RM0430
参考手册

STM32F413/423 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 STM32F413/423 微控制器存储器与外设的完整信息。

STM32F413/423 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 Arm® 带 FPU 的 Cortex®-M4 内核的信息，请参见 Cortex®-M4 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：

- STM32F413/423xG/xH 数据手册
- STM32F3 和 STM32F4 系列带 FPU 的 Cortex®-M4 编程手册 (PM0214)，提供关于 Arm® 带 FPU 的 Cortex®-M4 的信息。

目录

表格索引

图片索引

1 文档约定

1.1 一般信息

STM32F413/423 器件具有 Arm®(a) Cortex®-M4 with FPU 内核

1.2 寄存器相关缩写词列表

寄存器说明中使用以下缩写词(b)：

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

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

b. 这是一个适用于 STM 微控制器的所有缩写词的详尽列表，其中一些缩写词可能未在当前文档中使用。

1.3 词汇表

本节简要介绍本文档中所用首字母缩略词和缩写词的定义：

- **字**：32 位数据。
- **半字**：16 位数据。
- **字节**：8 位数据。
- **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。

1.4 外设可用性

有关各型号产品的外设可用性及数量的信息，请参见特殊器件数据手册。

2 系统和存储器概述

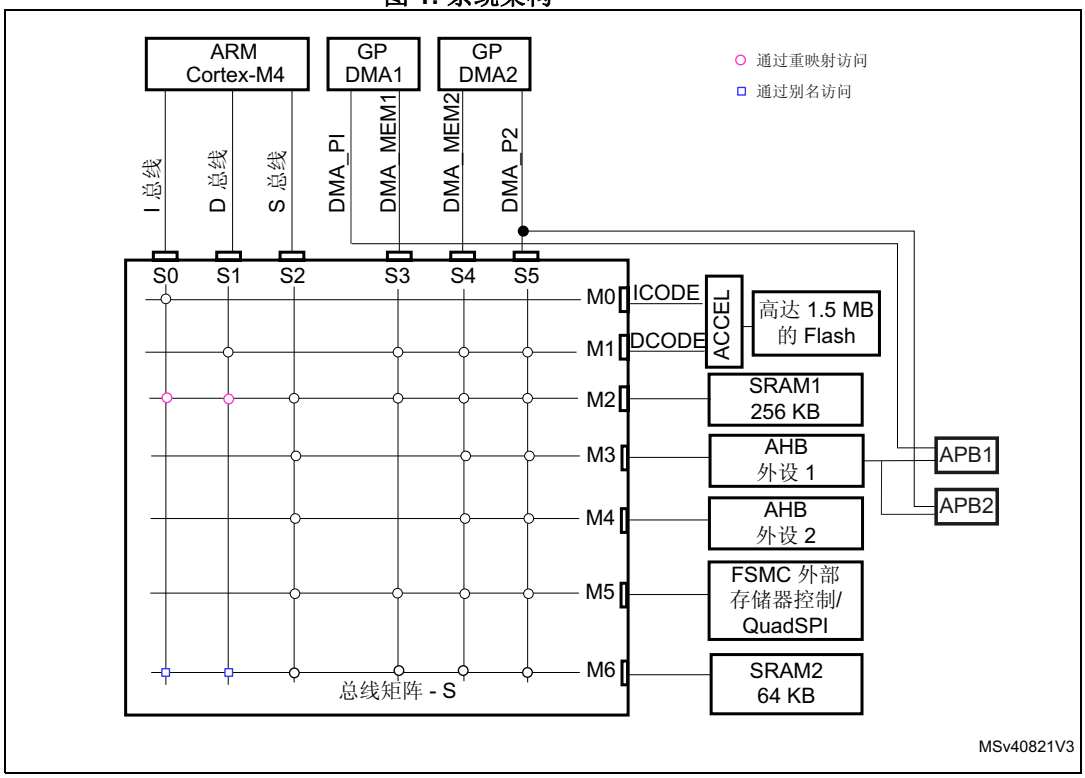
2.1 系统架构

STM32F413/423 的主系统由 32 位多层 AHB 总线矩阵构成，可实现以下部分的互连：

- 六条主控总线：
  - 带 FPU 的 Cortex®-M4 内核 I 总线、D 总线和 S 总线
  - DMA1 存储器总线
  - DMA2 存储器总线
  - DMA2 外设总线
- 七条被控总线：
  - 内部 Flash ICode 总线
  - 内部 Flash DCode 总线
  - 主内部 SRAM1 (256 KB)
  - 辅助内部 SRAM2 (64 KB)
  - AHB1 外设（包括 AHB-APB 总线桥和 APB 外设）
  - AHB2 外设
  - FSMC/QuadSPI

借助总线矩阵，可以实现主控总线到被控总线的访问，这样即使在多个高速外设同时运行期间，系统也可以实现并发访问和高效运行。此架构如*图 1* 所示。

图 1. 系统架构



2.1.1 I 总线

此总线用于将带 FPU 的 Cortex®-M4 内核的指令总线连接到总线矩阵。内核通过此总线获取指令。此总线访问的对象是包含代码的存储器（内部 Flash/SRAM1/SRAM2）。

2.1.2 D 总线

此总线用于将带 FPU 的 Cortex®-M4 的数据总线连接到总线矩阵。内核通过此总线进行立即数加载和调试访问。此总线访问的对象是包含代码或数据的存储器（内部 Flash/SRAM1/SRAM2）。

2.1.3 S 总线

此总线用于将带 FPU 的 Cortex®-M4 内核的系统总线连接到总线矩阵。此总线用于访问位于外设、SRAM1 或 SRAM2 中的数据。也可通过此总线获取指令（效率低于 ICode）。此总线访问的对象是内部 SRAM1/SRAM2、包括 APB 外设在内的 AHB1 外设、AHB2 外设和外部存储器（通过外设接口 FSMC 和 QUADSPI）。

2.1.4 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。此总线访问的对象是数据存储器：内部 Flash、内部 SRAM1/SRAM2 以及 S4 中包括 APB 外设在内的 AHB1/AHB2 外设。

2.1.5 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。此总线的访问对象是 AHB 和 APB 外设以及数据存储器：Flash 和内部 SRAM1/SRAM2。

2.1.6 总线矩阵

总线矩阵用于主控总线之间的访问仲裁管理。仲裁采用循环调度算法。

2.1.7 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 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。*

2.2 存储器组织结构

2.2.1 简介

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

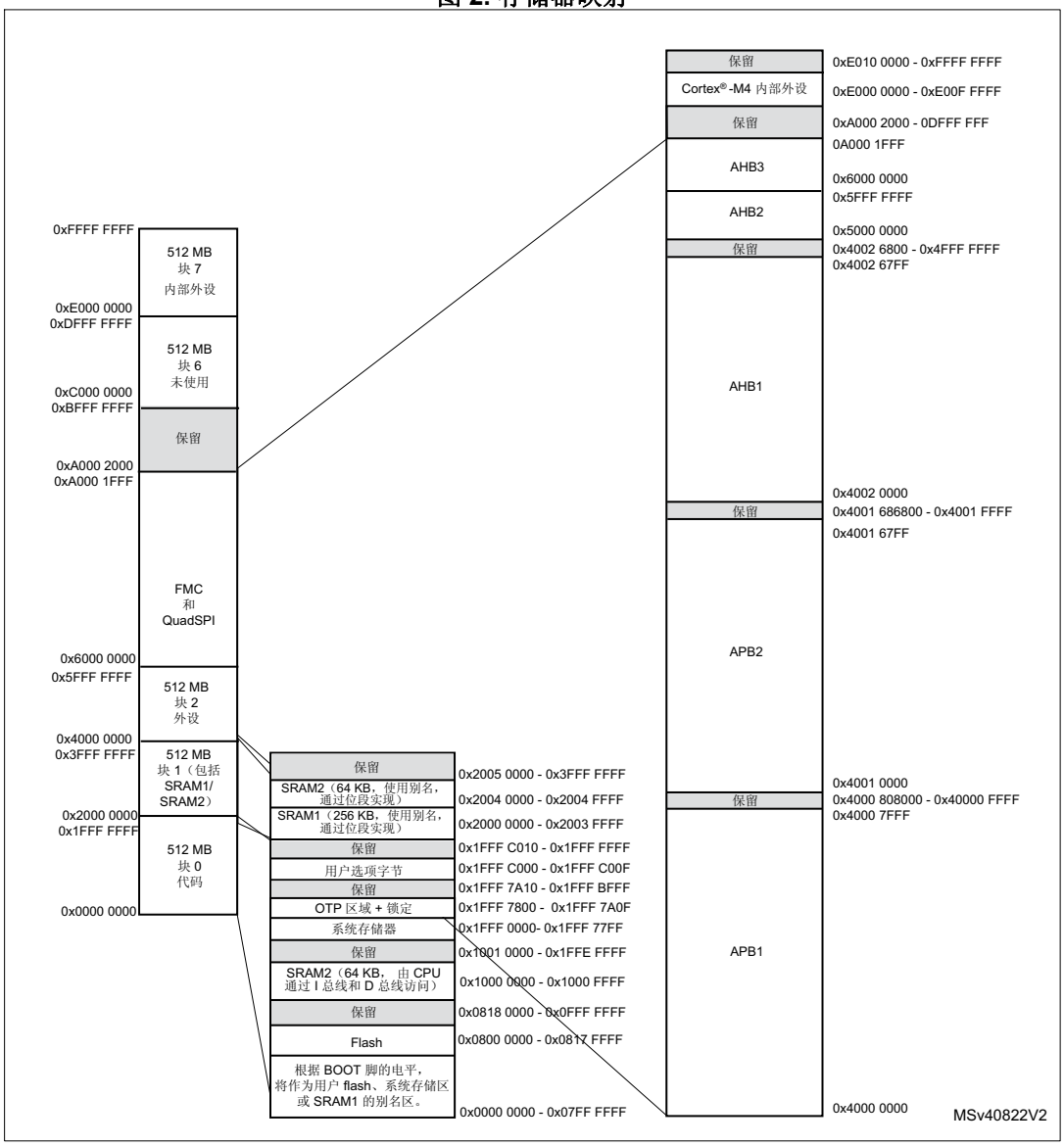
各字节按小端格式在存储器中编码。一个存储字单元中编号最低的字节被视为该字的最低有效字节，而编号最高的字节被视为最高有效字节。

可寻址的存储空间分为 8 个主要块，每个块为 512 MB。

可访问的地址空间取决于主控总线，有关详细信息，请参见*第 2 部分：存储器和总线架构*。

2.2.2 存储器映射和寄存器边界地址

图 2. 存储器映射



未分配给片上存储器和外设的所有存储器映射区域均视为“保留区”。有关可用存储器和寄存器区域的详细映射，请参见下表。

下表给出了器件中可用外设的边界地址。

表 1. 寄存器边界地址

总线	边界地址	外设
-	0xE010 0000 - 0xFFFF FFFF	保留
Cortex®-M4	0xE000 0000 - 0xE00F FFFF	Cortex-M4 内部外设
AHB3	0xA000 2000 - 0xDFFF FFFF	保留
	0xA000 1000 - 0xA000 1FFF	QuadSPI 控制寄存器
	0xA000 0000 - 0xA000 0FFF	FSMC 控制寄存器
	0x9000 0000 - 0x9FFF FFFF	QUADSPI
	0x7000 0000 - 0x08FFF FFFF	保留
	0x6000 0000 - 0x6FFF FFFF	FSMC
AHB2	0x5006 0C00 - 0x5FFF FFFF	保留
	0x5006 0800 0x5006 0BFF	RNG
	0x5006 0400 - 0x5006 07FF	保留
	0x5006 0000 - 0x5006 03FF	AES(1)
	0x5004 0000 - 0x5005 FFFF	保留
	0x5000 0000 - 0x5003 FFFF	USB OTG FS
AHB1	0x4002 6800 - 0x4FFF FFFF	保留
	0x4002 6400 - 0x4002 67FF	DMA2
	0x4002 6000 - 0x4002 63FF	DMA1
	0x4002 4000 - 0x4002 5FFF	保留
	0x4002 3C00 - 0x4002 3FFF	Flash 接口寄存器
	0x4002 3800 - 0x4002 3BFF	RCC
	0x4002 3400 - 0x4002 37FF	保留
	0x4002 3000 - 0x4002 33FF	CRC
	0x4002 2000 - 0x4002 2FFF	保留
	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. 寄存器边界地址（续）

总线	边界地址	外设
APB2	0x4001 6800 - 0x4001 FFFF	保留
	0x4001 6400 - 0x4001 67FF	DFSDM2
	0x4001 6000 - 0x4001 63FF	DFSDM1
	0x4001 5C00 - 0x4001 5FFF	保留
	0x4001 5800 - 0x4001 5BFF	SAI1
	0x4001 5400 - 0x4001 57FF	保留
	0x4001 5000 - 0x4001 53FF	SPI5/I2S5
	0x4001 4C00 - 0x4001 4FFF	保留
	0x4001 4800 - 0x4001 4BFF	TIM11
	0x4001 4400 - 0x4001 47FF	TIM10
	0x4001 4000 - 0x4001 43FF	TIM9
	0x4001 3C00 - 0x4001 3FFF	EXTI
	0x4001 3800 - 0x4001 3BFF	SYSCFG
	0x4001 3400 - 0x4001 37FF	SPI4/I2S4
	0x4001 3000 - 0x4001 33FF	SPI1/I2S1
	0x4001 2C00 - 0x4001 2FFF	SDIO
	0x4001 2400 - 0x4001 2BFF	保留
	0x4001 2000 - 0x4001 23FF	ADC1
	0x4001 1C00 - 0x4001 1FFF	UART10
	0x4001 1800 - 0x4001 1BFF	UART9
	0x4001 1400 - 0x4001 17FF	USART6
	0x4001 1000 - 0x4001 13FF	USART1
	0x4001 0800 - 0x4001 0FFF	保留
	0x4001 0400 - 0x4001 07FF	TIM8
	0x4001 0000 - 0x4001 03FF	TIM1

表 1. 寄存器边界地址（续）

总线	边界地址	外设
APB1	0x4000 8000 - 0x4000 FFFF	保留
	0x4000 7C00 - 0x4000 7FFF	UART8
	0x4000 7800 - 0x4000 7BFF	UART7
	0x4000 7400 - 0x4000 77FF	DAC
	0x4000 7000 - 0x4000 73FF	PWR
	0x4000 6C00 - 0x4000 6FFF	CAN3
	0x4000 6800 - 0x4000 6BFF	CAN2
	0x4000 6400 - 0x4000 67FF	CAN1
	0x4000 6000 - 0x4000 63FF	I2CFMP1
	0x4000 5C00 - 0x4000 5FFF	I2C3
	0x4000 5800 - 0x4000 5BFF	I2C2
	0x4000 5400 - 0x4000 57FF	I2C1
	0x4000 5000 - 0x4000 53FF	UART5
	0x4000 4C00 - 0x4000 4FFF	UART4
	0x4000 4800 - 0x4000 4BFF	USART3
	0x4000 4400 - 0x4000 47FF	USART2
	0x4000 4000 - 0x4000 43FF	I2S3ext
	0x4000 3C00 - 0x4000 3FFF	SPI3 / I2S3
	0x4000 3800 - 0x4000 3BFF	SPI2 / I2S2
	0x4000 3400 - 0x4000 37FF	I2S2ext
	0x4000 3000 - 0x4000 33FF	IWDG
	0x4000 2C00 - 0x4000 2FFF	WWDG
	0x4000 2800 - 0x4000 2BFF	RTC 和 BKP 寄存器
	0x4000 2400 - 0x4000 27FF	LPTIM1
	0x4000 2000 - 0x4000 23FF	TIM14
	0x4000 1C00 - 0x4000 1FFF	TIM13
	0x4000 1800 - 0x4000 1BFF	TIM12
	0x4000 1400 - 0x4000 17FF	TIM7
	0x4000 1000 - 0x4000 13FF	TIM6
	0x4000 0C00 - 0x4000 0FFF	TIM5
	0x4000 0800 - 0x4000 0BFF	TIM4
	0x4000 0400 - 0x4000 07FF	TIM3
	0x4000 0000 - 0x4000 03FF	TIM2

1. AES 仅在 STM32F423xx 上可用。该边界地址为 STM32F413xx 保留。

2.3 嵌入式 SRAM

STM32F413/423 器件具有 320 KB 的系统 SRAM。

嵌入式 SRAM 可按字节、半字（16 位）或全字（32 位）访问。读写操作以 CPU 速度执行，且等待周期为 0。

嵌入式 SRAM 可分为两个块：

- 映射到地址 0x2000 0000 的 SRAM1，可供所有 AHB 主控总线访问。
- 映射到地址 0x2004 0000 的 SRAM2，可供所有 AHB 主控总线访问。

如果选择从 SRAM1 自举或选择物理重映射（请参见*第 8.2.1 节：SYSCFG 存储器重映射寄存器 (SYSCFG_MEMRMP)*），则 CPU 可通过系统总线或 I-Code/D-Code 总线访问嵌入式 SRAM1。

为了保证程序在 SRAM1 执行时实现最佳性能，应选择物理重映射（通过自举管脚及软件配置来选择）。

当 SRAM2 映射到地址 0x1000 0000 - 0x1000 FFFF 范围内时，CPU 可通过系统总线或 I-Code/D-Code 总线访问嵌入式 SRAM2。为了保证程序在 SRAM2 执行时实现最佳性能，应选择映射到地址 0x1000 0000。

要想利用内置 SRAM 获得最佳性能，应将代码放在 SRAM1/SRAM2 中，通过 I-CODE 总线执行代码，并将数据保存在 SRAM1/SRAM2 中

2.4 Flash 概述

Flash 接口可管理 CPU 通过 AHB I-Code 和 D-Code 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 接口通过指令领取和缓存机制加速代码执行。

flash 存储器结构如下：

- 主存储器块分为多个扇区。
- 系统存储区，器件在系统存储区自举模式下从该存储区启动
- 512 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 接口*。

2.5 位段

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

在 STM32F4x3xx 器件中，外设寄存器和 SRAM1 均映射到一个位段区域，这样可实现单个位段的读写操作。这些操作仅适用于带 FPU 的 Cortex®-M4 内核访问，对于其他总线主接口（如 DMA）无效。

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

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

其中：

– *bit_word_addr* 代表别名区域中将映射到目标位的字的地址
– *bit_band_base* 代表别名区域的起始地址
– *byte_offset* 代表目标位所在位段区域中的字节编号
– *bit_number* 代表目标位的位位置 (0-7)

示例

下例说明如何将 SRAM1 地址 0x20000300 处字节的位 2 映射到别名区域：

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

对地址 0x22006008 执行写操作相当于在 SRAM1 地址 0x20000300 处字节的位 2 执行读-修改-写操作。

对地址 0x22006008 执行读操作将返回 SRAM1 地址 0x20000300 处字节的位 2 的值（0x01 表示位置 1，0x00 表示位复位）。

有关位段的详细信息，请参见带 FPU 的 Cortex®-M4 *编程手册*（请参见*第 1 页的相关文档*）。

2.6 自举配置

存储器采用固定的存储器映射，代码区域起始地址为 0x0000 0000（通过 ICode/DCode 总线访问），而数据区域 (SRAM) 起始地址为 0x2000 0000（通过系统总线访问）。带 FPU 的 Cortex®-M4 CPU 始终通过 ICode 总线获取复位向量，这意味着只有代码区域（通常为 Flash）可以提供自举空间。STM32F4xx 微控制器实施一种特殊机制，可以从其他存储器（如内部 SRAM）进行自举。

在 STM32F4x3xx 中，可通过 BOOT[1:0] 引脚选择三种不同的自举模式，如*表 2* 所示。

表 2. 自举模式

自举模式选择引脚		自举模式	自举空间
BOOT1	**BOOT0**		
x	0	主Flash	选择主 Flash 作为自举空间
0	1	系统存储器	选择系统存储器作为自举空间
1	1	嵌入式 SRAM	选择嵌入式 SRAM 作为自举空间

复位后，在 SYSCLK 的第四个上升沿锁存 BOOT 引脚的值。复位后，用户可以通过设置 BOOT1 和 BOOT0 引脚来选择需要的自举模式。

BOOT0 为专用引脚，而 BOOT1 则与 GPIO 引脚共用。一旦完成对 BOOT1 的采样，相应 GPIO 引脚即进入空闲状态，可用于其他用途。

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。这样的启动延迟结束后，CPU 将从地址 0x0000 0000 获取栈顶值，然后从始于 0x0000 0004 的自举存储器开始执行代码。

注： *如果器件从 SRAM 自举，在应用程序初始化代码中，需要使用 NVIC 异常及中断向量表和偏移寄存器来重新分配 SRAM 中的向量表。*

内部自举程序

嵌入式自举程序模式用于通过*表 32* 中所列的任一接口来重新编程 Flash。接口可用性取决于封装。

表 3. 嵌入式自举程序接口

封装	USART1 PA9/ PA10	USART2 PD6/ PD5	USART3 PB11/ PB10	I2C1 PB6/ PB7	I2C2 PF0/ PF1	I2C3 PA8/ PB4	I2C FMP1 PB14/ PB15	SPI1 PA4/ PA5/ PA6/ PA7	SPI3 PA15/ PC10/ PC11/ PC12	SPI4 PE11/ PE12/ PE13/ PE14	CAN2 PB5/ PB13	USB PA11 /P12
UFQFPN48	是	-	-	是	-	是	是	是	-	-	是	是
LQFP64	是	-	-	是	-	是	是	是	是	-	是	是
WLCSP81	是	-	-	是	-	是	是	是	是	是	是	是
LQFP100	是	是	-	是	-	是	是	是	是	是	是	是
LQFP144	是	是	是	是	是	是	是	是	是	是	是	是
UFBGA100	是	是	是	是	-	是	是	是	是	是	是	是
UFBGA144	是	是	是	是	是	是	是	是	是	是	是	是

USART 外设以内部 16 MHz 振荡器 (HSI) 频率运行，而 CAN 和 USB OTG FS 则需要相当于 1 MHz 数倍（4 MHz 到 26 MHz 之间）的外部时钟 (HSE) 频率。

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。有关详细信息，请参见应用笔记 AN2606。

STM32F413/423 中的物理重映射

选择自举引脚后，应用程序软件可以将某些存储器设定为从代码空间进行访问（这样，可通过 ICode 总线而非系统总线执行代码）。这样的修改通过在 SYSCFG 控制器中编程*第 8.2.1 节：SYSCFG 存储器重映射寄存器 (SYSCFG_MEMRMP)* 来实现。

因此可重映射以下存储器：

- 主 Flash
- 系统存储器
- 嵌入式 SRAM

表 4. STM32F413/423 中的存储器映射与自举模式/物理重映射

地址	主 Flash 中的自举/重映射	嵌入式 SRAM 中的自举/重映射	系统存储器中的自举/重映射
0x2000 0000 - 0x2003 FFFF	SRAM (256 KB)	SRAM (256 KB)	SRAM (256KB)
0x1FFF 0000 - 0x1FFF 77FF	系统存储器	系统存储器	系统存储器
0x0802 0000 - 0x1FFE FFFF	保留	保留	保留
0x0800 0000 - 0x080F FFFF	Flash	Flash	Flash
0x0400 000 - 0x07FF FFFF	保留	保留	保留
0x0000 0000 - 0x0003 FFFF(1)	Flash (1M) 使用别名区	SRAM1 (256 KB) 使用别名区	系统存储器 (30 KB) 使用别名区

1. 即使在自举存储空间中使用别名区，相关存储器仍可通过其原始存储空间进行访问。

3 嵌入式 Flash 接口

3.1 简介

Flash 接口可管理 CPU 通过 AHB I-Code 和 D-Code 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。

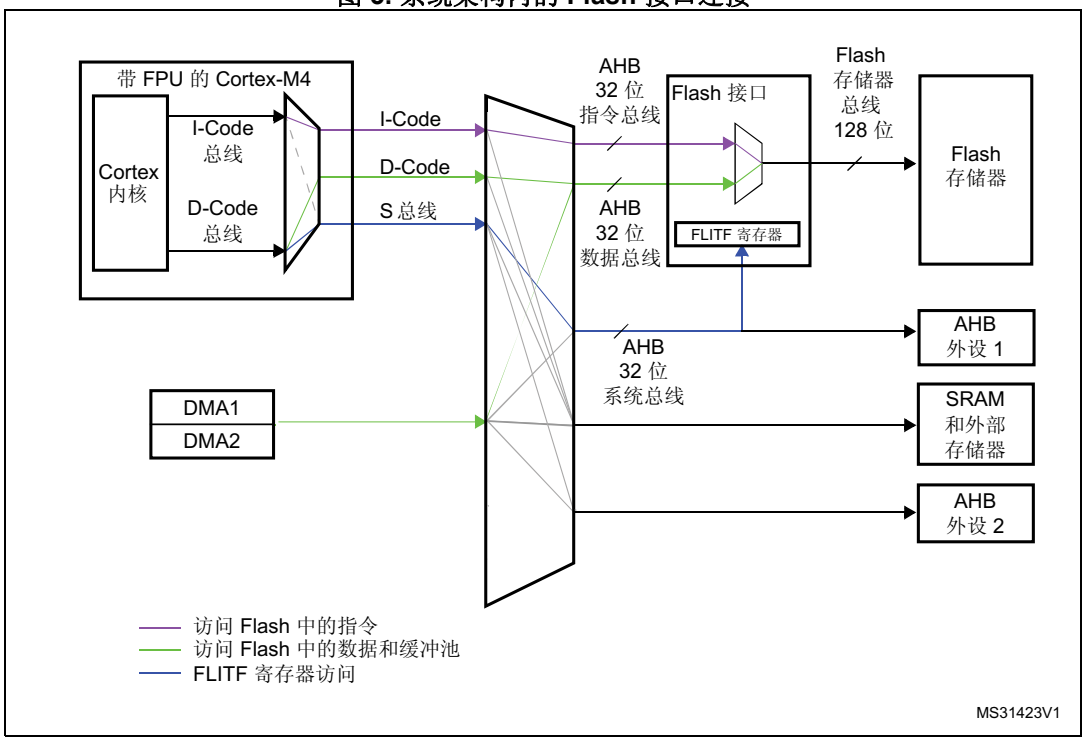
Flash 接口通过指令预取和缓存机制加速代码执行。

3.2 主要特性

- Flash 读操作
- Flash 编程/擦除操作
- 读/写保护
- I-Code 上的预取操作
- I-Code 上的 64 个缓存（128 位宽）
- D-Code 上的 8 个缓存（128 位宽）

图 3 所示为系统架构内的 Flash 接口连接。

图 3. 系统架构内的 Flash 接口连接



3.3 嵌入式 Flash

Flash 具有以下主要特性：

- 容量高达 1.5 MB
- 128 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
- 存储器组织结构

  Flash 结构如下：
  - 主存储器块，分为 4 个 16 KB 扇区、1 个 64 KB 扇区和 11 个 128 KB 扇区。
  - 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。
  - 512 字节 OTP（一次性可编程），用于存储用户数据。

    OTP 区域还有 32 个附加位，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、软件/硬件看门狗以及器件处于待机或停止模式下的复位。
- 低功耗模式（有关详细信息，请参见参考手册的“电源控制 (PWR)”部分）

表 5. Flash 模块构成

块	名称	块基址	大小
主存储器	扇区 0	0x0800 0000 - 0x0800 3FFF	16 KB
	扇区 1	0x0800 4000 - 0x0800 7FFF	16 KB
	扇区 2	0x0800 8000 - 0x0800 BFFF	16 KB
	扇区 3	0x0800 C000 - 0x0800 FFFF	16 KB
	扇区 4	0x0801 0000 - 0x0801 FFFF	64 KB
	扇区 5	0x0802 0000 - 0x0803 FFFF	128 KB
	扇区 6	0x0804 0000 - 0x0805 FFFF	128 KB
	扇区 7	0x0806 0000 - 0x0807 FFFF	128 KB
	扇区 8	0x0808 0000 - 0x0809 FFFF	128 KB
	扇区 9	0x080A 0000 - 0x080B FFFF	128 KB
	扇区 10	0x080C 0000 - 0x080D FFFF	128 KB
	扇区 11	0x080E 0000 - 0x080F FFFF	128 KB
	扇区 12	0x08010 0000 - 0x0811 FFFF	128 KB
	扇区 13	0x0812 0000 - 0x0813 FFFF	128 KB
	扇区 14	0x0814 0000 - 0x0815 FFFF	128 KB
	扇区 15	0x0816 0000 - 0x0817 FFFF	128 KB
系统存储器		0x1FFF 0000 - 0x1FFF 77FF	30 KB
OTP 区域		0x1FFF 7800 - 0x1FFF 7A0F	528 字节
选项字节		0x1FFF C000 - 0x1FFF C00F	16 字节

3.4 读接口

3.4.1 CPU 时钟频率与 Flash 读取时间之间的关系

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)。

- 当 VOS[1:0] = 0x01 时，f_{HCLK} 的最大值 = 64 MHz。
- 当 VOS[1:0] = 0x10 时，f_{HCLK} 的最大值 = 84 MHz。
- 当 VOS[1:0] = 0x11 时，f_{HCLK} 的最大值 = 100 MHz。

表 6. 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 1.7 V - 2.1 V
0 WS （1 个 CPU 周期）	0 < HCLK ≤ 25	0 < HCLK ≤ 20	0 < HCLK ≤ 18	0 < HCLK ≤ 16
1 WS （2 个 CPU 周期）	25 < HCLK ≤ 50	20 < HCLK ≤ 40	18 < HCLK ≤ 36	16 <HCLK ≤ 32
2 WS （3 个 CPU 周期）	50 < HCLK ≤ 75	40 < HCLK ≤ 60	36 < HCLK ≤ 54	32 < HCLK ≤ 48
3 WS （4 个 CPU 周期）	75 < HCLK ≤ 100	60 < HCLK ≤ 80	54 < HCLK ≤ 72	48 < HCLK ≤ 64
4 WS （5 个 CPU 周期）	-	80 < HCLK ≤ 100	72 < HCLK ≤ 90	64 < HCLK ≤ 80
5 WS （6 个 CPU 周期）	-	-	90 < HCLK ≤ 100	80 < HCLK ≤ 96
6 WS （7 个 CPU 周期）	-	-	-	96 < HCLK ≤ 100

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 频率调整在访问 Flash 时所需的等待周期数。

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。*

3.4.2 自适应实时存储器加速器 (ART Accelerator™)

专有的自适应实时 (ART) 存储器加速器面向 STM32 工业标准 Arm® 带 FPU 的 Cortex®-M4 处理器进行了优化。该加速器很好地体现了 Arm® 带 FPU 的 Cortex®-M4 在 Flash 技术方面的固有性能优势，克服了通常条件下，高速的处理器在运行中需要经常等待 Flash 读取的情况。

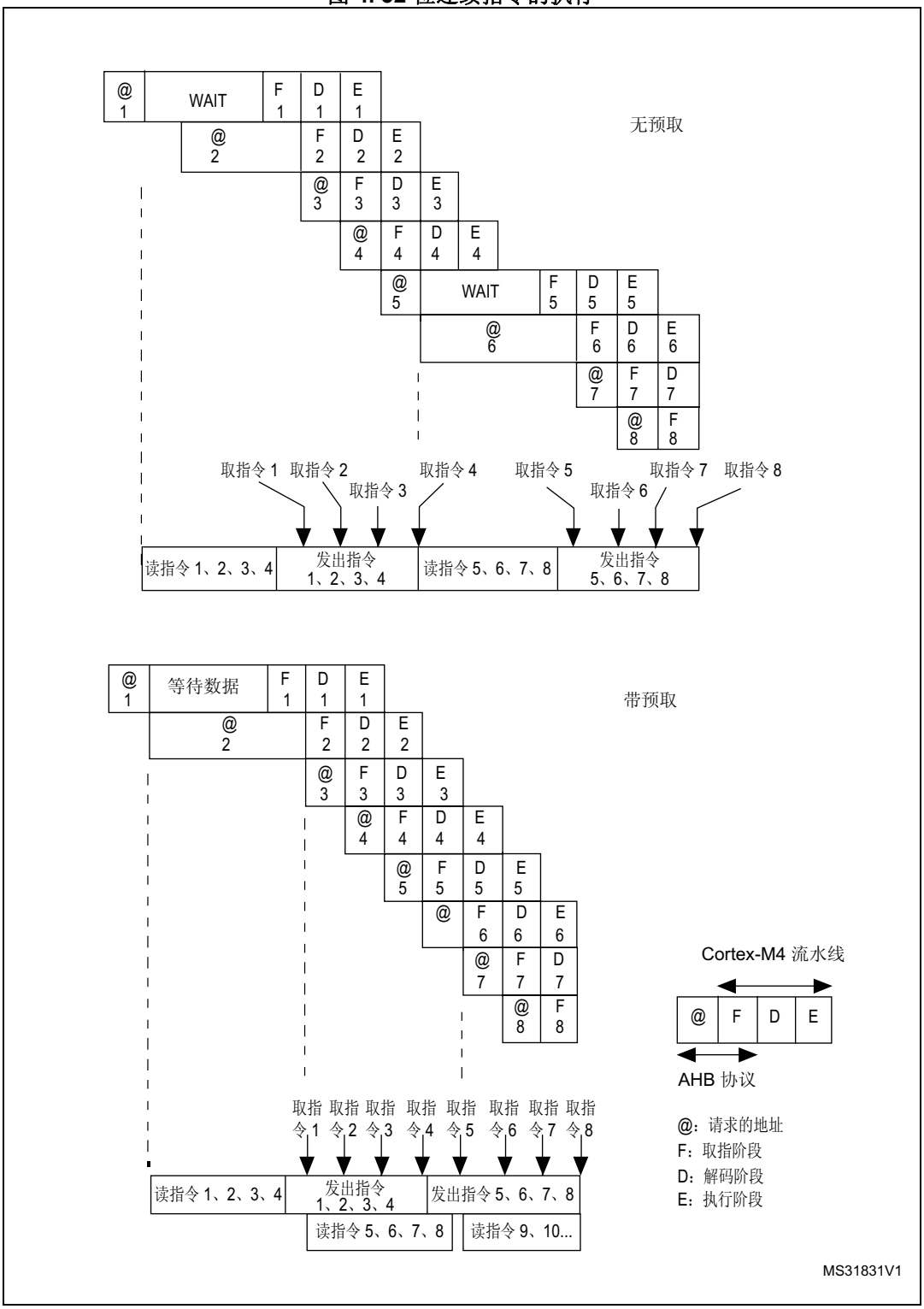
为了发挥处理器的全部性能，该加速器将实施指令预取队列和分支缓存，从而提高了 128 位 Flash 的程序执行速度。根据 CoreMark 基准测试，凭借 ART 加速器所获得的性能相当于 Flash 在 CPU 频率高达 100 MHz 时以 0 个等待周期执行程序。

指令预取

每个 Flash 读操作可读取 128 位，可以是 4 条 32 位指令，也可以是 8 条 16 位指令，具体取决于烧写在 Flash 中的程序。因此对于顺序执行的代码，至少需要 4 个 CPU 周期来执行前一次读取的 128 位指令行。在 CPU 请求当前指令行时，可使用 I-Code 总线的预取操作读取 Flash 中的下一个连续存放的 128 位指令行。可将 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。

图 4 所示为需要 3 WS（3 个等待周期）访问 Flash 时连续 32 位指令的执行过程，图中分别介绍了使用和不使用预取操作两种情况。

图 4. 32 位连续指令的执行



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或预取的指令行中。这种情况下，CPU 等待时间至少等于等待周期数。

指令缓存存储器

为了减少因指令跳转而产生的时间损耗，可将 64 行 128 位的指令保存到指令缓存存储器中。可将 FLASH_ACR 寄存器中的指令缓存使能 (ICEN) 位置 1，来使能这一特性。每当出现指令缺失（即请求的指令未存在于当前使用的指令行、预取指令行或指令缓存存储器中）时，系统会将新读取的行复制到指令缓存存储器中。如果 CPU 请求的指令已存在于指令缓存区中，则无需任何延时即可立即获取。指令缓存存储器存满后，可采用 LRU（最近最少使用）策略确定指令缓存存储器中待替换的指令行。此特性非常适用于包含循环的代码。

数据管理

在 CPU 流水线执行阶段，将通过 D-Code 总线访问 Flash 中的数据缓冲池。因此，直到提供了请求的数据后，CPU 流水线才会继续执行。为了减少因此而产生的时间损耗，通过 AHB 数据总线 D-Code 进行的访问优先于通过 AHB 指令总线 I-Code 进行的访问。

如果频繁使用某些数据，可将 FLASH_ACR 寄存器中的数据缓存使能 (DCEN) 位置 1，来使能数据缓存存储器。此特性的工作原理与指令缓存存储器类似，但保留的数据大小限制在 8 行 128 位/行以内。

注： *用户配置扇区中的数据无法缓存。*

3.5 擦除和编程操作

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。

在对 STM32F4xx 的 Flash 执行写入或擦除操作期间，任何读取 Flash 的尝试都会导致总线阻塞。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。

3.5.1 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。*

3.5.2 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 擦除操作只能针对扇区或整个 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 7 提供了正确的 PSIZE 值。

表 7. 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 1.7 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。

3.5.3 擦除

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，并从主存储块的 16 个扇区中选择要擦除的扇区 (SNB)
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。

3.5.4 编程

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

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。

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。

如果未遵循标准的编程顺序（例如，在 PG 位未置 1 时尝试向 Flash 地址写入数据），则操作将中止并且 FLASH_SR 寄存器中的编程顺序错误标志位 (PGSERR) 将置 1。

编程与缓存

如果 Flash 写访问涉及数据缓存中的某些数据，Flash 写访问将修改 Flash 中的数据和缓存中的数据。

如果 Flash 中的擦除操作也涉及数据或指令缓存中的数据，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 DCRST 和 ICRST 位置 1，以刷新缓存。

注： *I/D 缓存只有在被禁止 (I/DCEN = 0) 的情况下才能刷新。*

3.5.5 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、PGSERR（编程错误标志）
- WRPERR（保护错误标志）

这种情况下，FLASH_CR 寄存器中的操作错误位 (OPERR) 置 1，并且如果 FLASH_SR 寄存器中的错误中断使能位 (ERRIE) 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 8. 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、PGSERR	ERRIE

3.6 选项字节

3.6.1 关于用户选项字节的说明

选项字节由最终用户根据具体的应用要求进行配置。*表 9* 介绍了这些字节在用户配置扇区内的构成。

表 9. 选项字节构成

地址	[63:16]	[15:0]
0x1FFF C0000	保留	ROP 和用户选项字节 (RDP & USER)
0x1FFF C008	保留	扇区 0 到 15 的写保护 nWRP 位

表 10. 关于选项字节的说明

选项字节（字，地址 0x1FFF C000）	
RDP：*读保护选项字节 (Read protection option byte)* 读保护用于保护 Flash 中存储的软件代码。	
位 15:8	0xAA：级别 0，无保护 0xCC：级别 2，芯片保护（禁止调试和从 RAM 自举功能） 其他：级别 1，存储器读保护（调试功能受限）
USER：用户选项字节 (User option byte) 此字节用于配置以下功能： – 选择看门狗事件：硬件或软件 – 进入停止模式时产生复位事件 – 进入待机模式时产生复位事件	
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：进入停机模式时产生复位。 1：不产生复位
位 5	**WDG_SW** 0：硬件独立看门狗 1：软件独立看门狗
位 4	0x1：未使用
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	0x1：未使用
选项字节（字，地址 0x1FFF C008）	
位 15	**SPRMOD**：nWPRi 位保护模式选择 (Selection of Protection Mode of nWPRi bits) 0：nWPRi 位用于扇区 i 写保护（默认） 1：nWPRi 位用于扇区 i PCROP 保护（扇区）
nWRP：*Flash 写保护选项字节 (Flash memory write protection option bytes)* 扇区 0 到 15 可采用写保护	

表 10. 关于选项字节的说明

位 14	**nWRP15_14**：扇区 15 和 14 的低有效写保护 (Non Write Protection of sector 15 and 14) 如果 SPRMOD 复位（默认值）： 0：激活扇区 15 和 14 的写保护。 1：未激活扇区 15 和 14 的写保护。 如果 SPRMOD 置 1（激活）： 0：未激活扇区 15 和 14 的 PCROP 保护。 1：激活扇区 15 和 14 的 PCROP 保护。
位 13:0	**nWRPi** 如果 SPRMOD 复位（默认值）： 0：激活扇区 i 的写保护 1：未激活扇区 i 的写保护 如果 SPRMOD 置 1（激活）： 0：未激活扇区 i 的 PCROP 保护。 1：激活扇区 i 的 PCROP 保护。

3.6.2 用户选项字节编程

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： *硬件会自动先擦除用户配置扇区，然后以 FLASH_OPTCR 寄存器中包含的值对所有选项字节进行编程。*

3.6.3 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，在所有启动配置（用户 Flash 启动、调试或从 RAM 启动）中，均可执行对 Flash 的所有读/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 或系统存储器启动程序启动时，不能对 Flash 进行访问（读取、擦除、编程）。读请求将导致总线错误。

  – 从 Flash 启动时，允许通过用户代码对 Flash 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 执行批量擦除。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他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 或系统存储器启动。

  – 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 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。*

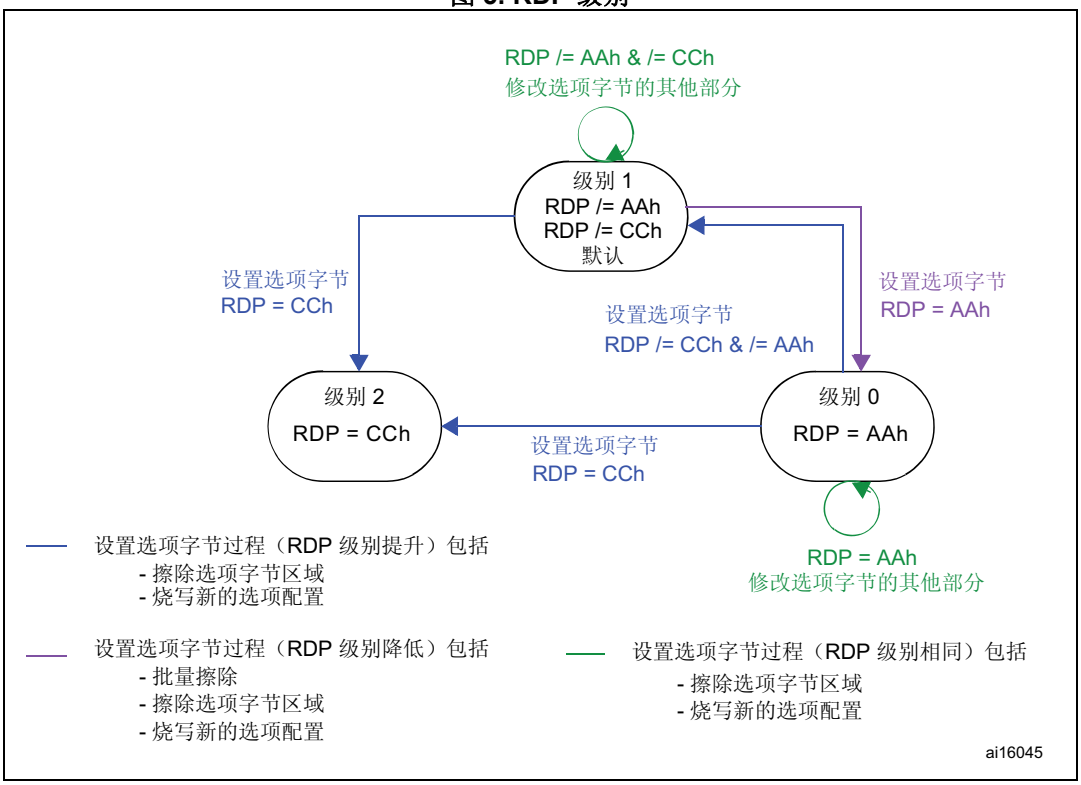
表 11. 不同读保护级别下的访问限制

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	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		NA	是		NA
	级别 2	否		NA	是		NA

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

图 5 所示为 RDP 级别切换的过程。

图 5. RDP 级别



3.6.4 写保护

Flash 中有多达 16 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 寄存器中的低有效写保护 nWRPi 位 (0 ≤ i ≤ 14) 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。*
扇区 14 和 15 连接在一起，无法单独实施写保护。

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：

- 配置批量擦除、扇区擦除（MER 或 MER/MER1 和 SER = 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_OPTCRx 寄存器中的 MER 或 MER/MER1 = 1 且 nWRPi = 0，其中 0 ≤ i ≤ 14 位）
- 请求针对写保护扇区执行扇区擦除。（FLASH_OPTCRx 寄存器中的 SER = 1、SNB = i 且 nWRPi = 0，其中 0 ≤ i ≤ 14 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
- 针对用户配置扇区执行写操作
- 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

3.6.5 专有代码读保护（PCROP）

通过对 Flash 用户扇区（0 到 15）实施专有读保护 (PCROP)，可防止通过 D 总线进行的读访问。

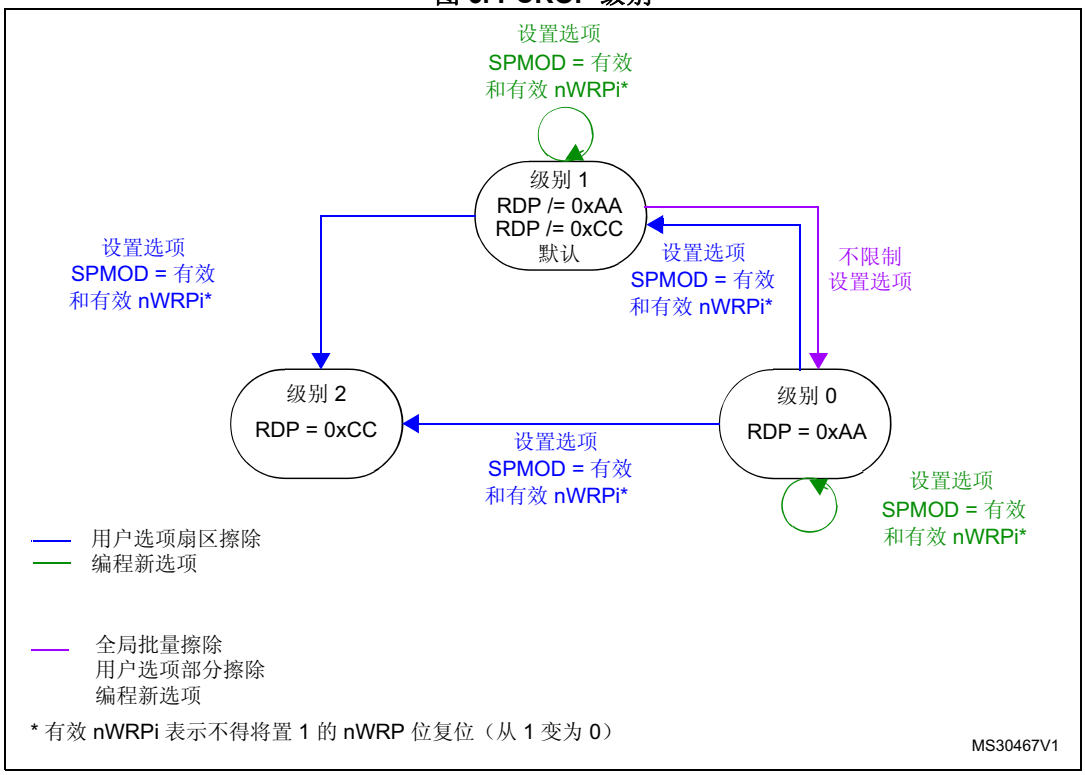
用户可通过 FLASH_CR 寄存器中的 SPRMOD 选项位按如下所述选择 PCROP 保护：

- SPRMOD = 0：nWRPi 控制相应用户扇区的写保护
- SPRMOD = 1：nWRPi 控制相应用户扇区的读写保护 (PCROP)

当某个扇区受到读保护（激活 PCROP 模式）时，只能对其执行代码获取操作，并且必须使用 Flash 接口通过 ICODE 总线进行访问：

- 通过 D 总线执行的任何读访问都会触发 RDERR 标志错误。
- 对具有 PCROP 保护的扇区执行的任何编程/擦除操作都会触发 WRPERR 标志错误。

图 6. PCROP 级别



只有在 RDP 从级别 1 切换为级别 0 时，才会禁用 SPRMOD 和/或取消对用户扇区的 PCROP 保护。如果不满足该条件，则取消用户选项字节修改并将写错误 WRPERR 标志置 1。由于有效的 nWRPi 位均未复位且 SPRMOD 保持有效，因此允许对用户选项字节（BOR_LEV、RST_STDBY、..）进行修改。

注： *当激活 PCROP 模式 (SPRMOD =1) 时，nWRPi 位的有效值取反。*

3.7 一次性可编程字节

表 12 所示为 OTP 区域中一次性可编程 (OTP) 部分的构成。

表 12. OTP 区域构成

块	[128:96]	[95:64]	[63:32]	[31:0]	地址字节 0
0	OTP0	OTP0	OTP0	OTP0	0x1FFF 7800
	OTP0	OTP0	OTP0	OTP0	0x1FFF 7810
1	OTP1	OTP1	OTP1	OTP1	0x1FFF 7820
	OTP1	OTP1	OTP1	OTP1	0x1FFF 7830
. . .	. . .				. . .
15	OTP15	OTP15	OTP15	OTP15	0x1FFF 79E0
	OTP15	OTP15	OTP15	OTP15	0x1FFF 79F0
锁定块	LOCKB15 ... LOCKB12	LOCKB11 ... LOCKB8	LOCKB7 ... LOCKB4	LOCKB3 ... LOCKB0	0x1FFF 7A00

OTP 区域划分为 16 个 32 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。

3.8 Flash 接口寄存器

3.8.1 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.	DCRST	ICRST	DCEN	ICEN	PRFTEN	Res.	Res.	Res.	Res.	LATENCY			
			rw	w	rw	rw	rw					rw	rw	rw	rw

位 31:13 保留，必须保持清零。

位 12 **DCRST**：数据缓存复位 (Data cache reset)
0：数据缓存不复位
1：数据缓存复位
只有在关闭数据缓存时才能在该位中写入值。

位 11 **ICRST**：指令缓存复位 (Instruction cache reset)
0：指令缓存不复位
1：指令缓存复位
只有在关闭指令缓存时才能在该位中写入值。

位 10 **DCEN**：数据缓存使能 (Data cache enable)
0：禁止数据缓存
1：使能数据缓存

位 9 **ICEN**：指令缓存使能 (Instruction cache enable)
0：禁止指令缓存
1：使能指令缓存

位 8 **PRFTEN**：预取使能 (Prefetch enable)
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**：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

3.8.2 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

3.8.3 Flash 选项密钥寄存器 (FLASH_OPTKEYR)

Flash option key register

借助 Flash 选项密钥寄存器，可允许在用户配置扇区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

3.8.4 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.	RDERR	PGSERR	PGPERR	PGAERR	WRPERR	Res.	Res.	OPERR	EOP
							rw	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 15:9 保留，必须保持清零。

位 8 **RDERR**：读保护错误 (PCROP) (Read Protection Error (PCROP))

如果通过 D 总线读取的地址属于 Flash 中处于读保护状态的区域，将由硬件为该位置 1。
写入 1 即可将该位复位。

位 7 **PGSERR**：编程顺序错误 (Programming 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 **PGPERR**：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。

3.8.5 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 (Program size)
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**：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
...
1010 扇区 10
1011 扇区 11
1100 扇区 12
1101 扇区 13
1110 扇区 14
1111 扇区 15

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。

3.8.6 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：0x0FFF FFED。复位信号释放时，硬件将 Flash 中的值加载到这些选项位。

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
SPR MOD	nWRP 14_15	nWRP[13:0]													
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
RDP[7:0]								nRST_ STDBY	nRST_ STOP	WDG_ SW	Res.	BOR_LEV		OPT STRT	OPT LOCK
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rs	rs

位 31 **SPRMOD**：nWPRi 位保护模式选择 (Selection of Protection Mode of nWPRi bits)

0：禁止 PCROP，nWPRi 位用于扇区 i 写保护

1：使能 PCROP，nWPRi 位用于扇区 i PCROP 保护

位 30 **nWRP14_15**：无写保护 (Not write protect)

该位包含扇区 14 和 15 的写保护选项字节值。通过对这些位执行写操作，可将新的写保护值编程到 Flash。扇区 14 和 15 连接在一起，无法单独执行编程操作

0：激活扇区 14 和 15 的写保护

1：未激活扇区 14 和 15 的写保护

这些位包含复位后扇区 14 和 15 的写保护和读保护 (PCROP) 选项字节值。通过对这些位执行写操作，可将新的写保护值或 PCROP 值编程到 Flash。

如果 SPRMOD 复位：

0：激活扇区 14 和 15 的写保护

1：未激活扇区 14 和 15 的写保护

如果 SPRMOD 置 1：

0：未激活扇区 14 和 15 的 PCROP 保护

1：激活扇区 14 和 15 的 PCROP 保护

位 29:16 **nWRP[13:0]**：无写保护 (Not write protect)

这些位包含复位后扇区写保护选项字节的值。通过对这些位执行写操作，可将新的写保护值编程到 Flash。

0：激活所选扇区的写保护

1：未激活所选扇区的写保护

这些位包含复位后扇区 0 到 13 的写保护和读保护 (PCROP) 选项字节值。通过对这些位执行写操作，可将新的写保护值或 PCROP 值编程到 Flash。

如果 SPRMOD 复位：

0：激活扇区 i 的写保护

1：未激活扇区 i 的写保护

如果 SPRMOD 置 1：

0：未激活扇区 i 的 PCROP 保护

1：激活扇区 i 的 PCROP 保护

位 15:8 **RDP**：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 7:5 **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：WDG_SW

注：　当 WDG 模式从硬件切换到软件或从软件切换到硬件时，需要执行系统复位才能更改生效。

位 4 保留，必须保持清零。始终读为 0。

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。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，请参见器件数据手册中的“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。

3.8.7 Flash 接口寄存器映射

表 13. 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.	DCRST	ICRST	DCEN	ICEN	PRFTEN	Res.	Res.	Res.	Res.	LATENCY			
	Reset value																				0	0	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.	RDERR	PGSERR	PGPERR	PGAERR	WRPERR	Res.	Res.	OPERR	EOP
	Reset value																0								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	SPRMOD	nWRP14_15	nWRP[13:0]														RDP[7:0]								nRST_STDBY	nRST_STOP	WDG_SW	Res.	BOR_LEV		OPTSTRT	OPTLOCK
	Reset value	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	1	1		1	1	0	1

4 CRC 计算单元

4.1 CRC 简介

CRC（循环冗余校验）计算单元使用一个固定的多项式发生器从一个 32 位的数据字中产生 CRC 码。

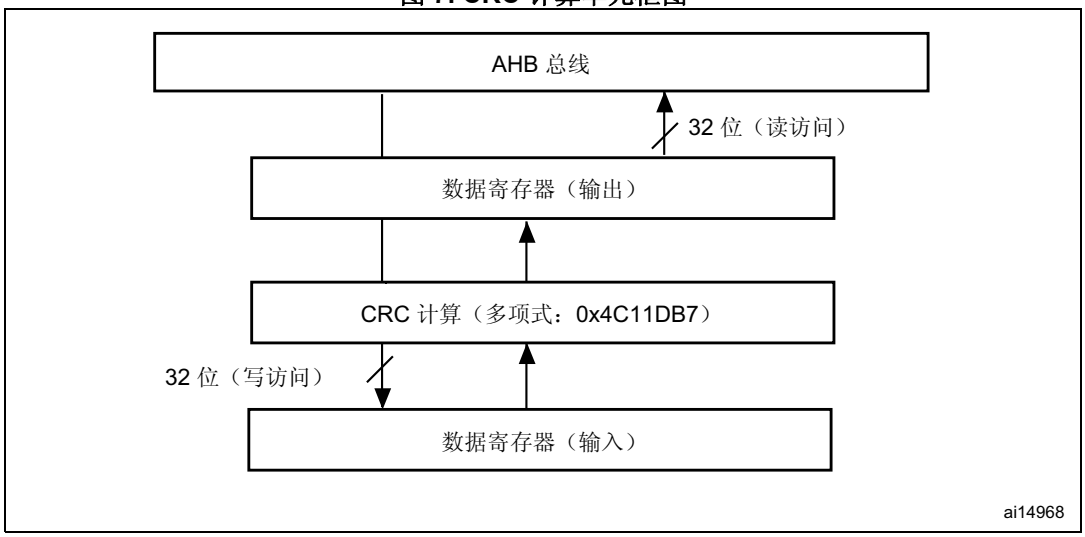
在众多的应用中，基于 CRC 的技术还常用来验证数据传输或存储的完整性。根据 EN/IEC 60335-1 标准的规定，这些技术提供了验证 Flash 完整性的方法。CRC 计算单元有助于在运行期间计算软件的识别标志，并将该识别标志与链接时生成并存储在指定存储单元的参考识别标志加以比较。

4.2 CRC 主要特性

- 使用 CRC-32（以太网）多项式：0x4C11DB7
  – $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- 单输入/输出 32 位数据寄存器
- 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）

框图如*图 7* 所示。

图 7. CRC 计算单元框图



4.3 CRC 功能说明

CRC 计算单元主要由单个 32 位数据寄存器组成，该寄存器：

- 用作输入寄存器，向 CRC 计算器中输入新数据（向寄存器写入数据时）
- 可保存之前的 CRC 计算结果（读取寄存器时）

对数据寄存器的每个写操作都会把当前新输入的数值和之前生成在数据寄存器中的 CRC 值再做一次 CRC 计算（CRC 计算针对整个 32 位数据字完成，而非逐字节进行）。

CRC 计算的时候，写操作被阻塞，因此允许执行背靠背写访问或连续的写读访问。

使用 CRC_CR 寄存器中的 RESET 控制位即可将 CRC 计算器复位为 0xFFFF FFFF。此操作不影响 CRC_IDR 寄存器的内容。

4.4 CRC 寄存器

CRC 计算单元包含两个数据寄存器和一个控制寄存器。外设 CRC 寄存器必须按字（32 位）访问。

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

位 31:0 **数据寄存器位**

向 CRC 计算器写入新数据时用作输入寄存器。
读取寄存器时可读出之前的 CRC 计算结果。

4.4.2 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	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:0 **通用的 8 位数据寄存器位**

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。

4.4.3 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RESET
															w

位 31:1 保留，必须保持复位值。

位 0 **RESET 位**

复位 CRC 计算单元并将数据寄存器设为 0xFFFF FFFF。

此位只能置 1，将由硬件自动进行清零。

4.4.4 CRC 寄存器映射

表 14. CRC 计算单元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**	Data register																															
	Reset value	0xFFFF FFFF																															
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.	Independent data register							
	Reset value																									0x0000							
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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RESET
	Reset value																																0

5 电源控制器 (PWR)

5.1 电源

供电方案主要有以下两种：

- VDD = 1.7 到 3.6 V：I/O 的外部电源（禁用内部调压器），通过 VDD 引脚从外部提供。需要使用外部电源监控器，连接到 VDD 和 PDR_ON 引脚。
- VDD = 1.8 到 3.6 V：I/O 和内部调压器（若启用）的外部电源，通过 VDD 引脚从外部提供。
- V_{DD_USB} = 3.0 到 3.6 V

  V_{DD_USB} 是为全速收发器供电的专用独立 USB 电源。

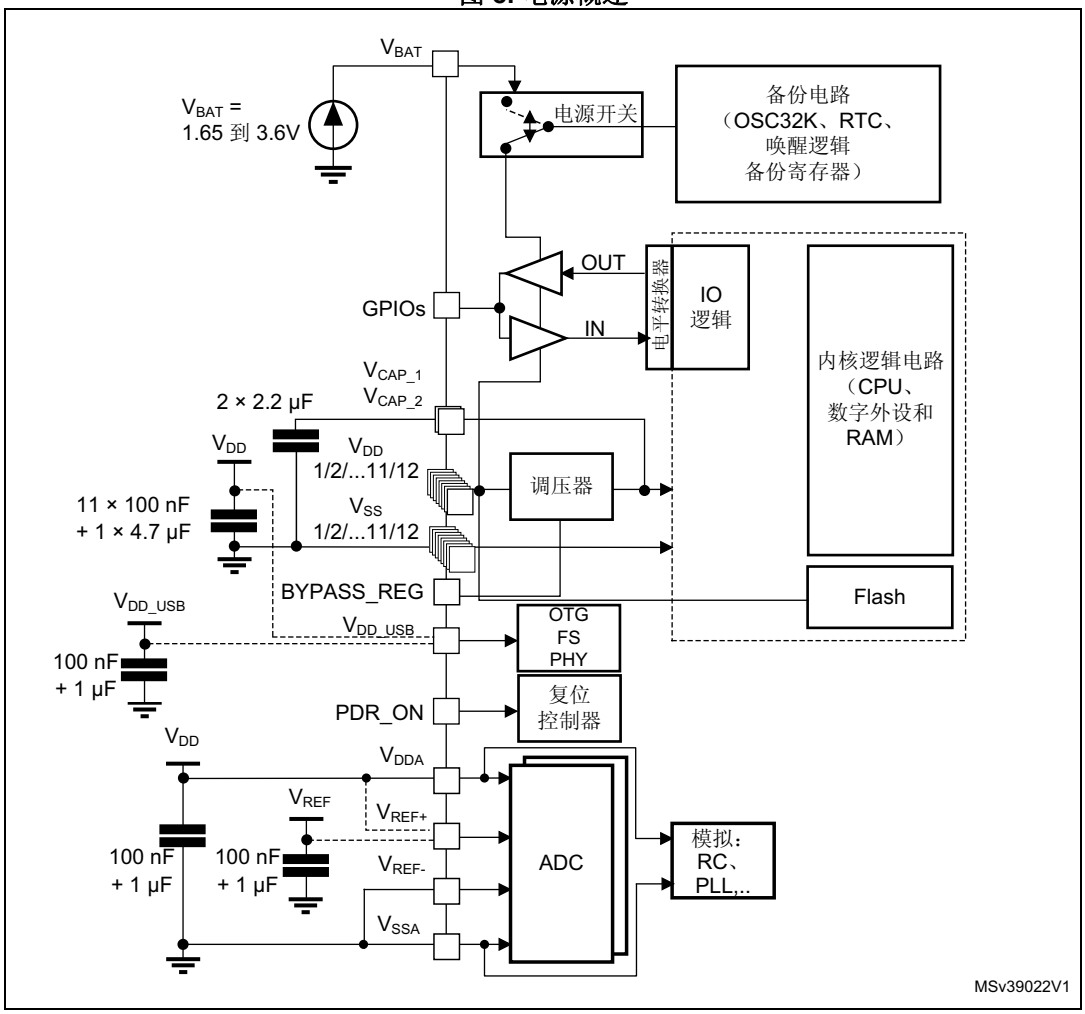
注： *V_{DD_USB} 值与 V_{DD} 和 V_{DDA} 无关。但是，V_{DD_USB} 电源必须最后提供给器件并且最先关闭。当关闭三个电源时，如果 V_{DD_USB} 在短时间内保持激活并且 V_{DDA}/V_{DDIO} 低于功能范围，则器件不会损坏。*

当 V_{DD_USB} 关闭时，器件仍能工作。

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

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。有关详细信息，请参见数据手册中“通用工作条件”一节。*

图 8. 电源概述



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

5.1.1 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} 上的电压范围为 1.7 V 到 V_{DDA}。

5.1.2 电池备份域

备份域说明

要在 V_{DD} 断电后保留 RTC 备份寄存器的内容并为 RTC 供电，可以将 V_{BAT} 引脚连接到电池或其他备用电源上。

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
- PC13 到 PC15 I/O

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 可用作 GPIO，也可配置为其他功能（有关此引脚配置的详细信息，请参见*表 27：RTC 附加功能*）

注： *由于该开关的灌电流能力有限 (3 mA)，因此在输出模式下使用 PC13 到 PC15 GPIO 时存在以下限制：速率不得超过 2 MHz，最大负载为 30 pF，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 RTC 附加功能引脚（有关此引脚配置的详细信息，请参见*表 27：RTC 附加功能*）

备份域访问

复位后，备份域（RTC 寄存器和 RTC 备份寄存器）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 6.3.14 节：RCC AHB3 外设时钟使能寄存器 (RCC_AHB3ENR)*）
2. 将*第 5.4.1 节*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 6.2.8 节：RTC/AWU 时钟*
4. 通过对*第 6.3.23 节：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

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 20 个备份数据寄存器（80 字节），在检测到入侵事件时将复位。有关详细信息，请参见*第 25 节：实时时钟 (RTC)*。

5.1.3 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。

该调压器需要为专用引脚 V_{CAP_1}（某些封装中还提供另一个专用引脚 V_{CAP_2}）连接外部电容。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，可采用三种不同的模式工作。

- 在**运行模式**中，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_CR 寄存器的 VOS[1:0] 位配置为级别 1、级别 2 或级别 3）。复位后，VOS 寄存器将设为级别 2。当 PLL 关闭时，调压器将设为级别 3，而与 VOS 寄存器内容无关。只有激活 PLL 且选择 HSI 或 HSE 作为时钟源时，才考虑 VOS 寄存器内容。

  当器件运行频率低于最高工作频率时，电压调节特性可使功耗得到优化。
- 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)。在停止模式下，设置的电压输出级别保持不变：

  微控制器进入停止模式后将自动选择电压输出级别 3（参见*第 5.4.1 节：PWR 电源控制寄存器 (PWR_CR)*）。
- 在**待机模式**中，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： *有关详细信息，请参见 STM32F413/423 数据手册的调压器部分。*

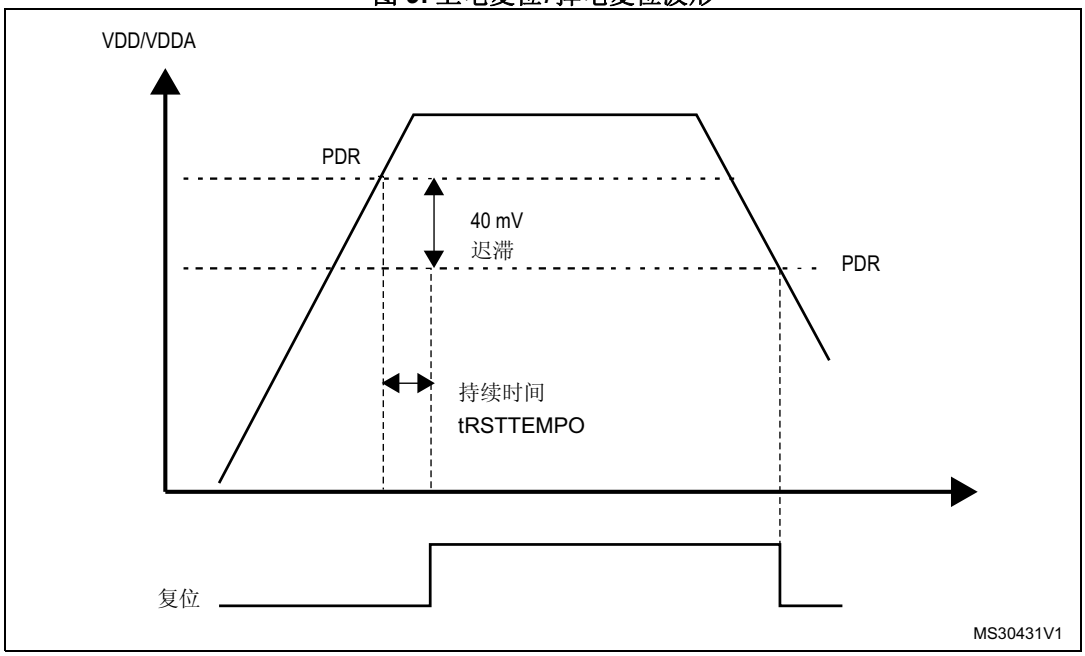
5.2 电源监控器

5.2.1 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。

要在低于 1.8 V 时使用器件，必须使用 PDR_ON 引脚关闭内部电源监控器（请参见 STM32F413/423 数据手册的电源监控器部分）。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 9. 上电复位/掉电复位波形



5.2.2 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。

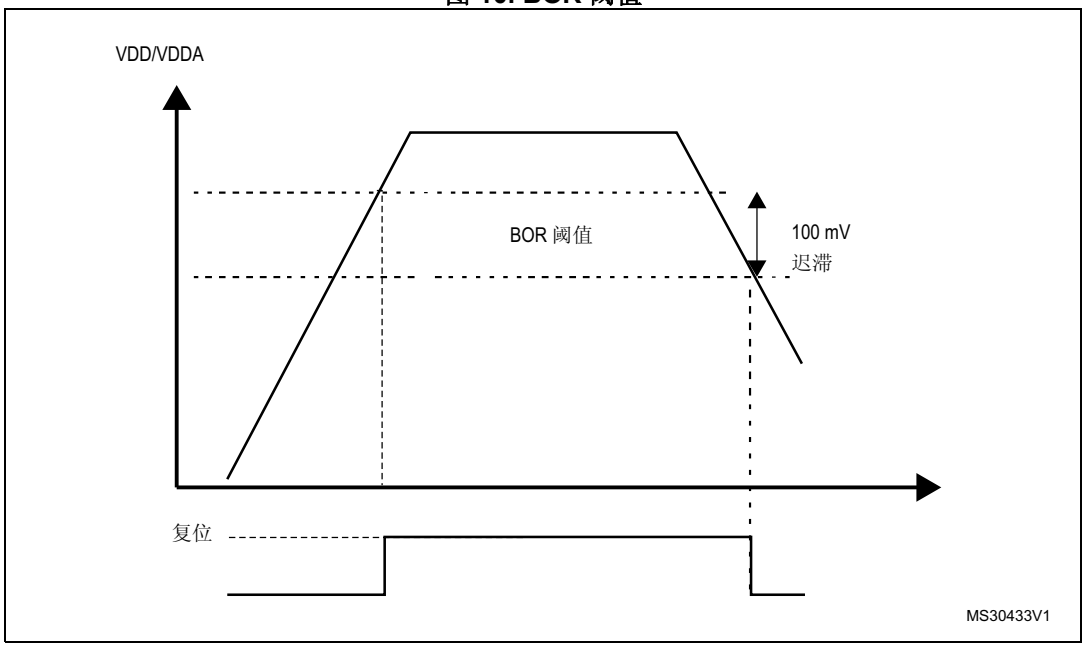
注： *有关 BOR 特性的完整详情，请参见器件数据手册中的“电气特性”部分。*

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。这种情况下，上电和掉电的监控可以通过 POR/PDR 或者外部电源监控器（如果 PDR 被 PDR_ON 引脚关闭）（请参见*第 5.2.1 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 10. BOR 阈值



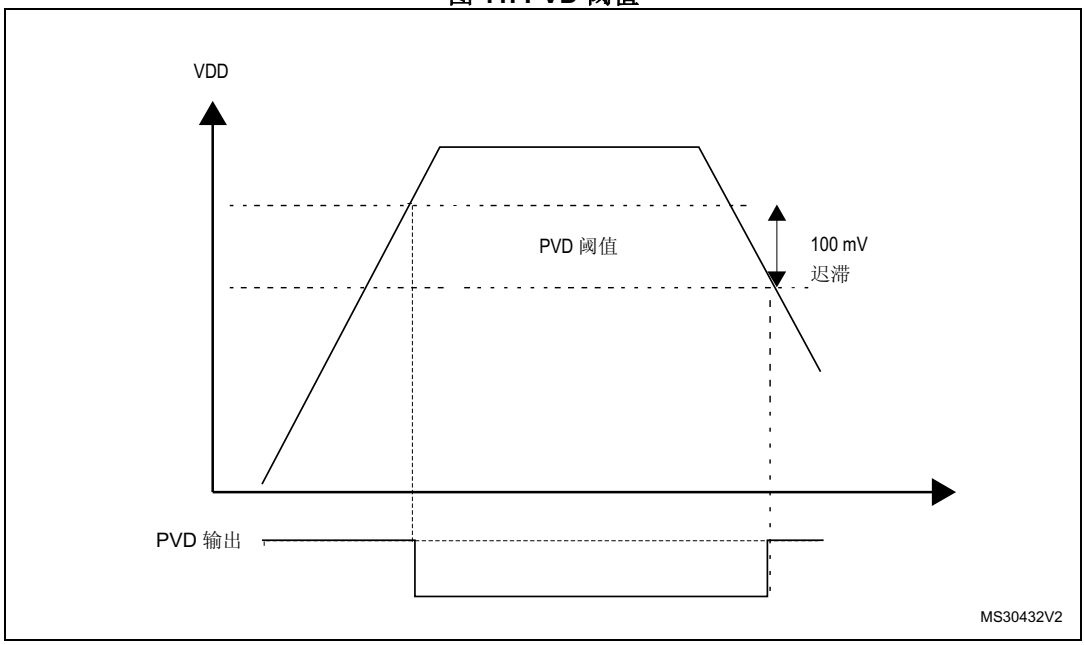
5.2.3 可编程电压检测器 (PVD)

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_CR)* 中的 PLS[2:0] 位所选的阈值进行比较。

通过将 PVDE 位置 1 来使能 PVD。

PWR 电源控制/状态寄存器 (PWR_CSR) 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 11. PVD 阈值



5.3 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。

器件有四种低功耗模式：

- 睡眠模式（带 FPU 的 Cortex®-M4 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）
- 批采集模式 (BAM)：器件处于睡眠模式，Flash 关闭，所需外设保持运行，仍可通过 DMA 进行数据传输。

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

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

仅当没有中断或事件挂起时，才可通过 WFI 或 WFE 进入低功耗模式。

退出低功耗模式

MCU 根据进入睡眠和停止模式的方式退出这两种低功耗模式：

- 如果使用 WFI 指令或从 ISR 返回进入低功耗模式，通过 NVIC 确认的任何外设中断都能唤醒器件。
- 如果使用 WFE 指令进入低功耗模式，MCU 将在有事件发生时立即退出低功耗模式。唤醒事件可通过以下方式产生：
  - NVIC IRQ 中断：

    当带 FPU 的 Cortex®-M4 系统控制寄存器中的 SEVONPEND = 0 时：在外设控制寄存器和 NVIC 中使能中断。当 MCU 从 WFE 恢复时，必须清零外设中断挂起位和 NVIC 外设 IRQ 通道挂起位（在 NVIC 中断清零挂起寄存器中）。只有当 NVIC 中断的优先级足够高时，才能唤醒和中断 MCU。

    当带 FPU 的 Cortex®-M4 系统控制寄存器中的 SEVONPEND = 1 时：在外设控制寄存器中使能中断，也可选择在 NVIC 中使能中断。当 MCU 从 WFE 恢复时，必须清零外设中断挂起位和使能的外设 IRQ 通道挂起位（在 NVIC 中断清零挂起寄存器中）。所有的 NVIC 中断都将唤醒 MCU，甚至是禁用的中断。只有当使能的 NVIC 中断的优先级足够高时，才能唤醒和中断 MCU。
  - 事件

    通过在事件模式下配置 EXTI 线来实现。当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置 1，不必清零 EXTI 外设的中断挂起位或 NVIC IRQ 通道挂起位。可能需要清零外设中的中断标志。

当发生外部复位（NRST 引脚）、IWDG 复位、使能的 WKUPx 引脚上出现上升沿或者触发 RTC 事件时，MCU 退出待机低功耗模式（请参见*图 254：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。
只有当已使能 NVIC 中断的优先级足够高时，才能唤醒和中断 MCU。

表 15. 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
睡眠和 BAM(1)（立即睡眠或退出时睡眠）	WFI 或从 ISR 返回	任意中断	CPU CLK 关闭对其他时钟或模拟时钟源无影响	无	开启
	WFE	唤醒事件			
停止	SLEEPDEEP 位 + WFI，从 ISR 或 WFE 返回	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_CR)*）
待机	PDDS 位 + SLEEPDEEP 位 + WFI，从 ISR 返回或 WFE	WKUP 引脚上升沿、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

1. 有关 BAM 进入和退出的特定要求，请参见*第 5.3.4 节：批采集模式*。

5.3.1 降低系统时钟速度

在运行模式下，可通过对预分频寄存器编程来降低系统时钟（SYSCLK、HCLK、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 6.3.3 节：RCC 时钟配置寄存器 (RCC_CFGR)*。

5.3.2 外设时钟门控

在运行模式下，可随时停止各外设和存储器的 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) 和 AHB2 外设时钟使能寄存器 (RCC_AHB2ENR) 进行控制（请参见*第 6.3.11 节：RCC AHB1 外设时钟使能寄存器 (RCC_AHB1ENR)*、*第 6.3.12 节：STM32F413xx 的 RCC AHB2 外设时钟使能寄存器 (RCC_AHB2ENR)* 和*第 6.3.14 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。

5.3.3 睡眠模式

进入睡眠模式

当带 FPU 的 Cortex®-M4 系统控制寄存器的 SLEEPDEEP 位清零时，将根据*进入低功耗模式*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 16* 和*表 17*。

注： *在进入睡眠模式前，必须清零所有的中断挂起位。*

退出睡眠模式

根据*退出低功耗模式*退出睡眠模式。

有关如何退出睡眠模式的详细信息，请参见*表 16* 和*表 17*。

表 16. 进入和退出立即睡眠

立即睡眠模式	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（对于 WFI）或事件（对于 WFE）挂起。 请参见带 FPU 的 Cortex®-M4 系统控制寄存器。
	从 ISR 返回，且： – SLEEPDEEP = 0 及 – SLEEPONEXIT = 1 – 没有中断挂起。 请参见带 FPU 的 Cortex®-M4 系统控制寄存器。

表 16. 进入和退出立即睡眠（续）

立即睡眠模式	说明
退出模式	如果使用 WFI 或从 ISR 返回进入： 中断：请参见*表 40：STM32F413/423 的向量表* 如果使用 WFE 进入且 SEVONPEND = 0 唤醒事件：请参见*第 10.2.3 节：唤醒事件管理* 如果使用 WFE 进入且 SEVONPEND = 1 中断（即使在 NVIC 中禁止时）：请参见*表 40：STM32F413/423 的向量表*）或唤醒事件（请参见*第 10.2.3 节：唤醒事件管理*）。
唤醒延迟	无

表 17. 进入和退出退出时睡眠

退出时睡眠	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（对于 WFI）或事件（对于 WFE）挂起。 请参见带 FPU 的 Cortex®-M4 系统控制寄存器。
	从 ISR 返回，且： – SLEEPDEEP = 0 及 – SLEEPONEXIT = 1 及 – 没有中断挂起。 请参见带 FPU 的 Cortex®-M4 系统控制寄存器。
退出模式	中断：请参见*表 40：STM32F413/423 的向量表*
唤醒延迟	无

5.3.4 批采集模式

进入 BAM

当带 FPU 的 Cortex®-M4 系统控制寄存器的 SLEEPDEEP 位清零时，将根据*进入低功耗模式*一节进入 BAM。

有关如何进入睡眠模式的详细信息，请参见*表 18* 和*表 19*。

进入睡眠模式前，必须用软件将 Flash 配置为在所需的低功耗模式下工作。在 BAM 期间，如果需要将数据从外设传输到 RAM，则必须在进入睡眠模式前使能 DMA。

退出 BAM

根据*退出低功耗模式*一节退出 BAM。

有关如何退出睡眠模式的详细信息，请参见*表 18* 和*表 19*。

从 BAM 唤醒后，如果从 Flash 重新开始执行代码，则必须先唤醒 Flash。

唤醒时间必须由内部 SRAM 中运行的软件进行管理。

表 18. 进入和退出立即 BAM

立即睡眠模式	说明
进入模式	将 Flash 设置为低功耗模式： – PWR_CR 寄存器的 FISSR/FMSSR 和 FPDS 位 WFI（等待中断）或 WFE（等待事件），且： – SLEEPDEEP = 0 及 – SLEEPONEXIT = 0 请参见带 FPU 的 Cortex®-M4 系统控制寄存器。
退出模式	如果使用 WFI 进入： 中断：请参见*表 40：STM32F413/423 的向量表* 如果使用 WFE 进入 唤醒事件：请参见*第 10.2.3 节：唤醒事件管理* 如果 Flash 需要唤醒时间，则必须将 PWR_CR 寄存器的 FISSR/FMSSR 位置 1
唤醒延迟	如果从 RAM 执行代码，则无延迟 从 Flash 重新开始执行代码前的低功耗模式 Flash 唤醒时间（请参见数据手册中电气特性部分的 Flash 唤醒时间）。

表 19. 进入和退出退出时 BAM

退出时睡眠	说明
进入模式	将 Flash 设置为低功耗模式： – PWR_CR 寄存器的 FISSR/FMSSR 和 FPDS 位 WFI（等待中断），且： – SLEEPDEEP = 0 及 – SLEEPONEXIT = 1 请参见带 FPU 的 Cortex®-M4 系统控制寄存器。
退出模式	中断：请参见*表 40：STM32F413/423 的向量表* 如果 Flash 需要唤醒时间，则必须将 PWR_CR 寄存器的 FISSR/FMSSR 位置 1
唤醒延迟	如果从内部 SRAM 执行代码，则无延迟 从 Flash 重新开始执行代码前的低功耗模式 Flash 唤醒时间（请参见数据手册中电气特性部分的 Flash 唤醒时间）。

注： *已通过添加 SRAM2 增强 BAM，允许通过 I 总线和 D 总线执行 SRAM 代码，从而提高了代码执行性能。*

5.3.5 停止模式

停止模式基于带 FPU 的 Cortex®-M4 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。

PWR_CR 寄存器中的某些设置可进一步降低功耗。Flash 处于掉电模式时，将器件从停止模式唤醒需要额外的启动延时（请参见*表 20：停止工作模式*和*第 5.4.1 节：PWR 电源控制寄存器 (PWR_CR)*）。

表 20. 停止工作模式

	停止模式	MRLV 位	LPLV 位	FPDS 位	LPDS 位	唤醒延迟
正常模式	STOP MR	0	-	0	0	HSI RC 启动时间
	STOP MRFPD	0	-	1	0	HSI RC 启动时间 + Flash 从深度掉电模式唤醒的时间
	STOP LP	0	0	0	1	HSI RC 启动时间 + 调压器从 LP 模式唤醒的时间
	STOP LPFPD	-	0	1	1	HSI RC 启动时间 + Flash 从深度掉电模式唤醒的时间 + 调压器从 LP 模式唤醒的时间
	STOP MRLV	1	-	-	0	HSI RC 启动时间 + Flash 从深度掉电模式唤醒的时间 + 主调压器从低压模式唤醒的时间
	STOP LPLV	-	1	-	1	HSI RC 启动时间 + Flash 从深度掉电模式唤醒的时间 + 调压器从低压 LP 模式唤醒的时间

进入停止模式

当带 FPU 的 Cortex®-M4 系统控制寄存器的 SLEEPDEEP 位置 1 时，将根据*进入低功耗模式*一节进入停止模式。

有关如何进入停止模式的详细信息，请参见*表 21*。

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。这可通过 *PWR 电源控制寄存器 (PWR_CR)* 的 LPDS 位进行配置。

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。

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 23.3 节*中的*第 23 节：窗口看门狗 (WWDG)*。
- 实时时钟 (RTC)：通过*第 6.3.23 节：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*第 6.3.24 节：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*第 6.3.23 节：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 也会产生功耗，除非在进入停止模式前将其禁止。要禁止该转换器，必须向 ADC_CR2 寄存器中的 ADON 位写入 0。

注： *如果应用程序需要在进入停止模式之前禁止外部时钟，必须首先禁止 HSEON 位并将系统时钟切换到 HSI。*

否则，如果在进入停止模式前 HSEON 位保持使能并且可以移除外部时钟（外部振荡器），则必须使能时钟安全系统 (CSS) 功能以检测任何外部振荡器故障并避免进入停止模式时出现故障行为。

退出停止模式

根据*退出低功耗模式*一节退出停止模式。

有关如何退出停止模式的详细信息，请参见*表 21*。

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。

当调压器在低功耗模式下工作时，将器件从停止模式唤醒将需要额外的延时。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。

表 21. 进入和退出停止模式

停止模式	说明
进入模式	WFI（等待中断）或 WFE（等待事件），且： – 没有中断（对于 WFI）或事件（对于 WFE）挂起， – 将带 FPU 的 Cortex®-M4 系统控制寄存器中的 SLEEPDEEP 位置 1， – 将电源控制寄存器 (PWR_CR) 中的 PDDS 位清零， – 通过配置 PWR_CR 中的 LPDS 位选择调压器模式。
	从 ISR 返回时： – 没有中断挂起， – 将带 FPU 的 Cortex®-M4 系统控制寄存器中的 SLEEPDEEP 位置 1， – SLEEPONEXIT = 1， – 将电源控制寄存器 (PWR_CR) 中的 PDDS 位清零。
	注： *要进入停止模式，所有 EXTI 线挂起位（在第 10.3.6 节：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 返回进入： 任何配置为中断模式的 EXTI 线（必须在 NVIC 中使能对应的 EXTI 中断向量）。中断源可以是外部中断或具有唤醒功能的外设。请参见*表 40：STM32F413/423 的向量表*。 如果使用 WFE 进入且 SEVONPEND = 0： 任何配置为事件模式的 EXTI 线。请参见*第 10.2.3 节：唤醒事件管理*。 如果使用 WFE 进入且 SEVONPEND = 1： – 任何配置为中断模式的 EXTI 线（即使在 NVIC 中禁止对应的 EXTI 中断向量）。中断源可以是外部中断或具有唤醒功能的外设。请参见*表 40：STM32F413/423 的向量表*。 – 唤醒事件：请参见*第 10.2.3 节：唤醒事件管理*
唤醒延迟	请参见*表 20：停止工作模式*

5.3.6 待机模式

待机模式下可达到最低功耗。待机模式基于带 FPU 的 Cortex®-M4 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 寄存器和 RTC 备份寄存器）和待机电路中的寄存器外，SRAM 和寄存器内容都将丢失（请参见*图 8*）。

进入待机模式

当带 FPU 的 Cortex®-M4 系统控制寄存器的 SLEEPDEEP 位置 1 时，将根据*进入低功耗模式*一节进入待机模式。
有关如何进入待机模式的详细信息，请参见*表 22*。

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 22.3 节*中的*独立看门狗 (IWDG)*。
- 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

根据*退出低功耗模式*一节退出待机模式。PWR_CR 中的 SBF 状态标志（请参见*第 5.4.2 节：PWR 电源控制/状态寄存器 (PWR_CSR)*）指示 MCU 已处于待机模式。从待机模式唤醒后，除 PWR_CR 外，所有寄存器都将复位。

有关如何退出待机模式的详细信息，请参见*表 22*。

表 22. 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： – 将带 FPU 的 Cortex®-M4 系统控制寄存器中的 SLEEPDEEP 位置 1， – 将电源控制寄存器 (PWR_CR) 中的 PDDS 位置 1， – 没有中断（对于 WFI）或事件（对于 WFE）挂起， – 将电源控制寄存器 (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、入侵或时间戳标志）对应的 RTC 标志清零
	从 ISR 返回，且： – 将带 FPU 的 Cortex®-M4 系统控制寄存器中的 SLEEPDEEP 位置 1，且 – SLEEPONEXIT = 1 及 – 将电源控制寄存器 (PWR_CR) 中的 PDDS 位置 1，且 – 没有中断挂起， – 将电源控制/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、入侵或时间戳标志）对应的 RTC 标志清零。
退出模式	WKUP 引脚上升沿、RTC 闹钟（闹钟 A 和闹钟 B）、RTC 唤醒事件、RTC 入侵事件、RTC 时间戳事件、NRST 引脚外部复位 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）
- RTC_AF1 引脚 (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/PC0/PC1)（如果使能）

调试模式

默认情况下，如果使用调试功能时应用程序将 MCU 置于停止模式或待机模式，调试连接将中断。这是因为带 FPU 的 Cortex®-M4 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 34.16.1 节：对低功耗模式的调试支持*。

5.3.7 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*第 6.3.23 节：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

如果在 PWR 唤醒标志 (WUTF) 清零之前将所选 RTC 复用功能置 1，则出现下一事件时无法检测到相关功能，因为检测操作只在信号上升沿到来时执行一次。

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) 将 PWR 唤醒 (WUF) 标志清零
  d) 使能 RTC 闹钟中断
  e) 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) 将 RTC 唤醒 (WUTF) 标志清零
  c) 将 PWR 唤醒 (WUF) 标志清零
  d) 使能 RTC 唤醒中断
  e) 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) 将 PWR 唤醒 (WUF) 标志清零
  d) 使能 RTC 入侵中断
  e) 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) 将 PWR 唤醒 (WUF) 标志清零
  d) 使能 RTC 时间戳中断
  e) 重新进入低功耗模式

5.4 电源控制寄存器

5.4.1 PWR 电源控制寄存器 (PWR_CR)

PWR power control register

偏移地址：0x00

复位值：0x0000 8000（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.	FISSR	FMSSR	Res.	Res.	Res.	Res.
										rw	rw				

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

位 31:22 保留，必须保持复位值。

位 21 **FISSR**：系统运行时停止 Flash 接口 (Flash Interface Stop while System Run)

0：Flash 接口时钟运行（默认值）。

1：Flash 接口时钟关闭。

注： 使用 Flash 本身执行时不能将该位置 1， 而应从 RAM 执行特定程序来实现。

位 20 **FMSSR**：Flash 睡眠系统运行 (Flash Memory Sleep System Run)

0：Flash 标准模式（默认值）

1：Flash 被硬件强制为停止或深度掉电模式（取决于 **FPDS** 位的值）。

注： 使用 Flash 本身执行时不能将该位置 1， 而应从 RAM 执行特定程序来实现。

位 19:16 保留，必须保持复位值。

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，请参见相应的数据手册）。

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，则会将调压器设置为级别 3，而与 VOS 寄存器内容无关。

00：保留（选择级别 3 模式）

01：级别 3 模式 <= 64 MHz

10：级别 2 模式（复位值）<= 84 MHz

11：级别 1 模式 <= 100 MHz

位 13 **ADCDC1**：

0：不起作用。

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 之间的电源电压范围内工作且关闭 Flash 预取指功能时，才可以设置此位。

位 12 保留，必须保持复位值。

位 11 **MRLVDS**：深度睡眠模式时的主调压器低压 (Main regulator Low Voltage in Deep Sleep)

0：器件处于停止模式时，主调压器为电压级别 3。

1：器件处于停止模式时，主调压器处于低压模式，Flash 处于深度睡眠模式。

位 10 **LPLVDS**：深度睡眠模式时的低功耗调压器低压 (Low-power regulator Low Voltage in Deep Sleep)

0：器件处于停止模式时，低功耗调压器开启。

1：器件处于停止模式时，如果将 LPDS 位置 1，则低功耗调压器处于低压模式，Flash 处于深度睡眠模式。

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_CSR 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：不可访问 RTC 和 RTC 备份寄存器。

1：可访问 RTC 和 RTC 备份寄存器。

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

000：2.2 V

001：2.3 V

010：2.4 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：不起作用。

1：将 SBF 待机标志位清零（写）。

位 2 **CWUF**：将唤醒标志清零 (Clear wakeup flag)

此位始终读为 0。

0：不起作用。

1：2 个系统时钟周期后将 WUF 唤醒标志清零

位 1 **PDDS**：掉电深度睡眠 (Power-down deepsleep)

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：停止模式下调压器开启。
1：停止模式下低功耗调压器开启。

5.4.2 PWR 电源控制/状态寄存器 (PWR_CSR)

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.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	VOS RDY	Res.	Res.	Res.	Res.	BRE	EWUP 1	EWUP 2	EWUP 3	Res.	Res.	BRR	PVDO	SBF	WUF
	r					rw	rw	rw	rw			r	r	r	r

位 31:15 保留，必须保持复位值。

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（用于保持备份域内容）。如果 BRE 复位，备份调压器关闭。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} 模式下会保持写入备份寄存器中的数据。
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 **EWUP1**：使能 WKUP1 引脚 (PA0) (Enable WKUP1 pin (PA0))
此位由软件置 1 和清零。
0：WKUP1 引脚用于通用 I/O。WKUP1 引脚上的事件不会从待机模式唤醒器件。
1：WKUP1 引脚用于从待机模式唤醒器件并被强制配置为输入下拉（WKUP1 引脚出现上升沿时从待机模式唤醒系统）。
注： 此位通过系统复位进行复位。

位 7 **EWUP2**：使能 WKUP2 引脚 (PC0) (Enable WKUP2 pin (PC0))
此位由软件置 1 和清零。
0：WKUP2 引脚用于通用 I/O。WKUP2 引脚上的事件不会从待机模式唤醒器件。
1：WKUP2 引脚用于从待机模式唤醒器件并被强制配置为输入下拉（WKUP2 引脚出现上升沿时从待机模式唤醒系统）。
注： 此位通过系统复位进行复位。

位 6 **EWUP3**：使能 WKUP3 引脚 (PC1) (Enable WKUP3 pin (PC1))

此位由软件置 1 和清零。

0：WKUP3 引脚用于通用 I/O。WKUP3 引脚上的事件不会从待机模式唤醒器件。

1：WKUP3 引脚用于从待机模式唤醒器件并被强制配置为输入下拉（WKUP3 引脚出现上升沿时从待机模式唤醒系统）。

注： 此位通过系统复位进行复位。

位 5:4 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_CR 寄存器中的 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 **WUF**：唤醒标志 (Wakeup flag)

该位通过硬件置 1，通过系统复位或将 PWR_CR 寄存器中的 CWUF 位置 1 清零。

0：未发生唤醒事件

1：收到唤醒事件，可能来自 WKUP 引脚、RTC 闹钟（闹钟 A 和闹钟 B）、RTC 入侵事件、RTC 时间戳事件或 RTC 唤醒事件。

注： 如果使能 WKUP 引脚（将 EWUP 位置 1）时 WKUP 引脚已为高电平，系统将检测到另一唤醒事件。

5.5 PWR 寄存器映射

下表对 PWR 寄存器进行了汇总。

表 23. 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_CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FISSR	FMSSR	Res.	Res.	Res.	Res.	VOS[1:0]		ADCDC1	Res.	MRLVDS	LPLVDS	FPDS	DBP	PLS[2:0]			PVDE	CSBF	CWUF	PDDS	LPDS
	Reset value											0	0					1	1	0		0	0	0	0	0	0	0	0	0	0	0	0
0x004	**PWR_CSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VOSRDY	Res.	Res.	Res.	Res.	BRE	EWUP1	EWUP2	EWUP3	Res.	Res.	BRR	PVDO	SBF	WUF
	Reset value																		0					0	0	0	0			0	0	0	0

有关寄存器边界地址的信息，请参见*第 54 页的第 2.2.2 节*。

6 STM32F413/423 的复位和时钟控制 (RCC)

6.1 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。

6.1.1 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，系统复位会将其他全部寄存器都复位为复位值。

只要发生以下事件之一，就会产生系统复位：

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，必须将带 FPU 的 Cortex®-M4 应用中断和复位控制寄存器中的 SYSRESETREQ 位置 1。有关详细信息，请参见带 FPU 的 Cortex®-M4 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。

有关用户选项字节的详细信息，请参见 STM32F413/423 Flash 编程手册，该手册可从 ST 销售办事处获取，也可以从 ST 网站 www.st.com 获得文档信息。

6.1.2 电源复位

只要发生以下事件之一，就会产生电源复位：

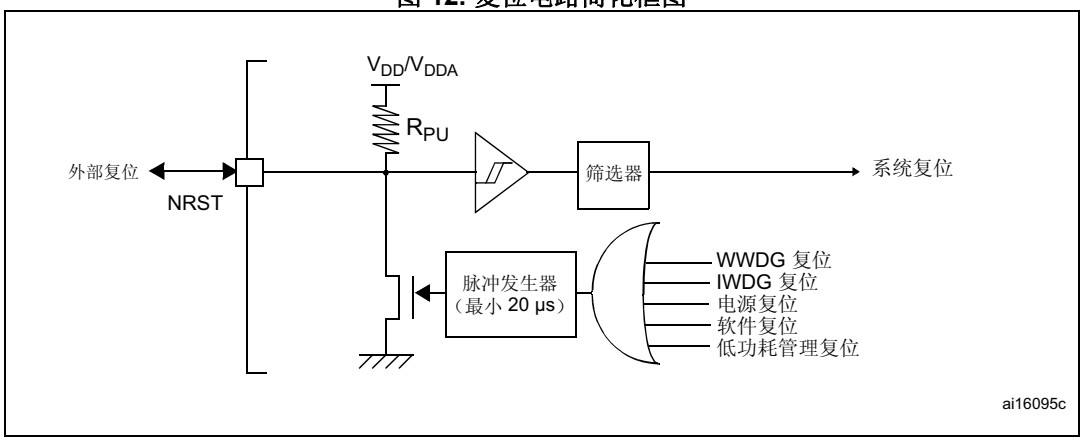
1. 上电/掉电复位（POR/PDR 复位）或欠压 (BOR) 复位
2. 在退出待机模式时

除备份域外，电源复位会将所有寄存器设为其复位值。

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 12. 复位电路简化框图



6.1.3 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。

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。

6.2 时钟

可以使用三种不同的时钟源来驱动系统时钟 (SYSCLK)：

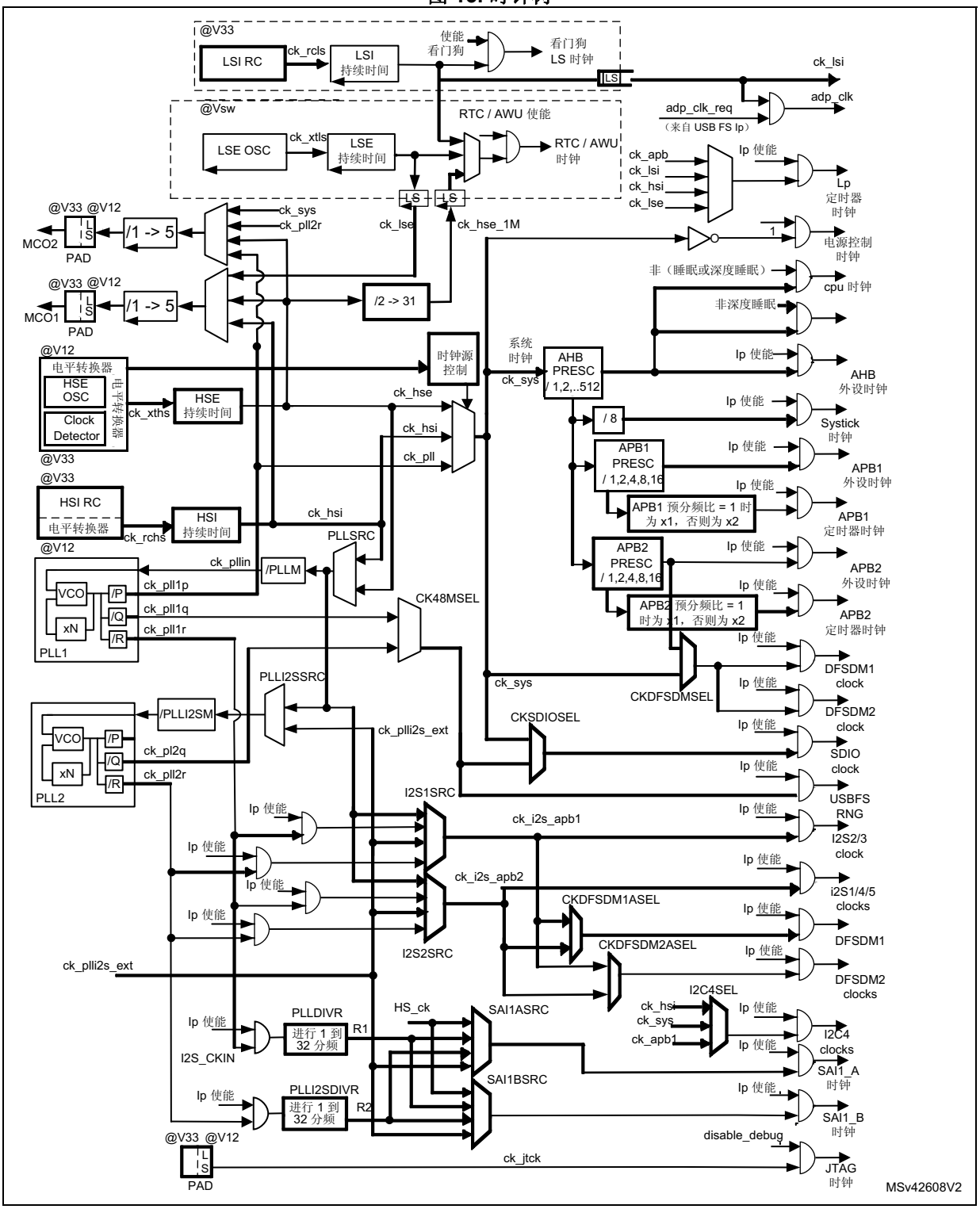
- HSI 振荡器时钟
- HSE 振荡器时钟
- 主 PLL (PLL) 时钟

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 13. 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。

时钟控制器为应用带来了高度的灵活性，用户在运行内核和外设时可选择使用外部晶振或者使用振荡器，既可采用最高的频率，也可为 USB OTG FS、I2S 和 SDIO 等需要特定时钟的外设保证合适的频率。

可通过多个预分频器配置 AHB 频率、高速 APB (APB2) 和低速 APB (APB1)。AHB 域的最大频率为 100 MHz。高速 APB2 域的最大允许频率为 100 MHz。低速 APB1 域的最大允许频率为 50 MHz

除以下时钟外，所有外设时钟均由系统时钟 (SYSCLK) 提供：

- 来自于特定 PLL 输出 (PLL48CLK) 的 USB OTG FS 时钟 (48 MHz) 和 SDIO 时钟 (≤ 48 MHz)
- 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 29.6.4 节：时钟发生器*。
- I2CFMP1 时钟，该时钟可由 HSI、SYSCLK 或 APB1 时钟提供。

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。

定时器时钟频率由硬件自动设置。根据 RCC_DCKCFGR 寄存器中 TIMPRE 位的取值，共分为两种情况：

- 如果 TIMPRE 位复位：

  如果将 APB 预分频器的分频系数配置为 1，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 域的频率的二倍：TIMxCLK = 2xPCLKx。
- 如果 TIMPRE 位置 1：

  如果将 APB 预分频器的分频系数配置为 1，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。

FCLK 充当带 FPU 的 Cortex®-M4 的自由运行时钟。有关详细信息，请参见带 FPU 的 Cortex®-M4 技术参考手册。

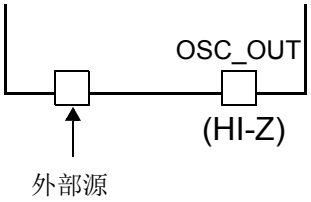
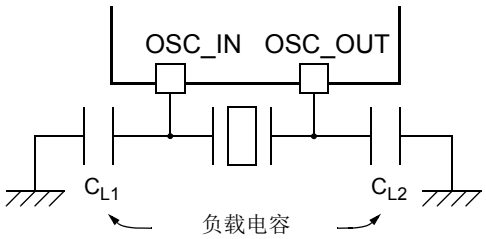
6.2.1 HSE 时钟

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 14. 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 14*。

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 14*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，硬件将该位置 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。

6.2.2 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 晶振快，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 RC 输出时钟才可以使用。

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116页的第6.2.7节：时钟安全系统 (CSS)*。

6.2.3 PLL 配置

STM32F413/423 器件具有两个 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 100 MHz）
  - 第二个输出用于生成 USB OTG FS (48 MHz)、RNG 和 SDIO (≤ 50 MHz) 时钟。
- 专用 PLL (PLLI2S) 用于生成精确时钟，从而在 I2S 接口实现高品质音频性能。

由于在 PLL 使能后不可更改 PLL 配置参数，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、P、Q 和倍频系数 N）。

PLLI2S 与主 PLL 使用相同的输入时钟（HSI 或 HSE）。但是，PLLI2S 具有专门的使能/禁止和分频系数配置位。请参见*第 6.3.1 节：RCC 时钟控制寄存器 (RCC_CR)*、*第 6.3.2 节：RCC PLL 配置寄存器 (RCC_PLLCFGR)* 和*第 6.3.26 节：RCC PLLI2S 配置寄存器 (RCC_PLLI2SCFGR)*。在 PLLI2S 使能后，配置参数便不能更改。

当进入停机和待机模式后，两个 PLL 将由硬件禁止；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，两个 PLL 也将由硬件禁止。*RCC PLL 配置寄存器 (RCC_PLLCFGR)* 和*RCC 时钟配置寄存器 (RCC_CFGR)* 可分别用于配置 PLL 和 PLLI2S。

6.2.4 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/日历或其他定时功能，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，硬件将该位置 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 引脚应保持为高阻态 (Hi-Z)。请参见*图 14*。

6.2.5 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，硬件将该位置 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。

6.2.6 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。

6.2.7 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 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 与带 FPU 的 Cortex®-M4 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 已使能，则在 HSE 发生故障时也会将其禁止。

6.2.8 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。
  - RTC 在系统复位后仍可获得时钟并保持正常工作。
- 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 6.2.5 节：LSI 时钟*。
- 如果使用 HSE 时钟作为 RTC 时钟：
  - 如果 V_{DD} 电源掉电或者内部调压器关闭（切断 1.2 V 域的供电），则 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。*

6.2.9 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。

6.2.10 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。

MCO 输出时钟不得超过 100 MHz（最大 I/O 速度）。

6.2.11 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 15* 和*图 16* 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。

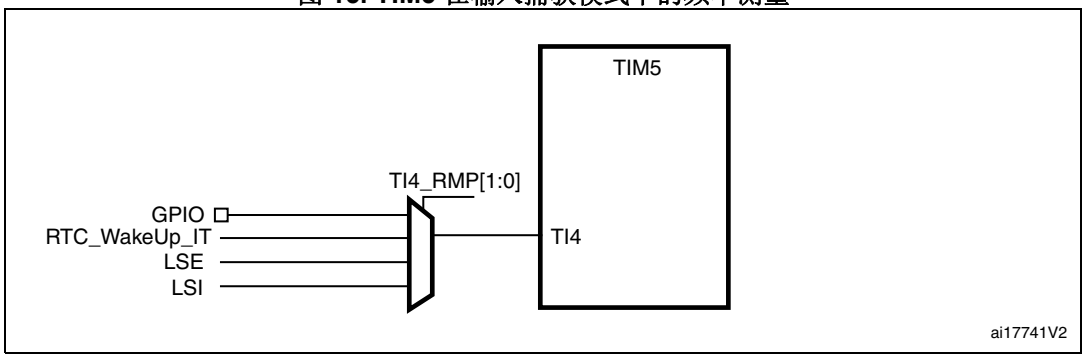
其基本原理是基于相对的测量（例如，HSI/LSE 比）：因此，精度与两个时钟源之比紧密相关。比率越大，测量效果越好。

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. 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。

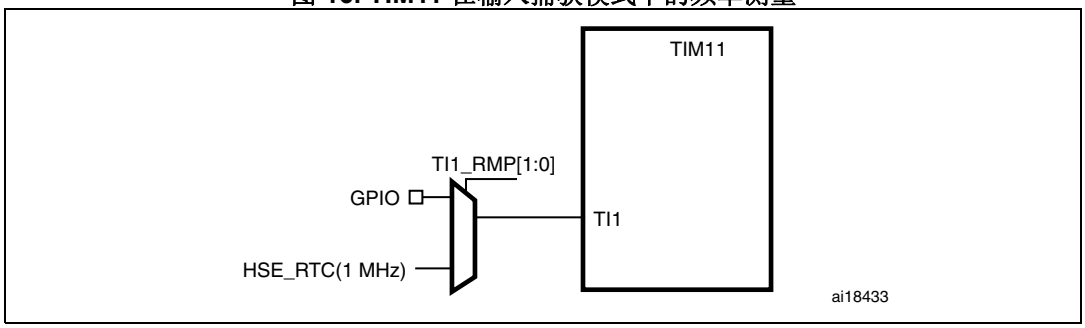
图 15. TIM5 在输入捕获模式下的频率测量



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 16. TIM11 在输入捕获模式下的频率测量



6.3 RCC 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.2 节：寄存器相关缩写词列表*。

6.3.1 RCC 时钟控制寄存器 (RCC_CR)

RCC clock control register

偏移地址：0x00

复位值：0x0000 XX81，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.	Res.	Res.	PLLI2S RDY	PLLI2S ON	PLLRDY	PLLON	Res.	Res.	Res.	Res.	CSS ON	HSE BYP	HSE RDY	HSE ON
				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.	HSI RDY	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 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，用于在进入停止或待机模式时停止 HSE 振荡器。如果 HSE 振荡器直接或间接用作系统时钟，则该位不可复位。

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

6.3.2 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：

- $f_{(VCO\ 时钟)} = f_{(PLL\ 时钟输入)} \times (PLLN / PLLM)$
- $f_{(PLL\ 常规时钟输出)} = f_{(VCO\ 时钟)} / PLLP$
- $f_{(USB\ OTG\ FS、SDIO、RNG\ 时钟输出)} = f_{(VCO\ 时钟)} / PLLQ$
- $f_{(I2S,\ DFSDM\ 时钟输出)} = f_{(VCO\ 时钟)} / PLLR$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	PLLR[2:0]			PLLQ[3:0]				Res.	PLLSRC	Res.	Res.	Res.	Res.	PLLP[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
Res.	PLLN[8:0]									PLLM[5:0]					
	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 30:28 **PLLR[2:0]**：适用于 I2S 和 DFSDM 时钟的主 PLL (PLL) 分频系数 (Main PLL (PLL) division factor for I2S, DFSDM clocks)

由软件置 1 和清零，用于控制时钟的频率。仅在 PLL 已禁止时才能写入这些位。

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

011：PLLR = 3

...

111：PLLR = 7

位 27:24 **PLLQ[3:0]**：适用于 USB OTG FS、SDIO 和随机数发生器时钟的主 PLL (PLL) 分频系数 (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 SDIO 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 SDIO 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 100 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) multiplication factor for VCO)

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 100 和 432 MHz 之间。（另请参见*第 6.3.26 节：RCC PLLI2S 配置寄存器 (RCC_PLLI2SCFGR)*）

VCO 输出频率 = VCO 输入频率 × PLLN，其中 50 ≤ PLLN ≤ 432
000000000：PLLN = 0，错误配置
000000001：PLLN = 1，错误配置
...
000110010：PLLN = 50
...
001100011：PLLN = 99
001100100：PLLN = 100
...
110110000：PLLN = 432
110110001：PLLN = 433，错误配置
...
111111111：PLLN = 511，错误配置

注： *对于 1 MHz 以上的 VCO 输入频率，可以使用倍频系数，但应注意必须满足上述指定的 VCO 输出频率范围。*

位 5:0 **PLLM[5:0]**：适用于主 PLL (PLL) 输入时钟的分频系数 (Division factor for the main PLL (PLL) 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

6.3.3 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 ≤ 等待状态 ≤ 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[1:0]		MCO2 PRE[2:0]			MCO1 PRE[2:0]			Res.	MCO1[1:0]		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]				SWS[1:0]		SW[1:0]	
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 29:27 **MCO2PRE[1:0]**：MCO2 预分频器 (MCO2 prescaler)

由软件置 1 和清零，用于配置 MCO2 的预分频器。对此预分频器进行修改可能会对 MCO2 造成干扰。强烈建议仅在复位后且在使能外部振荡器和 PLL 之前进行此分频器的更改。

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

位 26:24 **MCO1PRE[1:0]**：MCO1 预分频器 (MCO1 prescaler)

由软件置 1 和清零，用于配置 MCO1 的预分频器。对此预分频器进行修改可能会对 MCO1 造成干扰。强烈建议仅在复位后且在使能外部振荡器和 PLL 之前进行此分频器的更改。

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

位 23 保留，始终读为 0。

位 22:21 **MCO1[1:0]**：微控制器时钟输出 1 (Microcontroller clock output 1)

由软件置 1 和清零。时钟源选择可能会造成对 MCO1 的干扰。强烈建议仅在复位后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[4:0]**：适用于 RTC 时钟的 HSE 分频系数 (HSE division factor for RTC clocks)

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[2:0]**：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 100 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[2:0]**：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 50 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[3:0]**：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[1:0]**：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[1:0]**：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

6.3.4 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	Res.	PLLI2S RDYC	PLL RDYC	HSE RDYC	HSI RDYC	LSE RDYC	LSI RDYC
								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.	PLLI2S RDYIE	PLL RDYIE	HSE RDYIE	HSI RDYIE	LSE RDYIE	LSI RDYIE	CSSF	Res.	PLLI2S RDYF	PLL RDYF	HSE RDYF	HSI RDYF	LSE RDYF	LSI RDYF
		rw	rw	rw	rw	rw	rw	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：不起作用
1：将 CSSF 标志清零

位 22 保留，必须保持复位值。

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：不起作用
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：不起作用
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：不起作用
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：不起作用
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：不起作用
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：不起作用
1：已将 LSIRDYF 清零

位 15:14 保留，必须保持复位值。

位 13 **PLLI2SRDYIE**：PLLI2S 就绪中断使能 (PLLI2S ready interrupt enable)
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)
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)
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)
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)
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)
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由硬件置 1。
CSSC 位置 1 时由软件清零。
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 保留，必须保持复位值。

位 5 **PLLI2SRDYF**：PLLI2S 就绪中断标志 (PLLI2S ready interrupt flag)
当 PLLI2S 锁定并且 PLLI2SRDYDIE 置 1 时由硬件置 1。
PLLRI2SDYC 位置 1 时由软件清零。
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 锁定并且 PLLRDYDIE 置 1 时由硬件置 1。
PLLRDYC 位置 1 时由软件清零。
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)
当外部高速时钟变为稳定且 HSERDYDIE 置 1 时由硬件置 1。
HSERDYC 位置 1 时由软件清零。
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)
当内部高速时钟变为稳定且 HSIRDYDIE 置 1 时由硬件置 1。
HSIRDYC 位置 1 时由软件清零。
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)
当外部低速时钟变为稳定且 LSERDYDIE 置 1 时由硬件置 1。
LSERDYC 位置 1 时由软件清零。
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)
当内部低速时钟变为稳定且 LSIRDYDIE 置 1 时由硬件置 1。
LSIRDYC 位置 1 时由软件清零。
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

6.3.5 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA2 RST	DMA1 RST	Res.	Res.	Res.	Res.	Res.
									rw	rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRCRST	Res.	Res.	Res.	Res.	GPIOH RST	GPIOG RST	GPIOF RST	GPIOE RST	GPIOD RST	GPIOC RST	GPIOB RST	GPIOA RST
			rw					rw	rw	rw	rw	rw	rw	rw	rw

位 31:23 保留，必须保持复位值。

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

位 21 **DMA1RST**：DMA1 复位 (DMA1 reset)
由软件置 1 和清零。
0：不复位 DMA1
1：复位 DMA1

位 20:13 保留，必须保持复位值。

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

位 11:8 保留，必须保持复位值。

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

6.3.6 STM32F413xx 的 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	Res.	Res.	Res.	Res.	Res.	Res.
								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 **RNGSRST**：RNG 模块复位 (RNG module reset)
由软件置 1 和清零。
0：不复位 RNG 模块
1：复位 RNG 模块

位 5:0 保留，必须保持复位值。

6.3.7 STM32F423xx 的 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.	OTGFSRST	RNGRST	Res.	CRYPRST	Res.	Res.	Res.	Res.
								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 **RNGSRST**：RNG 模块复位 (RNG module reset)
由软件置 1 和清零。
0：不复位 RNG 模块
1：复位 RNG 模块

位 5 保留，必须保持复位值。

位 4 **CRYPRST**：CRYP 模块复位 (CRYP module reset)
由软件置 1 和复位。
0：不复位 CRYP 模块
1：复位 CRYP 模块

位 3:0 保留，必须保持复位值。

6.3.8 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	FSMC RST
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **QSPIRST**：QUADSPI 模块复位 (QUADSPI module reset)
由软件置 1 和清零。
0：不复位 QUADSPI 模块
1：复位 QUADSPI 模块

位 0 **FSMCRST**：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 FSMC 模块
1：复位 FSMC 模块

6.3.9 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
UART8 RST	UART7 RST	DAC RST	PWR RST	CAN3 RST	CAN2 RST	CAN1 RST	I2C4 RST	I2C3 RST	I2C2 RST	I2C1 RST	UART5 RST	UART4 RST	USART3 RST	USART2 RST	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3 RST	SPI2 RST	Res.	Res.	WWDG RST	Res.	LPTIMER1 RST	TIM14 RST	TIM13 RST	TIM12 RST	TIM7 RST	TIM6 RST	TIM5 RST	TIM4 RST	TIM3 RST	TIM2 RST
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 和复位。
0：不复位 UART8
1：复位 UART8

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

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

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

位 27 **CAN3RST**：CAN3 复位 (CAN3 reset)
由软件置 1 和复位。
0：不复位 CAN3
1：复位 CAN3

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 和复位。
0：不复位 I2C4
1：复位 I2C4

位 23 **I2C3RST**：I2C3 复位 (I2C3 reset)
由软件置 1 和复位。
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 和复位。
0：不复位 I2C1
1：复位 I2C1

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

位 19 **UART4RST**：UART4 复位 (UART4 reset)
由软件置 1 和复位。
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 保留，必须保持复位值。

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 **LPTIMER1RST**：LPTimer1 复位 (LPTimer1 reset)
由软件置 1 和复位。
0：不复位定时器 1
1：复位定时器 1

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

6.3.10 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.	Res.	DFSDM2 RST	DFSDM1 RST	Res.	SAI1 RST	Res.	SPI5 RST	Res.	TIM11 RST	TIM10 RST	TIM9 RST
						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	SDIO RST	Res.	Res.	ADC RST	UART10 RST	UART9 RST	USART6 RST	USART1 RST	Res.	Res.	TIM8 RST	TIM1 RST
	rw	rw	rw	rw			rw	rw	rw	rw	rw			rw	rw

位 31:26 保留，必须保持复位值。

位 25 **DFSDM2RST**：DFSDM2 复位 (DFSDM2 reset)
由软件置 1 和清零。
0：不复位 DFSDM2
1：复位 DFSDM2

位 24 **DFSDM1RST**：DFSDM1 复位 (DFSDM1 reset)
由软件置 1 和清零。
0：不复位 DFSDM1
1：复位 DFSDM1

位 23 保留，始终读为 0。

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

位 21 保留，始终读为 0。

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)
由软件置 1 和清零。
0：不复位 TIM11
1：复位 TIM11

位 17 **TIM10RST**：TIM10 复位 (TIM10 reset)
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)
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)
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)
由软件置 1 和复位。
0：不复位 SPI4
1：复位 SPI4

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

位 11 **SDIORST**：SDIO 复位 (SDIO reset)
由软件置 1 和清零。
0：不复位 SDIO 模块
1：复位 SDIO 模块

位 10:9 保留，必须保持复位值。

位 8 **ADCRST**：ADC 接口复位 (ADC interface reset)
由软件置 1 和清零。
0：不复位 ADC 接口
1：复位 ADC 接口

位 7 **UART10RST**：UART10 复位 (UART10 reset)
由软件置 1 和清零。
0：不复位 UART10
1：复位 UART10

位 6 **UART9RST**：UART9 复位 (UART9 reset)
由软件置 1 和清零。
0：不复位 UART9
1：复位 UART9

位 5 **USART6RST**：USART6 复位 (USART6 reset)
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)
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)
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)
由软件置 1 和清零。
0：不复位 TIM1
1：复位 TIM1

6.3.11 RCC AHB1 外设时钟使能寄存器 (RCC_AHB1ENR)

RCC AHB1 peripheral clock enable register

偏移地址：0x30

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.	DMA2EN	DMA1EN	Res.	Res.	Res.	Res.	Res.
									rw	rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRCEN	Res.	Res.	Res.	Res.	GPIOHEN	GPIOGEN	GPIOFEN	GPIOEEN	GPIODEN	GPIOCEN	GPIOBEN	GPIOAEN
			rw					rw	rw	rw	rw	rw	rw	rw	rw

位 31:23 保留，必须保持复位值。

位 22 **DMA2EN**：DMA2 时钟使能 (DMA2 clock enable)
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)
由软件置 1 和清零。
0：禁止 DMA1 时钟
1：使能 DMA1 时钟

位 20:13 保留，必须保持复位值。

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

位 11:8 保留，必须保持复位值。

位 7 **GPIOHEN**：IO 端口 H 时钟使能 (IO port H clock enable)
由软件置 1 和复位。
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)
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)
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)
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)
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)
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)
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)
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

6.3.12 STM32F413xx 的 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	Res.	Res.	Res.	Res.	Res.	Res.
								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)
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**：RNG 时钟使能 (RNG clock enable)
由软件置 1 和清零。
0：禁止 RNG 时钟
1：使能 RNG 时钟

位 5:0 保留，始终读为 0。

6.3.13 STM32F423xx 的 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	MGEN	Res.	CRYPEN	Res.	Res.	Res.	Res.
								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)
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**：RNG 时钟使能 (RNG clock enable)
由软件置 1 和清零。
0：禁止 RNG 时钟
1：使能 RNG 时钟

位 5 保留，始终读为 0。

位 4 **CRYPEN**：CRYP 时钟使能 (CRYP clock enable)
由软件置 1 和复位。
0：禁止 CRYP 时钟
1：使能 CRYP 时钟

位 3:0 保留，始终读为 0。

6.3.14 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.	QSPI EN	FSMC EN
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **QSPIEN**：QUADSPI 存储器控制器模块时钟使能 (QUADSPI memory controller module clock enable)

由软件置 1 和清零。
0：禁止 QUADSPI 时钟
1：使能 QUADSPI 时钟

位 0 **FSMCEN**：灵活的存储器控制器模块时钟使能 (Flexible memory controller module clock enable)

由软件置 1 和清零。
0：禁止 FSMC 模块时钟
1：使能 FSMC 模块时钟

6.3.15 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 0400

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	CAN3 EN	CAN2 EN	CAN1 EN	I2CFMP1 EN	I2C3 EN	I2C2 EN	I2C1 EN	UART4 EN	UART4 EN	USART3 EN	USART2 EN	Res.
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3 EN	SPI2 EN	Res.	Res.	WWDG EN	RTCAPB EN	LPTIMER1 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	rw

位 31 **UART8EN**：UART8 时钟使能 (UART8 clock enable)
由软件置 1 和复位。
0：禁止 UART8 时钟
1：使能 UART8 时钟

位 30 **UART7EN**：UART7 时钟使能 (UART7 clock enable)
由软件置 1 和复位。
0：禁止 UART7 时钟
1：UART7 时钟使能

位 29 **DACEN**：DAC 时钟使能 (DAC clock enable)
由软件置 1 和复位。
0：禁止 DAC 时钟
1：使能 DAC 时钟

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

位 27 **CAN3EN**：CAN 3 时钟使能 (CAN 3 clock enable)
此位由软件置 1 和清零。
0：禁止 CAN 3 时钟
1：使能 CAN 3 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 **I2CFMP1EN**：I2CFMP1 时钟使能 (I2CFMP1 clock enable)
此位由软件置 1 和清零。
0：禁止 I2CFMP1 时钟
1：使能 I2CFMP1 时钟

位 23 **I2C3EN**：I2C3 时钟使能 (I2C3 clock enable)
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)
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)
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)
由软件置 1 和复位。
0：禁止 UART5 时钟
1：使能 UART5 时钟

位 19 **UART4EN**：UART4 时钟使能 (UART4 clock enable)
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)
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)
由软件置 1 和清零。
0：禁止 USART2 时钟
1：使能 USART2 时钟

位 16 保留，必须保持复位值。

位 15 **SPI3EN**：SPI3 时钟使能 (SPI3 clock enable)
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)
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)
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 **RTC**：APB 时钟使能 (APB clock enable)
由软件置 1 和清零。
0：禁止 RTC APB 时钟
1：使能 RTC APB 时钟（默认值）。

位 9 **LPTIMER1EN**：LPTimer 1 时钟使能 (LPTimer 1 clock enable)
由软件置 1 和复位。
0：禁止 LPTimer 1 时钟
1：使能 LPTimer 1 时钟

位 8 **TIM14EN**：TIM14 复位 (TIM14 reset)
由软件置 1 和清零。
0：不复位 TIM14
1：复位 TIM14

位 7 **TIM13EN**：TIM13 复位 (TIM13 reset)
由软件置 1 和清零。
0：不复位 TIM13
1：复位 TIM13

位 6 **TIM12EN**：TIM12 复位 (TIM12 reset)
由软件置 1 和清零。
0：不复位 TIM12
1：复位 TIM12

位 5 **TIM7EN**：TIM7 复位 (TIM7 reset)
由软件置 1 和清零。
0：不复位 TIM7
1：复位 TIM7

位 4 **TIM6EN**：TIM6 复位 (TIM6 reset)
由软件置 1 和清零。
0：不复位 TIM6
1：复位 TIM6

位 3 **TIM5EN**：TIM5 时钟使能 (TIM5 clock enable)
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)
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)
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)
由软件置 1 和清零。
0：禁止 TIM2 时钟
1：使能 TIM2 时钟

6.3.16 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 8000

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.	DFSDM2EN	DFSDM1EN	Res.	SAI1EN	Res.	SPI5EN	Res.	TIM11EN	TIM10EN	TIM9EN
							rw				rw		rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTITEN	SYSCFGEN	SPI4EN	SPI1EN	SDIOEN	Res.	Res.	ADC1EN	UART10EN	UART9EN	USART6EN	USART1EN	Res.	Res.	TIM8EN	TIM1EN
rw	rw	rw	rw	rw			rw	rw	rw	rw	rw			rw	rw

位 31:26 保留，始终读为 0。

位 25 **DFSDM2EN**：DFSDM2 时钟使能 (DFSDM2 clock enable)
由软件置 1 和清零
0：禁止 DFSDM2 时钟
1：使能 DFSDM2 时钟

位 24 **DFSDM1EN**：DFSDM1 时钟使能 (DFSDM1 clock enable)
由软件置 1 和清零
0：禁止 DFSDM1 时钟
1：使能 DFSDM1 时钟

位 23 保留，始终读为 0。

位 22 **SAI1EN**：SAI1 时钟使能 (SAI1 clock enable)
由软件置 1 和清零。
0：禁止 SAI1 时钟
1：使能 SAI1 时钟

位 21 保留，始终读为 0。

位 20 **SPI5EN**：SPI5 时钟使能 (SPI5 clock enable)
由软件置 1 和清零
0：禁止 SPI5 时钟
1：使能 SPI5 时钟

位 19 保留，始终读为 0。

位 18 **TIM11EN**：TIM11 时钟使能 (TIM11 clock enable)
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)
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)
由软件置 1 和清零。
0：禁止 TIM9 时钟
1：使能 TIM9 时钟

位 15 **EXITEN**：Extit Apb sysctrl pfree 时钟使能 (Extit Apb sysctrl pfree clock enable)
由软件置 1 和清零。
0：禁止 Extit Apb sysctrl pfree 时钟
1：使能 Extit Apb sysctrl pfree 时钟

位 14 **SYSCFGEN**：系统配置控制器时钟使能 (System configuration controller clock enable)
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)
由软件置 1 和复位。
0：禁止 SPI4 时钟
1：使能 SPI4 时钟

位 12 **SPI1EN**：SPI1 时钟使能 (SPI1 clock enable)
由软件置 1 和清零。
0：禁止 SPI1 时钟
1：使能 SPI1 时钟

位 11 **SDIOEN**：SDIO 时钟使能 (SDIO clock enable)
由软件置 1 和清零。
0：禁止 SDIO 模块时钟
1：使能 SDIO 模块时钟

位 8 **ADC1EN**：ADC1 时钟使能 (ADC1 clock enable)
由软件置 1 和清零。
0：禁止 ADC1 时钟
1：使能 ADC1 时钟

位 7 **UART10EN**：UART10 时钟使能 (UART10 clock enable)
由软件置 1 和清零。
0：禁止 UART10 时钟
1：使能 UART10 时钟

位 6 **UART9EN**：UART9 时钟使能 (UART9 clock enable)
由软件置 1 和清零。
0：禁止 UART9 时钟
1：使能 UART9 时钟

位 5 **USART6EN**：USART6 时钟使能 (USART6 clock enable)
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)
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)
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)
由软件置 1 和清零。
0：禁止 TIM1 时钟
1：使能 TIM1 时钟

6.3.17 低功耗模式下的 RCC AHB1 外设时钟使能寄存器 (RCC_AHB1LPENR)

RCC AHB1 peripheral clock enable in low power mode register

偏移地址：0x50

复位值：0x0063 90FF

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.	DMA2 LPEN	DMA1 LPEN	Res.	Res.	Res.	SRAM2 LPEN	SRAM1 LPEN
									rw	rw					rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLITF LPEN	Res.	Res.	CRC LPEN	Res.	Res.	Res.	Res.	GPIOH LPEN	GPIOG LPEN	GPIOF LPEN	GPIOE LPEN	GPIOD LPEN	GPIOC LPEN	GPIOB LPEN	GPIOA LPEN
rw			rw					rw	rw	rw	rw	rw	rw	rw	rw

位 31:23 保留，必须保持复位值。

位 22 **DMA2LPEN**：睡眠模式期间的 DMA2 时钟使能 (DMA2 clock enable during Sleep mode)
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)
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 20:18 保留，必须保持复位值。

位 17 **SRAM2LPEN**：睡眠模式期间的 SRAM2 接口时钟使能 (SRAM2 interface clock enable during Sleep mode)
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)
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)

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 14:13 保留，必须保持复位值。

位 12 **CRCLPEN**：睡眠模式期间的 CRC 时钟使能 (CRC clock enable during Sleep mode)

由软件置 1 和清零。

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 11:8 保留，必须保持复位值。

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)

由软件置 1 和复位。

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)

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)

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)

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)

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)

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)

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

6.3.18 STM32F413xx 的低功耗模式下的 RCC AHB2 外设时钟使能寄存器 (RCC_AHB2LPENR)

RCC AHB2 peripheral clock enable in low power mode register

偏移地址：0x54

复位值：0x0000 00C0

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	Res.	Res.	Res.	Res.	Res.	Res.
								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)

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 RNG 时钟使能 (RNG clock enable during Sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 RNG 时钟

1：睡眠模式期间使能 RNG 时钟

位 5:0 保留，必须保持复位值。

6.3.19 STM32F423xx 的低功耗模式下的 RCC AHB2 外设时钟使能寄存器 (RCC_AHB2LPENR)

RCC AHB2 peripheral clock enable in low power mode register

偏移地址：0x54

复位值：0x0000 00D0

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	Res.	CRYP LPEN	Res.	Res.	Res.	Res.
								rw	rw		rw				

位 31:8 保留，必须保持复位值。

位 7 **OTGSLPEN**：睡眠模式期间的 USB OTG FS 时钟使能 (USB OTG FS clock enable during Sleep mode)

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 RNG 时钟使能 (RNG clock enable during sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 RNG 时钟

1：睡眠模式期间使能 RNG 时钟

位 5 保留，始终读为 0。

位 4 **CRYPLPEN**：睡眠模式期间的 CRYPG 时钟使能 (CRYPG clock enable during sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 CRYP 时钟

1：睡眠模式期间使能 CRYP 时钟

位 3:0 保留，必须保持复位值。

6.3.20 低功耗模式下的 RCC AHB3 外设时钟使能寄存器 (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	FSMC LPEN
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **QSPILPEN**：睡眠模式期间的 QUADSPI 存储器控制器模块时钟使能 (QUADSPI memory controller module clock enable during Sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 QUADSPI 模块时钟

1：睡眠模式期间使能 QUADSPI 模块时钟

位 0 **FSMCLPEN**：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)

由软件置 1 和清零。

0：睡眠模式期间禁止 FSMC 时钟

1：睡眠模式期间使能 FSMC 时钟

6.3.21 低功耗模式下的 RCC APB1 外设时钟使能寄存器 (RCC_APB1LPENR)

RCC APB1 peripheral clock enable in low power mode register

偏移地址：0x60

复位值：0xFFFF CFFF

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	DACLP LPEN	PWR LPEN	CAN3 LPEN	CAN2 LPEN	CAN1 LPEN	I2C4 LPEN	I2C3 LPEN	I2C2 LPEN	I2C1 LPEN	UART5 LPEN	UART4 LPEN	USART3 LPEN	USART2 LPEN	Res.
			rw		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	RTCAPB LPEN	IPTIMER1 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	rw	rw

位 31 **UART8LPEN**：睡眠模式期间的 UART8 时钟使能 (UART8 clock enable during Sleep mode)
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)
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 时钟使能 (DAC clock enable during Sleep mode)
由软件置 1 和清零。
0：睡眠模式期间禁止 DAC 时钟
1：睡眠模式期间使能 DAC 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)
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 **CAN3LPEN**：睡眠模式期间的 CAN3 时钟使能 (CAN3 clock enable during Sleep mode)
由软件置 1 和清零。
0：睡眠模式期间禁止 CAN3 时钟
1：睡眠模式期间使能 CAN3 时钟

位 26 **CAN2LPEN**：睡眠模式期间的 CAN2 时钟使能 (CAN2 clock enable during Sleep mode)
由软件置 1 和清零。
0：睡眠模式期间禁止 CAN2 时钟
1：睡眠模式期间使能 CAN2 时钟

位 25 **CAN1LPEN**：睡眠模式期间的 CAN1 时钟使能 (CAN1 clock enable during Sleep mode)
由软件置 1 和清零。
0：睡眠模式期间禁止 CAN1 时钟
1：睡眠模式期间使能 CAN1 时钟

位 24 **I2CFMP1LPEN**：睡眠模式期间的 I2CFMP1 时钟使能 (I2CFMP1 clock enable during Sleep mode)
由软件置 1 和清零。
0：睡眠模式期间禁止 I2CFMP1 时钟
1：睡眠模式期间使能 I2CFMP1 时钟

位 23 **I2C3LPEN**：睡眠模式期间的 I2C3 时钟使能 (I2C3 clock enable during Sleep mode)
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)
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)
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)
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)
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)
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)
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 保留，必须保持复位值。

位 15 **SPI3LPEN**：睡眠模式期间的 SPI3 时钟使能 (SPI3 clock enable during Sleep mode)
由软件置 1 和清零。
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)
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)
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 **RTCAPBEN**：睡眠模式期间的 RTC APB 时钟使能 (RTC APB clock enable during Sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 RTC APB 时钟

1：睡眠模式期间使能 RTC APB 时钟

位 9 **LPTIMER1LPEN**：睡眠模式期间的 LPTimer 1 时钟使能 (LPTimer 1 clock enable during Sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 LPTimer 1 时钟

1：睡眠模式期间使能 LPTimer 1 时钟

位 8 **TIM14LPEN**：睡眠模式期间的 TIM14 时钟使能 (TIM14 clock enable during Sleep mode)

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)

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)

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)

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)

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)

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)

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)

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)

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

6.3.22 低功耗模式下的 RCC APB2 外设时钟使能寄存器 (RCC_APB2LPENR)

RCC APB2 peripheral clock enabled in low power mode register

偏移地址：0x64

复位值：0x0317 F9F3h

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.	DFSDM2 LPEN	DFSDM1 LPEN	Res.	SAI1 IPEN	Res.	SPI5 LPEN	Res.	TIM11 LPEN	TIM10 LPEN	TIM9 LPEN
							rw				rw		rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTIT LPEN	SYSCFG LPEN	SPI4LPEN	SPI1 LPEN	SDIO LPEN	Res.	Res.	ADC1 LPEN	UART10 LPEN	UART9 LPEN	USART6 LPEN	USART1 LPEN	Res.	Res.	TIM8 LPEN	TIM1 LPEN
rw	rw	rw	rw	rw			rw	rw	rw	rw	rw			rw	rw

位 31:26 保留，必须保持复位值。

位 25 **DFSDM2LPEN**：睡眠模式期间的 DFSDM2 时钟使能 (DFSDM2 clock enable during Sleep mode)

此位由软件置 1 和清零

0：睡眠模式期间禁止 DFSDM2 时钟

1：睡眠模式期间使能 DFSDM2 时钟

位 24 **DFSDM1LPEN**：睡眠模式期间的 DFSDM1 时钟使能 (DFSDM1 clock enable during Sleep mode)

此位由软件置 1 和清零

0：睡眠模式期间禁止 DFSDM1 时钟

1：睡眠模式期间使能 DFSDM1 时钟

位 23 保留，必须保持复位值。

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 保留，必须保持复位值。

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)

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)

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)

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 **EXTITLPEN**：睡眠模式期间的 EXTIT APB 和 SYSCTRL PFREE 时钟使能 (EXTIT APB and SYSCTRL PFREE clock enable during Sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 EXTIT APB 和 SYSCTRL PFREE 时钟

1：睡眠模式期间使能 EXTIT APB 和 SYSCTRL PFREE 时钟

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)

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)

由软件置 1 和复位。

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)

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 **SDIOLPEN**：睡眠模式期间的 SDIO 时钟使能 (SDIO clock enable during Sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 SDIO 模块时钟

1：睡眠模式期间使能 SDIO 模块时钟

位 10:9 保留，必须保持复位值。

位 8 **ADC1LPEN**：睡眠模式期间的 ADC1 时钟使能 (ADC1 clock enable during Sleep mode)

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 **UART10LPEN**：睡眠模式期间的 UART10 时钟使能 (UART10 clock enable during Sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 UART10 时钟

1：睡眠模式期间使能 UART10 时钟

位 6 **UART9LPEN**：睡眠模式期间的 UART9 时钟使能 (UART9 clock enable during Sleep mode)

由软件置 1 和清零。

0：睡眠模式期间禁止 UART9 时钟

1：睡眠模式期间使能 UART9 时钟

位 5 **USART6LPEN**：睡眠模式期间的 USART6 时钟使能 (USART6 clock enable during Sleep mode)

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)

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)

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)

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

6.3.23 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 ≤ 等待状态 ≤ 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*第 5.4.1 节：PWR 电源控制寄存器 (PWR_CR)* 中的 DBP 位置 1 才能修改这些位。有关详细信息，请参见*第 5.4.2 节：PWR 电源控制/状态寄存器 (PWR_CSR)*。只有备份域复位后，这些位才能复位（请参见*第 6.1.3 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.	Res.	LSEMOD	LSEBYP	LSERDY	LSEON
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)

由软件置 1 和清零。
0：复位未激活
1：复位整个备份域

位 15 **RTCEN**：RTC 时钟使能 (RTC clock enable)

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)

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 7:4 保留，必须保持复位值。

位 3 **LSEMOD**：外部低速振荡器模式 (External low-speed oscillator mode)

由软件置 1 和复位，用于选择低速振荡器的晶振模式。可使用两种功耗模式。

0：选择 LSE 振荡器“低功耗”模式

1：选择 LSE 振荡器“高驱动”模式

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，用于旁路调试模式下的振荡器。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)

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)

由软件置 1 和清零。

0：LSE 时钟关闭

1：LSE 时钟开启

6.3.24 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，其他通过系统复位进行复位。

访问：0 ≤ 等待状态 ≤ 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	BORRS TF	RMVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r	r	r	r	r	r	rt_w								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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，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)
发生来自 V_{DD} 域的独立看门狗复位时，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，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，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，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，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)
由软件置 1，用于将复位标志清零。
0：不起作用
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)

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)

由软件置 1 和清零。

0：LSI RC 振荡器关闭

1：LSI RC 振荡器开启

6.3.25 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[14:3]											
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
INCSTEP[2:0]			MODPER[11:0]												
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)
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)
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[14:0]**：增量步长 (Incrementation step)
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[11:0]**：调制周期 (Modulation period)
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。

6.3.26 RCC PLLI2S 配置寄存器 (RCC_PLLI2SCFGR)

RCC PLLI2S configuration register

偏移地址：0x84

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式配置 PLLI2S 时钟输出：

- $f_{(VCO\ 时钟)} = f_{(PLLI2S\ 时钟输入)} \times (PLLI2SN / PLLI2SM)$
- $f_{(USB\ OTG\ FS、SDIO、RNG\ 时钟输出)} = f_{(VCO\ 时钟)} / PLLQ$
- $f_{(DFSDM,\ I2S\ 时钟输出)} = f_{(VCO\ 时钟)} / PLLR$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	PLLI2SR[2:0]			PLLI2SQ[3:0]				Res.	PLLI2SSRC	Res.	Res.	Res.	Res.	Res.	Res.
	rw	rw	rw	rw	rw	rw	rw		rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PLLI2SN[8:0]									PLLI2SM[5:0]					
	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 30:28 **PLLI2SR[2:0]**：适用于 I2S 时钟的 PLLI2S 分频系数 (PLLI2S division factor for I2S clocks)

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 29.6.4 节：时钟发生器*。

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 USB OTG FS/SDIO/RNG 时钟的 PLLI2S 分频系数 (PLLI2S division factor for USB OTG FS/SDIO/RNG clock)

由软件置 1 和清零，用于控制 USB OTG FS/SDIO/RNG 时钟频率。这些位只能在 PLLI2S 已禁止时写入。

USB OTG FS/SDIO/RNG 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 保留，必须保持复位值。

位 22 **PLLI2SSRC**：PLLI2S 输入时钟源 (PLLI2S entry clock source)

由软件置 1 和清零，用于选择 PLLI2S 时钟源。该位只能在 PLLI2S 已禁止时写入。

0：HSE 或 HSI（取决于 PLLCFGR 的 PLLSRC）

1：选择外部 AFI 时钟 (CK_I2S_EXT) 作为 PLL 时钟输入

位 21:15 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)

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 100 MHz 和 432 MHz 之间。VCO 输入频率介于 1 MHz 到 2 MHz 之间（请参见*图 14* 和 *RCC PLL 配置寄存器 (RCC_PLLCFGR)* 的分频系数 M）

VCO 输出频率 = VCO 输入频率 × PLLI2SN，其中 50 ≤ PLLI2SN ≤ 432

000000000：PLLI2SN = 0，错误配置

000000001：PLLI2SN = 1，错误配置

...

001100010：PLLI2SN = 50

...

001100011：PLLI2SN = 99

001100100：PLLI2SN = 100

001100101：PLLI2SN = 101

001100110：PLLI2SN = 102

...

110110000：PLLI2SN = 432

110110000：PLLI2SN = 433，错误配置

...

111111111：PLLI2SN = 511，错误配置

注： *对于 1 MHz 以上的 VCO 输入频率，可选择的倍频系数介于 50 和 99 之间。但应注意必须满足上述指定的最小 VCO 输出频率。*

位 5:0 **PLLI2SM[5:0]**：适用于主 PLL (PLL) 和音频 PLL (PLLI2S) 输入时钟的分频系数 (Division factor for the main PLL (PLL) and audio PLL (PLLI2S) 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 MHz 到 2 MHz 之间。建议选择 2 MHz 的频率，以限制 PLL 抖动。

VCO 输入频率 = PLL 输入时钟频率/PLLI2SM，其中 2 ≤ PLLI2SM ≤ 63

000000：PLLI2SM = 0，错误配置

000001：PLLI2SM = 1，错误配置

000010：PLLI2SM = 2

000011：PLLI2SM = 3

000100：PLLI2SM = 4

.....

111110：PLLI2SM = 62

111111：PLLI2SM = 63

6.3.27 RCC 专用时钟配置寄存器 (RCC_DCKCFGR)

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。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CKDFSDM1SEL	Res.	Res.	I2S2RC[1:0]		I2S1RC[1:0]		TIMPRE	SAI1BSRC		SAI1ASRC		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
CKDFSDM1ASEL	CKDFSDM2ASEL	PLLDIVR					Res.	Res.	Res.	Res.	PLL2SDIVR				
rw	rw	rw									rw				

位 31 **CKDFSDMSEL**：DFSDM1 和 DFSDM2 内核时钟选择 (DFSDM1 & DFSDM2 Kernel clock selection)

0：将 APB2 时钟用作内核时钟
1：将系统时钟用作内核时钟

位 30:29 保留，必须保持复位值。

位 28:27 **I2S2SRC**：I2S APB2 时钟源选择 (I2S1/4/5) (I2S APB2 clocks source selection (I2S1/4/5))

由软件置 1 和复位。
这些位应在 PLL 和 PLLI2S 已禁止时写入。
00：I2S APB2 时钟频率 = $f_{(PLLI2S\_R)}$
01：I2S APB2 时钟频率 = 来自焊盘的外部 I2S 时钟 - 复用功能输入频率
10：I2S APB2 时钟频率 = $f_{(PLL\_R)}$
11：I2S APB2 时钟频率 = HSI/HSE（取决于 PLLSRC (PLLCFGR(22)))

位 26:25 **I2S1SRC**：I2S APB1 时钟源选择 (I2S2/3) (I2S APB1 clocks source selection (I2S2/3))

由软件置 1 和清零，用于控制 APB1 I2S 时钟的频率。
这些位应在 PLL 和 PLLI2S 已禁止时写入。
00：I2S APB1 时钟频率 = $f_{(PLLI2S\_R)}$
01：I2S APB1 时钟频率 = 来自焊盘的外部 I2S 时钟 - 复用功能输入频率
10：I2S APB1 时钟频率 = $f_{(PLL\_R)}$
11：I2S APB1 时钟频率 = HSI/HSE（取决于 PLLSRC (PLLCFGR(22)))

位 24 **TIMPRE**：定时器时钟预分频器选择 (Timers clocks prescalers selection)

由软件置 1 和复位，用于控制 APB1 或 APB2 域上所有定时器内核（ck_tim 和 ck_tgo）的时钟频率。
0：如果 PPREx 对应于 1 或 2 分频，则定时器内核时钟预分频比等于 HPRE；如果 PPREx 对应于 4 或以上分频，则定时器内核时钟预分频比等于 [(HPRE * PPREx) / 2]。
(Fck_tim = 2*Fck_pclk)。
1：如果 PPREx 对应于 1、2 或 4 分频，则定时器内核时钟预分频比等于 HPRE；如果 PPREx 对应于 8 或以上分频，则定时器内核时钟预分频比等于 [(HPRE * PPREx) / 4]。
(Fck_tim = 4 * Fck_pclk)。

位 23:22 **SAII1BSRC**：SAI1 B 时钟选择 (SAI1 B clock selection)
由软件置 1 和复位。
00：对 PLLI2S_R 分频 (R2) 作为 SAI1 B 时钟
01：I2S_CLIN 作为 SAI1 B 时钟
00：对 PLL_R 分频 (R1) 作为 SAI1 B 时钟
11：HS_CK 作为 SAI1 B 时钟

位 21:20 **SAII1ASRC**：SAI1 A 时钟选择 (SAI1 A clock selection)
由软件置 1 和复位。
00：对 PLLI2S_R 分频 (R2) 作为 SAI1 A 时钟
01：I2S_CLIN 作为 SAI1 A 时钟
00：对 PLL_R 分频 (R1) 作为 SAI1 A 时钟
11：HS_CK 作为 SAI1 A 时钟

位 19:16 保留，必须保持复位值。

位 15 **CKDFSDM1ASEL**：DFSDM1 音频时钟选择 (DFSDM1 audio clock selection)
0：选择 CK_I2S_APB1 作为音频时钟
1：选择 CK_I2S_APB2 作为音频时钟

位 14 **CKDFSDM2ASEL**：DFSDM2 音频时钟选择 (DFSDM2 audio clock selection)
0：选择 CK_I2S_APB1 作为音频时钟
1：选择 CK_I2S_APB2 作为音频时钟

位 13 保留，必须保持复位值。

位 12:8 **PLLDIVR**：适用于 SAI1 A/B 时钟的 PLL 分频系数 (PLL division factor for SAI1 A/B clock)
由软件置 1 和复位，用于控制 PLL_R1 时钟的分频系数。
这些位应在 PLL 已禁止时写入。
00000：PLL_R1 = 错误配置
00001：PLL_R1 = div/1
....
10000：PLL_R1 = div/16
....
11111：PLL_R1 = div/31

位 7:5 保留，必须保持复位值。

位 4:0 **PLLI2SDIVR**：适用于 SAI1 A/B 时钟的 PLLI2S 分频系数 (PLLI2S division factor for SAI1 A/B clock)
由软件置 1 和复位，用于控制 PLLI2S_R2 时钟的分频系数。
这些位应在 PLLI2S 已禁止时写入。
00000：PLLI2S_R2 = 错误配置
00001：PLLI2S_R2 = div/1
....
10000：PLLI2S_R2 = div/16
....
11111：PLLI2S_R2 = div/31

6.3.28 RCC 时钟门控使能寄存器 (CKGATENR)

RCC clocks gated enable register

偏移地址：0x90

复位值：0x0000 0000

访问：无等待周期，按字、半字和字节访问。

该寄存器允许使能或禁止指定 IP 的时钟门控。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EVTCL_CKEN	RCC_CKEN	FLITF_CKEN	SRAM_CKEN	SPARE_CKEN	CM4DBG_CKEN	AHB2APB2_CKEN	AHB2APB1_CKEN
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7 **EVTCL_CKEN**

0：使能时钟门控

1：禁止时钟门控，始终使能时钟

位 6 **RCC_CKEN**：RCC 时钟使能 (RCC clock enable)

0：使能时钟门控

1：禁止时钟门控，始终使能时钟。

位 5 **FLITF_CKEN**：Flash 接口时钟使能 (Flash Interface clock enable)

0：使能时钟门控

1：禁止时钟门控，始终使能时钟。

位 4 **SRAM_CKEN**：SRAM（SRAM1 和 SRAM2）控制器时钟使能 (SRAM (SRAM1 and SRAM2) controller clock enable)

0：使能时钟门控

1：禁止时钟门控，始终使能时钟。

位 3 **SPARE_CKEN**：备用时钟使能 (Spare clock enable)

0：使能时钟门控

1：禁止时钟门控，始终使能时钟。

位 2 **CM4DBG_CKEN**：Cortex M4 ETM 时钟使能 (Cortex M4 ETM clock enable)

0：使能时钟门控

1：禁止时钟门控，始终使能时钟。

位 1 **AHB2APB2_CKEN**：AHB-APB2 总线桥时钟使能 (AHB to APB2 Bridge clock enable)

0：使能时钟门控

1：禁止时钟门控，始终使能时钟。

位 0 **AHB2APB1_CKEN**：AHB-APB1 总线桥时钟使能 (AHB to APB1 Bridge clock enable)

0：使能时钟门控

1：禁止时钟门控，始终使能时钟。

6.3.29 RCC 专用时钟配置寄存器 (RCC_DCKCFGR2)

RCC Dedicated Clocks Configuration Register

偏移地址：0x94

复位值：0x0000 0000

该寄存器允许使能或禁止指定 IP 的时钟门控。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIMER1 SEL		Res.	SDIO SEL	CK48M SEL	Res.			I2CFMP1 SEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.
rw			rw	rw				rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

位 31:30 **LPTIMER1SEL**：LPTIMER1 内核时钟源选择 (LPTIMER1 kernel clock source selection)
00：选择 APB 时钟作为 LPTIMER1 时钟
01：选择 HSI 时钟作为 LPTIMER1 时钟
10：选择 LSI 时钟作为 LPTIMER1 时钟
11：选择 LSE 时钟作为 LPTIMER1 时钟

位 29 保留，必须保持复位值。

位 28 **CKSDIOSEL**：SDIO 时钟选择 (SDIO clock selection)
0：CK_48MHz（请参见 CK48MSEL 位定义）
1：时钟系统

位 27 **CK48MSEL**：SDIO/USBFS 时钟选择 (SDIO/USBFS clock selection)
0：$f_{(PLL\_Q)}$
1：$f_{(PLLI2S\_Q)}$

位 26:24 保留，必须保持复位值。

位 23:22 **I2CFMP1SEL[1:0]**：I2CFMP1 内核时钟源选择 (I2CFMP1 kernel clock source selection)
00：选择 APB 时钟作为 I2CFMP1 时钟
01：选择系统时钟作为 I2CFMP1 时钟
10：选择 HSI 时钟作为 I2CFMP1 时钟
11：选择 APB 时钟作为 I2CFMP1（与“00”一样）

位 21:0 保留，必须保持复位值。

6.3.30 RCC 寄存器映射

表 24 给出了寄存器映射和复位值。

表 24. STM32F413/423 的 RCC 寄存器映射和复位值

地址偏移	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	RCC_CR	Res.	Res.	Res.	Res.	PLL I2SRDY	PLL I2SON	PLL RDY	PLL ON	Res.	Res.	Res.	Res.	CSSON	HSEBYP	HSERDY	HSEON	HSICAL[7:0]								HSITRIM[4:0]					Res.	HSIRDY	HSION
0x04	RCC_PLLCFGR	Res.	PLLR[2:0]			PLLQ[3:0]				Res.	PLLSRC	Res.	Res.	Res.	Res.	PLLP[1:0]		Res.	PLLN[8:0]								PLLM[5:0]						
0x08	RCC_CFGR	MCO2[1:0]		MCO2PRE[2:0]			MCO1PRE[2:0]			Res.	MCO1[1:0]		RTCPRE[4:0]					PPRE2[2:0]			PPRE1[2:0]			Res.	Res.	HPRE[3:0]				SWS[1:0]		SW[1:0]	
0x0C	RCC_CIR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSSC	Res.	PLLI2SRDYC	PLLRDYC	HSERDYC	HSIRDYC	LSERDYC	LSIRDYC	Res.	Res.	PLLI2SRDYIE	PLLRDYIE	HSERDYIE	HSIRDYIE	LSERDYIE	LSIRDYIE	CSSF	Res.	PLLI2SRDYF	PLLRDYF	HSERDYF	HSIRDYF	LSERDYF	LSIRDYF
0x10	RCC_AHB1RSTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA2RST	DMA1RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRCRST	Res.	Res.	Res.	Res.	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	Res.	CRYPRST[1]	Res.	Res.	Res.	Res.
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	FSMCRST
0x1C	Reserved																																
0x20	RCC_APB1RSTR	UART8RST	UART7RST	DACRST	PWRRST	CAN3RST	CAN2RST	CAN1RST	I2CFMP1RST	I2C3RST	I2C2RST	I2C1RST	UART5RST	UART4RST	USART3RST	USART2RST	Res.	SPI3RST	SPI2RST	Res.	Res.	WWDGRST	Res.	LPTIMER1RST	TIM14RST	TIM13RST	TIM12RST	TIM7RST	TIM6RST	TIM5RST	TIM4RST	TIM3RST	TIM2RST
0x24	RCC_APB2RSTR	Res.	Res.	Res.	Res.	Res.	Res.	DFSDM2RST	DFSDM1RST	Res.	SAI1RST	Res.	SPI5RST	Res.	TIM11RST	TIM10RST	TIM9RST	Res.	SYSCFGRST	SP45RST	SPI1RST	SDIORST	Res.	Res.	ADCRST	UART10RST	UART9RST	USART6RST	USART1RST	Res.	Res.	TIM8RST	TIM1RST
0x28	Reserved																																
0x2C	Reserved																																
0x30	RCC_AHB1ENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA2EN	DMA1EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CRCEN	Res.	Res.	Res.	Res.	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	Res.	CRYPEN[1]	Res.	Res.	Res.	Res.

表 24. STM32F413/423 的 RCC 寄存器映射和复位值（续）

地址偏移	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
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	FSMCEN
0x3C	Reserved																																
0x40	RCC_APB1ENR	UART8EN	UART7EN	DACEN	PWREN	CAN3EN	CAN2EN	CAN1EN	I2CFMP1EN	I2C3EN	I2C2EN	I2C1EN	UART5EN	UART4EN	USART3EN	USART2EN	Res.	SPI3EN	SPI2EN	Res.	Res.	WWDGEN	RTCAPBEN	LPTIMER1EN	TIM14EN	TIM13EN	TIM12EN	TIM7EN	TIM6EN	TIM5EN	TIM4EN	TIM3EN	TIM2EN
0x44	RCC_APB2ENR	Res.	Res.	Res.	Res.	Res.	Res.	DFSDM2EN	DFSDM1EN	Res.	SAI1EN	Res.	SPI5EN	Res.	TIM11EN	TIM10EN	TIM9EN	Res.	SYSCFGEN	SPI4EN	SPI1EN	SDIOEN	Res.	Res.	ADC1EN	UART10EN	UART9EN	USART6EN	USART1EN	Res.	Res.	TIM8EN	TIM1EN
0x48	Reserved																																
0x4C	Reserved																																
0x50	RCC_AHB1LPENR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA2LPEN	DMA1LPEN	Res.	Res.	Res.	SRAM2LPEN	SRAM1LPEN	FLITFLPEN	Res.	Res.	CRCLPEN	Res.	Res.	Res.	Res.	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	Res.	CRYPLPEN(1)	Res.	Res.	Res.	Res.
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	FSMCLPEN
0x5C	Reserved																																
0x60	RCC_APB1LPENR	UART8LPEN	UART7LPEN	DACLPEN	PWRLPEN	CAN3LPEN	CAN2LPEN	CAN1LPEN	I2CFMP1LPEN	I2C3LPEN	I2C2LPEN	I2C1LPEN	UART5LPEN	UART4LPEN	USART3LPEN	USART2LPEN	Res.	SPI3LPEN	SPI2LPEN	Res.	Res.	WWDGLPEN	Res.	LPTIMER1EN	TIM14LPEN	TIM13LPEN	TIM12LPEN	TIM7LPEN	TIM6LPEN	TIM5LPEN	TIM4LPEN	TIM3LPEN	TIM2LPEN
0x64	RCC_APB2LPENR	Res.	Res.	Res.	Res.	Res.	Res.	DFSDM2LPEN	DFSDM1LPEN	Res.	SAI1LPEN	Res.	SPI5LPEN	Res.	TIM11LPEN	TIM10LPEN	TIM9LPEN	EXTITEN	SYSCFGLPEN	SPI4LPEN	SPI1LPEN	SDIOLPEN	Res.	Res.	ADC1LPEN	UART10LPEN	UART9LPEN	USART6LPEN	USART1LPEN	Res.	Res.	TIM8LPEN	TIM1LPEN
0x68	Reserved																																
0x6C	Reserved																																
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[1:0]		Res.	Res.	Res.	Res.	LSEMOD	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	Reserved																																
0x7C	Reserved																																

表 24. STM32F413/423 的 RCC 寄存器映射和复位值（续）

地址偏移	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
0x80	RCC_SSCGR	SSCGEN	SPREADSEL	Res.	Res.	INCSTEP[14:0]															MODPER[11:0]												
0x84	RCC_PLLI2SCFGR	Res.	PLLI2SR[2:0]			PLLI2SQ[3:0]				Res.	PLLI2SSRC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLLI2SN[8:0]								PLLI2SM[5:0]					
0x88	Reserved																																
0x8C	RCC_DCKCFGR	CKDFSDM1SEL	Res.	Res.	I2S2SRC[1:0]		I2S1SRC[1:0]		TIMPRE	SAI1BSR[1:0]		SAI1ASR[1:0]		Res.	Res.	Res.	Res.	CKDFSDM1ASEL	CKDFSDM2ASEL	Res.	PLLDIVR[4:0]					Res.	Res.	Res.	PLLI2SDIVR[4:0]				
0x90	CKGATENR																									EVTCL_CKEN	RCC_CKEN	FLITF_CKEN	SRAM12_CKGA_BPEN	SPARE_CKEN	CM4DBG_CKEN	AHB2APB2_CKEN	AHB2APB1_CKEN
0x94	RCC_DCKCFGR2	LPTIMER1SEL1	LPTIMER1SEL0	Res.	SDIOSEL	CK48MSEL	Res.	Res.	Res.	I2CFMP1SEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

1. 仅在 STM32F423xx 上可用。

有关寄存器边界地址的信息，请参见*表 1*。

7 通用 I/O (GPIO)

7.1 GPIO 简介

每个通用 I/O 端口包括 4 个 32 位配置寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR 和 GPIOx_PUPDR）、2 个 32 位数据寄存器（GPIOx_IDR 和 GPIOx_ODR）、1 个 32 位置 1/复位寄存器 (GPIOx_BSRR)、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）。

7.2 GPIO 主要特性

- 受控 I/O 多达 16 个
- 输出状态：推挽或开漏 + 上拉/下拉
- 从输出数据寄存器 (GPIOx_ODR) 或外设（复用功能输出）输出数据
- 可为每个 I/O 选择不同的速度
- 输入状态：浮空、上拉/下拉、模拟
- 将数据输入到输入数据寄存器 (GPIOx_IDR) 或外设（复用功能输入）
- 置 1 和复位寄存器 (GPIOx_BSRR)，对 GPIOx_ODR 具有按位写权限
- 锁定机制 (GPIOx_LCKR)，可冻结 I/O 配置
- 模拟功能
- 复用功能输入/输出选择寄存器（一个 I/O 最多可具有 16 个复用功能）
- 快速翻转，每次翻转最快只需要两个时钟周期
- 引脚复用非常灵活，允许将 I/O 引脚用作 GPIO 或多种外设功能中的一种

7.3 GPIO 功能描述

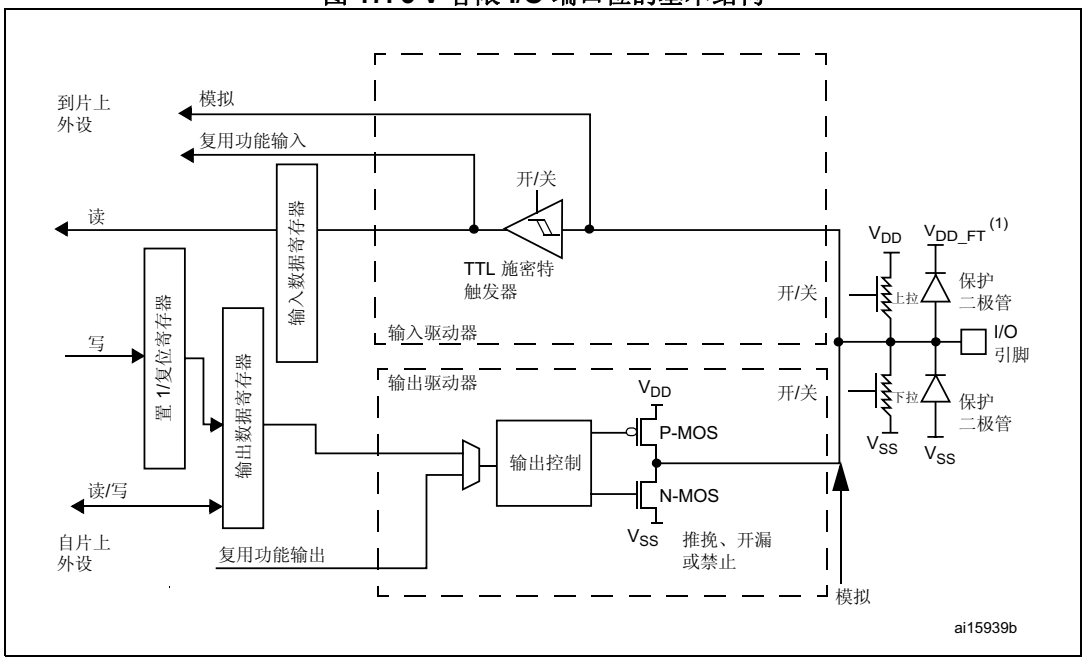
根据数据手册中列出的每个 I/O 端口的特性，可通过软件将通用 I/O (GPIO) 端口的各个端口位分别配置为多种模式：

- 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 寄存器旨在实现对 GPIO 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 17 显示了 5 V 容限 I/O 端口位的基本结构。*表 25* 给出了可能的端口位配置方案。

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 25. 端口位配置表(1)

MODER(i) [1:0]	OTYPER(i)	OSPEEDR(i) [B:A]	PUPDR(i) [1:0]		I/O 配置	
01	0	SPEED [B:A]	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 [B:A]	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	

表 25. 端口位配置表[1]（续）

MODER(i) [1:0]	OTYPER(i)	OSPEEDR(i) [B:A]		PUPDR(i) [1:0]		I/O 配置	
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。

7.3.1 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，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（输出 0 时仅激活 N-MOS）。

输入数据寄存器 (GPIOx_IDR) 每隔 1 个 AHB1 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。

7.3.2 I/O 引脚复用器和映射

微控制器 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 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：

- 完成复位后，所有 I/O 都会连接到系统的复用功能 0 (AF0)。
- 外设的复用功能映射到 AF1 至 AF13。
- 带 FPU 的 Cortex®-M4 EVENTOUT 映射到 AF15。

下面的*图 18：在 STM32F413/423 上选择复用功能*对此结构进行了说明。

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，请按照以下步骤操作：

- **系统功能**

  将 I/O 连接到 AF0，然后根据所用功能进行配置：

  – JTAG/SWD：在各器件复位后，会将这些引脚指定为专用引脚，可供片上调试模块立即使用（不受 GPIO 控制器控制）

  – RTC_REFIN：此引脚应配置为输入浮空模式。

  – MCO1 和 MCO2：这些引脚必须配置为复用功能模式。

注： *可禁止部分或全部 JTAG/SWD 引脚，以释放相关联的引脚供 GPIO 使用。*

有关详细信息，请参见*第 6.2.10 节：时钟输出功能*。

表 26. 灵活的 SWJ-DP 引脚分配

可用的调试端口	用到的 SWJ I/O 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				

- **GPIO**

  在 GPIOx_MODER 寄存器中将所需 I/O 配置为输出或输入。

- **外设复用功能**

  对于 ADC，在 GPIOx_MODER 寄存器中将所需 I/O 配置为模拟通道。

  对于其他外设：

  – 在 GPIOx_MODER 寄存器中将所需 I/O 配置为复用功能

  – 通过 GPIOx_OTYPER、GPIOx_PUPDR 和 GPIOx_OSPEEDR 寄存器，分别选择类型、上拉/下拉以及输出速度

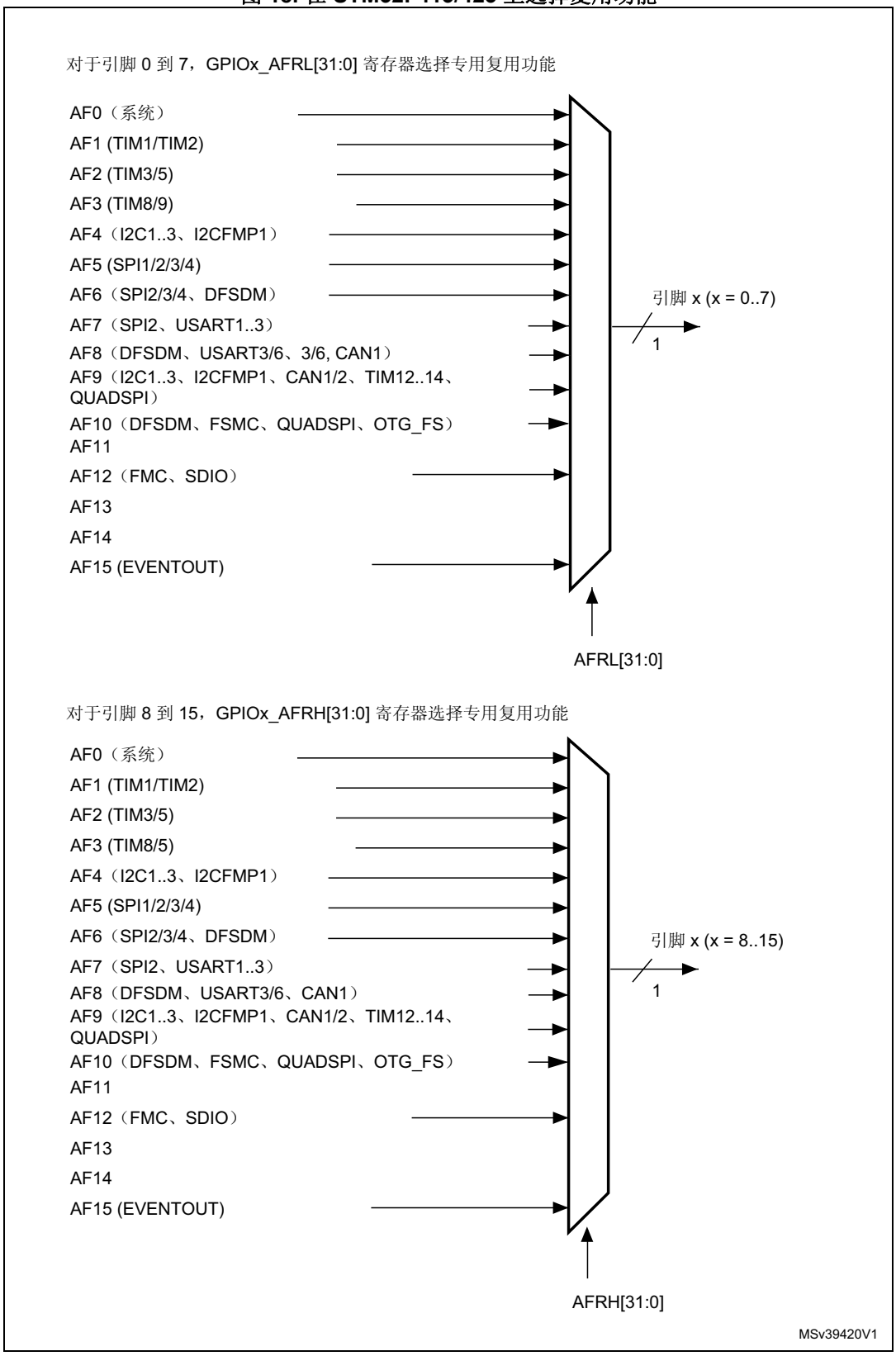
  – 在 GPIOx_AFRL 或 GPIOx_AFRH 寄存器中，将 I/O 连接到所需 AFx

- **EVENTOUT**

  配置用于输出带 FPU 的 Cortex®-M4 EVENTOUT 信号的 I/O 引脚（通过将其连接到 AF15）

注： *有关系统和外设的复用功能 I/O 引脚映射的详细信息，请参见数据手册中的“复用功能映射”表。*

图 18. 在 STM32F413/423 上选择复用功能



7.3.3 I/O 端口控制寄存器

每个 GPIO 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 方向（输入、输出、AF、模拟）。GPIOx_OTYPER 和 GPIOx_OSPEEDR 寄存器分别用于选择输出类型（推挽或开漏）和速度 (无论采用哪种 I/O 方向，都会直接将 I/O 速度引脚连接到相应的 GPIOx_OSPEEDR 寄存器位）。无论采用哪种 I/O 方向，GPIOx_PUPDR 寄存器都用于选择上拉/下拉。

7.3.4 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 7.4.5 节：GPIO 端口输入数据寄存器 (GPIOx_IDR) (x = A...H)*和*第 7.4.6 节：GPIO 端口输出数据寄存器 (GPIOx_ODR) (x = A...H)*。

7.3.5 I/O 数据位操作

置 1/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 1 和复位操作。置 1/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：BSRR(i) 和 BSRR(i+SIZE)。当写入 1 时，BSRR(i) 位会置 1 对应的 ODR(i) 位。当写入 1 时，BSRR(i+SIZE) 位会清零 ODR(i) 对应的位。

在 GPIOx_BSRR 中向任何位写入 0 都不会对 GPIOx_ODR 中的对应位产生任何影响。如果在 GPIOx_BSRR 中同时尝试对某个位执行置 1 和复位操作，则置 1 操作优先。

使用 GPIOx_BSRR 寄存器更改 GPIOx_ODR 中各个位的值是一个“单次”操作，不会锁定 GPIOx_ODR 位。随时都可以直接访问 GPIOx_ODR 位。GPIOx_BSRR 寄存器提供了一种执行原子按位处理的方法。

在对 GPIOx_ODR 进行位操作时，软件无需禁止中断：在一次原子 AHB1 写访问中，可以修改一个或多个位。

7.3.6 GPIO 锁定机制

通过将特定的写序列应用到 GPIOx_LCKR 寄存器，可以冻结 GPIO 控制寄存器。冻结的寄存器包括 GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR、GPIOx_AFRL 和 GPIOx_AFRH。

要对 GPIOx_LCKR 寄存器执行写操作，必须应用特定的写/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 7.4.8 节：GPIO 端口配置锁定寄存器 (GPIOx_LCKR) (x = A...H)*）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 1。

有关详细信息，请参见*第 7.4.8 节：GPIO 端口配置锁定寄存器 (GPIOx_LCKR) (x = A...H)* 中的 LCKR 寄存器说明。

7.3.7 I/O 复用功能输入/输出

有两个寄存器可用来从每个 I/O 可用的十六个复用功能输入/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他某个引脚。
这意味着可使用 GPIOx_AFRL 和 GPIOx_AFRH 复用功能寄存器在每个 GPIO 上复用多个可用的外设功能。这样一来，应用程序可为每个 I/O 选择任何一个可用功能。由于 AF 选择信号由复用功能输入和复用功能输出共用，所以只需为每个 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，请参见数据手册。

注： *对于每个 I/O 而言，应用程序一次只能为其选择一个可用的外设功能。*

7.3.8 外部中断线/唤醒线

所有端口都具有外部中断功能。要使用外部中断线，必须将端口配置为输入模式，请参见 *第 10.2 节：外部中断/事件控制器 (EXTI)* 和 *第 10.2.3 节：唤醒事件管理*。

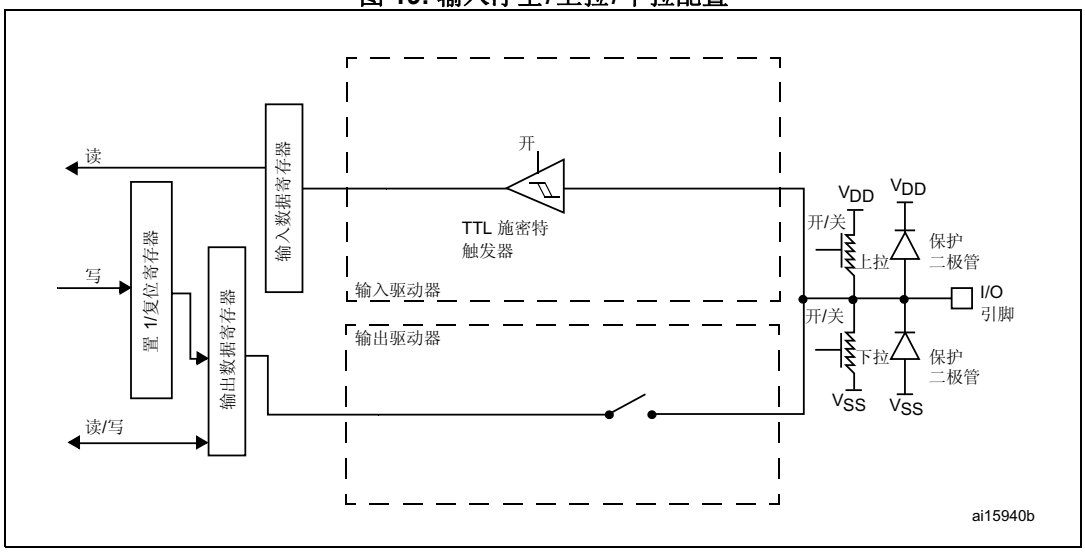
7.3.9 输入配置

对 I/O 端口进行编程作为输入时：

- 输出缓冲器被关闭
- 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 AHB1 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 19 说明了 I/O 端口位的输入配置。

图 19. 输入浮空/上拉/下拉配置



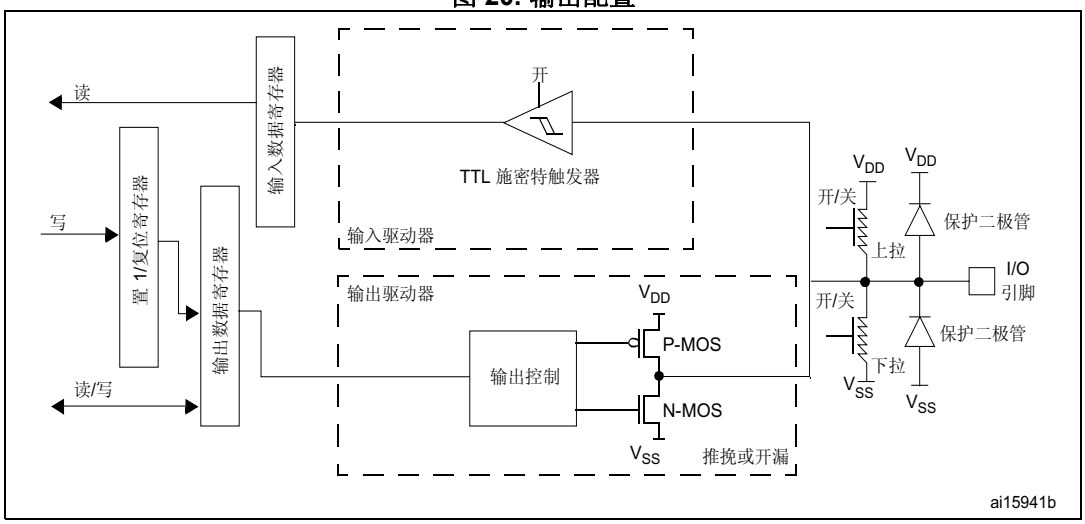
7.3.10 输出配置

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弱上拉电阻和下拉电阻
- 输入数据寄存器每隔 1 个 AHB1 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 20 说明了 I/O 端口位的输出配置。

图 20. 输出配置



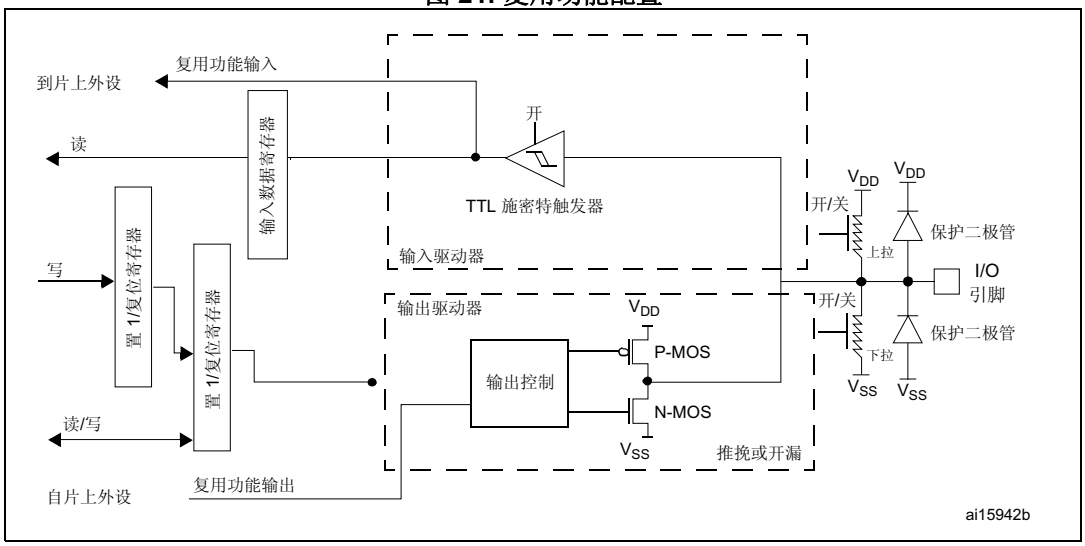
7.3.11 复用功能配置

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
- 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 AHB1 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 21 说明了 I/O 端口位的复用功能配置。

图 21. 复用功能配置



7.3.12 模拟配置

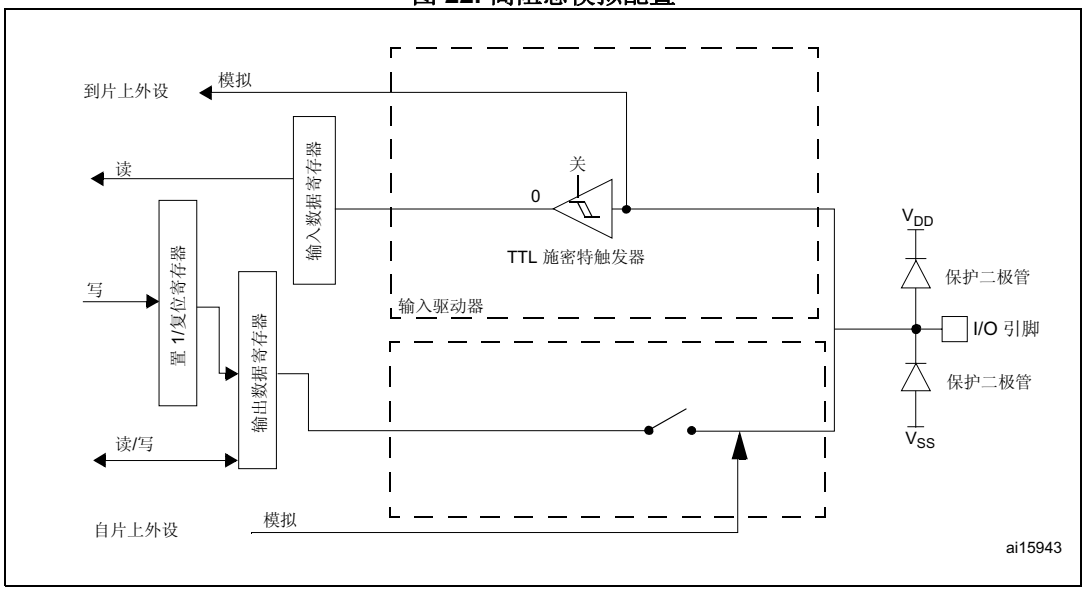
对 I/O 端口进行编程作为模拟配置时：

- 输出缓冲器被关闭
- 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关闭
- 对输入数据寄存器的读访问值为“0”

注： *在模拟配置中，I/O 引脚不能为 5 V 容限。*

图 22 说明了 I/O 端口位的高阻态模拟输入配置。

图 22. 高阻态模拟配置



7.3.13 将 OSC32_IN/OSC32_OUT 引脚用作 GPIO PC14/PC15 端口引脚

当 LSE 振荡器处于关闭状态时，可分别将 LSE 振荡器引脚 OSC32_IN 和 OSC32_OUT 用作通用 PC14 I/O 和 PC15 I/O。当 LSE 振荡器处于开启状态时，PC14 和 PC15 I/O 只能被配置为 LSE 振荡器引脚 OSC32_IN 和 OSC32_OUT。可通过将 RCC_BDCR 寄存器中的 LSEON 位置 1 来完成此操作。LSE 的优先级高于 GPIO 功能。

注： *当 1.2 V 域掉电时（因器件进入待机模式），或者当备份域由 V_{BAT} 供电时（V_{DD} 不再供电），PC14/PC15 GPIO 功能将丢失。在此情况下，I/O 被设置为模拟输入模式。*

7.3.14 将 OSC_IN/OSC_OUT 引脚用作 GPIO PH0/PH1 端口引脚

当 HSE 振荡器处于关闭状态时，可分别将 HSE 振荡器引脚 OSC_IN 和 OSC_OUT 用作通用 PH0 I/O 和 PH1 I/O。（完成复位后，HSE 振荡器处于关闭状态）。当 HSE 振荡器处于开启状态时，PH0/PH1 I/O 只能被配置为 HSE 振荡器引脚 OSC_IN/OSC_OUT。可通过将 RCC_CR 寄存器中的 HSEON 位置 1 来完成此操作。HSE 的优先级高于 GPIO 功能。

7.3.15 选择 RTC 附加功能

STM32F4xx 具有一个 GPIO 引脚 RTC_AF1，可用于检测入侵或时间戳事件、RTC_ALARM 或 RTC_CALIB RTC 输出。

- RTC_AF1 (PC13) 可用于以下目的：

RTC_ALARM 输出：此输出可以是 RTC 闹钟 A、RTC 闹钟 B 或 RTC 唤醒，具体取决于 RTC_CR 寄存器中的 OSEL[1:0] 位

- RTC_CALIB 输出：可通过将 RTC_CR 寄存器中的 COE[23] 置 1 来使能此功能
- RTC_TAMP1：入侵事件检测
- RTC_TS：时间戳事件检测

可如下所示通过 RTC_TAFCR 寄存器来选择相应的引脚：

- TAMP1INSEL 用于选择要用作 RTC_TAMP1 入侵输入的引脚
- TSINSEL 用于选择要用作 RTC_TS 时间戳输入的引脚
- ALARMOUTTYPE 用于选择是以推挽模式还是开漏模式输出 RTC_ALARM

输出机制遵循*表 27* 中所列的优先级顺序。

表 27. RTC 附加功能(1)

引脚配置和功能	使能	使能	入侵使能	时间戳使能	TAMP1INSEL TAMPER1 引脚选择	TSINSEL TIMESTAMP 引脚选择	ALARMOUTTYP 配置
闹钟输出开漏	1	无关	无关	无关	无关	无关	0
闹钟输出推挽	1	无关	无关	无关	无关	无关	1
校准输出推挽	0	1	无关	无关	无关	无关	无关
TAMPER1 输入浮空	0	0	1	0	0	无关	无关
TIMESTAMP 和 TAMPER1 输入浮空	0	0	1	1	0	0	无关
TIMESTAMP 输入浮空	0	0	0	1	无关	0	无关
标准 GPIO	0	0	0	0	无关	无关	无关

1. OD：开漏；PP：推挽。

7.4 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 28*。

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

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

GPIO port mode register

偏移地址：0x00

复位值：

- 0xA800 0000（端口 A）
- 0x0000 0280（端口 B）
- 0x0000 0000（其他端口）

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:2y+1 **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 方向模式。

00：输入（复位状态）

01：通用输出模式

10：复用功能模式

11：模拟模式

7.4.2 GPIO 端口输出类型寄存器 (GPIOx_OTYPER) (x = A...H)

GPIO port output type register

偏移地址：0x04

复位值：0x0000 0000

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

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

位 31:16 保留，必须保持复位值。

位 15:0 **OTy**：端口 x 配置位 (Port x configuration bits) (y = 0..15)

这些位通过软件写入，用于配置 I/O 端口的输出类型。

0：推挽输出（复位状态）

1：开漏输出

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

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（其他端口）

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:2y+1 **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。

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

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（其他端口）

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:2y+1 **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

7.4.5 GPIO 端口输入数据寄存器 (GPIOx_IDR) (x = A...H)

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)

这些位为只读形式，只能在字模式下访问。它们包含相应 I/O 端口的输入值。

7.4.6 GPIO 端口输出数据寄存器 (GPIOx_ODR) (x = A...H)

GPIO port output data register

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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： 对于原子位置 1/ 复位，通过写入 GPIOx_BSRR 寄存器，可分别对 ODR 位进行置 1 和复位 (x = A...H)。

7.4.7 GPIO 端口置 1/复位寄存器 (GPIOx_BSRR) (x = A...H)

GPIO port bit set/reset register

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位 31:16 **BRy**：端口 x 复位位 y (Port x reset bit y) (y = 0..15)

这些位为只写形式，只能在字、半字或字节模式下访问。若读取这些位则返回 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 1，则 BSx 的优先级更高。

位 15:0 **BSy**：端口 x 置 1 位 y (Port x set bit y) (y= 0..15)

这些位为只写形式，只能在字、半字或字节模式下访问。若读取这些位则返回 0x0000。

0：不会对相应的 ODRx 位执行任何操作

1：将相应的 ODRx 位置 1

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

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。在此写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

访问：仅 32 位字，读/写寄存器

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.	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[16]**：锁定键 (Lock key)

可随时读取此位。可使用锁定键写序列对其进行修改。

0：端口配置锁定键未激活

1：端口配置锁定键已激活。直到 MCU 复位或外设复位时，才锁定 GPIOx_LCKR 寄存器。

锁定键写序列：

WR LCKR[16] = ‘1’ + LCKR[15:0]

WR LCKR[16] = ‘0’ + LCKR[15:0]

WR LCKR[16] = ‘1’ + LCKR[15:0]

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 CPU 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

7.4.9 GPIO 复用功能低位寄存器 (GPIOx_AFRL) (x = A...H)

GPIO alternate function low register

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFRL7[3:0]				AFRL6[3:0]				AFRL5[3:0]				AFRL4[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
AFRL3[3:0]				AFRL2[3:0]				AFRL1[3:0]				AFRL0[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **AFRLy**：端口 x 位 y 的复用功能选择 (Alternate function selection for port x bit y) (y = 0..7)

这些位通过软件写入，用于配置复用功能 I/O。

AFRLy 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

7.4.10 GPIO 复用功能高位寄存器 (GPIOx_AFRH) (x = A...H)

GPIO alternate function high register

偏移地址：0x24

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFRH15[3:0]				AFRH14[3:0]				AFRH13[3:0]				AFRH12[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
AFRH11[3:0]				AFRH10[3:0]				AFRH9[3:0]				AFRH8[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **AFRHy**：端口 x 位 y 的复用功能选择 (Alternate function selection for port x bit y) (y = 8..15)

这些位通过软件写入，用于配置复用功能 I/O。

AFRHy 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

7.4.11 GPIO 寄存器映射

下表列出了 GPIO 寄存器映射和复位值。

表 28. 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	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
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...H)	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

表 28. 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
0x04	**GPIOx_ OTYPER** (where x = A...H)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	**GPIOx_ OSPEEDR** (where x = C...H)	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
0x08	**GPIOA_ OSPEEDER**	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
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
0x0C	**GPIOx_PUPDR** (where x = C...H)	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	0	0	0	0	0	0	0	0	0
0x10	**GPIOx_IDR** (where x = A...H)	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...H)	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...H)	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0	BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 28. 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
0x1C	**GPIOx_LCKR** (where x = A...H)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LCKK	LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	**GPIOx_AFRL** (where x = A...H)	AFRL7[3:0]				AFRL6[3:0]				AFRL5[3:0]				AFRL4[3:0]				AFRL3[3:0]				AFRL2[3:0]				AFRL1[3:0]				AFRL0[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...H)	AFRH15[3:0]				AFRH14[3:0]				AFRH13[3:0]				AFRH12[3:0]				AFRH11[3:0]				AFRH10[3:0]				AFRH9[3:0]				AFRH8[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*存储器映射部分*。

8 系统配置控制器 (SYSCFG)

系统配置控制器主要用于管理对可执行代码存储区域的地址重映射，以及管理 GPIO 的外部中断线连接。

8.1 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。

8.2 SYSCFG 寄存器

8.2.1 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重映射进行配置：

- 使用两个位来配置可在地址 0x0000 0000 访问的存储器区域， 从而通过软件选择物理重映射，而旁路 BOOT 引脚。
- 这两个位的复位值和复位时 BOOT 引脚的设置相同。当 BOOT0 引脚设为 0 从主 Flash 中自举时，此寄存器值为 0x00。

在重映射模式下，CPU 可以通过 ICode 总线（而不是 System 总线）访问外部存储器来提高性能。

偏移地址：0x00

复位值：0x0000 000X（X 为 BOOT 引脚选择的存储器模式）

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.	Res.	Res.	Res.	Res.	Res.
15	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.	MEM_MODE	
														rw	rw

位 31:2 保留，必须保持复位值。

位 1:0 **MEM_MODE**：存储器映射选择 (Memory mapping selection）

由软件置 1 和清零。此位控制哪块存储区域被映射到地址 0x0000 0000。这两个位的复位值和复位时 Boot 引脚的设置相同。

00：将主 Flash 映射到地址 0x0000 0000
01：将系统 Flash 映射到地址 0x0000 0000
10：保留
11：将 SRAM 映射到地址 0x0000 0000。

注： 有关在地址 0x0000 0000 的存储器映射的详细信息，请参见图 2：存储器映射。

8.2.2 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADC1DC2
															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 31:17 保留，必须保持复位值。

位 16 **ADC1DC2**：

0：不起作用。

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 ADC1DC2 位。

- 当 PWR_CR 寄存器中的 ADCDC1 位置 1 时，这些位不能置 1。

位 15:0 保留，必须保持复位值。

8.2.3 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（为 EXTI3 和 EXTI2 配置保留）
其他配置：保留

8.2.4 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 = 4 to 7))

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
其他配置：保留

8.2.5 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 = 8 to 11))
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
其他配置：保留

8.2.6 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 = 12 to 15))
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

8.2.7 SYSCFG 配置寄存器 2 (SYSCFG_CFGR2)

SYSCFG configuration register 2

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.	Res.	Res.	Res.	Res.	Res.	PVDL	Res.	CLL
													rw		rw

位 31:3 保留，必须保持复位值。

位 2 **PVDL**：PVD 锁定 (PVD lock)

该位由软件置 1， 并且只能通过系统复位的方式清零。它可以启用和锁定 PVD 与 TIM1/8 中断输入的连接， 也可以锁定（写保护）PWR_CR 寄存器的 PVDE 和 PVDS[2:0] 位。

0：PVD 中断未连接到 TIM1/8 中断输入。可以读取和修改 PVDE 和 PVDS[2:0]

1：PVD 中断连接到 TIM1/8 中断输入。PVDE 和 PVDS[2:0] 是只读的

位 1 保留，必须保持复位值。

位 0 **CLL**：内核 LOCKUP 锁定 (core LOCKUP lock)

该位由软件置 1 和清零。它可以启用和锁定带 FPU 的 Cortex®-M4 内核的 LOCKUP（硬故障）输出与 TIM1/8 中断输入的连接。

0：带 FPU 的 Cortex®-M4 LOCKUP 输出未连接到 TIM1/8 中断输入。

1：带 FPU 的 Cortex®-M4 LOCKUP 输出连接到 TIM1/8 中断输入。

8.2.8 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 7:1 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）

8.2.9 SYSCFG 配置寄存器 (SYSCFG_CFGR)

SYSCFG configuration 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	I2CFMP1_SDA	I2CFMP1_SCL
														rw	rw

位 31:2 保留，必须保持复位值。

位 1 **I2CFMP1_SDA**

由软件置 1 和清零。此位置 1 时，可强制在 I2CFMP1_SDA 引脚（通过 GPIO 端口模式寄存器和 GPIO 复用功能选择位选择）上使能 FM+ 驱动能力。

位 0 **I2CFMP1_SCL**

由软件置 1 和清零。此位置 1 时，可强制在 I2CFMP1_SCL 引脚（通过 GPIO 端口模式寄存器和 GPIO 复用功能选择位选择）上使能 FM+ 驱动能力。

8.2.10 DFSDM 多通道延迟控制寄存器 (SYSCFG_MCHDLYCR)

DFSDM Multi-channel delay control 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.	DFSDM2_CKOSEL	DFSDM2_CFG	DFSDM2_CK37SEL
r	r	r	r	r	r	r	r	r	r	r	r	r	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DFSDM2_CK26SEL	DFSDM2_CK15SEL	DFSDM2_CK04SEL	DFSDM2_D6SEL	DFSDM2_D4SEL	DFSDM2_D2SEL	DFSDM2_D0SEL	MCHDLYEN2	DFSDM1_CKOSEL	DFSDM1_CFG	DFSDM1_CK13SEL	DFSDM1_CK02SEL	DFSDM1_D2SEL	DFSDM1_D0SEL	MCHDLYEN1	BSCKSEL
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 **DFSDM2_CKOSEL**：DFSDM2_CKOUT 的源选择 (Source selection for DFSDM2_CKOUT)（M2 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：DFSDM2_CKOUT 的源是由 DFSDM2 生成的 **CkOut**
1：DFSDM2_CKOUT 的源是 M27 的输出

位 17 **DFSDM2_CFG**：DFSDM2 的 **CkIn** 源选择 (CkIn source selection for DFSDM2)（M9、M10、M11、M12、M13、M14、M15、M16 和 M2 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：**CkIn[7:0]** 信号源由引脚 DFSDM2_CKINy 提供 (M[16:9] = 0)
1：**CkIn[7:0]** 信号源由 DM[6:3] 的输出提供 (M[16:9] = 1)

位 16 **DFSDM2_CK37SEL**：由 TIM3 OC1 门控的 DFSDM2 比特流时钟分配 (Distribution of the DFSDM2 bitstream clock gated by TIM3 OC1)（DM3 多路解复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：门控时钟分配到 **CkIn3** (DM3 = 0)
1：门控时钟分配到 **CkIn7** (DM3 = 1)

位 15 **DFSDM2_CK26SEL**：由 TIM3 OC2 门控的 DFSDM2 比特流时钟分配 (Distribution of the DFSDM2 bitstream clock gated by TIM3 OC2)（DM4 多路解复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：门控时钟分配到 **CkIn2** (DM4 = 0)
1：门控时钟分配到 **CkIn6** (DM4 = 1)

位 14 **DFSDM2_CK15SEL**：由 TIM3 OC3 门控的 DFSDM2 比特流时钟分配 (Distribution of the DFSDM2 bitstream clock gated by TIM3 OC3)（DM5 多路解复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：门控时钟分配到 **CkIn1** (DM5 = 0)
1：门控时钟分配到 **CkIn5** (DM5 = 1)

位 13 **DFSDM2_CK04SEL**：由 TIM3 OC4 门控的 DFSDM2 比特流时钟分配 (Distribution of the DFSDM2 bitstream clock gated by TIM3 OC4)（DM6 多路解复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：门控时钟分配到 **CkIn0** (DM6 = 0)
1：门控时钟分配到 **CkIn4** (DM6 = 1)

位 12 **DFSDM2_D6SEL**：DFSDM2 的 **DatIn6** 的源选择 (Source selection for DatIn6 of DFSDM2)（M20 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：**DatIn6** 的源来自于 DFSDM2_DATIN6 引脚 (M20 = 0)
1：**DatIn6** 与 **DatIn7** 共享相同的数据 (M20 = 1)

位 11 **DFSDM2_D4SEL**：DFSDM2 的 **DatIn4** 的源选择 (Source selection for DatIn4 of DFSDM2)（M19 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：**DatIn4** 的源来自于 DFSDM2_DATIN4 引脚 (M19 = 0)
1：**DatIn4** 与 **DatIn5** 共享相同的数据 (M19 = 1)

位 10 **DFSDM2_D2SEL**：DFSDM2 的 **DatIn2** 的源选择 (Source selection for DatIn2 of DFSDM2)（M18 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：**DatIn2** 的源来自于 DFSDM2_DATIN2 引脚 (M18 = 0)
1：**DatIn2** 与 **DatIn3** 共享相同的数据 (M18 = 1)

位 9 **DFSDM2_D0SEL**：DFSDM2 的 **DatIn0** 的源选择 (Source selection for DatIn0 of DFSDM2)（M17 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：**DatIn0** 的源来自于 DFSDM2_DATIN0 引脚 (M17 = 0)
1：**DatIn0** 与 **DatIn1** 共享相同的数据 (M17 = 1)

位 8 **MCHDLYEN2**：DFSDM2 的 MCHDLY 时钟使能 (MCHDLY clock enable for DFSDM2)（G3、G4、G5、G6 门控信号，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：禁止 DFSDM2 的延迟时钟 (G[6:3] = 0)
1：使能 DFSDM2 的延迟时钟 (G[6:3] = 1)

位 7 **DFSDM1_CKOSEL**：DFSDM1_CKOUT 的源选择 (Source selection for DFSDM1_CKOUT)（M1 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：DFSDM1_CKOUT 的源是由 DFSDM1 生成的 **CkOut** (M1 = 0)
1：DFSDM1_CKOUT 的源是 M27 的输出 (M1 = 1)

位 6 **DFSDM1_CFG**：DFSDM1 的 **CkIn** 源选择 (CkIn source selection for DFSDM1)（M3、M4、M5、M6 多路复用器，请参见图 1）
0：**CkIn**[3:0] 信号源由引脚 DFSDM1_CKINy 提供 (M[6:3] = 0)
1：**CkIn**[3:0] 信号源由 DM[2:1] 的输出提供 (M[6:3] = 1)

位 5 **DFSDM1_CK13SEL**：由 TIM4 OC1 门控的 DFSDM1 比特流时钟分配 (Distribution of the DFSDM1 bitstream clock gated by TIM4 OC1)（DM1 多路解复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：门控时钟分配到 **CkIn1** (DM1 = 0)
1：门控时钟分配到 **CkIn3** (DM1 = 1)

位 4 **DFSDM1_CK02SEL**：由 TIM4 OC2 门控的 DFSDM1 比特流时钟分配 (Distribution of the DFSDM1 bitstream clock gated by TIM4 OC2)（DM2 多路解复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：门控时钟分配到 **CkIn0** (DM2 = 0)
1：门控时钟分配到 **CkIn2** (DM2 = 1)

位 3 **DFSDM1_D2SEL**：DFSDM1 的 **DatIn2** 的源选择 (Source selection for DatIn2 of DFSDM1)（M8 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：**DatIn2** 的源来自于 DFSDM1_DATIN2 引脚 (M8 = 0)
1：**DatIn2** 与 **DatIn3** 共享相同的数据 (M8 = 1)

位 2 **DFSDM1_D0SEL**：DFSDM1 的 **DatIn0** 的源选择 (Source selection for DatIn0 of DFSDM1)（M7 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：**DatIn0** 的源来自于 DFSDM1_DATIN0 引脚 (M7 = 0)
1：**DatIn0** 与 **DatIn1** 共享相同的数据 (M7 = 1)

位 1 **MCHDLYEN1**：DFSDM1 的 MCHDLY 时钟使能 (MCHDLY clock enable for DFSDM1)（G1、G2 门控信号，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：禁止 DFSDM1 的延迟时钟 (G[2:1] = 0)
1：使能 DFSDM1 的延迟时钟 (G[2:1] = 1)

位 0 **BSCKSEL**：比特流时钟源选择 (Bitstream clock source selection)（M27、M28、M29 多路复用器，请参见*图 81：用于脉冲跳过的多通道延迟模块*）
0：停止比特流时钟的时钟源 (M[29:27] = 0)
1：选择 DFSDM2 作为比特流时钟的时钟源 (M[29:27] = 1)

8.2.11 SYSCFG 寄存器映射

下表列出了 SYSCFG 寄存器映射和复位值。

表 29. 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM_MODE	
	Reset value																															x	x
0x04	**SYSCFG_PMC**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADC1DC2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																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
0x1C	**SYSCFG_CFGR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PVDL	Res.	CLL
	Reset value																														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
0x2C	**SYSCFG_CFGR**	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.	I2CFMP1_SDA	I2CFMP1_SCL
	Reset value																									0						0	0
0x30	**SYSCFG_MCHDLYCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DFSDM2_CKOSEL	DFSDM2_CFG	DFSDM2_CK37SEL	DFSDM2_CK26SEL	DFSDM2_CK15SEL	DFSDM2_CK04SEL	DFSDM2_D6SEL	DFSDM2_D4SEL	DFSDM2_D2SEL	DFSDM2_D0SEL	MCHDLYEN2	DFSDM1_CKOSEL	DFSDM1_CFG	DFSDM1_CK13SEL	DFSDM1_CK02SEL	DFSDM1_D2SEL	DFSDM1_D0SEL	MCHDLYEN1	BSCKSEL
	Reset value														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 54 页的第 2.2.2 节*。

9 直接存储器访问控制器 (DMA)

9.1 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 资源可供其他操作使用。

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（DMA1 和 DMA2）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 16 个通道（或称请求）。每个 DMA 控制器都有一个仲裁器，用于处理 DMA 请求间的优先级。

9.2 DMA 主要特性

DMA 主要特性是：

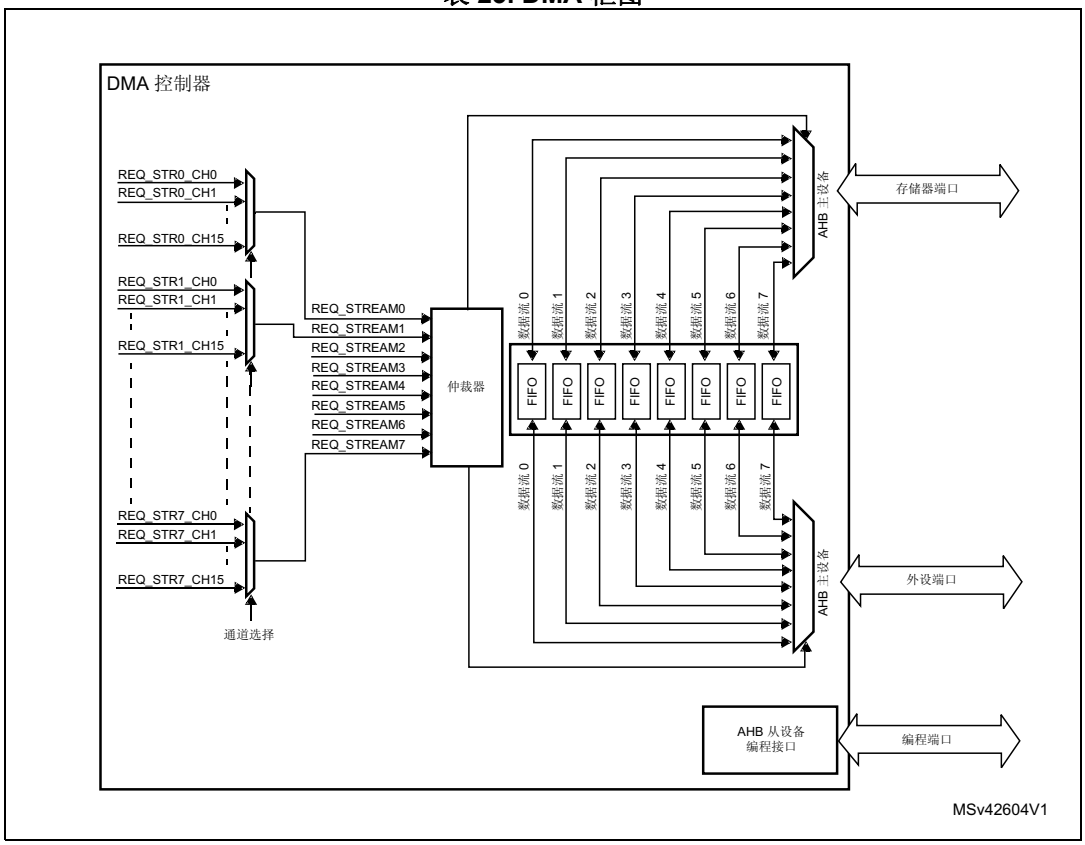
- 双 AHB 主总线架构，一个用于存储器访问，另一个用于外设访问
- 仅支持 32 位访问的 AHB 从编程接口
- 每个 DMA 控制器有 8 个数据流，每个数据流有多达 16 个通道（或称请求）
- 每个数据流有四个字深的 32 位先进先出存储器缓冲区 (FIFO)，可用于 FIFO 模式或直接模式：
  - FIFO 模式：可通过软件将阈值级别选取为 FIFO 大小的 1/4、1/2 或 3/4
  - 直接模式：每个 DMA 请求会立即启动对存储器的传输。当在直接模式（禁止 FIFO）下将 DMA 请求配置为以存储器到外设模式传输数据时，DMA 仅会将一个数据从存储器预加载到内部 FIFO，从而确保一旦外设触发 DMA 请求时则立即传输数据。
- 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
- 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 16 个。此选择可由软件配置，允许多个外设发起 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

9.3 DMA 功能说明

9.3.1 DMA 框图

表 23 显示了 DMA 的框图。

表 23. DMA 框图



9.3.2 DMA 概述

DMA 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）。

9.3.3 DMA 事务

DMA 事务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：

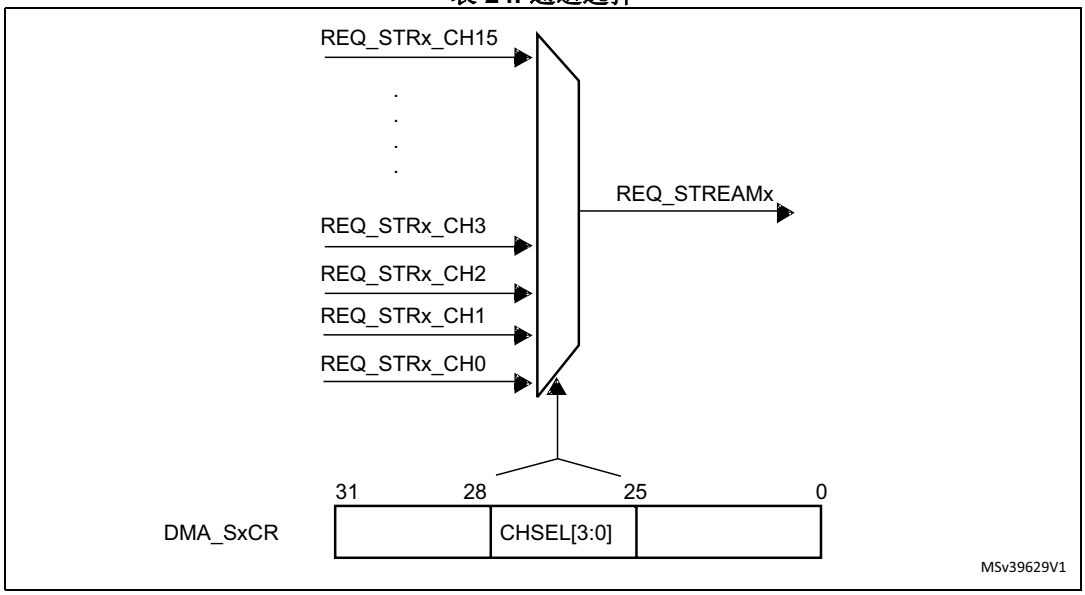
- 通过 DMA_SxPAR 或 DMA_SxM0AR 寄存器寻址，从外设数据寄存器或存储器单元中加载数据
- 通过 DMA_SxPAR 或 DMA_SxM0AR 寄存器寻址，将加载的数据存储到外设数据寄存器或存储器单元
- DMA_SxNDTR 寄存器在数据存储结束后递减，该寄存器中包含仍需执行的事务数。

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。

9.3.4 通道选择

每个数据流都与一个 DMA 请求相关联，此 DMA 请求可以从 16 个可能的通道请求中选出。此选择由 DMA_SxCR 寄存器中的 CHSEL[3:0] 位控制。

表 24. 通道选择



来自外设的 16 个请求（例如，TIM、ADC、SPI、I2C）独立连接到每个通道，具体的连接取决于产品实现情况。

*表 30*和*表 31* 给出了 DMA 请求映射的示例。

表 30. 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	I2C1_TX	SPI3_RX	SPI2_RX	SPI2_TX	SPI3_TX	-	SPI3_TX
通道 1	I2C1_RX	I2C3_RX	TIM7_UP	I2CFMP1_RX	TIM7_UP	I2C1_RX	I2C1_TX	I2C1_TX
通道 2	TIM4_CH1	I2CFMP1_TX	I2S3_EXT_RX	TIM4_CH2	I2S2_EXT_TX	I2S3_EXT_TX	TIM4_UP	TIM4_CH3
通道 3	I2S3_EXT_RX	TIM2_UP TIM2_CH3	I2C3_RX	I2S2_EXT_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	I2CFMP1_TX
通道 5	UART8_RX	UART7_TX	TIM3_UP TIM3_CH4	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	I2C3_TX	TIM5_UP	USART2_RX
通道 7	I2CFMP1_RX	TIM6_UP	I2C2_RX	I2C2_RX	USART3_TX	DAC1	DAC2	I2C2_TX
通道 8	-	-	-	-	-	-	-	UART5_TX

表 31. 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	UART9_RX
通道 1	UART9_TX	-	-	-	SAI1B	-	-	-
通道 2	-	-	SPI1_TX	SPI5_RX	SPI5_TX	AES_OUT(1)	AES_IN(1)	-
通道 3	SPI1_RX	DFSDM1_ FLT1	SPI1_RX	SPI1_TX	DFSDM1_ FLT1	SPI1_TX	DFSDM1_ FLT0	QUADSPI
通道 4	SPI4_RX	SPI4_TX	USART1_RX	SDIO	SPI4_RX	USART1_RX	SDIO	USART1_TX
通道 5	UART10_RX	USART6_RX	USART6_RX	SPI4_RX	SPI4_TX	SPI5_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	UART10_TX
通道 7	DFSDM1_ FLT0	TIM8_UP	TIM8_CH1	TIM8_CH2	TIM8_CH3	SPI5_RX	SPI5_TX	TIM8_CH4 TIM8_TRIG TIM8_COM
通道 8	DFSDM2_ FLT0	DFSDM2_ FLT1	DFSDM2_ FLT2	DFSDM2_ FLT3	DFSDM2_ FLT0	DFSDM2_ FLT1	DFSDM2_ FLT2	DFSDM2_ FLT3
通道 9	-	-	-	UART10_RX	-	UART10_TX	-	-

1. 仅在 STM32F423xx 上可用。

9.3.5 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。

9.3.6 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/自其他缓冲区的写/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。

9.3.7 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 32* 介绍了相应的源和目标地址。

表 32. 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

表 25 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。

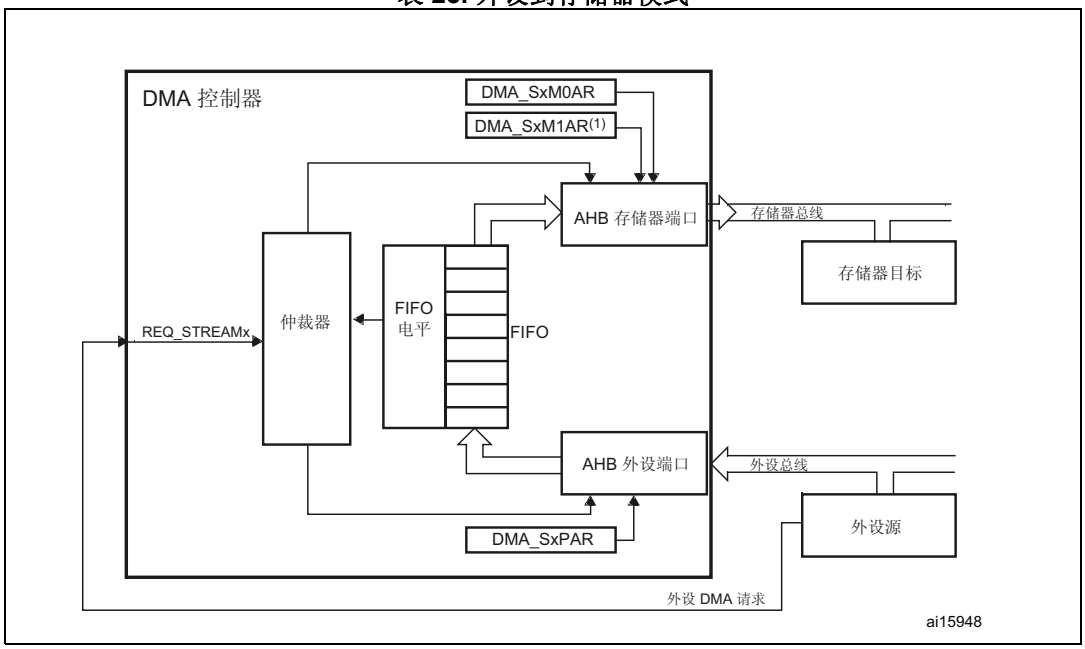
达到 FIFO 的阈值级别时，FIFO 的内容移出并存储到目标中。

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。

表 25. 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

表 26 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。

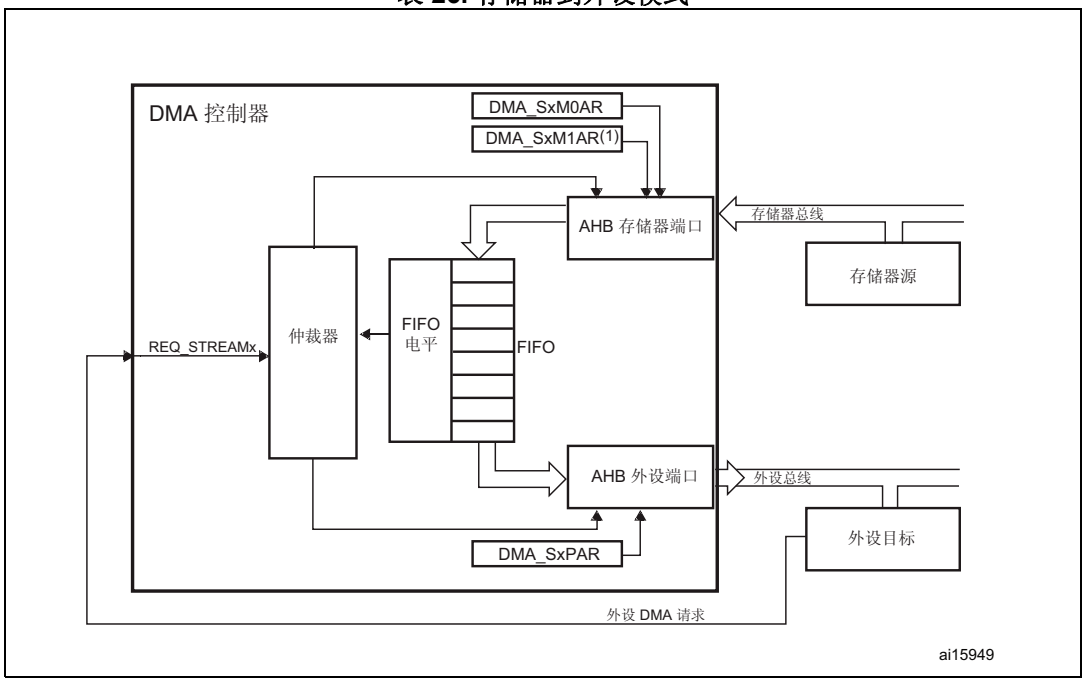
每次发生外设请求，FIFO 的内容都会移出并存储到目标中。当 FIFO 的级别小于或等于预定义的阈值级别时，将使用存储器中的数据完全重载 FIFO。

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 位域的值。

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。

表 26. 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*表 27* 中介绍的存储器到存储器模式。

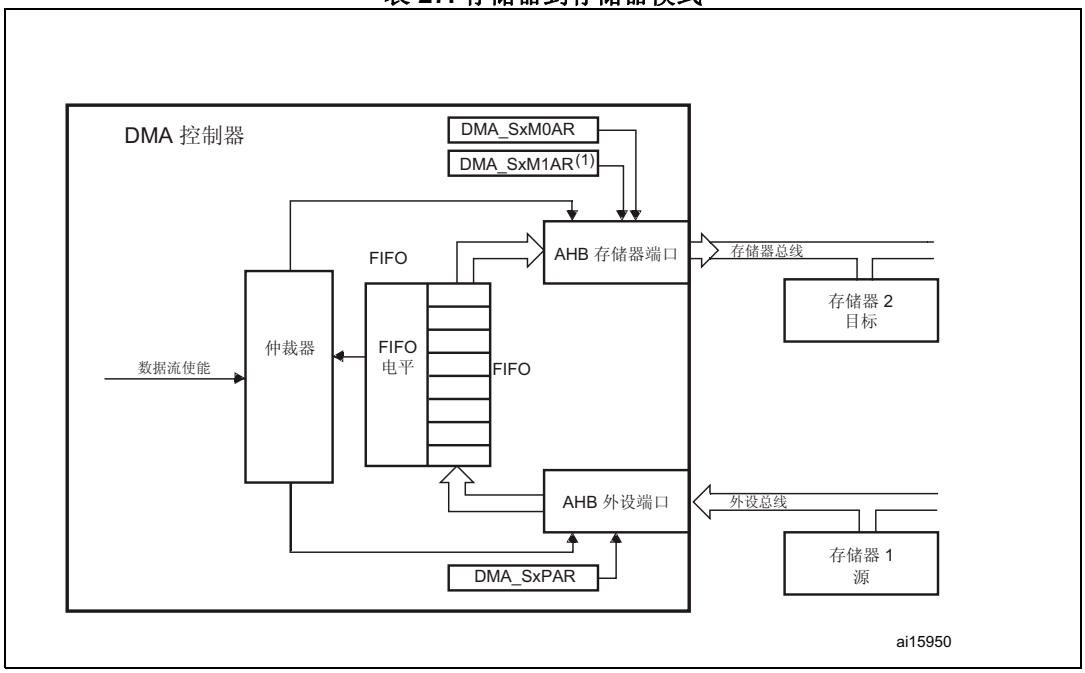
通过将 DMA_SxCR 寄存器中的使能位 (EN) 置 1 来使能数据流时，数据流会立即开始填充 FIFO，直至达到阈值级别。达到阈值级别后，FIFO 的内容便会移出，并存储到目标中。

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。*

表 27. 存储器到存储器模式



1. 用于双缓冲区模式。

9.3.8 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。

9.3.9 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。

9.3.10 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 33：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）。

对于所有其他模式（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 33. 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。

9.3.11 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 34：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 9.3.12 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 寄存器中的 PSIZE 位定义，MSIZE 位的状态是“无关”。

表 34. 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]

表 34. 封装/解封和字节序行为（位 PINC = MINC = 1）（续）

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
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 35* 中。

表 35. 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

9.3.12 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**不得**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。

9.3.13 FIFO

FIFO 结构

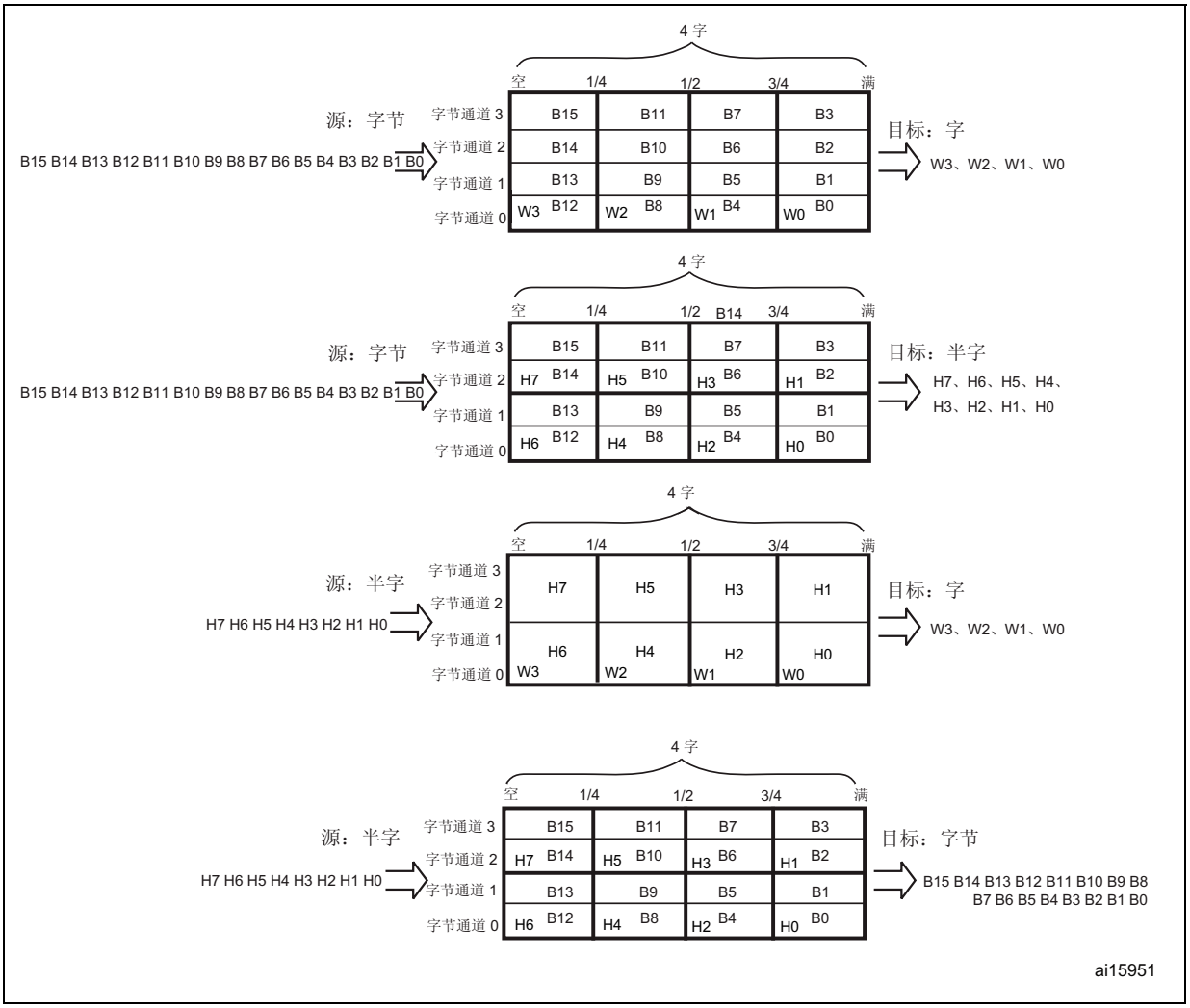
FIFO 用于在源数据传输到目标之前临时存储这些数据。

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 28：FIFO 结构*中介绍。

表 28. 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 36*中介绍。禁止的配置在表中以灰色突出显示。

表 36. 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 36. 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。

9.3.14 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。

9.3.15 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： *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。*

9.3.16 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。

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。*

9.3.17 可能的 DMA 配置汇总

表 37 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 37. 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		

9.3.18 流配置过程

配置 DMA 数据流 x（其中 x 是数据流编号）时必须遵守下面的顺序：

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。必须将先前的数据块 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[3:0] 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。**

9.3.19 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 9.3.10 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 36：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。*

9.4 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 38* 所示。

表 38. 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，必须将相应的事件标志清零，否则会立即产生中断。*

9.5 DMA 寄存器

DMA 寄存器必须按字（32 位）进行访问。

9.5.1 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
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

位 31:28、15:12 保留，必须保持复位值。

位 27、21、11、5 **TCIF[3:0]**：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 **HTIF[3:0]**：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 **TEIF[3:0]**：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 **DMEIF[3:0]**：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 **FEIF[3:0]**：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

9.5.2 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 **TCIF[7:4]**：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 **HTIF[7:4]**：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 **TEIF[7:4]**：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 **DMEIF[7:4]**：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 **FEIF[7:4]**：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

9.5.3 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 **CTCIF[3:0]**：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 **CHTIF[3:0]**：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 **CTEIF[3:0]**：数据流 x 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 **CDMEIF[3:0]**：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 **CFEIF[3:0]**：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

9.5.4 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 **CTCIF[7:4]**：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 **CHTIF[7:4]**：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 **CTEIF[7:4]**：数据流 x 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 **CDMEIF[7:4]**：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 **CFEIF[7:4]**：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

9.5.5 DMA 数据流 x 配置寄存器 (DMA_SxCR)

DMA stream x configuration register

此寄存器用于配置相关数据流。

偏移地址：0x10 + 0x18 * x，（x = 0 到 7）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	CHSEL[3:0]				MBURST[1:0]		PBURST[1:0]		Res.	CT	DBM	PL[1:0]	
			rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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 31:29 保留，必须保持复位值。

位 28:25 **CHSEL[3:0]**：通道选择 (Channel selection)

这些位将由软件置 1 和清零。

0000：选择通道 0
0001：选择通道 1
0010：选择通道 2
0011：选择通道 3
0100：选择通道 4
0101：选择通道 5
0110：选择通道 6
0111：选择通道 7
1000：选择通道 8
1001：选择通道 9
1010：选择通道 10
1011：选择通道 11
1100：选择通道 12
1101：选择通道 13
1110：选择通道 14
1111：选择通道 15

这些位受到保护，只有 EN 为“0”时才可以写入

位 24:23 **MBURST[1:0]**：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。

9.5.6 DMA 数据流 x 数据项数寄存器 (DMA_SxNDTR)

DMA stream x number of data register

偏移地址：0x14 + 0x18 * x，（x = 0 到 7）

复位值：0x0000 0000

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

9.5.7 DMA 数据流 x 外设地址寄存器 (DMA_SxPAR)

DMA stream x peripheral address register

偏移地址：0x18 + 0x18 * x，（x = 0 到 7）

复位值：0x0000 0000

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

9.5.8 DMA 数据流 x 存储器 0 地址寄存器 (DMA_SxM0AR)

DMA stream x memory 0 address register

偏移地址：0x1C + 0x18 * x，（x = 0 到 7）

复位值：0x0000 0000

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

9.5.9 DMA 数据流 x 存储器 1 地址寄存器 (DMA_SxM1AR)

DMA stream x memory 1 address register

偏移地址：0x20 + 0x18 * x，（x = 0 到 7）

复位值：0x0000 0000

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

9.5.10 DMA 数据流 x FIFO 控制寄存器 (DMA_SxFCR)

DMA stream x FIFO control register

偏移地址：0x24 + 0x24 * x，（x = 0 到 7）

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

其他：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。

9.5.11 DMA 寄存器映射

表 39 汇总了 DMA 寄存器。

表 39. DMA 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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	Res.	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		FEIF7	TCIF6	HTIF6	TEIF6	DMEIF6		FEIF6					TCIF5	HTIF5	TEIF5	DMEIF5		FEIF5	TCIF4	HTIF4	TEIF4	DMEIF4		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	Res	CFEIF3	CTCIF2	CHTIF2	CTEIF2	CDMEIF2	Res	CFEIF2	Res	Res	Res	Res	CTCIF1	CHTIF1	CTEIF1	CDMEIF1	Res	CFEIF1	CTCIF0	CHTIF0	CTEIF0	CDMEIF0	Res	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	Res	CFEIF7	CTCIF6	CHTIF6	CTEIF6	CDMEIF6	Res	CFEIF6	Res	Res	Res	Res	CTCIF5	CHTIF5	CTEIF5	CDMEIF5	Res	CFEIF5	CTCIF4	CHTIF4	CTEIF4	CDMEIF4	Res	CFEIF4
	Reset value					0	0	0	0		0	0	0	0	0		0					0	0	0	0		0	0	0	0	0		0
0x0010	DMA_S0CR	Res	Res	Res	CHSEL[3:0]				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	0	0
0x0014	DMA_S0NDTR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:0]															
	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	FEIE	Res	FS[2:0]			DMDIS	FTH[1:0]		
	Reset value																								0		1	0	0	0	0	1	
0x0028	DMA_S1CR	Res	Res	Res	CHSEL[3:0]				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	0	0
0x002C	DMA_S1NDTR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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

