送 CRC 寄存器 (SPIx_TXCRCR)

SPI Tx CRC register

偏移地址：0x18

复位值：0x0000

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

位 15:0 **TxCRC[15:0]**：发送 CRC 寄存器 (Tx CRC register)

使能 CRC 计算后，TxCRC[7:0] 位将包含后续发送字节在计算后所得到的 CRC 值。当 SPIx_CR1 寄存器中的 CRCEN 位写入 1 时，此寄存器复位。CRC 通过 SPIx_CRCPR 寄存器中编程的多项式连续计算。

数据帧格式设置为 8 位数据（SPIx_CR1 的 CRCL 位清零）时，仅考虑 8 个 LSB 位。CRC 计算依据任意 CRC8 标准进行。

选择 16 位数据帧格式（SPIx_CR1 寄存器的 CRCL 位置 1）时，考虑此寄存器的全部 16 个位。CRC 计算依据任意 CRC16 标准进行。

注：　当 BSY 标志置 1 时，读取此寄存器可能返回一个不正确的值。

这些位不适用于 I^2S 模式。

32.9.8 SPIx_I²S 配置寄存器 (SPIx_I2SCFGR)

SPIx_I²S configuration register

偏移地址：0x1C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	ASTRTEN	I2SMOD	I2SE	I2SCFG		PCMSYNC	Res.	I2SSTD		CKPOL	DATLEN		CHLEN
			rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw

位 15:13 保留：由硬件强制为 0

位 12： **ASTRTEN**：异步启动使能 (Asynchronous start enable)

0：禁止异步启动。

在从模式下使能 I2S 后，若接收到 I2S 时钟并且在 WS 信号上检测到适当的边沿，则硬件将启动传输。

1：使能异步启动。

在从模式下使能 I2S 后，若接收到 I2S 时钟并且在 WS 信号上检测到适当的边沿，则硬件将启动传输。

*注： 使用 I2S Philips 标准时，对应的**边沿**为 WS 信号的下降沿；使用其它标准时，则为上升沿。*

*使用 I2S Philips 标准时，对应的**电平**为 WS 信号的低电平；使用其它标准时，则为高电平。*

更多相关信息，请参见第 32.7.4 节：时钟发生器。

位 11 **I2SMOD**：I2S 模式选择 (I2S mode selection)

0：选择 SPI 模式

1：选择 I2S 模式

注： 应在 SPI 禁止时配置此位。

位 10 **I2SE**：I2S 使能 (I2S enable)

0：禁止 I²S 外设

1：使能 I²S 外设

注： 该位不适用于 SPI 模式。

位 9:8 **I2SCFG**：I2S 配置模式 (I2S configuration mode)

00：从模式 - 发送

01：从模式 - 接收

10：主模式 - 发送

11：主模式 - 接收

注： 应在 I²S 禁止时配置这些位。

这些位不适用于 SPI 模式。

位 7 **PCMSYNC**：PCM 帧同步 (PCM frame synchronization)

0：短帧同步

1：长帧同步

注： 只有在 I2SSTD = 11（使用 PCM 标准）时，此位才有意义。

该位不适用于 SPI 模式。

位 6 保留：由硬件强制为 0

位 5:4 **I2SSTD**：I²S 标准选择 (I²S standard selection)

00：I²S 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 I²S 标准的详细信息，请参见*第 991 页上的第 32.7.2 节*

注： 为确保正确运行，应在 I²S 禁止时配置这些位。

这些位不适用于 SPI 模式。

位 3 **CKPOL**：无效状态时钟极性 (Inactive state clock polarity)

0：I²S 时钟无效状态为低电平

1：I²S 时钟无效状态为高电平

注： 为确保正确运行，应在 I²S 禁止时配置此位。

该位不适用于 SPI 模式。

位 CKPOL 不影响用于接收或发送 SD 和 WS 信号得到 CK 边沿灵敏度。

位 2:1 **DATLEN**：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： 为确保正确运行，应在 I²S 禁止时配置这些位。

这些位不适用于 SPI 模式。

位 0 **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

只有在 DATLEN = 00 时，此位的写操作才有意义，否则无论写入何值，通道长度始终由硬件固定为 32 位。

注： 为确保正确运行，应在 I²S 禁止时配置此位。

该位不适用于 SPI 模式。

32.9.9 SPIx_I²S 预分频器寄存器 (SPIx_I2SPR)

SPIx_I²S prescaler register

偏移地址：0x20

复位值：0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV							
						rw	rw	rw							

位[15:10] 保留：由硬件强制为 0

位 9 **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

注： 应在 I²S 禁止时配置此位。只有在 I²S 处于主模式时，才会使用此位。该位不适用于 SPI 模式。

位 8 **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

请参见*第 999 页上的第 32.7.4 节*。

注： 应在 I²S 禁止时配置此位。只有在 I²S 为主模式时，才会使用此位。该位不适用于 SPI 模式。

位[7:0] **I2SDIV**：I²S 线性预分频器 (I²S linear prescaler)

I2SDIV [7:0] = 0 或 I2SDIV [7:0] = 1 为禁用值。

请参见*第 999 页上的第 32.7.4 节*。

注： 应在 I²S 禁止时配置这些位。只有在 I²S 为主模式时，才会使用这些位。这些位不适用于 SPI 模式。

32.9.10 SPI/I2S 寄存器映射

表 179 给出了 SPI/I2S 寄存器映射和复位值。

表 179. SPI 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	**SPIx_CR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BIDIMODE	BIDIOE	CRCEN	CRCNEXT	CRCL	RXONLY	SSM	SSI	LSBFIRST	SPE	BR [2:0]			MSTR	CPOL	CPHA
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	**SPIx_CR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LDMA_TX	LDMA_RX	FRXTH	DS[3:0]				TXEIE	RXNEIE	ERRIE	FRF	NSSP	SSOE	TXDMAEN	RXDMAEN
	Reset value																		0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0x08	**SPIx_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FTLVL[1:0]		FRLVL[1:0]		FRE	BSY	OVR	MODF	CRCERR	UDR	CHSIDE	TXE	RXNE
	Reset value																				0	0	0	0	0	0	0	0	0	0	0	1	0
0x0C	**SPIx_DR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	**SPIx_CRCPR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRCPOLY[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
0x14	**SPIx_RXCRCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RxCRC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	**SPIx_TXCRCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TxCRC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	**SPIx_I2SCFGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ASTRTEN	I2SMOD	I2SE	I2SCFG		PCMSYNC	Res.	I2SSTD		CKPOL	DATLEN		CHLEN
	Reset value																					0	0	0	0	0		0	0	0	0	0	0
0x20	**SPIx_I2SPR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV							
	Reset value																							0	0	0	0	0	0	0	0	1	0

有关寄存器边界地址的信息，请参见*第 64 页上的第 2.2.2 节*。

33 串行音频接口 (SAI)

33.1 简介

SAI 接口（串行音频接口）灵活性高、配置多样，可支持多种音频协议。该接口适用于许多立体声或单声道应用。例如，它可配置为支持 I2S 标准、LSB 或 MSB 对齐、PCM/DSP、TDM 和 AC'97 等协议。将音频模块配置为发送器时，SAI 接口可提供 SPDIF 输出。

SAI 通过两个完全独立的音频子模块来实现这种灵活性和可配置性。每个模块都有自己的时钟发生器和 I/O 线控制器。

SAI 可以配置为主模式或配置为从模式。音频子模块既可作为接收器，又可作为发送器；既可与另一模块同步，又可以不同步。

SAI 可与其它 SAI 相连接来同步运行。

33.2 SAI 主要特性

- 具有两个独立的音频子模块，子模块既可作为接收器，也可作为发送器，并带有自身的 FIFO。
- 每个音频子模块集成多达 8 个字，每个字32位的 FIFO。
- 两个音频子模块间可以是同步或异步模式。
- 多个 SAI 之间可实现同步。
- 两个音频子模块的主/从配置相互独立。
- 当两个音频子模块都配置为以主模式工作时，每个子模块的时钟发生器采用独立的音频采样频率。
- 数据大小可配置：8 位、10 位、16 位、20 位、24 位或 32 位。
- 音频协议：I2S、LSB 或 MSB 对齐、PCM/DSP、TDM、AC'97。
- 如果需要，可提供 SPDIF 输出。
- 高达 16 个大小可配置的 Slot。
- 每帧的位数可配置。
- 帧同步有效电平可配置（偏移、位长、电平）。
- 可配置 Slot 中第一个有效位的位置。
- 支持 LSB 或 MSB 数据传输。
- 支持静音模式。
- 具有立体声/单声道音频帧功能。
- 通信时钟选通边沿可配置 (SCK)。
- 错误标志对应相应中断（分别使能时）。
  - 上溢和下溢检测，
  - 从模式下的帧同步信号提前检测，
  - 从模式下的帧同步信号滞后检测，
  - 接收时编码解码器未针对 AC'97 模式就绪。

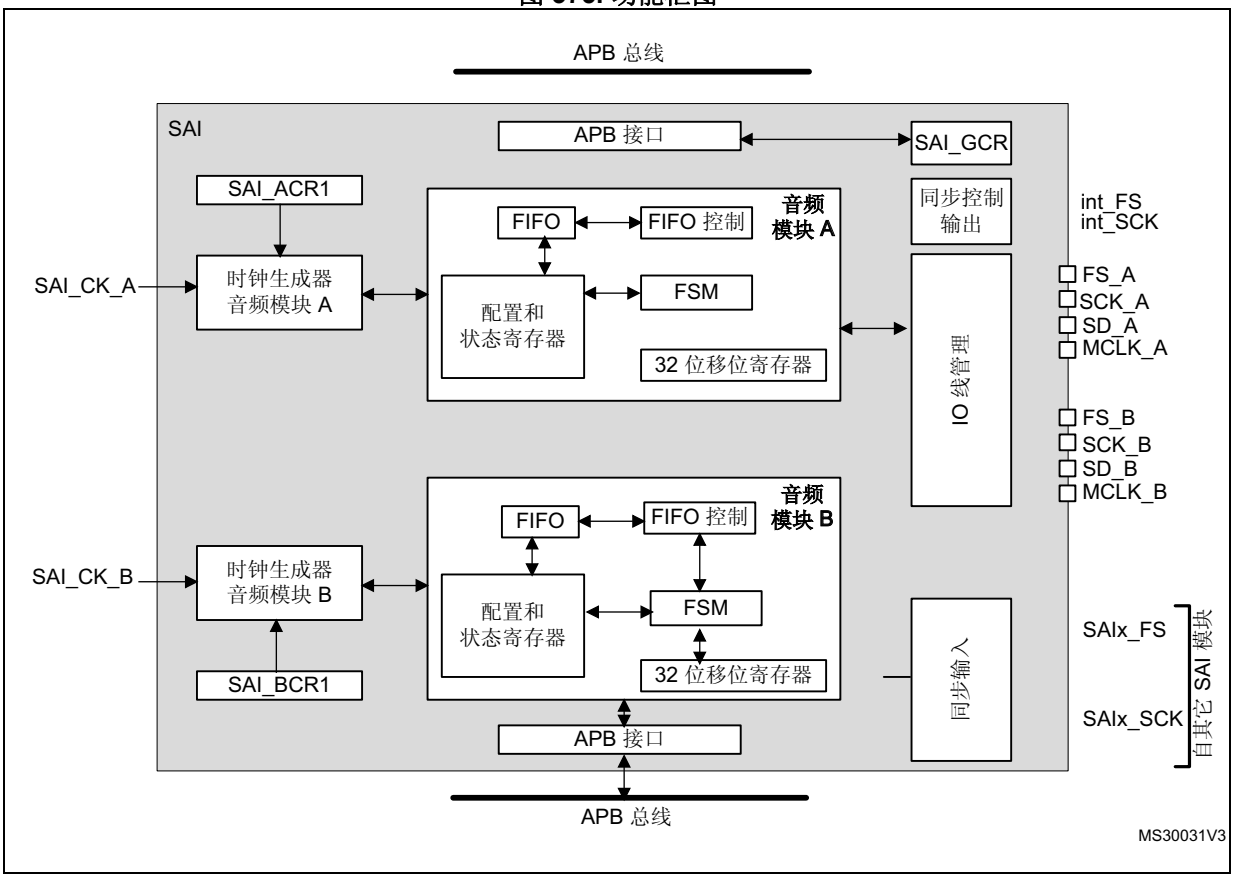
- 支持如下中断源（使能时）：
  - 错误，
  - FIFO 请求。
- DMA 接口有 2 个通道。

33.3 SAI 功能说明

33.3.1 SAI 框图

SAI 框图如*图 375* 所示。

图 375. 功能框图



SAI 主要由两个各自带有时钟发生器的音频子模块组成。每个音频模块集成一个 32 位移位寄存器，该寄存器由模块自身的功能状态机控制。数据存储和读取都是通过专用的 FIFO 来完成。FIFO 可通过 CPU 访问，也可通过 DMA 访问以减轻 CPU 的通信负担。每个音频模块是独立的。这两个音频子模块可彼此同步。

I/O 线控制器管理 SAI 中指定音频模块的 4 个专用引脚（SD、SCK、FS、MCLK）。如果将两个子模块声明为同步模块，则可以共用其中某些引脚，从而留出一些引脚用作通用 I/O。MCLK 引脚是否可用作输出引脚取决于实际应用和解码要求以及音频模块是否配置为主模式。

如果将一个 SAI 配置为与其它 SAI 同步运行，可以释放更多 I/O（引脚 SD_x 除外）。

可配置功能状态机来处理多种音频协议。一些寄存器用于设置所需协议（音频帧波形发生器）。

音频子模块在主模式或从模式下均可用作发送器或接收器。主模式意味着从 SAI 生成 SCK_x 位时钟和帧同步信号，而从模式则意味着 SCK_x 位时钟和帧同步信号来自另一外部或内部主器件。在特殊情况下，FS 信号方向与主模式或从模式定义不直接相关。在 AC'97 协议中，即使 SAI（链接控制器）设置为消耗 SCK 时钟，FS 信号也会是 SAI 输出（从模式下也是如此）。

注： *为方便阅读此部分，符号 SAI_x 指 SAI_A 或 SAI_B，其中“x”代表 SAI A 子模块或 SAI B 子模块。*

33.3.2 SAI 的主要模式

SAI 的每个音频子模块均可通过所选音频模块的 SAI_xCR1 寄存器中的 MODE 位配置为主模式或从模式。

主模式

在主模式下，SAI 会向连接的外部器件提供时钟信号：

- 位时钟和帧同步分别在引脚 SCK_x 和 FS_x 上输出。
- 如果需要，SAI 也可以在 MCLK_x 引脚上生成主时钟。

SCK_x、FS_x 和 MCLK_x 均配置为输出。

从模式

SAI 从外部器件接收时钟信号。

- 如果将 SAI 子模块配置为异步模式，则 SCK_x 引脚和 FS_x 引脚将被配置为输入。
- 如果将 SAI 子模块配置为与另一个 SAI 接口或第二个音频子模块同步运行，则会释放相应的 SCK_x 引脚和 FS_x 引脚以用作通用 I/O。

在从模式下，不使用 MCLK_x 引脚，可将其分配给其它功能。

建议在使能主器件前先使能从器件。

配置和使能 SAI 模式

可通过相应音频模块的 SAI_xCR1 寄存器中的 MODE 位将每个音频子模块独立定义为发送器或接收器。为此，SAIx_SD 引脚将分别被配置为输出或输入。

可使用两个不同的 MCLK 和 SCK 时钟频率配置同一 SAI 中的两个主音频模块。在这种情况下，必须将这两个音频模块配置为异步模式。

SAI 中的每个音频模块均通过 SAI_xCR1 寄存器中的 SAIXEN 位使能。在从模式下，此位一经激活，发送器或接收器便会对时钟线、数据线和同步线上的活动敏感。

在主 TX 模式下，即使 FIFO 中没有数据，使能音频模块也会立即为外部从模块产生位时钟，但 FS 信号的产生受 FIFO 中是否存在数据的控制。FIFO 接收到要发送的第一个数据后，此数据将输出到外部从模块。如果 FIFO 中没有要发送的数据，则随后将在音频帧中传送值 0，并会产生一个下溢标志。

在从模式下，使能音频模块时和检测到 SOF 位时开始音频帧。

在从 TX 模式下，使能音频模块后的第一个帧上不可能出现下溢事件，因为此时的强制操作顺序如下：

1. 通过软件或 DMA 写入 SAI_xDR。
2. 等待至 FIFO 阈值 (FLH) 标志与 000b（FIFO 为空）不同。
3. 使能音频模块为从发送模式。

33.3.3 SAI 同步模式

同步模式存在两个同步级别，即音频子模块级别和 SAI 级别。

内部同步

将音频子模块配置为与同一 SAI 中的第二个音频子模块同步运行。在这种情况下，将共用位时钟和帧同步信号，以减少通信时占用外部引脚的数量。配置为同步模式的音频模块将释放其 SCK_x、FS_x 和 MCLK_x 引脚以用作 GPIO，而配置为异步模式的音频模块将使用其 FS_x、SCK_x 和 MCLK_x I/O 引脚（如果该音频模块被配置为主模块）。

通常，同步模式下的音频模块可用于在全双工模式下配置 SAI。两个音频模块中的一个可配置为主模块，另一个为从模块；也可将两个均配置为从模块；一个模块为异步模块（SAI_xCR1 中的相应位 SYNCEN[1:0] 设置为 00），另一个为同步模块（SAI_xCR1 中的相应位 SYNCEN[1:0] 设置为 01）。

注： *由于存在内部重新同步阶段，因此 PCLK APB 频率必须大于比特率时钟频率的二倍。*

外部同步

音频子模块也可以配置为与其它 SAI 同步运行，具体方法如下：

1. 配置为其它 SAI 的同步源的 SAI 必须定义其音频子模块中的哪个子模块应向其它 SAI 提供 FS 和 SCK 信号。此操作通过编程 SYNCOUT[1:0] 位完成。
2. 接收同步信号的 SAI 必须通过为 SYNCIN[1:0] 位设置合适的值来选择哪个 SAI 将提供同步。之后，对于这两个 SAI 音频子模块的每个子模块，用户都必须通过 SYNCEN 位指定其是否与其它 SAI 同步运行。

注： *SYNCIN[1:0] 位和 SYNCOUT[1:0] 位位于 SAI_GCR 寄存器中，SYNCEN 位位于 SAI_xCR1 寄存器中。*

如果指定 SAI 中的两个音频子模块都需要与另一个 SAI 同步，则可以选择以下配置之一：

- 通过 SYNCEN[1:0] 位将每个音频子模块均配置为与另一个 SAI 模块同步。
- 通过 SYNCEN[1:0] 位将一个音频子模块配置为与另一个 SAI 模块同步。随后，通过 SYNCEN[1:0] 位将其它音频子模块配置为与第二个 SAI 音频子模块同步。

下表显示了如何根据所使用的 SAI 模块选择合适的同步信号。例如，SAI2 可通过将 SAI2 SYNCIN 设置为 0 来选择通过 SAI1 进行同步。如果 SAI1 要选择通过 SAI2 进行同步，则必须将 SAI1 SYNCIN 置 1。标注为“保留”的位不可使用。

表 180. 外部同步选择

模块实例	SYNCIN= 3	SYNCIN= 2	SYNCIN= 1	SYNCIN= 0
SAI1	保留	保留	SAI2 同步	保留
SAI2	保留	保留	保留	SAI1 同步

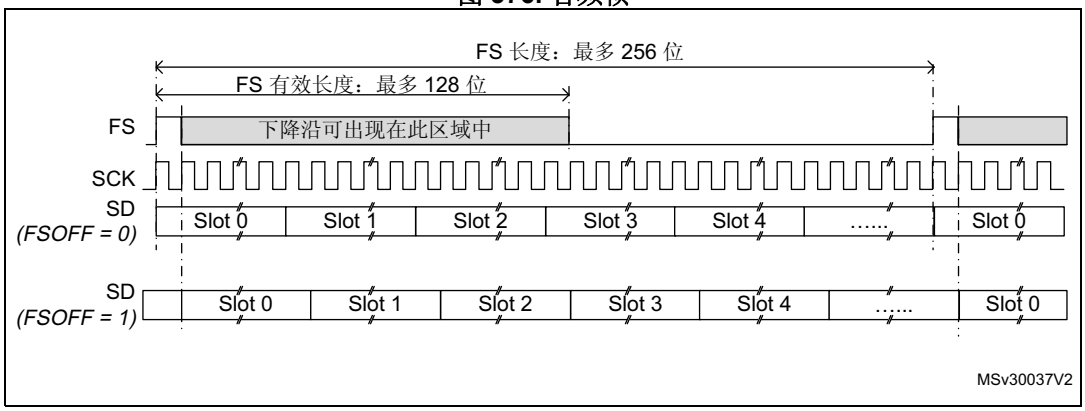
33.3.4 音频数据大小

通过配置 SAI_xCR1 寄存器中的 DS[2:0] 位，配置音频帧的数据大小。数据大小可以是 8 位、10 位、16 位、20 位、24 位或 32 位。在传输期间，将首先发送数据的 MSB 或 LSB，具体取决于 SAI_xCR1 寄存器中的 LSBFIRST 位的配置。

33.3.5 帧同步

FS 信号用作音频帧中的帧同步信号（SOF）。此信号的波形完全可配置，以在帧同步时，支持各种具有特殊规格的音频协议。这一可重新配置性通过寄存器 SAI_xFRCR 来实现。*图 376* 说明了这种灵活性。

图 376. 音频帧



在 AC'97 模式或 SPDIF 模式下（SAI_xCR1 寄存器中的位 PRTCFG[1:0] = 10 或 PRTCFG[1:0] = 01），帧同步信号的波形被强制配置为支持 AC'97 协议。SAI_xFRCR 寄存器值被忽略。

每个音频模块相互独立，因此均需要特定的配置。

帧长度

- 主模式

  将 SAI_xFRCR 寄存器中的 FRL[7:0] 字段置 1，可将音频帧的长度配置为最长 256 个位时钟周期。

  如果帧长度大于为该帧声明的 Slot 数，则要发送的剩余位将用 0 填充，或者 SD 线将释放为高阻态，具体取决于 SAI_xCR2 寄存器中的位 TRIS 的状态（见*FS 信号的作用*一节）。在接收模式下，剩余位被忽略。

  如果将位 NODIV 清零，(FRL+1) 必须等于 2 的几次幂（从 8 到 256）以确保音频帧的每个位时钟周期都包含整数个 MCLK 脉冲。

  如果将位 NODIV 置 1，则 (FRL+1) 字段可采用从 8 到 256 的任意值。请参见*第 33.3.7 节：SAI 时钟发生器*。

- 从模式

  音频帧的长度主要用于指定由外部主模块向从模块发送的每个音频帧的位时钟周期数。它主要用于从主模块中检测音频帧传输期间出现的提前或滞后帧同步信号。这种情况下会出现错误。有关详细信息，请参见*第 33.3.12 节：错误标志*。

  在从模式下，SAI_xFRCR 寄存器中的 FRL[7:0] 位的配置不受任何限制。

帧中的位数等于 FRL[7:0] + 1。

音频帧中要传输的最小位数为 8。

帧同步极性

SAI_xFRCR 寄存器中的 FSPOL 位用于设置 FS 引脚的有效极性，通过该极性来启动帧。SOF 信号对边沿敏感。

在从模式下，音频模块等待一个有效帧来启动发送或接收。SOF 信号与此信号同步。只有通信期间未检测到 SOF 信号并且 SOF 信号与预期 SOF 信号相同时，帧同步极性才有效（请参见*第 33.3.12 节：错误标志*）。

在主模式下，每次音频帧完成时均会发送帧同步信号，直至 SAI_xCR1 寄存器中的 SAIXEN 位清零。如果前一个音频帧结束时 FIFO 中不存在数据，则将按照*第 33.3.12 节：错误标志*所述管理下溢条件，但音频通信流不会中断。

帧同步有效电平长度

SAI_xFRCR 寄存器的 FSALL[6:0] 位用于配置帧同步信号的有效电平的长度。该长度可设置为 1 到 128 个位时钟周期。

例如，有效长度在 I2S、LSB 或 MSB 对齐模式下为帧长的一半，在 PCM/DSP 或 TDM 模式下为 1 位。

帧同步偏移

基于应用中支持的音频协议（例如 I2S 标准协议和 MSB 对齐协议），可以在发送音频帧的最后一位或第一位时将帧同步信号置为有效。通过 SAI_xFRCR 寄存器中的 FSOFF 位选择两个配置之一。

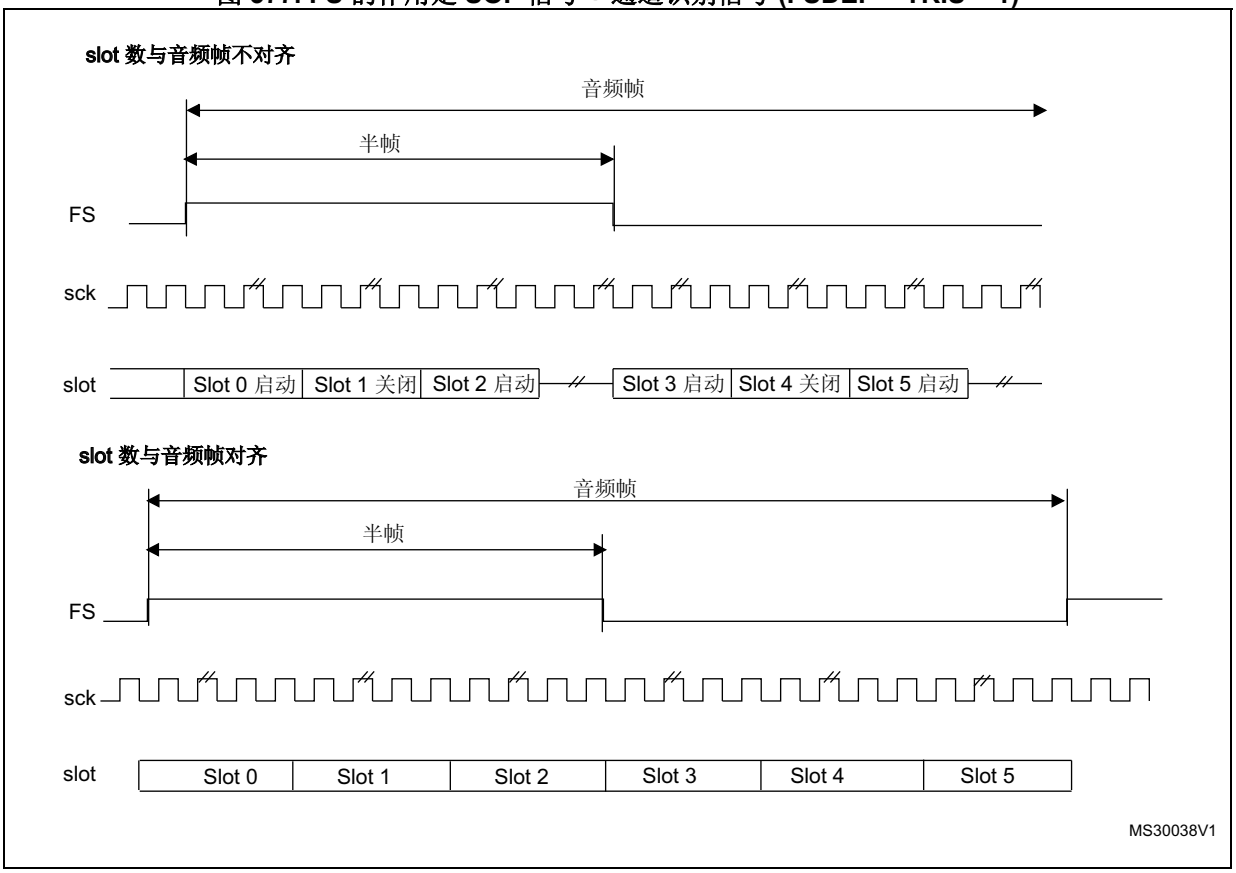
FS 信号的作用

FS 信号可以有不同的含义，具体取决于 FS 的功能。SAI_xFRCR 寄存器中的 FSDEF 位用于选择 FS 信号的含义。

- 0：SOF 信号，例如 PCM/DSP、TDM、AC’97 和音频协议，
- 1：音频帧内的 SOF 信号和通道识别信号，例如 I2S、MSB 或 LSB 对齐协议。

当 FS 信号被视为帧内的 SOF 信号和通道识别信号时，声明的 Slot 数必须是一半用于左通道，一半用于右通道。如果半个音频帧上的位时钟周期数大于某个通道的专用 Slot 数，若 TRIS = 0，则 SAI_xCR2 寄存器中的剩余位时钟周期将发送 0。否则若 TRIS = 1，SD 线将释放为高阻态。接收时，直到通道发生变化，才会考虑剩余位时钟。

图 377. FS 的作用是 SOF 信号 + 通道识别信号 (FSDEF = TRIS = 1)

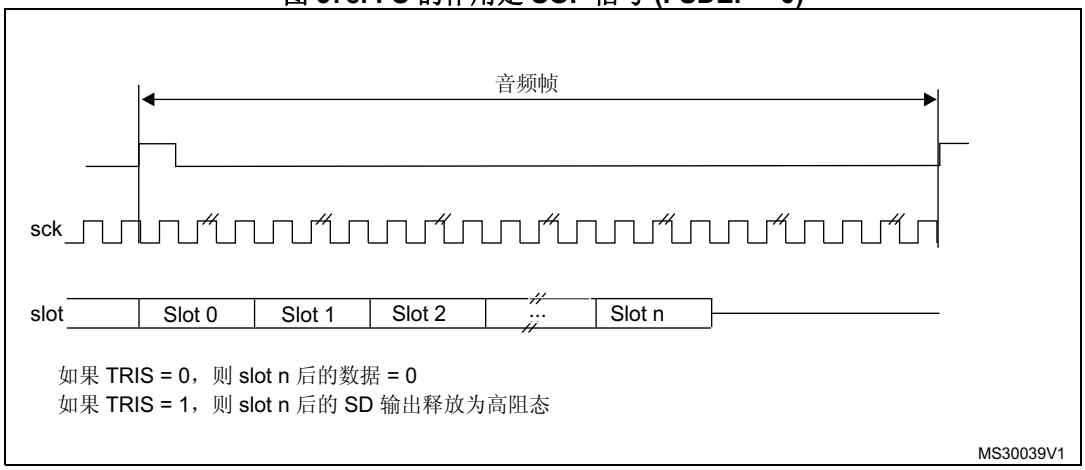


1. 帧长度应为偶数。

如果 SAI_xFRCR 中的 FSDEF 位保持清零状态，则 FS 信号等效于 SOF 信号，如果 SAI_xSLOTR 中的 NBSLOT[3:0] 位定义的 Slot 数乘以 SAI_xSLOTR 中的 SLOTSZ[1:0] 位配置的 Slot 位数所得的结果小于帧大小（SAI_xFRCR 寄存器中的 FRL[7:0] 位），则：

- 如果 SAI_xCR2 寄存器中的 TRIS = 0，则最后一个 Slot 后的剩余位将强制为 0，直至发送器的帧结束，
- 如果 TRIS = 1，则传输这些剩余位时，数据线将释放为高阻态。在接收模式下，这些位被丢弃。

图 378. FS 的作用是 SOF 信号 (FSDEF = 0)



在发送模式下将音频模块配置为获取 SD 线上的 SPDIF 输出时，将不使用 FS 信号。相应的 FS I/O 会被释放，留作他用。

33.3.6 Slot 配置

Slot 是音频帧中的基本元素。音频帧中的 Slot 数等于 NBSLOT[3:0] + 1。

每个音频帧的最大 Slot 数固定为 16。

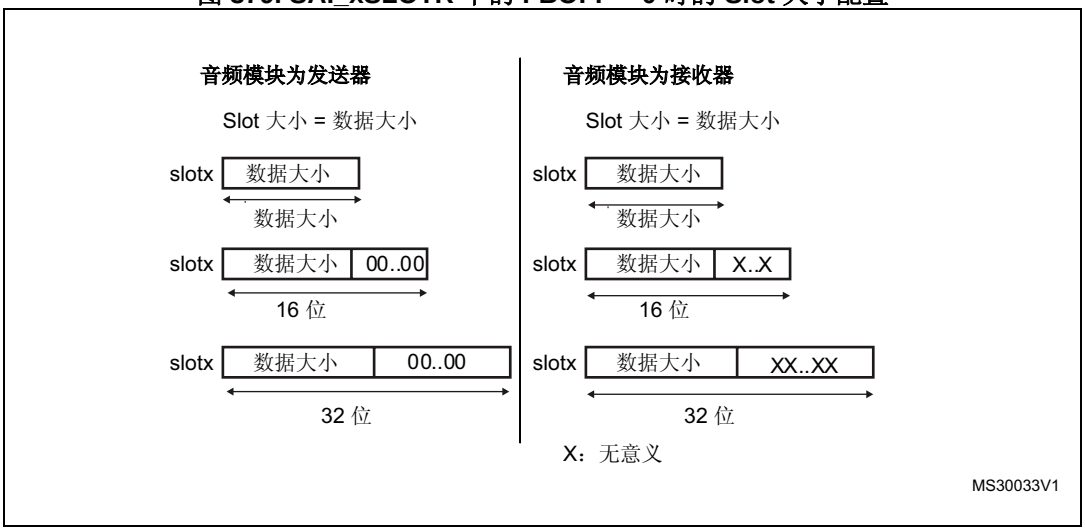
对于 AC'97 协议或 SPDIF 协议（位 PRTCFG[1:0] = 10 或 PRTCFG[1:0] = 01 时），Slot 数自动按照协议规范设置，NBSLOT[3:0] 的值被忽略。

通过设置 SAI_xSLOTR 寄存器的 SLOTEN[15:0] 位，可将各个 Slot 定义为有效 Slot 或无效 Slot。

传输一个无效 Slot 时，SD 数据线将强制为 0 或释放为高阻态，具体取决于 TRIS 位在发送模式下的配置（请参见*无效 Slot 上的输出数据线管理*一节）。在接收模式下，从该 Slot 结束后接收的数据被忽略。结果，不会有 FIFO 访问，也不会有与此无效 Slot 状态相关的 FIFO 读/写请求。

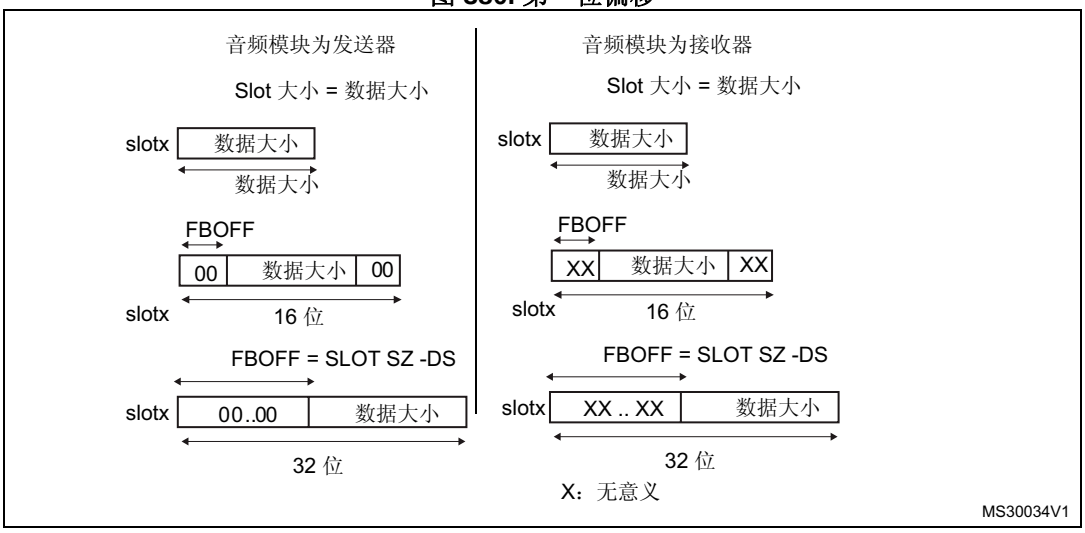
Slot 大小也可配置，如*图 379* 所示。通过将 SAI_xSLOTR 寄存器中的 SLOTSZ[1:0] 位置 1 来选择 Slot 大小。该大小适用于音频帧中的每个 Slot。

图 379. SAI_xSLOTR 中的 FBOFF = 0 时的 Slot 大小配置



可以选择 Slot 内要传输的第一个数据位的位置， 此偏移通过 SAI_xSLOTR 寄存器中的 FBOFF[4:0] 位配置。在发送模式下，将从 Slot 开始时注入 0 值，直至到达此偏移位置。接收时，偏移阶段中的位被忽略。此特性适用于 LSB 对齐协议（如果偏移等于 Slot 大小减去数据大小）。

图 380. 第一位偏移



要避免出现故障 SAI 行为，必须遵循以下条件：

FBOFF ≤ (SLOTSZ - DS)，

DS ≤ SLOTSZ，

NBSLOT x SLOTSZ ≤ FRL（帧长度），

SAI_xFRCR 寄存器中的 FSDEF 位置 1 时，Slot 数必须为偶数。

在 AC’97 和 SPDIF 协议（位 PRTCFG[1:0] = 10 或 PRTCFG[1:0] = 01）中，Slot 大小按照 *第 33.3.9 节：AC’97 链路控制器*中的定义自动设置。

有关AC’97 模式下时钟发生器编程的详细信息，请参见 *第 33.3.9 节：AC’97 链路控制器*；有关 SPDIF 模式下时钟发生器编程的详细信息，请参见 *第 33.3.10 节：SPDIF 输出*。

33.3.7 SAI 时钟发生器

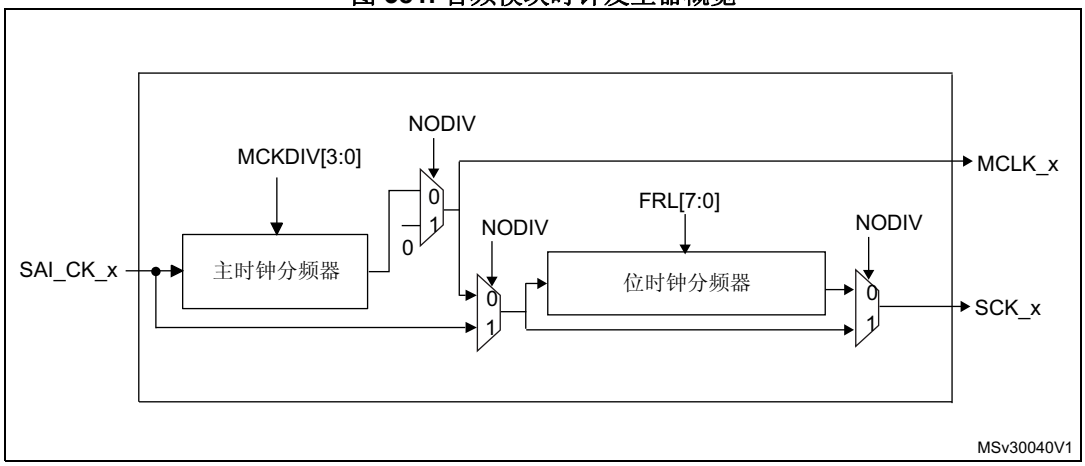
每个音频模块都有自己的时钟发生器，旨在使这两个模块完全独立。这两个时钟发生器的功能没有任何区别。

将音频模块配置为主模块时，时钟发生器将提供通信时钟（位时钟）以及用于外部解码器的主时钟。

当音频模块定义为从模块时，时钟发生器将关闭。

图 381 给出了音频模块时钟发生器的架构。

图 381. 音频模块时钟发生器概览



注： *当 NODIV 置 1 时，如果 MCLK_x 引脚配置为 GPIO 外设中的 SAI 引脚，则其信号电平将置 0。*

时钟发生器的时钟源来自乘积时钟控制器。SAI_CK_x 时钟等效于主时钟，可通过 MCKDIV[3:0] 位为外部解码器分频：

如果 MCKDIV[3:0] 不等于 0000，则 MCLK_x = SAI_CK_x / (MCKDIV[3:0] * 2)

如果 MCKDIV[3:0] 等于 0000，则 MCLK_x = SAI_CK_x

MCLK_x 信号仅在 TDM 中使用。

分频必须均匀，使 MCLK 输出上和 SCK_x 时钟上都保持 50% 的占空比。如果位 MCKDIV[3:0] = 0000，采用一分频可使 MCLK_x 等于 SAI_CK_x。

在 SAI 中，使用单一比率 MCLK/FS = 256。大多数情况下，将遇到三个频率范围，如*表 181* 所示。

表 181. 可能的音频采样范围示例

输入 SAI_CK_x 时钟频率	可获得的常见音频采样频率	MCKDIV[3:0]
192 kHz x 256	192 kHz	MCKDIV[3:0] = 0000
	96 kHz	MCKDIV[3:0] = 0001
	48 kHz	MCKDIV[3:0] = 0010
	16 kHz	MCKDIV[3:0] = 0110
	8 kHz	MCKDIV[3:0] = 1100

表 181. 可能的音频采样范围示例（续）

输入 SAI_CK_x 时钟频率	可获得的常见音频采样频率	MCKDIV[3:0]
44.1 kHz x 256	44.1 kHz	MCKDIV[3:0] = 0000
	22.05 kHz	MCKDIV[3:0] = 0001
	11.025 kHz	MCKDIV[3:0] = 0010
SAI_CK_x = MCLK(1)	MCLK	MCKDIV[3:0] = 0000

1. 当乘积时钟控制器选择一个外部时钟源而非 PLL 时钟时，会出现这种情况。

如果相应音频模块声明为主模块且 SAI_xCR1 寄存器中的位 NODIV = 0，可通过 I/O 口为外部解码器生成主时钟。在从模式下，由于时钟发生器关闭，将忽略这最后一位中设置的值，且 MCLK_x I/O 引脚用作通用 I/O 使用。

位时钟通过主时钟导出。位时钟分频器按照以下公式设置位时钟 (SCK_x) 和主时钟 (SCK_x) 间的分频系数：

SCK_x = MCLK x (FRL[7:0] +1) / 256

其中：

256 是 MCLK 和音频采样频率之间的固定比率。

FRL[7:0] 是音频帧中的位时钟周期 - 1，在 SAI_xFRCR 寄存器中配置。

主模式下，(FRL[7:0] +1) 必须等于 2 的几次幂（请参见*第 33.3.5 节：帧同步*），以通过位时钟周期获得偶数个 MCLK_x 脉冲。位时钟 (SCK_x) 上将保证 50% 的占空比。

SAI_CK_x 时钟还可等于位时钟频率。在这种情况下，SAI_xCR1 寄存器中的 NODIV 位应置 1，MCKDIV 分频器内的值以及位时钟分频器内的值将被忽略。此时，每个帧的位数完全可配置，而无需等于 2 的几次幂。

SCK 上的位时钟选通边沿可通过 SAI_xCR1 寄存器中的 CKSTR 位配置。

有关 SPDIF 模式下的时钟发生器编程的详细信息，请参见*第 33.3.10 节：SPDIF 输出*。

33.3.8 内部 FIFO

SAI 中的每个音频模块都有自己的 FIFO。根据模块是定义为发送器还是接收器，可相应的读取或写入其 FIFO。因此只存在一个 FIFO 请求与 SAI_xSR 寄存器中的 FREQ 位相关。

如果 SAI_xIM 寄存器中的 FREQIE 位使能，将产生中断。这取决于：

- FIFO 阈值设置（SAI_xCR2 中的 FLTH 位）
- 通信方向（发送器或接收器）。请参见*在发送模式下产生中断*一节和*在接收模式下产生中断*一节。

在发送模式下产生中断

根据发送模式下的 FIFO 配置产生中断：

- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 为空时（FTH[2:0] 置为 000b），如果寄存器中没有数据（SAI_xSR 中的 FLTH[2:0] 位小于 001b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当 FIFO 不再为空时（SAI_xSR 中的 FLTH[2:0] 位不是 000b），即 FIFO 中存储一个或多个数据时，此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。

- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 四分之一满时（FTH[2:0] 置为 001b），如果不到四分之一的 FIFO 包含数据（SAI_xSR 中的 FLTH[2:0] 位小于 010b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当至少四分之一的 FIFO 包含数据时（SAI_xSR 中的 FLTH[2:0] 位大于等于 010b），此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。
- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 半满时（FTH[2:0] 置为 010b），如果不到一半的 FIFO 包含数据（SAI_xSR 中的 FLTH[2:0] 位小于 011b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当至少一半的 FIFO 包含数据时（SAI_xSR 中的 FLTH[2:0] 位大于或等于 011b），此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。
- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 四分之三满时（FTH[2:0] 置为 011b），如果不到四分之三的 FIFO 包含数据（SAI_xSR 中的 FLTH[2:0] 位小于 100b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当至少四分之三的 FIFO 包含数据时（SAI_xSR 中的 FLTH[2:0] 位大于或等于 100b），此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。
- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 为满时（FTH[2:0] 置为 100b），如果 FIFO 不满（SAI_xSR 中的 FLTH[2:0] 位小于 101b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当 FIFO 已满时（SAI_xSR 中的 FLTH[2:0] 位等于值 101b），此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。

在接收模式下产生中断

根据接收模式下的 FIFO 配置产生中断：

- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 为空时（FTH[2:0] 置为 000b），如果 SAI_xDR 寄存器中至少有一个数据（SAI_xSR 中的 FLTH[2:0] 位大于或等于 001b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当 FIFO 变为空时（SAI_xSR 中的 FLTH[2:0] 位等于 000b），即 FIFO 中未存储数据时，此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。
- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 四分之一满时（FTH[2:0] 置为 001b），如果至少有四分之一的 FIFO 数据单元可用（SAI_xSR 中的 FLTH[2:0] 位大于或等于 010b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当不到四分之一的 FIFO 数据单元可用时（SAI_xSR 中的 FLTH[2:0] 位小于 010b），此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。
- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 半满时（FTH[2:0] 置为 010b），如果至少有一半的 FIFO 数据单元可用（SAI_xSR 中的 FLTH[2:0] 位大于或等于 011b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当不到一半的 FIFO 数据单元可用时（SAI_xSR 中的 FLTH[2:0] 位小于 011b），此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。
- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 四分之三满时（FTH[2:0] 置为 011b），如果至少有四分之三的 FIFO 数据单元可用（SAI_xSR 中的 FLTH[2:0] 位大于或等于 100b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当不到四分之三的 FIFO 数据单元可用时（SAI_xSR 中的 FLTH[2:0] 位小于 100b），此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。
- 当 SAI_xCR2 寄存器中的 FIFO 阈值位配置为 FIFO 满时（FTH[2:0] 置为 100b），如果 FIFO 已满（SAI_xSR 中的 FLTH[2:0] 位等于 101b），将产生中断（SAI_xSR 寄存器中的 FREQ 位由硬件置 1）。当 FIFO 不满时（SAI_xSR 中的 FLTH[2:0] 位小于 101b），此中断（SAI_xSR 寄存器中的 FREQ 位）由硬件清零。

与中断的产生类似，如果 SAI_xCR1 寄存器中的 DMAEN 位置 1，则 SAI 可使用 DMA。FREQ 位的有效机制与上述 FREQIE 的中断产生机制相同。

每个 FIFO 均是一个 8 字 FIFO。无论访问大小为何，每次对 FIFO 进行读写时，均针对 1 个字的 FIFO 单元进行操作。每个 FIFO 字包含一个音频 Slot。每次访问 SAI_xDR 寄存器后，FIFO 指针递增一个字。

数据应以右对齐方式写入 SAI_xDR。

接收到的数据将以右对齐方式存储到 SAI_xDR。

将 SAI_xCR2 寄存器中的 FFLUSH 位置 1 禁止 SAI 后，可重新初始化 FIFO 指针。如果 FFLUSH 在 SAI 使能情况下置 1，则 FIFO 中的数据将自动清除。

33.3.9 AC’97 链路控制器

SAI 可用作 AC’97 链路控制器。在此协议中：

- Slot 数和 Slot 大小固定。
- 帧同步信号定义完善并且波形固定。

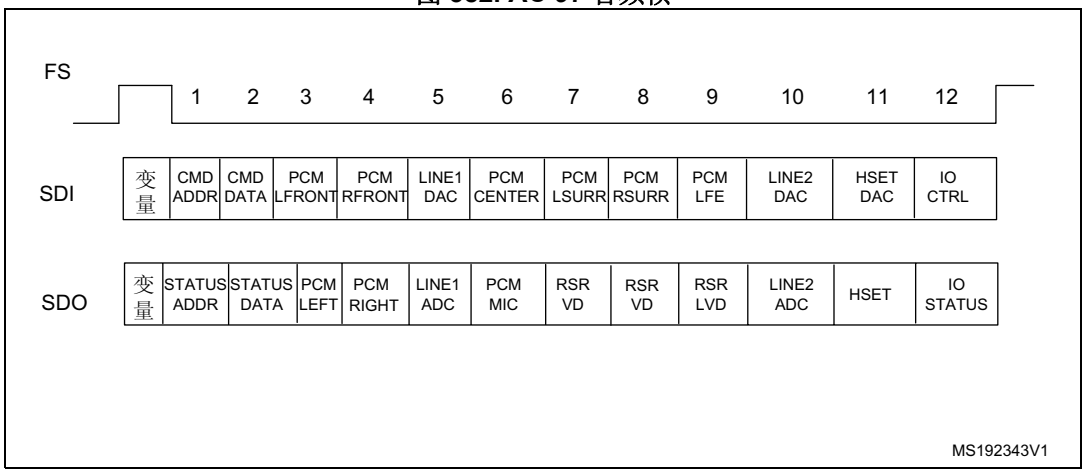
要选择此协议，可将 SAI_xCR1 寄存器中的 PRTCFG[1:0] 位置为 10。选择 AC’97 模式时，只能使用 16 位或 20 位的数据大小，否则将无法保证 SAI 运行正常。

- 因此，NBSLOT[3:0] 和 SLOTSZ[1:0] 位将被忽略。
- Slot 数固定为 13。第一个 Slot 为 16 位宽，所有其它 Slot 均为 20 位宽（数据 Slot）。
- SAI_xSLOTR 寄存器中的 FBOFF[4:0] 位被忽略。
- SAI_xFRCR 寄存器被忽略。
- 未使用 MCLK。

无论采用主配置还是从配置，由于 AC’97 链路控制器驱动 FS 信号，异步模块发出的 FS 信号将自动配置为输出。

图 382 给出了 AC’97 音频帧的结构。

图 382. AC’97 音频帧



注： *在 AC’97 协议中，TAG 的位 2 将保留（始终为 0），因此无论 SAI FIFO 中写入何值，TAG 的位 2 均强制为 0。*

有关 TAG 表示的详细信息，请参见 AC'97 协议标准。

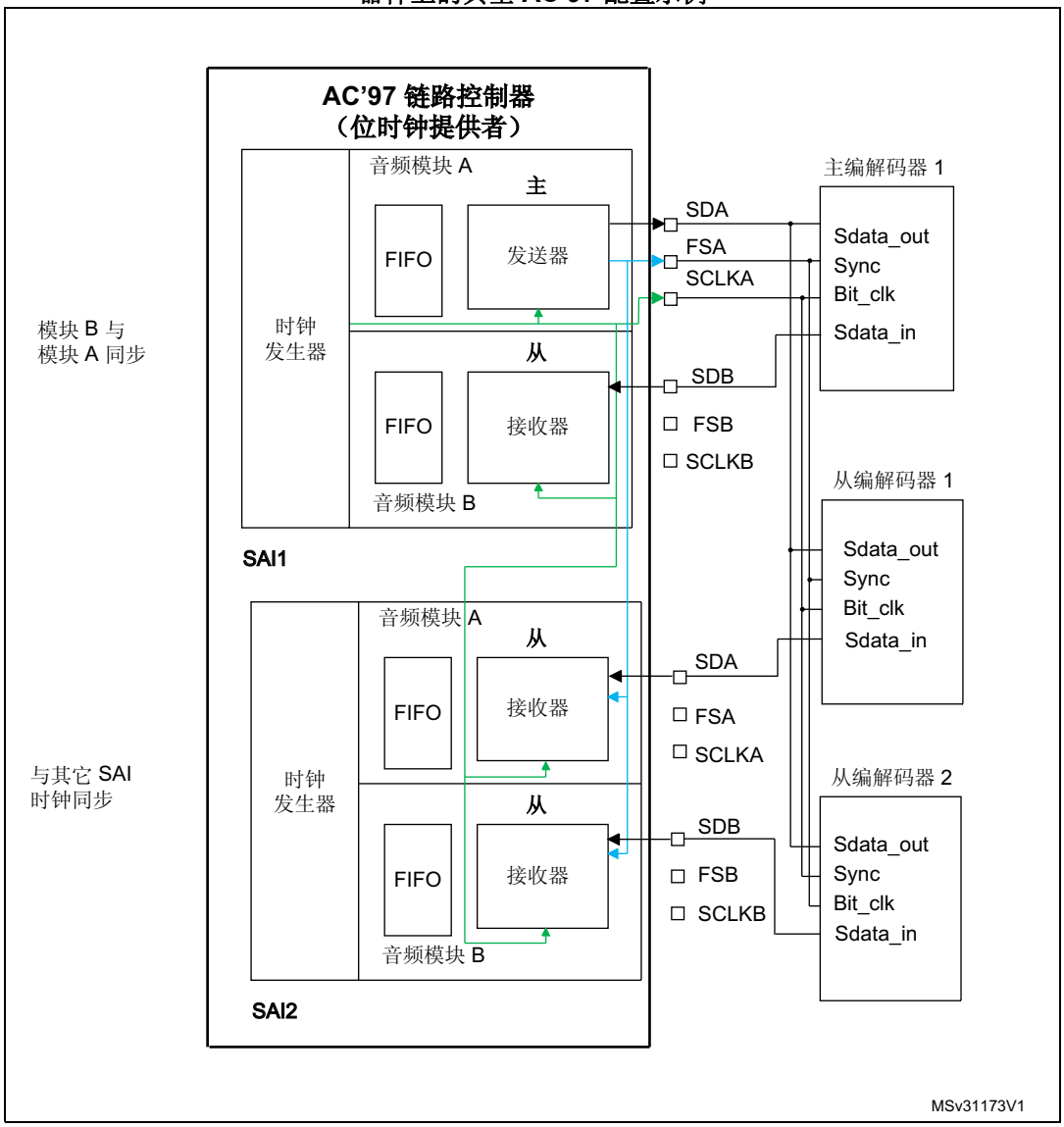
可将一个 SAI 用于 AC'97 点对点通信。

使用两个 SAI（对于具有两个嵌入式 SAI 的器件）可以控制三个外部 AC'97 解码器，如*图 383* 所示。

在 SAI1 中，必须将音频模块 A 声明为异步主发送器，而将音频模块 B 定义为从接收器，并在内部与音频模块 A 同步。

在从接收器模式下，同时针对音频模块 A 和音频模块 B 将 SAI2 配置为与外部 SAI1 同步。

图 383. 具有至少 2 个嵌入式 SAI（三个外部 AC'97 解码器）的器件上的典型 AC'97 配置示例



在接收器模式下，用作 AC'97 链路控制器的 SAI 无需 FIFO 请求，因此当 Slot 0 中的编码解码器就绪位解码为低电平时，FIFO 中不存储任何数据。如果 SAI_xIM 寄存器中的 CNRDYIE 位使能，则 SAI_xSR 寄存器中的标志 CNRDY 将置 1 并会产生一个中断。此标志专用于 AC'97 协议。

AC’97 模式下的时钟发生器编程

在 AC’97 模式下，帧长度固定为 256 位，其频率应设置为 48 kHz。*第 33.3.7 节：SAI 时钟发生器*中给出的规则应在 FRL = 255 时使用，以生成适当的帧速率 (F_{FS_x})。

33.3.10 SPDIF 输出

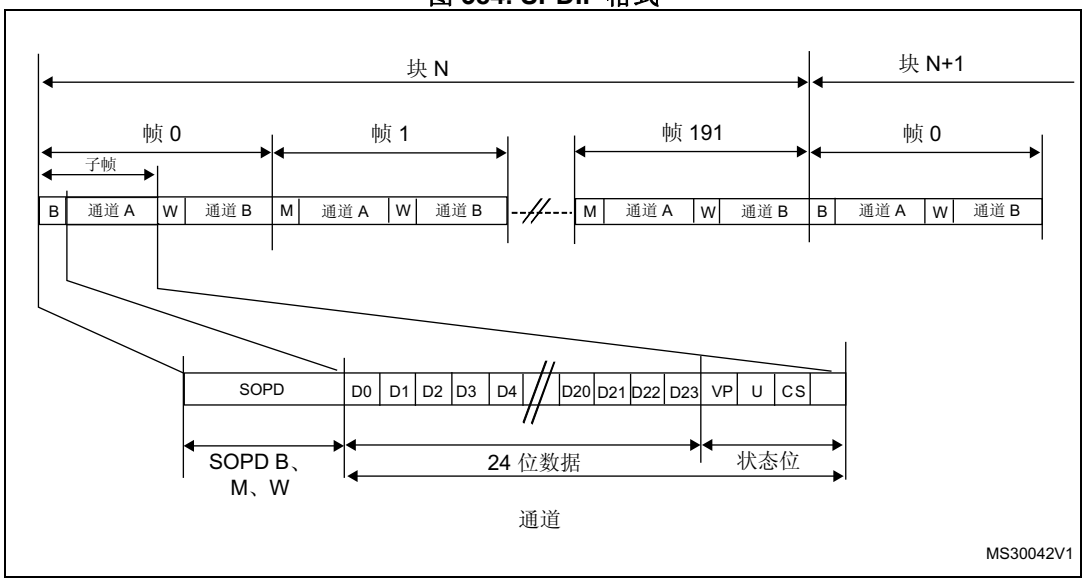
SPDIF 接口仅在发送模式下可用。它支持 IEC60958 音频标准。

要选择 SPDIF 模式，需将 SAI_xCR1 寄存器中的 PRTCFG[1:0] 位置为 01。

对于 SPDIF 协议：

- 仅使能 SD 数据线。
- 释放 FS、SCK、MCLK I/O 引脚。
- 为使能 SAI 的时钟发生器并管理 SD 线上的数据速率，将 MODE[1] 位强制设置为 0 以选择主模式。
- 数据大小强制设置为 24 位。忽略在 SAI_xCR1 寄存器的 DS[2:0] 位中设置的值。
- 必须配置时钟发生器以定义符号率，已知位时钟应为符号率的两倍。数据采用曼彻斯特协议进行编码。
- 忽略 SAI_xFRCR 和 SAI_xSLOTR 寄存器。在内部配置 SAI 使其符合 SPDIF 协议要求，如*图 384* 所示。

图 384. SPDIF 格式



SPDIF 块包含 192 个帧。每个帧由两个 32 位的子帧构成，通常一个子帧用于左通道，一个用于右通道。每个子帧由一个 SOPD 模式（4 位）构成，用于指定该子帧是否为块的开始（并用于识别通道 A），或者是否标识块中某处的通道 A，或者是否指代通道 B（请参见*表 182*）。接下来的 28 位是通道信息，由 24 个数据位和 4 个状态位组成。

表 182. SOPD 模式

SOPD	报头编码		说明
	最后一位是 0	最后一位是 1	
B	11101000	00010111	块开始处的通道 A 数据
W	11100100	00011011	块中某处的通道 B 数据
M	11100010	00011101	通道 A 数据

必须按如下方式填充 SAI_xDR 中存储的数据：

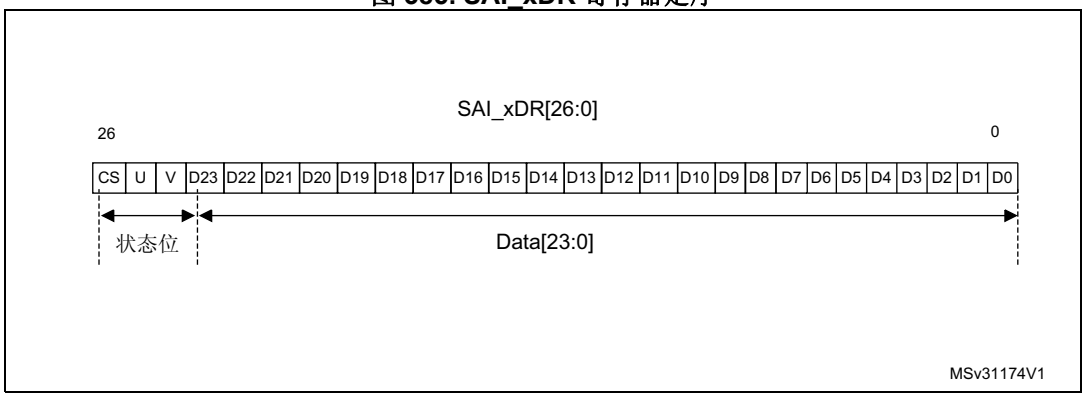
- SAI_xDR[26:24] 包含通道状态位、用户位和有效性位。
- SAI_xDR[23:0] 包含所考虑通道的 24 位数据。

如果数据大小为 20 位，应将数据映射到 SAI_xDR[23:4] 上。

如果数据大小为 16 位，应将数据映射到 SAI_xDR[23:8] 上。

SAI_xDR[23] 始终代表 MSB。

图 385. SAI_xDR 寄存器定序



注： *执行传输时，LSB 始终优先。*

SAI 首先在块中发送每个子帧的适当报头。随后在 SD 线上发送 SAI_xDR（以曼彻斯特协议进行编码）。SAI 通过传输按*表 183* 所述计算的奇偶校验位来结束子帧。

表 183. 奇偶校验位计算

SAI_xDR[26:0]	传输奇偶校验位 P 值
奇数个 0	0
奇数个 1	1

在 SPDIF 模式下，下溢是在 SAI_xSR 寄存器唯一个用到的错误标志。因为 SAI 只能在发送模式下工作。因此，要从通过下溢中断或下溢状态位检测到的下溢错误中恢复，应按顺序执行以下步骤：

1. 如果已使用 DMA，则禁止 DMA 流（通过 DMA 外设）。
2. 禁止 SAI 并通过轮询 SAI_xCR1 寄存器中的 SAIXEN 位确认已从物理上禁止外设。
3. 将 SAI_xCLRFR 寄存器中的 COVRUNDR 标志清零。
4. 通过将 SAI_xCR2 寄存器中的 FFLUSH 位置 1 来刷新 FIFO。
   软件需要指向与新块开始（报头 B 的数据）相对应的后续数据的地址。如果已使用 DMA，应相应地更新 DMA 源起始地址指针。
5. 如果用于管理数据的 DMA 根据新的源起始地址来传输数据，则再次使能 DMA 流（DMA 外设）。
6. 通过将 SAI_xCR1 寄存器中的 SAIXEN 位置 1 再次使能 SAI。

SPDIF 发生器模式下的时钟发生器编程

对于 SPDIF 发生器，SAI 应提供与符号率相等的位时钟。下表给出了符号率相对于音频采样率的通常示例。

表 184. 音频采样频率与符号率 (SHARK)

音频采样频率 (F_S)	符号率
44.1 kHz	2.8224 MHz
48 kHz	3.072 MHz
96 kHz	6.144 MHz
192 kHz	12.288 MHz

通常，音频采样率 (F_S) 与位时钟速率 (F_{SCK_X}) 之间的关系由以下公式给出：

$$F_S = \frac{F_{SAI_CK_x}}{64}$$

33.3.11 特性

根据所选的音频协议，SAI 接口内嵌了一些特定的实用功能。这些功能可通过 SAI_xCR2 寄存器的特定位来访问。

静音模式

当音频子模块用作发送器或接收器时，可使用静音模式。

发送模式下的音频子模块

在发送模式下，可随时选择静音模式。静音模式对于全部音频帧均有效。在帧传输过程中将 SAI_xCR2 寄存器的 MUTE 置 1，将使能静音模式。

该静音模式位仅在帧结束时选通。如果帧结束时置 1，静音模式将在新的音频帧开始时激活，并持续整个帧长度，直至下次帧结束；然后将选通该位，以确定下一帧是否仍为静音帧。

如果通过 SAI_xSLOTR 寄存器的 NBSLOT[3:0] 位设置的 Slot 数小于或等于 2，可指定静音模式下发送的值是否为 0 或该值是否为每个 Slot 的最后一个值。通过 SAI_xCR2 寄存器中的 MUTEVAL 位进行选择。

如果在 SAI_xSLOTR 寄存器的 NBSLOT[3:0] 位中设置的 Slot 数大于 2，由于在各 Slot 的每位上都发送值 0，因此 SAI_xCR2 中的 MUTEVAL 位没有意义。

在静音模式下，FIFO 指针仍递增， 这意味着将丢弃 FIFO 中请求在静音模式下传输的数据。

接收模式下的音频子模块

在接收模式下，对于给定数量的连续音频帧（SAI_xCR2 寄存器中的 MUTECNT[5:0] 位），如果在音频帧所有声明的有效 Slot 接收到 0，则可检测到从外部发送器发来的静音模式。

检测到相应数量的静音帧时，SAI_xSR 寄存器中的 MUTEDET 标志置 1 并会在 SAI_xCR2 中的 MUTEDETIE 位置 1 情况下产生中断。

在音频子模块禁止时或在有效 Slot 内至少接收到音频帧中的一个数据时，静音帧计数器清零。计数器达到 MUTECNT[5:0] 位中指定的值时，仅产生一次中断。随后中断事件在计数器清零时重新初始化。

注： *静音模式不可用于 SPDIF 音频模块。*

单声道 / 立体声模式

在发送模式下，如果 Slot 数等于 2（SAI_xSLOTR 中的 NBSLOT[3:0] = 0001），则可寻址单声道模式，而无需在存储器中进行任何数据预处理。在这种情况下，由于发送时 Slot 0 的数据被复制到数据 Slot 1 中，FIFO 的访问时间将减少一半。

要使能单声道模式，

1. 将 SAI_xCR1 寄存器中的 MONO 位置 1。
2. 将 SAI_xSLOTR 中的 NBSLOT 置 1，并将 SLOTEN 设置为 3。

在接收模式下，MONO 位可置 1 并且仅在 Slot 数等于 2（与发送模式下相同）时才有意义。当该位置 1 时，只有 Slot 0 的数据将存储到 FIFO 中。Slot 1 的数据由于被认为与前一个 Slot 的数据相同而被丢弃。如果接收模式下的数据流量是左声道数据和右声道数据明显不同的真立体声音频流，则 MONO 位没有意义。由软件完成从输出立体声文件到等效单声道文件的转换。

压扩模式

移动通信应用可能需要通过数据压扩算法处理待发送或待接收的数据。

应用可根据 SAI_xCR2 寄存器中的 COMP[1:0] 位（仅当选择 TDM 模式时使用），选择在 SD 串行输出线（压缩）发送数据前是否处理数据，以及是否在 SD 串行输入线（扩展）接收数据后扩展数据，如 *图 386* 所示。所支持的两个压扩模式是 μ-Law 和 A-Law，二者是 CCITT G.711 推荐标准的一部分。

美国和日本采用的压扩标准是 μ-Law， 该标准允许 14 位动态范围（SAI_xCR2 寄存器中的 COMP[1:0] = 10）。

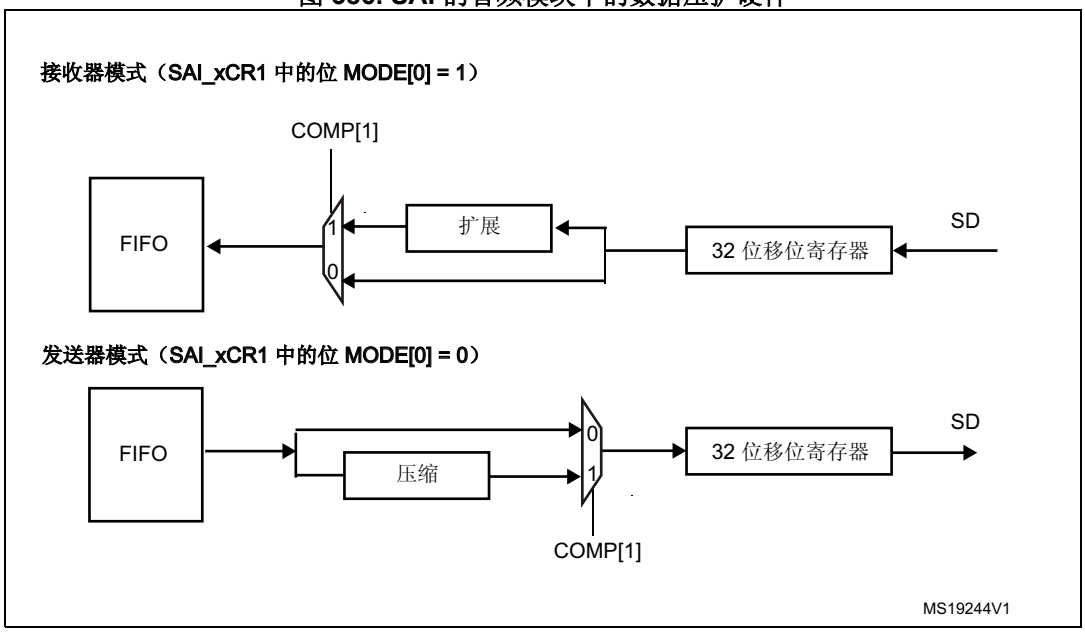
欧洲压扩标准是 A-Law，该标准支持 13 位动态范围（SAI_xCR2 寄存器中的 COMP[1:0] = 11）。

可根据 1 的补码或 2 的补码表示来计算 μ-Law 或 A-Law 压扩标准，具体取决于 SAI_xCR2 寄存器中的 CPL 位设置。

在 μ-Law 和 A-Law 标准中，数据将编码为采用 MSB 对齐的 8 位。压扩数据始终为 8 位宽。因此，当 SAI 音频模块使能（SAI_xCR1 寄存器中的位 SAIXEN = 1）并且通过 COMP[1:0] 位选择这两个压扩模式之一时，SAI_xCR1 寄存器中的 DS[2:0] 位将强制为 010。

如果无需压扩处理，则 COMP[1:0] 位应保持清零。

图 386. SAI 的音频模块中的数据压扩硬件



1. 选择 AC'97 时不适用。

通过 SAI_xCR2 自动选择扩展模式和压缩模式：

- 如果 SAI 音频模块配置为发送器，且 SAI_xCR2 寄存器中的 COMP[1] 位置 1，将采用压缩模式。
- 如果 SAI 音频模块声明为接收器，将采用扩展算法。

无效 Slot 上的输出数据线管理

在发送模式下，当在数据线上发送无效 Slot 时，可选择 SD 线输出的行为（通过 TRIS 位实现）。

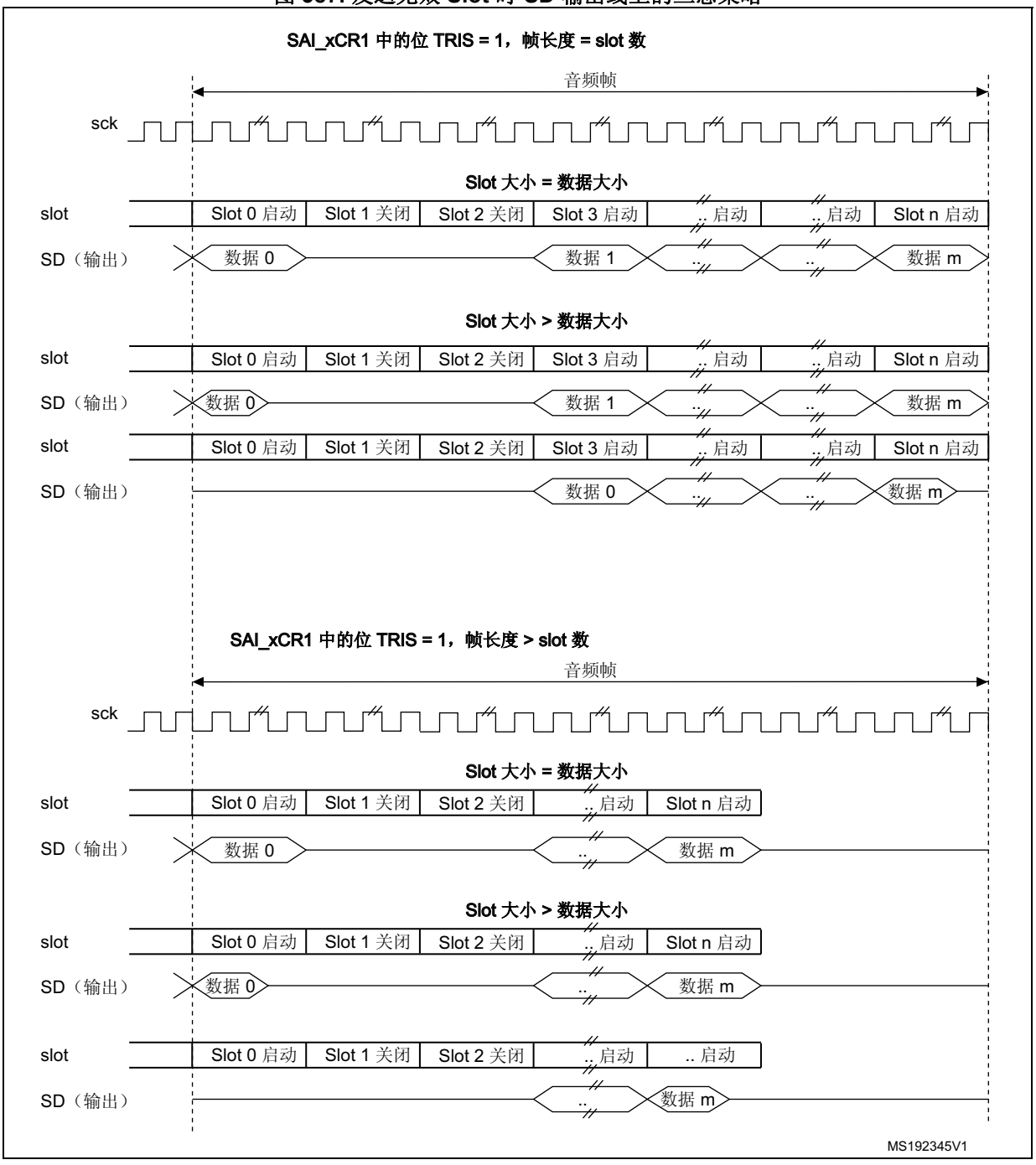
- 发送无效 Slot 时，SAI 将 SD 输出线强制为 0。
- 该输出线在最有一个数据位传输结束时释放为高阻态，为另一个连接此节点的发送器释放该数据线。

切记不要让两个发送器同时驱动同一个 SD 输出引脚，否则会导致短路。为确保存在发送间隙，如果数据低于 32 位，可通过在 SAI_xSLOTR 寄存器中设置 SLOTSZ[1:0] = 10 将数据扩展到 32 位。随后，如果下一 Slot 声明为无效，则 SD 输出引脚将在有效 Slot 的 LSD 结束时（将数据扩展到 32 位的填 0 阶段）置为三态。

此外，如果 Slot 数乘以 Slot 大小所得结果小于帧长度，则在通过填 0 来补充音频帧结束时，SD 输出线置为三态。

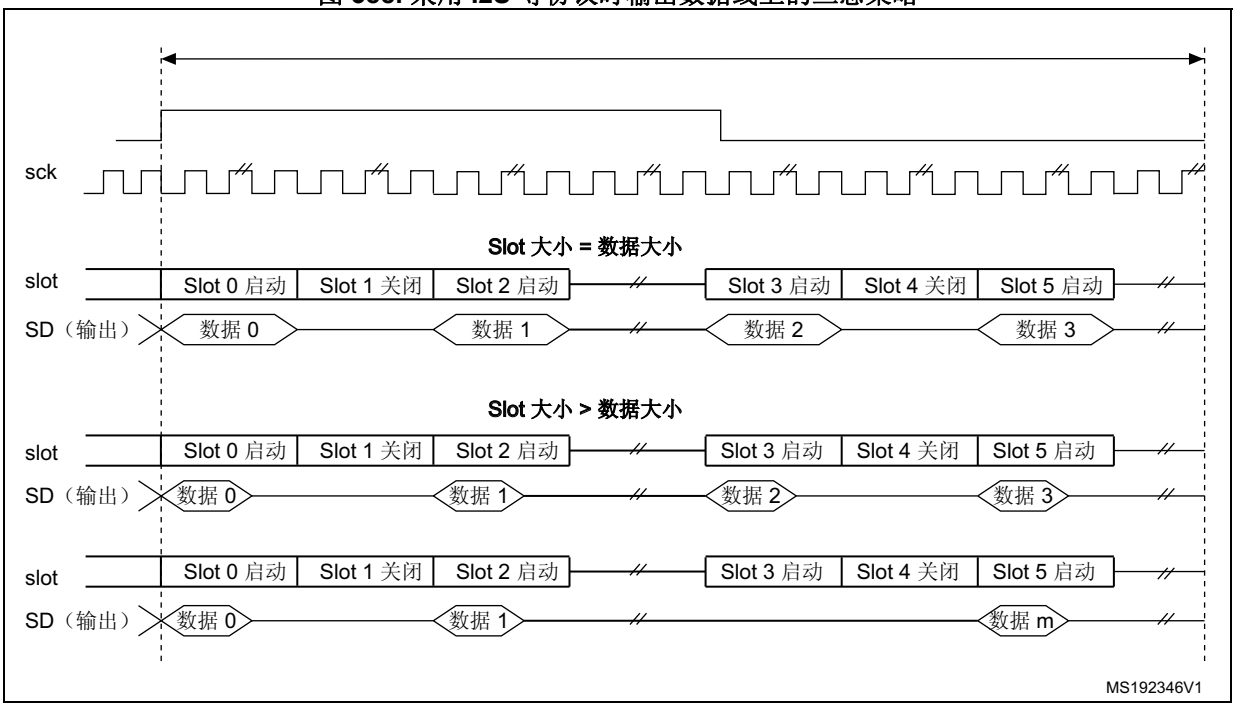
图 387 说明了这些行为。

图 387. 发送无效 Slot 时 SD 输出线上的三态策略



当所选音频协议使用 FS 信号作为 SOF 信号或通道识别信号（SAI_xFRCR 寄存器中的位 FSDEF = 1）时，将按照*图 388* 管理三态模式（其中，SAI_xCR1 寄存器中的位 TRIS = 1，FSDEF = 1，半帧长大于 Slot 数/2 且 NBSLOT = 6）。

图 388. 采用 I2S 等协议时输出数据线上的三态策略



如果 SAI_xCR2 寄存器中的 TRIS 位清零，*图 387* 和*图 388* 上的 SD 输出线上的所有高阻态将替换为使用值 0 驱动。

33.3.12 错误标志

SAI 使用以下错误标志：

- FIFO 上溢/下溢
- 帧同步提前检测
- 帧同步滞后检测
- 编解码器未就绪（仅限 AC'97）
- 主模式时钟配置错误。

FIFO 上溢/下溢 (OVRUDR)

FIFO 上溢/下溢位是 SAI_xSR 寄存器中的 OVRUDR 位。

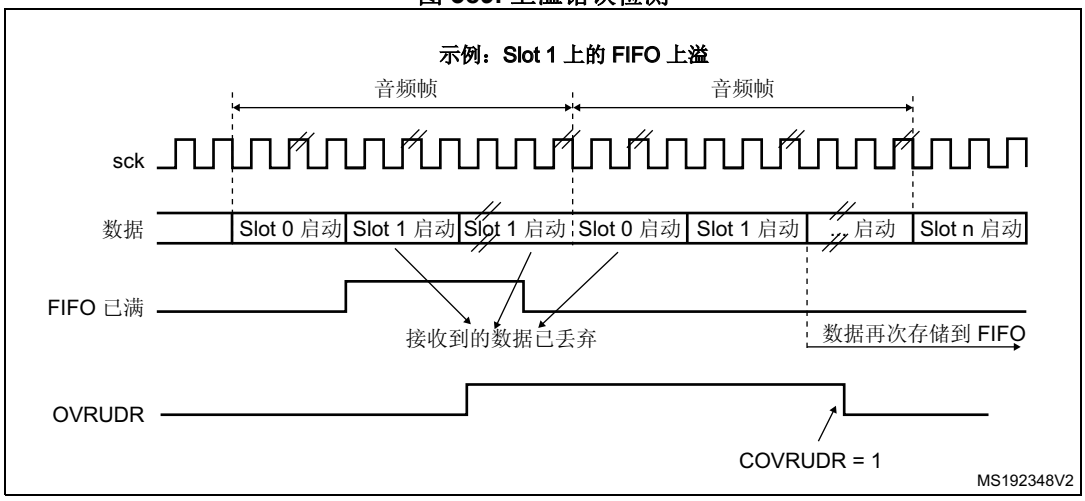
由于音频模块既可作为接收器，又可作为发送器，并且指定 SAI 中的每个音频模块都具有自己的 SAI_xSR 寄存器，因此上溢或下溢错误共用同一位。

上溢

若音频模块配置为接收器，则在 FIFO 已满且无法再存储接收数据的情况下又收到音频帧数据时，将出现上溢情况。这种情况下，接收数据将丢失，SAI_xSR 寄存器中的 OVRUDR 标志置 1；如果 SAI_xIM 寄存器中的 OVRUDRIE 位置 1，还将生成中断。内部将记录发生上溢时的 Slot 编号。FIFO 无法再存储更多数据，直至释放出空间存储新数据为止。在 FIFO 释放了至少一个数据的空间时，SAI 音频模块接收器将从检测到上溢后内部记录的 Slot 编号开始接收来自新音频帧的新数据， 这样可避免目标存储器中出现数据 Slot 不对齐的情况（请参见*图 389*）。

SAI_xCLRFR 寄存器中的 COVRUDR 位置 1 时将清除 OVRUDR 标志。

图 389. 上溢错误检测



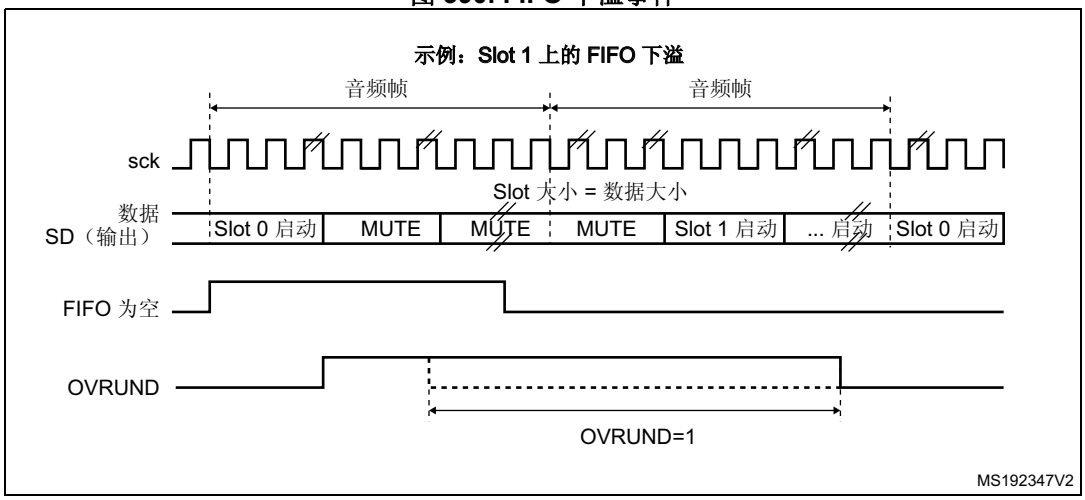
下溢

当 SAI 中的音频模块用作发送器时，如果需要发送数据时 FIFO 为空，则可能出现下溢。如果检测到下溢，则将存储发生事件的 Slot 编号并发送 MUTE 值 (00)，直到 FIFO 准备好发送与检测到下溢的 Slot 对应的数据（请参见*图 390*）。这样可避免存储器指针与音频帧中的 Slot 之间发生同步失效。

下溢事件会使 SAI_xSR 寄存器中的 OVRUDR 标志置 1，如果 SAI_xIM 寄存器中的 OVRUDRIE 位置 1，还将生成中断。要清除该标志，可将 SAI_xCLRFR 寄存器中的 COVRUDR 位置 1。

将音频子模块配置为主模式或从模式时，会发生下溢事件。

图 390. FIFO 下溢事件



帧同步提前检测 (AFSDET)

AFSDET 标志仅在从模式下使用。主模式下不会使能该标志。由于帧长度、帧极性和帧偏移已定义且已知，该标志用于指示是否比预期更早检测到帧同步 (FS) 信号。

出现帧检测提前时，SAI_xSR 寄存器中的 AFSDET 标志将置 1。

对 FS 提前不敏感的当前音频帧不受该检测影响。也就是说 FS 信号的“寄生”事件将被标记但不干扰当前音频帧。

如果 SAI_xIM 寄存器中的 AFSDETIE 位置 1，将生成中断。要清除 AFSDET 标志，必须将 SAI_xCLRFR 寄存器中的 CAFSDET 位置 1。

为了在出现帧检测提前错误后重新与主模块同步，需要执行以下四个步骤：

1. 通过复位 SAI_xCR1 寄存器中的 SAIXEN 位禁止 SAI 模块。为确保禁止 SAI，读回 SAIXEN 位并检查其是否已设置为 0。
2. 通过 SAI_xCR2 寄存器的 FFLUS 位刷新 FIFO。
3. 重新使能 SAI 外设（SAIXEN 位置 1）。
4. SAI 模块将等待 FS 使能以重新开始与主模块同步。

注： *AC’97 模式下不使能 SAIXEN 标志，原因是 SAI 音频模块作为链路控制器，即使在声明为从模块时也会生成 FS 信号。由于未使用 FS 信号，因此它在 SPDIF 模式下无意义。*

帧同步滞后检测

只有当 SAI 音频模块用作从模块时，SAI_xSR 寄存器中的 LFSDET 标志才可置 1。SAI_xFRCR 寄存器中，帧长度、帧极性和帧偏移配置均已知。

如果外部主模块未在预定时间发送 FS 信号生成过晚），LFSDET 标志将置 1，如果 SAI_xIM 寄存器中的 LFSDETIE 位置 1，还将生成中断。

SAI_xCLRFR 寄存器中的 CLFSDET 位置 1 时将清除 LFSDET 标志。

在检测到相应错误时帧同步滞后检测标志置 1， SAI 需要重新与主模块同步（请参见 *第节：帧同步提前检测 (AFSDET)* 中所述的顺序）。

在噪声环境中，音频模块的状态机可能会错误检测到对 SCK 时钟的干扰，并将 SAI 数据移位到错误的帧位置。SAI 会检测到此事件，并将其报告为帧同步滞后检测错误。

如果外部主模块不是在连续模式下管理音频数据帧发送，则不会对帧造成破坏，大多数应用都不会出现这种情况。这种情况下 LFSDET 标志将置 1。

注： *AC’97 模式下不使能 LFSDET 标志，原因是 SAI 音频模块作为链路控制器，即使在声明为从模块时也会生成 FS 信号。由于协议未使用 FS 信号，因此该位在 SPDIF 模式下无意义。*

编解码器未就绪 (CNRDY AC’97)

仅当 SAI 音频模块配置为在 AC’97 模式下工作时（SAI_xCR1 寄存器中的 PRTCFG[1:0] = 10），SAI_xSR 寄存器中的 CNRDY 标志才有意义。如果 SAI_xIM 寄存器中的 CNRDYIE 位置 1，则在 CNRDY 标志置 1 时将生成中断。

在接收 AC’97 音频帧的 TAG 0 (slot0) 期间，当编解码器未准备好进行通信时，将使能 CNRDY。这种情况下，在 TAG 0 指示编解码器就绪之前，数据都不会自动存储到 FIFO，原因是编解码器未就绪。编解码器就绪后将捕获 SAI_xSLOTR 寄存器中定义的所有有效 Slot。

要清除 CNRDY 标志，必须将 SAI_xCLRFR 寄存器中的 CCNRDY 位置 1。

主模式时钟配置错误 (NODIV = 0)

当音频模块在主模式下工作（MODE[1] = 0）且 NODIV 位等于 0 时，如果满足以下条件，则只要使能 SAI，WCKCFG 标志便会置 1：

- (FRL+1) 不是 2 的几次幂，并且
- (FRL+1) 不在 8 和 256 之间。

MODE 位、NODIV 位和 SAIXEN 位属于 SAI_xCR1 寄存器，FRL 位属于 SAI_xFRCR 寄存器。

如果 WCKCFGIE 位置 1，则当 SAI_xSR 寄存器中的 WCKCFG 标志置 1 时将生成中断。要清除该标志，可将 SAI_xCLRFR 寄存器中的 CWCKCFG 位置 1。

当 WCKCFG 位置 1 时，音频模块会自动禁止，因此将对 SAIXEN 位执行硬件清零。

33.3.13 禁止 SAI

可随时通过清零 SAI_xCR1 寄存器中的 SAIXEN 位禁止 SAI 音频模块。所有已开始的帧将在 SAI 停止工作前自动完成。SAIXEN 位将保持高电平，直到当前音频帧传输结束时 SAI 完全关闭。

如果 SAI 中有与另一个音频模块同步工作的音频模块，则必须先禁止以主模式工作的音频模块。

33.3.14 SAI DMA 接口

为减轻 CPU 负担和优化总线带宽，每个 SAI 音频模块都具有独立的 DMA 接口以便对 SAI_xDR 寄存器进行读/写操作（访问内部 FIFO）。每个音频子模块都有一个支持基本 DMA 请求/应答协议的 DMA 通道。

要为 DMA 传输配置音频子模块，可将 SAI_xCR1 寄存器中的 DMAEN 位置 1。DMA 请求直接由 FIFO 控制器管理，具体取决于 FIFO 阈值（更多详细信息，请参见*第 33.3.8 节：内部 FIFO*）。DMA 传输方向与 SAI 音频子模块配置相关：

- 如果音频模块用作发送器，则音频模块的 FIFO 控制器将输出 DMA 请求以向 FIFO 加载 SAI_xDR 寄存器中写入的数据。
- 如果音频模块用作接收器，则 DMA 请求与来自 SAI_xDR 寄存器的读取操作相关。

按照下面的顺序将 SAI 接口配置为 DMA 模式：

1. 配置 SAI 和 FIFO 阈值以指定何时启动 DMA 请求。
2. 配置 SAI DMA 通道。
3. 使能 DMA。
4. 使能 SAI 接口。

注： *配置 SAI 模块前，必须禁止 SAI DMA 通道。*

33.4 SAI 中断

SAI 支持 7 个中断源，如*表 185* 所示。

表 185. SAI 中断源

中断源	中断组	音频模块模式	中断使能	中断清零
FREQ	FREQ	主或从 接收器或发送器	SAI_xIM 寄存器中的 FREQIE	取决于： – FIFO 阈值设置（SAI_xCR2 中的 FLTH 位） – 通信方向（发送器或接收器） 更多详细信息，请参见*第 33.3.8 节：内部 FIFO*
OVRUDR	ERROR	主或从 接收器或发送器	SAI_xIM 寄存器中的 OVRUDRIE	SAI_xCLRFR 寄存器中的 COVRUDR = 1
AFSDET	ERROR	从 （不适用于 AC’97 模式和 SPDIF 模式）	SAI_xIM 寄存器中的 AFSDETIE	SAI_xCLRFR 寄存器中的 CAFSDET = 1
LFSDET	ERROR	从 （不适用于 AC’97 模式和 SPDIF 模式）	SAI_xIM 寄存器中的 LFSDETIE	SAI_xCLRFR 寄存器中的 CLFSDET = 1
CNRDY	ERROR	从 （仅限 AC’97 模式）	SAI_xIM 寄存器中的 CNRDYIE	SAI_xCLRFR 寄存器中的 CCNRDY = 1
MUTEDET	MUTE	主或从 仅限接收模式	SAI_xIM 寄存器中的 MUTEDETIE	SAI_xCLRFR 寄存器中的 CMUTEDET = 1
WCKCFG	ERROR	主模式且 SAI_xCR1 寄存器中的 NODIV = 0	SAI_xIM 寄存器中的 WCKCFGIE	SAI_xCLRFR 寄存器中的 CWCKCFG = 1

按照以下顺序使能中断：

1. 禁止 SAI 中断。
2. 配置 SAI。
3. 配置 SAI 中断源。
4. 使能 SAI。

33.5 SAI 寄存器

33.5.1 全局配置寄存器 (SAI_GCR)

Global configuration 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.	Res.	Res.	Res.	Res.	Res.	SYNCOUT[1:0]		Res.	Res.	SYNCIN[1:0]	
										rw	rw			rw	rw

位 31:6 保留，始终读为 0。

位 5:4 **SYNCOUT[1:0]**：同步输出 (Synchronization outputs)

这些位将由软件置 1 和清零。

00：无同步输出信号。将音频模块配置为 SPDIF 模式时，应将 SYNCOUT[1:0] 配置为无同步输出信号

01：模块 A 用于其它 SAI 的进一步同步

10：模块 B 用于其它 SAI 的进一步同步

11：保留。必须在音频模块（A 和 B）禁止的情况下设置这些位。

位 3:2 保留，始终读为 0。

位 1:0 **SYNCIN[1:0]**：同步输入 (Synchronization inputs)

这些位将由软件置 1 和清零。

有关如何编程此字段的信息，请参见*表 180：外部同步选择*。

必须在音频模块（A 和 B）禁止的情况下配置这些位。

如果将两个音频模块之一定义为与外部 SAI 在同步模式下工作（SAI_ACR1 或 SAI_BCR1 寄存器中的 SYNCEN[1:0] = 01），这些位便有意义。

33.5.2 配置寄存器 1 (SAI_ACR1/SAI_BCR1)

Configuration register 1

偏移地址：模块 A 为 0x004

偏移地址：模块 B 为 0x024

复位值：0x0000 0040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MCKDIV[3:0]				NODIV	Res.	DMAEN	SAIXEN
								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.	OUTDRIV	MONO	SYNCEN[1:0]		CKSTR	LSBFIRST	DS[2:0]			Res.	PRTCFG[1:0]		MODE[1:0]	
		rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw

位 31:24 保留，始终读为 0。

位 23:20 **MCKDIV[3:0]**：主时钟分频器 (Master clock divider)。

这些位由软件置 1 和清零。当音频模块在从模式下工作时，这些位无意义。必须在音频模块禁止的情况下配置这些位。

0000：主时钟输入 1 分频。

其它：主时钟频率将根据以下公式计算：

$$F_{SCK_x} = \frac{F_{SAI_CK_x}}{MCKDIV \times 2}$$

位 19 **NODIV**：无分频器 (No divider)。

此位由软件置 1 和清零。

0：使能主时钟发生器

1：在时钟发生器中不使用分频器（此时主时钟分频器位不起作用）

位 18 保留，始终读为 0。

位 17 **DMAEN**：DMA 使能 (DMA enable)。

此位由软件置 1 和清零。

0：禁止 DMA

1：使能 DMA

注： 在接收模式下，必须在 DMAEN 位置 1 前配置 MODE[1:0] 位，以避免 DMA 请求，原因是复位后音频模块将默认以发送模式工作。

位 16 **SAIXEN**：音频模块使能 (Audio block enable)，其中 x 为 A 或 B。

该位由软件置 1。

要关闭音频模块，必须由应用软件将该位编程为 0 并轮询该位，直到该位读回 0，这表示该模块已完全禁止。将该位置 1 前，先检查该位是否已置 0，否则不会执行此使能命令。

该位可用于控制 SAIx 音频模块的状态。如果在音频帧传输期间禁止该位，则正在进行的传输将完成并且单元在该音频帧传输结束时完全禁止。

0：禁止 SAIx 音频模块

1：使能 SAIx 音频模块。

注： 当 SAIx 模块配置为主模式时，SAIx 的输入中必须有内核时钟，然后才能将 SAIXEN 位置 1。

位 15:14 保留，始终读为 0。

位 13 **OUTDRIV**：输出驱动 (Output drive)。

此位由软件置 1 和清零。

0：当 SAIXEN 置 1 时驱动音频模块输出

1：在该位置 1 后立即驱动音频模块输出。

注： 该位必须在音频模块配置后的使能前置 1。

位 12 **MONO**：单声道模式 (Mono mode)。

此位由软件置 1 和清零。仅当 Slot 数为 2 时该位才有意义。如果选择了单声道模式，则当音频模块用作发送器时，Slot 0 的数据将复制到 Slot 1 上。在接收模式下，将丢弃 Slot 1 并仅存储从 Slot 0 接收的数据。更多详细信息，请参见 *单声道/ 立体声模式*一节。

0：立体声模式

1：单声道模式。

位 11:10 **SYNCEN[1:0]**：同步使能 (Synchronization enable)。

这些位将由软件置 1 和清零。必须在音频子模块禁止的情况下配置这些位。

00：音频子模块处于异步模式。

01：音频子模块与另一个内部音频子模块同步。这种情况下，必须将该音频子模块配置为从模式。

10：音频子模块与外部 SAI 的嵌入式外设同步。这种情况下，应将音频子模块配置为从模式。

11：保留

注： 使能 SPDIF 模式后，应将音频子模块配置为异步。

位 9 **CKSTR**：时钟选通边沿 (Clock strobing edge)。

此位由软件置 1 和清零。必须在音频模块禁止的情况下配置该位。SPDIF 音频协议下该位没有意义。

0：在 SCK 上升沿更改 SAI 生成的信号，而在 SCK 下降沿对 SAI 接收的信号进行采样。

1：在 SCK 下降沿更改 SAI 生成的信号，而在 SCK 上升沿对 SAI 接收的信号进行采样。

位 8 **LSBFIRST**：最低有效位优先 (Least significant bit first)。

此位由软件置 1 和清零。必须在音频模块禁止的情况下配置该位。AC’97 音频协议下该位没有意义，原因是传输 AC’97 数据时，数据的 MSB 位优先。SPDIF 音频协议下该位没有意义，原因是传输 SPDIF 数据时，数据的 LSB 位优先。

0：优先传输数据的 MSB

1：优先传输数据的 LSB

位 7:5 **DS[2:0]**：数据大小 (Data size)。

这些位将由软件置 1 和清零。选择 SPDIF 协议时（位 PRTCFG[1:0]），将忽略这些位，原因是在这种情况下帧和数据大小是固定的。通过 COMP[1:0] 位选择压扩模式时，将忽略 DS[1:0]，原因是算法已将数据大小固定为 8 位。

必须在音频模块禁止的情况下配置这些位。

000：保留

001：保留

010：8 位

011：10 位

100：16 位

101：20 位

110：24 位

111：32 位

位 4 保留，始终读为 0。

位 3:2 **PRTCFG[1:0]**：协议配置 (Protocol configuration)。

这些位将由软件置 1 和清零。必须在音频模块禁止的情况下配置这些位。

00：自由协议。通过设置大部分配置寄存器位以及帧配置寄存器，自由协议允许用户使用音频模块这一强大的配置功能来处理特定的音频协议（如 I2S、LSB/MSB 对齐、TDM、PCM/DSP...）。

01：SPDIF 协议

10：AC’97 协议

11：保留

位 1:0 **MODE[1:0]**：SAIx 音频模块模式 (SAIx audio block mode)。

这些位将由软件置 1 和清零。必须在 SAIx 音频模块禁止的情况下配置这些位。

00：主发送器

01：主接收器

10：从发送器

11：从接收器

注： 如果将音频模块配置为 SPDIF 模式，则将强制设置主发送模式 (MODE[1:0] = 00)。在主发送模式下，音频模块将立即开始生成 FS 和时钟。

33.5.3 配置寄存器 2 (SAI_ACR2/SAI_BCR2)

Configuration register 2

偏移地址：模块 A 为 0x008

偏移地址：模块 B 为 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
COMP[1:0]		CPL	MUTECNT[5:0]						MUTE VAL	MUTE	TRIS	FFLUSH	FTH		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	w	rw	rw	rw

位 31:16 保留，始终读为 0

位 15:14 **COMP[1:0]**：压扩模式 (Companding mode)。

这些位将由软件置 1 和清零。μ-Law 和 A-Law 算法是 CCITT G.711 建议的一部分，要使用何种补码类型取决于 *CPL 位*。

数据扩展还是数据压缩由 MODE[0] 位的状态确定。

如果将音频模块配置为发送器，则应用数据压缩。

如果将音频模块配置为接收器，则自动应用数据扩展。

更多详细信息，请参见*压扩模式*一节。

00：不支持压扩算法

01：保留。

10：μ-Law 算法

11：A-Law 算法

注： 仅当选择了 TDM 协议时才能使用压扩模式。

位 13 **CPL**：补码位 (Complement bit)。

此位由软件置 1 和清零。

该位定义用于压扩模式的补码类型。

0：1 的补码表示。

1：2 的补码表示。

注： 仅当压扩模式为 μ-Law 算法或 A-Law 算法时该位才有效。

位 12:7 **MUTECNT[5:0]**：静音计数器 (Mute counter)。

这些位将由软件置 1 和清零。这些位仅用于接收模式。

这些位中所设置的值将与接收模式下检测到的连续静音帧数量进行比较。当静音帧数量与该值相等时，MUTEDET 标志置 1，并且在 MUTEDETIE 位置 1 的情况下，还将生成中断。

更多详细信息，请参见*静音模式*一节。

位 6 **MUTEVAL**：静音值 (Mute value)。

该位由软件置 1 和清零。必须在使能音频模块 (SAIxEN) 前写入。仅当音频模块用作发送器、Slot 数小于或等于 2 并且 MUTE 位置 1 时，该位才有意义。

如果声明了 2 个以上的 Slot，则无论 MUTEVAL 位的值为何，静音模式下发送的位值都将等于 0。

如果 Slot 数小于或等于 2 且 MUTEVAL = 1，则为每个 Slot 发送的 MUTE 值将是上一帧期间发送的值。

更多详细信息，请参见*静音模式*一节。

0：静音模式期间发送位值 0。

1：静音模式期间发送上一个值。

注： 该位对 SPDIF 音频模块无意义，从而也不使用。

位 5 **MUTE**：静音 (Mute)。

此位由软件置 1 和清零。仅当音频模块用作发送器时该位才有意义。Slot 数小于或等于 2 时，MUTE 值与 MUTEVAL 值相关；Slot 数大于 2 时，MUTE 值等于 0。

更多详细信息，请参见*静音模式*一节。

0：禁止静音模式。

1：使能静音模式。

注： 该位对 SPDIF 音频模块无意义，从而也不使用。

位 4 **TRIS**：数据线的三态管理 (Tristate management on data line)。

此位由软件置 1 和清零。仅当将音频模块配置为发送器时该位才有意义。当音频模块配置为 SPDIF 模式时不使用该位。应在 SAI 禁止时配置此位。

更多详细信息，请参见*无效 Slot 上的输出数据线管理*一节。

0：Slot 无效时，SD 输出线仍由 SAI 驱动。

1：SD 输出线将在上一个有效 Slot（下一个 Slot 无效）的最后一个数据位传输结束时释放（高阻态）。

位 3 **FFLUSH**：FIFO 刷新 (FIFO flush)。

该位由软件置 1， 始终读为 0。应在 SAI 禁止时配置此位。

0：禁止 FIFO 刷新。

1：FIFO 刷新。将此位编程为 1 可触发 FIFO 刷新。所有的内部 FIFO 指针（读和写）将清零。这种情况下，仍存留在 FIFO 中的数据丢失（发送或接收数据不会继续丢失）。刷新 SAI 前，必须禁止 DMA 数据流 / 中断。

位 2:0 **FTH**：FIFO 阈值 (FIFO threshold)。

此位由软件置 1 和清零。

000：FIFO 为空

001：¼ FIFO

010：½ FIFO

011：¾ FIFO

100：FIFO 已满

101：保留

110：保留

111：保留

33.5.4 帧配置寄存器 (SAI_AFRCR/SAI_BFRCR)

Frame configuration register

偏移地址：模块 A 为 0x00C

偏移地址：模块 B 为 0x02C

复位值：0x0000 0007

注： *该寄存器对于 AC'97 和 SPDIF 音频协议无意义*

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.	FSOFF	FSPOL	FSDEF
													rw	rw	r

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

位 31:19 保留，始终读为 0。

位 18 **FSOFF**：帧同步偏移 (Frame synchronization offset)。

此位由软件置 1 和清零。该位对 AC'97 或 SPDIF 音频模块配置无意义，从而也不使用。必须在音频模块禁止的情况下配置该位。

0：在 Slot 0 的第一位上使能 FS。

1：在 Slot 0 第一位的前一位上使能 FS。

位 17 **FSPOL**：帧同步极性 (Frame synchronization polarity)。

此位由软件置 1 和清零。该位用于配置 FS 信号上的 SOF 电平。该位对 AC'97 或 SPDIF 音频模块配置无意义，从而也不使用。

必须在音频模块禁止的情况下配置该位。

0：FS 为低电平有效（下降沿）

1：FS 为高电平有效（上升沿）

位 16 **FSDEF**：帧同步定义 (Frame synchronization definition)。

此位由软件置 1 和清零。

0：FS 信号为起始帧信号

1：FS 信号为 SOF 信号 + 通道识别信号

此位置 1 时，SAI_xSLOTR 寄存器中定义的 Slot 数必须为偶数。这意味着有半数 Slot 将用于左通道，其它 Slot 用于右通道（例如，对于 I2S 或 MSB/LSB 对齐等协议，该位必须置 1）。

此位对 AC'97 或 SPDIF 音频模块配置无意义，从而也不使用。必须在音频模块禁止的情况下配置该位。

位 15 保留，始终读为 0。

位 14:8 **FSALL[6:0]**：帧同步有效电平长度 (Frame synchronization active level length)。
这些位将由软件置 1 和清零。这些位用于指定音频帧中 FS 信号的有效电平长度，以位时钟数 (SCK) + 1 (FSALL[6:0] + 1) 表示这些位对 AC'97 或 SPDIF 音频模块配置无意义，从而也不使用。
必须在音频模块禁止的情况下配置这些位。

位 7:0 **FRL[7:0]**：帧长度 (Frame length)。
这些位将由软件置 1 和清零。这些位用于定义以 SCK 时钟周期数表示的音频帧长度：帧中的位数等于 FRL[7:0] + 1。
音频帧中发送的位数必须大于或等于 8，否则音频模块将出现操作异常。数据大小为 8 位且在 SAI_xSLOTR 寄存器的 NBSLOT[4:0] 中只定义了一个 Slot (NBSLOT[3:0] = 0000) 时便属于这种情况。
在主模式下，如果使用主时钟（MCLK_x 引脚上提供），则帧长度应为 8 到 256 之间的一个等于 2 的几次幂的数。不使用主时钟 (NODIV = 1) 时，建议将帧长度编程为 8 到 256 之间的值。
这些位对 AC'97 或 SPDIF 音频模块配置无意义，从而也不使用。

33.5.5 Slot 寄存器 (SAI_ASLOTR/SAI_BSLOTR)

Slot register

偏移地址：模块 A 为 0x010

偏移地址：模块 B 为 0x030

复位值：0x0000 0000

注： *该寄存器对于 AC'97 和 SPDIF 音频协议无意义*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLOTEN[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.	NBSLOT[3:0]				SLOTSZ[1:0]		Res.	FBOFF[4:0]				
				rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw

位 31:16 **SLOTEN[15:0]**：Slot 使能 (Slot enable)。
这些位将由软件置 1 和清零。
每个 SLOTEN 位都对应于从 0 到 15 的一个 Slot 位置（最多 16 个 Slot）。
0：无效 Slot。
1：有效 Slot。
必须在音频模块禁止的情况下使能 Slot。
在 AC'97 或 SPDIF 模式下会忽略这些位。

位 15:12 保留，始终读为 0。

位 11:8 **NBSLOT[3:0]**：音频帧中的 Slot 数 (Number of slots in an audio frame)。
这些位将由软件置 1 和清零。
此位字段中设置的值表示音频帧中的 Slot 数 + 1（包括无效 Slot 数）。Slot 数最大值为 16。
SAI_xFRCR 寄存器中的 FSDEF 位置 1 时，Slot 数应为偶数。
必须在音频模块禁止的情况下配置 Slot 数。
在 AC'97 或 SPDIF 模式下会忽略这些位。

位 7:6 **SLOTSZ[1:0]**：Slot 大小 (Slot size)。
此位由软件置 1 和清零。
Slot 大小必须大于或等于数据大小。如果不满足该条件，SAI 的行为将不确定。
有关如何驱动 SD 线的信息，请参见*无效 Slot 上的输出数据线管理*一节。
必须在音频模块禁止的情况下配置这些位。
在 AC'97 或 SPDIF 模式下会忽略这些位。
00：Slot 大小与数据大小（在 SAI_xCR1 寄存器的 DS[3:0] 位中指定）相当。
01：16 位
10：32 位
11：保留

位 1 保留，始终读为 0。

位 4:0 **FBOFF[4:0]**：第一个位偏移 (First bit offset)。
这些位将由软件置 1 和清零。
此位字段中设置的值定义 Slot 中第一个数据传输位的位置。它表示一个偏移值。在发送模式下，此数据字段以外的位将强制清零。在接收模式下，将丢弃额外接收的位。
必须在音频模块禁止的情况下配置这些位。
在 AC'97 或 SPDIF 模式下会忽略这些位。

33.5.6 中断屏蔽寄存器 2 (SAI_AIM/SAI_BIM)

Interrupt mask register 2

偏移地址：模块 A 为 0x014

偏移地址：模块 B 为 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.	Res.	Res.	Res.	Res.	LFSDETIE	AFSDETIE	CNRDYIE	FREQIE	WCKCFGIE	MUTEDETIE	OVRUDRIE
									rw	rw	rw	rw	rw	rw	rw

位 31:7 保留，始终读为 0。

位 6 **LFSDETIE**：帧同步滞后检测中断使能 (Late frame synchronization detection interrupt enable)。
此位由软件置 1 和清零。
0：禁止中断
1：使能中断
该位置 1 时，若 SAI_xSR 寄存器中的 LFSDET 位置 1，则生成中断。
该位对于 AC'97、SPDIF 模式无意义；若音频模块为主模块，该位也无意义。

位 5 **AFSDETIE**：帧同步提前检测中断使能 (Anticipated frame synchronization detection interrupt enable)。
此位由软件置 1 和清零。
0：禁止中断
1：使能中断
该位置 1 时，若 SAI_xSR 寄存器中的 AFSDET 位置 1，则生成中断。
该位对于 AC'97、SPDIF 模式无意义；若音频模块为主模块，该位也无意义。

位 4 **CNRDYIE**：编解码器未就绪中断使能 (Codec not ready interrupt enable) (AC’97)。
此位由软件置 1 和清零。
0：禁止中断
1：使能中断
若使能该中断，音频模块将在 AC’97 帧的 Slot 0 (tag0) 中检测连接到该线路的编解码器是否就绪。
如果未就绪，SAI_xSR 寄存器中的 CNRDY 标志置 1，并生成中断。
仅当通过 PRTCFG[1:0] 位选择了 AC’97 模式且音频模块用作接收器时，该位才有意义。

位 3 **FREQIE**：FIFO 请求中断使能 (FIFO request interrupt enable)。
此位由软件置 1 和清零。
0：禁止中断
1：使能中断
该位置 1 时，若 SAI_xSR 寄存器中的 FREQ 位置 1，则生成中断。
在接收模式下，必须在 FREQIE 位置 1 前配置 MODE 位，以避免寄生中断，原因是复位后音频模块将默认用作发送器。

位 2 **WCKCFGIE**：时钟配置错误中断使能 (Wrong clock configuration interrupt enable)。
此位由软件置 1 和清零。
0：禁止中断
1：使能中断
仅在将音频模块配置为主模式 (MODE[1] = 0) 且 NODIV = 0 时才执行此位。
该位在 SAI_xSR 寄存器中的 WCKCFG 标志置 1 时生成中断。
注：　该位仅用于 TDM 模式，其它模式下没有意义。

位 1 **MUTEDETIE**：静音检测中断使能 (Mute detection interrupt enable)。
此位由软件置 1 和清零。
0：禁止中断
1：使能中断
该位置 1 时，若 SAI_ASR 寄存器中的 MUTEDET 位置 1，则生成中断。
仅当音频模块配置为以发送模式工作时该位才有意义。

位 0 **OVRUDRIE**：上溢/下溢中断使能 (Overrun/underrun interrupt enable)。
此位由软件置 1 和清零。
0：禁止中断
1：使能中断
该位置 1 时，若 SAI_ASR 寄存器中的 OVRUDR 位置 1，则生成中断。

33.5.7 状态寄存器 (SAI_ASR/SAI_BSR)

Status register

偏移地址：模块 A 为 0x018

偏移地址：模块 B 为 0x038

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.	FLTH		
													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.	LFSDET	AFSDET	CNRDY	FREQ	WCKCFG	MUTEDET	OVRUDR
									r	r	r	r	r	r	r

位 31:19 保留，始终读为 0。

位 18:16 **FLTH**：FIFO 阈值 (FIFO level threshold)。

该位为只读。FIFO 阈值标志只通过硬件管理，其设置取决于 SAI 模块的配置（发送器或接收器模式）。

如果将 SAI 模块配置为发送器：

000：FIFO 为空

001：FIFO <= ¼，但非空

010：¼ < FIFO <= ½

011：½ < FIFO <= ¾

100：¾ < FIFO，但未满

101：FIFO 已满

如果将 SAI 模块配置为发送器：

000：FIFO 为空

001：FIFO < ¼，但非空

010：¼ <= FIFO < ½

011：½ =< FIFO < ¾

100：¾ =< FIFO，但未满

101：FIFO 已满

位 15:7 保留，始终读为 0。

位 6 **LFSDET**：帧同步滞后检测 (Late frame synchronization detection)。

该位为只读。

0：无错误。

1：帧同步信号未在正确的时刻出现。

仅当音频模块配置为以从模式工作时，此标志才能置 1。

不适用于 AC'97 或 SPDIF 模式。

它可在 SAI_xIM 寄存器中的 LFSDETIE 位置 1 时生成中断。

在软件将 SAI_xCLRFR 寄存器中的 CLFSDET 位置 1 时清除该标志。

位 5 **AFSDET**：帧同步提前检测 (Anticipated frame synchronization detection)。
该位为只读。
0：无错误。
1：提前检测到帧同步信号。
仅当音频模块配置为以从模式工作时，此标志才能置 1。
不适用于 AC'97 或 SPDIF 模式。
它可在 SAI_xIM 寄存器中的 AFSDETIE 位置 1 时生成中断。
在软件将 SAI_xCLRFR 寄存器中的 CAFSDET 位置 1 时清除该标志。

位 4 **CNRDY**：编解码器未就绪 (Codec not ready)。
该位为只读。
0：外部 AC'97 编解码器已就绪
1：外部 AC'97 编解码器未就绪
仅当在 SAI_xCR1 寄存器中选择了 AC'97 音频模块并且音频模块配置为接收器模式时，才使用该位。
它可在 SAI_xIM 寄存器中的 CNRDYIE 位置 1 时生成中断。
在软件将 SAI_xCLRFR 寄存器中的 CCNRDY 位置 1 时清除该标志。

位 3 **FREQ**：FIFO 请求 (FIFO request)。
该位为只读。
0：无 FIFO 请求。
1：FIFO 请求读取或写入 SAI_xDR。
请求内容取决于音频模块的配置：
– 如果模块配置为发送模式，则 FIFO 请求与向 SAI_xDR 中写入相关。
– 如果音频模块配置为接收模式，则 FIFO 请求与从 SAI_xDR 中读取相关。
此标志可在 SAI_xIM 寄存器中的 FREQIE 位置 1 时生成中断。

位 2 **WCKCFG**：时钟配置错误标志 (Wrong clock configuration flag)。
该位为只读。
0：时钟配置正确
1：时钟配置不符合*第 33.3.5 节：帧同步*中定义的帧长度规范（SAI_xFRCR 寄存器中 FRL[7:0] 位的配置）
该位仅在音频模块工作在主模式 (MODE[1] = 0) 下且 NODIV = 0 时使用。
它可在 SAI_xIM 寄存器中的 WCKCFGIE 位置 1 时生成中断。
在软件将 SAI_xCLRFR 寄存器中的 CWCKCFG 位置 1 时清除该标志。

位 1 **MUTEDET**：静音检测 (Mute detection)。
该位为只读。
0：SD 输入线上未检测到 MUTE 值
1：在SD 输入线上检测到指定数量的连续音频帧中的 MUTE 值（0 值）
如果在指定音频帧的每个 Slot 或在一定数量（在 SAI_xCR2 寄存器中的 MUTECNT 位中设置）的连续音频帧中接收到连续的 0 值，则该标志置 1。
它可在 SAI_xIM 寄存器中的 MUTEDETIE 位置 1 时生成中断。
在软件将 SAI_xCLRFR 寄存器中的 CMUTEDET 位置 1 时清除该标志。

位 0 **OVRUDR**：上溢/下溢 (Overrun/underrun)。
该位为只读。
0：无上溢/下溢错误。
1：检测到上溢/下溢错误。
仅当音频模块分别被配置为接收器和发送器时才会发生上溢和下溢情况。
它可在 SAI_xIM 寄存器中的 OVRUDRIE 位置 1 时生成中断。
在软件将 SAI_xCLRFR 寄存器中的 COVRUDR 位置 1 时清除该标志。

33.5.8 清除标志寄存器 (SAI_ACLRFR/SAI_BCLRFR)

Clear flag register

偏移地址：模块 A 为 0x01C

偏移地址：模块 B 为 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.	Res.	Res.	Res.	Res.

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.	CLFSDET	CAFSDET	CCNRDY	Res.	CWCKCFG	CMUTEDET	COVRUDR
									w	w	w		w	w	w

位 31:7 保留，始终读为 0。

位 6 **CLFSDET**：清除帧同步滞后检测标志 (Clear late frame synchronization detection flag)。
该位为只写位。
将此位编程为 1 可清除 SAI_xSR 寄存器中的 LFSDET 标志。
该位不适用于 AC’97 或 SPDIF 模式。
读取该位将始终返回值 0。

位 5 **CAFSDET**：清除帧同步提前检测标志 (Clear anticipated frame synchronization detection flag)。
该位为只写位。
将此位编程为 1 可清除 SAI_xSR 寄存器中的 AFSDET 标志。
不适用于 AC’97 或 SPDIF 模式。
读取该位将始终返回值 0。

位 4 **CCNRDY**：清除编解码器未就绪标志 (Clear codec not ready flag)。
该位为只写位。
将此位编程为 1 可清除 SAI_xSR 寄存器中的 CNRDY 标志。
仅当在 SAI_xCR1 寄存器中选择了 AC’97 音频协议时，才使用该位。
读取该位将始终返回值 0。

位 3 保留，始终读为 0。

位 2 **CWCKCFG**：清除时钟配置错误标志 (Clear wrong clock configuration flag)。
该位为只写位。
将此位编程为 1 可清除 SAI_xSR 寄存器中的 WCKCFG 标志。
仅当音频模块设置为主模块时 (MODE[1] = 0) 且 SAI_xCR1 寄存器中的 NODIV = 0 时，才使用此位。
读取该位将始终返回值 0。

位 1 **CMUTEDET**：静音检测标志 (Mute detection flag)。
该位为只写位。
将此位编程为 1 可清除 SAI_xSR 寄存器中的 MUTEDET 标志。
读取该位将始终返回值 0。

位 0 **COVRUDR**：清除上溢/下溢标志 (Clear overrun/underrun)。
该位为只写位。
将此位编程为 1 可清除 SAI_xSR 寄存器中的 OVRUDR 标志。
读取该位将始终返回值 0。

33.5.9 数据寄存器 (SAI_ADR/SAI_BDR)

Data register

偏移地址：模块 A 为 0x020

偏移地址：模块 B 为 0x040

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]**：数据 (Data)

若 FIFO 未满，写入该寄存器可向 FIFO 加载数据。

若 FIFO 非空，读取该寄存器可清空 FIFO。

33.5.10 SAI 寄存器映射

下表对 SAI 寄存器进行了汇总。

表 186. SAI 寄存器映射和复位值

偏移	寄存器和复位值	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
0x0000	SAI_GCR	Reserved																										SYNCOUT[1:0]		Reserved		SYNCIN[1:0]	
	Reset value																											0	0			0	0
0x0004 或 0x0024	SAI_xCR1	Reserved								MCJDIV[3:0]				NODIV	Reserved	DMAEN	SAIXEN			OUTDRIV	MONO	SYNCEN[1:0]		CKSTR	LSBFIRST	DS[2:0]		Reserved		PRTCFG[1:0]		MODE[1:0]	
	Reset value									0	0	0	0	0		0	0			0	0	0	0	0	0	0	1	0		0	0	0	0
0x0008 或 0x0028	SAI_xCR2	Reserved																COMP[1:0]		CPL	MUTECN[5:0]							MUTE VAL	MUTE	TRIS	FFLUS	FTH	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x000C 或 0x002C	SAI_xFRCR	Reserved													FSOFF	FSPOL	FSDEF	Reserved	FSALL[6:0]							FRL[7:0]								
	Reset value														0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

表 186. SAI 寄存器映射和复位值（续）

偏移	寄存器和复位值	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
0x0010 或 0x0030	SAI_xSLOTR	SLOTEN[15:0]																Reserved	Reserved	Reserved	Reserved	NBSLOT[3:0]				SLOTSZ[1:0]		Reserved	FBOFF[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	0
0x0014 或 0x0034	SAI_xIM	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	LFSDET	AFSDETIE	CNRDYIE	FREQIE	WCKCFG	MUTEDET	OVRUDRIE
	Reset value																										0	0	0	0	0	0	0
0x0018 或 0x0038	SAI_xSR	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	FLVL[2:0]			Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	LFSDET	AFSDET	CNRDY	FREQ	WCKCFG	MUTEDET	OVRUDR
	Reset value														0	0	0										0	0	0	0	1	0	0
0x001C 或 0x003C	SAI_xCLRFR	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	LFSDET	CAFSDET	CNRDY	Reserved	WCKCFG	MUTEDET	OVRUDR
	Reset value																										0	0	0		0	0	0
0x0020 或 0x0040	SAI_xDR	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

有关寄存器边界地址的信息，请参见*第 2.2.2 节：存储器映射和寄存器边界地址*。

34 SPDIF 接收器接口 (SPDIFRX)

34.1 SPDIFRX 接口简介

SPDIFRX 接口可以管理 S/PDIF 音频协议。

34.2 SPDIFRX 主要特性

- 提供多达 4 路输入
- 自动符号率检测
- 最大符号率：12.288 MHz
- 支持 8 kHz 到 192 kHz 的立体声
- 支持 IEC-60958 和 IEC-61937 音频标准，消费类应用
- 奇偶校验位管理
- 使用 DMA 通信进行音频采样
- 对于控制和用户信息采用 DMA 通信
- 中断功能

34.3 SPDIFRX 功能说明

SPDIFRX 外设用于接收符合 IEC-60958 和 EC-61937 标准的 S/PDIF 流。这些标准支持高采样率的简单立体声以及压缩的多通道环绕声，例如由 Dolby 或 DTS 定义的音频。

接收器提供所有必要的功能来检测符号率并解码传入的数据。可以使用用户和通道信息的专用路径来简化接口处理过程。*图 391* 给出了简化框图。

SPDIFRX_DC 模块负责解码从 SPDIFRX_IN[4:1] 输入接收的 S/PDIF 数据流。该模块重新采样传入的信号，解码曼彻斯特数据流，并识别帧、子帧和块元素。它传送到 REG_IF 部分、解码数据和相关的状态标志。

该外设可通过 APB1 总线完全控制并且能够处理两个 DMA 通道：

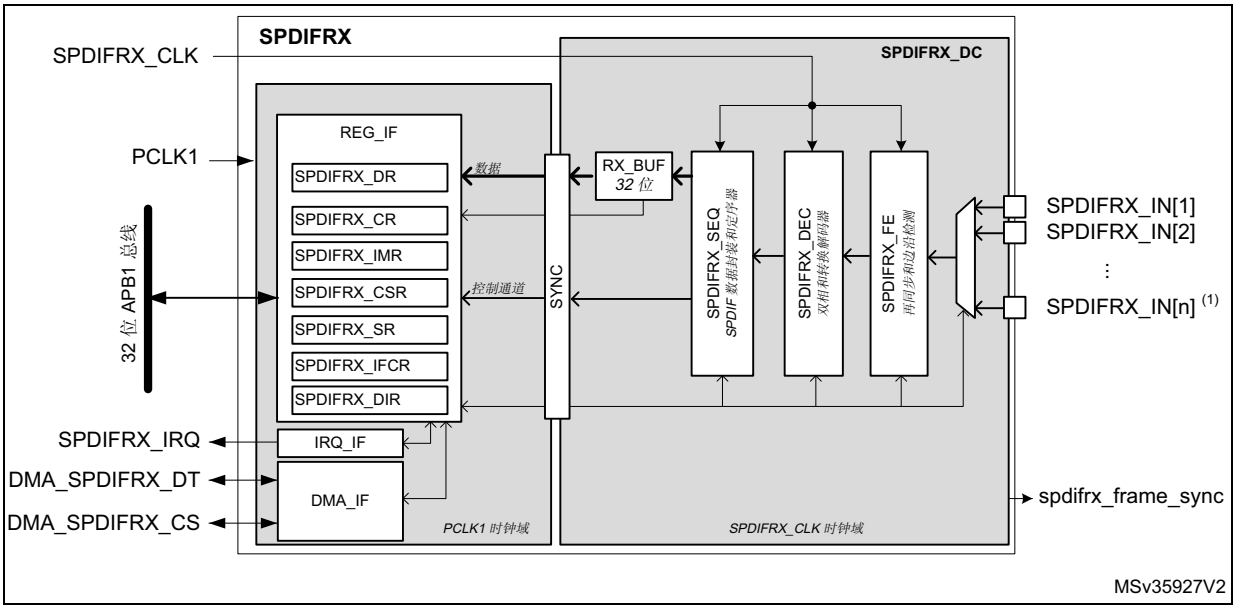
- 一个专用于传输音频采样的 DMA 通道
- 一个专用于传输 IEC60958 通道状态和用户信息的 DMA 通道

此外，还提供了中断服务，既可用作 DMA 的复用功能，也可用来指示外设的错误或关键状态。

SPDIFRX 还提供名为 **spdifrx_frame_sync** 的信号，该信号会在每次检测到子帧的报头时翻转。因此占空比为 50%，频率等于帧速率。

该信号可与定时器事件相连，以计算频率偏移。

图 391. SPDIFRX 框图



1. “n”固定为 4

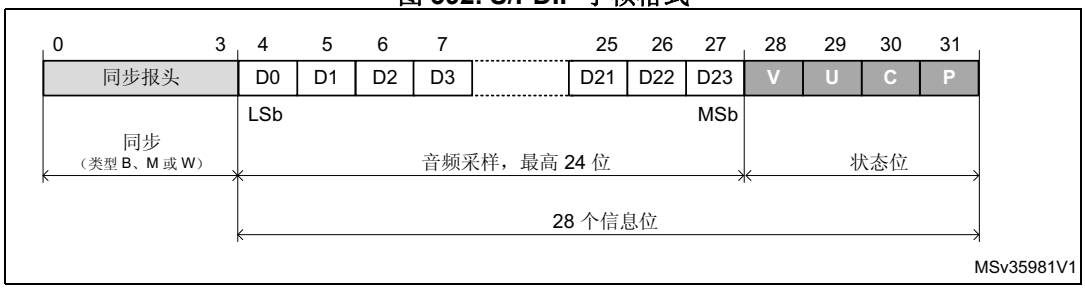
34.3.1 S/PDIF 协议 (IEC-60958)

S/PDIF 块

一个 S/PDIF 帧由两个子帧组成（见*图 393*）。每个子帧包含 32 位（或时隙）：

- 位 0 至位 3 包含同步报头之一
- 位 4 至位 27 包含以线性 2 的补码表示的音频采样字。最高有效位 (MSB) 为位 27。使用 20 位编码范围时，位 8 至位 27 包含音频采样字，其中位 8 为 LSB。
- 位 28（有效性位“V”）表示数据是否有效（例如转换为模拟数据）。
- 位 29（用户数据位“U”）包含用户数据信息，如光盘的音轨编号。
- 位 30（通道状态位“C”）包含通道状态信息，如采样率和复制保护。
- 位 31（奇偶检验位“P”）包含奇偶校验位，这样位 4 至位 31（含）将包含偶数个“1”和偶数个“0”（偶校验）。

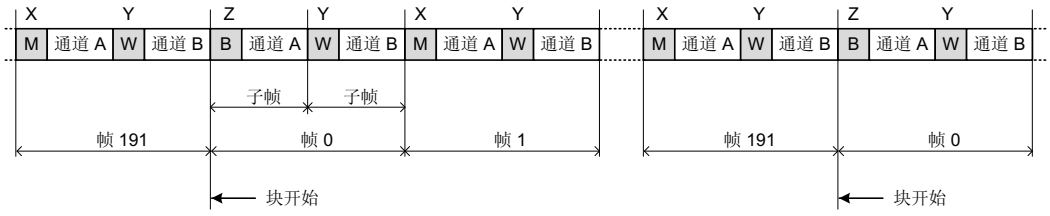
图 392. S/PDIF 子帧格式



对于线性编码音频应用，第一个子帧（立体声操作中的左声道或“A”通道以及单声道操作中的主通道）通常以报头“M”开始。但是，报头每 192 帧切换为报头“B”一次，以识别用于组织通道状态和用户信息的块结构的开始。第二个子帧（立体声操作中的右声道或“B”通道以及单声道操作中的辅助通道）始终以报头“W”开始。

一个 S/PDIF 块包含 192 对 32 位子帧。

图 393. S/PDIF 块格式



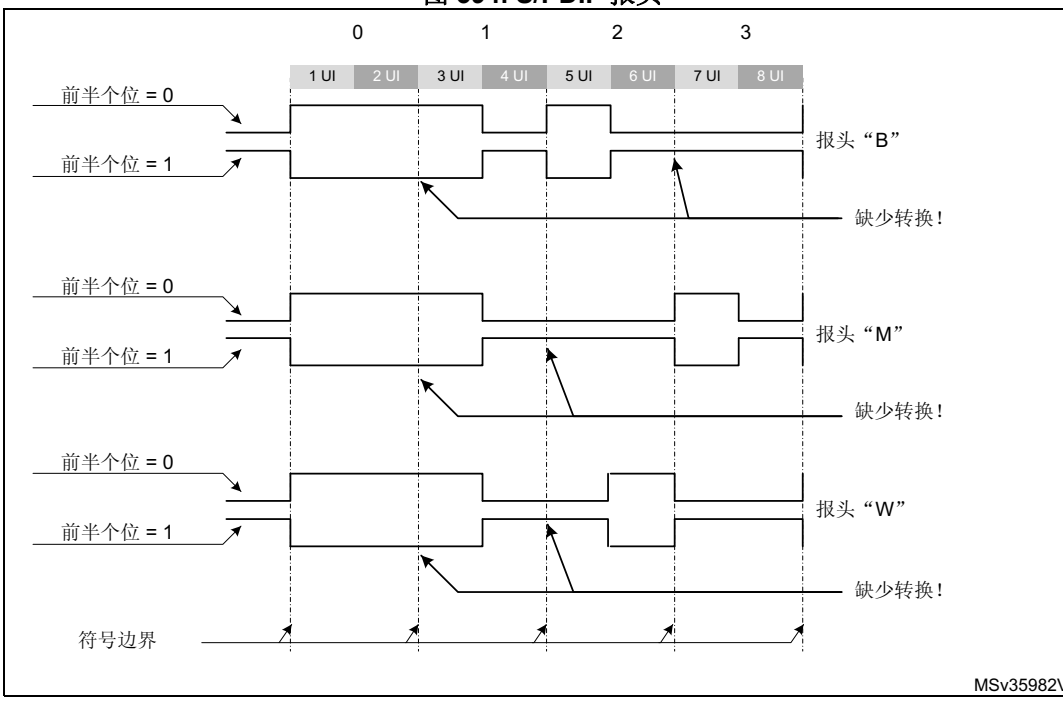
注
出于历史原因，报头“B”、“M”和“W”在专业应用中使用时分别称为“Z”、“X”以及“Y”。

MSv35923V1

同步报头

报头模式反转或者不符合前半位值。使能第一个帧的第一个“B”报头的传输前，此前半位值为线路的电平。对于其它报头，此前半位值为之前子帧的奇偶校验位的第二个半位。*图 394* 中介绍了报头模式 B、M 和 W。

图 394. S/PDIF 报头



MSv35982V1

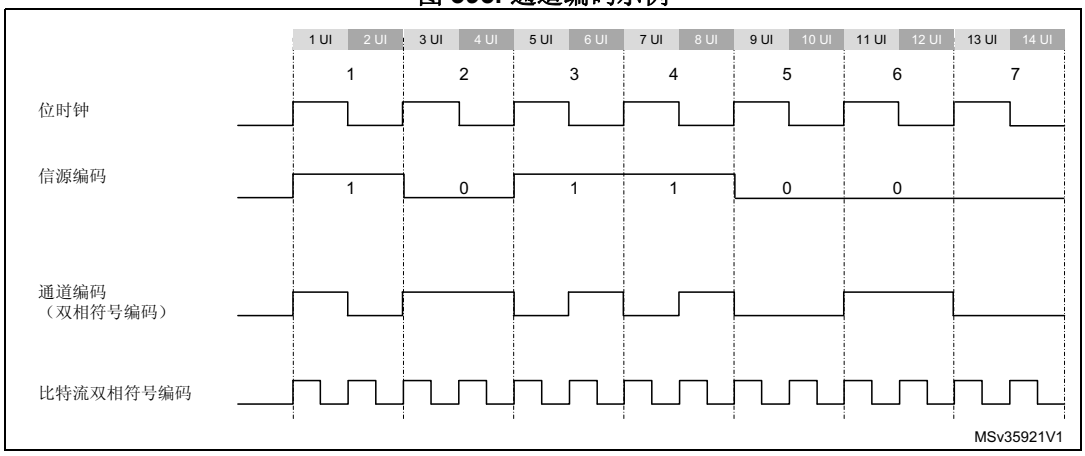
信息位编码

为最大程度减小传输线路上的直流分量值，并加快时钟从数据流中恢复的速度，位 4 至位 31 采用双相符号编码。

要传输的各个位通过由两个连续二进制状态构成的符号表示。符号的第一个状态始终不同于前一个符号的第二个状态。如果要传输的位为逻辑 0，则符号的第二个状态与第一个状态相同。但如果该位为逻辑 1，则两个状态不同。在 IEC-60958 规范中，这些状态称为“UI”（单位间隔）。

首先传输 24 个数据位中的 LSb。

图 395. 通道编码示例



34.3.2 SPDIFRX 解码器 (SPDIFRX_DC)

主要原则

SPDIFRX 用来解码 S/PDIF 数据流的技术以测量两个连续边沿之间时间间隔为基础。可在 S/PDIF 数据流中找到三类时间间隔：

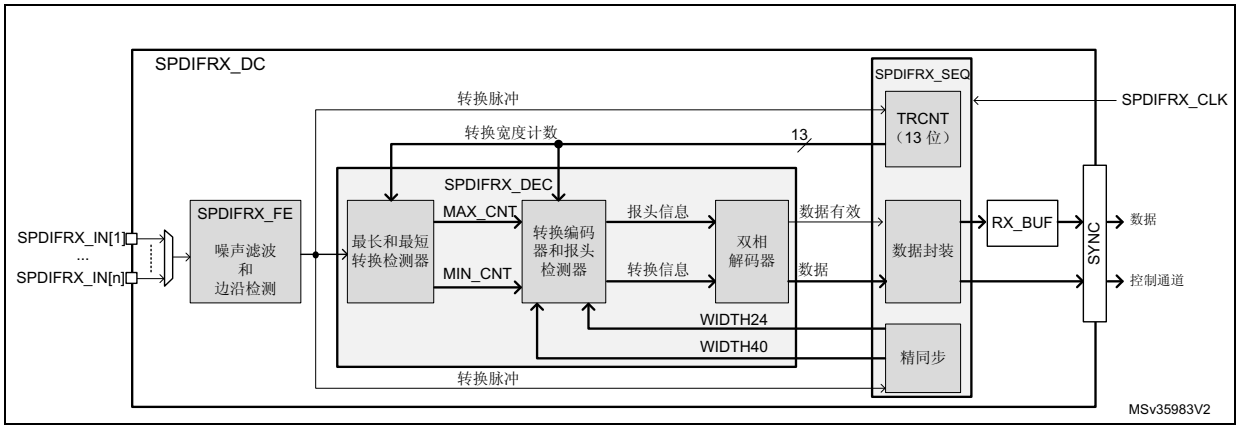
- 长时间间隔，持续 3 个 UI，称为 TL。仅在报头期间出现。
- 中时间间隔，持续 2 个 UI，称为 TM。在一些报头或信息域中出现。
- 短时间间隔，持续 1 个 UI，称为 TS。在一些报头或信息域中出现。

SPDIFRX_DC 模块负责解码接收到的 S/PDIF 数据流。它负责以下功能：

- 对传入信号进行重新采样和滤波
- 估算时间间隔
- 估算符号率和同步性
- 解码串行数据并验证完整性
- 检测块和子帧报头
- 连续跟踪符号率

图 396 详细说明了 SPDIFRX 解码器。

图 396. SPDIFRX_DC 解码器

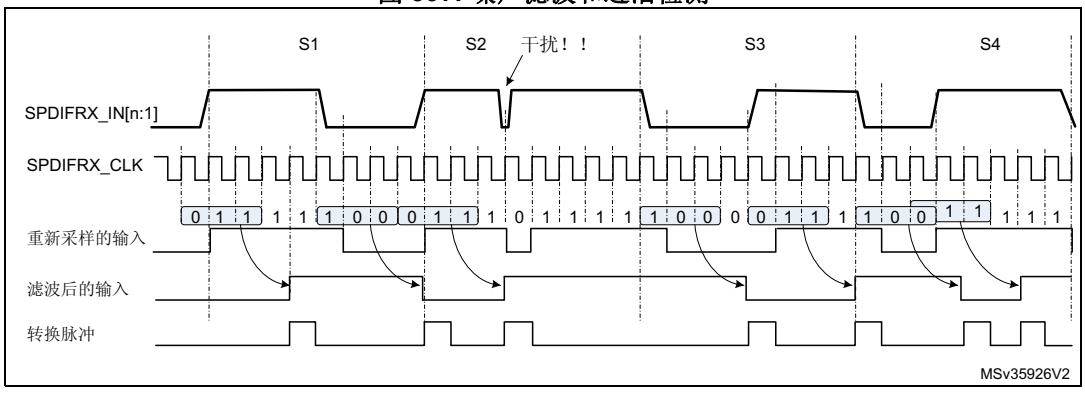


噪声滤波和上升/下降沿检测

使用 SPDIFRX_CLK 时钟（采集时钟）重新采样所选的 SPDIFRX_IN 上接收到的 S/PDIF 信号。采用简单滤波来去除杂波。通过检测边沿转换的阶段执行此过程。边沿转换的检测步骤如下：

- 当采样到序列 0 后跟两个 1 时，检测到上升沿。
- 当采样到序列 1 后跟两个 0 时，检测到下降沿。
- 在上升沿后，预期为下降沿序列。
- 在下降沿后，预期为上升沿序列。

图 397. 噪声滤波和边沿检测



最长和最短转换检测器

最长和最短转换检测器模块可检测两次转换间的最大 (MAX_CNT) 和最小 (MIN_CNT) 持续时间。TRCNT 计数器用于测量时间间隔的持续时间。它由 SPDIFRX_CLK 时钟信号驱动。对于每个转换脉冲，都会存储计数器值并复位计数器以重新开始计数。

最大持续时间通常出现在报头周期内。最大持续时间将作为 MAX_CNT 发出。最小持续时间将作为 MIN_CNT 发出。

当转换定时器过期时，最长和最短转换的搜索将停止。转换定时器类似于每转换 70 次传入信号就触发一次的看门狗定时器。请注意，计数 70 次转换可确保延迟略长于一个子帧。

请注意，当 TRCNT 因两个脉冲间的时间间隔过长而发生上溢时，SPDIFRX 将停止并且 SPDIFRX_SR 寄存器的标志 TERR 将置 1。

转换编码器和报头检测器

“转换编码器和报头检测器”块接收 MAX_CNT 和 MIN_CNT。它还接收 TRCNT 计数器中的当前转换宽度（请参见*图 396*）。该块通过将当前转换宽度与两个不同阈值（TH_{HI} 和 TH_{LO}）进行比较来编码当前转换宽度。

- 如果当前转换宽度小于 (TH_{LO} - 1)，则接收到的数据为数据位“1”的一半，并编码为 TS。
- 如果当前转换宽度大于 (TH_{LO} - 1) 且小于 TH_{HI}，则接收到的数据为数据位“0”，并编码为 TM。
- 如果当前转换宽度大于 TH_{HI}，则接收到的数据为报头的长脉冲，并编码为 TL。
- 否则将生成错误代码（FERR 标志置 1）。

使用两种不同方法来详细阐述阈值 TH_{HI} 和 TH_{LO}。

如果外设正在执行其初始同步（“粗同步”），则阈值的计算方法如下：

- TH_{LO} = MAX_CNT / 2。
- TH_{HI} = MIN_CNT + MAX_CNT / 2。

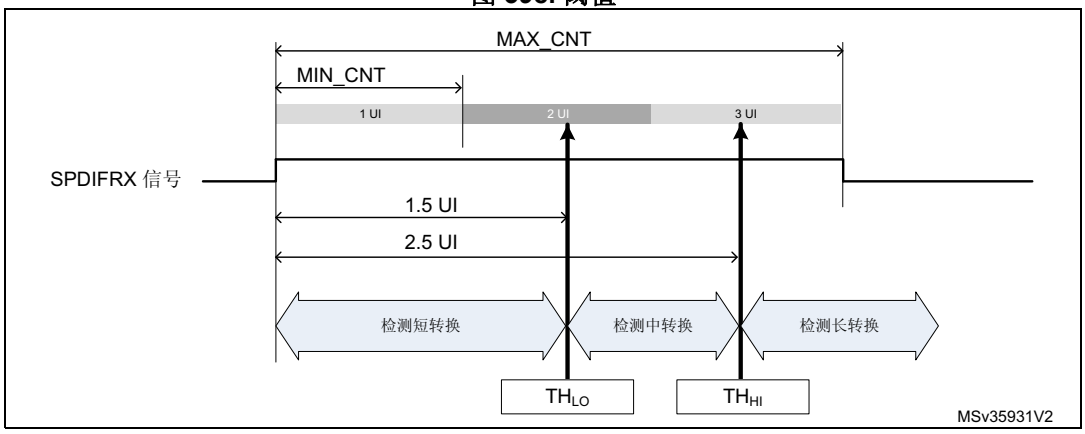
“粗同步”完成后，SPDIFRX 将使用更精确的参考来详细阐述阈值。SPDIFRX 将测量 24 个信号 (WIDTH24) 的长度来定义 TH_{LO}，测量 40 个信号 (WIDTH40) 的长度来定义 TH_{HI}。TH_{HI} 和 TH_{LO} 的计算方法如下：

- TH_{LO} = (WIDTH24) / 32
- TH_{HI} = (WIDTH40) / 32

第二个同步阶段称为“精同步”。更多相关信息，请参见*图 400*。

如后面的图所示，理想情况下 TH_{LO} 等于 1.5 个 UI，TH_{HI} 等于 2.5 个 UI。

图 398. 阈值



报头检测器检查特定序列的四个连续转换来确定它们是否构成报头的一部分。假设 TRANS0、TRANS1、TRANS2 和 TRANS3 表示四个连续的转换，编码方式如上面所述。*表 187* 给出了这四个转换的值以构成报头。缺少该模式表示这些转换构成子帧中数据的一部分并且双相解码器将对它们进行解码。

表 187. 报头的转换序列

报头类型	双相数据模式	TRANS3	TRANS2	TRANS1	TRANS0
报头 B	11101000	TL	TS	TS	TL
报头 M	11100010	TL	TL	TS	TS
报头 W	11100100	TL	TM	TS	TM

双相解码器

双相解码器使用由“转换编码器和报头检测器”块提供的转换信息来解码输入的双相符号数据流。首先，它等待报头检测信息。在报头检测后，它将解码以下转换信息：

- 如果传入的转换信息为 TM，则解码为“0”。
- 两个连续的 TS 解码为“1”。
- 任何其它转换序列将生成错误信号（FERR 置 1）。

通过这种方式解码 28 个数据位后，此模块将查找紧跟的报头数据。如果新报头与预期不同，则该模块将生成一个错误信号（FERR 置 1）。有关错误标志的更多信息，请参见*第 34.3.8 节：接收错误*。

数据封装

该模块负责解码 IEC-60958 帧和块。它还处理向 RX_BUF 或 SPDIFRX_CSR 寄存器写入数据。

34.3.3 SPDIFRX 对时钟偏差的容差

SPDIFRX 对时钟偏差的容差取决于一个位时隙中的采样时钟周期数。SPDIFRX_CLK 越快，接收越稳定。SPDIFRX_CLK 频率和信号率间的比率必须至少为 11。

有两种现象（至少！）会降低接收质量：

- 反映两个连续转换之间长度差异的周期间抖动。
- 反映周期间抖动积累效果的长期抖动。它可被看作低频符号调制。

34.3.4 SPDIFRX 同步

将 SPDIFRXEN 设置为 0b01 或 0b11 时，同步阶段开始。*图 399* 给出了同步过程。

如果 SPDIFRX_CR 寄存器的 WFA 位置 1，则在开始同步过程前，外设必须首先检测所选 SPDIFRX_IN 线路上的活动。通过检测所选 SPDIFRX_IN 上的四个转换来执行活动检测。外设保持该状态，直到未检测到转换。因为仅当所选 SPDIFRX_IN 输入上出现活动时，IP 才切换为粗同步模式，因此该功能在避免同步错误时特别有用。更多信息，请参见*第 34.4 节：编程步骤*。
用户仍然可以通过将 SPDIFRXEN 设置为 0 来使 SPDIFRX 进入 STATE_IDLE。如果 WFA 设置为 0，外设将直接启动粗同步而不检查活动。

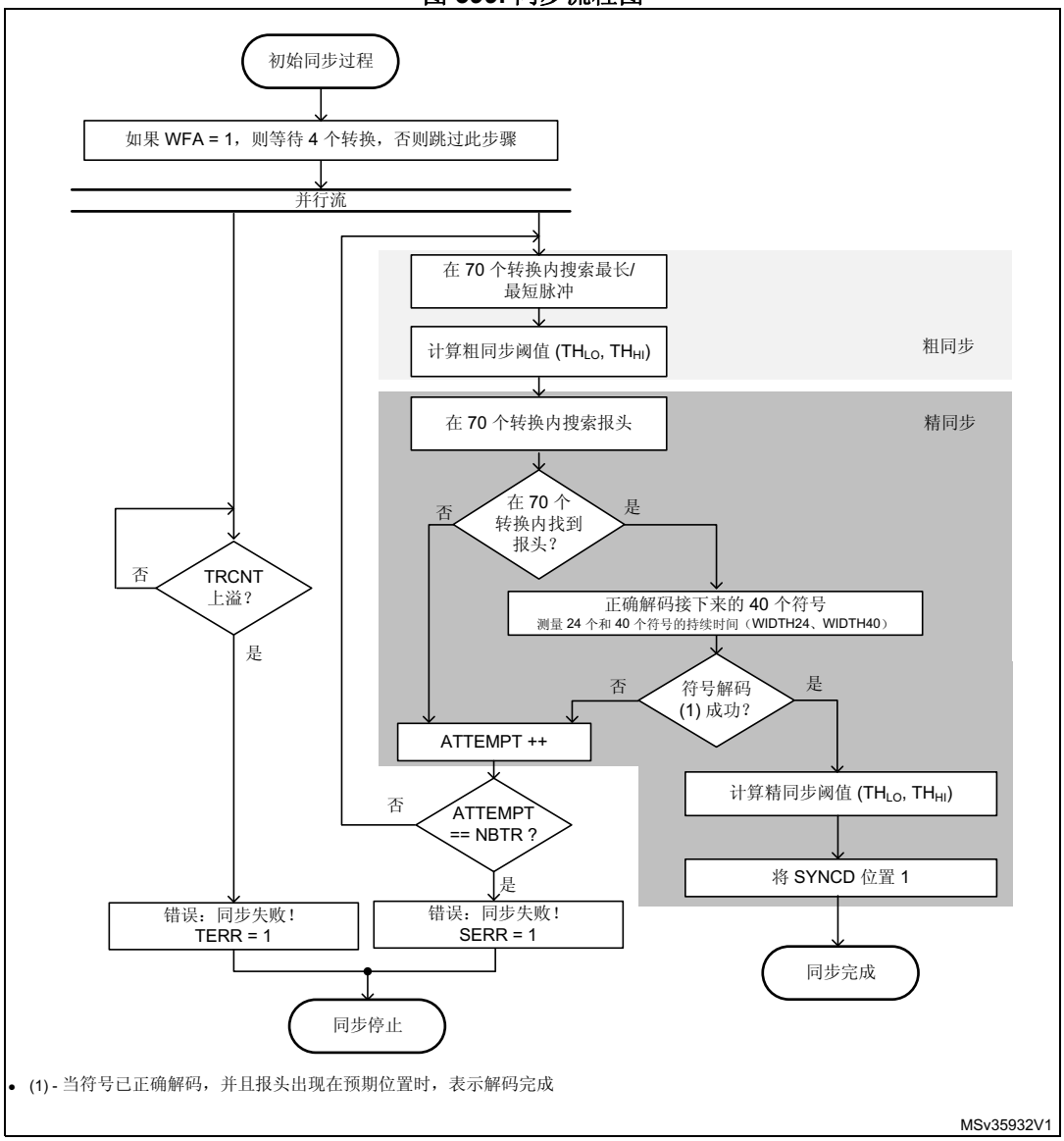
下一步将初步估算阈值（粗同步），以便执行精细同步（精同步）。由于 SPDIFRX 线路上的干扰，第一次可能无法正确执行该过程。为此，用户可在将 SERR 错误标志置 1 前编程允许的重试次数 (NBTR)。
当 SPDIFRX 能够正确测量 24 个和 40 个连续符号的持续时间并完成精同步时，阈值将更新并且标志 SYNCD 置 1。有关更多信息，请参见*转换编码器和报头检测器*一节。

可以检测到两类错误：

- TRCNT 的上溢，这通常意味着输入线路上没有有效的 S/PDIF 数据流。该上溢由 TERR 标志指示。
- 重试次数达到了设定的值。这意味着 S/PDIF 信号上存在较强的抖动。该错误由 SERR 标志指示。

完成第一次精同步后，如果检测到下一个“B”报头，将开始接收通道状态 (C) 和用户数据 (U)（请参见*图 403*）。随后，用户可通过 SPDIFRX_CSR 寄存器读取 IEC-60958 C 和 U 位。根据此信息，用户可以为 DRFMT 和 RXSTEO 选择合适的设置。例如，如果用户检测到当前音频流传输编码数据，则可在开始数据接收之前将 RXSTEO 设置为 0，并将 DRFMT 设置为 0b10。请注意，当 SPDIFRXEN = 0b11 时，不能修改 DRFMT 和 RXSTEO。如果 SPDIFRXEN 已经设置为 0b11，则忽略对这些字段的写操作，尽管可使用导致 SPDIFRXEN 变为 0b11 的相同写指令来修改这些字段。
然后在开始保存音频采样前，SPDIFRX 将等待 SPDIFRXEN = 0b11 和“B”报头。

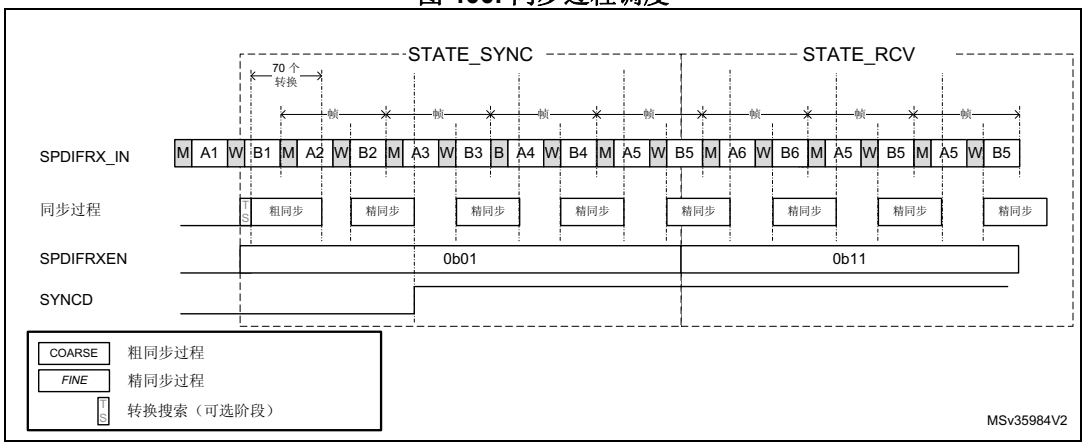
图 399. 同步流程图



有关 TRCNT 上溢的更多信息，请参见*帧结构和同步错误*。

如*图 400* 中所示，每一帧都将重新触发精同步过程来更新阈值，以便连续跟踪 S/PDIF 同步。

图 400. 同步过程调度



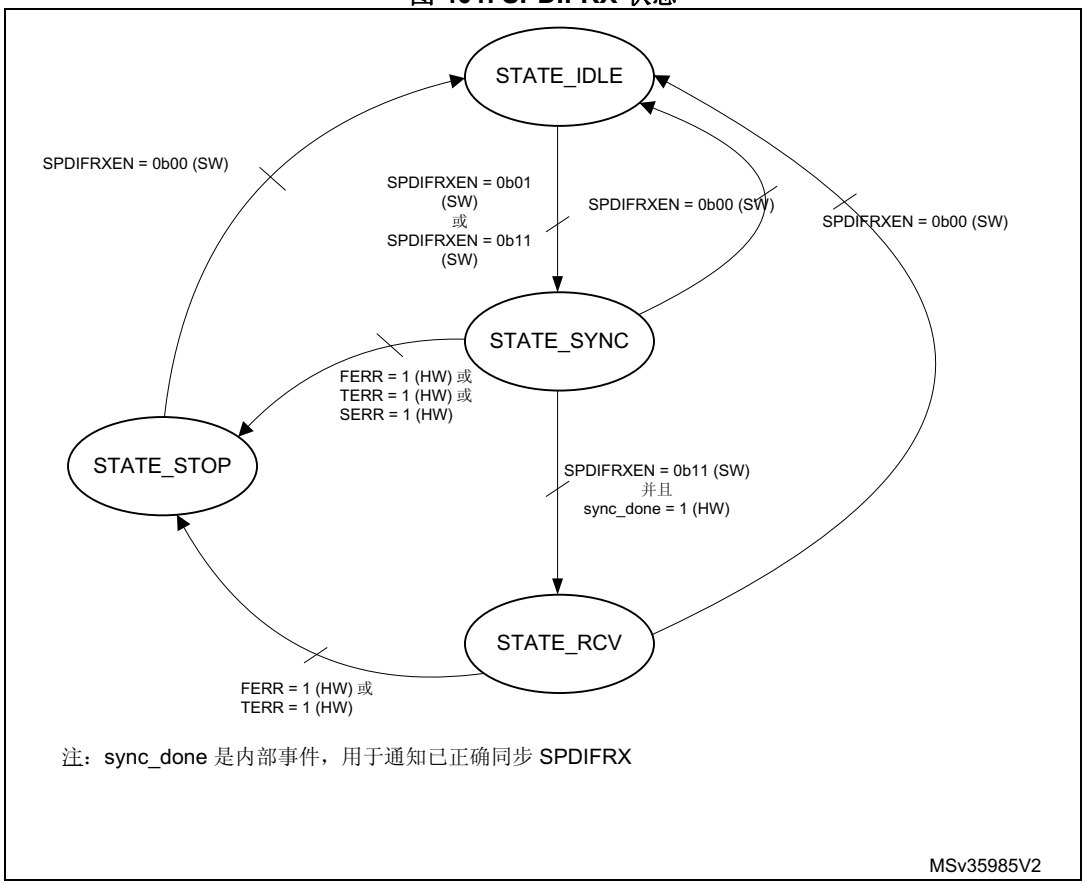
34.3.5 SPDIFRX 处理

软件可以通过 SPDIFRXEN 字段控制 SPDIFRX 的状态。SPDIFRX 可进入以下状态之一：

- STATE_IDLE：
  禁止外设，SPDIFRX_CLK 域复位。PCLK1 域功能正常。
- STATE_SYNC：
  外设与数据流同步，阈值定期更新，可通过 DMA 的中断读取用户和通道状态。音频采样未提供给接收缓冲区。
- STATE_RCV：
  外设与数据流同步，阈值定期更新，可通过中断或 DMA 通道读取用户、通道状态和音频采样。当 SPDIFRXEN 变为 0b11 时，SPDIFRX 将在开始保存音频采样前一直等待“B”报头。
- STOP_STATE：
  外设不再同步，用户、通道状态和音频采样的接收将停止。预期软件将重启 SPDIFRX。

图 401 给出了 SPDIFRX 可能的状态以及如何从一个状态转换为其它状态。受软件控制的位后跟说明“(SW)”，受 IP 控制的位后跟说明“(HW)”。

图 401. SPDIFRX 状态



当 SPDIFRX 处于 STATE_IDLE 时：

– 通过将 SPDIFRXEN 设置为 0b01 或 0b11，软件可转换为 STATE_SYNC。

当 SPDIFRX 处于 STATE_SYNC 时：

– 如果同步失败或者接收的数据未正确解码且无法在不进行再同步的情况下恢复（FERR 或 SERR 或 TERR = 1），则 SPDIFRX 将进入 STATE_STOP 状态并且等待软件应答。

– 当同步阶段完成时，如果 SPDIFRXEN = 0b01，外设将保持该状态。

– 软件可随时将 SPDIFRXEN 设置为 0，随后 SPDIFRX 立刻返回至 STATE_IDLE。如果 DMA 传输正在进行，则将正确完成。

– 如果 SPDIFRXEN = 0b11 且 SYNCD = 1，则 SPDIFRX 进入 STATE_RCV。

当 SPDIFRX 处于 STATE_RCV 时：

– 如果接收的数据未正确解码且无法在不进行再同步的情况下恢复（FERR 或 SERR 或 TERR = 1），则 SPDIFRX 将进入 STATE_STOP 状态并且等待软件应答。

– 软件可随时将 SPDIFRXEN 设置为 0，随后 SPDIFRX 立刻返回至 STATE_IDLE。如果 DMA 传输正在进行，则将正确完成。

当 SPDIFRX 处于 STATE_STOP 时：

– SPDIFRX 停止接收和同步，并等待软件将 SPDIFRXEN 位设置为 0，以清零错误标志。

当 SPDIFRXEN 设置为 0 时，IP 被禁止，这意味着所有状态机被复位，并且 RX_BUF 被刷新。还要注意，标志 FERR、SERR 和 TERR 也会复位。

34.3.6 数据接收管理

SPDIFRX 为音频采样接收提供了双缓冲区。32 位缓冲区位于 SPDIFRX_CLK 时钟域 (RX_BUF) 和 SPDIFRX_DR 寄存器中。如果 SPDIFRX_DR 为空，则包含在 RX_BUF 中的有效数据将立刻传输至 SPDIFRX_DR。

当满足以下两个条件时，包含在 RX_BUF 中的有效数据将传输至 SPDIFRX_DR：

– 检测到奇偶校验位 (P) 和下一个报头之间的转换（这表示字已经完全接收到）。

– SPDIFRX_DR 为空。

双字缓冲区为延迟限制提供了更高的灵活性。

允许的最大延迟为 $T_{SAMPLE} - 2T_{PCLK} - 2T_{SPDIFRX_CLK}$

其中，T_{SAMPLE} 是接收的立体声音频采样的音频采样率，T_{PCLK} 是 PCLK1 时钟的周期，$T_{SPDIFRX_CLK}$ 是 SPDIFRX_CLK 时钟的周期。

SPDIFRX 可以使用 DMA (spdifrx_dma_req/clr_d) 或中断将音频采样传输到存储器中。建议选择 DMA，有关更多信息，请参见*第 34.3.10 节：DMA 接口*。

SPDIFRX 提供多种处理接收数据的方法。用户可以分别处理控制信息流和音频采样流，或将二者一起处理。

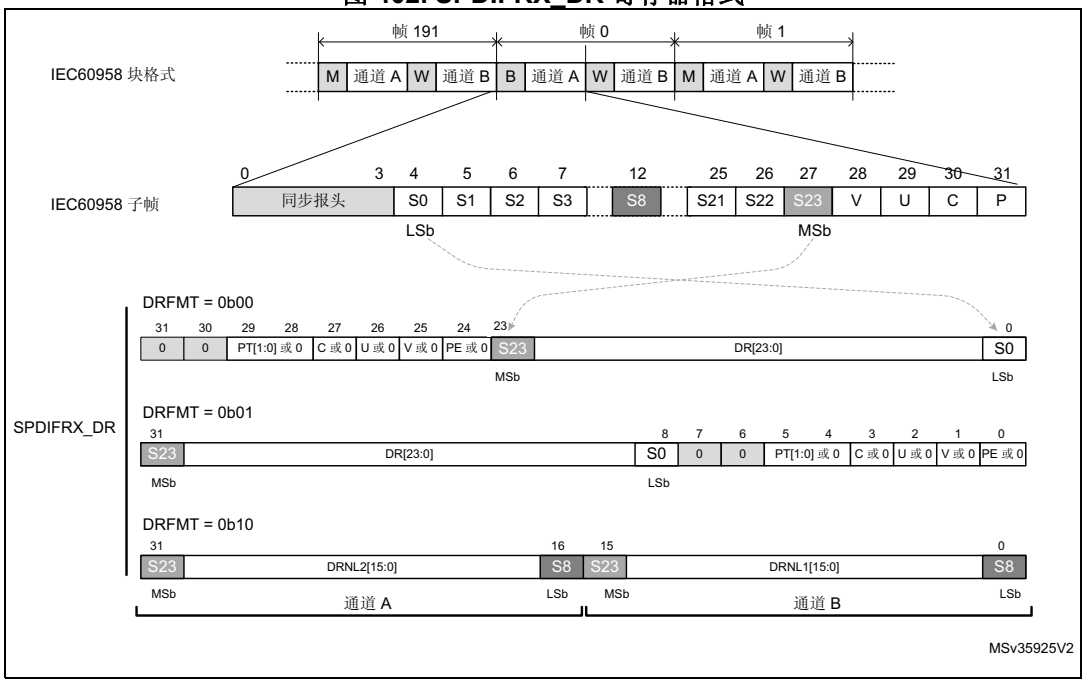
对于各子帧，数据接收寄存器 SPDIFRX_DR 包含 24 个数据位和可选的 V、U、C、PE 状态位以及 PT（请参见*混合数据和控制流*）。

请注意，PE 位表示奇偶校验错误位，该位将在解码的子帧中检测到奇偶校验错误时置 1。PT 字段包含报头类型（B、M 或 W）。

V、U 和 C 是从 S/PDIF 接口接收的值的直接副本。

DRFMT 位允许选择*图 402* 中所示的 3 种音频格式。

图 402. SPDIFRX_DR 寄存器格式



将 DRFMT 设置为 0b00 或 0b01，可以使数据在 SPDIFRX_DR 寄存器中左对齐或右对齐。可以根据软件所需的处理方式启用状态信息或将其强制设置为零。

通过 DRFMT = 0b10 得出的格式在非线性模式下相关，因为每个子帧仅使用 16 位。使用此格式，两个连续子帧的数据存储到 SPDIFRX_DR 中，存储器占用量将除以二。请注意，当 RXSTEO = 1 时，不存在偏离风险（即，来自通道 A 的数据始终存储在 SPDIFRX_DR[31:16] 中）。如果 RXSTEO = 0，则在上溢情况下存在偏离风险。此时，SPDIFRX_DR[31:16] 始终包含最早的值，SPDIFRX_DR[15:0] 则包含较新的值（请参见*图 404*）。

在此格式中，状态信息不能与数据混合，但用户仍可通过 SPDIFRX_CSR 寄存器获取这些数据，并使用专用的 DMA 通道或中断将它们传输到存储器中（请参见*第 34.3.7 节：专用控制流*）

