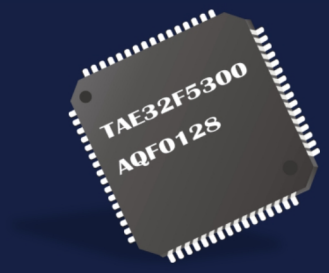
珠海泰为电子有限公司
Zhuhai Tai-Action Electronics CO., LTD.

Power Engine 系列
数字电源工业级芯片



芯片手册
Specification

版本号：Rev 1.0
修订日期：2021 年 3 月 20 日

目录

1 文档说明

1.1 缩写词

缩写词	说明
读写(R/W)	软件可以读写这些位
只读(R)	软件只能读取这些位
只写(W)	软件只能写入该位，读取该位时将返回复位值
读清 0(RC/W)	软件可以读写该位，读取该位时，将自动清零
写清 0(R/WC)	软件可以读写该位，写入 0 或 1 时，将自动清零
写 1 清 0(R/W1C)	软件可以读写该位，写入 1 时，将自动清零
写 0 清 0(R/W0C)	软件可以读写该位，写入 0 时，将自动清零

1.2 词汇表

本节简要介绍本文档中所用首字母缩略词和缩写词的定义：

- 在本文档中，将 Cortex™-M3 内核称为 M3
- CPU 内核集成了 SWD 调试端口：
  - SWD 调试端口(SWD-DP)提供基于串行线调试(SWD)协议的 2 引脚（时钟和数据）接口。

    有关 SWD 协议的信息，请参见《Cortex™-M3 技术参考手册》。
- 字：32 位数据/指令。
- 半字：16 位数据/指令。
- 字节：8 位数据。
- 双字：64 位数据。
- IAP（在应用中编程）：IAP 是指可以在用户程序运行期间对微控制器的 FLASH 进行重新编程。
- ICP（在线编程）：ICP 是指可以在器件安装于用户应用电路板上时使用 SWD 协议或自举程序对微控制器的 FLASH 进行编程。
- I-Code：此总线用于将 CPU 内核的指令总线连接到 FLASH 指令接口，通过此总线可执行预取操作。
- D-Code：此总线用于将 CPU 的 D-Code 总线（数据加载和调试访问）连接到 FLASH 数据接口。
- 选项字节：存储于 FLASH 中的产品配置位。
- AHB：高级高性能总线。
- CPU：指 Cortex™-M3 内核。
- FLASH：Embedded FLASH 的简称。

1.3 相关文档

- Cortex™-M3 技术参考手册，可按下述链接下载：
  http://infocenter.arm.com/help/topic/com.arm.doc.100165_0201_00_en/arm_cortexm3_processor_trm_100165_0201_00_en.pdf

- Cortex™-M3 Device 通用用户指南(含架构、指令集、核内外设等)，可按下述链接下载：
  http://infocenter.arm.com/help/topic/com.arm.doc.dui0552a/DUI0552A_cortex_m3_dgug.pdf

2 特性列表

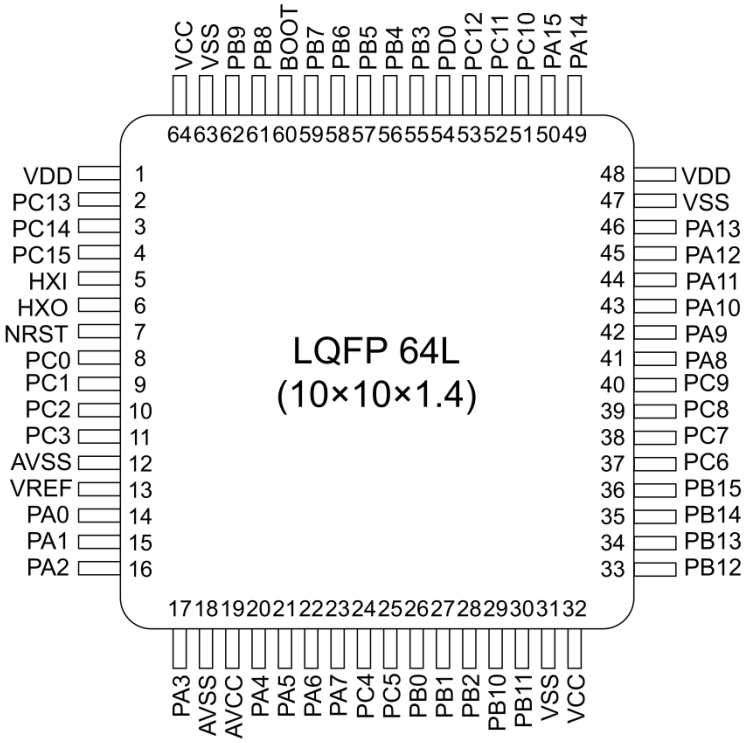
- 采用 ARM Cortex™-M3 32 位内核
  - 系统工作频率 90MHz
  - 内核内置 32 位硬件乘/除法
- 硬件加速器(ERPU)
  - 内置 5 路 IIR 单元,支持 1/2/3/4 阶灵活可配
  - 内置硬件计量模块(ECU),支持电流有效值(Irms)/电压有效值(Urms)/有功功率(P)/无功功率(Q)/视在功率(S)/功率因数(PF)/信号频率(f)的计算(负载 30%~50%时，精度范围为±%1；负载>50%~满载时，精度范围为±%0.5)
- 存储资源
  - 内置 80KB 容量、128bit 位宽 FLASH 程序存储器，支持 ECC 纠错、读写擦保护、加密及零延迟执行
  - 内置 16KB 容量数据 FLASH 存储器
  - 内置 16KB 系统 SRAM+8KB 算法 SRAM(两块 SRAM 4KB+4KB)
- 时钟、复位和电源管理
  - 支持单电源输入，输入范围为 3.1V-3.6V
  - 支持上下电复位及欠压复位
  - 内置独立看门狗(IWDG)和窗口看门狗(WWDG)
  - 内置高频 RC 振荡器:8MHz(全温度范围±1%精度)
  - 支持带晶振和无晶振方案，支持 6M-26MHz 晶体振荡器
  - 内置 3 套 PLL，支持 SSC 展频功能
  - 内置时钟安全系统，晶振异常检测电路并自动切换
  - 内置温度传感器
- DMA 控制器
  - 2 个独立的 DMA 通道
  - 支持内存与外设之间任意组合传输
- 通用 I/O(GPIO)
  - 64PIN 封装有 49 个 I/O、40PIN 封装有 30 个 I/O
  - 支持 I/O 功能复用映射，支持位操作
  - 所有 I/O 内建上拉/下拉电阻
  - 支持外部中断输入，支持上下边沿触发，可用于产生中断、事件唤醒,支持一路 NMI 中断
  - 内置 2 档驱动能力可调，最大 24mA 驱动能力
- 通用定时器(TIMER)
  - 内置 4 路 16 位定时器和 4 路 32 位定时器
  - 支持定时、PWM 输出及 Capture 捕获功能
  - 支持自动重载功能
- 高精度脉宽调制器(HRPWM)
  - 支持 6 对 PWM(12 路)输出，每路 PWM 输出最小分辨率为 195ps
  - 支持互补/独立输出模式，支持对称/不对称波形输出
  - 内置推挽、死区及斩波插入机制，支持 6 路故障和事件输入，内置故障及异常保护功能
  - 支持多路 HRPWM 间的同步机制
- 模数转换器(ADC)
  - 内置 2 个完全独立高速模数转换器(ADC)
  - 支持 2.2M SPS 采样速率，ADC 分辨率达到 13 位，有效位数 11 位
  - 支持 24 路模拟采样通道(每路 ADC 12 路，其中采样引脚 18 路)
  - 支持规则序列转换和注入序列转换，支持增益补偿与偏置补偿机制
  - 支持单次/不连续/连续采样模式，支持事件触发采样，支持 HRPWM 同步触发采样
  - 内置 ADC 基准源，支持内部基准电压
- 数模转换器(DAC)及比较器(CMP)
  - 内置 4 路 DAC 和 CMP
  - DAC 分辨率为 12 位， INL/DNL 小于 5 个 LSB
  - 支持锯齿波、三角波输出，支持方向与幅度可编程
  - CMP 转换延迟为最小值/典型值/最大值为 15ns/18ns/21ns
- 通讯接口外设
  - 内置 2 套 UART 接口，支持 RS485 通信
  - 内置 2 套 I2C 接口，支持 SMBus
  - 内置 1 套 CAN 控制器，支持 CAN2.0b
  - 内置 1 套 USB 接口，支持 USB2.0 协议，支持全速模式

- 内置 1 套 DALI 接口，支持 Master 和 Slave 模式，支持 1.2/2.4/4.8K 多种波特率
- PVD 欠压监测功能(4 档可调)
- 128 位芯片唯一标识码
- 两种低功耗模式: 睡眠模式和停机模式
- 支持 2 线 SWD 调试接口
- 工作环境温度：-40℃~125℃，整机 ESD: ≥±8KV
- 封装：LQFP64(10*10)、WQFN40(6*6)

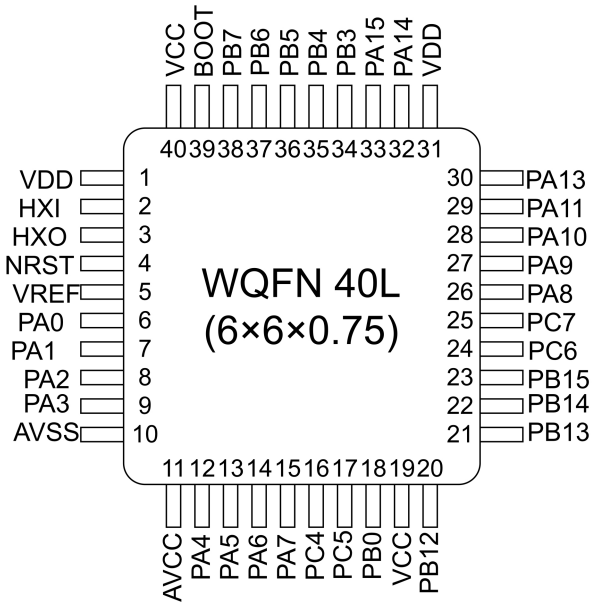
3 引脚定义

3.1 引脚框图

LQFP 64L:



WQFN 40L:



3.2 引脚描述

表 3-1 引脚定义缩写说明

名称	缩写	定义
引脚名称	除非在引脚名称下使用括号特别说明，否则在复位期间和复位之后的引脚功能均与实际引脚名称相同。	
引脚类型	S	电源引脚
	SO	电源输出引脚
	I	仅输入引脚
	I/O	输入/输出引脚
引脚架构	TT	3.3V 容忍 I/O
	B	专用 BOOT 引脚
	RST	嵌入了弱上拉电阻的复位引脚
注释	除非通过注释特别说明，否则所有 I/O 在复位期间和复位之后均设置为悬空状态	
复用功能	通过 GPIOx_MUX 寄存器选择的功能	

表 3-2 引脚定义

Pin Number			Pin name	Pin type	I/O structure	Digital Functions	Analog Functions
	LQFP40 WQFN40	LQFP64 WQFN64					
	1	1	VDD	SO	-	-	-
	-	2	PC13	I/O	TT	TMR5, TMR6, DALI_TX	-
	-	3	PC14	I/O	TT	TMR0, DALI_RX	-
	-	4	PC15	I/O	TT	TMR1, TMR4	-
	-	-	VSS			-	-
	2	5	HXI	-		-	-
	-	-	AVSS			-	-
	3	6	HXO	-		-	-
	4	7	NRST	I	RST	-	-
	-	8	PC0	I/O	TT	TMR4	ADC0/1_IN6
	-	9	PC1	I/O	TT	-	ADC0/1_IN7
	-	10	PC2	I/O	TT	TMR6	ADC0/1_IN8
	-	11	PC3	I/O	TT	TMR7	ADC0/1_IN9
	-	12	AVSS	S	-	-	-
	5	13	VREF	SO	-	-	-
	6	14	PA0	I/O	TT	TMR4, DALI_TX, CMP3_OUT	ADC0_IN0
	7	15	PA1	I/O	TT	TMR0, TMR5, DALI_RX, I2C1_SMBA, UART1_DE	ADC0_IN1
	8	16	PA2	I/O	TT	TMR0, TMR6, I2C1_SCL, UART1_TX, CMP0_OUT	ADC0_IN2, CMP0_INM_0

	9	17	PA3	I/O	TT	TMR1, TMR7, I2C1_SDA, UART1_RX	ADC0_IN3
	10	18	AVSS	S	-	-	-
	11	19	AVCC	S	-	-	-
	12	20	PA4*	I/O	TT	TMR1, DALI_TX, I2C1_SMBA, UART0_TX	ADC1_IN0, DAC0_OUT, CMP0_INM_1, CMP1_INM_1, CMP2_INM_1, CMP3_INM_1
	13	21	PA5*	I/O	TT	TMR4, CAN_TX, DALI_RX, I2C1_SCL, UART0_RX, CMP3_OUT	ADC1_IN1, DAC1_OUT
	14	22	PA6	I/O	TT	TMR0, TMR2, CAN_RX, I2C1_SDA, UART0_DE	ADC_IN2, DAC2_OUT
	15	23	PA7	I/O	TT	TMR1, TMR3, TMR4, I2C1_SMBS	ADC1_IN3, CMP0_INP
	16	24	PC4(1)	I/O	TT	TMR4, UART0_TX, CMP3_OUT	ADC1_IN4, CMP3_INP
	17	25	PC5(1)	I/O	TT	UART0_RX	ADC1_IN5, DAC3_OUT
	18	26	PB0	I/O	TT	TMR2, TMR5	ADC0_IN4, CMP1_INP
	-	-	VDD	SO	-	-	-
	-	-	VSS	S	-	-	-
	-	-	NRST	I	RST		
	-	27	PB1	I/O	TT	TMR3, TMR6, UART1_DE, HRPWM_EVT0, HRPWM_SCIN, HRPWM_SCOUT, CMP1_OUT	ADC0_IN5
	-	28	PB2	I/O	TT	CAN_TX, I2C0_SCL, UART1_TX, HRPWM_SCIN	ADC1_IN10, CMP1_INM_0
	-	29	PB10	I/O	TT	TMR6, CAN_RX, I2C0_SDA, UART1_RX, UART1_TX, HRPWM_FLT2	ADC0_IN10
	-	30	PB11	I/O	TT	TMR7, UART1_RX, HRPWM_FLT3	CMP2_INP
	-	31	VSS	S	-	-	-
	19	32	VCC	S	-	-	-

	20	33	PB12	I/O	TT	DALI_TX, HRPWM_OUT2A	-
	21	34	PB13	I/O	TT	TMR4, DALI_RX, HRPWM_OUT2B	-
	22	35	PB14	I/O	TT	TMR0, TMR5, UART1_DE, HRPWM_OUT3A	CMP3_INM_0
	23	36	PB15	I/O	TT	TMR2, TMR1, TMR6, HRPWM_OUT3B	CMP2_INM_0
	24	37	PC6	I/O	TT	TMR0, HRPWM_EVT4, HRPWM_OUT5A	-
	25	38	PC7	I/O	TT	TMR1, UART0_DE, HRPWM_FLT4, HRPWM_OUT5B	-
	-	39	PC8	I/O	TT	TMR2, HRPWM_OUT4A	-
	-	40	PC9	I/O	TT	TMR3, HRPWM_OUT4B	-
	26	41	PA8	I/O	TT	MCO, TMR4, CAN_TX, UART0_RX, HRPWM_OUT0A	-
	27	42	PA9	I/O	TT	TMR6, TMR5, CAN_RX, UART0_TX, HRPWM_OUT0B	-
	28	43	PA10	I/O	TT	TMR7, TMR6, CAN_TX, I2C0_SMBA, UART0_RX, HRPWM_OUT1A, CMP2_OUT	-
	29	44	PA11	I/O	TT	TMR5, TMR7, CAN_RX, I2C0_SMBS, UART0_TX, HRPWM_OUT1B	-
	-	45	PA12	I/O	TT	TMR2, TMR4, TMR5, CAN_TX, UART0_DE, HRPWM_FLT0, CMP0_OUT	-
	30	46	PA13	I/O	TT	SWDAT, TMR2, I2C1_SCL, HRPWM_FLT4, CMP2_OUT	-
	-	47	VSS	S	-	-	-
	31	48	VDD	S	-	-	-

	32	49	PA14	I/O	TT	SWCLK, TMR5, I2C0_SDA, I2C1_SDA, UART1_TX, HRPWM_FLT5, HRPWM_SCIN	-
	33	50	PA15	I/O	TT	TMR4, I2C0_SMBA, I2C0_SCL, UART1_TX, UART1_RX, HRPWM_FLT1	-
	-	51	PC10	I/O	TT	I2C0_SMBS,UART1_RX, UART0_TX, HRPWM_FLT4, HRPWM_FLT0	-
	-	-	VSS	S	-	-	-
	-	52	PC11	I/O	TT	UART0_RX, HRPWM_EVT1	-
	-	53	PC12	I/O	TT	UART1_DE, UART0_DE, HRPWM_EVT0, HRPWM_FLT2	-
	-	54	PD0	I/O	TT	TMR0, HRPWM_FLT5, HRPWM_FLT3, CMP2_OUT	-
	34	55	PB3	I/O	TT	SWO, TMR1, TMR5, I2C1_SCL, UART1_TX, HRPWM_SCOUT, HRPWM_EVT2	USB-DM
	35	56	PB4	I/O	TT	TMR0, I2C1_SDA, UART1_RX, HRPWM_EVT1	USB-DP
	36	57	PB5	I/O	TT	TMR2, I2C0_SMBA, I2C1_SMBA, HRPWM_EVT5, CMP3_OUT	-
	37	58	PB6	I/O	TT	TMR2, I2C0_SCL, I2C1_SMBS, UART0_TX, HRPWM_EVT0, HRPWM_SCIN, HRPWM_EVT3	-
	38	59	PB7	I/O	TT	TMR3, TMR0, I2C0_SDA, UART0_RX, HRPWM_EVT4, CMP1_OUT	-
	39	60	BOOT	I	B	-	

	-	61	PB8	I/O	TT	TMR2, CAN_RX, I2C0_SCL, UART1_RX, HRPWM_EVT2	-
	-	62	PB9	I/O	TT	TMR3, CAN_TX, I2C0_SDA, UART1_TX, HRPWM_EVT3, CMP0_OUT	-
	-	63	VSS	S	-	-	-
	40	64	VCC	S	-	-	-
	-	-	VSS	S	-	-	-
	-	-	VDD	SO	-	-	-

**注意：(1)PC4、PC5 保持悬空。*

表 2-3 引脚功能复用表

Pin name	Functions															
	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	AF8	AF9	AF10	AF11	AF12	AF13	AF14	AF15
	IN	OUT	SYS	TMR0-3	TMR0-3	TMR4-7	TMR4-7	CAN/I2C0	I2C0/DALI	I2C1/UART1	UART0/UART1	HRPWM/USB	HRPWM	HRPWM	CMP0-3	Analog function
PA0	-	-	-	-	-	TMR4	-	-	DALI_TX	-			-	-	CMP3_OUT	ADC0_IN0
PA1	-	-	-	-	TMR0	TMR5	-	-	DALI_RX	I2C1_SMBA	UART1_DE		-	-	-	ADC0_IN1
PA2	-	-	-	-	TMR0	TMR6	-	-	-	I2C1_SCL	UART1_TX		-	-	CMP0_OUT	ADC0_IN2,CMP0_INM_0
PA3	-	-	-	-	TMR1	TMR7	-	-	-	I2C1_SDA	UART1_RX		-	-	-	ADC0_IN3
PA4	-	-	-	TMR1	-	-	-	-	DALI_TX	I2C1_SMBA	UART0_TX		-		-	ADC1_IN0,DAC0_OUT, CMP0_INM_1,CMP1_INM_1, CMP2_INM_1,CMP3_INM_1
PA5	-	-	-	-	-	TMR4	-	CAN_TX	DALI_RX	I2C1_SCL	UART0_RX		-		CMP3_OUT	ADC1_IN1,DAC1_OUT
PA6	-	-	-	TMR0	TMR2	-	-	CAN_RX	-	I2C1_SDA	UART0_DE		-	-		ADC1_IN2,DAC2_OUT
PA7	-	-	-	TMR1	TMR3	-	TMR4	-	-	I2C1_SMBS	-		-		-	ADC1_IN3,CMP0_INP
PA8	-	-	MCO	-	-	-	TMR4	CAN_TX		-	UART0_RX	-	-	HRPWM_OUT0A	-	-
PA9	-	-	-	-		TMR6	TMR5	CAN_RX	-	-	UART0_TX	-	-	HRPWM_OUT0B	-	-
PA10	-	-	-	-		TMR7	TMR6	CAN_TX	I2C0_SMBA	-	UART0_RX		-	HRPWM_OUT1A	CMP2_OUT	-
PA11	-	-	-	-	-	TMR5	TMR7	CAN_RX	I2C0_SMBS	-	UART0_TX	-	-	HRPWM_OUT1B	-	-
PA12	-	-	-	-	TMR2	TMR4	TMR5	CAN_TX	-	-	UART0_DE	-	-	HRPWM_FLT0	CMP0_OUT	-
PA13	-	-	SWDAT	-	TMR2	-	-	-		I2C1_SCL	-	HRPWM_FLT4	-	-	CMP2_OUT	-
PA14	-	-	SWCLK	-	-	-	TMR5	-	I2C0_SDA	I2C1_SDA	UART1_TX	HRPWM_FLT5	HRPWM_SCIN	-	-	-
PA15	-	-	-	-	-	TMR4	-	I2C0_SMBA	I2C0_SCL	UART1_TX	UART1_RX		-	HRPWM_FLT1		-
PB0	-	-	-	TMR2	-	-	TMR5	-	-	-	-		-	-	-	ADC0_IN4,CMP1_INP
PB1	-	-	-	TMR3	-	-	TMR6	-	-	UART1_DE		HRPWM_EVT0	HRPWM_SCIN	HRPWM_SCOUT	CMP1_OUT	ADC0_IN5
PB2	-	-	-	-	-		-	CAN_TX	I2C0_SCL	UART1_TX	-		-	HRPWM_SCIN	-	ADC1_IN10,CMP1_INM_0
PB3	-	-	SWO	TMR1	-	TMR5	-		-	I2C1_SCL	UART1_TX	-	HRPWM_SCOUT	HRPWM_EVT2		USB-DM
PB4	-	-	-	TMR0		-	-		-	I2C1_SDA	UART1_RX			HRPWM_EVT1	-	USB-DP
PB5	-	-	-		TMR2	-	-	-	I2C0_SMBA	I2C1_SMBA	-	-		HRPWM_EVT5	CMP3_OUT	-

PB6	-	-	-	-	TMR2		-		I2C0_SCL	I2C1_SMBS	UART0_TX	HRPWM_EVT0	HRPWM_SCIN	HRPWM_EVT3	-	-
PB7	-	-	-	TMR3	TMR0		-		I2C0_SDA	-	UART0_RX		-	HRPWM_EVT4	CMP1_OUT	-
PB8	-	-	-	-	TMR2	-	-	CAN_RX	I2C0_SCL	-	UART1_RX		-	HRPWM_EVT2	-	-
PB9	-	-	-	-	TMR3	-	-	CAN_TX	I2C0_SDA	-	UART1_TX	-	-	HRPWM_EVT3	CMP0_OUT	-
PB10	-	-	-	-	-	TMR6	-	CAN_RX	I2C0_SDA	UART1_RX	UART1_TX		-	HRPWM_FLT2	-	ADC0_IN10
PB11	-	-	-	-	-	TMR7	-	-		-	UART1_RX	-		HRPWM_FLT3	-	CMP2_INP
PB12	-	-	-	-	-	-	-	-	DALI_TX	-			-	HRPWM_OUT2A	-	-
PB13	-	-	-	-	-	-	TMR4	-	DALI_RX	-	-		-	HRPWM_OUT2B	-	-
PB14	-	-	-	-	TMR0	-	TMR5	-	-	-	UART1_DE		-	HRPWM_OUT3A	-	CMP3_INM_0
PB15	-	-	-	TMR2	TMR1	-	TMR6	-	-	-	-	-	-	HRPWM_OUT3B	-	CMP2_INM_0
PC0	-	-	-	-	-	-	TMR4	-	-	-	-	-	-		-	ADC0/1_IN6
PC1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ADC0/1_IN7
PC2	-	-	-	-	-	-	TMR6	-	-	-	-	-	-		-	ADC0/1_IN8
PC3	-	-	-	-	-	-	TMR7	-	-	-	-	-	-	-	-	ADC0/1_IN9
PC4	-	-	-	-	-	-	TMR4	-	-	-	UART0_TX	-	-	-	CMP3_OUT	ADC1_IN4,CMP3_INP
PC5	-	-	-	-		-	-	-	-	-	UART0_RX		-	-	-	ADC1_IN5,DAC3_OUT
PC6	-	-	-	TMR0	-	-	-	-	-	-	-	HRPWM_EVT4	-	HRPWM_OUT5A		-
PC7	-	-	-	TMR1	-	-	-	-	-	-	UART0_DE	HRPWM_FLT4	-	HRPWM_OUT5B	-	-
PC8	-	-	-	TMR2	-	-	-	-		-	-	HRPWM_OUT4A	-	-		
PC9	-	-	-	TMR3	-	-	-	-		-	-	HRPWM_OUT4B	-	-		
PC10	-	-	-	-	-	-	-	I2C0_SMBS		UART1_RX	UART0_TX		HRPWM_FLT4	HRPWM_FLT0	-	-
PC11	-	-	-	-	-	-	-				UART0_RX	HRPWM_EVT1	-		-	-
PC12	-	-	-	-	-	-	-			UART1_DE	UART0_DE	HRPWM_EVT0	-	HRPWM_FLT2	-	-
PC13	-	-	-	-	-	TMR6	TMR5	-	DALI_TX	-	-	-	-	-	-	-
PC14	-	-	-	TMR0	-	-		-	DALI_RX	-	-	-	-	-	-	-
PC15	-	-	-	-	TMR1	TMR4	-	-	-	-	-	-	-	-	-	-
PD0	-	-	-	TMR0	-	-	-	-	-	-	-	HRPWM_FLT5	-	HRPWM_FLT3	CMP2_OUT	-

4 存储器和总线架构

4.1 系统架构

芯片主系统由多层 AHB 总线矩阵互联，如下图所示：

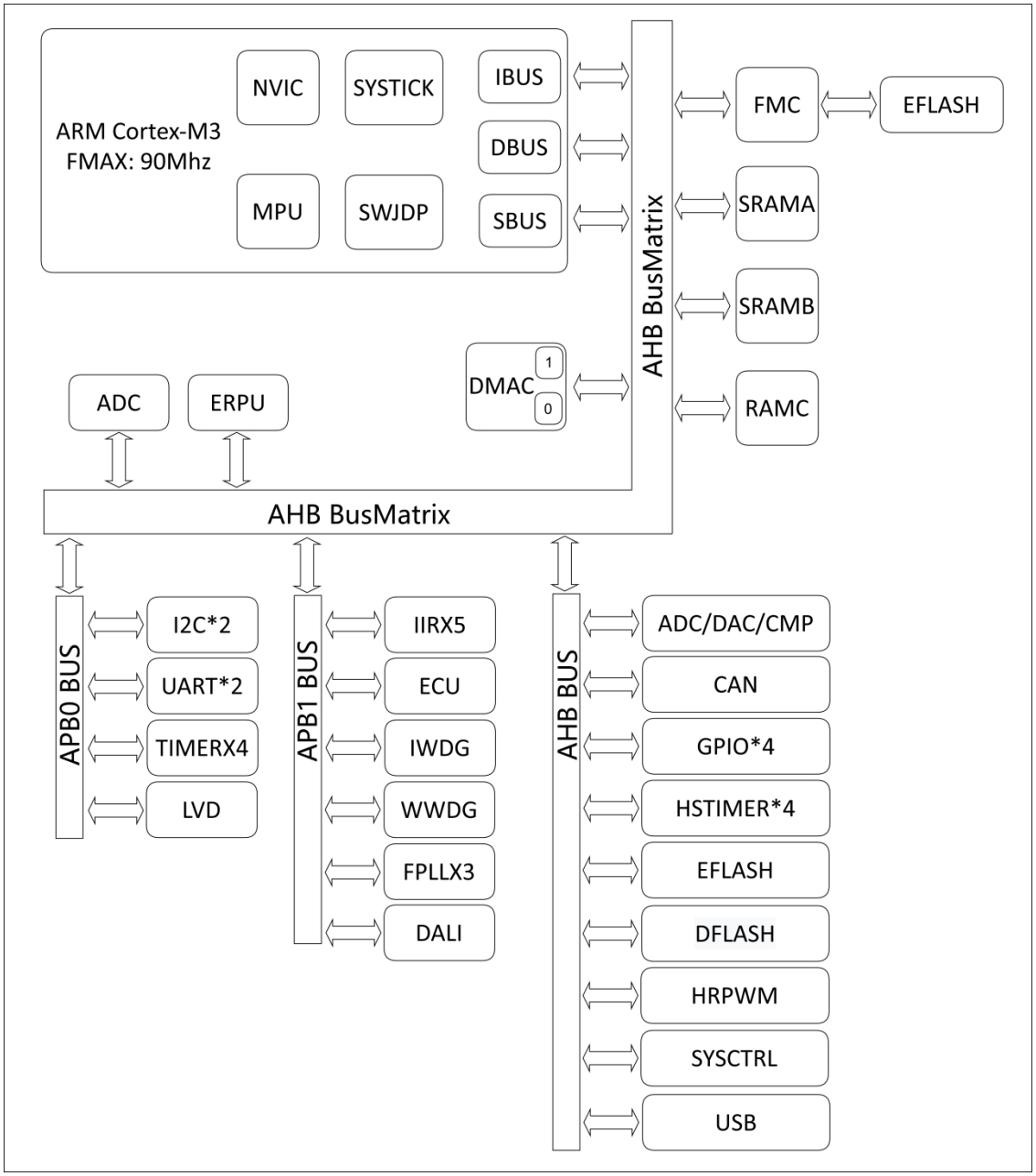


图 4-1 总线架构

4.2 总线矩阵

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：

- 五条主控总线：
  - Cortex™-M3 内核 I 总线、D 总线和 S 总线
  - DMA 存储器总线 0
  - DMA 存储器总线 1
- 八条被控总线：
  - EFLASH ICode 总线
  - EFLASH DCode 总线
  - SRAMA(16KB)
  - SRAMB(4KB)（用于 ADC 和 ERPU 数据存储）
  - SRAMC(4KB)（用于 ADC 和 ERPU 数据存储）
  - AHB0 外设
  - APB0 外设
  - APB1 外设

通过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 4-1 所示。

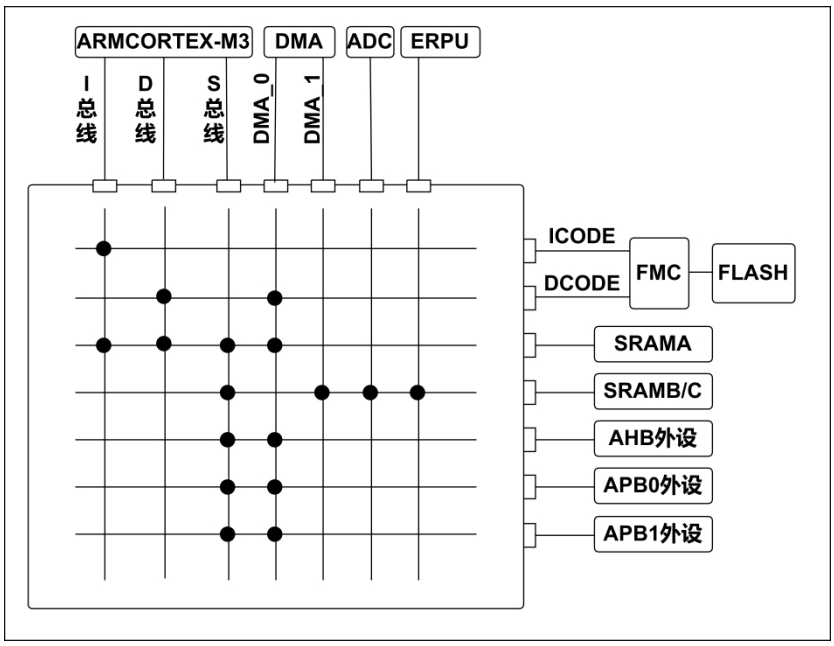


图 4-2　系统总线互联矩阵

I 总线：此总线用于将 Cortex™-M3 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/SRAM）。

D 总线：此总线用于将 Cortex™-M3 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/SRAM）。

S 总线：此总线用于 Cortex™-M3 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或 SRAM 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 16KB、

4KB、4KB 的内部 SRAM、包括 APB 外设在内的 AHB1 外设、AHB2 外设。

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/SRAM（16KB、4KB、4KB）。

总线矩阵：总线矩阵用于主控总线之间的访问仲裁管理。

AHB/APB 总线桥：通过一个 AHB 和两个 APB 总线桥（APB0/APB1），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。