表 39. DMA 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0034	DMA_S1M0AR	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	CHSEL[3:0]				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	0
0x0044	DMA_S2NDTR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:0]															
	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	CHSEL[3:0]				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	0
0x005C	DMA_S3NDTR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:0]															
	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
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

表 39. DMA 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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	CHSEL[3:0]				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	0
0x0074	DMA_S4NDTR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:0]															
	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	CHSEL[3:0]				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	0
0x008C	DMA_S5NDTR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:0]															
	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	CHSEL[3:0]				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	0

表 39. DMA 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00A4	DMA_S6NDTR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:0]															
	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
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	CHSEL[3:0]				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	0
0x00BC	DMA_S7NDTR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	NDT[15:0]															
	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 54 页的第 2.2.2 节*。

10 中断和事件

10.1 嵌套向量中断控制器 (NVIC)

10.1.1 NVIC 特性

嵌套向量中断控制器 NVIC 包含以下特性：

- 52 个可屏蔽中断通道（不包括带 FPU 的 Cortex®-M4 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 PM0214。

10.1.2 SysTick 校准值寄存器

SysTick 校准值设置为 10500。当 SysTick 时钟设置为 10.5 MHz（HCLK/8，HCLK 设为 84 MHz），会产生 1 ms 时间基准。

10.1.3 中断和异常向量

请参见*表 40*，了解 STM32F413/423 器件的向量表。

10.2 外部中断/事件控制器 (EXTI)

外部中断/事件控制器包含多达 23 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。

表 40. STM32F413/423 的向量表

位置	优先级	优先级类型	缩略语	说明	地址
-	-	-	-	保留	0x0000 0000
-	-3	固定	Reset	复位	0x0000 0004
-	-2	固定	NMI	不可屏蔽中断，时钟安全系统	0x0000 0008
-	-1	固定	HardFault	所有类型的错误	0x0000 000C
-	0	可设置	MemManage	存储器管理	0x0000 0010
-	1	可设置	BusFault	预取指失败，存储器访问失败	0x0000 0014
-	2	可设置	UsageFault	未定义的指令或非法状态	0x0000 0018
-	-	-	-	保留	0x0000 001C - 0x0000 002B
-	3	可设置	SVCall	通过 SWI 指令调用的系统服务	0x0000 002C
-	4	可设置	Debug Monitor	调试监控器	0x0000 0030
-		-	-	保留	0x0000 0034
-	5	可设置	PendSV	可挂起的系统服务请求	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

表 40. STM32F413/423 的向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
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 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
26	33	可设置	TIM_TRG_COM_TIM11	TIM1 触发和通信中断 和 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_EVT	I2C1 全局事件中断	0x0000 00BC
32	39	可设置	I2C1_ERR	I2C1 全局错误中断	0x0000 00C0
33	40	可设置	I2C2_EVT	I2C2 全局事件中断	0x0000 00C4
34	41	可设置	I2C2_ERR	I2C2 全局错误中断	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	USART 3	USART3 全局中断	0x0000 00DC
40	47	可设置	EXTI15_10	EXTI 线 [15:10] 中断	0x0000 00E0

表 40. STM32F413/423 的向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
41	48	可设置	EXTI17/ RTC Alarm	EXTI 线 17 中断/ 连接到 EXTI 线的 RTC 闹钟（A 和 B）中断	0x0000 00E4
42	49	可设置	EXTI18/OTG_FS_WKUP	EXTI 线 18 中断/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 TIM12 全局中断	0x0000 00EC
44	51	可设置	TIM8_UP_TIM13	TIM8 更新中断 TIM13 全局中断	0x0000 00F0
45	52	可设置	TIM8_TRG_COM_TIM14	TIM8 触发和通信中断 TIM14 全局中断	0x0000 00F4
46	53	可设置	TIM8_CC	TIM8 Cap/Com 中断	0x0000 00F8
47	54	可设置	DMA1_Stream7	DMA1 通道 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	SDIO	SDIO 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
54	61	可设置	TIM6_GLB_IT/DAC1/DAC2	TIMER6、DAC1 和 DAC2 全局中断	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	DFSDM1_FLT0	SD 滤波器 0 全局中断	0x0000 0134
62	69	可设置	DFSDM1_FLT1	SD 滤波器 1 全局中断	0x0000 0138
63	70	可设置	CAN2_TX	CAN2 TX 中断	0x0000 013C
64	71	可设置	CAN2_RX0	BXCAN2 RX0 中断	0x0000 0140
65	72	可设置	CAN2_RX1	BXCAN2 RX1 中断	0x0000 0144
66	73	可设置	CAN2_SCE	CAN2 SCE 中断	0x0000 0148

表 40. STM32F413/423 的向量表（续）

位置	优先级	优先级类型	缩略语	说明	地址
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	CAN3_TX	CAN 3 TX 中断	0x0000 0168
75	82	可设置	CAN3_RX0	BxCAN 3 RX0 中断	0x0000 016C
76	83	可设置	CAN3_RX1	BxCAN 3 RX1 中断	0x0000 0170
77	84	可设置	CAN3_SCE	CAN 3 SCE 中断	0x0000 0174
79	86	可设置	CRYPTO(1)	AES 全局中断	0x0000 017C
80	87	可设置	RNG	RNG 全局中断	0x0000 0180
81	88	可设置	FPU	FPU 全局中断	0x0000 0184
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
87	94	可设置	SAI1	SAI1 全局中断	0x0000 019C
88	95	可设置	UART9	UART9 全局中断	0x0000 01A0
89	96	可设置	UART10	UART10 全局中断	0x0000 01A4
92	99	可设置	Quad-SPI	Quad-SPI 全局中断	0x0000 01B0
95	102	可设置	I2CFMP1 event	I2CFMP1 事件中断	0x0000 01BC
96	103	可设置	I2CFMP1 error	I2CFMP1 错误中断	0x0000 01C0
97	104	可设置	EXTI23/LPTIM1	LP 定时器全局中断或 EXTI 中断线 23	0x0000 01C4
98	105	可设置	DFSDM2_FLT0	DFSDM2 SD 滤波器 0 全局中断	0x0000 01C8
99	106	可设置	DFSDM2_FLT1	DFSDM2 SD 滤波器 1 全局中断	0x0000 01CC
100	107	可设置	DFSDM2_FLT2	DFSDM2 SD 滤波器 2 全局中断	0x0000 01D0
101	108	可设置	DFSDM2_FLT3	DFSDM2 SD 滤波器 3 全局中断	0x0000 01D4

1. 仅在 STM32F423xx 上可用。

10.2.1 EXTI 主要特性

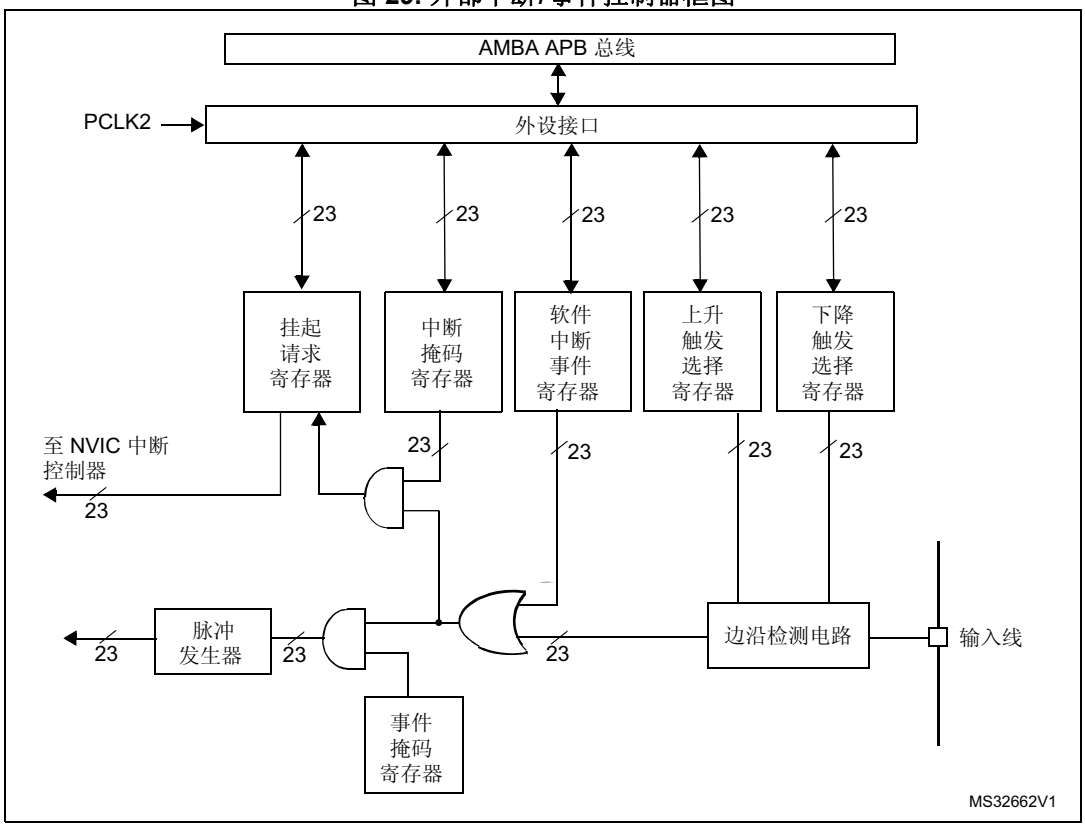
EXTI 控制器的主要特性如下：

- 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 23 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 STM32F4xx 数据手册的电气特性部分。

10.2.2 EXTI 框图

图 29 显示了框图。

图 29. 外部中断/事件控制器框图



10.2.3 唤醒事件管理

STM32F4xx 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，同时使能带 FPU 的 Cortex®-M4 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 IRQ 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 IRQ 通道挂起位。

使用外部线作为唤醒事件，请参见*第 10.2.4 节：功能描述*。

10.2.4 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。

硬件中断选择

要配置 23 根线作为中断源，请执行以下步骤：

- 配置 23 根中断线的屏蔽位 (EXTI_IMR)
- 配置中断线的触发选择位（EXTI_RTSR 和 EXTI_FTSR）
- 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 23 个中断线中的请求可以被正确地响应。

硬件事件选择

要配置 23 根线作为事件源，请执行以下步骤：

- 配置 23 根事件线的屏蔽位 (EXTI_EMR)
- 配置事件线的触发选择位（EXTI_RTSR 和 EXTI_FTSR）

软件中断/事件选择

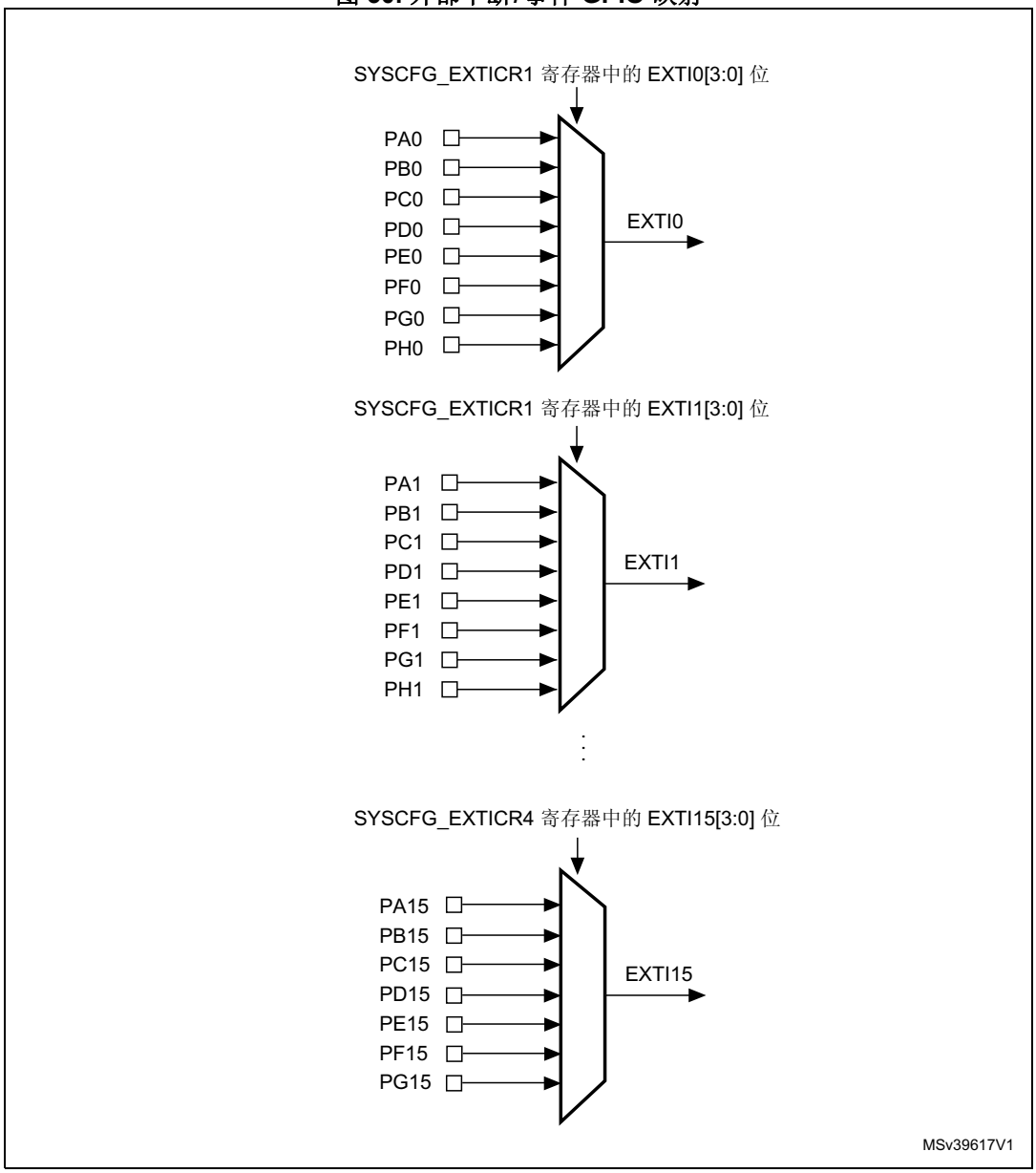
可将这 23 根线配置为软件中断/事件线。以下为产生软件中断的步骤。

- 配置 23 根中断/事件线的屏蔽位（EXTI_IMR、EXTI_EMR）
- 在软件中断寄存器设置相应的请求位 (EXTI_SWIER)

10.2.5 外部中断/事件线映射

STM32F413/423 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



另外五根 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 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

10.3 EXTI 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.2 节：寄存器相关缩写词列表*。

10.3.1 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	Res.	Res.	MR18	MR17	MR16
								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 23:21 **MR[23:21]**：x 线上的中断屏蔽 (Interrupt 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

位 20:19 保留，必须保持复位值。

位 18:0 **MR[18:0]**：x 线上的中断屏蔽 (Interrupt 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

10.3.2 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	Res.	Res.	MR18	MR17	MR16
								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 23:21 **MR[23:21]**：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

位 20:19 保留，必须保持复位值。

位 18:0 **MR[18:0]**：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

10.3.3 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	Res.	Res.	TR18	TR17	TR16
								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 23:21 **TR[23:21]**：x 线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）

位 20:19 保留，必须保持复位值。

位 18:0 **TR[18:0]**：x 线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 1。
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。

10.3.4 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	Res.	Res.	TR18	TR17	TR16
								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 23:21 **TR[23:21]**：x 线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）。

位 20:19 保留，必须保持复位值。

位 18:0 **TR[18:0]**：x 线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 1。
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。

10.3.5 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	Res.	Res.	SWIER 18	SWIER 17	SWIER 16
								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 23:21 **SWIER[23:21]**：x 线上的软件中断 (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”。

位 20:19 保留，必须保持复位值。

位 18:0 **SWIER[18:0]**：x 线上的软件中断 (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”。

10.3.6 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	Res.	Res.	PR18	PR17	PR16
								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 23:21 **PR[22:21]**：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。

位 20:19 保留，必须保持复位值。

位 18:0 **PR[18:0]**：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。

10.3.7 EXTI 寄存器映射

表 41 给出了 EXTI 寄存器映射和复位值。

表 41. 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 [23:21]			Res.	Res.	MR[18:0]																		
	Reset value									0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	**EXTI_EMR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MR [23:21]			Res.	Res.	MR[18:0]																		
	Reset value									0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	**EXTI_RTSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TR [23:21]			Res.	Res.	TR[18:0]																		
	Reset value									0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	**EXTI_FTSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TR [23:21]			Res.	Res.	TR[18:0]																		
	Reset value									0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	**EXTI_SWIER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWIER [23:21]			Res.	Res.	SWIER[18:0]																		
	Reset value									0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	**EXTI_PR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR [23:21]			Res.	Res.	PR[18:0]																		
	Reset value									0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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*。

11 灵活的静态存储控制器 (FSMC)

可变静态存储控制器 (FSMC) 包括一个存储控制器：

- NOR/PSRAM 存储控制器

11.1 FSMC 主要特性

FSMC 功能块可连接：同步和异步静态存储器。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。FSMC 一次只能访问一个外部器件。

FSMC 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
  - PSRAM（4 个存储区域）
- 连接并行 LCD 模块，支持 Intel 8080 和 Motorola 6800 模式
- 支持突发模式，能够更快速地访问同步器件（如 NOR Flash，PSRAM）
- 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 器件使用
- 外部异步等待控制
- 16 x 32 位深度写 FIFO

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）。在此情况下，AHB 突发传输将分成两个 FIFO 条目。

通过将 FSMC_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 FSMC 引脚进行配置。应用程序未使用的 FSMC I/O 引脚可用于其他用途。

定义外部器件类型和其特性的 FSMC 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。

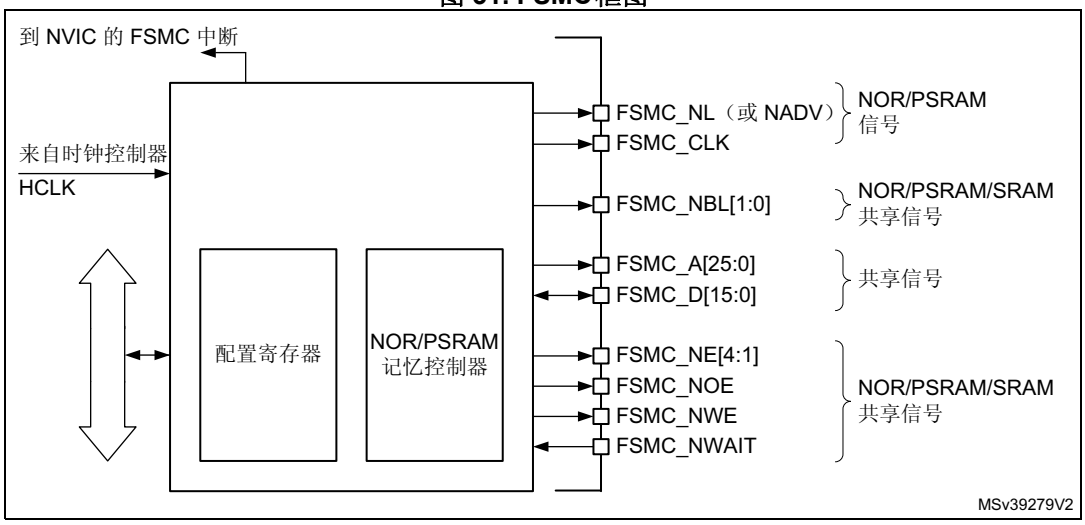
11.2 FSMC 框图

FSMC 包含以下主要模块：

- AHB 接口（包括 FSMC 配置寄存器）
- NOR Flash/PSRAM/SRAM 控制器

框图如下图所示。

图 31. FSMC框图



11.3 AHB 接口

CPU 和其他 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。除非在使能扩展模式时在 D 模式下进行访问，否则 FSMC 片选 (FSMC_NEx) 在连续两次访问之间不会翻转。

出现以下条件时，FSMC 将产生 AHB 错误：

- 读取或写入未使能的 FSMC 存储区域（存储区域 1 到 4）。
- 在 FSMC_BCRx 寄存器中的 FACCEN 位复位时读取或写入 NOR Flash 存储区域。

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 的 Cortex®-M4 CPU，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 FSMC 的参考时钟。

11.3.1 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，FSMC 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，FSMC 片选 (FSMC_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 和 PSRAM）
    FSMC 允许读/写事务并通过其字节选择通道 NBL[1:0] 访问恰当的数据。
    通过 NBL[1:0] 寻址要写入的字节。
    读取所有存储器字节（NBL[1:0] 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)
    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 的回卷支持

不支持同步存储器的回绕突发模式。存储器必须按未定义长度的线性突发模式进行配置。

配置寄存器

可通过一组寄存器配置 FSMC。有关 NOR Flash/PSRAM 控制寄存器的详细说明，请参见 *第 11.5.6 节*。

11.4 外部设备地址映射

从 FSMC 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 32*）：

- 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

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 32. FSMC 存储区域

11.4.1 NOR/PSRAM 地址映射

HADDR[27:26] 位用于从*表 42* 中所示的四个存储区域之中选择其中一个存储区域。

表 42. 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 43. 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

1. 如果外部存储器的宽度为 16 位，FSMC 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 FSMC_A[24:0]。
   无论外部存储器的宽度是多少，FSMC_A[0] 都应连接到外部存储器地址 A[0]。

11.5 NOR Flash/PSRAM 控制器

FSMC 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
- PSRAM (CellularRAM™)
  - 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

FSMC 会为每个存储区域输出唯一的片选信号 NE[4:1]。所有其他信号（地址、数据和控制）均为共享信号。

FSMC 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 (FSMC_CLK) 输出

FSMC 时钟 (FSMC_CLK) 是 HCLK 时钟的约数。该时钟可传送到选定的外部器件，根据 FSMC_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 位复位，则 FSMC 仅在同步访问（读/写事务）期间生成时钟 (CLK)。
- 如果 CCLKEN 位置 1，则 FSMC 将在异步和同步访问期间生成连续时钟。要生成 FSMC_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 11.5.6 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 FSMC_CLK 频率将根据 AHB 数据事务发生变化（有关 FSMC_CLK 分频比公式的信息，请参见*第 11.5.5 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 11.5.6 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 44*）和对等待管理的支持（用于在突发模式下访问 NOR Flash 和 PSRAM）。

表 44. 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

11.5.1 外部存储器接口信号

表 45、*表 46* 和*表 47* 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 45. 非复用 I/O NOR Flash

FSMC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[15:0]	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	FSMC 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 46. 16 位复用 I/O NOR Flash

FSMC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	FSMC 的 NOR Flash 等待输入信号

最大容量为 512 Mb。

PSRAM/SRAM，非复用 I/O

表 47. 非复用 I/O PSRAM/SRAM

FSMC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[15:0]	I/O	数据双向总线
NE[x]	O	片选，x = 1..4 （在 PSRAM 应用中被称作 NCE（CellularRAM™，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	FSMC 的 PSRAM 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。

PSRAM，16 位复用 I/O

表 48. 16 位复用 I/O PSRAM

FSMC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（CellularRAM™，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	FSMC 的 PSRAM 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）。

11.5.2 支持的存储器和事务

下面的*表 49* 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 FSMC 不允许（或不支持）的事务以灰色显示。

表 49. 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	否	-
	异步	R	16	16	是	-
	异步	W	16	16	是	-
	异步	R	32	16	是	分为 2 次 FSMC 访问
	异步	W	32	16	是	分为 2 次 FSMC 访问
	异步页	R	-	16	否	不支持该模式
	同步	R	8	16	否	-
	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 FSMC 访问
	异步	W	32	16	是	分为 2 次 FSMC 访问
	异步页	R	-	16	否	不支持该模式
	同步	R	8	16	否	-
	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 FSMC 访问
	异步	W	32	16	是	分为 2 次 FSMC 访问，使用字节通道 NBL[1:0]

11.5.3 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 FSMC_CLK 时钟的下降沿变化。
  - NOEH/NWEH/NEH/NOEH/NBLH/ 地址有效输出在 FSMC_CLK 时钟的上升沿变化。

11.5.4 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
- FSMC 总是会先对数据进行采样，而后再禁止信号 NOE。这样可以确保符合存储器数据保持时序的要求（数据转换的最小芯片使能高电平通常为 0 ns）
- 如果使能扩展模式（FSMC_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（FSMC_BCRx 寄存器中的 EXTMOD 位复位），则 FSMC 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（FSMC_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（FSMC_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 FSMC _BCRx 和 FSMC_BTRx/ FSMC_BWTRx 寄存器配置。

图 33. 模式 1 读取访问波形

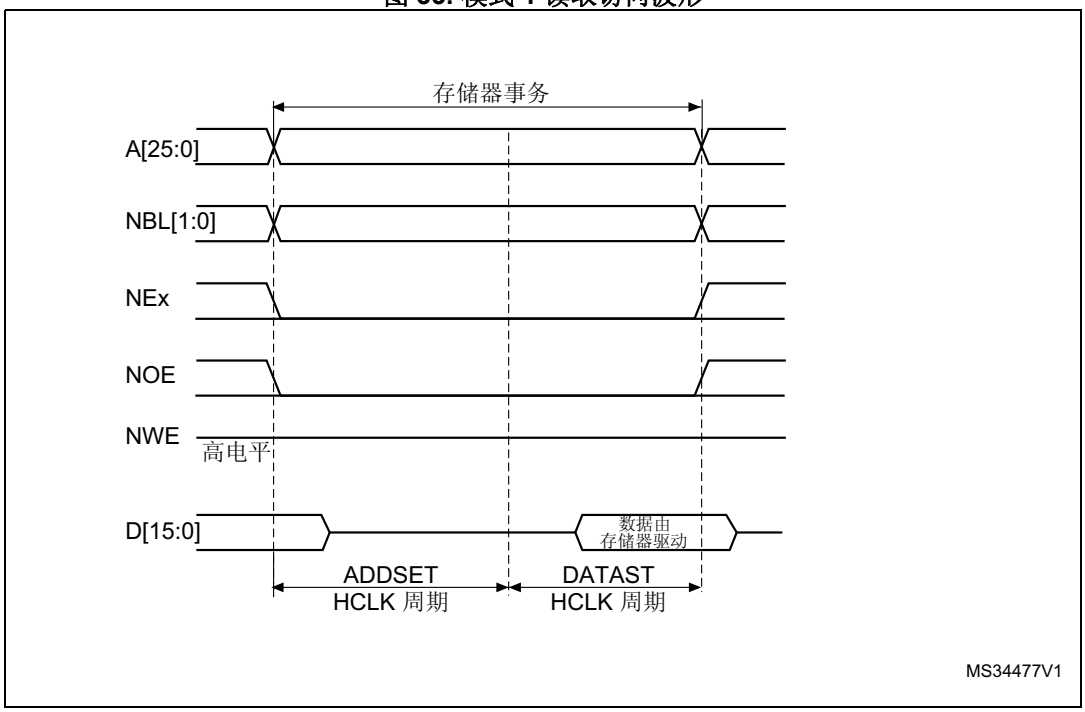
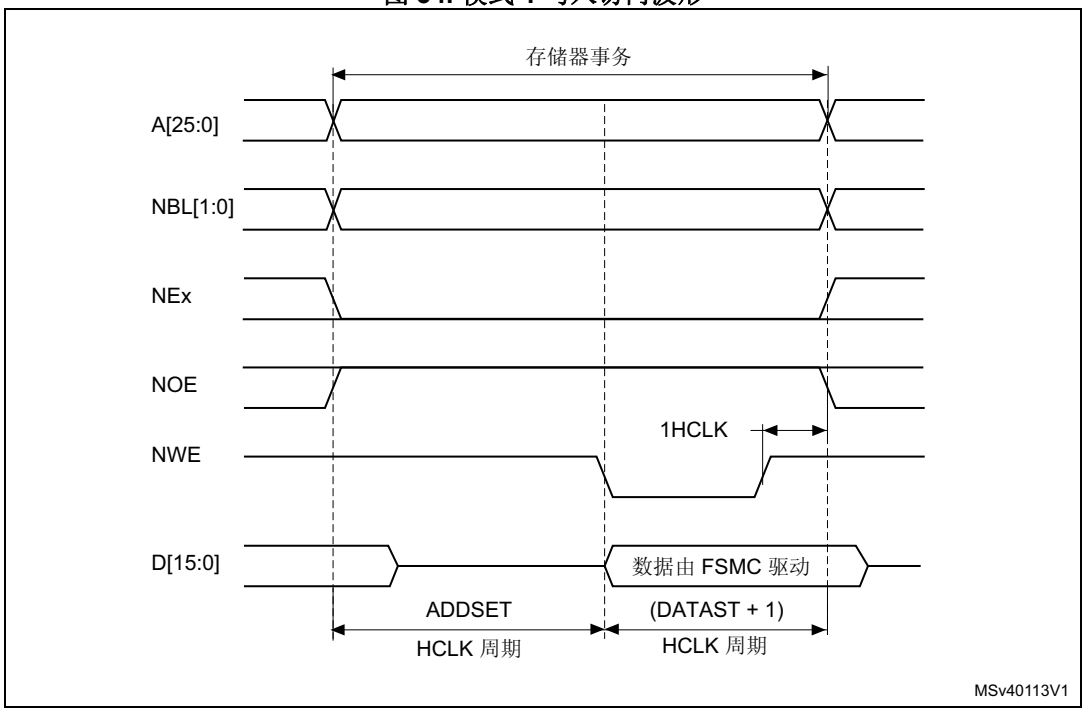


图 34. 模式 1 写入访问波形



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 50. FSMC_BCRx 位域

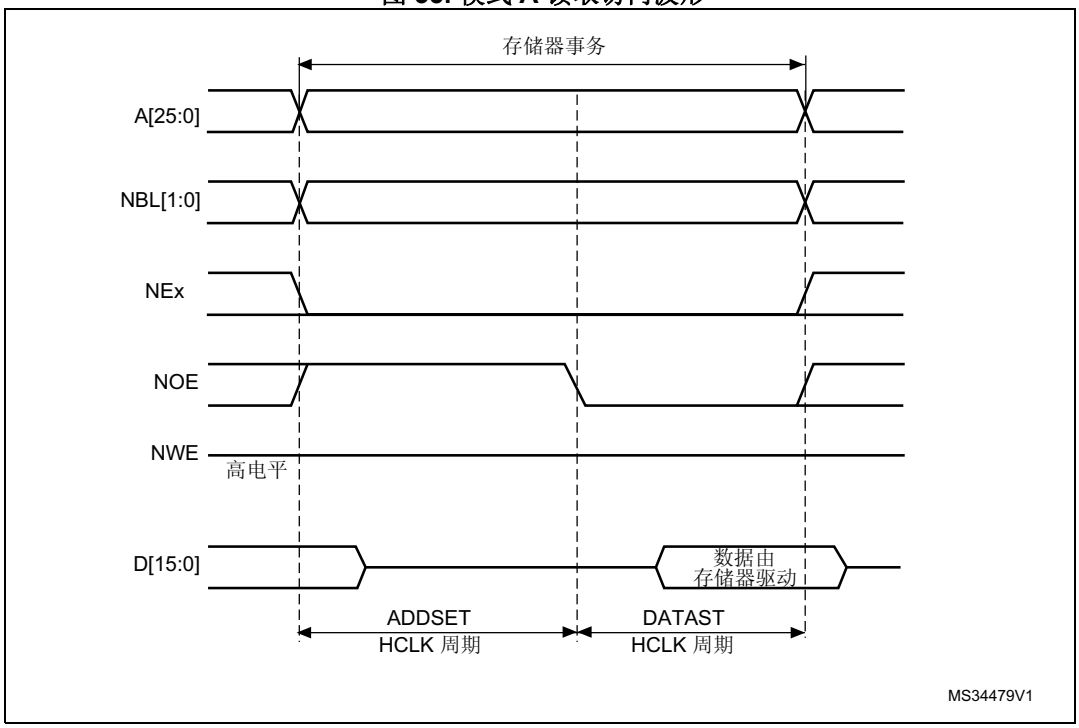
位号	位名	要设置的值
31:22	保留	0x000
21	WFDIS	根据需要进行设置
20	CCLKEN	根据需要进行设置
19	CBURSTRW	0x0（对异步模式没有影响）
18:16	CPSIZE	0x0（对异步模式没有影响）
15	ASYNCWAIT	如果存储器支持该特性，则置为 1。否则，保持为 0
14	EXTMOD	0x0
13	WAITEN	0x0（对异步模式没有影响）
12	WREN	根据需要进行设置
11	保留	0x0
10	保留	0x0
9	WAITPOL	仅当位 15 为 1 时才有意义
8	BURSTEN	0x0
7	保留	0x1
6	FACCEN	无关
5:4	MWID	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	MUXE	0x0
0	MBKEN	0x1

表 51. FSMC_BTRx 位域

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。

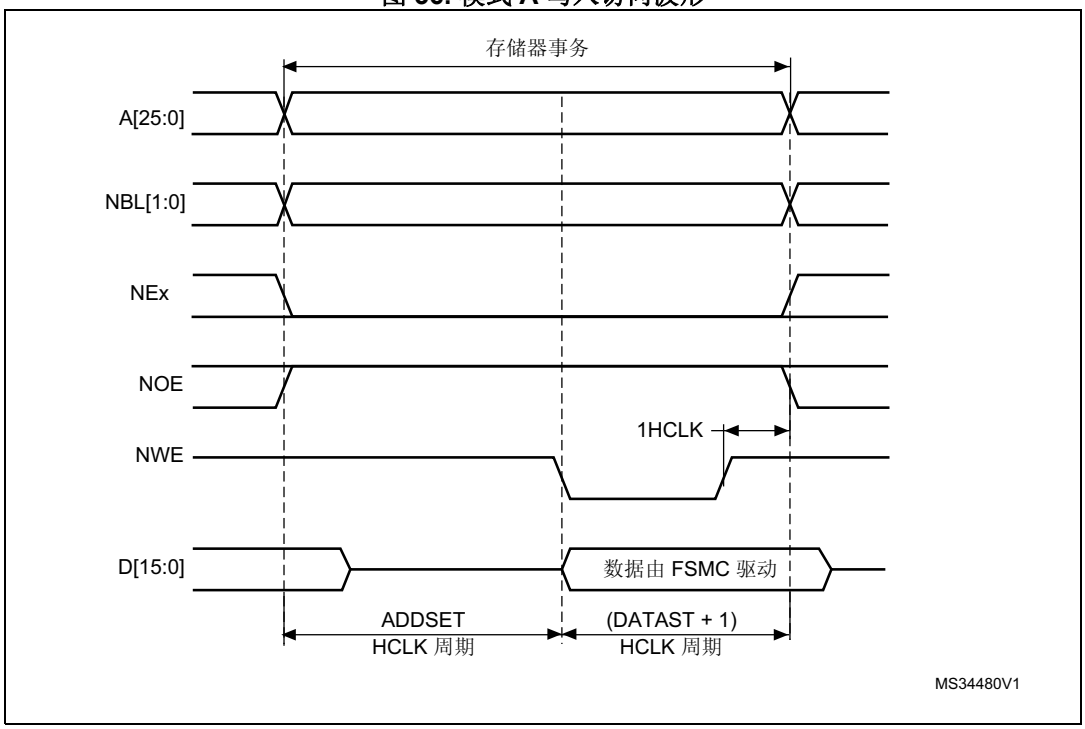
模式 A - SRAM/PSRAM (CRAM) OE 切换

图 35. 模式 A 读取访问波形



1. NBL[1:0] 在进行读取访问时为低电平

图 36. 模式 A 写入访问波形



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 52. FSMC_BCRx 位域

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	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 53. FSMC_BTRx 位域

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	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 54. FSMC_BWTRx 位域

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 37. 模式 2 和模式 B 读取访问波形

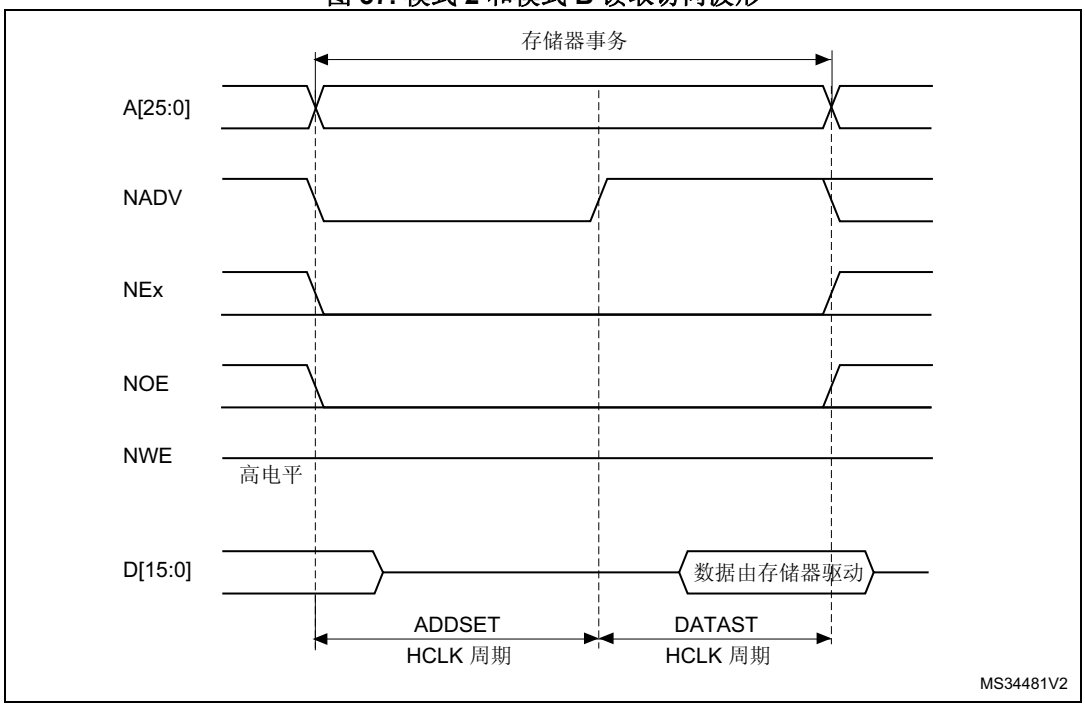


图 38. 模式 2 写入访问波形

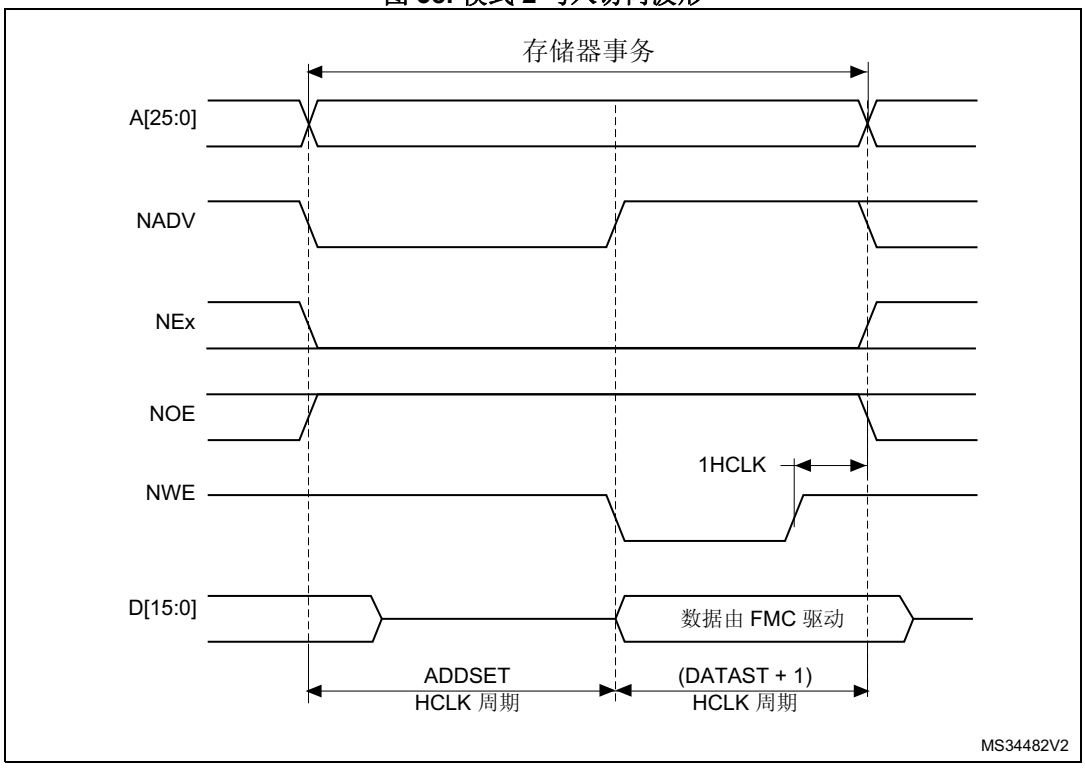
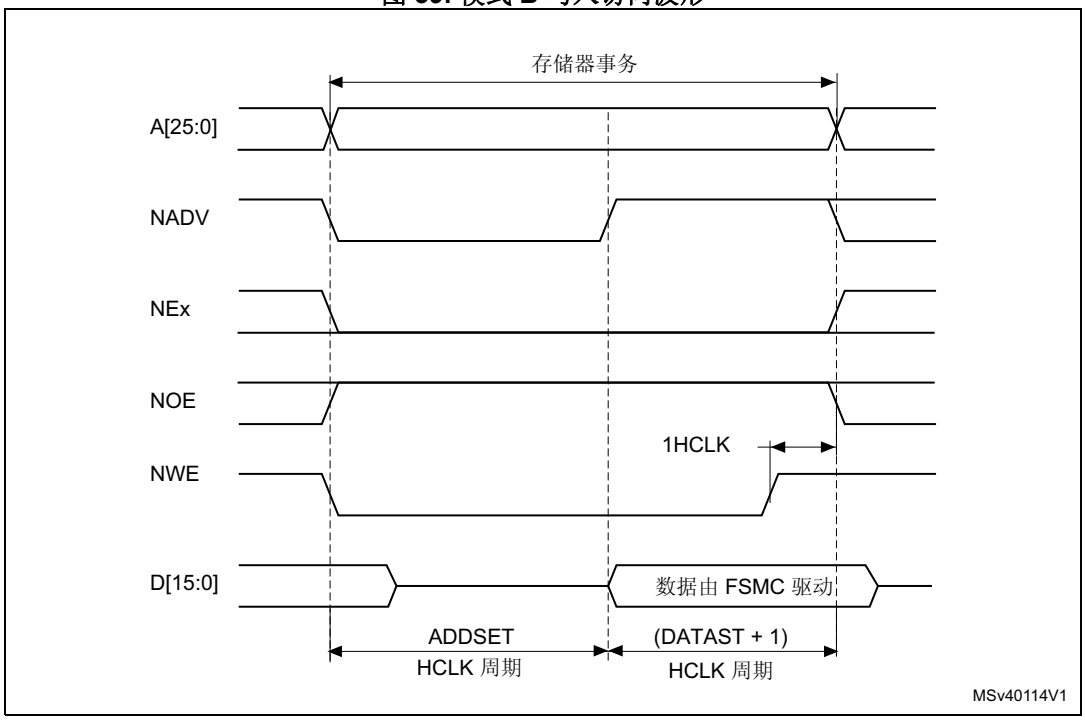


图 39. 模式 B 写入访问波形



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 55. FSMC_BCRx 位域

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 56. FSMC_BTRx 位域

位号	位名	要设置的值
31:30	保留	0x0
29:28	ACCMOD	如果设置了扩展模式，则为 0x1
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 57. FSMC_BWTRx 位域

位号	位名	要设置的值
31:30	保留	0x0
29:28	ACCMOD	如果设置了扩展模式，则为 0x1
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，FSMC_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 40. 模式 C 读取访问波形

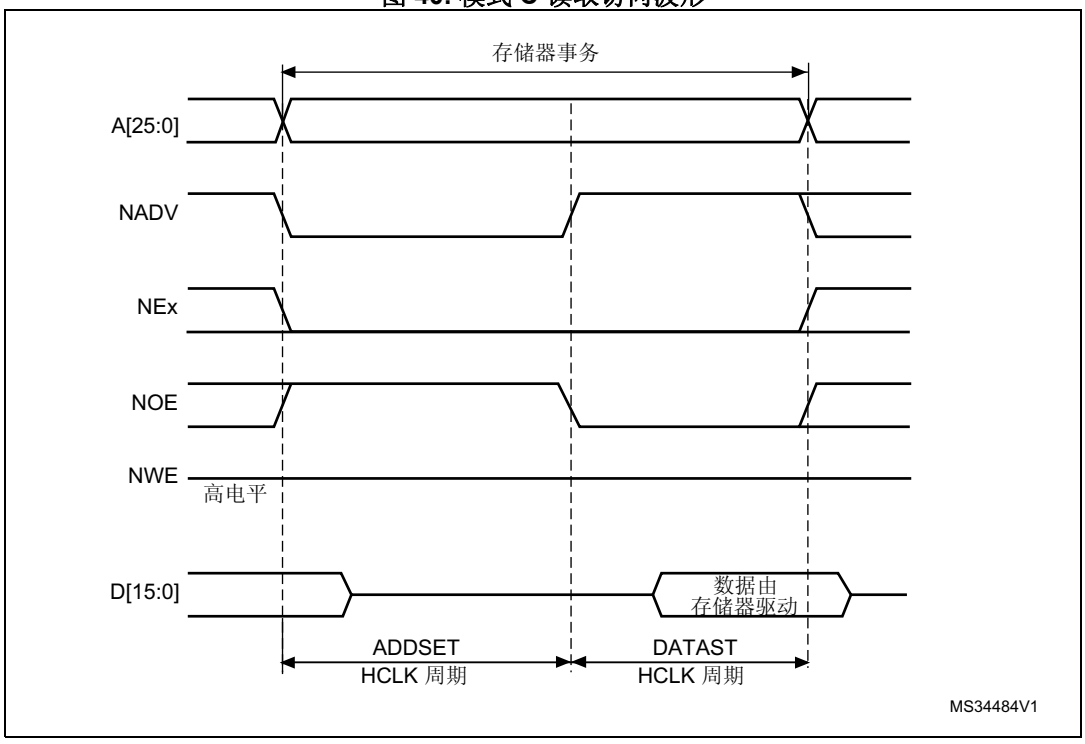
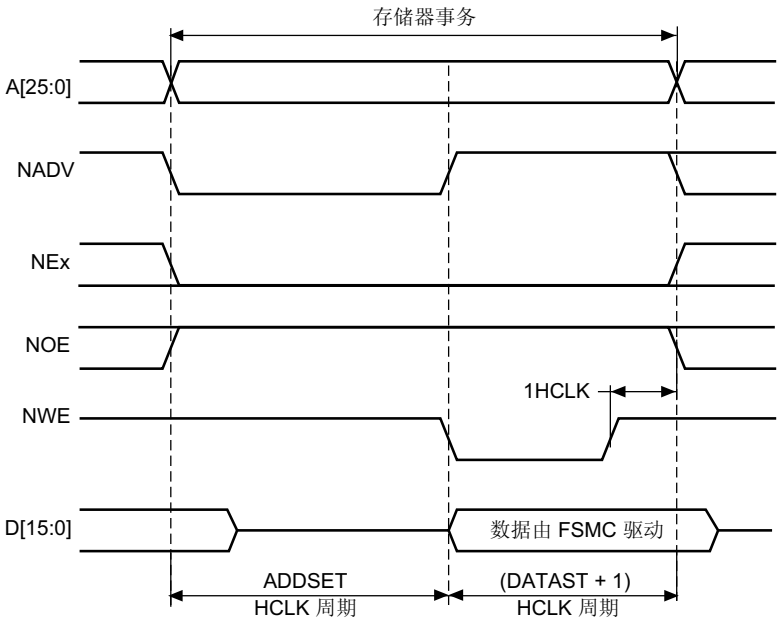


图 41. 模式 C 写入访问波形



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 58. FSMC_BCRx 位域

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
5:4	MWID	根据需要进行设置

表 58. FSMC_BCRx 位域（续）

位号	位名	要设置的值
3:2	MTYP	0x02 (NOR Flash)
1	MUXEN	0x0
0	MBKEN	0x1

表 59. FSMC_BTRx 位域

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	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 60. FSMC_BWTRx 位域

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 42. 模式 D 读取访问波形

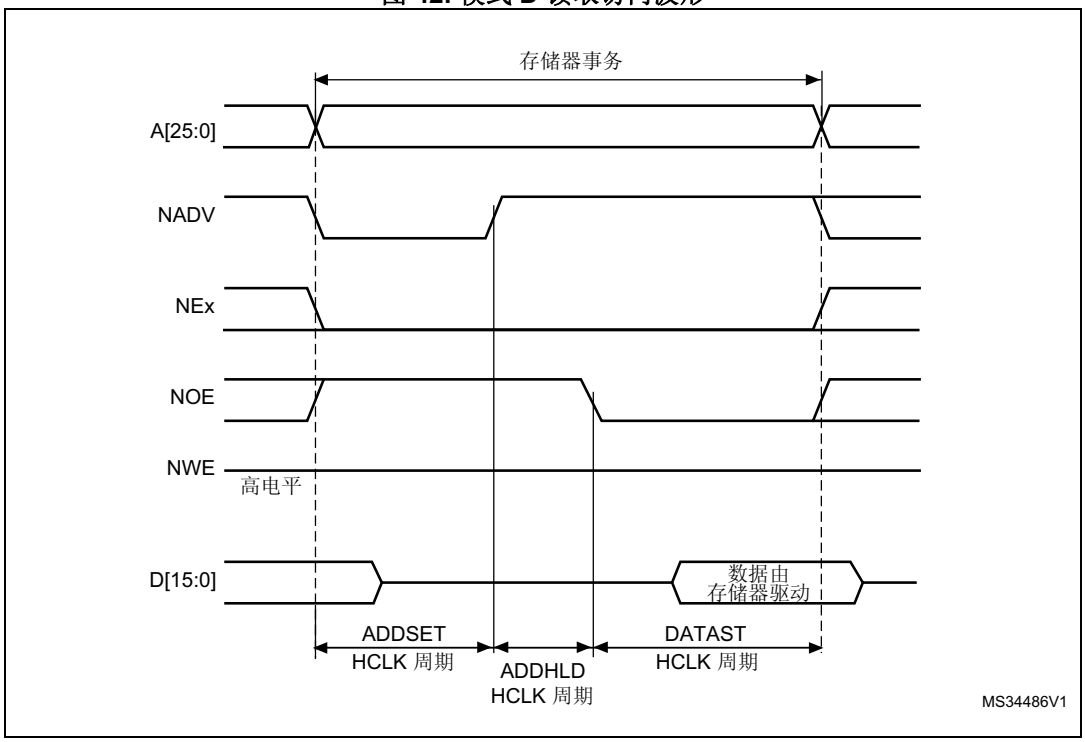
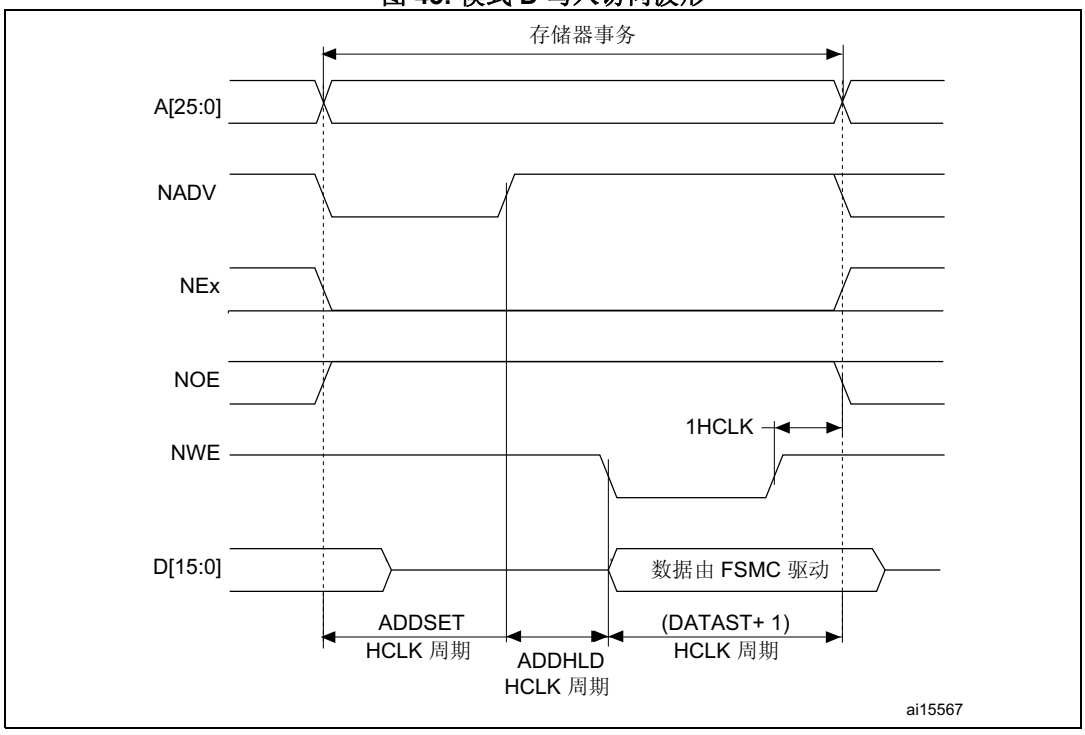


图 43. 模式 D 写入访问波形



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 61. FSMC_BCRx 位域

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 62. FSMC_BTRx 位域

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	读取访问第二个阶段的持续时间（DATAST 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	读取访问第一个阶段的持续时间（ADDSET 个 HCLK 周期）。 ADDSET 的最小值为 1。

表 63. FSMC_BWTRx 位域

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	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 44. 复用读取访问波形

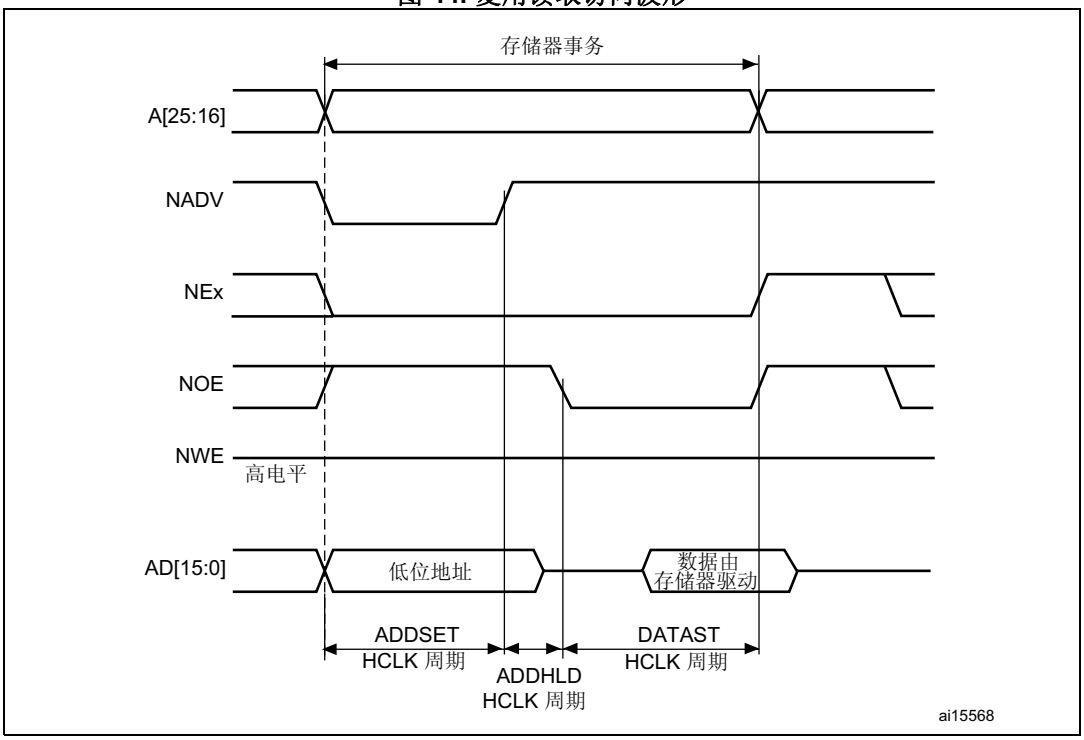
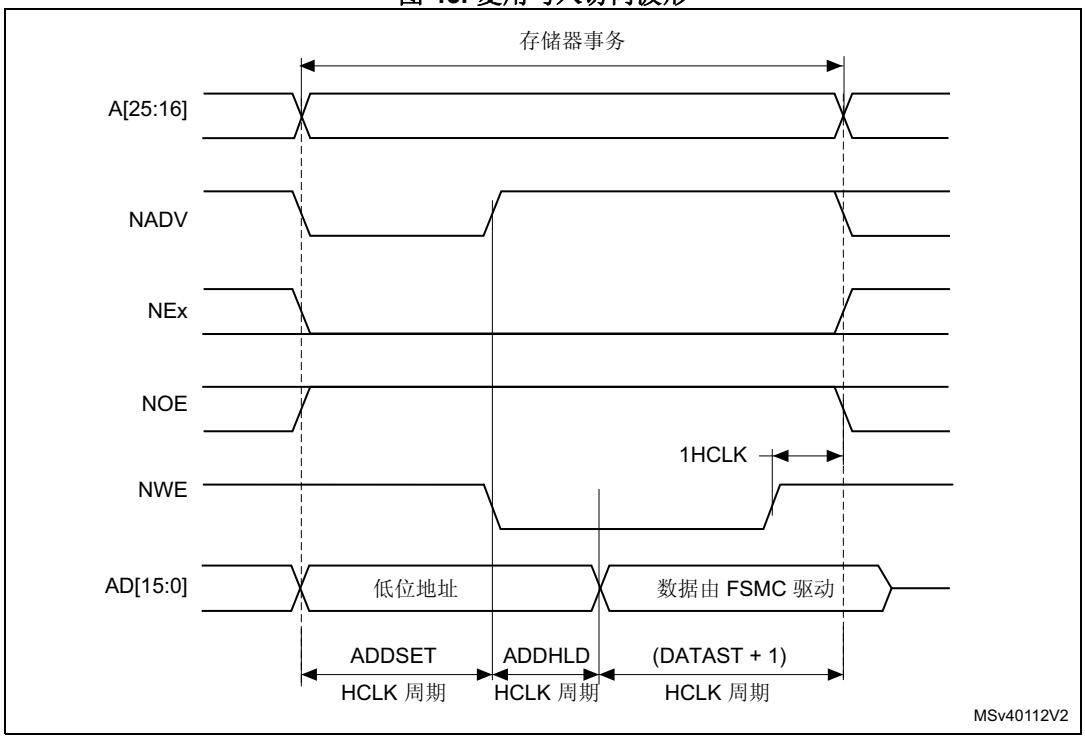


图 45. 复用写入访问波形



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 64. FSMC_BCRx 位域

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
5:4	MWID	根据需要进行设置

表 64. FSMC_BCRx 位域（续）

位号	位名	要设置的值
3:2	MTYP	0x2 (NOR Flash) 或 0x1 (PSRAM)
1	MUXEN	0x1
0	MBKEN	0x1

表 65. FSMC_BTRx 位域

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 FSMC_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：

$$DATAST \geq (4 \times HCLK) + (max\_wait\_assertion\_time - address\_phase - hold\_phase)$$

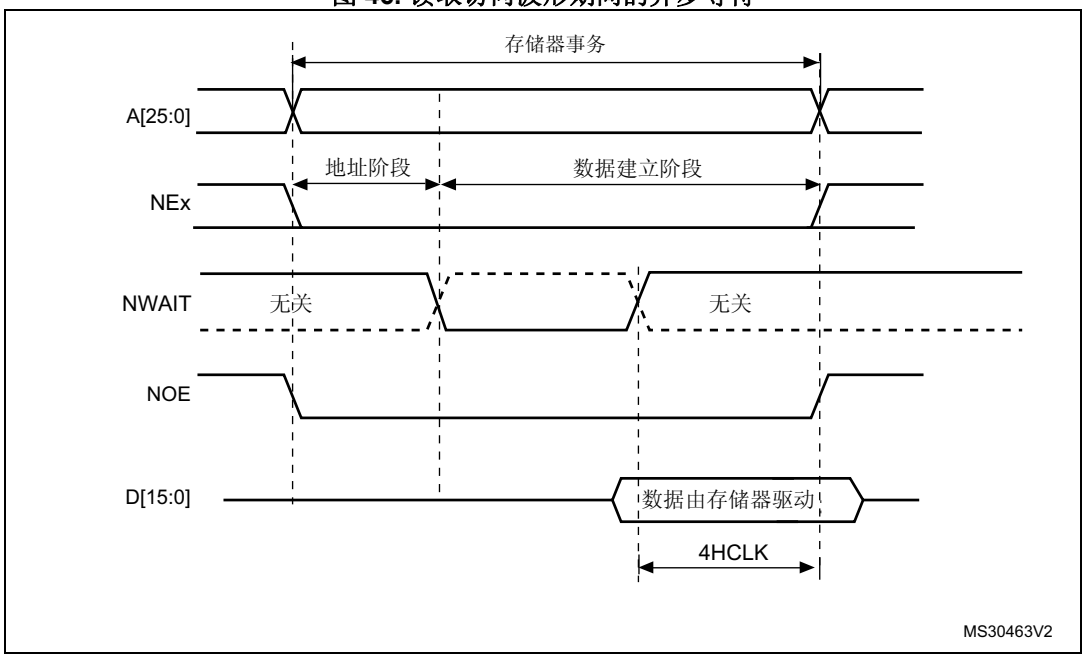
   否则

$$DATAST \geq 4 \times HCLK$$

   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。

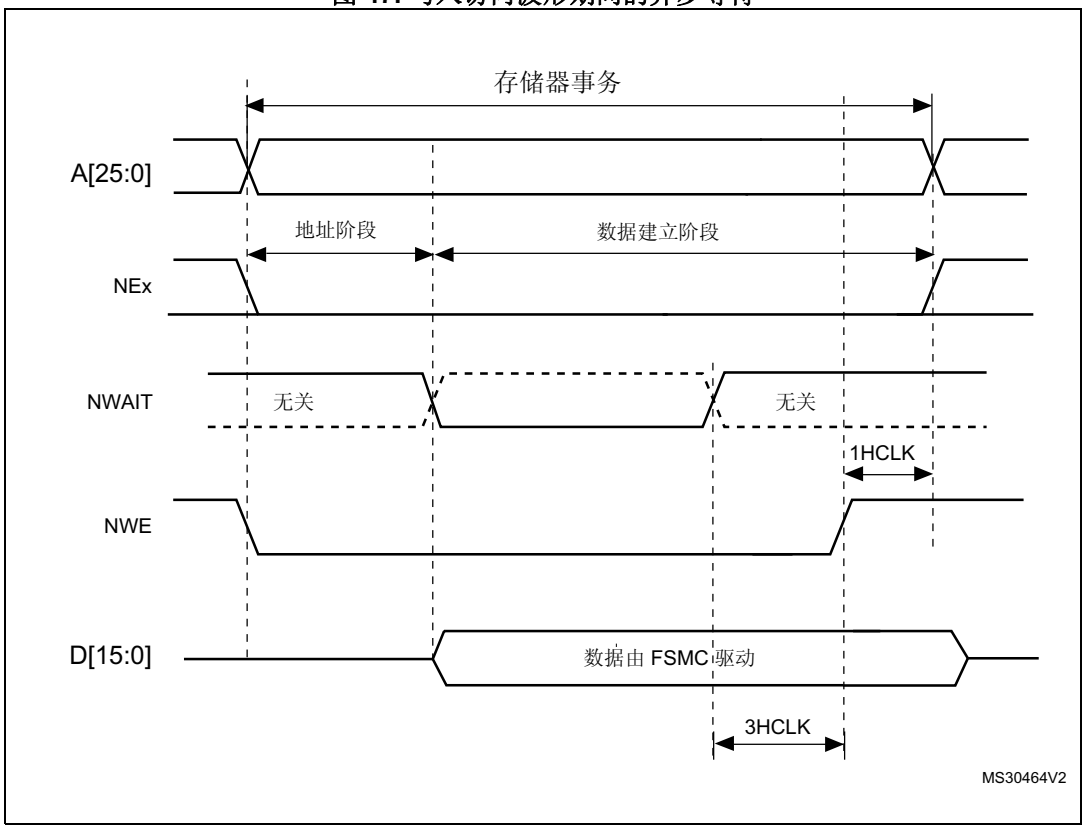
图 46 和*图 47* 显示了异步存储器释放 WAIT 之后，在存储器访问阶段增加的 HCLK 时钟周期的个数（与上述情况无关）。

图 46. 读取访问波形期间的异步等待



1. NWAIT 极性取决于 FSMC_BCRx 寄存器中的 WAITPOL 位设置。

图 47. 写入访问波形期间的异步等待



1. NWAIT 极性取决于 FSMC_BCRx 寄存器中的 WAITPOL 位设置。

11.5.5 同步事务

存储器时钟 FSMC_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：

$$\text{FSMC\_CLK分频比} = \max(\text{CLKDIV} + 1, \text{MWID(AHB数据大小)})$$

无论 MWID 是 16 位还是 8 位，FSMC_CLK 分频比始终由 CLKDIV 的设定值定义。

示例：

- CLKDIV = 1，MWID = 16 位，AHB 数据大小 = 8 位时，FSMC_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，FSMC 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，FSMC 不会计入时钟周期。

注意： 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 FSMC 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，FSMC 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，FSMC 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 FSMC 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 FSMC 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 CellularRAM™ 1.5 的边界页

CellularRAM™ 1.5 不允许突发访问越过页边界。根据存储器页大小配置 FSMC_BCR1 寄存器中的 CPSIZE 位来达到存储器页大小时，FSMC 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

FSMC 支持这两种 NOR Flash 等待周期配置，通过 FSMC_BCRx 寄存器的 WAITCFG 位 (x = 0..3) 针对每个片选进行配置。

图 48. 等待配置波形

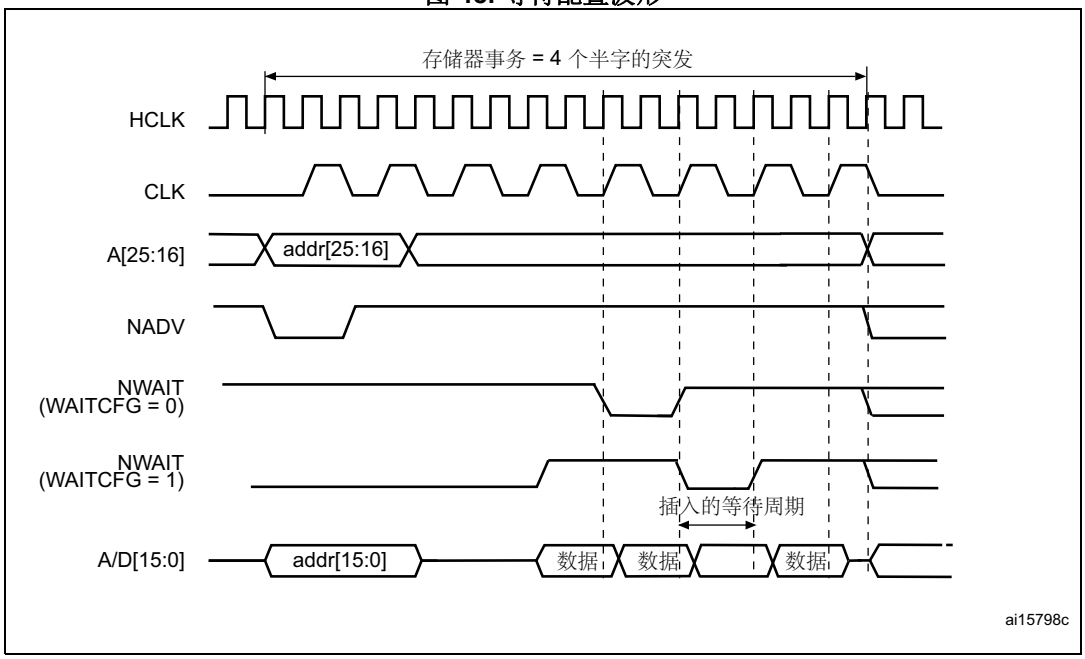
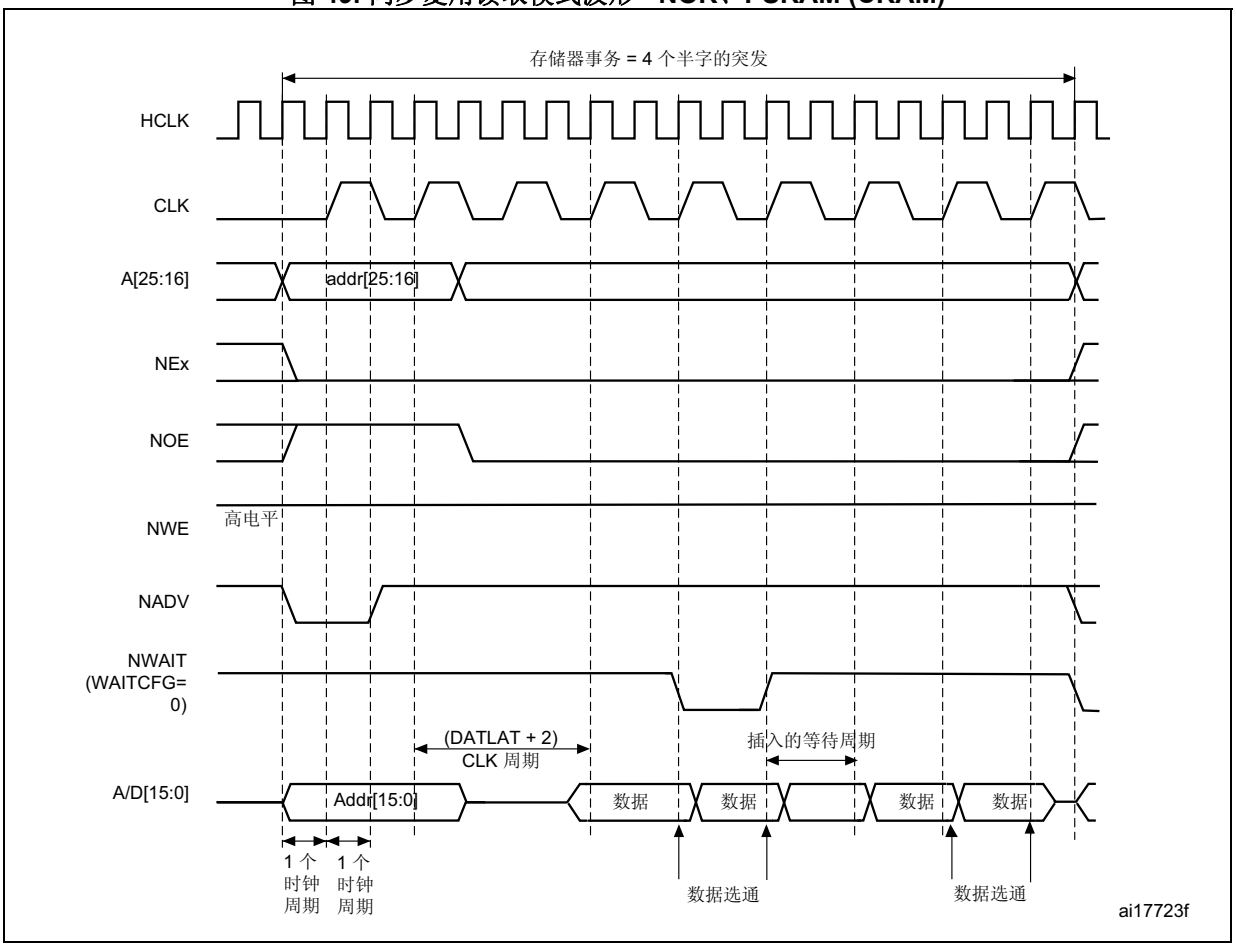

存储器事务 = 4 个半字的突发
HCLK
CLK
A[25:16]
addr[25:16]
NADV
NWAIT
(WAITCFG = 0)
NWAIT
(WAITCFG = 1)
插入的等待周期
A/D[15:0]
addr[15:0]
数据
数据
数据
ai15798c

图 49. 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 66. FSMC_BCRx 位域

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	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 1，否则保持为 0
12	WREN	对同步读取没有影响
11	WAITCFG	是否置 1 视存储器情况而定
10	保留	0x0

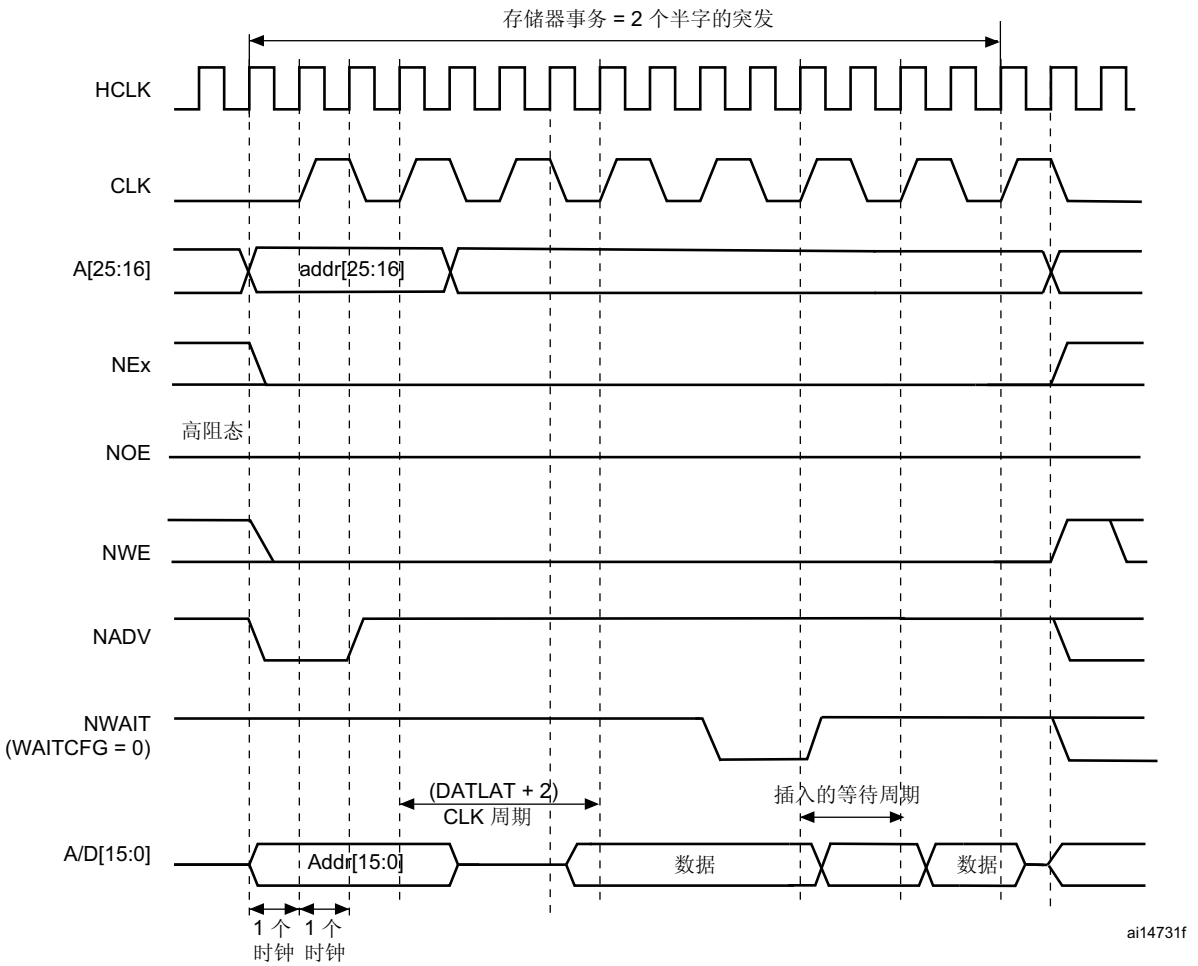
表 66. FSMC_BCRx 位域（续）

位号	位名	要设置的值
9	WAITPOL	是否置 1 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 1 (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 67. FSMC_BTRx 位域

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 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 50. 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 68. FSMC_BCRx 位域

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	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 1，否则保持为 0

表 68. FSMC_BCRx 位域（续）

位号	位名	要设置的值
12	WREN	0x1
11	WAITCFG	0x0
10	保留	0x0
9	WAITPOL	是否置 1 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 69. FSMC_BTRx 位域

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 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

11.5.6 NOR/PSRAM 控制寄存器

存储区域 x 的 SRAM/NOR-Flash 片选控制寄存器 (FSMC_BCRx)（x = 1 到 4）

SRAM/NOR-Flash chip-select control register for bank x

偏移地址：8 * (x – 1)，（x = 1 到 4）

复位值：存储区域 1：0x0000 30DB

复位值：存储区域 2：0x0000 30D2

复位值：存储区域 3：0x0000 30D2

复位值：存储区域 4：0x0000 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[1:0]		MTYP[1:0]		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 FSMC 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： FSMC_BCR2..4 寄存器的 WFDIS 位为“无关”位。只能通过 FSMC_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 FSMC_CLK 时钟。

0：仅在同步存储器访问（读/写事务）期间生成 FSMC_CLK。FSMC_CLK 时钟分频比由 FSMC_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 FSMC_CLK。CCLKEN 置 1 将激活 FSMC_CLK 时钟。

注： FSMC_BCR2..4 寄存器的 CCLKEN 位为“无关”位。只能通过 FSMC_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 FSMC_CLK 连续时钟。

注： 如果 CCLKEN 位置 1，则 FSMC_CLK 时钟分频比将由 FSMC_BTR1 寄存器的 CLKDIV 值指定。FSMC_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 之外的其他存储区域相连的同步存储器将共用同一个时钟源（FSMC_BTR2..4 寄存器和 FSMC_BWTR2..4 寄存器中的 CLKDIV 值对其他存储区域不起作用。）

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 FSMC_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 CellularRAM™ 1.5，CellularRAM™ 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，FSMC 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

其他：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 FSMC 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)。

FSMC 可对 FSMC_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 FSMC_BWTR 寄存器中的值（复位后的默认值）

1：考虑 FSMC_BWTR 寄存器中的值

注： 如果禁用扩展模式，FSMC 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FSMC：

0：FSMC 禁止在存储区域内写入，如果进行写操作将报告 AHB 错误，

1：FSMC 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：保留

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（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，其他存储区域均禁止。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

存储区域 x 的 SRAM/NOR-Flash 片选时序寄存器 (FSMC_BTRx)

SRAM/NOR-Flash chip-select timing register for bank x

偏移地址：0x04 + 8 * (x – 1)，（x = 1 到 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 FSMC_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（FSMC_BWTRx 寄存器）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	ACCMOD[1:0]		DATLAT[3:0]				CLKDIV[3:0]				BUSTURN[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
DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
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 29:28 **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 FSMC_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]**：（请参见位说明下的注释）同步存储器的数据延迟 (Data latency for synchronous memory)

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 FSMC_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**：FMC_CLK 信号的时钟分频比 (Clock divide ratio (for FSMC_CLK signal))

定义 FSMC_CLK 时钟输出信号的周期，以 HCLK 周期数表示：

0000：FSMC_CLK 周期 = 1× HCLK 周期

0001：FSMC_CLK 周期 = 2 × HCLK 周期

0010：FSMC_CLK 周期 = 3 × HCLK 周期

1111：FSMC_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 FSMC_CLK 分频比公式的信息，请参见第 11.5.5 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。该延迟可以匹配连续事务之间的最短时间（tEHEL 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 (tEHQZ)。编程的总线周转延迟插入到对静态存储区域的异步读取（复用模式或 D 模式）/写入事务与任何其他异步/同步读取/写入事务之间。如果执行读取操作，则存储区域可以相同也可以不同；如果执行写入操作，则除了复用模式或 D 模式之外，存储区域可以不同。

在某些情况下，无论编程的 BUSTURN 值是多少，总线周转延迟都固定

如下：

- 除了复用模式和 D 模式之外，总线周转延迟不会插入到对同一静态存储区域的两个连续异步写入传输之间。
- 下列情况下的总线周转延迟为 1 个 HCLK 时钟周期：
  - 对同一静态存储区域的两次连续的异步读取操作之间（复用模式和 D 模式除外）。
  - 对任何静态存储区域/动态存储区域的异步读取操作与异步/同步写入操作之间（复用模式和 D 模式除外）。
  - 异步（模式1、2、A、B 或 C）读取以及从另一个静态存储区域读取。
- 下列情况下的总线周转延迟为 2 个 HCLK 时钟周期：
  - 对同一存储区域的两次连续的同步写入操作之间（在突发或单次模式下）。
  - 对静态存储区域的同步写入（突发或单次）访问与异步写入/读取传输之间（在读取操作的情况下，存储区域可以相同也可以不同）。
  - 对另一个静态存储区域的任何同步/异步读取或写入操作之后的两次连续的同步读取操作之间（在突发或单次模式下）。
- 下列情况下的总线周转延迟为 3 个 HCLK 时钟周期：
  - 对同一存储区域的两次连续的同步写入操作之间（在突发或单次模式下）。
  - 对同一或不同存储区域的同步写入（在突发或单次模式下）访问与同步读取访问之间。

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 x 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 33* 到*图 45*），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 33* 到*图 45*）。

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 33* 到*图 45*），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 33* 到*图 45*）。

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*地址设置*阶段的持续时间（请参见*图 33* 到*图 45*），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建立阶段持续时间的信息，请参见相应图片（*图 33* 到*图 45*）。

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。

在模式 1 和 PSRAM 存储器下，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 FSMC 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 (FSMC_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 FSMC_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[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUSTURN[3:0]			
		rw	rw									rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAST[7:0]								ADDHLD[3:0]				ADDSET[3:0]			
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 29:28 **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 FSMC_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)

编程的总线周转延迟插入到对静态存储区域的异步写入传输与任何其他异步/同步读取或写入传输之间。如果执行读取操作，则存储区域可以相同也可以不同；如果执行写入操作，则除了复用模式或 D 模式之外，存储区域可以不同。

在某些情况下，无论编程的 BUSTURN 值是多少，总线周转延迟都固定如下：

- 除了复用模式和 D 模式之外，总线周转延迟不会插入到对同一静态存储区域的两个连续异步写入传输之间。
- 下列情况下的总线周转延迟为 2 个 HCLK 时钟周期：
  - 对同一存储区域的两次连续的同步写入操作之间（在突发或单次模式下）。
  - 对静态存储区域的同步写入（在突发或单次模式下）传输与异步写入/读取传输之间。
- 下列情况下的总线周转延迟为 3 个 HCLK 时钟周期：
  - 对同一存储区域的两次连续的同步写入操作之间（在突发或单次模式下）。
  - 对同一或不同存储区域的同步写入（在突发或单次模式下）传输与同步读取传输之间。

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 33* 到*图 45*），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 42* 到*图 45*），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 33* 到*图 45*），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。

11.6 FSMC 寄存器映射

表 70. FSMC 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	**FSMC_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	**FSMC_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	**FSMC_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	**FSMC_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	**FSMC_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	**FSMC_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	**FSMC_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

表 70. FSMC 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
0x1C	**FSMC_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	**FSMC_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
0x10C	**FSMC_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	**FSMC_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	**FSMC_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

有关寄存器边界地址的信息，请参见*第 54 页的第 2.2.2 节*。

12 Quad-SPI 接口 (QUADSPI)

12.1 简介

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 Flash 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。

12.2 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 Flash 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

12.3 QUADSPI 功能说明

12.3.1 QUADSPI 框图

图 51. QUADSPI 功能框图（双 Flash 模式禁止）

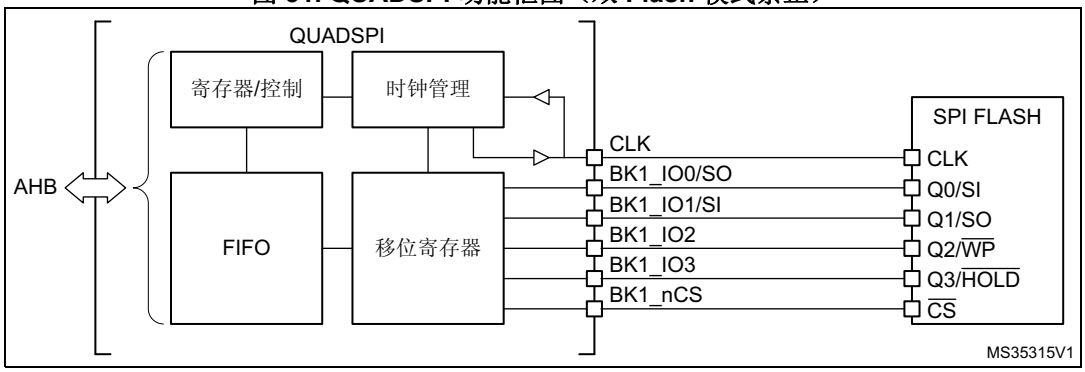
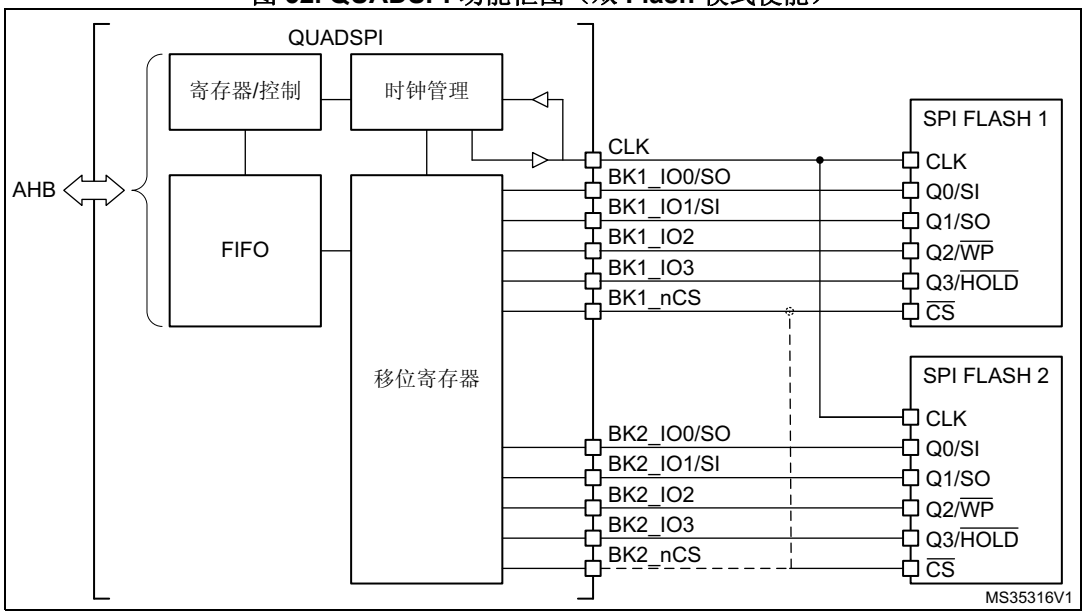


图 52. QUADSPI 功能框图（双 Flash 模式使能）



12.3.2 QUADSPI 引脚

表 71 列出了 QUADSPI 引脚，QUADSPI 可使用 6 个引脚连接单个 FLASH，或者在双 Flash 模式下使用 10 到 11 个引脚连接两个 Flash（FLASH 1 和 FLASH 2）。

表 71. QUADSPI 引脚

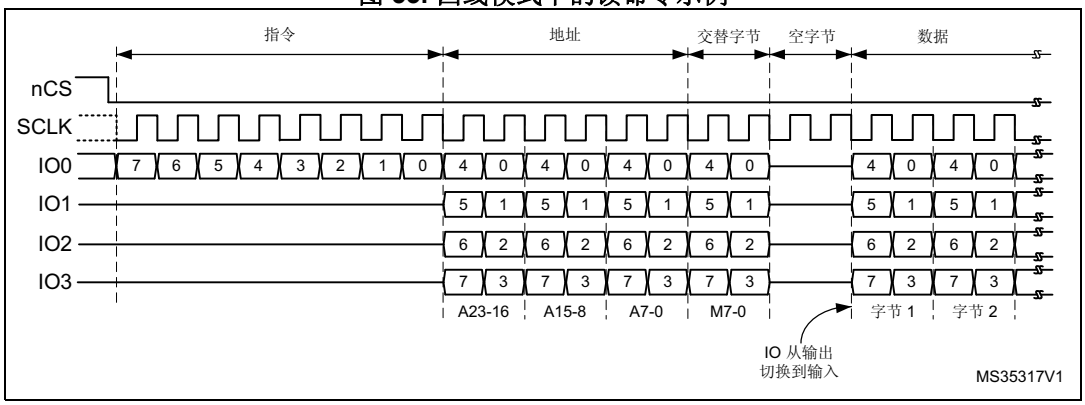
信号名称	信号类型	说明
CLK	数字输出	FLASH 1 和 FLASH 2 的时钟
BK1_IO0/SO	数字输入/输出	在双线/四线模式中为双向 IO，单线模式中为串行输出，适用于 FLASH 1
BK1_IO1/SI	数字输入/输出	在双线/四线模式中为双向 IO，单线模式中为串行输入，适用于 FLASH 1
BK1_IO2	数字输入/输出	在四线模式中为双向 IO，适用于 FLASH 1
BK1_IO3	数字输入/输出	在四线模式中为双向 IO，适用于 FLASH 1
BK2_IO0/SO	数字输入/输出	在双线/四线模式中为双向 IO，单线模式中为串行输出，适用于 FLASH 2
BK2_IO1/SI	数字输入/输出	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	数字输入/输出	在四线模式中为双向 IO，适用于 FLASH 2
BK2_IO3	数字输入/输出	在四线模式中为双向 IO，适用于 FLASH 2
BK1_nCS	数字输出	片选（低电平有效），适用于 FLASH 1。如果 QUADSPI 始终在双 Flash 模式下工作，则其也可用于 FLASH 2。
BK2_nCS	数字输出	片选（低电平有效），适用于 FLASH 2。如果 QUADSPI 始终在双 Flash 模式下工作，则其也可用于 FLASH 1。

12.3.3 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 53. 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] 位域进行配置。

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] 位域中指定。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] 位域进行配置。

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。

12.3.4 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

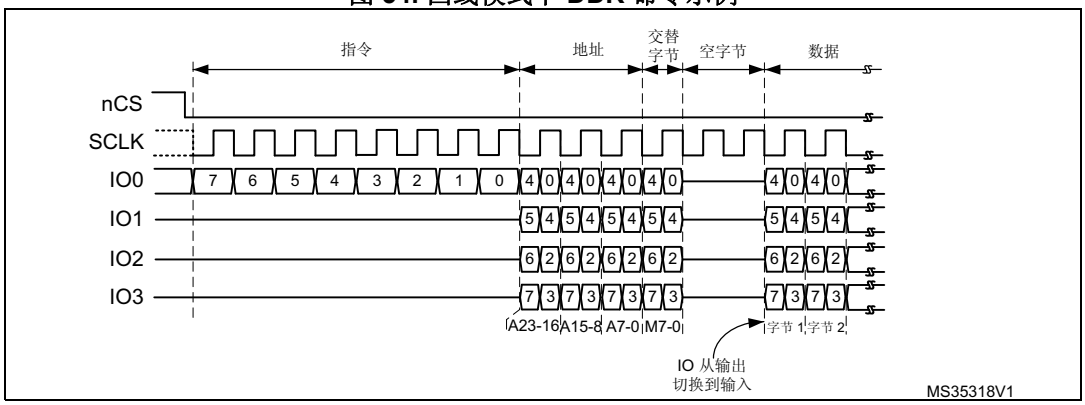
若 DDRM 位 (QUADSPI_CCR[31]) 置 1，则 QUADSPI 在双倍数据速率 (DDR) 模式下工作。

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 的位 4）。因此，在半个 CLK 周期后（下一个反向边沿）对信号采样。

图 54. 四线模式下 DDR 命令示例



双 Flash 模式

若 DFM 位（QUADSPI_CR 的位 6）为 1，QUADSPI 处于双 Flash 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 位数据（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 Flash 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 Flash 模式下读取 Flash 状态寄存器时，需要读取的字节数是单 Flash 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 Flash 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 Flash 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 Flash 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。

12.3.5 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 标志保持为“0”。

12.3.6 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 的位 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。

12.3.7 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 256 MB 范围内，则生成总线错误。此错误的影响具体取决于尝试进行访问的总线主设备：

- 如果为 Cortex® CPU，则会生成总线故障异常（使能总线故障时）或硬性故障异常（禁用总线故障时）
- 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 的位 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。

12.3.8 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）。

12.3.9 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 的位 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。

12.3.10 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 Flash 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）。

12.3.11 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。

12.3.12 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。

12.3.13 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，当主器件执行的访问超出范围或 QUADSPI 被禁止时：将产生总线错误，以响应故障总线主请求。
- 当主机访问存储器映射空间，但内存映射模式被禁止时：将产生总线错误，以响应故障总线主请求。

12.3.14 QUADSPI 的 busy 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。

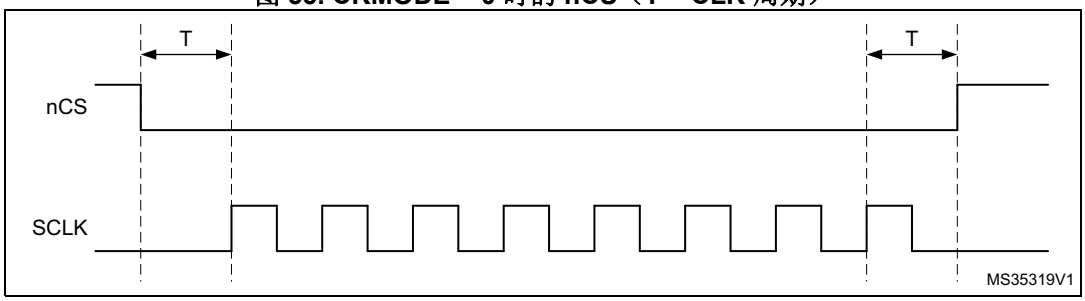
注： *如果中止对状态寄存器的写入操作，有些 Flash 可能发生错误行为。*

12.3.15 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。

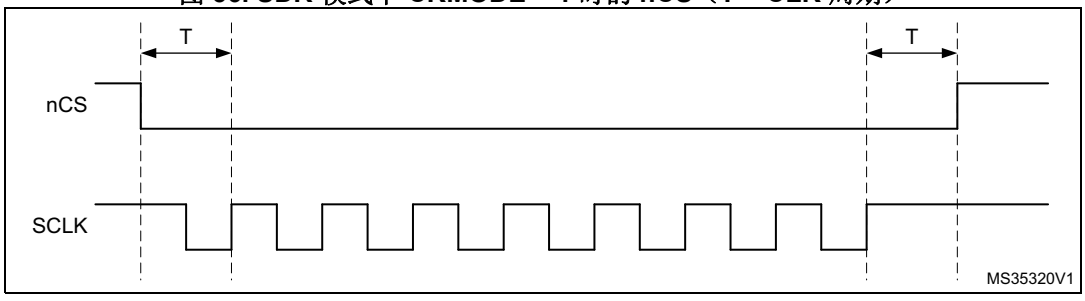
当 CKMODE = 0（“模式 0”，在未进行任何操作时 CLK 保持低电平）时，nCS 在操作首次升高 CLK 边沿时的一个 CLK 周期前降至低电平，在操作最后一次升高 CLK 边沿时的一个 CLK 周期后升至高电平，如*表 55* 所示。

图 55. CKMODE = 0 时的 nCS（T = CLK 周期）



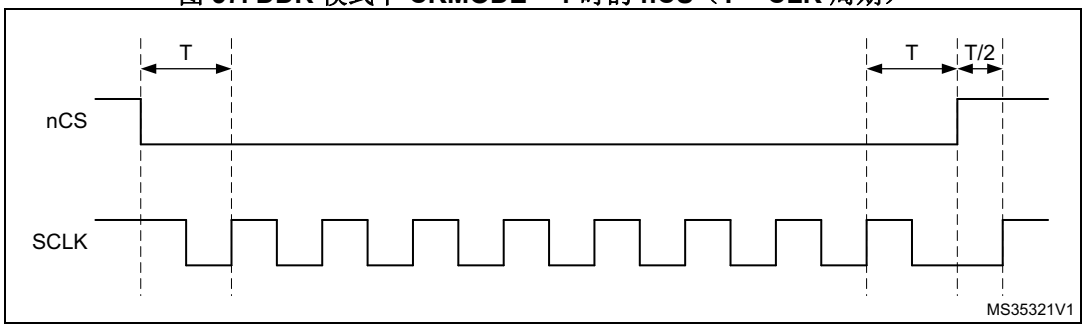
当 CKMODE = 1（“模式 3”，在未进行任何操作时 CLK 升至高电平）且 DDRM = 0（SDR 模式）时，nCS 仍在操作首次升高 CLK 边沿时的一个 CLK 周期前降至低电平，在操作最后一次升高 CLK 边沿时的一个 CLK 周期后升至高电平，如*表 56* 所示。

图 56. SDR 模式下 CKMODE = 1 时的 nCS（T = CLK 周期）



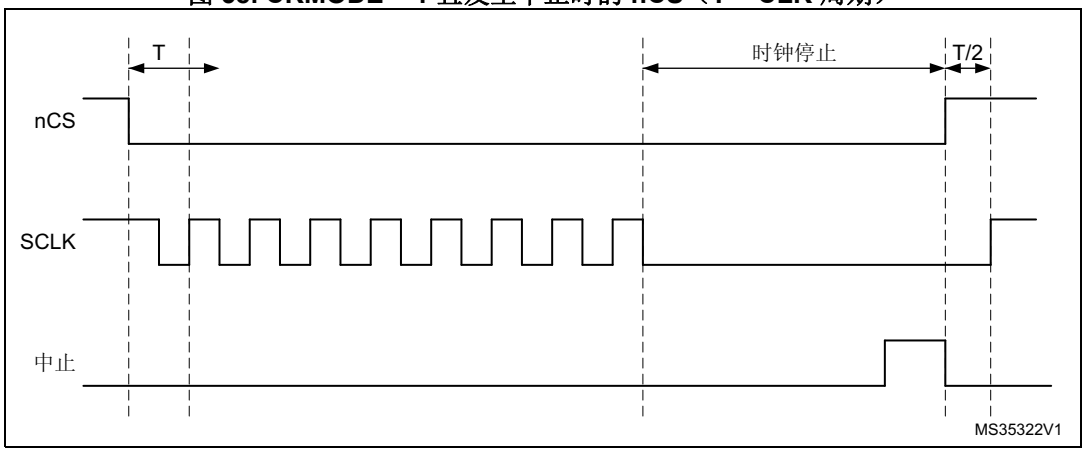
当 CKMODE = 1（模式 3）且 DDRM = 1（DDR 模式）时，nCS 在操作首次升高 CLK 边沿时的一个 CLK 周期前降至低电平，在操作最后一次升高 CLK 边沿时的一个 CLK 周期后升至高电平，如*表 57* 所示。由于 DDR 操作必须伴随下降沿完成，当 nCS 变为高电平时，CLK 为低电平并在经过半个周期后恢复高电平。

图 57. 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*表 58* 所示。

图 58. CKMODE = 1 且发生中止时的 nCS（T = CLK 周期）



不处于双 Flash 模式 (DFM = 0) 时，仅访问 FLASH 1，因此 BK2_nCS 保持高电平。在双 Flash 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 Flash 模式，则 FLASH 2 可使用 BK1_nCS，而 BK2_nCS 引脚输出可用于其他功能。

12.4 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 72. 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

12.5 QUADSPI 寄存器

12.5.1 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[7:0]								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[4:0]					FSEL	DFM	Res.	SSHIFT	TCEN	DMAEN	ABORT	EN
			rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw

位 31:24 **PRESCALER[7:0]**：时钟预分频器 (Clock prescaler)

该位域定义基于 AHB 时钟生成 CLK 所用的分频系数（值 + 1）。

0：$F_{CLK} = F_{AHB}$，AHB 时钟直接用作 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%。时钟信号的低电平持续时间比高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 Flash 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 Flash 模式 (Dual-flash mode)
该位激活双 Flash 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 Flash 模式
1：使能双 Flash 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 **TCEN**：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 的位 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

12.5.2 QUADSPI 器件配置寄存器 (QUADSPI_DCR)

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

12.5.3 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。

12.5.4 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
											w	w		w	w

位 31:5 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

12.5.5 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 Flash 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。

12.5.6 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[1:0]		Res.	DCYC[4:0]					ABSIZE[1:0]	
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
ABMODE[1:0]		ADSIZE[1:0]		ADMODE[1:0]		IMODE[1:0]		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 (Send instruction only once mode)
请参见*第 299 页的第 12.3.12 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 **ADMODE[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。

12.5.7 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 Flash 模式下，由于地址始终为偶地址，ADDRESS[0] 自动保持为“0”

12.5.8 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。

12.5.9 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]。

12.5.10 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。

12.5.11 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。

12.5.12 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。

12.5.13 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。

12.5.14 QUADSPI 寄存器映射

表 73. QUADSPI 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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 54 页的第 2.2.2 节*。

13 模数转换器 (ADC)

13.1 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 有模拟看门狗功能，可在应用中用于检测输入电压是否超过了用户设定的阈值上限或下限。

13.2 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/三重交替模式下可配置的转换间延迟
- 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 59 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}。*

13.3 ADC 功能说明

图 59 给出了单个 ADC 的框图，*表 74* 列出了 ADC 的引脚说明。

图 59. 单个 ADC 框图

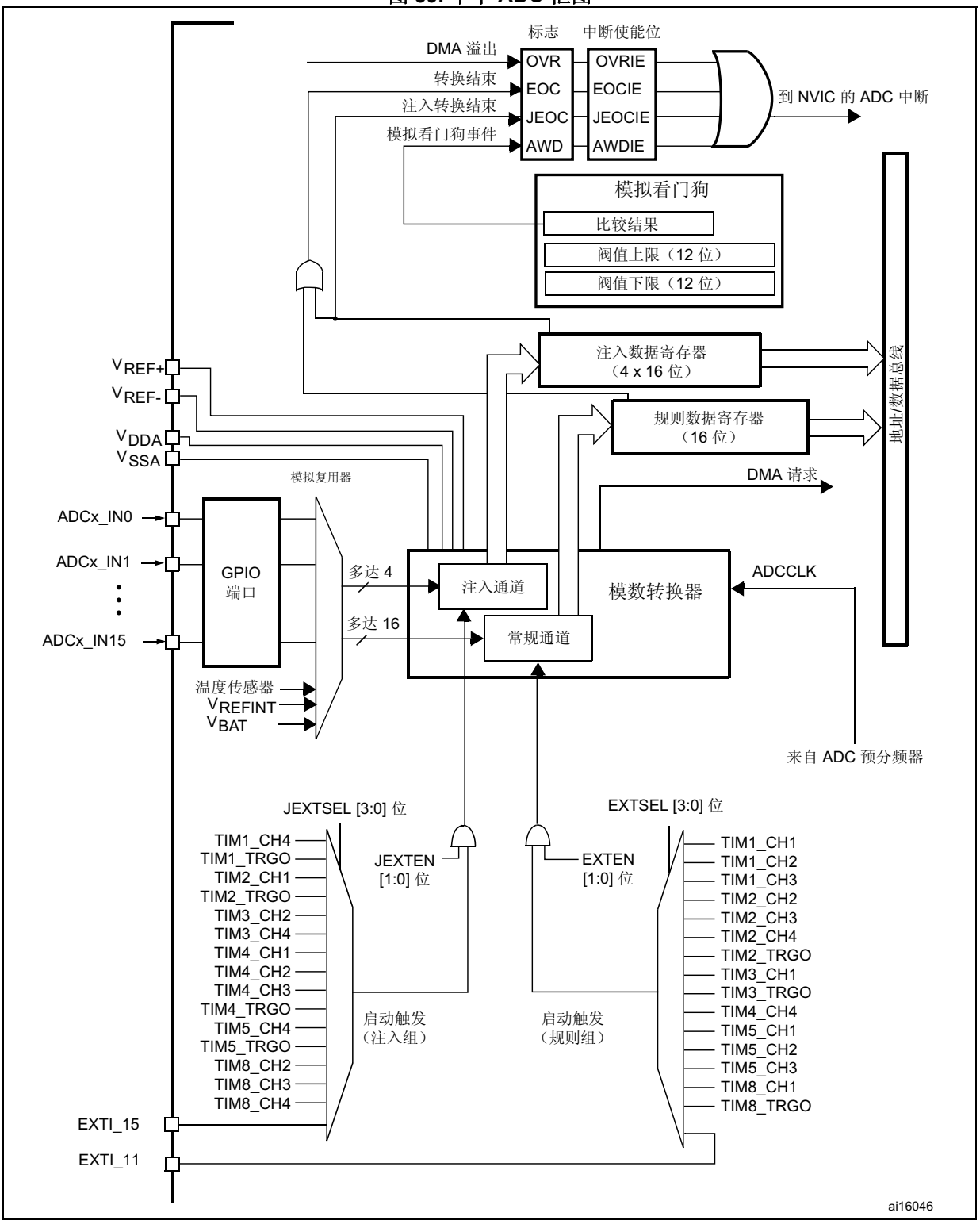

标志
中断使能位
DMA 溢出
转换结束
注入转换结束
模拟看门狗事件
OVR
EOC
JEOC
AWD
OVRIE
EOCIE
JEOCIE
AWDIE
到 NVIC 的 ADC 中断
模拟看门狗
比较结果
阈值上限（12 位）
阈值下限（12 位）
注入数据寄存器
（4 x 16 位）
规则数据寄存器
（16 位）
地址/数据总线
DMA 请求
VREF+
VREF-
VDDA
VSSA
模拟复用器
ADCx_IN0
ADCx_IN1
ADCx_IN15
GPIO
端口
多达 4
多达 16
注入通道
常规通道
模数转换器
ADCCLK
温度传感器
VREFINT
VBAT
来自 ADC 预分频器
JEXTSEL [3:0] 位
EXTSEL [3:0] 位
TIM1_CH4
TIM1_TRGO
TIM2_CH1
TIM2_TRGO
TIM3_CH2
TIM3_CH4
TIM4_CH1
TIM4_CH2
TIM4_CH3
TIM4_TRGO
TIM5_CH4
TIM5_TRGO
TIM8_CH2
TIM8_CH3
TIM8_CH4
JEXTEN
[1:0] 位
EXTEN
[1:0] 位
TIM1_CH1
TIM1_CH2
TIM1_CH3
TIM2_CH2
TIM2_CH3
TIM2_CH4
TIM2_TRGO
TIM3_CH1
TIM3_TRGO
TIM4_CH4
TIM5_CH1
TIM5_CH2
TIM5_CH3
TIM8_CH1
TIM8_TRGO
启动触发
（注入组）
启动触发
（规则组）
EXTI_15
EXTI_11
ai16046

表 74. 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

13.3.1 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）。

13.3.2 ADC 时钟

ADC 具有两个时钟方案：

- 用于模拟电路的时钟：ADCCLK

  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。

13.3.3 通道选择

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。

13.3.4 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。

13.3.5 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：

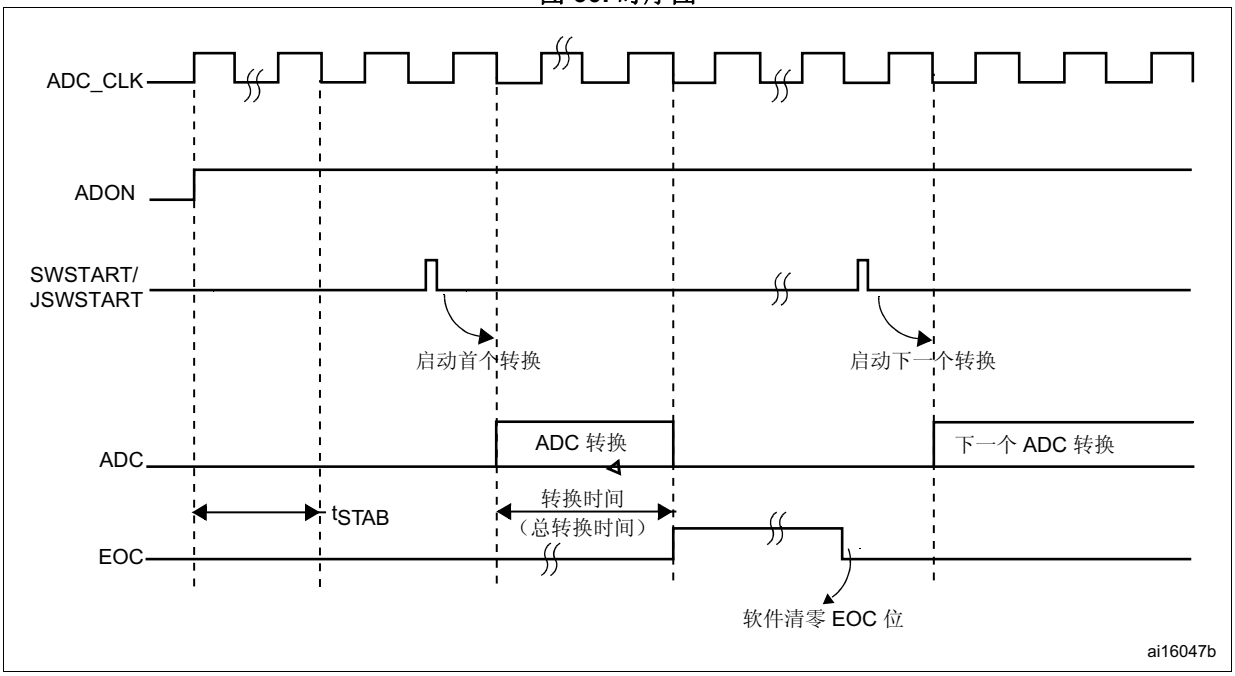
- 如果转换了常规通道组：
  - 上次转换的数据存储在 16 位 ADC_DR 寄存器中
  - EOC（转换结束）标志置 1
  - EOCIE 位置 1 时将产生中断

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。*

13.3.6 时序图

如*图 60* 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 60. 时序图



13.3.7 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 75 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 61. 模拟看门狗的保护区域

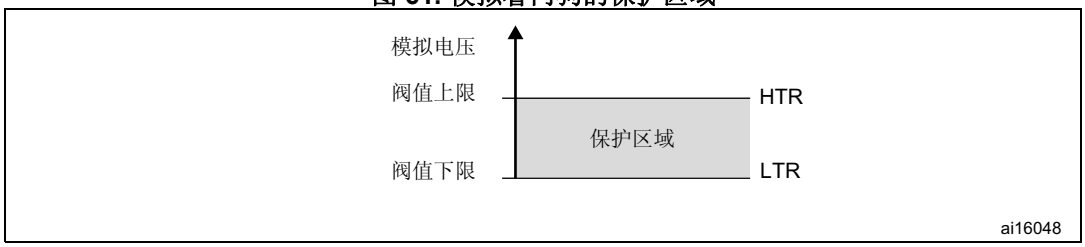


表 75. 模拟看门狗通道选择

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

13.3.8 扫描模式

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。

13.3.9 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 62* 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

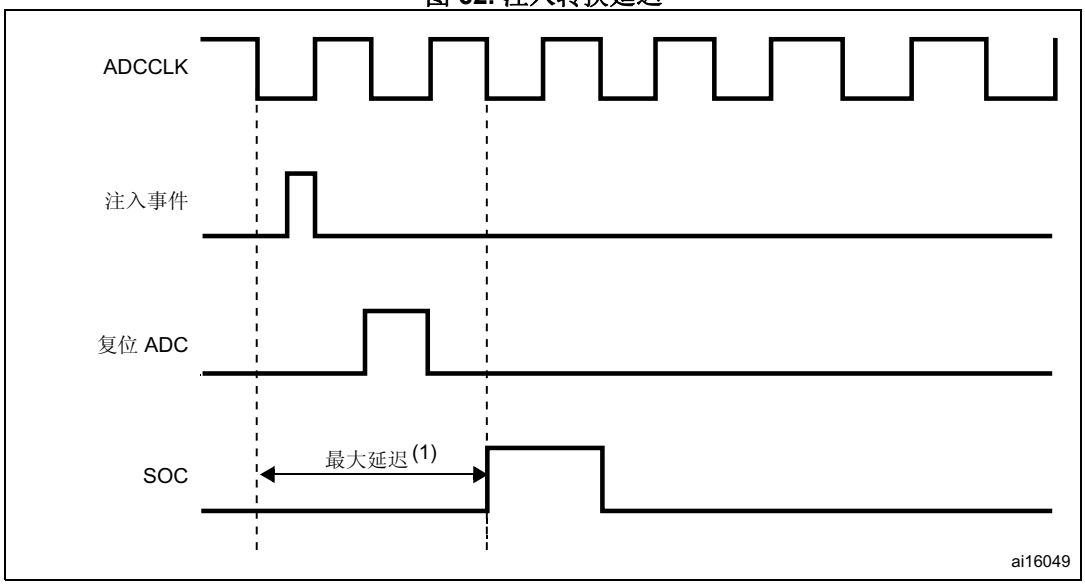
如果将 JAUTO 位置 1，则注入组中的通道会在常规组通道之后自动转换。这可用于转换最多由 20 个转换构成的序列，这些转换在 ADC_SQRx 和 ADC_JSQR 寄存器中编程。

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 62. 注入转换延迟



1. 有关延迟的最大值，请参见 STM32F413/423 数据手册中的电气特性部分。

13.3.10 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 ≤ 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。每次转换后都产生 EOC 事件
- 第二次触发：转换序列 3、6、7。每次转换后都产生 EOC 事件
- 第三次触发：转换序列 9、10。每次转换后都产生 EOC 事件
- 第四次触发：转换序列为 0、1、2。每次转换后都产生 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。

13.4 数据对齐

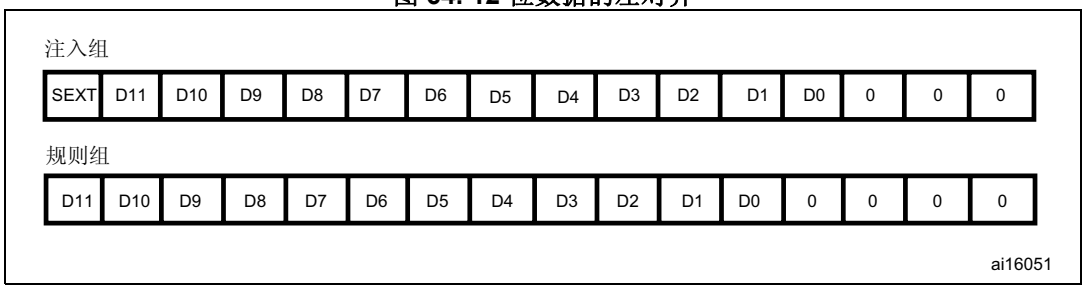
ADC_CR2 寄存器中的 ALIGN 位用于选择转换后存储的数据的对齐方式。可选择左对齐和右对齐两种方式，如*图 63* 和*图 64* 所示。

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。

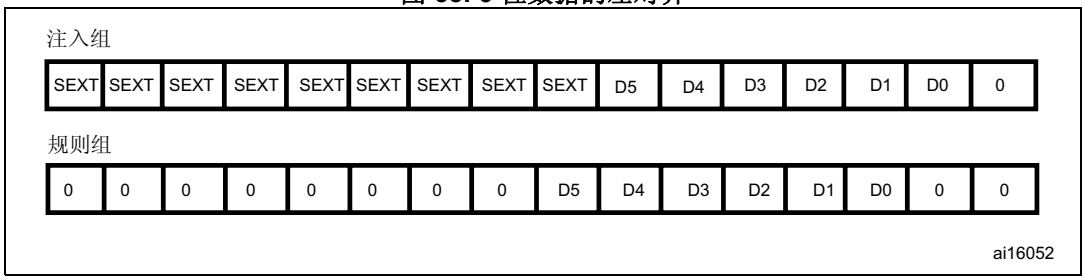
图 63. 12 位数据的右对齐

图 64. 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 65* 所示。

图 65. 6 位数据的左对齐



13.5 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）

13.6 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 76* 提供了 EXTEN[1:0] 和 JEXTEN[1:0] 值与触发极性之间的对应关系。

表 76. 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 77*给出了可用于常规转换的外部触发。

表 77. 常规通道的外部触发

源	类型	EXTSEL[3:0]
TIM1_CH1 事件	片上定时器的内部信号	0000
TIM1_CH2 事件		0001
TIM1_CH3 事件		0010
TIM2_CH2 事件		0011
TIM2_CH3 事件		0100
TIM2_CH4 事件		0101
TIM2_TRGO 事件		0110
TIM3_CH1 事件		0111
TIM3_TRGO 事件		1000
TIM4_CH4 事件		1001
TIM5_CH1 事件		1010
TIM5_CH2 事件		1011
TIM5_CH3 事件		1100
TIM8_CH1 事件		1101
TIM8_TRGO 事件		1110
EXTI 线 11	外部引脚	1111

*表 78*给出了可用于注入转换的外部触发。

表 78. 注入通道的外部触发

源	连接类型	JEXTSEL[3:0]
TIM1_CH4 事件	片上定时器的内部信号	0000
TIM1_TRGO 事件		0001
TIM2_CH1 事件		0010
TIM2_TRGO 事件		0011
TIM3_CH2 事件		0100
TIM3_CH4 事件		0101
TIM4_CH1 事件		0110
TIM4_CH2 事件		0111
TIM4_CH3 事件		1000
TIM4_TRGO 事件		1001
TIM5_CH4 事件		1010
TIM5_TRGO 事件		1011
TIM8_CH2 事件		1100
TIM8_CH3 事件		1101
TIM8_CH4 事件		1110
EXTI 线 15	外部引脚	1111

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。*

13.7 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 个 ADCCLK 周期
- 10 位：3 + 10 = 13 个 ADCCLK 周期
- 8 位：3 + 8 = 11 个 ADCCLK 周期
- 6 位：3 + 6 = 9 个 ADCCLK 周期

13.8 数据管理

13.8.1 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。

13.8.2 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。

13.8.3 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。

13.9 温度传感器

温度传感器可用于测量器件的环境温度 (T_A)。

图 66 显示了温度传感器框图。

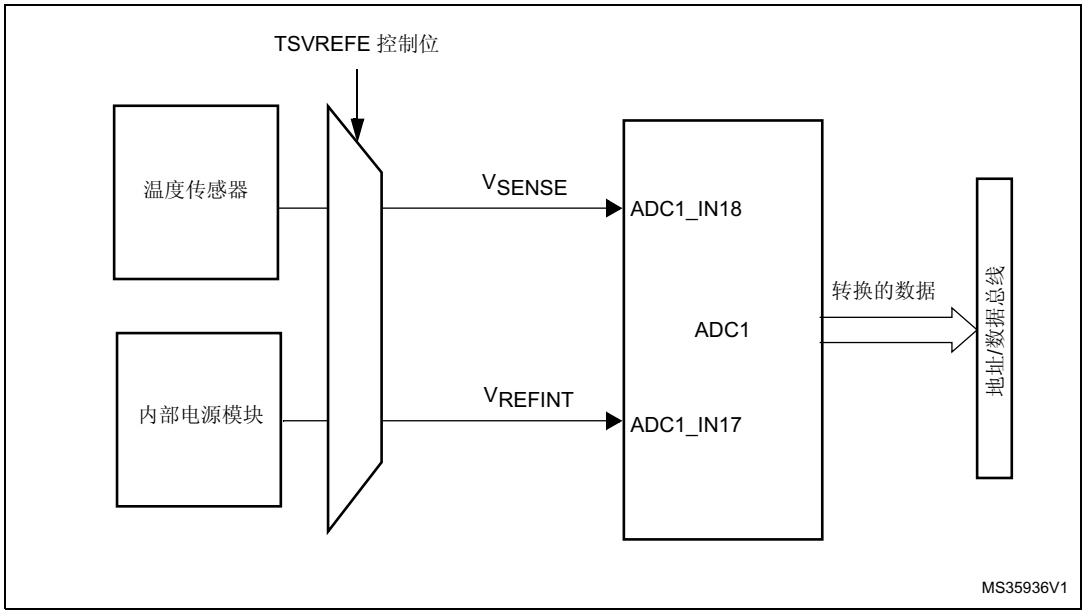
不使用时可将传感器置于掉电模式。

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 66. 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：

3. 选择 ADC1_IN18 输入通道。
4. 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。
5. 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。
6. 通过将 SWSTART 位置 1（或通过外部触发）开始 ADC 转换。
7. 读取 ADC 数据寄存器中生成的 V_{SENSE} 数据。
8. 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。

13.10 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。*

13.11 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 79. 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
溢出	OVR	OVRIE

13.12 ADC 寄存器

有关寄存器说明中使用的缩写，请参见*第 50 页的第 1.2 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）。

13.12.1 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

13.12.2 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（最小 15 个 ADCCLK 周期）
01：10 位（最小 13 个 ADCCLK 周期）
10：8 位（最小 11 个 ADCCLK 周期）
11：6 位（最小 9 个 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
保留其他值

13.12.3 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 2 CC3 事件
0101：定时器 2 CC4 事件
0110：定时器 2 TRGO 事件
0111：定时器 3 CC1 事件
1000：定时器 3 TRGO 事件
1001：定时器 4 CC4 事件
1010：定时器 5 CC1 事件
1011：定时器 5 CC2 事件
1100：定时器 5 CC3 事件
1101：定时器 8 CC1 事件
1110：定时器 8 TRGO 事件
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
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 CC4 事件
0001：定时器 1 TRGO 事件
0010：定时器 2 CC1 事件
0011：定时器 2 TRGO 事件
0100：定时器 3 CC2 事件
0101：定时器 3 CC4 事件
0110：定时器 4 CC1 事件
0111：定时器 4 CC2 事件
1000：定时器 4 CC3 事件
1001：定时器 4 TRGO 事件
1010：定时器 5 CC4 事件
1011：定时器 5 TRGO 事件
1100：定时器 8 CC2 事件
1101：定时器 8 CC3 事件
1110：定时器 8 CC4 事件
1111：EXTI 线 15

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 63*和*图 64*。
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 DMA 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。

0：禁止 ADC 转换并转至掉电模式

1：使能 ADC

13.12.4 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

13.12.5 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

13.12.6 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。

13.12.7 ADC 看门狗阈值上限寄存器 (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]**：模拟看门狗阈值上限 (Analog watchdog higher threshold)
通过软件写入这些位可为模拟看门狗定义阈值上限。

注： *软件可以在 ADC 转换期间写入这些寄存器，编程的值将在下一次转换完成后生效。对该寄存器执行写操作时存在写入延迟，这可能会导致所编程新值的生效时间出现不确定性。*

13.12.8 ADC 看门狗阈值下限寄存器 (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]**：模拟看门狗阈值下限 (Analog watchdog lower threshold)
通过软件写入这些位可为模拟看门狗定义阈值下限。

注： *软件可以在 ADC 转换期间写入这些寄存器，编程的值将在下一次转换完成后生效。对该寄存器执行写操作时存在写入延迟，这可能会导致所编程新值的生效时间出现不确定性。*

13.12.9 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)

13.12.10 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 29:25 **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)

13.12.11 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)

13.12.12 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。

13.12.13 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 63* 和*图 64* 所示。

13.12.14 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 63* 和*图 64* 所示。

13.12.15 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 基址 + 0x300 相关）

复位值：0x0000 0000

该寄存器可用于查看 ADC1 的各个状态位。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.	Res.	Res.	Res.	Res.	Res.	Res.
15	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.	OVR1	STRT1	JSTRT1	JEOC 1	EOC1	AWD1
										r	r	r	r	r	r

位 31:6 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。

13.12.16 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
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 **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，用于选择 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 15:0 保留，必须保持复位值。

13.12.17 ADC 寄存器映射

下表对 ADC 寄存器进行了汇总。

表 80. ADC 全局寄存器映射

偏移	寄存器
0x000 - 0x04C	ADC1
0x050 - 0x2FC	保留
0x300 - 0x308	通用寄存器

表 81. 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.	DISC NUM [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

表 81. 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
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
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 82. ADC 寄存器映射和复位值（通用 ADC 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.	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_CCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSVREFE	VBATE	Res.	Res.	Res.	Res.	ADCPRE[1:0]		Res.	Res.	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 54 页的第 2.2.2 节*。

14 数模转换器 (DAC)

14.1 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。

14.2 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 67 所示为 DAC 通道的框图，*表 83* 则给出了引脚说明。

图 67. DAC 通道框图

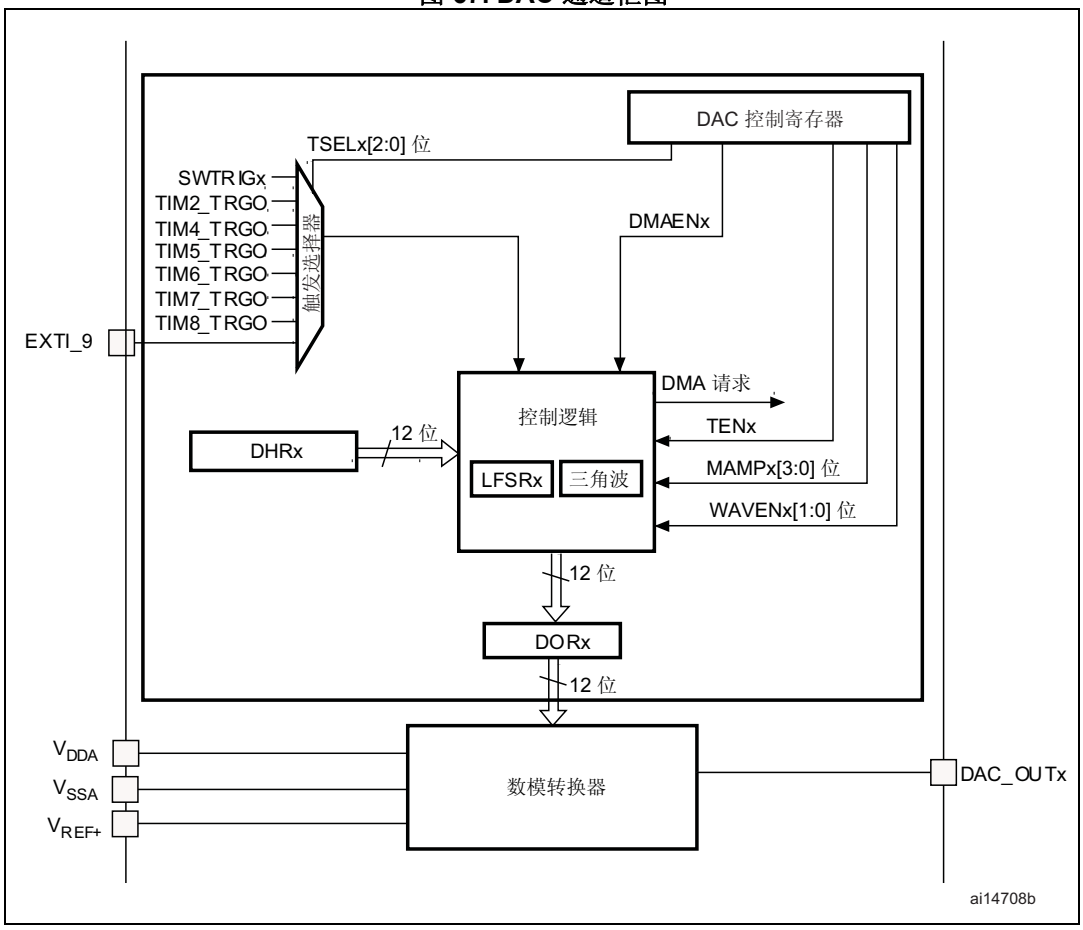


表 83. DAC 引脚

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，$V \leq V_{REF+} \leq V_{DDA}$
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)。*

14.3 DAC 功能说明

14.3.1 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 x 通道宏单元。即使 ENx 位复位，DAC x 通道数字接口仍处于使能状态。*

14.3.2 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。

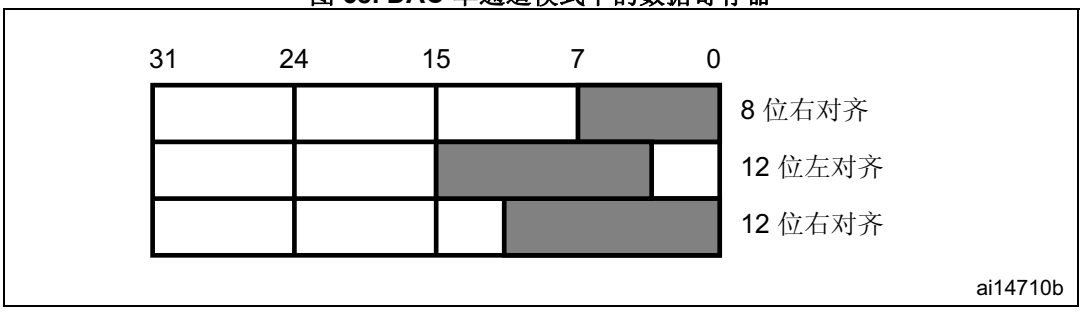
14.3.3 DAC 数据格式

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。

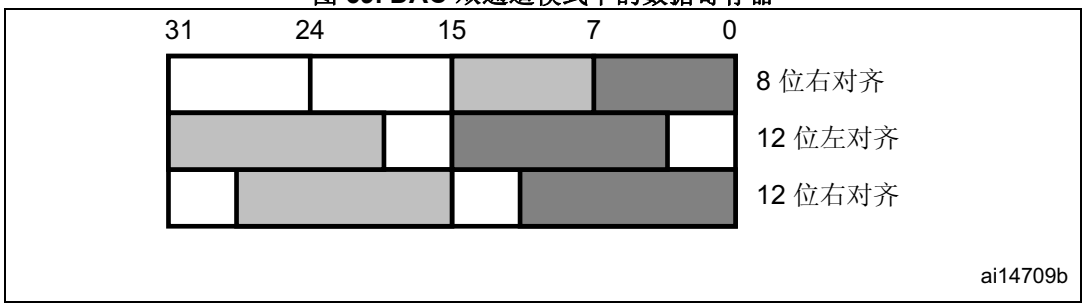
图 68. DAC 单通道模式下的数据寄存器



- 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 69. DAC 双通道模式下的数据寄存器



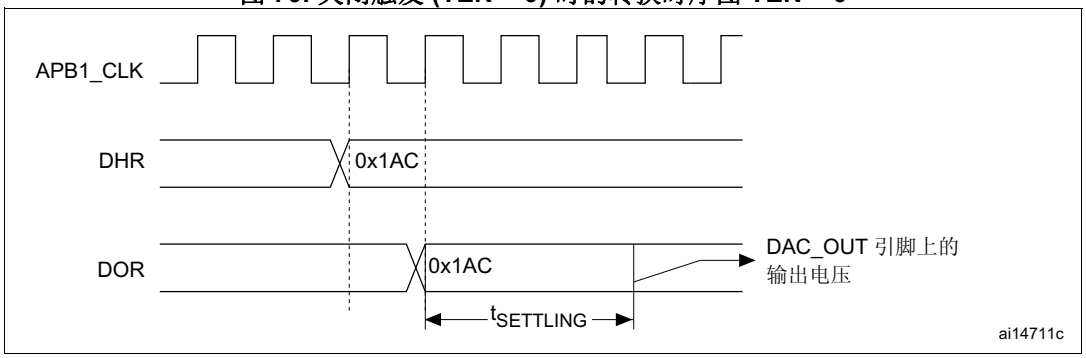
14.3.4 DAC 转换

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_DHR12LD 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 1 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 70. 关闭触发 (TEN = 0) 时的转换时序图 TEN = 0



14.3.5 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：

$$DACoutput = V_{REF} \times \frac{DOR}{4096}$$

14.3.6 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 84* 所示。

表 84. 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。

14.3.7 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，用户应仅将相应 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 DMA 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。

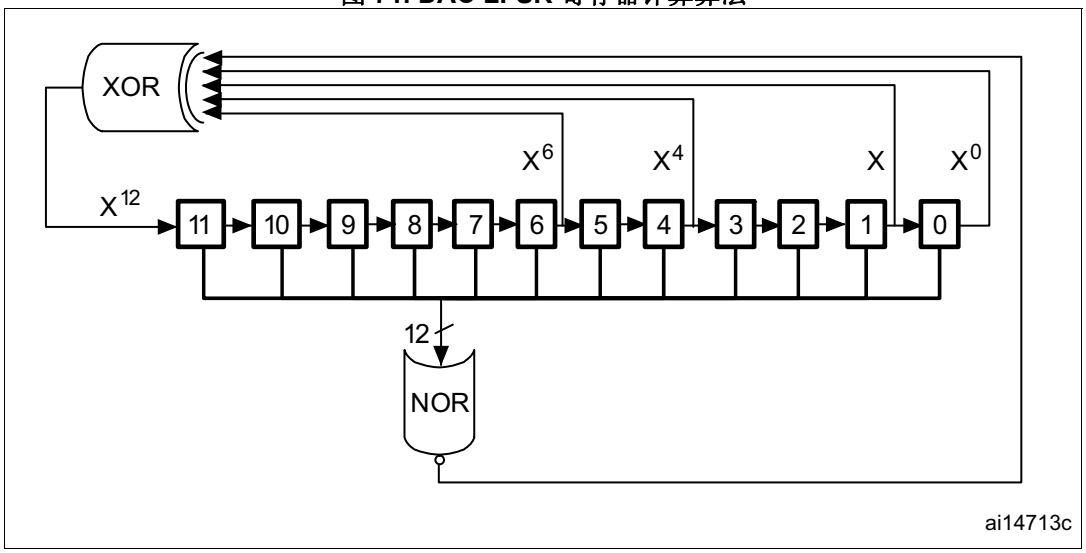
软件应通过写入“1”来将 DMAUDRx 标志清零，将所用 DMA 数据流的 DMAEN 位清零，并重新初始化 DMA 和 DAC 通道，以便正确地重新开始 DMA 传输。软件应修改 DAC 触发转换频率或减轻 DMA 工作负载，以避免再次发生 DMA 下溢。最后，可通过使能 DMA 数据传输和转换触发来继续完成 DAC 转换。

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。

14.3.8 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 71. DAC LFSR 寄存器计算算法

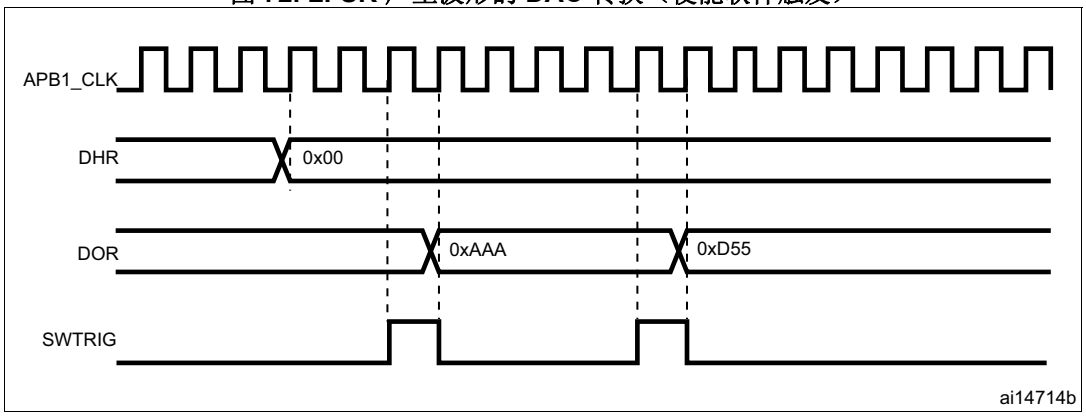


LFSR 值可以通过 DAC_CR 寄存器中的 MAMPx[3:0] 位来部分或完全屏蔽，在不发生溢出的情况下，该值将与 DAC_DHRx 的内容相加，然后存储到 DAC_DORx 寄存器中。

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 72. 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。*

14.3.9 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 73. 生成 DAC 三角波

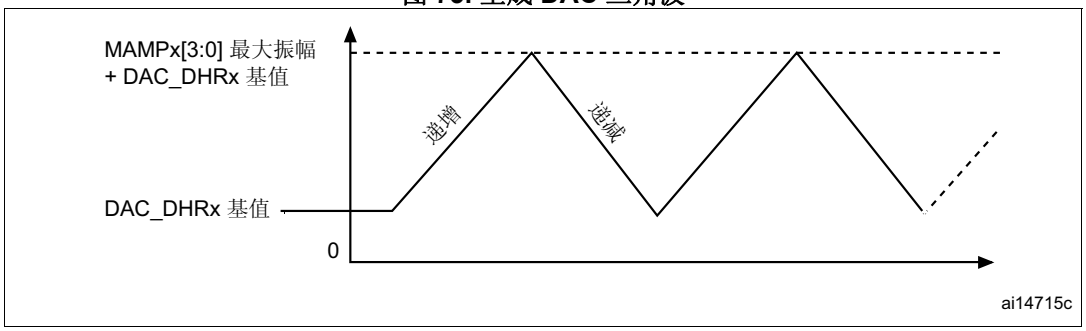
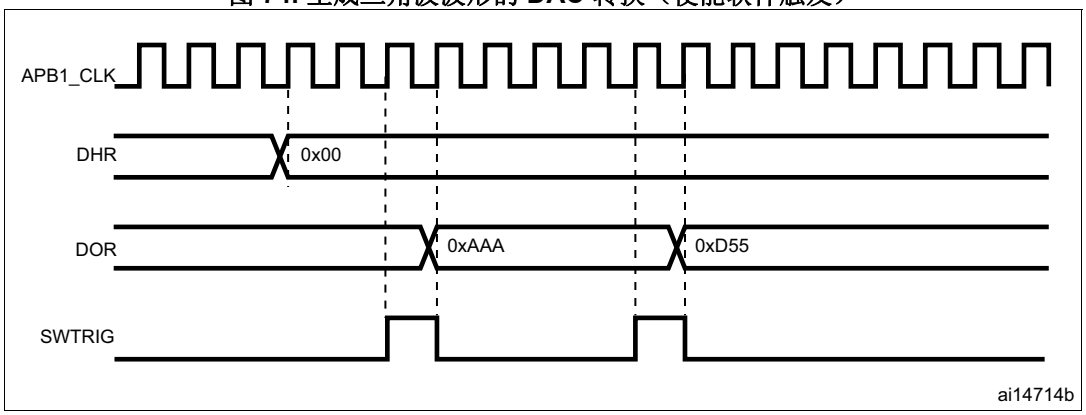


图 74. 生成三角波波形的 DAC 转换（使能软件触发）



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。*

14.4 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。

14.4.1 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）。

14.4.2 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 1 通道触发信号到达时，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 2 通道触发信号到达时，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。

14.4.3 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 1 通道触发信号到达时，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 2 通道触发信号到达时，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。

14.4.4 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 1 通道触发信号到达时，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。

14.4.5 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。

14.4.6 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。

14.4.7 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）。

14.4.8 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，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。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。

14.4.9 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。

14.4.10 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。

14.4.11 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 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。
同时，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。

14.5 DAC 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.2 节：寄存器相关缩写词列表*。

外设寄存器必须按字（32 位）进行访问。

14.5.1 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
Reserved		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
Reserved		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

14.5.2 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														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）。

14.5.3 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				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。

14.5.4 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC1DHR[11:0]												Reserved			
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。

14.5.5 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								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 8-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。

14.5.6 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				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。

14.5.7 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC2DHR[11:0]												Reserved			
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。

14.5.8 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								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 8-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。

14.5.9 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
Reserved				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
Reserved				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。

14.5.10 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]												Reserved			
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]												Reserved			
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。

14.5.11 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
Reserved															
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 8-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 8-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。

14.5.12 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				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。

14.5.13 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				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。

14.5.14 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
Reserved		DMAUDR2 rc_w1	Reserved												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DMAUDR1 rc_w1	Reserved												

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。

14.5.15 DAC 寄存器映射

表 85 汇总了 DAC 寄存器。

表 85. DAC 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	**DAC_CR**	Reserved		DMAUDRIE2	DMAEN2	MAMP2[3:0]				WAVE 2[2:0]		TSEL2[2:0]			TEN2	BOFF2	EN2	Reserved		DMAUDRIE1	DMAEN1	MAMP1[3:0]				WAVE 1[2:0]		TSEL1[2 :0]			TEN1	BOFF1	EN1
0x04	**DAC_ SWTRIGR**	Reserved																														SWTRIG2	SWTRIG1
0x08	**DAC_ DHR12R1**	Reserved																		DACC1DHR[11:0]													
0x0C	**DAC_ DHR12L1**	Reserved																DACC1DHR[11:0]												Reserved			
0x10	**DAC_ DHR8R1**	Reserved																								DACC1DHR[7:0]							
0x14	**DAC_ DHR12R2**	Reserved																		DACC2DHR[11:0]													

表 85. DAC 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	DAC_ DHR12L2	Reserved																DACC2DHR[11:0]												Reserved			
0x1C	DAC_ DHR8R2	Reserved																								DACC2DHR[7:0]							
0x20	DAC_ DHR12RD	Reserved				DACC2DHR[11:0]												Reserved				DACC1DHR[11:0]											
0x24	DAC_ DHR12LD	DACC2DHR[11:0]												Reserved				DACC1DHR[11:0]												Reserved			
0x28	DAC_ DHR8RD	Reserved																DACC2DHR[7:0]								DACC1DHR[7:0]							
0x2C	DAC_ DOR1	Reserved																				DACC1DOR[11:0]											
0x30	DAC_ DOR2	Reserved																				DACC2DOR[11:0]											
0x34	DAC_SR	Reserved		DMAUDR2		Reserved														DMAUDR1		Reserved											

请参见*第 2.2.2 节：存储器映射和寄存器边界地址*。

15 ∑Δ 调制器数字滤波器 (DFSDM)

15.1 简介

ΣΔ 调制器数字滤波器 (DFSDM) 是一种高性能模块，专用于将外部 ΣΔ 调制器连接到微控制器。它具有多达 8 个外部数字串行接口（通道）和多达 4 个数字滤波器，具有灵活的 ΣΔ 流数字处理选项，可提供高达 24 位 ADC 最终分辨率。DFSDM 还特有来自微控制器存储器的并行数据流输入可供选择。

外部 ΣΔ 调制器将其模拟输入的模拟值转换为数字数据流。此数字数据流通过串行接口发送到 DFSDM 输入通道。DFSDM 支持多种标准来连接各种 ΣΔ 调制器输出：SPI 接口和曼彻斯特编码单线接口（都具有可调参数）。DFSDM 模块支持连接多达 8 个复用输入数字串行通道，这些通道可与多达 4 个 DFSDM 模块共享。DFSDM 模块还支持来自多达 8 个内部 16 位数据通道（来自微控制器存储器）的可选并行数据输入。

DFSDM 将输入数据流转换为最终数字数据字，该数字字表示 ΣΔ 调制器模拟输入上的模拟输入值。该转换基于以下可配置的数字处理而完成：对输入串行数据流进行数字滤波和抽取。

转换速度和分辨率可根据以下可配置的数字处理参数进行调整：滤波器类型、滤波器阶数、滤波器长度和积分器长度。最大输出数据分辨率高达 24 位。有两种转换模式：单次转换模式和连续转换模式。数据可通过 DMA 自动存储在系统 RAM 缓冲区中，因此降低了软件开销。

可使用灵活的定时器触发系统来控制启动 DFSDM 转换。此定时控制可触发即时转换或延时转换，延时时间可以编程。

DFSDM 具有模拟看门狗功能。模拟看门狗可以分配给任何输入通道数据流或最终输出数据。模拟看门狗本身可以对输入数据流进行数字滤波，以达到被监视数据的所需速度和分辨率。

为了检测控制应用中的短路情况，提供了一个短路检测器。该模块监视各个输入通道数据流是否在定义的持续时间内保持稳定（即输入数据流中有若干 0 或 1）。

极值检测器模块监视最终输出数据并存储最大和最小输出数据值。存储的极值可通过软件重新启动。

支持两种功率模式：正常模式和停止模式。

15.2 DFSDM 主要特性

- 多达 8 个复用输入数字串行通道：
  - 可配置的 SPI 接口，用于连接各种 ΣΔ 调制器
  - 支持可配置的曼彻斯特编码单线接口
  - 用于 ΣΔ 调制器的时钟输出
- 来自多达 8 个内部数字并行通道的可选输入：
  - 分辨率高达 16 位的输入
  - 内部源：存储器（CPU/DMA 写）数据流
- 可调节的数字信号处理：
  - $Sinc^x$ 滤波器：滤波器阶数/类型 (1..5)，过采样率（高达 1..1024）
  - 积分器：过采样率 (1..256)
- 高达 24 位的输出数据分辨率：
  - 最终数据的右移位器（0..31 位）
- 有符号输出数据格式
- 自动数据偏移校正（偏移值由用户存储在寄存器中）
- 连续或单次转换
- 可通过以下途径同步启动转换：
  - 软件触发
  - 内部定时器
  - 外部事件
  - 使用第一个 DFSDM 滤波器 (DFSDM_FLT0) 同步启动转换
- 模拟看门狗功能：
  - 下限和上限数据阈值寄存器
  - 自带可配置的 $Sinc^x$ 数字滤波器（阶数 = 1..3，过采样率 = 1..32）
  - 输入来自输出数据寄存器或来自一个或多个输入数字串行通道
  - 独立于标准转换的连续监视
- 短路检测器，用于检测饱和的模拟输入值（下限和上限）：
  - 高达 8 位计数器，用于检测输入数据流中 1..256 个连续的 0 或 1
  - 连续监视各个通道（8 个串行通道收发器输出）
- 当发生模拟看门狗事件或短路检测器事件时，生成断路
- 极值检测器：
  - 存储最小和最大输出数据值
  - 由软件刷新
- DMA 可用于读取转换数据
- 中断：结束转换，溢出，模拟看门狗，短路，通道时钟缺失
- “常规”或“注入”转换：
  - “常规”转换可随时进行请求，即使在连续模式下亦如此，且不会对“注入”转换的时序产生影响

15.3 DFSDM 实现

本部分介绍在 DFSDMx 中实现的配置。

表 86. DFSDMx 实现

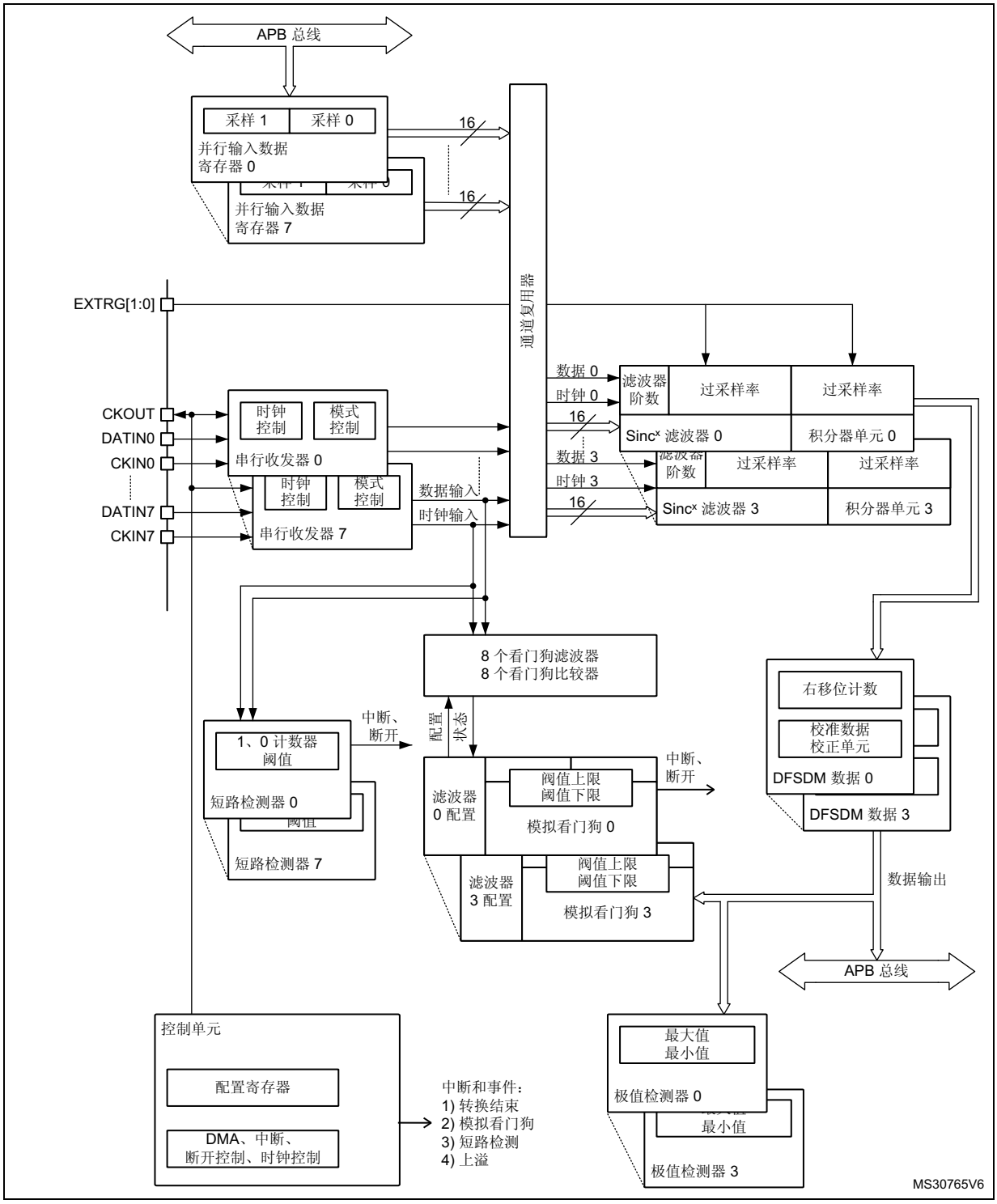
DFSDM 特性	DFSDM1	DFSDM2
通道数	4	8
滤波器数	2	4
来自内部 ADC 的输入	-	-
支持的触发源	10	10
脉冲跳过器	X(1)	X(1)
ID 寄存器支持	-	-

1. 具体实现的脉冲跳过器（请参见*脉冲跳过器*一节）。

15.4 DFSDM 功能说明

15.4.1 DFSDM 框图

图 75. 单个 DFSDM 框图



1. 本示例显示了 4 个 DFSDM 滤波器和 8 个输入通道（最大 配置）。

15.4.2 DFSDM 引脚和内部信号

表 87. DFSDM 外部引脚

名称	信号类型	备注
VDD	电源	数字电源。
VSS	电源	数字接地电源。
CKIN[7:0]	时钟输入	从外部 ΣΔ 调制器提供的时钟信号。FT 输入。
DATIN[7:0]	数据输入	从外部 ΣΔ 调制器提供的数据信号。FT 输入。
CKOUT	时钟输出	时钟输出，用于为外部 ΣΔ 调制器提供时钟信号。
EXTRG[1:0]	外部触发信号	来自两个 EXTI 信号的输入触发，用于启动模拟转换（来自 GPIO：EXTI11 和 EXTI15）。

表 88. DFSDM 内部信号

名称	信号类型	备注
dfsdm_jtrg[10:0]	内部/外部触发信号	来自内部/外部触发源的输入触发，用于启动模拟转换，有关详细信息，请参见*表 89* 和*表 90*。
dfsdm_break[3:0]	断路信号输出	由模拟看门狗或短路检测器触发的断路信号事件
dfsdm_dma[3:0]	DMA 请求信号	来自各个 DFSDM_FLTx (x=0..3) 的 DMA 请求信号：结束注入转换事件。
dfsdm_it[3:0]	中断请求信号	各个 DFSDM_FLTx (x=0..3) 的中断信号

表 89. DFSDM1 触发连接

触发器名称	触发源
dfsdm_jtrg0	TIM1_TRGO2
dfsdm_jtrg1	TIM3_TRGO2
dfsdm_jtrg2	TIM8_TRGO2
dfsdm_jtrg3	TIM10_OC1
dfsdm_jtrg4	N/A
dfsdm_jtrg5	TIM4_TRGO2
dfsdm_jtrg6	N/A
dfsdm_jtrg7	TIM6_TRGO1
dfsdm_jtrg8	N/A
dfsdm_jtrg9	EXTI11
dfsdm_jtrg10	EXTI15

表 90. DFSDM2 触发器连接

触发器名称	触发源
dfsdm_jtrg0	TIM1_TRGO3
dfsdm_jtrg1	TIM3_TRGO3
dfsdm_jtrg2	TIM8_TRGO4
dfsdm_jtrg3	TIM10_OC1
dfsdm_jtrg4	TIM2_TRGO2
dfsdm_jtrg5	TIM4_TRGO4
dfsdm_jtrg6	TIM11_OC1
dfsdm_jtrg7	TIM6_TRGO2
dfsdm_jtrg8	TIM7_TRGO2
dfsdm_jtrg9	EXTI11
dfsdm_jtrg10	EXTI15

表 91. DFSDM 断路连接

断路名称	断路目标
dfsdm_break[0]	TIM1 断路
dfsdm_break[1]	-
dfsdm_break[2]	TIM8 断路
dfsdm_break[3]	-

15.4.3 DFSDM 复位和时钟

DFSDM 开关控制

通过将 DFSDM_CH0CFGR1 寄存器的 DFSDMEN 置 1，全局使能 DFSDM 接口。DFSDM 被全局使能后，所有输入通道 (y=0..7) 和数字滤波器 DFSDM_FLTx (x=0..3) 会在其使能位（DFSDM_CHyCFGR1 中的通道使能位 CHEN 和 DFSDM_FLTxCR1 中的 DFSDM_FLTx 使能位 DFEN）置 1 时开始工作。

数字滤波器 x DFSDM_FLTx (x=0..3) 通过将 DFSDM_FLTxCR1 寄存器的 DFEN 置 1 来使能。使能 DFSDM_FLTx (DFEN=1) 后，$Sinc^x$ 数字滤波器单元和积分器单元会重新初始化。

通过将 DFEN 清零，正在进行的所有转换都会立即停止，DFSDM_FLTx 会置于停止模式。除 DFSDM_FLTxAWSR 和 DFSDM_FLTxISR（均被复位）外，所有寄存器设置保持不变。

通过将 DFSDM_CHyCFGR1 寄存器的 CHEN 置 1 使能通道 y (y=0..7)。通道使能后，会从外部 ΣΔ 调制器或并行内部数据源（存储器的 CPU/DMA 写操作）接收串行数据。

在停止系统时钟，使器件进入停止模式之前，必须全局禁止 DFSDM（通过在 DFSDM_CH0CFGR1 中设置 DFSDMEN = 0）。

DFSDM 时钟

内部 DFSDM 时钟 $f_{DFSDMCLK}$ 用于驱动通道收发器、数字处理模块（数字滤波器、积分器）和后续附加模块（模拟看门狗、短路检测器、极值检测器、控制模块），此时钟由 RCC 模块生成，源自系统时钟 SYSCLK 或外设时钟 PCLK2（请参见中的 DFSDMSEL 位说明）。DFSDM 时钟在停止模式下自动停止（对于所有 DFSDM_FLTx，x=0..3，DFEN = 0 时）。

DFSDM 串行通道收发器可以接收外部串行时钟，以采样外部串行数据流。如果使用标准 SPI 编码，内部 DFSDM 时钟必须至少比外部串行时钟快 4 倍；如果使用曼彻斯特编码，则须比外部串行时钟快 6 倍。

DFSDM 可提供一个外部输出时钟信号，以驱动外部 ΣΔ 调制器时钟输入。它通过 CKOUT 引脚提供。该输出时钟信号必须在给定器件手册中的指定范围之内，可以通过分频器取自 DFSDM 时钟，也可以取自音频时钟（请参见 DFSDM_CH0CFGR1 寄存器 CKOUTSRC 位），分频器可以编程，范围为 2 - 256（DFSDM_CH0CFGR1 寄存器中的 CKOUTDIV）。音频时钟源为 SAI1 时钟，可通过 RCC 配置中 SAI1SEL[1:0] 字段进行选择（请参见相关章节）。

15.4.4 串行通道收发器

有 8 个复用串行数据通道，可以通过各个滤波器、模拟看门狗或者短路检测器将其选择用于转换。这些串行收发器接收来自外部 ΣΔ 调制器的数据流。数据流能以 SPI 格式或曼彻斯特编码格式进行发送（请参见 DFSDM_CHyCFGR1 寄存器的 SITP[1:0] 位）。
通过将 DFSDM_CHyCFGR1 寄存器的 CHEN 置 1 可以使能通道。

通道输入选择

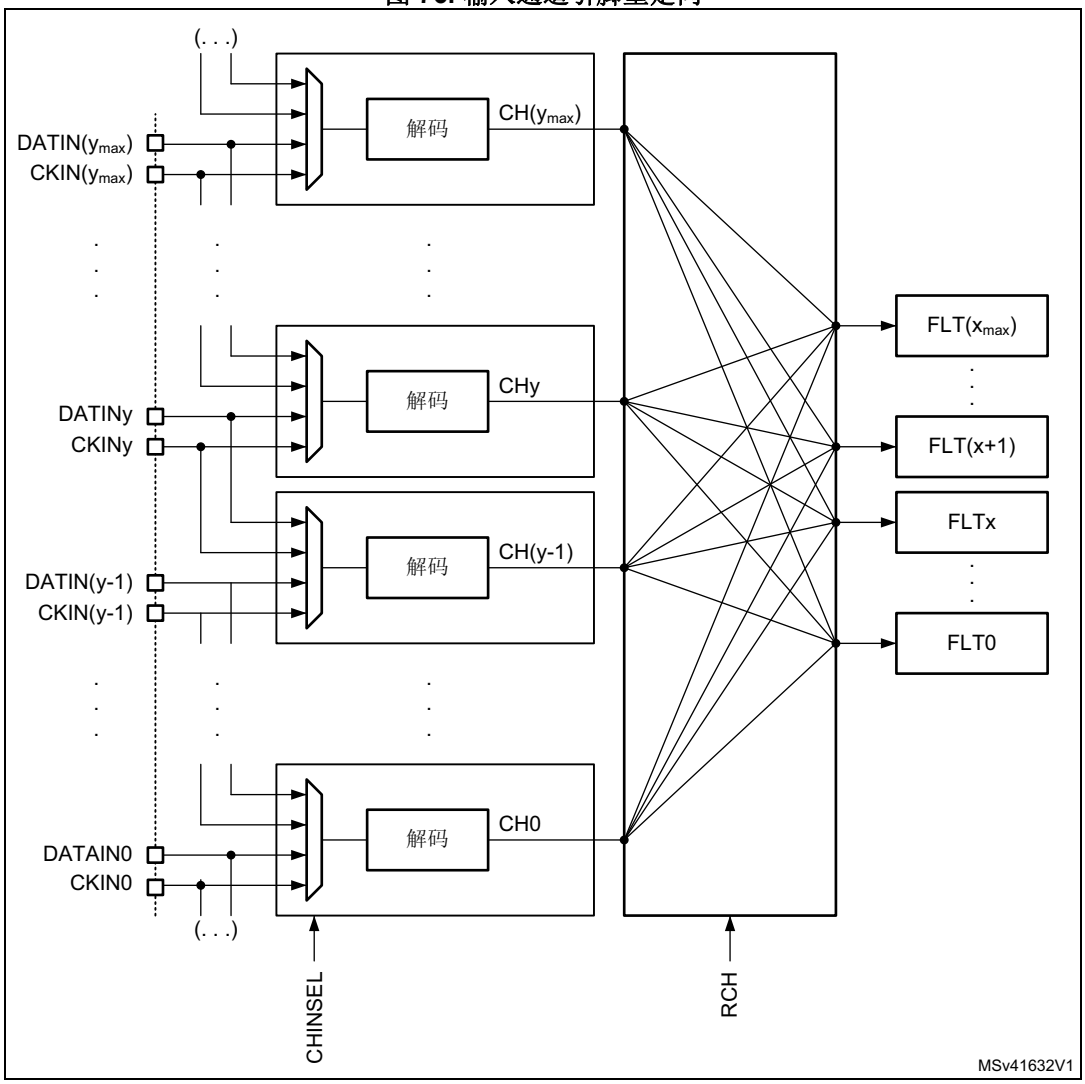
DATINy 和 CKINy 引脚的串行输入（数据和时钟信号）可从后续通道引脚重定向。此串行输入通道重定向可通过 DFSDM_CHyCFGR1 寄存器中的 CHINSEL 位设置。

通道重定向可用于从 PDM（脉冲密度调制）立体声麦克风中采集音频数据。PDM 立体声麦克风包含一个数据和一个时钟信号。数据信号为左右音频通道提供信息（时钟上升沿采样用于左通道，时钟下降沿采样用于右通道）。

PDM 麦克风输入串行通道的配置：

- PDM 麦克风信号（数据、时钟）将连接到 DFSDM 输入串行通道 y（DATINy，CKOUT）引脚。
- 将通道 y 进行以下配置：CHINSEL = 0（输入来自给定通道引脚：DATINy 和 CKINy）。
- 将通道 (y-1)（模8）进行以下配置：CHINSEL = 1（输入来自后续通道 ((y-1)+1) 引脚：DATINy 和 CKINy）。
- 通道 y：SITP[1:0] = 0（上升沿选通数据）=> 通道 y 的左音频通道。
- 通道 (y-1)：SITP[1:0] = 1（下降沿选通数据）=> 通道 y-1 的右音频通道。
- 两个 DFSDM 滤波器将分配至通道 y 和通道 (y-1)（用于对 PDM 麦克风的左右通道进行滤波）。

图 76. 输入通道引脚重定向



输出时钟生成

可在 CKOUT 引脚上提供一个时钟信号，来驱动外部 ΣΔ 调制器时钟输入。此 CKOUT 信号的频率通过预分频器（请参见 DFSDM_CH0CFGR1 寄存器 CKOUTDIV 位）取自 DFSDM 时钟或音频时钟（请参见 DFSDM_CH0CFGR1 寄存器 CKOUTSRC 位）。如果输出时钟停止，则 CKOUT 信号会被置于低电平状态（通过在 DFSDM_CHyCFGR1 寄存器中设置 CKOUTDIV = 0 或 在 DFSDM_CH0CFGR1 寄存器中设置 DFSDMEN = 0，可以停止输出时钟）。在以下时间执行输出时钟停止：

- DFSDMEN 清零后 4 个系统时钟（当 CKOUTSRC = 0 时）
- DFSDMEN 清零后 1 个系统时钟和 3 个音频时钟（CKOUTSRC = 1 时）

更改 CKOUTSRC 之前，软件必须等待 CKOUT 停止，以避免在 CKOUT 引脚上产生毛刺信号。输出时钟信号频率必须介于 0 - 20 MHz 范围内。

SPI 数据输入格式操作

在 SPI 格式下，数据流通过数据和时钟信号以串行格式进行发送。数据信号始终由 DATINy 引脚提供。时钟信号可通过 CKINy 引脚从外部提供，也可以通过取自 CKOUT 信号源的信号从内部提供。

如果选择外部时钟源 (SPICKSEL[1:0] = 0)，则根据（DFSDM_CHyCFGR1 寄存器中的）SITP[1:0] 位设置在（CKINy 引脚的）时钟上升沿或下降沿采样（DATINy 引脚上的）数据信号。

内部时钟源——请参见 DFSDM_CHyCFGR1 寄存器中的 SPICKSEL[1:0]：

- CKOUT 信号：
  - 用于连接外部 ΣΔ 调制器，调制器直接使用其时钟输入（来自 CKOUT）来生成输出串行通信时钟。
  - 采样点：上升沿/下降沿，具体取决于 SITP[1:0] 设置。
- CKOUT/2 信号（在 CKOUT 上升沿生成）：
  - 用于连接外部 ΣΔ 调制器，调制器将时钟输入（来自 CKOUT）除以 2，来生成输出串行通信时钟（该输出时钟变化在各个时钟输入上升沿有效）。
  - 采样点：每第二个 CKOUT 下降沿。
- CKOUT/2 信号（在 CKOUT 下降沿生成）：
  - 用于连接外部 ΣΔ 调制器，调制器将时钟输入（来自 CKOUT）除以 2，来生成输出串行通信时钟（该输出时钟变化在各个时钟输入下降沿有效）。
  - 采样点：每第二个 CKOUT 上升沿。

注： *只有在外部 ΣΔ 调制器将 CKOUT 信号用作时钟输入（以实现同步时钟和数据操作）时，才能使用内部时钟源。*

使用内部时钟源可省去 CKINy 引脚连接（CKINy 引脚可用于其他用途）。

如果采用 SPI 编码，时钟源信号频率必须介于 0 - 20 MHz 范围内，且小于 $f_{DFSDMCLK}/4$。

曼彻斯特编码数据输入格式操作

如果采用曼彻斯特编码格式，数据流仅通过 DATINy 引脚以串行格式进行发送。经过曼彻斯特解码之后，数据和时钟信号从串行流中恢复。可以有两种曼彻斯特编码设置（请参见 DFSDM_CHyCFGR1 寄存器 SITP[1:0] 位）：

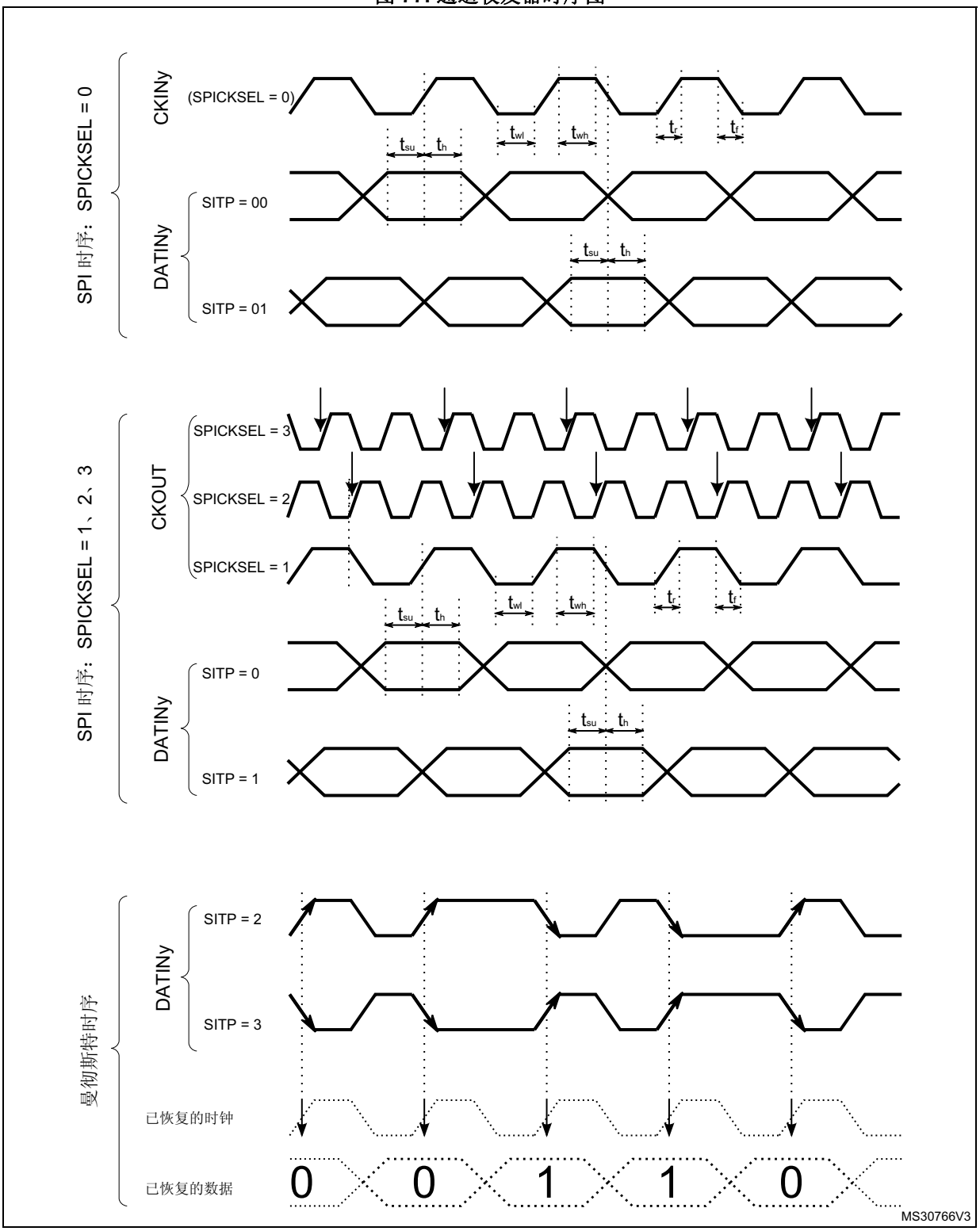
- 信号上升沿 = 逻辑 0；信号下降沿 = 逻辑 1
- 信号上升沿 = 逻辑 1；信号下降沿 = 逻辑 0

如果采用曼彻斯特编码，则恢复后的时钟信号频率必须介于 0 - 10 MHz 范围内，且小于 $f_{DFSDMCLK}/6$。

为了正确接收曼彻斯特编码数据，必须根据预期的曼彻斯特数据速率对（DFSDM_CH0CFGR1 寄存器中的）CKOUTDIV 分频器进行设置，设置公式为：

$$((CKOUTDIV + 1) \times T_{SYSCLK}) < T_{Manchester\ clock} < (2 \times CKOUTDIV \times T_{SYSCLK})$$

图 77. 通道收发器时序图



时钟缺失检测

可检查通道串行时钟输入的时钟缺失/存在情况，以确保转换和错误报告正常运行。可通过 DFSDM_CHyCFGR1 寄存器 CKABEN 位在各个输入通道 y 上使能或禁止时钟缺失检测。如果使能了给定通道的时钟缺失检测，则会对其连续执行时钟缺失检测。如果出现输入时钟错误，则时钟缺失标志会置 1 (CKABF[y] = 1)，并会调用中断（CKABIE = 1 时）（请参见 DFSDM_FLT0ISR 寄存器的 CKABF[7:0] 和 DFSDM_CHyCFGR1 的 CKABEN）。时钟缺失标志清零（通过 DFSDM_FLT0ICR 寄存器的 CLRCKABF）后，会刷新时钟缺失标志。相应通道 y 禁止时，则时钟缺失状态位 CKABF[y] 还会由硬件置 1（如果 CHEN[y] = 0，则 CKABF[y] 保持在置 1 状态）。

如果发生了时钟缺失事件，则数据转换（和/或模拟看门狗和短路检测器）会提供错误的数据。当报告了时钟缺失时，用户应管理此事件并丢弃给定数据。

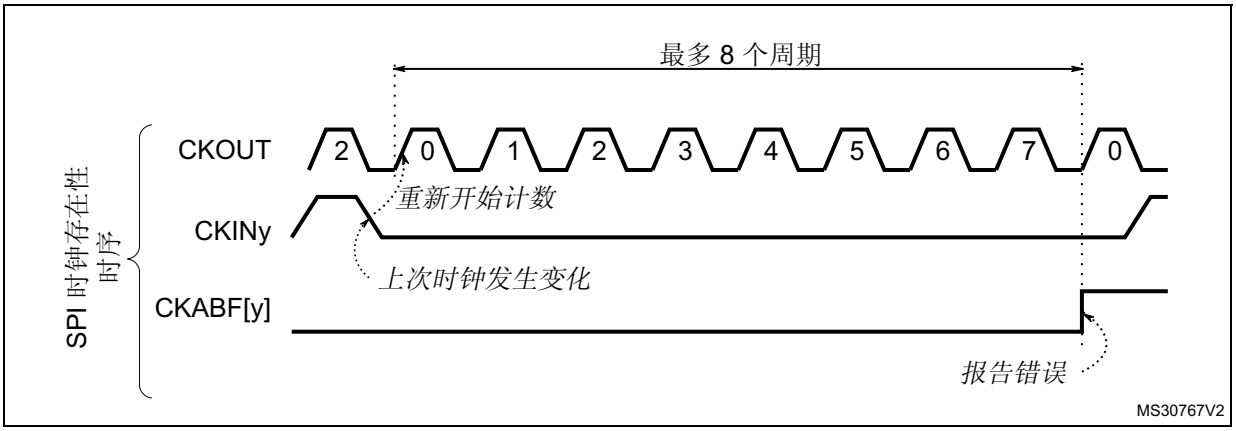
只有在将系统时钟用于 CKOUT 信号（DFSDM_CH0CFGR1 寄存器中的 CKOUTSRC = 0）时，时钟缺失功能才可用。

当收发器尚未被同步时，时钟缺失标志会置 1，且无法由（DFSDM_FLT0ICR 寄存器中）CLRCKABF[y] 位清零。与时钟缺失检测功能相关的软件序列应：

- 通过 CHEN = 1 使能给定通道
- 尝试将时钟缺失标志清零（通过 CLRCKABF = 1），直至时钟缺失标志已实际清零 (CKABF = 0)。此时，收发器被同步（信号时钟有效），能够接收数据。
- 使能时钟缺失功能 CKABEN = 1 且相关中断 CKABIE = 1，以检测 SPI 时钟是否丢失或曼彻斯特数据边沿是否缺失。

如果使用 SPI 数据格式，将基于外部输入时钟与输出时钟生成（CKOUT 信号）的比较进行时钟缺失检测。输入通道中的外部输入时钟信号必须每经过 CKOUT 信号（由 DFSDM_CH0CFGR1 寄存器 CKOUTDIV 字段控制）的 8 个信号周期更改一次。

图 78. SPI 时钟缺失时序图



如果使用曼彻斯特数据格式，则时钟缺失表示无法从曼彻斯特编码信号中恢复时钟。为了能够正常恢复时钟，首先必须接收带 1 - 0 或 0 - 1 转换的数据（有关曼彻斯特同步，请参见 *图 80*）。

如果使用曼彻斯特编码格式，则时钟缺失检测（在首次成功同步之后）是基于编码串行数据输入信号与输出时钟生成（CKOUT 信号）之间的变化比较。在 CKOUT 信号（由 DFSDM_CH0CFGR1 寄存器 CKOUTDIV 位控制）的两个周期期间，DATINy 引脚电平必须发生变化。该条件还定义了能正确恢复曼彻斯特编码的数据和时钟信号的最小数据速率。

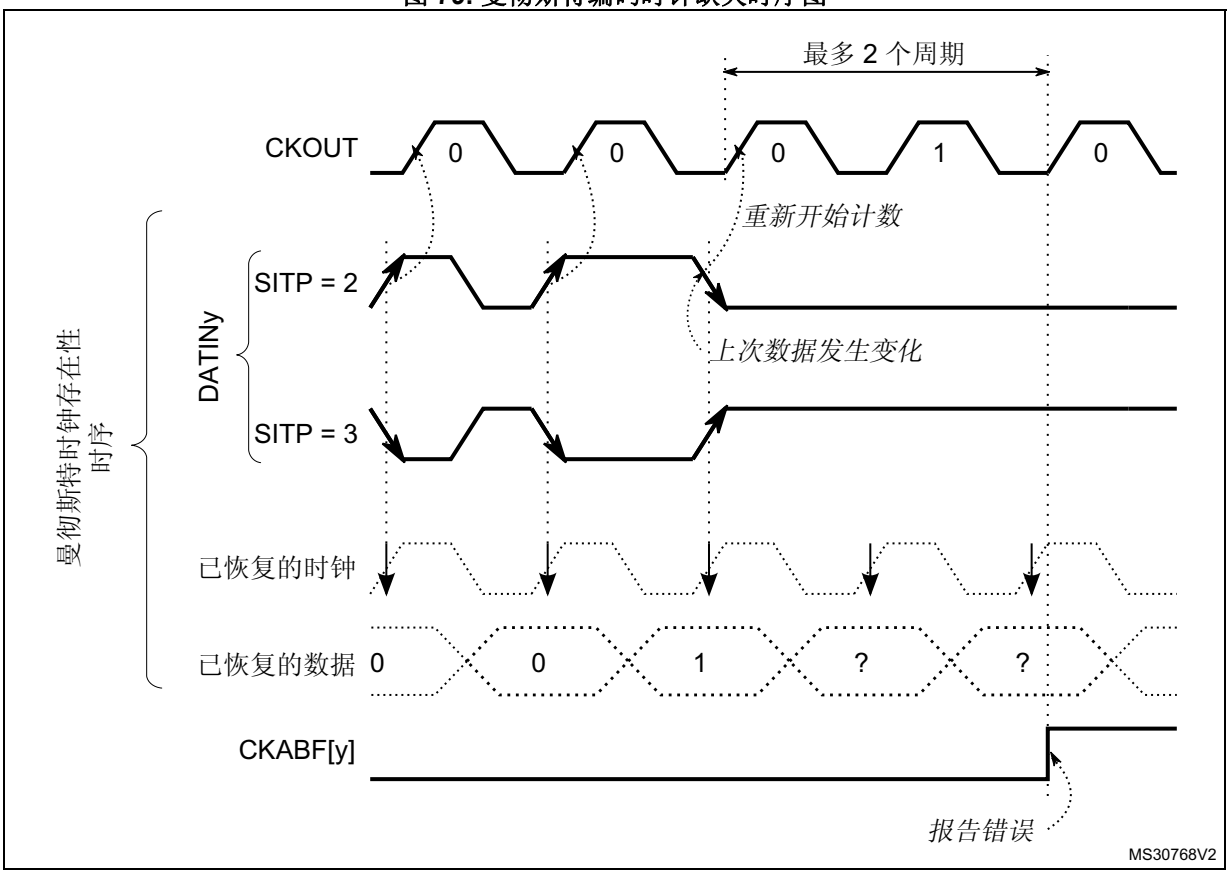
曼彻斯特编码数据的最大数据速率必须小于 CKOUT 信号。

因此，为正确接收曼彻斯特编码数据，必须根据以下公式设置 CKOUTDIV 分频器：

$$((CKOUTDIV + 1) \times T_{SYSCLK}) < T_{Manchester\ clock} < (2 \times CKOUTDIV \times T_{SYSCLK})$$

如果出现输入时钟恢复错误，则时钟缺失标志会置 1 (CKABF[y] = 1)，并会调用中断（CKABIE=1 时）（请参见 DFSDM_FLT0ISR 寄存器的 CKABF[7:0] 和 DFSDM_CHyCFGR1 的 CKABEN）。时钟缺失标志清零（通过 DFSDM_FLT0ICR 寄存器的 CLRCKABF）后，会刷新时钟缺失标志。

图 79. 曼彻斯特编码时钟缺失时序图



曼彻斯特/SPI 编码同步

必须在使能通道（DFSDM_CHyCFGR1 寄存器 CHEN = 1）后，首次同步曼彻斯特编码数据流。当接收到 0 - 1 或 1 - 0 数据转换（以便能检测有效数据边沿）时，同步结束。在通过 DFSDM_FLT0ICR 的 CLRCKABF [y] 将给定通道的 CKABF[y] 清零之后，可以按照以下详细描述的软件序列通过轮询 CKABF [y] = 0 来检查同步是否结束：

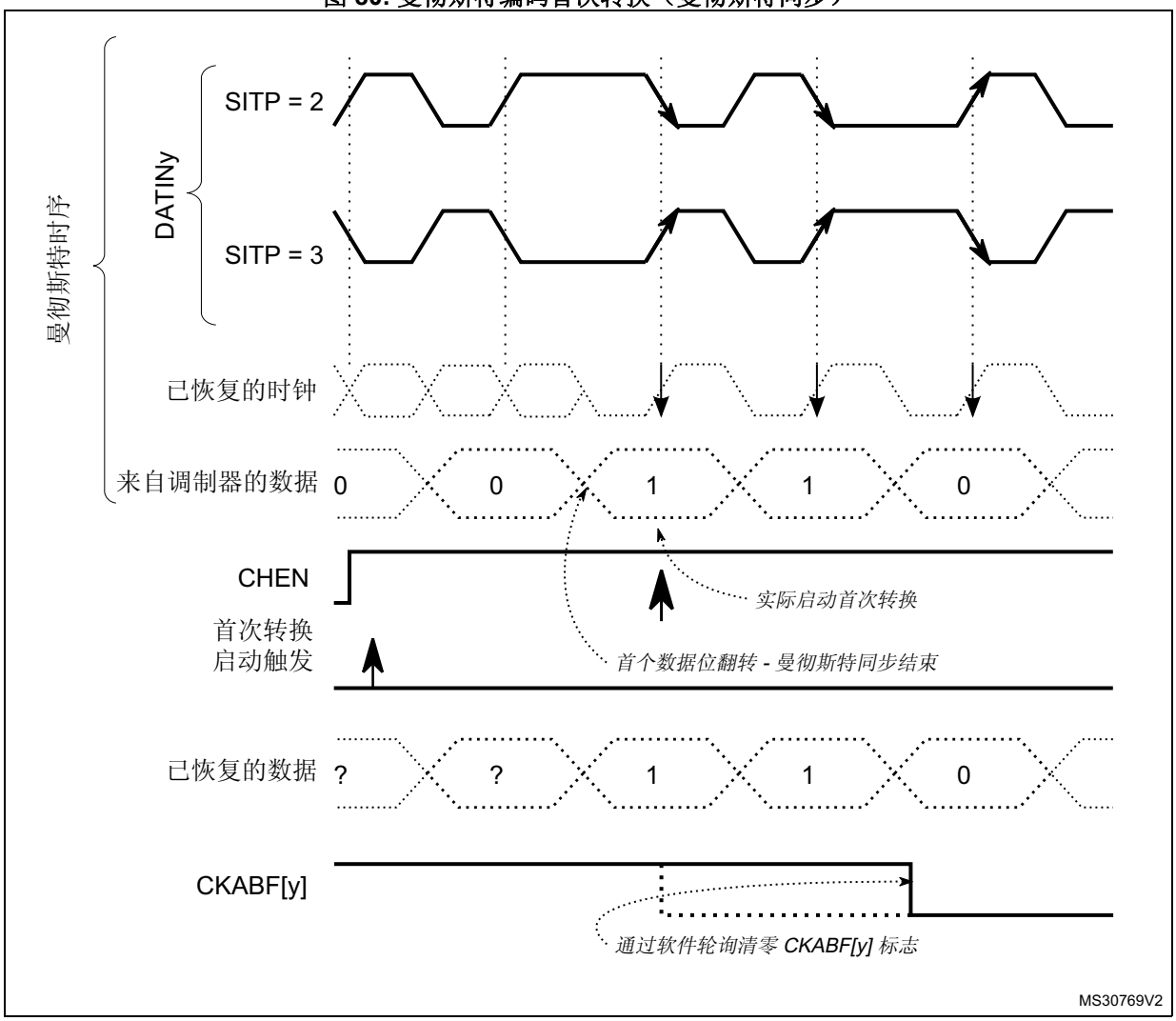
通过将 CLRCKABF[y] 位置 1 将 CKABF[y] 标志清零。如果通道 y 尚未同步，则硬件会立即将 CKABF[y] 标志置 1。软件随后将回读 CKABF[y] 标志，如果该标志置 1，则会再次通过将 CLRCKABF[y] 位置 1 来将其清零。此软件序列（CKABF[y] 标志的轮询）将继续进行，直至 CKABF[y] 标志置 1（表示曼彻斯特码流已同步）。为了能够同步/接收曼彻斯特编码数据，必须根据预期的曼彻斯特数据速率对（DFSDM_CH0CFGR1 寄存器中的）CKOUTDIV 分频器进行设置，以下为设置公式：

$$((CKOUTDIV + 1) \times T_{SYSCLK}) < T_{Manchester\ clock} < (2 \times CKOUTDIV \times T_{SYSCLK})$$

SPI 码流在第一次检测到时钟输入信号（有效的上升沿/下降沿）后同步。

注： *当收发器尚未被同步时，时钟缺失标志会置 1，且无法由（DFSDM_FLT0ICR 寄存器中）CLRCKABF[y] 位清零。*

图 80. 曼彻斯特编码首次转换（曼彻斯特同步）



外部串行时钟频率测量

通过测量通道串行时钟输入频率可提供来自外部 ΣΔ 调制器的实时数据速率，这对应用目的很重要。

外部串行时钟输入频率可以通过在一个转换持续时间内对 DFSDM 时钟（$f_{DFSDMCLK}$）进行计数的定时器来测量。计数在转换触发（常规或注入）后的第一个输入数据时钟处开始，并在转换结束（转换结束标志置 1）前的最后一个输入数据时钟处结束。转换完成（JEOCF =1 或 REOCF = 1）时，会在寄存器 DFSDM_FLTxCNVTIMR 的计数器 CNVCNT[27:0] 中更新每次转换的时间（第一个串行采样和最后一个串行采样之间的时间）。然后用户可以根据数字滤波器设置（FORD、FOSR、IOSR、FAST）计算数据速率。只有在滤波器被旁路（FOSR = 0，只有积分器有效，DFSDM_FLTxCNVTIMR 寄存器 CNVCNT[27:0] = 0）时，才会停止外部串行频率测量。

对于并行数据输入（*第 15.4.6 节：并行数据输入*），测量的频率为一次转换期间内的平均输入数据速率。

注： *转换被中断（例如，通过禁止/使能所选通道）时，还会在 CNVCNT[27:0] 中对中断时间进行计数。因此，为得到正确的转换时间结果，建议不要中断转换。*

转换时间：

FAST = 0 时的注入转换或常规转换（或者 FAST=1 时的首次转换）：

对于 $Sinc^x$ 滤波器 (x =1..5)：

$t = CNVCNT/f_{DFSDMCLK} = [F_{OSR} * (I_{OSR}-1 + F_{ORD}) + F_{ORD}] / f_{CKIN}$

对于 FastSinc 滤波器：

$t = CNVCNT/f_{DFSDMCLK} = [F_{OSR} * (I_{OSR}-1 + 4) + 2] / f_{CKIN}$

FAST = 1 时的常规转换（首次转换除外）：

对于 $Sinc^x$ 和 FastSinc 滤波器：

$t = CNVCNT/f_{DFSDMCLK} = [F_{OSR} * I_{OSR}] / f_{CKIN}$

如果 F_{OSR} = FOSR[9:0] + 1 = 1（滤波器旁路，仅积分器有效），则：

$t = I_{OSR} / f_{CKIN}$（... 但是 CNVCNT = 0）

其中：

- f_{CKIN} 为（给定通道 CKINy 引脚上的）通道输入时钟频率或（并行数据输入时的）输入数据速率
- F_{OSR} 为滤波器过采样率：F_{OSR} = FOSR[9:0] + 1（请参见 DFSDM_FLTxFCR 寄存器）
- I_{OSR} 为积分器过采样率：I_{OSR} = IOSR[7:0] + 1（请参见 DFSDM_FLTxFCR 寄存器）
- F_{ORD} 为滤波器阶数：F_{ORD} = FORD[2:0]（请参见 DFSDM_FLTxFCR 寄存器）

通道偏移设置

每个通道都有自身的偏移设置（在寄存器中），该偏移值最终会从给定通道的各个转换结果（注入或常规）中减去。在数据右移位后执行偏移校正。偏移值以 24 位有符号值的形式存储在 DFSDM_CHyCFGR2 寄存器的 OFFSET[23:0] 字段中。

数据右移位

要使结果与 24 位值对齐，每个通道均定义了一系列右移位，这些右移位将应用于给定通道的各个转换结果（注入或常规）。右移位数存储在 DFSDM_CHyCFGR2 寄存器的 DTRBS[4:0] 位中。

右移位将结果四舍五入为最接近的整数值。移位结果的符号保留，以便得到有效 24 位有符号格式的结果数据。

脉冲跳过器

脉冲跳过器的用途是为给定数量的输入通道实现类似于延迟线的行为。这样，可以使来自输入串行数据流（仅串行流）的给定数量采样在进入滤波器前被丢弃。该数据丢弃操作是通过跳过给定数量的采样输入时钟脉冲来实现的（滤波器将忽略给定数量的串行数据采样）。采样时钟由脉冲跳过器功能进行门控，用以获得给定数量的时钟脉冲。跳过给定数量的时钟脉冲后，将继续为后面的输入数据进行滤波。与非跳过数据流不同的是，此操作可使滤波器的最终输出采样（及后续采样）基于较新的输入数据进行计算。最终采样会略显超前——因为是基于较新的输入采样（而非“非跳过”采样）计算的。由于跳过的输入数据采样势必会由后面的输入数据采样替代，因此最终的“跳过采样”会稍后进行转换。最终的数据缓冲区行为（跳过和非跳过输出数据缓冲区比较）显示非跳过数据流稍有延迟——两个数据缓冲区都将发生相移。

时钟跳过是基于名为 MCHDLY 的模块（多通道延迟模块）实现的，此模块添加在 DFSDM 的顶层（请参见*图 81*）。此 MCHDLY 模块通过 MCHDLYCR 寄存器（请参见*第 8.2.10 节：*

DFSDM 多通道延迟控制寄存器 (SYSCFG_MCHDLYCR)）进行控制，该寄存器中包含所有的 MCHDLY 控制位。

通过使用 MCHDLY 模块，DFSDM 可用于最多 6 个数字麦克风的波束赋形。在波束赋形模式下，数字麦克风（或 ∑Δ 调制器）的时钟由 DFSDM2 时钟输出（*图 81* 中的 dfsdm2_ckout 信号）提供。此时钟输出信号随后被分配给（请参见*图 81*）：

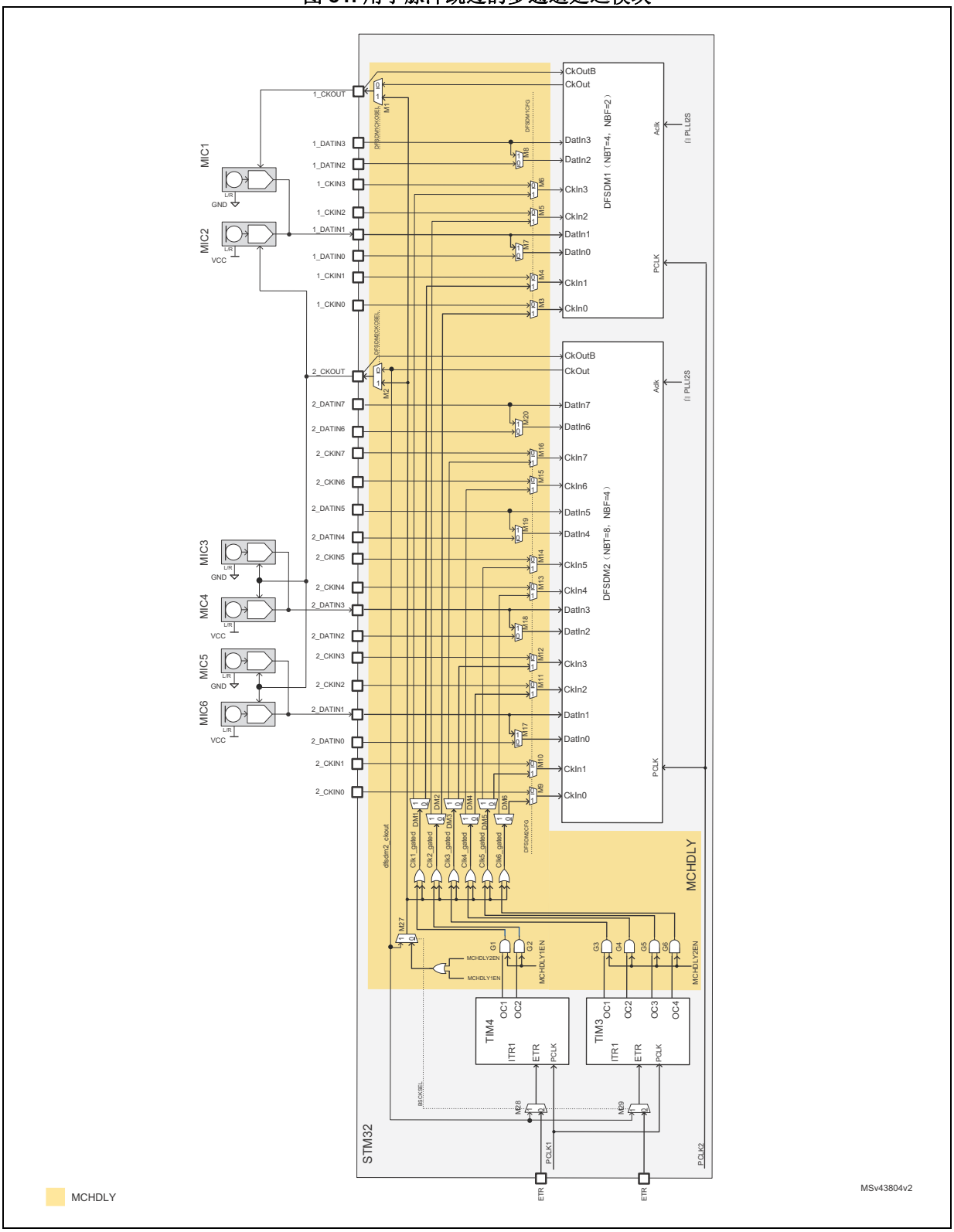
- 或门（用于通过时钟屏蔽/门控实现脉冲跳过）
- 两个定时器的触发输入 (ETR)：TIM4 和 TIM3（用于定义将跳过的脉冲数量）
- 连接到 M1 和 M2 多路复用器的 DFSDM1_CKOUT 和 DFSDM2_CKOUT 引脚输出（用于在 DFSDMx 引脚上生成输出时钟信号）

或门用于跳过提供给 DFSDM 的输入串行时钟脉冲，从而在相应的输入通道上生成延迟。

此时钟门控由两个定时器（TIM4 和 TIM3）控制。这两个定时器在单触发模式下进行编程，用以生成指定长度的屏蔽脉冲，从而对所需数量的时钟脉冲进行门控。

在*图 81* 中，MIC1 通过另一个时钟输出引脚（DFSDM1_CKOUT 引脚）接收其串行时钟，该引脚可提供来自 DFSDM2 或 DFSDM1 CKOUT 信号的时钟输出。当禁止 DFSDM2_CKOUT 并使能 DFSDM1_CKOUT 时，此配置适用于只有一个麦克风 (MIC1) 的低功耗用例（例如语音检测）；当使能 DFSDM2_CKOUT 时，此配置还允许将 MIC1 用于波束赋形用例。

图 81. 用于脉冲跳过的多通道延迟模块



DFSDM 应进行如下配置：

- 对于所有通道，必须将 CHINSEL 设为 0（从同一通道的各引脚获取通道数据）
- 对于所有通道，必须将 SPICKSEL 设为 0（以选择外部 CKINy 作为输入时钟）

TIM3 和 TIM4 应进行如下配置：

- 当无需门控时，TIM4 输出 (OC[2:1]) 和 TIM3 输出 (OC[4:1]) 在未激活模式下必须为低电平。当应用程序需要对麦克风提供延迟时，应将定时器编程为单脉冲模式 (OPM)。在此模式下，定时器允许计数器启动以响应 ETRx（或 ITRx）输入上出现的上升沿，并生成一段长度可编程的脉冲。
- 当 DFSDM2 生成比特流时钟时，必须使用 ETRx 输入。
- 定时器使用的参考时钟可以是其 APB 时钟或 APB 时钟的 2 倍或 4 倍。频率越高，参考时钟越好。
  当 TIM3 和 TIM4 通过 ETRx 输入与 DFSDM2 搭配使用时，定时器参考时钟频率必须至少为输入比特流时钟频率的 12 倍（前提是比特流时钟的占空比为 50%）。这是因为 ETRx 输入的传播延迟较高：长达 5 个定时器参考时钟周期。
- 遵循上述规则将有效防止违反时序。

如果需要同步 DFSDM1 和 DFSDM2 滤波器转换（例如，波束赋形应用），应按照以下顺序操作（请参见*图 81*）：

- 使能 DFSDM 所使用的音频时钟
- 编程 DFSDM 以使用 CkIn
- 使用 DFSDM2 CkOut 将 DFSDM2 音频时钟注入 CkIn
- 阻止 DFSDM2 CkOut 信号：位 BSCKSEL=0
- 启动所有滤波器转换（使用的全部 DFSDM1 和 DFSDM2 滤波器）
- 释放 DFSDM2 CkOut 信号：位 BSCKSEL=1

图 82. 脉冲跳过器操作

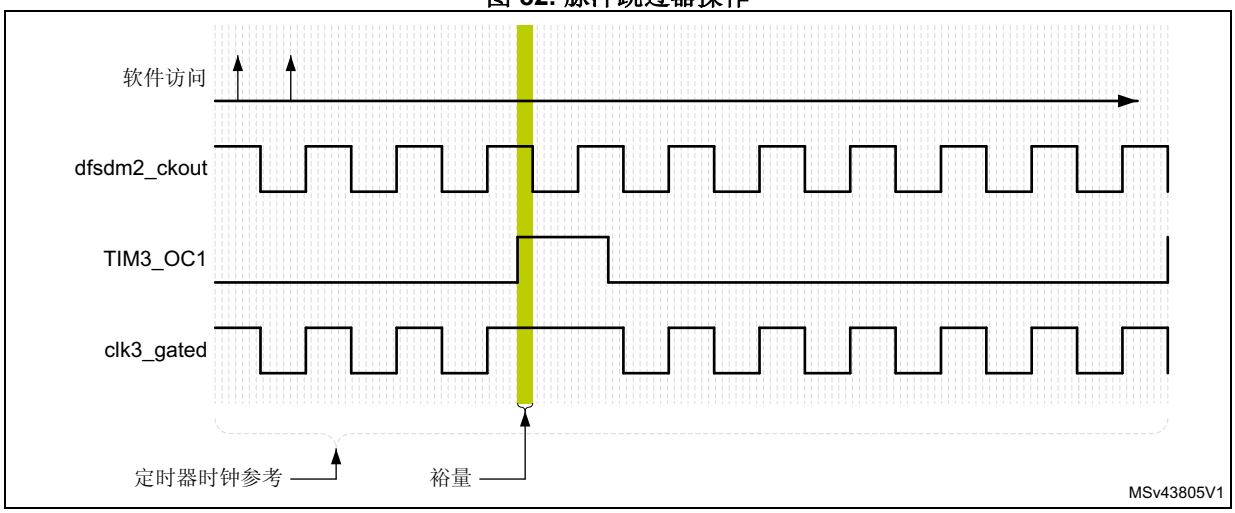


表 92. 多路解复用器 (DM[6:1]) 操作

控制	输出 0	输出 1
0	输入	L
1	L	输入

表 93. 波束赋形应用案例

用例	TIM4	TIM3	DFSDM2	DFSDM1	DFSDM2_CKOSEL	DFSDM2_CFG	DFSDM2_CK37SEL	DFSDM2_CK26SEL	DFSDM2_CK15SEL	DFSDM2_CK04SEL	DFSDM2_D6SEL	DFSDM2_D4SEL	DFSDM2_D2SEL	DFSDM2_D0SEL	MCHDLYEN2	DFSDM1_CKOSEL	DFSDM1_CFG	DFSDM1_CK13SEL	DFSDM1_CK02SEL	DFSDM1_D2SEL	DFSDM1_D0SEL	MCHDLYEN1	BSCKSEL
原理图上的多路复用器/逻辑门	-	-	-	-	M2	M[16:9]	DM3	DM4	DM5	DM6	M20	M19	M18	M17	G[6:3]	M1	M[6:3]	DM1	DM2	M8	M7	G[2:1]	M[29:27]
Speech recognition_1	关闭	关闭	关闭	开启	x	x	x	x	x	x	x	x	x	x	0	0	x	x	x	x	x	0	0
Beamforming4_ DFSDM_1_3	关闭	开启	开启	关闭	1	1	0	0	0	0	x	x	1	1	1	0	x	x	x	x	x	0	1
Beamforming4_ DFSDM_3_5	关闭	开启	开启	关闭	1	1	0	0	1	1	x	1	1	x	1	0	x	x	x	x	x	0	1
Beamforming4_ DFSDM_5_7	关闭	开启	开启	关闭	1	1	1	1	1	1	1	1	x	x	1	0	x	x	x	x	x	0	1
Beamforming4_ DFSDM_1_7	关闭	开启	开启	关闭	1	1	1	1	0	0	1	x	x	1	1	0	x	x	x	x	x	0	1
Beamforming6_ DFSDM_1_3_1	开启	开启	开启	开启	1	1	0	0	0	0	x	x	1	1	1	1	1	0	0	x	1	1	1
Beamforming6_ DFSDM_1_3_3	开启	开启	开启	开启	1	1	0	0	0	0	x	x	1	1	1	1	1	1	1	1	x	1	1

15.4.5 配置输入串行接口

必须为输入串行接口配置以下参数：

- **输出时钟预分频器**。提供了可编程预分频器，用来从 DFSDM 时钟生成输出时钟 (2 - 256)。它由 DFSDM_CH0CFGR1 寄存器中的 CKOUTDIV[7:0] 位定义。
- **串行接口类型和输入时钟相位。**选择 SPI 或曼彻斯特编码以及输入时钟的采样边沿。它由 DFSDM_CHyCFGR1 寄存器中的 SITP [1:0] 位定义。
- **输入时钟源。**来自 CKINy 引脚的外部源或来自 CKOUT 引脚的内部源。它由 DFSDM_CHyCFGR1 寄存器中的 SPICKSEL[1:0] 字段定义。
- **最终数据右移位。**定义最终数据右移位，以将结果与 24 位值对齐。它由 DFSDM_CHyCFGR2 寄存器中的 DTRBS[4:0] 定义。
- **每个通道的通道偏移。**定义给定串行通道的模拟偏移（所连接外部 ΣΔ 调制器的偏移）。它由 DFSDM_CHyCFGR2 寄存器中的 OFFSET[23:0] 位定义。
- **每个通道的短路检测器和时钟缺失使能。**用于在 DFSDM_CHyCFGR1 寄存器中使能或禁止给定串行通道的短路检测器（通过 SCDEN 位）和时钟缺失检测（通过 CKABEN 位）。
- **模拟看门狗滤波器和短路检测器阈值设置。**用于配置通道模拟看门狗滤波器参数和通道短路检测器参数。在 DFSDM_CHyAWSCDR 寄存器中定义这些配置。

15.4.6 并行数据输入

每个输入通道均为 16 位并行数据输入提供一个寄存器（除了串行数据输入外）。每个 16 位并行输入只能源自内部数据源：

- 直接 CPU/DMA 写操作。

通过 DFSDM_CHyCFGR1 寄存器的 DATMPX[1:0] 字段选择是对给定通道使用串行还是并行数据输入。DATMPX[1:0] 中还定义了并行数据源：直接由 CPU/DMA 进行的写操作。

每个通道均包含一个 32 位数据输入寄存器 DFSDM_CHyDATINR，在该寄存器中可写入 16 位数据。数据采用 16 位有符号格式。这些数据还可用作接受 16 位并行数据的数字滤波器的输入。

如果选择串行数据输入 (DATMPX[1:0] = 0)，则 DFSDM_CHyDATINR 寄存器被写保护。

来自存储器的输入（直接由 CPU/DMA 写入）

对于由 CPU 或 DMA (DATMPX[1:0] = 2) 直接写入到 DFSDM_CHyDATINR 寄存器中的数据，可以将其用作数据输入，以处理来自存储器或外设的数字数据流。

数据可由 CPU 或 DMA 写入到 DFSDM_CHyDATINR 寄存器中：

1. **CPU 数据写入：**

   输入数据直接由 CPU 写入到 DFSDM_CHyDATINR 寄存器中。

2. **DMA 数据写入：**

   DMA 应配置为存储器到存储器传输模式，以将存储器缓冲区中的数据传输至 DFSDM_CHyDATINR 寄存器中。目标存储器地址是 DFSDM_CHyDATINR 寄存器的地址。数据以 DMA 传输速度从存储器传输至 DFSDM 并行输入。

   此 DMA 不同于读取 DFSDM 转换结果所使用的 DMA。这两个 DMA 可同时使用，第一个 DMA（配置为存储器到存储器传输）用于输入数据写入，第二个 DMA（配置为外设到存储器传输）用于数据结果读取。

对 DFSDM_CHyDATINR 的访问可为 16 位或 32 位宽，允许分别在一次写操作中加载一个或两个采样。32 位输入数据寄存器 (DFSDM_CHyDATINR) 可用一个或两个 16 位数据采样进行填充，具体取决于在 DFSDM_CHyCFGR1 寄存器的 DATPACK[1:0] 字段中定义的数据封装操作模式：

1. **标准模式 (DATPACK[1:0] = 0)：**

   只有一个样本存储在 DFSDM_CHyDATINR 寄存器的 INDAT0[15:0] 字段中，其用作通道 y 的输入数据。高 16 位 (INDAT1[15:0]) 被忽略，且被写保护。数字滤波器必须执行一次输入采样（通过 INDAT0[15:0]），以便清空由 CPU/DMA 填充后的数据寄存器。该模式与对 DFSDM_CHyDATINR 寄存器的 16 位 CPU/DMA 访问结合使用，以在每次写操作时加载一个采样。

2. **交错模式 (DATPACK[1:0] = 1)：**

   DFSDM_CHyDATINR 寄存器被用作两个采样缓冲区。第一个采样存储在 INDAT0[15:0] 中，第二个采样存储在 INDAT1[15:0] 中。数字滤波器必须通过通道 y 执行两次输入采样，以清空 DFSDM_CHyDATINR 寄存器。该模式与对 DFSDM_CHyDATINR 寄存器的 32 位 CPU/DMA 访问结合使用，以在每次写操作时加载两个采样。

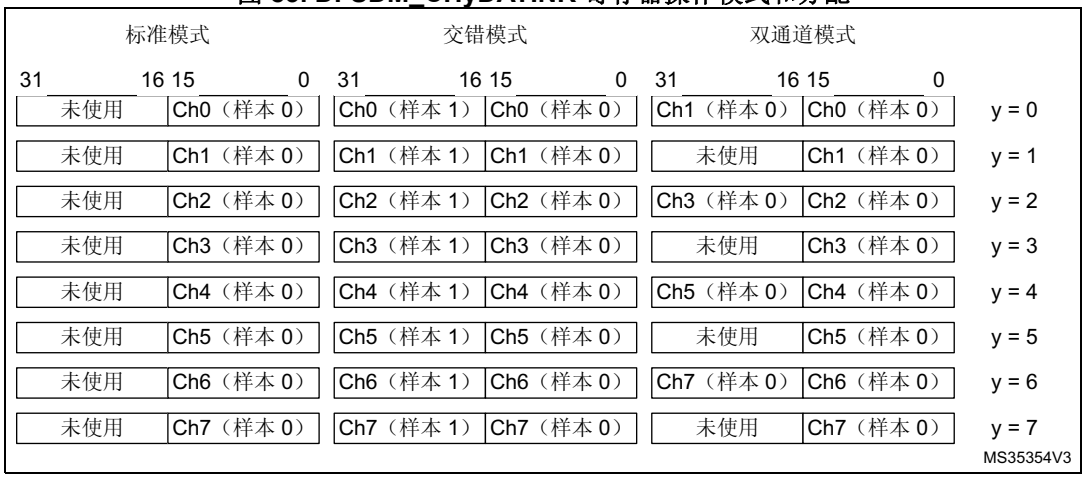
3. **双通道模式 (DATPACK[1:0] = 2)：**

   将两个采样写入到 DFSDM_CHyDATINR 寄存器中。INDAT0[15:0] 中的数据用于通道 y，INDAT1[15:0] 中的数据用于通道 y+1。INDAT1[15:0] 中的数据是自动复制的下一 (y+1) 通道数据寄存器 DFSDM_CH[y+1]DATINR 的 INDAT0[15:0]。数字滤波器必须执行两次采样（一次来自通道 y，另一次来自通道 (y+1)），以清空 DFSDM_CHyDATINR 寄存器。

   只能针对偶数通道数（y = 0、2、4、6）进行双通道模式设置 (DATPACK[1:0] = 2)。如果将奇数通道（y = 1、3、5、7）设为双通道模式，则对于该通道，INDAT0[15:0] 和 INDAT1[15:0] 部分均被写保护。如果偶数通道被设为双通道模式，则下一奇数通道必须设为标准模式 (DATPACK[1:0] = 0)，以便与偶数通道正确配合。

有关 DFSDM_CHyDATINR 寄存器数据模式和通道数据样本分配，请参见*图 83*。

图 83. DFSDM_CHyDATINR 寄存器操作模式和分配



首先使能所选输入通道（通道 y），以便进行数据采集（启动通道 y 转换），然后必须向 DFSDM_CHyDATINR 寄存器进行写操作，以加载一个或两个采样。否则，在下次处理时会丢失写入的数据。

例如：对于单次转换和交错模式，在启动单次转换前，不要开始向 DFSDM_CHyDATINR 写入成对数据样本（开始转换前 DFSDM_CHyDATINR 中存在的数据均会被丢弃）。

15.4.7 通道选择

存在 8 个复用通道，可选择用于注入通道组转换和/或常规通道转换。

注入通道组可以选择 8 个通道中的任一通道或其全部。DFSDM_FLTxJCHGR 寄存器中的 JCHG[7:0] 选择注入组通道，其中 JCHG[y] = 1 表示已选择通道 y。

注入转换可工作在扫描模式 (JSCAN = 1) 或单个模式 (JSCAN = 0) 下。在扫描模式下，每个所选通道都会依次转换。最低通道（通道 0，如果已选择）最先转换，紧接着转换下一个较高通道，直至 JCHG[7:0] 选择的所有通道均被转换。在单个模式 (JSCAN = 0) 下，只会转换所选的一个通道，通道选择会移至下一通道。JSCAN = 0 时，对 JCHG[7:0] 进行写入操作会将通道选择复位为选择的最低通道。

注入转换可由软件或触发器启动。此类转换从不会被常规转换中断。

常规通道只选择 8 个通道的其中一个。DFSDM_FLTxCR1 寄存器中的 RCH[2:0] 指示所选通道。

常规转换只能由软件启动（不能由触发器启动）。请求注入转换时，会暂时中断连续常规转换序列。

对已禁止通道（DFSDM_CHyCFGR1 寄存器中的 CHEN = 0）执行转换会导致转换将永远不会结束，这是因为未提供输入数据的原因（无时钟信号）。在这种情况下，需要使能给定通道（DFSDM_CHyCFGR1 寄存器中的 CHEN = 1），或停止转换（通过 DFSDM_FLTxCR1 寄存器的 DFEN = 0）。

15.4.8 数字滤波器配置

DFSDM 包含 $Sinc^x$ 类型数字滤波器实现。此 $Sinc^x$ 滤波器执行输入数字数据流滤波，这会导致减小输出数据速率（抽取）以及增大输出数据分辨率。可对 $Sinc^x$ 数字滤波器进行配置，以达到所需输出数据速率和所需输出数据分辨率。可配置参数有：

- 滤波器阶数/类型：（请参见 DFSDM_FLTxFCR 寄存器中的 FORD[2:0] 位）：
  - FastSinc
  - $Sinc^1$
  - $Sinc^2$
  - $Sinc^3$
  - $Sinc^4$
  - $Sinc^5$
- 滤波器过采样/抽取率：（请参见 DFSDM_FLTxFCR 寄存器中的 FOSR[9:0] 位）：
  - FOSR = 1-1024 —— 适用于 FastSinc 滤波器和 $Sinc^x$ 滤波器 x = F_{ORD} = 1..3
  - FOSR = 1-215 —— 适用于 $Sinc^x$ 滤波器 x = F_{ORD} = 4
  - FOSR = 1-73 —— 适用于 $Sinc^x$ 滤波器 x = F_{ORD} = 5

滤波器支持以下传递函数（H 域中的冲激响应）：

- $Sinc^x$ 滤波器类型： $$H(z) = \left(\frac{1 - z^{-FOSR}}{1 - z^{-1}}\right)^x$$
- FastSinc 滤波器类型： $$H(z) = \left(\frac{1 - z^{-FOSR}}{1 - z^{-1}}\right)^2 \cdot (1 + z^{-(2 \cdot FOSR)})$$

图 84. 示例：Sinc^3 滤波器响应

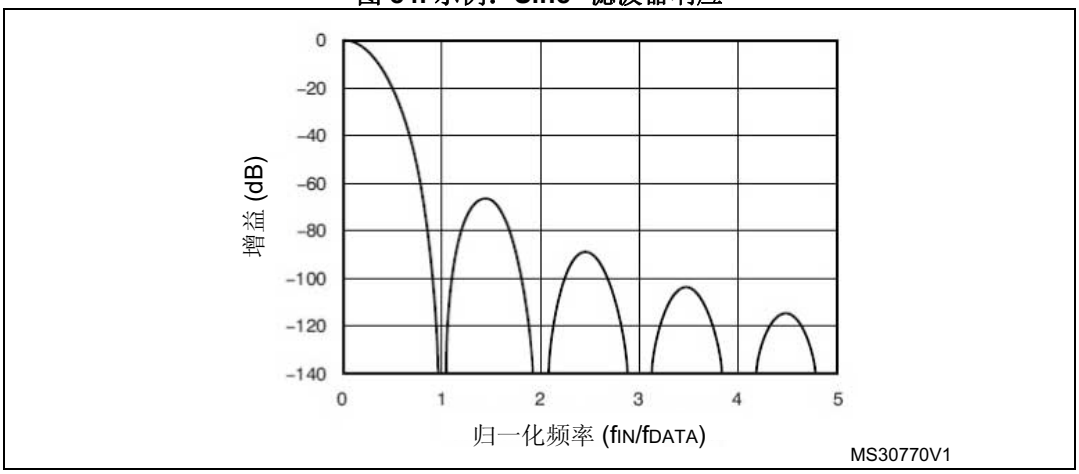


表 94. 滤波器最大输出分辨率（来自滤波器输出的峰值数据值），基于某些 FOSR 值

FOSR	Sinc^1	Sinc^2	FastSinc	Sinc^3	Sinc^4	Sinc^5
x	+/- x	+/- x^2	+/- 2x^2	+/- x^3	+/- x^4	+/- x^5
4	+/- 4	+/- 16	+/- 32	+/- 64	+/- 256	+/- 1024
8	+/- 8	+/- 64	+/- 128	+/- 512	+/- 4096	-
32	+/- 32	+/- 1024	+/- 2048	+/- 32768	+/- 1048576	+/- 33554432
64	+/- 64	+/- 4096	+/- 8192	+/- 262144	+/- 16777216	+/- 1073741824
128	+/- 128	+/- 16384	+/- 32768	+/- 2097152	+/- 268435456	
256	+/- 256	+/- 65536	+/- 131072	+/- 16777216	在满量程输入的条件下，结果会溢出（> 32位有符号整数值）	
1024	+/- 1024	+/- 1048576	+/- 2097152	+/- 1073741824		

有关 Sinc 滤波器类型属性和使用的更多信息，建议了解下数字滤波器的理论知识（可从 Internet 上下载更多资源）。

15.4.9 积分器单元

积分器会对来自数字滤波器的数据进一步提升抽取率和分辨率。对于来自滤波器的给定数量的数据采样，积分器对其进行简单的求和。

积分器过采样率参数定义了将多少个数据的和作为积分器的一个数据输出。IOSR 可设置为介于 1-256 范围内（请参见 DFSDM_FLTxFCR 寄存器中的 IOSR[7:0] 位说明）。

表 95. 积分器最大输出分辨率（来自积分器输出的峰值数据值），基于某些 IOSR 值，FOSR = 256 以及 $Sinc^3$ 滤波器类型（最大数据）

IOSR	$Sinc^1$	$Sinc^2$	FastSinc	$Sinc^3$	$Sinc^4$	$Sinc^5$
x	+/- FOSR. x	+/- $FOSR^2$.x	+/- 2.$FOSR^2$.x	+/- $FOSR^3$.x	+/- $FOSR^4$.x	+/- $FOSR^5$.x
4	-	-	-	+/- 67 108 864	-	-
32	-	-	-	+/- 536 870 912	-	-
128	-	-	-	+/- 2 147 483 648	-	-
256	-	-	-	+/- 2^{32}	-	-

15.4.10 模拟看门狗

模拟看门狗用于在达到或超出给定阈值上限或者达到或低于给定阈值下限时，触发外部信号（断路或中断）。随即可调用中断/事件/断路生成。

每个模拟看门狗都会根据（DFSDM_FLTxCR1 寄存器中）AWFSEL 位设置监视串行数据接收器输出（在每个通道上模拟看门狗滤波器之后）或数据输出寄存器（当前注入或常规转换结果）。是否需要通过模拟看门狗 x 监视的输入通道可以通过 DFSDM_FLTxCR2 寄存器中的 AWDCH[7:0] 选择。

输入通道上的模拟看门狗转换与标准转换无关。在标准转换模式下，模拟看门狗在每个输入通道上使用自己的滤波器和信号处理，与注入或常规主转换无关。在连续转换模式下，对所选输入通道执行模拟看门狗转换，以便在注入或常规主转换暂停（RCIP = 0，JCIP = 0）时也仍能对通道进行监视。

阈值上限和阈值下限寄存器与给定数据值（通过 DFSDM_FLTxAWHTR 寄存器的 AWHT[23:0] 位和 DFSDM_FLTxAWLTR 寄存器的 AWLT[23:0] 位进行设置）进行比较。

可以选择 2 种情况将阈值寄存器与数据值进行比较：

- 选择 1：在这种情况下，输入数据取自最终输出数据寄存器 (AWFSEL = 0)。该选择具有以下特性：
  - 高输入数据分辨率（高达 24 位）
  - 响应时间慢——不适用于过流检测等快速响应应用
  - 为了进行比较，最终数据需要经过移位和偏移数据校正
  - 最终数据仅在执行常规或注入主转换后才可用
  - 可在并行输入数据源（DFSDM_CHyCFGR1 寄存器中的 DATMPX[1:0] ≠ 0）情况下使用

- 选择 2：在这种情况下，输入数据取自任一串行数据接收器输出 (AWFSEL = 1)。该选择具有以下特性：
  - 输入串行数据通过专用模拟看门狗 $Sinc^x$ 通道滤波器进行处理，滤波器的过采样率 (1..32) 和滤波器阶数 (1..3) 可以配置（请参见 DFSDM_CHyAWSCDR 寄存器中的 AWFOSR[4:0] 和 AWFORD[1:0] 位设置）
  - 较低分辨率（最高为 16 位）
  - 响应时间快——适用于需要快速响应的应用，如过流/过压检测
  - 数据以连续模式提供，与常规或注入主转换活动无关

进行输入通道监视 (AWFSEL = 1) 时，用于跟阈值进行比较的数据取自 AWDCH[7:0] 字段（DFSDM_FLTxCR2 寄存器）所选的通道。将每个所选通道滤波器结果与一个阈值对 (AWHT[23:0] / AWLT[23:0]) 进行比较。此时，只有高 16 位 (AWHT[23:8] / AWLT[23:8]) 会定义与模拟看门狗滤波器输出进行比较的 16 位阈值，因为来自模拟看门狗滤波器的数据的分辨率最高为 16 位。在这种情况下 (AWFSEL=1)，不对 AWHT[7:0] / AWLT[7:0] 位进行比较。

在 DFSDM_CHyAWSCDR 寄存器中设置各个输入通道的模拟看门狗滤波器配置的参数（滤波器阶数 AWFORD[1:0] 和滤波器过采样率 AWFOSR[4:0]）。

每个输入通道都配有自身的比较器，用于比较模拟看门狗数据（来自模拟看门狗滤波器）和模拟看门狗阈值 (AWHT/AWLT)。选择多个通道（DFSDM_FLTxCR2 寄存器的字段 AWDCH[7:0]）时，可同时接收多个比较请求。在这种情况下，首先管理编号最小的通道请求，然后继续管理编号较大的所选通道。对于每个通道，结果均可记录在单独的标志（DFSDM_FLTxAWSR 寄存器的位域 AWHTF[7:0] 和位域 AWLTF[7:0]）中。每个通道请求均在 8 个 DFSDM 时钟周期内执行。因此，各个通道的带宽被限制为 8 个 DFSDM 时钟周期（AWDCH[7:0] = 0xFF 时）。因为最大输入通道采样时钟频率为 DFSDM 时钟频率的四分之一，因此在该输入时钟速度下，不能对模拟看门狗配置 AWFOSR = 0（模拟看门狗滤波器被旁路）。因此，用户必须根据输入采样时钟速度和 DFSDM 频率正确配置监视的通道数和模拟看门狗滤波器参数。

给定通道 y 的模拟看门狗过滤器数据可通过固件从 DFSDM_CHyWDATR 寄存器 WDATA[15:0] 字段中读取。如果 DFSDM_CHyCFGR1 寄存器中的 CHEN = 1，则该模拟看门狗滤波器数据被连续转换，其数据速率由模拟看门狗滤波器设置和通道输入时钟频率给定。

模拟看门狗滤波器转换的工作方式类似于常规的快速连续转换（无积分器）。模拟看门狗滤波器输出（通道输入时钟频率为 f_{CKIN}）一个结果所需的串行采样数量：

首次转换：

对于 $Sinc^x$ 滤波器 (x = 1..5)：采样数 = [F_{OSR} * F_{ORD} + F_{ORD} + 1]

对于 FastSinc 滤波器：采样数 = [F_{OSR} * 4 + 2 + 1]

后续转换：

对于 $Sinc^x$ 和 FastSinc 滤波器：采样数 = [FOSR * IOSR]

其中：

F_{OSR} .....滤波器过采样率：F_{OSR} = AWFOSR[4:0] + 1（请参见 DFSDM_CHyAWSCDR 寄存器）

F_{ORD} .....滤波器阶数：F_{ORD} = AWFORD[1:0]（请参见 DFSDM_CHyAWSCDR 寄存器）

对于输出数据寄存器监视 (AWFSEL = 0)，在完成最终数据的右移位和偏移校正（请参见 DFSDM_CHyCFGR2 寄存器的 OFFSET[23:0] 和 DTRBS[4:0] 字段）后进行比较。在由（DFSDM_FLTxCR2 寄存器中）AWDCH[7:0] 字段所选通道的每次注入或常规转换结束后，进行比较。

模拟看门狗事件的状态反映在 DFSDM_FLTxAWSR 寄存器中，该寄存器中锁定了给定事件。AWHTF[y] = 1 标志指示在通道 y 上是否超出了 AWHT[23:0] 值。AWLTF[y] = 1 标志指示在通道 y 上是否超出了 AWLT[23:0] 值。对于 DFSDM_FLTxAWSR 寄存器中锁存的事件，将通过向 DFSDM_FLTxAWCFR 寄存器中的相应清零位 CLRAWHTF[y] 或 CLRAWLTF[y] 写入“1”来清除。

模拟看门狗的全局状态通过 DFSDM_FLTxISR 寄存器中的 AWDF 标志位来反映（其用于快速检测中断源）。AWDF = 1 表示至少发生了一个看门狗事件（至少对于一个通道，AWHTF[y] = 1 或 AWLTF[y] = 1）。所有 AWHTF[7:0] 和 AWLTF[7:0] 均清零时，AWDF 位才清零。

可将模拟看门狗事件分配至断路输出信号。可将四个断路输出分配至超出阈值上限或低于阈值下限的事件 (dfsdm_break[3:0])。通过 DFSDM_FLTxAWHTR 和 DFSDM_FLTxAWLTR 寄存器中的 BKAWH[3:0] 和 BKAWL[3:0] 字段将断路信号分配至给定模拟看门狗事件。

15.4.11 短路检测器

使用短路检测器的目的在于，当模拟信号达到饱和值（超出满量程范围）并保持给定的时间时，短路检测器可以在极快的响应时间内发出信号。该特性可检测短路或断路故障（例如过流或过压）。可调用中断/事件/断路生成。

短路检测器的输入数据取自通道收发器输出。

各个输入通道上都有一个递增计数器，用于对串行数据接收器输出上的连续 0 或 1 进行计数。如果接收的数据流发生变化（数据信号从 1 到 0 或从 0 到 1），则会重新启动计数器。如果该计数器达到短路阈值寄存器值（DFSDM_CHyAWSCDR 寄存器中的 SCDT[7:0] 位），则会调用短路事件。每个输入通道都配有短路检测器。通过将 SCDEN 位（DFSDM_CHyCFGR1 寄存器中）置 1，可选择对任一通道进行连续监视，每个通道都有其自己的短路检测器设置（SCDT[7:0] 位、状态位 SCDF[7:0]、状态清零位 CLRSCDF[7:0] 中的阈值）。禁止相应通道 y (CHEN[y] = 0) 时，状态标志 SCDF[y] 还可由硬件清零。

对于各个通道，可将短路检测器事件分配至断路输出信号 dfsdm_break[3:0]。可将四个断路输出分配至短路检测器事件。通过 DFSDM_CHyAWSCDR 寄存器中的 BKSCD[3:0] 字段将断路信号分配至给定通道短路检测器事件。

如果选择了并行输入数据通道（DFSDM_CHyCFGR1 寄存器中的 DATMPX[1:0] ≠ 0），则无法使用短路检测器。

总共有四个断路输出（与模拟看门狗功能共享）。

15.4.12 极值检测器

极值检测器用于采集最终输出数据字的最小值和最大值（峰峰值）。

如果输出数据字高于极值检测器最大值寄存器（DFSDM_FLTxEXMAX 寄存器中的 EXMAX[23:0] 位）中存储的值，则该寄存器将用当前输出数据字值进行更新，且存储其数据的通道位于（DFSDM_FLTxEXMAX 寄存器的）EXMAXCH[2:0] 位中。

如果输出数据字低于极值检测器最小值寄存器（DFSDM_FLTxEXMIN 寄存器中的 EXMIN[23:0] 位）中存储的值，则该寄存器将用当前输出数据字值进行更新，且存储其数据的通道位于（DFSDM_FLTxEXMIN 寄存器的）EXMINCH[2:0] 位中。

最小值和最大值寄存器值可由软件（通过读取给定的 DFSDM_FLTxEXMAX 或 DFSDM_FLTxEXMIN 寄存器）进行刷新。刷新后，极值检测器最小值数据寄存器 DFSDM_FLTxEXMIN 用 0x7FFFFF（最大正值）进行填充，极值检测器最大值寄存器 DFSDM_FLTxEXMAX 用 0x800000（最小负值）进行填充。

在完成右移位和偏移数据校正之后，极值检测器才进行比较。对于每个极值检测器，在（DFSDM_FLTxCR2 寄存器中的）EXCH[7:0] 位中选择计算极值时所涉及的输入通道。

15.4.13 数据单元模块

数据单元模块是整个处理路径的最后一个模块：外部 ΣΔ 调制器——串行收发器——Sinc 滤波器——积分器——数据单元块。

输出数据速率取决于串行数据流速率以及滤波器和积分器设置。最大输出数据速率为：

$$\text{数据速率}[\text{采样/s}] = \frac{f_{CKIN}}{F_{OSR} \cdot (I_{OSR} - 1 + F_{ORD}) + (F_{ORD} + 1)} \quad \text{...FAST = 0， Sincx 滤波器}$$

$$\text{数据速率}[\text{采样/s}] = \frac{f_{CKIN}}{F_{OSR} \cdot (I_{OSR} - 1 + 4) + (2 + 1)} \quad \text{...FAST = 0， FastSinc 滤波器}$$

或

$$\text{数据速率}[\text{采样/s}] = \frac{f_{CKIN}}{F_{OSR} \cdot I_{OSR}} \quad \text{...FAST = 1}$$

并行数据输入时的最大输出数据速率：

$$\text{数据速率}[\text{采样/s}] = \frac{f_{DATAIN\_RATE}}{F_{OSR} \cdot (I_{OSR} - 1 + F_{ORD}) + (F_{ORD} + 1)} \quad \text{...FAST = 0， Sincx 滤波器}$$

或

$$\text{数据速率}[\text{采样/s}] = \frac{f_{DATAIN\_RATE}}{F_{OSR} \cdot (I_{OSR} - 1 + 4) + (2 + 1)} \quad \text{...FAST = 0， FastSinc 滤波器}$$

或

$$\text{数据速率}[\text{采样/s}] = \frac{f_{DATAIN\_RATE}}{F_{OSR} \cdot I_{OSR}} \quad \text{...FAST=1 或任何滤波器旁路情况 } (F_{OSR} = 1)$$

其中：f_{DATAIN_RATE}... 来自 CPU/DMA 的输入数据速率

在该模块中执行最终数据的右移位，因为最终数据宽度为 24 位，而来自处理路径的数据最高可为 32 位。对于每个所选输入通道，此右移位均可在 0-31 位的范围内进行配置（请参见 DFSDM_CHyCFGR2 寄存器的 DTRBS[4:0] 位）。右移位将结果四舍五入为最接近的整数值。移位结果的符号保留，以便得到有效 24 位有符号格式的结果数据。

接下来对结果进行偏移校正。从给定通道的输出数据中减去偏移校正值（存储在寄存器 DFSDM_CHyCFGR2 中的 OFFSET[23:0]）。OFFSET[23:0] 字段中的数据由软件通过相应的校准程序进行设置。

因数字处理过程中的所有操作均在 32 位有符号寄存器上执行，因此为保证结果不会发生溢出，必须满足以下条件：

$FOSR^{FORD} \cdot IOSR <= 2^{31}$ ... 适用于 $Sinc^x$ 滤波器 (x = 1..5)
$2 \cdot FOSR^2 \cdot IOSR <= 2^{31}$ ... 适用于 FastSinc 滤波器

注： *滤波器和积分器旁路时（IOSR[7:0] = 0，FOSR[9:0] = 0），输入数据速率 (f_{DATAIN_RATE}) 必须被限制为能够读取所有输出数据：*
$f_{DATAIN\_RATE} \leq f_{APB}$
其中，f_{APB} 为 DFSDM 外设连接的总线频率。

15.4.14 有符号数据格式

每个 DFSDM 输入串行通道均可连接至一个外部 ΣΔ 调制器。一个外部 ΣΔ 调制器可配有 2 个差分输入（正负），这两个输入可用于差分或单端信号测量。

始终假定 ΣΔ 调制器输出采用有符号格式（来自 ΣΔ 调制器的 0/1 数据流表示值 -1 和 +1）。

寄存器的有符号数据格式：在最终输出数据、模拟看门狗、极值检测器和偏移校正的寄存器中的数据是有符号格式的数据。输出数据字的 msb 表示值的符号（二进制补码格式）。

15.4.15 启动转换

注入转换可使用以下方法启动：

- 软件：向 DFSDM_FLTxCR1 寄存器中的 JSWSTART 写入“1”。
- 触发：JEXTSEL[2:0] 在 JEXTEN 激活触发时选择触发信号，同时选择有效边沿（请参见 DFSDM_FLTxCR1 寄存器）。
- 如果 JSYNC = 1，则使用 DFSDM_FLT0 同步启动：对于 DFSDM_FLTx (x > 0)，注入转换会在 DFSDM_FLT0 中自动启动；注入转换由软件启动（DFSDM_FLT0CR2 寄存器中的 JSWSTART = 1）。DFSDM_FLTx (x > 0) 中的每个注入转换都始终根据其本地配置设置（JSCAN 和 JCHG 等）执行。

如果使能扫描转换（位 JSCAN = 1），则每次触发注入转换时，注入组（DFSDM_FLTxJCHGR 寄存器中的 JCHG[7:0] 位）的全部所选通道将从最低通道（通道 0，如果已选择）开始，被依次转换。

如果禁止扫描转换（位 JSCAN = 0），则每次触发注入转换时，只会转换注入组（DFSDM_FLTxJCHGR 寄存器中的 JCHG[7:0] 位）的所选通道之一，通道选择随即会移至下一通道。JSCAN = 0 时，对 JCHG[7:0] 位进行写操作会将通道选择置为选择的最低注入通道。

在给定时间内只能进行一次注入转换。因此，如果已发出但尚未完成某个注入转换请求，则会忽略启动注入转换的其他请求。

常规转换可使用以下方法启动：

- 软件：向 DFSDM_FLTxCR1 寄存器中的 RSWSTART 写入“1”。
- 如果 RSYNC = 1，则使用 DFSDM_FLT0 同步启动：对于 DFSDM_FLTx (x > 0)，常规转换会在 DFSDM_FLT0 中自动启动；常规转换由软件启动（DFSDM_FLT0CR2 寄存器中的 RSWSTART = 1）。DFSDM_FLTx (x > 0) 中的每个常规转换都始终根据其本地配置设置（RCONT 和 RCH 等）执行。

在给定时间内只能有一个常规转换处于待处理或正在执行状态。因此，如果已发出但尚未完成某个常规转换请求，则会忽略启动常规转换的其他请求。如果常规转换被注入转换中断，或者常规转换在进行注入转换的过程中启动，则常规转换处于待处理状态。此待处理状态的常规转换随即被延迟，并在完成所有注入转换时执行。延迟的常规转换通过 DFSDM_FLTxRDATAR 寄存器的 RPEND 位进行表示。

15.4.16 连续和快速连续模式

如果将 DFSDM_FLTxCR1 寄存器中的 RCONT 置 1，则会在连续模式下执行常规转换。RCONT = 1 表示向 RSWSTART 写入“1”后，重复转换 RCH[2:0] 所选的通道。

可通过向 RCONT 写入“0”停止在连续模式下执行的常规转换。将 RCONT 清零后，会立即停止正在进行的转换。

在连续模式下，可通过将 DFSDM_FLTxCR1 寄存器的 FAST 位置 1 来提高数据速率。此时，如果对一个通道进行连续转换，则滤波器无需用新的刷新数据重新填充，因为滤波器内部数据有效，这些数据是先前通过对连续数据进行采样而得。速度提升与所选的滤波器阶数有关。通过 RSWSTART = 1 启动连续转换后，将充分进行快速模式 (FAST = 1) 下的首次转换（与 FAST = 0 时相同），之后会在较短的间隔内完成各个后续转换。

连续模式下的转换时间：

FAST = 0 时（或 FAST = 1 时的快速转换）：

对于 $Sinc^x$ 滤波器：

$t = CNVCNT/f_{DFSDMCLK} = [F_{OSR} * (I_{OSR}-1 + F_{ORD}) + F_{ORD}] / f_{CKIN}$

对于 FastSinc 滤波器：

$t = CNVCNT/f_{DFSDMCLK} = [F_{OSR} * (I_{OSR}-1 + 4) + 2] / f_{CKIN}$

FAST = 1 时（首次转换除外）：

对于 $Sinc^x$ 和 FastSinc 滤波器：

$t = CNVCNT/f_{DFSDMCLK} = [F_{OSR} * I_{OSR}] / f_{CKIN}$

F_{OSR} = FOSR[9:0] + 1 = 1 时（滤波器旁路，仅积分器有效）：

$t = I_{OSR} / f_{CKIN}$（... 但是 CNVCNT = 0）

连续模式不适用于注入转换。注入转换可通过定时器触发进行启动，以便使用精确时序仿真连续模式。

如果常规连续转换正在进行 (RCONT = 1)，并且如果对请求常规连续转换 (RCONT = 1) 的 DFSDM_FLTxCR1 寄存器进行写访问，则会从下一转换周期重新启动常规连续转换（类似于新的常规连续转换应用于新通道选择，即使 DFSDM_FLTxCR1 寄存器中无更改，亦如此）。

15.4.17 请求优先级

注入转换的优先级高于常规转换。正在进行的常规转换会立即被注入转换请求中断；在注入转换完成后将重新启动此常规转换。

如果某个注入转换处于待处理状态或正在进行，则不能启动其他注入转换：只要（DFSDM_FLTxISR 寄存器中）位 JCIP 为“1”，就会忽略启动注入转换（通过 JSWSTART 或触发）的请求。

同理，如果某个常规转换处于待处理状态或正在进行，则不能启动其他常规转换：只要（DFSDM_FLTxISR 寄存器中）位 RCIP 为“1”，就会忽略启动常规转换（使用 RSWSTART）的请求。

不过，如果在进行某个常规转换时请求注入转换，则常规转换会立即停止，并会启动注入转换。常规转换之后会重新启动，并且此延迟的重新启动会通过位 RPEND 进行表示。

注入转换的优先级高于常规转换，因为注入转换可暂时中断连续的常规转换序列。完成注入转换序列时，如果 RCONT 仍置 1 则会重新启动连续常规转换（RPEND 位将发出基于首次常规转换结果的延迟启动信号）。

通过对 DFSDM 进行相同的写操作启动相关操作时，或者，如果在其他操作结束时有多个操作处于待处理状态，则优先级也至关重要。例如，假设注入转换正在进行 (JCIP = 1)，则对 DFSDM_FLTxCR1 进行单次写操作将向 RSWSTART 写入“1”，以请求常规转换。完成注入序列时，优先级指示接下来执行常规转换，并且通过 RPEND 位指示相应的延迟启动。

15.4.18 在运行模式下的功耗优化

为降低功耗，DFSDM 滤波器和积分器会在未用于转换时自动进入空闲状态（RCIP = 0，JCIP = 0）。

15.5 DFSDM 中断

为提升 CPU 性能，已实现了一组与 CPU 事件发生相关的中断：

- 注入转换结束中断：
  - 由 DFSDM_FLTxCR2 寄存器中的 JEOCIE 位使能
  - 由 DFSDM_FLTxISR 寄存器中的 JEOCF 位指示
  - 通过读取 DFSDM_FLTxJDATAR 寄存器（注入数据）清除
  - 指示哪一个通道发生了转换结束，并在 DFSDM_FLTxJDATAR 寄存器的 JDATACH[2:0] 位中进行报告
- 常规转换结束中断：
  - 由 DFSDM_FLTxCR2 寄存器中的 REOCIE 位使能
  - 由 DFSDM_FLTxISR 寄存器中的 REOCF 位指示
  - 通过读取 DFSDM_FLTxRDATAR 寄存器（常规数据）清除
  - 指示哪一个通道发生了转换结束，并在 DFSDM_FLTxRDATAR 寄存器的 RDATACH[2:0] 位中进行报告
- 注入转换数据溢出中断：
  - 当注入的转换数据未从 DFSDM_FLTxJDATAR 寄存器中读取 （通过 CPU 或 DMA），并被新注入转换覆盖时，会发生此中断
  - 由 DFSDM_FLTxCR2 寄存器中的 JOVRIE 位使能
  - 由 DFSDM_FLTxISR 寄存器中的 JOVRF 位指示
  - 通过向 DFSDM_FLTxICR 寄存器中的 CLRJOVRF 位写入“1”进行清除
- 常规转换数据溢出中断：
  - 当常规转换数据未从 DFSDM_FLTxRDATAR 寄存器中读取（通过 CPU 或 DMA），并被新常规转换覆盖时，会发生此中断
  - 由 DFSDM_FLTxCR2 寄存器中的 ROVRIE 位使能
  - 由 DFSDM_FLTxISR 寄存器中的 ROVRF 位指示
  - 通过向 DFSDM_FLTxICR 寄存器中的 CLRROVRF 位写入“1”进行清除
- 模拟看门狗中断：
  - 当转换数据（输出数据或模拟看门狗滤波器中的数据——具体取决于 DFSDM_FLTxCR1 寄存器中的 AWFSEL 位设置）超出 DFSDM_FLTxAWHTR/DFSDM_FLTxAWLTR 寄存器中的阈值上限或低于相应阈值下限时，会发生此中断
  - 通过（所选通道 AWDCH[7:0] 的）DFSDM_FLTxCR2 寄存器 AWDIE 位使能
  - 由 DFSDM_FLTxISR 寄存器中的 AWDF 位指示

  - 通过 DFSDM_FLTxAWSR 寄存器中的 AWHTF[7:0] 和 AWLTF[7:0] 字段单独指示模拟看门狗阈值上限或下限错误
  - 通过向 DFSDM_FLTxAWCFR 寄存器中的相应 CLRAWHTF[7:0] 或 CLRAWLTF[7:0] 位写入“1”进行清除
- 短路检测器中断：
  - 稳定的数据数超出 DFSDM_CHyAWSCDR 寄存器中的阈值时会发生此中断
  - 通过（通过 DFSDM_CHyCFGR1 寄存器 SCDEN 位选择的通道的）DFSDM_FLTxCR2 寄存器的 SCDIE 位使能
  - 由 DFSDM_FLTxISR 寄存器的 SCDF[7:0] 位指示（还会报告发生短路检测器事件的通道）
  - 通过向 DFSDM_FLTxICR 寄存器中的相应 CLRSCDF[7:0] 位写入“1”进行清除
- 通道时钟缺失中断：
  - CKINy 引脚上出现时钟缺失时会发生此中断（请参见*第 15.4.4 节：串行通道收发器的时钟缺失检测*）
  - 通过（通过 DFSDM_CHyCFGR1 寄存器 CKABEN 位选择的通道的）DFSDM_FLTxCR2 寄存器的 CKABIE 位使能
  - 由 DFSDM_FLTxISR 寄存器中的 CKABF[y] 位指示
  - 通过向 DFSDM_FLTxICR 寄存器中的 CLRCKABF[y] 位写入“1”进行清除

表 96. DFSDM 中断请求

中断事件	事件标志	事件/中断清除方法	中断使能控制位
注入转换结束	JEOCF	读取 DFSDM_FLTxJDATAR	JEOCIE
常规转换结束	REOCF	读取 DFSDM_FLTxRDATAR	REOCIE
注入数据溢出	JOVRF	写入 CLRJOVRF = 1	JOVRIE
常规数据溢出	ROVRF	写入 CLRROVRF = 1	ROVRIE
模拟看门狗	AWDF、AWHTF[7:0]、AWLTF[7:0]	写入 CLRAWHTF[7:0] = 1 写入 CLRAWLTF[7:0] = 1	AWDIE，(AWDCH[7:0])
短路检测器	SCDF[7:0]	写入 CLRSCDF[7:0] = 1	SCDIE、(SCDEN)
通道时钟缺失	CKABF[7:0]	写入 CLRCKABF[7:0] = 1	CKABIE、(CKABEN)

15.6 DFSDM DMA 传输

为降低 CPU 干预，可使用 DMA 传输将转换传输至存储器。注入转换的 DMA 传输通过将 DFSDM_FLTxCR1 寄存器的 JDMAEN 位置 1 使能。常规转换的 DMA 传输通过将 DFSDM_FLTxCR1 寄存器的 RDMAEN 位置 1 使能。

注： *通过 DMA 传输，中断标志在注入或常规转换结束时自动清零（DFSDM_FLTxISR 寄存器中的 JEOCF 或 REOCF 位），因为 DMA 正在读取 DFSDM_FLTxJDATAR 或 DFSDM_FLTxRDATAR 寄存器。*

15.7 DFSDM 通道 y 寄存器 (y=0..7)

15.7.1 DFSDM 通道 y 配置寄存器 (DFSDM_CHyCFGR1)

DFSDM channel y configuration register

该寄存器指定通道 y 所使用的参数。

偏移地址：0x00 + 0x20 * y（y = 0 到 7）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DFSDMEN	CKOUTSRC	Res.	Res.	Res.	Res.	Res.	Res.	CKOUTDIV[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
DATPACK[1:0]		DATMPX[1:0]		Res.	Res.	Res.	CHINSEL	CHEN	CKABEN	SCDEN	Res.	SPICKSEL[1:0]		SITP[1:0]	
rw	rw	rw	rw				rw	rw	rw	rw		rw	rw	rw	rw

位 31 **DFSDMEN**：DFSDM 接口全局使能 (Global enable for DFSDM interface)

0：禁止 DFSDM 接口

1：使能 DFSDM 接口

如果已使能 DFSDM 接口，则该接口将根据使能的 y 通道和使能的 x 滤波器设置（DFSDM_CHyCFGR1 的 CHEN 位和 DFSDM_FLTxCR1 的 DFEN 位）进行工作。通过设置 DFSDMEN = 0 清除数据：

– 所有寄存器 DFSDM_FLTxISR 均被设为复位状态 (x = 0..3)

– 所有寄存器 DFSDM_FLTxAWSR 均被设为复位状态 (x = 0..3)

注： DFSDMEN 仅存在于 DFSDM_CH0CFGR1 寄存器（通道 y = 0）中

位 30 **CKOUTSRC**：输出串行时钟源选择 (Output serial clock source selection)

0：输出时钟的源来自系统时钟

1：输出时钟的源来自音频时钟

只有在（DFSDM_CH0CFGR1 寄存器中）DFSDMEN = 0时，才能修改该值。

注： CKOUTSRC 仅存在于 DFSDM_CH0CFGR1 寄存器（通道 y = 0）中

位 29:24 保留，必须保持复位值。

位 23:16 **CKOUTDIV[7:0]**：输出串行时钟分频器 (Output serial clock divider)

0：禁止输出时钟生成（CKOUT 信号被设为低电平状态）

1 - 255：为 CKOUT 信号定义串行时钟输出的系统时钟分频，范围为 2 - 256（分频器值 = CKOUTDIV + 1）。

CKOUTDIV 还定义了时钟缺失检测的阈值。

只有在（DFSDM_CH0CFGR1 寄存器中）DFSDMEN = 0 时，才能修改该值。

如果 DFSDMEN = 0（DFSDM_CH0CFGR1 寄存器中），则 CKOUT 信号被置为低电平状态（在 DFSDMEN = 0 后的一个 DFSDM 时钟周期内执行设置）。

注： CKOUTDIV 仅存在于 DFSDM_CH0CFGR1 寄存器（通道 y = 0）中

位 15:14 **DATPACK[1:0]**：DFSDM_CHyDATINR 寄存器数据封装模式 (Data packing mode in DFSDM_CHyDATINR register)

0：标准：DFSDM_CHyDATINR 寄存器中的输入数据仅存储在 INDAT0[15:0] 中。要清空 DFSDM_CHyDATINR 寄存器，必须由 DFSDM 滤波器从通道 y 中读取一个采样。

1：交错：DFSDM_CHyDATINR 寄存器中的输入数据存储为两个采样：

– 第一个采样位于 INDAT0[15:0] 中（已分配至通道 y）

– 第二个采样位于 INDAT1[15:0] 中（已分配至通道 y）

要清空 DFSDM_CHyDATINR 寄存器，必须由数字滤波器从通道 y 中读取两个采样（INDAT0[15:0] 部分被读为第一个采样，而 INDAT1[15:0] 部分则被读为下一个采样）。

2：双通道：DFSDM_CHyDATINR 寄存器中的输入数据存储为两个采样：

– 第一个采样位于 INDAT0[15:0] 中（已分配至通道 y）

– 第二个样采样于 INDAT1[15:0] 中（已分配至通道 y+1）

要清空 DFSDM_CHyDATINR 寄存器，必须由数字滤波器从通道 y 中读取第一个样本，且必须由另一数字滤波器从通道 y+1 中读取第二个样本。只有在通道数为偶数（y = 0、2、4、6）时才支持双通道模式，对于通道数为奇数（y = 1、3、5、7）的情况，DFSDM_CHyDATINR 被写保护。如果偶数通道被设为双通道模式，则下一奇数通道必须进入标准模式 (DATPACK[1:0] = 0)，以便与偶数通道正确配合。

3：保留

只有在（DFSDM_CHyCFGR1 寄存器中）CHEN = 0时，才能修改该值。

位 13:12 **DATMPX[1:0]**：通道 y 输入数据复用器 (Input data multiplexer for channel y)

0：通道 y 输入数据取自外部串行输入，为 1 位值。DFSDM_CHyDATINR 寄存器被写保护。

1：保留

2：通道 y 输入数据取自内部 DFSDM_CHyDATINR 寄存器（通过 CPU/DMA 直接写操作实现）。可写入一个或两个 16 位数据采样，具体取决于 DATPACK[1:0] 位域设置。

3：保留

注： 只有在（DFSDM_CHyCFGR1 寄存器中）CHEN = 0时，才能修改该值。

位 11:9 保留，必须保持复位值。

位 8 **CHINSEL**：通道输入选择 (Channel inputs selection)

0：通道输入取自同一通道 y 的引脚。

1：通道输入取自下一通道（通道 (y+1) 取 8 的模）的引脚。

只有在（DFSDM_CHyCFGR1 寄存器中）CHEN = 0时，才能修改该值。

位 7 **CHEN**：通道 y 使能 (Channel y enable)

0：禁止通道 y

1：使能通道 y

如果使能了通道 y，则会根据给定通道设置开始接收串行数据。

位 6 **CKABEN**：通道 y 时钟缺失检测器使能 (Clock absence detector enable on channel y)

0：禁止通道 y 时钟缺失检测器

1：使能通道 y 时钟缺失检测器

位 5 **SCDEN**：通道 y 短路检测器使能 (Short-circuit detector enable on channel y)

0：输入通道 y 将不受短路检测器保护

1：输入通道 y 将持续受短路检测器保护

位 4 保留，必须保持复位值。

位 3:2 **SPICKSEL[1:0]**：通道 y SPI 时钟选择 (SPI clock select for channel y)

0：时钟来自外部 CKINy 输入——基于 SITP[1:0] 的采样点

1：时钟来自内部 CKOUT 输出——基于 SITP[1:0] 的采样点

2：时钟来自内部 CKOUT——在 CKOUT 每个第二个下降沿的采样点。
用于连接外部 ΣΔ 调制器，调制器将时钟输入（来自 CKOUT）除以 2，来生成输出串行通信时钟（该输出时钟变化在各个时钟输入上升沿有效）。

3：时钟来自内部 CKOUT 输出——在 CKOUT 每个第二个上升沿的采样点。
用于连接外部 ΣΔ 调制器，调制器将时钟输入（来自 CKOUT）除以 2，来生成输出串行通信时钟（该输出时钟变化在各个时钟输入下降沿有效）。

只有在（DFSDM_CHyCFGR1 寄存器中）CHEN = 0时，才能修改该值。

位 1:0 **SITP[1:0]**：通道 y 串行接口类型 (Serial interface type for channel y)

00：上升沿选通数据 SPI

01：下降沿选通数据 SPI

10：DATINy 引脚曼彻斯特编码输入：上升沿 = 逻辑 0，下降沿 = 逻辑 1

11：DATINy 引脚曼彻斯特编码输入：上升沿 = 逻辑 1，下降沿 = 逻辑 0

只有在（DFSDM_CHyCFGR1 寄存器中的）CHEN = 0 时，才能修改该值。

15.7.2 DFSDM 通道 y 配置寄存器 (DFSDM_CHyCFGR2)

DFSDM channel y configuration register

该寄存器指定通道 y 所使用的参数。

偏移地址：0x04 + 0x20 * y（y = 0 到 7）

复位值：0x0000 0000

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

位 31:8 **OFFSET[23:0]**：通道 y 24 位校准偏移 (24-bit calibration offset for channel y)

对于通道 y 的每一次转换结果，都执行 OFFSET。

该值由软件设置。

位 7:3 **DTRBS[4:0]**：通道 y 数据右移位 (Data right bit-shift for channel y)

0-31：定义积分器输出数据结果的移位——将向右移多少位以获得最终结果。在进行偏移校正之前执行移位。数据移位将结果四舍五入为最接近的整数值。移位结果的符号保留（以便得到有效 24 位有符号格式的结果数据）。

只有在（DFSDM_CHyCFGR1 寄存器中）CHEN = 0时，才能修改该值。

位 2:0 保留，必须保持复位值。

15.7.3 DFSDM 通道 y 模拟看门狗和短路检测器寄存器 (DFSDM_CHyAWSCDR)

DFSDM channel y analog watchdog and short-circuit detector register

通道 y 短路检测器和模拟看门狗设置

偏移地址：0x08 + 0x20 * y（y = 0 到 7)）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWFORD[1:0]		Res.	AWFOSR[4:0]				
								rw	rw		rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BKSCD[3:0]				Res.	Res.	Res.	Res.	SCDT[7:0]							
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 23:22 **AWFORD[1:0]**：通道 y 模拟看门狗 Sinc 滤波器阶数 (Analog watchdog Sinc filter order on channel y)

0：FastSinc 滤波器类型
1：$Sinc^1$ 滤波器类型
2：$Sinc^2$ 滤波器类型
3：$Sinc^3$ 滤波器类型

$Sinc^x$ 滤波器类型传递函数：

$$H(z) = \left(\frac{1 - z^{-FOSR}}{1 - z^{-1}}\right)^x$$

FastSinc 滤波器类型传递函数：

$$H(z) = \left(\frac{1 - z^{-FOSR}}{1 - z^{-1}}\right)^2 \cdot (1 + z^{-(2 \cdot FOSR)})$$

只有在（DFSDM_CHyCFGR1 寄存器中）CHEN = 0时，才能修改该位。

位 21 保留，必须保持复位值。

位 20:16 **AWFOSR[4:0]**：通道 y 模拟看门狗滤波器过采样率（抽取率）(Analog watchdog filter oversampling ratio (decimation rate) on channel y)

0 - 31：定义 Sinc 类型滤波器的长度，长度范围为 1 - 32 (AWFOSR + 1)。该数字还表示模拟数据速率的抽取率。

只有在（DFSDM_CHyCFGR1 寄存器中）CHEN = 0时，才能修改该位。

注： 如果 AWFOSR = 0，则滤波器无影响（滤波器旁路）。

位 15:12 **BKSCD[3:0]**：通道 y 短路检测器断路信号分配 (Break signal assignment for short-circuit detector on channel y)

BKSCD[i] = 0：断路 i 信号未分配至通道 y 短路检测器
BKSCD[i] = 1：断路 i 信号已分配至通道 y 短路检测器

位 11:8 保留，必须保持复位值。

位 7:0 **SCDT[7:0]**：通道 y 的短路检测器阈值 (short-circuit detector threshold for channel y)

这些位可由软件写入，用来定义短路检测器的计数器阈值。如果达到该值，则在给定通道上发生短路检测器事件。

15.7.4 DFSDM 通道 y 看门狗滤波器数据寄存器 (DFSDM_CHyWDATR)

DFSDM channel y watchdog filter data register

该寄存器包含的数据来自于与输入通道 y 关联的模拟看门狗滤波器。

偏移地址：0x0C + 0x20 * y（y = 0 到 7）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA[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 **WDATA[15:0]**：输入通道 y 看门狗数据 (Input channel y watchdog data)
由输入通道 y 模拟看门狗滤波器转换的数据。该通道进行连续的数据转换（无触发），转换分辨率受一定限制（OSR=1..32/sinc 阶数 = 1..3）。

15.7.5 DFSDM 通道 y 数据输入寄存器 (DFSDM_CHyDATINR)

DFSDM channel y data input register

该寄存器包含的 16 位输入数据将通过 DFSDM 滤波器模块进行处理。

偏移地址：0x10 + 0x20 * y（y = 0 到 7）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INDAT1[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
INDAT0[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 **INDAT1[15:0]**：通道 y 或通道 y+1 的输入数据 (Input data for channel y or channel y+1)

当 DATMPX[1:0] = 1 或 DATMPX[1:0] = 2 时，输入并行通道数据将通过数字滤波器进行处理。
数据可由 CPU/DMA 写入（如果 DATMPX[1:0]=2）。
当 DATPACK[1:0] = 0（标准模式）时
INDAT0[15:0] 被写保护（不用于输入采样）。
当 DATPACK[1:0] = 1（交错模式）时
第二个通道 y 数据采样存储到 INDAT1[15:0] 中。第一个通道 y 数据采样存储到 INDAT0[15:0] 中。
DFSDM_FLTx 滤波器依次读取该两个采样，作为两个通道 y 数据采样。
当 DATPACK[1:0] = 2（双通道模式）时
对于偶数 y 通道：INDAT1[15:0] 中的采样自动复制到通道 (y+1) 的 INDAT0[15:0] 中。
对于奇数 y 通道：INDAT1[15:0] 被写保护。
请参见*第 15.4.6 节：并行数据输入*以获得更多信息。
INDAT0[15:1] 采用 16 位带符号格式。

位 15:0 **INDAT0[15:0]**：通道 y 输入数据 (Input data for channel y)

当 DATMPX[1:0] = 1 或 DATMPX[1:0] = 2 时，输入并行通道数据将通过数字滤波器进行处理。
数据可由 CPU/DMA 写入（如果 DATMPX[1:0]=2）。
当 DATPACK[1:0] = 0（标准模式）时
通道 y 数据采样存储到 INDAT0[15:0] 中。
当 DATPACK[1:0] = 1（交错模式）时
第一个通道 y 数据采样存储到 INDAT0[15:0] 中。第二个通道 y 数据采样存储到 INDAT1[15:0] 中。
DFSDM_FLTx 滤波器依次读取该两个采样，作为两个通道 y 数据采样。
当 DATPACK[1:0] = 2（双模式）时
对于偶数 y 通道：通道 y 数据采样存储到 INDAT0[15:0] 中。
对于奇数 y 通道：INDAT0[15:0] 被写保护。
请参见*第 15.4.6 节：并行数据输入*以获得更多信息。
INDAT0[15:0] 采用 16 位带符号格式。

15.8 DFSDM 滤波器 x 模块寄存器 (x=0..3)

15.8.1 DFSDM 滤波器 x 控制寄存器 1 (DFSDM_FLTxCR1)

DFSDM filter x control register 1

偏移地址：0x100 + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	AWFSEL	FAST	Res.	Res.	RCH[2:0]			Res.	Res.	RDMAEN	Res.	RSYNC	RCONT	RSWSTART	Res.
	rw	rw			rw	rw	rw			rw		rw	rw	rt_w1	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	JEXTEN[1:0]		Res.	Res.	JEXTSEL[2:0]			Res.	Res.	JDMAEN	JSCAN	JSYNC	Res.	JSWSTART	DFEN
	rw	rw			rw	rw	rw			rw	rw	rw		rt_w1	rw

位 31 保留，必须保持复位值。

位 30 **AWFSEL**：模拟看门狗快速模式选择 (Analog watchdog fast mode select)

0：数据输出值的模拟看门狗（在数字滤波器之后）。在偏移校正和移位后进行比较

1：通道收发器值的模拟看门狗（在看门狗滤波器之后）

位 29 **FAST**：常规转换的快速转换模式选择 (Fast conversion mode selection for regular conversions)

0：禁止快速转换模式

1：使能快速转换模式

在连续模式下进行常规转换时，如果使能了快速模式，则每次转换（第一次转换除外）都比在标准模式下的转换执行得快。该位对于不连续的转换没有影响。

只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位。

如果 FAST=0（或者 FAST=1 连续模式下的第一次转换），则：

$t = [F_{OSR} * (I_{OSR}-1 + F_{ORD}) + F_{ORD}] / f_{CKIN}$.....（适用于 $Sinc^x$ 滤波器）

$t = [F_{OSR} * (I_{OSR}-1 + 4) + 2] / f_{CKIN}$.....（适用于 FastSinc 滤波器）

如果在连续模式下 FAST=1（第一次转换除外），则：

$t = [F_{OSR} * I_{OSR}] / f_{CKIN}$

如果 $F_{OSR} = F_{OSR}[9:0] + 1 = 1$ 时（滤波器旁路，仅积分器有效），则：

$t = I_{OSR} / f_{CKIN}$（... 但是 CNVCNT = 0）

其中，f_{CKIN} 为（给定通道 CKINy 引脚上的）通道输入时钟频率，在并行数据输入的情况下，其表示输入数据速率。

位 28:27 保留，必须保持复位值。

位 26:24 **RCH[2:0]**：常规通道选择 (Regular channel selection)

0：将通道 0 选作常规通道

1：将通道 1 选作常规通道

...