混合数据和控制流

用户可以选择使用此模式，以完全灵活地处理控制流。用户可以选择应保存到数据寄存器 (SPDIFRX_DR) 中的字段。

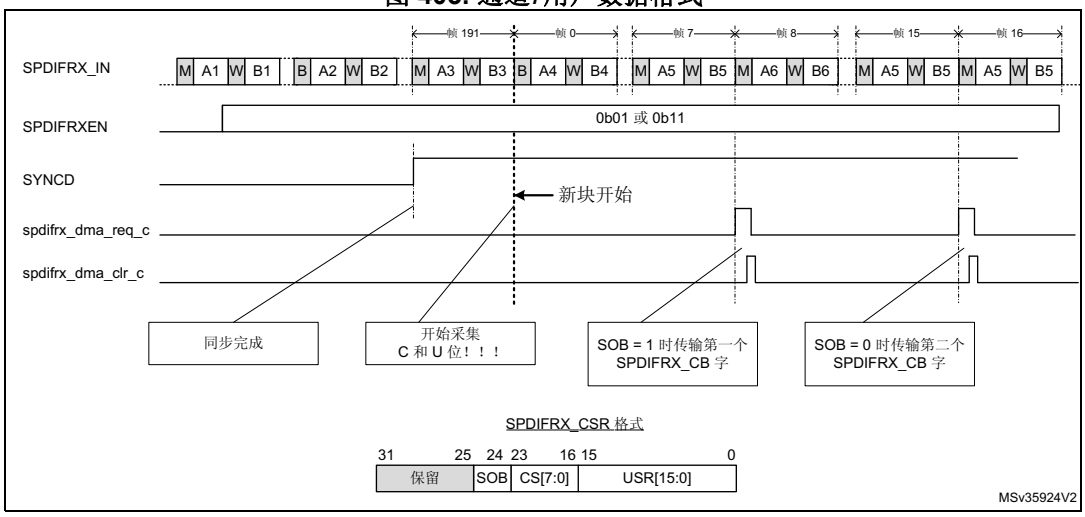
- 当位 PMSK = 1 时，将屏蔽奇偶校验错误信息（设置为 0），否则将其复制到 SPDIFRX_DR 中。
- 当位 VMSK = 1 时，将屏蔽有效性信息（设置为 0），否则将其复制到 SPDIFRX_DR 中。
- 当位 CUMSK = 1 时，将屏蔽通道状态和使用的数据信息（设置为 0），否则将其复制到 SPDIFRX_DR 中。
- 当位 PTMSK = 1 时，将屏蔽报头类型信息（设置为 0），否则将其复制到 SPDIFRX_DR 中。

34.3.7 专用控制流

SPDIFRX 可以通过专用 DMA 通道捕获用户数据和通道状态信息。该功能允许 SPDIFRX 连续采集通道状态和用户信息。采集将在 IEC 60958 块开始时启动。可使用两个字段控制该路径：CBDMAEN 和 SPDIFRXEN。当 SPDIFRXEN 设置为 0b01 或 0x11 时，采集将在同步阶段完成后开始。当接收到 8 个通道状态和 16 个用户数据位时，会将它们封装并存储到 SPDIFRX_CSR 寄存器中。如果 CBDMAEN 位置 1，则触发 DMA 请求（请参见*图 403*）。

如果 CS[0] 对应新块的第一位，则 SOB 位将置 1。请参见*第 34.5.8 节：通道状态寄存器 (SPDIFRX_SR)*。可使用 CHSEL 位选择用户想要从通道 A 还是通道 B 选择通道状态信息 (C)。

图 403. 通道/用户数据格式



注： *检测到第一个块开始（B 报头）后，SPDIFRX 将每 8 个帧检测一次报头类型。*

注： *SPDIFRX_DR 寄存器中的上溢错误不影响该路径。*

34.3.8 接收错误

帧结构和同步错误

发生下列情况之一时，SPDIFRX 将检测到错误：

- 出现以下情况时，FERR 位置 1：
  - 对于 28 个信息位中的每个位，如果一个符号转换序列不正确：例如短脉冲未按对分组。
  - 如果报头出现在异常位置，或未接收到预期的报头。
- 当同步失败时，SERR 位置 1，原因是重试次数超过了设定的值。
- 当用于估算两次传输的时间间隔的计数器 (TRCNT) 产生上溢时，TERR 位置 1。在 8192 个 SPDIFRX_CLK 时钟周期内未检测到转换时，将发生上溢。这表示最大时间间隔为 11.6 帧。

当这些标志之一变为 1 时，将忽略所选 SPDIFRX_IN 上的通信，并将在 SPDIFRX_CR 寄存器的 IFEIE 位置 1 时生成中断。

发生这些错误之一时的正常步骤为：

– 将 SPDIFRXEN 设置为 0 以清零错误标志
– 将 SPDIFRXEN 设置为 0b01 或 0b11 以重新启动 IP

有关更多信息，请参见*图 401*。

奇偶校验错误

对于各子帧，28 个信息位中预期包含偶数个 0 和 1。否则，SPDIFRX_SR 寄存器的 PERR 位将被置位，并且如果 SPDIFRX_CR 寄存器中的奇偶校验中断使能 PERRIE 位置 1，则将生成中断。不会暂停输入数据的接收，并且即使中断仍处于挂起状态，SPDIFRX 也会继续将数据传输到 SPDIFRX_DR 中。

通过 PERRCF 位清零 PERR 标志来确认中断。

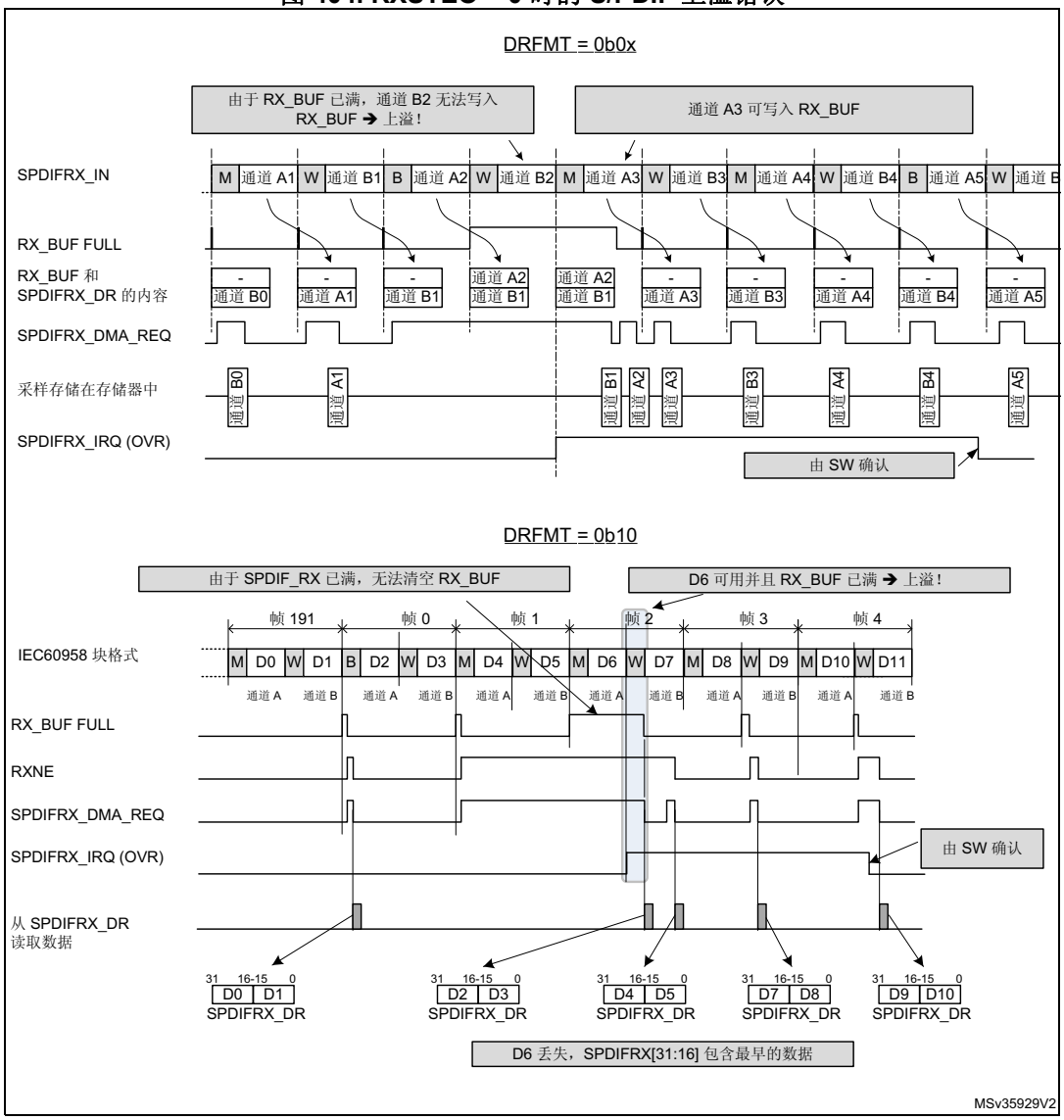
如果软件要确保 SPDIFRX_DR 寄存器中读取的数据与 PERR 位的值保持一致，必须将 PMSK 位设置为 0。

上溢错误

如果 SPDIFRX_DR 和 RX_BUF 均已满，而 SPDIFRX_DC 又需要向 RX_BUF 中写入新采样，则将丢弃这一新采样，并触发上溢条件。SPDIFRX_SR 寄存器中的上溢错误标志 OVR 置 1，并且如果 SPDIFRX_CR 寄存器的 OVRIE 位置 1，则将生成中断。

如果 RXSTEO 位设置为 0，则只要 RX_BUF 为空，IP 便会存储接下来传入的数据，即使 OVR 标志仍然处于挂起状态。主要目的是尽可能地减少丢失的采样数。请注意，无论 DRFMT 的值为何，这种行为都是类似的。请参见*图 404*。

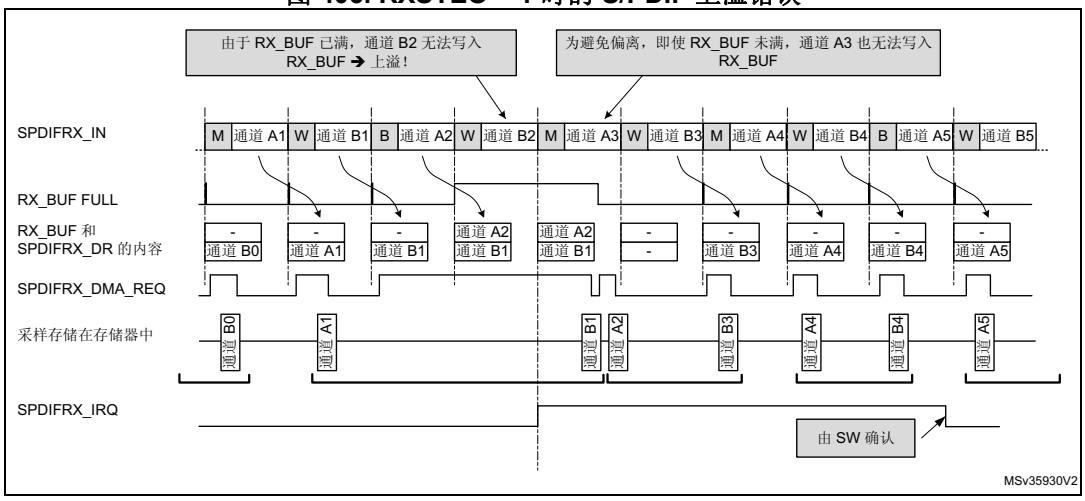
图 404. RXSTEO = 0 时的 S/PDIF 上溢错误



如果 RXSTEO 位置 1，则意味着传输的是立体声数据，此时 SPDIFRX 必须避免左右通道之间的偏离。因此，即使在 RX_BUF 内仍有空间，外设也必须丢弃一秒的采样以避免发生偏离。随后，传入的采样可正常写入到 RX_BUF 中，即使 OVR 标志仍处于挂起状态。请参见 *图 405*。

在 OVRCF 位置 1 时，OVR 标志由软件清零。

图 405. RXSTEO = 1 时的 S/PDIF 上溢错误



34.3.9 时钟策略

SPDIFRX 块需要两个不同的时钟：

- APB1 时钟 (PCLK1)，用于寄存器接口，
- SPDIFRX_CLK，主要由 SPDIFRX_DC 部分使用。这些时钟不应锁相，因此需要重新同步这些时钟域中的所有信号（*图 391* 中的 SYNC 块）。

为正确解码传入的 S/PDIF 数据流，SPDIFRX_DC 应以至少比最大符号率高 11 倍或者比音频采样率高 704 倍的时钟重新采样接收的数据。例如，如果用户预期接收到最高 12.288 MHz 的符号率，则采样率至少为 135.2 MHz。SPDIFRX_DC 使用的时钟为 SPDIFRX_CLK。

PCLK1 的频率必须至少等于符号率。

表 188. SPDIFRX_CLK 的最小频率与音频采样率

符号率	SPDIFRX_CLK 的最小频率	注释
3.072 MHz	33.8 MHz	用于 48 kHz 数据流
6.144 MHz	67.6 MHz	用于 96 kHz 数据流
12.288 MHz	135.2 MHz	用于 192 kHz 数据流

34.3.10 DMA 接口

SPDIFRX 能够使用 DMA 执行通信。

注： *有关 DMA 控制器可用性的信息，请参见产品规范。*

SPDIFRX 提供两个独立的 DMA 通道：

- 一个专门用于数据传输的 DMA 通道
- 一个专用于传输通道状态和用户信息的 DMA 通道

可通过将 SPDIFRX_CR 寄存器中的 RXDMAEN 位置 1 来启用数据的 DMA 模式。这种情况下，只要 SPDIFRX_DR 不为空，SPDIFRX 接口便会将传输请求发送给 DMA。DMA 通过 SPDIFRX_DR 寄存器读取接收的数据，无需 CPU 干预。

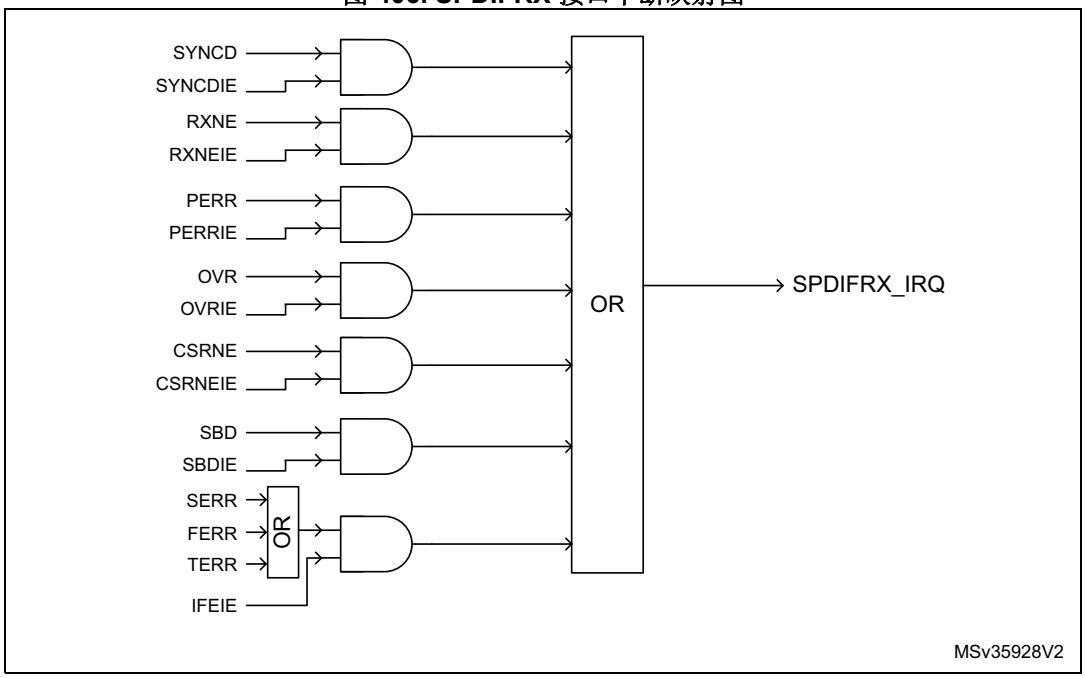
有关控制数据的 DMA 的用法，请参见*第 34.3.7 节：专用控制流*。

34.3.11 中断生成

下列各项共享一条中断线：

- 数据流的接收事件 (RXNE)
- 控制流的接收事件 (CSRNE)
- 数据损坏检测 (PERR)
- 传输流中断 (OVR)
- 帧结构和同步错误（SERR、TERR 和 FERR）
- 新块中段的开始 (SBD)
- 同步完成 (SYNCD)

图 406. SPDIFRX 接口中断映射图



清零中断源

- 读取 SPDIFRX_DR 寄存器时，将清零 RXNE。
- 读取 SPDIFRX_CSR 寄存器时，将清零 CSRNE。
- 当 SPDIFRXEN 被设置为 0 时，将清零 FERR。
- 当 SPDIFRXEN 被设置为 0 时，将清零 SERR。
- 当 SPDIFRXEN 被设置为 0 时，将清零 TERR。
- 其它中断源通过 SPDIFRX_IFCR 寄存器清零。

注： *仅当 SPDIFRX 与输入数据流同步时 (SYNCD = 1)，才会发生 SBD 事件。*

当包含 B 报头的子帧因上溢而丢失时，将无法保证 SBD 标志行为。

34.3.12 寄存器保护

SPDIFRX 块内嵌某种硬件保护功能以避免错误使用控制寄存器。下面的表格根据 SPDIFRX 的状态说明了位字段属性。

表 189. 位字段属性与 SPDIFRX 状态

寄存器	字段	SPDIFRXEN		
		0b00 (STATE_IDLE)	0b01 (STATE_SYNC)	0b11 (STATE_RCV)
SPDIFRX_CR	INSEL	rw	r	r
	WFA	rw	r	r
	NBTR	rw	r	r
	CHSEL	rw	r	r
	CBDMAEN	rw	rw	rw
	PTMSK	rw	rw	rw
	CUMSK	rw	rw	rw
	VMSK	rw	rw	rw
	PMSK	rw	rw	rw
	DRFMT	rw	rw	r
	RXSTEO	rw	rw	r
	RXDMAEN	rw	rw	rw
SPDIFRX_IMR	所有字段	rw	rw	rw

此表格明确说明了 INSEL 等字段必须在 IP 处于 STATE_IDLE 时进行编程。在其它 IP 状态下，硬件将阻止写入该字段。

注： *即使硬件允许“实时”写入 CBDMAEN 和 RXDMAEN，也不推荐在 IP 接收数据时启用 DMA。*

注： *请注意，各屏蔽位（PMSK、VMSK...）可以在任意 IP 状态下进行“实时”更改，但任何更改都不影响已经保存到 SPDIFRX_DR 中的数据。*

34.4 编程步骤

下列示例将说明 SPDIFRX 块的完整激活顺序。数据路径以及通道状态和用户信息都将使用专用的 DMA 通道。激活顺序随即分为下面几个步骤：

- 等待所选 SPDIFRX_IN 输入上的有效数据
- 与 S/PDIF 数据流同步
- 读取通道状态和用户数据以建立完整的音频路径
- 开始采集数据

一种检查有效数据是否可以进入 SPDIFRX_IN 线路的简单方法是通过将 WFA 位置 1 将 SPDIFRX 切换为 STATE_SYNC。后面将重点说明这种情况。不过，也可以按如下方式实现此功能：

- 软件必须不时检查（例如，每隔 100 ms）SPDIFRX 是否可找到同步。可以通过检查 TERR 位是否置 1 来执行此操作。当该位置 1 时，表示没有发现活动。
- 将 SPDIFRX_IN 输入连接到外部中断事件块以便检测 SPDIFRX_IN 线路的转换。当检测到活动时，可将 SPDIFRXEN 设置为 0b01 或 0b11。

  对于这两种实现，WFA 位设置为 0。

34.4.1 初始化阶段

初始化功能具有如下形式：

- 为音频采样和 IEC60958 通道状态和用户信息配置 DMA 传输（DMA 通道选择和活动、优先级、要传输的数据数量、循环/非循环模式和 DMA 中断）
- 配置目标地址：
  - 将 SPDIFRX_CSR 寄存器的地址配置为 IEC60958 通道状态和用户信息的源地址
  - 将 SPDIFRX_DR 寄存器的地址配置为音频采样的源地址
- 使能 SPDIFRX_CLK 的生成。要定义最小时钟频率与支持的音频采样率，请参见 *表 188*。
  请注意，无法提前获知已接收数据流的音频采样率。这意味着用户必须选择至少比应用应处理的最大音频采样率高 704 倍的 SPDIFRX_CLK 频率：例如，如果应用能够处理采样率最大为 96 kHz 的数据流，则 $F_{SPDIFRX_CLK}$ 至少应为
  704 x 96 kHz = 67.6 MHz
- 使能错误和事件信号的中断（IFEIE = SYNCDIE = OVRIE，PERRIE = 1，其它位设置为 0）。注意，SYNCDIE 可设置为 0。

- 配置 SPDIFRX_CR 寄存器：
  - INSEL 将选择所需输入
  - NBTR = 2，WFA = 1（允许重试 16 次，等待活动，直至进入同步阶段）
  - PTMSK = CUMSK = 1（报头，C 和 U 位不与数据混合）
  - VMSK = PMSK = 0（奇偶校验错误和有效性位与数据混合）
  - CHSEL = 0 （从子帧 A 读取通道状态）
  - DRFMT = 0b01（数据左对齐）
  - RXSTEO = 1（预期立体声模式为线性）
  - CBDMAEN = RXDMAEN = 1（使能 DMA 通道）
  - SPDIFRXEN = 0b01（将 SPDIFRX 切换为 STATE_SYNC）
- CPU 可以进入 WFI 模式

随后 CPU 将收到来自 DMA 或 SPDIFRX 的中断。

34.4.2 处理来自 SPDIFRX 的中断

当收到来自 SPDIFRX 的中断时，软件必须通过读取 SPDIFRX_SR 寄存器来确认中断源。

- 如果 SYNCD 置 1，则意味着同步已正常完成。此示例中无需执行任何操作，因为 DMA 已进行编程。软件仅需要等待 DMA 中断以读取通道状态信息。
  必须通过将 SPDIFRX_IFCR 寄存器的 SYNCDCF 位置 1 来清零 SYNCD 标志。
- 如果 TERR 或 SERR 或 FERR 设置为 1，则软件必须将 SPDIFRXEN 设置为 0，并从初始化阶段重新启动。
  - TERR 表示在同步阶段期间或之后发生超时。
  - SERR 表示因为达到允许的最大重试次数而同步失败。
  - FERR 表示同失败步后读取信息（异常报头、错误数据解码...）。
- 如果 PERR 置 1，则意味着已检测到奇偶校验错误，从而表明接收到的音频采样、通道状态或用户数据位中的某一位损坏。这里采取的操作取决于应用：一个操作是丢弃当前通道状态块，原因是其不可靠。无需从初始化阶段重新启动，因为同步不会丢失。
  必须通过将 SPDIFRX_IFCR 寄存器的 PERRCF 位置 1 来清零 PERR 标志。

34.4.3 处理来自 DMA 的中断

如果中断来自通道状态所用的 DMA 通道 (SPDIFRX_CSR)：

如果未发生错误（即 PERR），则 CPU 可以开始解码通道信息。例如，通道状态的位 1 通知用户当前数据流是否为线性。该信息对于建立正常处理链十分重要。同样，通道状态的位 24 至位 27 给出了传入数据流的采样频率。

借助这些信息，用户可在启动数据接收之前配置 RXSTEO 位和 DRFMT 字段。例如，如果当前数据流为非线性 PCM，则 RXSTEO 将被设置为 0，而 DRFMT 将被设置为 0b10。随后用户可通过将 SPDIFRXEN 设置为 0b11 来使能数据接收。

SOB 位置 1 时表示新块开始。该信息有助于软件识别通道状态的位 0。请注意，如果每次将 24 个值传输到存储器 DMA 便生成一个中断，则第一个字将始终对应新块的开始位置。

如果中断来自音频采样所用的 DMA 通道 (SPDIFRX_DR)：

这里执行的过程取决于数据类型（线性或非线性）以及所选的数据格式。

例如，在线性模式中，如果 PE 或 V 位置 1，则将在本地执行一个特殊过程以避免输出上出现杂波。在非线性模式下，这些位并不重要，因此数据帧具有各自的校验和。

34.5 SPDIFRX 接口寄存器

34.5.1 控制寄存器 (SPDIFRX_CR)

Control register

偏移地址：0x00

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INSEL[1]		
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	WFA[1]	NBTR[1:0][1]		CHSEL[1]	CBDMAEN[1]	PTMSK[1]	CUMSK[1]	VMSK[1]	PMSK[1]	DRFMT[1]		RXSTEO[1]	RXDMAEN[1]	SPDIFRXEN[1:0][1]	
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

1. 有关字段属性的更多信息，请参见*第 34.3.12 节：寄存器保护*。

位 31:19 保留，由硬件强制清零。

位 18:16 **INSEL**：SPDIFRX 输入选择 (SPDIFRX input selection)
0b000：选择 SPDIFRX_IN1
0b001：选择 SPDIFRX_IN2
0b010：选择 SPDIFRX_IN3
0b011：选择 SPDIFRX_IN4
其它保留

位 15 保留，由硬件强制清零。

位 14 **WFA**：等待活动 (Wait For Activity)
该位由软件置 1/复位。
1：执行同步前，SPDIFRX 等待 SPDIFRX_IN 线路上的活动（4 次转换）
0：执行同步前，SPDIFRX 不等待 SPDIFRX_IN 线路上的活动

位 13:12 **NBTR**：同步阶段期间允许的最大重试次数 (Maximum allowed re-tries during synchronization phase)
0b00：不允许重试（仅尝试一次）
0b01：允许重试 3 次
0b10：允许重试 15 次
0b11：允许重试 63 次

位 11 **CHSEL**：通道选择 (Channel Selection)
该位由软件置 1/复位。
1：控制流从通道 B 获取通道状态
0：控制流从通道 A 获取通道状态

位 10 **CBDMAEN**：控制流的控制缓冲区 DMA 使能 (Control Buffer DMA ENable for control flow)
该位由软件置 1/复位。
1：使能 DMA 模式以接收通道状态和使用的数据信息。
0：禁止 DMA 模式接收通道状态和使用的数据信息。
当此位置 1 时，每当 CSRNE 标志置 1 时，即产生 DMA 请求。

位 9 **PTMSK**：报头类型的屏蔽位 (Mask of Preamble Type bits)
该位由软件置 1/复位。
1：不将报头类型位复制到 SPDIFRX_DR 中，写入 0 来替代
0：将报头类型位复制到 SPDIFRX_DR 中

位 8 **CUMSK**：通道状态和用户的屏蔽位 (Mask of channel status and user bits)
该位由软件置 1/复位。
1：不将通道状态和用户位复制到 SPDIFRX_DR 中，写入 0 来替代
0：将通道状态和用户位复制到 SPDIFRX_DR 中

位 7 **VMSK**：有效性的屏蔽位 (Mask of Validity bit)
该位由软件置 1/复位。
1：不将有效性位复制到 SPDIFRX_DR 中，写入 0 来替代
0：将有效性位复制到 SPDIFRX_DR 中

位 6 **PMSK**：屏蔽奇偶校验错误位 (Mask Parity error bit)
该位由软件置 1/复位。
1：不将奇偶校验错误位复制到 SPDIFRX_DR 中，写入 0 来替代
0：将奇偶校验错误位复制到 SPDIFRX_DR 中

位 5:4 **DRFMT**：RX 数据格式 (RX Data format)
该位由软件置 1/复位。
0b11：保留
0b10：通过将两个 16 位采样置于一个 32 位字中来封装数据采样
0b01：数据采样采用左对齐 (MSB)
0b00：数据采样采用右对齐 (LSB)

位 3 **RXSTEO**：立体声模式 (STerEO Mode)
该位由软件置 1/复位。
1：外设处于立体声模式
0：外设处于单声道模式
该位用于在发生上溢时处理偏离

位 2 **RXDMAEN**：数据流的接收器 DMA 使能 (Receiver DMA ENable for data flow)
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
当此位置 1 时，每当 RXNE 标志置 1 时，即产生 DMA 请求。

位 1:0 **SPDIFRXEN**：外设块使能 (Peripheral Block Enable)
此字段由软件修改。
该位用于在三种可能的状态（STATE_IDLE、STATE_SYNC 和 STATE_RCV）间更改外设阶段。

0b00：禁止 SPDIFRX (STATE_IDLE)。
0b01：仅使能 SPDIFRX 同步
0b10：保留
0b11：使能 SPDIF 接收器

注： *1 无法从 STATE_RCV 转换为 STATE_SYNC，用户应首先进入 STATE_IDLE。*
2 可以从 STATE_IDLE 转换为 STATE_RCV：在这种情况下，外设从 STATE_IDLE 转换为 STATE_SYNC，并且一旦执行完同步便进入 STATE_RCV。

34.5.2 中断屏蔽寄存器 (SPDIFRX_IMR)

Interrupt mask register

偏移地址：0x04

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IFEIE	SYNCDIE	SBLKIE	OVRIE	PERRIE	CSRNEIE	RXNEIE
									rw	rw	rw	rw	rw	rw	rw

位 31:7 保留，由硬件强制清零。

位 6 **IFEIE**：串行接口错误中断使能 (Serial Interface Error Interrupt Enable)
此位由软件置 1 和清零。
0：禁止中断
1：当 SPDIFRX_SR 寄存器中 SERR = 1、TERR = 1 或 FERR = 1 时，生成 SPDIFRX 接口中断。

位 5 **SYNCDIE**：同步完成 (Synchronization Done)
此位由软件置 1 和清零。
0：禁止中断
1：当 SPDIFRX_SR 寄存器中 SYNCD = 1 时，生成 SPDIFRX 接口中断。

位 4 **SBLKIE**：同步块检测中断使能 (Synchronization Block Detected Interrupt Enable)
此位由软件置 1 和清零。
0：禁止中断
1：当 SPDIFRX_SR 寄存器中 SBD = 1 时，生成 SPDIFRX 接口中断。

位 3 **OVRIE**：上溢错误中断使能 (Overrun error Interrupt Enable)
此位由软件置 1 和清零。
0：禁止中断
1：当 SPDIFRX_SR 寄存器中 OVR = 1 时，生成 SPDIFRX 接口中断

位 2 **PERRIE**：奇偶校验错误中断使能 (Parity error interrupt enable)
此位由软件置 1 和清零。
0：禁止中断
1：当 SPDIFRX_SR 寄存器中 PERR = 1 时，生成 SPDIFRX 接口中断

位 1 **CSRNEIE：控制缓冲区就绪中断使能 (Control Buffer Ready Interrupt Enable)**
此位由软件置 1 和清零。
0：禁止中断
1：当 SPDIFRX_SR 寄存器中 CSRNE = 1 时，生成 SPDIFRX 接口中断。

位 0 **RXNEIE**：RXNE 中断使能 (RXNE interrupt enable)
此位由软件置 1 和清零。
0：禁止中断
1：当 SPDIFRX_SR 寄存器中 RXNE = 1 时，生成 SPDIFRX 接口中断

34.5.3 状态寄存器 (SPDIFRX_SR)

Status register

偏移地址：0x08

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	WIDTH5[14:0]														
	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.	Res.	Res.	Res.	TERR	SERR	FERR	SYNCD	SBD	OVR	PERR	CSRNE	RXNE
							r	r	r	r	r	r	r	r	r

位 31 保留，由硬件强制清零。

位 30:16 **WIDTH5**：使用 SPDIFRX_CLK 计数 5 个符号的持续时间 (Duration of 5 symbols counted with SPDIFRX_CLK)
该值表示 5 个连续符号的时间内包含的 SPDIFRX_CLK 时钟周期数。该值可用于估算 S/PDIF 符号率。其精度受 SPDIFRX_CLK 的频率限制。
例如，如果 SPDIFRX_CLK 固定为 84 MHz，则 WIDTH5 = 147d。S/PDIF 数据流的采样率估算值为：
$Fs = 5 \times F_{SPDIFRX_CLK} / (WIDTH5 \times 64) \sim 44.6$ kHz，因此最接近的标准采样率为 44.1 kHz。
请注意，当 SYNCD 变为高电平时，WIDTH5 将由硬件更新，随后每帧都更新。

位 15:9 保留，由硬件强制清零。

位 8 **TERR**：超时错误 (Time-out error)
当计数器 TRCNT 值达到最大值时，该位由硬件置 1。它表示两次转换之间的时间间隔过长。这通常意味着 SPDIFRX_IN 输入上没有有效信号。
该标志通过将 SPDIFRXEN 写为 0 来清零
如果 SPDIFRX_IMR 寄存器中 IFEIE=1，则会生成中断
0：未检测到序列错误
1：检测到序列错误

位 7 **SERR**：同步错误 (Synchronization error)
当同步由于重试次数过多而失败时，该位由硬件置 1。
该标志通过将 SPDIFRXEN 写为 0 来清零
如果 SPDIFRX_IMR 寄存器中 IFEIE=1，则会生成中断。
0：未检测到同步错误
1：检测到同步错误

位 6 **FERR**：帧错误 (Framing error)
当在接收数据期间发生错误时，该位由硬件置 1：报头未出现在预期位置，短转换未按对分组...
仅当同步完成时 (SYNCD = 1)，该位才能由硬件置 1。
该标志通过将 SPDIFRXEN 写为 0 来清零
如果 SPDIFRX_IMR 寄存器中 IFEIE=1，则会生成中断。
0：未检测到曼彻斯特编码违例
1：检测到曼彻斯特编码违例

位 5 **SYNCD**：同步完成 (Synchronization Done)
当初始同步阶段正确完成时，该位由硬件置 1。
向 SPDIFRX_CLR_SR 寄存器中的相应位写入 1 即可将该标志清零。
如果 SPDIFRX_IMR 寄存器中 SYNCDIE = 1，则会生成中断
0：同步挂起
1：同步完成

位 4 **SBD**：检测到同步块 (Synchronization Block Detected)
当检测到“B”报头时，该位由硬件置 1。
向 SPDIFRX_CLR_SR 寄存器中的相应位写入 1 即可将该标志清零。
如果 SPDIFRX_IMR 寄存器中 SBLKIE = 1，则会生成中断
0：未检测到“B”报头
1：已检测到“B”报头

位 3 **OVR**：上溢错误 (Overrun error)
在 RXNE = 1 的情况下，当 SPDIFRX_DR 寄存器中的接收数据准备好传输时，该位由硬件置 1。
向 SPDIFRX_CLR_SR 寄存器中的相应位写入 1 即可将该标志清零。
如果 SPDIFRX_IMR 寄存器中 OVRIE=1，则会生成中断。
0：无上溢错误
1：检测到上溢错误
注： 当该位置 1 时，SPDIFRX_DR 寄存器的内容不会丢失，但最后接收的数据会丢失。

位 2 **PERR**：奇偶校验错误 (Parity error)
当所接收子帧的数据和状态位中包含奇数个 0 和 1 时，该位由硬件置 1。
向 SPDIFRX_CLR_SR 寄存器中的相应位写入 1 即可将该标志清零。
如果 SPDIFRX_IMR 寄存器中 PIE = 1，则会生成中断。
0：无奇偶校验错误
1：奇偶校验错误

位 1 **CSRNE**：控制缓冲区寄存器不为空 (The Control Buffer register is not empty)
当有效的控制信息就绪时，该位由硬件置 1。
当读取 SPDIFRX_CSR 寄存器时，该标志清零。
如果 SPDIFRX_IMR 寄存器中 CBRDYIE = 1，则会生成中断
0：SPDIFRX_CSR 寄存器中无控制字可用
1：SPDIFRX_CSR 寄存器中有控制字可用

位 0 **RXNE**：读取数据寄存器不为空 (Read data register not empty)
当有效数据进入 SPDIFRX_DR 寄存器时，该位由硬件置 1。
可通过读取 SPDIFRX_CSR 寄存器清零该标志。
如果 SPDIFRX_IMR 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

34.5.4 中断标志清零寄存器 (SPDIFRX_IFCR)

Interrupt Flag Clear register

偏移地址：0x0C

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYNCDCF	SBDCF	OVRCF	PERRCF	Res.	Res.
										w	w	w	w		

位 31:6 保留，由硬件强制清零。

位 5 **SYNCDCF**：清零同步完成标志 (Clears the Synchronization Done flag)
向该位写入 1 可清零 SPDIFRX_SR 寄存器中的 SYNCD 标志。
读取该位将始终返回值 0。

位 4 **SBDCF**：清零同步块检测标志 (Clears the Synchronization Block Detected flag)
向该位写入 1 可清零 SPDIFRX_SR 寄存器中的 SBD 标志。
读取该位将始终返回值 0。

位 3 **OVRCF**：清零上溢错误标志 (Clears the Overrun error flag)
向该位写入 1 可清零 SPDIFRX_SR 寄存器中的 OVR 标志。
读取该位将始终返回值 0。

位 2 **PERRCF**：清零奇偶校验错误标志 (Clears the Parity error flag)
向该位写入 1 可清零 SPDIFRX_SR 寄存器中的 PERR 标志。
读取该位将始终返回值 0。

位 1:0 保留

34.5.5 数据输入寄存器 (SPDIFRX_DR)

Data input register

偏移地址：0x10

复位值：0x00000000

根据 DRFMT 的值，该寄存器可采用 3 种不同格式。当 DRFMT = 0b00 时，格式如下：

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	PT[1:0]		C	U	V	PE	DR[23:16]							
		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
DR[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:30 保留：由硬件强制清零

位 29:28 **PT**：报头类型 (Preamble Type)
这些位表示接收到的报头。
00：未使用
01：接收到报头 B
10：接收到报头 M
11：接收到报头 W
请注意，如果 PTMSK = 1，该字段将强制被设置为零

位 27 **C**：通道状态位 (Channel Status bit)
如果 CUMSK = 0，则包含接收到的通道状态位，否则被强制设置为 0

位 26 **U**：用户位 (User bit)
如果 CUMSK = 0，则包含接收到的用户位，否则被强制设置为 0

位 25 **V**：有效性位 (Validity bit)
如果 VMSK = 0，则包含接收到的有效性位，否则被强制设置为 0

位 24 **PE**：奇偶校验错误位 (Parity Error bit)
如果 PMSK = 0，则包含 PERR 位的副本，否则被强制设置为 0

位 23:0 **DR**：数据值 (Data value)
包含 24 个接收到的数据位，与 D[23] 对齐

34.5.6 数据输入寄存器 (SPDIFRX_DR)

Data input register

偏移地址：0x10

复位值：0x00000000

根据 DRFMT 的值，该寄存器可采用 3 种不同格式。当 DRFMT = 0b01 时，格式如下：

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DR[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
DR[7:0]								Res.	Res.	PT[1:0]		C	U	V	PE
r	r	r	r	r	r	r	r			r	r	r	r	r	r

位 31:8 **DR**：数据值 (Data value)
包含 24 个接收到的数据位，与 D[23] 对齐

位 7:6 保留：由硬件强制清零

位 5:4 **PT**：报头类型 (Preamble Type)
这些位表示接收到的报头。
00：未使用
01：接收到报头 B
10：接收到报头 M
11：接收到报头 W
请注意，如果 PTMSK = 1，该字段将强制被设置为零

位 3 **C**：通道状态位 (Channel Status bit)
如果 CUMSK = 0，则包含接收到的通道状态位，否则被强制设置为 0

位 2 **U**：用户位 (User bit)
如果 CUMSK = 0，则包含接收到的用户位，否则被强制设置为 0

位 1 **V**：有效性位 (Validity bit)
如果 VMSK = 0，则包含接收到的有效性位，否则被强制设置为 0

位 0 **PE**：奇偶校验错误位 (Parity Error bit)
如果 PMSK = 0，则包含 PERR 位的副本，否则被强制设置为 0

34.5.7 数据输入寄存器 (SPDIFRX_DR)

Data input register

偏移地址：0x10

复位值：0x00000000

根据 DRFMT 的值，该寄存器可采用 3 种不同格式。

DRFMT = 0b10 时的数据格式专用于非线性模式，因为仅使用了 16 位（S/PDIF 子帧中的位 23 至位 8）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRNL2[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
DRNL1[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 **DRNL2**：数据值 (Data value)
该字段包含通道 A

位 15:0 **DRNL1**：数据值 (Data value)
该字段包含通道 B

34.5.8 通道状态寄存器 (SPDIFRX_SR)

Channel Status register

偏移地址：0x14

复位值：0x00000000

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

位 31:25 保留

位 24 **SOB**：块开始 (Start Of Block)
该位表示位 CS[0] 是否对应于新块的第一位
0：CS[0] 不是新块的第一位
1：CS[0] 是新块的第一位

位 23:16 **CS[7:0]**：通道 A 状态信息 (Channel A status information)
位 CS[0] 是最早的值

位 15:0 **USR[15:0]**：用户数据信息 (User data information)
位 USR[0] 是最早的值，来自通道 A，USR[1] 来自通道 B。
因此，n 为偶数时，USR[n] 位来自通道 A，否则来自通道 B。

34.5.9 调试信息寄存器 (SPDIFRX_DIR)

Debug Information register

偏移地址：0x18

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	TLO[12:0]												
			r	r	r	r	r	r	r	r	r	r	r	r	r

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

位 31:29 保留，由硬件强制清零。

位 16:28 **TLO**：阈值下限 (Threshold LOW) (TLO = 1.5 x UI / $T_{SPDIFRX_CLK}$)
该字段包含当前阈值下限的估算值。该值可用于估算接收数据流的采样率。TLO 的精度受限于 SPDIFRX_CLK 的周期。采样率按如下公式估算：
采样率 = [2 x TLO x $T_{SPDIFRX_CLK}$ +/- $T_{SPDIFRX_CLK}$] x 2/3
请注意，当 SYNCD 变为高电平时，TLO 将由硬件更新，随后每帧都更新。

位 15:13 保留，由硬件强制清零。

位 12:0 **THI**：阈值上限 (Threshold HIGH) (THI = 2.5 x UI / $T_{SPDIFRX_CLK}$)
该字段包含当前阈值上限的估算值。该值可用于估算接收数据流的采样率。THI 的精度受限于 SPDIFRX_CLK 的周期。采样率按如下公式估算：
采样率 = [2 x THI x $T_{SPDIFRX_CLK}$ +/- $T_{SPDIFRX_CLK}$] x 2/5
请注意，当 SYNCD 变为高电平时，THI 将由硬件更新，随后每帧都更新。

34.5.10 SPDIFRX 接口寄存器映射

下表提供了 SPDIFRX 接口寄存器映射和复位值。

表 190. SPDIFRX 接口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	**SPDIFRX_CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INSEL[2:0]			Res.	WFA	NBTR		CHSEL	CBDMAEN	PTMSK	CUMSK	VMSK	PMSK	DRFMT[1:0]		RXSTEO	RXDMAEN	SPDIFRXEN	
	Reset value														0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	**SPDIFRX_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.	IFEIE	SYNCDIE	SBLKIE	OVRIE	PERRIE	CSRNEIE	RXNEIE
	Reset value																										0	0	0	0	0	0	0
0x08	**SPDIFRX_SR**	Res.	WIDTH5[14:0]															Res.	Res.	Res.	Res.	Res.	Res.	Res.	TERR	SERR	FERR	SYNCD	SBD	OVR	PERR	CSRNE	RXNE
	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	**SPDIFRX_IFCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYNCDCF	SBDCF	OVRCF	PERRCF	Res.	Res.
	Reset value																											0	0	0	0		
0x10	**SPDIFRX_DR**	Res.	Res.	PT[1:0]		C	U	V	PE	DR[23:0]																							
		DR[23:0]																								Res.	PT[1:0]		C	U	V	PE	
		DRNL2[15:0]																DRNL1[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
0x14	**SPDIFRX_CSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SOB	CS[7:0]								USR[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
0x18	**SPDIFRX_DIR**	Res.	Res.	Res.	TLO[12:0]													Reserved			THI[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

35 SD/SDIO/MMC 卡主机接口 (SDMMC)

35.1 SDMMC 主要特性

SD/SDIO MMC 卡主机接口 (SDMMC) 提供 APB2 外设总线与多媒体卡 (MMC)、SD 存储卡以及 SDIO 卡之间的接口。

多媒体卡协会网站上提供了由 MMCA 技术委员会发布的多媒体卡系统规范。

SD 卡协会网站上提供了 SD 存储卡和 SD I/O 卡系统规范。

SDMMC 具有以下特性：

- 完全兼容多媒体卡系统规范版本 4.2。卡支持三种不同数据总线模式：1 位（默认）、4 位和 8 位
- 完全兼容先前版本的多媒体卡（向前兼容性）
- 完全兼容 *SD 存储卡规范版本 2.0*
- 完全兼容 SD I/O 卡规范版本 2.0：卡支持两种不同的数据总线模式：1 位（默认）和 4 位
- 对于 8 位模式，数据传输高达 48 MHz
- 数据和命令输出使能信号，控制外部双向驱动程序

注： *1* *SDMMC 不具备兼容 SPI 的通信模式。*

2 *SD 存储卡协议是多媒体卡系统规范版本 2.11 定义的多媒体卡协议的超集。一些 SD 存储卡器件所需的命令，仅 SD-I/O 的卡或组合卡的 I/O 部分都不支持。部分上述命令（如擦除命令）无法在 SD I/O 器件中使用，因此在 SDIO 协议中不受支持。此外，一些命令在 SD 存储卡和 SD I/O 卡之间有所不同，因此在 SDIO 协议中不受支持。有关详细信息，请参见 SD I/O 卡规范版本 1.0。*

多媒体卡/SD 总线将卡连接到控制器。

当前版本的 SDMMC 每次只支持一个 SD/SDIO/MMC4.2 卡，但支持多个 MMC4.1 或之前版本的卡。

35.2 SDMMC 总线拓扑

总线的通信基于命令和数据传输。

多媒体卡/SD/SD I/O 总线上的基本事务是命令/响应事务。这些类型的总线事务直接在命令或响应结构中传输其信息。此外，某些操作具有数据令牌。

与 SD/SDIO 存储卡的数据相互传输在数据块中执行。与 MMC 的数据相互传输在数据块或流中执行。

图 407. “无响应”和“无数据”操作

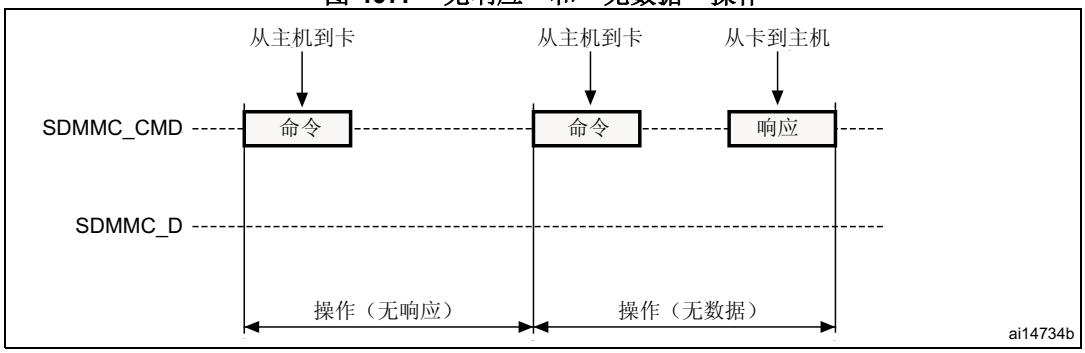


图 408. （多个）块读取操作

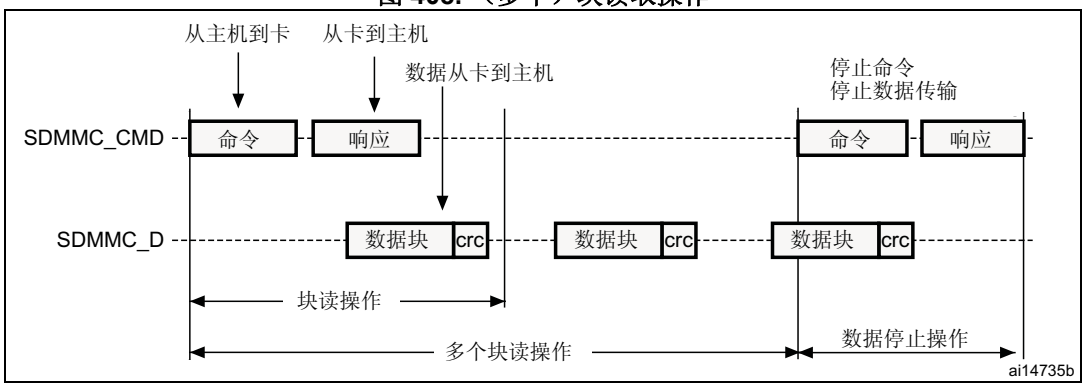
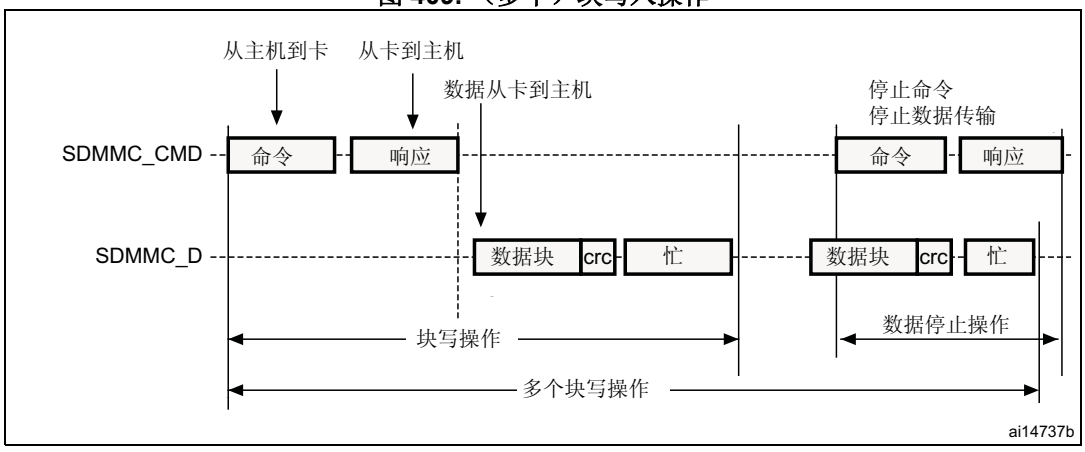


图 409. （多个）块写入操作



注：　只要发出“繁忙”信号（SDMMC_D0 被拉到低电平），SDMMC 便不会发送任何数据。

图 410. 连续读取操作

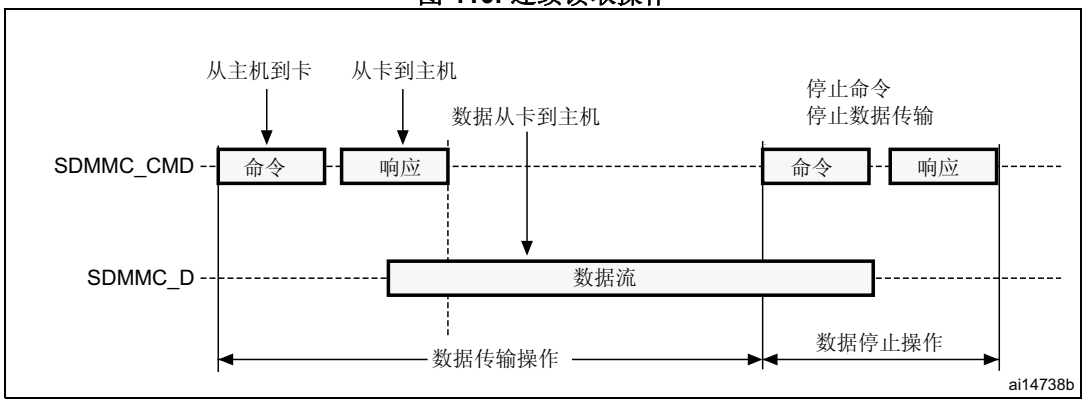
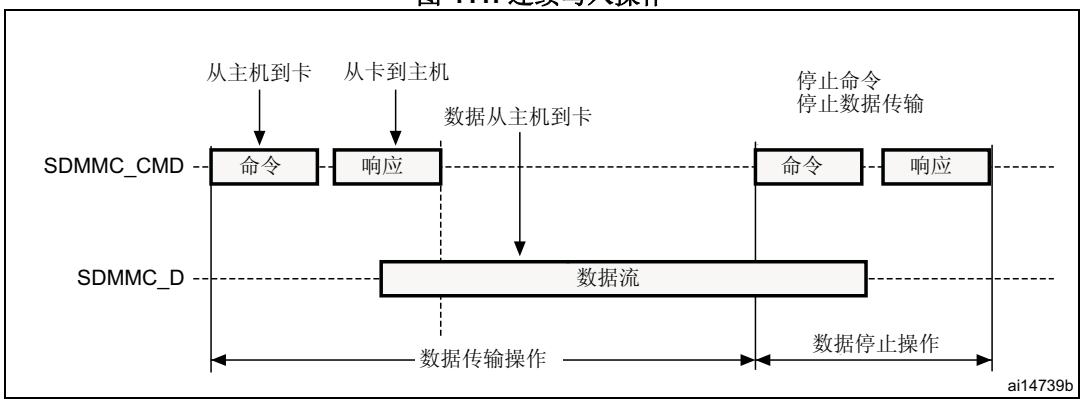


图 411. 连续写入操作

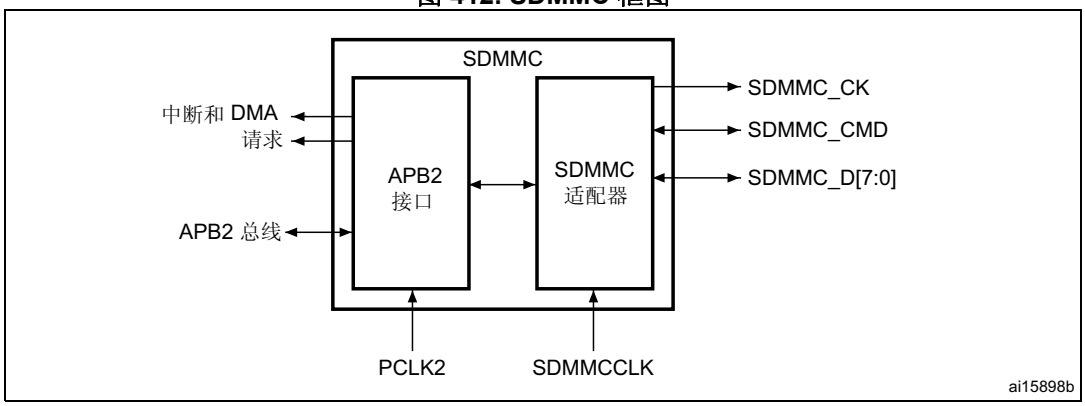


35.3 SDMMC 功能描述

SDMMC 由两部分组成：

- SDMMC 适配器块提供特定于 MMC/SD/SD I/O 卡的所有功能，如时钟生成单元、命令和数据传输。
- APB2 接口访问 SDMMC 适配器寄存器，并且生成中断和 DMA 请求信号。

图 412. SDMMC 框图



默认情况下，SDMMC_D0 用于数据传输。初始化后，主机可以更改数据总线宽度。

如果多媒体卡连接到总线，则 SDMMC_D0、SDMMC_D[3:0] 或 SDMMC_D[7:0] 可以用于数据传输。MMC V3.31 或更低版本仅支持 1 位数据，因此只能使用 SDMMC_D0。

如果 SD 或 SD I/O 卡连接到总线，则主机可以将数据传输配置为使用 SDMMC_D0 或 SDMMC_D[3:0]。所有数据线均以推挽模式运行。

SDMMC_**CMD** 有两种操作模式：

- 开漏引脚，用于初始化（仅限于 MMC V3.31 或更低版本）
- 推挽，用于命令传输（SD/SD I/O 卡 MMC4.2 还将推挽驱动程序用于初始化）

SDMMC_**CK** 是与卡相连的时钟：在每个时钟周期内同时在命令线和数据线上传输一位。对于 MMC V3.31，时钟频率可以在 0 MHz 到 20 MHz 之间变化，对于 MMC V4.0/4.2，可以在 0 到 48 MHz 之间变化，对于 SD/SD I/O 卡，可以在 0 到 25 MHz 之间变化。

SDMMC 使用两个时钟信号：

- SDMMC 适配器时钟 (SDMMCCLK = 48 MHz)
- APB2 总线时钟 (PCLK2)

PCLK2 和 SDMMC_CK 时钟频率必须满足以下条件：

$$频率(PCLK2) > ((3x宽度)/32) \times 频率(SDMMC_CK)$$

表 191 中的信号在多媒体卡/SD/SD I/O 卡总线中使用。

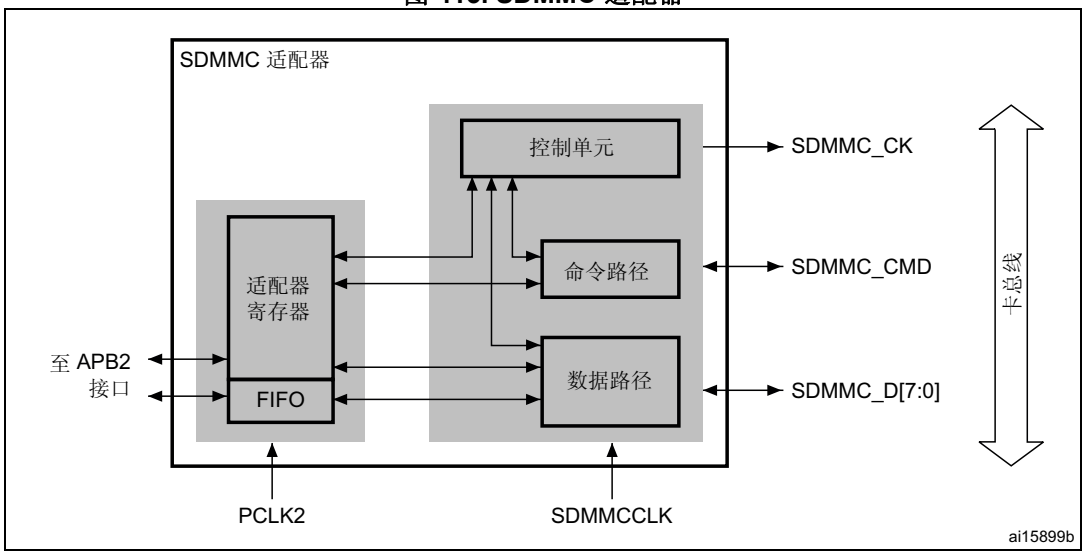
表 191. SDMMC I/O 定义

引脚	方向 (Direction)	说明
SDMMC_CK	输出	多媒体卡/SD/SDIO 卡时钟。该引脚是从主机到卡之间的时钟。
SDMMC_CMD	双向	多媒体卡/SD/SDIO 卡命令。该引脚是双向命令/响应信号。
SDMMC_D[7:0]	双向	多媒体卡/SD/SDIO 卡数据。这些引脚是双向数据总线。

35.3.1 SDMMC 适配器

图 413 显示了 SDMMC 适配器的简化框图。

图 413. SDMMC 适配器



SDMMC 适配器是一个多媒体卡/安全数字存储卡总线主设备，提供与多媒体卡堆栈或安全数字存储卡的接口。该适配器由五个子单元组成：

- 适配器寄存器块
- 控制单元
- 命令路径
- 数据路径
- 数据 FIFO

注： *适配器寄存器和 FIFO 使用 APB2 总线时钟域 (PCLK2)。控制单元、命令路径和数据路径使用 SDMMC 适配器时钟域 (SDMMCCLK)。*

适配器寄存器块

适配器寄存器模块包含所有系统寄存器。该模块还生成将多媒体卡中的静态标志清零的信号。将 1 写入 SDMMC 清零寄存器中对应位的单元时，将生成清零信号。

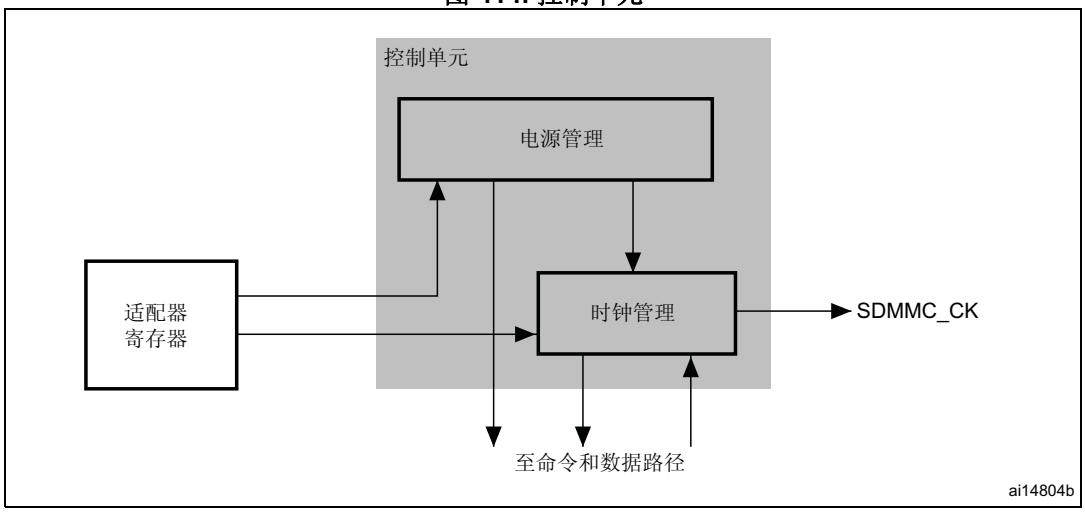
控制单元

控制单元包含电源管理功能和存储卡时钟的时钟分频器。

有三个电源阶段：

- 掉电
- 上电
- 通电

图 414. 控制单元



控制单元如*图 414* 所示。控制单元由电源管理子单元和时钟管理子单元组成。

电源管理子单元会在断电阶段和上电阶段中禁止卡总线输出信号。

时钟管理子单元负责生成和控制 SDMMC_CK 信号。SDMMC_CK 输出可以使用时钟分频或时钟旁路模式。时钟输出在以下情况下无效：

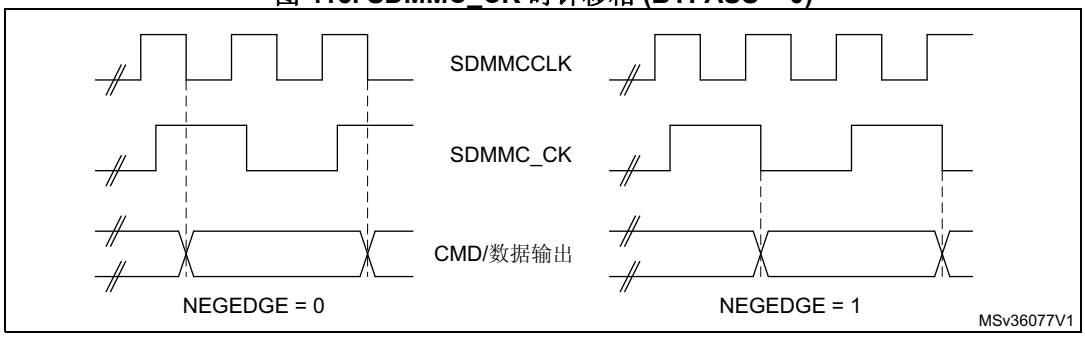
- 复位后
- 在断电或上电阶段中
- 在使能了节电模式并且卡总线处于空闲状态的情况下（命令和数据路径子单元进入空闲阶段后的八个时钟周期之后）

时钟管理子单元负责控制 SDMMC_CK 移相。如果未处于旁路模式下，则当 SDMMC_CLKCR[13] 位复位 (NEGEDGE = 0) 时，将在 SDMMC_CK 上升沿后的 SDMMCCLK 下降沿生成 SDMMC 命令和数据输出（SDMMC_CK 上升沿在 SDMMCCLK 上升沿时出现）。当 SDMMC_CLKCR[13] 位置 1 (NEGEDGE = 1) 时，将在 SDMMC_CK 下降沿更改 SDMMC 命令和数据。

当 SDMMC_CLKCR[10] 置 1 (BYPASS = 1) 时，SDMMC_CK 上升沿在 SDMMCCLK 上升沿时出现。无论 NEGEDGE 值为多少，都将在 SDMMCCLK 下降沿更改数据和命令。

数据和命令响应通过 SDMMC_CK 上升沿来锁存。

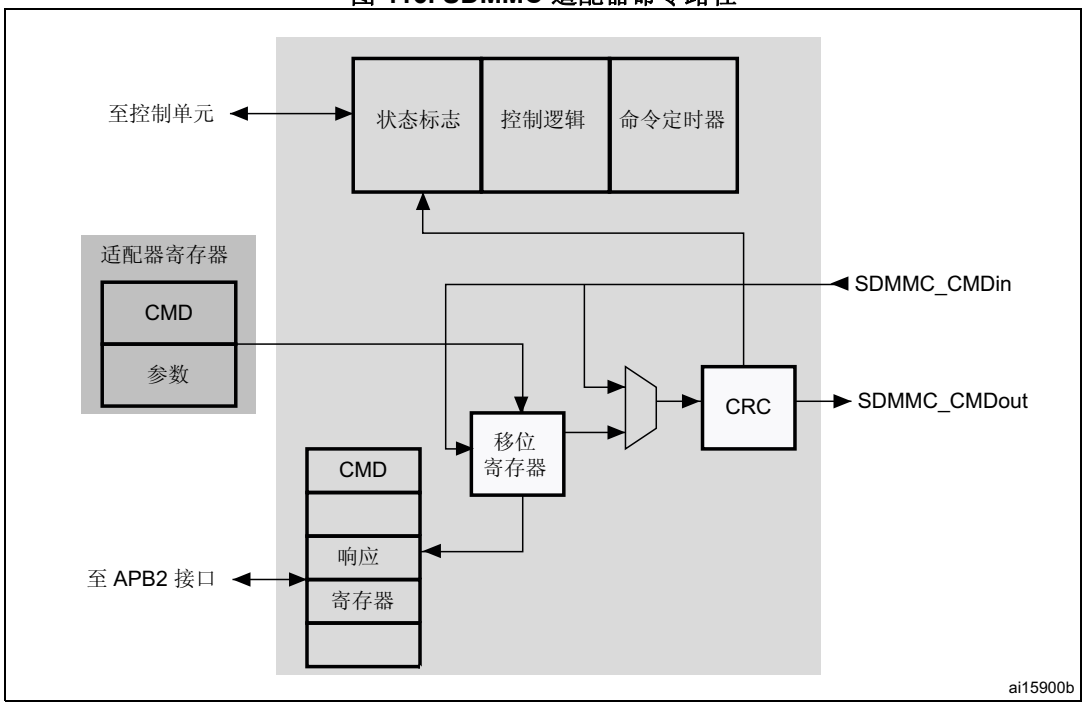
图 415. SDMMC_CK 时钟移相 (BYPASS = 0)



命令路径

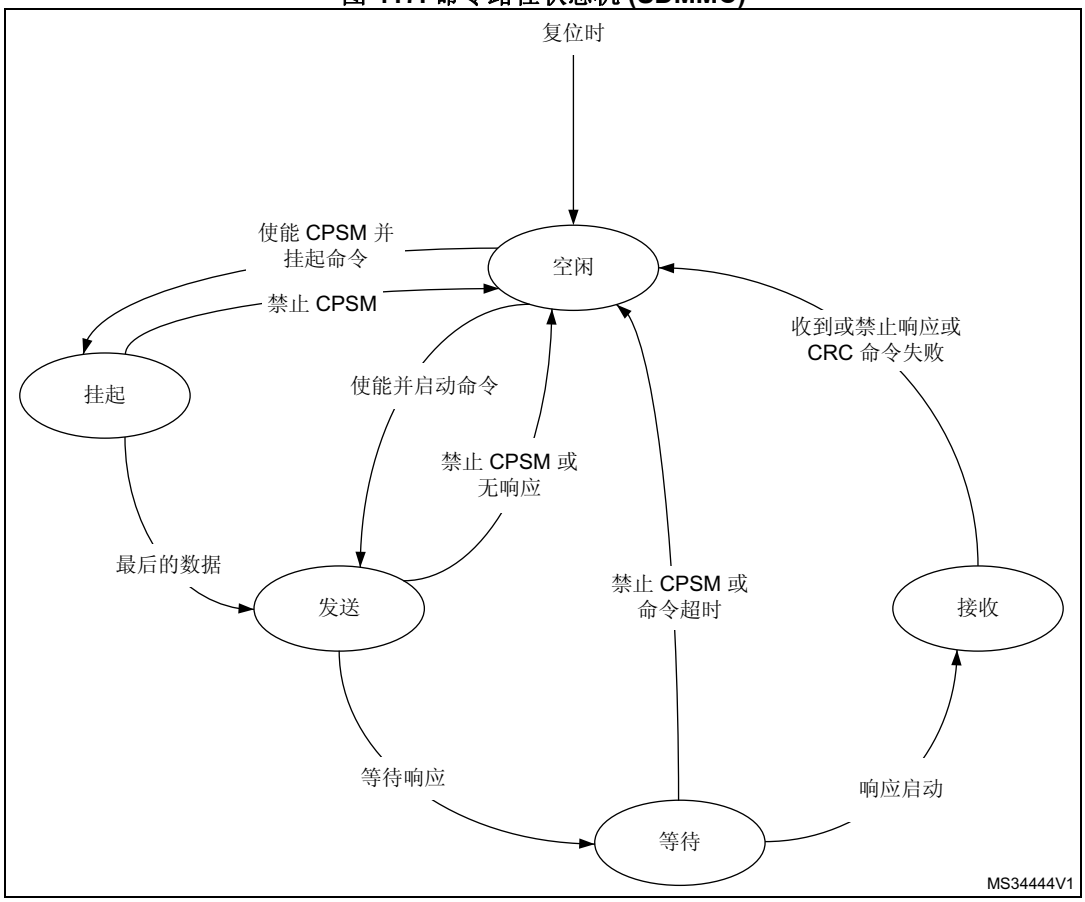
命令路径单元向卡发送命令并从卡接收响应。

图 416. SDMMC 适配器命令路径



- 命令路径状态机 (CPSM)
  - 写入命令寄存器并且将使能位置 1 后，命令传输将开始。发送了命令后，命令路径状态机 (CPSM) 将将状态标志置 1，并且在不需要响应时进入空闲状态。如果需要响应，则等待响应（请参见*第 1096 页上的图 417*）。收到响应时，将比较生成的 CRC 代码和内部生成代码，并且将相应的状态标志置 1。

图 417. 命令路径状态机 (SDMMC)



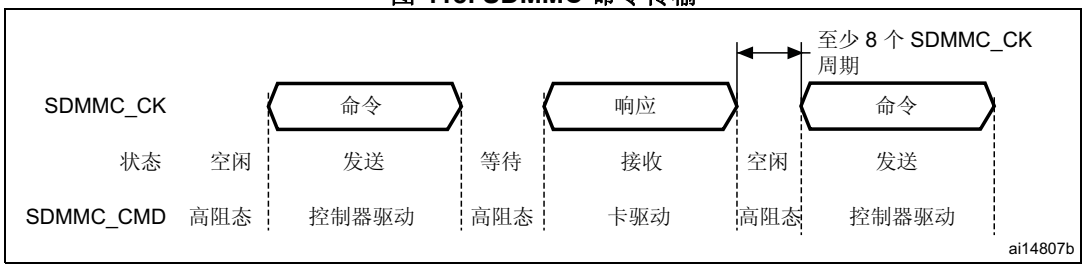
进入等待状态后，命令计时器便开始运行。如果 CPSM 尚未变为接收状态便已达到超时，则将超时标志置 1 并且进入空闲状态。

注： *命令超时为固定值：64 个 SDMMC_CK 时钟周期。*

如果在命令寄存器中将中断位置 1，则禁止计时器运行并且 CPSM 等待来自其中一个卡的中断请求。如果在命令寄存器中将挂起位置 1，则 CPSM 进入挂起状态并等待来自数据路径子单元的 CmdPend 信号。检测到 CmdPend 时，CPSM 将变为发送状态。这会使能数据计数器，用以触发停止命令传输。

注： *CPSM 保持空闲状态至少八个 SDMMC_CK 周期以满足 N_{CC} 和 N_{RC} 时序限制。N_{CC} 是两个主机命令之间的最小延迟，N_{RC} 是主机命令和卡响应之间的最小延迟。*

图 418. SDMMC 命令传输



- 命令格式
  - 命令：命令是用于启动操作的令牌。命令从主机发送到单个卡（编址命令），或发送到所有已连接的卡（MMC V3.31 或更低版本可以使用广播命令）。命令在 CMD 线上以串行方式传输。所有命令都为固定长度 48 位。*表 192* 中显示了多媒体卡、SD 存储卡和 SDIO 卡的命令令牌的常规格式。

    命令路径以半双工模式运行，因此可以发送或者接收命令和响应。如果 CPSM 未处于发送状态，则 SDMMC_CMD 输出处于高阻态，如*第 1097 页上的图 418* 中所示。SDMMC_CMD 上的数据与 SDMMC_CK 上升沿同步。*表 192* 给出了命令格式。

表 192. 命令格式

位的位置	宽度	值	说明
47	1	0	起始位
46	1	1	传输位
[45:40]	6	-	命令索引 (Command index)
[39:8]	32	-	参数
[7:1]	7	-	CRC7
0	1	1	结束位

  - 响应：响应是一个令牌，它作为对先前接收命令的应答，从已寻址到的卡（或者，对于 MMC V3.31 或更低版本，同步地从所有已连接的卡）发送到主机。响应在 CMD 线上以串行方式传输。

SDMMC 支持两种响应类型。两种类型均使用 CRC 错误校验：

- 48 位短响应
- 136 位长响应

注： *如果响应不包含 CRC（CMD1 响应），则设备驱动程序必须忽略 CRC 失败状态。*

表 193. 短响应格式

位的位置	宽度	值	说明
47	1	0	起始位
46	1	0	传输位
[45:40]	6	-	命令索引 (Command index)
[39:8]	32	-	参数
[7:1]	7	-	CRC7（或 1111111）
0	1	1	结束位

表 194. 长响应格式

位的位置	宽度	值	说明
135	1	0	起始位
134	1	0	传输位
[133:128]	6	111111	保留
[127:1]	127	-	CID 或 CSD（包括内部 CRC7）
0	1	1	结束位

命令寄存器包含命令索引（发送到卡的六位）和命令类型。它们确定命令是否需要响应以及响应的长度是 48 位还是 136 位（请参见*第 1129 页上的第 35.8.4 节*）。命令路径实施*表 195* 中显示的状态标志：

表 195. 命令路径状态标志

标志	说明
CMDREND	在响应 CRC 正常的情况下将此标志置 1
CCRCFAIL	在响应 CRC 失败的情况下将此标志置 1
CMDSENT	发送了不需要响应的命令时将此标志置 1
CTIMEOUT	响应超时。
CMDACT	正在进行命令传输。

CRC 生成器计算 CRC 代码前面所有位的 CRC 校验和。这包括起始位、传输位、命令索引和命令参数（或卡状态）。对于长响应格式，将为 CID 或 CSD 的前 120 位计算 CRC 校验和。请注意，CRC 计算中不使用起始位、传输位和六个保留位。

CRC 校验和是一个 7 位值：

CRC[6:0] = 余数 [(M(x) * x^7) / G(x)]

G(x) = $x^7 + x^3 + 1$

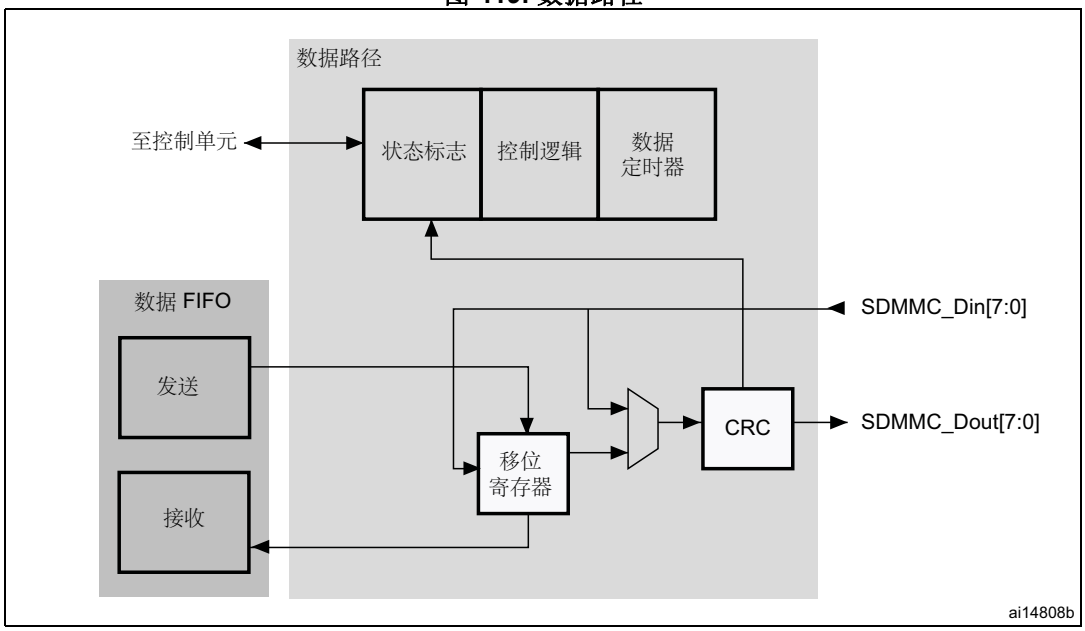
M(x) = (起始位) * x^{39} + ... + (CRC 前的最后一位) * x^0，或者

M(x) = (起始位) * x^{119} + ... + (CRC 前的最后一位) * x^0

数据路径

数据路径子单元负责与卡相互传输数据。*图 419* 显示了数据路径的框图。

图 419. 数据路径



可以使用时钟控制寄存器对卡数据总线宽度进行编程。如果使能了 4 位宽度的总线模式，则使用所有四个数据信号线 (SDMMC_D[3:0]) 在每个时钟周期内传输 4 个数据位。如果使能了 8 位宽度的总线模式，则使用所有八个数据信号线 (SDMMC_D[7:0]) 在每个时钟周期内传输 8 个数据位。如果未使能宽总线模式，则使用 SDMMC_D0 在每个时钟周期内仅传输一位。

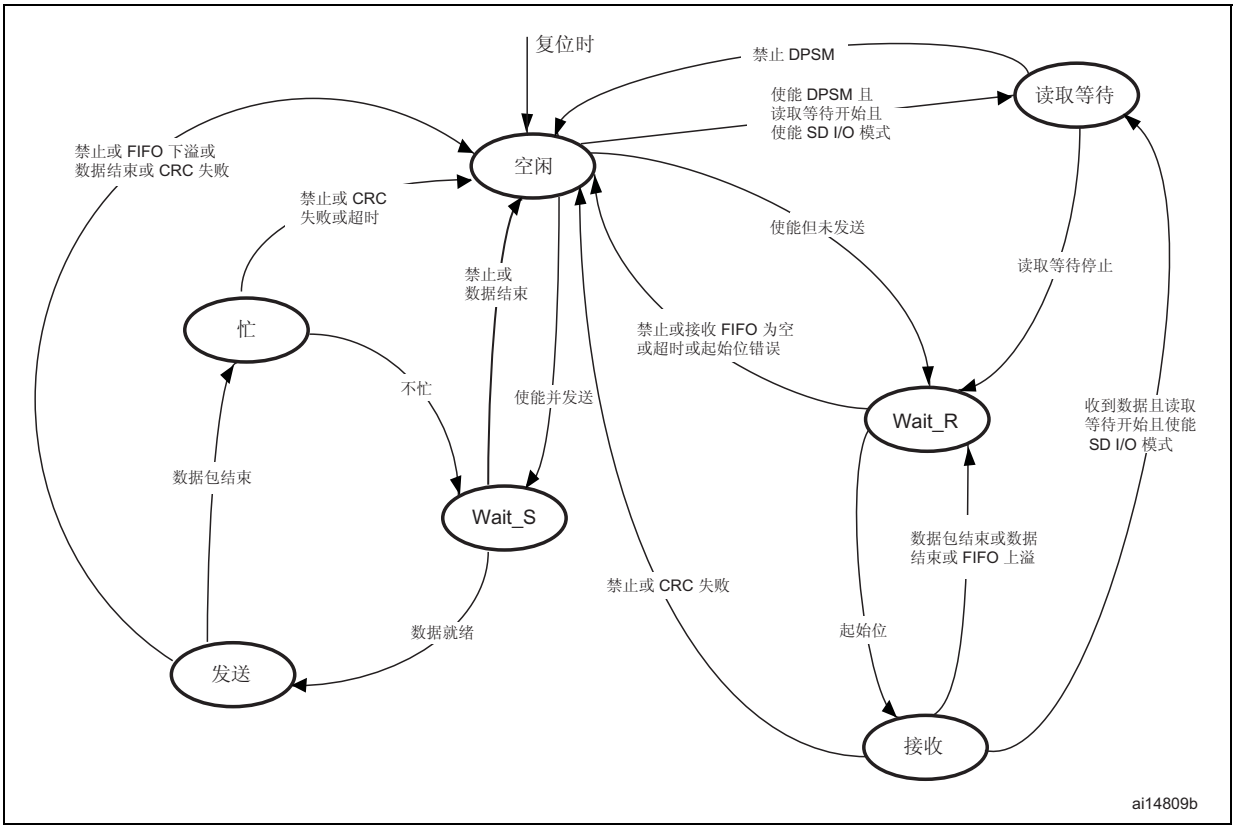
根据传输方向（发送或接收），数据路径状态机 (DPSM) 将变为 Wait_S 或 Wait_R 状态（如果已使能）：

- 发送：DPSM 变为 Wait_S 状态。如果传输 FIFO 中有数据，则 DPSM 变为发送状态并且数据路径子单元开始向卡发送数据。
- 接收：DPSM 变为 Wait_R 状态并等待起始位。在 DPSM 收到起始位时，将变为接收状态，并且数据路径子单元开始从卡中接收数据。

数据路径状态机 (DPSM)

DPSM 以 SDMMC_CK 频率运行。卡总线信号上的数据与 SDMMC_CK 的上升沿保持同步。DPSM 有六种状态，如*图 420：数据路径状态机 (DPSM)* 中所示。

图 420. 数据路径状态机 (DPSM)



- 空闲：数据路径处于无效状态，SDMMC_D[7:0] 输出处于高阻态。当写入数据控制寄存器并将使能位置 1 后，DPSM 将使用新值来加载数据计数器，根据数据方向位，将变为 Wait_S 或 Wait_R 状态。
- Wait_R：如果数据计数器等于零，则 DPSM 会在接收 FIFO 为空时变为空闲状态。如果数据计数器不为零，则 DPSM 等待 SDMMC_D 上的起始位。如果 DPSM 在超时之前收到了起始位，则 DPSM 变为接收状态，并加载数据块计数器。如果 DPSM 尚未检测到起始位便已超时，则变为空闲状态并且将超时状态标志置 1。
- 接收：从卡收到的串行数据以字节为单位打包，并写入到数据 FIFO。根据数据控制寄存器中的传输模式位，数据传输模式可以是块或流：
  - 在块模式中，当数据块计数器达到零时，DPSM 将等待直至其收到 CRC 代码。如果收到的代码与内部生成的 CRC 代码相匹配，则 DPSM 变为 Wait_R 状态。如果不匹配，则将 CRC 失败状态标志置 1 并且 DPSM 变为空闲状态。
  - 在流模式中，DPSM 在数据计数器不为零时接收数据。当计数器为零时，会将移位寄存器中其余数据写入到数据 FIFO，并且 DPSM 将变为 Wait_R 状态。

  如果发生 FIFO 溢出错误，则 DPSM 将 FIFO 错误标志置 1 并变为空闲状态：
- Wait_S：如果数据计数器为零，则 DPSM 变为空闲状态。如果不为零，则 DPSM 将等待直至停止发出 FIFO 空标志，并变为发送状态。

注： *DPSM 在 Wait_S 状态下需要至少保持两个时钟周期以满足 N_{WR} 计时要求，N_{WR} 是从收到卡响应到开始从主机传输数据这一期间的时钟周期数。*

- 发送：DPSM 开始向卡发送数据。根据数据控制寄存器中的传输模式，数据传输模式可以是块或流：
  - 在块模式中，当数据块计数器达到零时，DPSM 将发送一个内部生成的 CRC 代码和结束位，并变为繁忙状态。
  - 在流模式中，DPSM 将在使能位处于高电平并且数据计数器不为零时，向卡发送数据。然后它会变为空闲状态。

  如果发生 FIFO 下溢错误，则 DPSM 将 FIFO 错误标志置 1 并变为空闲状态。
- 忙碌：DPSM 等待 CRC 状态标志：
  - 如果未收到正确的 CRC 状态，则变为空闲状态并且将 CRC 失败状态标志置 1。
  - 如果收到正确的 CRC 状态，则在 SDMMC_D0 未处于低电平（卡未处于繁忙状态）的情况下变为 Wait_S 状态。

  如果 DPSM 处于繁忙状态时发生超时，则数据超时标志置 1 并变为空闲状态。

  当 DPSM 处于 Wait_R 或繁忙状态时将使能数据计时器，并生成数据超时错误：
  - 发送数据时，如果 DPSM 处于繁忙状态的时间超过了编程的超时周期，则发生超时错误。
  - 接收数据时，如果数据的结尾不为“真”，并且 DPSM 处于 Wait_R 状态的时间超过了编程的超时周期，则发生超时。
- **数据：**可以将数据从卡发送到主机，或从主机到卡。数据通过数据线来传输。数据存储在 32 字的 FIFO 中，每个字为 32 位宽。

表 196. 数据令牌格式

说明	起始位	数据	CRC16	结束位
块数据	0	-	是	1
流数据	0	-	否	1

DPSM 标志

数据路径子单元传输状态由几个状态标志来报告。

表 197. DPSM 标志

标志	说明
DBCKEND	当发送/接收数据块的 CRC 校验通过时，设置为高电平。 在 SDIO 多字节传输模式下，该标志将在传输结束时置 1（多字节传输被主机视为单块传输）。
DATAEND	当 SDMMC_DCOUNT 寄存器递减并达到 0 时，设置为高电平。 DATAEND 表示 SDMMC 数据线上的传输结束。
DTIMEOUT	当达到数据超时周期时，设置为高电平。 当数据定时器达到 0 时且 DPSM 处于 Wait_R 状态或繁忙状态时，超时置 1。如果 DPSM 保持繁忙状态的时间超过编程的周期，则 DTIMEOUT 可在 DATAEND 后置 1。
DCRCFAIL	当发送/接收数据块的 CRC 校验失败时，设置为高电平。

数据 FIFO

数据 FIFO（先进先出）子单元是一个数据缓冲器，带发送和接收单元。

FIFO 包含一个宽度为 32 位且深度为 32 字的数据缓冲器和发送/接收逻辑。由于数据 FIFO 在 APB2 时钟域 (PCLK2) 中运行，因此来自 SDMMC 时钟域 (SDMMCCLK) 的子单元中的所有信号都将重新同步。

根据 TXACT 和 RXACT 标志，可以将 FIFO 禁止、使能传输或使能接收。TXACT 和 RXACT 是由数据路径子单元驱动，并且两者互斥：

- 发出 TXACT 时，传输 FIFO 引用传输逻辑和数据缓冲器
- 发出 RXACT 时，接收 FIFO 引用接收逻辑和数据缓冲器

- 传输 FIFO：

  如果使能了 SDMMC 以用于传输，那么可以通过 APB2 接口将数据写入到传输 FIFO。

  可以通过 32 个连续地址来访问传输 FIFO。传输 FIFO 包含一个数据输出寄存器，其中存储了读取指针所指向的数据字。在数据路径子单元已加载了其移位寄存器之后，该子单元会将读取指针递增并将新数据推出。

  如果禁止了传输 FIFO，则停止发出所有状态标志。数据路径子单元在传输数据时置位 TXACT。

表 198. 传输 FIFO 状态标志

标志	说明
TXFIFOF	当所有 32 个传输 FIFO 字都包含有效数据时，设置为高电平。
TXFIFOE	当传输 FIFO 不包含有效数据时，设置为高电平。
TXFIFOHE	当 8 个或更多个传输 FIFO 字为空时，设置为高电平。该标志可以用作 DMA 请求。
TXDAVL	当传输 FIFO 包含有效数据时，设置为高电平。该标志与 TXFIFOE 标志正好相反。
TXUNDERR	发生下溢错误时，设置为高电平。通过写入到 SDMMC 清零寄存器可将该标志清零。 *注：* 使用 TXUNDERR 和 DMA 填充 SDMMC FIFO 时，*则用户软件应禁止 DMA 流，然后向 SDMMC_DCTRL 中的 DMAEN 位写入“0”（以禁止生成 DMA 请求）。*

- 接收 FIFO

  当数据路径子单元接收到一个字的数据时，该子单元将在写入数据总线上驱动数据。在写入操作完成后，写入指针递增。在读取端，读取指针的当前值指向的 FIFO 字内容被推入读取数据总线中。如果禁止了接收 FIFO，则会停止发出所有状态标志，并且将读取和写入指针复位。数据路径子单元在接收数据时置位 RXACT。*表 199* 列出了接收 FIFO 状态标志。可以通过 32 个连续地址来访问接收 FIFO。

表 199. 接收 FIFO 状态标志

标志	说明
RXFIFOF	当所有 32 个接收 FIFO 字都包含有效数据时，设置为高电平
RXFIFOE	当接收 FIFO 不包含有效数据时，设置为高电平。
RXFIFOHF	当 8 个或更多个接收 FIFO 字包含有效数据时，设置为高电平。该标志可以用作 DMA 请求。
RXDAVL	当接收 FIFO 非空时，设置为高电平。该标志与 RXFIFOE 标志正好相反。
RXOVERR	发生溢出错误时，设置为高电平。通过写入到 SDMMC 清零寄存器可将该标志清零。 *注：使用 RXOVERR 和 DMA 读取 SDMMC FIFO 时，用户软件应禁止 DMA 流，然后向 SDMMC_DCTRL 中的 DMAEN 位写入“0”（以禁止生成 DMA 请求）。*

35.3.2 SDMMC APB2 接口

APB2 接口生成中断和 DMA 请求，并且访问 SDMMC 适配器寄存器和数据 FIFO。该接口由数据路径、寄存器解码器和中断/DMA 逻辑构成。

SDMMC 中断

中断逻辑生成一个中断请求信号，当至少其中一个所选状态标志为高电平时，便发出此信号。提供了一个屏蔽寄存器，用于选择将生成中断的条件。如果对应的屏蔽标志置 1，则状态标志将生成中断请求。

SDMMC/DMA 接口

SDMMC APB 接口控制所有子单元以在主机和卡之间执行传输

使用 DMA 读取的步骤示例

发送 CMD17 (READ_BLOCK)，如下所示：

a) 对 SDMMC 数据长度寄存器进行编程（在卡识别过程之前，SDMMC 数据定时器寄存器应该已编程）。

b) 编程 DMA 通道（请参见 *SDMMC 控制器的 DMA 配置*）。

c) 对 SDMMC 数据控制寄存器进行编程：DTEN 为“1”（使能 SDMMC 卡主机以发送数据）；DTDIR 为“1”（从卡到控制器）；DTMODE 为“0”（块数据传输）；DMAEN 为“1”（使能 DMA）；DBLOCKSIZE 为 0x9（512 字节）。其它字段无关。

d) 使用数据传输目标卡的地址单元对 SDMMC 参数寄存器进行编程。

e) 对 SDMMC 命令寄存器进行编程：CmdIndex 为 17 (READ_BLOCK)；WaitResp 为“1”（SDMMC 卡主机等待响应）；CPSMEN 为“1”（使能 SDMMC 卡主机，以发送命令）。其它字段是其各自的复位值。

f) 等待 SDMMC_STA[6] = CMDREND 中断，（如果命令路径上没有错误，CMDREND 将置 1）。

g) 等待 SDMMC_STA[10] = DBCKEND，（没有错误时，DBCKEND 将置 1，直至 CRC 校验通过）。

h) 等待 FIFO 为空，当 FIFO 为空时，SDMMC_STA[5] = RXOVERR 值必须进行校验以确保读取成功。

注： *若在最后 1-4 个字节发生 FIFO 上溢错误，则可能发生以下情况：在 DATAEND 标志置 1 后再经过 2 个 APB 时钟周期，RXOVERR 标志置 1。要保证读取操作成功，RXOVERR 必须在 FIFO 为空后进行校验。*

使用 DMA 写入的步骤示例

发送 CMD24 (WRITE_BLOCK)，如下所示：

a) 对 SDMMC 数据长度寄存器进行编程（在卡识别过程之前，SDMMC 数据定时器寄存器应该已编程）。

b) 对 DMA 通道进行编程（请参见 *SDMMC 控制器的 DMA 配置*）。

c) 使用数据传输目标卡的地址单元对 SDMMC 参数寄存器进行编程。

d) 对 SDMMC 命令寄存器进行编程：CmdIndex 为 24 (WRITE_BLOCK)；WaitResp 为“1”（SDMMC 卡主机等待响应）；CPSMEN 为“1”（使能 SDMMC 卡主机，以发送命令）。其它字段是其各自的复位值。

e) 等待 SDMMC_STA[6] = CMDREND 中断，然后对 SDMMC 数据控制寄存器进行编程：DTEN 为“1”（使能 SDMMC 卡主机以发送数据）；DTDIR 为“0”（从控制器到卡）；DTMODE 为“0”（块数据传输）；DMAEN 为“1”（使能 DMA）；DBLOCKSIZE 为 0x9（512 字节）。其它字段无关。

f) 等待 SDMMC_STA[10] = DBCKEND（没有错误时，DBCKEND 将置 1）。

SDMMC 控制器的 DMA 配置

a) 使能 DMA2 控制器并将所有挂起的中断清零。

b) 使用存储单元的基址对 DMA2_Stream3 (or DMA2_Stream6) Channel4 源地址寄存器进行编程，并使用 SDMMC_FIFO 寄存器地址对 DMA2_Stream3 (or DMA2_Stream6) Channel4 目标地址寄存器进行编程。

c) 对 DMA2_Stream3 (or DMA2_Stream6) Channel4 控制寄存器进行编程（存储器递增、非外设递增，外设和源宽度是字）。

d) 对 DMA2_Stream3 (or DMA2_Stream6) Channel4 进行编程以选择外设作为流控制器（将 DMA_S3CR（或 DMA_S6CR）配置寄存器中的 PFCTRL 位置 1）。

e) 在 DMA2_Stream3 (or DMA2_Stream6) Channel4 中将递增突发传输配置为 4 个节拍（至少来自于外设端）。

f) 使能 DMA2_Stream3 (or DMA2_Stream6) Channel4。

注： *SDMMC 主机只允许在外设流控制器模式下使用 DMA。用于 SDMMC 的 DMA 流必须配置为外设流控制器模式。*

SDMMC 仅生成针对 DMA 控制器的 DMA 突发请求。在外设端，DMA 必须配置为递增突发模式。

35.4 卡功能说明

35.4.1 卡识别模式

在卡识别模式下，主机复位所有卡，验证运行电压范围，识别卡并在总线上为每个卡设置相对卡地址 (RCA)。卡识别模式中的所有数据通信都仅使用命令行 (CMD)。

35.4.2 智能卡复位

GO_IDLE_STATE 命令 (CMD0) 是软件复位命令，它会将多媒体卡和 SD 存储器置于空闲状态。IO_RW_DIRECT 命令 (CMD52) 可复位 SD I/O 卡。在上电或 CMD0 之后，所有卡输出总线驱动程序都处于高阻态状态，并且这些卡将使用默认的相对卡地址 (RCA=0x0001) 和默认的驱动程序阶段寄存器设置（最低速度，最高驱动电流容量）进行初始化。

35.4.3 工作电压范围验证

所有卡都可以使用规范范围内的任何工作电压与 SDMMC 卡主机进行通信。卡上的运行条件寄存器 (OCR) 定义了支持的最小和最大 V_{DD} 值。

在有效负载存储器中存储卡标识号 (CID) 和卡特定数据 (CSD) 的卡仅能够在数据传输 V_{DD} 条件下传播此信息。当 SDMMC 卡主机模块和卡具有不兼容的 V_{DD} 范围时，卡无法完成识别周期且无法发送 CSD 数据。因此，SEND_OP_COND (CMD1)、SD_APP_OP_COND（用于 SD 存储器的 ACMD41）和 IO_SEND_OP_COND（用于 SD I/O 的 CMD5）等特殊命令旨在提供一种机制，用于识别和拒绝与 SDMMC 卡主机要求的 V_{DD} 范围不匹配的卡。SDMMC 卡主机会发送所需的 V_{DD} 电压窗口作为这些命令的运算对象。无法在指定范围内执行数据传输的卡将断开与总线的连接，并且变为无效状态。

如果在使用这些命令时不将电压范围作为运算对象包括在内，则 SDMMC 卡主机可能会查询每个卡并确定通用电压范围，然后再将超出范围的卡置于无效状态。当 SDMMC 卡主机能够选择通用电压范围或者当用户要求就卡无法使用这一情况进行通知时，将会使用此查询。

35.4.4 卡识别过程

多媒体卡和 SD 卡的卡识别过程有所不同。对于多媒体卡，识别过程以时钟速率 F_{od} 开始。SDMMC_CMD 线输出驱动器是开漏引脚，在此识别过程中允许并行的卡操作。注册过程以如下方式完成：

1. 激活总线。
2. SDMMC 卡主机广播 SEND_OP_COND (CMD1) 以接收运行条件。
3. 响应是来自所有卡的运行条件寄存器的线与运算。
4. 不兼容的卡将被置于无效状态。
5. SDMMC 卡主机向所有有效卡广播 ALL_SEND_CID (CMD2)。
6. 有效卡同时以串行方式发送其 CID 号。如果卡的输出 CID 位与命令线上的位不匹配，则卡将停止传输，必须等待下一个识别周期。成功地将完整 CID 传输到 SDMMC 卡主机的卡进入识别状态。
7. SDMMC 卡主机向该卡发出 SET_RELATIVE_ADDR (CMD3)。这一新地址称为相对卡地址 (RCA)；它比 CID 更短，可对卡进行寻址。分配的卡变为待机状态，它不会对进一步的识别周期进行响应，并且其输出将从开漏切换为推挽。
8. SDMMC 卡主机不断重复步骤 5 到 7，直到收到超时条件为止。

对于 SD 卡，识别过程是以时钟速率 F_{od} 开始，并且 SDMMC_CMD 线输出驱动是推挽驱动器，而非开漏引脚。对准过程以如下方式完成：

1. 激活总线。
2. SDMMC 卡主机广播 SD_APP_OP_COND (ACMD41)。
3. 卡以其运行条件寄存器的内容进行响应。
4. 不兼容的卡将被置于无效状态。
5. SDMMC 卡主机向所有有效卡广播 ALL_SEND_CID (CMD2)。
6. 这些卡将发回其唯一的卡识别号 (CID) 并进入识别状态。
7. SDMMC 卡主机向某个地址的有效卡发出 SET_RELATIVE_ADDR (CMD3)。这一新地址称为相对卡地址 (RCA)；它比 CID 更短，可对卡进行寻址。分配的卡变为待机状态。SDMMC 卡主机可以重新发出此命令以更改 RCA。卡的 RCA 是最后分配的值。
8. SDMMC 卡主机对所有有效卡重复步骤 5 到 7。

对于 SD I/O 卡，对准过程以如下方式完成：

1. 激活总线。
2. SDMMC 卡主机发送 IO_SEND_OP_COND (CMD5)。
3. 卡以其运行条件寄存器的内容进行响应。
4. 不兼容的卡将被设置无效状态。
5. SDMMC 卡主机向某个地址的有效卡发出 SET_RELATIVE_ADDR (CMD3)。这一新地址称为相对卡地址 (RCA)；它比 CID 更短，可对卡进行寻址。分配的卡变为待机状态。SDMMC 卡主机可以重新发出此命令以更改 RCA。卡的 RCA 是最后分配的值。

35.4.5 块写入

在块写入 (CMD24 - 27) 期间，一个或多个数据块从主机传输到卡，并且由主机在每个块的结尾追加一个 CRC。支持块写入的卡始终可以接受 WRITE_BL_LEN 所定义的数据块。如果 CRC 失败，卡将在 SDMMC_D 线上指示失败，已传输的数据将废弃而不会写入，并且将忽略所有进一步传输的块（在多块写入模式中）。

如果主机使用不完整的块（其累计长度没有进行块对齐）并且不允许块偏离（未设置 CSD 参数 WRITE_BLK_MISALIGN），则卡会在第一个偏离块之前检测块偏离错误。（ADDRESS_ERROR 错误位在状态寄存器中设置）。如果主机尝试在写保护区域进行写入，也将会中止写入操作。但是，在此情况下，卡会将 WP_VIOLATION 位置 1。

对 CID 和 CSD 寄存器的编程不需要先前的块长度设置。传输的数据也受到 CRC 保护。如果 CSD 或 CID 寄存器的某一部分存储在 ROM 中，则这一无法更改的部分必须与接收缓冲器的对应部分相匹配。如果不匹配，则卡将报告错误并且不会更改任何寄存器内容。某些卡可能需要很长且无法预测的时间来写入数据块。收到数据块并完成 CRC 校验后，卡开始进行写入，如果卡的写入缓冲器已满并且无法接受来自新 WRITE_BLOCK 命令的新数据，则会保持 SDMMC_D 线的低电平。主机可能随时使用 SEND_STATUS 命令 (CMD13) 轮询卡状态，而卡将使用其状态进行响应。READY_FOR_DATA 状态位指示卡是否可以接受新数据以及写入进程是否仍在进行中。主机可以通过发出 CMD7（用于选择不同的卡）来取消选择卡，这会将卡置于断开连接状态并释放 SDMMC_D 线，但不会中断写入操作。重新选择卡时，如果编程仍在进行中并且写入缓冲器不可用，则卡会通过将 SDMMC_D 拉到低电平来重新激活忙碌指示。

35.4.6 块读取

在块读取模式中，数据传输的基本单位是块，其最大大小在 CSD (READ_BL_LEN) 中定义。如果将 READ_BL_PARTIAL 置 1，则还可以传输更小的块，这些块的开始和结束地址全部包含在一个物理块中（如 READ_BL_LEN 所定义）。在每个块的结尾会追加一个 CRC，以确保数据传输完整性。CMD17 (READ_SINGLE_BLOCK) 会启动块读取，在完成传输后，卡将返回到传输状态。

CMD18 (READ_MULTIPLE_BLOCK) 会启动多个连续块的传输。

主机在多个块操作中可以随时中止读取，无论其类型如何。通过发送停止传输命令可中止传输。

在多个块读取操作期间（全部两种类型），如果卡检测到错误（例如，超出范围、地址未对齐或内部错误），卡将停止数据传输并保持处于数据状态。然后，主机必须通过发送停止传输命令来中止操作。读取错误将在停止传输命令的响应中报告。

如果主机在卡以预定义的块数量传输了多个块操作的最后一个块之后发送了停止传输命令，则会以非法命令来响应它，因为卡已经不再处于数据状态。如果主机使用不完整的块（其累计长度没有进行块对齐）并且不允许块偏离，则卡会在第一个偏离块的开头检测块偏离错误条件（在状态寄存器中 ADDRESS_ERROR 错误位置 1）。

35.4.7 流访问、流写入和流读取（仅限多媒体卡）

在流模式中，数据以字节进行传输，并且不会在每个块的结尾追加 CRC。

流写入（仅限多媒体卡）

WRITE_DAT_UNTIL_STOP (CMD20) 会启动从 SDMMC 卡主机到卡的数据传输，从指定地址开始并继续，直到 SDMMC 卡主机发出停止命令为止。如果允许不完整的块（CSD 参数 WRITE_BL_PARTIAL 置 1），则数据库可以在卡地址空间内的任何地址开始和停止，否则只能在时钟边界处开始和停止。由于未提前确定要传输的数据量，因此无法使用 CRC。如果在发送数据时达到了存储器范围的结尾并且 SDMMC 卡主机未发出停止命令，则将丢弃已传输的任何多余的数据。

使用以下特定于卡的数据寄存器等式字段，可确定流写入操作的最大时钟频率：

$$\text{Maximumspeed} = \text{MIN}(\text{TRANSPEED}, \frac{(8 \times 2^{\text{writebllen}})(-\text{NSAC})}{\text{TAAC} \times \text{R2WFACTOR}})$$

- Maximumspeed = 最大写入频率
- TRANSPEED = 最大数据传输速率
- writebllen = 写入数据块最大长度
- NSAC = CLK 周期中数据读取访问时间 2
- TAAC = 数据读取访问时间 1
- R2WFACTOR = 写入速度系数

如果主机尝试使用更高的频率，卡可能无法处理数据并停止编程，在状态寄存器中将 OVERRUN 错误位置 1，同时忽略所有进一步数据传输，并在接收数据状态下等待停止命令。如果主机尝试在写保护区域进行写入，也将会中止写入操作。但是，在此情况下，卡会将 WP_VIOLATION 位置 1。

流读取（仅限多媒体卡）

READ_DAT_UNTIL_STOP (CMD11) 控制面向流的数据传输。

此命令指示卡从指定地址开始发送其数据，直到 SDMMC 卡主机发送 STOP_TRANSMISSION (CMD12) 为止。由于命令是串行传输，因此停止命令在执行时会有延迟，数据传输会在停止命令的结束位之后停止。如果在发送数据时达到了存储器范围的结尾并且 SDMMC 卡主机未发出停止命令，则发送的任何后续数据都被视为未定义。

使用以下特定于卡的数据寄存器等式字段，可确定流读取操作的最大时钟频率。

$$\text{Maximumspeed= MIN(TRANSPEED,}\frac{(8\times 2^{\text{readbllen}})(-\text{NSAC})}{\text{TAAC}\times \text{R2WFACTOR}}\text{)}$$

- Maximumspeed = 最大读取频率
- TRANSPEED = 最大数据传输速率
- readbllen = 读取数据块最大长度
- writebllen = 写入数据块最大长度
- NSAC = CLK 周期中数据读取访问时间 2
- TAAC = 数据读取访问时间 1
- R2WFACTOR = 写入速度系数

如果主机尝试使用更高的频率，则卡无法持续进行数据传输。如果发生此情况，卡会在状态寄存器中将 UNDERRUN 错误位置 1，中止传输并在数据状态下等待停止命令。

35.4.8 擦除：组擦除和扇区擦除

多媒体卡的可擦除单元是擦除组。擦除组以写入块来测量，这些写入块是卡的基本可写入单元。擦除组的大小是特定于卡的参数，在 CSD 中定义。

主机可以擦除某一连续范围内的擦除组。启动擦除是一个三步的过程。

首先，主机使用 ERASE_GROUP_START (CMD35) 命令定义范围的起始地址，然后使用 ERASE_GROUP_END (CMD36) 命令定义范围的最终地址，最后通过发出 ERASE (CMD38) 命令来启动擦除过程。擦除命令中的地址字段是以字节为单位的擦除组地址。卡将忽略小于擦除组大小的所有 LSB，从而有效地将地址向下取整为擦除组边界。

如果未按顺序接收擦除命令，则卡会在状态寄存器中将 ERASE_SEQ_ERROR 位置 1 并复位整个序列。

如果收到了一个乱序命令（两个都不是擦除命令，但 SEND_STATUS 除外），则卡会在状态寄存器中将 ERASE_RESET 状态位置 1、复位擦除序列并执行最后的命令。

如果擦除范围包括写保护块，则它们将保留不变，仅会擦除不受保护的块。状态寄存器中的 WP_ERASE_SKIP 状态位将会置 1。

卡通过将 SDMMC_D 保持低电平来指示擦除正在进行中。实际擦除时间可能很长，主机可以发出 CMD7 来取消选择卡。

35.4.9 宽总线选择或取消选择

使用 SET_BUS_WIDTH (ACMD6) 来选择或取消选择宽总线（4 位总线宽度）操作模式。加电或 GO_IDLE_STATE (CMD0) 后，默认总线宽度是 1 位。SET_BUS_WIDTH (ACMD6) 仅在传输状态下有效，这意味着只有在通过 SELECT/DESELECT_CARD (CMD7) 选择了卡之后才能更改总线宽度。

35.4.10 保护管理

SDMMC 卡主机模块中支持三种针对卡的写保护方法。

1. 卡内部写保护（卡的职责）
2. 机械写保护开关（仅由 SDMMC 卡主机模块负责）
3. 密码保护的卡锁定操作

卡内部写保护

可以防止对卡数据进行写入和擦除。通过在 CSD 中设置永久或临时写保护位，制造商或内容供应商可以将整个卡永久进行写保护。如果卡支持对扇区组进行写保护（通过将 CSD 中的 WP_GRP_ENABLE 位置 1），可以保护部分数据，并且写保护可以由应用程序进行更改。写保护是以 CSD 中指定的 WP_GRP_SIZE 扇区为单位。SET_WRITE_PROT 和 CLR_WRITE_PROT 命令控制编址组的保护。SEND_WRITE_PROT 命令类似于单个块读取命令。卡将发送一个数据块，其中包含 32 个写保护位（表示从指定地址开始的 32 个写保护组），后跟 16 个 CRC 位。写保护命令中的地址字段是以字节为单位的组地址。

卡将忽略小于组大小的所有 LSB。

机械写保护开关

一个位于卡侧面的机械拨片，用户可以利用它来设置或清除卡上的写保护。当拨片位于窗口打开的位置时，卡受到写保护，当窗口关闭时，可以更改卡内容。插座一侧的匹配开关向 SDMMC 卡主机模块表明，卡受到写保护。SDMMC 卡主机模块负责保护卡。卡的内部电路并不知道写保护开关的位置。

密码保护

密码保护功能使 SDMMC 卡主机模块可以使用密码来锁定和解锁卡。密码存储在 128 位 PWD 寄存器中，其大小在 8 位 PWD_LEN 寄存器中设置。这些寄存器具有非易失性，因此断电不会擦除它们。锁定的卡将响应和执行某些命令。这意味着 SDMMC 卡主机模块可以进行复位、初始化、选择以及查询状态，但是不允许访问卡的数据。设置了密码后（由 PWD_LEN 的非零值来指示），卡将在上电后自动锁定。与 CSD 和 CID 寄存器的写命令一样，仅在传输状态中才可以使用锁定/解锁命令。在此情况下，该命令不包括地址参数，必须首先选择卡然后才能使用。卡锁定/解锁命令具备常规单一块写入命令的结构和总线事务类型。传输的数据块包括命令的所有必需信息（密码设置模式、PWD 本身以及卡锁定/解锁）。命令数据块大小由 SDMMC 卡主机模块在发送卡锁定/解锁命令之前定义，命令数据块的结构如*表 213* 中所示。

位设置如下所示：

- ERASE：设置后会强制执行擦除操作。所有其它位都必须为零，并且仅发送命令字节
- LOCK_UNLOCK：置 1 后会锁定卡。LOCK_UNLOCK 可以与 SET_PWD 一起设置，但不能与 CLR_PWD 一起设置
- CLR_PWD：设置后会清除密码数据
- SET_PWD：设置后会将密码数据保存到存储器
- PWD_LEN：它定义密码的长度（以字节为单位）
- PWD：密码（新的或当前使用的，取决于命令）

以下各节列出了有关设置/复位密码、锁定/解锁卡以及强制擦除的命令。

设置密码

1. 如果未选择任何卡，请选择一个 (SELECT/DESELECT_CARD, CMD7)。
2. 定义要发送的块长度 (SET_BLOCKLEN, CMD16)（由 8 位卡锁定/解锁模式来确定）、8 位 PWD_LEN 以及新密码的字节数。完成了命令替换后，块大小必须考虑到如下情况：无论是旧密码还是新密码都会与命令一起发送。
3. 在数据线上，按相应的数据块大小发送 LOCK/UNLOCK (CMD42)，包括 16 位 CRC。数据块指示模式 (SET_PWD = 1)、长度 (PWD_LEN) 和密码自身 (PWD)。完成密码替换后，长度值 (PWD_LEN) 会包括新旧两个密码的长度，PWD 字段包括旧密码（当前使用的）后跟新密码。
4. 如果密码匹配，则新密码及其大小会分别保存到 PWD 和 PWD_LEN 字段。如果发送的旧密码与预期的密码不符（大小和/或内容），会在卡状态寄存器中将 LOCK_UNLOCK_FAILED 错误位置 1，而密码不会更改。

密码长度字段 (PWD_LEN) 指示当前是否设置了密码。如果此字段非零，则表示设置了密码并且卡在上电后锁定自身。通过将 LOCK_UNLOCK 位置 1（在设置密码时）或通过发送用于卡锁定的其它命令，可以在当前上电会话中立即锁定卡。

复位密码

1. 如果未选择任何卡，请选择一个 (SELECT/DESELECT_CARD, CMD7)。
2. 定义要发送的块长度 (SET_BLOCKLEN, CMD16)（由 8 位卡锁定/解锁模式来确定）、8 位 PWD_LEN 以及当前使用的密码的字节数。
3. 在数据线上，按相应的数据块大小发送 LOCK/UNLOCK (CMD42)，包括 16 位 CRC。数据块指示模式 (CLR_PWD = 1)、长度 (PWD_LEN) 和密码自身 (PWD)。将忽略 LOCK_UNLOCK 位。
4. 如果密码匹配，会清除 PWD 字段并将 PWD_LEN 置 0。如果发送的密码与预期的密码不符（大小和/或内容），会在卡状态寄存器中将 LOCK_UNLOCK_FAILED 错误位置 1，而密码不会更改。

锁定卡

1. 如果未选择任何卡，请选择一个 (SELECT/DESELECT_CARD, CMD7)。
2. 定义要发送的块长度 (SET_BLOCKLEN, CMD16)（由 8 位卡锁定/解锁模式来确定，请参见*表 213* 中的字节 0）、8 位 PWD_LEN 以及当前密码的字节数。
3. 在数据线上，与相应的数据块大小一起发送 LOCK/UNLOCK (CMD42)，包括 16 位 CRC。数据块指示模式 (LOCK_UNLOCK = 1)、长度 (PWD_LEN) 和密码自身 (PWD)。
4. 如果密码匹配，则卡被锁定并且卡状态寄存器中 CARD_IS_LOCKED 状态位会置 1。如果发送的密码与预期的密码不符（大小和/或内容），会在卡状态寄存器中将 LOCK_UNLOCK_FAILED 错误位置 1，且锁定会失败。

可以按照相同的顺序来设置密码和锁定卡。在此情况下，SDMMC 卡主机模块会执行用于设置密码的所有必需步骤（请参见*第 1110 页上的设置密码*），但是发送新密码命令时，需要在步骤 3 中将 LOCK_UNLOCK 位置 1。

如果之前设置过密码（PWD_LEN 不为 0），则卡在上电复位后会自动锁定。尝试锁定已锁定的卡或者尝试锁定没有密码的卡都会失败，并且会在卡状态寄存器中将 LOCK_UNLOCK_FAILED 错误位置 1。

解锁卡

1. 如果未选择任何卡，请选择一个 (SELECT/DESELECT_CARD, CMD7)。
2. 定义要发送的块长度 (SET_BLOCKLEN, CMD16)（由 8 位卡锁定/解锁模式来确定，请参见*表 213* 中的字节 0）、8 位 PWD_LEN 以及当前密码的字节数。
3. 在数据线上，按相应的数据块大小发送 LOCK/UNLOCK (CMD42)，包括 16 位 CRC。数据块指示模式 (LOCK_UNLOCK = 0)、长度 (PWD_LEN) 和密码自身 (PWD)。
4. 如果密码匹配，则卡被解锁并且卡状态寄存器中 CARD_IS_LOCKED 状态位会清零。如果发送的密码在大小和/或内容方面不正确并且与预期的密码不符，会在卡状态寄存器中将 LOCK_UNLOCK_FAILED 错误位置 1，且卡保持锁定。

解锁功能仅对当前上电会话有效。如果 PWD 字段未清零，则卡会在下一次上电时自动锁定。

尝试解锁已解锁的卡会导致失败，并且会在卡状态寄存器中将 LOCK_UNLOCK_FAILED 错误位置 1。

强制擦除

如果用户忘记了密码（PWD 内容），可以在清除卡上所有数据之后访问卡。这种强制擦除操作将擦除所有卡数据和所有密码数据。

1. 如果未选择任何卡，请选择一个 (SELECT/DESELECT_CARD, CMD7)。
2. 将块长度 (SET_BLOCKLEN, CMD16) 设置为 1 个字节。仅发送 8 位卡锁定/解锁字节（*表 213* 中的字节 0）。
3. 在数据线上，按相应的数据字节发送 LOCK/UNLOCK (CMD42)，包括 16 位 CRC。数据块指示模式 (ERASE = 1)。所有其它位都必须为零。
4. 如果数据字段中仅设置了 ERASE 位，会擦除所有卡内容，包括 PWD 和 PWD_LEN 字段，并且卡不再锁定。如果设置了任何其它位，则会在卡状态寄存器中将 LOCK_UNLOCK_FAILED 错误位置 1，而卡会保留其所有数据并保持锁定。

尝试在锁定的卡上使用强制擦除会导致失败，并且会在卡状态寄存器中将 LOCK_UNLOCK_FAILED 错误位置 1。

35.4.11 卡状态寄存器

响应格式 R1 包含一个由 32 位字段命名的卡状态。此字段旨在将卡状态信息（可能存储在本地状态寄存器中）传输到主机。除非另有说明，否则状态条目始终与先前发出的命令有关。

表 200 定义了状态的不同条目。表中的类型和清除条件字段以如下方式缩写：

类型：

- E：错误位
- S：状态位
- R：已检测到并已针对实际命令响应进行了设置
- X：已检测到并在命令执行期间进行了设置。SDMMC 卡主机要对卡进行轮询必须发出状态命令来读取这些位。

清除条件：

- A：根据卡的当前状态
- B：始终与先前命令有关。收到有效命令后会将其清零（延迟一个命令）
- C：通过读取清零

表 200. 卡状态

位	标识符	类型	值	说明	清除条件
31	ADDRESS_ OUT_OF_RANGE	E R X	“0” = 无错误 “1” = 有错误	命令地址参数超出了此卡允许的范围。 多个块或流读取/写入操作正在尝试进行超出卡容量的读取或写入（尽管是从有效地址开始）。	C
30	ADDRESS_MISALIGN		“0” = 无错误 “1” = 有错误	命令地址参数（与当前设置的块长度一致）会将第一个偏离的数据块定位到卡的物理块。 多个块读取/写入操作正在尝试读取或写入与卡的物理块未对齐的数据块（尽管是以有效的地址/块长度组合开始）。	C
29	BLOCK_LEN_ERROR		“0” = 无错误 “1” = 有错误	可能是 SET_BLOCKLEN 命令的参数超过了卡的最大允许值，或者先前定义的块长度对当前命令而言非法（例如，主机发出了写入命令，而当前块长度小于卡的最大允许值并且不允许写入不完整的块）。	C
28	ERASE_SEQ_ERROR		“0” = 无错误 “1” = 有错误	擦除命令序列中发生错误。	C
27	ERASE_PARAM	E X	“0” = 无错误 “1” = 有错误	为擦除选择的擦除组无效。	C
26	WP_VIOLATION	E X	“0” = 无错误 “1” = 有错误	尝试对写保护块进行编程。	C
25	CARD_IS_LOCKED	S R	“0” = 表示卡已解锁 “1” = 卡已锁定	如果设置，则表示卡已由主机锁定	A
24	LOCK_UNLOCK_ FAILED	E X	“0” = 无错误 “1” = 有错误	如果在锁定/解锁卡命令中检测到序列或密码错误，则会设置。	C
23	COM_CRC_ERROR	E R	“0” = 无错误 “1” = 有错误	先前命令的 CRC 校验失败。	B
22	ILLEGAL_COMMAND	E R	“0” = 无错误 “1” = 有错误	命令对于卡状态不合法。	B
21	CARD_ECC_FAILED	E X	“0” = 成功 “1” = 失败	应用了卡内部 ECC，但未能更正数据。	C
20	CC_ERROR	E R	“0” = 无错误 “1” = 有错误	（标准中未定义）发生了卡错误，该错误与主机命令无关。	C
19	ERROR	E X	“0” = 无错误 “1” = 有错误	（标准中未定义）与最后一个主机命令的执行（以及在此期间的检测）有关的一般卡错误（例如，读取或写入失败）。	C
18	保留				
17	保留				
16	CID/CSD_OVERWRITE	E X	“0” = 无错误， “1” = 有错误	可以是以下错误之一： – CID 寄存器已写入且无法覆盖 – CSD 的只读部分与卡内容不匹配 – 尝试恢复已设置的副本（设置为原始值）或永久 WP（未受保护）位	C

表 200. 卡状态（续）

位	标识符	类型	值	说明	清除条件
15	WP_ERASE_SKIP	E X	“0” = 未受保护 “1” = 受保护	仅当由于现有写入而导致部分地址空间被擦除时才会设置。	C
14	CARD_ECC_DISABLED	S X	“0” = 使能 “1” = 禁用	命令已在未使用内部 ECC 的情况下执行。	A
13	ERASE_RESET		“0” = 已清零 “1” = 已置 1	在执行之前已清除了擦除序列，因为收到的不是擦除序列命令（CMD35、CMD36、CMD38 或 CMD13 之外的命令）。	C
12:9	CURRENT_STATE	S R	0 = 空闲 1 = 就绪 2 = 识别 3 = 待机 4 = 传输 5 = 数据 6 = 接收 7 = 进行中 8 = 断开 9 = 突发 10-15 = 保留	接收命令时卡的状态。如果命令执行导致了状态变化，主机可以在下一个命令的响应中看到。这四个位将被解释为 0 到 15 之间的二进制数字。	B
8	READY_FOR_DATA	S R	“0” = 未就绪， “1” = 就绪	对应于总线上的缓冲器空信令。	-
7	SWITCH_ERROR	E X	“0” = 无错误 “1” = 开关错误	如果设置，则卡不会根据 SWITCH 命令的请求切换到预期模式。	B
6	保留				
5	APP_CMD	S R	“0” = 禁用 “1” = 使能	卡将预期收到 ACMD，或者命令已解释为 ACMD 的指示。	C
4	保留，用于 SD I/O 卡				
3	AKE_SEQ_ERROR	E R	“0” = 无错误 “1” = 有错误	认证过程序列中出现错误。	C
2	保留，用于应用程序特定的命令				
1	保留，用于制造商测试模式				
0					

35.4.12 SD 状态寄存器

SD 状态包含与 SD 存储卡专有功能相关的状态位，在将来可用于特定于应用的场合中。SD 状态的大小是一个 512 位的数据块。如果发送了 ACMD13（CMD55，后面带有 CMD13），此寄存器的内容将传输到 SDMMC 卡主机。ACMD13 只能发送到处于传输状态的卡（卡已被选定）。

表 201 定义了 SD 状态寄存器的不同条目。表中的类型和清除条件字段以如下方式缩写：

类型：

- E：错误位
- S：状态位
- R：已检测到并已针对实际命令响应进行了设置
- X：已检测到并在命令执行期间进行了设置。SDMMC 卡主机要对卡进行轮询必须发出状态命令来读取这些位

清除条件：

- A：根据卡的当前状态
- B：始终与先前命令有关。收到有效命令后会将其清零（延迟一个命令）
- C：通过读取清零

表 201. SD 状态

位	标识符	类型	值	说明	清除条件
511: 510	DAT_BUS_WIDTH	S R	“00” = 1（默认值） “01” = 保留 “10” = 4 位宽度 “11” = 保留	显示了当前定义的数据总线宽度（由 SET_BUS_WIDTH 命令定义）。	A
509	SECURED_MODE	S R	“0” = 未处于安全模式 “1” = 处于安全模式	卡处于安全操作模式（请参见“SD 安全规范”）。	A
508: 496	保留				
495: 480	SD_CARD_TYPE	S R	“00xxh” = SD 存储卡，按照物理规范版本 1.01-2.00 中所定义（“x” = 无关）。目前定义了以下卡： “0000” = 常规 SD RD/WR 卡。 “0001” = SD ROM 卡	将来，8 个 LSB 将用于定义 SD 存储卡的不同变体（每个位将定义不同的 SD 类型）。8 个 MSB 将用于定义不符合当前 SD 物理层规范的 SD 卡。	A
479: 448	SIZE_OF_PROTECT ED_AREA	S R	受保护区域的大小（请参见以下内容）	（请参见以下内容）	A
447: 440	SPEED_CLASS	S R	卡的速度等级（请参见以下内容）	（请参见以下内容）	A
439: 432	PERFORMANCE_MOVE	S R	按 1 [MB/s] 步长表示的移动性能。（请参见以下内容）	（请参见以下内容）	A
431:428	AU_SIZE	S R	AU 大小 （请参见以下内容）	（请参见以下内容）	A
427:424	保留				
423:408	ERASE_SIZE	S R	一次要擦除的 AU 数	（请参见以下内容）	A
407:402	ERASE_TIMEOUT	S R	擦除 UNIT_OF_ERASE_AU 指定的区域时的超时值	（请参见以下内容）	A

表 201. SD 状态（续）

位	标识符	类型	值	说明	清除条件
401:400	ERASE_OFFSET	S R	为擦除时间增加的固定偏移值。	（请参见以下内容）	A
399:312	保留				
311:0	保留，用于制造商				

SIZE_OF_PROTECTED_AREA

此字段的设置在标准容量卡和高容量卡之间有所不同。如果是标准容量卡，受保护区域的容量计算方式如下：

受保护区域 = SIZE_OF_PROTECTED_AREA_* MULT * BLOCK_LEN。

SIZE_OF_PROTECTED_AREA 是以 MULT*BLOCK_LEN 为单位来指定。

如果是高容量卡，受保护区域的容量在以下字段中指定：

受保护区域 = SIZE_OF_PROTECTED_AREA

SIZE_OF_PROTECTED_AREA 是以字节为单位来指定。

SPEED_CLASS

此 8 位字段指示速度等级，该值可以通过 $P_W/2$ 来计算（其中 P_W 是写入性能）。

表 202. 速度等级代码字段

SPEED_CLASS	值定义
00h	等级 0
01h	等级 2
02h	等级 4
03h	等级 6
04h – FFh	保留

PERFORMANCE_MOVE

此 8 位字段指示 Pm（移动性能），该值可通过 1 [MB/s] 步长进行设置。如果卡不移动已使用的 RU（记录单元），则 Pm 应被视为无穷大。将该字段设置为 FFh 即表示无穷大。

表 203. 移动性能字段

PERFORMANCE_MOVE	值定义
00h	未定义
01h	1 [MB/s]
02h	02h 2 [MB/s]
---------	---------
FEh	254 [MB/s]
FFh	无穷大

AU_SIZE

该 4 位字段指示 AU 大小，其取值范围是 2 的 n 次方（起始值为 16KB）。

表 204. AU_SIZE 字段

AU_SIZE	值定义
00h	未定义
01h	16 KB
02h	32 KB
03h	64 KB
04h	128 KB
05h	256 KB
06h	512 KB
07h	1 MB
08h	2 MB
09h	4 MB
Ah – Fh	保留

AU 的最大大小，具体取决于卡的容量（在*表 205* 中进行了定义）。可将卡的容量大小设置为介于 RU 大小和最大 AU 大小之间的任何 AU 大小。

表 205. 最大 AU 大小

容量	16 MB-64 MB	128 MB-256 MB	512 MB	1 GB-32 GB
最大 AU 大小	512 KB	1 MB	2 MB	4 MB

ERASE_SIZE

该 16 位字段指示 NERASE。擦除 NERASE 个 AU 后，将由 ERASE_TIMEOUT 指定超时值（请参见*ERASE_TIMEOUT*）。主机应确定要在一个操作中擦除的正确 AU 数量，以便主机可以显示擦除操作的进度。如果将该字段设置为 0，则不支持擦除超时计算。

表 206. 擦除大小字段

ERASE_SIZE	值定义
0000h	不支持擦除超时计算。
0001h	1 个 AU
0002h	2 个 AU
0003h	3 个 AU
---------	---------
FFFFh	65535 个 AU

ERASE_TIMEOUT

该 6 位字段指示 TERASE，该值指示当擦除 ERASE_SIZE 所指定的多个 AU 时，与偏移量之间的擦除超时。ERASE_TIMEOUT 的取值范围最大可定义为 63 秒，卡制造商可以选择 ERASE_SIZE 和 ERASE_TIMEOUT 的任意组合，具体取决于实现情况。确定 ERASE_TIMEOUT 即确定了 ERASE_SIZE。

表 207. 擦除超时字段

ERASE_TIMEOUT	值定义
00	不支持擦除超时计算。
01	1 [秒]
02	2 [秒]
03	3 [秒]
---------	---------
63	63 [秒]

ERASE_OFFSET

该 2 位字段指示 TOFFSET，可选择四个值之一。如果 ERASE_SIZE 和 ERASE_TIMEOUT 字段设置为 0，则该字段无意义。

表 208. 擦除偏移字段

ERASE_OFFSET	值定义
0h	0 [秒]
1h	1 [秒]
2h	2 [秒]
3h	3 [秒]

35.4.13 SD I/O 模式

SD I/O 中断

为使 SD I/O 卡能够中断多媒体卡/SD 模块，在 SD 接口的一个引脚上提供了一个中断功能。引脚 8（当以 4 位 SD 模式运行时用作 SDMMC_D1）可以向多媒体卡/SD 模块发出卡中断信号。每个卡或者卡中的功能都可以使用中断。SD I/O 中断为电平敏感型，这意味着，在多媒体卡/SD 模块识别出中断线并对其采取操作或者由于中断周期结束而使其失效之前，中断线必须保持有效状态（低电平）。在多媒体卡/SD 模块处理完中断后，将通过在 SD I/O 卡内部寄存器中的相应位中进行 I/O 写入来清除中断状态位。所有 SD I/O 卡的中断输出均为低电平有效，并且应用程序必须在所有数据线 (SDMMC_D[3:0]) 上提供外部上拉电阻。只有在中断周期内，多媒体卡/SD 模块才会将引脚 8 (SDMMC_D/IRQ) 的电平仅采样到中断检测器中。而在所有其他时候，多媒体卡/SD 模块将忽略此值。

中断周期适用于存储器和 I/O 操作。对于具有单个块的操作和多块数据传输，二者的中断周期定义有所不同。

SD I/O 挂起和恢复

在多功能 SD I/O 中或者在同时具备 I/O 和存储器功能的卡中，有多个设备（I/O 和存储器）共享对 MMC/SD 总线的访问权限。要在多个设备之间共享对 MMC/SD 模块的访问权限，SD I/O 和组合卡可以有选择性地实施“挂起/恢复”这一概念。如果卡支持挂起/恢复，则 MMC/SD 模块可以临时停止对某一功能或存储器的数据传输操作（挂起）以释放总线，从而将总线用于其他功能或存储器的更高优先级传输。此更高优先级传输完成后，原始传输将从中断位置恢复（重新开始）。是否支持“挂起/恢复”功能具体取决于卡。要在 MMC/SD 总线上执行挂起/恢复，MMC/SD 模块将执行以下步骤：

1. 确定当前使用 SDMMC_D [3:0] 线的功能
2. 请求将优先级较低或速度较慢的事务挂起
3. 等待事务挂起完成
4. 开始更高优先级的事务
5. 等待更高优先级事务完成
6. 恢复挂起的事务

SD I/O 读取等待

可选的读取等待 (RW) 操作仅针对 SD 的 1 位和 4 位模式进行了定义。“读取等待”操作允许 MMC/SD 模块发出如下信号：卡正在读取多个寄存器 (IO_RW_EXTENDED, CMD53) 以临时停止数据传输，同时允许 MMC/SD 模块向 SD I/O 设备中的任何功能发送命令。要确定卡是否支持“读取等待”协议，MMC/SD 模块必须测试内部卡寄存器中的功能位。“读取等待”的计时是基于中断周期的。

35.4.14 命令和响应

应用程序特定的命令和常规命令

SDMMC 卡主机模块系统旨在为各种应用程序提供一个标准接口。在此环境中，需要有特定的客户/应用程序功能。为实现这些功能，在标准中定义了两种类型的通用命令：应用程序特定的命令 (ACMD) 和常规命令 (GEN_CMD)。

当卡接收到 APP_CMD (CMD55) 命令时，卡所需的下一个命令是应用程序特定的命令。ACMD 与常规多媒体卡命令的结构相同，且 CMD 编号也可以相同。卡将其识别 ACMD 的依据是：它出现在 APP_CMD (CMD55) 的后面。如果后面紧跟 APP_CMD (CMD55) 的命令不是已定义的应用程序特定的命令，则会使用标准命令。例如，如果卡具有 SD_STATUS (ACMD13) 的定义并且接收到了后跟 APP_CMD (CMD55) 的 CMD13，则此命令被解释为 SD_STATUS (ACMD13)。但是，如果卡接收到后跟 APP_CMD (CMD55) 的 CMD7，但是卡没有 ACMD7 的定义，则此命令被解释为标准的 (SELECT/DESELECT_CARD) CMD7。

要使用制造商特定的 ACMD，SD 卡主机必须执行以下步骤：

1. 发送 APP_CMD (CMD55)
   卡对多媒体卡/SD 模块进行响应，指示已将 APP_CMD 位置 1，并指示当前需要的是 ACMD。
2. 发送所需的 ACMD
   卡对多媒体卡/SD 模块进行响应，指示已将 APP_CMD 位置 1，并指示接受的命令被解释为 ACMD。如果发送非 ACMD，则卡会将其作为普通多媒体卡命令进行处理，卡状态寄存器中的 APP_CMD 位保持清零。

如果发送了无效命令（既不是 ACMD 也不是 CMD），则将作为标准的多媒体卡非法命令错误来处理。

GEN_CMD 的总线事务与单块读/写命令（WRITE_BLOCK, CMD24 或 READ_SINGLE_BLOCK,CMD17）的总线事务相同。在此情况下，参数表示数据传输的方向而非地址，数据块具有供应商特定的格式和含义。

发送 GEN_CMD (CMD56) 之前，必须选择卡（处于传输状态）。数据块大小由 SET_BLOCKLEN (CMD16) 定义。对 GEN_CMD (CMD56) 的响应为 R1b 格式。

命令类型

应用程序特定的命令和常规命令均具有以下四种类型：

- **广播命令 (BC)**：发送到所有卡；不返回任何响应。
- **具有响应的广播命令 (BCR)**：发送到所有卡；同时接收来自所有卡的响应。
- **编址（点到点）命令 (AC)**：发送到选定的卡；不包含 SDMMC_D 线上的数据传输。
- **编址（点到点）数据传输命令 (ADTC)**：发送到选定的卡；包含 SDMMC_D 线上的数据传输。

命令格式

请参见*第 1097 页上的表 192* 以了解命令格式。

多媒体卡/SD 模块的命令

表 209. 面向块的写入命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD23	ac	[31:16] 设置为 0 [15:0] 块数量	R1	SET_BLOCK_COUNT	用于定义将在后面的多块读/写命令中传输的块数量。
CMD24	adtc	[31:0] 数据地址	R1	WRITE_BLOCK	写入一个块，可通过 SET_BLOCKLEN 命令选择其大小。
CMD25	adtc	[31:0] 数据地址	R1	WRITE_MULTIPLE_BLOCK	连续写入数据块，直至出现 STOP_TRANSMISSION，或者直至已接收到所请求的块数量。
CMD26	adtc	[31:0] 填充位	R1	PROGRAM_CID	对卡标识寄存器进行编程。每个卡只能发出一次该命令。卡包含的硬件可防止在首次编程后执行此操作。通常该命令留给供制造商使用。
CMD27	adtc	[31:0] 填充位	R1	PROGRAM_CSD	对 CSD 的可编程位进行编程。

表 210. 面向块的写保护命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD28	ac	[31:0] 数据地址	R1b	SET_WRITE_PROT	如果卡具有写保护功能，此命令会将编址组的写保护位置 1。写保护的属性以卡特定的数据 (WP_GRP_SIZE) 进行编码。
CMD29	ac	[31:0] 数据地址	R1b	CLR_WRITE_PROT	如果卡提供了写保护功能，此命令会将编址组的写保护位清零。
CMD30	adtc	[31:0] 写保护数据地址	R1	SEND_WRITE_PROT	如果卡提供了写保护功能，此命令将要求卡发送写保护位的状态。
CMD31	保留				

表 211. 擦除命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD32 ... CMD34	保留。不能使用这些命令索引来保持与较旧版本多媒体卡的向后兼容性。				
CMD35	ac	[31:0] 数据地址	R1	ERASE_GROUP_START	在要选择进行擦除的某个范围内设置第一个擦除组的地址。
CMD36	ac	[31:0] 数据地址	R1	ERASE_GROUP_END	在要选择进行擦除的某个连续范围内设置最后一个擦除组的地址。
CMD37	保留。不能使用此命令索引来保持与较旧版本多媒体卡的向后兼容性。				
CMD38	ac	[31:0] 填充位	R1	ERASE	擦除所有先前选择的写入块。

表 212. I/O 模式命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD39	ac	[31:16] RCA [15:15] 寄存器写入标志 [14:8] 寄存器地址 [7:0] 寄存器数据	R4	FAST_IO	用于写入和读取 8 位（寄存器）数据字段。该命令用于对卡和寄存器寻址，如果将写入标志置 1，则提供要写入的数据。R4 响应包含从寻址寄存器中读取的数据。该命令可访问与应用程序相关、未在多媒体卡标准中定义的寄存器。
CMD40	bcr	[31:0] 填充位	R5	GO_IRQ_STATE	使系统处于中断模式。
CMD41	保留				

表 213. 锁定卡

CMD 索引	类型	参数	响应格式	缩写	说明
CMD42	adtc	[31:0] 填充位	R1b	LOCK_UNLOCK	设置/重置密码或锁定/解锁卡。数据块的大小由 SET_BLOCK_LEN 命令设置。
CMD43 ... CMD54	保留				

表 214. 应用程序特定的命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD55	ac	[31:16] RCA [15:0] 填充位	R1	APP_CMD	告知卡下一个命令位是应用程序特定的命令，而非标准命令。
CMD56	adtc	[31:1] 填充位 [0]：RD/WR	-	-	用于将数据块传输到卡，或从卡中获取数据块以用于常规命令/应用程序特定的命令。数据块的大小由 SET_BLOCK_LEN 命令设置。
CMD57 ... CMD59	保留。				
CMD60 ... CMD63	留给制造商使用。				

35.5 响应格式

所有响应都是通过 SDMMC 命令线 SDMMC_CMD 来发送。响应传输始终从对应于响应代码字的位字符串的左侧位开始。代码长度取决于响应类型。

响应始终从起始位（始终为 0）开始，后跟用于指示传输方向的位 (card = 0)。下表中标有 x 的值表示变量项。除 R3 响应类型之外的所有响应均受 CRC 保护。每个命令代码字由结束位（始终为 1）来终止。

有五种类型的响应。其格式定义如下：

35.5.1 R1（正常响应命令）

代码长度 = 48 位。45:40 位用于指示待响应命令的索引，该值被解释为二进制编码的数字（介于 0 和 63 之间）。卡的状态采用 32 位进行编码。

表 215. R1 响应

位的位置	宽度（位）	值	说明
47	1	0	起始位
46	1	0	传输位
[45:40]	6	X	命令索引 (Command index)
[39:8]	32	X	卡状态
[7:1]	7	X	CRC7
0	1	1	结束位

35.5.2 R1b

同 R1，并且有一个可选的繁忙信号在数据线上传输。根据卡在接收命令之前所处的状态，卡在接收到这些命令之后可能会变为繁忙状态。

35.5.3 R2（CID 和 CSD 寄存器）

代码长度 = 136 位。发送 CID 寄存器的内容以响应 CMD2 和 CMD10 命令。发送 CSD 寄存器的内容以响应 CMD9。仅传输 CID 和 CSD 的位 [127...1]，这些寄存器的保留位 [0] 将被替换为响应的结束位。卡通过将 SDMMC_D0 保持低电平来指示擦除正在进行中。实际擦除时间可能很长，主机可以发出 CMD7 来取消选择卡。

表 216. R2 响应

位的位置	宽度（位）	值	说明
135	1	0	起始位
134	1	0	传输位
[133:128]	6	‘111111’	命令索引 (Command index)
[127:1]	127	X	卡状态
0	1	1	结束位

35.5.4 R3（OCR 寄存器）

代码长度：48 位。发送 OCR 寄存器的内容以响应 CMD1。电平编码如下所示：受限电压窗口 = 低电平，卡繁忙 = 低电平。

表 217. R3 响应

位的位置	宽度（位）	值	说明
47	1	0	起始位
46	1	0	传输位
[45:40]	6	‘111111’	保留
[39:8]	32	X	OCR 寄存器
[7:1]	7	‘1111111’	保留
0	1	1	结束位

35.5.5 R4（快速 I/O）

代码长度：48 位。参数字段包含寻址卡的 RCA、要读出或写入的寄存器地址及其内容。

表 218. R4 响应

位的位置		宽度（位）	值	说明
47		1	0	起始位
46		1	0	传输位
[45:40]		6	‘100111’	CMD39
[39:8] 参数字段	[31:16]	16	X	RCA
	[15:8]	8	X	寄存器地址
	[7:0]	8	X	读取寄存器内容
[7:1]		7	X	CRC7
0		1	1	结束位

35.5.6 R4b

仅限 SD I/O：接收 CMD5 的 SDIO 卡将以唯一的 SDIO 响应 R4 进行响应。格式为：

表 219. R4b 响应

位的位置		宽度（位）	值	说明
47		1	0	起始位
46		1	0	传输位
[45:40]		6	X	保留
[39:8] 参数字段	39	16	X	卡就绪
	[38:36]	3	X	I/O 功能的数量
	35	1	X	存在存储器
	[34:32]	3	X	填充位
	[31:8]	24	X	I/O ORC

表 219. R4b 响应（续）

位的位置	宽度（位）	值	说明
[7:1]	7	X	保留
0	1	1	结束位

一旦 SD I/O 卡接收到 CMD5，即会使能该卡的 I/O 部分以正常响应所有后续命令。I/O 卡中的此功能 I/O 使能将始终保持有效，直至卡接收到复位、电源重启或对 I/O 复位执行写入操作的 CMD52。请注意，仅具存储器的 SD 卡可对 CMD5 进行响应。仅具存储器的卡的正确响应将为：*存在存储器* = 1，*I/O 功能的数量* = 0。仅具存储器的卡（其设计标准符合 SD 存储卡规范版本 1.0）会将 CMD5 视为非法命令，并且不予响应。可感知 I/O 的主机将发送 CMD5。如果卡以响应 R4 进行响应，则主机会根据 R4 响应中包含的数据来确定卡的配置。

35.5.7 R5（中断请求）

仅用于多媒体卡。代码长度：48 位。如果响应由主机生成，则参数中的 RCA 字段将为 0x0。

表 220. R5 响应

位的位置		宽度（位）	值	说明
47		1	0	起始位
46		1	0	传输位
[45:40]		6	‘101000’	CMD40
[39:8] 参数字段	[31:16]	16	X	胜出的卡或主机的 RCA [31:16]
	[15:0]	16	X	未定义。可用于 IRQ 数据
[7:1]		7	X	CRC7
0		1	1	结束位

35.5.8 R6

仅用于 SD I/O。存储设备对 CMD3 的正常响应。相关内容在*表 221* 中进行了说明。

表 221. R6 响应

位的位置		宽度（位）	值	说明
47		1	0	起始位
46		1	0	传输位
[45:40]		6	‘101000’	CMD40
[39:8] 参数字段	[31:16]	16	X	胜出的卡或主机的 RCA [31:16]
	[15:0]	16	X	未定义。可用于 IRQ 数据
[7:1]		7	X	CRC7
0		1	1	结束位

当 CMD3 发送到 I/O 专用卡时，卡的状态位 [23:8] 将更改。在此情况下，响应的 16 位代表仅限 SD I/O 的值：

- 位 [15] COM_CRC_ERROR
- 位 [14] ILLEGAL_COMMAND
- 位 [13] ERROR
- 位 [12:0] 保留

35.6 SDIO I/O 卡专用操作

以下功能是专用于 SD I/O 的操作：

- 通过触发 SDMMC_D2 执行的 SDIO 读取等待操作
- 通过停止时钟执行的 SDIO 读取等待操作
- SDIO 挂起/恢复操作（写入和读取挂起）
- SDIO 中断

仅当 SDMMC_DCTRL[11] 位置 1 时，SDMMC 才支持这些操作，但读取挂起操作除外，该操作不需要专门的硬件实现。

35.6.1 通过触发 SDMMC_D2 执行的 SDIO I/O 读取等待操作

可以在收到第一个数据块前启动读取等待间隔：如果使能数据路径（SDMMC_DCTRL[0] 位置 1）并使能 SDIO 专用操作（SDMMC_DCTRL[11] 位置 1），则当读取等待开始（SDMMC_DCTRL[10] =0 且 SDMMC_DCTRL[8] =1）并且数据方向为从卡到 SDMMC (SDMMC_DCTRL[1] = 1) 时，DPSM 直接从空闲状态进入读取等待状态。在读取等待状态下，DPSM 在 2 个 SDMMC_CK 时钟周期后将 SDMMC_D2 驱动为 0。在此状态下，如果将 RWSTOP 位 (SDMMC_DCTRL[9]) 置 1，DPSM 保持等待状态的时间将多出两个 SDMMC_CK 时钟周期，以将 SDMMC_D2 驱动为 1 并保持一个时钟周期（根据 SDIO 规范）。之后，DPSM 重新开始等待，直至从卡中接收到数据。当 DPSM 接收到数据块时，即使读取等待开始位置 1，也不会启动读取等待间隔：读取等待间隔将在接收到 CRC 后启动。必须将 RWSTOP 位清零，才能开始新的读取等待操作。在读取等待间隔内，SDMMC 可以在 SDMMC_D1 上检测 SDIO 中断。

35.6.2 通过停止 SDMMC_CK 执行的 SDIO 读取等待操作

如果 SDIO 卡不支持上述读取等待方法，SDMMC 可以通过停止 SDMMC_CK 来执行读取等待（SDMMC_DCTRL 置 1，这一点与*第 35.6.1 节*所述方法相同，但 SDMMC_DCTRL[10] =1）：在当前接收数据块的结束位后，DSPM 将在两个 SDMMC_CK 周期后停止时钟，并在读取等待开始位置 1 后再次启动时钟。

SDMMC_CK 停止时，可以向卡发出任何命令。在读取/等待间隔期间，SDMMC 可以在 SDMMC_D1 上检测 SDIO 中断。

35.6.3 SDIO 挂起/恢复操作

向卡发送数据时，SDMMC 可以挂起写操作。SDMMC_CMD[11] 位将置 1 并向 CPSM 指示当前命令为挂起命令。CPSM 将分析响应，一旦从卡接收到 ACK（接受挂起）信号，即确认 CPSM 在接收到当前数据块的 CRC 令牌后将变为空闲状态。

硬件不会保存要完成挂起的操作（恢复）还需发送的剩余数据块数量。

写操作可以由软件挂起，只需在从卡接收到挂起命令的 ACK 信号时禁止 DPSM (SDMMC_DCTRL[0] =0)。DPSM 随即进入空闲状态。

要挂起读取操作：DPSM 将以 Wait_r 状态进行等待，因为要挂起的功能会在停止数据事务前发送一个完整的数据包。应用程序将继续读取 RxFIFO 直至 FIF0 为空，然后 DPSM 自动进入空闲状态。

35.6.4 SDIO 中断

一旦 SDMMC_DCTRL[11] 位置 1，即会在 SDMMC_D1 线路上检测到 SDIO 中断。

检测到 SDIO 中断时，SDMMC_STA[22] (SDIOIT) 位将置 1。此静态位可通过清零位 SDMMC_ICR[22] (SDIOITC) 来清零。如果 SDIOIT 位置 1，可产生中断。单独的中断使能 SDMMC_MASK[22] 位 (SDIOITE) 可用于使能和禁止中断请求。

发生 SD 卡中断（SDMMC_STA[22] 位置 1）时，主机软件将按以下步骤处理。

1. 将 SDIOITE 位清零（SDMMC_MASK[22] = “0”）以禁止 SDIOIT 中断信号；
2. 处理卡中断请求，并清除 SD 卡上的中断源；
3. 向 SDIOITC 位写入“1”（SDMMC_ICR[22] = “1”）以将 SDIOIT 位清零；
4. 向 SDIOITE 位写入“1”（SDMMC_MASK[22] = “1”）以使能 SDIOIT 中断信号。

可以在中断服务例程外执行步骤 2 到步骤 4。

35.7 硬件流控制

硬件流控制功能用于避免 FIFO 下溢（发送模式）和上溢（接收模式）错误。

该功能可停止 SDMMC_CK 并冻结 SDMMC 状态机。数据传输将停止，而 FIFO 无法发送或接收数据。只有由 SDMMCCLK 提供时钟的状态机才会冻结，APB2 接口仍保持活动状态。因此，即使激活流控制，仍可填充或清空 FIFO。

要使能硬件流控制，SDMMC_CLKCR[14] 寄存器位必须置 1。复位后，流控制将禁止。

35.8 SDMMC 寄存器

器件通过 32 位控制寄存器（可通过 APB2 访问）与系统进行通信。

35.8.1 SDMMC 电源控制寄存器 (SDMMC_POWER)

SDMMC power 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.	Res.	Res.	Res.