4.3 存储器地址映射

程序存储器、数据存储器、寄存器和 I/O 端口排列在同一个顺序的 4GB 地址空间内。

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

未分配给片上存储器和外设的所有存储区域均视为“保留区”，有关外设寄存器映射的详细信息，请参见外设寄存器映射章节。

遵循 ARM® Cortex™-M3 对存储器的规定，存储器基本组织架构如下：

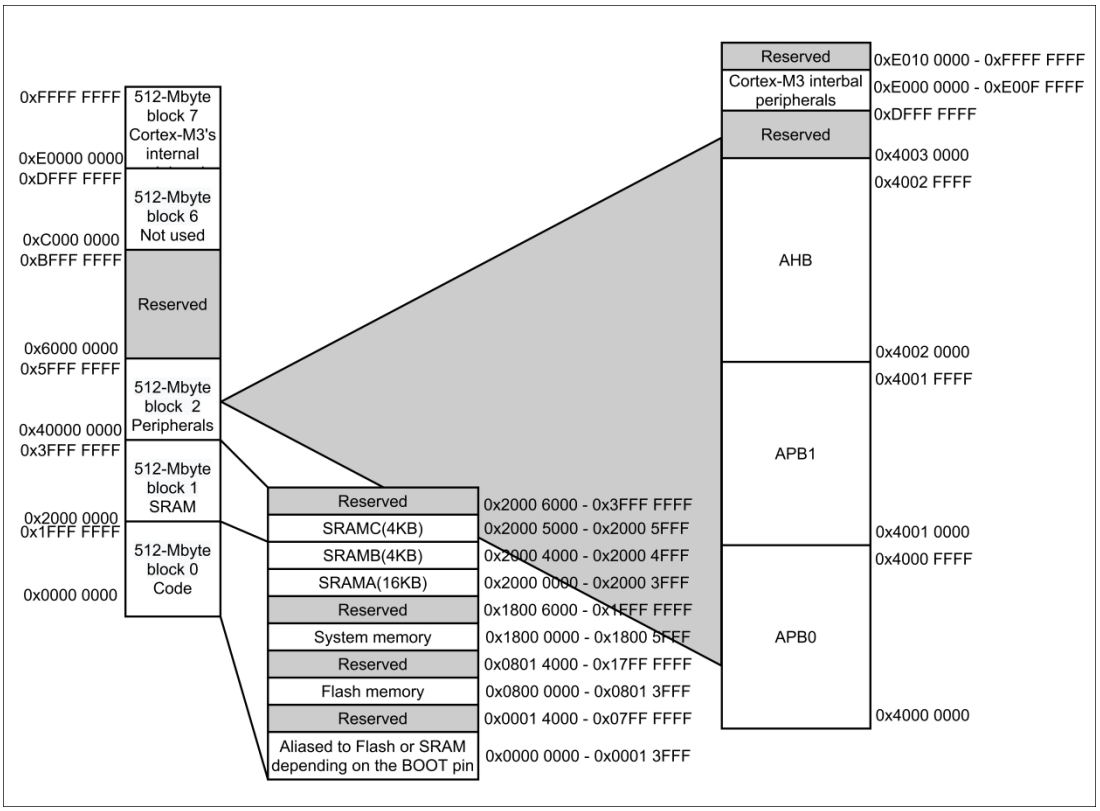


图 4-3 存储映射图

4.4 外设寄存器映射

下表提供了芯片外设地址边界划分，详细请参考表 4-1：

表 4-1 芯片外设地址映射表

总线	边界地址	外设
Reserved	0xE010 0000 - 0xFFFF FFFF	Reserved
Cortex™-M3	0xE000 0000 - 0xE00F FFFF	Cortex™-M3 internal peripherals
Reserved	0x4003 0000 - 0xDFFF FFFF	Reserved
AHB	0x4002 0000 - 0x4002 0FFF	DMA
	0x4002 2000 - 0x4002 2FFF	CAN
	0x4002 3000 - 0x4002 3FFF	SYSCTRL
	0x4002 4000 - 0x4002 4FFF	GPIOA
	0x4002 5000 - 0x4002 5FFF	GPIOB
	0x4002 6000 - 0x4002 6FFF	GPIOC
	0x4002 7000 - 0x4002 7FFF	GPIOD
	0x4002 8000 - 0x4002 80FF	HSTIMER0
	0x4002 8100 - 0x4002 81FF	HSTIMER1
	0x4002 8200 - 0x4002 82FF	HSTIMER2
	0x4002 8300 - 0x4002 83FF	HSTIMER3
	0x4002 8400 - 0x4002 84FF	HSTIMER COMMON
	0x4002 9000 - 0x4002 9FFF	FLASH CTRL
	0x4002 A000 - 0x4002 AFFF	DFLASH CTRL
	0x4002 C000 - 0x4002 CFFF	USB
	0x4002 D000 - 0x4002 DFFF	HRPWM
	0x4002 E000 - 0x4002 E3FF	ADC0
	0x4002 E400 - 0x4002 E7FF	ADC1
	0x4002 E800 - 0x4002 EBFF	DAC
	0x4002 EC00 - 0x4002 EFFF	CMP
APB1	0x4001 0000 - 0x4001 0FFF	CAN
	0x4001 7000 - 0x4001 7FFF	IWDG
	0x4001 8000 - 0x4001 8FFF	WWDG
	0x4001 9000 - 0x4001 9FFF	FPLL0
	0x4001 A000 - 0x4001 AFFF	FPLL1
	0x4001 B000 - 0x4001 BFFF	FPLL2
	0x4001 F500 - 0x4001 F9FF	IIR
	0x4001 FF00 - 0x4001 FFFF	ECU
APB0	0x4000 0000 - 0x4000 0FFF	I2C0
	0x4000 1000 - 0x4000 1FFF	I2C1
	0x4000 4000 - 0x4000 4FFF	UART0
	0x4000 5000 - 0x4000 5FFF	UART1
	0x4000 C000 - 0x4000 C0FF	TIMER0

4.4.1 SRAM

系统 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，系统 SRAM 分为三个块：

- 映射在地址 0x2000 0000 的 SRAMA(16KB)块，可供所有 AHB 主控总线访问。
- 映射在地址 0x2000 4000 的 SRAMA(4KB)块，可供所有 AHB 主控总线，DMA 主控总线及 ADC、ERPU 访问。AHB 主总线支持并发 SRAM 访问，例如，当 CPU 对 4 KB SRAM 进行读/写操作时，ADC、ERPU 可以同时对 4KB SRAM 进行读/写操作。
- 映射在地址 0x2000 5000 的 SRAMA(4KB)块，可供所有 AHB 主控总线，DMA 主控总线及 ADC、ERPU 访问。AHB 主总线支持并发 SRAM 访问，例如，当 CPU 对 4 KB SRAM 进行读/写操作时，ADC、ERPU 可以同时对 4KB SRAM 进行读/写操作。
- 在地址 0x1000 0000 映射的 16KB 块，可供所有 AHB 主控总线访问。

如果选择从 SRAM 自举，则 CPU 可通过系统总线或 I-Code/D-Code 总线访问系统 SRAM。要在 SRAM 执行期间获得最佳的性能，应选择物理重映射（通过自举管脚）。

4.4.2 FLASH

Embedded FLASH Controller 接口用于管理 CPU 通过 I 总线(I-Code Bus)及 D 总线(D-Code Bus)对 Embedded FLASH 的访问。通过 Embedded FLASH Controller 可以对 FLASH 存储体执行擦除与编程操作，并对 FLASH 存储体实现读/写保护机制，并实现了 FLASH 存储体的数据加/解密功能。

FLASH 结构如下：

- 主存储器块分为多个扇区。
- 系统存储器，芯片在系统存储器自举模式下从该存储器启动。
- 选项字节，用于配置读保护、写保护及加密等功能。

4.5 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），而数据区域起始地址为 0x2000 0000（通过系统总线访问）。Cortex™-M3 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。芯片实施一种特殊机制，可以从其它存储器（如内部 SRAM）进行自举。

在芯片中，可通过 BOOT 引脚选择两种不同的自举模式，如表 4-2 所示。

表 4-2 自举模式

自举模式选择引脚	自举模式	自举空间
BOOT		
0	FLASH	选择 FLASH 作为自举空间
1	SRAM	选择 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 BOOT

引脚来选择需要的自举模式。

BOOT 引脚只有在芯片重新上电时，才会对 BOOT 引脚重新采样。因此，这样的启动结束后，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。

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

5 嵌入式 FLASH 接口（FLASH）

5.1 简介

FLASH Controller 接口用于管理 CPU 通过 I 总线(I-Code Bus)及 D 总线(D-Code Bus)对 FLASH 的访问。通过 FLASH Controller 可以对 FLASH 存储体执行擦除与编程操作，并对 FLASH 存储体实现读/写保护机制，并实现了 FLASH 存储体的数据加/解密功能。

5.2 结构框图

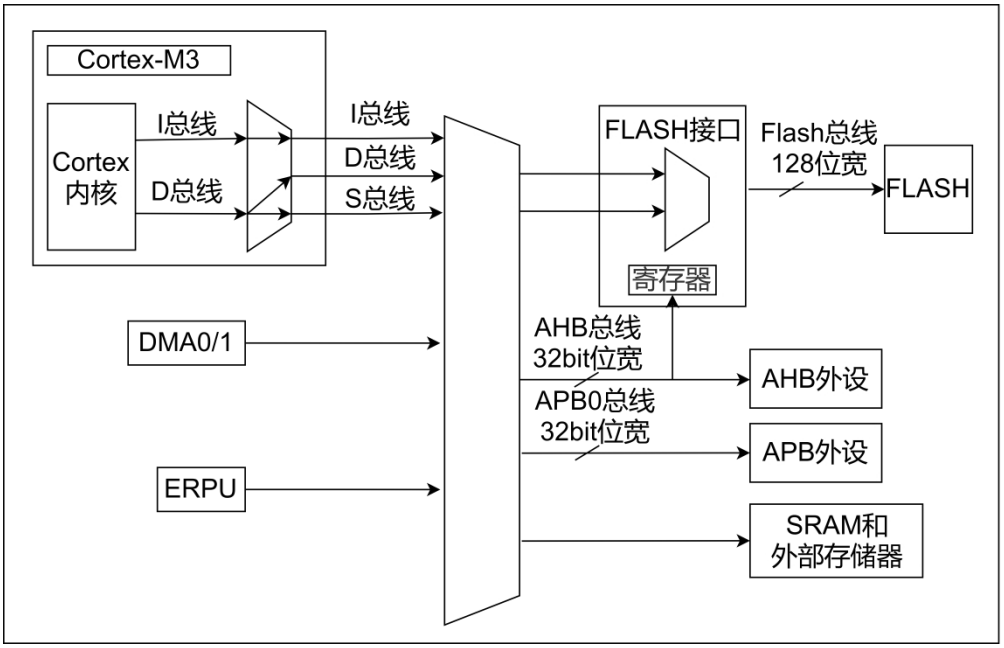


图 5-1 FLASH 顶层结构框图

5.3 主要特性

FLASH 主要具有以下特性：

- 存储器高达 80KB(其中 5K 用于 ECC，实际可用为 75K)
- 高达 160 扇区，每个扇区 480 字节
- 支持按整片/扇区擦除(Chip/Sector Erase)
- FLASH 可作为启动代码空间(通过 BOOT 引脚选择)存储和执行用户代码
- FLASH 支持 128 位宽编程功能，支持按字节/半字/字读取 FLASH
- 增强安全功能
  - FLASH 读保护功能(RDP)
  - FLASH 写保护功能(WRP)
  - FLASH 存储体加密存储，实现对用户代码的保护，并实现零等待加解密操作
- FLASH I/D 总线 128Bit 位宽预取、缓存功能
  - FLASH I-Code Bus 上 512 Byte 缓存
  - FLASH D-Code Bus 上 256 Byte 缓存
- 误码矫正(ECC)，支持纠一检错
- FLASH 低功耗模式

5.4 功能描述

5.4.1 实时加速器

为了高效地发挥处理器的性能，通过加速器实现指令预取队列及分支缓存机制，减少 FLASH 读取指令时等待的情况，从而提高了 128 位 FLASH 程序的执行速度。

指令预期

FLASH 每次读操作可读取 128 位数据，即可以是 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 读取当前指令行时，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。可通过 FLASH_ECR 寄存器中的 DBPE/ IBPE 位置 1 来使能预取功能。

图 5-2、图 5-3 为需要 3 个等待周期访问 FLASH 时连续 32 位指令的执行过程，分别介绍了不使用和使用预取操作两种情况。

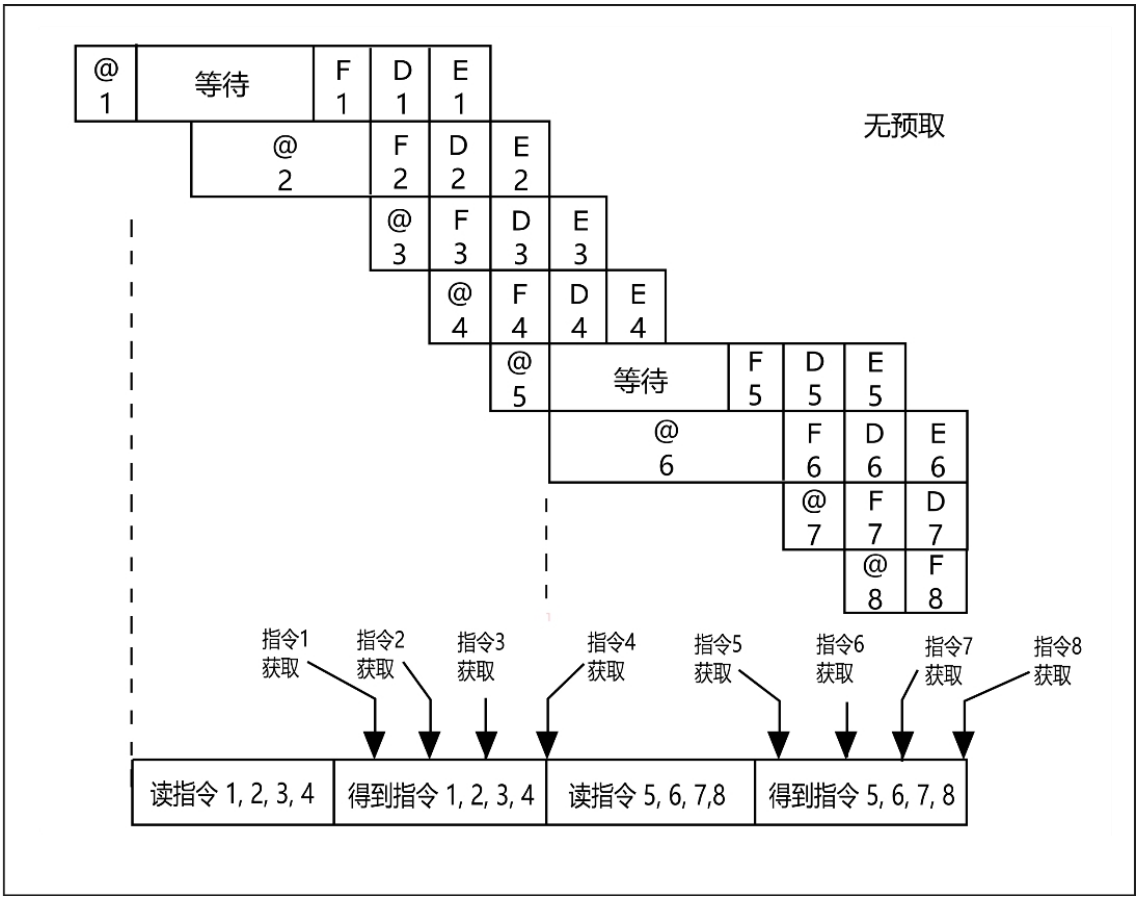


图 5-2　连续 32 位指令的执行过程（不使用预取操作）

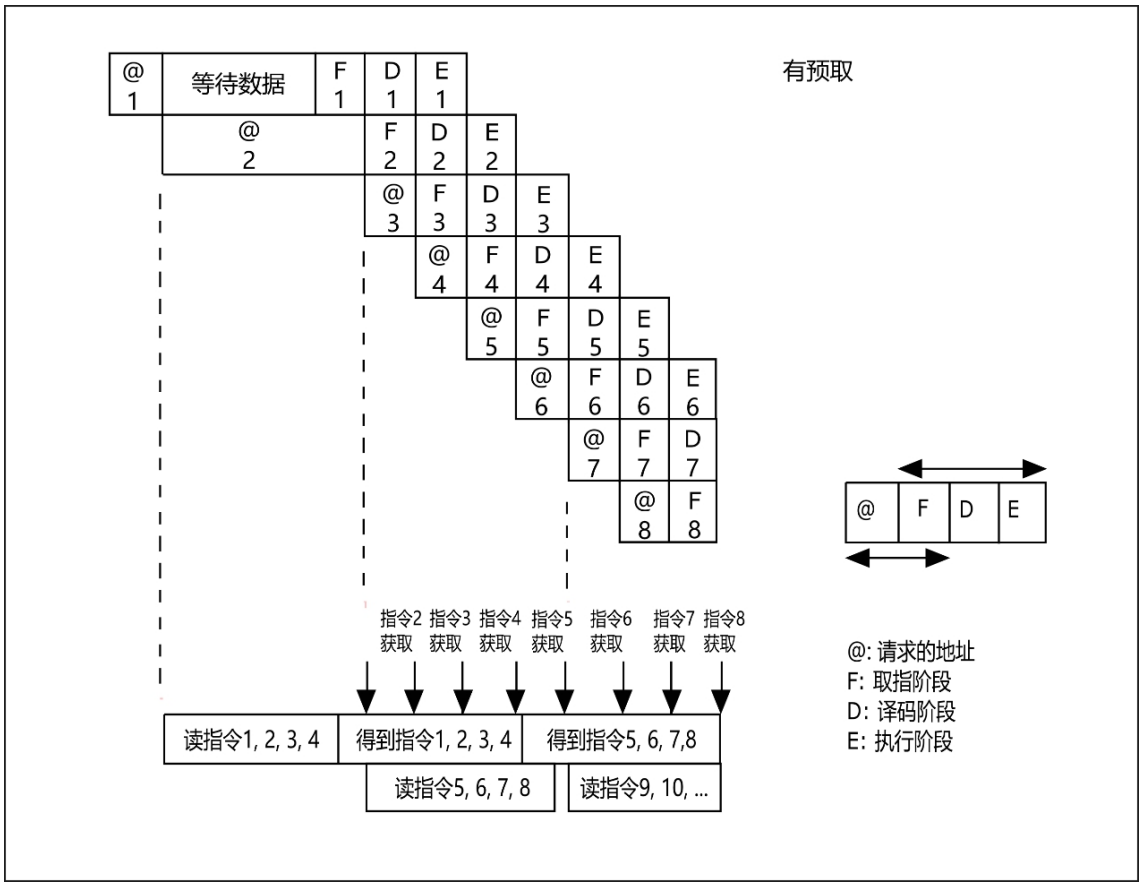


图 5-3　连续 32 位指令的执行过程（使用预取操作）

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消耗，可将 32 行 128 位(512 Byte)的指令保存到指令缓存(I-Cache)存储器中。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，同样添加了数据缓存(D-Cache)功能，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 16 行 128 位(256 Byte)以内。

5.4.2 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，建议将 HCLK 时钟频率配置为 80M。

在对 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，写入/擦除操作进行期间不能从 FLASH 中执行代码或数据获取操作。

注意：FLASH 所有操作均为串行执行，确保上一笔操作完成后再发起下一笔操作。

5.4.2.1 FLASH 控制寄存器解锁

FLASH 中用于解锁控制寄存器的 Key 值有以下几种：

- KEY1 = 0x45670123
- KEY2 = 0xCDEF89AB
- OPTKEY= 0x50035003
- LPKEY = 0x5003

FLASH 的写/擦除控制解锁

复位后，FLASH 控制寄存器(FLASH_ECR)不允许执行写/擦操作（读不需要解锁），以防因电气干扰等原因出现对 FLASH 的意外操作。当需要执行 FLASH 的写/擦操作时，必须先执行解锁，然后发起相应的操作。

- 解锁方式

向 FLASH 密钥寄存器(FLASH_KEY)中依次写入 KEY1 和 KEY2，成功解锁后，FLASH_CR 寄存器中的 LOCK 位将会自动置零，这时即可对 FLASH 进行写/擦除操作。

- 加锁方式

完成所需的写/擦除操作后，通过软件将 FLASH_CR 寄存器中的 LOCK 位设置为 1，将重新锁定 FLASH，实现加锁功能，下次再进行写/擦除操作前，需要重新进行解锁操作。

注意：

- *进行写/擦除操作前，必须对 FLASH 进行解锁，否则将导致写/擦除操作无效；*
- *解锁过程必须严格按照解锁顺序，如果顺序错误，则会解锁失败，FLASH_CR 寄存器中的 LOCK 标志位将不会清除，解锁无效；*
- *建议在完成目标操作后，对 FLASH 进行加锁，避免意外操作。*

FLASH 的读/写保护操作解锁

在对 FLASH 进行读/写保护修改前，需要写入特定的密钥进行操作解锁，避免意外的读/写保护产生。

- 解锁方式

向 FLASH 密钥寄存器(FLASH_KEY)写入 OPTKEY 值即可解锁。

- 加锁方式
  向 FLASH 密钥寄存器(FLASH_KEY)写入其他非 OPTKEY 值即重新加锁。

注意：读/写保护解锁只要 FLASH 密钥寄存器(FLASH_KEY)保持 OPTKEY 值，即可对读/写保护进行修改，直至 FLASH_KEY 写入其他非 OPTKEY 值。因此建议在完成读/写保护的修改后，对 OPTKEY 写入其他值(推荐写 0)加锁，避免意外操作。

FLASH 的低功耗操作解锁

FLASH 支持低功耗模式，可以按需进入/退出 Standby 模式，降低系统功耗。操作前需要对低功耗操作进行解锁，避免意外的操作。

- 解锁方式
  向 FLASH 低功耗寄存器的高 4 字节(FLASH_LPR[31:16])写入 LPKEY 值即可解锁操作。
- 加锁方式
  向FLASH低功耗寄存器的高4字节(FLASH_LPR[31:16])写入其他值即可加锁（推荐写0）。

5.4.2.2 FLASH 擦除

FLASH 擦除操作可针对扇区擦除(Sector Erase)或整个闪存擦除 (或称整片擦除，Chip Erase)执行。

注意：擦除操作，不会解除 FLASH 的读/写保护功能。

扇区擦除(Sector Erase)

扇区擦除的具体步骤如下：

1. 检查 FLASH_SR 寄存器中的 BSY 位是否为 0，否则等待上一笔操作完成；
2. 将 FLASH_ECR 寄存器中的 EMODE 位置 0；
3. 将目标扇区号写入 FLASH_ECR[8:0]中；
4. 将 FLASH_CR 寄存器中的 ES 位置 1；
5. 等待 BSY 位清零，擦除动作完成。

注意：扇区擦除的扇区号，不得超过 FLASH 的扇区范围，否则状态寄存器(FLASH_ISR)中的操作错误标志位(OPTEIF)和写保护错误标志位(WPEIF) 将置 1 (如果使能中断，将导致触发错误中断)

闪存擦除(Chip Erase)

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 位是否为 0，否则等待上一笔操作完成；
2. 在 FLASH_ECR 寄存器中，将 EMODE 位置 1；
3. 将 FLASH_CR 寄存器中的 ES 位置 1；
4. 等待 BSY 位清零，擦除动作完成。

5.4.2.3 FLASH 编程

FLASH 的编程顺序如下：

1. 检查 FLASH_SR 寄存器中的 BSY 位是否为 0，否则等待上一笔操作完成；
2. 向 FLASH_DRx(x = 0 … 3) 写入需要编程的 128bit 数据；
3. 向 FLASH_ADDR 中写入编程起始地址；
4. 将 FLASH_CR 寄存器中的 PS 位置 1；
5. 等待 BSY 位清零，编程操作完成。

技巧：把 FLASH 的单元从逻辑“1”写为逻辑“0”时，可以无需执行擦除操作即可进行写操作，这样可以减少 FLASH 擦写次数。但把 FLASH 单元从逻辑“0”写为逻辑“1”时，则必须先执行 FLASH 擦除操作。

注意：

- *如果同时发出擦除和编程操作请求时(ES 和 PS 同时置位)，将认为无效操作，不会执行任何操作，需重新发起对应操作；*
- *当 FLASH_SR 寄存器中的 BSY 位为 1 时，将不能对 FLASH 再次发起其他操作，否则会导致 FLASH 异常，必须等到 BSY 位清零才能发起下一笔操作；*
- *FLASH 写操作位宽必须按照 128 位对齐，若发起非对齐操作将不会执行，并且状态寄存器(FLASH_ISR)中的操作错误标志位(OPTEIF)将置 1，并引发中断；*
- *FLASH 写操作必须处于有效的地址范围内，不能越界。否则操作将不会执行，并且状态寄存器(FLASH_ISR)中的操作错误标志位(OPTEIF)和写保护错误标志位(WPEIF) 将置 1，并引发中断。*

编程与缓存(Cache)

如果 FLASH 编程操作涉及数据缓存(Data Cache)中的某些数据，FLASH 的写操作将修改 FLASH 存储中的数据，同时更新数据缓存中的数据。

如果 FLASH 中的擦除操作也涉及数据或指令缓存(D-Cache/I-Cache)中的数据，则也会同样实现缓存更新操作，将缓存进行初始化，保证与 FLASH 存储数据一致。

5.4.2.4 FLASH ECC 校验

由于 FLASH 存储数据受电子干扰或其他因素导致数据错误，这样会导致系统工作异常。为了提高系统稳定性及芯片可靠性，对 FLASH 加入了 ECC 校验功能，采用 8bit 纠 128bit 的 ECC 策略，可以实现 128bit 纠 1bit。

在对 FLASH 发起 Program 写操作时，ECC 功能模块会自动对 128bit 数据进行 ECC 计数，生成 ECC 校验码，同时将 Program 数据和 ECC 校验码写入 FLASH 中；当进行读操作时，将数据和 ECC 校验码同时读回，进行 ECC 校验，如果存在 1bit 错误会自动将该 bit 纠正，同时产生 ECC 错误(FLASH_ISR 寄存器中的 ECCEIF)，并触发中断(如果使能 EIE 中断)。

注意：内部 ECC 功能只能纠正 1bit 错误，超过 1bit 错误无法纠回。

5.4.2.5 FLASH 加密功能

为了提高芯片的安全特性，加强对用户数据的保护，增加了 FLASH 加密特性，并对指令代码实现 0 等待获取，因而不会对 FLASH 执行代码效率造成影响。

当对 FLASH 发起 Program 写操作时，加密单元会对写入数据执行加密操作，最终将加密后的数据写入 FLASH 存储体；当 CPU 读取 FLASH 时，读返回的数据也会经过加密单元进行解密操作，然后返回到 CPU。

5.4.2.6 FLASH 中断

FLASH 有下述几种情况将会引发中断，FLASH_ISR 寄存器中相应的中断标志位将会置位，通过对相应的位置进行写 1，将清除相应的中断标志位。

表 5-1 引发 FLASH 中断的情况

中断事件	中断使能	中断标志	备注
操作完成	DIE	DIF	编程完成、擦除完成、低功耗操作完成
操作错误	EIE	OPTEIF	编程地址不对齐、扇区擦除编号超过扇区范围、编程地址超过闪存地址范围、编程全 FF 的数据
读保护错误	EIE	RPEIF	读保护模式，并在调试模式下对 FLASH 进行擦除/编程操作；读操作还将同时产生 BusFault 异常中断
写保护错误	EIE	WPEIF	扇区擦除/编程区域命中带读保护的区域、编程地址超过闪存地址范围、扇区擦除编号超过扇区范围
ECC 错误	EIE	ECCEIF	ECC 校验到错误

5.4.2.7 FLASH 读保护(RDP)

对 FLASH 中的用户代码区域实施读保护机制，防止用户代码泄露。读保护分三个级别。

级别 0(RDP Level 0)：无读保护

将 0xAA 写入读保护寄存器 (FLASH_RDPR[7:0]) 时，读保护级别即设为 0。此时，在所有自举配置（FLASH 用户自举、调试或从 RAM 自举）中，均可执行对 FLASH 的读取/编程操作（如果未设置其他保护）。

级别 1(RDP Level 1)：存储器读保护(调试功能受限)

将任意值(分别用于设置级别 0 和级别 2 的 0xAA 和 0xCC 除外) 写入读保护寄存器 (FLASH_RDPR[7:0])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 自举时，则不允许 FLASH 进行访问(读取、擦除、编程)。如果发起读操作请求，将导致总线错误(BusFault 异常)。
- 从 FLASH 自举时(未连接调试器)，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，如果修改读保护寄存器回退到级别 0(RDP 降级)，则将对 FLASH 执行闪存擦除（在读保护降级为级别 0 前，进行擦除动作）。

注意：

- *如果通过代码执行闪存擦除(Chip Erase)操作，将仅擦除闪存区域，读/写保护配置将不受影响，闪存擦除动作之后，依然保持原来的设置。*
- *只有在已激活级别 1 并回退到级别 0 时，才会执行闪存擦除，当提高保护级别时，不会执行擦除。*

级别 2(RDP Level 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
- 调试口(SW)处于禁止状态，RAM 自举也将无法通过调试口载入代码
- 从 FLASH 自举时，允许用户代码对 FLASH 进行读取、擦除、编程操作
- 读保护级别 2 的设置不可逆，无法再降级回到级别 0 或级别 1

不同读保护级别下的访问限制

表 5-2　不同读保护级别下的访问限制

存储区	保护级别	自举=FLASH			自举 != FLASH，或连接调试器		
		读	写	擦除	读	写	擦除
用户 FLASH	级别 1	Y			N		
	级别 2	Y			N[1]		

别 2 的读保护权限下，调试口将被禁止。

不同保护级别之间的转换方案

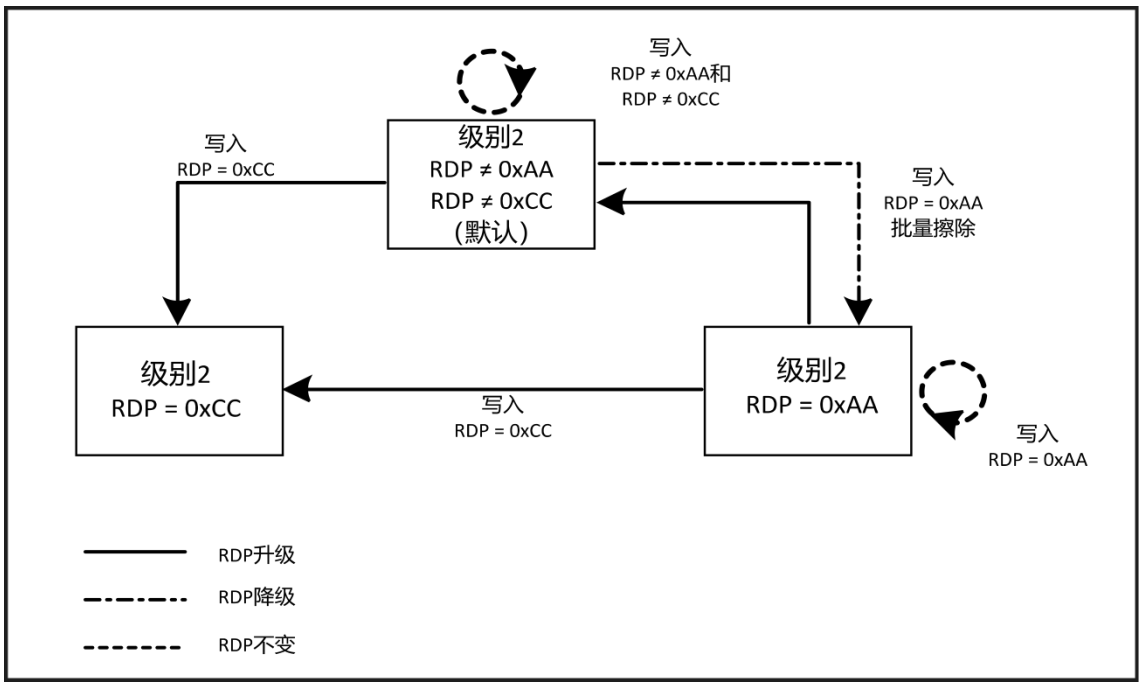


图 5-4　RDP 级别切换

图 5-4 为 RDP 级别切换的过程。

注意：

- *只有级别 1 才能回退到级别 0，并导致闪存擦除(Chip Erase)*
- *级别 2 不能回退到级别 1 或级别 0*

5.4.2.8 FLASH 写保护(WRP)

FLASH Main 区域共 160 个用户扇区都具备写保护功能，用于避免 FLASH 的非易失性代码、数据被意外修改。写保护寄存器（FLASH_WRPR[19 : 0]）中每个 bit 对应 FLASH 8 个扇区的写保护功能，对该寄存器的写保护位的清 0/置 1，可以为相应的扇区执行/去除写保护功能。

受到写保护的扇区，既不能对保护扇区进行扇区擦除、也不能编程。如果对 FLASH 中处于写保护状态的区域执行扇区擦除/编程操作，则 FLASH_ISR 寄存器中的写保护错误标志位(WPEIF)以及操作错误标志位(OPTEIF)将置 1。

注意：

- *如果执行闪存擦除(Chip Erase)，将无视写保护功能，对整个闪存执行擦除。*
- *如果设置了读保护级别 1，并连接了调试器进行调试，即使指定扇区未设置写保护，也无法对相应的区域进行编程/擦除(包括扇区擦除和闪存擦除)操作。*
- *如果当前处于读保护级别 1，并执行读保护级别 1 退回到级别 0，即使设置了写保护，也同样发起闪存擦除操作，将整片 FLASH 擦除。*
- *写保护逻辑在处于读保护级别 2 时，将与级别 0 的一致，区别仅为读保护级别 2 不能通过调试器访问芯片。*

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_ISR 寄存器中的写保护错误标志位 WPEP 将置 1。