7：将通道 7 选作常规通道

当 RCIP = 1 时写入这些位，并在下一个常规转换开始时生效。这在连续模式下（当 RCONT = 1 时）特别有用。它还会影响待处理的常规转换（因正在进行注入转换而暂停）。

位 23:22 保留，必须保持复位值。

位 21 **RDMAEN**：使能 DMA 通道以读取常规转换数据 (DMA channel enabled to read data for the regular conversion)

0：未使能 DMA 通道来读取常规数据

1：已使能 DMA 通道来读取常规数据

只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位。

位 20 保留，必须保持复位值。

位 19 **RSYNC**：使用 DFSDM_FLT0 同步启动常规转换 (Launch regular conversion synchronously with DFSDM_FLT0)

0：不使用 DFSDM_FLT0 同步启动常规转换

1：在 DFSDM_FLT0 中启动常规转换的同时，在该 DFSDM_FLTx 中启动常规转换

只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位。

位 18 **RCONT**：常规转换的连续模式选择 (Continuous mode selection for regular conversions)

0：对于每个转换请求，只转换一次常规通道

1：在发出每个转换请求后，重复转换常规通道

在连续常规转换进行过程中，向该位写入“0”将立即停止连续模式。

位 17 **RSWSTART**：软件启动常规通道转换 (Software start of a conversion on the regular channel)

0：写入“0”无影响

1：写入“1”会发出启动常规通道转换的请求，并会使 RCIP 变为“1”。如果已经设置 RCIP=1，则对 RSWSTART 进行的写操作无效。如果 RSYNC = 1，则写入“1”无效。

此位始终读为“0”。

位 16:15 保留，必须保持复位值。

位 14:13 **JEXTEN[1:0]**：注入转换触发使能和触发边沿选择 (Trigger enable and trigger edge selection for injected conversions)

00：禁止触发检测

01：所选触发的每个上升沿都会发出启动注入转换的请求

10：所选触发的每个下降沿都会发出启动注入转换的请求

11：所选触发的上升沿和下降沿都会发出启动注入转换的请求

只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位。

位 12:11 保留，必须保持复位值。

位 10:8 **JEXTSEL[2:0]**：启动注入转换的触发信号选择 (Trigger signal selection for launching injected conversions)

0x0-0x7：根据下表选择的触发输入。

只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位。

	DFSDM1_FLT0	DFSDM1_FLT1
0x00	dfsdm_jtrg0	dfsdm_jtrg0
0x01	dfsdm_jtrg1	dfsdm_jtrg1
0x02	dfsdm_jtrg2	dfsdm_jtrg2
0x03	dfsdm_jtrg3	dfsdm_jtrg3
0x04	dfsdm_jtrg5	dfsdm_jtrg5
0x05	dfsdm_jtrg7	dfsdm_jtrg7
0x06	dfsdm_jtrg9	dfsdm_jtrg9
0x07	dfsdm_jtrg10	dfsdm_jtrg10

请参见*表 89：DFSDM1 触发连接*。

	DFSDM2_FLT0	DFSDM2_FLT1	DFSDM2_FLT2	DFSDM2_FLT3
0x00	dfsdm_jtrg0	dfsdm_jtrg0	dfsdm_jtrg0	dfsdm_jtrg0
0x01	dfsdm_jtrg1	dfsdm_jtrg1	dfsdm_jtrg1	dfsdm_jtrg1
0x02	dfsdm_jtrg2	dfsdm_jtrg2	dfsdm_jtrg2	dfsdm_jtrg2
0x03	dfsdm_jtrg3	dfsdm_jtrg3	dfsdm_jtrg3	dfsdm_jtrg4
0x04	dfsdm_jtrg5	dfsdm_jtrg5	dfsdm_jtrg5	dfsdm_jtrg6
0x05	dfsdm_jtrg7	dfsdm_jtrg7	dfsdm_jtrg8	dfsdm_jtrg8
0x06	dfsdm_jtrg9	dfsdm_jtrg9	dfsdm_jtrg9	dfsdm_jtrg9
0x07	dfsdm_jtrg10	dfsdm_jtrg10	dfsdm_jtrg10	dfsdm_jtrg10

请参见*表 90：DFSDM2 触发器连接*。

位 7:6 保留，必须保持复位值。

位 5 **JDMAEN**：为注入通道组使能 DMA 以读取数据 (DMA channel enabled to read data for the injected channel group)

0：未使能 DMA 通道来读取注入数据

1：已使能 DMA 通道来读取注入数据

只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位。

位 4 **JSCAN**：注入转换的扫描转换模式 (Scanning conversion mode for injected conversions)

0：对注入通道组执行一次通道转换，然后选择该通道组下一个通道。

1：从选择的最低通道开始，对注入通道组执行连续转换。

只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位。

如果 JSCAN=0，则对 JCHG 进行写入操作会将通道选择复位为最小所选通道。

位 3 **JSYNC**：使用 DFSDM_FLT0 JSWSTART 触发同步启动注入转换 (Launch an injected conversion synchronously with the DFSDM_FLT0 JSWSTART trigger)

0：不使用 DFSDM_FLT0 同步启动注入转换

1：在 DFSDM_FLT0 中通过 JSWSTART 触发启动注入转换的同时，在该 DFSDM_FLTx 中启动注入转换

只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位。

位 2 保留，必须保持复位值。

位 1 **JSWSTART**：启动注入通道组转换 (Start a conversion of the injected group of channels)

0：写入“0”无影响。

1：写入“1”会发出转换注入通道组请求，同时会使 JCIP 变为“1”。如果已经设置 JCIP = 1，则对 JSWSTART 进行的写操作无效。如果 JSYNC = 1，则写入“1”无效。

此位始终读为“0”。

位 0 **DFEN**：DFSDM_FLTx 使能 (DFSDM_FLTx enable)

0：禁止 DFSDM_FLTx。给定 DFSDM_FLTx 的所有转换均立即停止，DFSDM_FLTx 的所有功能均停止。

1：使能 DFSDM_FLTx。如果使能了 DFSDM_FLTx，则 DFSDM_FLTx 会根据其设置开始工作。

通过设置 DFEN = 0 清零的数据：

– 寄存器 DFSDM_FLTxISR 被设为复位状态

– 寄存器 DFSDM_FLTxAWSR 被设为复位状态

15.8.2 DFSDM 滤波器 x 控制寄存器 2 (DFSDM_FLTxCR2)

DFSDM filter x control register 2

偏移地址：0x104 + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDCH[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
EXCH[7:0]								Res.	CKABIE	SCDIE	AWDIE	ROVRIE	JOVRIE	REOCIE	JEOCIE
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 **AWDCH[7:0]**：模拟看门狗通道选择 (Analog watchdog channel selection)

这些位用于选择由模拟看门狗持续监控的输入通道

AWDCH[y] = 0：在通道 y 上禁止模拟看门狗

AWDCH[y] = 1：在通道 y 上使能模拟看门狗

位 15:8 **EXCH[7:0]**：极值检测器通道选择 (Extremes detector channel selection)

这些位选择极值检测器要采用的输入通道

EXCH[y] = 0：极值检测器不接受通道 y 的数据

EXCH[y] = 1：极值检测器接受通道 y 的数据

位 7 保留，必须保持复位值。

位 6 **CKABIE**：时钟缺失中断使能 (Clock absence interrupt enable)
0：禁止检测通道输入时钟缺失中断
1：使能检测通道输入时钟缺失中断
请参见 DFSDM_FLTxISR 中的 CKABF[7:0] 说明。
注： CKABIE 仅存在于 DFSDM_FLT0CR2 寄存器中（滤波器 x = 0）

位 5 **SCDIE**：短路检测器中断使能 (Short-circuit detector interrupt enable)
0：禁止短路检测器中断
1：使能短路检测器中断
请参见 DFSDM_FLTxISR 中的 SCDF[7:0] 说明。
注： SCDIE 仅存在于 DFSDM_FLT0CR2 寄存器中（滤波器 x = 0）

位 4 **AWDIE**：模拟看门狗中断使能 (Analog watchdog interrupt enable)
0：禁止模拟看门狗中断
1：使能模拟看门狗中断
请参见 DFSDM_FLTxISR 中的 AWDF 说明。

位 3 **ROVRIE**：常规数据溢出中断使能 (Regular data overrun interrupt enable)
0：禁止常规数据溢出中断
1：使能常规数据溢出中断
请参见 DFSDM_FLTxISR 中的 ROVRF 说明。

位 2 **JOVRIE**：注入数据溢出中断使能 (Injected data overrun interrupt enable)
0：禁止注入数据溢出中断
1：使能注入数据溢出中断
请参见 DFSDM_FLTxISR 中的 JOVRF 说明。

位 1 **REOCIE**：常规转换结束中断使能 (Regular end of conversion interrupt enable)
0：禁止常规转换结束中断
1：使能常规转换结束中断
请参见 DFSDM_FLTxISR 中的 REOCF 说明。

位 0 **JEOCIE**：注入转换结束中断使能 (Injected end of conversion interrupt enable)
0：禁止注入转换结束中断
1：使能注入转换结束中断
请参见 DFSDM_FLTxISR 中的 JEOCF 说明。

15.8.3 DFSDM 滤波器 x 中断和状态寄存器 (DFSDM_FLTxISR)

DFSDM filter x interrupt and status register

偏移地址：0x108 + 0x80 * x，（x = 0 到 3）

复位值：0x00FF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SCDF[7:0]								CKABF[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.	RCIP	JCIP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDF	ROVRF	JOVRF	REOCF	JEOCF
	r	r									r	r	r	r	r

位 31:24 **SCDF[7:0]**：短路检测器标志 (short-circuit detector flag)

SDCF[y] = 0：通道 y 上未发生任何短路检测器事件

SDCF[y] = 1：通道 y 上的短路检测器计数器达到在 DFSDM_CHyAWSCDR 寄存器中编程的值

此位由硬件置 1， 该位可由软件使用 DFSDM_FLTxICR 寄存器中的相应 CLRSCDF[y] 位清零。当 CHEN[y] = 0（给定通道被禁止）时，SCDF[y] 还可以由硬件清零。

注： SCDF[7:0] 仅存在于 DFSDM_FLT0ISR 寄存器中（滤波器 x = 0）

位 23:16 **CKABF[7:0]**：时钟缺失标志 (Clock absence flag)

CKABF[y] = 0：通道 y 上存在时钟信号。

CKABF[y] = 1：通道 y 上不存在时钟信号。

当检测到通道 y 上不存在时钟信号时，给定 y 位由硬件置 1。当 CHEN = 0 时（请参见 DFSDM_CHyCFGR1 寄存器），它由硬件保持为 CKABF[y] = 1 状态。当收发器尚未被同步时，它由硬件保持为 CKABF[y] = 1 状态。它可由软件使用 DFSDM_FLTxICR 寄存器中的相应 CLRCKABF[y] 位清零。

注： CKABF[7:0] 仅存在于 DFSDM_FLT0ISR 寄存器中（滤波器 x = 0）

位 15 保留，必须保持复位值。

位 14 **RCIP**：常规转换正在进行状态 (Regular conversion in progress status)

0：未发出转换常规通道的请求

1：常规通道转换正在进行，或一个常规转换请求正在等待处理

当 RCIP = 1 时，忽略启动常规转换的请求。

位 13 **JCIP**：注入转换正在进行状态 (Injected conversion in progress status)

0：未发出转换注入通道组的请求（软件和触发器均未发出请求）

1：注入通道组转换处于正在进行状态（向 JSWSTART 写入了“1”）或者一个注入转换请求正在等待处理（由于触发检测）

当 JCIP = 1 时，忽略启动注入转换的请求。

位 12:5 保留，必须保持复位值。

位 4 **AWDF**：模拟看门狗 (Analog watchdog)

0：未发生模拟看门狗事件

1：模拟看门狗模块检测到电压超出在 DFSDM_FLTxAWLTR 或 DFSDM_FLTxAWHTR 寄存器中编程的值。

此位由硬件置 1， 它由软件通过以下方式清零：将 DFSDM_FLTxAWSR 寄存器中的所有源标志位 AWHTF[7:0] 和 AWLTF[7:0] 清零（通过向 DFSDM_FLTxAWCFR 寄存器中的清零位写入“1”实现）。

位 3 **ROVRF**：常规转换溢出标志 (Regular conversion overrun flag)

0：未发生常规转换溢出

1：发生了常规转换溢出，也就是说，常规转换已经完成，但 REOCF 仍然为“1”。RDATAR 不受溢出影响

此位由硬件置 1， 该位可由软件使用 DFSDM_FLTxICR 寄存器中的 CLRROVRF 位清零。

位 2 **JOVRF**：注入转换溢出标志 (Injected conversion overrun flag)

0：未发生注入转换溢出

1：发生了注入转换溢出，也就是说，注入转换已经完成，但 JEOCF 仍然为“1”。JDATAR 不受溢出影响

此位由硬件置 1， 该位可由软件使用 DFSDM_FLTxICR 寄存器中的 CLRJOVRF 位清零。

位 1 **REOCF**：常规转换结束标志 (End of regular conversion flag)

0：未完成常规转换

1：已完成常规转换并可以读取其数据

此位由硬件置 1， 当软件或 DMA 读取 DFSDM_FLTxRDATAR 时，该位清零。

位 0 **JEOCF**：注入转换结束标志 (End of injected conversion flag)
0：未完成注入转换
1：已完成注入转换并可以读取其数据
此位由硬件置 1， 当软件或 DMA 读取 DFSDM_FLTxJDATAR 时，该位清零。

注： *对于每个标志位，可通过将 DFSDM_FLTxCR2 中的相应位置 1 来使能中断。如果调用了中断，则必须在退出中断服务程序之前将该标志清零。*
当 DFEN = 0 时，DFSDM_FLTxISR 的所有位都会自动复位。

15.8.4 DFSDM 滤波器 x 中断标志清零寄存器 (DFSDM_FLTxICR)

DFSDM filter x interrupt flag clear register

偏移地址：0x10C + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLRSCDF[7:0]								CLRCKABF[7:0]							
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
15	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.	CLRROVRF	CLRJOVRF	Res.	Res.
												rc_w1	rc_w1		

位 31:24 **CLRSCDF[7:0]**：清零短路检测器标志 (Clear the short-circuit detector flag)
CLRSCDF[y] = 0：写入“0”无影响
CLRSCDF[y] = 1：向位置 y 写入“1”会将 DFSDM_FLTxISR 寄存器中的相应 SCDF[y] 位清零
注： CLRSCDF[7:0] 仅存在于 DFSDM_FLT0ICR 寄存器中（滤波器 x = 0）

位 23:16 **CLRCKABF[7:0]**：清零时钟缺失标志 (Clear the clock absence flag)
CLRCKABF[y] = 0：写入“0”无影响
CLRCKABF[y] = 1：向位置 y 写入“1”会将 DFSDM_FLTxISR 寄存器中的相应 CKABF[y] 位清零。当收发器尚未被同步时，时钟缺失标志会置 1，且无法由 CLRCKABF[y] 清零。
注： CLRCKABF[7:0] 仅存在于 DFSDM_FLT0ICR 寄存器中（滤波器 x = 0）

位 15:4 保留，必须保持复位值。

位 3 **CLRROVRF**：将常规转换溢出标志清零 (Clear the regular conversion overrun flag)
0：写入“0”无影响
1：写入“1”会将 DFSDM_FLTxISR 寄存器中的 ROVRF 位清零

位 2 **CLRJOVRF**：将注入转换溢出标志清零 (Clear the injected conversion overrun flag)
0：写入“0”无影响
1：写入“1”会将 DFSDM_FLTxISR 寄存器中的 JOVRF 位清零

位 1:0 保留，必须保持复位值。

注： *DFSDM_FLTxICR 的位始终读为“0”。*

15.8.5 DFSDM 滤波器 x 注入通道组选择寄存器 (DFSDM_FLTxJCHGR)

DFSDM filter x injected channel group selection register

偏移地址：0x110 + 0x80 * x，（x = 0 到 3）

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.	JCHG[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 **JCHG[7:0]**：注入通道组选择 (Injected channel group selection)

JCHG[y] = 0：通道 y 不属于注入组

JCHG[y] = 1：通道 y 属于注入组

如果 JSCAN = 1，则依次转换每一个所选通道。首先转换最小通道（通道 0，如果已选择），最后转换最大所选通道。

如果 JSCAN = 0，则只转换其中一个所选通道，然后通道选择移至下一个通道。如果 JSCAN=0，则对 JCHG 进行写入操作会将通道选择复位为最小所选通道。

必须始终至少为注入组选择一个通道。如果写操作导致所有 JCHG 位均变为零，则忽略写操作。

15.8.6 DFSDM 滤波器 x 控制寄存器 (DFSDM_FLTxFCR)

DFSDM filter x control register

偏移地址：0x114 + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

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

位 31:29 **FORD[2:0]**：Sinc 滤波器阶数 (Sinc filter order)

0：FastSinc 滤波器类型
1：$Sinc^1$ 滤波器类型
2：$Sinc^2$ 滤波器类型
3：$Sinc^3$ 滤波器类型
4：$Sinc^4$ 滤波器类型
5：$Sinc^5$ 滤波器类型
6-7：保留
$Sinc^x$ 滤波器类型传递函数：

$$H(z) = \left(\frac{1 - z^{-FOSR}}{1 - z^{-1}}\right)^x$$

FastSinc 滤波器类型传递函数：

$$H(z) = \left(\frac{1 - z^{-FOSR}}{1 - z^{-1}}\right)^2 \cdot (1 + z^{-(2 \cdot FOSR)})$$

只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位。

位 28:26 保留，必须保持复位值。

位 25:16 **FOSR[9:0]**：Sinc 滤波器过采样率（抽取率）(Sinc filter oversampling ratio (decimation rate))

0 - 1023：定义 Sinc 类型滤波器的长度，长度范围为 1 - 1024 (F_{OSR} = FOSR[9:0] + 1)。该数字还表示滤波器输出数据速率的抽取率。
只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位

注： 如果 FOSR = 0，则滤波器无影响（滤波器旁路）。

位 15:8 保留，必须保持复位值。

位 7:0 **IOSR[7:0]**：积分器过采样率（平均长度）(Integrator oversampling ratio (averaging length))

0 - 255：积分器的长度，长度范围为 1 - 256 (IOSR + 1)。定义一个积分器输出数据采样是由多少个 Sinc 滤波器采样而合成的。积分器的输出数据速率将被减去该数值（额外数据抽取率）。
只有在 DFEN = 0 (DFSDM_FLTxCR1) 时才能修改该位

注： 如果 IOSR = 0，则积分器无影响（积分器旁路）。

15.8.7 注入组的 DFSDM 滤波器 x 数据寄存器 (DFSDM_FLTxJDATAR)

DFSDM filter x data register for injected group

偏移地址：0x118 + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
JDATA[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
JDATA[7:0]								Res.	Res.	Res.	Res.	Res.	JDATACH[2:0]		
r	r	r	r	r	r	r	r						r	r	r

位 31:8 **JDATA[23:0]**：注入组转换数据 (Injected group conversion data)
每次转换完注入组中的一个通道时，得到的数据都会存储在该字段中。JEOCF =1 时数据有效。读取该寄存器会将相应的 JEOCF 清零。

位 7:3 保留，必须保持复位值。

位 2:0 **JDATACH[2:0]**：最近转换的注入通道 (Injected channel most recently converted)
每次转换完注入组中的一个通道时，都会更新 JDATACH[2:0] 以指示转换了哪一个通道。因此，JDATA[23:0] 保持的数据对应于 JDATACH[2:0] 所指示的通道。

注： *可以使用 DMA 从该寄存器中读取数据。可以使用半字访问只读取转换数据的最高有效位 (MSB)。*
读取该寄存器还会将 DFSDM_FLTxISR 的 JEOCF 清零。因此，如果为从该寄存器中读取数据而激活了 DMA，则固件不能读取该寄存器。

15.8.8 常规通道的 DFSDM 滤波器 x 数据寄存器 (DFSDM_FLTxRDATAR)

DFSDM filter x data register for the regular channel

偏移地址：0x11C + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDATA[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
RDATA[7:0]								Res.	Res.	Res.	RPEND	Res.	RDATACH[2:0]		
r	r	r	r	r	r	r	r				r		r	r	r

位 31:8 **RDATA[23:0]**：常规通道转换数据 (Regular channel conversion data)
每次完成常规转换时，相应的数据都会存储在该寄存器中。当 REOCF =1 时，数据有效。读取该寄存器会将相应的 REOCF 清零。

位 7:5 保留，必须保持复位值。

位 4 **RPEND**：常规通道待处理数据 (Regular channel pending data)
因在转换期间发生注入通道触发而导致 RDATA[23:0] 中的常规数据被延迟处理

位 3 保留，必须保持复位值。

位 2:0 **RDATACH[2:0]**：最近转换的常规通道 (Regular channel most recently converted)
每次完成常规转换时，都会更新 RDATACH[2:0] 以指示转换了哪一个通道（因为在常规转换期间，可更新 DFSDM_FLTxCR1 寄存器中的常规通道选择 RCH[2:0]）。因此，RDATA[23:0] 保持的数据对应于 RDATACH[2:0] 所指示的通道。

注： *可以使用半字访问只读取转换数据的最高有效位 (MSB)。*
读取该寄存器还会将 DFSDM_FLTxISR 的 REOCF 清零。

15.8.9 DFSDM 滤波器 x 模拟看门狗阈值上限寄存器 (DFSDM_FLTxAWHTR)

DFSDM filter x analog watchdog high threshold register

偏移地址：0x120 + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

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

位 31:8 **AWHT[23:0]**：模拟看门狗阈值上限 (Analog watchdog high threshold)

这些位由软件写入，用于定义模拟看门狗的阈值上限。

注： 对于通道收发器监视器 (AWFSEL = 1)，高 16 位 (AWHT[23:8]) 会定义 16 位阈值，该阈值与模拟看门狗滤波器输出进行比较（因为来自模拟看门狗滤波器的数据的分辨率最高为 16 位）。在这种情况下，不对位 AWHT[7:0] 进行比较。

位 7:4 保留，必须保持复位值。

位 3:0 **BKAWH[3:0]**：模拟看门狗阈值上限事件的断路信号分配 (Break signal assignment to analog watchdog high threshold event)

BKAWH[i] = 0：断路 i 信号不分配到模拟看门狗阈值上限事件

BKAWH[i] = 1：断路 i 信号分配到模拟看门狗阈值上限事件

15.8.10 DFSDM 滤波器 x 模拟看门狗阈值下限寄存器 (DFSDM_FLTxAWLTR)

DFSDM filter x analog watchdog low threshold register

偏移地址：0x124 + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

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

位 31:8 **AWLT[23:0]**：模拟看门狗阈值下限 (Analog watchdog low threshold)

这些位由软件写入，用于定义模拟看门狗的阈值下限。

注： 对于通道收发器监视 (AWFSEL = 1)，仅高 16 位 (AWLT[23:8]) 定义 16 位阈值，该阈值与模拟看门狗滤波器输出进行比较（因为来自模拟看门狗滤波器的数据的分辨率最高为 16 位）。在这种情况下，不对位 AWLT[7:0] 进行比较。

位 7:4 保留，必须保持复位值。

位 3:0 **BKAWL[3:0]**：模拟看门狗阈值下限事件的断路信号分配 (Break signal assignment to analog watchdog low threshold event)

BKAWL[i] = 0：断路 i 信号不分配到模拟看门狗阈值下限事件
BKAWL[i] = 1：断路 i 信号分配到模拟看门狗阈值下限事件

15.8.11 DFSDM 滤波器 x 模拟看门狗状态寄存器 (DFSDM_FLTxAWSR)

DFSDM filter x analog watchdog status register

偏移地址：0x128 + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AWHTF[7:0]								AWLTF[7:0]							
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 15:8 **AWHTF[7:0]**：模拟看门狗阈值上限标志 (Analog watchdog high threshold flag)

AWHTF[y] = 1 指示通道 y 上存在阈值上限错误。该位由硬件置 1。该位可由软件使用 DFSDM_FLTxAWCFR 寄存器中的相应 CLRAWHTF[y] 位清零。

位 7:0 **AWLTF[7:0]**：模拟看门狗阈值下限标志 (Analog watchdog low threshold flag)

AWLTF[y] = 1 指示通道 y 上存在阈值下限错误。该位由硬件置 1。该位可由软件使用 DFSDM_FLTxAWCFR 寄存器中的相应 CLRAWLTF[y] 位清零。

注： DFEN = 0 时，DFSDM_FLTxAWSR 的所有位都会自动复位。

15.8.12 DFSDM 滤波器 x 模拟看门狗清零标志寄存器 (DFSDM_FLTxAWCFR)

DFSDM filter x analog watchdog clear flag register

偏移地址：0x12C + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRAWHTF[7:0]								CLRAWLTF[7:0]							
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 31:16 保留，必须保持复位值。

位 15:8 **CLRAWHTF[7:0]**：清零模拟看门狗阈值上限标志 (Clear the analog watchdog high threshold flag)
CLRAWHTF[y] = 0：写入“0”无影响
CLRAWHTF[y] = 1：向位置 y 写入“1”会将 DFSDM_FLTxAWSR 寄存器中的相应 AWHTF[y] 位清零。

位 7:0 **CLRAWLTF[7:0]**：将模拟看门狗阈值下限标志清零 (Clear the analog watchdog low threshold flag)
CLRAWLTF[y] = 0：写入“0”无影响
CLRAWLTF[y] = 1：向位置 y 写入“1”会将 DFSDM_FLTxAWSR 寄存器中的相应 AWLTF[y] 位清零。

15.8.13 DFSDM 滤波器 x 极值检测器最大值寄存器 (DFSDM_FLTxEXMAX)

DFSDM filter x Extremes detector maximum register

偏移地址：0x130 + 0x80 * x，（x = 0 到 3）

复位值：0x8000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EXMAX[23:8]															
rs_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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXMAX[7:0]								Res.	Res.	Res.	Res.	Res.	EXMAXCH[2:0]		
rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r	rc_r						r	r	r

位 31:8 **EXMAX[23:0]**：极值检测器最大值 (Extremes detector maximum value)
这些位由硬件置 1，用于指示 DFSDM_FLTx 所转换的最大值。通过读取该寄存器将 EXMAX[23:0] 位复位为值 (0x800000)。

位 7:3 保留，必须保持复位值。

位 2:0 **EXMAXCH[2:0]**：极值检测器最大值数据通道 (Extremes detector maximum data channel)
这些位包含的信息涉及哪一个通道的数据被存储到 EXMAX[23:0]。可通过读取该寄存器将这些位清零。

15.8.14 DFSDM 滤波器 x 极值检测器最小值寄存器 (DFSDM_FLTxEXMIN)

DFSDM filter x Extremes detector minimum register

偏移地址：0x134 + 0x80 * x，（x = 0 到 3）

复位值：0x7FFF FF00

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

位 31:8 **EXMIN[23:0]**：极值检测器最小值 (Extremes detector minimum value)
这些位由硬件置 1，用于指示 DFSDM_FLTx 所转换的最小值。通过读取该寄存器将 EXMIN[23:0] 位复位为值 (0x7FFFFF)。

位 7:3 保留，必须保持复位值。

位 2:0 **EXMINCH[2:0]**：极值检测器最小值数据通道 (Extremes detector minimum data channel)
这些位包含的信息涉及哪一个通道的数据被存储到 EXMIN[23:0]。可通过读取该寄存器将这些位清零。

15.8.15 DFSDM 滤波器 x 转换定时器寄存器 (DFSDM_FLTxCNVTIMR)

DFSDM filter x conversion timer register

偏移地址：0x138 + 0x80 * x，（x = 0 到 3）

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNVCNT[27:12]															
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
CNVCNT[11:0]												Res.	Res.	Res.	Res.
r	r	r	r	r	r	r	r	r	r	r	r				

位 31:4 **CNVCNT[27:0]**：28 位定时器计数转换时间 (28-bit timer counting conversion time) t = CNVCNT[27:0] / $f_{DFSDMCLK}$

定时器的输入时钟来自 DFSDM 时钟（系统时钟 $f_{DFSDMCLK}$）。转换时间测量始于每次转换开始，并止于每次转换结束（即第一次和最后一次串行采样之间的间隔）。只有在滤波器旁路 (FOSR[9:0] = 0) 的情况下，转换时间测量才会停止，且 CNVCNT[27:0] = 0。时间计时如下：

如果 FAST=0（或者 FAST=1 连续模式下的第一次转换），则：

$t = [F_{OSR} * (I_{OSR}-1 + F_{ORD}) + F_{ORD}] / f_{CKIN}$.....（适用于 $Sinc^x$ 滤波器）

$t = [F_{OSR} * (I_{OSR}-1 + 4) + 2] / f_{CKIN}$.....（适用于 FastSinc 滤波器）

如果在连续模式下 FAST=1（第一次转换除外），则：

$t = [F_{OSR} * I_{OSR}] / f_{CKIN}$

如果 F_{OSR} = FOSR[9:0] + 1 = 1（滤波器旁路，仅积分器有效），则：

CNVCNT = 0（时间计时停止，转换时间：$t = I_{OSR} / f_{CKIN}$）

其中，f_{CKIN} 为（给定通道 CKINy 引脚上的）通道输入时钟频率；在并行数据输入（来自 CPU/DMA 写操作）的情况下，其表示输入数据速率。

注： 当转换被中断时（例如通过禁止/使能所选通道），定时器也将该中断时间计算在内。

位 3:0 保留，必须保持复位值。

15.8.16 DFSDM 寄存器映射

下表对 DFSDM 寄存器进行了汇总。

表 97. DFSDM 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**DFSDM_CH0CFGR1**	DFSDMEN	CKOUTSRC	Res.	Res.	Res.	Res.	Res.	Res.	CKOUTDIV[7:0]								DATPACK[1:0]		DATMPX[1:0]		Res.	Res.	Res.	CHINSEL	CHEN	CKABEN	SCDEN	Res.	SPICKSEL[1:0]		SITP[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
0x04	**DFSDM_CH0CFGR2**	OFFSET[23:0]																								DTRBS[4:0]						Res.	Res.	Res.
	reset value	0																								0							
0x08	**DFSDM_CH0AWSCDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWFORD[1:0]		Res.	AWFOSR[4:0]					BKSCD[3:0]				Res.	Res.	Res.	Res.	SCDT[7:0]							
	reset value									0	0		0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0
0x0C	**DFSDM_CH0WDATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATA[15:0]															
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	**DFSDM_CH0DATINR**	INDAT1[15:0]																INDAT0[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 - 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	**DFSDM_CH1CFGR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATPACK[1:0]		DATMPX[1:0]		Res.	Res.	Res.	CHINSEL	CHEN	CKABEN	SCDEN	Res.	SPICKSEL[1:0]		SITP[1:0]	
	reset value																	0	0	0	0				0	0	0	0		0	0	0	0

表 97. DFSDM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x24	**DFSDM_CH1CFGR2**	OFFSET[23:0]																								DTRBS[4:0]					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	0	0			
0x28	**DFSDM_CH1AWSCDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWFORD[1:0]		Res.	AWFOSR[4:0]					BKSCD[3:0]				Res.	Res.	Res.	Res.	SCDT[7:0]							
	reset value									0	0		0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0
0x2C	**DFSDM_CH1WDATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATA[15:0]															
	reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	**DFSDM_CH1DATINR**	INDAT1[15:0]																INDAT0[15:0]															
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34 - 0x3C	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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	**DFSDM_CH2CFGR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATPACK[1:0]		DATMPX[1:0]		Res.	Res.	Res.	CHINSEL	CHEN	CKABEN	SCDEN	Res.	SPICKSEL[1:0]		SITP[1:0]	
	reset value																	0	0	0	0				0	0	0	0		0	0	0	0
0x44	**DFSDM_CH2CFGR2**	OFFSET[23:0]																								DTRBS[4:0]					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	0	0			
0x48	**DFSDM_CH2AWSCDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWFORD[1:0]		Res.	AWFOSR[4:0]					BKSCD[3:0]				Res.	Res.	Res.	Res.	SCDT[7:0]							
	reset value									0	0		0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0
0x4C	**DFSDM_CH2WDATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATA[15:0]															
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	**DFSDM_CH2DATINR**	INDAT1[15:0]																INDAT0[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
0x54 - 0x5C	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.
0x60	**DFSDM_CH3CFGR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATPACK[1:0]		DATMPX[1:0]		Res.	Res.	Res.	CHINSEL	CHEN	CKABEN	SCDEN	Res.	SPICKSEL[1:0]		SITP[1:0]	
	reset value																	0	0	0	0				0	0	0	0		0	0	0	0
0x64	**DFSDM_CH3CFGR2**	OFFSET[23:0]																								DTRBS[4:0]					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	0	0			

表 97. DFSDM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x68	**DFSDM_CH3AWSCDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWFORD[1:0]		Res.	AWFOSR[4:0]					BKSCD[3:0]				Res.	Res.	Res.	Res.	SCDT[7:0]							
	reset value									0	0		0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0
0x6C	**DFSDM_CH3WDATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATA[15:0]															
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x70	**DFSDM_CH3DATINR**	INDAT1[15:0]																INDAT0[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
0x74 - 0x7C	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.
0x80	**DFSDM_CH4CFGR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATPACK[1:0]		DATMPX[1:0]		Res.	Res.	Res.	CHINSEL	CHEN	CKABEN	SCDEN	Res.	SPICKSEL[1:0]		SITP[1:0]	
	reset value																	0	0	0	0				0	0	0	0		0	0	0	0
0x84	**DFSDM_CH4CFGR2**	OFFSET[23:0]																								DTRBS[4:0]					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	0	0			
0x88	**DFSDM_CH4AWSCDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWFORD[1:0]		Res.	AWFOSR[4:0]					BKSCD[3:0]				Res.	Res.	Res.	Res.	SCDT[7:0]							
	reset value									0	0		0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0
0x8C	**DFSDM_CH4WDATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATA[15:0]															
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x90	**DFSDM_CH4DATINR**	INDAT1[15:0]																INDAT0[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
0x94 - 0x9C	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.
0xA0	**DFSDM_CH5CFGR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATPACK[1:0]		DATMPX[1:0]		Res.	Res.	Res.	CHINSEL	CHEN	CKABEN	SCDEN	Res.	SPICKSEL[1:0]		SITP[1:0]	
	reset value																	0	0	0	0				0	0	0	0		0	0	0	0
0xA4	**DFSDM_CH5CFGR2**	OFFSET[23:0]																								DTRBS[4:0]					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	0	0			
0xA8	**DFSDM_CH5AWSCDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWFORD[1:0]		Res.	AWFOSR[4:0]					BKSCD[3:0]				Res.	Res.	Res.	Res.	SCDT[7:0]							
	reset value									0	0		0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0

表 97. DFSDM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xAC	**DFSDM_CH5WDATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATA[15:0]															
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xB0	**DFSDM_CH5DATINR**	INDAT1[15:0]																INDAT0[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
0xB4 - 0xBC	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.
0xC0	**DFSDM_CH6CFGR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATPACK[1:0]		DATMPX[1:0]		Res.	Res.	Res.	CHINSEL	CHEN	CKABEN	SCDEN	Res.	SPICKSEL[1:0]		SITP[1:0]	
	reset value																	0	0	0	0				0	0	0	0		0	0	0	0
0xC4	**DFSDM_CH6CFGR2**	OFFSET[23:0]																								DTRBS[4:0]						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	0	0			
0xC8	**DFSDM_CH6AWSCDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWFORD[1:0]		Res.	AWFOSR[4:0]					BKSCD[3:0]				Res.	Res.	Res.	Res.	SCDT[7:0]							
	reset value								0	0		0	0	0	0	0		0	0	0	0					0	0	0	0	0	0	0	0
0xCC	**DFSDM_CH6WDATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATA[15:0]															
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xD0	**DFSDM_CH6DATINR**	INDAT1[15:0]																INDAT0[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
0xD4 - 0xDC	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.
0xE0	**DFSDM_CH7CFGR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATPACK[1:0]		DATMPX[1:0]		Res.	Res.	Res.	CHINSEL	CHEN	CKABEN	SCDEN	Res.	SPICKSEL[1:0]		SITP[1:0]	
	reset value																	0	0	0	0				0	0	0	0		0	0	0	0
0xE4	**DFSDM_CH7CFGR2**	OFFSET[23:0]																								DTRBS[4:0]						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	0	0			
0xE8	**DFSDM_CH7AWSCDR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWFORD[1:0]		Res.	AWFOSR[4:0]					BKSCD[3:0]				Res.	Res.	Res.	Res.	SCDT[7:0]							
	reset value								0	0		0	0	0	0	0		0	0	0	0					0	0	0	0	0	0	0	0
0xEC	**DFSDM_CH7WDATR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDATA[15:0]															
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xF0	**DFSDM_CH7DATINR**	INDAT1[15:0]																INDAT0[15:0]															
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 97. DFSDM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xF4 - 0xFC	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x100	**DFSDM_ FLT0CR1**	Res.	AWFSEL	FAST	Res.	Res.	RCH[2:0]			Res.	Res.	RDMAEN	Res.	RSYNC	RCONT	RSW START	Res.	Res.	JEXTEN[1:0]		Res.	Res.	JEXTSEL[2:0]			Res.	Res.	JDMAEN	JSCAN	JSYNC	Res.	JSW START	DFEN
	reset value		0	0			0	0	0			0		0	0	0			0	0			0	0	0			0	0	0		0	0
0x104	**DFSDM_ FLT0CR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDCH[7:0]								EXCH[7:0]								Res.	CKABIE	SCDIE	AWDIE	ROVRIE	JOVRIE	REOCIE	JEOCIE
	reset value									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0
0x108	**DFSDM_ FLT0ISR**	SCDF[7:0]								CKABF[7:0]								Res.	RCIP	JCIP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDF	ROVRF	JOVRF	REOCF	JEOCF
	reset value	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1		0	0									0	0	0	0	0
0x10C	**DFSDM_ FLT0ICR**	CLRSCDF[7:0]								CLRCKABF[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLR ROVRF	CLR JOVRF	Res.	Res.
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0													0	0		
0x110	**DFSDM_ FLT0JCHGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JCHG[7:0]							
	reset value																									0	0	0	0	0	0	0	1
0x114	**DFSDM_ FLT0FCR**	FORD[2:0]			Res.	Res.	Res.	FOSR[9:0]										Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IOSR[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
0x118	**DFSDM_ FLT0JDATAR**	JDATA[23:0]																								Res.	Res.	Res.	Res.	Res.	JDATACH [2:0]		
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						0	0	0
0x11C	**DFSDM_ FLT0RDATAR**	RDATA[23:0]																								Res.	Res.	Res.	RPEND	Res.	RDATA CH[2:0]		
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0		0	0	0
0x120	**DFSDM_ FLT0AWHTR**	AWHT[23:0]																								Res.	Res.	Res.	Res.	BKAWH[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
0x124	**DFSDM_ FLT0AWLTR**	AWLT[23:0]																								Res.	Res.	Res.	Res.	BKAWL[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
0x128	**DFSDM_ FLT0AWSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWHTF[7:0]								AWLTF[7:0]							
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x12C	**DFSDM_ FLT0AWCFR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLRAWHTF[7:0]								CLRAWLTF[7:0]							
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 97. DFSDM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x130	**DFSDM_ FLT0EXMAX**	EXMAX[23:0]																								Res.	Res.	Res.	Res.	Res.	EXMAXCH[2:0]		
	reset value	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						0	0	0
0x134	**DFSDM_ FLT0EXMIN**	EXMIN[23:0]																								Res.	Res.	Res.	Res.	Res.	EXMINCH[2:0]		
	reset value	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1						0	0	0
0x138	**DFSDM_ FLT0CNVTIMR**	CNVCNT[27:0]																												Res.	Res.	Res.	Res.
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x13C - 0x17C	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.
0x180	**DFSDM_ FLT1CR1**	Res.	AWFSEL	FAST	Res.	Res.	RCH[2:0]			Res.	Res.	RDMAEN	Res.	RSYNC	RCONT	RSW START	Res.	Res.	JEXTEN[1:0]		Res.	Res.	JEXTSEL[2:0]			Res.	Res.	JDMAEN	JSCAN	JSYNC	Res.	JSW START	DFEN
	reset value		0	0			0	0	0			0		0	0	0			0	0			0	0	0			0	0	0		0	0
0x184	**DFSDM_ FLT1CR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDCH[7:0]								EXCH[7:0]								Res.	Res.	Res.	AWDIE	ROVRIE	JOVRIE	REOCIE	JEOCIE
	reset value									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0	0	0	0	0
0x188	**DFSDM_ FLT1ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RCIP	JCIP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDF	ROVRF	JOVRF	REOCF	JEOCF
	reset value																		0	0									0	0	0	0	0
0x18C	**DFSDM_ FLT1ICR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLR ROVRF	CLR JOVRF	Res.	Res.
	reset value																													0	0		
0x190	**DFSDM_ FLT1JCHGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JCHG[7:0]							
	reset value																									0	0	0	0	0	0	0	1
0x194	**DFSDM_ FLT1FCR**	FORD[2:0]			Res.	Res.	Res.	FOSR[9:0]										Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IOSR[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
0x198	**DFSDM_ FLT1JDATAR**	JDATA[23:0]																								Res.	Res.	Res.	Res.	Res.	JDATACH[2:0]		
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						0	0	0

表 97. DFSDM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x19C	**DFSDM_ FLT1RDATAR**	RDATA[23:0]																								Res.	Res.	Res.	RPEND	Res.	RDATA CH[2:0]		
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0		0	0	0
0x1A0	**DFSDM_ FLT1AWHTR**	AWHT[23:0]																								Res.	Res.	Res.	Res.	BKAWH[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
0x1A4	**DFSDM_ FLT1AWLTR**	AWLT[23:0]																								Res.	Res.	Res.	Res.	BKAWL[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
0x1A8	**DFSDM_ FLT1AWSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWHTF[7:0]								AWLTF[7:0]							
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1AC	**DFSDM_ FLT1AWCFR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLRAWHTF[7:0]								CLRAWLTF[7:0]							
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1B0	**DFSDM_ FLT1EXMAX**	EXMAX[23:0]																								Res.	Res.	Res.	Res.	Res.	EXMAXCH[2:0]		
	reset value	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						0	0	0
0x1B4	**DFSDM_ FLT1EXMIN**	EXMIN[23:0]																								Res.	Res.	Res.	Res.	Res.	EXMINCH[2:0]		
	reset value	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1						0	0	0
0x1B8	**DFSDM_ FLT1CNVTIMR**	CNVCNT[27:0]																												Res.	Res.	Res.	Res.
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x1BC - 0x1FC	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.
0x200	**DFSDM_ FLT2CR1**	Res.	AWFSEL	FAST	Res.	Res.	RCH[2:0]			Res.	Res.	RDMAEN	Res.	RSYNC	RCONT	RSW START	Res.	Res.	JEXTEN[1:0]		Res.	Res.	JEXTSEL[2:0]			Res.	Res.	Res.	JDMAEN	JSCAN	JSYNC	Res.	JSW START	DFEN
	reset value		0	0			0	0	0			0		0	0	0			0	0			0	0	0				0	0	0		0	0
0x204	**DFSDM_ FLT2CR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDCH[7:0]								EXCH[7:0]								Res.	Res.	Res.	AWDIE	ROVRIE	JOVRIE	REOCIE	JEOCIE
	reset value									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0	0	0	0	0
0x208	**DFSDM_ FLT2ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RCIP	JCIP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDF	ROVRF	JOVRF	REOCF	JEOCF
	reset value																		0	0									0	0	0	0	0
0x20C	**DFSDM_ FLT2ICR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLR ROVRF	CLR JOVRF	Res.	Res.
	reset value																													0	0		

表 97. DFSDM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x210	**DFSDM_FLT2JCHGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JCHG[7:0]							
	reset value																									0	0	0	0	0	0	0	1
0x214	**DFSDM_FLT2FCR**	FORD[2:0]			Res.	Res.	Res.	FOSR[9:0]										Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IOSR[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
0x218	**DFSDM_FLT2JDATAR**	JDATA[23:0]																								Res.	Res.	Res.	Res.	Res.	JDATACH[2:0]		
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						0	0	0
0x21C	**DFSDM_FLT2RDATAR**	RDATA[23:0]																								Res.	Res.	Res.	RPEND	Res.	RDATACH[2:0]		
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0		0	0	0
0x220	**DFSDM_FLT2AWHTR**	AWHT[23:0]																								Res.	Res.	Res.	Res.	BKAWH[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
0x224	**DFSDM_FLT2AWLTR**	AWLT[23:0]																								Res.	Res.	Res.	Res.	BKAWL[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
0x228	**DFSDM_FLT2AWSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWHTF[7:0]								AWLTF[7:0]							
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x22C	**DFSDM_FLT2AWCFR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLRAWHTF[7:0]								CLRAWLTF[7:0]							
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x230	**DFSDM_FLT2EXMAX**	EXMAX[23:0]																								Res.	Res.	Res.	Res.	Res.	EXMAXCH[2:0]		
	reset value	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						0	0	0
0x234	**DFSDM_FLT2EXMIN**	EXMIN[23:0]																								Res.	Res.	Res.	Res.	Res.	EXMINCH[2:0]		
	reset value	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1						0	0	0
0x238	**DFSDM_FLT2CNVTIMR**	CNVCNT[27:0]																												Res.	Res.	Res.	Res.
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x23C - 0x27C	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.
0x280	**DFSDM_FLT3CR1**	Res.	AWFSEL	FAST	Res.	Res.	RCH[2:0]			Res.	Res.	RDMAEN	Res.	RSYNC	RCONT	RSW START	Res.	Res.	JEXTEN[1:0]		Res.	Res.	JEXTSEL[2:0]			Res.	Res.	JDMAEN	JSCAN	JSYNC	Res.	JSW START	DFEN
	reset value		0	0			0	0	0			0		0	0	0			0	0			0	0	0			0	0	0		0	0

表 97. DFSDM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x284	**DFSDM_FLT3CR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDCH[7:0]								EXCH[7:0]								Res.	Res.	Res.	AWDIE	ROVRIE	JOVRIE	REOCIE	JEOCIE
	reset value									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0	0	0	0	0
0x288	**DFSDM_FLT3ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RCIP	JCIP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDF	ROVRF	JOVRF	REOCF	JEOCF
	reset value																		0	0									0	0	0	0	0
0x28C	**DFSDM_FLT3ICR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLR ROVRF	CLR JOVRF	Res.	Res.
	reset value																													0	0		
0x290	**DFSDM_FLT3JCHGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JCHG[7:0]							
	reset value																									0	0	0	0	0	0	0	1
0x294	**DFSDM_FLT3FCR**	FORD[2:0]			Res.	Res.	Res.	FOSR[9:0]										Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IOSR[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
0x298	**DFSDM_FLT3JDATAR**	JDATA[23:0]																								Res.	Res.	Res.	Res.	Res.	JDATACH[2:0]		
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						0	0	0
0x29C	**DFSDM_FLT3RDATAR**	RDATA[23:0]																								Res.	Res.	Res.	RPEND	Res.	RDATACH[2:0]		
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0		0	0	0
0x2A0	**DFSDM_FLT3AWHTR**	AWHT[23:0]																								Res.	Res.	Res.	Res.	BKAWH[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
0x2A4	**DFSDM_FLT3AWLTR**	AWLT[23:0]																								Res.	Res.	Res.	Res.	BKAWL[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
0x2A8	**DFSDM_FLT3AWSR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWHTF[7:0]								AWLTF[7:0]							
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2AC	**DFSDM_FLT3AWCFR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLRAWHTF[7:0]								CLRAWLTF[7:0]							
	reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2B0	**DFSDM_FLT3EXMAX**	EXMAX[23:0]																								Res.	Res.	Res.	Res.	Res.	EXMAXCH[2:0]		
	reset value	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						0	0	0

表 97. DFSDM 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2B4	**DFSDM_ FLT3EXMIN**	EXMIN[23:0]																								Res.	Res.	Res.	Res.	Res.	EXMINCH[2:0]		
	reset value	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1						0	0	0
0x2B8	**DFSDM_ FLT3CNVTIMR**	CNVCNT[27:0]																												Res.	Res.	Res.	Res.
	reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x2BC - 0x3FC	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 54 页的第 2.2.2 节*。

16 真随机数发生器 (RNG)

16.1 简介

RNG 是一个真正的随机数发生器，它基于模拟噪声源连续提供 32 位熵采样。应用程序可以将其用作实时熵源来构建符合 NIST 标准的确定性随机位发生器 (DRBG)。

RNG 真随机数发生器已按照德国 AIS-31 标准进行了验证。

16.2 RNG 主要特性

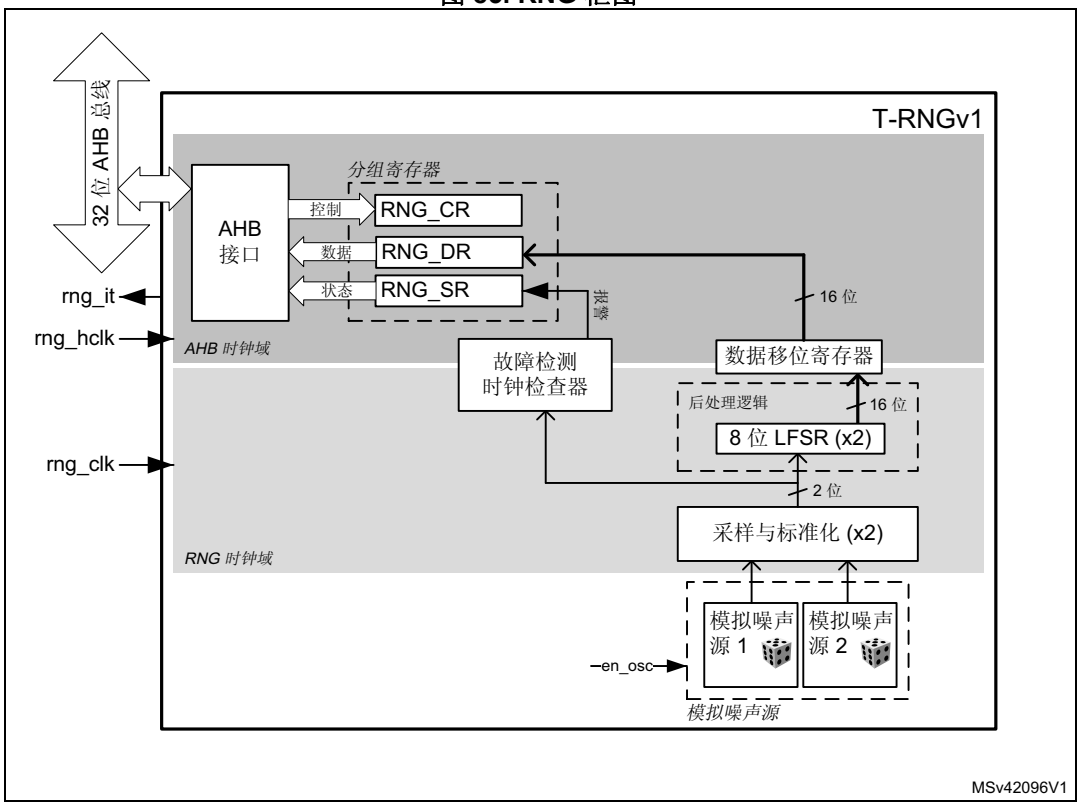
- RNG 提供的 32 位真随机数由模拟熵源生成并使用线性反馈移位寄存器 (LFSR) 进行后期处理。
- RNG 按照 AIS-31 预定义类 PTG.2 评估方法进行验证，该评估方法属于德国通用标准 (CC) 方案的组成部分。
- RNG 每 42 个 RNG 时钟周期（专用时钟）会生成一个 32 位随机采样。
- 内置带有相关错误处理的连续基本健康测试
  - 包括过低采样时钟检测和重复计数测试。
- 可被禁止以降低功耗。
- 具有 AMBA AHB 从外设，只能通过 32 位字进行单次访问（否则会生成 AHB 总线错误）。警告！任何不等于 32 位的写操作都可能破坏寄存器内容。

16.3 RNG 功能说明

16.3.1 RNG 框图

图 85 显示了 RNG 框图。

图 85. RNG 框图



16.3.2 RNG 内部信号

表 98 中列出了 RNG 级而非 STM32 产品级（焊盘上）所提供的内部信号，对这些信号有所了解会很有用。

表 98. RNG 内部输入/输出信号

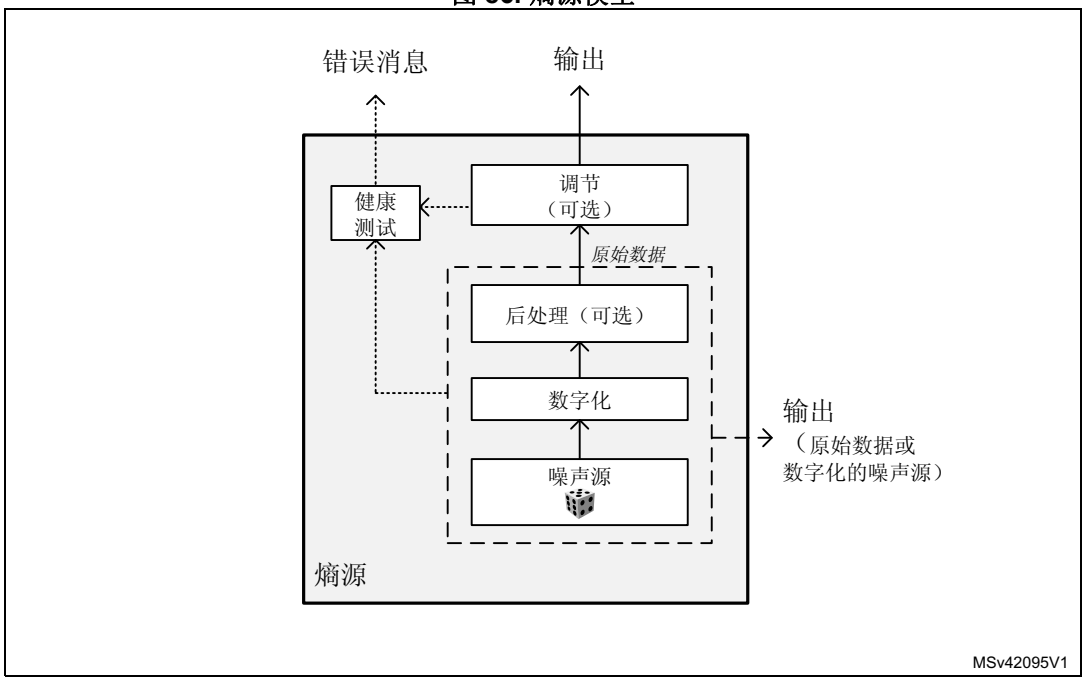
信号名称	信号类型	说明
rng_it	数字输出	RNG 全局中断请求
rng_hclk	数字输入	AHB 时钟
rng_clk	数字输入	RNG 专用时钟，与 rng_hclk 异步

16.3.3 随机数生成

真随机数发生器 (RNG) 按确定的间隔通过其 AHB 接口提供真随机数。RNG 可实现*图 86* 所示的熵源模型，并为应用程序提供三个主要功能：

- 采集熵源的位串输出
- 获取噪声源采样，用于验证目的
- 采集连续健康测试得到的错误消息

图 86. 熵源模型



RNG 的主要组件包括：

- 物理随机源（模拟噪声源）
- 该模拟噪声源的数字化处理级
- 后期处理噪声源（原始数据）提供级
- 原始数据的输出缓冲区。如果应用程序要求进一步加密调节，则需要由软件执行。
- 数字化噪声源的可选输出（无缓冲，在数字焊盘上）
- 数字化噪声源上的基本健康测试

下文对所有组件进行了详细介绍。

噪声源

噪声源组件包含不确定性的、能够提供熵值的活动，其最终决定了与熵源位串输出相关联的不确定性。它包括：

- 两个模拟噪声源，每个噪声源由三个自由运行的环形振荡器输出异或 (XOR) 而成。可以禁止这些模拟振荡器以实现节能，如*第 16.4 节：RNG 低功耗使用*所述。
- 这些输出的采样级由专用时钟输入 (**rng_clk**) 提供时钟信号，输出 2 位原始数据。

该噪声源采样与 AHB 接口时钟频率 (**rng_hclk**) 无关。

注： *第 16.7 节：熵源验证中提供了建议使用的 RNG 时钟频率。*

后期处理

从真随机噪声源获取的采样值由 2 位的位串组成。由于该噪声源输出进行了偏置，因此 RNG 会利用后期处理组件将偏置降至可接受的水平。

RNG 后期处理分为两个阶段，适用于每个噪声源位：

- 一半由 RNG 取自采样噪声源，一半取自取反的采样噪声源。因此，如果噪声源生成的“1”多于“0”（或者相反），则噪声源会被过滤掉
- 线性反馈移位寄存器 (LFSR) 执行白化处理，生成 8 位字符串。

该组件由 RNG 时钟提供时钟信号。

有关生成两个随机数之间所需的时间，以及 RNG 初始化与第一个采样可用之间所需的时间，请参见*第 16.6 节：RNG 处理时间*。

输出缓冲区

RNG_DR 数据输出寄存器最多可存储两个从后期处理组件 (LFSR) 输出的 16 位字。为了读回 32 位随机采样，需要等待 42 个 RNG 时钟周期。

当 RNG_DR 寄存器存在可用的随机数时，DRDY 标志就从“0”变为“1”。该标志会保持在置 1 状态，直至从 RNG_DR 寄存器读取一个字后输出缓冲区变空。

注： *如果使能了中断，则当该数据就绪标志从“0”变为“1”时，会产生中断。随后，中断会自动由 RNG 清零，如上所述。*

健康检查

该组件可确保整个熵源（包括其噪声源）正常启动并按预期运行，能够快速准确地定位故障，从而提供了高可靠性保证。

RNG 可实现以下健康检查功能：

1. 行为测试，可在熵源*运行时*进行
   - 重复计数测试，在以下情况时会指示发生错误：
   a) 其中一个噪声源持续不变地输出了 64 位或以上的“0”或“1”
   b) 其中一个噪声源持续不变地输出了 32 位或以上的“01”或“10”
2. 针对采样时钟源的连续测试
   - 实时“过慢”采样时钟检测器，如果一个 RNG 时钟周期小于 AHB 时钟周期的 16 分之一，则该检测器会指示错误。

RNG_SR 寄存器中的 CECS 和 SECS 状态位会在检测到错误条件时进行指示，*第 16.3.7 节：错误管理*对此进行了详细说明。

注： *检测到错误时可产生中断。*

16.3.4 RNG 初始化

如果发生硬件复位，则会发生以下一系列事件：

1. 使能模拟噪声源，逻辑在四个 RNG 时钟周期之后开始采样模拟输出、填充 LFSR 移位寄存器和相关的 16 位后期处理移位寄存器。
2. 输出缓冲区会按照 RNG 使用情况自动重填。

相关初始化时间请参见*第 16.6 节：RNG 处理时间*。

16.3.5 RNG 操作

正常工作

要利用中断运行 RNG，建议执行以下步骤：

1. 通过将 RNG_CR 寄存器中的 IE 位置 1 来使能中断。同时通过将 RNGEN 位置 1 来使能 RNG。
2. 这以后，准备好随机数时或出现错误时就生成中断。因此，每次发生中断时，应检查：
   - 未发生错误。RNG_SR 寄存器中的 SEIS 和 CEIS 位应该为“0”。
   - 随机数准备就绪。RNG_SR 寄存器中的 DRDY 位应该为“1”。
   - 如果满足以上两个条件，可读取 RNG_DR 寄存器的内容。

要在轮询模式下运行 RNG，建议执行以下步骤：

1. 通过将 RNG_CR 寄存器中的 RNGEN 位置“1”使能随机数生成。
2. 读取 RNG_SR 寄存器并检查：
   - 未发生错误（SEIS 和 CEIS 位应该为“0”）
   - 随机数准备就绪（DRDY 位应该为“1”）
3. 如果满足以上两个条件，可读取 RNG_DR 寄存器的内容。

注： *如果数据未就绪（DRDY = “0”），则 RNG_DR 会返回 0。*

低功耗运行

如果应用程序比较关注功耗，则可以使用低功耗策略，如*第 432 页的第 16.4 节：RNG 低功耗使用*所述。

软件后期处理

如果需要安全强度为 128 位、符合 NIST 要求的 DRBG，则必须基于 RNG 真随机数发生器构建符合要求的随机数发生器软件。

16.3.6 RNG 时钟

RNG 在两个不同的时钟上运行：AHB 总线时钟和专用 RNG 时钟。

AHB 时钟用于为 AHB 存储寄存器和后期处理组件提供时钟信号。RNG 时钟用于噪声源采样。*第 16.7 节：熵源验证*中详细介绍了建议使用的时钟配置。

注意： 如果 RNG_CR 寄存器中的 CED 位置“0”，那么 RNG 时钟频率**必须大于** AHB 时钟频率的 16 分之一，否则时钟检查器会指示时钟错误（RNG_SR 寄存器中的 CECS 或 CEIS），RNG 将停止生成随机数。

详细信息参见*第 16.3.1 节：RNG 框图*（AHB 和 RNG 时钟域）。

16.3.7 错误管理

健康检查模块在生成随机数的同时还验证噪声源行为以及 RNG 源时钟频率是正常的，具体说明见本部分。另外，还描述了相关的错误状态。

时钟错误检测

当使能时钟错误检测 (CED = 0) 时，如果 RNG 时钟频率过低，RNG 会停止生成随机数，并会将 **CEIS** 和 **CECS** 位都置“1”，指示发生了时钟错误。在这种情况下，应用程序应检查 RNG 时钟是否配置正确（参见*第 16.3.6 节：RNG 时钟*），随后必须将中断标志 CEIS 位清零。RNG 时钟正常运行后，CECS 将立即自动清零。

仅当 CECS 位置“0”时，RNG 才能正常运行。但需要注意的是，时钟错误对之前生成的随机数没有影响，因此 RNG_DR 寄存器内容仍可使用。

噪声源错误检测

如果发生噪声源（或种子）错误，RNG 会停止生成随机数，并会将 **SEIS** 和 **SECS** 位都置“1”，指示发生了种子错误。如果 RNG_DR 寄存器中有可用值，不能使用该值，因为它可能没有足够的熵。

为了从种子错误完全恢复，应用程序必须通过向 SEIS 位写入“0”来将其清零，然后将 RNGEN 位清零和置 1 以重新初始化并重新启动 RNG。

16.4 RNG 低功耗使用

考虑到功耗问题，可在 DRDY 位置“1”后立即禁止 RNG，具体方法为将 RNG_CR 寄存器中的 PNGEN 位设为“0”。存储在 RNG_DR 寄存器中的 32 位随机值将仍然可用。如果需要新的随机值，则应用程序将需要重新使能 RNG 并等待 42+4 个 RNG 时钟周期。

禁止 RNG 时，用户会禁用所有模拟种子发生器，各发生器的功耗列于期间手册中的电气特性部分。

16.5 RNG 中断

在 RNG 中，发生以下事件时会产生中断：

- 数据就绪标志
- 种子错误，请参见*第 16.3.7 节：错误管理*
- 时钟错误，请参见*第 16.3.7 节：错误管理*

可以使用专用的中断使能控制位，如*表 99* 所示。

表 99. RNG 中断请求

中断事件	事件标志	使能控制位
数据就绪标志	DRDY	IE
种子错误标志	SEIS	IE
时钟错误标志	CEIS	IE

用户可以通过更改 RNG_CR 寄存器中的屏蔽位或通用中断控制位 IE 的方式分别使能或禁止上述中断源。可以从 RNG_SR 寄存器中读取各个中断源的状态。

注： *仅当 RNG 已使能时，才会产生中断。*

16.6 RNG 处理时间

RNG 每 42 个 RNG 时钟周期可生成一个 32 位随机数。

在使用 RNGEN 位使能或重新使能 RNG 之后，需要等待 46 个 RNG 时钟周期才能生成随机数据。

16.7 熵源验证

16.7.1 简介

为了评估 RNG 可提供的熵大小，意法半导体按照 AIS-31 PTG.2 测试集对外设进行了测试。测试结果可根据需要提供，客户也可以使用 AIS 参考软件重现测量结果。客户还可以按照较早的 NIST SP800-22 测试集对 RNG 进行测试。

16.7.2 验证条件

意法半导体在以下条件下对 RNG 真随机数发生器进行了验证：

- RNG 时钟 rng_clk = 48 MHz（RNG_CR 寄存器中的 CED 位 = 0）且 rng_clk = 400 kHz（RNG_CR 中的 CED 位 = 1）
- AHB 时钟 rng_hclk= 60 MHz

16.7.3 数据采集

如果需要读取原始数据（而非预先处理的数据），则需请开发人员联系意法半导体以获取正确的操作步骤。

16.8 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。

16.8.1 RNG 控制寄存器 (RNG_CR)

RNG control register

偏移地址：0x000

复位值：0x0000 0000

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

位 31:6 保留，必须保持复位值

位 5 **CED**：时钟错误检测 (Clock error detection)

0：使能时钟错误检测

1：禁止时钟错误检测

当使能 RNG 时，不能实时使能或禁止时钟错误检测，即必须禁止 RNG 才能使能或禁止 CED。

位 4 保留，必须保持复位值

位 3 **IE**：中断使能 (Interrupt Enable)

0：禁止 RNG 中断

1：使能 RNG 中断 如果 RNG_SR 寄存器中 DRDY = “1”、SEIS = “1”或 CEIS = “1”，则存在待处理的中断。

位 2 **RNGEN**：真随机数发生器使能 (True random number generator enable)

0：禁止真随机数发生器。逻辑噪声源关闭，并会对由 RNG 时钟提供时钟信号的逻辑系统进行门控。

1：使能真随机数发生器。

位 1:0 保留，必须保持复位值

16.8.2 RNG 状态寄存器 (RNG_SR)

RNG status register

偏移地址：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SEIS	CEIS	Res.	Res.	SECS	CECS	DRDY
									rc_w0	rc_w0			r	r	r

位 31:7 保留，必须保持复位值

位 6 **SEIS**：种子错误中断状态 (Seed error interrupt status)

此位与 SECS 同时置 1，通过向其写入“0”来清零。

0：未检测到错误序列

1：至少检测到一个错误序列。有关详细信息，请参见 **SECS** 位说明。

如果 RNG_CR 寄存器中 IE =“1”，则存在待处理的中断。

位 5 **CEIS**：时钟错误中断状态 (Clock error interrupt status)

此位与 CECS 同时置 1，通过向其写入“0”来清零。

0：RNG 时钟正确 (fRNGCLK > fHCLK/16)

1：已检测到 RNG 时钟过慢 (fRNGCLK < fHCLK/16)

如果 RNG_CR 寄存器中 IE =“1”，则存在待处理的中断。

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：其中一个噪声源持续不变地输出了 64 位或以上的“0”或“1”，或者 32 个或以上的“01”或“10”。

位 1 **CECS**：时钟错误当前状态 (Clock error current status)

0：RNG 时钟正确 (fRNGCLK> fHCLK/16) 如果 CEIS 位置 1，则意味着已检测到时钟过慢并已恢复正常。

1：RNG 时钟过慢 (fRNGCLK< fHCLK/16)。

注：　只有 RNG_CR 寄存器中的 CED 位置 0 时，CECS 位才有效。

位 0 **DRDY**：数据就绪 (Data Ready)

0：RNG_DR 寄存器尚未有效，无可用随机数据。

1：RNG_DR 寄存器包含有效随机数据。

读取 RNG_DR 寄存器后，此位恢复到 0，直到生成新的随机值。

如果 RNG_CR 寄存器中的 IE =“1”，则在 DRDY =“1”时生成中断。

16.8.3 RNG 数据寄存器 (RNG_DR)

RNG data register

偏移地址：0x008

复位值：0x0000 0000

RNG_DR 寄存器是只读寄存器，在读取时提供 32 位随机数值。读取该寄存器后，如果输出 FIFO 为空，则该寄存器将在 42 个 RNG 时钟周期之后提供一个新的随机值。

当 DRDY = “1” 时，即使 RNGEN = “0”，该寄存器的内容也是有效的。

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

位 31:0 **RNDATA[31:0]**：随机数据 (Random data)
DRDY = “1” 时有效的 32 位随机数据。DRDY = “0” 时，RNDATA 值为 0。

16.8.4 RNG 寄存器映射

表 100 给出了 RNG 寄存器映射和复位值。

表 100. RNG 寄存器映射和复位映射

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
0x000	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.	CED	Res.	IE	RNGEN	Res.	Res.
	Reset value																											0		0	0		
0x004	RNG_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SEIS	CEIS	Res.	Res.	SECS	CECS	DRDY
	Reset value																										0	0			0	0	0
0x008	RNG_DR	RNDATA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17 高级控制定时器（TIM1 和 TIM8）

17.1 TIM1 和 TIM8 简介

高级控制定时器（TIM1 和 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。

此类定时器可用于多种用途，包括测量输入信号的脉冲宽度（输入捕获），或者生成输出波形（输出比较、PWM 和带死区插入的互补 PWM）。

使用定时器预分频器和 RCC 时钟控制器预分频器，可将脉冲宽度和波形周期从几微秒调制到几毫秒。

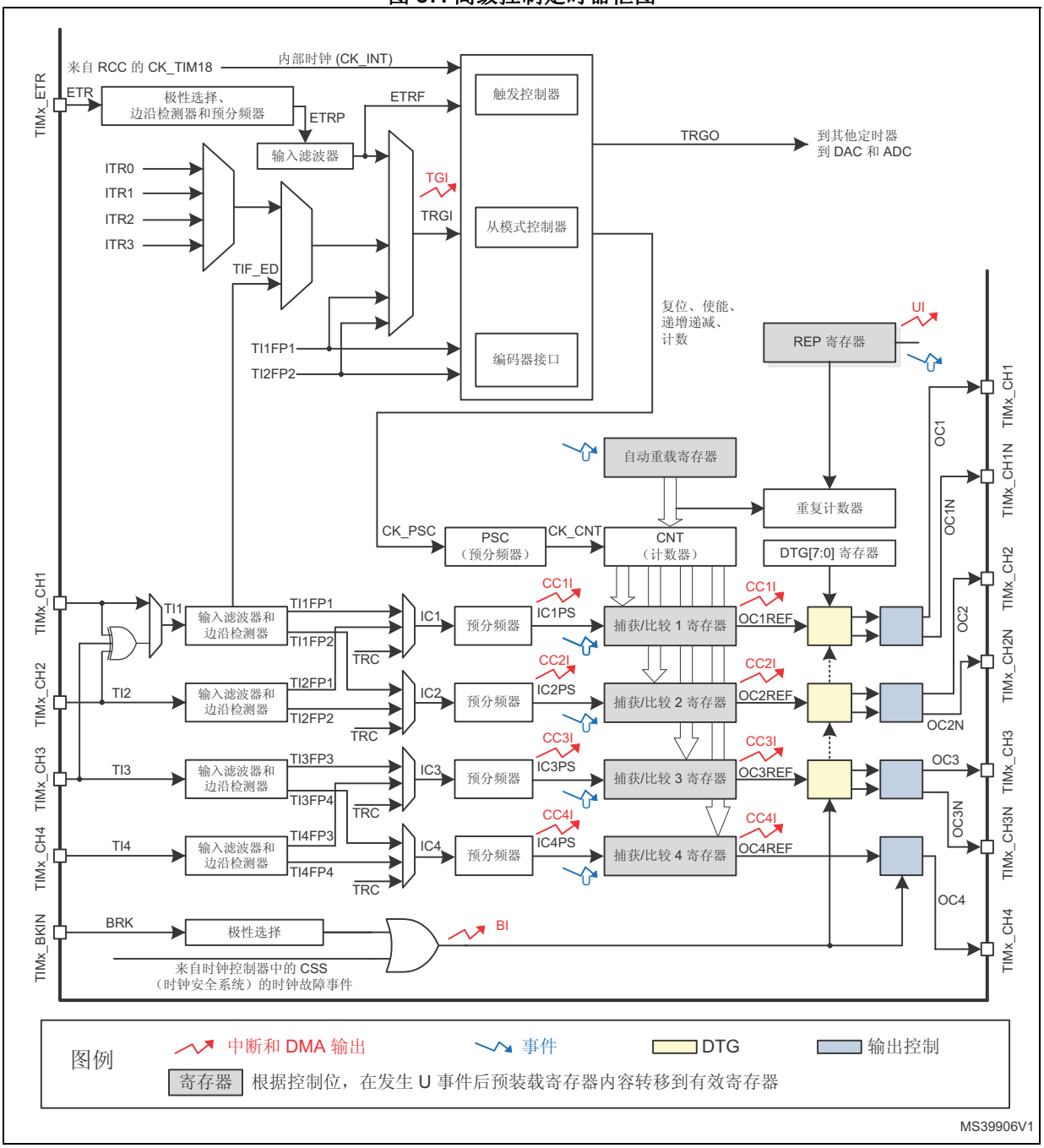
高级控制定时器（TIM1 和 TIM8）和通用 (TIMx) 定时器彼此完全独立，不共享任何资源。如*第 17.3.20 节*中所述，它们可以同步操作。

17.2 TIM1 和 TIM8 主要特性

TIM1 和 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 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
- 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。
- 用于将定时器的输出信号置于复位状态或已知状态的断路输入。
- 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
  - 断路输入
- 支持定位用增量（正交）编码器和霍尔传感器电路。
- 外部时钟触发输入或逐周期电流管理。

图 87. 高级控制定时器框图



17.3 TIM1 和 TIM8 功能说明

17.3.1 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立即传送到影子寄存器，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 16 位寄存器 TIMx_PSC 所控制的 16 位计数器。由于该控制寄存器具有缓冲功能，因此预分频器可实现实时更改。而新的预分频比将在下一更新事件发生时被采用。

图 88 和 *图 89* 以一些示例说明在预分频比实时变化时计数器的行为：

图 88. 预分频器分频由 1 变为 2 时的计数器时序图

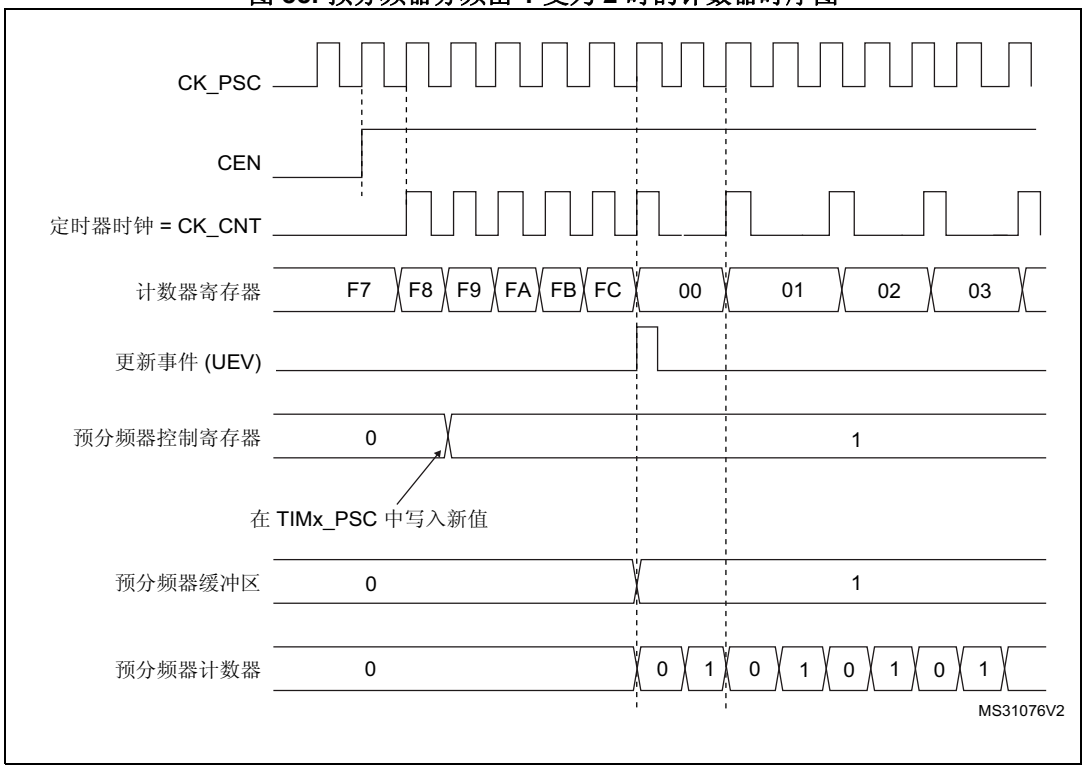
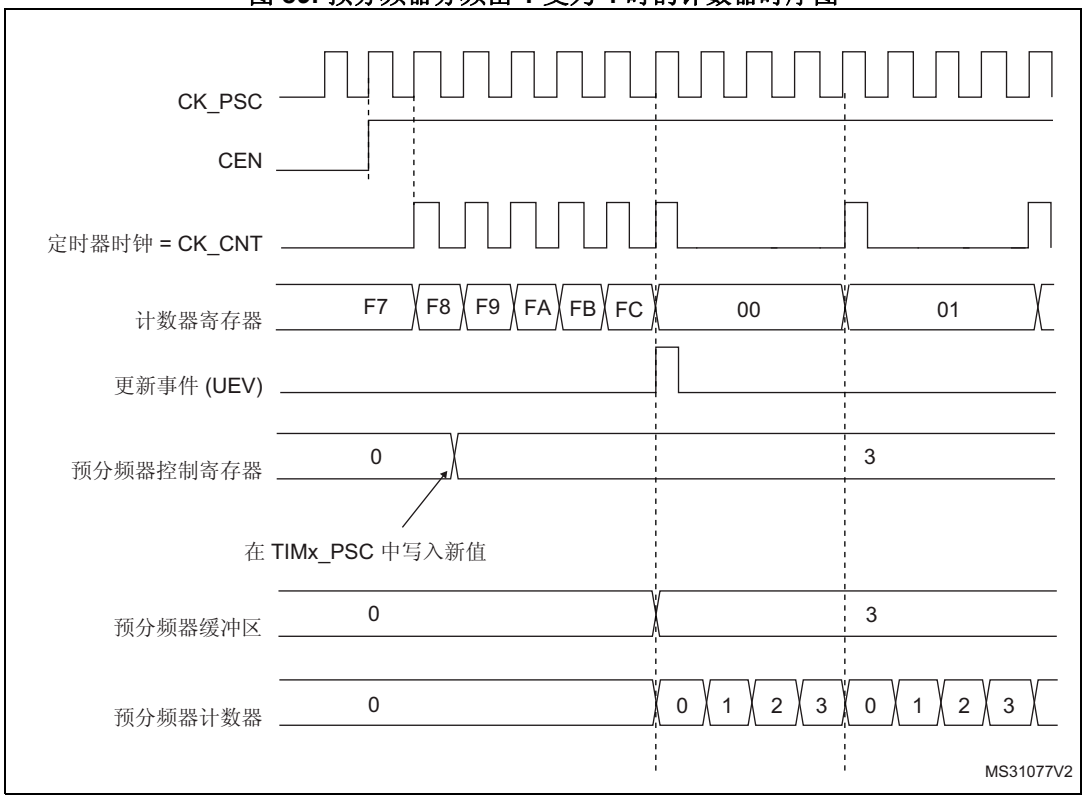


图 89. 预分频器分频由 1 变为 4 时的计数器时序图



17.3.2 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设定的次数加一次 (TIMx_RCR+1) 后，将生成更新事件 (UEV)。否则，将在每次计数器上溢时产生更新事件。

将 TIMx_EGR 寄存器的 UG 位置 1（通过软件或使用从模式控制器）时，也将产生更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变）。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。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 90. 计数器时序图，1 分频内部时钟

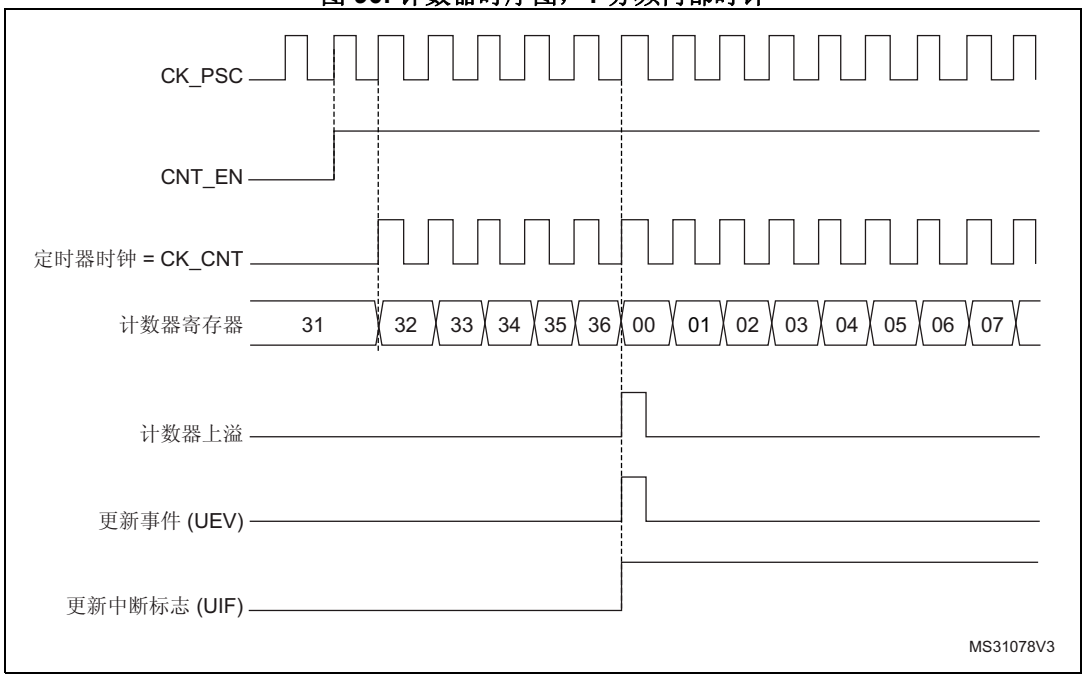


图 91. 计数器时序图，2 分频内部时钟

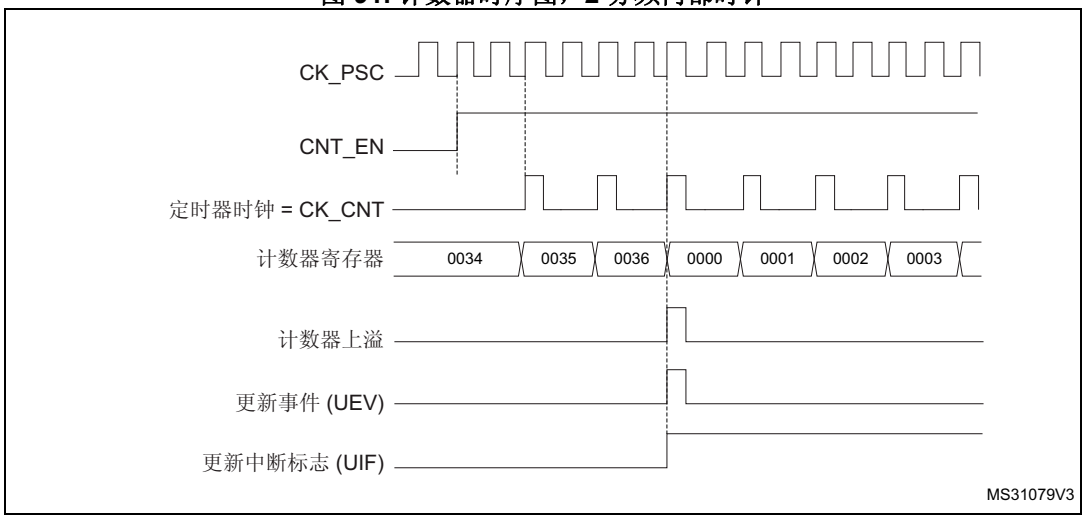


图 92. 计数器时序图，4 分频内部时钟

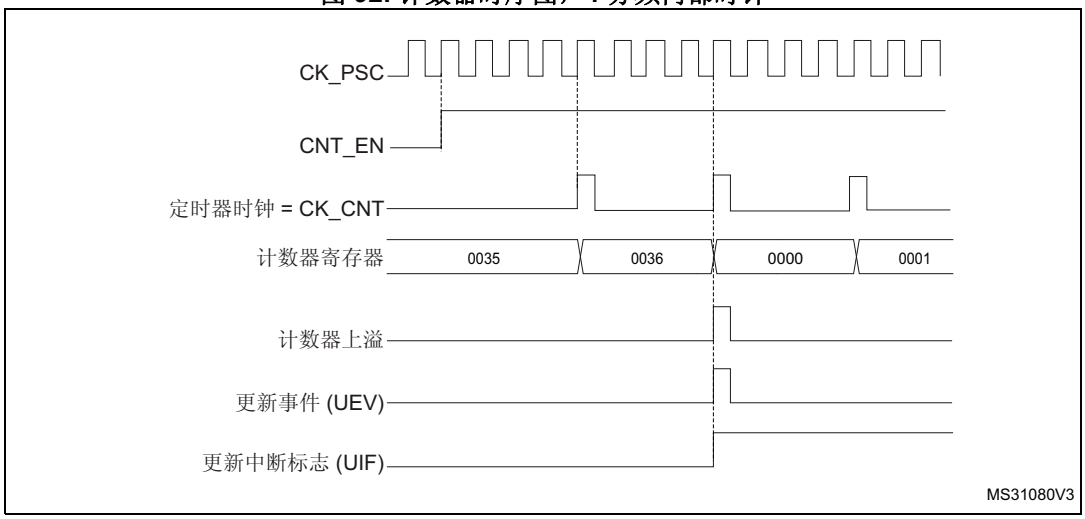


图 93. 计数器时序图，N 分频内部时钟

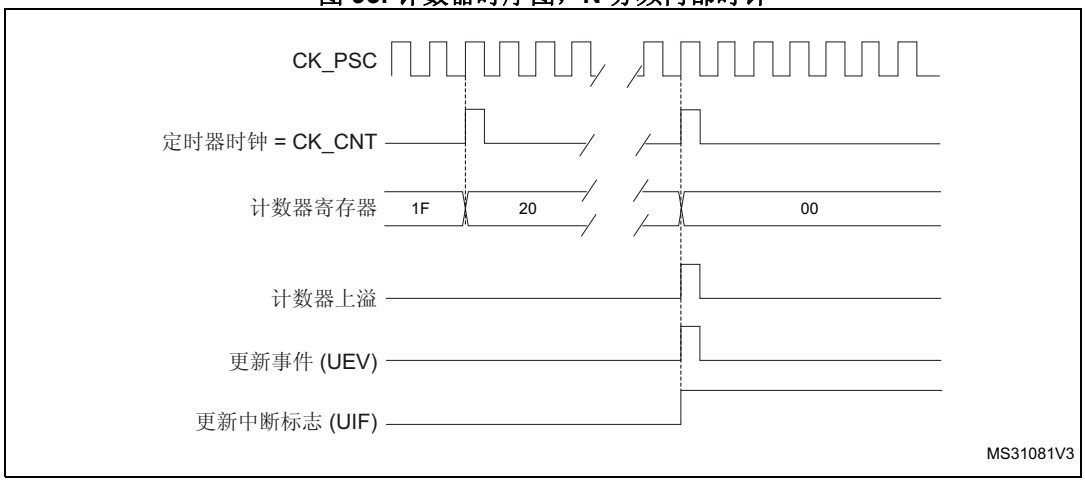


图 94. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载）

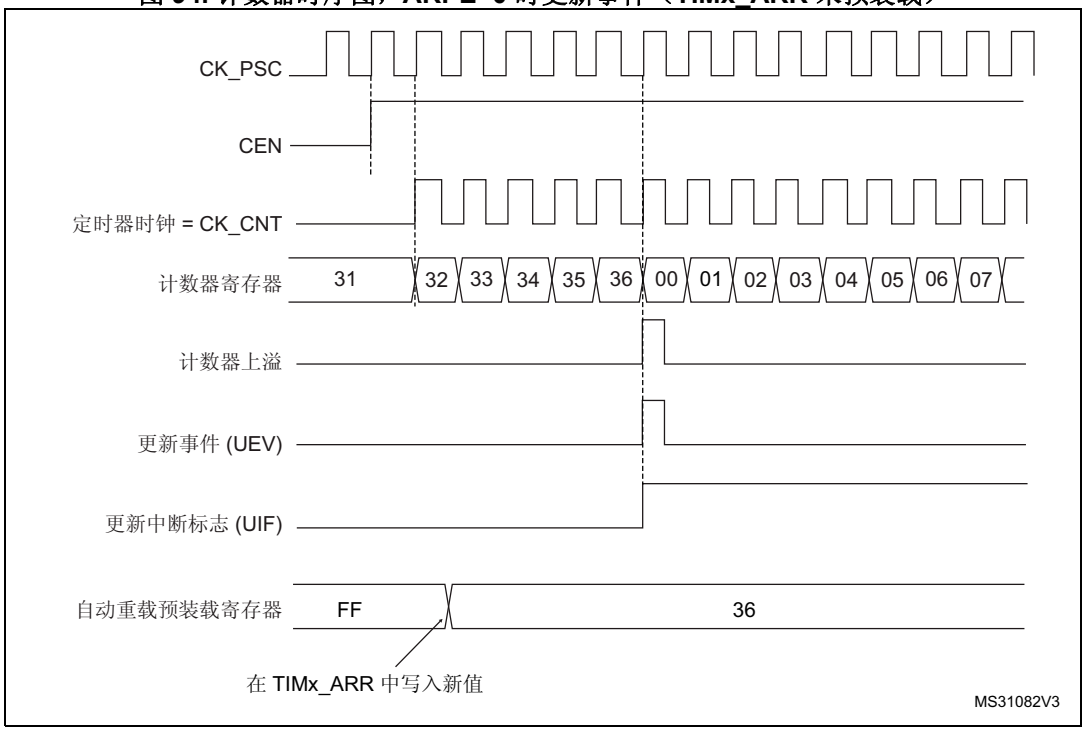
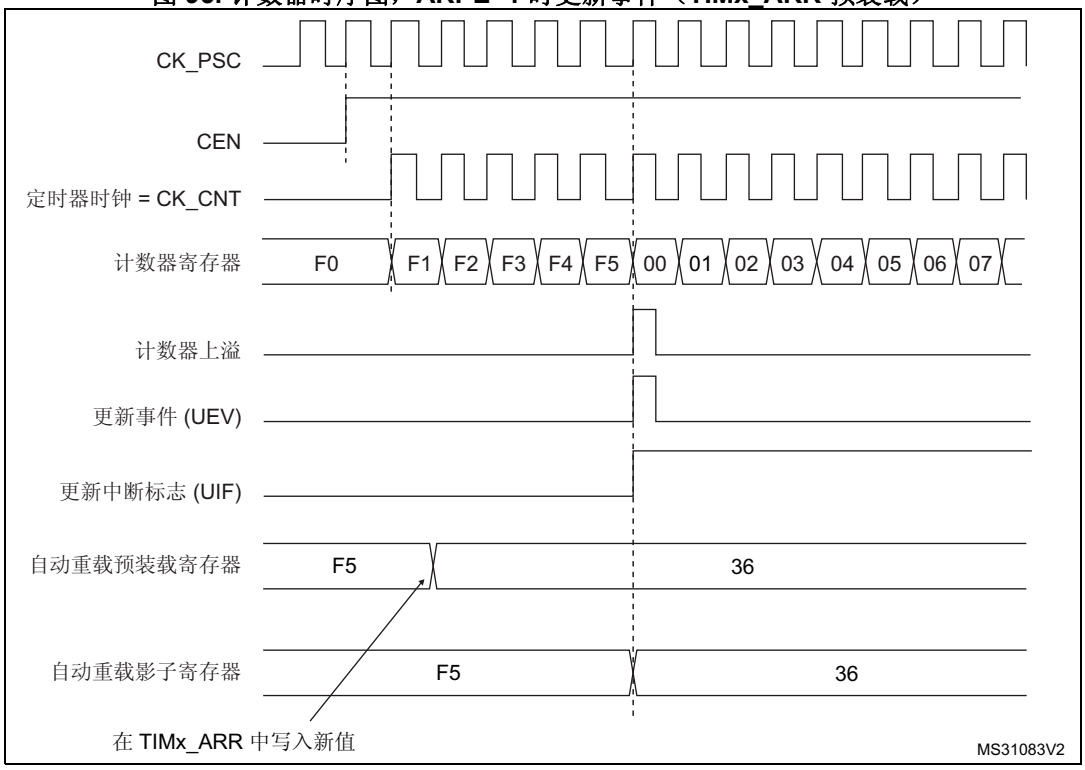


图 95. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 预装载）



递减计数模式

在递减计数模式下，计数器从自动重载值（TIMx_ARR 寄存器的内容）开始递减计数到 0，然后重新从自动重载值开始计数并生成计数器下溢事件。

如果使用重复计数器，则当递减计数的重复次数达到重复计数器寄存器中设定的次数加一次 (TIMx_RCR+1) 后，将生成更新事件 (UEV)。否则，将在每次计数器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）
- 自动重载有效寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，自动重载寄存器会在计数器重载之前得到更新，因此，下一个计数周期就是我们所希望的新的周期长度

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 96. 计数器时序图，1 分频内部时钟

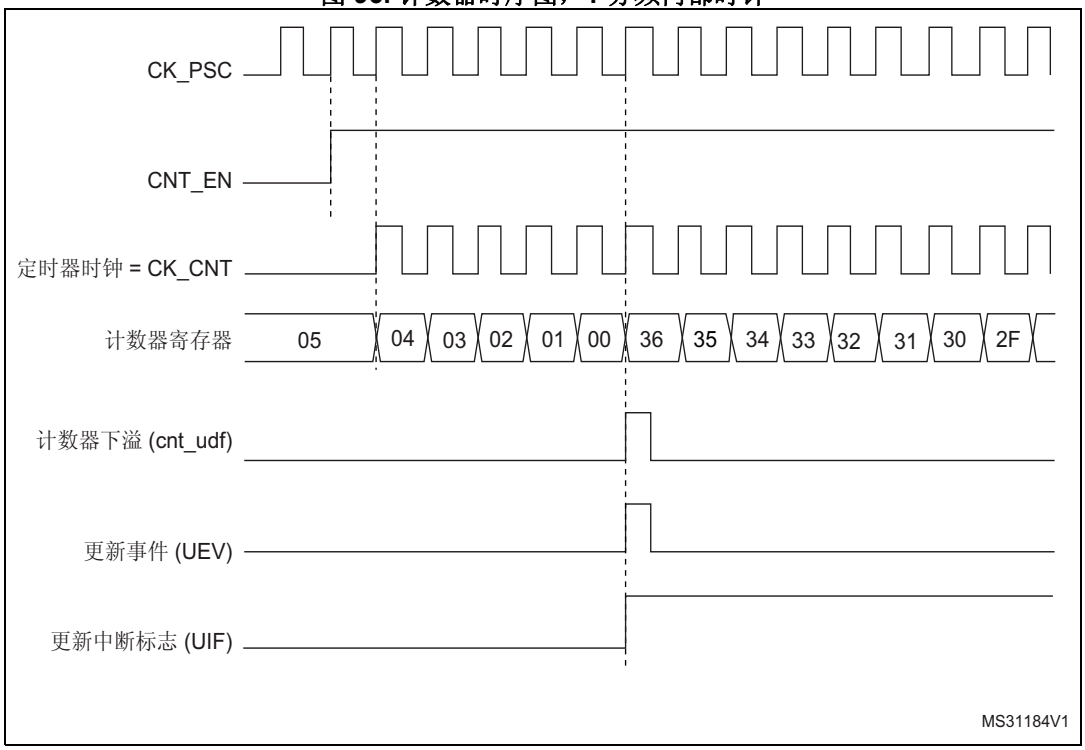


图 97. 计数器时序图，2 分频内部时钟

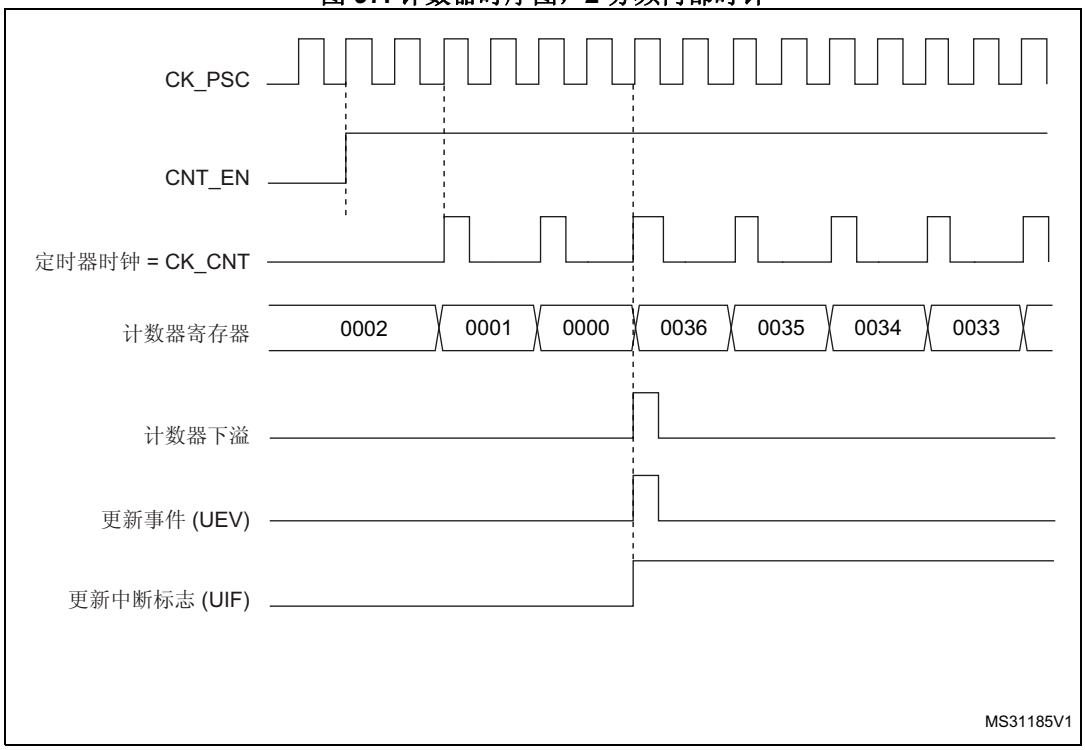


图 98. 计数器时序图，4 分频内部时钟

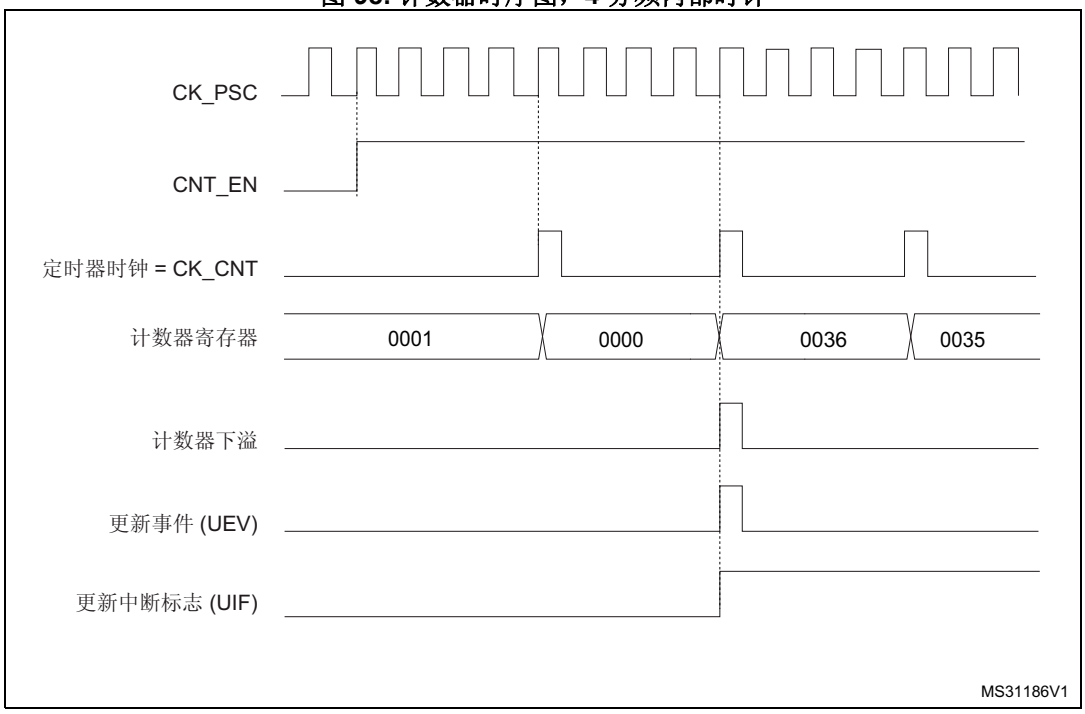


图 99. 计数器时序图，N 分频内部时钟

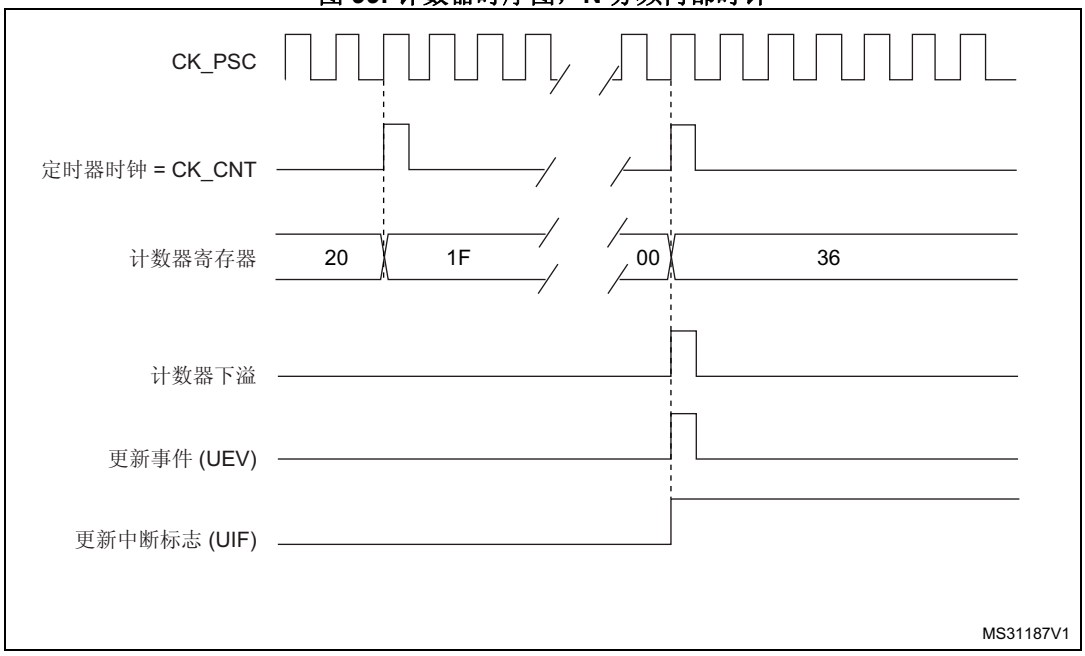
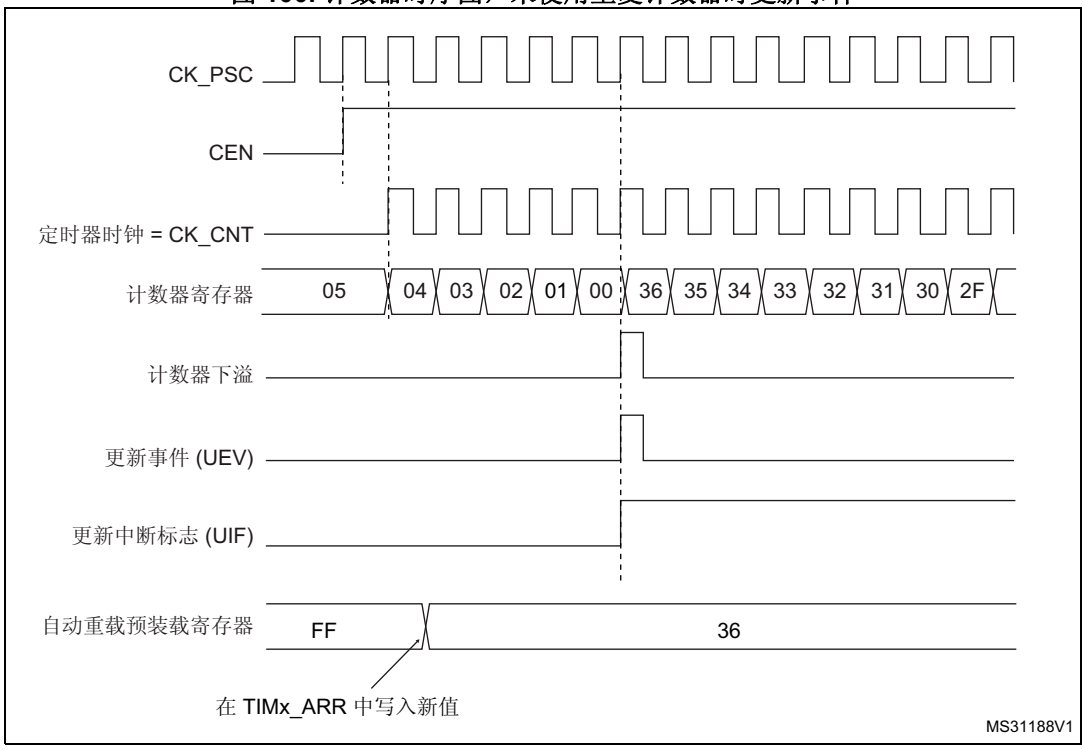


图 100. 计数器时序图，未使用重复计数器时更新事件



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。

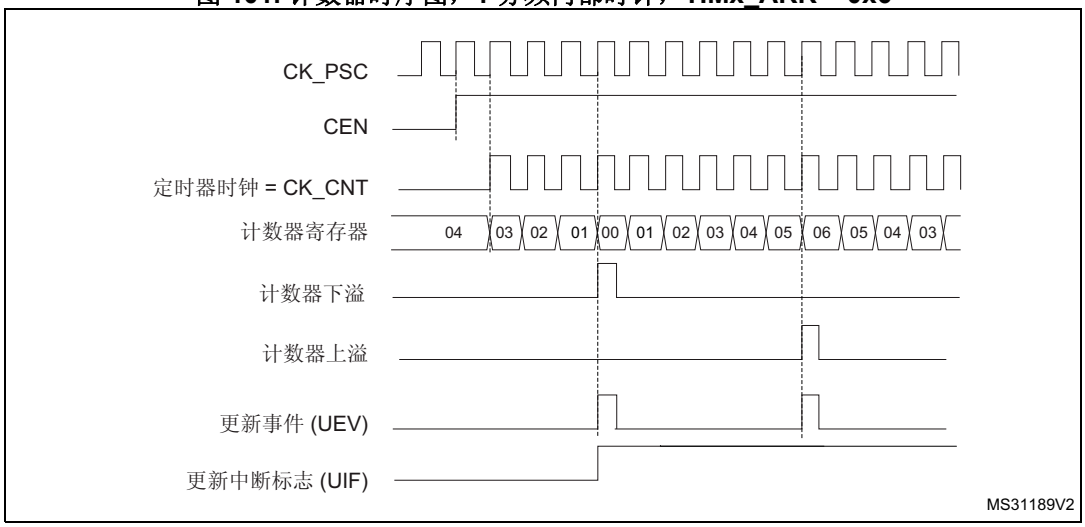
此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会产生 UEV 更新事件，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

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）
- 自动重载有效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 101. 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 17.4 节：TIM1 和 TIM8 寄存器*）。

图 102. 计数器时序图，2 分频内部时钟

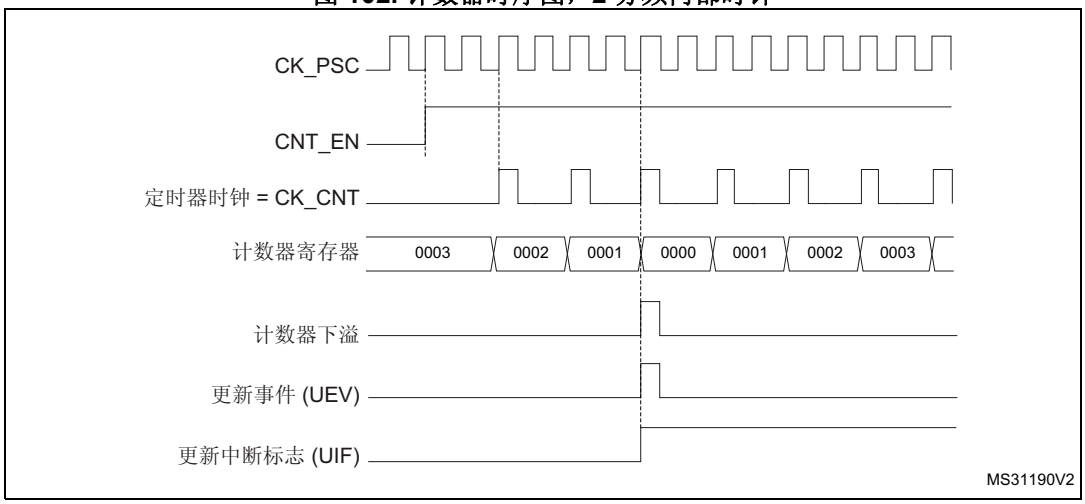
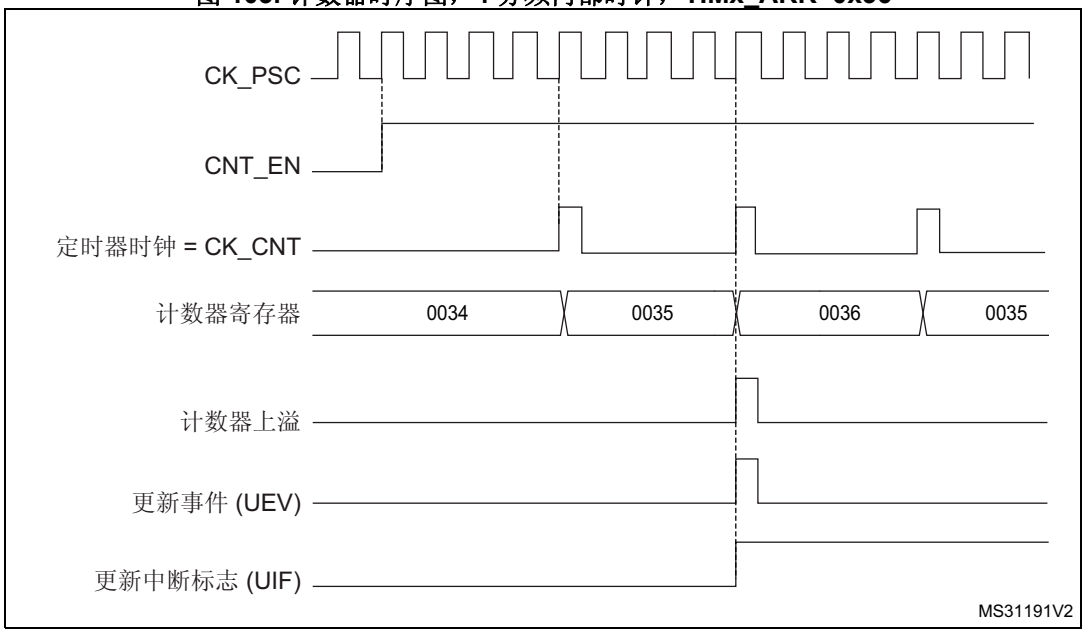


图 103. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36



1. 中心对齐模式 2 或模式 3 与上溢 UIF 结合使用。

图 104. 计数器时序图，N 分频内部时钟

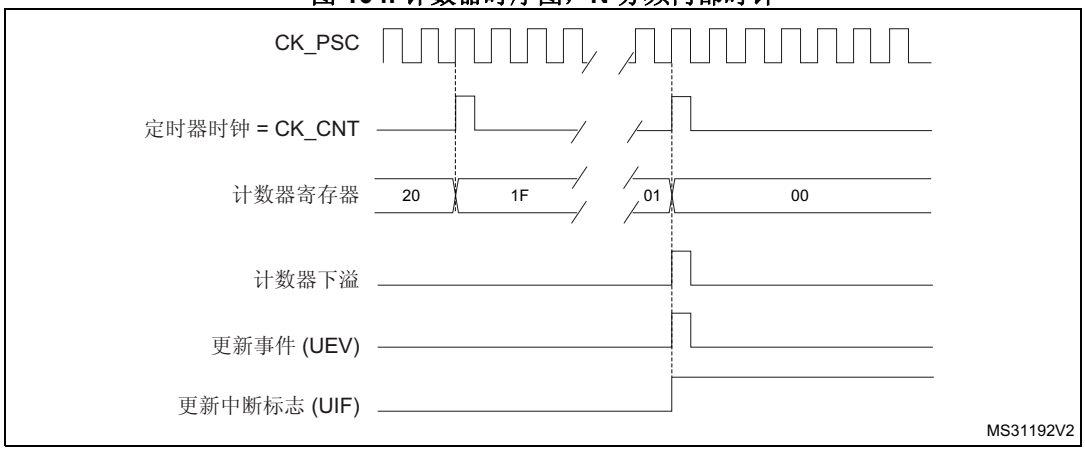


图 105. 计数器时序图，ARPE=1 时更新事件（计数器下溢）

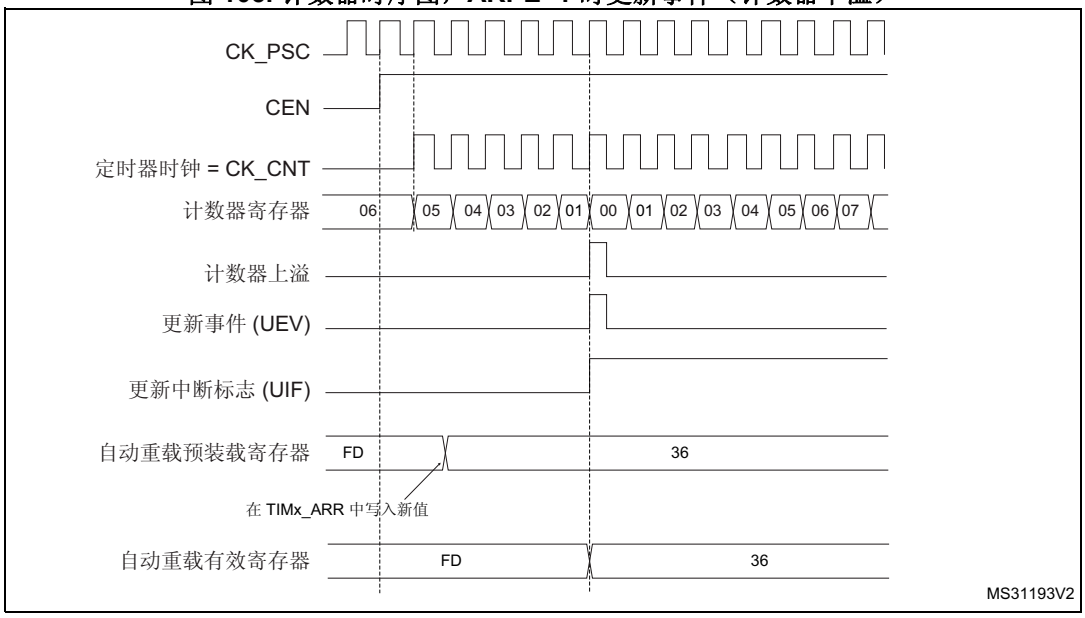
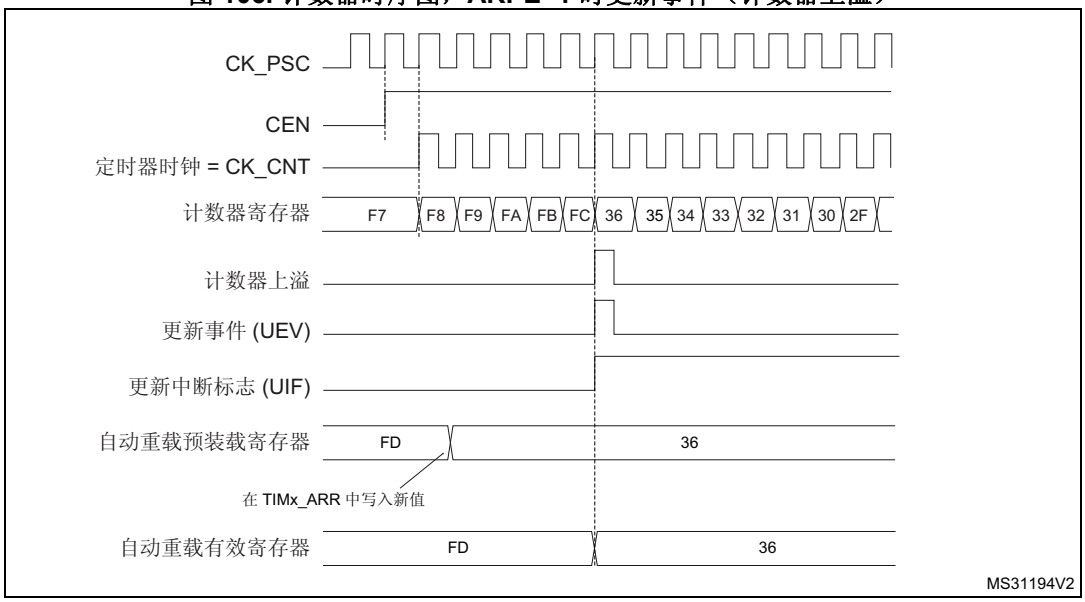


图 106. 计数器时序图，ARPE=1 时更新事件（计数器上溢）



17.3.3 重复计数器

*第 17.3.1 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）。

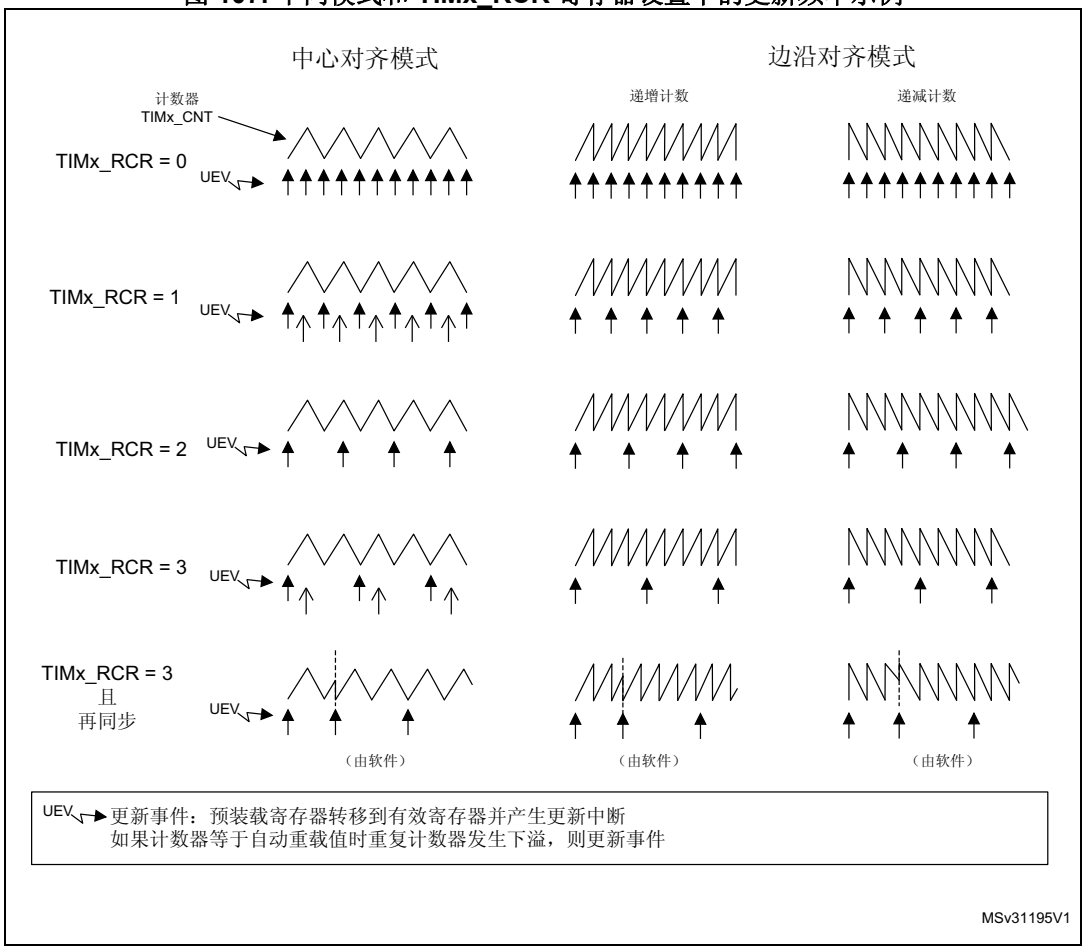
重复计数器在下列情况下递减：

- 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 128 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 107*）。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 107. 不同模式和 TIMx_RCR 寄存器设置下的更新频率示例



17.3.4 时钟选择

计数器时钟可由下列时钟源提供：

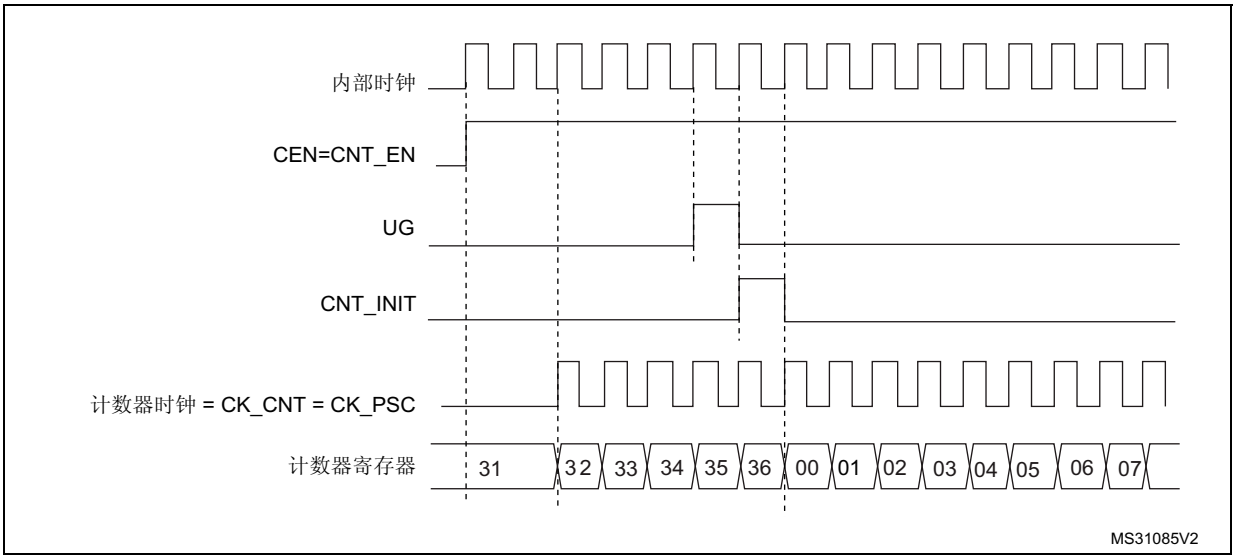
- 内部时钟 (CK_INT)
- 外部时钟模式 1：外部输入引脚
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入 (ITRx)：使用一个定时器作为另一定时器的预分频器，例如，可将定时器 1 配置为定时器 2 的预分频器。更多详细信息，请参见*将一个定时器用作另一个定时器的预分频器*。

内部时钟源 (CK_INT)

如果禁止从模式控制器 (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 108 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）。

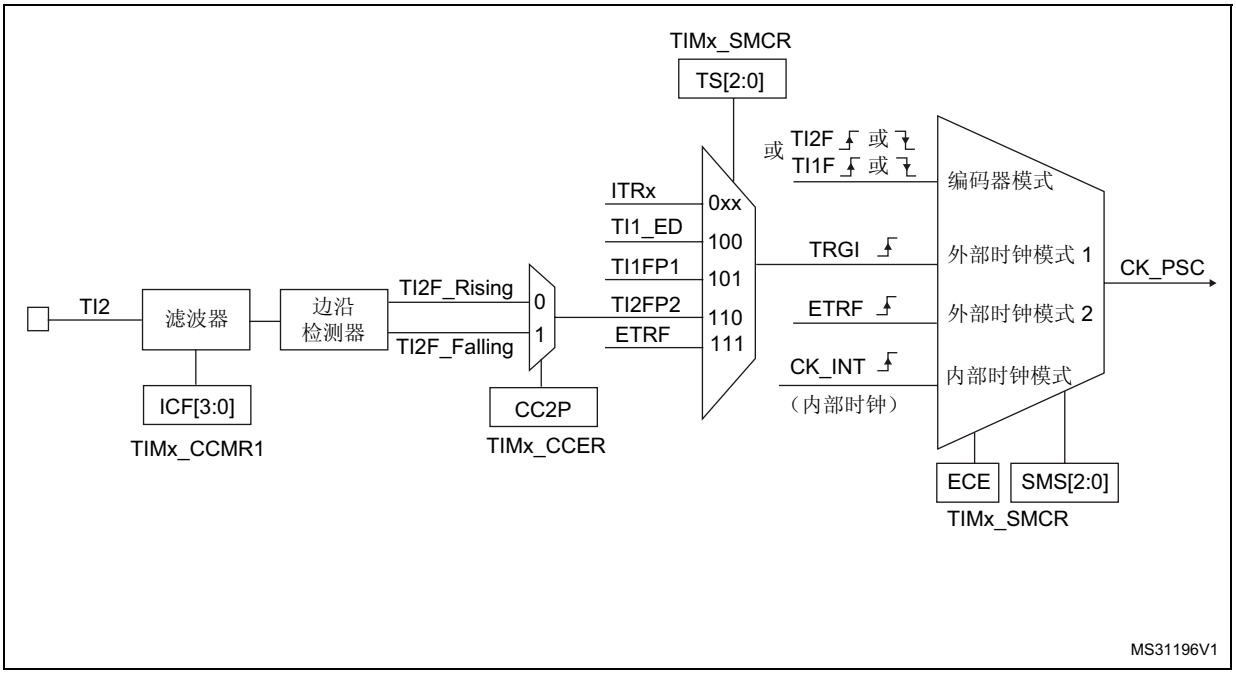
图 108. 正常模式下的控制电路，1 分频内部时钟



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 109. 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：

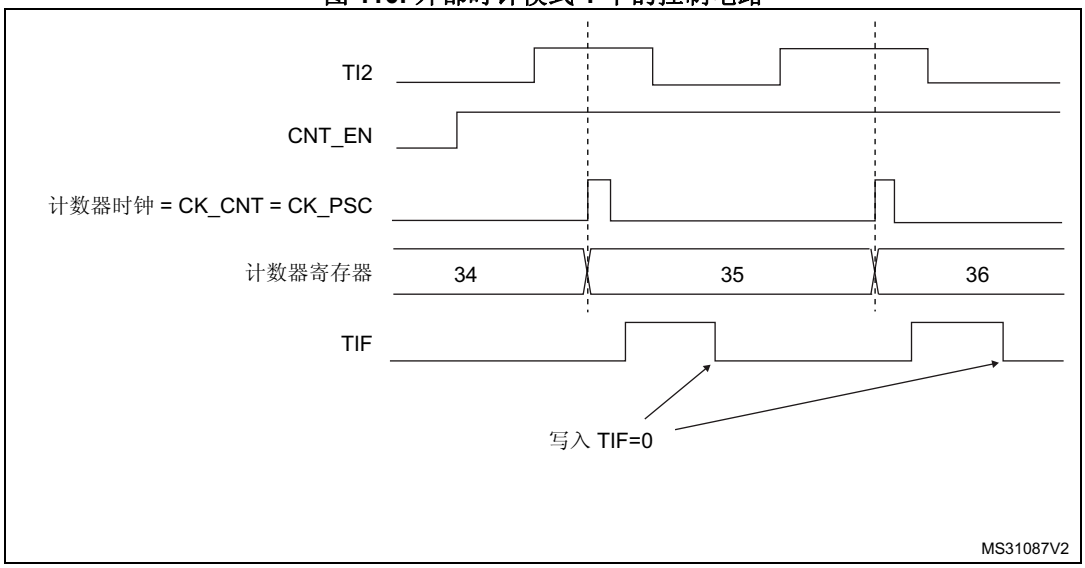
1. 通过在 TIMx_CCMR1 寄存器中写入 CC2S = “01” 来配置通道 2，使其能够检测 TI2 输入的上升沿。
2. 通过在 TIMx_CCMR1 寄存器中写入 IC2F[3:0] 位来配置输入滤波带宽（如果不需要任何滤波器，请保持 IC2F=0000）。
3. 通过在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=0 来选择上升沿极性。
4. 通过在 TIMx_SMCR 寄存器中写入 SMS=111，使定时器在外部时钟模式 1 下工作。
5. 通过在 TIMx_SMCR 寄存器中写入 TS=110 来选择 TI2 作为触发输入源。
6. 通过在 TIMx_CR1 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 110. 外部时钟模式 1 下的控制电路



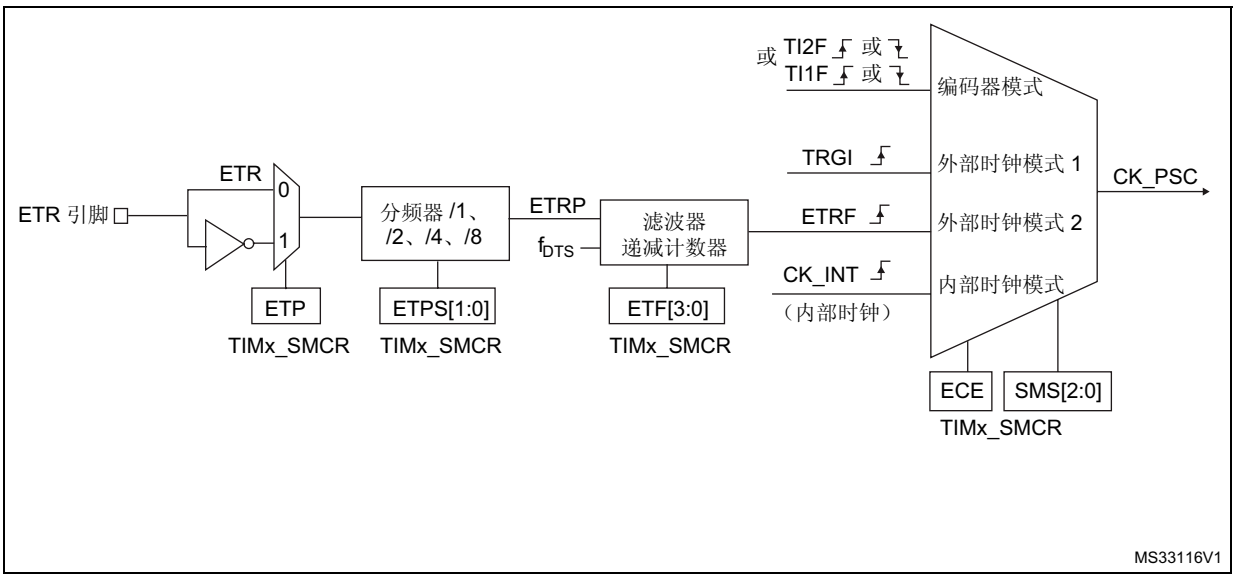
外部时钟源模式 2

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 111 简要介绍了外部触发输入模块。

图 111. 外部触发输入模块



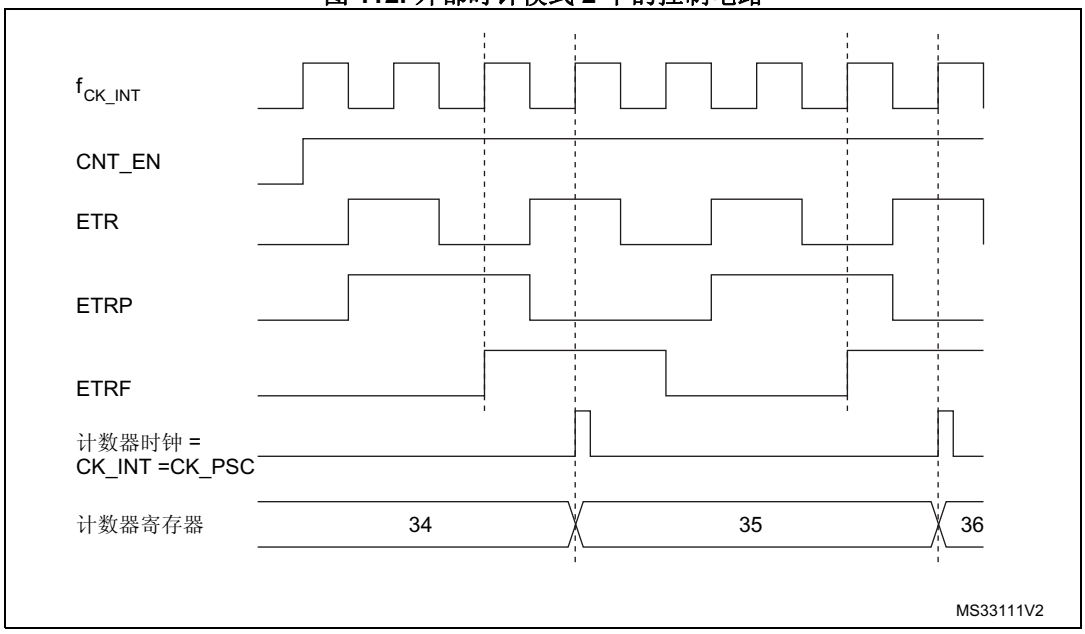
例如，要使递增计数器在 ETR 每出现 2 个上升沿时计数，请执行以下步骤：

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 112. 外部时钟模式 2 下的控制电路



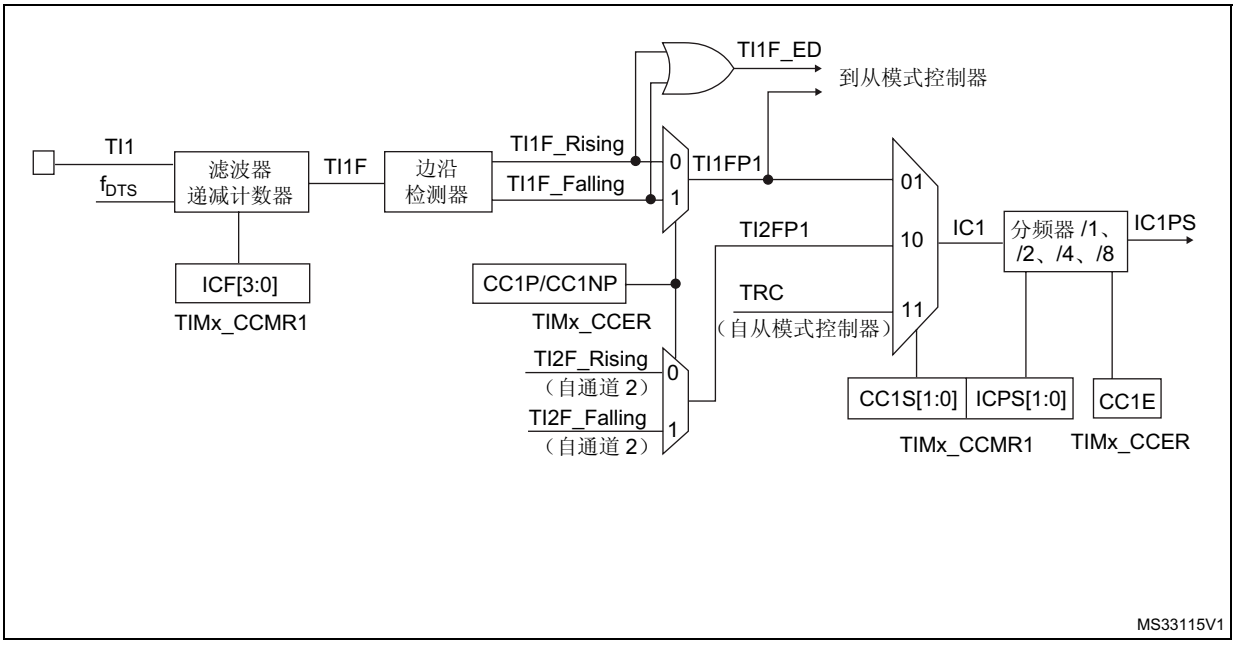
17.3.5 捕获/比较通道

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 113 到*图 116* 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 113. 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 114. 捕获/比较通道 1 主电路

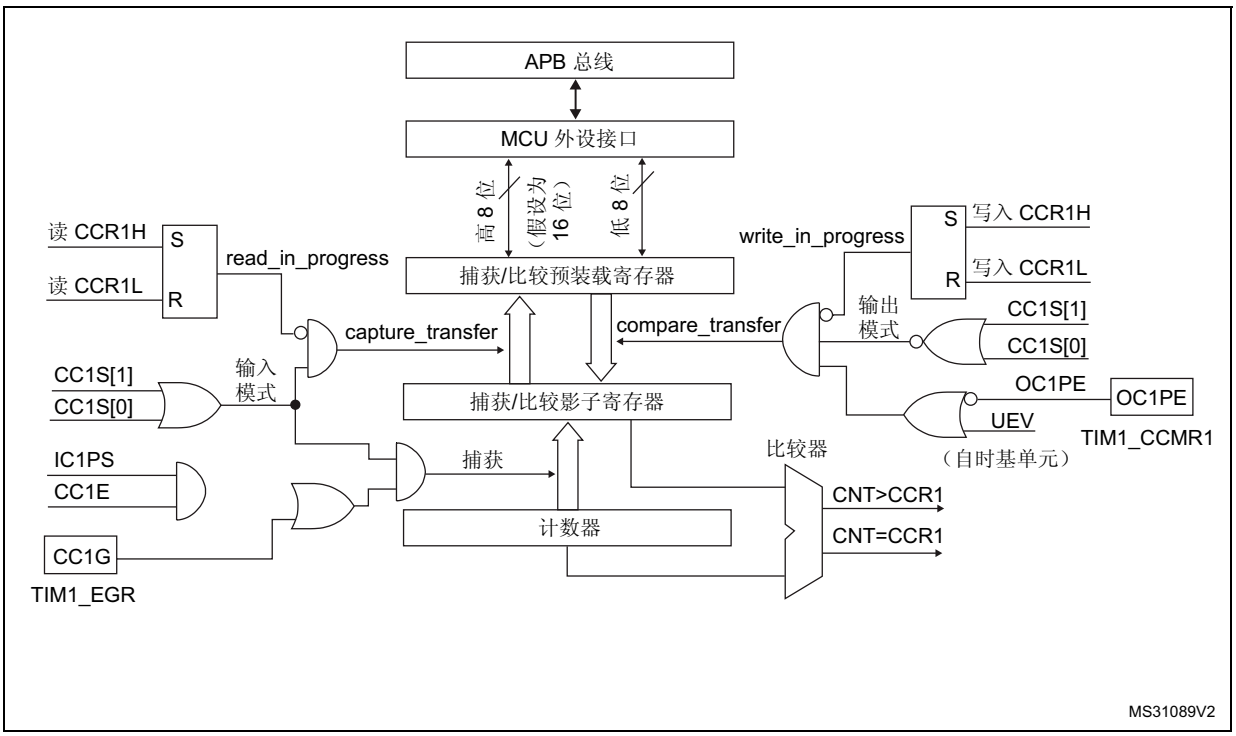


图 115. 捕获/比较通道的输出阶段（通道 1 到 3）

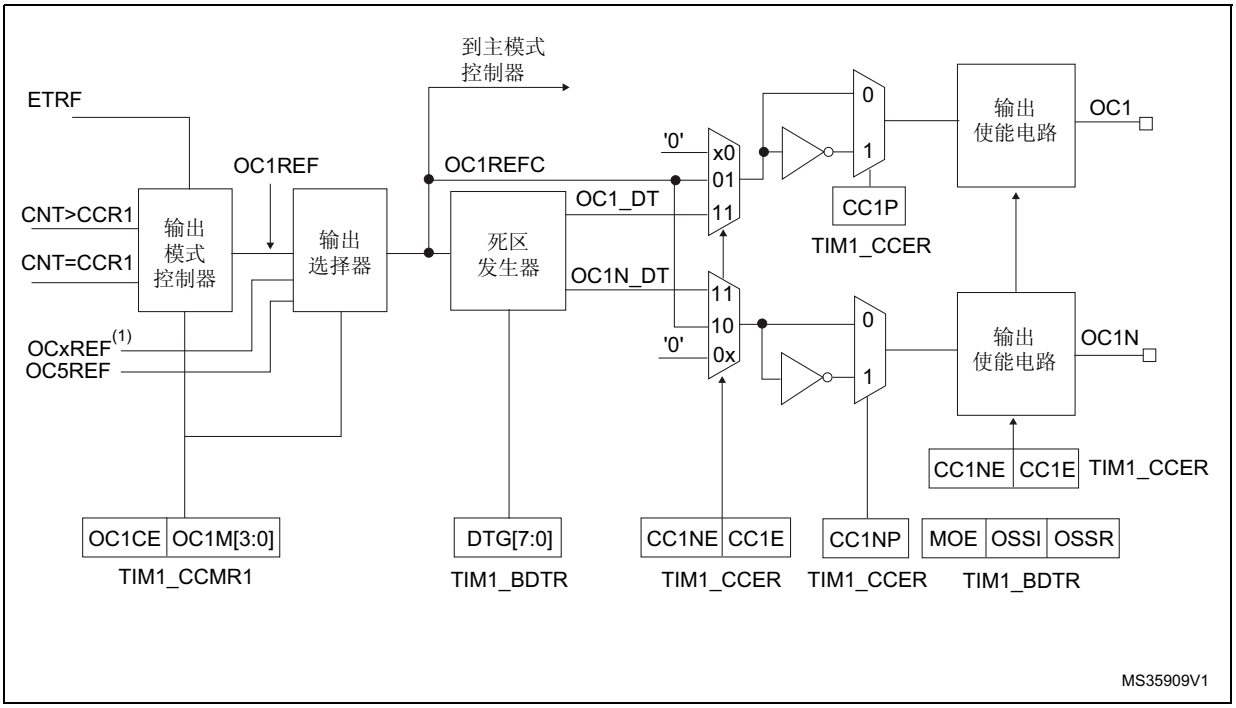
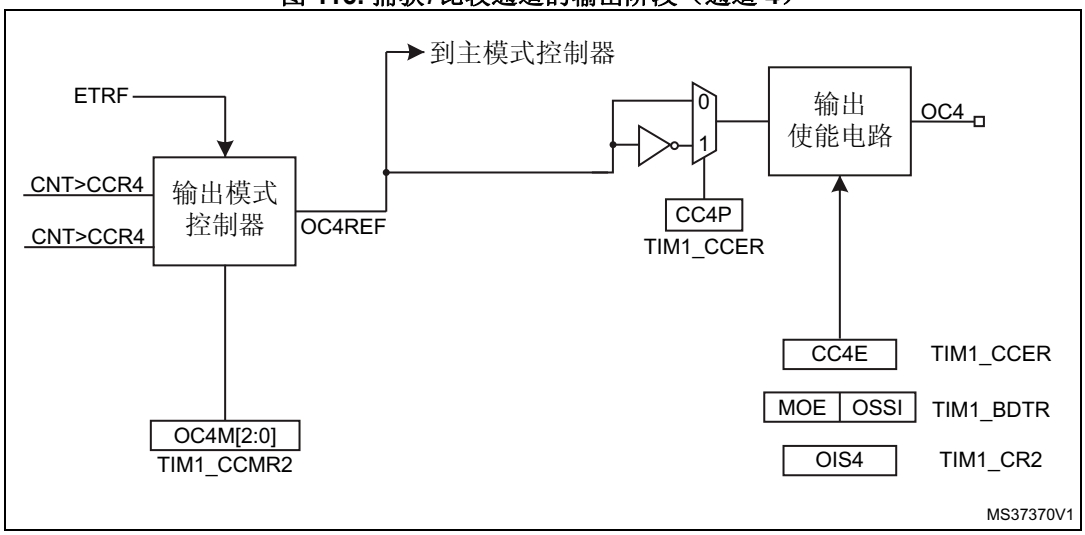


图 116. 捕获/比较通道的输出阶段（通道 4）



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。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。

17.3.6 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 输入，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。*

17.3.7 PWM 输入模式

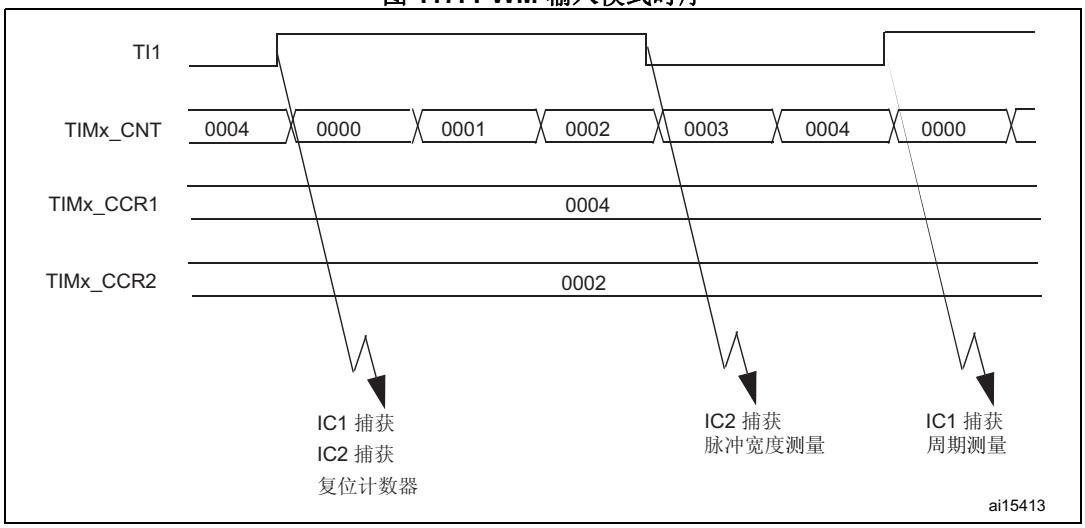
此模式是输入捕获模式的一个特例。其实现步骤与输入捕获模式基本相同，仅存在以下不同之处：

- 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“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 100。
- 使能捕获：向 TIMx_CCER 寄存器中的 CC1E 位和 CC2E 位写入“1”。

图 117. PWM 输入模式时序



17.3.8 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 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。

17.3.9 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。

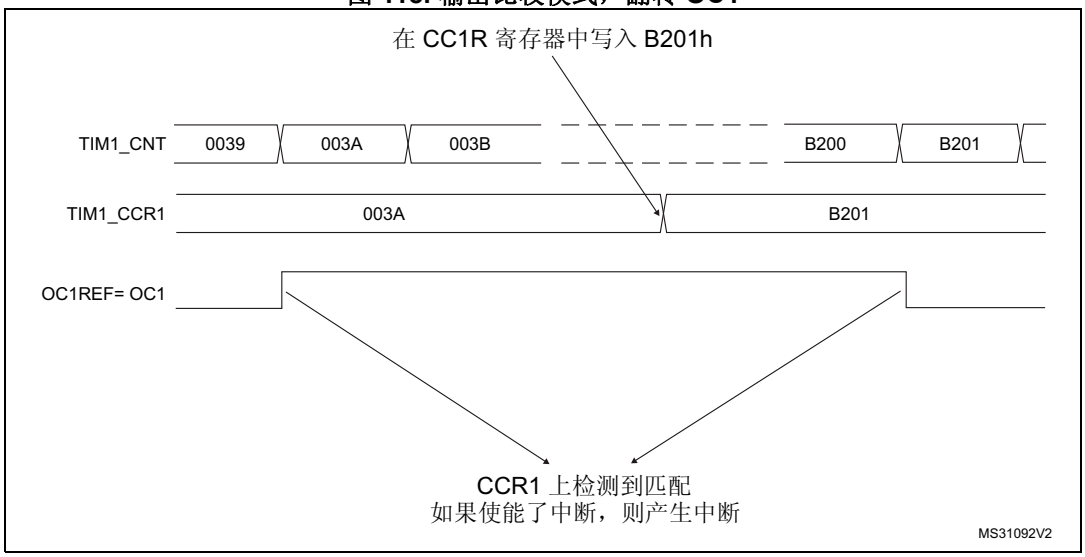
在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出毫无影响。同步的精度可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

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 118* 给出了一个示例。

图 118. 输出比较模式，翻转 OC1



17.3.10 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。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。

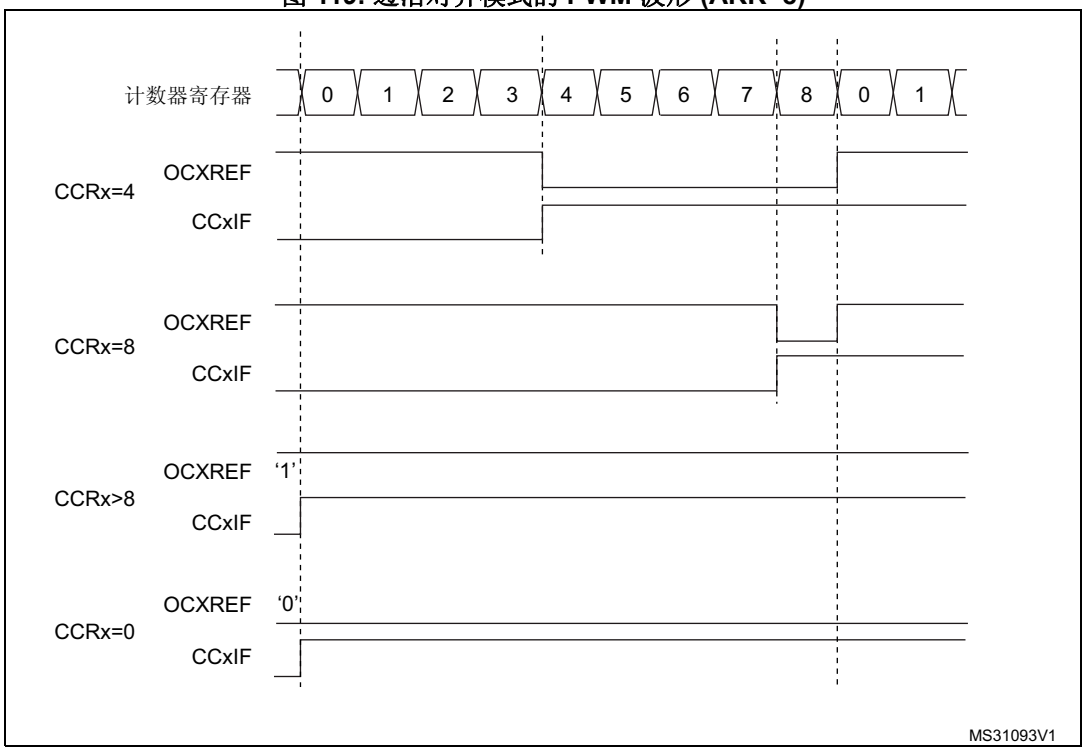
PWM 边沿对齐模式

- 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*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 119* 举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

图 119. 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*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。

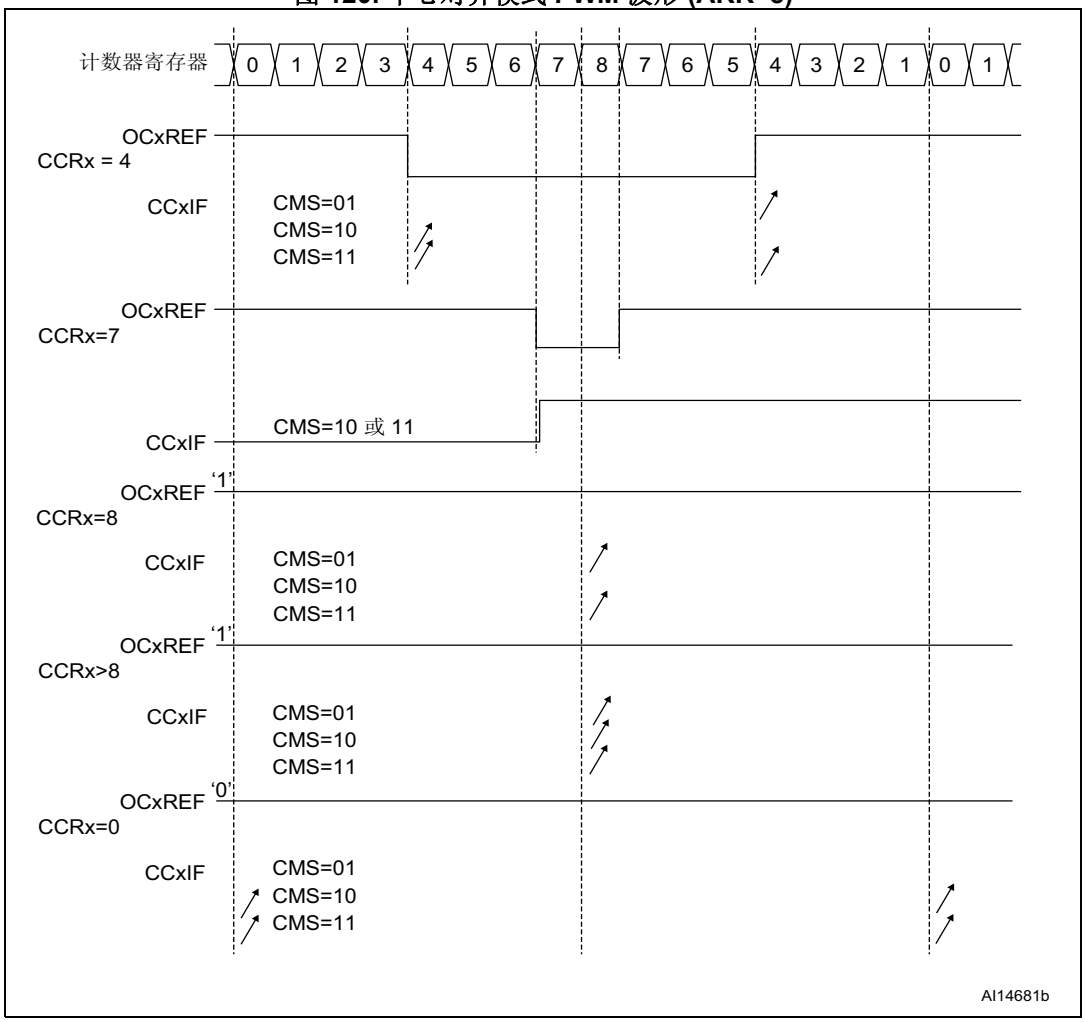
PWM 中心对齐模式

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*中心对齐模式（递增/递减计数）*。

图 120 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 120. 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。

17.3.11 互补输出和死区插入

高级控制定时器（TIM1 和 TIM8）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*图 103*。应当注意，切换至 IDLE（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。TIMx_BDTR 寄存器中的 DTG[7:0] 位用于控制所有通道的死区生成。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 121. 带死区插入的互补输出

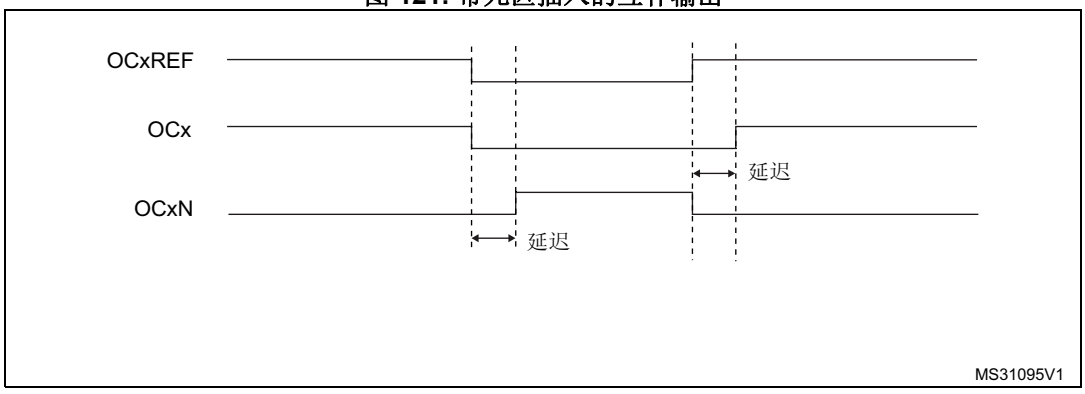


图 122. 延迟时间大于负脉冲宽度的死区波形

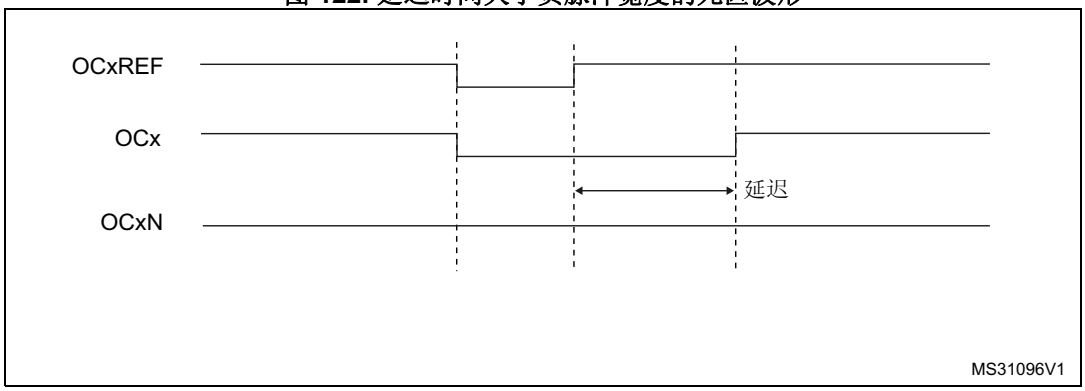
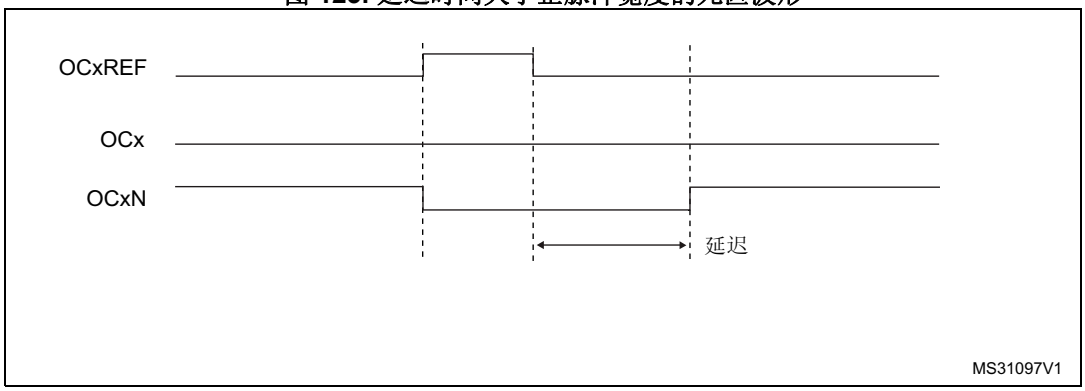


图 123. 延迟时间大于正脉冲宽度的死区波形



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 17.4.18 节：TIM1 和 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。*

17.3.12 使用断路功能

使用断路功能时，根据其他控制位（TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位以及 TIMx_CR2 寄存器中的 OISx 和 OISxN 位）修改输出使能信号和无效电平。任何情况下，OCx 和 OCxN 输出都不能同时置为有效电平。更多详细信息，请参见*图 103*。

断路源可以是断路输入引脚，也可以是时钟故障事件，后者由复位时钟控制器中的时钟安全系统 (CSS) 生成。有关时钟安全系统的详细信息，请参见*第 6.2.7 节：时钟安全系统 (CSS)*。

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 位置 1，可使能断路功能。断路输入的极性可通过该寄存器中的 BKP 位来选择。BKE 和 BKP 位可同时修改。对 BKE 和 BKP 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。

发生断路（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、空闲状态或复位状态（通过 OSSI 位进行选择）。即使 MCU 振荡器关闭，该功能仍然有效。
- MOE=0 时，将以 TIMx_CR2 寄存器 OISx 位中设定的电平驱动每个输出通道。如果 OSSI=0，则定时器将释放使能输出，否则使能输出始终保持高电平。
- 使用互补输出时：
  - 输出首先置于复位状态或无效状态（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 位中设定的电平驱动输出。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长一些（约 2 个 ck_tim 时钟周期）。
  - 如果 OSSI=0，则定时器会释放使能输出，否则只要 CCxE 位或 CCxNE 位处于高电平，使能输出就会保持或变为高电平。
- 将断路状态标志（TIMx_SR 寄存器中的 BIF 位）置 1。如果 TIMx_DIER 寄存器中的 BIE 位置 1，可产生中断。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。这一特性有许多用处，比如，可用于实现调节器的功能。否则，MOE 将始终保持低电平，直到再次向该位写入“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 清零。*

断路可由 BRK 输入生成，该输入具有可编程极性，其使能位 BKE 位于 TIMx_BDTR 寄存器中。

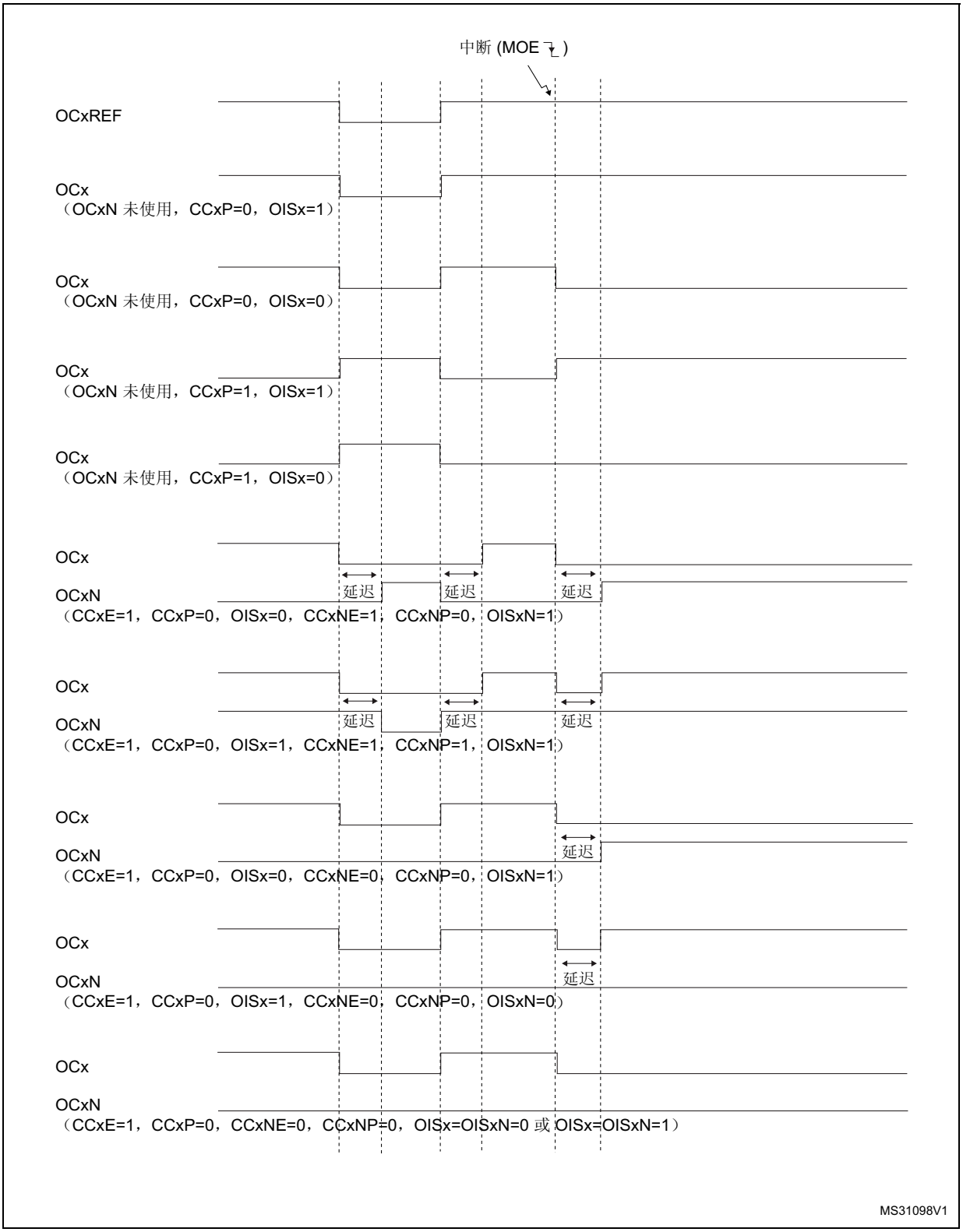
断路有以下两种生成方案：

- 使用 BRK 输入生成，该输入具有可编程极性，其使能位 BKE 位于 TIMx_BDTR 寄存器中。
- 由软件通过 TIMx_EGR 寄存器中的 BG 位生成。

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）。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 17.4.18 节：TIM1 和 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 124 所示为输出对断路响应行为的示例。

图 124. 响应断路的输出行为



17.3.13 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”），可使 OCxREF 信号变为低电平。OCxREF 信号将保持低电平，直到发生下一更新事件 (UEV)。

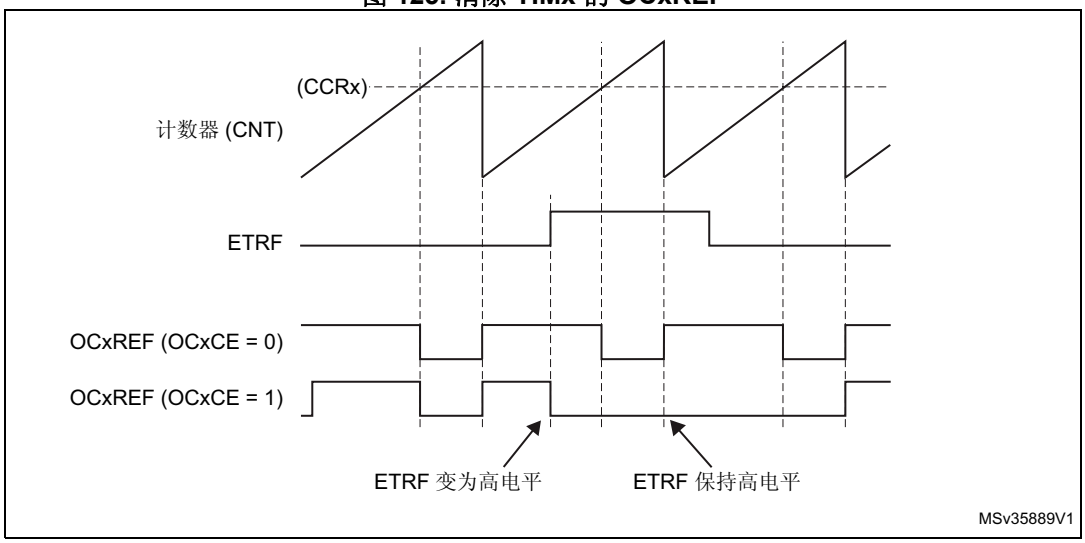
此功能仅能用于输出比较模式和 PWM 模式，而不适用于强制输出模式。

例如，ETR 信号可以连接到比较器的输出，用于控制电流。此时，ETR 必须如下配置：

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 125 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 125. 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。*

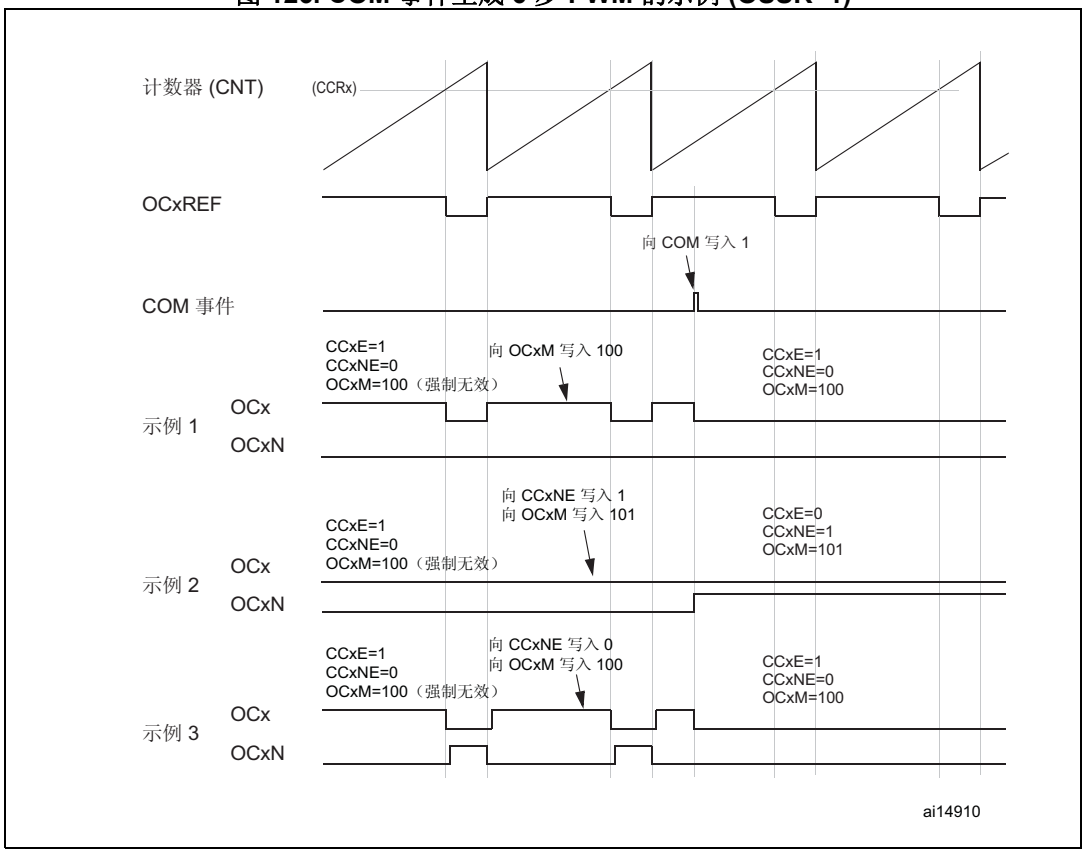
17.3.14 生成 6 步 PWM

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 126 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 126. COM 事件生成 6 步 PWM 的示例 (OSSR=1)



17.3.15 单脉冲模式

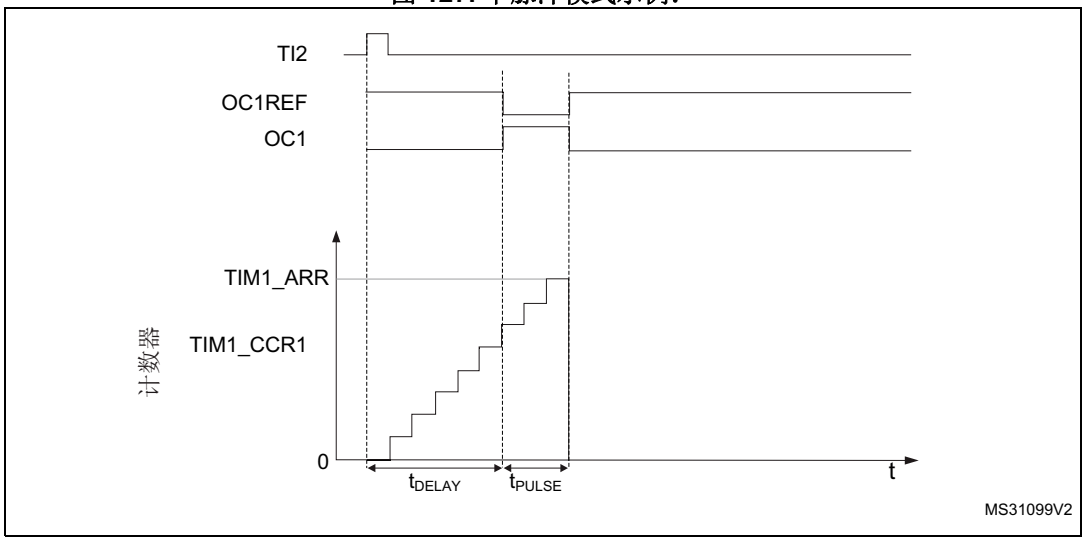
单脉冲模式 (OPM) 是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。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 127. 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。

17.3.16 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 两个输入用于连接增量编码器。请参见 *表 101*。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，计数器会根据增量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。*图 101* 汇总了可能的组合（假设 TI1 和 TI2 不同时切换）。

表 101. 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 128 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，TI1FP2 映射到 TI2 上）。
- CC1P=“0”，CC1NP=“0”，且 IC1F =“0000”（TIMx_CCER 寄存器，TI1FP1 未反相，TI1FP1=TI1）。
- CC2P=“0”，CC2NP=“0”，且 IC2F =“0000”（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 128. 编码器接口模式下的计数器工作示例。

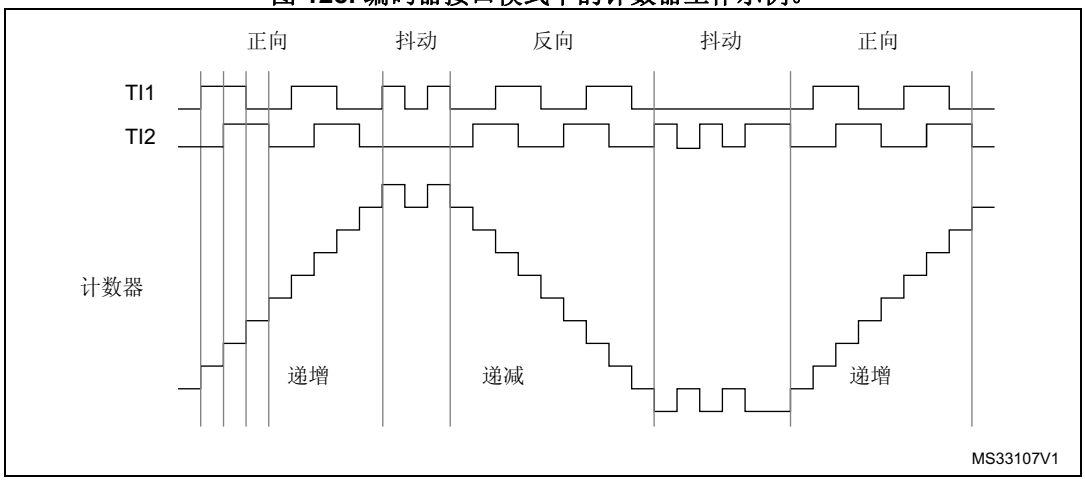
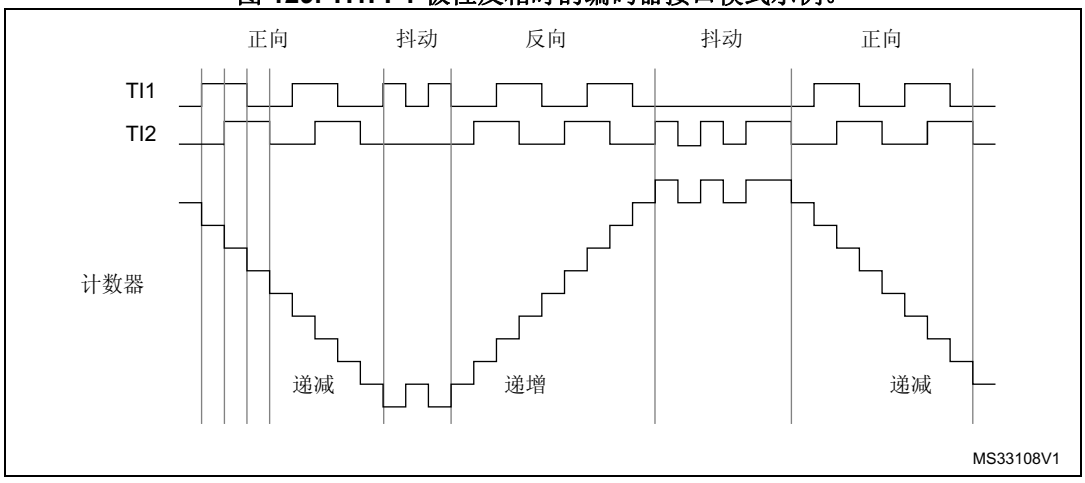


图 129 举例说明 TI1FP1 极性反相时计数器的行为（除 CC1P=“1” 外，其他配置与上例相同）。

图 129. 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。

17.3.17 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 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。下面的*第 17.3.18 节*以连接霍尔传感器为例介绍了此功能。

17.3.18 连接霍尔传感器

可通过用于生成电机驱动 PWM 信号的高级控制定时器（TIM1 或 TIM8）以及*图 130* 中称为“接口定时器”的另一个定时器 TIMx（TIM2、TIM3、TIM4 或TIM5），实现与霍尔传感器的连接。3 个定时器输入引脚（TIMx_CH1、TIMx_CH2 和 TIMx_CH3）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*图 113*）。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）。

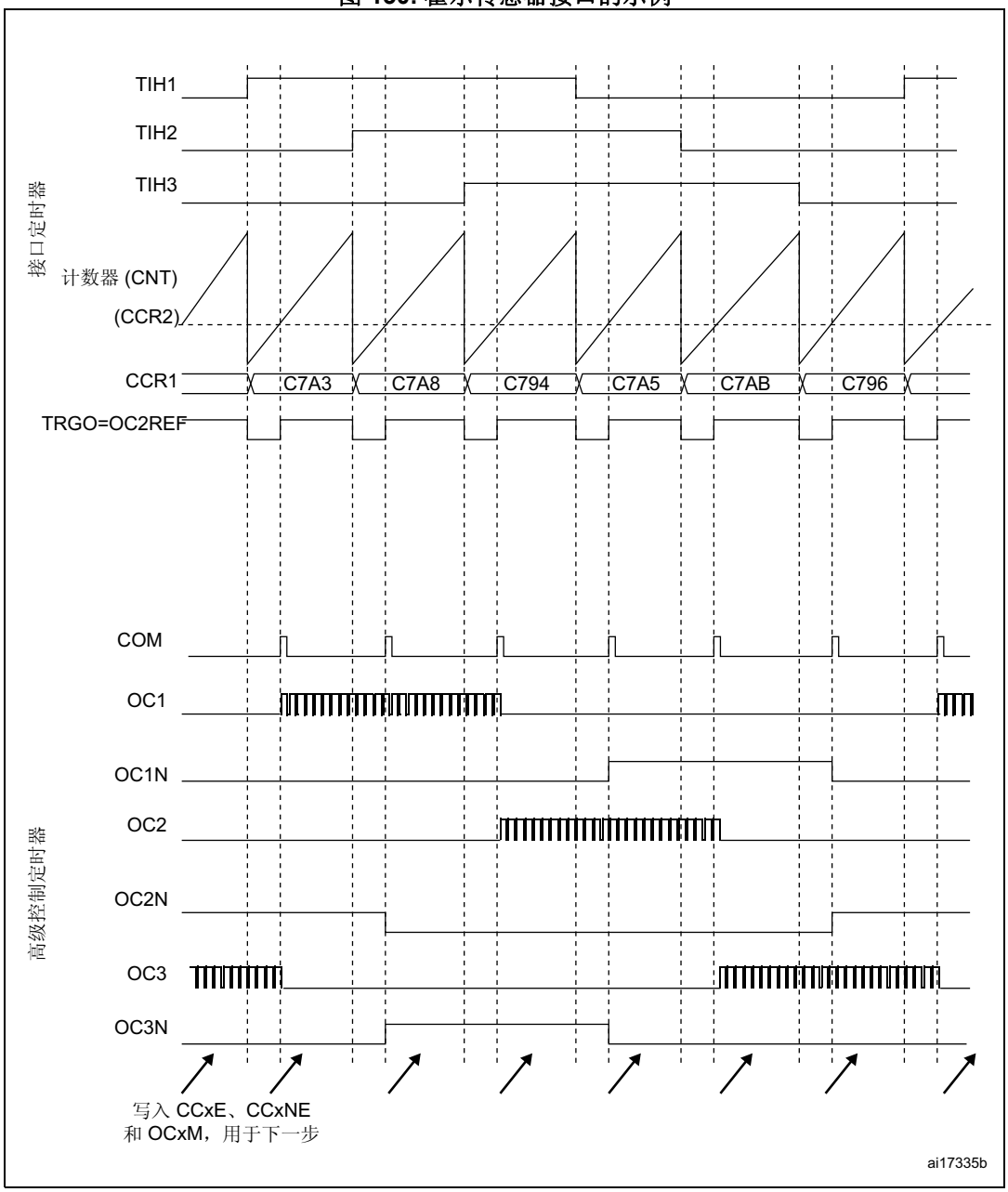
示例：霍尔输入与一个 TIMx 定时器相连接，每当霍尔输入发生更改，需要在所编程的延迟过后更改高级控制定时器 TIM1 的 PWM 配置。

- 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“11”。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 130 为本示例的示意图。

图 130. 霍尔传感器接口的示例



17.3.19 TIMx 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 位为 0，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。

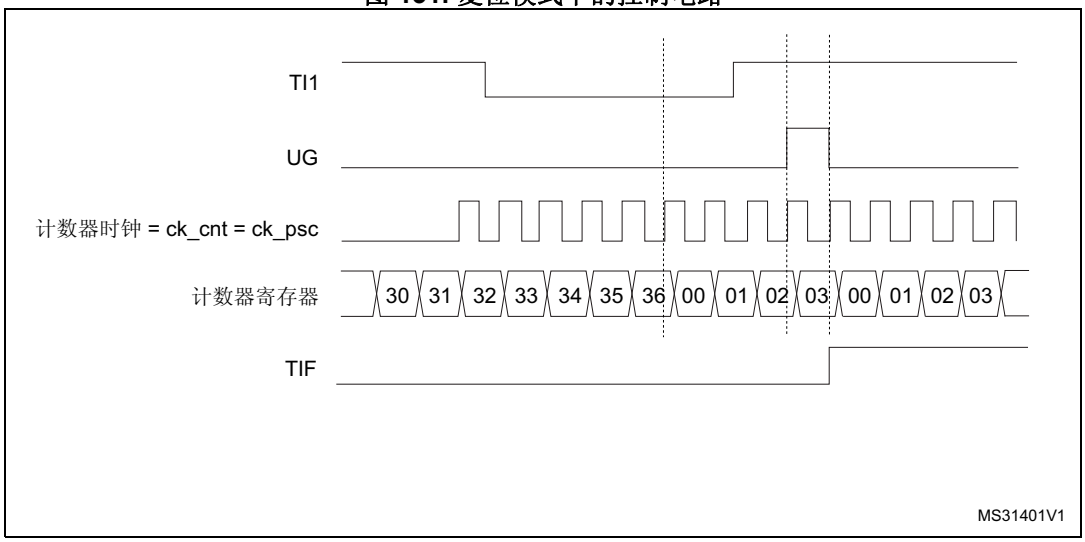
在以下示例中，TI1 输入上出现上升沿时，递增计数器清零：

- 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 131. 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。

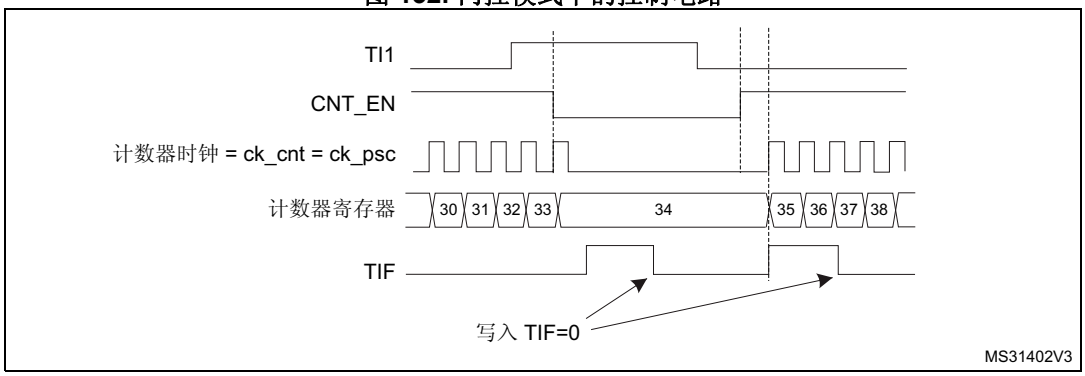
在以下示例中，递增计数器仅在 TI1 输入为低电平时计数：

- 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 132. 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。

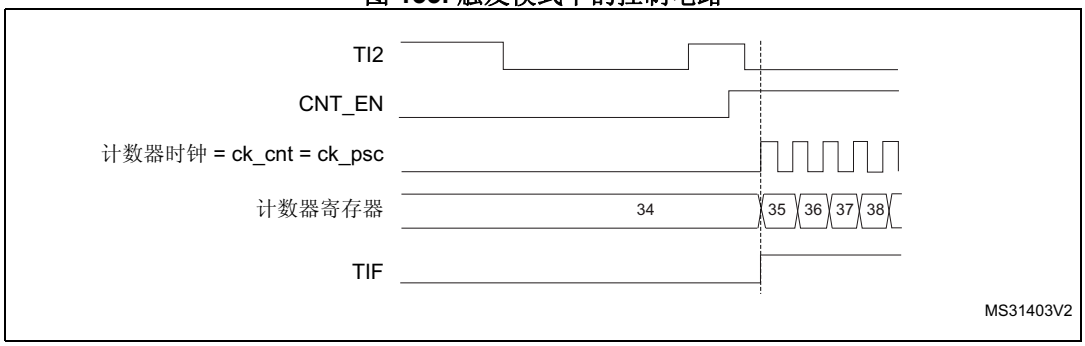
在以下示例中，TI2 输入上出现上升沿时，递增计数器启动：

- 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 133. 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。

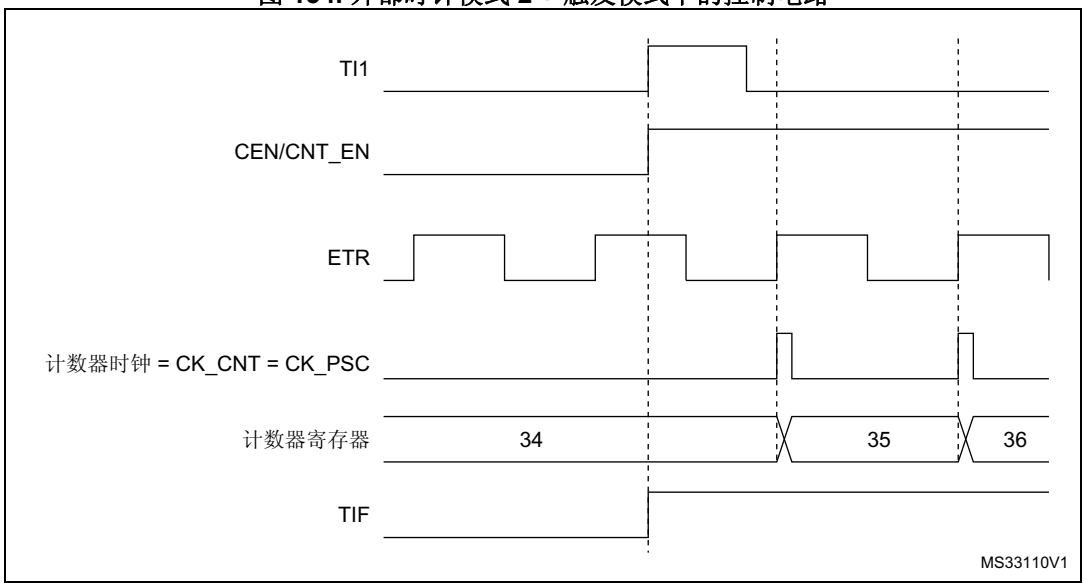
在以下示例中，只要 TI1 出现上升沿，递增计数器即会在 ETR 信号的每个上升沿处递增：

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 134. 外部时钟模式 2 + 触发模式下的控制电路



17.3.20 定时器同步

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 536 页的第 18.3.15 节：定时器同步*。

17.3.21 调试模式

当微控制器进入调试模式（带 FPU 的 Cortex®-M4 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 34.16.2 节：对定时器、看门狗、bxCAN 和 I2C 的调试支持*。

17.4 TIM1 和 TIM8 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.2 节：寄存器相关缩写词列表*。

外设寄存器必须按半字（16 位）或字（32 位）进行写访问。而读访问则可按字节（8 位）、半字（16 位）或字（32 位）进行。

17.4.1 TIM1 和 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.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15:10 保留，必须保持复位值。

位 9:8 **CKD[1:0]**：时钟分频 (Clock division)

此位域指示定时器时钟 (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 CCxS=00）的输出比较中断标志才置 1。
10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志才置 1。
11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志都会置 1。

注： 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。*

17.4.2 TIM1 和 TIM8 控制寄存器 2 (TIMx_CR2)

TIM1&TIM8 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.	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 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))

参见 OIS1 位

位 13 **OIS3N**：输出空闲状态 3（OC3N 输出）(Output Idle state 3 (OC3N output))

参见 OIS1N 位

位 12 **OIS3**：输出空闲状态 3（OC3 输出）(Output Idle state 3 (OC3 output))

参见 OIS1 位

位 11 **OIS2N**：输出空闲状态 2（OC2N 输出）(Output Idle state 2 (OC2N output))

参见 OIS1N 位

位 10 **OIS2**：输出空闲状态 2（OC2 输出）(Output Idle state 2 (OC2 output))

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=0 时，（如果 OC1N 有效，则经过死区时间之后）OC1=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 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[2:0]**：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)

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。

17.4.3 TIM1 和 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

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 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)

此位域可定义 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)

此位域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*表 102：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：编码器模式 1——计数器根据 TI1FP1 电平在 TI2FP2 边沿递增/递减计数。

010：编码器模式 2——计数器根据 TI2FP2 电平在 TI1FP1 边沿递增/递减计数。

011：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。

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被控制。

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——由所选触发信号 (TRGI) 的上升沿提供计数器时钟。

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。

表 102. 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 或 LPTIM1(1)	TIM4
TIM8	TIM1	TIM2	TIM4	TIM5

1. 通过 LPTIM1_OR 寄存器位 2 选择 TIM3 或 LPTIM1。默认情况下选择 TIM3。

17.4.4 TIM1 和 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

17.4.5 TIM1 和 TIM8 状态寄存器 (TIMx_SR)

TIM1&TIM8 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.	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

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 保留，必须保持复位值。

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。

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 17.4.3 节：TIM1 和 TIM8 从模式控制寄存器 (TIMx_SMCR)*）。

17.4.6 TIM1 和 TIM8 事件生成寄存器 (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.	Res.	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
								w	w	w	w	w	w	w	w

位 15:8 保留，必须保持复位值。

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)。

17.4.7 TIM1 和 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

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。此寄存器的所有其他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
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 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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道配置为输出

01：CC2 通道配置为输入，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)

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。

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： ***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 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。*

3：*此位域将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成 COM 事件时，OC1M 有效位才会从预装载位获取新值。*

位 3 **OC1PE**：输出比较 1 预装载使能 (Output Compare 1 preload enable)

0：禁止与 TIMx_CCR1 相关的预装载寄存器。可随时向 TIMx_CCR1 写入数据，写入后将立即使用新值。

1：使能与 TIMx_CCR1 相关的预装载寄存器。可读/写访问预装载寄存器。TIMx_CCR1 预装载值在每次生成更新事件时都会装载到有效寄存器中。

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）。其他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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出

01：CC1 通道配置为输入，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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC2 通道配置为输出
01：CC2 通道配置为输入，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)

此位域可定义 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)

此位域定义 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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出

01：CC1 通道配置为输入，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。

17.4.8 TIM1 和 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

请参见上述 CCMR1 寄存器说明。

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 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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC4 通道配置为输出

01：CC4 通道配置为输入，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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC3 通道配置为输出
01：CC3 通道配置为输入，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 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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC4 通道配置为输出
01：CC4 通道配置为输入，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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC3 通道配置为输出
01：CC3 通道配置为输入，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。

17.4.9 TIM1 和 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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15:14 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： 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。
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。

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/边沿触发

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：　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。

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。

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 103. 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	0	0	0	禁止输出（不由定时器驱动） OCx=0、OCx_EN=0	禁止输出（不由定时器驱动） OCxN=0、OCxN_EN=0
				1	禁止输出（不由定时器驱动） OCx=0、OCx_EN=0	OCxREF + 极性 OCxN=OCxREF 异或 CCxNP、 OCxN_EN=1
			1	0	OCxREF + 极性 OCx=OCxREF 异或 CCxP、 OCx_EN=1	禁止输出（不由定时器驱动） OCxN=0、OCxN_EN=0
				1	OCREF + 极性 + 死区 OCx_EN=1	OCREF 互补项（对 OCREF 进行“非”运算） + 极性 + 死区 OCxN_EN=1
		1	0	0	禁止输出（不由定时器驱动） OCx=CCxP、OCx_EN=0	禁止输出（不由定时器驱动） OCxN=CCxNP、OCxN_EN=0
				1	关闭状态（输出使能为无效状态） OCx=CCxP、OCx_EN=1	OCxREF + 极性 OCxN=OCxREF 异或 CCxNP、 OCxN_EN=1
			1	0	OCxREF + 极性 OCx=OCxREF 异或 CCxP、 OCx_EN=1	关闭状态（输出使能为无效状态） OCxN=CCxNP、OCxN_EN=1
				1	OCREF + 极性 + 死区 OCx_EN=1	OCREF 互补项（对 OCREF 进行“非”运算）+ 极性 + 死区 OCxN_EN=1
0	0	X	0	0	禁止输出（不由定时器驱动） OCx=CCxP、OCx_EN=0	禁止输出（不由定时器驱动） OCxN=CCxNP、OCxN_EN=0
				1	禁止输出（不由定时器驱动） 异步：OCx=CCxP、OCx_EN=0、OCxN=CCxNP、OCxN_EN=0 那么如果时钟存在：在死区后 OCx=OISx 且 OCxN=OISxN，从而假定 OISx 和 OISxN 在有效状态下与 OCX 和 OCxN 不对应。	
			1	0		
				1		
	1		0	0	禁止输出（不由定时器驱动） OCx=CCxP、OCx_EN=0	禁止输出（不由定时器驱动） OCxN=CCxNP、OCxN_EN=0
				1	关闭状态（输出使能为无效状态） 异步：OCx=CCxP、OCx_EN=1、OCxN=CCxNP、OCxN_EN=1 那么如果时钟存在：在死区后 Ocx = OISx 且 OCxN = OISxN，从而假定 OISx 和 OISxN 在有效状态下与 OCX 和 OCxN 不对应	
			1	0		
				1		

1. 如果一个通道的两个输出均未使用 (CCxE = CCxNE = 0)，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。*

17.4.10 TIM1 和 TIM8 计数器 (TIMx_CNT)

TIM1&TIM8 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)

17.4.11 TIM1 和 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）时要装载到有效预分频器寄存器的值。

17.4.12 TIM1 和 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]**：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 17.3.1 节：时基单元*。
当自动重载值为空时，计数器不工作。

17.4.13 TIM1 和 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

位 15:8 保留，必须保持复位值。

位 7:0 **REP[7:0]**：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，从预装载寄存器向有效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。

17.4.14 TIM1 和 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 是捕获/比较寄存器 1 的预装载值。

如果没有通过 TIMx_CCMR1 寄存器中的 OC1PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 1）。

有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 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。

17.4.15 TIM1 和 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 是捕获/比较寄存器 2 的预装载值。
如果没有通过 TIMx_CCMR2 寄存器中的 OC2PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 2）。
有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 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。

17.4.16 TIM1 和 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 3 值 (Capture/Compare 3 value)

如果通道 CC3 配置为输出：
CCR3 是捕获/比较寄存器 3 的预装载值。
如果没有通过 TIMx_CCMR3 寄存器中的 OC3PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 3）。
有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 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。

17.4.17 TIM1 和 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 4 值 (Capture/Compare 4 value)

如果通道 CC4 配置为输出：
CCR4 是捕获/比较寄存器 4 的预装载值。
如果没有通过 TIMx_CCMR4 寄存器中的 OC4PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获/比较寄存器 4）。
有效捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 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。

17.4.18 TIM1 和 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

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 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 15 **MOE**：主输出使能 (Main output enable)

只要断路输入变为有效状态，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：OC 和 OCN 输出禁止或被强制为空闲状态。
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 493 页的第 17.4.9 节：TIM1 和 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)

0：禁止断路输入（BRK 和 CSS 时钟故障事件）

1：使能断路输入（BRK 和 CSS 时钟故障事件）

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 493 页的第 17.4.9 节：TIM1 和 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（OC/OCN 使能输出信号 = 0）。

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。然后设置 OC/OCN 使能输出信号 = 1

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)

此位在 MOE=0 时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 493 页的第 17.4.9 节：TIM1 和 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（OC/OCN 使能输出信号 = 0）。

1：处于无效状态时，一旦 CCxE = 1 或 CCxNE = 1，便将 OC/OCN 输出首先强制为其空闲电平。然后设置 OC/OCN 使能输出信号 = 1

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)

此位域定义插入到互补输出之间的死区持续时间。DT 与该持续时间相对应。

DTG[7:5]=0xx => DT=DTG[7:0]x t_{dtg}，其中 $t_{dtg}=t_{DTS}$。

DTG[7:5]=10x => DT=(64+DTG[5:0])xt_{dtg}，其中 $T_{dtg}=2xt_{DTS}$。

DTG[7:5]=110 => DT=(32+DTG[4:0])xt_{dtg}，其中 $T_{dtg}=8xt_{DTS}$。

DTG[7:5]=111 => DT=(32+DTG[4:0])xt_{dtg}，其中 $T_{dtg}=16xt_{DTS}$。

示例：如果 T_{DTS}=125ns (8MHz)，则可能的死区值为：

0 到 15875 ns（步长为 125 ns），

16 us 到 31750 ns（步长为 250 ns），

32 us 到 63us（步长为 1 us），

64 us 到 126 us（步长为 2 us）

注： 只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，此位域即无法修改。

17.4.19 TIM1 和 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 位定义了 DMA 在连续模式下的传送长度（当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送）。