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.	PWRCTRL	
														rw	rw

位 31:2 保留，必须保持复位值。

[1:0] **PWRCTRL**：电源控制位 (Power supply control bits)。

这些位用于定义卡时钟的当前功能状态：
00：掉电：停止为卡提供时钟。
01：保留
10：保留，上电
11：通电：为卡提供时钟。

注： *此寄存器的两次写访问至少需要相隔七个 PCLK2 时钟周期。*

注： *写入数据后，在三个 SDMMCCLK (48 MHz) 时钟周期以及两个 PCLK2 时钟周期内无法向该寄存器写入数据。*

35.8.2 SDMMC 时钟控制寄存器 (SDMMC_CLKCR)

SDMMC clock control register

偏移地址：0x04

复位值：0x0000 0000

SDMMC_CLKCR 寄存器控制 SDMMC_CK 输出时钟。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	HWFC_EN	NEGEDGE	WIDBUS		BYPASS	PWRSAV	CLKEN	CLKDIV							
	rw	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 **HWFC_EN**：硬件流控制使能 (HW Flow Control enable)

0b：禁止硬件流控制

1b：使能硬件流控制

如果使能硬件流控制，请参见*第 35.8.11 节*的 SDMMC 状态寄存器定义来确定 TXFIFOE 和 RXFIFOF 中断信号的含义。

位 13 **NEGEDGE**：SDMMC_CK 移相选择位 (SDMMC_CK dephasing selection bit)

0b：在 SDMMC_CK 上升沿后的 SDMMCCLK 下降沿更改命令和数据（SDMMC_CK 上升沿在 SDMMCCLK 上升沿时出现）。

1b：在 SDMMC_CK 下降沿更改命令和数据。

BYPASS 激活时，无论 NEGEDGE 值是多少，都将在 SDMMCCLK 下降沿更改数据和命令。

位 12:11 **WIDBUS**：宽总线模式使能位 (Wide bus mode enable bit)

00：默认总线模式：使用 SDMMC_D0

01：4 位宽总线模式：使用 SDMMC_D[3:0]

10：8 位宽总线模式：使用 SDMMC_D[7:0]

位 10 **BYPASS**：时钟分频器旁路使能位 (Clock divider bypass enable bit)

0：禁止旁路：在驱动 SDMMC_CK 输出信号前，根据 CLKDIV 值对 SDMMCCLK 进行分频。

1：使能旁路：SDMMCCLK 直接驱动 SDMMC_CK 输出信号。

位 9 **PWRSAV**：节能模式配置位 (Power saving configuration bit)

要实现节能模式，可在总线空闲时通过将 PWRSAV 置 1 来禁止 SDMMC_CK 时钟输出：

0：始终使能 SDMMC_CK 时钟

1：仅在总线激活时使能 SDMMC_CK

位 8 **CLKEN**：时钟使能位 (Clock enable bit)

0：禁止 SDMMC_CK

1：使能 SDMMC_CK

位 7:0 **CLKDIV**：时钟分频系数 (Clock divide factor)

该字段定义输入时钟 (SDMMCCLK) 与输出时钟 (SDMMC_CK) 之间的分频系数：SDMMC_CK 频率 = SDMMCCLK / [CLKDIV + 2]。

注： *1 当 SD/SDIO 卡或 MMC 处于识别模式时，SDMMC_CK 频率必须小于 400 kHz。*

2 如果为所有卡分配相对卡地址，则可以将时钟频率更改为卡总线最大频率。

3 写入数据后，在三个 SDMMCCLK (48 MHz) 时钟周期以及两个 PCLK2 时钟周期内无法向该寄存器写入数据。在 SD I/O 卡的读取等待间隔期间也可以停止 SDMMC_CK：在此情况下，SDMMC_CLKCR 寄存器不对 SDMMC_CK 进行控制。

35.8.3 SDMMC 参数寄存器 (SDMMC_ARG)

SDMMC argument register

偏移地址：0x08

复位值：0x0000 0000

SDMMC_ARG 寄存器包含一个 32 位命令参数，该参数作为命令消息的一部分发送到卡。

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

位 31:0 **CMDARG**：命令参数 (Command argument)

作为命令消息的一部分发送给卡的命令参数。如果命令包含参数，则在将命令写入到命令寄存器之前，必须将参数加载到此寄存器中。

35.8.4 SDMMC 命令寄存器 (SDMMC_CMD)

SDMMC command register

偏移地址：0x0C

复位值：0x0000 0000

SDMMC_CMD 寄存器包含命令索引和命令类型位。命令索引作为命令消息的一部分而发送给卡。命令类型位控制命令路径状态机 (CPSM)。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	SDIO Suspend	CPSM EN	WAIT PEND	WAIT INT	WAITRESP		CMDINDEX					
				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 **SDIOSuspend**：SD I/O 挂起命令 (SD I/O suspend command)

如果此位置 1，则要发送的命令为挂起命令（仅用于 SDIO 卡）。

位 10 **CPSMEN**：命令路径状态机 (CPSM) 使能位 (Command path state machine (CPSM) Enable bit)

如果此位置 1，则使能 CPSM。

位 9 **WAITPEND**：CPSM 等待数据传输结束（CmdPend 内部信号）(CPSM Waits for ends of data transfer (CmdPend internal signal))。

如果此位置 1，则 CPSM 将等到数据传输结束后才开始发送命令。仅当流数据传输模式 SDMMC_DCTRL[2] = 1 时才可使用该功能。

位 8 **WAITINT**：CPSM 等待中断请求 (CPSM waits for interrupt request)
如果此位置 1，则 CPSM 禁止命令超时并等待中断请求。

位 7:6 WAITRESP：等待响应位 (Wait for response bits)
这些位用于配置 CPSM 是否等待响应，如果等待，将等待哪种类型的响应。
00：无响应，但 CMDSENT 标志除外
01：短响应，但 CMDREND 或 CCRCFAIL 标志除外
10：无响应，但 CMDSENT 标志除外
11：长响应，但 CMDREND 或 CCRCFAIL 标志除外

位 5:0 **CMDINDEX**：命令索引 (Command index)
命令索引作为命令消息的一部分发送给卡。

注： *1* *写入数据后，在三个 SDMMCCLK (48 MHz) 时钟周期以及两个 PCLK2 时钟周期内无法向该寄存器写入数据。*

2 *MMC 可以发送两种类型的响应：短响应（48 位）或长响应（136 位）。SD 卡和 SD I/O 卡则只能发送短响应，参数因响应类型而异：软件将根据发送的命令区分响应类型。*

35.8.5 SDMMC 命令响应寄存器 (SDMMC_RESPCMD)

SDMMC command response register

偏移地址：0x10

复位值：0x0000 0000

SDMMC_RESPCMD 寄存器包含接收到的最后一个命令响应的命令索引字段。如果命令响应传输不包含命令索引字段（长响应或 OCR 响应），则 RESPCMD 字段为未知，但该字段必须包含 111111b（响应中保留字段的值）。

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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RESPCMD					
										r	r	r	r	r	r

位 31:6 保留，必须保持复位值。

位 5:0 **RESPCMD**：响应命令索引 (Response command index)
只读位字段。包含接收到的最后一个命令响应的命令索引。

35.8.6 SDMMC 响应 1..4 寄存器 (SDMMC_RESPx)

SDMMC response 1..4 register

偏移地址：(0x10 + (4 × x))； x = 1..4

复位值：0x0000 0000

SDMMC_RESP1/2/3/4 寄存器包含卡的状态，该状态来自接收的响应。

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

位 31:0 **CARDSTATUSx**：请参见*表 222*。

卡状态为 32 位或 127 位，具体取决于响应类型。

表 222. 响应类型和 SDMMC_RESPx 寄存器

寄存器	短响应	长响应
SDMMC_RESP1	卡状态 [31:0]	卡状态 [127:96]
SDMMC_RESP2	未使用	卡状态 [95:64]
SDMMC_RESP3	未使用	卡状态 [63:32]
SDMMC_RESP4	未使用	卡状态 [31:1]0b

首先接收卡状态的最高有效位。SDMMC_RESP4 寄存器 LSB 始终为 0b。

35.8.7 SDMMC 数据定时器寄存器 (SDMMC_DTIMER)

SDMMC data timer register

偏移地址：0x24

复位值：0x0000 0000

SDMMC_DTIMER 寄存器包含以卡总线时钟周期表示的数据超时周期。

计数器加载 SDMMC_DTIMER 寄存器的值，并在数据路径状态机 (DPSM) 进入 Wait_R 或繁忙状态后开始递减。如果定时器达到 0 而 DPSM 处于上述一种状态，则超时状态标志置 1。

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

位 31:0 **DATATIME**：数据超时周期 (Data timeout period)
以卡总线时钟周期表示的数据超时周期。

注： *数据传输必须首先写入到数据计时器寄存器和数据长度寄存器，然后才写入到数据控制寄存器。*

35.8.8 SDMMC 数据长度寄存器 (SDMMC_DLEN)

SDMMC data length register

偏移地址：0x28

复位值：0x0000 0000

SDMMC_DLEN 寄存器包含要传输的数据字节数。当数据传输开始时，值将加载到数据计数器中。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATALENGTH[24:16]								
							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
DATALENGTH[15:0]															
rw	rw	rw	rw	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 24:0 **DATALENGTH**：数据长度值 (Data length value)
要传输的数据字节数量。

注： *对于块数据传输，数据长度寄存器中的值必须是块大小的倍数（请参见 SDMMC_DCTRL）。数据传输必须首先写入到数据计时器寄存器和数据长度寄存器，然后才写入到数据控制寄存器。*

对于 SDMMC 多字节传输，数据长度寄存器中的值必须在 1 到 512 之间。

35.8.9 SDMMC 数据控制寄存器 (SDMMC_DCTRL)

SDMMC data control register

偏移地址：0x2C

复位值：0x0000 0000

SDMMC_DCTRL 寄存器控制数据路径状态机 (DPSM)。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	SDIO EN	RW MOD	RW STOP	RW START	DBLOCKSIZE				DMA EN	DT MODE	DTDIR	DTEN
				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 **SDIOEN**：SD I/O 使能功能 (SD I/O enable functions)
如果将该位置 1，则 DPSM 执行特定于 SD I/O 卡的操作。

位 10 **RWMOD**：读取等待模式 (Read wait mode)
0：通过停止 SDMMC_D2 进行读取等待控制
1：使用 SDMMC_CK 进行读取等待控制

位 9 **RWSTOP**：读取等待停止 (Read wait stop)
0：如果将 RWSTART 位置 1，则读取等待正在进行中
1：如果将 RWSTART 位置 1，则使能读取等待停止

位 8 **RWSTART**：读取等待开始 (Read wait start)
如果将该位置 1，则读取等待操作开始。

位 7:4 **DBLOCKSIZE**：数据块大小 (Data block size)
定义在选择了块数据传输模式时数据块的长度：
0000：（十进制数 0）块长度 = 2^0 = 1 字节
0001：（十进制数 1）块长度 = 2^1 = 2 字节
0010：（十进制数 2）块长度 = 2^2 = 4 字节
0011：（十进制数 3）块长度 = 2^3 = 8 字节
0100：（十进制数 4）块长度 = 2^4 = 16 字节
0101：（十进制数 5）块长度 = 2^5 = 32 字节
0110：（十进制数 6）块长度 = 2^6 = 64 字节
0111：（十进制数 7）块长度 = 2^7 = 128 字节
1000：（十进制数 8）块长度 = 2^8 = 256 字节
1001：（十进制数 9）块长度 = 2^9 = 512 字节
1010：（十进制数 10）块长度 = 2^{10} = 1024 字节
1011：（十进制数 11）块长度 = 2^{11} = 2048 字节
1100：（十进制数 12）块长度 = 2^{12} = 4096 字节
1101：（十进制数 13）块长度 = 2^{13} = 8192 字节
1110：（十进制数 14）块长度 = 2^{14} = 16384 字节
1111：（十进制数 15）保留

位 3 **DMAEN**：DMA 使能位 (DMA enable bit)
0：禁止 DMA。
1：使能 DMA。

位 2 **DTMODE**：数据传输模式选择 1 (Data transfer mode selection 1)：流或 SDIO 多字节数据传输。

0：块数据传输。
1：流或 SDIO 多字节数据传输。

位 1 **DTDIR**：数据传输方向选择 (Data transfer direction selection)

0：从控制器到卡。
1：从卡到控制器。

[0] **DTEN**：数据传输使能位 (Data transfer enabled bit)

如果 1 写入到 DTEN 位，则数据传输开始。根据方向位 DTDIR，如果在传输开始时立即将 RW 置 1开始，则 DPSM 变为 Wait_S 状态、Wait_R 状态或读取等待状态。在数据传输结束后不需要将使能位清零，但必须更新 SDMMC_DCTRL 以使能新的数据传输

注： *写入数据后，在三个 SDMMCCLK (48 MHz) 时钟周期以及两个 PCLK2 时钟周期内无法向该寄存器写入数据。*

根据 SDIOEN 位的值，DTMODE 位的含义将有所不同。如果 SDIOEN=0 且 DTMODE=1，则使能 MMC 流模式，如果 SDIOEN=1 且 DTMODE=1，则外设使能 SDIO 多字节传输。

35.8.10 SDMMC 数据计数器寄存器 (SDMMC_DCOUNT)

SDMMC data counter register

偏移地址：0x30

复位值：0x0000 0000

当 DPSM 从空闲状态变为 Wait_R 或 Wait_S 状态时，SDMMC_DCOUNT 寄存器将从数据长度寄存器中加载值（请参见 SDMMC_DLEN）。在传输数据时，计数器将值递减直至计数器达到 0。然后 DPSM 将变为空闲状态，并且将数据状态结束标志 DATAEND 置 1。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATACOUNT[24:16]								
							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
DATACOUNT[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:25 保留，必须保持复位值。

位 24:0 **DATACOUNT**：数据计数值 (Data count value)

读取该位时，将返回要传输的剩余数据字节数量。写入没有任何效果。

注： *仅当数据传输完成后才应读取该寄存器。*

35.8.11 SDMMC 状态寄存器 (SDMMC_STA)

SDMMC status register

偏移地址：0x34

复位值：0x0000 0000

SDMMC_STA 寄存器是一个只读寄存器。它包含两种类型的标志：

- 静态标志（位 [23:22,10:0]）：在通过写入到 SDMMC 中断清零寄存器来清零这些位之前，会一直保持这些位（请参见 SDMMC_ICR）。
- 动态标志（位 [21:11]）：这些位根据底层逻辑的状态来更改状态（例如，随着数据写入到 FIFO，发出和停止发出 FIFO 满和空标志）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SDIOIT	RXD AVL	TXD AVL	RX FIFOE	TX FIFOE	RX FIFOF	TX FIFOF
									r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX FIFO HF	TX FIFO HE	RXACT	TXACT	CMD ACT	DBCK END	Res.	DATA END	CMDS ENT	CMDR END	RX OVERR	TXUND ERR	DTIME OUT	CTIME OUT	DCRC FAIL	CCRC FAIL
r	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 **SDIOIT**：收到了 SDIO 中断 (SDIO interrupt received)

位 21 **RXDAVL**：接收 FIFO 中有数据可用 (Data available in receive FIFO)

位 20 **TXDAVL**：发送 FIFO 中有数据可用 (Data available in transmit FIFO)

位 19 **RXFIFOE**：接收 FIFO 为空 (Receive FIFO empty)

位 18 **TXFIFOE**：发送 FIFO 为空 (Transmit FIFO empty)
如果使能了硬件流控制，则 TXFIFOE 信号在 FIFO 包含 2 个字时激活。

位 17 **RXFIFOF**：接收 FIFO 已满 (Receive FIFO full)
如果使能了硬件流控制，则 RXFIFOF 信号在 FIFO 差 2 个字便变满之前激活。

位 16 **TXFIFOF**：发送 FIFO 已满 (Transmit FIFO full)

位 15 **RXFIFOHF**：接收 FIFO 半满：FIFO 中至少有 8 个字 (Receive FIFO half full: there are at least 8 words in the FIFO)

位 14 **TXFIFOHE**：发生 FIFO 半空：至少可以写入 8 个字到 FIFO (Transmit FIFO half empty: at least 8 words can be written into the FIFO)

位 13 **RXACT**：数据接收正在进行中 (Data receive in progress)

位 12 **TXACT**：数据发送正在进行中 (Data transmit in progress)

位 11 **CMDACT**：命令传输正在进行中 (Command transfer in progress)

位 10 **DBCKEND**：已发送 / 接收数据块（CRC 校验通过）(Data block sent/received (CRC check passed))

位 9 保留，必须保持复位值。

位 8 **DATAEND**：数据结束（数据计数器 SDIDCOUNT 为零）(Data end (data counter, SDIDCOUNT, is zero))

位 7 **CMDSENT**：命令已发送（不需要响应）(Command sent (no response required))

位 6 **CMDREND**：已接收命令响应（CRC 校验通过）(Command response received (CRC check passed))

位 5 **RXOVERR**：收到了 FIFO 上溢错误 (Received FIFO overrun error)

注： 如果 DMA 用于读取 SDMMC FIFO（SDMMC_DCTRL 寄存器中的 DMAEN 位置 1），用户软件应禁止 DMA 流，然后写入“0”（以禁止生成 DMA 请求）。

位 4 **TXUNDERR**：发送 FIFO 下溢错误 (Transmit FIFO underrun error)

注： 如果 DMA 用于填充 SDMMC FIFO（SDMMC_DCTRL 寄存器中的 DMAEN 位置 1），用户软件应禁止 DMA 流，然后向 DMAEN 写入“0”（以禁止生成 DMA 请求）。

位 3 **DTIMEOUT**：数据超时 (Data timeout)

位 2 **CTIMEOUT**：命令响应超时 (Command response timeout)

命令超时周期为固定值：64 个 SDMMC_CK 时钟周期。

位 1 **DCRCFAIL**：已发送/接收数据块（CRC 校验失败）(Data block sent/received (CRC check failed))

位 0 **CCRCFAIL**：已接收命令响应（CRC 校验失败）(Command response received (CRC check failed))

35.8.12 SDMMC 中断清零寄存器 (SDMMC_ICR)

SDMMC interrupt clear register

偏移地址：0x38

复位值：0x0000 0000

SDMMC_ICR 寄存器是一个只写寄存器。向某个位写入 1 会将 SDMMC_STA 状态寄存器中的对应位清零。

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.	SDIOITC	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.	DBCKENDC	Res.	DATAENDC	CMDSENTC	CMDRENDC	RXOVERRC	TXUNDERRC	DTIMEOUTC	CTIMEOUTC	DCRCFAILC	CCRCFAILC
					rw		rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:23 保留，必须保持复位值。

位 22 **SDIOITC**：SDIOIT 标志清零位 (SDIOIT flag clear bit)

由软件置 1，用于将 SDIOIT 标志清零。

0：未将 SDIOIT 清零

1：已将 SDIOIT 清零

位 21:11 保留，必须保持复位值。

位 10 **DBCKENDC**：DBCKEND 标志清零位 (DBCKEND flag clear bit)

由软件置 1，用于将 DBCKEND 标志清零。

0：未将 DBCKEND 清零

1：已将 DBCKEND 清零

位 9 保留，必须保持复位值。

位 8 **DATAENDC**：DATAEND 标志清零位 (DATAEND flag clear bit)
由软件置 1，用于将 DATAEND 标志清零。
0：未将 DATAEND 清零
1：已将 DATAEND 清零

位 7 **CMDSENTC**：CMDSENT 标志清零位 (CMDSENT flag clear bit)
由软件置 1，用于将 CMDSENT 标志清零。
0：未将 CMDSENT 清零
1：已将 CMDSENT 清零

位 6 **CMDRENDC**：CMDREND 标志清零位 (CMDREND flag clear bit)
由软件置 1，用于将 CMDREND 标志清零。
0：未将 CMDREND 清零
1：已将 CMDREND 清零

位 5 **RXOVERRC**：RXOVERR 标志清零位 (RXOVERR flag clear bit)
由软件置 1，用于将 RXOVERR 标志清零。
0：未将 RXOVERR 清零
1：已将 RXOVERR 清零

位 4 **TXUNDERRC**：TXUNDERR 标志清零位 (TXUNDERR flag clear bit)
由软件置 1，用于将 TXUNDERR 标志清零。
0：未将 TXUNDERR 清零
1：已将 TXUNDERR 清零

位 3 **DTIMEOUTC**：DTIMEOUT 标志清零位 (DTIMEOUT flag clear bit)
由软件置 1，用于将 DTIMEOUT 标志清零。
0：未将 DTIMEOUT 清零
1：已将 DTIMEOUT 清零

位 2 **CTIMEOUTC**：CTIMEOUT 标志清零位 (CTIMEOUT flag clear bit)
由软件置 1，用于将 CTIMEOUT 标志清零。
0：未将 CTIMEOUT 清零
1：已将 CTIMEOUT 清零

位 1 **DCRCFAILC**：DCRCFAIL 标志清零位 (DCRCFAIL flag clear bit)
由软件置 1，用于将 DCRCFAIL 标志清零。
0：未将 DCRCFAIL 清零
1：已将 DCRCFAIL 清零

位 0 **CCRCFAILC**：CCRCFAIL 标志清零位 (CCRCFAIL flag clear bit)
由软件置 1，用于将 CCRCFAIL 标志清零。
0：未将 CCRCFAIL 清零
1：已将 CCRCFAIL 清零

35.8.13 SDMMC 屏蔽寄存器 (SDMMC_MASK)

SDMMC mask register

偏移地址：0x3C

复位值：0x0000 0000

中断屏蔽寄存器通过将对应的位置 1 来确定哪一个状态标志位可以产生中断。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SDIOITIE	RXDAVLIE	TXDAVLIE	RXFIFOEIE	TXFIFOEIE	RXFIFOFIE	TXFIFOFIE
									rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXFIFOHFIE	TXFIFOHEIE	RXACTIE	TXACTIE	CMDACTIE	DBCKENDIE	Res.	DATAENDIE	CMDSENTIE	CMDRENDIE	RXOVERRIE	TXUNDERRIE	DTIMEOUTIE	CTIMEOUTIE	DCRCFAILIE	CCRCFAILIE
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 **SDIOITIE**：接收到 SDIO 模式中断时中断使能 (SDIO mode interrupt received interrupt enable)

由软件置 1 和清零，用于使能/禁止收到 SDIO 模式中断时生成中断。
0：禁止接收到 SDIO 模式中断时中断
1：使能接收到 SDIO 模式中断时中断

位 21 **RXDAVLIE**：Rx FIFO 中数据可用时中断使能 (Data available in Rx FIFO interrupt enable)

由软件置 1 和清零，用于使能/禁止由于 Rx FIFO 中存在数据而生成的中断。
0：禁止 Rx FIFO 中数据可用时中断
1：使能 Rx FIFO 中数据可用时中断

位 20 **TXDAVLIE**：Tx FIFO 中数据可用时中断使能 (Data available in Tx FIFO interrupt enable)

由软件置 1 和清零，用于使能/禁止由于 Tx FIFO 中存在数据而生成的中断。
0：禁止 Tx FIFO 中数据可用时中断
1：使能 Tx FIFO 中数据可用时中断

位 19 **RXFIFOEIE**：Rx FIFO 为空时中断使能 (Rx FIFO empty interrupt enable)

由软件置 1 和清零，用于使能/禁止由 Rx FIFO 为空引起的中断。
0：禁止 Rx FIFO 为空时中断
1：使能 Rx FIFO 为空时中断

位 18 **TXFIFOEIE**：Tx FIFO 为空时中断使能 (Tx FIFO empty interrupt enable)

由软件置 1 和清零，用于使能/禁止由 Tx FIFO 为空引起的中断。
0：禁止 Tx FIFO 为空时中断
1：使能 Tx FIFO 为空时中断

位 17 **RXFIFOFIE**：Rx FIFO 变满时中断使能 (Rx FIFO full interrupt enable)

由软件置 1 和清零，用于使能/禁止由 Rx FIFO 变满引起的中断。
0：禁止 Rx FIFO 变满时中断
1：使能 Rx FIFO 变满时中断

位 16 **TXFIFOFIE**：Tx FIFO 变满时中断使能 (Tx FIFO full interrupt enable)

由软件置 1 和清零，用于使能/禁止由 Tx FIFO 变满引起的中断。
0：禁止 Tx FIFO 变满时中断
1：使能 Tx FIFO 变满时中断

位 15 **RXFIFOHFIE**：Rx FIFO 半满时中断使能 (Rx FIFO half full interrupt enable)
由软件置 1 和清零，用于使能/禁止由 Rx FIFO 半满引起的中断。
0：禁止 Rx FIFO 半满时中断
1：使能 Rx FIFO 半满时中断

位 14 **TXFIFOHEIE**：Tx FIFO 半空时中断使能 (Tx FIFO half empty interrupt enable)
由软件置 1 和清零，用于使能/禁止由 Tx FIFO 半空引起的中断。
0：禁止 Tx FIFO 半空时中断
1：使能 Tx FIFO 半空时中断

位 13 **RXACTIE**：数据接收操作中断使能 (Data receive acting interrupt enable)
由软件置 1 和清零，用于使能/禁止由正在接收的数据（数据接收操作）导致的中断。
0：禁止数据接收操作中断
1：使能数据接收操作中断

位 12 **TXACTIE**：数据发送操作中断使能 (Data transmit acting interrupt enable)
由软件置 1 和清零，用于使能/禁止由正在传输的数据（数据传输操作）导致的中断。
0：禁止数据发送操作中断
1：使能数据发送操作中断

位 11 **CMDACTIE**：命令操作中断使能 (Command acting interrupt enable)
由软件置 1 和清零，用于使能/禁止由正在传输的命令（命令操作）导致的中断。
0：禁止命令操作中断
1：使能命令操作中断

位 10 **DBCKENDIE**：数据块结束中断使能 (Data block end interrupt enable)
由软件置 1 和清零，用于使能/禁止由数据块结束引起的中断。
0：禁止数据块结束中断
1：使能数据块结束中断

位 9 保留，必须保持复位值。

位 8 **DATAENDIE**：数据结束中断使能 (Data end interrupt enable)
由软件置 1 和清零，用于使能/禁止由数据结束引起的中断。
0：禁止数据结束中断
1：使能数据结束中断

位 7 **CMDSENTIE**：命令发送中断使能 (Command sent interrupt enable)
由软件置 1 和清零，用于使能/禁止由发送命令引起的中断。
0：禁止命令发送中断
1：使能命令发送中断

位 6 **CMDRENDIE**：命令响应接收中断使能 (Command response received interrupt enable)
由软件置 1 和清零，用于使能/禁止由接收命令响应引起的中断。
0：禁止命令响应接收中断
1：使能命令响应接收中断

位 5 **RXOVERRIE**：Rx FIFO 上溢错误中断使能 (Rx FIFO overrun error interrupt enable)
由软件置 1 和清零，用于使能/禁止由 Rx FIFO 上溢错误引起的中断。
0：禁止 Rx FIFO 上溢错误中断
1：使能 Rx FIFO 上溢错误中断

位 4 **TXUNDERRIE**：Tx FIFO 下溢错误中断使能 (Tx FIFO underrun error interrupt enable)
由软件置 1 和清零，用于使能/禁止由 Tx FIFO 下溢错误引起的中断。
0：禁止 Tx FIFO 下溢错误中断
1：使能 Tx FIFO 下溢错误中断

位 3 **DTIMEOUTIE**：数据超时中断使能 (Data timeout interrupt enable)
由软件置 1 和清零，用于使能/禁止由数据超时引起的中断。
0：禁止数据超时中断
1：使能数据超时中断

位 2 **CTIMEOUTIE**：命令超时中断使能 (Command timeout interrupt enable)
由软件置 1 和清零，用于使能/禁止由命令超时引起的中断。
0：禁止命令超时中断
1：使能命令超时中断

位 1 **DCRCFAILIE**：数据 CRC 失败中断使能 (Data CRC fail interrupt enable)
由软件置 1 和清零，用于使能/禁止由数据 CRC 失败引起的中断。
0：禁止数据 CRC 失败中断
1：使能数据 CRC 失败中断

位 0 **CCRCFAILIE**：命令 CRC 失败中断使能 (Command CRC fail interrupt enable)
由软件置 1 和清零，用于使能/禁止由命令 CRC 失败引起的中断。
0：禁止命令 CRC 失败中断
1：使能命令 CRC 失败中断

35.8.14 SDMMC FIFO 计数器寄存器 (SDMMC_FIFOCNT)

SDMMC FIFO counter register

偏移地址：0x48

复位值：0x0000 0000

SDMMC_FIFOCNT 寄存器包含要在 FIFO 中写入或读取的剩余字数。如果将数据控制寄存器（SDMMC_DCTRL 寄存器）中的数据传输使能位 DTEN 置 1，并且 DPSM 处于空闲状态，则 FIFO 计数器从数据长度寄存器中加载值（请参见 SDMMC_DLEN）。如果数据长度未进行字对齐（4 的倍数），则剩余的 1 到 3 个字节被视为一个字。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFOCOUNT[23:16]							
								r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIFOCOUNT[15:0]															
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 23:0 **FIFOCOUNT**：要在 FIFO 中写入或读取的剩余字数。

35.8.15 SDMMC 数据 FIFO 寄存器 (SDMMC_FIFO)

SDMMC data FIFO register

偏移地址：0x80

复位值：0x0000 0000

接收和发送 FIFO 可以被当做 32 位宽的寄存器来读出或写入。FIFO 在 32 个连续地址上包含 32 个入口。这使 CPU 可以使用其加载 (load) 和存储 (store) 多操作符命令来读写 FIFO。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIF0Data[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
FIF0Data[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **FIFOData**：接收和发送 FIFO 数据 (Receive and transmit FIFO data)

FIFO 数据占用 32 位字的 32 个入口，从地址：SDMMC 基址 + 0x080 到 SDMMC 基址 + 0xFC。

35.8.16 SDMMC 寄存器映射

下表对 SDMMC 寄存器进行了汇总。

表 223. SDMMC 寄存器映射

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	SDMMC_ POWER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWRCTRL	
	Reset value																															0	0
0x04	SDMMC_ CLKCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HWFC_EN	NEGEDGE	WIDBUS	BYPASS	PWRSAV	CLKEN	CLKDIV								
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	SDMMC_ARG	CMDARG																															
	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	SDMMC_CMD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SDIOSuspend	CPSMEN	WAITPEND	WAITINT	WAITRESP		CMDINDEX					
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x10	SDMMC_ RESPCMD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RESPCMD					
	Reset value																											0	0	0	0	0	0
0x14	SDMMC_ RESP1	CARDSTATUS1																															
	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 223. SDMMC 寄存器映射（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
0x18	**SDMMC_ RESP2**	CARDSTATUS2																															
	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	**SDMMC_ RESP3**	CARDSTATUS3																															
	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	**SDMMC_ RESP4**	CARDSTATUS4																															
	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	**SDMMC_ DTIMER**	DATATIME																															
	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	**SDMMC_ DLEN**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATALENGTH																								
	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
0x2C	**SDMMC_ DCTRL**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SDIOEN	RWMOD	RWSTOP	RWSTART	DBLOCKSIZE			DMAEN	DTMODE	DTDIR	DTEN	
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x30	**SDMMC_ DCOUNT**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATACOUNT																								
	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
0x34	**SDMMC_STA**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SDIOIT	RXDAVL	TXDAVL	RXFIFOE	TXFIFOE	RXFIFOF	TXFIFOF	RXFIFOHF	TXFIFOHE	RXACT	TXACT	CMDACT	DBCKEND	Res.	DATAEND	CMDSENT	CMDREND	RXOVERR	TXUNDERR	DTIMEOUT	CTIMEOUT	DCRCFAIL	CCRCFAIL
	Reset value										0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
0x38	**SDMMC_ICR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SDIOITC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBCKENDC	Res.	DATAENDC	CMDSENTC	CMDRENDC	RXOVERRC	TXUNDERRC	DTIMEOUTC	CTIMEOUTC	DCRCFAILC	CCRCFAILC
	Reset value										0												0		0	0	0	0	0	0	0	0	0
0x3C	**SDMMC_ MASK**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SDIOITIE	RXDAVLIE	TXDAVLIE	RXFIFOEIE	TXFIFOEIE	RXFIFOFIE	TXFIFOFIE	RXFIFOHFIE	TXFIFOHEIE	RXACTIE	TXACTIE	CMDACTIE	DBCKENDIE	Res.	DATAENDIE	CMDSENTIE	CMDRENDIE	RXOVERRIE	TXUNDERRIE	DTIMEOUTIE	CTIMEOUTIE	DCRCFAILIE	CCRCFAILIE
	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	**SDMMC_ FIFOCNT**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFOCOUNT																							
	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
0x80	**SDMMC_FIFO**	FIF0Data																															
	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 2.2.2 节：存储器映射和寄存器边界地址*。

36 控制器区域网络 (bxCAN)

36.1 前言

基本扩展 CAN 外设又称 bxCAN，可与 CAN 网络进行交互。该外设支持 2.0A 和 B 版本的 CAN 协议，旨在以最少的 CPU 负载高效管理大量的传入消息， 并可按需要的优先级实现消息发送。

在有关安全性的应用中，CAN 控制器提供所有必要的硬件功能来支持 CAN 时间触发通信方案。

36.2 bxCAN 主要特性

- 支持 2.0 A 及 2.0 B Active 版本 CAN 协议
- 比特率高达 1 Mb/s
- 支持时间触发通信方案

发送

- 三个发送邮箱
- 可配置的发送优先级
- SOF 发送时间戳

接收

- 两个具有三级深度的接收 FIFO
- 可调整的滤波器组：
  - CAN1 和 CAN2 之间共享 28 个滤波器组
- 标识符列表功能
- 可配置的 FIFO 上溢
- SOF 接收时间戳

时间触发通信方案

- 禁止自动重发送模式
- 16 位自由运行定时器
- 在最后两个数据字节发送时间戳

管理

- 可屏蔽中断
- 在唯一地址空间通过软件实现高效的邮箱映射

双 CAN

- CAN1：主 bxCAN，用于管理 bxCAN 与 512 字节 SRAM 存储器之间的通信
- CAN2：从 bxCAN，无法直接访问 SRAM 存储器。
- 两个 bxCAN 单元共享 512 字节 SRAM 存储器（请参见*图 422：双 CAN 框图*）

36.3 bxCAN 一般说明

在如今的 CAN 应用中，网络节点数量日益增多，经常需要通过网关将数个网络连接在一起。系统中的消息数量（以及各个节点需要处理的消息）也有了显著增加。除应用程序消息外，还引入了网络管理和诊断消息。

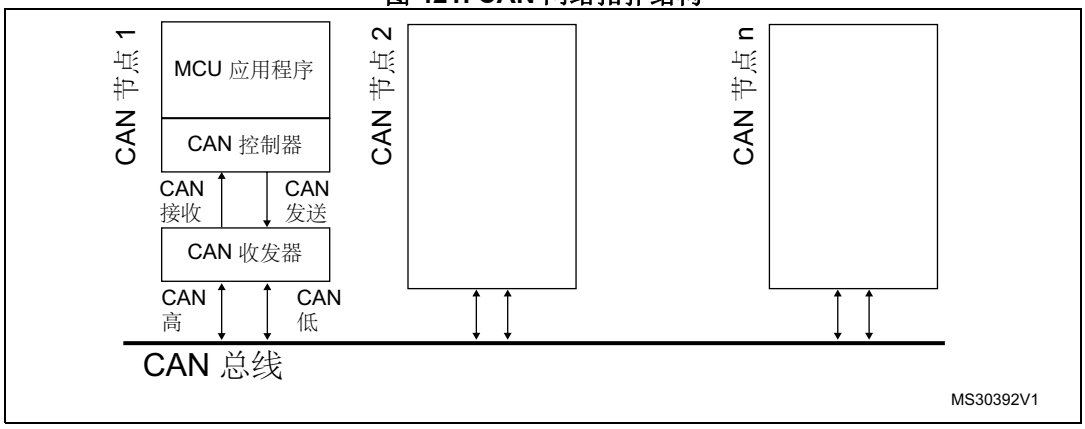
- 各种类型的消息需要一个增强的筛选机制进行处理。

此外，应用程序任务需要更多的 CPU 时间，因此必须减少因消息接收而对实时处理造成的限制。

- 接收 FIFO 方案使 CPU 能够长时间专门处理应用程序任务，而又不致丢失消息。

基于标准 CAN 驱动程序的标准 HLP（更高层协议）需要一个高效接口来与 CAN 控制器连接。

图 421. CAN 网络拓扑结构



36.3.1 CAN 2.0B 主动内核

bxCAN 模块可完全自主地处理 CAN 消息的发送和接收。标准标识符（11 位）和扩展标识符（29 位）完全由硬件支持。

36.3.2 控制、状态和配置寄存器

应用程序使用这些寄存器进行以下操作：

- 配置 CAN 参数，例如波特率
- 请求发送
- 处理接收
- 管理中断
- 获取诊断信息

36.3.3 发送邮箱

软件可通过三个发送邮箱设置消息。发送调度程序负责决定首先发送哪个邮箱的内容。

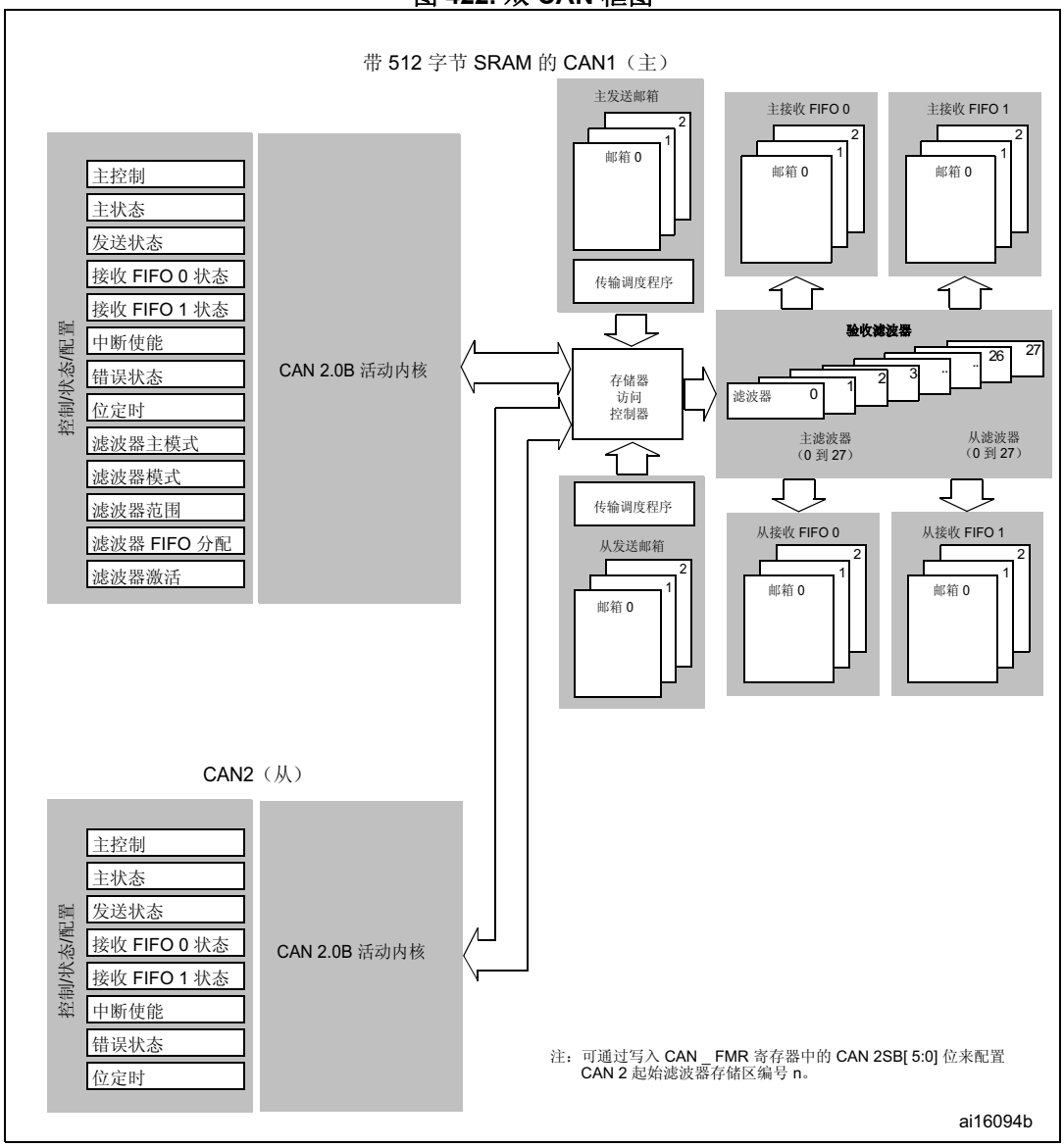
36.3.4 验收滤波器

bxCAN 提供了 28 个可调整/可配置的标识符滤波器组，用于选择软件所需的传入消息并丢弃其余消息。

接收 FIFO

硬件使用两个接收 FIFO 来存储传入消息。每个 FIFO 中可以存储三条完整消息。FIFO 完全由硬件管理。

图 422. 双 CAN 框图



36.4 bxCAN 工作模式

bxCAN 有三种主要的工作模式：**初始化模式**、**正常模式**和**睡眠模式**。硬件复位后，bxCAN 进入睡眠模式以降低功耗，同时 CANTX 上的内部上拉电阻激活。软件将 CAN_MCR 寄存器的 INRQ 或 SLEEP 位置 1，以请求 bxCAN 进入**初始化**或**睡眠**模式。一旦进入该模式，bxCAN 即将 CAN_MSR 寄存器的 INAK 或 SLAK 位置 1，以确认该模式，同时禁止内部上拉电阻。如果 INAK 和 SLAK 均未置 1，则 bxCAN 将处于**正常**模式。进入**正常**模式之前，bxCAN 必须始终在 CAN 总线上实现**同步**。为了进行同步，bxCAN 将等待 CAN 总线空闲（即，已监测到 CANRX 上的 11 个连续隐性位）。

36.4.1 初始化模式

当硬件处于初始化模式时，可以进行软件初始化。为进入该模式，软件将 CAN_MCR 寄存器的 INRQ 位置 1，并等待硬件通过将 CAN_MCR 寄存器的 INAK 位置 1 来确认请求。

为退出初始化模式，软件将 INQR 位清零。一旦硬件将 INAK 位清零，bxCAN 即退出初始化模式。

在初始化模式下，所有从 CAN 总线传入和传出的消息都将停止，并且 CAN 总线输出 CANTX 的状态为隐性（高）。

进入初始化模式不会更改任何配置寄存器。

为初始化 CAN 控制器，软件必须设置位定时 (CAN_BTR) 和 CAN 选项 (CAN_MCR) 寄存器。

为初始化与 CAN 滤波器组相关的寄存器（模式、尺度、FIFO 分配、激活和滤波器值），软件必须将 FINIT 位 (CAN_FMR) 置 1。滤波器的初始化也可以在初始化模式之外进行。

注： *FINIT=1 时，CAN 接收停用。*

滤波器值也可通过停用（CAN_FA1R 寄存器的）相关滤波器激活位来修改。

如果某个滤波器组未使用，建议将其保持未激活状态（将相应 FACT 位保持清零）。

36.4.2 正常模式

一旦初始化完成，软件必须向硬件请求进入正常模式，这样才能在 CAN 总线上进行同步，并开始接收和发送。

进入正常模式的请求可通过将 CAN_MCR 寄存器的 INRQ 位清零来发出。bxCAN 进入正常模式，并与 CAN 总线上的数据传输实现同步后，即可参与总线活动。执行这一步时，需要等待出现一个由 11 个连续隐性位（总线空闲状态）组成的序列。硬件通过将 CAN_MSR 寄存器的 INAK 位清零，来确认切换到正常模式。

滤波器值的初始化与初始化模式无关，但必须要在滤波器处于未激活状态（相应 FACTx 位清零）时进行。滤波器尺度和模式配置必须在进入正常模式之前完成。

36.4.3 睡眠模式（低功耗）

为降低能耗功耗，bxCAN 具有低功耗模式，称为睡眠模式。软件通过将 CAN_MCR 寄存器的 SLEEP 位置 1 而发出请求后，即可进入该模式。该模式下，bxCAN 时钟停止，但软件仍可访问 bxCAN 邮箱。

在 bxCAN 处于**睡眠**模式时，如果软件通过将 INRQ 位置 1 来请求进入**初始化**模式，则必须同时将 SLEEP 位清零。

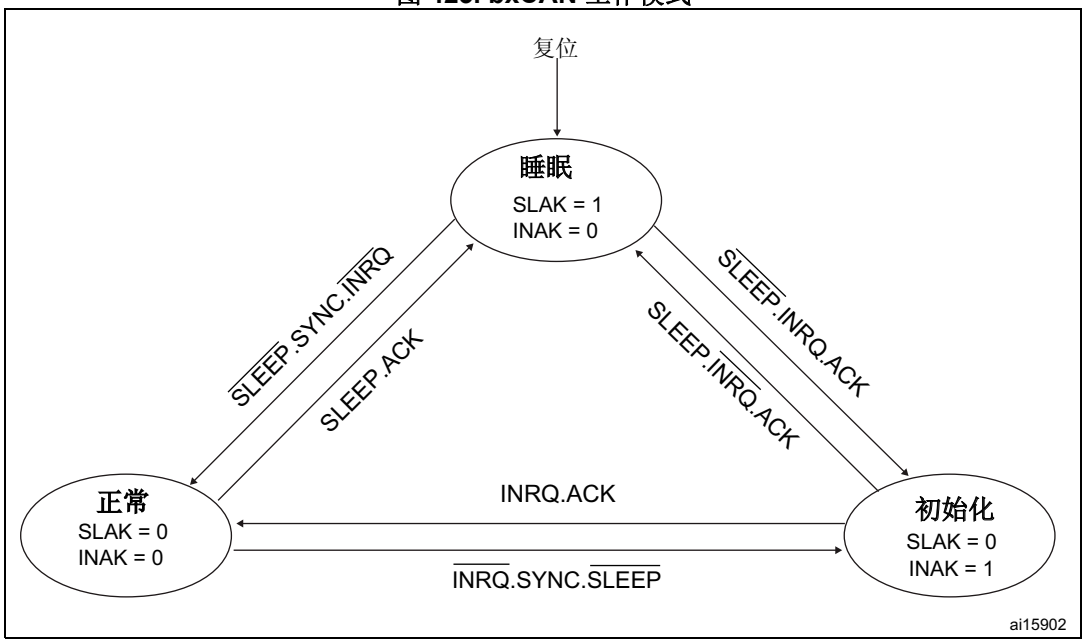
软件将 SLEEP 位清零或是检测到 CAN 总线活动时，bxCAN 即被唤醒（退出睡眠模式）。

检测到 CAN 总线活动后，如果 CAN_MCR 寄存器的 AWUM 位置 1，硬件将通过清零 SLEEP 位来自动执行唤醒序列。如果 AWUM 位清零，在发生唤醒中断时，软件必须将 SLEEP 位清零才能退出睡眠模式。

注： *如果使能唤醒中断（CAN_IER 寄存器的 WKUIE 位置 1），即使 bxCAN 自动执行唤醒序列，一旦检测到 CAN 总线活动，也会发生唤醒中断。*

SLEEP 位清零后，一旦 bxCAN 与 CAN 总线同步，即会退出睡眠模式，请参见 *图 423：bxCAN 工作模式*。一旦硬件将 SLAK 位清零，即会退出睡眠模式。

图 423. bxCAN 工作模式



1. ACK = 硬件通过将 CAN_MSR 寄存器的 INAK 或 SLAK 位置 1 来确认请求的等待状态
2. SYNC = bxCAN 等待 CAN 总线变为空闲（即在 CANRX 上监测到连续 11 个隐性位）的状态

36.5 测试模式

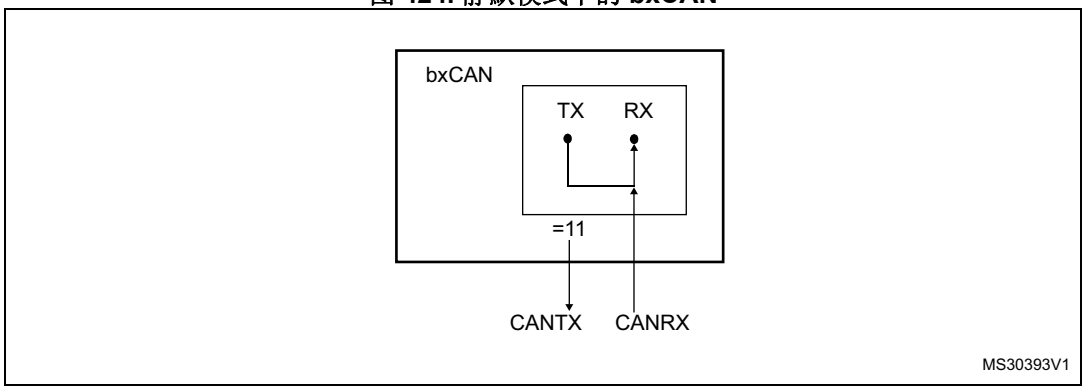
可以通过 CAN_BTR 寄存器中的 SILM 和 LBKM 位来选择测试模式。这些位必须在 bxCAN 处于初始化模式时进行配置。选择测试模式后，必须复位 CAN_MCR 寄存器中的 INRQ 位才能进入正常模式。

36.5.1 静默模式

可以通过将 CAN_BTR 寄存器的 SILM 位置 1，将 bxCAN 置于静默模式。

在静默模式下，bxCAN 可以接收有效数据帧和有效遥控帧，但仅在 CAN 总线上发送隐性位，并且无法启动发送。如果 bxCAN 必须发送一个显性位（ACK 位、溢出标志、活动错误标志），该位将在内部被改道发送，以便 CAN 内核可以监视该显性位，但 CAN 总线可以保持隐性状态。静默模式可用于分析 CAN 总线上的流量，同时又不会因发送显性位（确认位、错误帧）对其造成影响。

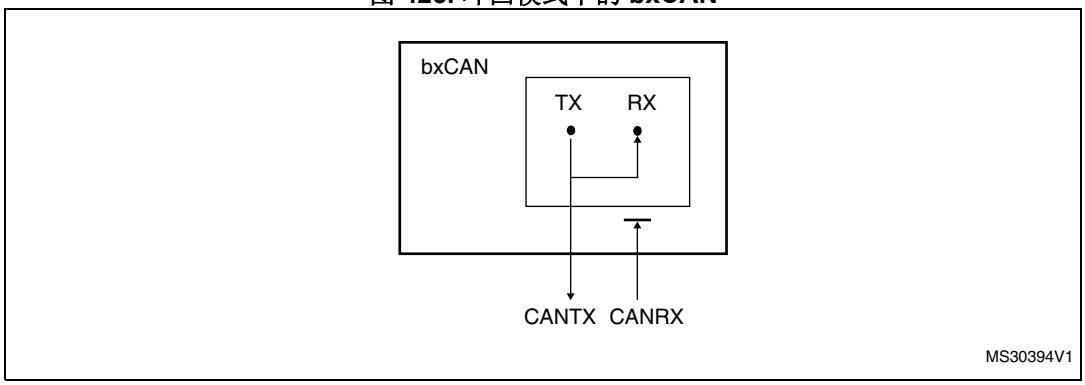
图 424. 静默模式下的 bxCAN



36.5.2 环回模式

可以通过将 CAN_BTR 寄存器的 LBKM 位置 1，将 bxCAN 置于环回模式。在环回模式下，bxCAN 将其自身发送的消息作为接收的消息来处理并存储（如果这些消息通过了验收筛选）在接收邮箱中。

图 425. 环回模式下的 bxCAN

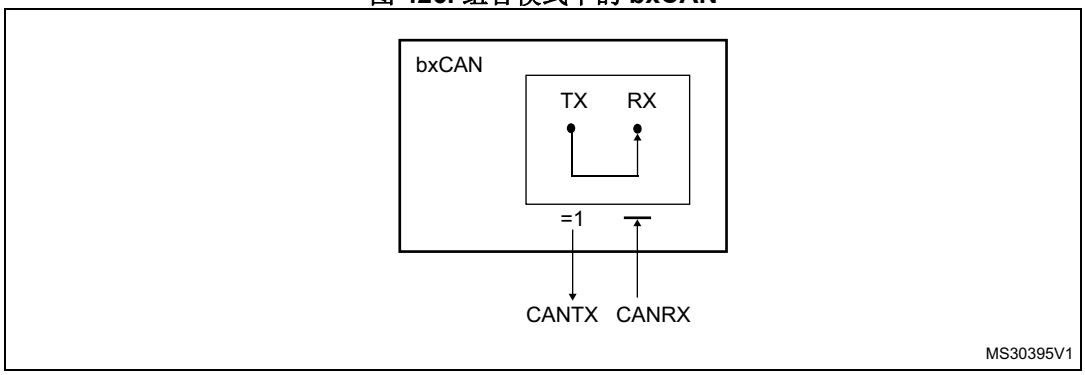


该模式为自检功能提供。为了不受外部事件的影响，CAN 内核在环回模式下将忽略确认错误（在数据/远程帧的确认时隙不对显性位采样）。在此模式下，bxCAN 将执行从发送输出到接收输入的内部反馈。bxCAN 将忽略 CANRX 输入引脚的实际值。从 CANTX 引脚可以监视发送的消息。

36.5.3 环回与静默组合模式

可以通过将 CAN_BTR 寄存器的 LBKM 和 SILM 位置 1，将环回模式和静默模式组合起来。该模式可用于“热自检”，也就是说，bxCAN 可以像在环回模式下一样进行检测，同时又不会影响与 CANTX 和 CANRX 引脚相连接的运行中的 CAN 系统。在此模式下，CANRX 引脚与 bxCAN 断开连接，CANTX 引脚则保持隐性。

图 426. 组合模式下的 bxCAN



36.6 调试模式下的行为

当微控制器进入调试模式（Cortex®-M7 内核停止）时，bxCAN 可以继续正常工作，也可以停止工作，具体取决于如下条件：

- DBG 模块中用于 CAN1 的 DBG_CAN1_STOP 位或者用于 CAN2 的 DBG_CAN2_STOP 位。
- CAN_MCR 中的 DBF 位，有关更多详细信息，请参见*第 36.9.2 节：CAN 控制和状态寄存器*。

36.7 bxCAN 功能说明

36.7.1 发送处理

为了发送消息，应用程序必须在请求发送前，通过将 CAN_TIxR 寄存器的相应 TXRQ 位置 1，选择一个**空**发送邮箱，并设置标识符、数据长度代码 (DLC) 和数据。一旦邮箱退出**空**状态，软件即不再具有对邮箱寄存器的写访问权限。TXRQ 位置 1 后，邮箱立即进入**挂起**状态，等待成为优先级最高的邮箱，请参见*发送优先级*。一旦邮箱拥有最高优先级，即被**安排**发送。CAN 总线变为空闲后，被**安排**好的邮箱中的消息即开始发送（进入**发送**状态）。邮箱一旦发送成功，即恢复**空**状态。硬件通过将 CAN_TSR 寄存器的 RQCP 和 TXOK 位置 1，来表示发送成功。

如果发送失败，失败原因将由 CAN_TSR 寄存器的 ALST 位（仲裁丢失）和/或 TERR 位（检测到发送错误）指示。

发送优先级

按标识符

当多个发送邮箱挂起时，发送顺序由邮箱中所存储消息的标识符来确定。根据 CAN 协议的仲裁，标识符值最低的消息具有最高的优先级。如果标识符值相等，则首先安排发送编号较小的邮箱。

按发送请求顺序

可以通过设置 CAN_MCR 寄存器中的 TXFP 位，将发送邮箱配置为发送 FIFO。在此模式下，优先级顺序按照发送请求顺序来确定。

该模式对分段发送非常有用。

中止

可以通过将 CAN_TSR 寄存器的 ABRQ 位置 1，来中止发送请求。在**挂起**或**已安排**状态下，邮箱立即中止。如果在邮箱处于**发送**状态时请求中止，则会出现两种结果。如果邮箱发送成功，将变为**空**状态，同时 CAN_TSR 寄存器的 TXOK 位置 1。如果发送失败，邮箱变为**已安排**状态，发送中止并变为**空**状态，同时 TXOK 位清零。在所有情况下，邮箱至少在当前发送结束时都会恢复**空**状态。

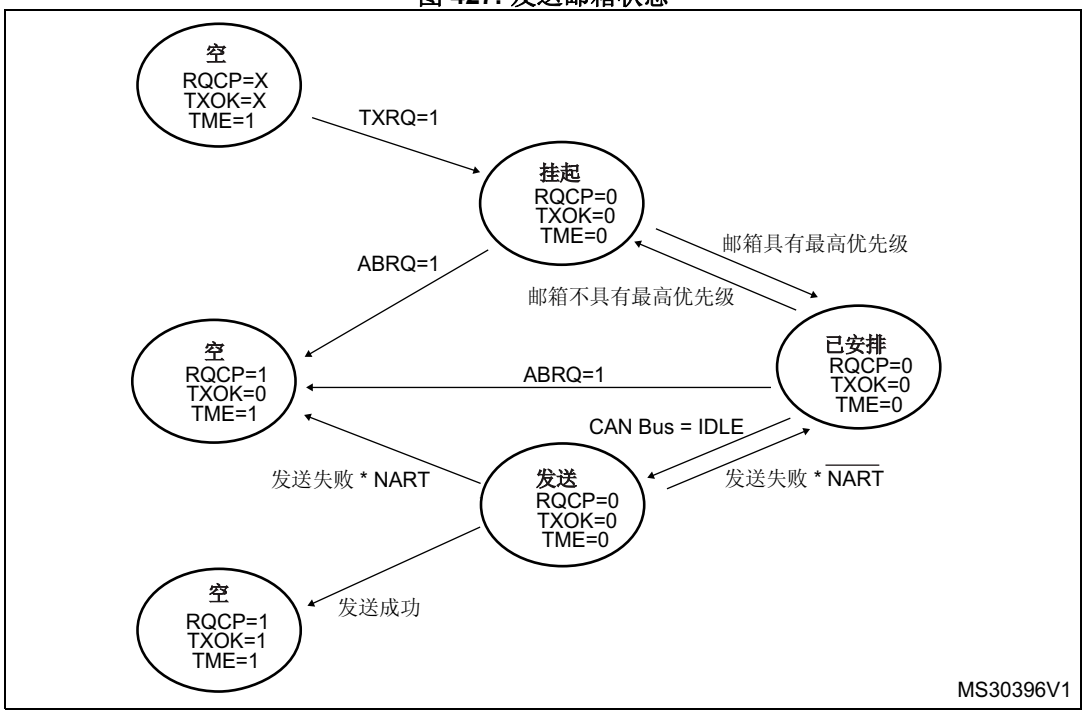
禁止自动重发送模式

该模式旨在满足 CAN 标准的时间触发通信方案的要求。要将硬件配置为此模式，必须将 CAN_MCR 寄存器的 NART 位置 1。

在此模式下，每个发送仅启动一次。如果第一次尝试失败，由于仲裁丢失或错误，硬件将不会自动重新启动消息发送。

第一次发送尝试结束时，硬件将认为请求已完成，并将 CAN_TSR 寄存器的 RQCP 位置 1。发送结果由 CAN_TSR 寄存器的 TXOK、ALST 和 TERR 位来指示。

图 427. 发送邮箱状态



36.7.2 时间触发通信模式 (Time triggered communication mode)

在此模式下，CAN 硬件的内部计数器激活，用于为接收和发送邮箱生成时间戳值，这些值分别存储在 CAN_RDTxR/CAN_TDTxR 寄存器中。内部计数器在每个 CAN 位时间递增（请参见*第 36.7.7 节：位时序*）。在接收和发送时，都会在帧起始位的采样点捕获内部计数器。

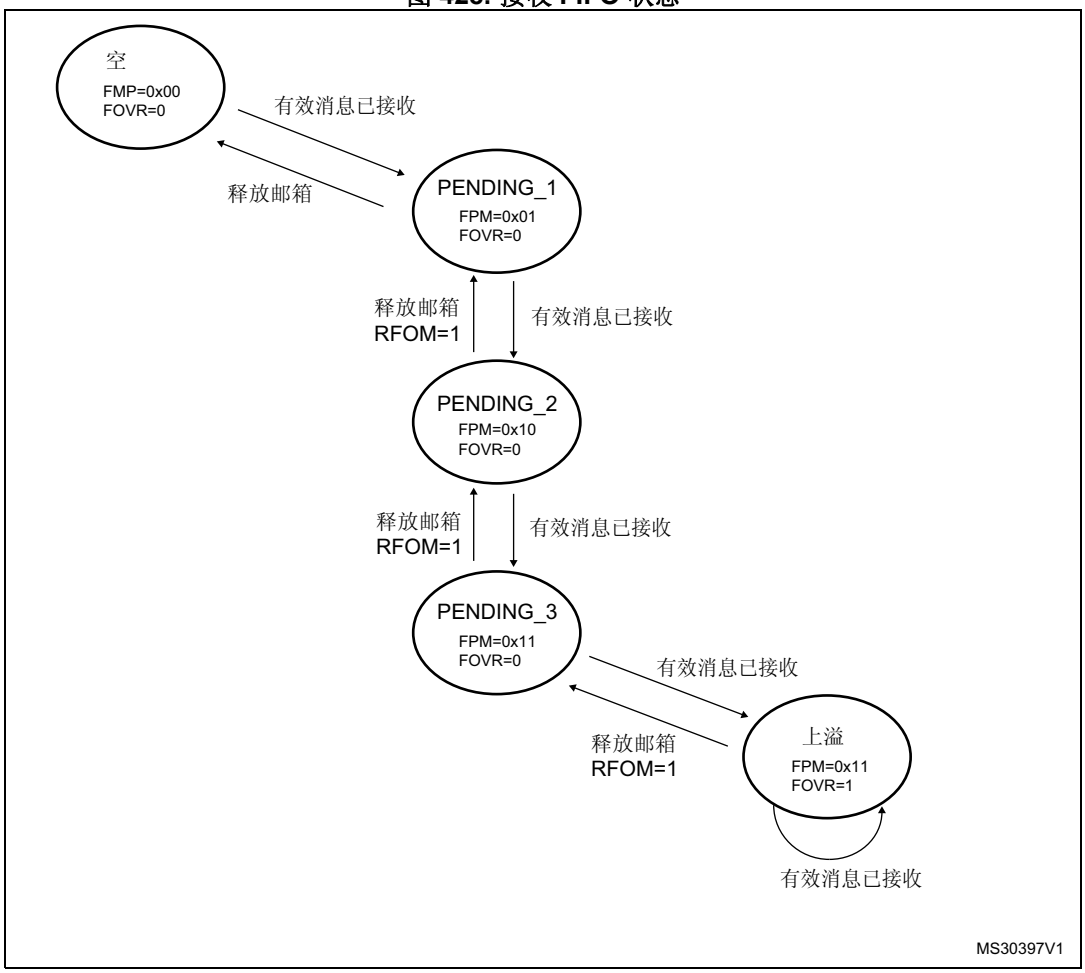
36.7.3 接收处理

为了接收 CAN 消息，提供了构成 FIFO 的三个邮箱。为了节约 CPU 负载，简化软件并保证数据一致性，FIFO 完全由硬件进行管理。应用程序通过 FIFO 输出邮箱访问 FIFO 中所存储的消息。

有效消息

当消息依据 CAN 协议正确接收（直到 EOF 字段的倒数第二位都没有发送错误）并且成功通过标识符筛选后，该消息将视为有效，请参见*第 36.7.4 节：标识符筛选*。

图 428. 接收 FIFO 状态



FIFO 管理

FIFO 开始时处于**空**状态，在接收的第一条有效消息存储在其中后，变为 **pending_1** 状态。硬件通过将 CAN_RFR 寄存器的 FMP[1:0] 位置为 01b 来指示该事件。消息将在 FIFO 输出邮箱中供读取。软件将读取邮箱内容，并通过将 CAN_RFR 寄存器的 RFOM 位置 1，来将邮箱释放。FIFO 随即恢复**空**状态。如果同时接收到新的有效消息，FIFO 将保持 **pending_1** 状态，新消息将在输出邮箱中供读取。

如果应用程序未释放邮箱，下一条有效消息将存储在 FIFO 中，使其进入 **pending_2** 状态 (FMP[1:0] = 10b)。下一条有效消息会重复该存储过程，同时将 FIFO 变为 **pending_3** 状态 (FMP[1:0] = 11b)。此时，软件必须通过将 RFOM 位置 1 来释放输出邮箱，从而留出一个空邮箱来存储下一条有效消息。否则，下一次接收到有效消息时，将导致消息丢失。

另请参见*第 36.7.5 节：消息存储*。

上溢

一旦 FIFO 处于 **pending_3** 状态（即三个邮箱均已满），则下一次接收到有效消息时，将导致**上溢**并丢失一条消息。硬件通过将 CAN_RFR 寄存器的 FOVR 位置 1 来指示上溢状况。丢失的消息取决于 FIFO 的配置：

- 如果禁止 FIFO 锁定功能（CAN_MCR 寄存器的 RFLM 位清零），则新传入的消息将覆盖 FIFO 中存储的最后一条消息。在这种情况下，应用程序将始终能访问到最新的消息。
- 如果使能 FIFO 锁定功能（CAN_MCR 寄存器的 RFLM 位置 1），则将丢弃最新的消息，软件将提供 FIFO 中最早的三条消息。

与接收相关的中断

消息存储到 FIFO 中后，FMP[1:0] 位即会更新，如果 CAN_IER 寄存器的 FMPIE 位置 1，将产生中断请求。

FIFO 存满消息（即存储了第三条消息）后，CAN_RFR 寄存器的 FULL 位置 1，如果 CAN_IER 寄存器的 FFIE 位置 1，将产生中断。

出现上溢时，FOVR 位将置 1，如果 CAN_IER 寄存器的 FOVIE 位置 1，将产生中断。

36.7.4 标识符筛选

在 CAN 协议中，消息的标识符与节点地址无关，但与消息内容有关。因此，发送器将消息广播给所有接收器。在接收到消息时，接收器节点会根据标识符的值来确定软件是否需要该消息。如果需要，该消息将复制到 SRAM 中。如果不需要，则必须在无软件干预的情况下丢弃该消息。

为了满足这一要求，bxCAN 控制器为应用程序提供了 28 个可配置且可调整的滤波器组 (27-0)。在其他器件中，bxCAN 控制器为应用程序提供了 14 个可配置且可调整的滤波器组 (13-0)，以便仅接收软件需要的消息。此硬件筛选功能可以节省软件筛选所需的 CPU 资源。每个滤波器组 x 均包含两个 32 位寄存器，分别是 CAN_FxR0 和 CAN_FxR1。

可调整的宽度

为了根据应用程序的需求来优化和调整滤波器，每个滤波器组可分别进行伸缩调整。根据滤波器尺度不同，一个滤波器组可以：

- 为 STDID[10:0]、EXTID[17:0]、IDE 和 RTR 位提供一个 32 位滤波器。
- 为 STDID[10:0]、RTR、IDE 和 EXTID[17:15] 位提供两个 16 位滤波器。

请参见*图 429*。

此外，滤波器还可配置为掩码模式或标识符列表模式。

掩码模式

在**掩码**模式下，标识符寄存器与掩码寄存器关联，用以指示标识符的哪些位“必须匹配”，哪些位“无关”。

标识符列表模式

在**标识符列表**模式下，掩码寄存器用作标识符寄存器。这时，不会定义一个标识符和一个掩码，而是指定两个标识符，从而使单个标识符的数量加倍。传入标识符的所有位都必须与滤波器寄存器中指定的位匹配。

滤波器组尺度和模式配置

滤波器组通过相应的 CAN_FMR 寄存器进行配置。为了配置滤波器组，必须通过将 CAN_FAR 寄存器的 FACT 位清零而将其停用。滤波器尺度通过 CAN_FS1R 寄存器的相应 FSCx 位进行配置，请参见*图 429*。相应掩码/标识符寄存器的**标识符列表**或**标识符掩码**模式通过 CAN_FMR 寄存器的 FBMx 位进行配置。

要筛选一组标识符，应将掩码/标识符寄存器配置为掩码模式。

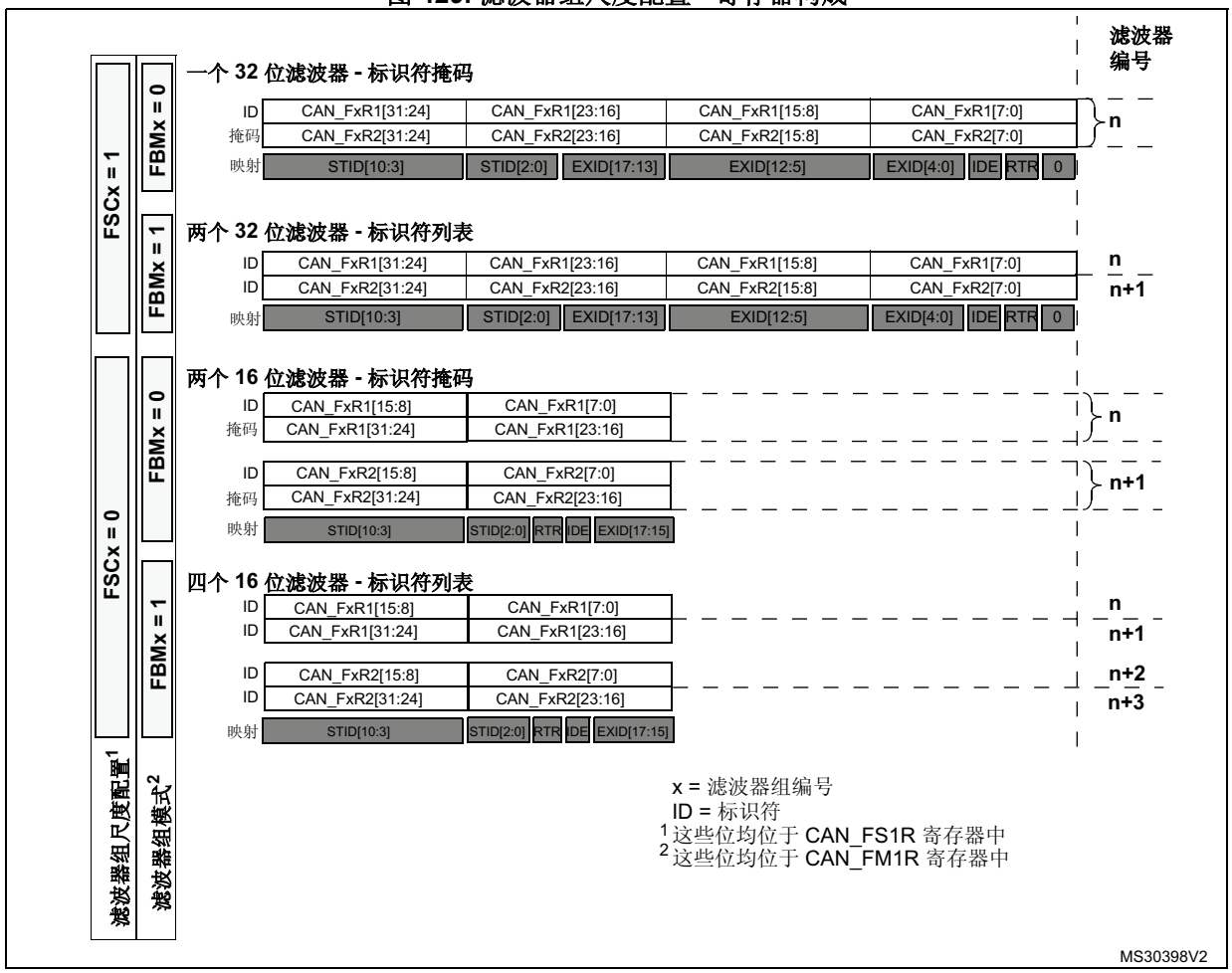
要选择单个标识符，应将掩码/标识符寄存器配置为标识符列表模式。

未由应用程序使用的滤波器应保持停用。

滤波器组中的每个滤波器将按从 0 到最大值的顺序进行编号（称为滤波器编号），具体取决于每个滤波器组的模式和尺度。

有关滤波器配置，请参见*图 429*。

图 429. 滤波器组尺度配置 - 寄存器构成



滤波器匹配索引 (Filter match index)

消息接收到 FIFO 中后，即可供应用程序使用。应用程序数据通常会复制到 SRAM 中的位置。为了将数据复制到正确的位置，应用程序必须通过标识符来识别数据。为了避免这种情况，方便访问 SRAM 位置，CAN 控制器提供了一个滤波器匹配索引。

该索引根据滤波器优先级规则与消息一同存储在邮箱中。因此，每条收到的消息都有相关联的滤波器匹配索引。

滤波器匹配索引的使用方法有两种：

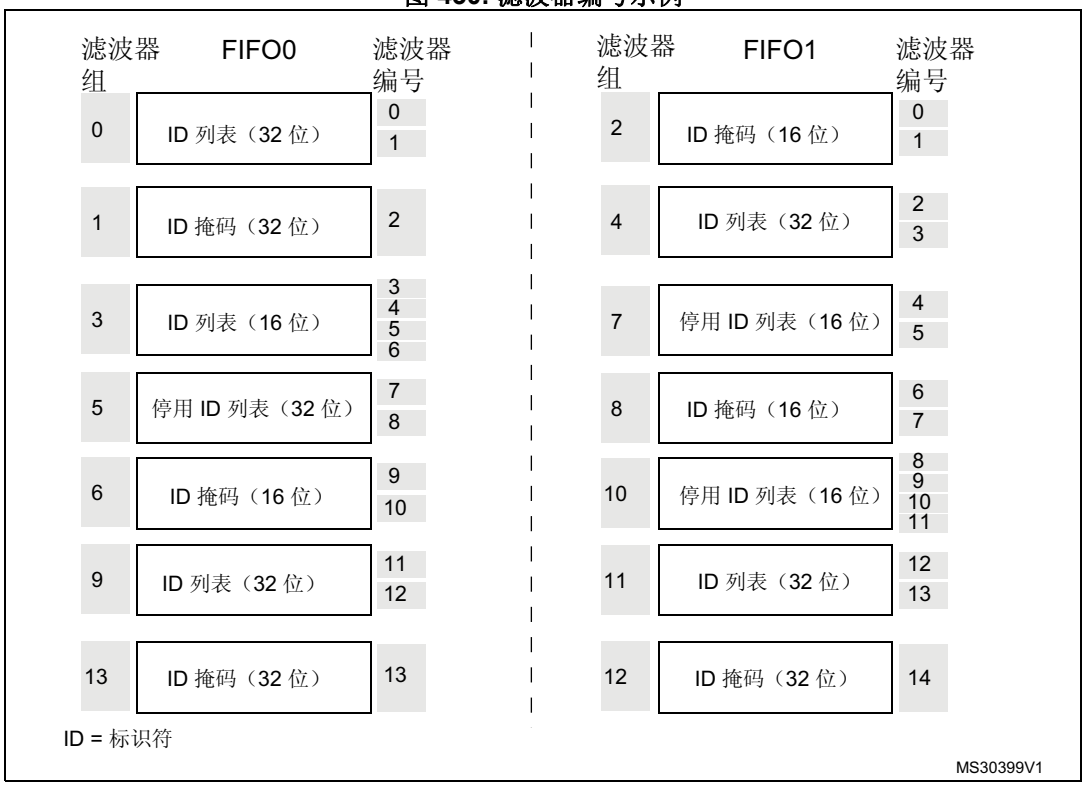
- 将滤波器匹配索引与预期值列表进行比较。
- 将滤波器匹配索引用作阵列索引，以访问数据目标位置。

对于非屏蔽滤波器，软件不再需要比较标识符。

如果滤波器有屏蔽，软件则只需比较屏蔽位。

滤波器编号的索引值与滤波器组的激活状态无关。此外，还将使用两个独立的编号方案，每个 FIFO 各一个。相关示例，请参见*图 430*。

图 430. 滤波器编号示例

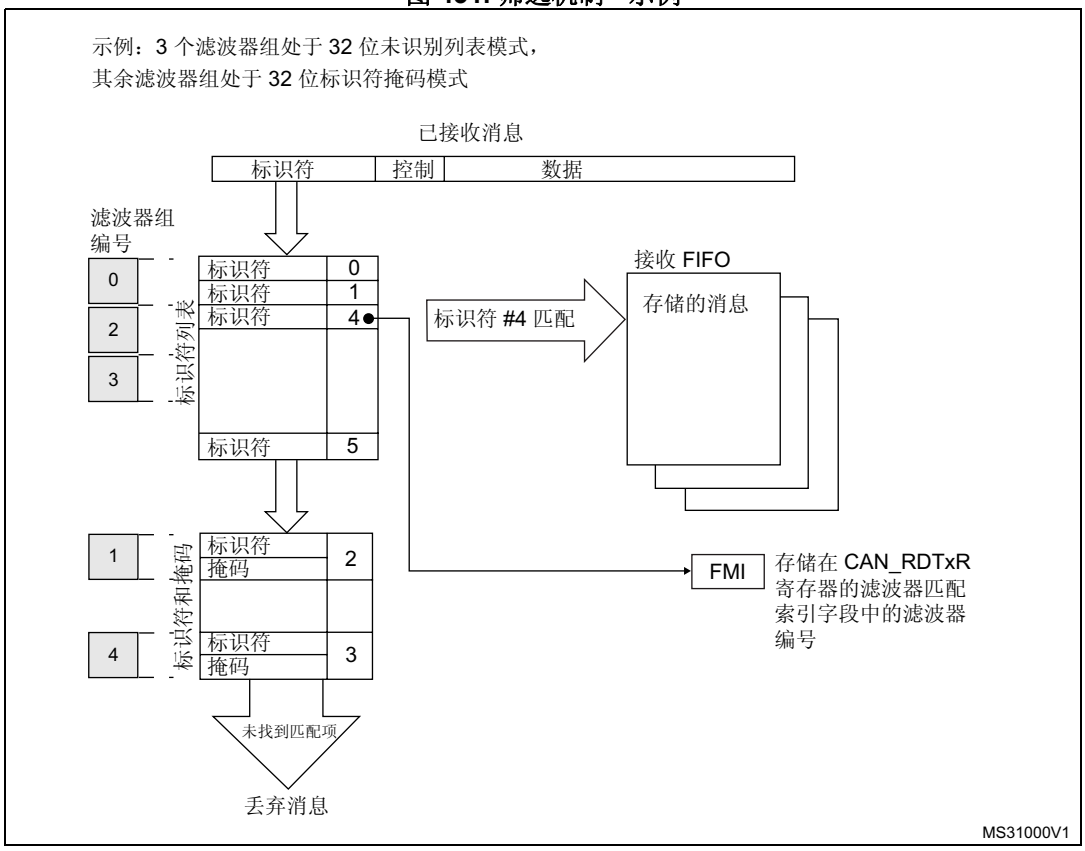


滤波器优先级规则

根据滤波器组合，可能会出现一个标识符成功通过数个滤波器的情况。这种情况下，将根据以下优先级规则选择接收邮箱中存储的滤波器匹配值：

- 32 位滤波器优先于 16 位滤波器。
- 对于尺度相等的滤波器，标识符列表模式优先于标识符掩码模式。
- 对于尺度和模式均相等的滤波器，则按滤波器编号确定优先级（编号越低，优先级越高）。

图 431. 筛选机制 - 示例



以上示例说明了 bxCAN 的筛选原则。接收到消息后，首先将标识符与标识符列表模式中配置的滤波器进行比较。如果匹配，消息将存储在相应 FIFO 中，匹配滤波器的索引则存储在滤波器匹配索引中。如本例所示，标识符与标识符 #2 匹配，因此消息内容和 FMI 2 存储到该 FIFO 中。

如果不匹配，则将传入标识符与掩码模式中配置的滤波器进行比较。

如果标识符与滤波器中配置的任何标识符均不匹配，硬件会在不干扰软件的情况下丢弃该消息。

36.7.5 消息存储

CAN 消息软件与硬件之间的接口通过邮箱实现。邮箱中包含所有与消息相关的信息：标识符、数据、控制、状态和时间戳信息。

发送邮箱

软件在空发送邮箱中设置将要发送的消息。发送状态由硬件在 CAN_TSR 寄存器中进行指示。

表 224. 发送邮箱映射

与发送邮箱基址之间的偏移	寄存器名称
0	CAN_TIxR
4	CAN_TDTxR
8	CAN_TDLxR
12	CAN_TDHxR

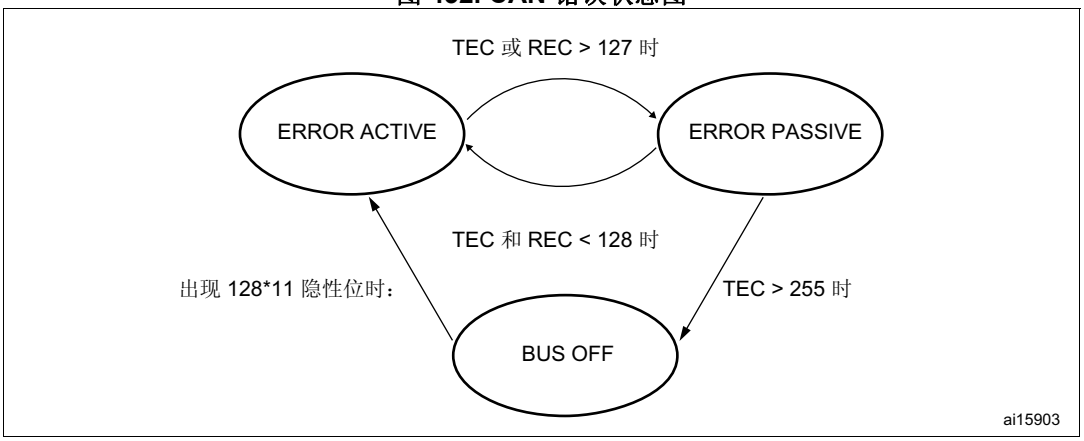
接收邮箱

消息在接收到后，将在 FIFO 输出邮箱中供软件使用。一旦软件对消息进行了处理（例如读取），则必须通过 CAN_RFR 寄存器的 RFOM 位释放 FIFO 输出邮箱，以接收下一条传入消息。滤波器匹配索引存储在 CAN_RDTxR 寄存器的 MFMI 字段中。16 位时间戳值则存储在 CAN_RDTxR 的 TIME[15:0] 字段中。

表 225. 接收邮箱映射

与接收邮箱基址之间的偏移（字节）	寄存器名称
0	CAN_RIxR
4	CAN_RDTxR
8	CAN_RDLxR
12	CAN_RDHxR

图 432. CAN 错误状态图



36.7.6 错误管理

如 CAN 协议所述，错误管理完全由硬件通过发送错误计数器（CAN_ESR 寄存器中的 TEC 值）和接收错误计数器（CAN_ESR 寄存器中的 REC 值）来处理，这两个计数器根据错误状况进行递增或递减。有关 TEC 和 REC 管理的详细信息，请参见 CAN 标准。

两者均可由软件读取，用以确定网络的稳定性。此外，CAN 硬件还将在 CAN_ESR 寄存器中提供当前错误状态的详细信息。通过 CAN_IER 寄存器（ERRIE 位等），软件可以非常灵活地配置在检测到错误时生成的中断。

总线关闭恢复

当 TEC 大于 255 时，达到总线关闭状态，该状态由 CAN_ESR 寄存器的 BOFF 位指示。在总线关闭状态下，bxCAN 不能再发送和接收消息。

bxCAN 可以自动或者应软件请求而从总线关闭状态中恢复（恢复错误主动状态），具体取决于 CAN_MCR 寄存器的 ABOM 位。但在两种情况下，bxCAN 都必须至少等待 CAN 标准中指定的恢复序列完成（在 CANRX 上监测到 128 次 11 个连续隐性位）。

如果 ABOM 位置 1，bxCAN 将在进入总线关闭状态后自动启动恢复序列。

如果 ABOM 位清零，则软件必须请求 bxCAN 先进入再退出初始化模式，从而启动恢复序列。

注： *在初始化模式下，bxCAN 不会监视 CANRX 信号，因此无法完成恢复序列。**要进行恢复，bxCAN 必须处于正常模式。***

36.7.7 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后续的边沿进行再同步。

通过将标称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。它只有一个时间片的固定长度 (1 x t_q)。
- **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。

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 (CAN_BTR) 只能在器件处于待机模式时进行配置。

注： *有关 CAN 位时序和再同步机制的详细说明，请参见 ISO 11898 标准。*

图 433. 位时序

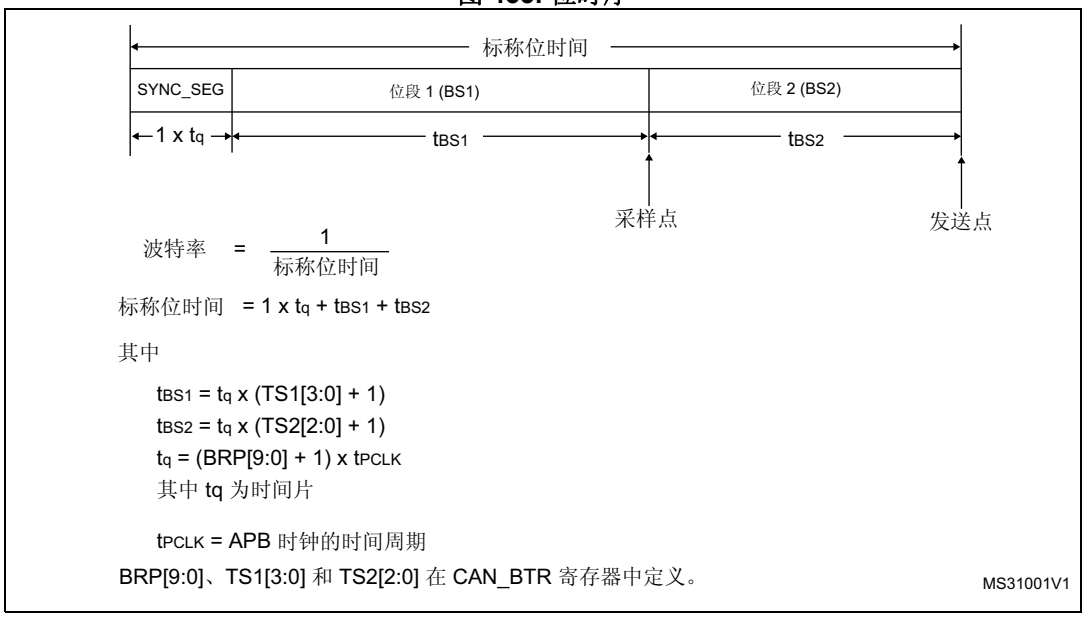
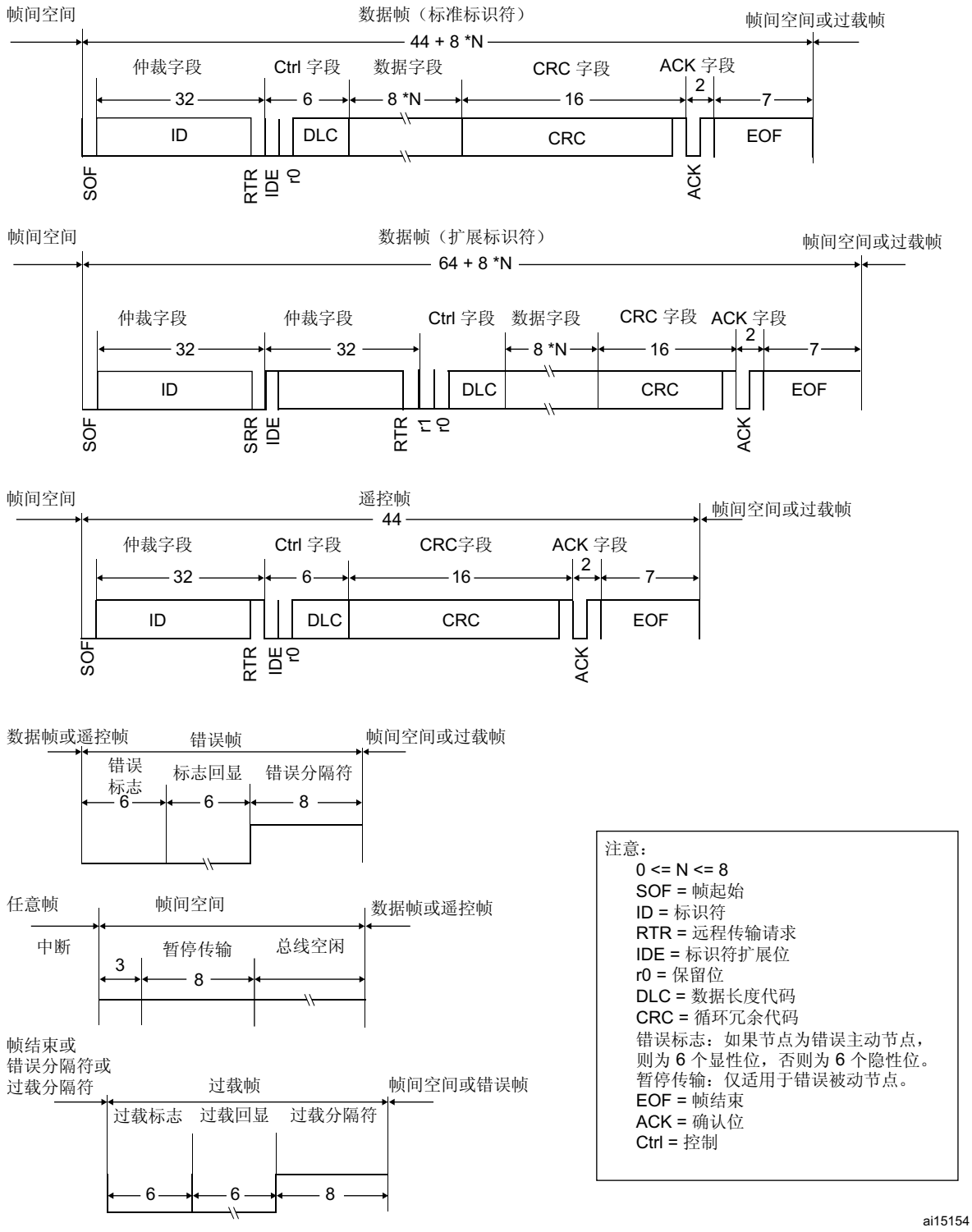


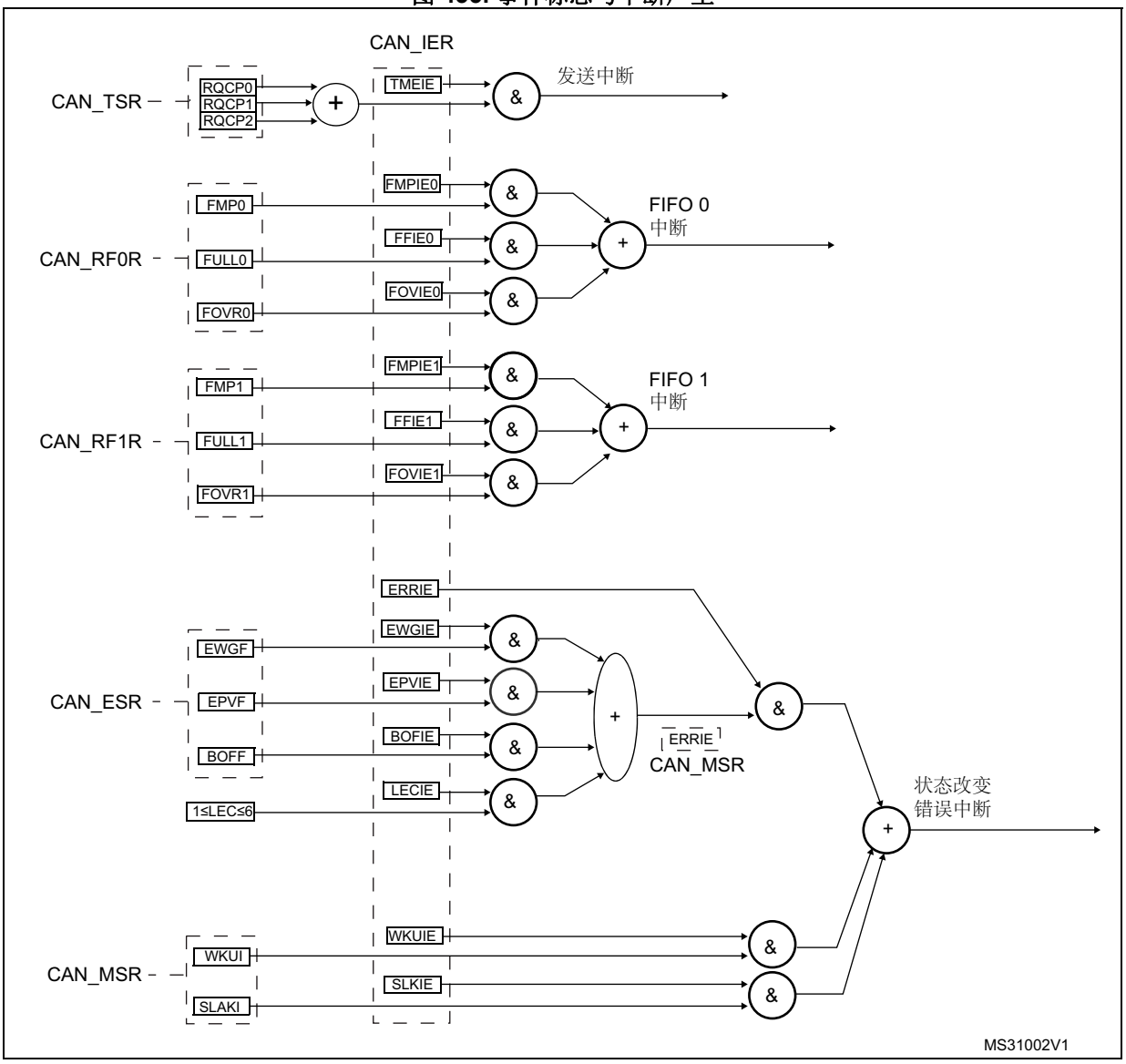
图 434. CAN 帧


帧间空间
数据帧（标准标识符）
帧间空间或过载帧
44 + 8 *N
仲裁字段
Ctrl 字段
数据字段
CRC 字段
ACK 字段
2
32
6
8 *N
16
7
ID
DLC
CRC
EOF
SOF
RTR
IDE
r0
ACK
帧间空间
数据帧（扩展标识符）
帧间空间或过载帧
64 + 8 *N
仲裁字段
仲裁字段
Ctrl 字段
数据字段
CRC 字段
ACK 字段
2
32
32
8 *N
16
7
ID
DLC
CRC
EOF
SOF
SRR
IDE
RTR
r1
r0
ACK
帧间空间
遥控帧
帧间空间或过载帧
44
仲裁字段
Ctrl 字段
CRC字段
ACK 字段
2
32
6
16
7
ID
DLC
CRC
EOF
SOF
RTR
IDE
r0
ACK
数据帧或遥控帧
错误帧
帧间空间或过载帧
错误
标志
标志回显
错误分隔符
6
6
8
任意帧
帧间空间
数据帧或遥控帧
中断
暂停传输
总线空闲
3
8
帧结束或
错误分隔符或
过载分隔符
过载帧
帧间空间或错误帧
过载标志
过载回显
过载分隔符
6
6
8
注意：
0 <= N <= 8
SOF = 帧起始
ID = 标识符
RTR = 远程传输请求
IDE = 标识符扩展位
r0 = 保留位
DLC = 数据长度代码
CRC = 循环冗余代码
错误标志：如果节点为错误主动节点，则为 6 个显性位，否则为 6 个隐性位。
暂停传输：仅适用于错误被动节点。
EOF = 帧结束
ACK = 确认位
Ctrl = 控制
ai15154

36.8 bxCAN 中断

bxCAN 共有四个专用的中断向量。每个中断源均可通过 CAN 中断使能寄存器 (CAN_IER) 来单独地使能或禁止。

图 435. 事件标志与中断产生



- **发送中断**可由以下事件产生：
  - 发送邮箱 0 变为空，CAN_TSR 寄存器的 RQCP0 位置 1。
  - 发送邮箱 1 变为空，CAN_TSR 寄存器的 RQCP1 位置 1。
  - 发送邮箱 2 变为空，CAN_TSR 寄存器的 RQCP2 位置 1。
- **FIFO 0 中断**可由以下事件产生：
  - 接收到新消息，CAN_RF0R 寄存器的 FMP0 位不是“00”。
  - FIFO0 满，CAN_RF0R 寄存器的 FULL0 位置 1。
  - FIFO0 上溢，CAN_RF0R 寄存器的 FOVR0 位置 1。

- **FIFO 1 中断**可由以下事件产生：
  - 接收到新消息，CAN_RF1R 寄存器的 FMP1 位不是“00”。
  - FIFO1 满，CAN_RF1R 寄存器的 FULL1 位置 1。
  - FIFO1 上溢，CAN_RF1R 寄存器的 FOVR1 位置 1。
- **错误和状态改变中断**可由以下事件产生：
  - 错误状况，有关错误状况的更多详细信息，请参见 CAN 错误状态寄存器 (CAN_ESR)。
  - 唤醒状况，CAN Rx 信号上监测到 SOF。
  - 进入睡眠模式。

36.9 CAN 寄存器

外设寄存器必须按字（32 位）进行访问。

36.9.1 寄存器访问保护

错误访问某些配置寄存器可能会导致硬件暂时性地干扰整个 CAN 网络。因此，只有在 CAN 硬件处于初始化模式时，才可以通过软件修改 CAN_BTR 寄存器。

尽管传输错误的数据不会引发 CAN 网络级别的故障，但可能会对应用程序造成严重干扰。发送邮箱只能在处于空状态时通过软件进行修改，请参见*图 427：发送邮箱状态*。

可以通过停用相应滤波器组或将 FINIT 位置 1 来修改滤波器值。此外，只有在 CAN_FMR 寄存器的滤波器初始化模式位置 1 (FINIT=1) 时，才能对 CAN_FMxR、CAN_FSxR 和 CAN_FFAR 寄存器中的滤波器配置（大小、模式和 FIFO 分配）进行修改。

36.9.2 CAN 控制和状态寄存器

有关寄存器说明中使用的缩写，请参见*第 1.1 节*。

CAN 主控制寄存器 (CAN_MCR)

CAN master control register

偏移地址：0x00
复位值：0x0001 0002

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TTCM	ABOM	AWUM	NART	RFLM	TXFP	SLEEP	INRQ
rs								rw	rw	rw	rw	rw	rw	rw	rw

位 31:17 保留，必须保持复位值。

位 16 **DBF**：调试冻结 (Debug freeze)

0：调试期间 CAN 处于工作状态

1：调试期间 CAN 处于接收/发送冻结状态。接收 FIFO 仍可正常访问/控制。

位 15 **RESET**：bxCAN 软件主复位 (bxCAN software master reset)

0：正常工作。

1：强制 bxCAN 进行主复位 -> 复位后激活睡眠模式（FMP 位和 CAN_MCR 寄存器初始化为复位值）。此位自动复位为 0。

位 14:8 保留，必须保持复位值。

位 7 **TTCM**：时间触发通信模式 (Time triggered communication mode)

0：禁止时间触发通信模式。

1：使能时间触发通信模式

注： 有关时间触发通信模式的更多信息，请参见第 36.7.2 节：时间触发通信模式 (Time triggered communication mode)。

位 6 **ABOM:**：自动总线关闭管理 (Automatic bus-off management)

此位控制 CAN 硬件在退出总线关闭状态时的行为。

0：在软件发出请求后，一旦监测到 128 次连续 11 个隐性位，并且软件将 CAN_MCR 寄存器的 INRQ 位先置 1 再清零，即退出总线关闭状态。

1：一旦监测到 128 次连续 11 个隐性位，即通过硬件自动退出总线关闭状态。

有关总线关闭状态的详细信息，请参见第 36.7.6 节：错误管理。

位 5 **AWUM**：自动唤醒模式 (Automatic wakeup mode)

此位控制 CAN 硬件在睡眠模式下接收到消息时的行为。

0：在软件通过将 CAN_MCR 寄存器的 SLEEP 位清零发出请求后，退出睡眠模式。

1：一旦监测到 CAN 消息，即通过硬件自动退出睡眠模式。

CAN_MCR 寄存器的 SLEEP 位和 CAN_MCR 寄存器的 SLAK 位由硬件清零。

位 4 **NART**：禁止自动重发送 (No automatic retransmission)

0：CAN 硬件将自动重发送消息，直到根据 CAN 标准成功发送消息。

1：无论发送结果如何（成功、错误或仲裁丢失），消息均只发送一次。

位 3 **RFLM**：接收 FIFO 锁定模式 (Receive FIFO locked mode)

0：接收 FIFO 上溢后不锁定。接收 FIFO 装满后，下一条传入消息将覆盖前一条消息。

1：接收 FIFO 上溢后锁定。接收 FIFO 装满后，下一条传入消息将被丢弃。

位 2 **TXFP**：发送 FIFO 优先级 (Transmit FIFO priority)

此位用于控制在几个邮箱同时挂起时的发送顺序。

0：优先级由消息标识符确定

1：优先级由请求顺序（时间顺序）确定

位 1 **SLEEP**：睡眠模式请求 (Sleep mode request)

此位由软件置 1，用于请求 CAN 硬件进入睡眠模式。一旦当前 CAN 活动（发送或接收 CAN 帧）结束，即进入睡眠模式。

此位由软件清零时，将退出睡眠模式。

当 AWUM 位置 1 以及在 CAN RX 信号上检测到 SOF 位时，硬件即将此位清零。

复位后，此位将置 1 - CAN 启动睡眠模式。

位 0 **INRQ**：初始化请求 (Initialization request)

软件通过将此位清零，来将硬件切换到正常模式。一旦在 Rx 信号上监测到连续 11 个隐性位，CAN 硬件即完成同步并准备进行发送和接收。硬件通过将 CAN_MSR 寄存器的 INAK 位清零来指示此事件。

软件通过将此位置 1 来请求 CAN 硬件进入初始化模式。一旦软件将 INRQ 位置 1，CAN 硬件将等待当前 CAN 活动（发送或接收）结束，然后进入初始化模式。硬件通过将 CAN_MSR 寄存器的 INAK 位置 1 来指示此事件。

CAN 主状态寄存器 (CAN_MSR)

CAN master status register

偏移地址：0x04
复位值：0x0000 0C02

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	RX	SAMP	RXM	TXM	Res.	Res.	Res.	SLAKI	WKUI	ERRI	SLAK	INAK
				r	r	r	r				rc_w1	rc_w1	rc_w1	r	r

位 31:12 保留，必须保持复位值。

位 11 **RX**：CAN Rx 信号 (CAN Rx signal)
监视 **CAN_RX** 引脚的实际值。

位 10 **SAMP**：上一个采样点 (Last sample point)
上一个采样点的 RX 值（当前接收的位值）。

位 9 **RXM**：接收模式 (Receive mode)
CAN 硬件当前为接收器。

位 8 **TXM**：发送模式 (Transmit mode)
CAN 硬件当前为发送器。

位 7:5 保留，必须保持复位值。

位 4 **SLAKI**：睡眠确认中断 (Sleep acknowledge interrupt)
当 SLKIE=1 时，硬件将此位置 1，以指示 bxCAN 已经进入睡眠模式。如果 CAN_IER 寄存器的 SLKIE 位置 1，则当此位置 1 后将产生状态改变中断。
SLAK 清零后，此位由软件或硬件清零。
注： SLKIE=0 时，无法对 SLAKI 进行轮询。在这种情况下，可以轮询 SLAK 位。

位 3 **WKUI**：唤醒中断 (Wakeup interrupt)
此位由硬件置 1，用于指示在 CAN 硬件处于睡眠模式期间检测到一个 SOF 位。如果 CAN_IER 寄存器的 WKUIE 位置 1，则当此位置 1 后将产生状态改变中断。
此位由软件清零。

位 2 **ERRI**：错误中断 (Error interrupt)
如果在检测到错误时 CAN_ESR 的一个位置 1，并且使能 CAN_IER 中的相应中断，则硬件将此位置 1。如果 CAN_IER 寄存器的 ERRIE 位置 1，则当此位置 1 后将产生状态改变中断。
此位由软件清零。

位 1 **SLAK**：睡眠确认 (Sleep acknowledge)

此位由硬件置 1，用于向软件指示 CAN 硬件此时处于睡眠模式。此位可确认软件的睡眠模式请求（CAN_MCR 寄存器的 SLEEP 位置 1）。
CAN 硬件退出睡眠模式（以在 CAN 总线上进行同步）时，此位由硬件清零。要进行同步，硬件必须在 CAN RX 信号上监测到由 11 个连续隐性位组成的一个序列。

注： CAN_MCR 寄存器的 SLEEP 位清零时，将触发退出睡眠模式程序。有关 SLEEP 位清零的详细信息，请参阅 CAN_MCR 寄存器 AWUM 位的说明。

位 0 **INAK**：初始化确认 (Initialization acknowledge)

此位由硬件置 1，用于向软件指示 CAN 硬件此时处于初始化模式。此位可确认软件的初始化请求（CAN_MCR 寄存器的 INRQ 位置 1）。
CAN 硬件退出初始化模式（以在 CAN 总线上进行同步）时，此位由硬件清零。要进行同步，硬件必须在 CAN RX 信号上监测到由 11 个连续隐性位组成的一个序列。

CAN 发送状态寄存器 (CAN_TSR)

CAN transmit status register

偏移地址：0x08
复位值：0x1C00 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOW2	LOW1	LOW0	TME2	TME1	TME0	CODE[1:0]		ABRQ2	Res.	Res.	Res.	TERR2	ALST2	TXOK2	RQCP2
r	r	r	r	r	r	r	r	rs				rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABRQ1	Res.	Res.	Res.	TERR1	ALST1	TXOK1	RQCP1	ABRQ0	Res.	Res.	Res.	TERR0	ALST0	TXOK0	RQCP0
rs				rc_w1	rc_w1	rc_w1	rc_w1	rs				rc_w1	rc_w1	rc_w1	rc_w1

位 31 **LOW2**：邮箱 2 最低优先级标志 (Lowest priority flag for mailbox 2)

当多个邮箱挂起等待发送且邮箱 2 优先级最低时，此位由硬件置 1。

位 30 **LOW1**：邮箱 1 最低优先级标志 (Lowest priority flag for mailbox 1)

当多个邮箱挂起等待发送且邮箱 1 优先级最低时，此位由硬件置 1。

位 29 **LOW0**：邮箱 0 最低优先级标志 (Lowest priority flag for mailbox 0)

当多个邮箱挂起等待发送且邮箱 0 优先级最低时，此位由硬件置 1。

注： 当只有一个邮箱挂起时，LOW[2:0] 位置为零。

位 28 **TME2**：发送邮箱 2 空 (Transmit mailbox 2 empty)

当邮箱 2 没有挂起的发送请求时，此位由硬件置 1。

位 27 **TME1**：发送邮箱 1 空 (Transmit mailbox 1 empty)

当邮箱 1 没有挂起的发送请求时，此位由硬件置 1。

位 26 **TME0**：发送邮箱 0 空 (Transmit mailbox 0 empty)

当邮箱 0 没有挂起的发送请求时，此位由硬件置 1。

位 25:24 **CODE[1:0]**：邮箱代码 (Mailbox code)

如果至少一个发送邮箱空闲，代码值等于下一个空闲发送邮箱的编号。
如果所有发送邮箱均挂起，则代码值等于优先级最低的发送邮箱的编号。

位 23 **ABRQ2**：邮箱 2 中止请求 (Abort request for mailbox 2)
由软件置 1，用于中止相应邮箱的发送请求。
邮箱变为空后，此位由硬件清零。
邮箱未挂起等待发送时，将此位置 1 没有任何作用。

位 22:20 保留，必须保持复位值。

位 19 **TERR2**：邮箱 2 发送错误 (Transmission error of mailbox 2)
如果上一次发送因错误而失败，此位将置 1。

位 18 **ALST2**：邮箱 2 仲裁丢失 (Arbitration lost for mailbox 2)
如果上一次发送因仲裁丢失而失败，此位将置 1。

位 17 **TXOK2**：邮箱 2 发送成功 (Transmission OK of mailbox 2)
每次发送尝试后，硬件都将更新此位。
0：上一次发送失败
1：上一次发送成功
当邮箱 2 的发送请求成功完成时，此位由硬件置 1。请参见*图 427*。

位 16 **RQCP2**：邮箱 2 请求完成 (Request completed mailbox 2)
最后一个请求（发送或中止）执行完毕时，由硬件置 1。
由软件通过写入“1”清零，或是在发生发送请求（CAN_TMID2R 寄存器中的 TXRQ2 位置 1）时由硬件清零。
如果将此位清零，邮箱 2 的所有状态位（TXOK2、ALST2 和 TERR2）都将清零。

位 15 **ABRQ1**：邮箱 1 中止请求 (Abort request for mailbox 1)
由软件置 1，用于中止相应邮箱的发送请求。
邮箱变为空后，此位由硬件清零。
邮箱未挂起等待发送时，将此位置 1 没有任何作用。

位 14:12 保留，必须保持复位值。

位 11 **TERR1**：邮箱 1 发送错误 (Transmission error of mailbox 1)
如果上一次发送因错误而失败，此位将置 1。

位 10 **ALST1**：邮箱 1 仲裁丢失 (Arbitration lost for mailbox 1)
如果上一次发送因仲裁丢失而失败，此位将置 1。

位 9 **TXOK1**：邮箱 1 发送成功 (Transmission OK of mailbox 1)
每次发送尝试后，硬件都将更新此位。
0：上一次发送失败
1：上一次发送成功
当邮箱 1 的发送请求成功完成时，此位由硬件置 1。请参见*图 427*。

位 8 **RQCP1**：邮箱 1 请求完成 (Request completed mailbox 1)
最后一个请求（发送或中止）执行完毕时，由硬件置 1。
由软件通过写入“1”清零，或是在发生发送请求（CAN_TI1R 寄存器中的 TXRQ1 位）时由硬件清零。
如果将此位清零，邮箱 1 的所有状态位（TXOK1、ALST1 和 TERR1）都将清零。

位 7 **ABRQ0**：邮箱 0 中止请求 (Abort request for mailbox 0)
由软件置 1，用于中止相应邮箱的发送请求。
邮箱变为空后，此位由硬件清零。
邮箱未挂起等待发送时，将此位置 1 没有任何作用。

位 6:4 保留，必须保持复位值。

位 3 **TERR0**：邮箱 0 发送错误 (Transmission error of mailbox 0)
如果上一次发送因错误而失败，此位将置 1。

位 2 **ALST0**：邮箱 0 仲裁丢失 (Arbitration lost for mailbox 0)
如果上一次发送因仲裁丢失而失败，此位将置 1。

位 1 **TXOK0**：邮箱 0 发送成功 (Transmission OK of mailbox 0)
每次发送尝试后，硬件都将更新此位。
0：上一次发送失败
1：上一次发送成功
当邮箱 1 的发送请求成功完成时，此位由硬件置 1。请参见*图 427*。

位 0 **RQCP0**：邮箱 0 请求完成 (Request completed mailbox 0)
最后一个请求（发送或中止）执行完毕时，由硬件置 1。
由软件通过写入“1”清零，或是在发生发送请求（CAN_TI0R 寄存器的 TXRQ0 位置 1）时由硬件清零。
如果将此位清零，邮箱 0 的所有状态位（TXOK0、ALST0 和 TERR0）都将清零。

CAN 接收 FIFO 0 寄存器 (CAN_RF0R)

CAN receive FIFO 0 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.	RFOM0	FOVR0	FULL0	Res.	FMP0[1:0]	
										rs	rc_w1	rc_w1		r	r

位 31:6 保留，必须保持复位值。

位 5 **RFOM0**：释放 FIFO 0 输出邮箱 (Release FIFO 0 output mailbox)
由软件置 1，用于释放 FIFO 的输出邮箱。FIFO 中至少有一条消息挂起时，才能释放输出邮箱。FIFO 为空时，将此位置 1 没有任何作用。如果 FIFO 中至少有两条消息挂起，软件必须释放输出邮箱，才能访问下一条消息。
输出邮箱释放后，此位由硬件清零。

位 4 **FOVR0**：FIFO 0 上溢 (FIFO 1 overrun)
FIFO 填满时，如果接收到新消息并且通过滤波器，此位将由硬件置 1。
此位由软件清零。

位 3 **FULL0**：FIFO 0 满 (FIFO 0 full)
FIFO 中存储三条消息后，由硬件置 1。
此位由软件清零。

位 2 保留，必须保持复位值。

位 1:0 **FMP0[1:0]**：FIFO 0 消息挂起 (FIFO 0 message pending)
这些位用于指示接收 FIFO 中挂起的消息数。
硬件每向 FIFO 存储一条新消息，FMP 即会增加。软件每次通过将 RFOM0 位置 1 来释放输出邮箱，FMP 即会减小。

CAN 接收 FIFO 1 寄存器 (CAN_RF1R)

CAN receive FIFO 1 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.	Res.	Res.	Res.	Res.	RFOM1	FOVR1	FULL1	Res.	FMP1[1:0]	
										rs	rc_w1	rc_w1		r	r

位 31:6 保留，必须保持复位值。

位 5 **RFOM1**：释放 FIFO 1 输出邮箱 (Release FIFO 1 output mailbox)

由软件置 1，用于释放 FIFO 的输出邮箱。FIFO 中至少有一条消息挂起时，才能释放输出邮箱。FIFO 为空时，将此位置 1 没有任何作用。如果 FIFO 中至少有两条消息挂起，软件必须释放输出邮箱，才能访问下一条消息。
输出邮箱释放后，此位由硬件清零。

位 4 **FOVR1**：FIFO 1 上溢 (FIFO 1 overrun)

FIFO 填满时，如果接收到新消息并且通过滤波器，此位将由硬件置 1。
此位由软件清零。

位 3 **FULL1**：FIFO 1 满 (FIFO 1 full)

FIFO 中存储三条消息后，由硬件置 1。
此位由软件清零。

位 2 保留，必须保持复位值。

位 1:0 **FMP1[1:0]**：FIFO 1 消息挂起 (FIFO 1 message pending)

这些位用于指示接收 FIFO1 中挂起的消息数。
硬件每向 FIFO1 存储一条新消息，FMP1 即会增加。软件每次通过将 RFOM1 位置 1 来释放输出邮箱，FMP 即会减小。

CAN 中断使能寄存器 (CAN_IER)

CAN interrupt enable 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.	SLKIE	WKUIE
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRIE	Res.	Res.	Res.	LEC IE	BOF IE	EPV IE	EWG IE	Res.	FOV IE1	FF IE1	FMP IE1	FOV IE0	FF IE0	FMP IE0	TME IE
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 **SLKIE**：睡眠中断使能 (Sleep interrupt enable)
0：SLAKI 位置 1 时不产生中断。
1：SLAKI 位置 1 时产生中断。

位 16 **WKUIE**：唤醒中断使能 (Wakeup interrupt enable)
0：WKUI 置 1 时不产生中断。
1：WKUI 位置 1 时产生中断。

位 15 **ERRIE**：错误中断使能 (Error interrupt enable)
0：CAN_ESR 中有挂起的错误状况时，不会产生中断。
1：CAN_ESR 中有挂起的错误状况时，会产生中断。

位 14:12 保留，必须保持复位值。

位 11 **LECIE**：上一个错误代码中断使能 (Last error code interrupt enable)
0：如果在检测到错误后硬件将 LEC[2:0] 中的错误代码置 1，则不会将 ERRI 位置 1。
1：如果在检测到错误后硬件将 LEC[2:0] 中的错误代码置 1，会将 ERRI 位置 1。

位 10 **BOFIE**：总线关闭中断使能 (Bus-off interrupt enable)
0：BOFF 置 1 时，不会将 ERRI 位置 1。
1：BOFF 置 1 时，将 ERRI 位置 1。

位 9 **EPVIE**：错误被动中断使能 (Error passive interrupt enable)
0：EPVF 置 1 时，不会将 ERRI 位置 1。
1：EPVF 置 1 时，将 ERRI 位置 1。

位 8 **EWGIE**：错误警告中断使能 (Error warning interrupt enable)
0：EWGF 置 1 时，不会将 ERRI 位置 1。
1：EWGF 置 1 时，将 ERRI 位置 1。

位 7 保留，必须保持复位值。

位 6 **FOVIE1**：FIFO 上溢中断使能 (FIFO overrun interrupt enable)。
0：FOVR 置 1 时不产生中断。
1：FOVR 置 1 时产生中断。

位 5 **FFIE1**：FIFO 变满时中断使能 (FIFO full interrupt enable)
0：FULL 位置 1 时不产生中断。
1：FULL 位置 1 时产生中断。

位 4 **FMPIE1**：FIFO 消息挂起中断使能 (FIFO message pending interrupt enable)
0：FMP[1:0] 位的状态不是 00b 时，不产生中断。
1：FMP[1:0] 位的状态不是 00b 时，产生中断。

位 3 **FOVIE0**：FIFO 上溢中断使能 (FIFO overrun interrupt enable)。
0：FOVR 位置 1 时不产生中断。
1：FOVR 位置 1 时产生中断。

位 2 **FFIE0**：FIFO 变满时中断使能 (FIFO full interrupt enable)
0：FULL 位置 1 时不产生中断。
1：FULL 位置 1 时产生中断。

位 1 **FMPIE0**：FIFO 消息挂起中断使能 (FIFO message pending interrupt enable)
0：FMP[1:0] 位的状态不是 00b 时，不产生中断。
1：FMP[1:0] 位的状态不是 00b 时，产生中断。

位 0 **TMEIE**：发送邮箱空中断使能 (Transmit mailbox empty interrupt enable)
0：RQCPx 位置 1 时不产生中断。
1：RQCPx 位置 1 时产生中断。

注：　请参见第 36.8 节：bxCAN 中断。

CAN 错误状态寄存器 (CAN_ESR)

CAN error status 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
REC[7:0]								TEC[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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LEC[2:0]			Res.	BOFF	EPVF	EWGF
									rw	rw	rw		r	r	r

位 31:24 **REC[7:0]**：接收错误计数器 (Receive error counter)
CAN 协议故障隔离机制的实施部分。如果接收期间发生错误，该计数器按 1 或 8 递增，具体取决于 CAN 标准所定义的错误状况。每次成功接收后，该计数器按 1 递减，如果其数值大于 128，则复位为 120。计数器值超过 127 时，CAN 控制器进入错误被动状态。

位 23:16 **TEC[7:0]**：9 位发送错误计数器最低有效字节 (Least significant byte of the 9-bit transmit error counter)
CAN 协议故障隔离机制的实施部分。

位 15:7 保留，必须保持复位值。

位 6:4 **LEC[2:0]**：上一个错误代码 (Last error code)
该字段由硬件置 1，其中的代码指示 CAN 总线上检测到的上一个错误的错误状况。如果消息成功传送（接收或发送）且未发生错误，该字段将清为“0”。
LEC[2:0] 位可由软件置为 0b111 值。这些位由硬件更新，以指示当前通信状态。
000：无错误
001：填充错误
010：格式错误
011：确认错误
100：位隐性错误
101：位显性错误
110：CRC 错误
111：由软件置 1

位 3 保留，必须保持复位值。

位 2 **BOFF**：总线关闭标志 (Bus-off flag)
此位由硬件在进入睡眠状态时置 1。TEC 上溢（超过 255）时，进入总线关闭状态，请参见*第 1157 页上的第 36.7.6 节*。

位 1 **EPVF**：错误被动标志 (Error passive flag)
达到错误被动极限（接收错误计数器或发送错误计数器 > 127）时，此位由硬件置 1。

位 0 **EWGF**：错误警告标志 (Error warning flag)
达到警告极限时，此位由硬件置 1
（接收错误计数器或发送错误计数器≥96）。

CAN 位时序寄存器 (CAN_BTR)

CAN bit timing register

偏移地址：0x1C
复位值：0x0123 0000

只有 CAN 硬件处于初始化模式时，才能由软件访问此寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SILM	LBKM	Res.	Res.	Res.	Res.	SJW[1:0]		Res.	TS2[2:0]			TS1[3:0]			
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.	BRP[9:0]									
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **SILM**：静默模式（调试）(Silent mode (debug))
0：正常工作
1：静默模式

位 30 **LBKM**：环回模式（调试）(Loop back mode (debug))
0：禁止环回模式
1：使能环回模式

位 29:26 保留，必须保持复位值。

位 25:24 **SJW[1:0]**：再同步跳转宽度 (Resynchronization jump width)
这些位定义 CAN 硬件在执行再同步时最多可以将位加长或缩短的时间片数目。
$t_{RJW} = t_q \times (SJW[1:0] + 1)$

位 23 保留，必须保持复位值。

位 22:20 **TS2[2:0]**：时间段 2 (Time segment 2)
这些位定义时间段 2 中的时间片数目。
$t_{BS2} = t_q \times (TS2[2:0] + 1)$

位 19:16 **TS1[3:0]**：时间段 1 (Time segment 1)
这些位定义时间段 1 中的时间片数目
$t_{BS1} = t_q \times (TS1[3:0] + 1)$
有关位时序的详细信息，请参见*第 1157 页上的第 36.7.7 节：位时序*。

位 15:10 保留，必须保持复位值。

位 9:0 **BRP[9:0]**：波特率预分频器 (Baud rate prescaler)
这些位定义一个时间片的长度。
$t_q = (BRP[9:0]+1) \times t_{PCLK}$

36.9.3 CAN 邮箱寄存器

本章介绍发送和接收邮箱的寄存器。有关详细的寄存器映射，请参见*第 1156 页上的第 36.7.5 节：消息存储*。

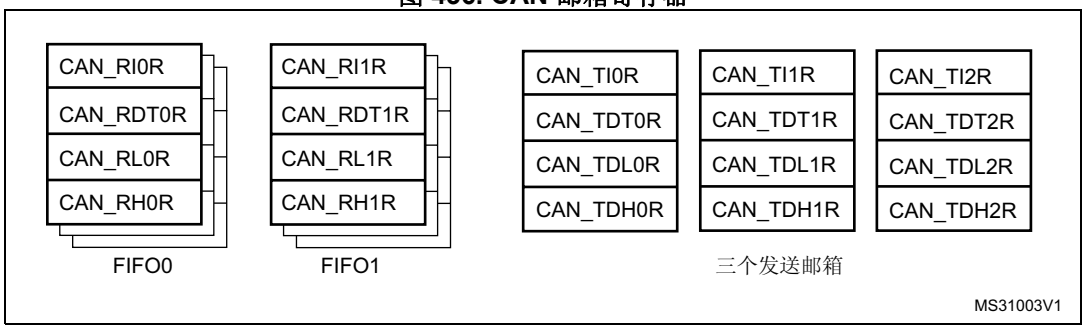
除了以下情况外，发送邮箱和接收邮箱使用相同的寄存器：

- CAN_RDTxR 寄存器中的 FMI 字段。
- 接收邮箱始终处于写保护状态。
- 发送邮箱仅在为空时才处于可写状态（CAN_TSR 寄存器中的相应 TME 位置 1）。

发送邮箱共有 3 个，接收邮箱共有 2 个。每个接收邮箱允许访问一个深度为 3 级的 FIFO，访问仅限于 FIFO 中最早接收到的消息。

每个邮箱由 4 个寄存器组成。

图 436. CAN 邮箱寄存器



CAN 发送邮箱标识符寄存器 (CAN_TIxR) (x=0..2)

CAN TX mailbox identifier register

偏移地址：0x180、0x190、0x1A0
复位值：0xXXXX XXXX（位 0 除外，TXRQ = 0）

当邮箱处于发送挂起状态（TMEx 复位）时，所有发送寄存器均为写保护状态。

该寄存器还会进行发送请求控制（位 0）- 复位值 0。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STID[10:0]/EXID[28:18]											EXID[17:13]				
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
EXID[12:0]													IDE	RTR	TXRQ
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:21 **STID[10:0]/EXID[28:18]**：标准标识符或扩展标识符 (Standard identifier or extended identifier)
标准标识符或扩展标识符的 MSB（取决于 IDE 位的值）。

位 20:3 **EXID[17:0]**：扩展标识符 (Extended identifier)
扩展标识符的 LSB。

位 2 **IDE**：标识符扩展 (Identifier extension)
此位用于定义邮箱中消息的标识符类型。
0：标准标识符。
1：扩展标识符。

位 1 **RTR**：远程发送请求 (Remote transmission request)
0：数据帧
1：远程帧

位 0 **TXRQ**：发送邮箱请求 (Transmit mailbox request)
由软件置 1，用于请求发送相应邮箱的内容。
邮箱变为空后，此位由硬件清零。

CAN 邮箱数据长度控制和时间戳寄存器 (CAN_TDTxR) (x = 0..2)

CAN mailbox data length control and time stamp register

当邮箱未处于空状态时，该寄存器的所有位均为写保护状态。

偏移地址：0x184、0x194、0x1A4
复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIME[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.	DLC[3:0]			
												rw	rw	rw	rw

位 31:16 **TIME[15:0]**：消息时间戳 (Message time stamp)
此字段包含在进行 SOF 发送时所捕获的 16 位定时器值。

位 15:9 保留，必须保持复位值。

位 8 **TGT**：发送全局时间 (Transmit global time)
只有硬件处于时间触发通信模式（CAN_MCR 寄存器的 TTCM 位置 1）时，此位才会激活。
0：不发送时间戳 TIME[15:0]。
1：在 8 字节消息的最后两个数据字节中发送时间戳 TIME[15:0] 的值。数据字节 7 对应 TIME[7:0]，数据字节 6 对应 TIME[15:8]，该值将替换 CAN_TDHxR[31:16] 寄存器（DATA6[7:0] 和 DATA7[7:0]）中写入的数据。DLC 必须编程为 8，才能通过 CAN 总线发送这两个字节。

位 7:4 保留，必须保持复位值。

位 3:0 **DLC[3:0]**：数据长度代码 (Data length code)
该字段定义数据帧或遥控帧请求中的数据字节数。
一条消息可以包含 0 到 8 个数据字节，具体取决于 DLC 字段的值。

CAN 邮箱数据低位寄存器 (CAN_TDLxR) (x=0..2)

CAN mailbox data low register

当邮箱未处于空状态时，该寄存器的所有位均为写保护状态。

偏移地址：0x188、0x198、0x1A8
复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA3[7:0]								DATA2[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
DATA1[7:0]								DATA0[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **DATA3[7:0]**：数据字节 3 (Data byte 3)
消息的数据字节 3。

位 23:16 **DATA2[7:0]**：数据字节 2 (Data byte 2)
消息的数据字节 2。

位 15:8 **DATA1[7:0]**：数据字节 1 (Data byte 1)
消息的数据字节 1。

位 7:0 **DATA0[7:0]**：数据字节 0 (Data byte 0)
消息的数据字节 0。
一条消息可以包含 0 到 8 个数据字节，从字节 0 开始。

CAN 邮箱数据高位寄存器 (CAN_TDHxR) (x=0..2)

CAN mailbox data high register

当邮箱未处于空状态时，该寄存器的所有位均为写保护状态。

偏移地址：0x18C、0x19C、0x1AC
复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA7[7:0]								DATA6[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
DATA5[7:0]								DATA4[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 **DATA7[7:0]**：数据字节 7 (Data byte 7)
消息的数据字节 7。
注： 如果 TTCM 以及此消息的 TGT 有效，则 DATA7 和 DATA6 将以时间戳值替换。

位 23:16 **DATA6[7:0]**：数据字节 6 (Data byte 6)
消息的数据字节 6。

位 15:8 **DATA5[7:0]**：数据字节 5 (Data byte 5)
消息的数据字节 5。

位 7:0 **DATA4[7:0]**：数据字节 4 (Data byte 4)
消息的数据字节 4。

CAN 接收FIFO 邮箱标识符寄存器 (CAN_RIxR) (x = 0..1)

CAN receive FIFO mailbox identifier register

偏移地址：0x1B0、0x1C0
复位值：0xXXXX XXXX

所有接收寄存器均受到写保护。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STID[10:0]/EXID[28:18]											EXID[17:13]				
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
EXID[12:0]													IDE	RTR	Res
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	

位 31:21 **STID[10:0]/EXID[28:18]**：标准标识符或扩展标识符 (Standard identifier or extended identifier)
标准标识符或扩展标识符的 MSB（取决于 IDE 位的值）。

位 20:3 **EXID[17:0]**：扩展标识符 (Extended identifier)
扩展标识符的 LSB。

位 2 **IDE**：标识符扩展 (Identifier extension)
此位用于定义邮箱中消息的标识符类型。
0：标准标识符。
1：扩展标识符。

位 1 **RTR**：远程发送请求 (Remote transmission request)
0：数据帧
1：远程帧

位 0 保留，必须保持复位值。

CAN 接收 FIFO 邮箱数据长度控制和时间戳寄存器 (CAN_RDTxR) (x = 0..1)

CAN receive FIFO mailbox data length control and time stamp register

偏移地址：0x1B4、0x1C4
复位值：0xXXXX XXXX

所有接收寄存器均受到写保护。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIME[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
FMI[7:0]								Res.	Res.	Res.	Res.	DLC[3:0]			
r	r	r	r	r	r	r	r					r	r	r	r

位 31:16 **TIME[15:0]**：消息时间戳 (Message time stamp)
此字段包含在进行 SOF 检测时所捕获的 16 位定时器值。

位 15:8 **FMI[7:0]**：滤波器匹配索引 (Filter match index)
该寄存器包含滤波器索引，邮箱中存储的消息需要经过该滤波器。有关标识符筛选的更多详细信息，请参见*第 1152 页上的第 36.7.4 节：标识符筛选* - **滤波器匹配索引**一段。

位 7:4 保留，必须保持复位值。

位 3:0 **DLC[3:0]**：数据长度代码 (Data length code)
该字段定义一个数据帧所包含的数据字节数（0 到 8）。如果是远程帧请求，则为 0。

CAN 接收 FIFO 邮箱数据低位寄存器 (CAN_RDLxR) (x = 0..1)

CAN receive FIFO mailbox data low register

当邮箱未处于空状态时，该寄存器的所有位均为写保护状态。

偏移地址：0x1B8、0x1C8
复位值：0xXXXX XXXX

所有接收寄存器均受到写保护。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA3[7:0]								DATA2[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
DATA1[7:0]								DATA0[7:0]							
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:24 **DATA3[7:0]**：数据字节 3 (Data byte 3)
消息的数据字节 3。

位 23:16 **DATA2[7:0]**：数据字节 2 (Data byte 2)
消息的数据字节 2。

位 15:8 **DATA1[7:0]**：数据字节 1 (Data byte 1)
消息的数据字节 1。

位 7:0 **DATA0[7:0]**：数据字节 0 (Data byte 0)
消息的数据字节 0。
一条消息可以包含 0 到 8 个数据字节，从字节 0 开始。

CAN 接收 FIFO 邮箱数据高位寄存器 (CAN_RDHxR) (x = 0..1)

CAN receive FIFO mailbox data high register

偏移地址：0x1BC、0x1CC
复位值：0xXXXX XXXX

所有接收寄存器均受到写保护。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA7[7:0]								DATA6[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
DATA5[7:0]								DATA4[7:0]							
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:24 **DATA7[7:0]**：数据字节 7 (Data Byte 7)
消息的数据字节 3。

位 23:16 **DATA6[7:0]**：数据字节 6 (Data Byte 6)
消息的数据字节 2。

位 15:8 **DATA5[7:0]**：数据字节 5 (Data Byte 5)
消息的数据字节 1。

位 7:0 **DATA4[7:0]**：数据字节 4 (Data Byte 4)
消息的数据字节 0。

36.9.4 CAN 滤波器寄存器

CAN 滤波器主寄存器 (CAN_FMR)

CAN filter master register

偏移地址：0x200
复位值：0x2A1C 0E01

此寄存器的所有位均由软件置 1 和清零。

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

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

位 31:14 保留，必须保持复位值。

位 13:8 **CANSB[5:0]**：CAN 起始存储区 (CAN start bank)
这些位将由软件置 1 和清零。它们为处于 1 到 27 范围内的 CAN 接口（从模式）定义起始存储区。

位 7:1 保留，必须保持复位值。

位 0 **FINIT**：滤波器初始化模式 (Filter initialization mode)
滤波器组的初始化模式
0：滤波器工作模式。
1：滤波器初始化模式。

CAN 滤波器模式寄存器 (CAN_FM1R)

CAN filter mode register

偏移地址：0x204
复位值：0x0000 0000

仅当 CAN_FMR 寄存器中设置了滤波器初始化模式 (FINIT=1) 时，才能对此寄存器执行写操作。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	FBM27	FBM26	FBM25	FBM24	FBM23	FBM22	FBM21	FBM20	FBM19	FBM18	FBM17	FBM16
				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
FBM15	FBM14	FBM13	FBM12	FBM11	FBM10	FBM9	FBM8	FBM7	FBM6	FBM5	FBM4	FBM3	FBM2	FBM1	FBM0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

注： *请参见第 1153 页上的图 429：滤波器组尺度配置 - 寄存器构成。*

位 31:28 保留，必须保持复位值。

位 27:0 **FBM*x***：滤波器模式 (Filter mode)

滤波器 x 的寄存器的模式。

0：滤波器组 x 的两个 32 位寄存器处于标识符屏蔽模式。

1：滤波器组 x 的两个 32 位寄存器处于标识符列表模式。

注： 只有在连接线器件中，位 27:14 才可用，否则保留。

CAN 滤波器尺度寄存器 (CAN_FS1R)

CAN filter scale register

偏移地址：0x20C
复位值：0x0000 0000

仅当 CAN_FMR 寄存器中设置了滤波器初始化模式 (FINIT=1) 时，才能对此寄存器执行写操作。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	FSC27	FSC26	FSC25	FSC24	FSC23	FSC22	FSC21	FSC20	FSC19	FSC18	FSC17	FSC16
				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
FSC15	FSC14	FSC13	FSC12	FSC11	FSC10	FSC9	FSC8	FSC7	FSC6	FSC5	FSC4	FSC3	FSC2	FSC1	FSC0
rw	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 27:0 **FSC*x***：滤波器尺度配置 (Filter scale configuration)

这些位用于定义滤波器 13-0 尺度配置。

0：双 16 位尺度配置

1：单 32 位尺度配置

注： 只有在连接线器件中，位 27:14 才可用，否则保留。

注： 请参见第 1153 页上的图 429：滤波器组尺度配置 - 寄存器构成。

CAN 滤波器 FIFO 分配寄存器 (CAN_FFA1R)

CAN filter FIFO assignment register

偏移地址：0x214
复位值：0x0000 0000

仅当 CAN_FMR 寄存器中设置了滤波器初始化模式 (FINIT=1) 时，才能对此寄存器执行写操作。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	FFA27	FFA26	FFA25	FFA24	FFA23	FFA22	FFA21	FFA20	FFA19	FFA18	FFA17	FFA16
				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
FFA15	FFA14	FFA13	FFA12	FFA11	FFA10	FFA9	FFA8	FFA7	FFA6	FFA5	FFA4	FFA3	FFA2	FFA1	FFA0
rw	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 27:0 **FFA*x***：滤波器 x 的滤波器 FIFO 分配 (Filter FIFO assignment for filter x)
通过此滤波器的消息将存储在指定的 FIFO 中。
0：滤波器分配给 FIFO 0
1：滤波器分配给 FIFO 1

注： 只有在连接线器件中，位 27:14 才可用，否则保留。

CAN 滤波器激活寄存器 (CAN_FA1R)

CAN filter activation register

偏移地址：0x21C
复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	FACT27	FACT26	FACT25	FACT24	FACT23	FACT22	FACT21	FACT20	FACT19	FACT18	FACT17	FACT16
				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
FACT15	FACT14	FACT13	FACT12	FACT11	FACT10	FACT9	FACT8	FACT7	FACT6	FACT5	FACT4	FACT3	FACT2	FACT1	FACT0
rw	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 27:0 **FACT*x***：滤波器激活 (Filter active)
软件将此位置 1 可激活滤波器 x。要修改滤波器 x 寄存器 (CAN_FxR[0:7])，必须将 FACTx 位清零，或将 CAN_FMR 寄存器中的 FINIT 位置 1。
0：滤波器 x 未激活
1：滤波器 x 激活

注： 只有在连接线器件中，位 27:14 才可用，否则保留。

滤波器组 i 寄存器 x (CAN_FiRx)（i = 0..27，x = 1，2）

Filter bank i register x

偏移地址：0x240 到 0x31C
复位值：0xXXXX XXXX

共有 28 个滤波器组，其中，*i= 0 到* 27。每个滤波器组 i 由两个 32 位寄存器组成，即 CAN_FiR[2:1]。

仅当 CAN_FAxR 寄存器的 FACTx 位清零，或者 CAN_FMR 寄存器的 FINIT 位置 1 时，才能修改此寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FB31	FB30	FB29	FB28	FB27	FB26	FB25	FB24	FB23	FB22	FB21	FB20	FB19	FB18	FB17	FB16
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
FB15	FB14	FB13	FB12	FB11	FB10	FB9	FB8	FB7	FB6	FB5	FB4	FB3	FB2	FB1	FB0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

在所有配置中：

位 31:0 **FB[31:0]**：滤波器位 (Filter bits)

标识符
寄存器的每一位用于指定预期标识符相应位的级别。
0：需要显性位
1：需要隐性位

掩码
寄存器的每一位用于指定相关标识符寄存器的位是否必须与预期标识符的相应位匹配。
0：无关，不使用此位进行比较
1：必须匹配，传入标识符的此位必须与滤波器相应标识符寄存器中指定的级别相同。

注： *每个寄存器的功能可能因滤波器的尺度和模式配置而异。有关滤波器映射、功能说明和掩码寄存器关联的信息，请参见第 1152 页上的第 36.7.4 节：标识符筛选。*

掩码模式中，掩码/标识符寄存器的位映射与**标识符列表**模式中的相同。

有关滤波器组的寄存器映射/地址信息，请参见第 1181 页上的表 226。

36.9.5 bxCAN 寄存器映射

有关寄存器边界地址的信息，请参见第 64 页上的第 2.2.2 节。寄存器仅在 CAN1 中位于偏移量为 0x200 到 31C 的地址处。

表 226. bxCAN 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	**CAN_MCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBF	RESET	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TTCM	ABOM	AWUM	NART	RFLM	TXFP	SLEEP	INRQ
	Reset value																1	0								0	0	0	0	0	0	1	0
0x004	**CAN_MSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RX	SAMP	RXM	TXM	Res.	Res.	Res.	SLAKI	WKUI	ERRI	SLAK	INAK
	Reset value																					1	1	0	0				0	0	0	1	0
0x008	**CAN_TSR**	LOW[2:0]			TME[2:0]			CODE[1:0]		ABRQ2	Res.	Res.	Res.	TERR2	ALST2	TXOK2	RQCP2	ABRQ1	Res.	Res.	Res.	TERR1	ALST1	TXOK1	RQCP1	ABRQ0	Res.	Res.		TERR0	ALST0	TXOK0	RQCP0
	Reset value	0	0	0	1	1	1	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0
0x00C	**CAN_RF0R**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RFOM0	FOVR0	FULL0	Res.	FMP0[1:0]	
	Reset value																											0	0	0		0	0
0x010	**CAN_RF1R**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RFOM1	FOVR1	FULL1	Res.	FMP1[1:0]	
	Reset value																											0	0	0		0	0
0x014	**CAN_IER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SLKIE	WKUIE	ERRIE	Res.	Res.	Res.	LECIE	BOFIE	EPVIE	EWGIE	Res.	FOVIE1	FFIE1	FMPIE1	FOVIE0	FFIE0	FMPIE0	TMEIE
	Reset value															0	0	0				0	0	0	0		0	0	0	0	0	0	0
0x018	**CAN_ESR**	REC[7:0]								TEC[7:0]									Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LEC[2:0]			Res.	BOFF	EPVF	EWGF
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0										0	0	0		0	0	0
0x01C	**CAN_BTR**	SILM	LBKM	Res.	Res.	Res.	Res.	SJW[1:0]		Res.	TS2[2:0]			TS1[3:0]					Res.	Res.	Res.	Res.	Res.	Res.	BRP[9:0]									
	Reset value	0	0					0	0		0	1	0	0	0	1	1							0	0	0	0	0	0	0	0	0	0
0x020-0x17F		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x180	**CAN_TI0R**	STID[10:0]/EXID[28:18]								EXID[17:0]																					IDE	RTR	TXRQ
	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	0

表 226. bxCAN 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
0x184	**CAN_TDT0R**	TIME[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	TGT	Res.	Res.	Res.	Res.	DLC[3:0]			
	Reset value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x								x					x	x	x	x
0x188	**CAN_TDL0R**	DATA3[7:0]								DATA2[7:0]								DATA1[7:0]								DATA0[7:0]							
	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
0x18C	**CAN_TDH0R**	DATA7[7:0]								DATA6[7:0]								DATA5[7:0]								DATA4[7:0]							
	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
0x190	**CAN_TI1R**	STID[10:0]/EXID[28:18]											EXID[17:0]																		IDE	RTR	TXRQ
	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	0
0x194	**CAN_TDT1R**	TIME[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	TGT	Res.	Res.	Res.	Res.	DLC[3:0]			
	Reset value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x								x					x	x	x	x
0x198	**CAN_TDL1R**	DATA3[7:0]								DATA2[7:0]								DATA1[7:0]								DATA0[7:0]							
	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
0x19C	**CAN_TDH1R**	DATA7[7:0]								DATA6[7:0]								DATA5[7:0]								DATA4[7:0]							
	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
0x1A0	**CAN_TI2R**	STID[10:0]/EXID[28:18]											EXID[17:0]																		IDE	RTR	TXRQ
	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	0
0x1A4	**CAN_TDT2R**	TIME[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	TGT	Res.	Res.	Res.	Res.	DLC[3:0]			
	Reset value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x								x					x	x	x	x
0x1A8	**CAN_TDL2R**	DATA3[7:0]								DATA2[7:0]								DATA1[7:0]								DATA0[7:0]							
	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
0x1AC	**CAN_TDH2R**	DATA7[7:0]								DATA6[7:0]								DATA5[7:0]								DATA4[7:0]							
	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
0x1B0	**CAN_RI0R**	STID[10:0]/EXID[28:18]											EXID[17:0]																		IDE	RTR	Res.
	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		

表 226. bxCAN 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
0x1B4	**CAN_RDT0R**	TIME[15:0]																FMI[7:0]								Res.	Res.	Res.	Res.	DLC[3:0]			
	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
0x1B8	**CAN_RDL0R**	DATA3[7:0]								DATA2[7:0]								DATA1[7:0]								DATA0[7:0]							
	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
0x1BC	**CAN_RDH0R**	DATA7[7:0]								DATA6[7:0]								DATA5[7:0]								DATA4[7:0]							
	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
0x1C0	**CAN_RI1R**	STID[10:0]/EXID[28:18]											EXID[17:0]																		IDE	RTR	Res.
	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	
0x1C4	**CAN_RDT1R**	TIME[15:0]																FMI[7:0]								Res.	Res.	Res.	Res.	DLC[3:0]			
	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
0x1C8	**CAN_RDL1R**	DATA3[7:0]								DATA2[7:0]								DATA1[7:0]								DATA0[7:0]							
	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
0x1CC	**CAN_RDH1R**	DATA7[7:0]								DATA6[7:0]								DATA5[7:0]								DATA4[7:0]							
	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
0x1D0-0x1FF		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x200	**CAN_FMR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CANSB[5:0]						Res.	Res.	Res.	Res.	Res.	Res.	Res.	FINIT
	Reset value																			0	0	1	1	1	0								1
0x204	**CAN_FM1R**	Res.	Res.	Res.	Res.	FBM[27:0]																											
	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
0x208		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x20C	**CAN_FS1R**	Res.	Res.	Res.	Res.	FSC[27:0]																											
	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
0x210		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x214	**CAN_FFA1R**	Res.	Res.	Res.	Res.	FFA[27:0]																											
	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

表 226. bxCAN 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
0x218		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x21C	**CAN_FA1R**	Res.	Res.	Res.	Res.	FACT[27:0]																											
	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
0x220		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x224-0x23F		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x240	**CAN_F0R1**	FB[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
0x244	**CAN_F0R2**	FB[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
0x248	**CAN_F1R1**	FB[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
0x24C	**CAN_F1R2**	FB[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
. . . .	. . . .	. . . .																															
0x318	**CAN_F27R1**	FB[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
0x31C	**CAN_F27R2**	FB[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

37 USB on-the-go 全速/高速 (OTG_FS/OTG_HS)

37.1 前言

Portions Copyright (c) 2004, 2005 Synopsys, Inc. 保留所有权利。使用须经许可。

本节介绍了 OTG_FS/OTG_HS 控制器的架构和编程模型。

本节中使用以下缩写：

FS	全速
LS	低速
HS	高速
MAC	介质访问控制器
OTG	On-the-go
PFC	数据包 FIFO 控制器
PHY	物理层
USB	通用串行总线
UTMI	USB 2.0 收发器宏单元接口 (UTMI)
UTMI	USB 收发器宏单元接口
ULPI	UTMI+ 引脚数目少的接口
ADP	连接检测协议
LPM	链路层电源管理
BCD	电池充电检测
HNP	主机协商协议
SRP	会话请求协议

参考文档如下：

- USB On-The-Go 补充标准，第 1.3 版
- USB On-The-Go 补充标准，第 2.0 版
- 通用串行总线规范第 2.0 版
- 基于 USB 2.0 规范的 USB 2.0 链路层电源管理补充工程变更通知，2007 年 7 月 16 日
- USB 2.0 ECN 的勘误表：链路层电源管理 (LPM) - 2007 年 7 月

SB OTG 是一款双角色设备 (DRD) 控制器，同时支持从机功能和主机功能，完全符合 USB 2.0 规范的 On-The-Go 补充标准。此外，该控制器也可配置为“仅主机”模式或“仅从机”模式，完全符合 *USB 2.0 规范*。在主机模式下，OTG_HS 支持高速（HS，480 Mb/s），OTG_FS 支持全速（FS，12 Mb/s）和低速（LS，1.5 Mb/s）传输，而在从机模式下，仅支持全速（FS，12 Mb/s）传输。USB OTG 同时支持 HNP 和 SRP。OTG 模式下需要的唯一外部器件是提供 V_{BUS} 电压的电荷泵。

37.2 USB_OTG 主要特性

主要特性可分为三类：通用特性、主机模式特性和从机模式特性。

37.2.1 通用特性

OTG_FS/OTG_HS 接口的通用特性如下：

- 经 USB-IF 认证，符合通用串行总线规范第 2.0 版
- OTG HS 支持 3 个 PHY 接口
  - 片上全速 PHY
  - 连接外部全速 PHY 的 I2C 接口
  - 连接外部高速 PHY 的 ULPI 接口
- 模块内嵌的 PHY 还完全支持定义在标准规范 OTG 补充第 1.3 版中的 OTG 协议
  - 支持 A-B 器件识别（ID 线）
  - 支持主机协商协议 (HNP) 和会话请求协议 (SRP)
  - 允许主机关闭 V_{BUS} 以在 OTG 应用中节省电池电量
  - 支持通过内部比较器对 V_{BUS} 电平采取 OTG 监控
  - 支持主机到从机的角色动态切换
- 可通过软件配置为以下角色：
  - 具有 SRP 功能的 USB FS/HS 从机（B 器件）
  - 具有 SRP 功能的 USB FS/HS/LS 主机（A 器件）
  - USB On-The-Go 全速双角色设备
- 支持 FS/HS SOF 和 LS Keep-alive 令牌
  - SOF 脉冲可通过 PAD 输出
  - SOF 脉冲从内部连接到定时器 (TIMx)
  - 可配置的帧周期
  - 可配置的帧结束中断
- OTG HS 内嵌 DMA，并可软件配置 AHB 的批量传输类型。
- 具有省电功能，例如在 USB 挂起期间停止系统、关闭数字模块时钟、对 PHY 和 DFIFO 电源加以管理
- 具有采用高级 FIFO 控制的 1.25K[FS]/4K[HS] 字节专用 RAM：
  - 可将 RAM 空间划分为不同 FIFO，以便灵活有效地使用 RAM
  - 每个 FIFO 可存储多个数据包
  - 动态分配存储区
  - FIFO 大小可配置为非 2 的幂次方值，以便连续使用存储单元

37.2.2 主机模式特性

OTG_FS/OTG_HS 接口在主机模式下具有以下主要特性和要求：

- 通过外部电荷泵生成 V_{BUS} 电压。
- 多达 12[FS]/16[HS] 个主机通道（又称之为管道）：每个通道都可以动态实现重新配置，可支持任何类型的 USB 传输。
- 内置硬件调度器可：
  - 在周期性传输硬件队列中存储多达 12[FS]/16[HS] 个中断加同步传输请求
  - 在非周期性传输硬件队列中存储多达 12[FS]/16[HS] 个控制加批量传输请求
- 管理一个共享 Rx FIFO、一个周期性传输 Tx FIFO 和一个非周期性传输 Tx FIFO，以有效使用 USB 数据 RAM。

37.2.3 从机模式特性

OTG_FS/OTG_HS 接口在从机模式下具有以下主要特性：

- 1 个双向控制端点 0
- 5[FS]/7[HS] 个 IN 端点 (EP)，可配置为支持批量传输、中断传输或同步传输
- 具有 5[FS]/7[HS] 个 OUT 端点，可配置为支持批量、中断或同步传输
- 管理一个共享 Rx FIFO 和一个 Tx-OUT FIFO，以高效使用 USB 数据 RAM
- 管理多达 6[FS]/8[HS] 个专用 Tx-IN FIFO（分别用于每个使能的 IN EP），降低应用程序负荷
- 支持软断开功能。

37.3 USB_OTG 实现

表 227. STM32F75xxx 的 USB_OTG 实现(1)

USB 功能	OTG_FS	OTG_HS
设备双向端点（包括 EP0）	6	8
主机模式通道	12	16
专用 SRAM 的大小	1.2 KB	4 KB
USB 2.0 链路层电源管理 (LPM)	X	
支持 OTG 版本	1.3 & 2.0	
连接检测协议 (ADP) 支持	-	
电池充电检测 (BCD) 支持	-	

1. “X” = 支持，“-” = 不支持

37.4 USB OTG 功能说明

37.4.1 USB OTG 框图

图 437. 全速 OTG 模块框图

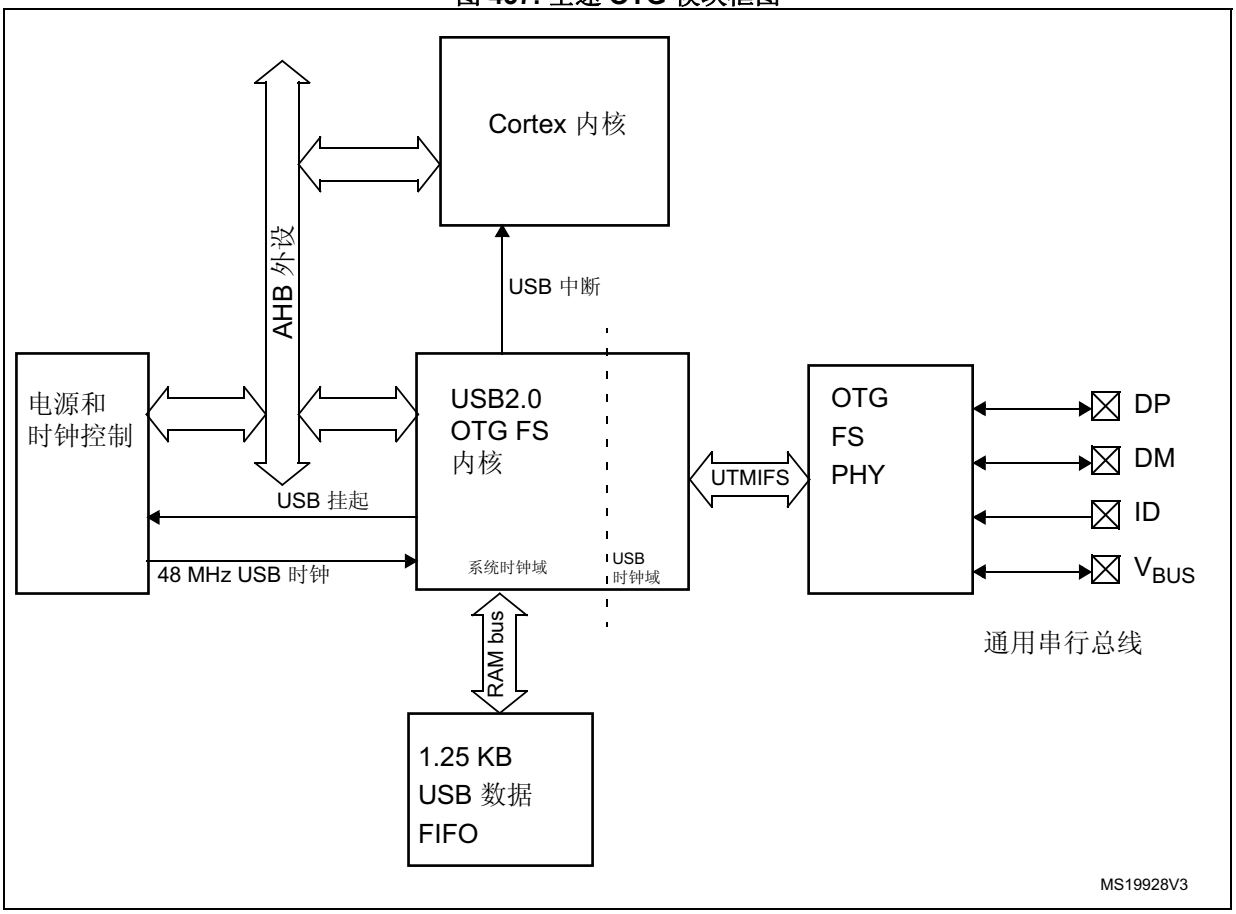
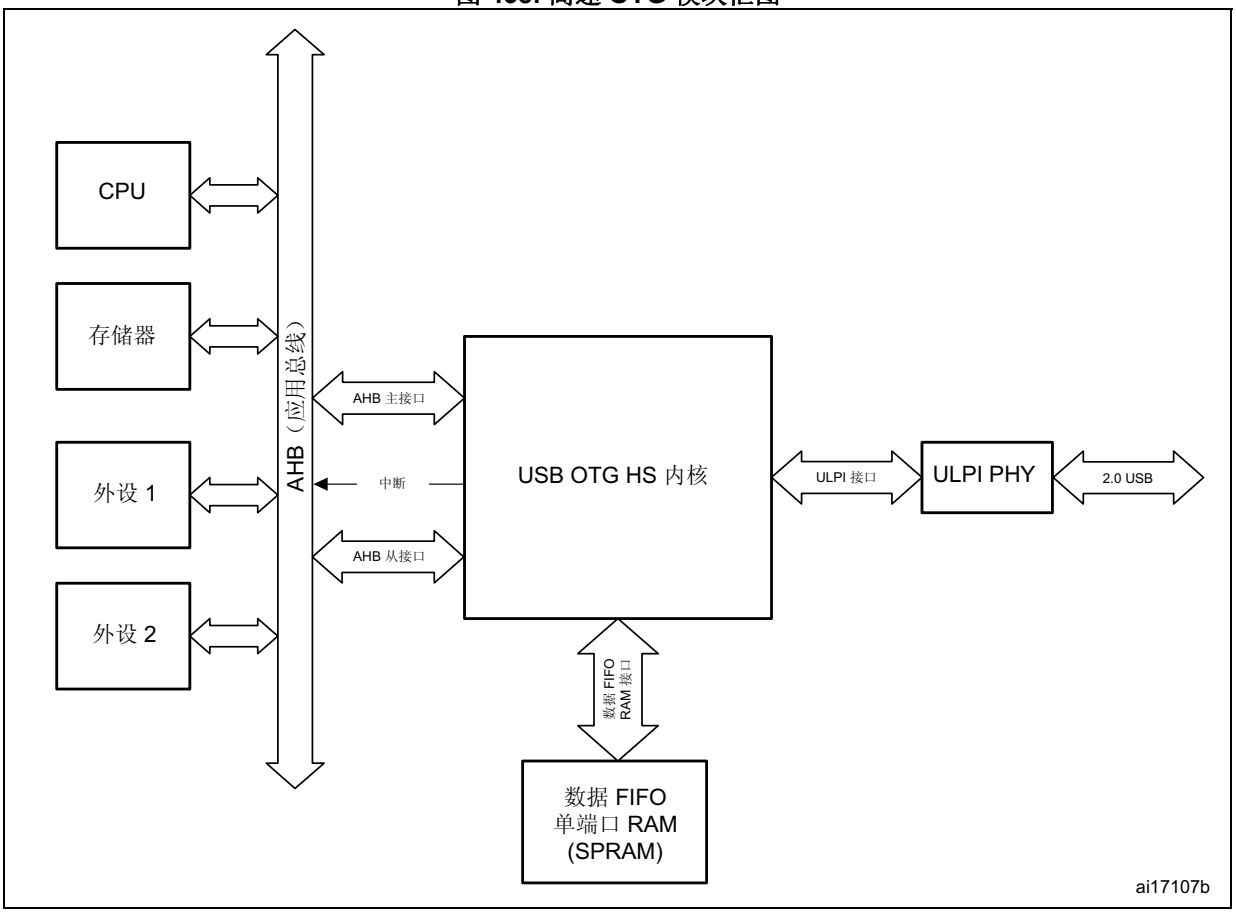


图 438. 高速 OTG 模块框图



37.4.2 OTG 模块

USB OTG 通过复位和时钟控制模块接收来自外部晶振的 48 MHz ±0.25% 的时钟。USB 时钟用于在全速通信时驱动 48 MHz 时钟域，必须在配置 OTG 模块前使能。

CPU 通过 AHB 外设总线对 OTG 模块寄存器进行读写操作， 通过*第 37.13 节: OTG_FS/OTG_HS 中断*中所述的 USB OTG 中断线接收 USB 事件通知。

CPU 通过向特定的 OTG 单元（压栈寄存器）写入 32 位字来向 USB 提交数据。数据随即自动存储到 USB 数据 RAM 中配置的数据发送 FIFO 中。每个 IN 端点（从机模式）或 OUT 通道（主机模式）都有一个 Tx FIFO 压栈寄存器。

CPU 从特定的 OTG 地址（出栈寄存器）读取 32 位字，以读取来自 USB 的数据。数据随即从在 1.25[FS]/4[HS] KB USB 数据 RAM 内配置的共享 Rx FIFO 中弹出。每个 OUT 端点或 IN 通道都有一个 Rx FIFO 出栈寄存器。

USB 协议层通过串行接口引擎 (SIE) 驱动，并通过片上物理层 (PHY) 中的收发器模块经由 USB 进行数据的串行通信， 也可通过外部高速 OTG PHY 或外部 I2C 接口的全速 OTG PHY 进行数据的串行通信。

37.4.3 全速 OTG PHY

嵌入式全速 OTG PHY 由 OTG FS 模块控制，通过 UTMI+ 总线 (UTMIFS) 的全速子集传送 USB 控制和数据信号。为 USB 连接提供物理支持。
全速 OTG PHY 包括以下组成部分：

- 供主机和设备使用的 FS/LS 收发器模块。直接在单端 USB 线上驱动发送和接收操作。
- 集成 ID 上拉电阻，用于对 ID 线进行采样，以便识别 A/B 器件。
- 由 OTG_FS 模块控制的 DP/DM 集成上拉电阻和下拉电阻，具体使能哪种电阻取决于设备的当前角色。作为外设使用时，只要检测到 V_{BUS} 为有效电平（B 会话有效），立即使能 DP 上拉电阻，以告知全速设备的连接。主机模式下则使能 DP/DM 上的下拉电阻。通过主机协商协议 (HNP) 更改设备角色时，将在上拉电阻和下拉电阻之间动态切换。
- 上拉/下拉电阻 ECN 电路。根据适用于 USB 2.0 版本的电阻 ECN 规定，DP 上拉电路包括 2 个由 OTG_FS 单独进行控制的电阻。对 DP 上拉阻值的动态调整可以提高噪声抑制能力和 Tx/Rx 信号质量。
- 带滞回功能的 V_{BUS} 感应比较器，用于检测 V_{BUS} 有效、A-B 会话有效和会话端电压阈值。执行 USB 操作期间，这些比较器用于驱动会话请求协议 (SRP)、检测会话的有效启动和结束条件，以及持续监视 V_{BUS} 供电情况。
- V_{BUS} 脉冲电路，用于在 SRP 期间通过电阻对 V_{BUS} 充电/放电（驱动力较弱）。

注意： 为确保 USB OTG FS 模块正常工作，AHB 频率应大于 14.2 MHz。

注： *本节内容仅适用于 USB OTG FS。*

37.4.4 嵌入式全速 OTG PHY

全速 OTG PHY 包括以下组成部分：

- 供主机和设备使用的 FS/LS 收发器模块。直接在单端 USB 线上驱动发送和接收操作。
- 集成 ID 上拉电阻，用于对 ID 线进行采样，以便识别 A/B 器件。
- 由 OTG_HS 模块控制的 DP/DM 集成上拉电阻和下拉电阻，具体使能哪种电阻取决于设备的当前角色。作为外设使用时，只要检测到 V_{BUS} 为有效电平（B 会话有效），立即使能 DP 上拉电阻，以告知全速设备的连接。主机模式下则使能 DP/DM 上的下拉电阻。通过主机协商协议 (HNP) 更改从机角色时，将在上拉电阻和下拉电阻之间动态切换。
- 上拉/下拉电阻 ECN 电路。根据适用于 USB 2.0 版的电阻 ECN 规定，DP 上拉电路包括 2 个由 OTG_HS 单独进行控制的电阻。对 DP 上拉阻值的动态调整可以提高噪声抑制能力和 Tx/Rx 信号质量。
- 带滞回功能的 V_{BUS} 感应比较器，用于检测 V_{BUS} 有效、A-B 会话有效和会话端电压阈值。执行 USB 操作期间，这些比较器用于驱动会话请求协议 (SRP)、检测会话的有效启动和结束条件，以及持续监视 V_{BUS} 供电情况。
- V_{BUS} 脉冲电路，用于在 SRP 期间通过电阻对 V_{BUS} 充电/放电（驱动力较弱）。

为确保 USB OTG HS 模块正常工作，AHB 频率应大于 30 MHz。

注： *本节内容仅适用于 USB OTG HS。*

37.4.5 高速 OTG PHY

USB OTG HS 模块嵌入了一个 ULPI 接口以连接外部 HS PHY。

注： *本节内容仅适用于 USB OTG HS。*

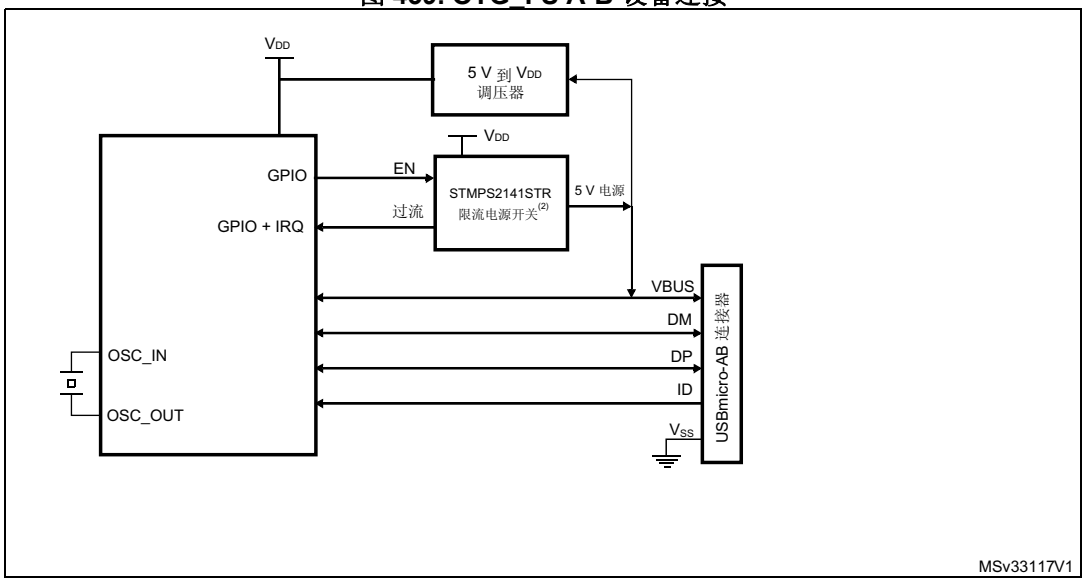
37.4.6 使用 I2C 接口的外部全速 OTG PHY

USB OTG HS 模块嵌入了一个 I2C 接口，用于连接到外部 FS PHY。

注： *本节内容仅适用于 USB OTG HS。*

37.5 OTG 双角色设备 (DRD)

图 439. OTG_FS A-B 设备连接



1. 只有在构建由 VBUS 供电的器件时才需要外部调压器。
2. 只有在应用必须支持由 VBUS 供电的器件时才需要 STMPS2141STR。如果应用电路板提供 5 V 电源，则可以使用基本电源开关。

37.5.1 ID 线检测

采取主机还是从机（默认设置）角色取决于 ID 输入引脚的电平。插入 USB 时即可根据哪一端 USB 电缆连接到 micro-AB 插座来确定 ID 线状态。

- 如果 USB 电缆的 B 端连入，其 ID 线悬空，则由于设备在 ID 线上的集成上拉电阻设备将检测到 ID 高电平并确认采取默认的从机角色。在此配置中，OTG_FS/OTG_HS 符合“USB2.0 On-The-Go 规范第 1.3 版补充标准中第 6.8.2 节 On-The-Go B 器件”中所述的 FSM 标准。
- 如果 USB 电缆的 A 端连入，其 ID 线接地，则 OTG_FS/OTG_HS 将发出 ID 线状态更改中断（OTG_GINTSTS 中的 CIDSCHG 位）以告知应用程序初始化主机，并自动切换为主机角色。在此配置中，OTG_FS/OTG_HS 符合“USB2.0 On-The-Go 规范第 1.3 版补充标准中第 6.8.1 节 On-The-Go A 器件”中所述的 FSM 标准。

37.5.2 HNP 双角色设备

全局 USB 配置寄存器中的 HNP 使能位（OTG_ GUSBCFG 中的 HNPCAP 位）可使 OTG_FS/OTG_HS 模块根据主机协商协议 (HNP) 动态切换角色，例如从 A 主机切换为 A 从机（反之亦然），或者从 B 从机切换为 B 主机（反之亦然）。通过全局 OTG 控制和状态寄存器中的连接器 ID 状态位（OTG_GOTGCTL 中的 CIDSTS 位）及全局中断和状态寄存器中的当前工作模式位（OTG_GINTSTS 中的 CMOD 位）二者的组合值可读取设备当前状态。

*第 37.16 节：OTG_FS/OTG_HS 编程模型*详细介绍了 HNP 编程模型。

37.5.3 SRP 双角色设备

全局 USB 配置寄存器中的 SRP 使能位（OTG_GUSBCFG 中的 SRPCAP 位）可使 OTG_FS/OTG_HS 模块关闭 V_{BUS} 供电，为 A 器件节省电能。注意，无论 OTG_FS/OTG_HS 采取主机角色还是从机角色，A 器件将始终负责提供 V_{BUS}。

*第 37.16 节：OTG_FS/OTG_HS 编程模型*详细介绍了 SRP A/B 器件编程模型。

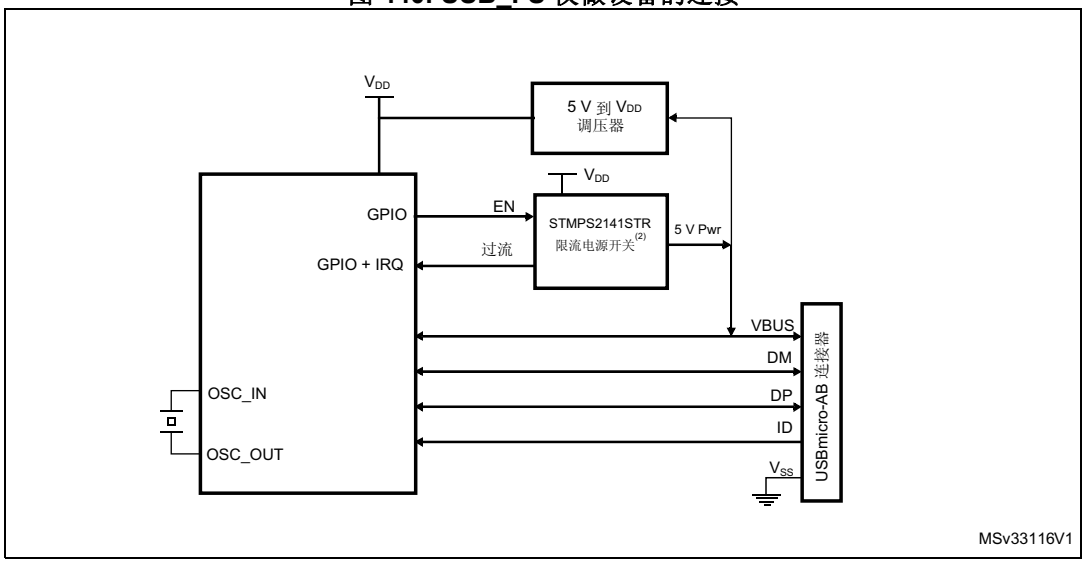
37.6 USB 设备

本节介绍 OTG_FS/OTG_HS 在 USB 设备模式下所具有的功能。在以下情形下，OTG_FS/OTG_HS 用作 USB 设备：

- OTG B 设备
  - OTG B 器件插入 USB 电缆 B 端时的默认状态
- OTG A 设备
  - OTG A 器件被 HNP 把 OTG_FS/OTG_HS 切换为设备角色后的状态
- B 设备
  - 如果 ID 线有连接，器件与 USB 电缆的 B 端相连，并且全局 USB 配置寄存器中的 HNP 功能位（OTG_GUSBCFG 中的 HNPCAP 位）清零（请参见 On-The-Go 第 1.3 版的第 6.8.3 节）。
- 仅作设备（请参见*图 440：USB_FS 仅做设备的连接*）
  - *第 37.15.4 节：OTG USB 配置寄存器 (OTG_GUSBCFG)* 中的强制设备模式位 (FDMOD) 置 1，强制 OTG_FS/OTG_HS 模块仅用作 USB 设备（请参见 On-The-Go 第 1.3 版的第 6.8.3 节）。这种情况下，即使 USB 连接器上存在 ID 线，模块也会忽略 ID 线上的电平。

注： *要在 B 设备或仅作设备配置情形下构建总线供电的设备方案，需要添加一个外部调压器，用于从 V_{BUS} 生成 V_{DD} 芯片电源。*

图 440. USB_FS 仅做设备的连接



1. 使用调压器构建总线供电设备。

37.6.1 支持 SRP 功能的 USB 设备

全局 USB 配置寄存器中的 SRP 使能位（OTG_GUSBCFG 中的 SRPCAP 位）可使 OTG_FS/OTG_HS 支持会话请求协议 (SRP)。这样一来，远程 A 器件便可以在 USB 会话挂起时，通过关闭 V_{BUS} 来节省电能。

*B 器件会话请求协议*一节详细介绍了 SRP 设备的编程模型。

37.6.2 USB 设备状态

供电状态

V_{BUS} 输入检测到 B 会话有效电压，就会使 USB 设备进入供电状态（请参见 USB2.0 第 9.1 节）。然后，OTG_FS/OTG_HS 自动连接 DP 上拉电阻，发出全速设备与主机相连的信号并生成会话请求中断（OTG_GINTSTS 中的 SRQINT 位），指示进入供电状态。

此外，V_{BUS} 输入还可确保主机在 USB 操作期间提供有效的 V_{BUS} 电平。如果检测到 V_{BUS} 降至 B 会话有效电压以下（例如，因电源干扰或主机端口关闭引发），OTG_FS/OTG_HS 将自动断开连接并生成检测到会话结束中断（OTG_GOTGINT 中的 SEDET 位），指示 OTG_FS/OTG_HS 已退出供电状态。

供电状态下，OTG_FS/OTG_HS 期望收到来自主机的复位信号。其它 USB 操作则无法执行。收到复位信号后，立即生成检测到复位中断（OTG_GINTSTS 中的 USBRST）。复位信号结束后，将生成枚举完成中断（OTG_GINTSTS 中的 ENUMDNE 位），OTG_FS/OTG_HS 随即进入默认状态。

软断开

供电状态可借助软断开功能通过软件退出。将设备控制寄存器中的软断开位（OTG_DCTL 中的 SDIS 位）置 1 即可移除 DP 上拉电阻，此时尽管没有从主机端口实际拔出 USB 电缆，但主机端仍会发生设备断开检测中断。

默认状态

默认状态下，OTG_FS/OTG_HS 期望从主机收到 SET_ADDRESS 命令。其它 USB 操作则无法执行。当 USB 上解码出有效 SET_ADDRESS 命令时，应用程序会将相应的数值写入设备配置寄存器中的设备地址字段（OTG_DCFG 中的 DAD 位）。OTG_FS/OTG_HS 随即进入地址状态，并准备好以所配置的 USB 地址对主机事务进行应答。

挂起状态

OTG_FS/OTG_HS 设备持续监视 USB 活动。在 USB 空闲时间达到 3 ms 后，将发出早期挂起中断（OTG_GINTSTS 中的 ESUSP 位），并在 3 ms 后由挂起中断（OTG_GINTSTS 中的 USBSUSP 位）确认设备进入挂起状态。然后，设备状态寄存器中的设备挂起位（OTG_DSTS 中的 SUSPSTS 位）自动置 1，OTG_FS/OTG_HS 随即进入挂起状态。

可通过设备本身退出挂起状态。这种情况下，应用程序会将设备控制寄存器中的远程唤醒信号位（OTG_DCTL 中的 RWUSIG 位）置 1，并在 1 ms 到 15 ms 后将其清零。

但若设备检测到主机发出的恢复信号时，将生成恢复中断（OTG_GINTSTS 中的 WKUPINT 位），设备挂起位自动清零。

37.6.3 USB 设备端点

OTG_FS/OTG_HS 模块实现了以下 USB 端点：

- 控制端点 0：
  - 双向且仅处理控制消息
  - 使用一组单独的寄存器来处理 IN 和 OUT 事务
  - 专用控制 (OTG_DIEPCTL0/OTG_DOEPCTL0) 寄存器、传输配置 (OTG_DIEPTSIZ0/OTG_DOEPTSIZ0) 寄存器和状态中断 (OTG_DIEPINT0/OTG_DOEPINT0) 寄存器。控制和传输大小寄存器中可用的位组与其它端点中稍有不同
- 5[FS]/7[HS] IN 端点
  - 每个端点都可配置为支持同步传输、批量传输或中断传输类型
  - 每个端点都有专用控制 (OTG_DIEPCTL*x*) 寄存器、传输配置 (OTG_DIEPTSIZ*x*) 寄存器和状态中断 (OTG_DIEPINT*x*) 寄存器
  - 设备 IN 端点通用中断屏蔽寄存器 (OTG_DIEPMSK) 可用于使能/禁止所有 IN 端点（包括 EP0）上的同一类端点中断源
  - 支持未完成的同步 IN 传输中断（OTG_GINTSTS 中的 IISOIXFR 位），该中断将在当前帧中至少有一个同步 IN 端点上的传输未完成时触发。该中断和周期帧结束中断 (OTG_GINTSTS/EOPF) 一起触发。
- 5[FS]/7[HS] OUT 端点
  - 每个端点都可配置为支持同步传输、批量传输或中断传输类型
  - 每个端点都有专用控制 (OTG_DOEPCTL*x*) 寄存器、传输配置 (OTG_DOEPTSIZ*x*) 寄存器和状态中断 (OTG_DOEPINT*x*) 寄存器
  - 设备 OUT 端点通用中断屏蔽寄存器 (OTG_DOEPMSK) 可用于使能/禁止所有 OUT 端点（包括 EP0）上的同一类端点中断源
  - 支持未完成的同步 OUT 传输中断（OTG_GINTSTS 中的 INCOMPISOOUT 位），该中断将在当前帧中至少有一个同步 OUT 端点上的传输未完成时触发。该中断和周期帧结束中断 (OTG_GINTSTS/EOPF) 一起触发

端点控制

- 应用程序可通过设备端点 x IN/OUT 控制寄存器 (OTG_DIEPCTL*x*/OTG_DOEPCTL*x*) 对端点采取以下控制：
  - 端点使能/禁止
  - 在当前配置下激活端点
  - 设置 USB 传输类型（同步、批量和中断）
  - 设置支持的数据包大小
  - 设置与 IN 端点相关的 Tx FIFO 编号
  - 设置希望收到的或发送时要使用到的 data0/data1 PID（仅限批量/中断传输）
  - 设置接收或发送事务时所对应的奇/偶帧（仅限同步传输）
  - 可以设置 NAK 位，从而不论此时 FIFO 的状态如何，都对主机的请求回复 NAK
  - 可以设置 STALL 位，使得主机对该端点的令牌都被硬件回复 STALL
  - 可以将 OUT 端点设置为侦听模式，即对接收到的数据不进行 CRC 检查

端点传输

设备端点 x 传输大小寄存器 (OTG_DIEPTSIZ*x*/OTG_DOEPTSIZ*x*) 允许应用程序对传输大小参数进行编程并读取传输状态。必须在端点控制寄存器中的端点使能位置 1 之前完成对此寄存器的设置。使能端点后，这些字段立即变为只读状态，同时 OTG_FS/OTG_HS 模块根据当前传输状态对这些字段进行更新。

可对以下传输参数进行编程：

- 以字节为单位的传输大小
- 构成整个传输的数据包个数

端点状态/中断

设备端点 x 中断寄存器 (OTG_DIEPINT*x*/OTG_DOPEPINT*x*) 指示端点在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的 OUT 端点中断位或 IN 端点中断位（分别为 OTG_GINTSTS 中的 OEPINT 位或 OTG_GINTSTS 中的 IEPINT 位）置 1 时，应用程序必须读取这些寄存器以获得详细信息。在应用程序读取这些寄存器之前，必须先读取设备全体端点中断 (OTG_DAINT) 寄存器，以获取设备端点 x 中断寄存器的端点编号。应用程序必须将此寄存器中的相应位清零，才能将 OTG_DAINT 和 OTG_GINTSTS 寄存器中的相应位清零

模块提供以下状态检查和中断产生功能：

- 传输完成中断，指示应用程序 (AHB) 和 USB 端均已完成数据传输
- Setup 阶段已完成（仅针对控制传输类型的 OUT 端点）
- 相关的发送 FIFO 为半空或全空状态（IN 端点）
- NAK 应答已发送到主机（仅针对同步传输的 IN 端点）
- Tx FIFO 为空时接收到 IN 令牌（仅针对批量和中断传输类型的 IN 端点）
- 尚未使能端点时接收到 OUT 令牌
- 检测到串扰错误
- 应用程序关闭端点生效
- 应用程序对端点设置 NAK 生效（仅针对同步传输类型的 IN 端点）
- 接收到 3 个以上连续 setup 数据包（仅针对控制类型的 OUT 端点）
- 检测到超时状况（仅针对控制传输类型的 IN 端点）
- 同步传输类型的数据包未产生中断而丢失

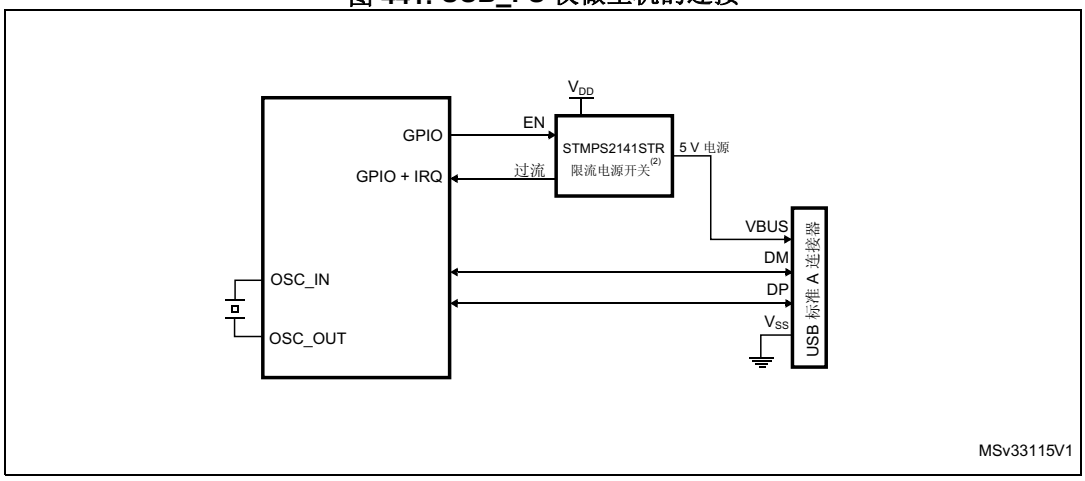
37.7 USB 主机

本节介绍 OTG_FS/OTG_HS 在 USB 主机模式下所具有的功能。在以下情形下，OTG_FS/OTG_HS 用作 USB 主机：

- OTG A 主机
  - OTG A 器件在插入 USB 电缆 A 端时的默认状态
- OTG B 主机
  - OTG B 器件被 HNP 切换为主机角色后的状态
- A 主机
  - 如果 ID 线有连接，器件与 USB 电缆的 A 端相连，全局 USB 配置寄存器中的 HNP 功能位（OTG_GUSBCFG 中的 HNPCAP 位）清零。DP/DM 线上的集成下拉电阻自动使能。
- 仅作主机
  - *37.15.4* 全局 USB 配置寄存器中的强制模式位（OTG_GUSBCFG 中的 FHMOD 位）将强制 OTG_FS/OTG_HS 模块作为 USB 仅作主机运行。这种情况下，即使 USB 连接器上存在 ID 线，模块也会忽略 ID 线上的电平。DP/DM 线上的集成下拉电阻自动使能。