如果请求执行擦除操作，则以下情况下 WPE/OEP 位置 1：

- 配置扇区擦除（SER = 1）导致 WPE 置 1
- 请求执行扇区擦除但扇区编号字段无效导致 OEP 置 1
- 请求针对写保护扇区执行扇区擦除
- 当 FLASH 处于读保护状态，但企图发起擦除操作

如果请求执行编程操作，则以下情况下 WPE/OEP 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
- 当 FLASH 处于读保护状态，但企图发起写操作

5.5 寄存器描述

5.5.1 寄存器列表

Name	Offset	Width	Description
FLASH_CR	0x00	32bits	FLASH 控制寄存器
FLASH_LPR	0x04	32bits	FLASH 低功耗寄存器
FLASH_ISR	0x08	32bits	FLASH 中断状态寄存器
EFLASH_SR	0x0C	32bits	FLASH 状态寄存器
FLASH_DR0	0x10	32bits	FLASH 数据寄存器 0
FLASH_DR1	0x14	32bits	FLASH 数据寄存器 1
FLASH_DR2	0x18	32bits	FLASH 数据寄存器 2
FLASH_DR3	0x1C	32bits	FLASH 数据寄存器 3
FLASH_ADDR	0x20	32bits	FLASH 编程地址寄存器
FLASH_ECR	0x28	32bits	FLASH 擦除控制寄存器
FLASH_TR0	0x30	32bits	FLASH Timing 寄存器 0
FLASH_TR1	0x34	32bits	FLASH Timing 寄存器 1
FLASH_TR2	0x38	32bits	FLASH Timing 寄存器 2
FLASH_TR3	0x3C	32bits	FLASH Timing 寄存器 3
FLASH_KEYR	0x50	32bits	FLASH 键寄存器
FLASH_RDPR	0x60	32bits	FLASH 读保护寄存器
FLASH_WRPR	0x70	32bits	FLASH 写保护寄存器
FLASH_UIDx	0x80	32bits	FLASH 用户 IDx 寄存器

5.5.2 寄存器详细描述

5.5.2.1 FLASH 控制寄存器 (FLASH_CR)

- **Name**：FLASH Control Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	LOCK	R/W	0x1	FLASH 编程/擦除锁定标志 (FLASH Program/Erase Lock Flag) 仅写 1 有效，用于对 FLASH 进行加锁。 读取该位： 1：指示当前 FLASH 控制器已加锁，编程/擦除功能被屏蔽。 0：指示当前 FLASH 控制器已解锁，可以进行编程/擦除功能。 注意：是否能够成功进行编程/擦除功能，还将受读/写保护设定限制。
[30-18]	Reserved	Reserved	Reserved	Reserved
[17]	EIE	R/W	0x0	FLASH 错误中断使能控制 (FLASH Error Interrupt Enable) 1：Error 中断使能 0：Error 中断关闭
[16]	DIE	R/W	0x0	FLASH 操作完成中断使能控制 (FLASH Operation Done Interrupt Enable)（默认关闭） 1：Done 中断使能 0：Done 中断关闭
[15-10]	Reserved	Reserved	Reserved	Reserved
[9]	DBPE	R/W	0x1	FLASH D 总线预取使能控制 (FLASH Dbus Prefetch Enable) 1：开启 D 总线预取功能 0：关闭 注意：预取开关使能，当系统主频工作在 40M 以下，关闭预取功能。
[8]	IBPE	R/W	0x1	FLASH I 总线预取使能控制 (FLASH Ibus Prefetch Enable) 1：开启 I 总线预取功能 0：关闭 注意：预取开关使能，当系统主频工作在 40M 以下，关闭预取功能。
[7-2]	Reserved	Reserved	Reserved	Reserved
[1]	ES	R/WAC	0x0	FLASH 擦除开始控制位 (FLASH Erase Start) 1：执行擦写操作 0：无效
[0]	PS	R/WAC	0x0	FLASH 编程开始控制位 (FLASH Program Start) 1：执行编程操作 0：无效

5.5.2.2 FLASH 低功耗寄存器 (FLASH_LPR)

- **Name**：FLASH Lowpower Register
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	LPKEY	R/W	0x0	FLASH 唤醒/待机解锁标志位 (FLASH Wakeup/Standby Unlock Flag) 用于解锁低功耗的控制功能，在配置低功耗 Standby/Wakeup 前，需要先在 LPKEY 位段中写入 0x5003 解锁，否则操作无效。
[15-2]	Reserved	Reserved	Reserved	Reserved
[1]	WKUP	R/WAC	0x0	FLASH 唤醒模式启动位 (FLASH Wakeup Start) 1：启动 Wakeup 0：无效 启动 FLASH Wakeup 操作后，会向 FLASH 发起 Wakeup 命令，退出 Standby 状态。
[0]	STDBY	R/WAC	0x0	FLASH 待机模式启动位 (FLASH Standby Start) 1：启动 Standby 0：无效 启动 FLASH Standby 操作后，会向 FLASH 发起进入 Standby 的命令。

5.5.2.3 FLASH 中断状态寄存器 (FLASH_ISR)

- **Name**：FLASH Interrupt Status Register
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-5]	Reserved	Reserved	Reserved	Reserved
[4]	DIF	R/W1C	0x0	FLASH 操作完成标志 (FLASH Operation Done Pending) 1：Pending 0：No Pending 操作完成标志，详见“FLASH 中断”。
[3]	RPEIF	R/W1C	0x0	FLASH 读保护错误标志 (FLASH Read Protect Error Pending) 1：读保护下非法执行 FLASH 写/擦操作，该位置 1；若执行读操作则会产生 BusFault 中断； 0：No Pending 读保护错误标志，详见“FLASH 中断”。
[2]	ECCEIF	R/W1C	0x0	FLASH ECC 标志 (FLASH ECC Pending) 1：ECC 纠错中发现错误（FLASH 中存在 bit 错误） 0：ECC 正常 ECC 错误标志，详见“FLASH 中断”。
[1]	WPEIF	R/W1C	0x0	FLASH 写保护错误标志 (FLASH Write Protect Error Pending) 1：Pending 0：No Pending 写保护错误标志，详见“FLASH 中断”。
[0]	OPTEIF	R/W1C	0x0	FLASH 操作错误标志 (FLASH Operation Error Pending) 1：Pending 0：No Pending 操作错误标志，详见“FLASH 中断”。

5.5.2.4 FLASH 状态寄存器 (FLASH_SR)

- **Name**：FLASH Status Register
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	BSY	R	0x0	FLASH 忙碌状态 (FLASH Busy Status) FLASH 发起了 Program/Erase/Read 或者其他操作，操作还未完成为 1，当操作完成后会自动清 0 1：Busy 0：Idle

5.5.2.5 FLASH 数据寄存器 0 (FLASH_DR0)

- **Name**：FLASH Data Register0
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DR0	R/W	0x0	FLASH 编程数据 0 (FLASH Program Data0) FLASH 编程数据 128bit 的第一个 Word 注意：128Bit 数据如果是全 F，硬件会屏蔽该写操作，产生操作异常(OPTEIF 置位并触发中断)。

5.5.2.6 FLASH 数据寄存器 1 (FLASH_DR1)

- **Name**：FLASH Data Register1
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DR1	R/W	0x0	FLASH 编程数据 1 (FLASH Program Data1) FLASH 编程数据 128bit 的第二个 Word 注意：128Bit 数据如果是全 F，硬件会屏蔽该写操作，产生操作异常(OPTEIF 置位并触发中断)。

5.5.2.7 FLASH 数据寄存器 2 (FLASH_DR2)

- **Name**：FLASH Data Register2
- **Size**：32bits
- **Offset**：0x18
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DR2	R/W	0x0	Flas 编程数据 2 (FLASH Program Data2) FLASH 编程数据 128bit 的第三个 Word 注意：128Bit 数据如果是全 F，硬件会屏蔽该写操作，产生操作异常(OPTEIF 置位并触发中断)。

5.5.2.8 FLASH 数据寄存器 3 (FLASH_DR3)

- **Name**：FLASH Data Register3
- **Size**：32bits
- **Offset**：0x1C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DR3	R/W	0x0	FLASH 编程数据 3 (FLASH Program Data3) FLASH 编程数据 128bit 的第四个 Word 注意：128Bit 数据如果是全 F，硬件会屏蔽该写操作，产生操作异常(OPTEIF 置位并触发中断)。

5.5.2.9 FLASH 编程地址寄存器 (FLASH_ADDR)

- **Name**：FLASH Program Address Register
- **Size**：32bits
- **Offset**：0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	ADDR	R/W	0x0	FLASH 编程地址 (FLASH Program Address) FLASH 编程地址，地址必须 128bit 对齐，否则将产生操作异常(OPTEIF 置位并触发中断)。 FLASH 存储有效区域范围：[0x00000000-0x00012C00) 共 75KB(共计 160 扇区，每个扇区 512 字节)。

5.5.2.10 FLASH 擦除控制寄存器 (FLASH_ECR)

- **Name**：FLASH Erase Control Register
- **Size**：32bits
- **Offset**：0x28
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	EMODE	R/W	0x0	FLASH 擦除模式 (FLASH Erase Mode) 1：Chip Erase 0：Sector Erase
[30-9]	Reserved	Reserved	Reserved	Reserved
[8-0]	ESNB	R/W	0x0	FLASH 擦除扇区号选择 (FLASH Erase Sector Number select) 仅当 EMODE=0 时有效。 擦除扇区号选择：0 – 159（共计 160 个扇区，每个扇区 512 字节）。

5.5.2.11 FLASH Timing 寄存器 0 (FLASH_TR0)

- **Name**：FLASH Timing Register0
- **Size**：32bits
- **Offset**：0x30
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19-16]	PGH	R/W	0x2	FLASH Program Hold Timing：Min-15ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 2 个 Cycle。
[15-12]	ADS	R/W	0x2	FLASH Addr/Data Setup Timing：Min-15ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 2 个 Cycle。
[11-8]	ADH	R/W	0x2	FLASH Addr/Data Hold Timing：Min-15ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 2 个 Cycle。
[7-4]	OLTCY	R/W	0x9	Operation Latency：Min-100ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 9 个 Cycle。
[3-0]	RC	R/W	0x3	FLASH Read Cycle Time：Min-25ns/30ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 3 个 Cycle。

5.5.2.12 FLASH Timing 寄存器 1 (FLASH_TR1)

- **Name**：FLASH Timing Register1
- **Size**：32bits
- **Offset**：0x34
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-0]	USU	R/W	0x9	FLASH Us Unit Counter: 以 10Mhz 为基准，所以数 1Us 需要 10 个 Cycle；实际使用中，可以根据 FLASH 时钟频率进行调整。

5.5.2.13 FLASH Timing 寄存器 2 (FLASH_TR2)

- **Name**：FLASH Timing Register2
- **Size**：32bits
- **Offset**：0x38
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	CRCV	R/W	0xC9	FLASH Chip Recovery Time：Min-200us 以 1Us 为单位
[23-22]	Reserved	Reserved	Reserved	Reserved
[21-16]	SRCV	R/W	0x33	FLASH Sector Recovery Time：Min-50us 以 1Us 为单位
[15-12]	PRCV	R/W	0x6	FLASH Program Recovery Time：Min-5us 以 1Us 为单位
[11-8]	PROG	R/W	0x7	FLASH Program Byte Time：Min-6us，Max-7.5us 以 1Us 为单位
[7-4]	PGS	R/W	0x6	FLASH Prog2 Setup Time：Min-5us，Max-6.5us 以 1Us 为单位
[3-0]	NVS	R/W	0x7	FLASH PROG/ERASE/CEb/NVR/Address To Web Setup Time：Min-6us 以 1Us 为单位

5.5.2.14 FLASH Timing 寄存器 3 (FLASH_TR3)

- **Name**：FLASH Timing Register3
- **Size**：32bits
- **Offset**：0x3C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-28]	WKPT	R/W	0xB	FLASH Wakeup Time：Min-10us 以 1Us 为单位
[27-12]	CERS	R/W	0xA28	FLASH ChipErase Time：Min-20ms 以 8Us 为单位，计数 5140
[11-0]	SERS	R/W	0x200	FLASH SectorErase Time：Min-4ms，Max-5ms 以 8Us 为单位，计数 512

5.5.2.15 FLASH 键寄存器 (FLASH_KEYR)

- **Name**：FLASH Key Register
- **Size**：32bits
- **Offset**：0x50
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	KEY	R/W	0x0	FLASH 解锁键寄存器 (FLASH Unlock Key Reigster) 用于解锁 FLASH 的擦除/编程控制、读/写保护操作操作。 详请参考“5.4.2.1FLASH 控制寄存器解锁”章节。

5.5.2.16 FLASH 读保护寄存器 (FLASH_RDPR)

- **Name**：FLASH Read Protect Register
- **Size**：32bits
- **Offset**：0x60
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-0]	RDP	R/W	0xFF	FLASH 读保护级别 (FLASH Read Protect Level) FLASH 读保护控制，详细请参考“5.4.2.7FLASH 读保护(RDP)”章节 。 Level0：0xAA，此 Level 下无读保护功能，允许调试接口对 FLASH 的读写及擦除操作； Level1：其他(除了 0xAA 和 0xCC)，此 Level 下有读保护功能，禁止 RAM Boot 调试接口对 FLASH 的读写及擦除操作； Level2：0xCC，此 Level 下有读保护功能，而且禁用调试接口，并且保护级别不能回退。

5.5.2.17 FLASH 写保护寄存器 (FLASH_WRPR)

- **Name**：FLASH Write Protect Register
- **Size**：32bits
- **Offset**：0x70
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	WRP	R/W	0xFFFFFFFF	FLASH 用户写权限控制 (FLASH User Write Permission Control) FLASH 写权限控制，FLASH 一个 Sector 480Byte 空间，每个 bit 对应 FLASH 8 个 Secotr 区域，详情请参考“5.4.2.8 FLASH 写保护(WRP)”章节。 1：存在写权限 0：关闭写权限

5.5.2.18 FLASH 用户 IDx 寄存器 (FLASH_UIDx)

- **Name**：FLASH User IDx Register (x = 0 ... 3)
- **Size**：32bits
- **Offset**：0x80/0x84/0x88/0x8C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	UIDx	R	0xFFFFFFFF	FLASH 用户 ID 寄存器 (FLASH User ID Register) 存储芯片独一无二身份码

6 数据存储 FLASH 接口（DFLASH）

6.1 简介

DFLASH Controller 接口用于管理 CPU 通过 AHB S-Code Bus 对 DFLASH 的访问。可以通过 DFLASH Controller 对 DFLASH 存储执行擦除与编程操作。

6.2 结构框图

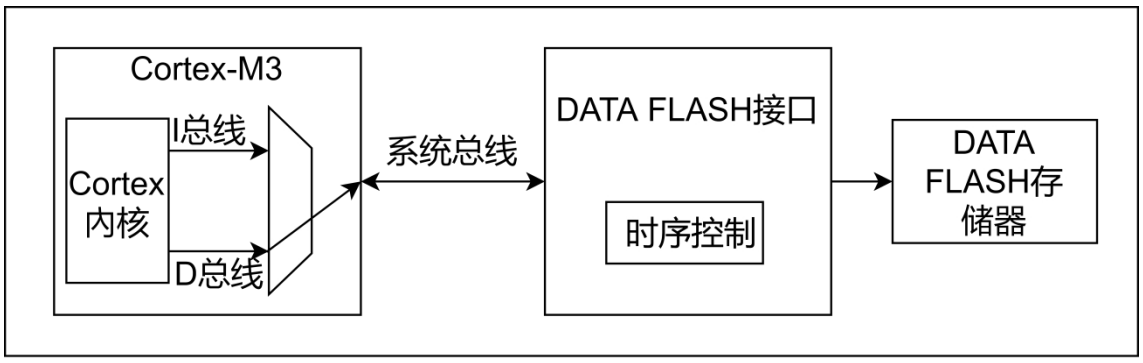


图 6-1 DFLASH 结构框图

6.3 主要特性

DFLASH 主要具有以下特性：

- 高达 18KByte 存储空间
- DFLASH 存储器包括 1 个主存储区(Main Memory)+ 1 个副存储区(Secondary Memory)
  - 主存储区包含 32 个扇区，每个扇区 512Byte，共计 16 KByte
  - 副存储区包含 4 个扇区，每个扇区 512Byte，共计 2KByte
  - 主存储区支持扇区擦除（Sector Erase）和闪存擦除(或称整片擦除，Chip Erase)
  - 副存储区仅支持扇区擦除（Sector Erase），不受闪存擦除影响
- 支持寄存器方式的读取/编程/擦除操作
- 32Bit 总线位宽，支持按字节(Byte)或按字(Word)两种操作模式
- 支持加锁/解锁保护，避免数据误操作
- 支持 Standby/Wakeup 进入/退出低功耗模式

6.4 功能描述

6.4.1 DFLASH 擦除和编程操作

执行任何 DFLASH 编程/擦除操作时，主频 (FCLK)不能低于 1 MHz，建议将主频配置为 80M。如果在 DFLASH 操作期间发生器件复位，将无法保证 DFLASH 中的内容。

在对 DFLASH 执行写入或擦除操作期间，任何读取 DFLASH 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，写入/擦除操作进行期间不能从 DFLASH 中并行获取数据操作。

注意：所有操作均为串行执行，需确保上一笔操作完成后再发起新的操作。完成状态可通过查询 DFLASH 状态寄存器(DFLASH_SR) 中的 BSY 标志位进行判断，等待操作完成后 BSY 标志将自动清 0。BSY 为 1 时不允许发起新的操作，否则无法保证 DFLASH 的工作状态。

6.4.1.1 DFLASH 控制寄存器解锁

复位后，DFLASH 控制寄存器(DFLASH_CR)不允许执行写/擦操作，以防因电气干扰等原因出现对 DFLASH 的意外操作。当需要执行 DFLASH 的写/擦操作时，必须先执行解锁，然后发起相应的操作，DFLASH 写/擦操作解锁顺序如下：

1. 对 DFLASH 密钥寄存器(DFLASH_KEYR)中写入 KEY1 = 0x45670123
2. 对 DFLASH 密钥寄存器(DFLASH_KEYR)中再写入 KEY2 = 0xCDEF89AB

解锁过程必须严格按照上述顺序，解锁成功后，DFLASH_CR 寄存器中的 LOCK 标志位将会自动清 0，之后允许执行写/擦除操作。如解锁错误 LOCK 标志不会清 0，解锁无效。

当解锁完成并执行完对应操作后，也可通过软件将 DFLASH_CR 寄存器中的 LOCK 位置 1，重新对 DFLASH 进行加锁。

DFLASH 写操作支持 Byte 及 Word 位宽操作，不允许非对齐操作，操作中必须按照指定 Size 对齐，否则会提示 Error 错误。

6.4.1.2 DFLASH 擦除

擦除操作分为扇区擦除(Sector Erase)和闪存擦除(或称整片擦除，Chip Erase)：

- DFLASH 的主存储区支持扇区擦除和闪存擦除两种擦除方式
- DFLASH 的副存储区仅支持扇区擦除，不受闪存擦除影响

扇区擦除（Sector Erase）

扇区擦除的具体步骤如下：

1. 检查 DFLASH_SR 寄存器中的 BSY 位是否为 0，否则等待上一笔操作完成；
2. 将 DFLASH_ECR 寄存器中的 EMODE 位置 0；

3. 将目标扇区号写入 DFLASH_ECR[4:0]中；
4. 将 DFLASH_CR 寄存器中的 ES 位置 1；
5. 等待 DFLASH_CR 寄存器中 BSY 位清零，擦除动作完成。

注意：扇区编号必须满足 DFLASH 的扇区范围：主存储区为 0 - 31 扇区号，副存储区为 32 - 35 扇区号。否则操作不执行，并且 DFLASH_ISR 寄存器中的 EIF 错误标志位将置 1，并触发中断(如果使能)。

闪存擦除（Chip Erase）

要执行闪存擦除，建议采用以下步骤：

1. 检查 DFLASH_SR 寄存器中的 BSY 位是否为 0，否则等待上一笔操作完成；
2. 将 DFLASH_ECR 寄存器中的 EMODE 位置 1；
3. 将 DFLASH_CR 寄存器中的 ES 位置 1；
4. 等待 DFLASH_CR 寄存器中 BSY 位清零，擦除动作完成。

6.4.1.3 DFLASH 编程/读取

DFLASH 编程的步骤如下：

1. 检查 DFLASH_SR 寄存器中的 BSY 位是否为 0，否则等待上一笔操作完成；
2. 配置 DFLASH_CR 寄存器内的 PDS 位选择需要编程的数据位宽：8/32 bit；
3. 向 DFLASH_DR 寄存器写入相应长度的数据；
4. 向 DFLASH_ADDR 寄存器写入编程起始地址；
5. 将 DFLASH_CR 寄存器内的 PS 位置 1；
6. 等待 DFLASH_CR 寄存器中 BSY 位清零，擦除动作完成。

技巧：把 DFLASH 的单元从逻辑“1”写为逻辑“0”时，可以无需执行擦除操作即可进行写操作，这样可以减少 DFLASH 擦写次数。但 DFLASH 单元从逻辑“0”写为逻辑“1”时，则必须先执行 DFLASH 擦除操作。

DFLASH 读取数据的步骤如下：

1. 检查 DFLASH_SR 寄存器中的 BSY 位是否为 0，否则等待上一笔操作完成；
2. 配置 DFLASH_CR 寄存器内的 PDS 位选择需要读取的数据位宽：8/32 bit；
3. 向 DFLASH_ADDR 寄存器写入读取起始地址，向 DFLASH_DR 寄存器内写入 0(可选操作，用于避免数据残留)；
4. 将 DFLASH_CR 寄存器内的 RS 位置 1；
5. 等待 DFLASH_CR 寄存器中 BSY 位清零，读取动作完成。
6. 从 DFLASH_DR 寄存器读取数据。

注意：

- *DFLASH_ADDR 内写入的编程起始地址，必须按 DFLASH_CR 的 PDS 位所选择编程数据位宽对齐（例如数据位宽选择 32bit 位宽，则编程起始地址需按 32bit 对齐）。否则该笔非对齐操作将不会被执行，并且 DFLASH_ISR 寄存器中的 EIF 错误标志位将置 1，并触*

发中断(如果使能)。

- *DFLASH_ADDR 内写入的地址也必须满足有效的地址范围内，不得越界。否则操作不执行，并且 DFLASH_ISR 寄存器中的 EIF 错误标志位将置 1，并触发中断(如果使能)。*
- *DFLASH_SR 寄存器中的 BSY 位自动清 0 前，不允许对 DFLASH 发起新的编程操作，否则可能引发未知异常。*
- *不得同时设置擦除/编程/读取操作，否则将判定为无效操作，DFLASH 不执行任何实际动作。*

6.4.1.4 DFLASH 中断

DFLASH 有下述几种情况将会引发中断，DFLASH_ISR 寄存器中相应的中断标志位将会置位，通过对相应的位置进行写 1，将清除相应的中断标志位。

表 6-1 引发 DFLASH 中断的情况

中断事件	中断使能	中断标志	备注
操作完成	DIE	DIF	读取完成、编程完成、擦除完成、低功耗操作完成
操作错误	EIE	EIF	编程地址不对齐、扇区擦除编号超过扇区范围、编程地址超过闪存地址范围、编程全 FF 的数据

6.5 寄存器描述

6.5.1 寄存器列表

Name	Offset	Width	Description
DFLASH_CR	0x00	32bits	DFLASH 控制寄存器
DFLASH_LPR	0x04	32bits	DFLASH 低功耗寄存器
DFLASH_ISR	0x08	32bits	DFLASH 中断状态寄存器
DFLASH_SR	0x0C	32bits	DFLASH 状态寄存器
DFLASH_DR	0x10	32bits	DFLASH 数据寄存器
DFLASH_ADDR	0x14	32bits	DFLASH 地址寄存器
DFLASH_ECR	0x18	32bits	DFLASH 擦除控制寄存器
DFLASH_TR0	0x20	32bits	DFLASH Timing 寄存器 0
DFLASH_TR1	0x24	32bits	DFLASH Timing 寄存器 1
DFLASH_TR2	0x28	32bits	DFLASH Timing 寄存器 2
DFLASH_TR3	0x2C	32bits	DFLASH Timing 寄存器 3
DFLASH_KEYR	0x30	32bits	DFLASH 操作键寄存器

6.5.2 寄存器详细描述

6.5.2.1 DFLASH 控制寄存器 (DFLASH_CR)

- **Name**：DFLASH Control Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	LOCK	R/W	0x1	DFLASH 编程/擦除锁定标志位 (DFLASH Program/Erase Lock Flag) 在编程/擦除之前需要解锁。 仅写 1 有效，用于对 DFLASH 进行加锁。读取该位含义： 1：已加锁，编程/擦除功能被屏蔽。 0：已解锁，可以进行编程/擦除功能。 注意：解锁操作参看“6.4.1.1DFLASH 控制寄存器解锁”章节。
[30-18]	Reserved	Reserved	Reserved	Reserved
[17]	EIE	R/W	0x0	DFLASH 错误中断使能 (DFLASH Error Interrupt Enable) 1：开启操作 Error 中断使能 0：关闭操作 Error 中断使能
[16]	DIE	R/W	0x0	DFLASH 操作完成中断使能 (DFLASH Operation Done Interrupt Enable) 1：开启操作 Done 中断使能 0：关闭操作 Done 中断使能
[15-5]	Reserved	Reserved	Reserved	Reserved
[4]	PDS	R/W	0x0	编程数据位宽选择 (Program DataSize) 1：32Bit 0：8Bit 注意：必须在读取/编程操作启动前配置，中途不允许切换。
[3]	Reserved	Reserved	Reserved	Reserved
[2]	ES	R/WAC	0x0	擦除开始控制位 (Erase Start) 1：开始擦写操作 0：无效
[1]	RS	R/WAC	0x0	读取开始控制位 (Read Start) 1：开始读取操作 0：无效
[0]	PS	R/WAC	0x0	编程开始控制位 (Program Start) 1：开始编程操作 0：无效 写 1 开始编程操作，硬件会自动清 0，写 0 无效。

6.5.2.2 DFLASH 低功耗寄存器 (DFLASH_LPR)

- **Name**：DFLASH Lowpower Register
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	LPKEY	R/W	0x0	DFLASH 唤醒/待机解锁标志位 (DFLASH Wakeup/Standby Unlock Flag) 需要在唤醒和待机之前解锁。 用于解锁低功耗的控制功能，在配置低功耗 Standby/Wakeup 前，需要先在 LPKEY 位段中写入 0x5003 解锁，否则低功耗操作无效。
[15-2]	Reserved	Reserved	Reserved	Reserved
[1]	WKUP	R/WAC	0x0	启动唤醒模式控制位 (Wakeup Start) 1：启动 Wakeup 0：无效 启动 DFLASH Wakeup 操作后，会向 DFLASH 发起 Wakeup 命令，退出 Standby 状态；
[0]	STDBY	R/WAC	0x0	启动待机模式控制位 (Standby Start) 1：启动 Standby 0：无效 启动 DFLASH Standby 操作后，会向 DFLASH 发起进入 Standby 的命令；

6.5.2.3 DFLASH 中断状态寄存器 (DFLASH_ISR)

- **Name**：DFLASH Interrupt Status Register
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1]	DIF	R/W1C	0x0	DFLASH 完成中断标志位 (DFLASH Done Interrupt Flag) 1：Pending 0：No Pending 操作完成中断标志位，详请参考“6.4.1.4DFLASH 中断”章节
[0]	EIF	R/W1C	0x0	操作错误中断标志位 (DFLASH Error Interrupt Flag) 1：Pending 0：No Pending DFLASH 操作错误标志，包括地址不对齐/地址或扇区越界/写全 FF 数据。详请参考“6.4.1.4DFLASH 中断”章节

6.5.2.4 DFLASH 状态寄存器 (DFLASH_SR)

- **Name**：DFLASH Status Register
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	BSY	R	0x0	DFLASH 繁忙状态位 (DFLASH In Busy Status) DFLASH 发起了 Program/Erase/Read 或者其他操作，操作还未完成，当操作完成后会自动清 0 1: 操作忙 0: 空闲

6.5.2.5 DFLASH 数据寄存器 (DFLASH_DR)

- **Name**：DFLASH Data Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DR	R/W	0x0	DFLASH 编程/读取数据寄存器 (DFLASH Program/Read Data Register) DFLASH 编程 32bit 数据，如果 Size 为 8Bit，则取前 8Bit 数据； 需要注意：32Bit 数据如果是全 F，硬件会屏蔽该写操作。

6.5.2.6 DFLASH 地址寄存器 (DFLASH_ADDR)

- **Name**：DFLASH Address Register
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	ADDR	R/W	0x0	DFLASH 编程/读取地址 (DFLASH Program/Read Address)：Byte 地址 DFLASH 编程地址，地址必须根据配置的 Size 对齐； 详请参看“6.4.1.3DFLASH 编程/读取”章节。

6.5.2.7 DFLASH 擦除控制寄存器 (DFLASH_ECR)

- **Name**：DFLASH Erase Control Register
- **Size**：32bits
- **Offset**：0x18
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	EMODE	R/W	0x0	DFLASH 擦除模式选择 (DFLASH Erase Mode) 1：闪存擦除 0：扇区擦除
[30-5]	Reserved	Reserved	Reserved	Reserved
[4-0]	ESNB	R/W	0x0	擦除目标扇区号选择 (DFLASH Erash Sector Number select) 仅当 EMODE=0 时有效。 主存储区扇区号范围（共 32 个扇区）：0 to 31 副存储区扇区号范围（共 4 个扇区）：32 to 35

6.5.2.8 DFLASH Timing 寄存器 0 (DFLASH_TR0)

- **Name**：DFLASH Timing Register0
- **Size**：32bits
- **Offset**：0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19-16]	PGH	R/W	0x2	DFLASH 编程保持时间 (DFLASH Program Hold Timing)：Min-15ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 2 个 Cycle（配置 1）；
[15-12]	ADS	R/W	0x2	DFLASH 地址/数据建立时间 (DFLASH Addr/Data Setup Timing)：Min-15ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 2 个 Cycle；
[11-8]	ADH	R/W	0x2	DFLASH 地址/数据保持时间 (DFLASH Addr/Data Hold Timing)：Min-15ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 2 个 Cycle；
[7-4]	OLTCY	R/W	0x9	操作延迟 (Operation Latency)：Min-100ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 9 个 Cycle；
[3-0]	RC	R/W	0x3	读取周期时间 (Read Cycle Time)：Min-25ns/30ns 以 80Mhz 为基准，1Cycle 大概为 12.5ns，需要 3 个 Cycle；

6.5.2.9 DFLASH Timing 寄存器 1 (DFLASH_TR1)

- **Name**：DFLASH Timing Register1
- **Size**：32bits
- **Offset**：0x24
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-0]	UNIT	R/W	0x9	DFLASH Us Unit Counter: 以 10Mhz 为基准，所以数 1Us 需要 10 个 Cycle；实际使用中，可以根据 DFLASH 时钟频率进行调整。

6.5.2.10 DFLASH Timing 寄存器 2 (DFLASH_TR2)

- **Name**：DFLASH Timing Register2
- **Size**：32bits
- **Offset**：0x28
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	CRCV	R/W	0xC9	DFLASH Chip Recovery Time:Min-200us 以 1Us 为单位
[23-22]	Reserved	Reserved	Reserved	Reserved
[21-16]	SRCV	R/W	0x33	DFLASH Sector Recovery Time:Min-50us 以 1Us 为单位
[15-12]	PRCV	R/W	0x7	DFLASH Program Recovery Time:Min-5us 以 1Us 为单位
[11-8]	PROG	R/W	0x7	DFLASH Program Byte Time:Min-6us，Max-7.5us 以 1Us 为单位
[7-4]	PGS	R/W	0x6	DFLASH Prog2 Setup Time:Min-5us，Max-6.5us 以 1Us 为单位
[3-0]	NVS	R/W	0x7	DFLASH PROG/ERASE/CEb/NVR/Address To Web Setup Time:Min-6us 以 1Us 为单位

6.5.2.11 DFLASH Timing 寄存器 3 (DFLASH_TR3)

- **Name**：DFLASH Timing Register3
- **Size**：32bits
- **Offset**：0x2C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-28]	WKUP	R/W	0xA	DFLASH Wakeup Time:Min-10us 以 1Us 为单位
[27-12]	CERS	R/W	0xA28	DFLASH ChipErase Time:Min-20ms 以 8Us 为单位，计数 5140
[11-0]	SERS	R/W	0x200	DFLASH SectorErase Time:Min-4ms，Max-5ms 以 8Us 为单位，计数 512

6.5.2.12 DFLASH 操作键寄存器 (DFLASH_KEYR)

- **Name**：DFLASH Option Key Register
- **Size**：32bits
- **Offset**：0x30
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	KEY	R/W	0x0	DFLASH Unlock Key Register: 用于解锁 FLASH 的擦除/编程控制、读/写保护操作操作。 详请参考“6.4.1.1DFLASH 控制寄存器解锁”章节