TIMx_DMAR 地址）

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。

17.4.20 TIM1 和 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。

DMA 连续传送功能使用方法示例

本例中，定时器 DMA 连续传送功能用于将 CCRx 寄存器（x = 2、3、4）的内容更新为通过 DMA 传输到 CCRx 寄存器中的多个半字。

具体操作步骤如下：

1. 将相应的 DMA 通道配置如下：
   - DMA 通道外设地址为 DMAR 寄存器地址。
   - DMA 通道存储器地址为包含要通过 DMA 传输到 CCRx 寄存器的数据的 RAM 缓冲区地址。
   - 要传输的数据量 = 3（参见下文注释）。
   - 禁止循环模式。
2. 通过将 DBA 和 DBL 位域配置如下来配置 DCR 寄存器：
   DBL = 3 次传输，DBA = 0xE。
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

注： *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。*

17.4.21　TIM1 和 TIM8 寄存器映射

TIM1 和 TIM8 寄存器可映射为 16 位可寻址寄存器，如下表所述：

表 104. TIM1 和 TIM8 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**TIMx_CR1**		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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.	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
0x08	**TIMx_SMCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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
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	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	**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.	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset value																				0	0	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.	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																									0	0	0	0	0	0	0	0
0x18	**TIMx_CCMR1** Output Compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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
	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
0x1C	**TIMx_CCMR2** Output Compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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
	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
0x20	**TIMx_CCER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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
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
0x30	**TIMx_RCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
	Reset value																									0	0	0	0	0	0	0	0

表 104. TIM1 和 TIM8 寄存器映射和复位值 （续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x34	**TIMx_CCR1**																	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	**TIMx_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	**TIMx_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	**TIMx_BDTR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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
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

有关寄存器边界地址的信息，请参见*第 54 页的第 2.2.2 节*。

18 通用定时器（TIM2 到 TIM5）

18.1 TIM2 到 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。

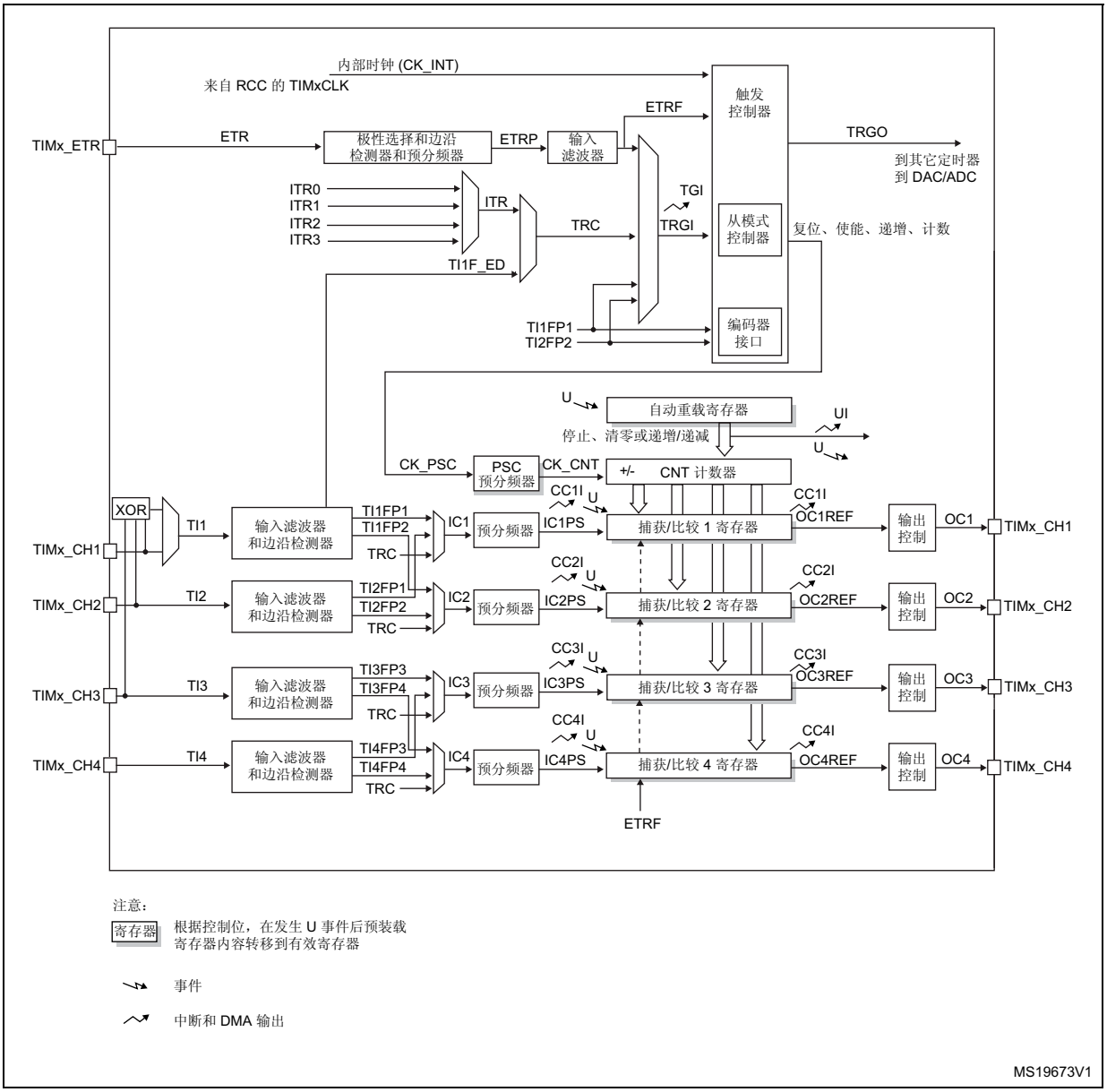
这些定时器彼此完全独立，不共享任何资源。如*第 18.3.15 节*中所述，它们可以同步操作。

18.2 TIM2 到 TIM5 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（即运行时修改），分频系数介于 1 到 65536 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
- 外部时钟触发输入或逐周期电流管理

图 135. 通用定时器框图



18.3 TIM2 到 TIM5 功能说明

18.3.1 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。此计数器可采用递增方式计数。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（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 有效后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 136 和 *图 137* 以一些示例说明在预分频比实时变化时计数器的行为：

图 136. 预分频器分频由 1 变为 2 时的计数器时序图

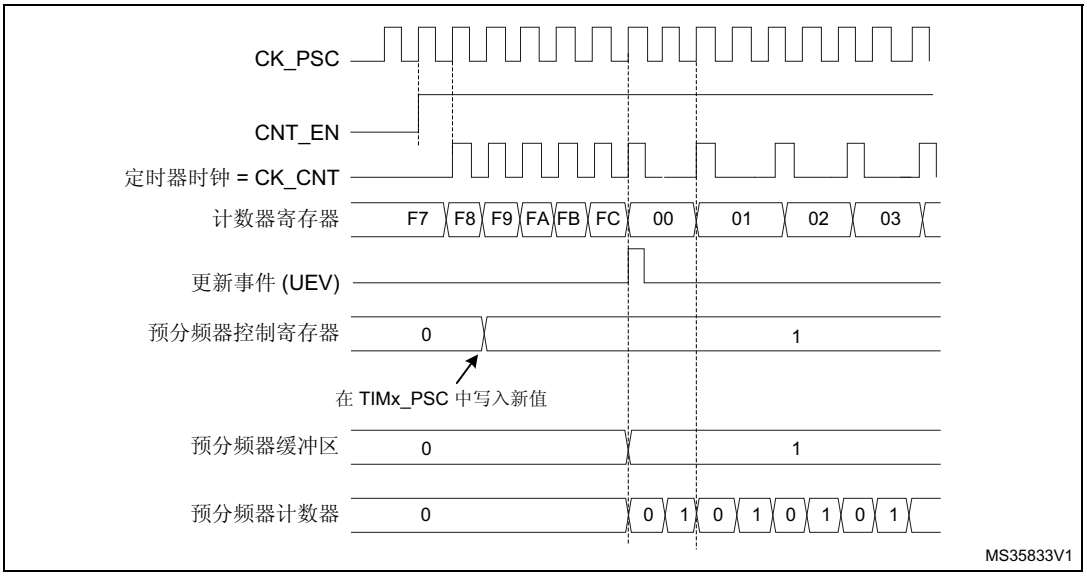
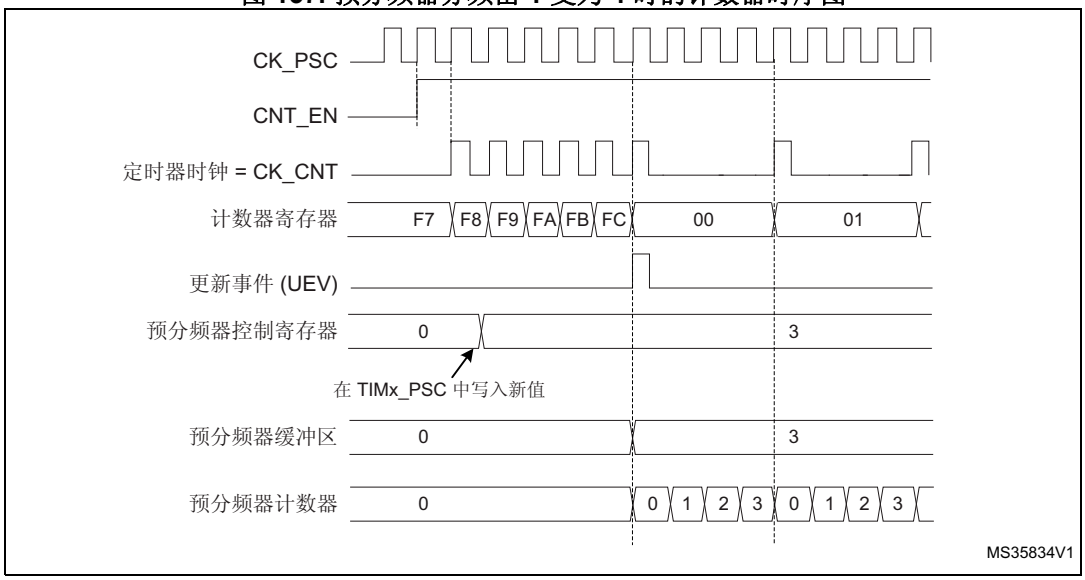


图 137. 预分频器分频由 1 变为 4 时的计数器时序图



18.3.2 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 138. 计数器时序图，1 分频内部时钟

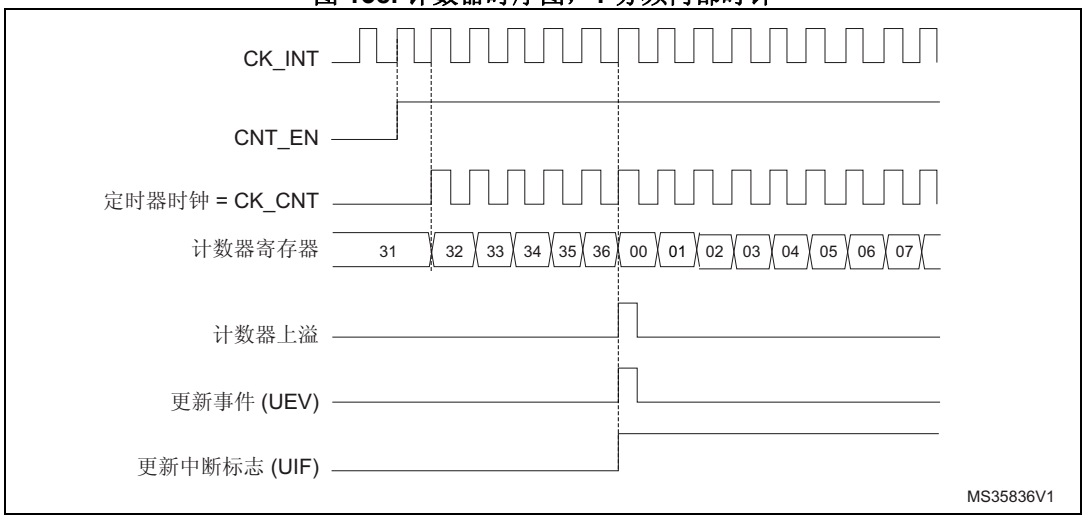


图 139. 计数器时序图，2 分频内部时钟

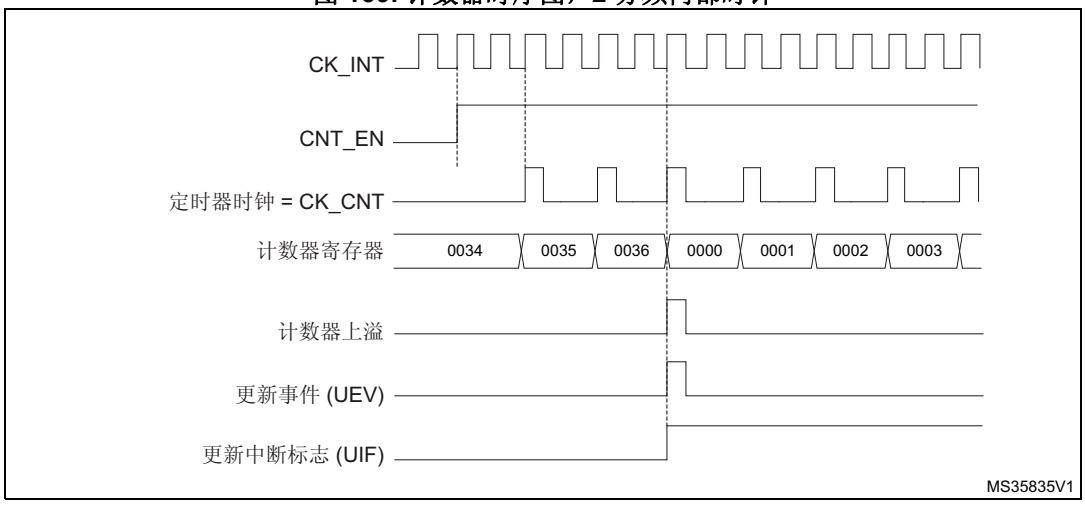


图 140. 计数器时序图，4 分频内部时钟

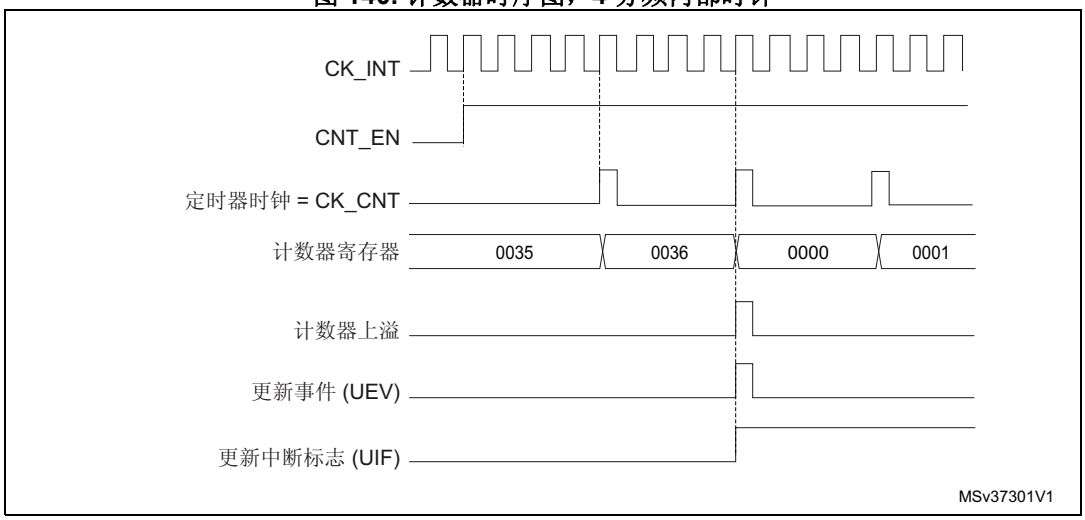


图 141. 计数器时序图，N 分频内部时钟

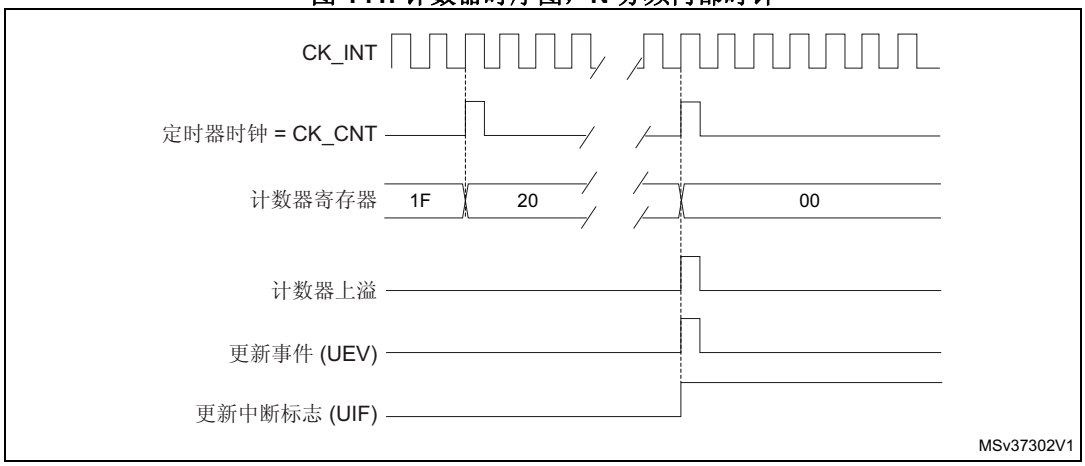


图 142. 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载）

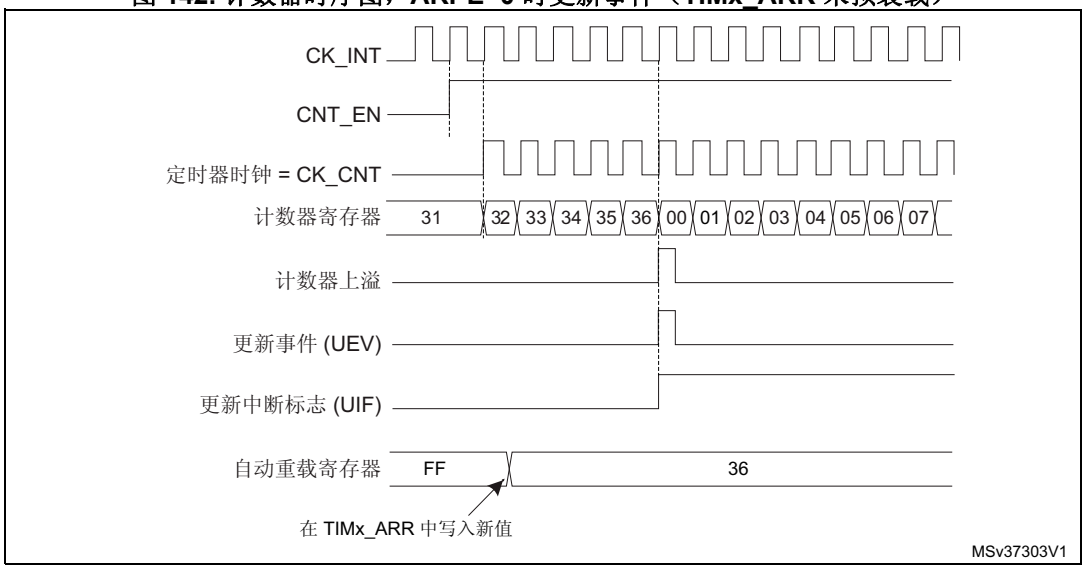
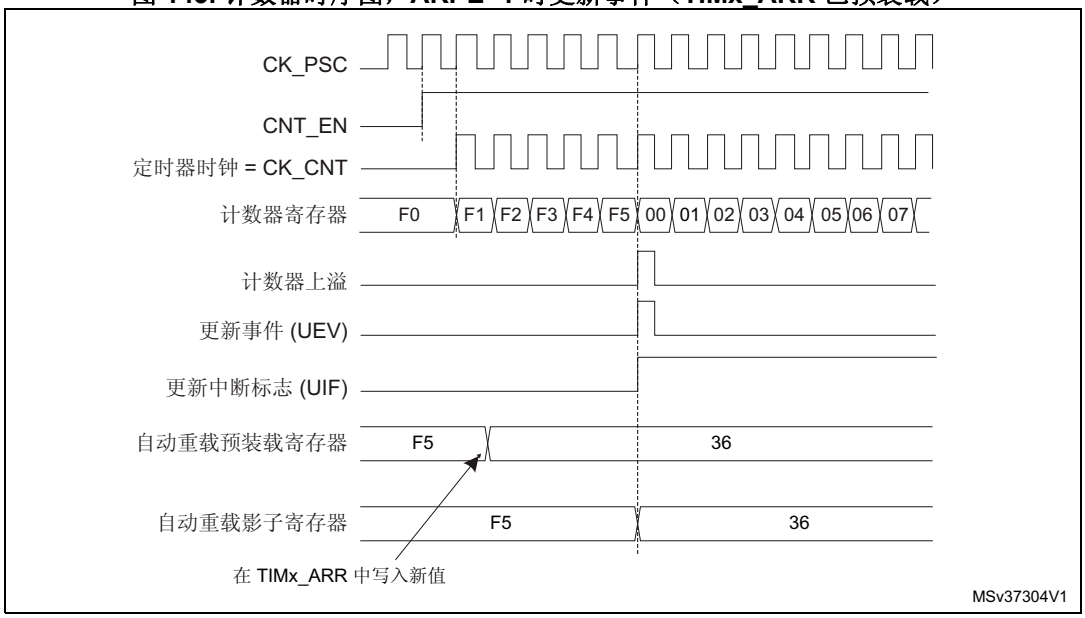


图 143. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 已预装载）



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 寄存器更新在计数器重载之前被更新，因此下一个周期就是预期的值。

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

图 144. 计数器时序图，1 分频内部时钟

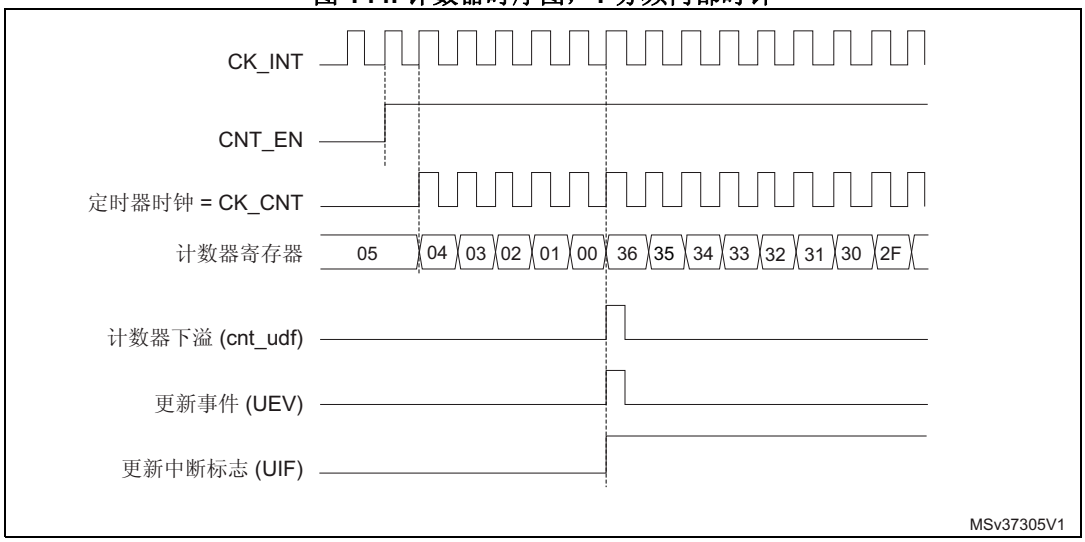


图 145. 计数器时序图，2 分频内部时钟

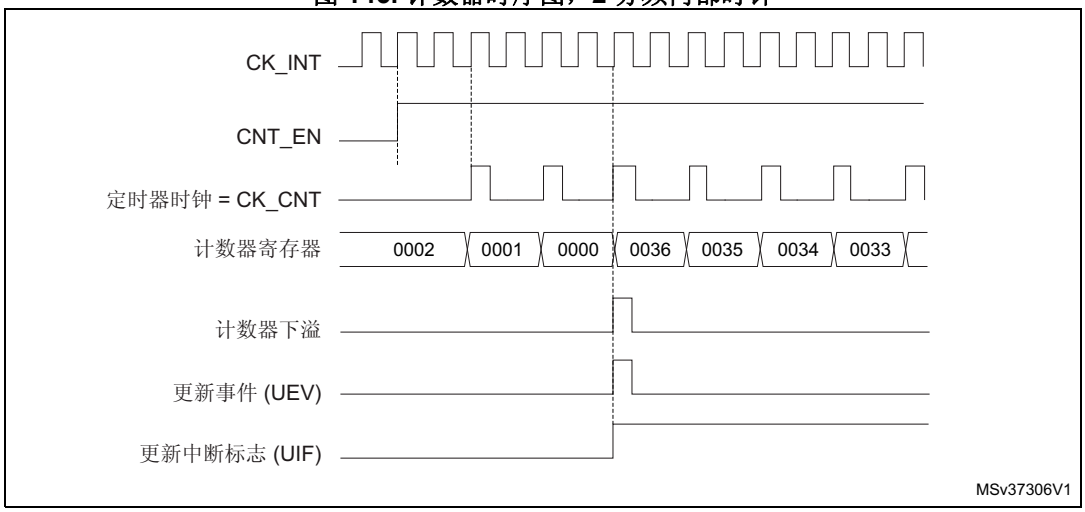


图 146. 计数器时序图，4 分频内部时钟

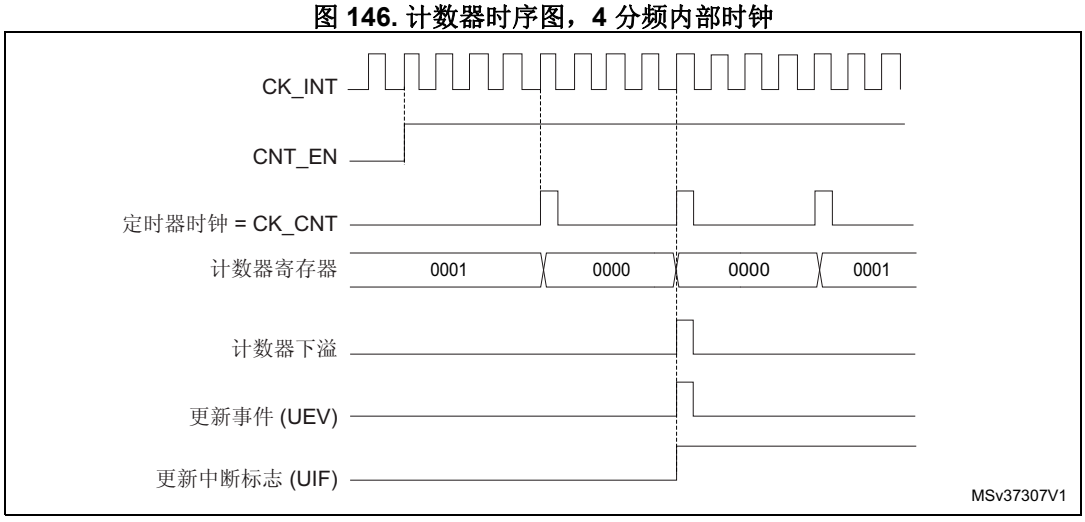


图 147. 计数器时序图，N 分频内部时钟

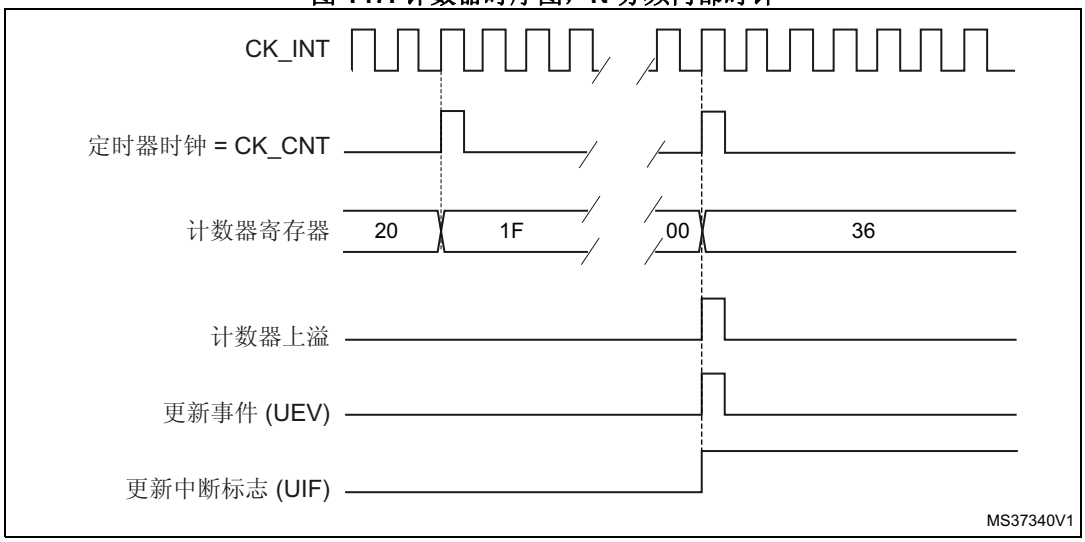
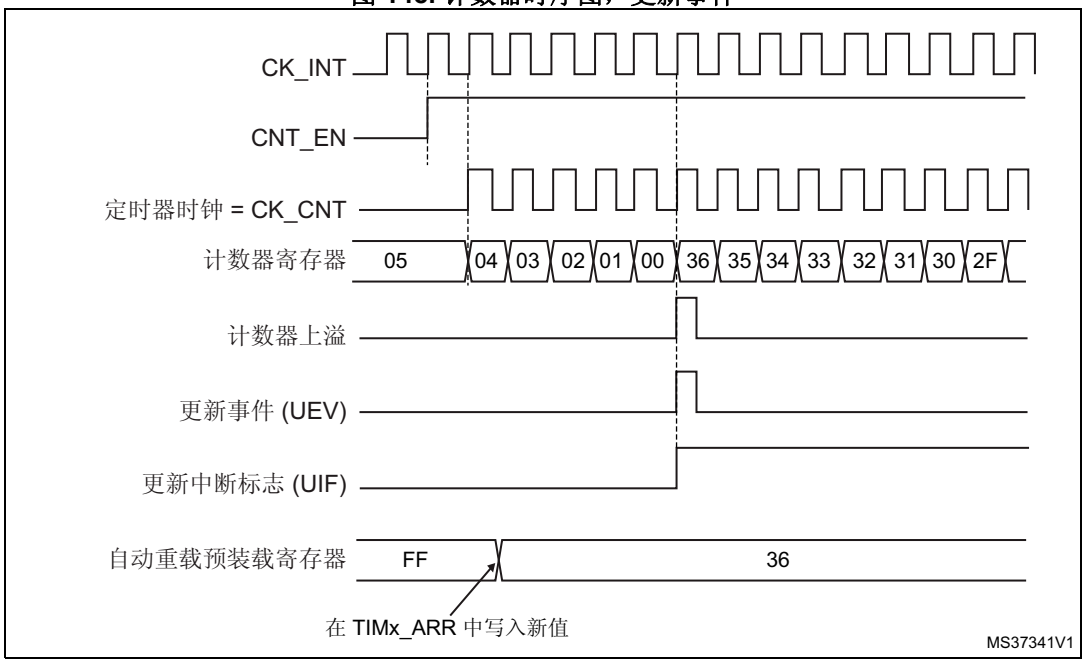


图 148. 计数器时序图，更新事件



中心对齐模式（递增/递减计数）

在中心对齐模式下，计数器从 0 开始计数到自动重载值（TIMx_ARR 寄存器的内容）– 1，生成计数器上溢事件；然后从自动重载值开始向下计数到 1 并生成计数器下溢事件。之后从 0 开始重新计数。

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时，中心对齐模式有效。将通道配置为输出模式时，其输出比较中断标志将在以下模式下置 1，即：计数器递减计数（中心对齐模式 1，CMS =“01”）、计数器递增计数（中心对齐模式 2，CMS =“10”）以及计数器递增/递减计数（中心对齐模式 3，CMS =“11”）。

此模式下无法写入方向位（TIMx_CR1 寄存器中的 DIR 位）。而是由硬件更新并指示当前计数器方向。

每次发生计数器上溢和下溢时都会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。这种情况下，计数器以及预分频器计数器将重新从 0 开始计数。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器仍会根据当前自动重载值进行递增和递减计数。

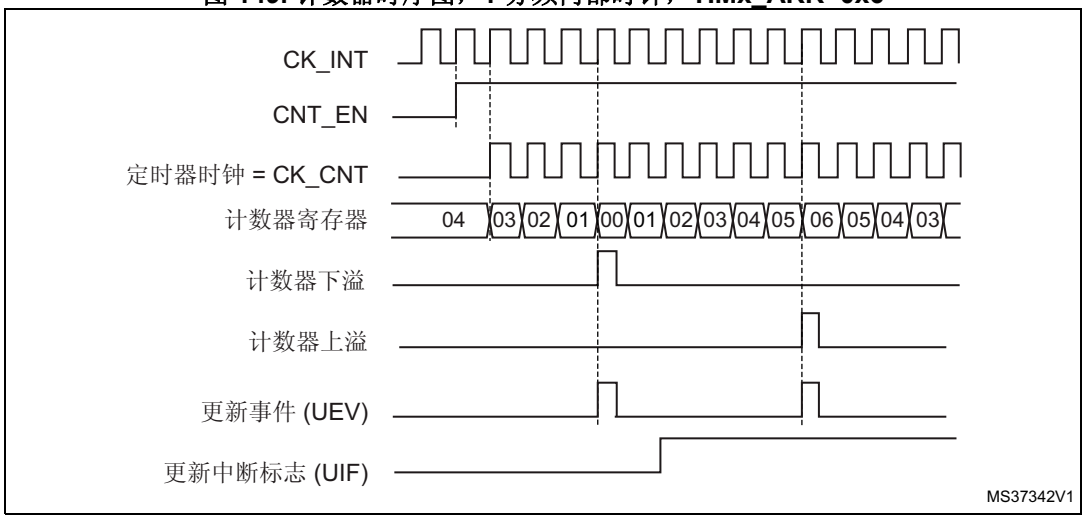
此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断或 DMA 请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

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 149. 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 542 页的第 18.4.1 节：TIMx 控制寄存器 1 (TIMx_CR1)*）。

图 150. 计数器时序图，2 分频内部时钟

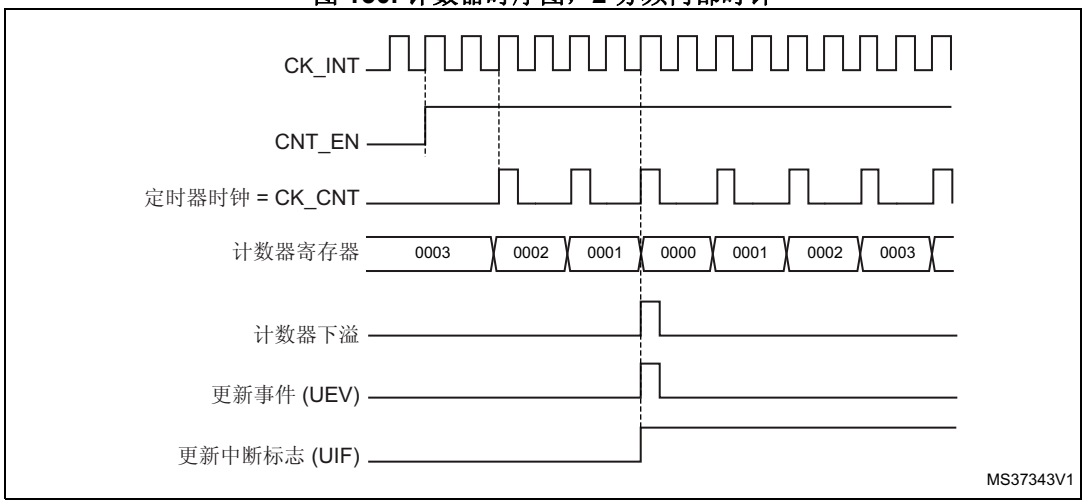
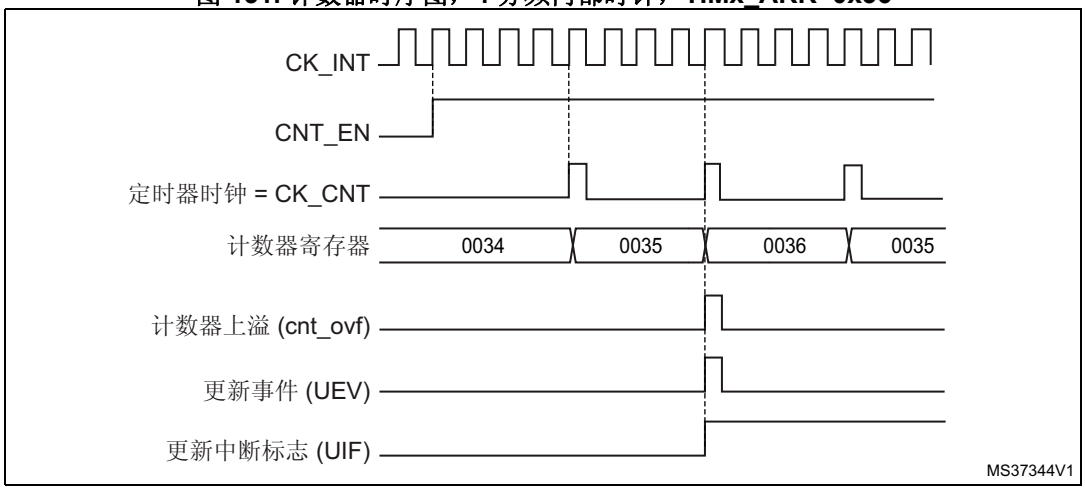


图 151. 计数器时序图，4 分频内部时钟，TIMx_ARR=0x36



1. 中心对齐模式 2 或模式 3 与上溢 UIF 结合使用。

图 152. 计数器时序图，N 分频内部时钟

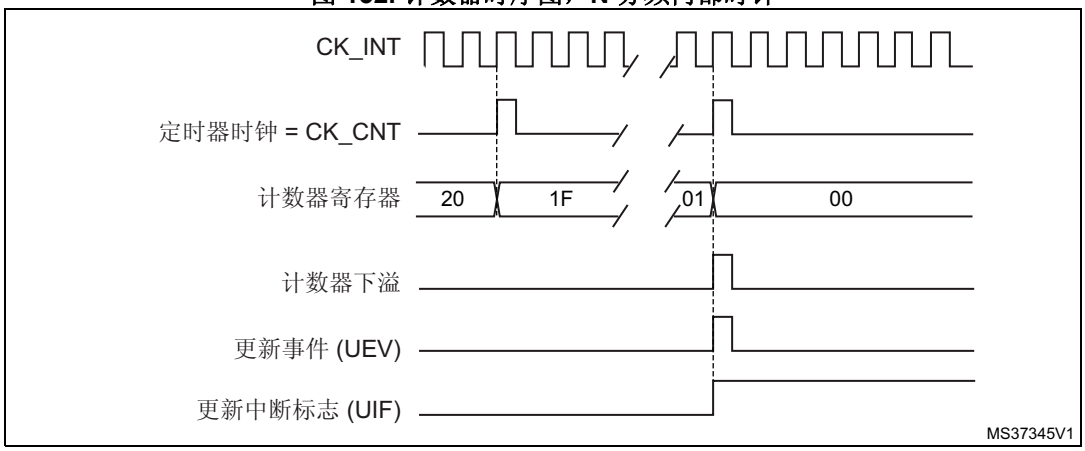


图 153. 计数器时序图，ARPE=1 时更新事件（计数器下溢）

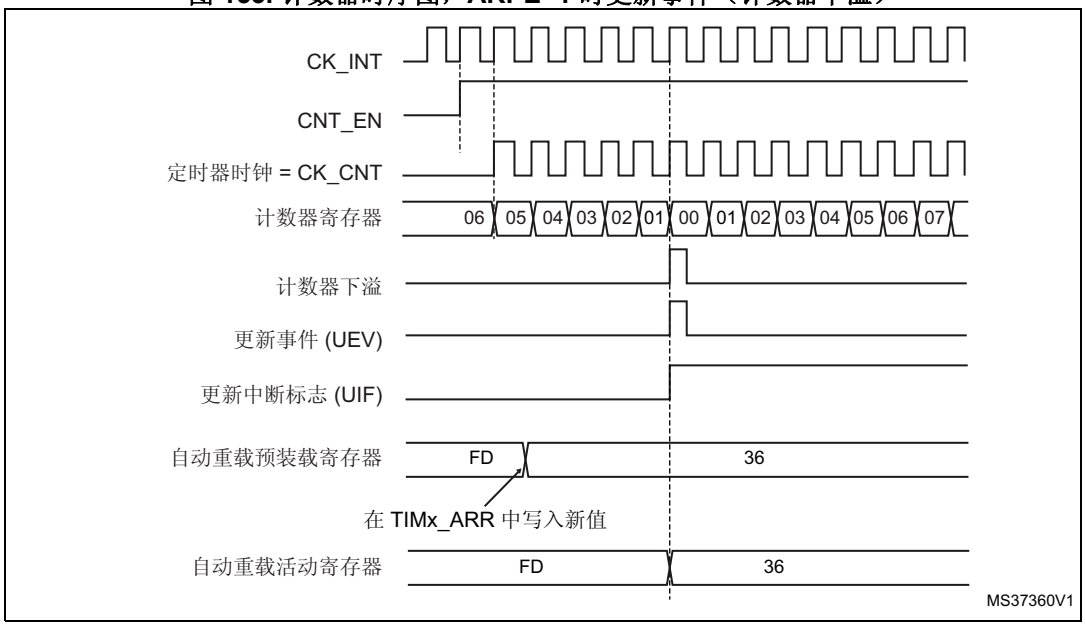
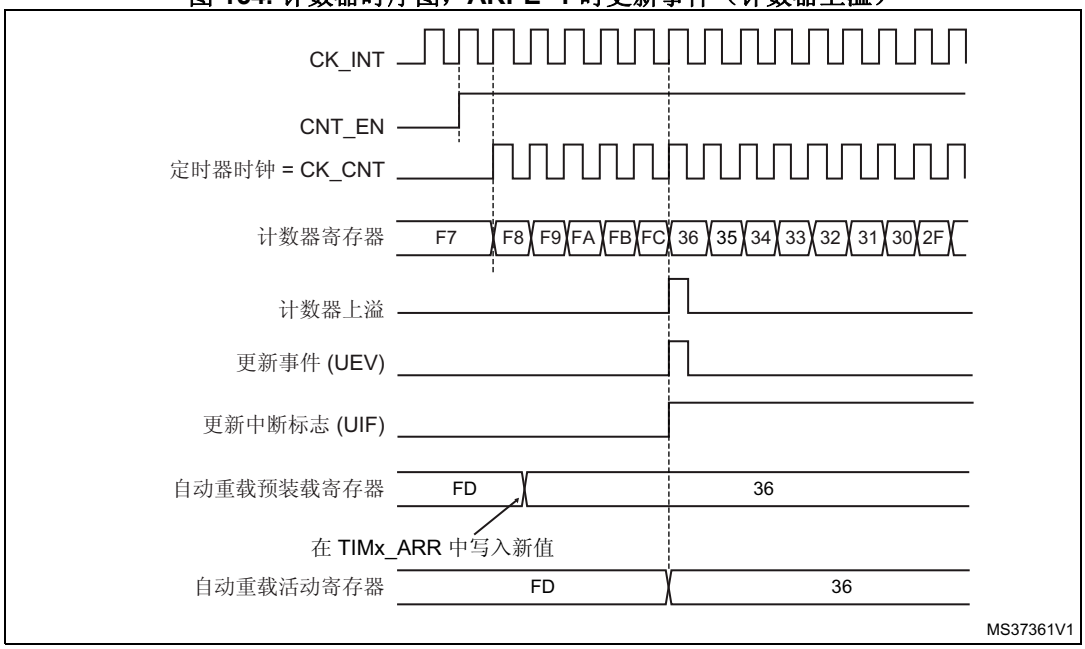


图 154. 计数器时序图，ARPE=1 时更新事件（计数器上溢）



18.3.3 时钟选择

计数器时钟可由下列时钟源提供：

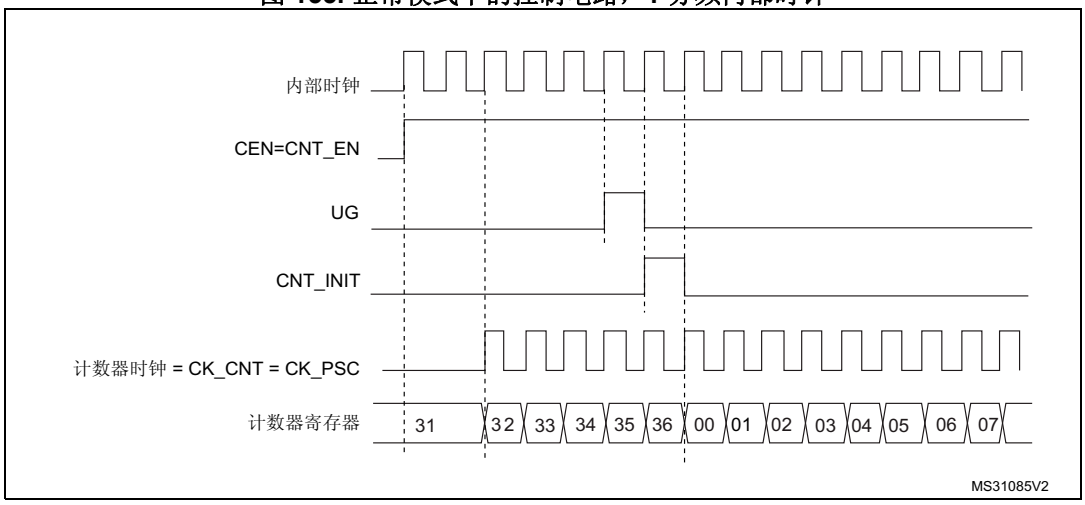
- 内部时钟 (CK_INT)
- 外部时钟模式 1：外部输入引脚 (TIx)
- 外部时钟模式 2：外部触发输入 (ETR)，仅适用于 TIM2、TIM3 和 TIM4。
- 内部触发输入 (ITRx)：使用一个定时器作为另一个定时器的预分频器，例如可以将定时器配置为定时器 2 的预分频器。更多详细信息，请参见*将一个定时器用作另一个定时器的预分频器*。

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 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）。

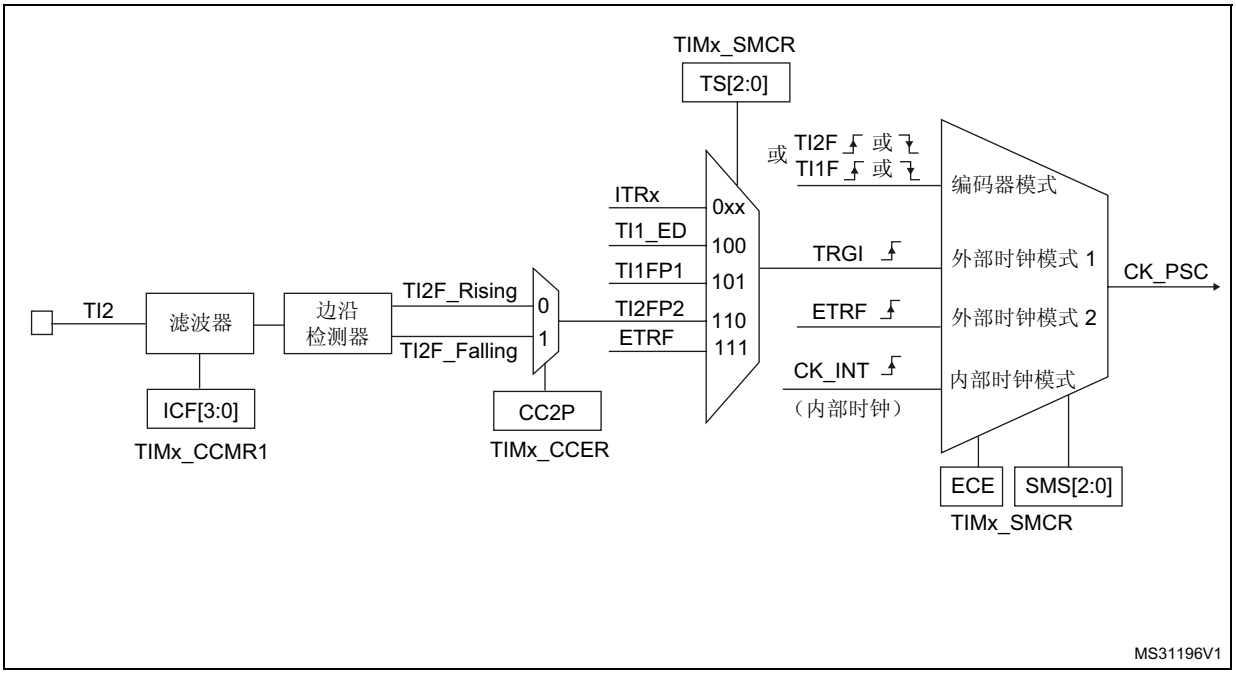
图 155. 正常模式下的控制电路，1 分频内部时钟



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）。

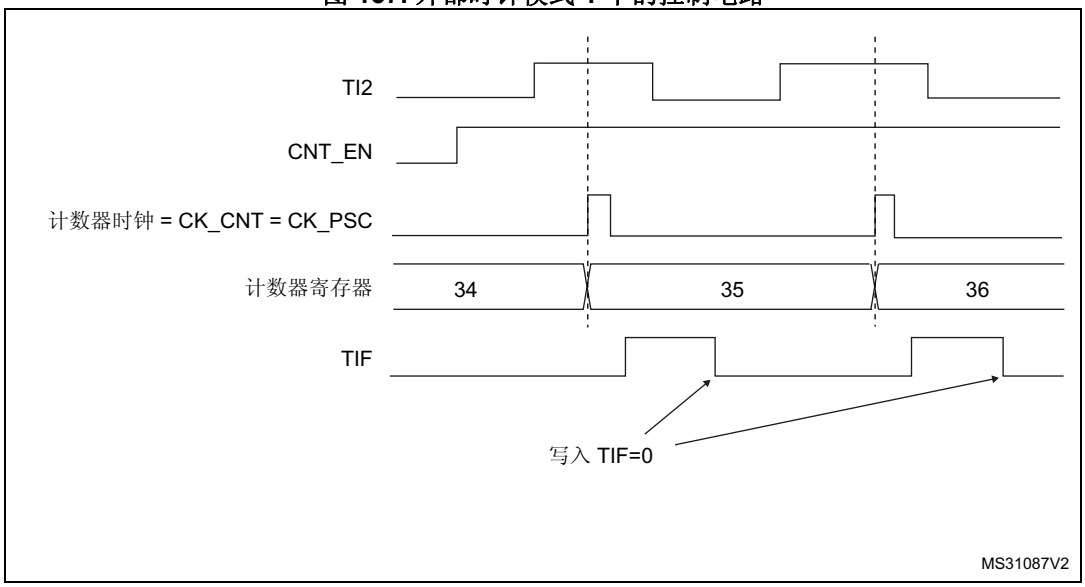
注： *由于捕获预分频器不用于触发操作，因此无需对其进行配置。*

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 157. 外部时钟模式 1 下的控制电路



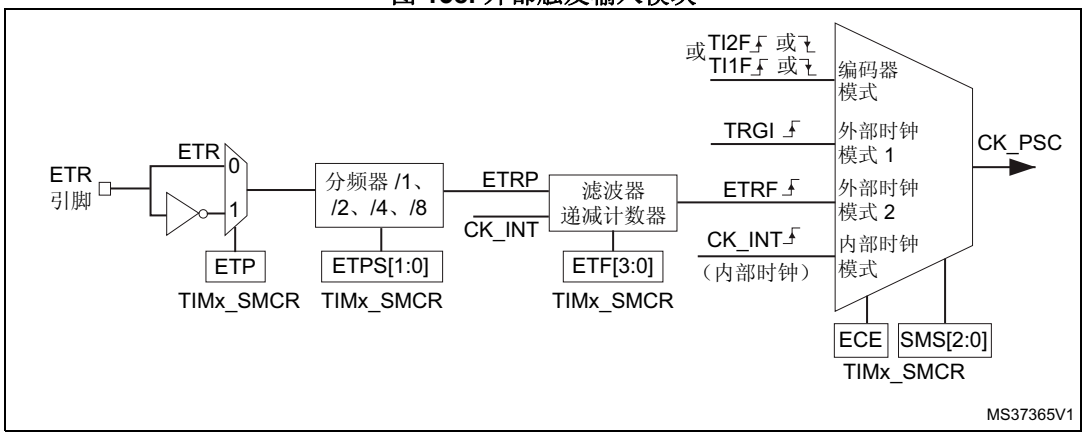
外部时钟源模式 2

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



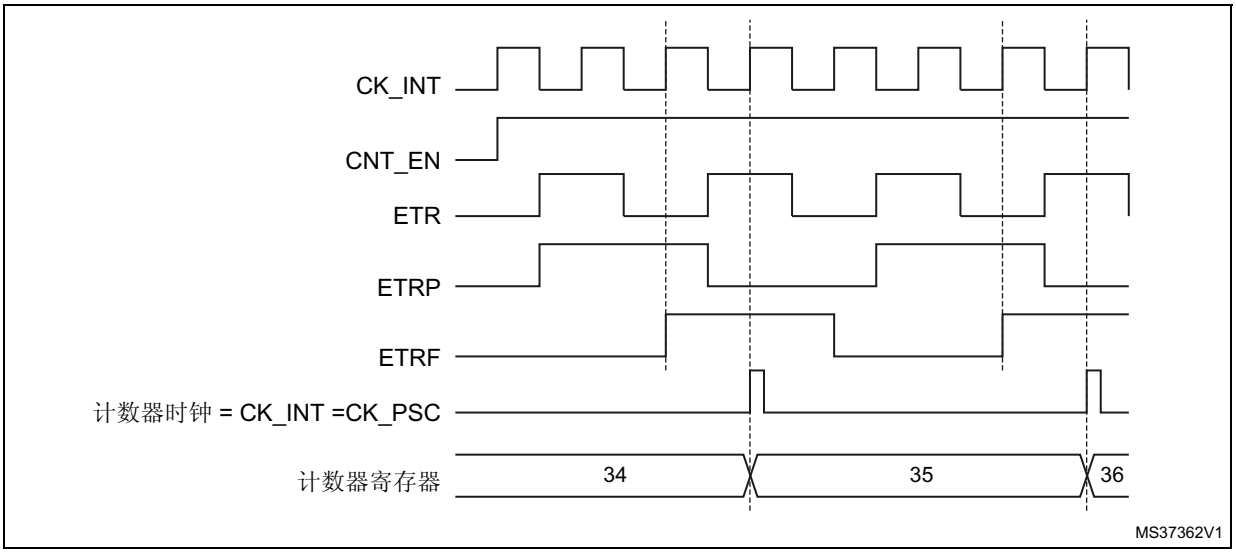
例如，要使递增计数器在 ETR 每出现 2 个上升沿时计数，请执行以下步骤：

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



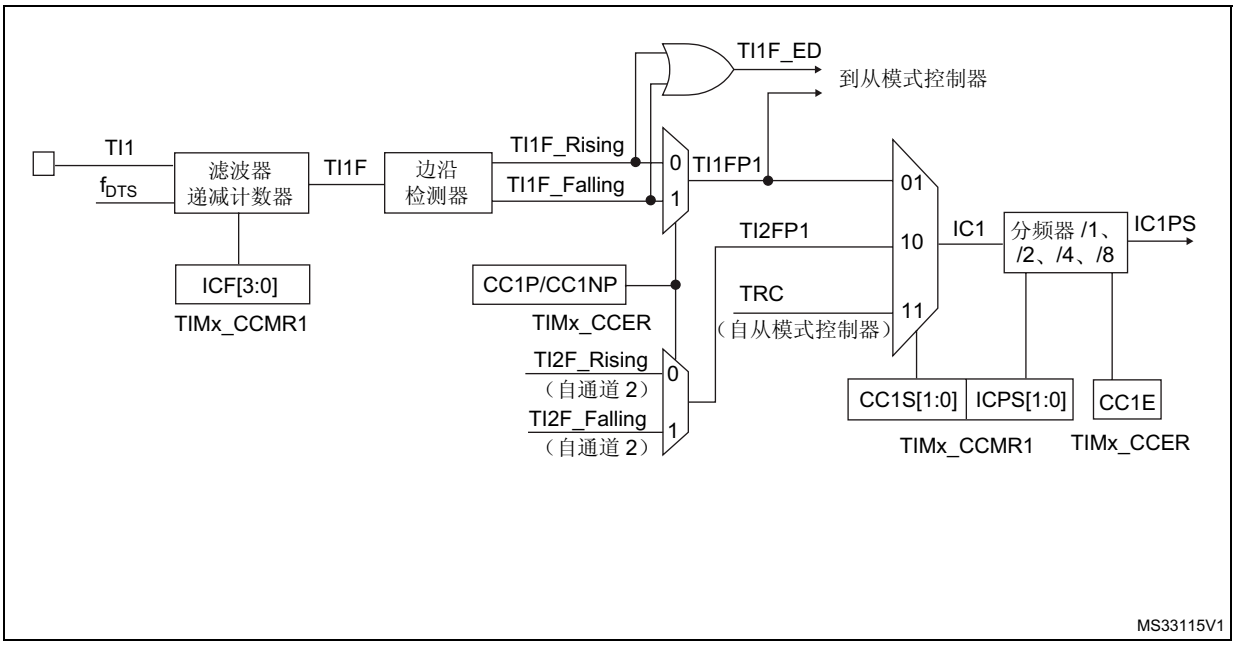
18.3.4 捕获/比较通道

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。

输入级对相应的 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

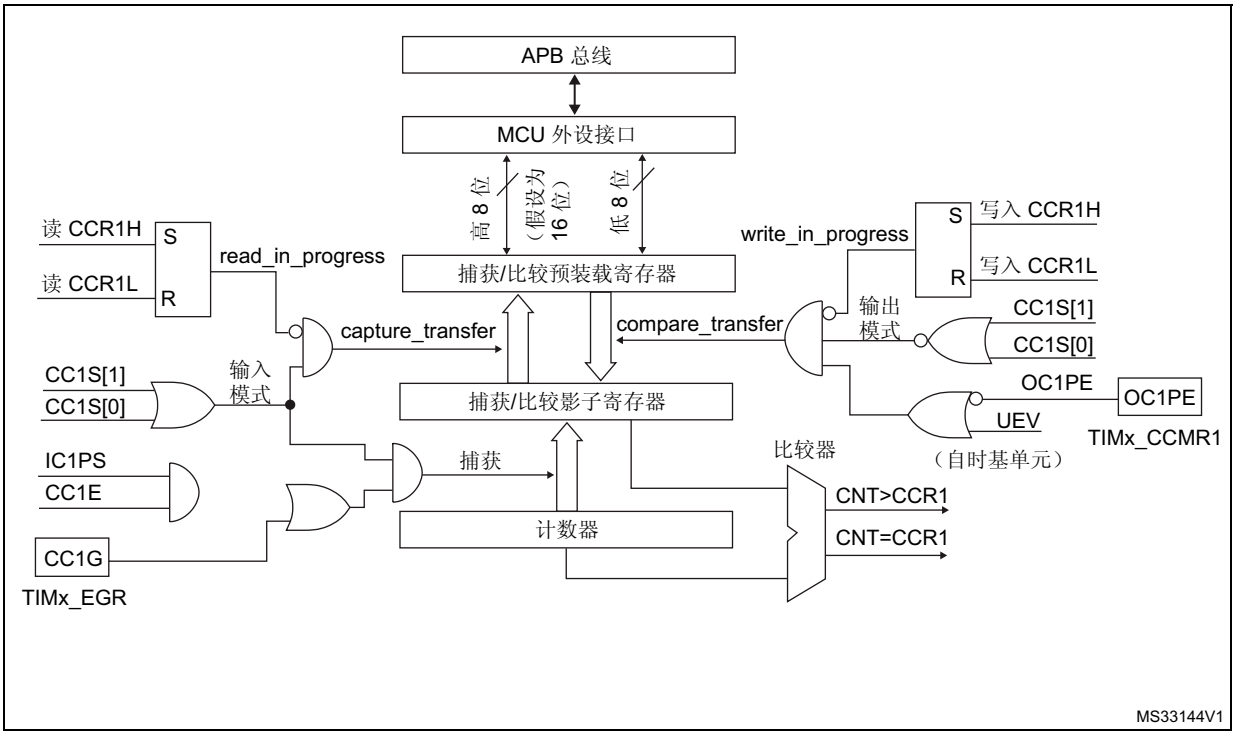
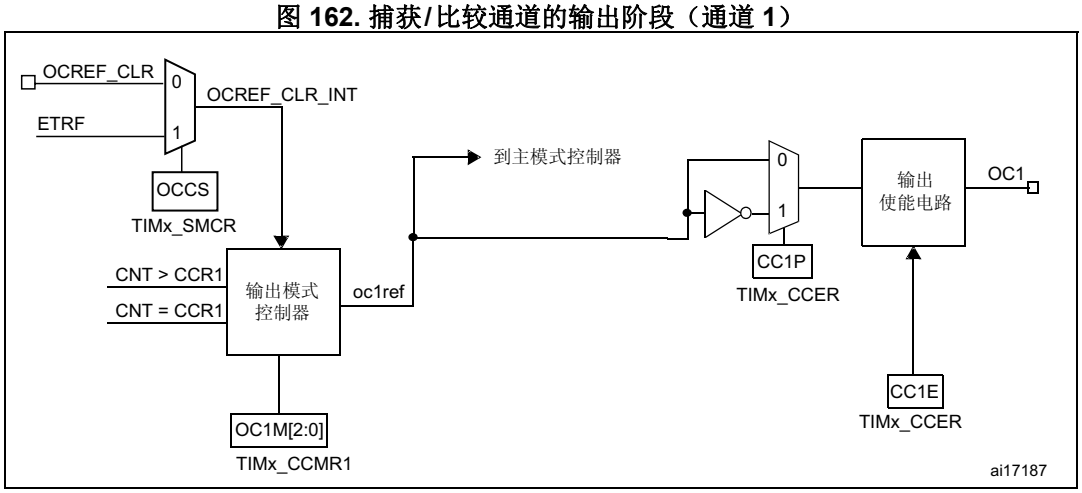


图 162. 捕获/比较通道的输出阶段（通道 1）



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。

18.3.5 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：

- 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 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。
- 通过向 TIMx_CCER 寄存器中的 CC1P 位和 CC1NP 位写入 0，选择 TI1 通道的有效转换边沿（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。*

18.3.6 PWM 输入模式

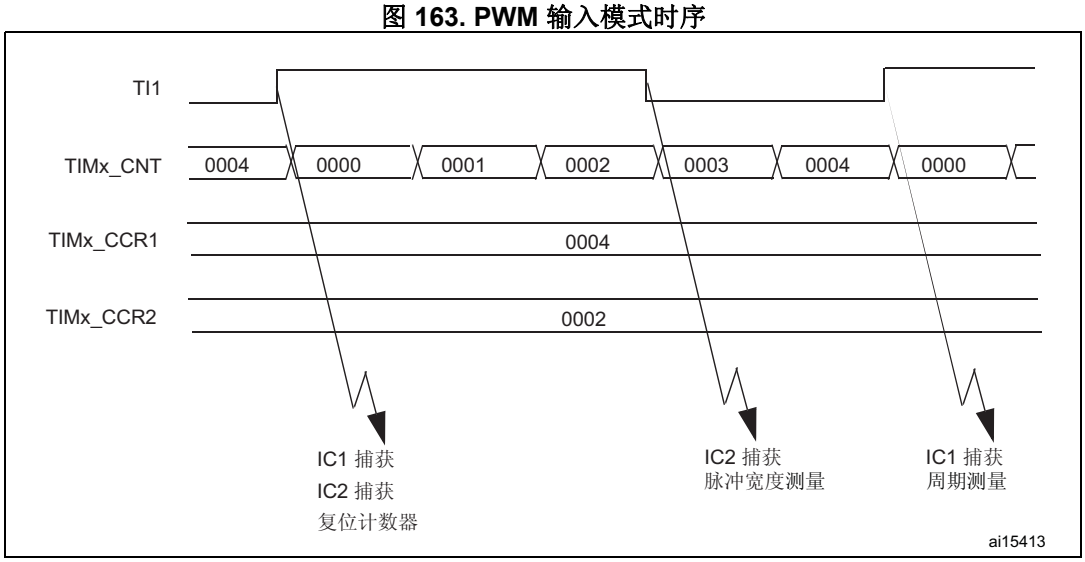
此模式是输入捕获模式的一个特例。其实现步骤与输入捕获模式基本相同，仅存在以下不同之处：

- 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）：

- 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（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）。
- 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 TIMx_CCR2 中的捕获）：向 CC2P 位写入“1”，向 CC2NP 位写入“0”（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 100。
- 使能捕获：向 TIMx_CCER 寄存器中的 CC1E 位和 CC2E 位写入“1”。

图 163. PWM 输入模式时序



18.3.7 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。

18.3.8 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。

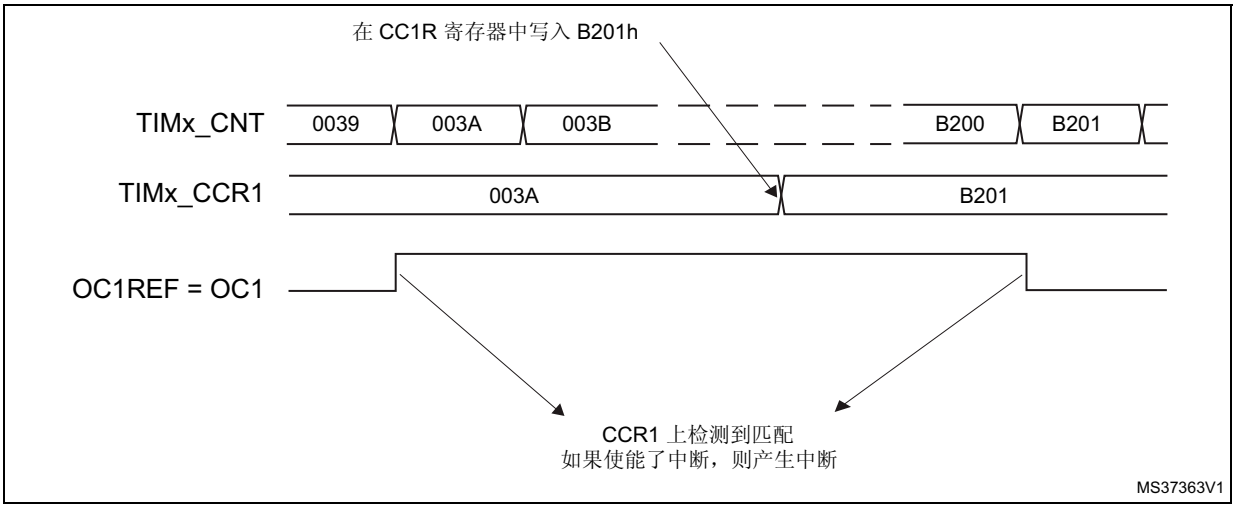
在输出比较模式下，更新事件 UEV 对 ocxref 和 OCx 输出毫无影响。同步的精度可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

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 164* 给出了一个示例。

图 164. 输出比较模式，翻转 OC1



18.3.9 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。

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，为了与 ETRF 相符（在下一个 PWM 周期之前，ETR 信号上的一个外部事件能够清除 OCxREF），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。

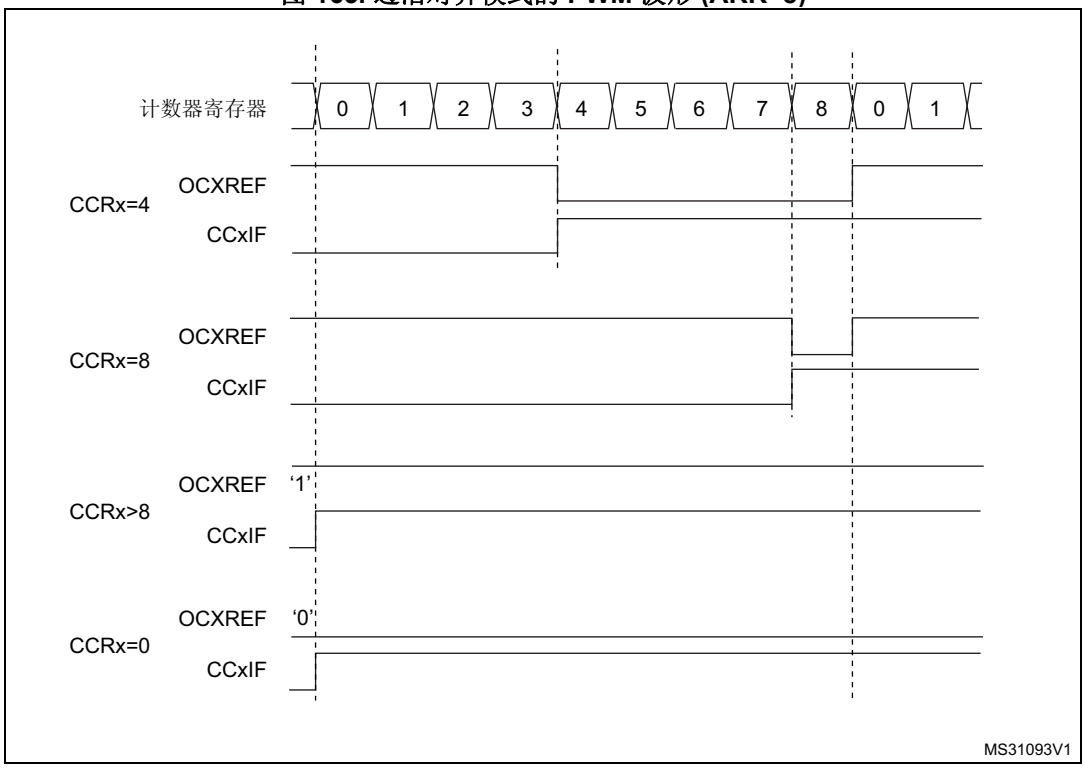
PWM 边沿对齐模式

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 509 页的递增计数模式*。

以下以 PWM 模式 1 为例。只要 TIMx_CNT < TIMx_CCRx，PWM 参考信号 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 165* 举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

图 165. 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 512 页的递减计数模式*。

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。

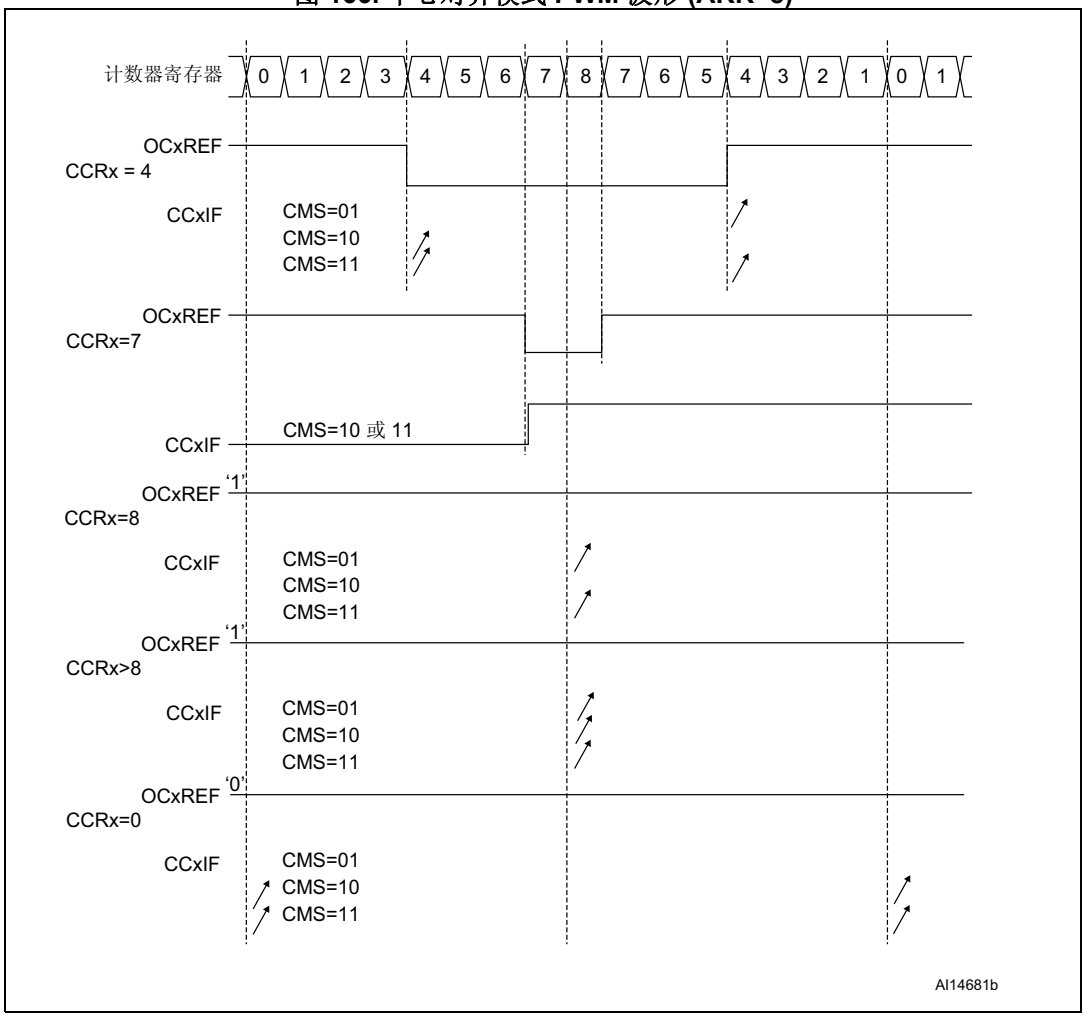
PWM 中心对齐模式

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 514 页的中心对齐模式（递增/递减计数）*。

图 166 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 166. 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。

18.3.10 单脉冲模式

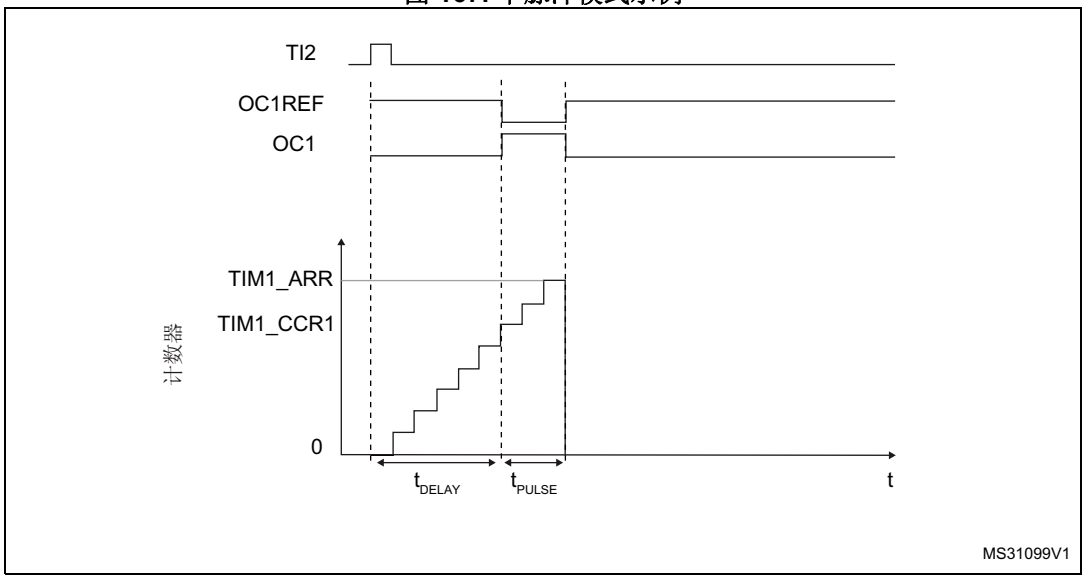
单脉冲模式 (OPM) 是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

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：

- 递增计数时：CNT < CCRx ≤ ARR（特别注意，0 < CCRx），
- 递减计数时：CNT>CCRx。

图 167. 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_CCR + 1) 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。

18.3.11 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”），可使 OCxREF 信号变为低电平。OCxREF 信号将保持低电平，直到发生下一更新事件 (UEV)。

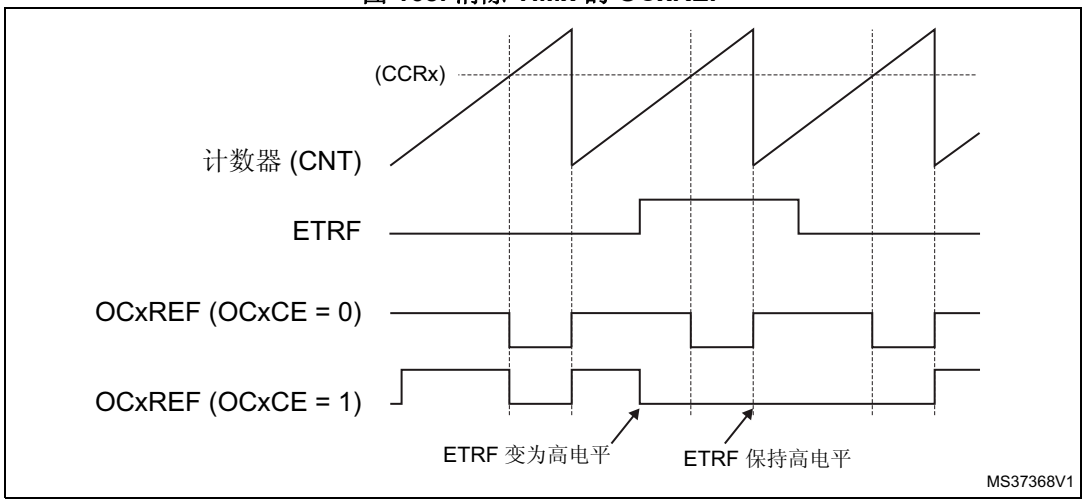
此功能仅能用于输出比较模式和 PWM 模式，而不适用于强制输出模式。

例如，ETR 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 168 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 168. 清除 TIMx 的 OCxREF



1. 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。

18.3.12 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。

TI1 和 TI2 两个输入用于连接增量编码器。请参见*表 105*。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，计数器会根据增量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 105. 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 169 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，TI1FP2 映射到 TI2 上）
- CC1P=“0”，CC1NP=“0”，IC1F=“0000”（TIMx_CCER 寄存器，TI1FP1 未反相，TI1FP1=TI1）
- CC2P=“0”，CC2NP=“0”，IC2F=“0000”（TIMx_CCER 寄存器，TI2FP2 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 169. 编码器接口模式下的计数器工作示例

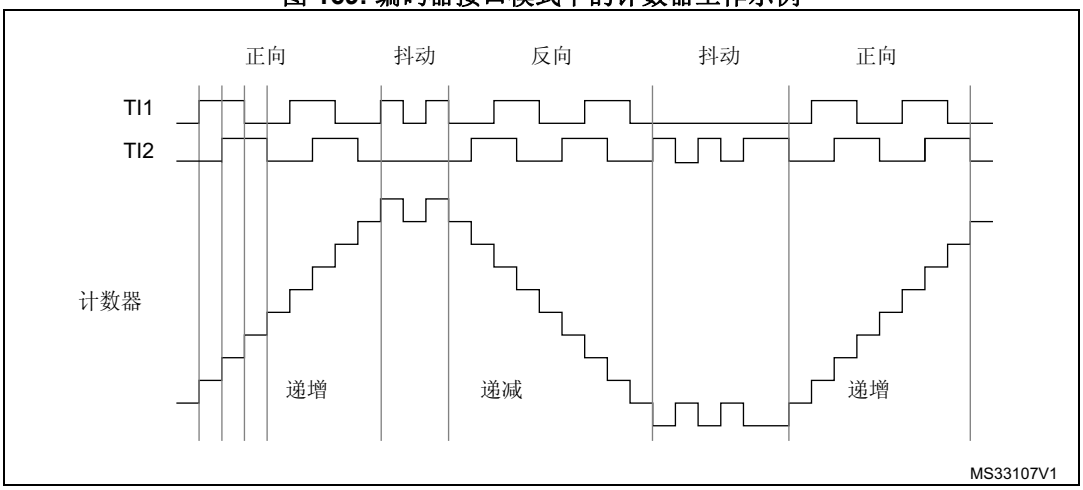
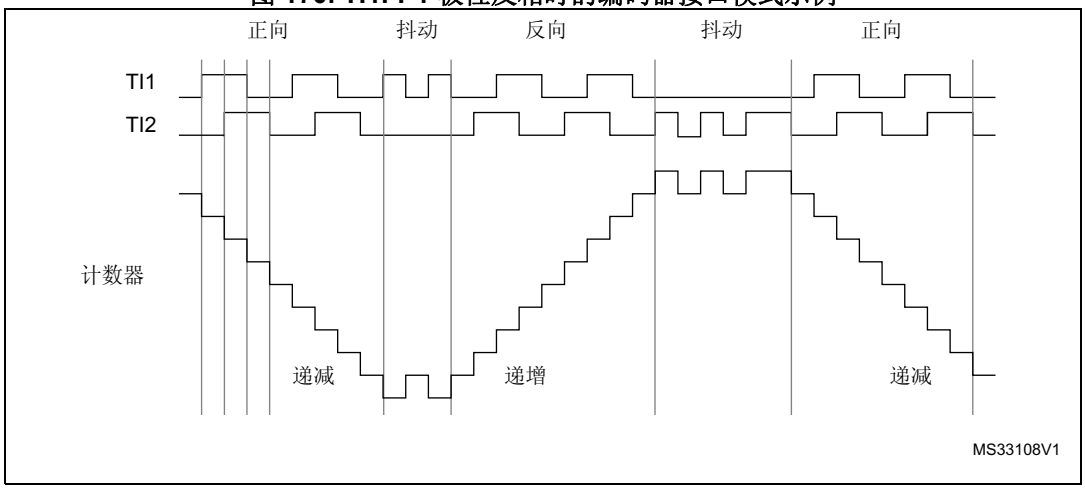


图 170 说明了 TI1FP1 极性反相时的计数器行为示例（除 CC1P=1 外，其他配置与上例相同）。

图 170. 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。

18.3.13 定时器输入异或功能

借助 TIM_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。

18.3.14 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。

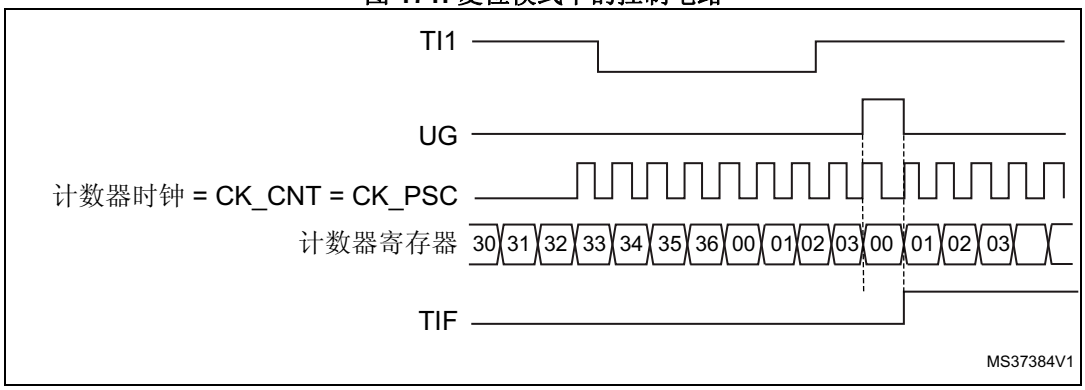
在以下示例中，TI1 输入上出现上升沿时，递增计数器清零：

- 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 171. 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。

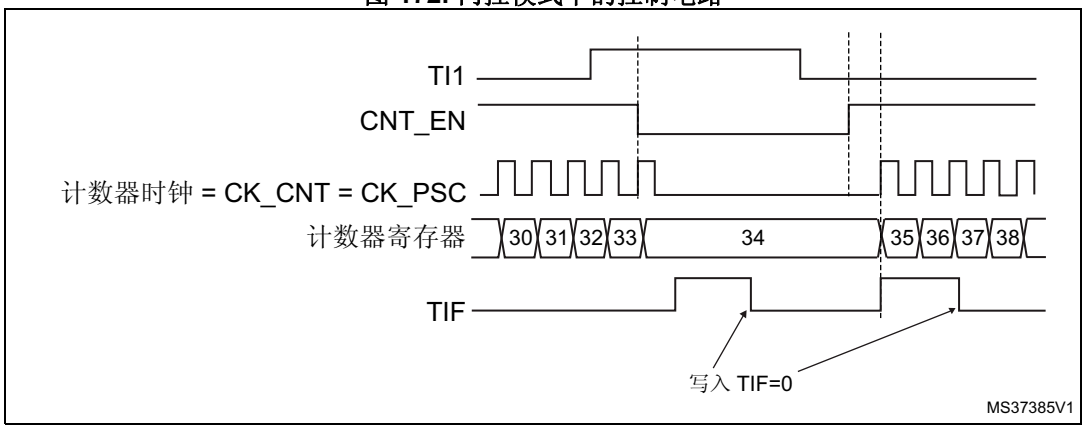
在以下示例中，递增计数器仅在 TI1 输入为低电平时计数：

- 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，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 172. 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。

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。

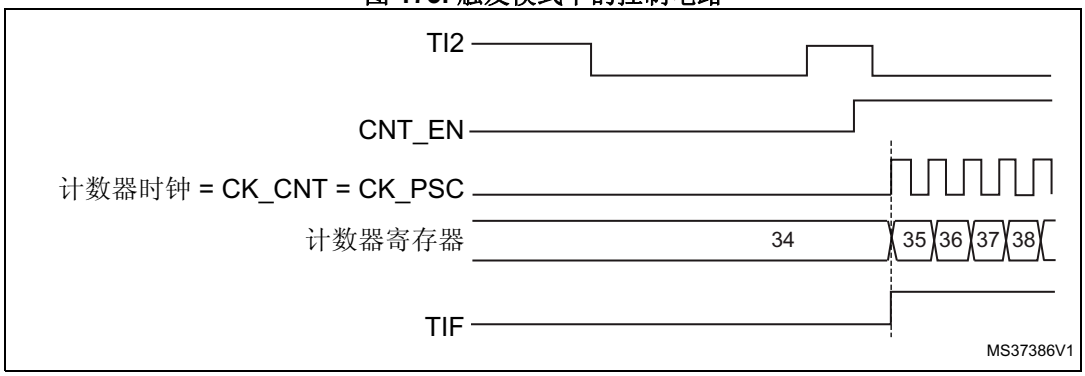
在以下示例中，TI2 输入上出现上升沿时，递增计数器启动：

- 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，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 173. 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。

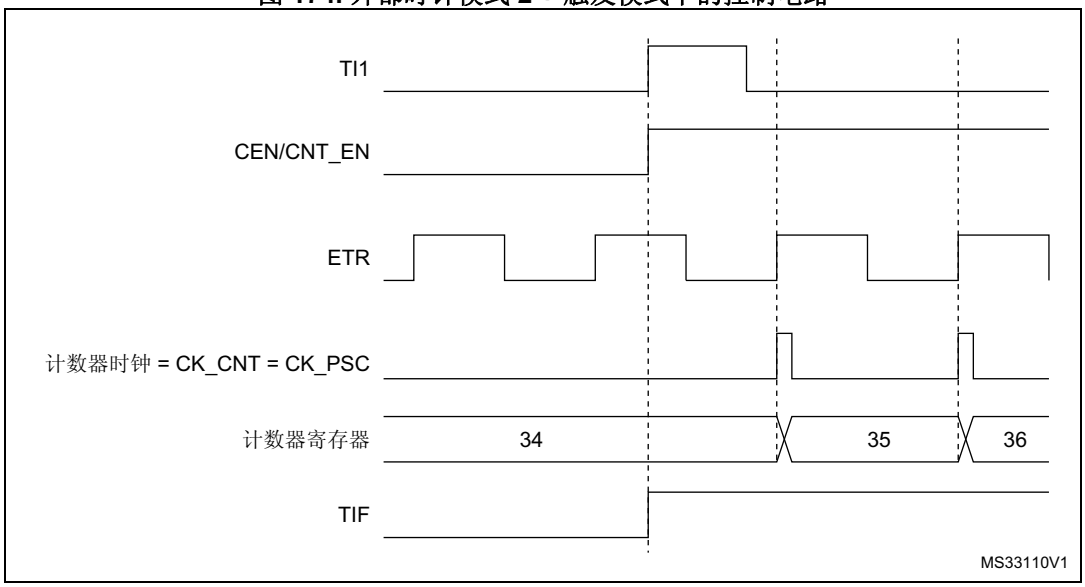
在以下示例中，只要 TI1 出现上升沿，递增计数器即会在 ETR 信号的每个上升沿处递增：

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 寄存器中的 CC1S = 01，只选择输入捕获源
   - TIMx_CCER 寄存器中的 CC1P = 0，确认极性（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 174. 外部时钟模式 2 + 触发模式下的控制电路



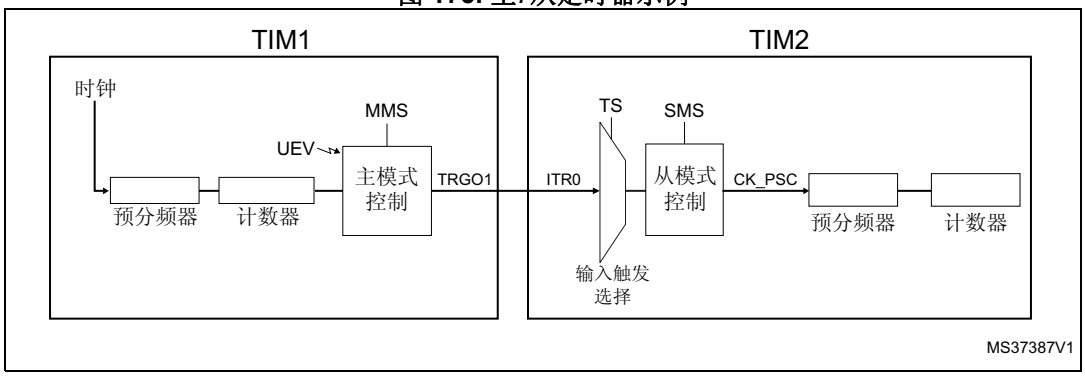
18.3.15 定时器同步

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 175*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 175. 主/从定时器示例



例如，可以将定时器 1 配置为定时器 2 的预分频器。请参见*图 175*。为此：

- 将定时器 1 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 TIM1_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。
- 要将定时器 1 的 TRGO1 输出连接到定时器 2，必须将定时器 2 配置为从模式，使用 ITR0 作为内部触发。通过 TIM2_SMCR 寄存器中的 TS 位（写入 TS=000）可对此进行选择。
- 然后将从模式控制器设为外部时钟模式 1（在 TIM2_SMCR 寄存器中写入 SMS=111）。这样一来，定时器 2 的时钟将由定时器 1 周期性触发信号的上升沿（与定时器 1 的计数器上溢对应）提供。
- 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： *如果选择定时器 1 的 OCx 信号作为触发输出 (MMS=1xx)，该信号的上升沿将用于驱动定时器 2 的计数器。*

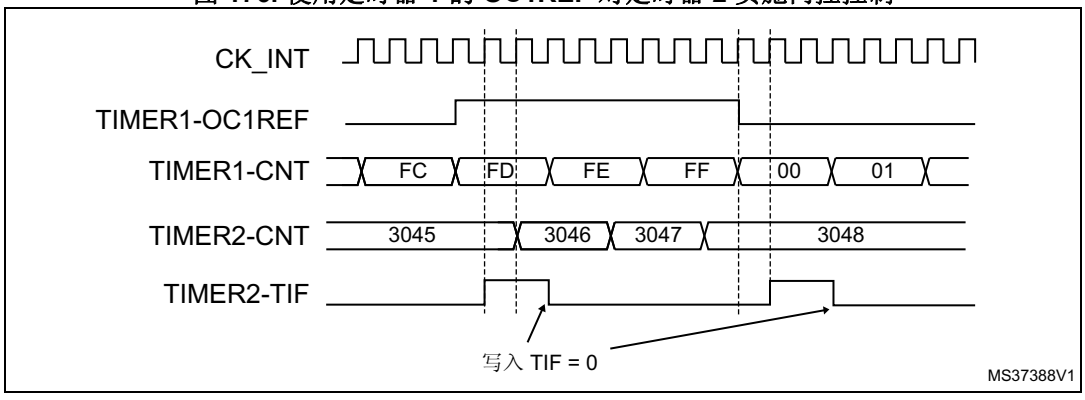
使用一个定时器使能另一个定时器

本例中通过定时器 1 的输出比较 1 来使能定时器 2。相关连接图，请参见*图 175*。仅当定时器 1 的 OC1REF 为高电平时，定时器 2 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 ($f_{CK\_CNT} = f_{CK\_INT}/3$)。

- 将定时器 1 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（TIM1_CR2 寄存器中的 MMS=100）。
- 配置定时器 1 的 OC1REF 波形（TIM1_CCMR1 寄存器）。
- 配置定时器 2 以接收来自定时器 1 的输入触发（TIM2_SMCR 寄存器中的 TS=000）。
- 将定时器 2 配置为门控模式（TIM2_SMCR 寄存器中的 SMS=101）。
- 通过向 CEN 位（TIM2_CR1 寄存器）写入“1”使能定时器 2。
- 通过向 CEN 位（TIM1_CR1 寄存器）写入“1”启动定时器 1。

注： *计数器 2 的时钟与计数器 1 不同步，此模式仅影响定时器 2 的计数器使能信号。*

图 176. 使用定时器 1 的 OC1REF 对定时器 2 实施门控控制

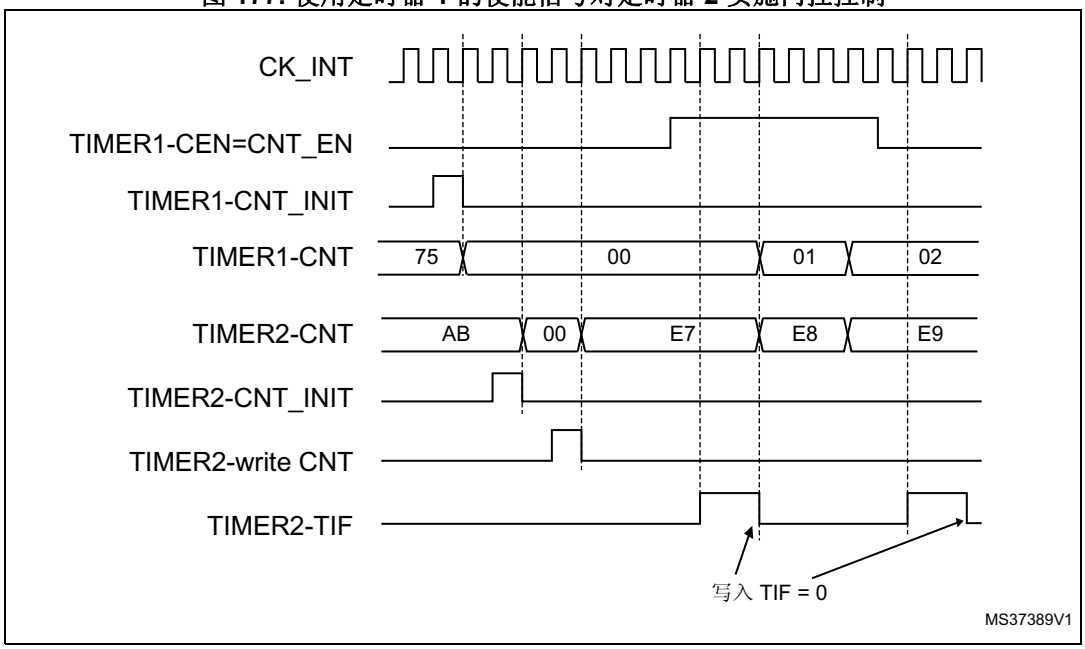


在*图 176* 的示例中，定时器 2 的计数器和预分频器在启动前未进行初始化。因此从各自的当前值开始计数。启动定时器 1 之前，通过复位这两个定时器可以从指定值开始计数。这样便可以在定时器计数器中写入所需的任意值。两个定时器都可通过软件使用 TIMx_EGR 寄存器中的 UG 位轻松复位。

在下一示例中，定时器 1 与定时器 2 同步。定时器 1 为主模式，从 0 开始计数。定时器 2 为从模式，从 0xE7 开始计数。两个定时器的预分频比相同。在 TIM1_CR1 寄存器中通过向 CEN 位写入“0”来禁止定时器 1 时，定时器 2 将停止：

- 将定时器 1 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（TIM1_CR2 寄存器中的 MMS=100）。
- 配置定时器 1 的 OC1REF 波形（TIM1_CCMR1 寄存器）。
- 配置定时器 2 以接收来自定时器 1 的输入触发（TIM2_SMCR 寄存器中的 TS=000）。
- 将定时器 2 配置为门控模式（TIM2_SMCR 寄存器中的 SMS=101）。
- 通过向 UG 位（TIM1_EGR 寄存器）写入“1”复位定时器 1。
- 通过向 UG 位（TIM2_EGR 寄存器）写入“1”复位定时器 2。
- 通过在定时器 2 的计数器 (TIM2_CNTL) 中写入“0xE7”使定时器 2 初始化为 0xE7。
- 通过向 CEN 位（TIM2_CR1 寄存器）写入“1”使能定时器 2。
- 通过向 CEN 位（TIM1_CR1 寄存器）写入“1”启动定时器 1。
- 通过向 CEN 位（TIM1_CR1 寄存器）写入“0”停止定时器 1。

图 177. 使用定时器 1 的使能信号对定时器 2 实施门控控制

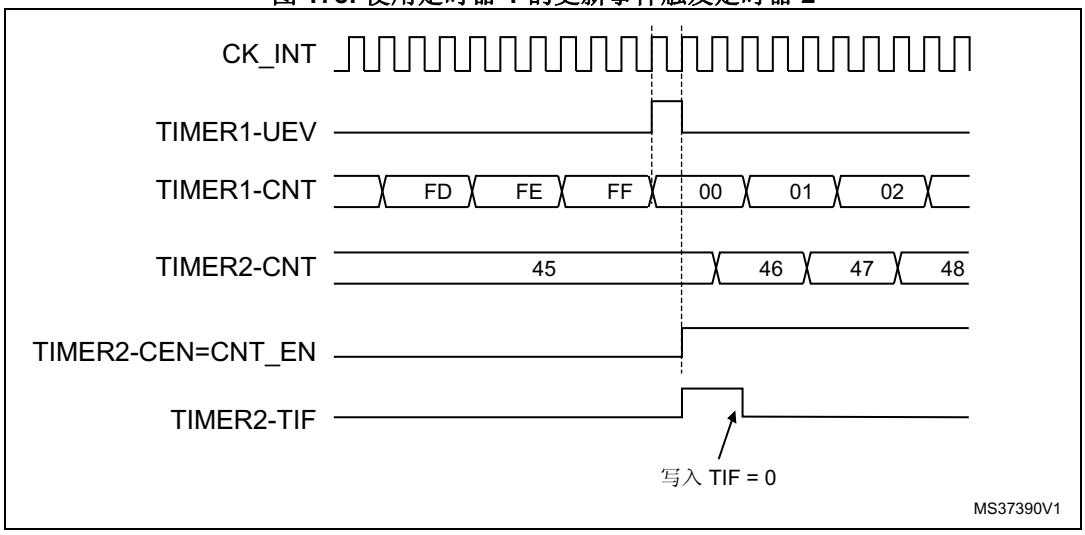


使用一个定时器启动另一个定时器

本例中使用定时器 1 的更新事件使能定时器 2。相关连接图，请参见*图 175*。只要定时器 1 生成更新事件，定时器 2 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 2 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 TIM2_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 ($f_{CK\_CNT} = f_{CK\_INT}/3$)。

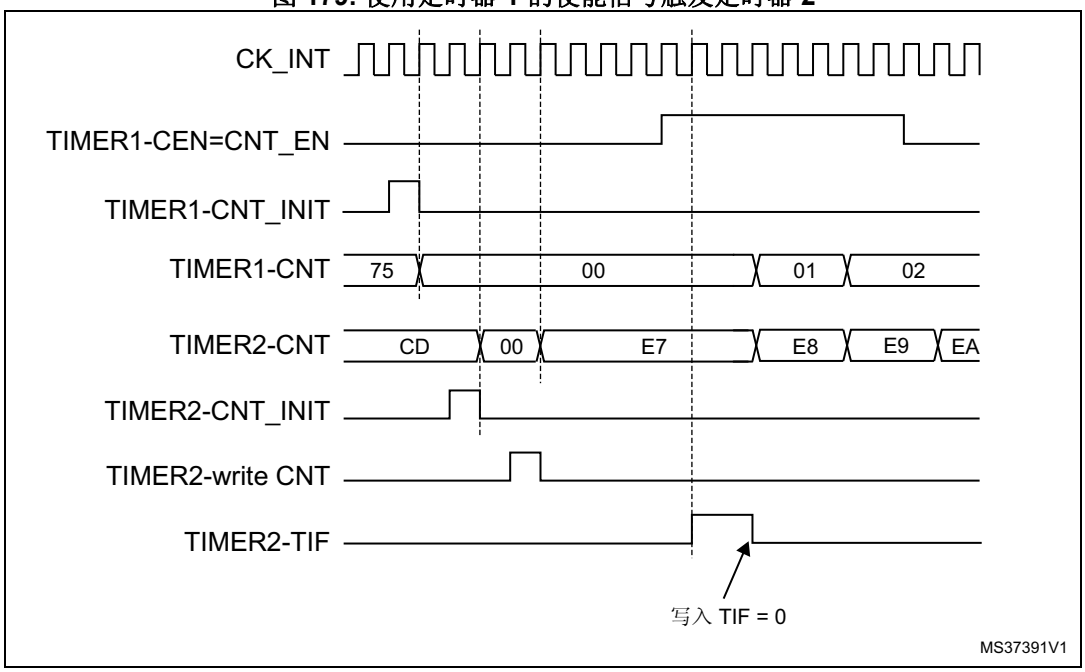
- 将定时器 1 配置为主模式，发送其更新事件 (UEV) 作为触发输出（TIM1_CR2 寄存器中的 MMS=010）。
- 配置定时器 1 的周期（TIM1_ARR 寄存器）。
- 配置定时器 2 以接收来自定时器 1 的输入触发（TIM2_SMCR 寄存器中的 TS=000）。
- 将定时器 2 配置为触发模式（TIM2_SMCR 寄存器中的 SMS=110）。
- 通过向 CEN 位（TIM1_CR1 寄存器）写入“1”启动定时器 1。

图 178. 使用定时器 1 的更新事件触发定时器 2



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 179* 显示了与*图 176* 具有相同配置，只不过处于触发模式（TIM2_SMCR 寄存器中的 SMS=110）而非门控模式的计数行为。

图 179. 使用定时器 1 的使能信号触发定时器 2



将一个定时器用作另一个定时器的预分频器

例如，可以将定时器 1 配置为定时器 2 的预分频器。相关连接图，请参见*图 175*。为此：

- 将定时器 1 配置为主模式，发送其更新事件 (UEV) 作为触发输出（TIM1_CR2 寄存器中的 MMS=010）。这样便会在计数器每次发生上溢时输出一个周期性信号。
- 配置定时器 1 的周期（TIM1_ARR 寄存器）。
- 配置定时器 2 以接收来自定时器 1 的输入触发（TIM2_SMCR 寄存器中的 TS=000）。
- 将定时器 2 配置为外部时钟模式（TIM2_SMCR 寄存器中的 SMS=111）。
- 通过向 CEN 位（TIM2_CR1 寄存器）写入“1”启动定时器 2。
- 通过向 CEN 位（TIM1_CR1 寄存器）写入“1”启动定时器 1。

使用一个外部触发同步的启动 2 个定时器

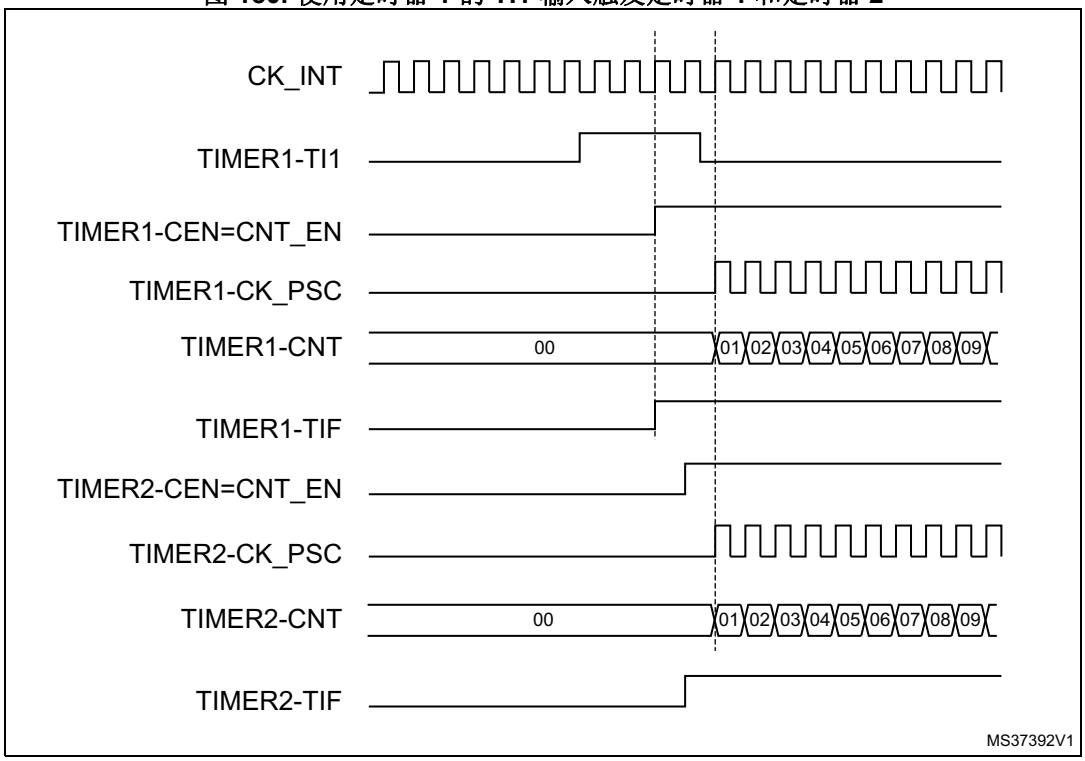
本例中，定时器 1 的 TI1 输入出现上升沿时使能定时器 1，使能定时器 1 的同时使能定时器 2。相关连接图，请参见*图 175*。要确保两个计数器对齐，定时器 1 必须配置为主/从模式（对应的 TI1 为从，对应定时器 2 为主）：

- 将定时器 1 配置为主模式，发送其使能信号作为触发输出（TIM1_CR2 寄存器中的 MMS=001）。
- 将定时器 1 配置为从模式以接收来自 TI1 的输入触发（TIM1_SMCR 寄存器中的 TS=100）。
- 将定时器 1 配置为触发模式（TIM1_SMCR 寄存器中的 SMS=110）。
- 通过写入 MSM=1（TIMx_SMCR 寄存器）将定时器 1 配置为主/从模式。
- 配置定时器 2 以接收来自定时器 1 的输入触发（TIM2_SMCR 寄存器中的 TS=000）。
- 将定时器 2 配置为触发模式（TIM2_SMCR 寄存器中的 SMS=110）。

当 TI1（定时器 1）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/从模式在定时器 1 的 CNT_EN 与 CK_PSC 之间产生了延迟。*

图 180. 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2



18.3.16 调试模式

当微控制器进入调试模式时（带 FPU 的 Cortex®-M4 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 34.16.2 节：对定时器、看门狗、bxCAN 和 I2C 的调试支持*。

18.4 TIM2 到 TIM5 寄存器

有关寄存器说明中使用的缩写，请参见*第 50 页的第 1.2 节*。

32 位外设寄存器必须按字（32 位）写入数据。所有其他外设寄存器则必须按半字（16 位）或字（32 位）写入数据。而读访问可支持字节（8 位）、半字（16 位）或字（32 位）。

18.4.1 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.	Res.	Res.	CKD[1:0]		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	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)

此位域指示定时器时钟 (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 CCxS=00）的输出比较中断标志才置 1。
10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志才置 1。
11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CCxS=00）的输出比较中断标志都会置 1。

注： 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。

18.4.2 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）

位 6:4 **MMS[2:0]**：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)

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。

18.4.3 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

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 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**：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)

此位域可定义 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)

此位域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*表 106*。

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：编码器模式 1——计数器根据 TI1FP1 电平在 TI2FP2 边沿递增/递减计数。

010：编码器模式 2——计数器根据 TI2FP2 电平在 TI1FP1 边沿递增/递减计数。

011：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。

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被控制。

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——由所选触发信号 (TRGI) 的上升沿提供计数器时钟。

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。

表 106. TIMx 内部触发连接

从 TIM	ITR0 (TS = 000)	ITR1 (TS = 001)	ITR2 (TS = 010)	ITR3 (TS = 011)
TIM2	TIM1	TIM8	TIM3	TIM4
TIM3	TIM1	TIM2	TIM5	TIM4
TIM4	TIM1	TIM2	TIM3	TIM8
TIM5	TIM2	TIM3 或 LPTIM1(1)	TIM4	TIM8

1. 通过 LPTIM1_OR 寄存器位 3 选择 TIM3 或 LPTIM1。默认情况下选择 TIM3。

18.4.4 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，始终读为 0

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

18.4.5 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：

上溢或下溢（对于 TIM2 到 TIM5）以及当 TIMx_CR1 寄存器中 UDIS = 0 时。

TIMx_CR1 寄存器中的 URS = 0 且 UDIS = 0，并且由软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。

TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且 CNT 由触发事件重新初始化时（参见同步控制寄存器说明）。

18.4.6 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)。

18.4.7 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。此寄存器的所有其他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
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 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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道配置为输出

01：CC2 通道配置为输入，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)

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。

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： **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）。其他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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出。

01：CC1 通道配置为输入，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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC2 通道配置为输出。
01：CC2 通道配置为输入，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)

此位域可定义 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)

此位域定义 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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC1 通道配置为输出
01：CC1 通道配置为输入，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。

18.4.8 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。

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 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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC4 通道配置为输出
01：CC4 通道配置为输入，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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC3 通道配置为输出
01：CC3 通道配置为输入，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 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)
此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC4 通道配置为输出
01：CC4 通道配置为输入，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)
此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC3 通道配置为输出
01：CC3 通道配置为输入，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。

18.4.9 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 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　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 互补输出极性 (Capture/Compare 2 complementary 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：
在这种情况下，CC1NP 必须保持清零。
CC1 通道配置为输入：
此位与 CC1P 配合使用，用以定义 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。
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 107. 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 引脚的状态取决于通道 OCx 的状态以及 GPIO 寄存器。*

18.4.10 TIMx 计数器 (TIMx_CNT)

TIMx 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)

18.4.11 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。

18.4.12 TIMx 自动重载寄存器 (TIMx_ARR)

TIMx 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 507 页的第 18.3.1 节：时基单元*。

当自动重载值为空时，计数器不工作。

18.4.13 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 0000

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 (High Capture/Compare 1 value)（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。

18.4.14 TIMx 捕获/比较寄存器 2 (TIMx_CCR2)

TIMx capture/compare register 2

偏移地址：0x38

复位值：0x0000 0000

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 (High Capture/Compare 2 value)（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_CCMR 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。

18.4.15 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 0000

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 (High Capture/Compare 3 value)（对于 TIM2 和 TIM5）

位 15:0 **CCR3[15:0]**：捕获/比较 3 值的低 16 位 (Low Capture/Compare 3 value)

如果通道 CC3 配置为输出：

CCR3 是捕获/比较寄存器 3 的预装载值。

如果没有通过 TIMx_CCMR 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。

18.4.16 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 0000

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 (High Capture/Compare 4 value)（对于 TIM2 和 TIM5）

位 15:0 **CCR4[15:0]**：捕获/比较 4 值的低 16 位 (Low Capture/Compare 4 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_CCMR 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。

18.4.17 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。

18.4.18 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。

DMA 连续传送功能使用方法示例

本例中，定时器 DMA 连续传送功能用于将 CCRx 寄存器（x = 2、3、4）的内容更新为通过 DMA 传输到 CCRx 寄存器中的多个半字。

具体操作步骤如下：

1. 将相应的 DMA 通道配置如下：
   - DMA 通道外设地址为 DMAR 寄存器地址
   - DMA 通道存储器地址为包含要通过 DMA 传输到 CCRx 寄存器的数据的 RAM 缓冲区地址。
   - 要传输的数据量 = 3（参见下文注释）。
   - 禁止循环模式。
2. 通过将 DBA 和 DBL 位域配置如下来配置 DCR 寄存器：
   DBL = 3 次传输，DBA = 0xE。
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

注： *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。*

18.4.19 TIM2 选项寄存器 (TIM2_OR)

TIM2 option register

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		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**：内部触发 1 重映射 (Internal trigger 1 remap)
由软件置 1 和清零。
00：TIM8_TRGOUT
01：保留
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。

18.4.20 TIM5 选项寄存器 (TIM5_OR)

TIM5 option register

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		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**：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。

18.4.21 TIMx 寄存器映射

TIMx 寄存器按照下表所述方式映射：

表 108. TIM2 到 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.	Res.	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.	Res.	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
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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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
	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
0x1C	**TIMx_CCMR2** Output Compare mode	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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
	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
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
0x24	**TIMx_CNT**	CNT[31:16] (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

表 108. TIM2 到 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
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	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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**	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	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																					0	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.	IT4_RMP		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 2.2.2 节：存储器映射和寄存器边界地址*。

19 通用定时器（TIM9 到 TIM14）

19.1 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 19.3.12 节*中所述，它们可以同步操作。

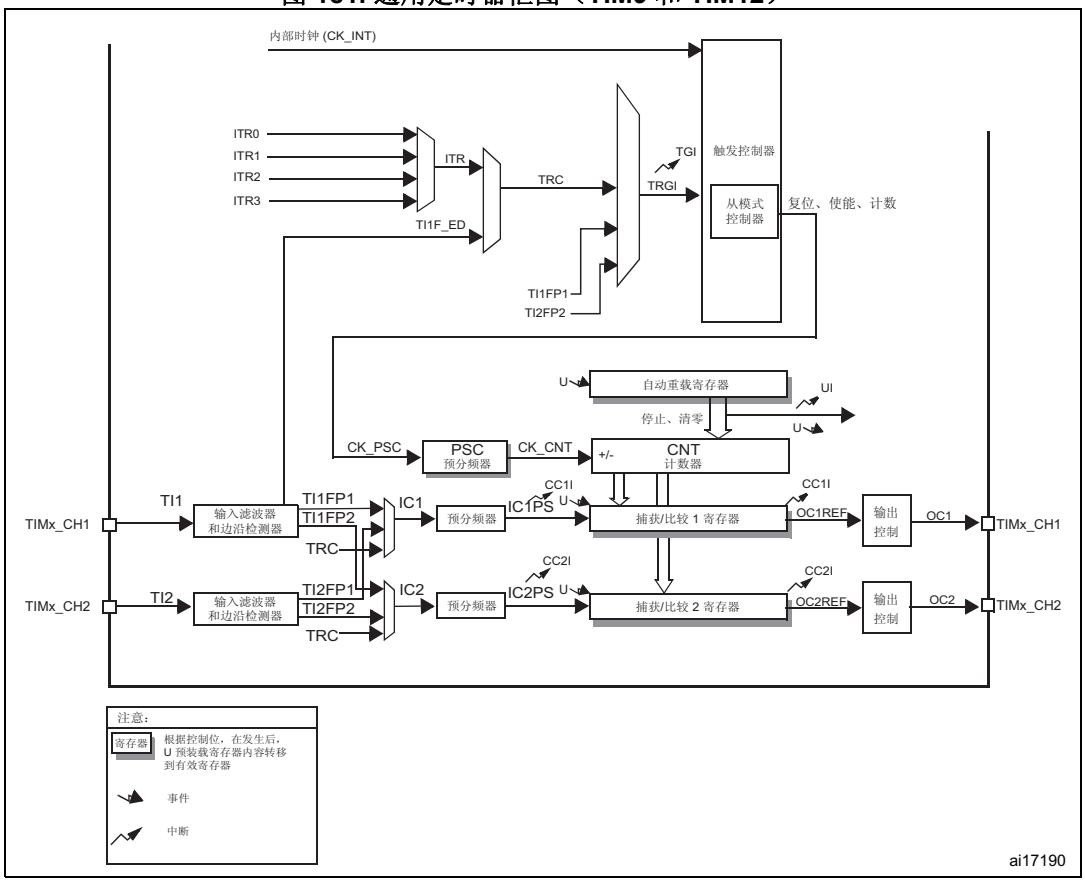
19.2 TIM9 到 TIM14 主要特性

19.2.1 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 181. 通用定时器框图（TIM9 和 TIM12）

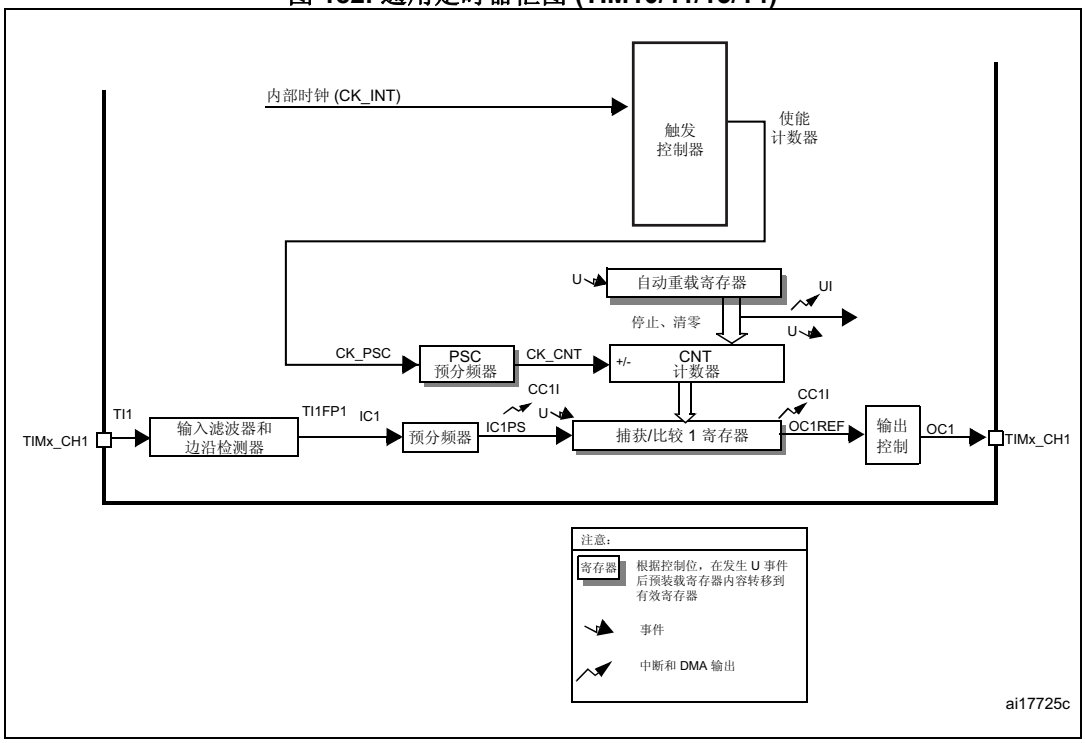


19.2.2 TIM10/TIM11 和 TIM13/TIM14 主要特性

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 182. 通用定时器框图 (TIM10/11/13/14)



19.3 TIM9 到 TIM14 功能说明

19.3.1 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 183 和 *图 184* 以一些示例说明在预分频比实时变化时计数器的行为。

图 183. 预分频器分频由 1 变为 2 时的计数器时序图

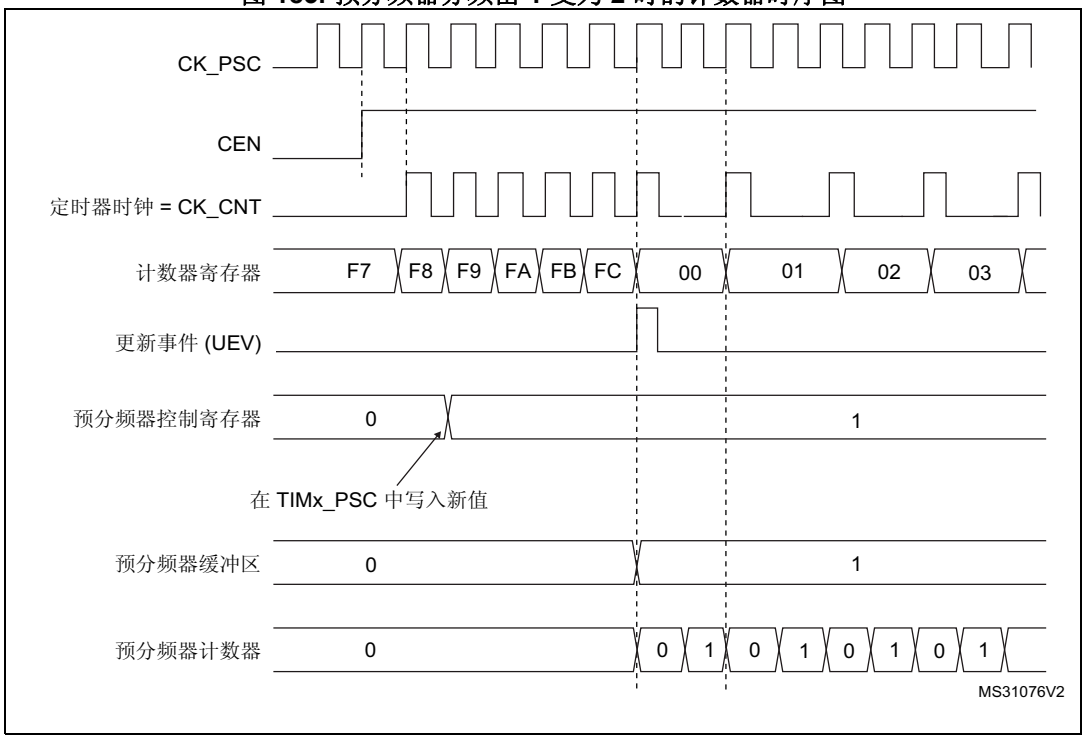
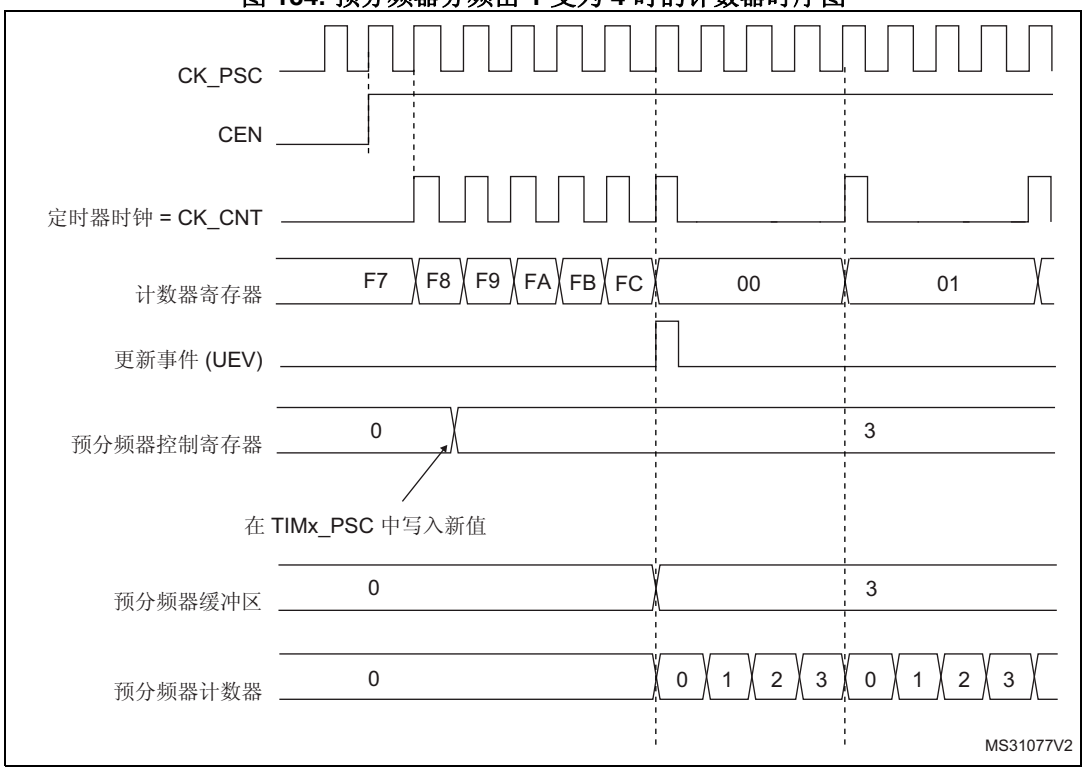


图 184. 预分频器分频由 1 变为 4 时的计数器时序图



19.3.2 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。

让 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 185. 计数器时序图，1 分频内部时钟

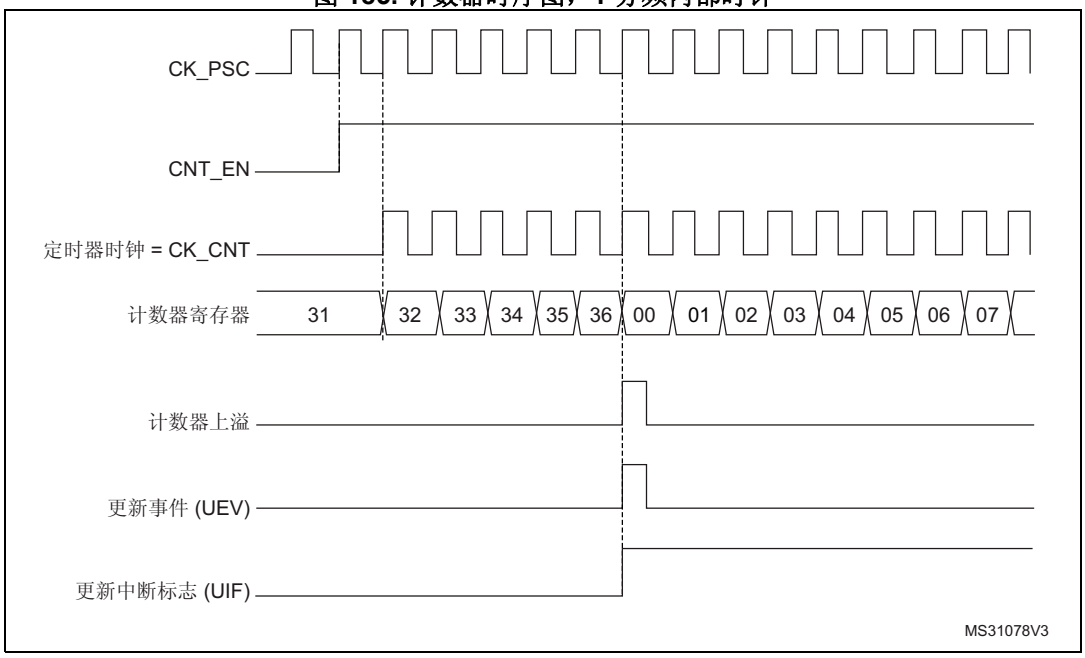


图 186. 计数器时序图，2 分频内部时钟

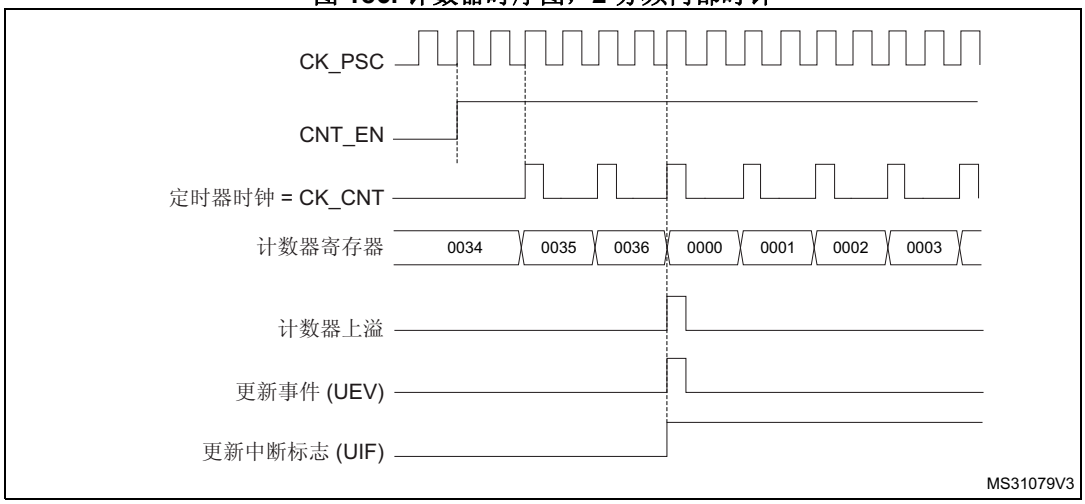


图 187. 计数器时序图，4 分频内部时钟

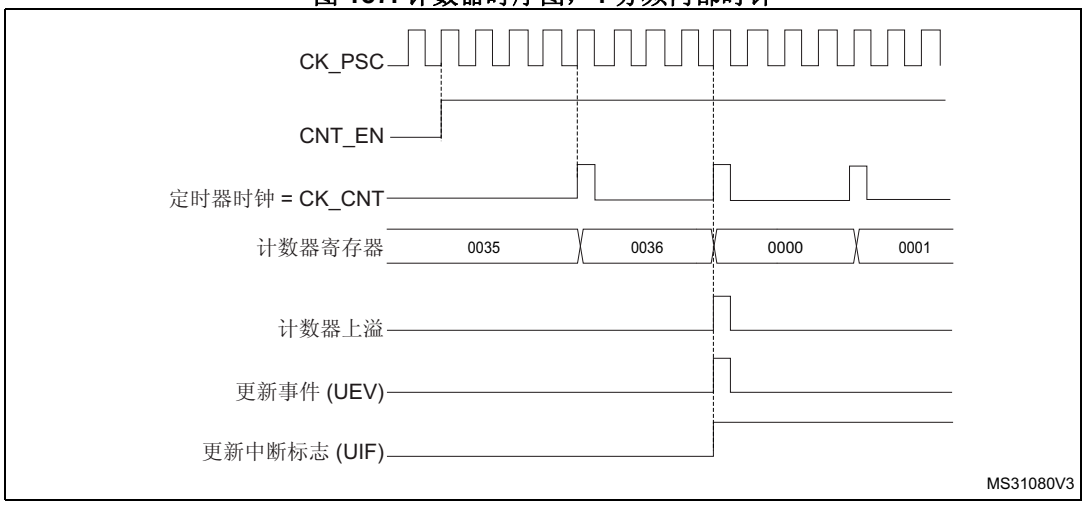


图 188. 计数器时序图，N 分频内部时钟

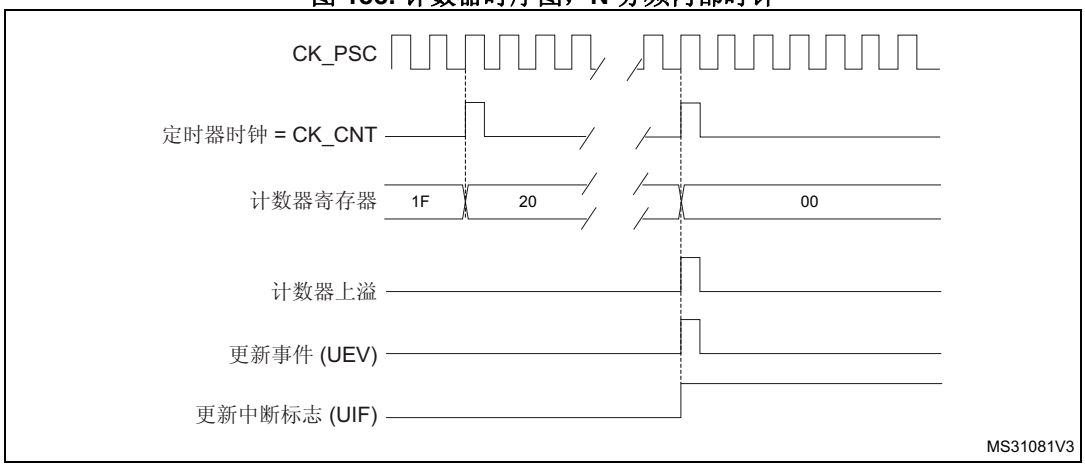


图 189. 计数器时序图，ARPE=0 时更新事件 （TIMx_ARR 未预装载）

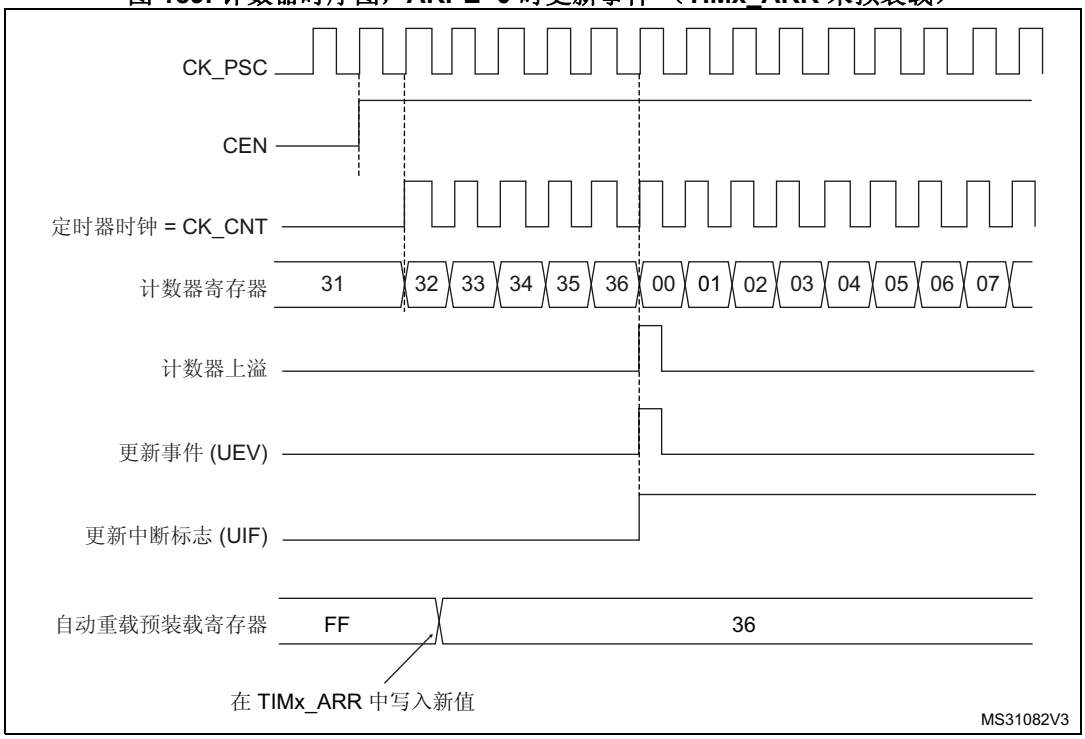
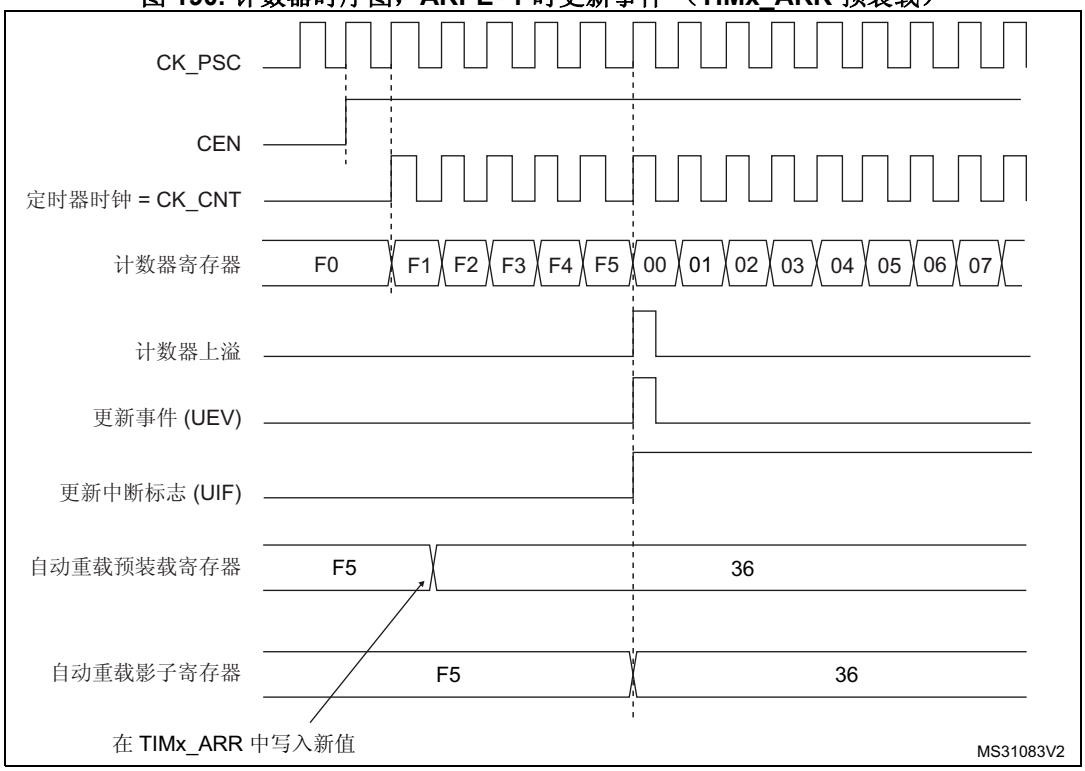


图 190. 计数器时序图，ARPE=1 时更新事件 （TIMx_ARR 预装载）



19.3.3 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**）：连接来自其他计数器的触发输出。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*。

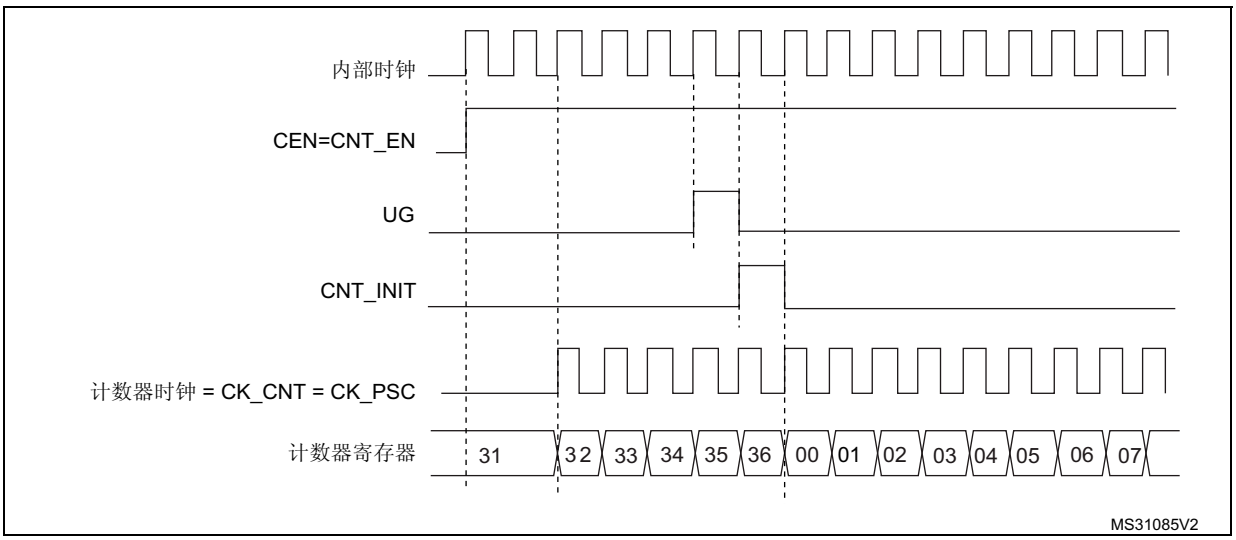
内部时钟源 (CK_INT)

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 191 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）。

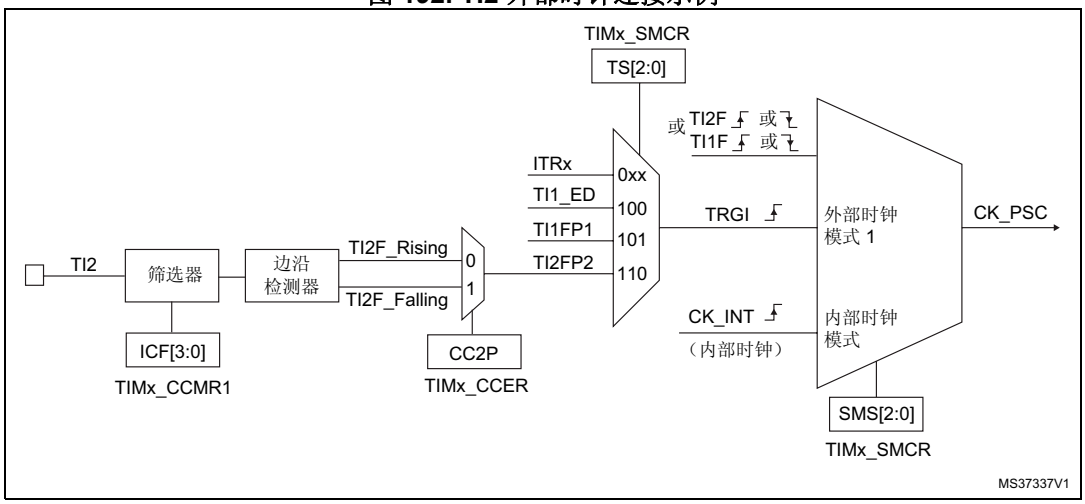
图 191. 正常模式下的控制电路，1 分频内部时钟



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 192. 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：

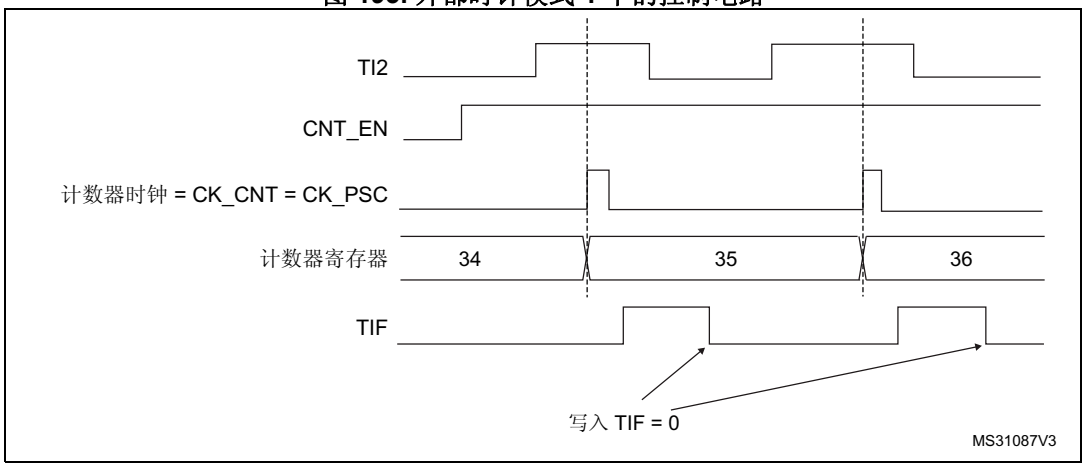
1. 通过在 TIMx_CCMR1 寄存器中写入 CC2S = “01”来配置通道 2，使其能够检测 TI2 输入的上升沿。
2. 通过在 TIMx_CCMR1 寄存器中写入 IC2F[3:0] 位来配置输入滤波带宽（如果不需要任何滤波器，请保持 IC2F= “0000”）。
3. 通过在 TIMx_CCER 寄存器中写入 CC2P= “0”和 CC2NP= “0”来选择上升沿极性。
4. 通过在 TIMx_SMCR 寄存器中写入 SMS= “111”，使定时器在外部时钟模式 1 下工作。
5. 通过在 TIMx_SMCR 寄存器中写入 TS= “110”来选择 TI2 作为触发输入源。
6. 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 193. 外部时钟模式 1 下的控制电路



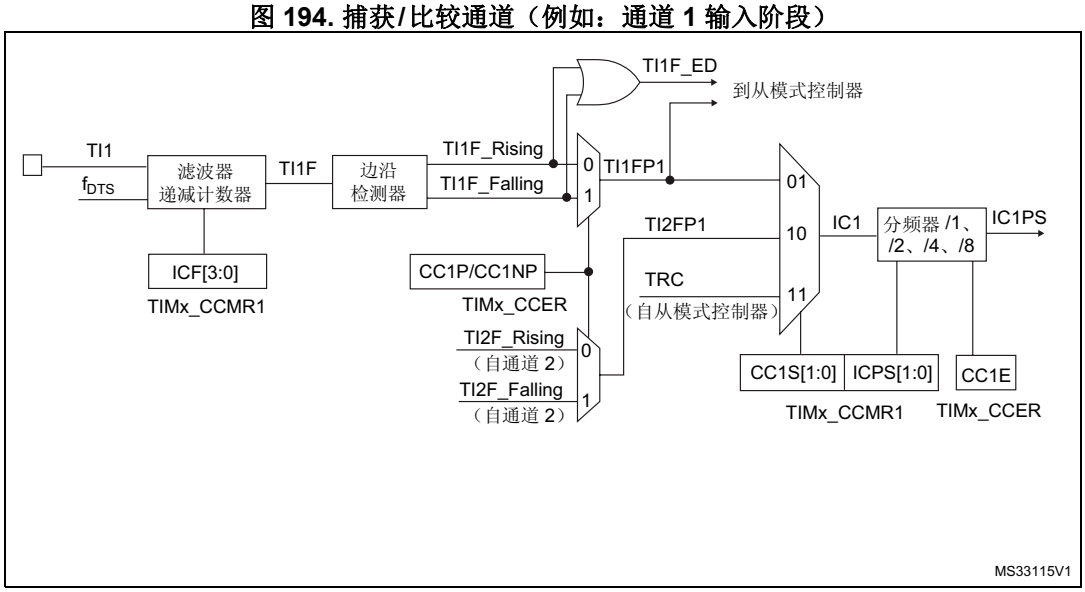
19.3.4 捕获/比较通道

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 194 到*图 196* 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 194. 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 195. 捕获/比较通道 1 主电路

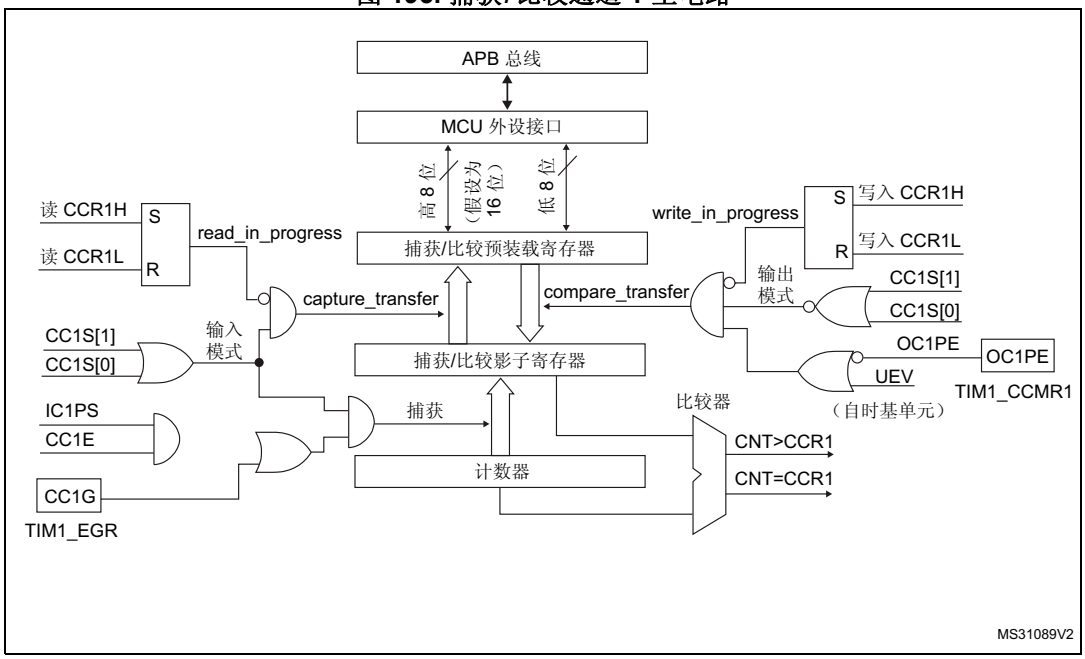
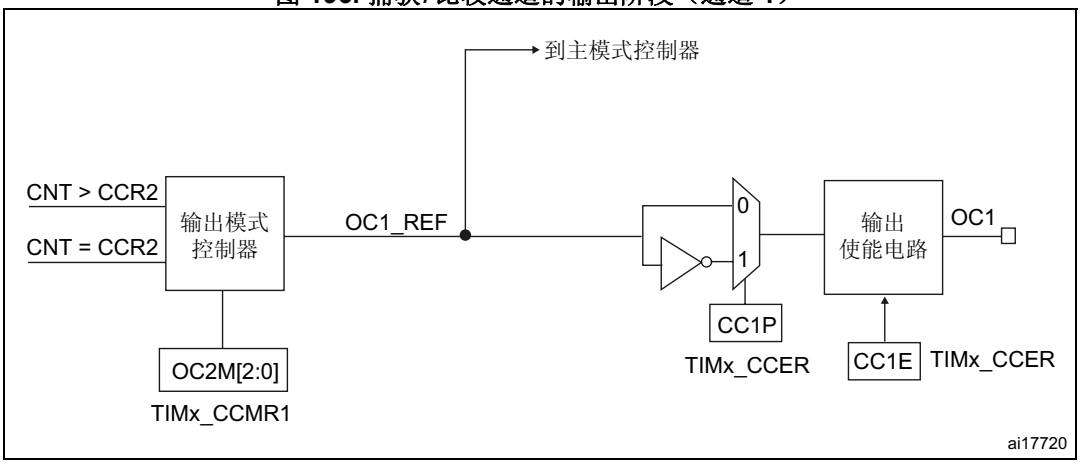


图 196. 捕获/比较通道的输出阶段（通道 1）



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。

19.3.5 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。*

19.3.6 PWM 输入模式（仅适用于 TIM9/12）

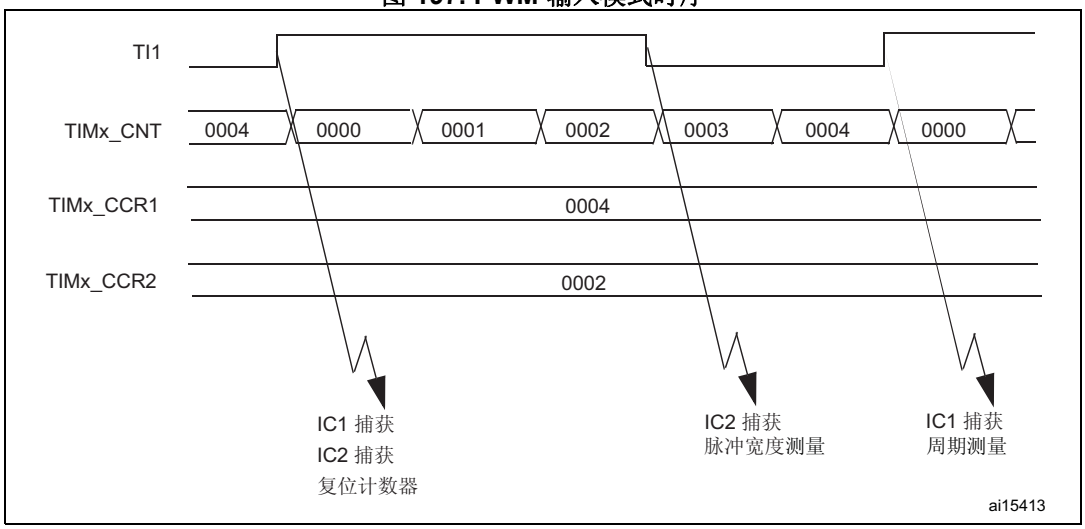
此模式是输入捕获模式的一个特例。其实现步骤与输入捕获模式基本相同，仅存在以下不同之处：

- 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 197. 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。

19.3.7 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。

19.3.8 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。

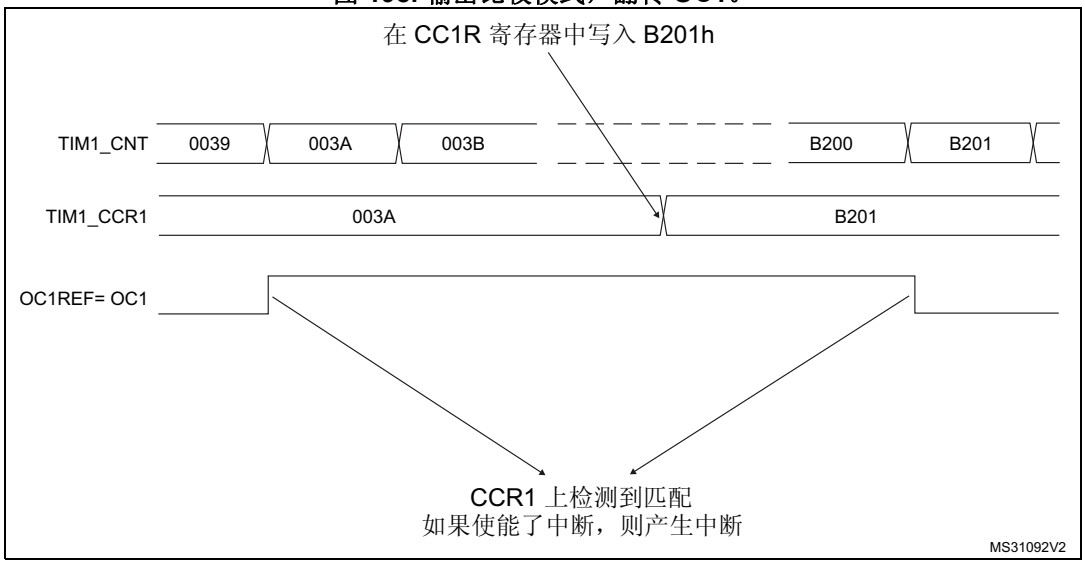
在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出毫无影响。同步的精度可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

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 198* 给出了一个示例。

图 198. 输出比较模式，翻转 OC1。



19.3.9 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。

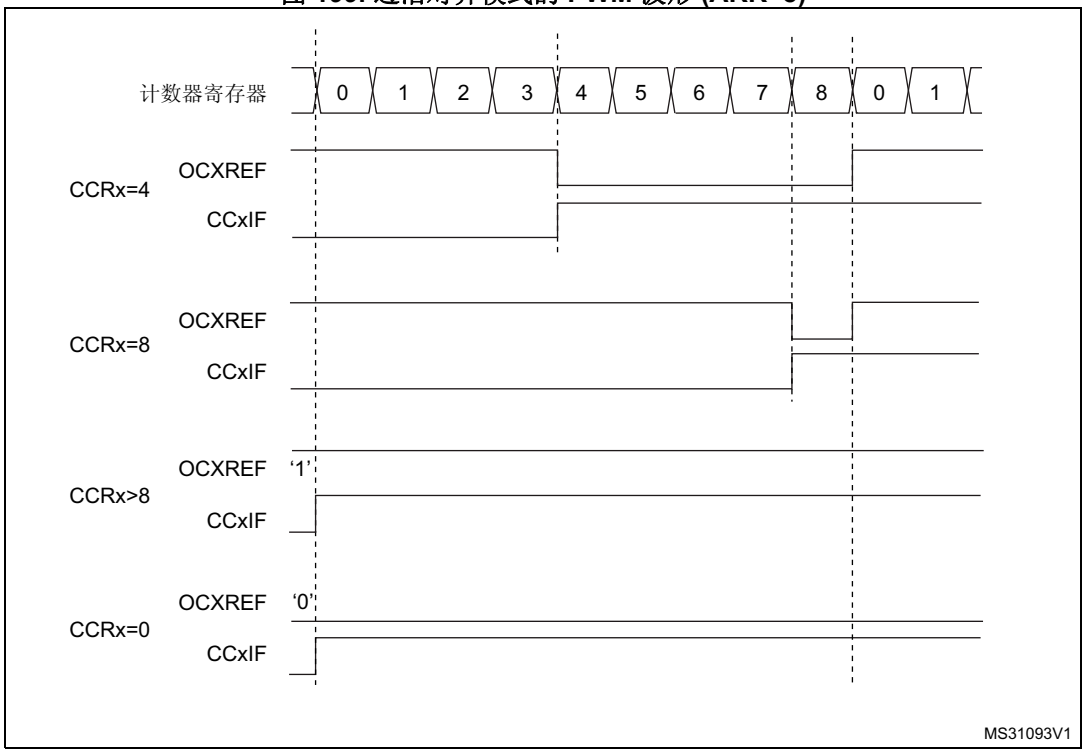
在 PWM 模式（1 或 2）下，TIMx_CNT 始终与 TIMx_CCRx 进行比较，以确定 TIMx_CNT ≤ TIMx_CCRx 是否成立。

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 199* 举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

图 199. 边沿对齐模式的 PWM 波形 (ARR=8)



19.3.10 单脉冲模式

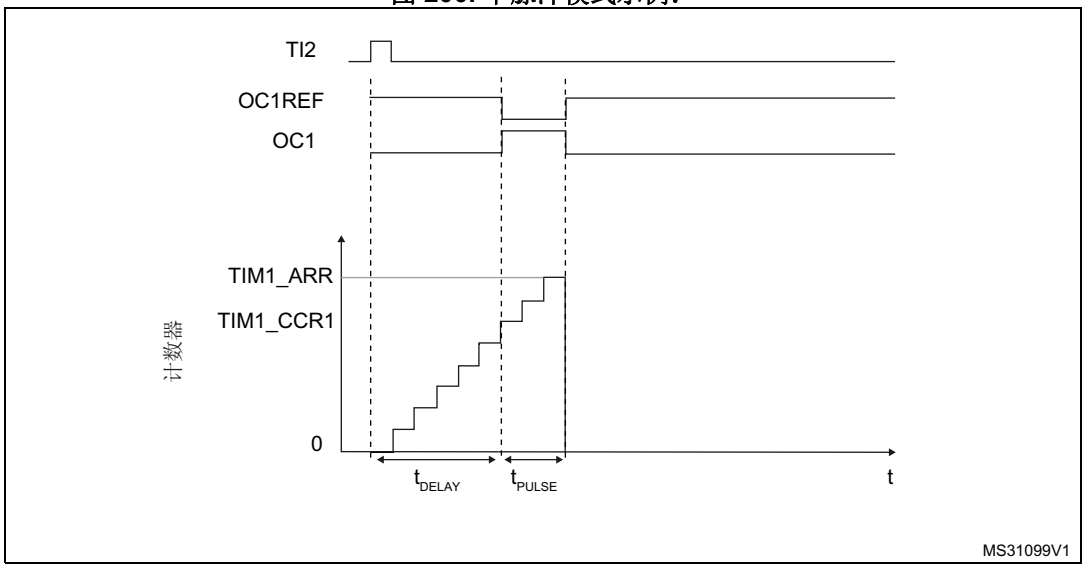
单脉冲模式 (OPM) 是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

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 200. 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。

19.3.11 TIM9/12 外部触发同步

TIM9/12 定时器可与外部触发实现同步通过下列模式：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。

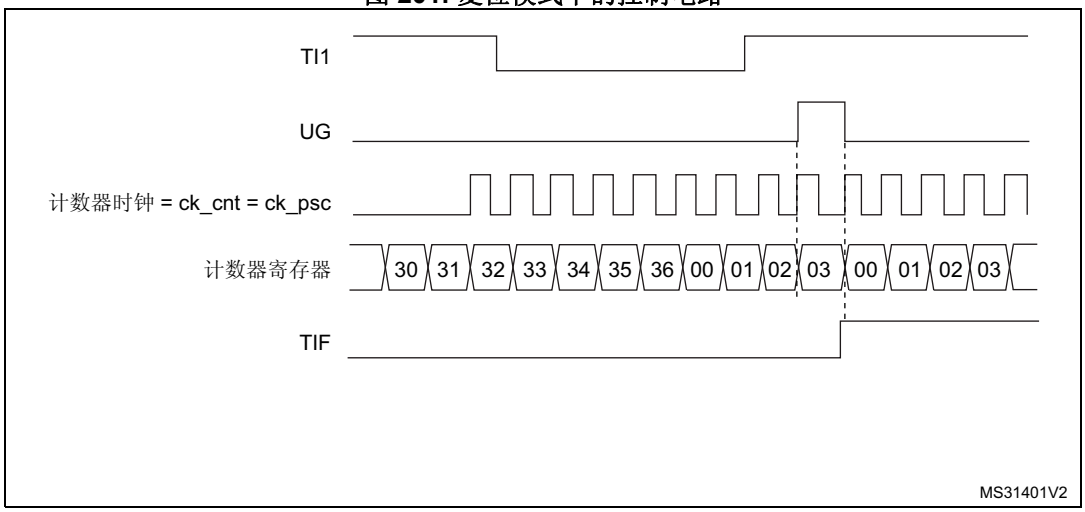
在以下示例中，TI1 输入上出现上升沿时，递增计数器清零：

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 201. 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。

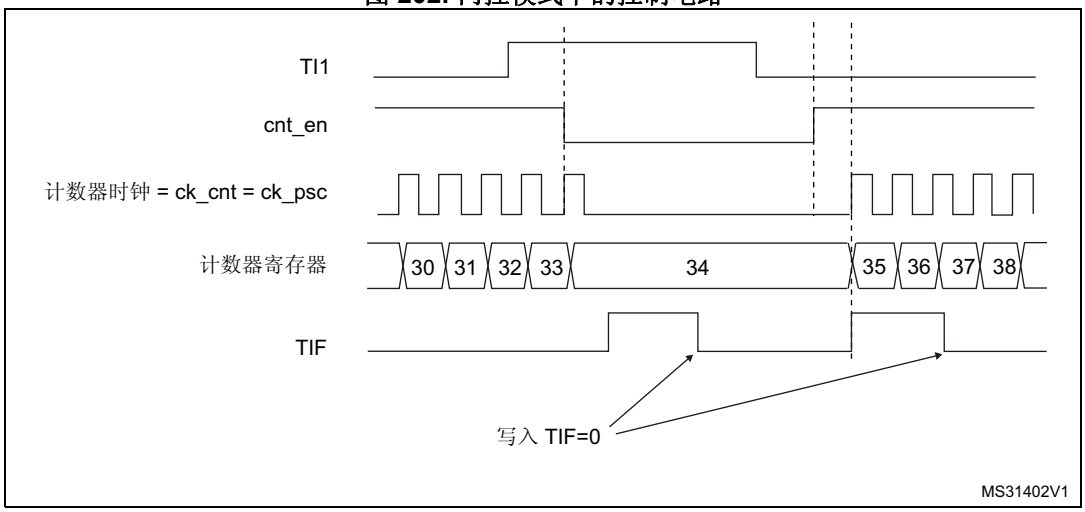
在以下示例中，递增计数器仅在 TI1 输入为低电平时计数：

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 202. 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。

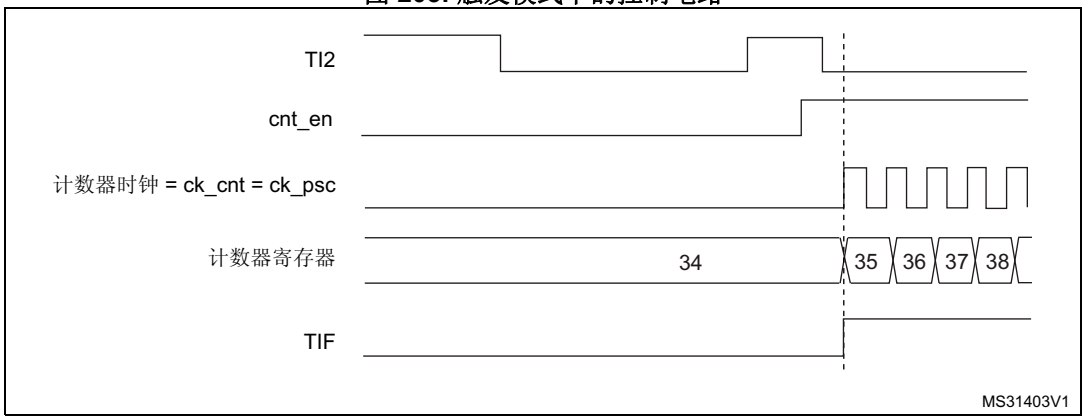
在以下示例中，TI2 输入上出现上升沿时，递增计数器启动：

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 203. 触发模式下的控制电路



19.3.12 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 18.3.15 节：定时器同步*。

19.3.13 调试模式

当微控制器进入调试模式（带 FPU 的 Cortex®-M4 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 34.16.2 节：对定时器、看门狗、bxCAN 和 I2C 的调试支持*。

19.4 TIM9 和 TIM12 寄存器

有关寄存器说明中使用的缩写，请参见*第 50 页的第 1.2 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）。

19.4.1 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)

此位域指示定时器时钟 (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。

19.4.2 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)

此位域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。

有关各定时器的 ITRx 含义的更多详细信息，请参见*表 109*。

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。

表 109. TIMx 内部触发连接

从 TIM	ITR0 （TS = “000”）	ITR1 （TS = “001”）	ITR2 （TS = “010”）	ITR3 （TS = “011”）
TIM9	TIM2	TIM3 或 LPTIM1(1)	TIM10_OC	TIM11_OC
TIM12	TIM4	TIM5	TIM13_OC	TIM14_OC

1. 通过 LPTIM1_OR 寄存器位 4 选择 TIM3 或 LPTIM1。默认情况下选择 TIM3。

19.4.3 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：禁止触发中断。

1：使能触发中断。

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。

19.4.4 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。

19.4.5 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。

19.4.6 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。该寄存器的所有其他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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC2 通道配置为输出
01：CC2 通道配置为输入，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）。其他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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出

01：CC1 通道配置为输入，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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道配置为输出

01：CC2 通道配置为输入，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)

此位域可定义 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=2　　1001：$f_{SAMPLING}=f_{DTS}/8$，N=8

0010：$f_{SAMPLING}=f_{CK\_INT}$，N=4　　1010：$f_{SAMPLING}=f_{DTS}/16$，N=5

0011：$f_{SAMPLING}=f_{CK\_INT}$，N=8　　1011：$f_{SAMPLING}=f_{DTS}/16$，N=6

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6　　1100：$f_{SAMPLING}=f_{DTS}/16$，N=8

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8　　1101：$f_{SAMPLING}=f_{DTS}/32$，N=5

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6　　1110：$f_{SAMPLING}=f_{DTS}/32$，N=6

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8　　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)

此位域定义 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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出

01：CC1 通道配置为输入，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。

19.4.7 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 互补输出极性 (Capture/Compare 2 complementary 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 110. 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。*

19.4.8 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)

19.4.9 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。

19.4.10 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 更新和行为的详细信息，请参见*第 19.3.1 节：时基单元*。

当自动重载值为空时，计数器不工作。

19.4.11 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。

19.4.12 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。

19.4.13 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 111. 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
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
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	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**	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	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

表 111. 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
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 54 页的第 2.2.2 节*。

19.5 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）。

19.5.1 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)

此位域指示定时器时钟 (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

19.5.2 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

19.5.3 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。

19.5.4 TIM10/11/13/14 事件产生寄存器 (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。

19.5.5 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。此寄存器的所有其他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）。其他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)

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出。

01：CC1 通道配置为输入，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)

此位域可定义 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)

此位域定义 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)

此位域定义通道方向（输入/输出）以及所使用的输入。
00：CC1 通道配置为输出
01：CC1 通道配置为输入，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。

19.5.6 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 112. 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。*

19.5.7 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)

19.5.8 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。

19.5.9 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 19.3.1 节：时基单元*。
当自动重载值为空时，计数器不工作。

19.5.10 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。

19.5.11 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。

00,01,11：TIM11 通道 1 连接到 GPIO（请参见数据手册中的复用功能映射表）。

10：HSE_RTC 时钟（由可编程预分频器分频的 HSE）连接到 TIM11_CH1 输入，用于测量用途。

19.5.12 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 113. 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
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.

表 113. 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
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 54 页的第 2.2.2 节*。

20 基本定时器 (TIM6/7)

20.1 简介

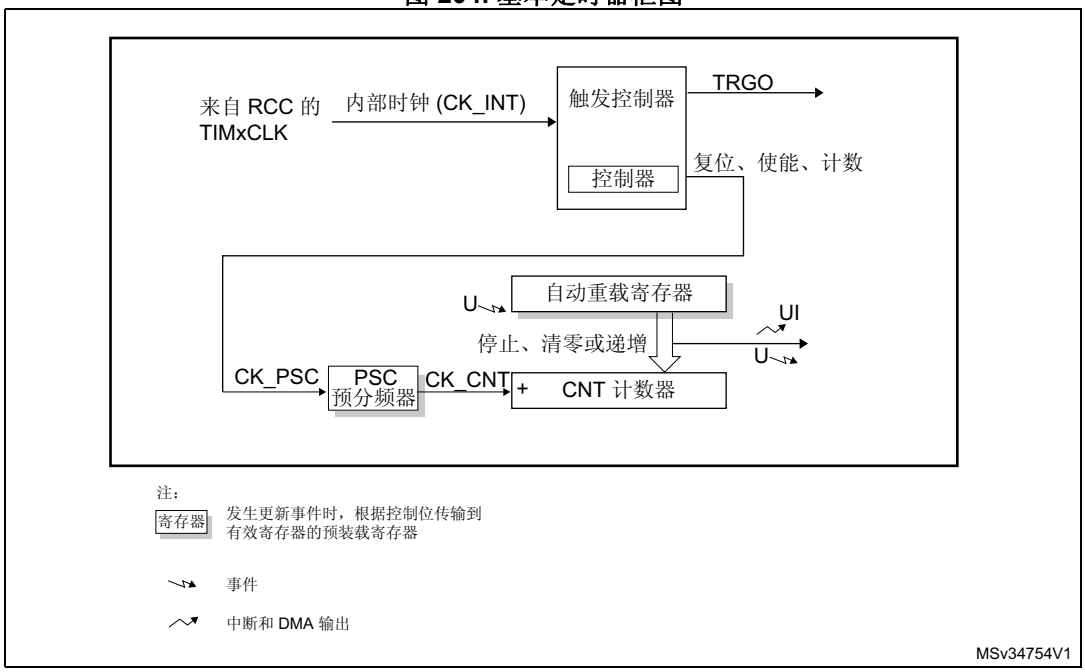
基本定时器 TIM6 和 TIM7 包含一个 16 位自动重载计数器，该计数器由可编程预分频器驱动。

20.2 TIM6/7 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（即运行时修改），分频系数介于 1 和 65536 之间
- 发生如下更新事件时会生成中断/DMA 请求：计数器上溢

图 204. 基本定时器框图



20.3 TIM6/7 功能说明

20.3.1 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立即传送到影子寄存器，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 16 位寄存器 TIMx_PSC 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 205 和 *图 206* 以一些示例说明在预分频比实时变化时计数器的行为。

图 205. 预分频器分频由 1 变为 2 时的计数器时序图

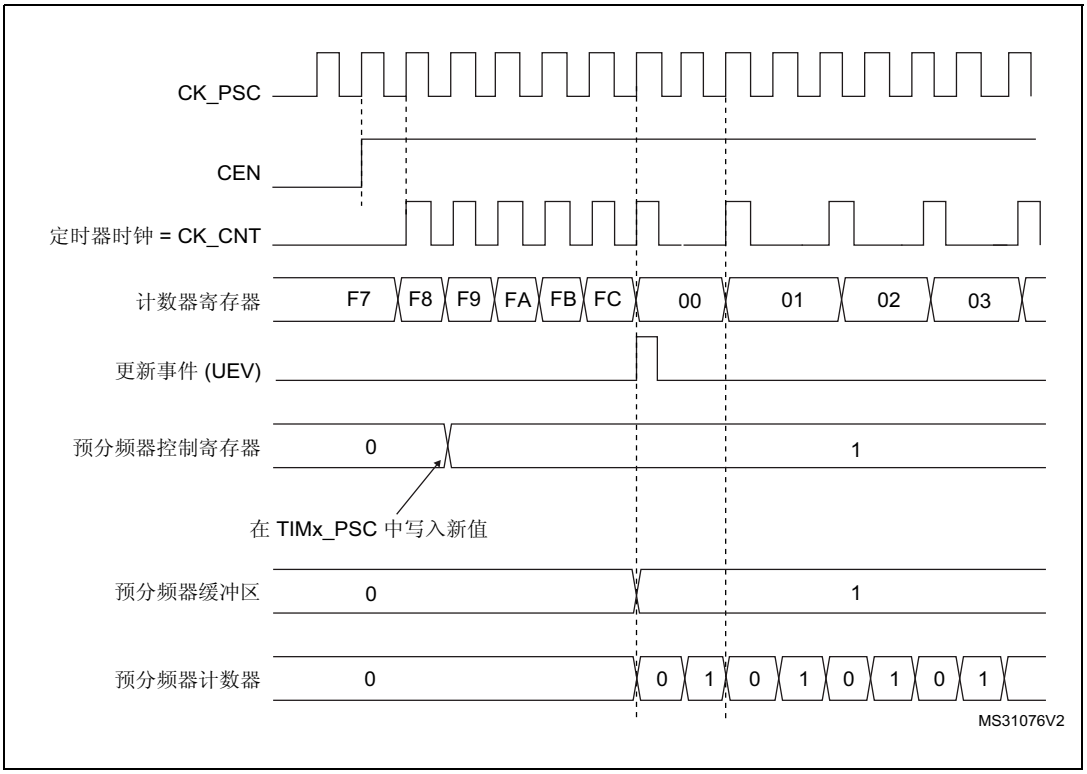
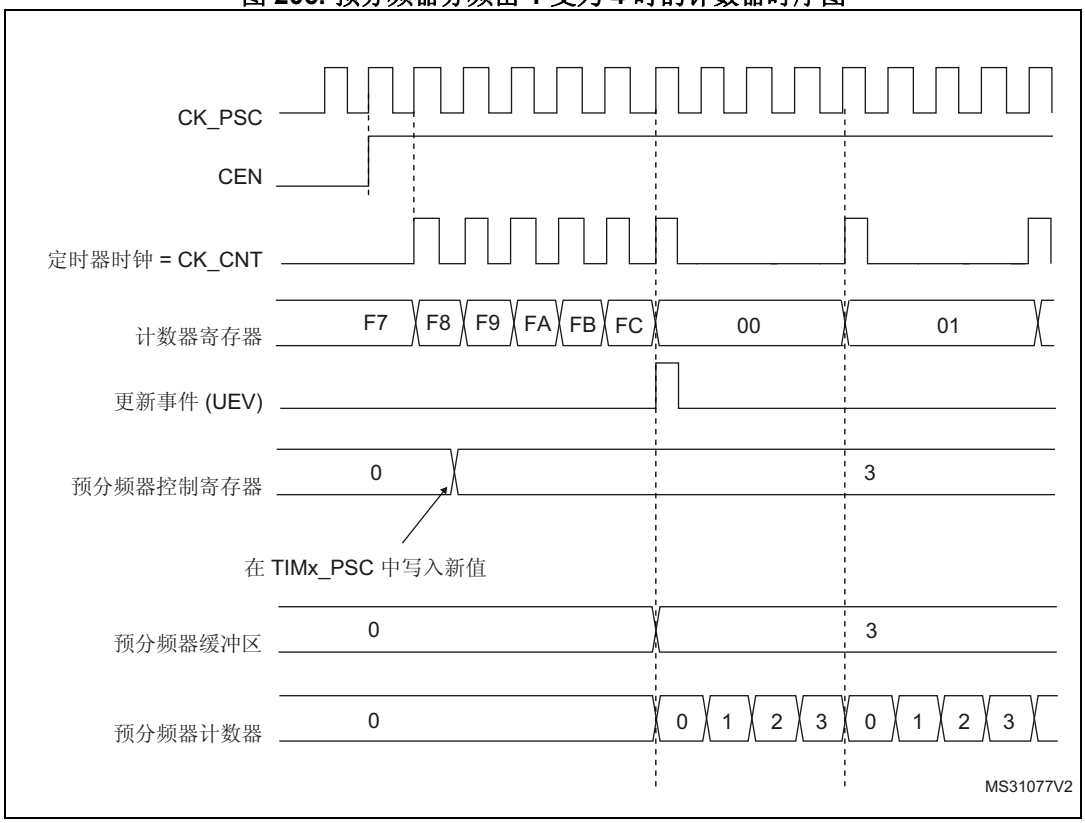


图 206. 预分频器分频由 1 变为 4 时的计数器时序图



20.3.2 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 207. 计数器时序图，1 分频内部时钟

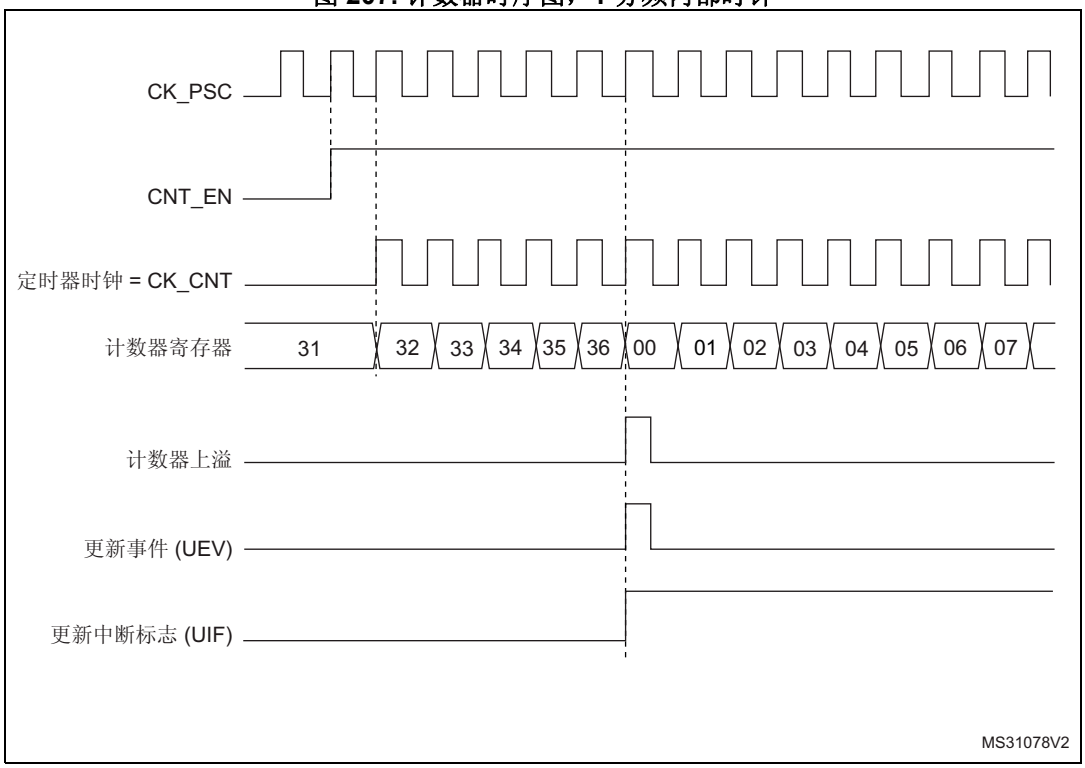


图 208. 计数器时序图，2 分频内部时钟

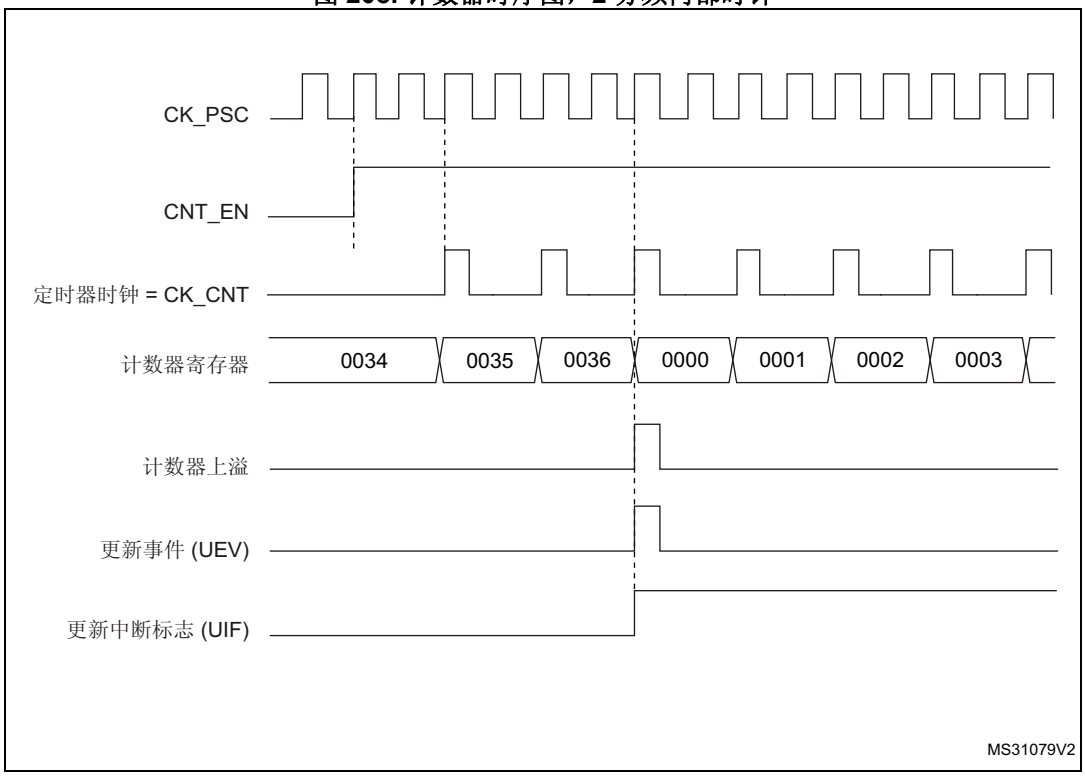


图 209. 计数器时序图，4 分频内部时钟

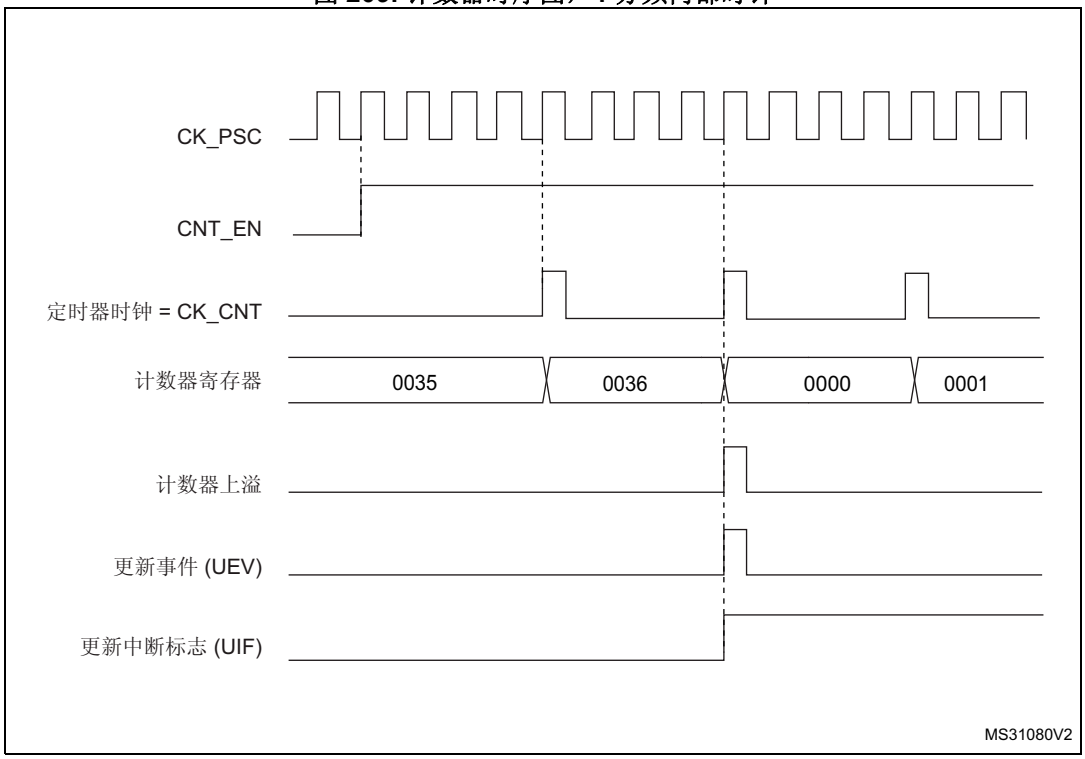


图 210. 计数器时序图，N 分频内部时钟

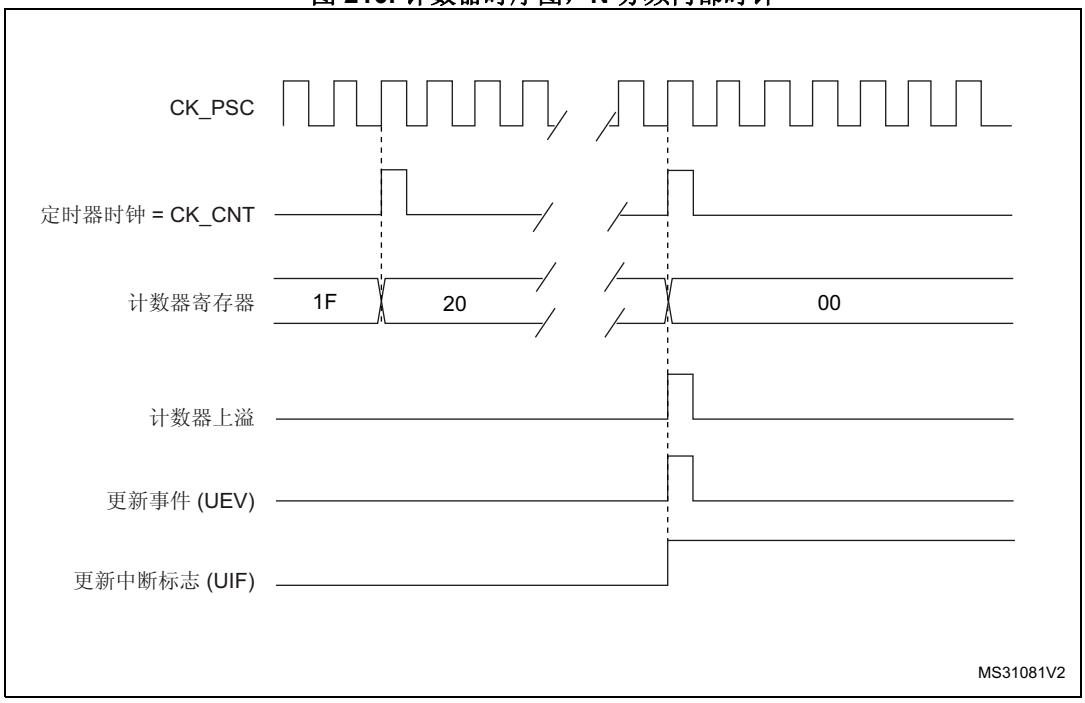


图 211. 计数器时序图，ARPE = 0 时更新事件（TIMx_ARR 未预装载）

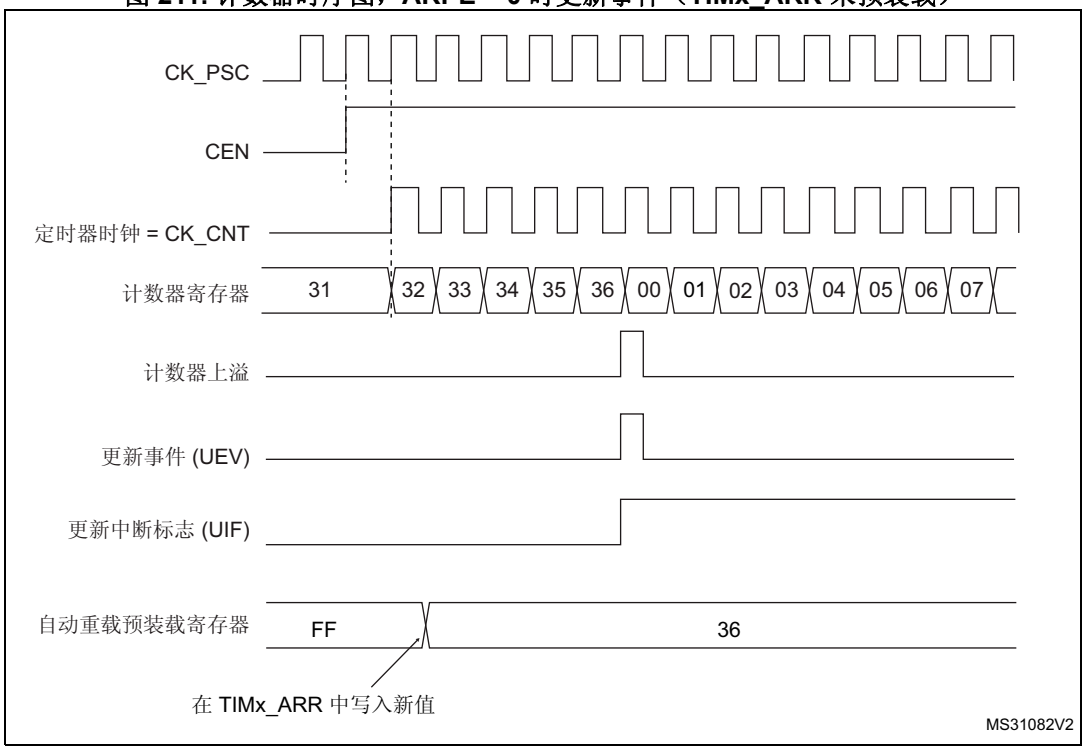
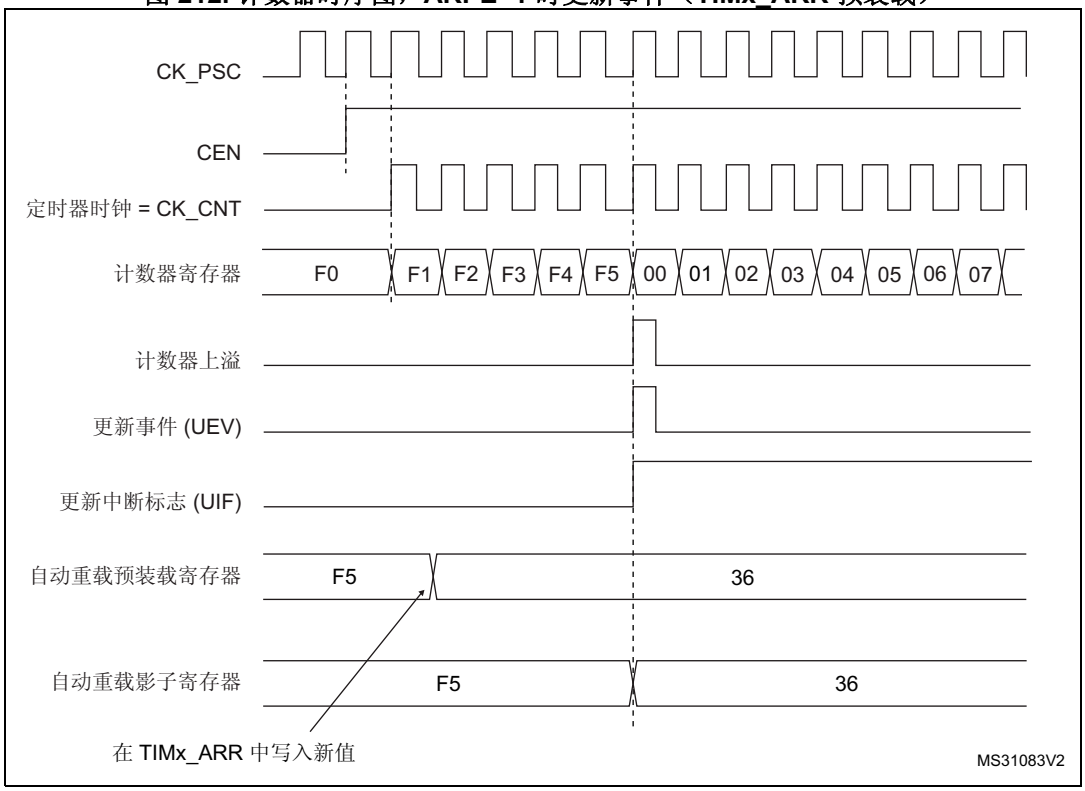


图 212. 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 预装载）



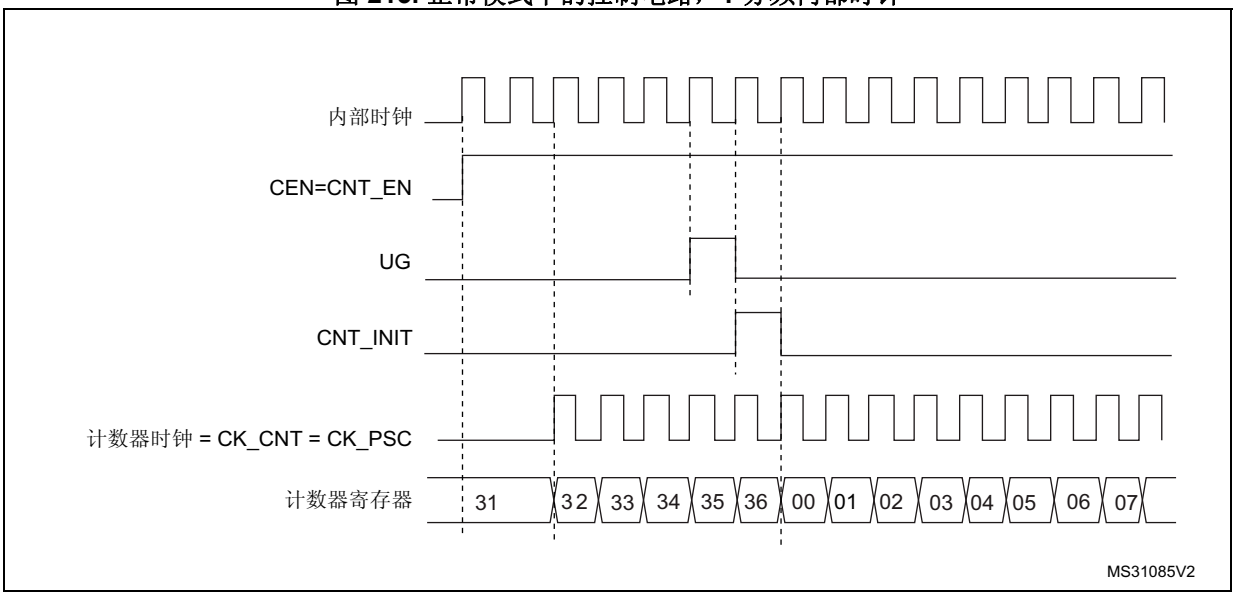
20.3.3 时钟源

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（除了 UG 位被自动清零外）。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 213 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 213. 正常模式下的控制电路，1 分频内部时钟



20.3.4 调试模式

当微控制器进入调试模式时（带 FPU 的 Cortex®-M4 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 34.16.2 节：对定时器、看门狗、bxCAN 和 I2C 的调试支持*。

20.4 TIM6/7 寄存器

有关寄存器说明中使用的缩写，请参见*第 1.2 节：寄存器相关缩写词列表*。

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）。

20.4.1 TIM6/7 控制寄存器 1 (TIMx_CR1)

TIM6/7 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.	Res.	Res.	ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
								rw				rw	rw	rw	rw

位 15:8 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

然后带缓冲的寄存器被加载为预加载数值。
1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR 和 PSC）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

位 0 **CEN**：计数器使能 (Counter enable)

0：禁止计数器
1：使能计数器

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。

20.4.2 TIM6/7 控制寄存器 2 (TIMx_CR2)

TIM6/7 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。

位 3:0 保留，必须保持复位值。

20.4.3 TIM6/7 DMA/中断使能寄存器 (TIMx_DIER)

TIM6/7 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。

20.4.4 TIM6/7 状态寄存器 (TIMx_SR)

TIM6/7 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。

– 如果 TIMx_CR1 寄存器中 URS = 0 且 UDIS = 0，通过软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。

20.4.5 TIM6/7 事件产生寄存器 (TIMx_EGR)

TIM6/7 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）。

20.4.6 TIM6/7 计数器 (TIMx_CNT)

TIM6/7 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)

20.4.7 TIM6/7 预分频器 (TIMx_PSC)

TIM6/7 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）时要装载到有效预分频器寄存器的值。

20.4.8 TIM6/7 自动重载寄存器 (TIMx_ARR)

TIM6/7 auto-reload register

偏移地址：0x2C

复位值：0xFFFF

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 608 页的第 20.3.1 节：时基单元*。

当自动重载值为空时，计数器不工作。

20.4.9 TIM6/7 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 114. TIM6/7 寄存器映射和复位值

偏移	寄存器	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.	ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
	Reset value									0				0	0	0	0
0x04	**TIMx_CR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MMS[2:0]			Res.	Res.	Res.	Res.
	Reset value										0	0	0				
0x08	Res.																
0x0C	**TIMx_DIER**	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.	UIF
	Reset value																0
0x14	**TIMx_EGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UG
	Reset value																0
0x18	Res.																
0x1C	Res.																
0x20	Res.																
0x24	**TIMx_CNT**	CNT[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	**TIMx_PSC**	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[15:0]															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

有关寄存器边界地址的信息，请参见*第 54 页的第 2.2.2 节*。

21 低功耗定时器 (LPTIM)

21.1 简介

LPTIM 是一个 16 位定时器，可从降低功耗的最终发展中受益。由于 LPTIM 的时钟源具有多样性，因此 LPTIM 能够在所有电源模式（待机模式除外）下保持运行状态。即使没有内部时钟源，LPTIM 也能运行，鉴于这一点，可将其用作“脉冲计数器”，这种脉冲计数器在某些应用中十分有用。此外，LPTIM 还能将系统从低功耗模式唤醒，因此非常适合实现“超时功能”，而且功耗极低。

LPTIM 引入了一个灵活的时钟方案，该方案能够提供所需的功能和性能，同时还能最大程度地降低功耗。

21.2 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
  - LPTIM 输入的外部时钟源（在没有 LP 振荡器运行的情况下工作，由脉冲计数器应用使用）
- 16 位 ARR 自动重载寄存器
- 16 位比较寄存器
- 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

21.3 LPTIM 实现

表 115 介绍了 STM32F413/423 器件上的 LPTIM 实现。

表 115. STM32F413/423 LPTIM 特性

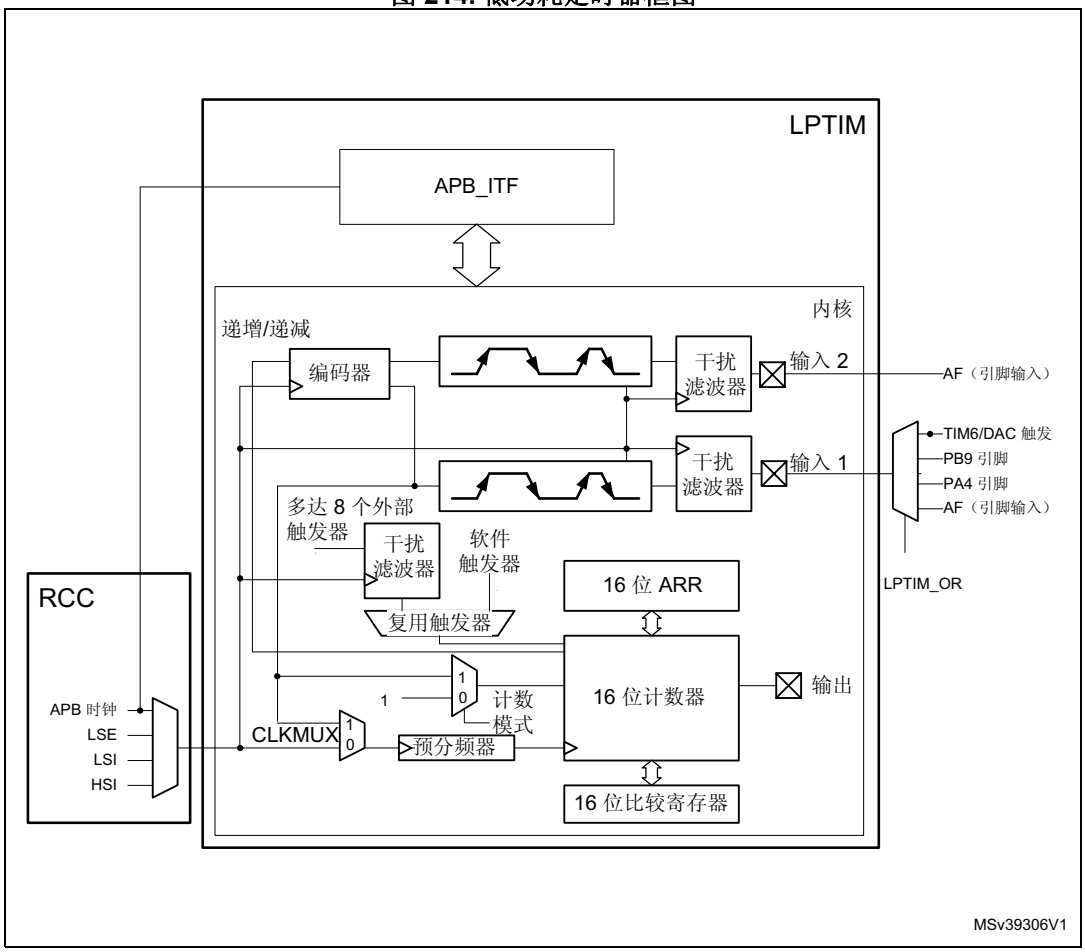
LPTIM 模式/特性(1)	LPTIM1
编码器模式	X

1. X = 支持。

21.4 LPTIM 功能说明

21.4.1 LPTIM 框图

图 214. 低功耗定时器框图



21.4.2 LPTIM 触发映射

下面详细介绍了 LPTIM 外部触发连接：

表 116. LPTIM1 外部触发连接

TRIGSEL	外部触发信号
lptim_ext_trig0	AF1 上的 PB6 或 PC3 输入
lptim_ext_trig1	RTC 闹钟 A 输出信号
lptim_ext_trig2	RTC 闹钟 B 输出信号
lptim_ext_trig3	RTC 入侵输出信号
lptim_ext_trig4	TIM1 触发输出 (4) 输出信号
lptim_ext_trig5	TIM5 触发输出 (3) 输出信号

表 116. LPTIM1 外部触发连接（续）

TRIGSEL	外部触发信号
lptim_ext_trig6	Reserved
lptim_ext_trig7	Reserved

21.4.3 LPTIM input1 复用

可以通过 LPTMI 选项寄存器 (LPTIM1_OR) 为 LPTIM1 输入 1 选择各种输入。

该输入可以连接到由 LPTIM 复用功能 (AF1) 选择的焊盘，也可以直接从内部连接到 PA4、PB9 焊盘或 TIM6/DAC 触发信号。

从内部连接到 PA4 或 PB9 时，为此焊盘选定的复用功能会指定连接定时器的外设。

PA4 和 PB9 也可以配置为 GPIO。

21.4.4 LPTIM 复位和时钟

LPTIM 可通过多个时钟源提供时钟。它可以由内部时钟信号提供时钟，内部时钟信号可通过复位和时钟控制器 (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。

21.4.5 干扰滤波器

外部（映射到微控制器 GPIO）或内部（映射到芯片级或其它嵌入式外设，例如嵌入式比较器）LPTIM 输入由数字滤波器保护，避免任何毛刺和噪声干扰在 LPTIM 内部传播，从而防止产生意外计数或触发。

在激活数字滤波器之前，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。

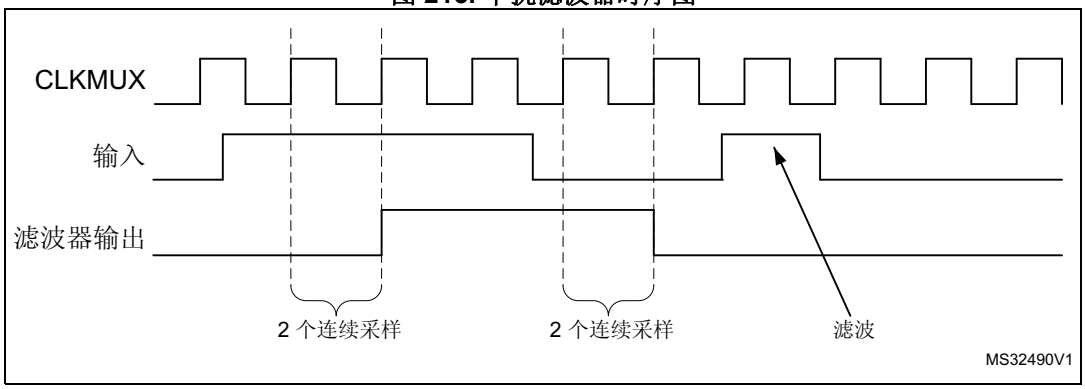
数字滤波器分为两组：

- 第一组数字滤波器保护 LPTIM 外部输入。数字滤波器的灵敏度由 CKFLT 位控制。
- 第二组数字滤波器保护 LPTIM 内部触发输入。数字滤波器的灵敏度由 TRGFLT 位控制。

注： *数字滤波器的灵敏度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灵敏度。*

滤波器的灵敏度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 215* 给出了编程 2 个连续采样时，干扰滤波器行为的示例。

图 215. 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。*

21.4.6 预分频器

LPTIM 16 位计数器前面要有一个可配置的 2 次幂预分频器。预分频器的分频比由 PRESC[2:0] 3 位域进行控制。下表列出了所有可能的分频比：

表 117. 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

21.4.7 触发多路复用器

LPTIM 计数器可通过软件启动，也可以在 8 个触发输入之一上检测到有效边沿后启动。

TRIGEN[1:0] 用于确定 LPTIM 触发源：

- TRIGEN[1:0] 等于“00”时，LPTIM 计数器会在通过软件将 CNTSTRT 位或 SNGSTRT 位其中之一置 1 后立即启动。TRIGEN[1:0] 的其余三个可能的值用于配置触发输入使用的有效边沿。LPTIM 计数器会在检测到有效边沿后立即启动。
- TRIGEN[1:0] 不等于“00”时，TRIGSEL[2:0] 用于选择使用 8 个触发输入中的哪一个来启动计数器。

外部触发信号视为 LPTIM 的异步信号。因此，检测到触发信号后，由于同步问题，需要延迟两个计数器时钟周期，定时器才能开始运行。

如果在定时器已启动时发生新的触发事件，则此事件将被忽略（除非已使能超时功能）。

注： *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。*

21.4.8 工作模式

LPTIM 支持以下两种工作模式：

- 连续模式：定时器自由运行，由触发事件启动并且直到被禁止才会停止
- 单触发模式：定时器由触发事件启动，当达到 ARR 值时停止。

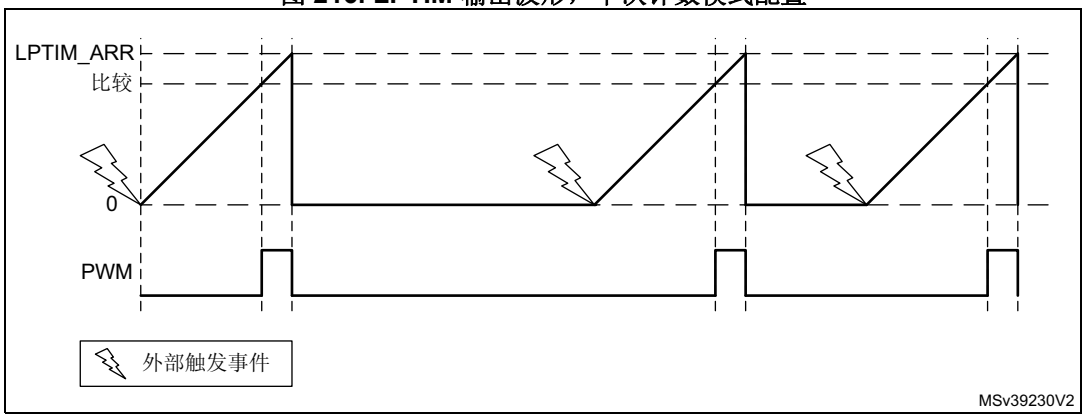
单触发模式

要使能单触发计数，必须将 SNGSTRT 位置 1。

新的触发事件将重新启动定时器。从计数器启动到计数器达到 ARR，这段时间内发生的任何触发事件均将被丢弃。

选择外部触发时，在 SNGSTRT 位置 1 后以及计数器寄存器停止后（包含零值）到达的每个外部触发事件都将为计数器启动新的单触发计数周期，如*图 216* 所示。

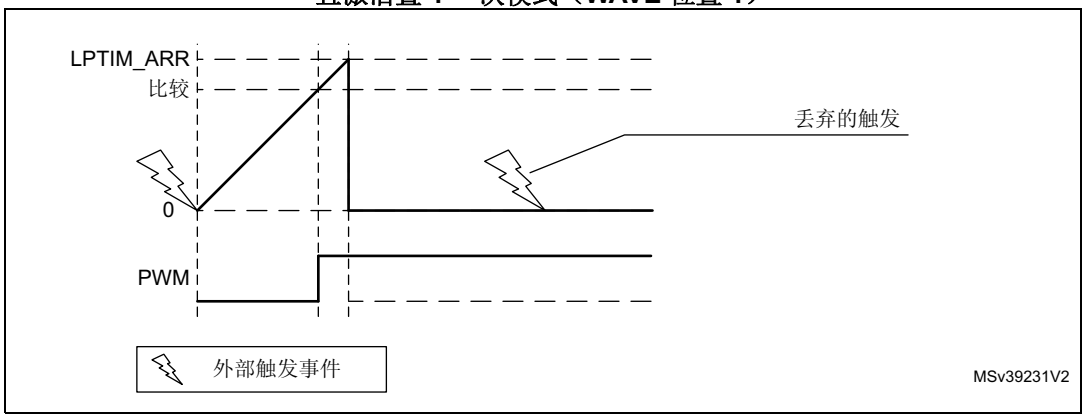
图 216. LPTIM 输出波形，单次计数模式配置



- 置 1 一次模式激活：

应注意，当 LPTIM_CFGR 寄存器中的 WAVE 位域置 1 时，将激活置 1 一次模式。在这种情况下，计数器仅会在第一个触发事件后启动一次，任何后续触发事件都将被丢弃，如*图 217* 所示。

图 217. LPTIM 输出波形，单次计数模式配置且激活置 1 一次模式（WAVE 位置 1）



若通过软件启动（TRIGEN[1:0] = “00”），将 SNGSTRT 置 1 会使计数器进行单触发计数。

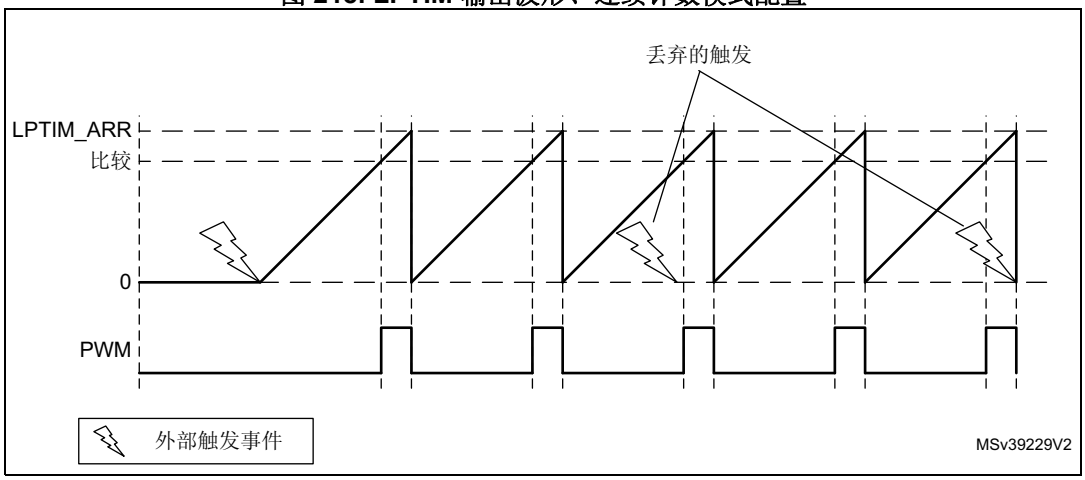
连续模式

要使能连续计数，必须将 CNTSTRT 位置 1。

若选择外部触发，则在 CNTSTRT 置 1 后到达的外部触发事件将启动计数器进行连续计数。任何后续的外部触发事件都将被丢弃，如*图 218* 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 218. LPTIM 输出波形、连续计数模式配置



SNGSTRT 和 CNTSTRT 位只能在定时器使能时（ENABLE 位置 1）置 1。可以“实时的”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。

21.4.9 超时功能

若在一个选定的触发输入上检测到有效边沿，则可用于复位 LPTIM 计数器。该功能通过 TIMOUT 位进行控制。

第一个触发事件将启动定时器，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。

21.4.10 生成波形

两个 16 位寄存器，LPTIM_ARR（自动重载寄存器）和 LPTIM_CMP（比较寄存器）用于在 LPTIM 输出上生成多个不同的波形。

定时器可生成以下波形：

- PWM 模式：若 LPTIM_CMP 寄存器与 LPTIM_CNT 寄存器匹配，则会立即将 LPTIM 输出置 1。若 LPTIM_ARR 寄存器与 LPTIM_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 LPTIM_ARR 寄存器的值严格大于 LPTIM_CMP 寄存器的值。

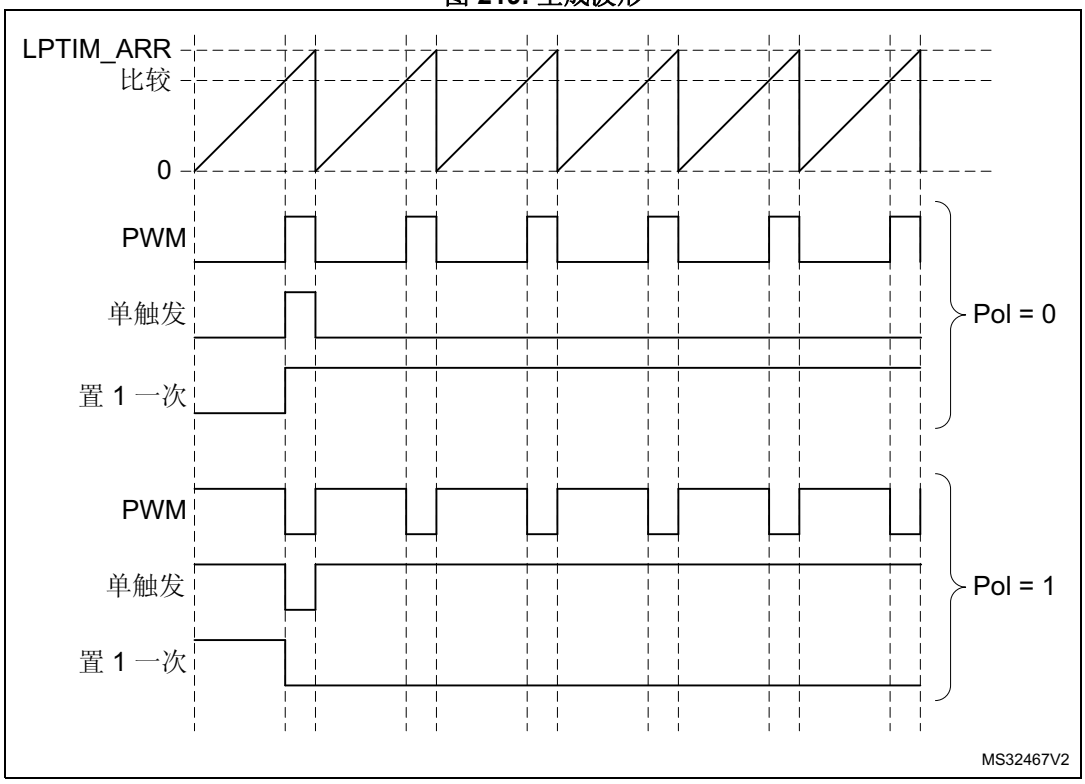
LPTIM 输出波形可通过 WAVE 位配置，具体如下：

- 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 一次模式波形。

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 219* 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 219. 生成波形



21.4.11 寄存器更新

LPTIM_ARR 寄存器和 LPTIM_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 LPTIM_ARR 寄存器和 LPTIM_CMP 寄存器的更新方式：

- 当 PRELOAD 位复位为 0 时，LPTIM_ARR 寄存器和 LPTIM_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，LPTIM_ARR 寄存器和 LPTIM_CMP 寄存器会在当前周期结束时更新。

LPTIM APB 接口和 LPTIM 内核逻辑使用的时钟不同，因此在 APB 写操作后，需要经过一定的延迟，写入值才能用于计数器比较器。在此延迟期间，必须避免向这些寄存器执行其他写操作。

LPTIM_ISR 寄存器中的 ARROK 标志和 CMPOK 标志分别指示 LPTIM_ARR 寄存器和 LPTIM_CMP 寄存器的写操作已完成。

向 LPTIM_ARR 寄存器和 LPTIM_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。

21.4.12 计数器模式

LPTIM 计数器可用于对 LPTIM Input1 上的外部事件进行计数，也可用于对内部时钟周期进行计数。CKSEL 位和 COUNTMODE 位用于控制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

    LPTIM 配置为由内部时钟源提供时钟，LPTIM 计数器配置为在每个内部时钟脉冲后进行更新。
  - COUNTMODE = 1

    LPTIM 外部 Input1 通过提供给 LPTIM 的内部时钟进行采样。

    因此，为了不丢失任何事件，外部 Input1 信号变化的频率决不应超过提供给 LPTIM 的内部时钟的频率。此外，不得对提供给 LPTIM 的内部时钟进行预分频 (PRESC[2:0] = 000)。
- CKSEL = 1：LPTIM 由外部时钟源提供时钟

  COUNTMODE 值不相关。

  在这种配置下，LPTIM 无需内部时钟源（已使能干扰滤波器时除外）。注入到 LPTIM 外部 Input1 的信号用作 LPTIM 的系统时钟。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 的信号也可用于为 LPTIM 内核逻辑提供时钟，计数器递增计数前存在一些初始延时（使能 LPTIM 后）。更确切地说，LPTIM 外部 Input1 的前五个有效边沿将丢失（使能 PTIM 后）。

21.4.13 定时器使能

LPTIM_CR 寄存器中的 ENABLE 位用于使能/禁止 LPTIM 内核逻辑。将 ENABLE 位置 1 后，需要延迟两个计数器时钟周期，才能真正使能 LPTIM。

LPTIM_CFGR 和 LPTIM_IER 寄存器必须在禁止 LPTIM 后才能修改。

21.4.14 编码器模式

此模式用于处理来自正交编码器的信号，此正交编码器用于检测旋转元件的角度位置。编码器接口模式就相当于带有方向选择的外部时钟。这意味着，计数器仅在 0 到 LPTIM_ARR 寄存器中编程的自动重载值之间进行连续计数（根据具体方向，从 0 递增计数到 ARR，或从 ARR 递减计数到 0）。因此，在启动前必须先配置 LPTIM_ARR。通过两个外部输入信号 Input1 和 Input2 生成时钟信号作为 LPTIM 计数器时钟。这两个信号间的相位确定计数方向。

仅当 LPTIM 由内部时钟源提供时钟时才可使用编码器模式。Input1 和 Input2 输入上的信号频率不得超过 LPTIM 内部时钟频率 4 分频。必须满足此条件才能确保 LPTIM 正常工作。

方向变化由 LPTIM_ISR 寄存器中的两个递减和递增标志指示。此外，如果通过 DOWNIE 位使能，还可为两种方向变化事件产生中断。

要激活编码器模式，必须将 ENC 位置 1。LPTIM 必须首先配置为连续模式。

当编码器模式激活时，LPTIM 计数器按照增量编码器的速度和方向自动修改。因此，其内容始终代表编码器的位置。计数方向由递增和递减标志指示，对应于编码器转子的旋转方向。

根据使用 CKPOL[1:0] 位配置的边沿敏感性，可得几种不同的计数方案。下表汇总了可能的组合（假设 Input1 和 Input2 不同时切换）。

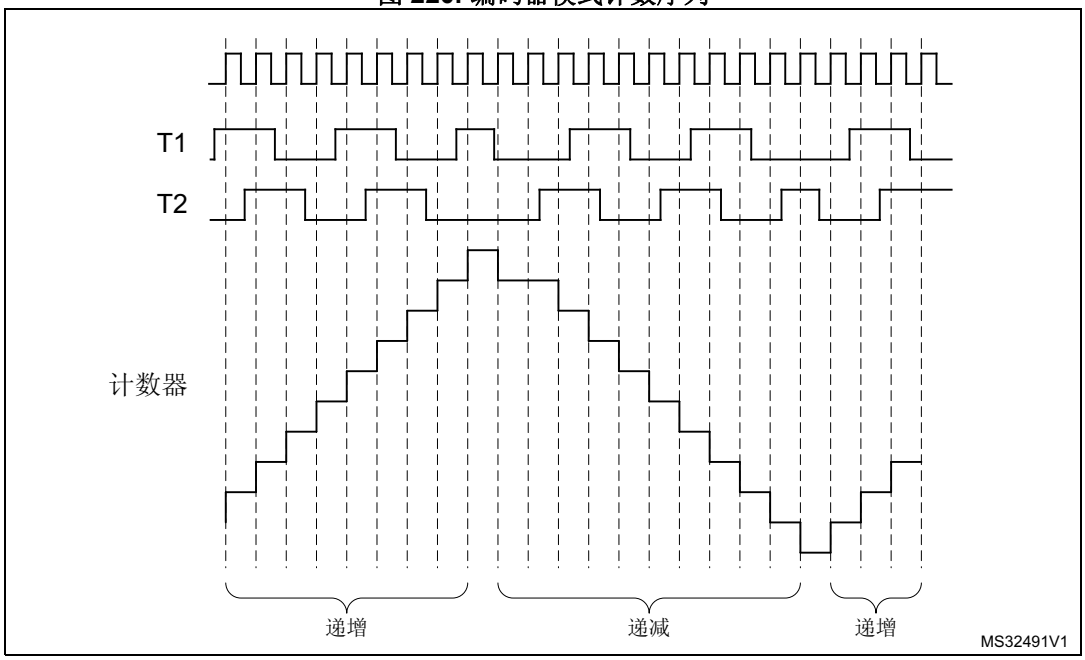
表 118. 编码器计数方案

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。

注意： 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 220. 编码器模式计数序列



21.4.15 调试模式

当微控制器进入调试模式时（内核停止），LPTIM 计数器会根据 DBG 模块中的 DBG_LPTIM_STOP 配置位选择继续正常工作或者停止工作。

21.5 LPTIM 中断

若以下事件通过 LPTIM_IER 寄存器使能，则这些事件会生成中断/唤醒事件：

- 与设定的数值比较后匹配
- 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 LPTIM_IER 寄存器（中断使能寄存器）中的位在 LPTIM_ISR 寄存器（状态寄存器）中相应标志置 1 后置 1，就不会触发中断。*

表 119. 中断事件

中断事件	说明
与设定的数值比较后匹配	当计数器寄存器 (LPTIM_CNT) 的内容与比较寄存器 (LPTIM_CMP) 的内容匹配时，生成中断标志。
自动重载匹配	当计数器寄存器 (LPTIM_CNT) 的内容与自动重新加载寄存器 (LPTIM_ARR) 的内容匹配时，生成中断标志。
外部触发事件	当检测到外部触发事件时，会生成中断标志。
自动重载寄存器更新成功	当对 LPTIM_ARR 寄存器的写操作完成时，生成中断标志。
比较寄存器更新成功	当对 LPTIM_CMP 寄存器的写操作完成时，生成中断标志。
方向更改	用于编码器模式。嵌入两个中断标志以发出方向更改的信号： – UP 标志发出递增计数方向更改的信号。 – DOWN 标志发出递减计数方向更改的信号。

21.6 LPTIM 寄存器

21.6.1 LPTIM 中断和状态寄存器 (LPTIM_ISR)

LPTIM interrupt and status register

偏移地址：0x000

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWN	UP	ARROK	CMPOK	EXTTRIG	ARRM	CMPM
									r	r	r	r	r	r	r

位 31:22 保留，必须保持复位值。

位 21 保留，必须保持复位值。

位 20 保留，必须保持复位值。

位 19 保留，必须保持复位值。

位 18:16 保留，必须保持复位值。

位 15 保留，必须保持复位值。

位 14 保留，必须保持复位值。

位 13 保留，必须保持复位值。

位 12 保留，必须保持复位值。

位 11 保留，必须保持复位值。

位 10 保留，必须保持复位值。

位 9 保留，必须保持复位值。

位 8:7 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 LPTIM_ARR 寄存器的 APB 总线写操作已成功完成。

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 LPTIM_CMP 寄存器的 APB 总线写操作已成功完成。

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 LPTIM_CNT 寄存器的值已达到 LPTIM_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 LPTIM_CNT 寄存器的值已达到 LPTIM_CMP 寄存器的值。

21.6.2 LPTIM 中断清零寄存器 (LPTIM_ICR)

LPTIM interrupt clear register

偏移地址：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWNCF	UPCF	ARROKCF	CMPOKCF	EXTTRIGCF	ARRMCF	CMPMCF
									w	w	w	w	w	w	w

位 31:22 保留，必须保持复位值。

位 21 保留，必须保持复位值。

位 20 保留，必须保持复位值。

位 19 保留，必须保持复位值。

位 18:16 保留，必须保持复位值。

位 15 保留，必须保持复位值。

位 14 保留，必须保持复位值。

位 13 保留，必须保持复位值。

位 12 保留，必须保持复位值。

位 11 保留，必须保持复位值。

位 10 保留，必须保持复位值。

位 9 保留，必须保持复位值。

位 8:7 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，LPTIM_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，LPTIM_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，LPTIM_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，LPTIM_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，LPTIM_ISR 寄存器中的 EXTTRIG 标志将清零

位 1 **ARRMCF**：自动重载匹配清零标志 (Autoreload match clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 ARRM 标志将清零

位 0 **CMPMCF**：比较匹配清零标志 (Compare match clear flag)
将 1 写入此位时，LPTIM_ISR 寄存器中的 CMP 标志将清零

21.6.3 LPTIM 中断使能寄存器 (LPTIM_IER)

LPTIM interrupt enable register

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWNIE	UPIE	ARROKIE	CMPOKIE	EXTTRIGIE	ARRMIE	CMPMIE
									rw	rw	rw	rw	rw	rw	rw

位 31:28 保留，必须保持复位值。

位 27 保留，必须保持复位值。

位 26 保留，必须保持复位值。

位 25 保留，必须保持复位值。

位 24 保留，必须保持复位值。

位 23 保留，必须保持复位值。

位 22 保留，必须保持复位值。

位 21 保留，必须保持复位值。

位 20 保留，必须保持复位值。

位 19 保留，必须保持复位值。

位 18:17 保留，必须保持复位值。

位 16 保留，必须保持复位值。

位 15 保留，必须保持复位值。

位 14 保留，必须保持复位值。

位 13 保留，必须保持复位值。

位 12 保留，必须保持复位值。

位 11 保留，必须保持复位值。

位 10 保留，必须保持复位值。

位 9 保留，必须保持复位值。

位 8:7 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

注意： 必须在 LPTIM 已禁止时（ENABLE 位复位为 0）才能修改 LPTIM_IER 寄存器

21.6.4 LPTIM 配置寄存器 (LPTIM_CFGR)

LPTIM configuration register

偏移地址：0x00C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENC	COUNT MODE	PRELOAD	WAVPOL	WAVE	TIMOUT	TRIGEN[1:0]		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[2:0]			Res.	PRESC[2:0]			Res.	TRGFLT[1:0]		Res.	CKFLT[1:0]		CKPOL[1:0]		CKSEL
rw	rw	rw		rw	rw	rw		rw	rw		rw	rw	rw	rw	rw

位 31:30 保留，必须保持复位值。

位 29 保留，必须保持复位值。

位 28:25 保留，必须保持复位值。

位 24 **ENC**：编码器模式使能 (Encoder mode enable)