注： *微控制器不能输出 5 V 以提供 V_{BUS}。为此，必须在微控制器以外添加电荷泵或基本电源开关（如果应用电路板提供 5 V 电源）来驱动 5 V V_{BUS} 线。外部电荷泵可通过任何 GPIO 输出驱动。OTG A 主机、A 器件和仅作主机配置都需要使用电荷泵。*

图 441. USB_FS 仅做主机的连接



1. 只有在应用必须支持由 V_{BUS} 供电的器件时才需要 STMPS2141STR。如果应用电路板提供 5 V 电源，则可以使用基本电源开关。
2. V_{DD} 范围介于 2 V 到 3.6 V 之间。

37.7.1 支持 SRP 功能的 USB 主机

全局 USB 配置寄存器中的 SRP 使能位（OTG_GUSBCFG 中的 SRPCAP 位）可提供 SRP 支持。使能 SRP 功能后，主机可在 USB 会话挂起时通过关闭 V_{BUS} 电源来节省电能。

*A 器件会话请求协议*一节详细介绍了 SRP 主机的编程模型。

37.7.2 USB 主机状态

给主机端口供电

微控制器不能输出 5 V 以提供 V_{BUS}。为此，必须在微控制器以外添加电荷泵或电源开关（如果应用电路板提供 5 V 电源）来驱动 5 V V_{BUS} 线。外部电荷泵可通过任何 GPIO 输出驱动。当应用程序确定使用 GPIO 来控制外部器件输出 V_{BUS}，还必须将主机端口控制和状态寄存器中的端口电源位（OTG_HPRT 中的 PPWR 位）置 1。

V_{BUS} 有效

使能 HNP 或 SRP 后，应将 VBUS 感应引脚连接到 V_{BUS}。V_{BUS} 输入可确保电荷泵在 USB 操作期间提供有效的 V_{BUS} 电平。如果 V_{BUS} 电压意外降至 V_{BUS} 有效阈值 (4.25 V) 以下，将通过会话结束检测位（OTG_GOTGINT 中的 SEDET 位）触发 OTG 中断。之后应用必须断开 V_{BUS} 电源并使端口电源位清零。

当 HNP 和 SRP 均禁止时，无需将 VBUS 感应引脚连接到 V_{BUS}，该引脚可用作 GPIO。

电荷泵过流标志也可用来防止电气损坏。将电荷泵的过流标志输出连接到任意 GPIO 输入，然后将其配置为出现有效电平时生成端口中断。过流 ISR 必须立即关闭 V_{BUS} 并清零端口电源位。

主机检测设备连接

如果使能 SRP 或 HNP，即使可以随时连接 USB 外设或 B 器件，OTG_FS/OTG_HS 都不会检测到任何总线连接，直到检测到 V_{BUS} 为有效电平 (5 V)。当 V_{BUS} 处于有效电平且已连接远程 B 器件时，OTG_FS/OTG_HS 模块将发出主机端口中断信号，该中断由主机端口控制和状态寄存器中的设备连接位（OTG_HPRT 中的 PCDET 位）触发。

在 HNP 和 SRP 同时关闭的情况下，USB 设备或 B 器件将在连接后立即被检测到。OTG_FS/OTG_HS 模块将发出主机端口中断信号，该中断由主机端口控制和状态寄存器中的设备连接位（OTG_HPRT 中的 PCDET 位）触发。

主机检测设备断开

设备断开事件将触发断开连接检测中断（OTG_GINTSTS 中的 DISCINT 位）。

主机枚举

检测到设备连接后，若又有新的设备连接进来，主机必须通过向新的设备发送 USB 复位和配置命令来启动枚举过程。

开始驱动 USB 复位前，应用程序必须等待去抖动完成位（OTG_GOTGINT 中的 DBCDNE 位）触发 OTG 中断，这表示由于在 DP (FS) 或 DM (LS) 上连接上拉电阻而发生电气抖动之后，总线恢复稳定状态。

应用程序通过将主机端口控制和状态寄存器中的端口复位位（OTG_HPRT 中的 PRST 位）置 1，并保持最少 10 ms，最多 20 ms，来在 USB 总线上发出 USB 复位信号（单端零）。应用程序计算这个过程的持续时间，然后将端口复位位清零。

USB 复位序列完成后，端口使能/禁止更改位（OTG_HPRT 中的 PENCHNG 位）立即触发主机端口中断， 进而向应用程序发出通知，指示可从主机端口控制和状态寄存器中的端口速度字段（OTG_HPRT 中的 PSPD）读取枚举的设备速度，以及主机已经开始驱动 SOF (FS) 或 Keep-alive 令牌 (LS)。此时主机已就绪，可通过对设备发送命令来完成对设备的枚举。

主机挂起

应用程序通过将主机端口控制和状态寄存器中的端口挂起位（OTG_HPRT 中的 PSUSP）置 1 来挂起 USB 活动。OTG_FS/OTG_HS 模块停止发送 SOF 并进入挂起状态。

可由远程设备的自主活动（远程唤醒）使总线退出挂起状态。这种情况下，远程唤醒信号将触发远程唤醒中断（OTG_GINTSTS 中的 WKUPINT 位），硬件把主机端口控制和状态寄存器中的端口恢复位（OTG_HPRT 中的 PRES 位）自动置位，并通过 USB 自动驱动恢复信号。应用程序必须为恢复窗口定时，然后将端口恢复位清零以退出挂起状态并重新发送 SOF。

如果由主机发起退出挂起状态，则应用程序必须将端口恢复位置 1 以启动主机端口上的恢复信号，为恢复窗口定时并最终将端口恢复位清零。

37.7.3 主机通道

OTG_FS/OTG_HS 内核实现了 12[FS]/16[HS] 个主机通道。每个主机通道均可用于 USB 主机传输（USB 管道）。主机最多能同时处理 12[FS]/16[HS] 个传输请求。如果应用程序有 12[FS]/16[HS] 个以上的传输请求挂起，则在通道从之前任务释放后（即，接收到传输完成和通道停止中断后），主机控制器驱动器 (HCD) 必须为未处理的传输请求重新分配通道。

每个主机通道都可配置为支持输入/输出以及周期性/非周期性事务。每个主机通道都使用专用控制 (OTG_HCCHAR*x*) 寄存器、传输配置 (OTG_HCTSIZ*x*) 寄存器状态/中断 (OTG_HCINT*x*) 寄存器以及和其相关的中断屏蔽寄存器 (OTG_HCINTMSK*x*)。

主机通道控制

- 应用程序可通过主机通道 x 特性寄存器 (OTG_HCCHAR*x*) 对主机通道作以下控制：
  - 通道使能/禁止
  - 设置目标 USB 设备的 HS/FS/LS 速度
  - 设置目标 USB 设备的地址
  - 设置与该通道通信的目标 USB 设备上的端点的编号
  - 设置该通道上的传输方向：IN/OUT
  - 设置该通道上的 USB 传输的类型：控制/批量/中断/同步
  - 设置与该通道通信的设备端点的最大包长
  - 设置要进行周期传输的帧：奇帧/偶帧

主机通道传输

主机通道传输大小寄存器 (OTG_HCTSIZ*x*) 允许应用程序对传输大小参数进行编程并读取传输状态。必须在主机通道特性寄存器中的通道使能位置 1 之前完成对此寄存器的设置。使能端点后，数据包计数字段立即变为只读状态，同时 OTG_FS/OTG_HS 模块根据当前传输状态对该字段进行更新。

- 可对以下传输参数进行编程：
  - 以字节为单位的传输大小
  - 构成整个传输大小的数据包个数
  - 初始数据 PID

主机通道状态/中断

主机通道 *x* 中断寄存器 (OTG_HCINT*x*) 指示通道在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的主机通道中断位（OTG_GINTSTS 中的 HCINT 位）置 1 时，应用程序必须读取这些寄存器以获得详细信息。在读取这些寄存器之前，应用程序必须先读取主机全体通道中断 (OTG_HAINT) 寄存器，以获取主机通道 x 中断寄存器的通道编号。应用程序必须将该寄存器中的相应位清零，才能将 OTG_DAINT 和 OTG_GINTSTS 寄存器中的对应位清零。OTG_HCINTMSKx 寄存器还提供每个通道各中断源的屏蔽位。

- 主机模块提供以下状态检查和中断产生功能：
  - 传输完成中断，指示应用程序 (AHB) 和 USB 端均已完成数据传输
  - 通道因传输完成、USB 事务错误或应用程序发出禁止命令而停止
  - 相关的发送 FIFO 为半空或全空状态（IN 端点）
  - 接收到 ACK 响应
  - 接收到 NAK 响应
  - 接收到 STALL 响应
  - 由于 CRC 校验失败、超时、位填充错误和错误的 EOP 导致 USB 事务错误
  - 串扰错误
  - 帧上溢
  - 数据同步错误

37.7.4 主机调度器

主机模块内置硬件调度器，可自主对应用程序发出的 USB 事务请求重新排序和管理。每一帧开始时，主机都先执行周期性（同步和中断）事务，然后执行非周期性（控制和批量）事务，以符合 USB 规范对同步和中断传输高优先级的保证。

主机通过请求队列（一个周期性请求队列和一个非周期请求队列）处理 USB 事务。每个请求队列最多可存储 8 个条目。每个条目代表一个应用程序发起但还未得到响应的 USB 事务请求，并存储了执行该 USB 事务所用到的 IN 或 OUT 通道的编号，以及其它相关信息。USB 事务请求在队列中的写入顺序决定了事务在 USB 接口上的执行顺序。

每一帧开始时，主机都先处理周期性请求队列，然后处理非周期性请求队列。如果当前帧结束时，计划在当前帧执行的同步或中断类型的 USB 传输事务请求仍处于挂起状态，则主机将发出未完成周期性传输中断（OTG_GINTSTS 中的 IPXFR 位）。OTG_FS/OTG_HS 模块全面负责对周期性和非周期性请求队列的管理。周期性发送 FIFO 和队列状态寄存器 (OTG_HPTXSTS) 与非周期性发送 FIFO 和队列状态寄存器 (OTG_HNPTXSTS) 都为只读寄存器，应用程序可使用它们来读取各请求队列的状态。其中包括：

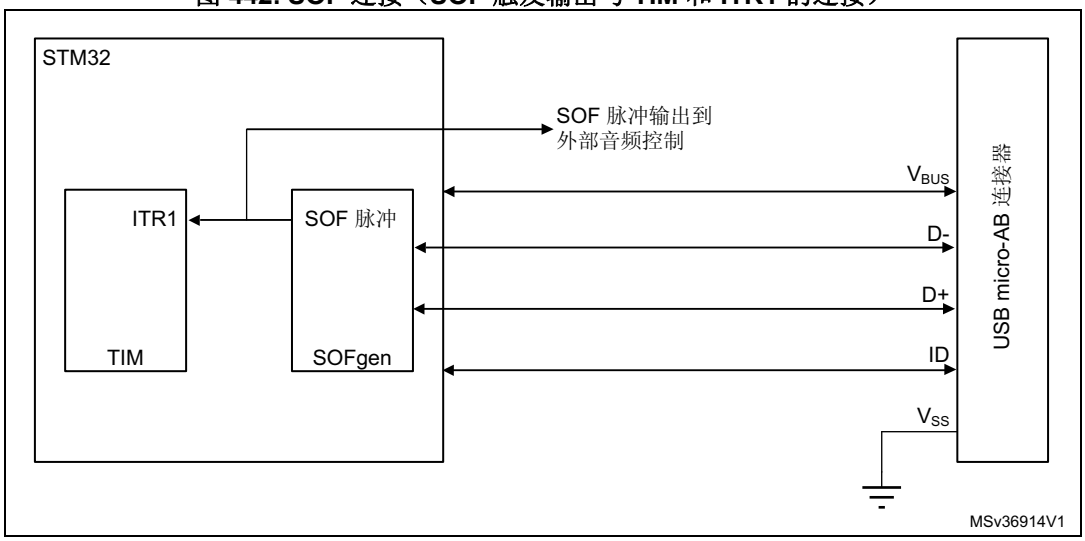
- 周期性（非周期性）请求队列中当前可用的空闲条目数（最多 8 个）
- 周期性（非周期性）Tx FIFO（OUT 事务）中当前可用的空闲空间
- IN/OUT 令牌、主机通道编号和其它状态信息。

由于每个请求队列最多可存储 8 个 USB 事务请求，因此应用程序可以把主机 USB 事务请求提前发送给调度器；实际的通信最晚会在调度器处理完已挂起的 8 个周期事务和 8 个非周期事务完成之后出现在 USB 总线上。

要向主机调度器（队列）发出事务请求，应用程序必须读取 OTG_HNPTXSTS 寄存器中的 PTXQSAV 位或 OTG_HNPTXSTS 寄存器中的 NPTQXSAV 位，确保周期性（非周期性）请求队列中至少有一个可用空间来存储当前请求。

37.8 SOF 触发

图 442. SOF 连接（SOF 触发输出与 TIM 和 ITR1 的连接）



OTG_FS/OTG_HS 模块在主机和设备模式下都可以监视、跟踪和配置 SOF 帧并且还具备 SOF 脉冲输出连接功能。

此功能尤其适用于自适应音频时钟生成，其中音频设备需要与 PC 提供的同步音频数据流实现同步，或者主机需要根据音频设备的要求调整数据帧率。

37.8.1 主机 SOF

主机模式下，可以在主机帧间隔寄存器 (HFIR) 中对所产生的两个连续 SOF (HS/FS) 或 Keep-alive (LS) 令牌期间所出现的 PHY 时钟数进行编程，进而应用程序可对 SOF 帧周期进行控制。帧开始（OTH_GINTSTS 中的 SOF 位）时都将生成中断。当前帧编号和出现下一个 SOF 前剩余的时间应用程序在主机帧编号寄存器 (HFNUM) 中能够进行跟踪。

使用全局控制和配置寄存器中的 SOFOUTEN 位，可以使任何 SOF 令牌发出的同时产生的、宽度为 12 个系统时钟周期的 SOF 脉冲信号从 SOF 引脚输出。此外，SOF 脉冲信号还在芯片内部与定时器的输入触发相连。

37.8.2 设备 SOF

在设备模式下，USB 每次接收到 SOF 令牌时，都将触发帧开始中断（OTH_GINTSTS 中的 SOF 位）。相应的帧编号可从设备状态寄存器（OTG_DSTS 中的 FNSOF 位）读取。使用全局控制和配置寄存器中的 SOF 输出使能位（OTG_GCCFG 中的 SOFOUTEN）位，还可以生成宽度为 12 个系统时钟周期的 SOF 脉冲信号，并使该信号在 SOF 引脚输出，以实现外部可用。此外，SOF 脉冲信号还在内部与 TIM2 的输入触发相连，因此可通过 SOF 脉冲触发输入捕获功能、输出比较功能和定时器。SOF 脉冲与 TIM2 的连接通过 TIM2 选项寄存器 (TIM2_OR) 中的 ITR1_RMP 位使能。

周期性帧结束中断 (OTG_GINTSTS/EOPF) 用于在经过了 80%、85%、90% 或 95% 的帧间隔时间时通知应用程序，具体取决于设备配置寄存器中的周期性帧间隔字段（OTG_DCFG 中的 PFIVL 位）。此功能可用于确定该帧的所有同步通信是否完成。

37.9 电源选项

OTG PHY 的功耗由通用模块配置寄存器中的以下两个位控制：

- PHY 掉电 (OTG_GCCFG/PWRDWN)
  用于开启/关闭 PHY 的全速收发器模块。先置位后才允许后续的 USB 操作。
- V_{BUS} 检测使能 (OTG_GCCFG/VBDEN)
  用于开启/关闭与 OTG 操作关联的 V_{BUS} 感应比较器。

USB 会话没有开始或设备未连接时，可以在 USB 挂起状态下使用功率降低技术。

- 停止 PHY 时钟（OTG_PCGCCTL 中的 STPPCLK 位）
  将时钟门控控制寄存器中的停止 PHY 时钟位置 1 时，OTG 全速模块的大多数 48 MHz 内部时钟域均由时钟门控关闭。即使应用程序仍提供时钟输入，也会节省掉模块由于时钟信号翻转带来的动态功耗。
  还会关掉收发器的大部分单元，只有负责检测异步恢复事件或远程唤醒事件的部分还保持工作状态。
- HCLK 门控（OTG_PCGCCTL 中的 GATEHCLK 位）
  将时钟门控控制寄存器中的 GATEHCLK 位置 1 时，OTG_FS/OTG_HS 模块内部的大多数系统时钟域均由时钟门控关闭。只有寄存器读取和写入接口保持活动状态。即使应用程序仍提供时钟输入，也会节省掉模块由于时钟信号翻转带来的动态功耗。
- USB 系统停止
  当 OTG_FS/OTG_HS 处于 USB 挂起状态时，应用程序可通过将 USB 系统中的所有时钟源全部关闭来显著降低总功耗。USB 系统停止可通过以下方式激活：首先将停止 PHY 时钟位置 1，然后在电源控制系统模块 (PWR) 中将系统配置为深度睡眠模式。
  OTG_FS/OTG_HS 模块通过对 USB 上的远程唤醒（作为主机）或恢复（作为设备）信号进行异步检测，自动重新激活系统时钟和 USB 时钟。

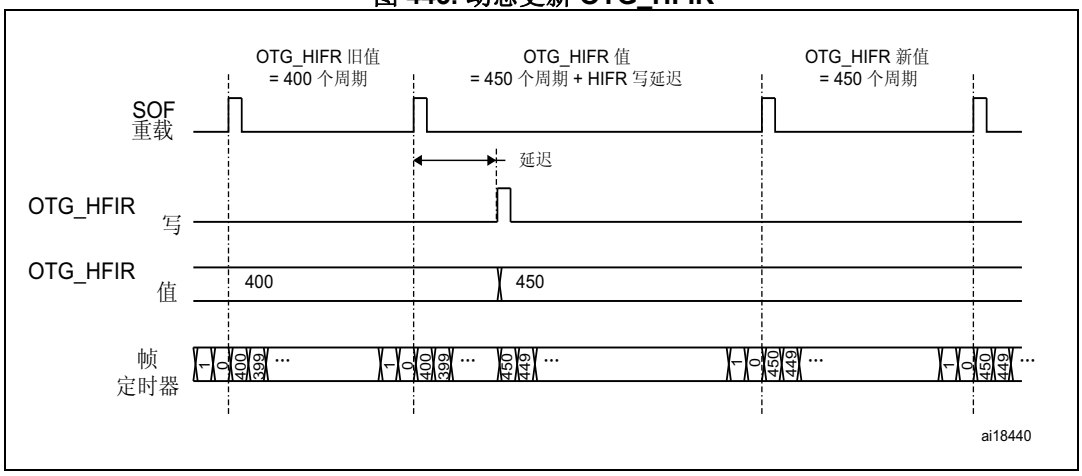
为了节省动态功耗，只在 USB 数据 FIFO 被 OTG_FS/OTG_HS 模块访问时为其提供时钟。

37.10 动态更新 OTG_HFIR 寄存器

主机模式下，USB 模块具有对微帧 [HS]/帧 [FS] 周期进行动态微调的功能，能够将外部设备与微帧 [HS]/帧 [FS] 进行同步。

如果 OTG_HFIR 寄存器在当前微帧 [HS]/帧 [FS] 内发生更改，则将在下一个帧中对 SOF 周期进行相应修正，具体说明请参见*图 443*。

图 443. 动态更新 OTG_HFIR

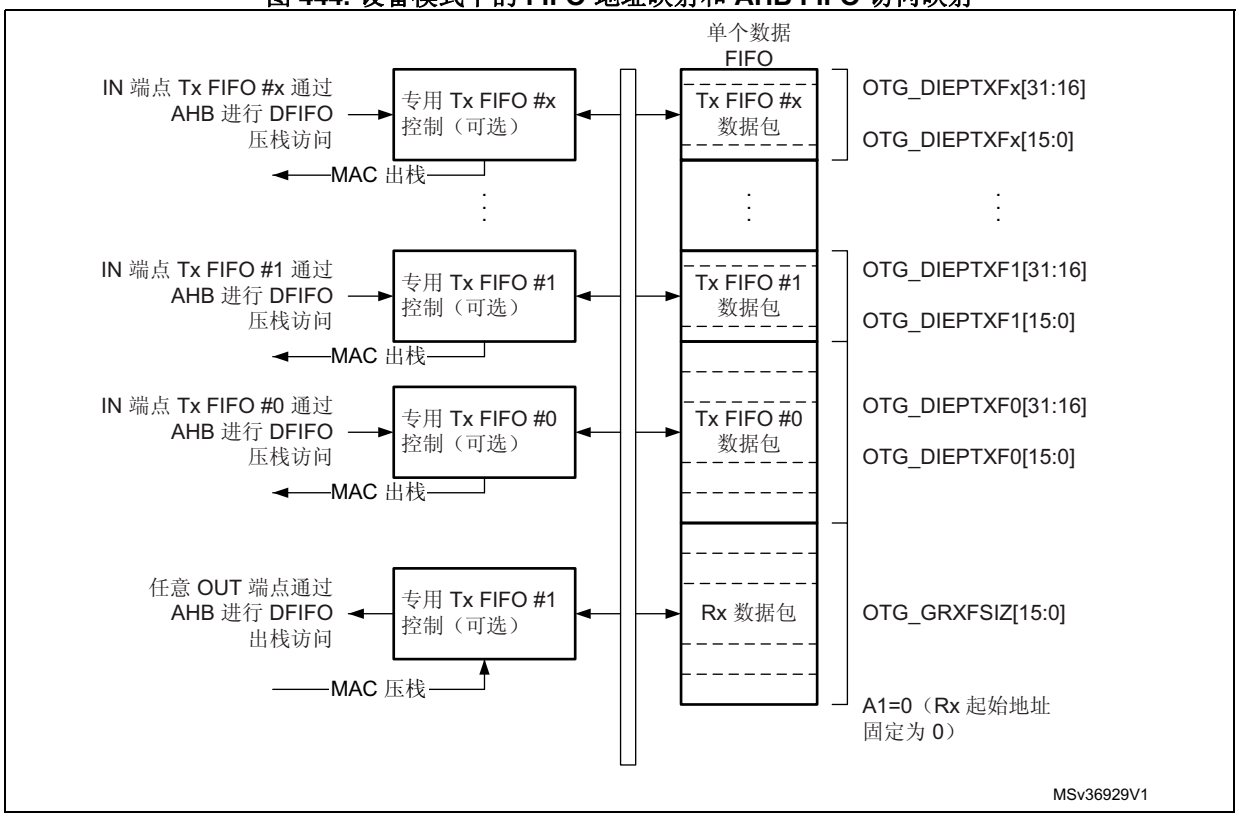


37.11 USB 数据 FIFO

USB 系统具有 1.25[FS]/4[HS] KB 专用 RAM，采用复杂的 FIFO 控制机制。OTG_FS/OTG_HS 模块中的数据包 FIFO 控制器模块将 RAM 空间划分为多个 Tx FIFO（USB 传输前，应用程序将数据压入其中进行短暂存储）和单个 Rx FIFO（从 USB 接收到的数据被应用程序读取之前，在其中进行短暂存储）。RAM 中所构建的 FIFO 的数量与组织方式取决于设备的角色。设备模式下，为每个激活的 IN 端点配置一个 Tx FIFO。FIFO 的大小均由软件配置，以更好地满足应用要求。

37.11.1 设备 FIFO 架构

图 444. 设备模式下的 FIFO 地址映射和 AHB FIFO 访问映射



设备 Rx FIFO

OTG 设备使用单个接收 FIFO 接收发送到所有 OUT 端点的数据。只要 Rx FIFO 中有空余空间，收到的数据包就挨个填入 Rx FIFO。除了有效数据外，接收到的数据包状态（包含 OUT 端点目标编号、字节数、数据 PID 和对所接收数据的验证）也由模块进行存储。没有可用空间时，设备会回复主机事务 NACK 应答并在被寻址的端点上触发中断。接收 FIFO 的大小在接收 FIFO 大小寄存器 (OTG_GRXFSIZ) 中配置。

单个接收 FIFO 架构使得 USB 设备更高效地填充接收 RAM 缓冲区：

- 所有 OUT 端点共享同一个 RAM 缓冲区（共享 FIFO）
- OTG_FS/OTG_HS 模块可将主机发出的任何 OUT 通信序列填充到接收 FIFO，直到没有多余空闲空间

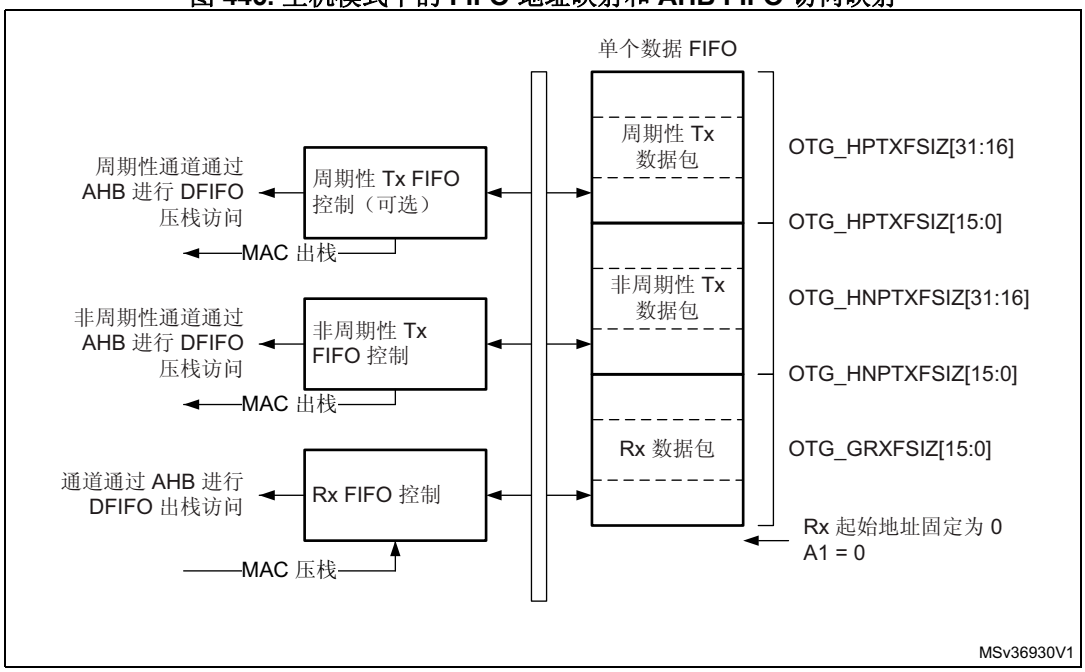
只要至少有一个数据包在 Rx FIFO 中可供读取，应用程序就会一直接收 Rx FIFO 非空中断（OTG_GINTSTS 中的 RXFLVL 位）。应用程序从接收状态读取和出栈寄存器 (OTG_GRXSTSP) 中读取数据包信息，最后通过读取与端点相关的出栈地址从接收 FIFO 读出相应数据。

设备 Tx FIFO

模块为各个 IN 端点提供了专用的 FIFO。应用程序通过端点 0 发送 FIFO 大小寄存器 (OTG_DIEPTXF0) 为 IN 端点 0 配置 FIFO 大小；通过设备 IN 端点发送 FIFOx 寄存器 (OTG_DIEPTXFx) 为 IN 端点 x 配置 FIFO 大小。

37.11.2 主机 FIFO 架构

图 445. 主机模式下的 FIFO 地址映射和 AHB FIFO 访问映射



主机 Rx FIFO

主机使用一个接收 FIFO 处理所有周期和非周期事务。FIFO 用作接收缓冲区以保存从 USB 接收到的数据（接收到的数据包的数据部分），直至这些数据传输到系统存储器。只要 FIFO 中有空间，来自设备 IN 端点的数据包就接收进来并挨个存储。接收到的每个数据包的状态（包含主机目标通道、字节数、数据 PID 和对所接收数据的校验）也存储在 FIFO 中。接收 FIFO 的大小在接收 FIFO 大小寄存器 (OTG_GRXFSIZ) 中配置。

单个接收 FIFO 架构使得 USB 主机高效地填充接收数据缓冲区：

- 所有 IN 配置主机通道共享同一个 RAM 缓冲区（共享 FIFO）
- OTG_FS/OTG_HS 模块可将主机发出的任何 IN 通信序列带来的接收数据填充到接收 FIFO，直到没有多余空闲空间

只要至少有一个数据包在 Rx FIFO 中可供读取，应用程序就会接收 Rx FIFO 非空中断。应用程序从接收状态读取和出栈寄存器中读取数据包信息，最后从接收 FIFO 中读出数据。

主机 Tx FIFO

主机使用一个发送 FIFO 处理所有非周期（控制和批量）OUT 事务，使用另一个发送 FIFO 处理所有周期（同步和中断）OUT 事务。FIFO 用作发送缓冲区以保存要通过 USB 发送的数据（发送数据包）。周期（非周期）Tx FIFO 的大小在主机周期（非周期）发送 FIFO 大小 (OTG_HPTXFSIZ/OTG_HNPTXFSIZ) 寄存器中配置。

两个 Tx FIFO 按优先级实施操作，周期性通信的优先级较高，因此在 USB 一帧的时间内首先进行周期性通信。帧起始时，内置的主机调度器先处理周期请求队列，再处理非周期请求队列。

两个发送 FIFO 的架构使得 USB 主机能够对周期和非周期发送数据缓冲区分别进行优化管理：

- 配置为支持周期（非周期）OUT 事务的所有主机通道共享同一个 RAM 缓冲区（共享 FIFO）
- OTG_FS/OTG_HS 模块可将主机发出的任何 OUT 通信填充到周期性（非周期性）发送 FIFO，直到没有多余空闲空间

只要周期性 Tx FIFO 为半空或全空，OTG_FS/OTG_HS 模块就会发出周期性 Tx FIFO 空中断（OTG_GINTSTS 中的 PTXFE 位），具体取决于 AHB 配置寄存器中的周期性 Tx FIFO 空门限位（OTG_GAHBCFG 中的 PTXFELVL 位）的值。只要周期性 Tx FIFO 和周期性请求队列中均存在空闲空间，应用程序便可提前写入发送数据。可通过读取主机周期性发送 FIFO 和队列状态寄存器 (OTG_HPTXSTS) 来了解二者的可用空间。

只要非周期性 Tx FIFO 为半空或全空，OTG_FS/OTG_HS 模块就会发出非周期性 Tx FIFO 空中断（OTG_GINTSTS 中的 NPTXFE 位），具体取决于 AHB 配置寄存器中的非周期性 Tx FIFO 空门限位（OTG_GAHBCFG 中的 TXFELVL 位）。只要非周期性 Tx FIFO 和非周期性请求队列中均存在空闲空间，应用程序便可写入发送数据。可通过读取主机非周期性发送 FIFO 和队列状态寄存器 (OTG_HNPTXSTS) 来了解二者的可用空间。

37.11.3 FIFO RAM 分配

设备模式

接收 FIFO RAM 分配：应用程序应为 SETUP 数据包分配 RAM：接收 FIFO 中必须保留 10 个位置以在控制端点上接收 SETUP 数据包。OTG 模块不会向这些为 SETUP 数据包保留的位置写入任何其它数据。要为全局 OUT NAK 分配一个位置。状态信息随各个接收数据包写入 FIFO。因此，必须至少为接收数据包分配（最大数据包大小 / 4）+ 1 的空间。如果使能了多个同步端点，则为接收连续数据包分配的空间必须至少为（最大数据包大小 / 4） 的两倍 + 1。通常，推荐的空间为（最大数据包/4 + 1）的两倍，这样当上一个数据包向 CPU 传送时，USB 可同时接收后续的数据包。

传输完成状态信息和该端点收到的最后一个数据包会一起被推入 FIFO。通常情况下，推荐为每个 OUT 端点分配一个位置存储该端点上的传输状态信息。

发送 FIFO RAM 分配：各个 IN 端点发送 FIFO 所需的最小 RAM 空间为该特定 IN 端点的最大数据包大小。

注： *为发送 IN 端点 FIFO 分配的空间越多，USB 的性能就越高。*

主机模式

接收 FIFO RAM 分配：

状态信息随各个接收数据包写入 FIFO。因此，必须至少为接收数据包分配（最大数据包大小 / 4）+ 1 的空间。如果使能了多个同步通道，则为接收连续数据包分配的空间必须至少为（最大数据包大小 / 4）的两倍 + 1。通常，推荐的空间为（最大数据包/4 + 1）的两倍，这样当上一个数据包向 CPU 传送时，USB 可同时接收后续的数据包。

传输完成状态信息和主机通道中的最后一个数据包会一起被推入 FIFO。因此，必须为此分配一个位置。

发送 FIFO RAM 分配：

主机非周期性发送 FIFO 所需的最小 RAM 为所支持的所有非周期性 OUT 通道上传输的最大数据包的大小。

通常，推荐的空间为最大数据包大小的两倍，这样当 USB 正在发送当前数据包的同时，AHB 可以往发送 FIFO 填入下一个数据包。

主机周期性发送 FIFO 所需的最小 RAM 为所支持的所有周期性 OUT 通道上传输的最大数据包的大小。如果至少有一个同步 OUT 端点，则空间必须至少为该通道中最大数据包大小的两倍。

注： *为非周期性发送 FIFO 分配的空间越多，USB 的性能就越高。*

37.12 OTG_FS 系统性能

凭借大容量 RAM 缓冲区、高度可配置的 FIFO 大小、通过 AHB 压栈/出栈寄存器进行 32 位 FIFO 快速访问，尤其是高级 FIFO 控制机制可获得最佳 USB 和系统性能。实际上，无论当前 USB 序列如何，OTG_FS 均可通过该机制高效填充可用的 RAM 空间。借助这些特性：

- 应用程序有足够的裕量来计算并校正CPU的负载，从而优化 CPU 带宽利用率：
  - 应用程序可先积累大量发送数据，再通过 USB 发送出去
  - 可带来足够的时间裕量，以从接收 FIFO 读取数据
- USB 模块能够保持全速工作状态，也就是提供最大的全速带宽（尽量多的硬件自动运行，尽量少的软件参与）：
  - USB 模块可提前积累大量发送数据供其支配，从而可对 USB 数据发送进行自主管理
  - 接收缓冲区中有大量空白空间，可通过 USB 中的数据自动填满

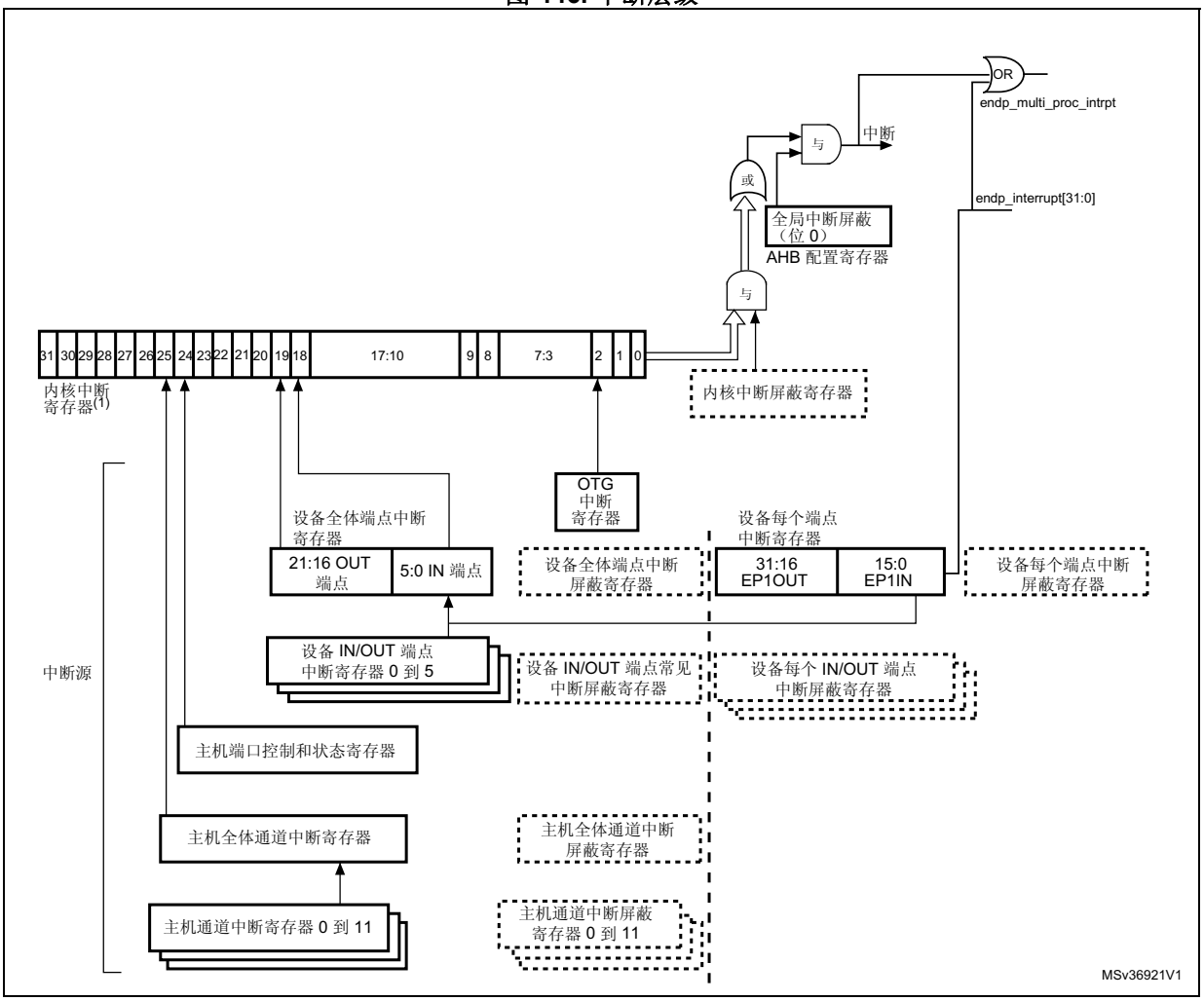
由于 OTG_FS 模块能够高效填充 1.25 KB RAM 缓冲区且 1.25 KB 发送/接收数据足以满足一个全速帧所能容纳的数据量，因此 USB 系统在一帧 (1 ms) 之内可以无需 CPU 干预达到最大 USB 带宽。

37.13 OTG_FS/OTG_HS 中断

当 OTG_FS/OTG_HS 控制器在一种模式下（设备模式或主机模式）工作时，应用程序不得以另一种角色模式访问寄存器。如果发生了非法访问，将会产生模式不匹配中断并在模块中断寄存器（OTG_GINTSTS 寄存器中的 MMIS 位）中反映。当模块从一种角色模式切换到另一种角色模式时，新工作模式下的寄存器必须重新编程为上电复位后的状态。

图 446 显示了中断层级。

图 446. 中断层级



1. 有关模块中断寄存器位，请参见*第 1224 页上的OTG 模块中断寄存器 (OTG_GINTSTS)*。

37.14 OTG_FS/OTG_HS 控制和状态寄存器

应用程序通过 AHB 从接口对控制和状态寄存器 (CSR) 进行读写操作，以此来控制 OTG_FS/OTG_HS 控制器。这些都是 32 位寄存器，其地址按 32 位对齐， 因此只能以字（32 位）的方式访问 OTG_FS/OTG_HS 寄存器。

CSR 分为以下几类：

- 模块全局寄存器
- 主机模式寄存器
- 主机全局寄存器
- 主机端口 CSR
- 主机通道相关寄存器
- 设备模式寄存器
- 设备全局寄存器
- 设备端点相关寄存器
- 电源和时钟门控寄存器
- 数据 FIFO (DFIFO) 访问寄存器

只有模块全局寄存器、电源和时钟门控寄存器、数据 FIFO 访问寄存器以及主机端口控制和状态寄存器，可在主机模式和设备模式下进行访问。当 OTG_FS/OTG_HS 控制器在一种模式下（设备模式或主机模式）工作时，应用程序不得以另一种角色模式访问寄存器。如果发生了非法访问，将会产生模式不匹配中断并在模块中断寄存器（OTG_GINTSTS 寄存器中的 MMIS 位）中反映。当模块从一种角色模式切换到另一种角色模式时，新工作模式下的寄存器必须重新编程为上电复位后的状态。

37.14.1 CSR 存储器映射

主机模式寄存器和设备模式寄存器占据不同的地址。所有寄存器均在 AHB 时钟域内实现。

全局 CSR 地址映射

这些寄存器在主机模式和设备模式下均可用。

表 228. 模块全局控制和状态寄存器 (CSR)

缩略语	偏移地址	寄存器名称
OTG_GOTGCTL	0x000	*第 1214 页上的OTG 控制和状态寄存器 (OTG_GOTGCTL)*
OTG_GOTGINT	0x004	*第 1217 页上的OTG 中断寄存器 (OTG_GOTGINT)*
OTG_GAHBCFG	0x008	*第 1218 页上的OTG AHB 配置寄存器 (OTG_GAHBCFG)*
OTG_GUSBCFG	0x00C	*第 1219 页上的OTG USB 配置寄存器 (OTG_GUSBCFG)*
OTG_GRSTCTL	0x010	*第 1222 页上的OTG 复位寄存器 (OTG_GRSTCTL)*
OTG_GINTSTS	0x014	*第 1224 页上的OTG 模块中断寄存器 (OTG_GINTSTS)*
OTG_GINTMSK	0x018	*第 1229 页上的OTG 中断屏蔽寄存器 (OTG_GINTMSK)*
OTG_GRXSTSR	0x01C	*第 1232 页上的OTG_FS 接收状态调试读取/OTG 状态读取和出栈寄存器 (OTG_GRXSTSR/OTG_GRXSTSP)*
OTG_GRXSTSP	0x020	
OTG_GRXFSIZ	0x024	*第 1234 页上的OTG 接收 FIFO 大小寄存器 (OTG_GRXFSIZ)*
OTG_HNPTXFSIZ/ OTG_DIEPTXF0(1)	0x028	*OTG 主机非周期性发送 FIFO 大小寄存器 (OTG_HNPTXFSIZ)/ 端点 0 发送 FIFO 大小 (OTG_DIEPTXF0)*
OTG_HNPTXSTS	0x02C	*第 1235 页上的OTG 非周期性发送 FIFO/ 队列状态寄存器 (OTG_HNPTXSTS)*

表 228. 模块全局控制和状态寄存器 (CSR)（续）

缩略语	偏移地址	寄存器名称
OTG_GI2CCTL	0x030	*第 1236 页上的OTG I2C 访问寄存器 (OTG_GI2CCTL)*
OTG_GCCFG	0x038	*第 1237 页上的OTG 通用模块配置寄存器 (OTG_GCCFG)*
OTG_CID	0x03C	*第 1238 页上的OTG 模块 ID 寄存器 (OTG_CID)*
OTG_HPTXFSIZ	0x100	*第 1242 页上的OTG 主机周期性发送 FIFO 大小寄存器 (OTG_HPTXFSIZ)*
OTG_DIEPTXFx	0x104 0x124 ... 0x204	*第 1242 页上的OTG 设备 IN 端点发送 FIFO 大小寄存器（OTG_DIEPTXFx）（x = 1..5[FS] / 7[HS]，其中 x 为 FIFO 编号）*，用于 USB_OTG FS
OTG_DIEPTXFx	0x104 0x124 ... 0x244	*第 1242 页上的OTG 设备 IN 端点发送 FIFO 大小寄存器（OTG_DIEPTXFx）（x = 1..5[FS] / 7[HS]，其中 x 为 FIFO 编号）*，用于 USB_OTG HS

1. 通用规则是：主机模式下使用 OTG_HNPTXFSIZ；设备模式下使用 OTG_DIEPTXF0。

主机模式 CSR 地址映射

模块每次切换到主机模式时都必须对这些寄存器进行设置。

表 229. 主机模式控制和状态寄存器 (CSR)

缩略语	偏移地址	寄存器名称
OTG_HCFG	0x400	*第 1243 页上的OTG 主机配置寄存器 (OTG_HCFG)*
OTG_HFIR	0x404	*第 1244 页上的OTG 主机帧间隔寄存器 (OTG_HFIR)*
OTG_HFNUM	0x408	*第 1245 页上的OTG 主机帧编号/帧剩余时间寄存器 (OTG_HFNUM)*
OTG_HPTXSTS	0x410	*第 1245 页上的OTG_Host 周期性发送 FIFO/队列状态寄存器 (OTG_HPTXSTS)*
OTG_HAINT	0x414	*第 1246 页上的OTG 主机全体通道中断寄存器 (OTG_HAINT)*
OTG_HAINTMSK	0x418	*第 1247 页上的OTG 主机全体通道中断屏蔽寄存器 (OTG_HAINTMSK)*
OTG_HPRT	0x440	*第 1247 页上的OTG 主机端口控制和状态寄存器 (OTG_HPRT)*
OTG_HCCHARx	0x500 0x520 ... 0x660	*第 1250 页上的OTG 主机通道 x 特性寄存器 (OTG_HCCHARx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）*，用于 USB_OTG FS
OTG_HCCHARx	0x500 0x520 ... 0x6E0	*第 1250 页上的OTG 主机通道 x 特性寄存器 (OTG_HCCHARx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）*，用于 USB_OTG HS

表 229. 主机模式控制和状态寄存器 (CSR)（续）

缩略语	偏移地址	寄存器名称
OTG_HCSPLTx	0x504 0x524 .... 0x6E4	*第 1251 页上的OTG 主机通道 x 分离控制寄存器 (OTG_HCSPLTx)（x = 0..15，其中 x = 通道编号）*
OTG_HCDMAx	0x514 0x534 .... 0x6F4	*第 1256 页上的OTG 主机通道 x DMA 地址寄存器 (OTG_HCDMAx)（x = 0..15，其中 x = 通道编号）*
OTG_HCINTx	0x508 0x528 .... 0x668	*第 1252 页上的OTG 主机通道 x 中断寄存器 (OTG_HCINTx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）*，用于 USB_OTG FS
OTG_HCINTx	0x508 0x528 .... 0x6E8	*第 1252 页上的OTG 主机通道 x 中断寄存器 (OTG_HCINTx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）*，用于 USB_OTG HS
OTG_HCINTMSKx	0x50C 0x52C 0x66C	*第 1253 页上的OTG 主机通道 x 中断屏蔽寄存器 (OTG_HCINTMSKx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）*，用于 USB_OTG FS
OTG_HCINTMSKx	0x50C 0x52C 0x6EC	*第 1253 页上的OTG 主机通道 x 中断屏蔽寄存器 (OTG_HCINTMSKx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）*，用于 USB_OTG HS
OTG_HCTSIZx	0x510 0x530 .... 0x670	*第 1255 页上的OTG 主机通道 x 传输大小寄存器 (OTG_HCTSIZx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）*，用于 USB_OTG FS
OTG_HCTSIZx	0x510 0x530 .... 0x6F0	*第 1255 页上的OTG 主机通道 x 传输大小寄存器 (OTG_HCTSIZx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）*，用于 USB_OTG HS

设备模式 CSR 地址映射

模块每次切换到设备模式时都必须对这些寄存器进行编程。

表 230. 设备模式控制和状态寄存器

缩略语	偏移地址	寄存器名称
OTG_DCFG	0x800	*第 1256 页上的OTG 设备配置寄存器 (OTG_DCFG)*
OTG_DCTL	0x804	*第 1258 页上的OTG 设备控制寄存器 (OTG_DCTL)*
OTG_DSTS	0x808	*第 1260 页上的OTG 设备状态寄存器 (OTG_DSTS)*
OTG_DIEPMSK	0x810	*第 1261 页上的OTG 设备 IN 端点通用中断屏蔽寄存器 (OTG_DIEPMSK)*
OTG_DOEPMSK	0x814	*第 1262 页上的OTG 设备 OUT 端点通用中断屏蔽寄存器 (OTG_DOEPMSK)*
OTG_DAINT	0x818	*第 1264 页上的OTG 设备全体端点中断寄存器 (OTG_HAINT)*
OTG_DAINTMSK	0x81C	*第 1264 页上的OTG 全体端点中断屏蔽寄存器 (OTG_DAINTMSK)*
OTG_DVBUSDIS	0x828	*第 1265 页上的OTG 设备 V_{BUS} 放电时间寄存器 (OTG_DVBUSDIS)*
OTG_DVBUSPULSE	0x82C	*第 1266 页上的OTG 设备 V_{BUS} 脉冲时间寄存器 (OTG_DVBUSPULSE)*
OTG_DTHRCTL	0x0830	*第 1266 页上的OTG 设备阈值控制寄存器 (OTG_DTHRCTL)*
OTG_DIEPEMPMSK	0x834	*第 1267 页上的OTG 设备 IN 端点 FIFO 空中断屏蔽寄存器 (OTG_DIEPEMPMSK)*
OTG_DEACHINT	0x838	*第 1267 页上的OTG 设备单个端点中断寄存器 (OTG_DEACHINT)*
OTG_DEACHINTMSK	0x83C	*第 1268 页上的OTG 设备单个端点中断屏蔽寄存器 (OTG_DEACHINTMSK)*
OTG_DIEPCTL0	0x900	*第 1269 页上的OTG 设备控制 IN 端点 0 控制寄存器 (OTG_DIEPCTL0)*，用于 USB_OTG FS
OTG_DIEPCTLx	0x920 0x940 ... 0x9A0	*第 1270 页上的OTG 设备端点 x 控制寄存器 (OTG_DIEPCTLx)（x = 1..5[FS] / 0..7[HS]，其中 x = 端点编号）*， 用于USB_OTG FS
OTG_DIEPCTLx	0x900 0x920 ... 0x9E0	*第 1270 页上的OTG 设备端点 x 控制寄存器 (OTG_DIEPCTLx)（x = 1..5[FS] / 0..7[HS]，其中 x = 端点编号）*， 用于 USB_OTG HS
OTG_DIEPINTx	0x908 0x928 .... 0x9A8	*第 1276 页上的OTG 设备端点 x 中断寄存器 (OTG_DIEPINTx)（x = 0..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG FS

表 230. 设备模式控制和状态寄存器（续）

缩略语	偏移地址	寄存器名称
OTG_DIEPINTx	0x908 0x928 ... 0x9E8	*第 1276 页上的OTG 设备端点 x 中断寄存器 (OTG_DIEPINTx)（x = 0..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG HS
OTG_DIEPTSIZ0	0x910	*第 1279 页上的OTG 设备 IN 端点 0 传输大小寄存器 (OTG_DIEPTSIZ0)*
OTG_DTXFSTSx	0x918 0x938 .... 0x9B8	*第 1282 页上的OTG 设备 IN 端点发送 FIFO 状态寄存器 (OTG_DTXFSTSx)（x = 0..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG FS
OTG_DTXFSTSx	0x918 0x938 ..... 0x9F8	*第 1282 页上的OTG 设备 IN 端点发送 FIFO 状态寄存器 (OTG_DTXFSTSx)（x = 0..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG HS
OTG_DIEPTSIZx	0x930 0x950 ... 0x9B0	*第 1281 页上的OTG 设备 IN 端点 x 传输大小寄存器 (OTG_DIEPTSIZx)（x = 1..5[FS] / 7[HS]，其中 x= 端点编号）*，用于 USB_OTG FS
OTG_DIEPTSIZx	0x930 0x950 ... 0x9F0	*第 1281 页上的OTG 设备 IN 端点 x 传输大小寄存器 (OTG_DIEPTSIZx)（x = 1..5[FS] / 7[HS]，其中 x= 端点编号）*，用于 USB_OTG HS
OTG_DOEPCTL0	0xB00	*第 1272 页上的OTG 设备控制 OUT 端点 0 控制寄存器 (OTG_DOEPCTL0)*
OTG_DOEPCTLx	0xB20 0xB40 ... 0xBA0	*第 1274 页上的OTG 设备端点 x 控制寄存器 (OTG_DOEPCTLx)（x = 1..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG FS
OTG_DOEPCTLx	0xB20 0xB40 ... 0xBE0	*第 1274 页上的OTG 设备端点 x 控制寄存器 (OTG_DOEPCTLx)（x = 1..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG HS
OTG_DOEPINTx	0xB08 0xB28 ... 0xBA8	*第 1278 页上的OTG 设备端点 x 中断寄存器 (OTG_DIEPINTx)（x = 0..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG FS
OTG_DOEPINTx	0xB08 0XB28 ... 0xBE8	*第 1278 页上的OTG 设备端点 x 中断寄存器 (OTG_DIEPINTx)（x = 0..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG HS
OTG_DOEPTSIZ0	0xB10	*第 1280 页上的OTG 设备 OUT 端点 0 传输大小寄存器 (OTG_DOEPTSIZ0)*

表 230. 设备模式控制和状态寄存器（续）

缩略语	偏移地址	寄存器名称
OTG_DOEPTSIZx	0xB30 0xB50 ... 0xBB0	*第 1282 页上的OTG 设备 OUT 端点 x 传输大小寄存器 (OTG_DOEPTSIZx)（x = 1..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG FS
OTG_DOEPTSIZx	0xB30 0xB50 .. 0xBF0	*第 1282 页上的OTG 设备 OUT 端点 x 传输大小寄存器 (OTG_DOEPTSIZx)（x = 1..5[FS] / 7[HS]，其中 x = 端点编号）*，用于 USB_OTG HS

数据 FIFO (DFIFO) 访问寄存器地址映射

这些寄存器在主机模式和设备模式下均可用，用于对给定方向的特定端点或通道的 FIFO 空间进行读写操作。如果主机通道类型为 IN，则只能对该通道上的 FIFO 进行读操作。同样地，如果主机通道类型为 OUT，则只能对该通道上的 FIFO 进行写操作。

表 231. 数据 FIFO (DFIFO) 访问寄存器地址映射

FIFO 访问寄存器段	地址范围	访问
设备 IN 端点 0/主机 OUT 通道 0：DFIFO 写访问 设备 OUT 端点 0/主机 IN 通道 0：DFIFO 读访问	0x1000–0x1FFC	w r
设备 IN 端点 1/主机 OUT 通道 1：DFIFO 写访问 设备 OUT 端点 1/主机 IN 通道 1：DFIFO 读访问	0x2000–0x2FFC	w r
...	...	...
设备 IN 端点 x(1)/主机 OUT 通道 x(1)：DFIFO 写访问 设备 OUT 端点 x(1)/主机 IN 通道 x(1)：DFIFO 读访问	0xX000–0xXFFC	w r

1. 其中，设备模式下 x 为 5[FS] / 7[HS]，主机模式下 x 为 11[FS] / 15[HS]。

电源和时钟门控 CSR 地址映射

由一个单独寄存器对电源和时钟门控进行管理。在主机模式和设备模式下均可使用。

表 232. 电源和时钟门控控制和状态寄存器

寄存器名称	缩略语	偏移地址：0xE00–0xFFF
电源和时钟门控控制寄存器	PCGCR	0xE00-0xE04
保留	-	0xE05–0xFFF

37.15 OTG_FS/OTG_HS 寄存器

这些寄存器在主机模式和设备模式下都可用，且在这两个模式间切换时无需对其进行重新编程。

除非特别说明，否则寄存器描述中的位值以二进制表示。

37.15.1 OTG 控制和状态寄存器 (OTG_GOTGCTL)

OTG control and status register

偏移地址：0x000

复位值：0x0001 0000

OTG_GOTGCTL 寄存器控制模块的 OTG 功能并反映其状态。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTG VER	BSVLD	ASVLD	DBCT	CID STS
											rw	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	EHEN	DHNP EN	HSHNP EN	HNP RQ	HNG SCS	Res.	Res.	Res.	Res.	Res.	Res.	SRQ	SRQ SCS
			rw	rw	rw	rw	r							rw	r

注： *USB OTG FS 的配置寄存器*

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.	BSVLD	ASVLD	DBCT	CID STS
												r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	EHEN	DHNP EN	HSHNP EN	HNP RQ	HNG SCS	BVALO VAL	BVALO EN	AVALO VAL	AVALO EN	VBVAL OVAL	VBVAL OEN	SRQ	SRQ SCS
			rw	rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	r

注： *USB OTG HS 的配置寄存器*

位 31:21 保留，必须保持复位值。

位 20 保留，必须保持复位值。

位 20 **OTGVER**：OTG 版本 (OTG version)

选择 OTG 版本。

0：OTG 版本 1.3。在此版本中，模块支持通过数据线脉冲和 Vbus 脉冲实现 SRP。

1：OTG 版本 2.0。在此版本中，模块仅支持通过数据线脉冲实现 SRP。

位 19 **BSVLD**：B 会话有效 (B-session valid)

指示设备模式下收发器的状态。

0：B 会话无效。

1：B 会话有效。

在 OTG 模式下，该位可用于确定设备处于连接状态还是断开状态。

注： *仅可在设备模式下访问。*

位 18 **ASVLD**：A 会话有效 (A-session valid)

指示主机模式下收发器的状态。

0：A 会话无效

1：A 会话有效

注： *仅可在主机模式下访问。*

位 17 **DBCT**：长/短去抖动时间 (Long/short debounce time)

指示已检测到连接的去抖动时间。

0：长去抖动时间，用于物理连接 (100 ms + 2.5 µs)

1：短去抖动时间，用于软连接 (2.5 µs)

注： 仅可在主机模式下访问。

位 16 **CIDSTS**：连接器 ID 状态 (Connector ID status)

指示发生连接事件时的连接器 ID 状态。

0：OTG_FS/OTG_HS 控制器处于 A 器件模式

1：OTG_FS/OTG_HS 控制器处于 B 器件模式

注： 在设备模式和主机模式均可访问。

位 15:13 保留，必须保持复位值。

位 12 **EHEN**：嵌入式主机使能 (Embedded host enable)

用于在 OTG A 器件状态机和嵌入式主机状态机之间选择。

0：选择 OTG A 器件状态机

1：选择嵌入式主机状态机

位 11 **DHNPEN**：使能设备 HNP 特性 (Device HNP enabled)

从所连 USB 主机处成功接收到 SetFeature.SetHNPEnable 命令时，应用程序将该位置 1。

0：应用不使能 HNP

1：应用使能 HNP

注： 仅可在设备模式下访问。

位 10 **HSHNPEN**：主机设置 HNP 使能 (Host set HNP enable)

（使用 SetFeature.SetHNPEnable 命令）在所连接设备上成功使能 HNP 后，应用程序将该位置 1。

0：主机未对设备设置了 HNP 使能

1：主机已对设备设置了 HNP 使能

注： 仅可在主机模式下访问。

位 9 **HNPRQ**：HNP 请求 (HNP request)

应用程序将该位置 1 时，将对所连接 USB 主机发起 HNP 请求。当 OTG_GOTGINT 寄存器中的主机协商成功状态更改位（OTG_GOTGINT 中的 HNSSCHG 位）置 1 时，应用程序可通过写 0 将该位清零。HNSSCHG 位清零时，模块会将该位清零。

0：无 HNP 请求

1：HNP 请求

注： 仅可在设备模式下访问。

位 8 **HNGSCS**：主机协商成功 (Host negotiation success)

当主机协商成功时，模块会将该位置 1。当该寄存器中的 HNP 请求位 (HNPRQ) 置 1 时，模块会将该位清零。

0：主机协商失败

1：主机协商成功

注： 仅可在设备模式下访问。

位 7 **BVALOVAL**：B 器件会话有效覆盖值 (B-peripheral session valid override value)

此位用于在 BVALOEN 位置 1 时设置 Bvalid 信号的覆盖值。

0：BVALOEN = 1 时，Bvalid 值为“0”

1：BVALOEN = 1 时，Bvalid 值为“1”

注： 仅可在设备模式下访问。

位 6 **BVALOEN**：B 器件会话有效覆盖使能 (B-peripheral session valid override enable)

该位用于使能/禁止软件通过 BVALOVAL 位来覆盖 Bvalid 信号。

0：禁止覆盖，模块使用来自所选 PHY 的 Bvalid 信号

1：从 PHY 内部接收的 Bvalid 由 BVALOVAL 位的值覆盖

注： 仅可在设备模式下访问。

位 5 **AVALOVAL**：A 器件会话有效覆盖值 (A-peripheral session valid override value)

此位用于在 AVALOEN 位置 1 时设置 Avalid 信号的覆盖值。

0：AVALOEN = 1 时，Avalid 值为“0”

1：AVALOEN = 1 时，Avalid 值为“1”

注： 仅可在主机模式下访问。

位 4 **AVALOEN**：A 器件会话有效覆盖使能 (A-peripheral session valid override enable)

该位用于使能/禁止软件通过 AVALOVAL 位来覆盖 Avalid 信号。

0：禁止覆盖，模块使用来自所选 PHY 的 Avalid 信号

1：从 PHY 内部接收的 Avalid 由 AVALOVAL 位的值覆盖

注： 仅可在主机模式下访问。

位 3 **VBVALOVAL**：V_{BUS} 有效覆盖值 (V_{BUS} valid override value)

此位用于在 VBVALOEN 位置 1 时设置 vbusvalid 信号的覆盖值。

0：VBVALOEN = 1 时，vbusvalid 值为“0”

1：VBVALOEN = 1 时，vbusvalid 值为“1”

注： 仅可在主机模式下访问。

位 2 **VBVALOEN**：V_{BUS} 有效覆盖使能 (V_{BUS} valid override enable)

该位用于使能/禁止软件通过 VBVALOVAL 位来覆盖 vbusvalid 信号。

0：禁止覆盖，模块使用来自所选 PHY 的 vbusvalid 信号

1：从 PHY 内部接收的 vbusvalid 由 VBVALOVAL 位的值覆盖

注： 仅可在主机模式下访问。

位 2:7 保留，必须保持复位值。

位 1 **SRQ**：会话请求 (Session request)

应用程序将该位置 1 时，将在 USB 上发起会话请求。当 OTG_GOTGINT 寄存器中的主机协商成功状态更改位（OTG_GOTGINT 中的 HNSSCHG 位）置 1 时，应用程序可通过写 0 将该位清零。HNSSCHG 位清零时，模块会将该位清零。

如要使用 USB 1.1 全速串行收发器接口来启动会话请求，则应用程序必须在该寄存器中的 B 会话有效位（OTG_GOTGCTL 中的 BSVLD 位）清零后，等待 V_{BUS} 放电至 0.2 V。该放电时间因 PHY 不同而异，可从 PHY 供应商处了解相关信息。

0：无会话请求

1：会话请求

注： 仅可在设备模式下访问。

位 0 **SRQSCS**：会话请求成功 (Session request success)

当会话请求成功时，模块会将该位置 1。

0：会话请求失败

1：会话请求成功

注： 仅可在设备模式下访问。

37.15.2 OTG 中断寄存器 (OTG_GOTGINT)

OTG interrupt register

偏移地址：0x04

复位值：0x0000 0000

应用程序会在出现 OTG 中断时读取该寄存器，并通过将该寄存器中的位清零来清除 OTG 中断。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDCHNG	DBCDNE	ADTOCHG	HNGDET	Res.
											rc_w1	rc_w1	rc_w1	rc_w1	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	HNSSCHG	SRSSCHG	Res.	Res.	Res.	Res.	Res.	SEDET	Res.	Res.
						rc_w1	rc_w1						rc_w1		

位 31:21 保留，必须保持复位值。

位 20 **IDCHNG：**
该位置 1 时，表示 ID 输入引脚的值发生了变化。

位 19 **DBCDNE：**去抖动完成 (Debounce done)
模块会在设备连接后且去抖动结束时将该位置 1。应用程序会在看见该中断后，开始驱动 USB 复位。该位仅在 OTG_GUSBCFG 寄存器中的 HNP 功能位或 SRP 功能位（分别为 OTG_GUSBCFG 中的 HNPCAP 位或 SRPCAP 位）置 1 时有效。
注： 仅可在主机模式下访问。

位 18 **ADTOCHG：**A 器件超时更改 (A-device timeout change)
模块将该位置 1 时，指示 A 器件在等待 B 器件连接时超时。
注： 在设备模式和主机模式均可访问。

位 17 **HNGDET：**检测到主机协商 (Host negotiation detected)
当检测到 USB 上的主机协商请求时，模块会将该位置 1。
注： 在设备模式和主机模式均可访问。

位 16:10 保留，必须保持复位值。

位 9 **HNSSCHG：**主机协商成功状态更改 (Host negotiation success status change)
模块将在 USB 主机协商请求成功或失败时将此位置 1。应用程序必须读取 OTG_GOTGCTL 寄存器中的主机协商成功位（OTG_GOTGCTL 中的 HNGSCS 位）来检查请求成功还是失败。
注： 在设备模式和主机模式均可访问。

位 7:3 保留，必须保持复位值。

位 8 **SRSSCHG**：会话请求成功状态更改 (Session request success status change)

模块将在会话请求成功或失败时将此位置 1。应用程序必须读取 OTG_GOTGCTL 寄存器中的会话请求成功位（OTG_GOTGCTL 中的 SRQSCS 位）来检查请求成功还是失败。

注： 在设备模式和主机模式均可访问。

位 2 **SEDET**：检测到会话结束 (Session end detected)

模块将该位置 1 时，指示 V_{BUS} < 0.8 V 时，V_{BUS} 上的电压不再适用于 B 器件会话。

注： 在设备模式和主机模式均可访问。

位 1:0 保留，必须保持复位值。

37.15.3 OTG AHB 配置寄存器 (OTG_GAHBCFG)

OTG AHB configuration register

偏移地址：0x008

复位值：0x0000 0000

该寄存器可用于在上电后或更改角色模式时对模块进行配置。该寄存器主要包含 AHB 系统相关的配置参数。请勿在初始编程后更改该寄存器。应用程序必须在开始任何 AHB 或 USB 事务前对该寄存器进行编程。

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

注： USB OTG FS 的配置寄存器

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.	PTXFE LVL	TXFE LVL	Res.	DMAEN	HBSTLEN				GINT MSK
							rw	rw		rw	rw	rw	rw	rw	rw

注： USB OTG HS 的配置寄存器

位 31:20 保留，必须保持复位值。

位 8 **PTXFELVL**：周期性 Tx FIFO 空门限 (Periodic Tx FIFO empty level)

指示 OTG_GINTSTS 寄存器中的周期性 Tx FIFO 空中断位（OTG_GINTSTS 中的 PTXFE 位）何时触发。

0：PTXFE（位于 OTG_GINTSTS）中断表示周期性 Tx FIFO 为半空状态

1：PTXFE（位于 OTG_GINTSTS）中断表示周期性 Tx FIFO 为全空状态

注： 仅可在主机模式下访问。

位 7 **TXFELVL**：Tx FIFO 空门限 (Tx FIFO empty level)

在设备模式下，该位指示 IN 端点发送 FIFO 空中断（OTG_DIEPINTx 中的 TXFE）何时触发：

0：TXFE（位于 OTG_DIEPINTx）中断表示 IN 端点 Tx FIFO 为半空状态

1：TXFE（位于 OTG_DIEPINTx）中断表示 IN 端点 Tx FIFO 为全空状态

在主机模式下，该位指示非周期性 Tx FIFO 空中断（OTG_GINTSTS 中的 NPTXFE 位）何时触发。

0：NPTXFE（位于 OTG_GINTSTS）中断表示非周期性 Tx FIFO 为半空状态

1：NPTXFE（位于 OTG_GINTSTS）中断表示非周期性 Tx FIFO 为全空状态

位 6:1 保留，对于 USB OTG FS，必须保持复位值。

位 6 保留，对于 USB OTG HS，必须保持复位值。

位 5 **DMAEN**：为 USB OTG HS 使能 DMA (DMA enabled for USB OTG HS)

0：模块不使用 DMA 进行数据收发

1：模块使用 DMA 进行数据收发

位 4:1 **HBSTLEN**：USB OTG HS 的突发长度/类型 (Burst length/type for USB OTG HS)

0000 单次
0001 INCR
0011 INCR4
0101 INCR8
0111 INCR16
其它值：保留

位 0 **GINTMSK**：全局中断屏蔽 (Global interrupt mask)

该位用于屏蔽全局中断或使能全局中断。中断状态寄存器由模块进行更新，与此位的设置无关。

0：对应用程序屏蔽中断。

1：不对应用程序屏蔽中断。

注： 在设备模式和主机模式均可访问。

37.15.4 OTG USB 配置寄存器 (OTG_GUSBCFG)

OTG USB configuration register

偏移地址：0x00C

复位值：0x0000 1440

该寄存器可用于在上电或更改角色模式后对模块进行配置。其中包含与 USB 和 USB-PHY 相关的配置参数。应用程序必须在开始任何 AHB 或 USB 事务前对该寄存器进行编程。请勿在初始编程后更改该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	FD MOD	FH MOD	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.	TRDT				HNP CAP	SRP CAP	Res.	PHY SEL	Res.	Res.	Res.	TOCAL		
		rw				rw	rw		r				rw		

注：　USB OTG FS 的配置寄存器

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	FD MOD	FH MOD	Res.	Res.	Res.	ULPIIP D	PTCI	PCCI	TSDPS	ULPIE VBUSI	ULPIE VBUSD	ULPIC SM	ULPIA R.	ULPIFS L.	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
PHYL PC.	Res.	TRDT				HNP CAP	SRP CAP	Res.	PHY SEL	Res.	Res.	Res.	TOCAL		
rw		rw				rw	rw		r				rw		

注：　USB OTG HS 的配置寄存器

位 31　保留，必须保持复位值。

位 30　**FDMOD**：强制设备模式 (Force device mode)

向该位写入 1 时，可将模块强制为设备模式，而无需考虑 OTG_ID 输入引脚。

0：正常模式

1：强制设备模式

将强制位置 1 后，应用程序必须等待至少 25 ms 后更改方可生效。

注：　在设备模式和主机模式均可访问。

位 29　**FHMOD**：强制主机模式 (Force host mode)

向该位写入 1 时，可将模块强制为主机模式，而无需考虑 OTG_ID 输入引脚。

0：正常模式

1：强制主机模式

将强制位置 1 后，应用程序必须等待至少 25 ms 后更改方可生效。

注：　在设备模式和主机模式均可访问。

位 28:24　保留，必须保持复位值。

位 25:15　保留，对于 USB OTG FS，必须保持复位值。

位 25　**ULPIIPD**：USB OTG HS 的 ULPI 接口保护禁止 (ULPI interface protect disable for USB OTG HS)

该位控制 PHY 中内置的电路，以保护 ULPI 接口。该功能所采用的任何上拉或下拉电阻均可禁止。有关详细信息，请参见 ULPI 规范。

0：使能接口保护电路

1：禁止接口保护电路

位 24　**PTCI**：USB OTG HS 的通过指示 (Indicator pass through for USB OTG HS)

该位将控制互补输出是否需要通过内部 V_{BUS} 有效电平比较器的验证，之后方可用于 RX CMD 中的 V_{BUS} 状态。有关详细信息，请参见 ULPI 规范。

0：互补输出信号通过内部 V_{BUS} 有效电平比较器的验证

1：互补输出信号不通过内部 V_{BUS} 有效电平比较器的验证

位 23 **PCCI**：USB OTG HS 的互补指示 (Indicator complement for USB OTG HS)

该位控制 PHY 将 ExternalVbusIndicator 输入信号反转并生成互补输出。有关详细信息，请参见 ULPI 规范。

0：PHY 不将 ExternalVbusIndicator 信号反转

1：PHY 将 ExternalVbusIndicator 信号反转

位 22 **TSDPS**：USB OTG HS 的 TermSel DLine 脉冲选择 (TermSel DLine pulsing selection for USB OTG HS)

该位选择 utmi_termselect 来驱动 SRP（会话请求协议）期间的数据线脉冲。

0：使用 utmi_txvalid 驱动数据线脉冲（默认）

1：使用 utmi_termsel 驱动数据线脉冲

位 21 **ULPIEVBUSI**：USB OTG HS 的 ULPI 外部 V_{BUS} 指示 (ULPI external V_{BUS} indicator for USB OTG HS)

该位指示 ULPI PHY 使用外部 V_{BUS} 过流指示。

0：PHY 使用内部 V_{BUS} 有效比较器

1：PHY 使用外部 V_{BUS} 有效比较器

位 20 **ULPIEVBUSD**：USB OTG HS 的 ULPI 外部 V_{BUS} 驱动器 (ULPI External V_{BUS}Drive for USB OTG HS)

该位选择 ULPI PHY 使用内部或外部电源来驱动 V_{BUS} 上的 5 V 电压。

0：PHY 使用内部电荷泵驱动 V_{BUS}（默认）

1：PHY 使用外部电源驱动 V_{BUS}。

位 19 **ULPICSM**：USB OTG HS 的 ULPI 时钟 SuspendM (ULPI Clock SuspendM for USB OTG HS)

该位置位 ULPI PHY 接口控制寄存器中的 ClockSuspendM 位。该位仅适用于串行和 carkit 模式。

0：PHY 在挂起期间断开内部时钟的电源

1：PHY 不断开内部时钟的电源

位 18 **ULPIAR**：USB OTG HS 的 ULPI 自动恢复 (ULPI Auto-resume for USB OTG HS)

该位置位 ULPI PHY 接口控制寄存器中的 AutoResume 位。

0：PHY 不使用自动恢复功能

1：PHY 使用自动恢复功能

位 17 **ULPIFSLS**：USB OTG HS 的 ULPI FS/LS 选择 (ULPI FS/LS select for USB OTG HS)

应用程序使用该位为 ULPI PHY 选择 FS/LS 串行接口。该位仅在 ULPI PHY 上选择了 FS 串行收发器的情况下有效。

0：ULPI 接口

1：ULPI FS/LS 串行接口

位 16 保留，对于 USB OTG HS，必须保持复位值。

位 15 **PHYLPCS**：USB OTG HS 的 PHY 低功耗时钟选择 (PHY Low-power clock select for USB OTG HS)

该位用于选择 480 MHz 或 48 MHz（低功耗）PHY 模式。在 FS 和 LS 模式下，PHY 通常可使用 48 MHz 时钟运行，从而节约功耗。

0：480 MHz 内部 PLL 时钟

1：48 MHz 外部时钟

在 480 MHz 模式下，UTMI 接口以 60 MHz 或 30MHz 运行，具体取决于选择的是 8 位还是 16 位数据宽度。在 48 MHz 模式下，UTMI 接口在 FS 和 LS 模式下以 48 MHz 运行。

位 14 保留，必须保持复位值。

位 13:10 **TRDT**：USB 周转时间 (USB turnaround time)

以 PHY 时钟数为单位设置周转时间。

要计算 TRDT 的值，请使用如下公式：

TRDT = 4 × AHB 时钟 + 1 个 PHY 时钟

例如：

如果 AHB 时钟频率 = 72 MHz（PHY 时钟频率 = 48 MHz），则 TRDT 设置为 9。

如果 AHB 时钟频率 = 48 MHz（PHY 时钟频率 = 48 MHz），则 TRDT 设置为 5。

注： 仅可在设备模式下访问。

位 9 **HNPCAP**：HNP 功能 (HNP-capable)

应用程序使用该位控制 OTG_FS/OTG_HS 控制器的 HNP 功能。

0：不使能 HNP 功能。

1：使能 HNP 功能。

注： 在设备模式和主机模式均可访问。

位 8 **SRPCAP**：SRP 功能 (SRP-capable)

应用程序使用该位控制 OTG_FS/OTG_HS 控制器的 SRP 功能。如果模块作为无 SRP 功能的 B 器件，则无法请求连接的 A 器件（主机）激活 V_{BUS} 并开始会话。

0：不使能 SRP 功能。

1：使能 SRP 功能。

注： 在设备模式和主机模式均可访问。

位 7 保留，必须保持复位值。

位 6 **PHYSEL**：全速串行收发器选择 (Full Speed serial transceiver select)

此位为只读位且始终为 1。

位 5:3 保留，必须保持复位值。

位 2:0 **TOCAL**：FS 超时校准 (FS timeout calibration)

PHY 引入的额外延迟包括应用程序在该字段中设置的 PHY 时钟数，以及模块的全速数据包间超时间隔。不同 PHY 引入的延迟对数据线状态的影响是不同的。

全速操作的 USB 标准超时值为 16 到 18（含）个位时间。应用程序必须根据枚举速度编程该字段。每个 PHY 时钟增加的位时间数为 0.25 个位时间。

37.15.5 OTG 复位寄存器 (OTG_GRSTCTL)

OTG reset register

偏移地址：0x10

复位值为：0x8000 0000

应用程序通过此寄存器复位模块中的各项硬件特性。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AHB IDL	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.	TXFNUM					TXF FLSH	RXF FLSH	Res.	FCRST	PSRST	CSRST
					rw					rs	rs		rs	rs	rs

注： USB OTG FS 的配置寄存器

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AHB IDL	DMAR EQ	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.	TXFNUM					TXF FLSH	RXF FLSH	Res.	Res.	PSRST	CSRST
					rw					rs	rs		rs	rs	rs

注： *USB OTG HS 的配置寄存器*

位 31 **AHBIDL**：AHB 主器件空闲 (AHB master idle)

指示 AHB 主器件状态机处于空闲情况。

注： *在设备模式和主机模式均可访问。*

位 30:11 保留，对于 USB OTG FS，必须保持复位值

位 30 **DMAREQ**：为 USB OTG HS 使能 DMA 请求信号 (DMA request signal enabled for USB OTG HS)

该位指示 DMA 请求正在进行中。用于调试。

位 29:11 保留，对于 USB OTG HS，必须保持复位值

位 10:6 **TXFNUM**：Tx FIFO 编号 (Tx FIFO number)

使用 Tx FIFO 刷新位进行刷新的 FIFO 编号。只有在模块将 Tx FIFO 刷新位清零后，方可更改此字段。

00000：

– 主机模式下刷新非周期性 Tx FIFO

– 设备模式下刷新 Tx FIFO 0

00001：

– 主机模式下刷新周期性 Tx FIFO

– 设备模式下刷新 Tx FIFO 1

00010：设备模式下刷新 Tx FIFO 2

...

01111：设备模式下刷新 Tx FIFO 15

10000：在设备模式或主机模式下刷新所有的发送 FIFO。

注： *在设备模式和主机模式均可访问。*

位 5 **TXFFLSH**：Tx FIFO 刷新 (Tx FIFO flush)

此位选择性地刷新一个或所有的发送 FIFO，但当模块处理通信事务时无法执行该操作。

只有在确认模块当前未对 Tx FIFO 执行读写操作后，应用程序方可对此位执行写操作。使用以下寄存器进行确认：

读 — NAK 有效中断可确保模块当前未对 FIFO 执行读操作

写 — OTG_GRSTCTL 中的 AHBIDL 位可确保模块当前未对 FIFO 执行任何写操作。

通常建议在重新配置 FIFO 时进行刷新。还建议在设备端点禁止期间进行 FIFO 刷新。应用程序必须等待模块将此位清零，才能执行任意操作。该位需要八个时钟来清零（使用较慢的 phy_clk 或 hclk 时钟）。

注： *在设备模式和主机模式均可访问。*

位 4 **RXFFLSH**：Rx FIFO 刷新 (Rx FIFO flush)

应用程序可使用此位刷新整个 Rx FIFO，但必须首先确保模块当前未在处理事务。

只有在确认模块当前未对 Rx FIFO 执行读写操作后，应用程序方可对此位执行写操作。

应用程序必须等到此位清零后，方可执行其它操作。通常需要等待 8 个时钟周期（以 PHY 或 AHB 时钟中最慢的为准）。

注： *在设备模式和主机模式均可访问。*

位 3 保留，必须保持复位值。

位 2 **FCRST**：主机帧计数器复位 (Host frame counter reset)

应用程序对该位执行写操作时，模块中的帧数计数器复位。帧计数器复位后，由模块发送的下一个 SOF 的帧号为 0。

当应用程序向此位写入“1”后，可能无法回读该值，原因是模块将在几个时钟周期内将此位清零。

注： 仅可在主机模式下访问。

位 1 **PSRST**：部分软复位 (Partial soft reset)

复位内部状态机，但保留枚举信息。可用于恢复一些特定的 PHY 错误。

注： 在设备模式和主机模式均可访问。

位 0 **CSRST**：模块软复位 (Core soft reset)

按如下所述将 HCLK 和 PHY 时钟域复位：

除以下各位外，将各个中断和所有 CSR 寄存器位清零：

– OTG_PCGCCTL 中的 GAYEHCLK 位
– OTG_PCGCCTL 中的 STPPCLK 位
– OTG_HCFG 中的 FSLSPCS 位
– OTG_DCFG 中的 DSPD 位
– OTG_DCTL 中的 SDIS 位
– OTG_GCCFG 寄存器
– OTG_GPWRDN 寄存器

将所有模块状态机（AHB 从器件除外）复位至空闲状态，并清空所有发送 FIFO 和接收 FIFO。

在 AHB 传输的最后数据阶段结束后，尽快终止 AHB 主器件上的所有事务。立即终止 USB 上的所有事务。

应用程序可在需要复位模块时随时对该位执行写操作。该位为自清零位，模块将在其中所有必要逻辑复位后将该位清零，该过程需要若干个时钟的时间，具体取决于模块的当前状态。该位一旦清零，软件必须等待至少 3 个 PHY 时钟后才可以访问 PHY 域（同步延迟）。此外，软件还必须在确定该寄存器中的位 31 置 1（AHB 主器件空闲）后方可开始运行。

软件复位通常在两种情况下使用，一是软件开发期间，二是用户动态更改以上所列 USB 配置寄存器中的 PHY 选择位后。用户更改 PHY 时，将为 PHY 选择相应的时钟并用于 PHY 域中。一旦选择了新的时钟，则必须复位 PHY 域，才能保证正常运行。

注： 在设备模式和主机模式均可访问。

37.15.6 OTG 模块中断寄存器 (OTG_GINTSTS)

OTG core interrupt register

偏移地址：0x014

复位值：0x1400 0020

该寄存器用于在当前模式（设备模式或主机模式）下借助系统级别的事件来中断应用程序。

该寄存器中的某些位仅在主机模式下有效，而其它位则仅在设备模式下有效。此外，该寄存器还可指示当前模式。要将 rc_w1 类型中断状态位清零，应用程序必须向该位写入 1。

FIFO 状态中断为只读；如果软件在处理这些中断期间对 FIFO 执行读写操作，则 FIFO 中断标志将自动清零。

使能中断位前，应用程序必须在初始化时将 OTG_GINTSTS 寄存器清零，才可以避免在初始化前产生任何中断。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WKUP INT	SRQ INT	DISC INT	CIDS CHG	LPM INT	PTXFE	HCINT	HPRT INT	RST DET	Res.	IPXFR/ 输入 COMP ISO 输出	IISOI XFR	OEP INT	IEPINT	Res.	Res.
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	r	r	r	rc_w1		rc_w1	rc_w1	r	r		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EOPF	ISOO DRP	ENUM DNE	USB RST	USB SUSP	ESUSP	Res.	Res.	GO NAK EFF	GI NAK EFF	NPTXF E	RXF LVL	SOF	OTG INT	MMIS	CMOD
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1			r	r	r	r	rc_w1	r	rc_w1	r

注： *USB OTG FS 的配置寄存器*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WKUP INT	SRQ INT	DISC INT	CIDS CHG	Res.	PTXFE	HCINT	HPRT INT	Res.	DATAF SUSP	IPXFR/ 输入 COMP ISO 输出	IISOI XFR	OEP INT	IEPINT	Res.	Res.
rc_w1	rc_w1	rc_w1	rc_w1		r	r	r		rc_w1	rc_w1	rc_w1	r	r		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EOPF	ISOO DRP	ENUM DNE	USB RST	USB SUSP	ESUSP	Res.	Res.	GO NAK EFF	GI NAK EFF	NPTXF E	RXF LVL	SOF	OTG INT	MMIS	CMOD
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1			r	r	r	r	rc_w1	r	rc_w1	r

注： *USB OTG HS 的配置寄存器*

位 31 **WKUPINT**：检测到恢复/远程唤醒中断 (Resume/remote wakeup detected interrupt)

挂起 (L2) 或 LPM(L1) 状态期间的唤醒中断。

– 挂起 (L2) 期间：

在设备模式下，当 USB 总线上检测到恢复信号时，将触发该中断。在主机模式下，当 USB 上检测到远程唤醒时，将触发该中断。

– LPM(L1) 期间：

USB 上检测到主机发起的恢复信号或设备发起的远程唤醒时，将触发该中断。

注： *在设备模式和主机模式均可访问。*

位 30 **SRQINT**：检测到会话请求/新会话中断 (Session request/new session detected interrupt)

在主机模式下，当检测到来自设备的会话请求时，将触发该中断。在设备模式下，当 V_{BUS} 在 B 外设设备的有效范围内时，将触发该中断。在设备模式和主机模式均可访问。

位 29 **DISCINT**：检测到断开连接中断 (Disconnect detected interrupt)

当检测到设备断开连接时触发该中断。

注： *仅可在主机模式下访问。*

位 28 **CIDSCHG**：连接器 ID 线状态更改 (Connector ID status change)

当连接器 ID 线状态发生更改时，模块将该位置 1。

注： *在设备模式和主机模式均可访问。*

位 27 **LPMINT**：LPM 中断 (LPM interrupt)

在设备模式下，当设备接收到 LPM 事务并以非 ERROR 应答时，将触发该中断。

在主机模式下，当设备以非 ERROR 应答 LPM 事务或主机模块完成所编程次数（OTG_GLPMCFG 中的 RETRYCNT 位）的 LPM 事务时，将触发该中断。

只有 OTG_GLPMCFG 中的 LPMCAP 位置 1 时，该字段才有效。

位 27 保留，对于 USB OTG FS，必须保持复位值。

位 26 **PTXFE**：周期性 Tx FIFO 为空 (Periodic Tx FIFO empty)

当周期性发送 FIFO 为半空或全空状态，且周期性请求队列中存在可写入至少一个条目的空间时，将触发该中断。该 FIFO 为半空状态还是全空状态由 OTG_GAHBCFG 寄存器中的周期性 Tx FIFO 空门限位（OTG_GAHBCFG 中的 PTXFELVL 位）决定。

注： 仅可在主机模式下访问。

位 25 **HCINT**：主机通道中断 (Host channels interrupt)

模块将该位置 1 时，指示模块中一个通道上存在挂起的中断（在主机模式下）。应用程序必须读取 OTG_HAINT 寄存器，以确定发生中断的通道的准确编号，然后读取相应的 OTG_HCINTx 寄存器，以确定引发中断的确切原因。应用程序必须先将 OTG_HCINTx 寄存器的相应状态位清零，之后才能将该位清零。

注： 仅可在主机模式下访问。

位 24 **HPRTINT**：主机端口中断 (Host port interrupt)

模块将该位置 1 时，指示主机模式下 OTG_FS/OTG_HS 控制器端口的状态发生变化。应用程序必须读取 OTG_HPRT 寄存器，以确定引发此中断的确切事件。应用程序必须先将 OTG_HPRT 寄存器的相应状态位清零，之后才能将该位清零。

注： 仅可在主机模式下访问。

位 23 **RSTDET**：检测到复位中断 (Reset detected interrupt)

在设备模式下，若设备处于挂起状态，则在部分断电模式下的 USB 上检测到复位时，将触发该中断。

注： 仅可在设备模式下访问。

位 23 保留，对于 USB OTG HS，必须保持复位值。

位 22 保留，对于 USB OTG FS，必须保持复位值。

位 22 **DATAFSUSP**：USB OTG HS 的数据获取挂起 (Data fetch suspended for USB OTG HS)

该中断仅在 DMA 模式下有效。该中断指示，模块因 TxFIFO 空间或请求队列空间不可用而停止为 IN 端点获取数据。应用程序进行端点不匹配处理时会用到该中断。例如，在检测到端点不匹配后，应用程序将执行以下操作：

- 将全局非周期性 IN NAK 握手信号置 1
- 禁止 IN 端点
- 清空 FIFO
- 根据 IN 令牌序列学习队列确定令牌序列
- 重新使能端点

如果全局非周期性 IN NAK 已清零但模块尚未为 IN 端点获取数据，同时又已接收到 IN 令牌，则清零全局非周期性 IN NAK 握手信号：模块将产生“FIFO 为空时接收到 IN 令牌”中断。然后，OTG 将 NAK 响应发送到主机。为避免这种情况的发生，应用程序可以检查 OTG_GINTSTS 中的 FetSusp 中断，该中断可确保在 FIFO 存满后再将全局 NAK 握手信号清零。或者，应用程序可以在将全局 IN NAK 握手信号清零时屏蔽“当 FIFO 为空时接收到 IN 令牌”中断。

位 21 **IPXFR**：未完成周期性传输 (Incomplete periodic transfer)

在主机模式下，如果存在仍处于挂起状态的未完成周期性事务，而这些事务计划在当前帧期间完成，则模块会将该中断位置 1。

INCOMPISOOUT：未完成 OUT 同步传输 (Incomplete isochronous OUT transfer)

在设备模式下，模块将该中断置 1 时，指示当前帧中至少有一个同步 OUT 端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断 (EOPF) 位一同触发。

位 20 **IISOIXFR**：未完成 IN 同步传输 (Incomplete isochronous IN transfer)

模块将该中断置 1 时，指示当前帧中至少有一个同步 IN 端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断 (EOPF) 位一同触发。

注： 仅可在设备模式下访问。

位 19 **OEPINT**：OUT 端点中断 (OUT endpoint interrupt)

模块将该位置 1 时，指示模块中一个 OUT 端点上存在挂起的中断（在设备模式下）。应用程序必须读取 OTG_DAINT 寄存器，以确定发生中断的 OUT 端点的准确编号，然后读取相应的 OTG_DOEPINTx 寄存器，以确定引发中断的确切原因。应用程序必须先将相应 OTG_DOEPINTx 寄存器的相应状态位清零，之后才能将该位清零。

注： 仅可在设备模式下访问。

位 18 **IEPINT**：IN 端点中断 (IN endpoint interrupt)

模块将该位置 1 时，指示模块中一个 IN 端点上存在挂起的中断（在设备模式下）。应用程序必须读取 OTG_DAINT 寄存器，以确定发生中断的 IN 端点的准确编号，然后读取相应的 OTG_DIEPINTx 寄存器，以确定引发中断的确切原因。应用程序必须先将相应 OTG_DIEPINTx 寄存器的相应状态位清零，之后才能将该位清零。

注： 仅可在设备模式下访问。

位 17:16 保留，必须保持复位值。

位 15 **EOPF**：周期性帧结束中断 (End of periodic frame interrupt)

指示当前帧已达到 OTG_DCFG 寄存器中周期性帧间隔字段（OTG_DCFG 中的 PFIVL 位）所指定的周期。

注： 仅可在设备模式下访问。

位 14 **ISOODRP**：丢弃同步 OUT 数据包中断 (Isochronous OUT packet dropped interrupt)

如果由于 Rx FIFO 空间不足，无法容纳同步 OUT 端点的最大数据包，从而导致模块无法向 Rx FIFO 写入同步 OUT 数据包，模块会将该位置 1。

注： 仅可在设备模式下访问。

位 13 **ENUMDNE**：枚举完成 (Enumeration done)

模块将该位置 1 时，指示速度枚举已完成。应用程序必须读取 OTG_DSTS 寄存器来获取枚举速度。

注： 仅可在设备模式下访问。

位 12 **USBRST**：USB 复位 (USB reset)

模块将该位置 1 时，指示在 USB 上检测到复位信号。

注： 仅可在设备模式下访问。

位 11 **USBSUSP**：USB 挂起 (USB suspend)

模块将该位置 1 时，指示在 USB 上检测到挂起状态。当数据线上的空闲状态保持一段额外的时间后，模块进入挂起状态。

注： 仅可在设备模式下访问。

位 10 **ESUSP**：早期挂起 (Early suspend)

模块将该位置 1 时，指示已检测到 USB 处于空闲状态的时间达到 3 ms。

注： 仅可在设备模式下访问。

位 9:8 保留，必须保持复位值。

位 7 **GONAKEFF**：全局 OUT NAK 有效 (Global OUT NAK effective)

指示 OTG_DCTL 寄存器中由应用程序设置的“将全局 OUT NAK 置 1”位（OTG_DCTL 中的 SGONAK 位）已在模块中生效。通过写入 OTG_DCTL 寄存器中的“将全局 OUT NAK 清零”位（OTG_DCTL 中的 CGONAK 位），可将该位清零。

注： 仅可在设备模式下访问。

位 6 **GINAKEFF**：全局非周期性 IN NAK 有效 (Global IN nonperiodic NAK effective)

指示 OTG_DCTL 寄存器中由应用程序设置的“将全局非周期性 IN NAK 置 1”位（OTG_DCTL 中的 SGINAK 位）已在模块中生效。也就是说，模块已对应用程序设置的全局 IN NAK 位进行采样，结果已生效。通过清零 OTG_DCTL 寄存器中的“将全局非周期性 IN NAK 清零”位（OTG_DCTL 中的 CGINAK 位），可将该位清零。

此中断不一定表示 USB 上已发送了一个 NAK 握手信号。STALL 位优先级高于 NAK 位。

注： 仅可在设备模式下访问。

位 5 **NPTXFE**：非周期性 Tx FIFO 为空 (Non-periodic Tx FIFO empty)

当非周期性 Tx FIFO 为半空或全空状态，且非周期性发送请求队列中至少存在可写入一个条目的空间时，将触发该中断。该 FIFO 为半空状态还是全空状态由 OTG_GAHBCFG 寄存器中的非周期性 Tx FIFO 空门限位（OTG_GAHBCFG 中的 TXFELVL 位）决定。

注： 仅可在主机模式下访问。

位 4 **RXFLVL**：Rx FIFO 非空 (Rx FIFO non-empty)

指示 Rx FIFO 中至少有一个数据包等待读取。

注： 在主机模式和设备模式均可访问。

位 3 **SOF**：帧起始 (Start of frame)

在主机模式下，模块将该位置 1 时，指示 USB 上已发送一个 SOF (FS) 或 Keep-Alive (LS) 信号。应用程序必须将此位置 1 才可清除该中断。

在设备模式下，模块将该位置 1 时，指示 USB 上已接收到一个 SOF 令牌。应用程序可通过读取 OTG_DSTS 寄存器来获得当前的帧编号。只有在模块以 FS 模式运行时，才会出现此中断。

注： 注意：如果上电复位后立即读取，该寄存器可能返回“1”。如果寄存器位在上电复位后立即读取“1”，则不表示已发送 SOF（主机模式下）或已接收 SOF（设备模式下）。只有在主机和设备之间建立有效连接后，此中断的读取值才有效。如果此位在上电复位后置 1，应用程序可把该位清零。

注： 在主机模式和设备模式均可访问。

位 2 **OTGINT**：OTG 中断 (OTG interrupt)

模块将该位置 1 时，指示出现 OTG 协议事件。应用程序必须读取 OTG 中断状态 (OTG_GOTGINT) 寄存器，以确定引发此中断的确切事件。应用程序必须先将 OTG_GOTGINT 寄存器的相应状态位清零，之后才能将该位清零。

注： 在主机模式和设备模式均可访问。

位 1 **MMIS**：模式不匹配中断 (Mode mismatch interrupt)

当应用程序尝试访问以下寄存器时，模块将该位置 1：

– 模块运行在设备模式下访问主机模式寄存器
– 模块运行在主机模式下访问设备模式寄存器

寄存器访问在 AHB 上以 OKAY 响应结束，但该访问在内部被模块忽略并且不会影响模块运行。

注： 在主机模式和设备模式均可访问。

位 0 **CMOD**：当前工作模式 (Current mode of operation)

指示当前模式。

0：设备模式

1：主机模式

注： 在主机模式和设备模式均可访问。

37.15.7 OTG 中断屏蔽寄存器 (OTG_GINTMSK)

OTG interrupt mask register

偏移地址：0x018

复位值：0x0000 0000

该寄存器与模块中断寄存器结合使用，以中断应用程序。如果将某个中断位屏蔽，则不会产生与该位相关的中断。但是，与该中断相对应的模块中断 (OTG_GINTSTS) 寄存器位仍会置 1。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WUIM	SRQIM	DISCINT	CIDSCHGM	LPMINTM	PTXFEM	HCIM	PRTIM	RSTDETM	Res.	IPXFRM/IISOOXFRM	IISOIXFRM	OEPINT	IEPINT	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	r	rw		rw	rw	rw	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EOPFM	ISOODRPM	ENUMDNEM	USBRST	USBSUSPM	ESUSPM	Res.	Res.	GONAKEFFM	GINAKEFFM	NPTXFEM	RXFLVLM	SOFM	OTGINT	MMISM	Res.
rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	rw	

注： *USB OTG FS 的配置寄存器*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WUIM	SRQIM	DISCINT	CIDSCHGM	LPMINTM	PTXFEM	HCIM	PRTIM	RSTDETM	FSUSPM	IPXFRM/IISOOXFRM	IISOIXFRM	OEPINT	IEPINT	Res.	Res.
rw	rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw	rw	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EOPFM	ISOODRPM	ENUMDNEM	USBRST	USBSUSPM	ESUSPM	Res.	Res.	GONAKEFFM	GINAKEFFM	NPTXFEM	RXFLVLM	SOFM	OTGINT	MMISM	Res.
rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	rw	

注： *USB OTG HS 的配置寄存器*

位 31 **WUIM**：检测到恢复/远程唤醒中断屏蔽 (Resume/remote wakeup detected interrupt mask)

0：屏蔽中断

1：使能中断

注： *在主机模式和设备模式均可访问。*

位 30 **SRQIM**：检测到会话请求/新会话中断屏蔽 (Session request/new session detected interrupt mask)

0：屏蔽中断

1：使能中断

注： *在主机模式和设备模式均可访问。*

位 29 **DISCINT**：检测到断开连接中断屏蔽 (Disconnect detected interrupt mask)

0：屏蔽中断

1：使能中断

注： *仅可在设备模式下访问。*

位 28 **CIDSCHGM**：连接器 ID 状态更改屏蔽 (Connector ID status change mask)
0：屏蔽中断
1：使能中断
注： 在主机模式和设备模式均可访问。

位 27 **LPMINTM**：LPM 中断屏蔽 (LPM interrupt mask)
0：屏蔽中断
1：使能中断
注： 在主机模式和设备模式均可访问。

位 26 **PTXFEM**：周期性 Tx FIFO 空屏蔽 (Periodic Tx FIFO empty mask)
0：屏蔽中断
1：使能中断
注： 仅可在主机模式下访问。

位 25 **HCIM**：主机通道中断屏蔽 (Host channels interrupt mask)
0：屏蔽中断
1：使能中断
注： 仅可在主机模式下访问。

位 24 **PRTIM**：主机端口中断屏蔽 (Host port interrupt mask)
0：屏蔽中断
1：使能中断
注： 仅可在主机模式下访问。

位 23 **RSTDETM**：检测到复位中断屏蔽 (Reset detected interrupt mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 22 保留，对于 USB OTG FS，必须保持复位值。

位 22 **FSUSPM**：USB OTG HS 的数据获取挂起屏蔽 (Data fetch suspended mask for USB OTG HS)
0：屏蔽中断
1：使能中断
仅可在设备模式下访问。

位 21 **IPXFRM**：未完成周期性传输中断屏蔽 (Incomplete periodic transfer mask)
0：屏蔽中断
1：使能中断
注： 仅可在主机模式下访问。
IISOOXFRM：未完成 OUT 同步传输中断屏蔽 (Incomplete isochronous OUT transfer mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 20 **IISOIXFRM**：未完成 IN 同步传输中断屏蔽 (Incomplete isochronous IN transfer mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 19 **OEPINT**：OUT 端点中断屏蔽 (OUT endpoints interrupt mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 18 **IEPINT**：IN 端点中断屏蔽 (IN endpoints interrupt mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 17:16 保留，必须保持复位值。

位 15 **EOPFM**：周期性帧结束中断屏蔽 (End of periodic frame interrupt mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 14 **ISOODRPM**：丢弃同步 OUT 数据包中断屏蔽 (Isochronous OUT packet dropped interrupt mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 13 **ENUMDNEM**：枚举完成中断屏蔽 (Enumeration done mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 12 **USBRST**：USB 复位中断屏蔽 (USB reset mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 11 **USBSUSPM**：USB 挂起中断屏蔽 (USB suspend mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 10 **ESUSPM**：早期挂起中断屏蔽 (Early suspend mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 9:8 保留，必须保持复位值。

位 7 **GONAKEFFM**：全局 OUT NAK 生效中断屏蔽 (Global OUT NAK effective mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 6 **GINAKEFFM**：全局非周期性 IN NAK 生效中断屏蔽 (Global non-periodic IN NAK effective mask)
0：屏蔽中断
1：使能中断
注： 仅可在设备模式下访问。

位 5 **NPTXFEM**：非周期性 Tx FIFO 空中断屏蔽 (Non-periodic Tx FIFO empty mask)
0：屏蔽中断
1：使能中断
注： 仅可在主机模式下访问。

位 4 **RXFLVLM**：接收 FIFO 非空中断屏蔽 (Receive FIFO non-empty mask)
0：屏蔽中断
1：使能中断
注：　在设备模式和主机模式均可访问。

位 3 **SOFM**：帧起始中断屏蔽 (Start of frame mask)
0：屏蔽中断
1：使能中断
注：　在设备模式和主机模式均可访问。

位 2 **OTGINT**：OTG 中断屏蔽 (OTG interrupt mask)
0：屏蔽中断
1：使能中断
注：　在设备模式和主机模式均可访问。

位 1 **MMISM**：模式不匹配中断屏蔽 (Mode mismatch interrupt mask)
0：屏蔽中断
1：使能中断
注：　在设备模式和主机模式均可访问。

位 0 保留，必须保持复位值。

37.15.8 OTG_FS 接收状态调试读取/OTG 状态读取和出栈寄存器 (OTG_GRXSTSR/OTG_GRXSTSP)

OTG_FS Receive status debug read/OTG status read and pop registers

读取的偏移地址：0x01C

出栈的偏移地址：0x020

复位值：0x0000 0000

读取接收状态调试读取寄存器将返回接收 FIFO 顶部的内容。读取接收状态读取和出栈寄存器将额外弹出 Rx FIFO 顶部的数据条目。

接收状态内容在主机模式和设备模式下的解释不同。当接收 FIFO 为空时，模块会忽略对该寄存器的读取或出栈操作，并返回值 0x0000 0000。当模块中断寄存器的接收 FIFO 非空位（OTG_GINTSTS 中的 RXFLVL 位）置位时，应用程序必须仅弹出接收状态 FIFO。

主机模式：

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.	Res.	Res.	PKTSTS				DPID
											r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPID	BCNT											CHNUM			
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:21 保留，必须保持复位值。

位 20:17 **PKTSTS**：数据包状态 (Packet status)
指示接收的数据包的状态
0010：接收到 IN 数据包
0011：IN 传输完成（触发中断）
0101：数据同步错误（触发中断）
0111：暂停通道（触发中断）
其它值：保留

位 16:15 **DPID**：数据 PID (Data PID)
指示接收的数据包的数据 PID
00：DATA0
10：DATA1
01：DATA2
11：MDATA

位 14:4 **BCNT**：字节数 (Byte count)
指示接收的 IN 数据包的字节数。

位 3:0 **CHNUM**：通道编号 (Channel number)
指示当前接收的数据包所属的通道编号。

设备模式：

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRMNUM				PKTSTS				DPID
							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
DPID	BCNT											EPNUM			
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:25 保留，必须保持复位值。

位 24:21 **FRMNUM**：帧编号 (Frame number)
这是在 USB 上接收的数据包帧编号的 4 个最低有效位。只有当支持同步 OUT 端点时才支持此字段。

位 20:17 **PKTSTS**：数据包状态 (Packet status)
指示接收的数据包的状态。
0001：全局 OUT NAK（触发中断）
0010：接收到 OUT 数据包
0011：OUT 传输完成（触发中断）
0100：SETUP 事务完成（触发中断）
0110：接收到 SETUP 数据包
其它值：保留

位 16:15 **DPID**：数据 PID (Data PID)
指示接收的 OUT 数据包的数据 PID。
00：DATA0
10：DATA1
01：DATA2
11：MDATA

位 14:4 **BCNT**：字节数 (Byte count)
指示接收的数据包的字节数。

位 3:0 **EPNUM**：端点编号 (Endpoint number)
指示当前接收的数据包所属的端点编号。

37.15.9 OTG 接收 FIFO 大小寄存器 (OTG_GRXFSIZ)

OTG Receive FIFO size register

偏移地址：0x024

复位值：0x0000 0200

此应用程序可以对分配给 Rx FIFO 的 RAM 大小进行编程。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXFD															
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 **RXFD**：Rx FIFO 深度 (Rx FIFO depth)
以 32 位字为单位。
最小值为 16
最大值为 1024
编程值必须遵循可分配的 FIFO 存储区，不得超过上电值。

37.15.10 OTG 主机非周期性发送 FIFO 大小寄存器 (OTG_HNPTXFSIZ)/ 端点 0 发送 FIFO 大小 (OTG_DIEPTXF0)

OTG Host non-periodic transmit FIFO size register

偏移地址：0x028

复位值：0x0200 0200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NPTXFD/TX0FD															
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
NPTXFSA/TX0FSA															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

主机模式

位 31:16 **NPTXFD**：非周期性 Tx FIFO 深度 (Non-periodic Tx FIFO depth)
以 32 位字为单位。
最小值为 16
编程值必须遵循可分配的 FIFO 存储区，不得超过上电值。

位 15:0 **NPTXFSA**：非周期性发送 RAM 起始地址 (Non-periodic transmit RAM start address)
此字段配置非周期性发送 FIFO RAM 的存储器起始地址。

设备模式

位 31:16 **TX0FD**：端点 0 Tx FIFO 深度 (Endpoint 0 Tx FIFO depth)
以 32 位字为单位。
最小值为 16
编程值必须遵循可分配的 FIFO 存储区，不得超过上电值。

位 15:0 **TX0FSA**：端点 0 发送 RAM 起始地址 (Endpoint 0 transmit RAM start address)
此字段配置端点 0 发送 FIFO RAM 的存储器起始地址。

37.15.11 OTG 非周期性发送 FIFO/队列状态寄存器 (OTG_HNPTXSTS)

OTG non-periodic transmit FIFO/queue status register

偏移地址：0x02C

复位值：0x0008 0200

注： *在设备模式下，此寄存器无效。*

此只读寄存器包含非周期性 Tx FIFO 和非周期性发送请求队列的空闲空间信息。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	NPTXQTOP							NPTQXSAV							
	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
NPTXFSAV															
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 30:24 **NPTXQTOP**：非周期性发送请求队列顶部 (Top of the non-periodic transmit request queue)
非周期性发送请求队列中 MAC 目前正在处理的条目。
位 30:27：通道/端点编号 (Channel/endpoint number)
位 26:25：
00：IN/OUT 令牌
01：长度为零的发送数据包（设备 IN/主机 OUT）
11：通道停止命令
位 24：结束（所选通道/端点的最后一个条目）
(Terminate (last entry for selected channel/endpoint))

位 23:16 **NPTQXSAV**：非周期性发送请求队列可用空间 (Non-periodic transmit request queue space available)

指示非周期性发送请求队列中的可用空闲空间大小。该队列既包含 IN 请求，又包含 OUT 请求。

0：非周期性发送请求队列已满

1：1 个位置可用

2：2 个位置可用

n：n 个位置可用 (0 ≤ n ≤ 8)

其它值：保留

位 15:0 **NPTXFSAV**：非周期性 Tx FIFO 可用空间 (Non-periodic Tx FIFO space available)

指示非周期性 Tx FIFO 中的可用空闲空间大小。

以 32 位字为单位。

0：非周期性 TxFIFO 已满

1：1 个字可用

2：2 个字可用

n：n 个字可用（其中，0 ≤ n ≤ 512）

其它值：保留

37.15.12 OTG I²C 访问寄存器 (OTG_GI2CCTL)

OTG I²C access register

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
BSY DNE	RW	Res.	I2CD ATSE	I2CDEVADR		Res.	ACK	I2CEN	ADDR						
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
REGADDR								RWDATA							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **BSYDNE**：I2C 忙碌/完成 (I2C Busy/Done)

应用程序将该位置 1 可启动 I²C 接口上的请求。传输完成后，模块会将此位清零。只要该位置 1，即表示 I²C 接口忙碌，应用程序便无法启动该接口的另一请求。

位 30 **RW**：读/写指示符 (Read/Write Indicator)

该位指示接口上必须执行读寄存器传输还是写寄存器传输。

0：写

1：读

注：　寄存器不支持突发读/写操作。

位 29 保留，必须保持复位值。

位 28 **I2CDATSE0**：I²C DatSe0 USB 模式 (I²C DatSe0 USB mode)
该位用于选择全速接口 USB 模式。
0：VP_VM USB 模式
1：DAT_SE0 USB 模式

位 27:26 **I2CDEVADR**：I²C 器件地址 (I²C Device Address)
利用此位可选择模块用于进行 OTG 信号传输所使用的 USB 1.1 全速串行收发器的 I²C 从器件的地址。

位 25 保留，必须保持复位值。

位 24 **ACK**：I²C ACK
该位指示是否从 I²C 从器件中接收到 ACK 应答。当应用程序启动 I²C 访问，模块将 BSYDNE 位清零后，该位才有效。
0：NAK
1：ACK

位 23 **I2CEN**：I²C 使能 (I²C Enable)
该位可使能 I²C 主器件在 I²C 接口上启动通信事务。

位 22:16 **ADDR**：I²C 地址 (I²C Address)
此位是 7 位 I²C 器件地址，应用程序利用此地址来访问任何外部 I²C 从器件，包括 USB 1.1 OTG 全速串行收发器上的 I²C 从器件。

位 15:8 **REGADDR**：I²C 寄存器地址 (I²C Register Address)
这些位用于编程要读取/写入的寄存器的地址。

位 7:0 **RWDATA**：I²C 读/写数据 (I²C Read/Write Data)
执行寄存器读操作之后，这些位可存储应用程序的读取数据。
执行写操作期间，应用程序可使用此寄存器编程要写入寄存器的数据。

注： *配置寄存器只适用于 USB OTG HS*

37.15.13 OTG 通用模块配置寄存器 (OTG_GCCFG)

OTG general core configuration register

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.	VBDEN	Res.	Res.	Res.	Res.	PWR DWN
										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 保留，必须保持复位值。

位 21 **VBDEN**：USB V_{BUS} 检测使能 (USB V_{BUS} detection enable)
使能 V_{BUS} 感应比较器检测 USB 主机和设备工作模式下的 V_{BUS} 引脚上的 V_{BUS} 有效电平。如果使能 HNP 和/或 SRP 支持，将自动使能 V_{BUS} 比较器，而与 VBDEN 值无关。
0 = 禁止 V_{BUS} 检测
1 = 使能 V_{BUS} 检测

位 20:17 保留，必须保持复位值。

位 16 **PWRDWN**：掉电控制 (Power down control)
用于在发送/接收时激活收发器 复位时，收发器保持掉电状态。0 = 禁止 USB FS 收发器
1 = 使能 USB FS 收发器

位 15:4 保留，必须保持复位值。

位 3:0 保留，必须保持复位值。

37.15.14 OTG 模块 ID 寄存器 (OTG_CID)

OTG core ID register

偏移地址：0x03C

复位值：0x0000 2000

该寄存器为只读寄存器，包含产品 ID。

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

位 31:0 **PRODUCT_ID**：产品 ID 字段 (Product ID field)
可通过应用程序编程的 ID 字段。

37.15.15 OTG 模块 LPM 配置寄存器 (OTG_GLPMCFG)

OTG core LPM configuration 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.	EN BESL	LPMRCNTSTS			SND LPM	LPMRCNT			LPMCHIDX				L1RSM OK
			rw	r	r	r	rs	rw	rw	rw	rw	rw	rw	rw	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLP STS	LPMRSP		L1DS EN	BESLTHRS				L1SS EN	REM WAKE	BESL				LPM ACK	LPM EN
r	r	r	rw	rw	rw	rw	rw	rw	rw/r	rw/r	rw/r	rw/r	rw/r	rw	rw

位 31:29 保留，必须保持复位值。

位 28 **ENBESL**：使能尽力服务延迟 (Enable best effort service latency)

该位用于使能 LPM 勘误表中定义的 BESL 功能：

0：模块按照以下文档中的说明工作：
基于 USB 2.0 规范的 USB 2.0 链路层电源管理补充工程变更通知，2007 年 7 月 16 日

1：模块按照 LPM 勘误表中的说明工作：
USB 2.0 ECN 的勘误表：链路层电源管理 (LPM) - 2007 年 7 月

注： 本文档只考虑（LPM 勘误表中说明的）更新行为，因此 ENBESL 位应由应用程序软件置 1。

位 27:25 **LPMRCNTSTS**：LPM 重试计数状态 (LPM retry count status)

当前 LPM 序列仍要发送的 LPM 主机重试次数

注： 仅可在主机模式下访问。

位 24 **SNDLPM**：发送 LPM 事务 (Send LPM transaction)

当应用程序软件将该位置 1 时，将发送包含两个令牌 EXT 和 LPM 的 LPM 事务。从设备接收到有效响应（STALL、NYET 或 ACK）或模块发送完编程的 LPM 重试次数后，硬件即将该位清零。

注： 只有当主机连接到本地端口时，该位才可置 1。

注： 仅可在主机模式下访问。

位 23:21 **LPMRCNT**：LPM 重试计数 (LPM retry count)

当设备发出 ERROR 响应后，该位表示接收到有效设备响应（STALL、NYET 或 ACK）之前，主机需额外执行的 LPM 重试次数。

注： 仅可在主机模式下访问。

位 20:17 **LPMCHIDX**：LPM 通道索引 (LPM Channel Index)

向本地设备发送 LPM 事务时，必须应用到 LPM 事务的通道编号。基于 LPM 通道编号，模块可自动向 LPM 事务插入相应通道中编程的设备地址和端点号。

注： 仅可在主机模式下访问。

位 16 **L1RSMOK**：睡眠状态恢复正常 (Sleep State Resume OK)

表示设备或主机可以从睡眠状态启动恢复。该位在 LPM 睡眠 (L1) 状态下有效。它在睡眠模式下经 50 µs ($T_{L1Residency}$) 延时后置 1。

该位在 SLPSTS 为 0 时复位。

1：应用程序或主机可以从睡眠状态启动恢复

0：应用程序或主机无法从睡眠状态启动恢复

位 15 **SLPSTS**：端口睡眠状态 (Port sleep status)

设备模式

只要 USB 总线上存在睡眠条件，该位就会置 1。当 ACK 响应发送给 LPM 事务且 $T_{L1TokenRetry}$ 定时器过期时，模块将进入睡眠模式。要停止 PHY 时钟，应用程序必须将 OTG_PCGCCTL 中的 STPPCLK 位置 1，这将触发 PHY 挂起输入信号。

应用程序必须依靠 LPMRSP 中的 SLPSTS 而不是 ACK 来确认切换到睡眠状态。

出现以下情况时，模块会退出睡眠状态：

– USB 线上有活动时

– 当应用程序对 OTG_DCTL 中的 RWUSIG 位执行写操作或者复位或软断开设备时。

主机模式

模块通过来自设备的 ACK 响应成功将 LPM 事务发送给本地端口后，主机将切换到睡眠 (L1) 状态。此位的读取值反映该端口的当前睡眠状态。

在以下事件后，模块会将该位清零：

– 模块检测到远程 L1 唤醒信号，

– 应用程序将 OTG_HPRT 寄存器中的 PRST 位或 PRES 位置 1，或者

– 应用程序将模块中断寄存器中的“检测到 L1 恢复/远程唤醒中断”位或“检测到断开连接中断”位（分别为 OTG_GINTSTS 中的 WKUPINT 或 DISCINT 位）置 1。

0：模块未处于 L1

1：模块处于 L1

位 14:13 **LPMRST**：LPM 响应 (LPM response)

设备模式

这两位反映模块对所接收的 LPM 事务的响应。

主机模式

从本地设备接收、用于 LPM 事务的握手响应

11：ACK

10：NYET

01：STALL

00：ERROR（无握手响应）

位 12 **L1DSEN**：L1 深度睡眠使能 (L1 deep sleep enable)

在 L1 睡眠模式下使能挂起 PHY。为了在 L1 睡眠模式下最大程度节能，所有情况下该位均应由应用程序软件置 1。

位 11:8 **BESLTHRS**：BESL 阈值 (BESL threshold)

设备模式

当 BESL 值大于或等于 BESL_Thres[3:0] 字段中定义的值时，模块将 PHY 置于 L1 下的深度低功耗模式。

主机模式

模块将 PHY 置于 L1 下的深度低功耗模式。BESLTHRS[3:0] 用于指定要由主机在 USB 总线上反映的恢复信号的持续时间 ($T_{L1HubDrvResume2}$)。

在主机模式下，BESLTHRS 不得编程为大于 1100b 的值，原因是这将超出最大 $T_{L1HubDrvResume2}$。

Thres[3:0]	主模式恢复信号的持续时间 (µs)
0000:	75
0001:	100
0010:	150
0011:	250
0100:	350
0101:	450
0110:	950
所有其它值：	保留

位 7 **L1SSEN**：L1 浅度睡眠使能 (L1 Shallow Sleep enable)

在 L1 睡眠模式下使能挂起 PHY。为了在 L1 睡眠模式下最大程度节能，所有情况下该位均应由应用程序软件置 1。

位 6 **REMWAKE**：bRemoteWake 值 (bRemoteWake value)

主机模式

要在 LPM 事务的 wIndex 字段中发送的远程唤醒值。

器件模式（只读）：

当 ACK、NYET 或 STALL 响应发送给 LPM 事务时，该字段用接收的 LPM 令牌 bRemoteWake bmAttribute 进行更新。

位 5:2 **BESL**：尽力服务延迟 (Best effort service latency)

主机模式

要在 LPM 事务中发送的 BESL 值 该值还用于发起持续时间为 $T_{L1HubDrvResume1}$ 的恢复信号，以实现主机发起的恢复操作。

器件模式（只读）

当 ACK、NYET 或 STALL 响应发送给 LPM 事务时，该字段用接收的 LPM 令牌 BESL bmAttribute 进行更新。

BESL[3:0]	T_{BESL} (µs)
0000:	125
0001:	150
0010:	200
0011:	300
0100:	400
0101:	500
0110:	1000
0111:	2000
1000:	3000
1001:	4000
1010:	5000
1011:	6000
1100:	7000
1101:	8000
1110:	9000
1111:	10000

位 1 **LPMACK**：LPM 令牌应答使能 (LPM token acknowledge enable)

器件应用程序软件预编程的 LPM 令牌握手响应。

1：ACK

即使已预编程 ACK，也只有在成功完成 LPM 事务后，模块才能以 ACK 进行响应。LPM 事务成功的前提是：

– EXT 令牌或 LPM 令牌中没有 PID/CRC5 错误（否则会出现 ERROR）
– LPM 事务中接收到 Valid bLinkState = 0001B (L1)（否则会出现 STALL）
– 发送队列中没有待处理的数据（否则会出现 NYET）

0：NYET

在以下情况下，为响应 LPM 令牌，会重写预编程的软件位：

– 接收到的 bLinkState 不是 L1（STALL 响应），或者
– 由于任一 LPM 令牌数据包损坏而在其中检测到错误（ERROR 响应）。

注： 仅可在设备模式下访问。

位 0 **LPMEN**：LPM 支持使能 (LPM support enable)
应用程序使用该位控制 OTG_FS/OTG_HS 模块的 LPM 功能。
如果模块以不支持 LPM 功能的主机工作，则无法请求所连接的设备或集线器激活 LPM 模式。
如果模块以不支持 LPM 功能的设备工作，则无法响应任何 LPM 事务。
0：不使能 LPM 功能
1：使能 LPM 功能

37.15.16 OTG 主机周期性发送 FIFO 大小寄存器 (OTG_HPTXFSIZ)

OTG Host periodic transmit FIFO size register

偏移地址：0x100

复位值：0x0200 0400

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

位 31:16 **PTXFD**：主机周期性传输 Tx FIFO 深度 (Host periodic Tx FIFO depth)
以 32 位字为单位。
最小值为 16

位 15:0 **PTXSA**：主机周期性传输 Tx FIFO 起始地址 (Host periodic Tx FIFO start address)
此字段用于配置周期性发送 FIFO RAM 的存储器起始地址。

37.15.17 OTG 设备 IN 端点发送 FIFO 大小寄存器（OTG_DIEPTXFx)（x = 1..5[FS] / 7[HS]，其中 x 为 FIFO 编号）

OTG device IN endpoint transmit FIFO size register

偏移地址：0x104 + (FIFO 编号 – 1) × 0x04

复位值：

FIFO 编号 = 7[HS] / 5[FS] : 0x0200 0200 + (7[HS] / 5[FS] * 0x200)

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

位 31:16 **INEPTXFD**：IN 端点 Tx FIFO 深度 (IN endpoint Tx FIFO depth)
以 32 位字为单位。
最小值为 16

位 15:0 **INEPTXSA**：IN 端点发送 FIFOx RAM 起始地址 (IN endpoint FIFOx transmit RAM start address)
此字段包含 IN 端点发送 FIFOx 的存储器起始地址。该地址必须与 32 位存储器位置对齐。

37.15.18 主机模式寄存器

Host-mode registers

除非特别说明，否则寄存器描述中的位值以二进制表示。

主机模式寄存器会影响主机模式下的模块操作。在设备模式下不得访问主机模式寄存器，因为产生的结果不明确。主机模式寄存器可进行如下分类：

37.15.19 OTG 主机配置寄存器 (OTG_HCFG)

OTG Host configuration register

偏移地址：0x400

复位值：0x0000 0000

此寄存器将在上电后对模块进行配置。请勿在初始化主机后更改此寄存器。

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.	Res.	Res.	Res.	FSLSS	FSLSPCS	
													r	rw	rw

位 31:3 保留，必须保持复位值。

位 2 **FSLSS**：仅支持 FS 和 LS (FS- and LS-only support)
应用程序使用此位控制模块的枚举速度。使用此位，应用程序可使模块工作为 FS 主机，即使所连接的设备支持 HS 通信也是如此。请勿在初始编程后更改此字段。
1：仅限 FS/LS，即使所连接设备可支持 HS（只读）

位 1:0 **FSLSPCS**：FS/LS PHY 时钟选择 (FS/LS PHY clock select)
当模块处于 FS 主机模式时
01：PHY 时钟以 48 MHz 运行
其它值：保留
当模块处于 LS 主机模式时
00：保留
01：选择 48 MHz PHY 时钟频率
10：选择 6 MHz PHY 时钟频率
11：保留

注： *当设备连上主机时，必须依照所连接设备的速度设置 FSLSPCS（更改此位后，必须进行软件复位）。*

37.15.20 OTG 主机帧间隔寄存器 (OTG_HFIR)

OTG Host frame interval register

偏移地址：0x404

复位值：0x0000 EA60

此寄存器用于存储 OTG_FS/OTG_HS 控制器对已连接设备当前速度所设定的帧间隔信息。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRIVL															
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 **RLDCTRL**：重载控制 (Reload control)

该位允许在运行期间动态重载 HFIR 寄存器。

0：无法动态重载 HFIR

1：运行期间可动态重载 HFIR。

该位需要在初始配置期间编程，并且不得在运行期间更改其值。

位 15:0 **FRIVL**：USB OTG FS 的帧间隔 (Frame interval for USB OTG FS)

应用程序在此字段编程的值用于指定两个连续 SOF (FS) 或 Keep-Alive 令牌 (LS) 之间的时间间隔。此字段包含构成所需帧间隔的 PHY 时钟数。只有将主机端口控制和状态寄存器的端口使能位（OTG_HPRT 的 PENA 位）置 1 后，应用程序才能向此寄存器中写入值。如果未对值进行编程，模块将根据在主机配置寄存器的 FS/LS PHY 时钟选择字段（OTG_HCFG 中的 FSLSPCS）中指定的 PHY 时钟来计算。除非 RLDCTRL 位置 1，否则请勿在初始配置后更改此字段的值。在这种情况下，FRIVL 将在发生每个 SOF 事件时进行重载。

位 15:0 **FRIVL**：USB OTG HS 的帧间隔 (Frame interval for USB OTG HS)

应用程序在此字段编程的值用于指定两个连续 micro-SOF (HS) 或 Keep-Alive 令牌 (LS) 之间的时间间隔。此字段包含构成所需帧间隔的 PHY 时钟数。只有将主机端口控制和状态寄存器的端口使能位（OTG_HPRT 的 PENA 位）置 1 后，应用程序才能向此寄存器中写入值。如果未对值进行编程，模块将根据在主机配置寄存器的 FS/LS PHY 时钟选择字段（OTG_HCFG 中的 FSLSPCS）中指定的 PHY 时钟来计算。除非 RLDCTRL 位置 1，否则请勿在初始配置后更改此字段的值。在这种情况下，FRIVL 将在发生每个 SOF 事件时进行重载。

37.15.21 OTG 主机帧编号/帧剩余时间寄存器 (OTG_HFNUM)

OTG Host frame number/frame time remaining register

偏移地址：0x408

复位值：0x0000 3FFF

此寄存器用于指示当前帧编号。它还指示当前帧的剩余时间（以 PHY 时钟数为单位）。

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

位 31:16 **FTREM**：帧剩余时间 (Frame time remaining)

指示当前帧的剩余时间（以 PHY 时钟数为单位）。每过去 1 个 PHY 时钟，此字段递减 1。当值达到零时，此字段将重新装载帧间隔寄存器中的值，并由模块在 USB 上发送一个新 SOF。

位 15:0 **FRNUM**：帧编号 (Frame number)

当在 USB 上发送 1 个新 SOF 时此字段的值将递增 1，当达到 0x3FFF 时会清零。

37.15.22 OTG_Host 周期性发送 FIFO/队列状态寄存器 (OTG_HPTXSTS)

OTG_Host periodic transmit FIFO/queue status register

偏移地址：0x410

复位值：0x0008 0100

此只读寄存器包含周期性 Tx FIFO 和周期性发送请求队列的空闲空间信息。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTXQTOP								PTXQSAV							
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
PTXFSAVL															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:24 **PTXQTOP**：周期性传输发送请求队列顶部 (Top of the periodic transmit request queue)
指示周期性传输 Tx 请求队列中 MAC 当前正在处理的项。
该寄存器用于调试。
位 31：奇数/偶数帧 (Odd/Even frame)
0：以偶数帧发送
1：以奇数帧发送
位 30:27：通道/端点编号 (Channel/endpoint number)
位 26:25：类型 (Type)
00：IN/OUT
01：零长度数据包
11：禁止通道命令
位 24：结束（所选通道/端点的最后一个条目）
(Terminate (last entry for the selected channel/endpoint))

位 23:16 **PTXQSAV**：周期性传输发送请求队列可用空间 (Periodic transmit request queue space available)
指示可供写入的周期性传输发送请求队列的空闲位置的数量。该队列既包含 IN 请求，又包含 OUT 请求。
00：周期性发送请求队列已满
01：1 个位置可用
10：2 个位置可用
bxn：n 个位置可用 (0 ≤ n ≤ 8)
其它值：保留

位 15:0 **PTXFSAVL**：周期性传输发送数据 FIFO 可用空间 (Periodic transmit data FIFO space available)
指示可供写入的周期性传输 Tx FIFO 的空闲位置的数量。
以 32 位字为单位
0000：周期性 Tx FIFO 已满
0001：1 个字可用
0010：2 个字可用
bxn：n 个字可用（其中 0 ≤ n ≤ PTXFD）
其它值：保留

37.15.23 OTG 主机全体通道中断寄存器 (OTG_HAINT)

OTG Host all channels interrupt register

偏移地址：0x414

复位值：0x0000 000

当通道上有事件发生时，主机全体通道中断寄存器会使用模块中断寄存器中的主机通道中断位（OTG_GINTSTS 中的 HCINT 位）中断应用程序。相关内容如*图 446* 所示。每个通道对应 1 个中断位，最多有 16 个位。当应用程序通过相应主机通道 x 中断寄存器清零中断时，该寄存器中的位也会清零。

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
HAINT															
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 **HAINT**：通道中断 (Channel interrupts)
每个通道一位：通道 0 对应位 0，通道 15 对应位 15

37.15.24 OTG 主机全体通道中断屏蔽寄存器 (OTG_HAINTMSK)

OTG Host all channels interrupt mask register

偏移地址：0x418

复位值：0x0000 0000

主机全体通道中断屏蔽寄存器与主机全体通道中断寄存器结合使用，进而在通道上发生事件时中断应用程序。每个通道对应 1 个中断屏蔽位，最多有 16 个位。

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
HAINTM															
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 **HAINTM**：通道中断屏蔽 (Channel interrupt mask)
0：屏蔽中断
1：使能中断
每个通道一位：通道 0 对应位 0，通道 15 对应位 15

37.15.25 OTG 主机端口控制和状态寄存器 (OTG_HPRT)

OTG Host port control and status register

偏移地址：0x440

复位值：0x0000 0000

该寄存器仅在主机模式下可用。当前，OTG 主机仅支持一个端口。

该寄存器包含与 USB 端口相关的信息，例如 USB 复位、使能、挂起、恢复、连接状态以及测试模式。相关内容在*图 446* 中进行了说明。该寄存器中的 rc_w1 位可通过模块中断寄存器中的主机端口中断位（OTG_GINTSTS 中的 HPRTINT 位）触发应用程序中断。发生端口中断时，应用程序必须读取该寄存器，并将引起中断的位清零。对于 rc_w1 位，应用程序必须向该位写入 1 以清除该中断。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSPD		PTCTL
													r	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTCTL			PPWR	PLSTS		Res.	PRST	PSUSP	PRES	POC CHNG	POCA	PEN CHNG	PENA	PCDET	PCSTS
rw	rw	rw	rw	r	r		rw	rs	rw	rc_w1	r	rc_w1	rc_w0	rc_w1	r

位 31:19 保留，必须保持复位值。

位 18:17 **PSPD**：端口速度 (Port speed)

指示连接到该端口的设备的速度。

01：全速

10：低速

11：保留

00：高速

位 16:13 **PTCTL**：端口测试控制 (Port test control)

应用程序向该字段写入一个非零值，以将端口置于测试模式，同时端口上会产生对应模式的信号。

0000：禁止测试模式

0001：Test_J 模式

0010：Test_K 模式

0011：Test_SE0_NAK 模式

0100：Test_Packet 模式

0101：Test_Force_Enable

其它值：保留

位 12 **PPWR**：端口电源 (Port power)

应用程序使用该字段控制该端口的电源，且发生过流情况时，模块会将该位清零。

0：掉电

1：上电

位 11:10 **PLSTS**：端口线状态 (Port line status)

指示 USB 数据线的当前逻辑电平

位 10：OTG_FS_DP 的逻辑电平

位 11：OTG_FS_DM 的逻辑电平

位 9 保留，必须保持复位值。

位 8 **PRST**：端口复位 (Port reset)

应用程序将该位置 1 时，会在该端口上启动复位序列。应用程序必须为复位周期定时，并在复位序列完成后将该位清零。

0：端口未处于复位状态

1：端口处于复位状态

应用程序必须将该位置 1 并最少保持 10 ms，以在端口上启动复位。除所需的最少持续时间之外，在将该位清零前，应用程序可将该位置 1 的状态再保持 10 ms，即使 USB 标准并没有设置最大限制。

高速：50 ms

全速/低速：10 ms

位 7 **PSUSP**：端口挂起 (Port suspend)

应用程序将此位置 1 以将此端口置于挂起模式。只有此位置 1 时，模块才会停止发送 SOF。要停止 PHY 时钟，应用程序必须将端口时钟停止位置 1，这会使能 PHY 的挂起输入引脚。

此位的读取值反映该端口的当前挂起状态。检测到远程唤醒信号，或者应用程序将此寄存器中的端口复位位或端口恢复位置 1 后，模块可将此位清零；或应用程序将模块中断寄存器中的检测到恢复/远程唤醒中断位或检测到断开连接中断位（分别为 OTG_GINTSTS 中的 WKUINT 或 DISCINT）置 1，模块也可将此位清零。

0：端口未处于挂起模式

1：端口处于挂起模式

位 6 **PRES**：端口恢复 (Port resume)

应用程序将此位置 1 以在该端口上驱动恢复信号。模块会持续驱动恢复信号直到应用程序将此位清零。

如模块中断寄存器中的检测到端口恢复/远程唤醒中断位（OTG_GINTSTS 中的 WKUINT 位）指示，如果模块检测到 USB 远程唤醒序列，则开始驱动恢复信号，而无需应用程序进行干预；如果模块检测到断开连接的情况，则将此位清零。此位的读取值指示当前模块是否正在驱动恢复信号。

0：不驱动恢复信号

1：驱动恢复信号

当 LPM 已使能且模块处于 L1 状态时，此位的影响如下：

1. 应用程序将此位置 1 以在该端口上驱动恢复信号。

2. 模块继续驱动恢复信号，直至达到 OTG_GLPMCFG 寄存器的 BESLTHRS[3:0] 字段预定的时间。

3. 如模块中断寄存器中的“检测到端口 L1 恢复/远程 L1 唤醒中断”位（OTG_GINTSTS 中的 WKUPINT 位）指示，如果模块检测到 USB 远程唤醒序列，则开始驱动恢复信号，而无需应用程序进行干预，并在恢复结束时，将此位清零。此位可以由模块和应用程序置 1 和清零。即使主机未连接任何设备，此位也可由模块清零。

位 5 **POCCHNG**：端口过流变化 (Port overcurrent change)

该寄存器中端口过流激活位（位 4）状态发生变化时，模块将此位置 1。

位 4 **POCA**：端口过流激活 (Port overcurrent active)

此位指示端口的过流状况。

0：无过流状况

1：有过流状况

位 3 **PENCHNG**：端口使能/禁止变化 (Port enable/disable change)

该寄存器中的端口使能位 2 的状态发生变化时，模块将此位置 1。

位 2 **PENA**：端口使能 (Port enable)

端口执行复位序列后，只能由模块使能，并且可以由过流状况、断开连接状况或应用程序将此位清零来禁止。应用程序无法通过对寄存器执行写操作将此位置 1。只能将此位清零来禁止端口。对此位的操作不会触发应用程序的任何中断。

0：禁止端口

1：使能端口

位 1 **PCDET**：检测到端口连接 (Port connect detected)

当检测到设备连接时，模块将此位置 1，以使用模块中断寄存器中的主机端口中断位（OTG_GINTSTS 中的 HPRTINT 位）触发应用程序的中断。应用程序必须将此位置 1 才可清除该中断。

位 0 **PCSTS**：端口连接状态 (Port connect status)

0：端口未连接设备

1：端口已连接设备

37.15.26 OTG 主机通道 x 特性寄存器 (OTG_HCCHARx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）

OTG Host channel-x characteristics register

偏移地址：0x500 + (通道编号 × 0x20)

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHENA	CHDIS	ODD FRM	DAD							MCNT		EPTYP		LSDEV	Res.
rs	rs	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
EPDIR	EPNUM				MPSIZ										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **CHENA**：通道使能 (Channel enable)

此字段由应用程序软件置 1，并由 OTG 主机硬件清零。

0：禁止通道

1：使能通道

位 30 **CHDIS**：禁止通道 (Channel disable)

应用程序将此位置 1 以停止通过通道发送/接收数据，即使通过该通道的传输还未完成，停止操作仍然生效。应用程序必须等待禁止通道的中断以确认通道已经被禁止。

位 29 **ODDFRM**：奇数帧 (Odd frame)

此字段由应用程序置位或复位，以分别指示 OTG 主机必须传输奇数帧或偶数帧。此字段只适用于周期性（同步和中断）事务。

0：偶数帧

1：奇数帧

位 28:22 **DAD**：设备地址 (Device Address)

此字段用于指定要与该主机通信的特定设备。

位 21:20 **MCNT**：多重计数 (Multicount)

此字段向主机指示该周期性端点每帧必须执行的事务数。该字段不用于非周期性传输。

00：保留。对该字段的操作会产生不明确的结果

01：1 个事务

10：该端点每帧需要发出 2 个事务

11：该端点每帧需要发出 3 个事务

注：　此字段至少须置为 01。

位 19:18 **EPTYP**：端点类型 (Endpoint type)

指示选择的传输类型。

00：控制

01：同步

10：批量

11：中断

位 17 **LSDEV**：低速设备 (Low-speed device)

此字段由应用程序置 1，表示此通道正在与一个低速设备进行通信。

位 16 保留，必须保持复位值。

位 15 **EPDIR**：端点方向 (Endpoint direction)
指示通信事务的方向是输入还是输出。
0：OUT
1：IN

位 14:11 **EPNUM**：端点编号 (Endpoint number)
指示要与该主机通道通信的 USB 设备的端点号。

位 10:0 **MPSIZ**：最大数据包大小 (Maximum packet size)
指示与该主机通道通信的设备端点的最大数据包大小。

37.15.27 OTG 主机通道 x 分离控制寄存器 (OTG_HCSPLTx)（x = 0..15，其中 x = 通道编号）

OTG Host channel-x split control register

偏移地址：0x504 + (通道编号 × 0x20)

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XACTPOS		HUBADDR							PRTADDR						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **SPLITEN**：分离使能 (Split enable)
应用程序将此位置 1 时，指示允许该通道执行分离通信事务。

位 30:17 保留，必须保持复位值。

位 16 **COMPLSPLT**：执行完全分离 (Do complete split)
应用程序将此位置 1 时，可请求 OTG 主机执行完全分离通信事务。

位 15:14 **XACTPOS**：事务位置 (Transaction position)
此字段用于决定随各 OUT 事务发送全部、第一个、中间还是最后一个有效负载。
11：全部。指此事务的全部数据有效负载（小于或等于 188 字节）
10：起始。指此事务的第一个数据有效负载（大于 188 字节）
00：中间。指此事务的中间有效负载（大于 188 字节）
01：结尾。指此事务的最后一个有效负载（大于 188 字节）

位 13:7 **HUBADDR**：集线器地址 (Hub address)
此字段存储事务转发器的集线器设备地址。

位 6:0 **PRTADDR**：端口地址 (Port address)
此字段是接收方事务转发器的端口编号。

注： *配置寄存器只适用于 USB OTG HS*

37.15.28 OTG 主机通道 x 中断寄存器 (OTG_HCINTx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）

OTG Host channel-x interrupt register

偏移地址：0x508 + (通道编号 × 0x20)

复位值：0x0000 0000

该寄存器指示在出现 USB 和 AHB 相关事件时通道的状态。相关内容在*图 446* 中进行了说明。当模块中断寄存器中的主机通道中断位（OTG_GINTSTS 中的 HCINT 位）置 1 时，应用程序必须读取该寄存器。在对寄存器执行读操作之前，应用程序必须先读取主机全体通道中断 (OTG_HAINT) 寄存器，以获取主机通道 x 中断寄存器的准确通道编号。应用程序必须将该寄存器中的相应位清零，才能将 OTG_HAINT 和 OTG_GINTSTS 寄存器中的对应位清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	DTERR	FRMOR	BBERR	TXERR	Res.	ACK	NAK	STALL	Res.	CHH	XFRC
					rc_w1	rc_w1	rc_w1	rc_w1		rc_w1	rc_w1	rc_w1		rc_w1	rc_w1

注： *USB OTG FS 的配置寄存器*

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.	DTERR	FRMOR	BBERR	TXERR	NYET	ACK	NAK	STALL	AHBERR	CHH	XFRC
					rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

注： *USB OTG HS 的配置寄存器*

位 31:11 保留，必须保持复位值。

位 10 **DTERR**：数据同步错误 (Data toggle error)

位 9 **FRMOR**：帧溢出 (Frame overrun)

位 8 **BBERR**：串扰错误 (Babble error)

位 7 **TXERR**：通信事务错误 (Transaction error)

指示 USB 上发生下列错误之一：
CRC 校验失败
超时
位填充错误
错误的 EOP

位 6 保留，对于 USB OTG FS，必须保持复位值。

位 6 **NYET**：USB OTG HS 的收到尚未就绪响应中断 (Not yet ready response received interrupt for USB OTG HS)

位 5 **ACK**：收到/发出 ACK 响应中断 (ACK response received/transmitted interrupt)

位 4 **NAK**：收到 NAK 响应中断 (NAK response received interrupt)

位 3 **STALL**：收到 STALL 响应中断 (STALL response received interrupt)

位 2 保留，对于 USB OTG FS，必须保持复位值。

位 2 **AHBERR**：USB OTG HS 的 AHB 错误 (AHB error for USB OTG HS)

仅当处于内部 DMA 模式下且 AHB 读/写操作期间发生 AHB 错误时才生成此错误。应用程序可读取相应的 DMA 通道地址寄存器来获取错误地址。

位 1 **CHH**：通道停止 (Channel halted)

因任意 USB 事务错误或为响应应用程序的禁止请求而导致传输非正常结束。

位 0 **XFRC**：传输完成 (Transfer completed)

未出现任何错误，正常完成传输。

37.15.29 OTG 主机通道 x 中断屏蔽寄存器 (OTG_HCINTMSKx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）

OTG Host channel-x interrupt mask register

偏移地址：0x50C + (通道编号 × 0x20)

复位值：0x0000 0000

此寄存器反映了先前部分中介绍的各通道状态的屏蔽情况。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	DTERRM	FRMORM	BBERRM	TXERRM	Res.	ACKM	NAKM	STALLM	Res.	CHHM	XFRCM
					rw	rw	rw	rw		rw	rw	rw		rw	rw

注： *USB OTG FS 的配置寄存器*

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.	DTERRM	FRMORM	BBERRM	TXERRM	NYET	ACKM	NAKM	STALLM	AHBERRM	CHHM	XFRCM
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

注： *USB OTG HS 的配置寄存器*

位 31:11 保留，必须保持复位值。

位 10 **DTERRM**：数据同步错误屏蔽 (Data toggle error mask)
0：屏蔽中断
1：使能中断

位 9 **FRMORM**：帧溢出屏蔽 (Frame overrun mask)
0：屏蔽中断
1：使能中断

位 8 **BBERRM**：串扰错误屏蔽 (Babble error mask)
0：屏蔽中断
1：使能中断

位 7 **TXERRM**：通信事务错误屏蔽 (Transaction error mask)
0：屏蔽中断
1：使能中断

位 6 保留，对于 USB OTG FS，必须保持复位值。

位 6 **NYET**：USB OTG HS 的收到响应中断屏蔽 (response received interrupt mask for USB OTG HS)
0：屏蔽中断
1：使能中断

位 5 **ACKM**：接收/发送 ACK 响应中断屏蔽 (ACK response received/transmitted interrupt mask)
0：屏蔽中断
1：使能中断

位 4 **NAKM**：NAK 响应接收中断屏蔽 (NAK response received interrupt mask)
0：屏蔽中断
1：使能中断

位 3 **STALLM**：STALL 响应接收中断屏蔽 (STALL response received interrupt mask)
0：屏蔽中断
1：使能中断

位 2 **AHBERR**：USB OTG HS 的 AHB 错误 (AHB error for USB OTG HS)
0：屏蔽中断
1：使能中断

位 2 保留，对于 USB OTG FS，必须保持复位值

位 1 **CHHM**：通道停止中断屏蔽 (Channel halted mask)
0：屏蔽中断
1：使能中断

位 0 **XFRCM**：传输完成中断屏蔽 (Transfer completed mask)
0：屏蔽中断
1：使能中断

37.15.30 OTG 主机通道 x 传输大小寄存器 (OTG_HCTSIZx)（x = 0..15[HS] / 11[FS]，其中 x = 通道编号）

OTG Host channel-x transfer size register

偏移地址：0x510 + (通道编号 × 0x20)

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	DPID		PKTCNT										XFRSIZ		
	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
XFRSIZ															
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 30:29 **DPID**：数据 PID (Data PID)

应用程序在此字段设置数据通信的初始同步 PID。后续传输的数据 PID 由主机硬件维护。

00：DATA0

01：DATA2

10：DATA1

11：SETUP（控制传输）/保留[FS]MDATA[HS]（非控制传输）

位 28:19 **PKTCNT**：数据包计数 (Packet count)

应用程序在此字段中设置将要发送 (OUT) 或接收 (IN) 的数据包数。

主机每成功发送或接收一个 OUT/IN 数据包便递减一次计数值。此值达到 0 后，将中断应用程序来指示操作正常完成。

位 18:0 **XFRSIZ**：传输大小 (Transfer size)

对于 OUT 操作，此字段为传输期间主机发送的数据字节数。

对于 IN 操作，此字段为应用程序保留给传输的缓冲区大小。对于 IN 事务（周期性和非周期性），应用程序会将此字段编程为最大数据包大小的整数倍。

37.15.31 OTG 主机通道 x DMA 地址寄存器 (OTG_HCDMAx)（x = 0..15，其中 x = 通道编号）

OTG Host channel-x DMA address register

偏移地址：0x514 + (通道编号 × 0x20)

复位值：0x0000 0000

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

位 31:0 **DMAADDR**：DMA 地址 (DMA address)

此字段存储主机从设备端点获取数据或往设备端点发送数据所用 DMA 传输的存储器的地址。每次进行 AHB 传输，该寄存器都会递增。

注： *配置寄存器只适用于 USB OTG HS*

37.15.32 设备模式寄存器

模块每次切换到设备模式时都必须对这些寄存器进行编程。

37.15.33 OTG 设备配置寄存器 (OTG_DCFG)

OTG device configuration register

偏移地址：0x800

复位值：0x0220 0000

此寄存器在上电、执行某些控制命令或枚举后，会将模块配置为设备模式。请勿在初始编程后更改该寄存器。

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
ERRATIM	Res.	Res.	PFIVL		DAD							Res.	NZLSOHSK	DSPD	
rw			rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw

注： *USB OTG FS 的配置寄存器*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	PERSCHIVL		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
ERRATIM	Res.	Res.	PFIVL		DAD							Res.	NZLSOHSK	DSPD	
rw			rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw

注： *USB OTG HS 的配置寄存器*

位 31:16 保留，对于 USB OTG FS，必须保持复位值

位 31:26 保留，对于 USB OTG HS，必须保持复位值

位 25:24 **PERSCHIVL**：USB OTG HS 的周期性调度间隔 (Periodic schedule interval for USB OTG HS)

此字段指定内部 DMA 引擎必须为获取周期性 IN 端点数据分配的时间。根据周期性端点数量的不同，必须将此值指定为 25%、50% 或 75% 的（微）帧。

- 只要存在活动的周期性端点，内部 DMA 引擎便会为获取周期性 IN 端点数据分配指定的时间
- 没有任何活动周期性端点时，内部 DMA 引擎将仅用于非周期性端点并忽略此字段
- 经过一（微）帧中指定的时间后，DMA 切换为获取非周期性端点上的传输数据

00：25%（微）帧
01：50%（微）帧
10：75%（微）帧
11：保留

位 23:16 保留，对于 USB OTG HS，必须保持复位值

位 15 **ERRATIM**：不定错误中断屏蔽 (Erratic error interrupt mask)

1：发送不定错误时不触发早期挂起中断
0：发生不定错误时生成早期挂起中断

位 14:13 保留，必须保持复位值。

位 12:11 **PFIVL**：周期性帧间隔 (Periodic frame interval)

指示一帧内必须使用周期性帧中断通知应用程序的时间点。此功能可用于确定该帧的所有同步通信是否完成。

00：80% 帧间隔
01：85% 帧间隔
10：90% 帧间隔
11：95% 帧间隔

位 10:4 **DAD**：设备地址 (Device Address)

应用程序必须在执行每个 SetAddress 控制命令后根据命令参数对该字段进行设置。

位 3 保留，必须保持复位值。

位 2 **NZLSOHSK**：非零长度状态 OUT 握手信号 (Non-zero-length status OUT handshake)

在控制传输状态阶段的 OUT 事务期间，当模块收到非零长度数据包后，应用程序可以使用此字段选择要发送的握手信号。

1：收到非零长度状态 OUT 事务时，回复 STALL 握手信号，收到的 OUT 数据包不发送给应用程序。

0：将收到的 OUT 数据包（零长度或非零长度）发送给应用程序，并基于设备端点控制寄存器中端点的 NAK 和 STALL 位回复握手信号。

位 1:0 **DSPD**：设备速度 (Device speed)

指示应用程序要求模块进行枚举所采用的速度，或应用程序支持的最大速度。但是，实际总线速度只有在完成 chirp 序列后才能确定，同时此速度基于与模块连接的 USB 主机的速度。

00：保留

01：保留

10：保留

11：全速（USB 1.1 收发器时钟为 48 MHz）

位 1:0 **DSPD**：设备速度 (Device speed)

指示应用程序要求模块进行枚举所采用的速度，或应用程序支持的最大速度。但是，实际总线速度只有在完成 chirp 序列后才能确定，同时此速度基于与模块连接的 USB 主机的速度。

00：高速

01：使用外部 ULPI PHY 进行全速通信

10：保留

11：使用内置 PHY 进行全速通信

37.15.34 OTG 设备控制寄存器 (OTG_DCTL)

OTG device control register

偏移地址：0x804

复位值：0x0000 0002

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	PO PRG DNE	CGO NAK	SGO NAK	CGI NAK	SGI NAK	TCTL			GON STS	GIN STS	SDIS	RWU SIG
				rw	w	w	w	w	rw	rw	rw	r	r	rw	rw

位 31:19 保留，必须保持复位值。

位 18 **DSBESLRJCT**：深度睡眠 BESL 拒绝 (Deep sleep BESL reject)

模块拒绝 BESL 值大于编程的 BESL 阈值的 LPM 请求。将为 BESL 值大于 BESL 阈值的 LPM 令牌发送 NYET 响应。默认情况下，禁用深度睡眠 BESL 拒绝功能。

位 17:12 保留，必须保持复位值。

位 11 **POPRGDNE**：上电编程完成 (Power-on programming done)

应用程序使用此位指示寄存器从掉电模式唤醒后已完成编程。

位 10 **CGONAK**：将全局 OUT NAK 清零 (Clear global OUT NAK)

对此位执行写操作会将全局 OUT NAK 清零。

位 9 **SGONAK**：将全局 OUT NAK 置 1 (Set global OUT NAK)

对此位执行写操作会将全局 OUT NAK 置 1。

应用程序使用此位在所有 OUT 端点发送 NAK 握手信号。

应用程序只有确定模块中断寄存器中的全局 OUT NAK 有效位（OTG_GINTSTS 中 GONAKEFF 位）已清零时，才可以将此位置 1。

位 8 **CGINAK**：将全局 IN NAK 清零 (Clear global IN NAK)

对此位执行写操作会将全局 IN NAK 清零。

位 7 **SGINAK**：将全局 IN NAK 置 1 (Set global IN NAK)

对此字段执行写操作会将全局非周期性 IN NAK 置 1。应用程序使用此位使所有非周期性 IN 端点发送 NAK 握手信号。

应用程序只有确定模块中断寄存器中的全局 IN NAK 有效位（OTG_GINTSTS 中的 GINAKEFF 位）已清零时，才可以将此位置 1。

位 6:4 **TCTL**：测试控制 (Test control)

000：禁止测试模式

001：Test_J 模式

010：Test_K 模式

011：Test_SE0_NAK 模式

100：Test_Packet 模式

101：Test_Force_Enable

其它值：保留

位 3 **GONSTS**：全局 OUT NAK 状态 (Global OUT NAK status)

0：将根据 FIFO 状态和 NAK 和 STALL 位设置发送握手信号。

1：无论 Rx FIFO 中是否还有空闲空间都不接收数据。除 SETUP 事务之外，对所有收到的数据包回复 NAK 握手信号。所有同步类型的 OUT 数据包都将被丢弃。

位 2 **GINSTS**：全局 IN NAK 状态 (Global IN NAK status)

0：根据发送 FIFO 中是否有待发送的数据回复握手信号。

1：使所有非周期性 IN 端点回复 NAK 握手信号，无需考虑发送 FIFO 中是否有待发送的数据。

位 1 **SDIS**：软断开 (Soft disconnect)

应用程序使用该位向 USB OTG 模块发出执行软断开的信号。该位置 1 时，主机不会看到设备已连接，且该设备也不会接收 USB 上的信号。在应用程序将此位清零之前，模块会保持断开状态。

0：正常工作。此位在软断开之后清零，会使主机收到设备已连接的事件。重新连接设备之后，USB 主机会重新启动设备枚举。

1：模块使 USB 主机收到设备断开连接的事件。

位 0 **RWUSIG**：发送远程唤醒信号 (Remote wakeup signaling)

应用程序将此位置 1 时，模块会启动远程发送信号，以唤醒 USB 主机。应用程序必须将此位置 1 以使模块退出挂起状态。根据 USB 2.0 规范，应用程序必须在将此位置 1 之后的 1 ms 到 15 ms 内将其清零。

如果 LPM 已使能并且模块处于 L1（睡眠）状态，则当应用程序将此位置 1 时，模块会启动 L1 远程信号来唤醒 USB 主机。应用程序必须将此位置 1 以使模块退出睡眠状态。按照 LPM 规范中的规定，在应用程序将此位置 1 后经过 50 µs ($T_{L1DevDrvResume}$)，硬件会自动将此位清零。当前一个 LPM 事务中的 bRemoteWake 为零时（请参见 GLPMCFG 寄存器中的 REMWAKE 位），应用程序不得将此位置 1。

表 233 显示了为使 USB 主机检测到设备断开连接所需的软断开 (SDIS) 位置 1 的最短时间（取决于设备状态）。为了协调时钟抖动，建议应用程序在指定的最小时间基础上再加入一段延迟。

表 233. 软断开的最小时间

运行速度	设备状态	最小时间
全速	挂起	1 ms + 2.5 µs
全速	空闲	2.5 µs
全速	非空闲或挂起（正在通信时）	2.5 µs
高速	非空闲或挂起（正在通信时）	125 µs

37.15.35 OTG 设备状态寄存器 (OTG_DSTS)

OTG device status register

偏移地址：0x808

复位值：0x0000 0010

此寄存器指示模块在出现 USB 相关事件时的状态。必须在发生设备全体中断寄存器 (OTG_DAINT) 中断时读取它。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FNSOF								Res.	Res.	Res.	Res.	EERR	ENUMSPD		SUSP STS
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 23:22 **DEVLNSTS**：设备线状态 (Device line status)

指示 USB 数据线的当前逻辑电平。

位 [23]：D+ 的逻辑电平

位 [22]：D- 的逻辑电平

位 21:8 **FNSOF**：接收 SOF 的帧编号 (Frame number of the received SOF)

位 7:4 保留，必须保持复位值。

位 3 **EERR**：不定错误 (Erratic error)

模块将该位置 1 以报告任何不定错误。

由于不定错误，OTG_FS/OTG_HS 控制器会进入挂起状态，并且会以 OTG_GINTSTS 寄存器的早期挂起位（OTG_GINTSTS 中的 ESUSP 位）为应用程序生成一个中断。如果早期挂起中断是由不定错误触发，则应用程序只能执行软断开以恢复通信。

位 2:1 **ENUMSPD**：枚举速度 (Enumerated speed)

指示 OTG_FS/OTG_HS 控制器通过 chirp 序列检测速度后被枚举成的速度。

01：保留

10：保留

11：全速（PHY 时钟运行频率为 48 MHz）

其它值：保留

位 0 **SUSPSTS**：挂起状态 (Suspend status)

在设备模式下，只要在 USB 上检测到挂起状态，该位就会置 1。当 USB 数据线上的空闲状态保持 3 ms，模块便会进入挂起状态。出现以下情况时，模块会退出挂起状态：

– USB 数据线上有活动

– 应用程序对 OTG_DCTL 寄存器的远程唤醒信号位（OTG_DCTL 中的 RWUSIG 位）执行写操作。

37.15.36 OTG 设备 IN 端点通用中断屏蔽寄存器 (OTG_DIEPMSK)

OTG device IN endpoint common interrupt mask register

偏移地址：0x810

复位值：0x0000 0000

此寄存器与全体端点的各个 OTG_DIEPINTx 寄存器配合使用，以便在每个 IN 端点上生成中断。通过对此寄存器的相应位执行写操作，可屏蔽 OTG_DIEPINTx 寄存器中特定状态的 IN 端点中断。默认情况下，状态中断都被屏蔽。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	NAKM	Res.	Res.	Res.	Res.	Res.	Res.	INEPNEM	INEPNMM	ITTXFEMSK	TOM	Res.	EPDM	XFRCM
		rw							rw	rw	rw	rw		rw	rw

注： *USB OTG FS 的配置寄存器*

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.	NAKM	Res.	Res.	Res.	BMA	TXFURM	Res.	INEPNEM	INEPNMM	ITTXFEMSK	TOM	Res.	EPDM	XFRCM
		rw				rw	rw		rw	rw	rw	rw		rw	rw

注： *USB OTG HS 的配置寄存器*

位 31:14 保留，必须保持复位值。

位 13 **NAKM**：NAK 中断屏蔽 (NAK interrupt mask)

0：屏蔽中断

1：使能中断

位 12:7 保留，对于 USB OTG FS，必须保持复位值。

位 12:10 保留，对于 USB OTG HS，必须保持复位值。

位 9 **BIM**：USB OTG HS 的 BNA 中断屏蔽 (BNA interrupt mask mask for USB OTG HS)
0：屏蔽中断
1：使能中断

位 8 **TXFURM**：USB OTG HS 的 FIFO 下溢屏蔽 (FIFO underrun mask for USB OTG HS)
0：屏蔽中断
1：使能中断

位 7 保留，对于 USB OTG HS，必须保持复位值。

位 6 **INEPNEM**：IN 端点 NAK 有效中断屏蔽 (IN endpoint NAK effective mask)
0：屏蔽中断
1：使能中断

位 5 **INEPNMM**：EP 不匹配时接收到 IN 令牌中断屏蔽 (IN token received with EP mismatch mask)
0：屏蔽中断
1：使能中断

位 4 **ITTXFEMSK**：Tx FIFO 为空时接收到 IN 令牌中断屏蔽 (IN token received when Tx FIFO empty mask)
0：屏蔽中断
1：使能中断

位 3 **TOM**：超时中断屏蔽（非同步端点）(Timeout condition mask (Non-isochronous endpoints))
0：屏蔽中断
1：使能中断

位 2 保留，必须保持复位值。

位 1 **EPDM**：端点禁止中断屏蔽 (Endpoint disabled interrupt mask)
0：屏蔽中断
1：使能中断

位 0 **XFRCM**：传输完成中断屏蔽 (Transfer completed interrupt mask)
0：屏蔽中断
1：使能中断

37.15.37 OTG 设备 OUT 端点通用中断屏蔽寄存器 (OTG_DOEPMSK)

OTG device OUT endpoint common interrupt mask register

偏移地址：0x814

复位值：0x0000 0000

此寄存器与所有端点的各个 OTG_DOEPINTx 寄存器配合使用，以便可以在每个 OUT 端点上生成中断。通过对此寄存器的相应位执行写操作，可屏蔽 OTG_DOEPINTx 寄存器中特定状态的 OUT 端点中断。默认情况下，状态中断都被屏蔽。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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.	OTEPDM	STUPM	Res.	EPDM	XFRCM
											rw	rw		rw	rw

注： USB OTG FS 的配置寄存器

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.	NYETMSK	Res.	Res.	Res.	Res.	BOIM	TXFURM	Res.	B2BSTUP	Res.	OTEPDM	STUPM	Res.	EPDM	XFRCM
	rw					rw	rw		rw		rw	rw		rw	rw

注： USB OTG HS 的配置寄存器

位 31:5 保留，对于 USB OTG FS，必须保持复位值。

位 31:15 保留，对于 USB OTG HS，必须保持复位值。

位 14 **NYET**：USB OTG HS 的 NYET 中断屏蔽 (NYET interrupt mask for USB OTG HS)
0：屏蔽中断
1：使能中断

位 9 **BOIM**：USB OTG HS 的 BNA 中断屏蔽 (BNA interrupt mask for USB OTG HS)
0：屏蔽中断
1：使能中断

位 8 **TXFURM**：USB OTG HS 的 FIFO 下溢屏蔽 (FIFO underrun mask for USB OTG HS)
0：屏蔽中断
1：使能中断

位 6 保留，对于 USB OTG FS，必须保持复位值。

位 7 保留，对于 USB OTG HS，必须保持复位值。

位 6 **B2BSTUP**：接收到连续的 SETUP 数据包中断屏蔽 (Back-to-back SETUP packets received mask) 仅适用于控制 OUT 端点。这适用于 USB OTG HS。
0：屏蔽中断
1：使能中断

位 4 **OTEPDM**：端点禁止时接收到 OUT 令牌中断屏蔽 (OUT token received when endpoint disabled mask) 仅适用于控制 OUT 端点。
0：屏蔽中断
1：使能中断

位 3 **STUPM**：SETUP 阶段完成中断屏蔽 (SETUP phase done mask) 仅适用于控制端点。
0：屏蔽中断
1：使能中断

位 2 保留，必须保持复位值。

位 1 **EPDM**：端点禁止中断屏蔽 (Endpoint disabled interrupt mask)
0：屏蔽中断
1：使能中断

位 0 **XFRCM**：传输完成中断屏蔽 (Transfer completed interrupt mask)
0：屏蔽中断
1：使能中断

37.15.38 OTG 设备全体端点中断寄存器 (OTG_HAINT)

OTG device all endpoints interrupt register

偏移地址：0x818

复位值：0x0000 0000

当端点上发生有效事件时，OTG_DAINT 寄存器将通过 OTG_GINTSTS 寄存器中的设备 OUT 端点中断位或设备 IN 端点中断位（分别为 OTG_GINTSTS 中的 OEPINT 或 IEPINT 位）来中断应用程序。每个端点对应一个中断位，OUT 端点和 IN 端点均最多有 16 个中断位。双向端点将使用相应的 IN 和 OUT 中断位。当应用程序将相应设备端点 x 中断寄存器 (OTG_DIEPINTx/OTG_DOEPINTx) 中的位置 1 和清零时，此寄存器中的相应位也将置 1 和清零。

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

位 31:16 **OEPINT**：OUT 端点中断位 (OUT endpoint interrupt bits)
每个 OUT 端点对应一位：
OUT 端点 0 对应位 16，而 OUT 端点 3 对应位 18。

位 15:0 **IEPINT**：IN 端点中断位 (IN endpoint interrupt bits)
每个 IN 端点对应一位：
IN 端点 0 对应位 0，而 IN 端点 3 对应位 3。

37.15.39 OTG 全体端点中断屏蔽寄存器 (OTG_DAINTMSK)

OTG all endpoints interrupt mask register

偏移地址：0x81C

复位值：0x0000 0000

OTG_DAINTMSK 寄存器与设备端点中断寄存器结合使用，在设备端点上发生事件时中断应用程序。但是，与该中断相对应的 OTG_DAINT 寄存器位仍会置 1。

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

位 31:16 **OEPM**：OUT EP 中断屏蔽位 (OUT EP interrupt mask bits)
每个 OUT 端点对应一位：
OUT EP 0 对应位 16，而 OUT EP 3 对应位 18
0：屏蔽中断
1：使能中断

位 15:0 **IEPM**：IN EP 中断屏蔽位 (IN EP interrupt mask bits)
每个 IN 端点对应一位：
IN EP 0 对应位 0，而 IN EP 3 对应位 3
0：屏蔽中断
1：使能中断

37.15.40 OTG 设备 V_{BUS} 放电时间寄存器 (OTG_DVBUSDIS)

OTG device V_{BUS} discharge time register

偏移地址：0x0828

复位值：0x0000 17D7

该寄存器指定 SRP 期间 V_{BUS} 发出脉冲之后的 V_{BUS} 放电时间。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBUSDT															
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 **VBUSDT**：设备 V_{BUS} 放电时间 (Device V_{BUS} discharge time)
指定 SRP 期间 V_{BUS} 发出脉冲之后的 V_{BUS} 放电时间。该时间值等于：
V_{BUS} 放电时间（PHY 时钟数）/1024
该值可基于不同的 V_{BUS} 负载根据需要进行调整。

37.15.41 OTG 设备 V_{BUS} 脉冲时间寄存器 (OTG_DVBUSPULSE)

OTG device V_{BUS} pulsing time register

偏移地址：0x082C

复位值：0x0000 05B8

该寄存器指定 SRP 期间的 V_{BUS} 脉冲时间。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVBUSP															
rw	rw	rw	rw	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 **DVBUSP**：设备 V_{BUS} 脉冲时间 (Device V_{BUS} pulsing time)
指定 SRP 期间的 V_{BUS} 脉冲时间。
V_{BUS} 脉冲时间（PHY 时钟数）/1024