7 电源控制器（PWR）

7.1 电源

芯片的工作电压(AVCC/VCC) 要求介于 3.0 V 到 3.6V 之间，通过内部 LDO（线性调压器）用于提供内部 1.5V 数字电源。

注意：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。

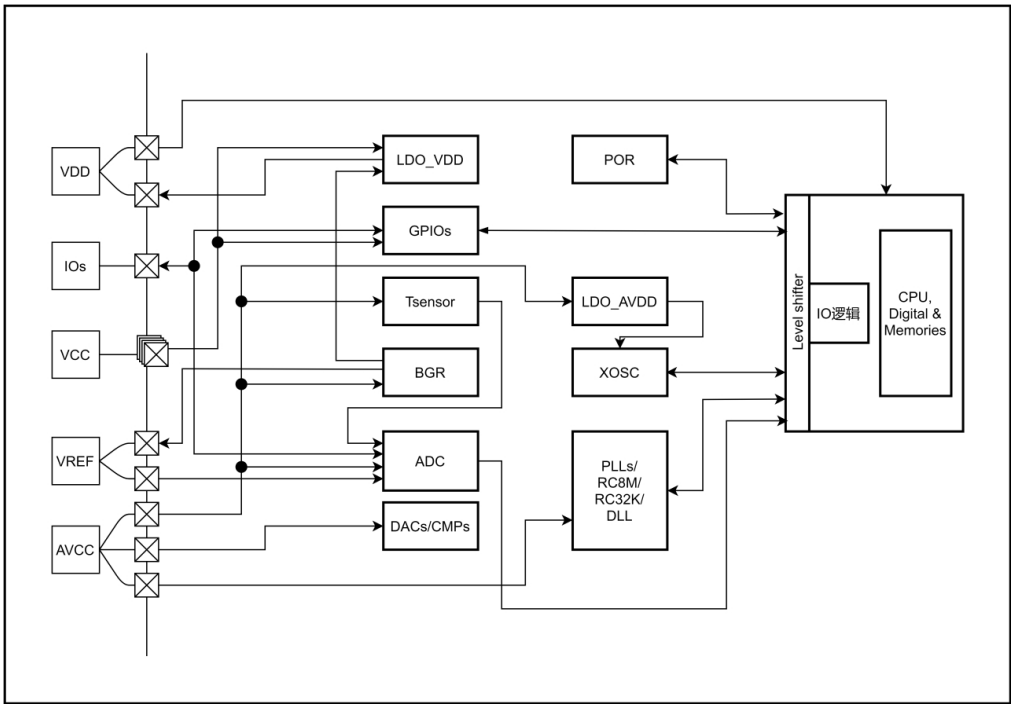


图 7-1 电源框架图

7.1.1 LDO（线性调压器）

嵌入式线性调压器为所有数字电路供电，调压器输出电压约为 1.5V。

此调压器需要将两个外部电容连接到专用引脚 VDD，所有封装都配有 VDD 引脚，具体引脚与封装有关。

调压器在复位后始终处于使能状态，根据应用模式的不同，可选择以下三种不同的工作模式。

- 运行模式，调压器为 1.5 V 域（内核、存储器和数字外设）提供全功率，在此模式下，可将调压器的输出电压通过软件配置调整为 1.5V 的电压值（PMUCR 寄存器的位[15:13]）。
- 待机模式，调压器为 1.5 V 域（内核、存储器和数字外设）提供低功率，可以保留寄存器和内部 SRAM 中的内容。在此模式下，将调压器的输出电压通过软件配置调整为 1.1V（PMUCR 寄存器的位[15:13]）。

7.2 电源监控器

7.2.1 上电复位（POR）/掉电复位（PDR）

本芯片内部集成有 POR/PDR 电路，可以从 2.8V 开始正常工作。

当 VCC/AVCC 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状态。有关上电/掉电复位阈值的相关详细信息，请参考的电气特性部分。

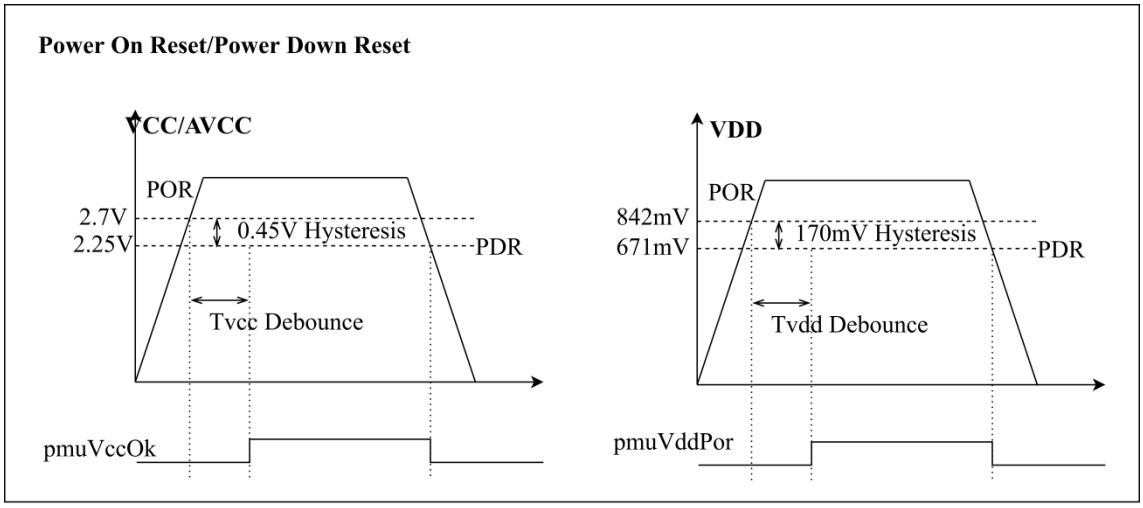


图 7-2 上电复位/掉电复位波形

7.2.2 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。

BOR 默认状态为关闭，可以选择 4 个 V_{BOR} 阈值。

- BOR 级别 0(VBOR0)：1.80 V 到 2.40 V 电压范围的复位阈值级别
- BOR 级别 1(VBOR1)：2.40 V 到 2.55 V 电压范围的复位阈值级别
- BOR 级别 2(VBOR2)：2.55 V 到 2.70 V 电压范围的复位阈值级别
- BOR 级别 3(VBOR3)：2.70 V 到 3.00 V 电压范围的复位阈值级别

当电源电压(VCC)降至所选 V_{BOR} 阈值以下时，将使芯片复位。

BOR 阈值滞回电压约为 400 mV（电源电压的上升沿与下降沿之间）。

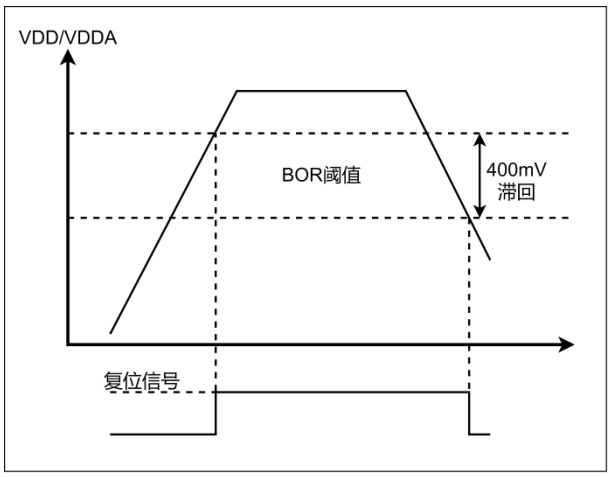


图 7-3 BOR 阈值

7.2.3 可编程电压检测器（PVD）

可以使用 PVD 单元监视 VCC/VDD 的电源电压，对 VCC/VDD 低电压和过流监测，将其与芯片的 PVD 低电压检测模块中的电压阈值控制寄存器(LACR)所选的阈值进行比较。

VCC/VDD 低电压及过流监测功能分别通过设置 PVD 模块中的电压阈值控制寄存器(LACR)的 Enable 来使能 PVD 对应的功能。

PVD 单元中的电源控制/状态寄存器(LCR)中提供了 VCC/VDD 欠压或过流标志，用于指示电源电压是否小于 PVD 阈值或者电源电流是否大于 PVD 阈值。该事件内部连接到 NMI 中断，如果中断使能，则可以产生不可屏蔽中断；该事件内部连接到系统复位，如果复位使能，则可以产生系统复位；该事件内部连接到事件系统，该功能的用处是可以在异常情况下执行紧急关闭的任务。

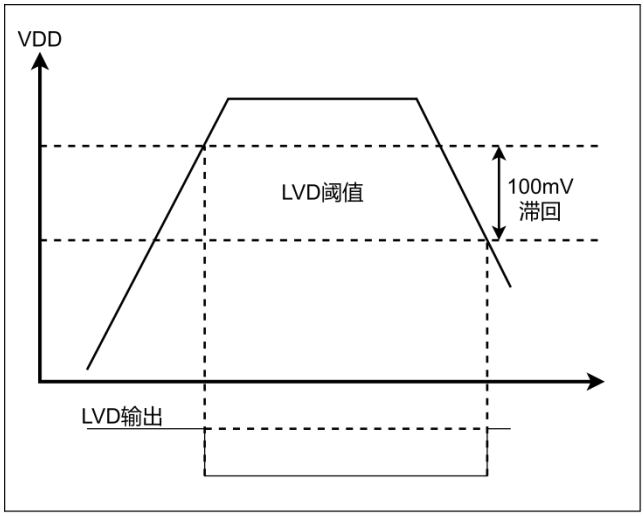


图 7-4 LVD 阈值

7.3 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。

芯片有两个低功耗模式：

- 睡眠模式（Cortex™-M3 内核停止，外设保持运行）
- 停止模式（除 LSI 时钟外，所有时钟都停止）

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 APB 和 AHB 外设时，将对应的外设时钟关闭

表 7-1 低功耗模式汇总

模式名称	进入	唤醒	对 VDD 域时钟的影响	调压器
睡眠（立即休眠或退出时休眠）	WFI	任意中断	CPU CLK 关闭对其它时钟或模拟时钟源无影响	开启
	WFE	唤醒事件		
停止	SLEEPDEEP 位 + WFI 或 WFE	任意中断	除 LSI 时钟外，所有时钟都停止	开启

7.3.1 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、PCLK0 和 PCLK1）速度。
在进入睡眠模式之前，也可以使用这些预分频器来降低外设运行速度。
有关详细信息，请参见第 9.1.2.4 节：System Clock Control Register (SYSCTRL_SYSCLKCR)。

7.3.2 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 HCLK 和 PCLK 以降低功耗。要进一步降低睡眠模式的功耗，可在执行 WFI 或 WFE 指令之前禁止外设时钟。

AHB0 外设时钟门控由 AHB0 外设时钟使能寄存器(SYSCTRL_AHB0CLKGCR)。参见第 9.1.2.11 节：AHB0 ClockGating Register (SYSCTRL_AHB0CLKGCR)。

APB0 外设时钟门控由 APB0 外设时钟使能寄存器(SYSCTRL_APB0CLKGCR)。参见第 9.1.2.9 节：APB0 ClockGating Register (SYSCTRL_APB0CLKGCR)。

APB1 外设时钟门控由 APB1 外设时钟使能寄存器(SYSCTRL_APB1CLKGCR)。参见第 9.1.2.10 节：APB1 ClockGating Register (SYSCTRL_APB1CLKGCR)。

7.3.3 睡眠模式

进入睡眠模式

执行 WFI（等待中断）或 WFE（等待事件）指令即可进入睡眠模式。根据 Cortex™-M3 系统控制寄存器中 SLEEPONEXIT 位的设置，可以通过两种方案选择睡眠模式进入机制：

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

有关如何进入睡眠模式的详细信息，请参见表 7-2 和表 7-3。

退出睡眠模式

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

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

- 在外设的控制寄存器使能一个中断，但不在 NVIC 中使能，同时使能 Cortex™-M3 系统控制寄存器中的 SEVONPEND 位。当 CPU 从 WFE 恢复时，需要清除相应外设的中断挂起位和外设 NVIC 中断通道挂起位（在 NVIC 中断清除挂起寄存器中）。
- 配置一个外部或内部中断为事件模式。当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置位，不必清除相应外设的中断挂起位或 NVIC 中断通道挂起位。

由于没有在进入/退出中断时浪费时间，此模式下的唤醒时间最短。

有关如何退出睡眠模式的详细信息，请参见表 7-2 和表 7-3。

表 7-2 进入和退出立即休眠

立即休眠模式	说明
进入模式	WFI（等待中断）或 WFE（等待事件），且： ➢ SLEEPDEEP = 0 及 ➢ SLEEPONEXIT = 0 请参见 Cortex™-M3 系统控制寄存器。
退出模式	如果使用 WFI 进入： 中断：请参见表 12-1（芯片的中断向量表） 如果使用 WFE 进入： 唤醒事件：请参见第 12.1.4 节：唤醒事件

表 7-3 进入和退出退出时休眠

退出时休眠	说明
进入模式	WFI（等待中断），且： ➢ SLEEPDEEP = 0 及 ➢ SLEEPONEXIT = 1 请参见 Cortex™-M3 系统控制寄存器。
退出模式	中断：请参见表 12-1（芯片的中断向量表）

7.3.4 停止模式

停止模式基于 Cortex™-M3 深度睡眠模式与外设时钟门控。在停止模式下，1.5V 域中除了 32K 外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。

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 7-4。

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。通过用于芯片的电源控制寄存器(PMUCR)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：

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

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 7-4。

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。

在停止模式下，调压器处于低功耗模式下工作，当从停止模式唤醒，可将调压器设置为运行模式。

表7-4 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： ➢ 将Cortex™-M3系统控制寄存器中的SLEEPDEEP位置1 ➢ 将电源控制寄存器(PMUCR)中的PDDS位清零 ➢ 通过配置PMUCR中的LPDS位选择调压器模式。 注意：要进入停止模式之前，所有中断需处于挂起状态，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 进入： 所有配置为中断模式的中断，必须在NVIC中使能对应的中断向量，请参见表12-1：芯片的中断向量表。 如果使用 WFE 进入： 所有配置为事件模式的中断，请参见第12.1.4节：唤醒事件。

调试模式

默认情况下，如果使用调试功能时应用程序将 CPU 置于停止模式，调试连接将中断。这是因为 Cortex™-M3 内核时钟停止工作。)

7.4 PWR 电源控制寄存器

参考 SYSCTRL_PMUCR 寄存器描述。

7.5 PVD 低电压监测寄存器

7.5.1 寄存器列表

Name	Offset	Width	Description
LVDCTRL_LACR	0x00	32bits	LVD 模拟控制寄存器
LVDCTRL_LCR	0x04	32bits	LVD 控制寄存器

7.5.2 寄存器详细描述

7.5.2.1 LVD Analog Control Register (LVDCTRL_LACR)

- **Name**：LVD Analog Contorl Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	VDDOC_ST	R	0x0	VDD Over Current Status: 0：正常 1：低电压 Note：状态位 Pending，该 Bit 位只读，状态有硬件控制；
[30]	VDDLV_ST	R	0x0	VDD Low Voltage Status: 0：正常 1：低电压 Note：状态位 Pending，该 Bit 位只读，状态有硬件控制；
[29]	VCCLV_ST	R	0x0	VCC Low Voltage Status: 0：正常 1：低电压 Note：状态位 Pending，该 Bit 位只读，状态有硬件控制；
[28]	AVCCLV_ST	R	0x0	AVCC Low Voltage Status: 0：正常 1：低电压 Note：状态位 Pending，该 Bit 位只读，状态有硬件控制；

[27]	VDDOC_BYP_EN	R/W	0x0	VDD Over Current Bypass Enable: 0：不使能 1：使能
[26]	VDDLV_BYP_EN	R/W	0x0	VDD Low Voltage Bypass Enable: 0：不使能 1：使能
[25]	VCCLV_BYP_EN	R/W	0x0	VCC Low Voltage Bypass Enable: 0：不使能 1：使能
[24]	AVCCLV_BYP_EN	R/W	0x0	AVCC Low Voltage Bypass Enable: 0：不使能 1：使能
[23-16]	ANAIN_DBC_LIMIT	R/W	0x7	Analog Input Signal Debounce Limit: 0000：1 0001：2 … N：N+1
[15-13]	Reserved	Reserved	Reserved	Reserved
[12-11]	VCCLV_SET	R/W	0x2	VCC Low Voltage Threshold Setting:VCC 低电压阈值设置 00：2.4 01：2.55 10：2.7 11：3
[10-9]	AVCCLV_SET	R/W	0x2	AVCC Low Voltage Threshold Setting:AVCC 低电压阈值设置 00：2.4 01：2.55 10：2.7 11：3
[8-7]	VDDOC_SET	R/W	0x1	VDD 过流检测档位: 00：300mA 01：350mA 10：400mA 11：450mA
[6-4]	VDDLV_SET	R/W	0x7	VDD Low Voltage Threshold Setting:VDD 低电压阈值设置 000：0.8 001：0.9 010：1.0 011：1.1 100：1.2 101：1.3 110：1.35 (eFLASH 低于 1.35V 工作不正常，低电压阈值设为 1.35V) 111：1.4

[3]	VCCLV_EN	R/W	0x1	VCC Low Voltage Detection Enable: 0：不使能 1：使能
[2]	AVCCLV_EN	R/W	0x1	VDD 过流检测使能: 0：不使能 1：使能
[1]	VDDOC_EN	R/W	0x0	VCC Low Voltage Detection Enable: 0：不使能 1：使能
[0]	VDDLV_EN	R/W	0x1	VDD Low Voltage Detection Enable: 0：不使能 1：使能

7.5.2.2 LVD Control Register (LVDCTRL_LCR)

- **Name**：LVD Control Register
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19]	VDDOC_BRK_EN	R/W	0x0	VDD Over Current Breaking Enable: 0：不使能 1：使能
[18]	VDDLV_BRK_EN	R/W	0x0	VDD Low Voltage Breaking Enable: 0：不使能 1：使能
[17]	VCCLV_BRK_EN	R/W	0x0	VCC Low Voltage Breaking Enable: 0：不使能 1：使能
[16]	AVCCLV_BRK_EN	R/W	0x0	AVCC Low Voltage Breaking Enable: 0：不使能 1：使能
[15-8]	Reserved	Reserved	Reserved	Reserved
[7]	VDDOC_INT_EN	R/W	0x0	VDD Over Current Interrupt Enable: 0：不使能 1：使能
[6]	VDDLV_INT_EN	R/W	0x0	VDD Low Voltage Interrupt Enable: 0：不使能 1：使能

[5]	VCCLV_INT_EN	R/W	0x0	VCC Low Voltage Interrupt Enable: 0：不使能 1：使能
[4]	AVCCLV_INT_EN	R/W	0x0	AVCC Low Voltage Interrupt Enable: 0：不使能 1：使能
[3]	VDDOC_RST_EN	R/W	0x0	VDD Over Current Reset Enable: 0：不使能 1：使能
[2]	VDDLV_RST_EN	R/W	0x0	VDD Low Voltage Reset Enable: 0：不使能 1：使能
[1]	VCCLV_RST_EN	R/W	0x0	VCC Low Voltage Reset Enable: 0：不使能 1：使能
[0]	VCCLV_RST_EN	R/W	0x0	AVCC Low Voltage Reset Enable: 0：不使能 1：使能

8 复位和时钟控制（RCU）

8.1 复位

共有二种类型的复位，分别为系统复位、电源复位。

8.1.1 系统复位

除了 SYSCTRL 模块中复位标志位、FLASH 自检参数外，系统复位将复位所有寄存器至复位状态，当发生以下任一事件时，将产生一个系统复位：

- NRST 引脚上的低电平(外部复位)
- 独立看门狗计数完成(IWDG 复位)
- 窗口看门狗计数完成(WWDG 复位)
- 软件复位

可通过查看 SYSCTRL 内部 SRSTSR 状态寄存器中的复位状态标志位识别复位事件来源。若要对芯片进行软件复位，必须将 Cortex™-M3 应用中断和复位控制寄存器中的 SYSRESETREQ 位置 1。有关详细信息，请参见 Cortex™-M3 技术参考手册。

8.1.1.1 引脚复位

芯片具有一个独立的复位引脚 NRST，默认上拉，拉低该引脚将引起系统复位。

8.1.1.2 独立看门狗复位

详见独立看门狗模块介绍。

8.1.1.3 窗口看门狗复位

详见窗口看门狗模块介绍。

8.1.1.4 软件复位

通过将 Cortex™-M3 内核的“应用中断与复位控制寄存器”中的 SYSRESETREQ 位置 1，可实现软件复位（内核和外设）；该寄存器中的另一个控制位 VECTRESET 则只复位 Cortex™-M3 内核，不复位外设。

8.1.2 电源复位

以下事件之一发生时，将产生电源复位：

- 上电复位（POR 复位）
- 低电压复位（LVD 复位）

电源复位将复位所有寄存器。

8.1.2.1 上电复位

芯片内部有一个完整的上电复位（POR）电路，当供电电压达到 V_{POR} 时系统即能正常工作。当 VDD 低于指定的限位电压 V_{POR} 时，系统保持为复位状态。

8.1.2.2 低电压复位

低电压复位是一种强制性保护复位，用户可以通过配置使能位开启，当供电电压低于检测阈值时， CPU 将产生复位。

8.2 时钟

可以使用四种不同的时钟源来驱动系统时钟(SYSCLK)：

- LSI 振荡器时钟
- HSE 振荡器时钟
- PLL 时钟
- HSI 振荡器时钟

器件具有以下时钟源：

- 32KHz 低速内部 RC 振荡器(LSI)，该 RC 用于驱动独立看门狗

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。

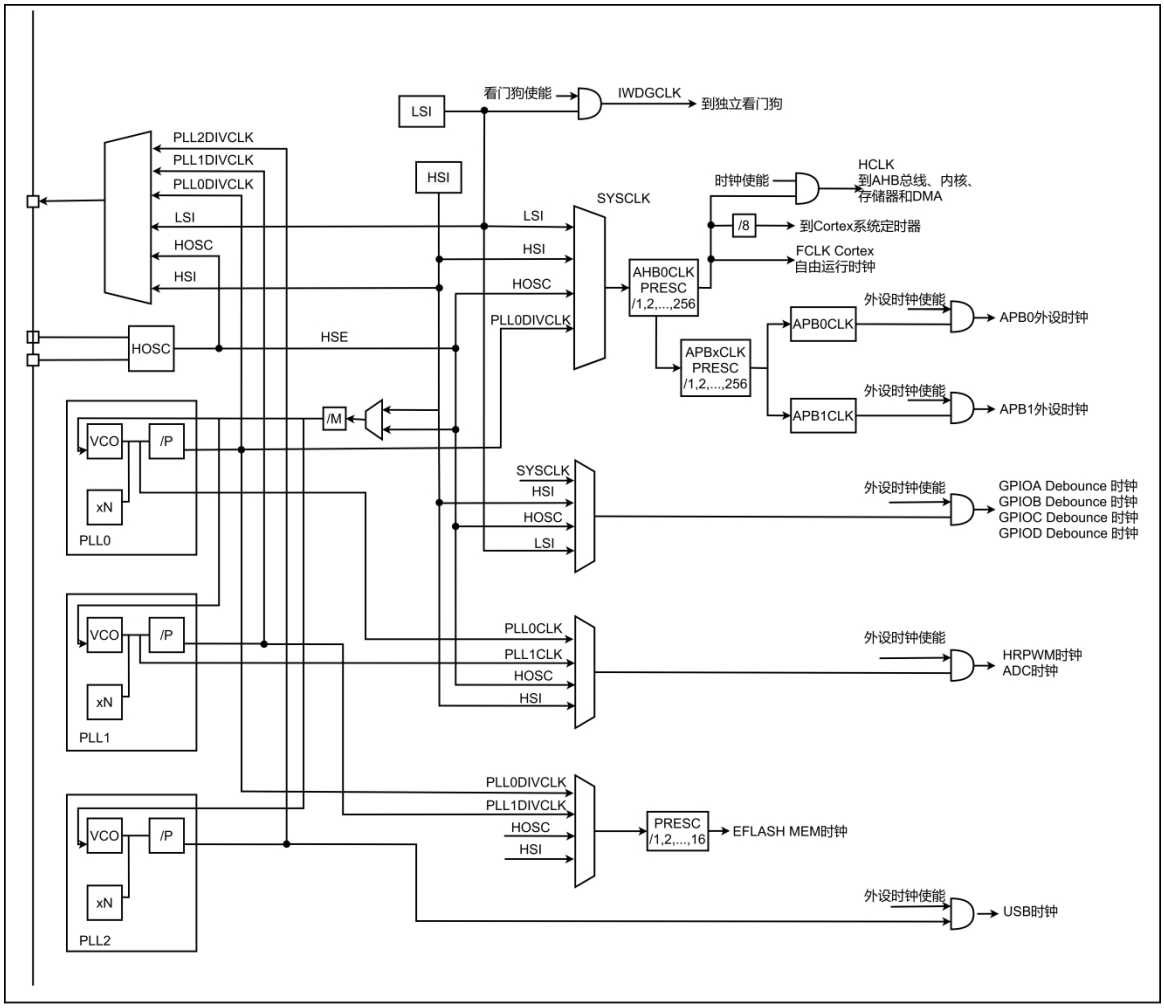


图 8-1　时钟电路简图

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，既可采用最高的时钟频率，也可为 USB 以及 HRPWM、ADC 和 GPIO 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(APB0)频率和低速 APB(APB1)频率。AHB 域及 APB0/1 的最高时钟频率均为 90MHz。

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 PLL2 输出的 USB 时钟(60MHz)
- 来自于特定 PLL1 输出的 ADC 及 HRPWM 时钟(160MHz)

RCU 送出一个 8 分频的 AHB 时钟(HCLK)到 Cortex 系统定时器(SysTick)，SysTick 可使用此时钟作为时钟源，也可使用 HCLK 作为时钟源，具体可在 SysTick 控制和状态寄存器中配置。

FCLK 充当 Cortex™-M3 的自由运行时钟。有关详细信息，请参见 Cortex™-M3 技术参考手册。

8.2.1 HSE 晶振

外部晶振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。

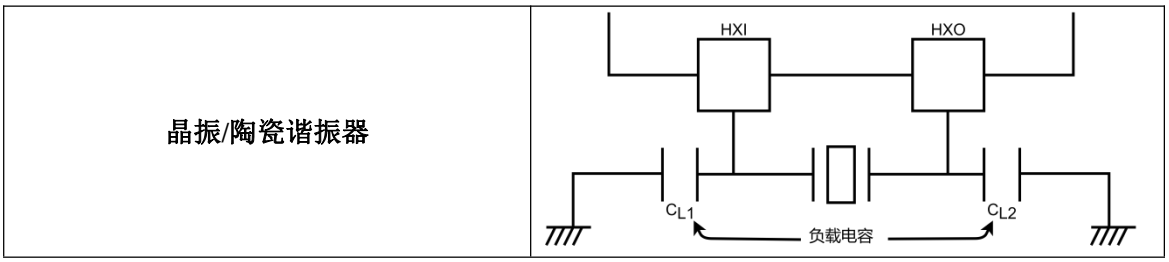


图 8-2 HSE 时钟源

外部晶振/陶瓷谐振器

HSE 的特点是精度非常高。

相关的硬件配置如图 8-2 所示。

在芯片启动时，上电复位默认晶振起振使能位置 1，起振电路可以使用。

晶振起振电路可通过 SYSCFG 控制寄存器(XOSCCR)中的 XOSCEN 位打开或关闭。

8.2.2 HSI 振荡器

HSI 时钟信号由内部 8 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。

HSI RC 可通过 SYSCFG 控制寄存器(XOSCCR)中的 HSIEN 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 8.2.6 节：时钟安全系统(CSS)。

8.2.3 PLL 设置

芯片内部具有三个 PLL：

- 主 PLL (PLL0) 由 HSE 或 HSI 振荡器提供时钟信号，并具有三个不同的输出时钟：
  - 第一个输出用于生成高速系统时钟（最高达 90MHz）
  - 第二个输出用于生成 EFLASH 工作时钟
- 专用 PLL (PLL1) 用于生成高速时钟 160M，用于 ADC 及 HRPWM 模块；
- 专用 PLL (PLL2) 用于生成精准时钟，用于 USB 接口传输。

由于在 PLL 使能后主 PLL 配置参数便不可更改，所以建议先对 PLL 进行配置，然后再使能，首先确定选择 HSI 或 HSE 振荡器作为 PLL 时钟源，并配置分频系数。

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，可自由配置选择切换方案：第一种可以直接切换系统时钟至内部 HSI；第二种可以切换 PLL 的参考时钟至 HSI。

8.2.4 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)和自动唤醒单元使用，时钟频率在 32 kHz 左右，LSI RC 低频振荡器不可关闭。

8.2.5 系统时钟（SYSCLK）选择

在系统复位后，默认系统时钟为 LSI，若需要切换时钟，需要在目标时钟源已准备就绪（时钟在启动或 PLL 锁相后稳定），才可从一个时钟源切换到另一个。

8.2.6 时钟安全系统（CSS）

为了增强系统的可靠性，防止因时钟失效造成系统出错甚至死机的严重后果， 芯片内部增加了一个时钟安全监控（CSS）模块。 CSM 监控用于监测 HSE 的振荡情况，包含晶振、陶振、或外部时钟输入的情况，一旦发生停振或振荡异常（频率低于正常值或高于正常值），则发送异常标志，该异常标志可作为中断输入或 PWM 异常事件信号输入。

用户可通过配置 SYSCTRL 中 XOSCMCR 寄存器来使能 CSS 模块，当发生异常时，用户可以选择系统时钟或 PLL 参考时钟是否自动切换内部 HSI。

注意：一旦 CSS 中断产生，引起 NMI 被不断执行，直到 CSS 中断挂起位被清除。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 的参考时钟，并且该 PLL 时钟为系统时钟）并且检出故障，用户可选择将系统时钟切换到 HSI 振荡器。

如果 HSE 振荡器时钟是充当系统时钟的 PLL 的参考时钟源，则在发生故障时，PLL 也会被禁止，用户可选择将 PLL 的参考时钟切换到 HSI 振荡器，由于内部 HIS 时钟偏差会导致 PLL 时钟输出出现稍微偏差。

8.2.7 看门狗时钟

如果独立看门狗(IWDG)通过软件设置的方式启动，则 LSI 振荡器将为独立看门狗(IWDG)提供运行计数时钟，并且 LSI 振荡器不可关闭。

8.3 RCU 寄存器

参考 SYSCTRL_ PLL0CR ~ SYSCTRL_XOSCMCR 寄存器描述。

9 系统配置控制器（SYSCTRL）

系统配置控制器主要用于管理系统时钟、复位及调试相关功能，具体参考下节寄存器描述。

9.1 寄存器描述

9.1.1 寄存器列表

Name	Offset	Width	Description
SYSCTRL_PLL0CR	0x00	32bits	PLL0 Control Register
SYSCTRL_PLL1CR	0x04	32bits	PLL1 Control Register
SYSCTRL_PLL2CR	0x08	32bits	PLL2 Control Register
SYSCTRL_ SYSCLKCR	0x10	32bits	System Clock Control Register
SYSCTRL_ BUSCLKCR	0x14	32bits	SystemBus Clock Control Register
SYSCTRL_ CLKSRCR	0x18	32bits	Clock Source Register
SYSCTRL_ CLKDIVR0	0x20	32bits	Clock Divider Register0
SYSCTRL_ CLKDIVR2	0x28	32bits	Clock Divider Register2
SYSCTRL_ APB0CLKGCR	0x30	32bits	APB0 ClockGating Register
SYSCTRL_ APB1CLKGCR	0x34	32bits	APB1 ClockGating Register
SYSCTRL_ AHB0CLKGCR	0x38	32bits	AHB0 ClockGating Register
SYSCTRL_ FUNCLKGCR	0x3C	32bits	Function ClockGating Register
SYSCTRL_ SRSTCR	0x40	32bits	System SoftReset Register
SYSCTRL_ APB0RSTCR	0x44	32bits	APB0 SoftReset Register
SYSCTRL_ APB1RSTCR	0x48	32bits	APB1 Reset Control Register
SYSCTRL_ AHBRSTCR	0x4C	32bits	AHB Reset Control Register
SYSCTRL_ XOSCCR	0x60	32bits	XOSC Control Register
SYSCTRL_ XOSCMCR	0x64	32bits	XOSC Monitor Control Register
SYSCTRL_ATODCR	0x68	32bits	Analog Tout Control Regiter
SYSCTRL_SYSCFGCR	0x6C	32bits	System Config Control Register
SYSCTRL_SRSTSR	0x70	32bits	System Reset Status Regiter
SYSCTRL_SYSDBGR	0x74	32bits	System Debug Control Register
SYSCTRL_LOCKKEY	0x80	32bits	System LockKey Register
SYSCTRL_PMUCR	0x100	32bits	PMU Control Register

9.1.2 寄存器详细描述

9.1.2.1 PLL0 Control Register (PLL0CR)

- **Name**：PLL0 Control Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	PLL_ENABLE	R/W	0x0	PLL 使能控制: 0：关闭 1：启动
[30]	PLL_LOCKED	R	0x0	PLL 锁定标志位: 0：未锁定 1：锁定
[29]	PLL_OPA	R/W	0x1	PLL VI 转换器输入控制: 0：PMOS 对输入 1：轨对轨输入
[28]	PLL_LPF	R/W	0x0	PLL 模拟输入参考时钟频率: 1：26M 0：8M
[27-24]	PLL_LOCKUP	R/W	0x3	PLL LockUp Limit:Lock 比较值 可配值：0-15； Note：实际延迟大小为 pllLockUp*2；
[23]	Reserved	Reserved	Reserved	Reserved
[22-21]	PLL_BAND	R/W	0x1	PLL 输出中心频率配置(VCO 输出频率，二分频前): 0：312M 1：396M 2：466M 3：520M
[20]	PLL_VCODET	R/W	0x0	PLL VCO 测试电压使能控制: 0：关闭 1：启动
[19-18]	PLL_GVCO	R/W	0x2	PLL VCO 频率调节增益控制: 00：最小值 11：最大值
[17-16]	PLL_GCP	R/W	0x3	PLL Charge Pump 电流控制: 00：1uA 01：2uA 10：3uA 11：4uA
[15-8]	Reserved	Reserved	Reserved	Reserved