ENC 位控制编码器模式

0：禁止编码器模式

1：使能编码器模式

位 23 **COUNTMODE**：计数器模式使能 (counter mode enabled)

COUNTMODE 位用于选择 LPTIM 使用哪个时钟源来为计数器提供时钟：

0：计数器在每个内部时钟脉冲后递增

1：计数器在 LPTIM 外部 Input1 上的每个有效时钟脉冲后递增

位 22 **PRELOAD**：寄存器更新模式 (Registers update mode)

PRELOAD 位控制 LPTIM_ARR 寄存器和 LPTIM_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 LPTIM_ARR 寄存器和 LPTIM_CMP 寄存器之间的比较结果。

1：LPTIM 输出反映与 LPTIM_ARR 寄存器和 LPTIM_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： 停用置 1 一次模式和 PWM/单脉冲波形（具体取决于 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[1:0]**：触发使能和极性 (Trigger enable and polarity)

TRIGEN 位控制 LPTIM 计数器是否由外部触发信号启动。如果已选择由外部触发信号启动，触发有效边沿的配置有以下三种：

00：软件触发（由软件启动计数）

01：上升沿为有效边沿

10：下降沿为有效边沿

11：上升沿和下降沿均为有效边沿

位 16 保留，必须保持复位值。

位 15:13 **TRIGSEL[2:0]**：触发源选择器 (Trigger selection)

TRIGSEL 位用于选择作为 LPTIM 触发事件的触发源，可用触发源包括以下 8 种：

000：lptim_ext_trig0

001：lptim_ext_trig1

010：lptim_ext_trig2

011：lptim_ext_trig3

100：lptim_ext_trig4

101：lptim_ext_trig5

110：lptim_ext_trig6

111：lptim_ext_trig7

详细信息，请参见*第 21.4.2 节：LPTIM 触发映射*。

位 12 保留，必须保持复位值。

位 11:9 **PRESC[2:0]**：时钟预分频器 (Clock prescaler)

PRESC 位配置预分频器的分频系数。分频系数可从以下分频系数中选择：

000： /1

001： /2

010： /4

011： /8

100： /16

101： /32

110： /64

111： /128

位 8 保留，必须保持复位值。

位 7:6 **TRGFLT[1:0]**：触发信号的可配置数字滤波器 (Configurable digital filter for trigger)

TRGFLT 值用于设置连续相同采样的数量，若在内部触发信号电平发生变化时检测到此类连续采样，才会将此电平变化视为有效电平切换。必须存在内部时钟源才能使用此功能

00：任何触发信号有效电平变化均视为有效触发

01：触发信号有效电平变化必须至少稳定 2 个时钟周期，才能将其视为有效触发。

10：触发信号有效电平变化必须至少稳定 4 个时钟周期，才能将其视为有效触发。

11：触发信号有效电平变化必须至少稳定 8 个时钟周期，才能将其视为有效触发。

位 5 保留，必须保持复位值。

位 4:3 **CKFLT[1:0]**：外部时钟的可配置数字滤波器 (Configurable digital filter for external clock)

CKFLT 值用于设置连续相同采样的数量，若在外部时钟信号电平发生变化时检测到此类连续采样，才会将此电平变化视为有效电平切换。必须存在内部时钟源才能使用此功能

00：任何外部时钟信号电平变化均视为有效切换

01：外部时钟信号电平变化必须至少稳定 2 个时钟周期，才能将其视为有效切换。

10：外部时钟信号电平变化必须至少稳定 4 个时钟周期，才能将其视为有效切换。

11：外部时钟信号电平变化必须至少稳定 8 个时钟周期，才能将其视为有效切换。

位 2:1 **CKPOL[1:0]**：时钟极性 (Clock Polarity)

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 21.4.14 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

注意： 必须在 LPTIM 已禁止时（ENABLE 位复位为 0）才能修改 LPTIM_CFGR 寄存器。

21.6.5 LPTIM 控制寄存器 (LPTIM_CR)

LPTIM control register

偏移地址：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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 LPTIM_ARR 寄存器和 LPTIM_CNT 寄存器下一次匹配时定时器不会停止，LPTIM 计数器将继续以连续模式计数。

只有在使能 LPTIM 时才能将此位置 1。此位由硬件自动复位。

位 1 **SNGSTRT**：LPTIM 以单脉冲模式启动 (LPTIM start in Single mode)

此位通过软件置 1，通过硬件清零。

若通过软件启动（TRIGEN[1:0] = “00”），将此位置 1 会使 LPTIM 以单脉冲模式启动。

如果禁止软件启动（TRIGEN[1:0] 不等于“00”），将此位置 1 会使 LPTIM 在检测到外部触发信号后立即以单脉冲模式启动。

如果在 LPTIM 处于连续计数模式时将此位置 1，LPTIM 将在 LPTIM_ARR 寄存器和 LPTIM_CNT 寄存器下一次匹配时停止。

只有在使能 LPTIM 时才能将此位置 1。此位由硬件自动复位。

位 0 **ENABLE**：LPTIM 使能 (LPTIM enable)

ENABLE 位由软件置 1 和清零。

0：禁止 LPTIM

1：使能 LPTIM

21.6.6 LPTIM 比较寄存器 (LPTIM_CMP)

LPTIM compare register

偏移地址：0x014

复位值：0x0000 0000

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

CMP 为 LPTIM 所使用的比较值。

注意： 必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIM_CMP 寄存器。

21.6.7 LPTIM 自动重载寄存器 (LPTIM_ARR)

LPTIM autoreload register

偏移地址：0x018

复位值：0x0000 0001

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

位 31:16 保留，必须保持复位值。

位 15:0 **ARR[15:0]**：自动重载值 (Auto reload value)
ARR 为 LPTIM 的自动重载值。
此值必须严格大于 CMP[15:0] 的值。

注意： 必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIM_ARR 寄存器。

21.6.8 LPTIM 计数器寄存器 (LPTIM_CNT)

LPTIM counter register

偏移地址：0x01C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 保留，必须保持复位值。

位 15:0 **CNT[15:0]**：计数器值 (Counter value)
当 LPTIM 通过异步时钟运行时，读取 LPTIM_CNT 寄存器会返回不可靠的值。因此在这种情况下，必须连续执行读访问两次，并验证两次返回的值是否相同。
应注意的是，为了实现可靠的 LPTIM_CNT 寄存器读访问，必须执行两次连续的读访问并进行比较。当两次连续读访问的值相等时，可认为读访问可靠。

21.6.9 LPTIM1 选项寄存器 (LPTIM1_OPTR)

LPTIM1 option register

偏移地址：0x020

复位值：0x0000 0000

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

15	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.	TIM9_ITR1_RMP	TIM5_ITR1_RMP	TIM1_ITR2_RMP	LPT_IN1_RMP	
											rw	rw	rw	rw	

位 31:5 保留，必须保持复位值。

位 4 **TIM9_ITR1_RMP**：TIMER9 输入触发 1 重映射 (TIMER9 input Trigger 1 remap)
由软件置 1 和清零。
0： TIM3 输出触发
1： LPTIMER 的输出通道

位 3 **TIM5_ITR1_RMP**：TIMER5 输入触发 1 重映射 (TIMER9 input Trigger 1 remap)
由软件置 1 和清零。
0： TIM3 输出触发
1： LPTIMER 的输出通道

位 2 **TIM1_ITR2_RMP**：TIMER1 输入触发 2 重映射 (TIMER9 input Trigger 1 remap)
由软件置 1 和清零。
0： TIM3 输出触发
1： LPTIMER 的输出通道

位 1:0 **LPT_IN1_RMP**：LPTimer 输入触发 2 重映射 (LPTimer input Trigger 2 remap)
由软件置 1 和清零。
00：端口 B5 或端口 C0，通过 AF1 的复用功能选择
01：端口 A4 直接 IO 输入（信号输入与任何复用功能选择均无关）
10：端口 B9 直接 IO 输入（信号输入与任何复用功能选择均无关）
11：LPTIMER 的输出通道

21.6.10 LPTIM 寄存器映射

下表对 LPTIM 寄存器进行了汇总。

表 120. LPTIM 寄存器映射和复位值

偏移	寄存器名	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	LPTIM_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
0x004	LPTIM_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
0x008	LPTIM_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
0x00C	LPTIM_CFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENC	COUNTMODE	PRELOAD	WAVPOL	WAVE	TIMOUT	TRIGEN		Res.	TRIGSEL[2:0]			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
0x010	LPTIM_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSTARE	Res.	CNTSTRT	SNGSTRT	ENABLE
	Reset value																												0		0	0	0
0x014	LPTIM_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
0x018	LPTIM_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	
0x01C	LPTIM_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
0x020	LPTIM_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.	TIM9_ITR1_RMP	TIM5_ITR1_RMP	TIM1_ITR2_RMP	LPT_IN1_RMP	
	Reset value																												0	0	0	0	

有关寄存器边界地址的信息，请参见*第 54 页的第 2.2.2 节*。

22 独立看门狗 (IWDG)

22.1 IWDG 简介

此器件具有两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。两个看门狗外设（独立和窗口）均可用于检测并解决由软件错误导致的故障；当计数器达到给定的超时值时，触发一个中断（仅适用于窗口型看门狗）或产生系统复位。

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。窗口看门狗 (WWDG) 时钟由 APB1 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。

IWDG 最适合应用于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应用。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。有关窗口看门狗的详细信息，请参见*第 23 节：窗口看门狗 (WWDG)*。

22.2 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）
- 当递减计数器值达到 0x000 时产生复位（如果看门狗已激活）

22.3 IWDG 功能说明

图 221 给出了独立看门狗模块的功能框图。

通过向键寄存器 (IWDG_KR) 中写入值 0xCCCC 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）。

任何时候将关键字 0xAAAA 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。

22.3.1 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，若软件没有向关键字寄存器写入相应的值，则系统会产生复位。

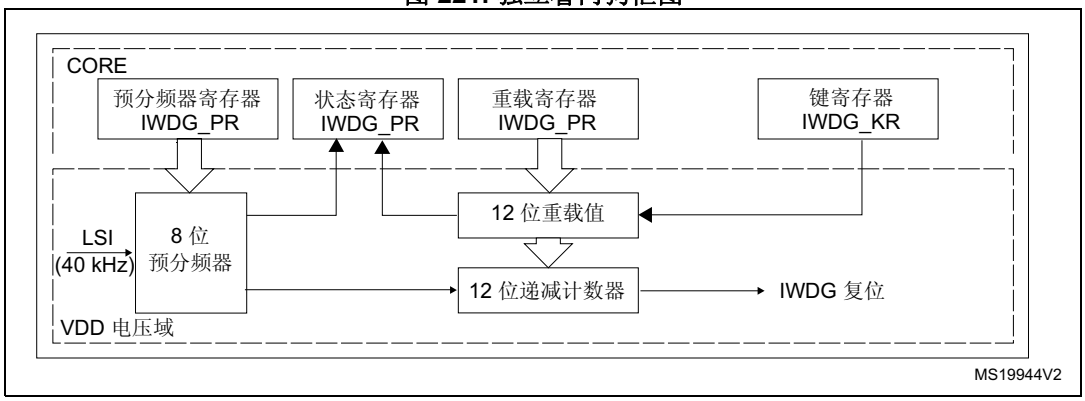
22.3.2 寄存器访问保护

IWDG_PR 和 IWDG_RLR 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 0x5555。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 0xAAAA）也会启动写保护功能。
状态寄存器指示预分频值和递减计数器是否正在被更新。

22.3.3 调试模式

当微控制器进入调试模式时（带 FPU 的 Cortex®-M4 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。有关详细信息，请参见*第 34.16.4 节：MCU APB1 调试冻结寄存器 (DBGMCU_APB1_FZ)*。

图 221. 独立看门狗框图



注： *看门狗功能由 V_{DD} 电压域供电，在停止模式和待机模式下仍能工作。*

表 121. 32 kHz (LSI) 频率条件下 IWDG 超时周期的最小值/最大值(1)

预分频器	PR[2:0] 位	最短超时 (ms) RL[11:0] = 0x000	最长超时 (ms) RL[11:0] = 0xFFF
/4	0	0.125	512
/8	1	0.25	1024
/16	2	0.5	2048
/32	3	1	4096
/64	4	2	8192
/128	5	4	16384
/256	6	8	32768

1. 这些时间均针对 32 kHz 时钟给出。实际上，微控制器内部的 RC 频率会有所变化。有关最大值和最小值的信息，请参见器件数据手册中的 LSI 振荡器特性表。

22.4 IWDG 寄存器

有关寄存器说明中使用的缩写，请参见*第 50 页的第 1.2 节*。

必须按半字（16 位）或字（32 位）访问外设寄存器。

22.4.1 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 0000h）

必须每隔一段时间便通过软件对这些位写入键值 AAAAh，否则当计数器计数到 0 时，看门狗会产生复位。

写入键值 5555h 可使能对 IWDG_PR 和 IWDG_RLR 寄存器的访问（请参见*第 22.3.2 节*）

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）

22.4.2 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 22.3.2 节*。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。

22.4.3 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 22.3.2 节*。这个值由软件设置，每次对 IWDG_KR 寄存器写入值 AAAAh 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。请参见*表 121* 。

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。

22.4.4 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.	Res.	RVU	PVU
														r	r

位 31:2 保留，必须保持复位值。

位 1 **RVU**：看门狗计数器重载值更新 (Watchdog counter reload value update)

该位由硬件置 1 以指示重载值正在更新。当在 V_{DD} 电压域下完成重载值更新操作后（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)

该位由硬件置 1 以指示预分频器值正在更新。当在 V_{DD} 电压域下完成预分频器值更新操作后（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或预分频器值，则必须等到 RVU 位被清零后才能更改重载值，而且必须等到 PVU 位被清零后才能更改预分频器值。但是，在更新预分频器和/或重载值之后，则无需等到 RVU 或 PVU 复位后再继续执行代码（即便进入低功耗模式，也会继续执行写操作至完成）*

22.4.5 IWDG 寄存器映射

下表提供了 IWDG 寄存器映射和复位值。

表 122. 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.	Res.	RVU	PVU
	Reset value																															0	0

有关寄存器边界地址的信息，请参见*第 54 页的第 2.2.2 节*。

23 窗口看门狗 (WWDG)

23.1 WWDG 简介

窗口看门狗通常被用来监测，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。除非程序在 T6 位变成 0 前刷新递减计数器的值，否则看门狗电路在达到预置的时间周期时，会产生一个 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。

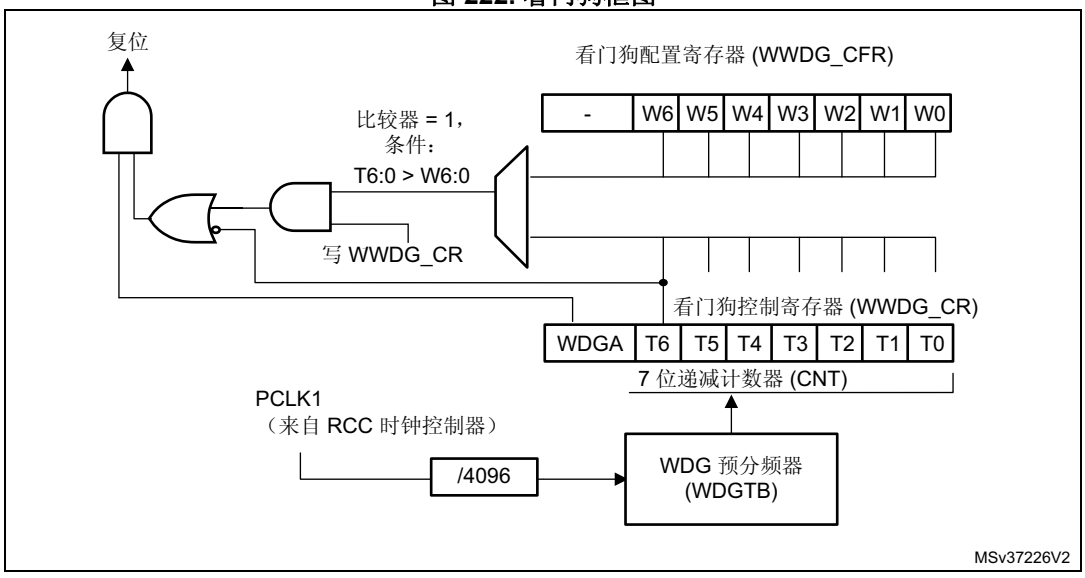
23.2 WWDG 主要特性

- 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 223*）
- 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）

23.3 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 222. 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。

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。

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 223*）。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 223* 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）。*

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。

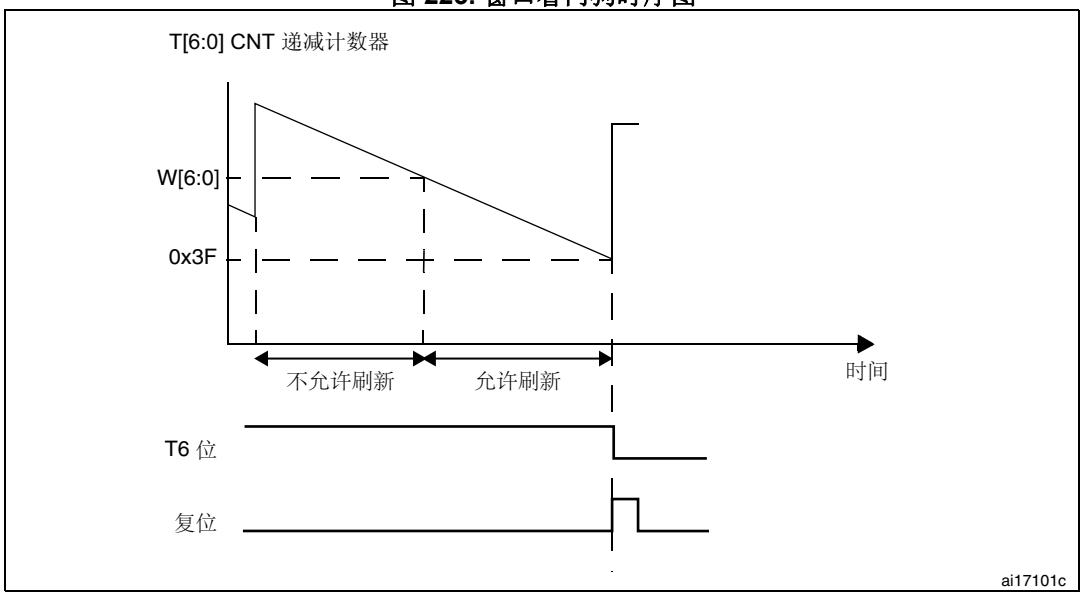
注： *当由于在更高优先级任务中有系统锁定而无法使用 EWI 中断时，最终会产生 WWDG 复位。*

23.4 如何设置看门狗超时

必须使用*图 223* 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 223. 窗口看门狗时序图



超时值的计算公式如下：

$$t_{WWDG} = t_{PCLK1} \times 4096 \times 2^{WDGTB[1:0]} \times ([T5:0] + 1) \quad (ms)$$

其中：

t_{WWDG}：WWDG 超时

t_{PCLK1}：APB1 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 APB1 频率等于 24 MHz，将 WDGTB[1:0] 设置为 3，将 T[5:0] 设置为 63：

$$t_{WWDG} = 1/24000 \times 4096 \times 2^{3} \times (63 + 1) = 21.85 \text{ ms}$$

有关 t_{WWDG} 的最小值和最大值，请参见数据手册。

23.5 调试模式

当微控制器进入调试模式时（带 FPU 的 Cortex®-M4 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 34.16.2 节：对定时器、看门狗、bxCAN 和 I2C 的调试支持*。

23.6 WWDG 寄存器

有关寄存器说明中使用的缩写，请参见*第 50 页的第 1.2 节*。

必须按半字（16 位）或字（32 位）访问外设寄存器。

23.6.1 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 (7-bit counter)（MSB 到 LSB）

这些位用来存储看门狗计数器的值。它每隔 (4096 x $2^{WDGTB[1:0]}$) PCLK1 个周期递减一次。当它从 0x40 滚动到 0x3F（T6 清零）时会产生复位。

23.6.2 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						

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 (PCLK1 div 4096) 分频器 1

01：CK 计数器时钟 (PCLK1 div 4096) 分频器 2

10：CK 计数器时钟 (PCLK1 div 4096) 分频器 4

11：CK 计数器时钟 (PCLK1 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。

23.6.3 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。

23.6.4 WWDG 寄存器映射

下表提供了 WWDG 寄存器映射和复位值。

表 123. 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 54 页的第 2.2.2 节*。

24 AES 硬件加速器 (AES)

24.1 简介

AES 硬件加速器 (AES) 使用完全符合联邦信息处理标准 (FIPS) 出版物 197 中规定的高级加密标准 (AES) 的算法来对数据进行加密或解密。

支持多种链接模式（ECB、CBC、CTR、GCM、GMAC、CCM），支持 128 或 256 位密钥大小。

AES 加速器为 32 位 AHB 外设。它支持对传入和传出数据进行 DMA 单次传输（需要两个 DMA 通道）。

AES 外设可为 STM32 加密库所实现的 AES 加密算法提供硬件加速。

AES 为 AMBA AHB 从外设，仅支持 32 位字单次访问（否则，会生成 AHB 总线错误，并会忽略写访问）。

24.2 AES 主要特性

- 自 2001 年 11 月起，符合 NIST *“高级加密标准 (AES)，FIPS 出版物 197”*
- 128 位数据块处理
- 支持 128 位和 256 位密钥长度
- 基于多种链接模式的加密和解密：
  - 电子密码本 (ECB) 模式
  - 密码块链接 (CBC) 模式
  - 计数器 (CTR) 模式
  - Galois 计数器模式 (GCM)
  - Galois 消息认证码 (GMAC) 模式
  - CBC-MAC 计数器 (CCM) 模式
- 在 ECB 模式下，51 或 75 个时钟周期延迟用于处理一个包含 128 位或 256 位密钥的 128 位数据块
- 集成有密钥调度器，包括密钥生成阶段（仅限 ECB 或 CBC 解密）
- AMBA AHB 从外设，仅支持 32 位字单次访问
- 256 位寄存器，用于存储加密密钥（8 个 32 位寄存器）
- 128 位寄存器，用于存储初始化向量（4 个 32 位寄存器）
- 32 位缓冲器，用于数据输入和输出
- 采用自动数据流控制，支持单次传输直接存储器访问 (DMA)（使用两个通道，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
- 在 AES 需要处理优先级更高的消息时，可通过软件将当前消息挂起，然后恢复原始消息

24.3 AES实现

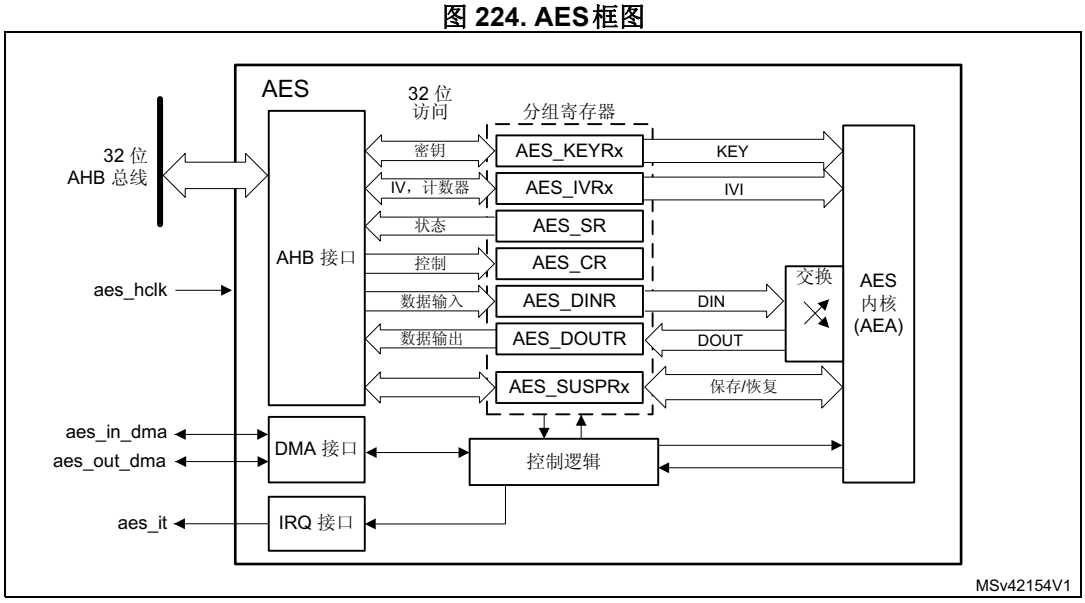
该器件具有单个AES外设实例。

24.4 AES 功能描述

24.4.1 AES框图

图 224 显示了 AES 的框图。

图 224. AES框图



24.4.2 AES 内部信号

表 124 描述了连接 AES 外设的用户相关内部信号。

表 124. AES 内部输入/输出信号

信号名称	信号类型	说明
aes_hclk	数字输入	AHB 总线时钟
aes_it	数字输出	AES中断请求
aes_in_dma	数字输入/输出	输入 DMA 单次请求/确认
aes_out_dma	数字输入/输出	输出 DMA 单次请求/确认

24.4.3 AES 加密内核

概述

AES 加密内核由以下部分组成：

- AES 算法 (AEA)
- 基于二元 Galois 域的乘法器 (GF2mul)
- 密钥输入
- 初始化向量 (IV) 输入
- 链接算法逻辑（XOR、反馈/计数器、屏蔽）

AES 内核适用于密钥长度为 128 位或 256 位的 128 位数据块（四字）。根据链接模式的不同，AES 可能需要一个 96 位初始化向量 IV（和一个 32 位计数器字段），也可能不需要。

AES 具有以下工作模式：

- **模式 1：**
  使用存储在 AES_KEYRx 寄存器中的密钥实现明文加密
- **模式 2：**
  ECB 或 CBC 解密密钥准备。在选择 ECB 或 CBC 链接模式的模式 3 之前，必须先使用此模式。准备解密的密钥自动存储在 AES_KEYRx 寄存器中。现在，AES 外设准备好切换到模式 3 以执行数据解密。
- **模式 3：**
  使用存储在 AES_KEYRx 寄存器中的密钥实现密文解密。当选择 ECB 和 CBC 链接模式时，必须通过模式 2 预先准备好密钥。
- **模式 4：**
  使用存储在 AES_KEYRx 寄存器中的密钥实现 ECB 或 CBC 密文单次解密（初始密钥自动生成）。

注： *仅在执行 ECB 和 CBC 解密时使用模式 2 和模式 4。*

当选择模式 4 时，只能进行一次解密，因此推荐使用模式 2 和模式 3。

通过对 AES_CR 寄存器中的 MODE[1:0] 位域进行编程来选择工作模式。只能在 AES 外设已禁止时进行。

典型数据处理

AES 的典型使用如*第 660 页的第 24.4.4 节：用于执行密码操作的 AES 过程*中所述。

注： *中间 AEA 阶段的输出始终不会在加密边界外显示，IVI 位域除外。*

链接模式

AES 支持以下链接模式，可通过 AES_CR 寄存器的 CHMOD[2:0] 位域选择：

- 电子密码本 (ECB)
- 密码块链接 (CBC)
- 计数器 (CTR)
- Galois 计数器模式 (GCM)
- Galois 消息认证码 (GMAC)
- CBC-MAC 计数器模式 (CCM)。

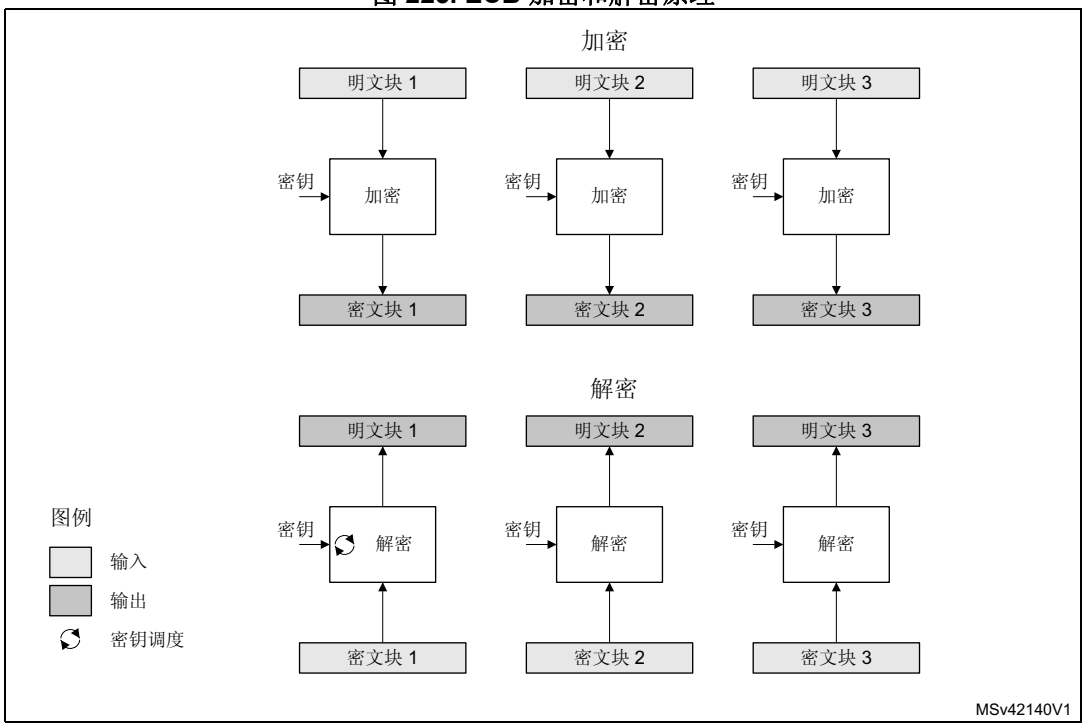
注： *只有在 AES 禁止（AES_CR 寄存器的位 EN 置 0）时，才能更改链接模式。*

后续子章节对每个 AES 链接模式的原理进行了介绍。

详细信息请参见从*第 24.4.8 节：AES 基本链接模式（ECB 和 CBC）*开始的专用章节。

电子密码本 (ECB) 模式

图 225. ECB 加密和解密原理

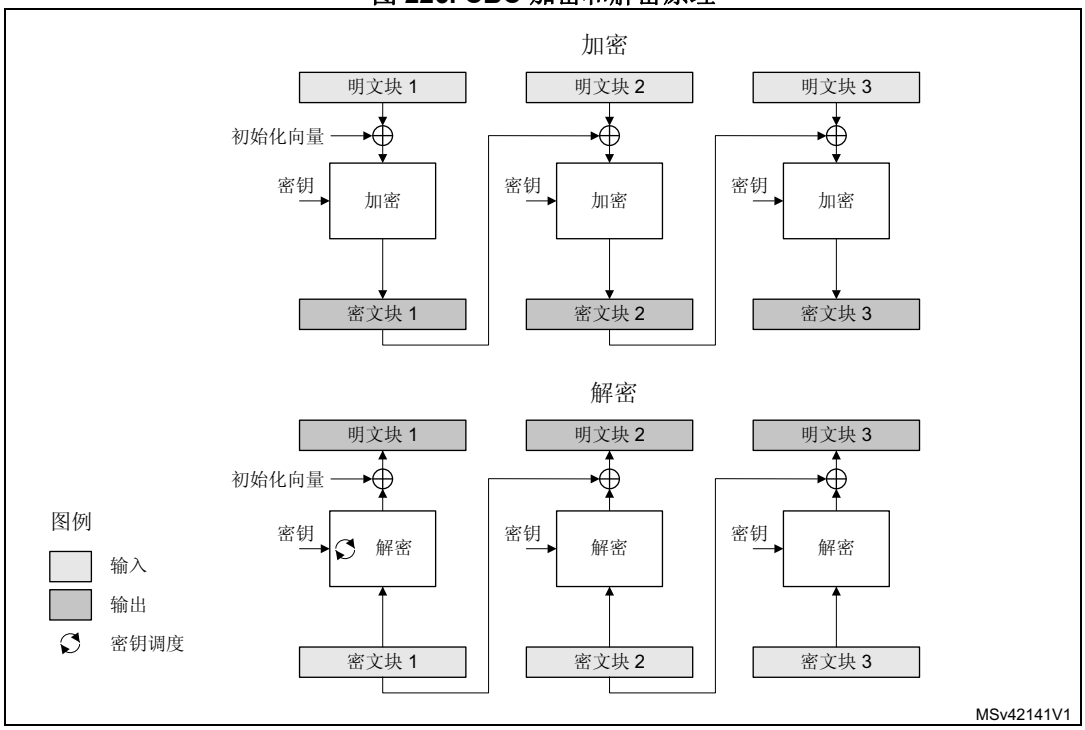


ECB 是最简单的工作模式。无链接操作，且无特殊初始化阶段。消息会划分到各个块中，并对各个块分别进行加密或解密。

注：　对于解密，在处理第一个块之前需要特殊的密钥调度。

密码块链接 (CBC) 模式

图 226. CBC 加密和解密原理

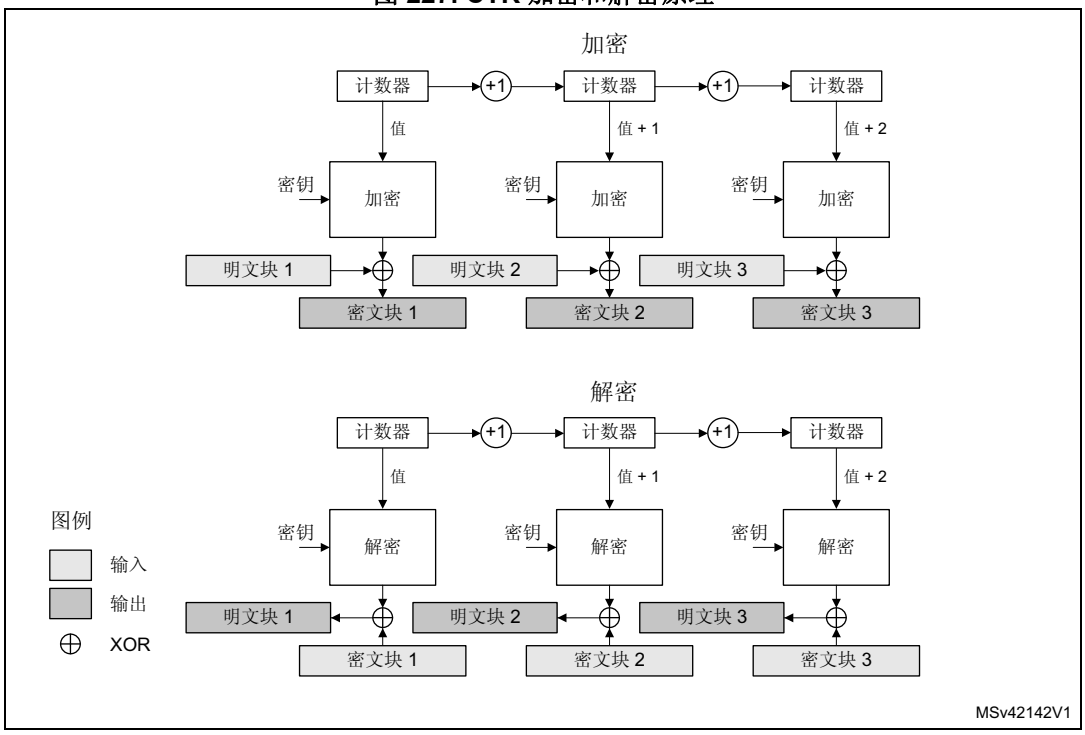


在 CBC 模式下，每个块的输出均与下一个块的输入相连。为了确保每条消息都是唯一的，会在第一个块处理过程中使用初始化向量。

注： *对于解密，在处理第一个块之前需要特殊的密钥调度。*

计数器 (CTR) 模式

图 227. CTR 加密和解密原理

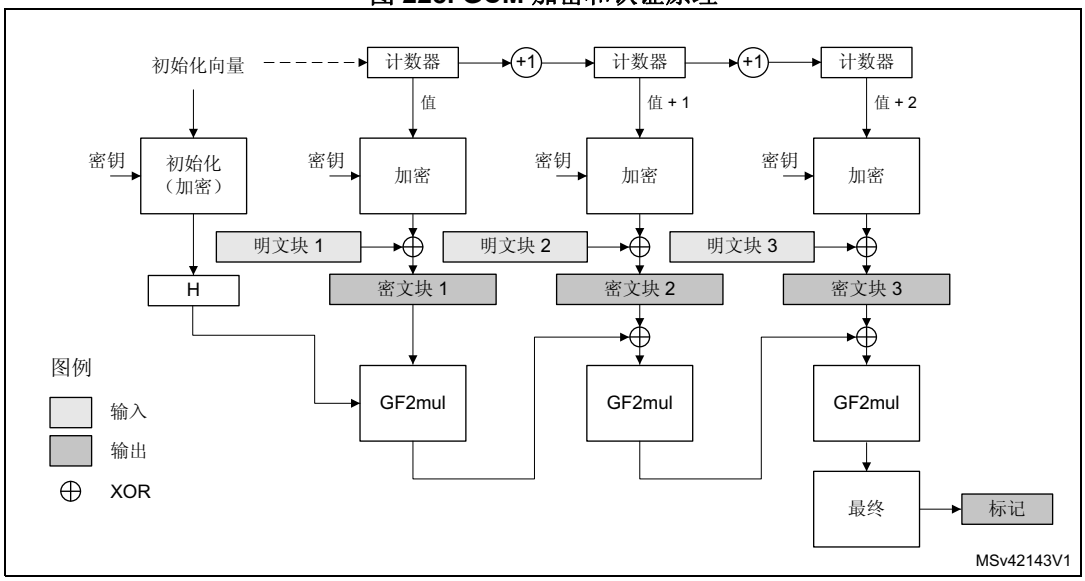


CTR 模式使用 AES 内核生成密钥流。这些密钥将与明文进行逻辑异或运算，以获得密文（如 NIST *特别出版物 800-38A，块密码工作模式的建议*中所述）。

注： *与 ECB 和 CBC 模式不同，CTR 解密无需密钥调度，因为在该链接方案中，AES 内核始终在加密模式下用于生成密钥流或计数器块。*

Galois/计数器模式 (GCM)

图 228. GCM 加密和认证原理

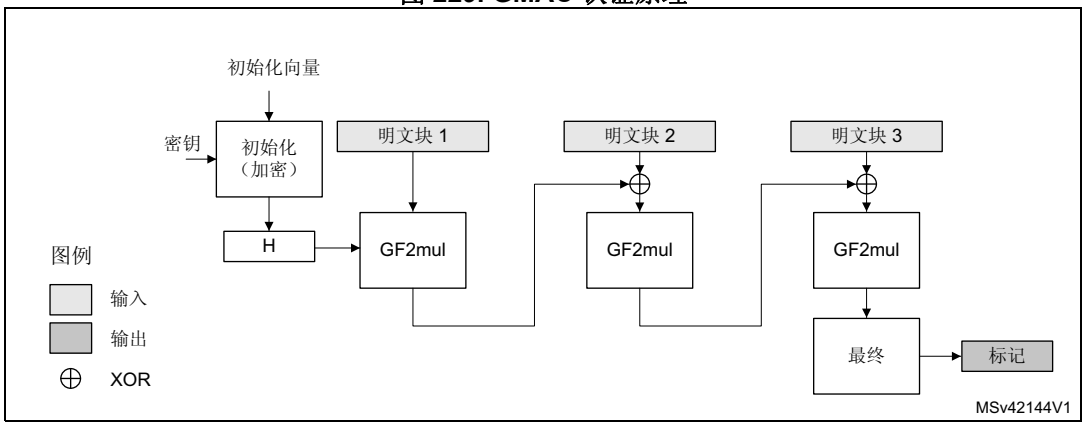


在 Galois/计数器模式 (GCM) 下，将对明文消息进行加密，同时会对消息认证码 (MAC) 进行计算，从而生成相应的密文及其 MAC（也称为认证标记）。NIST *特别出版物 800-38D，块密码工作模式的建议 - Galois/计数器模式 (GCM) 和 GMAC* 中对此进行了相关定义。

GCM 模式基于 AES 计数器模式，可实现保密性。它使用固定有限域乘法器来计算消息认证码。消息末尾处需要一个初始值和一个特定的 128 位块。

Galois 消息认证码 (GMAC) 原理

图 229. GMAC 认证原理

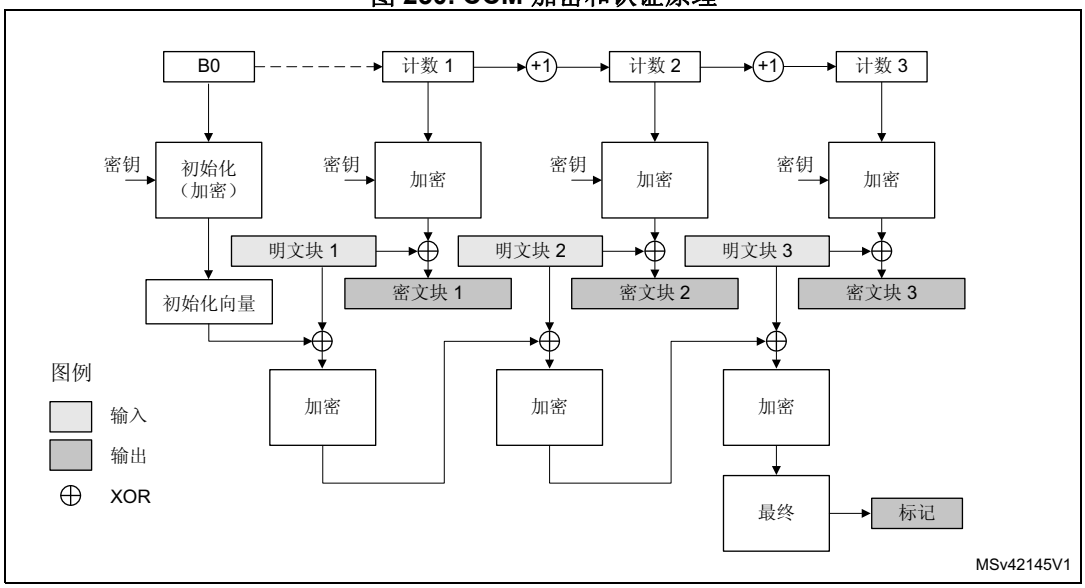


Galois 消息认证码 (GMAC) 允许认证消息并生成相应的消息认证码 (MAC)。NIST *特别出版物 800-38D，块密码工作模式的建议 - Galois/计数器模式 (GCM) 和 GMAC* 中对此进行了相关定义。

GMAC 与 GCM 类似，只不过它应用于仅由明文认证数据组成的消息（即只有标头，无有效负载）。

CBC-MAC 计数器 (CCM) 原理

图 230. CCM 加密和认证原理



在密码块链接-消息认证码计数器 (CCM) 模式下，将对明文消息进行加密，同时会对消息认证码 (MAC) 进行计算，从而生成相应的密文以及相应的 MAC（也称为标记）。NIST *特别出版物 800-38C，块密码工作模式的建议 - CCM 模式实现认证和保密性*中对此进行了介绍。

CCM 模式基于 AES 计数器模式，可实现保密性，并且使用 CBC 来计算消息认证码。它需要一个初始值。

与 GCM 类似，CCM 链接模式可应用于仅由明文认证数据组成的消息（即只有标头，无有效负载）。请注意，这种 CCM 使用方式不被称为 CMAC（它并非类似于 GCM/GMAC），NIST 不推荐这种用途。

24.4.4 用于执行密码操作的 AES 过程

简介

下面对典型密码操作进行了介绍。详细信息请参见从*第 24.4.8 节：AES 基本链接模式（ECB 和 CBC）*开始的章节。

图 231 和*图 232* 中的流程图介绍了 STM32 加密库如何实现 AES 算法。AES 可在 ECB、CBC、CTR、CCM 和 GCM 工作模式下加速执行 AES-128 和 AES-256 加密算法。

注：有关加密库的更多详细信息，请参见 UM1924 用户手册“STM32 加密库”（可从 www.st.com 获取）。

图 231. STM32 加密库 AES 流程图示例

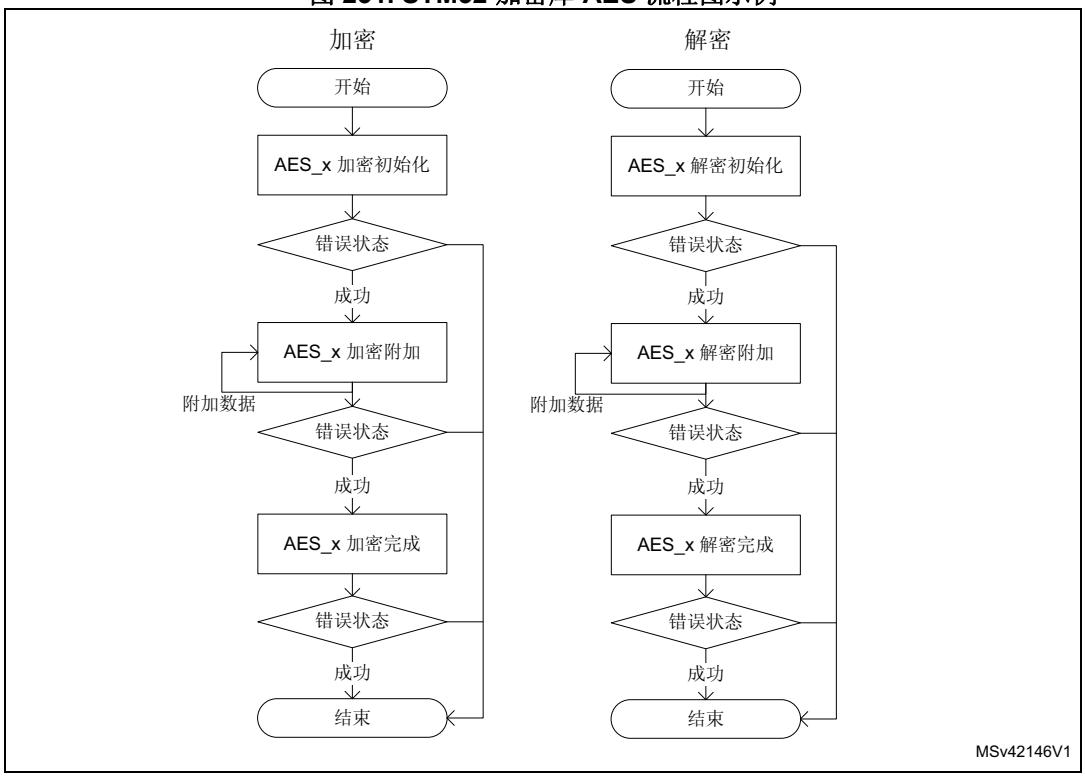
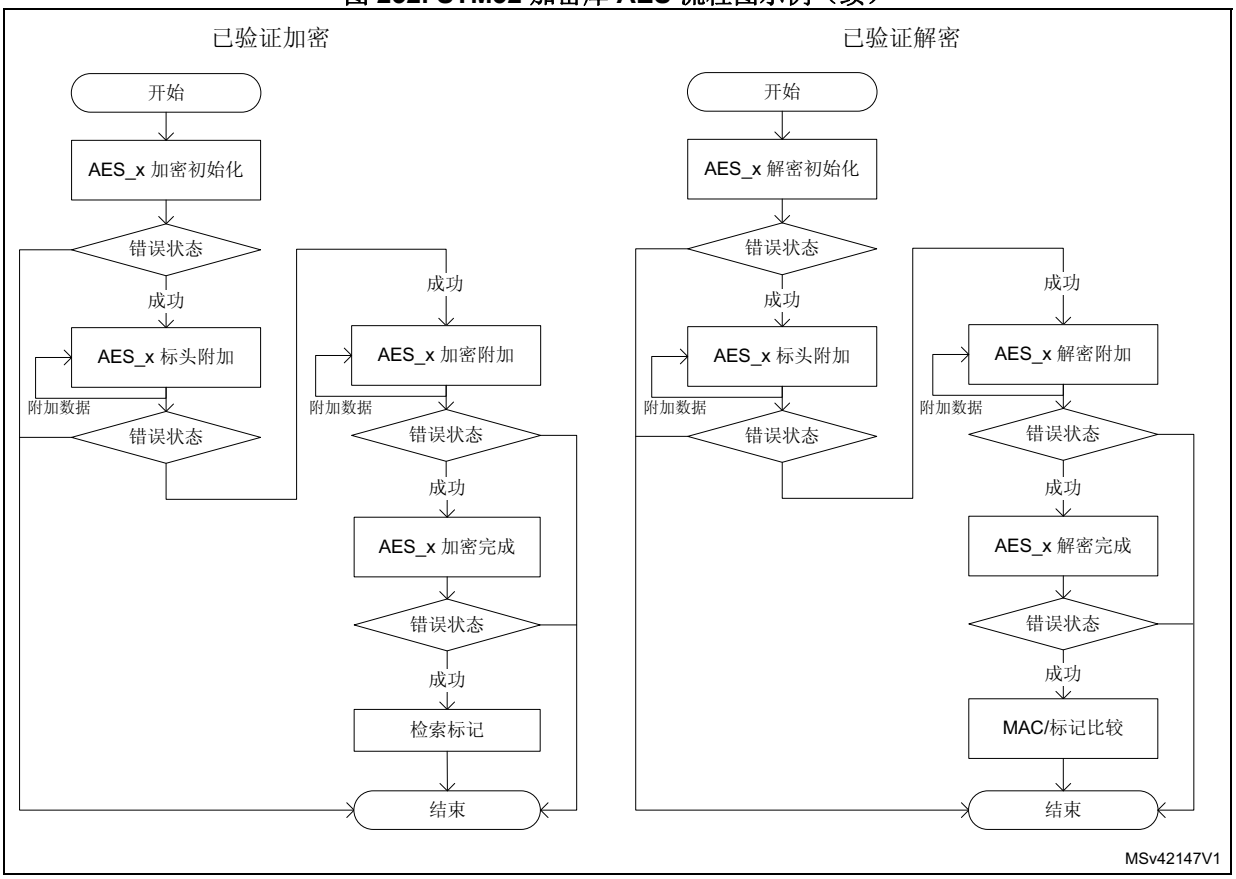


图 232. STM32 加密库 AES 流程图示例（续）



AES 初始化

要初始化 AES，首先通过将 AES_CR 寄存器中的 EN 位清零来禁止它。然后以任何顺序执行以下步骤：

- 通过对 AES_CR 寄存器中的 MODE[1:0] 位域进行编程来配置 AES 模式。
  - 对于加密，必须选择模式 1 (MODE[1:0] = 00)。
  - 对于解密，必须选择模式 3 (MODE[1:0] = 10)，除非使用 ECB 或 CBC 链接模式。在后一种情况下，必须执行加密密钥的初始密钥生成，如*第 24.4.5 节：AES 解密密钥准备*中所述。
- 通过对 AES_CR 寄存器中的 CHMOD[2:0] 位域进行编程来选择链接模式
- 使用 AES_CR 寄存器中的 KEYSIZE 位域配置密钥大小（128 位或 256 位）
- 将对称密钥写入 AES_KEYRx 寄存器（4 或 8 个寄存器，具体取决于密钥大小）。
- 使用 AES_CR 寄存器中的 DATATYPE[1:0] 位域配置数据类型（1 位、8 位、16 位或 32 位）
- 必要时（例如，CBC 或 CTR 链接模式），向 AES_IVRx 寄存器中写入初始化向量。

附加数据

本节介绍附加数据进行处理的不同方式，其中要处理数据的大小不是 128 位的倍数。

有关 ECB、CBC 和 GCM 加密模式，请参见*第 24.4.6 节：AES 密文窃取和数据填充*。这些情况下的倒数第二个和最后一个块管理比本节描述序列下的更复杂。

通过轮询附加数据

此方法使用标志轮询来控制附加数据。

对于所有其他情况，通过以下序列附加数据：

1. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
2. 重复以下子序列，直到完全处理有效负载：
   a) 将四个输入数据字写入 AES_DINR 寄存器。
   b) AES_SR 中的状态标志 CCF 置 1 后，从 AES_DOUTR 寄存器读取四个数据字。
   c) 通过将 AES_CR 寄存器中的 CCFC 位置 1，将 CCF 标志清零。
   d) 如果刚刚处理的数据块是消息的倒数第二个块，并且要处理的最后一个块中的有效数据低于 128 位，则用零填充最后一个块的其余部分。
3. 丢弃不属于有效负载的数据，然后通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。

注： *在对 AES_DINR 寄存器的两次连续写入之间自动插入最多三个等待周期，以便将密钥发送到 AES 处理器。*

使用中断附加数据

此方法通过以下序列，使用 AES 外设中断来控制附加数据：

1. 通过将 AES_CR 寄存器中的 CCFIE 位置 1 来使能 AES 中断。
2. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
3. 将前四个输入数据字写入 AES_DINR 寄存器。
4. 中断时处理 AES 中断服务程序中的数据：
   a) 从 AES_DOUTR 寄存器读取四个输出数据字。
   b) 通过将 AES_CR 寄存器中的 CCFC 位置 1，将 CCF 标志和挂起中断清零
   c) 如果刚刚处理的数据块是消息的倒数第二个块，并且要处理的最后一个块中的有效数据低于 128 位，则用零填充最后一个块的其余部分。然后继续进行 4e)。
   d) 如果刚刚处理的数据块是消息的最后一块，则丢弃不属于有效负载的数据，然后通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设，并退出中断服务程序。
   e) 将接下来的四个输入数据字写入 AES_DINR 寄存器，并退出中断服务程序。

注： *AES 允许连续读取或写入操作之间存在延迟，例如在两次 AES 计算之间要响应的另一个外设的中断。*

使用 DMA 附加数据

借助此方法，所有传输和处理过程均由 DMA 和 AES 管理。要使用此方法，请按照以下步骤操作：

1. 用零填充块的其余部分来准备最后四个字的数据块（如果要处理的数据没有完全填充它）。
2. 将 DMA 控制器配置为将来自存储器的要处理的数据传输到 AES 外设输入以及将来自 AES 外设输出的已处理数据传输到存储器，如*第 24.4.16 节：AES DMA 接口*所述。配置 DMA 控制器，以在传输完成时产生中断。
3. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设
4. 将 AES_CR 寄存器中的 DMAINEN 和 DMAOUTEN 位置 1，以使能 DMA 请求。
5. 当 DMA 中断指示传输完成时，从存储器中获取 AES 处理过的数据。

注： *CCF 标志在这种方法中没有作用，因为读取 AES_DOUTR 寄存器是通过 DMA 自动管理的，在计算过程结束时没有任何软件操作。*

24.4.5 AES 解密密钥准备

对于 ECB 或 CBC 解密，第一轮解密的密钥必须从最后一轮加密的密钥中生成。因此，在执行解密之前需要完整的加密密钥调度。除 ECB 或 CBC 模式外的 AES 解密**不需要**密钥准备。

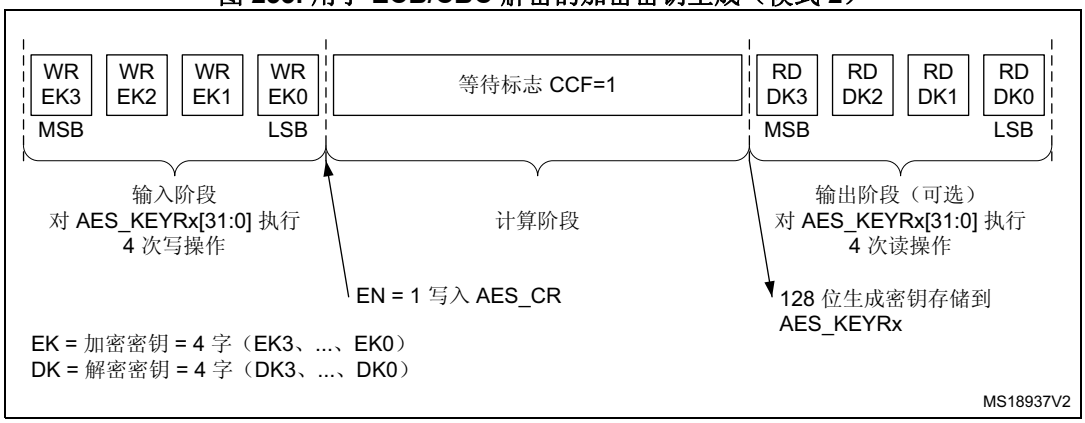
推荐方法是通过将 AES_CR 的 MODE[1:0] 位域设置为 01 来选择模式 2（仅密钥处理），然后通过将 MODE[1:0] 设置为 10 来继续解密（模式 3，仅解密）。模式 2 的使用说明如下：

1. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。
2. 通过将 AES_CR 的 MODE[1:0] 位域设置为 01 来选择模式 2。由于此密钥生成模式独立于所选的链接算法，因此 CHMOD[2:0] 位域在这种情况下是没有意义的。
3. 通过 AES_CR 寄存器的 KEYSIZE 位，将密钥长度设置为 128 或 256 位。
4. 将加密密钥写入 AES_KEYRx 寄存器（128 或 256 位），如*图 233* 所示。对 AES_IVRx 寄存器执行写入操作不起作用。
5. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
6. 等待 AES_SR 寄存器中的 CCF 标志置 1。
7. 生成的密钥位于 AES 内核中，可用于解密。如果需要，应用程序还可以读取 AES_KEYRx 寄存器以获得生成的密钥，如*图 233* 所示（已处理密钥自动加载到 AES_KEYRx 寄存器中）。

注： *生成密钥可用时，将由硬件禁止 AES。*

要重新开始计算生成密钥，请重复步骤 4、5、6 和 7。

图 233. 用于 ECB/CBC 解密的加密密钥生成（模式 2）



如果软件存储了为解密准备的初始密钥，则对于要用给定密钥解密的所有数据，仅执行一次密钥调度操作就已足够。

注： *密钥准备操作持续 80 或 109 个时钟周期，具体取决于密钥大小（128 或 256 位）。*

注： *备用密钥准备是通过将 AES_CR 寄存器中的 MODE[1:0] 位域设置为 11 来选择模式 4。这种情况下，模式 3 不可用。*

24.4.6 AES 密文窃取和数据填充

在 ECB 或 CBC 模式下使用 AES 管理的消息大小不是块大小（128 位）的整数倍时，请使用密文窃取技术，如 NIST *特别出版物 800-38A，块密码工作模式的建议：CBC 模式密文窃取的三种变型*中介绍的窃取技术。由于器件上的 AES 外设不支持这类技术，因此输入数据的**最后两个块**必须由应用程序以特殊方式处理。

注： *本参考手册中未对密文窃取技术进行介绍。*

类似地，在除 ECB 或 CBC 之外的模式下使用 AES 时，必须先用零填充不完整的输入数据块（即，输入数据短于 128 位的块），然后再加密（即，必须在数据串的尾端附加额外的位）。解密后必须丢弃额外填充的位。由于 AES 不会对**最后一个块**执行自动数据填充操作，因此应用程序必须遵循*第 660 页的第 24.4.4 节：用于执行密码操作的 AES 过程*中给出的建议来管理大小不是 128 位倍数的消息。

注： *填充数据根据 AES_CR 寄存器中的 DATATYPE[1:0] 字段，以类似于正常数据的方式进行交换（有关详细信息，请参见第 682 页的第 24.4.13 节：.AES 数据寄存器和数据交换）。*

当最后一个块的输入数据**低于 128 位**时，需要采取解决方案以便在进行 **GCM 加密**时正确计算认证标记。在 GCM 加密有效负载阶段且在插入小于 128 位的最后一个明文块之前，应用程序必须执行以下步骤：

1. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。
2. 通过向 AES_CR 寄存器中的 CHMOD[2:0] 位域写入 010 来将模式更改为 CTR。
3. 用零填充最后一个块（小于 128 位），以便获得 128 位的完整块，然后将其写入 AES_DINR 寄存器中。
4. 加密完成后，从 AES_DOUTR 寄存器中读取 128 位密文，并将其存储为中间数据。
5. 通过向 AES_CR 寄存器中的 CHMOD[2:0] 位域写入 011 来将模式再次更改为 GCM。
6. 通过向 AES_CR 寄存器的 GCMPH[1:0] 位域写入 11 来选择最后阶段。
7. 在中间数据中，将最后有效负载块的填充位所对应的位置零，然后将得到的数据插入 AES_DINR 寄存器中。
8. 等待操作完成，然后读取 AES_DOUTR 上的数据。这些数据将被丢弃。
9. 按*第 672 页的第 24.4.10 节：AES Galois/ 计数器模式 (GCM)*所述应用正常最后阶段。

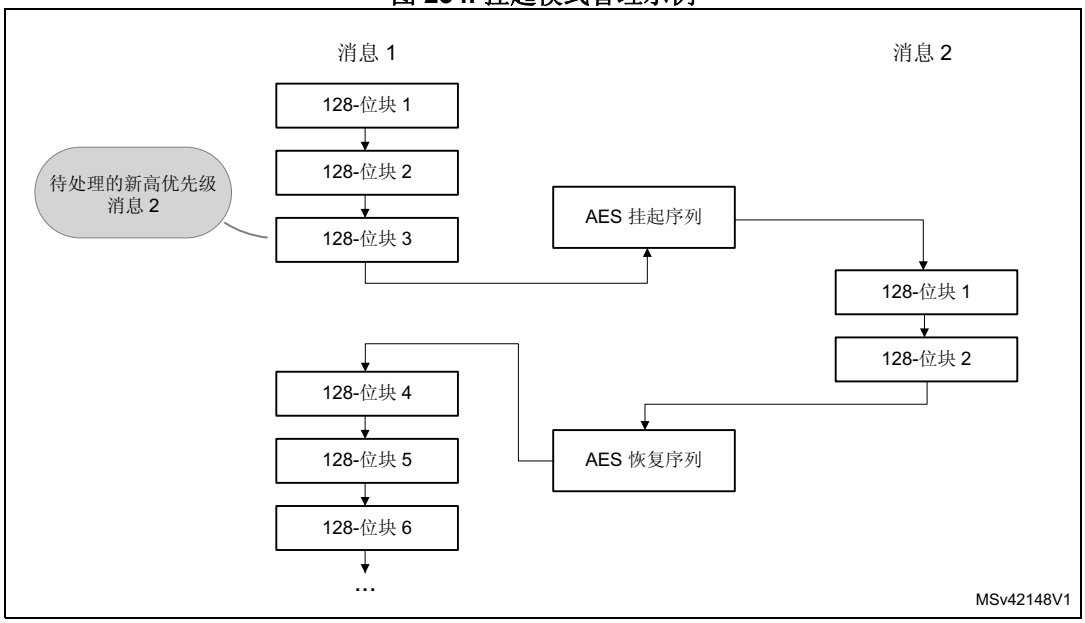
24.4.7 AES 任务挂起和恢复

如果必须处理另一条优先级较高的消息，则可将消息挂起。该最高优先级消息完成发送后，可在加密或解密模式下恢复挂起的消息。

挂起/恢复操作不会破坏链接运算，并且再次使能 AES 以接收下一个数据块时，可立即恢复消息处理。

图 234 所示为挂起/恢复操作示例：为发送长度更短、优先级更高的消息 2 而将消息 1 挂起。

图 234. 挂起模式管理示例



有关挂起/恢复操作的详细说明，请参见每个 AES 模式的相应章节。

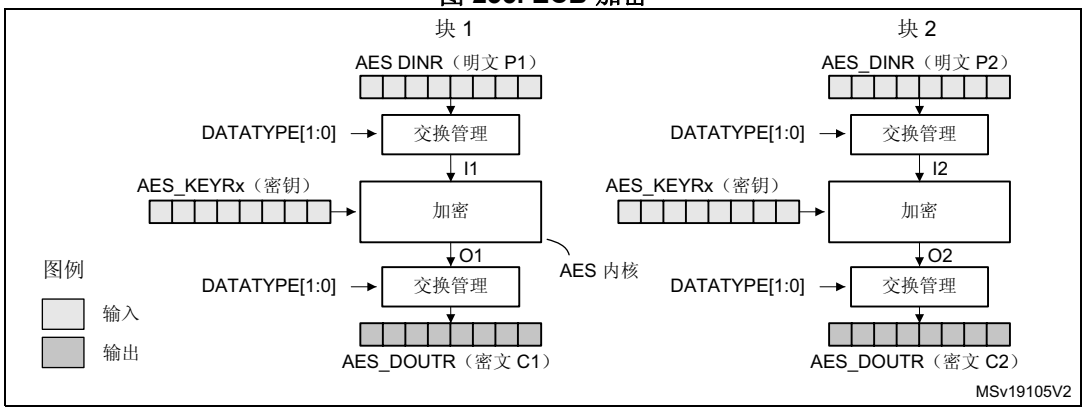
24.4.8 AES 基本链接模式（ECB 和 CBC）

概述

本节简要介绍 AES 计算内核提供的四种基本工作模式：ECB 加密、ECB 解密、CBC 加密和 CBC 解密。有关详细信息，请参见 2001 年 11 月 26 日的 FIPS 出版物 197。

图 235 介绍了电子密码本 (ECB) 加密。

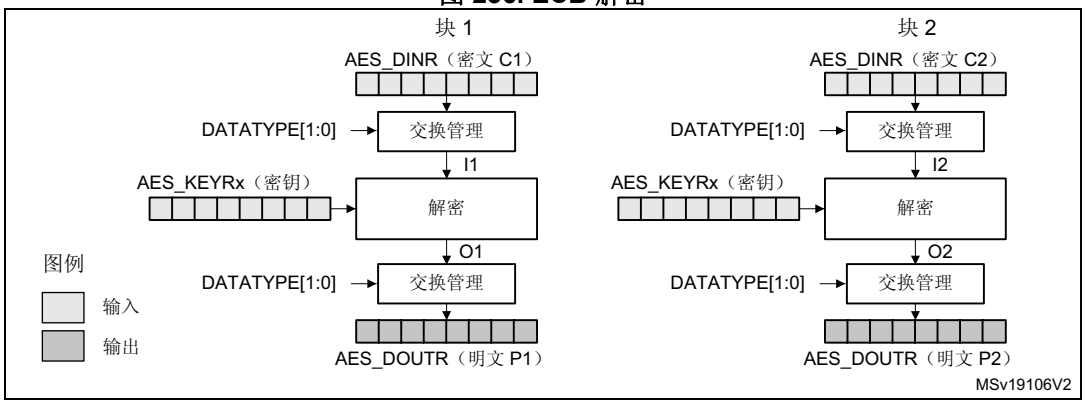
图 235. ECB 加密



在 ECB 加密模式下，AES_DINR 寄存器中的 128 位明文输入数据块 Px 首先进行位/字节/半字交换。使用 128 位或 256 位密钥，使用设置为加密模式的 AES 内核对交换结果 Ix 进行处理。加密结果 Ox 经过位/字节/半字交换，然后作为 128 位密文输出数据块 Cx 存储在 AES_DOUTR 寄存器中。ECB 加密以这种方式继续，直到最后一个完整的明文块被加密。

图 236 介绍了电子密码本 (ECB) 解密。

图 236. ECB 解密

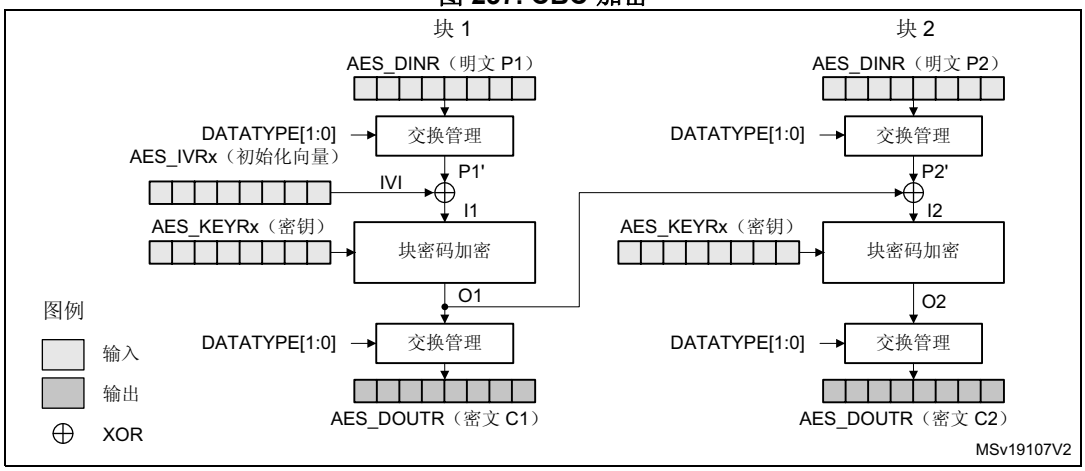


要在 ECB 模式下执行 AES 解密，需要通过收集最后一轮加密密钥准备密钥（需要针对加密首先执行完整的密钥调度），并将其用作解密密文的第一个轮密钥。该准备由 AES 内核提供支持。

在 ECB 解密模式下，AES_DINR 寄存器中的 128 位密文输入数据块 C1 首先进行位/字节/半字交换。该密钥序列与 ECB 加密中的密钥序列相反。使用先前准备的解密密钥，通过在解密模式下设置的 AES 内核对交换结果 I1 进行处理。解密结果经过位/字节/半字交换，然后作为 128 位明文输出数据块 P1 存储在 AES_DOUTR 寄存器中。ECB 解密以这种方式继续，直到最后一个完整的密文块被解密。

图 237 说明了密码块链接 (CBC) 加密模式。

图 237. CBC 加密

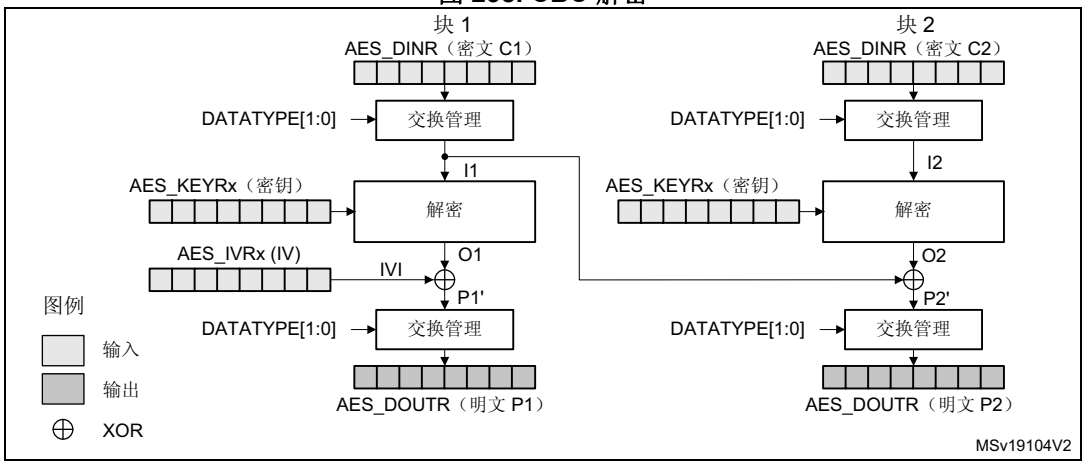


在 CBC 加密模式下，执行位/字节/半字交换 (P1') 后的第一个明文输入块与一个 128 位 IVI 位域（初始化向量和计数器）进行逻辑异或运算，生成 I1 输入数据，然后通过 AES 内核使用 128 位或 256 位密钥进行加密。得到的 128 位输出块 O1 经过交换操作之后用作密文 C1。然后将 O1 数据与第二块明文数据 P2' 进行逻辑异或运算，生成用于 AES 内核的 I2 输入数据以生成第二块密文数据。数据块的链接以这种方式继续，直到消息中的最后一个明文块被加密。

如果消息大小不是 128 位的倍数，则将按照*第 24.4.6 节：AES 密文窃取和数据填充*中说明的方式对最后的不完整数据块进行加密。

图 238 说明了密码块链接 (CBC) 解密模式。

图 238. CBC 解密



在 CBC 解密模式下，类似于 ECB 解密模式，必须准备密钥才可以执行 AES 解密。

完成密钥准备过程后，按下述方式执行解密：将第一个 128 位密文块（在交换操作之后）作为 AES 内核输入块 I1，使用 128 位或 256 位密钥进行解密操作。其输出 O1 与 128 位 IVI 字段（必须与加密期间使用的字段相同）进行逻辑异或运算以生成第一个明文块 P1。

除了使用来自第一个块的 I1 数据代替初始化向量外，第二个密文块的处理方式与第一个块相同。

解密以这种方式继续，直到最后一个完整的密文块被解密。

如果消息大小不是 128 位的倍数，则将按照*第 24.4.6 节：AES 密文窃取和数据填充*中说明的方式对最后的不完整数据块进行解密。

有关数据交换的更多信息，请参见*第 24.4.13 节：.AES 数据寄存器和数据交换*。

ECB/CBC 加密序列

执行 ECB/CBC 加密的事件序列（详见*第 24.4.4 节*）：

1. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。
2. 通过将 AES_CR 寄存器的 MODE[1:0] 位域设置为 00 来选择模式 1，通过将 AES_CR 寄存器的 CHMOD[2:0] 位域设置为 000 或 001 来分别选择 ECB 或 CBC 链接模式。也可以使用 DATATYPE[1:0] 位域定义数据类型。
3. 通过 AES_CR 寄存器中的 KEYSIZE 位选择 128 位或 256 位密钥长度。
4. 将加密密钥写入 AES_KEYRx 寄存器（128 或 256 位）。如果选择了 CBC 模式，则用初始化向量数据填充 AES_IVRx 寄存器。
5. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
6. 对 AES_DINR 寄存器执行四次写入操作，以输入明文（MSB 优先），如*图 239* 所示。
7. 等待 AES_SR 寄存器中的 CCF 标志置 1。
8. 对 AES_DOUTR 寄存器进行四次读取操作，以获取密文（MSB 优先），如*图 239* 所示。然后通过将 AES_CR 寄存器中的 CCFC 位置 1，将 CCF 标志清零。
9. 重复执行步骤 6、7、8，处理使用相同加密密钥的所有块。

图 239. ECB/CBC 加密（模式 1）

ECB/CBC 解密序列

执行 AES ECB/CBC 解密的事件序列如下（详见*第 24.4.4 节*）：

1. 按照*第 663 页的第 24.4.5 节：AES 解密密钥准备*中所述的步骤，准备 AES 内核中的解密密钥。
2. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。
3. 通过将 AES_CR 寄存器的 MODE[1:0] 位域设置为 10 来选择模式 3，通过将 AES_CR 寄存器的 CHMOD[2:0] 位域设置为 000 或 001 来分别选择 ECB 或 CBC 链接模式。也可以使用 DATATYPE[1:0] 位域定义数据类型。
4. 通过 AES_CR 寄存器的 KEYSIZE 位域，选择 128 或 256 位密钥长度。
5. 将初始化向量写入 AES_IVRx 寄存器（仅在 CBC 模式下需要此步骤）。
6. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES。
7. 对 AES_DINR 寄存器执行四次写入操作，以输入密文（MSB 优先），如*图 240* 所示。
8. 等待 AES_SR 寄存器中的 CCF 标志置 1。
9. 对 AES_DOUTR 寄存器进行四次读取操作，以获取明文（MSB 优先），如*图 240* 所示。然后通过将 AES_CR 寄存器中的 CCFC 位置 1，将 CCF 标志清零。
10. 重复执行步骤 7、8、9，处理使用相同密钥加密的所有块。

图 240. ECB/CBC 解密（模式 3）

ECB/CBC 模式下的挂起/恢复操作

要挂起消息处理，请执行以下操作：

1. 如果使用 DMA，则通过将 AES_CR 寄存器中的 DMAINEN 位清零来停止 AES DMA 对 IN FIFO 的数据传输。
2. 如果未使用 DMA，则读取四次 AES_DOUTR 寄存器以保存最后处理的块。如果使用 DMA，等待 AES_SR 寄存器中的 CCF 标志置 1，然后通过将 AES_CR 寄存器中的 DMAOUTEN 输出位清零来停止 DMA 对 OUT FIFO 的数据传输。
3. 如果未使用 DMA，轮询 AES_SR 寄存器的 CCF 标志，直到它变为 1（计算完成）。
4. 通过将 AES_CR 寄存器中的 CCFC 位置 1，将 CCF 标志清零。
5. 保存初始化向量寄存器（仅在 CBC 模式下需要此步骤，因为 AES_IVRx 寄存器在数据处理过程中已发生更改）。
6. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。
7. 将当前 AES 配置保存到存储器中（AES 初始化向量值除外）。
8. 如果使用 DMA，则保存 DMA 控制器状态（指向 IN 和 OUT 数据传输的指针以及剩余字节数等）。

注： *在第 7 步中，如果中断的进程是解密，则可以选择将存储在 AES_KEYRx 寄存器中的已生成密钥信息保存在存储器中。否则无需保存这些寄存器，因为应用程序已知初始密钥值*

要恢复消息处理，请执行以下操作：

1. 如果使用 DMA，则配置 DMA 控制器以完成 FIFO IN 和 FIFO OUT 传输的其余部分。
2. 确保 AES 禁止（AES_CR 的 EN 位必须为 0）。
3. 使用已保存配置的值恢复 AES_CR 和 AES_KEYRx 寄存器设置。在解密的情况下，可将生成的密钥信息写入 AES_KEYRx 寄存器，而不是原始密钥值。
4. 按*第 24.4.5 节：AES 解密密钥准备*所述准备解密密钥（仅 ECB 或 CBC 解密需要此步骤）。如果已将生成的密钥信息加载到 AES_KEYRx 寄存器中，则此步骤不是必需的。
5. 使用保存的配置恢复 AES_IVRx 寄存器（仅在 CBC 模式下需要此步骤）。
6. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
7. 如果使用 DMA，将 AES_CR 寄存器中的 DMAINEN 和 DMAOUTEN 位置 1，以使能 AES DMA 传输。

使用模式 4 执行备用单次 ECB/CBC 解密

使用模式 4 执行单次 ECB/CBC 解密的事件序列为：

1. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。
2. 通过将 AES_CR 寄存器的 MODE[1:0] 位域设置为 11 来选择模式 4，通过将 AES_CR 寄存器的 CHMOD[2:0] 位域设置为 000 或 001 来分别选择 ECB 或 CBC 链接模式。
3. 通过 AES_CR 寄存器的 KEYSIZE 位域，选择 128 或 256 位密钥长度。
4. 将加密密钥写入 AES_KEYRx 寄存器。如果选择 CBC 模式，则写入 AES_IVRx 寄存器。
5. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
6. 对 AES_DINR 寄存器执行四次写入操作，以输入密文（MSB 优先）。
7. 等待 AES_SR 寄存器中的 CCF 标志置 1。
8. 对 AES_DOUTR 寄存器进行四次读取操作，以获取明文（MSB 优先）。然后通过将 AES_CR 寄存器中的 CCFC 位置 1，将 CCF 标志清零。

注： *选择模式 4 时，不能使用模式 3。*

在模式 4 下，AES_KEYRx 寄存器包含所有处理阶段的加密密钥。生成密钥不会存储在这些寄存器中。它会存储在 AES 内部。

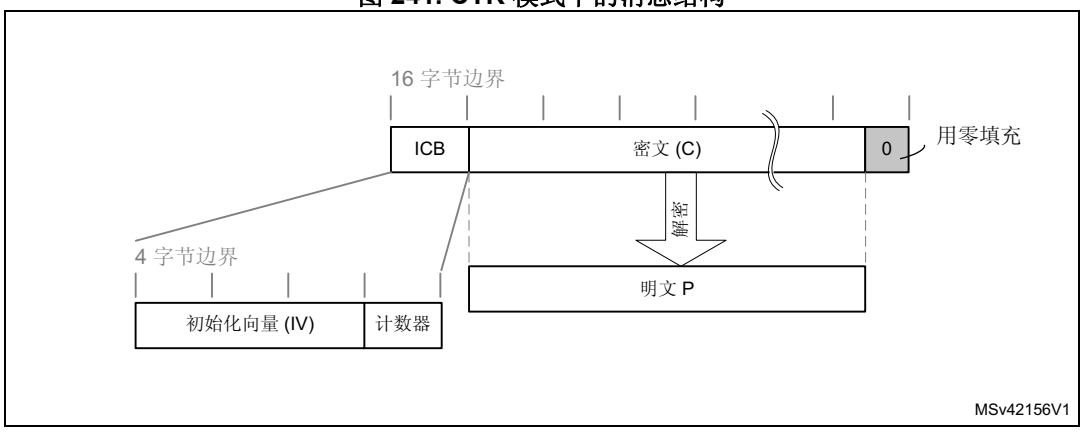
24.4.9 AES 计数器 (CTR) 模式

概述

计数器模式 (CTR) 使用 AES 作为密钥流生成器。然后，生成的密钥与明文进行逻辑异或运算来获得密文。

NIST *特别出版物 800-38A，块密码工作模式的建议*对 CTR 链接进行了相关定义。*图 241* 给出了 CTR 模式下的典型消息结构。

图 241. CTR 模式下的消息结构



该消息的结构为：

- 16 字节的初始计数器块 (ICB)，它由两个不同的字段组成：
  - **初始化向量** (IV)：96 位值，对于给定密钥下每个加密周期，该值都必须唯一。
  - **计数器**：32 位大端模式的整数，随着块的处理次数而递增。应将计数器的初始值设为 1。
- 明文 P 被加密为密文 C（长度已知）。该长度可以不是 16 字节的倍数，在这种情况下需要明文填充。

CTR 加密和解密

图 242 和*图 243* 分别描述了 AES 外设中实现的 CTR 加密和解密过程。通过向 AES_CR 寄存器中的 CHMOD[2:0] 位域写入 010 来选择 CTR 模式。

图 242. CTR 加密

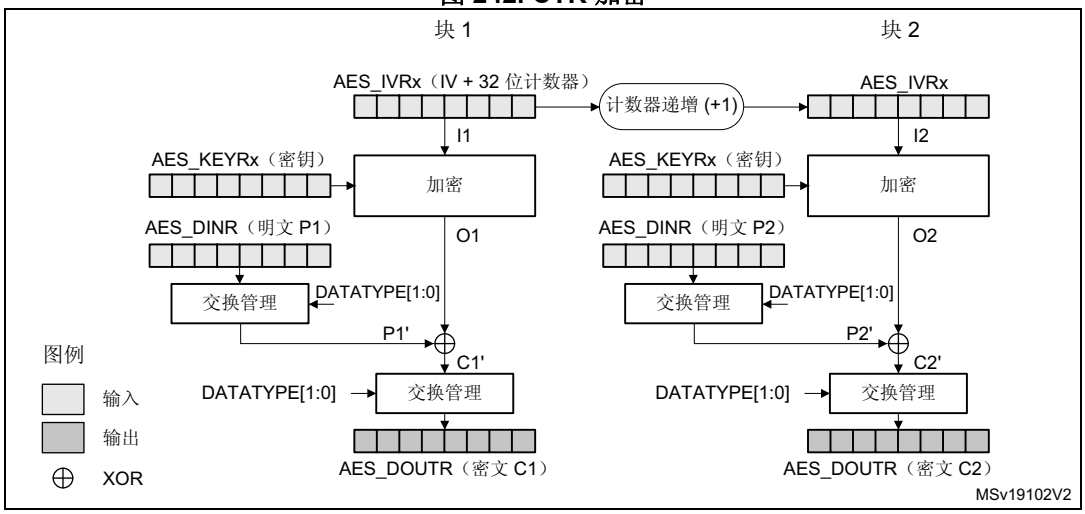
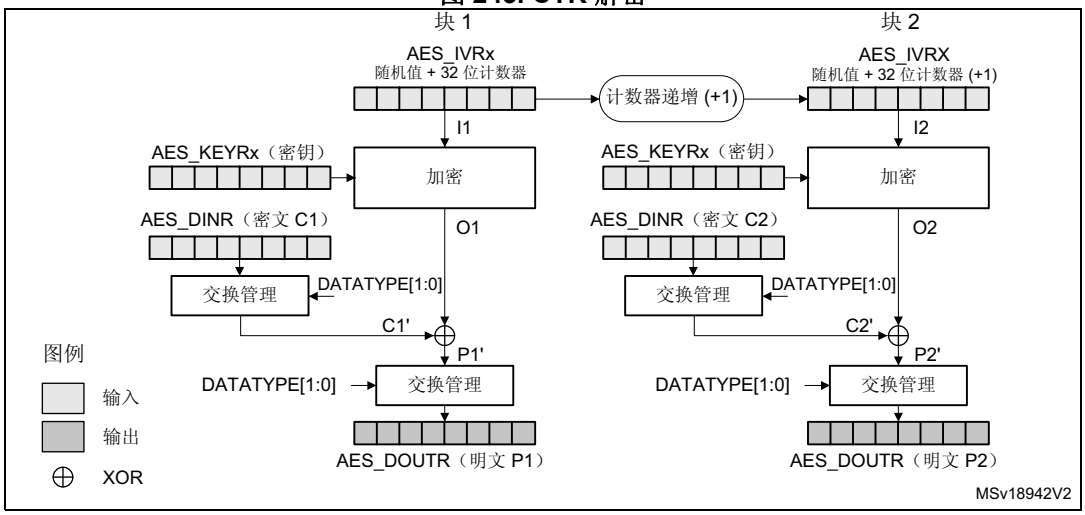


图 243. CTR 解密



在 CTR 模式中，将加密核心输出（也称为密钥流）Ox 与相关输入块（用于加密的 Px'，用于解密的 Cx'）进行逻辑异或运算，以生成正确的输出块（用于加密的 Cx'，用于解密的 Px'）。AES 中的初始化向量必须按*表 125* 所示进行初始化。

表 125. CTR 模式初始化矢量定义

AES_IVR3[31:0]	AES_IVR2[31:0]	AES_IVR1[31:0]	AES_IVR0[31:0]
Nonce[31:0]	Nonce[63:32]	Nonce[95:64]	32 位计数器 = 0x0001

在 CBC 模式下，处理第一个数据块时仅使用 AES_IVRx 寄存器一次，而在 CTR 模式下则有所不同，处理每个数据块都使用 AES_IVRx 寄存器，并且 AES 外设会使初始化向量的计数器位递增（随机值位保持不变）。

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，该密钥数据流与明文（CTR 加密）或密文（CTR 解密）输入进行异或运算。在 CTR 模式下，MODE[1:0] 位域设置 11、10 或 00 将全部默认为加密模式，同时禁止设置 01（密钥生成）。

在 CTR 链接模式下执行加密或解密的事件序列：

1. 确保 AES 禁止（AES_CR 的 EN 位必须为 0）。
2. 通过将 AES_CR 寄存器中的 CHMOD[2:0] 位域设置为 010 来选择 CTR 链接模式。将 MODE[1:0] 位域设置为除 01 之外的任何值。
3. 初始化 AES_KEYRx 寄存器，并按*表 125*所述加载 AES_IVRx 寄存器。
4. 将 AES_CR 寄存器的 EN 位置 1，开始加密当前计数器（计算完成时 EN 自动复位）。
5. 如果是最后一个块，可在需要时用零填充数据以获得完整的块。
6. 在 AES 中附加数据并读取结果。*第 24.4.4 节：用于执行密码操作的 AES 过程*介绍了三种可能的情况。
7. 重复上一步，直到处理完倒数第二个块。对于最后一个块，执行前两步并将不属于有效负载的一部分的位丢弃（如果最后一个输入块中有效数据的大小小于 16 字节）。

CTR 模式下的挂起/恢复操作

该模式与 CBC 模式相似，可以中断消息、发送优先级更高的消息，并可恢复已中断的消息。有关 CBC 挂起/恢复序列的详细信息，请参见*第 24.4.8 节：AES 基本链接模式（ECB 和 CBC）*。

注： *与 CBC 模式类似，恢复操作期间必须重新加载 AES_IVRx 寄存器。*

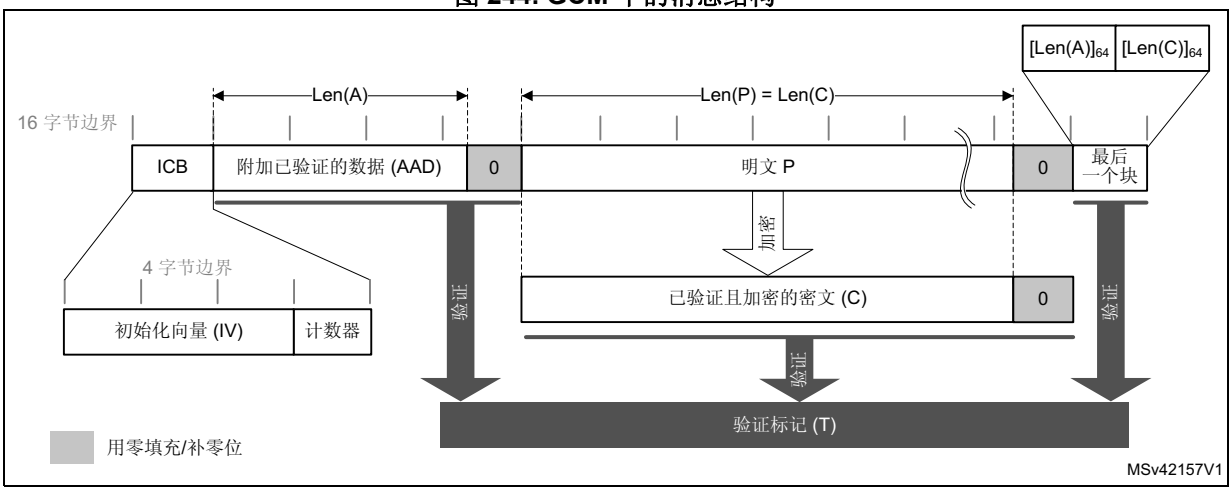
24.4.10 AES Galois/计数器模式 (GCM)

概述

AES Galois/计数器模式 (GCM) 允许将明文消息加密和验证为对应的密文和标记（也称为消息认证码）。为实现保密性，GCM 算法基于 AES 计数器模式。它使用固定有限域乘法器来生成标记。

NIST *特别出版物 800-38D，块密码工作模式的建议 - Galois/计数器模式 (GCM) 和 GMAC* 中对 GCM 链接进行了相关定义。*图 244* 所示为 GCM 模式下的典型消息结构。

图 244. GCM 中的消息结构



消息具有以下结构：

- **16 字节的初始计数器块 (ICB)**，它由两个不同的字段组成：
  - **初始化向量 (IV)**：96 位值，对于给定密钥下每个加密周期，该值都必须唯一。请注意，GCM 标准支持短于 96 位的 IV，在这种情况下应遵循严格的规则。
  - **计数器**：32 位大端模式的整数，随着块的处理次数而递增。根据 NIST 规范，处理有效负载的第一个块时，计数器值为 0x2。

- **认证的标头 AAD**（也称为附加认证数据）具有已知长度 Len(A)，此长度值可以不是 16 字节的倍数，但不能超过 $2^{64} – 1$ 位。消息的这一部分仅经过认证，未被加密。
- **明文消息 P** 将经过认证且被加密为密文 C，其具有已知长度 Len(P)，该长度值可以不是 16 字节的倍数，但不能超过 $2^{32} - 2$ 个 128 位块。
- **最后一个块**包含 AAD 标头长度（位 [32:63]）和有效负载长度（位 [96:127]）信息，如 *表 126* 所示。

GCM 标准规定密文 C 的位长度与明文 P 的位长度相同。

当消息某一部分（AAD 或 P）的长度不是 16 字节的倍数时，需要采取特殊填充方案。

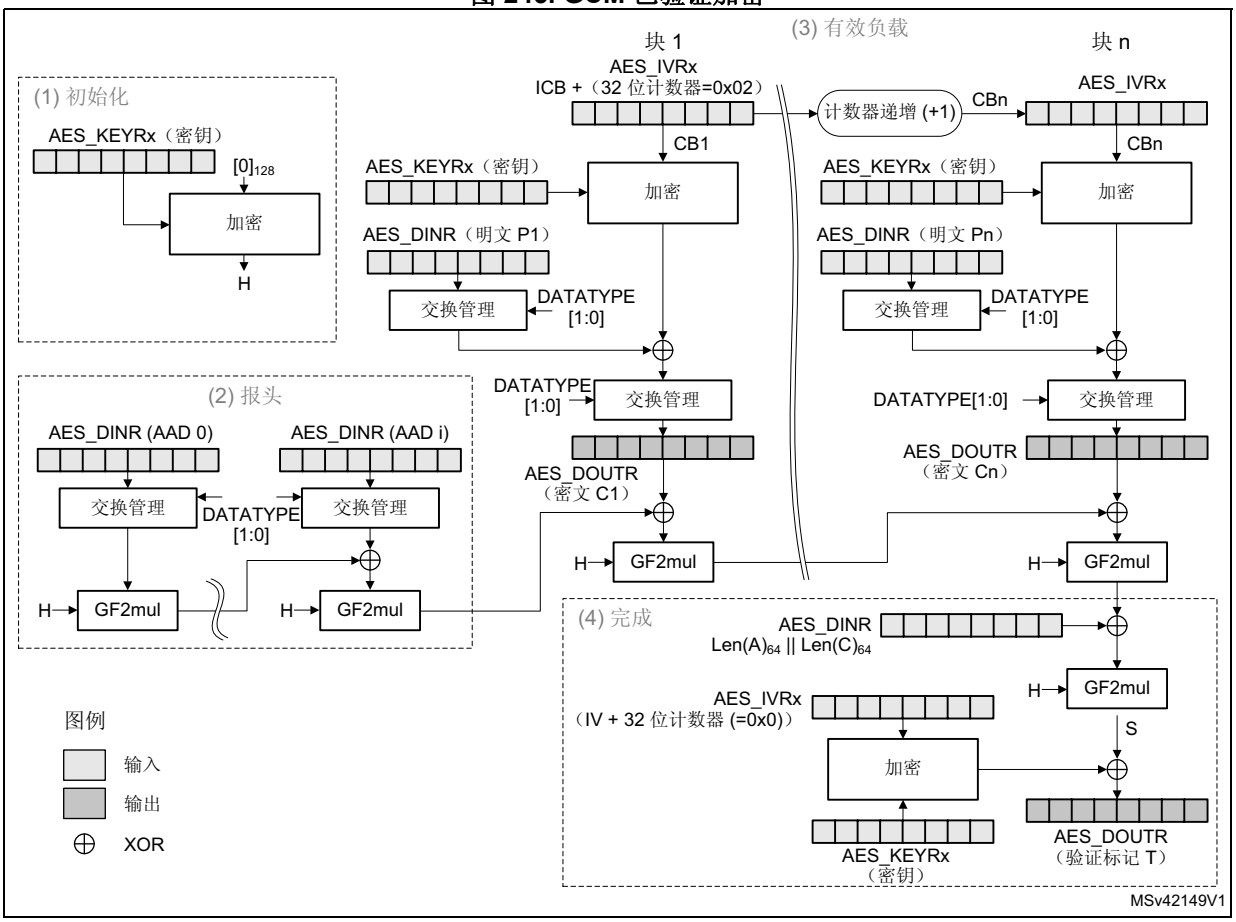
表 126. GCM 最后一个块定义

字节序	位 [0] ---------- 位 [31]	位 [32]---------- 位 [63]	位 [64] -------- 位 [95]	位 [96] --------- 位 [127]
输入数据	0x0	AAD 长度 [31:0]	0x0	有效负载长度 [31:0]

GCM 处理

图 245 列出了 AES 外设中的 GCM 实现。通过向 AES_CR 寄存器中的 CHMOD[2:0] 位域写入 011 来选择 GCM。

图 245. GCM 已验证加密



GCM 模式下明文的保密性机制与计数器模式下的类似，其具有特定递增函数（表示为 32 位递增），可生成输入计数器块序列。

保持数据的**计数器块**的 AES_IVRx 寄存器用于处理每个数据块。AES 外设自动递增计数器 [31:0] 位域。第一个计数器块 (CB1) 由应用软件从初始计数器块 ICB 生成（请参见*表 127*）。

表 127. GCM 模式 IVI 位域初始化

寄存器	AES_IVR3[31:0]	AES_IVR2[31:0]	AES_IVR1[31:0]	AES_IVR0[31:0]
输入数据	ICB[31:0]	ICB[63:32]	ICB[95:64]	Counter[31:0] = 0x2

注： *在 GCM 模式下，禁止将 MODE[1:0] 位域设置为 01 和 11。*

GCM 模式下的认证机制基于散列函数（称为 **GF2mul**），其在二元 Galois 域内执行与固定参数（称为散列子密钥 (H)）的乘法。

GCM 消息通过以下阶段进行处理（将在下一小节中对此进行进一步介绍）：

- **初始化阶段**：AES准备 GCM 散列子密钥 (H)。
- **标头阶段**：AES 处理附加认证数据 (AAD)（仅使用散列计算）。
- **有效负载阶段**：AES 基于散列计算、计数器块加密和数据异或运算处理明文 (P)。其操作方式与密文 (C) 类似。
- **最后阶段**：AES 使用消息的最后一个块生成认证标记 (T)。

GCM 初始化阶段

在此第一步中，将在内部计算和保存 GCM 散列子密钥 (H)，供处理所有块使用。建议序列为：

1. 确保 AES 禁止（AES_CR 的 EN 位必须为 0）。
2. 通过将 AES_CR 寄存器的 CHMOD[2:0] 位域设置为 011 选择 GCM 链接模式，并将 DATATYPE[1:0] 位域设置为 00（无数据交换）。
3. 通过将 AES_CR 寄存器中的 GCMPH[1:0] 位域设置为 00 来指示初始化阶段。
4. 将 AES_CR 寄存器的 MODE[1:0] 位域设置为 00 或 10。
5. 使用密钥初始化 AES_KEYRx 寄存器，使用*表 127* 中定义的信息初始化 AES_IVRx 寄存器。
6. 开始通过将 AES_CR 寄存器的 EN 位设置为 1 来计算散列密钥（计算完成时 EN 自动复位）。
7. 等待直到计算完成，由 AES_SR 的 CCF 标志变为 1 来指示。或者，使用相应的中断。
8. 通过将 AES_CR 寄存器的 CCFC 位设置为 1，将 AES_SR 寄存器的 CCF 标志清零，并且可选择使用 DATATYPE[1:0] 位域设置数据类型（1 位、8 位或 16 位）。

GCM 标头阶段

GCM 初始化阶段之后的这个阶段必须在有效负载阶段之前完成。要执行的序列为（加密与解密完全相同）：

1. 通过将 AES_CR 寄存器中的 GCMPH[1:0] 位域设置为 01 来指示标头阶段。请勿修改初始化阶段中设置的 MODE[1:0] 位域。
2. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
3. 如果它是最后一个块，并且块中的 AAD 大小低于 128 位，则用零填充块的其余部分。然后按*第 660 页的第 24.4.4 节：用于执行密码操作的 AES 过程*中所述的其中一种方式将数据块附加到 AES。
4. 重复步骤 3，直到处理完最后一个附加认证数据块。

注： *如无 AAD（即 Len(A) = 0），则可以跳过标头阶段。*

GCM 有效负载阶段

在 GCM 标头阶段之后执行此阶段（加密和解密相同）。在此阶段，加密/解密的有效负载将存储在 AES_DOUTR 寄存器中。要执行的序列为：

1. 如果跳过了标头阶段，则通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
2. 通过将 AES_CR 寄存器中的 GCMPH[1:0] 位域设置为 10 来指示有效负载阶段。请勿修改初始化阶段中设置的 MODE[1:0] 位域。
3. 如果它是最后一个块，并且块中的明文（加密）或密文（解密）大小低于 128 位，则用零填充块的其余部分。
4. 按*第 660 页的第 24.4.4 节：用于执行密码操作的 AES 过程*中所述的其中一种方式将数据块附加到 AES，并读取结果。
5. 重复上一步，直到倒数第二个明文块被加密或直到最后一个密文块被解密。对于最后一个明文块（仅限加密），执行前两步。对于最后一个块，当其大小小于 16 字节时，将不属于有效负载的一部分的位丢弃。

注： *如无有效负载数据（即 Len(C) = 0），则可以跳过有效负载阶段（请参见 GMAC 模式）。*

GCM 最后阶段

在此最后一个阶段中，AES 外设会生成 GCM 认证标记并将其存储在 AES_DOUTR 寄存器中。要执行的序列为：

1. 通过将 AES_CR 寄存器中的 GCMPH[1:0] 位域设置为 11 来指示最后阶段。通过将 AES_CR 寄存器中的 MODE[1:0] 位域设置为 00 来选择加密模式。
2. 通过连接 AAD 位长度和有效负载位长度来组成块的数据，如*表 126* 中所示。将块写入 AES_DINR 寄存器。
3. 等待直到计算完成，由 AES_SR 的 CCF 标志变为 1 来指示。
4. 对 AES_DOUTR 寄存器进行四次读取操作，以获取 GCM 认证标记。
5. 通过将 AES_CR 寄存器中的 CCFC 位设置为 1，将 AES_SR 寄存器中的 CCF 标志清零。
6. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。如果它是经验证的解密，请将生成的标记与通过消息传递的预期标记进行比较。

注： *在最后阶段，必须根据 AES_CR 寄存器的 DATATYPE[1:0] 位域中设置的数据类型交换数据。*

当从报头或有效负载阶段过渡到最后阶段时，不得禁止 AES 外设，否则结果会是错误的。

GCM 模式下的挂起/恢复操作

要挂起消息处理，请执行以下操作：

1. 如果使用 DMA，则通过将 AES_CR 寄存器中的 DMAINEN 位清零来停止 AES DMA 对 IN FIFO 的数据传输。如果未使用 DMA，确保当前计算已完成，完成由 AES_SR 寄存器的 CCF 标志设置为 1 指示。
2. 在有效负载阶段，如果未使用 DMA，则读取四次 AES_DOUTR 寄存器以保存最后处理的块。如果使用 DMA，等待 AES_SR 寄存器中的 CCF 标志置 1，然后通过将 AES_CR 寄存器中的 DMAOUTEN 位清零来停止 DMA 对 OUT FIFO 的数据传输。
3. 通过将 AES_CR 寄存器中的 CCFC 位设置为 1，将 AES_SR 寄存器中的 CCF 标志清零。
4. 将 AES_SUSPxR 寄存器保存到存储器中，其中 x 等于 0 到 7。
5. 在有效负载阶段，保存 AES_IVRx 寄存器，因为在数据处理过程中，它们相较于初始值发生更改。在标头阶段，则不需要此步骤。
6. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。
7. 将当前 AES 配置保存到存储器中（初始化向量寄存器 AES_IVRx 除外）。无需保存密钥寄存器，因为应用程序已知初始密钥值。
8. 如果使用 DMA，则保存 DMA 控制器状态（指向 IN 数据传输的指针以及剩余字节数等）。在有效负载阶段，也必须保存指向 OUT 数据传输的指针。

要恢复消息处理，请执行以下操作：

1. 如果使用 DMA，则重新配置 DMA 控制器以完成 FIFO IN 传输的其余部分。在有效负载阶段，也必须在 DMA 控制器中配置 FIFO OUT 的其余部分。
2. 确保 AES 外设已禁止（AES_CR 寄存器的 EN 位必须为 0）。
3. 将先前保存在存储器中的挂起寄存器值回写到其相应的 AES_SUSPxR 寄存器中，其中 x 等于 0 到 7。
4. 在有效负载阶段，将先前保存在存储器中的初始化向量寄存器值写回到其相应的 AES_IVRx 寄存器中。在标头阶段，将初始化设置值写回到 AES_IVRx 寄存器中。
5. 恢复 AES_CR 和 AES_KEYRx 寄存器中的初始化设置值。
6. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
7. 如果使用 DMA，将 AES_CR 寄存器中的 DMAINEN 位（和 DMAOUTEN 位，如果在有效负载阶段）置 1，以使能 AES DMA 请求。

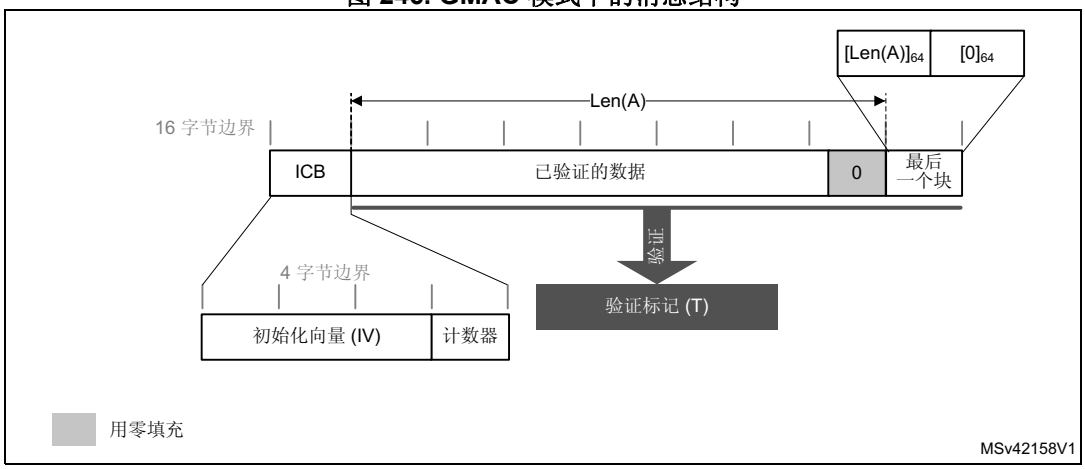
24.4.11 AESGalois 消息认证码 (GMAC)

概述

Galois 消息认证码 (GMAC) 支持认证明文，生成相应的标记信息（也称为消息认证码）。它基于 GCM 算法，NIST *特别出版物 800-38D，块密码工作模式的建议 - Galois/ 计数器模式 (GCM) 和 GMAC* 中对此进行了相关定义。

图 246 给出了 GMAC 下的典型消息结构。

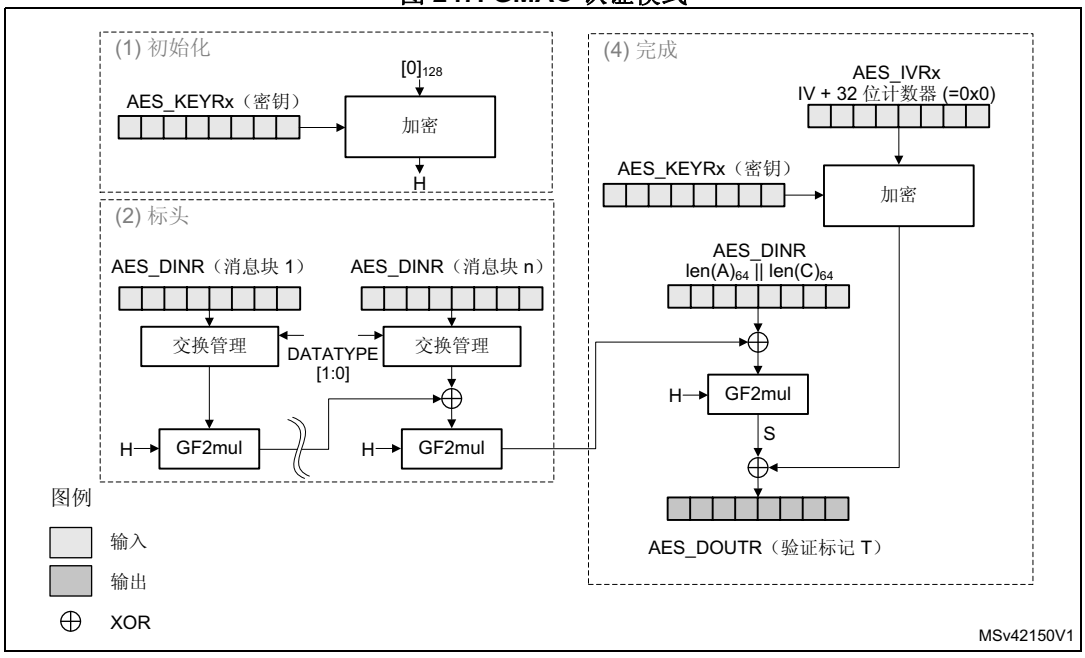
图 246. GMAC 模式下的消息结构



AES GMAC 处理

图 247 列出了 AES 外设中的 GMAC 模式实现。通过向 AES_CR 寄存器中的 CHMOD[2:0] 位域写入 011 来选择此模式。

图 247. GMAC 认证模式



GMAC 算法相当于应用在仅包含标头的消息上的 GCM 算法。因此，除了忽略有效负载阶段之外，所有步骤和设置均与 GCM 相同。

GMAC 模式下的挂起/恢复操作

在 GMAC 模式下，除了只能中断标头阶段外，针对 GCM 描述的序列适用。

24.4.12 AES CBC-MAC 计数器模式 (CCM)

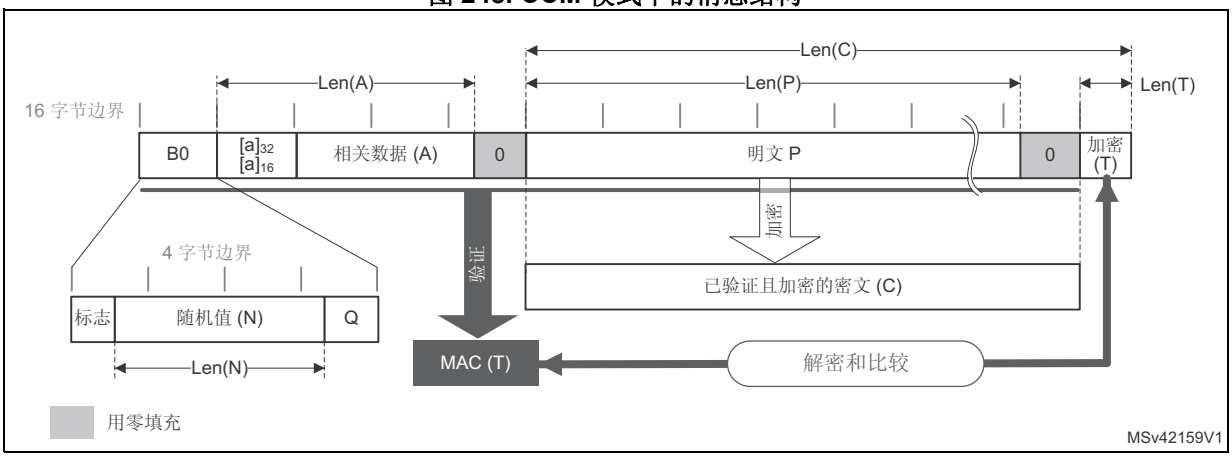
概述

AES **密码块链接-消息认证码计数器模式** (CCM) 算法可用于加密和认证明文，生成对应的密文和标记（也称为消息认证码）。为实现保密性，CCM 算法基于 AES 计数器模式。它使用密码块链接技术来生成消息认证码。这通常被称为 CBC-MAC。

注： *NIST 不允许将该 CBC-MAC 用作 CCM 规范上下文之外的认证模式。*

NIST *特别出版物 800-38C，块密码工作模式的建议 - CCM 模式实现认证和保密性*中对 CCM 链接进行了介绍。*图 248* 给出了 CCM 下的典型消息结构。

图 248. CCM 模式下的消息结构



消息的结构为：

- **16 字节的首次认证块 (B0)**，它由三个不同的字段组成：
  - **Q**：位字符串，表示 P 的八位字节长度 (Len(P))
  - **随机值 (N)**：大小为 Len(N) 的一次性值（即，应为每一次新的通信分配新的随机值）。Len(N) + Len(P) 的和必须等于 15 个字节。
  - **标志**：最高有效八位字节，包含四个控制信息标志（根据标准规定）。它包含两个 3 位字符串，用于编码值 **t**（MAC 长度，以字节表示）和 **Q**（明文长度，例如 Len(P) < 2^{8q} 个字节）。与 **Q** 相关的计数器块范围为 2^{8Q-4}，即，如果 **Q** 的最大值为 8，则密码中使用的计数器块应为 60 位。
- 与相关数据 (A) 关联的 **16 字节块 (B)**。
  消息的这一部分仅经过认证，未被加密。这部分具有已知长度 Len(A)，此长度值可以不是 16 字节的倍数（请参见*图 248*）。标准还具有以下规定：对于第一个消息块 (B1) 的 MSB 位，以字节表示的相关数据长度 (a) 必须按如下所述进行编码：
  - 如果 $0 < a < 2^{16} - 2^8$，则将其编码为 $[a]_{16}$，即两个字节。
  - 如果 $2^{16} - 2^8 < a < 2^{32}$，则将其编码为 0xff || 0xfe || $[a]_{32}$，即六个字节。
  - 如果 $2^{32} < a < 2^{64}$，则将其编码为 0xff || 0xff || $[a]_{64}$，即十个字节。
- 与明文消息 P 相关的 **16 字节块 (B)**，此明文消息将经过认证并被加密为密文 C（具有已知长度 Len(P)）。此长度可以不是 16 字节的倍数（请参见*图 248*）。
- 长度为 Len(T) 的**加密 MAC (T)** 被附加到总长度为 Len(C) 的密文 C。

当消息某一部分（A 或 P）的长度不是 16 字节的倍数时，需要采取特殊填充方案。

注： *CCM 链接模式同样只能与相关数据搭配使用（即，无有效负载）。*

例如，NIST 特别出版物 800-38C 的 C.1 部分给出了以下值（十六进制数）：

N：10111213 141516（Len(N)= 56 位或 7 字节）

A：00010203 04050607（Len(A)= 64 位或 *8* 字节）

P：**20212223**（Len(P)= 32 位或 **4** 字节）

T：6084341B（Len(T)= 32 位或 t = 4）

B0：4F101112 13141516 00000000 0000000**4**

B1：*0008***0001 *02030405 *0607**0000 00000000

B2：**20212223** 00000000 00000000 00000000

CTR0：07*10111213 141516* 00000000 00000000

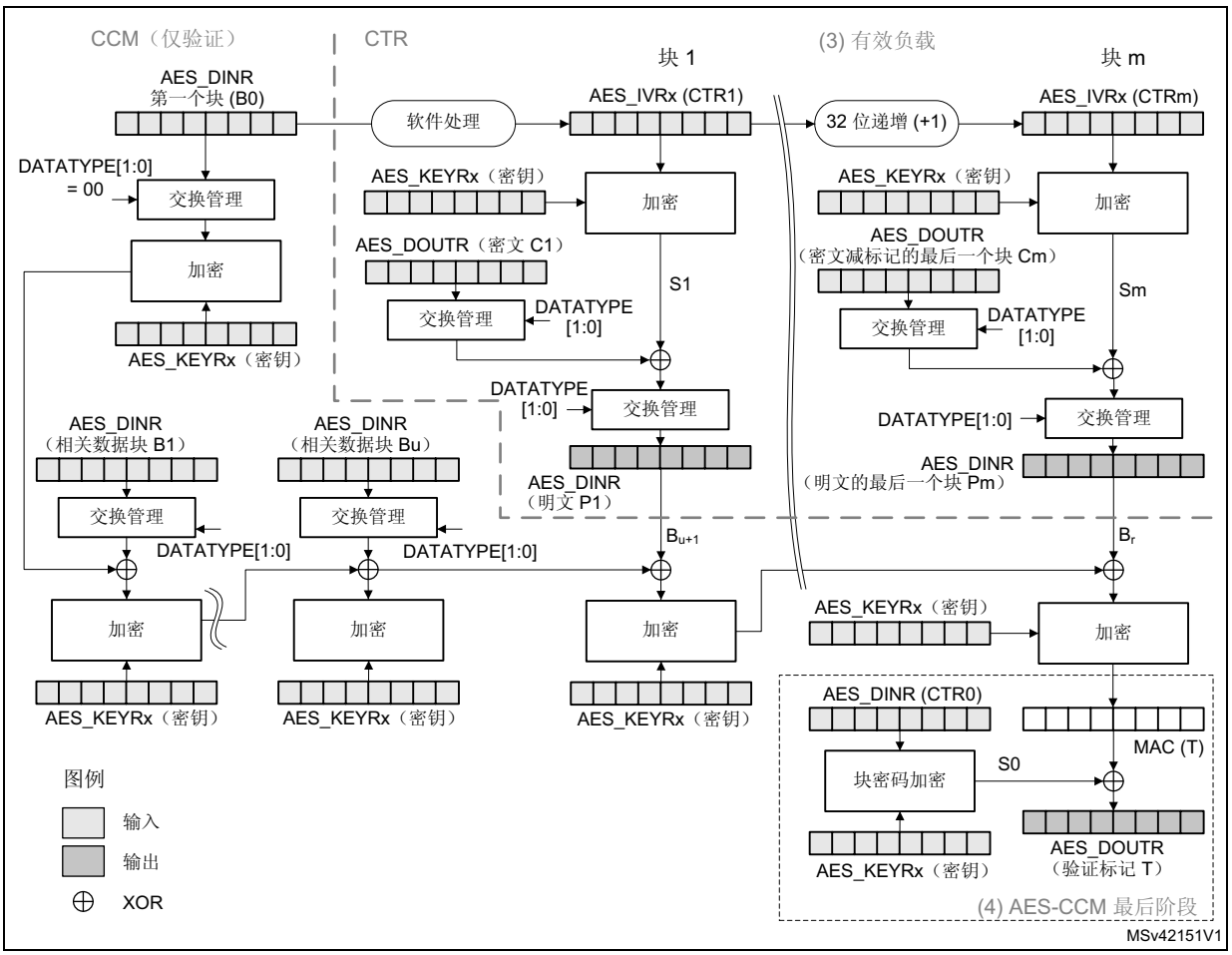
CTR1：07*10111213 141516* 00000000 00000001

格式化输入数据块 Bx（特别是 B0 和 B1）的生成必须由应用程序管理。

CCM 处理

图 249 列出了 AES 外设中的 CCM 实现（解密示例）。

图 249. CCM 模式认证解密



生成-加密过程的数据输入包括一个有效随机值、一个有效的有效负载字符串和一个有效的相关数据字符串，这些数据均经过正确格式化。对经过格式化的明文数据应用 CBC 链接机制可生成长度已知的 MAC。计数器模式加密需要足够长的计数器块序列作为输入，其将应用于有效负载字符串并单独应用于 MAC。得到的密文 C 是明文 P 的生成-加密过程的输出。

AES_IVRx 寄存器用于处理每个数据块，AES 以第一个块 B0 定义的位长度自动使 CTR 计数器递增。*表 128* 显示了应用程序必须如何加载 B0 数据。

表 128. 在 CCM 模式下初始化 AES_IVRx 寄存器

寄存器	AES_IVR3[31:0]	AES_IVR2[31:0]	AES_IVR1[31:0]	AES_IVR0[31:0]
输入数据	B0[31:0]	B0[63:32]	B0[95:64]	B0[127:96]

CCM 消息通过两个不同的过程进行处理 - 首先是**有效负载加密或解密**，此时会在 CTR 模式下配置 AES 外设，然后是**相关数据和有效负载认证**，其中 AES 外设首先执行 CCM 标头阶段，然后执行 CCM 最后阶段。

有效负载加密/解密

该步骤独立于标记计算执行。它使用标准 CTR 链接模式。有关详细信息，请参见*第 24.4.9 节：AES 计数器 (CTR) 模式*。在 NIST 特别出版物 800-38C 中对加载到 AES_IVRx 寄存器的 CTR1 初始化向量的结构（请参见*图 249*）进行了相关定义。

注： *如果没有有效负载数据，即当 Len(P) = 0 或 Len(C) = Len(T) 时，可以跳过此阶段。*

解密密文 C 时，移除 $LSB_{Len(T)}(C)$ 加密标记信息。

相关数据和有效负载认证

为了计算与明文消息相关的 CCM 认证标记，建议执行以下标头阶段序列：

1. 确保 AES 外设已禁止（AES_CR 的 EN 位必须为 0）。
2. 通过将 AES_CR 寄存器的 CHMOD[2:0] 位域设置为 100 选择 CCM 链接模式，设置 DATATYPE[1:0] 位域。
3. 通过将 AES_CR 寄存器中的 GCMPH[1:0] 位域设置为 01 来指示标头阶段。通过将 AES_CR 寄存器中的 MODE[1:0] 位域设置为 00 来选择加密模式。
4. 使用密钥初始化 AES_KEYRx 寄存器，使用零值初始化 AES_IVRx 寄存器。
5. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
6. 如*表 128* 中所示，将 B0 写入 AES_DINR 寄存器。必须根据 AES_CR 寄存器的 DATATYPE[1:0] 位域交换 B0 数据。
7. 等待 AES_SR 寄存器的计算完成标志 CCF 置 1。
8. 通过将 AES_CR 寄存器中的 CCFC 位置 1，将 AES_SR 寄存器中的 CCF 标志清零。
9. 处理数据块。如果它是相关数据或明文的最后一个块，并且块中的数据大小低于 128 位，则用零填充块的其余部分。然后按*第 660 页的第 24.4.4 节：用于执行密码操作的 AES 过程*中所述的其中一种方式将数据块附加到 AES。
10. 重复上一步来处理所有数据块，从相关数据的第一个块开始，到明文有效负载数据的最后一个块结束。

在最后阶段中，AES 外设会生成 CCM 认证标记并将其存储在 AES_DOUTR 寄存器中：

11. 通过将 AES_CR 寄存器中的 GCMPH[1:0] 位域设置为 11 来指示最后阶段。保持 MODE[1:0] 位域中的加密模式不变。
12. 将最后一个数据输入写入 AES_DIN 寄存器四次。该输入必须是从原始 B0 数据包格式化的 128 位值 CTR0（即，5 个标志位设置为 0，Q 长度位设置为 0）。
13. 等待 AES_SR 寄存器的计算完成标志 CCF 置 1。
14. 读取 AES_DOUTR 寄存器四次：此输出即为加密的 CCM 认证标记。
15. 通过将 AES_CR 寄存器中的 CCFC 位置 1，将 AES_SR 寄存器中的 CCF 标志清零。
16. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。
17. 对于认证解密，将生成的加密标记与密文中填充的加密标记进行比较。

注： *在此最后阶段，必须根据 AES_CR 寄存器的 DATATYPE[1:0] 位域交换数据。*

当从报头阶段过渡到最后阶段时，不得禁止 AES 外设，否则结果会是错误的。

应用程序必须使用标记长度屏蔽认证标记输出以获取有效标记。

CCM 模式下的挂起/恢复操作

要挂起相关数据和有效负载的认证 (GCMPH[1:0]= 01)，请执行以下操作。*第 670 页的第 24.4.9 节：AES 计数器 (CTR) 模式*对在加密/解密阶段挂起消息进行了说明。

1. 如果使用 DMA，则通过将 AES_CR 寄存器中的 DMAINEN 位清零来停止 AES DMA 对 IN FIFO 的数据传输。如果未使用 DMA，确保当前计算已完成，完成由 AES_SR 寄存器的 CCF 标志设置为 1 指示。
2. 通过将 AES_CR 寄存器中的 CCFC 位设置为 1，将 AES_SR 寄存器中的 CCF 标志清零。
3. 将 AES_SUSPxR 寄存器（其中 x 等于 0 到 7）保存到存储器中。
4. 保存 AES_IVRx 寄存器，因为在数据处理过程中，它们相较于初始值发生更改。
5. 通过将 AES_CR 寄存器中的 EN 位清零来禁止 AES 外设。
6. 将当前 AES 配置保存到存储器中（初始化向量寄存器 AES_IVRx 除外）。无需保存密钥寄存器，因为应用程序已知初始密钥值。
7. 如果使用 DMA，则保存 DMA 控制器状态（指向 IN 数据传输的指针以及剩余字节数等）。

要恢复相关数据和有效负载的认证 (GCMPH[1:0]= 01 或 11)，请执行以下操作：

1. 如果使用 DMA，则重新配置 DMA 控制器以完成 FIFO IN 传输的其余部分。
2. 确保 AES 处理器已禁止（AES_CR 寄存器的 EN 位必须为 0）。
3. 将先前保存在存储器中的挂起寄存器值回写到其相应的 AES_SUSPxR 寄存器中（其中 x 等于 0 到 7）。
4. 将先前保存在存储器中的初始化向量寄存器值写回到其相应的 AES_IVRx 寄存器中。
5. 恢复 AES_CR 和 AES_KEYRx 寄存器中的初始化设置值。
6. 通过将 AES_CR 寄存器中的 EN 位置 1 来使能 AES 外设。
7. 如果使用 DMA，将 AES_CR 寄存器中的 DMAINEN 位置 1，以使能 AES DMA 请求。

注： *在 CCM 模式下，禁止将 MODE[1:0] 位域设置为 01 和 11（密钥生成）。*

24.4.13 .AES 数据寄存器和数据交换

数据输入和输出

通过将四个连续的 32 位字写入 AES_DINR 寄存器（位域 DIN[127:0]），将 128 位数据块输入到 AES 外设，先最高有效字（位 [127:96]），后最低有效字（位 [31:0]）。

通过从 AES_DOUTR 寄存器（位域 DOUT[127:0]）读取四个连续的 32 位字，从 AES 外设检索 128 位数据块，先最高有效字（位 [127:96]），后最低有效字（位 [31:0]）。

写入 AES_DINR 寄存器或从 AES_DOUTR 寄存器读取的 32 位数据字使用大端模式进行组织，即：

- 要写入 AES_DINR 的字的最高有效字节必须存储在保留要写入的字的四个相邻存储单元的最低地址中，或
- 从 AES_DOUTR 读取的字的最高有效字节存储在接收字的四个相邻存储单元的最低地址中

要使用 DMA 将输入数据块写入 AES，输入块的四个字必须以大端模式连续存储在存储器中，即最高有效字存储在最低地址中。请参见*第 24.4.16 节：AES DMA 接口*。

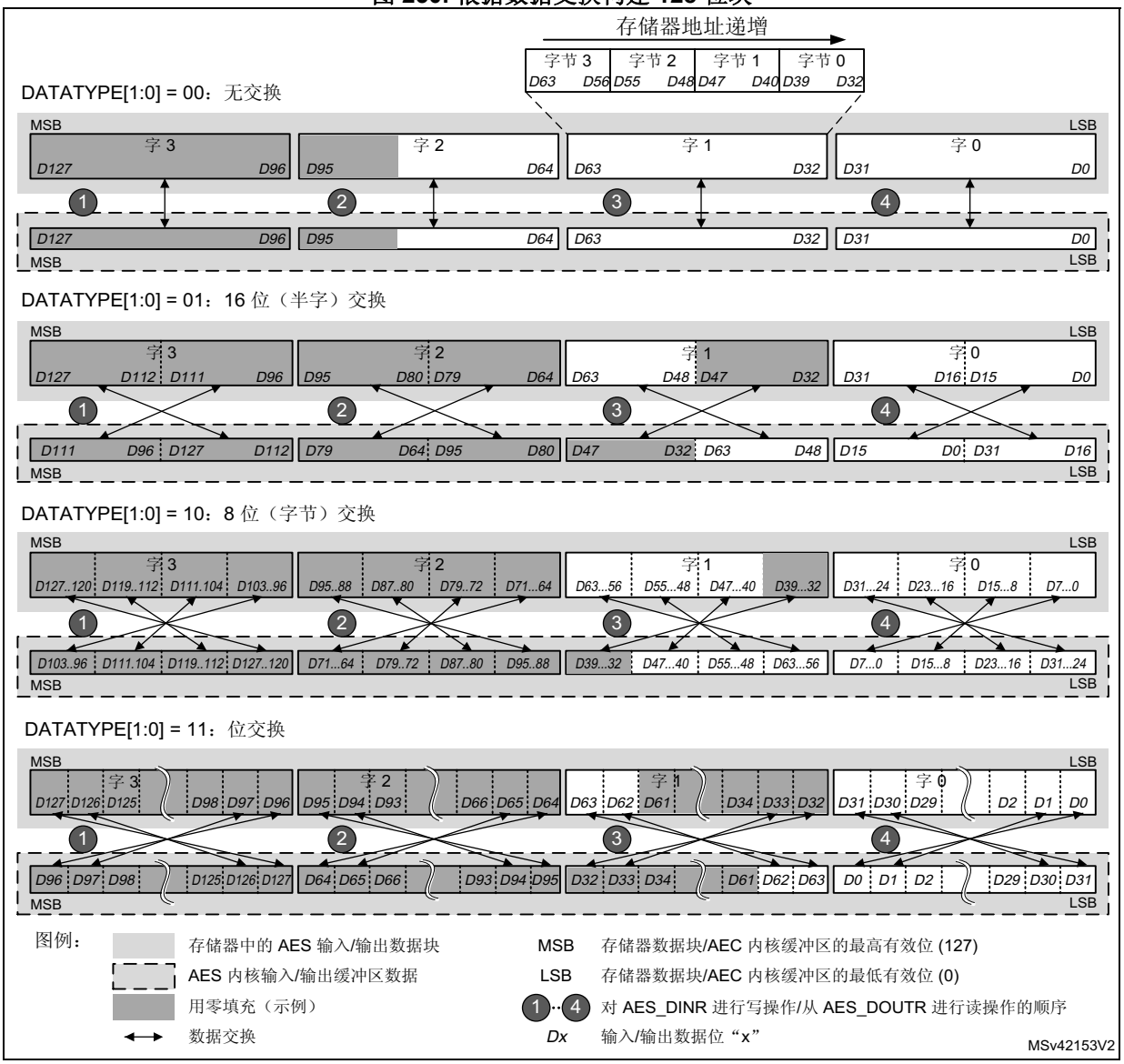
数据交换

可将 AES 外设配置为在将其加载到 AES 处理内核前对 AES_DINR 寄存器中的输入数据字，以及在将其发送到 AES_DOUTR 寄存器前对 AES 处理内核的输出数据，执行位、字节、半字或无交换。选择取决于数据类型。例如，对 ASCII 文本流使用字节交换。

通过 AES_CR 寄存器中的 DATATYPE[1:0] 位域来选择数据交换类型。该选择适用于 AES 内核的输入和输出。

对于不同的数据交换类型，*图 250* 显示了通过 AES_DINR 寄存器输入的输入数据的 AES 处理内核输入缓冲区数据 P127..0 的结构，或通过 AES_DOUTR 寄存器的输出数据的 AES 处理内核输出缓冲区数据 P127..0 的结构。

图 250. 根据数据交换构建 128 位块



注： *AES 密钥寄存器 (AES_KEYRx) 和初始化寄存器 (AES_IVRx) 中的数据对所选交换模式不敏感。*

数据填充

图 250 还给出了一个用零填充存储器数据块的示例，这样数据交换后的补零位在 AES 内核输入缓冲区的 MSB 端形成一个连续区域。该示例显示了输入数据块的填充，包含：

- 48 个消息位，DATATYPE[1:0] = 01
- 56 个消息位，DATATYPE[1:0] = 10
- 34 个消息位，DATATYPE[1:0] = 11

24.4.14 AES 密钥寄存器

AES_KEYRx 寄存器存储加密或解密密钥位域 KEY[127:0] 或 KEY[255:0]。要写入每个寄存器或从中读取的数据以小端模式组织在存储器中，即最高有效字节存储在最高地址中。

密钥分布在八个寄存器中，如*表 129* 中所示。

表 129. AES_KEYRx 寄存器中的密钥字节序（128 位或 256 位密钥长度）

AES_KEYR7 [31:0]	AES_KEYR6 [31:0]	AES_KEYR5 [31:0]	AES_KEYR4 [31:0]	AES_KEYR3 [31:0]	AES_KEYR2 [31:0]	AES_KEYR1 [31:0]	AES_KEYR0 [31:0]
-	-	-	-	KEY[127:96]	KEY[95:64]	KEY[63:32]	KEY[31:0]
KEY[255:224]	KEY[223:192]	KEY[191:160]	KEY[159:128]	KEY[127:96]	KEY[95:64]	KEY[63:32]	KEY[31:0]

AES 外设禁止时，加密或解密的密钥可以写入这些寄存器。

密钥寄存器不受由 AES_CR 寄存器的 DATATYPE[1:0] 位域控制的数据交换影响。

24.4.15 AES 初始化向量寄存器

四个 AES_IVRx 寄存器保持初始化向量输入位域 IVI[127:0]。要写入每个寄存器或从中读取的数据以小端模式组织在存储器中，即最高有效字节存储在最高地址中。寄存器也以最低地址 (AES_IVR0) 到最高地址 (AES_IVR3) 的顺序排序。

位域中数据的有效性取决于选择的链接模式。使用时，位域在 AES 内核的每个计算周期内都会更新。

AES 外设使能时，对 AES_IVRx 寄存器的写入操作对寄存器内容没有影响。要修改 AES_IVRx 寄存器的内容，必须首先清零 AES_CR 寄存器的 EN 位。

读取 AES_IVRx 寄存器会返回最新计数器值（用于管理挂起模式）。

AES_IVRx 寄存器不受由 CRYP_CR 寄存器的 DATATYPE[1:0] 位域控制的数据交换功能影响。

24.4.16 AES DMA 接口

AES 外设使用一个接口连接 DMA（直接存储器访问）控制器。DMA 操作通过 AES_CR 寄存器进行控制。

使用 DMA 的数据输入

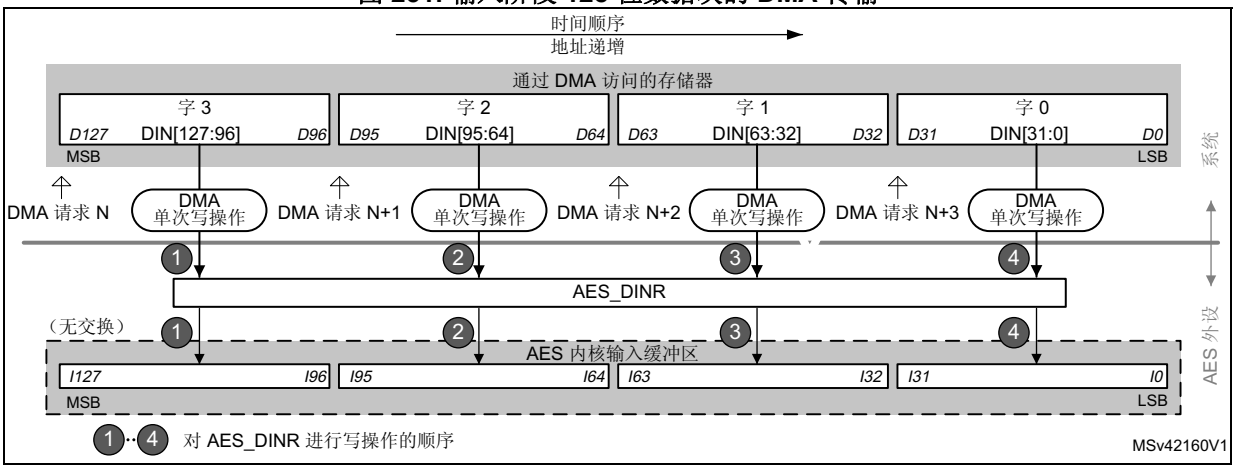
通过将 AES_CR 寄存器的 DMAINEN 位置 1，可使能 DMA 写入 AES。然后，AES 外设每次需要向 AES_DINR 寄存器中写入字时，都会在输入阶段发出 DMA 请求。它使能四个 DMA 请求，传输存储器的一个 128 位（四字）输入数据块，如*图 251* 中所示。

有关推荐的 DMA 配置，请参见*表 130*。

表 130. 用于存储器到 AES 数据传输的 DMA 通道配置

DMA 通道控制寄存器字段	推荐配置
传输数据量大小	消息长度：128 位的倍数。 根据所选算法和模式，可能需要采用特殊填充/密文窃取技术。有关详细信息，请参见*第 664 页的第 24.4.6 节：AES 密文窃取和数据填充*。
源突发大小（存储器）	单
目标突发大小（外设）	单
DMA FIFO 大小	AES FIFO_size = 4 个字节
源传输宽度（存储器）	32 位字
目标传输宽度（外设）	32 位字
源地址递增（存储器）	是，每次 32 位传输后
目标地址递增（外设）	AES_DINR 的固定地址（无递增）

图 251. 输入阶段 128 位数据块的 DMA 传输



使用 DMA 的数据输出

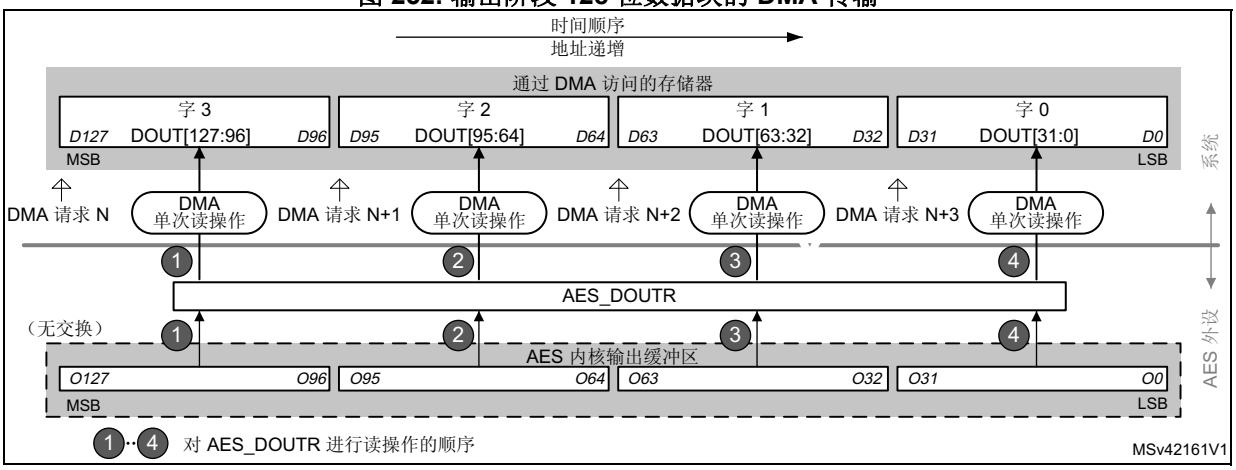
通过将 AES_CR 寄存器的 DMAOUTEN 位置 1，可使能 DMA 从 AES 读取。然后，AES 外设每次需要从 AES_DOUTR 寄存器中读取字时，都会在输出阶段发出 DMA 请求。它使能四个 DMA 请求，将一个 128 位（四字）输出数据块传输到存储器，如*图 252* 中所示。

有关推荐的 DMA 配置，请参见*表 131*。

表 131. 用于 AES 到存储器数据传输的 DMA 通道配置

DMA 通道控制寄存器字段	推荐配置
传输数据量大小	消息长度，AES 块大小（4 个字）的倍数。根据情况，需要丢弃额外的字节。
源突发大小（外设）	单
目标突发大小（存储器）	单
DMA FIFIO 大小	AES FIFO_size = 4 个字节
源传输宽度（外设）	32 位字
目标传输宽度（存储器）	32 位字
源地址递增（外设）	AES_DINR 的固定地址（无递增）
目标地址递增（存储器）	是，每次 32 位传输后

图 252. 输出阶段 128 位数据块的 DMA 传输



不同工作模式下的 DMA 操作

当通过 AES_CR 寄存器的 MODE[1:0] 位域选择模式 1（加密）或模式 3（解密）时，DMA 操作可用。在模式 2（密钥生成）下，AES_KEYRx 寄存器必须通过软件写入，通过 AES_CR 寄存器的 DMAINEN 和 DMAOUTEN 位使能 DMA 传输在该模式下无效。

AES 会一直产生 DMA 单次请求，直到被禁用。因此，128 位数据块处理结束时的数据输出阶段之后，AES 会自动切换到下一个数据块的新数据输入阶段（如果存在）。

当 AES 和存储器之间的数据传输由 DMA 管理时，CCF 标志无意义，可通过软件忽略（置 1）。只有在转换回由软件管理的数据传输时才能清零此标志。请参见*第 24.4.8 节：AES 基本链接模式（ECB 和 CBC）*中的*ECB/CBC 模式下的挂起/恢复操作*作为示例。

24.4.17 AES 错误管理

如果检测到未预期的读取或写入操作，则会将 AES_SR 寄存器中的读取错误标志 (RDERR) 和写入错误标志 (WRERR) 位置 1。如果 AES_CR 寄存器中的错误中断使能 (ERRIE) 位置 1，则会产生中断。有关详细信息，请参见*第 24.5 节：AES 中断*。

注： *检测到错误后，AES 不会禁止，而是会继续处理。*

可随时通过将 AES_CR 寄存器中的 EN 位清零再置 1 来重新初始化 AES。

读取错误标志 (RDERR)

如果在计算阶段或输入阶段检测到未预期的读取操作时，则会将 AES_SR 寄存器中的 AES 读取错误标志 (RDERR) 置 1。如果 AES_CR 寄存器中的 ERRIE 位置 1，则会产生中断。

通过将 AES_CR 寄存器中的相应 ERRC 位置 1 来清零 RDERR 标志。

写入错误标志 (WDERR)

如果在计算阶段或输出阶段检测到未预期的写入操作时，则会将 AES_SR 寄存器中的 AES 写入错误标志 (WRERR) 置 1。如果 AES_CR 寄存器中的 ERRIE 位置 1，则会产生中断。

通过将 AES_CR 寄存器中的相应 ERRC 位置 1 来清零 WDERR 标志。

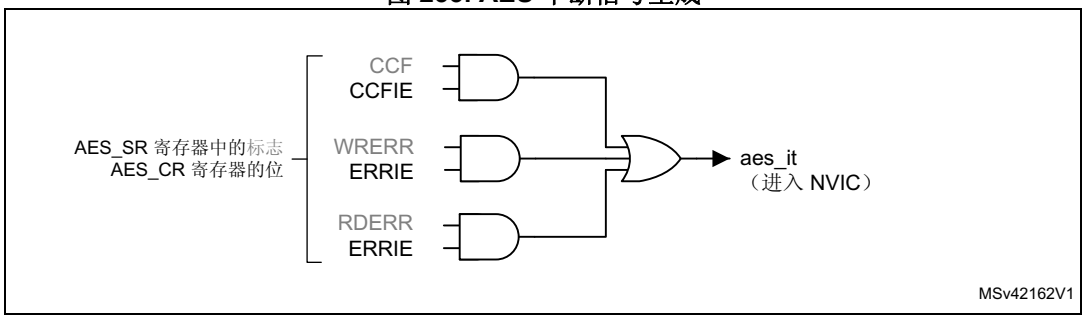
24.5 AES 中断

AES 外设可生成三个独立的可屏蔽中断源，用以通知发生以下事件：

- 计算完成
- 读取错误，请参见*第 24.4.17 节*
- 写入错误，请参见*第 24.4.17 节*

这三个中断源合并为一个连接到 NVIC（嵌套向量中断控制器）的通用中断信号 aes_it。

图 253. AES 中断信号生成



可通过将 AES_CR 寄存器的相应使能位置 1/清零，分别使能/禁止每个 AES 中断源。请参见*图 253*。

可以从 AES_SR 寄存器中读取各个可屏蔽中断源的状态。

表 132 对中断源及其事件标志和使能位进行了总结。

表 132. AES 中断请求

AES 中断事件	事件标志	使能位
计算完成标志	CCF	CCFIE
读取错误标志	RDERR	ERRIE
写入错误标志	WRERR	ERRIE

24.6 AES 处理延迟

下表列出了每种工作模式下处理 128 位块的延迟。

表 133. ECB、CBC 和 CTR 模式下的处理延迟（以时钟周期计）

密钥大小	工作模式	算法	输入阶段	计算阶段	输出阶段	总计
128 位	模式 1：加密	ECB、CBC、CTR	8	202	4	**214**
	模式 2：密钥生成	-	-	80	-	**80**
	模式 3：解密	ECB、CBC、CTR	8	202	4	**214**
	模式 4：密钥生成和解密	ECB、CBC	8	276	4	**288**
256 位	模式 1：加密	ECB、CBC、CTR	8	286	4	**298**
	模式 2：密钥生成	-	-	109	-	**109**
	模式 3：解密	ECB、CBC、CTR	8	286	4	**298**
	模式 4：密钥生成和解密	ECB、CBC	8	380	4	**392**

表 134. GCM 和 CCM 模式下的处理延迟（以时钟周期计）

密钥大小	工作模式	算法	初始化阶段	标头阶段	有效负载阶段	标记阶段
128 位	模式 1：加密 / 模式 3：解密	GCM	215	67	202	202
	-	CCM 认证	-	206	-	202
256 位	模式 1：加密 / 模式 3：解密	GCM	299	67	286	286
	-	CCM 认证	-	290	-	286

注： *数据插入可以包括 AES 在 AHB 总线上强制的等待周期（最多 3 个周期，通常为 1 个周期）。这适用于所有标头/有效负载/标记阶段。*

24.7 AES 寄存器

24.7.1 AES 控制寄存器 (AES_CR)

AES 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.	KEYSIZE	Res.	CHMOD[2]
													rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	GCMPH[1:0]		DMAOUTEN	DMAINEN	ERRIE	CCFIE	ERRC	CCFC	CHMOD[1:0]		MODE[1:0]		DATATYPE[1:0]		EN
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:19 保留，必须保持为零

位 18 **KEYSIZE**：密钥大小选择 (Key size selection)
此位域定义了 AES 加密内核中使用的密钥长度（以位为单位）。
0: 128
1: 256

仅当 AES 禁止时才允许更改位值，以避免不可预测的行为。

位 17 保留，必须保持为零

位 16 **CHMOD[2]**：链接模式选择，位 [2] (Chaining mode selection, bit [2])
有关 CHMOD[2:0] 位域的说明，请参见寄存器的位 [5:6]

位 15 保留，必须保持为零

位 14:13 **GCMPH[1:0]**：GCM 或 CCM 阶段选择 (GCM or CCM phase selection)
该位域选择 GCM、GMAC 或 CCM 算法的阶段：
00：初始化阶段
01：标头阶段
10：有效负载阶段
11：最后阶段

如果选择 GCM、GMAC 或 CCM 以外的算法（通过 ALGOMODE 位域），则该位域不起作用。

位 12 **DMAOUTEN**：DMA 输出使能 (DMA output enable)
该位在输出阶段使能/禁止使用 DMA 进行数据传输：
0：禁止
1：使能

该位置 1 时，AES 会在输出数据阶段自动生成 DMA 请求。该功能仅在通过 MODE[1:0] 位域选择了模式 1 或模式 3 时才有效。它不适用于模式 2（密钥生成）。
不建议在模式 4 下（单次解密）使用 DMA。

位 11 **DMAINEN**：DMA 输入使能 **(DMA input enable)**
该位在输入阶段使能/禁止使用 DMA 进行数据传输：
0：禁止
1：使能

该位置 1 时，AES 会在输入数据阶段自动生成 DMA 请求。该功能仅在通过 MODE[1:0] 位域选择了模式 1 或模式 3 时才有效。它不适用于模式 2（密钥生成）。
不建议在模式 4 下（单次解密）使用 DMA。

位 10 **ERRIE**：错误中断使能 (Error interrupt enable)
当 RDERR 和/或 WRERR 置 1 时，该位使能或禁止（屏蔽）AES 中断：
0：禁止（屏蔽）
1：使能

位 9 **CCFIE**：CCF 中断使能 (CCF interrupt enable)
当 CCF（计算完成标志）置 1 时，该位使能或禁止（屏蔽）AES 中断：
0：禁止（屏蔽）
1：使能

位 8 **ERRC**：错误标志清零 (Error flag clear)

写入 1 后，该位将 AES_SR 寄存器中的 RDERR 和 WRERR 错误标志清零：
0：不起作用
1：清零 RDERR 和 WRERR 标志

读取标志始终返回零。

位 7 **CCFC**：计算完成标志清零 (Computation complete flag clear)

写入 1 后，该位将 AES_SR 寄存器中的计算完成标志 (CCF) 清零：
0：不起作用
1：清零 CCF

读取标志始终返回零。

位 6:5 **CHMOD[1:0]**：链接模式选择，位 [1:0](Chaining mode selection, bits [1:0])

这些位与位 CHMOD[2]（请参见此寄存器的位 16）一起构成 CHMOD[2:0] 位域，用于选择 AES 链接模式：
000：电子密码本 (ECB)
001：密码块链接 (CBC)
010：计数器模式 (CTR)
011：Galois 计数器模式 (GCM) 和 Galois 消息认证码 (GMAC)
100：CBC-MAC 计数器模式 (CCM)
>100：保留

仅当 AES 禁止时才允许更改位域值，以避免不可预测的行为。

位 4:3 **MODE[1:0]**：AES 工作模式 (AES operating mode)

此位域选择 AES 工作模式：
00：模式 1：加密
01：模式 2：密钥生成（或针对 ECB/CBC 解密准备密钥）
10：模式 3：解密
11：模式 4：密钥生成和单次解密

仅当 AES 禁止时才允许更改位域值，以避免不可预测的行为。在未选择 ECB 或 CBC 链接模式时选择模式 4 的任何尝试，都将默认为有效选择模式 3。在模式 4 之后不能选择模式 3。

位 2:1 **DATATYPE[1:0]**：数据类型选择 (Data type selection)

此位域通过选择数据交换模式，定义了写入 AES_DINR 寄存器或从 AES_DOUTR 寄存器中读取的数据的格式：
00：无
01：半字（16 位）
10：字节（8 位）
11：位

有关详细信息，请参见*第 24.4.13 节：.AES 数据寄存器和数据交换*。
仅当 AES 禁止时才允许更改位域值，以避免不可预测的行为。

位 0 **EN**：AES 使能 (AES enable)

该位用于使能/禁止 AES 外设：
0：禁止
1：使能

可随时通过将该位清零再置 1 来重新初始化 AES 外设。
当密钥准备过程结束时（模式 2），该位由硬件自动清零。

24.7.2 AES 状态寄存器 (AES_SR)

AES 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.
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.	Res.	Res.	Res.	BUSY	WRERR	RDERR	CCF
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:4 保留，必须保持为零

位 3 **BUSY**：忙 **(Busy)**

该标志指示 AES 在 GCM 有效负载**加密**阶段是空闲还是繁忙：
0：空闲
1：忙

该标志由硬件控制。当标志指示“空闲”时，可以挂起当前消息处理以处理更高优先级的消息。
该标志仅在 GCM 有效负载加密阶段有效。在其他链接模式下，或在除有效负载加密外的 GCM 阶段，必须忽略该标志。

位 2 **WRERR**：写入错误 (Write error)

此标志指示检测到 AES_DINR 寄存器中发生了未预期的写入操作（计算阶段或数据输出阶段）：
0：未检测到
1：已检测到

该标志由硬件置 1。此位用软件清零，方法是将 AES_CR 寄存器中的 ERRC 位置 1。
标志置 1 时，如果通过 AES_CR 寄存器的 ERRIE 位使能，则会产生中断。
标志置 1 对 AES 操作没有影响。
当选择密钥生成模式或 GCM/CCM 初始化阶段时，该标志无效。

位 1 **RDERR**：读取错误标志 (Read error flag)

此标志指示检测到 AES_DOUTR 寄存器中发生了未预期的读取操作（计算阶段或数据输入阶段）：
0：未检测到
1：已检测到

该标志由硬件置 1。此位用软件清零，方法是将 AES_CR 寄存器中的 ERRC 位置 1。
标志置 1 时，如果通过 AES_CR 寄存器的 ERRIE 位使能，则会产生中断。
标志置 1 对 AES 操作没有影响。
当选择密钥生成模式或 GCM/CCM 初始化/标头阶段时，该标志无效。

位 0 **CCF**：计算完成标志 (Computation completed flag)

该标志指示计算是否完成：
0：未完成
1：已完成

计算完成后，该标志由硬件置 1。此位用软件清零，方法是将 AES_CR 寄存器中的 CCFC 位置 1。
标志置 1 时，如果通过 AES_CR 寄存器的 CCFIE 位使能，则会产生中断。
仅当 DMAOUTEN 位为 0 时，该标志才有效。DMA_EN 为 1 时，该位可能保持为 1。

24.7.3 AES 数据输入寄存器 (AES_DINR)

AES data input register

偏移地址：0x08

复位值：0x0000 0000

仅支持 32 位访问类型。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIN[x+31:x+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
DIN[x+15:x]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **DIN[x+31:x]**：写入外设的 128 位输入数据块的四个 32 位字之一

该位域提供一个 32 位输入缓冲区。在输入阶段，对该位域进行 4 次连续写入，实质上就是将一个完整的 128 位输入数据块写入 AES 外设。在每次写入时，数据交换块根据 DATATYPE[1:0] 位域对输入缓冲区的数据进行处理，然后写入 AES 内核 128 位输入缓冲区。

第一次到第四次写入操作分别将“x”替换为：96、64、32 和 0。也就是说，第一次到第四次写入操作的数据为：DIN[127:96]、DIN[95:64]、DIN[63:32] 和 DIN[31:0]。

输入数据块的数据有效性取决于 AES 工作模式：

- **模式 1**（加密）：明文
- **模式 2**（密钥生成）：未使用该位域（将 AES_KEYRx 寄存器用于输入）
- **模式 3**（解密）和**模式 4**（密钥生成和单次解密）：密文

*第 682 页的第 24.4.13 节：.AES 数据寄存器和数据交换*对数据交换操作进行了介绍。

24.7.4 AES 数据输出寄存器 (AES_DOUTR)

AES data output register

偏移地址：0x0C

复位值：0x0000 0000

仅支持 32 位访问类型。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DOUT[x+31:x+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
DOUT[x+15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:0 **DOUT[x+31:x]**：从外设读取的 128 位输出数据块的四个 32 位字之一

该位域获取一个 32 位输出缓冲区。计算完成后（CCF 置 1），对该位域进行 4 次连续读取，实质上就是从 AES 外设读取一个完整的 128 位输出数据块。在到达输出缓冲区之前，数据交换块根据 DATATYPE[1:0] 位域对 AES 内核产生的数据进行处理。

第一次到第四次读取操作分别将 DOUT[x+31:x] 替换为：96、64、32 和 0。也就是说，第一次到第四次读取操作的数据为：DOUT[127:96]、DOUT[95:64]、DOUT[63:32] 和 DOUT[31:0]。

输出数据块的数据有效性取决于 AES 工作模式：

- **模式 1**（加密）：密文
- **模式 2**（密钥生成）：未使用该位域（将 AES_KEYRx 寄存器用于输出）
- **模式 3**（解密）和**模式 4**（密钥生成和单次解密）：明文

*第 682 页的第 24.4.13 节：.AES 数据寄存器和数据交换*对数据交换操作进行了介绍。

24.7.5 AES 密钥寄存器 0 (AES_KEYR0)

AES key register 0

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
KEY[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
KEY[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **KEY[31:0]**：加密密钥 (Cryptographic key)，位 [31:0]

该位域包含 AES 加密或解密密钥的位 [31:0]，具体取决于工作模式：

- **模式 1**（加密）、**模式 2**（密钥生成）和**模式 4**（密钥生成和单次解密）：写入位域的值是加密密钥。
- **模式 3**（解密）：写入位域的值是在用于解密之前生成的加密密钥。将加密密钥写入位域后，在使能 AES 之前的读取返回相同的值。使能 AES 后且将 CCF 标志置 1 后的读取返回从加密密钥生成的解密密钥。

注： 在模式 4（密钥生成和解密）中，位域始终包含加密密钥。

只有在禁止 AES 外设时，才能向 AES_KEYRx 寄存器执行写操作。

更多详细信息，请参见*第 684 页的第 24.4.14 节：AES 密钥寄存器*。

24.7.6 AES 密钥寄存器 1 (AES_KEYR1)

AES key register 1

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY[63:48]															
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
KEY[47:32]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **KEY[63:32]**：加密密钥 (Cryptographic key)，位 [63:32]

有关 KEY[255:0] 位域的说明，请参见 AES_KEYR0 寄存器。

24.7.7 AES 密钥寄存器 2 (AES_KEYR2)

AES key register 2

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY[95:80]															
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
KEY[79:64]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **KEY[95:64]**：加密密钥 (Cryptographic key)，位 [95:64]
有关 KEY[255:0] 位域的说明，请参见 AES_KEYR0 寄存器。

24.7.8 AES 密钥寄存器 3 (AES_KEYR3)

AES key register 3

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
KEY[127:112]															
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
KEY[111:96]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **KEY[127:96]**：加密密钥 (Cryptographic key)，位 [127:96]
有关 KEY[255:0] 位域的说明，请参见 AES_KEYR0 寄存器。

24.7.9 AES 初始化向量寄存器 0 (AES_IVR0)

AES initialization vector register 0

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
IVI[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
IVI[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **IVI[31:0]**：初始化向量输入 (Initialization vector input)，位 [31:0]
有关 IVI[127:0] 位域的说明，请参见*第 684 页的第 24.4.15 节：AES 初始化向量寄存器*。
初始化向量仅用于除 ECB 以外的链接模式。
只有在禁止 AES 外设时，才能向初始化向量执行写操作。

24.7.10 AES 初始化向量寄存器 1 (AES_IVR1)

AES initialization vector register 1

偏移地址：0x24

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IVI[63:48]															
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
IVI[47:32]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **IVI[63:32]**：初始化向量输入 (Initialization vector input)，位 [63:32]
有关 IVI[127:0] 位域的说明，请参见*第 684 页的第 24.4.15 节：AES 初始化向量寄存器*。
初始化向量仅用于除 ECB 以外的链接模式。
只有在禁止 AES 外设时，才能向初始化向量执行写操作。

24.7.11 AES 初始化向量寄存器 2 (AES_IVR2)

AES initialization vector register 2

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
IVI[95:80]															
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
IVI[79:64]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **IVI[95:64]**：初始化向量输入 (Initialization vector input)，位 [95:64]
有关 IVI[127:0] 位域的说明，请参见*第 684 页的第 24.4.15 节：AES 初始化向量寄存器*。
初始化向量仅用于除 ECB 以外的链接模式。
只有在禁止 AES 外设时，才能向初始化向量执行写操作。

24.7.12 AES 初始化向量寄存器 3 (AES_IVR3)

AES initialization vector register 3

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
IVI[127:112]															
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
IVI[111:96]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **IVI[127:96]**：初始化向量输入 (Initialization vector input)，位 [127:96]
有关 IVI[127:0] 位域的说明，请参见*第 684 页的第 24.4.15 节：AES 初始化向量寄存器*。
初始化向量仅用于除 ECB 以外的链接模式。
只有在禁止 AES 外设时，才能向初始化向量执行写操作。

24.7.13 AES 密钥寄存器 4 (AES_KEYR4)

AES key register 4

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
KEY[159:144]															
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
KEY[143:128]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **KEY[159:128]**：加密密钥 (Cryptographic key)，位 [159:128]
有关 KEY[255:0] 位域的说明，请参见 AES_KEYR0 寄存器。

24.7.14 AES 密钥寄存器 5 (AES_KEYR5)

AES key register 5

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
KEY[191:176]															
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
KEY[175:160]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **KEY[191:160]**：加密密钥 (Cryptographic key)，位 [191:160]
有关 KEY[255:0] 位域的说明，请参见 AES_KEYR0 寄存器。

24.7.15 AES 密钥寄存器 6 (AES_KEYR6)

AES key register 6

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
KEY[223:208]															
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
KEY[207:192]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **KEY[223:192]**：加密密钥 (Cryptographic key)，位 [223:192]
有关 KEY[255:0] 位域的说明，请参见 AES_KEYR0 寄存器。

24.7.16 AES 密钥寄存器 7 (AES_KEYR7)

AES key register 7

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
KEY[255:240]															
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
KEY[239:224]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:0 **KEY[255:224]**：加密密钥 (Cryptographic key)，位 [255:224]
有关 KEY[255:0] 位域的说明，请参见 AES_KEYR0 寄存器。

注： *仅当选择 256 位密钥长度时，才使用密钥寄存器 4 到密钥寄存器 7。当选择 128 位密钥长度时，它们不起作用（此时仅使用密钥寄存器 0 到密钥寄存器 3）。*

24.7.17 AES 挂起寄存器 (AES_SUSPxR)

AES suspend registers

偏移地址：0x040 + x * 0x4，（x = 0 到 7）

复位值：0x0000 0000

当前任务的 AES 处理被挂起以处理更高优先级的任务时，这些寄存器包含 AES 处理器的完整内部寄存器状态。
挂起后，软件在将 AES 处理器用于更高优先级的任务前，读取 AES_SUSPxR 寄存器内容（其中 x 等于 0 到 7）并将其保存到存储器中。完成后，软件在恢复原始任务前将保存的内容恢复到相应的挂起寄存器中。

注：　这些寄存器仅在选择了 GCM、GMAC 或 CCM 链接模式时才可使用。
仅当 AES 使能时才能读取这些寄存器。当 AES 禁止时读取这些寄存器将返回 0x0000 0000。

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

位 31:0 **SUSPx**：AES 挂起 (AES suspend)
挂起操作后，每个 AES_SUSPxR 寄存器的此位域取内部 AES 寄存器之一的值。

24.7.18 AES 寄存器映射

表 135. AES 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	AES_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	KEYSIZE	Res.	CHMOD[2]	Res.	GCMPH[1:0]		DMAOUTEN	DMAINEN	ERRIE	CCFIE	ERRC	CCFC	CHMOD[1:0]		MODE[1:0]		DATATYPE[1:0]		EN
	Reset value														0		0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0004	AES_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	WRERR	RDERR	CCF
	Reset value																													0	0	0	0
0x0008	AES_DINR x=96,64,32,0	DIN[x+31: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
0x000C	AES_DOUTR x=96,64,32,0	DOUT[x+31: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
0x0010	AES_KEYR0	KEY[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	AES_KEYR1	KEY[63:32]																															
	Reset value	0	0	0	0	0	0	0	0	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	AES_KEYR2	KEY[95:64]																															
	Reset value	0	0	0	0	0	0	0	0	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	AES_KEYR3	KEY[127:96]																															
	Reset value	0	0	0	0	0	0	0	0	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	AES_IVR0	IVI[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	AES_IVR1	IVI[63:32]																															
	Reset value	0	0	0	0	0	0	0	0	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 135. AES 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
0x0028	AES_IVR2	IVI[95:64]																															
	Reset value	0	0	0	0	0	0	0	0	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	AES_IVR3	IVI[127:96]																															
	Reset value	0	0	0	0	0	0	0	0	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	AES_KEYR4	KEY[159:128]																															
	Reset value	0	0	0	0	0	0	0	0	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	AES_KEYR5	KEY[191:160]																															
	Reset value	0	0	0	0	0	0	0	0	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	AES_KEYR6	KEY[223:192]																															
	Reset value	0	0	0	0	0	0	0	0	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	AES_KEYR7	KEY[255:224]																															
	Reset value	0	0	0	0	0	0	0	0	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	AES_SUSP0R	SUSP0[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
0x0044	AES_SUSP1R	SUSP1[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
0x0048	AES_SUSP2R	SUSP2[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	AES_SUSP3R	SUSP3[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	AES_SUSP4R	SUSP4[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	AES_SUSP5R	SUSP5[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
0x0058	AES_SUSP6R	SUSP6[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
0x005C	AES_SUSP7R	SUSP7[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

25 实时时钟 (RTC)

25.1 简介

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 还包含用于管理低功耗模式的自动唤醒单元。

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。

其他 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。

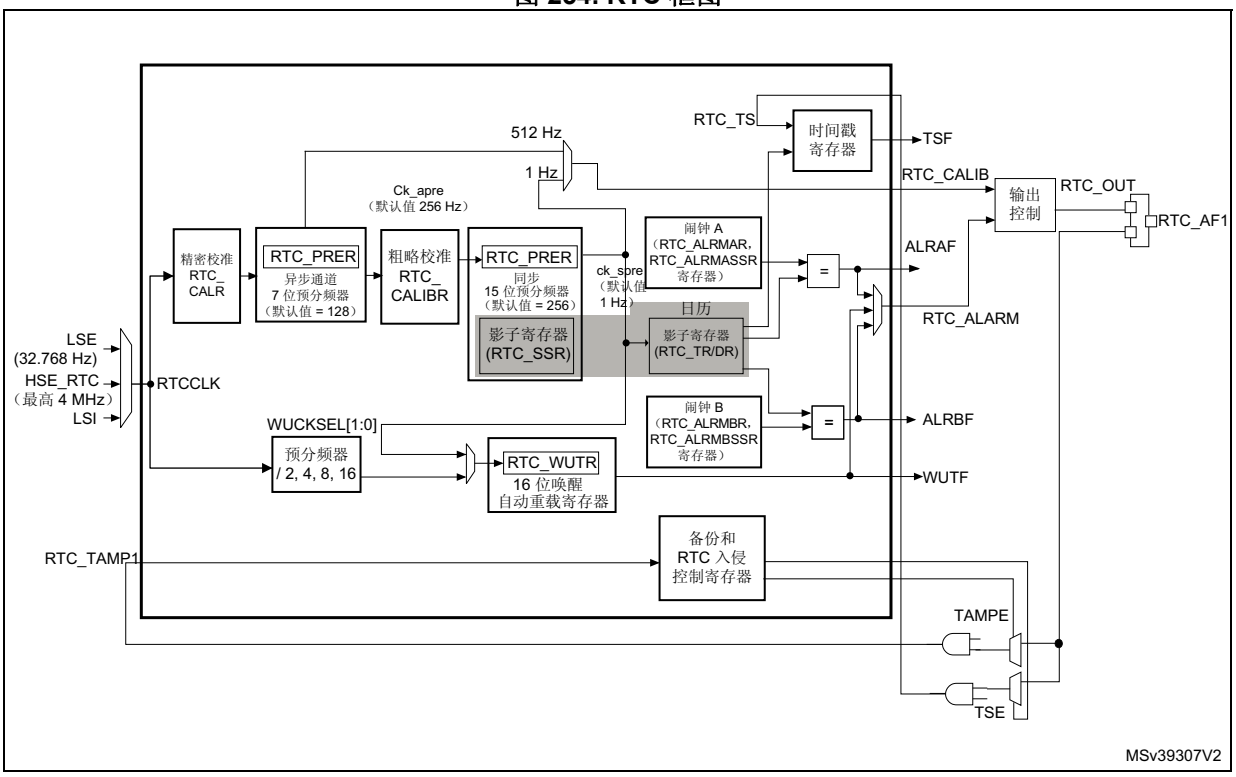
25.2 RTC 主要特性

RTC 单元的主要特性如下（参见*图 254*）：

- 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。
- 两个具有中断功能的可编程闹钟。可通过任意日历字段的组合驱动闹钟。
- 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。
- 可屏蔽中断/事件：
  - 闹钟 A
  - 闹钟 B
  - 唤醒中断
  - 时间戳
  - 入侵检测
- 数字校准电路（周期性计数器调整）
  - 精度为 5 ppm
  - 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（1 个事件）
- 入侵检测：
  - 2 个带可配置过滤器和内部上拉的入侵事件。
- 20 个备份寄存器（80 字节）。发生入侵检测事件时，将复位备份寄存器。

- 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。该输出可连接到器件 RTC_AF1 功能。
  - RTC_ALARM（闹钟 A、闹钟 B 或唤醒）。
    可通过配置 RTC_CR 寄存器的 OSEL[1:0] 位选择此输出。该输出可连接到器件 RTC_AF1 功能。
- _RTC 复用功能输入：
  - RTC_TS：时间戳事件检测。该输入可连接到器件 RTC_AF1 功能。
  - RTC_TAMP1：tamper1 事件检测。该输入可连接到器件 RTC_AF1 功能。
  - RTC_REFIN：参考时钟输入（通常为市电，50 Hz 或 60 Hz）。

图 254. RTC 框图



25.3 RTC 功能说明

25.3.1 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 6 节：STM32F413/423 的复位和时钟控制 (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 254：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 25.3.4 节*）。

25.3.2 实时时钟和日历

RTC 日历时间和日期寄存器可通过与 PCLK1（APB1 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 25.6.4 节*）。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，它将访问影子寄存器的内容。也可以通过将 RTC_CR 寄存器中的 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。

25.3.3 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。

可通过将 RTC_CR 寄存器中的 ALRAE 和 ALRBE 位置 1 来使能可编程闹钟功能。如果日历亚秒、秒、分钟、小时、日期或日分别与闹钟寄存器 RTC_ALRMASSR/RTC_ALRMAR 和 RTC_ALRMBSSR/RTC_ALRMBR 中编程的值相匹配，则 ALRAF 和 ALRBF 标志会被置为 1。可通过 RTC_ALRMAR 和 RTC_ALRMBR 寄存器的 MSKx 位以及 RTC_ALRMASSR 和 RTC_ALRMBSSR 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 和 ALRBIE 位使能闹钟中断。

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 极性。

注意： 如果秒字段被选择（RTC_ALRMAR 或 RTC_ALRMBR 中的 MSK0 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。

25.3.4 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*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）自动加载。

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 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。

25.3.5 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 PWR 电源控制寄存器 (PWR_CR) 的 DBP 位保护 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_ISR[13:8]、RTC_TAFCR 和 RTC_BKPxR 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。需要 1 到 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，应首先编程 RTC_PRER 寄存器中的同步预分频系数，然后编程异步预分频系数。即使只需要更改这两个字段中之一，也必须对 RTC_PRER 寄存器执行两次单独的写访问。
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（闹钟 A 或闹钟 B）进行编程或更新，必须执行类似的步骤：

1. 将 RTC_CR 寄存器中的 ALRAE 或 ALRBE 位清零以禁止闹钟 A 或闹钟 B。
2. 轮询 RTC_ISR 寄存器中的 ALRAWF 或 ALRBWF 位，直到其中一个置 1，以确保闹钟寄存器可以访问。需要 1 到 2 个 RTCCLK 时钟周期（由于时钟同步）。
3. 编程闹钟 A 或闹钟 B 寄存器（RTC_ALRMASSR/RTC_ALRMAR 或 RTC_ALRMBSSR/RTC_ALRMBR）。
4. 将 RTC_CR 寄存器中的 ALRAE 或 ALRBE 位置 1 以再次使能闹钟 A 或闹钟 B。

注意： *1 到 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。需要 1 到 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，WUTWF 位将会在 WUTE 被清零后的两个时钟周期被清零。

25.3.6 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），APB1 时钟频率 (f_{PCLK1}) 必须等于或大于 f_{RTCCLK} RTC 时钟频率的七倍。这可以确保同步机制行为的安全性。

如果 APB1 时钟频率低于 RTC 时钟频率的七倍，则软件必须读取日历时间寄存器和日期寄存器两次。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，APB1 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 个值一致，读取 RTC_SSR 或 RTC_TR 时会锁定高阶日历影子寄存器中的值，直到读取 RTC_DR。为避免软件对日历执行读访问的时间间隔小于 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。

注： *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 25.3.8 节），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。

注意： *当 BYPSHAD=1 时，读取日历寄存器的指令需要一个额外的 APB 周期才能完成。*

25.3.7 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_CALIBR 或 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_TAFCR)、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）。

此外，当由 LSE 提供时钟时，如果复位源并非备份域复位源（有关不受系统复位影响的 RTC 时钟源列表的详细信息，请参见 RCC），则 RTC 将在系统复位时保持运行状态。 发生备份域复位时，RTC 会停止工作，并且所有 RTC 寄存器都会设置为各自的复位值。

25.3.8 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。

注意： 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，硬件会将该位清零。

注意： 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。

25.3.9 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，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参考时钟边沿进行比较（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。

使能参考时钟检测后，必须将 PREDIV_A 和 PREDIV_S 设置为各自的默认值：

- PREDIV_A = 0x007F
- PREDIV_S = 0x00FF

注： *参考时钟检测在待机模式下不可用。*

注意： 参考时钟检测功能不能与粗略数字校准一起使用：当 REFCKON = 1 时，RTC_CALIBR 必须保持在 0x0000 0000。

25.3.10 RTC 粗略数字校准

可以使用两种数字校准方法：粗略校准和精密校准。要执行粗略校准，请参见*第 25.6.7 节：RTC 校准寄存器 (RTC_CALIBR)*。

这两种校准方法不能一起使用，应用必须从中选择一种。粗略校准出于兼容性原因而提供。要执行精密校准，请参见*第 25.3.11 节：RTC 精密数字校准*和*第 25.6.16 节：RTC 校准寄存器 (RTC_CALR)*

粗略数字校准功能可通过在异步预分频器 (ck_apre) 的输出端增加（正校准）或减少（负校准）时钟周期来实现晶振误差补偿。

将 RTC_CALIBR 寄存器中的 DCS 位置“0”和“1”可分别选择正校准和负校准。

当使能正校准时 （DCS =“0”），在 2xDC 分钟时间内，每分钟（约 15360 个 ck_apre 周期）增加 2 个 ck_apre 周期。这会提前更新日历，因此可将 RTC 的有效频率调高一些。

当使能负校准时（DCS =“1”），在 2xDC 分钟时间内，每分钟（约 15360 个 ck_apre 周期）减少 1 个 ck_apre 周期。这将推迟更新日历，因此可将 RTC 的有效频率调低一些。

可通过 RTC_CALIBR 寄存器的位 DC[4:0] 配置 DC。其数字范围为 0 到 31，对应的时间间隔 (2xDC) 范围为 0 到 62。

粗略数字校准只能在初始化模式下进行配置，并在 INIT 位清零后启动。整个校准周期将持续 64 分钟。可采用上述方法修改这 64 分钟周期的前 2xDC 分钟。

负校准的分辨率约为 2 ppm，而正校准的分辨率约为 4 ppm。最大校准范围为 -63 ppm 到 126 ppm。

当使用 LSE 或 HSE 作为 RTC 时钟时，可以使用这种校准方法。

注意： 如果 PREDIV_A < 6，数字校准可能无法正常工作。

例如当 RTCCLK=32.768 kHz 且 PREDIV_A+1=128 时

以下描述假定 ck_apre 频率为 256 Hz，由标称频率为 32.768 kHz 的 LSE 时钟提供，并且 PREDIV_A 设置为 127（默认值）。

ck_spre 时钟频率只能在 64 分钟周期的前 2xDC 分钟内被更改。例如，当 DC 等于 1 时，只有前两分钟被更改。这意味着，假定每个 ck_apre 周期表示 128 个 RTCCLK 周期（通过 PREDIV_A+1=128 计算得出），则对于每个 64 分钟周期的前 2xDC 分钟来说，每分钟都会有一秒减少 256 个 RTCCLK 周期或增加 128 个 RTCCLK 周期。

因此，对于每 125829120 个 RTCCLK 周期（64min x 60 s/min x 32768 周期/s），每个校准步骤都会增加 512 个振荡器周期或减少 256 个振荡器周期。这相当于每个校准步骤的分辨率为 +4.069 ppm 或 -2.035 ppm。因此，每个月的校准分辨率为 +10.5 或 -5.27 秒，每个月的总校准范围为 +5.45 到 -2.72 分钟。

为测量时钟偏差，需要输出一个 512 Hz 的时钟用于校准。请参见*第 25.3.14 节：校准时钟输出*。

25.3.11 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_CALRx) 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）。唯一相关的其他情况是，当 PREDIV_A 等于 0 时，应将 PREDIV_S 设置为 32759 而不是 32767（少 8）。

如果以这种方式减少 PREDIV_S，则采用以下公式计算校准输入时钟的有效频率：

$F_{CAL} = F_{RTCCLK} \times [1 + (256 - CALM) / (2^{20} + CALM - 256)]$

在这种情况下，如果 RTCCLK 恰好为 32768.00 Hz，则当 CALM[7:0] 等于 0x100 时（CALM 范围的中值），说明设置正确。

验证 RTC 校准

通过测量 RTCCLK 的精确频率，计算正确的 CALM 和 CALP 值以实现 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。

25.3.12 时间戳功能

将 RTC_CR 寄存器的 TSE 位置 1 可使能时间戳。

当在 TIMESTAMP 备用功能映射到的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。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。

通过将 RTC_CR 寄存器中的 TSIE 位置 1，可在发生时间戳事件时生成中断。

如果在时间戳标志 (TSF) 已置 1 的条件下检测到新的时间戳事件，则时间戳上溢标志 (TSOVF) 将置 1，而时间戳寄存器（RTC_TSTR 和 RTC_TSDR）将保持上一事件的结果。

注： *由于同步过程，TSF 将在时间戳事件后 2 个 ck_apre 周期置 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。

注意： 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 25.6.17 节：RTC 入侵和复用功能配置寄存器 (RTC_TAFCR)*。如果时间戳事件与配置为过滤模式的入侵事件（TAMPFLT 设置为非零值）使用同一引脚，则必须通过将 RTC_TAFCR 寄存器中的 TAMPTS 置“1”来选择入侵检测事件的时间戳模式。

时间戳复用功能

TIMESTAMP 附加功能映射到 RTC_AF1。

25.3.13 入侵检测

有两个入侵检测输入可用。这两个输入既可配置为边沿检测，也可配置为带过滤的电平检测。

RTC 备份寄存器

备份寄存器 (RTC_BKPxR) 包括 20 个 32 位寄存器，用于存储 80 字节的用户应用数据。这些寄存器在备份域中实现，可在 V_{DD} 电源关闭时通过 V_{BAT} 保持上电状态。备份寄存器不会在系统复位时或器件从待机模式唤醒时复位， 而是通过备份域复位进行复位。

发生入侵检测事件时，将复位备份寄存器。（请参见*第 25.6.20 节：RTC 备份寄存器 (RTC_BKPxR)* 和*第 710 页的入侵检测初始化*）

入侵检测初始化

两个入侵检测输入分别与 RTC_ISR2 寄存器中的标志 TAMP1F/TAMP2F 相关联。可通过将 RTC_TAFCR 寄存器中相应的 TAMP1E/TAMP2E 位置 1 来使能各输入。

入侵检测事件会复位所有备份寄存器 (RTC_BKPxR)。

通过将 RTC_TAFCR 寄存器中的 TAMPIE 位置 1，可在发生入侵检测事件时生成中断。

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的发生。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（TAMP1F、TAMP2F）也会随之置 1。

对入侵输入的边沿检测

如果 TAMPFLT 位为“00”，则在观测到上升沿或下降沿时（根据相应的 TAMPxTRG 位），TAMPER 引脚会生成入侵检测事件 (RTC_TAMP[2:1])。选择边沿检测时，会禁止入侵输入上的内部上拉电阻。

注意： 为避免丢失入侵检测事件，用于边沿检测的信号与 TAMPxE 使用逻辑与操作来检测入侵事件，以避免丢失在使能 TAMPERx 引脚前发生的入侵事件。

- TAMPxTRG = 0 时：如果 TAMPERx 复用功能在使能入侵检测之前已变为高电平（TAMPxE 位置 1），则使能 TAMPERx 后便会立即检测到入侵事件，即使在 TAMPxE 置 1 后 TAMPERx 引脚上未出现上升沿。
- TAMPxTRG = 1 时：如果 TAMPERx 复用功能在使能入侵检测之前已变为低电平，则使能 TAMPERx 后便可立即检测到入侵事件（即使在 TAMPxE 置 1 后 TAMPERx 引脚上未出现下降沿）。

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 TAMPERx 复用功能，然后再重新使能（TAMPxE 置 1）。这可防止应用在 TAMPERx 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 TAMPERx 复用功能的电平检测。

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 TAMPER 复用功能映射到的引脚从外部连接到正确的电平。*

对入侵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 (TAMP1TRG/TAMP2TRG) 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 内部上拉电阻对这些输入预充电。预充电的持续时间由 TAMPPRCH 位确定，允许增大入侵输入上的电容。

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。*

TAMPER 复用功能检测

TAMPER1 附加功能映射到 RTC_AF1 引脚。

25.3.14 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_CALIBR 寄存器中编程的校准值不会影响 RTC_CALIB 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）。

校准复用功能输出

当将 RTC_CR 寄存器中的 COE 位置 1 时，RTC_AF1 上会使能校准复用功能 (RTC_CALIB)。

注： *选择 RTC_CALIB 或 RTC_ALARM 时，RTC_AF1 会自动配置为输出复用功能。*

25.3.15 闹钟输出

闹钟输出有三种功能可供选择：ALRAF、ALRBF 和 WUTF。这些功能可反映 RTC_ISR 寄存器中相应标志的内容。

RTC_CR 寄存器中的 OSEL[1:0] 控制位用于激活 RTC_AF1 中的闹钟复用功能输出 (RTC_ALARM)，以及选择 RTC_ALARM 输出上的功能。

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_TAFCR 寄存器中的控制位 ALARMOUTTYPE 可将 RTC_ALARM 配置为输出开漏或输出上拉。

注： *使能 RTC_ALARM 之后，其优先级高于 RTC_CALIB（COE 位的设置与 RTC_AF1 无关）。*
选择 RTC_CALIB 或 RTC_ALARM 时，RTC_AF1 会自动配置为输出复用功能。

25.4 RTC 和低功耗模式

表 136. 低功耗模式对 RTC 的作用

模式	说明
睡眠	无任何作用 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。

25.5 RTC 中断

所有 RTC 中断均与 EXTI 控制器相连。

要使能 RTC 闹钟中断，需按照以下顺序操作：

1. 将 EXTI 线 17 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（闹钟 A 或闹钟 B）。

要使能 RTC 唤醒中断，需按照以下顺序操作：

1. 将 EXTI 线 22 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 以生成 RTC 唤醒定时器事件。

要使能 RTC 入侵中断，需按照以下顺序操作：

1. 将 EXTI 线 21 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 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 EXTI 线 21 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 TAMP_STAMP IRQ 通道并将其使能。
3. 配置 RTC 以检测 RTC 时间戳事件。

表 137. 中断控制位

中断事件	事件标志	使能控制位	退出睡眠模式	退出停止模式	退出待机模式
闹钟 A	ALRAF	ALRAIE	是	是(1)	是(1)
闹钟 B	ALRBF	ALRBIE	是	是(1)	是(1)
唤醒	WUTF	WUTIE	是	是(1)	是(1)
时间戳	TSF	TSIE	是	是(1)	是(1)
Tamper1 检测	TAMP1F	TAMPIE	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。

25.6 RTC 寄存器

有关寄存器说明中使用的缩写，请参见本参考手册中的*第 50 页的第 1.2 节*。

外设寄存器必须按字（32 位）进行访问。

25.6.1 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*日历初始化和配置*和*读取日历*。

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 31-24 保留，必须保持复位值

位 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)

注： *此寄存器受写保护。RTC 寄存器写保护中介绍了写访问的过程。*

25.6.2 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*日历初始化和配置*和*读取日历*。

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)

注： *此寄存器受写保护。RTC 寄存器写保护中介绍了写访问的过程。*

25.6.3 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.	Res.	COE	OSEL[1:0]		POL	COSEL	BKP	SUB1H	ADD1H
								rw	rw	rw	rw	rw	rw	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSIE	WUTIE	ALRBIE	ALRAIE	TSE	WUTE	ALRBE	ALRAE	DCE	FMT	BYPSHAD	REFCKON	TSEDGE	WUCKSEL[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:24 保留，必须保持复位值。

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 25.3.14 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：不起作用

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：不起作用

1：将当前时间增加 1 小时。这可用于夏季时间更改

位 15 **TSIE**：时间戳中断使能 (Timestamp interrupt enable)
0：禁止时间戳中断
1：使能时间戳中断

位 14 **WUTIE**：唤醒定时器中断使能 (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 **DCE**：粗略数字校准使能 (Coarse digital calibration enable)
0：禁止数字校准
1：使能数字校准
PREDIV_A 必须大于或等于 6

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 APB1 时钟的频率低于 7 倍的 RTCCLK 频率，则必须将 BYPSHAD 置“1”。

位 4 **REFCKON**：参考时钟检测使能（50 Hz 或 60 Hz）(Reference clock detection enable (50 or 60 Hz))
0：禁止参考时钟检测
1：使能参考时钟检测
注： PREDIV_S 必须为 0x00FF。

位 3 **TSEDGE**：时间戳事件有效边沿 (Timestamp event active edge)

0：TIMESTAMP 上升沿生成时间戳事件

1：TIMESTAMP 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： *WUT = 唤醒单元计数器值。当 WUCKSEL[2:1 = 11] 时，WUT = (0x0000 到 0xFFFF) + 0x10000（增加的值）。*

只能在初始化模式下 (RTC_ISR/INITF = 1) 对该寄存器的位 7、6 和 4 执行写操作。

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。RTC 寄存器写保护中介绍了写访问的过程。

25.6.4 RTC 初始化和状态寄存器 (RTC_ISR)

RTC initialization and status register

偏移地址：0x0C

备份域复位值：0x0000 0007

系统复位值：不受影响（INIT、INITF 和 RSF 除外，它们在复位时被清零）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RECALPF
															r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res	TAMP1F	TSOVF	TSF	WUTF	ALRBF	ALRAF	INIT	INITF	RSF	INITS	SHPF	WUT WF	ALRB WF	ALRA WF
		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rw	r	rc_w0	r	r	r	r	r

位 31:17 保留，必须保持复位值

位 16 **RECALPF**：重新校准挂起标志 (Recalibration pending Flag)

当软件对 RTC_CALR 寄存器执行写操作时，RECALPF 状态标志将自动置“1”，指示 RTC_CALR 寄存器已屏蔽。当采用新的校准设置时，该位恢复为“0”。请参见*动态重校准*。

位 15:14 保留，必须保持复位值。

位 13 **TAMP1F**：入侵检测标志 (Tamper detection flag)

当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 (Timestamp overflow flag)

当在 TSF 已置 1 的情况下发生时间戳事件时，由硬件将此标志置 1。

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 (Timestamp 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。

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。

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（备份域复位值状态），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。执行完相应平移操作后，此标志将由硬件清零。对 SHPF 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)

此位在 RTC_CR 中的 WUTE 位清零 2 个 RTCCLK 周期后由硬件置 1，在 WUTE 位置 1 后 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。*

此寄存器受写保护（RTC_ISR[13:8] 位除外）。RTC 寄存器写保护中介绍了写访问的过程。

25.6.5 RTC 预分频器寄存器 (RTC_PRER)

RTC prescaler register

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)

注： *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。请参见日历初始化和配置。*

此寄存器受写保护。RTC 寄存器写保护中介绍了写访问的过程。

25.6.6 RTC 唤醒定时器寄存器 (RTC_WUTR)

RTC wakeup timer register

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。

注： 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。

注： 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。RTC 寄存器写保护中介绍了写访问的过程。

25.6.7 RTC 校准寄存器 (RTC_CALIBR)

RTC calibration register

偏移地址：0x18

备份域复位值：0x0000 0000

系统复位：不受影响

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

位 31:8 保留，必须保持复位值

位 7 **DCS**：数字校准符号 (Digital calibration sign)

0：正校准：增加日历更新频率

1：负校准：降低日历更新频率

位 6:5 保留，必须保持复位值。

位 4:0 **DC[4:0]**：数字校准 (Digital calibration)

DCS = 0（正校准）

00000：+0 ppm

00001：+4 ppm（舍入值）

00010：+8 ppm（舍入值）

..

11111：+126 ppm（舍入值）

DCS = 1（负校准）

00000：-0 ppm

00001：-2 ppm（舍入值）

00010：-4 ppm（舍入值）

..

11111：-63 ppm（舍入值）

有关准确的步骤值，请参见*例如当 RTCCLK=32.768 kHz 且 PREDIV_A+1=128 时*。

注： *只能在初始化模式（RTC_ISR/INITF = “1”）下对此寄存器执行写操作。*

此寄存器受写保护。RTC 寄存器写保护中介绍了写访问的过程。

25.6.8 RTC 闹钟 A 寄存器 (RTC_ALRMAR)

RTC alarm A register

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)。

注：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。RTC 寄存器写保护中介绍了写访问的过程。

25.6.9 RTC 闹钟 B 寄存器 (RTC_ALRMBR)

RTC alarm B register

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)

注： *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。RTC 寄存器写保护中介绍了写访问的过程。*

25.6.10 RTC 写保护寄存器 (RTC_WPR)

RTC write protection register

偏移地址：0x24

备份域复位值：0x0000 0000

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

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*。

25.6.11 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.
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 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。

25.6.12 RTC 平移控制寄存器 (RTC_SHIFTR)

RTC shift control register

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：不起作用

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。

请参见*第 25.3.8 节：RTC 同步*。

注： 此寄存器受写保护。第 704 页的RTC 寄存器写保护中介绍了写访问的过程。

25.6.13 RTC 时间戳时间寄存器 (RTC_TSTR)

RTC time stamp time register

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)。

注：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。

25.6.14 RTC 时间戳日期寄存器 (RTC_TSDR)

RTC time stamp date register

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)

注： *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。*

25.6.15 RTC 时间戳亚秒寄存器 (RTC_TSSSR)

RTC timestamp sub second register

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.
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 保留

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。

注： *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。*

25.6.16 RTC 校准寄存器 (RTC_CALR)

RTC calibration register

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.
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
CALP	CALW8	CALW16	Res.	Res.	Res.	Res.	CALM[8:0]								
rw	rw	rw	r	r	r	r	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 25.3.11 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。
CALW8=“1”时，CALM[1:0] 将始终保持为“00”。
请参见*第 25.3.11 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，则必须将该位置“1”。
注： 当 CALW16=“1”时，CALM[0] 将始终保持为“0”。
请参见*第 25.3.11 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 708 页的第 25.3.11 节：RTC 精密数字校准*。

注： 此寄存器受写保护。RTC 寄存器写保护中介绍了写访问的过程。

25.6.17 RTC 入侵和复用功能配置寄存器 (RTC_TAFCR)

RTC tamper and alternate function 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.	Res.	Res.	Res.	Res.	Res.	Res.	ALARMOUT TYPE	TSIN SEL	TAMP1I NSEL
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAMP PUDIS	TAMP PRCH[1:0]		TAMP FLT[1:0]		TAMP FREQ[2:0]			TAMPTS	Res.	Res.	Res.	Res.	TAMPIE	TAMP1TRG	TAMP1E
rw	rw	rw	rw	rw	rw	rw	rw	rw					rw	rw	rw

位 31:19 保留，必须保持复位值。始终读为 0。

位 18 **ALARMOUTTYPE**：RTC_ALARM 输出类型 (RTC_ALARM output type)
0：RTC_ALARM 为开漏输出
1：RTC_ALARM 为推挽输出

位 17 **TSINSEL**：TIMESTAMP 映射 (TIMESTAMP mapping)
0：RTC_AF1 用作 TIMESTAMP
1：保留

位 16 **TAMP1INSEL**：TAMPER1 映射 (TAMPER1 mapping)
0：RTC_AF1 用作 TAMPER1
1：保留
注： 更改 TAMP1INSEL 时必须复位 TAMP1E，以避免将 TAMP1F 意外置 1。

位 15 **TAMPPUDIS**：TAMPER 上拉禁止 (TAMPER pull-up disable)
该位决定在每次采样之前是否对入侵引脚都进行预充电。
0：采样之前对入侵引脚进行预充电（使能内部上拉）
1：禁止对入侵引脚进行预充电
注：

位 14:13 **TAMPPRCH[1:0]**：入侵预充电持续时间 (Tamper 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 对每个入侵输入都有效。
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]**：入侵过滤器计数 (Tamper 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 对每个入侵输入都有效。
0x0：在入侵输入转变为有效电平的边沿激活入侵（入侵输入上无内部上拉）。
0x1：在有效电平上连续执行 2 次采样后激活入侵。
0x2：在有效电平上连续执行 4 次采样后激活入侵。
0x3：在有效电平上连续执行 8 次采样后激活入侵。

位 10:8 **TAMPFREQ[2:0]**：入侵采样频率 (Tamper sampling frequency)
决定对每个入侵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 6:3 保留。始终读为 0。

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

位 1 **TAMP1TRG**：入侵 1 的有效电平 (Active level for tamper 1)

如果 TAMPFLT ! = 00：

0：TAMPER1 保持低电平会触发入侵检测事件。

1：TAMPER1 保持高电平会触发入侵检测事件。

如果 TAMPFLT ! = 00：

0：TAMPER1 上升沿会触发入侵检测事件。

1：TAMPER1 下降沿会触发入侵检测事件。

注意： 如果 TAMPFLT = 0，则更改 TAMP1TRG 时必须复位 TAMP1E，以避免将 TAMP1F 意外置 1。

位 0 **TAMP1E**：入侵 1 检测使能 (Tamper 1 detection enable)

0：禁止入侵 1 检测

1：使能入侵 1 检测

25.6.18 RTC 闹钟 A 亚秒寄存器 (RTC_ALRMASSR)

RTC alarm A sub second register

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。

注： *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 704 页的 RTC 寄存器写保护中介绍了写访问的过程。*

25.6.19 RTC 闹钟 B 亚秒寄存器 (RTC_ALRMBSSR)

RTC alarm B sub second register

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.
r	r	r	r	rw	rw	rw	rw	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.	SS[14:0]														
r	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。

注： *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 寄存器写保护中介绍了写访问的过程。*

25.6.20 RTC 备份寄存器 (RTC_BKPxR)

RTC backup registers

偏移地址：0x50 到 0x9C

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。

25.6.21 RTC 寄存器映射

表 138. 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.	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	1	0	0	0	0	1			0	0	0	0	0	1
0x08	**RTC_CR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	COE	OSEL [1:0]		POL	COSEL	BKP	SUB1H	ADD1H	TSIE	WUTIE	ALRBIE	ALRAIE	TSE	WUTE	ALRBE	ALRAE	DCE	FMT	BYPSHAD	REFCKON	TSEDGE	WCKSEL [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
0x0C	**RTC_ISR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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	1	0	1
0x10	**RTC_PRER**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREDIV_A[6:0]							Res.	PREDIV_S[14:0]														
	Reset value										1	1	1	1	1	1	1		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

表 138. 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
0x18	**RTC_CALIBR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DCS	Res.	Res.	DC[4:0]				
	Reset value																									0			0	0	0	0	0
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[7:0]							
	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
0x38	**RTC_TSSSR**	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_TAFCR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ALARMOUTTYPE	TSINSEL	TAMP1INSEL	TAMPPUDIS	TAMPPRCH[1:0]		TAMPFLT[1:0]		TAMPFREQ[2:0]			TAMPTS	Res.	Res.	Res.	Res.	TAMPIE	TAMP1ETRG	TAMP1E
	Reset value														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

表 138. 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
0x50 to 0x9C	**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_BKP19R**	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 54 页的第 2.2.2 节*。

注意： *表 138* 中的复位值是指备份域复位后的值。大多数寄存器不受系统复位的影响。有关详细信息，请参见*第 25.3.7 节：复位 RTC*。

26 超快速内部集成电路 (FMPI2C) 接口

26.1 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。

26.2 FMPI2C 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
  - 可选的时钟延展
  - 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 26.3 节：FMPI2C 特性实现*）：

- 兼容 SMBus 规范第 3.0 版：
  - 具有 ACK 控制的硬件 PEC（数据包错误校验）生成和验证
  - 命令和数据应答控制
  - 支持地址解析协议 (ARP)
  - 支持主机和从设备
  - SMBus 报警
  - 超时和空闲条件检测
- 兼容 PMBus 第 1.3 版标准
- 独立时钟：选择独立时钟源可使 FMPI2C 通信速度不受 PCLK 时钟频率更改的影响

26.3 FMPI2C 特性实现

本手册介绍了 FMPI2C1 中实现的所有特性。

表 139. STM32F413/423FMPI2C 实现

I2C特性(1)	I2CFMP1
7 位寻址模式	X
10 位寻址模式	X
标准模式 （高达 100 kbps）	X
快速模式 （高达 400 kbps）	X
超快速模式 (2) （高达 1 Mbps）	X
独立时钟	X
从停止模式唤醒	-
SMBus/PMBus	X

1. X = 支持。
2. Fm+ 模式下不支持 20 mA 输出驱动器。

26.4 FMPI2C 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。

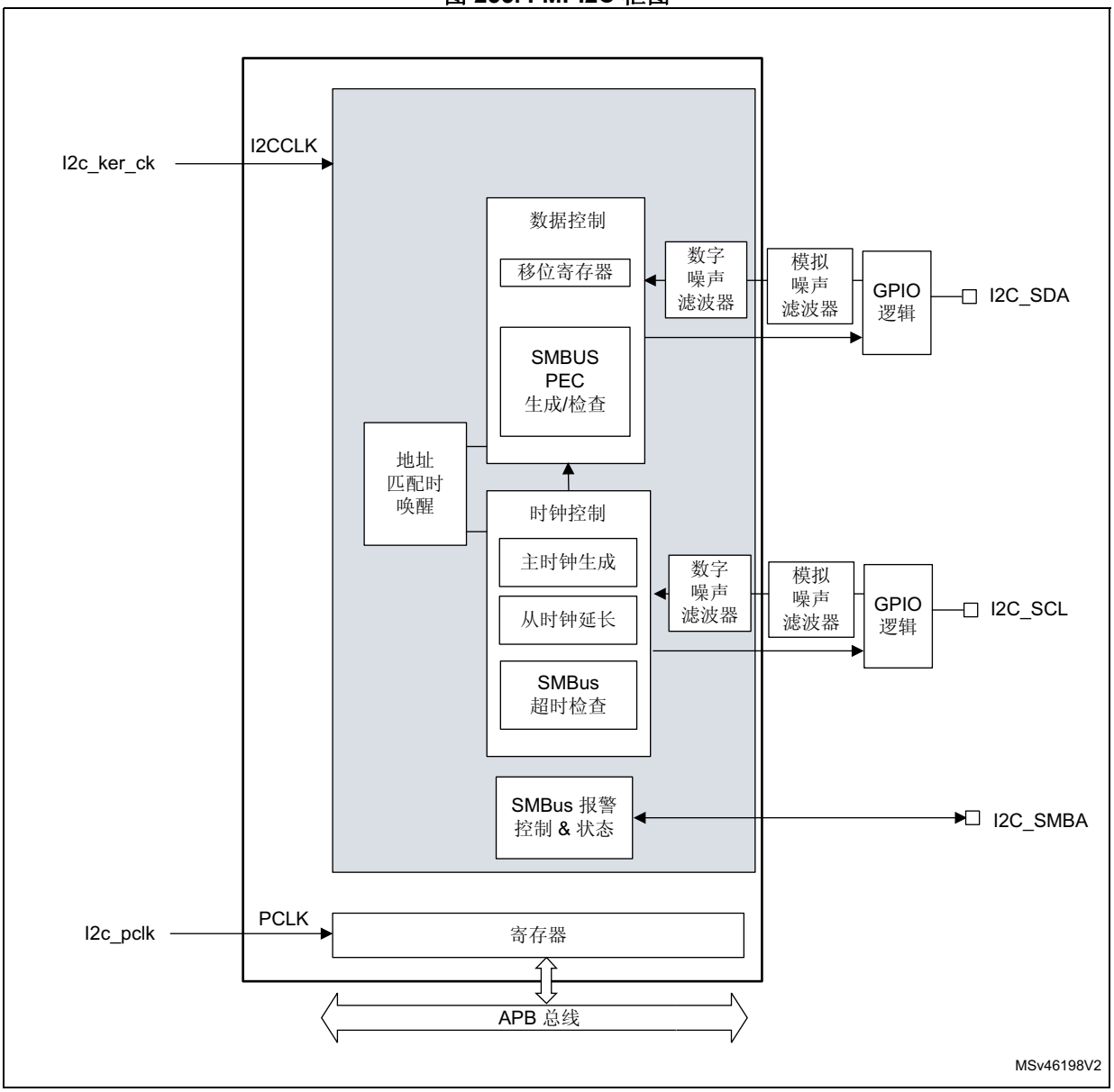
该接口也可通过数据引脚 (SDA) 和时钟引脚 (SCL) 连接到 SMBus。

如果支持 SMBus 功能：还可使用额外的可选 SMBus 报警引脚 (SMBA)。

26.4.1 FMPI2C 框图

FMPI2C 接口的框图如*图 255* 所示。

图 255. FMPI2C 框图



FMPI2C 的时钟由独立时钟源提供，这使得 FMPI2C 能够独立于 PCLK 频率工作。

该独立时钟源可从以下时钟源中选择：

- 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 6 节：STM32F413/423 的复位和时钟控制 (RCC)*。

26.4.2 FMPI2C 时钟要求

FMPI2C 内核的时钟由 FMPI2CCLK 提供。

FMPI2CCLK 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

注意： 当 FMPI2C 内核的时钟由 PCLK 提供时，该时钟必须遵循 t_{I2CCLK} 的条件。

26.4.3 模式选择

该接口在工作时可选用以下四种模式之一：

- 从机发送
- 从机接收
- 主机发送
- 主机接收

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

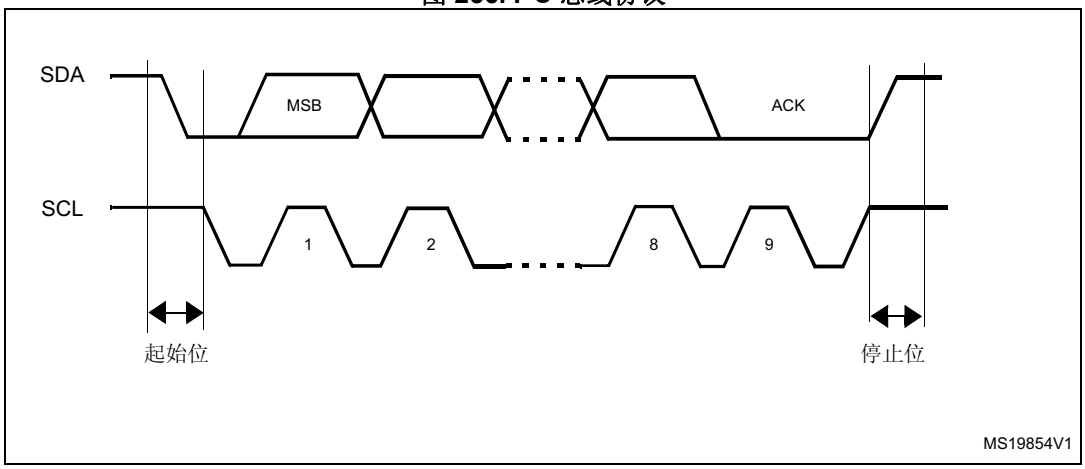
在主模式下，FMPI2C 接口会启动数据传输并生成时钟信号。串行数据传输始终是在出现起始位时开始，在出现停止位时结束。起始位和停止位均在主模式下由软件生成。

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 256. I²C 总线协议



应答位可由软件使能或禁止。FMPI2C 接口地址可通过软件进行选择。

26.4.4 FMPI2C 初始化

使能和禁止外设

FMPI2C 外设时钟必须在时钟控制器中进行配置和使能（请参见*第 6 节：STM32F413/423 的复位和时钟控制 (RCC)*）。

然后可通过将 FMPI2C_CR1 寄存器中的 PE 位置 1 使能 FMPI2C。

当禁止 FMPI2C (PE=0) 时，I²C 将执行软件复位。更多详细信息，请参见*第 26.4.5 节：软件复位*。

噪声滤波器

在通过将 FMPI2C_CR1 寄存器中的 PE 位置 1 来使能 FMPI2C 外设之前，如有必要，用户必须配置噪声滤波器。默认情况下，SDA 和 SCL 输入上开启了模拟噪声滤波器。该模拟滤波器符合 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。用户可通过将 ANFOFF 位置 1 来禁止该模拟滤波器，和/或通过配置 FMPI2C_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 FMPI2CCLK 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 FMPI2CCLK 周期可编程。

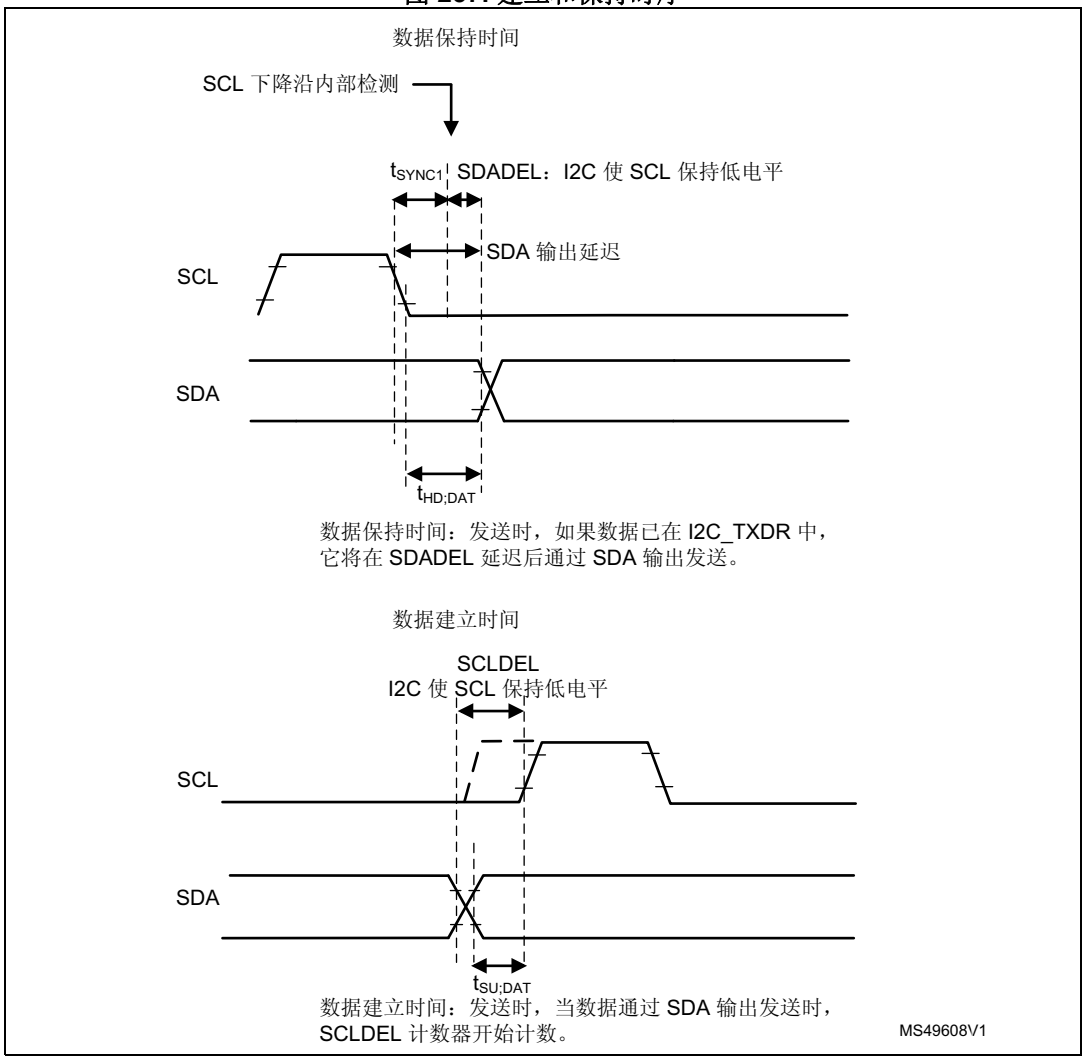
注意： 使能 FMPI2C 时，不允许更改滤波器配置。

FMPI2C 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 FMPI2C_TIMINGR 寄存器中的 PRESC[3:0]、SCLDEL[3:0] 和 SDADEL[3:0] 位。

STM32CubeMX 工具计算 I2C_TIMINGR 内容并在 I2C 配置窗口中进行显示。

图 257. 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 x t_{I2CCLK}
- SCL 与 FMPI2CCLK 时钟建立同步而产生的延时（2 到 3 个 FMPI2CCLK 周期）

为了桥接 SCL 下降沿的未定义区域，用户编程 SDADEL 时必须遵循以下条件：

$\{t_{f\ (max)} + t_{HD;DAT\ (min)} - t_{AF(min)} - [(DNF + 3) \times t_{I2CCLK}]\} / \{(PRESC + 1) \times t_{I2CCLK}\} \leq SDADEL$

$SDADEL \leq \{t_{HD;DAT\ (max)} - t_{AF(max)} - [(DNF+4) \times t_{I2CCLK}]\} / \{(PRESC + 1) \times t_{I2CCLK}\}$

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）。只有器件未延展 SCL 信号的低电平周期 (t_{LOW}) 时，才必须满足该最大值条件。如果时钟延展 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 值来延展 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 140：I2C-SMBUS 规范数据建立和保持时间*。

- 在 t_{SDADEL} 延时后，或在因数据未写入 I2C_TXDR 寄存器而导致从器件必须延展时钟的情况下发送 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) x t_{PRESC}，其中 t_{PRESC} = (PRESC+1) x t_{I2CCLK}。
  t_{SCLDEL} 会影响建立时间 $t_{SU;DAT}$。

为了桥接 SDA 跳变（上升沿通常为最坏情况）的未定义区域，编程 SCLDEL 时必须遵循以下条件：

$\{[t_{r\ (max)} + t_{SU;DAT\ (min)}] / [(PRESC+1)] \times t_{I2CCLK}]\} - 1 <= SCLDEL$

有关 t_r 和 $t_{SU;DAT}$ 标准值的信息，请参见*表 140：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。

注： *在发送和接收模式下，对于每个时钟脉冲，检测到 SCL 下降沿后，I2C 主器件或从器件会至少在 [(SDADEL+SCLDEL+1) x (PRESC+1) + 1] x t_{I2CCLK} 期间内延展 SCL 低电平时间。在发送模式下，如果 SDADEL 计数器计数结束后数据还未写入 I2C_TXDR，则 I2C 会继续延展 SCL 低电平时间，直到写入下一个数据。随后，会将新数据 MSB 发送到 SDA 输出，SCLDEL 计数器将开始计数，同时会继续延展 SCL 低电平时间以确保提供充足的数据建立时间。*

如果从模式下 NOSTRETCH=1，则 SCL 不会延展。因此，编程 SDADEL 时还必须确保提供充足的建立时间。

表 140. 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 FMPI2C_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}。

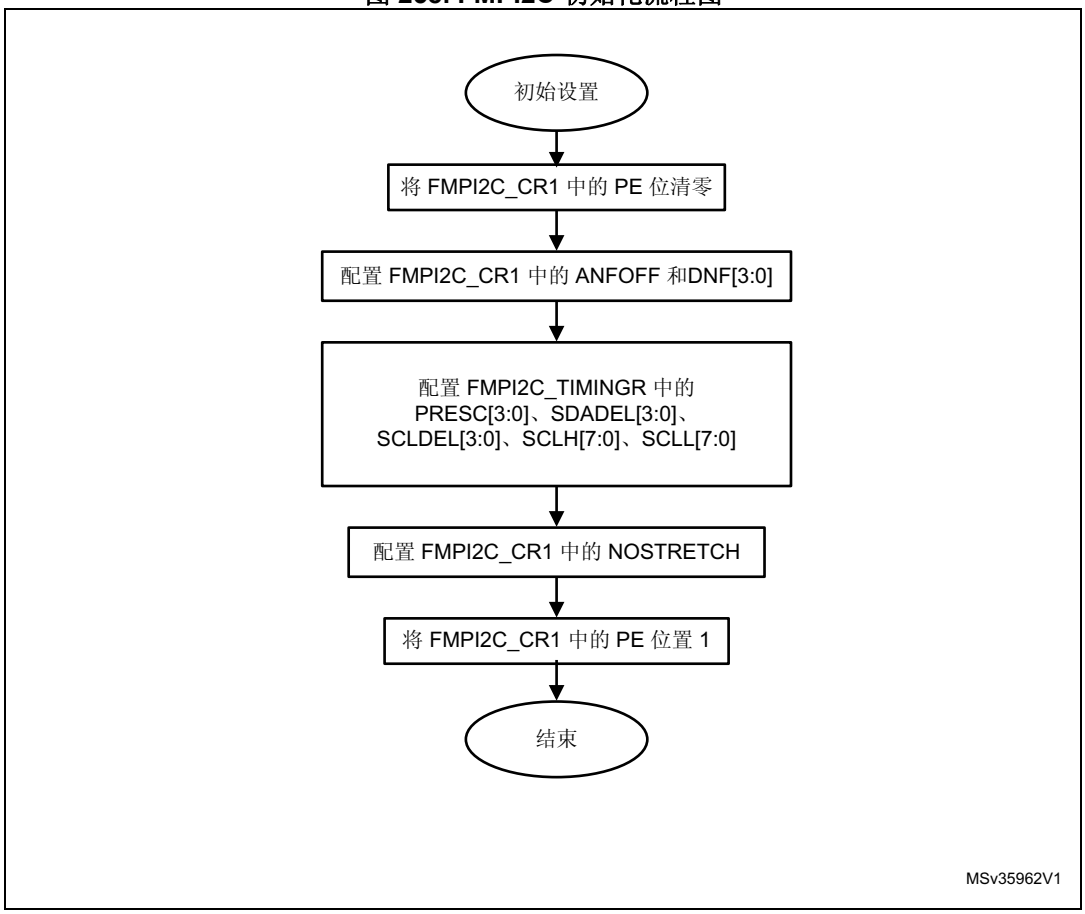
更多详细信息，请参见 *FMPI2C 主模式初始化*。

注意： 使能 FMPI2C 后，不允许更改时序配置。

此外，还必须在使能外设之前配置 FMPI2C 从 NOSTRETCH 模式。更多详细信息，请参见 *FMPI2C 从模式初始化*。

注意： 使能 FMPI2C 时，不允许更改 NOSTRETCH 配置。

图 258. FMPI2C 初始化流程图



26.4.5 软件复位

可通过将 FMPI2C_CR1 寄存器中的 PE 位清零来执行软件复位。在这种情况下，FMPI2C 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. FMPI2C_CR2 寄存器：START、STOP 和 NACK
2. FMPI2C_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. FMPI2C_CR2 寄存器：PECBYTE
2. FMPI2C_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。写入以下软件序列可确保这一点：- 写入 PE=0 - 检查 PE=0 - 写入 PE=1。

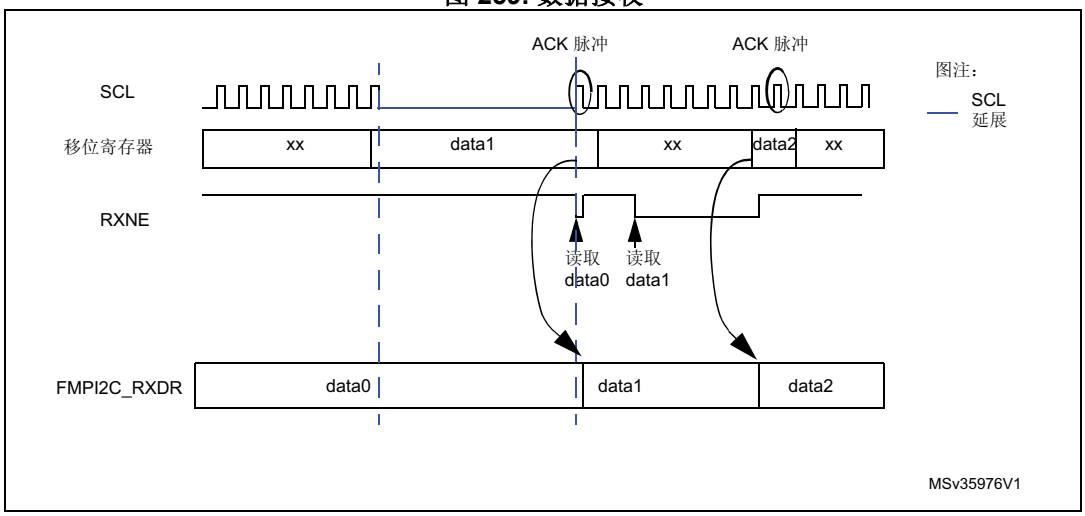
26.4.6 数据传输

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 FMPI2C_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），则将延展 SCL 线的低电平时间，直到读取了 FMPI2C_RXDR 为止。在第 8 个和第 9 个 SCL 脉冲之间（应答脉冲之前）插入一段延展的时间。

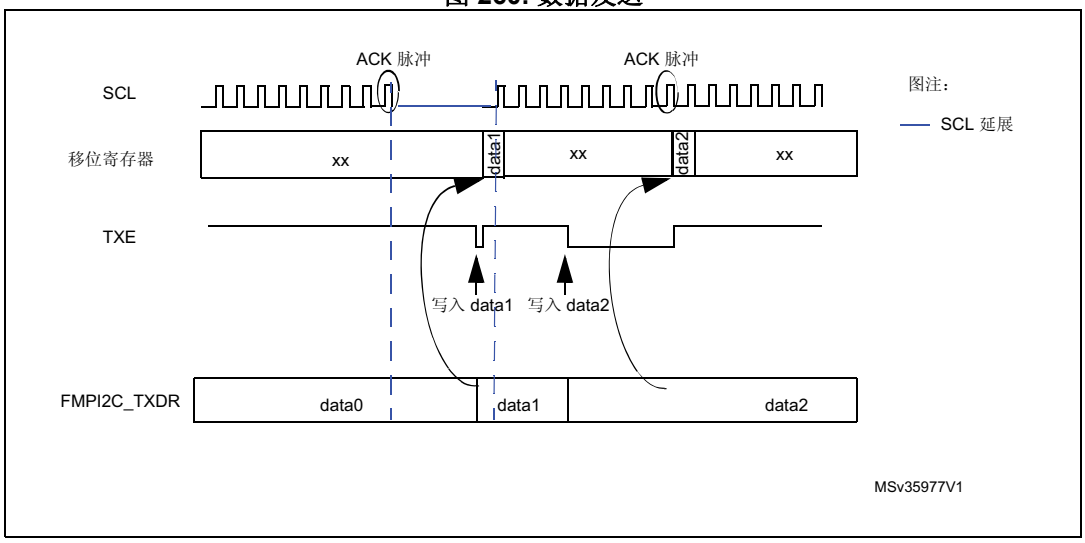
图 259. 数据接收



发送

如果 FMPI2C_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 FMPI2C_TXDR 内尚未写入任何数据），则将延展 SCL 线的低电平时间，直到写入了 FMPI2C_TXDR 为止。在第 9 个 SCL 脉冲后进行延展。

图 260. 数据发送



硬件传输管理

FMPI2C 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
- 从机接收模式下控制 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 FMPI2C_CR2 寄存器中的 SBC（从字节控制）位置 1。

待传输的字节数在 FMPI2C_CR2 寄存器的 NBYTES[7:0] 位域中进行编程。如果待传输的字节数 (NBYTES) 大于 255，或者接收方希望控制是否对接收到的数据字节进行应答，则必须选择重载模式，方法是将 FMPI2C_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 便会延展。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**（FMPI2C_CR2 寄存器中的 AUTOEND = “1”）。在该模式下，一旦完成 NBYTES[7:0] 位域中所编程字节数的数据传输，主器件便会自动发送停止位。
- **软件结束模式**（FMPI2C_CR2 寄存器中的 AUTOEND = “0”） 在该模式下，一旦完成 NBYTES[7:0] 位域中所编程字节数的数据传输，TC 标志将置 1，并且 TCIE 置 1 时将生成中断。只要 TC 标志置 1，SCL 信号便会延展，需要软件介入操作。当软件把 FMPI2C_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。

注意： 当 RELOAD 位置 1 时，AUTOEND 位将不起作用。

表 141. FMPI2C 配置

功能	SBC 位	RELOAD 位	AUTOEND 位
主 Tx/Rx NBYTES + STOP	x	0	1
主 Tx/Rx + NBYTES + RESTART	x	0	0
从 Tx/Rx 接收的所有字节都要回复应答	0	x	x
具有 ACK 控制的从 Rx	1	1	x

26.4.7 FMPI2C 从模式

FMPI2C 从模式初始化

要在从模式下工作，用户必须至少使能一个从地址。可使用 FMPI2C_OAR1 和 FMPI2C_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 FMPI2C_OAR1 寄存器的 OA1MODE 位置 1 配置为 10 位寻址模式。

  通过将 FMPI2C_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 FMPI2C_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 FMPI2C 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 FMPI2C_OAR1 或 FMPI2C_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 FMPI2C_OAR2 寄存器中的 OA2EN 位置 1 来使能 OA2。
- 通过将 FMPI2C_CR1 寄存器中的 GCEN 位置 1 来使能广播呼叫地址。

当通过 FMPI2C 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展功能（即必要时延展 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展，则必须对 FMPI2C 进行如下配置：将 FMPI2C_CR1 寄存器中 NOSTRETCH 位置 1。

接收到 ADDR 中断后，如果使能多个地址，则用户必须读取 FMPI2C_ISR 寄存器中的 ADDCODE[6:0] 位，以确定是哪个地址匹配。还必须检查 DIR 标志，以获悉传输方向。

带时钟延展的从模式 (NOSTRETCH = 0)

在默认模式下，FMPI2C 从器件会在以下情况下延展 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 FMPI2C_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 FMPI2C_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 FMPI2C_RXDR 寄存器但新的数据接收已完成。读取 FMPI2C_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，FMPI2C 会延展 SCL 的低电平时间（不超过 $[(SDADEL+SCLDEL+1) \times (PRESC+1) + 1] \times t_{I2CCLK}$）。

不带时钟延展的从模式 (NOSTRETCH = 1)

当 FMPI2C_CR1 寄存器中的 NOSTRETCH = 1 时，FMPI2C 从器件不会延展 SCL 信号。

- ADDR 标志置 1 时，不会延展 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 FMPI2C_TXDR 寄存器写入数据。否则，会发生下溢，FMPI2C_ISR 寄存器中的 OVR 标志将置 1，如果 FMPI2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。当第一次数据发送开始而 STOPF 位仍置 1（尚未清零）时，OVR 标志也将置 1。因此，如果写入下一次传输要发送的第一个数据后才清零上一次传输的 STOPF 标志，则应提供 OVR 状态，甚至对于待发送的第一个数据也是如此。
- 接收时，必须在下一个数据字节的第 9 个 SCL 脉冲（ACK 脉冲）出现之前，从 FMPI2C_RXDR 寄存器读取数据。否则，会发生上溢，FMPI2C_ISR 寄存器中的 OVR 标志将置 1，如果 FMPI2C_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 FMPI2C_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，从而延展 SCL 信号的第 8 个和第 9 个脉冲之间的低电平时间。用户可以从 FMPI2C_RXDR 寄存器中读取数据，然后通过配置 FMPI2C_CR2 寄存器中的 ACK 位来决定是否应答。通过将 NBYTES 编程为非零值来释放 SCL 延展：发送应答或不应答信号，然后可继续接收下一个字节。

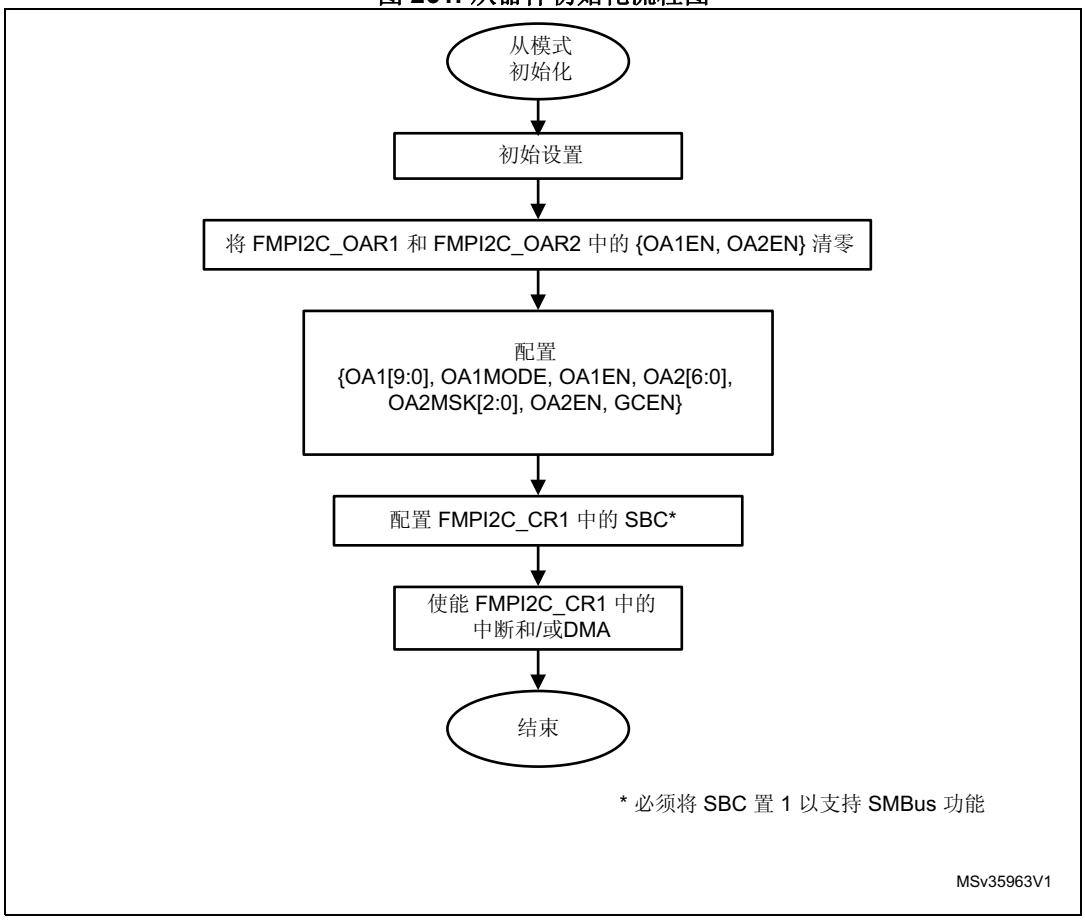
NBYTES 可加载大于 0x1 的值，在这种情况下，接收流在 NBYTES 个数据接收期间是连续的。

注： *SBC 位只能在 FMPI2C 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。

注意： 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 261. 从器件初始化流程图



从机发送

当 FMPI2C_TXDR 寄存器为空时，将生成发送中断状态 (TXIS)。如果 FMPI2C_CR1 寄存器中的 TXIE 位置 1，将生成中断。

FMPI2C_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，FMPI2C_ISR 寄存器中的 NACKF 位将置 1，如果 FMPI2C_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，以使主机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 FMPI2C_CR1 寄存器中的 STOPIE 位置 1 时，FMPI2C_ISR 寄存器中的 STOPF 标志将置 1 并且会生成中断。在大多数应用中，SBC 位通常编程为“0”。在这种情况下，如果接收到从地址 (ADDR=1) 时 TXE = 0，用户可以选择发送 FMPI2C_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 FMPI2C_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。

注意： 如果 NOSTRETCH=1，当 ADDR 标志置 1 时不会延展 SCL 时钟，因此用户无法在 ADDR 子程序中刷新 FMPI2C_TXDR 寄存器的内容，从而编程第一个数据字节。必须在 FMPI2C_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 FMPI2C_TXDR 寄存器，从而编程新的数据字节。STOPF 位必须在这些动作之后被清除，以保证他们在第一个数据传输开始前就执行，跟着地址的 ACK。

  如果第一次数据传输开始时 STOPF 仍置 1，则将生成下溢错误（OVR 标志置 1）。

  如果需要 TXIS 事件（发送中断或发送 DMA 请求），用户必须将 TXE 位和 TXIS 位均置 1，以便生成 TXIS 事件。

图 262. FMPI2C 从机发送的传输序列流程图 (NOSTRETCH=0)

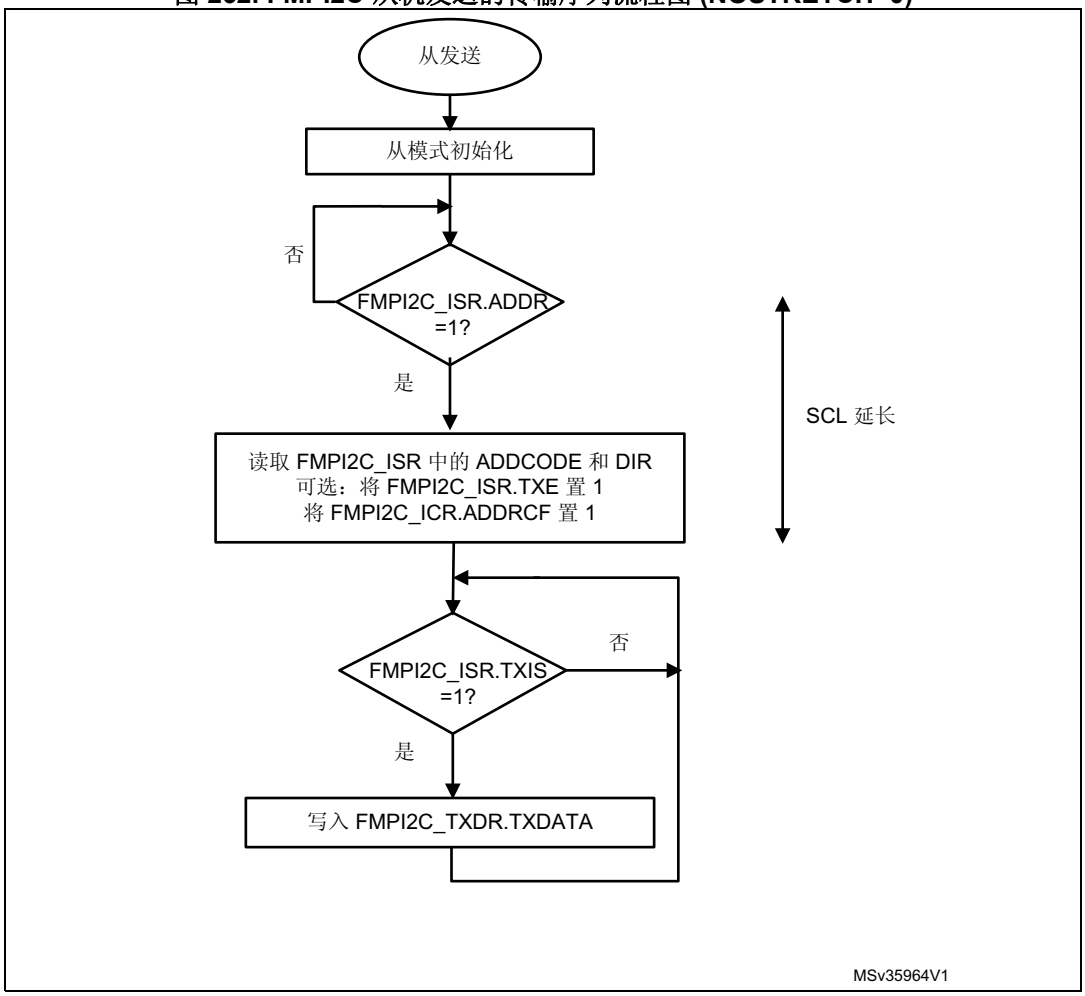


图 263. FMPI2C 从机发送的传输序列流程图 (NOSTRETCH=1)