37.15.42 OTG 设备阈值控制寄存器 (OTG_DTHRCTL)

OTG Device threshold control register

偏移地址：0x0830

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	ARPEN	Res.	RXTHRLEN									RXTH REN
				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.	TXTHRLEN									ISOT HREN	NONIS OTH REN
rw	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 **ARPEN**：仲裁器驻留使能 (Arbiter parking enable)
该位用于控制内部 DMA 仲裁对 IN 端点的驻留。当阈值使能且该位置 1 后，DMA 会对在 USB 上收到令牌的 IN 端点驻留其仲裁。采用这种方式可以避免出现下溢情况。默认情况下使能驻留。

位 26 保留，必须保持复位值。

位 25:17 **RXTHRLEN**：接收阈值长度 (Receive threshold length)
该字段以双字为单位指定接收阈值的大小。该字段还指定模块在 AHB 上启动传输之前在 USB 上接收的数据量。阈值长度最小值为八个双字。推荐 RXTHRLEN 的值和设定的 AHB 批量传输长度（OTG_GAHBCFG 中的 HBSTLEN 位）相同。

位 16 **RXTHREN**：接收阈值使能 (Receive threshold enable)
该位置 1 时，模块会使能接收方向的阈值。

位 15:11 保留，必须保持复位值。

位 10:2 **TXTHRLEN**：发送阈值长度 (Transmit threshold length)
该字段以双字为单位指定发送阈值的大小。该字段指定模块在 USB 上启动传输之前相应端点发送 FIFO 中的数据量（单位为字节）。阈值长度最小值为八个双字。该字段控制同步和非同步 IN 端点阈值。推荐 TXTHRLEN 的值和设定的 AHB 批量传输长度（OTG_GAHBCFG 中的 HBSTLEN 位）相同。

位 1 **ISOTHREN**：ISO IN 端点阈值使能 (ISO IN endpoint threshold enable)
该位置 1 时，模块会使能同步 IN 端点的阈值。

位 0 **NONISOTHREN**：非同步 IN 端点阈值使能 (Nonisochronous IN endpoints enable)
该位置 1 时，模块会使能非同步 IN 端点的阈值。

注： *配置寄存器只适用于 USB OTG HS*

37.15.43 OTG 设备单个端点中断寄存器 (OTG_DEACHINT)

OTG device each endpoint interrupt register

偏移地址：0x0838

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OEP1 INT	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.	IEP1 INT	Res.
														r	

位 31:18 保留，必须保持复位值。

位 17 **OEP1INT**：OUT 端点 1 中断位 (OUT endpoint 1 interrupt bit)

位 16:2 保留，必须保持复位值。

位 1 **IEP1INT**：IN 端点 1 中断位 (IN endpoint 1interrupt bit)

位 0 保留，必须保持复位值。

注： *配置寄存器只适用于 USB OTG HS*

37.15.44 OTG 设备 IN 端点 FIFO 空中断屏蔽寄存器 (OTG_DIEPEMPMSK)

OTG device IN endpoint FIFO empty interrupt mask register

偏移地址：0x834

复位值：0x0000 0000

此寄存器用于控制 IN 端点 FIFO 空中断的生成 (TXFE_OTG_DIEPINTx)。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INEPTXFEM															
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 **INEPTXFEM**：IN EP Tx FIFO 空中断屏蔽位 (IN EP Tx FIFO empty interrupt mask bits)
这些位用作 OTG_DIEPINTx 的屏蔽位。
每个位对应一个 IN 端点的 TXFE 中断：
IN 端点 0 对应位 0，而 IN 端点 3 对应位 3
0：屏蔽中断
1：使能中断

37.15.45 OTG 设备单个端点中断屏蔽寄存器 (OTG_DEACHINTMSK)

OTG device each endpoint interrupt register mask

偏移地址：0x083C

复位值：0x0000 0000

IN 端点 1 对应一个中断位，OUT 端点 1 对应一个中断位。

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.	OEP1 INTM	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.	IEP1I NTM	Res.
														rw	

位 31:18 保留，必须保持复位值。

位 17 **OEP1INTM**：OUT 端点 1 中断屏蔽位 (OUT Endpoint 1 interrupt mask bit)

位 16:2 保留，必须保持复位值。

位 1 **IEP1INTM**：IN 端点 1 中断屏蔽位 (IN Endpoint 1 interrupt mask bit)

位 0 保留，必须保持复位值。

注： *配置寄存器只适用于 USB OTG HS*

37.15.46 OTG 设备控制 IN 端点 0 控制寄存器 (OTG_DIEPCTL0)

OTG device control IN endpoint 0 control register

偏移地址：0x900

复位值：0x0000 0000

本节介绍 USB_OTG FS 的 OTG_DIEPCTL0 寄存器。非零控制端点使用端点 1-3 的寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPENA	EPDIS	Res.	Res.	SNAK	CNAK	TXFNUM				STALL	Res.	EPTYP		NAK STS	Res.
rs	rs			w	w	rw	rw	rw	rw	rs		r	r	r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USBA EP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MPSIZ	
r														rw	rw

位 31 **EPENA**：端点使能 (Endpoint enable)

应用程序将此位置 1 以在端点 0 上启动数据发送。

在此端点上触发以下任一中断之前，模块会将此位清零：

– 端点禁止

– 传输完成

位 30 **EPDIS**：端点禁止 (Endpoint disable)

即使在该端点上的传输完成之前，应用程序也可将此位置 1，以停止端点上的数据发送。应用程序必须等到发生端点禁止中断后，才能将端点视为禁止端点。在端点禁止中断位置 1 前，模块会将此位清零。只有在该端点的端点使能位置 1 后，应用程序才可将该位置 1。

位 29:28 保留，必须保持复位值。

位 27 **SNAK**：将 NAK 置 1 (Set NAK)

对此位进行写操作会将端点的 NAK 位置 1。

通过此位，应用程序可以控制端点上 NAK 握手信号的发送。模块也可在端点接收到 SETUP 数据包后将该端点的此位置 1。

位 26 **CNAK**：将 NAK 清零 (Clear NAK)

对此位进行写操作会将端点的 NAK 位清零。

位 25:22 **TXFNUM**：Tx FIFO 编号 (Tx FIFO number)

该值设置为分配给 IN 端点 0 的 FIFO 编号。

位 21 **STALL**：STALL 握手 (STALL handshake)

应用程序只能将此位置 1，端点接收到 SETUP 令牌时，模块会将此位清零。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位均置 1，则 STALL 位优先。

位 20 保留，必须保持复位值。

位 19:18 **EPTYP**：端点类型 (Endpoint type)

硬件设置为 '00'，表示控制类型的端点。

位 17 **NAKSTS**：NAK 状态 (NAK status)

指示以下结果：

0：模块根据 FIFO 状态回复非 NAK 握手。

1：模块在此端点上回复 NAK 握手。

当应用程序或模块将此位置 1 时，即使 Tx FIFO 中仍有数据可用，模块也会停止发送数据。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。

位 16 保留，必须保持复位值。

位 15 **USBAEP**：USB 活动端点 (USB active endpoint)
此位总是置 1，指示在所有配置和接口中控制端点 0 始终处于激活状态。

位 14:2 保留，必须保持复位值。

位 1:0 **MPSIZ**：最大数据包大小 (Maximum packet size)
应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。
00：64 字节
01：32 字节
10：16 字节
11：8 字节

注： *配置寄存器只适用于 USB OTG FS*

37.15.47 OTG 设备端点 x 控制寄存器 (OTG_DIEPCTLx)（x = 1..5[FS] / 0..7[HS]，其中 x = 端点编号）

OTG device endpoint-x control register

偏移地址：0x900 + (端点编号 × 0x20)

复位值：0x0000 0000

应用程序使用此寄存器控制各个逻辑端点（端点 0 除外）的行为。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPENA	EPDIS	SODD FRM	SD0 PID/ SEVN FRM	SNAK	CNAK	TXFNUM				STALL	Res.	EPTYP		NAK STS	EO NUM/ DPID
rs	rs	w	w	w	w	rw	rw	rw	rw	rw/rs		rw	rw	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USBA EP	Res.	Res.	Res.	Res.	MPSIZ										
rw					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **EPENA**：端点使能 (Endpoint enable)
应用程序将此位置 1 以在端点上启动数据发送。
在此端点上触发以下任一中断之前，模块会将此位清零：
– SETUP 阶段完成 (SETUP phase done)
– 端点禁止
– 传输完成

位 30 **EPDIS**：端点禁止 (Endpoint disable)
即使在该端点上的传送完成之前，应用程序也可将此位置 1，以停止端点上的数据发送/接收。应用程序必须等到发生端点禁止中断后，才能将端点视为禁止端点。在端点禁止中断位置 1 前，模块会将此位清零。只有在该端点的端点使能位置 1 后，应用程序才可将该位置 1。

位 29 **SODDFRM**：设置奇数帧 (Set odd frame)
仅适用于同步 IN 和 OUT 端点。
对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为奇数帧。

位 28 **SD0PID**：设置 DATA0 PID (Set DATA0 PID)

仅适用于中断/批量 IN 端点。

对此字段进行写操作会将此寄存器中的端点数据 PID (DPID) 字段设置为 DATA0。

SEVNFRM：设置偶数帧 (Set even frame)

仅适用于同步 IN 端点。

对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为偶数帧。

位 27 **SNAK**：将 NAK 置 1 (Set NAK)

对此位进行写操作会将端点的 NAK 位置 1。

通过此位，应用程序可以控制端点上 NAK 握手信号的发送。发生传输完成中断时或端点上接收到 SETUP 后，模块也可以将 OUT 端点的这个位置 1。

位 26 **CNAK**：将 NAK 清零 (Clear NAK)

对此位进行写操作会将端点的 NAK 位清零。

位 25:22 **TXFNUM**：Tx FIFO 编号 (Tx FIFO number)

这些位用于指定与此端点相关联的 FIFO 编号。必须为每个有效的 IN 端点设置单独的 FIFO 编号。

此字段仅针对 IN 端点有效。

位 21 **STALL**：STALL 握手 (STALL handshake)

仅适用于非控制、非同步 IN 端点（访问类型为 rw）。

应用程序将此位置 1 使得设备对来自 USB 主机的所有令牌都回复 STALL。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1，则 STALL 位优先。只有应用程序能够将此位清零，而模块则不能。

仅适用于控制端点（访问类型为 rs）

此端点接收到 SETUP 令牌时，应用程序只能将此位置 1，而模块会将其清零。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1，则 STALL 位优先。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。

位 20 保留，必须保持复位值。

位 19:18 **EPTYP**：端点类型 (Endpoint type)

以下是这个逻辑端点支持的传输类型。

00：控制

01：同步

10：批量

11：中断

位 17 **NAKSTS**：NAK 状态 (NAK status)

它指示以下结果：

0：模块根据 FIFO 状态回复非 NAK 握手。

1：模块在此端点上回复 NAK 握手。

当应用程序或模块将此位置 1 时：

对于非同步 IN 端点：即使 Tx FIFO 中存在可用数据，模块也会停止通过 IN 端点发送任何数据。

对于同步 IN 端点：即使 Tx FIFO 中存在可用数据，模块也会发送长度为零的数据包。

无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。

位 16 **EONUM**：偶数/奇数帧 (Even/odd frame)

仅适用于同步 IN 端点。

指示模块为此端点发送/接收同步的数据所在的帧的编号。应用程序必须通过此寄存器中的 SEVNFRM 和 SODDFRM 字段对偶数/奇数帧编号进行编程，以便此端点发送/接收同步数据。

0：偶数帧

1：奇数帧

DPID：端点数据 PID (Endpoint data PID)

仅适用于中断/批量 IN 端点。

包含此端点上将要接收或发送的数据包的 PID。端点激活后，应用程序必须对要在此端点上接收或发送的首个数据包的 PID 进行编程。应用程序使用 SD0PID 寄存器字段对 DATA0 或 DATA1 PID 进行编程。

0：DATA0

1：DATA1

位 15 **USBAEP**：USB 活动端点 (USB active endpoint)

指示此端点在当前配置和接口中是否激活。检测到 USB 复位后，模块会为所有端点（端点 0 除外）将此位清零。接收到 SetConfiguration 和 SetInterface 命令后，应用程序必须相应地对端点寄存器进行编程并将此位置 1。

位 14:11 保留，必须保持复位值。

位 10:0 **MPSIZ**：最大数据包大小 (Maximum packet size)

应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。此值以字节为单位。

37.15.48 OTG 设备控制 OUT 端点 0 控制寄存器 (OTG_DOEPCTL0)

OTG device control OUT endpoint 0 control register

偏移地址：0xB00

复位值：0x0000 8000

本节介绍 OTG_DOEPCTL0 寄存器。非零控制端点使用编号 1-3 的寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPENA	EPDIS	Res.	Res.	SNAK	CNAK	Res.	Res.	Res.	Res.	STALL	SNPM	EPTYP		NAK STS	Res.
w	r			w	w					rs	rw	r	r	r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USBA EP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MPSIZ	
r														r	r

位 31 **EPENA**：端点使能 (Endpoint enable)

应用程序将此位置 1 以在端点 0 上启动数据发送。

在此端点上触发以下任一中断之前，模块会将此位清零：

– SETUP 阶段完成 (SETUP phase done)

– 端点禁止

– 传输完成

位 30 **EPDIS**：端点禁止 (Endpoint disable)

应用程序无法禁止控制 OUT 端点 0。

位 29:28 保留，必须保持复位值。

位 27 **SNAK**：将 NAK 置 1 (Set NAK)

对此位进行写操作会将端点的 NAK 位置 1。

通过此位，应用程序可以控制端点上 NAK 握手信号的发送。发生传输完成中断时或端点上接收到 SETUP 后，模块也可以将此位置 1。

位 26 **CNAK**：将 NAK 清零 (Clear NAK)

对此位进行写操作会将端点的 NAK 位清零。

位 25:22 保留，必须保持复位值。

位 21 **STALL**：STALL 握手 (STALL handshake)

此端点接收到 SETUP 令牌时，应用程序只能将此位置 1，而模块会将其清零。如果 NAK 位、全局 OUT NAK 与此位同时置 1，则 STALL 位优先。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。

位 20 **SNPM**：监听模式 (Snoop mode)

此位用于将端点配置为监听模式。在监听模式下，模块不会在将 OUT 数据包传输到应用存储区前检查其是否正确。

位 19:18 **EPTYP**：端点类型 (Endpoint type)

硬件固定为二进制 00，表示端点为控制传输类型。

位 17 **NAKSTS**：NAK 状态 (NAK status)

指示以下结果：

0：模块根据 FIFO 状态回复非 NAK 握手。

1：模块在此端点上回复 NAK 握手。

当应用程序或模块将此位置 1 时，即使 Rx FIFO 中存在空间可继续容纳收到的数据包，模块也会停止接收数据。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。

位 16 保留，必须保持复位值。

位 15 **USBAEP**：USB 活动端点 (USB active endpoint)

此位总是置 1，指示在所有配置和接口中控制端点 0 始终处于激活状态。

位 14:2 保留，必须保持复位值。

位 1:0 **MPSIZ**：最大数据包大小 (Maximum packet size)

控制 OUT 端点 0 的最大数据包大小与在控制 IN 端点 0 中进行编程的值相同。

00：64 字节

01：32 字节

10：16 字节

11：8 字节

37.15.49 OTG 设备端点 x 控制寄存器 (OTG_DOEPCTLx)（x = 1..5[FS] / 7[HS]，其中 x = 端点编号）

OTG device endpoint-x control register

OUT 端点的偏移地址：0xB00 + (端点编号 × 0x20)

复位值：0x0000 0000

应用程序使用此寄存器控制各个逻辑端点（端点 0 除外）的行为。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPENA	EPDIS	SD1 PID/ SODD FRM	SD0 PID/ SEVN FRM	SNAK	CNAK	Res.	Res.	Res.	Res.	STALL	SNPM	EPTYP		NAK STS	EO NUM/ DPID
rs	rs	w	w	w	w					rw/rs	rw	rw	rw	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USBA EP	Res.	Res.	Res.	Res.	MPSIZ										
rw					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **EPENA**：端点使能 (Endpoint enable)

适用于 IN 和 OUT 端点。

应用程序将此位置 1 以在端点上启动数据发送。

在此端点上触发以下任一中断之前，模块会将此位清零：

– SETUP 阶段完成 (SETUP phase done)

– 端点禁止

– 传输完成

位 30 **EPDIS**：端点禁止 (Endpoint disable)

即使在该端点上的传送完成之前，应用程序也可将此位置 1，以停止端点上的数据发送/接收。应用程序必须等到发生端点禁止中断后，才能将端点视为禁止端点。在端点禁止中断位置 1 前，模块会将此位清零。只有在该端点的端点使能位置 1 后，应用程序才可将该位置 1。

位 29 **SD1PID**：设置 DATA1 PID (Set DATA1 PID)

仅适用于中断/批量 IN 和 OUT 端点。对此字段进行写操作会将此寄存器中的端点数据 PID (DPID) 字段设置为 DATA1。

SODDFRM：设置奇数帧 (Set odd frame)

仅适用于同步 IN 和 OUT 端点。对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为奇数帧。

位 28 **SD0PID**：设置 DATA0 PID (Set DATA0 PID)

仅适用于中断/批量 OUT 端点。

对此字段进行写操作会将此寄存器中的端点数据 PID (DPID) 字段设置为 DATA0。

SEVNFRM：设置偶数帧 (Set even frame)

仅适用于同步 OUT 端点。

对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为偶数帧。

位 27 **SNAK**：将 NAK 置 1 (Set NAK)

对此位进行写操作会将端点的 NAK 位置 1。

通过此位，应用程序可以控制端点上 NAK 握手信号的发送。发生传输完成中断时或端点上接收到 SETUP 后，模块也可以将 OUT 端点的这个位置 1。

位 26 **CNAK**：将 NAK 清零 (Clear NAK)

对此位进行写操作会将端点的 NAK 位清零。

位 25:22 保留，必须保持复位值。

位 21 **STALL**：STALL 握手 (STALL handshake)

仅适用于非控制、非同步 OUT 端点（访问类型为 rw）。

应用程序将此位置 1 使得设备对来自 USB 主机的所有令牌都回复 STALL。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1，则 STALL 位优先。只有应用程序能够将此位清零，而模块则不能。

仅适用于控制端点（访问类型为 rs）

此端点接收到 SETUP 令牌时，应用程序只能将此位置 1，而模块会将其清零。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1，则 STALL 位优先。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。

位 20 **SNPM**：监听模式 (Snoop mode)

此位用于将端点配置为监听模式。在监听模式下，模块不会在将 OUT 数据包传输到应用存储区前检查其是否正确。

位 19:18 **EPTYP**：端点类型 (Endpoint type)

以下是这个逻辑端点支持的传输类型。

00：控制

01：同步

10：批量

11：中断

位 17 **NAKSTS**：NAK 状态 (NAK status)

指示以下结果：

0：模块根据 FIFO 状态回复非 NAK 握手。

1：模块在此端点上回复 NAK 握手。

当应用程序或模块将此位置 1 时：

即使 Rx FIFO 存在空间可容纳传入数据包，模块也会停止在 OUT 端点上接收任何数据。

无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。

位 16 **EONUM**：偶数/奇数帧 (Even/odd frame)

仅适用于同步 IN 和 OUT 端点。

指示模块为此端点发送/接收同步的数据所在的帧的编号。应用程序必须通过此寄存器中的 SEVNFRM 和 SODDFRM 字段对偶数/奇数帧编号进行编程，以便此端点发送/接收同步数据。

0：偶数帧

1：奇数帧

DPID：端点数据 PID (Endpoint data PID)

仅适用于中断/批量 OUT 端点。

包含此端点上将要接收或发送的数据包的 PID。端点激活后，应用程序必须对要在此端点上接收或发送的首个数据包的 PID 进行编程。应用程序使用 SD0PID 寄存器字段对 DATA0 或 DATA1 PID 进行编程。

0：DATA0

1：DATA1

位 15 **USBAEP**：USB 活动端点 (USB active endpoint)

指示此端点在当前配置和接口中是否激活。检测到 USB 复位后，模块会为所有端点（端点 0 除外）将此位清零。接收到 SetConfiguration 和 SetInterface 命令后，应用程序必须相应地对端点寄存器进行编程并将此位置 1。

位 14:11 保留，必须保持复位值。

位 10:0 **MPSIZ**：最大数据包大小 (Maximum packet size)
应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。此值以字节为单位。

37.15.50 OTG 设备端点 x 中断寄存器 (OTG_DIEPINTx)（x = 0..5[FS] / 7[HS]，其中 x = 端点编号）

OTG device endpoint-x interrupt register

偏移地址：0x908 + (端点编号 × 0x20)

复位值：0x0000 0080

此寄存器指示端点在出现 USB 和 AHB 相关事件时的状态。相关内容在*图 446* 中进行了说明。当模块中断寄存器中的 IN 端点中断位（OTG_GINTSTS 中的 IEPINT 位）置 1 时，应用程序必须读取此寄存器。在应用程序能够读取此寄存器之前，必须先读取设备全体端点中该寄存器中的相应位清零，才能将 OTG_DAINT 和 OTG_GINTSTS 寄存器中的对应位清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFE	INEPNE	Res.	ITTXFE	TOC	Res.	EPDISD	XFRC
								r	rc_w1/rw		rc_w1	rc_w1		rc_w1	rc_w1

注： *USB OTG FS 的配置寄存器*

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.	NAK	BERR	PKTDRPSTS	Res.	BNA	TXFIFOUDRN	TXFE	INEPNE	Res.	ITTXFE	TOC	Res.	EPDISD	XFRC
		rc_w1	rc_w1	rc_w1		rc_w1	rc_w1	r	rc_w1/rw		rc_w1	rc_w1		rc_w1	rc_w1

注： *USB OTG HS 的配置寄存器*

位 31:8 保留，必须保持复位值。

位 31:14 保留，对于 USB OTG HS，必须保持复位值

位 13 **NAK**：USB OTG HS 的 NAK 输入 (NAK input for USB OTG HS)
当设备发出或收到 NAK 时，模块将生成该中断。
如果是同步 IN 端点，由于 Tx FIFO 中无数据可发而发送长度为零的数据包时也会生成该中断。

位 12 **BERR**：USB OTG HS 的串扰错误中断 (Babble error interrupt for USB OTG HS)

位 11 **PKTDRPSTS**：USB OTG HS 的数据包丢弃状态 (Packet dropped status for USB OTG HS)
该位用于向应用程序指示有 ISOC OUT 数据包被丢弃。该位没有相应的中断屏蔽位，也不会生成中断。

位 10 保留，对于 USB OTG HS，必须保持复位值。

位 9 **BNA**：USB OTG HS 的缓冲区不可用中断 (Buffer not available interrupt for USB OTG HS)
当所访问的描述符没有准备好供模块处理时（例如主机繁忙或 DMA 已完成）， 模块将生成该中断。

位 8 **TXFIFOUDRN**：USB OTG HS 的发送 FIFO 下溢 (TxfifoUndrn) (Transmit Fifo Underrun (TxfifoUndrn) for USB OTG HS)
当模块检测到该端点的发送 FIFO 下溢时，将生成该中断。相关性：该中断仅在使能了阈值时有效

位 7 **TXFE**：发送 FIFO 为空 (Transmit FIFO empty)
当此端点的 Tx FIFO 为半空或全空时，此中断被置位。Tx FIFO 为半空还是全空状态由 OTG_GAHBCFG 寄存器中的 Tx FIFO 空门限位（OTG_GAHBCFG 中的 TXFELVL 位）决定。

位 6 **INEPNE**：IN 端点 NAK 有效 (IN endpoint NAK effective)
当应用程序通过向 OTG_DIEPCTLx 中的 CNAK 位写入数据来将 IN 端点 NAK 清零时，此位可被清零。
该中断指示模块已对（由应用程序或模块）置 1 的 NAK 采样，结果已生效。该中断指示由应用程序置 1 的 IN 端点 NAK 位已在模块中起作用。
此中断不一定表示 USB 上已发送了一个 NAK 握手信号。STALL 位优先级高于 NAK 位。

位 5 保留，必须保持复位值。

位 4 **ITTXFE**：Tx FIFO 为空时接收到 IN 令牌 (IN token received when Tx FIFO is empty)
仅适用于非周期性 IN 端点。
当和该端点相对应的 Tx FIFO（周期性/非周期性）为空时，接收到 IN 令牌， 从而产生中断。

位 3 **TOC**：超时条件 (Timeout condition)
仅适用于控制 IN 端点。
指示该端点对最近收到的 IN 令牌响应超时。

位 2 保留，必须保持复位值。

位 1 **EPDISD**：端点禁止中断 (Endpoint disabled interrupt)
此位指示该端点已经由应用程序禁止掉。

位 0 **XFRC**：传输完成中断 (Transfer completed interrupt)
此字段指示在此端点上设置的传输已经在 USB 和 AHB 上传输完成。

37.15.51 OTG 设备端点 x 中断寄存器 (OTG_DIEPINTx)（x = 0..5[FS] / 7[HS]，其中 x = 端点编号）

OTG device endpoint-x interrupt register

偏移地址：0x908 + (端点编号 × 0x20)

复位值：0x0000 0080

此寄存器指示端点在出现 USB 和 AHB 相关事件时的状态。相关内容在*图 446* 中进行了说明。当 OTG_GINTSTS 寄存器中的 OUT 端点中断位（OTG_GINTSTS 中的 OEPINT 位）置 1 时，应用程序必须读取此寄存器。在应用程序能够读取此寄存器之前，必须先读取 OTG_DAINT 寄存器，以获取 OTG_DOEPINTx 寄存器的准确端点编号。应用程序必须将该寄存器中的相应位清零，才能将 OTG_DAINT 和 OTG_GINTSTS 寄存器中的对应位清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	B2B STUP	Res.	OTEP DIS	STUP	Res.	EP DISD	XFRC
									rc_w1/ rw		rc_w1	rc_w1		rc_w1	rc_w1

位 31:7 保留，必须保持复位值。

位 6 **B2BSTUP**：接收到连续的 SETUP 数据包 (Back-to-back SETUP packets received)
仅适用于控制 OUT 端点。
此位指示该端点已接收到三个以上的连续 SETUP 数据包。

位 5 保留，必须保持复位值。

位 4 **OTEPDIS**：端点禁止时接收到 OUT 令牌 (OUT token received when endpoint disabled)
仅适用于控制 OUT 端点。
指示在尚未使能端点时接收到 OUT 令牌， 从而产生中断。

位 3 **STUP**：SETUP 阶段完成 (SETUP phase done)
仅适用于控制 OUT 端点。
指示控制端点的 SETUP 阶段已完成，当前控制传输中不再接收到连续的 SETUP 数据包。在此中断上，应用程序可以对接收到的 SETUP 数据包进行解码。

位 2 保留，必须保持复位值。

位 1 **EPDISD**：端点禁止中断 (Endpoint disabled interrupt)
此位指示该端点已经由应用程序禁止掉。

位 0 **XFRC**：传输完成中断 (Transfer completed interrupt)
此字段指示在此端点上设置的传输已经在 USB 和 AHB 上传输完成。

37.15.52 OTG 设备 IN 端点 0 传输大小寄存器 (OTG_DIEPTSIZ0)

OTG device IN endpoint 0 transfer size register

偏移地址：0x910

复位值：0x0000 0000

在使能端点 0 之前，应用程序必须修改此寄存器。通过设备控制端点 0 控制寄存器中的端点使能位（OTG_DIEPCTL0 中的 EPENA）使能端点 0 后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

非零端点使用端点 1-3 的寄存器。

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.	PKTCNT		Res.	Res.	Res.
											rw	rw			

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

位 31:21 保留，必须保持复位值。

位 20:19 **PKTCNT**：数据包计数 (Packet count)

指示端点 0 的一次数据传输包含的数据包个数。

每次从 Tx FIFO 读取数据包（最大大小数据包或短数据包）时，此字段将递减。

位 18:7 保留，必须保持复位值。

位 6:0 **XFRSIZ**：传输大小 (Transfer size)

指示端点 0 的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。

每次向 Tx FIFO 写入来自外部存储器的数据包时，模块会使此字段递减。

37.15.53 OTG 设备 OUT 端点 0 传输大小寄存器 (OTG_DOEPTSIZ0)

OTG device OUT endpoint 0 transfer size register

偏移地址：0xB10

复位值：0x0000 0000

在使能端点 0 之前，应用程序必须修改此寄存器。通过 OTG_DOEPCTL0 寄存器中的端点使能位（OTG_DOEPCTL0 中的 EPENA 位）使能端点 0 后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

非零端点使用端点 1–5[FS] / 7[HS] 的寄存器。

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

位 31 保留，必须保持复位值。

位 30:29 **STUPCNT**：SETUP 数据包计数 (SETUP packet count)

此字段指定端点能连续接收的 SETUP 数据包数量。

01：1 个数据包

10：2 个数据包

11：3 个数据包

位 28:20 保留，必须保持复位值。

位 19 **PKTCNT**：数据包计数 (Packet count)

每向 Rx FIFO 写入一个数据包，此字段递减。

位 18:7 保留，必须保持复位值。

位 6:0 **XFRSIZ**：传输大小 (Transfer size)

指示端点 0 的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。

每次从 Rx FIFO 读取数据包并将其写入外部存储器时，模块会使此字段递减。

37.15.54 OTG 设备 IN 端点 x 传输大小寄存器 (OTG_DIEPTSIZx)（x = 1..5[FS] / 7[HS]，其中 x= 端点编号）

OTG device IN endpoint-x transfer size register

偏移地址：0x910 + (端点编号 × 0x20)

复位值：0x0000 0000

在使能该端点之前，应用程序必须修改此寄存器。通过 OTG_DIEPCTLx 寄存器中的端点使能位（OTG_DIEPCTLx 中的 EPENA 位）使能该端点后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	MCNT		PKTCNT										XFRSIZ		
	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
XFRSIZ															
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 30:29 **MCNT**：多重计数 (Multi count)

对于周期性 IN 端点，此字段指示在 USB 上每帧必须发送的数据包数。模块使用此字段计算同步 IN 端点的数据 PID。

01：1 个数据包

10：2 个数据包

11：3 个数据包

位 28:19 **PKTCNT**：数据包计数 (Packet count)

指示该端点上的一次数据传输包含的数据包个数。

每次从 Tx FIFO 读取数据包（最大大小数据包或短数据包）时，此字段将递减。

位 18:0 **XFRSIZ**：传输大小 (Transfer size)

此字段包含当前端点的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。

每次向 Tx FIFO 写入来自外部存储器的数据包时，模块会使此字段递减。

37.15.55 OTG 设备 IN 端点发送 FIFO 状态寄存器 (OTG_DTXFSTSx)（x = 0..5[FS] / 7[HS]，其中 x = 端点编号）

OTG device IN endpoint transmit FIFO status register

IN 端点的偏移地址：0x918 + (端点编号 × 0x20) 此只读寄存器包含设备 IN 端点 Tx FIFO 的空闲空间信息。

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

31:16 保留，必须保持复位值。

15:0 **INEPTFSAV**：IN 端点 Tx FIFO 可用空间 (IN endpoint Tx FIFO space available)
指示端点 Tx FIFO 中的可用空闲空间大小。
以 32 位字为单位：
0x0：端点 Tx FIFO 已满
0x1：1 个字可用
0x2：2 个字可用
0xn：n 个字可用
其它值：保留

37.15.56 OTG 设备 OUT 端点 x 传输大小寄存器 (OTG_DOEPTSIZx)（x = 1..5[FS] / 7[HS]，其中 x = 端点编号）

OTG device OUT endpoint-x transfer size register

偏移地址：0xB10 + (端点编号 × 0x20)

复位值：0x0000 0000

在使能该端点之前，应用程序必须修改此寄存器。通过 OTG_DOEPCTLx 寄存器中的端点使能位（OTG_DOEPCTLx 中的 EPENA 位）使能该端点后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	RXDPID/ STUPCNT		PKTCNT										XFRSIZ		
	r/rw	r/rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XFRSIZ															
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 30:29 **RXDPID**：接收到的数据 PID (Received data PID)
仅适用于同步 OUT 端点。
这是此端点收到的上一个数据包的 PID。
00：DATA0
01：DATA2
10：DATA1
11：MDATA

STUPCNT：SETUP 数据包计数 (SETUP packet count)
仅适用于控制 OUT 端点。
此字段指定端点能连续接收的 SETUP 数据包数量。
01：1 个数据包
10：2 个数据包
11：3 个数据包

位 28:19 **PKTCNT**：数据包计数 (Packet count)
指示该端点上的一次数据传输包含的数据包个数。
每次向 Rx FIFO 写入数据包（最大大小数据包或短数据包）后，此字段将递减。

位 18:0 **XFRSIZ**：传输大小 (Transfer size)
此字段包含当前端点的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。
每次从 Rx FIFO 读取数据包并将其写入外部存储器时，模块会使此字段递减。

37.15.57 OTG 电源和时钟门控控制寄存器 (OTG_PCGCCTL)

OTG power and clock gating control register

偏移地址：0xE00

复位值：0x0000 0000

此寄存器在主机模式和设备模式下均可用。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUSP	PHY SLEEP	ENL1 GTG	PHY SUSP	Res.	Res.	GATE HCLK	STPP CLK
								r	r	r/w	r			rw	rw

位 31:8 保留，必须保持复位值。

位 7 **SUSP**：深度睡眠 (Deep Sleep)
在 L1 状态下时，此位表示 PHY 处于深度睡眠状态。

位 6 **PHYSLEEP**：PHY 处于睡眠状态 (PHY in Sleep)
此位指示 PHY 处于睡眠状态。

位 5 **ENL1GTG**：使能睡眠时钟门控 (Enable Sleep clock gating)

此位置 1 时，如果模块无法触发 utmi_l1_suspend_n，则会在睡眠状态下使能模块内部时钟门控。当此位不置 1 时，不会在睡眠状态下使能 PHY 时钟门控。

位 4 **PHYSUSP**：PHY 挂起 (PHY Suspended)

指示 PHY 已挂起。应用程序将 STPPCLK 位置 1 后，一旦 PHY 挂起，此位就会更新。

位 3:2 保留，必须保持复位值。

位 1 **GATEHCLK**：门控 HCLK (Gate HCLK)

当 USB 通信挂起或会话无效时，应用程序会将此位置 1，以停止对除 AHB 总线从接口、主接口和唤醒逻辑之外的模块提供时钟。当 USB 恢复通信或新会话启动时，应用程序将此位清零。

位 0 **STPPCLK**：停止 PHY 时钟 (Stop PHY clock)

当 USB 通信挂起、会话无效或设备断开连接时，应用程序将此位置 1 以停止 PHY 时钟。当 USB 恢复通信或新会话启动时，应用程序将此位清零。

37.15.58 OTG_FS/OTG_HS 寄存器映射

下表提供了 USB OTG 寄存器映射和复位值。

表 234. OTG_FS/OTG_HS 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	OTG_ GOTGCTL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTGVER	BSVLD	ASVLD	DBCT	CIDSTS	Res.	Res.	Res.	EHEN	DHNPEN	HSHNPEN	HNPRQ	HNGSCS	BVALOVAL	BVALOEN	AVALOVAL	AVALOEN	VBVALOVAL	VBVALOEN	SRQ	SRQSCS
	Reset value												0	0	0	0	1				0	0	0	0	0	0	0	0	0	0	0	0	0
0x004	OTG_ GOTGINT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDCHNG	DBCDNE	ADTOCHG	HNGDET	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HNSSCHG	SRSSCHG	Res.	Res.	Res.	Res.	Res.	SEDET	Res.	Res.
	Reset value												0	0	0	0								0	0						0		
0x008	OTG_ GAHBCFG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PTXFELVL	TXFELVL	Res.	Res.	Res.	Res.	Res.	Res.	GINTMSK
	Reset value																								0	0							0
0x00C	OTG_ GUSBCFG	Res.	FDMOD	FHMOD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRDT				HNPCAP	SRPCAP	Res.	PHYSEL	Res.	Res.	Res.	TOCAL		
	Reset value		0	0																0	1	0	1	0	0		1				0	0	0
0x010	OTG_ GRSTCTL	AHBIDL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFNUM					TXFFLSH	RXFFLSH	Res.	FCRST	PSRST	CSRST
	Reset value	1																					0	0	0	0	0	0	0		0	0	0

表 234. OTG_FS/OTG_HS 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
0x014	**OTG_ GINTSTS**	WKUINT	SRQINT	DISCINT	CIDSCHG	LPMINT	PTXFE	HCINT	HPRTINT	RSTDET	Res.	IPXFR/INCOMPISOOUT	IISOIXFR	OEPINT	IEPINT	Res.	Res.	EOPF	ISOODRP	ENUMDNE	USBRST	USBSUSP	ESUSP	Res.	Res.	GONAKEFF	GINAKEFF	NPTXFE	RXFLVL	SOF	OTGINT	MMIS	CMOD
	Reset value	0	0	0	1	0	1	0	0	0		0	0	0	0			0	0	0	0	0	0			0	0	1	0	0	0	0	0
0x018	**OTG_ GINTMSK**	WUIM	SRQIM	DISCINT	CIDSCHGM	LPMINTM	PTXFEM	HCIM	PRTIM	RSTDETM	Res.	IPXFRM/IISOOXFRM	IISOIXFRM	OEPINT	IEPINT	Res.	Res.	EOPFM	ISOODRPM	ENUMDNEM	USBRST	USBSUSPM	ESUSPM	Res.	Res.	GONAKEFFM	GINAKEFFM	NPTXFEM	RXFLVLM	SOFM	OTGINT	MMISM	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	0	0	
0x01C	**OTG_ GRXSTSR (host mode)**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PKTSTS				DPID		BCNT											CHNUM			
	Reset value												0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	OTG_ GRXSTSR (Device mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRMNUM				PKTSTS				DPID		BCNT											EPNUM			
	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
0x020	**OTG_ GRXSTSR (host mode)**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PKTSTS				DPID		BCNT											CHNUM			
	Reset value												0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	OTG_ GRXSTSPR (Device mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRMNUM				PKTSTS				DPID		BCNT											EPNUM			
	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
0x024	**OTG_ GRXFSIZ**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RXFD															
	Reset value																	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0x028	**OTG_ HNPTXFSIZ/ OTG_ DIEPTXF0**	NPTXFD/TX0FD																NPTXFSA/TX0FSA															
	Reset value	0	0	0	0	0	0	1	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
0x02C	**OTG_ HNPTXSTS**	Res.	NPTXQTOP							NPTQXSAV								NPTXFSAV															
	Reset value		0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0x030	**OTG_ GI2CCTL**	BSYDNE	RW	Res.	I2CDATSE	I2CDEVADR		Res.	ACK	I2CEN	ADDR							REGADDR								RWDATA							
	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	**OTG_ GCCFG**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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																																
0x03C	**OTG_CID**	PRODUCT_ID																															
	Reset value	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	0	0	0	0

表 234. OTG_FS/OTG_HS 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
0x054	**OTG_ GLPMCFG**	Res.	Res.	Res.	ENBESL	LPMR CNTSTS			SNDLPM	LPM RCNT			LPMCHIDX				L1RSMOK	SLPSTS	LPM RSP		L1DSEN	BESLTHRS				L1SSEN	REMWAKE	BESL					LPMACK	LPMEN
	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
0x100	**OTG_ HPTXFSIZ**	PTXFSIZ																PTXSA															
	Reset value	0	0	0	0	0	0	1	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
0x104	**OTG_ DIEPTXF1**	INEPTXFD																INEPTXSA															
	Reset value	0	0	0	0	0	0	1	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
0x108	**OTG_ DIEPTXF2**	INEPTXFD																INEPTXSA															
	Reset value	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
. . . .	. . . .	. . . .																															
0x204	**OTG_ DIEPTXF5**	INEPTXFD																INEPTXSA															
	Reset value	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
. . . .	. . . .	. . . .																															
0x244	**OTG_ DIEPTXF7**	INEPTXFD																INEPTXSA															
	Reset value	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0x400	**OTG_ HCFG**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FSLSS	FSLSPCS	
	Reset value																														0	0	0
0x404	**OTG_ HFIR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RLDCTRL	FRIVL															
	Reset value																0	1	1	1	0	1	0	1	0	0	1	1	0	0	0	0	0
0x408	**OTG_ HFNUM**	FTREM																FRNUM															
	Reset value	0	0	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
0x410	**OTG_ HPTXSTS**	PTXQTOP								PTXQSAV								PTXFSAVL															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0x414	**OTG_ HAINT**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HAINT															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x418	**OTG_ HAINTMSK**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HAINTM															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 234. OTG_FS/OTG_HS 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
0x440	OTG_HPRT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSPD		PTCTL				PPWR	PLSTS		Res.	PRST	PSUSP	PRES	POCCHNG	POCA	PENCHNG	PENA	PCDET	PCSTS
	Reset value														0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
0x504	OTG_HCSPLT0	SPLITEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	COMPLSPLT	XACTPOS		HUBADDR							PRTADDR						
	Reset value	0															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x500	OTG_HCCHAR0	CHENA	CHDIS	ODDFRM	DAD							MCNT		EPTYP	LSDEV	Res.	EPDIR	EPNUM					MPSIZ										
	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
0x508	OTG_HCINT0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTERR	FRMOR	BBERR	TXERR	Res.	ACK	NAK	STALL	Res.	CHH	XFRC
	Reset value																						0	0	0	0		0	0	0		0	0
0x510	OTG_HCTSIZ0	Res.	DPID		PKTCNT										XFRSIZ																		
	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
0x50C	OTG_HCINTMSK0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTERRM	FRMORM	BBERRM	TXERRM	Res.	ACKM	NAKM	STALLM	Res.	CHHM	XFRCM
	Reset value																						0	0	0	0	0	0	0	0		0	0
0x514	OTG_HCDMA0	DMAADDR																															
	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
0x520	OTG_HCCHAR1	CHENA	CHDIS	ODDFRM	DAD							MCNT		EPTYP	LSDEV	Res.	EPDIR	EPNUM					MPSIZ										
	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
0x528	OTG_HCINT1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTERR	FRMOR	BBERR	TXERR	Res.	ACK	NAK	STALL	Res.	CHH	XFRC
	Reset value																						0	0	0	0		0	0	0		0	0
0x52C	OTG_HCINTMSK1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTERRM	FRMORM	BBERRM	TXERRM	Res.	ACKM	NAKM	STALLM	Res.	CHHM	XFRCM
	Reset value																						0	0	0	0	0	0	0	0		0	0
0x530	OTG_HCTSIZ1	Res.	DPID		PKTCNT										XFRSIZ																		
	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 234. OTG_FS/OTG_HS 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
0x660	OTG_ HCCHAR11	CHENA	CHDIS	ODDFRM	DAD							MCNT		EPTYP		LSDEV	Res.	EPDIR	EPNUM				MPSIZ										
	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
. . . .	. . . .	.																															
0x66C	OTG_ HCINTMSK11	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTERRM	FRMORM	BBERRM	TXERRM	Res. NYET	ACKM	NAKM	STALLM	Res.	CHHM	XFRCM
	Reset value																						0	0	0	0	0	0	0	0		0	0
. . . .	. . . .	.																															
0x670	OTG_ HCTSIZ11	Res.	DPID		PKTCNT											XFRSIZ																	
	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
. . . .	. . . .	.																															
0x6E0	OTG_ HCCHAR15	CHENA	CHDIS	ODDFRM	DAD							MCNT		EPTYP		LSDEV	Res.	EPDIR	EPNUM				MPSIZ										
	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
. . . .	. . . .	.																															
0x6E4	OTG_ HCSPLT15	SPLITEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	COMPLSPLT	XACTPOS	HUBADDR								PRTADDR						
	Reset value	0															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
. . . .	. . . .	.																															
0x6EC	OTG_ HCINTMSK15	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTERRM	FRMORM	BBERRM	TXERRM	Res. NYET	ACKM	NAKM	STALLM	Res.	CHHM	XFRCM
	Reset value																						0	0	0	0	0	0	0	0		0	0
. . . .	. . . .	.																															
0x6F0	OTG_ HCTSIZ15	Res.	DPID		PKTCNT											XFRSIZ																	
	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 234. OTG_FS/OTG_HS 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
. . . .	. . . .	. . . .																															
0x6F4	OTG_ HCDMA15	DMAADDR																															
	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
. . . .	. . . .	. . . .																															
0x728	OTG_ HCINT11	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTERR	FRMOR	BBERR	TXERR	Res.	ACK	NAK	STALL	Res.	CHH	XFRC
	Reset value																						0	0	0	0		0	0	0		0	0
. . . .	. . . .	. . . .																															
0x7A8	OTG_ HCINT15	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTERR	FRMOR	BBERR	TXERR	Res.	ACK	NAK	STALL	Res.	CHH	XFRC
	Reset value																						0	0	0	0		0	0	0		0	0
0x800	OTG_ DCFG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERRATIM	Res.	Res.	PFIVL		DAD							Res.	NZLSOHSK	DSPD	
	Reset value																	0			0	0	0	0	0	0	0	0	0		0	0	0
0x804	OTG_ DCTL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DSBESLRJCT	Res.	Res.	Res.	Res.	Res.	Res.	POPRGDNE	CGONAK	SGONAK	CGINAK	SGINAK	TCTL			GONSTS	GINSTS	SDIS	RWUSIG
	Reset value														0							0	0	0	0	0	0	0	0	0	0	1	0
0x808	OTG_ DSTS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DEV LN STS		FNSOF															Res.	Res.	Res.	Res.	EERR	ENUMSPD	SUSPSTS
	Reset value									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0
0x810	OTG_ DIEPMSK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NAKM	Res.	Res.	Res.	Res.	Res.	INEPNEM	INEPNMM	ITTXFEMSK	TOM	Res.	EPDM	XFRCM
	Reset value																				0						0	0	0	0		0	0
0x814	OTG_ DOEPMSK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OTEPDM	STUPM	Res.	EPDM	XFRCM
	Reset value																												0	0		0	0
0x818	OTG_ DAINT	OEPINT																IEPINT															
	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 234. OTG_FS/OTG_HS 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
0x81C	**OTG_ DAINTMSK**	OEPM																IEPM															
	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
0x828	**OTG_ DVBUSDIS**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VBUSDT															
	Reset value																	0	0	0	1	0	1	1	1	1	1	0	1	0	1	1	1
0x82C	**OTG_DVB USPULSE**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DVBUSP															
	Reset value																	0	0	0	0	0	1	0	1	1	0	1	1	1	0	0	0
0x830	**OTG_ DTHRCTL**	Res.	Res.	Res.	Res.	ARPEN	Res.	RXTHRLEN									RXTHREN	Res.	Res.	Res.	Res.	TXTHRLEN									ISOTHREN	NONISOTHREN
	Reset value					0											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x834	**OTG_DIE PEMPMSK**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INEPTXFEM															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x838	**OTG_ DEACHINT**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OEP1INT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IEP1INT	Res.
	Reset value															0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x83C	**OTG_DEACHI NTMSK**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OEP1INTM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IEP1INTM	Res.
	Reset value															0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x900	**OTG_ DIEPCTL0**	EPENA	EPDIS	Res.	Res.	SNAK	CNAK	TXFNUM				STALL	Res.	EPTYP		NAKSTS	Res.	USBAEP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MPSIZ	
	Reset value	0	0			0	0	0	0	0	0	0		0	0	0		1														0	0
0x900	**OTG_ DIEPCTL0**	EPENA	EPDIS	SODDFRM/SD1PID	SD0PID/SEVNFRM	SNAK	CNAK	TXFNUM				STALL	Res.	EPTYP		NAKSTS	EONUM/DPID	USBAEP	Res.	Res.	Res.	Res.	MPSIZ										
	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
0x908	**OTG_ DIEPINT0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFE	INEPNE	Res.	ITTXFE	TOC	Res.	EPDISD	XFRC
	Reset value																									1	0		0	0		0	0
0x910	**OTG_ DIEPTSIZ0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PKT CNT		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	XFRSIZ						
	Reset value												0	0													0	0	0	0	0	0	0
0x918	**OTG_ DTXFSTS0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INEPTFSAV															
	Reset value																	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

表 234. OTG_FS/OTG_HS 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
0x920	**OTG_ DIEPCTL1**	EPENA	EPDIS	SODDFRM/SD1PID	SD0PID/SEVNFRM	SNAK	CNAK	TXFNUM				STALL	Res.	EPTYP		NAKSTS	EONUM/DPID	USBAEP	Res.	Res.	Res.	Res.	MPSIZ										
	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
0x928	**OTG_ DIEPINT1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFE	INEPNE	Res.	ITTXFE	TOC	Res.	EPDISD	XFRC
	Reset value																									1	0		0	0		0	0
0x930	**OTG_ DIEPTSIZ1**	Res.	MCNT		PKTCNT										XFRSIZ																		
	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
0x938	**OTG_ DTXFSTS1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INEPTFSAV															
	Reset value																	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0x940	**OTG_ DIEPCTL2**	EPENA	EPDIS	SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	TXFNUM				STALL	Res.	EPTYP		NAKSTS	EONUM/DPID	USBAEP	Res.	Res.	Res.	Res.	MPSIZ										
	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
. . . .	. . . .	.																															
0x9A0	**OTG_ DIEPCTL5**	EPENA	EPDIS	SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	TXFNUM				STALL	Res.	EPTYP		NAKSTS	EONUM/DPID	USBAEP	Res.	Res.	Res.	Res.	MPSIZ										
	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
. . . .	. . . .	.																															
0x9A8	**OTG_ DIEPINT5**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFE	INEPNE	Res.	ITTXFE	TOC	Res.	EPDISD	XFRC
	Reset value																									1	0		0	0		0	0
. . . .	. . . .	.																															
0x9B8	**OTG_ DTXFSTS5**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INEPTFSAV															
	Reset value																	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

表 234. OTG_FS/OTG_HS 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
. . . .	. . . .	. . . .																															
0x9E0	**OTG_ DIEPCTL7**	EPENA	EPDIS	SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	TXFNUM				STALL	Res.	EPTYP		NAKSTS	EONUM/DPID	USBAEP	Res.	Res.	Res.	Res.	MPSIZ										
	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
. . . .	. . . .	. . . .																															
0x9B0	**OTG_ DIEPTSIZ5**	Res.	MCNT		PKTCNT											XFRSIZ																	
	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
0x9E8	**OTG_ DIEPINT7**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFE	INEPNE	Res.	ITTXFE	TOC	Res.	EPDISD	XFRC
	Reset value																									1	0		0	0		0	0
. . . .	. . . .	. . . .																															
0x9F0	**OTG_ DIEPTSIZ7**	Res.	MCNT		PKTCNT											XFRSIZ																	
	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
. . . .	. . . .	. . . .																															
0x9F8	**OTG_FS_ DTXFSTS7**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INEPTFSAV															
	Reset value																	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0xB00	**OTG_ DOEPCTL0**	EPENA	EPDIS	Res.	Res.	SNAK	CNAK	Res.	Res.	Res.	Res.	STALL	SNPM	EPTYP		NAKSTS	Res.	USBAEP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MPSIZ	
	Reset value	0	0			0	0					0	0	0	0	0		1														0	0
0xB08	**OTG_ DOEPINT0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Reserved	B2BSTUP	Res.	OTEPDIS	STUP	Res.	EPDISD	XFRC
	Reset value																										0		0	0		0	0
0xB10	**OTG_ DOEPTSIZ0**	Res.	STUPCNT		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PKTCNT		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	XFRSIZ						
	Reset value		0	0										0													0	0	0	0	0	0	0

表 234. OTG_FS/OTG_HS 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
0xB20	**OTG_ DOEPCTL1**	EPENA	EPDIS	SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	Res.	Res.	Res.	Res.	STALL	SNPM	EPTYP		NAKSTS	EONUM/DPID	USBAEP	Res.	Res.	Res.	Res.	MPSIZ										
	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
0xB28	**OTG_ DOEPINT1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Reserved	B2BSTUP	Res.	OTEPDIS	STUP	Res.	EPDISD	XFRC
	Reset value																										0		0	0		0	0
0xB30	**OTG_ DOEPTSIZ1**	Res.	RXDPID/ STUPCNT		PKTCNT											XFRSIZ																	
	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
0xB50	**OTG_ DOEPTSIZ2**	Res.	RXDPID/ STUPCNT		PKTCNT											XFRSIZ																	
	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
. . . .	. . . .	.																															
0xBA0	**OTG_ DOEPCTL5**	EPENA	EPDIS	SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	Res.	Res.	Res.	Res.	STALL	SNPM	EPTYP		NAKSTS	EONUM/DPID	USBAEP	Res.	Res.	Res.	Res.	MPSIZ										
	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
. . . .	. . . .	.																															
0xBE0	**OTG_ DOEPCTL7**	EPENA	EPDIS	SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	Res.	Res.	Res.	Res.	STALL	SNPM	EPTYP		NAKSTS	EONUM/DPID	USBAEP	Res.	Res.	Res.	Res.	MPSIZ										
	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
. . . .	. . . .	.																															
0xBA8	**OTG_ DOEPINT5**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Reserved	B2BSTUP	Res.	OTEPDIS	STUP	Res.	EPDISD	XFRC
	Reset value																										0		0	0		0	0

表 234. OTG_FS/OTG_HS 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
. . . .	. . . .	.																															
0xBB0	**OTG_ DOEPTSIZ5**	Res.	RXDPID/ STUPCNT		PKTCNT										XFRSIZ																			
	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
. . . .	. . . .	.																															
0xBE8	**OTG_ DOEPINT7**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Reserved	B2BSTUP	Res.	OTEPDIS	STUP	Res.	EPDISD	XFRC
	Reset value																										0		0	0		0	0
. . . .	. . . .	.																															
0xBF0	**OTG_ DOEPTSIZ7**	Res.	RXDPID/ STUPCNT		PKTCNT										XFRSIZ																			
	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
0xE00	**OTG_ PCGCCTL**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUSP	PHYSLEEP	ENL1GTG	PHYSUSP	Res.	Res.	GATEHCLK	STPPCLK
	Reset value																									0	0	0	0			0	0

有关寄存器边界地址的信息，请参见*第 2.2.2 节：存储器映射和寄存器边界地址*。

37.16 OTG_FS/OTG_HS 编程模型

37.16.1 模块初始化

应用程序必须执行模块初始化序列。如果上电期间连接电缆，则 OTG_GINTSTS 中的当前工作模式位（OTG_GINTSTS 中的 CMOD 位）将指示模式。连接“A 型”插头后，OTG_FS/OTG_HS 控制器进入主机模式；连接“B 型”插头后，OTG_FS/OTG_HS 控制器进入设备模式。

本节介绍了 OTG_FS/OTG_HS 控制器上电后的初始化过程。无论是以主机模式还是设备模式工作，应用程序都必须遵循初始化序列。根据模块配置对所有模块全局寄存器进行初始化：

1. 在 OTG_GAHBCFG 寄存器中编程以下字段：
   - 全局中断屏蔽位 GINTMSK = 1
   - Rx FIFO 非空（OTG_GINTSTS 中的 RXFLVL 位）
   - 周期性 Tx FIFO 空门限
2. 在 OTG_GUSBCFG 寄存器中编程以下字段：
   - HNP 功能位
   - SRP 功能位
   - OTG_FS/OTG_HS 超时校准字段
   - USB 周转时间字段
3. 软件必须使能 OTG_GINTMSK 寄存器中的以下位：

   OTG 中断屏蔽

   模式不匹配中断屏蔽
4. 通过读取 OTG_GINTSTS 中的 CMOD 位，软件可确定 OTG_FS/OTG_HS 控制器是在主机模式还是设备模式下工作。

37.16.2 主机初始化

要将模块作为主机进行初始化，应用程序必须执行以下步骤：

1. 编程 OTG_GINTMSK 寄存器中的 HPRTINT 以打开中断。
2. 编程 OTG_HCFG 寄存器以选择全速主机。
3. 将 OTG_HPRT 中的 PPWR 位编程为 1， 给 USB 总线提供 V_{BUS}。
4. 等待 OTG_HPRT0 中的 PCDET 中断。这表示某设备已连接到主机端口。
5. 将 OTG_HPRT 中的 PRST 位编程为 1， 在 USB 总线上发出复位信号。
6. 至少等待 10 ms，以便完成复位过程。
7. 将 OTG_HPRT 中的 PRST 位编程为 0。
8. 等待 OTG_HPRT 中的 PENCHNG 中断。
9. 读取 OTG_HPRT 中的 PSPD 位以获取枚举速度。
10. 使用所选 PHY 时钟 ，相应地设置 HFIR 寄存器。
11. 根据步骤 9 中检测到的设备速度编程 OTG_HCFG 寄存器中的 FSLSPCS 字段。如果 FSLSPCS 发生更改，则必须执行端口复位。
12. 编程 OTG_GRXFSIZ 寄存器以选择接收 FIFO 的大小。

13. 编程 OTG_HNPTXFSIZ 寄存器，以选择用于非周期性通信事务的非周期性发送 FIFO 的大小和起始地址。
14. 编程 OTG_HPTXFSIZ 寄存器，以选择用于周期性通信事务的周期性发送 FIFO 的大小和起始地址。

要与设备通信，系统软件必须初始化并使能至少一个通道。

37.16.3 设备初始化

上电期间或者从主机模式切换为设备模式后，应用程序必须执行下列步骤来将模块作为设备进行初始化。

1. 在 OTG_DCFG 寄存器中编程以下字段：
   - 设备速度
   - 非零长度状态 OUT 握手信号
2. 编程 OTG_GINTMSK 寄存器以使能以下中断：
   - USB 复位
   - 枚举完成
   - 早期挂起
   - USB 挂起
   - SOF
3. 编程 OTG_GCCFG 寄存器中的 VBUSBSEN 位，以使能“B”设备模式的 V_{BUS} 感应并把 DP 线上的上拉电阻上拉到 5V。
4. 等待 OTG_GINTSTS 中的 USBRST 中断。这表示已在 USB 上检测到复位信号，复位过程自接收到此中断后约持续 10 ms。

等待 OTG_GINTSTS 中的 ENUMDNE 中断。此中断指示 USB 上复位过程结束。接收到此中断时，应用程序必须读取 OTG_DSTS 寄存器以确定枚举速度并执行*第 1325 页上的枚举完成时的端点初始化*中所列的步骤。

此时，设备已准备好接受 SOF 数据包并在控制端点 0 上执行控制传输。

37.16.4 DMA 模式

OTG 主机通过 DMA 使用 AHB 主接口来获取应用要发送的数据包数据（AHB 到 USB）和接收更新数据（USB 到 AHB）。AHB 主接口使用设置的 DMA 地址（主机模式下的 HCDMAx 寄存器和设备模式下的 DIEPDMAx/DOEPDMAx 寄存器）来访问数据缓冲区。

37.16.5 主机编程模型

通道初始化

应用程序必须初始化一个或多个通道，之后才能与所连接的设备通信。要初始化和使能通道，应用程序必须执行以下步骤：

1. 编程 OTG_GINTMSK 寄存器以取消对以下位的中断屏蔽：
2. 通道中断
   - 用于 OUT 事务的非周期性发送 FIFO 为空（在流水线事务级别工作且数据包计数字段编程值大于 1 时适用）。
   - 用于 OUT 事务的非周期性发送 FIFO 为半空（在流水线事务级别工作且数据包计数字段编程值大于 1 时适用）。
3. 编程 OTG_HAINTMSK 寄存器以使能所选通道中断。

4. 编程 OTG_HCINTMSK 寄存器，以使能主机通道中断寄存器中反映的和通信事务有关的中断。
5. 编程所选通道的 OTG_HCTSIZx 寄存器，指定以字节为单位的总传输大小和包括短数据包在内的预期数据包个数。应用程序必须使用初始数据 PID（用于第一个 OUT 事务或预期从第一个 IN 事务获取）编程 PID 字段。
6. 编程所选通道的 OTG_HCCHARx 寄存器，指定设备的端点特性，例如类型、速度、方向等。（仅当应用程序准备好发送或接收数据包时，才能通过将通道使能位置 1 来使能通道）。
7. 使用集线器地址和端口地址在 OTG_HS_HCSPLTx 寄存器中对所选通道进行编程（仅分离事务）。
8. 使用缓冲区起始地址在 HCDMAx 寄存器中对所选通道进行编程（仅 DMA 事务）。

通道的停止

应用程序可以通过编程 OTG_HCCHARx 寄存器将 CHDIS 和 CHENA 位置 1 来禁止任何通道。这会使 OTG_FS/OTG_HS 主机清空之前在该通道上发出的请求（如果有）并生成通道停止中断。应用程序在将通道重新分配给其它通信事务之前，必须等待 OTG_HCINTx 中的 CHH 中断。OTG_FS/OTG_HS 主机不会中断已在 USB 上启动的通信事务。

在 DMA 模式下要禁止某个 DMA 通道，应用程序无需检查请求队列中是否有可用空间。OTG_HS 主机将检查是否有空间，在通道仲裁给要禁止的通道时，

写入通道禁止的请求。同时，当 HCCHARx 中的 CHDIS 位置 1 时，将从请求队列中丢弃所有已发出的请求。

禁止通道前，应用程序必须确保非周期性请求队列（禁止非周期性通道时）或周期性请求队列（禁止周期性通道时）中至少有一个空闲空间。应用程序可以在请求队列已满时（禁止通道之前），通过编程 OTG_HCCHARx 寄存器将 CHDIS 位置 1 和将 CHENA 位清零，清空已发出的请求。

出现以下任一情况时，应用程序将禁止通道：

1. IN 或 OUT 通道的 OTG_HCINTx 中接收到 STALL、TXERR、BBERR 或 DTERR 中断。应用程序在接收到通道停止信号之前，必须能够接收相同通道的其它中断（DTERR、Nak、Data、TXERR）。
2. 在非周期性 IN 传输或高带宽中断 IN 传输期间 OTG_HS_HCINTx 中接收到 XFRC 中断
3. 接收到 OTG_GINTSTS 中的 DISCINT（断开设备连接）中断。（应用程序将禁止所有已使能的通道）。
4. 应用程序在传输正常完成之前将其中止。

Ping 协议

当 OTG_HS 主机工作在高速模式下时，如果应用程序要与高速批量或控制（数据和状态阶段）OUT 端点进行通信，则必须使用 ping 协议。当应用程序接收到 NAK/NYET/TXERR 中断时，必须使用 ping 协议。当 HS_OTG 主机接收到上述其中一个响应时，它不会在该端点上继续执行任何通信事务，而是丢弃所有已发出或已获取的 OUT 请求（从请求队列中），然后清空发送 FIFO 中的相应数据。这仅在从模式下有效。在从模式下，应用程序可通过以下两种方式发送 ping 令牌：在使能通道之前，将 HCTSIZx 中的 DOPING 位置 1，或者在通道已使能之后，对 HCTSIZx 寄存器执行写操作时将 DOPING 位置 1。这将使能 HS_OTG 主机将 ping 请求写入到请求队列中。应用程序必须等待设备对 ping 令牌的响应（NAK、ACK 或 TXERR 中断），然后才能继续执行通信事务或发送另一个 ping 令牌。仅当应用程

序从设备的 OUT 端点接收到对 ping 令牌的 ACK 响应后，才能继续执行数据通信事务。在 DMA 模式下工作时，对于批量/控制 OUT 事务，应用程序不需要在收到 NAK/NYET 回复时将 HCTSIZx 中的 DOPING 位置 1。OTG_HS 主机会自动将 HCTSIZx 中的 DOPING 位置 1，然后在批量/控制 OUT 传输时发出 ping 令牌。HS_OTG 主机会持续发送 ping 令牌，直至收到 ACK 为止，随后自动切换到数据通信事务。

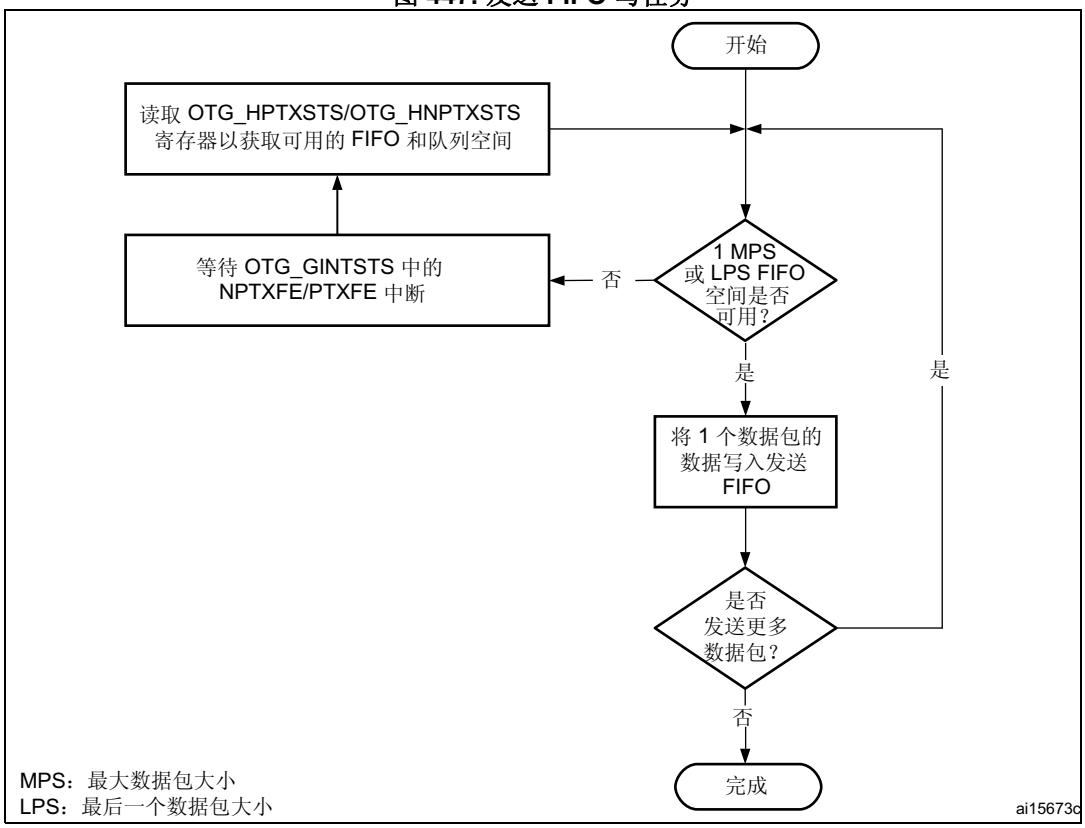
操作模型

应用程序必须初始化一个通道，之后才能与所连接的设备通信。本节介绍了针对不同 USB 事务类型要执行的操作序列。

- **写入发送 FIFO**

  应用程序对数据包执行最后一个 DWORD 写操作的同时，OTG_FS/OTG_HS 主机自动向周期性/非周期性请求队列写入一个条目（OUT 请求）。因此开始向发送 FIFO 写入数据之前，应用程序必须确保周期性/非周期性请求队列中至少有一个空闲空间。应用程序必须始终以 DWORD 形式向发送 FIFO 写入数据。如果数据包大小不是 DWORD 的整数倍，则应用程序必须将数据包填充到 DWORD 的整数倍。OTG_FS/OTG_HS 主机根据设定的最大数据包大小和传输大小确定实际数据包大小。

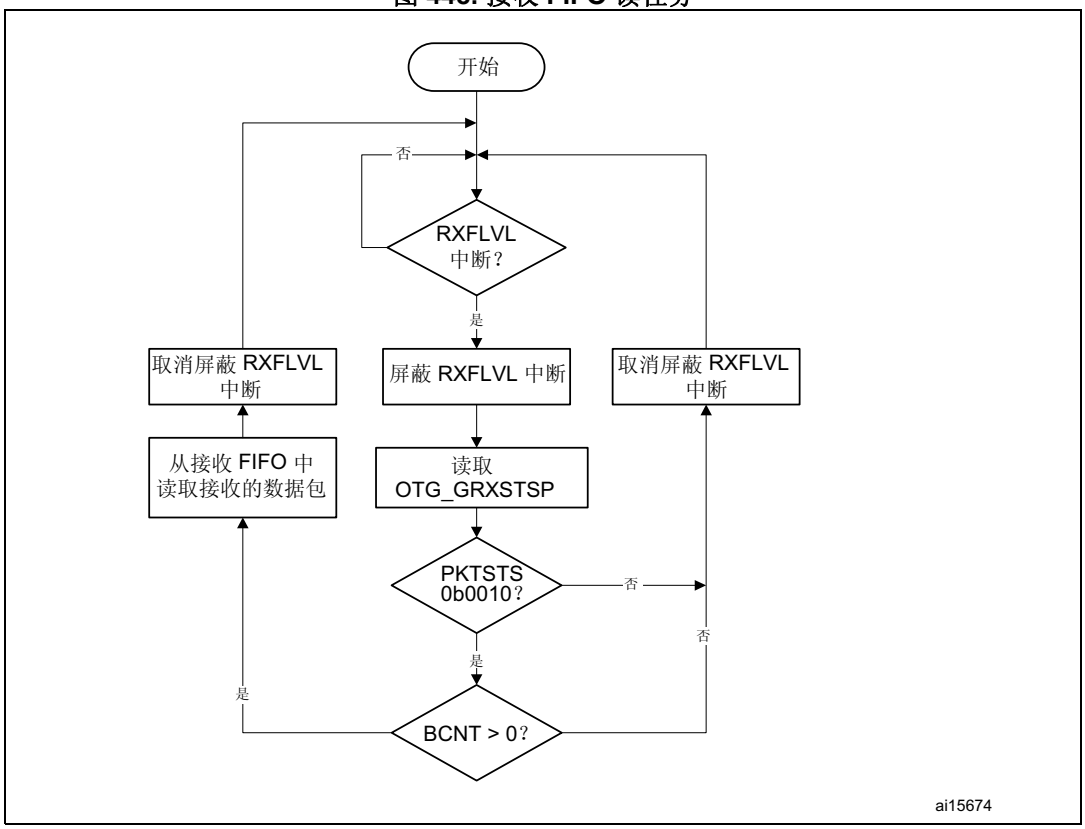
图 447. 发送 FIFO 写任务



- **读取接收 FIFO**

  应用程序必须忽略除 IN 数据包 (bx0010) 以外的所有数据包状态。

图 448. 接收 FIFO 读任务



- **批量和控制传输类型的 OUT/SETUP 通信事务**

  图 449 显示了典型的批量或控制 OUT/SETUP 流水线事务级操作。请参见通道 1 (ch_1)。该通道发送了两个批量 OUT 数据包。控制 SETUP 事务的工作方式相同，只不过只包含一个数据包。假设：

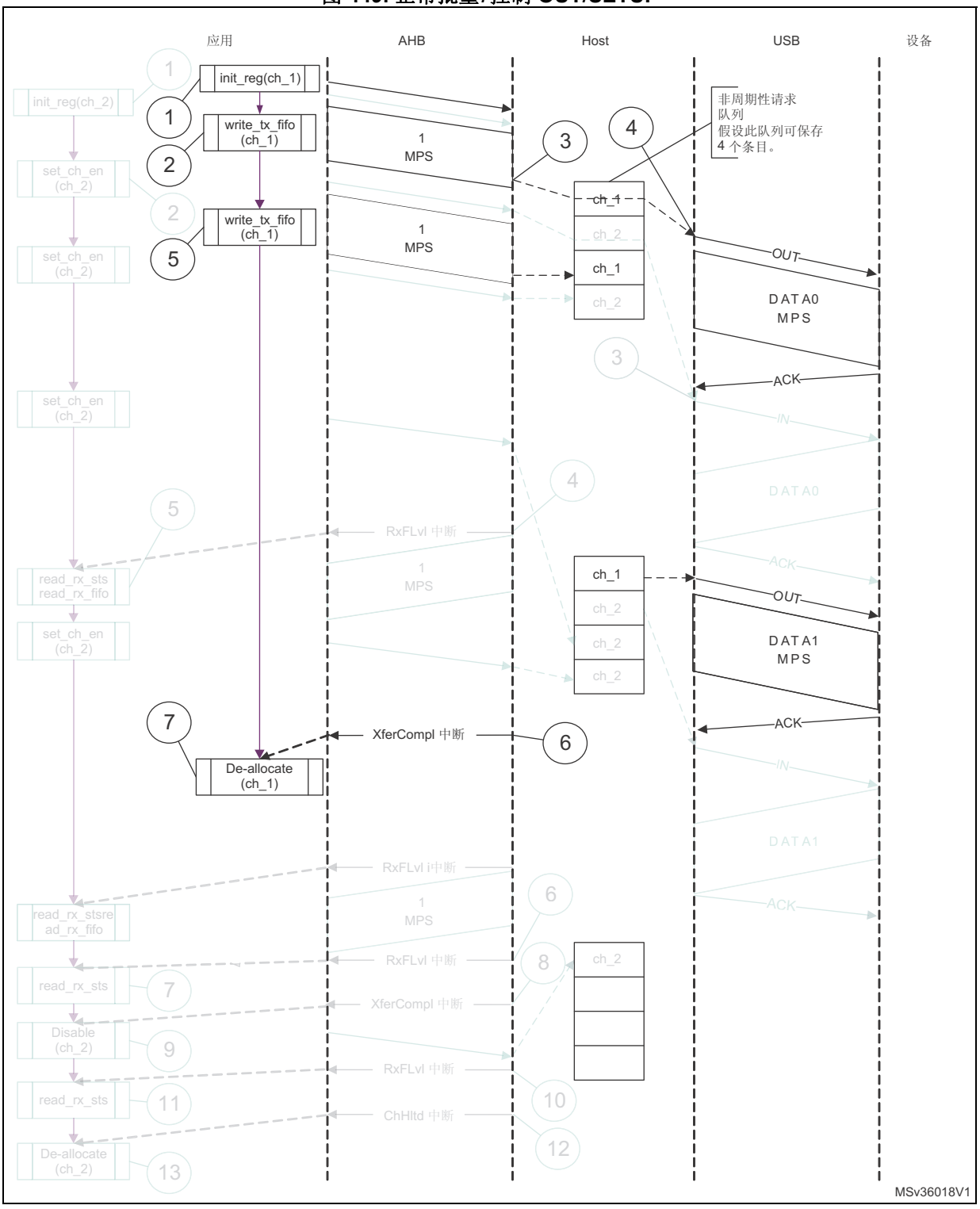
  – 应用程序尝试发送两个最大数据包大小的数据包（传输大小 = 1024 字节）。
  – 非周期性发送 FIFO 可存储两个数据包（HS 情况下 1 KB/FS 情况下 128 字节）。
  – 非周期性请求队列深度 = 4。

- **正常批量和控制传输类型的 OUT/SETUP 操作**

  （通道 1）中的操作顺序如下：

1. 初始化通道 1
2. 写入通道 1 的第一个数据包
3. 在应用执行最后一次 DWORD 写操作时，模块将向非周期性请求队列写入一个请求条目
4. 只要非周期性队列非空，模块即会尝试在当前帧内发送一个 OUT 令牌
5. 写入通道 1 的第二个（最后一个）数据包
6. 成功完成最后一个事务后，模块立即生成 XFRC 中断
7. 为了响应 XFRC 中断，将释放通道以供其它传输操作使用
8. 处理非 ACK 响应

图 449. 正常批量/控制 OUT/SETUP



以下代码示例说明了批量和控制 OUT/SETUP 传输类型的通信事务的通道相关中断服务程序。

- **批量/控制 OUT/SETUP 和批量/控制 IN 事务的中断服务程序**

  a) 批量/控制 OUT/SETUP

```
Unmask (NAK/TXERR/STALL/XFRC)
if (XFRC)
   {
   Reset Error Count
   Mask ACK
   De-allocate Channel
   }
else if (STALL)
   {
   Transfer Done = 1
   Unmask CHH
   Disable Channel
   }
else if (NAK or TXERR )
   {
   Rewind Buffer Pointers
   Unmask CHH
   Disable Channel
   if (TXERR)
      {
      Increment Error Count
      Unmask ACK
      }
   else
      {
      Reset Error Count
      }
      }
else if (CHH)
   {
   Mask CHH
   if (Transfer Done or (Error_count == 3))
      {
      De-allocate Channel
      }
   else
      {
      Re-initialize Channel
      }
   }
   else if (ACK)
      {
      Reset Error Count
      Mask ACK
      }
```

当发送 FIFO 和请求队列中有可用空间时，应用程序会将数据包写入发送 FIFO。应用程序可利用 OTG_GINTSTS 中的 NPTXFE 中断确定发送 FIFO 空间。

b) 批量/控制 IN

```
Unmask (TXERR/XFRC/BBERR/STALL/DTERR)
if (XFRC)
   {
   Reset Error Count
   Unmask CHH
   Disable Channel
   Reset Error Count
   Mask ACK
   }
else if (TXERR or BBERR or STALL)
   {
   Unmask CHH
   Disable Channel
   if (TXERR)
      {
      Increment Error Count
      Unmask ACK
      }
   }
else if (CHH)
   {
   Mask CHH
   if (Transfer Done or (Error_count == 3))
      {
      De-allocate Channel
      }
   else
      {
      Re-initialize Channel
      }
   }
else if (ACK)
   {
   Reset Error Count
   Mask ACK
   }
else if (DTERR)
   {
   Reset Error Count
   }
```

当请求队列空间可用时，应用程序会写入请求，直到接收到 XFRC 中断。

- **批量和控制 IN 事务**

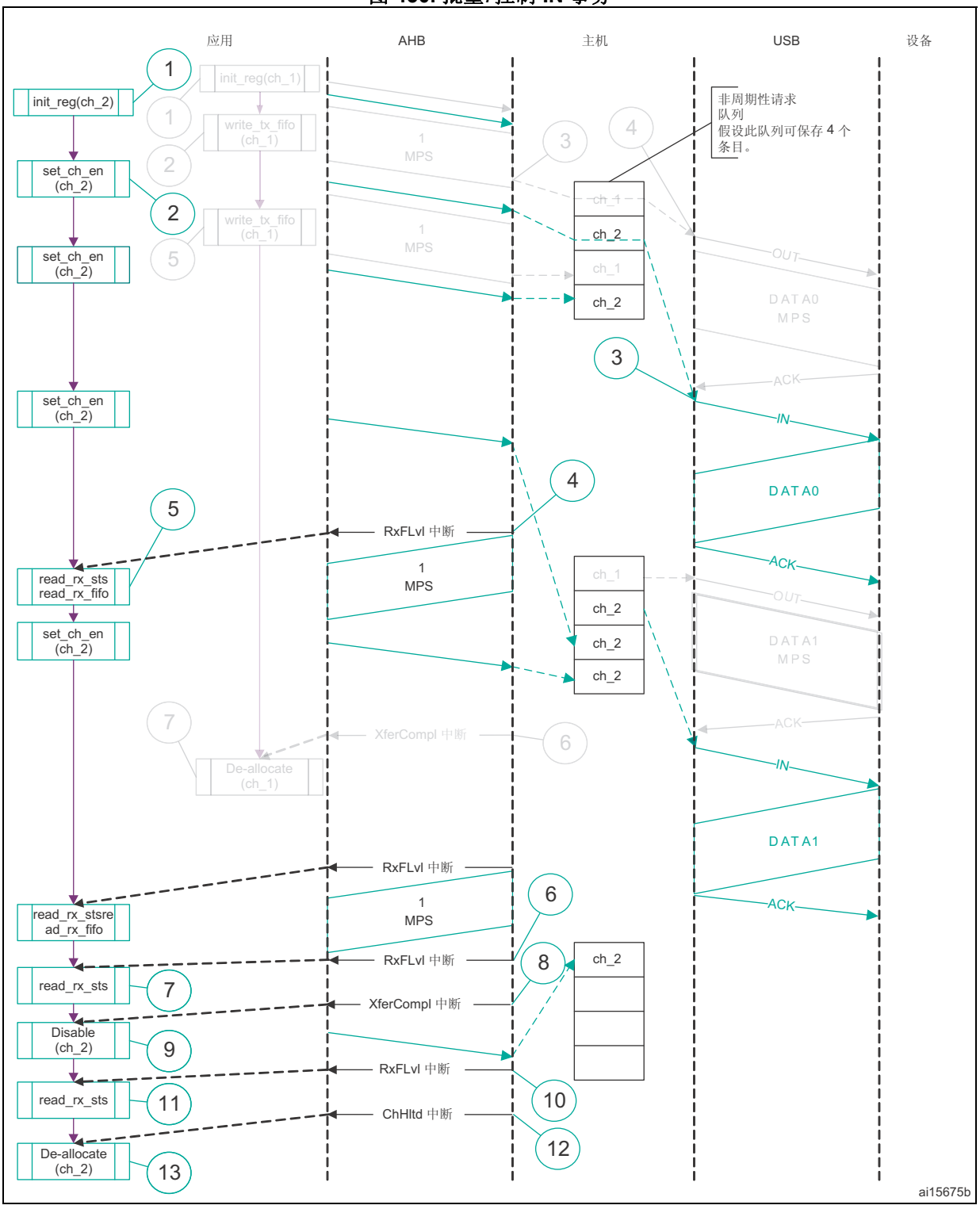
  图 450 显示了典型的批量或控制传输类型的 IN 流水线事务级操作。请参见通道 2 (ch_2)。假设：

  – 应用程序要接收两个最大数据包大小的数据包（传输大小 = 1024 字节）。

  – 接收 FIFO 可以包含至少一个最大数据包大小的数据包和每个数据包的两个状态字（FS 情况下 72 字节/HS 情况下 520 字节）。

  – 非周期性请求队列深度 = 4。

图 450. 批量/控制 IN 事务



操作顺序如下：

1. 初始化通道 2。
2. 将 OTG_HCCHAR2 中的 CHENA 位置 1，以向非周期性请求队列写入 IN 请求。
3. 模块尝试在完成当前 OUT 事务后发送 IN 令牌。
4. 接收到的数据包写入接收 FIFO 后，模块立即生成 RXFLVL 中断。
5. 为了响应 RXFLVL 中断，将屏蔽 RXFLVL 中断并读取接收到的数据包状态，以此确定接收的字节数，然后相应地读取接收 FIFO。之后使能 RXFLVL 中断。
6. 模块针对接收 FIFO 中的传输完成状态条目生成 RXFLVL 中断。
7. 应用程序必须读取接收数据包状态，并在不是 IN 数据包（OTG_GRXSTSR 中的 PKTSTS ≠ 0b0010）时忽略它。
8. 读取接收数据包状态后，模块立即生成 XFRC 中断。
9. 为了响应 XFRC 中断，将禁止通道并停止针对其它请求向 OTG_HCCHAR2 写入数据。向 OTG_HCCHAR2 寄存器写入数据时，模块立即向非周期性请求队列写入通道禁止请求。
10. 停止状态写入接收 FIFO 后，模块立即生成 RXFLVL 中断。
11. 读取并忽略接收数据包状态。
12. 只要停止状态从接收 FIFO 弹出，模块立即生成 CHH 中断。
13. 为了响应 CHH 中断，将释放通道以供其它传输操作使用。
14. 处理非 ACK 响应。

- **控制事务**

  控制传输的建立、数据和状态阶段必须作为三个独立的传输过程来执行。建立、数据和状态阶段的 OUT 事务与上文所述的批量 OUT 事务的执行方式类似。数据或状态阶段的 IN 事务与上文所述的批量 IN 事务的执行方式类似。在所有这三个阶段，应用程序都会将 OTG_HCCHAR1 中的 EPTYP 字段设置为控制传输类型。在建立阶段期间，应用程序会将 OTG_HCTSIZ1 中的 PID 字段设置为 SETUP。

- **中断 OUT 事务**

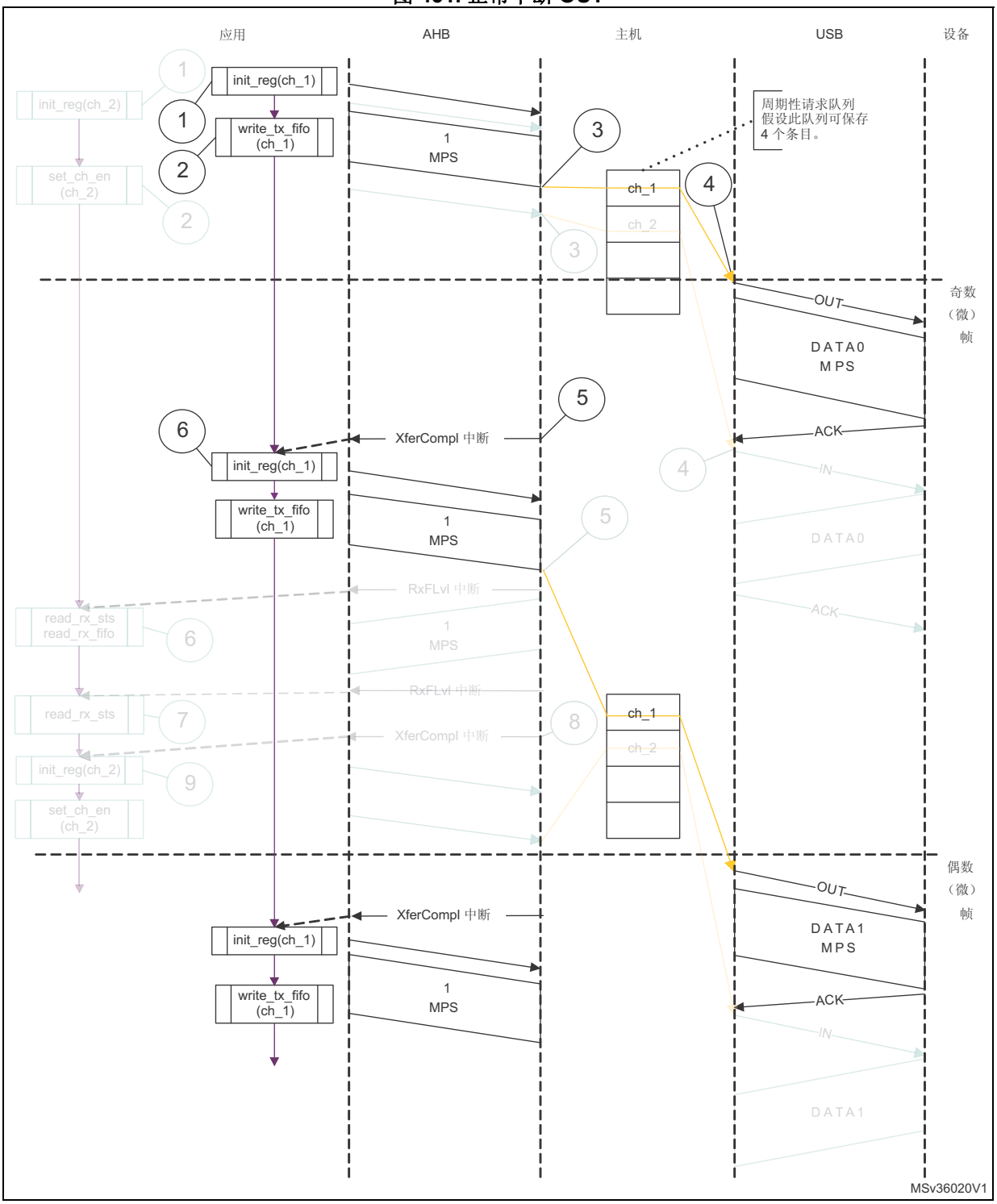
  图 451 显示了典型的中断 OUT 操作。假设：

  – 应用程序尝试从奇数帧开始，每帧发送一个数据包（高达 1 个最大数据包大小）（传输大小 = 1024 字节）
  – 周期性发送 FIFO 可存储一个数据包 (1 KB)
  – 周期性请求队列深度 = 4

  操作顺序如下：

1. 初始化和使能通道 1。应用程序必须将 OTG_HCCHAR1 中的 ODDFRM 位置 1。
2. 写入通道 1 的第一个数据包。
3. 应用程序在执行每个数据包的最后一次字写操作时，OTG_FS/OTG_HS 主机向周期性请求队列写入一个请求条目。
4. OTG_FS/OTG_HS 主机尝试在下一（奇数）帧发送 OUT 令牌。
5. 成功发送最后一个数据包后，OTG_FS/OTG_HS 主机立即生成 XFRC 中断。
6. 为了响应 XFRC 中断，将重新初始化通道以供下一传输操作使用。

图 451. 正常中断 OUT



- **中断 OUT/IN 事务的中断服务程序**

  a) 中断 OUT

```
Unmask (NAK/TXERR/STALL/XFRC/FRMOR)
if (XFRC)
  {
  Reset Error Count
  Mask ACK
  De-allocate Channel
  }
else
  if (STALL or FRMOR)
    {
    Mask ACK
    Unmask CHH
    Disable Channel
    if (STALL)
      {
      Transfer Done = 1
      }
    }
  else
    if (NAK or TXERR)
      {
      Rewind Buffer Pointers
      Reset Error Count
      Mask ACK
      Unmask CHH
      Disable Channel
      }
    else
      if (CHH)
      {
      Mask CHH
      if (Transfer Done or (Error_count == 3))
        {
        De-allocate Channel
        }
      else
        {
        Re-initialize Channel (in next b_interval - 1 Frame)
        }
      }
    else
      if (ACK)
      {
      Reset Error Count
      Mask ACK
      }
```

  应用程序利用 OTG_GINTSTS 中的 NPTXFE 中断确定发送 FIFO 空间。

  b) 中断 IN

```
Unmask (NAK/TXERR/XFRC/BBERR/STALL/FRMOR/DTERR)
if (XFRC)
   {
   Reset Error Count
   Mask ACK
   if (OTG_HCTSIZx.PKTCNT == 0)
      {
      De-allocate Channel
      }
   else
      {
      Transfer Done = 1
      Unmask CHH
      Disable Channel
      }
   }
else
   if (STALL or FRMOR or NAK or DTERR or BBERR)
      {
      Mask ACK
      Unmask CHH
      Disable Channel
      if (STALL or BBERR)
         {
         Reset Error Count
         Transfer Done = 1
         }
      else
         if (!FRMOR)
         {
         Reset Error Count
         }
      }
else
   if (TXERR)
      {
      Increment Error Count
      Unmask ACK
      Unmask CHH
      Disable Channel
      }
else
   if (CHH)
      {
      Mask CHH
      if (Transfer Done or (Error_count == 3))
         {
         De-allocate Channel
         }
      else
         Re-initialize Channel (in next b_interval - 1 /Frame)
      }
```

```
  }
else
  if (ACK)
    {
    Reset Error Count
    Mask ACK
    }
```

- **中断 IN 事务**

  假设：

  – 应用程序要从奇数帧开始，每帧接收一个数据包（最大 1 个最大数据包大小）（传输大小 = 1024 字节）。

  – 接收 FIFO 可以保存至少一个最大数据包大小的数据包和每个数据包的两个状态字（1031 字节）。

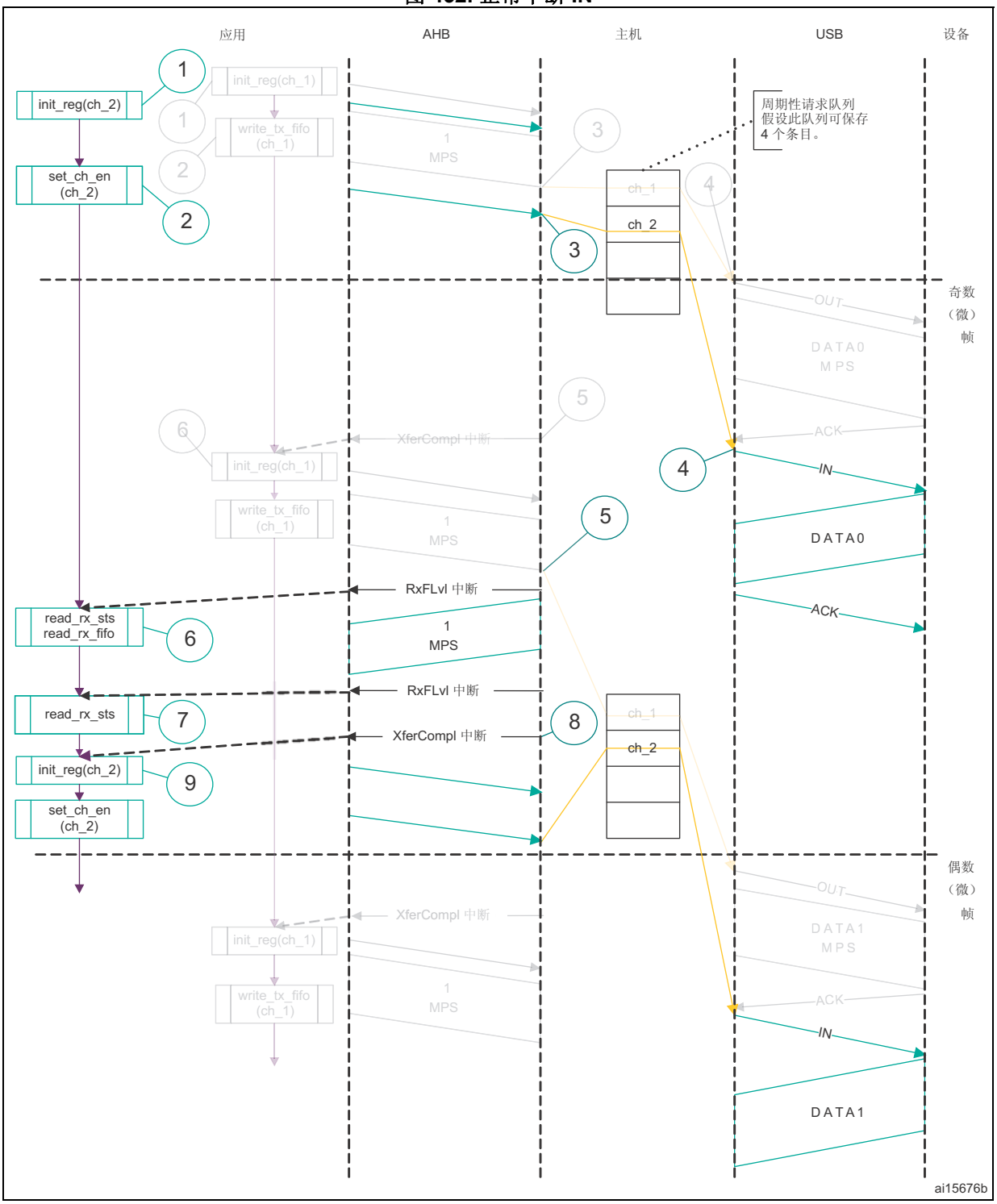
  – 周期性请求队列深度 = 4。

- **正常中断 IN 操作**

  操作顺序如下：

1. 初始化通道 2。应用程序必须将 OTG_HCCHAR2 中的 ODDFRM 位置 1。
2. 将 OTG_HCCHAR2 中的 CHENA 位置 1，以将 IN 请求写入周期性请求队列。
3. 只要 CHENA 置 1，对于每次 OTG_HCCHAR2 寄存器写操作，OTG_FS/OTG_HS 主机都会将一个 IN 请求写入周期性请求队列。
4. OTG_FS/OTG_HS 主机尝试在下一（奇数）帧发送 IN 令牌。
5. 接收到 IN 数据包并写入接收 FIFO 后，OTG_FS/OTG_HS 主机便会立即生成 RXFLVL 中断。
6. 为响应 RXFLVL 中断，读取接收到的数据包状态以确定接收的字节数，然后相应地读取接收 FIFO。应用程序必须在读取接收 FIFO 前屏蔽 RXFLVL 中断，并且在读取整个数据包后使能。
7. 模块针对接收 FIFO 中的传输完成状态条目生成 RXFLVL 中断。应用程序必须读取接收数据包状态，并在不是 IN 数据包（GRXSTSR 中的 PKTSTS ≠ 0b0010）时忽略它。
8. 读取接收数据包状态后，模块立即生成 XFRC 中断。
9. 为响应 XFRC 中断，将读取 OTG_HCTSIZ2 中的 PKTCNT 字段。如果 OTG_HCTSIZ2 中的 PKTCNT 位不等于 0，则在重新初始化通道以进行下次传输前（如果存在），禁止该通道。如果 OTG_HCTSIZ2 中的 PKTCNT 位等于 0，则重新初始化通道以进行下次传输。此时，应用程序必须复位 OTG_HCCHAR2 中的 ODDFRM 位。

图 452. 正常中断 IN


应用
AHB
主机
USB
设备
init_reg(ch_2)
set_ch_en (ch_2)
read_rx_sts read_rx_fifo
read_rx_sts
init_reg(ch_2)
set_ch_en (ch_2)
init_reg(ch_1)
write_tx_fifo (ch_1)
1 MPS
周期性请求队列
假设此队列可保存
4 个条目。
ch_1
ch_2
奇数（微）帧
偶数（微）帧
OUT
DATA0 MPS
DATA1 MPS
ACK
IN
DATA0
DATA1
XferCompl 中断
RxFLvl 中断
ai15676b

- **同步 OUT 事务**

  图 452 中显示了典型的同步 OUT 操作。假设：

  – 应用程序尝试从奇数帧开始，每帧发送一个数据包（最大 1 个最大数据包大小）（传输大小 = 1024 字节）。

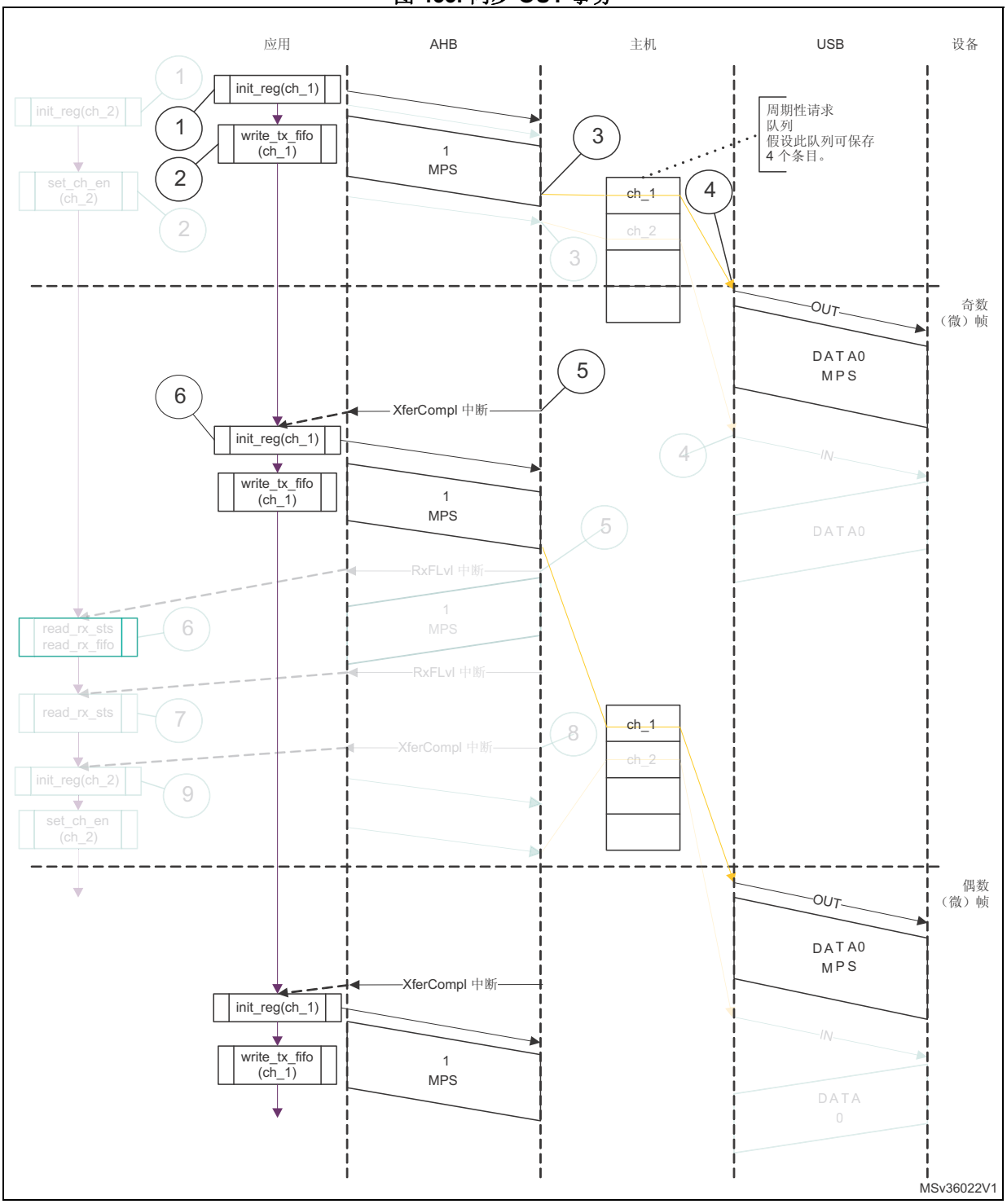
  – 周期性发送 FIFO 可存储一个数据包 (1 KB)。

  – 周期性请求队列深度 = 4。

  操作顺序如下：

1. 初始化和使能通道 1。应用程序必须将 OTG_HCCHAR1 中的 ODDFRM 位置 1。
2. 写入通道 1 的第一个数据包。
3. 应用程序在执行每个数据包的最后一次字写操作时，OTG_FS/OTG_HS 主机向周期性请求队列写入一个请求条目。
4. OTG_FS/OTG_HS 主机尝试在下一（奇数）帧发送 OUT 令牌。
5. 成功发送最后一个数据包后，OTG_FS/OTG_HS 主机立即生成 XFRC 中断。
6. 为了响应 XFRC 中断，将重新初始化通道以供下一传输操作使用。
7. 处理非 ACK 响应。

图 453. 同步 OUT 事务



- **同步 OUT/IN 事务的中断服务程序**

  代码示例：同步 OUT

```
Unmask (FRMOR/XFRC)
if (XFRC)
  {
  De-allocate Channel
  }
else
  if (FRMOR)
    {
    Unmask CHH
    Disable Channel
    }
  else
  if (CHH)
    {
    Mask CHH
    De-allocate Channel
    }
```

  代码示例：同步 IN

```
Unmask (TXERR/XFRC/FRMOR/BBERR)
if (XFRC or FRMOR)
  {
  if (XFRC and (OTG_HCTSIZx.PKTCNT == 0))
    {
    Reset Error Count
    De-allocate Channel
    }
  else
    {
    Unmask CHH
    Disable Channel
    }
  }
  else
    if (TXERR or BBERR)
      {
      Increment Error Count
      Unmask CHH
      Disable Channel
      }
    else
      if (CHH)
        {
        Mask CHH
        if (Transfer Done or (Error_count == 3))
          {
          De-allocate Channel
          }
        else
          {
```

```
        Re-initialize Channel
        }
    }
```

- **同步 IN 事务**

  假设：

  – 应用程序尝试从下一个奇数帧开始，每帧接收一个数据包（最大 1 个最大数据包大小）（传输大小 = 1024 字节）。

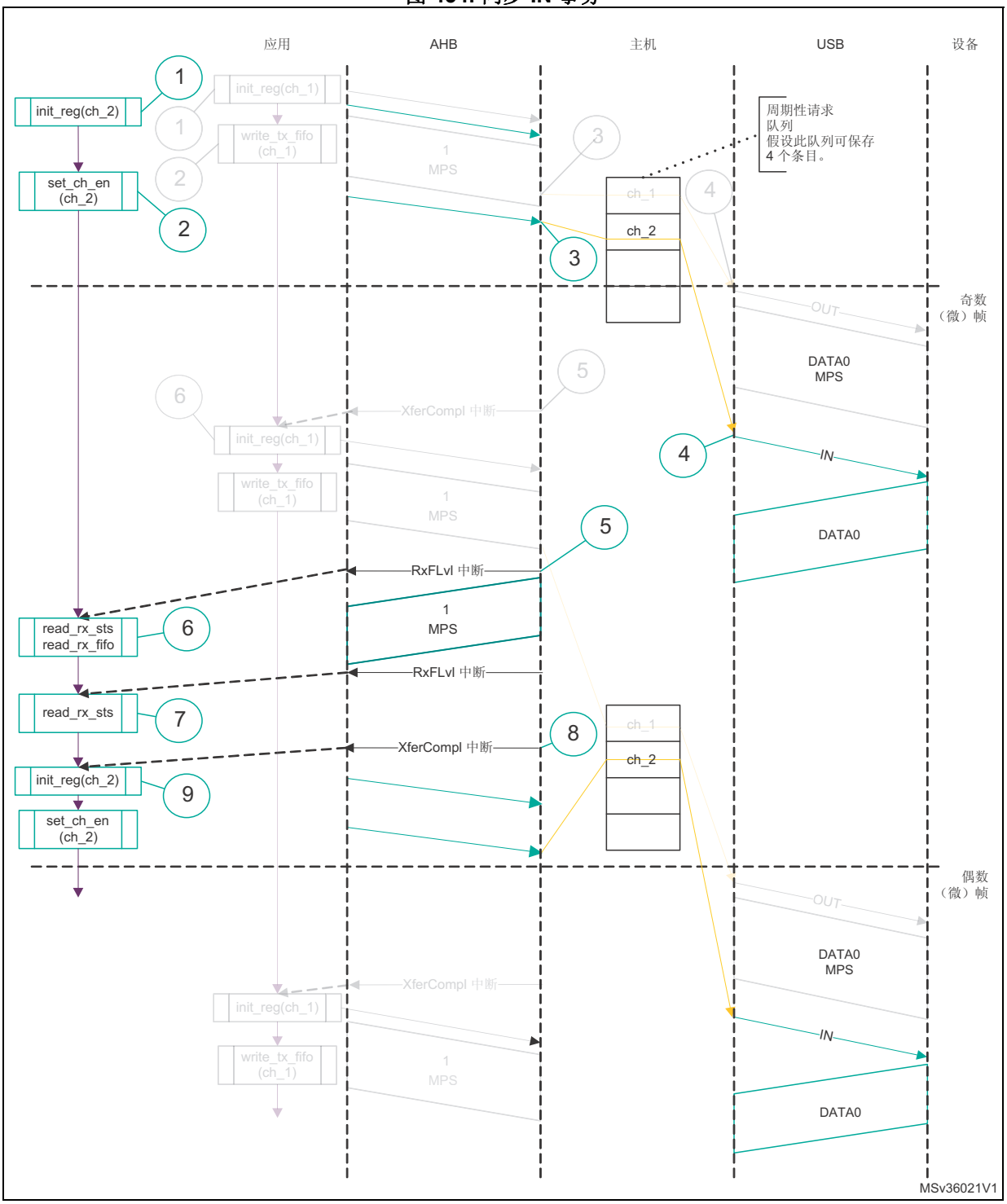
  – 接收 FIFO 可以保存至少一个最大数据包大小的数据包和每个数据包的两个状态字（1031 字节）。

  – 周期性请求队列深度 = 4。

  操作顺序如下：

1. 初始化通道 2。应用程序必须将 OTG_HCCHAR2 中的 ODDFRM 位置 1。
2. 将 OTG_HCCHAR2 中的 CHENA 位置 1，以将 IN 请求写入周期性请求队列。
3. 只要 CHENA 置 1，对于每次 OTG_HCCHAR2 寄存器写操作，OTG_FS/OTG_HS 主机都会将一个 IN 请求写入周期性请求队列。
4. OTG_FS/OTG_HS 主机尝试在下一奇数帧发送 IN 令牌。
5. 接收到 IN 数据包并写入接收 FIFO 后，OTG_FS/OTG_HS 主机便会立即生成 RXFLVL 中断。
6. 为响应 RXFLVL 中断，读取接收到的数据包状态以确定接收的字节数，然后相应地读取接收 FIFO。应用程序必须在读取接收 FIFO 前屏蔽 RXFLVL 中断，并且在读取整个数据包后使能。
7. 模块针对接收 FIFO 中的传输完成状态条目生成 RXFLVL 中断。应用程序必须读取接收数据包状态，并在不是 IN 数据包（OTG_GRXSTSR 中的 PKTSTS 位 ≠ 0b0010）时忽略它。
8. 读取接收数据包状态后，模块立即生成 XFRC 中断。
9. 为响应 XFRC 中断，将读取 OTG_HCTSIZ2 中的 PKTCNT 字段。如果在 OTG_HCTSIZ2 中的 PKTCNT≠ 0，则在重新初始化通道以进行下次传输前（如果存在），禁止该通道。如果 OTG_HCTSIZ2 中的 PKTCNT = 0，则重新初始化通道以进行下次传输。此时，应用程序必须复位 OTG_HCCHAR2 中的 ODDFRM 位。

图 454. 同步 IN 事务


应用
AHB
主机
USB
设备
init_reg(ch_2)
set_ch_en (ch_2)
init_reg(ch_1)
write_tx_fifo (ch_1)
1 MPS
周期性请求队列
假设此队列可保存 4 个条目。
ch_1
ch_2
奇数（微）帧
OUT
DATA0
MPS
XferCompl 中断
IN
RxFLvl 中断
read_rx_sts
read_rx_fifo
read_rx_sts
XferCompl 中断
偶数（微）帧
MSv36021V1

- **选择队列深度**

  请谨慎选择周期性和非周期性请求队列深度，以与要访问的周期性/非周期性端点数量相匹配。

  非周期性请求队列深度会影响非周期性传输的性能。队列越深（加上足够的 FIFO 大小），模块就更能对非周期性传输进行流水线处理。如果队列大小太小，则仅在队列空间释放时模块才能放入新请求。

  要按调度计划执行周期性传输，模块的周期性请求队列深度至关重要。根据一个微帧内要安排的周期性传输次数，选择周期性队列深度。如果周期性请求队列深度小于一个微帧内要安排的周期性传输数，则会发生帧溢出情况。

- **处理串扰情况**

  OTG_FS/OTG_HS 控制器处理两种情况的串扰：数据包串扰和端口串扰。如果设备发送的数据量超过通道的最大数据包大小，则会发生数据包串扰。如果模块从 EOF2（非常接近下一帧的 SOF）时刻还在从设备接收数据，则发生端口串扰。

  当 OTG_FS/OTG_HS 控制器检测到数据包串扰时，它停止向 Rx 缓冲区中写入数据并等待数据包结束信号 (EOP)。当该控制器检测到 EOP 时，它会清空 Rx 缓冲区中的已写入数据并对应用程序生成串扰中断。

  当 OTG_FS/OTG_HS 控制器检测到端口串扰时，它会清空 Rx FIFO 并禁止端口。模块随后将生成“端口已禁止”中断（OTG_GINTSTS 中的 HPRTINT 位和 OTG_HPRT 中的 PENCHNG 位）。在收到该中断时，应用程序必须通过检查 OTG_HPRT 中的 POCA 位，来确定该中断并非由过流（“端口已禁止”中断的另一个原因）所致，然后执行软复位。检测到端口串扰情况后，模块不再发送任何其它令牌。

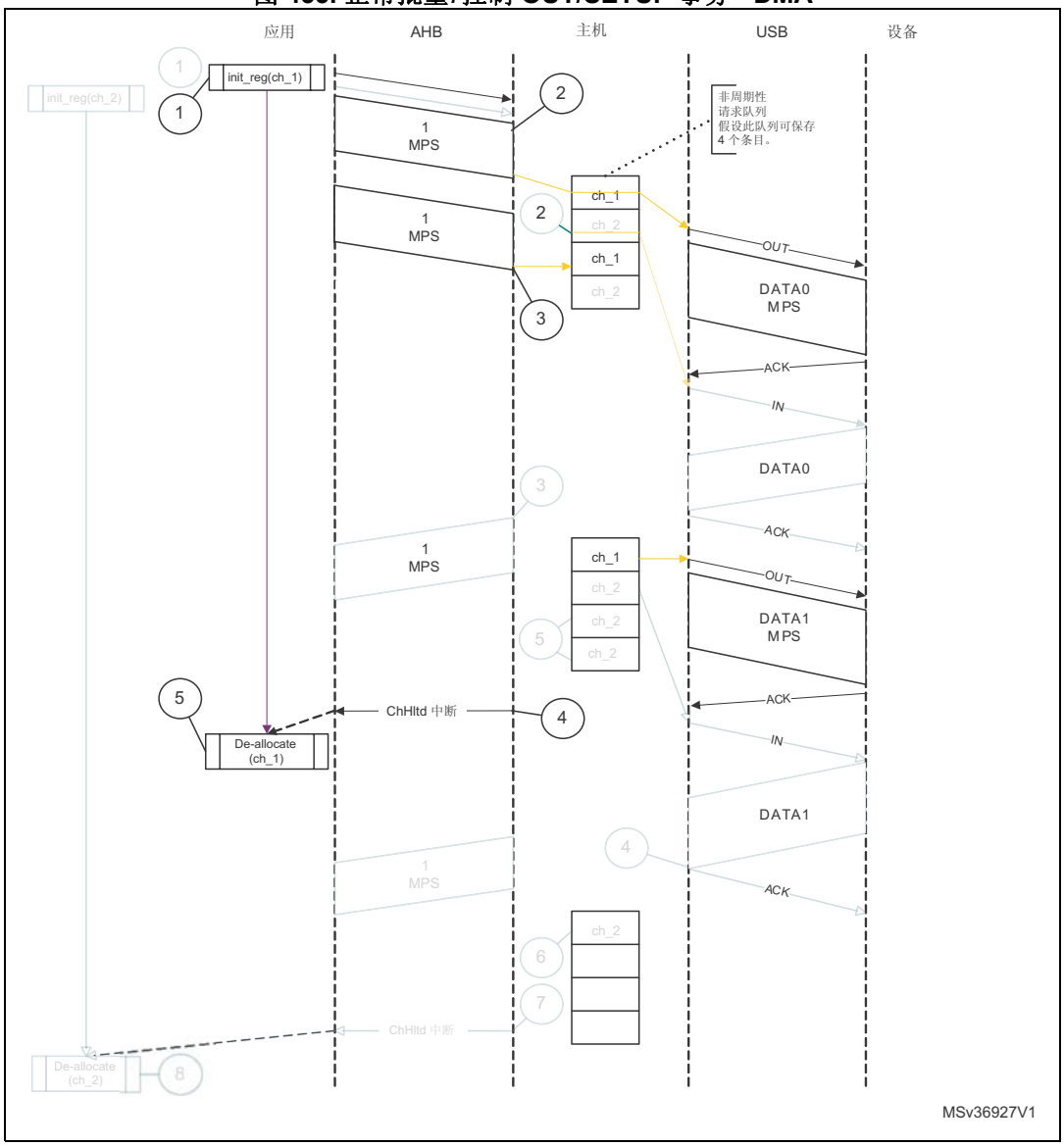
注： *下面涉及 DMA 配置的章节仅适用于 USB OTG HS。*

- **DMA 模式下的批量和控制 OUT/SETUP 事务**

  操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能通道 1。
2. 使能通道后，HS_OTG 主机即开始获取第一个数据包以发送。对于内部 DMA 模式，OTG_HS 主机使用经过编程的 DMA 地址来获取数据包。
3. 获取第二个（最后一个）数据包的最后一个 DWORD 后，OTG_HS 主机会在内部屏蔽通道 1，不参与仲裁。
4. 最后一个数据包发送出去后，HS_OTG 主机立即生成一个 CHH 中断。
5. 为了响应 CHH 中断，将释放通道以供其它传输操作使用。

图 455. 正常批量/控制 OUT/SETUP 事务 - DMA



- **使用内部 DMA 处理 NAK 和 NYET：**

1. OTG_HS 主机发送批量传输类型的 OUT 事务。
2. 设备回复 NAK 或 NYET。
3. 如果应用程序没有屏蔽 NAK 或 NYET 中断，模块将为应用程序生成相应的中断。应用程序并不需要处理这些中断，因为模块会负责调整缓冲区指针和重新初始化通道，而无需应用程序干预。
4. 模块自动发出一个 ping 令牌。
5. 当设备返回 ACK 后，模块会继续传输。应用程序也可以选择使用这些中断（在这种情况下，NAK 或 NYET 中断将被应用程序屏蔽）。

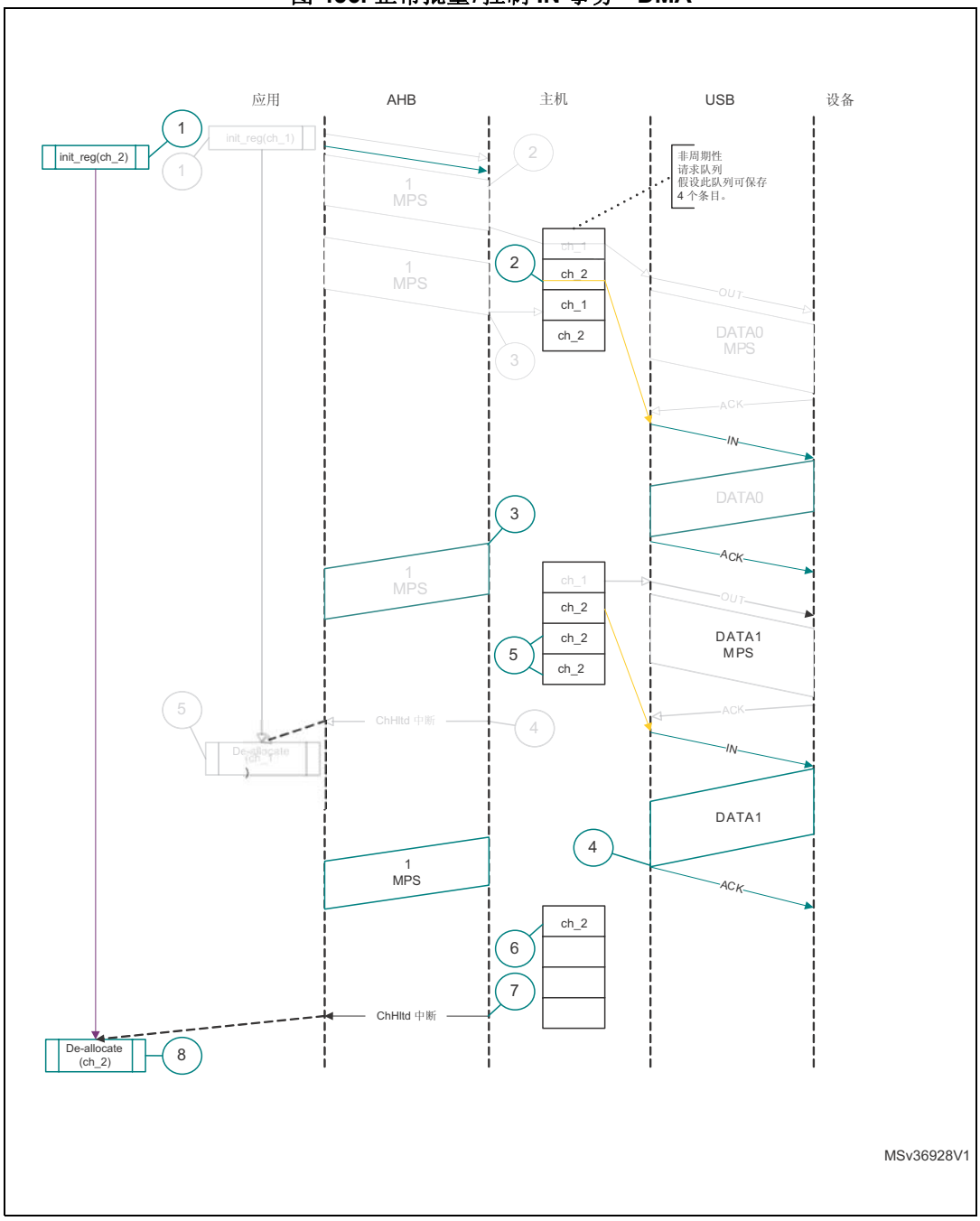
   当主机接收到 NAK 或 NYET 时，模块不会生成单独的中断。

- **DMA 模式下的批量和控制 IN 事务**

  操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能所使用的通道（通道 x）。
2. 当通道接收到来自仲裁器的授权后（以循环方式执行仲裁），OTG_HS 会立即向请求队列写入一个 IN 请求。
3. 当正确接收到最后一个字节后，OTG_HS 主机便开始将接收到的数据写入到系统存储器中。
4. 当接收到最后一个数据包后，OTG_HS 会将一个内部标志置 1，以便从请求队列中删除任何额外的 IN 请求。
5. OTG_HS 主机会清空额外请求。
6. 向请求队列写入最后一个禁止通道的请求。此时，会在内部屏蔽通道 2，不参与仲裁。
7. 当禁止请求到达队列顶端时，OTG_HS 主机即会生成 CHH 中断。
8. 为了响应 CHH 中断，将释放通道以供其它传输操作使用。

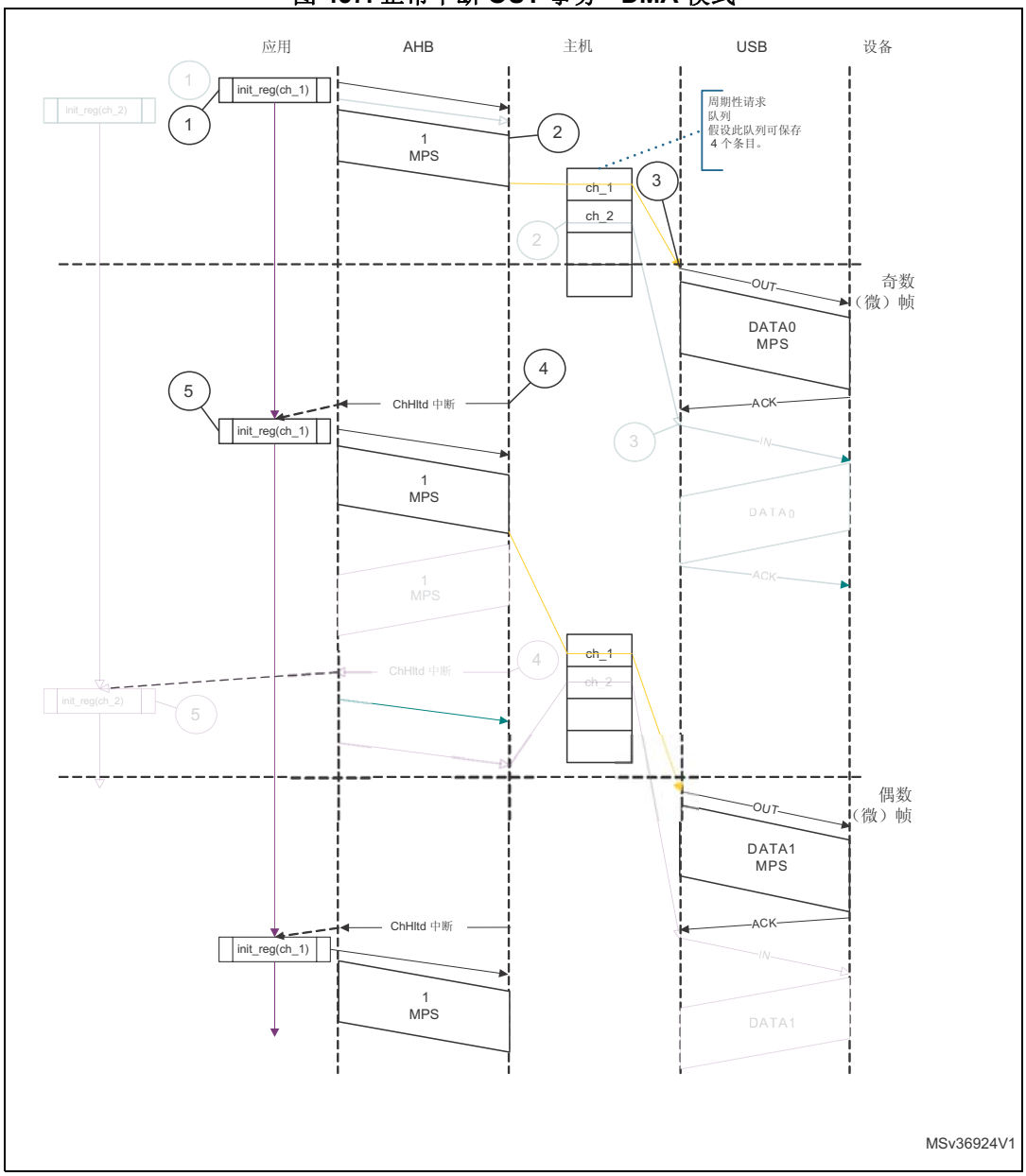
图 456. 正常批量/控制 IN 事务 - DMA



- **DMA 模式下的中断 OUT 事务**

1. 按照*通道初始化*一节中的说明初始化并使能通道 x。
2. 使能通道后，OTG_HS 主机开始获取第一个数据包以发送，并在获取数据包的最后一个 DWORD 后写入 OUT 请求。在高带宽传输模式下，HS_OTG 主机会继续获取下一个数据包（直至数据包数量达到 MC 字段中指定的数值），然后才能切换到下一个通道。
3. OTG_HS 主机尝试在下一个奇数帧/微帧的起始位置发送 OUT 令牌。
4. 成功发送数据包后，OTG_HS 主机将生成一个 CHH 中断。
5. 为了响应 CHH 中断，将重新初始化通道以供下一传输操作使用。

图 457. 正常中断 OUT 事务 - DMA 模式

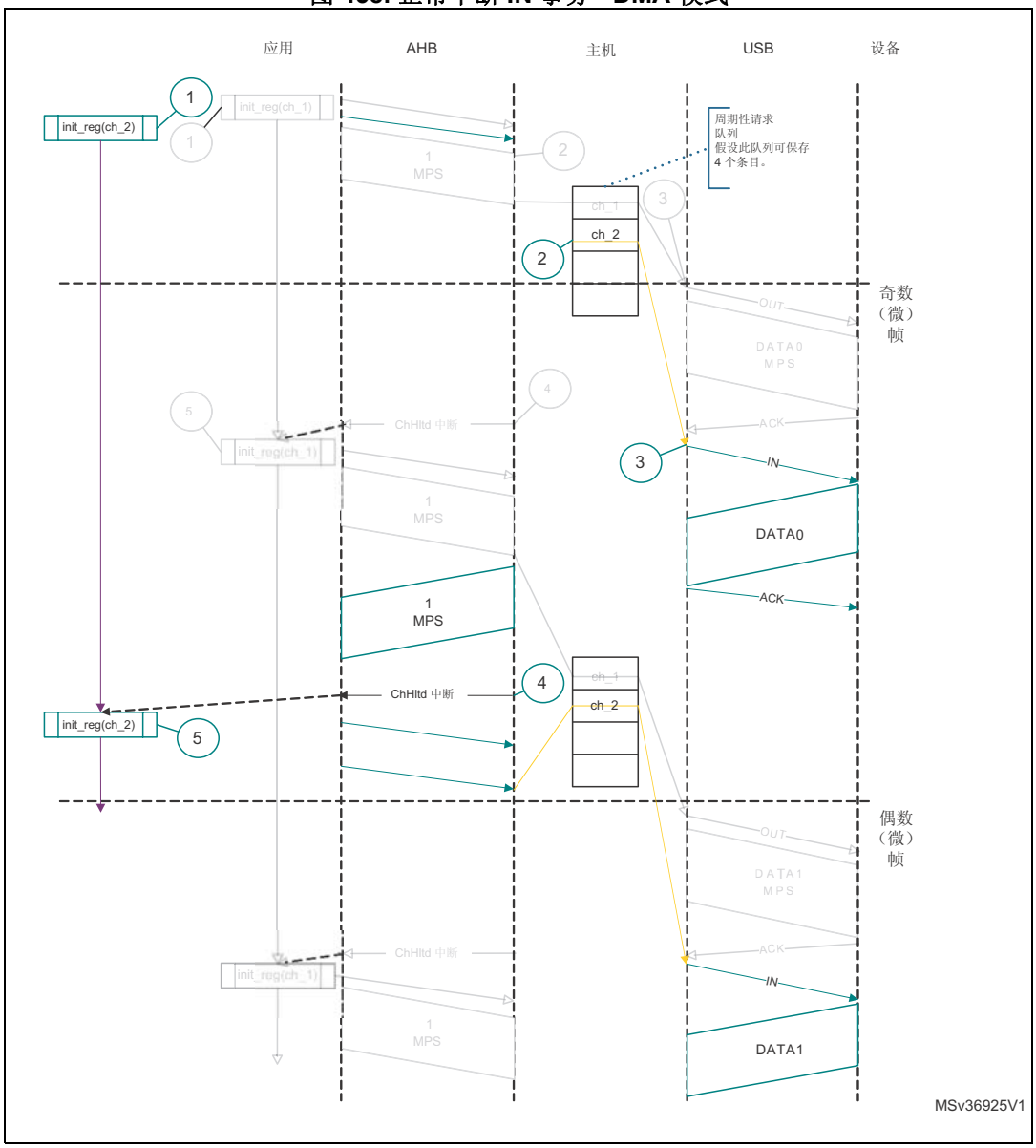


- **DMA 模式下的中断 IN 事务**

  通道 x 的操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能通道 x。
2. 当通道获取了来自仲裁器的授权后（以循环方式执行仲裁），OTG_HS 主机会立即向请求队列写入一个 IN 请求。在高带宽传输模式下，OTG_HS 主机将执行连续写入操作，直到写入的数据包个数达到 MC。
3. OTG_HS 主机尝试在下一个（奇数）帧/微帧的起始位置发送 IN 令牌。
4. 接收到 IN 数据包并写入接收 FIFO 后，OTG_HS 主机便会立即生成 CHH 中断。
5. 为了响应 CHH 中断，将重新初始化通道以供下一传输操作使用。

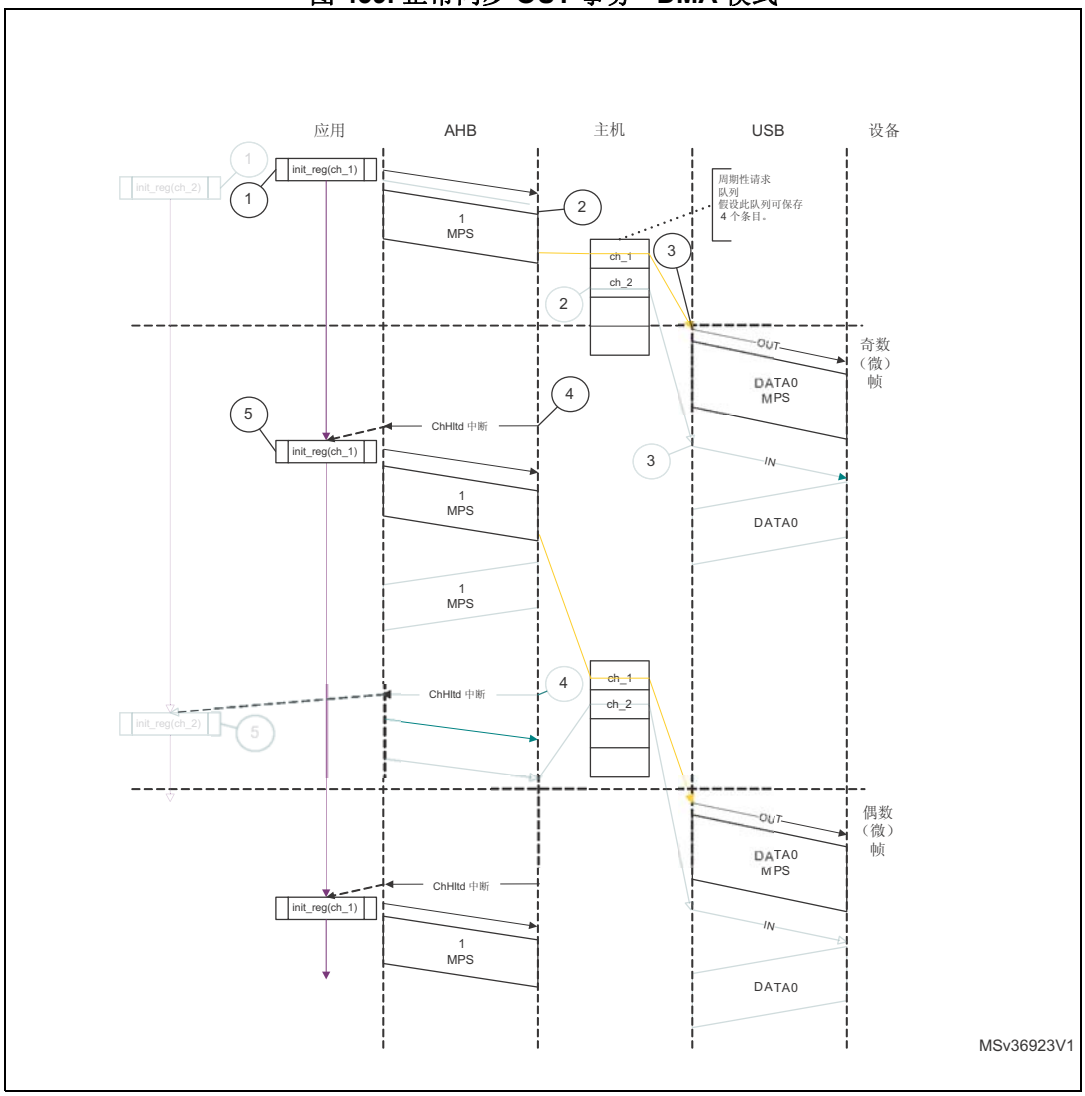
图 458. 正常中断 IN 事务 - DMA 模式



- **DMA 模式下的同步 OUT 事务**

1. 按照*通道初始化*一节中的说明初始化并使能通道 x。
2. 使能通道后，OTG_HS 主机开始获取第一个数据包，并在获取数据包的最后一个 DWORD 后写入 OUT 请求。在高带宽传输模式下，OTG_HS 主机会继续获取下一个数据包（直到写入的数据包个数达到 MC），然后才能切换到下一个通道。
3. OTG_HS 主机尝试在下一个（奇数）帧/微帧的起始位置发送 OUT 令牌。
4. 成功发送数据包后，HS_OTG 主机将生成一个 CHH 中断。
5. 为了响应 CHH 中断，将重新初始化通道以供下一传输操作使用。

图 459. 正常同步 OUT 事务 - DMA 模式

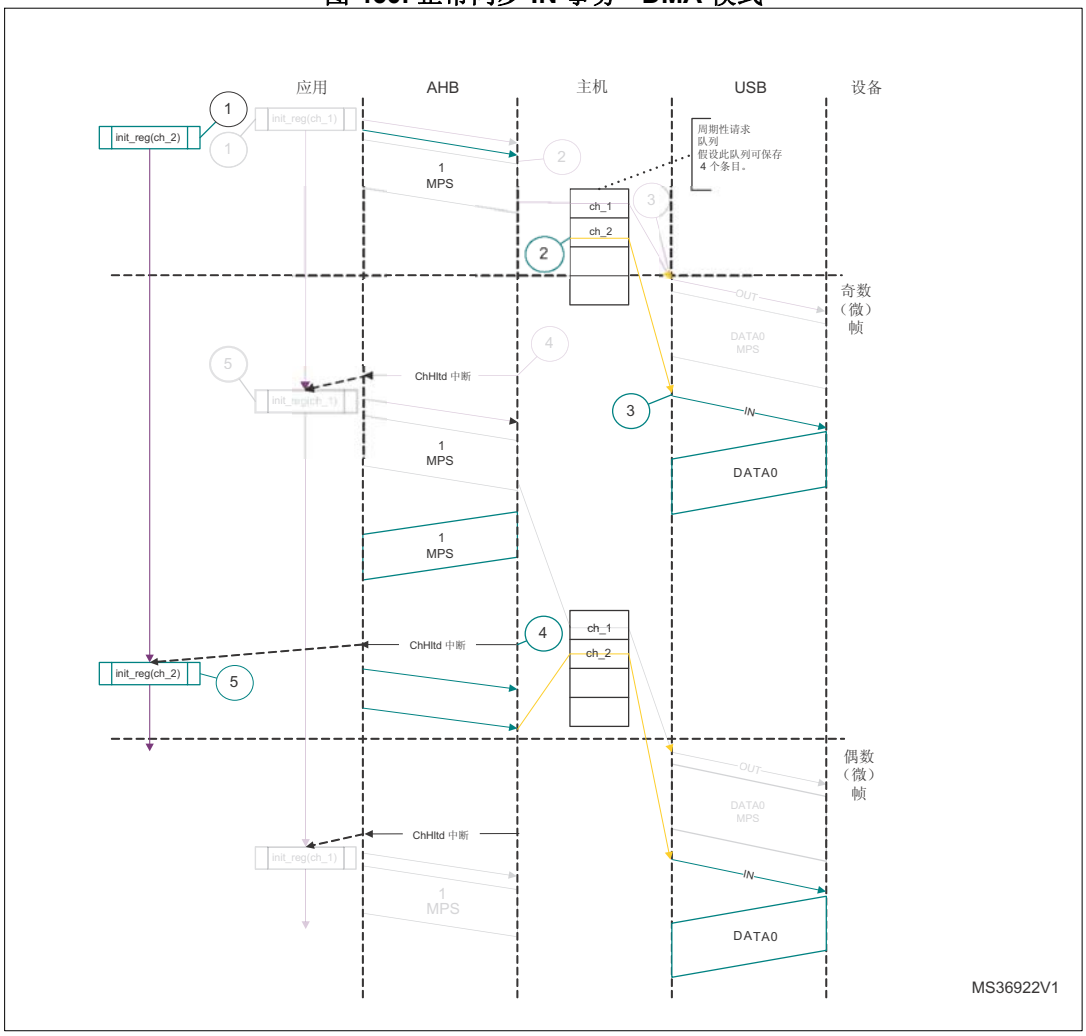


- **DMA 模式下的同步 IN 事务**

  通道 x 的操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能通道 x。
2. 当通道获取了来自仲裁器的授权后（以循环方式执行仲裁），OTG_HS 主机会立即向请求队列写入一个 IN 请求。在高带宽传输模式下，OTG_HS 主机将执行连续写入操作，直到写入的数据包个数达到 MC。
3. OTG_HS 主机尝试在下一个（奇数）帧/微帧的起始位置发送 IN 令牌。
4. 接收到 IN 数据包并写入接收 FIFO 后，OTG_HS 主机便会立即生成 CHH 中断。
5. 为了响应 CHH 中断，将重新初始化通道以供下一传输操作使用。

图 460. 正常同步 IN 事务 - DMA 模式



- **DMA 模式下的批量和控制 OUT/SETUP 分离事务**

（通道 x）中的操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能通道 x 以用于启动分离。
2. 使能通道后，OTG_HS 主机开始获取第一个数据包以发送，并在获取数据包的最后一个 DWORD 后写入 OUT 请求。
3. 成功发送启动分离后，OTG_HS 主机将生成 CHH 中断。
4. 作为对 CHH 中断的响应，将 HCSPLT1 中的 COMPLSPLT 位置 1，以发送完成分离。
5. 成功发送完成分离后，OTG_HS 主机将生成 CHH 中断。
6. 为了响应 CHH 中断，将会释放通道。

- **DMA 模式下的批量/控制 IN 分离事务**

通道 x 的操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能通道 x。
2. 获取了来自仲裁器的授权后，OTG_HS 主机将会向非周期性请求队列写入启动分离请求。写入该请求后，OTG_HS 主机将在内部屏蔽通道 x，不参与仲裁。
3. 发送 IN 令牌后，OTG_HS 主机立即生成 CHH 中断。
4. 作为对 CHH 中断的响应，将 HCSPLT2 中的 COMPLSPLT 位置 1 并重新使能通道，以发送完成分离令牌。这将使能通道 x，并让它参与后续仲裁。
5. 接收到来自仲裁器的授权后，OTG_HS 主机将会向非周期性请求队列写入完成分离请求。
6. 成功接收到数据包后，OTG_HS 主机将开始向系统存储器写入数据包。
7. 只要接收到的数据包写入系统存储器，OTG_HS 主机即会生成一个 CHH 中断。
8. 为了响应 CHH 中断，将会释放通道。

- **DMA 模式下的中断 OUT 分离事务**

（通道 x）中的操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能通道 1 以用于启动分离。应用程序必须将 HCCHAR1 中的 ODDFRM 位置 1。
2. HS_OTG 主机开始读取数据包。
3. HS_OTG 主机尝试发送启动分离事务。
4. 成功发送启动分离后，OTG_HS 主机将生成 CHH 中断。
5. 作为对 CHH 中断的响应，将 HCSPLT1 中的 COMPLSPLT 位置 1，以发送完成分离。
6. 成功结束完成分离事务后，OTG_HS 主机将生成 CHH 中断。
7. 为了响应 CHH 中断，将会释放通道。

- **DMA 模式下的中断 IN 分离事务**

（通道 x）中的操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能通道 x 以用于启动分离。
2. 当通道 x 接收到来自仲裁器的授权后，OTG_HS 主机会立即向请求队列写入一个 IN 请求。
3. OTG_HS 主机尝试在下一个奇数微帧的起始位置发送启动分离 IN 令牌。
4. 成功发送启动分离 IN 令牌后，OTG_HS 主机将生成 CHH 中断。
5. 作为对 CHH 中断的响应，将 HCSPLT2 中的 COMPLSPLT 位置 1，以发送完成分离。
6. 成功接收到数据包后，OTG_HS 主机即会开始向系统存储器写入数据。
7. 将接收到的数据传输到系统存储器后，OTG_HS 主机会生成 CHH 中断。
8. 为了响应 CHH 中断，将释放或重新初始化通道以供下一启动分离使用。

- **DMA 模式下的同步 OUT 分离事务**

通道 x 的操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能通道 x 以用于启动分离（开始）。应用程序必须将 HCCHAR1 中的 ODDFRM 位置 1。编程 MPS 字段。
2. HS_OTG 主机开始读取数据包。
3. 成功发送启动分离后（开始），HS_OTG 主机将生成 CHH 中断。
4. 为了响应 CHH 中断，将重新初始化寄存器以发送启动分离（结束）。
5. 成功发送启动分离后（结束），OTG_HS 主机将生成 CHH 中断。
6. 为了响应 CHH 中断，将会释放通道。

- **DMA 模式下的同步 IN 分离事务**

  通道 x 的操作顺序如下：

1. 按照*通道初始化*一节中的说明初始化并使能通道 x 以用于启动分离。
2. 当通道 x 接收到来自仲裁器的授权后，OTG_HS 主机会立即向请求队列写入一个 IN 请求。
3. OTG_HS 主机尝试在下一个奇数微帧的起始位置发送启动分离 IN 令牌。
4. 成功发送启动分离 IN 令牌后，OTG_HS 主机将生成 CHH 中断。
5. 作为对 CHH 中断的响应，将 HCSPLT2 中的 COMPLSPLT 位置 1，以发送完成分离。
6. 成功接收到数据包后，OTG_HS 主机即会开始向系统存储器写入数据。

   将接收到的数据传输到系统存储器后，OTG_HS 主机会生成 CHH 中断。为了响应 CHH 中断，将释放或重新初始化通道以供下一启动分离使用。

注： *本节内容仅适用于 USB OTG HS。*

37.16.6 设备编程模型

USB 复位时的端点初始化

1. 为所有 OUT 端点将 NAK 位置 1。
   - OTG_DOEPCTLx 中，SNAK = 1（对于所有 OUT 端点）
2. 使能以下中断位。
   - OTG_DAINTMSK 中，INEP0 = 1（控制 0 IN 端点）
   - OTG_DAINTMSK 中，OUTEP0 = 1（控制 0 OUT 端点）
   - OTG_DOEPMSK 中的 STUPM = 1
   - OTG_DOEPMSK 中的 XFRCM = 1
   - OTG_DIEPMSK 中的 XFRCM = 1
   - OTG_DIEPMSK 中的 TOM = 1
3. 为每个 FIFO 设置数据 FIFO RAM。
   - 对 OTG_GRXFSIZ 寄存器进行编程，以能够接收控制传输的 OUT 数据和设置数据。如果未使能阈值，则该寄存器必须至少等于控制端点 0 的 1 个最大数据包大小 + 2 个字（用于控制 OUT 数据包的状态）+ 10 个字（用于 SETUP 数据包）。
   - 对 OTG_DIEPTXF0 寄存器进行编程（取决于所选的 FIFO 编号），以能够发送控制 IN 数据。该寄存器至少必须等于控制端点 0 的 1 个最大数据包大小。
4. 对端点相关寄存器中的以下字段进行编程，以使控制 OUT 端点 0 接收 SETUP 数据包。
   - OTG_DOEPTSIZ0 中的 STUPCNT = 3（接收最多 3 个连续的 SETUP 数据包）。
5. 在 DMA 模式下的 USB OTG HS，DOEPDMA0 寄存器应具有一个有效的存储器地址，以存储接收到的任何 SETUP 数据包。

此时，接收 SETUP 数据包所需的所有初始化工作便已完成。

枚举完成时的端点初始化

1. 在枚举完成中断（OTG_GINTSTS 中的 ENUMDNE）中，读取 OTG_DSTS 寄存器以确定设备的枚举速度。
2. 对 OTG_DIEPCTL0 中的 MPSIZ 字段进行编程以设置最大数据包大小。该步骤配置控制端点 0。控制端点的最大数据包大小取决于枚举速度。
3. 对于 DMA 模式下的 USB OTG HS，编程 DOEPCTL0 寄存器来使能控制 OUT 端点 0，以接收 SETUP 数据包。

此时，设备已准备好接收 SOF 数据包并配置为在控制端点 0 上执行控制传输。

收到 SetAddress 命令时的端点初始化

本节介绍了应用程序在 SETUP 数据包中接收到 SetAddress 命令时必须执行的操作。

1. 使用在 SetAddress 命令中接收到的设备地址来对 OTG_DCFG 寄存器进行编程。
2. 对模块进行编程以发出状态阶段的 IN 数据包。

收到 SetConfiguration/SetInterface 命令时的端点初始化

本节介绍了应用程序在 SETUP 包中接收 SetConfiguration 或 SetInterface 命令时必须执行的操作。

1. 接收到 SetConfiguration 命令时，应用程序必须对端点寄存器进行编程，以使用新配置中有效端点的特性来配置这些端点寄存器。
2. 接收到 SetInterface 命令时，应用程序必须对命令指定的端点的端点寄存器进行编程。
3. 在先前配置或其它设置中有效的端点在新的配置或其它设置中无效。必须停用这些无效端点。
4. 使用 OTG_DAINTMSK 寄存器使能有效端点的中断，屏蔽无效端点的中断。
5. 为每个 FIFO 设置数据 FIFO RAM。
6. 配置完所有必需的端点后，应用程序必须对模块进行编程以发送状态阶段的 IN 数据包。

此时，设备模块已可以接收和发送任何类型的数据包。

端点激活

本节介绍激活设备端点或者将现有设备端点配置为新类型所需的步骤。

1. 在 OTG_DIEPCTLx 寄存器（对于 IN 或双向端点）或 OTG_DOEPCTLx 寄存器（对于 OUT 或双向端点）的以下字段中，对所需端点的特性进行编程。
   - 最大数据包大小
   - USB 活动端点位置 1
   - 端点初始数据同步位（对于中断和批量端点）
   - 端点类型
   - Tx FIFO 编号
2. 激活端点后，模块便开始解码发送到该端点的令牌，并在收到的令牌有效的情况下回复有效握手信号。

端点停用

本节介绍停用现有端点所需的步骤。

1. 在要停用的端点中，将 OTG_DIEPCTLx 寄存器（对于 IN 或双向端点）或 OTG_DOEPCTLx 寄存器（对于 OUT 或双向端点）中的 USB 活动端点位清零。
2. 停用端点后，模块便会忽略发送到该端点的令牌，从而导致 USB 超时。

注： *应用程序必须满足以下条件才能设置设备模块以处理通信：*
必须将 OTG_GINTMSK 寄存器中的 NPTXFEM 和 RXFLVLM 清零。

操作模型

SETUP 和 OUT 数据传输：

本节介绍了数据 OUT 传输和 SETUP 事务期间的内部数据流和应用程序操作步骤。

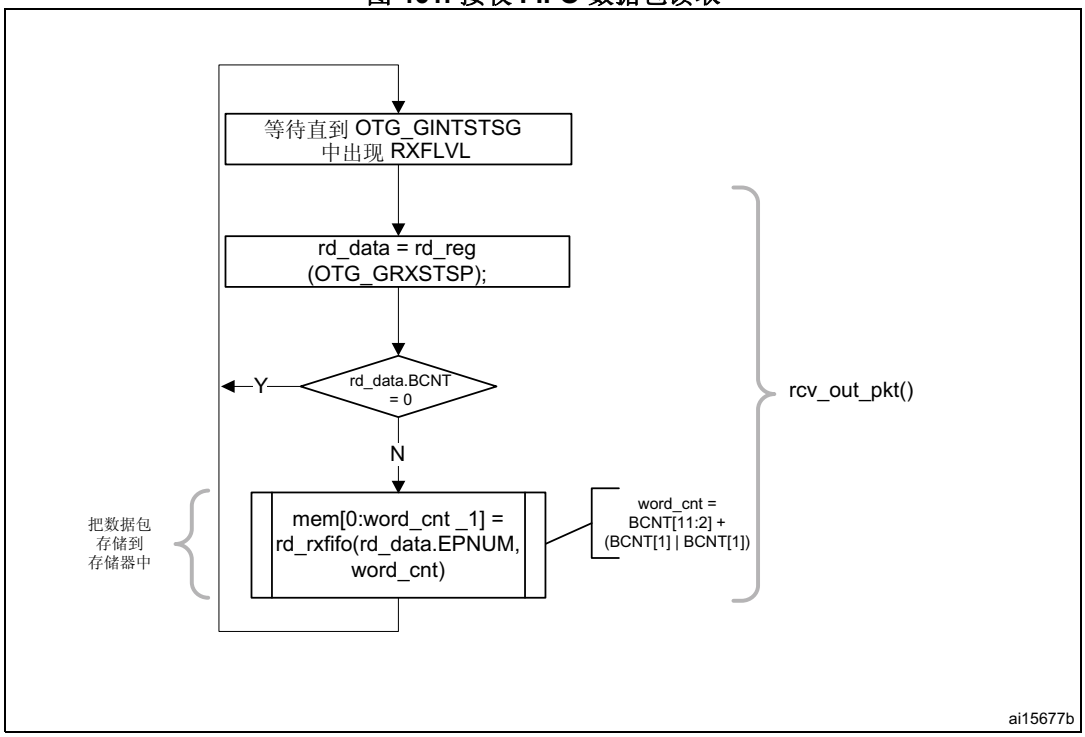
- **数据包读取**

本节介绍如何从接收 FIFO 读取数据包（OUT 数据和 SETUP 数据包）。

1. 捕获到 RXFLVL 中断（OTG_GINTSTS 寄存器）时，应用程序必须读取接收状态弹出寄存器 (OTG_GRXSTSP)。
2. 应用程序可以通过写入 RXFLVLM = 0（在 OTG_GINTMSK 中）来屏蔽 RXFLVL 中断（在 OTG_GINTSTS 中），直到它把数据包从接收 FIFO 中读取出来。
3. 如果已接收数据包的字节计数不是 0，则从接收数据 FIFO 中弹出这些数据并存储在存储器中。如果接收到的数据包字节计数为 0，则不会从接收数据 FIFO 中弹出任何数据。
4. 从接收 FIFO 读出的数据包状态有以下几种状态：
   a) 全局 OUT NAK：
   PKTSTS = 全局 OUT NAK，BCNT = 0x000，EPNUM = (0x0)，DPID = (0b00)。
   这些数据表示全局 OUT NAK 位已生效。
   b) SETUP 数据包：
   PKTSTS = SETUP，BCNT = 0x008，EPNUM = 控制端点编号，DPID = DATA0。
   这些数据表示指定端点上收到的 SETUP 数据包现在可从接收 FIFO 中读取。
   c) 建立阶段完成：
   PKTSTS = 建立阶段完成，BCNT = 0x0，EPNUM = 控制端点编号，
   DPID = (0b00)。
   这些数据表示指定端点的建立阶段完成并且数据阶段已启动。在此状态条目从接收 FIFO 中弹出后，模块将在该控制 OUT 端点上产生建立中断。
   d) OUT 数据包：
   PKTSTS = DataOUT，BCNT = 接收的 OUT 数据包的大小 (0 ≤ BCNT ≤ 1 024)，
   EPNUM = 收到数据包的端点编号，DPID = 实际数据 PID。
   e) 数据传输完成：
   PKTSTS = OUT 数据传输完成，BCNT = 0x0，EPNUM = 完成数据传输的 OUT 端点编号，DPID = (0b00)。
   这些数据表示指定 OUT 端点的 OUT 数据传输完成。在此状态条目从接收 FIFO 中弹出后，模块将在指定的 OUT 端点上引发“传输完成”中断。
5. 从接收 FIFO 中弹出数据后，必须使能 RXFLVL 中断 (OTG_GINTSTS)。
6. 每次应用程序检测到 OTG_GINTSTS 中的 RXFLVL 中断时，都将重复步骤 1 到 5。读取空的接收 FIFO 可能导致未定义的模块行为。

图 461 提供了上述过程的流程图。

图 461. 接收 FIFO 数据包读取



SETUP 事务

本节介绍了模块处理 SETUP 数据包的方式以及应用程序处理 SETUP 事务的顺序。

- **应用程序要求**

1. 要接收 SETUP 数据包，必须将控制 OUT 端点中的 STUPCNT 字段 (OTG_DOEPTSIZx) 编程为非零值。如果应用程序将 STUPCNT 字段编程为非零值，模块会接收 SETUP 数据包并将其写入接收 FIFO，而不考虑 NAK 状态和 OTG_DOEPCTLx 中的 EPENA 位设置。控制端点每收到一个 SETUP 数据包后，STUPCNT 字段都会递减。如果在接收 SETUP 数据包之前，未将 STUPCNT 字段编程为适当值，模块仍能接收 SETUP 数据包并使 STUPCNT 字段递减，但应用程序可能无法确定在控制传输的建立阶段中接收的 SETUP 数据包正确数量。
   – 在 OTG_DOEPTSIZx 中，STUPCNT = 3。
2. 应用程序必须始终在接收数据 FIFO 中分配一些额外空间，以便能够在控制端点上接收连续的最多三个 SETUP 数据包。
   – 预留空间 10 个字。第一个 SETUP 数据包需要 3 个字，“建立阶段完成”状态双字需要 1 个字，还需要 6 个字以存储两个额外的 SETUP 数据包。
   – 每个 SETUP 数据包需要 3 个字以存储 8 个字节的 SETUP 数据和 4 个字节的 SETUP 状态。模块将在接收 FIFO 中保留这些空间。
   – 这段 FIFO 仅用于存储 SETUP 包，绝对不会将该空间用于数据包。
3. 应用程序必须从接收 FIFO 中读取 SETUP 数据包的 2 个字。
4. 应用程序必须从接收 FIFO 中读取并丢弃“建立阶段完成”状态字。

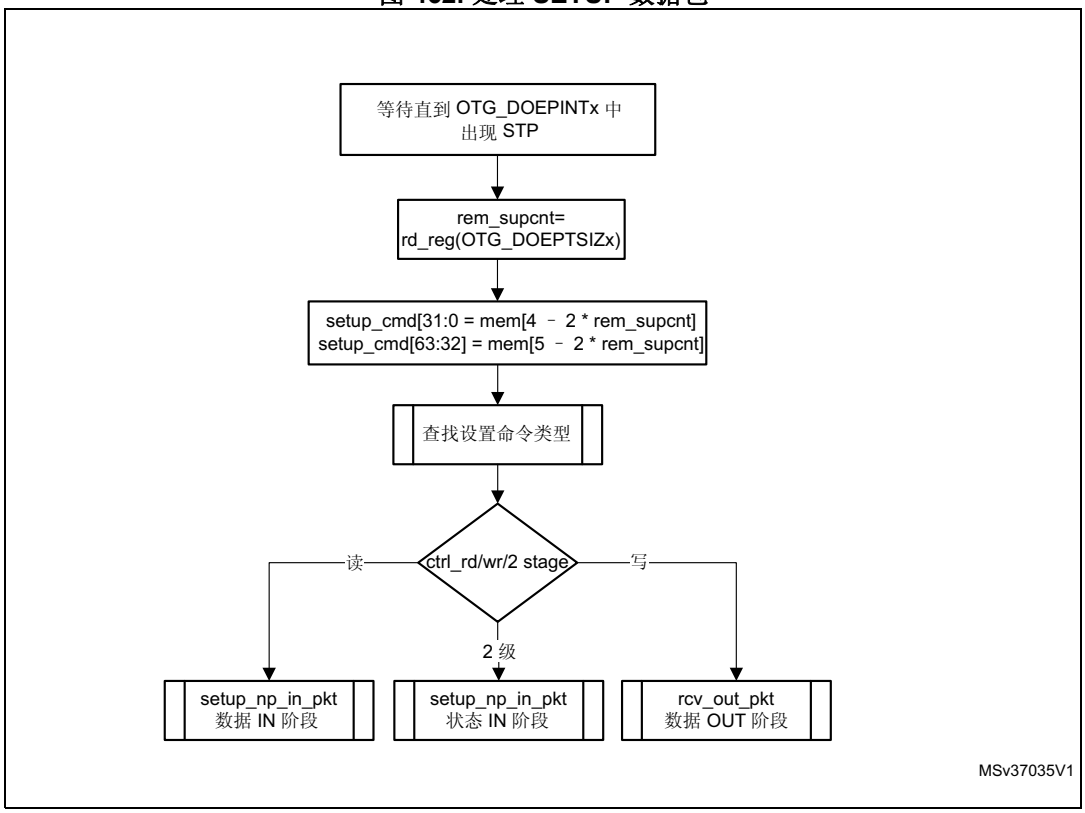
- **内部数据流**

1. 接收到 SETUP 数据包时，模块会将接收到的数据写入接收 FIFO，而不会检查接收 FIFO 中的可用空间，且不考虑端点的 NAK 和 STALL 位设置。
   – 模块会在内部将接收到 SETUP 数据包的控制 IN/OUT 端点的 IN NAK 和 OUT NAK 位置 1。
2. USB 上接收到的每个 SETUP 数据包，模块会将 3 个字的数据写入接收 FIFO，并且将 STUPCNT 字段递减 1。
   – 第一个字包含由模块所使用的内部控制信息
   – 第二个字包含 SETUP 命令的前 4 个字节
   – 第三个字包含 SETUP 命令的最后 4 个字节
3. 当建立阶段结束，数据 IN/OUT 阶段开始时，模块会将一个状态条目（“建立阶段完成”字）写入接收 FIFO，指示建立阶段完成。
4. 在 AHB 端，SETUP 数据包被应用程序读取。
5. 当应用程序从接收 FIFO 中弹出“建立阶段完成”字时，模块将使用 STUP 中断 (OTG_DOEPINTx) 来中断应用程序，指示其可以处理接收到的 SETUP 数据包。
6. 模块会将控制 OUT 端点的端点使能位清零。

- **应用程序编程顺序**

1. 对 OTG_DOEPTSIZx 寄存器进行编程。
   – STUPCNT = 3
2. 等待 RXFLVL 中断 (OTG_GINTSTS) 并且从接收 FIFO 中读取数据包。
3. STUP 中断的触发 (OTG_DOEPINTx) 表示 SETUP 数据传输成功完成。
   – 发生该中断时，应用程序必须读取 OTG_DOEPTSIZx 寄存器以确定接收的 SETUP 数据包数量并处理最后接收的 SETUP 数据包。

图 462. 处理 SETUP 数据包



- **处理三个以上连续的 SETUP 数据包**

根据 USB 2.0 规范，在 SETUP 数据包错误中，主机通常不会向同一个端点发送 3 个以上连续的 SETUP 数据包。但是，USB 2.0 规范并未限制主机可以向同一个端点发送的连续 SETUP 数据包数量。出现这种情况时，OTG_FS/OTG_HS 控制器将生成中断（OTG_DOEPINTx 中的 B2BSTUP）。

- **将全局 OUT NAK 置 1**

内部数据流：

1. 如果应用程序将全局 OUT NAK（OTG_DCTL 中的 SGONAK 位）置 1，模块将停止向接收 FIFO 中写入 SETUP 数据包以外的数据。无论接收 FIFO 中可用空间大小如何，设备都会对主机发送的非同步 OUT 令牌回复 NAK 握手，而对同步 OUT 数据包直接予以忽略。
2. 模块将全局 OUT NAK 写入接收 FIFO。应用程序必须为此留出足够空间。
3. 当应用程序从接收 FIFO 中弹出全局 OUT NAK 字时，模块会将 GONAKEFF 中断 (OTG_GINTSTS) 置 1。
4. 应用程序检测到该中断后，会认为模块处于全局 OUT NAK 模式。应用程序可以通过将 OTG_DCTL 中的 SGONAK 位清零来清除该中断。

应用程序编程顺序：

1. 要停止接收任何类型的数据到接收 FIFO 中，应用程序必须通过编程以下字段以将全局 OUT NAK 位置 1。
   – 在 OTG_DCTL 中，SGONAK = 1
2. 等待 OTG_GINTSTS 中的 GONAKEFF 中断。一旦被触发，该中断表示模块已停止接收 SETUP 数据包以外的任何类型数据。
3. 如果应用程序已将 OTG_DCTL 中的 SGONAK 位置 1，则在模块引发 GONAKEFF 中断 (OTG_GINTSTS) 之前，应用程序可以接收有效 OUT 数据包。
4. 应用程序可通过对 OTG_GINTMSK 寄存器中的 GONAKEFFM 位执行写操作来暂时屏蔽此中断。
   – 在 OTG_GINTMSK 寄存器中，GONAKEFFM = 0
5. 当应用程序准备退出全局 OUT NAK 模式时，必须将 OTG_DCTL 中的 SGONAK 位清零。此操作还会清除 GONAKEFF 中断 (OTG_GINTSTS)。
   – 在 OTG_DCTL 中，CGONAK = 1
6. 如果应用程序在之前已屏蔽此中断，则必须按以下方式使能该中断：
   – 在 OTG_GINTMSK 中，GONAKEFFM = 1

- **禁止 OUT 端点**

应用程序必须使用以下顺序禁止已使能的 OUT 端点。

应用程序编程顺序：

1. 禁止任何 OUT 端点前，应用程序必须在模块中使能全局 OUT NAK 模式。
   – 在 OTG_DCTL 中，SGONAK = 1
2. 等待 GONAKEFF 中断 (OTG_GINTSTS)。
3. 通过编程以下字段来禁止 OUT 端点：
   – 在 OTG_DOEPCTLx 中，EPDIS = 1
   – 在 OTG_DOEPCTLx 中，SNAK = 1
4. 等待 EPDISD 中断 (OTG_DOEPINTx)，该中断表示已完全禁止 OUT 端点。引发 EPDISD 中断时，模块还会将以下位清零：
   – 在 OTG_DOEPCTLx 中，EPDIS = 0
   – 在 OTG_DOEPCTLx 中，EPENA = 0
5. 应用程序必须将全局 OUT NAK 位清零，以开始从其它未禁止的 OUT 端点接收数据。
   – 在 OTG_DCTL 中，SGONAK = 0

- **通用非同步 OUT 数据传输**

本节介绍一种常规非同步 OUT 数据传输（控制、批量或中断）。

应用程序要求：

1. 建立 OUT 传输前，应用程序必须在存储器中分配一个缓冲区，以容纳要作为 OUT 传输的一部分而接收的所有数据。
2. 对于 OUT 传输，端点的传输大小寄存器中的传输大小字段必须是端点的最大数据包大小的倍数（且以字对齐）。
   - 传输大小[EPNUM] = n × (MPSIZ[EPNUM] + 4 – (MPSIZ[EPNUM] mod 4))
   - 数据包计数[EPNUM] = n
   - n > 0
3. 发生 OUT 端点中断时，应用程序必须读取端点的传输大小寄存器以计算存储器中有效数据量。接收的有效数据量可能小于编程的传输大小。
   - 存储器中的有效数据量 = 应用程序设置的初始传输量 – 模块更新后的剩余传输量
   - 接收到 USB 数据包数 = 应用程序设置的初始数据包数 – 模块更新后的剩余数据包数

内部数据流：

1. 应用程序必须在端点相关寄存器中设置传输大小和数据包计数字段，将 NAK 位清零，并使能端点来接收数据。
2. NAK 位清零后，模块便开始接收数据并将数据写入接收 FIFO（只要接收 FIFO 中有空间）。对于 USB 上接收的每个数据包，数据包及其状态都会写入接收 FIFO。写入接收 FIFO 的每个数据包（数据量达到最大数据包大小的数据包或短数据包）都会使该端点的数据包计数字段递减 1。
   - 收到的数据包若 CRC 无效，则自动被从接收 FIFO 中清除。
   - 在 USB 上为数据包回复 ACK 后，模块将丢弃主机因无法检测到 ACK 而重新发送的非同步 OUT 数据包。应用程序不会在具有相同数据 PID 的相同端点上检测到多个连续的 OUT 数据包。在这种情况下，数据包计数不会递减。
   - 如果接收 FIFO 中没有空间，则会忽略同步或非同步数据包并且不会将它们写入接收 FIFO。此外，非同步 OUT 令牌将会收到 NAK 握手应答。
   - 在上述所有三种情况中，数据包计数都不会递减，因为没有任何数据写入接收 FIFO。
3. 当数据包计数变为 0 或者在端点上接收到短数据包时，该端点的 NAK 位将置 1。NAK 位置 1 后，将忽略同步或非同步数据包并且不会将它们写入接收 FIFO，同时非同步 OUT 令牌会收到 NAK 握手应答。
4. 在数据写入接收 FIFO 后，应用程序将从接收 FIFO 中读取数据并将数据写入外部存储器，一次一个数据包，逐个端点过来。
5. 在 AHB 上向外部存储器写入完每个数据包后，端点的传输大小都会自动减去该数据包的大小。
6. 在以下情况时，OUT 端点的 OUT 数据传输完成状态将写入接收 FIFO：
   - 传输大小为 0 并且数据包计数为 0
   - 写入接收 FIFO 的最后一个 OUT 数据包是短数据包（0 ≤ 数据包大小 < 最大数据包大小）
7. 当应用程序弹出此状态条目（OUT 数据传输完成），并生成该端点的传输完成中断，同时清零端点使能位。