[7-4]	PLL_DIV	R/W	0x1	PLL 时钟分频设置: 0：无效 1：2 分频 … N：N+1 分频 Note：PLL 后除频最小为 2 分频，配置 0 无效，保持前一个分频值；
[3]	PLL_PREDIV	R/W	0x1	PLL 输入参考时钟前除频控制: 0：2 分频 1：不分频
[2]	Reserved	Reserved	Reserved	Reserved
[1-0]	PLL_REFCLK	R/W	0x0	PLL 参考时钟源选择: 00：XOSC 01：HSI 10：关闭 11：DFT 时钟

9.1.2.2 PLL1 Control Register (PLL1CR)

- **Name**：PLL1 Control Register
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	PLL_ENABLE	R/W	0x0	PLL 使能控制: 0：关闭 1：启动
[30]	PLL_LOCKED	R	0x0	PLL 锁定标志位: 0：未锁定 1：锁定
[29]	PLL_OPA	R/W	0x1	PLL VI 转换器输入控制: 0：PMOS 对输入 1：轨对轨输入
[28]	PLL_LPF	R/W	0x0	PLL 模拟输入参考时钟频率: 1：26M 0：8M
[27-24]	PLL_LOCKUP	R/W	0x3	PLL LockUp Limit:Lock 比较值 可配值：0-15； Note：实际延迟大小为 pllLockUp*2；
[23]	Reserved	Reserved	Reserved	Reserved
[22-21]	PLL_BAND	R/W	0x1	PLL 输出中心频率配置(VCO 输出频率，二分频前): 0：312M 1：396M 2：466M 3：520M
[20]	PLL_VCODET	R/W	0x0	PLL VCO 测试电压使能控制: 0：关闭 1：启动
[19-18]	PLL_GVCO	R/W	0x2	PLL VCO 频率调节增益控制: 00：最小值 11：最大值
[17-16]	PLL_GCP	R/W	0x3	PLL Charge Pump 电流控制: 00：1uA 01：2uA 10：3uA 11：4uA
[15-8]	Reserved	Reserved	Reserved	Reserved

[7-4]	PLL_DIV	R/W	0x1	PLL 时钟分频设置: 0：无效 1：2 分频 … N：N+1 分频 Note：PLL 后除频最小为 2 分频，配置 0 无效，保持前一个分频值；
[3]	PLL_PREDIV	R/W	0x1	PLL 输入参考时钟前除频控制: 0：2 分频 1：不分频
[2]	Reserved	Reserved	Reserved	Reserved
[1-0]	PLL_REFCLK	R/W	0x0	PLL 参考时钟源选择: 00：XOSC 01：HSI 10：关闭 11：DFT 时钟

9.1.2.3 PLL2 Control Register (PLL2CR)

- **Name**：PLL2 Control Register
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	PLL_ENABLE	R/W	0x0	PLL 使能控制: 0：关闭 1：启动
[30]	PLL_LOCKED	R	0x0	PLL 锁定标志位: 0：未锁定 1：锁定
[29]	PLL_OPA	R/W	0x1	PLL VI 转换器输入控制: 0：PMOS 对输入 1：轨对轨输入
[28]	PLL_LPF	R/W	0x0	PLL 模拟输入参考时钟频率: 1：26M 0：8M
[27-24]	PLL_LOCKUP	R/W	0x3	PLL LockUp Limit:Lock 比较值 可配值：0-15； Note：实际延迟大小为 pllLockUp*2；
[23]	Reserved	Reserved	Reserved	Reserved
[22-21]	PLL_BAND	R/W	0x1	PLL 输出中心频率配置(VCO 输出频率，二分频前): 0：312M 1：396M 2：466M 3：520M
[20]	PLL_VCODET	R/W	0x0	PLL VCO 测试电压使能控制: 0：关闭 1：启动
[19-18]	PLL_GVCO	R/W	0x2	PLL VCO 频率调节增益控制: 00：最小值 11：最大值
[17-16]	PLL_GCP	R/W	0x3	PLL Charge Pump 电流控制: 00：1uA 01：2uA 10：3uA 11：4uA
[15-8]	Reserved	Reserved	Reserved	Reserved

[7-4]	PLL_DIV	R/W	0x1	PLL 时钟分频设置: 0：无效 1：2 分频 … N：N+1 分频 Note：PLL 后除频最小为 2 分频，配置 0 无效，保持前一个分频值；
[3]	PLL_PREDIV	R/W	0x1	PLL 输入参考时钟前除频控制: 0：2 分频 1：不分频
[2]	Reserved	Reserved	Reserved	Reserved
[1-0]	PLL_REFCLK	R/W	0x0	PLL 参考时钟源选择: 00：XOSC 01：HSI 10：关闭 11：DFT 时钟

9.1.2.4 System Clock Control Register (SYSCLKCR)

- **Name**：System Clock Control Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	Reserved	Reserved	Reserved	Reserved
[23-16]	SYSCLK_DIV	R/W	0x0	SYSCLK 时钟分频设置: 0：1 分频(不分频) 1：2 分频 … N：N+1 分频
[15-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	SYSCLK_SRC	R/W	0x0	SYSCLK 时钟源选择: 00：LSI 01：HSE 10：PLL0 N 分频 11：HSI

9.1.2.5 SystemBus Clock Control Register (BUSCLKCR)

- **Name**：SystemBus Clock Control Register
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-19]	Reserved	Reserved	Reserved	Reserved
[18]	APB1CLK_EN	R/W	0x1	APB1 总线时钟使能: 0：关闭 1：启动
[17]	APB0CLK_EN	R/W	0x1	APB0 总线时钟使能: 0：关闭 1：启动
[16]	AHB0CLK_EN	R/W	0x1	AHB0 总线时钟使能: 0：关闭 1：启动
[15-8]	APB1CLK_DIV	R/W	0x0	APB1CLK 时钟分频设置(时钟源为 AHB0CLK): 0：1 分频(不分频) 1：2 分频 … N：N+1 分频
[7-0]	APB0CLK_DIV	R/W	0x0	APB0CLK 时钟分频设置(时钟源为 AHB0CLK): 0：1 分频(不分频) 1：2 分频 … N：N+1 分频

9.1.2.6 Clock Source Register (CLKSRCR)

- **Name**：Clock Source Register
- **Size**：32bits
- **Offset**：0x18
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	Reserved	Reserved	Reserved	Reserved
[23-22]	GPIOD_DBCCLK_SRC	R/W	0x0	GPIOD Debouce CLK 时钟源: 00：HSI 01：XOSC 10：SYSCLK 11：LSI
[21-20]	GPIOC_DBCCLK_SRC	R/W	0x0	GPIOC Debouce CLK 时钟源: 00：HSI 01：XOSC 10：SYSCLK 11：LSI
[19-18]	GPIOB_DBCCLK_SRC	R/W	0x0	GPIOB Debouce CLK 时钟源: 00：HSI 01：XOSC 10：SYSCLK 11：LSI
[17-16]	GPIOA_DBCCLK_SRC	R/W	0x0	GPIOA Debouce CLK 时钟源: 00：HSI 01：XOSC 10：SYSCLK 11：LSI
[15-9]	Reserved	Reserved	Reserved	Reserved
[8-7]	EEPROM_MEMCLK_SRC	R/W	0x0	EEPROM Memory CLK 时钟源: 00：HSI 01：PLL0DivCLk 10：PLL1DivCLk 11：PLL2DivCLk
[6-5]	EFLASH_MEMCLK_SRC	R/W	0x0	EFLASH Memory CLK 时钟源: 00：HSI 01：PLL0DivCLk 10：PLL1DivCLk 11：PLL2DivCLk
[4]	Reserved	Reserved	Reserved	Reserved

[3-2]	ADC_FUNCLK_SRC	R/W	0x0	ADC Function CLK 时钟源: 00：HSI 01：HSE 10：PLL0 11：PLL1
[1-0]	HRPWM_FUNCLK_SRC	R/W	0x0	HRPWM Function CLK 时钟源: 00：HSI 01：HSE 10：PLL0 11：PLL1

9.1.2.7 Clock Divider Register0 (CLKDIVR0)

- **Name**：Clock Divider Register0
- **Size**：32bits
- **Offset**：0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11-8]	EEPROM_MEMCLK_DIV	R/W	0x0	EEPROM Memory CLK 时钟分频: 0：1 分频(不分频) 1：2 分频 … N：N+1 分频
[7-4]	EFLASH_MEMCLK_DIV	R/W	0x0	EFLASH Memory CLK 时钟分频: 0：1 分频(不分频) 1：2 分频 … N：N+1 分频
[3-2]	ADC_FUNCLK_DIV	R/W	0x0	ADC Function CLK 时钟分频: 0：1 分频(不分频) 1：2 分频 … N：N+1 分频
[1-0]	HRPWM_FUNCLK_DIV	R/W	0x0	HRPWM Function CLK 时钟分频: 0：1 分频(不分频) 1：2 分频 … N：N+1 分频

9.1.2.8 Clock Divider Register2 (CLKDIVR2)

- **Name**：Clock Divider Register2
- **Size**：32bits
- **Offset**：0x28
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	GPIOD_DBCCLK_DIV	Reserved	Reserved	GPIOD Debounce CLK 时钟分频:(Debouce Clock 分频)0：1 分频(不分频) 1：2 分频 … N：N+1 分频
[23-16]	GPIOC_DBCCLK_DIV	R/W	0x0	GPIOC Debounce CLK 时钟分频:(Debouce Clock 分频)0：1 分频(不分频) 1：2 分频 … N：N+1 分频
[15-8]	GPIOB_DBCCLK_DIV	R/W	0x0	GPIOB Debounce CLK 时钟分频:(Debouce Clock 分频)0：1 分频(不分频) 1：2 分频 … N：N+1 分频
[7-0]	GPIOA_DBCCLK_DIV	R/W	0x0	GPIOA Debounce CLK 时钟分频:(Debouce Clock 分频)0：1 分频(不分频) 1：2 分频 … N：N+1 分频

9.1.2.9 APB0 ClockGating Register (APB0CLKGCR)

- **Name**：APB0 ClockGating Register
- **Size**：32bits
- **Offset**：0x30
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12]	TIMER_BUSCLK_EN	R/W	0x0	TIMER BusCLK 时钟使能: 0：关闭 1：启动
[11-6]	Reserved	Reserved	Reserved	Reserved
[5]	UART1_BUSCLK_EN	R/W	0x0	UART1 BusCLK 时钟使能: 0：关闭 1：启动
[4]	UART0_BUSCLK_EN	R/W	0x0	UART0 BusCLK 时钟使能: 0：关闭 1：启动
[3-2]	Reserved	Reserved	Reserved	Reserved
[1]	I2C1_BUSCLK_EN	R/W	0x0	I2C1 BusCLK 时钟使能: 0：关闭 1：启动
[0]	I2C0_BUSCLK_EN	R/W	0x0	I2C0 BusCLK 时钟使能: 0：关闭 1：启动

9.1.2.10 APB1 ClockGating Register (APB1CLKGCR)

- **Name**：APB1 ClockGating Register
- **Size**：32bits
- **Offset**：0x34
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-15]	Reserved	Reserved	Reserved	Reserved
[14]	ECU_BUSCLK_EN	R/W	0x0	ECU BusCLK 时钟使能: 0：关闭 1：启动
[13]	Reserved	Reserved	Reserved	Reserved
[12]	IIR4_BUSCLK_EN	R/W	0x0	IIR4 BusCLK 时钟使能: 0：关闭 1：启动
[11]	IIR3_BUSCLK_EN	R/W	0x0	IIR3 BusCLK 时钟使能: 0：关闭 1：启动
[10]	IIR2_BUSCLK_EN	R/W	0x0	IIR2 BusCLK 时钟使能: 0：关闭 1：启动
[9]	IIR1_BUSCLK_EN	R/W	0x0	IIR1 BusCLK 时钟使能: 0：关闭 1：启动
[8]	IIR0_BUSCLK_EN	R/W	0x0	IIR0 BusCLK 时钟使能: 0：关闭 1：启动
[7-0]	Reserved	Reserved	Reserved	Reserved

9.1.2.11 AHB0 ClockGating Register (AHB0CLKGCR)

- **Name**：AHB0 ClockGating Register
- **Size**：32bits
- **Offset**：0x38
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19]	RAM2CLK_EN	R/W	0x1	RAM2CLK 时钟使能: 0：关闭 1：启动
[18]	RAM1CLK_EN	R/W	0x1	RAM1CLK 时钟使能: 0：关闭 1：启动
[17]	RAM0CLK_EN	R/W	0x1	RAM0CLK 时钟使能: 0：关闭 1：启动
[16-15]	Reserved	Reserved	Reserved	Reserved
[14]	USB_BUSCLK_EN	R/W	0x0	USB BusCLK 时钟使能: 0：关闭 1：启动
[13]	EEPROM_BUSCLK_EN	R/W	0x0	EEPROM BusCLK 时钟使能(总线时钟): 0：关闭 1：启动
[12]	EFLASH_BUSCLK_EN	R/W	0x1	EFLASH BusCLK 时钟使能(总线时钟): 0：关闭 1：启动
[11]	Reserved	Reserved	Reserved	Reserved
[10]	HRPWM_BUSCLK_EN	R/W	0x0	HRPWM BusCLK 时钟使能: 0：关闭 1：启动
[9]	ADC_BUSCLK_EN	R/W	0x0	ADC BusCLK 时钟使能: 0：关闭 1：启动
[8]	DAC_BUSCLK_EN	R/W	0x0	DAC BusCLK 时钟使能: 0：关闭 1：启动
[7]	CMP_BUSCLK_EN	R/W	0x0	CMP BusCLK 时钟使能: 0：关闭 1：启动
[6]	GPIOD_BUSCLK_EN	R/W	0x1	GPIOD BusCLK 时钟使能: 0：关闭 1：启动

[5]	GPIOC_BUSCLK_EN	R/W	0x1	GPIOC BusCLK 时钟使能: 0：关闭 1：启动
[4]	GPIOB_BUSCLK_EN	R/W	0x1	GPIOB BusCLK 时钟使能: 0：关闭 1：启动
[3]	GPIOA_BUSCLK_EN	R/W	0x1	GPIOA BusCLK 时钟使能: 0：关闭 1：启动
[2]	HSTIMER_BUSCLK_EN	R/W	0x0	HSTIMER BusCLk 时钟使能: 0：关闭 1：启动
[1]	CAN_BUSCLK_EN	R/W	0x0	CAN BusCLK 时钟使能: 0：关闭 1：启动
[0]	DMA_BUSCLK_EN	R/W	0x0	DMA BusCLK 时钟使能: 0：关闭 1：启动

9.1.2.12 Function ClockGating Register (FUNCLKGCR)

- **Name**：Function ClockGating Register
- **Size**：32bits
- **Offset**：0x3C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	HRPWM_FUNCLK_EN	R/W	0x0	HRPWM Function CLK 时钟使能: 0：关闭 1：启动
[10]	ADC_FUNCLK_EN	R/W	0x0	ADC Function CLK 时钟使能: 0：关闭 1：启动
[9]	CAN_FUNCLK_EN	R/W	0x0	CAN Function CLK 时钟使能: 0：关闭 1：启动
[8]	ECU_FUNCLK_EN	R/W	0x0	ECU Function CLK 时钟使能: 0：关闭 1：启动
[7]	IIR4_FUNCLK_EN	R/W	0x0	IIR4 Function CLK 时钟使能: 0：关闭 1：启动

[6]	IIR3_FUNCLK_EN	R/W	0x0	IIR3 Function CLK 时钟使能: 0：关闭 1：启动
[5]	IIR2_FUNCLK_EN	R/W	0x0	IIR2 Function CLK 时钟使能: 0：关闭 1：启动
[4]	IIR1_FUNCLK_EN	R/W	0x0	IIR1 Function CLK 时钟使能: 0：关闭 1：启动
[3]	IIR0_FUNCLK_EN	R/W	0x0	IIR1 Function CLK 时钟使能: 0：关闭 1：启动
[2]	USB_FUNCLK_EN	R/W	0x0	USB Function CLK 时钟使能: 0：关闭 1：启动
[1]	EEPROM_MEMCLK_EN	R/W	0x0	EEPROM Memory CLK 时钟使能: 0：关闭 1：启动
[0]	EFLASH_MEMCLK_EN	R/W	0x1	EFLASH Memory CLK 时钟使能: 0：关闭 1：启动

9.1.2.13 System SoftReset Register (SRSTCR)

- **Name**：System SoftReset Register
- **Size**：32bits
- **Offset**：0x40
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	GPDDBC_SOFTRST	R/W	0x1	GPDDBC 软复位控制(GPIOD Debouce): 0：复位 1：释放
[10]	GPCDBC_SOFTRST	R/W	0x1	GPCDBC 软复位控制(GPIOD Debouce): 0：复位 1：释放
[9]	GPBDBC_SOFTRST	R/W	0x1	GPBDBC 软复位控制(GPIOD Debouce): 0：复位 1：释放
[8]	GPADBC_SOFTRST	R/W	0x1	GPADBC 软复位控制(GPIOD Debouce): 0：复位 1：释放
[7-4]	Reserved	Reserved	Reserved	Reserved
[3]	APB1BUS_SOFTRST	R/W	0x1	APB1BUS 软复位控制: 0：复位 1：释放
[2]	APB0BUS_SOFTRST	R/W	0x1	APB0BUS 软复位控制: 0：复位 1：释放
[1]	Reserved	Reserved	Reserved	Reserved
[0]	AHB0BUS_SOFTRST	R/W	0x1	AHB0BUS 软复位控制: 0：复位 1：释放

9.1.2.14 APB0 SoftReset Register (APB0RSTCR)

- **Name**：APB0 SoftReset Register
- **Size**：32bits
- **Offset**：0x44
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12]	TIMER_SOFTRST	R/W	0x1	TIMER 软复位控制: 0：复位 1：释放
[11-6]	Reserved	Reserved	Reserved	Reserved
[5]	UART1_SOFTRST	R/W	0x1	URAT1 软复位控制: 0：复位 1：释放
[4]	UART0_SOFTRST	R/W	0x1	URAT0 软复位控制: 0：复位 1：释放
[3-2]	Reserved	Reserved	Reserved	Reserved
[1]	I2C1_SOFTRST	R/W	0x1	I2C1 软复位控制: 0：复位 1：释放
[0]	I2C0_SOFTRST	R/W	0x1	I2C0 软复位控制: 0：复位 1：释放

9.1.2.15 APB1 Reset Control Register (APB1RSTCR)

- **Name**：APB1 Reset Control Register
- **Size**：32bits
- **Offset**：0x48
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-19]	Reserved	Reserved	Reserved	Reserved
[18]	ECU_SOFTRST	R/W	0x1	ECU 软复位控制: 0：复位 1：释放
[17]	Reserved	Reserved	Reserved	Reserved
[16]	IIR4_SOFTRST	R/W	0x1	IIR4 软复位控制: 0：复位 1：释放
[15]	IIR3_SOFTRST	R/W	0x1	IIR3 软复位控制: 0：复位 1：释放
[14]	IIR2_SOFTRST	R/W	0x1	IIR2 软复位控制: 0：复位 1：释放
[13]	IIR1_SOFTRST	R/W	0x1	IIR1 软复位控制: 0：复位 1：释放
[12]	IIR0_SOFTRST	R/W	0x1	IIR0 软复位控制: 0：复位 1：释放
[11-8]	Reserved	Reserved	Reserved	Reserved
[7]	FPLL2_SOFTRST	R/W	0x1	FPLL2RST 软复位控制: 0：复位 1：释放
[6]	FPLL1_SOFTRST	R/W	0x1	FPLL1RST 软复位控制: 0：复位 1：释放
[5]	FPLL0_SOFTRST	R/W	0x1	FPLL0RST 软复位控制: 0：复位 1：释放
[4-0]	Reserved	Reserved	Reserved	Reserved

9.1.2.16 AHB Reset Control Register (AHBRSTCR)

- **Name**：AHB Reset Control Register
- **Size**：32bits
- **Offset**：0x4C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-14]	Reserved	Reserved	Reserved	Reserved
[13]	EEPROM_SOFTRST	R/W	0x1	EEPROM 软复位控制: 0：复位 1：释放
[12]	HSTIMER_SOFTRST	R/W	0x1	HSTIMER 软复位控制: 0：复位 1：释放
[11]	GPIOD_SOFTRST	R/W	0x1	GPIOD 软复位控制: 0：复位 1：释放
[10]	GPIOC_SOFTRST	R/W	0x1	GPIOC 软复位控制: 0：复位 1：释放
[9]	GPIOB_SOFTRST	R/W	0x1	GPIOB 软复位控制: 0：复位 1：释放
[8]	GPIOA_SOFTRST	R/W	0x1	GPIOA 软复位控制: 0：复位 1：释放
[7]	USB_SOFTRST	R/W	0x1	USB 软复位控制: 0：复位 1：释放
[6]	HRPWM_SOFTRST	R/W	0x1	HRPWM 软复位控制: 0：复位 1：释放
[5]	DAC_SOFTRST	R/W	0x1	DAC 软复位控制: 0：复位 1：释放
[4]	ADC_SOFTRST	R/W	0x1	ADC 软复位控制: 0：复位 1：释放
[3]	CMP_SOFTRST	R/W	0x1	CMP 软复位控制: 0：复位 1：释放

[2]	EFLASH_SOFTRST	R/W	0x1	EFLASH 软复位控制: 0：复位 1：释放
[1]	CAN_SOFTRST	R/W	0x1	CAN 软复位控制: 0：复位 1：释放
[0]	DMA_SOFTRST	R/W	0x1	DMA 软复位控制: 0：复位 1：释放

9.1.2.17 XOSC Control Register (XOSCCR)

- **Name**：XOSC Control Register
- **Size**：32bits
- **Offset**：0x60
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-21]	Reserved	Reserved	Reserved	Reserved
[20]	HSI_EN	R/W	0x1	HSI 使能控制: 0：关闭 1：启动
[19-17]	Reserved	Reserved	Reserved	Reserved
[16]	XOSCLOSS_IRQEN	R/W	0x1	XOSC AutoSwitch 中断使能控制: 0：关闭 1：启动
[15]	XOSC_HYEN	R/W	0x1	XOSC 比较器迟滞控制: 0：不发生迟滞 1：迟滞 ±10%
[14-12]	XOSC_DR	R/W	0x5	XOSC 启动电流配置: 000：0.25mA 001：0.375mA 010：0.5mA 011：0.625mA 100：0.75mA 101：0.875mA 110：1.0mA 111：1.125mA
[11-8]	XOSC_CTO	R/W	0x0	XOSC HXO 谐振电容微调: 电容值=CTO*0.2pF； 最小值=0pF； 最大值=3pF；
[7-4]	XOSC_CTI	R/W	0x0	XOSC HXI 谐振电容微调: 电容值=CTI*0.2pF 最小值=0pF； 最大值=3pF；

[3-1]	XOSC_CS	R/W	0x5	XOSC 谐振电容粗调: 000：0pF 001：2.5pF 010：5pF 011：7.5pF 100：10pF 101：12.5pF 110：15pF; 111：17.5pF; 每个装置都有一个额外的 3.5pF 寄生电容;
[0]	XOSC_EN	R/W	0x1	XOSC 使能控制: 0：关闭 1：启动

9.1.2.18 XOSC Monitor Control Register (XOSCMCR)

- **Name**：XOSC Monitor Control Register
- **Size**：32bits
- **Offset**：0x64
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-29]	Reserved	Reserved	Reserved	Reserved
[28]	XOSC_LOSS_PENDING	R/W1C	0x0	XOSC Loss Pending 状态位: 晶振异常时产生 Pending 状态，只读软件写 1 清 0。
[27]	Reserved	Reserved	Reserved	Reserved
[26]	XOSC_SYSCLK_SWEN	R/W	0x0	XOSC 异常触发，系统时钟硬切换的使能控制: 0：关闭 1：启动 Note：使能后会自动硬件自动切换到 HSI;
[25]	XOSC_REFCLK_SWEN	R/W	0x1	XOSC 异常触发，PLL 参考时钟硬切换的使能控制: 0：关闭 1：启动 Note；使能后会自动硬件自动切换到 HSI;
[24]	XOSC_MNTEN	R/W	0x0	XOSC AutoSwitch 使能控制: 0：关闭 1：启动
[23-20]	XOSC_WIDTH	R/W	0x0	XOSC AutoSwitch 窗口宽度调整(单位 1us): 0：1us x3 = 3us 1：2us x3 = 6us …
[19-10]	XOSC_HIGH_LIMIT	R/W	0x1e	XOSC AutoSwitch 功能上限比较值:默认值 30; 将内部 HSI 按照最小 10M 计算:3000/(1000/10)=30; 实际计数与下限值类型，举例如下： 如果 HSI 频率实测为 11M，需要配 11x3=33;
[9-0]	XOSC_LOW_LIMIT	R/W	0x12	XOSC AutoSwitch 功能下限比较值:默认值 18; AutoSwitch 窗口值默认为 3us(窗口可调、最大 48us)，内部 HSI 按照最小 6M 计算:3000/(1000/6)=18; 实际 IC 中可以根据实测的 HSI 频率进行准确的计算，即按照 Wcycles/Thsi，举例如下： 如果 HSI 频率实测为 7.5M，需要配 7.5x3=22;

9.1.2.19 Analog Tout Control Register (SYSCTRL_ATODCR)

- **Name**：Analog Tout Control Regiter
- **Size**：32bits
- **Offset**：0x68
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-10]	ADCBUF_SRCSEL	R/W	0x0	ADC Buffer Source Selection: 000：TSEN；001：PLL0_CPOUT； 010：PLL1_CPOUT； 011：PLL2_CPOUT；1xx：ADC1_IN[10]；
[9]	ADCBUF_BYPASS	R/W	0x0	ADC Buffer Bypass Enable: 0：关闭 1：启动
[8]	ADCBUF_EN	R/W	0x0	ADC Buffer Enable: 0：关闭 1：启动
[7-5]	Reserved	Reserved	Reserved	Reserved
[4-0]	TOUT_SRC	R/W	0x0	Anlog CP Test TOUT 控制:(详细见 TOUT 真值表) 0x00-；0x01-AVSS；0x02-TSENSOR；0x03-AVDD； 0x04-VDDRC；0x05-DLL_CPOUT；0x06-DLL_CKO； 0x08-PLLA_ICO；0x09-PLLA_VFBL；0x0A-PLLA_CPOUT；

9.1.2.20 System Config Control Register (SYSCTRL_SYSCFGCR)

- **Name**：System Config Control Register
- **Size**：32bits
- **Offset**：0x6C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-28]	Reserved	Reserved	Reserved	Reserved
[27-26]	ADCCTRL_FANOUT_EN	R/W	0x0	ADC Ctrl Fanin/out 使能控制: 00：关闭 01：Fanout Enable 10：Fanin Enable 11：Reserved ADC 控制信号 Fanout 与 Fanin，共 9Bit 信号，分别为 Str、PhiF1、PhiR、PhiH、PhiS、Chsel，分别对应 GPIOC6 至 GPIOC14(参考 Pinmux)；
[25]	ADCDATA_FANOUT_SRC	R/W	0x0	ADC Data/Ctrl Fanout Srouce 选择: 0：ADC0 1：ADC1 ADC0 与 1 选择位，该 Bit 同时控制 Data 及 Ctrl 信号的源选择；
[24]	ADCDATA_FANOUT_EN	R/W	0x0	ADC Data Fanout 使能控制: 0：关闭 1：启动 ADC Analog Data Fanout 输出，具体参考 Pinmux；
[23-22]	Reserved	Reserved	Reserved	Reserved
[21]	I2C1_SMBUS_OE	R/W	0x0	I2C1 SMBUS 功能输出使能控制 0：关闭 1：启动
[20]	I2C0_SMBUS_OE	R/W	0x0	I2C0 SMBUS 功能输出使能控制 0：关闭 1：启动
[19]	Reserved	Reserved	Reserved	Reserved
[18]	JTAG_BUGFIX_EN	R/W	0x0	JTAG BUGFIX 功能使能控制 0：关闭 1：启动
[17]	CANFD_EN	R/W	0x0	CAN FD 功能使能: 0：关闭 1：启动

[16]	CPU_LOCKUPRST_EN	R/W	0x0	CPU LOCKUP RESET 使能控制:(CPU 出现 LOCKUP 复位) 0：关闭 1：启动
[15]	WWDG_DEBUG_EN	R/W	0x1	WWDG Debug Enable 使能控制: CPU 进入 Debug 调试模式，是否自动关闭 WDG； 0：关闭 1：启动
[14]	WWDG_TIMEOUTRST_EN	R/W	0x1	WWDG TimeOut Reset 使能控制(TimeOut 产生复位) 0：关闭 1：启动
[13]	IWDG_DEBUG_EN	R/W	0x1	IWDG Debug Enable 使能控制: CPU 进入 Debug 调试模式，是否自动关闭 WDG； 0：关闭 1：启动
[12]	IWDG_TIMEOUTRST_EN	R/W	0x1	IWDG TimeOut Reset 使能控制(TimeOut 产生复位) 0：关闭 1：启动
[11]	Reserved	Reserved	Reserved	Reserved
[10]	TMR1DBGEN	R/W	0x0	TMR1 DEBUG 使能控制 0：关闭 1：启动
[9]	TMR0DBGEN	R/W	0x0	TMR0 DEBUG 使能控制 0：关闭 1：启动
[8]	GPIO_NMIEN	R/W	0x0	GPIO Input NMI 中断使能控制: 0：关闭 1：启动
[7-4]	CLK_TEST_SRC	R/W	0x0	内部时钟源 FanOut 选择: 0000：LSI 0001：HSI 0010：XOSC 0100：PLL0DivClk 0101：PLL0/16 1000：PLL1DivClk 1001：PLL1/16 1100：PLL2DivClk 1101：PLL2/16 xxxx：Reserved
[3]	CLK_FANOUT_EN	R/W	0x0	内部时钟源 FanOut 输出使能:(具体时钟源见上) 0：关闭 1：启动

[2-1]	PMU_DEBUG_EN	R/W	0x0	PMU 测试信号输出使能控制(分别对应 DEBUG0 与 1): 0：关闭 1：启动 Note:Bit1 对应 DEBUG0，Bit2 对应 DEBUG1；
[0]	TEST_CLKIN_EN	R/W	0x0	TEST(ATE) CLK INPUT 使能控制: 0：关闭 1：启动

9.1.2.21 System Reset Status Register (SYSCTRL_SRSTSR)

- **Name**：System Reset Status Register
- **Size**：32bits
- **Offset**：0x70
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-5]	Reserved	Reserved	Reserved	Reserved
[4]	SYSREQ_RST_ST	R/W1C	0x0	SYSRESET REQ Reset 标志位(软件复位): 1：REQ 产生 Reset 0：未产生复位
[3]	MCLR_RST_ST	R/W1C	0x0	MCLR Reset 标志位(外部按键复位): 1：按键产生 Reset 0：未产生复位
[2]	LVD_RST_ST	R/W1C	0x0	LowPower Reset 标志位: 1：LowPower 产生 Reset 0：未产生复位
[1]	WWDG_RST_ST	R/W1C	0x0	wWdg Reset 标志位: 1：wWdg 产生 Reset 0：未产生复位
[0]	IWDG_RST_ST	R/W1C	0x0	iWdg Reset 标志位: 1：iWdg 产生 Reset 0：未产生复位

9.1.2.22 System Debug Register (SYSCTRL_SDBGR)

- **Name**：System Debug Register
- **Size**：32bits
- **Offset**：0x74
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	Reserved	Reserved	Reserved	Reserved
[30-28]	DBG_EN	R/W	0x0	System Debug Fanout Enable 使能位: 000：Disable 001：TIMER Debug Enable 010：ADDA Debug Enable 011：HRPWM Debug Enable 100：USB Debug Enable
[27-20]	Reserved	Reserved	Reserved	Reserved
[19-15]	USB_DBG_SRC	R/W	0x0	USB Debug Fanout Source 源选择
[14-10]	HRPWM_DBG_SRC	R/W	0x0	HRPWM Debug Fanout Source 源选择: 00000：HRPWM-DACRstTrg0 …… Note：详细参考附件；
[9-5]	ADDA_DBG_SRC	R/W	0x0	ADDA Debug Fanout Source 源选择: ADDA Fanout Bit 分配为高 1Bit 选择 ADDA0-1，低 4Bit 选择 ADDA 内部 Event 输出，举例如下: 00000：ADDA0-WdgOut0
[4-0]	TMR_DBG_SRC	R/W	0x0	Timer Debug Fanout Source 源选择: Timer Fanout Bit 分配为高 3Bit 选择 Timer0-7，低 2Bit 选择 Timer 内部 Trigger 分类，举例如下: 00000：Timer0-TrigO

9.1.2.23 Sysctrl Lock Key Register (SYSCTRL_LOCKKEY)

- **Name**：Sysctrl Lock Key Register
- **Size**：32bits
- **Offset**：0x80
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	SYSCTL_LOCKKEY	R/W	0x0	系统寄存器写操作的密码保护配置: 1、SysCtrl 内部控制寄存器保护 Key 为 0x3fac87e4，写寄存器之前必须写 Key 进行解锁； 2、SysCtrl 内部 FLS 寄存器保护 Key 为 0x1f2e3c4a，写寄存器之前必须写 Key 进行解锁；

9.1.2.24 PMU Control Register (PMUCR)

- **Name**：PMU Control Register
- **Size**：32bits
- **Offset**：0x100
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	Reserved	Reserved	Reserved	Reserved
[30-25]	PMU_IN	R/W	0x0	保留输入
[24-19]	CUR_RES	R/W	0x20	电阻值选择: 000000：13.6K …… 100000：20K …… 111111：26.2K 注意:电阻可调步长 200ohm，可调范围-32%~31%，此控制位可以从 eFLASH 中获取校准或从寄存器中获取；
[18-17]	CUR_CAL	R/W	0x0	精确电流产生电路校准使能控制: 00：关闭 01：将 60uA 精密电流源镜像到 CUR_RES3 端口 10:电阻输出端口可以分别输出 CUR_RES1 和 CUR_RES2 以校准电阻，在此模式下需要关闭精准电流产生电路； 11：非法状态
[16]	AVDD_DRD	R/W	0x1	AVDD 的无电容 LDO 下拉配置选择: 0：不发生下拉 1：0.5mA 下拉 启动时，如果打开下拉，可以帮助稳定和启动电路，当 26M 晶振开始工作时关闭下拉；