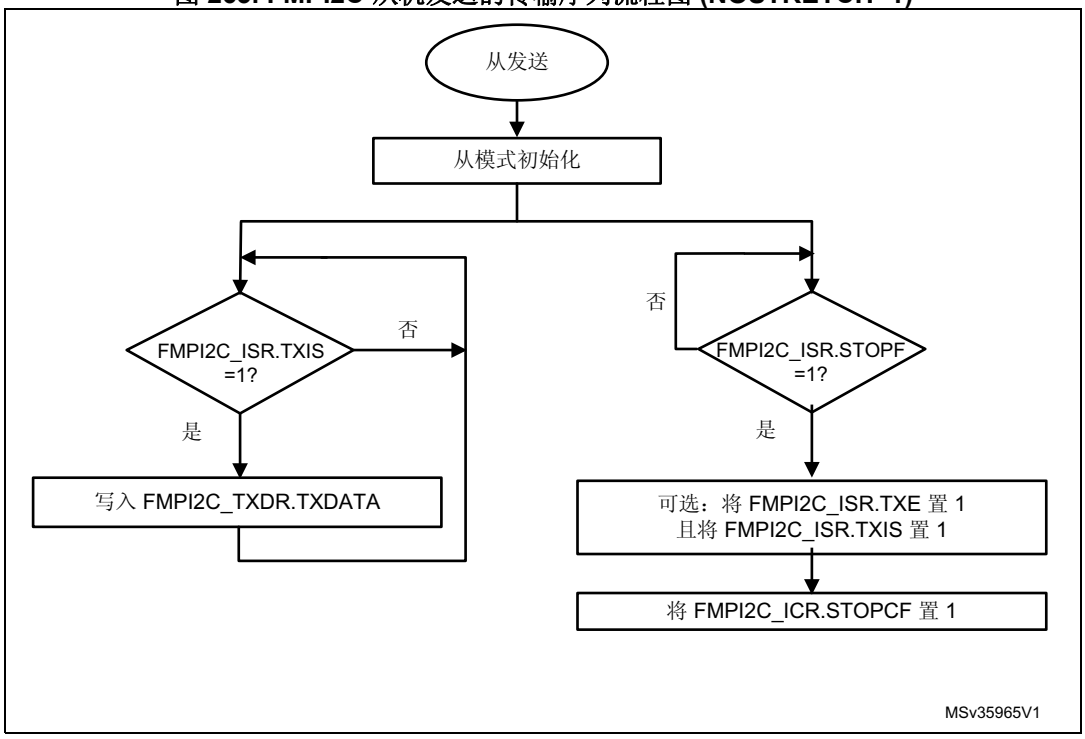
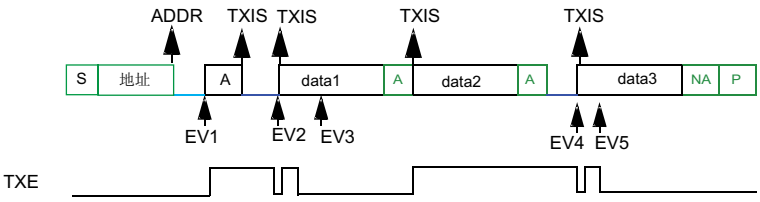


图 264. FMPI2C 从机发送的传输总线图

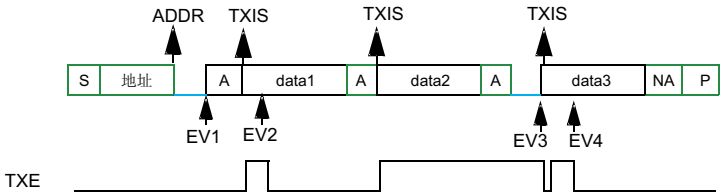
示例：FMPI2C 从机发送 3 个字节，刷新第 1 个数据
NOSTRETCH=0：



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）

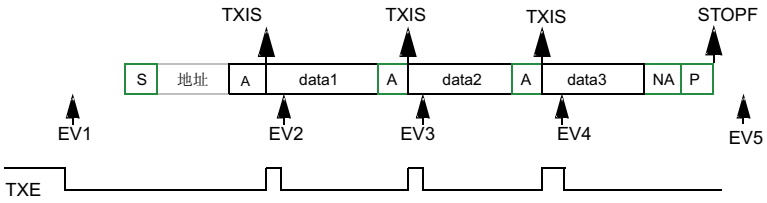
示例：FMPI2C 从机发送 3 个字节，刷新第 1 个数据，
NOSTRETCH=0：



图注：
发送
接收
SCL 延长

EV1：ADDR ISR：检查 ADDCODE 和 DIR，将 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：FMPI2C 从机发送 3 个字节，NOSTRETCH=1：



图注：
发送
接收
SCL 延长

EV1：写入数据 1
EV2：TXIS ISR：写入 data2
EV3：TXIS ISR：写入 data3
EV4：TXIS ISR：写入 data4（不发送）
EV5：STOPF ISR：（可选：将 TXE 和 TXIS 置 1），将 STOPCF 置 1

MS35975V1

从机接收

当 FMPI2C_RXDR 满时，FMPI2C_ISR 中的 RXNE 将置 1，如果 FMPI2C_CR1 中的 RXIE 置 1，还将生成中断。读取 FMPI2C_RXDR 时，将清零 RXNE。

接收到 STOP 条件且FMPI2C_CR1 寄存器中的 STOPIE 置 1 时，FMPI2C_ISR 中的 STOPF 将置 1 并且会生成中断。

图 265. 从机接收的传输序列流程图 (NOSTRETCH=0)

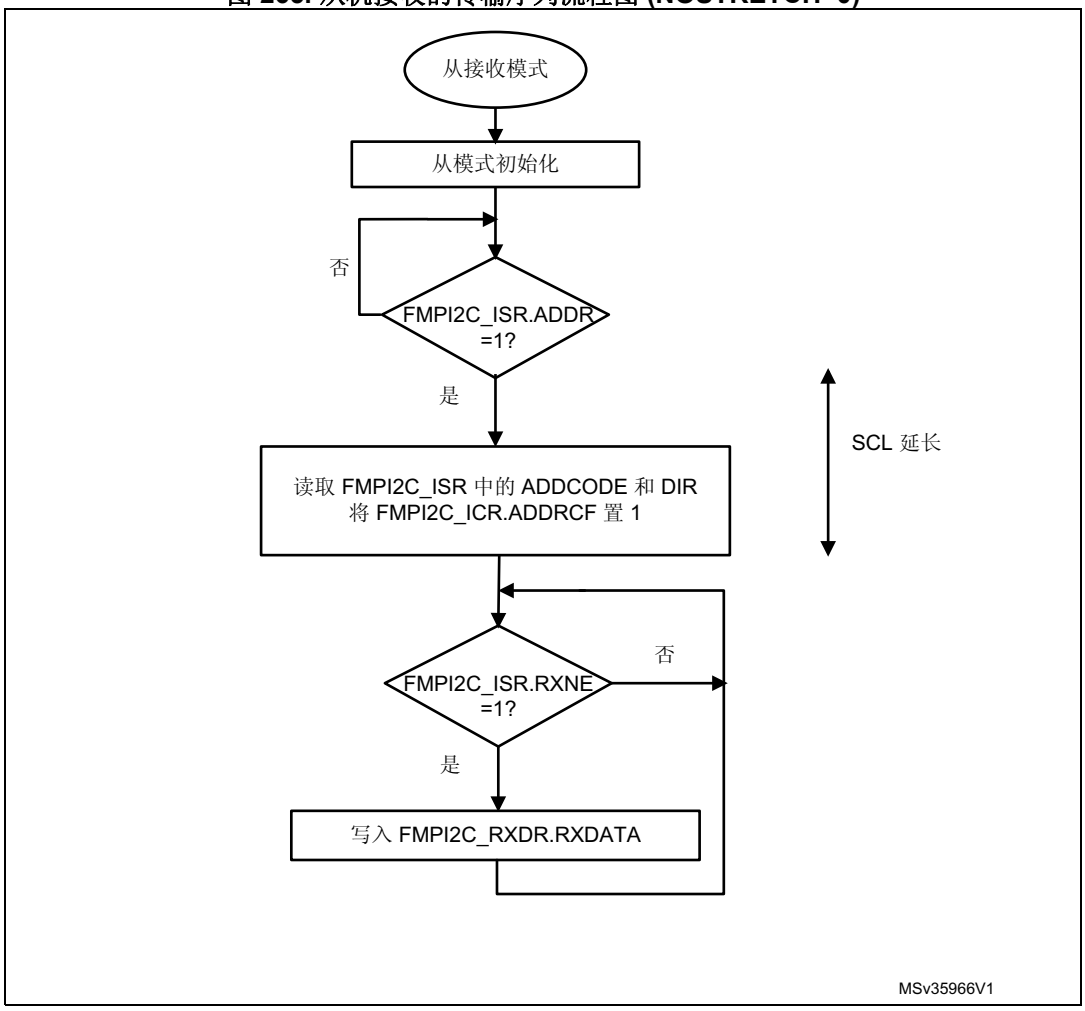


图 266. 从机接收的传输序列流程图 (NOSTRETCH=1)

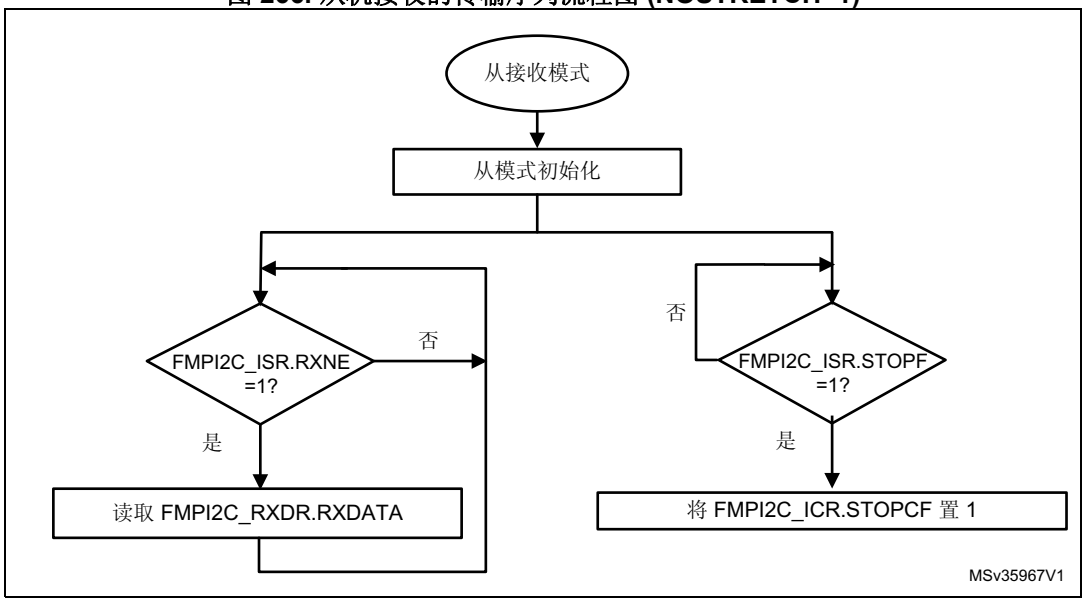
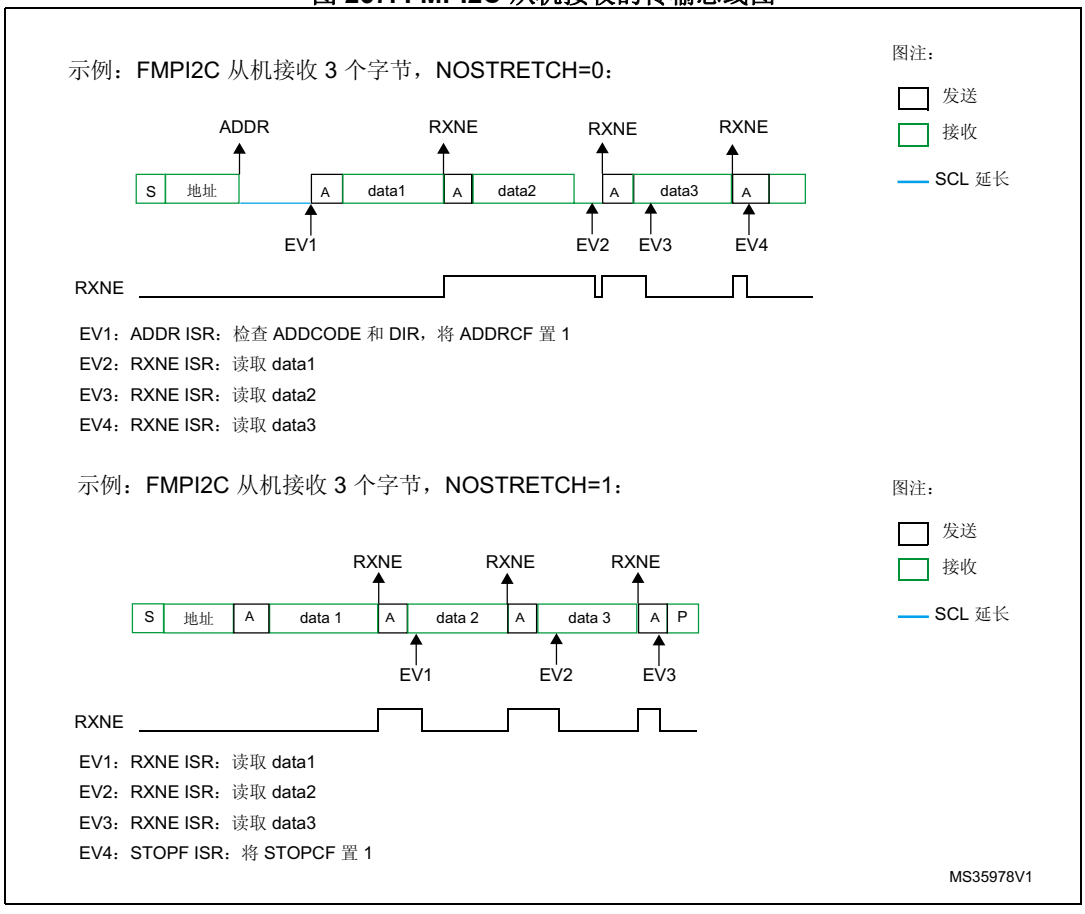


图 267. FMPI2C 从机接收的传输总线图



26.4.8 FMPI2C 主模式

FMPI2C 主模式初始化

使能外设前，必须通过设置 FMPI2C_TIMINGR 寄存器中的 SCLH 和 SCLL 位来配置 FMPI2C 主时钟。

STM32CubeMX 工具计算 I2C_TIMINGR 内容并在 I2C 配置窗口中进行显示。

为了支持多主环境和从时钟延展，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。

FMPI2C 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）并将 SCL 与 I2CxCLK 时钟的同步。一旦 SCLL 计数器达到 FMPI2C_TIMINGR 寄存器的 SCLL[7:0] 位中编程的值，FMPI2C 便会将 SCL 释放为高电平。

FMPI2C 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）并将 SCL 与 I2CxCLK 时钟的同步。一旦 SCLH 计数器达到 FMPI2C_TIMINGR 寄存器的 SCLH[7:0] 位中编程的值，FMPI2C 便会使 SCL 变为低电平。

因此，主时钟周期为：

$$t_{SCL} = t_{SYNC1} + t_{SYNC2} + \{[(SCLH+1) + (SCLL+1)] \times (PRESC+1) \times t_{I2CCLK}\}$$

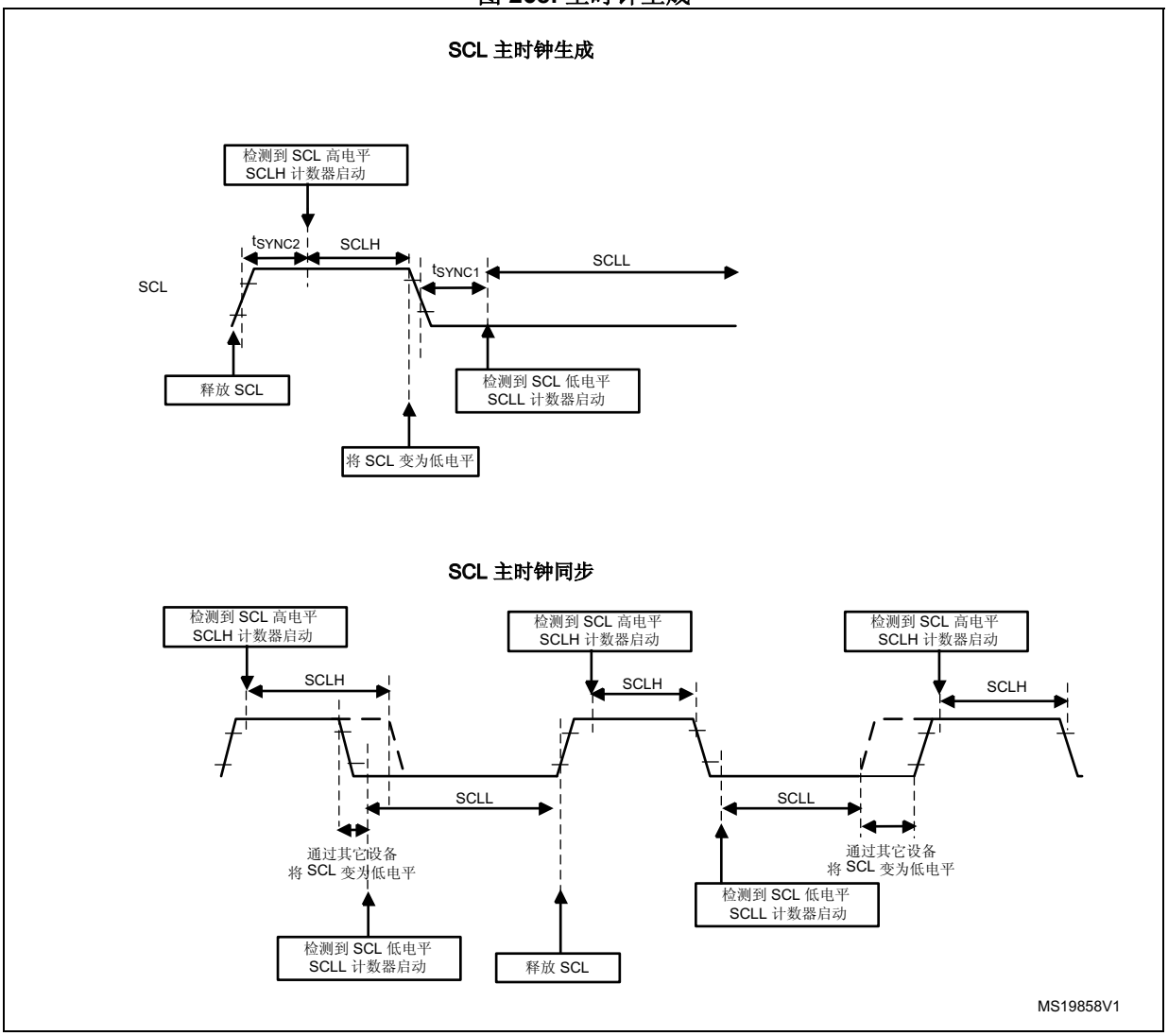
t_{SYNC1} 的持续时间取决于以下参数：

- 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 FMPI2CCLKI2CCLK 时钟建立同步而产生的延时（2 到 3 个 FMPI2CCLK 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 FMPI2CCLK 时钟建立同步而产生的延时（2 到 3 个 FMPI2CCLK 周期）

图 268. 主时钟生成



注意： 为了符合 I^2C 或 SMBus 规范，主时钟必须遵循下表中给出的时序：

表 142. 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
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 FMPI2C_TIMINGR 设置与 FMPI2CCLK 频率的示例，请参见*第 26.4.9 节：FMPI2C_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，用户必须在 FMPI2C_CR2 寄存器中为寻址的从器件编程以下参数：

- 寻址模式（7 位或 10 位）：ADD10
- 待发送的从地址：SADD[9:0]
- 传输方向：RD_WRN
- 读取 10 位地址时：HEAD10R 位。必须对 HEAD10R 进行相应配置，以指示传输方向变化时必须发送完整的地址序列，还是只发送地址头。
- 待传输的字节数：NBYTES[7:0]。如果字节数等于或大于 255，则初始化时必须将 NBYTES[7:0] 填充为 0xFF。

然后，用户必须将 FMPI2C_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。

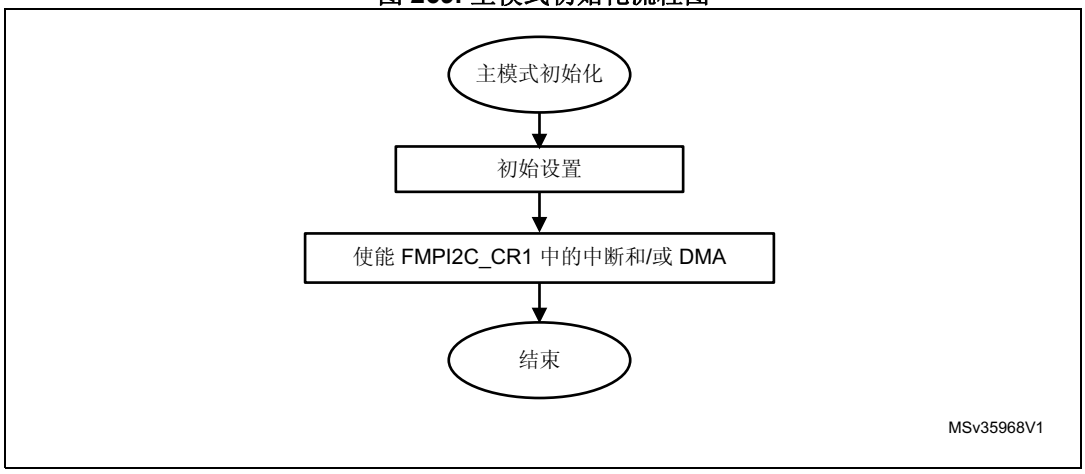
之后，当主器件检测到总线空闲 (BUSY = 0) 时，它会在经过 t_{BUF} 的延时后自动发送起始位，随后发出从机地址。

仲裁丢失时，主器件将自动切换回从模式，如果从机地址被选中，还将会自动发送 ACK 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。*
在 10 位寻址模式下，如果从器件不对从地址的前 7 位进行应答，则主器件将自动重新启动从地址发送，直至接收到 ACK。在这种情况下，如果从从器件接收到 NACK，则必须将 ADDRCF 置 1，以停止发送从地址。
如果当 START 位置 1 时，FMPI2C 作为从器件 (ADDR=1) 被寻址，则 FMPI2C 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。*

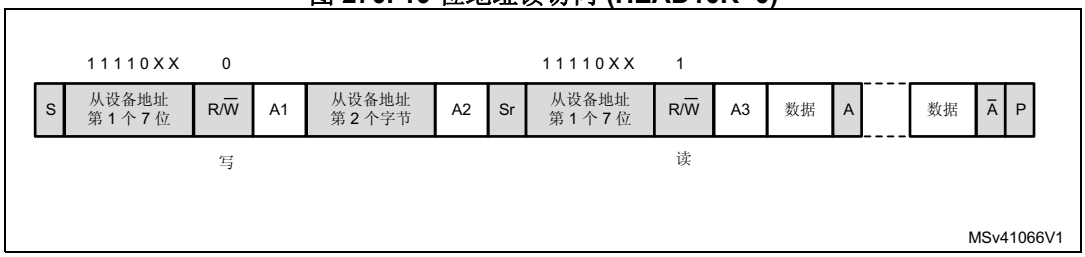
图 269. 主模式初始化流程图



主机接收器寻址 10 位地址从器件的初始化过程

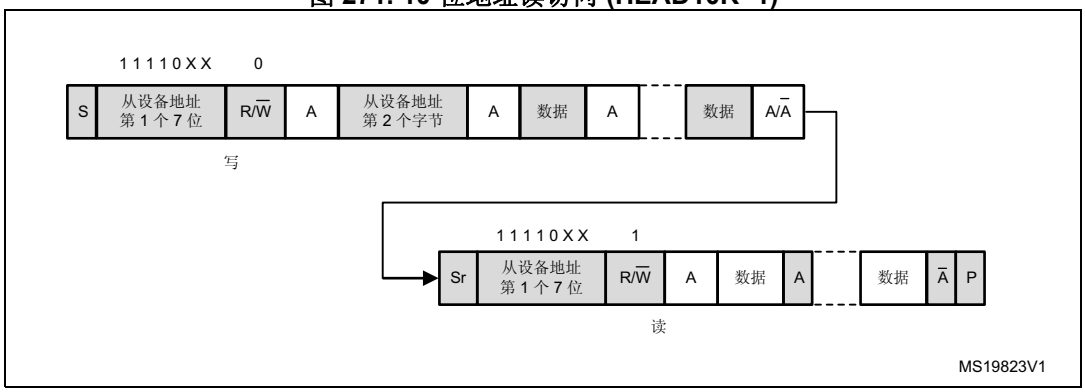
- 如果从地址采用 10 位格式，用户可选择将 FMPI2C_CR2 寄存器中的 HEAD10R 位清零来发送完整的读序列。在这种情况下，主器件会在 START 位置 1 后自动发送以下完整序列：（重复）起始位 + 带写方向的从器件 10 位地址头字节 + 从器件地址第 2 个字节 + 重复起始位 + 带读方向的从器件 10 位地址头字节。

图 270. 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 271. 10 位地址读访问 (HEAD10R=1)



主机发送

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 FMPI2C_CR1 寄存器中的 TXIE 位置 1，TXIS 事件将生成中断。当FMP FMPI2C_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 FMPI2C_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 FMPI2C_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 FMPI2C_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。FMPI2C_ISR 寄存器中的 NACKF 标志置 1，如果 NACKIE 位置 1，还将生成中断。

图 272. FMPI2C 主机发送的传输序列流程图（N ≤ 255 字节）

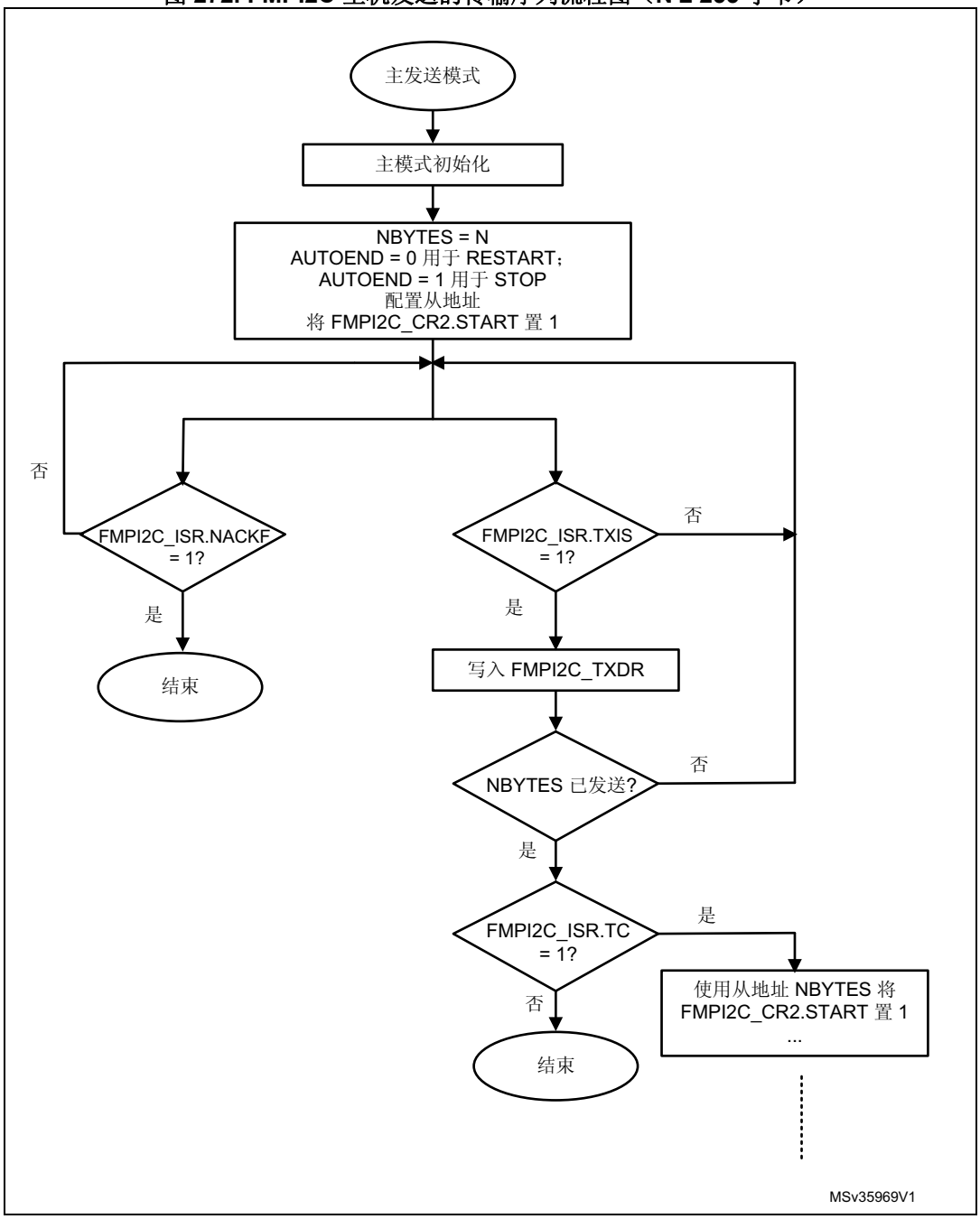


图 273. FMPI2C 主机发送的传输序列流程图（N>255 字节）

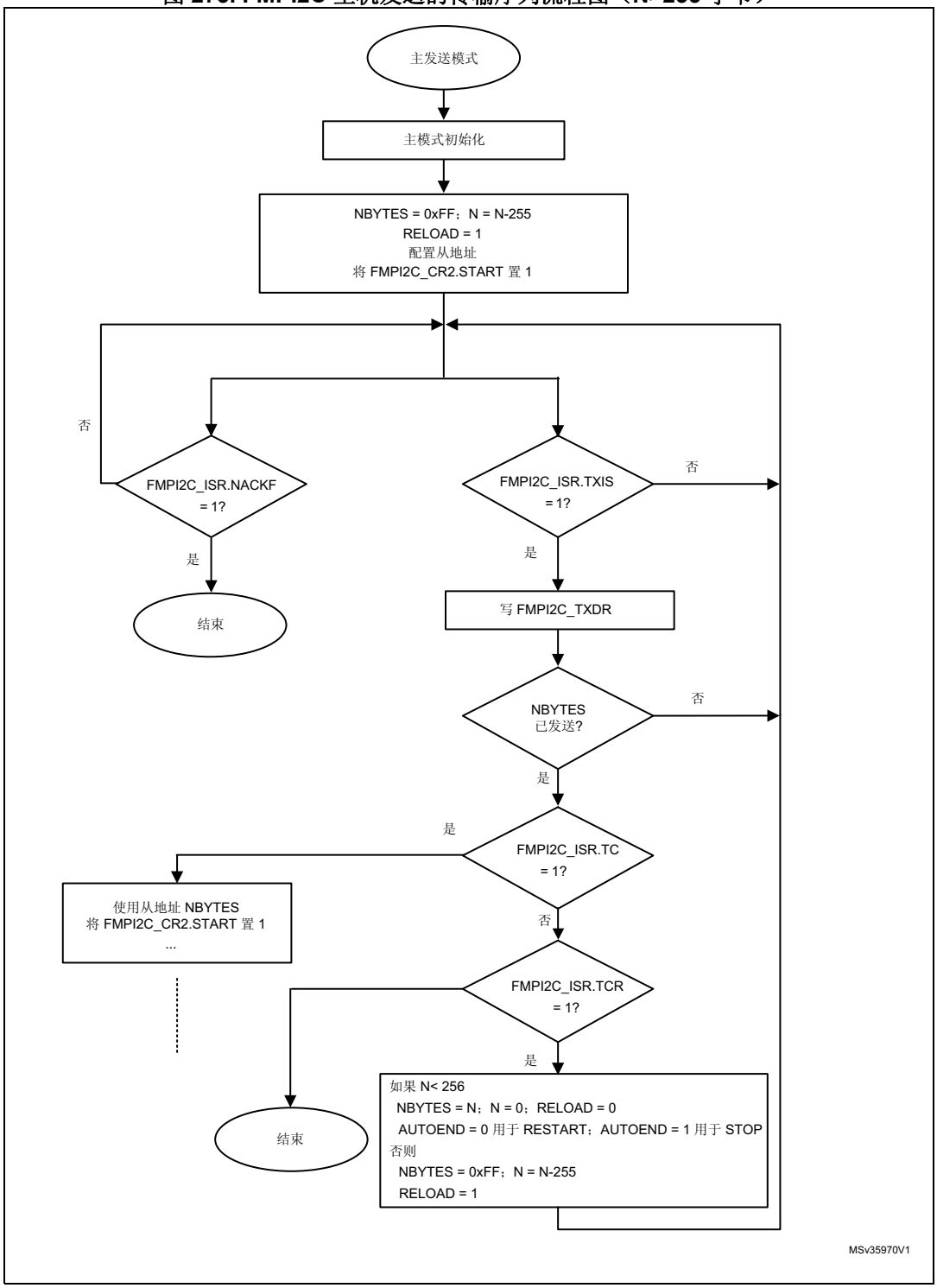
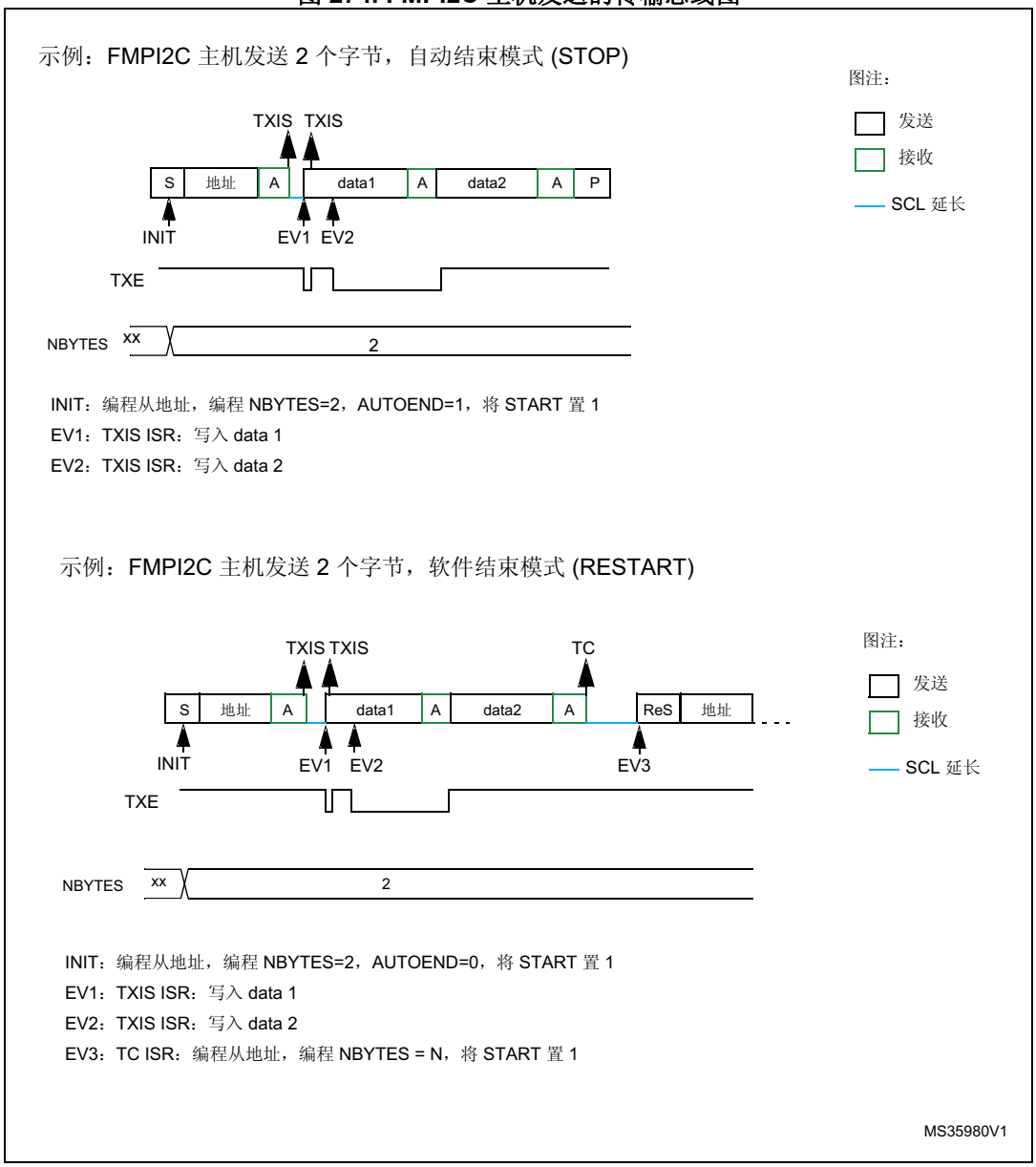


图 274. FMPI2C 主机发送的传输总线图



主机接收器

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 FMPI2C_CR1 寄存器中的 RXIE 位置 1，RXNE 事件将生成中断。读取 FMPI2C_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 FMPI2C_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 FMPI2C_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 FMPI2C_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 275. FMPI2C 主机接收的传输序列流程图（N ≤ 255 字节）

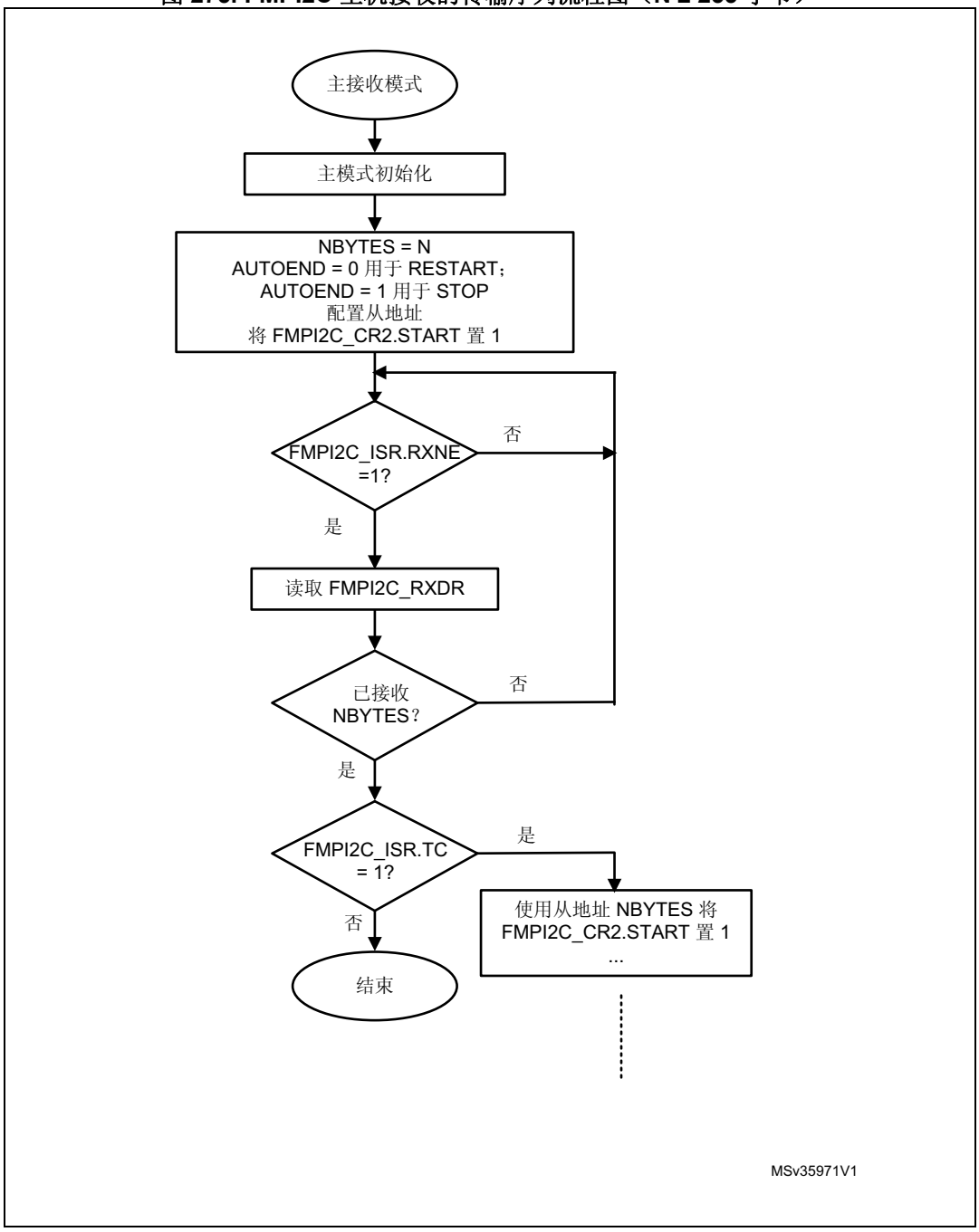


图 276. FMPI2C 主机接收的传输序列流程图（N>255 字节）

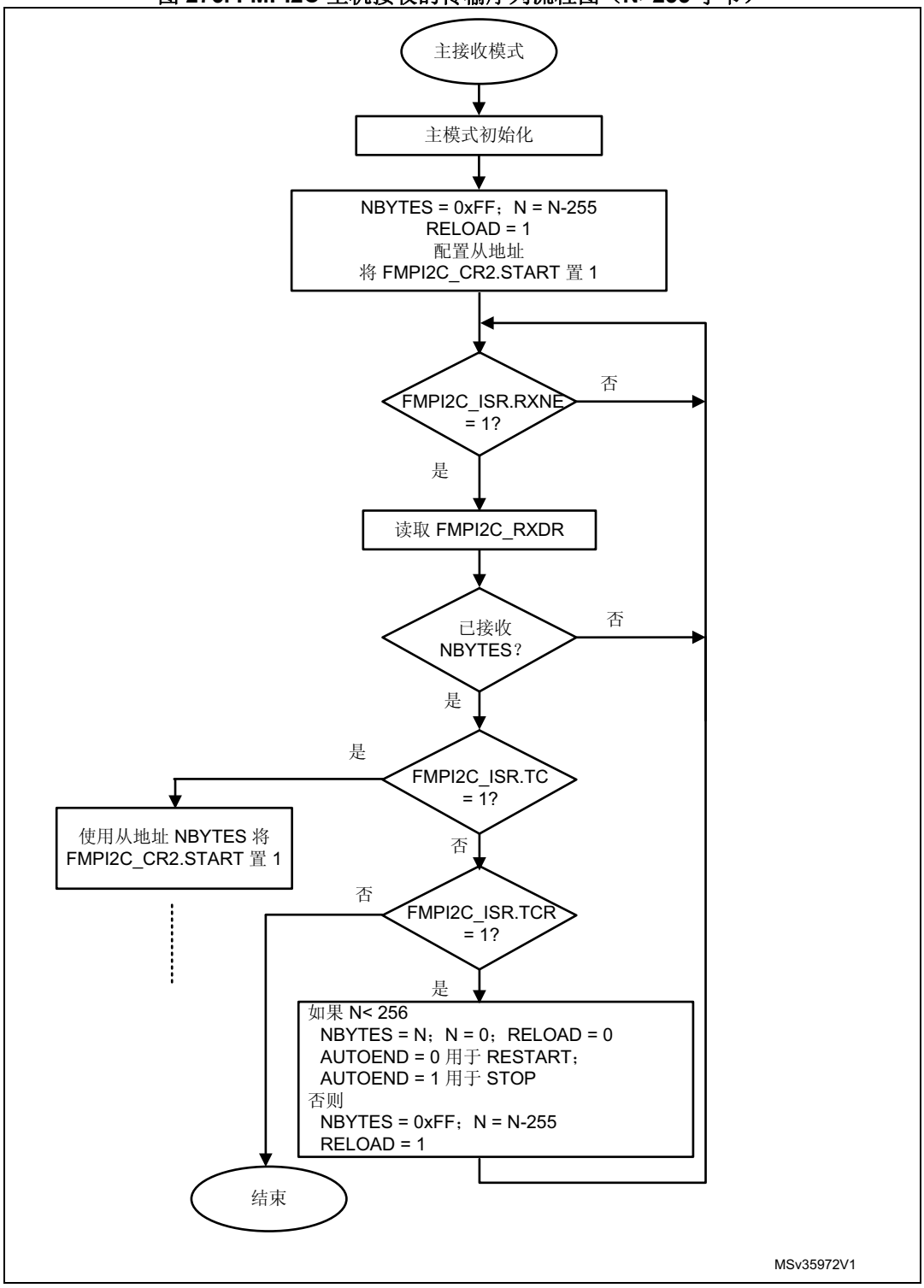
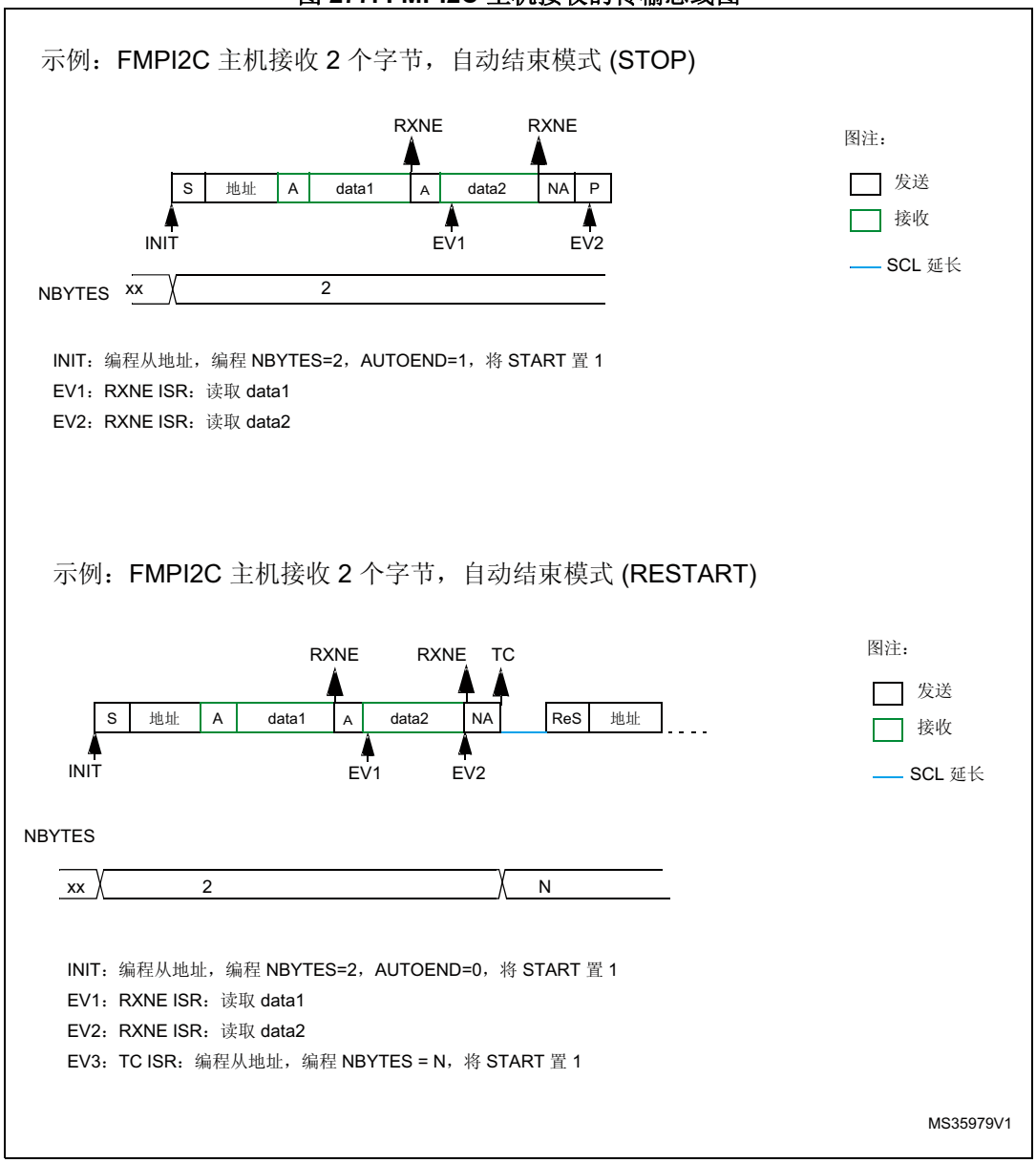


图 277. FMPI2C 主机接收的传输总线图



26.4.9 FMPI2C_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 FMPI2C_TIMINGR 才能获得符合 I²C 规范的时序。要获取更准确的配置值，应使用 STM32CubeMX 工具（I2C 配置窗口）。

表 143. f_{I2CCLK} = 8 MHz 时的时序设置示例

参数	标准模式 (Sm)		快速模式 (Fm)	超快速模式 (Fm+)
	10 kHz	100 kHz	400 kHz	500 kHz
PRESC	1	1	0	0
SCLL	0xC7	0x13	0x9	0x6
t_{SCLL}	200x250 ns = 50 µs	20x250 ns = 5.0 µs	10x125 ns = 1250 ns	7x125 ns = 875 ns
SCLH	0xC3	0xF	0x3	0x3
t_{SCLH}	196x250 ns = 49 µs	16x250 ns = 4.0µs	4x125ns = 500ns	4x125 ns = 500 ns
t_{SCL}(1)	~100 µs(2)	~10 µs(2)	~2500 ns(3)	~2000 ns(4)
SDADEL	0x2	0x2	0x1	0x0
t_{SDADEL}	2x250 ns = 500 ns	2x250 ns = 500 ns	1x125 ns = 125 ns	0 ns
SCLDEL	0x4	0x4	0x3	0x1
t_{SCLDEL}	5x250 ns = 1250 ns	5x250 ns = 1250 ns	4x125 ns = 500 ns	2x125 ns = 250 ns

1. 由于 SCL 内部检测存在延时， SCL 周期 t_{SCL} 大于 t_{SCLL} + t_{SCLH}。为 t_{SCL} 提供的值仅用于举例说明。
2. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 500 ns。 t_{SYNC1} + t_{SYNC2} = 1000 ns 时的示例
3. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 500 ns。 t_{SYNC1} + t_{SYNC2} = 750 ns 时的示例
4. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 500 ns。 t_{SYNC1} + t_{SYNC2} = 655 ns 时的示例

表 144. f_{I2CCLK} = 16 MHz 时的时序设置示例

参数	标准模式 (Sm)		快速模式 (Fm)	超快速模式 (Fm+)
	10 kHz	100 kHz	400 kHz	1000 kHz
PRESC	3	3	1	0
SCLL	0xC7	0x13	0x9	0x4
t_{SCLL}	200 x 250 ns = 50 µs	20 x 250 ns = 5.0 µs	10 x 125 ns = 1250 ns	5 x 62.5 ns = 312.5 ns
SCLH	0xC3	0xF	0x3	0x2
t_{SCLH}	196 x 250 ns = 49 µs	16 x 250 ns = 4.0 µs	4 x 125ns = 500 ns	3 x 62.5 ns = 187.5 ns
t_{SCL}(1)	~100 µs(2)	~10 µs(2)	~2500 ns(3)	~1000 ns(4)
SDADEL	0x2	0x2	0x2	0x0
t_{SDADEL}	2 x 250 ns = 500 ns	2 x 250 ns = 500 ns	2 x 125 ns = 250 ns	0 ns
SCLDEL	0x4	0x4	0x3	0x2
t_{SCLDEL}	5 x 250 ns = 1250 ns	5 x 250 ns = 1250 ns	4 x 125 ns = 500 ns	3 x 62.5 ns = 187.5 ns

1. 由于 SCL 内部检测存在延时， SCL 周期 t_{SCL} 大于 t_{SCLL} + t_{SCLH}。为 t_{SCL} 提供的值仅用于举例说明。
2. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 250 ns。 t_{SYNC1} + t_{SYNC2} = 1000 ns 时的示例
3. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 250 ns。 t_{SYNC1} + t_{SYNC2} = 750 ns 时的示例
4. t_{SYNC1} + t_{SYNC2} 最小值为 4 x t_{I2CCLK} = 250 ns。 t_{SYNC1} + t_{SYNC2} = 655 ns 时的示例

26.4.10 SMBus 特性

仅当支持 SMBus 功能时，才涉及本节内容。请参见*第 26.3 节：FMPI2C 特性实现*。

简介

系统管理总线 (SMBus) 是一个双线制接口，各器件可通过它在彼此之间或者与系统的其余部分进行通信。它以 I^2C 的工作原理为基础。SMBus 可针对系统和电源管理相关的任务提供控制总线。

该外设与 SMBUS 规范兼容 (http://smbus.org)。

系统管理总线规范涉及三类器件。

- 从机，用于接收或响应命令。
- 主机，用于发出命令、生成时钟和中止传输。
- HOST 是一个特殊的主机，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 HOST 通知协议。系统中只允许存在一个主机。

本外设可以配置为主机或从机，当然也可以作为 HOST。

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 (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 FMPI2C_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。

ARP 支持的仲裁动作也是在从机模式下完成。

有关 SMBus 地址解析协议的详细信息，请参见 SMBus 规范(http://smbus.org)。

接收的命令和数据应答控制

SMBus 接收器必须能够对接收到的每个命令或数据进行 NACK 应答。要在从模式下实现 ACK 控制，必须通过将 FMPI2C_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 748 页的从器件字节控制模式*。

HOST 通知协议

该外设通过将 FMPI2C_CR1 寄存器中的 SMBHEN 位置 1 来支持 HOST 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，本设备会作为主机，而 HOST 则会作为一个从机。

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 FMPI2C_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，FMPI2C_ISR 寄存器中的 ALERT 标志置 1。如果 FMPI2C_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 NACK 应答信号。

超时

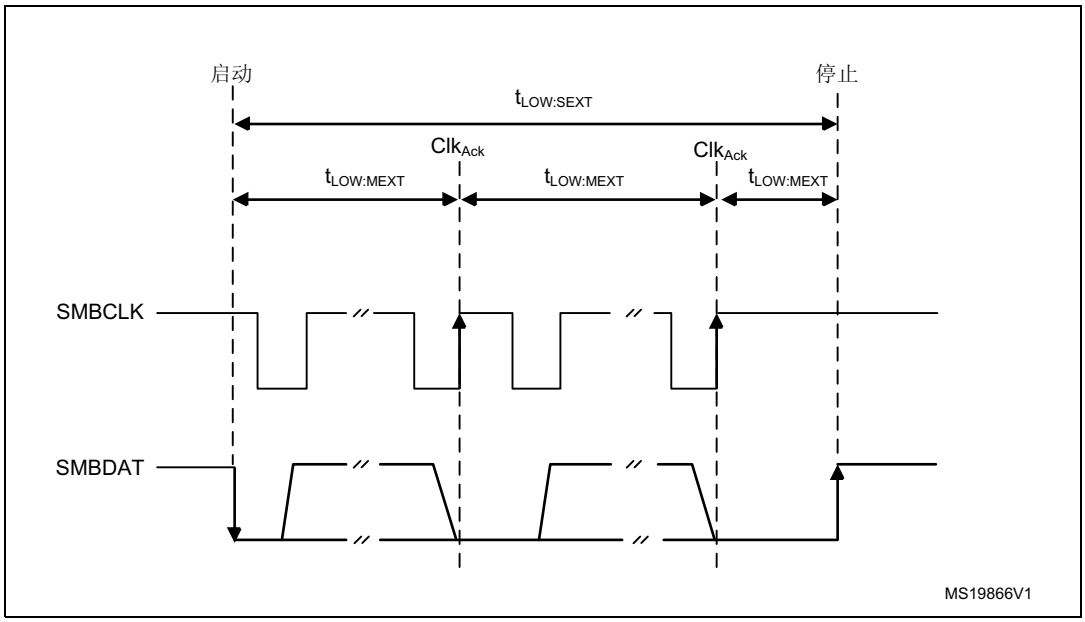
该外设内置了硬件定时器，以便符合 SMBus 规范中定义的 3 个超时。

表 145. 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)	累积时钟低电平延展时间（从器件）	-	25	ms
$t_{LOW:MEXT}$(2)	累积时钟低电平延展时间（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。其他从器件或主器件也可能延展时钟，进而导致时钟低电平总延展时间超过 $t_{LOW:SEXT}$。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。从器件或其他主器件也可能延展时钟，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 278. $t_{LOW:SEXT}$ 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 140：I2C-SMBUS 规范数据建立和保持时间*）。

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。

26.4.11 SMBus 初始化

仅当支持 SMBus 功能时，才涉及本节内容。请参见*第 26.3 节：FMPI2C 特性实现*。

除了 FMPI2C 初始化之外，还必须进行一些其他的特定初始化，以便执行 SMBus 通信：

接收的命令和数据应答控制（从模式）

SMBus 接收器必须能够对接收到的每个命令或数据进行 NACK 应答。要在从模式下实现 ACK 控制，必须通过将 FMPI2C_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 748 页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 770 页的总线空闲检测*。

- 通过将 FMPI2C_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 FMPI2C_CR1 寄存器中的 SMBHEN 位置 1 来使能 SMBus 主机地址 (0b0001 000)。
- 通过将 FMPI2C_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 FMPI2C_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（FMPI2C_CR2 寄存器中的 NBYTES[7:0]）来管理 PEC 传输。使能 FMPI2C 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。

注意： 使能 FMPI2C 时，不允许更改 PECEN 配置。

表 146. 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 FMPI2C_TIMEOUTR 寄存器中的 TIMOUTEN 和 TEXTEN 位置 1 来使能超时检测。定时器必须按如下方式编程：即在 SMBus 规范规定的时间最大值之前检测出超时情况。

- $t_{TIMEOUT}$ 检查
  要使能 $t_{TIMEOUT}$ 检查，必须将 12 位 TIMEOUTA[11:0] 位编程为定时器重载值，以检查 $t_{TIMEOUT}$ 参数。必须将 TIDLE 位配置为“0”，以检测 SCL 低电平超时。
  然后，通过将 FMPI2C_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}，FMPI2C_ISR 寄存器中的 TIMEOUT 标志将置 1。
  请参见*表 147：不同 FMPI2CCLK 频率下的 TIMEOUTA 设置示例（最大 tTIMEOUT = 25 ms）*。

注意： 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，用户可以为这两个参数选择相同的值。
  然后，通过将 FMPI2C_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 770 页的总线空闲检测*一节给出的超时间隔，则 FMPI2C_ISR 寄存器中的 TIMEOUT 标志将置 1。
  请参见*表 148：不同 FMPI2CCLK 频率下的 TIMEOUTB 设置示例*。

注意： 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 FMPI2C_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}，FMPI2C_ISR 寄存器中的 TIMEOUT 标志将置 1。

请参见*表 149：不同 FMPI2CCLK 频率下的 TIMEOUTA 设置示例（最大 tIDLE = 50 µs）*。

注意： TIMEOUTEN 置 1 时，不允许更改 TIMEOUTA 和 TIDLE 配置。

26.4.12 SMBus：FMPI2C_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 26.3 节：FMPI2C 特性实现*。

- 将 $t_{TIMEOUT}$ 的最大持续时间配置为 25 ms：

表 147. 不同 FMPI2CCLK 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 148. 不同 FMPI2CCLK 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 149. 不同 FMPI2CCLK 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

26.4.13 SMBus 从模式

仅当支持 SMBus 功能时，才涉及本节内容。请参见*第 26.3 节：FMPI2C 特性实现*。

除了 FMPI2C 从模式传输管理（请参见*第 26.4.7 节：FMPI2C 从模式*）之外，还提供了一些额外的软件流程图来支持 SMBus。

SMBus 从机发送

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，将 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 FMPI2C_PECR 寄存器的内容。

注意： 当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

图 279. SMBus 从机发送的传输序列流程图（N 字节 + PEC）

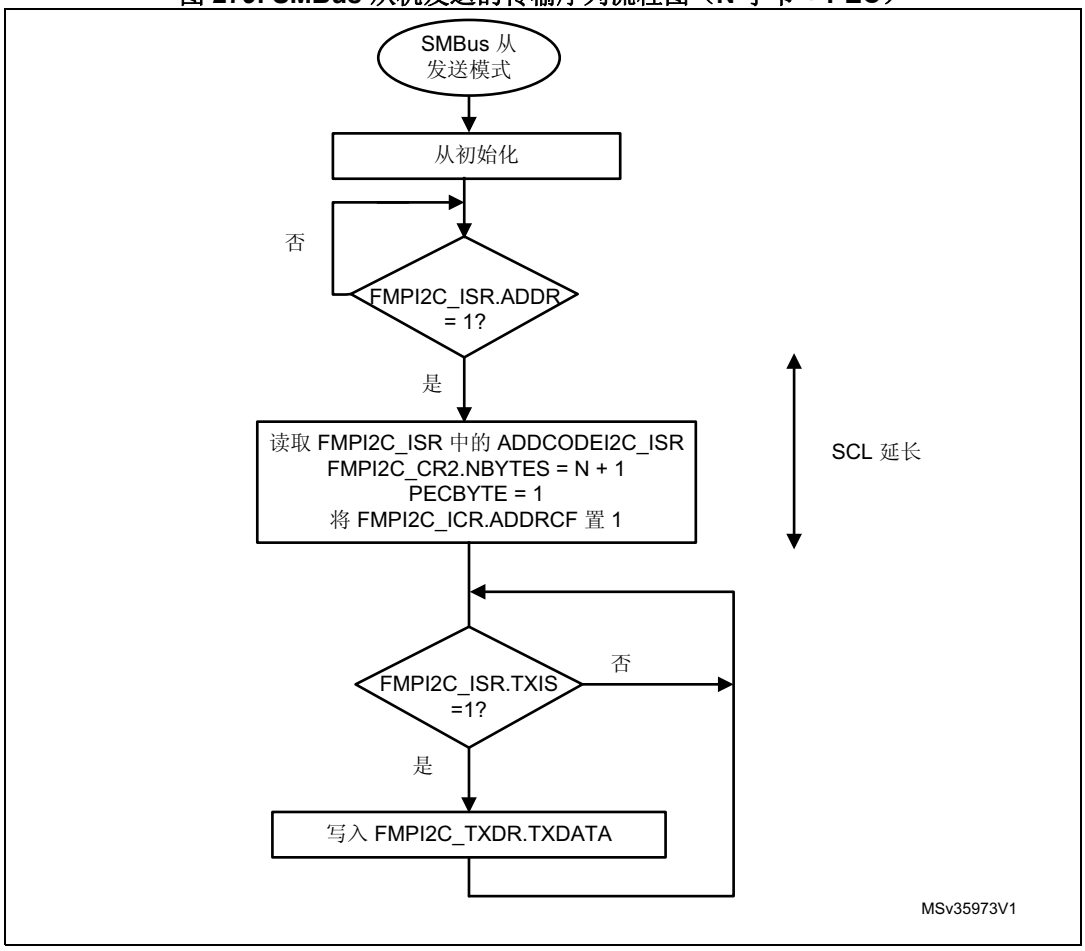
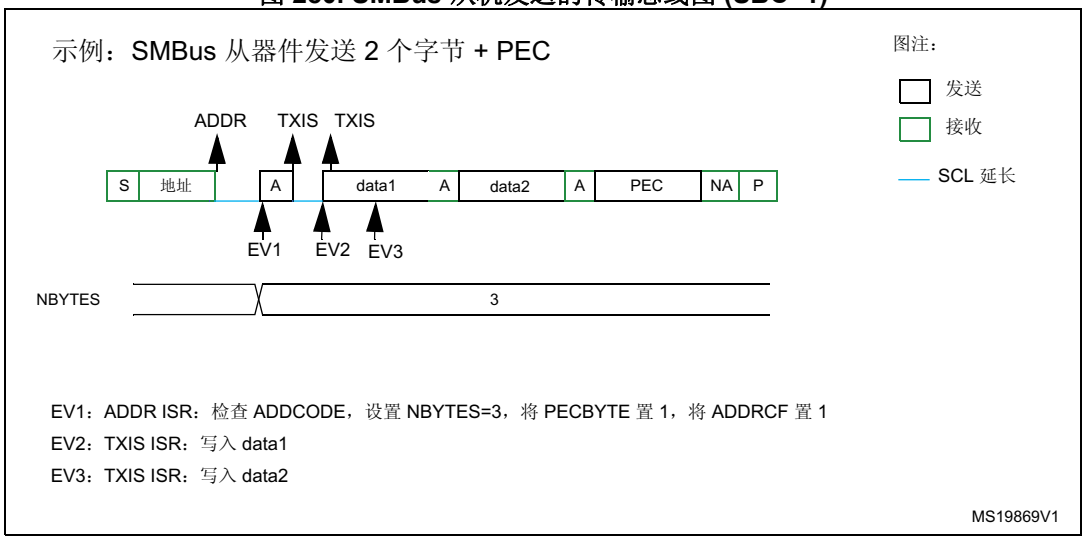


图 280. SMBus 从机发送的传输总线图 (SBC=1)



SMBus 从机接收

在 SMBus 模式下使用 FMPI2C 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 748 页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 FMPI2C_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，便会像任何其他数据一样复制到 FMPI2C_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 FMPI2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

如果无需 ACK 软件控制，用户可编程 PECBYTE=1，在同一写操作下，将 NBYTES 编程为连续接收的字节数。接收到 NBYTES-1 字节的数据后，会将接收的下一个字节视为 PEC 进行校验。

注意： 当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

图 281. SMBus 从机接收的传输序列流程图（N 字节 + PEC）

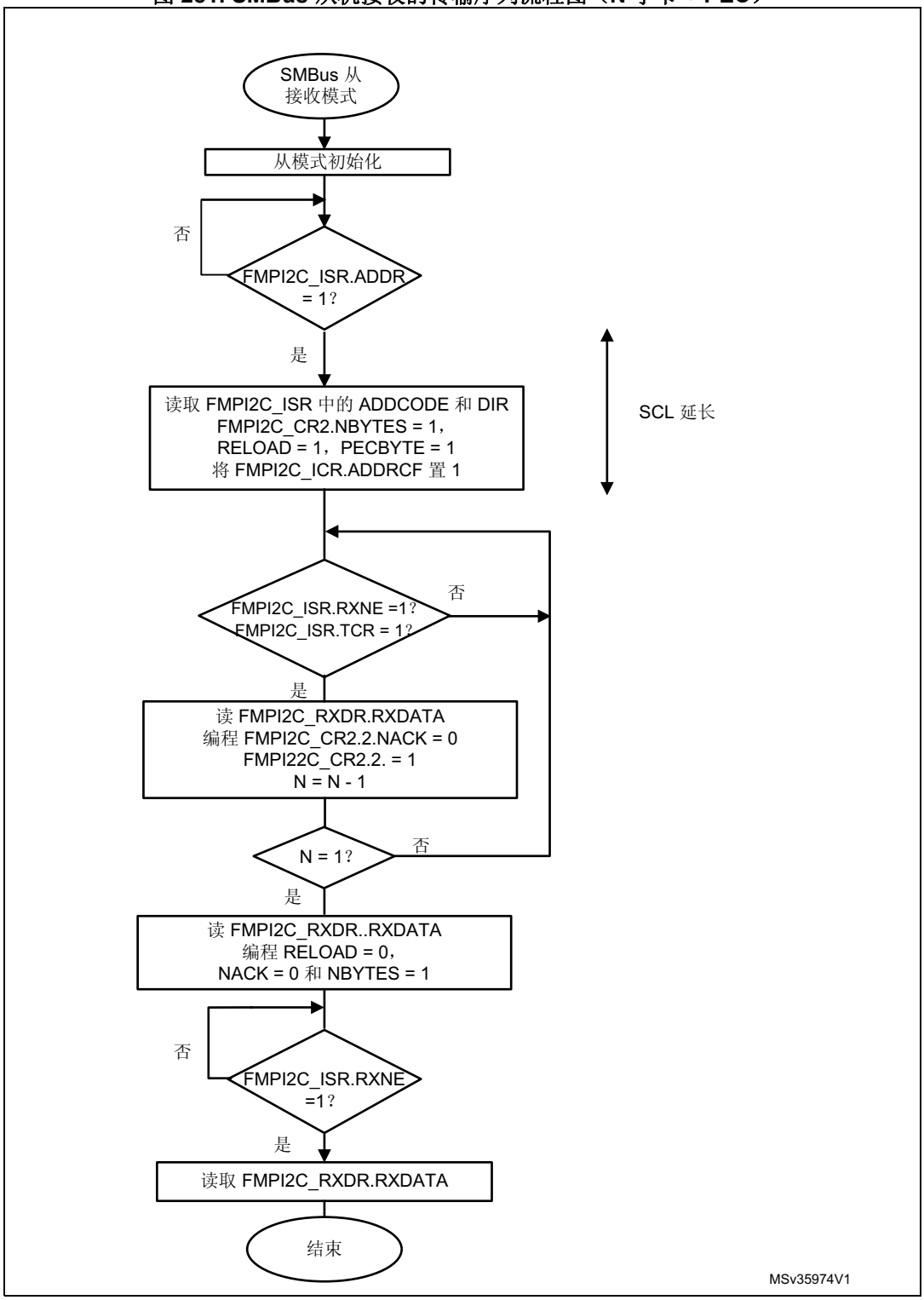
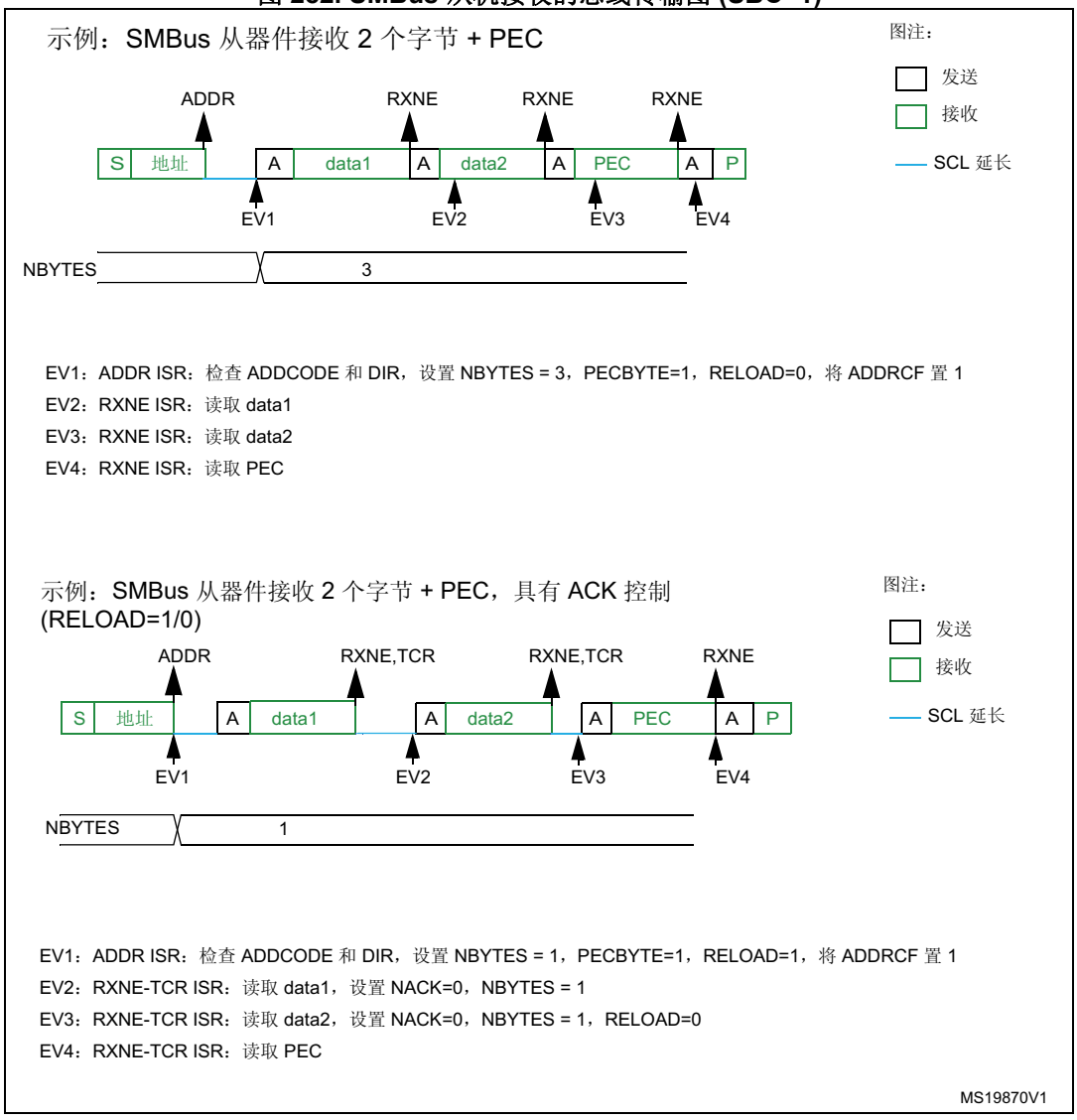


图 282. SMBus 从机接收的总线传输图 (SBC=1)



仅当支持 SMBus 功能时，才涉及本节内容。请参见*第 26.3 节：FMPI2C 特性实现*。

除了 FMPI2C 主模式传输管理（请参见*第 26.4.8 节：FMPI2C 主模式*）之外，还提供了一些额外的软件流程图来支持 SMBus。

SMBus 主机发送

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 FMPI2C_PECR 寄存器的内容。

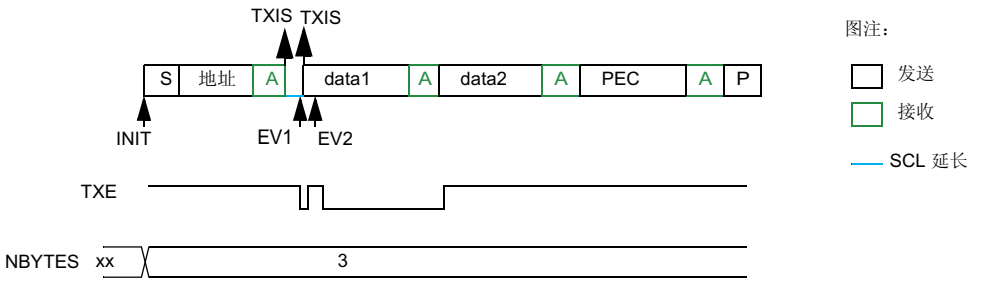
如果 SMBus 主器件想要在 PEC 后发送停止位，则应选择自动结束模式 (AUTOEND=1)。在这种情况下，传输 PEC 后将自动发送停止位。

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 FMPI2C_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展。必须在 TC 中断子程序中设置重复起始位。

注意： 当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

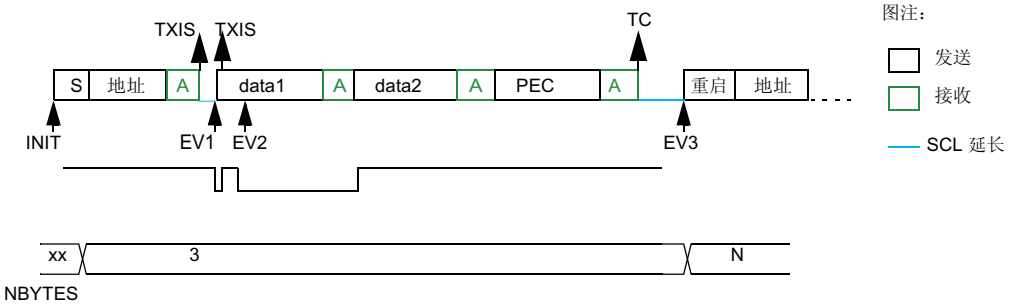
图 283. SMBus 主机发送的总线传输图

示例：SMBus 主器件发送 2 个字节 + PEC，自动结束模式 (STOP)



INIT：设置从地址，设置 NBYTES = 3，AUTOEND=1，将 PECBYTE 置 1，将 START 置 1
EV1：TXIS ISR：写入 data1
EV2：TXIS ISR：写入 data2

示例：SMBus 主器件发送 2 个字节 + PEC，软件结束模式 (RESTART)



INIT：设置从地址，设置 NBYTES = 3，AUTOEND=0，将 PECBYTE 置 1，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

MS19871V1

SMBus 主机接收

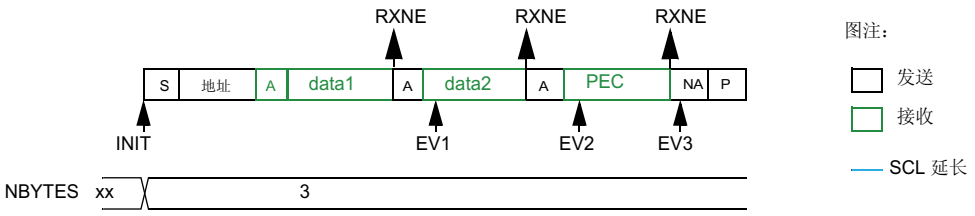
当 SMBus 主器件想要接收 PEC，并在传输结束后接收 STOP 时，可选择自动结束模式 (AUTOEND=1)。将 START 位置 1 之前，必须将 PECBYTE 位置 1 并设置从地址。在这种情况下，当接收到 NBYTES-1 字节的数据后，将自动使用 FMPI2C_PECR 寄存器的内容对接收的下一个字节进行校验。PEC 字节（其后跟有停止位）将得到 NACK 响应。

当 SMBus 主机接收想要接收 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 FMPI2C_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展。可以在 TC 中断子程序中设置重复起始位。

注意： 当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

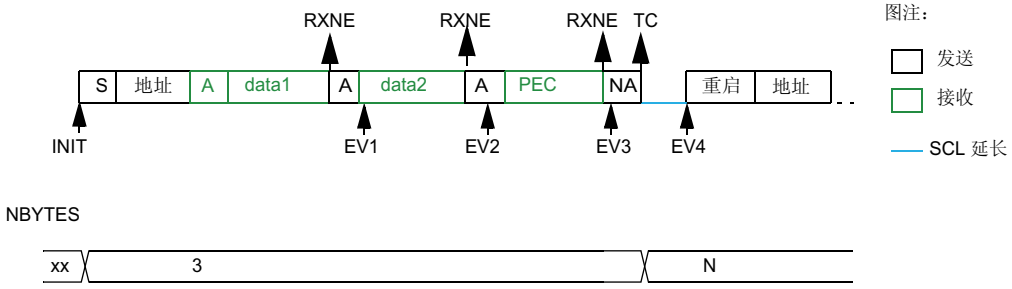
图 284. SMBus 主机接收的总线传输图

示例：SMBus 主器件接收 2 个字节 + PEC，自动结束模式 (STOP)



INIT：设置从地址，设置 NBYTES = 3，AUTOEND=1，将 PECBYTE 置 1，将 START 置 1
EV1：RXNE ISR：读取 data1
EV2：RXNE ISR：读取 data2
EV3：RXNE ISR：读取 PEC

示例：SMBus 主器件接收 2 个字节 + PEC，软件结束模式 (RESTART)



INIT：设置从地址，设置 NBYTES = 3，AUTOEND=0，将 PECBYTE 置 1，将 START 置 1
EV1：RXNE ISR：读取 data1
EV2：RXNE ISR：读取 data2
EV3：RXNE ISR：读取 PEC
EV4：TC ISR：设置从地址，设置 NBYTES = N，将 START 置 1

MS19872V1

26.4.14 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 FMPI2C 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。

在从模式下误检测到起始位或重复起始位时，FMPI2C 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，FMPI2C_ISR 寄存器中的 BERR 标志将置 1，如果 FMPI2C_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，FMPI2C_ISR 寄存器中的 ARLO 标志将置 1，如果 FMPI2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

上溢/下溢错误 (OVR)

当满足 NOSTRETCH=1 和以下条件时，将在从模式下检测到上溢或下溢错误：

- 在接收过程中接收到一个新的字节，但 RXDR 寄存器的值还未被读取。接收的新字节丢失，自动发送 NACK 来响应新字节。
- 在发送过程中：
  - 当 STOPF=1 且应发送第一个数据字节时。TXE=0 时发送 FMPI2C_TXDR 寄存器的内容，否则发送 0xFF。
  - 应发送一个新字节但尚未向 FMPI2C_TXDR 寄存器写入数据时，将发送 0xFF。

检测到上溢或下溢错误时，FMPI2C_ISR 寄存器中的 OVR 标志将置 1，如果 FMPI2C_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 26.3 节：FMPI2C 特性实现*。

当接收到的 PEC 字节与 FMPI2C_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，FMPI2C_ISR 寄存器中的 PECERR 标志将置 1，如果 FMPI2C_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 26.3 节：FMPI2C 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展时间达到了 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展时间达到了 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，FMPI2C_ISR 寄存器中的 TIMEOUT 标志将置 1，如果 FMPI2C_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 26.3 节：FMPI2C 特性实现*。

当 FMPI2C 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 FMPI2C_CR1 寄存器中的 ERRIE 位置 1，将生成中断。

26.4.15 DMA 请求

使用 DMA 进行发送

将 FMPI2C_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 DMA 规范）装载进 FMPI2C_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 759 页的主机发送*。
- 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 773 页的SMBus 从机发送*和*第 776 页的SMBus 主机发送*。

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 FMPI2C_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 FMPI2C_RXDR 寄存器装载进由 DMA 外设配置的 SRAM 区（请参见 DMA 规范）。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 26.3 节：FMPI2C 特性实现*）：PEC 传输由 NBYTES 计数器管理。请参见*第 774 页的SMBus 从机接收*和*第 778 页的SMBus 主机接收*。

注： *如果使用 DMA 进行接收，则无需使能 RXIE 位。*

26.4.16 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_ 配置位选择继续正常工作还是停止工作。

26.5 FMPI2C 低功耗模式

表 150. 低功耗模式

模式	说明
睡眠	对 I2C 通信无影响 FMPI2C 中断可使器件退出睡眠模式。
停止	FMPI2C 寄存器的内容仍被保持。
待机	FMPI2C 外设掉电，退出待机模式后必须重新初始化。

26.6 FMPI2C 中断

下表给出了 FMPI2C 中断请求列表。

表 151. FMPI2C 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	退出睡眠模式	退出停止模式
I2C_EV	接收缓冲区非空	RXNE	RXIE	读取 FMPI2C_RXDR 寄存器	是	否
	发送缓冲区中断状态	TXIS	TXIE	写入 FMPI2C_TXDR 寄存器		
	停止检测中断标志	STOPF	STOPIE	写入 STOPCF=1		
	传输完成等待重载	TCR	TCIE	写入 FMPI2C_CR2 (NBYTES[7:0] ≠ 0)		
	传输完成	TC		写入 START=1 或 STOP=1		
	地址匹配	ADDR	ADDRIE	写入 ADDRCF=1		
	NACK 接收	NACKF	NACKIE	写入 NACKCF=1		
I2C_ER	总线错误	BERR	ERRIE	写入 BERRCF=1	是	否
	仲裁丢失	ARLO		写入 ARLOCF=1		
	上溢 / 下溢	OVR		写入 OVRCF=1		
	PEC 错误	PECERR		写入 PECERRCF=1		
	超时 /t_{LOW} 错误	TIMEOUT		写入 TIMEOUTCF=1		
	SMBus 报警	ALERT		写入 ALERTCF=1		

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（FMPI2C 全局中断），也可以分配到 2 个不同的中断向量上（FMPI2C 事件中断和 FMPI2C 错误中断）。有关详细信息，请参见*第 235 页的STM32F413/423 的向量表*。

26.7 FMPI2C 寄存器

有关寄存器说明中使用的缩写，请参见*第 50 页上的第 1.2 节*。

外设寄存器按字（32 位）进行访问。

26.7.1 控制寄存器 1 (FMPI2C_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 FMPI2CCLK。

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	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 26.3 节：FMPI2C 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 26.3 节：FMPI2C 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 26.3 节：FMPI2C 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 26.3 节：FMPI2C 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**：时钟延展禁止 (Clock stretching disable)
该位用于在从模式下禁止时钟延展。它在主模式下必须保持清零。
0：使能时钟延展
1：禁止时钟延展
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 FMPI2C 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 FMPI2C 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 FMPI2C 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。*

26.7.2 控制寄存器 2 (FMPI2C_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 FMPI2CCLK。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	PEC BYTE	AUTO END	RE LOAD	NBYTES[7:0]							
					rs	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NACK	STOP	START	HEAD 10R	ADD10	RD_ WRN	SADD[9:0]									
rs	rs	rs	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:27 保留，必须保持复位值。

位 26 **PECBYTE**：数据包错误校验字节 (Packet error checking byte)

此位由软件置 1，并可在 PEC 传输完成时、接收到停止条件或匹配地址时、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 26.3 节：FMPI2C 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 的低电平时间将延展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 的低电平时间将延展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、接收到停止条件或匹配地址时、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：在主机接收模式下，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。

当从机接收 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 FMPI2C_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 FMPI2C 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 FMPI2C 处于从模式，也可将 START 位置 1。

当 RELOAD 置 1 时，该位不起作用。在 10 位寻址模式下，如果在地址的第一部分接收到 NACK，则 START 位不会由硬件清零，主器件将重新发送地址序列（除非 START 位被软件清零）

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（主机接收模式）(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]**：从地址位 9:8（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。

26.7.3 设备自身地址 1 寄存器 (FMPI2C_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 FMPI2CCLK。

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

位 31:16 保留，必须保持复位值。

位 15 **OA1EN**：设备自身地址 1 使能 (Own Address 1 enable)
0：禁止设备自身地址 1。不对接收的从地址 OA1 应答。
1：使能设备自身地址 1。对接收的从地址 OA1 应答。

位 14:11 保留，必须保持复位值。

位 10 **OA1MODE**：设备自身地址 1 10 位模式 (Own Address 1 10-bit mode)
0：设备自身地址 1 为 7 位地址。
1：设备自身地址 1 为 10 位地址。
注：　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)
7 位寻址模式：7 位地址
10 位寻址模式：10 位地址的位 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。

26.7.4 设备自身地址 2 寄存器 (FMPI2C_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 FMPI2CCLK。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA2EN	Res.	Res.	Res.	Res.	OA2MSK[2:0]			OA2[7:1]							Res.
rw					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

位 31:16 保留，必须保持复位值。

位 15 **OA2EN**：设备自身地址 2 使能 (Own Address 2 enable)

0：禁止设备自身地址 2。不对接收的从地址 OA2 应答。

1：使能设备自身地址 2。对接收的从地址 OA2 应答。

位 14:11 保留，必须保持复位值。

位 10:8 **OA2MSK[2:0]**：设备自身地址 2 屏蔽位 (Own Address 2 masks)

000：无屏蔽

001：OA2[1] 被屏蔽，为无关位。仅比较 OA2[7:2]。

010：OA2[2:1] 被屏蔽，为无关位。仅比较 OA2[7:3]。

011：OA2[3:1] 被屏蔽，为无关位。仅比较 OA2[7:4]。

100：OA2[4:1] 被屏蔽，为无关位。仅比较 OA2[7:5]。

101：OA2[5:1] 被屏蔽，为无关位。仅比较 OA2[7:6]。

110：OA2[6:1] 被屏蔽，为无关位。仅比较 OA2[7]。

111：OA2[7:1] 被屏蔽，为无关位。不进行比较，对接收到的全部 7 位地址（保留位除外）应答。

注： *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 FMPI2C 地址（0b0000xxx 和 0b1111xxx）应答。

位 7:1 **OA2[7:1]**：接口地址 (Interface address)

7 位寻址模式：7 位地址

注： *仅可在 OA2EN=0 时写入这些位。*

位 0 保留，必须保持复位值。

26.7.5 时序寄存器 (FMPI2C_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	rw					rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCLH[7:0]								SCLL[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:28 **PRESC[3:0]**：时序预分频因子 (Timing prescaler)

该字段用于对 FMPI2CCLK 进行预分频，以生成用于数据建立和保持计数器（请参见*第 741 页的 FMPI2C 时序*）以及 SCL 高电平和低电平计数器（请参见*第 755 页的 FMPI2C 主模式初始化*）的时钟周期 t_{PRESC}。

$t_{PRESC} = (PRESC+1) \times t_{I2CCLK}$

位 27:24 保留，必须保持复位值。

位 23:20 **SCLDEL[3:0]**：数据建立时间 (Data setup time)

该字段用于在 SDA 边沿和 SCL 上升沿之间生成延时 t_{SCLDEL}。在主模式和从模式下，如果 NOSTRETCH = 0，则 SCL 线的低电平时间将在 t_{SCLDEL} 期间延展。

$t_{SCLDEL} = (SCLDEL+1) \times t_{PRESC}$

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)

该字段用于在 SCL 下降沿和 SDA 边沿之间生成延时 t_{SDADEL}。在主模式和从模式下，如果 NOSTRETCH = 0，则 SCL 线的低电平时间将在 t_{SDADEL} 期间延展。

$t_{SDADEL} = SDADEL \times t_{PRESC}$

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 FMPI2C 禁止时 (PE = 0) 进行配置。

注： STM32CubeMX 工具计算 I2C_TIMINGR 内容并在 I2C 配置窗口中进行显示。

26.7.6 超时寄存器 (FMPI2C_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 FMPI2CCLK。

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

位 31 **TEXTEN**：时钟信号延展超时使能 (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 FMPI2C 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 26.3 节：FMPI2C 特性实现。

26.7.7 中断和状态寄存器 (FMPI2C_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	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUSY	Res.	ALERT	TIME OUT	PEC ERR	OVR	ARLO	BERR	TCR	TC	STOPF	NACKF	ADDR	RXNE	TXIS	TXE
r		r	r	r	r	r	r	r	r	r	r	r	r	rs	rs

位 31:24 保留，必须保持复位值。

位 23:17 **ADDCODE[6:0]**：地址匹配代码（从模式）(Address match code (Slave mode))
发生地址匹配事件时 (ADDR = 1)，这些位更新为接收到的地址。
在 10 位地址的情况下，ADDCODE 提供 10 位地址的头字节，后跟地址的 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 26.3 节：FMPI2C 特性实现。

位 12 **TIMEOUT**：超时或 t_{LOW} 检测标志 (Timeout or t_{LOW} detection flag)
发生超时或延展时钟超时时，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 26.3 节：FMPI2C 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 26.3 节：FMPI2C 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 1 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 1 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 FMPI2C_RXDR 寄存器且准备就绪可供读取时，该位由硬件置 1。读取 FMPI2C_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 FMPI2C_TXDR 寄存器为空时，该位由硬件置 1，待发送的数据必须写入 FMPI2C_TXDR 寄存器。下一个待发送的数据写入 FMPI2C_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 FMPI2C_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 FMPI2C_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 FMPI2C_TXDR。
注： 当 PE=0 时，该位由硬件置 1。

26.7.8 中断清零寄存器 (FMPI2C_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，FMPI2C_ISR 寄存器中 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 26.3 节：FMPI2C 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，FMPI2C_ISR 寄存器中 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 26.3 节：FMPI2C 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，FMPI2C_ISR 寄存器中 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 26.3 节：FMPI2C 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，FMPI2C_ISR 寄存器中 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，FMPI2C_ISR 寄存器中 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，FMPI2C_ISR 寄存器中 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，FMPI2C_ISR 寄存器中 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，FMPI2C_ISR 寄存器中的 NACKF 标志将清零。

位 3 **ADDRCF**：地址匹配标志清零 (Address Matched flag clear)
将 1 写入此位时，FMPI2C_ISR 寄存器中 ADDR 标志将清零。将 1 写入此位时，FMPI2C_CR2 寄存器中的 START 位也将清零。

位 2:0 保留，必须保持复位值。

26.7.9 PEC 寄存器 (FMPI2C_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	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **PEC[7:0]**：数据包错误校验寄存器 (Packet error checking register)
当 PECEN=1 时，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 26.3 节：FMPI2C 特性实现。*

26.7.10 接收数据寄存器 (FMPI2C_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	r	r	r	r	r	r	r

位 31:8 保留，必须保持复位值。

位 7:0 **RXDATA[7:0]**：8 位接收数据 (8-bit receive data)
从 I^2C 总线接收的数据字节。

26.7.11 发送数据寄存器 (FMPI2C_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	rw	rw	rw	rw	rw	rw	rw

位 31:8 保留，必须保持复位值。

位 7:0 **TXDATA[7:0]**：8 位发送数据 (8-bit transmit data)
待发送到 I^2C 总线的数据字节。
注： 仅可在 TXE=1 时写入这些位。

26.7.12 FMPI2C 寄存器映射

下表提供了 FMPI2C 寄存器映射和复位值。

表 152. FMPI2C 寄存器映射和复位值

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0	**FMPI2C_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	**FMPI2C_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	**FMPI2C_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	**FMPI2C_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	**FMPI2C_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	**FMPI2C_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	**FMPI2C_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	**FMPI2C_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	**FMPI2C_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	**FMPI2C_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 152. FMPI2C 寄存器映射和复位值（续）

偏移	寄存器名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x28	**FMPI2C_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 54 页上的第 2.2.2 节*。

27 内部集成电路 (I²C) 接口

27.1 I²C 简介

I²C（内部集成电路）总线接口用作微控制器和 I²C 串行总线之间的接口。它提供多主模式功能，可以控制所有 I²C 总线特定的序列、协议、仲裁和时序。它支持标准模式（Sm，最高 100 kHz）和快速模式（Fm，最高 400 kHz）。

它可以用于多种用途，包括 CRC 生成和验证、SMBus（系统管理总线）以及 PMBus（电源管理总线）。

根据器件的不同，可利用 DMA 功能来减轻 CPU 的工作量。

27.2 I²C 主要特性

- 并行总线/I²C 协议转换器
- 多主控能力：同一接口可作为主设备或从设备工作
- I²C 主模式特性：
  - 生成时钟
  - 起始位和停止位生成
- I²C 从模式特性：
  - 可编程的 I²C 地址检测
  - 双寻址模式，可对 2 个从地址应答
  - 停止位检测
- 7 位/10 位寻址以及广播呼叫的生成和检测
- 支持多种通信速度：
  - 标准速度（高达 100 kHz）
  - 快速速度（高达 400 kHz）
- 模拟噪声滤波器
- 可编程数字噪声滤波器
- 状态标志：
  - 发送/接收模式标志
  - 字节传输结束标志
  - I²C 忙标志
- 错误标志：
  - 主模式仲裁丢失状态
  - 地址/数据传输后，回应失败
  - 检测到错误的开始或停止条件
  - 禁止时钟延长后出现的上溢/下溢
- 2 个中断向量：
  - 一个中断由成功的地址/数据字节传输事件触发
  - 一个中断由错误状态触发
- 可选的时钟延长
- 带 DMA 功能的 1 字节缓冲

- 可配置的 PEC（数据包错误校验）生成或验证：
  - 在 Tx 模式下，可将 PEC 值作为最后一个字节进行传送
  - 针对最后接收字节的 PEC 错误校验
- SMBus 2.0 兼容性：
  - 25 ms 时钟低电平超时延迟
  - 10 ms 主器件累计时钟低电平延长时间
  - 25 ms 从器件累计时钟低电平延长时间
  - 具有 ACK 控制的硬件 PEC 生成/验证
  - 支持地址解析协议 (ARP)
- PMBus 兼容性

注： *上述某些特性可能不适用于某些产品。用户应参照产品数据手册来确定 I²C 接口实现所支持的特定特性。*

27.3 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 I²C 总线。它可以连接到标准（高达 100 kHz）或快速（高达 400 kHz）I²C 总线。

27.3.1 模式选择

该接口在工作时可选用以下四种模式之一：

- 从机发送
- 从机接收
- 主机发送
- 主机接收

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

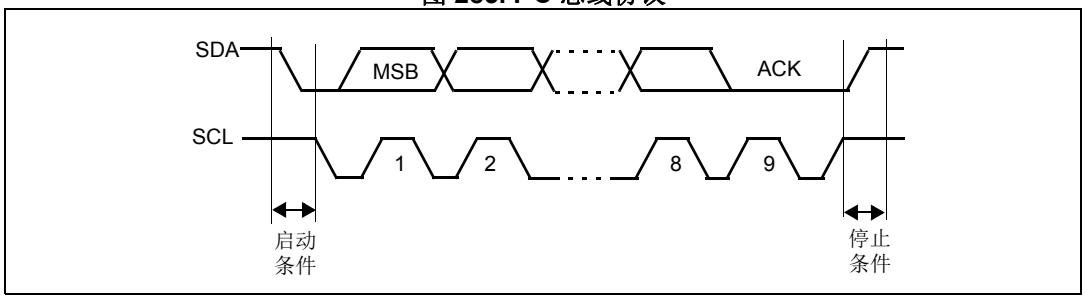
在主模式下，I²C 接口会启动数据传输并生成时钟信号。串行数据传输始终是在出现起始位时开始，在出现停止位时结束。起始位和停止位均在主模式下由软件生成。

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。

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*图 285*。

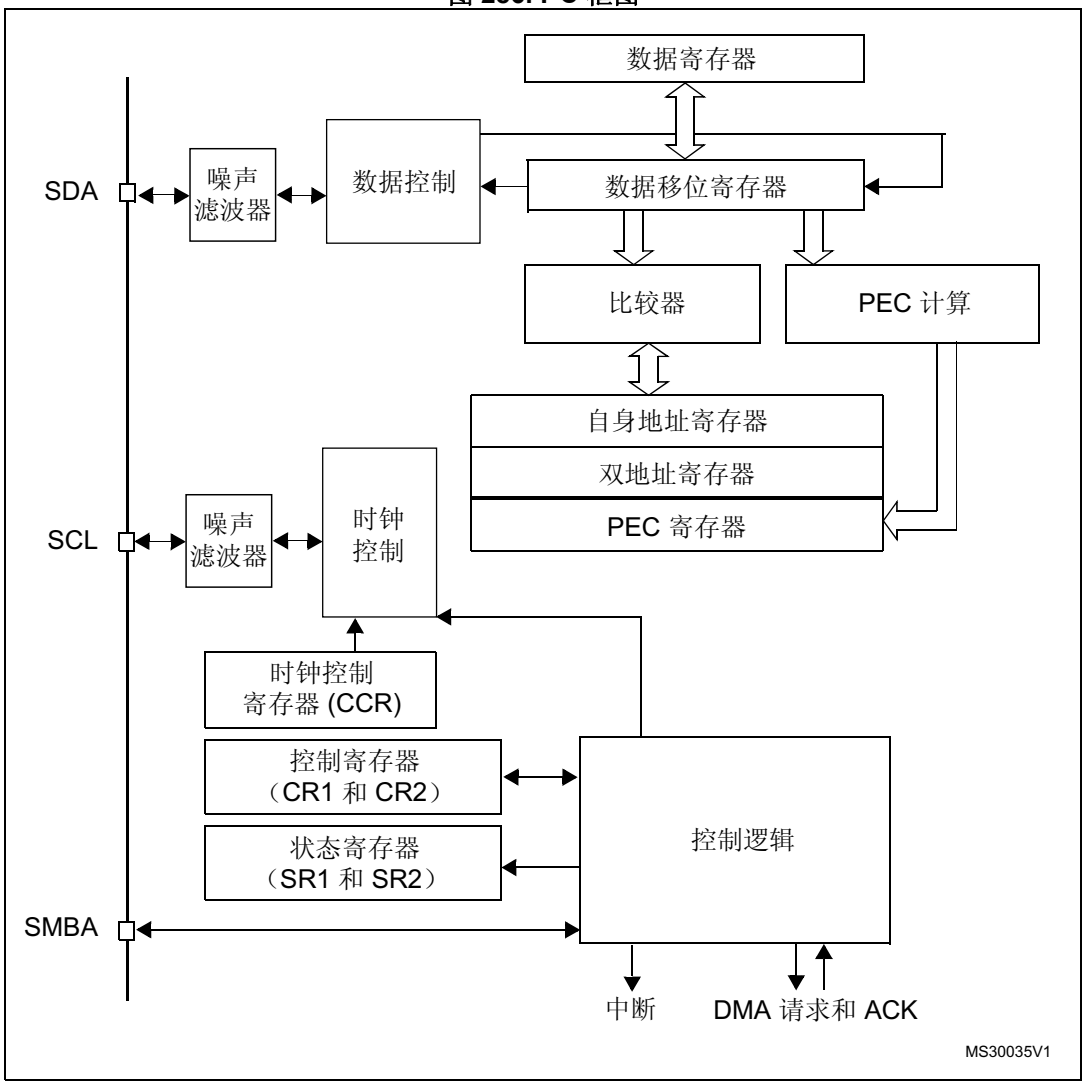
图 285. I²C 总线协议



应答位可由软件使能或禁止。I²C 接口地址（7 位/10 位双寻址模式和/或广播呼叫地址）可通过软件进行选择。

I²C 接口的框图如*图 286* 所示。

图 286. I²C 框图



1. SMBA 是 SMBus 模式下的一个可选信号。如果 SMBus 已禁止，则此信号不适用。

27.3.2 I²C 从模式

默认情况下，I²C 接口在从模式下工作。要将工作模式由默认的从模式切换为主模式，需要生成一个起始位。

为了生成正确的时序，必须在 I2C_CR2 寄存器中对外设输入时钟进行编程。外设输入时钟频率的下限为：

- Sm 模式下 2 MHz
- Fm 模式下 4 MHz

检测到起始位后，便会立即接收到来自 SDA 线的地址并将其送到移位寄存器。之后，会将其与接口地址 (OAR1) 和 OAR2（如果 ENDUAL=1）或者广播呼叫地址（如果 ENGC = 1）进行比较。

注： *在 10 位寻址模式下，比较对象还包括头序列 (11110xx0)，其中，xx 表示该地址的两个最高有效位。*

头或地址不匹配：接口会忽略它并等待下一个起始位。

头匹配（仅针对 10 位模式）：如果 ACK 位置 1，则接口会生成一个应答脉冲并等待 8 位从地址。

地址匹配：接口会依次：

- 发出应答脉冲（如果 ACK 位置 1）。
- ADDR 位会由硬件置 1 并在 ITEVFEN 位置 1 时生成一个中断。
- 如果 ENDUAL=1，则软件必须读取 DUALF 位状态来核对哪些从地址进行了应答。

在 10 位模式下，完成地址序列接收后，从模式始终处于接收模式。在接收到重复起始位以及一个匹配地址位和最低有效位均置 1 的头序列 (11110xx1) 后，它会进入发送模式。

TRA 位指示从设备是处于接收模式还是处于发送模式。

从机发送

在接收到地址并将 ADDR 清零后，从设备会通过内部移位寄存器将 DR 寄存器中的字节发送到 SDA 线。

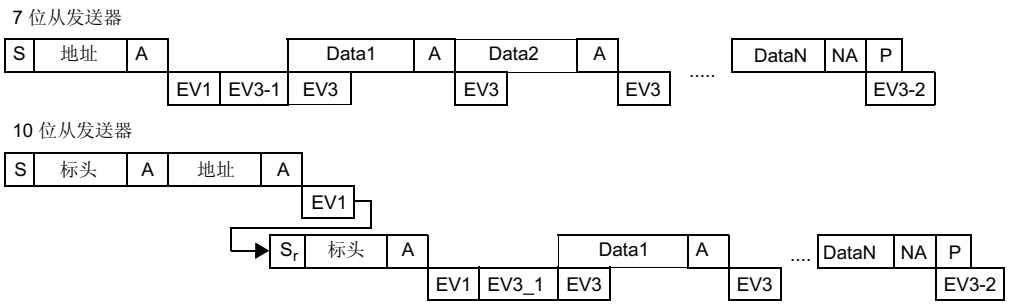
从设备会延长 SCL 低电平时间，直到 ADDR 位清零且 DR 寄存器中填满待发送数据为止（请参见*图 287* 传输序列 EV1 EV3）。

接收到应答脉冲时：

- TxE 位会由硬件置 1 并在 ITEVFEN 和 ITBUFEN 位均置 1 时生成一个中断。

如果在下一次数据传输结束之前 TxE 位已置 1 但某些数据尚未写入 I2C_DR 寄存器，则 BTF 位会置 1，而接口会一直延长 SCL 低电平，直到通过软件对 I2C_SR1 读操作，以及对 I2C_DR 写操作后，把 BTF 清零为止。

图 287. 从机发送的传输序列图



图例：S = 起始，S_r = 重复起始，P = 停止，A = 确认，NA = 非确认，EVx = 事件（ITEVFEN = 1 时具有中断）

EV1：ADDR=1，通过读取 SR1 后读取 SR2 来清零。
EV3-1：TxE=1，移位寄存器为空，数据寄存器为空，在 DR 中写入 Data1。
EV3-1：TxE=1，移位寄存器非空，数据寄存器为空，通过写入 DR 清零。
EV3-2：AF=1，AF 通过向 SR1 寄存器的 AF 位写入“0”来清零。

ai18209V2

1. EV1 和 EV3_1 事件可延长 SCL 低电平时间，直到相应的软件序列结束为止。
2. 如果软件序列在下一个字节传输结束之前未能完成，EV3 事件会延长 SCL 低电平时间。

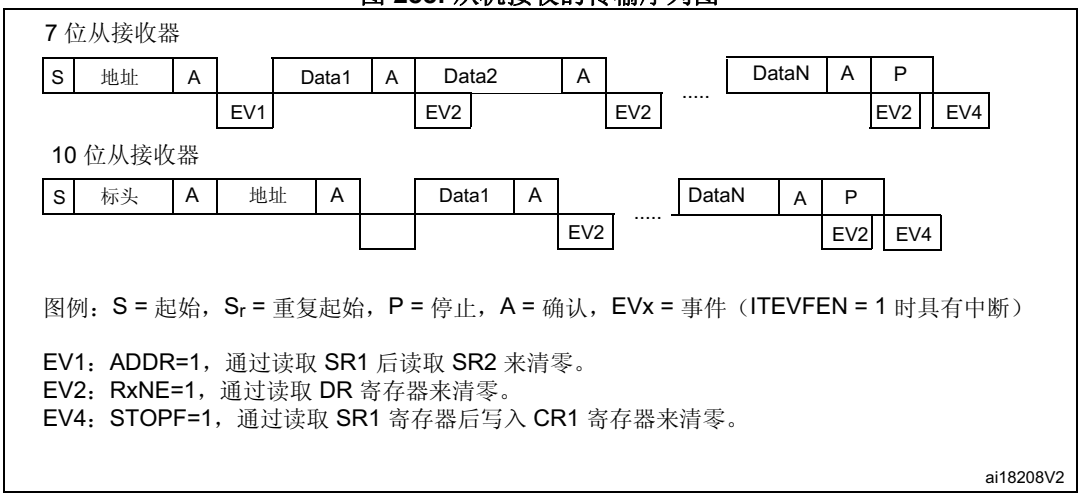
从机接收

在接收到地址并将 ADDR 位清零后，从设备会通过内部移位寄存器接收 SDA 线中的字节并将其保存到 DR 寄存器。在每个字节传输结束后，接口都会依次：

- 发出应答脉冲（如果 ACK 位置 1）。
- RxNE 位会由硬件置 1 并在 ITEVFEN 和 ITBUFEN 位均置 1 时生成一个中断。

如果在下一次数据接收结束之前 RxNE 位已置 1 但 DR 寄存器中的数据尚未读取，则 BTF 位会置 1，而接口会一直延长 SCL 低电平，直到软件通过读取 I2C_DR 寄存器来把 BTF 清零（请参见*图 288* 传输序列）。

图 288. 从机接收的传输序列图



1. EV1 事件可延长 SCL 低电平时间，直到相应的软件序列结束为止。
2. 如果软件序列在下一个字节接收结束之前未能完成，EV2 事件会延长 SCL 低电平时间。
3. 检查了 SR1 寄存器内容之后，用户应针对各个被置 1 的标志执行完整的清零序列。
   对于 ADDR 和 STOPF 标志，需要在 I2C 中断程序中执行以下序列：
   对 SR1 执行读操作
   如果 (ADDR == 1) {对 SR1 执行读操作；对 SR2 执行读操作}
   如果 (STOPF == 1) {对 SR1 执行读操作；对 CR1 执行写操作}
   这样做的目的是为了确保 ADDR 和 STOPF 这两个标志都清零（如果二者均为被置 1）。

关闭从设备通信

传输完最后一个数据字节之后，主设备会生成一个停止位。接口会检测此条件并：

- 将 STOPF 位置 1 并在 ITEVFEN 位置 1 时生成一个中断。

通过先读取 SR1 寄存器然后写入 CR1 寄存器的方式将 STOPF 位清零（请参见*图 288：从机接收的传输序列图* EV4）。

27.3.3 I²C 主模式

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。只要通过 START 位在总线上生成了起始位，即会选中主模式。

在主模式中要求执行以下序列。

- 在 I2C_CR2 寄存器中对外设输入时钟进行编程，以生成正确的时序
- 配置时钟控制寄存器
- 配置上升时间寄存器
- 对 I2C_CR1 寄存器进行编程，以便使能外设
- 将 I2C_CR1 寄存器的 START 位置 1，以生成起始位

外设输入时钟频率的下限为：

- Sm 模式下 2 MHz
- Fm 模式下 4 MHz

SCL 主时钟生成

CCR 位用于生成 SCL 时钟的高电平和低电平（分别从生成上升沿和下降沿开始）。由于从器件可能会延长 SCL 线，因此生成上升沿后，外设会在 TRISE 位中编程的时间结束时从总线检查 SCL 输入。

- 如果 SCL 线为低电平，则表示从器件正在延长总线，此时高电平计数器会停止计数，直到检测到 SCL 线为高电平。这样可保证 SCL 时钟参数的高电平周期为最小值。
- 如果 SCL 线为高电平，则高电平计数器继续计数。

实际上，即使没有从器件延长时钟，从外设生成 SCL 上升沿到外设检测到 SCL 上升沿这一反馈回路也需要一定的时间。此回送过程的持续时间与 SCL 上升时间相关（影响 SCL VIH 输入检测），外加由于 SCL 输入路径上的噪声滤波器引起的延迟以及由于内部 SCL 输入与 APB 时钟同步引起的延迟。在 TRISE 位中编程可供反馈回路使用的最长时间，从而确保无论 SCL 上升时间是多少，SCL 频率都保持稳定。

启动条件

START 位置 1 后，接口会在 BUSY 位清零后生成一个起始位并切换到主模式（MSL 位置 1）。

注： *在主设备下，START 位置 1 后，接口会在当前字节传输结束后生成一个重复起始位。*

起始位发出之后：

- SB 位会由硬件置 1 并在 ITEVFEN 位置 1 时生成一个中断。

接下来主设备会等待软件对 SR1 执行读操作，然后把从设备地址写入 DR 寄存器（请参见 *图 289* 和 *图 290* 传输序列 EV5）。

从地址传输

接下来从地址会通过内部移位寄存器发送到 SDA 线。

- 在 10 位寻址模式中，发送头序列会产生以下事件：
  - ADD10 位会由硬件置 1 并在 ITEVFEN 位置 1 时生成一个中断。

    接下来主设备会等待软件读取 SR1，然后把第二个地址字节写入 DR 寄存器（请参见 *图 289* 和 *图 290* 传输序列）。
  - ADDR 位会由硬件置 1 并在 ITEVFEN 位置 1 时生成一个中断。

    接下来主设备会等待对 SR1 寄存器执行读操作，然后对 SR2 寄存器执行读操作（请参见 *图 289* 和 *图 290* 传输序列）。
- 在 7 位寻址模式下，会发送一个地址字节。

  地址字节被发出后，
  - ADDR 位会由硬件置 1 并在 ITEVFEN 位置 1 时生成一个中断。

    接下来主设备会等待对 SR1 寄存器执行读操作，然后对 SR2 寄存器执行读操作（请参见 *图 289* 和 *图 290* 传输序列）。

主设备会根据发送的从地址字节 LSB 来决定是进入发送模式还是接收模式。

- 在 7 位寻址模式下，
  - 要进入发送模式，主设备会发送从地址并将 LSB 复位。
  - 要进入接收模式，主设备会发送从地址并将 LSB 置 1。
- 在 10 位寻址模式下，
  - 要进入发送模式，主设备会先发送头序列 (11110xx0)，然后发送从地址（其中 xx 表示该地址的两个最高有效位）。
  - 要进入接收模式，主设备会先发送头序列 (11110xx0)，然后发送从地址。接下来会发送一个重复起始位，然后再发送头序列 (11110xx1)（其中 xx 表示地址的两个最高有效位）。

  TRA 位指示主设备是处于接收模式还是处于发送模式。

主机发送

在发送出地址并将 ADDR 清零后，主设备会通过内部移位寄存器将 DR 寄存器中的字节发送到 SDA 线。

主设备会一直等待，直到首个数据字节被写入 I2C_DR 为止（请参见*图 289* 传输序列 EV8_1）。

接收到应答脉冲后，TxE 位会由硬件置 1 并在 ITEVFEN 和 ITBUFEN 位均置 1 时生成一个中断。

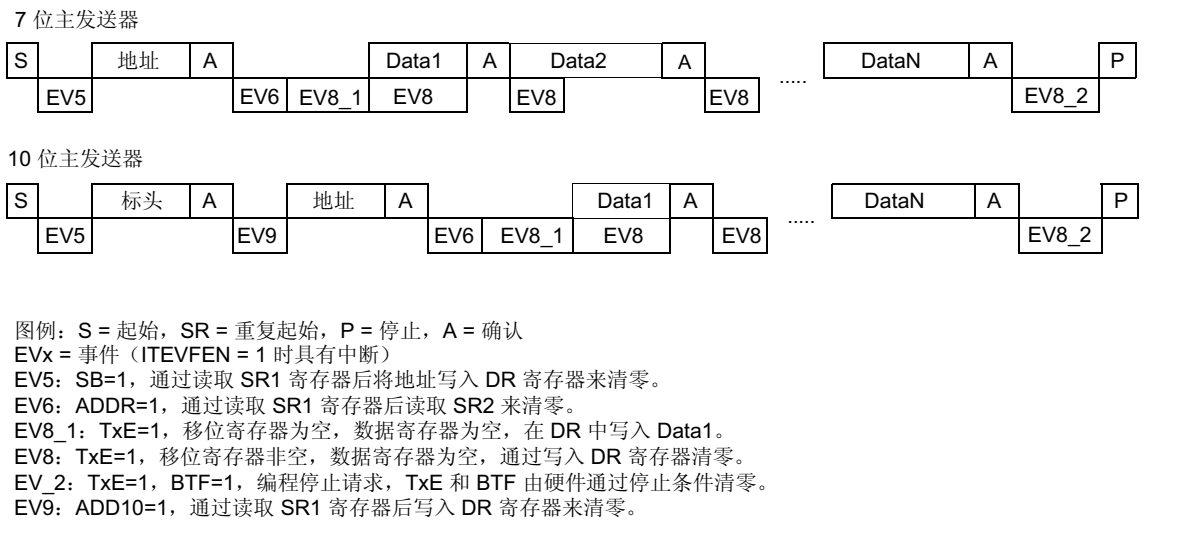
如果在上一次数据传输结束之前 TxE 位已置 1 但数据字节尚未写入 DR 寄存器，则 BTF 位会置 1，而接口会一直延长 SCL 低电平，等待 I2C_DR 寄存器被写入，以将 BTF 清零。

结束通信

当最后一个字节写入 DR 寄存器后，软件会将 STOP 位置 1 以生成一个停止位（请参见*图 289* 传输序列 EV8_2）。接口会自动返回从模式（MSL 位清零）。

注： *当 TxE 或 BTF 中的任何一个置 1 时，应在 EV8_2 事件期间对停止位进行编程。*

图 289. 主机发送的传输序列图



图例：S = 起始，SR = 重复起始，P = 停止，A = 确认
EVx = 事件（ITEVFEN = 1 时具有中断）
EV5：SB=1，通过读取 SR1 寄存器后将地址写入 DR 寄存器来清零。
EV6：ADDR=1，通过读取 SR1 寄存器后读取 SR2 来清零。
EV8_1：TxE=1，移位寄存器为空，数据寄存器为空，在 DR 中写入 Data1。
EV8：TxE=1，移位寄存器非空，数据寄存器为空，通过写入 DR 寄存器清零。
EV_2：TxE=1，BTF=1，编程停止请求，TxE 和 BTF 由硬件通过停止条件清零。
EV9：ADD10=1，通过读取 SR1 寄存器后写入 DR 寄存器来清零。

ai18210V2

1. EV5、EV6、EV9、EV8_1 和 EV8_2 事件可延长 SCL 低电平时间，直到相应的软件序列结束为止。
2. 如果软件序列在下一个字节传输结束之前未能完成，EV8 事件会延长 SCL 低电平时间。

主机接收

完成地址传输并将 ADDR 位清零后，I²C 接口会进入主接收模式。在此模式下，接口会通过内部移位寄存器接收 SDA 线中的字节并将其保存到 DR 寄存器。 在每个字节传输结束后，接口都会依次：

1. 发出应答脉冲（如果 ACK 位置 1）。
2. RxNE 位置 1 并在 ITEVFEN 和 ITBUFEN 位均置 1 时生成一个中断（参见*图 290* 传输序列 EV7）。

如果在上一次数据接收结束之前 RxNE 位已置 1 但 DR 寄存器中的数据尚未读取，则 BTF 位会由硬件置 1，而接口会一直延长 SCL 低电平，等待 I2C_DR 寄存器被写入，以将 BTF 清零。

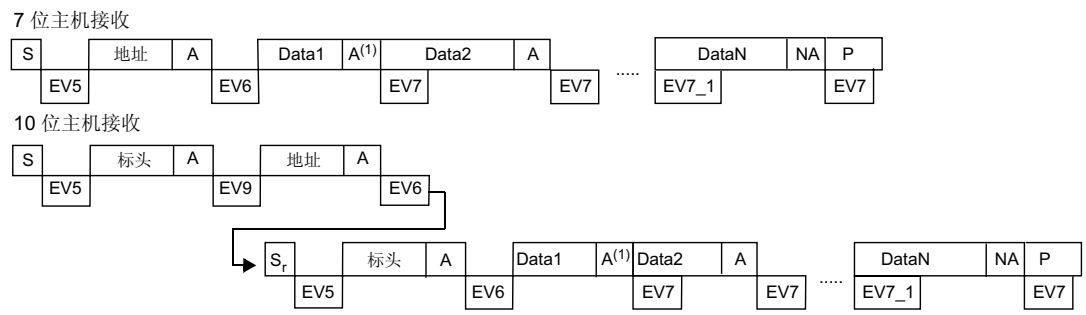
结束通信

主设备会针对自从设备接收的最后一个字节发送 NACK。在接收到此 NACK 之后，从设备会释放对 SCL 和 SDA 线的控制。随后，主设备可发送一个停止位/重复起始位。

1. 为了在最后一个接收数据字节后生成非应答脉冲，必须在读取倒数第二个数据字节后（倒数第二个 RxNE 事件之后）立即将 ACK 位清零。
2. 要生成停止位/重复起始位，软件必须在读取倒数第二个数据字节后（倒数第二个 RxNE 事件之后）将 STOP/START 位置 1。
3. 在只接收单个字节的情况下，会在 EV6 期间（在 ADDR 标志清零之前）禁止应答并在 EV6 之后生成停止位。

生成停止位后，接口会自动返回从模式（MSL 位清零）。

图 290. 主机接收的传输序列图



图例：S=起始，S_r=重复起始，P=停止，A=确认，NA=非确认，EVX=事件（ITEVFEN=1 时具有中断）
EV5：SB=1，通过读取 SR1 寄存器后写入 DR 寄存器来清零。
EV6：ADDR = 1，通过读取 SR1 寄存器后读取 SR2 来清零。在 10 位主机接收模式下，应通过写入 CR2 (START=1) 来遵循此顺序。
接收 1 个字节时，会在 EV6 事件期间（即，ADDR 标志清零之前）禁止确认。
EV7_1：RxNE=1，通过读取 DR 寄存器清零。
EV7：RxNE=1，通过读取 DR 寄存器清零，编程 ACK=0 和 STOP 请求。
EV9：ADD10=1，通过读取 SR1 寄存器后写入 DR 寄存器来清零。

ai17540d

1. 如果接收单个字节，则为 NA。
2. EV5、EV6 和 EV9 事件可延长 SCL 低电平时间，直到相应的软件序列结束为止。
3. 如果软件序列在下一个字节接收结束之前未能完成，EV7 事件会延长 SCL 低电平时间。
4. EV7_1 软件序列必须在当前字节传输的 ACK 脉冲之前结束。

如果 EV7_1 软件序列未能在当前字节传输的 ACK 脉冲之前结束，建议采用下列步骤。

必须遵循以下步骤以确保：

- 在结束上一个数据接收之前及时将 ACK 位复位。
- 在上一次数据接收后将 STOP 位置 1 且不接收补充数据。

针对 2 字节的接收：

- 等待 ADDR = 1（SCL 保持低电平，直到 ADDR 标志清零）
- 将 ACK 复位，POS 置 1
- 将 ADDR 标志清零
- 等待 BTF = 1（数据 1在 DR 中，数据 2 在移位寄存器中，SCL 保持低电平，直到读取了数据 1 为止）
- 将 STOP 置 1
- 读取数据 1 和数据 2

针对 N > 2 的字节接收，从 N-2 数据接收开始

- 等待 BTF = 1（数据 N-2 在 DR 中，数据 N-1 在移位寄存器中，SCL 保持低电平，直到读取了数据 N-2 为止）
- 将 ACK 复位
- 读取数据 N-2
- 等待 BTF = 1（数据 N-1 在 DR 中，数据 N 在移位寄存器中，SCL 保持低电平，直到读取了数据 N-1 为止）
- 将 STOP 置 1
- 读取数据 N-1 和数据 N

27.3.4 错误条件

以下错误条件可能导致通信失败。

总线错误 (BERR)

当 I²C 接口在传输地址或数据期间检测到外部停止位或起始位时，会出现此错误。在这种情况下：

- BERR 位置 1 并在 ITERREN 位置 1 时生成一个中断
- 在从模式下：硬件会丢弃数据并释放数据线：
  - 在误放起始位的情况下，从模式会认为它是重复起始位并会等待一个地址或停止位
  - 在误放停止位的情况下，从模式会按停止位操作且硬件会释放数据线
- 在主模式下：不会释放数据线且不会影响当前传输状态。由软件决定是否中止当前传输

应答失败 (AF)

当接口检测到未应答脉冲会出现此错误。在这种情况下：

- AF 位置 1 并在 ITERREN 位置 1 时生成一个中断
- 接收到 NACK 的发送器必须复位通信：
  - 如果在从模式下：硬件释放数据线
  - 如果在主模式下：必须由软件生成一个停止位或重复起始位

仲裁丢失 (ARLO)

当 I²C 接口检测到仲裁丢失时会出现此错误。在这种情况下，

- ARLO 位会由硬件置 1（并在 ITERREN 位置 1 时生成一个中断）
- I²C 接口会自动返回从模式（MSL 位清零）。I²C 失去仲裁时，将无法在本次传输中对自身从地址进行应答，但是能够在赢得仲裁的主设备发出重复起始位后对自身设备进行应答。
- 硬件释放数据线

上溢/下溢错误 (OVR)

当时钟延长已禁止且 I²C 接口正在接收数据时，从模式中可能出现上溢错误。接口已经收到一个字节 (RxNE=1)，但是收到下一个字节之前 DR 中的数据未被读走。在这种情况下，

- 会丢失接收的最后一个字节。
- 在出现上溢错误时，软件应将 RxNE 位清零，而发送器应重新发送最后一个字节。

当时钟延长已禁止且 I²C 接口正在发送数据时，从模式中可能出现下溢错误。下一个字节的时钟信号到来之前，从设备还未把下一个要发送的数据写进 DR (TxE=1)。在这种情况下，

- 会再次发送 DR 寄存器中的字节。
- 用户应确保在出现下溢错误期间从机接收端接收到的数据被丢弃，并确保在 I²C 总线标准指定的时钟低电平时间内写入下一个字节。

对于要传送的首个字节，必须在 ADDR 清零之后且出现首个 SCL 上升沿之前将其写入 DR 寄存器。否则，接收器必须丢弃首个数据。

27.3.5 可编程噪声滤波器

在 Fm 模式下，I²C 标准要求将 SDA 和 SCL 线上尖峰脉宽在 50 ns 以内的噪声都抑止掉。

SDA 和 SCL I/O 中采用了模拟噪声滤波器。此滤波器默认为使能，通过将 I2C_FLTR 寄存器中的 ANOFF 位置 1 可禁止它。

将 DNF[3:0] 位配置为一个非零值可使能数字噪声滤波器。这可以抑制 SDA 和 SCL 输入上脉宽小于 DNF[3:0] * T_{PCLK1} 的噪声。

使能数字噪声滤波器后，SDA 保持时间可增加 (DNF[3:0] +1)* T_{PCLK}。

为了符合 I²C 总线规范版本 2.1 (Thd:dat) 中对最大保持时间的要求，必须使用*表 153* 所示的限制条件对 DNF 位进行编程并假定模拟滤波器已禁止。

注： *DNF[3:0] 必须在 I²C 已禁止 (PE = 0) 的情况下配置。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。*

表 153. 符合 Thd:dat(max) 的 DNF[3:0] 最大值

PCLK1 频率	DNF 最大值	
	Sm 模式	Fm 模式
2 <= F_{PCLK1} <= 5	2	0
5 < F_{PCLK1} <= 10	12	0
10 < F_{PCLK1} <= 20	15	1
20 < F_{PCLK1} <= 30	15	7
30 < F_{PCLK1} <= 40	15	13
40 < F_{PCLK1} <= 50	15	15

注： *上表根据各个频率范围的最低频率给出了相应的限制条件。如果系统可以支持最大保持时间违例，则可使用更大的 DNF 值。*

27.3.6 SDA/SCL 线控制

- 如果时钟延长已使能：
  - 发送器模式：如果 TxE=1 且 BTF=1：接口会在发送数据之前保持时钟线为低电平，以等待微控制器将字节写入数据寄存器（缓冲寄存器和移位寄存器均为空）。
  - 接收器模式：如果 RxNE=1 且 BTF=1：接口会在接收数据之后保持时钟线为低电平，以等待微控制器将字节读入数据寄存器（缓冲寄存器和移位寄存器均已满）。
- 如果从模式中的时钟延长已禁止：
  - RxNE=1，且在下一个数据接收完成之前还未读走 DR 中的数据就出现上溢错误。会丢失接收的最后一个字节。
  - TxE=1，且在下一个数据的始终到来之前还未把发送数据写到 DR 中，就出现下溢错误。会再次发送同一字节。
  - 不会对写冲突进行管理。

27.3.7 SMBus

简介

系统管理总线 (SMBus) 是一个双线制接口，各器件可通过它在彼此之间或者与系统的其余部分进行通信。它以 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。系统可使用 SMBus 与设备进行消息传递，而无需切换各个控制线。

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。

SMBus 与 I²C 的相似之处

- 双线制总线协议（1 个时钟总线，1 个数据总线）+ 可选 SMBus 报警线。
- 主从通信，主器件提供时钟。
- 多主器件功能。
- SMBus 数据格式与 I²C 7 位地址格式相似（*图 285*）。

SMBus 与 I²C 之间的差异

下表介绍了 SMBus 与 I²C 之间的差异。

表 154. SMBus 与 I²C

SMBus	I²C
最大速度 100 kHz	最大速度 400 kHz
最小时钟速度 10 kHz	无最小时钟速度
35 ms 时钟低电平超时	无超时
逻辑电平固定	逻辑电平取决于 V_{DD}
地址类型不同（保留、动态等）	7 位、10 位和广播从模式地址类型
总线协议不同（快速命令、过程调用等）	无总线协议

SMBus 应用用途

通过系统管理总线，器件可以提供制造商信息、告诉系统它的型号/部件号、保存暂停事件的状态、报告不同的错误类型、接受控制参数并返回其状态。SMBus 可针对系统和电源管理相关的任务提供控制总线。

设备标识

系统管理总线中作为从器件的任何器件均具有一个唯一地址，被称为从地址。有关保留的从地址列表的信息，请参见 SMBus 规范版本 2.0 (http://smbus.org/)。

总线协议

SMBus 规范支持多达 9 类总线协议。有关这些协议和 SMBus 地址类型的详细信息，请参见 SMBus 规范版本 2.0.这些协议应通过用户软件实施。

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。地址解析协议 (ARP) 具有以下属性：

- 地址分配采用标准的 SMBus 物理层仲裁机制
- 器件通电时，分配的地址保持不变；器件断电时，也允许保留地址
- 完成地址分配后不会带来额外的 SMBus 数据包开销。（即，对已分配的从地址进行后续访问与访问固定地址的器件所产生的开销相同。）
- 任何 SMBus 主器件都可以遍历总线

唯一的器件标识符 (UDID)

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)。

有关 128 位 UDID 和 ARP 的详细信息，请参见 SMBus 规范版本 2.0。

SMBus 报警模式

SMBus 报警是带有中断线的可选信号，主要用于希望扩展它们的控制能力而牺牲一个引脚的器件。SMBA 是与 SCL 和 SDA 信号类似的线与信号。SMBA 与 SMBus 广播地址配合使用。使用 SMBus 调用的消息长度为 2 字节。

将 I2C_CR1 寄存器中的 ALERT 位置 1 后一个只具有从功能的器件可通过 SMBA 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*（简称 ARA，其值为 0001 100X）同步访问所有 SMBA 器件。只有那些将 SMBA 拉到低电平的器件会确认报警响应地址。通过 I2C_SR1 寄存器中的 SMBALERT 状态标志可确定这一状态。主机会执行修改后的接收字节操作。由从机发送件提供的 7 位器件地址被放置在字节的 7 个最高有效位。第 8 位可以是 0 或 1。

如果有不止一个器件将 SMBA 拉为低电平，则在从地址传输期间，具有最高优先级（最低位地址）的器件会通过标准仲裁获得通信权限。在确认从地址之后，器件必须释放 SMBA。如果消息传输结束后主机检测到 SMBA 仍为低电平，会再次读取 ARA。
未实现 SMBA 信号的主机会定期访问 ARA。

有关 SMBus 报警模式的详细信息，请参见 SMBus 规范版本 2.0 (http://smbus.org/)。

超时错误

SMBus 和 I²C 之间存在一些定时规范方面的差异。
SMBus 定义了一个时钟低电平超时，TIMEOUT 为 35 ms。另外，SMBus 指定 TLOW: SEXT 作为从器件的累积时钟低电平延长时间。SMBus 指定 TLOW: MEXT 作为主器件的累积时钟低电平延长时间。有关这些超时的详细信息，请参见 SMBus 规范版本 2.0。

I2C_SR1 寄存器中的 Timeout 或 Tlow 错误状态标志表明了此特性的状态。

如何在 SMBus 模式下使用该接口

要从 I²C 模式切换到 SMBus 模式，应执行以下步骤。

- 将 I2C_CR1 寄存器中的 SMBus 位置 1
- 根据应用的要求配置 I2C_CR1 寄存器中的 SMBTYPE 和 ENARP 位

如果要将某个器件配置为主器件，请按照*第 27.3.3 节：I2C 主模式*中介绍的起始位生成步骤进行操作。否则按照*第 27.3.2 节：I2C 从模式*中的顺序操作。

该应用需要通过软件控制各种 SMBus 协议。

- ENARP=1 且 SMBTYPE=0 时使用 SMB 器件默认地址
- ENARP=1 且 SMBTYPE=1 时使用 SMB 主机头字段
- SMBALERT=1 时使用 SMB 报警响应地址

27.3.8 DMA 请求

DMA 请求（在使能后）仅用于数据传输。当发送数据寄存器变空以及接收数据寄存器变满时会生成 DMA 请求。进行 I2C 数据传输之前，必须先初始化并使能 DMA。I2C_CR2 寄存器中的 DMAEN 位必须在 ADDR 事件之前置 1。在主模式或从模式下，如果已使能时钟延长，DMAEN 位也可以在 ADDR 事件期间于 ADDR 标志清零之前置 1。结束当前字节传输之前，必须发出 DMA 请求。当传输的数据量达到相应 DMA 数据流编程设定的值时，DMA 控制器会发送一个结束传输 EOT 信号给 I²C 接口，并生成一个传输完成中断（如果已使能）：

- 主机发送：在 EOT 中断后的中断程序中，禁止 DMA 请求，然后在等到 BTF 事件后设置停止位。
- 主机接收
  - 当要接收的字节数等于或大于二时，DMA 控制器会在收到倒数第二个数据字节（第 N - 1 个数据时）发送一个硬件信号 EOT_1。如果 I2C_CR2 寄存器中的 LAST 位置 1，I²C 会在 EOT_1 后的下一个字节之后自动发送一个 NACK。用户可在 DMA 传输完成中断（如果已使能）程序中生成停止位。
  - 当必须接收单个字节时：必须在 EV6 事件期间于 ADDR 标志清零之前对 NACK 进行编程，即当 ADDR=1 时编程设定 ACK=0。接下来，用户可在 ADDR 标志清零之后或者在执行 DMA 传输完成中断程序时编程设定停止位。

使用 DMA 进行发送

将 I2C_CR2 寄存器中的 DMAEN 位置 1 可以使能 DMA 模式进行发送。当 TXE 位置 1 时，数据将由 DMA 从预置的存储区装载进 I2C_DR 寄存器。要为 I²C 发送操作映射一个 DMA 数据流 x（其中 x 是数据流编号），需按顺序执行以下步骤：

1. 设置 DMA_SxPAR 寄存器中的 I2C_DR 寄存器地址。每次发生 TxE 事件后，数据都会从存储器移动到此地址。
2. 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。每次发生 TxE 事件后，数据都会从此存储器加载到 I2C_DR。
3. 在 MA_SxNDTR 寄存器中配置要传输的总字节数。在每次 TxE 事件后，此值都会递减。
4. 使用 DMA_SxCR 寄存器中的 PL[0:1] 位配置 DMA 数据流优先级。
5. 在完成半数传输或全部传输（取决于应用的需求）之后，将 DMA_SxCR 寄存器中的 DIR 位置 1 并配置中断。
6. 通过将 DMA_SxCR 寄存器中的 EN 位置 1 激活数据流。

当传输的数据量达到 DMA 控制器中编程设定的值时，DMA 控制器会发送一个结束传输 EOT/EOT_1 信号给 I²C 接口，而 DMA 会在 DMA 数据流中断向量上生成一个中断（如果已使能）。

注： *如果使用 DMA 进行传送，请勿使能 I2C_CR2 寄存器中的 ITBUFEN 位。*

使用 DMA 进行接收

将 I2C_CR2 寄存器中的 DMAEN 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 I2C_DR 寄存器加载到使用 DMA 外设配置的存储区中（参见 DMA 规范）。要为 I²C 接收操作映射一个 DMA 数据流 x（其中 x 是数据流编号），需按顺序执行以下步骤：

1. 设置 DMA_SxPAR 寄存器中的 I2C_DR 寄存器地址。每次发生 RxNE 事件后，数据都会从此地址移动到存储器。
2. 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。每次发生 RXNE 事件后，数据都会从 I2C_DR 寄存器加载到此存储区。
3. 在 DMA_SxNDTR 寄存器中配置要传输的总字节数。在每次 RxNE 事件后，此值都会递减。
4. 使用 DMA_SxCR 寄存器中的 PL[0:1] 位配置数据流优先级
5. 在完成半数传输或全部传输（取决于应用的需求）之后，将 DMA_SxCR 寄存器中的 DIR 位重新置 1 并配置中断。
6. 通过将 DMA_SxCR 寄存器中的 EN 位置 1 激活数据流。

当传输的数据量达到 DMA 控制器中编程设定的值时，DMA 控制器会发送一个结束传输 EOT/EOT_1 信号给 I²C 接口，而 DMA 会在 DMA 数据流中断向量上生成一个中断（如果已使能）。

注： *如果使用 DMA 进行接收，请勿使能 I2C_CR2 寄存器中的 ITBUFEN 位。*

27.3.9 数据包错误校验

PEC 计算器的应用目的是提高通信的可靠性。PEC 对各个位使用 CRC-8 多项式 $C(x) = x^8 + x^2 + x + 1$ 公式进行串行计算。

- 将 I2C_CR1 寄存器中的 ENPEC 位置 1 即可使能 PEC 计算。PEC 是针对所有消息字节（包括地址和 R/W 位）的 CRC-8 计算。
  - 在发送模式下：在与最后一个字节对应的 TxE 事件发生后，将 I2C_CR1 寄存器中的 PEC 传输位置 1。PEC 会在最后一个传输的字节之后进行传送。
  - 在接收模式下：在与最后一个字节对应的 RxNE 事件发生之后，将 I2C_CR1 寄存器中的 PEC 位置 1，以便接收器在接收到的下一个字节不等于内部计算的 PEC 时发送一个 NACK。在主机接收中，无论校验结果如何，PEC 后都将发送 NACK。在从模式下，必须在 CRC 接收的 ACK 之前设置 PEC。在主模式下，必须在复位 ACK 时设置 PEC 位。
- I2C_SR1 寄存器中还提供 PECERR 错误标志/中断。
- 如果 DMA 和 PEC 计算均已使能：
  - 在发送模式下：当 I²C 接口接收来自 DMA 控制器的 EOT 信号时，会在最后一个字节之后自动发送 PEC。
  - 在接收模式下：当 I²C 接口接收来自 DMA 控制器的 EOT_1 信号时，会自动将下一个字节视为 PEC 并对其进行校验。在 PEC 接收之后会生成一个 DMA 请求。
- 为了允许进行中间 PEC 传输，可使用 I2C_CR2 寄存器中的一个控制位（LAST 位）来确定它是否真的是最后一个 DMA 传输。如果确实是主机接收的最后一个 DMA 请求，则会在接收最后一个字节后自动发送一个 NACK。
- PEC 计算会因仲裁丢失而失效。

27.4 I²C 中断

下表给出了 I²C 中断请求列表。

表 155. I²C 中断请求

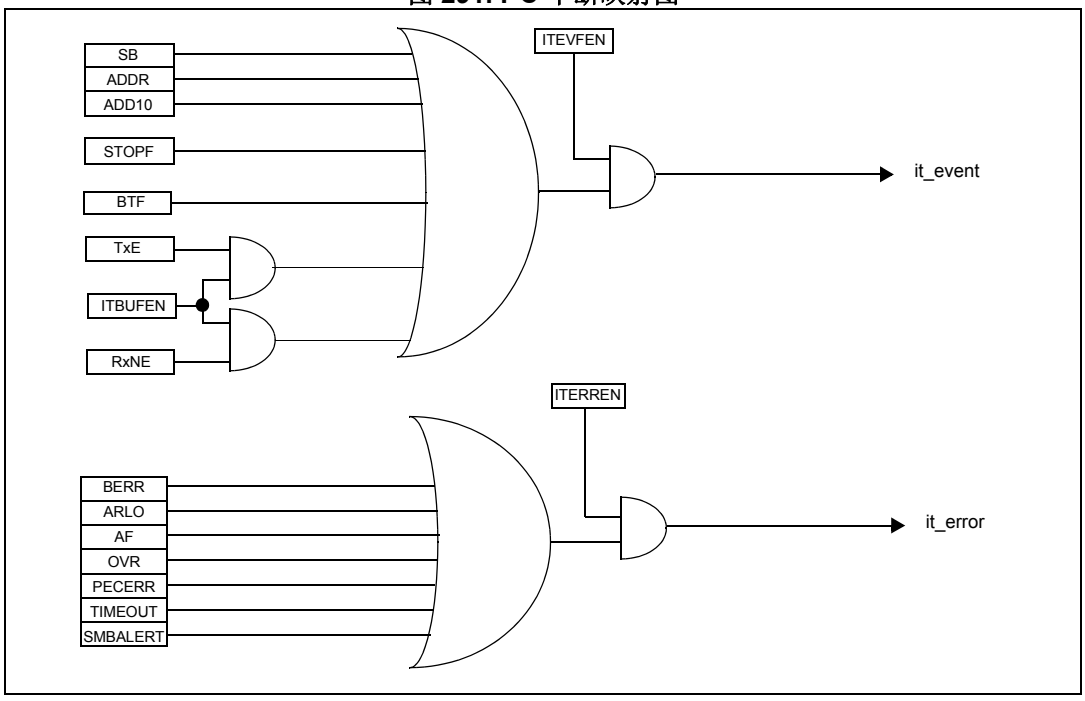
中断事件	事件标志	使能控制位
发送起始位（主模式）	SB	ITEVFEN
地址已发送（主模式）或地址匹配（从模式）	ADDR	
10 位地址的头段已发送（主模式）	ADD10	
已收到停止位（从模式）	STOPF	
完成数据字节传输	BTF	
接收缓冲区非空	RxNE	ITEVFEN 和 ITBUFEN
发送缓冲区为空	TxE	

表 155. I²C 中断请求（续）

中断事件	事件标志	使能控制位
总线错误	BERR	ITERREN
仲裁丢失（主模式）	ARLO	
应答失败	AF	
上溢/下溢	OVR	
PEC 错误	PECERR	
超时/Tlow 错误	TIMEOUT	
SMBus 报警	SMBALERT	

注： *SB、ADDR、ADD10、STOPF、BTF、RxNE 和 TxE 通过逻辑或映射到同一个中断通道上。*
BERR、ARLO、AF、OVR、PECERR、TIMEOUT 和 SMBALERT 通过逻辑或映射到同一个中断通道上。

图 291. I²C 中断映射图



27.5 I²C 调试模式

当微控制器进入调试模式时（带 FPU 的 Cortex®-M4 内核停止），SMBUS 超时会根据 DBG 模块中的 DBG_I2Cx_SMBUS_TIMEOUT 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 34.16.2 节：对定时器、看门狗、bxCAN 和 I2C 的调试支持*。

27.6 I²C 寄存器

有关寄存器说明中使用的缩写，请参见*第 50 页的第 1.2 节*。

必须按半字（16 位）或字（32 位）访问外设寄存器。

27.6.1 I²C 控制寄存器 1 (I2C_CR1)

I²C 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
SW RST	Res.	ALERT	PEC	POS	ACK	STOP	START	NO STRETCH	ENGC	ENPEC	ENARP	SMB TYPE	Res.	SM BUS	PE
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw

位 15 **SWRST**：软件复位 (Software reset)
当置 1 时，I2C 处于复位状态。在复位此位之前，确保 I2C 线已释放且总线空闲。
0：I²C 外设未处于复位状态
1：I²C 外设处于复位状态
注：　当出现错误或锁定状态后，可使用此位重新初始化外设。例如，如果 BUSY 位已置 1 但因母线干扰而不能复位，则可使用 SWRST 位退出此状态。

位 14 保留，必须保持复位值

位 13 **ALERT**：SMBus 报警 (SMBus alert)
此位由软件置 1 和清零，并可在 PE=0 时由硬件清零。
0：释放 SMBA 引脚使其变成高电平。报警响应地址头后跟 NACK。
1：驱动 SMBA 引脚使其变成低电平。报警响应地址头后跟 ACK。

位 12 **PEC**：数据包错误校验 (Packet error checking)
此位由软件置 1 和清零，并可在 PEC 传输完成时由硬件清零，或者在 PE=0 时或在检测到起始位或停止位时由硬件清零。
0：不传输 PEC
1：PEC 传输（在 Tx 或 Rx 模式下）
注：　PEC 计算会因仲裁丢失而失效。

位 11 **POS**：应答/PEC 位置 (Acknowledge/PEC Position)（针对接收数据）
此位由软件置 1 和清零，并可在 PE=0 时由硬件清零。
0：ACK 位控制移位寄存器中当前正在接收的字节的 (N)ACK。PEC 位指示移位寄存器中的当前字节是一个 PEC。
1：ACK 位控制移位寄存器中要接收的下一个字节的 (N)ACK。PEC 位指示移位寄存器的下一个字节是一个 PEC。
注：　POS 位只能用于主设备接收 2 个字节时。它必须在数据开始接收之前进行配置，如主机接收中建议的 2 字节接收步骤所述。

位 10 **ACK**：应答使能 (Acknowledge enable)
此位由软件置 1 和清零，并可在 PE=0 时由硬件清零。
0：不返回应答
1：在接收一个字节（匹配地址或数据）之后返回应答

位 9 **STOP**：停止位生成 (Stop generation)
该位由软件置 1 和清零，也可在检测到停止位时由硬件清零，在检测到超时错误时由硬件置 1。
在主模式下：
0：不生成停止位。
1：在传输当前字节或发送当前起始位后生成停止位。
在从模式下：
0：不生成停止位。
1：完成当前字节传输后释放 SCL 和 SDA 线。

位 8 **START**：起始位生成 (Start generation)
此位由软件置 1 和清零，并可在起始位发送完成后或 PE=0 时由硬件清零。
在主模式下：
0：不生成起始位
1：生成重复起始位
在从模式下：
0：不生成起始位
1：在总线空闲时生成起始位

位 7 **NOSTRETCH**：禁止时钟延长（从模式）(Clock stretching disable (Slave mode))
在从模式下，当 ADDR 或 BTF 标志置 1 时，此位用于禁止时钟延长，直到软件将其复位为止。
0：使能时钟延长
1：禁止时钟延长

位 6 **ENGC**：广播呼叫使能 (General call enable)
0：禁止广播呼叫。不对地址 00h 应答。
1：使能广播呼叫。对地址 00h 应答。

位 5 **ENPEC**：PEC 使能 (PEC enable)
0：禁止 PEC 计算
1：使能 PEC 计算

位 4 **ENARP**：ARP 使能 (ARP enable)
0：禁止 ARP
1：使能 ARP
SMBTYPE=0 时识别 SMBus 器件默认地址
SMBTYPE=1 时识别 SMBus 主机地址

位 3 **SMBTYPE**：SMBus 类型 (SMBus type)
0：SMBus 器件
1：SMBus 主机

位 2 保留，必须保持复位值

位 1 **SMBUS**：SMBus 模式 (SMBus mode)
0：I²C 模式
1：SMBus 模式

位 0 **PE**：外设使能 (Peripheral enable)
0：禁止外设
1：使能外设
注： 如果此位在通信进行过程中复位，在结束本次通信后回到 IDLE 状态，外设被禁止。由于通信结束时 PE=0，所有位均会复位。
在主模式下，此位不能在通信结束之前复位。

注： STOP、START 或 PEC 位置 1 之后，在硬件将该位清零之前，软件不能对 I2C_CR1 执行任何写操作。否则，可能会再次发出 STOP、START 或 PEC 置 1 请求。

27.6.2 I²C 控制寄存器 2 (I2C_CR2)

I²C 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.	LAST	DMA EN	ITBUF EN	ITEVT EN	ITERR EN	Res.	Res.	FREQ[5:0]					
			rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw

位 15:13 保留，必须保持复位值

位 12 **LAST**：最后一次 DMA 传输 (DMA last transfer)
0：下一个 DMA EOT 不是最后一次传输
1：下一个 DMA EOT 是最后一次传输
注： 此位用于主接收模式，可对最后接收的数据生成 NACK。

位 11 **DMAEN**：DMA 请求使能 (DMA requests enable)
0：禁止 DMA 请求
1：当 TxE=1 或 RxNE =1 时使能 DMA 请求

位 10 **ITBUFEN**：缓冲中断使能 (Buffer interrupt enable)
0：TxE = 1 或 RxNE = 1 时不生成任何中断。
1：TxE = 1 或 RxNE = 1 时生成事件中断（与 DMAEN 状态无关）

位 9 **ITEVTEN**：事件中断使能 (Event interrupt enable)
0：禁止事件中断
1：使能事件中断
满足以下条件时将生成此中断：
– SB = 1（主模式）
– ADDR = 1（主/从模式）
– ADD10= 1（主模式）
– STOPF = 1（从模式）
– BTF = 1，无 TxE 或 RxNE 事件
– ITBUFEN = 1 且 TxE 事件置 1
– ITBUFEN = 1 且 RxNE 事件置 1

位 8 **ITERREN**：错误中断使能 (Error interrupt enable)

0：禁止错误中断

1：使能错误中断

满足以下条件时将生成此中断：

- BERR = 1
- ARLO = 1
- AF = 1
- OVR = 1
- PECERR = 1
- TIMEOUT = 1
- SMBALERT = 1

位 7:6 保留，必须保持复位值

位 5:0 **FREQ[5:0]**：外设时钟频率 (Peripheral clock frequency)

FREQ 位必须使用 APB 时钟频率值进行配置（I2C 外设连接到 APB）。外设使用 FREQ 位域来生成符合 I2C 规范的数据建立和保持时间。允许的最小频率为 2 MHz，最大频率则受限于 APB 最大频率并且不能超过 50 MHz（外设固有上限值）。

0b000000：不允许使用

0b000001：不允许使用

0b000010：2 MHz

...

0b110010：50 MHz

高于 0b101010：不允许使用

27.6.3 I²C 自有地址寄存器 1 (I2C_OAR1)

I²C Own address register 1

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
ADD MODE	Res.	Res.	Res.	Res.	Res.	ADD[9:8]		ADD[7:1]							ADD0
rw						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15 **ADDMODE** 寻址模式（从模式）(Addressing mode (slave mode))

0：7 位从地址（无法应答 10 位地址）

1：10 位从地址（无法应答 7 位地址）

位 14 应通过软件始终保持为 1。

位 13:10 保留，必须保持复位值

位 9:8 **ADD[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

位 7:1 **ADD[7:1]**：接口地址 (Interface address)

地址位 7:1

位 0 **ADD0**：接口地址 (Interface address)

7 位寻址模式：无关

10 位寻址模式：地址位 0

27.6.4 I²C 自有地址寄存器 2 (I2C_OAR2)

I²C Own address register 2

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.	ADD2[7:1]							EN DUAL
								rw	rw	rw	rw	rw	rw	rw	rw

位 15:8 保留，必须保持复位值

位 7:1 **ADD2[7:1]**：接口地址 (Interface address)
双寻址模式下的地址位 7:1

位 0 **ENDUAL**：双寻址模式使能 (Dual addressing mode enable)
0：7 位寻址模式下仅对 OAR1 地址响应
1：7 位寻址模式下能对 OAR1 和 OAR2 两个地址响应

27.6.5 I²C 数据寄存器 (I2C_DR)

I²C Data 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.	DR[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

位 15:8 保留，必须保持复位值

位 7:0 **DR[7:0]**：8 位数据寄存器 (8-bit data register)
接收的字节或者要发送到总线的字节。
– 发送器模式：在 DR 寄存器中写入第一个字节时自动开始发送字节。如果在启动传送 (TxE=1) 后立即将下一个要传送的数据置于 DR 中，则可以保持连续的传送流
– 接收器模式：将接收到的字节复制到 DR 中 (RxNE=1)。如果在接收下一个数据字节 (RxNE=1) 之前读取 DR，则可保持连续的传送流。

注： *在从模式下，地址并不会复制到 DR 中。*
硬件不对写冲突进行管理（TxE=0 时也可对 DR 执行写操作）。
如果发出 ACK 脉冲时出现 ARLO 事件，则不会将接收到的字节复制到 DR 寄存器，因而也无法读取字节。

27.6.6 I²C 状态寄存器 1 (I2C_SR1)

I²C Status register 1

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
SMB ALERT	TIMEOUT	Res.	PEC ERR	OVR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	ADD10	BTF	ADDR	SB
rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	r	r		r	r	r	r	r

位 15 **SMBALERT**：SMBus 报警 (SMBus alert)
在 SMBus 主机模式下：
0：无 SMBALERT
1：引脚上发生 SMBALERT 事件
在 SMBus 从模式下：
0：无 SMBALERT 响应地址头
1：接收到指示 SMBALERT 低电平的 SMBALERT 响应地址头
– 由软件写入 0 来清零，或在 PE=0 时由硬件清零。

位 14 **TIMEOUT**：超时或 Tlow 错误 (Timeout or Tlow error)
0：无超时错误
1：SCL 低电平时长持续 25 ms（超时）
或
主器件累计时钟低电平延长时间超过 10 ms (Tlow:mext)
或
从器件累计时钟低电平延长时间超过 25 ms (Tlow:sext)
– 在从模式下置 1 时：从器件复位通信且硬件释放总线
– 在主模式下置 1 时：由硬件发送停止位
– 由软件写入 0 来清零，或在 PE=0 时由硬件清零。
注： 此功能仅在 SMBus 模式下可用。

位 13 保留，必须保持复位值

位 12 **PECERR**：接收期间的 PEC 错误 (PEC Error in reception)
0：无 PEC 错误：接收器在接收 PEC 后返回 ACK（如果 ACK=1）
1：PEC 错误：接收器在接收 PEC 后返回 NACK（无论 ACK 什么值）
– 由软件写入 0 来清零，或在 PE=0 时由硬件清零。
– 注：接收到错误的 CRC 时，如果在结束 CRC 接收之前 PEC 控制位没有置 1，则 PECERR 位在从模式下不会置 1。不过可以通过读取 PEC 值来判定接收到的 CRC 是否正确。

位 11 **OVR**：上溢 / 下溢 (Overrun/Underrun)
0：未发生上溢 / 下溢
1：上溢或下溢
– 在从模式下由硬件置 1，前提是满足 NOSTRETCH=1 且：
– 接收过程中接收到一个新字节（包括 ACK 脉冲）但尚未读取 DR 寄存器。新接收的字节将丢失。
– 发送过程中将发送一个新字节但尚未向 DR 寄存器写入数据。同一字节发送两次。
– 由软件写入 0 来清零，或在 PE=0 时由硬件清零。
注： 如果 DR 写操作时间与出现 SCL 上升沿的时间非常接近，则发出的数据不确定，并且出现数据保持时间错误。

位 10 **AF**：应答失败 (Acknowledge failure)

0：未发生应答失败
1：应答失败

– 无应答返回时由硬件置 1。
– 由软件写入 0 来清零，或在 PE=0 时由硬件清零。

位 9 **ARLO**：仲裁丢失（主模式）(Arbitration lost (master mode))

0：未检测到仲裁丢失
1：检测到仲裁丢失
当接口在竞争总线中输给另一个主设备时，由硬件将该位置 1

– 由软件写入 0 来清零，或在 PE=0 时由硬件清零。
发生 ARLO 事件后，接口会自动切换回从模式 (MSL=0)。

注：　在 SMBUS 中，从模式下的数据仲裁仅发生在数据阶段或发送确认期间（不适用于地址确认）。

位 8 **BERR**：总线错误 (Bus error)

0：无误放的起始或停止位
1：存在误放的起始或停止位

– SCL 为高电平时，若接口在字节传输期间检测到某个无效位置出现 SDA 上升沿或下降沿，则会由硬件将该位置 1。
– 由软件写入 0 来清零，或在 PE=0 时由硬件清零。

位 7 **TXE**：数据寄存器为空（发送器）(Data register empty (transmitters))

0：数据寄存器非空
1：数据寄存器为空

– 发送过程中 DR 为空时该位置 1。TxE 不会在地址阶段置 1。
– 由软件写入 DR 寄存器来清零，或在出现起始、停止位或者 PE=0 时由硬件清零。
如果接收到 NACK 或要发送的下一个字节为 PEC (PEC=1)，TxE 将不会置 1

注：　写入第一个要发送的数据或在 BTF 置 1 时写入数据都无法将 TxE 清零，因为这两种情况下数据寄存器仍为空。

位 6 **RxNE**：数据寄存器不为空（接收器） (Data register not empty (receivers))

0：数据寄存器为空
1：数据寄存器非空

– 接收模式下数据寄存器非空时置 1。RxNE 不会在地址阶段置 1。
– 由软件读取或写入 DR 寄存器来清零，或在 PE=0 时由硬件清零。
发生 ARLO 事件时 RxNE 不会置 1。

注：　BTF 置 1 时无法通过读取数据将 RxNE 清零，因为此时数据寄存器仍为满。

位 5 保留，必须保持复位值

位 4 **STOPF**：停止位检测（从模式）(Stop detection (slave mode))

0：未检测到停止位
1：检测到停止位

– 从设备在应答脉冲后（如果 ACK=1）检测到停止位，由硬件置 1。
– 由软件分别对 SR1 寄存器和 CR1 寄存器执行读操作和写操作来清零，或在 PE=0 时由硬件清零。

注：　收到 NACK 后 STOPF 位不会置 1。
建议在 STOPF 置 1 后执行完整的清零序列（首先读取 SR1，然后写入 CR1）。请参见第 806 页的图 288：从机接收的传输序列图。

位 3 **ADD10**：发送 10 位头（主模式）(10-bit header sent (Master mode))

0：未发生 ADD10 事件。

1：主器件已发送第一个地址字节（头）。

– 主器件在 10 位地址模式下已发送第一个字节时由硬件置 1。

– 由软件在读取 SR1 寄存器后在 DR 寄存器中写入第二个地址字节来清零，或在 PE=0 时由硬件清零。

注：　收到 NACK 后 ADD10 位不会置 1

位 2 **BTF**：完成数据字节传输 (Byte transfer finished)

0：数据字节传输未完成

1：数据字节传输成功完成

– 由硬件置 1，前提是满足 NOSTRETCH=0 且：

– 接收过程中接收到一个新字节（包括 ACK 脉冲）但尚未读取 DR 寄存器 (RxNE=1)。

– 发送过程中将发送一个新字节但尚未向 DR 寄存器写入数据 (TxE=1)。

– 由软件读或写 DR 寄存器来清零，或在发送过程中出现起始或停止位后由硬件清零，也可以在 PE=0 时由硬件清零。

注：　收到 NACK 后 BTF 位不会置 1

如果下一个要发送的字节为 PEC（I2C_SR2 寄存器中的 TRA=1，I2C_CR1 寄存器中的 PEC=1），则 BTF 位不会置 1

位 1 **ADDR**：地址已发送（主模式）/ 地址匹配（从模式）(Address sent (master mode)/matched (slave mode))

该位由软件在读取 SR1 寄存器后读取 SR2 寄存器来清零，或在 PE=0 时由硬件清零。

地址匹配（从模式）

0：地址不匹配或未接收到地址。

1：接收到的地址匹配。

– 当接收到的从地址与 OAR 寄存器内容、广播呼叫地址或 SMBus 器件默认地址匹配时，或者识别到 SMBus 主机或 SMBus 报警时，该位由硬件置 1。（根据配置确定何时使能）。

注：　在从模式下，建议在 ADDR 置 1 后执行完整的清零序列（首先读取 SR1，然后写入 SR2）。请参见第 806 页的图 288：从机接收的传输序列图。

地址已发送（主模式）

0：地址发送未结束

1：地址发送结束

– 在 10 位寻址模式下，接收到第二个地址字节的 ACK 后该位置 1。

– 在 7 位寻址模式下，接收到地址字节的 ACK 后该位置 1。

注：　收到 NACK 后 ADDR 位不会置 1

位 0 **SB**：起始位（主模式）(Start bit (Master mode))

0：无起始位

1：起始位已经发送。

– 生成启动条件时置 1。

– 由软件在读取 SR1 寄存器后写入 DR 寄存器来清零，或在 PE=0 时由硬件清零

27.6.7 I2C 状态寄存器 2 (I2C_SR2)

I2C Status register 2

偏移地址：0x18
复位值：0x0000

注： *读取 I2C_SR1 后再读取 I2C_SR2 可将 ADDR 标志清零，即使 ADDR 标志在读取 I2C_SR1 之后置 1 也如此。因此，必须仅在 I2C_SR1 中的 ADDR 位已置 1 或者 STOPF 位已清零后读取 I2C_SR2。*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PEC[7:0]								DUALF	SMB HOST	SMB DEFAULT	GEN CALL	Res.	TRA	BUSY	MSL
r	r	r	r	r	r	r	r	r	r	r	r		r	r	r

位 15:8 **PEC[7:0]**：数据包错误校验寄存器 (Packet error checking register)
ENPEC=1 时，此寄存器包含内部 PEC。

位 7 **DUALF**：双标志（从模式）(Dual flag (Slave mode))
0：接收到的地址与 OAR1 匹配
1：接收到的地址与 OAR2 匹配
– 出现停止位、重复起始位或 PE=0 时由硬件清零。

位 6 **SMBHOST**：SMBus 主机头（从模式）(SMBus host header (Slave mode))
0：无 SMBus 主机地址
1：SMBTYPE=1 且 ENARP=1 时接收到 SMBus 主机地址。
– 出现停止位、重复起始位或 PE=0 时由硬件清零。

位 5 **SMBDEFAULT**：SMBus 器件默认地址（从模式）(SMBus device default address (Slave mode))
0：无 SMBus 器件默认地址
1：ENARP=1 时接收到 SMBus 器件默认地址
– 出现停止位、重复起始位或 PE=0 时由硬件清零。

位 4 **GENCALL**：广播呼叫地址（从模式）(General call address (Slave mode))
0：无广播呼叫
1：ENGC=1 时接收到广播呼叫地址
– 出现停止位、重复起始位或 PE=0 时由硬件清零。

位 3 保留，必须保持复位值

位 2 **TRA**：发送器/接收器 (Transmitter/receiver)

0：接收器

1：发送器

此位在整个地址阶段的结尾处根据地址字节的 R/W 位状态进行置 1。

同样，检测到停止位 (STOPF=1)、重复起始位、总线仲裁丢失 (ARLO=1) 或当 PE=0 时该位也由硬件清零。

位 1 **BUSY**：总线繁忙 (Bus busy)

0：总线上无通信

1：总线正在进行通信

– 检测到 SDA 或 SCL 低电平时由硬件置 1

– 检测到停止位时由硬件清零。

该位指示总线上是否正在进行通信。即使禁止接口 (PE=0) 后此信息也会更新。

位 0 **MSL**：主/从 (Master/slave)

0：从模式

1：主模式

– 接口进入主模式时 (SB=1) 由硬件置 1。

– 检测到总线上的停止位、仲裁丢失 (ARLO=1) 或当 PE=0 时由硬件清零。

注： *读取 I2C_SR1 后再读取 I2C_SR2 可将 ADDR 标志清零，即使 ADDR 标志在读取 I2C_SR1 之后置 1 也如此。因此，必须仅在 I2C_SR1 中的 ADDR 位已置 1 或者 STOPF 位已清零后读取 I2C_SR2。*

27.6.8 I²C 时钟控制寄存器 (I2C_CCR)

I²C Clock control register

偏移地址：0x1C
复位值：0x0000

注： *f_{PCLK1} 必须至少为 2 MHz，才能达到 Sm 模式 I²C 频率。必须至少为 4 MHz 才能达到 Fm 模式 I²C 频率。必须为 10MHz 的倍数才能实现最大 400 kHz 的 I²C Fm 模式时钟。*

CCR 寄存器必须仅在禁止 I2C (PE = 0) 的情况下配置。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F/S	DUTY	Res.	Res.	CCR[11:0]											
rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15 **F/S**：I2C 主模式选择 (I2C master mode selection)

0：Sm 模式 I2C

1：Fm 模式 I2C

位 14 **DUTY**：Fm 模式占空比 (Fm mode duty cycle)
0：Fm 模式 t_{low}/t_{high} = 2
1：Fm 模式 t_{low}/t_{high} = 16/9（请参见 CCR）

位 13:12 保留，必须保持复位值

位 11:0 **CCR[11:0]**：Fm/Sm 模式下的时钟控制寄存器（主模式）(Clock control register in Fm/Sm mode (Master mode))
控制主模式下的 SCL 时钟。
Sm 模式或 SMBus 模式：
T_{high} = CCR * T_{PCLK1}
T_{low} = CCR * T_{PCLK1}
Fm 模式：
如果 DUTY = 0：
T_{high} = CCR * T_{PCLK1}
T_{low} = 2 * CCR * T_{PCLK1}
如果 DUTY = 1：（达到 400 kHz）
T_{high} = 9 * CCR * T_{PCLK1}
T_{low} = 16 * CCR * T_{PCLK1}
例如：要在 Sm 模式下生成 100 kHz 的 SCL 频率：
如果 FREQR = 08，T_{PCLK1} = 125 ns，则必须将 CCR 编程为 0x28
(0x28 <=> 40d x 125 ns = 5000 ns)。
注：　允许的最小值为 0x04，但快速占空比模式例外，其最小值为 0x01
t_{high} = $t_{r(SCL)}$ + $t_{w(SCLH)}$。有关参数的定义，请参见器件数据手册。
t_{low} = $t_{f(SCL)}$ + $t_{w(SCLL)}$。有关参数的定义，请参见器件数据手册。
I2C 通信速度，fSCL ~ 1/(thigh + tlow)。实际频率可能会因模拟噪声滤波器输入延迟而有所不同。
CCR 寄存器必须仅在禁止 I²C (PE = 0) 的情况下配置。

27.6.9 I²C TRISE 寄存器 (I2C_TRISE)

I²C TRISE 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.	Res.	Res.	Res.	Res.	TRISE[5:0]					
										rw	rw	rw	rw	rw	rw

位 15:6 保留，必须保持复位值

位 5:0 **TRISE[5:0]**：Fm/Sm 模式下的最大上升时间（主模式）(Maximum rise time in Fm/Sm mode (Master mode))
这些位应提供 SCL 反馈回路在主模式下的最大持续时间， 目的在于无论 SCL 上升沿持续多长时间，都能保持稳定的 SCL 频率。
这些位必须编程为 I²C 总线规范中给定的最大 SCL 上升时间加 1。
例如，Sm 模式下允许的最大 SCL 上升时间为 1000 ns。
如果 I2C_CR2 寄存器中 FREQ[5:0] 位的值等于 0x08 且 T_{PCLK1} = 125 ns，则 TRISE[5:0] 位必须编程为 09h。
(1000 ns / 125 ns = 8 + 1)
滤波器值也可以叠加到 TRISE[5:0]。
如果结果不为整数，则 TRISE[5:0] 必须编程为整数部分，以符合 t_{HIGH} 参数要求。
注：　TRISE[5:0] 必须仅在禁止 I2C (PE = 0) 的情况下配置。

27.6.10 I2C FLTR 寄存器 (I2C_FLTR)

I2C FLTR register

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ANOFF	DNF[3:0]			
											rw	rw	rw	rw	rw

位 15:5 保留，必须保持复位值

位 4 **ANOFF**：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： ANOFF 必须仅在禁止 I2C (PE = 0) 的情况下配置。

位 3:0 **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。数字滤波器可抑制脉宽达 DNF[3:0] * TPCLK1 以下的尖峰。

0000：禁止数字噪声滤波器

0001：使能数字噪声滤波器，滤波能力可达 1* TPCLK1。

...

1111：使能数字噪声滤波器，滤波能力可达 15* TPCLK1。

注： DNF[3:0] 必须仅在禁止 I2C (PE = 0) 的情况下配置。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。

27.6.11 I²C 寄存器映射

下表提供了 I²C 寄存器映射和复位值。

表 156. 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
0x00	I2C_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWRST	Res.	ALERT	PEC	POS	ACK	STOP	START	NOSTRETCH	ENGC	ENPEC	ENARP	SMBTYPE	Res.	SMBUS	PE
	Reset value																	0		0	0	0	0	0	0	0	0	0	0	0		0	0
0x04	I2C_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN	Res.	Res.	FREQ[5:0]					
	Reset value																				0	0	0	0	0			0	0	0	0	0	0
0x08	I2C_OAR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADDMODE	Res.	Res.	Res.	Res.	Res.	ADD[9:8]		ADD[7:1]							ADD0
	Reset value																	0						0	0	0	0	0	0	0	0	0	0
0x0C	I2C_OAR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADD2[7:1]							ENDUAL
	Reset value																									0	0	0	0	0	0	0	0
0x10	I2C_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x14	I2C_SR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMBALERT	TIMEOUT	Res.	PECERR	OVR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	ADD10	BTF	ADDR	SB
	Reset value																	0	0		0	0	0	0	0	0	0		0	0	0	0	0
0x18	I2C_SR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PEC[7:0]								DUALF	SMBHOST	SMBDEFAULT	GENCALL	Res.	TRA	BUSY	MSL
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0
0x1C	I2C_CCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	F/S	DUTY	Res.	Res.	CCR[11:0]											
	Reset value																	0	0			0	0	0	0	0	0	0	0	0	0	0	0
0x20	I2C_TRISE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRISE[5:0]					
	Reset value																											0	0	0	0	1	0
0x24	I2C_FLTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ANOFF	DNF[3:0]			
	Reset value																												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*。

28 通用同步收发器 (USART)/通用异步收发器 (UART)

28.1 USART 简介

通用同步异步收发器 (USART) 能够灵活地与外部设备进行全双工数据交换，满足外部设备对工业标准 NRZ 异步串行数据格式的要求。USART 通过小数波特率发生器提供了多种波特率。

它支持同步单向通信和半双工单线通信； 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)。而且，它还支持多处理器通信。

通过配置多个缓冲区使用 DMA 可实现高速数据通信。

28.2 USART 主要特性

- 全双工的异步通信
- 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
- 小数波特率发生器系统
  - 通用可编程收发波特率（有关最大 APB 频率时的波特率值，请参见数据手册）
- 数据字长度可编程（8 位或 9 位）
- 停止位可配置——支持 1 或 2 个停止位
- LIN 主模式同步停止符号发送功能和 LIN 从模式停止符号检测功能
  - 对 USART 进行 LIN 硬件配置时可生成 13 位停止符号和检测 10/11 位停止符号
- 用于同步发送的发送器时钟输出
- IrDA SIR 编码解码器
  - 正常模式下，支持 3/16 位持续时间
- 智能卡仿真功能
  - 智能卡接口支持符合 ISO 7816-3 标准中定义的异步协议智能卡
  - 智能卡工作模式下，支持 0.5 或 1.5 个停止位
- 单线半双工通信
- 使用 DMA（直接存储器访问）实现可配置的多缓冲区通信
  - 使用 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
- 传输检测标志：
  - 接收缓冲区已满
  - 发送缓冲区为空
  - 传输结束标志
- 奇偶校验控制：
  - 发送奇偶校验位
  - 检查接收的数据字节的奇偶性

- 四个错误检测标志：
  - 上溢错误
  - 噪声检测
  - 帧错误
  - 奇偶校验错误
- 十个具有标志位的中断源：
  - CTS 变化
  - LIN 停止符号检测
  - 发送数据寄存器为空
  - 发送完成
  - 接收数据寄存器已满
  - 接收到线路空闲
  - 上溢错误
  - 帧错误
  - 噪声错误
  - 奇偶校验错误
- 多处理器通信——如果地址不匹配，则进入静默模式
- 从静默模式唤醒（通过线路空闲检测或地址标记检测）
- 两个接收器唤醒模式：地址位（MSB，第 9 位），线路空闲

28.3 USART 实现

本节介绍了 USART1 中实现的所有特性。有关 USART 实例之间的差异，请参见*表 157：USART 特性*。

表 157. USART 特性

USART 模式/特性(1)	USART 1	USART 2	USART 3	UART 4	UART 5	USART 6	UART 7	UART 8	UART 9	UART 10
异步模式	X	X	X	X	X	X	X	X	X	X
硬件流控制	X	X	X	NA	NA	X	NA	NA	NA	NA
多缓冲区通信 (DMA)	X	X	X	X	X	X	X	X	X	X
多处理器通信	X	X	X	X	X	X	X	X	X	X
同步模式	X	X	X	NA	NA	X	NA	NA	NA	NA
智能卡模式	X	X	X	NA	NA	X	NA	NA	NA	NA
半双工（单线模式）	X	X	X	X	X	X	X	X	X	X
IrDA SIR ENDEC 模块	X	X	X	X	X	X	X	X	X	X
IrDA 模式	X	X	X	X	X	X	X	X	X	X
LIN 模式	X	X	X	X	X	X	X	X	X	X

1. X = 支持。

28.4 USART 功能说明

接口通过三个引脚从外部连接到其他设备（请参见*图 292*）。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：接收数据输入引脚就是串行数据输入引脚。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（USART 电平下，随后在 SW_RX 上接收数据）。

正常 USART 模式下，通过这些引脚以帧的形式发送和接收串行数据：

- 发送或接收前保持空闲线路
- 起始位
- 数据字（字长 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
- 该接口使用小数波特率发生器——带 12 位尾数和 4 位小数
- 状态寄存器 (USART_SR)
- 数据寄存器 (USART_DR)
- 波特率寄存器 (USART_BRR)——12 位尾数和 4 位小数
- 智能卡模式下的保护时间寄存器 (USART_GTPR)

有关各个位的定义，请参见*第 28.6 节：USART 寄存器*。

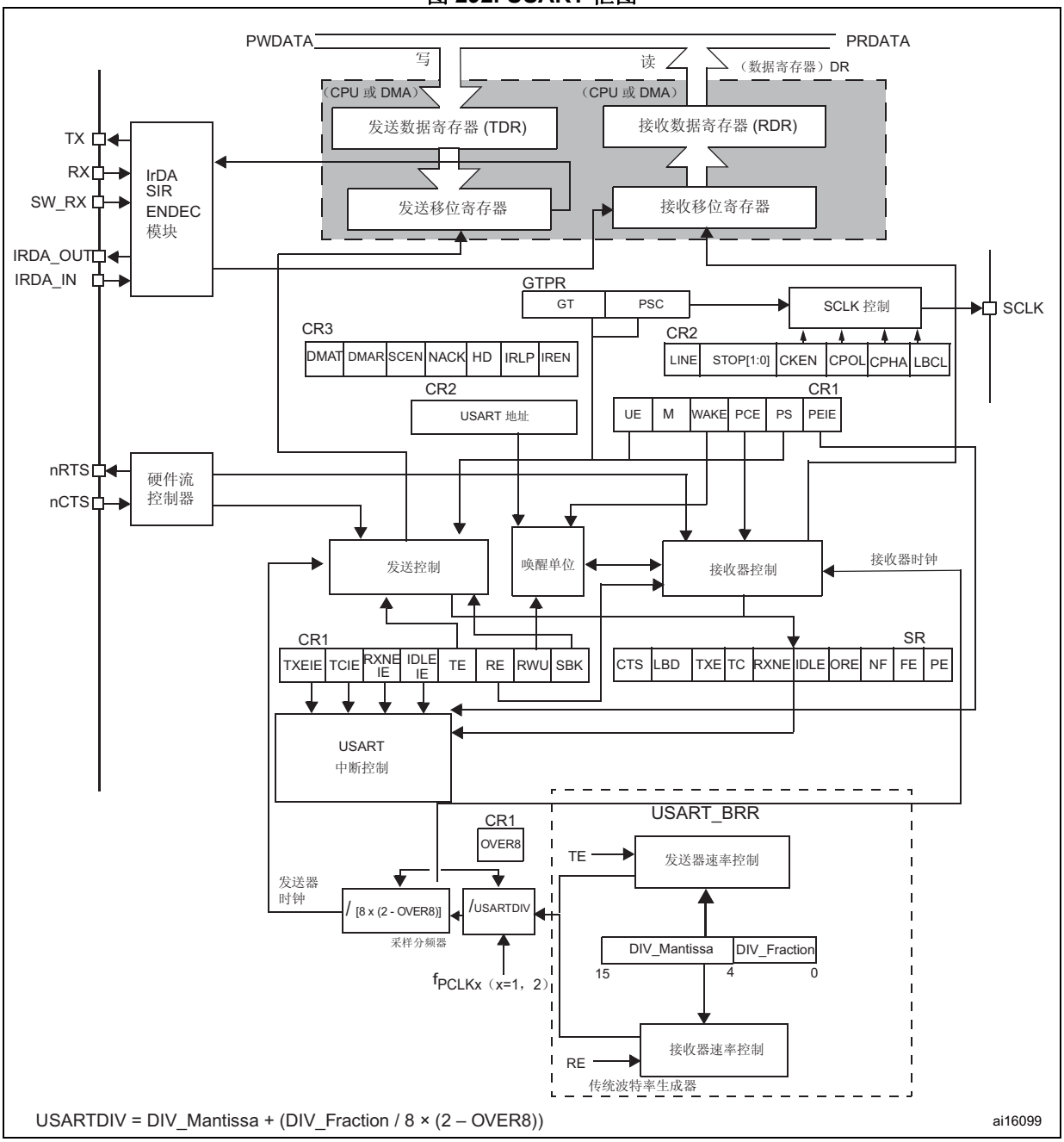
在同步模式下连接时需要以下引脚：

- **SCLK**：发送器时钟输出。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（例如 LCD 驱动器）。时钟相位和极性可通过软件编程。在智能卡模式下，SCLK 可向智能卡提供时钟。

在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）

图 292. USART 框图



28.4.1 USART 字符说明

可通过对 USART_CR1 寄存器中的 M 位进行编程来选择 8 位或 9 位的字长 （请参见*图 293*）。

TX 引脚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。

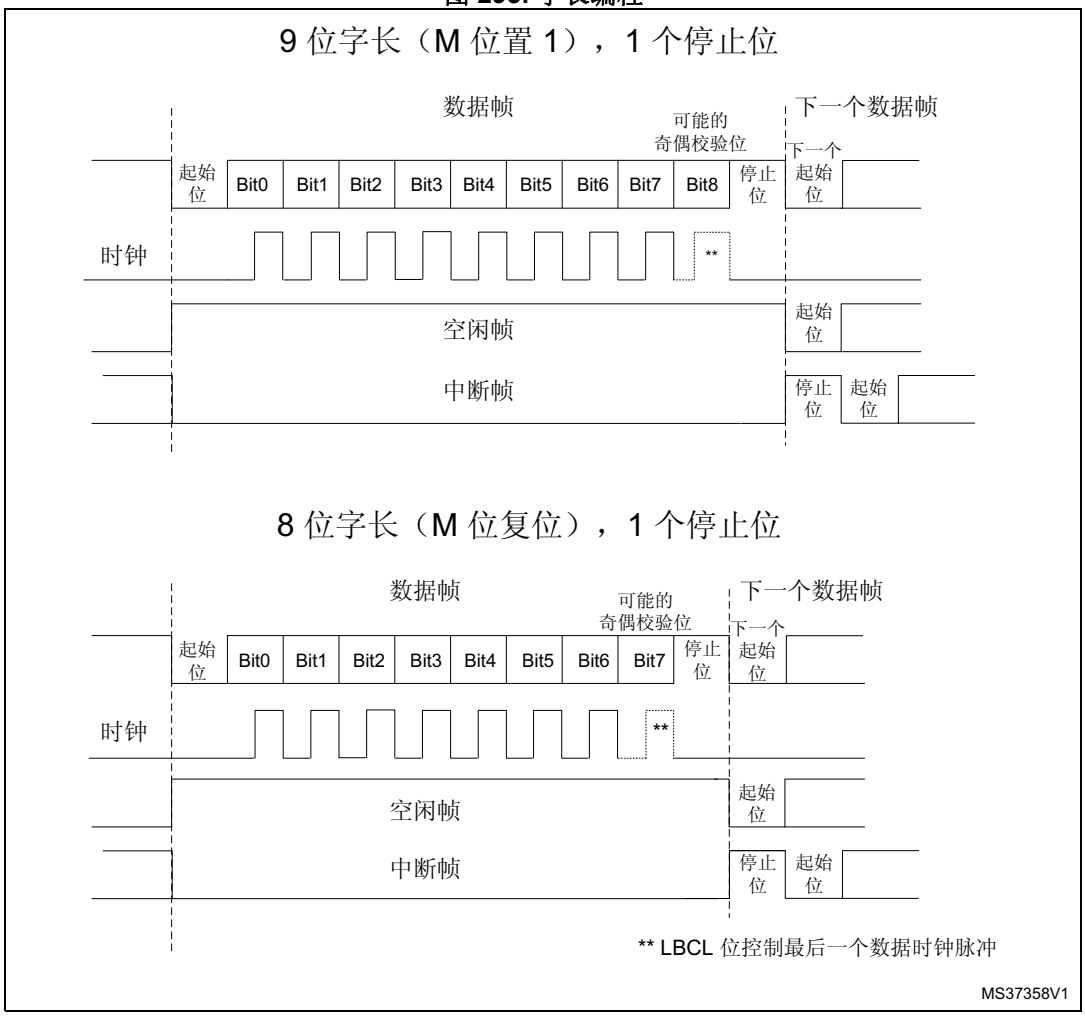
空闲字符可理解为整个帧周期内电平均为“1”（停止位的电平也是“1”），该字符后是下一个数据帧的起始位。

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 1 或 2 个停止位（逻辑“1”位）以确认起始位。

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 293. 字长编程



28.4.2 发送器

发送器可发送 8 位或 9 位的数据字，具体取决于 M 位的状态。发送使能位 (TE) 置 1 时，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。该模式下，USART_DR 寄存器的缓冲区 (TDR) 位于内部总线和发送移位寄存器之间（请参见*图 292*）。

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： *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

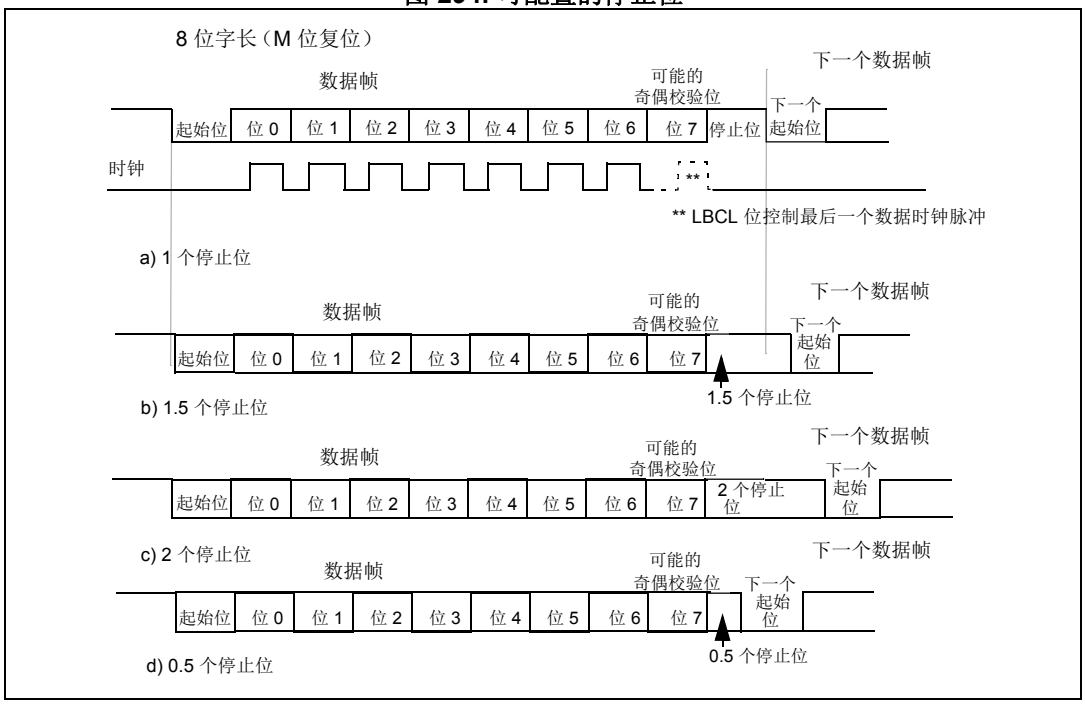
可以在控制寄存器 2 的位 13 和位 12 中编程将随各个字符发送的停止位的数量。

- ***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。
- ***0.5 个停止位***：在智能卡模式下接收数据时使用。
- ***1.5 个停止位***：在智能卡模式下发送和接收数据时使用。

空闲帧发送将包括停止位。

m = 0 时，中断发送是 10 个低电平位，然后是已配置数量的停止位；m = 1 时，中断发送是 11 个低电平位，然后是已配置数量的停止位。无法传送长中断（中断长度大于 10/11 个低电平位）。

图 294. 可配置的停止位



步骤：

1. 通过向 USART_CR1 寄存器中的 UE 位写入 1 使能 USART。
2. 对 USART_CR1 中的 M 位进行编程以定义字长。
3. 对 USART_CR2 中的停止位数量进行编程。
4. 如果将进行多缓冲区通信，请选择 USART_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。
5. 使用 USART_BRR 寄存器选择所需波特率。
6. 将 USART_CR1 中的 TE 位置 1 以便在首次发送时发送一个空闲帧。
7. 在 USART_DR 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 USART_DR 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

始终通过向数据寄存器写入数据来将 TXE 位清零。

TXE 位由硬件置 1，它表示：

- 数据已从 TDR 移到移位寄存器中且数据发送已开始。
- TDR 寄存器为空。
- USART_DR 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 USART_DR 寄存器的写指令中存有 TDR 寄存器中的数据，该数据将在当前发送结束时复制到移位寄存器中。

未发送时，要传入 USART_DR 寄存器的写指令直接将数据置于移位寄存器中，数据发送开始时，TXE 位立即置 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 USART_CR1 寄存器中的 TCIE 位置 1，将生成中断。

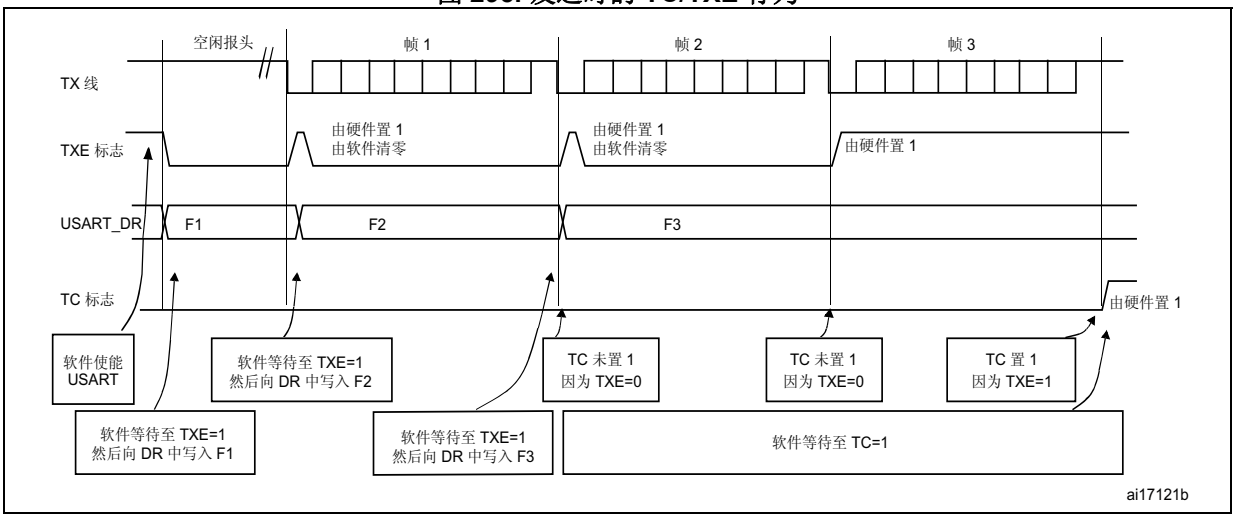
向 USART_DR 寄存器中写入最后一个数据后，必须等待至 TC=1，之后才可禁止 USART 或使微控制器进入低功率模式（请参见*图 295：发送时的 TC/TXE 行为*）。

TC 位通过以下软件序列清零：

1. 从 USART_SR 寄存器读取数据
2. 向 USART_DR 寄存器写入数据

注： *还可通过向 TC 位写入“0”将其清零。建议仅在多缓冲区通信时使用此清零序列。*

图 295. 发送时的 TC/TXE 行为



中断字符

将 SBK 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 293*）。

如果 SBK 位置“1”，当前字符发送完成后，将在 TX 线路上发送一个中断字符。中断字符发送完成时（发送中断字符的停止位期间），该位由硬件复位。USART 在上一个中断帧的末尾插入一个逻辑“1”位，以确保识别下个帧的起始位。

注： *如果软件在中断发送开始前对 SBK 位进行了复位，将不会发送中断字符。对于两个连续的中断，应在上一个中断的停止位发送完成后将 SBK 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。

28.4.3 接收器

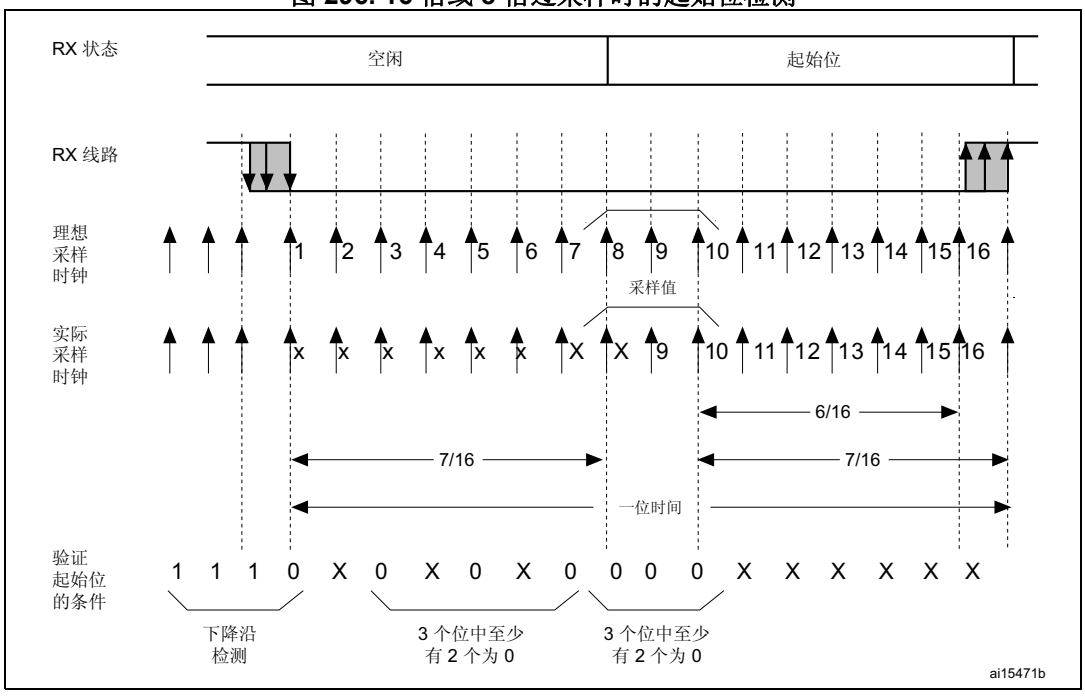
USART 可接收 8 位或 9 位的数据字，具体取决于 USART_CR1 寄存器中的 M 位。

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 0 X 0 0 0 0。

图 296. 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）。

如果两次采样时（对第 3 位、第 5 位和第 7 位进行采样以及对第 8 位、第 9 位和第 10 位进行采样），3 个采样位中至少有 2 个为 0，则可验证起始位（RXNE 标志位置 1，RXNEIE=1 时生成中断）但 NE 噪声标志位置 1。如果不满足此条件，则启动检测中止，接收器返回空闲状态（无标志位置 1）。

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，则可验证起始位但 NE 噪声标志位置 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。该模式下，USART_DR 寄存器的缓冲区 (RDR) 位于内部总线和接收移位寄存器之间。

步骤：

1. 通过向 USART_CR1 寄存器中的 UE 位写入 1 使能 USART。
2. 对 USART_CR1 中的 M 位进行编程以定义字长。
3. 对 USART_CR2 中的停止位数量进行编程。
4. 如果将进行多缓冲区通信，请选择 USART_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。
5. 使用波特率寄存器 USART_BRR 选择所需波特率。
6. 将 RE 位 USART_CR1 置 1。这一操作将使能接收器开始搜索起始位。

接收到字符时

- 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。
- 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 USART_DR 寄存器执行读操作将 RXNE 位清零。RXNE 标志也可以通过向该位写入零来清零。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。

注： *接收数据时，不应将 RE 位复位。如果接收期间禁止了 RE 位，则会中止接收当前字节。*

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 位为 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 USART_DR 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 与 DMAR 位均为 1，则会生成中断。
- 通过先后对 USART_SR 寄存器和 USART_DR 寄存器执行读操作将 ORE 位清除。

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。读取序列期间（在 USART_SR 寄存器读访问与 USART_DR 读访问之间）接收到新数据时也会发生该情况。

选择合适的过采样方法

接收器采用不同的用户可配置过采样技术（除了同步模式下），可以从噪声中提取有效数据。

可通过编程 USART_CR1 寄存器中的 OVER8 位来选择采样方法，且采样时钟可以是波特率时钟的 16 倍或 8 倍（请参见*图 297* 和*图 298*）。

根据应用：

- 选择 8 倍过采样 (OVER8=1) 以获得更高的速度（高达 $f_{PCLK}/8$）。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 28.4.5 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 $f_{PCLK}/16$

可通过编程 USART_CR3 寄存器中的 ONEBIT 位选择用于评估逻辑电平的方法。有两种选择：

- 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 158*），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 28.4.5 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 USART_DR 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，USART_CR3 寄存器中的 EIE 位置 1 时将发出中断。

通过先后对 USART_SR 寄存器和 USART_DR 寄存器执行读操作将 NF 位清零。

注： *智能卡、IrDA 和 LIN 模式下不可采用 8 倍过采样。在这些模式下，OVER8 位由硬件强制清零。*

图 297. 16 倍过采样时的数据采样

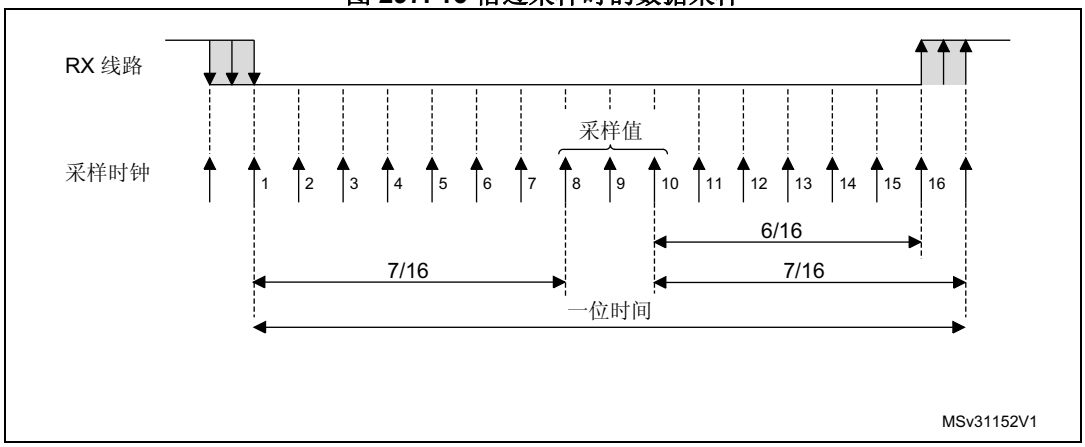


图 298. 8 倍过采样时的数据采样

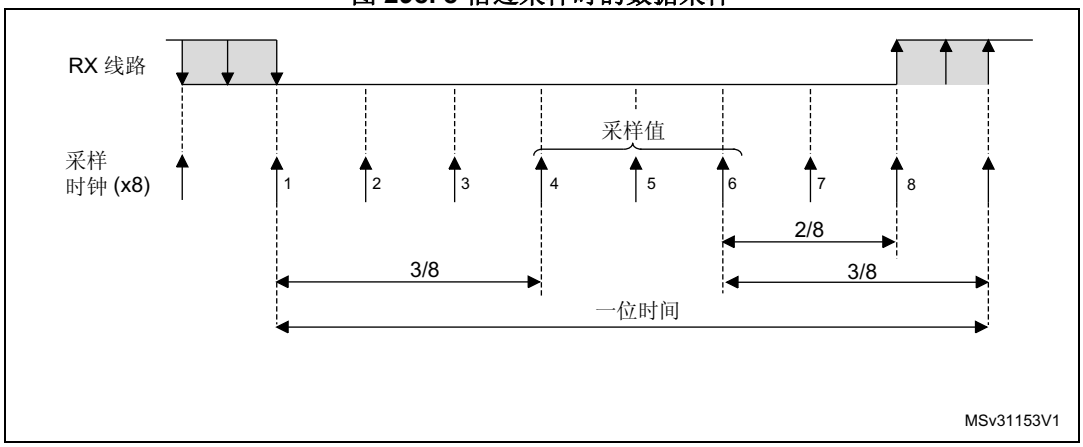


表 158. 通过采样数据进行噪声检测

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 USART_DR 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，USART_CR3 寄存器中的 EIE 位置 1 时将发出中断。

通过先后对 USART_SR 寄存器和 USART_DR 寄存器执行读操作将 FE 位清零。

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 或 2 个（正常模式下），也可以是 0.5 或 1.5 个（智能卡模式下）。

1. ***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。
2. ***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。
3. ***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（USART_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 28.4.11 节*。
4. ***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。

28.4.4 小数波特率生成

对 USARTDIV 的尾数值和小数值进行编程时，接收器和发送器（Rx 和 Tx）的波特率均设置为相同值。

公式 1：适用于标准 USART（包括 SPI 模式）的波特率

$$\text{Tx/Rx波特率} = \frac{f_{CK}}{8 \times (2 - OVER8) \times USARTDIV}$$

公式 2：智能卡、LIN 和 IrDA 模式下的波特率

$$\text{Tx/Rx波特率} = \frac{f_{CK}}{16 \times USARTDIV}$$

USARTDIV 是一个存放在 USART_BRR 寄存器中的无符号定点数。

- 当 OVER8=0 时，小数部分编码为 4 位并通过 USART_BRR 寄存器中的 DIV_fraction[3:0] 位编程。
- 当 OVER8=1 时，小数部分编码为 3 位并通过 USART_BRR 寄存器中的 DIV_fraction[2:0] 位编程，此时 DIV_fraction[3] 位必须保持清零状态。

注： *对 USART_BRR 执行写操作后，波特率计数器更新为波特率寄存器中的新值。因此，波特率寄存器的值不应在通信时发生更改。*

OVER8=0 时如何从 USART_BRR 寄存器值中获取 USARTDIV

示例 1：

如果 DIV_Mantissa = 0d27 且 DIV_Fraction = 0d12 (USART_BRR = 0x1BC)，则

尾数 (USARTDIV) = 0d27

小数 (USARTDIV) = 12/16 = 0d0.75

因此 USARTDIV = 0d27.75

示例 2：

要设定 USARTDIV = 0d25.62

这将导致：

DIV_Fraction = 16*0d0.62 = 0d9.92

最接近的实数为 0d10 = 0xA

DIV_Mantissa = 尾数 (0d25.620) = 0d25 = 0x19

则 USART_BRR = 0x19A，因此 USARTDIV = 0d25.625

示例 3：

要设定 USARTDIV = 0d50.99

这将导致：

DIV_Fraction = 16*0d0.99 = 0d15.84

最接近的实数为 0d16 = 0x10 => DIV_frac[3:0] 溢出 => 尾数必须添加进位

DIV_Mantissa = 尾数（0d50.990 + 进位）= 0d51 = 0x33

则 USART_BRR = 0x330，因此 USARTDIV = 0d51.000

OVER8=1 时如何从 USART_BRR 寄存器中获取 USARTDIV

示例 1：

如果 DIV_Mantissa = 0x27 且 DIV_Fraction[2:0]= 0d6 (USART_BRR = 0x1B6)，则

尾数 (USARTDIV) = 0d27

小数 (USARTDIV) = 6/8 = 0d0.75

因此 USARTDIV = 0d27.75

示例 2：

要设定 USARTDIV = 0d25.62

这将导致：

DIV_Fraction = 8*0d0.62 = 0d4.96

最接近的实数为 0d5 = 0x5

DIV_Mantissa = 尾数 (0d25.620) = 0d25 = 0x19

则 USART_BRR = 0x195 => USARTDIV = 0d25.625

示例 3：

要设定 USARTDIV = 0d50.99

这将导致：

DIV_Fraction = 8*0d0.99 = 0d7.92

最接近的实数为 0d8 = 0x8 => DIV_frac[2:0] 溢出 => 尾数必须添加进位

DIV_Mantissa = 尾数（0d50.990 + 进位）= 0d51 = 0x33

则 USART_BRR = 0x0330 => USARTDIV = 0d51.000

表 159. 采用 16 倍过采样时，在 f_{PCLK} = 8 MHz 或 f_{PCLK} = 12 MHz 下编程波特率时的误差计算(1)

16 倍过采样时 (OVER8=0)							
波特率		f_{PCLK} = 8 MHz			f_{PCLK} = 12 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	1.2 KBps	1.2 KBps	416.6875	0	1.2 KBps	625	0
2	2.4 KBps	2.4 KBps	208.3125	0.01	2.4 KBps	312.5	0
3	9.6 KBps	9.604 KBps	52.0625	0.04	9.6 KBps	78.125	0
4	19.2 KBps	19.185 KBps	26.0625	0.08	19.2 KBps	39.0625	0
5	38.4 KBps	38.462 KBps	13	0.16	38.339 KBps	19.5625	0.16
6	57.6 KBps	57.554 KBps	8.6875	0.08	57.692 KBps	13	0.16
7	115.2 KBps	115.942 KBps	4.3125	0.64	115.385 KBps	6.5	0.16
8	230.4 KBps	228.571 KBps	2.1875	0.79	230.769 KBps	3.25	0.16
9	460.8 KBps	470.588 KBps	1.0625	2.12	461.538 KBps	1.625	0.16

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 160. 采用 8 倍过采样时，在 f_{PCLK} = 8 MHz 或 f_{PCLK} = 12 MHz 下编程波特率时的误差计算(1)

8 倍过采样 (OVER8 = 1)							
波特率		f_{PCLK} = 8 MHz			f_{PCLK} = 12 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	1.2 KBps	1.2 KBps	833.375	0	1.2 KBps	1250	0
2	2.4 KBps	2.4 KBps	416.625	0.01	2.4 KBps	625	0
3	9.6 KBps	9.604 KBps	104.125	0.04	9.6 KBps	156.25	0
4	19.2 KBps	19.185 KBps	52.125	0.08	19.2 KBps	78.125	0
5	38.4 KBps	38.462 KBps	26	0.16	38.339 KBps	39.125	0.16

表 160. 采用 8 倍过采样时，在 f_{PCLK} = 8 MHz 或 f_{PCLK} = 12 MHz 下编程波特率时的误差计算(1)（续）

8 倍过采样 (OVER8 = 1)							
波特率		f_{PCLK} = 8 MHz			f_{PCLK} = 12 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
6	57.6 KBps	57.554 KBps	17.375	0.08	57.692 KBps	26	0.16
7	115.2 KBps	115.942 KBps	8.625	0.64	115.385 KBps	13	0.16
8	230.4 KBps	228.571 KBps	4.375	0.79	230.769 KBps	6.5	0.16
9	460.8 KBps	470.588 KBps	2.125	2.12	461.538 KBps	3.25	0.16
10	921.6 KBps	888.889 KBps	1.125	3.55	923.077 KBps	1.625	0.16

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 161. 采用 16 倍过采样时，在 f_{PCLK} = 16 MHz 或 f_{PCLK} = 24 MHz 下编程波特率时的误差计算(1)

16 倍过采样 (OVER8 = 0)							
波特率		f_{PCLK} = 16 MHz			f_{PCLK} = 24 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	1.2 KBps	1.2 KBps	833.3125	0	1.2	1250	0
2	2.4 KBps	2.4 KBps	416.6875	0	2.4	625	0
3	9.6 KBps	9.598 KBps	104.1875	0.02	9.6	156.25	0
4	19.2 KBps	19.208 KBps	52.0625	0.04	19.2	78.125	0
5	38.4 KBps	38.369 KBps	26.0625	0.08	38.4	39.0625	0
6	57.6 KBps	57.554 KBps	17.375	0.08	57.554	26.0625	0.08
7	115.2 KBps	115.108 KBps	8.6875	0.08	115.385	13	0.16
8	230.4 KBps	231.884 KBps	4.3125	0.64	230.769	6.5	0.16
9	460.8 KBps	457.143 KBps	2.1875	0.79	461.538	3.25	0.16
10	921.6 KBps	941.176 KBps	1.0625	2.12	923.077	1.625	0.16

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 162. 采用 8 倍过采样时，在 f_{PCLK} = 16 MHz 或 f_{PCLK} = 24 MHz 下编程波特率时的误差计算(1)

8 倍过采样 (OVER8=1)							
波特率		f_{PCLK} = 16 MHz			f_{PCLK} = 24 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	1.2 KBps	1.2 KBps	1666.625	0	1.2 KBps	2500	0
2	2.4 KBps	2.4 KBps	833.375	0	2.4 KBps	1250	0
3	9.6 KBps	9.598 KBps	208.375	0.02	9.6 KBps	312.5	0
4	19.2 KBps	19.208 KBps	104.125	0.04	19.2 KBps	156.25	0
5	38.4 KBps	38.369 KBps	52.125	0.08	38.4 KBps	78.125	0
6	57.6 KBps	57.554 KBps	34.75	0.08	57.554 KBps	52.125	0.08
7	115.2 KBps	115.108 KBps	17.375	0.08	115.385 KBps	26	0.16
8	230.4 KBps	231.884 KBps	8.625	0.64	230.769 KBps	13	0.16
9	460.8 KBps	457.143 KBps	4.375	0.79	461.538 KBps	6.5	0.16
10	921.6 KBps	941.176 KBps	2.125	2.12	923.077 KBps	3.25	0.16
11	2 MBps	2000 KBps	1	0	2000 KBps	1.5	0
12	3 MBps	NA	NA	NA	3000 KBps	1	0

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 163. 采用 16 倍过采样时，在 f_{PCLK} = 8 MHz 或 f_{PCLK} = 16 MHz 下编程波特率时的误差计算(1)

16 倍过采样时 (OVER8=0)							
波特率		f_{PCLK} = 8 MHz			f_{PCLK} = 16 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	2.4 KBps	2.400 KBps	208.3125	0.00%	2.400 KBps	416.6875	0.00%
2	9.6 KBps	9.604 KBps	52.0625	0.04%	9.598 KBps	104.1875	0.02%
3	19.2 KBps	19.185 KBps	26.0625	0.08%	19.208 KBps	52.0625	0.04%
4	57.6 KBps	57.554 KBps	8.6875	0.08%	57.554 KBps	17.3750	0.08%
5	115.2 KBps	115.942 KBps	4.3125	0.64%	115.108 KBps	8.6875	0.08%
6	230.4 KBps	228.571 KBps	2.1875	0.79%	231.884 KBps	4.3125	0.64%
7	460.8 KBps	470.588 KBps	1.0625	2.12%	457.143 KBps	2.1875	0.79%
8	896 KBps	NA	NA	NA	888.889 KBps	1.1250	0.79%
9	921.6 KBps	NA	NA	NA	941.176 KBps	1.0625	2.12%

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 164. 采用 8 倍过采样时，在 f_{PCLK} = 8 MHz 或 f_{PCLK} = 16 MHz 下编程波特率时的误差计算(1)

8 倍过采样 (OVER8=1)							
波特率		f_{PCLK} = 8 MHz			f_{PCLK} = 16 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	2.4 KBps	2.400 KBps	416.625	0.01%	2.400 KBps	833.375	0.00%
2	9.6 KBps	9.604 KBps	104.125	0.04%	9.598 KBps	208.375	0.02%
3	19.2 KBps	19.185 KBps	52.125	0.08%	19.208 KBps	104.125	0.04%
4	57.6 KBps	57.557 KBps	17.375	0.08%	57.554 KBps	34.750	0.08%
5	115.2 KBps	115.942 KBps	8.625	0.64%	115.108 KBps	17.375	0.08%
6	230.4 KBps	228.571 KBps	4.375	0.79%	231.884 KBps	8.625	0.64%
7	460.8 KBps	470.588 KBps	2.125	2.12%	457.143 KBps	4.375	0.79%
8	896 KBps	888.889 KBps	1.125	0.79%	888.889 KBps	2.250	0.79%
9	921.6 KBps	888.889 KBps	1.125	3.55%	941.176 KBps	2.125	2.12%
10	1.792 MBps	NA	NA	NA	1.7777 MBps	1.125	0.79%
11	1.8432 MBps	NA	NA	NA	1.7777 MBps	1.125	3.55%

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 165. 采用 16 倍过采样时，在 f_{PCLK} = 30 MHz 或 f_{PCLK} = 60 MHz 下编程波特率时的误差计算(1)(2)

16 倍过采样时 (OVER8=0)							
波特率		f_{PCLK} = 30 MHz			f_{PCLK} = 60 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	2.4 KBps	2.400 KBps	781.2500	0.00%	2.400 KBps	1562.5000	0.00%
2	9.6 KBps	9.600 KBps	195.3125	0.00%	9.600 KBps	390.6250	0.00%
3	19.2 KBps	19.194 KBps	97.6875	0.03%	19.200 KBps	195.3125	0.00%
4	57.6 KBps	57.582KBps	32.5625	0.03%	57.582 KBps	65.1250	0.03%
5	115.2 KBps	115.385 KBps	16.2500	0.16%	115.163 KBps	32.5625	0.03%
6	230.4 KBps	230.769 KBps	8.1250	0.16%	230.769 KBps	16.2500	0.16%
7	460.8 KBps	461.538 KBps	4.0625	0.16%	461.538 KBps	8.1250	0.16%
8	896 KBps	909.091 KBps	2.0625	1.46%	895.522 KBps	4.1875	0.05%
9	921.6 KBps	909.091 KBps	2.0625	1.36%	923.077 KBps	4.0625	0.16%
10	1.792 MBps	1.1764 MBps	1.0625	1.52%	1.8182 MBps	2.0625	1.36%
11	1.8432 MBps	1.8750 MBps	1.0000	1.73%	1.8182 MBps	2.0625	1.52%

表 165. 采用 16 倍过采样时，在 f_{PCLK} = 30 MHz 或 f_{PCLK} = 60 MHz 下编程波特率时的误差计算(1)(2)（续）

16 倍过采样时 (OVER8=0)							
波特率		f_{PCLK} = 30 MHz			f_{PCLK} = 60 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
12	3.584 MBps	NA	NA	NA	3.2594 MBps	1.0625	1.52%
13	3.6864 MBps	NA	NA	NA	3.7500 MBps	1.0000	1.73%

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。
2. 仅 USART1 和 USART6 使用 PCLK2 计时。其他 USART 使用 PCLK1 计时。有关 PCLK1 和 PCLK2 的最大值，请参见器件数据手册。

表 166. 采用 8 倍过采样时，在 f_{PCLK} = 30 MHz 或 f_{PCLK} = 60 MHz 下编程波特率时的误差计算(1) (2)

8 倍过采样 (OVER8=1)							
波特率		f_{PCLK} = 30 MHz			f_{PCLK} = 60 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	2.4 KBps	2.400 KBps	1562.5000	0.00%	2.400 KBps	3125.0000	0.00%
2	9.6 KBps	9.600 KBps	390.6250	0.00%	9.600 KBps	781.2500	0.00%
3	19.2 KBps	19.194 KBps	195.3750	0.03%	19.200 KBps	390.6250	0.00%
4	57.6 KBps	57.582 KBps	65.1250	0.16%	57.582 KBps	130.2500	0.03%
5	115.2 KBps	115.385 KBps	32.5000	0.16%	115.163 KBps	65.1250	0.03%
6	230.4 KBps	230.769 KBps	16.2500	0.16%	230.769 KBps	32.5000	0.16%
7	460.8 KBps	461.538 KBps	8.1250	0.16%	461.538 KBps	16.2500	0.16%
8	896 KBps	909.091 KBps	4.1250	1.46%	895.522 KBps	8.3750	0.05%
9	921.6 KBps	909.091 KBps	4.1250	1.36%	923.077 KBps	8.1250	0.16%
10	1.792 MBps	1.7647 MBps	2.1250	1.52%	1.8182 MBps	4.1250	1.46%
11	1.8432 MBps	1.8750 MBps	2.0000	1.73%	1.8182 MBps	4.1250	1.36%
12	3.584 MBps	3.7500 MBps	1.0000	4.63%	3.5294 MBps	2.1250	1.52%
13	3.6864 MBps	3.7500 MBps	1.0000	1.73%	3.7500 MBps	2.0000	1.73%
14	7.168 MBps	NA	NA	NA	7.5000 MBps	1.0000	4.63%
15	7.3728 MBps	NA	NA	NA	7.5000 MBps	1.0000	1.73%

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。
2. 仅 USART1 和 USART6 使用 PCLK2 计时。其他 USART 使用 PCLK1 计时。有关 PCLK1 和 PCLK2 的最大值，请参见器件数据手册。

表 167. 采用 16 倍过采样时，在 f_{PCLK} = 42 MHz 或 f_{PCLK} = 84 Hz 下编程波特率时的误差计算(1)(2)

16 倍过采样时 (OVER8=0)							
波特率		f_{PCLK} = 42 MHz			f_{PCLK} = 84 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	1.2 KBps	1.2 KBps	2187.5	0	1.2 KBps	4375	0
2	2.4 KBps	2.4 KBps	1093.75	0	2.4 KBps	2187.5	0
3	9.6 KBps	9.6 KBps	273.4375	0	9.6 KBps	546.875	0
4	19.2 KBps	19.195 KBps	136.75	0.02	19.2 KBps	273.4375	0
5	38.4 KBps	38.391 KBps	68.375	0.02	38.391 KBps	136.75	0.02
6	57.6 KBps	57.613 KBps	45.5625	0.02	57.613 KBps	91.125	0.02
7	115.2 KBps	115.068 KBps	22.8125	0.11	115.226 KBps	45.5625	0.02
8	230.4 KBps	230.769 KBps	11.375	0.16	230.137 KBps	22.8125	0.11
9	460.8 KBps	461.538 KBps	5.6875	0.16	461.538 KBps	11.375	0.16
10	921.6 KBps	913.043 KBps	2.875	0.93	923.076 KBps	5.6875	0.93
11	1.792 MBps	1.826 MBps	1.4375	1.9	1.787 MBps	2.9375	0.27
12	1.8432 MBps	1.826 MBps	1.4375	0.93	1.826 MBps	2.875	0.93
13	3.584 MBps	NA	NA	NA	3.652 MBps	1.4375	1.9
14	3.6864 MBps	NA	NA	NA	3.652 MBps	1.4375	0.93

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。
2. 仅 USART1 和 USART6 使用 PCLK2 计时。其他 USART 使用 PCLK1 计时。有关 PCLK1 和 PCLK2 的最大值，请参见器件数据手册。

表 168. 采用 8 倍过采样时，在 f_{PCLK} = 42 MHz 或 f_{PCLK} = 84 MHz 下编程波特率时的误差计算(1)(2)

8 倍过采样 (OVER8=1)							
波特率		f_{PCLK} = 42 MHz			f_{PCLK} = 84 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
1	1.2 KBps	1.2 KBps	4375	0	1.2 KBps	8750	0
2	2.4 KBps	2.4 KBps	2187.5	0	2.4 KBps	4375	0
3	9.6 KBps	9.6 KBps	546.875	0	9.6 KBps	1093.75	0
4	19.2 KBps	19.195 KBps	273.5	0.02	19.2 KBps	546.875	0
5	38.4 KBps	38.391 KBps	136.75	0.02	38.391 KBps	273.5	0.02
6	57.6 KBps	57.613 KBps	91.125	0.02	57.613 KBps	182.25	0.02
7	115.2 KBps	115.068 KBps	45.625	0.11	115.226 KBps	91.125	0.02

表 168. 采用 8 倍过采样时，在 f_{PCLK} = 42 MHz 或 f_{PCLK} = 84 MHz 下编程波特率时的误差计算(1)(2)（续）

8 倍过采样 (OVER8=1)							
波特率		f_{PCLK} = 42 MHz			f_{PCLK} = 84 MHz		
序号	所需值	实际值	波特率寄存器中编程的值	误差 % = (计算值 - 所需值)/所需波特率	实际值	波特率寄存器中编程的值	误差 %
8	230.4 KBps	230.769 KBps	22.75	0.11	230.137 KBps	45.625	0.11
9	460.8 KBps	461.538 KBps	11.375	0.16	461.538 KBps	22.75	0.16
10	921.6 KBps	913.043 KBps	5.75	0.93	923.076 KBps	11.375	0.93
11	1.792 MBps	1.826 MBps	2.875	1.9	1.787Mbps	5.875	0.27
12	1.8432 MBps	1.826 MBps	2.875	0.93	1.826 MBps	5.75	0.93
13	3.584 MBps	3.5 MBps	1.5	2.34	3.652 MBps	2.875	1.9
14	3.6864 MBps	3.82 MBps	1.375	3.57	3.652 MBps	2.875	0.93
15	7.168 MBps	NA	NA	NA	7 MBps	1.5	2.34
16	7.3728 MBps	NA	NA	NA	7.636 MBps	1.375	3.57
18	9 MBps	NA	NA	NA	9.333 MBps	1.125	3.7
20	10.5 MBps	NA	NA	NA	10.5 MBps	1	0

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。
2. 仅 USART1 和 USART6 使用 PCLK2 计时。其他 USART 使用 PCLK1 计时。有关 PCLK1 和 PCLK2 的最大值，请参见器件数据手册。

28.4.5 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

对于正常接收数据，USART 接收器的容差等于所容许的最大偏差，具体取决于以下选择：

- 由 USART_CR1 寄存器中的 M 位定义的 10 位或 11 位字符长度
- 由 USART_CR1 寄存器中的 OVER8 位定义的 8 倍或 16 倍过采样
- 是否使用小数波特率
- 使用 1 位或 3 位对数据进行采样，取决于 USART_CR3 寄存器中 ONEBIT 位的值

表 169. DIV_Fraction 为 0 时的 USART 接收器容差

M 位	OVER8 位 = 0		OVER8 位 = 1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
0	3.75%	4.375%	2.50%	3.75%
1	3.41%	3.97%	2.27%	3.41%

表 170. DIV_Fraction 不为 0 时的 USART 接收器容差

M 位	OVER8 位 = 0		OVER8 位 = 1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
0	3.33%	3.88%	2%	3%
1	3.03%	3.53%	1.82%	2.73%

注：当接收的帧包含 10 个位时间的空闲帧（M=0）或 11 个位时间的空闲帧（M=1）时，表 169 和表 170 中指定的数字可能与特例中的数字略微不同。

28.4.6 多处理器通信

可以与 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 输出与其他 USART 的 RX 输入相连；而其他 USART 为从 USART，其各自的 TX 输出在逻辑上通过与运算连在一起，并与主 USART 的 RX 输入相连。

在多处理器配置中，理想情况下通常只有预期的消息接收方主动接收完整的消息内容，从而减少由所有未被寻址的接收器造成的冗余 USART 服务开销。

可通过静默功能将未被寻址的器件置于静默模式下。在静默模式下：

- 不得将接收状态位置 1。
- 禁止任何接收中断。
- USART_CR1 寄存器中的 RWU 位置 1。RWU 可由硬件自动控制，或在特定条件下由软件写入。

根据 USART_CR1 寄存器中 WAKE 位的设置，USART 可使用以下两种方法进入或退出静音模式：

- 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

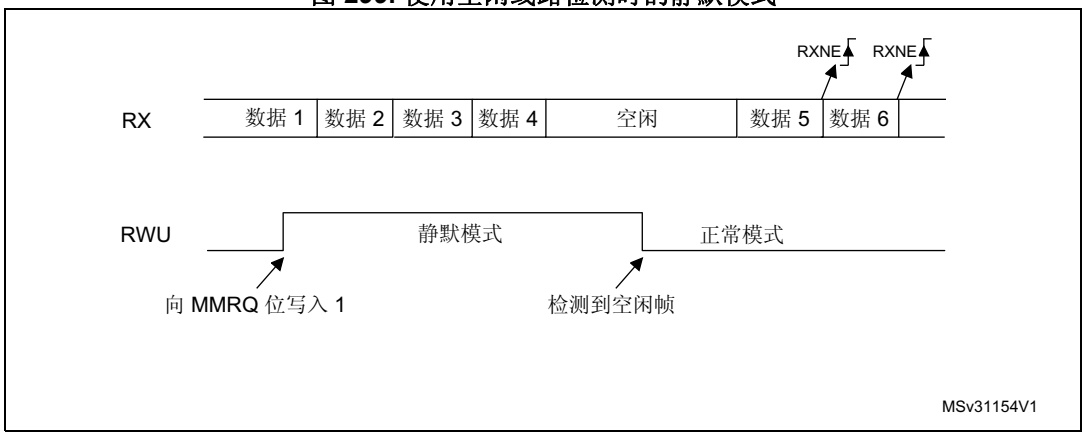
空闲线路检测 (WAKE=0)

当向 RWU 位写入 1 时，USART 进入静音模式。

当检测到空闲帧时，它会被唤醒。此时 RWU 位会由硬件清零，但 USART_SR 寄存器中的 IDLE 位不会置 1。还可通过软件向 RWU 位写入 0。

图 299 中给出了使用空闲线路检测时静默模式行为的示例。

图 299. 使用空闲线路检测时的静默模式



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 LSB 上。接收器会将此 4 位字与其地址进行比较，该接收器的地址在 USART_CR2 寄存器的 ADD 位中进行设置。

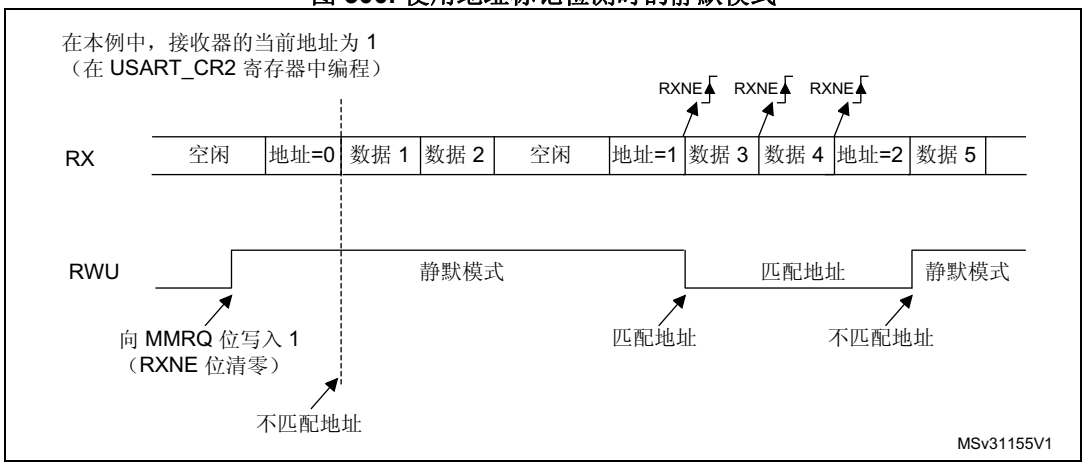
当接收到与其编程地址不匹配的地址字符时，USART 会进入静默模式。此时，RWU 位将由硬件置 1。由于当时 USART 已经进入了静音模式，所以 RXNE 标志不会针对此地址字节置 1，也不会发出中断或 DMA 请求。

当接收到与编程地址匹配的地址字符时，它会退出静音模式。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。

当接收器的缓冲区不包含任何数据（USART_SR 寄存器中 RXNE=0）时，可向 RWU 位写入 0 或 1。否则会忽略写尝试。

图 300 中给出了使用地址标记检测时静默模式行为的示例。

图 300. 使用地址标记检测时的静默模式



28.4.7 奇偶校验控制

将 USART_CR1 寄存器中的 PCE 位置 1，可以使能奇偶校验控制（发送时生成奇偶校验位，接收时进行奇偶校验检查）。根据 M 位定义的帧长度，*表 171* 中列出了可能的 USART 帧格式。

表 171. 帧格式

M 位	PCE 位	USART 帧(1)
0	0	| SB | 8 位数据 | STB |
0	1	| SB | 7 位数据 | PB | STB |
1	0	| SB | 9 位数据 | STB |
1	1	| SB | 8 位数据 PB | STB |

1. 图注：SB：起始位，STB：停止位，PB：奇偶校验位。

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 7 个或 8 个 LSB 位组成，具体取决于 M 等于 0 还是 1）。

例如：数据=00110101；4 个位置 1 => 如果选择偶校验（USART_CR1 寄存器中的 PS 位 = 0），则校验位是 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 7 个或 8 个 LSB 位组成，具体取决于 M 等于 0 还是 1）。

例如：数据=00110101；4 个位置 1 => 如果选择奇校验（USART_CR1 寄存器中的 PS 位 = 1），则校验位是 1。

接收时进行奇偶校验检查

如果奇偶校验检查失败，则 USART_SR 寄存器中的 PE 标志置 1；如果 USART_CR1 寄存器中 PEIE 位置 1，则会生成中断。PE 标志由软件序列清零（从状态寄存器中读取，然后对 USART_DR 数据寄存器执行读或写访问）。

注： *如果被地址标记唤醒：会使用数据的 MSB 位而非奇偶校验位来识别地址。此外，接收器不会对地址数据进行奇偶校验检查（奇偶校验出错时，PE 不置 1）。*

发送时的奇偶校验生成

如果 USART_CR1 寄存器中的 PCE 位置 1，则在数据寄存器中所写入数据的 MSB 位会进行传送，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）。

注： *用于管理发送过程的软件程序可以激活软件序列，进而将 PE 标志清零（从状态寄存器中读取，然后对数据寄存器执行读或写访问）。在半双工模式下工作时（具体取决于软件），这可能会导致 PE 标志意外清零。*

28.4.8 LIN（局域互连网络）模式

通过将 USART_CR2 寄存器中的 LINEN 位置 1 来选择 LIN 模式。在 LIN 模式下，必须将以下位清零：

- USART_CR2 寄存器中的 STOP[1:0] 和 CLKEN 位
- USART_CR3 寄存器中的 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 28.4.2 节*中介绍的相同步骤，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 SBK 位置 1 会发送 13 个“0”位作为断路字符。然后会发送值为“1”的位以进行下一启动检测。

LIN 接收

断路检测电路在 USART 接口上实现。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（USART_CR1 寄存器中 RE=1）使能后，电路便开始监测启动信号的 RX 输入。检测起始位的方法与搜索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（USART_CR2 中 LBDL = 0 时）或 11 个（USART_CR2 中 LBDL=1 时）连续位均检测为“0”，且其后跟随分隔符，则 USART_SR 寄存器中的 LBD 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。