应用程序编程顺序：

1. 使用传输大小和相应数据包个数对 OTG_DOEPTSIZx 寄存器进行编程。
2. 使用端点特性对 OTG_DOEPCTLx 寄存器进行编程，并将 EPENA 和 CNAK 位置 1。
   – 在 OTG_DOEPCTLx 中，EPENA = 1
   – 在 OTG_DOEPCTLx 中，CNAK = 1
3. 等待 RXFLVL 中断（在 OTG_GINTSTS 中）并且从接收 FIFO 中读走数据包。
   – 此步骤可重复多次，具体取决于传输大小。
4. 触发 XFRC 中断 (OTG_DOEPINTx)，以表示非同步 OUT 数据传输成功完成。
5. 读取 OTG_DOEPTSIZx 寄存器，以确定有效数据量。

- **通用同步 OUT 数据传输**

本节介绍常规的同步 OUT 数据传输。

应用程序要求：

1. 非同步 OUT 数据传输的所有应用程序要求均适用于同步 OUT 数据传输。
2. 对于同步 OUT 数据传输中的传输大小和数据包计数字段，必须始终将其设置为单个帧中可接收的最大数据包大小的数据包数目。同步类型的 OUT 数据传输事务必须在一个帧内完成。
3. 在周期性帧结束（OTG_GINTSTS 中的 EOPF 中断）之前，应用程序必须从接收 FIFO 中读取所有同步 OUT 数据包（数据条目和状态条目）。
4. 要接收下一帧中的数据，必须在 EOPF (OTG_GINTSTS) 之后 SOF (OTG_GINTSTS) 之前使能一个同步 OUT 端点。

内部数据流：

1. 同步 OUT 端点的内部数据流与非同步 OTU 端点的基本相同，但稍有差异。
2. 同步 OUT 端点通过将端点使能位置 1 并将 NAK 位清零来使能时，必须相应地将偶数/奇数帧位置 1。仅当符合以下条件时，模块才会在同步 OUT 端点上接收特定帧中的数据：
   – EONUM （在 OTG_DOEPCTLx 中）= FNSOF[0]（在 OTG_DSTS 中）
3. 当应用程序从接收 FIFO 中完整地读取一个同步 OUT 数据包（数据和状态）后，模块会根据从接收 FIFO 中读取的最后一个同步 OUT 数据包的数据 PID 更新 OTG_DOEPTSIZx 中的 RXDPID 字段。

应用程序编程顺序：

1. 使用传输大小和相应数据包个数对 OTG_DOEPTSIZx 寄存器进行编程
2. 使用端点特性对 OTG_DOEPCTLx 寄存器进行编程，并将端点使能位、清除 NAK 位和奇数/偶数帧位置 1。
   – EPENA = 1
   – CNAK = 1
   – EONUM =（0：偶数/1：奇数）
3. 等待 RXFLVL 中断（在 OTG_GINTSTS 中）并且从接收 FIFO 中读走数据包。
   – 此步骤可重复多次，具体取决于传输大小。
4. XFRC 中断（在 OTG_DOEPINTx 中）表示同步 OUT 数据传输完成。该中断不一定意味着存储器中的数据是有效的。

5. 对于同步 OUT 传输，应用程序可能并不总会检测到该中断。而是可以检测到 OTG_GINTSTS 中的 INCOMPISOOUT 中断。
6. 读取 OTG_DOEPTSIZx 寄存器以确定接收的传输大小以及帧中所接收数据的有效性。仅当符合以下条件之一时，应用程序才必须将存储器中接收的数据视为有效数据：
   - RXDPID = DATA0（在 OTG_DOEPTSIZx 中）并且接收该有效数据的 USB 数据包数量 = 1
   - RXDPID = DATA1（在 OTG_DOEPTSIZx 中）并且接收该有效数据的 USB 数据包数量 = 2
   - RXDPID = D2（在 OTG_DOEPTSIZx 中）并且接收该有效数据的 USB 数据包数量 = 3[HS]

     接收该有效数据的 USB 数据包数量 =
     应用程序编程的初始数据包个数 - 模块更新后的剩余数据包个数

   应用程序可将无效数据包丢弃。

- **不完整的同步 OUT 数据传输**

本节介绍了同步 OUT 数据包出现丢包时应用程序编程顺序。

内部数据流：

1. 对于同步 OUT 端点，可能并不总是触发 XFRC 中断（在 OTG_DOEPINTx 中）。如果模块丢弃同步 OUT 数据包，则在以下情况下，应用程序可能无法检测到 XFRC 中断 (OTG_DOEPINTx)：
   - 在接收 FIFO 无法容纳完整的 ISO OUT 数据包时，模块将丢弃接收到的 ISO OUT 数据
   - 接收到的同步 OUT 数据包存在 CRC 错误
   - 模块接收到的同步 OUT 令牌损坏
   - 应用程序从接收 FIFO 中读取数据的速度非常缓慢
2. 如果模块在所有同步 OUT 端点的传输完成前检测到周期性帧结束，将触发未完成同步 OUT 数据中断（OTG_GINTSTS 中的 INCOMPISOOUT），指示至少有一个同步 OUT 端点上未触发 XFRC 中断（在 OTG_DOEPINTx 中）。此时，未完成传输的端点仍保持使能，但在 USB 的该端点上，没有进行中的有效传输。

应用程序编程顺序：

1. 硬件触发 INCOMPISOOUT 中断 (OTG_GINTSTS) 表示当前帧中至少有一个同步 OUT 端点具有未完成的传输。
2. 如果因未从端点完全读取同步 OUT 数据而发生这种情况，应用程序必须确保首先从接收 FIFO 读取走所有同步 OUT 数据（包括数据条目和状态条目），然后再继续处理。
   - 从接收 FIFO 读取所有数据后，应用程序即可检测到 XFRC 中断 (OTG_DOEPINTx)。在此情况下，应用程序必须重新使能端点以接收下一个帧中的同步 OUT 数据。
3. 当应用程序接收到 INCOMPISOOUT 中断（在 OTG_GINTSTS 中）时，应用程序必须读取所有同步 OUT 端点的控制寄存器 (OTG_DOEPCTLx)，以确定哪些端点在当前微帧中具有不完整的传输。同时满足以下两个条件时，表示端点传输未完成：
   - EONUM 位（在 OTG_DOEPCTLx 中）= FNSOF[0]（在 OTG_DSTS 中）
   - EPENA = 1（在 OTG_DOEPCTLx 中）
4. 在检测到 SOF 中断（在 OTG_GINTSTS 中）前，必须执行完成上一步操作，以确保当前帧编号未发生更改。
5. 对于具有未完成传输的同步 OUT 端点，应用程序必须丢弃存储器中的数据，并通过将 OTG_DOEPCTLx 中的 EPDIS 位置 1 来禁止该端点。

6. 等待 EPDISD 中断（在 OTG_DOEPINTx 中），并使能该端点，以便接收下一个帧中的新数据。
   – 由于模块可能需要一些时间才能禁止端点，因此应用程序在接收到无效同步数据后，可能无法接收下一个帧中的数据。

- **停止非同步 OUT 端点**

本节介绍应用程序如何才能停止非同步端点。

1. 将模块置于全局 OUT NAK 模式。
2. 禁止所需的端点：
   – 禁止端点时，请设置 STALL = 1（在 OTG_DOEPCTL 中），而不是将 OTG_DOEPCTL 中的 SNAK 位置 1。

     STALL 位的优先级始终高于 NAK 位。
3. 当应用程序不再需要端点回复 STALL 握手信号时，必须将 STALL 位（在 OTG_DOEPCTLx 中）清零。
4. 如果应用程序由于收到主机的 SetFeature.Endpoint Halt 或 ClearFeature.Endpoint Halt 命令来设置或清除端点的 STALL 状态，则必须在该控制端点上的状态阶段传输前，将 STALL 位置 1 或清零。

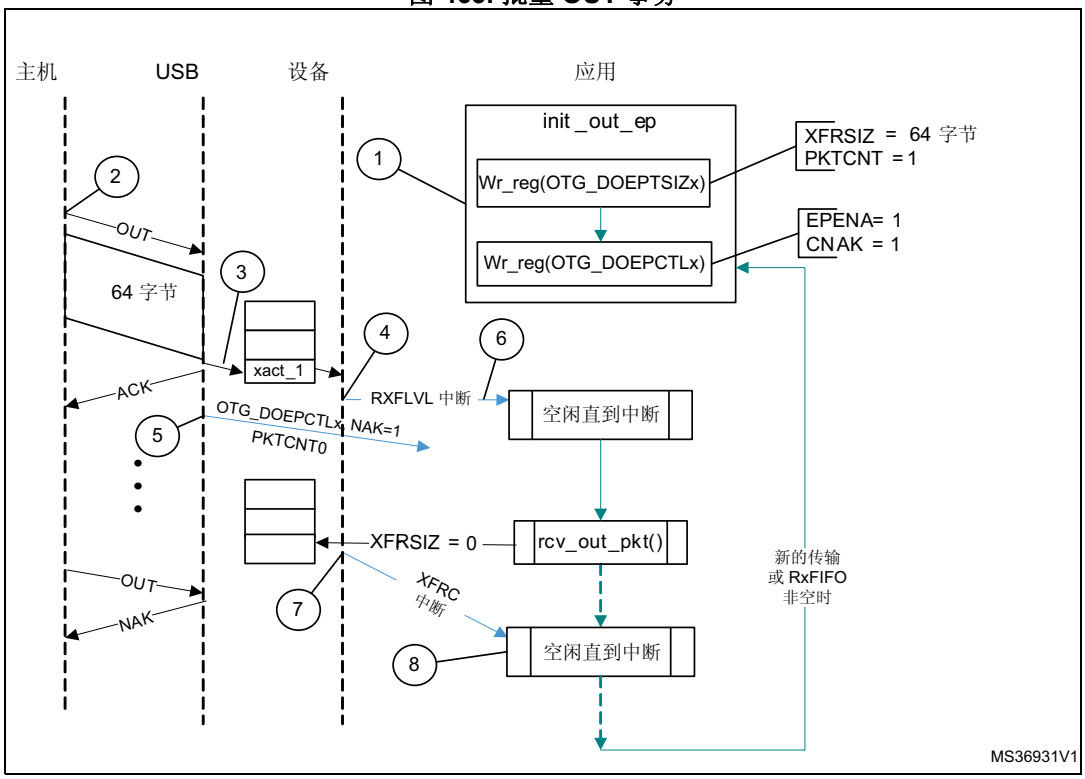
示例

本节介绍并描述了一些基本的传输类型和情景。

- 批量 OUT 事务

图 463 描述了将单个批量 OUT 数据包从 USB 接收到 AHB 中的过程，并介绍了这一过程中涉及到的事件。

图 463. 批量 OUT 事务



接收到 SetConfiguration/SetInterface 命令后，应用程序将初始化所有 OUT 端点：将 CNAK 和 EPENA 置 1（在 OTG_DOEPCTLx 中），并将 OTG_DOEPTSIZx 寄存器中的 XFRSIZ 和 PKTCNT 设置成合适的值。

1. 主机尝试将数据（OUT 令牌）发送到端点。
2. 当模块在 USB 上接收到 OUT 令牌时，模块会将数据包存储在 Rx FIFO 中，因为其中有可用空间。
3. 在 Rx FIFO 中写入完整数据包后，模块随后会触发 RXFLVL 中断（在 OTG_GINTSTS 中）。
4. 接收到 PKTCNT 所指定数量的 USB 数据包后，模块在内部将该端点的 NAK 位置 1，以避免其再接收任何数据包。
5. 应用程序处理中断并从 Rx FIFO 中读取数据。
6. 应用程序读取完所有数据后（相当于 XFRSIZ），模块将生成 XFRC 中断（在 OTG_DOEPINTx 中）。
7. 应用程序处理中断并通过 XFRC 中断位（在 OTG_DOEPINTx 中）的触发得知本次传输完成。

IN 数据传输

- **数据包写入**

本节介绍在已使能专用发送 FIFO 的情况下应用程序如何将数据包写入端点 FIFO。

1. 应用程序可以选择轮询模式或中断模式。
   – 在轮询模式下，应用程序通过读取 OTG_DTXFSTSx 寄存器来监视端点发送数据 FIFO 的状态，从而确定该数据 FIFO 中是否有足够空间。
   – 在中断模式中，应用程序等待 TXFE 中断（在 OTG_DIEPINTx 中），然后读取 OTG_DTXFSTSx 寄存器，以确定数据 FIFO 中是否有足够空间。
   – 要写入单个非零长度的数据包，数据 FIFO 中必须有足够的空间来容纳整个数据包。
   – 要写入零长度的数据包，应用程序不能查看 FIFO 空间。
2. 如果使用上述方法之一，当应用程序确定有足够空间来写入发送数据包时，应用程序必须首先对端点控制寄存器进行相应写操作，然后再将数据写入数据 FIFO。通常，应用程序必须对 OTG_DIEPCTLx 寄存器执行读-修改-写操作，以避免在将端点使能位置 1 的同时，修改寄存器中的其它内容。

如果有足够空间，应用程序可将同一端点的多个数据包写入发送 FIFO。对于周期性 IN 端点，应用程序只能一次写入一个微帧内的多个数据包。只有先前一个微帧的通信事务传输完成之后，应用程序才会写入下一个微帧内要发送的所有数据包。

- **将 IN 端点 NAK 置 1**

内部数据流：

1. 当应用程序将特定端点的 IN NAK 置 1 时，模块将停止端点上的数据发送，而不考虑端点发送 FIFO 中的数据是否可用。
2. 非同步端点收到 IN 令牌，回复 NAK 握手应答。
   – 同步端点收到 IN 令牌，返回零长度数据包
3. 模块在 OTG_DIEPINTx 中触发 INEPNE（IN 端点 NAK 有效）中断以响应 OTG_DIEPCTLx 中的 SNAK 位。
4. 应用程序检测到该中断后，便会认为端点处于 IN NAK 模式。应用程序可以通过将 OTG_DIEPCTLx 中的 CNAK 位置 1 来清除该中断。

应用程序编程顺序：

1. 要在特定 IN 端点上停止发送任何数据，应用程序必须将 IN NAK 位置 1。要将该位置 1，必须编程以下字段。
   – 在 OTG_DIEPCTLx 中，SNAK = 1
2. 等待 OTG_DIEPINTx 中的 INEPNE 中断触发。该中断表示模块已在端点上停止发送数据。
3. 在应用程序将 NAK 位置 1 但“NAK 有效”中断尚未触发时，模块可以在端点上发送有效 IN 数据。
4. 应用程序可通过写入 OTG_DIEPMSK 中的 INEPNEM 位来临时屏蔽该中断。
   – 在 OTG_DIEPMSK 中，INEPNEM = 0
5. 要退出端点 NAK 模式，应用程序必须将 OTG_DIEPCTLx 中的 NAK 状态位 (NAKSTS) 清零。此操作还会将 INEPNE 中断位（在 OTG_DIEPINTx 中）清零。
   – 在 OTG_DIEPCTLx 中，CNAK = 1
6. 如果应用程序已将该中断屏蔽，则必须按以下方式使能：
   – 在 OTG_DIEPMSK 中，INEPNEM = 1

- **禁止 IN 端点**

使用以下顺序来禁止先前已使能的特定 IN 端点。

应用程序编程顺序：

1. 应用程序必须先停止在 AHB 上写入数据，之后才能禁止 IN 端点。
2. 应用程序必须将端点设置为 NAK 模式。
   – 在 OTG_DIEPCTLx 中，SNAK = 1
3. 等待 OTG_DIEPINTx 中的 INEPNE 中断。
4. 将必须禁止的端点的 OTG_DIEPCTLx 寄存器中的以下位置 1。
   – 在 OTG_DIEPCTLx 中，EPDIS = 1
   – 在 OTG_DIEPCTLx 中，SNAK = 1
5. OTG_DIEPINTx 中的 EPDISD 中断的触发表示模块已完全禁止指定的端点。在触发中断的同时，模块还会将以下位清零：
   – 在 OTG_DIEPCTLx 中，EPENA = 0
   – 在 OTG_DIEPCTLx 中，EPDIS = 0
6. 应用程序必须读取周期性 IN EP 的 OTG_DIEPTSIZx 寄存器，以计算该端点在 USB 上发送的数据量。
7. 应用程序必须通过将 OTG_GRSTCTL 寄存器中的以下字段置 1，来清空端点发送 FIFO 中的数据：
   – TXFNUM（在 OTG_GRSTCTL 中）= 端点发送 FIFO 编号
   – TXFFLSH（在 OTG_GRSTCTL 中）= 1

应用程序必须轮询 OTG_GRSTCTL 寄存器，直至模块将 TXFFLSH 位清零，这表示 FIFO 清空操作结束。要在该端点上发送新数据，应用程序可以稍后重新使能该端点。

- **通用非周期性 IN 数据传输**

应用程序要求：

1. 建立 IN 传输前，应用程序必须确保组成一次 IN 传输的每个数据包都可以容纳在单个缓冲区中。
2. 对于 IN 传输，端点传输大小寄存器中的传输大小字段表示本次传输的有效数据量，它由多个最大数据包大小和单个短数据包组成。该短数据包在传输结束时发送。
   – 要发送多个最大数据包大小的数据包并在传输结束时外加一个短数据包：
     传输大小[EPNUM] = x × MPSIZ[EPNUM] + sp
     如果 (sp > 0)，数据包计数[EPNUM] = x + 1。
     否则，数据包计数[EPNUM] = *x*
   – 要发送单个零长度数据包：
     传输大小[EPNUM] = 0
     数据包计数[EPNUM] = 1
   – 要发送多个最大数据包大小的数据包并在传输结束时外加一个零长度数据包，应用程序必须将传输拆分为两个部分。第一部分发送最大数据包大小的数据包，第二部分仅发送零长度数据包。
     第一次传输：传输大小[EPNUM] = *x* × MPSIZ[epnum]；数据包计数 = *n*；
     第二次传输：传输大小[EPNUM] = 0；数据包计数 = 1；

3. 使能某个端点进行数据传输后，模块会更新传输大小寄存器。在 IN 传输结束时，应用程序必须读取传输大小寄存器，以确定送入发送 FIFO 中的数据已有多少通过 USB 发送出去。
4. 送入发送 FIFO 中的数据量 = 应用程序编程的初始传输大小 - 模块更新后的最终传输大小
   – 通过 USB 已经发送的数据量 = (应用程序编程的初始数据包计数 - 模块更新后的最终数据包计数) × MPSIZ[EPNUM]
   – 要通过 USB 发送的剩余数据量 = (应用程序编程的初始传输大小 - 已通过 USB 发送的数据量)

内部数据流：

1. 应用程序必须在特定端点的寄存器中设置传输大小和数据包计数字段，并使能该端点来发送数据。
2. 应用程序还必须向该端点的发送 FIFO 写入必需的数据。
3. 应用程序每向发送 FIFO 写入一个数据包，该端点的传输大小便会自动减去该数据包大小。应用程序持续从存储器获取数据来写入发送 FIFO，直到该端点的传输大小变为 0。向 FIFO 写入数据后，“FIFO 中的数据包数”计数会递增（这是一个 3 位计数，由模块在内部进行维护，每个 IN 端点发送 FIFO 对应一个。在 IN 端点 FIFO 中，模块所维护的最大数据包数始终为八个）。对于零长度数据包，每个 FIFO 均另有一个单独的标志，FIFO 中没有任何数据。
4. 当数据写入发送 FIFO 后，模块会在接收到 IN 令牌时将这些数据送出。每个非同步 IN 数据包发送出去并收到回复的 ACK 握手信号后，该端点的数据包计数都会递减 1，直到数据包计数变 0 为止。发生超时时，数据包计数不会递减。
5. 对于零长度数据包（由内部零长度标志指示），模块会针对 IN 令牌发出一个零长度数据包，并递减数据包计数字段的值。
6. 如果接收到 IN 令牌的端点对应的 FIFO 中无数据，且该端点的数据包计数字段为零，则模块会针对该端点生成一个“Tx FIFO 为空时接收到 IN 令牌”(ITTXFE) 中断（前提是该端点的 NAK 位未置 1）。模块在该非同步端点上回复 NAK 握手信号。
7. 模块会在内部使 FIFO 指针重新返回到开头，并且不会生成超时中断。
8. 当传输大小为 0 且数据包计数为 0 时，将生成该端点的传输完成 (XFRC) 中断，同时将端点使能清零。

应用程序编程顺序：

1. 使用传输大小和相应数据包计数对 OTG_DIEPTSIZx 寄存器进行编程。
2. 使用端点特性对 OTG_DIEPCTLx 寄存器进行编程，并将 CNAK 和 EPENA（端点使能）位置 1。
3. 发送非零长度数据包时，应用程序必须轮询 OTG_DTXFSTSx 寄存器（其中 x 为与该端点相关联的 FIFO 编号），以确定数据 FIFO 中是否有足够的空间。写入数据前，应用程序可以选择使用 TXFE（在 OTG_DIEPINTx 中）。

- **通用周期性 IN 数据传输**

本节介绍典型的周期性 IN 数据传输。

应用程序要求：

1. *第 1337 页上的通用非周期性 IN 数据传输*的应用程序要求 1、2、3 和 4 对周期性 IN 数据传输同样适用（只是对要求 2 稍加修改）。
   - 应用程序只能发送若干个最大数据包大小的数据包或若干个最大数据包大小的包，外加传输结束时的一个短数据包。要发送多个最大数据包大小的数据包并在传输结束时外加一个短数据包，必须满足以下条件：

     传输大小[EPNUM] = x × MPSIZ[EPNUM] + sp
     （其中 x 是 ≥ 0 的整数，且 0 ≤ sp < MPSIZ[EPNUM]）

     如果 (sp > 0)，数据包计数[EPNUM] = *x* + 1
     否则，数据包计数[EPNUM] = *x*；

     MCNT[EPNUM] = 数据包计数[EPNUM]
   - 应用程序无法在传输结束时发送零长度数据包。应用程序可以单独发送一个零长度数据包。要发送单个零长度数据包：
   - 传输大小[EPNUM] = 0

     数据包计数[EPNUM] = 1

     MCNT[EPNUM] = 数据包计数[EPNUM]
2. 应用程序一次只能安排一帧的数据传输。
   - (MCNT – 1) × MPSIZ ≤ XFERSIZ ≤ MCNT × MPSIZ
   - PKTCNT = MCNT（在 OTG_DIEPTSIZx 中）
   - 如果 XFERSIZ < MCNT × MPSIZ，则传输的最后一个数据包为短数据包。
   - 注意：MCNT 在 OTG_DIEPTSIZx 中，MPSIZ 在 OTG_DIEPCTLx 中，PKTCNT 在 OTG_DIEPTSIZx 中，而 XFERSIZ 在 OTG_DIEPTSIZx 中
3. 接收到 IN 令牌前，应用程序必须将要在帧中发送的完整数据写入到发送 FIFO 中。在接收到 IN 令牌时，即使发送 FIFO 中该帧要发送的数据只差 1 个双字未写进来，模块也会执行 FIFO 为空时的操作。当发送 FIFO 为空时：
   - 同步端点上将回复零长度数据包
   - 中断端点上将回复 NAK 握手信号

内部数据流：

1. 应用程序必须在特定端点的寄存器中设置传输大小和数据包计数字段，并使能该端点来发送数据。
2. 应用程序还必须向与该端点相关联的发送 FIFO 写入必需的数据。
3. 应用程序每向发送 FIFO 写入一个数据包，该端点的传输大小便会自动减去该数据包大小。应用程序持续从存储器获取数据来写入发送 FIFO，直到该端点的传输大小变为 0。
4. 当周期性端点接收到 IN 令牌时，模块将开始发送 FIFO 中的数据（如果 FIFO 中有数据）。如果 FIFO 中没有该帧要发送的数据的完整数据包（在专用 FIFO 模式下），则模块将为该端点生成一个“Tx FIFO 为空时接收到 IN 令牌”中断。
   - 同步端点上将回复零长度数据包
   - 中断端点上将回复 NAK 握手信号
5. 端点的数据包计数会在下列情况下递减 1：
   - 对于同步端点，发送一个零长度或非零长度的数据包时
   - 对于中断端点，在发送 ACK 握手信号时递减
   - 当传输大小和数据包计数均为 0 时，将生成该端点的传输完成中断，同时将端点使能位清零。
6. 在“周期性帧间隔”（由 OTG_DCFG 中的 PFIVL 位控制）内，当模块发现任何在当前帧内应为空的同步 IN 端点 FIFO 中的数据还未发送完成时，都会在 OTG_GINTSTS 中生成一个 IISOIXFR 中断。

应用程序编程顺序：

1. 使用端点特性对 OTG_DIEPCTLx 寄存器进行编程，并将 CNAK 和 EPENA 位置 1。
2. 将需要在下一帧中发送的数据写入发送 FIFO。
3. 硬件触发 ITTXFE 中断（在 OTG_DIEPINTx 中）表示应用程序尚未将需要发送的全部数据写入发送 FIFO。
4. 如果在检测到中断前已使能中断端点，则将忽略该中断。如果中断端点未使能，则使能此端点，以便数据能够在收到下一次 IN 令牌时发送出去。
5. 硬件触发 XFRC 中断（在 OTG_DIEPINTx 中）时如果 OTG_DIEPINTx 中未产生 ITTXFE 中断，则表示成功完成同步 IN 传输。读取 OTG_DIEPTSIZx 寄存器时始终得到传输大小 = 0 且数据包计数 = 0，表示所有数据都已通过 USB 发送完毕。
6. 无论是否产生 ITTXFE 中断（在 OTG_DIEPINTx 中），只要触发 XFRC 中断（在 OTG_DIEPINTx 中），即表示中断 IN 传输成功完成。读取 OTG_DIEPTSIZx 寄存器时始终得到传输大小 = 0 且数据包计数 = 0，表示所有数据都已通过 USB 发送完毕。
7. 如果在未产生任何前述中断的情况下在 OTG_GINTSTS 中触发未完成同步 IN 传输 (IISOIXFR) 中断，则表示模块在当前帧中至少未收到 1 个周期性 IN 令牌。

- **未完成同步 IN 数据传输**

本节介绍应用程序针对未完成同步 IN 数据传输必须执行的操作。

内部数据流：

1. 符合下列条件之一时，即认为同步 IN 传输未完成：
   a) 模块在至少一个同步 IN 端点上接收到损坏的同步 IN 令牌。此时，应用程序检测到未完成同步 IN 传输中断（OTG_GINTSTS 中的 IISOIXFR 位）。
   b) 应用程序向发送 FIFO 写入数据的速度过慢，在将完整数据写入 FIFO 之前便接收到 IN 令牌。此时，应用程序在 OTG_DIEPINTx 中检测到“Tx FIFO 为空时接收到 IN 令牌”中断。应用程序可忽略此中断，因为最终将在周期性帧结束时产生一个未完成同步 IN 传输中断（OTG_GINTSTS 中的 IISOIXFR 位）。

   模块会通过 USB 发送一个零长度数据包来响应接收到的 IN 令牌。
2. 应用程序必须尽快停止向发送 FIFO 写入数据。
3. 应用程序必须将端点的 NAK 位和禁止位置 1。
4. 模块会禁止该端点，将禁止位清零并触发端点的“端点禁止”中断。

应用程序编程顺序：

1. 应用程序可以在任何同步 IN 端点上忽略 OTG_DIEPINTx 中的“Tx FIFO 为空时接收到 IN 令牌”中断，因为最终这将产生一个未完成同步 IN 传输中断（在 OTG_GINTSTS 中）。
2. 硬件触发未完成同步 IN 传输中断（在 OTG_GINTSTS 中）表示在至少一个同步 IN 端点上存在未完成的同步 IN 传输。
3. 应用程序必须读取所有同步 IN 端点的“端点控制”寄存器来检测存在未完成 IN 数据传输的端点。
4. 应用程序必须停止向与这些端点相关联的“周期性发送 FIFO”写入数据。
5. 对 OTG_DIEPCTLx 寄存器中的下列字段进行编程以禁止端点：
   – 在 OTG_DIEPCTLx 中，SNAK = 1
   – 在 OTG_DIEPCTLx 中，EPDIS = 1
6. 硬件触发 OTG_DIEPINTx 中的“端点禁止”中断表示模块已禁止该端点。
   – 此时，应用程序必须清空相关联的发送 FIFO 中的数据，或者通过在下一微帧中使能新传输的端点来覆盖 FIFO 中的现有数据。要刷新数据，应用程序必须使用 OTG_GRSTCTL 寄存器。

- **停止非同步 IN 端点**

本节介绍应用程序如何才能停止非同步端点。

应用程序编程顺序：

1. 禁止要停止的 IN 端点。同时将 STALL 位置 1。
2. OTG_DIEPCTLx 中的 EPDIS = 1（当端点已使能时）
   – OTG_DIEPCTLx 中的 STALL = 1
   – STALL 位的优先级始终高于 NAK 位
3. 硬件触发“端点禁止”中断（在 OTG_DIEPINTx 中），应用程序获知模块已禁止指定的端点。
4. 应用程序必须根据端点类型清空非周期性或周期性发送 FIFO。对于非周期性端点，应用程序必须重新使能另一个无需停止的非周期性端点来发送数据。

5. 当应用程序准备好结束该端点的 STALL 握手信号时，必须将 OTG_DIEPCTLx 的 STALL 位清零。
6. 如果应用程序因收到来自主机的 SetFeature.Endpoint Halt 命令或 ClearFeature.Endpoint Halt 命令来设置或清除端点的 STALL 状态，则必须在该控制端点的状态阶段传输之前将 STALL 位置 1 或清零。

特例：停止控制 OUT 端点

如果在控制传输的数据阶段，主机发送的 IN/OUT 令牌数超过 SETUP 数据包指定的值，则模块必须对这些多余的 IN/OUT 令牌回复 STALL。在这种情况下，在控制传输的数据阶段，当模块传输完 SETUP 数据包指定的数据量后，应用程序必须使能 OTG_DIEPINTx 中的 ITTXFE 中断和 OTG_DOEPINTx 中的 OTEPDIS 中断。随后，当应用程序收到此中断时，必须将相应端点控制寄存器中的 STALL 位置 1 并清除此中断。

37.16.7 最坏情况下的响应时间

当 OTG_FS/OTG_HS 控制器作为设备使用时，对于任何跟随在同步 OUT 之后的令牌，都存在一个最坏响应时间。这个最坏情况响应时间随 AHB 时钟频率的不同而异。

模块寄存器位于 AHB 域内，在更新这些寄存器值之前，模块不会接受新令牌。对于任何跟随在同步 OUT 之后的令牌，最坏的情况是：由于同步事务不需要握手信号，所以下一个令牌可能很快就到达。当 AHB 与 PHY 时钟频率相同时，这个最坏情况值为 7 个 PHY 时钟。AHB 时钟越快，此值越小。

如果发生这种最坏情况，模块将以 NAK 响应批量/中断令牌，并丢弃同步和 SETUP 令牌。对于 SETUP，主机会将这种情况视为超时，并尝试重新发送 SETUP 数据包。对于同步传输，会产生未完成同步 IN 传输中断 (IISOIXFR) 和未完成同步 OUT 传输中断 (IISOOXFR)，来通知应用程序同步 IN/OUT 数据包被丢弃。

选择 OTG_GUSBCFG 中的 TRDT 的值

TRDT 的值 (OTG_GUSBCFG) 是指在接收到 IN 令牌后以 PHY 时钟计算的时间长度，MAC 将在这段时间内获取 FIFO 状态并从 PFC 模块获取第一个数据。这段时间包含 PHY 和 AHB 时钟之间的同步延迟。最坏情况延迟是当 AHB 时钟与 PHY 时钟相等时的延迟。这种情况下的延迟为 5 个时钟周期。

MAC 接收到 IN 令牌后，此信息（接收到的令牌）将由 PFC（PFC 以 AHB 时钟运行）同步到 AHB 时钟。随后，PFC 从 SPRAM 中读取数据并将它们写入双时钟源缓冲区。MAC 再从源缓冲区读出数据（4 个深度）。

如果 AHB 的运行频率高于 PHY，应用程序可以使用较小的 TRDT 值（在 OTG_GUSBCFG 中）。

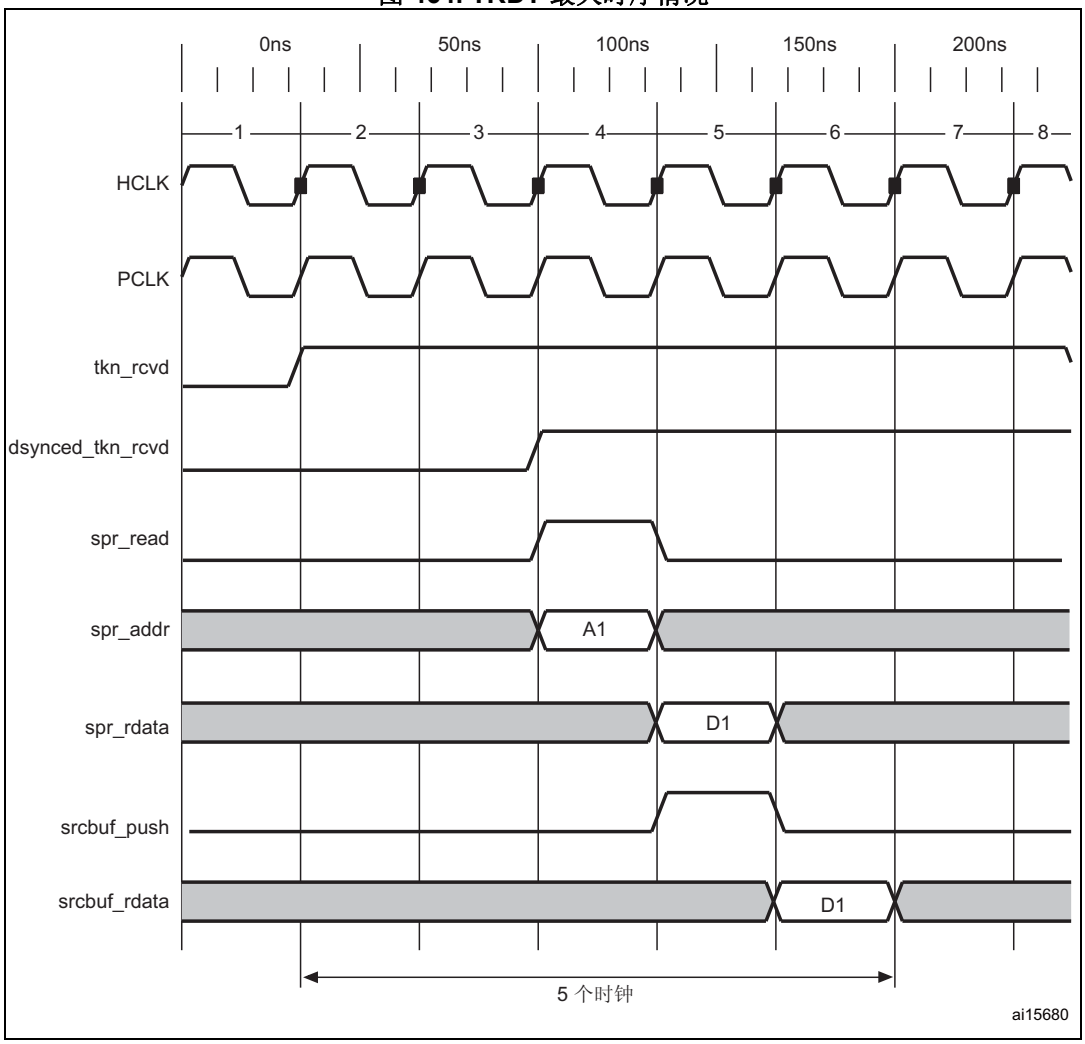
图 464 具有以下信号：

- tkn_rcvd：接收到令牌信息（从 MAC 到 PFC）
- dynced_tkn_rcvd：双倍同步 tkn_rcvd（从 PCLK 到 HCLK 域）
- spr_read：读取到 SPRAM
- spr_addr：寻址到 SPRAM
- spr_rdata：从 SPRAM 中读取数据
- srcbuf_push：压入源缓冲区
- srcbuf_rdata：从源缓冲区读取数据。MAC 看到的数据

应用程序可以使用下列公式来计算 TRDT 的值：

4 × AHB 时钟 + 1 个 PHY 时钟 = (2 个时钟进行同步 + 1 个时钟进行存储器寻址 + 1 个时钟从同步 RAM 读取存储器数据) + 1 个 PHY 时钟（下一个 PHY 时钟 MAC 可以对 2 个时钟 FIFO 输出进行采样）

图 464. TRDT 最大时序情况



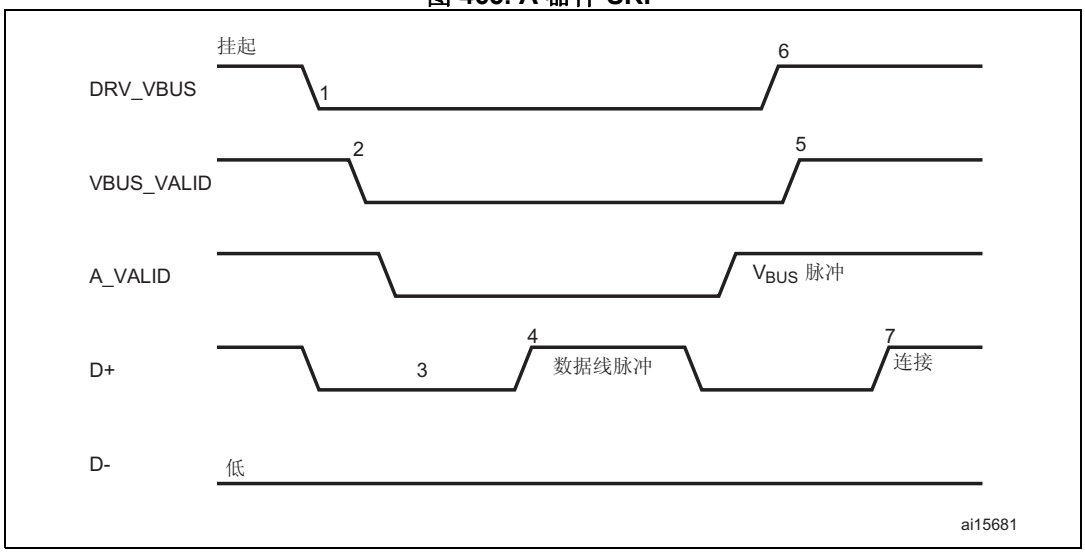
37.16.8 OTG 编程模型

OTG_FS/OTG_HS 控制器是一种支持 HNP 和 SRP 的 OTG 设备。该模块接口与“A 型”插头连接时，该模块称为 A 器件。该模块接口与“B 型”插头连接时，该模块称为 B 器件。在主机模式下，OTG_FS/OTG_HS 控制器将关闭 V_{BUS} 以节省电能。B 器件可以借助 SRP 请求 A 器件打开 V_{BUS} 电源。设备必须同时执行数据线脉冲和 V_{BUS} 脉冲，但主机可以检测数据线脉冲或者 V_{BUS} 脉冲中的一个用于 SRP。B 器件可以借助 HNP 协商并切换到主机角色。在协商模式下执行 HNP 后，B 器件会挂起总线并恢复到设备角色。

A 器件会话请求协议

应用程序必须将模块 USB 配置寄存器中的 SRP 使能位置 1。这将使能 OTG_FS/OTG_HS 控制器在 A 器件模式下检测到 SRP。

图 465. A 器件 SRP



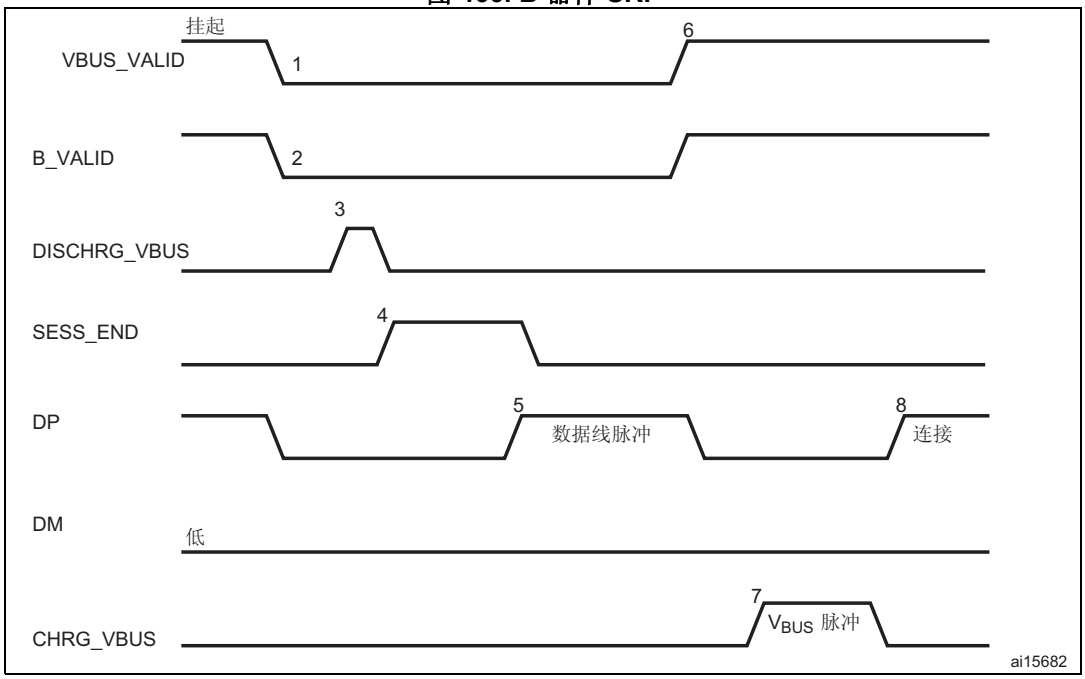
1. DRV_VBUS = 发送到 PHY 的 V_{BUS} 驱动信号
   VBUS_VALID = 来自 PHY 的 V_{BUS} 有效信号
   A_VALID = 发送到 PHY 的 A 器件 V_{BUS} 电平信号
   D+ = 正向数据线
   D- = 负向数据线

1. 为节省电能，在总线空闲时，应用程序会通过在主机端口控制和状态寄存器中写入端口挂起位和端口电源位来挂起设备并关闭端口电源。
2. PHY 通过禁止 VBUS_VALID 信号来指示端口电源已关闭。
3. 当 V_{BUS} 电源关闭时，设备必须检测到至少 2 ms 的 SE0 信号才可以启动 SRP。
4. 要启动 SRP，设备需要打开其数据线的上拉电阻并维持 5 到 10 ms。OTG_FS/OTG_HS 控制器会检测到数据线脉冲。
5. 设备会驱动 V_{BUS} 到 A 器件会话有效电平（最小 2.0 V）以上，以产生 V_{BUS} 脉冲。
   OTG_FS/OTG_HS 控制器检测到 SRP 时将中断应用程序。在全局中断状态寄存器中，检测到会话请求位（OTG_GINTSTS 中的 SRQINT 位）将置 1。
6. 应用程序必须响应“检测到会话请求”中断，并通过在主机端口控制和状态寄存器中写入端口电源位来打开端口电源位。PHY 通过输出 VBUS_VALID 信号来指示端口已通电。
7. 当 USB 通电后，设备将连接，从而完成 SRP 过程。

B 器件会话请求协议

应用程序必须将模块 USB 配置寄存器中的 SRP 使能位置 1。这将使能 OTG_FS/OTG_HS 控制器在 B 器件模式下启动 SRP。OTG_FS/OTG_HS 控制器可以借助 SRP 向主机请求新会话。

图 466. B 器件 SRP



1. VBUS_VALID = 来自 PHY 的 V_{BUS} 有效信号
   B_VALID = 发送到 PHY 的 B 器件有效会话
   DISCHRG_VBUS = 发送到 PHY 的放电信号
   SESS_END = 发送到 PHY 的会话结束信号
   CHRG_VBUS = 发送到 PHY 的 V_{BUS} 充电信号
   DP = 正向数据线
   DM = 负向数据线

1. 为节省电能，在总线空闲时主机会挂起并关闭端口电源。

   OTG_FS/OTG_HS 控制器会在总线空闲 3 ms 后，将模块中断寄存器中的早期挂起位置 1。随后，OTG_FS/OTG_HS 控制器会将模块中断寄存器中的 USB 挂起位置 1。

   OTG_FS/OTG_HS 控制器会通知 PHY 对 V_{BUS} 进行放电。
2. PHY 指示会话结束。这是 SRP 的初始条件。启动 SRP 之前，OTG_FS/OTG_HS OTG_FS 控制器需要 2 ms 的 SE0 信号。

   对于 USB 1.1 全速串行收发器，应用程序必须在 BSVLD 位（位于 OTG_GOTGCTL）被禁止后，等待 V_{BUS} 放电至 0.2 V。此放电时间可从收发器供应商处获取，该值可能因收发器而异。
3. OTG_FS/OTG_HS 模块通知 PHY 对 V_{BUS} 加速放电。
4. 应用程序通过将 OTG 控制和状态寄存器中的会话请求位置 1 来启动 SRP。
   OTG_FS/OTG_HS 控制器先执行数据线脉冲，随后执行 V_{BUS} 脉冲。
5. 主机会从数据线脉冲或 V_{BUS} 脉冲检测到 SRP，然后打开 V_{BUS}。PHY 指示 V_{BUS} 对设备上电。

6. OTG_FS/OTG_HS 控制器执行 V_{BUS} 脉冲。

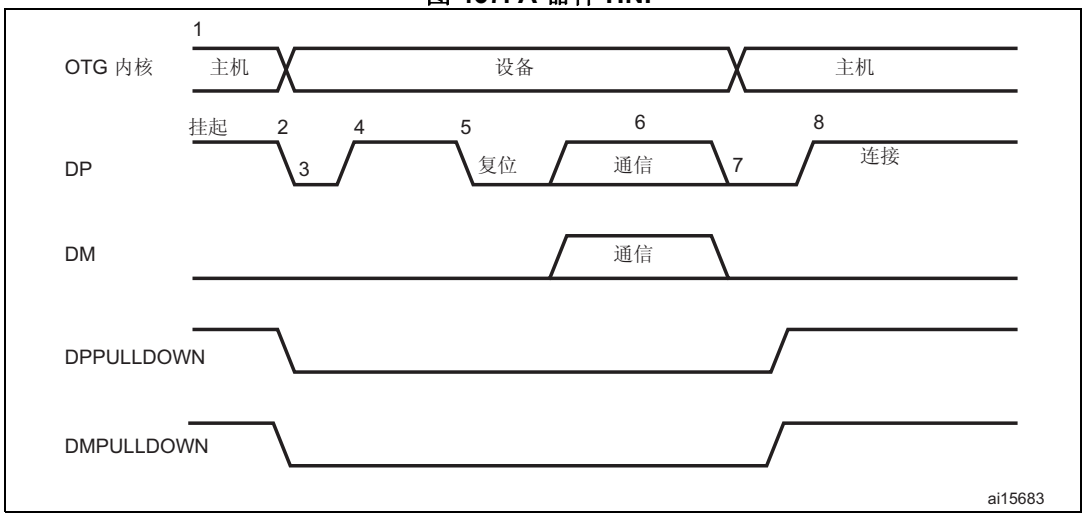
   主机通过打开 V_{BUS} 启动一个新会话，指示 SRP 已成功。OTG_FS/OTG_HS 控制器通过将 OTG 中断状态寄存器中的会话请求成功状态更改位置 1 来中断应用程序。应用程序读取 OTG 控制和状态寄存器中的会话请求成功位。

7. 当 USB 通电时，OTG_FS/OTG_HS 控制器将进行连接，并完成 SRP 过程。

A 器件主机协商协议

通过 HNP 可以将 USB 主机角色从 A 器件切换到 B 器件。应用程序必须将模块 USB 配置寄存器中的 HNP 使能位置 1，以使能 OTG_FS/OTG_HS 控制器在 A 器件模式下执行 HNP 功能。

图 467. A 器件 HNP



1. DPPULLDOWN = 从模块发送到 PHY 的信号，用于使能/禁止 PHY 内部 DP 线的下拉电阻。
   DMPULLDOWN = 从模块发送到 PHY 的信号，用于使能/禁止 PHY 内部 DM 线的下拉电阻。

1. OTG_FS/OTG_HS 控制器向 B 器件发送 SetFeature b_hnp_enable 描述符以使能 HNP 支持。B 器件的 ACK 响应表示 B 器件支持 HNP。应用程序必须将 OTG 控制和状态寄存器中的主机设置 HNP 使能位置 1，以向 OTG_FS/OTG_HS 控制器指示 B 器件支持 HNP。
2. 应用程序不再使用总线时，会通过在主机端口控制和状态寄存器中写入端口挂起位来挂起总线。
3. B 器件检测到 USB 挂起时，将断开连接，指示 HNP 的初始条件。B 器件只有在切换到主机角色时才会启动 HNP，否则总线会保持挂起状态。

   OTG_FS/OTG_HS 控制器将 OTG 中断状态寄存器中的检测到主机协商中断置 1，指示 HNP 已开始。

   OTG_FS/OTG_HS 控制器禁止 PHY 中的 DP 和 DM 下拉电阻，以指示设备角色。PHY 使能 OTG_DP 上拉电阻，以告知 B 器件 A 器件的连接。

   应用程序必须读取 OTG 控制和状态寄存器中的当前模式位，以确定设备工作模式。
4. B 器件检测到连接，发出 USB 复位信号，并枚举 OTG_FS/OTG_HS 控制器进行数据通信。

5. B 器件继续担当主机角色，发起数据通信，并在结束时挂起总线。

   OTG_FS/OTG_HS 控制器会在总线空闲 3 ms 后，将模块中断寄存器中的早期挂起位置 1。随后，OTG_FS/OTG_HS 控制器会将模块中断寄存器中的 USB 挂起位置 1。

6. 在协商模式下，OTG_FS/OTG_HS 控制器会检测到总线挂起，连接断开，然后切换回主机角色。OTG_FS/OTG_HS 控制器将使能 PHY 中的 DM 和 DM 下拉电阻，以指示其承担主机角色。

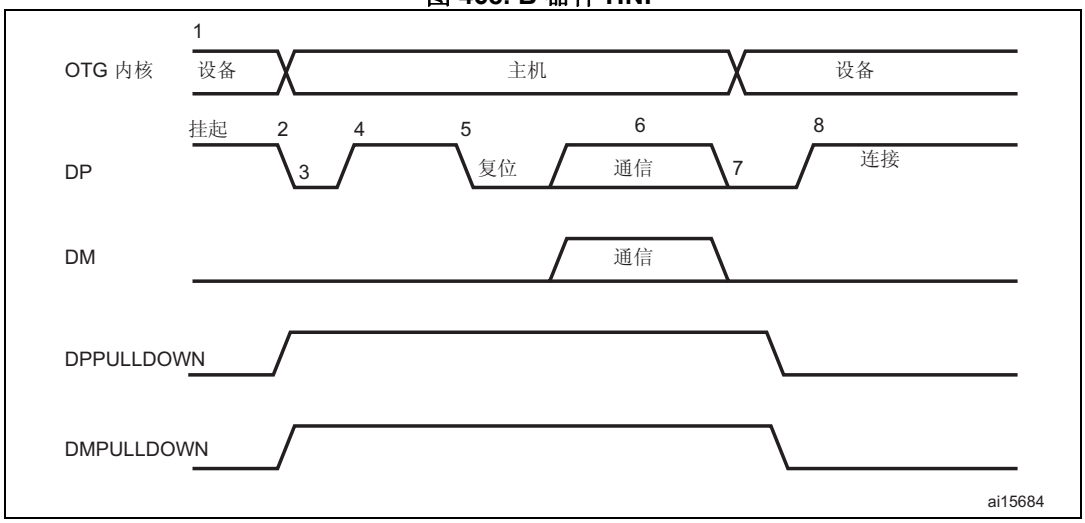
7. OTG_FS/OTG_HS 控制器将 OTG 中断状态寄存器中的连接器 ID 状态更改中断置 1。应用程序必须读取 OTG 控制和状态寄存器中的连接器 ID 状态位，以确定 OTG_FS/OTG_HS 控制器在 A 器件模式下工作。这向应用程序表示 HNP 已经完成。应用程序必须读取 OTG 控制和状态寄存器中的当前模式位，以确定主机工作模式。

8. 连接 B 器件，完成 HNP 过程。

B 器件主机协商协议

通过 HNP 可以将 USB 主机角色从 B 器件切换到 A 器件。应用程序必须将模块 USB 配置寄存器中的 HNP 使能位置 1，以使能 OTG_FS/OTG_HS 控制器在 B 器件模式下执行 HNP 功能。

图 468. B 器件 HNP



1. DPPULLDOWN = 从模块发送到 PHY 的信号，用于使能/禁止 PHY 内部 DP 线的下拉电阻。
   DMPULLDOWN = 从模块发送到 PHY 的信号，用于使能/禁止 PHY 内部 DM 线的下拉电阻。

1. A 器件发出一个 SetFeature b_hnp_enable 描述符以使能 HNP 支持。OTG_FS/OTG_HS 控制器的 ACK 响应表示它支持 HNP。应用程序必须将 OTG 控制和状态寄存器中的设备 HNP 使能位置 1，以指示支持 HNP。

   应用程序将 OTG 控制和状态寄存器中的 HNP 请求位置 1，以向 OTG_FS/OTG_HS 控制器指示启动 HNP。

2. A 器件不再使用总线时，会通过在主机端口控制和状态寄存器中写入端口挂起位来挂起总线。

   OTG_FS/OTG_HS 控制器会在总线空闲 3 ms 后，将模块中断寄存器中的早期挂起位置 1。随后，OTG_FS/OTG_HS 控制器会将模块中断寄存器中的 USB 挂起位置 1。

   OTG_FS/OTG_HS 控制器会断开连接，A 器件在总线上检测到 SE0，指示 HNP 即将开始。OTG_FS/OTG_HS 控制器将使能 PHY 中的 DP 和 DM 下拉电阻，以指示其承担主机角色。

   在检测到 SE0 的 3 ms 内，A 器件会通过激活 OTG_DP 上拉电阻进行响应。OTG_FS/OTG_HS 控制器检测到此信号时认为有设备接入。

   OTG_FS/OTG_HS 控制器将 OTG 中断状态寄存器中的主机协商成功状态更改中断位置 1，指示 HNP 的状态。应用程序必须读取 OTG 控制和状态寄存器中的主机协商成功位，以确定主机协商成功。应用程序必须读取模块中断寄存器 (OTG_GINTSTS) 中的当前模式位，以确定主机工作模式。
3. 应用程序将复位位（OTG_HPRT 中的 PRST 位）置 1，同时 OTG_FS/OTG_HS 控制器发出 USB 复位信号，并枚举 A 器件进行数据通信。
4. OTG_FS/OTG_HS 控制器继续保持发起通信的主机角色，在通信结束后，会通过将主机端口控制和状态寄存器中的端口挂起位置 1 来挂起总线。
5. 在协商模式下，当 A 器件检测到挂起时，会断开连接，然后切换回主机角色。OTG_FS/OTG_HS 控制器将禁止 PHY 中的 DP 和 DM 下拉电阻，以指示其承担设备角色。
6. 应用程序必须读取模块中断寄存器 (OTG_GINTSTS) 中的当前模式位，以确定主机工作模式。
7. OTG_FS/OTG_HS 控制器进行连接，完成 HNP 过程。

38 以太网 (ETH)：通过 DMA 控制器进行介质访问控制 (MAC)

38.1 以太网简介

Portions Copyright (c) 2004, 2005 Synopsys, Inc. 保留所有权利。使用须经许可。

借助以太网外设，STM32F75xxx 和 STM32F74xxx 可以通过以太网按照 IEEE 802.3-2002 标准发送和接收数据。

以太网提供了可配置、灵活的外设，用以满足客户的各种应用需求。它支持与外部物理层 (PHY) 相连的两个工业标准接口：默认情况下使用的介质独立接口 (MII)（在 IEEE 802.3 规范中定义）和简化介质独立接口 (RMII)。它有多种应用领域，例如交换机、网络接口卡等。

以太网遵守以下标准：

- IEEE 802.3-2002，用于以太网 MAC
- IEEE 1588-2008 标准，用于规定联网时钟同步的精度
- AMBA 2.0，用于 AHB 主/从端口
- RMII 联盟的 RMII 规范

38.2 以太网主要特性

以太网 (ETH) 外设包括以下特性（按类列出）：

38.2.1 MAC 内核特性

- 支持外部 PHY 接口实现 10/100 Mb/s 数据传输速率
- 通过符合 IEEE 802.3 的 MII 接口与外部快速以太网 PHY 进行通信
- 支持全双工和半双工操作
  - 支持适用于半双工操作的 CSMA/CD 协议
  - 支持适用于全双工操作的 IEEE 802.3x 流量控制
  - 全双工操作时可以将接收的暂停控制帧转发到用户应用程序
  - 半双工操作时提供背压流量控制
  - 全双工操作中如果流量控制输入信号消失，将自动发送零时间片暂停帧
- 报头和帧起始数据 (SFD) 在发送路径中插入、在接收路径中删除
- 可逐帧控制 CRC 和 pad 自动生成
- 接收帧时可自动去除 pad/CRC
- 可编程帧长度，支持高达 16 KB 的巨型帧
- 可编程帧间隔（40-96 位时间，以 8 为步长）
- 支持多种灵活的地址过滤模式：
  - 高达四个 48 位完美 (DA) 地址过滤器，对每个字节进行掩码操作
  - 高达三次 48 位 SA 地址比较检查，对每个字节进行掩码操作
  - 64 位 Hash 滤波器（可选），适用于多播和单播 (DA) 地址
  - 可传送所有多播地址帧
  - 支持混合模式，因此可传送所有帧，无需为网络监视进行过滤
  - 传送所有传入数据包时（每次过滤时）均附有一份状态报告

- 为发送和接收数据包分别返回 32 位状态
- 支持对接收帧进行 IEEE 802.1Q VLAN 变量检测
- 为应用程序提供单独的发送、接收和控制接口
- 支持通过 RMON/MIB 计数器 (RFC2819/RFC2665) 进行强制网络统计
- 使用 MDIO 接口配置和管理 PHY 设备
- 检测 LAN 唤醒帧和 AMD Magic Packet™ 帧
- 在接收功能中支持对接收到的由以太网帧封装的 IPv4 和 TCP 数据包进行校验和卸载
- 在增强型接收功能中支持检查 IPv4 头校验和以及在 IPv4 或 IPv6 数据包中封装的 TCP、UDP 或 ICMP 校验和
- 支持以太网帧时间戳（请参见 IEEE 1588-2008）。每个帧的发送或接收状态下给出 64 位时间戳
- 两组 FIFO：一个具有可编程阈值功能的 2 KB 发送 FIFO 和一个具有可配置阈值（默认为 64 个字节）功能的 2 KB 接收 FIFO
- 接收 FIFO 进行多帧存储时，通过在 EOF 传输后向接收 FIFO 插入接收状态矢量，从而使得接收 FIFO 无需存储这些帧的接收状态
- 在存储转发模式下，可以在接收时过滤所有的错误帧，此时不会将这些错误帧转发给应用程序
- 可以转发过小的好帧
- 为接收 FIFO 中丢失或损坏的帧（由于溢出）生成脉冲，借此支持数据统计
- 向 MAC 内核发送数据时支持存储转发机制
- 根据接收 FIFO 填充（阈值可配置）级别自动生成要发送至 MAC 内核的暂停帧控制或背压信号
- 发送时处理冲突帧的自动重新发送
- 丢弃延迟冲突、过度冲突、过度延迟和下溢条件下的帧
- 通过软件控制刷新 Tx FIFO
- 计算 IPv4 头校验和与 TCP、UDP 或 ICMP 校验和并将其插入在存储转发模式下发送的帧中
- 支持调试时通过 MII 进行内部回送

38.2.2 DMA 特性

- AHB 从接口中支持所有 AHB 突发类型
- 软件可以在 AHB 主接口中选择 AHB 突发类型（固定或不确定突发）
- 可以从 AHB 主端口选择地址对齐突发
- 通过帧定界符优化以数据包为导向的 DMA 传输
- 支持对数据缓冲区进行字节对齐寻址
- 双缓冲区（环）或链表（链接）描述符链接
- 采用描述符架构可以在 CPU 几乎不干预的情况传输大型数据块
- 每个描述符可传输高达 8 KB 的数据
- 报告正常工作和传输错误时的综合状态
- 可为发送和接收 DMA 引擎单独编程突发大小，以充分利用主总线
- 可编程中断选项，适用于不同的工作条件
- 按帧控制发送/接收完成中断
- 接收引擎和发送引擎间采用循环调度仲裁或固定优先级仲裁

- 启动/停止模式
- 当前 Tx/Rx 缓冲区指针作为状态寄存器
- 当前 Tx/Rx 描述符指针作为状态寄存器

38.2.3 PTP 特性

- 接收帧和发送帧时间戳
- 粗略校准法和精细校准法
- 系统时间大于目标时间时触发中断
- 输出秒脉冲（产品复用功能输出）

38.3 以太网引脚

表 235 显示了 MAC 信号和相应的 MII/RMII 信号映射。所有 MAC 信号均映射到 AF11，一些信号映射到不同的 I/O 引脚，这些应在复用功能模式下进行配置（有关详细信息，请参见 *第 6.3.2 节：I/O 引脚复用功能复用器和映射*）。

表 235. 复用功能映射

端口	AF11
	ETH
PA0-WKUP	ETH_MII_CRS
PA1	ETH_MII _RX_CLK/ETH_RMII _REF_CLK
PA2	ETH _MDIO
PA3	ETH _MII_COL
PA7	ETH_MII _RX_DV/ETH_RMII _CRS_DV
PB0	ETH _MII_RXD2
PB1	ETH _MII_RXD3
PB5	ETH _PPS_OUT
PB8	ETH _MII_TXD3
PB10	ETH_ MII_RX_ER
PB11	ETH _MII_TX_EN/ETH _RMII_TX_EN
PB12	ETH _MII_TXD0/ETH _RMII_TXD0
PB13	ETH _MII_TXD1/ETH _RMII_TXD1
PC1	ETH _MDC
PC2	ETH _MII_TXD2
PC3	ETH _MII_TX_CLK
PC4	ETH_MII_RXD0/ETH_RMII_RXD0
PC5	ETH _MII_RXD1/ETH _RMII_RXD1
PE2	ETH_MII_TXD3
PG8	ETH_PPS_OUT

表 235. 复用功能映射（续）

端口	AF11
	ETH
PG11	ETH _MII_TX_EN/ETH _RMII_TX_EN
PG13	ETH _MII_TXD0/ETH _RMII_TXD0
PG14	ETH _MII_TXD1/ETH _RMII_TXD1
PH2	ETH _MII_CRS
PH3	ETH _MII_COL
PH6	ETH _MII_RXD2
PH7	ETH _MII_RXD3
PI10	ETH _MII_RX_ER

38.4 以太网功能说明：SMI、MII 和 RMII

以太网外设包括带专用 DMA 控制器的 MAC 802.3（介质访问控制）。它支持介质独立接口 (MII) 和简化介质独立接口 (RMII)，并通过一个选择位在两个接口间进行切换（请参见 SYSCFG_PMC 寄存器）。

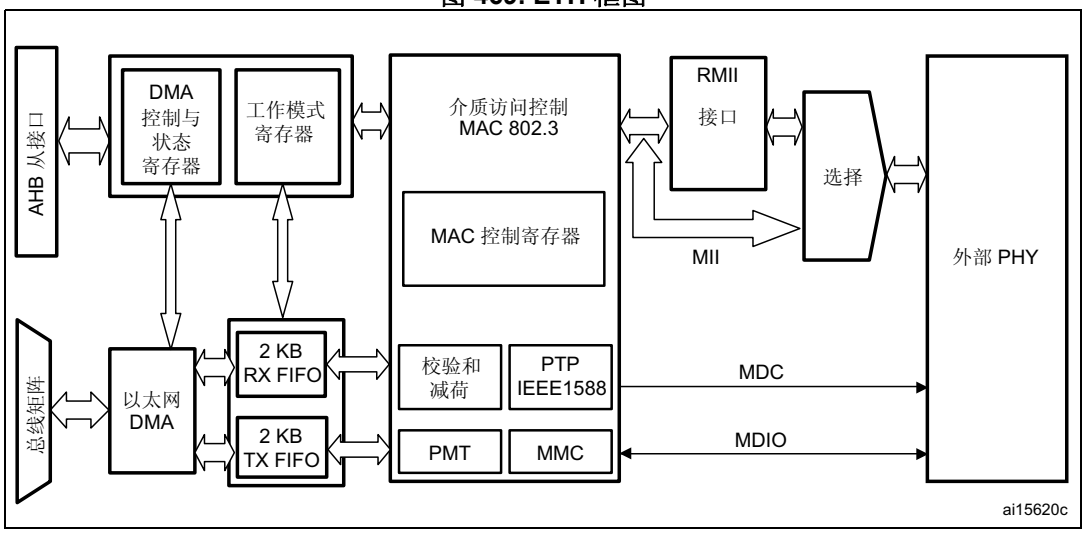
DMA 控制器通过 AHB 主从接口与内核和存储器相连。AHB 主接口用于控制数据传输，而 AHB 从接口则用于访问“控制和状态寄存器”(CSR) 的空间。

在进行数据发送时，首先将数据由系统存储器以 DMA 的方式送至发送 FIFO (Tx FIFO) 进行缓冲，再通过 MAC 内核发送。同样，接收 FIFO (Rx FIFO) 则存储通过线路接收的以太网帧，直到这些帧通过 DMA 传送到系统存储器。

以太网外设还包括用于与外部 PHY 通信的 SMI。通过一组配置寄存器，用户可以为 MAC 控制器和 DMA 控制器选择所需模式和功能。

注： *当使用以太网时，AHB 时钟频率必须至少为 25 MHz。*

图 469. ETH 框图



1. 有关 AHB 连接的信息，请参见*图 1：STM32F75xxx 和 STM32F74xxx 器件的系统架构*。

38.4.1 站管理接口：SMI

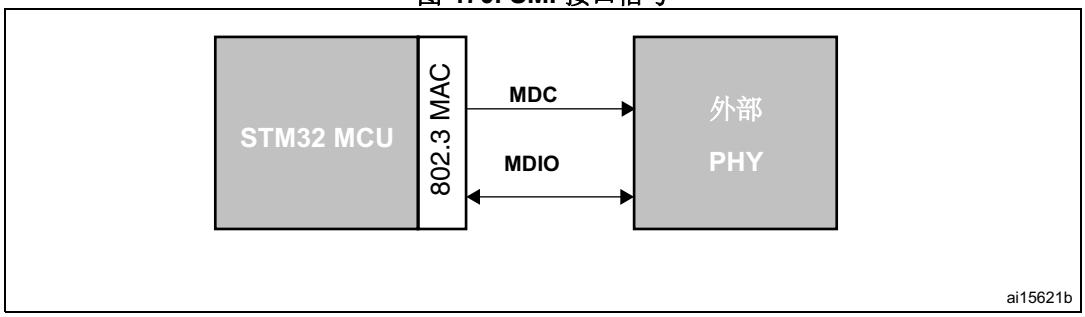
站管理接口 (SMI) 允许应用程序通过 2 线时钟和数据线访问任意 PHY 寄存器。该接口支持访问多达 32 个 PHY。

应用程序可以从 32 个 PHY 中选择一个 PHY，然后从任意 PHY 包含的 32 个寄存器中选择一个寄存器，发送控制数据或接收状态信息。任意给定时间内只能对一个 PHY 中的一个寄存器进行寻址。

MDC 时钟线和 MDIO 数据线在微控制器中均用作复用功能 I/O：

- MDC：周期性时钟，提供以最大频率 2.5 MHz 传输数据时的参考时序。MDC 的最短高电平时间和最短低电平时间必须均为 160 ns。MDC 的最小周期必须为 400 ns。在空闲状态下，SMI 管理接口将 MDC 时钟信号驱动为低电平。
- MDIO：数据输入/输出比特流，用于通过 MDC 时钟信号向/从 PHY 设备同步传输状态信息

图 470. SMI 接口信号



SMI 帧格式

表 236 中给出了与读操作或写操作有关的帧结构，位传输顺序必须从左到右。

表 236. 管理帧格式

	管理帧字段							
	报头（32 位）	起始	操作	PADDR	RADDR	TA	数据（16 位）	空闲
读取	1... 1	01	10	ppppp	rrrrr	Z0	dddddddddddddddd	Z
写入	1... 1	01	01	ppppp	rrrrr	10	dddddddddddddddd	Z

管理帧包括八个字段：

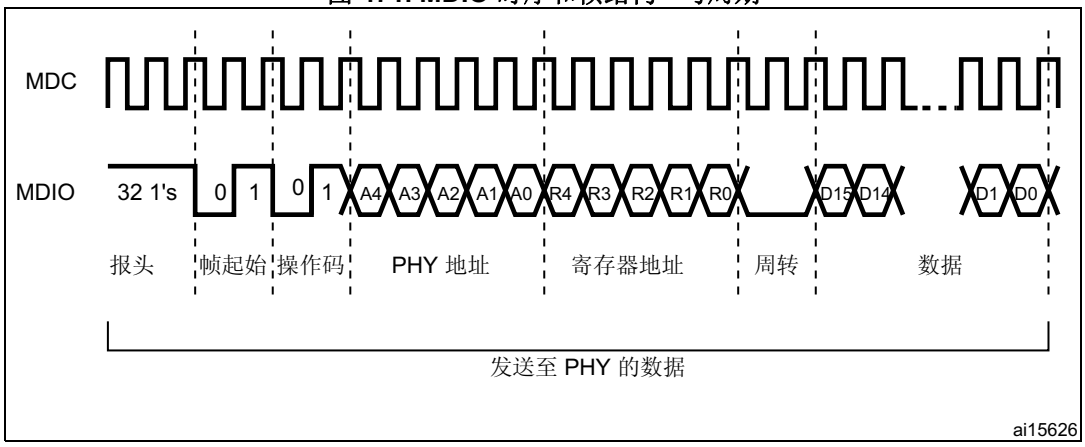
- **报头**：每个事务（读取或写入）均可通过报头字段启动，报头字段对应于 MDIO 线上 32 个连续的逻辑“1”位以及 MDC 上的 32 个周期。该字段用于与 PHY 设备建立同步。
- **起始**：帧起始由 <01> 模式定义，用于验证线路从默认逻辑“1”状态变为逻辑“0”状态，然后再从逻辑“0”状态变为逻辑“1”状态。
- **操作**：定义正在发生的事务（读取或写入）的类型。
- **PADDR**：PHY 地址有 5 位，可构成 32 个唯一 PHY 地址。最先发送和接收地址的 MSB 位。
- **PADDR**：寄存器地址有 5 位，从而可在所选 PHY 设备中对 32 个不同的寄存器进行寻址。最先发送和接收地址的 MSB 位。
- **TA**：周转字段在 RADDR 和 DATA 字段间定义了一个 2 位模式，以避免在读取事务期间出现竞争现象。读取事务时，MAC 控制器将 TA 的 2 个位驱动为 MDIO 线上的高阻态。PHY 设备必须将 TA 的第一位驱动为高阻态，将 TA 的第二位驱动为“0”。
  写入事务时，MAC 控制器针对 TA 字段驱动 <10> 模式。PHY 设备必须将 TA 的 2 个位驱动为高阻态。
- **数据**：数据字段为 16 位。最先发送和接收的位必须为 ETH_MIID 寄存器的位 15。
- **空闲**：MDIO 线驱动为高阻态。三态驱动器必须禁止，PHY 的上拉电阻使线路保持逻辑“1”状态。

SMI 写操作

当应用程序将 MII 写入位和繁忙位（在*以太网 MAC MII 地址寄存器 (ETH_MACMIIAR)* 中）置 1 时，SMI 将通过传输 PHY 地址、PHY 中的寄存器地址以及写入数据（在*以太网 MAC MII 数据寄存器 (ETH_MACMIIDR)* 中）来触发对 PHY 寄存器进行写操作。事务进行期间，应用程序不应更改 MII 地址寄存器的内容或 MII 数据寄存器。在此期间对 MII 地址寄存器或 MII 数据寄存器执行的写操作将会忽略（繁忙位处于高电平状态），事务将无错完成。写操作完成后，SMI 将通过复位繁忙位进行指示。

图 471 显示了写操作的帧格式。

图 471. MDIO 时序和帧结构 - 写周期

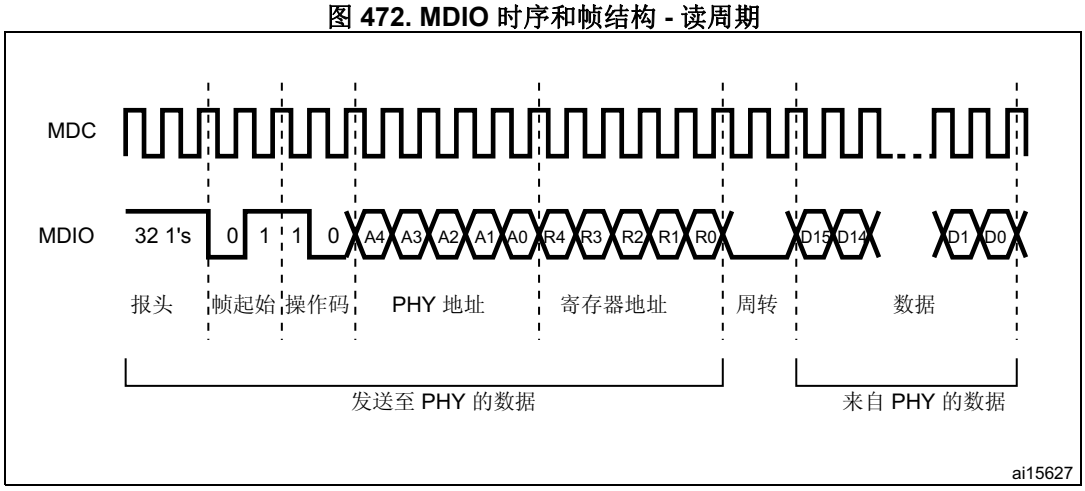


SMI 读操作

当用户将以太网 MAC MII 地址寄存器 (ETH_MACMIIAR) 中的 MII 繁忙位置 1、MII 写入位清零时，SMI 将通过传输 PHY 地址和 PHY 中的寄存器地址在 PHY 寄存器中触发读操作。事务进行期间，应用程序不应更改 MII 地址寄存器的内容或 MII 数据寄存器。在此期间对 MII 地址寄存器或 MII 数据寄存器执行的写操作将会忽略（繁忙位处于高电平状态），事务将无错完成。读操作完成后，SMI 将复位繁忙位，然后用从 PHY 中读取的数据更新 MII 数据寄存器。

图 472 显示了读操作的帧格式。

图 472. MDIO 时序和帧结构 - 读周期



SMI 时钟选择

MAC 启动管理写/读操作。SMI 时钟是一个分频时钟，其时钟源为应用时钟（AHB 时钟）。分频系数取决于 MII 地址寄存器中设置的时钟范围。

表 237 显示了如何设置时钟范围。

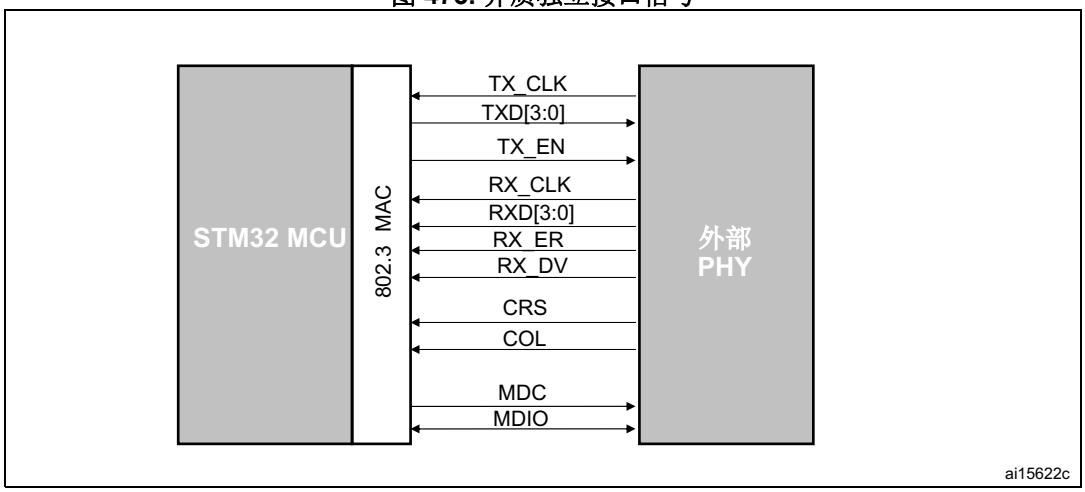
表 237. 时钟范围

选择	HCLK 时钟	MDC 时钟
000	60-100 MHz	AHB 时钟/42
001	100-150 MHz	AHB 时钟/62
010	20-35 MHz	AHB 时钟/16
011	35-60 MHz	AHB 时钟/26
100	150-216 MHz	AHB 时钟/102
101, 110, 111	保留	-

38.4.2 介质独立接口：MII

介质独立接口 (MII) 定义了 10 Mb/s 和 100 Mb/s 的数据传输速率下 MAC 子层与 PHY 之间的互连。

图 473. 介质独立接口信号



- MII_TX_CLK：连续时钟信号。该信号提供进行 TX 数据传输时的参考时序。标称频率为：速率为 10 Mb/s 时为 2.5 MHz；速率为 100 Mb/s 时为 25 MHz。
- MII_RX_CLK：连续时钟信号。该信号提供进行 RX 数据传输时的参考时序。标称频率为：速率为 10 Mb/s 时为 2.5 MHz；速率为 100 Mb/s 时为 25 MHz。
- MII_TX_EN：发送使能信号。该信号表示 MAC 当前正针对 MII 发送半字节。该信号必须与报头的前半字节进行同步 (MII_TX_CLK)，并在所有待发送的半字节均发送到 MII 时必须保持同步。
- MII_TXD[3:0]：数据发送信号。该信号是 4 个一组的数据信号，由 MAC 子层同步驱动，在 MII_TX_EN 信号有效时才为有效信号（有效数据）。MII_TXD[0] 为最低有效位，MII_TXD[3] 为最高有效位。禁止 MII_TX_EN 时，发送数据不会对 PHY 产生任何影响。
- MII_CRS：载波侦听信号。当发送或接收介质处于非空闲状态时，由 PHY 使能该信号。发送和接收介质均处于空闲状态时，由 PHY 禁止该信号。PHY 必须确保 MII_CS 信号在冲突条件下保持有效状态。该信号无需与 TX 和 RX 时钟保持同步。在全双工模式下，该信号没意义。
- MII_COL：冲突检测信号。检测到介质上存在冲突后，PHY 必须立即使能冲突检测信号，并且只要存在冲突条件，冲突检测信号必须保持有效状态。该信号无需与 TX 和 RX 时钟保持同步。在全双工模式下，该信号没意义。
- MII_RXD[3:0]：数据接收信号。该信号是 4 个一组的数据信号，由 PHY 同步驱动，在 MII_RX_DV 信号有效时才为有效信号（有效数据）。MII_RXD[0] 为最低有效位，MII_RXD[3] 为最高有效位。当 MII_RX_EN 禁止、MII_RX_ER 使能时，特定的 MII_RXD[3:0] 值用于传输来自 PHY 的特定信息（请参见*表 239*）。
- MII_RX_DV：接收数据有效信号。该信号表示 PHY 当前正针对 MII 接收已恢复并解码的半字节。该信号必须与恢复帧的头半字节进行同步 (MII_RX_CLK)，并且一直保持同步到恢复帧的最后半字节。该信号必须在最后半字节随后的第一个时钟周期之前禁止。为了正确地接收帧，MII_RX_DV 信号必须在时间范围上涵盖要接收的帧，其开始时间不得迟于 SFD 字段出现的时间。
- MII_RX_ER：接收错误信号。该信号必须保持一个或多个周期 (MII_RX_CLK)，从而向 MAC 子层指示在帧的某处检测到错误。该错误条件必须通过 MII_RX_DV 验证，如*表 239* 所示。

表 238. TX 接口信号编码

MII_TX_EN	MII_TXD[3:0]	说明
0	0000 到 1111	正常帧间
1	0000 到 1111	正常数据发送

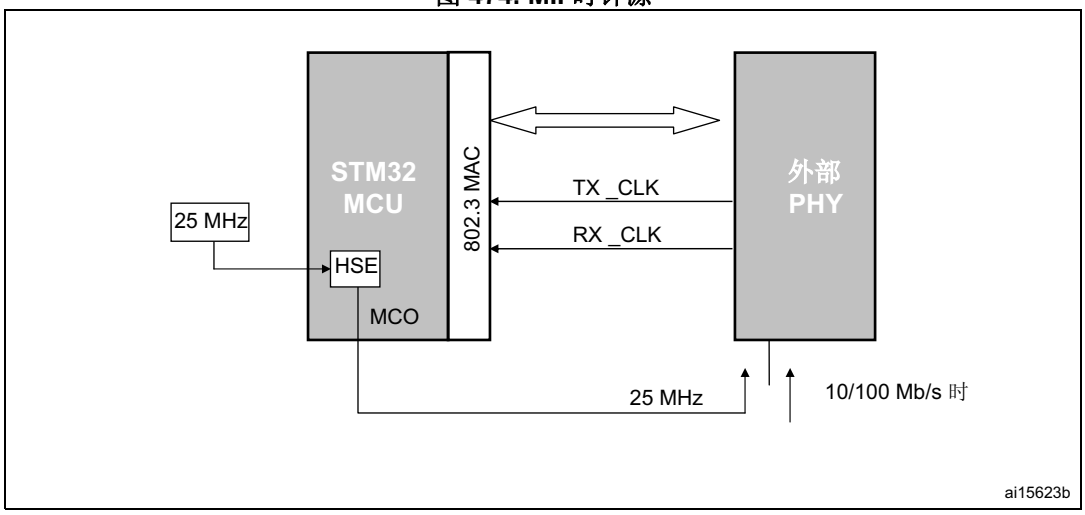
表 239. RX 接口信号编码

MII_RX_DV	MII_RX_ERR	MII_RXD[3:0]	说明
0	0	0000 到 1111	正常帧间
0	1	0000	正常帧间
0	1	0001 到 1101	保留
0	1	1110	错误载波检测
0	1	1111	保留
1	0	0000 到 1111	正常数据接收
1	1	0000 到 1111	数据接收出现错误

MII 时钟源

要生成 TX_CLK 和 RX_CLK 时钟信号，必须向外部 PHY 提供 25MHz 时钟，如*图 474* 所示。除了使用外部 25 MHz 石英晶体提供该时钟，还可以通过 STM32F75xxx 和 STM32F74xxx 微控制器的 MCO 引脚输出该信号。这种情况下，必须对 PLL 倍频进行配置，以通过 25 MHz 外部石英晶体在 MCO 引脚上获得所需频率。

图 474. MII 时钟源



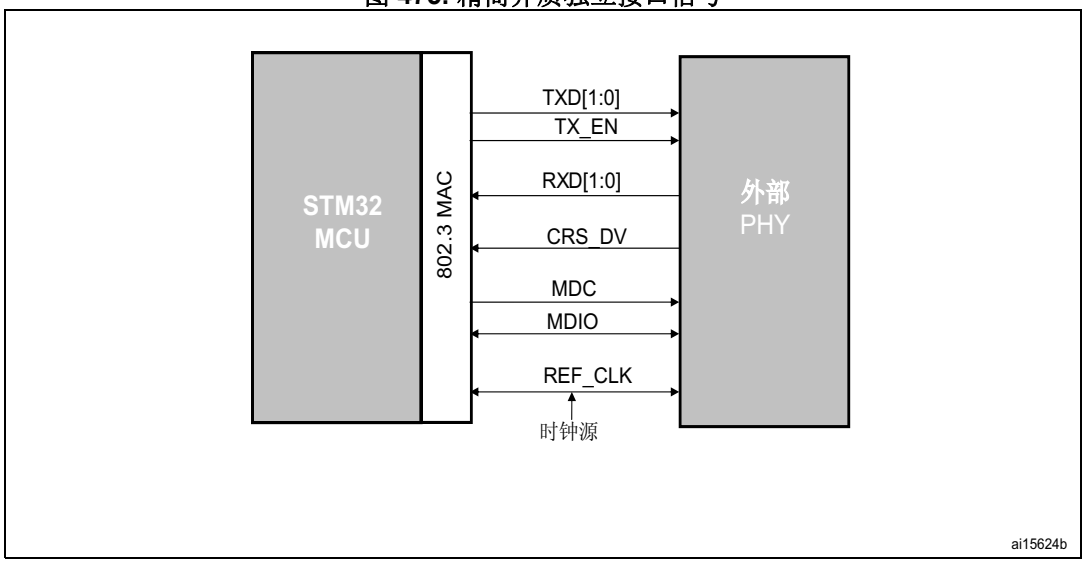
38.4.3 精简介质独立接口：RMII

精简介质独立接口 (RMII) 规范降低了 10/100 Mb/s 下微控制器以太网外设与外部 PHY 间的引脚数。根据 IEEE 802.3u 标准，MII 包括 16 个数据和控制信号的引脚。RMII 规范将引脚数减少为 7 个（引脚数减少 62.5%）。

RMII 是 MAC 和 PHY 之间的实例化对象。这有助于将 MAC 的 MII 转换为 RMII。RMII 具有以下特性：

- 支持 10 Mb/s 和 100 Mb/s 的运行速率
- 参考时钟必须是 50 MHz
- 相同的参考时钟必须从外部提供给 MAC 和外部以太网 PHY
- 它提供了独立的 2 位宽（双位）的发送和接收数据路径

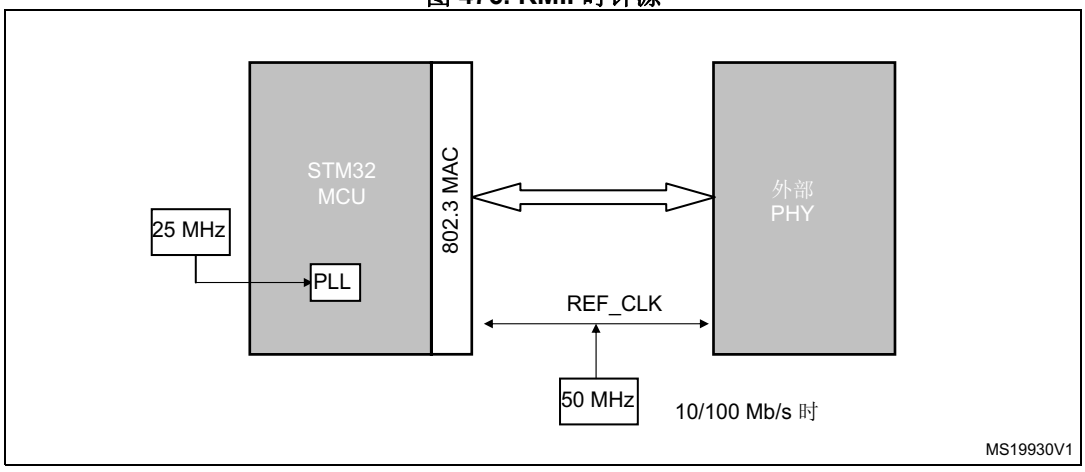
图 475. 精简介质独立接口信号



RMII 时钟源

使用外部 50 MHz 时钟驱动 PHY 或使用嵌入式 PLL 生成 50 MHz 频率信号来驱动 PHY。

图 476. RMII 时钟源



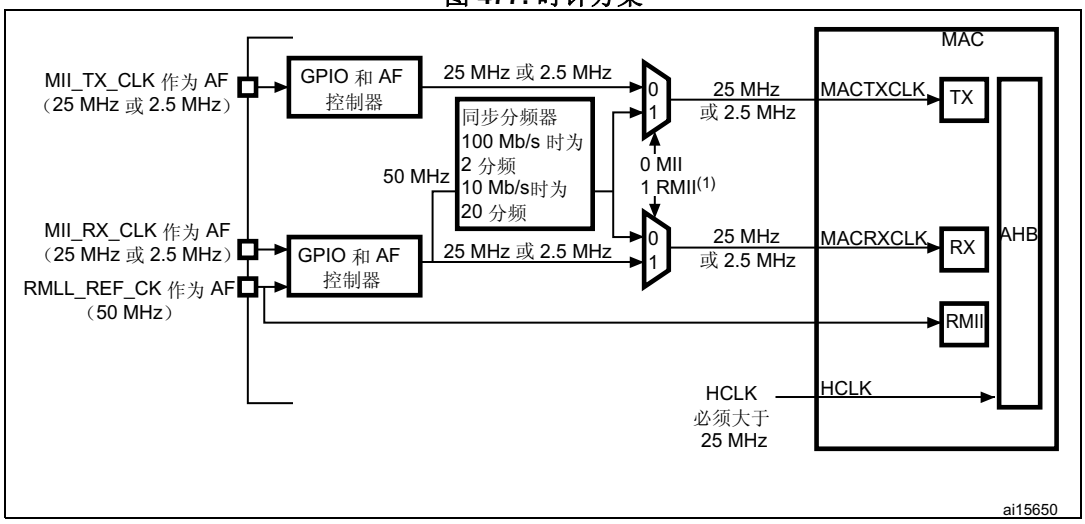
38.4.4 MII/RMII 选择

使用 SYSCFG_PMC 寄存器中的配置位 23 MII_RMII_SEL 选择 MII 或 RMII 模式。以太网控制器处于复位模式或使能时钟前，应用程序必须设置 MII/RMII 模式。

MII/RMII 内部时钟方案

支持 MII 和 RMII 以及 10 和 100 Mb/s 运行所需的时钟方案如*图 477* 所示。

图 477. 时钟方案



1. 通过 SYSCFG_PMC 寄存器中的位 23 MII_RMII_SEL 控制 MII/RMII 选择。

要节省引脚，需在同一个 GPIO 引脚上复用 RMII_REF_CK 和 MII_RX_CLK 这两个输入时钟信号。

38.5 以太网功能说明：MAC 802.3

适用于局域网 (LAN) 的 IEEE 802.3 国际标准将 CSMA/CD（带有冲突检测的载波侦听多路访问）用作访问方法。

以太网外设包括一个带介质独立接口 (MII) 的 MAC 802.3（介质访问控制）控制器和一个专用 DMA 控制器。

MAC 模块对以下系列的系统使用 LAN CSMA/CD 子层：数据速率为 10 Mb/s 和 100 Mb/s 的基带系统和宽带系统。支持半双工和全双工工作模式。冲突检测访问方法仅适用于半双工工作模式。支持 MAC 控制帧子层。

MAC 子层执行以下与数据链路控制步骤相关的功能：

- 数据封装（发送和接收）
  - 组帧（帧边界定界、帧同步）
  - 寻址（处理源地址和目标地址）
  - 错误检测
- 介质访问管理
  - 介质分配（冲突避免）
  - 竞争解决（冲突处理）

MAC 子层主要有两个工作模式：

- 半双工模式：站点使用 CSMA/CD 算法争用物理介质。
- 全双工模式：满足以下条件时，无需解决竞争问题（CSMA/CD 算法不是必需的）便可同时发送和接收数据：
  - 物理介质能够支持同步发送和接收
  - 正好有 2 个站点与 LAN 相连
  - 两个站均配置为全双工工作模式

38.5.1 MAC 802.3 帧格式

正如 IEEE 802.3-2002 标准规定，MAC 块使用 MAC 子层和可选 MAC 控制子层 (10/100 Mb/s)。

为使用 CSMA/CD MAC 的数据通信系统指定了两个帧格式：

- 基本 MAC 帧格式
- 标记 MAC 帧格式（扩展了基本 MAC 帧格式）

图 479 和*图 480* 介绍了包含以下字段的帧结构（未标记和标记）：

- 报头：7 字节字段，用于进行同步（PLS 电路）

  十六进制值：55-55-55-55-55-55-55

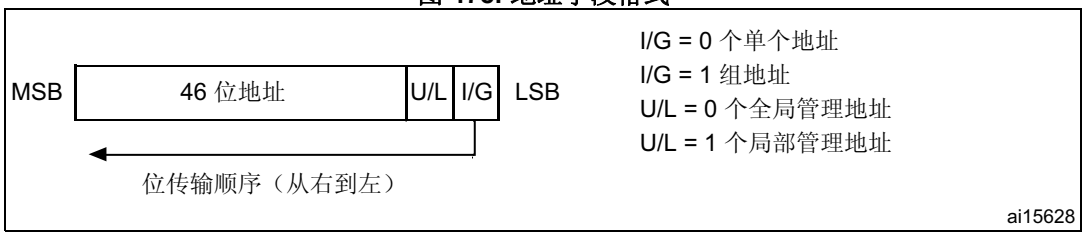
  位模式：01010101 01010101 01010101 01010101 01010101 01010101 01010101（从右到左进行位传输）
- 起始帧定界符 (SFD)：1 字节字段，用于指示帧的开始。
  十六进制值：D5

  位模式：11010101（从右到左进行位传输）
- 目标地址字段和源地址字段：6 字节字段，指示目标站和源站地址，具体如下（请参见*图 478*）：
  - 每个地址的长度为 48 位
  - 目标地址字段中的第一个 LSB 位 (I/G) 用于指示单个地址 (I/G = 0) 或组地址 (I/G = 1)。一个组地址可以标识“无”、一个或多个，或所有连接至 LAN 的站。在源地址中，第一位保留并复位为 0。
  - 第二位 (U/L) 用于区分局部 (U/L = 1) 或全局 (U/L = 0) 管理地址。对于广播地址，该位同样为 1。
  - 每个地址字段的各个字节必须最先发送最低有效位。

地址指定基于以下类型：

- 单个地址：与网络中的特殊站关联的物理地址。
- 组地址。与给定网络中一个或多个站关联的多目标地址。有两种多播地址：
  - 多播组地址：与一组逻辑相关站关联的地址。
  - 广播地址：一个特殊的预定义多播地址（目标地址字段中全为“1”），该地址总是表示给定 LAN 上的所有站。

图 478. 地址字段格式



- QTag 前缀：在源地址字段和 MAC 客户端长度/类型字段中插入的 4 字节字段。该字段是对基本帧（未标记）的扩展，用于获得标记的 MAC 帧。未标记的 MAC 帧不包括该字段。扩展的标记如下：
  - 2 字节常量长度/类型字段值（符合“类型”解析，大于 0x0600），等于 802.1Q 变量协议类型的值（十六进制 0x8100）。该常量字段用于区分标记和未标记的 MAC 帧。
  - 包含变量控制信息的 2 字节字段可以再分为：一个 3 位用户优先级、一个 1 位标准格式指示符 (CFI) 和一个 12 位 VLAN 标识符。标记 MAC 帧的长度通过 QTag 前缀扩展 4 个字节。
- MAC 客户端长度/类型：2 字节字段，具有不同含义（互斥），具体取决于其值：
  - 如果该值小于或等于 maxValidFrame (0d1500)，则该字段表示 802.3 帧的后续数据字段中所包含的 MAC 客户端数据字节的数量（长度解析）。
  - 如果该值大于或等于 MinTypeValue（十进制 0d1536，0x0600），则该字段表示与以太网帧相关的 MAC 客户端协议的种类（类型解析）。

  无论长度/类型字段的解析结果为何，如果数据字段的长度小于协议正确运行所需的最小长度，则将在数据字段之后、FCS（帧检查序列）字段之前添加一个 PAD 字段。发送和接收长度/类型字段时，高位字节在前。

  对于在 maxValidLength 到 minTypeValue 范围内（不包括边界）的长度/类型字段值，未指定 MAC 子层的行为：MAC 子层可能传递、也可能不传递这些值。
- 数据和 PAD 字段：n 字节数据字段。其数据完全透明，这意味着数据字段中可能出现任意顺序的字节值。PAD 的大小（如果存在）由数据字段的大小决定。数据和 PAD 字段的最大和最小长度为：
  - 最大长度 = 1500 字节
  - 无标记的 MAC 帧的最小长度 = 46 字节
  - 带标记的 MAC 帧的最小长度 = 42 字节

  当数据字段的长度小于要求的最小长度时，将添加 PAD 字段以匹配最小长度（带标记的帧为 42 字节，无标记的帧为 46 字节）。
- 帧检查序列：包含循环冗余校验 (CRC) 值的 4 字节字段。CRC 计算基于下列字段：源地址、目标地址、QTag 前缀、长度/类型、LLC 数据和 PAD（即，除报头、SFD 字段以外的所有字段）。生成的多项式如下：

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

帧的 CRC 值按如下方式计算：

- 帧的前 2 位是互补位
- 帧的 n 个位是 (n – 1) 次多项式 M(x) 的系数。目标地址的第一位对应于 x^{n-1} 项，数据字段的最后一位对应于 x^0 项
- M(x) 乘以 x^{32} 再除以 G(x)，得到 ≤ 31 次的余数 R(x)

- 将 R(x) 的系数视为一个 32 位序列
- 对位序列进行互补，结果为 CRC
- 32 位的 CRC 值放置在帧检查序列中。首先发送 x^{32} 项，最后发送 x^0 项

图 479. MAC 帧格式

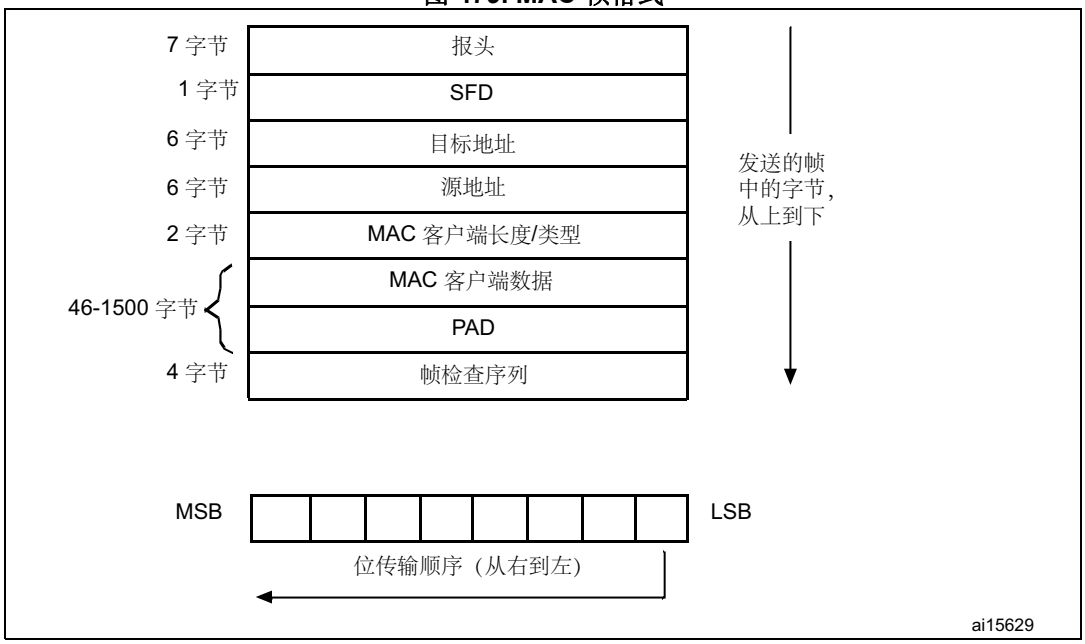
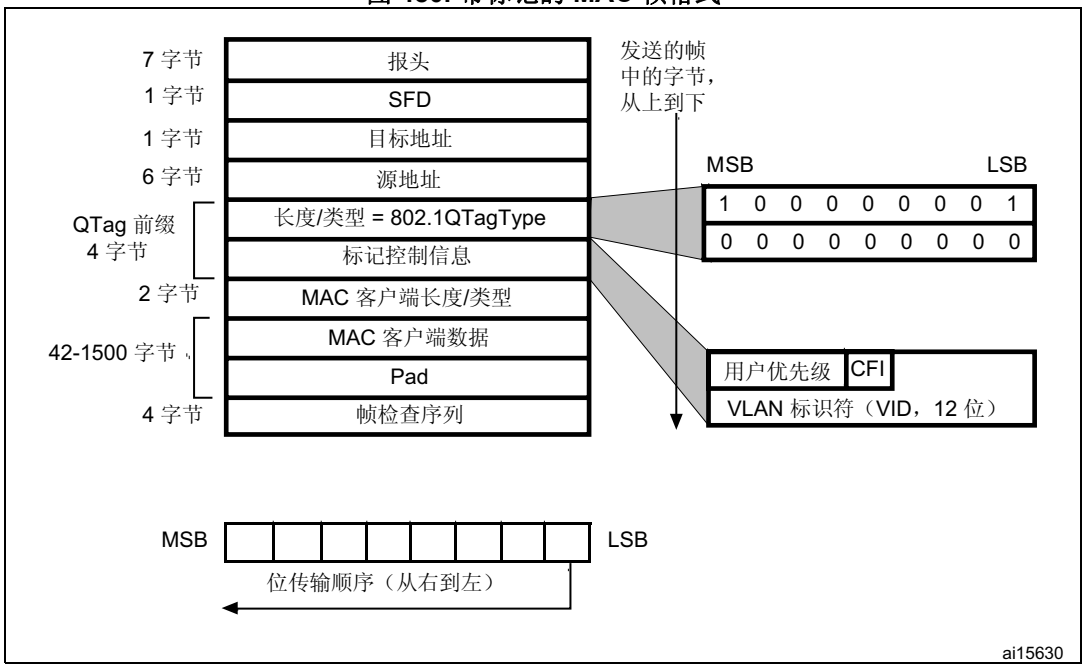


图 480. 带标记的 MAC 帧格式



发送 MAC 帧的每个字节（除 FSC 字段外）时，低位在前。

通过以下条件之一定义无效 MAC 帧：

- 帧长度与长度 / 类型字段指定的预期值不一致。如果长度 / 类型字段包含类型值，则认为帧长度与此字段一致（没有无效帧）
- 帧长度不是字节的整数倍（额外位）
- 根据传入帧计算出的 CRC 值与包含的 FCS 不匹配。

38.5.2 MAC 帧发送

DMA 控制发送路径的所有事务。从系统存储器读取的以太网帧由 DMA 推入 FIFO。然后将帧弹出并传输到 MAC 内核。帧传输结束时，从 MAC 内核获取发送状态并传回 DMA。发送 FIFO 的深度为 2 KB。FIFO 填充级别将指示给 DMA，以便 DMA 可通过 AHB 接口在所需的系统存储器突发中启动数据获取。来自 AHB 主接口的数据将推入 FIFO。

检测到 SOF 时，MAC 接受数据并开始向 MII 发送。在应用程序启动发送后，向 MII 发送帧数据所需的时间是可变的，具体取决于 IFG 延迟、发送报头/SFD 的时间以及半双工模式的任意回退延迟等延迟因素。EOF 传输到 MAC 内核后，内核将完成正常的发送，然后将发送的状态返回给 DMA。如果在发送过程中发生常规冲突（在半双工模式下），MAC 内核将使发送状态有效，然后接受并丢弃所有后续数据，直至收到下一 SOF。检测到来自 MAC 的重试请求（在状态中）时，应从 SOF 重新发送同一帧。如果发送期间未连续提供数据，MAC 将发出下溢状态。在帧的正常传输期间，如果 MAC 在未获得前一帧的 EOF 的情况下接收到 SOF，则将忽略该 SOF 并将新的帧视为前一帧的延续。

向 MAC 内核弹出数据有两种操作模式：

- 在阈值模式下，只要 FIFO 中的字节数超过配置的阈值（或在超过阈值前写入帧结束），数据就准备好弹出并转发到 MAC 内核。使用 ETH_DMABMR 的 TTC 位配置阈值。
- 在存储转发模式下，仅当在 FIFO 中存储完整的帧后，才会向 MAC 内核弹出帧。如果 Tx FIFO 的大小小于要发送的以太网帧，则在 Tx FIFO 接近填满时向 MAC 内核弹出帧。

应用可通过将 FTF 位（ETH_DMAOMR 寄存器 [20]）置 1 来清空发送 FIFO 的所有内容。此位自行清零，并将 FIFO 指针初始化为默认状态。如果向 MAC 内核传输帧时将 FTF 位置 1，则传输将停止，因为此时 FIFO 被视为空。因此，MAC 发送器将出现下溢事件并且相应的状态字将转发给 DMA。

自动 CRC 和 pad 生成

当从应用程序接收的字节数少于 60（DA+SA+LT+ 数据）时，会向发送帧附加零，使数据长度正好为 46 字节，以满足 IEEE 802.3 的最小数据字段要求。可将 MAC 编程为不附加任何填充值。计算帧检查序列 (FCS) 字段的循环冗余校验 (CRC) 并将其附加到正在发送的数据。如果将 MAC 编程为不将 CRC 值附加到以太网帧的末尾，则不发送计算出的 CRC。此规则有一种例外情况，即，将 MAC 编程为向小于 60 字节（DA+SA+LT+ 数据）的帧附加填充时，CRC 将附加在填充帧的末尾。

CRC 发生器计算以太网帧的 FCS 字段的 32 位 CRC。编码由下面的多项式定义。

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

发送协议

MAC 控制以太网帧的发送操作。它执行下列功能以满足 IEEE 802.3/802.3z 规范。包括：

- 生成报头和 SFD
- 在半双工模式下生成阻塞信号
- 控制 Jabber 超时
- 控制半双工模式下的流量（背压）
- 生成发送帧状态
- 包含符合 IEEE 1588 的时间戳快照逻辑

当请求发送新的帧时，MAC 将发送报头和 SFD，紧接着发送数据。报头定义为 0b10101010 模式的 7 个字节，SFD 定义为 0b10101011 模式的 1 个字节。冲突窗口定义为 1 个时隙（对于 10/100 Mb/s 以太网，为 512 个位时间）。阻塞信号生成仅适用于半双工模式，不适用于全双工模式。

在 MII 模式下，如果在开始传输帧到 CRC 字段结束之间的任何时间发生冲突，MAC 将在 MII 上发送 0x5555 5555 的 32 位阻塞信号，通知所有其它站已发生冲突。如果在报头发送阶段发生冲突，MAC 将完成报头和 SFD 的发送，然后发送阻塞信号。

系统使用一个 jabber 定时器，用于在传输的字节超过 2048 字节（默认值）时切断以太网帧的发送。在半双工模式下，MAC 使用延迟机制进行流量控制（背压）。当应用程序请求停止接收帧时，如果已使能发送流量控制，则只要 MAC 检测到接收帧，就会发送一个 32 字节的 JAM 信号。这会导致冲突并使远程站回退。应用程序通过将 ETH_MACFCR 寄存器中的 BPA 位（位 0）置 1 来请求流量控制。如果应用程序请求发送帧，则即使激活背压功能，也将按调度计划发送。请注意，如果背压功能长时间保持激活（发生的连续冲突事件超过 16 个），则远程站将由于冲突过多而中止发送。如果针对发送帧使能 IEEE 1588 时间戳功能，则将 SFD 置于发送 MII 总线时，此模块会获取系统时间的快照。

发送调度程序

MAC 负责调度 MII 上的帧发送。它可保持两个发送帧之间的帧间隔，并在半双工模式下遵循截断二进制指数回退算法。在满足 IFG 和回退延迟条件后，MAC 使能发送。它可确保两个发送帧之间的空闲期段，即配置的帧间隔（ETH_MACCR 寄存器中的 IFG 位）。如果要发送的帧在配置的 IFG 时间之前到达，则 MII 会等待来自 MAC 的使能信号，然后再开始发送。只要 MII 的载波信号进入无效状态，MAC 就会启动其 IFG 计数器。在编程的 IFG 值的末尾，MAC 将以全双工模式使能发送。在半双工模式下，当 IFG 配置为 96 个位时间时，MAC 将遵循 IEEE 802.3 规范第 4.2.3.2.1 节中指定的顺从规则。如果在 IFG 间隔的前三分之二时间内（对于所有 IFG 值都为 64 位时间）检测到载波，MAC 将复位其 IFG 计数器。如果在 IFG 间隔的后三分之一时间内检测到载波，MAC 将继续执行 IFG 计数并在 IFG 间隔结束后使能发送器。MAC 在半双工模式下工作时，实施截断二进制指数回退算法。

发送流量控制

在全双工模式下，当发送流量控制使能位（ETH_MACFCR 中的 TFE 位）置 1 时，MAC 将生成暂停帧并根据需要发送暂停帧。暂停帧与计算出的 CRC 附加在一起并发送。可以通过两种方式启动暂停帧的生成。

当应用程序将 ETH_MACFCR 寄存器中的 FCB 位置 1 或接收 FIFO 已满（数据包缓冲区）时，将发送暂停帧。

- 如果应用程序已通过将 ETH_MACFCR 寄存器中的 FCB 位置 1 来请求流量控制，则 MAC 将生成并发送单个暂停帧。生成的帧中的暂停时间值包含 ETH_MACFCR 中编程的暂停时间值。要在先前发送的暂停帧中指定的时间之前延长或结束暂停时间，应用程序必须以适当的值编程暂停时间值（ETH_MACFCR 寄存器中的 PT），然后请求另一次暂停帧发送。
- 如果接收 FIFO 填满时应用程序已请求流量控制，则 MAC 将生成并发送暂停帧。生成的帧中的暂停时间值为 ETH_MACFCR 中编程的暂停时间值。如果在此暂停时间结束前，接收 FIFO 在可配置的时隙数（ETH_MACFCR 中的 PLT 位）期间保持填满状态，将发送第二个暂停帧。只要接收 FIFO 保持填满状态，该过程将一直重复下去。如果在采样时间之前不再满足此条件，MAC 将发送一个暂停时间为零的暂停帧，向远程端表明接收缓冲区已准备好接收新数据帧。

单数据包发送操作

发送操作的常规事件序列如下：

1. 如果系统有要传输的数据，则 DMA 控制器将通过 AHB 主接口从存储器中获取这些数据并将其转发给 FIFO。它将继续接收数据直到传输帧结束。
2. 当超过阈值或 FIFO 接收到完整的数据包时，数据会被弹出并被传输到 MAC 内核。DMA 继续从 FIFO 传输数据，直到将完整的数据包传输到 MAC 为止。帧传输完成后，来自 MAC 的状态将通知 DMA 控制器。

发送操作——缓冲区中有两个数据包

1. 由于 DMA 必须先更新描述符状态才能将其释放给主机，因此一个发送 FIFO 内最多只能有两个帧。仅当 OSF（对第二个帧起作用）位置 1 时，DMA 才会获取第二个帧并将其置于 FIFO 中。如果此位未置 1，则仅当 MAC 完全处理该帧且 DMA 已释放描述符后，DMA 才会从存储器获取下一帧。
2. 如果 OSF 位已置 1，则 DMA 将第一个帧传输到 FIFO 后将立即开始获取第二个帧。不会等待状态更新。同时，在发送第一个帧时，FIFO 接收到第二个帧。第一帧的数据传输完成后，来自 MAC 的状态会通知 DMA。如果 DMA 已向 FIFO 发送第二个数据包，则第二次发送必须等待第一个数据包的状态，才能继续下一帧。

冲突期间的重新发送

在半双工模式下，向 MAC 传输帧时，可能在 MAC 线接口上发生冲突事件。MAC 甚至会在接收到帧结束之前就给出状态来指示重试。然后将使能重新发送并再次将帧从 FIFO 中弹出。当超过 96 个字节弹向 MAC 内核后，FIFO 控制器将释放该空间，使 DMA 可推入更多数据。这意味着超过阈值后或 MAC 内核指示延迟冲突事件时，无法重新发送。

发送 FIFO 刷新操作

MAC 为软件提供了一个控制权，使其可通过操作模式寄存器中的位 20 来清空发送 FIFO。清空操作是立即操作，即使 Tx FIFO 正在向 MAC 内核传输帧，Tx FIFO 和相应的指针也会清零到初始状态。这将导致 MAC 发送器中生成下溢事件，并且帧发送将中止。此类帧的状态将同时标记下溢和帧清空事件（TDES0 位 13 和 1）。清空操作期间，没有数据从应用程序 (DMA) 传输到 FIFO。根据清空的帧数（包括局部帧），将相应数量的传输发送状态字传输到应用程序。完全清空的帧的帧清空状态位 (TDES0 13) 将置 1。当应用程序 (DMA) 接受所有已清空的帧的状态字后，清空操作完成。发送 FIFO 清空控制寄存器位随后将清零。此时，将接受来自应用程序 (DMA) 的新帧。清空操作完成后，将丢弃所有为发送提交的数据，除非数据以 SOF 标记开头。

发送状态字

在向 MAC 内核传输以太网帧结束时以及内核完成帧的发送后，发送状态将提供给应用程序。发送状态的详细说明与 TDES0 中的位 [23:0] 相同。如果使能 IEEE 1588 时间戳功能，将返回特定帧的 64 位时间戳以及发送状态。

发送校验和减荷

通信协议（例如 TCP 和 UDP）将实施校验和字段，这有助于确定通过网络发送的数据的完整性。由于以太网最广泛的用途是通过 IP 数据报封装 TCP 和 UDP，因此以太网控制器具有发送校验和减荷功能，该功能支持校验和计算、发送路径中的校验和插入以及接收路径中的错误检测。本节将介绍发送帧的校验和减荷功能的操作。

注： *通过完整的帧来计算 TCP、UDP 或 ICMP 的校验和，然后将其插入相应的报头字段。由于此要求，仅当发送 FIFO 配置为存储转发模式（即，ETH_ETH_DMAOMR 寄存器中的 TSF 位置 1）时，才使能此功能。如果内核配置为阈值（直通）模式，则将绕过发送校验和减荷。*

必须确保发送 FIFO 足够深，使其能存储一个完整的帧，以便将该帧传输到 MAC 内核发送器。如果 FIFO 的深度小于输入以太网帧的太小，则将绕过有效负载 (TCP/UDP/ICMP) 校验和插入功能，并且仅修改帧的 IPv4 报头校验和，即使在存储转发模式下也是如此。

发送校验和减荷支持两种类型的校验和计算和插入。可通过将 CIC 位（TDES1 中的位 28:27，在*第 1394 页上的TDES1：发送描述符字 1*中介绍）置 1 来控制每个帧的校验和。

有关 IPv4、TCP、UDP、ICMP、IPv6 和 ICMPv6 数据包头规范的信息，请分别参见 IETF 规范 RFC 791、RFC 793、RFC 768、RFC 792、RFC 2460 和 RFC 4443。

- IP 报头校验和

  在 IPv4 数据报中，报头字段的完整性由 16 位头校验和字段（IPv4 数据报的第十一个字节和第十二个字节）指示。当以太网帧的类型字段值为 0x0800 且 IP 数据报的版本字段值为 0x4 时，校验和减荷将检测到 IPv4 数据报。计算期间，将忽略输入帧的校验和字段并将其替换为计算出的值。IPv6 报头没有校验和字段；因此，校验和减荷不修改 IPv6 报头字段。此 IP 报头校验和计算的结果由发送状态中的 IP 报头错误状态位（位 16）指示。只要以太网类型字段的值和 IP 报头版本字段的值不一致，或当以太网帧没有足够的数据（如 IP 报头长度字段所指示）时，此状态位将置 1。换言之，在以下情况下发生 IP 报头错误时，此位将置 1：

  a) 对于 IPv4 数据报：

  – 接收的以太网类型为 0x0800，但 IP 报头的版本字段不等于 0x4

  – IPv4 报头长度字段指示小于 0x5（20 字节）的值

  – 总的帧长度小于 IPv4 报头长度字段给定的值

  b) 对于 IPv6 数据报：

  – 以太网类型为 0x86DD，但 IP 报头版本字段不等于 0x6

  – 帧在 IPv6 报头（40 字节）之前结束，或已完全接收到扩展报头（如扩展报头中相应的报头长度字段中给定）。如果以太网类型字段指示 IPv4 有效负载，即使校验和减荷检测到 IP 报头错误，也会插入 IPv4 报头校验和。

- TCP/UDP/ICMP 校验和

  TCP/UDP/ICMP 校验和对 IPv4 或 IPv6 报头（包括扩展报头）进行处理，并确定封装的有效负载是 TCP、UDP 还是 ICMP。

  请注意：

  a) 对于非 TCP、非 UDP 或非 ICMP/ICMPv6 的有效负载，将绕过此校验和且不会对帧进行修改。

  b) 将绕过分段的 IP 帧（IPv4 或 IPv6）、带安全功能（例如，验证报头或封装的安全有效负载）的 IP 帧和带路由报头的 IPv6 帧，并且校验和不对这些帧进行处理。

  计算 TCP、UDP 或 ICMP 有效负载的校验和，然后将其插入报头中相应的字段。它可工作在以下两种模式：

  – 在第一种模式下，TCP、UDP 或 ICMPv6 伪报头并未包含在校验和计算中，并假定其存在于输入帧的校验和字段中。校验和字段包含在校验和计算中，然后替换为最终计算出的校验和。

  – 在第二种模式下，将忽略校验和字段，TCP、UDP 或 ICMPv6 伪报头数据包含在校验和计算中，并使用最终计算出的值覆盖校验和字段。

  请注意：对于 ICMP-over-IPv4 数据包，由于没有为其定义伪报头，因此在这两种模式下 ICMP 数据包中的校验和字段都必须始终为 0x0000。如果不等于 0x0000，可能向数据包插入不正确的校验和。

  此操作的结果由发送状态向量中的有效负载校验和错误状态位（位 12）指示。当检测到下列情况之一时，有效负载校验和错误状态位置 1：

  – 在存储转发模式下，帧已转发到 MAC 发送器，但帧结束未写入 FIFO。

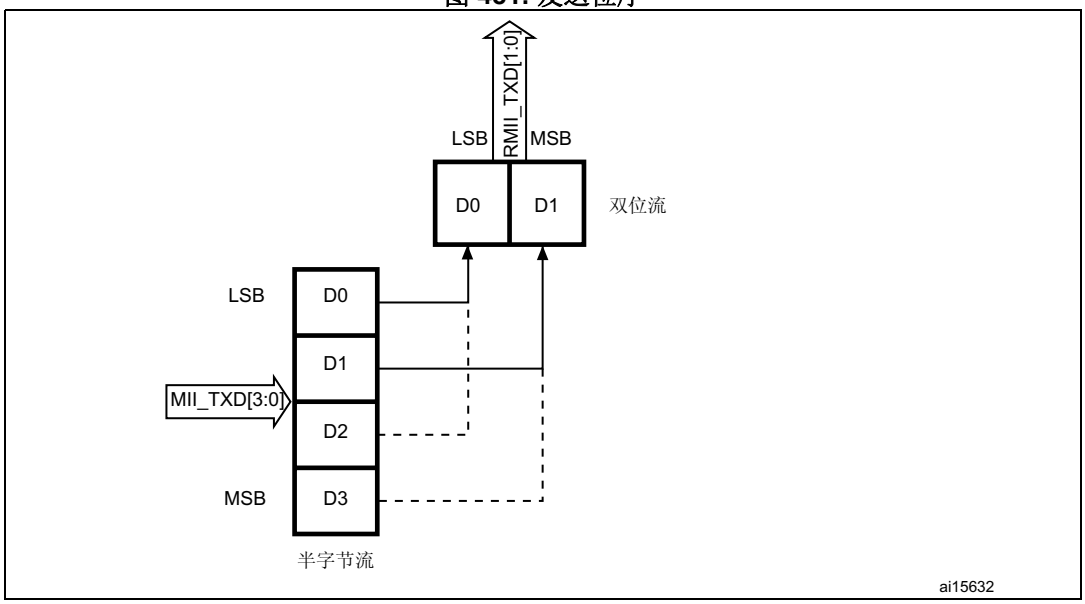
  – 在接收到 IP 报头中的有效负载长度字段指示的字节数前，数据包已结束。

  当数据包的长度大于指示的有效负载长度时，将字节作为填充字节忽略，并且不报告错误。检测到第一种类型的错误时，不修改 TCP、UDP 或 ICMP 报头。对于第二种错误类型，计算的校验和仍将插入相应的报头字段。

MII/RMII 发送位序

来自 MII 的每个半字节都在 RMII 上发送，一次发送双位，双位的发送顺序如*图 481* 所示。首先发送位序较低的位（D1 和 D0），再发送位序较高的位（D2 和 D3）。

图 481. 发送位序



MII/RMII 发送时序图

图 482. 无冲突发送

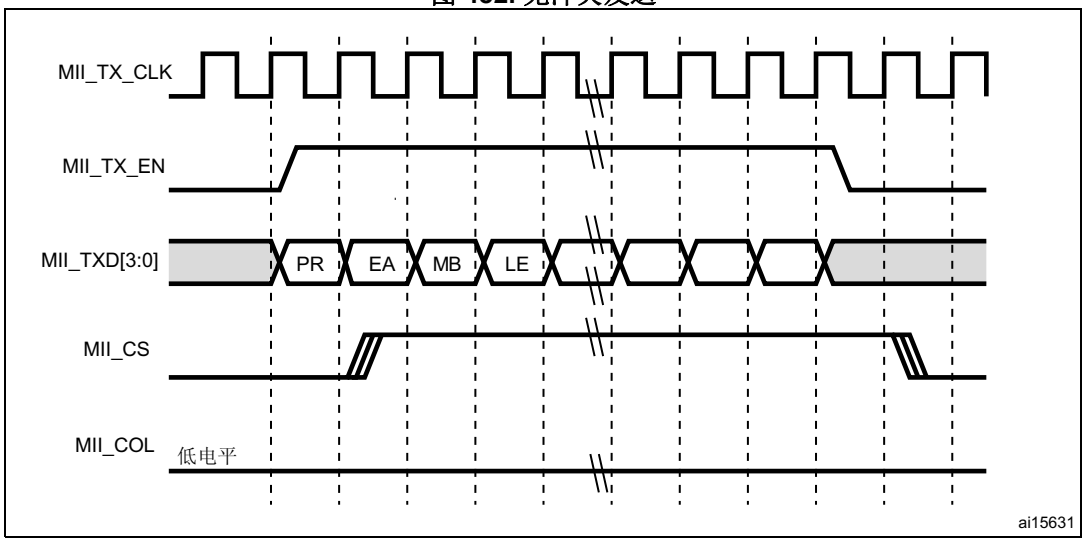


图 483. 有冲突发送

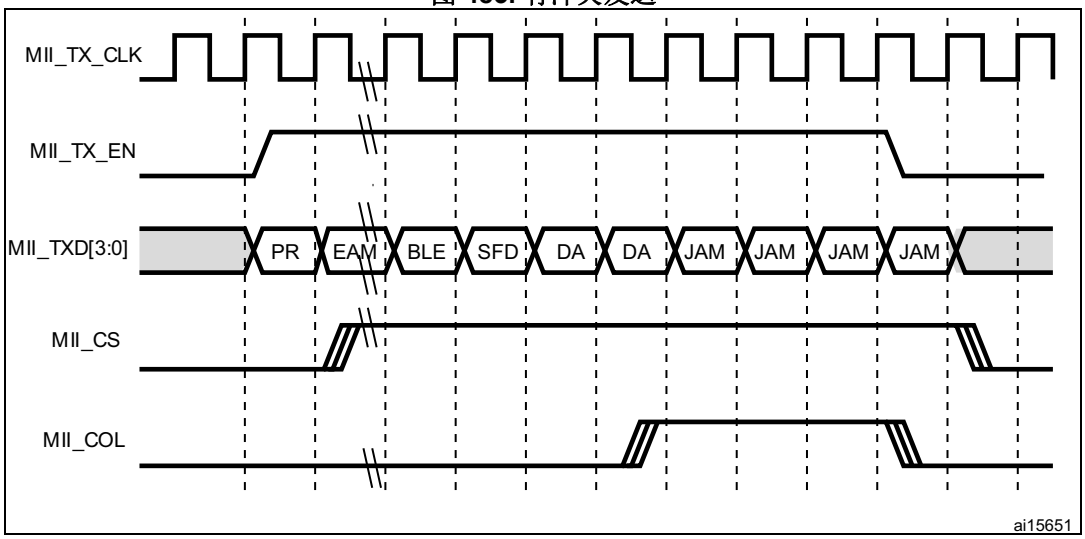
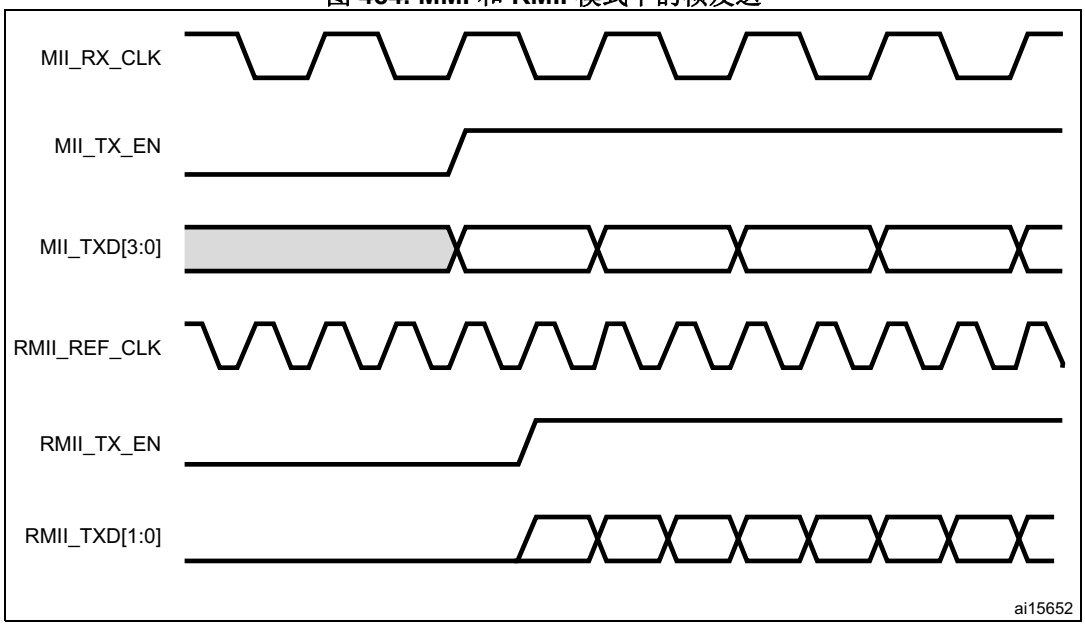


图 484 显示了 MII 和 RMII 中的帧发送。

图 484. MMI 和 RMII 模式下的帧发送



38.5.3 MAC 帧接收

MAC 接收的帧将推入 Rx FIFO。此 FIFO 的状态（填充级别）一旦超过配置的接收阈值（ETH_DMAOMR 寄存器中的 RTC），就会将其指示给 DMA，这样 DMA 可向 AHB 接口发起预配置的突发传输。

在默认直通模式下，当 FIFO 接收到 64 个字节（使用 ETH_DMAOMR 寄存器中的 RTC 位配置）或完整的数据包时，数据将弹出，其可用性将通知给 DMA。DMA 向 AHB 接口发起传输后，数据传输将从 FIFO 持续进行，直到传输完整个数据包。完成 EOF 帧的传输后，状态字将弹出并发送到 DMA 控制器。

在 Rx FIFO 存储转发模式（通过 ETH_DMAOMR 寄存器中的 RSF 位配置）下，仅在帧完全写入接收 FIFO 后才可读出帧。在此模式下，将丢弃所有错误帧（如果内核配置为执行此操作），这样只会读出有效帧并将其转发到应用程序。在直通模式下，某些错误帧不会被丢弃，因为在帧结束时接收到错误状态，而此时已从 FIFO 读出帧开始。

当 MAC 在 MII 上检测到 SFD 时，将启动接收操作。MAC 内核将去除报头和 SFD，然后再继续处理帧。检查报头字段以进行过滤，FCS 字段用于验证帧的 CRC。如果帧未通过地址滤波器，则在内核中丢弃该帧。

接收协议

去除接收的帧的报头和 SFD。检测到 SFD 后，MAC 开始向接收 FIFO 发送以太网帧数据，从 SFD 后面的第一个字节（目标地址）开始发送。如果使能 IEEE 1588 时间戳功能，则在 MII 上检测到任何帧的 SFD 时，都将获取系统时间的快照。除非 MAC 滤出并丢弃帧，否则此时间戳将传递给应用程序。

如果接收的帧长度/类型字段小于 0x600 并且为 MAC 编程了自动去除 CRC/pad 选项，则 MAC 将向接收 FIFO 发送帧数据（数据量不超过长度/类型字段中指定的数量），然后开始丢弃字节（包括 FCS 字段）。如果长度/类型字段大于或等于 0x600，则不管编程的自动 CRC 去除选项的值如何，MAC 都会向 Rx FIFO 发送所有接收到的以太网帧数据。默认情况下，使能 MAC 看门狗定时器，即，超过 2048 个字节（DA + SA + LT + 数据 + pad + FCS）的帧会被切断。可通过对 MAC 配置寄存器中的看门狗禁止 (WD) 位编程来禁止此功能。但是，即使禁止看门狗定时器，仍将切断大于 16 KB 的帧并给出看门狗超时状态。

接收 CRC：自动 CRC 和 pad 去除

MAC 将检查接收帧中的任何 CRC 错误。它将计算接收的帧（包括目标地址字段到 FCS 字段）的 32 位 CRC。编码由下面的多项式定义。

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

不管是否自动去除 pad/CRC，MAC 都将接收整个帧来计算所接收帧的 CRC 校验。

接收校验和减荷

对接收的以太网帧中的 IPv4 和 IPv6 帧进行检测和处理，以确保数据完整性。可通过将 ETH_MACCR 寄存器中的 IPCO 位置 1 来使能接收校验和减荷。MAC 通过检查所接收的以太网帧的类型字段中是否存在值 0x0800 或 0x86DD，来识别 IPv4 或 IPv6 帧。此识别方法也适用于带 VLAN 标记的帧。接收校验和减荷将计算 IPv4 报头校验和，并检查其是否与接收的 IPv4 报头校验和匹配。如果指示的有效负载类型（以太网类型字段）与 IP 报头版本之间有任何不匹配，或接收的帧的字节数小于 IPv4 报头的长度字段中指示的数量（IPv4 或 IPv6 报头中的可用字节数少于 20），IP 报头错误位将置 1。接收校验和减荷还可以识别接收的 IP 数据报（IPv4 或 IPv6）中的 TCP、UDP 或 ICMP 有效负载，并正确计算此类有效负载的校验和，如 TCP、UDP 或 ICMP 规范中所定义。它包括用于校验和计算的 TCP/UDP/ICMPv6 伪报头字节，并检查接收的校验和字段是否与计算的值匹配。此操作的结果由接收状态字中的有效负载校验和错误位给出。如果 TCP、UDP 或 ICMP 有效负载的长度与 IP 报头中给出的预期有效负载长度不匹配，此状态位也将置 1。如*第 1367 页上的 TCP/UDP/ICMP 校验和*中所述，接收校验和减荷将绕过分段 IP 数据报的有效负载、带安全功能的 IP 数据报、IPv6 路由报头以及除 TCP、UDP 或 ICMP 以外的有效负载。此信息在接收状态中给出（无论是否绕过校验和），如 *RDES0：接收描述符字 0* 一节所述。在此配置中，内核不会向接收的以太网帧附加任何有效负载校验和字节。

如*第 1400 页上的RDES0：接收描述符字 0* 中所述，某些寄存器位的含义会发生变化，如*表 240* 所示。

表 240. 帧状态

位 18：以太网帧	位 27：报头校验和错误	位 28：有效负载校验和错误	帧状态
0	0	0	帧为 IEEE 802.3 帧（长度字段值小于 0x0600）。
1	0	0	IPv4/IPv6 类型帧，在其中未检测到校验和错误。
1	0	1	IPv4/IPv6 类型帧，在其中检测到有效负载校验和错误（参见 PCE 的说明）。
1	1	0	IPv4/IPv6 类型帧，在其中检测到 IP 报头校验和错误（参见 IPCO HCE 的说明）。
1	1	1	IPv4/IPv6 类型帧，在其中检测到 PCE 和 IPCO HCE。
0	0	1	IPv4/IPv6 类型帧，其中不存在 IP HCE，并且由于不支持的有效负载而绕过有效负载检查。
0	1	1	既不是 IPv4 也不是 IPv6 的类型帧（校验和减荷完全绕过校验和的校验）。
0	1	0	保留。

接收帧控制器

如果复位 MAC CSR 帧过滤寄存器中的 RA 位，则 MAC 将根据目标/源地址执行帧过滤（如果应用程序决定不接收任何不良帧，如矮帧、CRC 错误帧等，则仍需要执行另一等级的过滤）。检测到过滤失败时，帧将被丢弃且不会传输到应用程序。当过滤参数动态改变时，如果 (DA-SA) 过滤失败，则剩余的帧将被丢弃并且接收状态字将立即更新（零帧长度位、CRC 错误位和矮帧错误位将置 1），指示过滤失败。在以太网掉电模式下，所有接收到的帧都将被丢弃且不会转发给应用程序。

接收流量控制

MAC 将检测接收暂停帧并暂停帧发送，暂停时间为接收的暂停帧内指定的延迟（仅限全双工模式）。可通过 ETH_MACFCR 中的 RFCE 位使能或禁止暂停帧检测功能。使能接收流量控制后，将开始监视接收帧的目标地址是否与控制帧的多播地址 (0x0180 C200 0001) 匹配。如果检测到匹配（接收的帧的目标地址与保留的控制帧的目标地址匹配），MAC 将根据 ETH_MACFFR 中的 PCF 位来决定是否将接收的控制帧传输到应用程序。

MAC 还将对接收控制帧的类型、操作码和暂停定时器字段进行解码。如果状态的字节计数指示 64 个字节，并且不存在任何 CRC 错误，则 MAC 发送器将暂停任何数据帧的发送，暂停时间为解码的暂停时间值乘以时隙（对于 10/100 Mb/s 模式，均为 64 字节时间）。同时，如果检测到另一个零暂停时间值的暂停帧，MAC 将复位暂停时间并管理新的暂停请求。

如果接收的控制帧与类型字段 (0x8808)、操作码 (0x00001) 以及字节长度（64 字节）均不匹配，或存在 CRC 错误，则 MAC 不会生成暂停。

如果暂停帧具有多播目标地址，MAC 将根据地址匹配来过滤帧。

对于具有单播目标地址的暂停帧，MAC 将根据 DA 是否与 MAC 地址 0 寄存器的内容匹配以及 ETH_MACFCR 中的 UPDF 位是否置 1（检测具有单播目标地址的暂停帧）来进行过滤。PCF 寄存器位（ETH_MACFFR 中的位 [7:6]）可对控制帧的过滤以及地址过滤进行控制。

接收操作多帧处理

由于接收数据后状态立即可用，因此只要 FIFO 未满，就可以向其中存储帧。

错误处理

如果在从 MAC 接收 EOF 数据之前 Rx FIFO 已满，则将声明上溢并丢弃整个帧，（ETH_DMAMFBOCR 寄存器）中的上溢计数器将递增。由于上溢，状态将指示这是一个部分帧。如果（使用 ETH_DMAOMR 中的 FEF 和 FUGF 位）使能相应功能，Rx FIFO 可过滤错误帧和过小帧。
如果将接收 FIFO 配置为在存储转发模式下工作，则可过滤并丢弃所有错误帧。
在直通模式下，如果在从 Rx FIFO 读取帧的 SOF 时，该帧的状态和长度可用，则可丢弃整个错误帧。DMA 可通过使能接收帧清空位来清空正在从 FIFO 读取的错误帧。然后到应用 (DMA) 的数据传输将停止，其余帧将从内部读取并丢弃。如果可用，则可以启动下一帧传输。

接收状态字

以太网帧接收结束时，MAC 向应用 (DMA) 输出接收状态。接收状态的详细说明与 RDES0 中的位 [31:0] 相同，请参见*第 1400 页上的RDES0：接收描述符字 0*。

帧长度接口

对于开关应用，应用和 MAC 之间的数据发送和接收以传输完整帧的形式进行。为了向出站端口传输帧，应用层应知道从入站端口接收的帧的长度。在每次帧接收结束时，MAC 内核在状态内提供每个接收的帧的长度。

注： *对于由于上溢而写入 Rx FIFO 的部分帧，将指定值为 0 的帧长度。*

MII/RMII 接收位序

每个半字节都从接收自 RMII 的双位发送到 MII，半字节发送顺序如*图 485* 所示。先接收位序较低的位（D0 和 D1），再接收位序较高的位（D2 和 D3）。

图 485. 接收位序

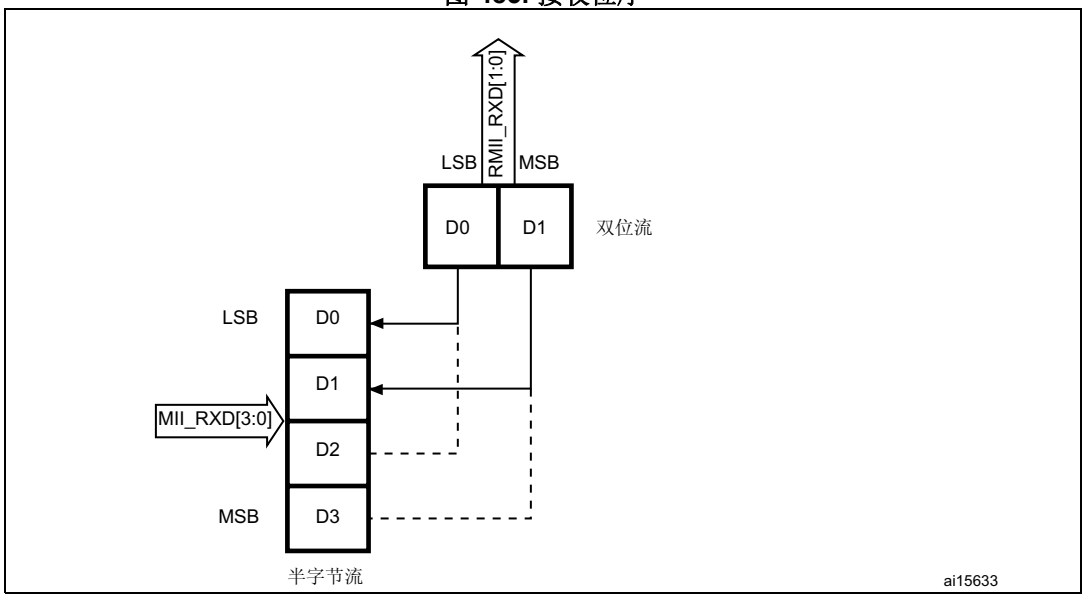


图 486. 无错误接收

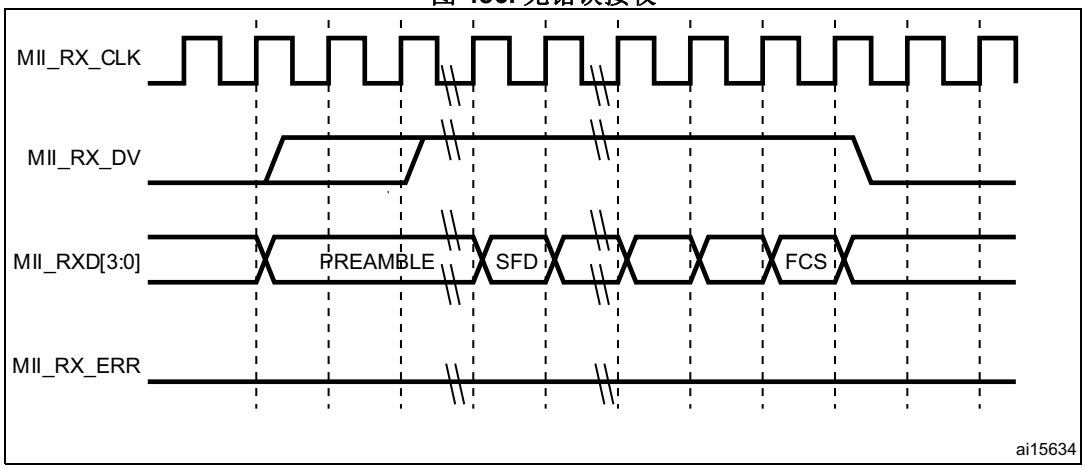


图 487. 有错误接收

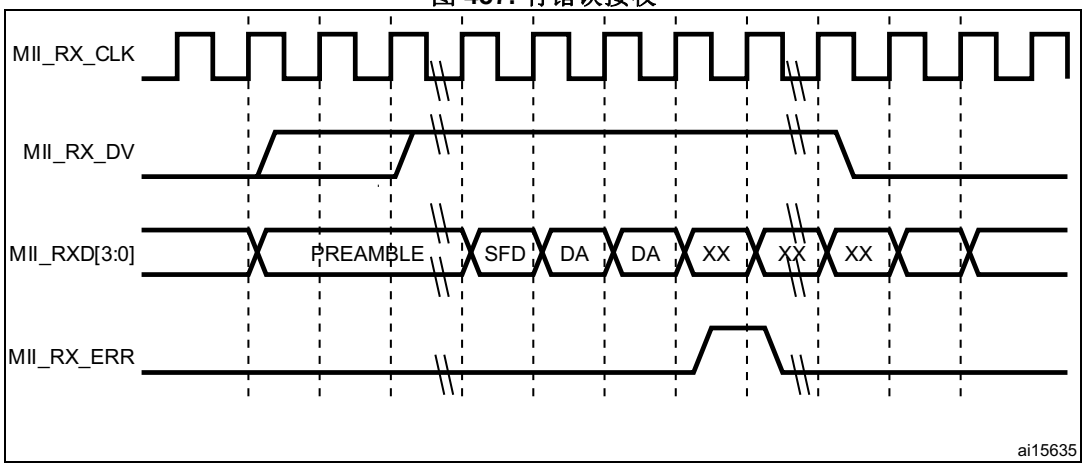
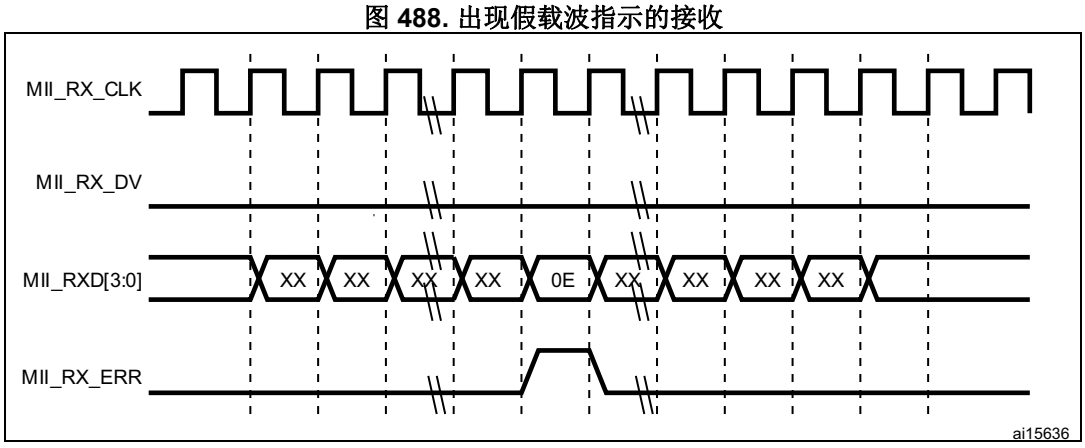


图 488. 出现假载波指示的接收



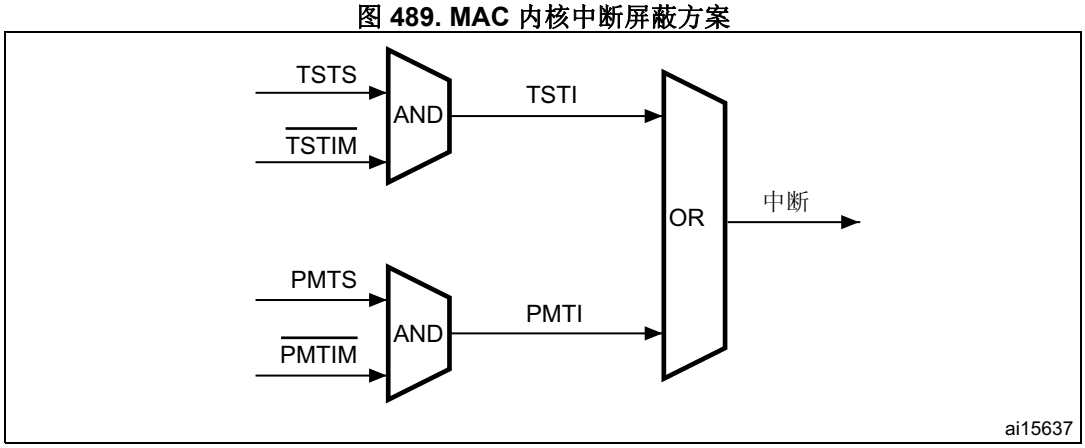
38.5.4 MAC 中断

MAC 内核可由于各种事件而生成中断。

ETH_MACSR 寄存器描述了可导致 MAC 内核生成中断的事件。可通过将中断屏蔽寄存器中相应的屏蔽位置 1 来阻止各事件生成中断。

中断寄存器位仅指示报告事件的模块。必须读取相应的状态寄存器和其它寄存器才能清除中断。例如，如果中断寄存器的位 3 设置为高电平，则指示在掉电模式下接收到魔术数据包或网络唤醒帧。必须读取 ETH_MACPMTCSR 寄存器才能清除此中断事件。

图 489. MAC 内核中断屏蔽方案



38.5.5 MAC 过滤

地址过滤

地址过滤将检查所有接收的帧的目标地址和源地址并相应报告地址过滤状态。地址检查基于应用程序选择的不同参数（帧过滤寄存器）。还可以识别过滤的帧：多播帧或广播帧。

地址过滤使用站的物理 (MAC) 地址和多播散列表进行地址检查。

单播目标地址过滤

MAC 支持多达 4 个用于单播完美过滤的 MAC 地址。如果选择完美过滤（复位帧过滤寄存器中的 HU 位），MAC 会将接收的单播地址的所有 48 位与编程的 MAC 地址进行比较来确定是否匹配。默认情况下，始终使能 MacAddr0，其它地址 MacAddr1–MacAddr3 则通过单独的使能位进行选择。将其它地址 (MacAddr1–MacAddr3) 的各个字节与接收的相应 DA 字节进行比较时，可以将寄存器中相应的屏蔽字节控制位置 1 来屏蔽该字节。这有助于 DA 的组地址过滤。在散列过滤模式（HU 位置 1）下，MAC 将使用 64 位散列表执行对单播地址的不完美过滤。对于散列过滤，MAC 将使用接收的目标地址的 6 个高 CRC（参见下文注释 1）位来索引散列表的内容。值为 000000 时，选取所选寄存器中的位 0；值为 111111 时，选取散列表寄存器中的位 63。如果相应位（由 6 位 CRC 指示）已置 1，将认为单播帧已通过散列过滤，否则认为帧未能通过散列过滤。

注： *CRC 是使用下列多项式编码的 32 位值（有关详细信息，请参见第 38.5.3 节：MAC 帧接收）：*

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

多播目标地址过滤

可通过将帧过滤寄存器中的 PAM 位置 1，将 MAC 编程为通过所有多播帧。如果 PAM 位复位，MAC 将根据帧过滤寄存器中的 HM 位执行对多播地址的过滤。在完美过滤模式下，将多播地址与编程的 MAC 目标地址寄存器 (1–3) 进行比较。组地址过滤也受到支持。在散列过滤模式下，MAC 将使用 64 位散列表执行不完美过滤。对于散列过滤，MAC 将使用接收的多播地址的 6 个高 CRC（参见下文注释 1）位来索引散列表的内容。值为 000000 时，选取所选寄存器中的位 0；值为 111111 时，选取散列表寄存器中的位 63。如果相应位已置 1，将认为多播帧已通过散列过滤，否则认为帧未能通过散列过滤。

注： *CRC 是使用下列多项式编码的 32 位值（有关详细信息，请参见第 38.5.3 节：MAC 帧接收）：*

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

散列或完美地址过滤

可通过将帧过滤寄存器中的 HPF 位置 1 并将相应的 HU 或 HM 位置 1，来将 DA 过滤配置为在其 DA 与散列过滤或完美过滤匹配时允许帧通过。此配置适用于单播帧和多播帧。如果 HPF 位复位，则只有一种过滤方式（散列或完美）应用于接收的帧。

广播地址过滤

在默认模式下，MAC 不过滤任何广播帧。但是，如果将帧过滤寄存器中的 BFD 位置 1 来将 MAC 编程为拒绝所有广播帧，则会丢弃任何广播帧。

单播源地址过滤

MAC 还可以根据接收的帧的源地址字段来执行完美过滤。默认情况下，MAC 将 SA 字段与 SA 寄存器中编程的值进行比较。可通过将相应寄存器中的位 30 置 1，来将 MAC 地址寄存器 [1:3] 配置为包含 SA 而不是 DA 进行比较。带 SA 的组地址过滤也受到支持。如果帧过滤寄存器中的 SAF 位置 1，则 MAC 将丢弃未通过 SA 过滤的帧。否则，SA 过滤的结果将通过接收状态字中的状态位给出（参见 *RDES0：接收描述符字 0*）。

SAF 位置 1 时，对 SA 过滤和 DA 过滤的结果进行与运算，以决定是否需要转发帧。这意味着任何一个过滤未通过都将丢弃帧。两个过滤必须都通过，才能将帧转发到应用。

反向过滤操作

对于目标地址和源地址过滤，可在最终输出时选择反转过滤匹配结果。这分别由帧过滤寄存器中的 DAIF 和 SAIF 位控制。DAIF 位同时适用于单播和多播 DA 帧。在此模式下，将反转单播/多播目标地址过滤的结果。类似地，当 SAIF 位置 1 时，将反转单播 SA 过滤的结果。*表 241* 和 *表 242* 按照接收帧的类型汇总了目标地址和源地址过滤。

表 241. 目标地址过滤

帧类型	PM	HPF	HU	DAIF	HM	PAM	DB	DA 过滤操作
广播	1	X	X	X	X	X	X	通过
	0	X	X	X	X	X	0	通过
	0	X	X	X	X	X	1	不通过
单播	1	X	X	X	X	X	X	通过所有帧
	0	X	0	0	X	X	X	完美/组过滤匹配时通过
	0	X	0	1	X	X	X	完美/组过滤匹配时不通过
	0	0	1	0	X	X	X	散列过滤匹配时通过
	0	0	1	1	X	X	X	散列过滤匹配时不通过
	0	1	1	0	X	X	X	散列或完美/组过滤匹配时通过
	0	1	1	1	X	X	X	散列或完美/组过滤匹配时不通过
多播	1	X	X	X	X	X	X	通过所有帧
	X	X	X	X	X	1	X	通过所有帧
	0	X	X	0	0	0	X	完美/组过滤匹配时通过，并在 PCF = 0x 时丢弃暂停控制帧
	0	0	X	0	1	0	X	散列过滤匹配时通过，并在 PCF = 0x 时丢弃暂停控制帧
	0	1	X	0	1	0	X	散列或完美/组过滤匹配时通过，并在 PCF = 0x 时丢弃暂停控制帧
	0	X	X	1	0	0	X	完美/组过滤匹配时不通过，并在 PCF = 0x 时丢弃暂停控制帧
	0	0	X	1	1	0	X	散列过滤匹配时不通过，并在 PCF = 0x 时丢弃暂停控制帧
	0	1	X	1	1	0	X	散列或完美/组过滤匹配时不通过，并在 PCF = 0x 时丢弃暂停控制帧

表 242. 源地址过滤

帧类型	PM	SAIF	SAF	SA 过滤操作
单播	1	X	X	通过所有帧
	0	0	0	完美/组过滤匹配时通过，但不丢弃未通过的帧
	0	1	0	完美/组过滤匹配时不通过，但不丢弃帧
	0	0	1	完美/组过滤器匹配时通过并将不通过的帧丢弃
	0	1	1	完美/组过滤器匹配时不通过并将不通过的帧丢弃

38.5.6 MAC 回送模式

MAC 支持对发送到其接收器的帧进行回送。默认情况下，禁止 MAC 回送功能，可通过编程 MAC ETH_MACCR 寄存器中的 Loopback 位使能该功能。

38.5.7 MAC 管理计数器：MMC

MAC 管理计数器 (MMC) 保持一组寄存器来收集有关已接收帧和已发送帧的统计信息。其中包括一个用于控制各寄存器行为的控制寄存器、两个包含当前中断状态的 32 位寄存器（接收和发送），以及两个包含中断寄存器掩码的 32 位寄存器（接收和发送）。这些寄存器均可从应用程序中访问。每个寄存器均为 32 位宽。

*第 38.8 节：以太网寄存器说明*介绍了各个计数器，并列出了各统计计数器的地址。该地址用于对所需发送/接收计数器进行读/写访问。

接收 MMC 计数器针对通过地址过滤的帧进行更新。已丢弃帧的统计信息不会进行更新，除非丢弃的帧为小于 6 字节的矮帧（DA 字节不能完全接收）。

好的发送帧与接收帧

如果帧成功发送，则将发送的帧视为“好帧”。换句话说，如果帧发送过程没有因为以下错误而中止，则认为发送的帧是好帧：

+ Jabber 超时
+ 无载波/载波丢失
+ 延迟冲突
+ 帧下溢
+ 过度延迟
+ 过度冲突

如果不存在以下错误，则认为接收帧是“好帧”：

+ CRC 错误
+ 矮帧（短于 64 字节）
+ 对齐错误（仅限 10/100 Mb/s）
+ 长度错误（仅限非类型帧）
+ 超出范围（仅限非类型帧，超过最大大小）
+ MII_RXER 输入错误

最大帧大小取决于帧类型，如下：

+ 无标记帧的最大大小 = 1518
+ VLAN 帧的最大大小 = 1522

38.5.8 电源管理：PMT

本部分介绍 MAC 支持的电源管理 (PMT) 机制。PMT 支持接收网络（远程）唤醒帧和魔术数据包帧。PMT 可为 MAC 接收到的唤醒帧和魔术数据包帧生成中断。可通过远程唤醒帧使能位以及魔术数据包使能位使能 PMT 模块。这些使能位（WFE 和 MPE）位于 ETH_MACPMTCSR 寄存器中，可由应用编程。在 PMT 中使能掉电模式时，MAC 将丢弃所有接收到的帧并且不会将这些帧转发给应用。仅当接收到魔术数据包或远程唤醒帧且使能相应检测时，MAC 才会退出掉电模式。

远程唤醒帧过滤寄存器

有八个唤醒帧过滤寄存器。要对每个寄存器执行写操作，需要逐个值加载唤醒帧过滤寄存器。连续加载唤醒帧过滤寄存器八次，便可对唤醒帧过滤寄存器加载所需的值。读操作与写操作相同。要读取八个值，必须读唤醒帧过滤寄存器八次后，才能到达最后一个寄存器。每次读/写操作都会将唤醒帧过滤寄存器指向下一个过滤寄存器。

图 490. 唤醒帧过滤寄存器



- 过滤器 i 字节屏蔽

  该寄存器定义过滤器 i（0、1、2 和 3）检测帧的哪些字节来确定帧是否为唤醒帧。MSB（第 31 位）必须为零。位 j [30:0] 为字节屏蔽。如果将字节屏蔽的位 j（字节数）置 1，则传入帧的过滤器 i 偏移 + j 由 CRC 模块处理；否则将忽略过滤器 i 偏移 + j。

- 过滤器 i 命令

  该 4 位命令控制过滤器 i 操作。位 3 指定地址类型，定义模式的目标地址类型。当该位置 1 时，模式只适用于多播帧。当该位复位时，模式只适用于单播帧。位 2 和位 1 为保留位。位 0 为过滤器 i 的使能位；如果位 0 未置 1，则将禁止过滤器 i。

- 过滤器 i 偏移

  该寄存器定义过滤器 i 要检测的帧的偏移（在帧范围内）。该 8 位模式偏移是要检测的过滤器 i 的第一个字节的偏移。允许的最小值为 12，表示帧的第 13 个字节（偏移值 0 表示帧的第一个字节）。

- 过滤器 i CRC-16

  该寄存器包含根据模式计算的 CRC_16 值，以及对唤醒过滤寄存器模块编程的字节屏蔽。

远程唤醒帧检测

当 MAC 处于睡眠模式并已使能 ETH_MACPMTCSR 寄存器中的远程唤醒位时，MAC 可在接收到远程唤醒帧后恢复正常工作。应用可通过连续对唤醒帧过滤寄存器地址执行写操作，来写入全部八个唤醒过滤寄存器。应用可通过向 ETH_MACPMTCSR 寄存器中的位 2 写入 1 来使能远程唤醒。PMT 支持四种可编程过滤器，这些过滤器提供不同的接收帧模式。如果传入帧通过过滤命令的地址过滤，并且过滤器 CRC-16 与传入的检测模式相匹配，则可接收唤醒帧。Filter_offset（最小值 12，表示帧的第 13 个字节）确定要检测的帧的偏移。过滤器字节屏蔽确定必须检测帧的哪些字节。必须将字节屏蔽的第 31 位置 0。只需要检查唤醒帧是否存在长度错误、FCS 错误、帧尾错误、MII 错误、冲突，确保其不是矮帧。即使唤醒帧长度超过 512 字节，但只要帧的 CRC 值有效，帧便有效。ETH_MACPMTCSR 寄存器中的唤醒帧检测会针对每个接收到的远程唤醒帧进行更新。此外，还会生成 PMT 中断（如果已使能）来指示已接收到远程唤醒帧。

魔术数据包检测

魔术数据包帧基于一种特殊的方法，这种方法使用 AMD 公司的魔术数据包技术对网络上处于睡眠模式下的器件上电。MAC 接收称为魔术数据包的特定信息包，此信息包的目标地址是网络上的节点。只对目标地址为器件或多播地址的魔术数据包进行检查，以确定这些数据包是否满足唤醒要求。对通过地址过滤（单播或多播）的魔术数据包进行检测，确定其是否符合远程唤醒数据格式，即 6 个字节的数据所有位为全 '1' 的数据包加上重复出现 16 次的 MAC 地址。应用通过向 ETH_MACPMTCSR 寄存器中的位 1 写入 1 来使能魔术数据包唤醒。PMT 模块持续监视目标地址为对应于指定魔术数据包模式的节点的每个帧。检查每个接收帧的目标地址和源地址字段之后是否为 0xFFFF FFFF FFFF 形式。之后，PMT 模块检查帧是否存在重复 16 次且没有任何断开或中断的 MAC 地址。如果重复 16 次的地址中存在中断，则会再次扫描传入帧中是否存在 0xFFFF FFFF FFFF 形式。这 16 次重复可位于帧中的任何位置，但必须在同步数据流后面 (0xFFFF FFFF FFFF)。此外，只要检测到重复 16 次的 MAC 地址，器件就可以接收多播帧。如果节点的 MAC 地址为 0x0011 2233 4455，则 MAC 扫描的数据序列为：

目标地址源地址 .....FFFF FFFF FFFF
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
…CRC

ETH_MACPMTCSR 寄存器中的魔术数据包检测会针对接收到的魔术数据包进行更新。此外，还会生成 PMT 中断（如果已使能）来指示已接收到魔术数据包。

掉电期间的系统的注意事项

如果使能 EXTI 中断线 19，以太网 PMT 模块能够在系统处于停止模式时检测帧。

MAC 接收器状态机在掉电模式期间应保持使能状态。这意味着，由于 ETH_MACCR 寄存器中的 RE 位与魔术数据包/网络唤醒帧检测有关，因此必须始终将该位置 1。但是，在掉电模式期间，应通过将 ETH_MACCR 寄存器中的 TE 位清零来关闭发送器状态机。此外，在掉电模式期间，应禁止以太网 DMA，原因是此时无需将魔术数据包/网络唤醒帧复制到 SRAM。要禁止以太网 DMA，需将 ETH_DMAOMR 寄存器中的 ST 位和 SR 位（分别对应发送 DMA 和接收 DMA）清零。

推荐的掉电和唤醒顺序如下：

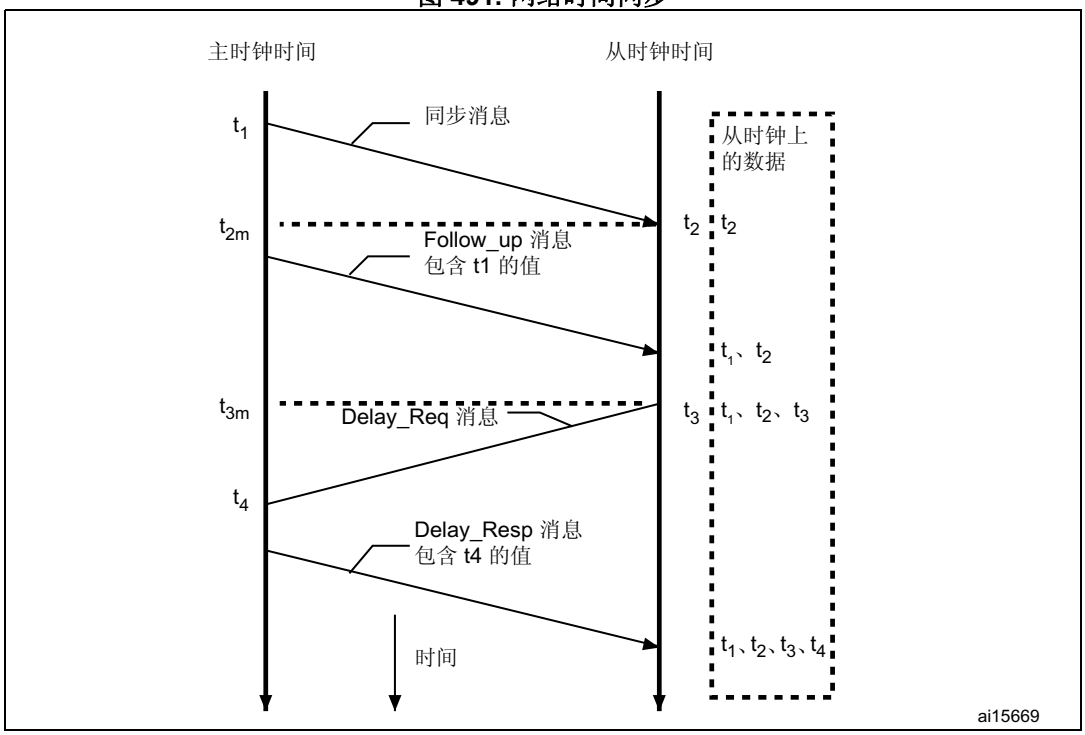
1. 禁止发送 DMA，并等待所有之前的帧发送完成。接收到发送中断 ETH_DMASR 寄存器 [0] 时，便可检测到这些帧发送。
2. 通过将 ETH_MACCR 配置寄存器中的 RE 位和 TE 位清零来禁止 MAC 发送器和 MAC 接收器。
3. 等待接收 DMA 清空 Rx FIFO 中的所有帧。
4. 禁止接收 DMA。
5. 配置并使能 EXTI 中断线 19，以生成一个事件或中断。
6. 如果将 EXTI 中断线 19 配置为生成中断，还必须正确配置 ETH_WKUP_IRQ 处理程序功能，此功能用于清零 EXTI 中断线 19 的挂起位。
7. 通过将 ETH_MACPMTCSR 寄存器中的 MFE/WFE 位置 1 来使能魔术数据包/网络唤醒帧检测。
8. 通过将 ETH_MACPMTCSR 寄存器中的 PD 位置 1 来使能 MAC 掉电模式。
9. 通过将 ETH_MACCR 寄存器中的 RE 位置 1 来使能 MAC 接收器。
10. 进入系统的停止模式（有关更多详细信息，请参见*第 4.3.5 节：停止模式*）。

11. 接收到有效唤醒帧时，以太网外设会退出掉电模式。
12. 读取 ETH_MACPMTCSR 以清零电源管理事件标志，使能 MAC 发送器状态机并接收和发送 DMA。
13. 配置系统时钟：使能 HSE 并设置时钟。

38.5.9 精密时间协议 (IEEE1588 PTP)