[15-14]	AVDD_SET	R/W	0x2	AVDD 电压控制， 默认为 1.5V: 00：1.3V 01：1.4V 10：1.5V 11：1.6V
[13-10]	VDD_SET	R/W	0x8	VDD(校准器)电压粗调控制，默认为 1.5V； 0000：1.1V 0001：1.15V 0010：1.2V 0011：1.25V 0100：1.3V 0101：1.35V 0110：1.4V 0111：1.45V 1000：1.5V 1001：1.55V 1010：1.6V 1011：1.65V 1100：1.7V 1101：1.75V 1110：1.8 1111：1.85V
[9]	CUR_ENABLE	R/W	0x1	精准电流产生电路的使能控制 0：关闭 1：启动 Note：当使用 8M RC FADC SARADC 时，精准电流产生电路一定要启动
[8]	AVDDLDO_ENABLE	R/W	0x1	AVDDLDO 使能控制 0：关闭 1：启动
[7]	TEMPSENSOR_ENABLE	R/W	0x1	温度传感器使能控制: 0：关闭 1：启动

[6-2]	BGR_VOL	R/W	0x10	带隙电压选择控制，默认为 1.5V; 00000：1.26V 00001：…… 01000：…… 01101：1.4775 01110：1.485V 01111：1.4925 10000：1.5V 10001：1.5075 10010：1.515 11111：1.7325 注意：为了调整带隙电压，最小步长可以设为 7.5mV, 此位可以由 eFLASH 或寄存器校准;
[1]	BGR_DRD	R/W	0x1	带隙下拉寄存器控制: 0：无下拉电阻 1：有下拉电阻 注意：默认情况下，总有一个下拉电阻对应约 0.5mA。 打开电源后，可以禁用软件以节省电量。上电成功后此位受寄存器控制，并且成功;
[0]	BGR_FILTER	R/W	0x0	带隙滤波器控制: 0：带隙滤波器无滤波电阻 1：带隙滤波器有滤波电阻 Note：在使用 DAC/ADC 之前要确保将此位置 1，否则会产生噪声。成功上电后此位由寄存器控制，并且电路会被钳制在默认值;

10 通用 I/O（GPIO）

10.1 简介

GPIO 是一个可编程通用 I/O 外设（programmable General Purpose Programming I/O），共包含高达 4 组 GPIO（GPIOA~D），GPIOA~C 共有 16 个 I/O，GPIOD 共有 1 个 I/O。

10.2 结构框图

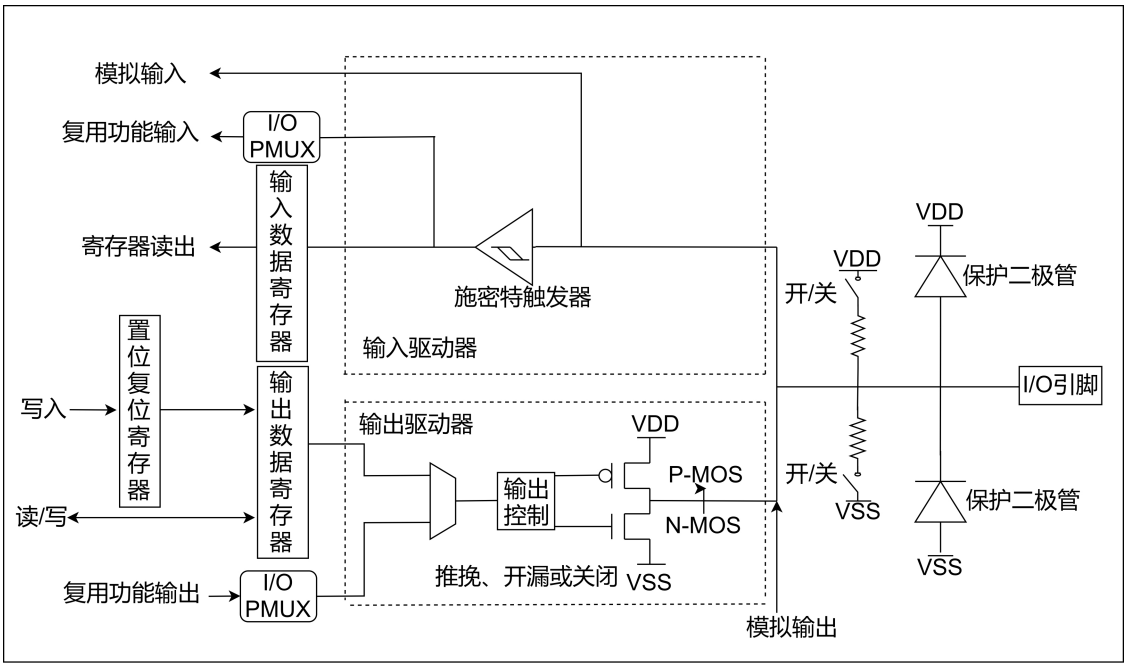


图 10-1 GPIO 结构框图

10.3 主要特性

GPIO 主要具有以下特性：

- 受控 I/O 高达 49 个
- 输出状态：推挽或开漏 + 上拉/下拉
- 输入状态：浮空、上拉/下拉、模拟
- 输出数据寄存器或外设(复用功能输出)输出数据
- 数据输入到输入数据寄存器或外设(复用功能输入)
- 模拟功能
- 置位和复位寄存器，对输出数据寄存器具有按位写权限
- 可选的同步功能和消抖功能
- 可配的电流驱动能力
- 每个 I/O 均可独立配置输出压摆率(Slew Rate)和输入迟滞(Hysteresis)
- 复用功能输入/输出选择寄存器
- 复用功能高灵活性，允许将 I/O 引脚配置为 GPIO 或各种外设功能
- 所有 I/O 均可独立配置外部输入触发中断使能，上升沿/下降沿/双边沿触发，独立的中断挂起标志

10.4 功能描述

通过软件可将 GPIO 各个端口分别配置为以下多种模式：

- 输入浮空
- 输入上拉
- 输入下拉
- 模拟功能
- 具有上拉或下拉的开漏输出
- 具有上拉或下拉的推挽输出
- 具有上拉或下拉的复用功能推挽
- 具有上拉或下拉的复用功能开漏

每个 I/O 端口的模式均可自由配置，但相关寄存器必须按 32 位字(Word)进行访问。GPIO BSRR 寄存器为了实现对 GPIO ODR 寄存器进行原子读取/写入操作。

10.4.1 I/O 复位状态

芯片复位期间：

- PC4/PC5 配置为下拉状态，作为芯片模式引脚，外部不允许上拉，保持下拉或浮空状态
- 其他端口为浮空高阻模式

复位完成后：

- PA13/PA14 将默认复用为调试功能(AF2)
  - PA13： SWDAT 处于上拉状态
  - PA14： SWCLK 处于下拉状态
- 其他端口为浮空高阻模式

10.4.2 通用 I/O

当引脚配置为输出后，写入到输出数据寄存器(ODR)的值将在 I/O 引脚上输出。

当引脚配置为输入后，输入数据寄存器(ODR)每隔 1 个 CPU 时钟周期(F_{CLK})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_PUR/GPIOx_PDR 寄存器中的值来选择打开/关闭。

10.4.3 引脚复用

I/O 引脚通过一个复用逻辑单元连接到不同外设模块，该复用逻辑单元一次仅允许一个外设复用到 I/O 引脚上，这样可以确保共用一个 I/O 引脚的外设之间不会发生冲突。

芯片内部每个 I/O 引脚都有一个复用逻辑单元，其支持 16 路复用功能输入(AF0 至 AF15)，可通过 GPIOx_PMUXR0(针对 I/O 引脚 0~7)和 GPIOx_PMUXR1(针对 I/O 引脚 8~15)寄存器对复用功能进行配置：

- AF0 为 I/O 输入模式，AF1 为 I/O 输出模式
- 外设的复用功能映射到 AF2 至 AF14
- AF15 为模拟功能(Analog function)
- 芯片复位完成后，除 PA13/PA14 外(详见 10.4.1 I/O 复位状态)，其他 I/O 都会映射到系统的复用功能 (AF15)

I/O 复用架构除了具有灵活性，各外设还可以将复用功能映射到不同 I/O 引脚上，这可以优化在小型封装中可使用外设的数量。

复用功能配置说明

系统及调试功能：

- SWD：复位后，会将 PA13/14 引脚指定为 SWD 功能，供片上调试使用。
- MCO：复位后，默认为 AF15，若要使用 Clock Fanout 复用功能必须配置为复用功能模式。

GPIO：

- 在 GPIOx_PMUXR0/GPIOx_PMUXR1 寄存器中将所需 I/O 配置为输出或输入功能。

外设复用功能：

- 对于 ADC/DAC/CMP 模拟功能以及 USB 模块，在 GPIOx_PMUXR0/GPIOx_PMUXR1 寄存器中将所需 I/O 配置为模拟通道复用(AF15)。
- 其他外设复用功能，在 GPIOx_PMUXR0/GPIOx_PMUXR1 寄存器中将所需 I/O 配置为对应的复用功能。引脚的特定复用功能分配请参见芯片数据手册 PinMux 部分。

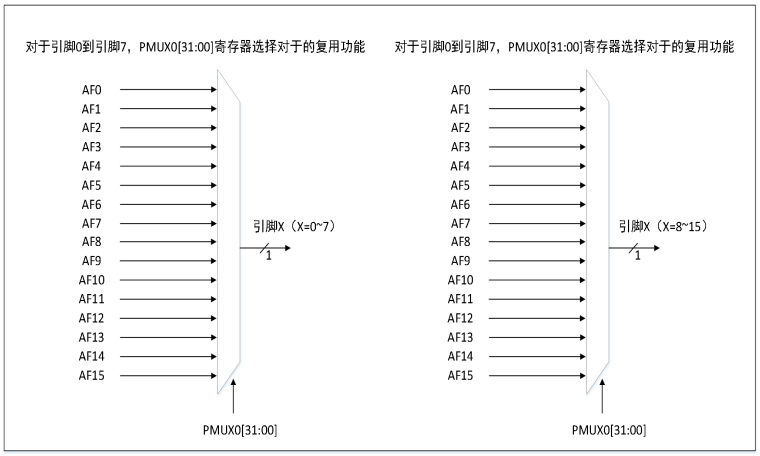


图 10-2 GPIO 选择复用功能示意图

10.4.4 I/O 控制寄存器

每个 GPIO 端口有 8 个控制寄存器(GPIOx_PMUXR0、GPIOx_PMUXR1、GPIOx_DR、GPIOx_PUR、GPIOx_PDR、GPIOx_OSRR、GPIOx_IHYR、GPIOx_DSR)，可配置多达 16 个 I/O。

其中 GPIOx_PMUXR0、GPIOx_PMUXR1 寄存器用于选择 I/O 复用功能(输入、输出、外设引脚复用、模拟功能)；GPIOx_DR、GPIOx_PUR、GPIOx_PDR 寄存器分别用于选择 I/O 输出类型(推挽或开漏)和上下拉功能；GPIOx_OSRR、GPIOx_IHYR、GPIOx_DSR 寄存器用于配置 I/O 输出摆率、输出驱动能力及输入迟滞。

注意：I/O 上下拉功能不区分输入和输出方向，都可以配置，有关寄存器说明的详细信息，请参见下面 10.5 寄存器描述。

10.4.5 I/O 数据寄存器

每个 GPIO 都具有 1 个 16 位数据寄存器，其中输入和输出数据寄存器（GPIOx_DR）既可用于存储输出数据，也可以用于存储输入数据。当用于输出时，GPIOx_DR 用于存储待输出数据，可对其进行读/写访问；当用于输入时，通过 I/O 输入的数据存储到输入数据寄存器(GPIOx_DR) 中。

注意：有关寄存器说明的详细信息，请参见下面 10.5 寄存器描述。

10.4.6 I/O 数据位操作

每个 GPIO 都具有 1 个 32 位置位复位寄存器 (GPIOx_BSRR)，其允许应用程序对输出数据中的各个单独的数据位执行置位和复位操作。置位复位寄存器是 32 位寄存器，其中低 16 位用于 Bit Set 功能，而高 16 位用于 Bit Reset 功能。当向 GPIOx_BSRR 寄存器的低 16 位中某一位写 1 时，其对应位 I/O 将置位；当向 GPIOx_BSRR 寄存器的高 16 位中某一位写 1 时，其对应位 I/O 将清零。

注意：

- *当向 GPIOx_BSRR 寄存器写入 0 对 Bit Set 和 Reset 都不会产生任何影响。*
- *当向 GPIOx_BSRR 寄存器中 Bit Set 和 Reset 同时写入 1 时也不会产生任何影响，不允许同时置位操作。*
- *通过 GPIOx_BSRR 寄存器可以实现 AHB 总线原子写操作，实现寄存器单 Bit 或多 Bit 修改操作。*

10.4.7 I/O 复用功能

每个 GPIO 都具有 2 个 32 位的 I/O 功能复用寄存器，共 16 个 I/O，每个 I/O 对应 4Bit，用来选择 16 个复用功能。通过 GPIOx_PMUXRn(n = 0 ~ 1)寄存器根据应用程序的要求将某一种复用功能映射到对应引脚上。

使用 GPIOx_PMUXRn 复用功能寄存器，可以在任意一个 I/O 上灵活选择一个功能复用。具体每个 GPIO 引脚上能够复用哪些功能，请参见数据手册相关章节。

10.4.8 外部中断功能

所有 GPIO 端口都具有外部中断功能，要使用外部中断功能，必须将端口配置为输入模式，同时配置 GPIOx_IER 寄存器开启 GPIO 端口的总中断使能，再配置 GPIOx_ITER 触发使能寄存器选取所指定的 Pin 脚允许触发中断，并配置 GPIOx_RFTSR 边缘触发设置寄存器选择需要触发的边缘(上升沿/下降沿/双边沿)。可通过 GPIOx_PR 挂起标志位寄存器检查，相应 Pin 是否触发了中断。

10.4.9 输入配置

当 I/O 端口配置为输入时：

- 输出寄存器被禁止
- 输入施密特触发可被激活(输入迟滞)
- 根据配置 GPIOx_PUR /GPIOx_PDR 寄存器来决定 I/O 是否打开上拉和下拉电阻
- 输入数据寄存器每隔 1 个 CPU 时钟周期(F_{CLK})对 I/O 引脚上的数据进行一次采样
- 读取输入数据寄存器可获得对应 I/O 的状态

输入模式

注意：所有 GPIO 引脚都具有一个上拉和下拉电阻，通过对应寄存器配置可以打开或关闭。

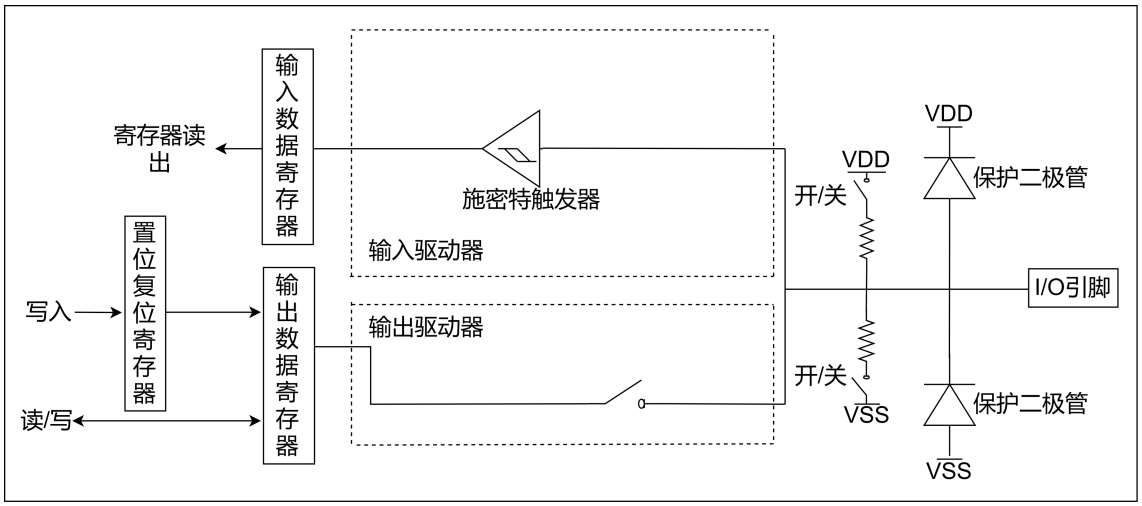


图 10-3 GPIO 输入上拉/下拉配置

浮空状态

当 I/O 引脚的上拉和下拉电阻都处于断开状态，此时 I/O 引脚处于悬空状态，容易收到外部干扰，输入数据寄存器中的数据并不准确。

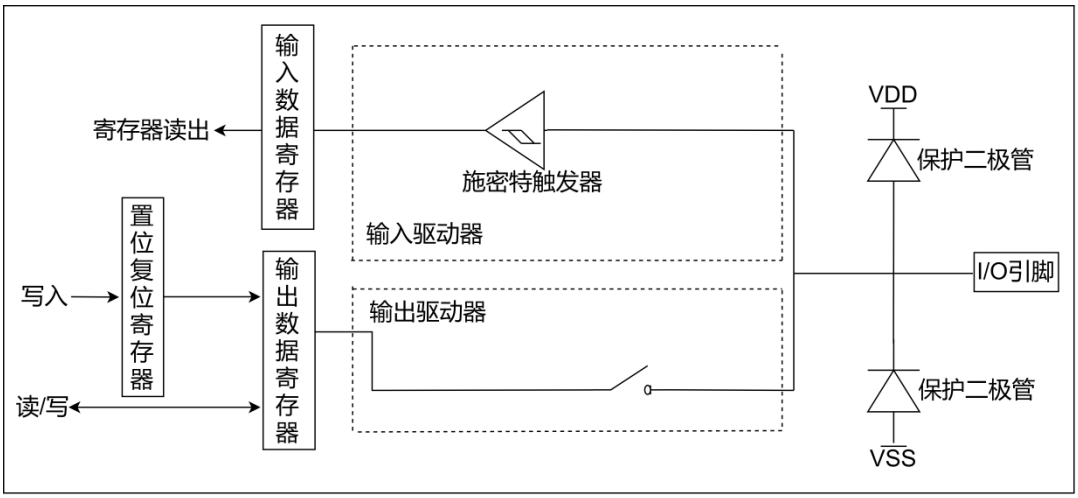


图 10-4 GPIO 输入浮空配置

输入上拉

打开 I/O 引脚的上拉电阻，将 I/O 引脚的不确定信号通过一个电阻嵌到高电平，电阻起到限流作用。

通常可以给 I/O 引脚外接一个开关，开关一端接 I/O 引脚，一端接地。当按下按钮时，读出低电平；松开按钮时，读出高电平。也可外接一个强上拉电阻，一直读取到高电平。

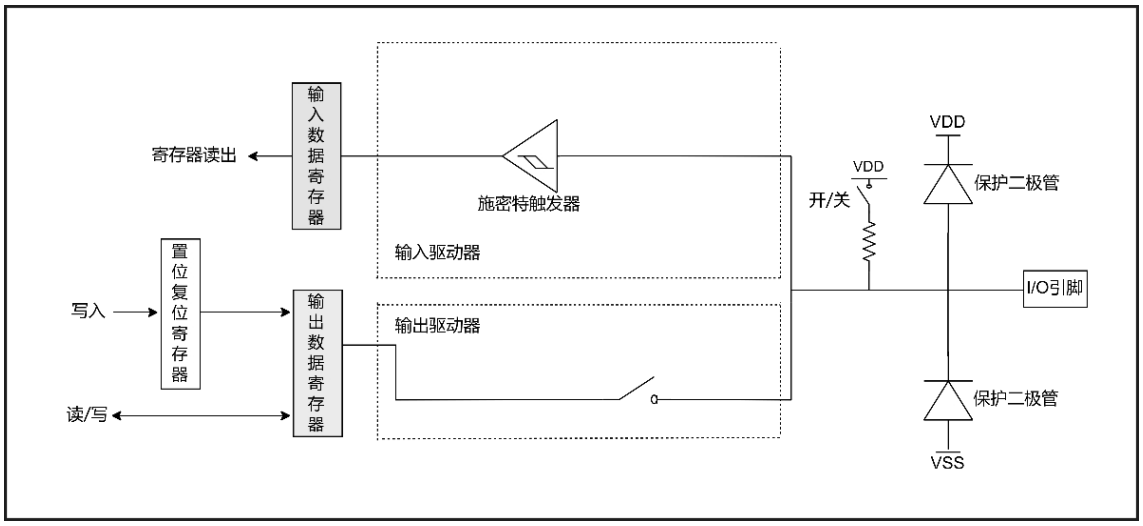


图 10-5 GPIO 输入上拉配置

输入下拉

与上拉输入原理一样，将电位拉到 GND，读到低电平。

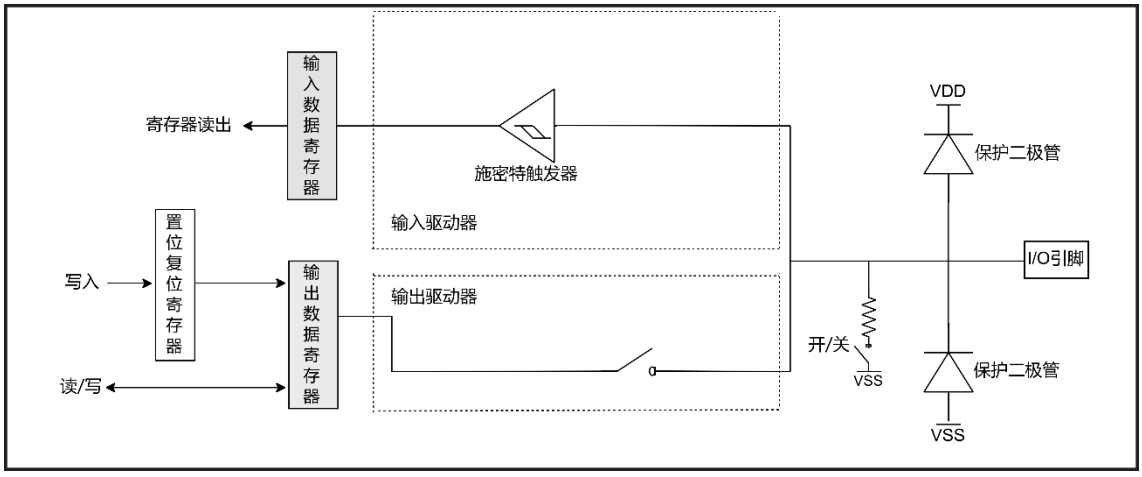


图 10-6 GPIO 输入下拉配置

10.4.10 输出配置

当 I/O 端口被配置为输出时：

- 输出数据寄存器被激活，输入数据寄存器被禁用
  - 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”会使端口保持高阻状态(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。
- 根据配置 GPIOx_PUR /PDR 寄存器来决定 I/O 是否打开上拉和下拉电阻
- 对输出数据寄存器的读访问得到最后一次写的值

输出模式

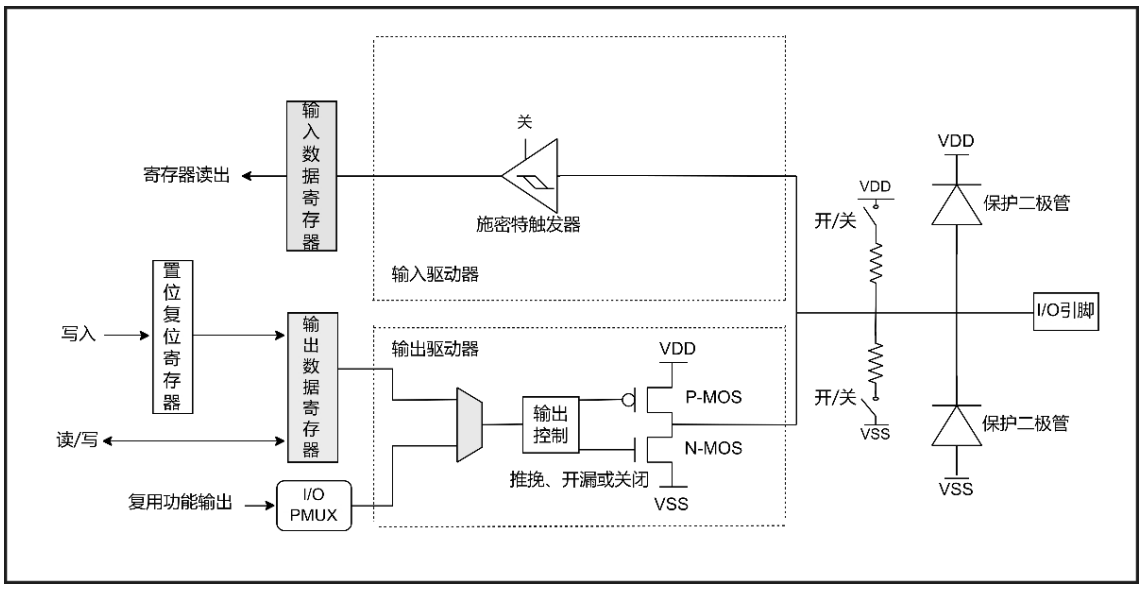


图 10-7 GPIO 输出推挽/开漏或关闭配置

推挽输出

可以输出高、低电平，连接数字器件；推挽结构一般是指两个三极管分别受两互补信号的控制，总是在一个三极管导通的时候另一个截止。

推挽输出，写 1

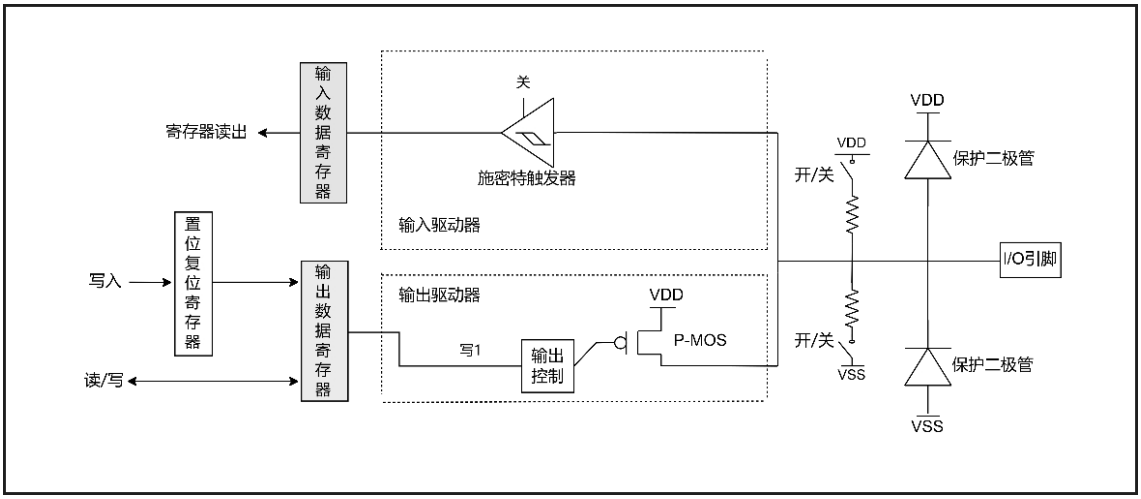


图 10-8 GPIO 推挽输出配置（写 1）

P-MOS 管激活，连接 VDD，I/O 引脚输出高电平，输入数据寄存器读到高电平。

推挽输出，写 0

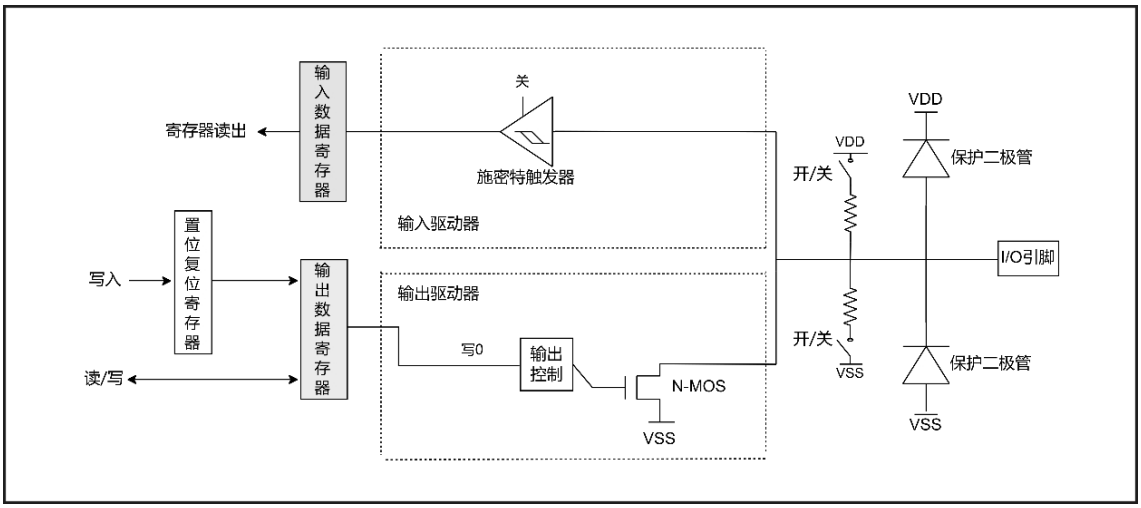


图 10-9 GPIO 推挽输出配置（写 0）

N-MOS 管被激活，连接到 VSS 接地，I/O 引脚输出低电平，输入数据寄存器读到低电平。

开漏输出

输出端相当于三极管的集电极，要得到高电平状态需要上拉电阻才行，适合于做电流型的驱动，其吸收电流的能力相对强。

开漏输出，写 1

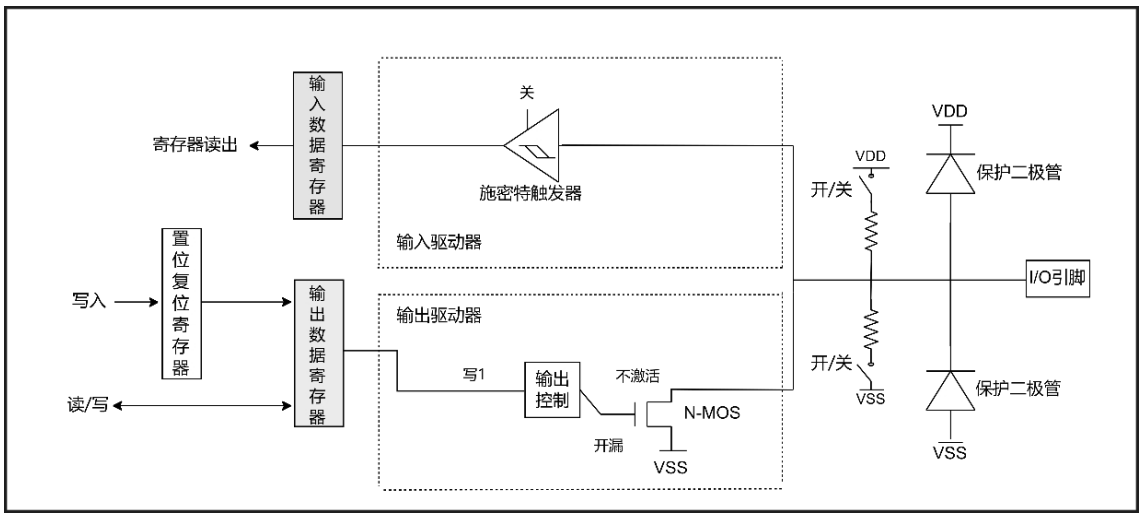


图 10-10 GPIO 开漏输出配置（写 1）

输出数据寄存器写入“1”，N-MOS 管不激活，引脚相当于浮空状态，引脚电平由外接电阻决定，接上拉电阻则为高电平，接下拉电阻相当于低电平。输出数据寄存器的写“1”不会使得引脚输出高电平 ，写“0”会使得引脚输出低电平。

开漏输出，写 0

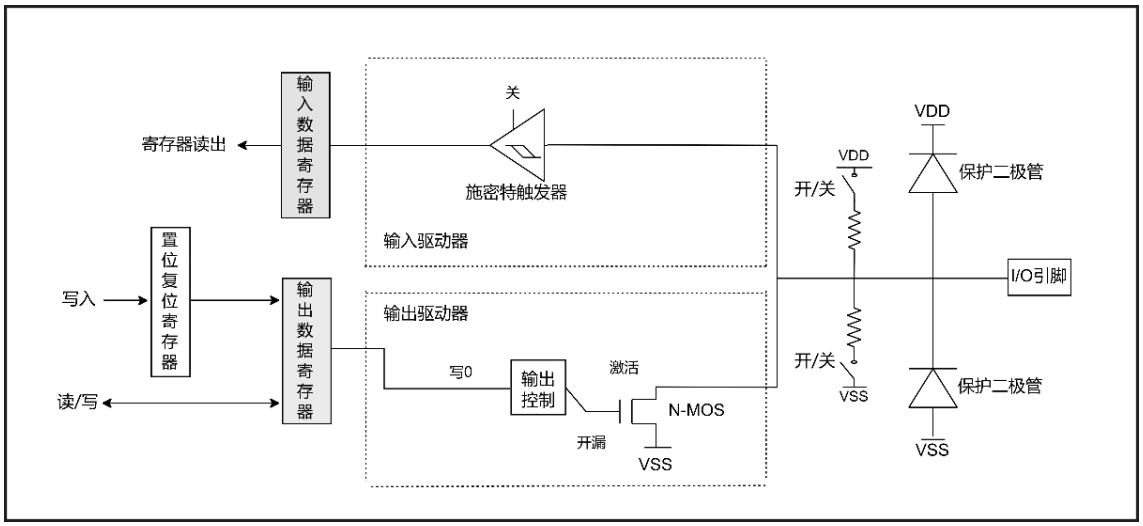


图 10-11 GPIO 开漏输出配置（写 0）

N-MOS 管激活，连接 VSS 接地，I/O 引脚输出低电平，输入数据寄存器也可以读到低电平。

10.4.11 复用功能配置

当 I/O 端口被配置为复用功能时：

- 可将输出配置为开漏或推挽式模式
- 输出缓冲器可由内置外设的信号驱动(复用功能输出)
- 根据配置 GPIOx_PUR/GPIOx_PDR 寄存器来决定 I/O 是否打开上拉和下拉电阻