图 301 中显示了断路检测器状态机和断路标志的行为。

在*图 302* 给出的断路帧示例中，假设 LBDL=1（11 位断路长度），M=0（8 位数据）。

图 301. LIN 模式下的断路检测（11 位断路长度——LBDL 位置 1）

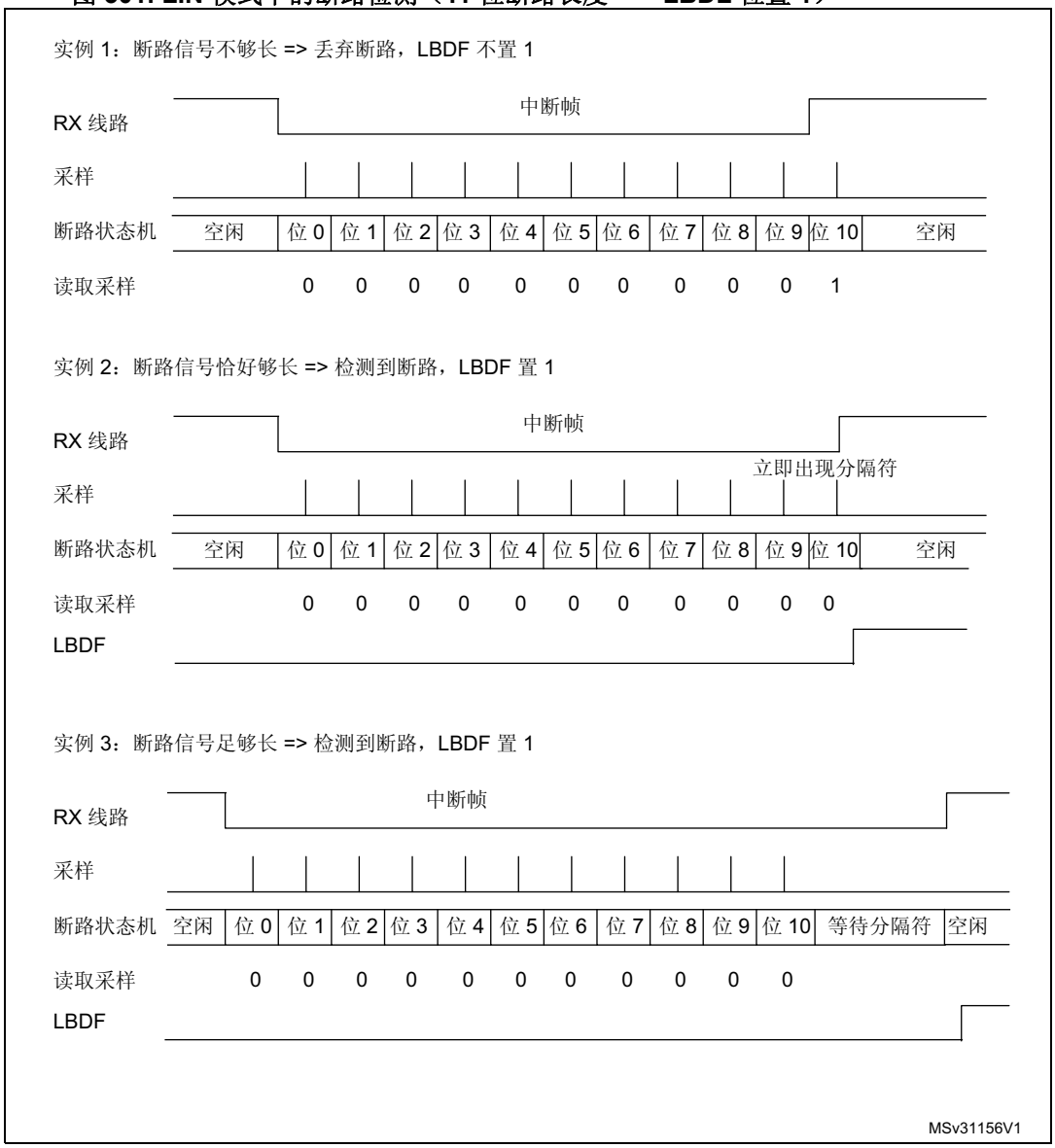
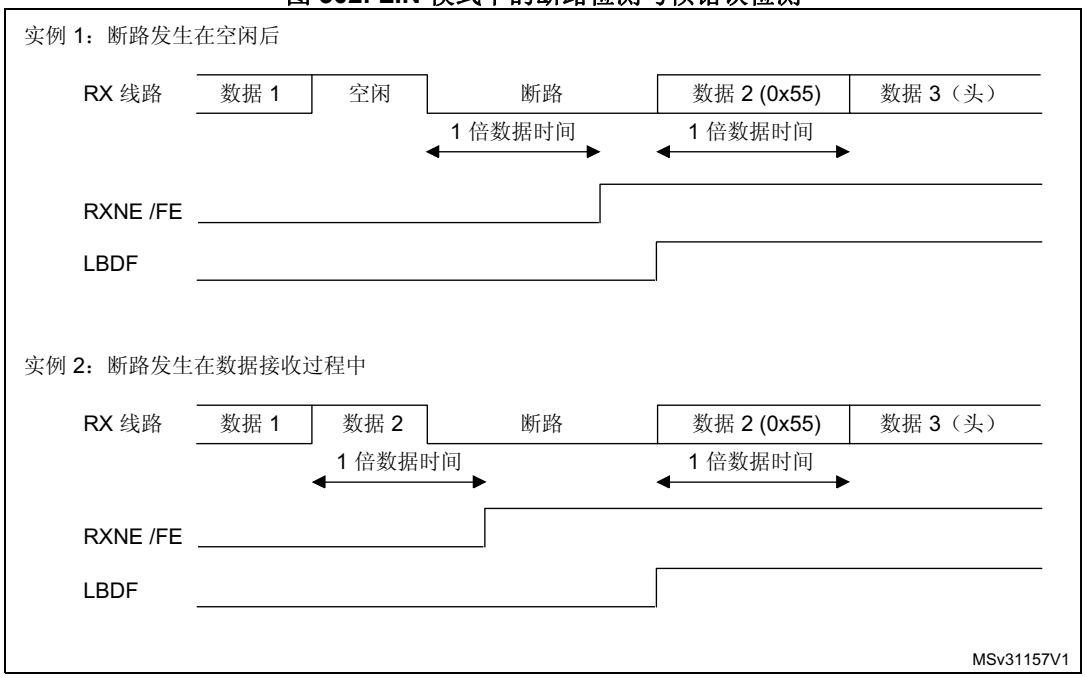


图 302. LIN 模式下的断路检测与帧错误检测



28.4.9 USART 同步模式

通过将 USART_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：

- USART_CR2 寄存器中的 LINEN 位，
- USART_CR3 寄存器中的 SCEN、HDSEL 和 IREN 位。

通过 USART，用户可以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，将会（也可能不会）生成时钟脉冲，这取决于 USART_CR2 寄存器中 LBCL 位的状态。通过 USART_CR2 寄存器中的 CPOL 位，用户可以选择时钟极性；通过 USART_CR2 寄存器中的 CPHA 位，用户可以选择外部时钟相位（参见*图 303*、*图 304* 和*图 305*）。

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（对数据寄存器 USART_DR 已被写入），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。*

当发送器和接收器 (TE=RE=0) 都被禁止时，必须选择 LBCL、CPOL 和 CPHA 位，以确保时钟脉冲正常工作。当使能发送器或接收器时，不得对这些位进行更改。

建议按照相同指令将 TE 和 RE 位置 1，以尽量缩短接收器的建立时间和保持时间。

USART 只支持主模式：它不能接收或发送与输入时钟相关的数据（SCLK 始终为输出）。

图 303. USART 同步发送示例

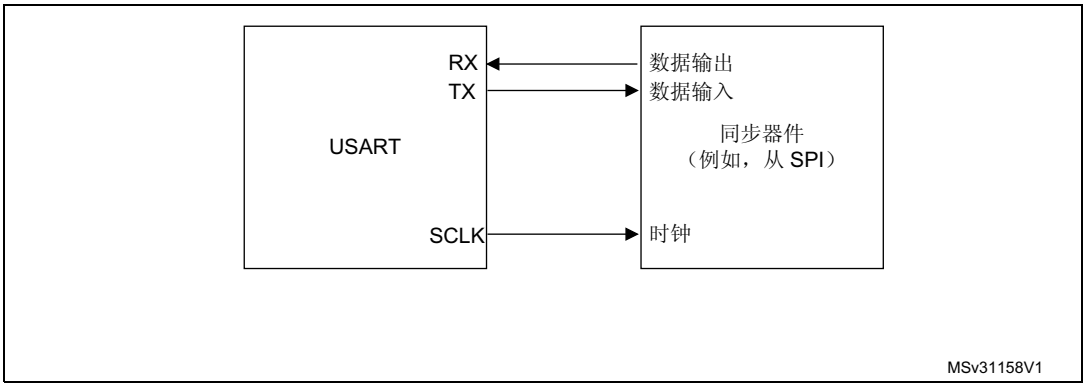


图 304. USART 数据时钟时序图 (M=0)

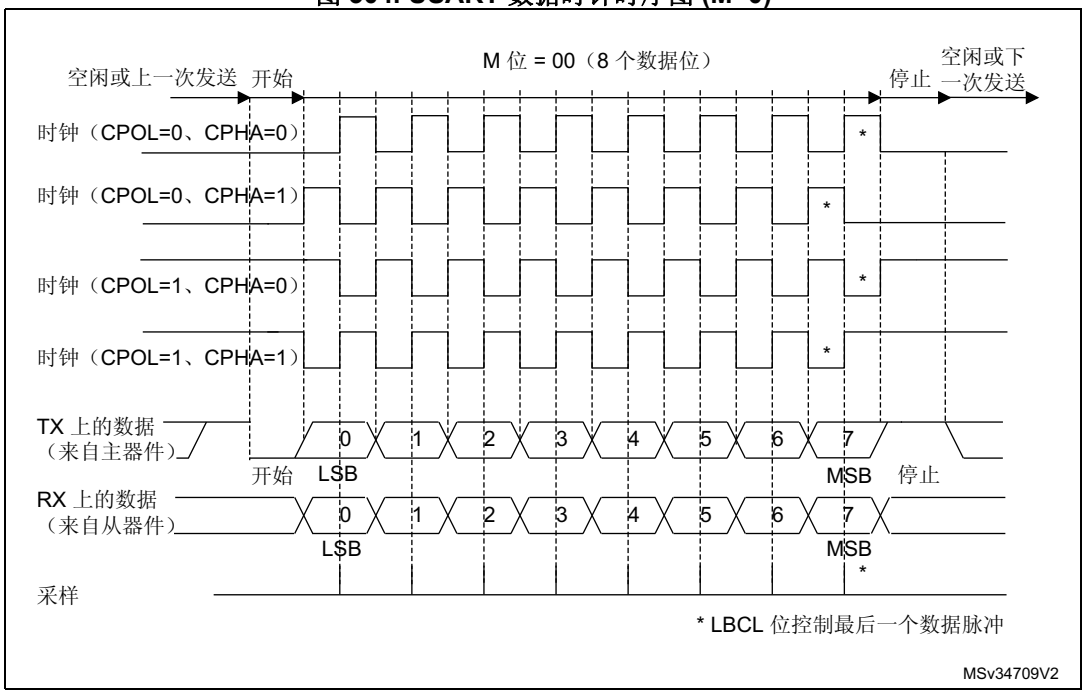


图 305. USART 数据时钟时序图 (M=1)

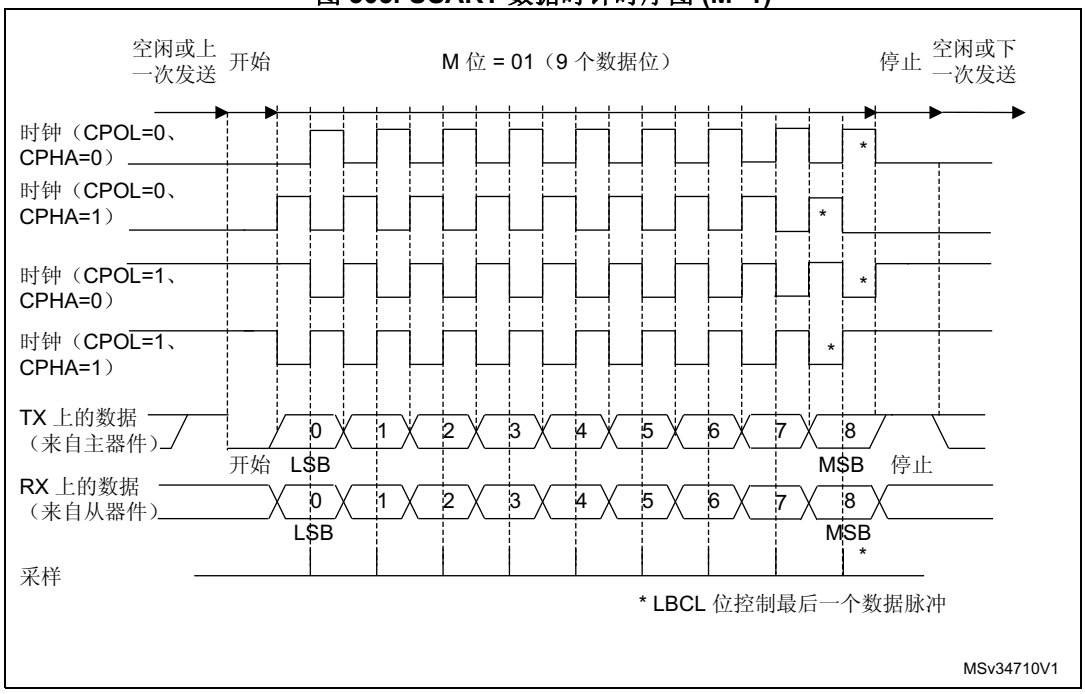
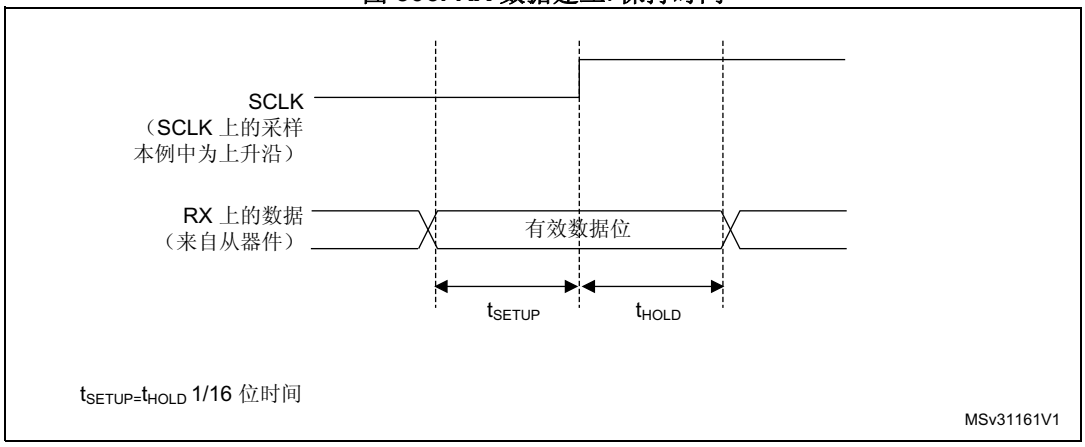


图 306. RX 数据建立/保持时间



注： *在智能卡模式下，SCLK 的功能有所不同。有关详细信息，请参见智能卡模式一章。*

28.4.10 单线半双工通信

通过将 USART_CR3 寄存器中的 HDSEL 位置 1 来选择单线半双工模式。在此模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN 和 CLKEN 位，
- USART_CR3 寄存器中的 SCEN 和 IREN 位。

USART 可以配置为遵循单线半双工协议，其中 TX 和 RX 线路从内部相连接。使用控制位“HALF DUPLEX SEL”（USART_CR3 寄存器中的 HDSEL 位），可以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，以便在未受 USART 驱动时，使 TX 成为浮空输入（或高电平开漏输出）。

除此之外，通信与正常 USART 模式下的通信相似。此线路上的冲突必须由软件进行管理（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。

28.4.11 智能卡

通过将 USART_CR3 寄存器中的 SCEN 位置 1 来选择智能卡模式。在智能卡模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN 位
- USART_CR3 寄存器中的 HDSEL 和 IREN 位

此外，可能需要将 CLKEN 位置 1，以便为智能卡提供时钟。

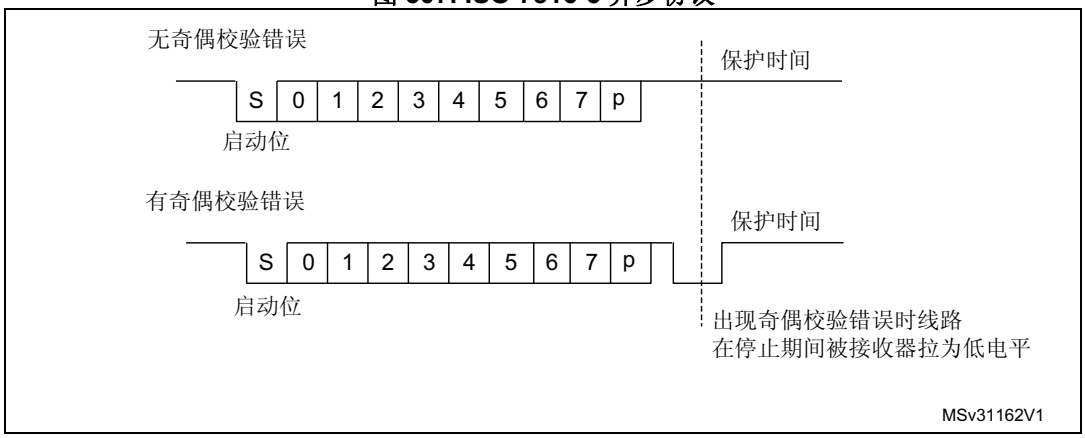
智能卡接口支持符合 ISO 7816-3 标准的异步协议智能卡。USART 应如下所示进行配置：

- 8 个位加奇偶校验：当 USART_CR1 寄存器中 M=1 且 PCE=1 时
- 发送和接收时使用 1.5 个停止位：当 USART_CR2 寄存器中 STOP=11 时

注： *接收时也可以选择 0.5 个停止位，但为了避免在两种配置之间切换，建议发送和接收时均使用 1.5 个停止位。*

图 307 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 307. 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。

智能卡是一个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。

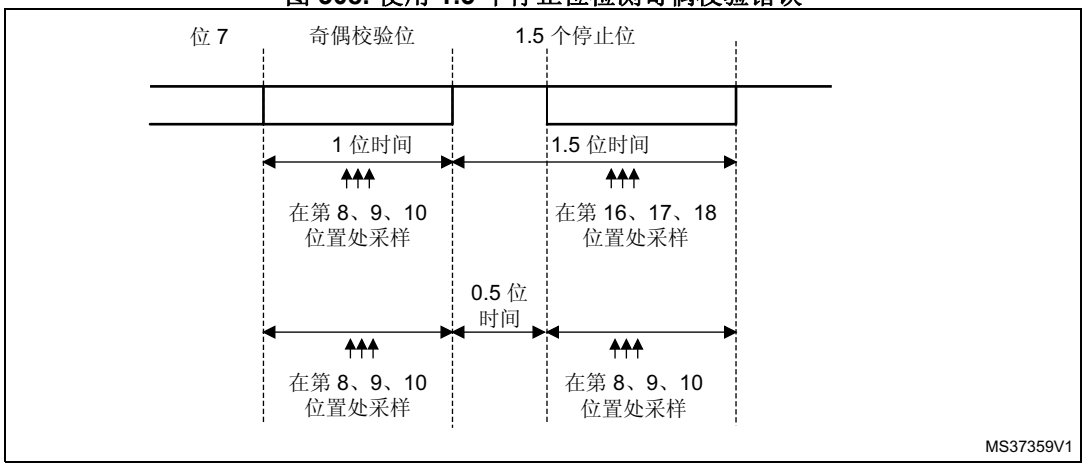
- 如果在接收一个使用 0.5 或 1.5 个停止位编程的帧期间检测到奇偶校验错误，则在完成接收帧后，发送线会被拉低一个时钟周期。这是为了向智能卡指出发送到 USART 的数据尚未正确接收。此 NACK 信号（将发送线拉低 1 个时钟周期）会导致发送器端（配置为 1.5 个停止位）出现帧错误。应用程序可根据协议重新发送数据。如果 NACK 控制位置 1，则接收器会向奇偶校验错误发送“NACK”信号；否则不会发送 NACK 信号。
- 通过对保护时间寄存器进行编程，可以延迟 TC 标志的置位。正常工作时，当发送移位寄存器为空且没有新的发送请求出现时，会对 TC 标志进行置位。在智能卡模式下，空的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 数据将被视为数据，而非中断。*

当翻转 TE 位时，不会发送空闲帧。空闲帧（在其他配置中进行了定义）在 ISO 协议中未进行定义。

图 308 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 308. 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 USART_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。

28.4.12 IrDA SIR ENDEC 模块

通过将 USART_CR3 寄存器中的 IREN 位置 1 来选择 IrDA 模式。在 IrDA 模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN、STOP 和 CLKEN 位，
- USART_CR3 寄存器中的 SCEN 和 HDSEL 位。

IrDA SIR 物理层规定使用反相归零 (RZI) 调制方案，它以红外光脉冲表示逻辑 0（参见*图 309*）。

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（例如，USART 正在向 IrDA 编码器发送数据），则 IrDA 解码器会忽略 IrDA 接收线上的所有数据；如果接收器忙（例如，USART 正在接收来自 USART 的解码数据），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 310*）。
- SIR 解码器用于将兼容 IrDA 的接收信号转换为 USART 的位流。
- SIR 接收逻辑将高电平状态视为逻辑“1”，将低电平脉冲视为逻辑“0”。
- 发送编码器输出的极性与解码器输入相反。SIR 输出在空闲时处于低电平状态。
- IrDA 规范要求脉冲容忍值要大于 1.41 us。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 IrDA 低功耗波特寄存器 USART_GTPR 中编程的预分频器值）。宽度小于 1 个 PSC 周期的脉冲都将被拒绝，但宽度大于 1 个而小于 2 个周期的脉冲可能被接受也可能被拒绝，而宽度大于 2 个周期的脉冲将被接受作为有效脉冲。当 PSC=0 时，IrDA 编码器/解码器不工作。
- 接收器能够与低功耗发送器进行通信。
- 在 IrDA 模式下，USART_CR2 寄存器中的 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/PSC 的脉冲。只有当持续时间长于 2 个 IrDA 低功耗波特时钟周期（USART_GTPR 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 309. IrDA SIR ENDEC——框图

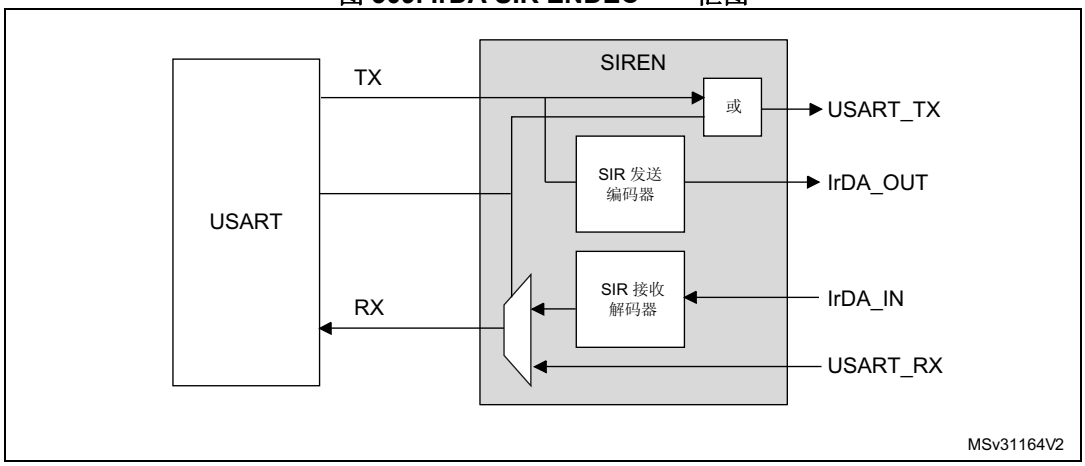
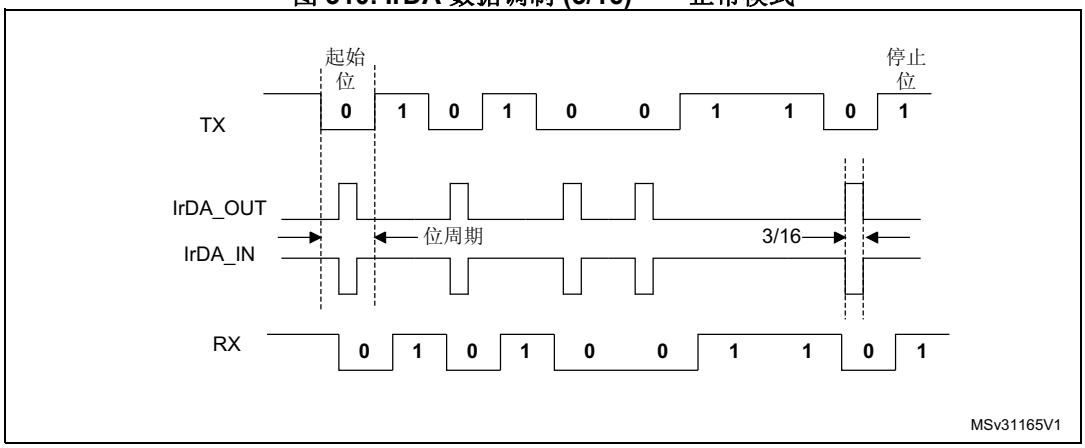


图 310. IrDA 数据调制 (3/16)——正常模式



28.4.13 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。

使用 DMA 进行发送

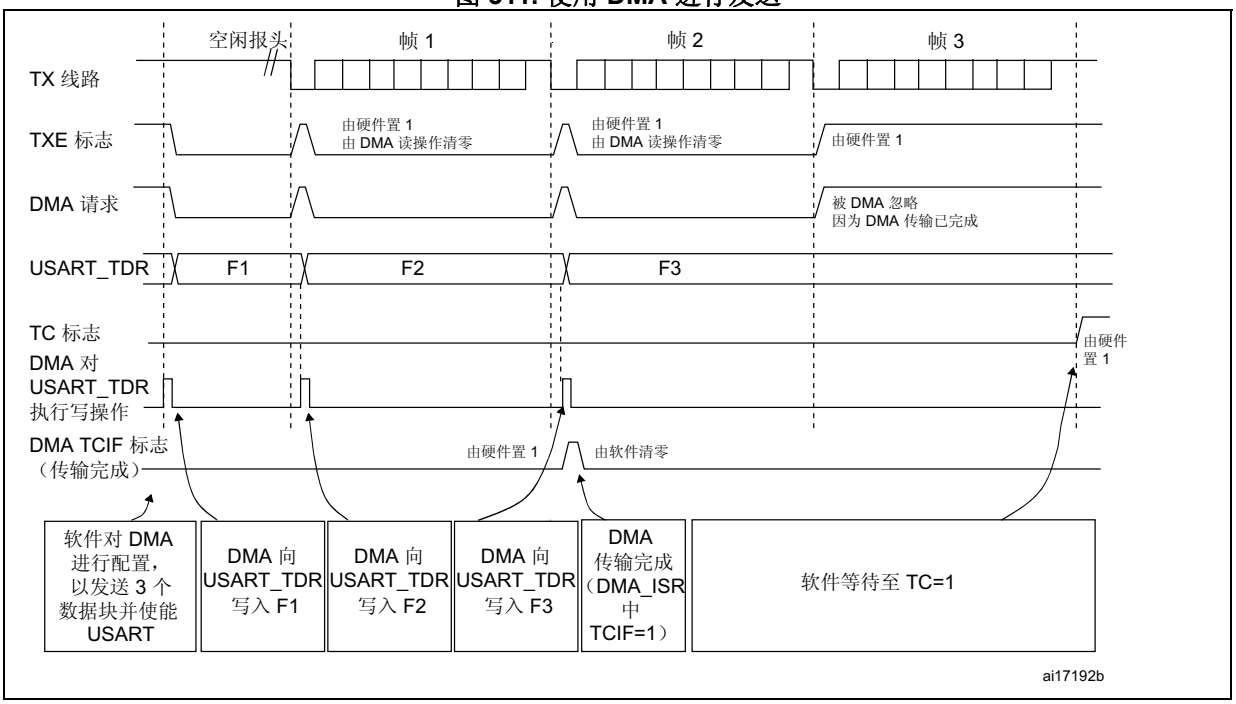
将 USART_CR3 寄存器中的 DMAT 位置 1 可以使能 DMA 模式进行发送。当 TXE 位置 1 时，可将数据从 SRAM 区（通过 DMA 配置，参见 DMA 部分）加载到 USART_DR 寄存器。要映射一个 DMA 通道以进行 USART 发送，请按以下步骤操作（x 表示通道编号）：

1. 在 DMA 控制寄存器中写入 USART_DR 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，数据都会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，数据都会从这个存储区域加载到 USART_DR 寄存器中。
3. 在 DMA 控制寄存器中配置要传输的总字节数。
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. 向 SR 寄存器中的 TC 位写入 0，将其清零。
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必须保持清零状态，然后在最后一帧发送结束后由硬件置 1。

图 311. 使用 DMA 进行发送



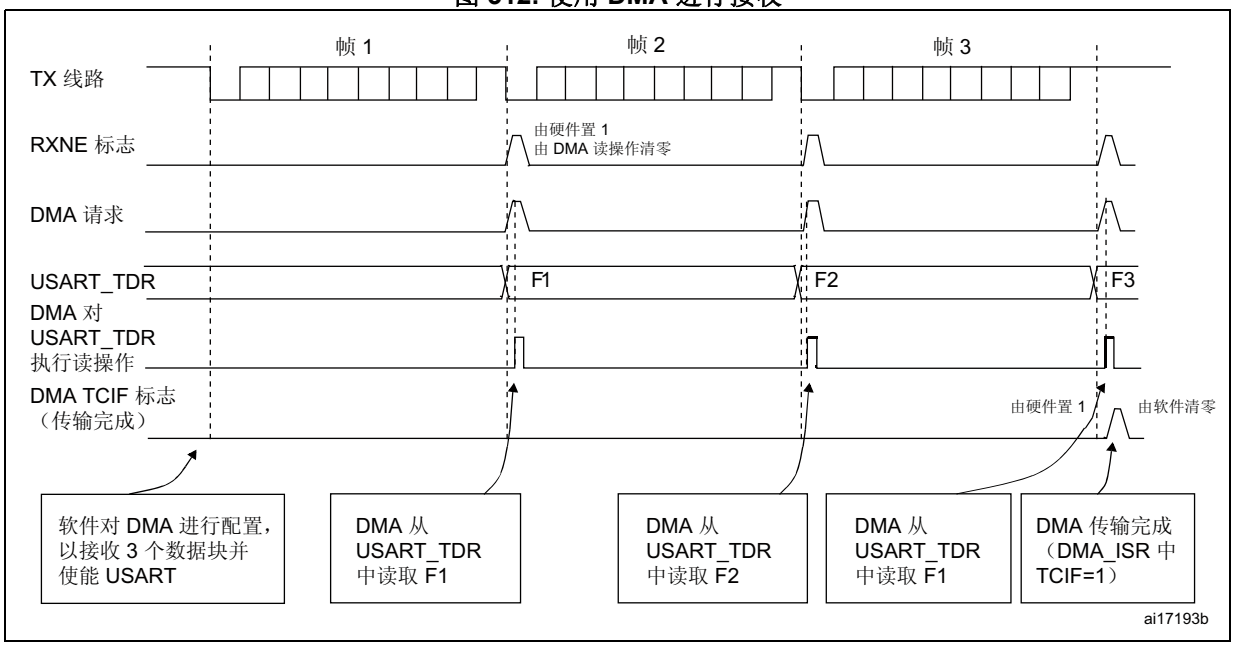
使用 DMA 进行接收

将 USART_CR3 寄存器中的 DMAR 位置 1 可以使能 DMA 模式进行接收。接收数据字节时，数据会从 USART_DR 寄存器加载到 SRAM 区域中（通过 DMA 配置，参见 DMA 规范）。要映射一个 DMA 通道以进行 USART 接收，请按以下步骤操作：

1. 在 DMA 控制寄存器中写入 USART_DR 寄存器地址，将其配置为传输的源地址。每次发生 RXNE 事件后，数据都会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，数据都会从 USART_DR 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。在中断子程序中，USART_CR3 寄存器中的 DMAR 位应由软件清零。

图 312. 使用 DMA 进行接收



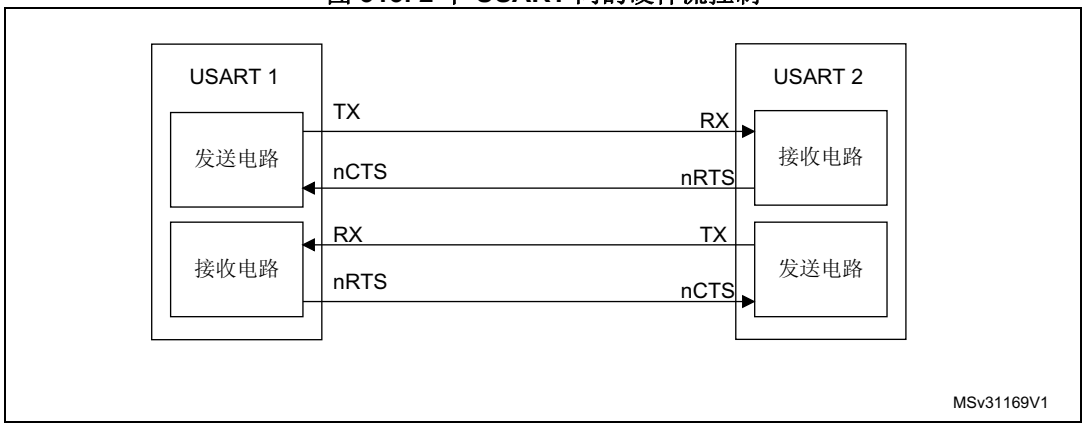
多缓冲区通信中的错误标志和中断生成

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（USART_CR3 寄存器中的 EIE 位）；如果该位置 1，则会因其中任何一个错误而在当前字节后产生中断。

28.4.14 硬件流控制

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 313* 显示了在这种模式下如何连接 2 个器件：

图 313. 2 个 USART 间的硬件流控制

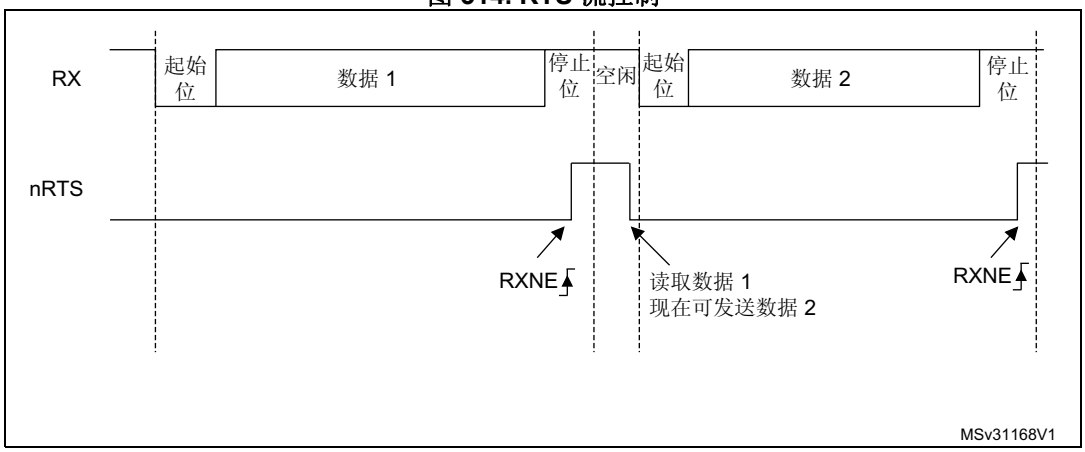


分别向 USART_CR3 寄存器中的 RTSE 位和 CTSE 位写入 1，可以分别使能 RTS 和 CTS 流控制。

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 314* 显示了在使能 RTS 流控制的情况下进行通信的示例。

图 314. RTS 流控制

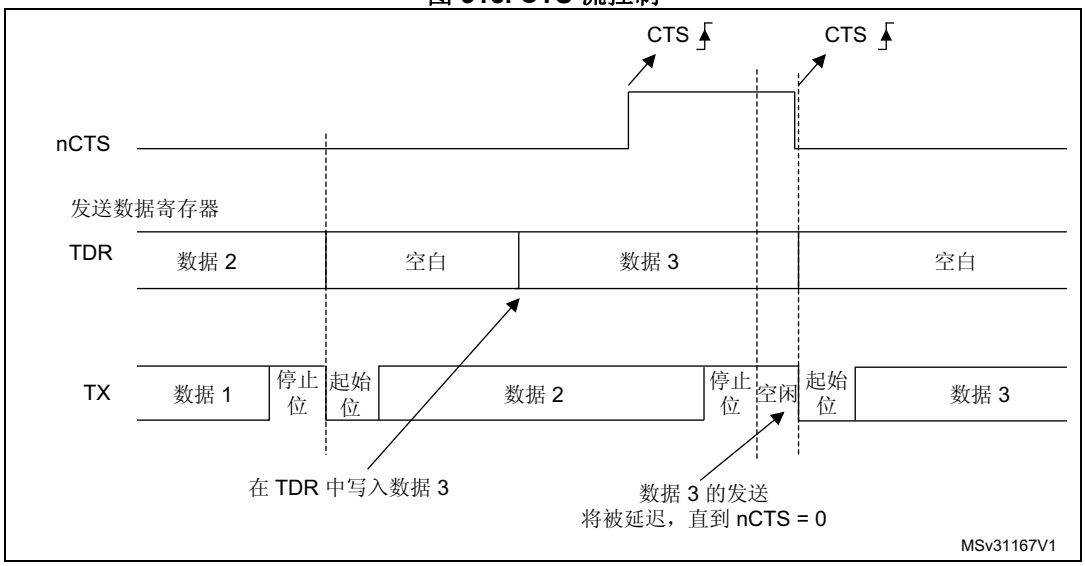


CTS 流控制

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 USART_CR3 寄存器中的 CTSIE 位置 1，则会产生中断。下图显示了在使能 CTS 流控制的情况下进行通信的示例。

图 315. CTS 流控制



注： ***停止帧的特殊行为：****使能 CTS 流后，发送器发送停止信号时将不检查 nCTS 输入状态。*

28.5 USART 中断

表 172. USART 中断请求

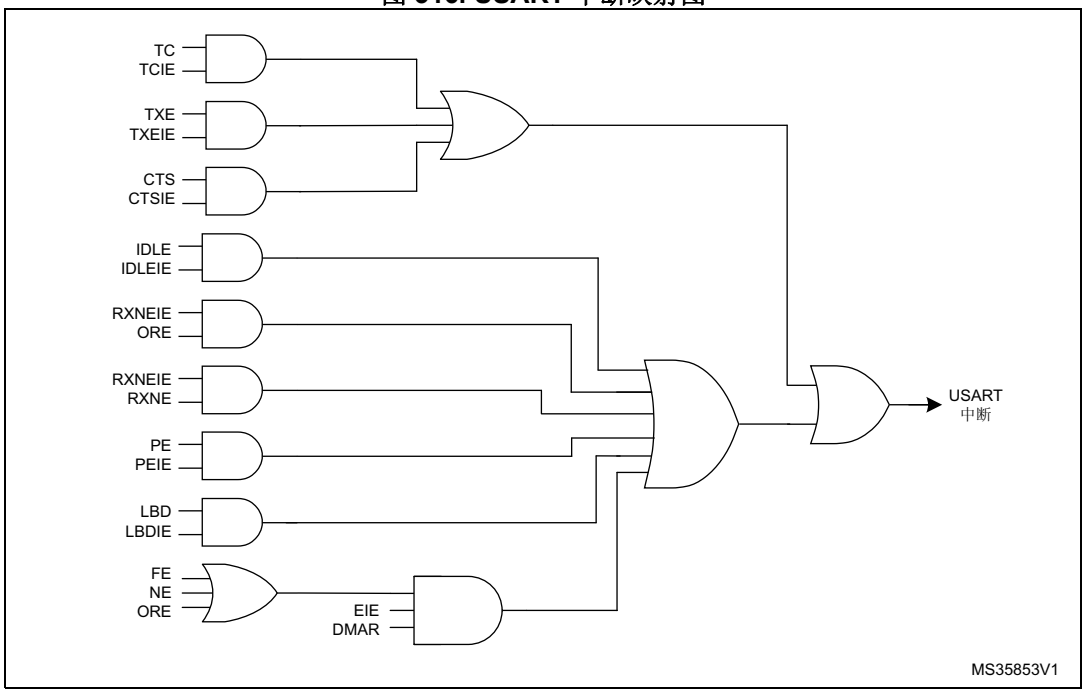
中断事件	事件标志	使能控制位
发送数据寄存器为空	TXE	TXEIE
CTS 标志	CTS	CTSIE
发送完成	TC	TCIE
准备好读取接收到的数据	RXNE	RXNEIE
检测到上溢错误	ORE	
检测到空闲线路	IDLE	IDLEIE
奇偶校验错误	PE	PEIE
断路标志	LBD	LBDIE
多缓冲区通信中的噪声标志、溢出错误和帧错误	NF 或 ORE 或 FE	EIE

USART 中断事件被连接到相同的中断向量（请参见*图 316*）。

- 发送期间：发送完成、清除以发送或发送数据寄存器为空中断。
- 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（仅限多缓冲区通信）和帧错误（仅限多缓冲区通信）

如果相应的使能控制位置 1，则这些事件会生成中断。

图 316. USART 中断映射图



28.6 USART 寄存器

有关寄存器说明中使用的缩写，请参见*第 50 页的第 1.2 节*。

必须按半字（16 位）或字（32 位）访问外设寄存器。

28.6.1 状态寄存器 (USART_SR)

Status register

偏移地址：0x00

复位值：0x00C0 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
						rc_w0	rc_w0	r	rc_w0	rc_w0	r	r	r	r	r

位 31:10 保留，必须保持复位值

位 9 **CTS**：CTS 标志 (CTS flag)

如果 CTSE 位置 1，当 nCTS 输入切换时，此位由硬件置 1。通过软件将该位清零（通过向该位中写入 0）。如果 USART_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： 该位不适用于 UART4 和 UART5。

位 8 **LBD**：LIN 断路检测标志 (LIN break detection flag)

检测到 LIN 断路时，此位由硬件置 1。通过软件将该位清零（通过向该位中写入 0）。如果 USART_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 LBDIE=1，则当 LBD=1 时生成中断

位 7 **TXE**：发送数据寄存器为空 (Transmit data register empty)

当 TDR 寄存器的内容已传输到移位寄存器时，该位由硬件置 1。如果 USART_CR1 寄存器中 TXEIE 位 = 1，则会生成中断。通过对 USART_DR 寄存器执行写入操作将该位清零。

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，则此位由硬件置 1。如果 USART_CR1 寄存器中 TCIE = 1，则会生成中断。该位由软件序列清零（读取 USART_SR 寄存器，然后写入 USART_DR 寄存器）。TC 位也可以通过向该位写入‘0’来清零。建议仅在多缓冲区通信时使用此清零序列。

0：传送未完成

1：传送已完成

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 USART_DR 寄存器时，该位由硬件置 1。如果 USART_CR1 寄存器中 RXNEIE = 1，则会生成中断。通过对 USART_DR 寄存器执行读入操作将该位清零。RXNE 标志也可以通过向该位写入零来清零。建议仅在多缓冲区通信时使用此清零序列。

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，此位由硬件置 1。如果 USART_CR1 寄存器中 IDLEIE = 1，则会生成中断。该位由软件序列清零（读入 USART_SR 寄存器，然后读入 USART_DR 寄存器）。

0：未检测到空闲线路

1：检测到空闲线路

注： 直到 RXNE 位本身已置 1 时（即，当出现新的空闲线路时）IDLE 位才会被再次置 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字准备好传输到 RDR 寄存器时，该位由硬件置 1。如果 USART_CR1 寄存器中 RXNEIE = 1，则会生成中断。该位由软件序列清零（读入 USART_SR 寄存器，然后读入 USART_DR 寄存器）。

0：无上溢错误

1：检测到溢出错误

注： *当该位置 1 时，RDR 寄存器的内容不会丢失，但移位寄存器会被覆盖。如果 EIE 位置 1，则在进行多缓冲区通信时会对 ORE 标志生成一个中断。*

位 2 **NF**：检测到噪声标志 (Noise detected flag)

当在接收的帧上检测到噪声时，此位由硬件置 1。该位由软件序列清零（读入 USART_SR 寄存器，然后读入 USART_DR 寄存器）。

0：未检测到噪声

1：检测到噪声

注： *如果 EIE 位置 1，则在进行多缓冲区通信时，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 位因 NF 标志而生成的时间相同。*

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 852 页的第 28.4.5 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，此位由硬件置 1。该位由软件序列清零（读入 USART_SR 寄存器，然后读入 USART_DR 寄存器）。

0：未检测到帧错误

1：检测到帧错误或中断字符

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。如果当前正在传输的字同时导致帧错误和上溢错误，则会传输该字，且仅有 ORE 位被置 1。*

如果 EIE 位置 1，则在进行多缓冲区通信时会对 FE 标志生成一个中断。

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，此位由硬件置 1。该位由软件序列清零（读取状态寄存器，然后对 USART_DR 数据寄存器执行读或写访问）。将 PE 位清零前软件必须等待 RXNE 标志被置 1。

如果 USART_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

28.6.2 数据寄存器 (USART_DR)

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.	DR[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 **DR[8:0]**：数据值 (Data value)

包含接收到数据字符或已发送的数据字符，具体取决于所执行的操作是“读取”操作还是“写入”操作。

因为数据寄存器包含两个寄存器，一个用于发送 (TDR)，一个用于接收 (RDR)，因此它具有双重功能（读和写）。

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（参见图 1）。

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。

在使能奇偶校验位的情况下（USART_CR1 寄存器中的 PCE 位被置 1）进行发送时，由于 MSB 的写入值（位 7 或位 8，具体取决于数据长度）会被奇偶校验位所取代，因此该值不起任何作用。

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。

28.6.3 波特率寄存器 (USART_BRR)

Baud rate register

注： *如果 TE 或 RE 位分别被禁止，则波特计数器会停止计数。*

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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV_Mantissa[11:0]												DIV_Fraction[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 15:4 **DIV_Mantissa[11:0]**：USARTDIV 的尾数 (mantissa of USARTDIV)

这 12 个位用于定义 USART 除数 (USARTDIV) 的尾数

位 3:0 **DIV_Fraction[3:0]**：USARTDIV 的小数 (fraction of USARTDIV)

这 4 个位用于定义 USART 除数 (USARTDIV) 的小数。当 OVER8 = 1 时，不考虑 DIV_Fraction3 位，且必须将该位保持清零。

28.6.4 控制寄存器 1 (USART_CR1)

Control 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVER8	Res.	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值

位 15 **OVER8**：过采样模式 (Oversampling mode)

0：16 倍过采样

1：8 倍过采样

注： 8 倍过采样在智能卡、IrDA 和 LIN 模式下不可用：当 SCEN=1、IREN=1 或 LINEN=1 时，OVER8 由硬件强制清零。

位 14 保留，必须保持复位值

位 13 **UE**：USART 使能 (USART enable)

该位清零后，USART 预分频器和输出将停止，并会结束当前字节传输以降低功耗。此位由软件置 1 和清零。

0：禁止 USART 预分频器和输出

1：使能 USART

位 12 **M**：字长 (Word length)

该位决定了字长。此位由软件置 1 或清零。

0：1 个起始位，8 个数据位，n 个停止位

1：1 个起始位，9 个数据位，n 个停止位

注： 在数据传输（发送和接收）期间不得更改 M 位

位 11 **WAKE**：唤醒方法 (Wakeup method)

该位决定了 USART 唤醒方法，该位由软件置 1 或清零。

0：空闲线路

1：地址标记

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

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。此位由软件置 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

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 USART_SR 寄存器中的 PE=1 时，生成 USART 中断

位 7 **TXEIE**：TXE 中断使能 (TXE interrupt enable)

此位由软件置 1 和清零。

0：禁止中断

1：当 USART_SR 寄存器中的 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 USART_SR 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 USART_SR 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 USART_SR 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。此位由软件置 1 和清零。

0：禁止发送器

1：使能发送器

注： *1：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）。*

2：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。此位由软件置 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 **RWU**：接收器唤醒 (Receiver wakeup)

该位决定 USART 是否处于静音模式。该位由软件置 1 和清零，并可在识别出唤醒序列时由硬件清零。

0：接收器处于活动模式

1：接收器处于静默模式

注： *1：选择静音模式前（通过将 RWU 位置 1），USART 必须首先接收一个数据字节，否则当由空闲线路检测到唤醒时，它无法于静音模式下正常工作。*

2：在地址标记检测唤醒配置（WAKE 位 = 1）中，RXNE 位置 1 时，RWU 位不能由软件进行修改。

位 0 **SBK**：发送断路 (Send break)

该位用于发送断路字符。该位可由软件置 1 和清零。该位应由软件置 1，并在断路停止位期间由硬件重置。

0：不发送断路字符

1：将发送断路字符

28.6.5 控制寄存器 2 (USART_CR2)

Control register 2

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	LINEN	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	Res.	LBDIE	LBDL	Res.	ADD[3:0]			
	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 **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 USART_CR1 寄存器中的 SBK 位发送 LIN 同步断路（13 个低位），并可检测 LIN 同步断路。

位 13:12 **STOP**：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

注： 0.5 个停止位和 1.5 个停止位不适用于 UART4 和 UART5。

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

该位不适用于 UART4 和 UART5。

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。

该位不适用于 UART4 和 UART5。

位 9 **CPHA**：时钟相位 (Clock phase)

该位允许用户在同步模式下选择 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 *304* 至 *305*）

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

注： 该位不适用于 UART4 和 UART5。

位 8 **LBCL**：最后一个位时钟脉冲 (Last bit clock pulse)

该位允许用户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

注： 1：最后一位为发送的第 8 或第 9 个数据位，具体取决于 USART_CR1 寄存器中 M 位所选择的 8 位或 9 位格式。

2：该位不适用于 UART4 和 UART5。

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 USART_SR 寄存器中 LBD = 1 时，生成中断

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

位 4 保留，必须保持复位值

位 3:0 **ADD[3:0]**：USART 节点的地址 (Address of the USART node)

该位域用于指定 USART 节点的地址。

将在多处理器通信时于静音模式下使用该位域，以通过地址标记检测进行唤醒。

注： 使能发送器时不应对这 3 个位（CPOL、CPHA、LBCL）进行写操作。

28.6.6 控制寄存器 3 (USART_CR3)

Control register 3

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.	ONEBIT	CTSIE	CTSE	RTSE	DMAT	DMAR	SCEN	NACK	HDSEL	IRLP	IREN	EIE
				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 **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

位 10 **CTSIE**：CTS 中断使能 (CTS interrupt enable)

0：禁止中断

1：当 USART_SR 寄存器中 CTS = 1 时，生成中断

注： 该位不适用于 UART4 和 UART5。

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。

注：　该位不适用于 UART4 和 UART5。

位 8 **RTSE**：RTS 使能 (RTS enable)

0：禁止 RTS 硬件流控制

1：使能 RTS 中断，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（连接到 0）。

注：　该位不适用于 UART4 和 UART5。

位 7 **DMAT**：DMA 使能发送器 (DMA enable transmitter)

此位由软件置 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)

此位由软件置 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

注：　该位不适用于 UART4 和 UART5。

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

注：　该位不适用于 UART4 和 UART5。

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

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

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

位 0 **EIE**：错误中断使能 (Error interrupt enable)

对于多缓冲区通信（USART_CR3 寄存器中 DMAR = 1），如果发生帧错误、上溢错误或出现噪声标志（USART_SR 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

1：当 USART_CR3 寄存器中的 DMAR = 1 并且 USART_SR 寄存器中的 FE = 1 或 ORE = 1 或 NF = 1 时，将生成中断。

28.6.7 保护时间和预分频器寄存器 (USART_GTPR)

Guard time and prescaler 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
GT[7:0]								PSC[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 保留，必须保持复位值

位 15:8 **GT[7:0]**：保护时间值 (Guard time value)

该位域提供保护时间值（以波特时钟数为单位）。

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。

注： 该位不适用于 UART4 和 UART5。

位 7:0 **PSC[7:0]**：预分频器值 (Prescaler value)

– **在 IrDA 低功耗模式下：**

PSC[7:0] = IrDA 低功耗波特率

用于编程预分频器，进行系统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

...

– **在正常的 IrDA 模式下：**PSC 必须设置为 00000001。

– 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，进行系统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

...

注： 1：如果使用智能卡模式，则位 [7:5] 不起作用。

2：该位不适用于 UART4 和 UART5。

28.6.8 USART 寄存器映射

下表提供了 USART 寄存器映射和复位值。

表 173. 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	**USART_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
	Reset value																							0	0	1	1	0	0	0	0	0	0
0x04	**USART_DR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[8:0]								
	Reset value																								0	0	0	0	0	0	0	0	0
0x08	**USART_BRR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV_Mantissa[15:4]												DIV_Fraction [3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	**USART_CR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OVER8	Res.	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
	Reset value																	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	**USART_CR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LINEN	STOP [1:0]		CLKEN	CPOL	CPHA	LBCL	Res.	LBDIE	LBDL	Res.	ADD[3:0]			
	Reset value																		0	0	0	0	0	0	0		0	0		0	0	0	0
0x14	**USART_CR3**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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
0x18	**USART_GTPR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GT[7:0]								PSC[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

有关寄存器边界地址的信息，请参见*第 54 页的第 2.2.2 节*。

29 串行外设接口/集成电路内置音频总线 (SPI/I2S)

29.1 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 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 (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全双工和半双工通信。

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。

警告： **由于部分 SPI1 引脚可能会映射到 JTAG 接口所使用的部分引脚上，用户可以将 SPI/I2S 映射到其他引脚、（调试应用时）在配置 SPI 引脚前禁止 JTAG 接口而使能 SWD 接口、或者禁止 JTAG/SWD 接口（脱离调试，应用单独运行时）。有关配置 JTAG/SWD 接口引脚的更多信息，请参见*第 7.3.2 节：I/O 引脚复用器和映射*。**

29.1.1 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
- 8 位到 16 位传输帧格式选择
- 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 功能的单字节/字收发缓冲器：发送和接收请求

29.1.2 SPI 扩展特性

- 支持 SPI TI 模式

29.1.3 I2S 功能

- 全双工通信
- 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）
- I^2S（I2S1、I2S2、I2S3、I2S4 和 I2S5）时钟可由 I2S_CKIN 引脚上的外部时钟提供。

29.2 SPI/I2S 实现

本手册介绍了 SPI1、SPI2、SPI3、SPI4 和 SPI5 中实现的所有特性。

表 174. STM32F413/423 SPI 实现

SPI 特性(1)	SPI1	SPI2	SPI3	SPI4	SPI5
硬件 CRC 计算	X	X	X	X	X
I2S 模式	X	X	X	X	X
TI 模式	X	X	X	X	X

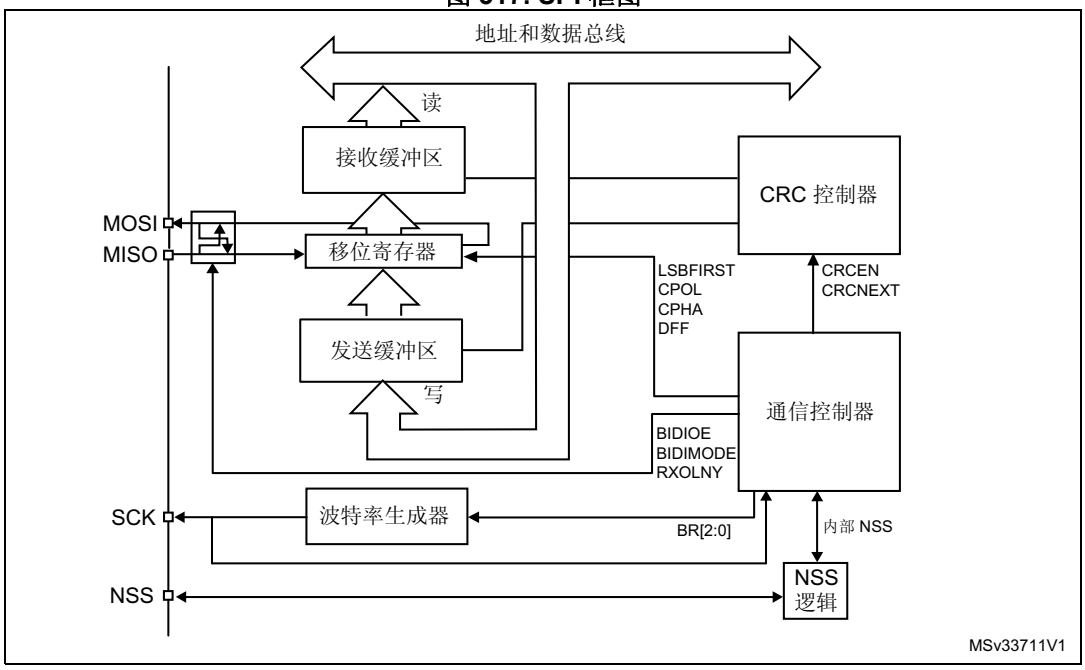
1. X = 支持。

29.3 SPI 功能说明

29.3.1 概述

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 317*。

图 317. 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 29.3.5 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。其他信号可以根据 SPI 节点间的数据交换及其从器件选择信号管理进行添加。

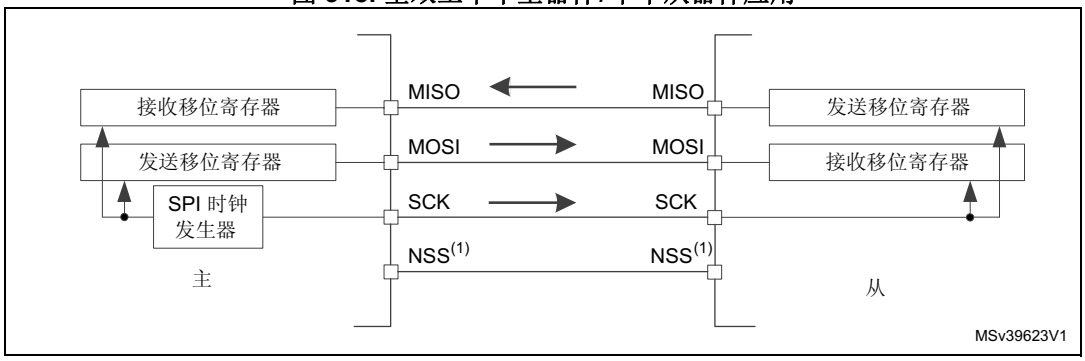
29.3.2 一个主器件和一个从器件之间的通信

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 318. 全双工单个主器件/单个从器件应用

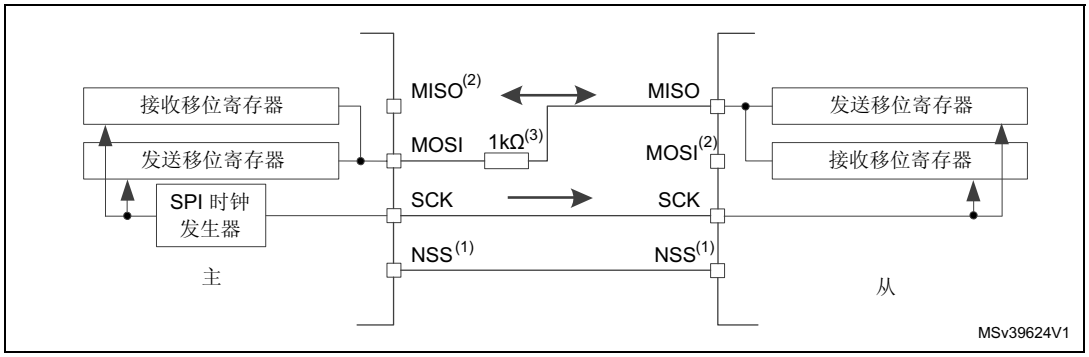


1. NSS 引脚可用于在主器件和从器件之间提供硬件控制流。外设也可选择不使用这些引脚。之后，必须在内部为主器件和从器件处理控制流。有关更多详细信息，请参见*第 29.3.5 节：从器件选择 (NSS) 引脚管理*。

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，可在其他应用中用作 GPIO。

图 319. 半双工单个主器件/单个从器件应用



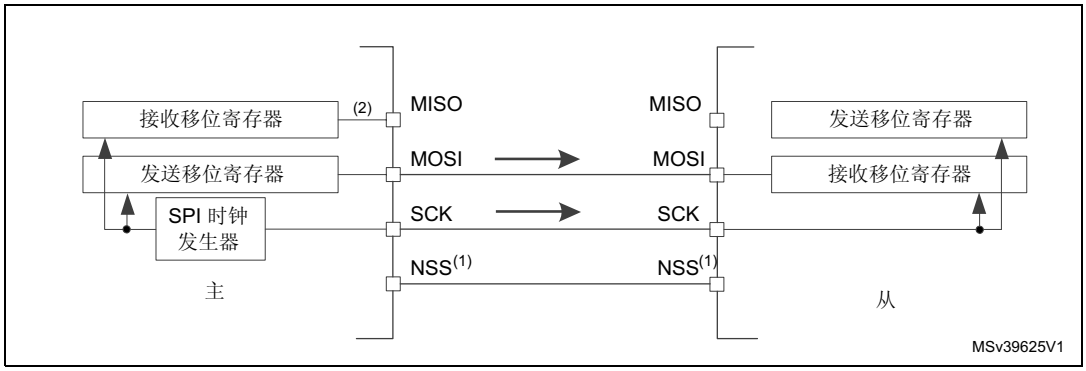
1. NSS 引脚可用于在主器件和从器件之间提供硬件控制流。外设也可选择不使用这些引脚。之后，必须在内部为主器件和从器件处理控制流。有关更多详细信息，请参见*第 29.3.5 节：从器件选择 (NSS) 引脚管理*。
2. 在这种配置下，主器件的 MISO 引脚和从器件的 MOSI 引脚可用作 GPIO。
3. 当以双向模式工作的两个节点间的通信方向不是同步变化时，会出现临界情况，新发送器访问共用数据线，而前一个发送器仍保持线路上的相反值（值取决于 SPI 配置和通信数据）。两个节点会出现冲突，在共用线上短暂提供相反的输出电平，直到下一个节点也相应地改变其方向设置。建议此模式下在 MISO 和 MOSI 引脚之间插入串行电阻以在这种情况下保护输出并限制电流在二者之间流过。

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 *29.3.5：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 320. 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）



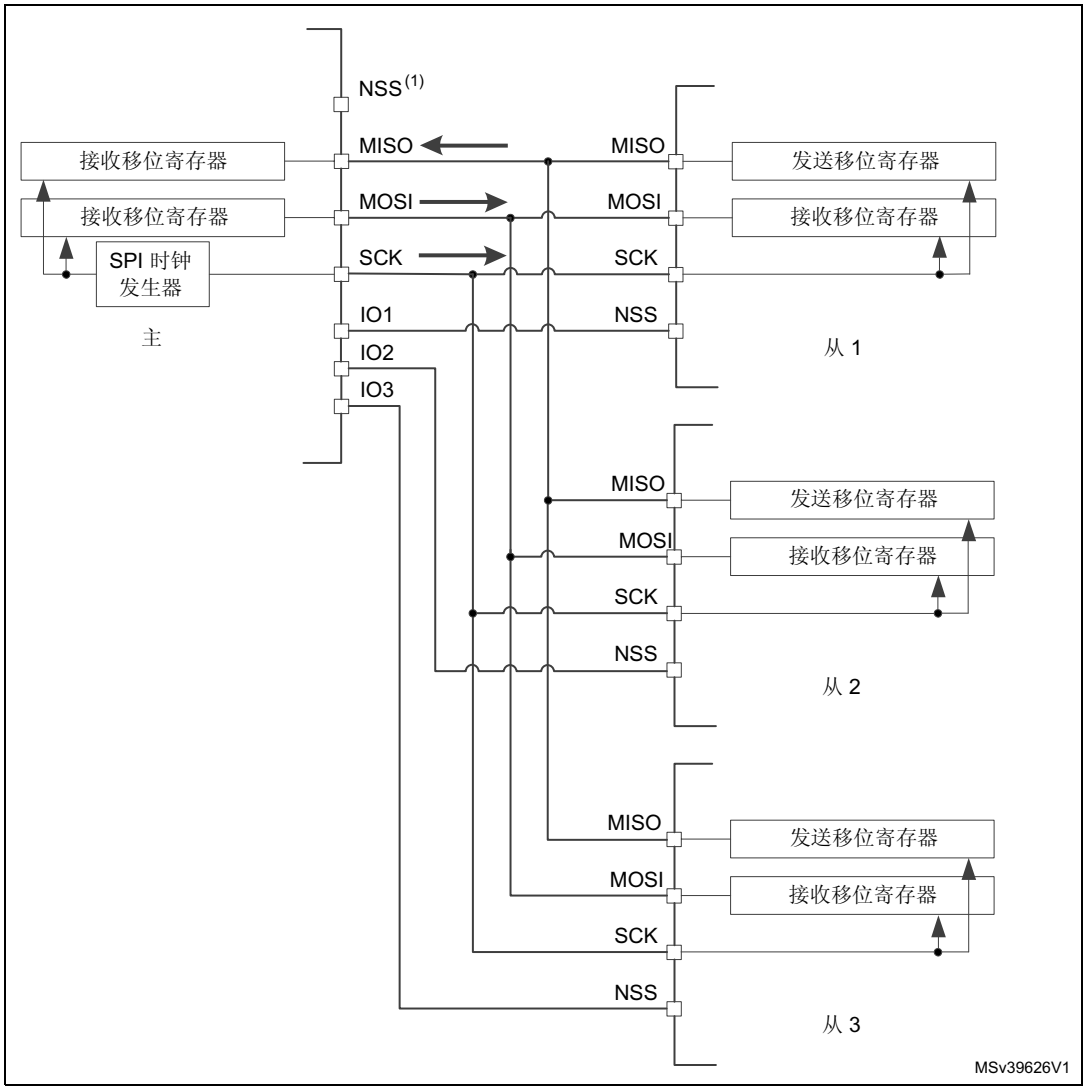
1. NSS 引脚可用于在主器件和从器件之间提供硬件控制流。外设也可选择不使用这些引脚。之后，必须在内部为主器件和从器件处理控制流。有关更多详细信息，请参见*第 29.3.5 节：从器件选择 (NSS) 引脚管理*。
2. 在发送器 Rx 移位寄存器的输入上捕获意外输入信息。标准只发送模式下必须忽略与发送器接收流相关的所有事件（例如 OVF 标志）。
3. 在这种配置下，两个 MISO 引脚均可用作 GPIO。

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（在保持 BDIO 位不变时，使能双向模式）。*

29.3.3 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 321*）。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 321. 主器件和三个独立的从器件



1. 此配置的主器件侧不使用 NSS 引脚。该引脚必须在内部管理 (SSM=1，SSI=1) 以避免任何 MODF 错误。
2. 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 179 页的第 7.3.7 节：I/O 复用功能输入/输出*）。

29.3.4 多主器件通信

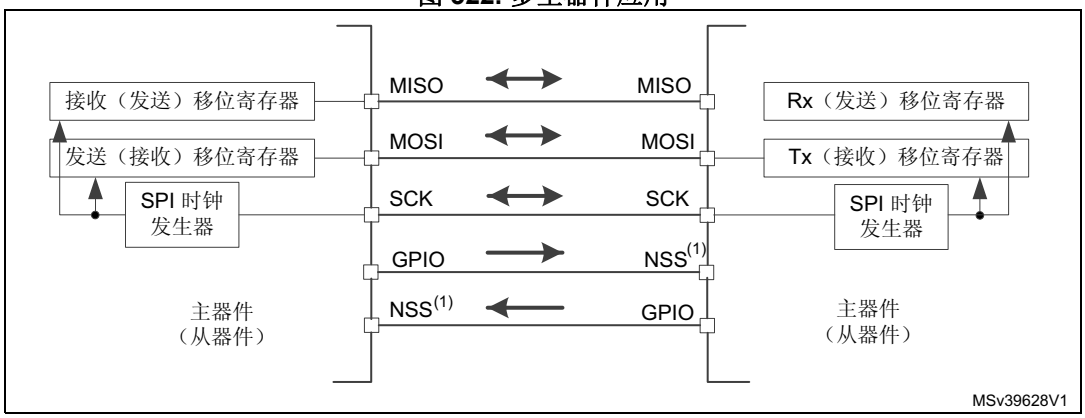
如果 SPI 总线未用于多主功能，用户可使用内置功能来检测尝试同时控制总线的两个节点间是否存在潜在冲突。对于该检测，NSS 引脚配置为硬件输入模式。

由于此时只有一个结点可将其输出施加到公用数据线上，因此无法连接超过两个以此模式工作的 SPI 节点。

当节点无效时，默认情况下均保持从模式。一旦一个节点要接管对总线的控制，它会将自身切换到主模式，然后通过专用 GPIO 引脚向其他节点的从器件选择输入施加有效电平。会话完成后，有效的从器件选择信号将被释放，控制总线的节点会短暂切换回被动从模式，等待下一个会话开始。

如果两个节点同时发出各自的控制请求，则会出现总线冲突（请参见模式故障 MODF 事件）。随后，用户可应用某个简单的仲裁过程（例如，在两个节点上施加不同的预定义超时来推迟下一个尝试）。

图 322. 多主器件应用



1. 在两个节点上，NSS 引脚配置为硬件输入模式。当无效节点配置为从器件时，其有效电平将使能 MISO 线输出控制。

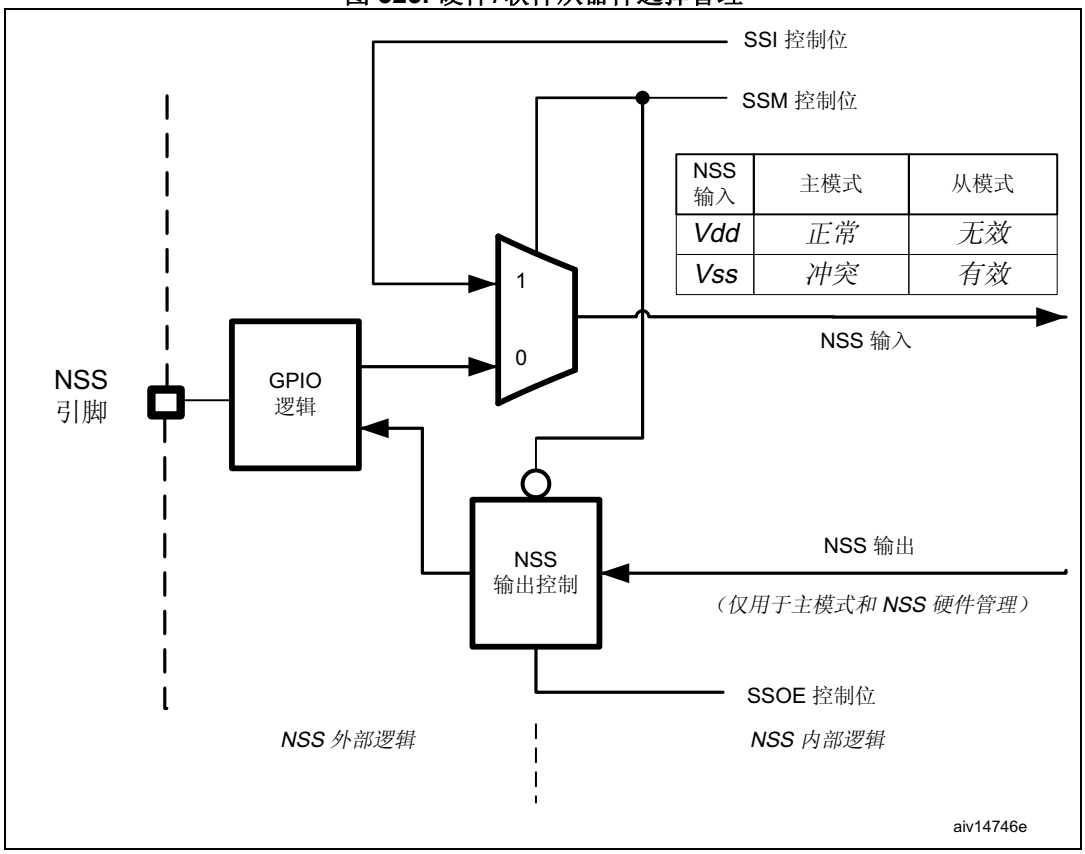
29.3.5 从器件选择 (NSS) 引脚管理

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，可供其他应用使用。
- **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)。
  - **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 323. 硬件/软件从器件选择管理



29.3.6 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。

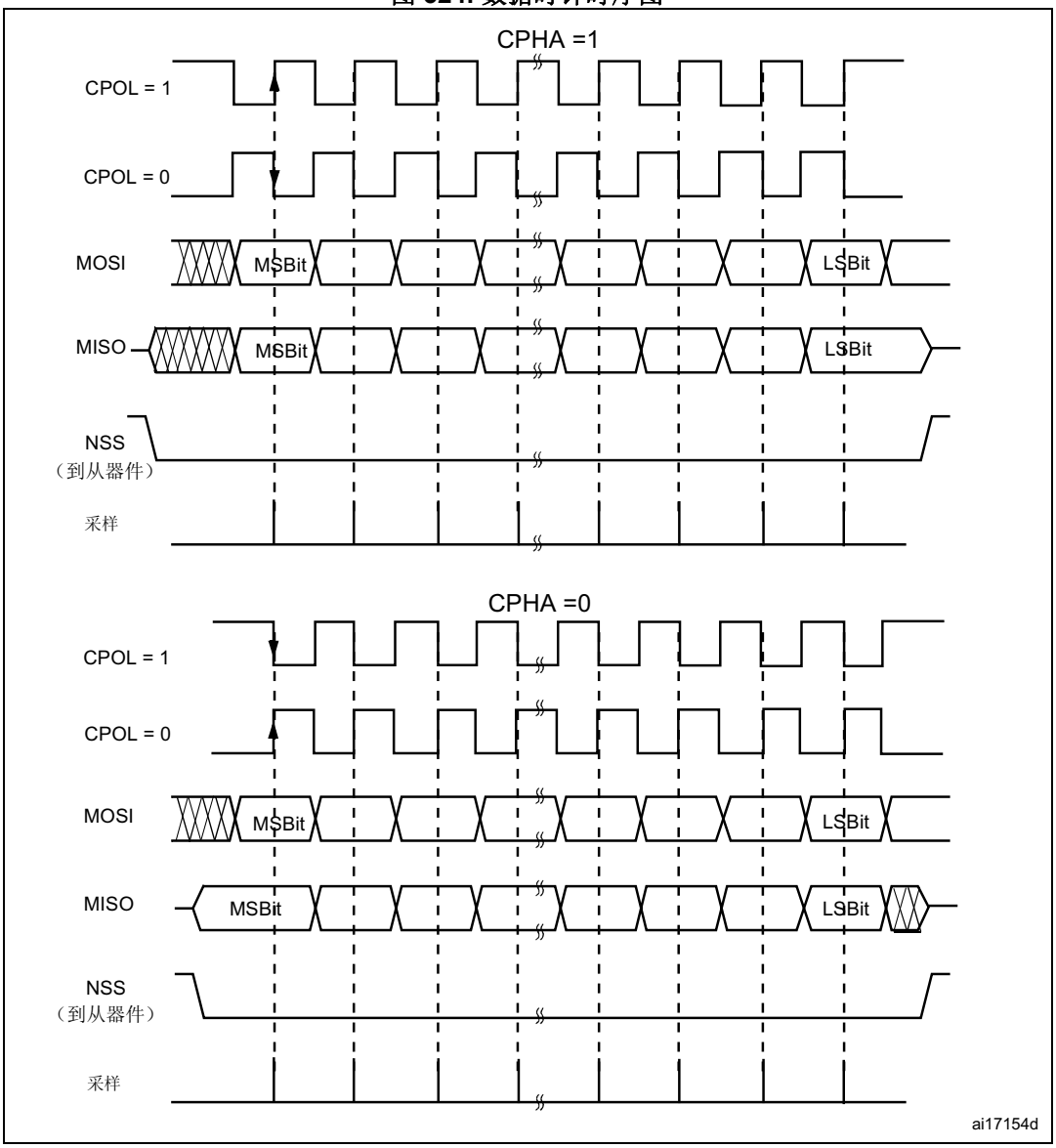
CPOL（时钟极性）和 CPHA（时钟相位）位的组合用于选择数据捕获时钟边沿。

图 324 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 324. 数据时钟时序图



注： *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。每个数据帧的长度均为 8 位或 16 位，具体取决于使用 SPI_CR1 寄存器中的 DFF 位编程的数据长度。所选的数据帧格式适用于发送和接收。

29.3.7 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. 对相应的 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：* 3）。
   b) 配置 CPOL 位和 CPHA 位组合，从数据与时钟四组时序定义中选择一种定义。（*注：* 2 - 在 TI 模式下使能 CRC 的情况除外）。
   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 CRCEN 和 CRCEN 位（SCK 时钟信号处于空闲状态时）。
   f) 配置 SSM 和 SSI（*注：* 2）。
   g) 配置 MSTR 位（在多主模式 NSS 配置下，如果配置为主器件，避免 NSS 上出现状态冲突，从而防止发生 MODF 错误）。
   h) 设置 DFF 位以配置数据帧格式（8 位或 16 位）。
3. 对 SPI_CR2 寄存器执行写操作：
   a) 配置 SSOE（*注：* 1 和 2）。
   b) 如果需要 TI 协议，请将 FRF 位置 1。
4. 对 SPI_CRCPR 寄存器执行写操作：需要时配置 CRC 多项式。
5. 对相应的 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) 从模式下无需此步骤，但从器件在 TI 模式下工作时除外。

29.3.8 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（不论是在通信时钟的第一个边沿，还是在时钟连续时，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。

在全双工（或任何只发送模式）下，将待发送数据写入发送缓冲区，并使能 SPI 后，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。

从器件接收到来自主器件的正确时钟信号时，它将开始通信。在 SPI 主器件启动传输前，从软件必须写入待发送的数据。

有关如何处理 DMA 的详细信息，请参见*第 29.3.11 节：使用 DMA（直接存储器寻址）进行通信*。

29.3.9 数据发送和接收过程

接收和发送缓冲区

在接收过程中，数据收到后，先存储到内部接收缓冲区中；而在发送过程中，先将数据存储到内部发送缓冲区中，然后发送数据。对 SPI_DR 寄存器的读访问将返回接收缓冲值，而对 SPI_DR 寄存器的写访问会将写入的数据存储到发送缓冲区中。

发送缓冲区处理

在第一个位传输期间，数据帧从发送缓冲区加载到移位寄存器。各个位随后从移位寄存器以串行方式移出到专用输出引脚，具体取决于 LSBFIRST 位设置。当数据从发送缓冲区传送到移位寄存器时，TXE 标志（发送缓冲区为空）置 1。该标志表示内部发送缓冲区已准备好加载接下来的数据。如果 SPI_CR2 寄存器中的 TXEIE 位置 1，可产生中断。通过对 SPI_DR 寄存器执行写操作将 TXE 位清零。

如果在前一次帧传输仍在进行时将要发送的下一个数据存储到发送缓冲区，则可保持连续的发送流。如果软件在 TXE 未置 1 时写入发送缓冲区，则等待发送的数据将被覆盖。

接收缓冲区处理

将数据从移位寄存器传输到接收缓冲区时，RXNE 标志（接收缓冲区非空）会在最后一个采样时钟边沿置 1。它表示已准备好从 SPI_DR 寄存器中读取数据。如果 SPI_CR2 寄存器中的 RXNEIE 位置 1，可产生中断。读取 SPI_DR 寄存器即会使 RXNE 位清零。

如果器件未将前一个数据传输产生的 RXNE 位清零，则在下一个值缓冲时会产生溢出条件。OVR 位置 1 并在 ERRIE 位置 1 时生成一个中断。

另一种管理数据交换的方式是使用 DMA（请参见*第 9.2 节：DMA 主要特性*）。

序列处理

当前数据帧传输正在进行时，BSY 位将置 1。当时钟信号连续运行时，BSY 标志在主器件侧的两个数据帧间保持置 1。不过，在从器件侧，它将在每个数据帧传输之间变为 0 并持续至少一个 SPI 时钟周期。

对于某些配置，可以在最后一次数据传输期间使用 BSY 标志来等待传输完成。

当主器件侧配置只接收模式时，无论是半双工（BIDIMODE=1 且 BIDIOE=0）还是单工配置（BIDIMODE=0 且 RXONLY=1），主器件都会在 SPI 使能后立即启动接收序列。随后，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能力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，在 SPI 模式下工作时不存在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，不必使用 NSS 来控制从器件。但是，可使用 NSS 脉冲将从器件与每个数据传输序列的开始同步。NSS 可通过软件或硬件进行管理（请参见*第 29.3.4 节：多主器件通信*）。

有关主器件/全双工和从器件/全双工模式下的连续传输的说明，请参见*图 325* 和*图 326*。

图 325. 主/全双工模式（BIDIMODE=0 且 RXONLY=0）下的 TXE/RXNE/BSY 时序（在连续传输的情况下）

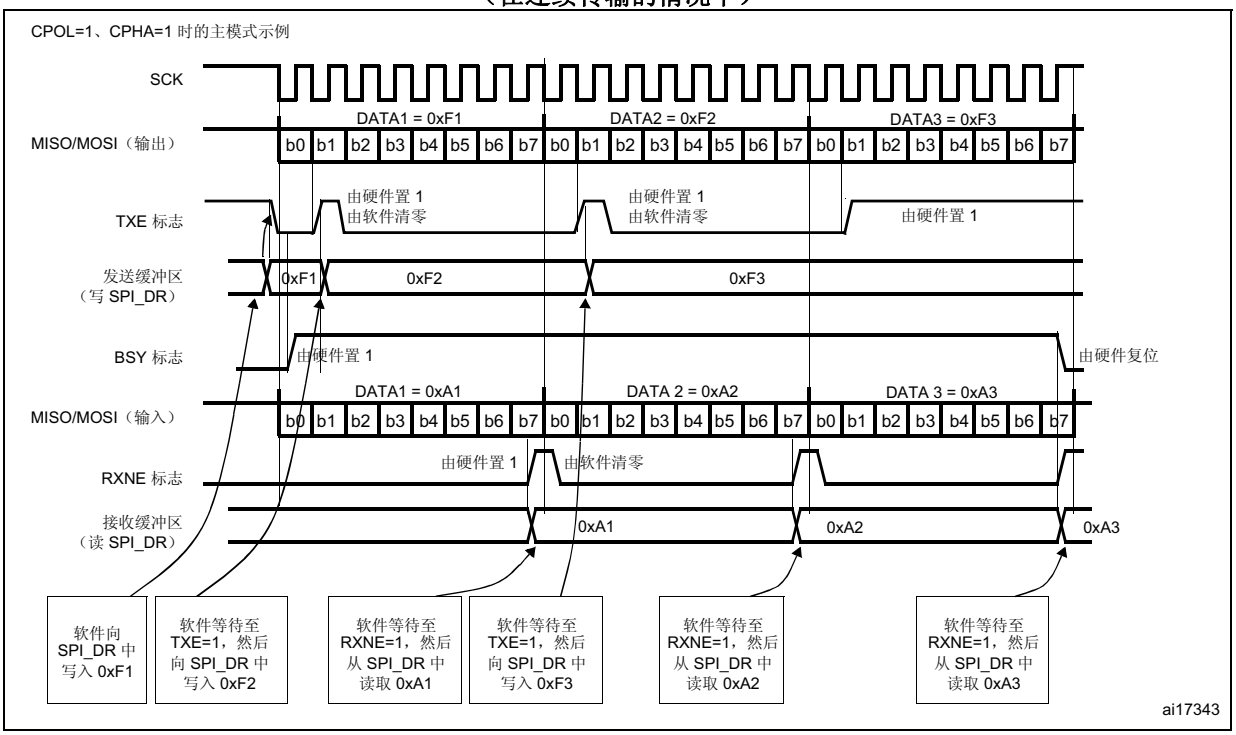
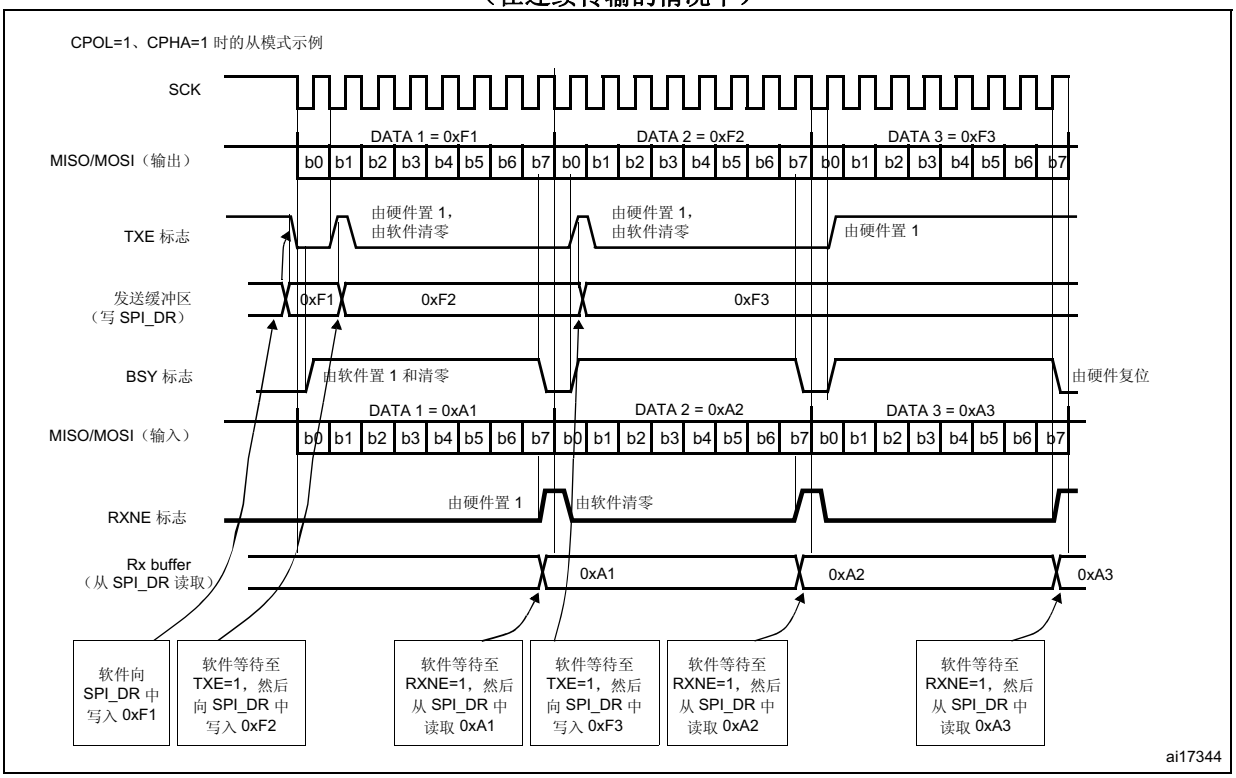


图 326. 从器件/全双工模式（BIDIMODE=0 且 RXONLY=0）下的 TXE/RXNE/BSY 时序（在连续传输的情况下）



29.3.10 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。

标准禁止步骤通过轮询 BSY 状态以及 TXE 标志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 信号由任意 GPIO 切换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 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 RXNE=1 以接收最后的数据。
2. 等待至 TXE=1，然后等待至 BSY=0，再关闭 SPI。
3. 读取接收的数据。

注： *在不连续通信期间，在对 SPI_DR 寄存器执行写操作与 BSY 位置 1 之间有 2 个 APB 时钟周期的延迟。因此，写入最后的数据后，必须先等待 TXE 位置 1，然后等待 BSY 位清零。*

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. 读取接收的数据。

注： *要停止连续的接收序列，必须在接收最后一个数据帧时遵循特定的时间窗口。该时间窗口在第一个位采样时开始，在最后一个位的传输开始前结束。*

29.3.11 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。

有关 DMA 发送和接收波形的说明，请参见*图 327* 和*图 328*。

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 TXE=1，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. 如果使能了 DMA，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。

图 327. 使用 DMA 进行发送

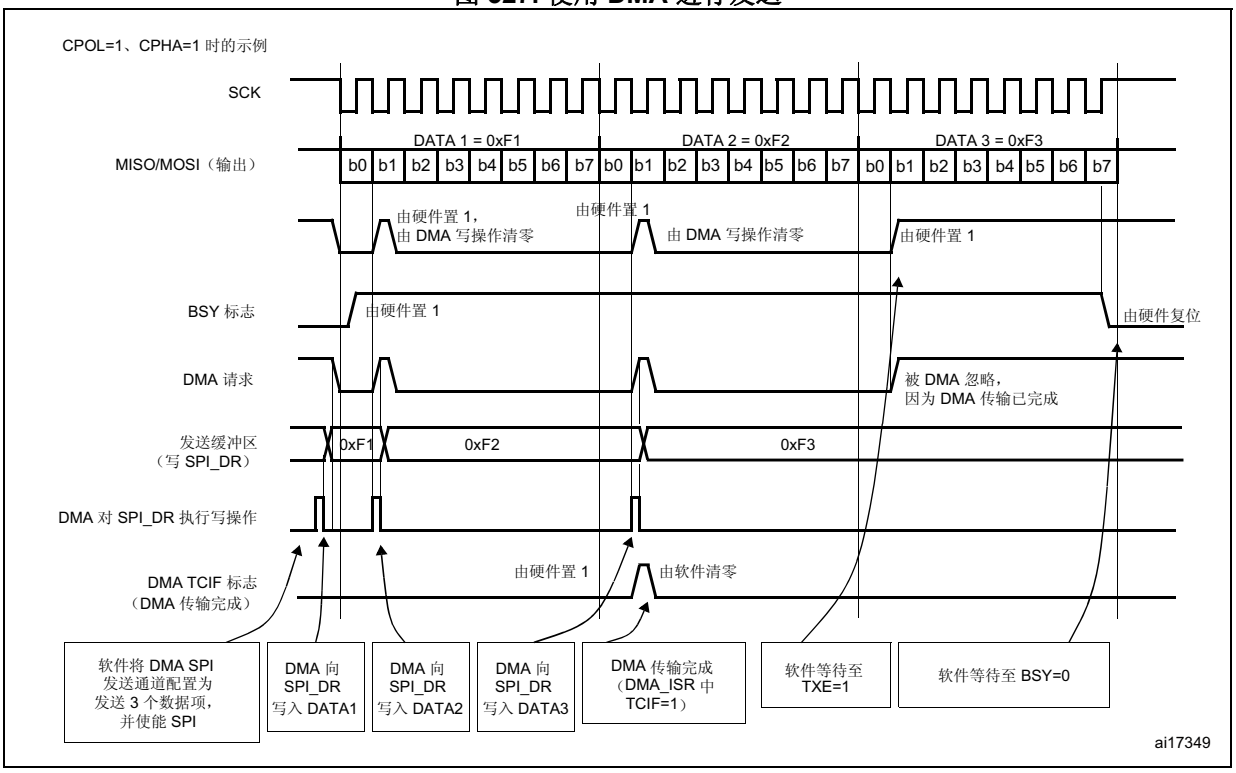
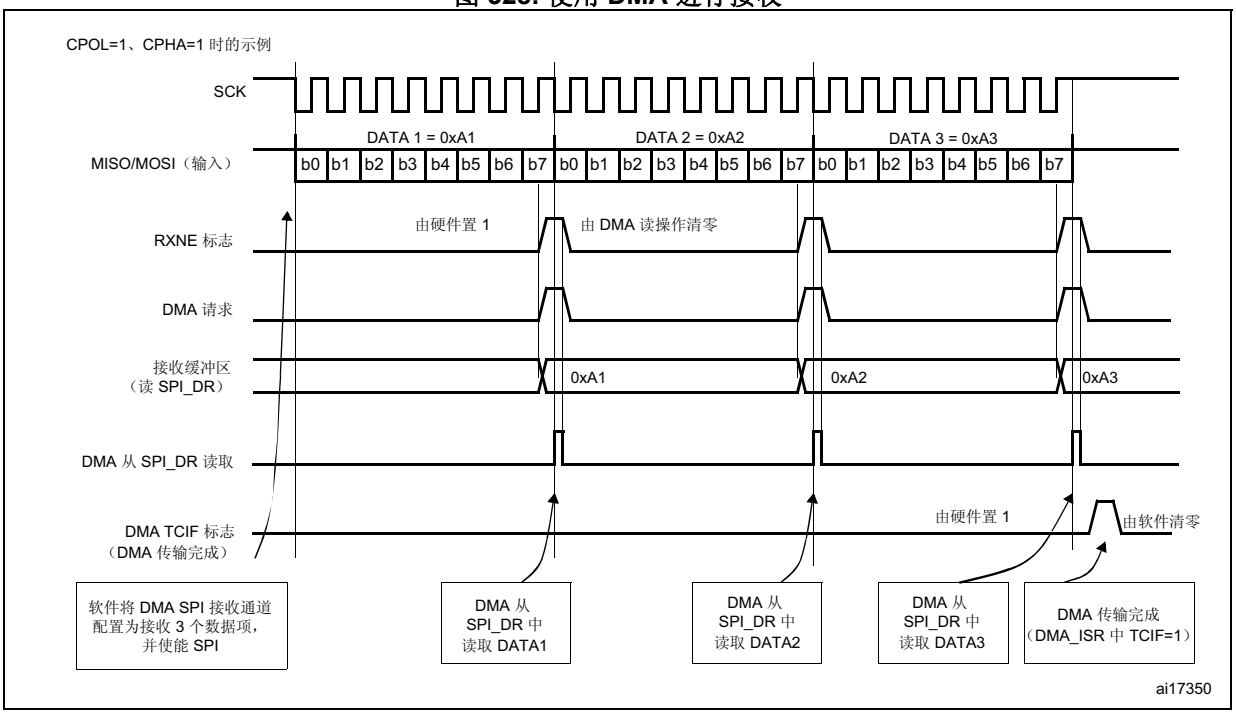


图 328. 使用 DMA 进行接收



29.3.12 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)

TXE 标志置 1 时，表示发送缓冲区为空，可以将待发送的下一个数据加载到缓冲区中。对 SPI_DR 寄存器执行写操作时，将清零 TXE 标志。

接收缓冲区非空 (RXNE)

RXNE 标志置 1 时，表示接收缓冲区中存在有效的已接收数据。通过对 SPI_DR 寄存器执行读取操作将该位清零。

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）。在主模式下的双向通信接收模式（MSTR=1 且 BDM=1 且 BDOE=0）有一个例外情况，BSY 标志在接收过程中保持置 0。

在某些模式下，可使用 BSY 标志来检测传输是否结束，从而避免在进入低功耗模式前禁止 SPI 外设时钟时或通过软件管理 NSS 脉冲结束时破坏最后一次传输。

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： *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。*

29.3.13 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)

主器件或者从器件在接收缓冲器中的上一个帧数未被读取时（RXNE 标志置 1），下一帧数据被接收完成，则会出现溢出条件。

在这种情况下，Rx 缓冲区的内容不会更新为接收的新数据。对 SPI_DR 寄存器执行的读操作将返回先前接收的帧。后续发送的所有其他数据均将丢失。

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_RXCRC 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。

29.4 SPI 特性

29.4.1 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 329*）。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：

$$\frac{t_{baud\_rate}}{2} + 4 \times t_{pclk} < t_{release} < \frac{t_{baud\_rate}}{2} + 6 \times t_{pclk}$$

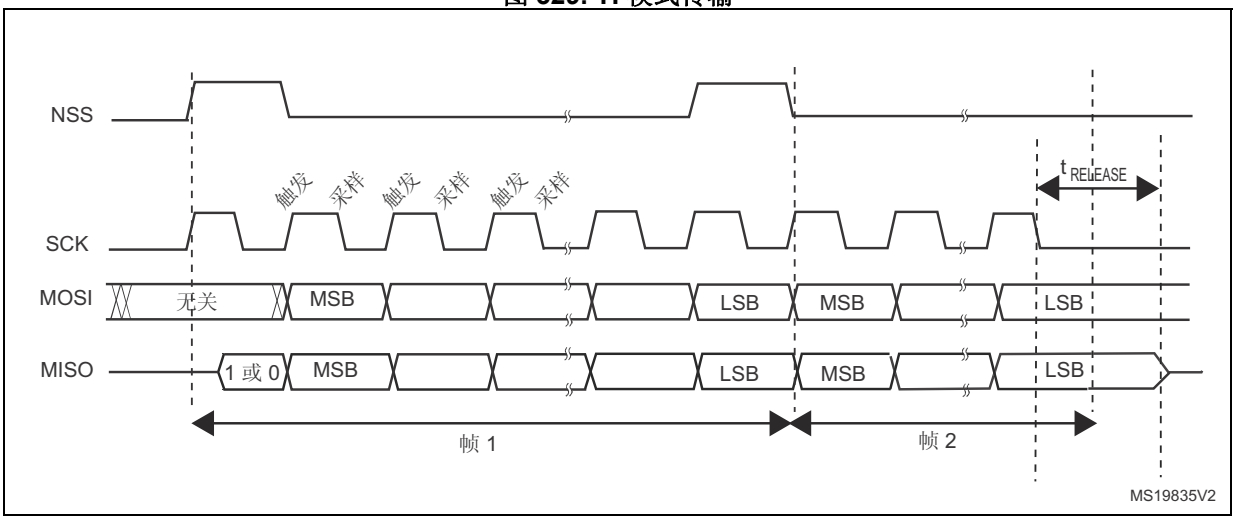
如果从器件在数据帧传输期间检测到错位的 NSS 脉冲，TIFRE 标志将置 1。

此特性不适用于 Motorola SPI 通信（FRF 位为 0）。

注： *要在从器件发送器模式下使用错误中断 (ERRIE = 1) 检测 TI 帧错误，必须通过将 SPI_CR1 寄存器中的 BIDIMODE 和 BIDIOE 置 1 来将 SPI 配置为双线单向模式。当 BIDIMODE 置为 0 时，OVR 将置 1，因为始终不会读取数据寄存器从而始终生成错误中断；而当 BIDIMODE 置 1 时，不会接收数据，也不会将 OVR 置 1。*

图 329 给出了选择 TI 模式时的 SPI 通信波形。

图 329. TI 模式传输



29.4.2 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（作用于发送和接收数据流）。SPI 提供 CRC8 或 CRC16 计算，具体取决于通过 DFF 位选择的数据格式。CRC 通过 SPI_CRCPR 寄存器中编程的多项式连续计算。

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。

与任何其他数据帧一样，接收的 CRC 存储在接收缓冲区内。

CRC 帧的传输通常在数据序列结束时再传输一个数据帧。

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 值存储到接收缓冲区中，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 SPI_DR 中接收的 CRC 帧（因为该信息始终加载到其中）。

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。

复位 SPIx_TXCRC 和 SPIx_RXCRC 值

当使能 CRC 计算时，SPIx_TXCRC 和 SPIx_RXCRC 值自动清零。

当 SPI 配置为从模式并且 CRC 功能已使能时，即使 NSS 引脚上为高电平，也会进行 CRC 计算。例如，在多从模式环境下可能出现这种情况，此时通信主器件会交替寻址从器件。

在禁止从器件（NSS 上为高电平）到使能新的从器件（NSS 上为低电平）的时间内，应在主器件和从器件两端同时将 CRC 值清零，以重新同步主从双方的 CRC 计算。

要将 CRC 清零，请按以下步骤操作：

1. 禁止 SPI
2. 将 CRCEN 位清零
3. 使能 CRCEN 位
4. 使能 SPI

注： *当 SPI 接口配置为从模式时，一旦 CRCNEXT 信号被释放，NSS 内部信号需要在处理 CRC 阶段的事务期间保持低电平（更多详细信息，请参见产品勘误表）。*

在 TI 模式下，尽管时钟相位和时钟极性设置固定并且与 SPIx_CR1 寄存器无关，但如果应用 CRC，则 SPIx_CR1 寄存器中必须保持相应的设置（CPOL = 0，CPHA = 1）。此外，CRC 计算必须通过 SPI 禁止序列在会话之间复位，方法是在主器件和从器件两侧重新使能上述 CRCEN 位，否则 CRC 计算可能在该特定模式下损坏。

29.5 SPI 中断

在 SPI 通信过程中，中断可由以下事件产生：

- 发送缓冲区准备就绪，可以装载数据
- 接收缓冲区中接收了数据
- 主模式故障
- 上溢错误
- TI 帧格式错误

中断可分别进行使能和禁止。

表 175. SPI 中断请求

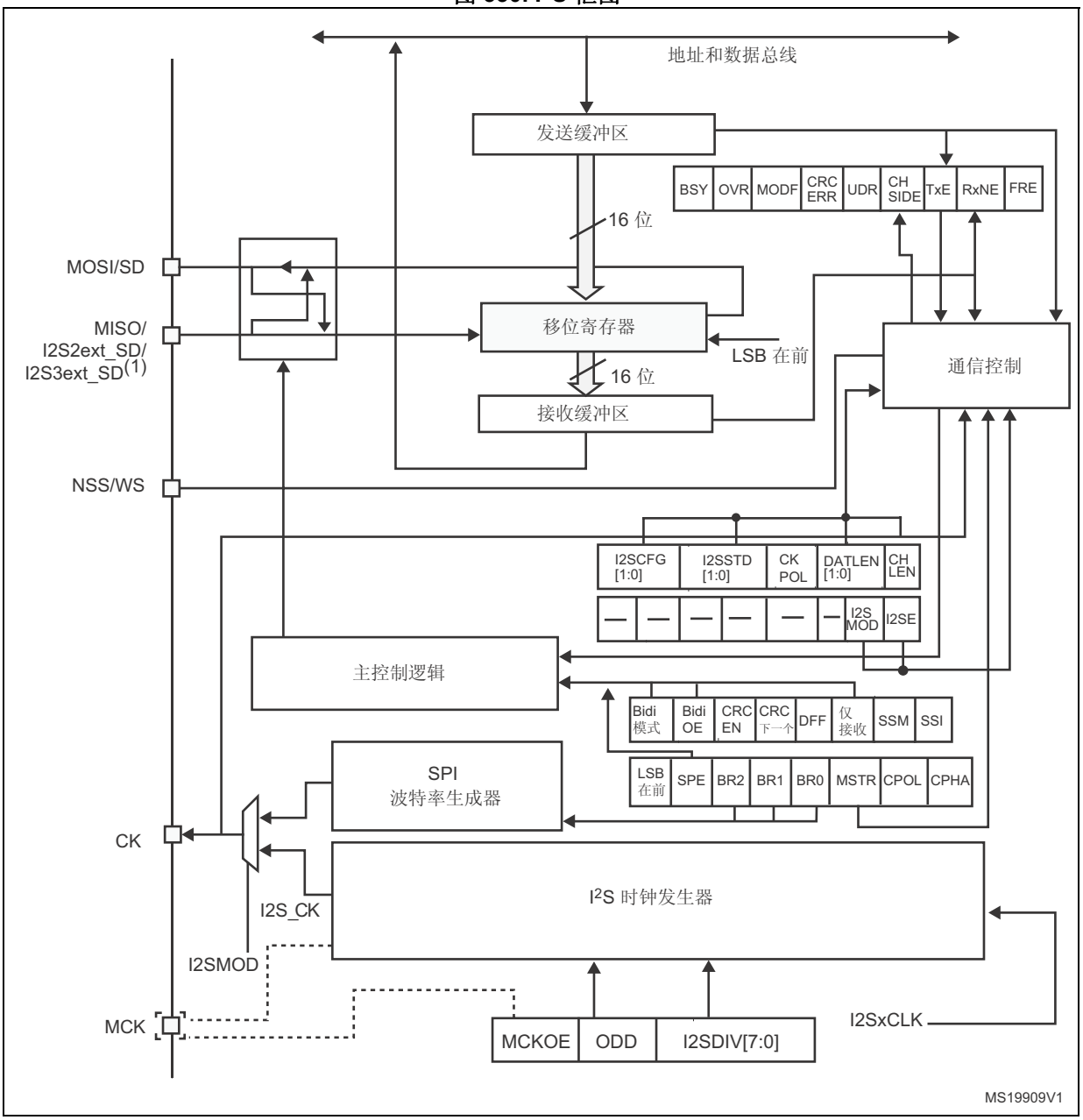
中断事件	事件标志	使能控制位
发送缓冲区准备就绪，可以装载数据	TXE	TXEIE
接收缓冲区中接收了数据	RXNE	RXNEIE
主模式故障	MODF	ERRIE
上溢错误	OVR	
CRC 错误	CRCERR	
TI 帧格式错误	FRE	

29.6 I²S 功能说明

29.6.1 I²S 一般说明

I²S 的框图如*图 330* 所示。

图 330. I²S 框图



1. I2S2ext_SD 和 I2S3ext_SD 为扩展 SD 引脚，用于控制 I2S 全双工模式。
2. MCK 映射到 MISO 引脚上。

当使能 I^2S 功能（将 SPIx_I2SCFGR 寄存器中的 I2SMOD 位置 1）后，SPI 可用作音频 I^2S 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，可以使用其他引脚：

- 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。

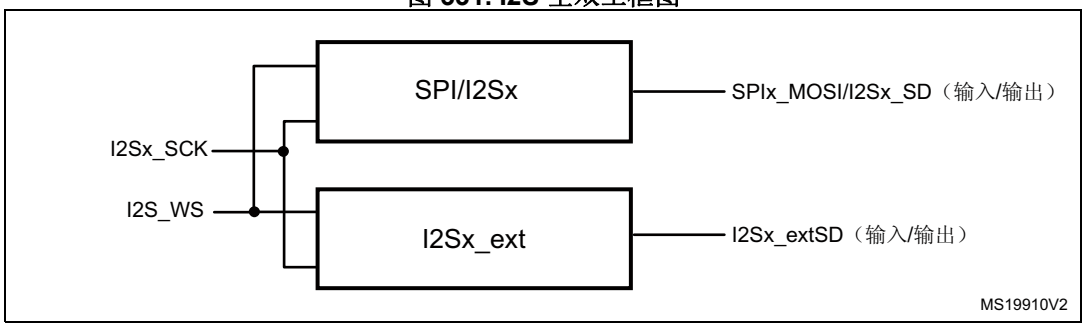
I^2S 使用相同的 SPI 寄存器 (SPIx_DR) 进行 16 位数据传输。

29.6.2 I2S 全双工

为支持 I2S 全双工模式，除了 I2S2 和 I2S3，还可以使用两个额外的 I^2S，它们称为扩展 I2S（I2S2_ext、I2S3_ext）（参见 *图 331*）。因此，第一个 I2S 全双工接口基于 I2S2 和 I2S2_ext，第二个基于 I2S3 和 I2S3_ext。

注： *I2S2_ext 和 I2S3_ext 仅用于全双工模式。*

图 331. I2S 全双工框图



1. 其中 x 可以是 2 或 3。

I2Sx 可以在主模式下工作。因此：

- 只有 I2Sx 可在半双工模式下输出 SCK 和 WS
- 只有 I2Sx 可在全双工模式下向 I2S2_ext 和 I2S3_ext 提供 SCK 和 WS

扩展 I2S (I2Sx_ext) 只能用于全双工模式。I2Sx_ext 始终在从模式下工作。

I2Sx 和 I2Sx_ext 均可用于发送和接收。

29.6.3 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 位寄存器进行发送或者接收。所以，需由软件将与每个通道对应的值写入数据寄存器，或者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 16 位数据封装在 32 位数据帧的格式时，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）。

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，或者当采样 DMA 时，则需要两次 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。

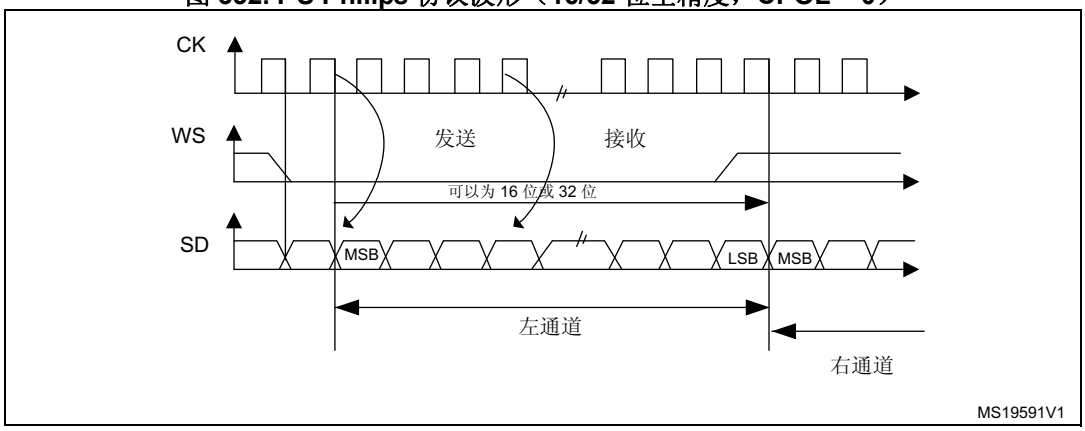
对于所有数据格式和通信标准而言，始终会先发送最高有效位（MSB 优先）。

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。

I^2S Philips 标准

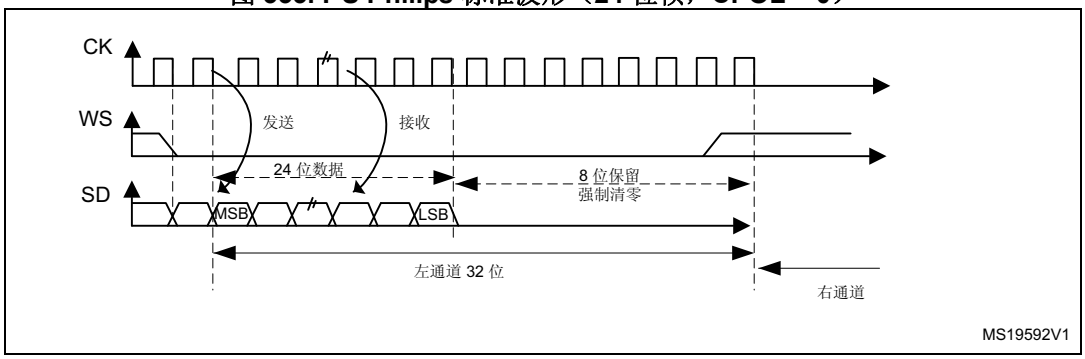
使用 WS 信号来指示当前正在发送的数据所属的通道。在第一个位 (MSB) 有效之前，存在一个时钟周期。

图 332. I^2S Philips 协议波形（16/32 位全精度，CPOL = 0）



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。

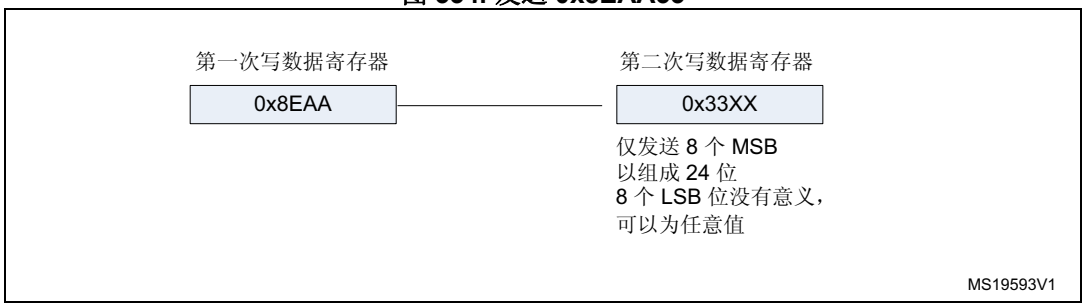
图 333. I²S Philips 标准波形（24 位帧，CPOL = 0）



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 334. 发送 0x8EAA33



- 在接收模式下：
  如果接收数据 0x8EAA33：

图 335. 接收 0x8EAA33

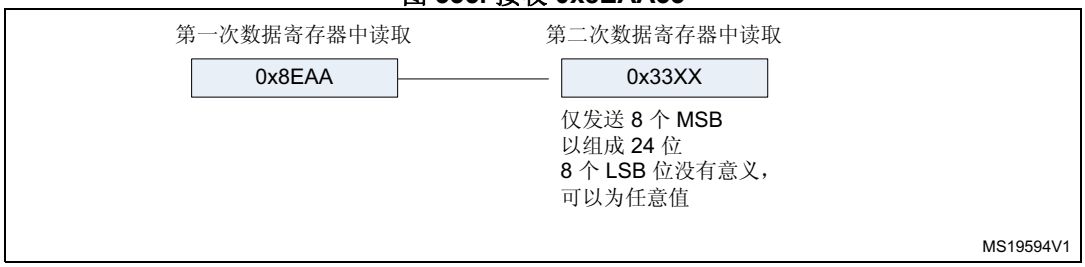
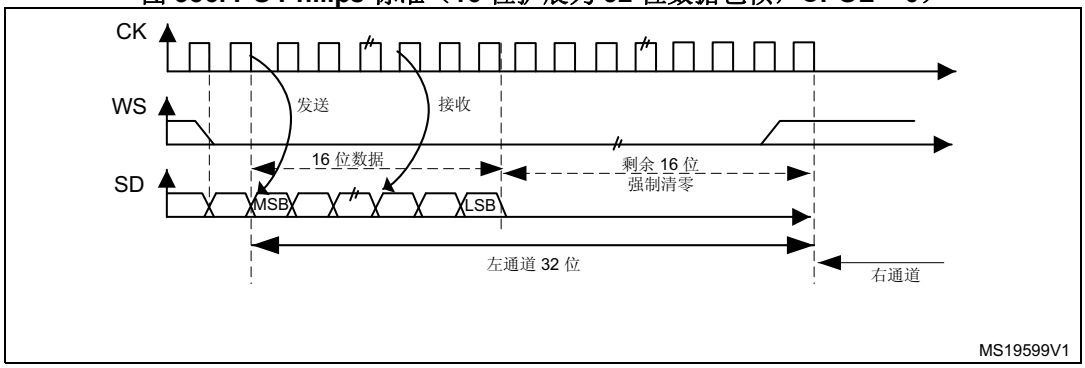


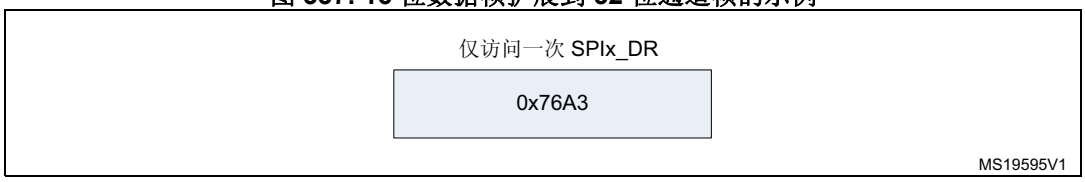
图 336. I²S Philips 标准（16 位扩展为 32 位数据包帧，CPOL = 0）



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。扩展到 32 位中低半字被硬件置为 0x0000。

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 337* 中显示的操作。

图 337. 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。

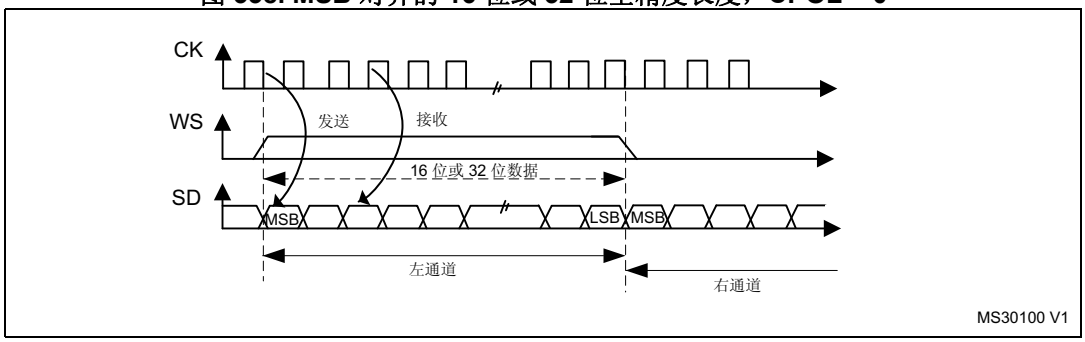
接收时，接收到第一个半字（高 16 位），则硬件将 RXNE 标志置 1，并在中断使能的情况下触发中断。

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 338. MSB 对齐的 16 位或 32 位全精度长度，CPOL = 0



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 339. MSB 对齐的 24 位帧长度，CPOL = 0

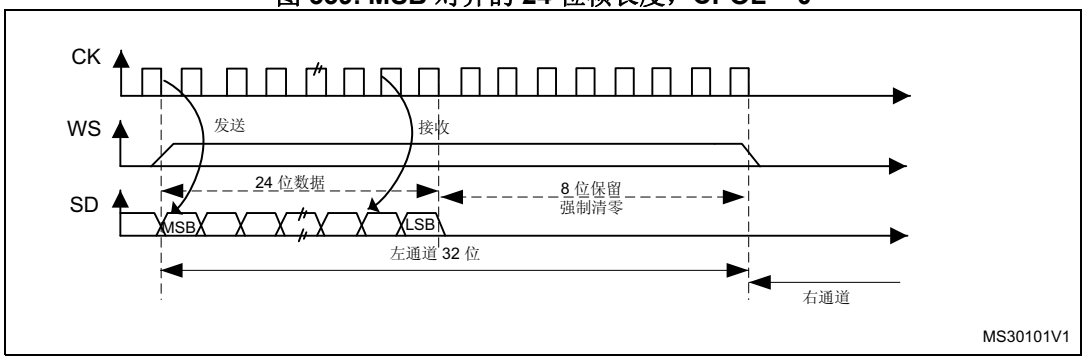
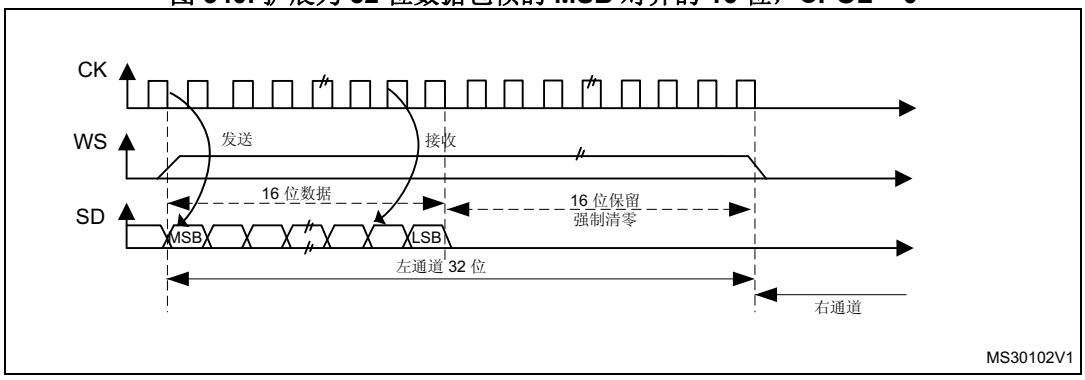


图 340. 扩展为 32 位数据包帧的 MSB 对齐的 16 位，CPOL = 0



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）。

图 341. LSB 对齐的 16 位或 32 位全精度，CPOL = 0

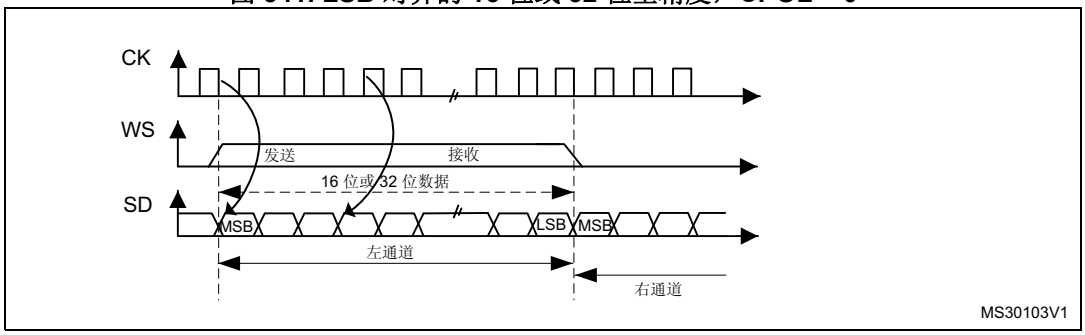
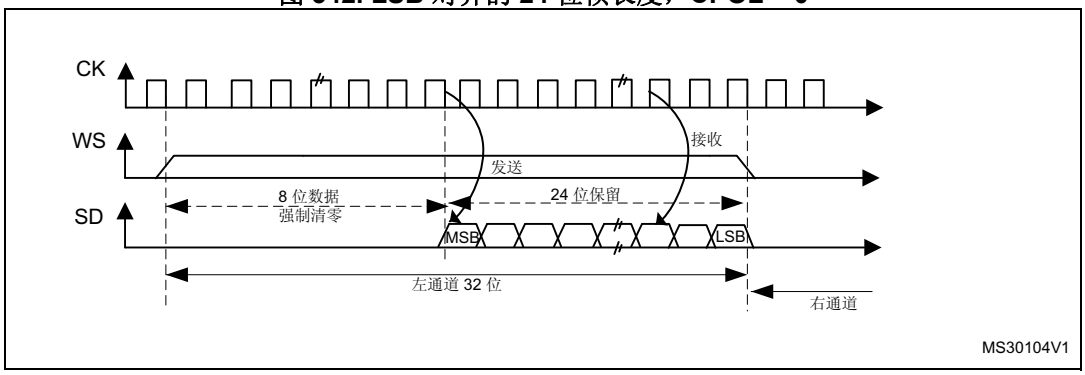


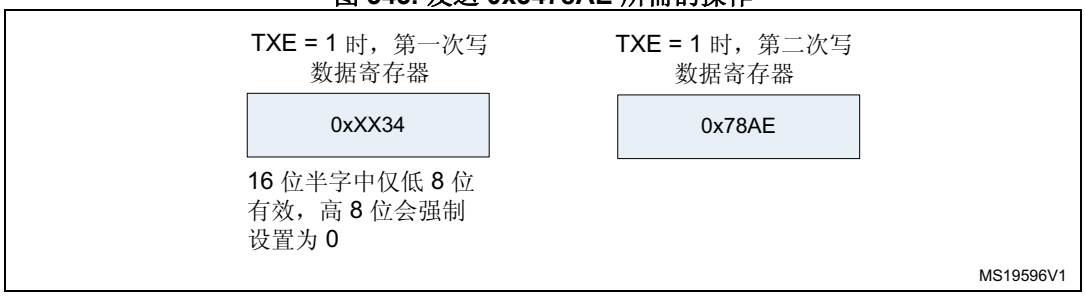
图 342. LSB 对齐的 24 位帧长度，CPOL = 0



- 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 343. 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 344. 接收 0x3478AE 时所需的操作

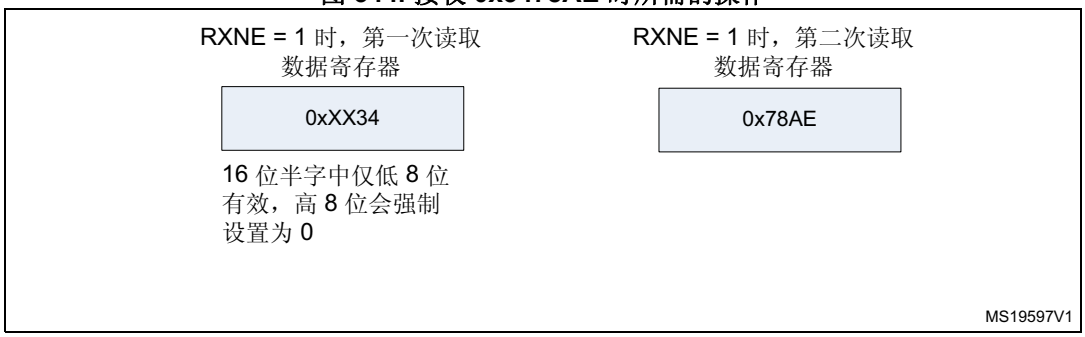
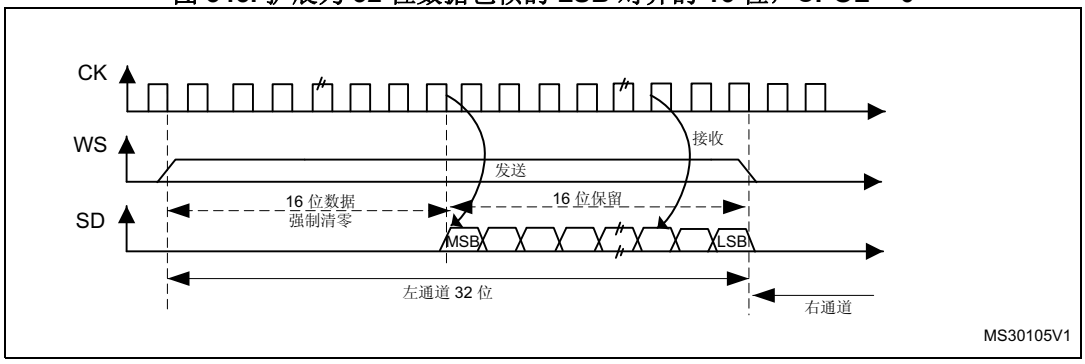


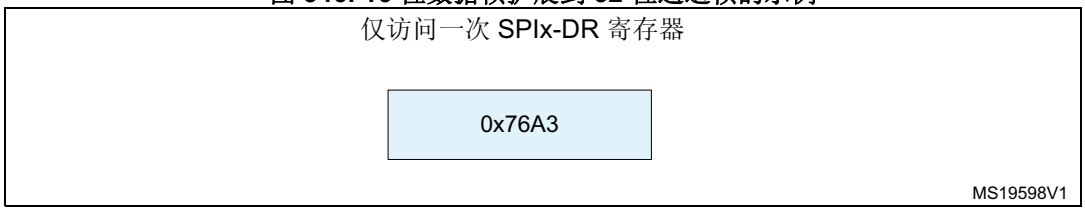
图 345. 扩展为 32 位数据包帧的 LSB 对齐的 16 位，CPOL = 0



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。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 346* 中显示的操作。

图 346. 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。

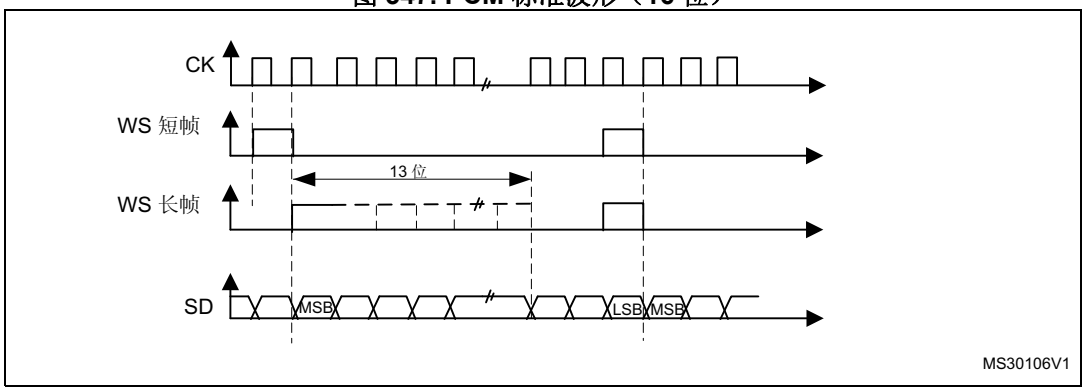
在接收模式下，当接收到有效半字后（而非 0x0000 字段），即会置位 RXNE。

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。

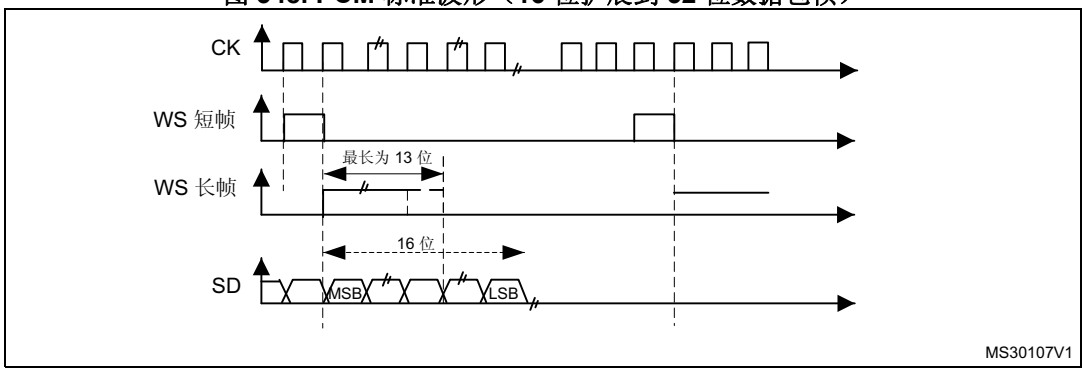
图 347. PCM 标准波形（16 位）



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 348. 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）。*

29.6.4 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。

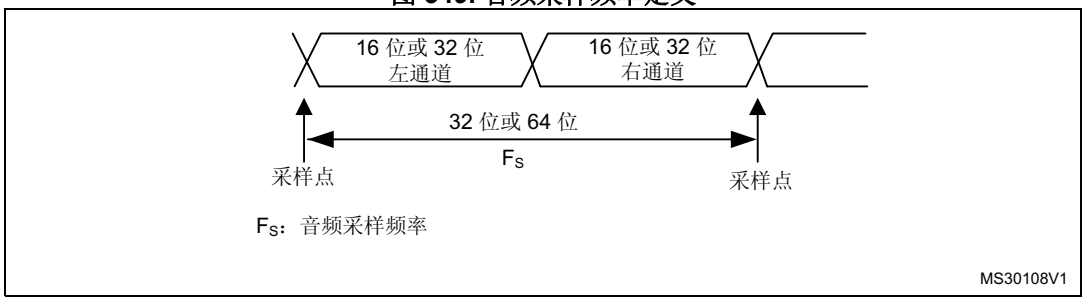
I^2S 比特率 = 每个通道的位数 × 通道数 × 音频采样频率

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。

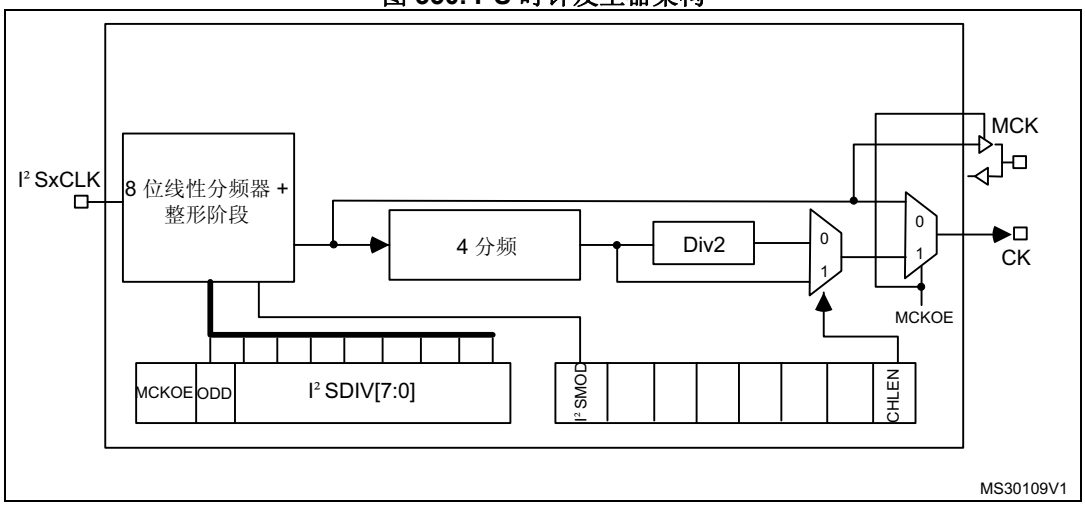
图 349. 音频采样频率定义



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 350 展示了通信时钟架构。I2Sx 时钟始终为系统时钟。

图 350. I²S 时钟发生器架构



1. 其中 x = 2。

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他值）。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 176 提供了针对不同时钟配置的示例精度值。

注： *还可以采用其他配置以达到更好的时钟精度。*

表 176. 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 f_S (Hz)	实际 f_s (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%
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。

29.6.5 I²S 主模式

I²S 可如下配置：

- 发送主器件或接收主器件（使用 I2Sx 的半双工模式）。
- 同时收发的主器件（使用 I2Sx 和 I2Sx_ext 的全双工模式）。

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，从而达到相应的音频采样频率。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 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 29.6.4 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 29.6.3 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 29.6.5 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 29.6.3 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 的所有其他组合，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。*

29.6.6 I²S 从模式

I²S 可如下配置：

- 发送从器件或接收从器件（使用 I2Sx 的半双工模式）
- 同时收发的从器件（使用 I2Sx 和 I2Sx_ext 的全双工模式）。

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： *必须在主器件发出的第一个时钟出现在 CK 线前至少 2 个 PCLK 周期置位 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 29.6.3 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 29.6.6 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 I^2S 标准模式的读操作的更多详细信息，请参见*第 29.6.3 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 I^2S，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。*

29.6.7 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 处于主机接收模式时除外），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。

29.6.8 I^2S 错误标志

I^2S 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他半字都将丢失。

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. 在 WS 线上检测到正确的电平时将其重新使能（WS 线在 I²S 模式下为高电平，在 MSB 对齐、LSB 对齐或 PCM 模式下为低电平）。

主器件与从器件之间的同步失效可能是由于 SCK 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。

29.6.9 I²S 中断

表 177 为 I²S 中断的列表。

表 177. 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	

29.6.10 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，没有其他差别。

29.7 SPI 和 I²S 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。

有关寄存器说明中使用的缩写，请参见*第 1.2 节*。

外设寄存器可支持半字（16 位）或字（32 位）访问。

29.7.1 SPI 控制寄存器 1 (SPI_CR1)（不用于 I²S 模式）

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	DFF	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 I²S 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： *主模式下使用 MOSI 引脚，而从模式下使用 MISO 引脚。*

该位不适用于 I²S 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。*

*不**适用于 I²S 模式。***

位 12 **CRCNEXT：**下一次 CRC 传输 (CRC transfer next)

0：数据阶段（无 CRC 阶段）

1：下一次传输为 CRC（CRC 阶段）

注： *当 SPI 配置为全双工或只发送器模式时，只要最后一个数据写入 SPI_DR 寄存器，就必须对 CRCNEXT 执行写操作。*

当 SPI 配置为只接收器模式时，必须在接收到倒数第二个数据之后将 CRCNEXT 置 1。

当传输由 DMA 管理时，此位应保持清零状态。

不适用于 I²S 模式。

位 11 **DFF**：数据帧格式 (Data frame format)

0：为发送/接收选择 8 位数据帧格式

1：为发送/接收选择 16 位数据帧格式

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。*

不适用于 I^2S 模式。

位 10 **RXONLY**：只接收模式使能 (Receive only mode enable)

该位用于使能通过一条双向线专门接收数据的单工通信。只接收模式激活时，使 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 IO 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：先发送 MSB

1：先发送 LSB

注： *正在通信时不应更改此位。*

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： ***该位不适用于 I^2S 模式。***

禁止 SPI 时，请按照第 29.3.10 节：禁止 SPI 的步骤中所述的步骤操作。

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。*

这些位不适用于 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。
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。
除了在 TI 模式下应用 CRC 的情况外，它不会用于 I^2S 模式和 SPI TI 模式。

位 0 **CPHA**：时钟相位 (Clock phase)
0：从第一个时钟边沿开始采样数据
1：从第二个时钟边沿开始采样数据
注：　正在通信时不应更改此位。
除了在 TI 模式下应用 CRC 的情况外，它不会用于 I^2S 模式和 SPI TI 模式。

29.7.2 SPI 控制寄存器 2 (SPI_CR2)

SPI 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.	TXEIE	RXNEIE	ERRIE	FRF	Res.	*SSOE*	TXDMAEN	RXDMAEN
								rw	rw	rw	rw		rw	rw	rw

位 15:8 保留，必须保持复位值。

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 OVR、CRCERR、MODF 和 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：　该位不适用于 ***I^2S 模式。***

位 3 保留。由硬件强制为零。

位 2 **SSOE**：SS 输出使能 (SS output enable)

0：在主模式下禁止 SS 输出，单元可在多主模式配置下工作。

1：在主模式下且使能单元后使能 SS 输出。不能在多主模式环境下工作。

*注：　该位不适用于 **I^2S 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 时，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 时，即产生 DMA 请求。

0：禁止接收缓冲区 DMA

1：使能接收缓冲区 DMA

29.7.3 SPI 状态寄存器 (SPI_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.	Res.	Res.	Res.	Res.	FRE	BSY	OVR	MODF	CRC ERR	UDR	CHSIDE	TXE	RXNE
							r	r	r	r	rc_w0	r	r	r	r

位 15:9 保留。由硬件强制为零。

位 8 **FRE**：帧错误 (Frame Error)

0：无帧错误

1：发生帧错误

该位由硬件置 1，在读取 SPI_SR 寄存器时由软件清零。

无论选择哪种音频协议，该位均在 SPI TI 模式或 I2S 模式下使用。检测从模式下，非预期时间上出现的 NSS 线或 WS 线变化，通知外部主器件和从器件之间的同步失效情况。

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 29.3.12 节：SPI 状态标志和第 29.3.10 节：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 29.3.13 节：SPI 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 898 页的第 29.4 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 SPI_RXCRCR 值匹配

1：接收到的 CRC 值与 SPI_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 29.6.8 节：I2S 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，在 PCM 模式下没有意义。

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

29.7.4 SPI 数据寄存器 (SPI_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。

数据寄存器分为 2 个缓冲区，一个用于写入（发送缓冲区），一个用于读取（接收缓冲区）。对数据寄存器执行写操作时，数据将写入发送缓冲区，从数据寄存器执行读取时，将返回接收缓冲区中的值。

注：　这些注适用于 SPI 模式：

发送或接收的数据为 8 位或 16 位，具体取决于数据帧格式选择位（SPI_CR1 寄存器中的 DFF）。必须在使能 SPI 前进行此项选择，以确保操作正确。

对于 8 位数据帧，缓冲区为 8 位，只有寄存器的 LSB (SPI_DR[7:0]) 用于发送/接收。在接收模式下，寄存器的 MSB (SPI_DR[15:8]) 强制为 0。

对于 16 位数据帧，缓冲区为 16 位，整个寄存器 SPI_DR[15:0] 均用于发送/接收。

29.7.5 SPI CRC 多项式寄存器 (SPI_CRCPR)（不用于 I^2S 模式）

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：　这些位不适用于 I^2S 模式。

29.7.6 SPI RX CRC 寄存器 (SPI_RXCRCR)（不用于 I^2S 模式）

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[15:0]															
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 SPI_CR1 寄存器中的 CRCEN 位写入 1 时，此寄存器复位。CRC 通过 SPI_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（SPI_CR1 的 DFF 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（SPI_CR1 寄存器的 DFF 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。

29.7.7 SPI TX CRC 寄存器 (SPI_TXCRCR)（不用于 I^2S 模式）

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 SPI_CR1 寄存器中的 CRCEN 位写入 1 时，此寄存器复位。CRC 通过 SPI_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（SPI_CR1 的 DFF 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（SPI_CR1 寄存器的 DFF 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。

29.7.8 SPI_I²S 配置寄存器 (SPI_I2SCFGR)

SPI_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.	ASTREN	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，必须保持复位值。

位 12 **ASTREN**：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（取决于所选的协议），I2S 从器件将启动传输。

1：使能异步启动。在从模式下使能 I2S 后，若从主器件接收到 I2S 时钟，I2S 从器件将立即启动传输，而无需在 WS 信号上检测到预期的边沿。

注： 注：使用 I2S Philips 标准时，适当的边沿为 WS 信号的下降沿；使用其他标准时，则为上升沿。

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 或 I²S 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 禁止时配置此位。
该位不适用于 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 903 页的第 29.6.3 节。不适用于 SPI 模式。*

注： 为确保正确运行，应在 I²S 禁止时配置这些位。

位 3 **CKPOL**：空闲状态时钟极性 (Steady state clock polarity)

0：I²S 时钟空闲状态为低电平

1：I²S 时钟空闲状态为高电平

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 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。

该位不适用于 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。*不适用于 SPI 模式。*

注： 为确保正确运行，应在 I²S 禁止时配置此位。

29.7.9 SPI_I²S 预分频器寄存器 (SPI_I2SPR)

SPI_I²S prescaler register

偏移地址：0x20

复位值：0000 0010 (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，必须保持复位值。

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 为主模式时，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 909 页的第 29.6.4 节。不适用于 SPI 模式。*

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。

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 909 页的第 29.6.4 节。不适用于 SPI 模式。*

注： 应在 I²S 禁止时配置这些位。只有在 I²S 为主模式时，才会使用此位。

29.7.10 SPI 寄存器映射

下表显示了 SPI 寄存器映射和复位值。

表 178. 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	**SPI_CR1**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BIDIMODE	BIDIOE	CRCEN	CRCNEXT	DFF	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	**SPI_CR2**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	RXNEIE	ERRIE	FRF	Res.	*SSOE*	TXDMAEN	RXDMAEN
	Reset value																								0	0	0	0		0	0	0
0x08	**SPI_SR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRE	BSY	OVR	MODF	CRCERR	UDR	CHSIDE	TXE	RXNE
	Reset value																								0	0	0	0	0	0	0	1	0
0x0C	**SPI_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	**SPI_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	**SPI_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	**SPI_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	**SPI_I2SCFGR**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ASTREN	I2SMOD	I2SE	I2SCFG	PCMSYNC	Res.	I2SSTD		CKPOL	DATLEN	CHLEN		
	Reset value																				0	0	0	0	0		0	0	0	0	0	0	0
0x20	**SPI_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 54 页的第 2.2.2 节*。

30 串行音频接口 (SAI)

30.1 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。

SAI 通过两个彼此完全独立的音频子模块来实现这种灵活性与可配置性。每个音频子模块与多达 4 个引脚（SD、SCK、FS 和 MCLK）相连。如果将两个子模块声明为同步模块，则其中一些引脚可以共用，从而可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。

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。

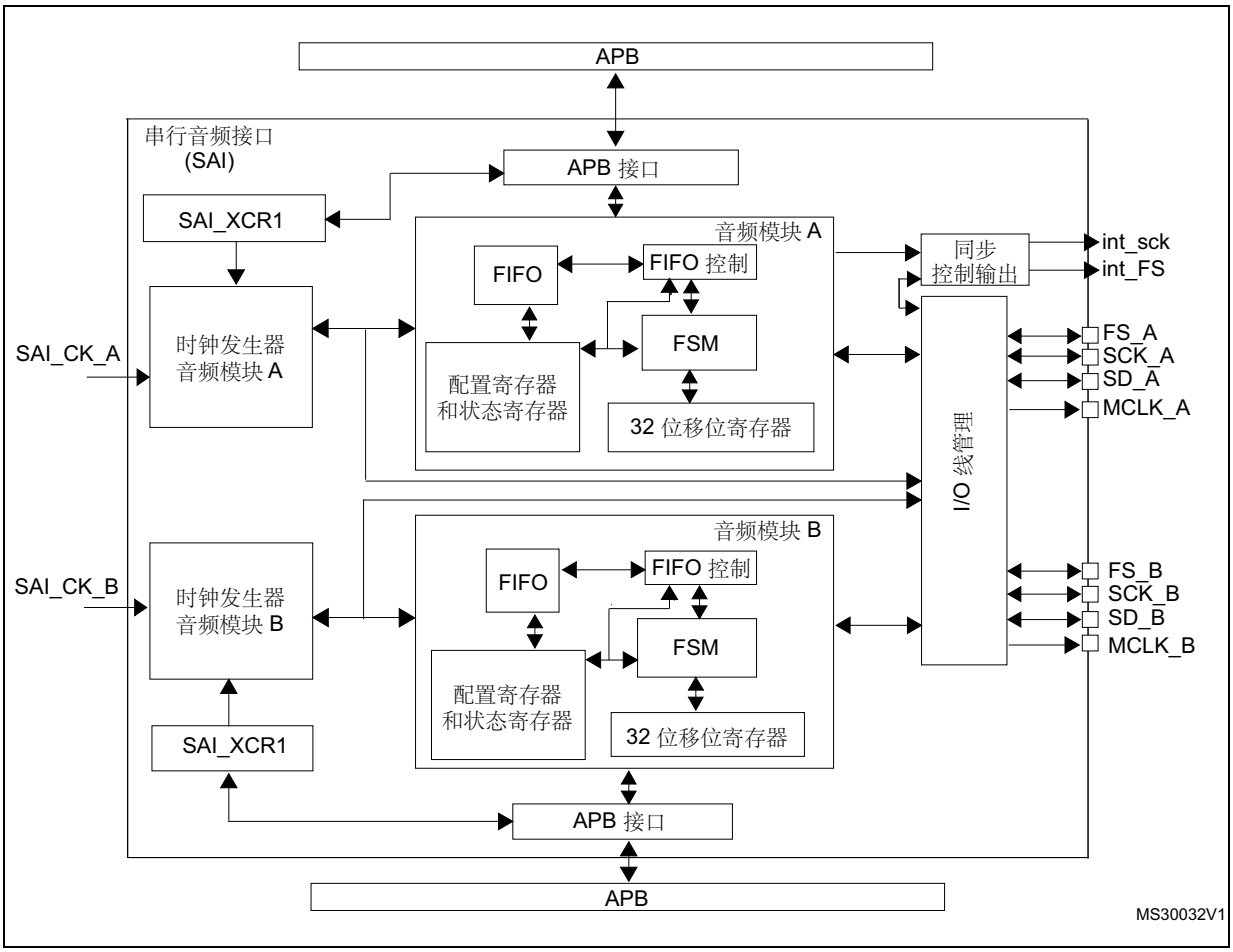
30.2 主要特性

- 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。
- 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主模式时，每个子模块的时钟发生器产生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。
- 外设的可配置性和灵活性高，支持以下音频协议：I2S、LSB 或 MSB 对齐、PCM/DSP、TDM 和 AC'97。
- 高达 16 个大小可配置的 Slot，可选择音频帧中的哪些 Slot 有效。
- 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 个专用通道，用于处理对每个 SAI 音频子模块的专用集成 FIFO 的访问。

30.3 功能框图

SAI 的框图如*图 351* 所示。

图 351. 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 中音频模块的各个专用引脚。如果两个模块同步，控制器将减少所使用 I/O 的数量，即释放 FS 引脚、SCK 引脚以及 MCLK 引脚，使它们作为通用 I/O。

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 生成位时钟 SCK 和帧同步信号，而从模式则意味着位时钟 SCK 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）。

30.4 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[0] 位配置为主模式或从模式。

在主模式下：

- SAI 使用时钟发生器在引脚 SCK_A 或 SCK_B（取决于具体哪个音频模块声明为 SAI 中的主模块）上产生位时钟。
- 专用引脚 SCK_x 被视为输出引脚。

在从模式下：

- 必须在使能主模式前使能从模式。
- 从音频模块若配置为在异步模式下工作，则其 SCK 时钟 I/O 引脚被视为输入引脚。
- 如果音频模块声明为与 SAI 中的第二个音频模块同步，则其 SCK I/O 引脚将用作通用 I/O，并从内部连接到将与其同步的器件的 SCK 引脚。

每个音频子模块均可通过相应 SAI_xCR1 寄存器中的 MODE[1] 位独立定义为发送器或接收器。I/O 引脚 SD 将分别定义为输出或输入。

可以在同一 SAI 中声明两个主音频模块，二者具有两种不同的 MCLK 和 SCK 时钟频率（但必须将两个模块声明为异步）。

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。

30.5 SAI 同步模式

内部同步

音频模块可声明为与第二个音频模块同步。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。声明为与另一个模块同步的音频模块将释放其 SCK_x、FS_x 和 MCLK_x 引脚以用作 GPIO。声明为异步的模块将使用其 I/O 引脚 FS_x、SCK_x 和 MCLK_x（如果该音频模块被视为主模块）。

通常，音频模块同步模式可用于在全双工模式下配置 SAI。两个音频模块的其中一个可配置为主模块，另一个为从模块；也可将二个均配置为从模块；一个模块声明为异步（SAI_xCR1 中的相应位 SYNCEN[1:0] = 00），另一个声明为同步（SAI_xCR1 中的相应位 SYNCEN[1:0] = 01）。

注： *由于存在内部重新同步阶段，APB 频率 PCLK 必须大于或等于比特率时钟频率的二倍。*

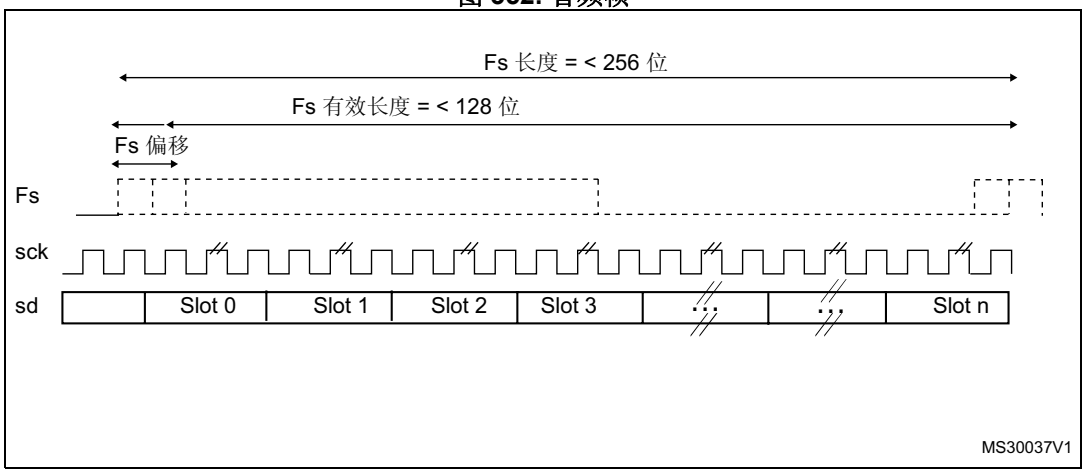
30.6 音频数据大小

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。

30.7 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。这一可配置性通过寄存器 SAI_xFRCR 来实现。*图 352* 说明这种灵活性。

图 352. 音频帧



在 AC'97 模式下（SAI_xCR1 寄存器中的位 PRTCFG[1:0] = 10），帧同步信号的波形被强制配置为支持这些协议。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。

30.7.1 帧长度

- 主模式：将 SAI_xFRCR 寄存器中的位 FRL[7:0] 置 1，可将音频帧的长度配置为最长 256 个位时钟。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 *第 30.12.4 节*）。在接收模式下，剩余位被忽略。
- 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数量。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。在这种情况下会产生错误。更多详细信息，请参见 *第 30.13 节*。

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。此时数据大小为 8 位且在 SAI_xSLOTR 寄存器的 NBSLOT[3:0] 中只定义了一个 Slot（对于 Slot 0，NBSLOT[3:0] = 0000）。

在主模式下：

- 如果 SAI_xCR 寄存器中的 NODIV 位清零，则帧长度应为 8 到 256 之间的一个等于 2 的 n 次幂的数。这是为了确保音频帧的每个位时钟包含整数个 MCLK 脉冲，这样可确保解码器内的外部 DAC/ADC 正确工作。如果 FRL[7:0] 中设置的值不遵循此规则，则会在音频模块使能时将 WCKCFG 标志置 1，并且会在 SAI_xIM 寄存器中的 WCKCFGIE 位置 1 时产生中断。SAI 自动禁止。
- 如果 SAI_xCR1 寄存器中的 NODIV 位置 1，则由于音频模块的输入时钟应等于位时钟，FRL[7:0] 位可能为任何值，而不受约束。不存在可输出的 MCLK_x 时钟。MCLK_x 输出自动禁止。

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。

30.7.2 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（见 *第 30.13 节*）。

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。如果前一个帧结束时 FIFO 中不存在数据，则将按照 *第 30.13 节*所述管理下溢条件，但音频通信流中不会出现中断。

30.7.3 帧同步有效电平长度

SAI_xFRCR 寄存器中的 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 SCK。

有效长度在 I2S、LSB 或 MSB 对齐模式下时为帧长的一半，在 PCM/DSP 或 TDM 模式下时为 1 位，而在 AC'97 模式下时甚至为 16 位。

30.7.4 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 位进行配置。

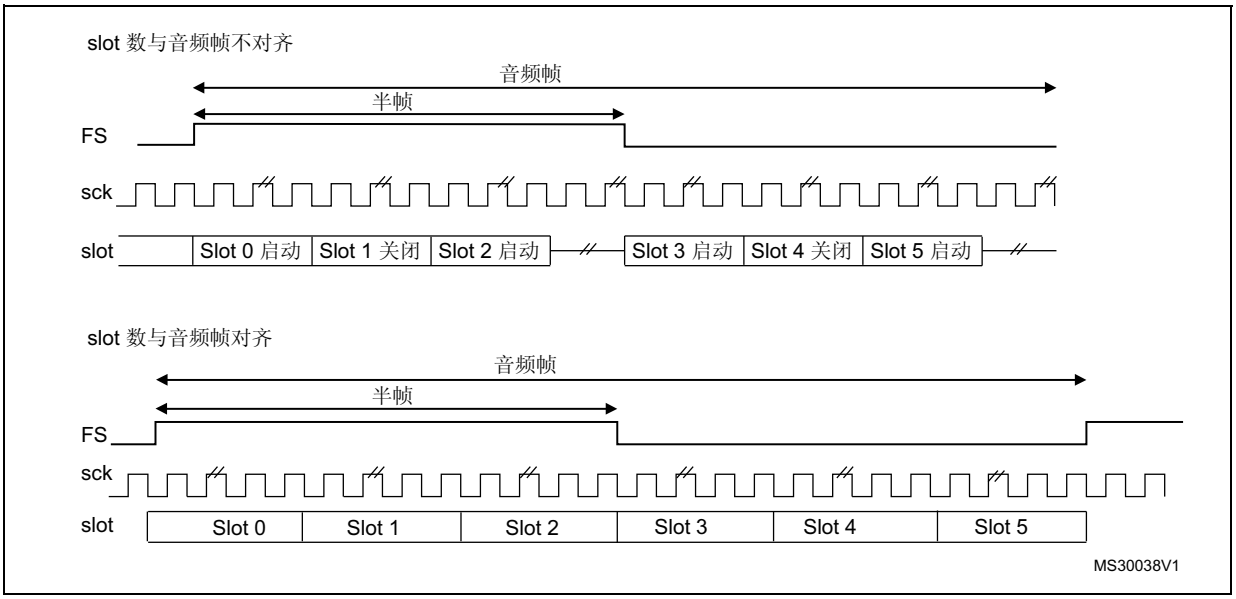
30.7.5 FS 信号的作用

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。该位可以为以下值：

- 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 353. FS 的作用是 SOF 信号 + 通道识别信号 (FSDEF = TRIS = 1)

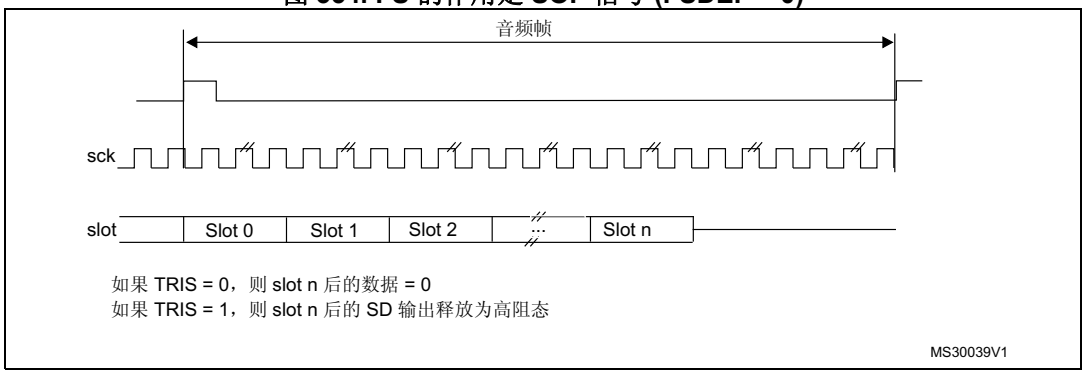


1. 帧长度应为偶数。

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，直至帧传输结束；
- 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 354. FS 的作用是 SOF 信号 (FSDEF = 0)



30.8 Slot 配置

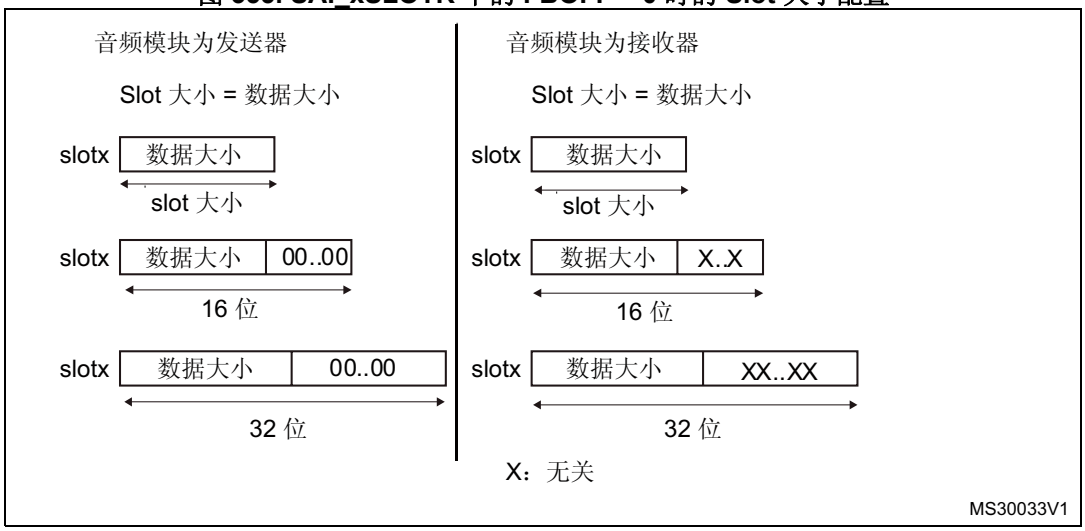
Slot 是音频帧中的基本元素。音频帧中的 Slot 数等于为 SAI_xSLOTR 寄存器中的 NBSLOT[3:0] 位配置的设置 + 1。
每个音频帧的最大 Slot 数固定为 16。

对于 AC'97 协议（位 PRTCFG[1:0] = 10 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 寄存器中的 SLOTEN[15:0] 位，可将各个 Slot 定义为有效 Slot 或无效 Slot。在音频帧中，传输一个无效 Slot 时，如果音频模块为发送器，数据线上将强制 0 值或 SD 数据线将释放为高阻态（见*第 30.12.4 节*），或者从此 Slot 结束后接收到的数据被忽略。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。

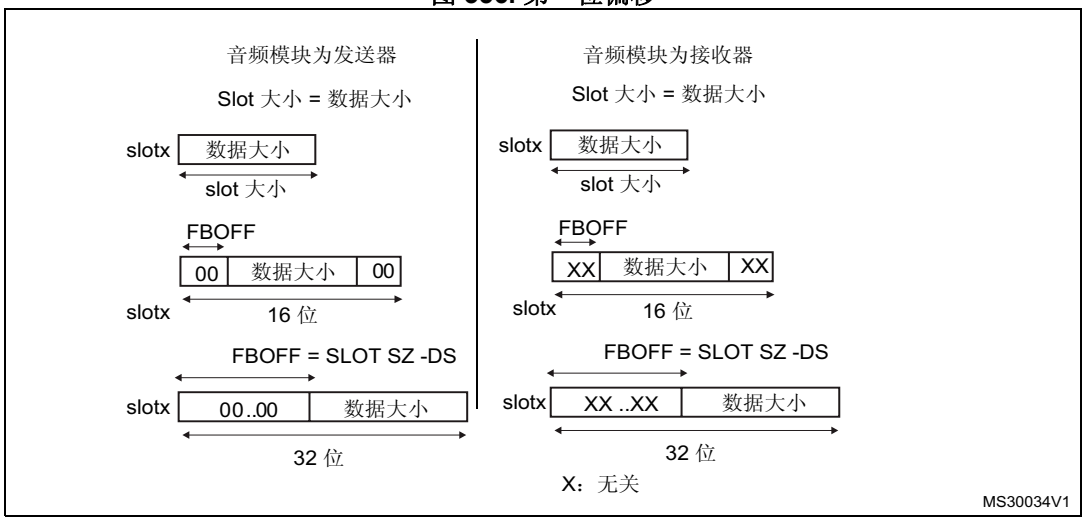
Slot 大小也可配置，如*图 355* 所示。通过将 SAI_xSLOTR 寄存器中的 SLOTSZ[1:0] 位置 1 来选择 Slot 大小。该大小适用于音频帧中的每个 Slot。

图 355. 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[5:0] 位进行配置。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 356. 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 数应为偶数。

在 AC'97（位 PRTCFG[1:0] = 10）中，Slot 大小按照*第 30.11 节*中的定义自动设置。

30.9 SAI 时钟发生器

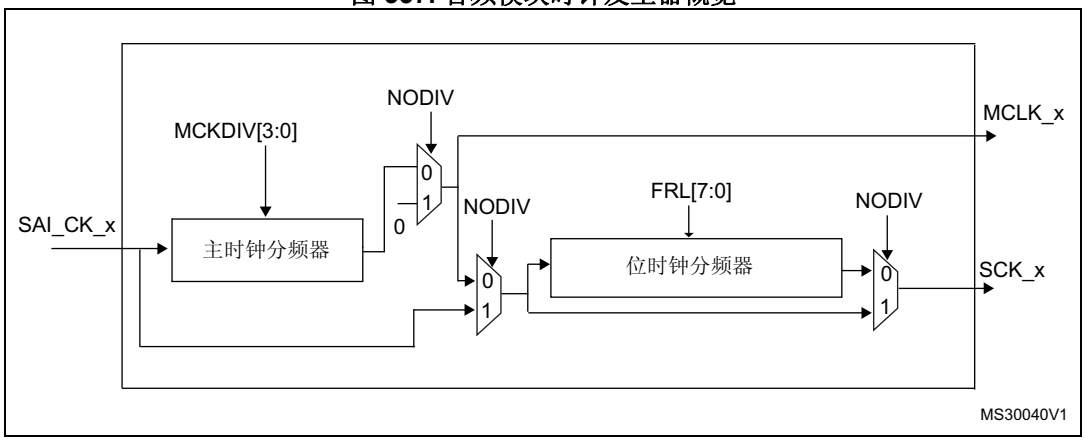
每个音频模块都有自己的时钟发生器，旨在使这两个模块完全独立。这两个时钟发生器的功能没有任何区别。它们完全一样。

当音频模块定义为主模块时，时钟发生器将产生通信时钟（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 357 给出了音频模块时钟发生器的架构。

图 357. 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 = 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 179* 所示。

表 179. 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] = 0100
	8 kHz	MCKDIV[3:0] = 1000
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 MCLK_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 - 1，在 SAI_xFRCR 寄存器中配置。

主模式下，(FRL[7:0] +1) 必须等于 2 的 n 次幂（见*第 30.7 节*），以使位时钟产生偶数个完整的 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。

30.10 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，将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_CR2 中的 FLTH 位）
- 通信方向为发送器还是接收器（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），如果数据寄存器 SAI_xDR 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 变为非空时（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 字包含一个音频帧。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。

30.11 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。

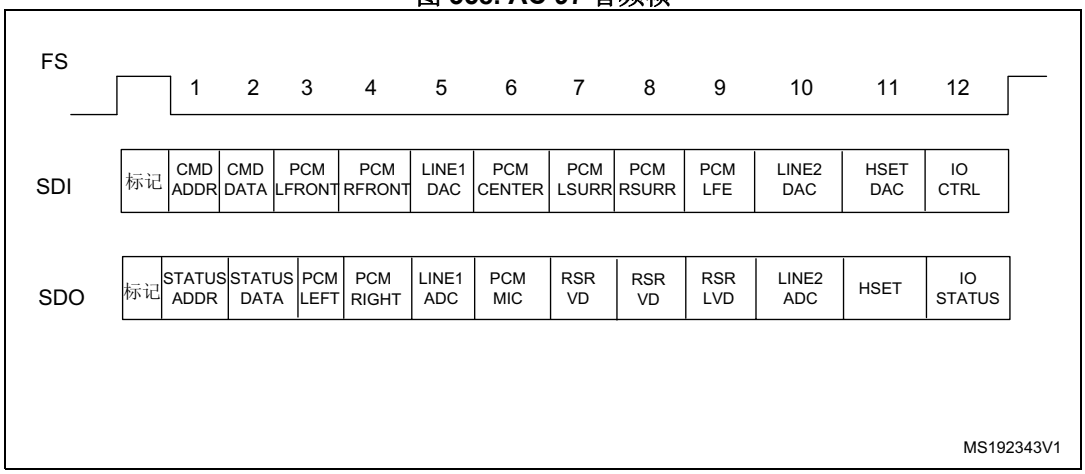
要选择此协议，可将 SAI_xCR1 寄存器中的 PRTCFG[1:0] 位置为 10。选择 AC’97 模式时，只能使用 16 位或 20 位的数据大小，否则将无法保证 SAI 运行正常。

- 因此，位 NBSLOT[3:0] 和 SLOTSZ[1:0] 将被忽略。
- Slot 数固定为 13。第一个 Slot 16 位宽，所有其他 Slot 均 20 位宽（数据 Slot）。
- SAI_xSLOTR 寄存器中的位 FBOFF[5:0] 被忽略。
- SAI_xFRCR 寄存器被忽略。

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 358 给出了 AC'97 音频帧的结构。

图 358. 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。

在接收模式下，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。

30.12 特性

根据所选的音频协议，SAI 可提供特定的一些实用功能。这些功能可通过 SAI_xCR2 寄存器中的特定位来访问。

30.12.1 静音模式

当音频模块用作发送器或接收器时，可使用静音模式。

发送器

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。SAI_xCR2 寄存器中的 MUTE 位若在帧传输期间置 1，将请求静音模式。

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。

如果在 SAI_xSLOTR 寄存器的 NBSLOT[3:0] 位中设置的 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。

接收器

在接收模式下，对于给定数量的连续音频帧（SAI_xCR2 寄存器中的位 MUTECNT[5:0]），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。计数器达到位 MUTECNT[5:0] 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再次生效。

30.12.2 MONO/STEREO 功能

在发送模式下，当 Slot 数等于 2（SAI_xSLOTR 中的 NBSLOT[3:0] = 0001）时，可支持单声道模式，而无需在存储器中进行任何数据处理。在这种情况下，由于发送时 Slot 0 的数据被复制到 Slot 1 中，FIFO 的访问操作将减少一半。

要选择单通道特性，可将 SAI_xCR1 寄存器中的 MONO 位置 1。

在接收模式下，位 MONO 可置 1 并且仅在 Slot 数等于 2（与发送模式相同）时有意义。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。如果接收时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，则位 MONO 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。

注： *要使能单声道模式，NBSLOT 和 SLOTEN 必须等于 2 且 MONO 位置 1。*

30.12.3 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。

软件可根据 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 359 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）。

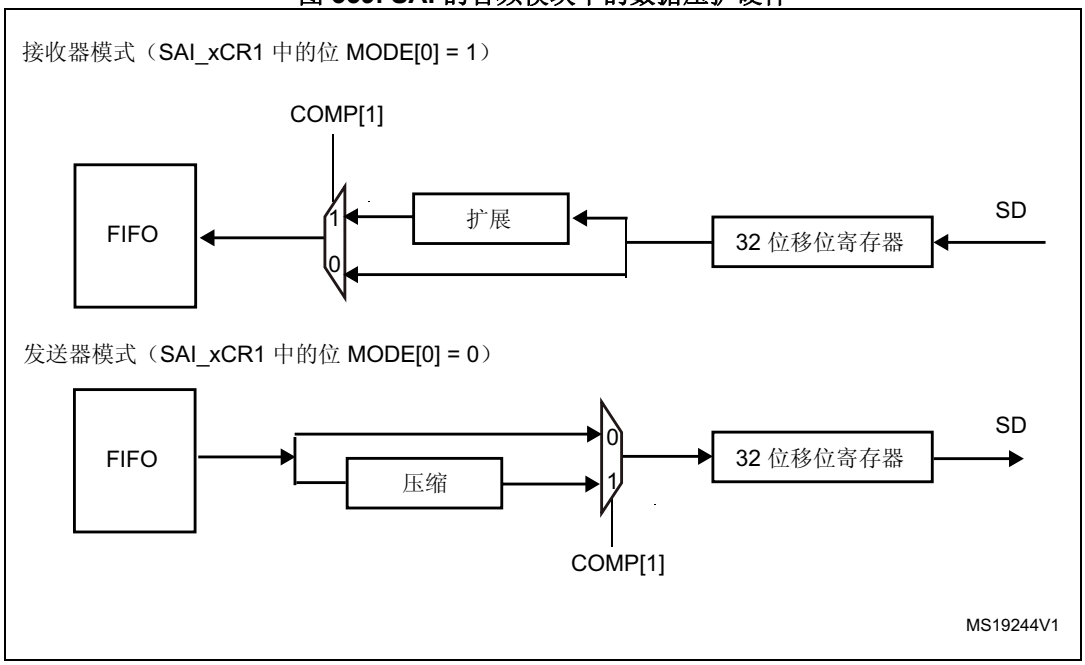
欧洲压扩标准是 A-Law，该标准允许 13 位动态范围（SAI_xCR2 寄存器中的 COMP[1:0] = 11）。

可根据 1 的补码或 2 的补码表示来计算压扩标准（μ-Law 或 A-Law），具体取决于 SAI_xCR2 寄存器中的 CPL 位设置。

μ-Law 和 A-Law 格式将数据解码为采用 MSB 对齐的 8 位代码元素。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 COMP[1:0] 位选择了这两个压扩模式之一时，SAI_xCR1 寄存器中的位 DS[2:0] 强制为 010。

如果无需压扩处理，则 SAI_xCR2 寄存器中的 COMP[1:0] 位应保持清零。

图 359. SAI 的音频模块中的数据压扩硬件



注： *选择 AC'97 时不适用。*

通过 SAI 配置自动选择扩展模式或压缩模式。

- 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。

30.12.4 无效 Slot 上的输出数据线管理

在发送模式下，在数据线上发送无效 Slot 时，可选择 SD 线的输出行为（当 SAI 禁止时通过 SAI_xCR2 寄存器中的 TRIS 位实现）。

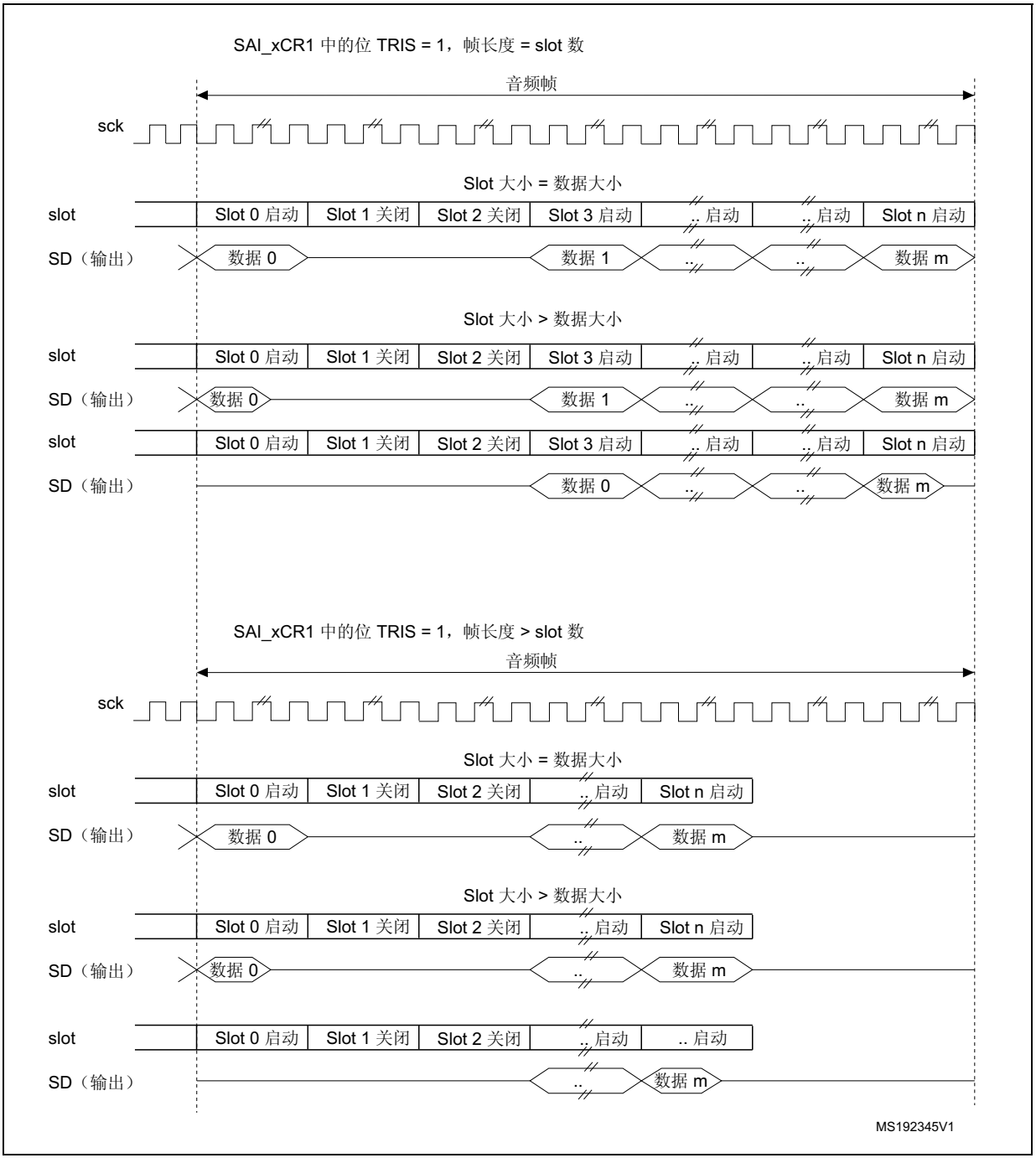
- 发送无效 Slot 时，SAI 将 SD 输出线强制为 0。
- 该输出线在最有一个数据位传输结束时释放为高阻态，为另一个连接此节点的发送器释放该数据线。

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。为了确保存在发送间隙，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 LSB 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。

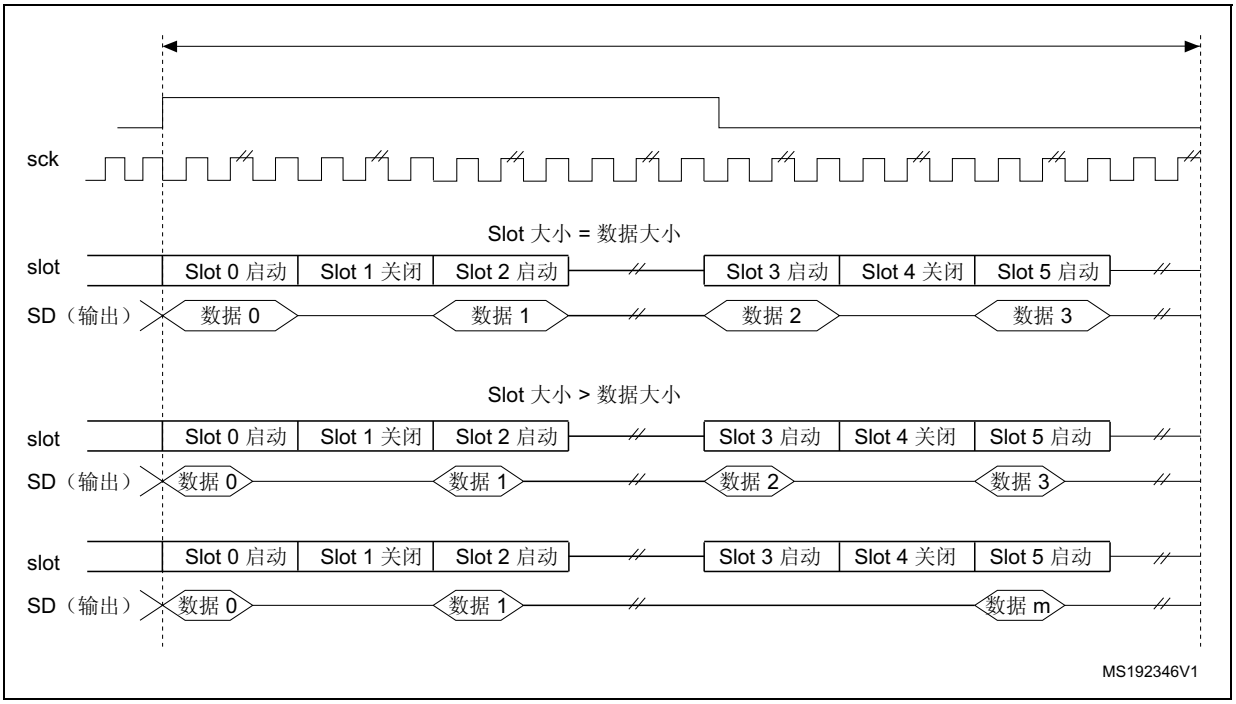
图 360 说明了这些行为。

图 360. 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 361* 管理三态模式（其中，SAI_xCR1 寄存器中的位 TRIS = 1，FSDEF = 1，半帧长 > Slot 数/2 且 NBSLOT = 6）。

图 361. 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 360* 和*图 361* 上的 SD 输出线上的所有高阻态将替换为使用值 0 驱动。

30.13 错误标志

SAI 内嵌有一些错误标志：

- 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

30.13.1 FIFO 上溢/下溢 (OVRUDR)

FIFO 上溢/下溢位是 SAI_xSR 寄存器中的 OVRUDR 位。

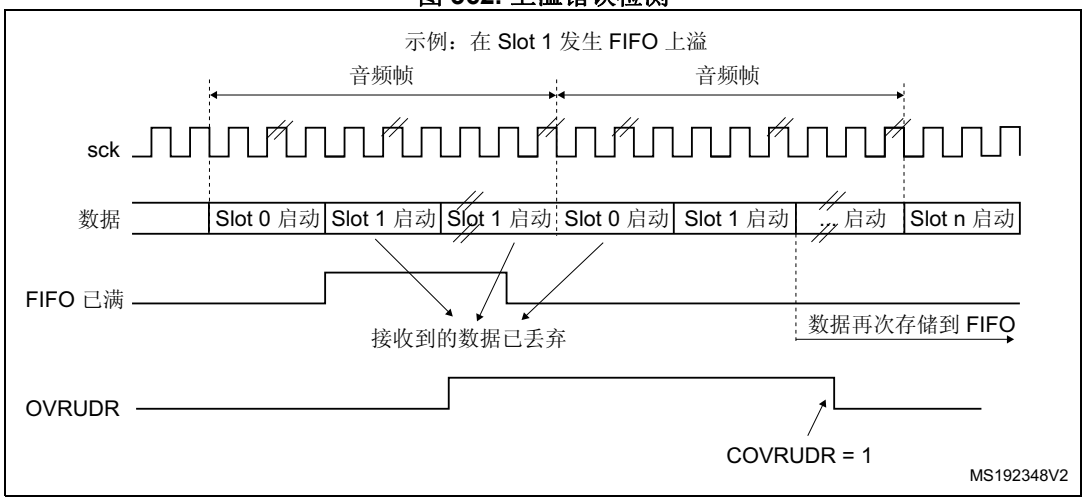
由于音频模块既可作为接收器，又可作为发送器，并且 SAI 中的每个音频模块都具有自己的 SAI_xSR 寄存器，因此上溢或下溢错误占用同一位。

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。内部将存储发生上溢时的 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存储的 Slot 编号开始存储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 362*）。

SAI_xCLRFR 寄存器中的 COVRUDR 位置 1 时将清除 OVRUDR 标志。

图 362. 上溢错误检测



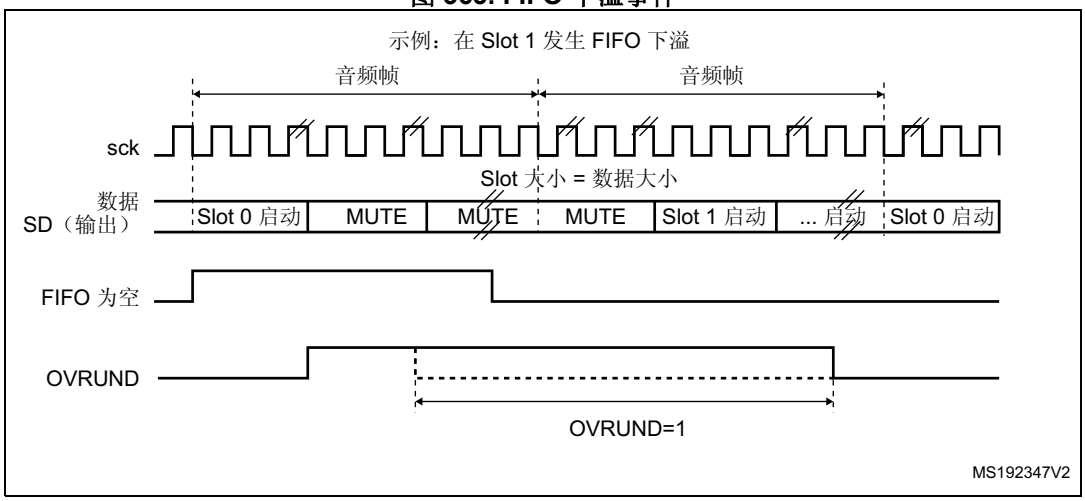
下溢

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（与音频模块配置为主模式还是从模式无关）。如果检测到下溢，则软件必须重新同步数据和 Slot。请按如下步骤操作：

1. 通过复位 SAI_xCR1 寄存器的 SAIEN 位禁止 SAI 外设。通过回读 SAIEN 位（SAIEN 应等于 0）检查是否已禁止 SAI。
2. 通过 SAI_xCR2 寄存器的 FFLUS 位刷新 Tx FIFO。
3. 在新帧的第一个有效 Slot 上重新分配要发送的正确数据。
4. 重新使能 SAI 外设（SAIEN 位置 1）。

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。

图 363. FIFO 下溢事件



30.13.2 帧同步提前检测 (AFSDET)

该 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，该标志用于告知是否比预期更早检测到帧同步 (FS) 信号。

出现提前检测时，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 寄存器的 AFSDETIE 位置 1，将产生中断。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之后重新与主时钟同步，应确保执行以下四个步骤：

1. 复位 SAI_xCR1 寄存器中的 SAIEN 位，禁止 SAI 模块，SAIEN 位等于 0（通过回读此位确定）可确保 SAI 已禁止。
2. 通过 SAI_xCR2 寄存器中的 FFLUS 位刷新 FIFO。
3. 重新使能 SAI 外设（SAIEN 位置 1），然后使能 SAI。
4. SAI 模块将等待 FS 使能以重新开始与主模块同步。

注： *AC’97 中不使能该标志，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。*

30.13.3 帧同步滞后检测

只有当 SAI 音频模块定义为从模块时，SAI_xSR 寄存器中的 LFSDET 位才可置 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（该信号生成过晚），SAI_xSR 寄存器中的 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 时该标志清零。

在检测到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（应确保按上述四个步骤操作）。

该检测和标志使能可检测到噪声环境中对 SCK 时钟的干扰，该检测由音频模块的状态机实现。存在这种干扰时，将使 SAI 音频模块状态机在当前音频帧中错误地从一个状态平移，从而破坏帧。

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 中不使能该标志，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。*

30.13.4 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 寄存器中的位 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 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，将使能该标志。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 。

要清除该标志，必须将 SAI_xCLRFR 寄存器中的 CCNRDY 位置 1。

30.13.5 主模式时钟配置错误 (NODIV = 0)

音频模块以主模式工作时（SAI_xCR1 寄存器中 MODE[1] = 0），如果 SAI_xCR1 中的 NODIV 位清零，在 SAI_xCR1 寄存器中的 SAIxEN 位置 1 时，若 SAI_xFRCR 中的 FRL[7:0] 位未置为合适值以遵循下列规则，则 WCKCFG 标志将置 1：

$$(FRL[7,0]) + 1 = 2^n$$

其中 n 的取值范围是 3 到 8。

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。

WCKCFG 位置 1 时，将自动禁止音频模块，并通过硬件将 SAI_xCR1 寄存器中的 SAIxEN 位清零。

以上公式用于保证音频帧的位时钟产生偶数个 MCLK 脉冲，并且位时钟生成的占空比为 50% 以保证良好的音响或音频采集质量。

30.14 中断源

SAI 有 7 个可能的中断源，如*表 180*所示。

表 180. 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_CR2 中的 FLTH 位） - 通信方向为发送器还是接收器 更多详细信息，请参见内部 FIFO 部分
OVRUDR	ERROR	主或从 接收器或发送器	SAI_xIM 寄存器中的 OVRUDRIE	COVRUDR = 1，在 SAI_xCLRFR 寄存器中
AFSDET	ERROR	从 （不适用于 AC’97 模式）	SAI_xIM 寄存器中的 AFSDETIE	CAFSDET = 1，在 SAI_xCLRFR 寄存器中
LFSDET	ERROR	从 （不适用于 AC’97 模式）	SAI_xIM 寄存器中的 LFSDETIE	CLFSDET = 1，在 SAI_xCLRFR 寄存器中
CNRDY	ERROR	从 （仅限 AC’97 模式）	SAI_xIM 寄存器中的 CNRDYIE	CCNRDY = 1，在 SAI_xCLRFR 寄存器中
MUTEDET	MUTE	主或从 仅限接收模式	SAI_xIM 寄存器中的 MUTEDETIE	CMUTEDET = 1，在 SAI_xCLRFR 寄存器中
WCKCFG	ERROR	主模式且 SAI_xCR1 寄存器中的 NODIV = 0	SAI_xIM 寄存器中的 WCKCFGIE	CWCKCFG = 1，在 SAI_xCLRFR 寄存器中

以下是出现中断时要遵守的 SAI 配置步骤：

1. 禁止 SAI 中断。
2. 配置 SAI。
3. 配置 SAI 中断源。
4. 使能 SAI。

30.15 禁止 SAI

可随时通过清零 SAI_xCR1 寄存器中的 SAIxEN 位禁止 SAI 中的音频模块。所有已开始的帧将在 SAI 完全关闭之前自动完成。SAI_xCR1 寄存器中的 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的音频模块，则必须先禁止以主模式工作的音频模块。

30.16 SAI DMA 接口

为了减轻 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)。每个音频通道都有一个遵循简单 DMA 请求/应答协议的 DMA 通道。

要配置音频模块以通过 DMA 接口传输数据，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，请参见内部 FIFO 部分）。DMA 方向与 SAI 音频模块配置相关：

- 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。

以下是使用 DMA 时要遵守的 SAI 配置步骤：

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

注： *配置 SAI 模块前，必须禁止 SAI DMA 通道。*

30.17 SAI 寄存器

30.17.1 SAI x 配置寄存器 1 (SAI_xCR1)，其中 x 为 A 或 B

SAI xConfiguration 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
Reserved								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
Reserved		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。

0000：主时钟输入 1 分频。

否则，主时钟频率将根据以下公式计算：

MCLK_x = SAI_CK_x/(MCKDIV[3:0] * 2)

音频模块为从模块时，这些位没有意义。

必须在音频模块禁止的情况下配置这些位。

位 19 **NODIV**：无分频器 (No divider)。此位由软件置 1 和清零。

0：使能主时钟分频器

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 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。由硬件将该位清零，通过软件将其禁止（位中写入低电平）后，音频将完全禁止（等待当前帧结束）。

0：禁止音频模块

1：使能音频模块，仅当写操作期间该位为 0 时才可将其置 1（即 SAI 在重新使能前被完全禁止）。该位可用以控制音频模块的状态。如果在音频帧中禁止该位，则仍将完成正在进行的传输并且该单元在相应音频帧传输结束时完全禁止。

注：　当 SAIx 模块配置为主模式时，SAI 的输入中必须有时钟，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 SAIEN 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。
0：立体声模式
1：单声道模式。
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*第 30.12.2 节*。

位 11:10 **SYNCEN[1:0]**：同步使能 (Synchronization enable)。此位由软件置 1 和清零。
00：音频模块异步。
01：音频模块与另一个内部音频模块同步。这种情况下，应将音频模块配置为从模式。
10：保留。
11：未使用
必须在音频模块禁止的情况下配置这些位。

位 9 **CKSTR**：时钟选通边沿 (Clock strobing edge)。此位由软件置 1 和清零。
0：数据选通边沿是 SCK 的下降沿
1：数据选通边沿是 SCK 的上升沿
在音频模块禁止时必须配置该位。

位 8 **LSBFIRST**：最低有效位优先 (Least significant bit first)。此位由软件置 1 和清零。
0：传输数据时，数据的 MSB 位优先。
1：传输数据时，数据的 LSB 位优先。
在音频模块禁止时必须配置该位。
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。

位 7:5 **DS[2:0]**：数据大小 (Data size)。这些位由软件置 1 和清零。
000：未使用
001：未使用
010：8 位
011：10 位
100：16 位
101：20 位
110：24 位
111：32 位
选择压扩模式时（COMP[1:0] 位），将忽略这些 DS[1:0]，原因是算法本身将数据大小固定为 8 位模式。
必须在音频模块禁止的情况下配置这些位。
注： 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。

位 4 保留，始终读为 0。

位 3:2 **PRTCFG[1:0]**：协议配置 (Protocol configuration)。这些位由软件置 1 和清零。
00：自由协议
01：未使用
10：AC’97 协议
11：未使用
自由协议选项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...），从而对大部分配置寄存器位以及帧配置寄存器进行设置。
必须在音频模块禁止的情况下配置这些位。

位 1:0 **MODE[1:0]**：音频模块模式 (Audio block mode)。这些位由软件置 1 和清零。
00：主机发送
01：主机接收
10：从机发送
11：从机接收
必须在音频模块禁止的情况下配置这些位。
注： 在主机发送模式下，音频模块将开始生成 FS 和时钟。

30.17.2 SAI x 配置寄存器 2 (SAI_xCR2)，其中 x 为 A 或 B

SAI xConfiguration 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
Reserved															
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	FFLUS	FTH		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	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)。这些位由软件置 1 和清零。
00：不支持压扩算法
01：保留。
10：μ-Law 算法
11：A-Law 算法
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 *ComPLement 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*第 30.12.3 节*。
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*第 30.12.1 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。
0：MUTE 模式期间发送位值 0。
1：MUTE 模式期间发送上一个值。
仅当音频模块用作发送器并且 Slot 数小于或等于 2，并且 MUTE 位已置 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 发送的静音值将是上一帧期间发送的值。
更多详细信息，请参见*第 30.12.1 节*。

位 5 **MUTE**：静音 (Mute)。此位由软件置 1 和清零。
0：禁止静音模式。
1：使能静音模式。
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*第 30.12.1 节*。

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。
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）。
仅当音频模块用作发送器时该位才有意义。
应在 SAI 禁止时配置此位。
更多详细信息，请参见*第 30.12.4 节*。

位 3 **FFLUSH**：FIFO 刷新 (FIFO flush)。该位由软件置 1， 并始终读为低电平。
0：禁止 FIFO 刷新。
1：FIFO 刷新。
向该位写入 1 时会触发 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 中的数据将丢失（发送或接收数据不会继续丢失）。
应在 SAI 禁止时配置此位。
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

30.17.3 SAI x 帧配置寄存器 (SAI_XFRCR)，其中 x 为 A 或 B

SAI xFrame 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 音频协议无意义。*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													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	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。
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。
该位对 AC'97 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 17 **FSPOL**：帧同步极性 (Frame synchronization polarity)。该位由软件置 1 和清零。
0：FS 为低电平有效（下降沿）
1：FS 为高电平有效（上升沿）
该位用于配置 FS 信号上的 SOF 电平。
该位对 AC'97 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 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
该位置 1 时，SAI_ASLOTR 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，其他 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）。
该位对 AC'97 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
这些位的设置值用于指定音频帧中 FS 信号的有效电平长度，以位时钟数 (SCK) + 1 (FSALL[6:0] + 1) 计算。
这些位对 AC'97 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)。这些位由软件置 1 和清零。
它们定义音频帧的长度。更确切的说是这些位定义每个音频帧的 SCK 时钟数。
帧中的位数等于 FRL[7:0] + 1。
音频帧中发送的位数要大于或等于 8，否则音频模块将出现操作异常。数据大小为 8 位且在 SAI_ASLOTR 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，如果主时钟 MCLK_x 引脚声明为输出引脚，则应为 8 到 256 之间的一个等于 2 的几次幂的数，以便在音频帧中确保位时钟的 MCLK 脉冲数为整数，从而保证解码器中的外部 DAC/ADC 能正常工作。
帧长度应为偶数。
这些位对 AC'97 音频模块配置无意义，从而也不使用。

30.17.4 SAI x Slot 寄存器 (SAI_xSLOTR)，其中 x 为 A 或 B

SAI xSlot 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 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
Reserved				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)。这些位由软件置 1 和清零。
SLOTEN 位中的每一位都标识从 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配置这些位。
在 AC'97 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)。这些位由软件置 1 和清零。
这些位寄存器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_AFRCR 寄存器中的 FSDEF 位置 1 时，Slot 数应为偶数。
如果 Slot 数大于数据大小，则当 SAI_xCR1 寄存器中的 TRIS 位清零时，剩余的位将强制为 0；否则在下一个 Slot 有效时强制为 0，或者在下一个 Slot 无效且 TRIS = 1 时，SD 线将强制为高阻态。
必须在音频模块禁止的情况下配置这些位。
在 AC'97 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。
00：Slot 大小与数据大小（在 SAI_ACR1 寄存器的 DS[3:0] 位中指定）相当。
01：16 位
10：32 位
11：保留
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。
必须在音频模块禁止的情况下配置这些位。
在 AC'97 模式下会忽略这些位。

位 5 保留，始终读为 0。

位 4:0 **FBOFF[4:0]**：第一个位偏移 (First bit offset)
这些位由软件置 1 和清零。
这些位中设置的值表示 Slot 中第一个数据传输位的位置。它表示一个偏移值。在此偏移阶段，在发送模式下数据线发送的值为 0。在接收模式下，将丢弃偏移阶段接收到的位。
必须在音频模块禁止的情况下配置这些位。
在 AC'97 模式下会忽略这些位。

30.17.5 SAI x 中断屏蔽寄存器 2(SAI_xIM)，其中 x 为 A 或 B

SAI xInterrupt mask register2

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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									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_ASR 寄存器中的 LFSDET 位置 1，则生成中断。
该位对于 AC'97 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_ASR 寄存器中的 AFSDET 位置 1，则生成中断。
该位对于 AC'97 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_ASR 寄存器中的 CNRDY 标志将置 1 并生成中断。

仅当选择了 AC97 模式（位 PRTCFG[1:0]）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_ASR 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以发送模式工作。

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

仅当音频模块配置为主模块（SAI_ACR1 寄存器中 MODE[1] = 0）且 SAI_xCR1 寄存器中的位 NODIV = 0 时才考虑该位。

该位在 SAI_ASR 寄存器中的 WCKCFG 标志置 1 时生成中断。

注： 该位仅用于 TDM 模式，其他模式下没有意义。

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。

30.17.6 SAI x 状态寄存器 (SAI_xSR)，其中 x 为 A 或 B

SAI xStatus 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
Reserved													FLTH		
													r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									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 模块配置为接收器：
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 模式。
该位在 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 模式。
该位在 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 音频模式并且音频模块配置为接收器模式时，才使用该位。

该位在 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。

如果音频模块配置为发送模式，则 FIFO 请求涉及向 SAI_xDR 中写入。

如果音频模块配置为接收模式，则 FIFO 请求涉及从 SAI_xDR 中读取。

该标志会在 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 30.7 节*中定义的帧长度规范（SAI_x FRCR 寄存器中 FRL[7:0] 位的配置）

仅当音频模块为主模块（SAI_xCR1 寄存器中 MODE[1] = 0）且 SAI_xCR1 寄存器中 NODIV = 0 时，才使用此位。

该位在 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）

如果在某个音频帧的每个 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。

该位在 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。

仅当音频模块配置为接收模式时，才会出现上溢错误。

仅当音频模块配置为发送模式时，才会出现下溢错误。

该位在 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。

30.17.7 SAI x 清除标志寄存器 (SAI_xCLRFR)，其中 X 为 A 或 B

SAI xClear 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									CLFSDET	CAFSDET	CCNRDY	Reserved	CWCKCFG	CMUTEDET	COVRUDR
									rw	rw	rw		rw	rw	rw

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。
向该位写入 1 可清除 SAI_xSR 寄存器中的 LFSDET 标志。
不适用于 AC'97 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。
向该位写入 1 可清除 SAI_xSR 寄存器中的 AFSDET 标志。
不适用于 AC'97 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。
向该位写入 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。
向该位写入 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（SAI_ACR1 寄存器中 MODE[1] = 0）且 SAI_xCR1 寄存器中的位 NODIV = 0 时，才使用此位。
读取该位将始终返回值 0。

位 1 **CMUTEDET**：清除静音检测标志 (Clear Mute detection flag)。该位为只写位。
向该位写入 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。
向该位写入 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。

30.17.8 SAI x 数据寄存器 (SAI_xDR)，其中 x 为 A 或 B

SAI xData 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，写入该寄存器的效果是向 FIFO 加载数据。

若 FIFO 非空，读取该寄存器的效果是从 FIFO 取走数据。

30.17.9 SAI 寄存器映射

下表对 SAI 寄存器进行了汇总。

表 181. 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
0x0004 or 0x0024	SAI_xCR1	Reserved								MCKDIV[3:0]				NODIV	Res.	DMAEN	SAIxEN	Reserved.		OutDri	MONO	SYNCEN[1:0]		CKSTR	LSBFIRST	DS[2:0]			Res.	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 or 0x0028	SAI_xCR2	Reserved																COMP[1:0]		CPL	MUTECNT[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	0
0x000C or 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
0x0010 or 0x0030	SAI_xSLOTR	SLOTEN[15:0]																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

表 181. 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
0x0014 or 0x0034	SAI_xIM	Reserved																									LFSDET	AFSDETIE	CNRDYIE	FREQIE	WCKCFG	MUTEDET	OVRUDRIE
	Reset value																										0	0	0	0	0	0	0
0x0018 or 0x0038	SAI_xSR	Reserved													FLVL[2:0]			Reserved									LFSDET	AFSDET	CNRDY	FREQ	WCKCFG	MUTEDET	OVRUDR
	Reset value														0	0	0										0	0	0	0	1	0	0
0x001C or 0x003C	SAI_xCLRFR	Reserved																									LFSDET	CAFSDET	CNRDY	Res.	WCKCFG	MUTEDET	OVRUDR
	Reset value																										0	0	0		0	0	0
0x0020 or 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*存储器映射部分*。

31 安全数字输入/输出接口 (SDIO)

31.1 SDIO主要特性

SD/SDIO MMC 卡主机接口 (SDIO) 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。

SDIO 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 50 MHz
- 数据和命令输出使能信号，控制外部双向驱动器。

注： *1* *SDIO 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，不被仅 SD I/O 卡或者复合卡的 I/O 支持。部分上述命令（如擦除命令）在 SD I/O 器件中不使用，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 SDIO 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。

31.2 SDIO 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 364. “无响应”和“无数据”操作

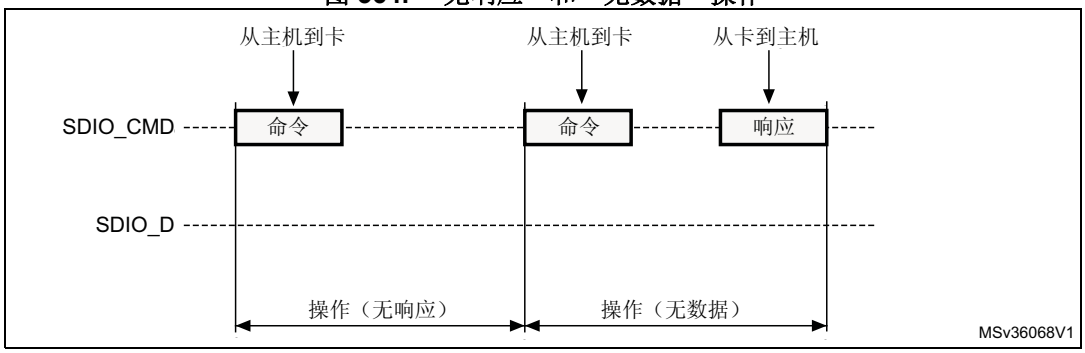


图 365. （多个）块读取操作

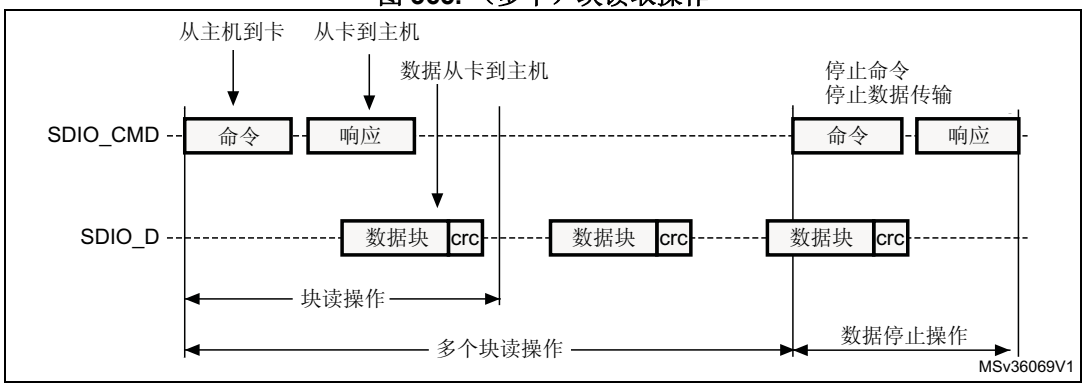
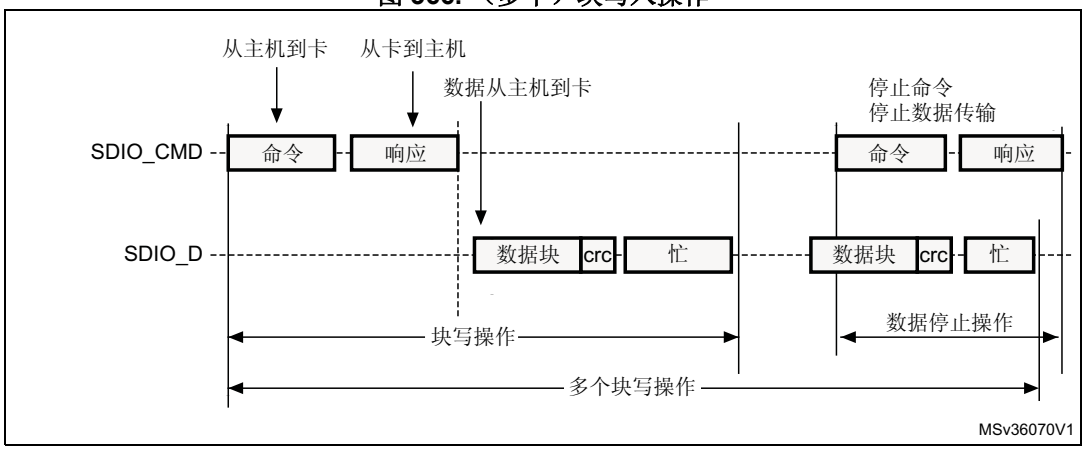


图 366. （多个）块写入操作



注：　只要发出“繁忙”信号（SDIO_D0 被拉到低电平），SDIO 便不会发送任何数据。

图 367. 连续读取操作

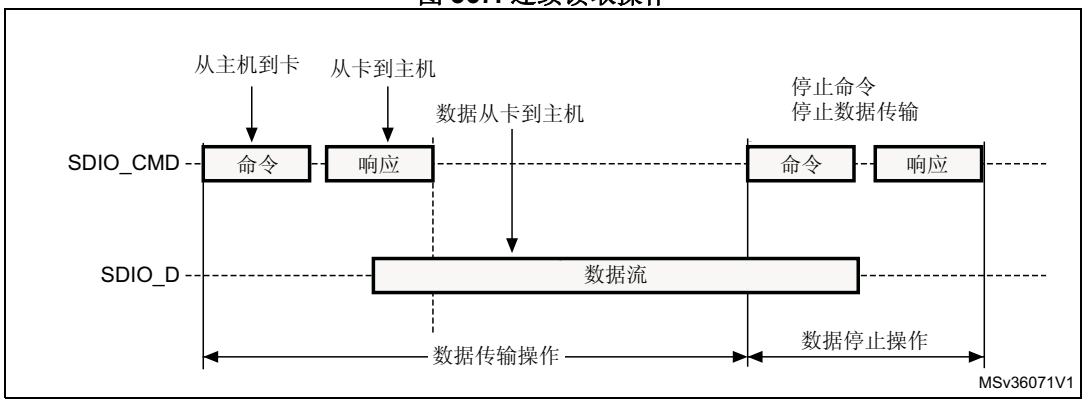
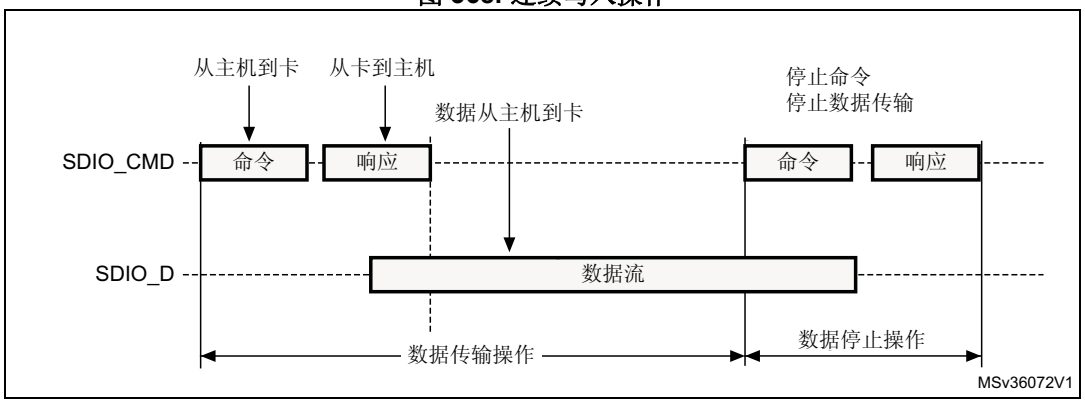


图 368. 连续写入操作

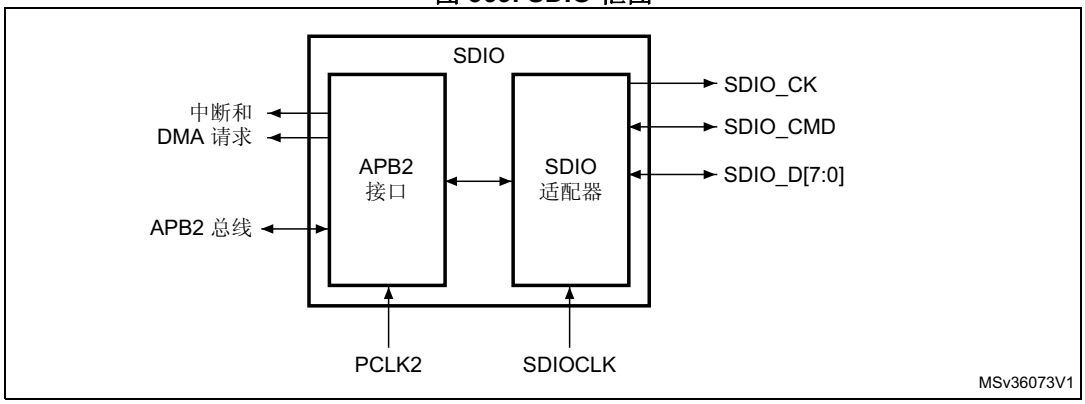


31.3 SDIO 功能描述

SDIO 由两部分组成：

- SDIO 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 SDIO 适配器寄存器，并且生成中断和 DMA 请求信号。

图 369. SDIO 框图



默认情况下，SDIO_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 SDIO_D0、SDIO_D[3:0] 或 SDIO_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 SDIO_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 SDIO_D0 或 SDIO_D[3:0]。所有数据线均以推挽模式运行。

SDIO_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 还将推挽驱动器用于初始化）

SDIO_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。

SDIO 使用两个时钟信号：

- SDIO 适配器时钟 (SDIOCLK = 50 MHz)
- APB2 总线时钟 (PCLK2)

PCLK2 和 SDIO_CK 时钟频率必须满足以下条件：

$$频率(PCLK2) > ((3x宽度)/32) \times 频率(SDIO\_CK)$$

表 182 中的信号在多媒体卡/SD/SD I/O 卡总线中使用。

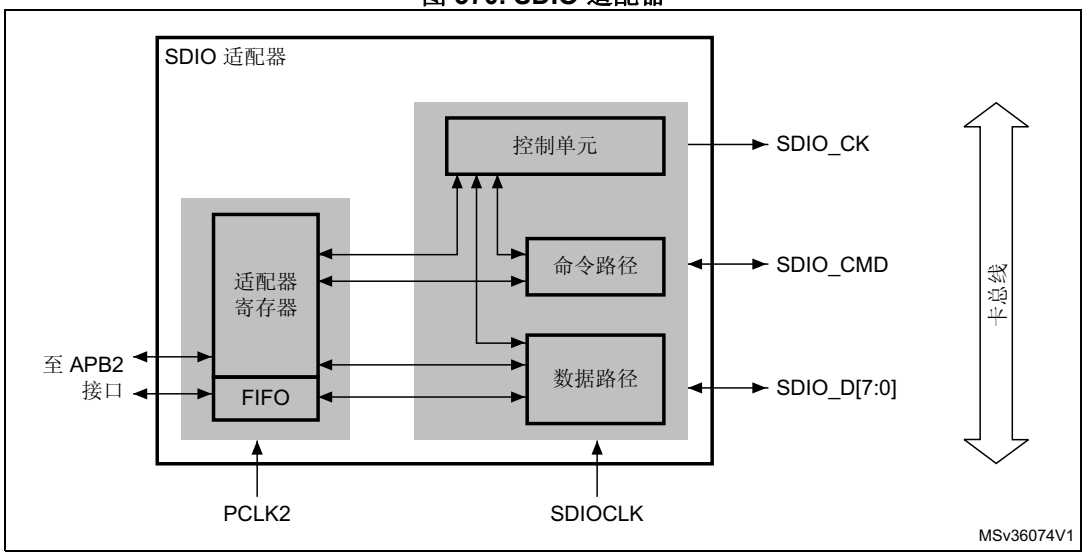
表 182. SDIO I/O 定义

引脚	方向	说明
SDIO_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。
SDIO_CMD	双向	多媒体卡/SD/SDIO 卡命令。该引脚是双向命令/响应信号。
SDIO_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。

31.3.1 SDIO 适配器

图 370 显示了 SDIO 适配器的简化框图。

图 370. SDIO 适配器



SDIO 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 SDIO 适配器时钟域 (SDIOCLK)。*

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 SDIO 清零寄存器中对应位的单元时，将生成清零信号。

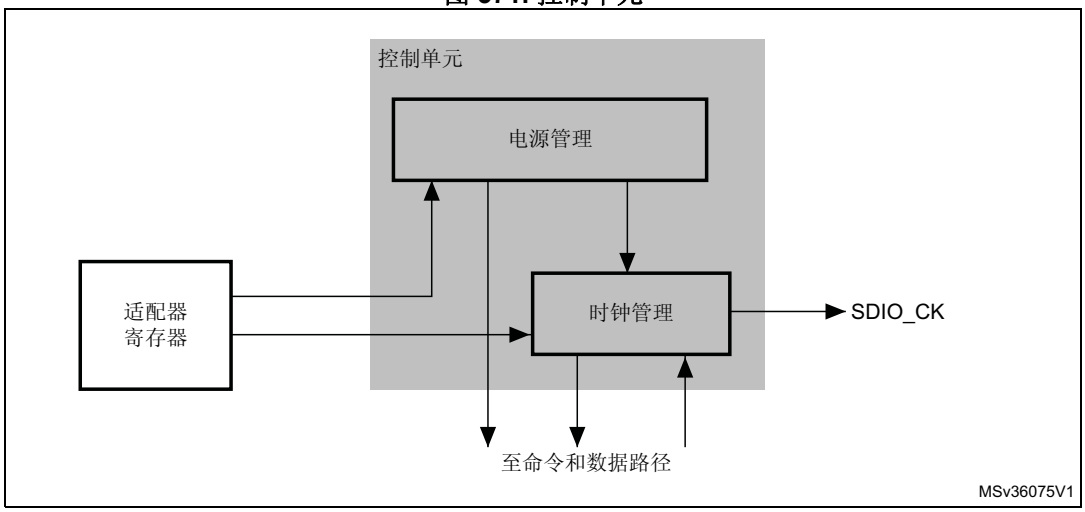
控制单元

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 371. 控制单元



控制单元如*图 371* 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 SDIO_CK 信号。SDIO_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：

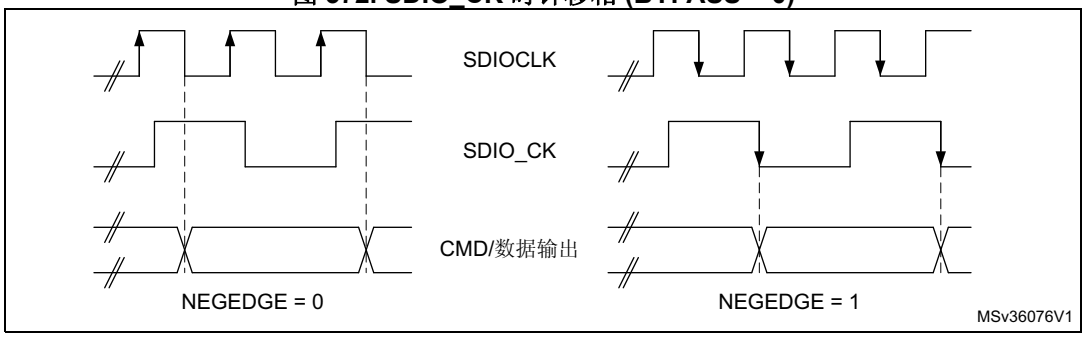
- 复位后
- 在断电或上电阶段中
- 在使能了节电模式并且卡总线处于空闲状态的情况下（命令和数据路径子单元进入空闲阶段后的八个时钟周期之后）

时钟管理子单元负责控制 SDIO_CK 移相。如果未处于旁路模式下，则当 SDIO_CLKCR[13] 位复位 (NEGEDGE = 0) 时，将在 SDIO_CK 上升沿后的 SDIOCLK 下降沿生成 SDIO 命令和数据输出（SDIO_CK 上升沿在 SDIOCLK 上升沿时出现）。当 SDIO_CLKCR[13] 位置 1 (NEGEDGE = 1) 时，将在 SDIO_CK 下降沿更改 SDIO 命令和数据。

当 SDIO_CLKCR[10] 置 1 (BYPASS = 1) 时，SDIO_CK 上升沿在 SDIOCLK 上升沿时出现。无论 NEGEDGE 值为多少，都将在 SDIOCLK 下降沿更改数据和命令。

数据和命令响应通过 SDIO_CK 上升沿来锁存。

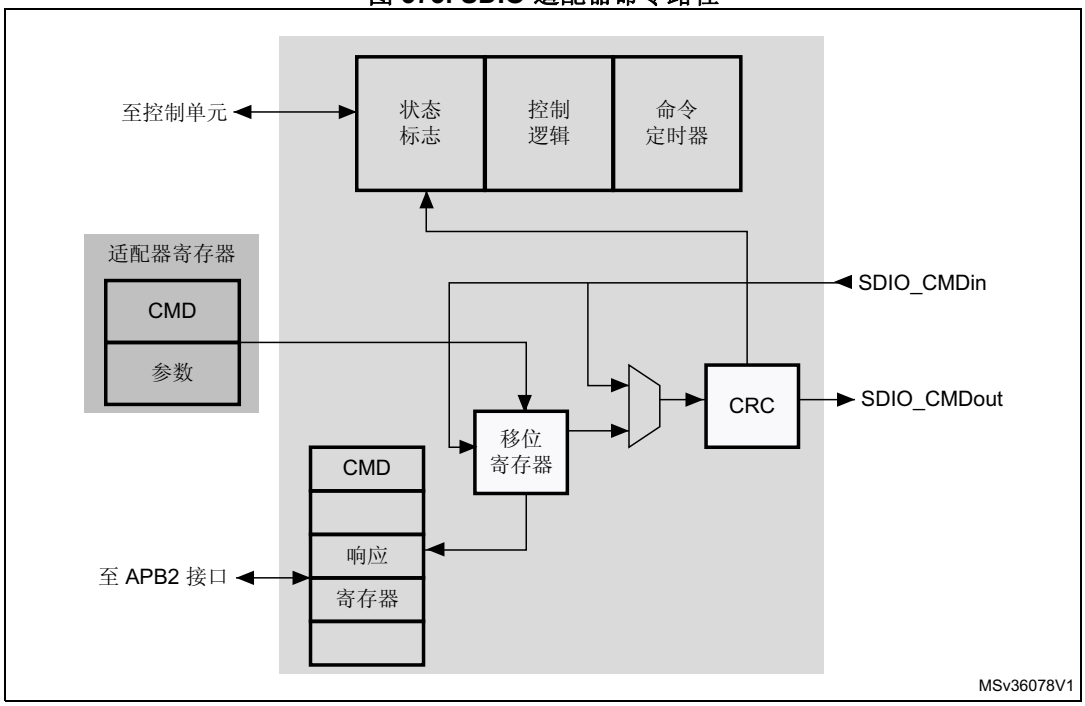
图 372. SDIO_CK 时钟移相 (BYPASS = 0)



命令路径

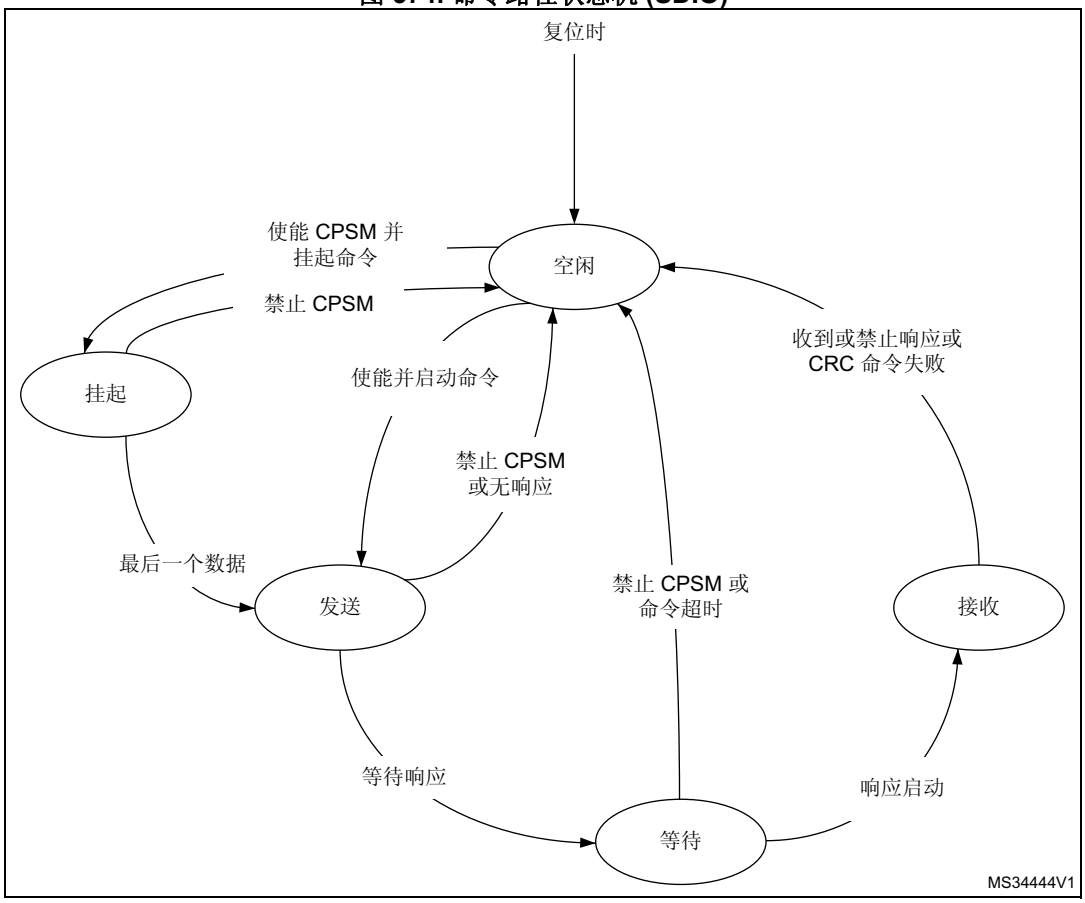
命令路径单元向卡发送命令并从卡接收响应。

图 373. SDIO 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 968 页的图 374*）。收到响应时，将比较接收的 CRC 代码和内部生成代码，并且将相应的状态标志置 1。

图 374. 命令路径状态机 (SDIO)



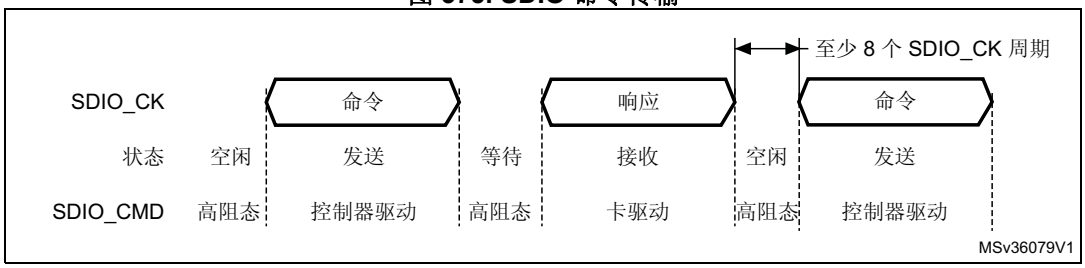
进入等待状态后，命令计时器便开始运行。如果 CPSM 尚未变为接收状态便已达到超时，则将超时标志置 1 并且进入空闲状态。

注： *命令超时为固定值：64 个 SDIO_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 SDIO_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 375. SDIO 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 183* 中显示了多媒体卡、SD 存储卡和 SDIO 卡的命令令牌的通用格式。

    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 SDIO_CMD 输出处于高阻态，如*第 969 页的图 375* 中所示。SDIO_CMD 上的数据与 SDIO_CK 上升沿同步。*表 183* 给出了命令格式。

表 183. 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
[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。

SDIO 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 184. 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
[39:8]	32	-	参数
[7:1]	7	-	CRC7（或 1111111）
0	1	1	结束位

表 185. 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 1001 页的第 31.8.4 节*）。命令路径实施*表 186*中显示的状态标志：

表 186. 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

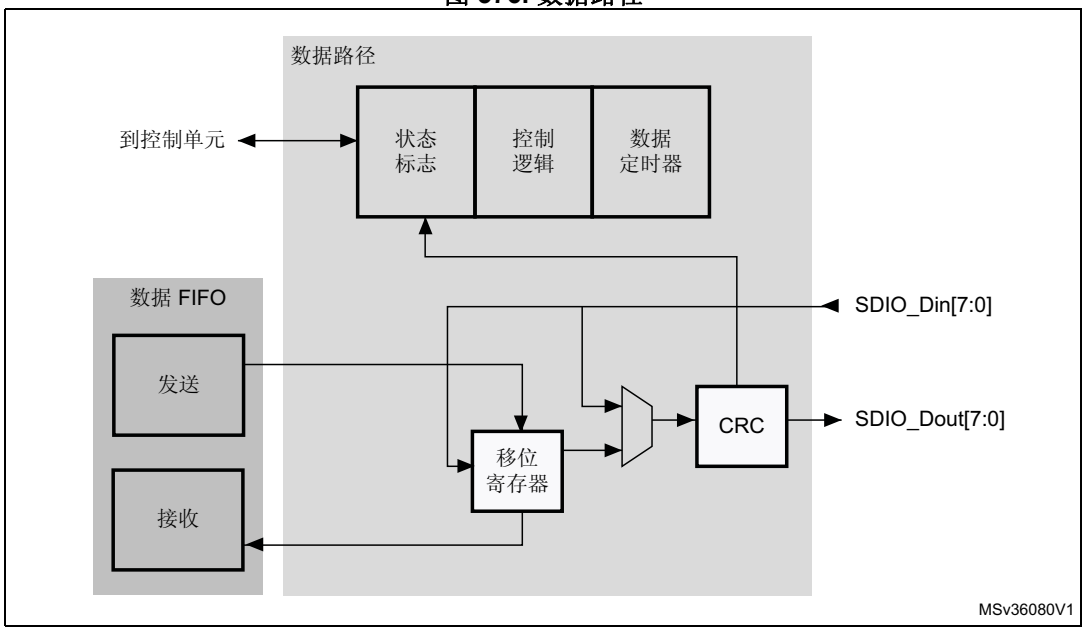
M(x) = (起始位) * x^{39} + ... + (CRC 前的最后一位) * x^0，或者

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 376* 显示了数据路径的框图。

图 376. 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 (SDIO_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 (SDIO_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 SDIO_D0 在每个时钟周期内仅传输一位。

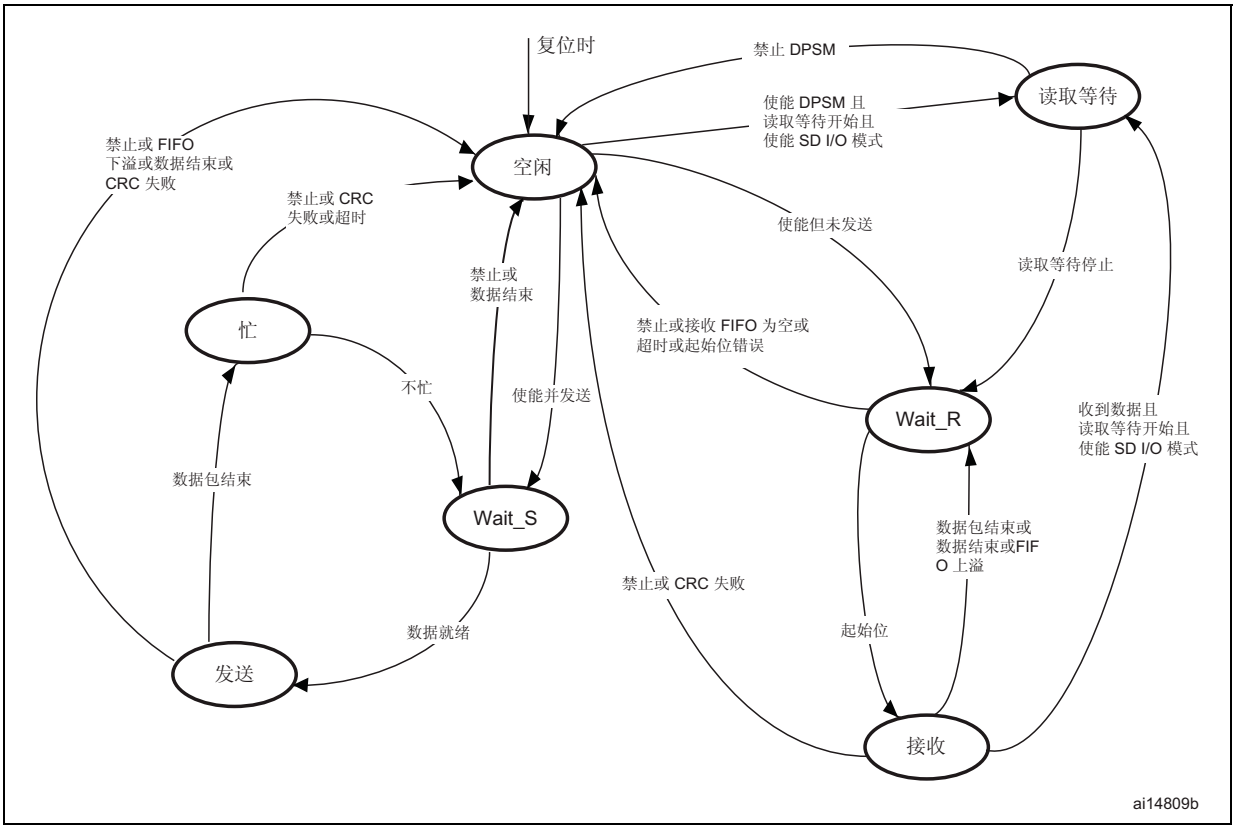
根据传输方向（发送或接收），数据路径状态机 (DPSM) 将变为 Wait_S 或 Wait_R 状态（如果已使能）：

- 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 SDIO_CK 频率运行。卡总线信号上的数据与 SDIO_CK 的上升沿保持同步。DPSM 有六种状态，如*图 377：数据路径状态机 (DPSM)* 中所示。

图 377. 数据路径状态机 (DPSM)



- 空闲：数据路径处于无效状态，SDIO_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 SDIO_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 FIFO 空标志失效，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} 时序要求，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 SDIO_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定时器，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，或反之亦然。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 187. 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 188. 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 SDIO_DCOUNT 寄存器递减并达到 0 时，设置为高电平。 DATAEND 表示 SDIO 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 SDIO 时钟域 (SDIOCLK) 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 SDIO 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 189. 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 SDIO 清零寄存器可将该标志清零。 *注：* 使用 TXUNDERR 和 DMA 填充 SDIO FIFO 时，*则用户软件应禁止 DMA 流，然后向 SDIO_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，则所有状态标志无效，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 190* 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 190. 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 SDIO 清零寄存器可将该标志清零。 *注：　使用 RXOVERR 和 DMA 读取 SDIO FIFO 时，用户软件应禁止 DMA 流，然后向 SDIO_DCTRL 中的 DMAEN 位写入“0”（以禁止生成 DMA 请求）。*

31.3.2 SDIO APB2 接口

APB2 接口生成中断和 DMA 请求，并且访问 SDIO 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。