IEEE 1588 标准定义了一种协议，此协议支持使用网络通信、局域计算和分布式对象等技术实现的测量和控制系统中的精密时钟同步。该协议适用于利用支持多播消息传送的局域网（包括但不限于以太网）进行通信的系统。该协议用于对非均匀系统进行同步，这类系统包含固有精度、分辨率和稳定性都不断变化的时钟。该协议支持亚秒范围的系统级同步精度，并且需要极少的网络和本地时钟计算资源。这种基于消息的协议称为精密时间协议 (PTP)，它通过 UDP/IP 传送。系统或网络归类为主节点和从节点，用于分配时序/时钟信息。该协议通过交换 PTP 消息来同步从节点与主节点，*图 491* 中对此有详细说明。

图 491. 网络时间同步



1. 主节点向其所有节点广播 PTP 同步消息。同步消息包含主节点的参考时间信息。消息离开主节点系统的时间为 t_1。对于以太网端口，必须通过 MII 接口来捕获该时间。
2. 从节点接收同步消息并利用其参考时序捕获准确时间 t_2。
3. 主节点随后会向从节点发送一个包含 t_1 信息的 Follow_up 消息以备后用。
4. 从节点向主节点发送一个 Delay_Req 消息，标出该帧离开 MII 的准确时间 t_3。
5. 主节点接收该消息，捕获消息进入其系统的准确时间 t_4。
6. 主节点将 Delay_Resp 消息中的 t_4 信息发送给从节点。
7. 从节点使用 t_1、t_2、t_3 和 t_4 这四个值来同步其本地参考时序与主节点的参考时序。

大多数协议都在 UDP 层之上通过软件实现。但如上所述，要通过 MII 接口来捕获特定 PTP 包进入或离开以太网端口时的准确时间，需要硬件支持。必须捕获该时序信息并将其返回给软件，以实现高精度的 PTP。

参考时序源

根据 IEEE 1588 规范的定义，要获取时间快照，内核需要一个 64 位格式的参考时间（分成两个 32 位通道，高 32 位表示时间的秒数，低 32 位表示时间的纳秒数）。

PTP 参考时钟输入用于在内部生成参考时间（亦称作系统时间）以及捕获时间戳。该参考时钟的频率必须大于等于时间戳计数器的分辨率。主节点与各从节点之间的同步精度目标约为 100 ns。

*系统时间校准方法*一节中介绍了系统时间的生成、更新与修改。

精度取决于 PTP 参考时钟输入周期、振荡器的特性（漂移）以及同步过程的频率。

由于从 Tx 和 Rx 时钟输入域到 PTP 参考时钟域的同步，时间戳锁存值的不确定度为 1 个参考时钟周期。如果加上分辨率导致的不确定度，则会增加一半的时间戳周期。

使用 PTP 功能发送帧

在 MII 上输出帧的 SFD 时，将捕获时间戳。对于需要捕获时间戳的帧，可按帧为单位进行控制。换句话说，可对每个发送帧进行标记以指示是否有必要捕获该帧的时间戳。无需处理发送帧即可识别 PTP 帧。可通过发送描述符中的控制位执行帧控制。捕获到的时间戳返回给应用的方式与提供帧状态的方式相同。时间戳会随帧的发送状态一同发送回相应的发送描述符内，从而自动将时间戳与特定 PTP 帧相连。64 位时间戳信息会写回 TDES2 和 TDES3 字段，其中 TDES2 会保持时间戳的 32 个最低有效位。

使用 PTP 功能接收帧

使能 IEEE 1588 时间戳功能时，以太网 MAC 将捕获 MII 上接收到的所有帧的时间戳。MAC 会在完成帧接收过程时提供时间戳。捕获到的时间戳返回给应用的方式与提供帧状态的方式相同。时间戳会随帧的接收状态一同发送回相应的接收描述符内。64 位时间戳信息会写回 RDES2 和 RDES3 字段，其中 RDES2 会保持时间戳的 32 个最低有效位。

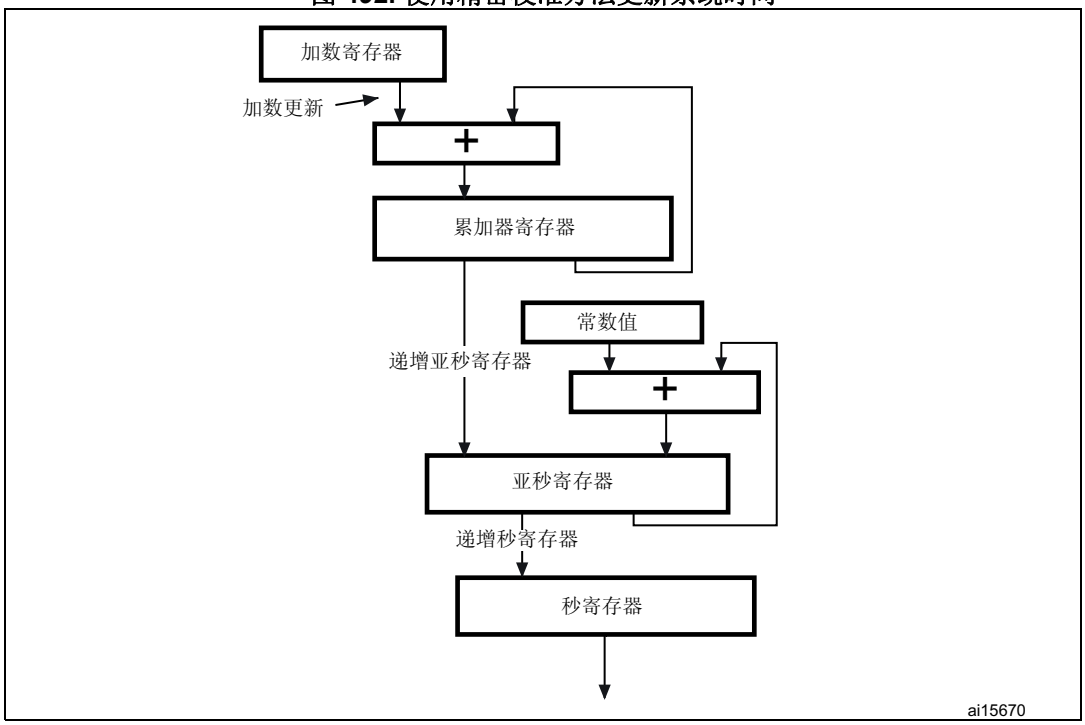
系统时间校准方法

使用 PTP 输入参考时钟 HCLK 更新 64 位 PTP 时间。该 PTP 时间可用作时钟源，以获取 MII 上发送或接收的以太网帧的快照（时间戳）。可使用粗略校准方法或精密校准方法对系统时间计数器进行初始化或校准。

使用粗略校准方法时，初始值或偏移值会写入时间戳更新寄存器（请参见*第 1430 页上的第 38.8.3 节：IEEE 1588 时间戳寄存器*）。对于初始化，会将时间戳更新寄存器中的值写入系统时间计数器；对于系统时间校准，会将偏移值（时间戳更新寄存器）加到系统时间中或从系统时间中减去。

使用精密校准方法时，从时钟（参考时钟）频率相对于主时钟（如 IEEE 1588 中定义）的偏移会在一段时间内进行校准，而不像粗略校准方法中那样，在单个时钟周期内进行校准。校准时间越长，越有助于保持线性时间，并且不会导致各 PTP 同步消息间隔之间的参考时间发生剧烈变化（或者大型抖动）。在此方法中，会使用一个累加器对加数寄存器中的内容求和，如*图 492* 所示。累加器生成的算数进位将用作使系统时间计数器递增的脉冲。累加器和加数寄存器均为 32 位寄存器。此处，累加器用作高精度频率乘法器或除法器。*图 492* 给出了该算法。

图 492. 使用精密校准方法更新系统时间



系统时间更新逻辑需要使用 50 MHz 的时钟频率，以达到 20 ns 的精度。分频是指参考时钟频率与所需时钟频率的比率。因此，如果我们说参考时钟 (HCLK) 的频率为 66MHz，则通过计算可知分频比为 66 MHz/50 MHz = 1.32。故此，寄存器中设置的默认加数值为 2^{32}/1.32，相当于 0xC1F0 7C1F。

如果参考时钟向下偏移，例如降至 65 MHz，则分频比为 65/50 或 1.3，加数寄存器中要设置的值为 2^{32}/1.30，相当于 0xC4EC 4EC4。如果时钟向上偏移，例如升至 67 MHz，则必须将加数寄存器设置为 0xBF0 B7672。当时钟偏移为零时，应编程的默认加法值为 0xC1F0 7C1F (2^{32}/1.32)。

在*图 492* 中，用于使亚秒寄存器递增的常数值为 0d43。这可使系统时间的精度达到 20 ns（换句话说，增量步长时间为 20 ns）。

软件必须根据同步消息对频率偏移进行计算，并相应更新加数寄存器。首先，使用加数寄存器中的 FreqCompensationValue0 设置从时钟。该值的计算公式如下：

$$FreqCompensationValue0 = 2^{32} / FreqDivisionRatio$$

如果起初假定 MasterToSlaveDelay 对于连续的同步消息是相同的，则必须应用下述算法。数个同步周期之后，频率会锁定。从时钟随后可确定 MasterToSlaveDelay 的精确值，并使用新值重新与主时钟同步。

该算法如下：

- 在 MasterSyncTime (n) 时刻，主时钟向从时钟发送同步消息。从时钟在其本地时钟为 SlaveClockTime (n) 时接收到该消息，并用如下公式计算 MasterClockTime (n)：
  MasterClockTime (n) = MasterSyncTime (n) + MasterToSlaveDelay (n)
- 当前同步周期的主时钟计数 MasterClockCount (n) 的计算公式如下：
  MasterClockCount (n) = MasterClockTime (n) – MasterClockTime (n – 1)（假定 MasterToSlaveDelay 对于同步周期 n 和 n – 1 是相同的）
- 当前同步周期的从时钟计数 SlaveClockCount (n) 的计算公式如下：
  SlaveClockCount (n) = SlaveClockTime (n) – SlaveClockTime (n – 1)
- 当前同步周期的主从时钟计数差值 SlaveClockCount (n) 的计算公式如下：
  ClockDiffCount (n) = MasterClockCount (n) – SlaveClockCount (n)
- 从时钟的分频系数 FreqScaleFactor (n) 的计算公式如下：
  FreqScaleFactor (n) = (MasterClockCount (n) + ClockDiffCount (n)) / SlaveClockCount (n)
- 加数寄存器的频率补偿值 FreqCompensationValue (n) 的计算公式如下：
  FreqCompensationValue (n) = FreqScaleFactor (n) × FreqCompensationValue (n – 1)

理论上，该算法可在一个同步周期内实现锁定；但是，由于网络传播延迟和工作条件会不断变化，因此可能需要多个周期。

该算法可进行自校准：如果出于某些原因导致最初通过主时钟设置的从时钟不正确，则该算法会花费更多同步周期将其校准。

系统时间生成初始化的编程步骤

通过将时间戳控制寄存器 (ETH__PTPTSCR) 中的位 0 置 1 来使能时间戳功能。但之后必须对时间戳计数器进行初始化才能启动时间戳操作。适当的顺序如下：

1. 通过将 MACIMR 寄存器中的位 9 置 1 以屏蔽时间戳触发中断。
2. 编程时间戳寄存器位 0 以使能时间戳。
3. 根据 PTP 时钟频率编程亚秒递增计数器。
4. 如果使用精密校准方法，编程时间戳加数寄存器并将时间戳控制寄存器的位 5 置 1（加数寄存器更新）。
5. 轮询时间戳控制寄存器，直到位 5 清零。
6. 要选择精密校准方法（根据需要），编程时间戳控制寄存器位 1。
7. 用适当的时间值编程时间戳高位更新寄存器和时间戳低位更新寄存器。
8. 将时间戳控制寄存器位 2 置 1（时间戳初始化）。
9. 用时间戳更新寄存器中写入的值初始化时间戳计数器后，时间戳计数器便开始运行。
10. 使能 MAC 接收器和发送器以使时间戳功能正常运行。

注： *如果将 ETH_PTPTSCR 寄存器中的位 0 清零来禁止时间戳操作，必须重复执行以上步骤以重新启动时间戳操作。*

使用粗略校准方法更新系统时间的编程步骤

要在一个过程（粗略校准方法）中同步或更新系统时间，请按照以下步骤执行：

1. 将偏移（正偏移或负偏移）写入时间戳更新高位和低位寄存器。
2. 将时间戳控制寄存器中的位 3 (TSSTU) 置 1。
3. 当 TSSTU 位清零时，时间戳更新寄存器中的值将加到系统时间中或者从系统时间中减去。

使用精密校准方法更新系统时间的编程步骤

要同步或更新系统时间以减少系统时间抖动（精密校准方法），请按照以下步骤执行：

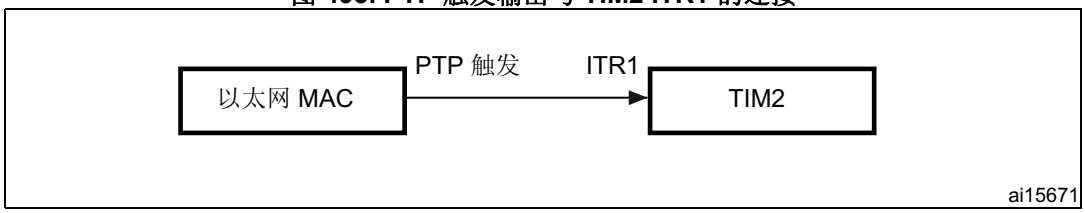
1. 借助*系统时间校准方法*一节中介绍的算法，计算想要增加或减少系统时间增量时采用的速率。
2. 更新时间戳。
3. 等待加数寄存器中的新值生效。这可以通过在系统时间达到目标值后，激活时间戳触发中断来实现。
4. 在目标时间高位和低位寄存器中编程所需目标时间。通过将 ETH_MACIMR 寄存器中的位 9 清零以取消屏蔽时间戳中断。
5. 将时间戳控制寄存器位 4 (TSARU) 置 1。
6. 当该寄存器产生中断时，读取 ETH_MACSR 寄存器。
7. 用旧值重新编程时间戳加数寄存器并再次将 ETH_TPTSCR 位 5 置 1。

与 TIM2 内部相连的 PTP 触发

当系统时间大于目标时间时，MAC 会提供触发中断。使用中断会引起已知延迟，并导致命令执行时间出现不确定性。

为避免这种不确定性，可在系统时间大于目标时间时将 PTP 触发输出信号置为高电平。该信号内部连接到 TIM2 输入触发。通过该信号，由已同步的 PTP 系统时间触发的输入捕获功能、输出比较功能以及定时器的波形均可使用。这时不会引起不确定性，因为定时器的时钟（PCLK1：TIM2 APB1 时钟）以及 PTP 参考时钟 (HCLK) 是同步的。
该 PTP 触发信号会连接到可用软件选择的 TIM2 ITR1 输入。该连接可通过 TIM2 选项寄存器 (TIM2_OR) 中的位 11 和 10 使能。*图 493* 显示了该连接。

图 493. PTP 触发输出与 TIM2 ITR1 的连接



PTP 每秒脉冲数输出信号

该 PTP 脉冲输出用于检查网络中所有节点之间的同步性。要能够测试本地从时钟与主参考时钟之间的差值，可为两个时钟提供每秒脉冲数 (PPS) 输出信号（根据需要，可将该信号连接到示波器）。这样便可测量两个信号间的偏差。PPS 输出的脉冲宽度为 125 ms。

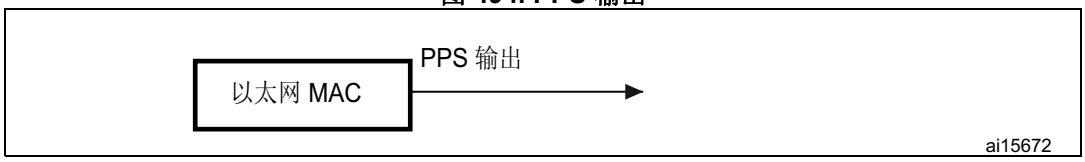
PPS 输出可通过 TIM2 选项寄存器 (TIM2_OR) 中的位 11 和 10 使能。

PPS 输出的默认频率为 1 Hz。可使用 PPSFREQ[3:0]（位于 ETH_PTPPPSCR 中）将 PPS 输出的频率设置为 $2^{PPSFREQ}$ Hz。

如果设置为 1 Hz，则使用二进制翻转（TSSSR=0，ETH_PTPTSCR 寄存器中的位 9）时，PPS 脉冲宽度为 125 ms；使用数字翻转 (TSSSR=1) 时，PPS 脉冲宽度为 100 ms。如果设置为 2 Hz 或更高频率，则使用二进制翻转时，PPS 输出的占空比为 50%。

使用数字翻转 (TSSSR=1) 时，建议不要使用频率为 1 Hz 以外的 PPS 输出，因为波形可能不规则（尽管其平均频率可能在任意一秒窗口期间始终正确）。

图 494. PPS 输出



38.6 以太网功能说明：DMA 控制器操作

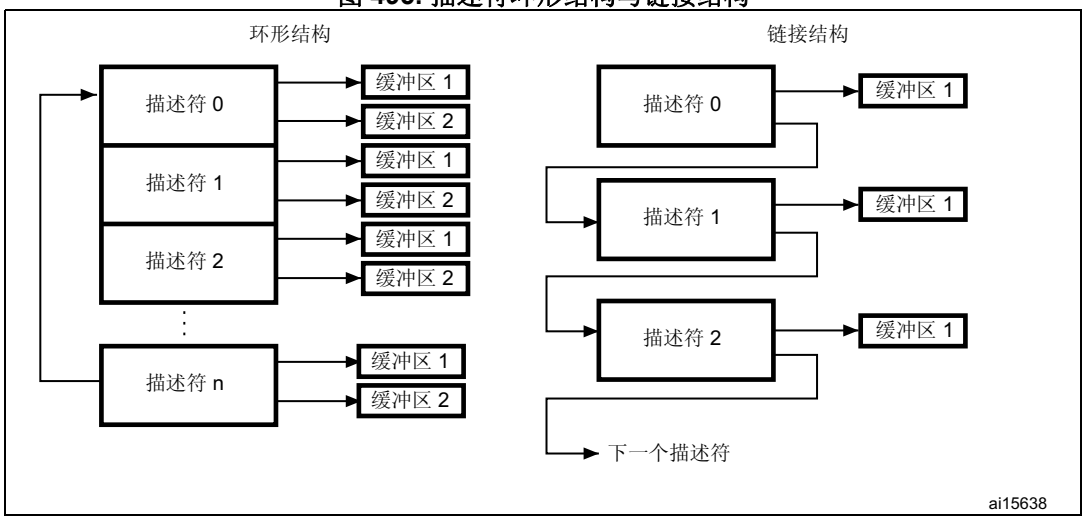
DMA 具有独立的发送和接收引擎以及相应的 CSR（控制和状态寄存器）空间。发送引擎将数据从系统存储器传送到 Tx FIFO，而接收引擎将数据从 Rx FIFO 传送到系统存储器。DMA 可以在 CPU 完全不干预的情况下，通过描述符有效地将数据从源传送到目标。DMA 专为面向包的数据传送（如以太网中的帧）而设计。该控制器经过编程后，可在完成帧发送和接收传送操作时以及其它正常 / 错误条件下产生 CPU 中断。DMA 以及 STM32F75xxx 和 STM32F74xxx 通过以下两种数据结构实现通信：

- 控制和状态寄存器 (CSR)
- 描述符列表和数据缓冲区。

*第 1408 页上的第 38.8 节*详细介绍了控制和状态寄存器。*第 1391 页*详细介绍了描述符。

DMA 既可将接收到的数据帧传送到 STM32F75xxx 和 STM32F74xxx 存储器中的接收缓冲区，也可以传送 STM32F75xxx 和 STM32F74xxx 存储器的发送缓冲区中的 1 数据帧。位于 STM32F75xxx 和 STM32F74xxx 存储器中的描述符用作指向这些缓冲区的指针。共有两个描述符列表：一个用于接收，一个用于发送。两个列表的基址分别写入 DMA 寄存器 3 和寄存器 4。描述符列表是一种前向链表（无论是隐式还是显式）。最后一个描述符会指回第一个描述符以构成环形结构。可通过配置接收和发送描述符（RDES1[14] 和 TDES0[20]）中链接的第二个地址来完成描述符的显式链接。描述符列表位于主机的物理存储空间。每个描述符最多可指向两个缓冲区。这样便能使用两个物理寻址的缓冲区替代存储器中两个连续的缓冲区。数据缓冲区位于主机的物理存储空间，通常由整个帧或部分帧组成，但不会超过单个帧。缓冲区中仅包含数据。其状态保存在描述符中。数据链接是指跨越多个数据缓冲区的帧。但是单个描述符不能跨越多个帧。检测到帧结束时，DMA 会跳到下一个帧缓冲区。可以使能或禁止数据链接。描述符的环形结构与链接结构如*图 495* 所示。

图 495. 描述符环形结构与链接结构



38.6.1 使用 DMA 进行传送的初始化

MAC 的初始化步骤如下：

1. 对 ETH_DMABMR 执行写操作以设置 STM32F75xxx 和 STM32F74xxx 总线访问参数。
2. 对 ETH_DMAIER 寄存器执行写操作以屏蔽不必要的中断源。
3. 软件驱动器创建发送和接收描述符列表。随后对 ETH_DMARDLAR 和 ETH_DMATDLAR 寄存器执行写操作，为 DMA 提供各列表的起始地址。
4. 对 MAC 寄存器 1、2 和 3 执行写操作以选择所需的过滤选项。
5. 对 MAC ETH_MACCR 寄存器执行写操作以配置和使能发送与接收工作模式。可根据自动协商结果将 PS 位与 DM 位置 1（读取自 PHY）。
6. 对 ETH_DMAOMR 寄存器执行写操作，将位 13 和位 1 置 1 以启动发送和接收。
7. 发送与接收引擎进入运行状态，并尝试从相应描述符列表中获取描述符。这两个引擎随后开始处理接收和发送操作。发送和接收处理过程彼此独立，可单独进行启动或停止。

38.6.2 主机总线突发访问

DMA 会尝试在 AHB 主接口上执行固定长度的突发传送（如果已使用 ETH_DMABMR 中的 FB 位进行相应配置）。突发的最大长度由 PBL 字段 (ETH_DMABMR [13:8]) 指示和限制。对于要读取的 16 个字节，接收和发送描述符始终采用可能的最大突发大小（由 PBL 限制）进行访问。

仅当发送 FIFO 中的空间足以容纳配置的突发或帧结束之前的字节数时（当帧短于配置的突发长度时），发送 DMA 才会启动数据传送。DMA 会向 AHB 主接口指示起始地址和所需的传送次数。当 AHB 接口配置为固定长度突发时，将使用 INCR4、INCR8、INCR16 和单独事务的最佳搭配来传送数据。否则（非固定长度突发）会使用 INCR（未定义长度）与单独事务传送数据。

仅当接收 FIFO 中有足够的数据用于配置的突发时，或者在接收 FIFO 中检测到帧结束时（当帧短于配置的突发长度时），接收 DMA 才会启动数据传送。DMA 会向 AHB 主接口指示起始地址和所需的传送次数。当 AHB 接口配置为固定长度突发时，将使用 INCR4、INCR8、INCR16 和单独事务的最佳搭配来传送数据。如果在 AHB 接口上的固定突发结束前已到达帧结束，将执行空传送以完成固定长度的突发传送。否则（复位 ETH_DMABMR 中的 FB 位）会使用 INCR（未定义长度）和单独事务传送数据。

当 AHB 接口配置为地址对齐的节拍时，两个 DMA 引擎会确保 AHB 启动的第一次突发传送小于或者等于已配置 PBL 的大小。这样，后续的所有节拍都会从与已配置 PBL 对齐的地址开始。由于 AHB 接口多于 INCR16 的传送，DMA 只能对齐节拍最大为 16（对于 PBL > 16）的地址。

38.6.3 主机数据缓冲区对齐

发送和接收数据缓冲区在起始地址对齐方面没有任何限制。在我们的系统（32 位寻址空间）中，缓冲区的起始地址可与四个字节中的任意一个对齐。但是，DMA 始终在地址与总线宽度对齐时启动传输，并且在不需要的字节通道上传输空数据。这通常发生在以太网帧的开始或结束传送期间。

- 缓冲区读操作示例：

  如果发送缓冲区地址为 0x0000 0FF2，并且需要传送 15 个字节，则 DMA 将从地址 0x0000 0FF0 读取 5 个全字，但在将数据传送到发送 FIFO 时，将丢弃或忽略额外的字节（前两个字节）。同样地，还将忽略最后一次传送的最后 3 个字节。DMA 始终可确保向发送 FIFO 传送的是全 32 位数据项，除非是帧结束。

- 缓冲区写操作示例：

  如果接收缓冲区地址为 0x0000 0FF2，并且需要传送接收到的帧的 16 个字节，则 DMA 将从地址 0x0000 0FF0 开始写入 5 个全 32 位数据项。但是，第一次传送的前 2 个字节与第三次传送的最后 2 个字节将包含空数据。

38.6.4 缓冲区大小计算

DMA 不会更新发送和接收描述符中的大小字段。DMA 只更新描述符的状态字段 (xDES0)。必须用驱动程序计算大小。发送 DMA 会向 MAC 内核传送准确的字节数（由 TDES1 中的缓冲区大小字段指示）。如果将描述符标记为第一个描述符（将 TDES0 中的 FS 位置 1），则 DMA 会将缓冲区的第一次传送标记为帧起始。如果将描述符标记为最后一个描述符（TDES0 中的 LS 位），则 DMA 会将数据缓冲区的最后一次传送标记为帧结束。接收 DMA 持续将数据传送至缓冲区，直到缓冲区已满或接收到帧结束为止。当描述符的 FS 位置 1 时，如果未将描述符标记为最后一个描述符（RDES0 中的 LS 位），则该描述符对应的缓冲区会填满，并且缓冲区中的有效数据量将通过缓冲区大小字段减去数据缓冲指针偏移得到的结果表示。当数据缓冲区指针与数据总线宽度对齐时，偏移为零。如果将描述符标记为最后一个描述符，则缓冲区不会填满（由 RDES1 中的缓冲区大小指示）。要计算该缓冲区中最终的有效数据量，驱动程序必须读取帧长度（RDES0[29:16] 中的 FL 位），并减去该帧中先前缓冲区的缓冲区大小之和。接收 DMA 始终使用新的描述符传送下一个帧的帧起始。

注： *即使当接收缓冲区的起始地址与系统数据总线宽度未对齐时，系统也应分配一个大小与系统总线宽度对齐的接收缓冲区。例如，如果系统分配一个起始地址为 0x1000、大小为 1024 字节 (1 KB) 的接收缓冲区，则软件可将接收描述符中的缓冲区起始地址编程为具有 0x1002 偏移。接收 DMA 将帧写入该缓冲区，其中前两个单元（0x1000 和 0x1001）中为空数据。实际帧从单元 0x1002 开始写入。因此，尽管已将缓冲区大小编程为 1024 字节，但由于存在起始偏移地址，因此该缓冲区的实际有用空间为 1022 字节。*

38.6.5 DMA 仲裁器

DMA 内的仲裁器会在发送和接收通道对 AHB 主接口进行的访问之间进行仲裁。可以使用两类仲裁：循环调度和固定优先级。如果选择循环调度仲裁（复位 ETH_DMABMR 中的 DA 位），仲裁器会在发送和接收 DMA 同时请求访问时，按照 ETH_DMABMR 中的 PM 位设置的比率分配数据总线。当 DA 位置 1 时，接收 DMA 进行数据访问的优先级始终高于发送 DMA。

38.6.6 对 DMA 的错误响应

对于由 DMA 通道发起的任何数据传送，如果从机给出错误响应，则相应 DMA 将停止所有操作并更新状态寄存器（ETH_DMASR 寄存器）中的错误位和致命总线错误位。此外，该 DMA 控制器只能在软复位或硬复位外设以及重新初始化 DMA 之后才能恢复操作。

38.6.7 Tx DMA 配置

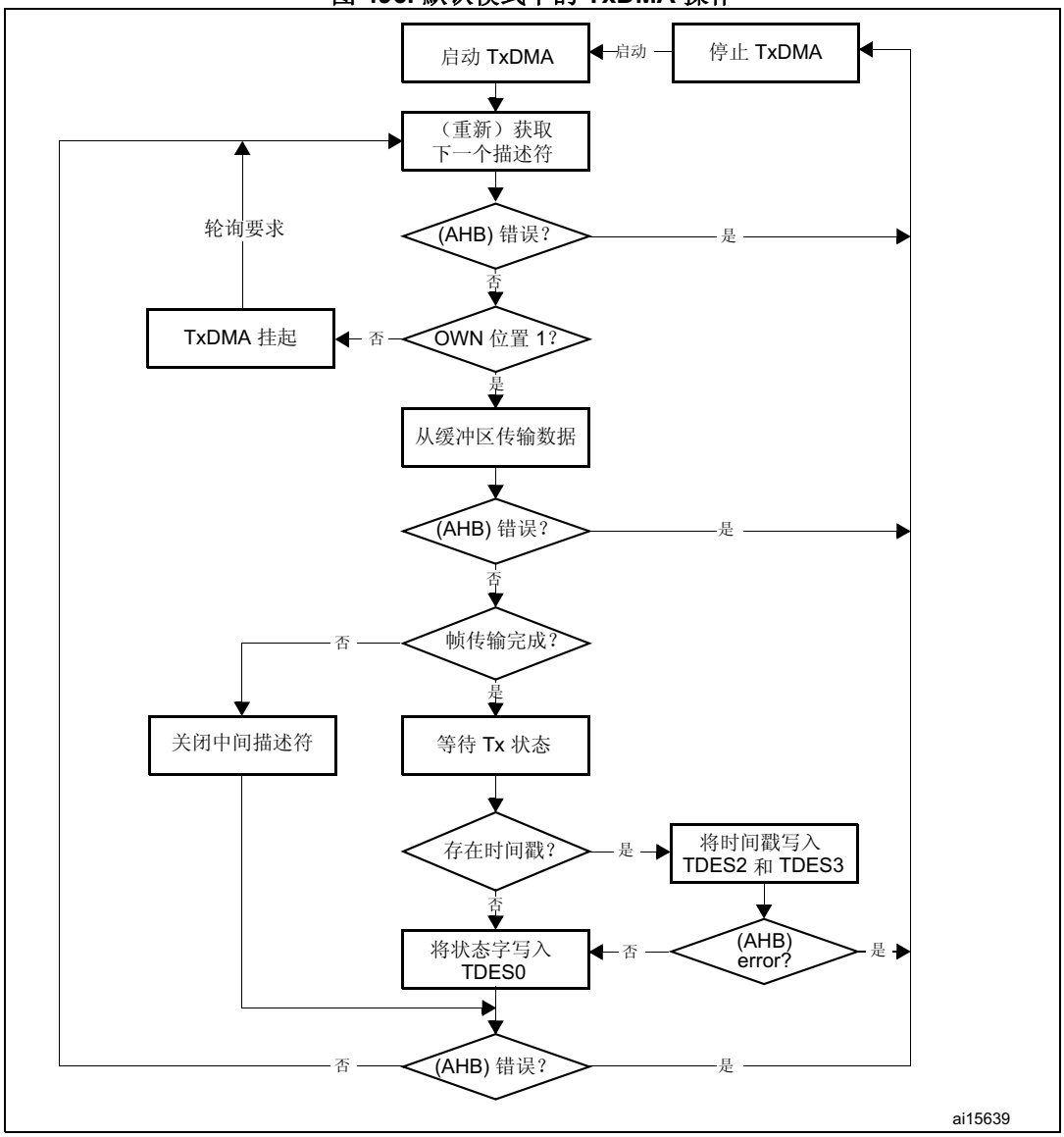
TxDMA 操作：默认（非 OSF）模式

发送 DMA 引擎在默认模式下的操作顺序如下：

1. 在为数据缓冲区设置好相应的以太网帧数据后，用户配置发送描述符 (TDES0-TDES3) 并将 OWN 位 (TDES0[31]) 置 1。
2. ST 位（ETH_DMAOMR 寄存器 [13]）置 1 后，DMA 会立即进入运行状态。
3. DMA 处于运行状态时，会轮询发送描述符列表以获得需要传送的帧的数据。轮询会持续进行直至数据发送完成（无论描述符列表采用环形结构或者链表结构都是一样的）。如果 DMA 在轮询描述符列表时检测到描述符持有者为 CPU，或者发生其他错误，则会挂起传送，并置位发送缓冲区不可用（ETH_DMASR 寄存器 [2]）和正常中断汇总（ETH_DMASR 寄存器 [16]）。发送引擎将执行步骤 9。
4. 如果获得的描述符标记为 DMA 所有（TDES0[31] 置 1），则 DMA 将根据取得的描述符对发送数据缓冲区地址进行解码。
5. DMA 从 STM32F75xxx 和 STM32F74xxx 存储器中获取发送数据并传送这些数据。
6. 如果以太网帧保存在多个描述符的数据缓冲区中，则 DMA 将关闭中间描述符并获取下一个描述符。重复执行步骤 3、4 和 5，直到完成以太网帧数据的传送。
7. 完成帧传送后，如果已为帧使能 IEEE 1588 时间戳（按发送状态中的指示），则时间戳值将写入包含帧结束缓冲区的发送描述符（TDES2 和 TDES3） 随后，状态信息将写入该发送描述符 (TDES0)。由于此步骤中会清零 OWN 位，因此该描述符现在由 CPU 所有。如果没有为该帧使能时间戳，DMA 不会更改 TDES2 和 TDES3 的内容。
8. 发送完在其最后一个描述符中将完成时中断 (TDES1[31]) 置 1 的帧后，发送中断（ETH_DMASR 寄存器 [0]）将置 1。随后，DMA 引擎返回步骤 3。
9. 在挂起状态下，DMA 会在接收到发送轮询要求时尝试重新获取描述符（因此返回步骤 3），并将下溢中断状态位清零。

图 496 给出了默认模式下 TxDMA 发送过程的流程图。

图 496. 默认模式下的 TxDMA 操作



TxDMA 操作：OSF 模式

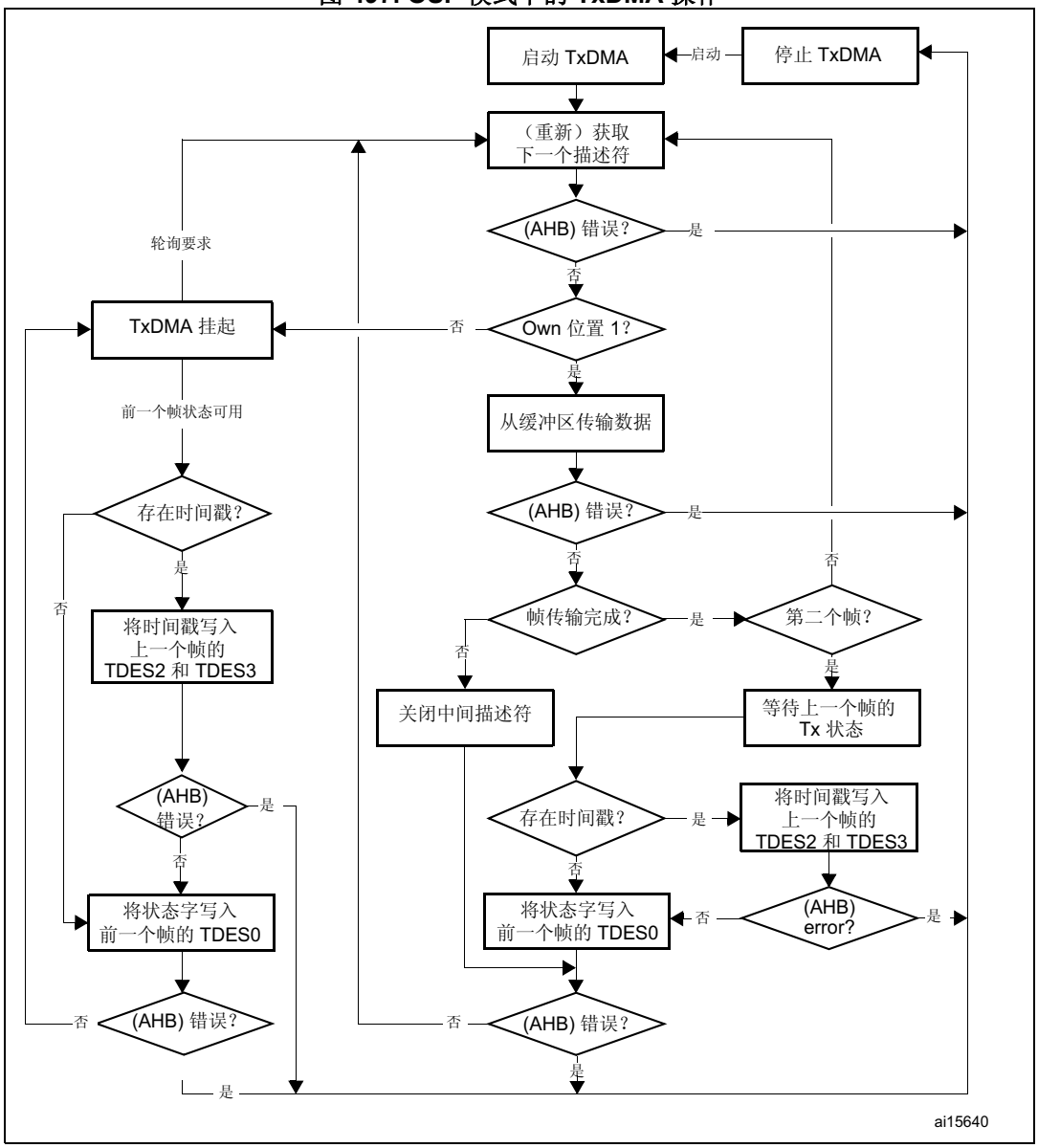
在运行状态下，发送过程无需关闭第一个帧的状态描述符便可同时获取两个帧（如果 ETH_DMAOMR 寄存器 [2] 中的 OSF 位置 1）。发送过程完成第一个帧的传送后，会立即轮询发第二个帧的发送描述符列表。如果第二个帧有效，发送过程会在写入第一个帧的状态信息前发送此帧。在 OSF 模式下，运行状态发送 DMA 按照以下顺序进行操作：

1. DMA 的操作过程如默认模式下 TxDMA 的步骤 1–6 中所述。
2. 无需关闭前一帧的最后一个描述符，DMA 便可获取下一个描述符。
3. 如果 DMA 拥有所需描述符，便会对该描述符中的发送缓冲区地址进行解码。如果 DMA 未拥有描述符，则会进入挂起模式并跳到步骤 7。
4. DMA 从 STM32F75xxx 和 STM32F74xxx 存储器中取得发送帧并发送该帧，直到帧数据发送结束，如果该帧拆分到多个描述符中，则会同时关闭中间描述符。

5. DMA 等待前一个帧的发送状态和时间戳。当状态可用时，如果捕获到时间戳（由状态位指示），DMA 会将这些时间戳写入 TDES2 和 TDES3。之后，OWN 位清零，DMA 将状态写入相应的 TDES0，进而关闭描述符。如果没有为前一个帧使能时间戳，DMA 不会更改 TDES2 和 TDES3 的内容。
6. 如果已使能时间戳，发送中断将置 1，DMA 获取下一个描述符，然后继续执行步骤 3（状态正常时）。如果上一个发送状态显示下溢错误，则 DMA 会进入挂起模式（步骤 7）。
7. 在挂起模式下，如果 DMA 接收到挂起状态和时间戳，则会将时间戳（如果已为当前帧使能时间戳）写入 TDES2 和 TDES3，随后将状态写入相应的 TDES0。之后，将相关中断置 1 并返回挂起模式。
8. 只有在接收到发送轮询要求（ETH_DMATPDR 寄存器）后，DMA 才会退出挂起模式并进入运行状态（转到步骤 1 或步骤 2，具体取决于挂起状态）。

图 497 显示了 OSF 模式下的基本流程图。

图 497. OSF 模式下的 TxDMA 操作



发送帧处理

发送 DMA 期望数据缓冲区包含完整的以太网帧，不包括报头、填充字节和 FCS 字段。DA、SA 和类型/长度字段包含有效数据。如果发送描述符指示 MAC 内核必须禁止插入 CRC 或填充字节，则缓冲区必须具有包括 CRC 字节的完整以太网帧（不包括报头）。帧可以采用数据链接形式，也可以跨越多个缓冲区。帧必须由第一个描述符 (TDES0[28]) 和最后一个描述符 (TDES0[29]) 定界。传送启动时，必须将第一个描述符的 TDES0[28] 置 1。之后，帧数据便从存储器缓冲区传送到发送 FIFO。与此同时，如果当前帧的最后一个描述符 (TDES0[29]) 清零，发送过程将尝试获取下一个描述符。发送过程期望 TDES0[28] 在此描述符中清零。如果 TDES0[29] 清零，则指示中间缓冲区。如果 TDES0[29] 置 1，则指示帧的最后一个缓冲区。当帧的最后一个缓冲区完成传送后，DMA 会将最终状态信息写回描述符的发送描述符 0 (TDES0) 字，该描述符在发送描述符 0 (TDES0[29]) 中将末段置 1。此时，如果完成时中断 (TDES0[30]) 置 1，则发送中断（ETH_DMASR 寄存器 [0] 中）将置 1，并获取下一个描述符，然后重复执行该过程。实际的帧传送过程在发送 FIFO 达到可编程的发送阈值时（ETH_DMAOMR 寄存器 [16:14]），或者 FIFO 中包含完整的帧时才会启动。此外，还可以选择存储转发模式（ETH_DMAOMR 寄存器 [21]）。DMA 完成帧的传送后会释放描述符（清零 OWN 位 TDES0[31]）。

发送轮询挂起

可通过以下任意条件暂停发送轮询：

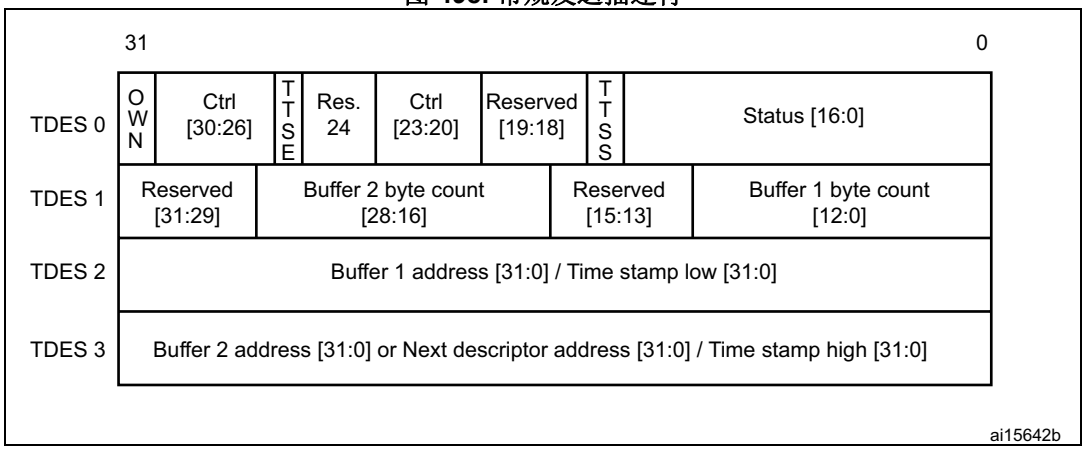
- DMA 检测到 CPU 所有的描述符 (TDES0[31]=0)，并且发送缓冲区不可用标志置 1（ETH_DMASR 寄存器 [2])。如要恢复，驱动程序必须将描述符的所有权交给 DMA，然后发出轮询查询要求命令。
- 检测到由下溢导致的传送错误时，将中止帧的传送。相应的发送描述符 0 (TDES0) 位将置 1。如果发生第二个条件，异常中断汇总位（ETH_DMASR 寄存器 [15] 中）与发送下溢位（ETH_DMASR 寄存器 [5] 中）都将置 1，并且信息将写入发送描述符 0，从而导致挂起。如果 DMA 由于第一个条件而进入挂起状态，则正常中断汇总位（ETH_DMASR 寄存器 [16]）与发送缓冲区不可用位（ETH_DMASR 寄存器 [2]）均置 1。两种情况下，发送列表中的位置都会保留。保留的位置是 DMA 关闭的最后一个描述符后面的描述符位置。纠正挂起原因后，驱动程序必须明确发出发送轮询要求命令。

常规 Tx DMA 描述符

常规发送描述符结构由四个 32 位字组成，如*图 498* 所示。下文给出了 TDES0、TDES1、TDES2 和 TDES3 的位说明。

请注意，如果时间戳已激活（ETH_PTPTSCR 位 0，TSE=1）或 IPv4 校验和减荷已激活（ETH_MACCR 位 10，IPCO=1），则必须使用增强的描述符。

图 498. 常规发送描述符



- **TDES0：发送描述符字 0**

  应用软件必须在描述符初始化期间编程控制位 [30:26]+[23:20] 以及 OWN 位 [31]。当 DMA 更新描述符（或将其写回）时，将复位所有控制位与 OWN 位，并且只报告状态信息。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWN	IC	LS	FS	DC	DP	TTSE	Res	CIC		TER	TCH	Res.		TTSS	IHE
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
ES	JT	FF	IPE	LCA	NC	LCO	EC	VF	CC				ED	UF	DB
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **OWN**：Own 位 (Own bit)

该位置 1 时，表示描述符为 DMA 所有。该位复位时，表示描述符为 CPU 所有。DMA 可在完成帧发送时或在描述符分配的缓冲区全部读取完成后将该位清零。将属于同一个帧的所有后续描述符置 1 后，必须将帧的第一个描述符的所有位置 1。

位 30 **IC**：完成后中断 (Interrupt on completion)

该位置 1 时，当前帧发送完毕后，发送中断位（寄存器 5[0]）置 1。

位 29 **LS**：末段 (Last segment)

该位置 1 时，指示缓冲区中包含帧的末段。

位 28 **FS**：首段 (First segment)

该位置 1 时，指示缓冲区中包含帧的首段。

位 27 **DC**：禁止 CRC (Disable CRC)

该位置 1 时，MAC 不会将循环冗余校验 (CRC) 附加到所发送帧的末尾。该位仅在首段位 (TDES0[28]) 置 1 的情况下有效。

位 26 **DP**：禁止 pad (Disable pad)

该位置 1 时，MAC 不会自动为不足 64 字节的帧添加补位项。该位复位时，DMA 会自动为不足 64 字节的帧添加补位项和 CRC，是否添加 CRC 字段与 DC (TDES0[27]) 位状态无关。该位仅在首段位 (TDES0[28]) 置 1 的情况下有效。

位 25 **TTSE**：发送时间戳使能 (Transmit time stamp enable)

TTSE 位置 1 且 TSE 置 1（ETH_PTPTSCR 位为 0）时，将针对描述符所描述的发送帧激活 IEEE1588 硬件时间戳功能。该字段仅在首段控制位 (TDES0[28]) 置 1 的情况下有效。

位 24 保留，必须保持复位值。

位 23:22 **CIC**：校验和插入控制 (Checksum insertion control)

这些位控制校验和的计算与插入。位编码如下所示：

00：禁止插入校验和

01：仅使能 IP 报头校验和的计算与插入

10：使能 IP 报头校验和以及有效负载校验和的计算与插入，但不会在硬件中计算伪报头校验和

11：使能 IP 报头校验和以及有效负载校验和的计算与插入，并在硬件中计算伪报头校验和。

位 21 **TER**：环发送结束 (Transmit end of ring)

该位置 1 时，表示描述符列表已到达其最后一个描述符。DMA 会返回描述符列表的基址，并形成一个描述符环。

位 20 **TCH**：链接的第二个地址 (Second address chained)

该位置 1 时，表示描述符中的第二个地址是下一个描述符地址，而非第二个缓冲区地址。TDES0[20] 置 1 时，TBS2 (TDES1[28:16]) 为“无关”值。TDES0[21] 优先级高于 TDES0[20]。

位 19:18 保留，必须保持复位值。

位 17 **TTSS**：发送时间戳状态 (Transmit time stamp status)

此字段用作状态位，指示已为所描述的发送帧捕获到时间戳。该位置 1 时，TDES2 和 TDES3 将保存为发送帧捕获到的时间戳值。该字段仅在描述符末段控制位 (TDES0[29]) 置 1 的情况下有效。

注意，使能增强的描述符时（ETH_DMABMR 中的 EDFE=1），TTSS=1 指示 TDES6 和 TDES7 中存有时间戳值。

位 16 **IHE**：IP 报头错误 (IP header error)

该位置 1 时，指示 MAC 发送器在 IP 数据报头中检测到错误。发送器将对照从应用程序接收的报头字节数检查 IPv4 数据包的报头长度，如果出现不匹配，则指示错误状态。对于 IPv6 帧，如果主报头长度不是 40 个字节，则报告一个报头错误。此外，IPv4 或 IPv6 帧的以太网长度/类型字段值必须与随数据包接收的 IP 报头版本匹配。对于 IPv4 帧，如果报头长度字段的值小于 0x5，也会指示一个错误状态。

位 15 **ES**：错误汇总 (Error summary)

指示以下位的逻辑或运算结果：

TDES0[14]：Jabber 超时 (Jabber timeout)

TDES0[13]：帧刷新 (Frame flush)

TDES0[11]：载波丢失 (Loss of carrier)

TDES0[10]：无载波 (No carrier)

TDES0[9]：延迟冲突 (Late collision)

TDES0[8]：过度冲突 (Excessive collision)

TDES0[2]：过度延迟 (Excessive deferral)

TDES0[1]：下溢错误 (Underflow error)

TDES0[16]：IP 报头错误 (IP header error)

TDES0[12]：IP 有效负载错误 (IP payload error)

位 14 **JT**：Jabber 超时 (Jabber timeout)

该位置 1 时，指示 MAC 发送器经历了 jabber 超时。只有在 MAC 配置寄存器的 JD 位未置 1 时，该位才会置 1。

位 13 **FF**：帧已刷新 (Frame flushed)

该位置 1 时，指示 DMA/MTL 已依照 CPU 发出的软件刷新命令刷新帧。

位 12 **IPE**：IP 有效负载错误 (IP payload error)

该位置 1 时，指示 MAC 发送器在 TCP、UDP 或 ICMP IP 数据报有效负载中检测到错误。发送器将对照从应用程序接收的 TCP、UDP 或 ICMP IP 数据包实际字节数检查 IPv4 或 IPv6 报头中接收的有效负载长度，如果出现不匹配，则指示错误状态。

位 11 **LCA**：载波丢失 (Loss of carrier)

该位置 1 时，指示帧发送期间丢失载波（即，帧发送期间有一个或多个发送时钟周期的 MII_CRS 信号无效）。该位仅对在 MAC 处于半双工模式时实现无冲突发送的帧有效。

位 10 **NC**：无载波 (No carrier)

该位置 1 时，指示在发送期间未由 PHY 触发载波侦听信号。

位 9 **LCO**：延迟冲突 (Late collision)

该位置 1 时，指示帧发送过程因冲突窗口（MII 模式下为 64 个字节时间，含报头）后出现冲突而中止。如果下溢错误位置 1，则该位无效。

位 8 **EC**：过度冲突 (Excessive collision)

该位置 1 时，指示尝试发送当前帧时因出现 16 个连续冲突而中止发送。如果 MAC 配置寄存器中的 RD（禁止重试）位置 1，则此位会在出现首个冲突后置 1 并且帧发送过程将中止。

位 7 **VF**：VLAN 帧 (VLAN frame)

该位置 1 时，指示所发送帧为 VLAN 类型的帧。

位 6:3 **CC**：冲突计数 (Collision count)

此 4 位计数器值指示发送帧之前出现的冲突个数。过度冲突位 (TDES0[8]) 置 1 时，该计数无效。

位 2 **ED**：过度延迟 (Excessive deferral)

该位置 1 时，如果 MAC 控制寄存器中的延迟检查 (DC) 位为高电平，则此位指示因出现超过 24288 个位时间的过度延迟而中止发送。

位 1 **UF**：下溢错误 (Underflow error)

该位置 1 时，指示 MAC 因 RAM 存储器的数据未及时到达而中止了帧发送。下溢错误表示 DMA 在帧发送期间遇到发送缓冲区为空的情况。发送过程将进入挂起状态并将发送下溢（寄存器 5[5]）和发送中断（寄存器 5[0]）位置 1。

位 0 **DB**：延迟位 (Deferred bit)

该位置 1 时，指示 MAC 在发送前因存在载波而延迟。该位仅在半双工模式下有效。

- **TDES1：发送描述符字 1**

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
Reserved			TBS2													Reserved			TBS1												
			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

31:29 保留，必须保持复位值。

28:16 **TBS2**：发送缓冲区 2 大小 (Transmit buffer 2 size)

这些位以字节为单位指示第二个数据缓冲区的大小。如果 TDES0[20] 位置 1，则该字段无效。

15:13 保留，必须保持复位值。

12:0 **TBS1**：发送缓冲区 1 大小 (Transmit buffer 1 size)

这些位以字节为单位指示第一个数据缓冲区的大小。如果该字段为 0，DMA 将忽略该缓冲区并使用缓冲区 2 或下一个描述符，具体取决于 TCH (TDES0[20]) 的值。

- **TDES2：发送描述符字 2**

  TDES2 包含描述符第一个缓冲区的地址指针，或者包含时间戳数据。

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
TBAP1/TBAP/TTSL																															
rw																															

位 31:0 **TBAP1**：发送缓冲区 1 地址指针 / 发送帧时间戳低位 (Transmit buffer 1 address pointer / Transmit frame time stamp low)

这些位有两种不同的功能：这些位向 DMA 指示数据在存储器中的位置，而当所有数据传输完毕后，DMA 可利用这些位传回时间戳数据。

TBAP：当软件为 DMA 提供此描述符（TDES0 中的 OWN 位置 1）时，这些位将指示缓冲区 1 的物理地址。缓冲区地址对齐方式不限。有关缓冲区地址对齐方式的详细信息，请参见*第 1387 页上的主机数据缓冲区对齐*。

TTSL：在将 TDES0 中的 OWN 位清零前，DMA 将使用为相应发送帧所捕获的时间戳的 32 个最低有效位更新该字段（覆盖 TBAP1 的值）。只有在激活该帧的时间戳功能（请参见 TDES0 的位 25 TTSE）并且描述符中的末段控制位 (LS) 置 1 时，该字段才包含时间戳。

- **TDES3：发送描述符字 3**

  TDES3 包含描述符中第二个缓冲区或下一个描述符的地址指针，或者包含时间戳数据。

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
TBAP2/TBAP2/TTSH																															
rw																															

位 31:0 **TBAP2**：发送缓冲区 2 地址指针（下一个描述符地址）/ 发送帧时间戳高位 (Transmit buffer 2 address pointer (Next descriptor address) / Transmit frame time stamp high)

这些位有两种不同的功能：这些位向 DMA 指示数据在存储器中的位置，而当所有数据传输完毕后，DMA 可利用这些位传回时间戳数据。

TBAP2：当软件为 DMA 提供此描述符（TDES0 中的 OWN 位置 1）并且使用描述符环结构时，这些位将指示缓冲区 2 的物理地址。如果链接的第二个地址位 (TDES1 [20]) 置 1，则该地址包含下一个描述符所在物理寄存器的指针。只有在 TDES1 [20] 位置 1 时，缓冲区地址指针才必须与总线宽度相符。（将在内部忽略 LSB。）

TTSH：在将 TDES0 中的 OWN 位清零前，DMA 将使用为相应发送帧所捕获的时间戳的 32 个最高有效位更新该字段（覆盖 TBAP2 的值）。只有在激活该帧的时间戳功能（请参见 TDES0 的位 25 TTSE）并且描述符中的末段控制位 (LS) 置 1 时，该字段才包含时间戳。

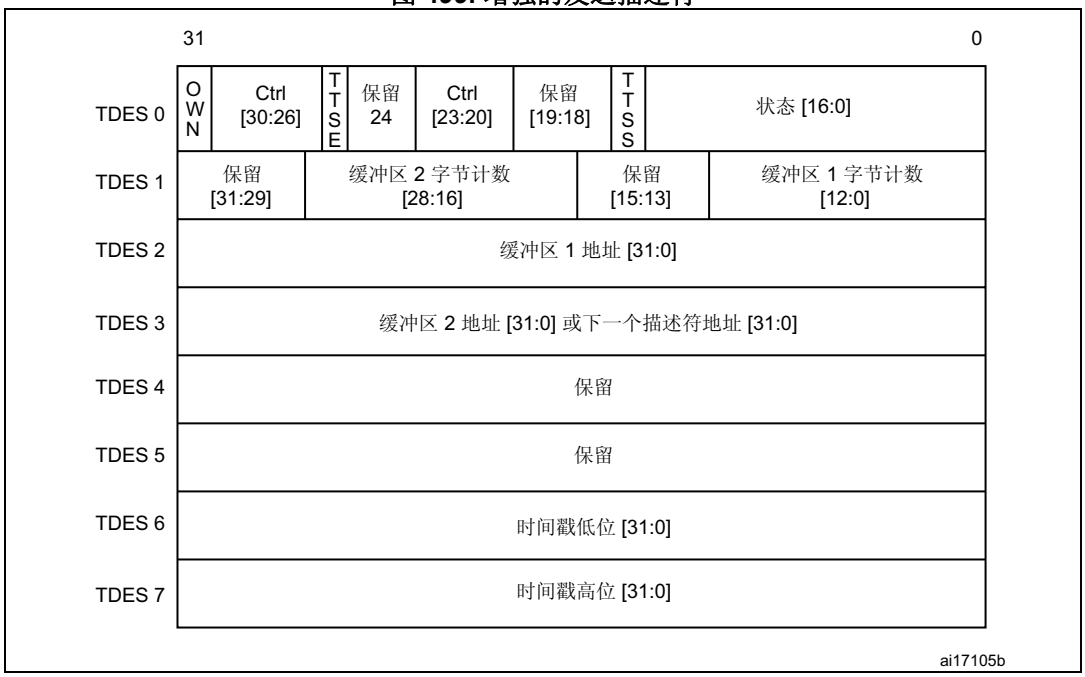
增强 Tx DMA 描述符

如果时间戳已激活（TSE=1，ETH_PTPTSCR 位 0）或 IPv4 校验和减荷已激活（IPCO=1，ETH_MACCR 位 10），则必须使用增强的描述符（通过 ETHDMABMR 位 7 EDFE=1 使能）。

增强的描述符由 8 个 32 位字组成，即描述符正常大小的两倍。TDES0、TDES1、TDES2 和 TDES3 的定义与常规发送描述符相同（请参见 *常规 Tx DMA 描述符*）。TDES6 和 TDES7 包含时间戳。TDES4、TDES5、TDES6 和 TDES7 的定义如下。

选择增强的描述符模式时，需要利用软件为每个描述符分配 32 个字节（8 个双字）的内存。如果未使用时间戳或 IPv4 校验和减荷，则可以禁止增强的描述符格式，而软件可使用默认大小为 16 字节的常规描述符。

图 499. 增强的发送描述符



- **TDES4：发送描述符字 4**

  保留

- **TDES5：发送描述符字 5**

  保留

- **TDES6：发送描述符字 6**

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
TTSL																															
rw																															

位 31:0 **TTSL**：发送帧时间戳低位 (Transmit frame time stamp low)

DMA 使用为相应发送帧所捕获的时间戳的 32 个最低有效位更新该字段。只有在描述符的末段控制位 (LS) 置 1 时，该字段才包含时间戳。

- **TDES7：发送描述符字 7**

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
TTSH																															
rw																															

位 31:0 **TTSH**：发送帧时间戳高位 (Transmit frame time stamp high)

DMA 使用为相应发送帧所捕获的时间戳的 32 个最高有效位更新该字段。只有在描述符的末段控制位 (LS) 置 1 时，该字段才包含时间戳。

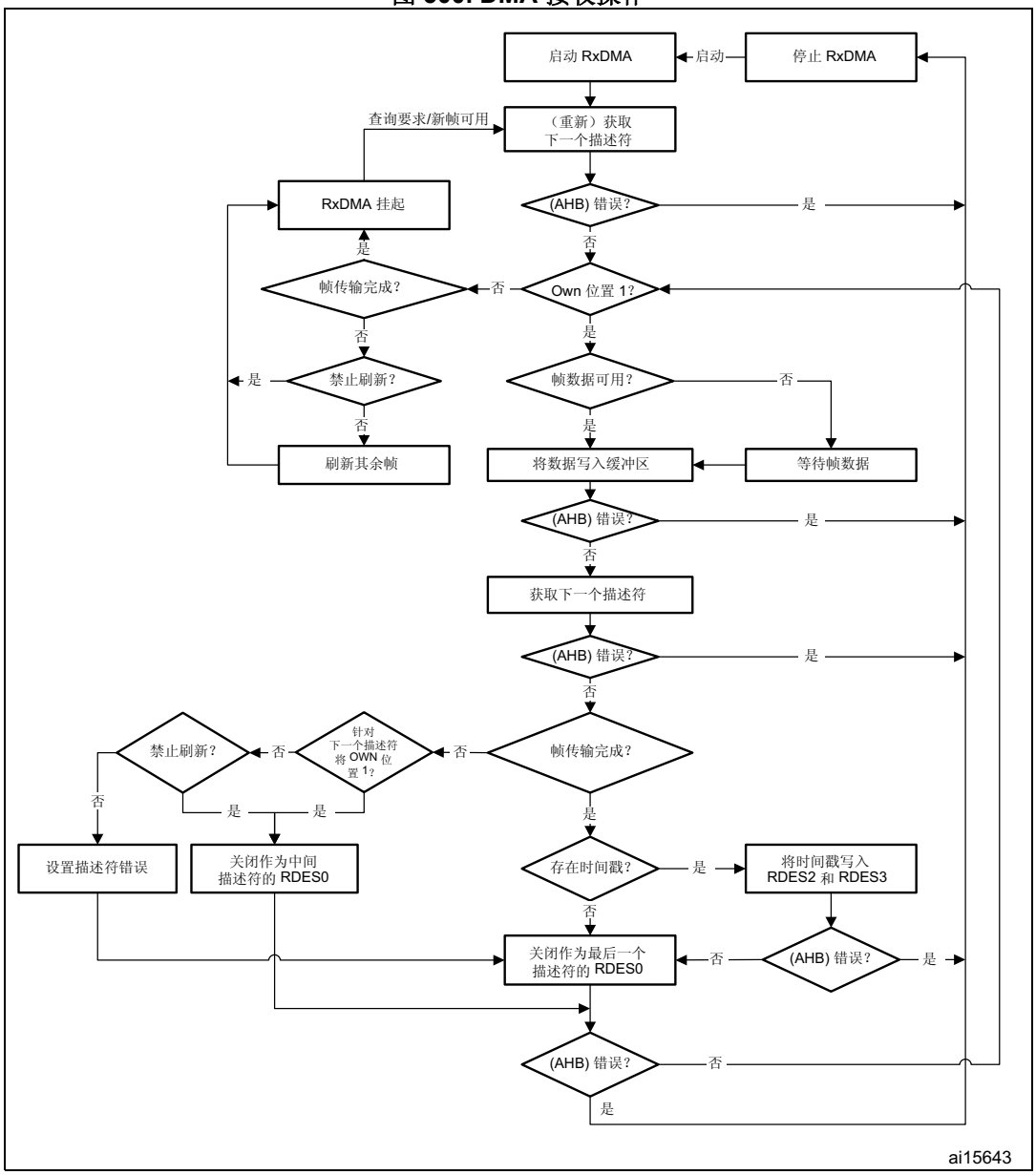
38.6.8 Rx DMA 配置

图 500 所示为接收 DMA 引擎的接收序列，具体说明如下：

1. CPU 设置接收描述符 (RDES0-RDES3) 并将 OWN 位 (RDES0[31]) 置 1。
2. SR 位（ETH_DMAOMR 寄存器 [1]）置 1 后，DMA 即进入运行状态。在运行状态下，DMA 轮询接收描述符列表，尝试获取空闲描述符。如果获得的描述符不空闲（由 CPU 所拥有），则 DMA 进入挂起状态并跳转到第 9 步。
3. DMA 将所获取描述符中的接收数据缓冲区地址解码。
4. 处理传入帧并将其放入所获取描述符的数据缓冲区。
5. 当缓冲区已满或帧传输完成时，接收引擎将获取下一个描述符。
6. 如果当前的帧传输完成，DMA 将继续执行第 7 步。如果 DMA 未拥有下一个获取的描述符且帧传输尚未完成（尚未传输 EOF），则 DMA 会将 RDES0 中的描述符错误位置 1（除非禁止刷新）。DMA 关闭当前描述符（将 OWN 位清零）并通过在 RDES1 值中将末段位 (LS) 清零来将其标记为中间描述符（如果禁止刷新，则将其标记为最后一个描述符），随后继续执行第 8 步。如果 DMA 已拥有下一个描述符，但当前的帧传输尚未完成，则 DMA 将关闭当前描述符作为中间值并返回第 4 步。
7. 如果已使能 IEEE 1588 时间戳功能，DMA 会将时间戳（如果可用）写入当前描述符的 RDES2 和 RDES3。DMA 随后获取所接收帧的状态并将该状态字写入当前描述符的 RDES0，同时 OWN 位清零且末段位置 1。
8. 接收引擎检查最新描述符的 OWN 位。如果 CPU 拥有该描述符（OWN 位为 0），则接收缓冲区不可用位（ETH_DMASR 寄存器 [7]）置 1 且 DMA 接收引擎进入挂起状态（第 9 步）。如果 DMA 拥有该描述符，引擎将返回第 4 步并等待下一个帧。
9. 在接收引擎进入挂起状态前，将从接收 FIFO 中刷新部分帧（可使用 ETH_DMAOMR 寄存器的位 24 控制刷新）。
10. 当收到接收轮询要求命令或者可以从接收 FIFO 获得下一个帧的起点时，接收 DMA 将退出挂起状态。引擎继续执行第 2 步并重新获取下一个描述符。

直到完成时间戳回写并准备将状态回写到描述符时，DMA 才会确认接收状态。如果软件已通过 CSR 使能时间戳功能，则当无法获得帧的有效时间戳值时（例如，由于在写入时间戳前接收 FIFO 已满），DMA 会向 RDES2 和 RDES3 的所有位写入 1。否则（即，未使能时间戳功能），RDES2 和 RDES3 保持不变。

图 500. DMA 接收操作



接收描述符获取

接收引擎始终会尝试为即将传入的帧获取一个额外的描述符。只要满足以下任一条件，即尝试获取描述符：

- DMA 进入运行状态后，接收启动/停止位（ETH_DMAOMR 寄存器 [1]）立即置 1。
- 在当前传输的帧结束前，当前描述符的数据缓冲区已满。
- 控制器已完成数据帧接收，但当前的接收描述符尚未关闭。
- 接收过程因缓冲区由 CPU 所拥有 (RDES0[31] = 0) 而挂起，并且接收到新帧。
- 已发出接收轮询要求命令。

接收帧处理

只有当帧通过地址过滤且其大小大于或等于为接收 FIFO 所设置的可配置阈值字节数时，或者在存储转发模式下将整个帧写入 FIFO 时，MAC 才会将接收的帧传输到 STM32F75xxx 和 STM32F74xxx 存储器。如果帧未通过地址过滤，将自行进入 MAC 块中（除非接收所有位 ETH_MACFFR [31] 置 1）。不足 64 字节的帧由于会发生冲突或过早结束，因而可从接收 FIFO 中清除。接收 64（可配置阈值）个字节后，DMA 块开始将帧数据传输到当前描述符所指定的接收缓冲区中。DMA AHB 接口准备好接收数据传输后，如果 DMA 当前未从存储器获取发送数据，则将第一个描述符位 (RDES0[9]) 置 1，以分隔该帧。数据缓冲区已满或者帧的末段已传输到接收缓冲区时，OWN (RDES0[31]) 位将复位为 0，此时将释放描述符。如果帧包含在单个描述符中，则最后一个描述符位 (RDES0[8]) 和第一个描述符位 (RDES0[9]) 均置 1。DMA 获取下一个描述符，将最后一个描述符位 (RDES0[8]) 置 1，并释放前一个帧描述符中的 RDES0 状态位。然后，DMA 将接收中断位（ETH_DMASR 寄存器 [6]）置 1。这一过程将重复执行，直至 DMA 遇到被标记为由 CPU 所有的描述符。如果出现这种情况，接收过程将接收缓冲区不可用位（ETH_DMASR 寄存器 [7]）置 1，随后进入挂起状态。但其在接收列表中的位置仍然保留。

接收过程挂起

如果在接收过程处于挂起状态时有新的接收帧到达，则 DMA 将重新获取 STM32F75xxx 和 STM32F74xxx 存储器中的当前描述符。如果该描述符现在由 DMA 所拥有，接收过程将重新进入运行状态并开始接收帧。如果该描述符仍由主机所拥有，则默认情况下，DMA 将丢弃 Rx FIFO 顶部的当前帧并将丢失帧计数器递增。如果 Rx FIFO 中存储了多个帧，将重复执行该过程。将 DMA 工作模式寄存器的位 24 (DFRF) 置 1 后，可避免丢弃或刷新 Rx FIFO 顶部的帧。在这种情况下，接收过程将接收缓冲区不可用状态位置 1 并返回到挂起状态。

常规 Rx DMA 描述符

常规接收描述符结构由四个 32 位字（16 字节）组成， 如*图 501* 所示。下文将介绍 RDES0、RDES1、RDES2 和 RDES3 的各个位。

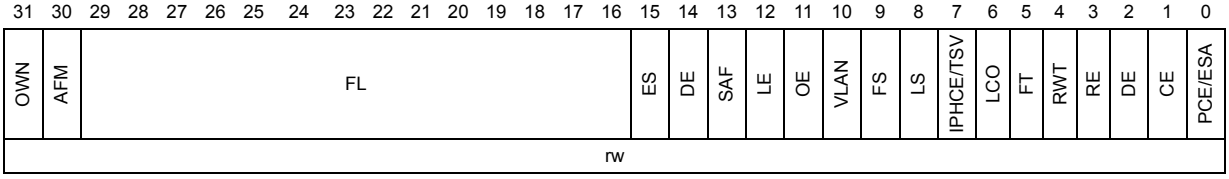
请注意，如果已激活时间戳功能（ETH_PTPTSCR 位 0 TSE=1）或 IPv4 校验和减荷（ETH_MACCR 位 10 IPCO=1），则必须使用增强的描述符。

图 501. 常规 Rx DMA 描述符结构



- **RDES0：接收描述符字 0**

  REDS0 包含接收帧状态、帧长度和描述符所有关系信息。



位 31 **OWN**：所有关系位 (Own bit)

该位置 1 时，指示描述符由 MAC 子系统的 DMA 所拥有。该位复位时，指示描述符由主机所拥有。DMA 在帧接收完成或此描述符的关联缓冲区已满时将该位清零。

位 30 **AFM**：目标地址过滤失败 (Destination address filter fail)

该位置 1 时，指示帧未能通过 MAC 内核中的 DA 过滤。

位 29:16 **FL**：帧长度 (Frame length)

这些位指示传输到主机存储器中的接收帧字节长度（含 CRC）。只有在最后一个描述符位 (RDES0[8]) 置 1 且描述符错误位 (RDES0[14]) 复位时，该字段才有效。

在最后一个描述符位 (RDES0[8]) 置 1 时，该字段有效。当最后一个描述符位和错误汇总位均未置 1 时，该字段指示已为当前帧传输的累计字节数。

位 15 **ES**：错误汇总 (Error summary)

指示以下位的逻辑或运算结果：

RDES0[1]：CRC 错误 (CRC error)

RDES0[3]：接收错误 (Receive error)

RDES0[4]：看门狗超时 (Watchdog timeout)

RDES0[6]：延迟冲突 (Late collision)

RDES0[7]：巨帧 (Giant frame)（当 RDES0[7] 指示 IPV4 报头校验和错误时，该位不适用。）

RDES0[11]：上溢错误 (Overflow error)

RDES0[14]：描述符错误 (Descriptor error)

只有最后一个描述符 (RDES0[8]) 置 1 时，该字段才有效。

位 14 **DE**：描述符错误 (Descriptor error)

该位置 1 时，指示某个帧因超过当前描述符缓冲区的大小而被截断以及 DMA 未拥有下一个描述符。帧被截断。只有最后一个描述符 (RDES0[8]) 置 1 时，该字段才有效。

位 13 **SAF**：源地址过滤失败 (Source address filter fail)

该位置 1 时，指示帧的 SA 字段未能通过 MAC 内核中的 SA 过滤。

位 12 **LE**：长度错误 (Length error)

该位置 1 时，指示接收帧的实际长度与长度/类型字段的值不符。该字段仅在帧类型位 (RDES0[5]) 复位后有效。

位 11 **OE**：上溢错误 (Overflow error)

该位置 1 时，指示接收帧因缓冲区上溢而损坏。

位 10 **VLAN**：VLAN 标记 (VLAN tag)

该位置 1 时，指示描述符所指向的帧是由 MAC 内核标记的 VLAN 帧。

位 9 **FS**：第一个描述符 (First descriptor)

该位置 1 时，指示此描述符包含帧的第一个缓冲区。如果第一个缓冲区的大小为 0，则第二个缓冲区将包含帧的帧头。如果第二个缓冲区的大小也为 0，则下一个描述符将包含帧的帧头。

位 8 **LS**：最后一个描述符 (Last descriptor)

该位置 1 时，指示此描述符指向的缓冲区为帧的最后一个缓冲区。

位 7 **IPHCE/TSV**：IPv 报头校验和错误/时间戳有效 (IPv header checksum error / time stamp valid)

IPHCE 位置 1 时，指示 IPv4 或 IPv6 报头中存在错误。导致此错误的原因可能是：以太网类型字段与 IP 报头版本字段值不一致，或者与 IPv4 中报头的校验和不匹配，或者以太网帧缺少所需的 IP 报头字节数。如*表 243* 中所指出的那样，该位可具有特殊的含义。

使能增强的描述符格式（EDFE=1，ETH_DMABMR 的位 7）后，该位具有 TSV 功能（否则其为 IPHCE）。TSV 位置 1 时，表示将在描述符字 6 (RDES6) 和 7 (RDES7) 中写入时间戳快照。只有最后一个描述符位 (RDES0[8]) 置 1 时，TSV 才有效。

位 6 **LCO**：延迟冲突 (Late collision)

该位置 1 时，表示在以半双工模式接收帧时发生了延迟冲突。

位 5 **FT**：帧类型 (Frame type)

该位置 1 时，表示接收帧为以太网类型帧（LT 字段大于或等于 0x0600）。该位复位时，表示接收的帧为 IEEE802.3 帧。该位不适用于少于 14 个字节的矮帧。使用正常描述符格式 (ETH_DMABMR EDFE=0) 时，FT 可具有特殊的含义，如*表 243* 中所指出的那样。

位 4 **RWT**：接收看门狗超时 (Receive watchdog timeout)

该位置 1 时，表示接收看门狗计时器在接收当前帧时超时，且当前帧在看门狗超时后被截断了。

位 3 **RE**：接收错误 (Receive error)

该位置 1 时，表示在帧接收期间，当发出 RX_DV 信号时，会发出 RX_ERR 信号。

位 2 **DE**：Dribble 位错误 (Dribble bit error)

该位置 1 时，表示接收的帧具有非整数倍数的字节（奇数半字节）。该位仅在 MII 模式下有效。

位 1 **CE**：CRC 错误 (CRC error)

该位置 1 时，表示接收的帧发生循环冗余校验 (CRC) 错误。只有最后一个描述符 (RDES0[8]) 置 1 时，该字段才有效。

位 0 **PCE/ESA**：有效负载校验和错误/扩展状态可用 (Payload checksum error / extended status available)

该位置 1 时，表示由内核计算的 TCP、UDP 或 ICMP 校验和与接收到的封装 TCP、UDP 或 ICMP 段校验和字段不匹配。当接收到的有效负载字节数与接收到的以太网帧中封装 IPv4 或 IPv6 数据图的长度字段中所指示的值不匹配时，该位也会被置 1。如*表 243* 中所指出的那样，该位可具有特殊的含义。

使能增强的描述符格式（EDFE=1，ETH_DMABMR 的位 7）后，该位具有 ESA 功能（否则其为 PCE）。ESA 置 1 时，表示描述符字 4 (RDES4) 中存在扩展状态。只有最后一个描述符位 (RDES0[8]) 置 1 时，ESA 才有效。

位 5、7 和 0 用于指示*表 243* 中讨论的情况。

表 243. 接收描述符 0 - 位 7、5 和 0 的编码（仅正常描述符格式，EDFE=0）

位 5：帧类型 (frame type)	位 7：IPC 校验和错误 (IPC checksum error)	位 0：有效负载校验和错误 (payload checksum error)	帧状态
0	0	0	IEEE 802.3 类型帧（长度字段的值小于 0x0600。）
1	0	0	IPv4/IPv6 类型帧，未检测到校验和错误
1	0	1	IPv4/IPv6 类型帧，检测到有效负载校验和错误（请参见 PCE 的相关说明）
1	1	0	IPv4/IPv6 类型帧，检测到 IP 报头校验和错误（请参见 IPCE 的相关说明）
1	1	1	IPv4/IPv6 类型帧，同时检测到 IP 报头和有效负载校验和错误
0	0	1	IPv4/IPv6 类型帧，无 IP 报头校验和错误，有效负载检查因有效负载不受支持而被绕过
0	1	1	既不是 IPv4 也不是 IPv6 的类型帧（校验和减荷引擎完全绕过校验和。）
0	1	0	保留

- **RDES1：接收描述符字 1**

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
DIC	RBS2		RBS2													RER	RCH	Reserved	RBS												
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **DIC**：完成时禁止中断 (Disable interrupt on completion)

置 1 时，该位会阻止状态寄存器的 RS 位 (CSR5[6]) 置 1，使得接收的帧在由该描述符所指示的缓冲区中结束， 从而会因该帧的 RS 而禁止引发主机的中断。

位 30:29 保留，必须保持复位值。

位 28:16 **RBS2**：接收缓冲区 2 大小 (Receive buffer 2 size)

这些位以字节为单位指示第二个数据缓冲区的大小。即使 RDES3（缓冲区 2 地址指针）的值未与总线宽度对齐，缓冲区大小也必须为 4、8 或 16 的倍数，具体取决于总线宽度（分别为 32、64 或 128）。如果缓冲区大小不是 4、8 或 16 的倍数，则不会定义产生的行为。如果将 RDES1[14] 置 1，则该字段无效。

位 15 **RER**：接收环结束 (Receive end of ring)

该位置 1 时，表示描述符列表已到达其最后一个描述符。DMA 会返回描述符列表的基址，并形成一个描述符环。

位 14 **RCH**：链接的第二个地址 (Second address chained)

该位置 1 时，表示描述符中的第二个地址是下一个描述符地址，而非第二个缓冲区地址。该位置 1 时，RBS2 (RDES1[28:16]) 为“无关”值。RDES1[15] 优先级高于 RDES1[14]。

位 13 保留，必须保持复位值。

位 12:0 **RBS1**：接收缓冲区 1 大小 (Receive buffer 1 size)

以字节为单位表示第一个数据缓冲区的大小。即使 RDES2（缓冲区 1 地址指针）的值未对齐，缓冲区大小也必须为 4、8 或 16 的倍数，具体取决于总线宽度（32、64 或 128）。如果缓冲区大小不是 4、8 或 16 的倍数，则不会定义产生的行为。如果该字段为 0，则 DMA 会忽略该缓冲区并使用缓冲区 2 或下一个描述符，具体取决于 RCH（位 14）的值。

- **RDES2：接收描述符字 2**

RDES2 包含描述符中第一个数据缓冲区的地址指针，或者包含时间戳数据。

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
RBP1/RTSL																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **RBAP1 / RTSL**：接收缓冲区 1 地址指针 / 接收帧时间戳低位 (Receive buffer 1 address pointer / Receive frame time stamp low)

这些位有两种不同的功能：应用程序使用它们向 DMA 指示数据在存储器中的存储位置，而当所有数据均传输完成后，DMA 可以使用这些位来传回时间戳数据。

RBAP1：当软件为 DMA 提供此描述符时（RDES0 中的 OWN 位置 1 时），这些位会指示缓冲区 1 的物理地址。在缓冲区地址对齐方面没有任何限制，但以下情况除外：当使用 RDES2 值存储帧起点时，DMA 会使用所配置的值生成其地址。请注意，在传输帧开始期间，DMA 将在 RDES2[3/2/1:0] 位为 0 的情况下执行写操作，但帧数据会根据实际的缓冲区地址指针进行移位。当地址指针指向存储有帧中间部分或最后部分的缓冲区时，DMA 将忽略 RDES2[3/2/1:0]（与总线宽度 128/64/32 相对应）。

RTSL：在将 RDES0 中的 OWN 位清零之前，DMA 会使用为对应接收帧捕获的时间戳的 32 个最低有效位更新该字段（覆盖 RBAP1 的值）。只有此帧的时间戳已激活且描述符中的末段控制位 (LS) 已置 1 时，该字段才具有时间戳。

- **RDES3：接收描述符字 3**

RDES3 包含描述符中第二个数据缓冲区或下一个描述符的地址指针，或者包含时间戳数据。

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
RBP2/RTSH																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **RBAP2/RTSH**：接收缓冲区 2 地址指针（下一个描述符地址）/ 接收帧时间戳高电平 (Receive buffer 2 address pointer (next descriptor address) / Receive frame time stamp high)

这些位有两种不同的功能：应用程序使用它们向 DMA 指示数据在存储器中的存储位置，而当所有数据均传输完成后，DMA 可以使用这些位来传回时间戳数据。

RBAP1：当软件为 DMA 提供此描述符时（当 RDES0 中的 OWN 位置 1 时），这些位会指示当使用描述符环结构时缓冲区 2 的物理地址。如果链接的第二个地址 (RDES1 [24]) 位置 1，则该地址包含指向下一个描述符所在物理存储器的指针。如果 RDES1 [24] 置 1，则缓冲区（下一个描述符）地址指针必须是总线宽度对齐形式（RDES3[3、2 或 1:0] = 0，与总线宽度 128、64 或 32 相对应， LSB 被内部忽略。）然而，当 RDES1 [24] 复位时，RDES3 值无任何限制，但以下情况除外：当使用 RDES3 值存储帧起点时，DMA 会使用所配置的值生成其缓冲区地址。当地址指针指向存储有帧中间部分或最后部分的缓冲区时，DMA 将忽略 RDES3[3、2 或 1:0]（与总线宽度 128、64 或 32 相对应）。

RTSH：在将 RDES0 中的 OWN 位清零之前，DMA 会使用为对应接收帧捕获的时间戳的 32 个最高有效位更新该字段（覆盖 RBAP2 的值）。只有时间戳已激活且描述符中的末段控制位 (LS) 已置 1 时，该字段才具有时间戳。

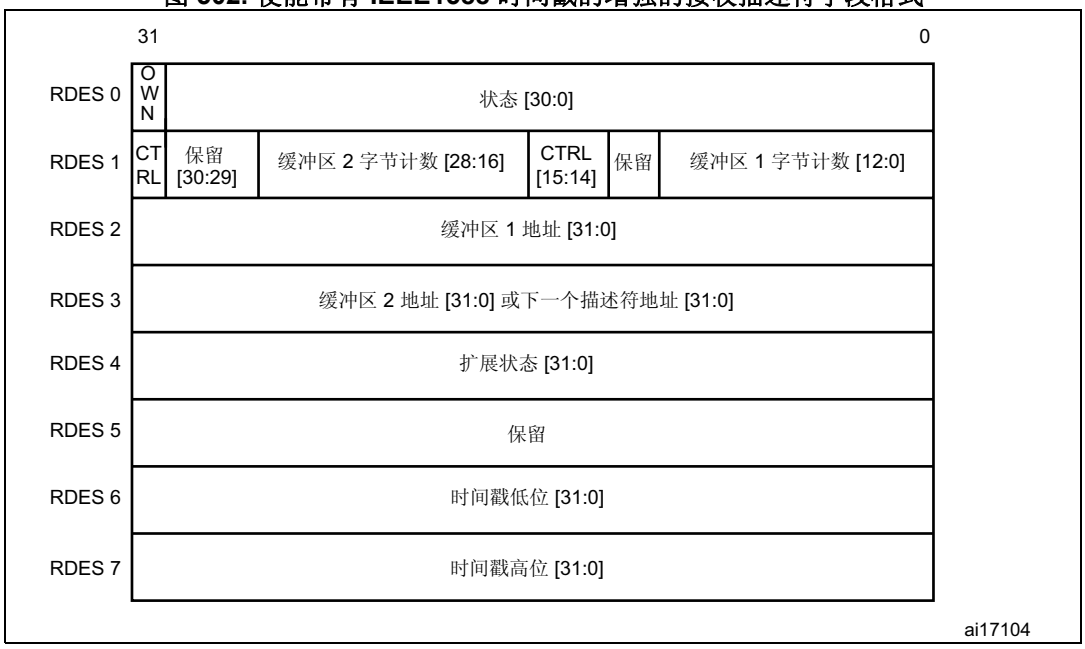
带有 IEEE1588 时间戳的增强的 Rx DMA 描述符

如果时间戳已激活（TSE=1，ETH_PTPTSCR 位 0）或 IPv4 校验和减荷已激活（IPCO=1，ETH_MACCR 位 10），则必须使用增强的描述符（通过 ETHDMABMR 位 7 EDFE=1 使能）。

增强的描述符由 8 个 32 位字组成，即描述符正常大小的两倍。RDES0、RDES1、RDES2 和 RDES3 的定义与常规接收描述符相同（请参见*常规 Rx DMA 描述符*）。RDES4 包含扩展状态，而 RDES6 和 RDES7 包含时间戳。RDES4、RDES5、RDES6 和 RDES7 的定义如下。

选择增强的描述符模式时，需要利用软件为每个描述符分配 32 个字节（8 个双字）内存。如果未使用时间戳或 IPv4 校验和减荷，则可以禁止增强的描述符格式，而软件可使用默认大小为 16 字节的常规描述符。

图 502. 使能带有 IEEE1588 时间戳的增强的接收描述符字段格式



- **RDES4：接收描述符字 4**

仅当存在与 IPv4 校验和或时间戳（由 RDES0 中的位 0 加以指示）相关的状态时，如下所示的扩展状态才有效。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PV	PFT	PMT				IPV6PR	IPV4PR	IPCB	IPPE	IPHE	IPPT		
																		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:14 保留，必须保持复位值。

位 13 **PV**：PTP 版本 (PTP version)

置 1 时，表示接收的 PTP 消息使用 IEEE 1588 的版本 2 格式。清零时，将使用版本 1 格式。仅当消息类型为非零时才有效。

位 12 **PFT**：PFT 帧类型 (PTP frame type)

该位置 1 时，表示 PTP 消息直接通过以太网发送。该位清零且消息类型为非零时，表示 PTP 消息通过 UDP-IPv4 或 UDP-IPv6 发送。可通过位 6 或位 7 获取有关 IPv4 或 IPv6 的信息。

位 11:8 **PMT**：PTP 消息类型 (PTP message type)

将对这些位进行编码，以给出所接收消息的类型。

– 0000：未接收到任何 PTP 消息
– 0001：SYNC（所有时钟类型）
– 0010：Follow_Up（所有时钟类型）
– 0011：Delay_Req（所有时钟类型）
– 0100：Delay_Resp（所有时钟类型）
– 0101：Pdelay_Req（在点对点透明时钟中）或 Announce（在普通时钟或边界时钟中）
– 0110：Pdelay_Resp（在点对点透明时钟中）或 Management（在普通时钟或边界时钟中）
– 0111：Pdelay_Resp_Follow_Up（在点对点透明时钟中）或 Signaling（在普通时钟或边界时钟中）
– 1xxx - 保留

位 7 **IPV6PR**：接收到 IPv6 数据包 (IPv6 packet received)

该位置 1 时，表示接收到的数据包为 IPv6 数据包。

位 6 **IPV4PR**：接收到 IPv4 数据包 (IPv4 packet received)

该位置 1 时，表示接收到的数据包为 IPv4 数据包。

位 5 **IPCB**：绕过 IP 校验和 (IP checksum bypassed)

该位置 1 时，表示绕过校验和减荷引擎。

位 4 **IPPE**：IP 有效负载错误 (IP payload error)

该位置 1 时，表示由内核计算的 16 位 IP 有效负载校验和（即 TCP、UDP 或 ICMP 校验和）与接收段中对应的校验和字段不匹配。当 TCP、UDP 或 ICMP 段长度与 IP 报头字段中的有效负载长度值不匹配时，它同样会置 1。

位 3 **IPHE**：IP 报头错误 (IP header error)

该位置 1 时，表示由内核计算的 16 位 IPv4 报头校验和与接收的校验和字节不匹配，或 IP 数据报版本与以太网类型值不一致。

位 2:0 **IPPT**：IP 有效负载字节 (IP payload type)

如果 IPv4 校验和减荷被激活（IPCO=1，ETH_MACCR 位 10），则这些位表示 IP 数据报中封装的有效负载类型。当 IP 报头出错或存在分段的 IP时，这些位为“00”。

– 000：未知，或未处理 IP 有效负载
– 001：UDP
– 010：TCP
– 011：ICMP
– 1xx：保留

- **RDES5：接收描述符字 5**

  保留。

- **RDES6：接收描述符字 6**

  下表介绍了当接收描述符关闭且使能时间戳时对 RDES6 而言具有不同含义的字段。

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
RTSL																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **RTSL**：接收帧时间戳低位 (Transmit frame time stamp low)

DMA 使用为对应接收帧捕获的时间戳的 32 最低有效位更新该字段。DMA 仅为最后描述符状态位 (RDES0[8]) 指示的接收帧的最后描述符更新该字段。当该字段和 RDES7 中的 RTSH 字段均显示 1 时，必须视为时间戳已损坏。

- **RDES7：接收描述符字 7**

  下表介绍了当接收描述符关闭且使能时间戳时对 RDES7 而言具有不同含义的字段。

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
RTSH																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **RTSH**：接收帧时间戳高位 (Receive frame time stamp high)

DMA 使用为对应接收帧捕获的时间戳的 32 个最高有效位更新该字段。DMA 仅为最后描述符状态位 (RDES0[8]) 指示的接收帧的最后描述符更新该字段。

当该字段和 RDES7 的 RTSL 字段都显示 1 时，必须将时间戳视为已损坏。

38.6.9 DMA 中断

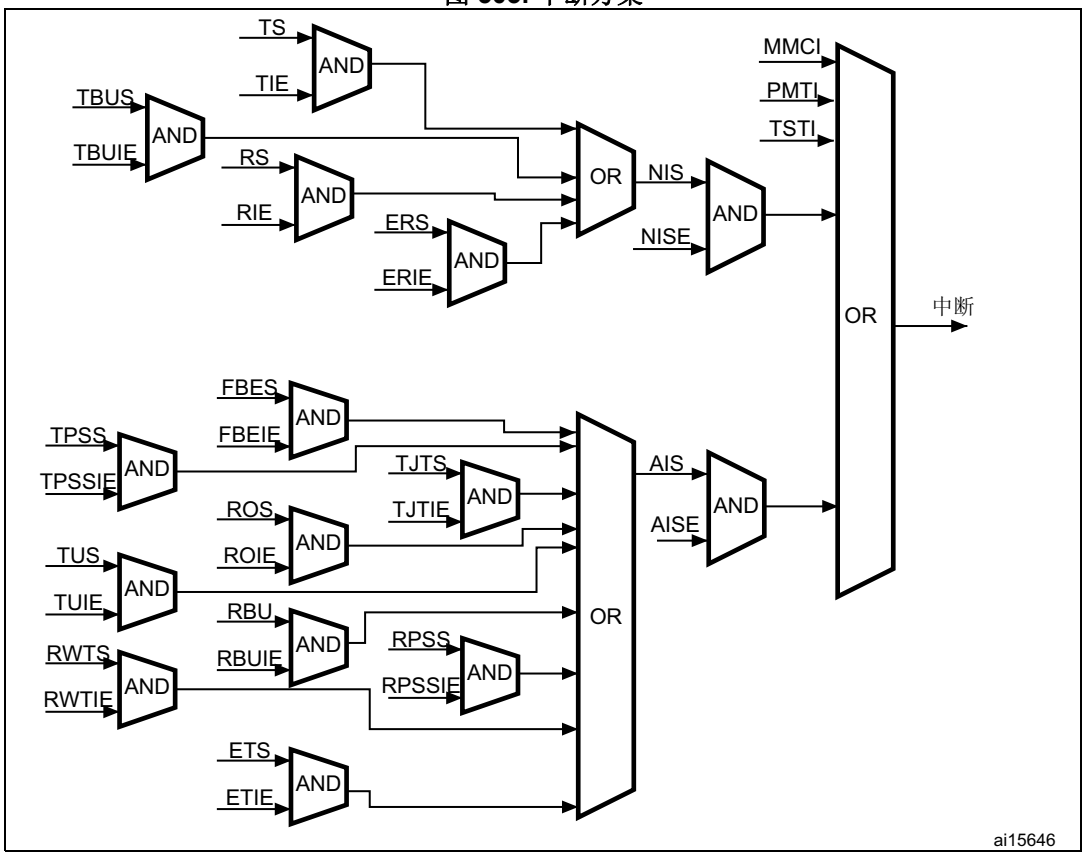
中断可以作为各种事件的结果而生成。ETH_DMASR 寄存器包含可能引起中断的所有位。ETH_DMAIER 寄存器对可能引起中断的每一个事件都包含一个使能位。

如 ETH_DMASR 寄存器中所描述，存在两组中断：正常中断和异常中断。通过向对应位位置写入 1 来清除中断。当清除了组内所有使能的中断时，对应的汇总位亦被清零。如果 MAC 内核是引发中断的原因，则 ETH_DMASR 寄存器中的任何 TSTS 或 PMTS 位都将置为高电平。

中断不会排队积压，如果中断事件在驱动程序对其做出响应之前发生，则不会再生成中断。例如，接收中断位（ETH_DMASR 寄存器 [6]）指示一个或多个帧被传输到 STM32F75xxx 和 STM32F74xxx 缓冲器。驱动程序必须扫描所有描述符，从 DMA 拥有的最后记录的位置到第一个位置。

对于同时发生的多个事件，一个中断只生成一次。驱动程序必须扫描 ETH_DMASR 寄存器，查找中断的原因。不会再次生成中断，除非在驱动程序已将 ETH_DMASR 寄存器中的适当位清零后发生新的中断事件。例如，控制器生成接收中断（ETH_DMASR 寄存器[6]）而驱动程序开始读取 ETH_DMASR 寄存器。然后，发生接收缓冲器不可用（ETH_DMASR 寄存器[7]）事件。驱动程序清除接收中断。即使是这时，也会由于活动或挂起的接收缓冲器不可用中断而生成一个新的中断。

图 503. 中断方案



38.7 以太网中断

以太网控制器有两个中断向量：一个专用于正常以太网操作，另一个仅当映射到 EXTI 线路 19 时用于以太网唤醒事件（带有唤醒帧或魔术数据包检测）。

如 *MAC 中断*和 *DMA 中断*章节所列，第一个以太网向量为 MAC 和 DMA 生成的中断而保留。

第二个向量为发生唤醒事件时 PMT 生成的中断而保留。唤醒事件对 EXTI 线路 19 的映射使 STM32F75xxx 和 STM32F74xxx 退出低功耗模式并生成一个中断。

当映射到 EXTI 线路 19 的以太网唤醒事件发生、使能了 MAC PMT 中断并且还使能了带上升沿检测的 EXTI 线路 19 中断时，会生成这两个中断。

可使用看门狗定时器（请参见 ETH_DMARSWTR 寄存器）灵活控制 RS 位（ETH_DMASR 寄存器）。当此看门狗定时器使用非零值编程时，只要 RxDMA 完成将接收的帧传输到系统存储器并且未触发接收状态（因未在相应接收描述符 (RDES1[31]) 使能接收状态），看门狗定时器即会被激活。当定时器按照编程值到时时，如果使能了 ETH_DMAIER 寄存器中的相应 RIE，则 RS 位会置 1 并会引发中断。当由于为该描述符将其使能而将一帧传输到存储器并且 RS 置 1 时，会在到时前禁止定时器。

注：　读取 PMT 控制和状态寄存器会自动将接收的唤醒帧和接收的魔术数据包 PMT 中断标志清零。但是，由于用于这些标志的寄存器位于 CLK_RX 域，因此在固件能发现此更新前可能有显著的延迟。当 RX 时钟很慢（在 10 Mbit 模式）和当 AHB 总线为高频时，该延迟会特别长。由于从 PMT 到 CPU 的中断请求基于 CLK_RX 域中的相同寄存器，所以即使在读取 PMT_CSR 之后，CPU 也可能错误地第二次调用中断例程。因此，可能需要固件轮询接收的唤醒帧和接收的魔术数据包位，并仅在发现它们都为 “0” 时退出中断服务程序。

38.8 以太网寄存器说明

外设寄存器可支持字节（8 位）、半字（16 位）或字（32 位）访问。

38.8.1 MAC 寄存器说明

以太网 MAC 配置寄存器 (ETH_MACCR)

Ethernet MAC configuration register

偏移地址：0x0000

复位值：0x0000 8000

MAC 配置寄存器是 MAC 的工作模式寄存器。它建立了接收和发送工作模式。

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
Res.	Res.	Res.	Res.	Res.	Res.	CSTF	Reserved	WD	JD	Res.	Res.	IFG			CSD	Res.	FES	ROD	LM	DM	IPCO	RD	Res.	APCS	BL		DC	TE	RE	Res.	Res.
						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。

CSTF：类型帧的 CRC 去除 (CRC stripping for Type frames)

位 25　该位置 1 时，在将帧转发到应用之前，
将去除并丢弃所有以太类型（类型字段大于 0x0600）帧的最后 4 个字节 (FCS)。

位 24保留，必须保持复位值。

位 23 **WD**：禁止看门狗 (Watchdog disable)
当该位置 1 时，MAC 禁止接收器上的看门狗定时器并可接收多达 16384 字节的帧。
当该位复位时，MAC 允许接收的帧不超过 2048 字节并会截断超过此限制接收的任何字节。

位 22 **JD**：禁止 Jabber (Jabber disable)
当该位置 1 时，MAC 禁止发送器上的 jabber 定时器并可发送多达 16384 字节的帧。
当该位复位时，如果发送期间应用程序发送超过 2 048 字节的数据，MAC 会截断发送器。

位 21:20 保留，必须保持复位值。

位 19:17 **IFG**：帧间间隙 (Interframe gap)
这些位控制发送期间帧间的最小间隙。
000：96 位时间
001：88 位时间
010：80 位时间
…
111：40 位时间
注：　在半双工模式下，最小 IFG 仅能配置为 64 位时间 (IFG = 100)。不考虑更低的值。

位 16 **CSD**：禁止载波侦听 (Carrier sense disable)

值设置得很高时，该位会使 MAC 发送器忽略半双工模式下帧发送期间的 MII CRS 信号。不会因在此发送期间载波丢失或无载波而生成错误。

当该位为低电平时，MAC 发送器会生成载波侦听错误，甚至中止发送。

位 15 保留，必须保持复位值。

位 14 **FES**：快速以太网速度 (Fast Ethernet speed)

指示快速以太网 (MII) 模式下的速度：

0：10 Mb/s

1：100 Mb/s

位 13 **ROD**：禁止接收自身 (Receive own disable)

当该位置 1 时，MAC 禁止在半双工模式下接收帧。

当该位复位时，MAC 接收发送时 PHY 提供的所有包。

如果 MAC 在全双工模式下工作，该位不适用。

位 12 **LM**：回送模式 (Loopback mode)

当该位置 1 时，MAC 在 MII 回送模式下工作。回送正确工作需要 MII 接收时钟输入 (RX_CLK)，因为发送时钟不是内部回送的。

位 11 **DM**：双工模式 (Duplex mode)

当该位置 1 时，MAC 在全双工模式下工作，此时它可以同时发送和接收。

位 10 **IPCO**：IPv4 校验和减荷 (IPv4 checksum offload)

置 1 时，该位使能接收的帧有效载荷的 TCP/UDP/ICMP 标头的 IPv4 校验和检查。当该位复位时，禁止接收器中的校验和减荷功能并始终将相应的 PCE 和 IP HCE 状态位（请参见 *第 1371 页上的表 240*）清零。

位 9 **RD**：禁止重试 (Retry disable)

当该位置 1 时，MAC 仅尝试一次发送。当在 MII 模式下发生冲突时，MAC 会忽略当前帧发送并以发送帧状态下过度冲突错误报告帧中止。

当该位复位时，MAC 会尝试根据 BL 的设置进行重试。

注：　该位仅在半双工模式下适用。

位 8 保留，必须保持复位值。

位 7 **APCS**：去除自动 pad/CRC (Automatic pad/CRC stripping)

当该位置 1 时，MAC 仅在长度字段值小于或等于 1500 字节时去除传入帧上的 Pad/FCS 字段。传送给应用程序的所有接收的帧，如果长度字段大于或等于 1501 字节，将都不去除 Pad/FCS 字段。

当该位复位时，MAC 会不加修改地传送所有传入的帧。

位 6:5 **BL**：后退限制 (Back-off limit)

后退限制决定发生冲突后重试期间 MAC 在重新安排一次发送之前等待的随机整数 (r) 个时间片延迟（对于 1000 Mb/s 为 4 096 位时间，对于 10/100 Mb/s 为 512 位时间）。

注：　该位仅在半双工模式下适用。

00：k = min (*n*, 10)

01：k = min (*n*, 8)

10：k = min (*n*, 4)

11：k = min (*n*, 1)

其中 *n* = 重新发送尝试的次数。随机整数 *r* 的取值范围为 $0 \le r < 2^k$。

位 4 **DC**：检查延迟 (Deferral check)

当该位置 1 时，便在 MAC 中使能了检查延迟功能。当发送状态机在 10/100 Mb/s 模式下延迟超过 24288 位时间时，MAC 将指示帧中止状态，同时在发送帧状态中将过度延迟错误位置 1。当发送器准备好发送但因 MII 上有有效 CRS（载波侦听）信号而被阻止时延迟开始。延迟时间是非累积性的。如果发送器延迟 10 000 位时间，然后发送、冲突、后退，然后在完成后退后不得不再次延迟，则延迟定时器复位为 0 并重新开始。

当该位复位时，禁止延迟检查功能，MAC 发生延迟，直到 CRS 信号变为无效信号。该位仅在半双工模式下适用。

位 3 **TE**：发送器使能 (Transmitter enable)

当该位置 1 时，针对 MII 上的发送使能 MAC 的发送状态机。当该位复位时，在当前帧的发送完成后禁止 MAC 发送状态机，并且不再发送任何帧。

位 2 **RE**：接收器使能 (Receiver enable)

当该位置 1 时，使能 MAC 的接收器状态机，以便从 MII 接收帧。当该位复位时，在当前帧接收完成后禁止 MAC 接收状态机，并且不再从 MII 接收任何帧。

位 1:0 保留，必须保持复位值。

以太网 MAC 帧过滤寄存器 (ETH_MACFFR)

Ethernet MAC frame filter register

偏移地址：0x0004
复位值：0x0000 0000

MAC 帧过滤寄存器包含用于接收帧的过滤控件。该寄存器的某些控件转到 MAC 的地址检查块，以执行第一级地址过滤。第二级过滤基于其它控件（例如传送不良帧和传送控制帧）对传入帧过滤。

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
RA	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HPF	SAF	SAIF	PCF		BFD	PAM	DAIF	HM	HU	PM
rw																					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **RA**：接收所有 (Receive all)

当该位置 1 时，MAC 接收器将所有接收的帧传送到应用程序，不管它们是否已通过地址过滤。SA/DA 过滤的结果在接收状态字的相应位更新（通过或失败）。当该位复位时，MAC 接收器传送给应用程序的只是通过了 SA/DA 地址过滤的那些帧。

位 30:11 保留，必须保持复位值。

位 10 **HPF**：散列或完美过滤器 (Hash or perfect filter)

当该位置 1 时，如果 HM 或 HU 位置 1，则地址过滤器传送与完美过滤或散列过滤匹配的帧。

当该位清零时，如果 HU 或 HM 位置 1，则仅传送与散列过滤器匹配的帧。

位 9 **SAF**：源地址过滤器 (Source address filter)

MAC 内核将接收的帧的 SA 字段与使能的 SA 寄存器中编程的值进行比较。如果比较结果匹配，则将 RxStatus 字中的 SAMatch 位设为高电平。当该位设为高电平且 SA 过滤失败时，MAC 放弃该帧。

当该位复位时，MAC 内核将接收的帧转发给应用程序。它还根据 SA 地址比较结果转发 RxStatus 中更新的 SA 匹配位。

位 8 **SAIF**：源地址反向过滤 (Source address inverse filtering)

当该位置 1 时，地址检查块在反向过滤模式下进行 SA 地址比较。SA 与 SA 寄存器匹配的帧被标记为 SA 地址过滤失败。

当该位复位时，SA 与 SA 寄存器不匹配的帧被标记为 SA 地址过滤失败。

位 7:6 **PCF**：传送控制帧 (Pass control frames)

这些位控制所有控制帧（包括单播和多播暂停帧）的转发。请注意暂停控制帧的处理只取决于流控制寄存器[2] 中的 RFCE。

00：MAC 阻止所有控制帧到达应用程序

01：MAC 将除了暂停控制帧以外的所有控制帧转发到应用程序

10：即使所有控制帧通过地址过滤失败，MAC 仍将它们转发给应用程序

11：MAC 转发通过地址过滤的控制帧。

这些位控制所有控制帧（包括单播和多播暂停帧）的转发。请注意暂停控制帧的处理只取决于流控制寄存器[2] 中的 RFCE。

00 或 01：MAC 阻止所有控制帧到达应用程序

10：即使所有控制帧通过地址过滤失败，MAC 仍将它们转发给应用程序

11：MAC 转发通过地址过滤的控制帧。

位 5 **BFD**：禁止广播帧 (Broadcast frames disable)

当该位置 1 时，地址过滤器对所有传入的广播帧进行过滤。

当该位复位时，地址过滤器传送所有接收到的广播帧。

位 4 **PAM**：传送所有多播 (Pass all multicast)

置 1 时，该位指示传送接收到的带多播目标地址（目标地址字段的第一位是 1）的所有帧。

复位时，多播帧的过滤取决于 HM 位。

位 3 **DAIF**：目标地址反向过滤 (Destination address inverse filtering)

当该位置 1 时，地址检查块在反向过滤模式下对单播和多播帧都进行 DA 地址比较。

复位时，正常过滤帧。

位 2 **HM**：散列多播 (Hash multicast)

置 1 时，MAC 根据散列表对接收到的多播帧执行目标地址过滤。

复位时，MAC 对多播帧执行完美目标地址过滤，即将 DA 字段与 DA 寄存器中编程的值进行比较。

位 1 **HU**：散列单播 (Hash unicast)

置 1 时，MAC 根据散列表对单播帧执行目标地址过滤。

复位时，MAC 对单播帧执行完美目标地址过滤，即将 DA 字段与 DA 寄存器中编程的值进行比较。

位 0 **PM**：混合模式 (Promiscuous mode)

当该位置 1 时，不论其目标或源地址为何，地址过滤器都传送所有传入的帧。PM 置 1 时，始终将接收状态字中 SA/DA 过滤失败状态位清零。

以太网 MAC 散列表高位寄存器 (ETH_MACHTHR)

Ethernet MAC hash table high register

偏移地址：0x0008

复位值：0x0000 0000

64 位散列表用于组地址过滤。对于散列过滤，传入帧中目标地址的内容通过 CRC 逻辑传送，CRC 寄存器中的高 6 位用于对散列表内容进行索引。CRC 是使用下列多项式编码的 32 位值（有关详细信息，请参见*第 38.5.3 节：MAC 帧接收*）：

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

最高有效位决定要使用的寄存器（散列表高位/散列表低位），其它 5 位决定寄存器中的位。散列值 0b0 0000 选取所选寄存器中的位 0，值 0b1 1111 选取所选寄存器中的位 31。

例如，如果传入帧的 DA 接收为 0x1F52 419C B6AF（0x1F 是 MII 接口上接收的第一个字节），则内部计算出的 6 位散列值为 0x2C 且为了过滤要检查 HTH 寄存器位[12]。如果传入帧的 DA 接收为 0xA00A 9800 0045，则计算出的 6 位散列值为 0x07 且为了过滤要检查 HTL 寄存器位[7]。

如果寄存器中对应的位值为 1，则接受该帧。否则拒绝该帧。如果 ETH_MACFFR 寄存器中的 PAM（传送所有多播）位置 1，则不论多播散列值为何都接受所有多播帧。

散列表高位寄存器包含多播散列表的高 32 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
HTH																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **HTH**：散列表高位 (Hash table high)
该字段包含散列表的高 32 位。

以太网 MAC 散列表低位寄存器 (ETH_MACHTLR)

Ethernet MAC hash table low register

偏移地址：0x000C

复位值：0x0000 0000

散列表低位寄存器包含多播散列表的低 32 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
HTL																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **HTL**：散列表低位 (Hash table low)
该字段包含散列表的低 32 位。

以太网 MAC MII 地址寄存器 (ETH_MACMIIAR)

Ethernet MAC MII address register

偏移地址：0x0010

复位值：0x0000 0000

MII 地址寄存器通过管理接口控制外部 PHY 的管理周期。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PA					MR					Res.	CR			MΩ	MB
																rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rc_w1

位 31:16 保留，必须保持复位值。

位 15:11 **PA**：PHY 地址 (PHY address)
该字段指示正在访问 32 个可能的 PHY 器件中的哪一个。

位 10:6 **MR**：MII 寄存器 (MII register)
这些位在所选 PHY 器件中选择需要的 MII 寄存器。

位 5 保留，必须保持复位值。

位 4:2 **CR**：时钟范围 (Clock range)

CR 时钟范围选项可确定 HCLK 频率并用于决定 MDC 时钟频率：
选项 HCLK MDC 时钟
000 60-100 MHz HCLK/42
001 100-150 MHzHCLK/62
010 20-35 MHz HCLK/16
011 35-60 MHz HCLK/26
100 150-168 MHz HCLK/102
101、110、111 保留 -

位 1 **MW**：MII 写 (MII write)

此位置 1 是在告知 PHY，将要启动一个使用 MII 数据寄存器的写操作。如果此位未置 1，则表示会启动一个读操作，将数据放入 MII 数据寄存器。

位 0 **MB**：MII 忙碌 (MII busy)

向 ETH_MACMIIAR 和 ETH_MACMIIDR 写入前，此位应读取逻辑 0。向 ETH_MACMIIAR 写入过程中，此位也必须复位为 0。在 PHY 寄存器访问过程中，此位由应用程序设为 0b1，指示读或写访问正在进行中。在对 PHY 进行写操作过程中，ETH_MACMIIDR（MII 数据）应始终保持有效，直到 MAC 将此位清零。在对 PHY 进行读操作过程中，ETH_MACMIIDR 始终无效，直到 MAC 将此位清零。在此位清零后，才可以向 ETH_MACMIIAR（MII 地址）写入。

以太网 MAC MII 数据寄存器 (ETH_MACMIIDR)

Ethernet MAC MII data register

偏移地址：0x0014

复位值：0x0000 0000

MAC MII 数据寄存器存储要写入 PHY 寄存器的数据，该寄存器地址在 ETH_MACMIIAR 中指定。ETH_MACMIIDR 也将存储从 PHY 寄存器读取的数据，该寄存器地址由 ETH_MACMIIAR 指定。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MD															
																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 **MD**：MII 数据 (MII data)

其中包含在某次管理读操作之后从 PHY 中读取的 16 位数据值，或在某次管理写操作之前要写入 PHY 的 16 位数据值。

以太网 MAC 流控制寄存器 (ETH_MACFCR)

Ethernet MAC flow control register

偏移地址：0x0018

复位值：0x0000 0000

流控制寄存器通过 MAC 来管理控制（暂停命令）帧的生成和接收。忙碌位置“1”时对寄存器的写操作将导致 MAC 生成暂停控制帧。控制帧的字段根据 802.3x 规范中的指定进行选择，此寄存器中的暂停时间值被用在控制帧的暂停时间字段中。忙碌位将保持置 1，直到将控制帧发送到电缆。主机必须确保向该寄存器写入前已将忙碌位清零。

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
PT																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ZQPD	Res.	PLT		UPFD	RFCE	TFCE	FCB/ BPA
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw									rw		rw	rw	rw	rw	rw	rc_w1 /rw

位 31:16 **PT**：暂停时间 (Pause time)

此字段保留发送控制帧中暂停时间字段要使用的值。如果将暂停时间位配置为与 MII 时钟域双同步，则只能在目标时钟域的至少 4 个时钟周期后执行对此寄存器的连续写操作。

位 15:8 保留，必须保持复位值。

位 7 **ZQPD**：零时间片暂停禁止 (Zero-quanta pause disable)

如果置 1，当来自 FIFO 层的流控制信号去断言后，此位会禁止自动生成零时间片暂停控制帧。

此位复位时，会使能自动零时间片暂停控制帧生成的正常操作。

位 6 保留，必须保持复位值。

位 5:4 **PLT**：暂停阈值下限 (Pause low threshold)

此字段配置暂停定时器的阈值，达到该值时，会自动重新传输暂停帧。该阈值应始终小于位 [31:16] 配置的暂停时间。例如，如果 PT = 100H（256 个时隙），而 PLT = 01，则在第一个 PAUSE 帧发送完成后，第二个 PAUSE 帧会在 228 (256 - 28) 个时隙启动时自动发送。

选项 阈值

00 暂停时间减去 4 个时隙

01 暂停时间减去 28 个时隙

10 暂停时间减去 144 个时隙

11 暂停时间减去 256 个时隙

时隙定义为 MII 接口每发送 512 位（64 字节）所需的时间。

位 3 **UPFD**：单播暂停帧检测 (Unicast pause frame detect)

当此位置 1 时，MAC 除了检测具有唯一多播地址的暂停帧外，还会检测具有 ETH_MACA0HR 和 ETH_MACA0LR 寄存器所指定的站单播地址的暂停帧。

当此位复位时，MAC 仅检测具有 802.3x 标准中指定的唯一多播地址的暂停帧。

位 2 **RFCE**：接收流控制使能 (Receive flow control enable)

当此位置 1 时，MAC 对接收到的暂停帧进行解码，并禁止其在指定时间（暂停时间）内发送。

当此位复位时，将禁止暂停帧的解码功能。

位 1 **TFCE**：发送流控制使能 (Transmit flow control enable)

在全双工模式下，当此位置 1 时，MAC 将使能流控制操作来发送暂停帧。当此位复位时，将禁止 MAC 中的流控制操作，MAC 不会传送任何暂停帧。

在半双工模式下，当此位置 1 时，MAC 将使能背压操作。当此位复位时，将禁止背压功能。

位 0 **FCB/BPA**：流控制忙碌/背压激活 (Flow control busy/back pressure activate)

此位在全双工模式下启动暂停控制帧，在半双工模式下，TFCE 位置 1 时，会激活背压功能。

全双工模式下，向流控制寄存器写入数据前此位应读为 0。要启动暂停控制帧，应用程序必须将此位置 1。在控制帧发送过程中，此位保持置 1 以指示帧发送正在进行中。当暂停控制帧发送完成后，MAC 会将此位复位为 0。将此位清零后，才可以对流控制寄存器执行写操作。

在半双工模式下，当此位置 1（TFCE 也置 1）时，MAC 内核将置位背压功能。在背压操作期间，当 MAC 接收到新帧时，发送器会开始发送一个导致冲突的 JAM 模式。当 MAC 配置为全双工模式时，将自动禁止 BPA。

以太网 MAC VLAN 标记寄存器 (ETH_MACVLANTR)

Ethernet MAC VLAN tag register

偏移地址：0x001C

复位值：0x0000 0000

VLAN 标记寄存器包含用于标识 VLAN 帧的 IEEE 802.1Q VLAN 标记。MAC 将所接收帧的第 13 及第 14 字节（长度/类型）与 0x8100 比较，随后的 2 个字节与 VLAN 标记比较，如果存在匹配，则将所接收帧状态中收到的 VLAN 位置 1。帧的合法长度从 1518 字节增加到 1522 字节。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VLANTC	VLANTI															
															rw	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 **VLANTC**：12 位 VLAN 标记比较 (12-bit VLAN tag comparison)

当此位置 1 时，使用 12 位 VLAN 标识符而不是完整的 16 位 VLAN 标记进行比较和过滤。VLAN 标记的位 [11:0] 与收到的 VLAN 标记帧的相应字段进行比较。

当此位复位时，收到的 VLAN 帧的第十五和第十六字节的全部 16 位都要进行比较。

位 15:0 **VLANTI**：VLAN 标记标识符 (VLAN tag identifier)（对应接收帧）

此字段包含用于标识 VLAN 帧的 802.1Q VLAN 标记，并与正在接收的 VLAN 帧的第十五和第十六字节进行比较。位 [15:13] 是用户优先级，位 [12] 是标准格式指示符 (CFI)，位 [11:0] 是 VLAN 标记的 VLAN 标识符 (VID) 字段。VLANTC 位置 1 时，仅使用 VID（位 [11:0]）进行比较。

如果 VLANTI（VLANTI[11:0]，如果 VLANTC 位置 1）全部为零，则 MAC 不会检查用于 VLAN 标记比较的第十五和第十六字节，而是将类型字段值为 0x8100 的所有帧均声明为 VLAN 帧。

以太网 MAC 远程唤醒帧过滤寄存器 (ETH_MACRWUFFR)

Ethernet MAC remote wakeup frame filter register

偏移地址：0x0028

复位值：0x0000 0000

应用程序将通过此地址对远程唤醒帧过滤寄存器进行写/读操作。实际上，唤醒帧过滤寄存器是八个（不透明的）此类唤醒帧过滤寄存器。对偏移 (0x0028) 的这个地址所进行的八个连续写操作会写入所有唤醒帧过滤寄存器。对偏移 (0x0028) 的这个地址所进行的八个连续读操作会读取所有唤醒帧过滤寄存器。此寄存器包含第 7 个 MAC 地址的高 16 位。有关详细信息，请参见*远程唤醒帧过滤寄存器*。

图 504. 以太网 MAC 远程唤醒帧过滤寄存器 (ETH_MACRWUFFR)

唤醒帧过滤寄存器 0	过滤器 0 字节掩码							
唤醒帧过滤寄存器 1	过滤器 1 字节掩码							
唤醒帧过滤寄存器 2	过滤器 2 字节掩码							
唤醒帧过滤寄存器 3	过滤器 3 字节掩码							
唤醒帧过滤寄存器 4	RSVD	过滤器 3 命令	RSVD	过滤器 2 命令	RSVD	过滤器 1 命令	RSVD	过滤器 0 命令
唤醒帧过滤寄存器 5	过滤器 3 偏移		过滤器 2 偏移		过滤器 1 偏移		过滤器 0 偏移	
唤醒帧过滤寄存器 6	过滤器 1 CRC - 16				过滤器 0 CRC - 16			
唤醒帧过滤寄存器 7	过滤器 3 CRC - 16				过滤器 2 CRC - 16			

ai15648

以太网 MAC PMT 控制和状态寄存器 (ETH_MACPMTCSR)

Ethernet MAC PMT control and status register

偏移地址：0x002C

复位值：0x0000 0000

ETH_MACPMTCSR 会配置唤醒事件请求并监视唤醒事件。

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
WFFRPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GU	Res.	Res.	WFR	MPR	Res.	Res.	WFE	MPE	PD
rs																						rw			rc_r	rc_r			rw	rw	rs

位 31 **WFFRPR**：唤醒帧过滤寄存器指针复位 (Wakeup frame filter register pointer reset)
当此位置 1 时，会将远程唤醒帧过滤寄存器指针复位为 0b000。它会在 1 个时钟周期后自动清零。

位 30:10 保留，必须保持复位值。

位 9 **GU**：全局单播 (Global unicast)
当此位置 1 时，它会将 MAC (DAF) 地址确认所过滤的任意单播包使能为唤醒帧。

位 8:7 保留，必须保持复位值。

位 6 **WFR**：接收到唤醒帧 (Wakeup frame received)
当此位置 1 时，表示因接收唤醒帧而生成了电源管理事件。通过对此寄存器执行读操作可将此位清零。

位 5 **MPR**：接收到魔术数据包 (Magic packet received)
当此位置 1 时，表示因接收魔术数据包而生成了电源管理事件。通过对此寄存器执行读操作可将此位清零。

位 4:3 保留，必须保持复位值。

位 2 **WFE**：唤醒帧使能 (Wakeup frame enable)
当此位置 1 时，会因为接收到唤醒帧而使能电源管理事件的生成。

位 1 **MPE**：魔术数据包使能 (Magic Packet enable)
当此位置 1 时，会因为接收到魔术数据包而使能电源管理事件的生成。

位 0 **PD**：掉电 (Power down)
当此位置 1 时，所有接收到的帧都将丢弃。接收到魔术数据包或唤醒帧时，此位会自动清零，同时禁止掉电模式。此位清零后收到的帧会转发到应用程序。只有魔术数据包使能 (Magic Packet Enable) 或唤醒帧使能 (Wakeup Frame Enable) 位置 1 时，才可以将此位置 1。

以太网 MAC 调试寄存器 (ETH_MACDBGR)

Ethernet MAC debug register

偏移地址：0x0034

复位值：0x0000 0000

此调试寄存器给出了发送及接收数据路径和 FIFO 的所有主要模块的状态。全零状态表示 MAC 内核处于空闲状态（FIFO 为空）并且数据路径中没有进行任何活动。

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
Res.	Res.	Res.	Res.	Res.	Res.	TFF	TFNE	Res.	TFWA	TFRS		MTP	MTFCS		MMTEA	Res.	Res.	Res.	Res.	Res.	Res.	RFFL		Res.	RFRCS		RFWRA	Res.	MSFRWCS		MMRPEA
						ro	ro		ro	ro	ro	ro	ro	ro	ro	Res	Res	Res	Res	Res	Res	ro	ro		ro	ro	ro		ro	ro	ro

位 31:26 保留，必须保持复位值。

位 25 **TFF**：Tx FIFO 已满 (Tx FIFO full)
当它为高电平时，表示 Tx FIFO 已满，因此无法再接受发送的帧。

位 24 **TFNE**：Tx FIFO 非空 (Tx FIFO not empty)
当它为高电平时，表示 TxFIFO 不为空，还留有一些数据要发送。

位 23 保留，必须保持复位值。

位 22 **TFWA**：Tx FIFO 写有效 (Tx FIFO write active)
当它为高电平时，表示 TxFIFO 写控制器有效且正在将数据传输到 TxFIFO。

位 21:20 **TFRS**：Tx FIFO 读状态 (Tx FIFO read status)
此字段指示 TxFIFO 读控制器的状态：
00：空闲状态
01：读状态（将数据传输到 MAC 发送器）
10：等待来自 MAC 发送器的 TxStatus
11：写入收到的 TxStatus 或刷新 TxFIFO

位 19 **MTP**：MAC 发送器处于暂停 (MAC transmitter in pause)
当它为高电平时，表示 MAC 发送器处于暂停条件（仅在全双工模式下），因此不会安排发送任何帧。

位 18:17 **MTFCS**：MAC 发送帧控制器状态 (MAC transmit frame controller status)
此字段指示 MAC 发送帧控制器的状态：
00：空闲
01：等待前一个帧的状态或 IFG/ 回退阶段结束
10：生成并发送暂停控制帧（在全双工模式下）
11：传输要发送的输入帧

位 16 **MMTEA**：MAC MII 发送引擎有效 (MAC MII transmit engine active)
当它为高电平时，表示 MAC MII 发送引擎正在主动发送数据而未处于空闲状态。

位 15:10 保留，必须保持复位值。

位 9:8 **RFFL**：Rx FIFO 填充级别 (Rx FIFO fill level)
它指示 Rx FIFO 填充级别的状态：
00：RxFIFO 为空
01：RxFIFO 填充级别低于流控制取消激活阈值
10：RxFIFO 填充级别高于流控制激活阈值
11：RxFIFO 已满

位 7 保留，必须保持复位值。

位 6:5 **RFRCS**：Rx FIFO 读控制器状态 (Rx FIFO read controller status)
它指示 Rx FIFO 读控制器的状态：
00：空闲状态
01：读取帧数据
10：读取帧状态（或时间戳）
11：刷新帧数据和状态

位 4 **RFWRA**：Rx FIFO 写控制器有效 (Rx FIFO write controller active)
当它为高电平时，表示 Rx FIFO 写控制器有效并正在将收到的帧传输到 FIFO。

位 3 保留，必须保持复位值。

位 2:1 **MSFRWCS**：MAC 小 FIFO 读 / 写控制器状态 (MAC small FIFO read / write controllers status)
当这些位为高电平时，表示 MAC 接收帧控制器模块的各个小 FIFO 读和写控制器的有效状态。

位 0 **MMRPEA**：MAC MII 接收协议引擎有效 (MAC MII receive protocol engine active)
当它为高电平时，表示 MAC MII 接收协议引擎正在主动接收数据而未处于空闲状态。

以太网 MAC 中断状态寄存器 (ETH_MACSR)

Ethernet MAC interrupt status register

偏移地址：0x0038

复位值：0x0000 0000

ETH_MACSR 寄存器内容在 MAC 中标识可生成中断的事件。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	TSTS	Res.	Res.	MMCTS	MMCRS	MMCS	PMTS	Res.	Res.	Res.
						rc_r			r	r	r	r			

位 15:10 保留，必须保持复位值。

位 9 **TSTS**：时间戳触发状态 (Time stamp trigger status)
当系统时间值等于或大于目标时间高位与低位寄存器中指定的值时，此位置 1。可通过读取 ETH_PTPTSSR 寄存器清零该标志。

位 8:7 保留，必须保持复位值。

位 6 **MMCTS**：MMC 发送状态 (MMC transmit status)
只要 ETH_MMCTIR 寄存器中生成中断，此位就为高电平。当此中断寄存器 (ETH_MMCTIR) 中的所有位都清零时，此位清零。

位 5 **MMCRS**：MMC 接收状态 (MMC receive status)
只要 ETH_MMCRIR 寄存器中生成中断，此位就为高电平。当此中断寄存器 (ETH_MMCRIR) 中的所有位都清零时，此位清零。

位 4 **MMCS**：MMC 状态 (MMC status)
只要位 6:5 中的任何一个位被设置为高电平，此位即会被设置为高电平。仅当这两个位均为低电平时，此位才被清零。

位 3 **PMTS**：PMT 状态 (PMT status)
只要在掉电模式下接收到魔术数据包或局域网唤醒帧，此位即被置 1（请参见 *第 1416 页上的以太网 MAC PMT 控制和状态寄存器 (ETH_MACPMTCSR)* ETH_MACPMTCSR 寄存器中的位 5 和 6）。当此最后一个寄存器的两个位 [6:5] 因对 ETH_MACPMTCSR 寄存器执行读取操作而被清零时，此位也将会清零。

位 2:0 保留，必须保持复位值。

以太网 MAC 中断屏蔽寄存器 (ETH_MACIMR)

Ethernet MAC interrupt mask register

偏移地址：0x003C

复位值：0x0000 0000

通过使用 ETH_MACIMR 寄存器位，可屏蔽因 ETH_MACSR 寄存器中的相应事件导致的中断信号。

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

位 15:10 保留，必须保持复位值。

位 9 **TSTIM**：时间戳触发中断屏蔽 (Time stamp trigger interrupt mask)
如果此位置 1，则会禁止生成时间戳中断。

位 8:4 保留，必须保持复位值。

位 3 **PMTIM**：PMT 中断屏蔽 (PMT interrupt mask)
如果此位置 1，则会禁止因将 ETH_MACSR 中的 PMT 状态位置 1 而触发中断信号。

位 2:0 保留，必须保持复位值。

以太网 MAC 地址 0 高位寄存器 (ETH_MACA0HR)

Ethernet MAC address 0 high register

偏移地址：0x0040

复位值：0x8000 FFFF

MAC 地址 0 高位寄存器可保存站的第一个 6 字节 MAC 地址的高 16 位。请注意，在 MII 接口上接收到的第一个 DA 字节与 MAC 地址低位寄存器的 LS 字节（位 [7:0]）相对应。例如，如果在 MII 上将 0x1122 3344 5566（0x11 为第一个字节）作为目标地址进行接收，则 MAC 地址 0 寄存器 [47:0] 将与 0x6655 4433 2211 进行比较。

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
MO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MACA0H															
1																rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **MO**：始终为 1 (Always 1)。

位 30:16 保留，必须保持复位值。

位 15:0 **MACA0H**：MAC 地址 0 高位 [47:32] (MAC address0 high [47:32])
此字段包含 6 字节 MAC 地址 0 的高 16 位 (47:32)。MAC 使用此字段过滤所接收的帧以及将 MAC 地址插入到发送流控制（暂停）帧中。

以太网 MAC 地址 0 低位寄存器 (ETH_MACA0LR)

Ethernet MAC address 0 low register

偏移地址：0x0044

复位值：0xFFFF FFFF

MAC 地址 0 低位寄存器可保存站的第一个 6 字节 MAC 地址的低 32 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
MACA0L																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **MACA0L**：MAC 地址 0 低位 [31:0] (MAC address0 low [31:0])

此字段包含 6 字节 MAC 地址 0 的低 32 位。MAC 使用此字段过滤所接收的帧以及将 MAC 地址插入到发送流控制（暂停）帧中。

以太网 MAC 地址 1 高位寄存器 (ETH_MACA1HR)

Ethernet MAC address 1 high register

偏移地址：0x0048

复位值：0x0000 FFFF

MAC 地址 1 高位寄存器可保存站的第二个 6 字节 MAC 地址的高 16 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
AE	SA	MBC						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MACA1H															
rw	rw	rw	rw	rw	rw	rw	rw									rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **AE**：地址使能 (Address enable)

此位置 1 时，地址过滤器使用 MAC 地址 1 实现完美过滤。此位清零时，地址过滤器会忽略用于过滤的地址。

位 30 **SA**：源地址 (Source address)

此位置 1 时，将使用 MAC 地址 1 [47:0] 与所接收帧的 SA 字段进行比较。
此位清零时，将使用 MAC 地址 1 [47:0] 与所接收帧的 DA 字段进行比较。

位 29:24 **MBC**：屏蔽字节控制 (Mask byte control)

这些位是用于比较每个 MAC 地址 1 字节的屏蔽控制位。当将它们设为高电平时，MAC 内核不会将所接收到的 DA/SA 的相应字节与 MAC 地址 1 寄存器的内容进行比较。每个位都用于控制字节的屏蔽，如下所示：

– 位 29：ETH_MACA1HR [15:8]
– 位 28：ETH_MACA1HR [7:0]
– 位 27：ETH_MACA1LR [31:24]
…
– 位 24：ETH_MACA1LR [7:0]

位 23:16 保留，必须保持复位值。

位 15:0 **MACA1H**：MAC 地址 1 高位 [47:32] (MAC address1 high [47:32])

此字段包含第二个 6 字节 MAC 地址的高 16 位 (47:32)。

以太网 MAC 地址 1 低位寄存器 (ETH_MACA1LR)

Ethernet MAC address1 low register

偏移地址：0x004C

复位值：0xFFFF FFFF

MAC 地址 1 低位寄存器可保存站的第二个 6 字节 MAC 地址的低 32 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
MACA1L																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **MACA1L**：MAC 地址 1 低位 [31:0] (MAC address1 low [31:0])

此字段包含 6 字节 MAC 地址 1 的低 32 位。如果在初始化过程之后，应用程序未加载此字段的内容，则该内容将不会被定义。

以太网 MAC 地址 2 高位寄存器 (ETH_MACA2HR)

Ethernet MAC address 2 high register

偏移地址：0x0050

复位值：0x0000 FFFF

MAC 地址 2 高位寄存器可保存站的第二个 6 字节 MAC 地址的高 16 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
AE	SA	MBC						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MACA2H															
rw	rw	rw	rw	rw	rw	rw	rw									rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **AE**：地址使能 (Address enable)

此位置 1 时，地址过滤器使用 MAC 地址 2 实现完美过滤。复位后，地址过滤器会忽略用于过滤的地址。

位 30 **SA**：源地址 (Source address)

此位置 1 时，将使用 MAC 地址 2 [47:0] 与所接收帧的 SA 字段进行比较。

此位被复位时，将使用 MAC 地址 2 [47:0] 与所接收帧的 DA 字段进行比较。

位 29:24 **MBC**：屏蔽字节控制 (Mask byte control)

这些位是用于比较每个 MAC 地址 2 字节的屏蔽控制位。当将其设为高电平时，MAC 内核不会将所接收到的 DA/SA 的相应字节与 MAC 地址 2 寄存器的内容进行比较。每个位都用于控制字节的屏蔽，如下所示：

– 位 29：ETH_MACA2HR [15:8]
– 位 28：ETH_MACA2HR [7:0]
– 位 27：ETH_MACA2LR [31:24]
…
– 位 24：ETH_MACA2LR [7:0]

位 23:16保留，必须保持复位值。

位 15:0 **MACA2H**：MAC 地址 2 高位 [47:32] (MAC address2 high [47:32])

此字段包含 6 字节 MAC 地址 2 的高 16 位 (47:32)。

以太网 MAC 地址 2 低位寄存器 (ETH_MACA2LR)

Ethernet MAC address 2 low register

偏移地址：0x0054

复位值：0xFFFF FFFF

MAC 地址 2 低位寄存器可保存站的第二个 6 字节 MAC 地址的低 32 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
MACA2L																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **MACA2L**：MAC 地址 2 低位 [31:0] (MAC address2 low [31:0])

此字段包含第二个 6 字节 MAC 地址 2 的低 32 位。如果在初始化过程之后，应用程序未加载此字段的内容，则该内容将不会被定义。

以太网 MAC 地址 3 高位寄存器 (ETH_MACA3HR)

Ethernet MAC address 3 high register

偏移地址：0x0058

复位值：0x0000 FFFF

MAC 地址 3 高位寄存器可保存站的第二个 6 字节 MAC 地址的高 16 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
AE	SA	MBC						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MACA3H															
rw	rw	rw	rw	rw	rw	rw	rw									rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **AE**：地址使能 (Address enable)

此位置 1 时，地址过滤器使用 MAC 地址 3 实现完美过滤。此位清零时，地址过滤器会忽略用于过滤的地址。

位 30 **SA**：源地址 (Source address)

此位置 1 时，将使用 MAC 地址 3 [47:0] 与所接收帧的 SA 字段进行比较。

此位被清零时，将使用 MAC 地址 3 [47:0] 与所接收帧的 DA 字段进行比较。

位 29:24 **MBC**：屏蔽字节控制 (Mask byte control)

这些位是用于比较每个 MAC 地址 3 字节的屏蔽控制位。当将这些位设为高电平时，MAC 内核不会将所接收到的 DA/SA 的相应字节与 MAC 地址 3 寄存器的内容进行比较。每个位都用于控制字节的屏蔽，如下所示：

– 位 29：ETH_MACA3HR [15:8]
– 位 28：ETH_MACA3HR [7:0]
– 位 27：ETH_MACA3LR [31:24]

…

– 位 24：ETH_MACA3LR [7:0]

位 23:16 保留，必须保持复位值。

位 15:0 **MACA3H**：MAC 地址 3 高位 [47:32] (MAC address3 high [47:32])

该字段包含 6 字节 MAC 地址 3 的高 16 位 (47:32)。

以太网 MAC 地址 3 低位寄存器 (ETH_MACA3LR)

Ethernet MAC address 3 low register

偏移地址：0x005C

复位值：0xFFFF FFFF

MAC 地址 3 低位寄存器可保存站的第二个 6 字节 MAC 地址的低 32 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
MACA3L																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **MACA3L**：MAC 地址 3 低位 [31:0] (MAC address3 low [31:0])

此字段包含第二个 6 字节 MAC 地址 3 的低 32 位。如果在初始化过程之后，应用程序未加载此字段的内容，则该内容将不会被定义。

38.8.2 MMC 寄存器说明

以太网 MMC 控制寄存器 (ETH_MMCCR)

Ethernet MMC control register

偏移地址：0x0100

复位值：0x0000 0000

以太网 MMC 控制寄存器用于建立管理计数器的工作模式。

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
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	MCFHP	MCP	MCF	ROR	CSR	CR
																										rw	rw	rw	rw	rw	rw

位 31:6 保留，必须保持复位值。

位 5 **MCFHP**：MMC 计数器全-半预设置 (MMC counter Full-Half preset)

当 MCFHP 为低电平且位 4 置 1 时，所有 MMC 计数器均预设为几乎一半值。所有帧计数器均预设为 0x7FFF_FFF0（一半值 - 16）

当 MCFHP 为高电平且位 4 置 1 时，所有 MMC 计数器均预设为几乎全值。所有帧计数器均预设为 0xFFFF_FFF0（全值 - 16）

位 4 **MCP**：MMC 计数器预设置 (MMC counter preset)

该位置 1 时，所有计数器都将根据上面的位 5 初始化或预设为几乎全值或几乎一半值。该位将在 1 个时钟周期后自动清零。该位连同位 5 用于调试和测试由于 MMC 计数器变为半全值或全值而导致的中断产生。

位 3 **MCF**：MMC 计数器冻结 (MMC counter freeze)

该位置 1 时，会冻结所有 MMC 计数器，使其保持为当前值。（该位清零后，才会因存在发送帧或接收帧而对 MMC 计数器进行更新。在此模式下，“读取时复位”位置 1 时，如果读取任何 MMC 计数器，则该计数器也会清零。）

位 2 **ROR**：读取时复位 (Reset on read)
该位置 1 时，读取 MMC 计数器后，该计数器会复位为零（复位后自清）。读取最低有效字节通道（位 [7:0]）后，计数器会清零。

位 1 **CSR**：计数器停止翻转 (Counter stop rollover)
该位置 1 时，计数器达到其最大值后，不会返回到零。

位 0 **CR**：计数器复位 (Counter reset)
该位置 1 时，所有计数器都将复位。该位在 1 个时钟周期后自动清零。

以太网 MMC 接收中断寄存器 (ETH_MMCRIR)

Ethernet MMC receive interrupt register

偏移地址：0x0104

复位值：0x0000 0000

以太网 MMC 接收中断寄存器用于保持当接收统计计数器达到其最大值的一半时所生成的中断。（计数器的 MSB 置 1。）该寄存器是一个 32 位宽的寄存器。当读取引发中断的各个 MMC 计数器时，会将中断位清零。必须读取各个计数器的最低有效字节通道（位 [7:0]），才能将中断位清零。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RGUFS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RFAES	RFCES	Res.	Res.	Res.	Res.	Res.
														rc_r											rc_r	rc_r					

位 31:18 保留，必须保持复位值。

位 17 **RGUFS**：接收的良好单播帧状态 (Received Good Unicast Frames Status)
当接收到的良好单播帧计数器达到其最大值的一半时，该位置 1。

位 16:7 保留，必须保持复位值。

位 6 **RFAES**：接收的帧对齐错误状态 (Received frames alignment error status)
当存在对齐错误的接收帧计数器达到其最大值的一半时，该位置 1。

位 5 **RFCES**：接收的帧 CRC 错误状态 (Received frames CRC error status)
当存在 CRC 错误的接收帧计数器达到其最大值的一半时，该位置 1。

位 4:0 保留，必须保持复位值。

以太网 MMC 发送中断寄存器 (ETH_MMCTIR)

Ethernet MMC transmit interrupt register

偏移地址：0x0108

复位值：0x0000 0000

以太网 MMC 发送中断寄存器用于保持当发送统计计数器达到其最大值的一半时所生成的中断。（计数器的 MSB 置 1。）该寄存器是一个 32 位宽的寄存器。当读取引发中断的各个 MMC 计数器时，会将中断位清零。必须读取各个计数器的最低有效字节通道（位 [7:0]），才能将中断位清零。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TGFS	Res.	Res.	Res.	Res.	Res.	TGFMSCS	TGFSCS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
										rc_r						rc_r	rc_r														

位 31:22 保留，必须保持复位值。

位 21 **TGFS**：发送的良好帧状态 (Transmitted good frames status)
当发送的良好帧计数器达到其最大值的一半时，该位置 1。

位 20:16 保留，必须保持复位值。

位 15 **TGFMSCS**：多个冲突后发送的良好帧状态 (Transmitted good frames more single collision status)
多个冲突后发送的良好帧计数器达到其最大值的一半时，该位置 1。

位 14 **TGFSCS**：单个冲突后发送的良好帧状态 (Transmitted good frames single collision status)
单个冲突后发送的良好帧计数器达到其最大值的一半时，该位置 1。

位 13:0 保留，必须保持复位值。

以太网 MMC 接收中断屏蔽寄存器 (ETH_MMCRIMR)

Ethernet MMC receive interrupt mask register

偏移地址：0x010C

复位值：0x0000 0000

以太网 MMC 接收中断屏蔽寄存器用于保持当接收统计计数器达到其最大值的一半时所生成中断的屏蔽。（计数器的 MSB 置 1。）该寄存器是一个 32 位宽的寄存器。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RGUFM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RFAEM	RFCEM	Res.	Res.	Res.	Res.	Res.
														rw											rw	rw					

位 31:18 保留，必须保持复位值。

位 17 **RGUFM**：接收的良好单播帧屏蔽 (Received good unicast frames mask)
如果该位置 1，当接收到的良好单播帧计数器达到其最大值的一半时，会屏蔽中断。

位 16:7 保留，必须保持复位值。

位 6 **RFAEM**：接收的帧对齐错误屏蔽 (Received frames alignment error mask)
如果该位置 1，当存在对齐错误的接收帧计数器达到其最大值的一半时，会屏蔽中断。

位 5 **RFCEM**：接收的帧 CRC 错误屏蔽 (Received frame CRC error mask)
如果该位置 1，当存在 CRC 错误的接收帧计数器达到其最大值的一半时，会屏蔽中断。

位 4:0 保留，必须保持复位值。

以太网 MMC 发送中断屏蔽寄存器 (ETH_MMCTIMR)

Ethernet MMC transmit interrupt mask register

偏移地址：0x0110

复位值：0x0000 0000

以太网 MMC 发送中断屏蔽寄存器用于保持当发送统计计数器达到其最大值的一半时所生成中断的屏蔽。（计数器的 MSB 置 1）。该寄存器是一个 32 位宽的寄存器。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TGFM	Res.	Res.	Res.	Res.	Res.	TGFMSCM	TGFSCM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
										rw						rw	rw														

位 31:22 保留，必须保持复位值。

位 21 **TGFM**：发送的良好帧屏蔽 (Transmitted good frames mask)
如果该位置 1，当发送的良好帧计数器达到其最大值的一半时，会屏蔽中断。

位 20:16 保留，必须保持复位值。

位 15 **TGFMSCM**：多个冲突后发送的良好帧屏蔽 (Transmitted good frames more single collision mask)
如果该位置 1，当多个冲突后发送的良好帧计数器达到其最大值的一半时，会屏蔽中断。

位 14 **TGFSCM**：单个冲突后发送的良好帧屏蔽 (Transmitted good frames single collision mask)
如果该位置 1，当单个冲突后发送的良好帧计数器达到其最大值的一半时，会屏蔽中断。

位 13:0 保留，必须保持复位值。

以太网 MMC 在单个冲突后发送的良好帧计数器寄存器 (ETH_MMCTGFSCCR)

Ethernet MMC transmitted good frames after a single collision counter register

偏移地址：0x014C

复位值：0x0000 0000

该寄存器包含在半双工模式下于单个冲突后成功发送的帧的数量。

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
TGFSCC																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **TGFSCC**：单个冲突后发送的良好帧计数器 (Transmitted good frames single collision counter)
单个冲突后发送的良好帧计数器。

以太网 MMC 在多个冲突后发送的良好帧计数器寄存器 (ETH_MMCTGFMSCCR)

Ethernet MMC transmitted good frames after more than a single collision counter register

偏移地址：0x0150

复位值：0x0000 0000

该寄存器包含在半双工模式下于多个冲突后成功发送的帧的数量。

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
TGFMSCC																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **TGFMSCC**：多个冲突后发送的良好帧计数器 (Transmitted good frames more single collision counter)

多个冲突后发送的良好帧计数器

以太网 MMC 发送的良好帧计数器寄存器 (ETH_MMCTGFCR)

Ethernet MMC transmitted good frames counter register

偏移地址：0x0168

复位值：0x0000 0000

该寄存器包含发送的良好帧的数量。

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
TGFC																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **TGFC**：发送的良好帧计数器 (Transmitted good frames counter)

带有 CRC 错误计数器寄存器的以太网 MMC 接收帧 (ETH_MMCRFCECR)

Ethernet MMC received frames with CRC error counter register

偏移地址：0x0194

复位值：0x0000 0000

此寄存器包含接收的含 CRC 错误的帧数。

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
RFCEC																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **RFCEC**：接收的帧 CRC 错误计数器 (Received frames CRC error counter)

接收的带有 CRC 错误计数器的帧

带有对齐错误计数器寄存器的以太网 MMC 接收帧 (ETH_MMCRFAECR)

Ethernet MMC received frames with alignment error counter register

偏移地址：0x0198

复位值：0x0000 0000

此寄存器包含接收的含对齐 (dribble) 错误的帧数。

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
RFAEC																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **RFAEC**：接收的帧对齐错误计数器 (Received frames alignment error counter)
接收的带有对齐错误计数器的帧

MMC 接收的良好单播帧计数器寄存器 (ETH_MMCRGUFCR)

MMC received good unicast frames counter register

偏移地址：0x01C4

复位值：0x0000 0000

此寄存器包含接收的良好单播帧数。

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
RGUFC																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **RGUFC**：接收的良好单播帧计数器 (Received good unicast frames counter)

38.8.3 IEEE 1588 时间戳寄存器

IEEE 1588 time stamp registers

本节介绍用于支持符合 IEEE 1588 标准的精确网络时钟同步功能的寄存器。

以太网 PTP 时间戳控制寄存器 (ETH_PTPTSCR)

Ethernet PTP time stamp control register

偏移地址：0x0700

复位值：0x0000 2000

此寄存器用于控制时间戳生成和更新逻辑。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSPFFMAE	TSCNT		TSSMRME	TSSEME	TSSIPV4FE	TSSIPV6FE	TSSPTPOEFE	TSPTPPSV2E	TSSSR	TSSARFE	Res.	Res.	TTSARU	TSITE	TSSTU	TSSTI	TSFCU	TSE
													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 **TSPFFMAE**：时间戳 PTP 帧过滤 MAC 地址使能 (Time stamp PTP frame filtering MAC address enable)

如果置 1，则当 PTP 直接通过以太网发送时，此位将使用 MAC 地址（除 MAC 地址 0）来过滤 PTP 帧。

位 17:16 **TSCNT**：时间戳时钟节点类型 (Time stamp clock node type)

可用的时钟节点类型如下：

00：普通时钟

01：边界时钟

10：端对端透明时钟

11：点对点透明时钟

位 15 **TSSMRME**：主节点相关消息时间戳快照使能 (Time stamp snapshot for message relevant to master enable)

当此位置 1 时，仅拍摄主节点相关消息的快照。当此位清零时，仅拍摄从节点相关消息的快照。这仅适用于普通时钟和边界时钟节点。

位 14 **TSSEME**：事件消息时间戳快照使能 (Time stamp snapshot for event message enable)

当此位置 1 时，仅拍摄事件消息的时间戳快照（SYNC、Delay_Re、Pdelay_Req 或 Pdelay_Resp）。当此位清零时，将拍摄除 Announce、Management 和 Signaling 以外其它所有消息的快照。

位 13 **TSSIPV4FE**：IPv4 帧时间戳快照使能 (Time stamp snapshot for IPv4 frames enable)

当此位置 1 时，将拍摄 IPv4 帧的时间戳快照。

位 12 **TSSIPV6FE**：IPv6 帧时间戳快照使能 (Time stamp snapshot for IPv6 frames enable)

当此位置 1 时，将拍摄 IPv6 帧的时间戳快照。

位 11 **TSSPTPOEFE**：PTP over ethernet 帧时间戳快照使能 (Time stamp snapshot for PTP over ethernet frames enable)

当此位置 1 时，将拍摄在以太网帧中也含有 PTP 消息 (PTP over Ethernet) 的帧的时间戳快照。默认情况下，将拍摄 UDP-IPEthernet PTP 数据包的快照。

位 10 **TSPTPPSV2E**：监听版本 2 格式时间戳 PTP 数据包使能 (Time stamp PTP packet snooping for version2 format enable)

当此位置 1 时，将使用版本 2 格式对 PTP 数据包进行监听。当此位清零时，将使用版本 1 格式对 PTP 数据包进行监听。

注： IEEE 1588 版本 1 和版本 2 格式符合 IEEE 标准 1588-2008（版本为 IEEE STD. 1588-2002）。

位 9 **TSSSR**：时间戳亚秒翻转：数字或二进制翻转控制 (Time stamp subsecond rollover: digital or binary rollover control)

当此位置 1 时，如果亚秒计数器达到值 0x3B9A C9FF（十进制 999 999 999）并递增了时间戳（高位）秒数，则时间戳低位寄存器将翻转。

当此位清零时，亚秒寄存器的翻转值将达到 0x7FFF FFFF。亚秒增量必须根据 PTP 参考时钟频率和此位的值正确进行编程。

位 8 **TSSARFE**：所有接收帧的时间戳快照使能 (Time stamp snapshot for all received frames enable)

当此位置 1 时，时间戳快照会针对内核所接收的全部帧处于使能状态。

位 7:6 保留，必须保持复位值。

位 5 **TSARU**：时间戳加数寄存器更新 (Time stamp addend register update)

当此位置 1 时，时间戳加数寄存器的内容会被更新到 PTP 进行精密校准。更新结束时此位会清零。此寄存器位必须在读为零后，才可以将其置 1。

位 4 **TSITE**：时间戳中断触发使能 (Time stamp interrupt trigger enable)

当此位置 1 时，如果系统时间大于在目标时间寄存器中写入的值，将产生时间戳中断。产生时间戳触发中断时，此位会清零。

位 3 **TSSTU**：时间戳系统时间更新 (Time stamp system time update)

当此位置 1 时，系统时间将通过在时间戳高位更新寄存器和时间戳低位更新寄存器中指定的值进行更新（增至其中或从中减去）。必须将 TSSTU 和 TSSTI 两位均读为零，才可以将此位置 1。硬件中的更新完成后，此位会清零。

位 2 **TSSTI**：时间戳系统时间初始化 (Time stamp system time initialize)

当此位置 1 时，系统时间将通过在时间戳高位更新寄存器和时间戳低位更新寄存器中指定的值进行初始化（覆盖）。此位必须在读为零后，才可以将其置 1。初始化完成后，此位会清零。

位 1 **TSFCU**：时间戳精密更新或粗略更新 (Time stamp fine or coarse update)

置 1 时，此位表示将使用精密更新方法更新系统时间戳。清零时，此位表示将使用粗略方法更新系统时间戳。

位 0 **TSE**：时间戳使能 (Time stamp enable)

当此位置 1 时，时间戳针对发送和接收帧处于使能状态。当此位清零时，时间戳功能将挂起，不会为发送和接收帧添加时间戳。由于保持的系统时间处于挂起状态，所以在将此位置为高电平后，始终都需要初始化时间戳功能（系统时间）。

下表指出了拍摄快照时的各种消息（取决于事件消息寄存器设置的时钟、使能主节点和使能快照）。

表 244. 时间戳快照对寄存器位的相依性

TSCNT（位 17:16）	TSSMRME（位 15）(1)	TSSEME（位 14）	拍摄快照时的消息
00 或 01	X(2)	0	SYNC、Follow_Up、Delay_Req、Delay_Resp
00 或 01	1	1	Delay_Req
00 或 01	0	1	SYNC
10	N/A	0	SYNC、Follow_Up、Delay_Req、Delay_Resp
10	N/A	1	SYNC、Follow_Up
11	N/A	0	SYNC、Follow_Up、Delay_Req、Delay_Resp、Pdelay_Req、Pdelay_Resp
11	N/A	1	SYNC、Pdelay_Req、Pdelay_Resp

1. N/A = 不适用。
2. X = 无关。

以太网 PTP 亚秒递增寄存器 (ETH_PTPSSIR)

Ethernet PTP subsecond increment register

偏移地址：0x0704

复位值：0x0000 0000

此寄存器包含亚秒寄存器递增时使用的 8 位值。在粗略更新模式下（ETH_PTPTSCR 中的 TSFCU 位），此寄存器中的值在每个 HCLK 时钟周期后都会被添加到系统时间。在精密更新模式下，此寄存器中的值将在累加器溢出时被添加到系统时间。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	STSSI							
																								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:0 **STSSI**：系统时间亚秒增量 (System time subsecond increment)

此寄存器中编程的值在每次更新时都会被增加到系统时间亚秒值的内容中。

例如，要得到 20 ns 的精度，该值为：20 / 0.467 = ~ 43（或 0x2A）。

以太网 PTP 时间戳高位寄存器 (ETH_PTPTSHR)

Ethernet PTP time stamp high register

偏移地址：0x0708

复位值：0x0000 0000

此寄存器包含最高有效（高位）32 个时间位。这个只读寄存器包含秒系统时间值。时间戳高位寄存器与时间戳低位寄存器可指示由 MAC 保持的系统时间的当前值。但是，它可以连续进行更新。

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
STS																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **STS**：系统时间秒 (System time second)

此字段中的值表示由内核维持的以秒为单位的系统时间当前值。

以太网 PTP 时间戳低位寄存器 (ETH_PTPTSLR)

Ethernet PTP time stamp low register

偏移地址：0x070C

复位值：0x0000 0000

此寄存器包含最低有效（低位）32 个时间位。这个只读寄存器包含亚秒系统时间值。

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
STPNS	STSS																														
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31 **STPNS**：系统时间正负号 (System time positive or negative sign)

该位指示正负时间值。置 1 时，该位指示时间表示为负值。清零时，该位指示时间表示为正值。由于系统时间始终应为正值，因此该位通常为零。

位 30:0 **STSS**：系统时间亚秒 (System time subseconds)

该字段中的值包含亚秒时间表示，精度为 0.46 ns。

以太网 PTP 时间戳高位更新寄存器 (ETH_PTPTSHUR)

Ethernet PTP time stamp high update register

偏移地址：0x0710

复位值：0x0000 0000

该寄存器包含要写入、增加到系统时间值或从系统时间值中减去的最高有效（高）32 位时间。时间戳高位更新寄存器与时间戳低位更新寄存器一起用于初始化或更新 MAC 维护的系统时间。在将时间戳控制寄存器中的 TSSTI 或 TSSTU 位置 1 前，必须写入这两个寄存器。

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
TSUS																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **TSUS**：时间戳更新秒 (Time stamp update second)

该字段中的值指示要初始化或增加到系统时间中的时间，单位为秒。

以太网 PTP 时间戳低位更新寄存器 (ETH_PTPTSLUR)

Ethernet PTP time stamp low update register

偏移地址：0x0714

复位值：0x0000 0000

该寄存器包含要写入、增加到系统时间值或从系统时间值中减去的最低有效（低）32 位时间。

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
TSUPNS	TSUSS																														
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 **TSUPNS**：时间戳更新正负号 (Time stamp update positive or negative sign)

该位指示正负时间值。置 1 时，该位指示时间表示为负值。清零时，该位指示时间表示为正值。当 TSSTI 置 1（系统时间初始化）时，该位应为零。如果当 TSSTU 置 1 时该位也置 1，则时间戳更新寄存器中的值要从系统时间中减去。否则，它要增加到系统时间中。

位 30:0 **TSUSS**：时间戳更新亚秒 (Time stamp update subseconds)

该字段中的值指示要初始化或增加到系统时间中的亚秒时间。该值的精度为 0.46 ns（换句话说，值 0x0000_0001 即为 0.46 ns）。

以太网 PTP 时间戳加数寄存器 (ETH_PTPTSAR)

Ethernet PTP time stamp addend register

偏移地址：0x0718

复位值：0x0000 0000

该寄存器由软件用来重新线性调节时钟频率，使其与主时钟频率匹配。仅当系统时间配置为微调更新模式（ETH_PTPTSCR 中的 TSFCU 位）时使用该寄存器值。该寄存器的内容在每个时钟周期添加到一个 32 位累加器中，系统时间在该累加器发生上溢时更新。

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
TSA																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **TSA**：时间戳加数 (Time stamp addend)

该寄存器指示要添加到累加器寄存器以实现时间同步的 32 位时间值。

以太网 PTP 目标时间高位寄存器 (ETH_PTPTTHR)

Ethernet PTP target time high register

偏移地址：0x071C

复位值：0x0000 0000

该寄存器包含要与用于中断事件生成的系统时间进行比较的高 32 位时间。目标时间高位寄存器与目标时间低位寄存器一起，用于在系统时间超过这些寄存器中编程的值时规划中断事件（ETH_PTPTSCR 中的 TSARU 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
TTSH																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **TTSH**：目标时间戳高位 (Target time stamp high)

该寄存器存储以秒为单位的时间。当时间戳的值匹配或同时超过两个目标时间戳寄存器时，MAC（如果已使能）会生成中断。

以太网 PTP 目标时间低位寄存器 (ETH_PTPTTLR)

Ethernet PTP target time low register

偏移地址：0x0720

复位值：0x0000 0000

该寄存器包含要与用于中断事件生成的系统时间进行比较的低 32 位时间。

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
TTSL																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **TTSL**：目标时间戳低位 (Target time stamp low)

该寄存器存储以纳秒为单位的时间（带符号）。当时间戳的值匹配或同时超过两个目标时间戳寄存器时，MAC（如果已使能）会生成中断。

以太网 PTP 时间戳状态寄存器 (ETH_PTPTSSR)

Ethernet PTP time stamp status register

偏移地址：0x0728

复位值：0x0000 0000

该寄存器包含时间戳状态寄存器。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSTTR	TSSO
																														ro	ro

位 31:2 保留，必须保持复位值。

位 1 **TSTTR**：达到时间戳目标时间 (Time stamp target time reached)

置 1 时，该位指示系统时间值大于或等于在目标时间高位和低位寄存器中指定的值。对 ETH_PTPTSSR 寄存器执行读操作时此位清零。

位 0 **TSSO**：时间戳秒上溢 (Time stamp second overflow)

置 1 时，该位指示时间戳的秒值已上溢，超过 0xFFFF FFFF。

以太网 PTP PPS 控制寄存器 (ETH_PTPPPSCR)

Ethernet PTP PPS control register

偏移地址：0x072C

复位值：0x0000 0000

该寄存器控制 PPS 输出的频率。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSTTR	TSSO
																														ro	ro

位 31:4 保留，必须保持复位值。

位 3:0 **PPSFREQ**：PPS 频率选择 (PPS frequency selection)

PPS 输出频率设置为 $2^{PPSFREQ}$ Hz。

0000：1 Hz，使用二进制翻转时，脉冲宽度为 125 ms，使用数字翻转时，脉冲宽度为 100 ms

0001：2 Hz，使用二进制翻转时，占空比为 50%（不建议使用数字翻转）

0010：4 Hz，使用二进制翻转时，占空比为 50%（不建议使用数字翻转）

0011：8 Hz，使用二进制翻转时，占空比为 50%（不建议使用数字翻转）

0100：16 Hz，使用二进制翻转时，占空比为 50%（不建议使用数字翻转）

...