复用推挽/开漏可以理解为 GPIO 口被用作第二功能时的配置情况，即非通用 I/O 口，GPIOx_PMUXRn 寄存器允许用户把一些复用功能重新映射到不同的引脚。

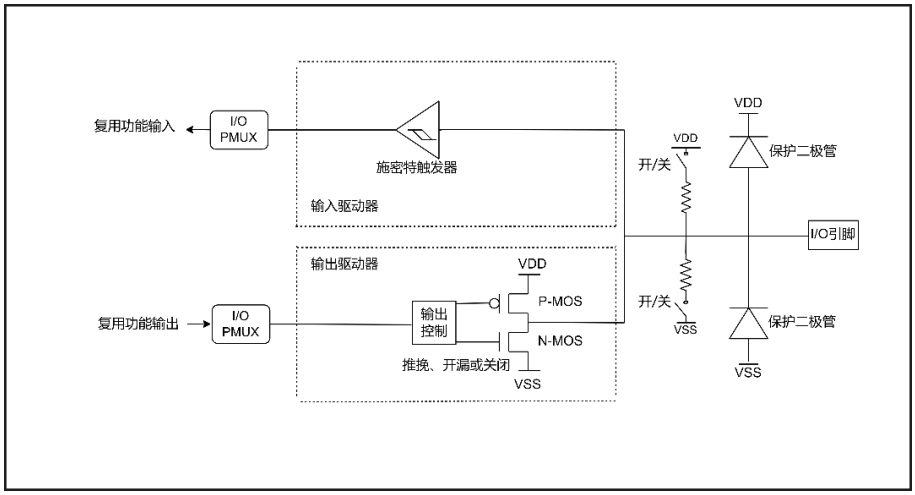


图 10-12 GPIO 复用功能配置

模拟输入配置

- 输出缓冲器被禁止
- 禁止施密特触发输入，实现了每个模拟 I/O 引脚上的零消耗。施密特触发输出值被强置为“0”
- 弱上拉和下拉电阻被禁止

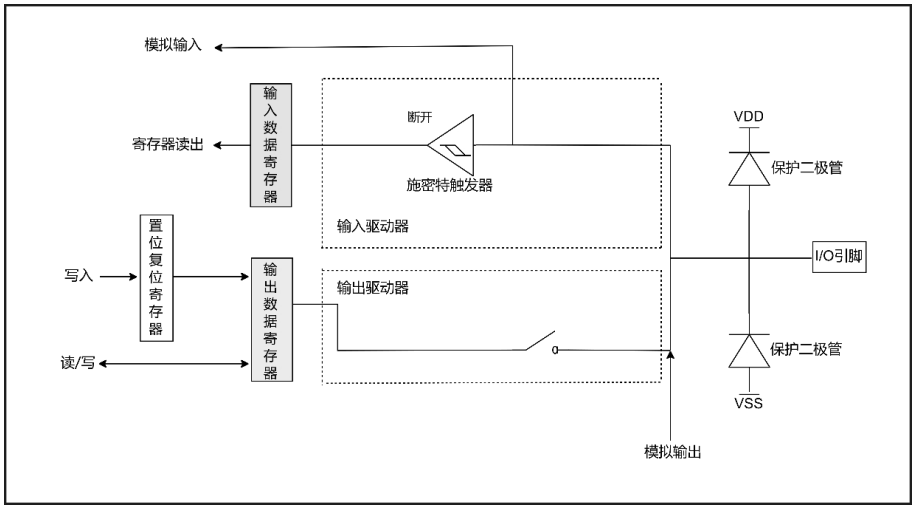


图 10-13 GPIO 模拟输入配置

注意：模拟输入，读到的不是高低电平，而是准确的电压值。

10.5 寄存器描述

10.5.1 寄存器列表

Name	Offset	Width	Description
GPIOx_BSRR	0x00	32bits	GPIO Bit Set/Reset Register
GPIOx_DR	0x04	32bits	GPIO Data Register
GPIOx_PUR	0x08	32bits	GPIO Pullup Register
GPIOx_PDR	0x0C	32bits	GPIO Pulldown Register
GPIOx_DSR	0x10	32bits	GPIO Driver Strength Register
GPIOx_IHYR	0x14	32bits	GPIO Input Hysteresis Register
GPIOx_OTYPR	0x18	32bits	GPIO Output Type Register
GPIOx_OSRR	0x1C	32bits	GPIO Output Slew Rate Register
GPIOx_IER	0x20	32bits	GPIO Interrupt Enable Register
GPIOx_ITER	0x24	32bits	GPIO Interrupt Trigger Register
GPIOx_RFTSR	0x28	32bits	GPIO Falling/Rising Trigger Selection Register
GPIOx_PR	0x2C	32bits	GPIO Pending Register
GPIOx_SDER	0x30	32bits	GPIO Sync/Debounce Enable Register
GPIOx_PMUXR0	0x40	32bits	GPIO Pin-Mux Register0
GPIOx_PMUXR1	0x44	32bits	GPIO Pin-Mux Register1

10.5.2 寄存器详细描述

10.5.2.1 GPIOx Bit Set/Reset Register (GPIOx_BSRR)

- **Name**：GPIOx Bit Set/Reset Register(x = A to D)
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	BR[y]	W	0x0	GPIOx Bit y Reset (y = 0 … 15): 0：无效 1：复位相应的位 注意： Bit Set 及 Reset 每位对应 Data 一个 Bit，并且 Set 与 Reset 互斥，同时只能一种有效。
[15-0]	BS[y]	W	0x0	GPIOx Bit y Set (y = 0 … 15): 0：无效 1：设置相应的位 注意： Set 及 Reset 每位对应 Data 一个 Bit，并且 Set 与 Reset 互斥，同时只能一种有效。

10.5.2.2 GPIOx Data Register (GPIOx_DR)

- **Name**：GPIOx Data Register(x = A to D)
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	DR[y]	R/W	0x0	GPIOx Bit y Input/Output Data (y = 0 … 15): GPIO 数据控制寄存器，用于控制端口输入/输出数据。 注意： 1、当 Pinmux 配置输入模式，DATA REGISTER 将端口数据采集到寄存器内，可以通过读取该寄存器来获取 IO 数据输入； 2、当 Pinmux 配置输出模式，DATA REGISTER 数据将会映射到端口上输出。

10.5.2.3 GPIOx Pullup Register (GPIOx_PUR)

- **Name**：GPIOx Pullup Register(x = A to D)
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	PU[y]	R/W	参考描述	GPIOx Bit y Pullup (y = 0 … 15): 0：关闭 1：使能上拉 注意： 1、上拉控制位，每位对应一个 IO 控制； 2、一些特殊功能的 IO(例如 SW 调试接口)需要默认做上拉处理，默认为上拉，需要特别注意； 3、GPIO 默认值：GPIOA13/A14(默认上拉)-0x6000，其他 GPIO-0x0。

10.5.2.4 GPIOx Pulldown Register (GPIOx_PDR)

- **Name**：GPIOx Pulldown Register(x = A to D)
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	PD[y]	R/W	参考描述	GPIOx Bit y Pulldown (y = 0 … 15): 0：关闭 1：使能下拉 注意： 1、下拉控制位，每位对应一个 IO 控制； 2、GPIO 默认值：GPIOC4/C5(复位期间默认下拉，复位完成后自动释放)，其他 GPIO-0x0。

10.5.2.5 GPIOx Driver Strength Register (GPIOx_DSR)

- **Name**：GPIOx Driver Strength Register(x = A to D)
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	DS[y]	R/W	0x0	GPIOx Bit y Driver Strength (y = 0 … 15): 0：8mA 1：24mA

10.5.2.6 GPIOx Input Hysteresis Register (GPIOx_IHYR)

- **Name**：GPIOx Input Hysteresis Register(x = A to D)
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	IHY[y]	R/W	0x0	GPIOx Bit y Input Hysteresis (y = 0 … 15): 0：关闭 1：使能输入迟滞(施密特触发器)

10.5.2.7 GPIOx Output Type Register (GPIOx_OTYPR)

- **Name**：GPIOx Output Type Register(x = A to D)
- **Size**：32bits
- **Offset**：0x18
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OT[y]	R/W	0x0	GPIOx Bit y Output Type(y = 0 … 15): 0：推挽输出(复位状态) 1：开漏输出

10.5.2.8 GPIOx Output Slew Rate Register (GPIOx_OSRR)

- **Name**：GPIOx Output Slew Rate Register(x = A to D)
- **Size**：32bits
- **Offset**：0x1C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OSR[y]	R/W	0x0	GPIOx Bit y Output Slew Rate(y = 0 … 15): 0：正常压摆率 1：增强压摆率 注意：增强压摆率可以提升 IO 口输出速率，但相应会带来高的 EMI 干扰。

10.5.2.9 GPIOx Interrupt Enable Register (GPIOx_IER)

- **Name**：GPIOx Interrupt Enable Register(x = A to D)
- **Size**：32bits
- **Offset**：0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	IE	R/W	0x0	GPIOx Interrupt Enable: 0：关闭 1：启动 GPIOx 中断功能 注意：GPIOx 的总中断使能

10.5.2.10 GPIOx Interrupt Trigger Enable Register (GPIOx_ITER)

- **Name**：GPIOx Interrupt Trigger Enable Register(x = A to D)
- **Size**：32bits
- **Offset**：0x24
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ITE[y]	R/W	0x0	GPIOx Bit y Interrupt Trigger Enable (y = 0 … 15): 0：关闭相应引脚触发中断 1：使能相应引脚触发中断

10.5.2.11 GPIOx Rising/Falling Trigger Selection Register (GPIOx_RFTSR)

- **Name**：GPIOx Rising/Falling Trigger Selection Register(x = A to D)
- **Size**：32bits
- **Offset**：0x28
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	FTS[y]	R/W	0x0	Falling Trigger Select on GPIOx Pin y(y = 0 … 15): 0：关闭 1：使能下降沿触发
[15-0]	RTS[y]	R/W	0x0	Rising Trigger Select on GPIOx Pin y(y = 0 … 15): 0：关闭 1：使能上升沿触发

10.5.2.12 GPIOx Pending Register (GPIOx_PR)

- **Name**：GPIOx Pending Register(x = A to D)
- **Size**：32bits
- **Offset**：0x2C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	PIF[y]	R/W1C	0x0	GPIOx Pin y Pending Interrupt Flag(y = 0 … 15): 0：未发生触发请求 1：检测到触发请求

10.5.2.13 GPIOx Sync/Debounce Enable Register (GPIOx_SDER)

- **Name**：GPIOx Sync/Debounce Enable Register(x = A to D)
- **Size**：32bits
- **Offset**：0x30
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	SYNE	R/W	0x0	GPIO input Sync Enable: 0: 关闭 1: 使能输入同步功能 说明：使能输入同步功能，输入信号将与内部时钟进行边沿对齐。
[15-0]	DBCE	R/W	0x0	GPIO input Debounce Enable: 0: 关闭 1: 使能输入消抖功能 说明: 使能输入消抖功能，外部信号经过 4 个 CPU 时钟(FCLK)消抖后输入

10.5.2.14 GPIOx Pin-Mux Register0 (GPIOx_PMUXR0)

- **Name**：GPIOx Pin-Mux Register0(x = A to D)
- **Size**：32bits
- **Offset**：0x34
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-28]	PMUX7	R/W	参考总述	GPIOx Pin7 Mux select: 参考 PMUX0
[27-24]	PMUX6	R/W	参考总述	GPIOx Pin6 Mux select: 参考 PMUX0
[23-20]	PMUX5	R/W	参考总述	GPIOx Pin5 Mux select: 参考 PMUX0
[19-16]	PMUX4	R/W	参考总述	GPIOx Pin4 Mux select: 参考 PMUX0
[15-12]	PMUX3	R/W	参考总述	GPIOx Pin3 Mux select: 参考 PMUX0
[11-8]	PMUX2	R/W	参考总述	GPIOx Pin2 Mux select: 参考 PMUX0
[7-4]	PMUX1	R/W	参考总述	GPIOx Pin1 Mux select: 参考 PMUX0
[3-0]	PMUX0	R/W	参考总述	GPIOx Pin0 Mux select: 0000：AF0 …… 1111：AF15 注意：GPIO 复用功能选择，具体每个 I/O 的复用映射请参考芯片数据手册内的 PinMux 章节。

10.5.2.15 GPIOx Pin-Mux Register1 (GPIOx_PMUXR1)

- **Name**：GPIOx Pin-Mux Register1(x = A to D)
- **Size**：32bits
- **Offset**：0x38
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-28]	PMUX15	R/W	参考总述	GPIOx Pin15 Mux select: 参考 PMUX8
[27-24]	PMUX14	R/W	参考总述	GPIOx Pin14 Mux select: 参考 PMUX8
[23-20]	PMUX13	R/W	参考总述	GPIOx Pin13 Mux select: 参考 PMUX8
[19-16]	PMUX12	R/W	参考总述	GPIOx Pin12 Mux select: 参考 PMUX8
[15-12]	PMUX11	R/W	参考总述	GPIOx Pin11 Mux select: 参考 PMUX8
[11-8]	PMUX10	R/W	参考总述	GPIOx Pin10 Mux select: 参考 PMUX8
[7-4]	PMUX9	R/W	参考总述	GPIOx Pin9 Mux select: 参考 PMUX8
[3-0]	PMUX8	R/W	参考总述	GPIOx Pin8 Mux select: 0000：AF0 …… 1111：AF15 注意：GPIO 复用功能选择，具体每个 I/O 的复用映射请参考芯片数据手册内的 PinMux 章节。

10.5.3 时钟输出功能

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

- LSI
- HSI
- HSE
- PLL0DivClk
- PLL0/16
- PLL1DivClk
- PLL1/16
- PLL2DivClk
- PLL2/16

所需的时钟源通过 SYSCFG 时钟配置寄存(SYSCFGCR) 中 MCOSRC[7:4]位选择。

对于 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）寄存器描述

10.6 寄存器描述

参考 SYSCTRL_PLL0CR 寄存器至 SYSCTRL_ AHBRSTCR 寄存器的描述。

11 DMA 控制器（DMA）

11.1 简介

直接存储器访问(DMA)用于在外设与存储器之间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 实现数据快速搬运，这样可以很大程度地节省 CPU 资源，减少中断进入次数，最终提高整体系统的性能。

DMA 控制器基于复杂的总线矩阵架构，DMA 控制器包含独立的 FIFO 和双 AHB 主总线架构，优化了系统带宽，有效地实现数据传输。

DMA 控制器有两个通道，每个通道总共可以有多达 16 个请求，两个通道可以分配给一个或多个特定的外围设备进行数据传输，每个通道都有一个仲裁器，用于处理 DMA 请求间的优先级。

11.2 结构框图

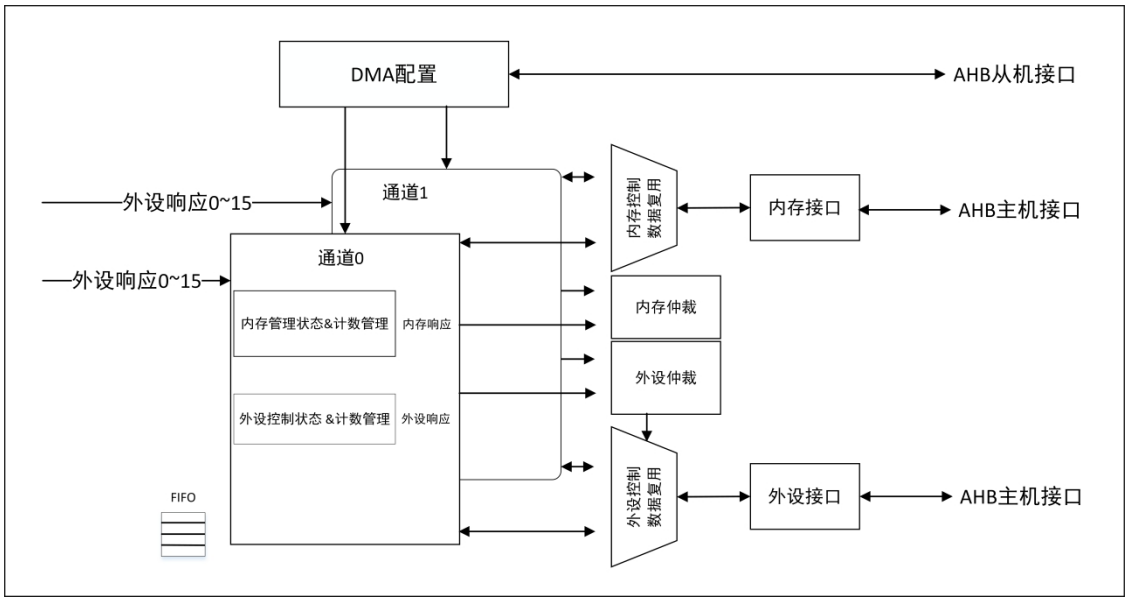


图 11-1　DMA 结构框图

11.3 主要特性

DMA 主要具有以下特性：

- 包含 2 个通道，每个通道连接 8 个特定的外设请求
- 存储器和外围设备支持单一传输，4 拍、8 拍和 16 拍增量突发传输
- 支持外设 DMA：2 个 UART，2 个 I2C
- 当外围设备向存储器发送数据时支持改变存储器
- 支持对所有内部存储的 DMA 访问
- 支持软件优先级(低/中/高/超高)和硬件优先级(通道数越低，优先级越高)
- 存储器和外设的数据传输宽度可配置为：字节/半字/字
- 存储器和外设的数据传输支持固定和增量寻址
- 支持循环传输模式
- 支持单数据传输和多种数据传输方式
  - 多种数据传输方式：当存储器数据和外围数据的宽度不同时，自动打包/解包数据;
  - 单数据传输模式：当且仅当 FIFO 为空时，数据从源地址读取，存储在 FIFO 中，FIFO 的数据写入目标地址。
- 每个通道有 5 种类型的事件标志和独立的中断，支持启用和重置中断

11.4 功能描述

11.4.1 DMA 传输

DMA 控制器执行直接存储传输：DMA 传输接口采用 AHB 总线，可以控制 AHB 总线矩阵来启动 DMA 传输事务。

DMA 传输事务主要包括以下几项：

- 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
- 外设到外设的传输

DMA 控制器提供两个 AHB 端口：AHB 存储器端口(连接存储器 SRAMB/C)和 AHB 外设端口(连接外设及 SRAMA)。

11.4.2 DMA 事务

一个 DMA 事务(Burst)由一串数据传输序列组成，需传输数据项的数目及宽度(8 位、16 位或 32 位)可由软件配置。

每个 DMA 传输都包含以下操作：

- 通过 SAR 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 DAR 寄存器寻址，将加载的数据存储到外设数据存储器或存储器单元。

11.4.3 DMA 通道

DMA 模块共包含 2 个通道，软件可以自由选择通道作为 DMA 传输通道。

DMA 每个通道可软件自由配置源端/目标端的外设握手接口。通过配置相应通道的 DMA_CH_CR3 寄存器中的 SHSIF[3:0]/DHSIF[3:0]位段，选择源外设/目标外设的外设握手接口源。

表 11-1 给出了 DMA 握手接口映射。

表 11-1 DMA 握手接口映射表

DMA 握手接口 SHSIF[3:0]/DHSIF[3:0]	连接外设请求信号	说明
DMA HS Interface 0	I2C0_TX_REQ	-
DMA HS Interface 1	I2C0_RX_REQ	-
DMA HS Interface 2	I2C1_TX_REQ	-
DMA HS Interface 3	I2C1_RX_REQ	-
DMA HS Interface 8	UART0_TX_REQ	-
DMA HS Interface 9	UART0_RX_REQ	-
DMA HS Interface 10	UART1_TX_REQ	-
DMA HS Interface 11	UART1_RX_REQ	-

说明：如果源端/目标端配置的是 Memory，则该配置无效。

11.4.4 DMA 仲裁

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 2 个 DMA 数据请求管理，并启动外设/存储器访问序列。

优先级管理分为两种：

- 软件：每个数据流优先级都可以在相应通道的 DMA_CH_CR2 寄存器中 PRI 位进行配置，分为两个级别：
  - 高优先级(0x1)
  - 低优先级(0x0)
- 硬件：如果两个请求具有相同的软件优先级，则编号低的通道优先于编号高的通道。例如，通道 0 的优先级高于通道 1。

11.4.5 DMA 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。

传输方向使用相应通道的 DMA_CH_CR0 寄存器中的 TTC[1:0]位进行配置，有四种可能的传输方向：存储器到外设、外设到存储器、存储器到存储器、外设到外设，下表介绍了四种传

输类型。

表 11-2 传输方向

TTC[1:0]位	方向
00	存储器到存储器(M2M)
01	存储器到外设(M2P)
10	外设到存储器(P2M)
11	外设到外设(P2P)

当数据宽度(相应通道的 DMA_CH_CR0 寄存器中的 STW/DTW 位编程)分别是半字或字时，写入该 DMA 通道的 DMA_CH_SAR 或 DMA_CH_DAR 寄存器的外设或存储器地址必须分别按字或半字地址的边界对齐。

存储器到存储器模式

当使能这种模式时，每次产生外设请求，DMA 都会启动数据源到 FIFO 的传输。达到 FIFO 的阈值级别时，将 FIFO 内容移出并存储到目标中。如果 DMA 使能位由软件清零或传输完成即会停止。

只有赢得了仲裁的通道才有权访问 AHB 的源或目标端口，软件使用可以通过配置相应通道的 DMA_CH_CR2 寄存器 PRI 位为相应的通道定义优先级。

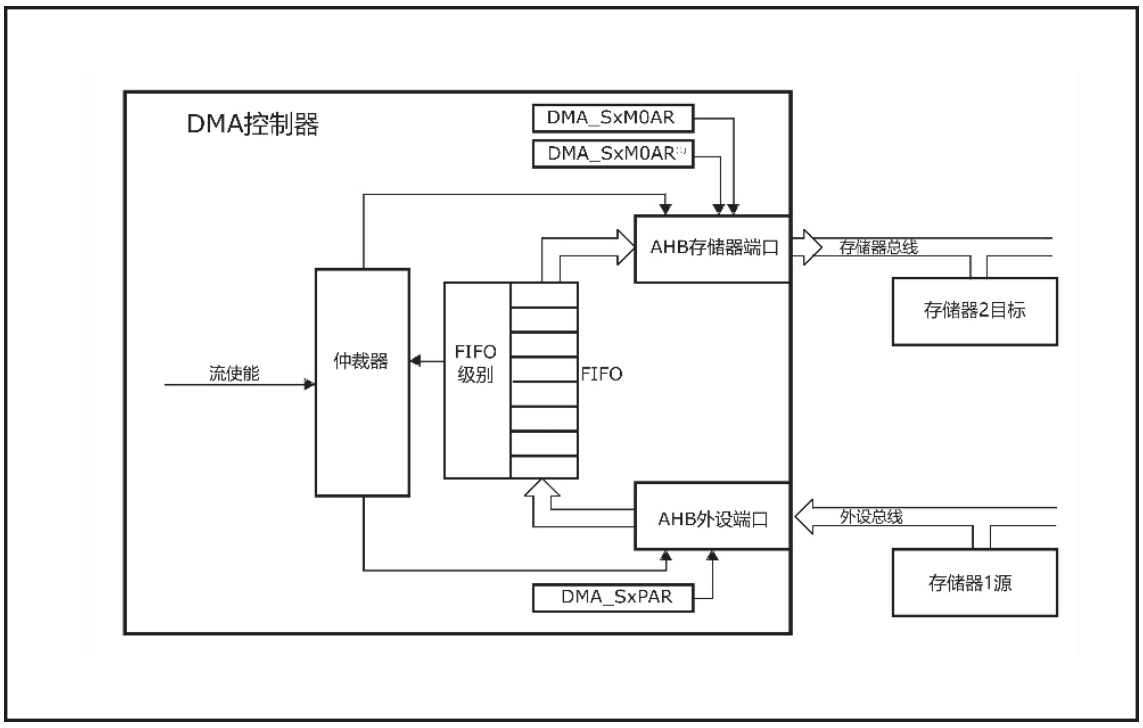


图 11-2 存储器到存储器模式示意图

外设到存储器模式

当使能这种模式时，每次产生外设请求，DMA 都会启动数据源到 FIFO 的传输。达到 FIFO 的阈值级别时，将 FIFO 内容移出并存储到目标中。如果 DMA 使能位由软件清零或传输完成即会停止。

在直接模式下，不使用 FIFO 的阈值级别(通道的 DMA_CH_CR3 寄存器中的 FMD 值为“0”）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。

只有赢得了仲裁的通道才有权访问 AHB 的源或目标端口，软件使用可以通过配置相应通道的 DMA_CH_CR2 寄存器 PRI 位为相应的通道定义优先级。

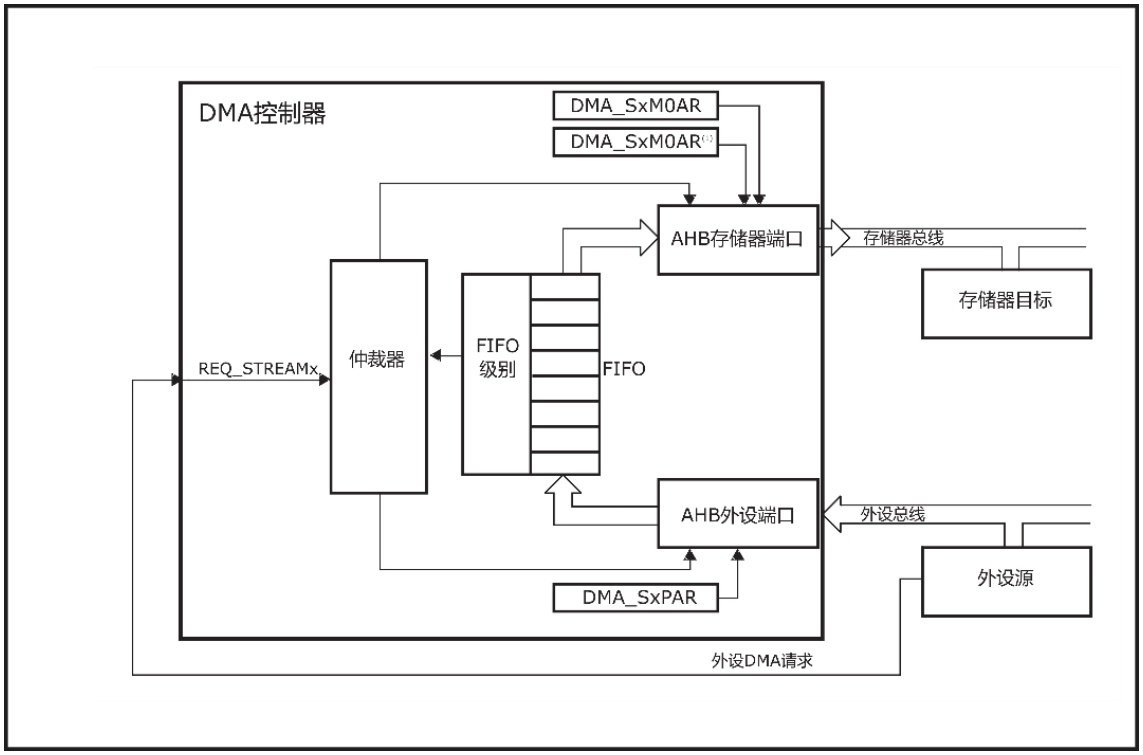


图 11-3 外设到存储器模式示意图

存储器到外设模式

当使能这种模式时，每次产生外设请求，DMA 都会启动数据源到 FIFO 的传输。达到 FIFO 的阈值级别时，将 FIFO 内容移出并存储到目标中。如果 DMA 使能位由软件清零或传输完成即会停止。

在直接模式下，不使用 FIFO 的阈值级别(通道的 DMA_CH_CR3 寄存器中的 FMD 值为“0”）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。

只有赢得了仲裁的通道才有权访问 AHB 的源或目标端口，软件使用可以通过配置相应通道的 DMA_CH_CR2 寄存器 PRI 位为相应的通道定义优先级。

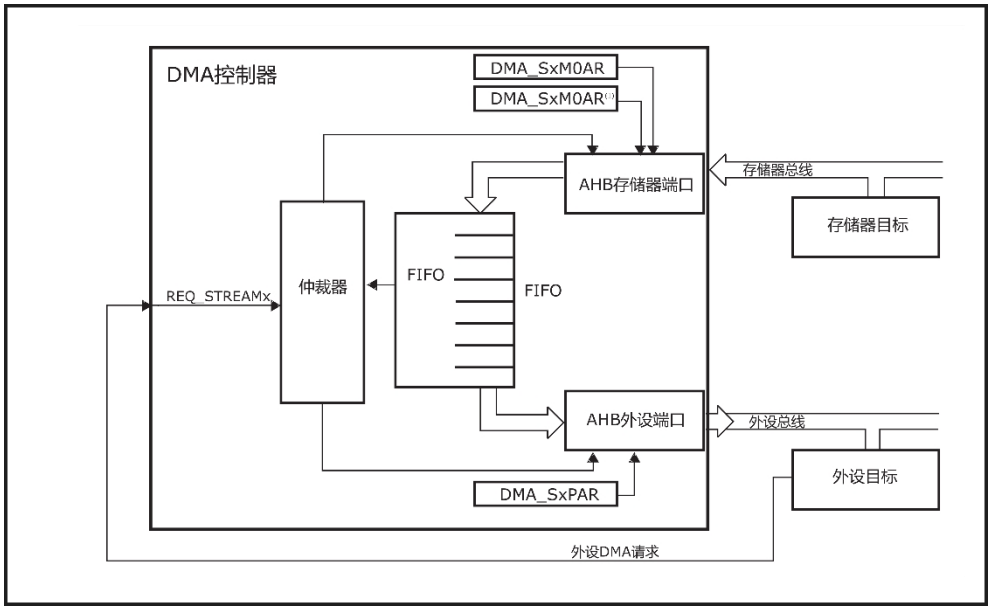


图 11-4 存储器到外设模式示意图

外设到外设模式

当使能这种模式时，每次产生外设请求，DMA 都会启动数据源到 FIFO 的传输。达到 FIFO 的阈值级别时，将 FIFO 内容移出并存储到目标中。如果 DMA 使能位由软件清零或传输完成即会停止。

在直接模式下，不使用 FIFO 的阈值级别(通道的 DMA_CH_CR3 寄存器中的 FMD 值为“0”）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。

只有赢得了仲裁的通道才有权访问 AHB 的源或目标端口，软件使用可以通过配置相应通道的 DMA_CH_CR2 寄存器 PRI 位为相应的通道定义优先级。

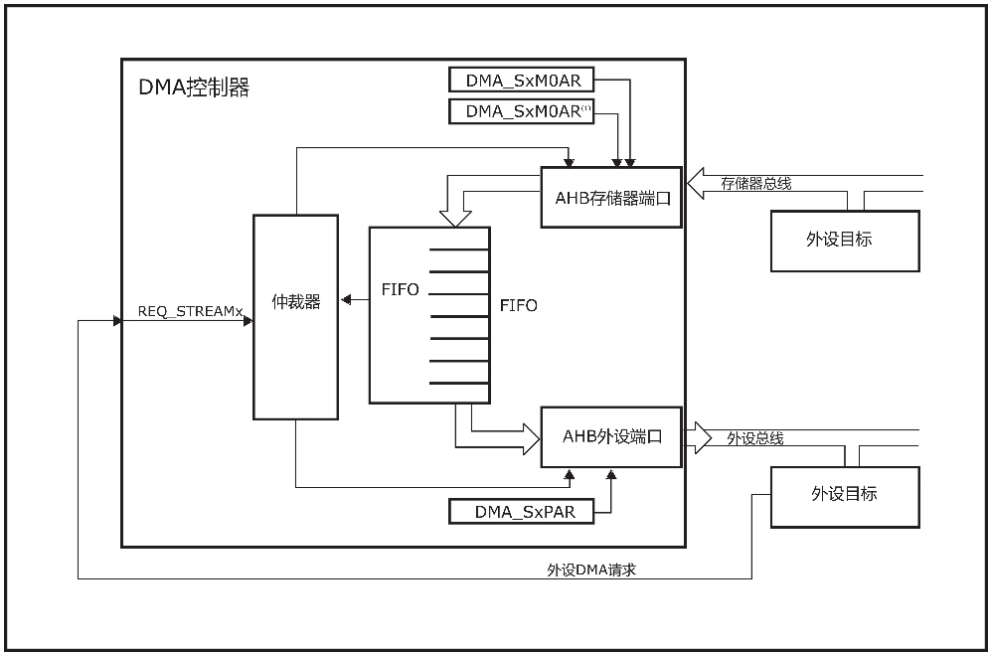


图 11-5 外设到外设模式示意图

11.4.6 DMA 地址控制

根据 DMA 通道的 DMA_CH_CR0 寄存器中 SINC 和 DINC 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、递减或保持常量，如果只通过单个寄存器访问外设源或目标数据时，可以禁止递增模式。