SDIO 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。

SDIO/DMA 接口

SDIO 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 SDIO 数据长度寄存器进行编程（在卡识别过程之前，SDIO 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 *SDIO 控制器的 DMA 配置*）

c) 对 SDIO 数据控制寄存器进行编程：DTEN 为“1”（使能 SDIO 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）。其他位域无关。

d) 使用数据传输目标卡的地址单元对 SDIO 参数寄存器进行编程。

e) 对 SDIO 命令寄存器进行编程：CmdIndex 为 17 (READ_BLOCK)；WaitResp 为“1”（SDIO 卡主机等待响应）；CPSMEN 为“1”（使能 SDIO 卡主机，以发送命令）。其他位域是其各自的复位值。

f) 等待 SDIO_STA[6] = CMDREND 中断，（如果命令路径上没有错误，CMDREND 将置 1）。

g) 等待 SDIO_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，SDIO_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 SDIO 数据长度寄存器进行编程（在卡识别过程之前，SDIO 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 *SDIO 控制器的 DMA 配置*）。

c) 使用数据传输目标卡的地址单元对 SDIO 参数寄存器进行编程。

d) 对 SDIO 命令寄存器进行编程：CmdIndex 为 24 (WRITE_BLOCK)；WaitResp 为“1”（SDIO 卡主机等待响应）；CPSMEN 为“1”（使能 SDIO 卡主机，以发送命令）。其他位域是其各自的复位值。

e) 等待 SDIO_STA[6] = CMDREND 中断，然后对 SDIO 数据控制寄存器进行编程：DTEN 为“1”（使能 SDIO 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）。其他位域无关。

f) 等待 SDIO_STA[10] = DBCKEND，（没有错误时，DBCKEND 将置 1）

SDIO 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) 使用内存单元的基址对 DMA2_Stream3（或 DMA2_Stream6）通道 4 源地址寄存器进行编程，并使用 SDIO_FIFO 寄存器地址对 DMA2_Stream3（或 DMA2_Stream6）通道 4 目标地址寄存器进行编程。

c) 对 DMA2_Stream3（或 DMA2_Stream6）通道 4 控制寄存器进行编程（存储器递增、非外设递增，外设和源宽度都是字）。

d) 对 DMA2_Stream3（或 DMA2_Stream6）通道 4 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（或 DMA2_Stream6）通道 4 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（或 DMA2_Stream6）通道 4。

注： *SDIO 主机只允许在外设流控制器模式下使用 DMA。用于 SDIO 的 DMA 流必须配置为外设流控制器模式。*

SDIO 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。

31.4 卡功能说明

31.4.1 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信号线 (CMD)。

31.4.2 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器都处于高阻态状态，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。

31.4.3 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 SDIO 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。

在负载存储区中存储卡标识号 (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 SDIO 卡主机模块和卡具有不兼容的 V_{DD} 范围时，卡无法完成识别过程且无法发送 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 SDIO 卡主机要求的 V_{DD} 范围不匹配的卡。SDIO 卡主机会发送所需的 V_{DD} 电压窗口作为这些命令的操作数。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操作数包括在内，则 SDIO 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 SDIO 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。

31.4.4 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。SDIO_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. SDIO 卡主机广播 SEND_OP_COND (CMD1) 以接收操作条件。
3. 响应是来自所有卡的操作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. SDIO 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 SDIO 卡主机的卡进入识别状态。
7. SDIO 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. SDIO 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 SDIO_CMD 线输出驱动是推挽驱动器，而非开漏引脚。注册过程以如下方式完成：

1. 激活总线。
2. SDIO 卡主机广播 SD_APP_OP_COND (ACMD41)。
3. 卡以其操作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. SDIO 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. SDIO 卡主机使用一个地址给一激活卡发出 ET_RELATIVE_ADDR (CMD3) 命令。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。SDIO 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. SDIO 卡主机对所有有效卡重复步骤 5 到 7。

对于 SD I/O 卡，注册过程以如下方式完成：

1. 激活总线。
2. SDIO 卡主机发送 IO_SEND_OP_COND (CMD5)。
3. 卡以其操作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. SDIO 卡主机使用一个地址给一激活卡发出 SET_RELATIVE_ADDR (CMD3) 命令。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。SDIO 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。

31.4.5 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 SDIO_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，则这不变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 SDIO_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 SDIO_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 SDIO_D 拉到低电平来重新激活忙碌指示。

31.4.6 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）。

31.4.7 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 SDIO 卡主机到卡的数据传输，从指定地址开始并继续，直到 SDIO 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），则数据流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 SDIO 卡主机未发出停止命令，则将丢弃已传输的任何额外的数据。

使用以下特定于卡的数据寄存器等式位域，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 = 最大写入数据块长度
- 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 SDIO 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 SDIO 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位域，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。

31.4.8 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。擦除命令中的地址位域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 SDIO_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。

31.4.9 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。

31.4.10 保护管理

SDIO 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 SDIO 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位域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 SDIO 卡主机模块表明，卡受到写保护。SDIO 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 SDIO 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 SDIO 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 SDIO 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 204* 中所示。

位设置如下所示：

- ERASE：设置后会强制执行擦除操作。所有其他位都必须为零，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 一起置 1，但不能与 CLR_PWD 一起置 1
- CLR_PWD：置 1 后会清除密码数据
- SET_PWD：置 1 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序列。

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 位锁/解锁模式、8 位 PWD_LEN 以及新密码字节数确定。当执行一个密码重置操作时，须考虑与命令一起发送的旧密码和新密码。
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。数据块模式 (SET_PWD = 1)、长度 (PWD_LEN) 和密码自身 (PWD)。执行密码重置时，长度值 (PWD_LEN) 会包括新旧两个密码的长度，PWD 位域包括旧密码（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 位域。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。

密码长度位域 (PWD_LEN) 指示当前是否设置了密码。如果此位域非零，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他命令，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 位锁/解锁模式、8 位 PWD_LEN 以及当前使用的密码字节数定。
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 位域并将 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 位锁/解锁模式（*表 204* 中的字节 0）、8 位 PWD_LEN 以及当前使用的密码字节数确定。
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。

可以按照相同的序列来设置密码和锁定卡。在此情况下，SDIO 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 982 页的设置密码*），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 位锁/解锁模式（*表 204* 中的字节 0）、8 位 PWD_LEN 以及以及当前使用的密码字节数确定。
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 位域未清零，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 204* 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。数据块模式 (ERASE = 1)。所有其他位都必须为零。
4. 如果数据位域中仅设置了 ERASE 位，会擦除所有卡内容，包括 PWD 和 PWD_LEN 位域，并且卡不再锁定。如果设置了任何其他位，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。

尝试在未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。

31.4.11 卡状态寄存器

响应格式 R1 包含一个由 32 位位域命名的卡状态。此位域旨在将卡状态信息（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 191 定义了状态的不同条目。表中的类型和清除条件位域以如下方式缩写：

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。SDIO 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 191. 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	如果置 1，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，则会置 1。	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 – 尝试撤销拷贝（设置为原始值）或永久 WP（未受保护）位	C

表 191. 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置 1。**	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	如果置 1，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					

31.4.12 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 SDIO 卡主机。ACMD13 只能发送到处于传输状态的卡（卡已被选定）。

表 192 定义了 SD 状态寄存器的不同条目。表中的类型和清除条件位域以如下方式缩写：

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。SDIO 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 192. 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 192. 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

此位域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位域中指定：

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 位位域指示速度等级，该值可以通过 $P_W/2$ 来计算（其中 P_W 是写入性能）。

表 193. 速度等级代码位域

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 位位域指示 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。将该位域设置为 FFh 即表示无穷大。

表 194. 移动性能位域

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 位位域指示 AU 大小，其取值范围是 2 的 n 次方（起始值为 16 KB）。

表 195. AU_SIZE 位域

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 196* 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 196. 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 位位域指示 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。如果将该位域设置为 0，则不支持擦除超时计算。

表 197. 擦除大小位域

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 位位域指示 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 198. 擦除超时位域

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 位位域指示 TOFFSET，可选择四个值之一。如果 ERASE_SIZE 和 ERASE_TIMEOUT 位域设置为 0，则该位域无意义。

表 199. 擦除偏移位域

ERASE_OFFSET	值定义
0h	0 [秒]
1h	1 [秒]
2h	2 [秒]
3h	3 [秒]

31.4.13 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 SDIO_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 (SDIO_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 (SDIO_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 和复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 SDIO_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”的时序是基于中断周期的。

31.4.14 命令和响应

应用程序特定的命令和常规命令

SDIO 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。
- **寻址（点到点）命令 (AC)**：发送到选定的卡；不包含 SDIO_D 线上的数据传输。
- **寻址（点到点）数据传输命令 (ADTC)**：发送到选定的卡；包含 SDIO_D 线上的数据传输。

命令格式

请参见*第 969 页的表 183* 以了解命令格式。

多媒体卡/SD 模块的命令

表 200. 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 201. 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 202. 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 203. 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）数据位域。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 204. 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 205. 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。				

31.5 响应格式

所有响应都是通过 SDIO 命令线 SDIO_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：

31.5.1 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 206. 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
[39:8]	32	X	卡状态
[7:1]	7	X	CRC7
0	1	1	结束位

31.5.2 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。

31.5.3 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 SDIO_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 207. 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
[127:1]	127	X	卡状态
0	1	1	结束位

31.5.4 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 208. 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

31.5.5 R4（快速 I/O）

代码长度：48 位。参数位域包含寻址卡的 RCA、要读出或写入的寄存器地址及其内容。

表 209. 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] 参数位域	[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

31.5.6 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 210. 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] 参数位域	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
[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。

31.5.7 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 位域将为 0x0。

表 211. 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] 参数位域	[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

31.5.8 R6

仅用于 SD I/O。存储设备对 CMD3 的正常响应。相关内容在*表 212* 中进行了说明。

表 212. 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] 参数位域	[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

31.6 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 SDIO_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 SDIO_DCTRL[11] 位置 1 时，SDIO 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。

31.6.1 通过触发 SDIO_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（SDIO_DCTRL[0] 位置 1）并使能 SDIO 专用操作（SDIO_DCTRL[11] 位置 1），则当读取等待开始（SDIO_DCTRL[10] =0 且 SDIO_DCTRL[8] =1）并且数据方向为从卡到 SDIO (SDIO_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 SDIO_CK 时钟周期后将 SDIO_D2 驱动为 0。在此状态下，如果将 RWSTOP 位 (SDIO_DCTRL[9]) 置 1，DPSM 保持等待状态的时间将多出两个 SDIO_CK 时钟周期，以将 SDIO_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，SDIO 可以在 SDIO_D1 上检测 SDIO 中断。

31.6.2 通过停止 SDIO_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，SDIO 可以通过停止 SDIO_CK 来执行读取等待（SDIO_DCTRL 置 1，这一点与*第 31.6.1 节*所述方法相同，但 SDIO_DCTRL[10] =1）：在当前接收数据块的结束位后，DSPM 将在两个 SDIO_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。

SDIO_CK 停止时，可以向卡发出任何命令。在读取/等待间隔期间，SDIO 可以在 SDIO_D1 上检测 SDIO 中断。

31.6.3 SDIO 挂起/恢复操作

向卡发送数据时，SDIO 可以挂起写操作。SDIO_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 DPSM 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 (SDIO_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。

31.6.4 SDIO 中断

一旦 SDIO_DCTRL[11] 位置 1，即会在 SDIO_D1 线路上检测到 SDIO 中断。

检测到 SDIO 中断时，SDIO_STA[22] (SDIOIT) 位将置 1。此静态位可通过清零位 SDIO_ICR[22] (SDIOITC) 来清零。如果 SDIOIT 位置 1，可产生中断。单独的中断使能 SDIO_MASK[22] 位 (SDIOITE) 可用于使能和禁止中断请求。

发生 SD 卡中断（SDIO_STA[22] 位置 1）时，主机软件将按以下步骤处理。

1. 将 SDIOITE 位清零（SDIO_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”（SDIO_ICR[22] = “1”）以将 SDIOIT 位清零；
4. 向 SDIOITE 位写入 “1”（SDIO_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。

31.7 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 SDIO_CK 并冻结 SDIO 状态机。当 FIFO 无法发送或者接受数据时，数据传输将停止。只有由 SDIOCLK 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，SDIO_CLKCR[14] 寄存器位必须置 1。复位后，流控制将禁止。

31.8 SDIO 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。

31.8.1 SDIO 电源控制寄存器 (SDIO_POWER)

SDIO 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。

位 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 SDIOCLK 时钟周期外加两个 PCLK2 时钟周期内无法向该寄存器写入数据。*

31.8.2 SDIO 时钟控制寄存器 (SDIO_CLKCR)

SDIO clock control register

偏移地址：0x04

复位值：0x0000 0000

SDIO_CLKCR 寄存器控制 SDIO_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 31.8.11 节*的 SDIO 状态寄存器定义来确定 TXFIFOE 和 RXFIFOF 中断信号的含义。

位 13 **NEGEDGE**：SDIO_CK 移相选择位 (SDMMC_CK dephasing selection bit)
0b：在 SDIO_CK 上升沿后的 SDIOCLK 下降沿更改命令和数据（SDIO_CK 上升沿在 SDIOCLK 上升沿时出现）。
1b：在 SDIO_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 SDIOCLK 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 SDIO_D0
01：4 位宽总线模式：使用 SDIO_D[3:0]
10：8 位宽总线模式：使用 SDIO_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 SDIO_CK 输出信号前，根据 CLKDIV 值对 SDIOCLK 进行分频。
1：使能旁路：SDIOCLK 直接驱动 SDIO_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 SDIO_CK 时钟输出：
0：始终使能 SDIO_CK 时钟
1：仅在总线激活时使能 SDIO_CK

位 8 **CLKEN**：时钟使能位 (Clock enable bit)
0：禁止 SDIO_CK
1：使能 SDIO_CK

位 7:0 **CLKDIV**：时钟分频系数 (Clock divide factor)
该位域定义输入时钟 (SDIOCLK) 与输出时钟 (SDIO_CK) 之间的分频系数：SDIO_CK 频率 = SDIOCLK / [CLKDIV + 2]。

注： *1 当 SD/SDIO 卡或 MMC 处于识别模式时，SDIO_CK 频率必须小于 400 kHz。*

2 如果为所有卡分配了相对卡地址，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 SDIOCLK 时钟周期外加两个 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 SDIO_CK；在此情况下，SDIO_CLKCR 寄存器不对 SDIO_CK 进行控制。

31.8.3 SDIO 参数寄存器 (SDIO_ARG)

SDIO argument register

偏移地址：0x08

复位值：0x0000 0000

SDIO_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。

31.8.4 SDIO 命令寄存器 (SDIO_CMD)

SDIO command register

偏移地址：0x0C

复位值：0x0000 0000

SDIO_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 SDIO_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，期待 CMDSENT 标志
01：短响应，期待 CMDSENT 标志或 CCRCFAIL 标志
10：无响应，期待 CMDSENT 标志
11：长响应，期待 CMDSENT 标志或 CCRCFAIL 标志

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 SDIOCLK 时钟周期外加两个 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。*

31.8.5 SDIO 命令响应寄存器 (SDIO_RESPCMD)

SDIO command response register

偏移地址：0x10

复位值：0x0000 0000

SDIO_RESPCMD 寄存器包含接收到的最后一个命令响应的命令索引位域。如果命令响应传输不包含命令索引位域（长响应或 OCR 响应），则 RESPCMD 位域为未知，但该位域必须包含 111111b（响应中保留位域的值）。

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)
只读位域。包含接收到的最后一个命令响应的命令索引。

31.8.6 SDIO 响应 1..4 寄存器 (SDIO_RESPx)

SDIO response 1..4 register

偏移地址：(0x10 + (4 × x))；x = 1..4

复位值：0x0000 0000

SDIO_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 213*。

卡状态为 32 位或 127 位，具体取决于响应类型。

表 213. 响应类型和 SDIO_RESPx 寄存器

寄存器	短响应	长响应
SDIO_RESP1	卡状态 [31:0]	卡状态 [127:96]
SDIO_RESP2	未使用	卡状态 [95:64]
SDIO_RESP3	未使用	卡状态 [63:32]
SDIO_RESP4	未使用	卡状态 [31:0]

首先接收卡状态的最高有效位。SDIO_RESP4 寄存器 LSB 始终为 0b。

31.8.7 SDIO 数据定时器寄存器 (SDIO_DTIMER)

SDIO data timer register

偏移地址：0x24

复位值：0x0000 0000

SDIO_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 SDIO_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。*

31.8.8 SDIO 数据长度寄存器 (SDIO_DLEN)

SDIO data length register

偏移地址：0x28

复位值：0x0000 0000

SDIO_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）。*
超时必须首先写入到数据定时器寄存器和数据长度寄存器，然后才能写入到数据控制寄存器。
对于 IO_RW_EXTENDED (CMD53)：
- 如果选择流或 SDIO 多字节传输，则数据长度寄存器中的值必须介于 1 到 512 之间。
- 如果选择块数据传输，则数据长度寄存器中的值必须介于 1 数据块大小和 512* 数据块大小之间。*

31.8.9 SDIO 数据控制寄存器 (SDIO_DCTRL)

SDIO data control register

偏移地址：0x2C

复位值：0x0000 0000

SDIO_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 SDIO_D2 进行读取等待控制
1：使用 SDIO_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。

位 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 SDIO_DCTRL 以使能新的数据传输。

注： *写入数据后，在三个 SDIOCLK 时钟周期外加两个 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。

31.8.10 SDIO 数据计数器寄存器 (SDIO_DCOUNT)

SDIO 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，SDIO_DCOUNT 寄存器将从数据长度寄存器中加载值（请参见 SDIO_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。*

31.8.11 SDIO 状态寄存器 (SDIO_STA)

SDIO status register

偏移地址：0x34

复位值：0x0000 0000

SDIO_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 SDIO 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 SDIO_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**：发送 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 SDIO FIFO（SDIO_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 SDIO FIFO（SDIO_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 SDIO_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))

31.8.12 SDIO 中断清零寄存器 (SDIO_ICR)

SDIO interrupt clear register

偏移地址：0x38

复位值：0x0000 0000

SDIO_ICR 寄存器是一个只写寄存器。向某个位写入 1 会将 SDIO_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

31.8.13 SDIO 屏蔽寄存器 (SDIO_MASK)

SDIO 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.	SDIO ITIE	RXD AVLIE	TXD AVLIE	RX FIFO EIE	TX FIFO EIE	RX FIFO FIE	TX FIFO FIE
									rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX FIFO HFIE	TX FIFO HEIE	RX ACTIE	TX ACTIE	CMD ACTIE	DBCK ENDIE	Res.	DATA ENDIE	CMD SENT IE	CMD REND IE	RX OVERR IE	TX UNDERR IE	DTIME OUTIE	CTIME OUTIE	DCRC FAILIE	CCRC FAILIE
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

31.8.14 SDIO FIFO 计数器寄存器 (SDIO_FIFOCNT)

SDIO FIFO counter register

偏移地址：0x48

复位值：0x0000 0000

SDIO_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（SDIO_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 SDIO_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。

31.8.15 SDIO 数据 FIFO 寄存器 (SDIO_FIFO)

SDIO 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) 多操作数命令来读写 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：SDIO 基址 + 0x080 到 SDIO 基址 + 0xFC。

31.8.16 SDIO 寄存器映射

下表对 SDIO 寄存器进行了汇总。

表 214. SDIO 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	SDIO_ 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	SDIO_ 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	SDIO_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	SDIO_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	SDIO_ 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	SDIO_ 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
0x18	SDIO_ 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	SDIO_ 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	SDIO_ 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	SDIO_ 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	SDIO_ 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	SDIO_ 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	SDIO_ 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

表 214. SDIO 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
0x34	**SDIO_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	**SDIO_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	**SDIO_ 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	**SDIO_ 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	**SDIO_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*。

32 控制器局域网络 (bxCAN)

32.1 简介

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。

32.2 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
- 可扩展的过滤器组：
  - 对于双 CAN，CAN1 和 CAN2 之间共享 28 个过滤器组
  - 对于单 CAN，则为 14 个过滤器组
- 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 外设配置

- CAN1：主 bxCAN，用于管理主/从 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*第 1018 页的图 379：双 CAN 框图*）

单 CAN 外设配置：

- CAN3：具有专用存储器访问控制器单元和 512 字节 SRAM 存储器的主 bxCAN

请参见*表 215*。

表 215. CAN 实现

CAN 特性	CAN1	CAN2	CAN3
SRAM 大小	两个 bxCAN 之间共用 512 字节		512 字节
过滤器组	CAN1 和 CAN2 之间共享 26 个过滤器组		14 个过滤器组

32.3 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。通常，系统中的消息（各个节点需要处理的消息）数量也有了显著增加。除应用程序消息外，还引入了网络管理和诊断消息。

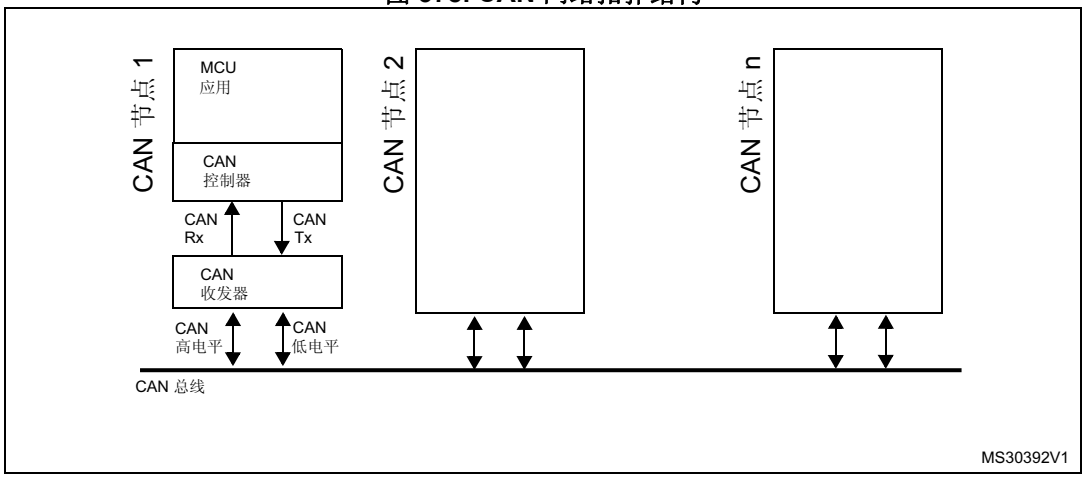
- 各种类型的消息需要一个增强的过滤机制进行处理。

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 378. CAN 网络拓扑结构



32.3.1 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）硬件完全支持。

32.3.2 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

32.3.3 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。

32.3.4 验收过滤器

bxCAN 在双 CAN 配置中提供多达 28 个可扩展/可配置的标识符过滤器组，或者在单 CAN 配置中提供多达 14 个可扩展/可配置的标识符过滤器组，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 379. 双 CAN 框图

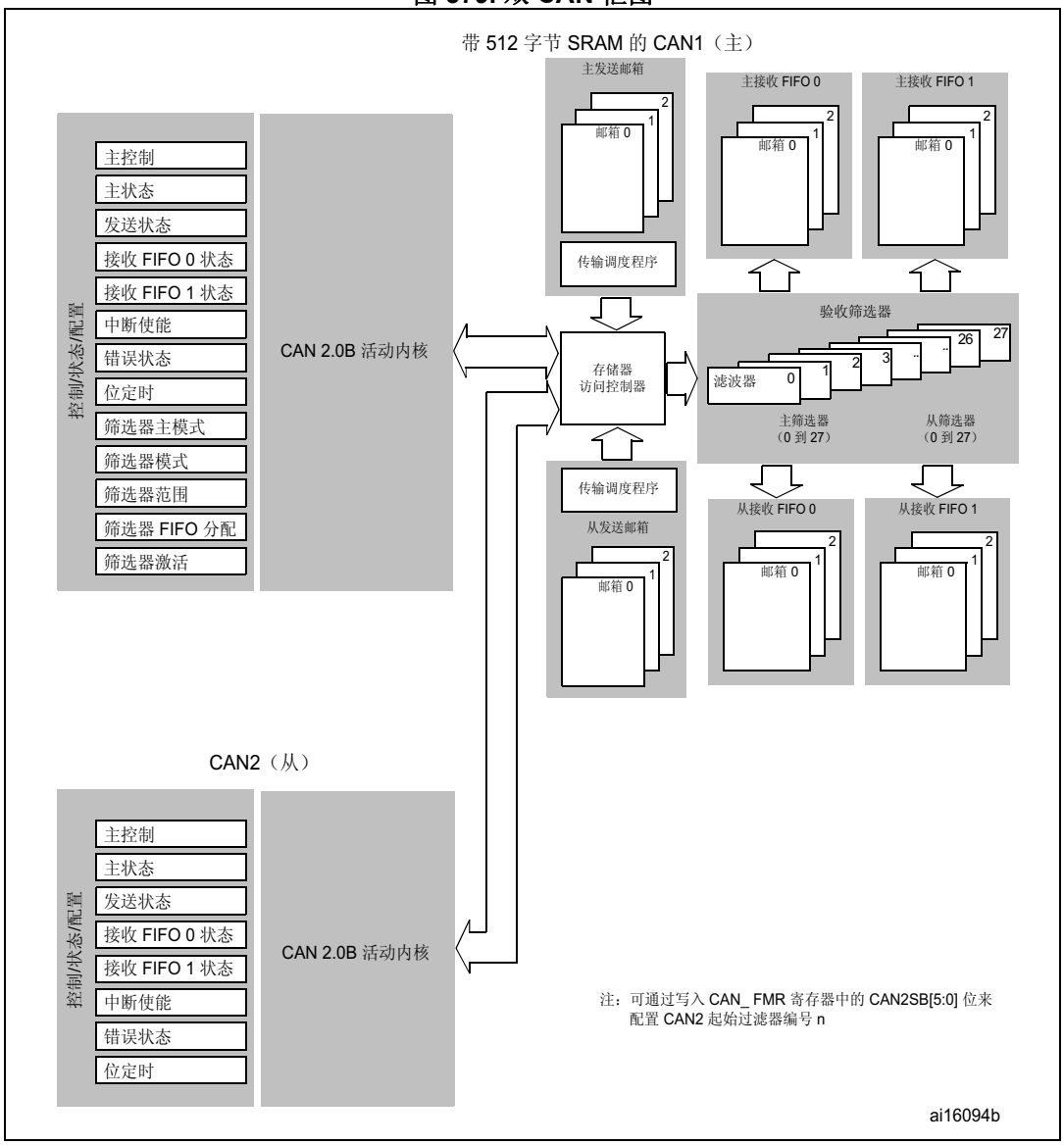
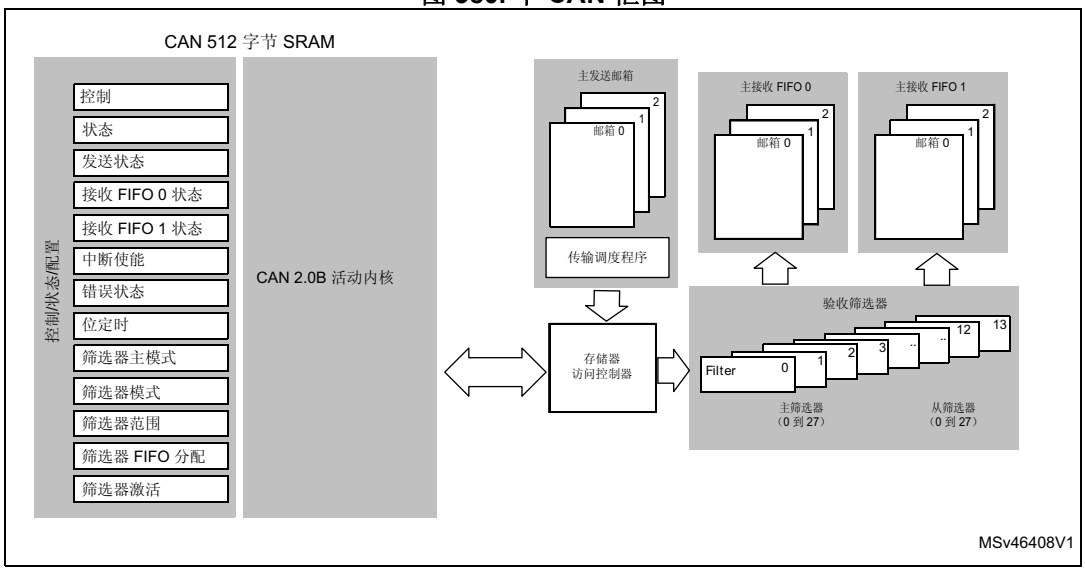

带 512 字节 SRAM 的 CAN1（主）
主发送邮箱
邮箱 0
传输调度程序
主接收 FIFO 0
主接收 FIFO 1
主控制
主状态
发送状态
接收 FIFO 0 状态
接收 FIFO 1 状态
中断使能
错误状态
位定时
筛选器主模式
筛选器模式
筛选器范围
筛选器 FIFO 分配
筛选器激活
控制/状态/配置
CAN 2.0B 活动内核
存储器
访问控制器
验收筛选器
滤波器
主筛选器
（0 到 27）
从筛选器
（0 到 27）
从发送邮箱
从接收 FIFO 0
从接收 FIFO 1
CAN2（从）
注：可通过写入 CAN_ FMR 寄存器中的 CAN2SB[5:0] 位来
配置 CAN2 起始过滤器编号 n
ai16094b

图 380. 单 CAN 框图



32.4 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）。

32.4.1 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_MSR 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 过滤器组相关的寄存器（模式、尺度、FIFO 分配、激活和过滤器值），软件必须将 FINIT 位 (CAN_FMR) 置 1。过滤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。*

过滤器值也可通过停用（CAN_FA1R 寄存器的）相关过滤器激活位来修改。

如果某个过滤器组未使用，建议将其保持未激活状态（将相应 FACT 位保持清零）。

32.4.2 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 位清零来发起。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。

过滤器值的初始化与初始化模式无关，但必须要在过滤器处于未激活状态（相应 FACTx 位清零）时进行。过滤器尺度和模式配置必须在进入正常模式之前完成。

32.4.3 睡眠模式（低功耗）

为降低功耗，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。

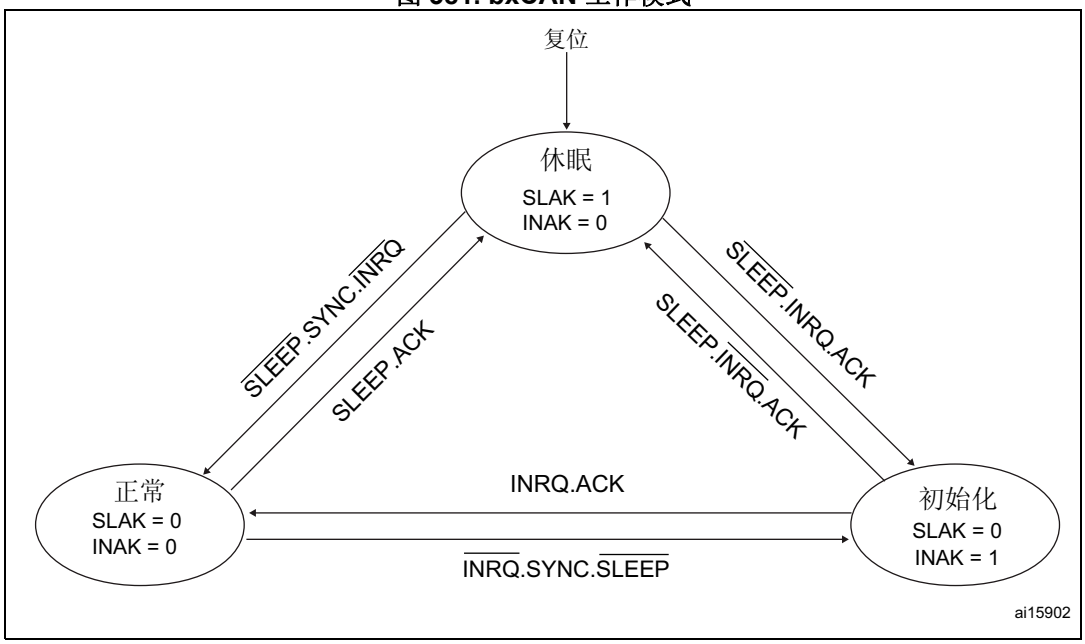
软件将 SLEEP 位清零或是检测到 CAN 总线活动时，bxCAN 即被唤醒（退出睡眠模式）。

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 381：bxCAN 工作模式*。一旦硬件将 SLAK 位清零，即会退出睡眠模式。

图 381. 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

32.5 测试模式

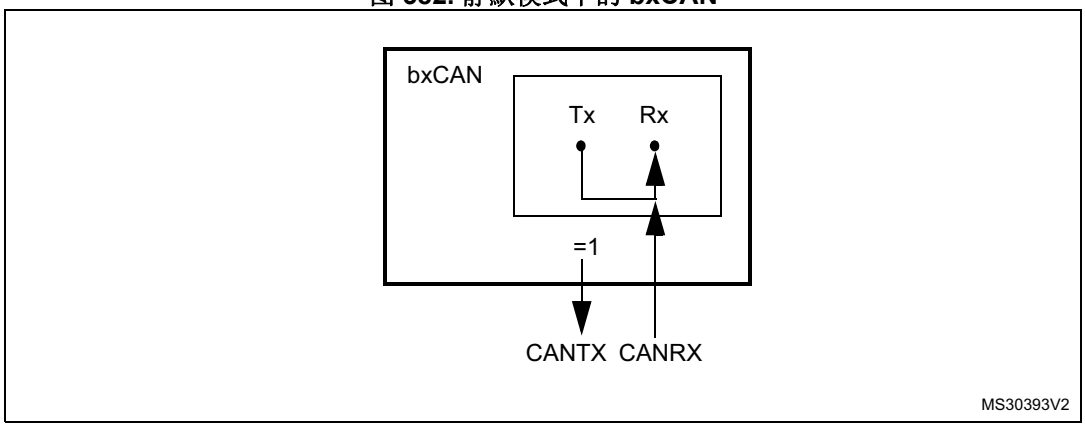
可以通过 CAN_BTR 寄存器中的 SILM 和 LBKM 位来选择测试模式。这些位必须在 bxCAN 处于初始化模式时进行配置。选择测试模式后，必须复位 CAN_MCR 寄存器中的 INRQ 位才能进入正常模式。

32.5.1 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。

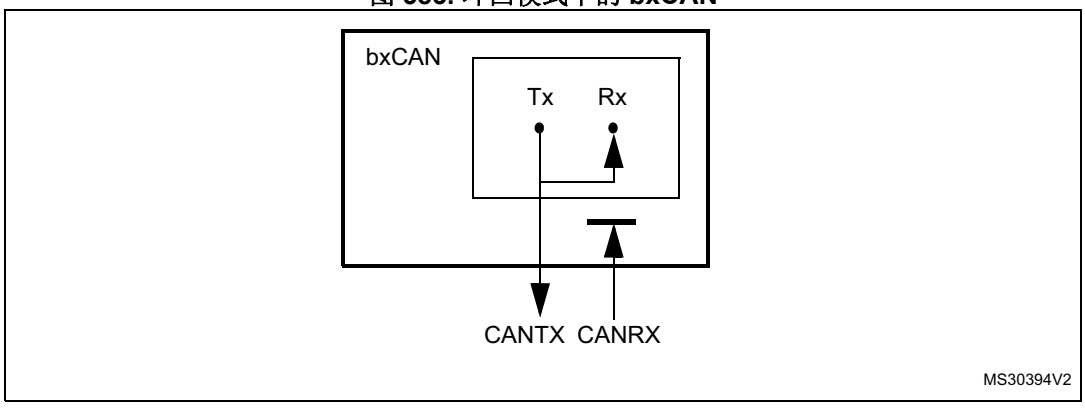
图 382. 静默模式下的 bxCAN



32.5.2 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过滤）在接收邮箱中。

图 383. 环回模式下的 bxCAN

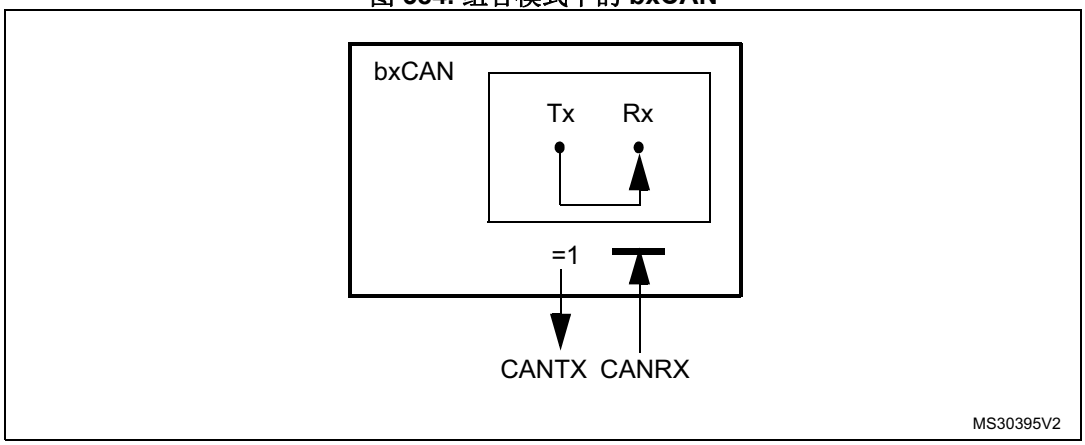


该模式为自检功能提供。为了不受外部事件的影响，CAN 内核在环回模式下将忽略确认错误（在数据/远程帧的确认时隙不对显性位采样）。在此模式下，bxCAN 将执行从发送输出到接收输入的内部反馈。bxCAN 将忽略 CANRX 输入引脚的实际值。从 CANTX 引脚可以监视发送的消息。

32.5.3 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 384. 组合模式下的 bxCAN



32.6 调试模式下的行为

当微控制器进入调试模式（Cortex®-M4 with FPU 内核停止）时，bxCAN 是继续正常工作还是停止工作，具体取决于如下条件：

- DBGMCU_APB1_FZ 寄存器中的 DBG_CAN1_STOP 位（针对 CAN1）、DBG_CAN2_STOP 位（针对 CAN2）或 DBG_CAN3_STOP 位（针对 CAN3）
- CAN_MCR 中的 DBF 位，有关更多详细信息，请参见*第 32.9.2 节：CAN 控制和状态寄存器*。

32.7 bxCAN 功能说明

32.7.1 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。

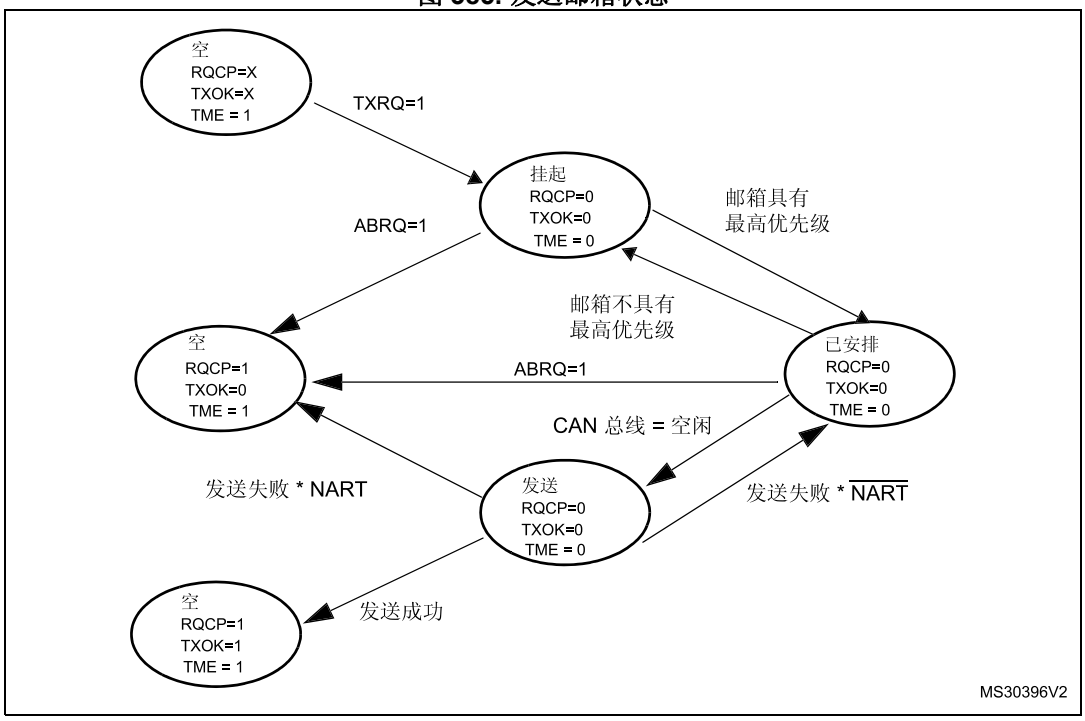
禁止自动重发送模式

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 385. 发送邮箱状态



32.7.2 时间触发通信模式

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 32.7.7 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。

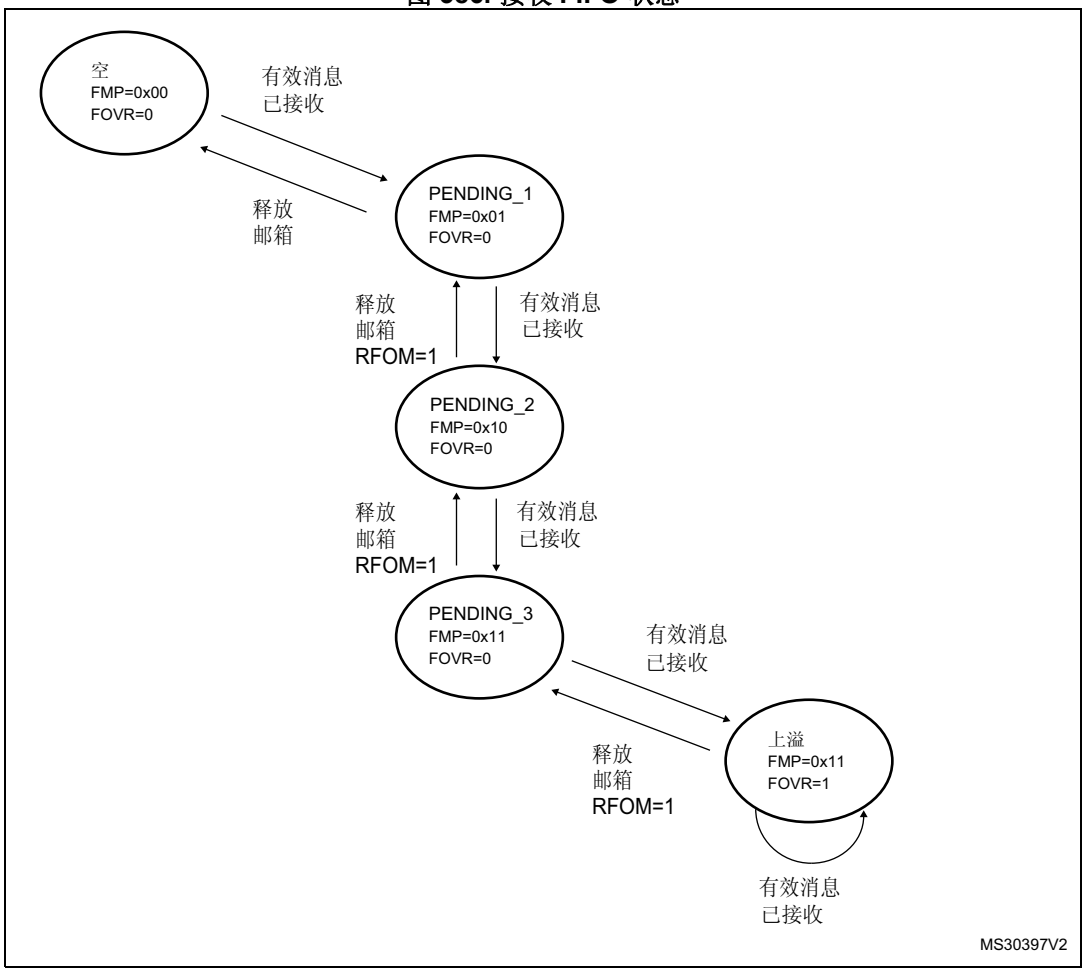
32.7.3 接收处理

为了接收 CAN 消息，提供了三个邮箱构成一个 FIFO。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

当一条消息被正常接收且完全符合 CAN 协议标准（直到最后一位 EOF 位域都没有错误），**并且**成功通过标识符过滤后，该消息将视为有效，请参见*第 32.7.4 节：标识符过滤*。

图 386. 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 32.7.5 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，软件将得到 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。

32.7.4 标识符过滤

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。

为了满足双 CAN 配置中的这一要求，bxCAN 控制器为应用程序提供了 28 个可配置且可调整的过滤器组 (27-0)。在单 CAN 配置中，bxCAN 控制器为应用程序提供了 14 个可配置且可调整的过滤器组 (13-0)，以便仅接收软件需要的消息。

此硬件过滤功能可以节省软件过滤所需的 CPU 资源。每个过滤器组 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过滤器，每个过滤器组可分别进行位宽调整。根据过滤器尺度不同，一个过滤器组可以：

- 为 STDID[10:0]、EXTID[17:0]、IDE 和 RTR 位提供一个 32 位过滤器。
- 为 STDID[10:0]、RTR、IDE 和 EXTID[17:15] 位提供两个 16 位过滤器。

请参见*图 387*。

此外，过滤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过滤器寄存器中指定的位匹配。

过滤器组尺度和模式配置

过滤器组通过相应的 CAN_FMR 寄存器进行配置。为了配置过滤器组，必须通过将 CAN_FAR 寄存器的 FACT 位清零而将其停用。过滤器尺度通过 CAN_FS1R 寄存器的相应 FSCx 位进行配置，请参见*图 387*。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。

要过滤一组标识符，应将掩码/标识符寄存器配置为掩码模式。

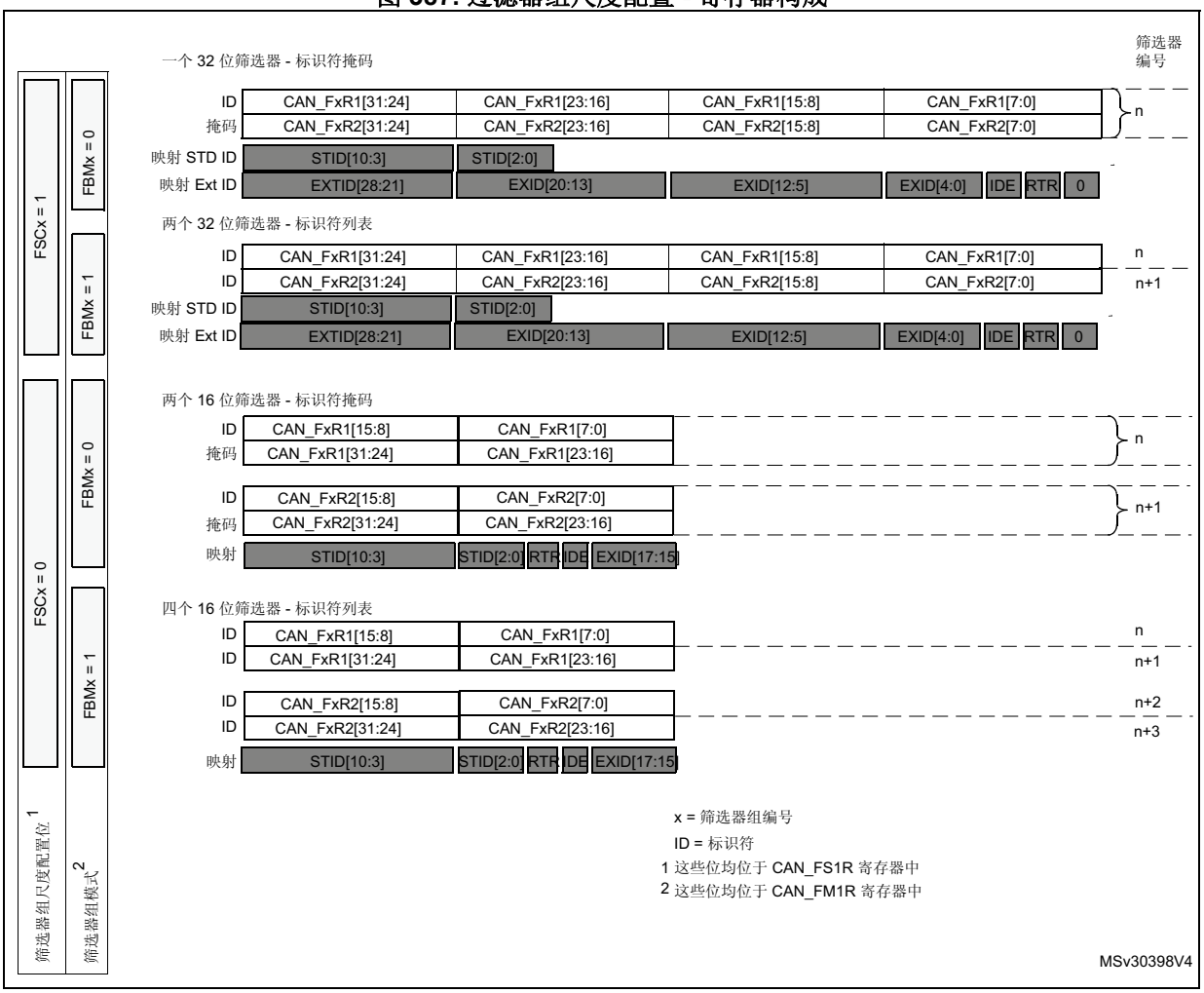
要选择单个标识符，应将掩码/标识符寄存器配置为标识符列表模式。

未由应用程序使用的过滤器应保持停用。

过滤器组中的每个过滤器将按从 0 到最大值的顺序进行编号（称为*过滤器编号*），具体取决于每个过滤器组的模式和尺度。

有关过滤器配置，请参见*图 387*。

图 387. 过滤器组尺度配置 - 寄存器构成



过滤器匹配索引

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 控制器提供了一个过滤器匹配索引。

该索引根据过滤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过滤器匹配索引。

过滤器匹配索引的使用方法有两种：

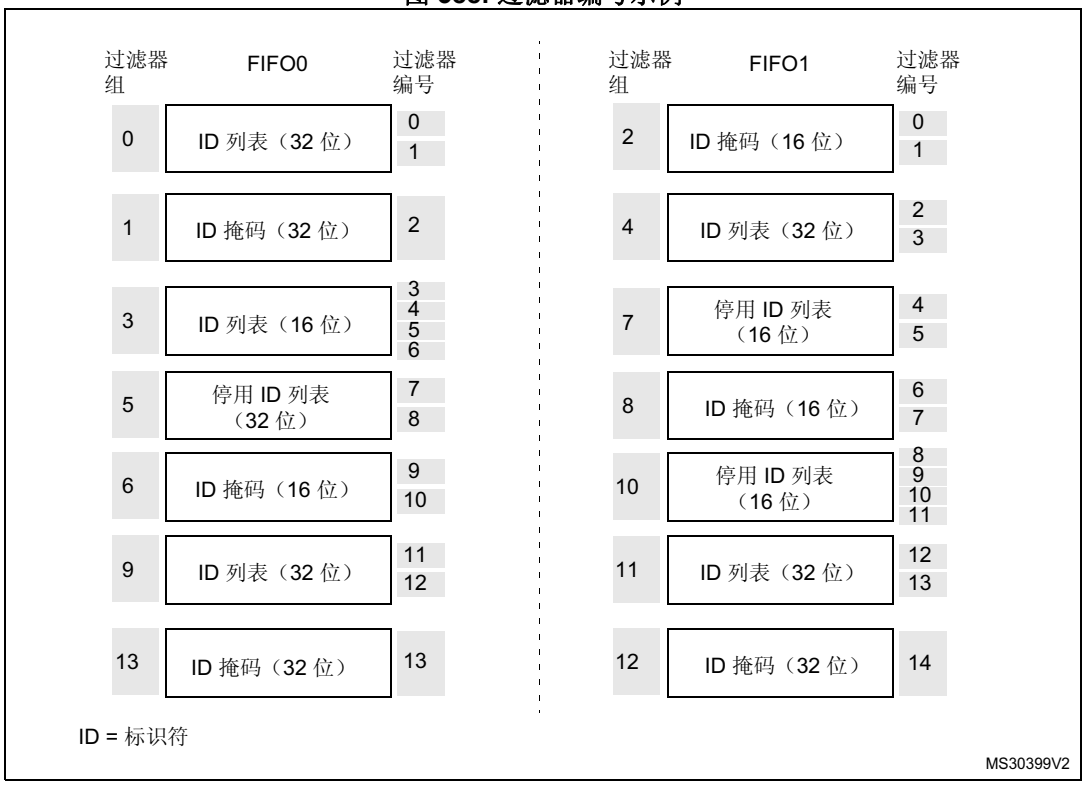
- 将过滤器匹配索引与预期值列表进行比较。
- 将过滤器匹配索引用作阵列索引，以访问数据目标位置。

对于非屏蔽过滤器，软件不再需要比较标识符。

如果过滤器有屏蔽，软件则只需比较屏蔽位。

过滤器编号的索引值与过滤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 388*。

图 388. 过滤器编号示例

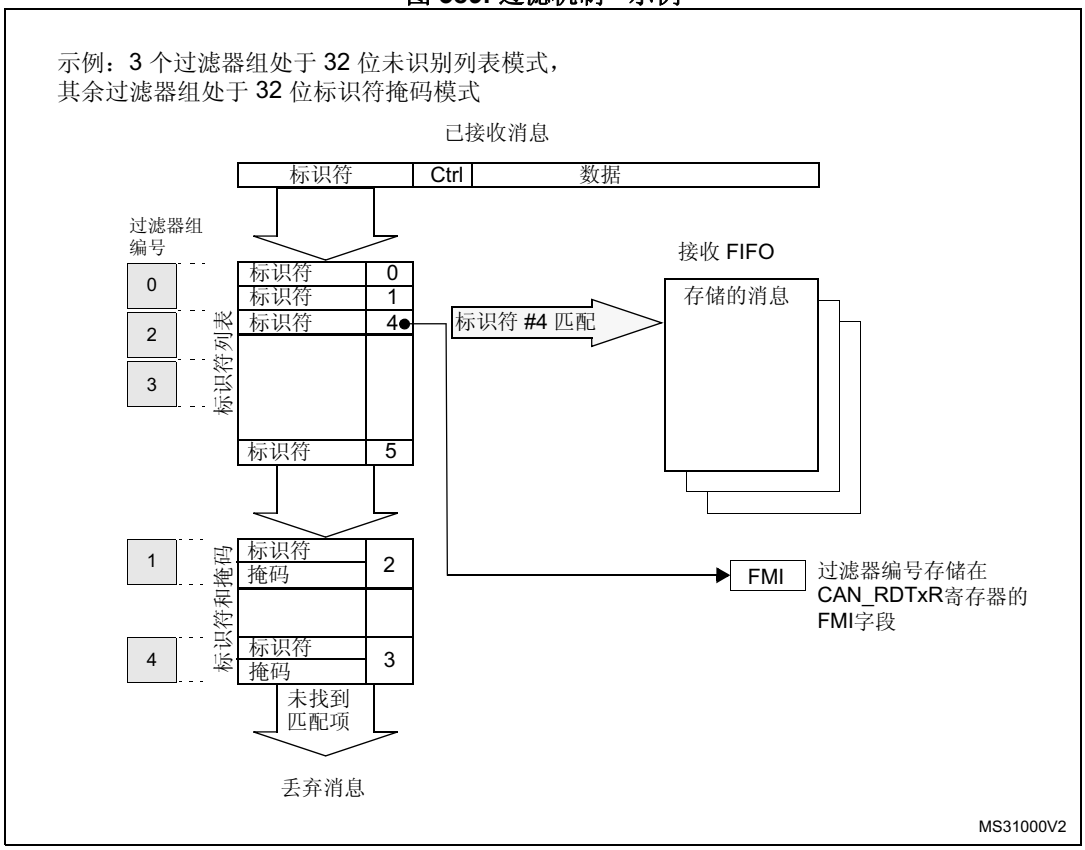


过滤器优先级规则

根据过滤器组合，可能会出现一个标识符成功通过数个过滤器的情况。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过滤器匹配值：

- 32 位过滤器优先于 16 位过滤器。
- 对于尺度相等的过滤器，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过滤器，则按过滤器编号确定优先级（编号越低，优先级越高）。

图 389. 过滤机制 - 示例



以上示例说明了 bxCAN 的过滤原则。接收到消息后，首先标识符与配成标识符列表模式的过滤器进行对比。如果匹配，消息将存储在相应 FIFO 中，对应的过滤器编号则存储到 FMI 字段中。如本例所示，标识符与标识符 #4 匹配，因此消息内容和 FMI 4 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过滤器进行比较。

如果标识符与过滤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。

32.7.5 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 216. 发送邮箱映射

与发送邮箱基址之间的偏移	寄存器名
0	CAN_TIxR
4	CAN_TDTxR
8	CAN_TDLxR
12	CAN_TDHxR

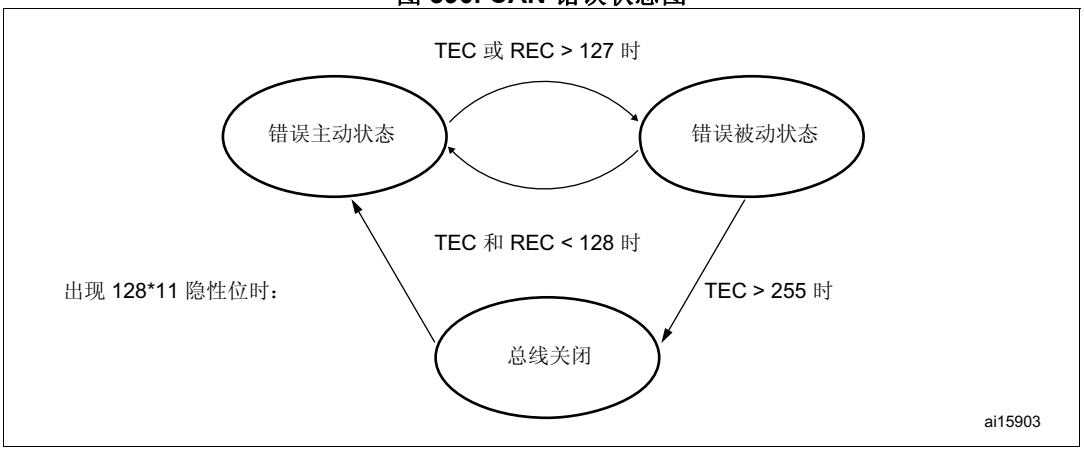
接收邮箱

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。过滤器匹配索引存储在 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 217. 接收邮箱映射

与接收邮箱基址之间的偏移（字节）	寄存器名
0	CAN_RIxR
4	CAN_RDTxR
8	CAN_RDLxR
12	CAN_RDHxR

图 390. CAN 错误状态图



32.7.6 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**。*

32.7.7 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在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 ($1 \times t_q$)。
- **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）。

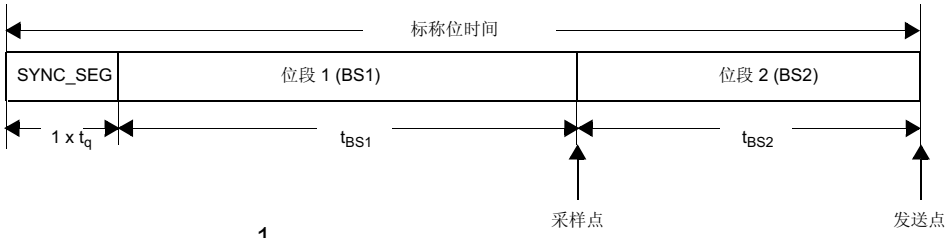
如果在 BS1 而不是 SYNC_SEG 中检测到有效边沿，则 BS1 会延长最多 SJW，以便延迟采样点。

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 391. 位时序



$$\text{Baud Rate} = \frac{1}{\text{NominalBitTime}}$$

$$\text{NominalBitTime} = 1 \times t_q + t_{BS1} + t_{BS2}$$

$t_{BS1} = t_q \times (TS1[3:0] + 1)$,

$t_{BS2} = t_q \times (TS2[2:0] + 1)$,

$t_q = (BRP[9:0] + 1) \times t_{PCLK}$

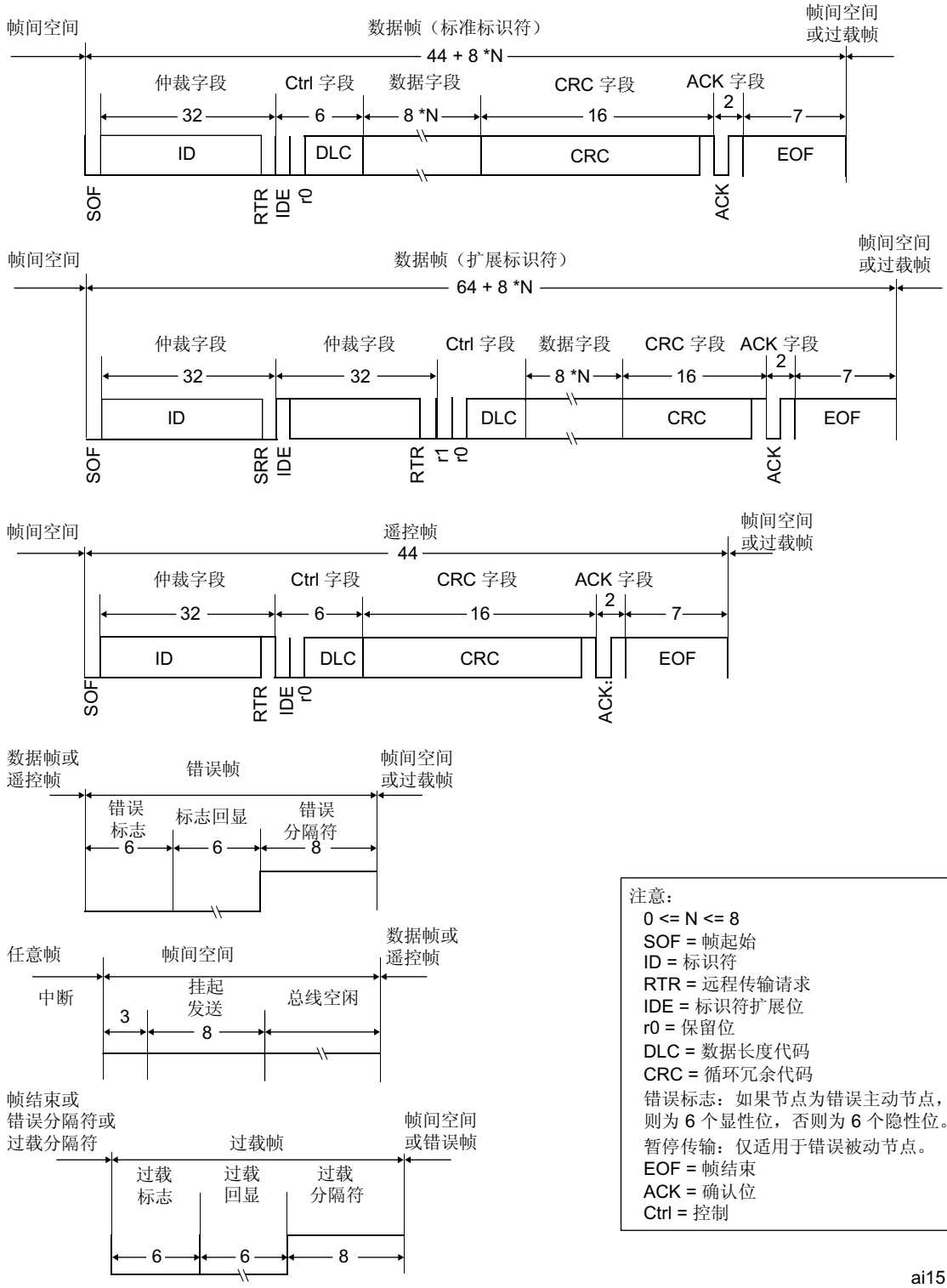
其中，t_q为时间片

t_{PCLK} = APB 时钟的时间周期

BRP[9:0]、TS1[3:0] 和 TS2[2:0] 在 CAN_BTR 寄存器中定义。

MS31001V2

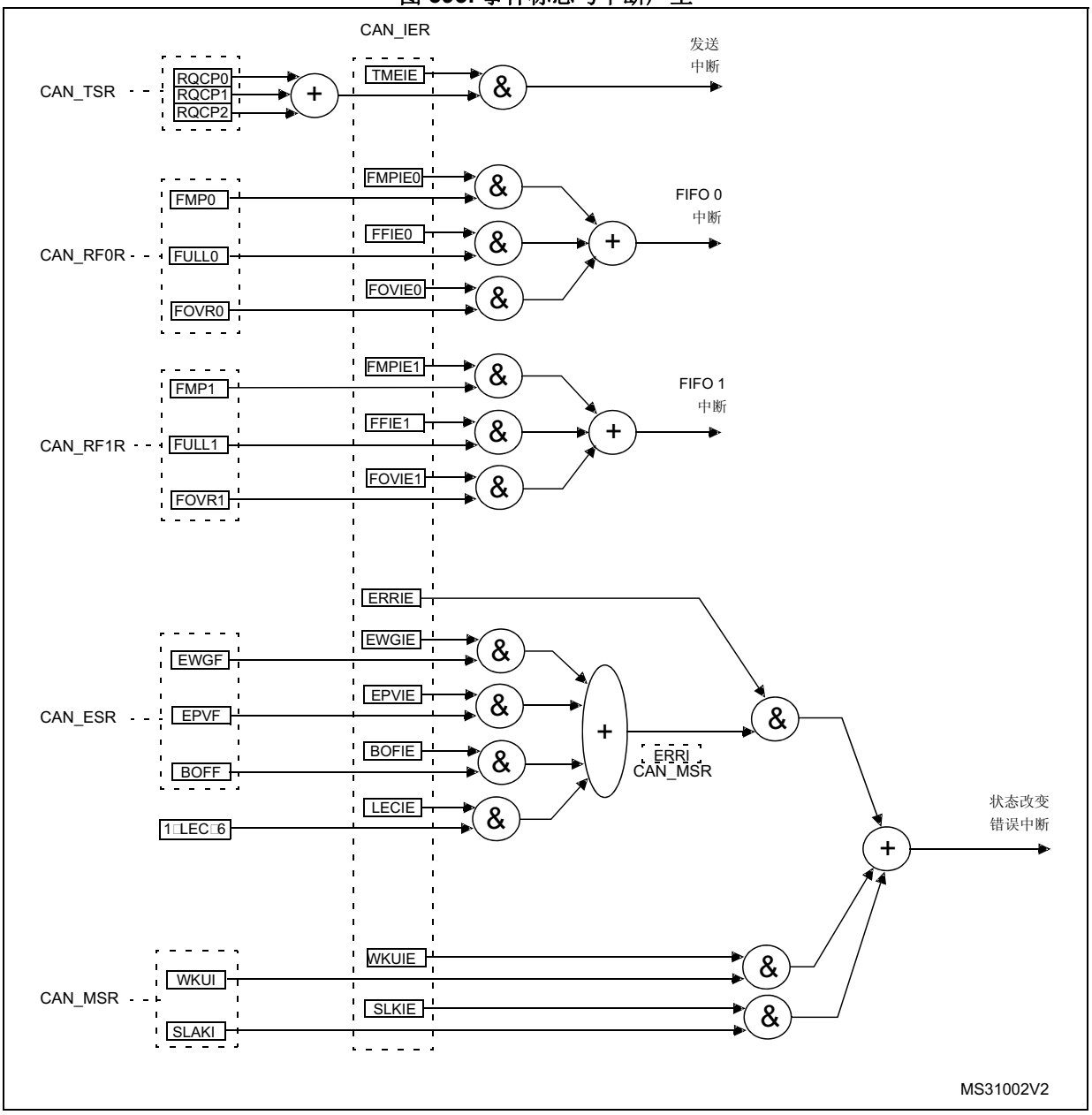
图 392. CAN 帧


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
32
6
8 *N
16
2
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
32
32
8 *N
16
2
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
32
6
16
2
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
挂起发送
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
ai15154b

32.8 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 393. 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。

32.9 CAN 寄存器

外设寄存器必须按字（32 位）进行访问。

32.9.1 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 385：发送邮箱状态*。

可以通过停用相应过滤器组或将 FINIT 位置 1 来修改过滤器值。此外，只有当过滤器被设为初始化模式时（寄存器 CAN_FMR，字段 FINIT=1），才能对 CAN_FMxR、CAN_FSxR 和 CAN_FFAR 寄存器中的过滤器配置（大小、模式和 FIFO 分配）进行修改。

32.9.2 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 1.2 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 32.7.2 节：时间触发通信模式。

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 32.7.6 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 385*。

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 385*。

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 385*。

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过滤器，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过滤器，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 32.8 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 1031 页的第 32.7.6 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 1031 页上的第 32.7.7 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。
$t_q = (BRP[9:0]+1) \times t_{PCLK}$

32.9.3 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 1029 页上的第 32.7.5 节：消息存储*。

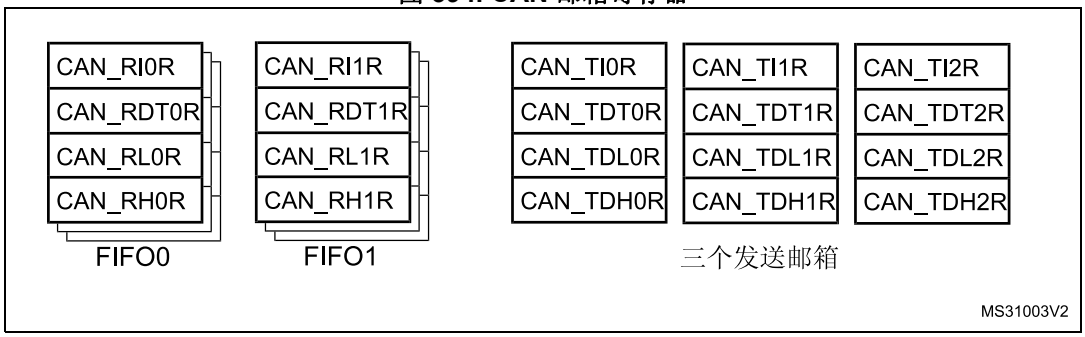
除了以下情况外，发送邮箱和接收邮箱使用相同的寄存器：

- 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 394. 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]**：过滤器匹配索引 (Filter match index)
该寄存器包含过滤器索引，邮箱中存储的消息需要经过该过滤器。有关标识符过滤的更多详细信息，请参见*第 1025 页上的第 32.7.4 节：标识符过滤* - **过滤器匹配索引**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。

32.9.4 CAN 过滤器寄存器

CAN 过滤器主寄存器 (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)
这些位由软件置 1 和清零。使用两个 CAN 时，这些位定义每个 CAN 接口的起始存储区：
000001 = 1 个过滤器分配给 CAN1，27 个过滤器分配给 CAN2
011011 = 27 个过滤器分配给 CAN1，1 个过滤器分配给 CAN2
- 要将所有过滤器分配给一个 CAN：将 CANSB 值设为零，并停用未使用的 CAN
- 要仅使用 CAN1：停止 CAN2 的时钟和/或将 CAN2 的 CAN_MCR.INRQ 置 1
- 要仅使用 CAN2：将 CAN1 的 CAN_MCR.INRQ 置 1，或停用 CAN1 的中断寄存器 CAN_IER

注： 位 [13:8] 仅用于双 CAN 外设配置，对于单 CAN 外设配置则为保留位。

位 7:1 保留，必须保持复位值。

位 0 **FINIT**：过滤器初始化模式 (Filter initialization mode)
过滤器组的初始化模式
0：过滤器工作模式。
1：过滤器初始化模式。

CAN 过滤器模式寄存器 (CAN_FM1R)

CAN filter mode register

偏移地址：0x204
复位值：0x0000 0000

仅当 CAN_FMR 寄存器中设置了过滤器初始化模式 (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 1027 页的图 387：过滤器组尺度配置 - 寄存器构成。*

位 31:28 保留，必须保持复位值。

位 27:0 **FBM*x***：过滤器模式 (Filter mode)
过滤器 x 的寄存器的模式。
0：过滤器组 x 的两个 32 位寄存器处于标识符屏蔽模式。
1：过滤器组 x 的两个 32 位寄存器处于标识符列表模式。

注： *位 27:14 仅用于双 CAN 配置，对于单 CAN 配置则为保留位。*

CAN 过滤器尺度寄存器 (CAN_FS1R)

CAN filter scale register

偏移地址：0x20C
复位值：0x0000 0000

仅当 CAN_FMR 寄存器中设置了过滤器初始化模式 (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 **FSCx**：过滤器尺度配置 (Filter scale configuration)
这些位用于定义过滤器 27-0 尺度配置。
0：双 16 位尺度配置
1：单 32 位尺度配置

注： *位 27:14 仅用于双 CAN 配置，对于单 CAN 配置则为保留位。*

注： *请参见第 1027 页的图 387：过滤器组尺度配置 - 寄存器构成。*

CAN 过滤器 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 寄存器中设置了过滤器初始化模式 (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***：过滤器 x 的过滤器 FIFO 分配 (Filter FIFO assignment for filter x)
通过此过滤器的消息将存储在指定的 FIFO 中。
0：过滤器分配给 FIFO 0
1：过滤器分配给 FIFO 1

注： 位 27:14 仅用于双 CAN 配置，对于单 CAN 配置则为保留位。

CAN 过滤器激活寄存器 (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.	FACT 27	FACT 26	FACT 25	FACT 24	FACT 23	FACT 22	FACT 21	FACT 20	FACT 19	FACT 18	FACT 17	FACT 16
				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
FACT 15	FACT 14	FACT 13	FACT 12	FACT 11	FACT 10	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***：过滤器激活 (Filter active)
软件将此位置 1 可激活过滤器 x。要修改过滤器 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：过滤器 x 未激活
1：过滤器 x 激活

注： 位 27:14 仅用于双 CAN 配置，对于单 CAN 配置则为保留位。

过滤器组 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

根据 CAN 外设配置的不同，过滤器组的数量也会有所不同。双 CAN 配置中*共有* 28 *个过滤器组*，而单 CAN 配置中共有 14 个过滤器组。每个过滤器组 i（双 CAN 配置中 i = 0 至 27，单 CAN 配置中 i = 0 至 13）由两个 32 位寄存器 CAN_FiR[2:1] 组成。

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]**：过滤器位 (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过滤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过滤器的尺度和模式配置而异。有关过滤器映射、功能说明和掩码寄存器关联的信息，请参见第 1025 页上的第 32.7.4 节：标识符过滤。*

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。

有关过滤器组的寄存器映射/地址信息，请参见第 1056 页的表 218。

32.9.5 bxCAN 寄存器映射

有关寄存器边界地址的信息，请参见*第 54 页的第 2.2.2 节*。寄存器仅在 CAN1 和 CAN3 中位于偏移量为 0x200 到 0x31C 的地址处。

表 218. 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.	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 218. 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	x	-

表 218. 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.

表 218. 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
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
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

33 USB on-the-go 全速 (OTG_FS)

33.1 简介

本节介绍了 OTG_FS 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
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)
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 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
- 电池充电规范第 1.2 版

USB 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*。OTG_FS 支持在下面的*表 219：OTG_FS 支持的速度*中定义的速度。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。

表 219. OTG_FS 支持的速度

-	HS (480 Mb/s)	FS (12 Mb/s)	LS (1.5 Mb/s)
主机模式	-	X	X
设备模式	-	X	-

33.2 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。

33.2.1 通用特性

OTG_FS 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
- 模块内嵌的 PHY 还完全支持定义在标准规范 OTG 补充第 2.0 版中的 OTG 协议
  - 支持 A-B 设备识别（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 从机（B 设备）
  - 具有 SRP 功能的 USB FS/LS 主机（A 设备）
  - USB On-The-Go 全速双角色设备
- 支持 FS SOF 和 LS Keep-alive 令牌
  - SOF 脉冲可通过 PAD 输出
  - SOF 脉冲通过内部连接到定时器 (TIMx)
  - 可配置的帧周期
  - 可配置的帧结束中断
- 具有省电功能，例如在 USB 挂起期间停止系统、关闭数字模块内部时钟域、对 PHY 和 DFIFO 电源加以管理。
- 具有采用高级 FIFO 控制的 1.25 KB 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
- 一帧 (1 ms) 之内可以无需应系统干预，以达到最大 USB 带宽。
- 它支持电池充电规范第 1.2 版中介绍的充电端口检测（仅限 FS PHY 收发器）。

33.2.2 主机模式特性

OTG_FS 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 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：
  - 在周期性硬件队列中存储多达 12 个中断加同步传输请求
  - 在非周期性硬件队列中存储多达 12 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。

33.2.3 从机模式特性

OTG_FS 接口在从机模式下具有以下特性：

- 1 个双向控制端点 0
- 5个 IN 端点 (EP)，可配置为支持批量传输、中断传输或同步传输
- 5个 OUT 端点，可配置为支持批量传输、中断传输或同步传输
- 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 个专用 Tx-IN FIFO（分别用于每个使能的 IN EP），以降低应用程序负荷
- 支持软断开功能。

33.2.4 USB 独立供电引脚

在某些封装选项中，可以通过 USB 独立供电引脚 V_{DDUSB} 给 USB 模块供电，其电压规范比 V_{DD} 更严格，使得后者 (V_{DD}) 可以在较低电压范围下工作（USB 模块通过 V_{DDUSB} 单独供电）。

在这种情况下，假设 V_{DD} 确实低于 V_{DDUSB} 的最小功能级别，请采取以下预防措施。

对于只有主机的应用：

- 无需采取特别的预防措施。

对于 VBUS 供电的 USB2.0 外设：

- 无需采取特别的预防措施。无需 V_{BUS} 检测，这是因为 V_{BUS} 状态直接通过 MCU 上是否存在电源来反映。

所有其它情况（需要 V_{BUS} 检测）：

- 请参见数据手册（电气特性和应用框图附录）。其他范围的 V_{DD} 电源将适用于这种情况。

33.3 OTG 实现

表 220. OTG 实现(1)

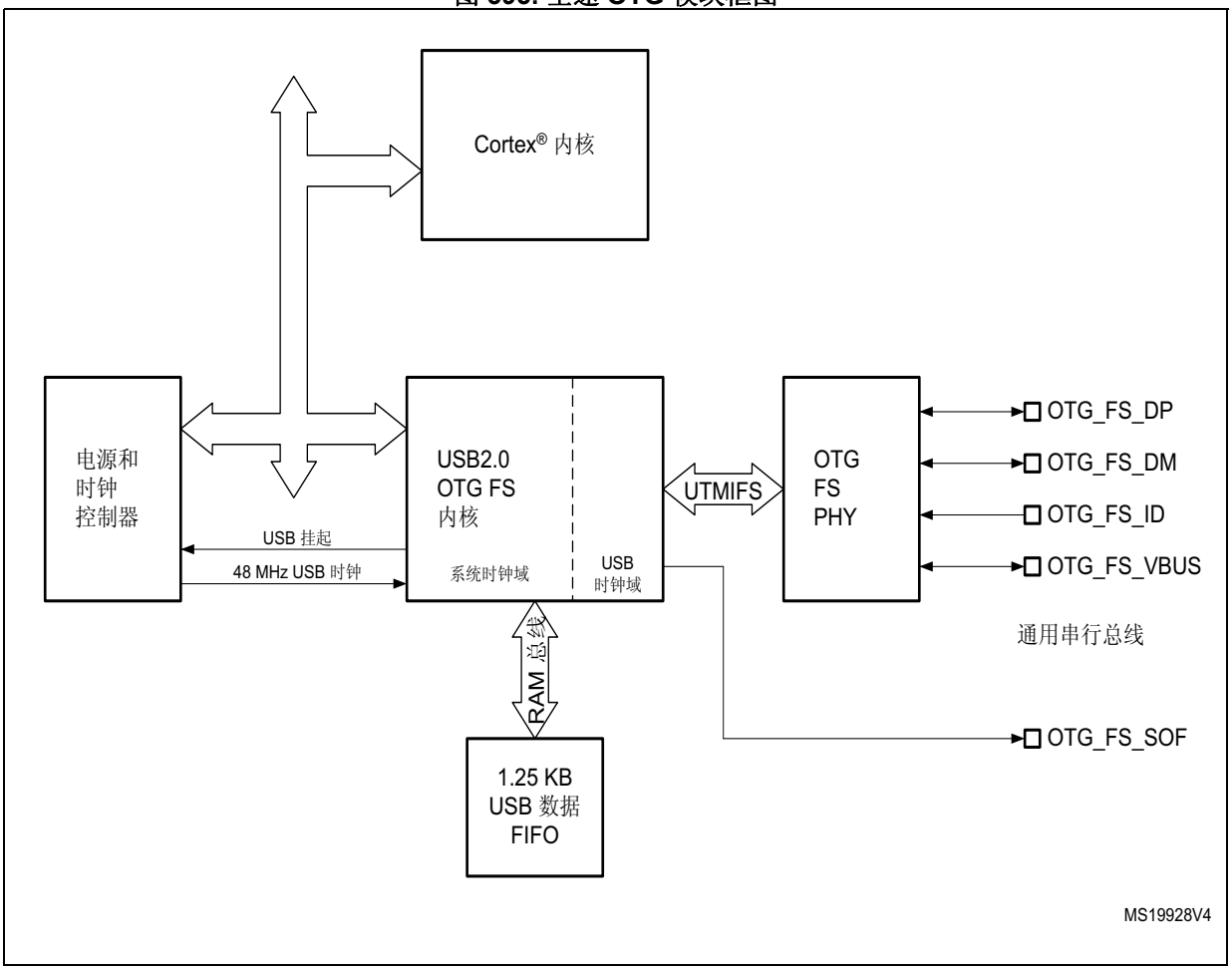
USB 功能	OTG_FS
设备双向端点（包括 EP0）	6
主机模式通道	12
专用 SRAM 的大小	1.2 KB
USB 2.0 链路层电源管理 (LPM)	X
支持 OTG 版本	2.0
连接检测协议 (ADP) 支持	-
电池充电检测 (BCD) 支持	X

1. “X” = 支持
“-” = 不支持

33.4 OTG 功能说明

33.4.1 OTG 框图

图 395. 全速 OTG 模块框图



33.4.2 USB OTG 引脚和内部信号

表 221. OTG_FS 输入/输出引脚

信号名称	信号类型	说明
OTG_FS_DP	数字输入/输出	USB OTG D+ 线
OTG_FS_DM	数字输入/输出	USB OTG D- 线
OTG_FS_ID	数字输入	USB OTG ID
OTG_FS_VBUS	模拟输入	USB OTG VBUS
OTG_FS_SOF	数字输出	USB OTG 帧起始位（可见）

表 222. OTG_FS 输入/输出信号

信号名称	信号类型	说明
usb_sof	数字输出	USB OTG 帧起始事件（用于片上外设）
usb_wkup	数字输出	USB OTG 唤醒事件输出
usb_gbl_it	数字输出	USB OTG 全局中断

33.4.3 OTG 模块

USB OTG 通过复位和时钟控制模块 (RCC) 接收来自外部晶振的 48 MHz 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 33.13 节：OTG_FS 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 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。

33.4.4 全速 OTG PHY

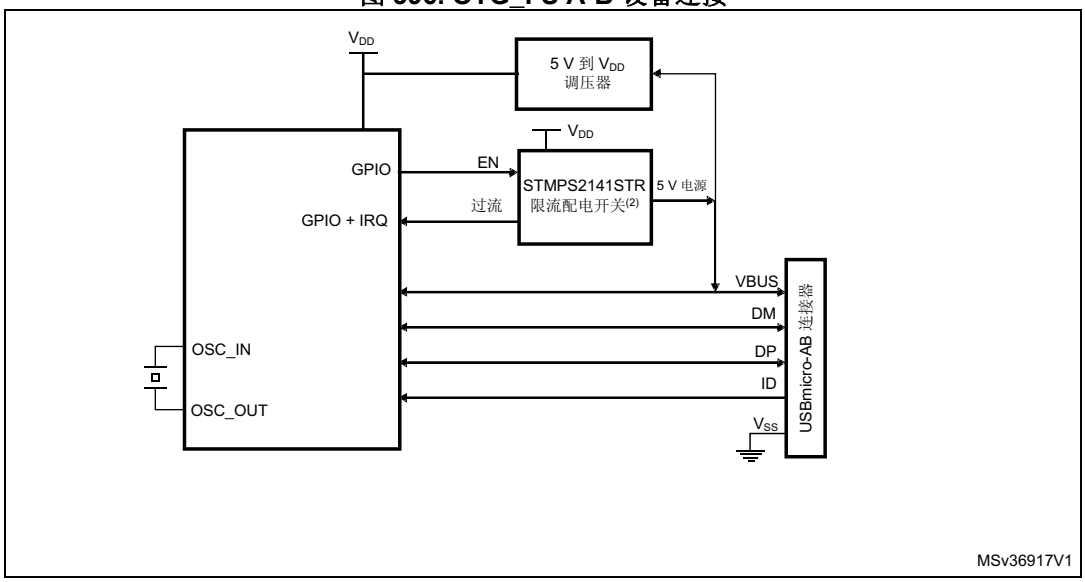
嵌入式全速 OTG PHY 由 OTG FS 模块控制，通过 UTMI+ 总线 (UTMIFS) 的全速子集传送 USB 控制和数据信号。为 USB 连接提供物理支持。
全速 OTG PHY 包括以下组成部分：

- 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 设备。
- 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 上拉电路包括两个由 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。

33.5 OTG 双角色设备 (DRD)

图 396. OTG_FS A-B 设备连接



1. 只有在构建由 VBUS 供电的设备时才需要外部调压器。
2. 只有在应用必须支持由 VBUS 供电的设备时才需要 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。

33.5.1 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 电缆时可根据是 MicroA 还是 MicroB 插头连接到 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 符合“USB2.0 On-The-Go 规范第 2.0 版补充标准中第 4.2.4 节 ID 引脚”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 将发出 ID 线状态更改中断（OTG_GINTSTS 中的 CIDSCHG 位）以初始化主机软件，并自动切换为主机角色。在此配置中，OTG_FS 符合“USB2.0 On-The-Go 规范第 2.0 版补充标准中第 4.2.4 节 ID 引脚”中所述的 FSM 标准。

33.5.2 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 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 33.16 节：OTG_FS 编程模型*详细介绍了 HNP 编程模型。

33.5.3 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 模块关闭 V_{BUS} 供电，为 A 设备节省电能。注意，无论 OTG_FS 采取主机角色还是从机角色，A 设备将始终负责 V_{BUS} 的提供。

*第 33.16 节：OTG_FS 编程模型*详细介绍了 SRP A/B 设备编程模型。

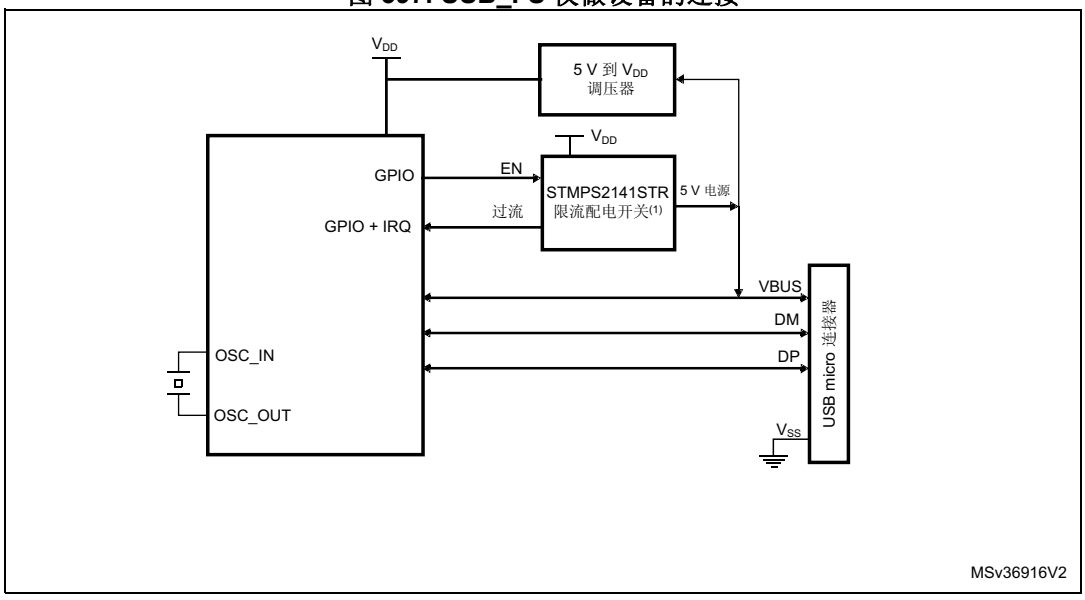
33.6 USB 设备

本节介绍了 OTG_FS 在 USB 设备模式下所具有的功能。在以下情形下，OTG_FS 用作 USB 设备：

- OTG B 设备
  - OTG B 设备插入 USB 电缆 B 端时的默认状态
- OTG A 设备
  - HNP 将 OTG_FS 切换到其设备角色后的 OTG A 设备状态
- B 设备
  - 如果 ID 线有连接，设备与 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。
- 仅作设备（请参见*图 397：USB_FS 仅做设备的连接*）
  - 将*第 33.15.4 节：OTG USB 配置寄存器 (OTG_GUSBCFG)* 中的强制设备模式位 (FDMOD) 置 1，从而将 OTG_FS 模块强制用作纯 USB 设备。这种情况下，即使 USB 连接器上存在 ID 线，也会将该 ID 线忽略。

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} 所需电源。*

图 397. USB_FS 仅做设备的连接



1. 使用调压器构建总线供电设备。

33.6.1 支持 SRP 功能的 USB 设备

全局 USB 配置寄存器中的 SRP 功能位（OTG_GUSBCFG 中的 SRPCAP 位）可使 OTG_FS 支持会话请求协议 (SRP)。这样一来，远程 A 设备便可以在 USB 会话挂起时，通过关闭 V_{BUS} 来节省电能。

*B 设备会话请求协议*一节详细介绍了 SRP 设备的编程模型。

33.6.2 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 部分）。然后，OTG_FS 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 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 已退出供电状态。

供电状态下，OTG_FS 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 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 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 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 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 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。

33.6.3 USB 设备端点

OTG_FS 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个 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个 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 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

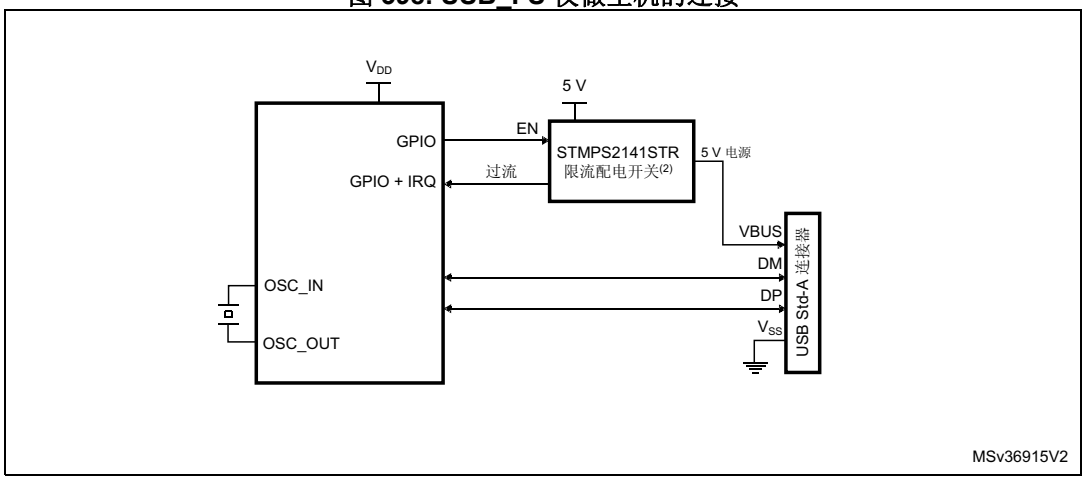
33.7 USB 主机

本节介绍了 OTG_FS 在 USB 主机模式下所具有的功能。在以下情形下，OTG_FS 用作 USB 主机：

- OTG A 主机
  - OTG A 设备在插入 USB 电缆 A 端时的默认状态。
- OTG B 主机
  - OTG B 设备被 HNP 切换为主机角色后的状态。
- A 主机
  - 如果 ID 线有连接，设备与 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
  - *OTG USB 配置寄存器 (OTG_GUSBCFG)* 中的强制主机模式位 (FHMOD) 将 OTG_FS 模块强制用作纯 USB 设备。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 设备和仅作主机配置都需要使用电荷泵。*

图 398. USB_FS 仅做主机的连接



1. V_{DD} 范围介于 2 V 到 3.6 V 之间。

33.7.1 支持 SRP 功能的主机

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 设备会话请求协议*一节详细介绍了 SRP 主机的编程模型。

33.7.2 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，或者通过连接至外部 PMIC（电源管理 IC）的 I^2C 接口驱动。当应用程序确定控制 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 (4.4 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。

在 HNP 和 SRP 同时关闭的情况下，无需将 VBUS 感应引脚连接到 V_{BUS}。

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 设备或 B 设备，但是 OTG_FS 也只有在 V_{BUS} 有效后 (5 V) 才能检测到设备的连接。当 V_{BUS} 处于有效电平且已连接远程 B 设备时，OTG_FS 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 设备将在连接后立即被检测到。OTG_FS 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 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。

33.7.3 主机通道

OTG_FS 模块实现了 12 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 个传输请求。如果应用程序有 12 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对通道进行分配。

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 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 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_HAINT 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

33.7.4 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 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：

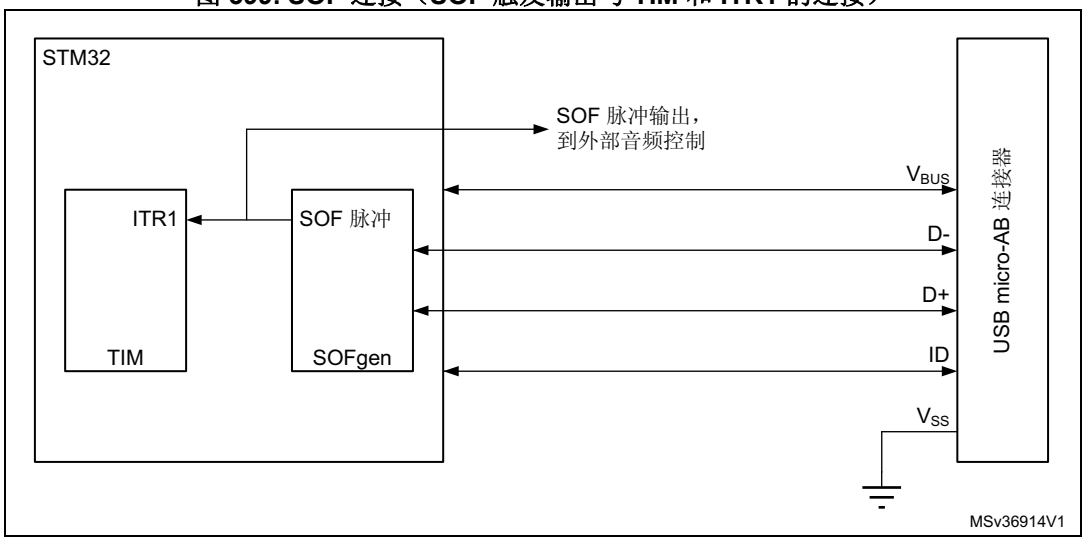
- 周期性（非周期性）请求队列中当前可用的空闲条目数（最多 8 个）
- 周期性（非周期性）Tx FIFO（OUT 事务）中当前可用的空闲空间
- IN/OUT 令牌、主机通道编号和其它状态信息。

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。

33.8 SOF 触发

图 399. SOF 连接（SOF 触发输出与 TIM 和 ITR1 的连接）



OTG_FS 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。

33.8.1 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 (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。

SOF 令牌发出的同时会产生 SOF 脉冲信号，并且宽度为 20 个 HCLK 周期。此外，SOF 脉冲还与定时器的输入触发相连，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。

33.8.2 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。还可以生成宽度为 20 个 HCLK 周期的 SOF 脉冲信号。此外，SOF 脉冲信号还在内部与 TIM 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。

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。

33.9 OTG 低功耗模式

下面的*表 223* 定义了 STM32 低功耗模式及其与 OTG 的兼容性。

表 223. STM32 低功耗模式与 OTG 的兼容性

模式	说明	USB 兼容性
运行	MCU 完全激活	USB 未处于挂起状态时需要。
睡眠	USB 挂起退出状态可使器件退出睡眠模式。外设寄存器内容保持不变。	USB 处于挂起状态时可用。
停止	USB 退出挂起状态可使器件退出停止模式。外设寄存器内容保持不变(1)。	USB 处于挂起状态时可用。
待机	掉电。在退出待机模式后，必须重新初始化外设。	与 USB 应用程序不兼容。

1. 在停止模式下，有不同的可能设置。也可能存在一些限制，请参见*第 5 节：电源控制器 (PWR)* 来了解使用 OTG 时适用的限制（如果存在）。

以下位和过程降低了功耗。

OTG PHY 的功耗由通用模块配置寄存器中的两个或三个位控制，具体取决于 OTG 支持的版本。

- 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 Gate HCLK 位置 1 时，OTG_FS 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。即使应用程序出于其它用途仍提供系统时钟，也会节省掉由于 USB 时钟信号翻转带来的动态功耗。
- USB 系统停止
  当 OTG_FS 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 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 模块访问时为其提供时钟。

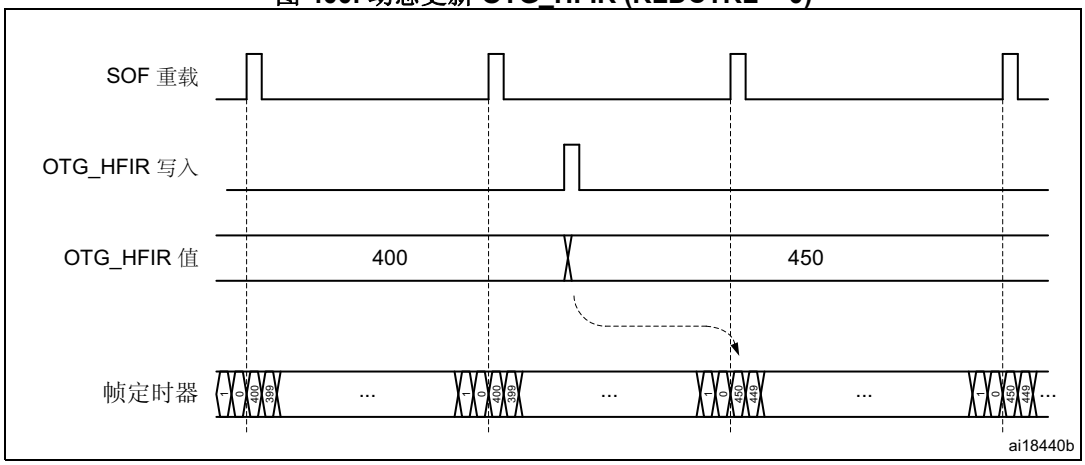
33.10 动态更新 OTG_HFIR 寄存器

主机模式下，USB 模块具有对 SOF 帧周期进行动态微调的功能，能够将外部设备与 SOF 帧进行同步。

如果 OTG_HFIR 寄存器在当前 SOF 帧内发生更改，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 400*。

对于动态更新，需要设置 RLDCTRL=0。

图 400. 动态更新 OTG_HFIR (RLDCTRL = 0)

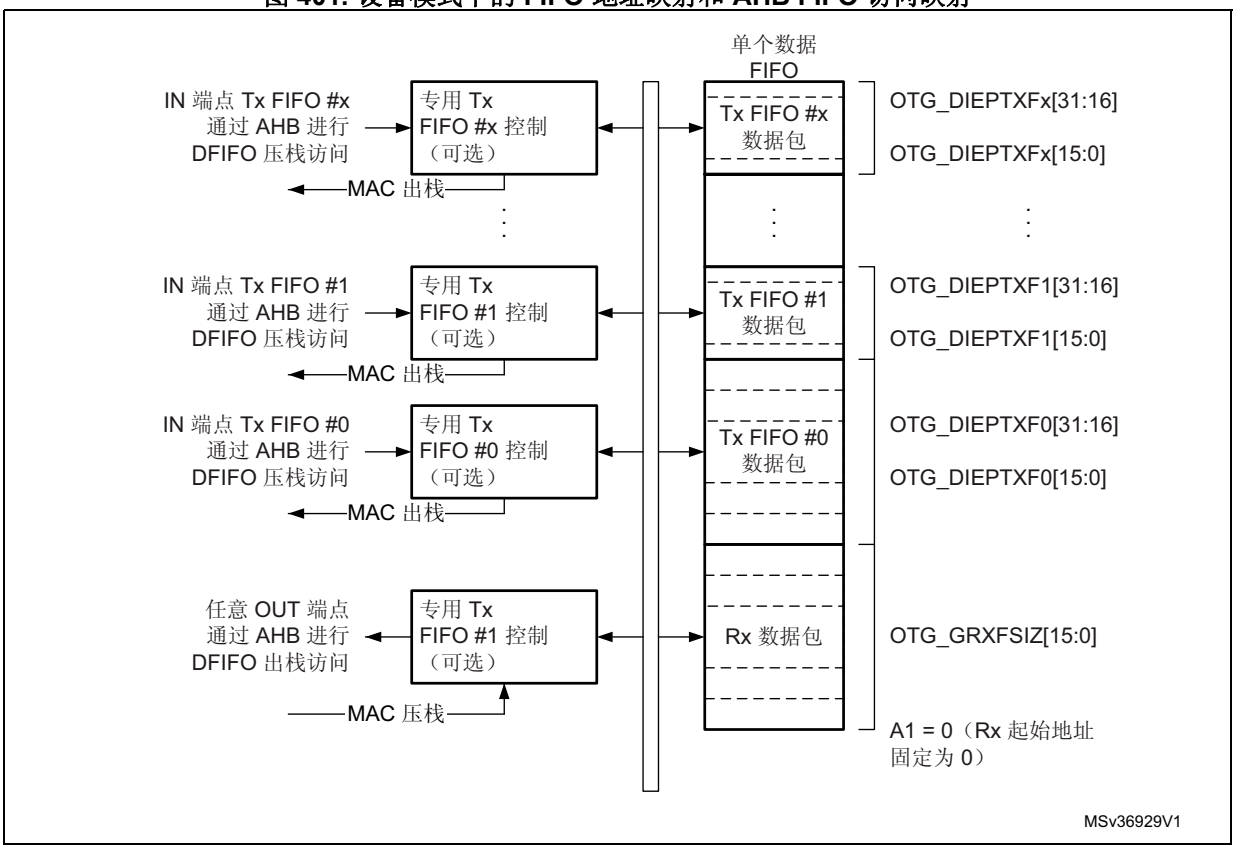


33.11 USB 数据 FIFO

USB 系统具有 1.25 KB 专用 RAM，采用复杂的 FIFO 控制机制。OTG_FS 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。

33.11.1 设备 FIFO 架构

图 401. 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 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

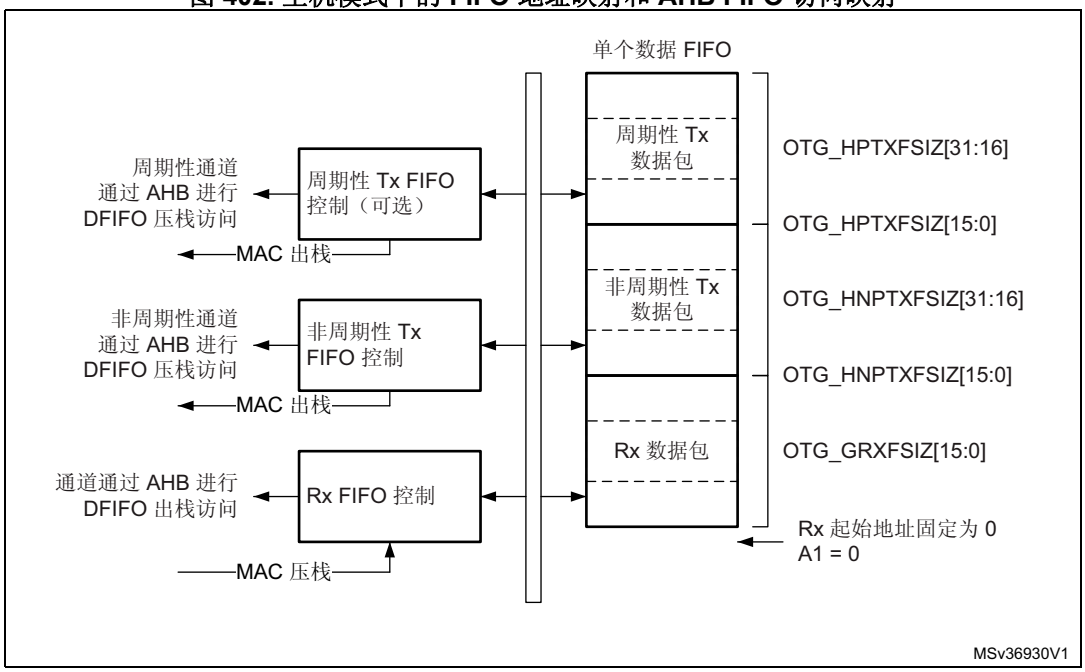
只要至少有一个数据包在 Rx FIFO 中可供读取，应用程序就会一直接收 Rx FIFO 非空中断（OTG_GINTSTS 中的 RXFLVL 位）。应用程序从接收状态读取和出栈寄存器 (OTG_GRXSTSP) 中读取数据包信息，最后通过读取与端点相关的出栈地址从接收 FIFO 读出相应数据。

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。

33.11.2 主机 FIFO 架构

图 402. 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 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 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 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 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。

33.11.3 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。
- 将会为全局 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。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。

器件 RxFIFO =

（5 * 控制端点数量 + 8）+（（所使用的最大 USB 数据包/4）+ 用于状态信息的 1）+（2 * OUT 端点数量）+ 用于全局 NAK 的 1

例如：周期性 USB 数据包的 MPS 是 1024 个字节，非周期性 USB 数据包的 MPS 是 512 个字节。有三个 OUT 端点、三个 IN 端点、一个控制端点和三个主机通道。

则器件 RxFIFO = (5 * 1 + 8) + ((1024/4) +1) + (2 * 4) + 1 = 279

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。

主机 RxFIFO =（（所用的最大 USB 数据包/4）+ 用于状态信息的 1）+ 用于传输完成的 1

例如：主机 RxFIFO = ((1024/4) + 1)+ 1 = 258

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，CPU 可获得下一个数据包。

非周期性 TxFIFO = 所用的最大非周期性 USB 数据包/4

例如：非周期性 TxFIFO = (512/4) = 128

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。

主机周期性 TxFIFO = 所用的最大周期性 USB 数据包/4

例如：主机周期性 TxFIFO = (1024/4) = 256

注： *为非周期性发送 FIFO 分配的空间越多，USB 的性能就越高。*

33.12 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

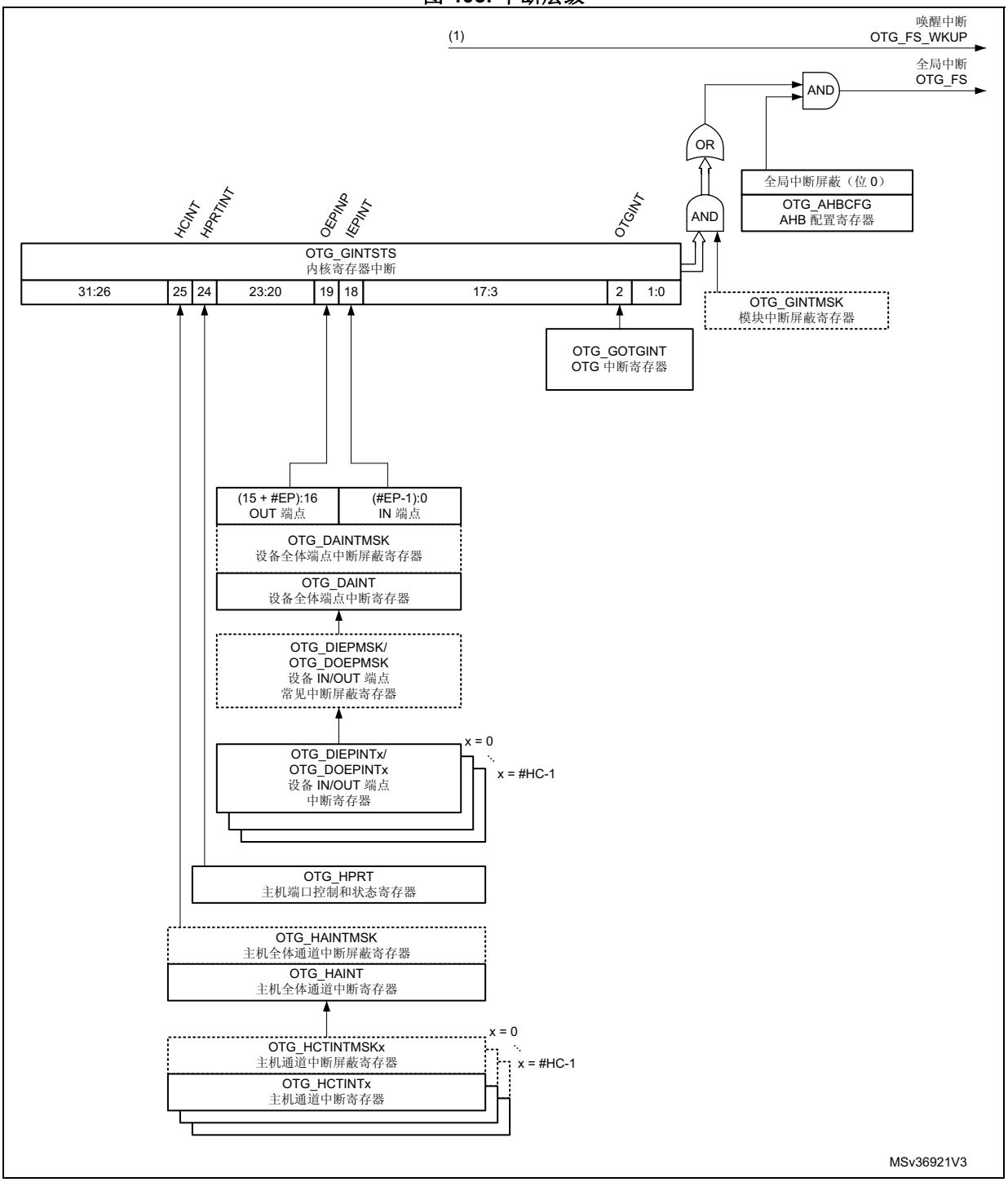
由于 OTG_FS 模块能够高效填充 1.25 KB RAM 缓冲区且 1.25 KB 发送/接收数据足以满足一个全速帧所能容纳的数据量，因此 USB 系统在一帧 (1 ms) 之内可以无需 CPU 干预达到最大 USB 带宽。

33.13 OTG_FS 中断

当 OTG_FS 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 403 显示了中断层级。

图 403. 中断层级



1. 处于 L1 睡眠或 L2 挂起状态期间发生恢复条件时，OTG_FS_WKUP 变为有效状态（高电平状态）。

33.14 OTG_FS 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 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 32 位的方式访问 OTG_FS 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 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。

33.14.1 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 224. 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 33.15.1 节：OTG 控制和状态寄存器 (OTG_GOTGCTL)*
OTG_GOTGINT	0x004	*第 33.15.2 节：OTG 中断寄存器 (OTG_GOTGINT)*
OTG_GAHBCFG	0x008	*第 33.15.3 节：OTG AHB 配置寄存器 (OTG_GAHBCFG)*
OTG_GUSBCFG	0x00C	*第 33.15.4 节：OTG USB 配置寄存器 (OTG_GUSBCFG)*
OTG_GRSTCTL	0x010	*第 33.15.5 节：OTG 复位寄存器 (OTG_GRSTCTL)*
OTG_GINTSTS	0x014	*第 33.15.6 节：OTG 模块中断寄存器 (OTG_GINTSTS)*
OTG_GINTMSK	0x018	*第 33.15.7 节：OTG 中断屏蔽寄存器 (OTG_GINTMSK)*
OTG_GRXSTSR	0x01C	*第 33.15.8 节：OTG 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 33.15.9 节：OTG 接收 FIFO 大小寄存器 (OTG_GRXFSIZ)*

表 224. 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_HNPTXFSIZ/ OTG_DIEPTXF0(1)	0x028	*第 33.15.10 节：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 33.15.11 节：OTG 非周期性发送 FIFO/队列状态寄存器 (OTG_HNPTXSTS)*
OTG_GCCFG	0x038	*第 33.15.12 节：OTG 通用模块配置寄存器 (OTG_GCCFG)*
OTG_CID	0x03C	*第 33.15.13 节：OTG 模块 ID 寄存器 (OTG_CID)*
OTG_GLPMCFG	0x54	*第 33.15.14 节：OTG 模块 LPM 配置寄存器 (OTG_GLPMCFG)*
OTG_HPTXFSIZ	0x100	*第 33.15.15 节：OTG 主机周期性发送 FIFO 大小寄存器 (OTG_HPTXFSIZ)*
OTG_DIEPTXFx	0x104 0x108 ... 0x114	*第 33.15.16 节：OTG 设备 IN 端点发送 FIFO 大小寄存器 (OTG_DIEPTXFx)（x = 1..5，其中 x 为 FIFO 编号）*

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 225. 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 33.15.18 节：OTG 主机配置寄存器 (OTG_HCFG)*
OTG_HFIR	0x404	*第 33.15.19 节：OTG 主机帧间隔寄存器 (OTG_HFIR)*
OTG_HFNUM	0x408	*第 33.15.20 节：OTG 主机帧编号/帧剩余时间寄存器 (OTG_HFNUM)*
OTG_HPTXSTS	0x410	*第 33.15.21 节：OTG_Host 周期性发送 FIFO/队列状态寄存器 (OTG_HPTXSTS)*
OTG_HAINT	0x414	*第 33.15.22 节：OTG 主机全体通道中断寄存器 (OTG_HAINT)*
OTG_HAINTMSK	0x418	*第 33.15.23 节：OTG 主机全体通道中断屏蔽寄存器 (OTG_HAINTMSK)*
OTG_HPRT	0x440	*第 33.15.24 节：OTG 主机端口控制和状态寄存器 (OTG_HPRT)*
OTG_HCCHARx	0x500 0x520 ... 0x660	*第 33.15.25 节：OTG 主机通道 x 特性寄存器 (OTG_HCCHARx)（x = 0..11，其中 x 表示通道编号）*
OTG_HCINTx	0x508 0x528 .... 0x668	*第 33.15.26 节：OTG 主机通道 x 中断寄存器 (OTG_HCINTx)（x = 0..11，其中 x 表示通道编号）*

表 225. 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_HCINTMSKx	0x50C 0x52C 0x66C	*第 33.15.27 节：OTG 主机通道 x 中断屏蔽寄存器 (OTG_HCINTMSKx)（x = 0..11，其中 x 表示通道编号）*
OTG_HCTSIZx	0x510 0x530 .... 0x670	*第 33.15.28 节：OTG 主机通道 x 传输大小寄存器 (OTG_HCTSIZx)（x = 0..11，其中 x 表示通道编号）*

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 226. 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 33.15.30 节：OTG 设备配置寄存器 (OTG_DCFG)*
OTG_DCTL	0x804	*第 33.15.31 节：OTG 设备控制寄存器 (OTG_DCTL)*
OTG_DSTS	0x808	*第 33.15.32 节：OTG 设备状态寄存器 (OTG_DSTS)*
OTG_DIEPMSK	0x810	*第 33.15.33 节：OTG 设备 IN 端点通用中断屏蔽寄存器 (OTG_DIEPMSK)*
OTG_DOEPMSK	0x814	*第 33.15.34 节：OTG 设备 OUT 端点通用中断屏蔽寄存器 (OTG_DOEPMSK)*
OTG_DAINT	0x818	*第 33.15.35 节：OTG 设备全体端点中断寄存器 (OTG_HAINT)*
OTG_DAINTMSK	0x81C	*第 33.15.36 节：OTG 全体端点中断屏蔽寄存器 (OTG_DAINTMSK)*
OTG_DVBUSDIS	0x828	*第 33.15.37 节：OTG 设备 VBUS 放电时间寄存器 (OTG_DVBUSDIS)*
OTG_DVBUSPULSE	0x82C	*第 33.15.38 节：OTG 设备 VBUS 脉冲时间寄存器 (OTG_DVBUSPULSE)*
OTG_DIEPEMPMSK	0x834	*第 33.15.39 节：OTG 设备 IN 端点 FIFO 空中断屏蔽寄存器 (OTG_DIEPEMPMSK)*
OTG_DIEPCTL0	0x900	*第 33.15.40 节：OTG 设备控制 IN 端点 0 控制寄存器 (OTG_DIEPCTL0)*
OTG_DIEPCTLx	0x920 0x940 ... 0x9A0	*第 33.15.41 节：OTG 设备 IN 端点 x 控制寄存器 (OTG_DIEPCTLx)（x = 1..5，其中 x 表示端点编号）*
OTG_DIEPINTx	0x908 0x928 .... 0x988	*第 33.15.42 节：OTG 设备 IN 端点 x 中断寄存器 (OTG_DIEPINTx)（x = 0..5，其中 x 表示端点编号）*

表 226. 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_DIEPTSIZ0	0x910	*第 33.15.43 节：OTG 设备 IN 端点 0 传输大小寄存器 (OTG_DIEPTSIZ0)*
OTG_DTXFSTSx	0x918 0x938 .... 0x998	*第 33.15.44 节：OTG 设备 IN 端点发送 FIFO 状态寄存器 (OTG_DTXFSTSx)（x = 0..5，其中 x = 端点编号）*
OTG_DIEPTSIZx	0x930 0x950 ... 0x9B0	*第 33.15.45 节：OTG 设备 IN 端点 x 传输大小寄存器 (OTG_DIEPTSIZx)（x = 1..5，其中 x 表示端点编号）*
OTG_DOEPCTL0	0xB00	*第 33.15.46 节：OTG 设备控制 OUT 端点 0 控制寄存器 (OTG_DOEPCTL0)*
OTG_DOEPINTx	0xB08 0xB28 ... 0xBA8	*第 33.15.47 节：OTG 设备 OUT 端点 x 中断寄存器 (OTG_DOEPINTx)（x = 0..5，其中 x 表示端点编号）*
OTG_DOEPTSIZ0	0xB10	*第 33.15.48 节：OTG 设备 OUT 端点 0 传输大小寄存器 (OTG_DOEPTSIZ0)*
OTG_DOEPCTLx	0xB20 0xB40 ... 0xBA0	*第 33.15.49 节：OTG 设备 OUT 端点 x 控制寄存器 (OTG_DOEPCTLx)（x = 1..5，其中 x 表示端点编号）*
OTG_DOEPTSIZx	0xB30 0xB50 ... 0xBB0	*第 33.15.50 节：OTG 设备 OUT 端点 x 传输大小寄存器 (OTG_DOEPTSIZx)（x = 1..5， 其中 x 表示端点编号）*

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 227. 数据 FIFO (DFIFO) 访问寄存器地址映射

FIFO 访问寄存器段	偏移地址	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

表 227. 数据 FIFO (DFIFO) 访问寄存器地址映射（续）

FIFO 访问寄存器段	偏移地址	访问
...	...	...
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，x 为 5（设备模式）或 11（主机模式）。

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 228. 电源和时钟门控控制和状态寄存器

缩略语	偏移地址	寄存器名称
OTG_PCGCCTL	0xE00–0xE04	*第 33.15.51 节：OTG 电源和时钟门控控制寄存器 (OTG_PCGCCTL)*

33.15 OTG_FS 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。

33.15.1 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.	CUR MOD	OTG VER	BSVLD	ASVLD	DBCT	CID STS
										r	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	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

位 31:22 保留，必须保持复位值。

位 21 **CURMOD**：当前工作模式 (Current mode of operation)

该位指示当前模式（主机或设备）。

0：设备模式

1：主机模式

位 20 **OTGVER**：OTG 版本 (OTG version)

选择 OTG 版本。

0：OTG 版本 1.3。OTG1.3 在新产品开发中遭到淘汰。

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 控制器处于 A 设备模式

1：OTG_FS 控制器处于 B 设备模式

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 设备状态机和嵌入式主机状态机之间选择。

0：选择 OTG A 设备状态机

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 设备会话有效覆盖值 (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 设备会话有效覆盖使能 (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 设备会话有效覆盖值 (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 设备会话有效覆盖使能 (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 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。

如果用户使用 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。

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。

33.15.2 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.	ID CHNG	DBC DNE	ADTO CHG	HNG DET	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.	HNSS CHG	SRSS CHG	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 设备超时更改 (A-device timeout change)

模块将该位置 1 时，指示 A 设备在等待 B 设备连接时超时。

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 设备会话。

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。

33.15.3 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

位 31:9 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，必须保持复位值。

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。

33.15.4 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		

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 28:26 保留，必须保持复位值。

位 22 保留，必须保持复位值。

位 15 保留，必须保持复位值。

位 14 保留，必须保持复位值。

位 13:10 **TRDT[3:0]**：USB 周转时间 (USB turnaround time)

这些位允许以 PHY 时钟数为单位设置周转时间。必须根据*表 229：TRDT 值 (FS)* 来配置这些位，具体取决于应用程序 AHB 频率。TRDT 值越高，USB 对 IN 令牌的响应时间就越长，从而可以弥补 AHB 对数据 FIFO 的较长读访问延迟。

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 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 控制器的 SRP 功能。如果模块作为无 SRP 功能的 B 设备，则无法请求连接的 A 设备（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 保留，必须保持复位值。

位 4 保留，必须保持复位值。

位 3 保留，必须保持复位值。

位 2:0 **TOCAL[2:0]**：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。

表 229. TRDT 值 (FS)

AHB 频率范围 (MHz)		TRDT 最小值
最小值	最大值	
14.2	15	0xF
15	16	0xE
16	17.2	0xD
17.2	18.5	0xC
18.5	20	0xB
20	21.8	0xA
21.8	24	0x9
24	27.5	0x8
27.5	32	0x7
32	-	0x6

33.15.5 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															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	r

位 31 **AHBIDL**：AHB 主设备空闲 (AHB master idle)

指示 AHB 主设备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，必须保持复位值。

位 10:6 **TXFNUM[4:0]**：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

将所有模块状态机（AHB 从设备除外）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 主设备上的所有事务。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 主设备空闲）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。

33.15.6 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/ IN COMP ISO OUT	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

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 LPMEN 位置 1 时，该字段才有效。

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 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 22 保留，必须保持复位值

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，指示速率枚举已完成。应用程序必须读取 OTG_DSTS 寄存器来获取枚举速率。

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： 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。

33.15.7 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	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	

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： 仅可在主机模式下访问。

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，必须保持复位值

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。

33.15.8 OTG 接收状态调试读取/OTG 状态读取和出栈寄存器 (OTG_GRXSTSR/OTG_GRXSTSP)

OTG 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[3:0]				DPID
											r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPID	BCNT[10:0]											CHNUM[3:0]			
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[3:0]**：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：数据翻转错误（触发中断）
0111：通道停止（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

位 14:4 **BCNT[10:0]**：字节数 (Byte count)
指示接收的 IN 数据包的字节数。

位 3:0 **CHNUM[3:0]**：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.	STSPHST	Res.	Res.	FRMNUM[3:0]				PKTSTS[3:0]				DPID[1]
				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
DPID[0]	BCNT[10:0]											EPNUM[3:0]			
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:28 保留，必须保持复位值。

位 27 **STSPHST**：状态阶段开始 (Status phase start)
指示控制写传输的状态阶段开始。该位与 OUT 传输完成的 PKTSTS 模式一同置 1。

位 26:25 保留，必须保持复位值。

位 24:21 **FRMNUM[3:0]**：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[3:0]**：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[1:0]**：数据 PID (Data PID)
指示接收的 OUT 数据包的数据 PID
00：DATA0
10：DATA1

位 14:4 **BCNT[10:0]**：字节数 (Byte count)
指示接收的数据包的字节数。

位 3:0 **EPNUM[3:0]**：端点编号 (Endpoint number)
指示当前接收的数据包所属的端点编号。

33.15.9 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[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 **RXFD[15:0]**：Rx FIFO 深度 (Rx 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。

33.15.10 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/Endpoint 0 Transmit FIFO size

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[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
NPTXFSA/TX0FSA[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

主机模式

位 31:16 **NPTXFD[15:0]**：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[15:0]**：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。

33.15.11 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[6:0]							NPTQXSAV[7:0]							
	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
NPTXFSAV[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31 保留，必须保持复位值。

位 30:24 **NPTXQTOP[6:0]**：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[7:0]**：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[15:0]**：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 个位置可用（其中，0 ≤ n ≤ 512）
其它值：保留

33.15.12 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 XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VBDEN	SDEN	PDEN	DCD EN	BCDEN	PWR DWN
										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.	PS2 DET	SDET	PDET	DCDET
												r	r	r	r

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 **SDEN**：二次检测 (SD) 模式使能 (Secondary detection (SD) mode enable)

该位由软件置 1，用于将 BCD 置于 SD 模式。要正确工作，应只选择一个检测模式（DCD、PD、SD 或 OFF）

位 19 **PDEN**：初次检测 (PD) 模式使能 (Primary detection (PD) mode enable)

该位由软件置 1，用于将 BCD 置于 PD 模式。要正确工作，应只选择一个检测模式（DCD、PD、SD 或 OFF）。

位 18 **DCDEN**：数据接触点检测 (DCD) 模式使能 (Data contact detection (DCD) mode enable)

该位由软件置 1，用于将 BCD 置于 DCD 模式。要正确工作，应只选择一个检测模式（DCD、PD、SD 或 OFF）。

位 17 **BCDEN**：电池充电器检测 (BCD) 使能 (Battery charging detector (BCD) enable)
该位由软件置 1，用于在 USB 设备内使能 BCD 支持。使能后，USB PHY 完全由 BCD 控制，无法用于正常通信。BCD 检测完成后，应通过将该位清零使 BCD 置于 OFF 模式，以实现正常的 USB 操作。

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。置 1 时，必须关闭 BCD 功能 (BCDEN=0)。
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 **PS2DET**：DM 上拉检测状态 (DM pull-up detection status)
该位只在 PD 期间有效，用于指示 DM 电压电平与 VLGC 阈值之间的比较结果。正常情况下，DM 电平应低于此阈值。如果 DM 电平高于此阈值，则意味着 DM 已外部拉为高电平。原因可能是连接到 PS2 端口（同时上拉 DP 和 DM 线）或某些不遵循 BCD 规范的专有充电器。
0：检测到正常端口（连接到 SDP、CDP 或 DCP）
1：检测到 PS2 端口或专有充电器

位 2 **SDEN**：二次检测 (SD) 状态 (Secondary detection (SD) status)
此位指示 SD 的结果。
0：检测到 CDP
1：检测到 DCP

位 1 **PDEN**：初次检测 (PD) 状态 (Primary detection (PD) status)
此位指示 PD 的结果。
0：未检测到 BCD 支持（连接到 SDP 或专有设备）。
1：检测到 BCD 支持（连接到 CDP 或 DCP）。

位 0 **DCDEN**：数据接触点检测 (DCD) 状态 (Data contact detection (DCD) status)
此位指示 DCD 的结果。
0：未检测到数据点接触
1：检测到数据点接触

33.15.13 OTG 模块 ID 寄存器 (OTG_CID)

OTG core ID register

偏移地址：0x03C

复位值：0x0000 2000

这是一个包含产品 ID 作为复位值的寄存器。

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

位 31:0 **PRODUCT_ID[31:0]**：产品 ID 字段 (Product ID field)
可通过应用程序编程的 ID 字段。

33.15.14 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[2:0]			SND LPM	LPMRCNT[2:0]			LPMCHIDX[3:0]				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[1:0]		L1DS EN	BESLTHRS[3:0]				L1SS EN	REM WAKE	BESL[3:0]				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[2:0]**：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[2:0]**：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[3:0]**：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[1:0]**：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[3:0]**：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] 主机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。

设备模式（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[3:0]**：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。

设备模式（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)

设备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 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

33.15.15 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[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
PTXSA[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 **PTXFSIZ[15:0]：**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[15:0]：**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。

33.15.16 OTG 设备 IN 端点发送 FIFO 大小寄存器 (OTG_DIEPTXFx)（x = 1..5，其中 x 为 FIFO 编号）

OTG device IN endpoint transmit FIFO size register

偏移地址：0x104 + (x – 1) * 0x04

复位值：0x0200 0200 + (x * 0x200)

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

位 31:16 **INEPTXFD[15:0]：**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[15:0]：**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。

33.15.17 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：

33.15.18 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.

15	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[1:0]	
													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[1:0]**：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）。

33.15.19 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 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[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:17 保留，必须保持复位值。

位 16 **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：运行期间可动态重载 HFIR。

1：无法动态重载 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。

Caution: 不建议 RLDCTRL = 1。

位 15:0 **FRIVL[15:0]**：帧间隔 (Frame interval)

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。

–帧间隔 = 1 ms × (FRIVL - 1)

33.15.20 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[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
FRNUM[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 **FTREM[15:0]**：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[15:0]**：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。

33.15.21 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[7:0]								PTXQSAV[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
PTXFSAVL[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:24 **PTXQTOP[7:0]**：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[7:0]**：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[15:0]**：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

33.15.22 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 0000

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 403* 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[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 **HAINT[15:0]**：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

33.15.23 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[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 **HAINTM[15:0]**：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

33.15.24 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 403* 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[1:0]		PTCTL[3]
													r	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTCTL[2:0]			PPWR	PLSTS[1:0]		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_w1	rc_w1	r

位 31:19 保留，必须保持复位值。

位 18:17 **PSPD[1:0]**：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

位 16:13 **PTCTL[3:0]**：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[1:0]**：端口线状态 (Port line status)

指示 USB 数据线的当前逻辑电平

位 10：OTG_DP 的逻辑电平

位 11：OTG_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 WKUPINT 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 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；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

33.15.25 OTG 主机通道 x 特性寄存器 (OTG_HCCHARx)（x = 0..11，其中 x 表示通道编号）

OTG host channel x characteristics register

偏移地址：0x500 + (x * 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[6:0]							MCNT[1:0]		EPTYP[1:0]		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[3:0]				MPSIZ[10:0]										
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[6:0]**：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[1:0]**：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[1:0]**：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[3:0]**：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[10:0]**：最大包长（以字节为单位）(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。

33.15.26 OTG 主机通道 x 中断寄存器 (OTG_HCINTx)（x = 0..11，其中 x 表示通道编号）

OTG host channel x interrupt register

偏移地址：0x508 + (x * 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 403* 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

位 31:11 保留，必须保持复位值。

位 10 **DTERR**：数据翻转错误 (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，必须保持复位值。

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，必须保持复位值。

位 1 **CHH**：通道停止 (Channel halted)。
指示因任意 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。

33.15.27 OTG 主机通道 x 中断屏蔽寄存器 (OTG_HCINTMSKx)（x = 0..11，其中 x 表示通道编号）

OTG host channel x interrupt mask register

偏移地址：0x50C + (x * 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

位 31:11 保留，必须保持复位值。

位 10 **DTERRM**：数据翻转错误屏蔽 (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，必须保持复位值。

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 保留，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

33.15.28 OTG 主机通道 x 传输大小寄存器 (OTG_HCTSIZx)（x = 0..11，其中 x 表示通道编号）

OTG host channel x transfer size register

偏移地址：0x510 + (x * 0x20)

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	DPID[1:0]		PKTCNT[9:0]										XFRSIZ[18:16]		
	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[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30:29 **DPID[1:0]**：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
10：DATA1
11：SETUP（控制传输）/保留（非控制传输）

位 28:19 **PKTCNT[9:0]**：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[18:0]**：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。

33.15.29 设备模式寄存器

Device-mode registers

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。

33.15.30 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[1:0]		DAD[6:0]							Res.	NZLSOHSK	DSPD[1:0]	
rw			rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw

位 31:16 保留，必须保持复位值。

位 15 **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 13 保留，必须保持复位值。

位 12:11 **PFIVL[1:0]**：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[6:0]**：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 位发送握手信号。

位 1:0 **DSPD[1:0]**：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）

33.15.31 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.	DSBESLRJCT	Res.	Res.
													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	POPRGDNE	CGONAK	SGONAK	CGINAK	SGINAK	TCTL[2:0]			GONSTS	GINSTS	SDIS	RWUSIG
				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)

对此字段写入 1 会将全局 OUT NAK 清零。

位 9 **SGONAK**：将全局 OUT NAK 置 1 (Set global OUT NAK)

对此字段写入 1 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)

对此字段写入 1 会将全局 IN NAK 清零。

位 7 **SGINAK**：将全局 IN NAK 置 1 (Set global IN NAK)

对此字段写入 1 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[2:0]**：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：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 230 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 230. 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

33.15.32 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[1:0]		FNSOF[13:8]					
								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[7:0]								Res.	Res.	Res.	Res.	EERR	ENUMSPD[1:0]		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[1:0]**：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[13:0]**：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 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[1:0]**：枚举速度 (Enumerated speed)

指示 OTG_FS 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。

33.15.33 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.	TXFURM	Res.	INEPNEM	INEPNMM	ITTXFEMSK	TOM	Res.	EPDM	XFRCM
		rw					rw		rw	rw	rw	rw		rw	rw

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 12:10 保留，必须保持复位值。

位 9 保留，必须保持复位值。

位 8 **TXFURM**：FIFO 下溢中断屏蔽 (FIFO underrun mask)
0：屏蔽中断
1：使能中断

位 7 保留，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

33.15.34 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.	NYET MSK	NAK MSK	BERR M	Res.	Res.	Res.	OUT PKT ERRM	Res.	Res.	STS PHSR XM	OTEPD M	STUPM	Res.	EPDM	XFRC M
	rw	rw	rw				rw			rw	rw	rw		rw	rw

位 31:15 保留，必须保持复位值。

位 14 **NYETMSK**：NYET 中断屏蔽 (NYET interrupt mask)
0：屏蔽中断
1：使能中断

位 13 **NAKMSK**：NAK 中断屏蔽 (NAK interrupt mask)
0：屏蔽中断
1：使能中断

位 12 **BERRM**：Babble 错误中断屏蔽 (Babble error interrupt mask)
0：屏蔽中断
1：使能中断

位 11:10 保留，必须保持复位值。

位 9 保留，必须保持复位值。

位 8 **OUTPKTERRM**：输出数据包错误屏蔽 (Out packet error mask)
0：屏蔽中断
1：使能中断

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 **STSPHSRXM**：接收到控制写屏蔽的状态阶段 (Status phase received for control write mask)
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

33.15.35 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[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
IEPINT[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 **OEPINT[15:0]**：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 19。

位 15:0 **IEPINT[15:0]**：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。

33.15.36 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[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
IEPM[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16 **OEPM[15:0]**：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 19
0：屏蔽中断
1：使能中断

位 15:0 **IEPM[15:0]**：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

33.15.37 OTG 设备 V_{BUS} 放电时间寄存器 (OTG_DVBUSDIS)

OTG device VBUS 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[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 **VBUSDT[15:0]**：设备 V_{BUS} 放电时间 (Device VBUS 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。

33.15.38 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[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 **DVBUSP[15:0]**：设备 V_{BUS} 脉冲时间 (Device V_{BUS} pulsing time)。该功能仅与 OTG1.3 相关。
指定 SRP 期间的 V_{BUS} 脉冲时间。
V_{BUS} 脉冲时间（PHY 时钟数）/1024

33.15.39 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[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 **INEPTXFEM[15:0]**：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

33.15.40 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[3:0]				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[1:0]	
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[3:0]**：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 激活端点 (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[1:0]**：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

33.15.41 OTG 设备 IN 端点 x 控制寄存器 (OTG_DIEPCTLx)（x = 1..5，其中 x 表示端点编号）

OTG device IN endpoint x control register

偏移地址：0x900 + (x * 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[3:0]				STALL	Res.	EPTYP[1:0]		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[10:0]										
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 端点的该位置 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[1:0]**：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 激活端点 (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[10:0]**：最大包长（以字节为单位）(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。

33.15.42 OTG 设备 IN 端点 x 中断寄存器 (OTG_DIEPINTx)（x = 0..5，其中 x 表示端点编号）

OTG device IN endpoint x interrupt register

偏移地址：0x908 + (x * 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 403* 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断 (OTG_DAINT) 寄存器，以获取设备端点 x 中断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.	NAK	Res.	PKTD RPSTS	Res.	Res.	TXFIF OUD RN	TXFE	IN EPNE	IN EPNM	ITTXFE	TOC	Res.	EP DISD	XFRC
		rc_w1		rc_w1			rc_w1	r	r	rc_w1	rc_w1	rc_w1		rc_w1	rc_w1

位 31:14 保留，必须保持复位值。

位 13 **NAK**：NAK 输入 (NAK input)
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 保留，必须保持复位值。

位 11 **PKTDRPSTS**：数据包丢弃状态 (Packet dropped status)
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，必须保持复位值。

位 9 保留，必须保持复位值。

位 8 **TXFIFOUDRN**：发送FIFO 下溢 (TxfifoUndrn) (Transmit Fifo Underrun (TxfifoUndrn))
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 **INEPNM**：EP 不匹配时收到 IN 令牌 (IN token received with EP mismatch)
指示非周期性 TxFIFO 顶部数据所属的端点不是收到 IN 令牌的端点。从而产生中断。

位 4 **ITTXFE**：Tx FIFO 为空时接收到 IN 令牌 (IN token received when Tx FIFO is empty)
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)
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。

33.15.43 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。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[1:0]		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[6:0]						
									rw	rw	rw	rw	rw	rw	rw

位 31:21 保留，必须保持复位值。

位 20:19 **PKTCNT[1:0]**：数据包计数 (Packet count)

指示端点 0 的一次数据传输包含的 USB 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[6:0]**：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。

33.15.44 OTG 设备 IN 端点发送 FIFO 状态寄存器 (OTG_DTXFSTSx)（x = 0..5，其中 x = 端点编号）

OTG device IN endpoint transmit FIFO status register

IN 端点的偏移地址：0x918 + (x * 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[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 **INEPTFSAV[15:0]**：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

33.15.45 OTG 设备 IN 端点 x 传输大小寄存器 (OTG_DIEPTSIZx)（x = 1..5，其中 x 表示端点编号）

OTG device IN endpoint x transfer size register

偏移地址：0x910 + (x * 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。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[1:0]		PKTCNT[9:0]										XFRSIZ[18:16]		
	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[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30:29 **MCNT[1:0]**：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[9:0]**：数据包计数 (Packet count)
指示该端点上的一次数据传输包含的 USB 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[18:0]**：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。

33.15.46 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[1:0]		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[1:0]	
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[1:0]**：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 激活端点 (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[1:0]**：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

33.15.47 OTG 设备 OUT 端点 x 中断寄存器 (OTG_DOEPINTx)（x = 0..5，其中 x 表示端点编号）

OTG device OUT endpoint x interrupt register

偏移地址：0xB08 + (x * 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 403* 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.	NYET	NAK	BERR	Res.	Res.	Res.	OUT PKT ERR	Res.	Res.	STSPH SRX	OTEP DIS	STUP	Res.	EP DISD	XFRC
	rc_w1	rc_w1	rc_w1				rc_w1			rc_w1	rc_w1	rc_w1		rc_w1	rc_w1

位 31:16 保留，必须保持复位值。

位 15 保留，必须保持复位值。

位 14 **NYET**：NYET 中断 (NYET interrupt)

当非同步 OUT 端点回复 NYET 握手信号时将产生此中断。

位 13 **NAK**：NAK 输入 (NAK input)
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**：Babble 错误中断 (Babble error interrupt)
当端点收到 babble 时，模块会生成此中断。

位 11:10 保留，必须保持复位值。

位 9 保留，必须保持复位值。

位 8 **OUTPKTERR**：OUT 数据包错误 (OUT packet error)
当模块检测到 OUT 数据包发生上溢或 CRC 错误时，该中断会置为有效。该中断仅在阈值已使能时有效。

位 7 保留，必须保持复位值。

位 6 保留，必须保持复位值。

位 5 **STSPHSRX**：接收到控制写的状态阶段 (Status phase received for control write)
该中断仅对控制 OUT 端点有效。只有当 OTG_FS 将主机在控制写传输的数据阶段发送的所有数据全部传输到系统存储器缓冲区后，才会生成该中断。该中断向应用程序指示主机已从控制写传输的数据阶段切换到状态阶段。完成数据阶段的解码后，应用程序可使用该中断向状态阶段作出 ACK 或 STALL 响应。

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。

33.15.48 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。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 的寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	STUPCNT[1:0]		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[6:0]						
									rw	rw	rw	rw	rw	rw	rw

位 31 保留，必须保持复位值。

位 30:29 **STUPCNT[1:0]**：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[6:0]**：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。

33.15.49 OTG 设备 OUT 端点 x 控制寄存器 (OTG_DOEPCTLx)（x = 1..5，其中 x 表示端点编号）

OTG device OUT endpoint x control register

OUT 端点的偏移地址：0xB00 + (x * 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[1:0]		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[10:0]										
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 端点的该位置 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[1:0]**：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 激活端点 (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[10:0]**：最大包长（以字节为单位）(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。

33.15.50 OTG 设备 OUT 端点 x 传输大小寄存器 (OTG_DOEPTSIZx)（x = 1..5，其中 x 表示端点编号）

OTG device OUT endpoint x transfer size register

偏移地址：0xB10 + (x * 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。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[1:0]		PKTCNT[9:0]										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[1:0]**：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

10：DATA1

STUPCNT[1:0]：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[9:0]**：数据包计数 (Packet count)

指示该端点上的一次数据传输包含的 USB 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。

33.15.51 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：0x200B 8000

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	rw	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。

33.15.52 OTG_FS 寄存器映射

下表提供了 USB OTG 寄存器映射和复位值。

表 231. OTG_FS 寄存器映射和复位值

偏移	寄存器名	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	OTG_GOTGCTL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CURMOD	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	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	0
0x014	OTG_GINTSTS	WKUPINT	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

表 231. OTG_FS 寄存器映射和复位值（续）

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
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.	STSPHST	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	0
0x020	**OTG_ GRXSTSP (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_ GRXSTSP (Device mode)	Res.	Res.	Res.	Res.	STSPHST	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	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
0x038	**OTG_ GCCFG**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VBDEN	SDEN	PDEN	DCDEN	BCDEN	.PWRDWN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PS2DET	SDET	PDET	DCDET
	Reset value											0	0	0	0	0	0													X	X	X	X
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
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	

表 231. OTG_FS 寄存器映射和复位值（续）

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
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
. . . .	. . . .	. . . .																															
0x114	**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
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	FSLS PCS	
	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
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
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

表 231. OTG_FS 寄存器映射和复位值（续）

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
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		ACKM	NAKM	STALLM	Res.	CHHM	XFRCM
	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
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		ACKM	NAKM	STALLM	Res.	CHHM	XFRCM
	Reset value																						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
. . . .	. . . .	.																															
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
0x668	**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
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		ACKM	NAKM	STALLM	Res.	CHHM	XFRCM
	Reset value																						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
0x800	**OTG_ DCFG**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ERRATIM	XCVRDLY	Res.	PFIVL		DAD							Res.	NZLSOHSK	DSPD	
	Reset value																	0	0		0	0	0	0	0	0	0	0	0		0	0	0

表 231. OTG_FS 寄存器映射和复位值（续）

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
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
0x810	**OTG_ DIEPMSK**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NAKM	Res.	Res.	Res.	Res.	TXFURM	Res.	INEPNEM	INEPNMM	ITTXFEMSK	TOM	Res.	EPDM	XFRCM
	Reset value																			0					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.	NYETMSK	NAKMSK	BERRM	Res.	Res.	Res.	OUTPKTERRM	Res.	Res.	Res.	OTEPDM	STUPM	Res.	EPDM	XFRCM
	Reset value																		0	0	0				0				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
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
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
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
0x908	**OTG_ DIEPINT0**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NAK	Res.	PKTDRPSTS	Res.	Res.	TXFIFOUDRN	TXFE	INEPNE	INEPNM	ITTXFE	TOC	Res.	EPDISD	XFRC
	Reset value																			0		0			0	1	0	0	0	0		0	0

表 231. OTG_FS 寄存器映射和复位值（续）

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
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
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.	NAK	Res.	PKTDRPSTS	Res.	Res.	TXFIFOUDRN	TXFE	INEPNE	INEPNM	ITTXFE	TOC	Res.	EPDISD	XFRC
	Reset value																			0		0			0	1	0	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
. . . .	. . . .	.																															

表 231. OTG_FS 寄存器映射和复位值（续）

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
0x9A8	OTG_ DIEPINT5	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NAK	Res.	PKTDRPSTS	Res.	Res.	TXFIFOUDRN	TXFE	INEPNE	INEPNM	ITTXFE	TOC	Res.	EPDISD	XFRC
	Reset value																			0		0			0	1	0	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
. . . .	. . . .	.																															
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
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.	NYET	NAK	BERR	Res.	Res.	Res.	OUTPKTERR	Res.	Res.	STSPHSRX	OTEPDIS	STUP	Res.	EPDISD	XFRC
	Reset value																		0	0	0				0			0	0	0		0	0
0xB10	OTG_ DOEPTSIZ0	Res.	STUP CNT		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PKTCNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	XFRSIZ						
	Reset value		0	0										0													0	0	0	0	0	0	0
0xB20	OTG_ DOEPCTL1	EPENA	EPDIS	SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	Res.	Res.	Res.	Res.	STALL	SNPM	EP TYP		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.	NYET	NAK	BERR	Res.	Res.	Res.	OUTPKTERR	Res.	Res.	STSPHSRX	OTEPDIS	STUP	Res.	EPDISD	XFRC
	Reset value																		0	0	0				0			0	0	0		0	0

表 231. OTG_FS 寄存器映射和复位值（续）

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
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
. . . .	. . . .	.																															
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
0xBA8	**OTG_ DOEPINT5**	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NYET	NAK	BERR	Res.	Res.	Res.	OUTPKTERR	Res.	Res.	STSPHSRX	OTEPDIS	STUP	Res.	EPDISD	XFRC
	Reset value																		0	0	0				0	-		0	0	0		0	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
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 54 页的第 2.2.2 节*。

33.16 OTG_FS 编程模型

33.16.1 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 控制器进入主机模式；连接“B 型”插头后，OTG_FS 控制器进入设备模式。

本节介绍了 OTG_FS 控制器在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 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 控制器是在主机模式还是设备模式下工作。

33.16.2 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。

33.16.3 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. 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 1178 页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。

33.16.4 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）。

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 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 主机不会中断已在 USB 上启动的通信事务。

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. 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）。
3. 应用程序在传输正常完成之前将其中止。

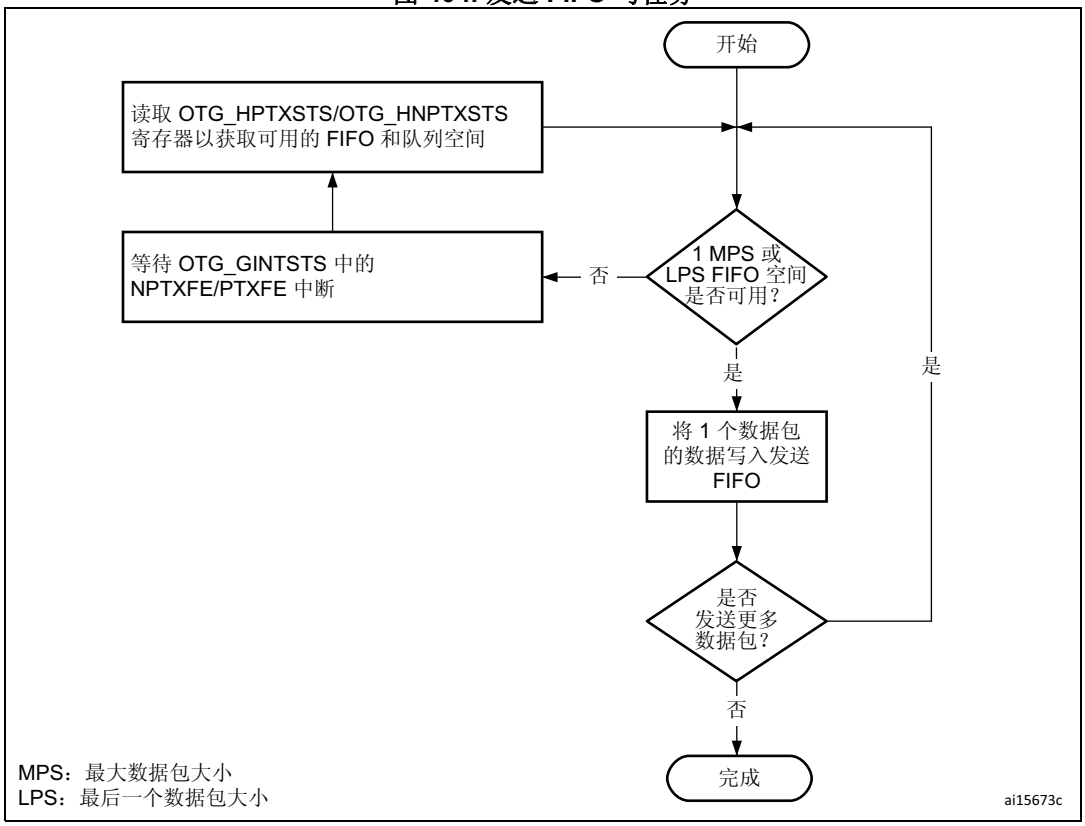
操作模型

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 32 位字写操作的同时，OTG_FS 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 32 位字形式向发送 FIFO 写入数据。如果数据包大小不是 32 位字的整数倍，则应用程序必须将数据包填充到 32 位字的整数倍。OTG_FS 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。

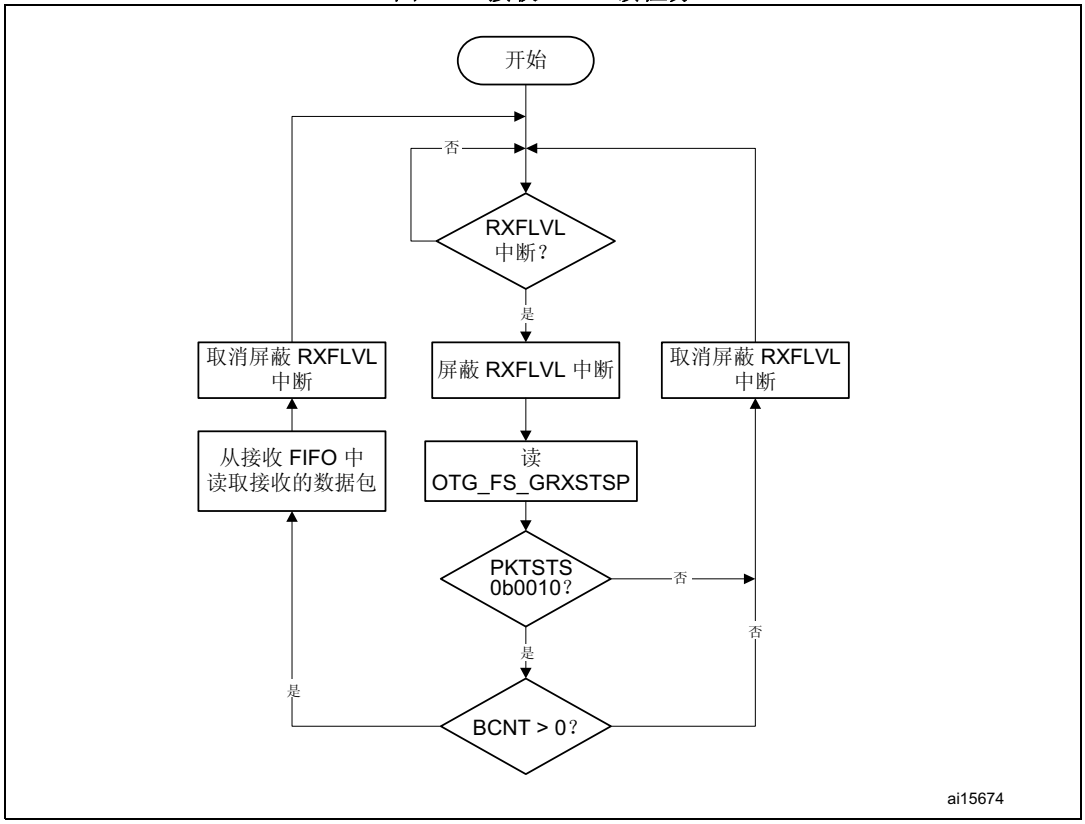
图 404. 发送 FIFO 写任务



- **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 405. 接收 FIFO 读任务



- **批量和控制传输类型的 OUT/SETUP 通信事务**

  图 406 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：

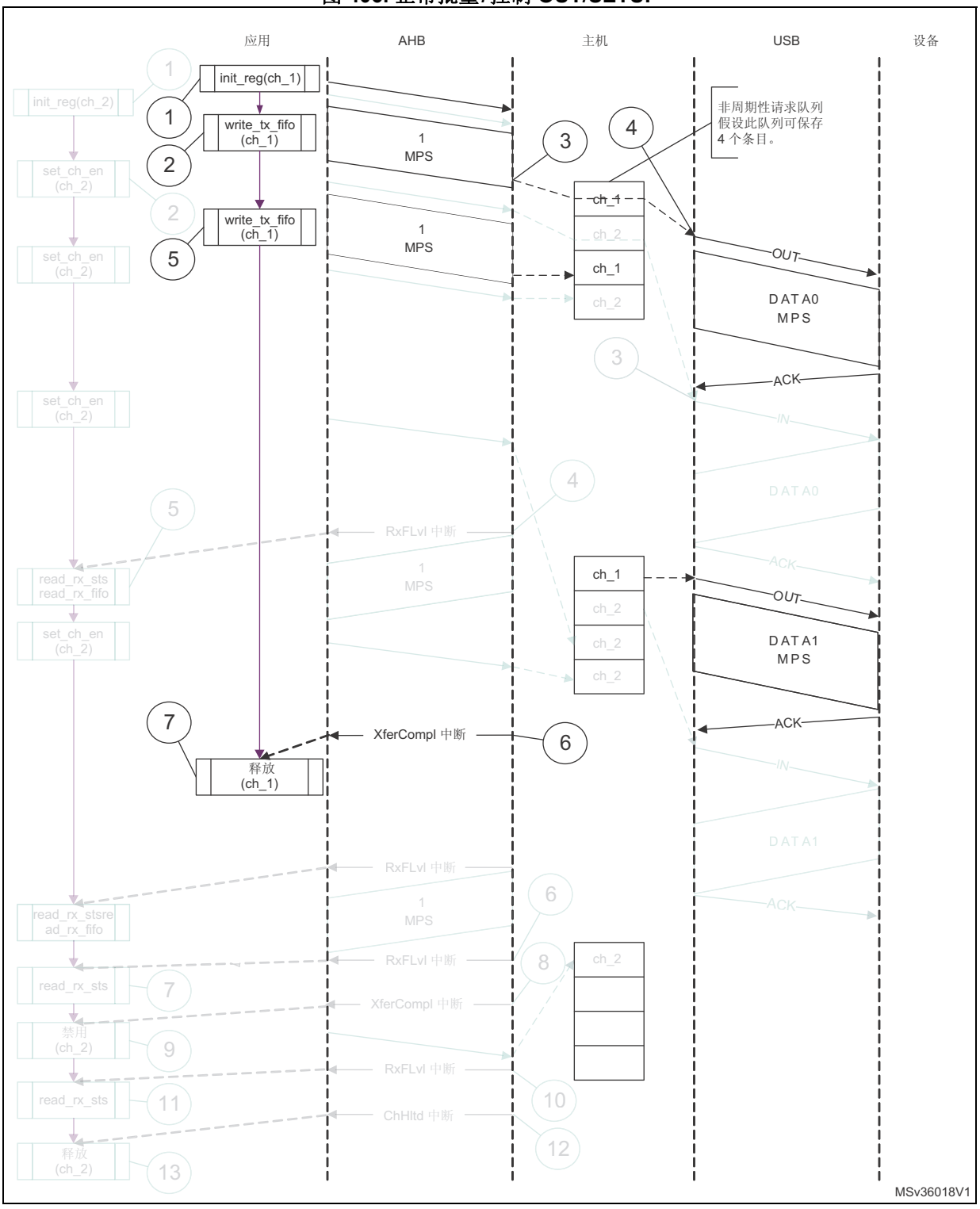
  – 应用程序尝试发送两个最大数据包大小的数据包（传输大小 = 1024 字节）。
  – 非周期性发送 FIFO 可存储两个数据包（FS 对应 128 字节）。
  – 非周期性请求队列深度 = 4。

- **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字写操作时，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 406. 正常批量/控制 OUT/SETUP



1. 灰显元素与此图的上下文不相关。

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
   }
```

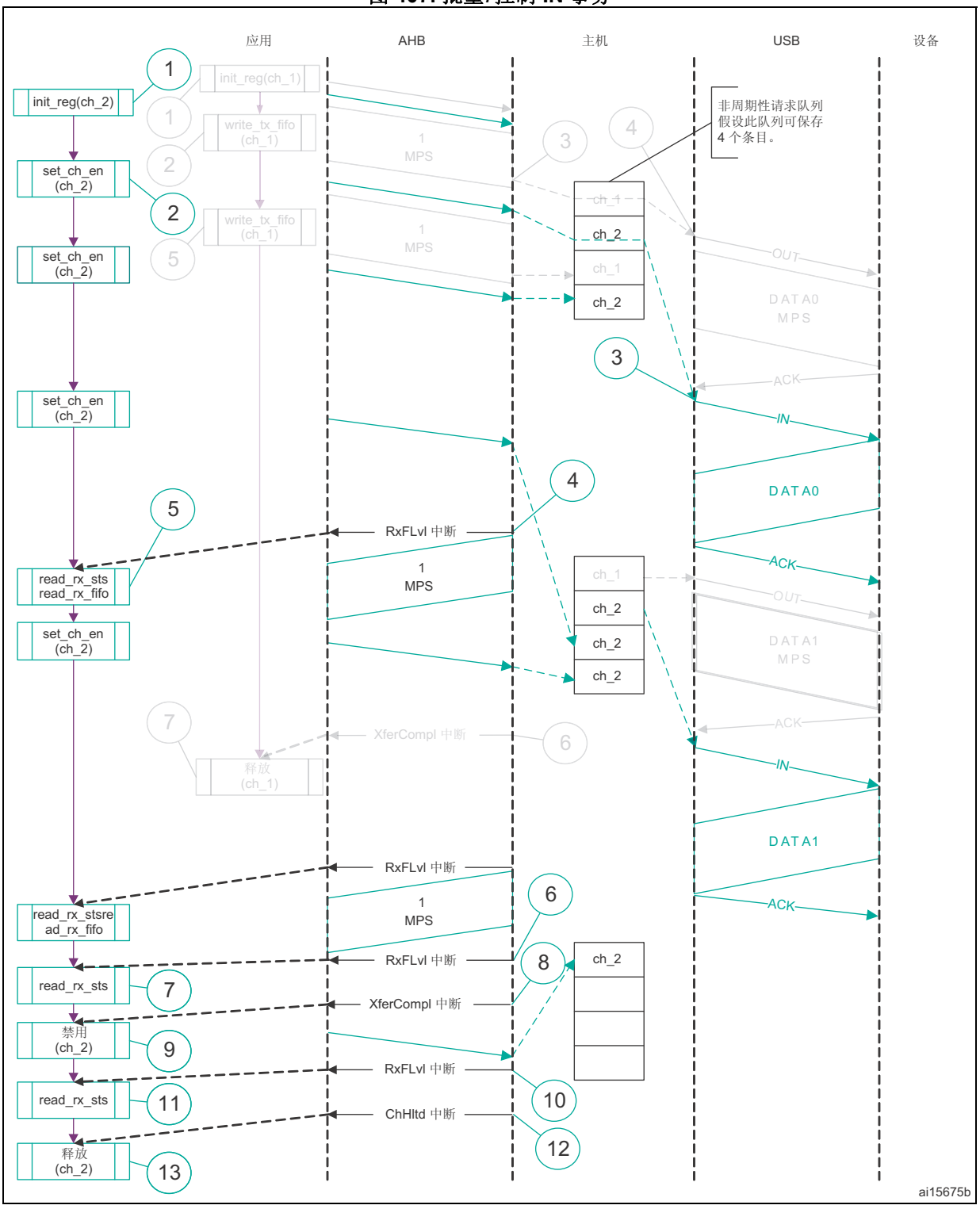
```
else if (DTERR)
   {
   Reset Error Count
   }
```

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 407 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 对应 72 字节）。
  - 非周期性请求队列深度 = 4。

图 407. 批量/控制 IN 事务



1. 灰显元素与此图的上下文不相关。

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 ¹ 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**

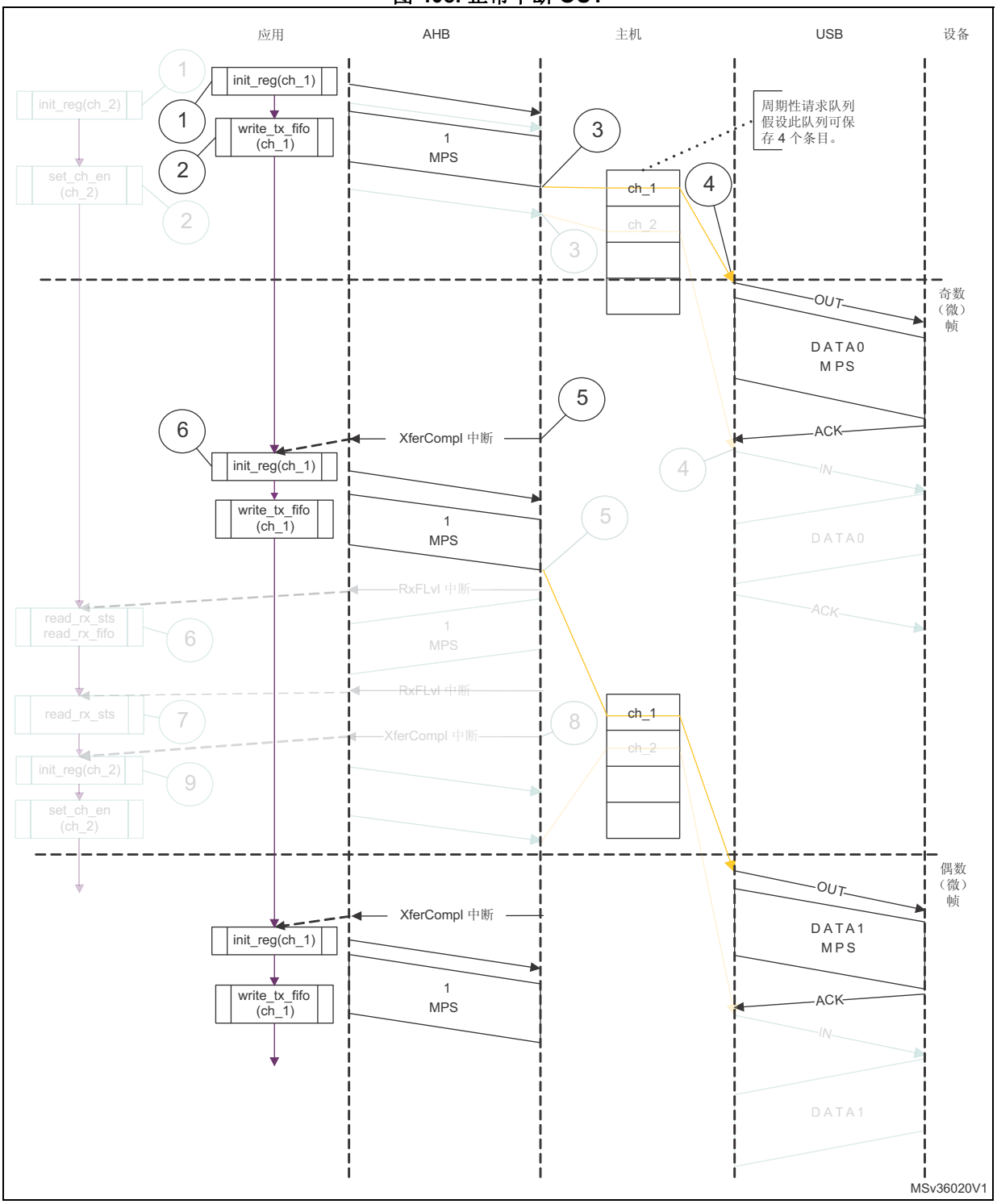
  图 408 显示了典型的中断 OUT 操作。假设：

  – 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 主机向周期性请求队列写入一个请求条目。
4. OTG_FS 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 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 408. 正常中断 OUT



1. 灰显元素与此图的上下文不相关。

- **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。

```
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）。

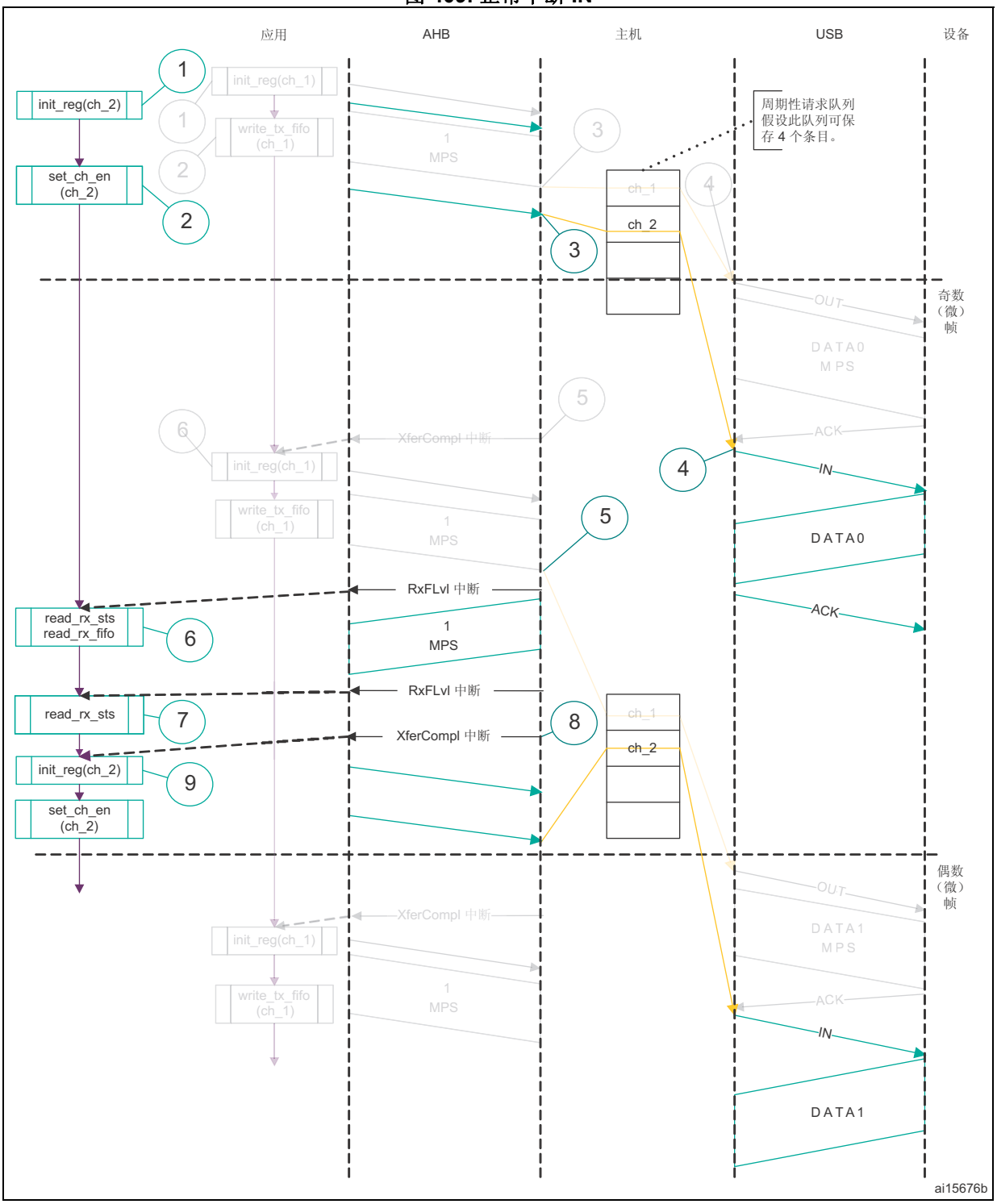
  – 周期性请求队列深度 = 4。

- **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 置位，对于每次 OTG_HCCHAR2 寄存器写操作，OTG_FS 主机都会将一个 IN 请求写入周期性请求队列。
4. OTG_FS 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 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 ¹ 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 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 409. 正常中断 IN



1. 灰显元素与此图的上下文不相关。

- **同步 OUT 事务**

  图 409 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）。

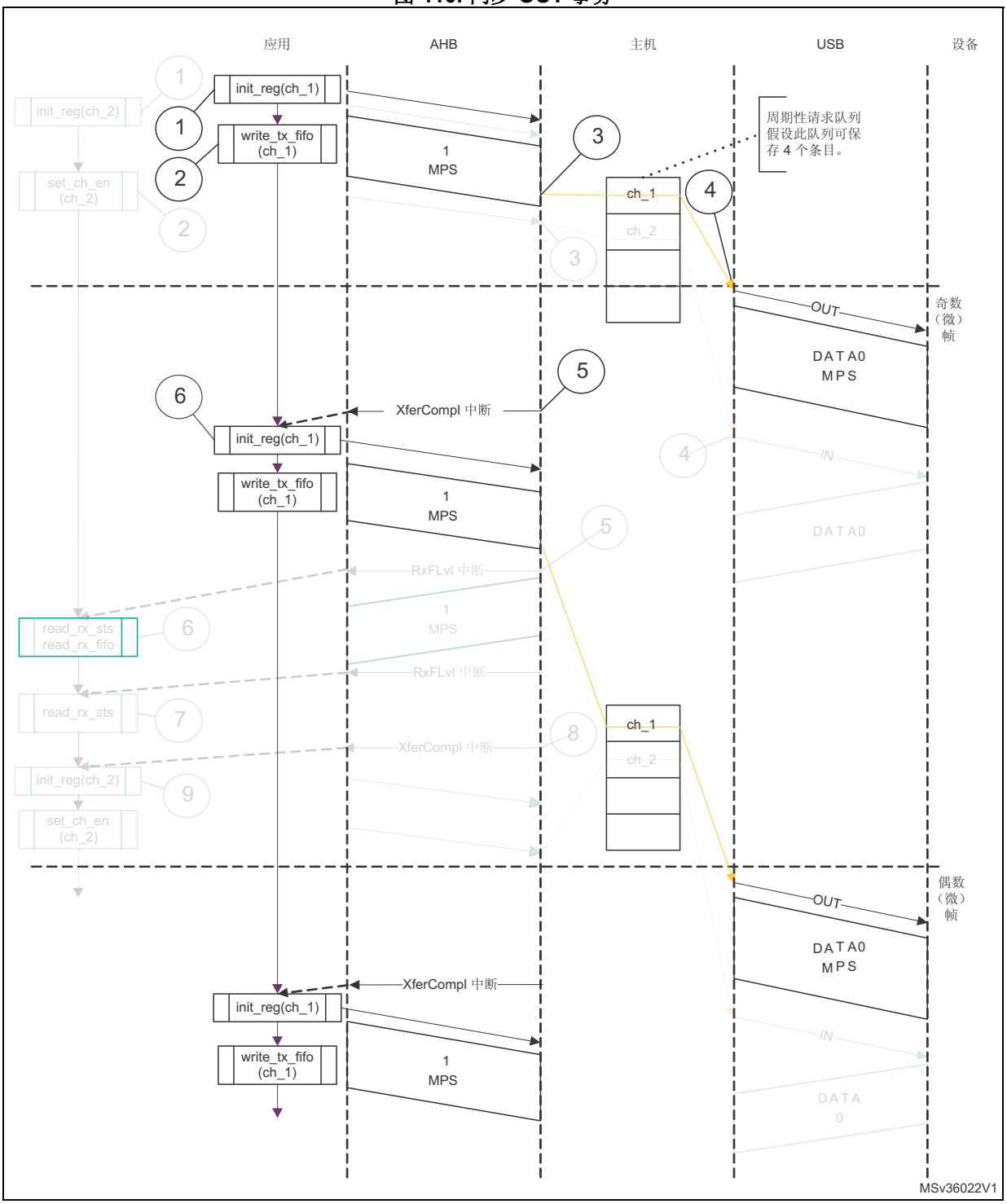
  – 周期性发送 FIFO 可存储一个数据包 (1 KB)。

  – 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 主机向周期性请求队列写入一个请求条目。
4. OTG_FS 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 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 410. 同步 OUT 事务



1. 灰显元素与此图的上下文不相关。

- **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）。

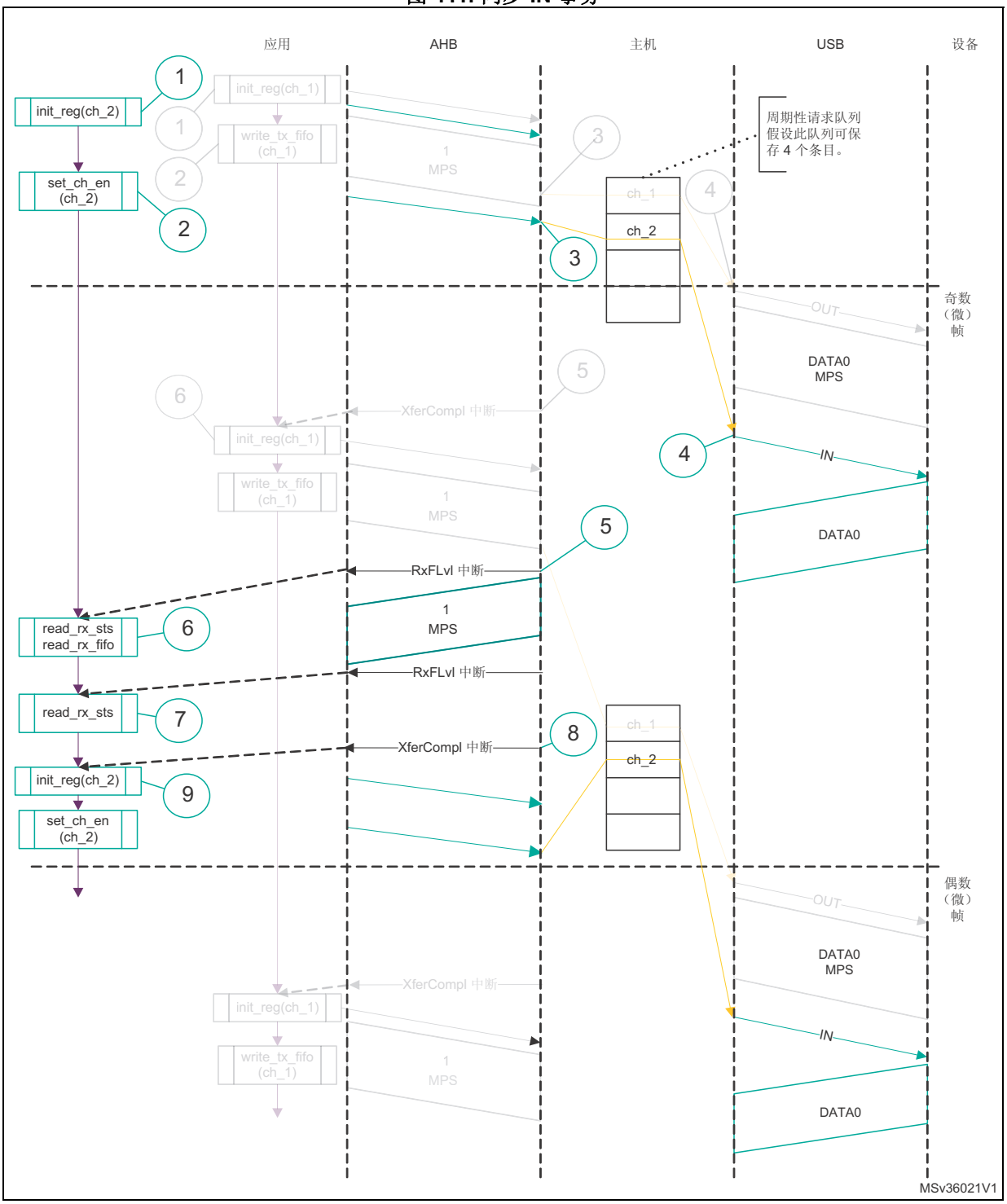
  – 接收 FIFO 可以保存至少一个最大数据包大小的数据包和每个数据包的两个状态字（1031 字节）。

  – 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 置位，对于每次 OTG_HCCHAR2 寄存器写操作，OTG_FS 主机都会将一个 IN 请求写入周期性请求队列。
4. OTG_FS 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 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 ¹ 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 411. 同步 IN 事务



1. 灰显元素与此图的上下文不相关。

- **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 控制器处理两种情况的 babble：数据包 babble 和端口 babble。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 控制器检测到数据包 babble 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 控制器检测到端口 babble 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。

33.16.5 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）

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。

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。

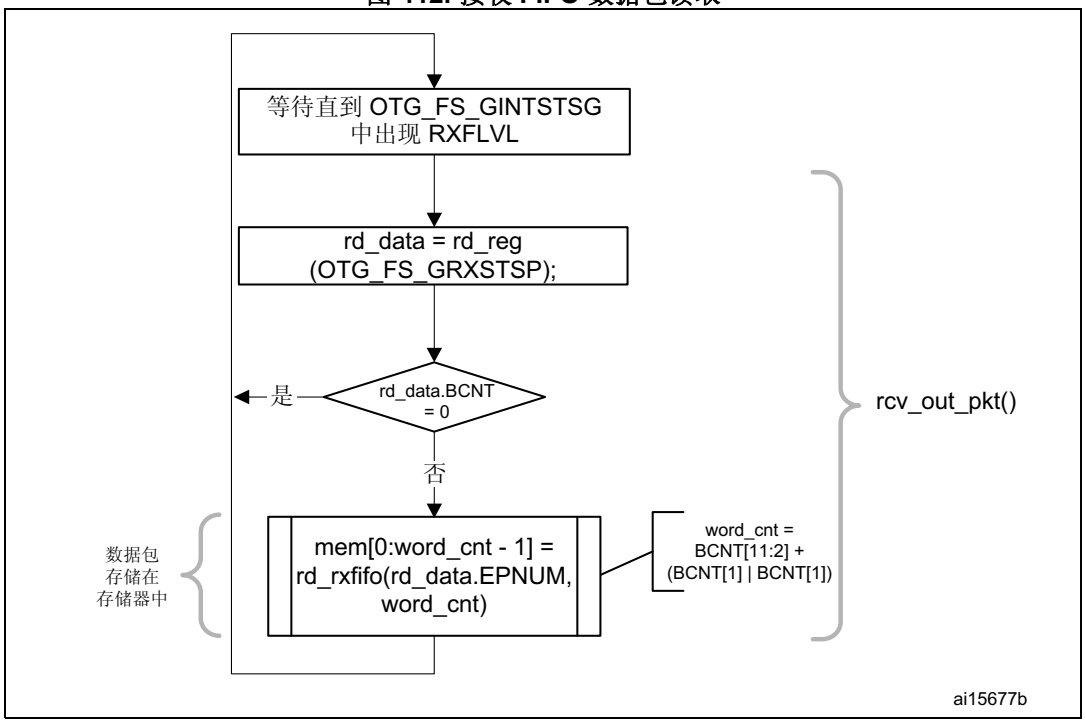
- **数据包读取**

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 中弹出数据的字节计数并存储在存储器中。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，模块将在指定控制 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 412 提供了上述过程的流程图。

图 412. 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 数据包的正确数量。
   - 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 个字以在所有控制端点间存储两个额外的 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（SETUP 数据包模式）。模块会在接收数据 FIFO 中保留此空间，仅写入 SETUP 数据，绝对不会将此空间用于数据包。
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。

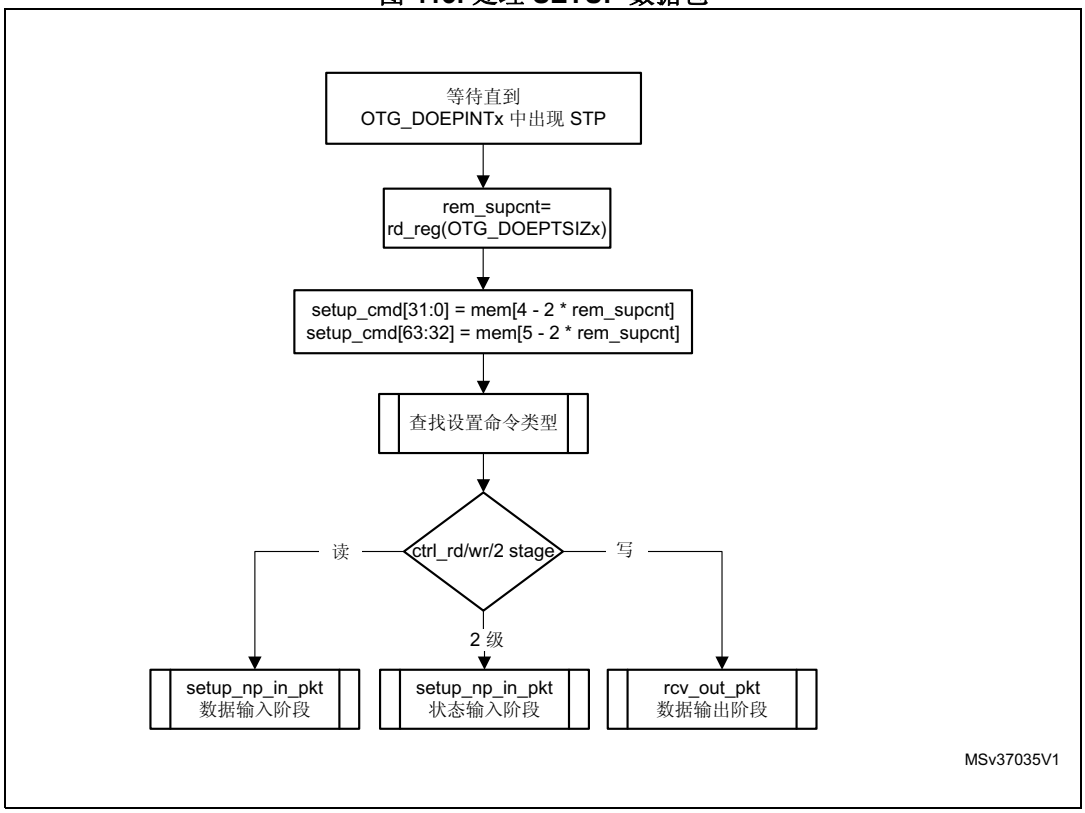
- **内部数据流**

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内部使用的控制信息
   – 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，表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 413. 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 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

   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，则必须在应用程序建立该控制端点上的状态阶段传输前，将 STALL 位置 1 或清零。

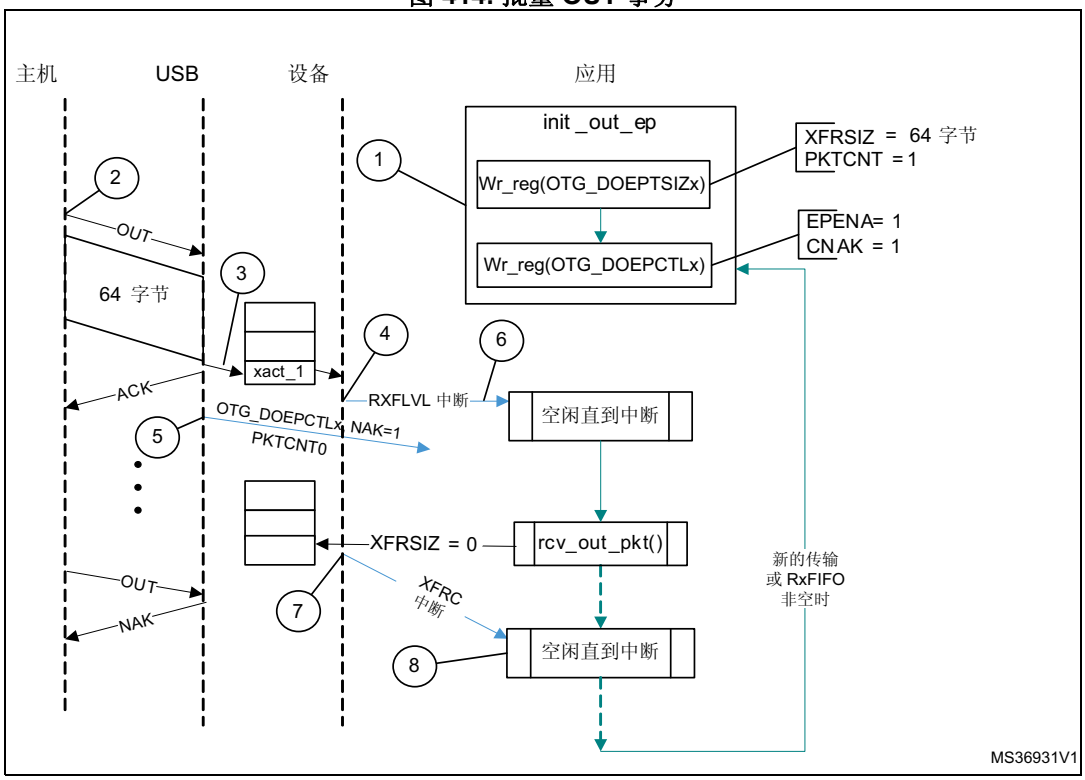
示例

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 414 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 414. 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的数据包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。
   – 已经通过 USB 发送的数据量 = (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) × 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 1190 页的通用非周期性 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，则模块将为该端点生成一个“TxFIFO 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。

33.16.6 最坏情况下的响应时间

当 OTG_FS 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）。

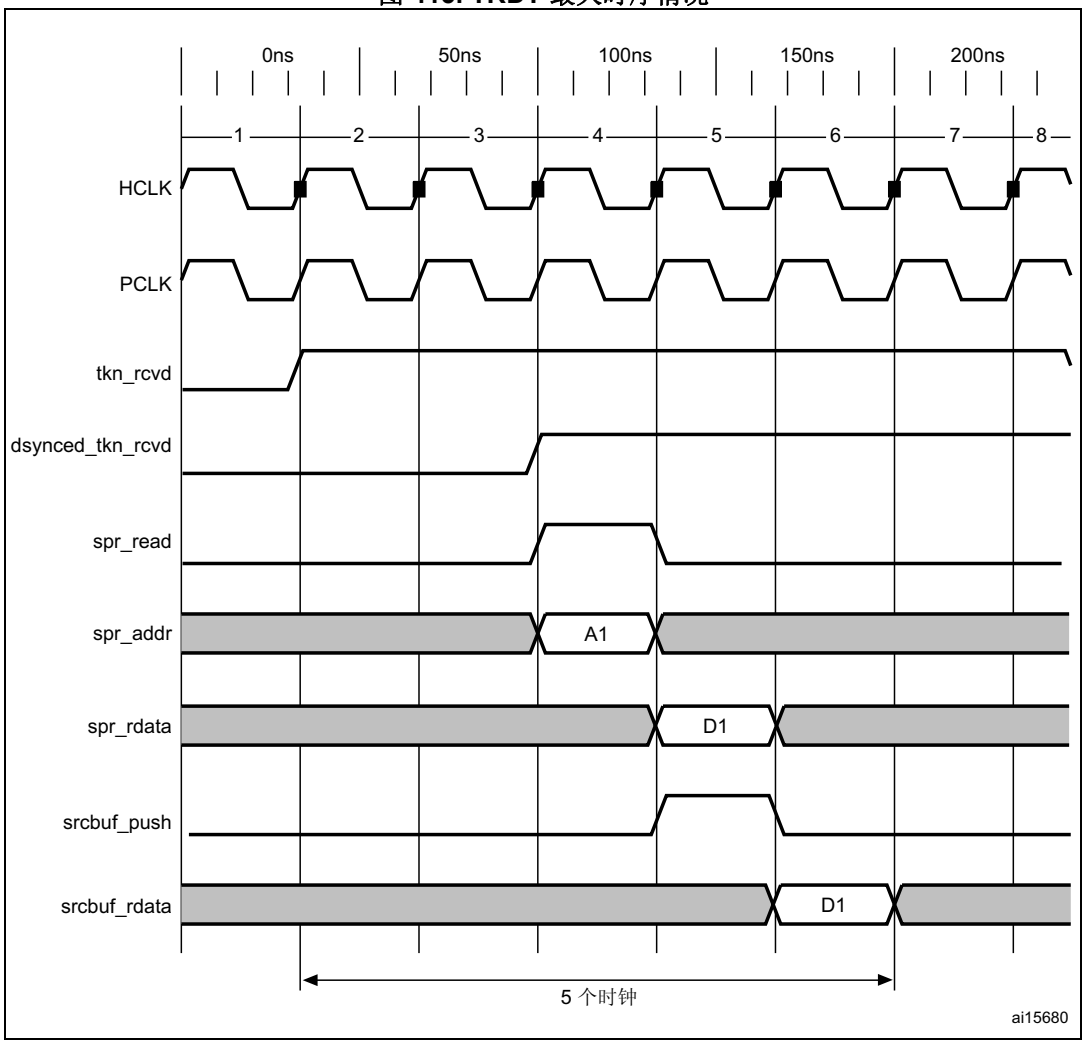
如果 AHB 的运行频率高于 PHY，应用程序可以使用较小的 TRDT 值（在 OTG_GUSBCFG 中）。

图 415 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

要计算 TRDT 的值，请参见*表 229：TRDT 值 (FS)* 。

图 415. TRDT 最大时序情况



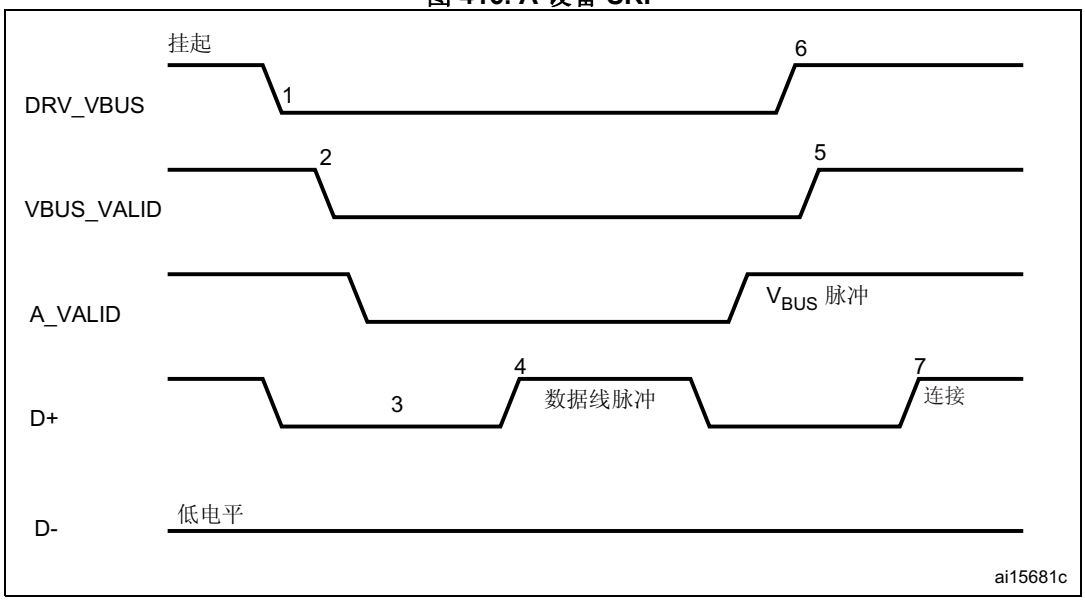
33.16.7 OTG 编程模型

OTG_FS 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 设备。该模块接口与“B 型”插头连接时，该模块称为 B 设备。在主机模式下，OTG_FS 控制器将关闭 V_{BUS} 以节省电能。B 设备可以借助 SRP 请求 A 设备打开 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 设备可以借助 HNP 协商并切换到主机角色。在协商模式下执行 HNP 后，B 设备会挂起总线并恢复到设备角色。

A 设备会话请求协议

应用程序必须将模块 USB 配置寄存器中的 SRP 功能位置 1。这将使能 OTG_FS 控制器在 A 设备模式下检测到 SRP。

图 416. A 设备 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 设备 V_{BUS} 电平信号
   D+ = 正向数据线
   D- = 负向数据线

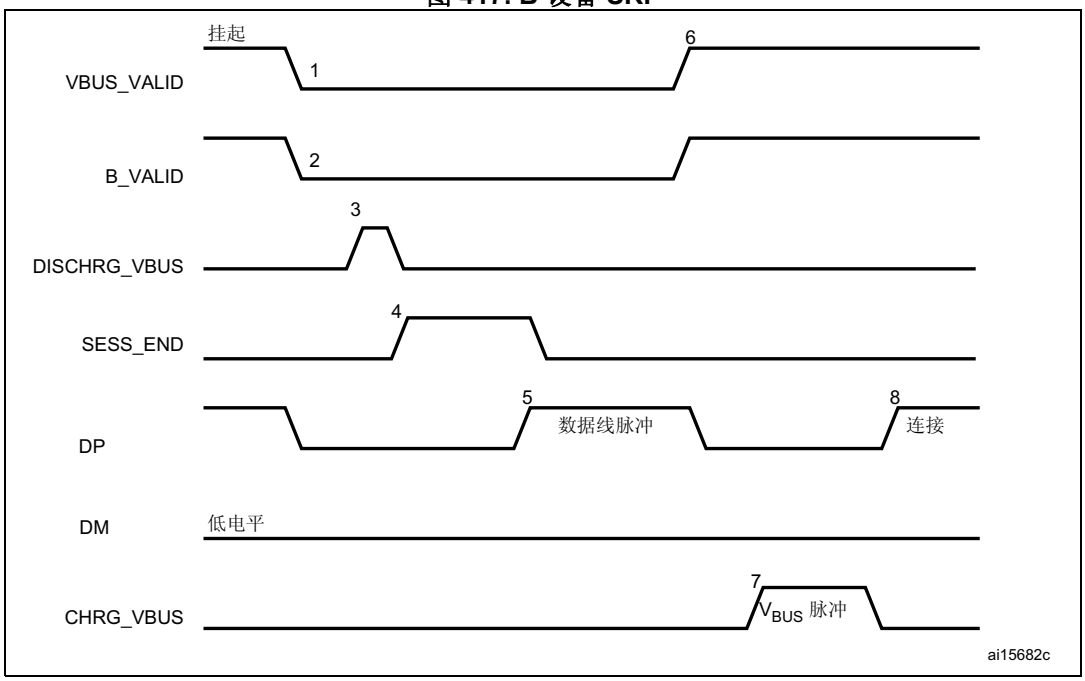
以下几点引用并描述了*图 416* 所示的信号过程：

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 控制器会检测到数据线脉冲。
5. 设备会驱动 V_{BUS} 到 A 设备会话有效电平（最小 2.0 V）以上，以产生 V_{BUS} 脉冲。OTG_FS 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 设备会话请求协议

应用程序必须将模块 USB 配置寄存器中的 SRP 功能位置 1。这将使能 OTG_FS 控制器在 B 设备模式下启动 SRP。OTG_FS 控制器可以借助 SRP 向主机请求新会话。

图 417. B 设备 SRP



1. VBUS_VALID = 来自 PHY 的 V_{BUS} 有效信号
   B_VALID = 发送到 PHY 的 B 设备有效会话
   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

以下几点引用并描述了*图 417* 所示的信号过程：

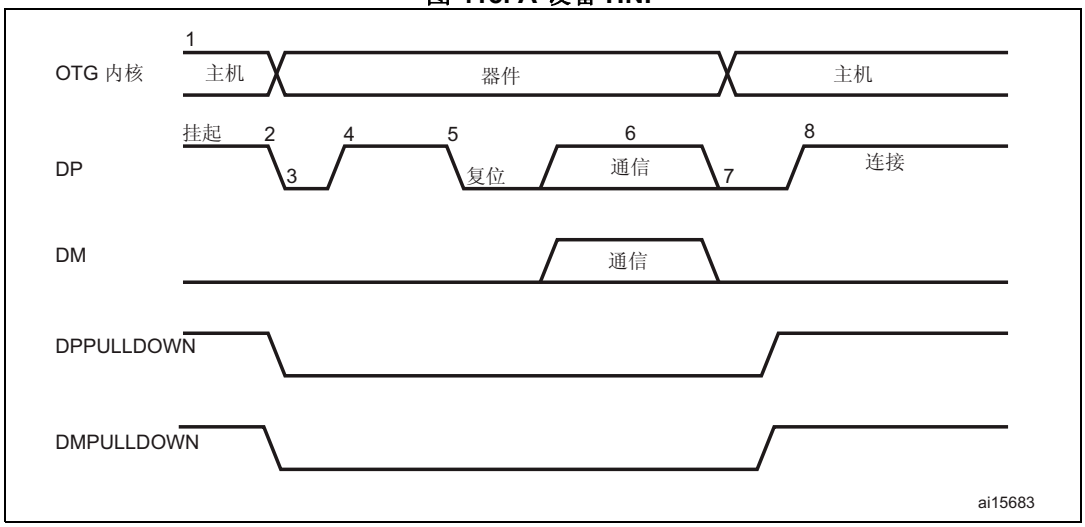
1. 为节省电能，在总线空闲时主机会挂起并关闭端口电源。
   OTG_FS 控制器会在总线空闲 3 ms 后，将模块中断寄存器中的早期挂起位置 1。随后，OTG_FS 控制器会将模块中断寄存器中的 USB 挂起位置 1。
   OTG_FS 控制器会通知 PHY 对 V_{BUS} 进行放电。
2. PHY 指示会话结束。这是 SRP 的初始条件。启动 SRP 之前，OTG_FS 控制器需要 2 ms 的 SE0 信号。
   对于 USB 1.1 全速串行收发器，应用程序必须在 BSVLD 位（位于 OTG_GOTGCTL）被禁止后，等待 V_{BUS} 放电至 0.2 V。此放电时间可从收发器供应商处获取，该值可能因收发器的不同而异。
3. OTG_FS 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 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 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 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 控制器将进行连接，并完成 SRP 过程。

A 设备主机协商协议

通过 HNP 可以将 USB 主机角色从 A 设备切换到 B 设备。应用程序必须将模块 USB 配置寄存器中的 HNP 功能位置 1，以使能 OTG_FS 控制器在 A 设备模式下的 HNP 功能。

图 418. A 设备 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。

以下几点引用并描述了*图 418* 所示的信号过程：

1. OTG_FS 控制器向 B 设备发送 SetFeature b_hnp_enable 描述符以使能 HNP 支持。B 设备的 ACK 响应表示 B 设备支持 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 控制器指示 B 设备支持 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 设备检测到 USB 挂起时，将断开连接，指示 HNP 的初始条件。B 设备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 控制器将 OTG 中断状态寄存器中检测到的主机协商中断位置 1，指示 HNP 的启动。

   OTG_FS 控制器将禁止 PHY 中 DM 线和 DM 线的下拉，以指示设备角色。PHY 使能 OTG_DP 上拉电阻，以告知 B 设备 A 设备的连接。

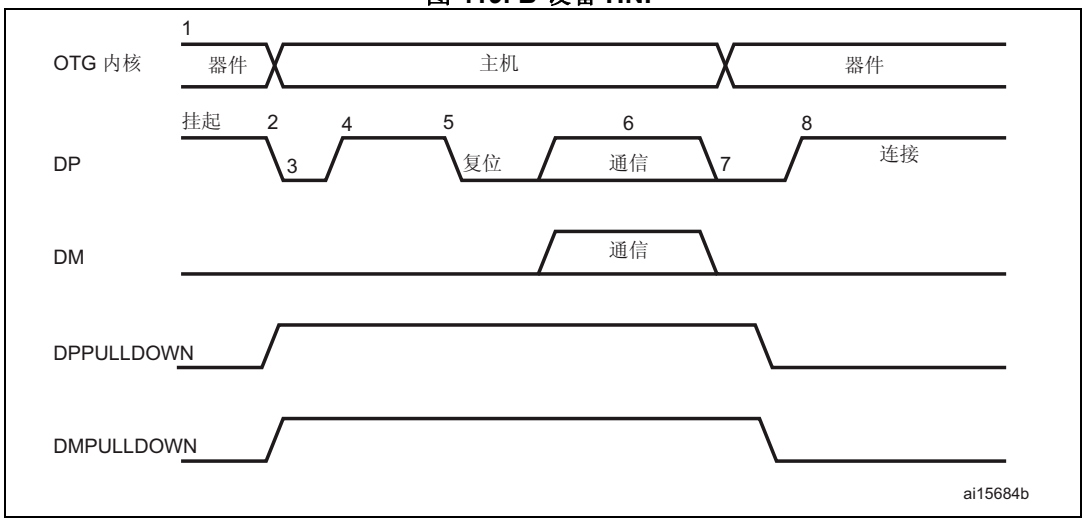
   应用程序必须读取 OTG 控制和状态寄存器中的当前模式位，以确定设备工作模式。
4. B 设备检测到连接，发出 USB 复位信号，并枚举 OTG_FS 控制器进行数据通信。
5. B 设备继续担当主机角色，发起数据通信，并在结束时挂起总线。

   OTG_FS 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 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 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 控制器将使能 PHY 中 DM 和 DM 下拉电阻，以指示其承担主机角色。
7. OTG_FS 控制器将 OTG 中断状态寄存器中的连接器 ID 状态变化中断位置 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 控制器在 A 设备模式下工作。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 设备，完成 HNP 过程。

B 设备主机协商协议

通过 HNP 可以将 USB 主机角色从 B 设备切换到 A 设备。应用程序必须将模块 USB 配置寄存器中的 HNP 功能位置 1，以使能 OTG_FS 控制器在 B 设备模式下的 HNP 功能。

图 419. B 设备 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。

以下几点引用并描述了*图 419* 所示的信号过程：

1. A 设备发出一个 SetFeature b_hnp_enable 描述符以使能 HNP 支持。OTG_FS 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 控制器指示启动 HNP。

2. A 设备不再使用总线时，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 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 控制器会将模块中断寄存器中的 USB 挂起位置 1。

   OTG_FS 控制器会断开连接，A 设备在总线上检测到 SE0，指示 HNP 即将开始。OTG_FS 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 设备会通过激活 OTG_DP 上拉电阻进行响应。OTG_FS 控制器检测到此信号时认为有设备接入。

   OTG_FS 控制器将 OTG 中断状态寄存器中的主机协商成功状态变化中断位置 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_FS_HPRT 中的 PRST 位）置 1，同时 OTG_FS 控制器发出 USB 复位信号，并枚举 A 设备进行数据通信。
4. OTG_FS 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 设备检测到挂起时，会断开连接，然后切换回主机角色。OTG_FS 控制器将禁止 PHY 中 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 控制器进行连接，完成 HNP 过程。

34 调试支持 (DBG)

34.1 概述

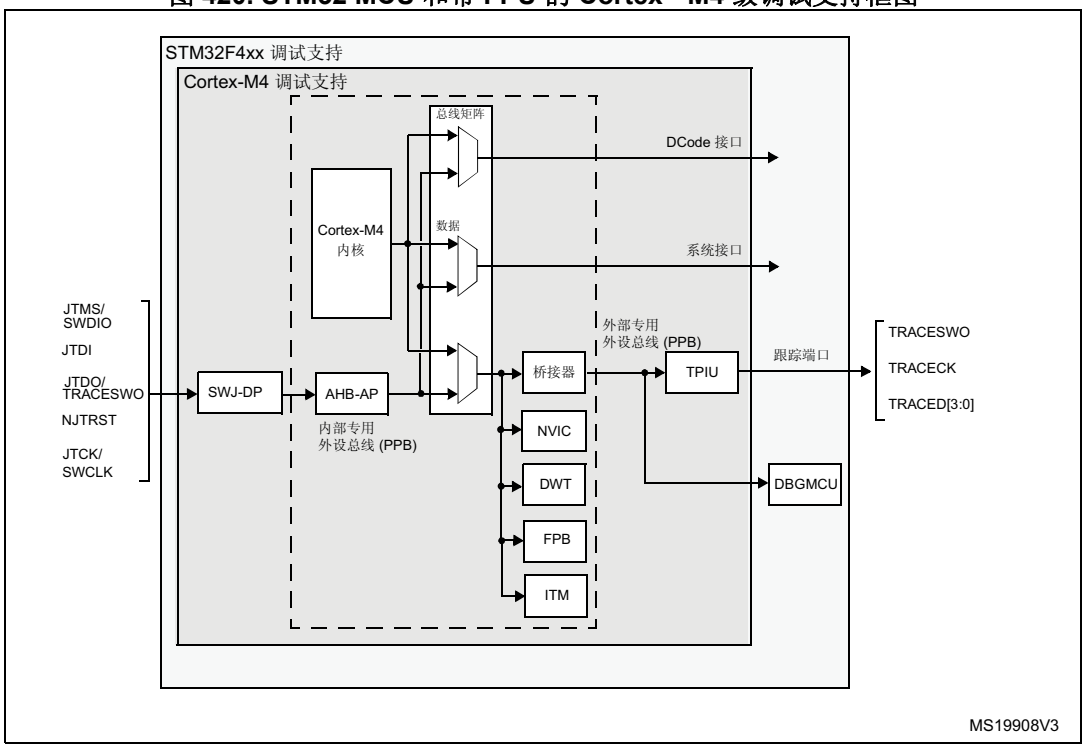
STM32F413/423 的内核是带 FPU 的 Cortex®-M4，该内核包含用于高级调试功能的硬件扩展。调试扩展允许内核可以在取指（指令断点）或取访问数据（数据断点）时停止内核。内核停止时，可以查询内核的内部状态和系统的外部状态。查询完成后，将恢复内核和系统并恢复程序执行。

当调试主机与 STM32F413/423 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 420. STM32 MCU 和带 FPU 的 Cortex®-M4 级调试支持框图



注： *带 FPU 的 Cortex®-M4 内核中内置的调试功能是 Arm® CoreSight 设计套件的一部分。*

Arm® 带 FPU 的 Cortex®-M4 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 补丁断点
- DWT：数据观察点触发
- TPUI：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 STM32F413/423 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 Arm® 带 FPU 的 Cortex®-M4 内核支持的调试功能的更多信息，请参见《带 FPU 的 Cortex®-M4-r0p1 技术参考手册》和《CoreSight 设计套件 r0p1 技术参考手册》（请参见第 34.2 节：Arm® 参考文档）。*

34.2 Arm® 参考文档

- 带 FPU 的 Cortex®-M4 r0p1 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
- Arm® CoreSight 设计套件版本 r0p1 技术参考手册

34.3 SWJ 调试端口（串行接口和 JTAG）

STM32F413/423 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 421. SWJ 调试端口

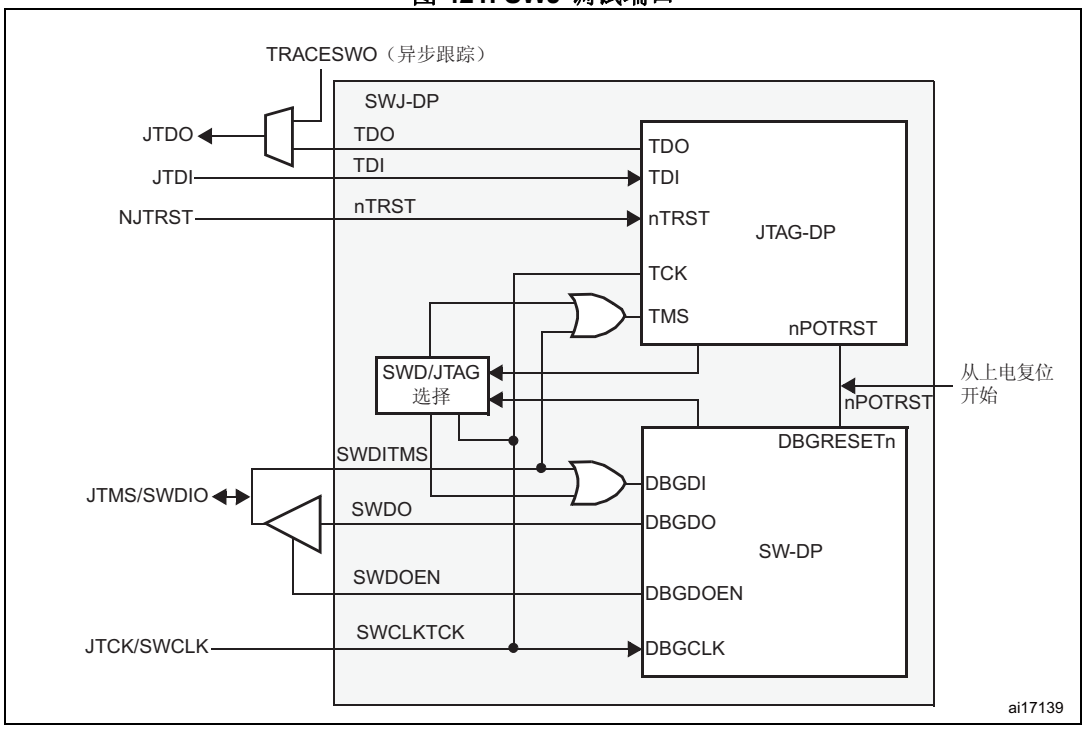


图 421 显示了异步 TRACE 输出 (TRACESWO) 与 TDO 复用。这意味着异步跟踪只能在 SW-DP 上实现，不能在 JTAG-DP 上实现。

34.3.1 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 优先传输）
3. 输出超过 50 个 TCK 周期的 TMS (SWDIO) = 1 信号

34.4 引脚排列和调试端口引脚

STM32F413/423 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可能随封装而不同。

34.4.1 SWJ 调试端口引脚

STM32F413/423 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 232. 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

34.4.2 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，可供调试主机立即使用（请注意，除非由调试主机明确编程，否则不分配跟踪输出）。

但是，STM32F413/423 MCU 可以禁止部分或全部 SWJ-DP 端口，进而释放相关引脚以用作通用 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7.3.2 节：I/O 引脚复用器和映射*。

表 233. 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（*nTRST*、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，施密特触发器激活等的控制）。

34.4.3 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 输入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 TCK，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。

34.4.4 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他 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），在 IOPORT 控制器中修改相应的普通 I/O 口配置是无效的。

34.5 JTAG TAP 连接

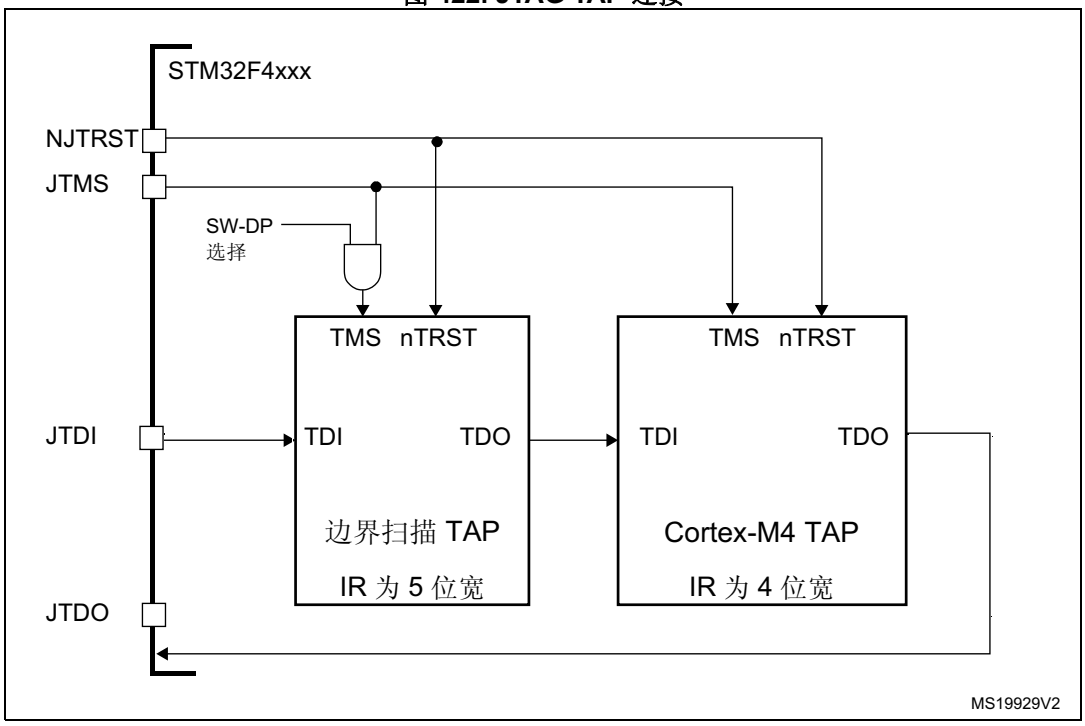
MCU 集成了两个串连的 JTAG TAP、边界扫描 TAP（IR 宽度为 5 位）和带 FPU 的 Cortex®-M4 TAP （IR 宽度为 4 位）。

要访问带 FPU 的 Cortex®-M4 的 TAP 以进行调试：

1. 首先，必须将 BYPASS 指令移位输入边界扫描 TAP。
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 TAP，必须输入 BYPASS 指令。
3. 移位输入数据时，不用的 TAP 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 TAP（JTMS 强制为高电平）。*

图 422. JTAG TAP 连接



34.6 ID 代码和锁定机制

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。

34.6.1 MCU 器件 ID 代码

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 1219 页的第 34.16 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)
此位域指示器件的版本：
0x1000 = 版本 A

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 0x463。

34.6.2 边界扫描 TAP

JTAG ID 代码

BSC（边界扫描）的 TAP 集成了 JTAG ID 代码（等于 0x0645 8041）。

34.6.3 带 FPU 的 Cortex®-M4 TAP

Arm® 带 FPU 的 Cortex®-M4 的 TAP 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 0x4BA00477（对应于带 FPU 的 Cortex®-M4 r0p1，请参见*第 34.2 节：Arm® 参考文档*）。

34.6.4 带 FPU 的 Cortex®-M4 JEDEC-106 ID 代码

Arm® 带 FPU 的 Cortex®-M4 集成了 JEDEC-106 ID 代码。它位于映射到内部 PPB 总线地址为 0xE00F FFD0_0xE00F FFE0 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。

34.7 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《r0p1 带 FPU 的 Cortex®-M4 技*术参考手册 (TRM)*》。*有关参考信息，请参见第 34.2 节：Arm® 参考文档*）。

表 234. 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 0x4BA0 0477（Arm® 带 FPU 的 Cortex®-M4 r0p1 ID 代码）
1010	DPACC [35 位]	调试端口访问寄存器 初始化调试端口，并允许访问调试端口寄存器。 – 传输 IN 数据时： 位 34:3 = DATA[31:0] = 为写请求传输的 32 位数据 位 2:1 = A[3:2] = 调试端口寄存器的 2 位地址。 位 0 = RnW = 读请求 (1) 或写请求 (0)。 – 传输 OUT 数据时： 位 34:3 = DATA[31:0] = 读请求后读取的 32 位数据 位 2:0 = ACK[2:0] = 3 位确认： 010 = OK/FAULT 001 = WAIT 其他 = 保留 有关 A(3:2) 位的说明，请参见*表 235*

表 234. 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 其他 = 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 235. 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）

34.8 SW 调试端口

34.8.1 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。

34.8.2 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 236. 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 寄存器的地址位域（请参见 *表 235*）
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®-M4 r0p1 *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 237. 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 238. 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。

34.8.3 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 0x2BA01477（相当于带 FPU 的 Cortex®-M4 r0p1）。

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®-M4 r0p1 TRM》和《CoreSight 设计套件 r0p1 TRM》。

34.8.4 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），这样即使在其他操作仍未完成时，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。

34.8.5 SW-DP 寄存器

当 APnDP=0 时能够访问这些寄存器

表 239. SW-DP 寄存器

A[3:2]	R/W	SELECT 寄存器的 CTRLSEL 位	寄存器	注释
00	读取	-	IDCODE	制造商代码未设置为 ST 代码。0x2BA01477（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	读取		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

34.8.6 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

34.9 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®-M4 的 AHB-AP 包括 9 个 32 位寄存器：

表 240. 带 FPU 的 Cortex®-M4AHB-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®-M4 r0p1 TRM》。

34.10 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 241. 内核调试寄存器

寄存器	说明
DHCSR	32 位调试停止控制和状态寄存器 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处理器寄存器。
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®-M4 r0p1 TRM》。

为了在复位后立即使内核进入调试停止状态，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)。

34.11 调试主机在系统复位状态下建立连接的功能

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®-M4 将调试部分的复位（通常为 PORRESETn）和其他部分 (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）。*

34.12 FPB（Flash 补丁断点）

FPB 单元：

- 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。

FPB 包括：

- 2 个内容比较器，用来比较代码区域取得的内容并重映射到系统区域的相关地址。
- 6 个指令比较器，用来比较代码区域的指令。这些比较器可用来实现软件补丁或者硬件断点功能。

34.13 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

34.14 ITM（指令跟踪宏单元）

34.14.1 概述

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®-M4 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。

34.14.2 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 242. 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 以解锁对其他 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 34.17.2 节：TRACE 引脚分配*和*第 34.16.3 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）

34.15 ETM（嵌入式跟踪宏单元）

34.15.1 概述

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 34.13 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 格式化器会添加一些额外的数据包（请参见*第 34.17 节：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。

34.15.2 信号协议和数据包类型

Arm® IHI 0014N 文档的第 7 章“ETMv3 信号协议”中介绍了这部分内容。

34.15.3 主要的 ETM 寄存器

有关寄存器的更多信息，请参见 Arm® IHI 0014N 规范的第 3 章。

表 243. 主要的 ETM 寄存器

地址	寄存器	详细信息
0xE0041FB0	ETM 锁定访问	写入 0xC5ACCE55 以解锁对其他 ETM 寄存器的写访问。
0xE0041000	ETM 控制	此寄存器控制 ETM 的一般操作，例如如何使能跟踪。
0xE0041010	ETM 状态	此寄存器提供有关跟踪和触发逻辑的当前状态的信息。
0xE0041008	ETM 触发事件	此寄存器定义控制触发的事件。
0xE004101C	ETM 跟踪使能控制	此寄存器定义选择哪个比较器。
0xE0041020	ETM 跟踪使能事件	此寄存器定义跟踪使能事件。
0xE0041024	ETM 跟踪启动/停止	此寄存器定义触发源启动和停止跟踪时分别使用的跟踪。

34.15.4 配置示例

从 TPIU 输出一个简单的数值：

1. 配置 TPIU 并使能 I/IO_TRACEN，以分配调试配置寄存器中的 TRACE I/O
2. 将 0xC5ACCE55 写入 ETM 锁定访问寄存器，以解锁对 ITM 寄存器的写保护
3. 将 0x00001D1E 写入控制寄存器（配置跟踪）
4. 将 0000406F 写入触发事件寄存器（定义触发事件）
5. 将 0000006F 写入跟踪使能事件寄存器（定义要启动/停止的事件）
6. 将 00000001 写入跟踪启动/停止寄存器（使能跟踪）
7. 将 0000191E 写入 ETM 控制寄存器（配置结束）

34.16 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、看门狗和 I2C 的时钟控制
- 跟踪引脚分配的控制

34.16.1 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。

34.16.2 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。

34.16.3 MCU 调试配置寄存器

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
- 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。

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_IOEN	Res.	Res.	DBG_STANDBY	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_IOEN**：跟踪引脚分配控制 (Trace pin assignment control)

– TRACE_IOEN=0 时：
TRACE_MODE=xx：未分配 TRACE 引脚（默认状态）

– TRACE_IOEN=1 时：

– TRACE_MODE=00：异步模式下的 TRACE 引脚分配

– TRACE_MODE=01：同步模式下的 TRACE 引脚分配，TRACEDATA 大小为 1

– TRACE_MODE=10：同步模式下的 TRACE 引脚分配，TRACEDATA 大小为 2

– TRACE_MODE=11：同步模式下的 TRACE 引脚分配，TRACEDATA 大小为 4

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取复位向量（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取复位向量

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）。

34.16.4 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 APB1 外设。它映射到外部 PPB 总线的地址 0xE004 2008。

寄存器通过 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_I2CFMP_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	Res.	DBG_TIM14_STOP	DBG_TI13_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 CAN1 (Debug CAN1 stopped when core is halted)

0：行为方式与正常模式下相同
1：冻结 CAN1 接收寄存器

位 24 **DBG_I2CFMP_SMBUS_TIMEOUT**：内核停止时停止 FMPI2C SMBUS 超时模式 (FMPI2C 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 I2C2 SMBUS 超时模式 (I2C2 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 I2C1 SMBUS 超时模式 (I2C1 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 保留，必须保持复位值。

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

34.16.5 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 1:0 **DBG_TIMx_STOP**：内核停止时 TIMx 计数器停止 (TIMx counter stopped when core is halted) (x=1..8)

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

34.17 TPIU（跟踪端口接口单元）

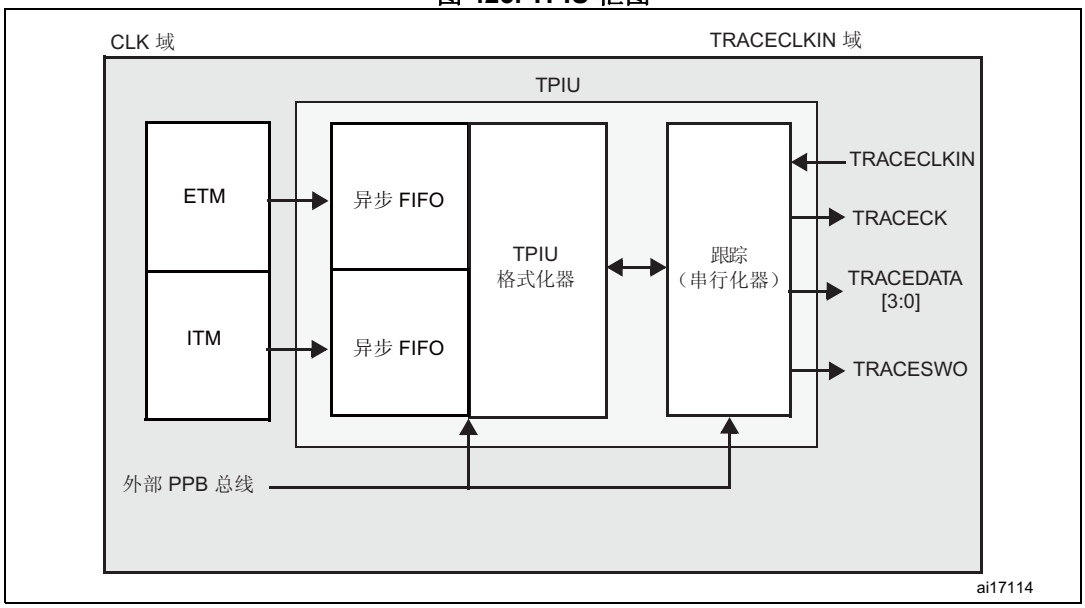
34.17.1 简介

TPIU 用作 ITM 和 ETM 中的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 423. TPIU 框图



34.17.2 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 244. 异步 TRACE 引脚分配

TPUI 引脚名称	跟踪同步模式		引脚分配
	类型	说明	
TRACESWO	O	TRACE 异步数据输出	PB3

- 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 245. 同步 TRACE 引脚分配

TPUI 引脚名称	跟踪同步模式		引脚分配
	类型	说明	
TRACECK	O	TRACE 时钟	PE2
TRACED[3:0]	O	TRACE 同步数据输出 可以是 1、2 或 4	PE[6:3]、PF[7:6]、PD3 和 PG[14:13]

TPUI 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_IOEN 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_IOEN 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 246. 灵活的 TRACE 引脚分配

DBGMCU_CR 寄存器		引脚用途	分配的 TRACE IO 引脚(1)					
TRACE_IOEN	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 (2)	-				
1	00	异步跟踪	TRACESWO	-	-	已释放（可用作 GPIO）		
1	01	同步跟踪 1 位	已释放 (2)	TRACECK	TRACED[0]	-	-	-
1	10	同步跟踪 2 位		TRACECK	TRACED[0]	TRACED[1]	-	-
1	11	同步跟踪 4 位		TRACECK	TRACED[0]	TRACED[1]	TRACED[2]	TRACED[3]

1. 请参见数据手册中的复用功能映射表。
2. 使用串行模式时，释放此引脚。但使用 JTAG 时，此引脚分配给 JTDO。

注： *默认情况下，TPIU 的 TRACECLKIN 输入时钟接地。因此，将在 TRACE_IOEN 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

34.17.3 TPUI 格式化器

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)*

34.17.4 TPUI 帧同步数据包

TPUI 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，就不会在任何其他时间出现此序列。

  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。

34.17.5 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 TPUI 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_IOEN 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 TPUI 格式化（添加跟踪源 ID）。

34.17.6 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。

34.17.7 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。

使用 SW-DP 引脚时，TRACESWO 与 JTDO 复用。这样，所有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%。

34.17.8 TRACECLKIN 连接

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：*

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_IOEN 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。

34.17.9 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 247. 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®-M4，始终读为 0x00000008

34.17.10 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

34.18 DBG 寄存器映射

下表对调试寄存器进行了汇总。

表 248. 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRACE_MODE[1:0]		TRACE_IOEN	Res.	Res.	DBG_STANDBY	DBG_STOP	DBG_SLEEP
	Reset value																									0	0	0			0	0	0
0xE004 2008	DBGMCU_APB1_FZ	Res.	Res.	Res.	Res.	Res.	DBG_CAN2_STOP	DBG_CAN1_STOP	DBG_I2CFMP_SMBUS_TIMEOUT	DBG_I2C3_SMBUS_TIMEOUT	DBG_I2C2_SMBUS_TIMEOUT	DBG_CAN2_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP		DBG_TIM14_STOP	DBG_TI13_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
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.	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 34.6.1 节：MCU 器件 ID 代码*。

35 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他外部设备将其接口与 STM32F4xx 微控制器的特性自动匹配。

35.1 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
- 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：0x1FFF 7A10

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 (31:0 unique ID bits)

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 (63:32 unique ID bits)

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 (63:32 unique ID bits)

35.2 Flash 大小

Flash size

基址：0x1FFF 7A22

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 (Kbytes) 表示的设备 Flash 大小。

35.3 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.
					r	r	r								

位 15:11　保留，必须保持复位值。

位 10:8　**PKG[2:0]**：封装类型 (Package type)
0x111：UFBGA144/LQFP144
0x110：预留
0x101：预留
0x100：LQFP100
0x011：UFBGA100/WLCSP81
0x010：保留
0x001：LQFP64
0x000：UFQFPN48

位 7:0　保留，必须保持复位值。

36 版本历史

表 249. 文档版本历史

日期	版本	变更
2016 年 11 月 3 日	1	初始版本。
2016 年 12 月 15 日	2	更新了： – *第 18 节：通用定时器（TIM2 到 TIM5）* – *第 19 节：通用定时器（TIM9 到 TIM14）* – *第 20 节：基本定时器 (TIM6/7)*
2017 年 3 月 9 日	3	更新了： – *第 11.1 节：FSMC 主要特性* – *第 11.4 节：外部设备地址映射* – *第 11.4.1 节：NOR/PSRAM 地址映射* 增加了： – *第 12.3.2 节：QUADSPI 引脚* – *表 71：QUADSPI 引脚*
2017 年 5 月 3 日	4	更新了： – *第 13 节：模数转换器 (ADC)* – *第 25 节：实时时钟 (RTC)* – *第 29 节：串行外设接口/集成电路内置音频总线 (SPI/I2S)*
2017 年 6 月 12 日	5	更新了： – *表 5：Flash 模块构成* – *第 3.4.1 节：CPU 时钟频率与 Flash 读取时间之间的关系* – *第 3.5.3 节：擦除* – *第 3.6.4 节：写保护* – *第 3.8.5 节：Flash 控制寄存器 (FLASH_CR)* – *表 40：STM32F413/423 的向量表* – *第 10.2.5 节：外部中断/事件线映射* – *第 10.3 节：EXTI 寄存器* – *表 41：外部中断/事件控制器寄存器映射和复位值*
2017 年 9 月 22 日	6	更新了： – *表 13：Flash 寄存器映射与复位值* – *第 32 节：控制器局域网络 (bxCAN)*

表 249. 文档版本历史（续）

日期	版本	变更
2018 年 2 月 18 日	7	更新了： – *第 24 节：AES 硬件加速器 (AES)* – *第 33 节：USB on-the-go 全速 (OTG_FS)*
2018 年 5 月 23 日	8	更新了： – *图 13：时钟树* – *第 6.3.27 节：RCC 专用时钟配置寄存器 (RCC_DCKCFGR)* – *第 26.6 节：FMPI2C 中断* – *图 32：FSMC 存储区域* 删除了： – *图 387：I2C 中断映射图。*

索引

E

F

G

I

L

O

P

Q

R

S

T

U

W