1111：32768 Hz，使用二进制翻转时，占空比为 50%（不建议使用数字翻转）

注：　如果使用数字翻转（TSSSR = 1，ETH_PTPTSCR 中的位 9），建议不要使用频率不是 1 Hz 的 PPS 输出。否则，使用数字翻转时，PPS 输出在更高的频率下的波形会不规则（尽管其平均频率在任何一秒窗口期间始终正确）。

38.8.4 DMA 寄存器说明

本节定义了每个 DMA 寄存器的位。只要地址是字对齐的，就允许非 32 位访问。

以太网 DMA 总线模式寄存器 (ETH_DMABMR)

Ethernet DMA bus mode register

偏移地址：0x1000

复位值：0x0002 0101

总线模式寄存器为 DMA 建立总线工作模式。

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
Res.	Res.	Res.	Res.	Res.	MB	AAB	FPM	USP	RDP						FB	PM		PBL						EDFE	DSL					DA	SR
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rs

位 31:27 保留，必须保持复位值。

位 26 **MB**：混合突发 (Mixed burst)

该位置为高电平且 FB 位置为低电平时，AHB 主接口将以 INCR（未定义突发）启动所有长度大于 16 的突发传输。该位清零时，对于长度等于或小于 16 的突发，将恢复为固定突发传输（INCRx 和 SINGLE）。

位 25 **AAB**：地址对齐的节拍 (Address-aligned beats)

当该位设置为高电平并且 FB 位等于 1 时，AHB 接口会生成与起始地址 LS 位对齐的所有突发。如果 FB 位等于 0，则第一个突发（访问数据缓冲器的起始地址）不对齐，但后续的突发与地址对齐。

位 24 **FPM**：4xPBL 模式 (4xPBL mode)

当设置为高电平时，该位会将编程的 PBL 值（位 [22:17] 和位 [13:8]）乘以四倍。因此，DMA 根据 PBL 值以最大 4、8、16、32、64 和 128 个节拍传输数据。

位 23 **USP**：使用单独的 PBL (Use separate PBL)

设置为高电平时，它会配置 RxDMA，将位 [22:17] 中配置的值用作 PBL，而位 [13:8] 中的 PBL 值仅适用于 TxDMA 操作。当该位清零时，位 [13:8] 中的 PBL 值同时适用于两个 DMA 引擎。

位 22:17 **RDP**：Rx DMA PBL

这些位指示要在一个 RxDMA 事务中传输的最大节拍数。这是在单个块读/写操作中使用的最大值。RxDMA 每次在主机总线上开始突发传输时，始终尝试按 RDP 中指定的方式进行突发。允许使用值 1、2、4、8、16 和 32 对 RDP 进行编程。任何其它值都会产生未定义的行为。

仅在 USP 设置为高电平时这些位才有效且适用。

位 16 **FB**：固定突发 (Fixed burst)

该位控制 AHB 主接口是否执行固定突发传输。置 1 时，AHB 在正常突发传输开始期间仅使用 SINGLE、INCR4、INCR8 或 INCR16。复位时，AHB 使用 SINGLE 和 INCR 突发传输操作。

位 15:14 **PM**：Rx Tx 优先级比 (Rx Tx priority ratio)

RxDMA 请求优先于 TxDMA 请求，优先级比如下：

00: 1:1

01: 2:1

10: 3:1

11: 4:1

这仅在 DA 位清零时有效。

位 13:8 **PBL**：可编程突发长度 (Programmable burst length)

这些位指示要在一个 DMA 事务中传输的最大节拍数。这是在单个块读/写操作中使用的最大值。DMA 每次在主机总线上开始突发传输时，始终尝试按 PBL 中指定的方式进行突发。允许使用值 1、2、4、8、16 和 32 对 PBL 进行编程。任何其它值都会产生未定义的行为。当 USP 置 1 时，此 PBL 值仅适用于 TxDMA 事务。

PBL 值有以下限制：

– 可能的最大节拍数 (PBL) 受 Tx FIFO 和 Rx FIFO 大小的限制。
– FIFO 有一个限制，即支持的最大节拍数等于 FIFO 深度的一半。
– 如果 PBL 由发送和接收 DMA 共用，则必须考虑 Rx FIFO 和 Tx FIFO 的最小深度。
– 请不要编程超出范围的 PBL 值，因为系统可能发生异常行为。

位 7 **EDFE**：增强描述符格式使能 (Enhanced descriptor format enable)

该位置 1 时，使能增强描述符格式，并将描述符大小增加至 32 字节（8 个 DWORD）。如果已激活时间戳功能（ETH_PTPTSCR 位 0 TSE=1）或 IPv4 校验和减荷（ETH_MACCR 位 10 IPCO=1），则必须使用此增强描述符。

位 6:2 **DSL**：描述符跳过长度 (Descriptor skip length)

该位指定两个未链接描述符之间跳过的字数。地址从当前描述符结束处开始跳到下一个描述符起始处。当 DSL 值等于零时，在环形模式下，DMA 会将描述符表视为连续的。

位 1 **DA**：DMA 仲裁 (DMA Arbitration)

0：循环调度，Rx:Tx 优先级比在位 [15:14] 中给出

1：Rx 优先于 Tx

位 0 **SR**：软件复位 (Software reset)

当该位置 1 时，MAC DMA 控制器会复位所有 MAC 子系统的内部寄存器和逻辑。在所有内核时钟域完成复位操作后，该位自动清零。重新编程任何内核寄存器之前，在该位中读取 0 值。

以太网 DMA 发送轮询要求寄存器 (ETH_DMATPDR)

Ethernet DMA transmit poll demand register

偏移地址：0x1004

复位值：0x0000 0000

应用程序使用此寄存器来指示 DMA 轮询发送描述符列表。发送轮询要求寄存器使能发送 DMA 来检查当前描述符是否为 DMA 所有。如果 TxDMA 处于挂起模式，则发出发送轮询要求命令将其唤醒。如果发送帧中出现下溢错误或发送 DMA 所拥有的描述符不可用，则 TxDMA 会进入挂起模式。用户可以随时发出此命令，当该命令开始重新获取主机存储器的当前描述符后，TxDMA 即会对其进行复位。

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
TPD																															
rw_wt																															

位 31:0 **TPD**：发送轮询要求 (Transmit poll demand)

向这些位写入任何值时，DMA 都会读取 ETH_DMACHTDR 寄存器指向的当前描述符。如果该描述符不可用（由主机所有），则发送会返回到挂起状态，并将 ETH_DMASR 寄存器位 2 进行置位。如果该描述符可用，则发送会继续进行。

以太网 DMA 接收轮询要求寄存器 (ETH_DMARPDR)

EHERNET DMA receive poll demand register

偏移地址：0x1008

复位值：0x0000 0000

应用程序使用此寄存器来指示 DMA 轮询接收描述符列表。接收轮询要求寄存器会使能接收 DMA 来检查新描述符。此命令用于将 RxDMA 从挂起状态唤醒。仅当 RxDMA 所拥有的描述符不可用时，它才会进入挂起状态。

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
RPD																															
rw_wt																															

位 31:0 **RPD**：接收轮询要求 (Receive poll demand)

向这些位写入任何值时，DMA 都会读取 ETH_DMACHRDR 寄存器指向的当前描述符。如果该描述符不可用（由主机所有），则接收会返回到挂起状态，且不会将 ETH_DMASR 寄存器位 7 进行置位。如果该描述符可用，接收 DMA 将返回到活动状态。

以太网 DMA 接收描述符列表地址寄存器 (ETH_DMARDLAR)

Ethernet DMA receive descriptor list address register

偏移地址：0x100C

复位值：0x0000 0000

接收描述符列表地址寄存器会指向接收描述符列表的起始处。描述符列表位于 STM32F75xxx 和 STM32F74xxx 的物理存储器空间，且必须为字对齐。DMA 会将描述符列表的对应 LS 位置为低电平，进而在内部将其转换为总线宽度对齐地址。仅当接收停止时，才允许对 ETH_DMARDLAR 寄存器执行写操作。停止后，必须先对 ETH_DMARDLAR 寄存器执行写操作，然后才能发出接收启动命令。

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
SRL																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **SRL**：接收列表的起始处 (Start of receive list)

此字段包含接收描述符列表中的首个描述符的基址。DMA 会在内部将 LSB 位 [1/2/3:0]（对应于 32/64/128 位的总线宽度）忽略，并将其值均视为零。因此，这些 LSB 位为只读。

以太网 DMA 发送描述符列表地址寄存器 (ETH_DMATDLAR)

Ethernet DMA transmit descriptor list address register

偏移地址：0x1010

复位值：0x0000 0000

发送描述符列表地址寄存器会指向发送描述符列表的起始处。描述符列表位于 STM32F75xxx 和 STM32F74xxx 的物理存储器空间，且必须为字对齐。DMA 会将描述符列表的对应 LSB 位置为低电平，进而在内部将其转换为总线宽度对齐地址。仅当发送停止时，才允许对 ETH_DMATDLAR 寄存器执行写操作。发送停止后，可以先对 ETH_DMATDLAR 寄存器执行写操作，然后再发出发送启动命令。

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
STL																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **STL**：发送列表的起始处 (Start of transmit list)

此字段包含发送描述符列表中的首个描述符的基址。DMA 会在内部将 LSB 位 [1/2/3:0]（对应于 32/64/128 位的总线宽度）忽略，并将其值均视为零。因此，这些 LSB 位为只读。

以太网 DMA 状态寄存器 (ETH_DMASR)

Ethernet DMA status register

偏移地址：0x1014

复位值：0x0000 0000

状态寄存器包含所有 DMA 向应用程序报告的状态位。软件驱动程序通常在中断服务例程或轮询期间读取 ETH_DMASR 寄存器。此寄存器中的大部分字段会导致主机中断。读取时 ETH_DMASR 寄存器位不会清零。向 ETH_DMASR 寄存器 [16:0] 中的（未保留）位写入 1 会将其清零，写入 0 则不起作用。通过屏蔽 ETH_DMAIER 寄存器中的位，可以屏蔽各个对应字段（位 [16:0]）。

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
Res.	Res.	TSTS	PMTS	MMCS	Res.	EBS			TPS			RPS			NIS	AIS	ERS	FBES	Res.	Res.	ETS	RWTS	RPSS	RBUS	RS	TUS	ROS	TJTS	TBUS	TPSS	TS
		r	r	r		r	r	r	r	r	r	r	r	r	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 31:30 保留，必须保持复位值。

位 29 **TSTS**：时间戳触发状态 (Time stamp trigger status)

此位指示 MAC 内核的时间戳发生器块中发生的中断事件。软件必须读取 MAC 内核的状态寄存器，将其源（位 9）清零，才能将此位复位为 0。当此位处于高电平时，使能后会产生中断。

位 28 **PMTS**：PMT 状态 (PMT status)

此位指示 MAC 内核的 PMT 中发生的事件。软件必须读取 MAC 内核中对应的寄存器，以获取产生中断的具体原因，然后将其源清零，才能将此位复位为 0。当此位处于高电平时，使能后会产生中断。

位 27 **MMCS**：MMC 状态 (MMC status)

此位反映 MAC 内核的 MMC 中发生的事件。软件必须读取 MAC 内核中对应的寄存器，以获取产生中断的具体原因，然后将中断源清零，才能将此位置 0。当此位处于高电平时，使能后会产生中断。

位 26 保留，必须保持复位值。

位 25:23 **EBS**：错误位状态 (Error bits status)

这些位指示导致总线错误（AHB 接口上的错误响应）的错误类型。仅在致命总线错误位（ETH_DMASR 寄存器 [13]）置 1 时有效。此字段不会产生中断。

位 231 TxDMA 传输数据时出错 (Error during data transfer by TxDMA)

0 RxDMA 传输数据时出错 (Error during data transfer by RxDMA)

位 24 1 读取传输时出错 (Error during read transfer)

0 写入传输时出错 (Error during write transfer)

位 25 1 访问描述符时出错 (Error during descriptor access)

0 访问数据缓冲区时出错 (Error during data buffer access)

位 22:20 **TPS**：发送过程状态 (Transmit process state)

这些位指示发送 DMA FSM 的状态。此字段不会产生中断。

000：停止；发出复位或停止发送命令

001：运行中；正在获取发送传输描述符

010：运行中；正在等待状态

011：运行中；正在读取主机存储器缓冲区中的数据并将其加入发送缓冲区 (Tx FIFO) 队列

100 和 101：保留供将来使用

110：挂起；发送描述符不可用或发送缓冲区下溢

111：运行中；正在关闭发送描述符

位 19:17 **RPS**：接收过程状态 (Receive process state)

这些位指示接收 DMA FSM 的状态。此字段不会产生中断。

000：停止：发出复位或停止接收命令

001：运行中：正在获取接收传输描述符

010：保留供将来使用

011：运行中：正在等待接收数据包

100：已挂起：接收描述符不可用

101：运行中：正在关闭接收描述符

110：保留供将来使用

111：运行中：将接收数据包的数据从接收缓冲区传输到主机存储器

位 16 **NIS**：正常中断汇总 (Normal interrupt summary)

当使能 ETH_DMAIER 寄存器中对应的中断位时，正常中断汇总位的值是以下位的逻辑或运算结果：

– ETH_DMASR [0]：发送中断 (Transmit interrupt)
– ETH_DMASR [2]：发送缓冲区不可用 (Transmit buffer unavailable)
– ETH_DMASR [6]：接收中断 (Receive interrupt)
– ETH_DMASR [14]：提前接收中断 (Early receive interrupt)

只有未屏蔽的位会影响正常中断汇总位。

它是黏着位，每当导致 NIS 位置 1 的对应位被清零时，必须同时将它也清零（通过向此位写入 1）。

位 15 **AIS**：异常中断汇总 (Abnormal interrupt summary)

当使能 ETH_DMAIER 寄存器中对应的中断位时，异常中断汇总位的值是以下位的逻辑或运算结果：

– ETH_DMASR [1]：发送过程停止 (Transmit process stopped)
– ETH_DMASR [3]：发送 jabber 超时 (Transmit jabber timeout)
– ETH_DMASR [4]：接收 FIFO 上溢 (Receive FIFO overflow)
– ETH_DMASR [5]：发送下溢 (Transmit underflow)
– ETH_DMASR [7]：接收缓冲区不可用 (Receive buffer unavailable)
– ETH_DMASR [8]：接收过程停止 (Receive process stopped)
– ETH_DMASR [9]：接收看门狗超时 (Receive watchdog timeout)
– ETH_DMASR [10]：提前发送中断 (Early transmit interrupt)
– ETH_DMASR [13]：致命总线错误 (Fatal bus error)

只有未屏蔽的位会影响异常中断汇总位。

它是黏着位，每当导致 AIS 位置 1 的对应位被清零时，必须同时将它也清零。

位 14 **ERS**：提前接收状态 (Early receive status)

此位指示 DMA 已填满数据包的首个数据缓冲区。接收中断 ETH_DMASR [6] 会自动将此位清零。

位 13 **FBES**：致命总线错误状态 (Fatal bus error status)
此位指示发生了总线错误，具体信息参见位 [25:23]。此位置 1 后，对应的 DMA 引擎会禁止其所有的总线访问。

位 12:11 保留，必须保持复位值。

位 10 **ETS**：提前发送状态 (Early transmit status)
此位指示要发送的帧已完全传输到发送 FIFO。

位 9 **RWTS**：接收看门狗超时状态 (Receive watchdog timeout status)
当接收到的帧的长度大于 2048 个字节时，会对此位进行置位。

位 8 **RPSS**：接收过程停止状态 (Receive process stopped status)
当接收过程进入停止状态时，会对此位进行置位。

位 7 **RBUS**：接收缓冲区不可用状态 (Receive buffer unavailable status)
此位指示接收列表中的下一个描述符由主机所拥有，DMA 无法获取。接收过程进入挂起状态。要恢复处理接收描述符，主机应更改描述符的拥有关系，然后发出接收轮询要求命令。如果未发出接收轮询要求命令，则当接收到下一个识别的传入帧时，接收过程会恢复。仅当上一接收描述符由 DMA 所拥有时，才能将 ETH_DMASR [7] 置 1。

位 6 **RS**：接收状态 (Receive status)
此位指示帧接收已完成。特定的帧状态信息已发布在描述符中。接收保持运行状态。

位 5 **TUS**：发送下溢状态 (Transmit underflow status)
此位指示在帧发送期间，发送缓冲区发生下溢。发送会进入挂起状态，且下溢错误 TDES0[1] 置 1。

位 4 **ROS**：接收上溢状态 (Receive overflow status)
此位指示在帧接收期间，接收缓冲区发生上溢。如果部分帧已传输到应用程序，则 RDES0[11] 中的上溢状态位置 1。

位 3 **TJTS**：发送 jabber 超时状态 (Transmit jabber timeout status)
此位指示发送 jabber 定时器已过期，这意味着发送器过度有效。发送过程会中止并将其置于停止状态。这会导致对发送 jabber 超时 TDES0[14] 标志进行置位。

位 2 **TBUS**：发送缓冲区不可用状态 (Transmit buffer unavailable status)
此位指示发送列表中的下一个描述符由主机所拥有，DMA 无法获取。发送会进入挂起状态。位 [22:20] 用于解释发送过程状态转换。要恢复处理发送描述符，主机应更改描述符位的拥有关系，然后发出发送轮询要求命令。

位 1 **TPSS**：发送过程停止状态 (Transmit process stopped status)
当发送停止时，此位置 1。

位 0 **TS**：发送状态 (Transmit status)
此位指示帧发送已完成，且首个描述符中的 TDES1[31] 位已置 1。

以太网 DMA 工作模式寄存器 (ETH_DMAOMR)

Ethernet DMA operation mode register

偏移地址：0x1018

复位值：0x0000 0000

工作模式寄存器将建立发送和接收工作模式及命令。作为 DMA 初始化过程的一部分，应将 ETH_DMAOMR 寄存器作为最后的 CSR 进行写操作。

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
Res.	Res.	Res.	Res.	Res.	DTCEFD	RSF	DFRF	Res.	Res.	TSF	FTF	Res.	Res.	Res.	TTC			ST	Res.	Res.	Res.	Res.	Res.	FEF	FUGF	Res.	RTC		OSF	SR	Res.
					rw	rw	rw			rw	rs				rw	rw	rw	rw						rw	rw		rw	rw	rw	rw	

位 31:27 保留，必须保持复位值。

位 26 **DTCEFD**：禁止丢弃 TCP/IP 校验和错误帧 (Dropping of TCP/IP checksum error frames disable)

当此位置 1 时，如果帧中仅存在由接收校验和减荷引擎检测出来的错误，则内核不会丢弃它。这类帧在 MAC 接收到的以太网帧中没有任何错误（包括 FCS 错误），而仅在封装的有效负载中有错误。当此位清零时，如果 FEF 位进行了复位，则会丢弃所有错误帧。

位 25 **RSF**：接收存储并转发 (Receive store and forward)

当此位置 1 时，向 Rx FIFO 写入完整帧后，可从中读取一个帧，同时忽略 RTC 位。当此位清零时，Rx FIFO 在直通模式下工作，具体取决于 RTC 位指定的阈值。

位 24 **DFRF**：禁止刷新接收帧 (Disable flushing of received frames)

当此位置 1 时，RxDMA 不会因为接收描述符/缓冲区不可用而刷新任何帧（通常情况下，当此位清零时它会这样做）。（见*第 1399 页上的接收过程挂起*）

位 23:22 保留，必须保持复位值。

位 21 **TSF**：发送存储并转发 (Transmit store and forward)

当此位置 1 时，如果发送 FIFO 中有一个完整帧，则发送会启动。当此位置 1 时，会忽略由 ETH_DMAOMR 寄存器位 [16:14] 指定的 TTC 值。

该位清零后，ETH_DMAOMR 寄存器位 [16:14] 指定的 TTC 值才会有效。

该位只有在已停止传输时才能更改。

位 20 **FTF**：刷新发送 FIFO (Flush transmit FIFO)

该位置 1 时，发送 FIFO 控制器逻辑会复位为默认值，因此，Tx FIFO 中的所有数据均会丢失/刷新。刷新操作结束时该位在内部清零。此位清零之前不得对工作模式寄存器执行写操作。

位 19:17 保留，必须保持复位值。

位 16:14 **TTC**：发送阈值控制 (Transmit threshold control)

这三个位用于控制发送 FIFO 的阈值级别。当发送 FIFO 中的帧大小大于阈值时启动发送。此外，还会发送长度小于阈值的全帧。这些位只有在 TSF 位（位 21）清零后才能使用。

000: 64
001: 128
010: 192
011: 256
100: 40
101: 32
110: 24
111: 16

位 13 **ST**：启动/停止发送 (Start/stop transmission)

该位置 1 时，发送过程会进入运行状态，DMA 会检查当前位置的发送列表查找待发送的帧。可通过发送列表中的当前位置，即由 ETH_DMATDLAR 寄存器设定的基址，或者上一次停止发送时的保留位置尝试获取描述符。如果当前描述符不属于 DMA，则发送过程会进入挂起状态，且发送缓冲区不可用位 (ETH_DMASR [2]) 置 1。启动发送命令只有在传输已停止时才有效。如果该命令在设置 DMA ETH_DMATDLAR 寄存器之前发出，则 DMA 行为无法预知。

该位清零时，发送过程会在完成发送当前帧的任务之后进入停止状态， 并保存发送列表中的下一个描述符位置，该位置在重启发送后会成为当前位置。停止发送命令只有在当前帧发送过程结束或发送过程处于挂起状态时才有效。

位 12:8 保留，必须保持复位值。

位 7 **FEF**：转发错误帧 (Forward error frame)

该位置 1 时，除短错误帧之外的所有帧都会转发到 DMA。

该位清零时，Rx FIFO 会丢弃带有错误状态（CRC 错误、冲突错误、巨帧、看门狗超时、上溢）的帧。不过，如果某个帧的起始字节（写）指针已传输到读控制器端（阈值模式下），则不会丢弃该帧。如果 ARI 总线上未传输（输出）帧的起始字节，则 Rx FIFO 会丢弃此错误帧。

位 6 **FUGF**：转发过小的好帧 (Forward undersized good frame)

该位置 1 时，Rx FIFO 会转发包含 pad 字节和 CRC 的过小帧（无错误但长度不足 64 字节的帧）。

该位清零时，Rx FIFO 会丢弃所有不足 64 字节的帧，除非因接收阈值下限更低（例如 RTC = 01）而导致此类帧已传输。

位 5 保留，必须保持复位值。

位 4:3 **RTC**：接收阈值控制 (Receive threshold control)

这两个位用于控制接收 FIFO 的阈值级别。当接收 FIFO 中的帧大小大于阈值时启动 DMA 传输（请求）。此外，长度小于阈值的全帧会自动传输。

注： 如果配置的接收 FIFO 大小为 128 字节，则不适合使用 11 作为阈值。

注： 这些位只有在 RSF 位为 0 时才有效，而当 RSF 位置 1 后会忽略这些位。

00: 64
01: 32
10: 96
11: 128

位 2 **OSF**：处理第二个帧 (Operate on second frame)

该位置 1 时会命令 DMA 处理第二个发送数据帧，即使尚未获得首个帧的状态。

位 1 **SR**：启动/停止接收 (Start/stop receive)

该位置 1 时，接收过程会进入运行状态。DMA 尝试从接收列表中获取描述符并处理传入帧。可通过列表中的当前位置，由 DMA ETH_DMARDLAR 寄存器设定的地址，或者上一次停止接收时的保留位置尝试获取描述符。如果 DMA 未占据任何描述符，则接收过程会进入挂起状态，且接收缓冲区不可用位 (ETH_DMASR [7]) 置 1。启动接收命令只有在已停止接收时才有效。如果该命令在设置 DMA ETH_DMARDLAR 寄存器之前发出，则 DMA 行为无法预知。

该位清零时，RxDMA 会在传输当前帧之后停止操作， 并保存接收列表中的下一个描述符位置，该位置在接收过程重启后会成为当前位置。停止接收命令只有在当前接收过程处于运行（等待接收数据包）或挂起状态时才有效。

位 0 保留，必须保持复位值。

以太网 DMA 中断使能寄存器 (ETH_DMAIER)

Ethernet DMA interrupt enable register

偏移地址：0x101C

复位值：0x0000 0000

中断使能寄存器可使能 ETH_DMASR 报告的中断。将一个位置 1 可使能相应的中断。完成硬件或软件复位之后，会禁止所有中断。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NISE	AISE	ERIE	FBEIE	Res.	Res.	ETIE	RWTIE	RPSIE	RBUIE	RIE	TUIE	ROIE	TJTIE	TBUIE	TPSIE	TIE
															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 **NISE**：正常中断汇总使能 (Normal interrupt summary enable)

该位置 1 时，会使能一个正常中断。该位清零时，会禁止一个正常中断。该位可使能以下位：

– ETH_DMASR [0]：发送中断 (Transmit interrupt)
– ETH_DMASR [2]：发送缓冲区不可用 (Transmit buffer unavailable)
– ETH_DMASR [6]：接收中断 (Receive interrupt)
– ETH_DMASR [14]：提前接收中断 (Early receive interrupt)

位 15 **AISE**：异常中断汇总使能 (Abnormal interrupt summary enable)

该位置 1 时，会使能一个异常中断。该位清零时，会禁止一个异常中断。该位可使能以下位：

– ETH_DMASR [1]：发送过程停止 (Transmit process stopped)
– ETH_DMASR [3]：发送 jabber 超时 (Transmit jabber timeout)
– ETH_DMASR [4]：接收上溢 (Receive overflow)
– ETH_DMASR [5]：发送下溢 (Transmit underflow)
– ETH_DMASR [7]：接收缓冲区不可用 (Receive buffer unavailable)
– ETH_DMASR [8]：接收过程停止 (Receive process stopped)
– ETH_DMASR [9]：接收看门狗超时 (Receive watchdog timeout)
– ETH_DMASR [10]：提前发送中断 (Early transmit interrupt)
– ETH_DMASR [13]：致命总线错误 (Fatal bus error)

位 14 **ERIE**：提前接收中断使能 (Early receive interrupt enable)

当该位通过正常中断汇总使能位（ETH_DMAIER 寄存器 [16]）置 1 时，可使能提前接收中断。

该位清零时，会禁止提前接收中断。

位 13 **FBEIE**：致命总线错误中断使能 (Fatal bus error interrupt enable)

当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能致命总线错误中断。

该位清零时，会禁止致命总线使能中断。

位 12:11 保留，必须保持复位值。

位 10 **ETIE**：提前发送中断使能 (Early transmit interrupt enable)

当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能提前发送中断。

该位清零时，会禁止提前发送中断。

位 9 **RWTIE**：接收看门狗超时中断使能 (receive watchdog timeout interrupt enable)
当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能接收看门狗超时中断。
该位清零时，会禁止接收看门狗超时中断。

位 8 **RPSIE**：接收过程停止中断使能 (Receive process stopped interrupt enable)
当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能接收停止中断。该位清零时，会禁止接收停止中断。

位 7 **RBUIE**：接收缓冲区不可用中断使能 (Receive buffer unavailable interrupt enable)
当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能接收缓冲区不可用中断。
该位清零时，会禁止接收缓冲区不可用中断。

位 6 **RIE**：接收中断使能 (Receive interrupt enable)
当该位通过正常中断汇总使能位（ETH_DMAIER 寄存器 [16]）置 1 时，可使能接收中断。
该位清零时，会禁止接收中断。

位 5 **TUIE**：下溢中断使能 (Underflow interrupt enable)
当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能发送下溢中断。
该位清零时，会禁止下溢中断。

位 4 **ROIE**：上溢中断使能 (Overflow interrupt enable)
当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能接收上溢中断。
该位清零时，会禁止上溢中断。

位 3 **TJTIE**：发送 jabber 超时中断使能 (Transmit jabber timeout interrupt enable)
当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能发送 jabber 超时中断。
该位清零时，会禁止发送 jabber 超时中断。

位 2 **TBUIE**：发送缓冲区不可用中断使能 (Transmit buffer unavailable interrupt enable)
当该位通过正常中断汇总使能位（ETH_DMAIER 寄存器 [16]）置 1 时，可使能发送缓冲区不可用中断。
该位清零时，会禁止发送缓冲区不可用中断。

位 1 **TPSIE**：发送过程停止中断使能 (Transmit process stopped interrupt enable)
当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能发送停止中断。
该位清零时，会禁止发送停止中断。

位 0 **TIE**：发送中断使能 (Transmit interrupt enable)
当该位通过正常中断汇总使能位（ETH_DMAIER 寄存器 [16]）置 1 时，可使能发送中断。
该位清零时，会禁止发送中断。

只有 DMA 状态寄存器的 TSTS 或 PMTS 位已置位但相应的中断均未带标记时，或者 NIS/AIS 状态位已置位并且相应的中断使能位 (NISE/AISE) 已使能时，才会生成以太网中断。

以太网 DMA 丢失帧和缓冲区上溢计数器寄存器 (ETH_DMAMFBOCR)

Ethernet DMA missed frame and buffer overflow counter register

偏移地址：0x1020

复位值：0x0000 0000

DMA 可维护两个计数器在接收过程中对丢失帧数目进行计数。该寄存器会报告计数器的当前值。计数器用于进行诊断。位 [15:0] 指示因 STM32F75xxx 和 STM32F74xxx 缓冲区不可用（无可用的接收描述符）而丢失的帧。位 [27:17] 指示因 Rx FIFO 存在上溢情况以及帧长度过短（不足 64 字节的好帧）而丢失的帧。

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
Res.	Res.	Res.	OFOC	MFA											OMFC	MFC															
			rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r

位 31:29 保留，必须保持复位值。

位 28 **OFOC**：FIFO 上溢计数器的上溢位 (Overflow bit for FIFO overflow counter)

位 27:17 **MFA**：应用程序所丢失的帧 (Missed frames by the application)
指示应用程序所丢失的帧数

位 16 **OMFC**：丢失帧计数器的上溢位 (Overflow bit for missed frame counter)

位 15:0 **MFC**：控制器所丢失的帧 (Missed frames by the controller)
指示控制器因主机接收缓冲区不可用而丢失的帧数。DMA 每丢弃一个传入帧，此计数器值加 1。

以太网 DMA 接收状态看门狗定时器寄存器 (ETH_DMARSWTR)

Ethernet DMA receive status watchdog timer register

偏移地址：0x1024

复位值：0x0000 0000

向该寄存器写入一个非零值时，可使能针对接收状态 (RS, ETH_DMASR[6]) 的看门狗定时器。

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSWTC							
																								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:0 **RSWTC**：接收状态 (RS) 看门狗定时器计数 (Receive status (RS) watchdog timer count)
指示 HCLK 时钟周期数乘以 256 之后的时间值，即看门狗定时器的设定时间值。看门狗定时器会在 RxDMA 结束帧传输之后由编程设定的值触发，但相应的 RS 状态位因相应描述符中的 RDES1[31] 已置 1 而未置 1。当看门狗定时器计时结束时，RS 位置 1 且定时器停止。由于 RS 根据每个接收帧的 RDES1[31] 进行自动设置而使 RS 位设为高电平时，看门狗定时器复位。

以太网 DMA 当前主机发送描述符寄存器 (ETH_DMACHTDR)

Ethernet DMA current host transmit descriptor register

偏移地址：0x1048

复位值：0x0000 0000

当前主机发送描述符寄存器会指向 DMA 所读取的当前发送描述符的起始地址。

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
HTDAP																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **HTDAP**：主机发送描述符地址指针 (Host transmit descriptor address pointer)
已清零。该指针在运行期间由 DMA 更新。

以太网 DMA 当前主机接收描述符寄存器 (ETH_DMACHRDR)

Ethernet DMA current host receive descriptor register

偏移地址：0x104C

复位值：0x0000 0000

当前主机接收描述符寄存器会指向 DMA 所读取的当前接收描述符的起始地址。

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
HRDAP																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **HRDAP**：主机接收描述符地址指针 (Host receive descriptor address pointer)
复位时清零。该指针在运行期间由 DMA 更新。

以太网 DMA 当前主机发送缓冲区地址寄存器 (ETH_DMACHTBAR)

Ethernet DMA current host transmit buffer address register

偏移地址：0x1050

复位值：0x0000 0000

当前主机发送缓冲区地址寄存器会指向 DMA 所读取的当前发送缓冲区地址。

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
HTBAP																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **HTBAP**：主机发送缓冲区地址指针 (Host transmit buffer address pointer)
复位时清零。该指针在运行期间由 DMA 更新。

以太网 DMA 当前主机接收缓冲区地址寄存器 (ETH_DMACHRBAR)

Ethernet DMA current host receive buffer address register

偏移地址：0x1054

复位值：0x0000 0000

当前主机接收缓冲区地址寄存器会指向 DMA 所读取的当前接收缓冲区地址。

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
HRBAP																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **HRBAP**：主机接收缓冲区地址指针 (Host receive buffer address pointer)
复位时清零。该指针在运行期间由 DMA 更新。

38.8.5 以太网寄存器映射

表 245 给出了 ETH 寄存器映射和复位值。

表 245. 以太网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	ETH_MACCR	Res.	Res.	Res.	Res.	Res.	Res.	CSTF	Res.	WD	JD	Res.	Res.	IFG			CSD	Res.	FES	ROD	LM	DM	IPCO	RD	Res.	APCS	BL		DC	TE	RE	Res.	Res.
	Reset value							0		0	0			0	0	0	0		0	0	0	0	0	0		0	0	0	0	0	0		
0x04	ETH_MACFFR	RA	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HPF	SAF		PCF		BFD	PAM	DAIF	HM	HU	PM
	Reset value	0																					0	0	0	0	0	0	0	0	0	0	0
0x08	ETH_MACHTHR	HTH[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
0x0C	ETH_MACHTLR	HTL[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	ETH_MACMIIAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PA					MR					CR				MW	MB
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	ETH_MACMIIDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MD															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	ETH_MACFCR	PT																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ZQPD	Res.	PLT		UPFD	RFCE	TFCE	FCB/BPA
	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
0x1C	ETH_MACVLANTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VLANTC	VLANTI															
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	ETH_MACRWUFFR	Frame filter reg0\Frame filter reg1\Frame filter reg2\Frame filter reg3\Frame filter reg4\...\Frame filter reg7																															
	Reset value	0																															
0x2C	ETH_MACPMTCSR	WFFRPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GU	Res.	Res.	WFR	MPR	Res.	Res.	WFE	MPE	PD
	Reset value	0																						0			0	0			0	0	0

表 245. 以太网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
0x34	ETH_MACDBGR	Res.	Res.	Res.	Res.	Res.	Res.	TFF	TFNEGU	Res.	TFWA	TFRS	MTP	MTFCS		MMTEA		Res.	Res.	Res.	Res.	Res.	Res.	RFFL		Res.	RFRCS		RFWRA	Res.	MSFRWCS		MMRPEA
	Reset value							0	0		0	0	0	0	0	0	0							0	0		0	0	0		0	0	0
0x38	ETH_MACCR																	Res.	Res.	Res.	Res.	Res.	Res.	TSTS	Res.	Res.	MMCTS	MMCRS	MMCS	PMTS	Res.	Res.	Res.
	Reset value																							0			0	0	0	0			
0x3C	ETH_MACIMR																	Res.	Res.	Res.	Res.	Res.	Res.	TSTIM	Res.	Res.	Res.	Res.	Res.	PMTIM	Res.	Res.	Res.
	Reset value																							0						0			
0x40	ETH_MACA0HR	MO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MACA0H															
	Reset value	1	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
0x44	ETH_MACA0LR	MACA0L																															
	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
0x48	ETH_MACA1HR	AE	SA	MBC[6:0]						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MACA1H															
	Reset value	0	0	0	0	0	0	0	0									1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x4C	ETH_MACA1LR	MACA1L																															
	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
0x50	ETH_MACA2HR	AE	SA	MBC						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MACA2H															
	Reset value	0	0	0	0	0	0	0	0									1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x54	ETH_MACA2LR	MACA2L																															
	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
0x58	ETH_MACA3HR	AE	SA	MBC						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MACA3H															
	Reset value	0	0	0	0	0	0	0	0									1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x5C	ETH_MACA3LR	MACA3L																															
	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
0x100	ETH_MMCCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MCFHP	MCP	MCF	ROR	CSR	CR
	Reset value																											0	0	0	0	0	0
0x104	ETH_MMCRIR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RGUFS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RFAES	RFCES	Res.	Res.	Res.	Res.	Res.
	Reset value															0											0	0					
0x108	ETH_MMCTIR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TGFS	Res.	Res.	Res.	Res.	Res.	TGFMSCS	TGFSCS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value											0						0	0														
0x10C	ETH_MMCRIMR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RGUFM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RFAEM	RFCEM	Res.	Res.	Res.	Res.	Res.
	Reset value															0											0	0					
0x110	ETH_MMCTIMR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TGFM	Res.	Res.	Res.	Res.	Res.	TGFMSCM	TGFSCM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value											0						0	0														
0x14C	ETH_MMCTGFSCCR	TGFSCC																															
	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 245. 以太网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
0x150	ETH_MMCTGFMSCCR	TGFMSCC																															
	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
0x168	ETH_MMCTGFCR	TGFC																															
	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
0x194	ETH_MMCRFCECR	RFCEC																															
	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	ETH_MMCRFAECR	RFAEC																															
	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
0x1C4	ETH_MMCRGUFCR	RGUFC																															
	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
0x700	ETH_PTPTSCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSPFFMAE	TSCNT		TSSMRME	TSSEME	TSSIPV4FE	TSSIPV6FE	TSSPTPOEFE	TSPTPPSV2E	TSSSR	TSSARFE	Res.	Res.	TTSARU	TSITE	TSSTU	TSSTI	TSFCU	TSE
	Reset value														0	0	0	0	0	1	0	0	0	0	0			0	0	0	0	0	0
0x704	ETH_PTPSSIR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	STSSI							
	Reset value																									0	0	0	0	0	0	0	0
0x708	ETH_PTPTSHR	STS[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
0x70C	ETH_PTPTSLR	STPNS	STSS																														
	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
0x710	ETH_PTPTSHUR	TSUS																															
	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
0x714	ETH_PTPTSLUR	TSUPNS	TSUSS																														
	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
0x718	ETH_PTPTSAR	TSA																															
	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
0x71C	ETH_PTPTTHR	TTSH																															
	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
0x720	ETH_PTPTTLR	TTSL																															
	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
0x728	ETH_PTPTSSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSTTR	TSSO
	Reset value																															0	0
0x1000	ETH_DMABMR	Res.	Res.	Res.	Res.	Res.	MB	AAB	FPM	USP	RDP						FB	PM		PBL						EDFE	DSL					DA	SR
	Reset value						0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
0x1004	ETH_DMATPDR	TPD																															
	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
0x1008	ETH_DMARPDR	RPD																															
	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 245. 以太网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
0x100C	ETH_DMARDLAR	SRL																															
	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
0x1010	ETH_DMATDLAR	STL																															
	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
0x1014	ETH_DMASR	Res.	Res.	TSTS	PMTS	MMCS	Res.	EBS			TPS			RPS			NIS	AIS	ERS	FBES	Res.	Res.	ETS	RWTS	RPSS	RBUS	RS	TUS	ROS	TJTS	TBUS	TPSS	TS
	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
0x1018	ETH_DMAOMR	Res.	Res.	Res.	Res.	Res.	DTCEFD	RSF	DFRF	Res.	Res.	TSF	FTF	Res.	Res.	Res.	TTC			ST	Res.	Res.	Res.	Res.	Res.	FEF	FUGF	Res.	RTC		OSF	SR	Res.
	Reset value						0	0	0			0	0				0	0	0	0						0	0		0	0	0	0	
0x101C	ETH_DMAIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NISE	AISE	ERIE	FBEIE	Res.	Res.	ETIE	RWTIE	RPSIE	RBUIE	RIE	TUIE	ROIE	TJTIE	TBUIE	TPSIE	TIE
	Reset value																0	0	0	0			0	0	0	0	0	0	0	0	0	0	0
0x1020	ETH_DMAMFBOCR	Res.	Res.	Res.	OFOC	MFA											OMFC	MFC															
	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
0x1024	ETH_DMARSWTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSWTC							
	Reset value																									0	0	0	0	0	0	0	0
0x1048	ETH_DMACHTDR	HTDAP																															
	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
0x104C	ETH_DMACHRDR	HRDAP																															
	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
0x1050	ETH_DMACHTBAR	HTBAP																															
	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
0x1054	ETH_DMACHRBAR	HRBAP																															
	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 64 页上的第 2.2.2 节*。

39 HDMI-CEC 控制器 (HDMI-CEC)

39.1 前言

消费性电子产品控制 (CEC) 是 HDMI（高清多媒体接口）标准的一部分，作为该标准的*附录 1*。

它包含在用户环境下为所有不同的视听产品提供高级控制功能的协议。工作速度低并且处理器和存储器的开销很少。

HDMI-CEC 控制器为此协议提供硬件支持。

39.2 HDMI-CEC 控制器主要特性

- 符合 HDMI-CEC v1.4 规范
- 32 kHz CEC 内核和 2 个时钟源选项
  - 具有固定预分频比 (HSI/488) 的 HSI RC 振荡器
  - LSE 振荡器
- 针对超低功耗应用可在停止模式下工作
- 开始传输前的空闲信号时间可配置
  - 通过硬件配置时，根据 CEC 状态和发送历史自动完成
  - 通过软件配置时，设置为固定值（7 个时序选项）
- 可配置外设地址 (OAR)
- 支持监听模式
  - 在不干涉 CEC 线路的情况下，能接收发送到 OAR 以外的目标地址的 CEC 消息
- 可配置的接收容差裕量
  - 标准容差
  - 扩展容差
- 接收错误检测
  - 位上升错误 (BRE)，可选择停止接收 (BRESTP)
  - 短位周期错误 (SBPE)
  - 长位周期错误 (LBPE)
- 可配置错误位生成
  - BRE 检测时 (BREGEN)
  - LBPE 检测时 (BREGEN)
  - 始终在 SBPE 检测时生成
- 发送错误检测 (TXERR)
- 仲裁丢失检测 (ARBLST)
  - 自动发送重试
- 发送下溢检测 (TXUDR)
- 接收上溢检测 (RXOVR)

39.3 HDMI-CEC 功能说明

39.3.1 HDMI-CEC 引脚

CEC 总线是一根双向线，通过它实现数据的输入输出。通过 27 kΩ 上拉电阻连接到 +3.3 V 电源电压。器件的输出级必须为漏极开路或集电极开路，以便进行线“与”连接。

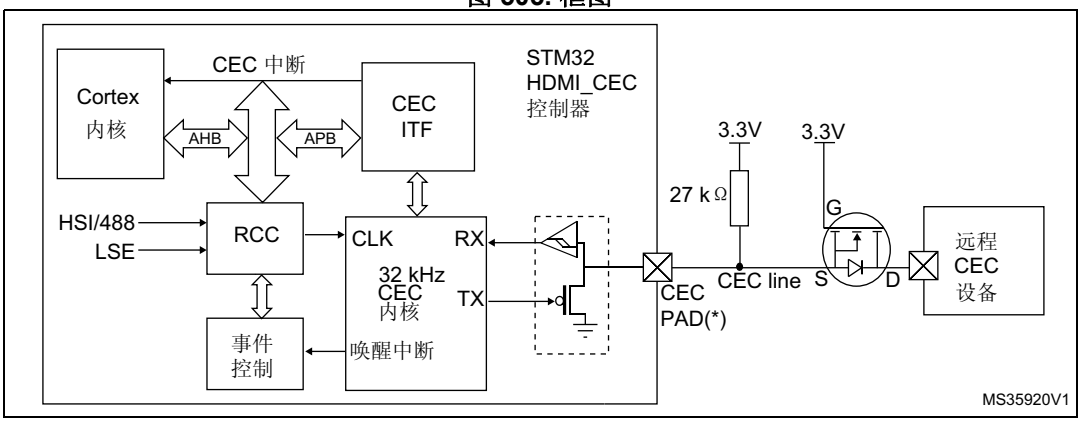
HDMI-CEC 控制器将 CEC 双向线作为标准 GPIO 的复用功能来管理，假设其配置为复用功能开漏输出。必须从外部为 STM32 添加 27 kΩ 上拉电阻。

要想在移除应用电源时不影响 CEC 总线，必须在此类情况下将 CEC 引脚与总线隔离。可通过 MOS 晶体管来实现此操作，如*图 505* 所示。

表 246. HDMI 引脚

名称	信号类型	备注
CEC	双向	两种状态： 1 = 高阻抗 0 = 低阻抗 必须从外部添加 27 kΩ 电阻。

图 505. 框图



1. GPIO 配置为输出漏极开路复用功能。
2. 配置为输出漏极开路复用功能时，施密特触发器仍为激活状态。

39.3.2 消息说明

CEC 线上的所有活动均包含一个发起方和一个或多个接收方。发起方负责发送消息和数据。跟随方为任意数据的接收方，负责设置任意应答位。

消息以单个帧的形式传送，帧中包含一个起始位、后跟一个报头块、一个可选的操作码和多个操作数。

这些块均由 8 位有效负载组成，首先发送最高有效位，接着发送消息结束 (EOM) 位和应答 (ACK) 位。

EOM 位在消息的最后一个块中置 1，在其它所有块中保持复位。如果指示 EOM 后消息中仍包含额外的块，则应忽略这些额外的块。可在报头块中将 EOM 位置 1 以向其它器件发送“ping”，确保这些器件已激活。

应答位始终由发起方设置为高阻态，这样一来，应答位便可由地址被包含在报头段的接收方或需要拒绝广播消息的接收方驱动为低电平。

报头包含源逻辑地址字段和目标逻辑地址字段。请注意，特定地址 0xF 用于广播消息。

图 506. 消息结构

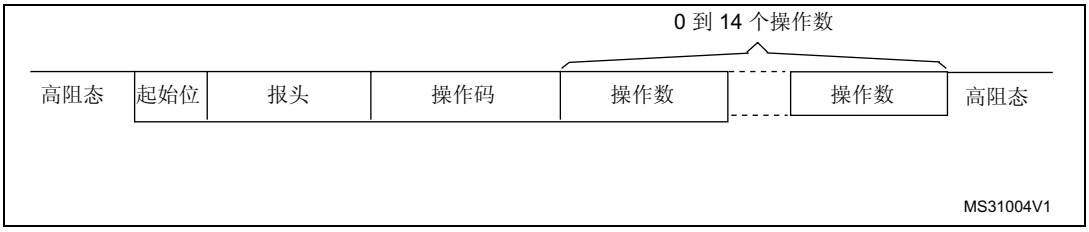
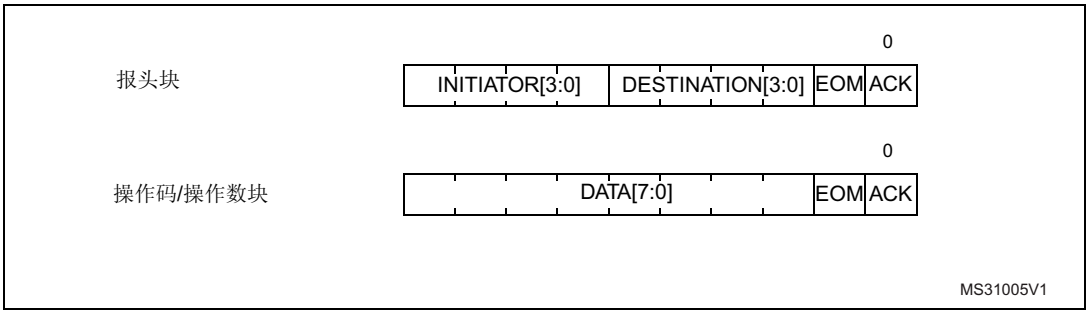


图 507. 块

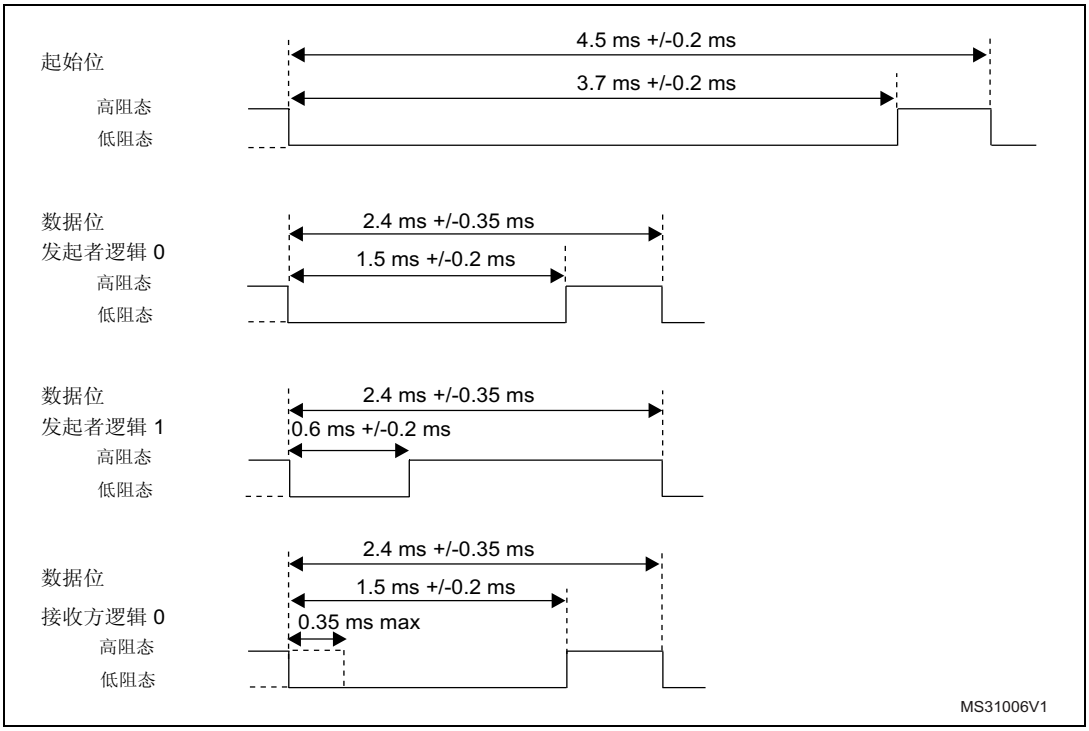


39.3.3 位时序

起始位的格式是唯一的，用于识别消息开始。应通过其低电平持续时间和总的持续时间识别起始位。

消息中起始位后的所有剩余数据位均有一致的时序。数据位结束时从高电平跳变为低电平即表示下一个数据位开始，但最后一位除外（此时 CEC 线保持高电平）。

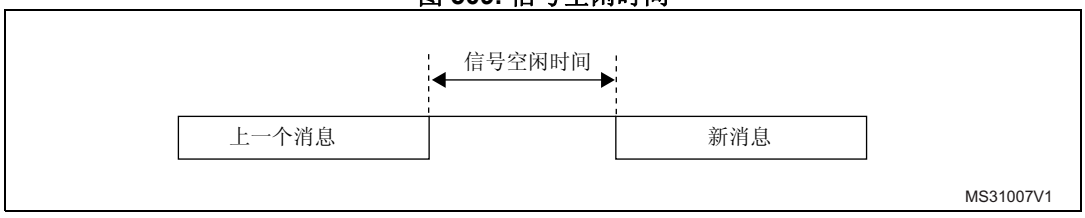
图 508. 位时序



39.4 仲裁

必须向 CEC 线发送或重新发送消息的所有器件必须确保 CEC 线已在多个位周期内处于无效状态。该信号空闲时间定义为从前一个帧的最后一位到下一条新消息开始的间隔，如下表所示。

图 509. 信号空闲时间



由于一次只允许一个发起方，因此提供了一种仲裁机制来避免多个发起方同时开始发送时发生冲突。

CEC 线的仲裁从起始位的上升沿开始，并持续到报头块内的发起方地址位结束时为止。在此周期内，发起方将监视 CEC 线，发起方驱动 CEC 线为高阻态时，同时会读取 CEC 线的状态，如果读回的状态为 0，随后它会认为丢失仲裁、停止发送并变为发送方。

图 510. 仲裁阶段

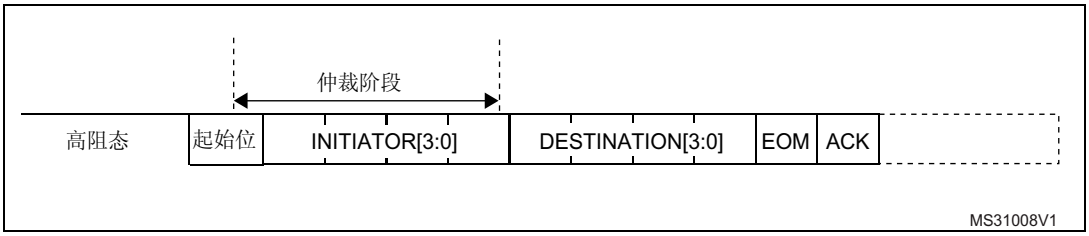
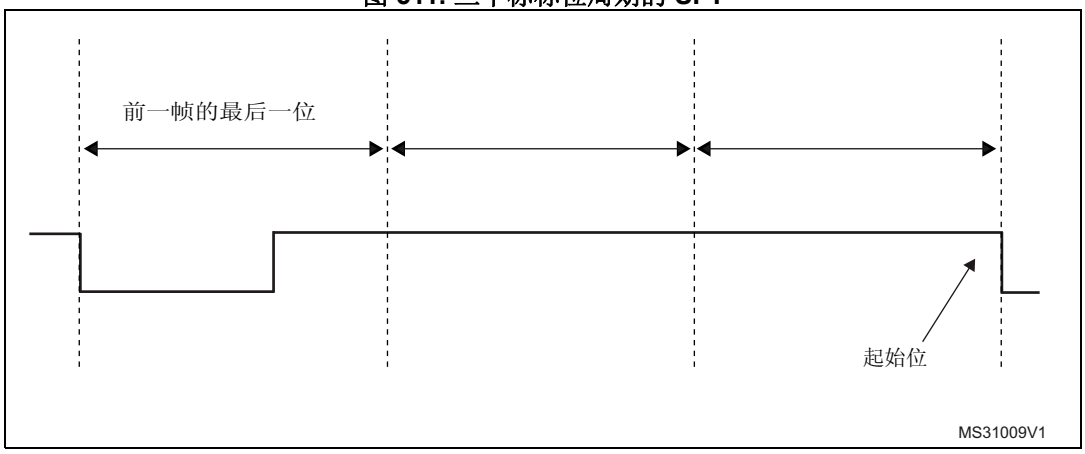


图 511 给出了三个标称位周期的 SFT 的示例。

图 511. 三个标称位周期的 SFT



可配置时间窗口在开始发送前计数。

在 SFT=0x0 配置下，HDMI-CEC 器件执行自动 SFT 计算，以确保符合 HDMI-CEC 标准：

- 如果 CEC 为最后一个发送失败的总线发起方，则为 2.5 个数据位周期
- 如果 CEC 是新的总线发起方，则为 4 个数据位周期
- 如果 CEC 为最后一个发送成功的总线发起方，则为 6 个数据位周期

这样的目的是保证将最高优先级赋予失败的发送，将最低优先级赋予最后一个成功完成发送的发起方。

另外，还可以将 SFT 位配置为计数固定的时序值。可能的值为 0.5、1.5、2.5、3.5、4.5、5.5 和 6.5 个数据位周期。

39.4.1 SFT 选项位

在 SFTOPT=0 的配置下，当由软件设置发送开始命令时 (TXSOM=1)，开始对 SFT 计数。

在 SFTOPT=1 时，若检测到总线空闲或线路错误条件，HDMI-CEC 器件将自动开始对 SFT 计数。如果在 TXSOM 命令时，SFT 定时器已经结束计数，则将立即开始发送，没有任何延迟。如果 SFT 定时器仍在运行，系统将等待定时器超时后再开始发送。

当 SFTOPT=1 时，下列检测到的总线事件将启动 SFT 定时器：

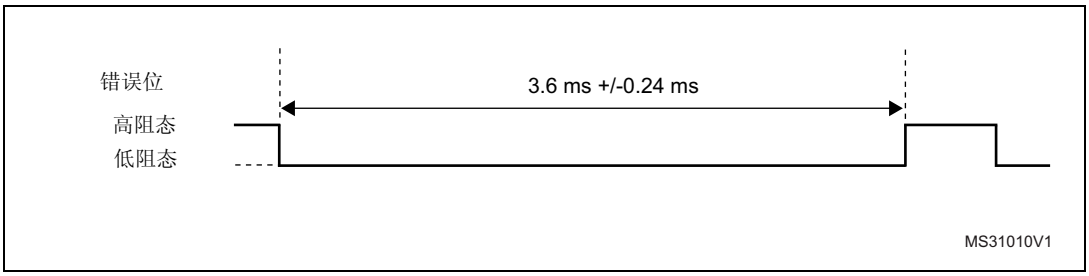
- 在常规结束发送/接收的情况下，当 TXEND/RXEND 位在消息中最后一位（ACK 位）的最短标称数据位持续时间内置 1 时。
- 在检测到发送错误的情况下，当检测到 TXERR 发送错误时 (TXERR=1) 启动 SFT 定时器。
- 在丢失 CEC 接收方应答的情况下，当在 ACK 位的标称采样时间内将 TXACKE 位置 1，将启动 SFT 定时器。
- 在出现发送下溢错误的情况下，当 TXUDR 位在 ACK 位结束时置 1 时，将启动 SFT 定时器。
- 在检测到表示接收中止的接收错误的情况下，将在检测到错误的同时启动 SFT 定时器。如果生成错误位，则在错误位结束时开始对 SFT 计数。
- 在出现错误起始位或任何未编码低阻抗空闲总线状态的情况下，当总线恢复为高阻抗空闲状态时，将立即重启 SFT 定时器。

39.5 错误处理

39.5.1 位错误

如果一个数据位（不包括起始位）被视为无效，则接收方应在 CEC 线上产生一个 1.4 倍到 1.6 倍标称数据位周期的低电平 (3.6 ms+/-0.24 ms) 来通知此错误。

图 512. 错误位时序



39.5.2 消息错误

在以下情况下，认为消息丢失，可重新发送：

- 在直接寻址的消息中未应答消息
- 在广播消息中否定应答消息
- 在 CEC 线上检测到意外的低阻抗（线路错误）

当 CEC 接口接收数据位时，可检测到三种错误标志：

39.5.3 位上升错误 (BRE)

BRE（位上升错误）：当在预期窗口外检测到位上升沿时置 1（请参见*图 513*）。BRE 标志还会在 BREIE=1 时生成 CEC 中断。

检测到 BRE 时，可根据 BRESTP 位的值停止接收消息，并会在 BREGEN 位置 1 时生成错误位。

若 BRESTP=1 时在广播消息中检测到 BRE，即使 BREGEN=0，也会生成错误位，以强制发起方在发送失败后重试。可通过配置 BREGEN=0 和 BRDNOGEN=1 禁止生成错误位。

39.5.4 短位周期错误 (SBPE)

提前检测到位下降沿时，SBPE 位将置 1（请参见*图 513*）。SBPE 标志还会在 SBPEIE=1 时生成 CEC 中断。

检测到 SBPE 错误时，始终会在线路上生成一个错误位。仅当设置监听模式 (LSTN=1) 并且满足以下条件时，检测到 SBPE 后才不生成错误位：

- 接收到包含 SBPE 的直接寻址消息
- 接收到包含 SBPE 和 BRDNOGEN 都为 1 的广播消息

39.5.5 长位周期错误 (LBPE)

未在有效窗口中检测到位下降沿时，LBPE 位置 1（请参见*图 513*）。LBPE 标志还会在 LBPEIE=1 时生成 CEC 中断。

LBPE 始终会停止接收过程，当 LBPEGEN 位置 1 时会在线路上生成错误位。

若在广播消息中检测到 LBPE，则即使 LBPEGEN=0 也会生成错误位，以强制发起方在发送失败后重试。可通过配置 LBPEGEN=0 和 BRDNOGEN=1 禁止生成错误位。

注： *必须避免 BREGEN=1 和 BRESTP=0 的配置。*

图 513. 错误处理

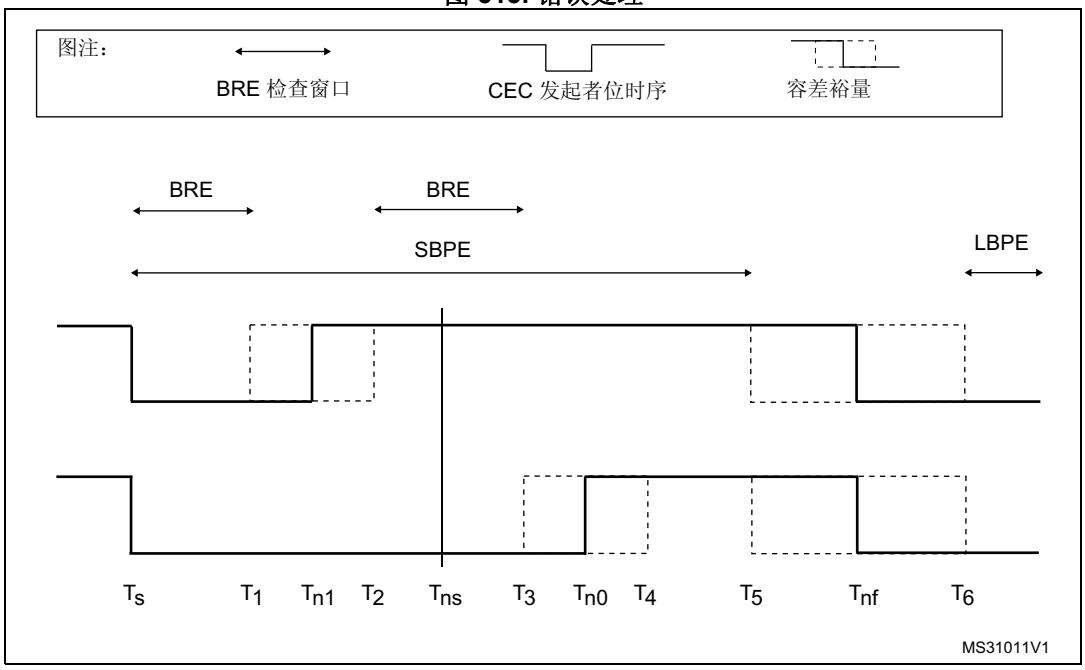


表 247. 错误处理时序参数

时间	RXTOL	ms	说明
T_s	x	0	位起始事件。
T_1	1	0.3	指示逻辑 1 时，低电平到高电平跳变的最早时间。
	0	0.4	
T_{n1}	x	0.6	指示逻辑 1 时，低电平到高电平跳变的标称时间。
T_2	0	0.8	指示逻辑 1 时，低电平到高电平跳变的最晚时间。
	1	0.9	
T_{ns}	x	1.05	标称采样时间。
T_3	1	1.2	允许器件恢复为高阻态（逻辑 0）的最早时间。
	0	1.3	
T_{n0}	x	1.5	允许器件恢复为高阻态（逻辑 0）的标称时间。
T_4	0	1.7	允许器件恢复为高阻态（逻辑 0）的最晚时间。
	1	1.8	
T_5	1	1.85	后续位开始的最早时间。
	0	2.05	
T_{nf}	x	2.4	标称数据位周期。
T_6	0	2.75	后续位开始的最晚时间。
	1	2.95	

39.5.6 发送错误检测 (TXERR)

如果 CEC 发起方在发送高阻抗且不需要接收方触发位时检测到 CEC 线处于低阻抗，则会将 TXERR 标志置 1。TXERR 标志还会在 TXERRIE=1 时生成 CEC 中断。

TXERR 触发后会停止发送消息。应用负责在发送失败后重试，最多可重试 5 次。

可按不同方式执行 TXERR 校验，具体取决于 CEC 线的不同状态和 RX 容差配置。

图 514. TXERR 检测

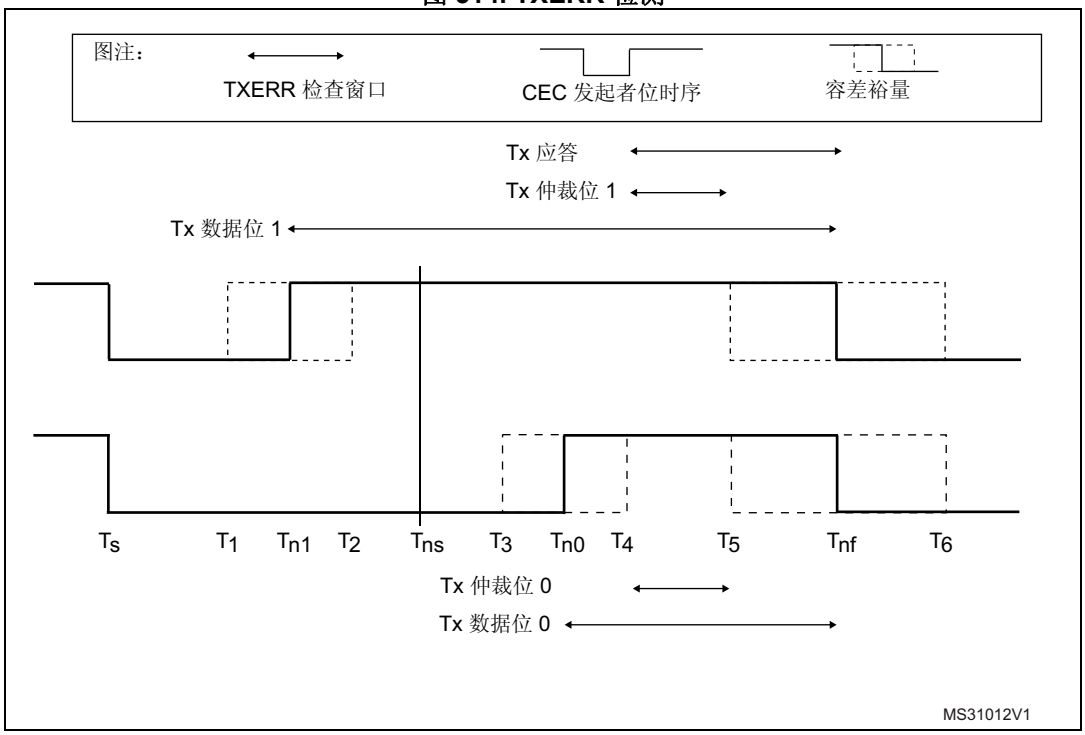


表 248. TXERR 时序参数

时间	RXTOL	ms	说明
T_s	x	0	位起始事件。
T_1	1	0.3	指示逻辑 1 时，低电平到高电平跳变的最早时间。
	0	0.4	
T_{n1}	x	0.6	指示逻辑 1 时，低电平到高电平跳变的标称时间。
T_2	0	0.8	指示逻辑 1 时，低电平到高电平跳变的最晚时间。
	1	0.9	
T_{ns}	x	1.05	标称采样时间。
T_3	1	1.2	允许器件恢复为高阻态（逻辑 0）的最早时间。
	0	1.3	
T_{n0}	x	1.5	允许器件恢复为高阻态（逻辑 0）的标称时间。
T_4	0	1.7	允许器件恢复为高阻态（逻辑 0）的最晚时间。
	1	1.8	
T_5	1	1.85	后续位开始的最早时间。
	0	2.05	
T_{nf}	x	2.4	标称数据位周期。
T_6	0	2.75	后续位开始的最晚时间。
	1	2.95	

39.6 HDMI-CEC 中断

以下情况下可以产生中断：

- 在接收期间，如果完成接收块传输或出现接收错误。
- 在发送期间，如果完成发送块传输或出现发送错误。

表 249. HDMI-CEC 中断

中断事件	事件标志	使能控制位
接收到 Rx 字节	RXBR	RXBRIE
接收结束	RXEND	RXENDIE
Rx 上溢	RXOVR	RXOVRIE
Rx 位上升错误	BRE	BREIE
Rx 短位周期错误	SBPE	SBPEIE
Rx 长位周期错误	LBPE	LBPEIE
Rx 丢失应答错误	RXACKE	RXACKEIE
仲裁丢失	ARBLST	ARBLSTIE
Tx 字节请求	TXBR	TXBRIE
发送结束	TXEND	TXENDIE
Tx 缓冲区下溢	TXUDR	TXUDRIE
Tx 错误	TXERR	TXERRIE
Tx 丢失应答错误	TXACKE	TXACKEIE

39.7 HDMI-CEC 寄存器

有关寄存器说明中使用的缩写，请参见*第 58 页上的第 1.1 节*。

39.7.1 CEC 控制寄存器 (CEC_CR)

CEC 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.	Res.	Res.	Res.
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.	TX EOM	TX SOM	CEC EN
													rs	rs	rw

位 31:3 保留，必须保持复位值。

位 2 **TXEOM**：Tx 消息结束 (Tx End Of Message)

TXEOM 位由软件置 1，来控制 CEC 消息的最后一个字节的发送。

TXEOM 由硬件清零，清零的时刻和条件与 TXSOM 相同。

0：EOM=0 时发送 TXDR 数据字节

1：EOM=1 时发送 TXDR 数据字节

注： CECEN=1 时，TXEOM 必须置 1

TXEOM 必须在向 TXDR 写入发送数据前置 1

如果 TXEOM 在 TXSOM=0 时置 1，发送的消息将仅包含 1 个字节 (HEADER)（PING 消息）

位 1 **TXSOM**：Tx 消息开始 (Tx Start Of Message)

TXSOM 位由软件置 1 来控制发送 CEC 消息的第一个字节。如果 CEC 消息只包含一个字节，TXEOM 必须在 TXSOM 之前置 1。

SFT 结束后，立刻在 CEC 线上发送起始位。如果 TXSOM 在接收消息的过程中置 1，则将在接收结束时开始发送。

在出现发送下溢 (TXUDR=1)、否定应答 (TXACKE=1) 和发送错误 (TXERR=1) 的情况下，在发送消息的最后一个字节和肯定应答后，TXSOM 将由硬件清零。此位还可通过 CECEN=0 清零。在仲裁丢失的情况下 (ARBLST=1)，此位不会清零，同时会自动重试发送。

TXSOM 还可用作状态位，用于通知应用程序是否存在正在挂起或执行的任何发送请求。应用程序可通过将 CECEN 位清零随时中止发送请求。

0：没有正在进行的 CEC 发送

1：CEC 发送命令

注： CECEN=1 时，TXSOM 必须置 1

发送数据可用于 TXDR 时，TXSOM 必须置 1

从 TXDR[7:4] 而非从仅用于接收的 CEC_CFGR.OAR 获取 HEADER 的前四位（包含自有外设地址）

位 0 **CECEN**：CEC 使能 (CEC Enable)

CECEN 位由软件置 1 和清零。CECEN=1 启动消息接收过程并使能 TXSOM 控制。CECEN=0 禁止 CEC 外设、将 CEC_CR 寄存器的所有位清零并中止任何正在进行的接收或发送过程。

0：CEC 外设关闭

1：CEC 外设开启

39.7.2 CEC 配置寄存器 (CEC_CFGR)

CEC configuration register

此寄存器用于配置 HDMI-CEC 控制器。

偏移地址：0x04

复位值：0x0000 0000

注意： 必须只在 CECEN=0 时对 CEC_CFGR 执行写操作。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LSTN	OAR[14:0]														
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.	SFT OPT	BRDN OGEN	LBPE GEN	BRE GEN	BRE STP	RX TOL	SFT[2:0]		
							rw	rw	rw	rw	rw	rw	rw		

位 31 **LSTN**：监听模式 (Listen mode)

LSTN 位由软件置 1 和清零。

0：CEC 外设只接收寻址到其自有地址 (OAR) 的消息。寻址到不同目标地址的消息将被忽略。始终接收广播消息。

1：CEC 外设接收寻址到其自有地址 (OAR) 的消息，此时会发送肯定应答。也接收寻址到不同目标地址的消息，但此时不发送应答。

位 30:16 **OAR**：自有地址配置 (Own addresses configuration)

OAR 位由软件置 1 来选择接收模式下必须考虑的目标逻辑地址。每个 bit 置 1 后，它所对应的位置的值，就是被使能的 CEC 逻辑地址的值。OAR 字段的最低位，对应 CEC 逻辑地址 0。

在 HEADER 接收结束时，所接收的目标地址将与使能的地址进行比较。在地址匹配的情况下，将应答并接收传入的消息。在地址不匹配的情况下，仅在监听模式下 LSTN=1) 接收传入的消息，但不发送应答。始终接收广播消息。

示例：

OAR = 0b000 0000 0010 0001 意味着 CEC 应答地址 0x0 和 0x5。因此，将接收传入这两个地址中之一的每个消息。

位 15:9 保留，必须保持复位值。

位 8 **SFTOP**：*SFT 选项位 (SFT Option Bit)*

SFTOPT 位由软件置 1 和清零。

0：当 TXSOM 由软件置 1 时，将启动 SFT 定时器

1：在消息发送/接收结束时自动启动 SFT 定时器

位 7 **BRDNOGEN**：避免在广播中生成错误位 (Avoid Error-Bit *Generation in Broadcast)*

BRDNOGEN 位由软件置 1 和清零。

0：在广播消息上检测到 BRE 且满足 BRESTP=1 和 BREGEN=0 时会在 CEC 线上生成错误位。在广播消息上检测到 LBPE 且满足 LBPEGEN=0 时会在 CEC 线上生成错误位。

1：在上述相同条件下不会生成错误位。即使在监听模式下，在广播消息中检测到 SBPE，也不会生成错误位。

位 6 **LBPEGEN**：在长位周期错误时生成错误位 (Generate Error-Bit on Long *Bit Period Error*)

LBPEGEN 位由软件置 1 和清零。

0：检测到 LBPE 时不在 CEC 线上生成错误位

1：检测到 LBPE 时在 CEC 线上生成错误位

注： 如果 BRDNOGEN=0，即使 LBPEGEN=0，也会在广播中检测到 LBPE 后生成错误位。

位 5 **BREGEN**：在位上升错误时生成错误位 (Generate Error-Bit on *Bit Rising Error*)

BREGEN 位由软件置 1 和清零。

0：检测到 BRE 时不在 CEC 线上生成错误位

1：检测到 BRE 时在 CEC 线上生成错误位（如果 BRESTP 置 1）

注： 如果 BRDNOGEN=0，即使 BREGEN=0，也会在广播中检测到 BRE 和 BRESTP=1 后生成错误位。

位 4 **BRESTP**：*在位上升错误时 Rx 停止 (Rx-Stop on Bit Rising Error)*

BRESTP 位由软件置 1 和清零。

0：检测到 BRE 时不停止接收 CEC 消息。在 1.05 ms 内对数据位进行采样。

1：检测到 BRE 时停止接收消息

位 3 **RXTOL**：Rx 容差 (Rx-Tolerance)

RXTOL 位由软件置 1 和清零。

0：标准容差裕量：

– 起始位，+/- 200 µs 上升，+/- 200 µs 下降。

– 数据位：+/- 200 µs 上升。+/- 350 µs 下降。

1：扩展容差

– 起始位，+/- 400 µs 上升，+/- 400 µs 下降

– 数据位，+/- -300 µs 上升，+/- 500 µs 下降

位 2:0 **SFT**：*信号空闲时间 (Signal Free Time)*

SFT 位由软件置 1。在 SFT=0x0 的配置下，发送前等待的标称数据位周期数由硬件根据发送历史管理。在所有其它配置下，SFT 数由软件决定。

″ 0x0

– 如果 CEC 为最后一个发送失败的总线发起方，则为 2.5 个数据位周期（ARBLST=1、TXERR=1、TXUDR=1 或 TXACKE= 1）

– 如果 CEC 是新的总线发起方，则为 4 个数据位周期

– 如果 CEC 为最后一个发送成功的总线发起方，则为 6 个数据位周期 (TXEOM=1)

″ 0x1：0.5 个标称数据位周期

″ 0x2：1.5 个标称数据位周期

″ 0x3：2.5 个标称数据位周期

″ 0x4：3.5 个标称数据位周期

″ 0x5：4.5 个标称数据位周期

″ 0x6：5.5 个标称数据位周期

″ 0x7：6.5 个标称数据位周期

39.7.3 CEC 发送数据寄存器 (CEC_TXDR)

CEC Tx data 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXD[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 **TXD[7:0]**：Tx 数据寄存器 (Tx Data register)。
TXD 是一个只写寄存器，其中包含要发送的数据字节。
注： TXSTART=1 时，必须对 TXD 执行写操作

39.7.4 CEC 接收数据寄存器 (CEC_RXDR)

CEC Rx Data 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RXD[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 **RXD[7:0]**：Rx 数据寄存器 (Rx Data register)。
RXD 是一个只读寄存器，包含从 CEC 线接收的最后一个数据字节。

39.7.5 CEC 中断和状态寄存器 (CEC_ISR)

CEC Interrupt and Status 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.	TX ACKE	TX ERR	TX UDR	TX END	TXBR	ARB LST	RX ACKE	LBPE	SBPE	BRE	RX OVR	RX END	RXBR
			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 31:13 保留，必须保持复位值。

位 12 **TXACKE**：Tx 丢失应答错误 (Tx-Missing Acknowledge Error)

在发送模式下，TXACKE 由硬件置 1 以通知应用程序未接收到任何应答。在广播发送时，TXACKE 通知应用程序已接收到否定应答。TXACKE 中止消息发送并清除 TXSOM 和 TXEOM 控制位。

TXACKE 由软件写 1 清零。

位 11 **TXERR**：Tx 错误 (Tx-Error)

在发送模式下，如果 CEC 发起方在 CEC 线释放时检测到 CEC 处于低阻抗，则 TXERR 将由硬件置 1。TXACKE 中止消息发送并清除 TXSOM 和 TXEOM 控制位。

TXACKE 由软件写 1 清零。

位 10 **TXUDR**：Tx 缓冲区下溢 (Tx-Buffer Underrun)

在发送模式下，如果应用程序未在下一个字节发送前及时加载 TXDR，则 TXUDR 将由硬件置 1。TXUDR 中止消息发送并清除 TXSOM 和 TXEOM 控制位。

TXUDR 由软件写 1 清零。

位 9 **TXEND**：发送结束 (End of Transmission)

TXEND 由硬件置 1 后，可向应用程序通知已成功发送 CEC 消息的最后一个字节。TXEND 将 TXSOM 和 TXEOM 控制位清零。

TXEND 由软件写 1 清零。

位 8 **TXBR**：Tx 字节请求 (Tx-Byte Request)

TXBR 由硬件置 1 后可通知应用程序必须向 TXDR 写入下一个发送数据。当发送完当前发送字节的第 4 位时，TXBR 将置 1。应用程序必须在出现发送下溢错误 (TXUDR) 前，在 6 个标称数据位周期内向 TXDR 发送下一个字节。

TXBR 由软件写 1 清零。

位 7 **ARBLST**：仲裁丢失 (Arbitration lost)

ARBLST 由硬件置 1 以通知应用程序：由于 TXSOM 命令后发生仲裁丢失事件，CEC 器件正切换为接收模式。引起 ARBLST 的原因可能是有别的 CEC 器件提前开始争用总线，或者是同时开始但是拥有较高的优先级。ARBLST 触发后，TXSOM 位将一直等待下一次发送尝试。

ARBLST 由软件写 1 清零。

位 6 **RXACKE**：Rx 丢失应答 (Rx-Missing Acknowledge)

在发送模式下，RXACKE 由硬件置 1 后以通知应用程序未在 CEC 线上检测到任何应答。RXACKE 仅适用于广播消息，在监听模式下仅适用于非直接寻址的消息（OAR 中不使能目标地址）。RXACKE 用于中止接收消息。

RXACKE 由软件写 1 清零。

位 5 **LBPE**：Rx 长位周期错误 (Rx-Long Bit Period Error)

当数据位波形上检测到长位周期错误时，LBPE 由硬件置 1。在仍需要下降沿的情况下，当 RXTOL 允许的最大位扩展容差结束时，LBPE 将置 1。LBPE 总是会停止接收 CEC 消息。如果 LBPEGEN=1，LBPE 会在 CEC 线上生成错误位。广播时，即使 LBPEGEN=0 也会生成错误位。

LBPE 由软件写 1 清零。

位 4 **SBPE**：Rx 短位周期错误 (Rx-Short Bit Period Error)

当数据位波形上检测到短位周期错误时，SBPE 由硬件置 1。出现所预期的下降沿时，SBPE 将置 1。SBPE 在 CEC 线上生成错误位。

SBPE 由软件写 1 清零。

位 3 **BRE**：Rx 位上升错误 (Rx-Bit Rising Error)

当数据位波形上检测到位上升错误时，BRE 由硬件置 1。在仍需要上升沿的情况下，在上升沿出现的时机不对时或 RXTOL 允许的最大 BRE 容差结束时，BRE 将置 1。如果 BRESTP=1，BRE 将停止接收消息。如果 BREGEN=1，BRE 会在 CEC 线上生成错误位。

BRE 由软件写 1 清零。

位 2 **RXOVR**：Rx 上溢 (Rx-Overrun)

如果在 CEC 线上接收到新字节并将新字节存储到 RXD 中时 RXBR 位已经是置位状态，则 RXOVR 将由硬件置 1。RXOVR 触发后会停止接收消息，因此不会发送应答。广播时，将发送否定应答。

RXOVR 由软件写 1 清零。

位 1 **RXEND**：接收结束 (End Of Reception)

RXEND 由硬件置 1 以通知应用程序已从 CEC 线接收到 CEC 消息的最后一个字节并将其存储到 RXD 缓冲区。RXEND 与 RXBR 同时置 1。

RXEND 由软件写 1 清零。

位 0 **RXBR**：接收到 Rx 字节 (Rx-Byte Received)

RXBR 位由硬件置 1 以通知应用程序已从 CEC 线接收到新字节并将其存储到 RXD 缓冲区。

RXBR 由软件写 1 清零。

39.7.6 CEC 中断使能寄存器 (CEC_IER)

CEC interrupt enable 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
Res.	Res.	Res.	TXACKIE	TXERRIE	TX UDRIE	TXENDIE	TXBRIE	ARBLSTIE	RXACKIE	LBPEIE	SBPEIE	BREIE	RXOVRIE	RXENDIE	RXBRIE
			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 **TXACKIE**：Tx 丢失应答错误中断使能 (Tx-Missing Acknowledge Error Interrupt Enable)
TXACKEIE 位由软件置 1 和清零。
0：禁止 TXACKE 中断
1：使能 TXACKE 中断

位 11 **TXERRIE**：Tx 错误中断使能 (Tx-Error Interrupt Enable)
TXERRIE 位由软件置 1 和清零。
0：禁止 TXERR 中断
1：使能 TXERR 中断

位 10 **TXUDRIE**：Tx 下溢中断使能 (Tx-Underrun Interrupt Enable)
TXUDRIE 位由软件置 1 和清零。
0：禁止 TXUDR 中断
1：使能 TXUDR 中断

位 9 **TXENDIE**：Tx 消息结束中断使能 (Tx-End Of Message Interrupt Enable)
TXENDIE 位由软件置 1 和清零。
0：禁止 TXEND 中断
1：使能 TXEND 中断

位 8 **TXBRIE**：Tx 字节请求中断使能 (Tx-Byte Request Interrupt Enable)
TXBRIE 位由软件置 1 和清零。
0：禁止 TXBR 中断
1：使能 TXBR 中断

位 7 **ARBLSTIE**：仲裁丢失中断使能 (Arbitration Lost Interrupt Enable)
ARBLSTIE 位由软件置 1 和清零。
0：禁止 ARBLST 中断
1：使能 ARBLST 中断

位 6 **RXACKIE**：Rx 丢失应答错误中断使能 (Rx-Missing Acknowledge Error Interrupt Enable)
RXACKIE 位由软件置 1 和清零。
0：禁止 RXACKE 中断
1：使能 RXACKE 中断

位 5 **LBPEIE**：长位周期错误中断使能 (Long Bit Period Error Interrupt Enable)
LBPEIE 位由软件置 1 和清零。
0：禁止 LBPE 中断
1：使能 LBPE 中断

位 4 **SBPEIE**：短位周期错误中断使能 (Short Bit Period Error Interrupt Enable)
SBPEIE 位由软件置 1 和清零。
0：禁止 SBPE 中断
1：使能 SBPE 中断

位 3 **BREIE**：位上升错误中断使能 (Bit Rising Error Interrupt Enable)
BREIE 位由软件置 1 和清零。
0：禁止 BRE 中断
1：使能 BRE 中断

位 2 **RXOVRIE**：Rx 缓冲区上溢中断使能 (Rx-Buffer Overrun Interrupt Enable)
RXOVRIE 位由软件置 1 和清零。
0：禁止 RXOVR 中断
1：使能 RXOVR 中断

位 1 **RXENDIE**：接收结束中断使能 (End Of Reception Interrupt Enable)
RXENDIE 位由软件置 1 和清零。
0：禁止 RXEND 中断
1：使能 RXEND 中断

位 0 **RXBRIE**：接收到 Rx 字节中断使能 (Rx-Byte Received Interrupt Enable)
RXBRIE 位由软件置 1 和清零。
0：禁止 RXBR 中断
1：使能 RXBR 中断

注意： (*) 必须只在 CECEN=0 时对 CEC_IER 执行写操作。

39.7.7 HDMI-CEC 寄存器映射

下表对 HDMI-CEC 寄存器进行了汇总。

表 250. HDMI-CEC 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	**CEC_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.	Res.	Res.	TXEOM	TXSOM	CECEN
	Reset value																														0	0	0
0x04	**CEC_CFGR**	LSTN	OAR[14:0]															Res.	Res.	Res.	Res.	Res.	Res.	Res.	SFTOPT	BRDNOGEN	LBPEGEN	BREGEN	BRESTP	RXTOL	SFT[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
0x08	**CEC_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.	TXD[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x0C	**CEC_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.	RXD[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x10	**CEC_ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXACKE	TXERR	TXUDR	TXEND	TXBR	ARBLST	RXACKE	LBPE	SBPE	BRE	RXOVR	RXEND	RXBR
	Reset value																				0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	**CEC_IER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXACKIE	TXERRIE	TXUDRIE	TXENDIE	TXBRIE	ARBLSTIE	RXACKIE	LBPEIE	SBPEIE	BREIE	RXOVRIE	RXENDIE	RXBRIE
	Reset value																				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 64 页上的第 2.2.2 节*。

40 调试支持 (DBG)

40.1 概述

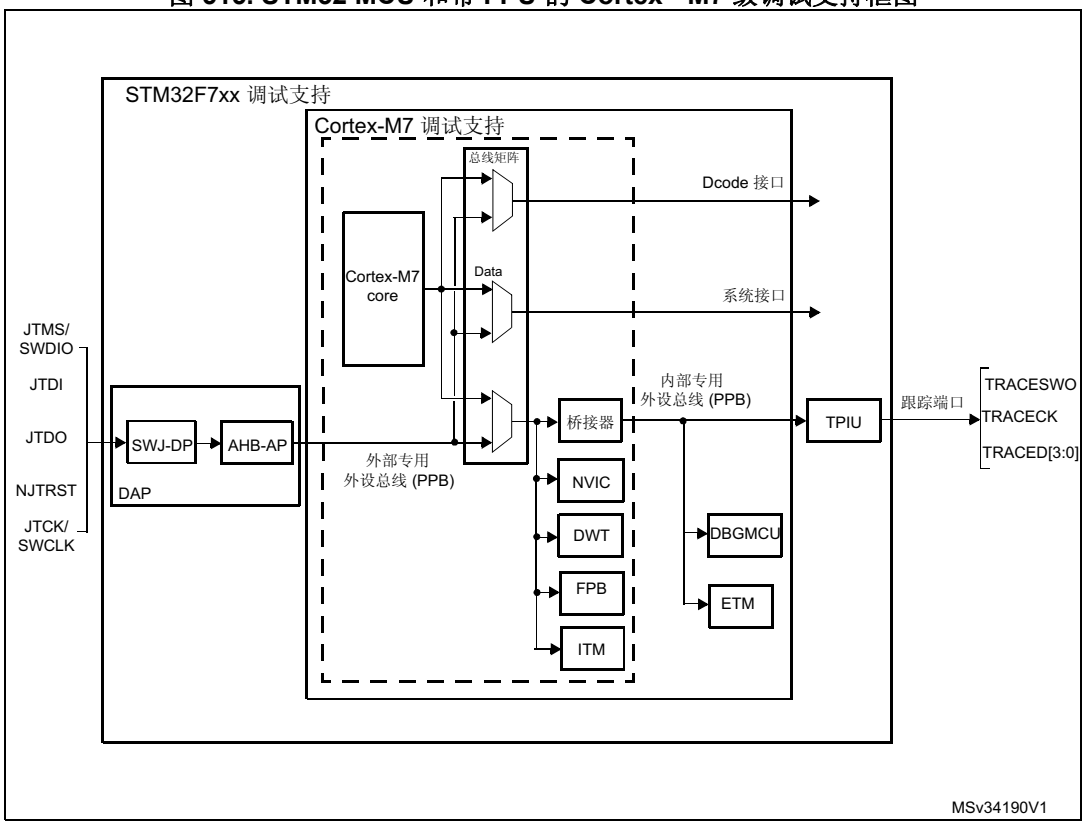
STM32F75xxx 和 STM32F74xxx 的内核是带 FPU 的 Cortex®-M7 内核，该内核包含用于高级调试功能的硬件。利用这些调试功能，可以在取指（指令断点）或取访问数据（数据断点）时停止内核。内核停止时，可以查询内核的内部状态和系统的外部状态。查询完成后，将恢复内核和系统并恢复程序执行。

当调试主机与 STM32F75xxx 和 STM32F74xxx MCU 相连并进行调试时，将使用调试功能。

提供两个调试接口：

- 串行接口
- JTAG 调试端口

图 515. STM32 MCU 和带 FPU 的 Cortex®-M7 级调试支持框图



注： *带 FPU 的 Cortex®-M7 内核中内置的调试功能是《ARM® CoreSight 组件技术参考手册》的一部分。*

带 FPU 的 ARM® Cortex®-M7 内核提供集成片上调试支持。它包括：

- SWJ-DP：串行线/JTAG 调试端口
- AHP-AP：AHB 访问端口
- ITM：指令跟踪宏单元
- FPB：Flash 指令断点
- DWT：数据观察点触发
- TPIU：跟踪端口单元接口（大封装上提供，其中会映射相应引脚）
- ETM：嵌入式跟踪宏单元（大封装上提供，其中会映射相应引脚）

它还包括专用于 STM32F75xxx 和 STM32F74xxx 的调试功能：

- 灵活调试引脚分配
- MCU 调试盒（支持低功耗模式和对外设时钟的控制等）

注： *有关带 FPU 的 ARM® Cortex®-M7 内核支持的调试功能的详细信息，请参见《带 FPU 的 Cortex®-M7 技术参考手册》和《CoreSight 组件技术参考手册》（请参见第 40.2 节：ARM® 参考文档）。*

40.2 ARM® 参考文档

- Cortex®-M7 带 FPU 的 Cortex®-M7 技术参考手册 (TRM)
  （请参见第 1 页上的相关文档）
- ARM® 调试接口 V5 架构规范
- ARM® CoreSight 组件技术参考手册

40.3 SWJ 调试端口（串行接口和 JTAG）

STM32F75xxx 和 STM32F74xxx 的内核集成有串行线/JTAG 调试端口 (SWJ-DP)。该端口是 ARM® 标准 CoreSight 调试端口，具有 JTAG-DP（5 引脚）接口和 SW-DP（2 引脚）接口。

- JTAG 调试端口 (JTAG-DP) 提供用于连接到 AHP-AP 端口的 5 引脚标准 JTAG 接口。
- 串行线调试端口 (SW-DP) 提供用于连接到 AHP-AP 端口的 2 引脚（时钟 + 数据）接口。

在 SWJ-DP 中，SW-DP 的 2 个 JTAG 引脚与 JTAG-DP 的 5 个 JTAG 引脚中的部分引脚复用。

图 516. SWJ 调试端口

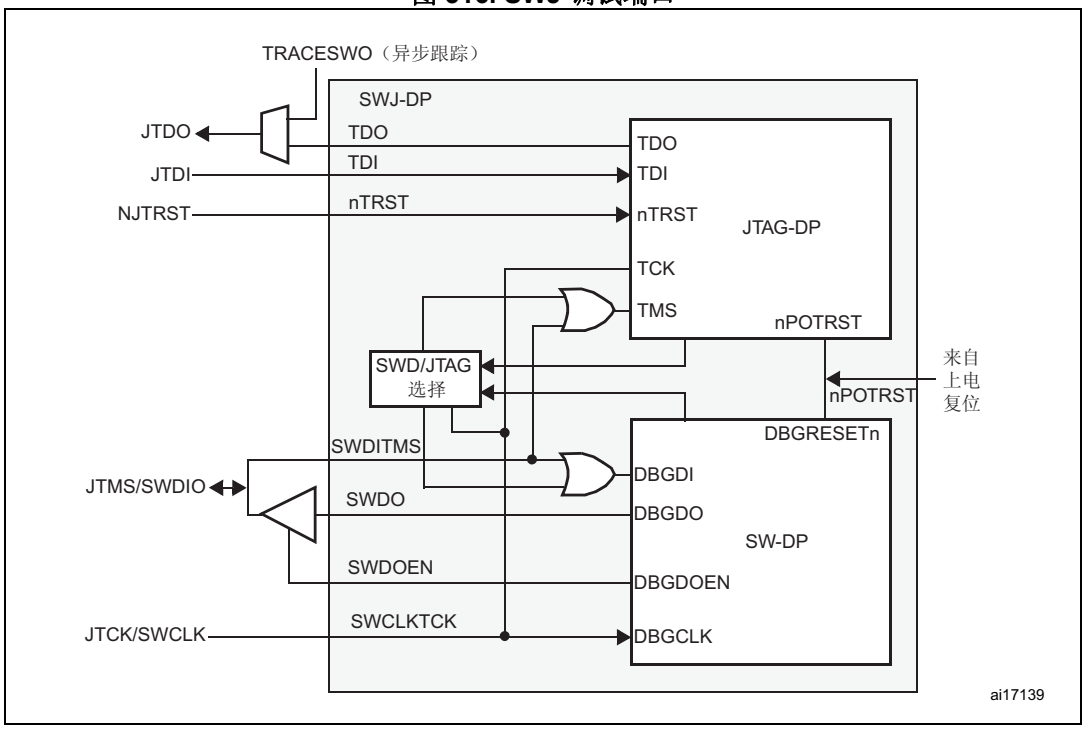


图 516 显示了异步 TRACE 输出 (TRACESWO) 与 TDO 复用。这意味着异步跟踪只能在 SW-DP 上实现，不能在 JTAG-DP 上实现。

40.3.1 JTAG-DP 或 SW-DP 的切换机制

默认调试接口是 JTAG 接口。

如果调试主机想要切换到 SW-DP，它必须在 TMS/TCK（分别映射到 SWDIO 和 SWCLK）上提供专用的 JTAG 序列，用于禁止 JTAG-DP 并使能 SW-DP。这样便可仅使用 SWCLK 和 SWDIO 引脚来激活 SWDP。

该序列为：

1. 输出超过 50 个 TCK 周期的 TMS (SWDIO) = 1 信号
2. 输出 16 个 TMS (SWDIO) 信号 0111100111100111 (MSB)
3. 输出超过 50 个 TCK 周期的 TMS (SWDIO) = 1 信号

40.4 引脚排列和调试端口引脚

STM32F75xxx 和 STM32F74xxx MCU 的不同封装有不同的有效引脚数。因此，某些与引脚相关的功能（TPIU 并行输出接口）可能随封装而不同。

40.4.1 SWJ 调试端口引脚

STM32F75xxx 和 STM32F74xxx 的五个通用 I/O 可用作 SWJ-DP 接口引脚。所有封装都提供这些引脚。

表 251. SWJ 调试端口引脚

SWJ-DP 引脚名称	JTAG 调试端口		SW 调试端口		引脚分配
	类型	说明	类型	调试分配	
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/TRACESWO	O	JTAG 测试数据输出	-	TRACESWO（如果使能异步跟踪）	PB3
NJTRST	I	JTAG 测试 nReset	-	-	PB4

40.4.2 灵活的 SWJ-DP 引脚分配

复位（SYSRESETn 或 PORESETn）后，会将用于 SWJ-DP 的全部 5 个引脚指定为专用引脚，可供调试工具立即使用（请注意，除非由调试工具明确编程，否则不分配跟踪输出）。

但是，STM32F75xxx 和 STM32F74xxx MCU 可以禁止部分或全部 SWJ-DP 端口，进而释放相关引脚（下表中的灰色部分）以用作通用 IO (GPIO)。有关如何禁止 SWJ-DP 端口引脚的更多详细信息，请参见*第 6.3.2 节：I/O 引脚复用功能复用器和映射*。

表 252. 灵活的 SWJ-DP 引脚分配

可用的调试端口	分配的 SWJ IO 引脚				
	PA13/ JTMS/ SWDIO	PA14 / JTCK/ SWCLK	PA15 / JTDI	PB3 / JTDO	PB4/ NJTRST
全部 SWJ (JTAG-DP + SW-DP) - 复位状态	X	X	X	X	X
全部 SWJ (JTAG-DP + SW-DP)，但不包括 NJTRST	X	X	X	X	
禁止 JTAG-DP 和使能 SW-DP	X	X			
禁止 JTAG-DP 和禁止 SW-DP	已释放				

注： *当 APB 桥的写缓冲区已满后，还需要一个额外的 APB 周期来写入 GPIO_AFR 寄存器。这是因为释放 JTAGSW 引脚需要两个周期，以保证输入内核的 nTRST 和 TCK 信号的平稳。*

- 周期 1：输入 1/0 的 JTAGSW 信号到内核（TRST、TDI 和 TMS 为 1，TCK 为 0）
- 周期 2：GPI/O 控制器获得 SWJTAG I/O 引脚的控制信号（如对方向，上拉/下拉，施密特触发等的控制）。

40.4.3 JTAG 引脚上的内部上拉和下拉

必须确保 JTAG 输入引脚不悬空，因为这些引脚直接连接到用于控制调试模式功能的触发器。还必须特别注意 SWCLK/TCK 引脚，该引脚直接连接到一些触发器的时钟。

为避免 IO 电平浮空，器件在 JTAG 引脚上内置有内部上拉和下拉：

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

用户软件释放 JTAG IO 后，GPIO 控制器便会再次取得控制权。GPIO 控制寄存器的复位状态会将 I/O 置于：

- NJTRST：AF 输入上拉
- JTDI：AF 输入上拉
- JTMS/SWDIO：AF 输入上拉
- JTCK/SWCLK：AF 输入下拉
- JTDO：AF 输出悬空

软件可以把这些 I/O 口作为普通的 I/O 口使用。

注： *JTAG IEEE 标准建议在 TDI、TMS 和 nTRST 上添加上拉，但对 TCK 没有特殊建议。但是，对于 JTCK，器件需要集成下拉。*

由于带有上拉和下拉电阻，因此无需添加外部电阻。

40.4.4 使用串行接口以及释放未使用的调试引脚以用作 GPIO

为了利用串行调试接口以便释放一些 GPIO，用户软件必须在 GPIO_MODER 寄存器中更改 GPIO（PA15、PB3 和 PB4）配置模式。这样便可释放 PA15、PB3 和 PB4，以将这些引脚用作 GPIO。

调试时，主机执行以下操作：

- 在系统复位状态下，分配所有 SWJ 引脚 (JTAG-DP + SW-DP)。
- 在系统复位状态下，调试主机发送 JTAG 序列，以从 JTAG-DP 切换到 SW-DP。
- 仍然在系统复位状态下，调试主机在复位地址处设置断点。
- 释放复位信号，内核停止在复位地址处。
- 从此所有调试通信均使用 SW-DP 完成，然后通过用户软件将其它 JTAG 引脚重新分配为 GPIO。

注： *对于用户软件设计，请注意：*

请记住，要释放调试引脚，在复位后一直到用户软件释放引脚这段期间，这些引脚仍然处于输入上拉（nTRST、TMS 和 TDI）、下拉 (TCK) 或输出三态 (TDO)。

当这些引脚被配置为专用引脚时（JTAG 或者 SW 或者 TRACE），修改相应的普通 I/O 口配置寄存器是无效的。

40.5 STM32F75xxx 和 STM32F74xxx JTAG 调试端口连接

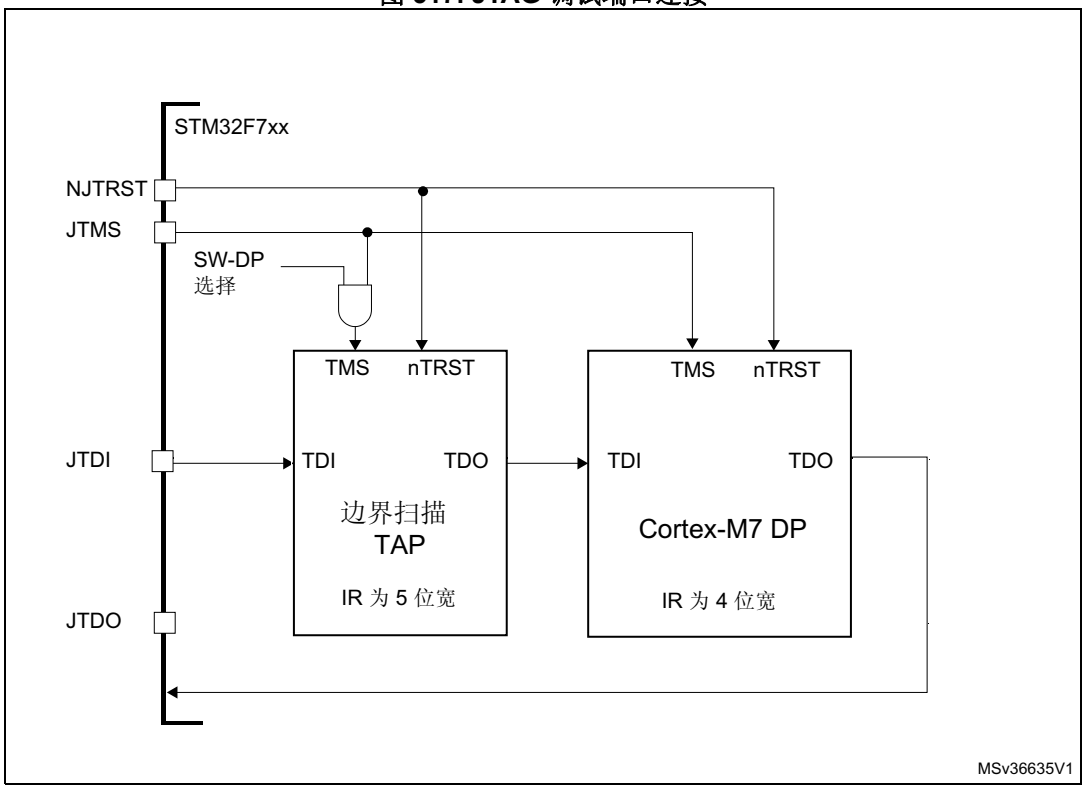
STM32F75xxx 和 STM32F74xxx MCU 集成了两个串联的 JTAG 调试端口、边界扫描调试端口（IR 宽度为 5 位）和带 FPU 的 Cortex®-M7 的调试端口（IR 宽度为 4 位）。

要访问带 FPU 的 Cortex®-M7 的调试端口以进行调试：

1. 首先，必须将 BYPASS 指令移位输入边界扫描调试端口。
2. 其次，在移位输入 IR 时，每个扫描链包含 9 个比特位 (=5+4)，对于不用的调试端口，必须输入 BYPASS 指令。
3. 移位输入数据时，不用的调试端口处于 BYPASS 模式下，因此数据扫描链需要额外添加一位比特位。

注： ***重要提示：****使用专用 ARM® JTAG 序列选择串行线后，将自动禁止边界扫描调试端口（JTMS 强制为高电平）。*

图 517. JTAG 调试端口连接



40.6 ID 代码和锁定机制

STM32F75xxx 和 STM32F74xxx MCU 中提供了一些 ID 代码。ST 强烈建议工具设计人员使用位于外部 PPB 存储器映射的地址 0xE0042000 中的 MCU DEVICE ID 代码锁定调试工具。

40.6.1 MCU 器件 ID 代码

STM32F75xxx 和 STM32F74xxx MCU 集成了 MCU ID 代码。此 ID 标识 ST MCU 的料号和晶片版本。它是 DBG_MCU 组件的一部分并映射到外部 PPB 总线上（请参见*第 1490 页上的第 40.16 节*）。可通过 JTAG 调试端口（4 到 5 个引脚）或 SW 调试端口（2 个引脚）或者用户软件访问此代码。即使在 MCU 处于系统复位状态下也可以访问。

调试软件/编程工具只能将 DEV_ID(11:0) 用于识别芯片。

DBGMCU_IDCODE

地址：0xE004 2000

仅支持 32 位访问。只读。

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

位 31:16 **REV_ID(15:0)**：版本标识符 (Revision identifier)
此字段指示器件的版本：
0x1000 = 版本 A
0x1001 = 版本 Z

位 15:12 保留，必须保持复位值。

位 11:0 **DEV_ID(11:0)**：器件标识符 (Device identifier)
器件 ID 为 0x449。

40.6.2 边界扫描调试端口

JTAG ID 代码

STM32F75xxx 和 STM32F74xxx BSC（边界扫描）调试端口集成了与 0x06449041 等效的 JTAG ID 代码。

40.6.3 Cortex®-M7 带 FPU 的 Cortex®-M7 的调试端口

带 FPU 的 ARM® Cortex®-M7 的调试端口集成了 JTAG ID 代码。此 ID 代码是 ARM® 的默认代码，未经修改。此代码只能通过 JTAG 调试端口进行访问。
此代码为 0x5BA00477（对应于带 FPU 的 Cortex®-M7，请参见*第 40.2 节：ARM® 参考文档*）。

40.6.4 Cortex®-M7 带 FPU 的 Cortex®-M7 的 JEDEC-106 ID 代码

带 FPU 的 ARM® Cortex®-M7 集成了 JEDEC-106 ID 代码。它位于映射到内部 PPB 总线地址为 0xE00FF000_0xE00FFFFF 的 4KB ROM 表中。

可通过 JTAG 调试端口（4 到 5 个引脚）或 SW 调试端口（2 个引脚）或者用户软件访问此代码。

40.7 JTAG 调试端口

标准 JTAG 状态机使用一个 4 位指令寄存器 (IR) 和五个数据寄存器实现（有关完整详细信息，请参见带 FPU 的 Cortex®-M7 *技术参考手册 (TRM)；有关参考信息，请参见第 40.2 节：ARM® 参考文档*）。

表 253. JTAG 调试端口数据寄存器

IR(3:0)	数据寄存器	详细信息
1111	BYPASS [1 位]	
1110	IDCODE [32 位]	ID CODE 0x06449041（带 FPU 的 ARM® Cortex®-M7 的 ID 代码）
1010	DPACC [35 位]	调试接口寄存器 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 其它 = 保留 有关 A(3:2) 位的说明，请参见表 254

表 253. JTAG 调试端口数据寄存器（续）

IR(3:0)	数据寄存器	详细信息
1011	APACC [35 位]	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] = 2 位地址（子地址 AP 寄存器）。 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 其它 = 保留 有多个 AP 寄存器（请参见 AHB-AP），这些寄存器按以下组合进行寻址： – 移位值 A[3:2] – DP SELECT 寄存器的当前值
1000	ABORT [35 位]	中止寄存器 – 位 31:1 = 保留 – 位 0 = DAPABORT：写入 1 以生成 DAP 中止。

表 254. 32 位调试端口寄存器，通过移位值 A[3:2] 进行寻址

地址	A(3:2) 值	说明
0x0	00	保留，必须保持复位值。
0x4	01	DP CTRL/STAT 寄存器。用于： – 请求系统或调试上电 – 配置 AP 访问的传输操作 – 控制比较和验证操作 – 读取一些状态标志（上溢和上电确认）
0x8	10	DP SELECT 寄存器：用于选择当前访问端口和活动的 4 字寄存器窗口。 – 位 31:24：APSEL：选择当前 AP – 位 23:8：保留 – 位 7:4：APBANKSEL：在当前 AP 上选择活动的 4 字寄存器窗口 – 位 3:0：保留
0xC	11	DP RDBUFF 寄存器：用于通过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）

40.8 SW 调试端口

40.8.1 SW 协议简介

此同步串行协议使用两个引脚：

- SWCLK：从主机到目标的时钟
- SWDIO：双向

利用该协议，可以同时读取和写入两组寄存器组（DPACC 寄存器组和 APACC 寄存器组）。

传输数据时，LSB 在前。

对于 SWDIO 双向管理，必须在电路板上对线路进行上拉（ARM® 建议采用 100 KΩ）。

每次在协议中更改 SWDIO 的方向时，都会插入转换时间，此时线路即不受主机驱动也不受目标驱动。默认情况下，此转换时间为一位时间，但可以通过配置 SWCLK 频率来调整。

40.8.2 SW 协议序列

每个序列包括三个阶段：

1. 主机发送的数据包请求（8 位）
2. 目标发送的确认响应（3 位）
3. 主机或目标发送的数据传输阶段（33 位）

表 255. 数据包请求（8 位）

位	名称	说明
0	启动	必须为 1
1	APnDP	0：DP 访问 1：AP 访问
2	RnW	0：写请求 1：读请求
4:3	A(3:2)	DP 或 AP 寄存器的地址字段（请参见*表 254*）
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。由于存在上拉，因此必须由目标读为 1

有关 DPACC 和 APACC 寄存器的详细说明，请参见带 FPU 的 Cortex®-M7 *TRM*。

数据包请求后面始终为转换时间（默认 1 位），此时主机和目标都不会驱动线路。

表 256. ACK 响应（3 位）

位	名称	说明
0..2	ACK	001：FAULT 010：WAIT 100：OK

仅当发生 READ 事务或者接收到 WAIT 或 FAULT 确认时，ACK 响应后才必须是转换时间。

表 257. DATA 传输（33 位）

位	名称	说明
0..31	WDATA 或 RDATA	写入或读取数据
32	奇偶校验	32 个数据位的单奇偶校验

仅当发生 READ 事务时，DATA 传输后才必须是转换时间。

40.8.3 SW-DP 状态机（复位、空闲状态、ID 代码）

SW-DP 的状态机有一个用于标识 SW-DP 的内部 ID 代码。该代码符合 JEP-106 标准。此 ID 代码是默认的 ARM® 代码，设置为 **0x5BA02477**（相当于带 FPU 的 Cortex®-M7）。

注： *请注意，在目标读取此 ID 代码前，SW-DP 状态机是不工作的。*

- 在上电复位后、DP 从 JTAG 切换到 SWD 后或者线路处于高电平超过 50 个周期后，SW-DP 状态机处于复位状态。
- 如果在复位状态后线路处于低电平至少两个周期，SW-DP 状态机处于空闲状态。
- 复位状态后，该状态机**必须**首先进入空闲状态，然后对 DP-SW ID CODE 寄存器执行读访问。否则，目标将在另一个事务上发出 FAULT 确认响应。

有关 SW-DP 状态机的更多详细信息，请参见带 FPU 的 *Cortex®-M7 TRM* 和《CoreSight 组件技术参考手册》。

40.8.4 DP 和 AP 读/写访问

- 不延迟对 DP 的读访问：可以立即发送目标响应（如果 ACK=OK），也可以延迟发送目标响应（如果 ACK=WAIT）。
- 延迟对 AP 的读访问。这意味着会在下次传输时返回访问结果。如果要执行的下次访问不是 AP 访问，则必须读取 DP-RDBUFF 寄存器来获取结果。
  每次进行 AP 读访问或 RDBUFF 读请求时都会更新 DP-CTRL/STAT 寄存器的 READOK 标志，以便了解 AP 读访问是否成功。
- SW-DP 有写缓冲区（用于 DP 或 AP 写入），这样即使在其它操作仍未完成时，也可以接受写入操作。如果写缓冲区已满，则目标确认响应为 WAIT。但 IDCODE 读取、CTRL/STAT 读取或 ABORT 写入除外，这几项操作在写缓冲区已满时也会被接受。
- 由于存在异步时钟域 SWCLK 和 HCLK，因此写操作后（奇偶校验位后）还需要两个额外的 SWCLK 周期，以使写入操作在内部生效。应在将线路驱动为低电平时（空闲状态）应用这些周期。
  在写 CTRL/STAT 寄存器以提出一个上电请求时，这一点特别重要。否则下一个操作（在内核上电后才有效的操作）会立即执行，这将导致失败。

40.8.5 SW-DP 寄存器

当 APnDP=0 时能够访问这些寄存器

表 258. SW-DP 寄存器

A(3:2)	R/W	SELECT 寄存器的 CTRLSEL 位	寄存器	注释
00	读	-	IDCODE	制造商代码未设置为 ST 代码。0x5BA02477（标识 SW-DP）
00	写	-	ABORT	-
01	读/写	0	DP-CTRL/STAT	目的： – 请求系统或调试上电 – 配置 AP 访问的传输操作 – 控制比较和验证操作 – 读取一些状态标志（上溢和上电确认）
01	读/写	1	WIRE CONTROL	用于配置物理串行端口协议（如转换时间的持续时间）
10	读	-	READ RESEND	允许从已损坏的调试软件传输中恢复读取数据，无需重复执行原始 AP 传输
10	写	-	SELECT	用于选择当前访问端口和活动的 4 字寄存器窗口
11	读/写	-	READ BUFFER	由于已发出 AP 访问，因此该读缓冲区非常有用（在执行下个 AP 事务时提供读取 AP 请求的结果） 此读取缓冲区捕获 AP 中的数据，显示为前一次读取的结果，无需启动新操作

40.8.6 SW-AP 寄存器

当 APnDP=1 时能够访问这些寄存器

有多个 AP 寄存器（请参见 AHB-AP），这些寄存器按以下组合进行寻址：

- 移位值 A[3:2]
- DP SELECT 寄存器的当前值

40.9 AHB-AP（AHB 访问端口）——对 JTAG-DP 和 SW-DP 同时有效

特性：

- 系统访问与处理器状态无关。
- SW-DP 或 JTAG-DP 访问 AHB-AP。
- AHB-AP 是总线矩阵内的 AHB 主设备。因此，它可以访问所有数据总线（Dcode 总线、系统总线、内部和外部 PPB 总线），但不能访问 ICode 总线。
- 支持位寻址的传输。
- 旁路 FPB 的 AHB-AP 传输。

32 位 AHP-AP 寄存器的地址宽度为 6 位（最多达 64 字或 256 字节），其中包括：

c) 位 [7:4] = DP SELECT 寄存器的位 [7:4] APBANKSEL

d) 位 [3:2] = SW-DP 的 35 位数据包请求的 2 个地址位 A(3:2)。

带 FPU 的 Cortex®-M7 的 AHB-AP 包括 9 个 32 位寄存器：

表 259. Cortex®-M7 带 FPU 的 Cortex®-M7 的 AHB-AP 寄存器

偏移地址	寄存器名	注释
0x00	AHB-AP 控制和状态字	配置和控制通过 AHB 接口的传输（大小、hprot、当前传输的状态、地址递增类型）
0x04	AHB-AP 传输地址	-
0x0C	AHB-AP 数据读/写	-
0x10	AHB-AP 分组数据 0	直接访问 4 个相连的字而不用重写访问地址。
0x14	AHB-AP 分组数据 1	
0x18	AHB-AP 分组数据 2	
0x1C	AHB-AP 分组数据 3	
0xF8	AHB-AP 调试 ROM 地址	调试接口的基址
0xFC	AHB-AP ID 寄存器	-

有关更多详细信息，请参见 *带 FPU 的 Cortex®-M7 TRM*。

40.10 内核调试

通过内核调试寄存器调试内核。对这些寄存器的调试访问通过*先进高性能总线* (AHB-AP) 端口进行。处理器可通过内部*专用外设总线* (PPB) 直接访问这些寄存器。

它由 4 个寄存器组成：

表 260. 内核调试寄存器

寄存器	说明
DHCSR	32 位调试控制和状态寄存器。 此寄存器提供有关处理器状态的信息，能够使内核进入调试停止状态并提供处理器步进功能。
DCRSR	17 位调试内核寄存器选择器寄存器： 此寄存器选择需要进行读写操作的内核寄存器。
DCRDR	32 位调试内核寄存器数据寄存器： 此寄存器保存在寄存器与 DCRSR（选择器）寄存器选择的处理器之间读取和写入的数据。
DEMCR	32 位调试异常和监视控制寄存器： 此寄存器提供向量捕获和调试监视控制。此寄存器包含一个名为 ***TRCENA*** 的位，该位用于使能 TRACE 功能。

注： ***重要提示：****这些寄存器在系统复位时不复位。它们只能通过上电复位来复位。*

有关更多详细信息，请参见*带 FPU 的 Cortex®-M7 TRM*。

为了在复位后立即使内核进入调试状态，必须：

- 使能调试和异常监视控制寄存器的位 0 (VC_CORRESET)
- 使能调试停止控制和状态寄存器的位 0 (C_DEBUGEN)。

40.11 调试主机在系统复位状态下建立连接的功能

STM32F75xxx 和 STM32F74xxx MCU 的复位系统包含以下复位源：

- POR（上电复位），在每次上电时将执行复位
- 内部看门狗复位
- 软件复位
- 外部复位

带 FPU 的 Cortex®-M7 将调试部分的复位（通常为 PORRESETn）和其它部分 (SYSRESETn) 的复位分开。

这样，调试主机便能够在系统复位期间建立连接，对内核调试寄存器进行编程，以在获取复位向量时停止内核。随后，调试主机释放系统复位，内核将立即停止并且不执行任何指令。此外，还可以在内核处于复位状态下时配置调试特性。

注： *强烈建议调试主机在系统复位状态下建立连接（在复位向量处设置断点）。*

40.12 FPB (Flash patch breakpoint)

在 Cortex-M 架构中，FPB 单元通常可以：

- 实现硬件断点
- 用系统区域的代码和数据取代代码区域的代码和数据。利用此功能可以修正位于代码存储空间中的软件错误。

软件补丁和硬件断点都必须单独使用。

但 Pelican FPB 中存在一些主要变化：

- 不再支持 Flash 指令（没有 FP_REMAP 寄存器）
- 所有比较器均可用于指令地址（最多 8 个指令断点）
- 增强了用于断点比较器的编程器模型，以在完整地址范围内支持硬件断点。

40.13 DWT（数据观察点触发）

DWT 单元由四个比较器组成。这些比较器可配置为：

- 硬件观察点或
- ETM 的触发或
- PC 采样器或
- 数据地址采样器

DWT 还能以某种方式提供一些概要信息。为此，可访问某些计数器以获得以下数据：

- 时钟周期
- 分支指令
- 存取单元操作
- 睡眠周期
- CPI（每条指令的执行时间）
- 中断开销

40.14 ITM（指令跟踪宏单元）

40.14.1 一般说明

ITM 是应用驱动的跟踪源，支持 *printf* 类的调试以跟踪*操作系统* (OS) 和应用程序事件，并发布诊断系统信息。ITM 以数据包形式发送跟踪信息，这些信息包括：

- **软件跟踪。**软件可以直接写入 ITM 激励寄存器以发布包信息。
- **硬件跟踪。**DWT 生成数据包，然后 ITM 发送这些数据包。
- **时间戳。**发送与数据包相关的时间戳。ITM 包含 21 位计数器，用于生成时间戳。计数器由带 FPU 的 Cortex®-M7 的时钟或*串行线查看器* (SWV) 输出的位时钟驱动。

ITM 发送的数据包输出到 TPIU（跟踪端口接口单元）。TPIU 的格式化器添加一些额外的数据包（请参见 TPIU），然后将整个数据包序列输出到调试主机。

在设置或使用 ITM 前，必须使能调试异常和监视控制寄存器的位 TRCEN。

40.14.2 时间戳数据包、同步和溢出数据包

时间戳数据包会对时间戳信息、通用控制和同步进行编码。它使用 21 位时间戳计数器（带可能的预分频器），该计数器在每次发送时间戳数据包时复位。此计数器可由 CPU 时钟或 SWV 时钟驱动。

同步数据包由 6 个字节组成，等于 0x80_00_00_00_00_00，以 00 00 00 00 00 80 形式（首先发送 LSB）发送到 TPIU。

同步数据包是时间戳数据包的控制信号。它在每次触发 DWT 时发送。

为此，DWT 必须配置为触发 ITM：必须将 DWT 控制寄存器的位 CYCCNTENA（位 0）置 1。此外，还必须将 ITM 跟踪控制寄存器的位 2 (SYNCENA) 置 1。

注： *如果未将 SYNENA 位置 1，DWT 产生给 TPIU 的同步触发，将仅发送 TPIU 同步数据包，不发送 ITM 同步数据包。*

溢出数据包是一种特殊的时间戳数据包，用于指示已写入数据但 FIFO 已满。

表 261. 主要的 ITM 寄存器

地址	寄存器	详细信息
@E0000FB0	ITM 锁定访问	写入 0xC5ACCE55 以解锁对其它 ITM 寄存器的写访问
@E0000E80	ITM 跟踪控制	位 31-24 = 始终为 0
		位 23 = 忙碌
		位 22-16 = 7 位 ATB ID，用于标识跟踪数据源
		位 15-10 = 始终为 0
		位 9:8 = TSPrescale = 时间戳预分频器
		位 7-5 = 保留
		位 4 = SWOENA = 使能 SWV 行为（由 SWV 时钟驱动时间戳计数器）
		位 3 = DWTENA：使能 DWT 激励
		位 2 = SYNCENA：此位必须置为 1 以使能 DWT，进而生成同步触发，这样 TPIU 便可发送同步数据包
		位 1 = TSENA（时间戳使能）
		位 0 = ITMENA：ITM 的全局使能位
@E0000E40	ITM 跟踪特权	位 3：置 1 以使能跟踪端口 31:24
		位 2：置 1 以使能跟踪端口 23:16
		位 1：置 1 以使能跟踪端口 15:8
		位 0：置 1 以使能跟踪端口 7:0
@E0000E00	ITM 跟踪使能	每个位使能相应的激励端口以产生跟踪
@E0000000-E000007C	激励端口寄存器 0-31	在要跟踪的选定激励端口上写入 32 位数据（32 个可用位）

配置示例

从 TPIU 输出一个简单的数值：

- 通过配置 DBGMCU_CR 来配置 TPIU 并分配 TRACE I/O（请参见*第 40.17.2 节：TRACE 引脚分配*和*第 40.16.3 节：MCU 调试配置寄存器*）
- 将 0xC5ACCE55 写入 ITM 锁定访问寄存器，以解锁对 ITM 寄存器的写保护
- 将 0x00010005 写入 ITM 跟踪控制寄存器，以使能 ITM（使能同步，ATB ID 不为 0x00）
- 将 0x1 写入 ITM 跟踪使能寄存器，以使能激励端口 0
- 将 0x1 写入 ITM 跟踪特权寄存器，以取消屏蔽激励端口 7:0
- 将要输出的值写入激励端口寄存器 0：此操作可用软件完成（使用 printf 函数）

40.15 ETM（嵌入式跟踪宏单元）

40.15.1 一般说明

ETM 能够重建程序执行。使用“数据观察点和跟踪”(DWT) 组件或“指令跟踪宏单元”(ITM) 跟踪数据，而使用“嵌入式跟踪宏单元”(ETM) 跟踪指令。

ETM 以数据包形式发送信息，由内置资源触发。这些资源必须单独编程，并且使用“触发事件寄存器”(0xE0041008) 选择触发源。事件可以是简单事件（地址比较器中的地址匹配），也可以是 2 个事件之间的逻辑运算结果。触发源是 DWT 模块的四个比较器之一，可以监视以下事件：

- 时钟周期匹配
- 数据地址匹配

有关触发资源的更多信息，请参见*第 40.13 节：DWT（数据观察点触发）*。

ETM 发送的数据包输出到 TPIU（跟踪端口接口单元）。TPIU 会添加一些额外的数据包（请参见*第 40.17 节：Pelican TPIU（跟踪端口接口单元）*），然后将整个数据包序列输出到调试主机。

注： *N.B：Cortex-M7 ETM 兼容 ARM ETM 架构 v4，编程模型并不向后兼容 Cortex-M4 ETM 架构（ETM 架构 v3.5）。*

40.15.2 信号协议和数据包类型

本部分将在第 6 章的“ETM v4 架构规范 (IHI0064B)”中介绍。

40.15.3 主要的 ETM 寄存器

有关寄存器的更多信息，请参见《Pelican ETM 技术参考手册》(DDI0494-2a) 和《ETM v4 架构规范》(IHI0064B)。

40.15.4 配置示例

从 TPIU 输出一个简单的数值：

- 配置跟踪 I/O：使能 STM32F75xxx 和 STM32F74xxx 调试配置寄存器 (DBGMCU_CR) 中的 TRACE_CLKINEN
- 写入 @ E000EDFC 01000000； SCS：将 TRCENA 置 1，否则无法访问跟踪寄存器
- 写入 @ E00400F0 00000000； TPIU：选择 SYNC PORT 模式
- 写入 @ E0040004 00000008； TPIU：选择 TPIU 端口大小 = 4
- 写入 @ E0001020 002002CA； WT：PC MATCH 比较器 (PC=0x2002CA)
- 写入 @ E0001024 00000000； DWT：比较器不应用任何屏蔽
- 写入 @ E0001028 00000008； DWT：匹配时 ETM 触发 PC

ETM：

- 写入 @ E0041004 00000000；禁止 ETM
- 读取 @ E004100C 00000003； ETM 应处于空闲状态
- 写入 @ E0041040 00000002；指令跟踪源 ID = 0x2
- 写入 @ E0041080 00000001； ViewInst 使能事件的资源“始终为真”
- 写入 @ E004108C 000000FF；启动时选择处理器比较器：pc_match0（=>DWT 匹配）
- 写入 @ E0041004 00000001；使能 ETM

40.16 MCU 调试组件 (DBGMCU)

MCU 调试组件帮助调试器为以下各项提供支持：

- 低功耗模式
- 断点期间的定时器、看门狗、 I2C 和 bxCAN 的时钟控制
- 跟踪引脚分配的控制

40.16.1 对低功耗模式的调试支持

要进入低功耗模式，必须执行指令 WFI 或 WFE。

MCU 支持多个低功耗模式，这些模式可以禁止 CPU 时钟或降低 CPU 功耗。

内核不允许在调试会话期间关闭 FCLK 或 HCLK。由于调试期间需要使用它们进行调试连接，因此其必须保持激活状态。MCU 集成了特殊方法，允许用户在低功耗模式下调试软件。

为此，调试主机首先必须设置一些调试配置寄存器，以更改低功耗模式行为：

- 在睡眠模式下，调试主机必须事先将 DBGMCU_CR 寄存器的 DBG_SLEEP 位置 1。这样便可为 HCLK 和 FCLK 提供相同的时钟（之前配置的系统时钟）。
- 在停止模式下，调试主机必须事先将 DBG_STOP 位置 1。这样便可使能内部 RC 振荡器时钟，以在停止模式下为 FCLK 和 HCLK 提供时钟。

40.16.2 对定时器、看门狗、bxCAN 和 I²C 的调试支持

断点期间，必须选择定时器和看门狗的计数器的行为方式：

- 在产生断点时，计数器继续计数。例如，当 PWM 控制电机时，通常需要这种方式。
- 在产生断点时，计数器停止计数。用于看门狗时需要这种方式。

对于 bxCAN，用户可以选择在断点期间阻止更新接收寄存器。

对于 I²C，用户可以选择在断点期间阻止 SMBUS 超时。

40.16.3 MCU 调试配置寄存器

Debug MCU configuration register

可利用此寄存器在调试状态下配置 MCU。其中涉及：

- 低功耗模式支持
- 定时器和看门狗计数器支持
- bxCAN 通信支持
- TRACE 引脚分配

此 DBGMCU_CR 映射到外部 PPB 总线的地址 0xE0042004。

它通过 PORESET（而非系统复位）异步复位。可在系统复位状态下用调试器写入该寄存器。

如果调试主机不支持这些功能，仍然可以通过用户软件写入这些寄存器。

40.16.4 DBGMCU_CR 寄存器

DBGMCU_CR register

地址：0xE004 2004

仅支持 32 位访问

POR 复位：0x0000 0000（不会被系统复位信号复位）

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.	Res.	TRACE_MODE [1:0]		TRACE_CLKIN EN	Res.	Res.	DBG_STAND BY	DBG_STOP	DBG_SLEEP
								rw	rw	rw			rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:5 **TRACE_MODE[1:0] 和 TRACE_CLKINEN**：跟踪时钟和引脚分配控制 (Trace clock and pin assignment control)

– 当 TRACE_CLKINEN=0 时：

TRACE_MODE=xx：禁止 TRACE 输出（同步和异步）

– 当 TRACE_CLKINEN=1 时：

– TRACE_MODE[1:0]=00：在引脚级别使能异步跟踪接口（使用串行线模式时只能在 TDO 引脚上实现 TRACESWO）/使能同步跟踪接口

– TRACE_MODE[1:0] 不等于 00 时：禁止异步跟踪接口/使能同步跟踪接口

位 4:3 保留，必须保持复位值。

位 2 **DBG_STANDBY**：调试待机模式 (Debug Standby mode)

0：（FCLK=关闭，HCLK=关闭）将整个数字部分断电。

从软件角度来看，退出待机模式相当于复位（指示 MCU 从待机状态恢复的几个状态位除外）

1：（FCLK=开启，HCLK=开启）在这种情况下，数字部分不断电，FCLK 和 HCLK 由仍保持激活状态的内部 RC 振荡器提供。此外，MCU 会在待机模式期间产生系统复位，这样退出待机模式相当于复位

位 1 **DBG_STOP**：调试停止模式 (Debug Stop mode)

0：（FCLK=关闭，HCLK=关闭）在停机模式下，时钟控制器禁止所有时钟（包括 HCLK 和 FCLK）。退出停机模式时，时钟配置与复位后的时钟配置相同（CPU 时钟由 8 MHz 内部 RC 振荡器 (HSI) 提供）。因此，软件必须重新编程时钟控制器以使能 PLL 和晶振等。

1：（FCLK=开启，HCLK=开启）在这种情况下，进入停机模式时，FCLK 和 HCLK 由在停机模式下仍保持激活状态的内部 RC 振荡器提供。退出停机模式时，软件必须重新编程时钟控制器以使能 PLL 和晶振等（操作步骤与在 DBG_STOP=0 时相同）

位 0 **DBG_SLEEP**：调试睡眠模式 (Debug Sleep mode)

0：（FCLK=开启，HCLK=关闭）在睡眠模式下，FCLK 由原先在 HCLK 禁止时通过软件配置好的系统时钟驱动。

在睡眠模式下，时钟控制器配置不复位，保持先前设置的状态。因此，退出睡眠模式时，软件不需要重新配置时钟控制器。

1：（FCLK=开启，HCLK=开启）在这种情况下，进入睡眠模式时，将为 HCLK 和 FCLK 馈送相同的时钟（软件先前配置的系统时钟）。

40.16.5 MCU APB1 调试冻结寄存器 (DBGMCU_APB1_FZ)

Debug MCU APB1 freeze register

在调试状态下，使用 DBGMCU_APB1_FZ 寄存器配置 MCU。它涉及 APB2 外设。它映射到外部 PPB 总线的地址 0xE004 2008

它通过 POR（而非系统复位）异步复位。可在系统复位状态下用调试器写入该寄存器。

地址：0xE004 2008

仅支持 32 位访问。

上电复位 (POR)：0x0000 0000（不会被系统复位信号复位）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	DBG_CAN2_STOP	DBG_CAN1_STOP	DBG_I2C4_SMBUS_TIMEOUT	DBG_I2C3_SMBUS_TIMEOUT	DBG_I2C2_SMBUS_TIMEOUT	DBG_I2C1_SMBUS_TIMEOUT	Res.	Res.	Res.	Res.	Res.
					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.	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	DBG_LPTIM1_STOP	DBG_TIM14_STOP	DBG_TIM13_STOP	DBG_TIM12_STOP	DBG_TIM7_STOP	DBG_TIM6_STOP	DBG_TIM5_STOP	DBG_TIM4_STOP	DBG_TIM3_STOP	DBG_TIM2_STOP
				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 **DBG_CAN2_STOP**：内核停止时停止调试 CAN2 (Debug CAN2 stopped when Core is halted)
0：行为方式与正常模式下相同
1：冻结 CAN2 接收寄存器

位 25 **DBG_CAN1_STOP**：内核停止时停止调试 CAN2 (Debug CAN2 stopped when Core is halted)
0：行为方式与正常模式下相同
1：冻结 CAN2 接收寄存器

位 24 **DBG_I2C4_SMBUS_TIMEOUT**：内核停止时停止 SMBUS 超时模式 (SMBUS timeout mode stopped when Core is halted)
0：行为方式与正常模式下相同
1：冻结 SMBUS 超时

位 23 **DBG_I2C3_SMBUS_TIMEOUT**：内核停止时停止 SMBUS 超时模式 (SMBUS timeout mode stopped when Core is halted)
0：行为方式与正常模式下相同
1：冻结 SMBUS 超时

位 22 **DBG_I2C2_SMBUS_TIMEOUT**：内核停止时停止 SMBUS 超时模式 (SMBUS timeout mode stopped when Core is halted)
0：行为方式与正常模式下相同
1：冻结 SMBUS 超时

位 21 **DBG_I2C1_SMBUS_TIMEOUT**：内核停止时停止 SMBUS 超时模式 (SMBUS timeout mode stopped when Core is halted)
0：行为方式与正常模式下相同
1：冻结 SMBUS 超时

位 20:13 保留，必须保持复位值。

位 12 **DBG_IWDG_STOP**：内核停止时停止调试独立看门狗 (Debug independent watchdog stopped when core is halted)

0：即使内核停止，独立看门狗计数器时钟仍继续工作
1：内核停止时，独立看门狗计数器时钟停止

位 11 **DBG_WWDG_STOP**：内核停止时停止调试窗口看门狗 (Debug Window Watchdog stopped when Core is halted)

0：即使内核停止，窗口看门狗计数器时钟仍继续工作
1：内核停止时，窗口看门狗计数器时钟停止

位 10 **DBG_RTC_STOP**：内核停止时停止 RTC (RTC stopped when Core is halted)

0：即使内核停止，RTC 计数器时钟仍继续工作
1：内核停止时，RTC 计数器时钟停止

位 9 **DBG_LPTIM1_STOP**：内核停止时 LPTIM1 计数器停止 (LPTIM1 counter stopped when core is halted)

0：即使内核停止，仍然馈送 LPTIM1 计数器时钟
1：内核停止时，LPTIM1 计数器时钟停止

位[8:0] **DBG_TIMx_STOP**：内核停止时 TIMx 计数器停止 (TIMx counter stopped when core is halted) (x=2..7, 12..14)

0：即使内核停止，仍然馈送相关定时器计数器的时钟
1：内核停止时停止相关定时器计数器的时钟

40.16.6 MCU APB2 调试冻结寄存器 (DBGMCU_APB2_FZ)

Debug MCU APB2 Freeze register

在调试状态下，使用 DBGMCU_APB2_FZ 寄存器配置 MCU。它涉及 APB2 外设。

此寄存器映射到外部 PPB 总线的地址 0xE004 200C。

它通过 POR（而非系统复位）异步复位。可在系统复位状态下用调试器写入该寄存器。

地址：0xE004 200C

仅支持 32 位访问。

POR：0x0000 0000（不会被系统复位信号复位）

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.	DBG_TIM11_STOP	DBG_TIM10_STOP	DBG_TIM9_STOP
													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.	DBG_TIM8_STOP	DBG_TIM1_STOP
														rw	rw

位 31:19 保留，必须保持复位值。

位 18:16 **DBG_TIMx_STOP**：内核停止时 TIMx 计数器停止 (TIMx counter stopped when core is halted) (x=9..11)

0：即使内核停止，仍然馈送相关定时器计数器的时钟
1：内核停止时停止相关定时器计数器的时钟

位 15:2 保留，必须保持复位值。

位 1 **DBG_TIM8_STOP**：内核停止时 TIM8 计数器停止 (TIM8 counter stopped when core is halted)
0：即使内核停止，仍然馈送相关定时器计数器的时钟
1：内核停止时停止相关定时器计数器的时钟

位 0 **DBG_TIM1_STOP**：内核停止时 TIM1 计数器停止 (TIM1 counter stopped when core is halted)
0：即使内核停止，仍然馈送相关定时器计数器的时钟
1：内核停止时停止相关定时器计数器的时钟

40.17 Pelican TPIU（跟踪端口接口单元）

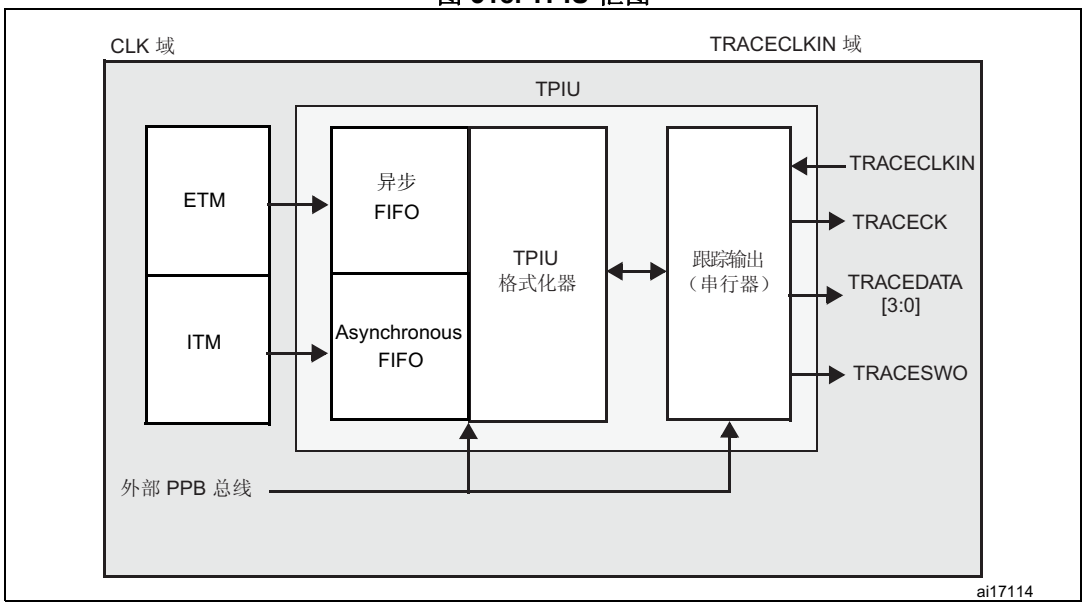
40.17.1 前言

TPIU 用作 ITM、ETM 和外部跟踪捕获器件中片上跟踪数据之间的桥接器。

输出数据流封装成跟踪源 ID，然后被*跟踪端口分析器* (TPA) 捕获。

内核内置专为低成本调试设计的简单 TPIU（由特殊版本的 CoreSight TPIU 组成）。

图 518. TPIU 框图



40.17.2 TRACE 引脚分配

- 异步模式

  异步模式需要 1 个额外引脚，并且适用于所有封装。仅在使用串线行模式时异步模式才可用（在 JTAG 模式下不可用）。

表 262. 异步 TRACE 引脚分配

TPIU 引脚名称	跟踪同步模式	
	类型	说明
TRACESWO	O	TRACE 异步数据输出

- 同步模式

  同步模式需要 2 到 6 个额外引脚，具体取决于所跟踪数据的长度，并且仅适用于较大型的封装。此外，同步模式在 JTAG 模式和串行模式下均可用，并可提供比异步跟踪更高的带宽输出能力。

表 263. 同步 TRACE 引脚分配

TPIU 引脚名称	跟踪同步模式	
	类型	说明
TRACECK	O	TRACE 时钟
TRACED[3:0]	O	TRACE 同步数据输出 可以是 1、2 或 4。

TPIU TRACE 引脚分配

默认情况下，不分配这些引脚。可通过将 ***MCU 调试组件配置寄存器***中的 TRACE_CLKINEN 和 TRACE_MODE 位置 1 来分配这些引脚。必须由调试主机来完成此配置。

此外，要分配的引脚数目取决于跟踪配置（异步跟踪或同步跟踪）。

- **异步模式：**需要 1 个额外引脚
- **同步模式：**需要 2 到 5 个额外引脚，具体数目取决于数据跟踪端口寄存器的数据长度（1、2 或 4）：
  - TRACECK
  - TRACED(0)（如果端口数据长度配置为 1、2 或 4）
  - TRACED(1)（如果端口数据长度配置为 2 或 4）
  - TRACED(2)（如果端口数据长度配置为 4）
  - TRACED(3)（如果端口数据长度配置为 4）

要分配 TRACE 引脚，调试主机必须对 MCU 调试配置寄存器 (DBGMCU_CR) 的位 TRACE_CLKINEN 和 TRACE_MODE[1:0] 进行编程。默认情况下不分配 TRACE 引脚。

此寄存器映射到外部 PPB 上，通过 PORESET（而非系统复位）复位。可在系统复位状态下用调试器写入该寄存器。

表 264. 灵活的 TRACE 引脚分配

DBGMCU_CR 寄存器		引脚用途	分配的 TRACE IO 引脚					
TRACE_CLKINEN	TRACE_MODE[1:0]		JTDO/TRACESWO	TRACECK	TRACED[0]	TRACED[1]	TRACED[2]	TRACED[3]
0	XX	无跟踪（默认状态）	已释放 (1)	-				
1	00	异步跟踪	TRACESWO	-	-	已释放（可用作 GPIO）		
1	不等于 00	同步跟踪 1 位(2)	已释放 (1)	TRACECK	TRACED[0]	-	-	-
1	不等于 00	同步跟踪 2 位(2)		TRACECK	TRACED[0]	TRACED[1]	-	-
1	不等于 00	同步跟踪 4 位(2)		TRACECK	TRACED[0]	TRACED[1]	TRACED[2]	TRACED[3]

1. 使用串行模式时，释放此引脚。但使用 JTAG 时，此引脚分配给 JTDO。
2. 通过 TPIU 寄存器的位 [3:0] 来选择当前端口大小。

注： *默认情况下，TPIU 的 TRACECLKIN 输入时钟接地。因此，将在 TRACE_CLKINEN 位置 1 后的两个时钟周期将该时钟分配给 HCLK。*

调试器必须通过对 TPIU 的 SPP_R（选择引脚协议）寄存器的 PROTOCOL[1:0] 位执行写操作来配置跟踪模式。

- PROTOCOL=00：跟踪端口模式（同步）
- PROTOCOL=01 或 10：串行线（曼彻斯特或 NRZ）模式（异步模式）。默认状态为 01

随后，它还会通过对 TPIU 的 CPSPS_R（当前同步端口大小寄存器）中的位 [3:0] 执行写操作来配置跟踪端口大小：

- 0x1 表示 1 个引脚（默认状态）
- 0x2 表示 2 个引脚
- 0x8 表示 4 个引脚

40.17.3 TPIU 格式化器

该格式化器的作用是构建包含跟踪数据的 128 位帧（这些跟踪数据可能来自 ETM 和 ITM），并以跟踪分析器级别在跟踪数据包与发送器之间建立关联。

格式化器协议以 16 字节帧形式输出数据：

- 七个字节的数据
- 八个字节的多用字节，包括：
  - 1 位 (LSB) 指示该字节是数据字节 (‘0) 还是 ID 字节 (‘1)。
  - 7 位 (MSB)，既可以是数据，也可用于更改源 ID 跟踪。
- 一个字节的辅助位，每个辅助位对应于八个多用字节中的一个：
  - 如果对应的字节是数据，则此位表示该数据的位 0。
  - 如果对应的字节是 ID 更改，则此位表示 ID 更改生效的时间。

注： *有关更多信息，请参见《ARM® CoreSight 架构规范 v1.0》(ARM® IHI 0029B)。*

40.17.4 TPIU 帧同步数据包

TPIU 可产生两种类型的同步数据包：

- 帧同步数据包（或全字同步数据包）

  它包含字：0x7F_FF_FF_FF（首先发送 LSB）。只要未使用 ID 源代码 0x7F，就不会在任何其它时间出现此序列。

  它会在各数据帧*之间*定期输出。

  在连续模式下，找到同步帧后，TPA 必须丢弃所有这些帧。
- 半字同步数据包

  它包含半字：0x7F_FF（首先发送 LSB）。

  它会在各数据帧*之间或内部*定期输出。

  这些数据包仅在连续模式下生成，能够使 TPA 检测到跟踪端口是否处于空闲模式（没有要捕获的 TRACE）。当 TPA 检测到这些数据包后，必须将其丢弃。

40.17.5 发送同步帧数据包

内核的 TPIU 中没有同步计数器寄存器。因此，只能通过 **DWT** 生成同步触发。请参见寄存器 DWT 控制寄存器（位 SYNCTAP[11:10]）和 DWT 当前 PC 采样器循环计数寄存器。

发送 TPIU 帧同步数据包 (0x7F_FF_FF_FF) 的条件：

- 每次 TPIU 复位释放后。此复位在 TRACECLKIN 时钟出现上升沿时同步释放。这意味着，当 DBGMCU_CFG 寄存器中的 TRACE_CLKINEN 位置 1 时传送此数据包。在这种情况下，字 0x7F_FF_FF_FF 后面不带任何格式化的数据包。
- 在每次触发 DWT 时（假设先前已配置 DWT）。存在两种情况：
  - 如果复位 ITM 的 SYNENA 位复位，则仅发送字 0x7F_FF_FF_FF，后面不带任何格式化的数据流。
  - 如果 ITM 的 SYNENA 位置 1，则 ITM 同步数据包将跟在 (0x80_00_00_00_00_00) 后面，并由 TPIU 格式化（添加跟踪源 ID）。

40.17.6 同步模式

跟踪数据输出大小可配置为 4 个、2 个或 1 个引脚：TRACED(3:0)。

输出时钟输出到调试器 (TRACECK)。

其中，TRACECLKIN 由内部驱动，并且仅在使用 TRACE 时连接到 HCLK。

注： *在此同步模式下，不需要提供稳定的时钟频率。*

TRACE I/O（包括 TRACECK）由 TRACLKIN 的上升沿驱动（等于 HCLK）。因此，TRACECK 的输出频率等于 HCLK/2。

40.17.7 异步模式

这是一种只使用 1 个引脚输出跟踪的低成本备用方案：该引脚是异步输出引脚 TRACESWO。显然，这种方案的带宽有限。

单 IO 跟踪模式通常适用于 ITM 跟踪输出。异步跟踪时还会禁止格式化器，因此无法合并 ETM 和 ITM 跟踪流。

使用 SW-DP 引脚时，TRACESWO 与 JTDO 复用。这样，所有 STM32F75xxx 和 STM32F74xxx 封装都可提供此功能。

此异步模式要求 TRACECLKIN 具有恒定的频率。对于标准 UART (NRZ) 捕获机制，需要 5% 的精度。曼彻斯特编码的容限最高可达 10%。

40.17.8 STM32F75xxx 和 STM32F74xxx 内的 TRACECLKIN 连接

在 STM32F75xxx 和 STM32F74xxx 中，此 TRACECLKIN 输入内部连接到 HCLK。这意味着，在异步跟踪模式下，应用程序应限制使用时间帧保证 CPU 频率的稳定。

注： ***重要提示：****使用异步跟踪时，了解以下内容非常重要：*

STM32F75xxx 和 STM32F74xxx MCU 的默认时钟是内部 RC 振荡器。复位状态下的频率与复位释放后的频率不同。原因是，由于系统复位状态下默认采用 RC 校准值，而在每次系统复位释放时会更新该 RC 校准值。

因此，跟踪端口分析器 (TPA) 在系统复位状态下不应使能跟踪（使用 TRACE_CLKINEN 位），原因是，在复位状态下的同步帧包的比特宽度与复位后的包不同。

40.17.9 TPIU 寄存器

仅当调试异常和监视控制寄存器 (DEMCR) 的位 TRCENA 置 1 时才能对 TPIU APB 寄存器进行读写操作。否则，这些寄存器将读为零（此位的输出会使能 TPIU 的 PCLK）。

表 265. 重要的 TPIU 寄存器

地址	寄存器	说明
0xE0040004	当前端口大小	允许选择跟踪端口大小： 位 0：端口大小 = 1 位 1：端口大小 = 2 位 2：端口大小 = 3，不支持 位 3：端口大小 = 4 只有 1 位必须置 1。默认情况下，端口大小为一位。(0x00000001)
0xE00400F0	选择引脚协议	允许选择跟踪端口协议： 位 1:0= 00：同步跟踪端口模式 01：串行线输出 - 曼彻斯特（默认值） 10：串行线输出 - NRZ 11：保留
0xE0040304	格式化器和刷新控制	位 31-9 = 始终为 0 位 8 = TrigIn = 始终为 1，表明已指示这些触发 位 7-4 = 始终为 0 位 3-2 = 始终为 0 位 1 = EnFCont。在同步跟踪模式下（选择引脚协议寄存器位 1:0=00），此位会强制置为 1：在连续模式下自动使能格式化器。在异步模式下（选择引脚协议寄存器位 1:0 <> 00），可写入该位以激活或取消激活格式化器。 位 0 = 始终为 0 产生的默认值为 0x102 **注意：**在同步模式下，由于不会在片外映射 TRACECTL 引脚，因此在连续模式下将始终使能格式化器。这样，格式化器便可插入一些控制数据包来标识跟踪数据包的源。
0xE0040300	格式化器和刷新状态	不适用于带 FPU 的 Cortex®-M7，始终读为 0x00000008

40.17.10 配置示例

- 将调试异常和监视控制寄存器 (DEMCR) 中的位 TRCENA 置 1
- 将所需值写入 TPIU 当前端口大小寄存器（对于 1 位端口大小，默认值为 0x1）
- 将 0x102 写入 TPIU 格式化器和刷新控制寄存器（默认值）
- 写入 TPIU 选择引脚协议以选择同步模式或异步模式。示例：0x2 表示异步 NRZ 模式（类似于 UART）
- 将 0x20 写入 DBGMCU 控制寄存器（位 IO_TRACEN），为异步模式分配 TRACE I/O。此时发送 TPIU 同步数据包 (FF_FF_FF_7F)
- 配置 ITM 并对 ITM 激励寄存器进行写操作以输出值

40.18 DBG 寄存器映射

下表对调试寄存器进行了汇总。

表 266. DBG 寄存器映射和复位值

地址	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
0xE004 2000	**DBGMCU_IDCODE**	REV_ID																Res.	Res.	Res.	Res.	DEV_ID											
	Reset value[1]	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
0xE004 2004	**DBGMCU_CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_TIM7_STOP	DBG_TIM6_STOP	DBG_TIM5_STOP	DBG_TIM8_STOP	DBG_I2C2_SMBUS_TIMEOUT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRACE_MODE [1:0]		TRACE_	Res.	Res.	DBG_STANDBY	DBG_STOP	DBG_SLEEP
	Reset value												0	0	0	0	0									0	0	0			0	0	0
0xE004 2008	**DBGMCU_APB1_FZ**	Res.	Res.	Res.	Res.	Res.	DBG_CAN2_STOP	DBG_CAN1_STOP	DBG_I2C4_SMBUS_TIMEOUT	DBG_I2C3_SMBUS_TIMEOUT	DBG_I2C2_SMBUS_TIMEOUT	DBG_I2C1_SMBUS_TIMEOUT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_LPTIM1_STOP	DBG_RTC_STOP	DBG_TIM14_STOP	DBG_TIM13_STOP	DBG_TIM12_STOP	DBG_TIM7_STOP	DBG_TIM6_STOP	DBG_TIM5_STOP	DBG_TIM4_STOP	DBG_TIM3_STOP	DBG_TIM2_STOP
	Reset value						0	0	0	0	0	0									0	0	0	0	0	0	0	0	0	0	0	0	0
0xE004 200C	**DBGMCU_APB2_FZ**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_TIM11_STOP	DBG_TIM10_STOP	DBG_TIM9_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_TIM8_STOP	DBG_TIM1_STOP
	Reset value														0	0	0															0	0

1. 复位值与产品有关。有关详细信息，请参见*第 40.6.1 节：MCU 器件 ID 代码*。

41 设备电子签名

电子签名存储在 Flash 区。可以使用 JTAG/SWD 或 CPU 对其进行读取。它包含出厂前编程的标识数据，这些标识数据允许用户固件或其它外部设备将其接口与 STM32F75xxx 和 STM32F74xxx 微控制器的特性自动匹配。

41.1 唯一设备 ID 寄存器（96 位）

Unique device ID register

唯一设备标识符最适合：

- 用作序列号（例如 USB 字符串序列号或其它终端应用程序）
- 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以提高 Flash 中代码的安全性
- 激活安全自举过程等

96 位的唯一设备标识符提供了一个对于任何设备和任何上下文都唯一的参考号码。用户永远不能改变这些位。

96 位的唯一设备标识符也可以以单字节/半字/字等不同方式读取，然后使用自定义算法连接起来。

基址：0x1FF0 F420

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	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U_ID(31:0)																															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **U_ID(31:0)**：31:0 唯一 ID 位

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
U_ID(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
U_ID(47:32)															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **U_ID(63:32)**：63:32 唯一 ID 位

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
U_ID(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
U_ID(79:64)															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **U_ID(95:64)**：*95:64 唯一 ID 位*

41.2 Flash 大小

基址：0x1FF0 F442

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
F_SIZE															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 15:0 F_ID(15:0)：Flash 大小 (Flash memory size)
此处的位字段指示以 KB 表示的设备 Flash 大小。
例如，0x0400 对应于 1024 KB。

41.3 封装数据寄存器

Package data register

基址：0x1FFF 7BF0

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.	PKG[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
					rw	rw	rw								

位 15:11 保留，必须保持复位值。

位 10:8 PGK[2:0]：封装类型
0x1xx：LQFP208 和 TFBGA216 封装
0x011：LQFP176 和 UFBGA176 封装
0x010：WLCSP143 和 LQFP144 封装
0x001：LQFP100 封装
0x000：预留

位 7:0 保留，必须保持复位值。

索引

A

ADC_CCR 426
ADC_CDR 428
ADC_CR1 415
ADC_CR2 417
ADC_CSR 425
ADC_DR 424
ADC_HTR 421
ADC_JDRx 424
ADC_JOFRx 420
ADC_JSQR 423
ADC_LTR 421
ADC_SMPR1 419
ADC_SMPR2 420
ADC_SQR1 422
ADC_SQR2 422
ADC_SQR3 423
ADC_SR 414

C

CAN_BTR 1170
CAN_ESR 1169
CAN_FA1R 1179
CAN_FFA1R 1179
CAN_FiRx 1180
CAN_FM1R 1177
CAN_FMR 1177
CAN_FS1R 1178
CAN_IER 1167
CAN_MCR 1161
CAN_MSR 1163
CAN_RDHxR 1176
CAN_RDLxR 1175
CAN_RDTxR 1175
CAN_RF0R 1166
CAN_RF1R 1167
CAN_RIxR 1174
CAN_TDHxR 1173
CAN_TDLxR 1173
CAN_TDTxR 1172
CAN_TIxR 1171
CAN_TSR 1164
CEC_CFGR 1465
CEC_CR 1463
CEC_IER 1470
CEC_ISR 1468
CEC_RXDR 1467
CEC_TXDR 1467
CRC_CR 283
CRC_DR 282
CRC_IDR 282
CRC_INIT 284
CRC_POL 284
CRYP_CR 534
CRYP_DIN 537
CRYP_DMACR 539
CRYP_DOUT 538
CRYP_IMSCR 539
CRYP_IV0LR 543
CRYP_IV0RR 543
CRYP_IV1LR 544
CRYP_IV1RR 544
CRYP_K0LR 541
CRYP_K0RR 541
CRYP_K1LR 542
CRYP_K1RR 542
CRYP_K2LR 542
CRYP_K2RR 542
CRYP_K3LR 542
CRYP_K3RR 543
CRYP_MISR 540
CRYP_RISR 540
CRYP_SR 536

D

DAC_CR 441
DAC_DHR12L1 445
DAC_DHR12L2 446
DAC_DHR12LD 447
DAC_DHR12R1 444
DAC_DHR12R2 445
DAC_DHR12RD 447
DAC_DHR8R1 445
DAC_DHR8R2 446
DAC_DHR8RD 448
DAC_DOR1 448
DAC_DOR2 448
DAC_SR 449
DAC_SWTRIGR 444
DBGMCU_APB2_FZ 1492, 1494
DBGMCU_CR 1491
DBGMCU_IDCODE 1479
DCMI_CR 462
DCMI_CWSIZE 469
DCMI_CWSTRT 469

DCMI_DR 470
DCMI_ESCR 468
DCMI_ESUR 468
DCMI_ICR 467
DCMI_IER 465
DCMI_MIS 466
DCMI_RIS 464
DCMI_SR 464
DMA_HIFCR 228
DMA_HISR 226
DMA_LIFCR 227
DMA_LISR 225
DMA_SxCR 229
DMA_SxFCR 234
DMA_SxM0AR 233
DMA_SxM1AR 234
DMA_SxNDTR 232
DMA_SxPAR 233

E

ETH_DMABMR 1437
ETH_DMACHRBAR 1450
ETH_DMACHRDR 1449
ETH_DMACHTBAR 1449
ETH_DMACHTDR 1449
ETH_DMAIER 1446
ETH_DMAMFBOCR 1448
ETH_DMAOMR 1444
ETH_DMARDLAR 1440
ETH_DMARPDR 1439
ETH_DMARSWTR 1448
ETH_DMASR 1441
ETH_DMATDLAR 1440
ETH_DMATPDR 1439
ETH_MACA0HR 1420
ETH_MACA0LR 1421
ETH_MACA1HR 1421
ETH_MACA1LR 1422
ETH_MACA2HR 1422
ETH_MACA2LR 1423
ETH_MACA3HR 1423
ETH_MACA3LR 1424
ETH_MACCR 1408
ETH_MACDBGR 1417
ETH_MACFCR 1414
ETH_MACFFR 1410
ETH_MACHTHR 1411
ETH_MACHTLR 1412
ETH_MACIMR 1420
ETH_MACMIIAR 1412
ETH_MACMIIDR 1413
ETH_MACPMTCSR 1416
ETH_MACRWUFFR 1416
ETH_MACSR 1419
ETH_MACVLANTR 1415
ETH_MMCCR 1424
ETH_MMCRFAECR 1429
ETH_MMCRFCECR 1428
ETH_MMCRGUFCR 1429
ETH_MMCRIMR 1426
ETH_MMCRIR 1425
ETH_MMCTGFCR 1428
ETH_MMCTGFMSCCR 1428
ETH_MMCTGFSCCR 1427
ETH_MMCTIMR 1427
ETH_MMCTIR 1426
ETH_PTPPPSCR 1437
ETH_PTPSSIR 1432
ETH_PTPTSAR 1435
ETH_PTPTSCR 1430
ETH_PTPTSHR 1433
ETH_PTPTSHUR 1434
ETH_PTPTSLR 1433
ETH_PTPTSLUR 1434
ETH_PTPTSSR 1436
ETH_PTPTTHR 1435
ETH_PTPTTLR 1436
EXTI_EMR 276
EXTI_FTSR 277
EXTI_IMR 276
EXTI_PR 278
EXTI_RTSR 277
EXTI_SWIER 278

F

FLITF_FCR 86
FLITF_FKEYR 84
FLITF_FOPTCR 87, 89
FLITF_FOPTKEYR 84
FLITF_FSR 85
FMC_BCR1..4 324
FMC_BTR1..4 327
FMC_BWTR1..4 329
FMC_ECCR 340
FMC_PATT 339
FMC_PCR 336
FMC_PMEM 338
FMC_SDCMR 354
FMC_SDCR1,2 351
FMC_SDRTR 356
FMC_SDSR 357
FMC_SDTR1,2 353

FMC_SR 337
FMPI2C_ISR 900

G

GPIOx_AFRH 196
GPIOx_AFRL 196
GPIOx_BSRR 194
GPIOx_IDR 193
GPIOx_LCKR 195
GPIOx_MODER 191
GPIOx_ODR 194
GPIOx_OSPEEDR 192
GPIOx_OTYPER 192
GPIOx_PUPDR 193

H

HASH_CR 556
HASH_CSRx 564
HASH_DIN 558
HASH_HR0 560
HASH_HR1 . . . . . 560, 562
HASH_HR2 561-562
HASH_HR3 561
HASH_HR4 561
HASH_IMR 562
HASH_SR 563
HASH_STR 559

I

I2C_CR1 891
I2C_CR2 894
I2C_ICR 902
I2C_ISR 900
I2C_OAR1 896
I2C_OAR2 897
I2C_PECR 903
I2C_RXDR 904
I2C_TIMEOUTR 899
I2C_TIMINGR 898
I2C_TXDR 904
I2Cx_CR2 894
IWDG_KR 796
IWDG_PR 797
IWDG_RLR 798
IWDG_SR 798
IWDG_WINR 799

L

LPTIM1_OR 791
LPTIM2_OR 792
LPTIMx_ARR 790
LPTIMx_CFGR 786
LPTIMx_CMP 790
LPTIMx_CNT 791
LPTIMx_CR 789
LPTIMx_ICR 784
LPTIMx_IER 785
LPTIMx_ISR 783

O

OTG_CID 1238
OTG_DAINT 1264
OTG_DAINTMSK 1264
OTG_DCFG 1256
OTG_DCTL 1258
OTG_DEACHINT 1267
OTG_DEACHINTMSK 1268
OTG_DIEPCTL0 1269
OTG_DIEPCTLx 1270
OTG_DIEPEMPMSK 1267
OTG_DIEPINTx 1276
OTG_DIEPMSK 1261
OTG_DIEPTSIZ0 1279
OTG_DIEPTSIZx 1281
OTG_DIEPTXF0 1234
OTG_DIEPTXFx 1242
OTG_DOEPCTL0 1272
OTG_DOEPCTLx 1274
OTG_DOEPINTx 1278
OTG_DOEPMSK 1262
OTG_DOEPTSIZ0 1280
OTG_DOEPTSIZx 1282
OTG_DSTS 1260
OTG_DTHRCTL 1266
OTG_DTXFSTSx 1282
OTG_DVBUSDIS 1265
OTG_DVBUSPULSE 1266
OTG_GAHBCFG 1218
OTG_GCCFG 1237
OTG_GI2CCTL 1236
OTG_GINTMSK 1229
OTG_GINTSTS 1224
OTG_GLPMCFG 1238
OTG_GOTGCTL 1214
OTG_GOTGINT 1217
OTG_GRSTCTL 1222
OTG_GRXFSIZ 1234
OTG_GRXSTSP 1232
OTG_GRXSTSR 1232
OTG_GUSBCFG 1219

OTG_HAINT 1246
OTG_HAINTMSK 1247
OTG_HCCHARx 1250
OTG_HCDMAx 1256
OTG_HCFG 1243
OTG_HCINTMSKx 1253
OTG_HCINTx 1252
OTG_HCSPLTx 1251
OTG_HCTSIZx 1255
OTG_HFIR 1244
OTG_HFNUM 1245
OTG_HNPTXFSIZ 1234
OTG_HNPTXSTS 1235
OTG_HPRT 1247
OTG_HPTXFSIZ 1242
OTG_HPTXSTS 1245
OTG_PCGCCTL 1283

P

PWR_CR 113
PWR_CSR 115, 117-118

Q

QUADSPI _PIR 385
QUADSPI _PSMAR 385
QUADSPI _PSMKR 384
QUADSPI_ABR 383
QUADSPI_AR 383
QUADSPI_CCR 381
QUADSPI_CR 374
QUADSPI_DCR 377
QUADSPI_DLR 380
QUADSPI_DR 384
QUADSPI_FCR 379
QUADSPI_LPTR 386
QUADSPI_SR 378

R

RCC_AHB1ENR 148
RCC_AHB1LPENR 157
RCC_AHB1RSTR 139
RCC_AHB2ENR 150
RCC_AHB2LPENR 160
RCC_AHB2RSTR 141
RCC_AHB3ENR 151
RCC_AHB3LPENR 161
RCC_AHB3RSTR 142
RCC_APB1ENR 152
RCC_APB1LPENR 161
RCC_APB1RSTR 143
RCC_APB2ENR 155
RCC_APB2LPENR 165
RCC_APB2RSTR 146
RCC_BDCR 167
RCC_CFGR 134
RCC_CIR 137
RCC_CR 131
RCC_CSR 168
RCC_PLLCFGR 132, 171-172
RCC_SSCGR 170
RNG_CR 506
RNG_DR 508
RNG_SR 507
RTC_ALRMAR 831
RTC_ALRMBR 832
RTC_ALRMBSSR 842
RTC_BKPxR 843
RTC_CALR 837
RTC_CR 825
RTC_DR 824
RTC_ISR 827
RTC_OR 843
RTC_PRER 829
RTC_SHIFTR 834
RTC_SSR 833
RTC_TR 823
RTC_TSDR 836
RTC_TSSSR 836
RTC_TSTR 835
RTC_WPR 833
RTC_WUTR 830

S

SDMMC_ARG 1129
SDMMC_CLKCR 1127
SDMMC_DCOUNT 1134
SDMMC_DCTRL 1133
SDMMC_DLEN 1132
SDMMC_DTIMER 1132
SDMMC_FIFO 1141
SDMMC_ICR 1136
SDMMC_MASK 1138
SDMMC_POWER 1127
SDMMC_RESPCMD 1130
SDMMC_RESPx 1131
SDMMC_STA 1135
SPIx_CR1 1006
SPIx_CR2 1008
SPIx_CRCPR 1012
SPIx_DR 1012
SPIx_I2SCFGR 1014

SPIx_I2SPR 1016
SPIx_RXCRCR 1013
SPIx_SR 1010
SPIx_TXCRCR 1013
SYSCFG_EXTICR1 201
SYSCFG_EXTICR2 202
SYSCFG_EXTICR3 202
SYSCFG_EXTICR4 203
SYSCFG_MEMRMP 199

T

TIM2_OR1 712-713
TIM3_OR1 713
TIMx_ARR 636, 709, 748, 757, 772
TIMx_BDTR 639
TIMx_CCER 633, 706, 746, 756
TIMx_CCMR1 . . . . . 628, 701, 743, 754
TIMx_CCMR2 . . . . . 631, 705
TIMx_CCMR3 643
TIMx_CCR1 . . . . . 637, 709, 748, 758
TIMx_CCR2 . . . . . 637, 710, 749
TIMx_CCR3 . . . . . 638, 710
TIMx_CCR4 . . . . . 638, 711
TIMx_CCR5 644
TIMx_CCR6 645
TIMx_CNT 636, 708, 747, 757, 772
TIMx_CR1 . . . . . 617, 692, 738, 751, 769
TIMx_CR2 . . . . . 618, 694, 770
TIMx_DCR 642, 711
TIMx_DIER 623, 697, 740, 752, 770
TIMx_DMAR 642, 712
TIMx_EGR 626, 700, 742, 753, 771
TIMx_PSC 636, 708, 747, 757, 772
TIMx_RCR 637
TIMx_SMCR 621, 695, 739
TIMx_SR 625, 699, 741, 752, 771

U

USART_CR1 . . . . . 1078, 1080
USART_DR 1084-1087
USART_SR 1081, 1083
USARTx_BRR 954
USARTx_CR1 945
USARTx_CR2 948
USARTx_CR3 951
USARTx_GTPR 955
USARTx_ICR 961
USARTx_ISR 958
USARTx_RDR 962
USARTx_RQR 957
USARTx_RTOR 956
USARTx_TDR 963

W

WWDG_CFR 805
WWDG_CR 804
WWDG_SR 805

42 修订历史

表 267. 文档版本历史

日期	版本	变更
2015 年 5 月 21 日	1	初始版本。
2015 年 7 月 21 日	2	**更新了 FMC 部分：** 更新了*表 81：可编程的 NAND Flash 访问参数*存储器建立时间和存储器保持最大值。 更新了 c7amba_fmc_V2_UserSpec： - 更新了*通用存储器空间时序寄存器 2..4 (FMC_PMEM)* 一节 MEMSEY、MEMHOLD 和 MEMHIZ 寄存器。 - 更新了*特性存储器空间时序寄存器 2..4 (FMC_PATT)* 一节 ATTSET、ATTHOLD 和 ATTHIZ 寄存器。 - 更新了*表 87：FMC 寄存器映射*和位字段的说明，增加了 [x:x]。 保留 FMC_SDCR2 位 13 至位 31。 - 更新了*SRAM/NOR-Flash 片选控制寄存器 1..4 (FMC_BCR1..4)* 一节 CPSIZE[2:0] 位的说明，增加了 011 配置。 **更新了 SYSARCHI 部分：** - 更新了*第 2.1.8 节：DMA 存储器总线*和*第 2.1.9 节：DMA 外设总线*，增加了“内部 Flash”。 - 更新了*第 2.1.6 节：CPU AHBS 总线*。 **更新了 Flash 部分：** - 更新了*第 3.3.1 节：Flash 构成*。 - 更新了*第 3.3.8 节：Flash 中断*，用 FLASH_CR 寄存器替换了 FLASH_SR 寄存器。 更新了 c7amba_ioport_UserSpec，删除了位字段和*表 22：GPIO 寄存器映射和复位值*中的 GPIO 端口位复位寄存器 (GPIOx_BRR) (x = A..K)。 更新了 c7amba_spi2s1_v3_x_UserSpec *第 32.1 节：简介*，隐藏了“全双工和”。 **更新了 LTDC 部分：** - 更新了所有寄存器，在寄存器位的说明以及寄存器映射中增加了 [x:x]，并加入了粗体字符格式。 - 更新了*第 18.4.2 节：层可编程参数*。 - 更新了*第 18.7.4 节：LTDC 总宽度配置寄存器 (LTDC_TWCR)* TOTALW[11:0] 位字段。 - 更新了*第 18.7.5 节：LTDC 全局控制寄存器 (LTDC_GCR)* 位 29 的说明。 更新了 USART 部分： - 更新了*第 31.5.13 节：智能卡模式*中有关 RTO 计数器启动的注。

表 267. 文档版本历史（续）

日期	版本	变更
2015 年 7 月 21 日	2（续）	- 更新了*表 173：USART 中断请求*，删除了线性接收器超时错误中的错误。 **更新了 PWR 部分：** 更新了：*表 14*、*表 15*、*表 17* 和*表 18*，增加了“没有中断（针对 WFI）和事件（针对 WFE）挂起”。 **更新了 RTC 部分：** - 更新了*编程唤醒定时器*一节的第 3 点：编程唤醒自动重载值。 - 更新了*第 29.6.4 节：RTC 初始化和状态寄存器 (RTC_ISR)* 位 2 “WUTWF：唤醒定时器写标志 (WUTWF: wakeup timer write flag)”的说明。 **更新了 DCMI 部分：** 更新了*第 17.1 节：DCMI 简介*。不支持输出模式。

表. 中文文档版本历史

日期	版本	变更
2015 年 9 月 11 日	1	中文初始版本

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。所有其他产品或服务名称均为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。

© 2015 STMicroelectronics - 保留所有权利