如果使能了递增模式，则根据 STW/DTW 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 外设端口数据位宽，将外设地址的增量偏移大小固定下来。例如将 DMA_DCTR0 寄存器中的 STW/DTW 位用于将增量偏移大小与外设 AHB 端口固定为 32 位地址，此时地址递增 4，需要确保地址按 32 位数据大小对齐。

FIFO 用于存储在源数据传输到目标之前的临时数据，每个通道都有一个独立的 FIFO，阈值级别可由软件配置为 1/2 启动传输(使用 FIFO 模式，非直接模式，FMD=1)。

11.4.7 DMA 中断

DMA 中每个通道都有 5 种类型的状态/中断：

- Transfer Finish：DMA 完成所有数据传输后，产生传输完成标志，挂起相应的中断请求标志，并关闭 DMA 的通道使能。
- Block Transfer：在 DMA 块传输完成最后一笔数据传递到目标端后，产生块传输完成标志，并挂起相应的中断请求标志。
- Destination Transfer：完成最后一笔数据传输到目标端后，产生目的端传输完成标志，并挂起相应的中断请求标志。
- Source Transfer：完成最后一笔数据从源端传输完成后，产生源端传输完成标志，并挂起相应的中断请求标志。
- Transfer Error：在传输过程中由于传输出现异常(例如外设总线选择错误 SMS/DMS)，产生传输错误标志，挂起相应的请求标志并且取消 DMA 传输、关闭通道使能。

注意：如果源地址或目的地址是存储器(Memory)，那么 Destination/Source Transfer 中断将无效。

11.5 寄存器描述

11.5.1 寄存器列表

Name	Offset	Width	Description
DMA_CH_SAR	0x00 + (n * 0x58)	32bits	DMA 通道 n 源地址寄存器
DMA_CH_DAR	0x08 + (n * 0x58)	32bits	DMA 通道 n 目标地址寄存器
DMA_CH_CR0	0x18 + (n * 0x58)	32bits	DMA 通道 n 控制寄存器 0
DMA_CH_CR1	0x1C + (n * 0x58)	32bits	DMA 通道 n 控制寄存器 1
DMA_CH_CR2	0x40 + (n * 0x58)	32bits	DMA 通道 n 控制寄存器 2
DMA_CH_CR3	0x44 + (n * 0x58)	32bits	DMA 通道 n 控制寄存器 3
DMA_TSR	0x2C0	32bits	DMA 传输状态寄存器
DMA_BTSR	0x2C8	32bits	DMA 块传输状态寄存器
DMA_STSR	0x2D0	32bits	DMA 源端传输状态寄存器
DMA_DTSR	0x2D8	32bits	DMA 目标端传输状态寄存器
DMA_TESR	0x2E0	32bits	DMA 传输错误状态寄存器
DMA_TIPR	0x2E8	32bits	DMA 传输中断挂起寄存器
DMA_BTIPR	0x2F0	32bits	DMA 块传输中断挂起寄存器
DMA_STIPR	0x2F8	32bits	DMA 源端传输中断挂起寄存器
DMA_DTIPR	0x300	32bits	DMA 目标端传输中断挂起寄存器
DMA_TEIPT	0x308	32bits	DMA 传输错误中断挂起寄存器
DMA_TIMR	0x310	32bits	DMA 传输中断掩码寄存器
DMA_BTIMR	0x318	32bits	DMA 块传输中断掩码寄存器
DMA_STIMR	0x320	32bits	DMA 源端传输中断掩码寄存器
DMA_DTIMR	0x328	32bits	DMA 目标端传输中断掩码寄存器
DMA_TEIMR	0x330	32bits	DMA 传输错误中断掩码寄存器
DMA_TCR	0x338	32bits	DMA 传输状态清除寄存器
DMA_BTCR	0x340	32bits	DMA 块传输状态清除寄存器
DMA_STCR	0x348	32bits	DMA 源端传输状态清除寄存器
DMA_DTCR	0x350	32bits	DMA 目标端传输状态清除寄存器
DMA_TECR	0x358	32bits	DMA 传输错误状态清除寄存器
DMA_CR0	0x398	32bits	DMA 控制寄存器 0
DMA_CR1	0x3A0	32bits	DMA 控制寄存器 1

11.5.2 寄存器详细描述

11.5.2.1 DMA 通道 n 源地址寄存器 (DMA_CH_SAR)

- **Name**：DMA Channel n Source Address Register (n=0 ...1)
- **Size**：32bits
- **Offset**：0x00 + (n * 0x58)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	SAR	R/W	0x0	当前 DMA 传输的源地址 (DMA Source Address) 注意： ● DMA 传输地址是否自动进行递增、递减或者保持不变由寄存器 DMA_CH_CR0[10:9]位决定 ● SAR 地址必须与 DMA_CH_CR0[6:4]保持对齐 ● DMA 的源地址由软件配置

11.5.2.2 DMA 通道 n 目标地址寄存器 (DMA_CH_DAR)

- **Name**：DMA Channel n Destination Address Register (n=0 ... 1)
- **Size**：32bits
- **Offset**：0x08 + (n * 0x58)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DAR	R/W	0x0	当前 DMA 传输的目标地址 (DMA Destination Address) 注意： ● DMA 传输地址是否进行递增、递减或者保持不变由寄存器 DMA_CH_CR0[8:7] (源端地址增长模式选择 Destination Address Increase)位域决定 ● DAR 地址必须与 DMA_CH_CR0[3:1]保持对齐 ● DMA 的目标地址由软件配置

11.5.2.3 DMA 通道 n 控制寄存器 0 (DMA_CH_CR0)

- **Name**：DMA Channel n Control Register0 (n=0 ... 1)
- **Size**：32bits
- **Offset**：0x18 + (n * 0x58)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-27]	Reserved	Reserved	Reserved	Reserved
[26-25]	SMS	R/W	0x0	源端外设总线选择控制位 (Source Master Select) 00：AHB Master1 01：AHB Master2 1x：Reserved 注意：参考目标设备所在的总线： ● 所有外设、FLASH 以及 SRAMA 挂在 AHB1 总线 ● SRAMB 以及 SRAMC 挂在 AHB2 总线
[24-23]	DMS	R/W	0x0	目标端外设总线选择控制位 (Destination Master Select) 00：AHB Master1 01：AHB Master2 1x：Reserved 注意：参考目标设备所在的总线 ● 所有外设、FLASH 以及 SRAMA 挂在 AHB1 总线 ● SRAMB 以及 SRAMC 挂在 AHB2 总线
[22-20]	TTC	R/W	0x0	传输类型控制位 (Transfer Type Controller) 00：存储设备到存储设备(Memory To Memory) 01：存储设备到外设(Memory To Peripheral) 10：外设到存储设备(Peripheral To Memory) 11：外设到外设(Peripheral To Peripheral) 说明： ● FLASH/SRAMA/SRAMB/SRAMC 都属于存储设备(Memory) ● 其他均支持 DMA 传输的模块属于外设(例如 I2C、UART)
[19-17]	Reserved	Reserved	Reserved	Reserved
[16-14]	SBTL	R/W	0x2	一笔源 Burst 事务的长度配置 (Source Burst Transaction Length) 000：1 001：4 010：8 其他：保留 注意：一笔 Burst 事务传输的数据总量 = Burst 事务长度 × 数据位宽
[13-11]	DBTL	R/W	0x0	一笔目标 Burst 事务的长度配置 (Destination Burst Transaction Length) 000：1 001：4 010：8 其他：保留 注意：一笔 Burst 事务传输的数据总量 = Burst 事务长度 × 数据位宽

[10-9]	SINC	R/W	0x0	源端地址增长模式选择 (Source address Increase select) 00：递增 01：递减 1x：不变 注意：该位指示 DMA 传输过程中，地址是否随 DMA 数据传输自动进行递增、递减或者保持不变。例如：对于外设，数据地址一般为外设的寄存器地址，寄存器地址是固定不变的，则配置为“不变”。
[8-7]	DINC	R/W	0x0	目标地址增长模式选择 (Destination address Increase select) 00：递增 01：递减 1x：不变 注意：该位指示 DMA 传输过程中，地址是否随 DMA 数据传输自动进行递增、递减或者保持不变。例如：对于外设，数据地址一般为外设的寄存器地址，寄存器地址是固定不变的，则配置为“不变”。
[6-4]	STW	R/W	0x0	源数据位宽设置 (Source Transfer Width) 000：8bit 数据宽度 001：16bit 数据宽度 010：32bit 数据宽度 其他：无效值 注意：软件需确保该位不被配置为“其他”值，避免未知的异常。
[3-1]	DTW	R/W	0x0	目标数据位宽设置 (Destination Transfer Width) 000：8bit 数据宽度 001：16bit 数据宽度 010：32bit 数据宽度 其他：无效 注意：软件需确保该位不被配置为“其他”值，避免未知的异常。
[0]	CHIE	R/W	0x1	通道中断使能 (Channel Interrupt Enable) 0：屏蔽该 DMA 通道中断 1：使能该 DMA 通道中断 注意： ● 该位置 1 后，还需要配置中断掩码相关的寄存器(DMA_xIMR)使能相应的中断触发事件 ● 状态寄存器(DMA_xSR)仍然有效

11.5.2.4 DMA 通道 n 控制寄存器 1 (DMA_CH_CR1)

- **Name**：DMA Channel n Control Register1 (n=0 ... 1)
- **Size**：32bits
- **Offset**：0x1C + (n * 0x58)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12]	DONE	R/W	0x0	DMA 块传输完成标志位 (DMA Block Transfer Done bit) 0：传输未完成 1：传输完成 软件可通过查询该位是否被置 1，检查块传输(Block Transfer)是否完成； 软件可通过向该位写 0 清除该标志位。
[11-0]	BTCNT	R/W	0x2	DMA 块传输的数据量 (DMA Block Transfer Count) 通过配置该位域，设置一次 DMA 块传输需要传输的数据量。 说明： ● 一次 DMA 块传输数据总量 = 块传输数据量 × 数据位宽 ● 一次 DMA 块(Block)传输会根据实际数据量以及 Burst 相关的配置，可能会被分为多个 Burst 事务在总线上进行传输

11.5.2.5 DMA 通道 n 控制寄存器 2 (DMA_CH_CR2)

- **Name**：DMA Channel n Control Register2 (n=0 ... 1)
- **Size**：32bits
- **Offset**：0x40 + (n * 0x58)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-30]	Reserved	Reserved	Reserved	Reserved
[29-20]	MBL	R/W	0x0	DMA 最大的 Burst 长度设置 (DMA Max Burst Length) 说明： ● 该位域用于限制通道的 DMA 传输最大 Burst 长度，避免 DMA 长时间占用总线 ● 将该位域配置位 0，则意味软件不限制 Burst 长度，由 DMA 控制器根据 Burst 相关配置自动设置
[19-12]	Reserved	Reserved	Reserved	Reserved
[11]	SHSM	R/W	0x1	源端握手模式 (Source Handshaking Mode) 当源端配置为外设(Peripheral)时，该位需置 0； 当源端配置为存储设备(Memory)，该位设置无效。
[10]	DHSM	R/W	0x1	目标端握手模式 (Destination Handshaking Mode) 当目标端配置为外设(Peripheral)时，该位需置 0； 当目标端配置为存储设备(Memory)，该位设置无效。
[9]	FIFO_EF	R	0x1	通道 FIFO 为空的指示位 (Channel FIFO Empty Flag) 通过读取该标志位获知当前 DMA 通道的 FIFO 是否还有数据。 1：通道 FIFO 为空 0：通道 FIFO 不为空 说明：可用于与 SUSP 位结合使用，用于暂停传输而不丢失数据。
[8]	SUSP	R/W	0x0	通道暂停控制 (Channel Suspend) 0：不暂停 1：暂停 DMA 源(Source)传输 说明： ● 该位置 1 后，将会暂停 DMA 从源端获取数据，直到该位重新置 0。 ● 由于暂停时无法保证当前事务已经完成，可以与 FIFO_EF 结合使用，以彻底禁用通道而不丢失任何数据，步骤如下： 1. 先将 SUSP 置 1，暂停从源端获取数据到 FIFO； 2. 软件持续检查 FIFO_EF 位，直到该位被硬件置 1； 3. 之后软件可以设置 DMA_CHER 寄存器，关闭相应的 DMA 实现暂停 DMA 的传输。 ● 需要退出暂停状态，继续之前的传输，步骤如下： 1. 配置 DMA_CHER 寄存器，使能相应的 DMA 通道； 2. 将 SUSP 置 0，之后 DMA 将回复正常方式传输。
[7-6]	Reserved	Reserved	Reserved	Reserved

[5]	PRI	R/W	CHn	通道优先级设置 (Channel Priority) 0：低优先级 1：高优先级 说明： ● 该位的初始值为相应的通道号，例如通道 1 该位的初始值即为 1 ● 如果同时收到两个通道的传输请求，通过优先级仲裁决定先响应优先级高的通道；如果通道优先级配置一样，则先处理通道号小的通道请求(通道 0) ● 如果正在执行优先级低的 DMA 传输请求，DMA 控制器将先完成低优先级的 burst 事务后，再切换处理高优先级请求。
[4-0]	Reserved	Reserved	Reserved	Reserved

11.5.2.6 DMA 通道 n 控制寄存器 3 (DMA_CH_CR3)

- **Name**：DMA Channel n Control Register3 (n=0 ... 1)
- **Size**：32bits
- **Offset**：0x44 + (n * 0x58)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-15]	Reserved	Reserved	Reserved	Reserved
[14-11]	DHSIF	R/W	0x0	目标端握手接口设置 (Destination Handshaking Interface) 0000：I2C0_Tx 0001：I2C0_Rx 0010：I2C1_Tx 0011：I2C1_Rx 1000：UART0_Tx 1001：UART0_Rx 1010：UART1_Tx 1011：UART1_Rx 其他值：保留，设置可能导致未知的异常 说明：如果目标端为 memory，则该设置无效。
[10-7]	SHSIF	R/W	0x0	源端握手接口设置 (DMA Source Handshaking Interface) 0000：I2C0_Tx 0001：I2C0_Rx 0010：I2C1_Tx 0011：I2C1_Rx 1000：UART0_Tx 1001：UART0_Rx 1010：UART1_Tx 1011：UART1_Rx 其他值：保留，设置可能导致未知的异常 说明：如果源端为 memory，则该设置无效。
[6-2]	Reserved	Reserved	Reserved	Reserved
[1]	FMD	R/W	0x0	FIFO 模式选择 (FIFO Mode select) 该位段决定 FIFO 内需要由多少剩余空间或有效数据，才能处理一次 Burst 事务传输。 0：剩余空间/有效数据满足一次源端/目标端的数据位宽 1：对于源端剩余空间超过 FIFO 深度一半时，且对于目标端可用数据超过 FIFO 深度的一半时（Burst 传输请求结束或块传输结束时不受此限制，剩余空间/数据将直接触发最后的传输事务）
[0]	FCMD	R/W	0x0	DMA 流控制模式 (DMA Channel Flow Control Mode) 0：源事务请求，启动数据预取 1：目的事务请求发生后，才开始源事务请求

11.5.2.7 DMA 传输状态寄存器 (DMA_TSR)

- **Name**：DMA Transfer Status Register
- **Size**：32bits
- **Offset**：0x2C0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	TS[n]	R	0x0	DMA 传输完成状态 (DMA Transfer Status) 该位不受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 传输未完成 1：DMA 传输完成 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除状态。

11.5.2.8 DMA 块传输状态寄存器 (DMA_BTSR)

- **Name**：DMA Block Transfer Status Register
- **Size**：32bits
- **Offset**：0x2C8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	BTS[n]	R	0x0	DMA 块传输完成状态 (DMA Block Transfer Status) 该位不受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 块传输未完成 1：DMA 块传输完成 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除状态。

11.5.2.9 DMA 源端传输状态寄存器 (DMA_STSR)

- **Name**：DMA Source Transfer Status Register
- **Size**：32bits
- **Offset**：0x2d0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	STS[n]	R	0x0	DMA 源端传输完成状态 (DMA Source Transfer Status) 该位不受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 源传输未完成 1：DMA 源传输完成 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除状态。

11.5.2.10 DMA 目标端传输状态寄存器 (DMA_DTSR)

- **Name**：DMA Destination Transfer Status Register
- **Size**：32bits
- **Offset**：0x2d8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	DTS[n]	R	0x0	DMA 目标端传输完成状态 (DMA Destination Transfer Raw Status) 该位不受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 目标端传输未完成 1：DMA 目标端传输完成 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除状态。

11.5.2.11 DMA 传输错误状态寄存器 (DMA_TESR)

- **Name：Name：**DMA Transfer Error Status Register
- **Size：**32bits
- **Offset：**0x2E0
- **Default：**0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	TES[n]	R	0x0	DMA 传输错误状态 (DMA Transfer Error Status) 该位不受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 传输正常 1：DMA 传输错误 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除状态。

11.5.2.12 DMA 传输中断挂起寄存器 (DMA_TIPR)

- **Name：**DMA Transfer Interrupt Pending Register
- **Size：**32bits
- **Offset：**0x2E8
- **Default：**0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	TIF[n]	R	0x0	DMA 传输完成中断挂起标志位 (DMA Transfer Interrupt pending Flag) 该位受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 传输未完成 1：DMA 传输完成 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除中断挂起标志。

11.5.2.13 DMA 块传输中断挂起寄存器 (DMA_BTIPR)

- **Name**：DMA Block Transfer Interrupt Pending Register
- **Size**：32bits
- **Offset**：0x2F0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	BTIF[n]	R	0x0	DMA 块传输完成中断挂起标志位 (DMA Block Transfer Interrupt pending Flag) 该位受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 块传输未完成 1：DMA 块传输完成 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除中断挂起标志。

11.5.2.14 DMA 源端传输中断挂起寄存器 (DMA_STIPR)

- **Name**：DMA Source Transfer Interrupt Pending Register
- **Size**：32bits
- **Offset**：0x2F8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	STIF[n]	R	0x0	DMA 源端传输完成中断挂起标志位 (DMA Source Transfer Interrupt pending Flag) 该位受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 源端传输未完成 1：DMA 源端传输完成 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除中断挂起标志。

11.5.2.15 DMA 目标端传输中断挂起寄存器 (DMA_DTIPR)

- **Name**：DMA Destination Transfer Interrupt Pending Register
- **Size**：32bits
- **Offset**：0x300
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	DTIF[n]	R	0x0	DMA 目标端传输完成中断挂起标志位 (DMA Destination Transfer Interrupt pending Flag) 该位受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 目标端传输未完成 1：DMA 目标端传输完成 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除中断挂起标志。

11.5.2.16 DMA 传输错误中断挂起寄存器 (DMA_TEIPR)

- **Name**：DMA Transfer Error Interrupt Pending Register
- **Size**：32bits
- **Offset**：0x308
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	TEIF[n]	R	0x0	DMA 传输错误中断挂起标志位 (DMA Transfer Error Interrupt pending Flag) 该位受中断掩码寄存器 DMA_xIMR 影响，每个位分别对应一个通道。 0：DMA 传输正常 1：DMA 传输错误 说明：通过向相应的 DMA_xCR 中的相应通道位写 1 清除中断挂起标志。

11.5.2.17 DMA 传输中断掩码寄存器 (DMA_TIMR)

- **Name**：DMA Transfer Interrupt Mask Register
- **Size**：32bits
- **Offset**：0x310
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-10]	Reserved	Reserved	Reserved	Reserved
[9-8]	TIWE[n]	W	0x0	DMA 传输完成中断掩码写使能 (DMA Transfer Interrupt mask Write Enable) 每个位分别对应一个通道。 0：关闭写使能 1：开启写使能
[7-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	TIE[n]	R/W	0x0	DMA 传输完成中断使能位 (DMA Transfer Interrupt Enable) 每个位分别对应一个通道。 0：关闭 1：使能 注意：设置中断使能位时，需开启掩码写使能，否则配置无效。

11.5.2.18 DMA 块传输中断掩码寄存器 (DMA_BTIMR)

- **Name**：DMA Block Transfer Interrupt Mask Register
- **Size**：32bits
- **Offset**：0x318
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-10]	Reserved	Reserved	Reserved	Reserved
[9-8]	BTIWE[n]	W	0x0	DMA 块传输完成中断掩码写使能 (DMA Block Transfer Interrupt mask Write Enable) 每个位分别对应一个通道。 0：关闭写使能 1：开启写使能
[7-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	BTIE[n]	R/W	0x0	DMA 块传输完成中断使能位 (DMA Block Transfer Interrupt Enable) 每个位分别对应一个通道。 0：关闭 1：使能 注意：设置中断使能位时，需开启掩码写使能，否则配置无效。

11.5.2.19 DMA 源端传输中断掩码寄存器 (DMA_STIMR)

- **Name**：DMA Source Transfer Interrupt Mask Register
- **Size**：32bits
- **Offset**：0x320
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-10]	Reserved	Reserved	Reserved	Reserved
[9-8]	STIWE[n]	W	0x0	DMA 源端传输完成中断掩码写使能 (DMA Source Transfer Interrupt mask Write Enable) 每个位分别对应一个通道。 0：关闭写使能 1：开启写使能
[7-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	STIE[n]	R/W	0x0	DMA 源端传输完成中断使能位 (DMA Source Transfer Interrupt Enable) 每个位分别对应一个通道。 0：关闭 1：使能 注意：设置中断使能位时，需开启掩码写使能，否则配置无效。

11.5.2.20 DMA 目标端传输中断掩码寄存器 (DMA_DTIMR)

- **Name**：DMA Destination Transfer Interrupt Mask Register
- **Size**：32bits
- **Offset**：0x328
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-10]	Reserved	Reserved	Reserved	Reserved
[9-8]	DTIWE[n]	W	0x0	DMA 目的端传输完成中断掩码写使能 (DMA Destination Transfer Interrupt mask Write Enable) 每个位分别对应一个通道。 0：关闭写使能 1：开启写使能
[7-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	DTIE[n]	R/W	0x0	DMA 目的端传输完成中断使能位 (DMA Destination Transfer Interrupt Enable) 每个位分别对应一个通道。 0：关闭 1：使能 注意：设置中断使能位时，需开启掩码写使能，否则配置无效。

11.5.2.21 DMA 传输错误中断掩码寄存器 (DMA_TEIMR)

- **Name**：DMA Transfer Error Interrupt Mask Register
- **Size**：32bits
- **Offset**：0x330
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-10]	Reserved	Reserved	Reserved	Reserved
[9-8]	TEIWE[n]	W	0x0	DMA 传输错误中断掩码写使能 (DMA Transfer Error Interrupt mask Write Enable) 每个位分别对应一个通道。 0：关闭写使能 1：开启写使能
[7-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	TEIE[n]	R/W	0x0	DMA 传输错误中断使能位 (DMA Transfer Error Interrupt Enable) 每个位分别对应一个通道。 0：关闭 1：使能 注意：设置中断使能位时，需开启掩码写使能，否则配置无效。

11.5.2.22 DMA 传输状态清除寄存器 (DMA_TCR)

- **Name**：DMA Transfer Clear Register
- **Size**：32bits
- **Offset**：0x338
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	TC[n]	W	0x0	DMA 传输完成状态清除 (DMA Transfer Clear) 每个位分别对应一个通道。 0：无效 1：清除标志位 说明：将会清除 DMA_xSR 和 DMA_xIPR 寄存器中对应通道的标志位。

11.5.2.23 DMA 块传输状态清除寄存器 (DMA_BTCR)

- **Name**：DMA Block Transfer Clear Register
- **Size**：32bits
- **Offset**：0x340
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	BTC[n]	W	0x0	DMA Block 传输状态清除 (DMA Block Transfer Clear) 每个位分别对应一个通道。 0：无效 1：清除标志位 说明：将会清除 DMA_xSR 和 DMA_xIPR 寄存器中对应通道的标志位。

11.5.2.24 DMA 源端传输状态清除寄存器 (DMA_STCR)

- **Name**：DMA Source Transfer Clear Register
- **Size**：32bits
- **Offset**：0x348
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	STC[n]	W	0x0	DMA 源端传输状态清除 (DMA Source Transfer Clear) 每个位分别对应一个通道。 0：无效 1：清除标志位 说明：将会清除 DMA_xSR 和 DMA_xIPR 寄存器中对应通道的标志位。

11.5.2.25 DMA 目标端传输状态清除寄存器 (DMA_DTCR)

- **Name**：DMA Destination Transfer Clear Register
- **Size**：32bits
- **Offset**：0x350
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	DTC[n]	W	0x0	DMA 目的端传输状态清除 (DMA Destination Transfer Clear) 每个位分别对应一个通道。 0：无效 1：清除标志位 说明：将会清除 DMA_xSR 和 DMA_xIPR 寄存器中对应通道的标志位。

11.5.2.26 DMA 传输错误状态清除寄存器 (DMA_TECR)

- **Name**：DMA Transfer Error Clear Register
- **Size**：32bits
- **Offset**：0x358
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	TEC[n]	W	0x0	DMA 传输错误状态清除 (DMA Transfer Error Clear) 每个位分别对应一个通道。 0：无效 1：清除标志位 说明：将会清除 DMA_xSR 和 DMA_xIPR 寄存器中对应通道的标志位。

11.5.2.27 DMA 控制寄存器 0 (DMA_CR0)

- **Name**：DMA Control Register0
- **Size**：32bits
- **Offset**：0x398
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	PEN	R/W	0x0	DMA 外设使能控制位 (DMA Peripheral Enable) 0：禁用 1：使能 说明：建议在 DMA 开始传输前，先完成相应 DMA 的源/目标外设配置。

11.5.2.28 DMA 控制寄存器 1 (DMA_CR1)

- **Name**：DMA Control Register1
- **Size**：32bits
- **Offset**：0x3A0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-10]	Reserved	Reserved	Reserved	Reserved
[9:8]	CHWE[n]	R/W	0x0	通道写入使能控制位 (DMA Channel Write Enable) 每个位分别对应一个通道。 0：关闭通道写入使能 1：开启通道写入使能
[7-2]	Reserved	Reserved	Reserved	Reserved
[1:0]	CHEN[n]	R/W	0x0	通道使能控制位 (DMA Channel Enable) 每个位分别对应一个通道。 0：禁用通道 1：启用通道 说明：设置通道使能位时，需开启通道写使能，否则配置无效。

12 中断和事件

12.1 嵌套向量中断控制器（NVIC）

12.1.1 NVIC 特性

嵌套向量中断控制器 NVIC 包含以下特性：

- 芯片具有 51 个可屏蔽中断通道（不包括 Cortex™-M3 的 16 根中断线）
- 8 个可编程优先级（使用了 3 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，请参考《ARM Cortex™-M3 技术参考手册》中的第 12.1.3 节：中断和异常向量和第 12.1 节：嵌套向量中断控制器。

12.1.2 Sys Tick 校准值寄存器

SysTick 校准值设置为 11250，当 SysTick 时钟设置为 11.25MHz（HCLK/8，HCLK 设为 90MHz），会产生 1 ms 时间基准。

12.1.3 中断和异常向量

请参见表 12-1，了解芯片的中断向量表。

表 12-1 芯片的中断向量表

位置	优先级	优先级类型	名称	说明	地址
	-	-	-	保留	0x0000 0000
	-3	固定	Reset	复位	0x0000 0004
	-2	固定	NMI	不可屏蔽中断。RCU 时钟安全系统(CSS)连接到 NMI 向量。	0x0000 0008
	-1	固定	HardFault	所有类型的错误	0x0000 000C
	0	可设置	MemManage	存储器管理	0x0000 0010
	1	可设置	BusFault	预取指失败，存储器访问失败	0x0000 0014
	2	可设置	UsageFault	未定义的指令或非法状态	0x0000 0018
	-	-	-	保留	0x0000 001C - 0x0000 002B

	3	可设置	SVCall	通过 SWI 指令调用的系统服务	0x0000 002C
	4	可设置	Debug Monitor	调试监控器	0x0000 0030
	-	-	-	保留	0x0000 0034
	5	可设置	PendSV	可挂起的系统服务	0x0000 0038
	6	可设置	SysTick	系统嘀嗒定时器	0x0000 003C
0	7	可设置	I2C0	I2C0 全局中断	0x0000 0040
1	8	可设置	I2C1	I2C1 全局中断	0x0000 0044
2	9	可设置	UART0	UART0 全局中断	0x0000 0048
3	10	可设置	UART1	UART1 全局中断	0x0000 004C
4	11	可设置	TMR0	TMR0 全局中断	0x0000 0050
5	12	可设置	TMR1	TMR1 全局中断	0x0000 0054
6	13	可设置	TMR2	TMR2 全局中断	0x0000 0058
7	14	可设置	TMR3	TMR3 全局中断	0x0000 005C
8	15	可设置	LVD	连接到 EXTI 线的可编程低电压检测(PVD)中断	0x0000 0060
9	16	可设置	HSTMR0	HSTMR 0 全局中断	0x0000 0064
10	17	可设置	HSTMR1	HSTMR 1 全局中断	0x0000 0068
11	18	可设置	HSTMR2	HSTMR 2 全局中断	0x0000 006C
12	19	可设置	HSTMR3	HSTMR 3 全局中断	0x0000 0070
13	20	可设置	IWDG	独立看门狗中断	0x0000 0074
14	21	可设置	WWDG	窗口看门狗中断	0x0000 0078
15	22	可设置	IIR0	IIR0 全局中断	0x0000 007C
16	23	可设置	IIR1	IIR1 全局中断	0x0000 0080
17	24	可设置	IIR2	IIR2 全局中断	0x0000 0084
18	25	可设置	IIR3	IIR3 全局中断	0x0000 0088
19	26	可设置	IIR4	IIR4 全局中断	0x0000 008C
20	27	可设置	ECU	ECU 全局中断	0x0000 0090
21	28	可设置	DMAC	DMAC 全局中断	0x0000 0094
22	29	可设置	CAN	CAN 中断	0x0000 0098
23	30	可设置	GPIOA	GPIOA 中断	0x0000 009C
24	31	可设置	GPIOB	GPIOB 中断	0x0000 00A0
25	32	可设置	GPIOC	GPIOC 中断	0x0000 00A4
26	33	可设置	GPIOD	GPIOD 中断	0x0000 00A8
27	34	可设置	EFLASH	EFLASH 全局中断	0x0000 00AC
28	35	可设置	DFLASH	DFLASH 全局中断	0x0000 00B0
29	36	可设置	HRPWM_Master	HRPWM 主定时器中断	0x0000 00B4
30	37	可设置	HRPWM_Slave0	HRPWM0 从定时器中断	0x0000 00B8
31	38	可设置	HRPWM_Slave1	HRPWM1 从定时器中断	0x0000 00BC
32	39	可设置	HRPWM_Slave2	HRPWM2 从定时器中断	0x0000 00C0
33	40	可设置	HRPWM_Slave3	HRPWM3 从定时器中断	0x0000 00C4
34	41	可设置	HRPWM_Slave4	HRPWM4 从定时器中断	0x0000 00C8
35	42	可设置	HRPWM_Slave5	HRPWM5 从定时器中断	0x0000 00CC
36	43	可设置	HRPWM_Fault	HRPWM 故障中断	0x0000 00D0

37	44	可设置	ADC0_Normal	ADC0 常规中断	0x0000 00D4
38	45	可设置	ADC0_Half	ADC0 DMA 半满中断	0x0000 00D8
39	46	可设置	ADC0_Full	ADC0 DMA 全满中断	0x0000 00DC
40	47	可设置	ADC0_Sample	ADC0 采样中断	0x0000 00E0
41	48	可设置	ADC1_Normal	ADC1 常规中断	0x0000 00E4
42	49	可设置	ADC1_Half	ADC1 DMA 半满中断	0x0000 00E8
43	50	可设置	ADC1_Full	ADC1 DMA 全满中断	0x0000 00EC
44	51	可设置	ADC1_Sample	ADC1 采样中断	0x0000 00F0
45	52	可设置	DAC	DAC 全局中断	0x0000 00F4
46	53	可设置	CMP	CMP 全局中断	0x0000 00F8
47	54	可设置	USB_CTL_INT	USB 控制中断	0x0000 00FC
48	55	可设置	USB_SOF_INT	USB 插入拔出中断	0x0000 0100
49	56	可设置	USB_LPM_INT	USB 电源管理中断	0x0000 0104
50	57	可设置	USB_EPI_INT	USB 端点中断	0x0000 0108

12.1.4 唤醒事件

芯片能够处理外部或内部事件来唤醒内核(WFE)。唤醒事件可通过以下方式产生：

- 在外设的控制寄存器使能一个中断，但不在 NVIC 中使能，同时使能 Cortex™-M3 系统控制寄存器中的 SEVONPEND 位。当 CPU 从 WFE 恢复时，需要清除相应外设的中断挂起位和外设 NVIC 中断通道挂起位（在 NVIC 中断清除挂起寄存器中）。
- 配置一个外部或内部中断为事件模式，当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置位，不必清除相应外设的中断挂起位或 NVIC 中断通道挂起位。

13 数字滤波单元（IIR）

13.1 简介

IIR 采用动态结构设计，可以软件配置 1/2/3/4 阶结构，系数 Bx/Ax 都是 16bit 位宽，输入和输出数据是 16 bit 位宽。

系统有 5 个 IIR，5 个 IIR 可以同时运行，每个 IIR 都有独立的配置信息和系数信息。

13.2 结构框图

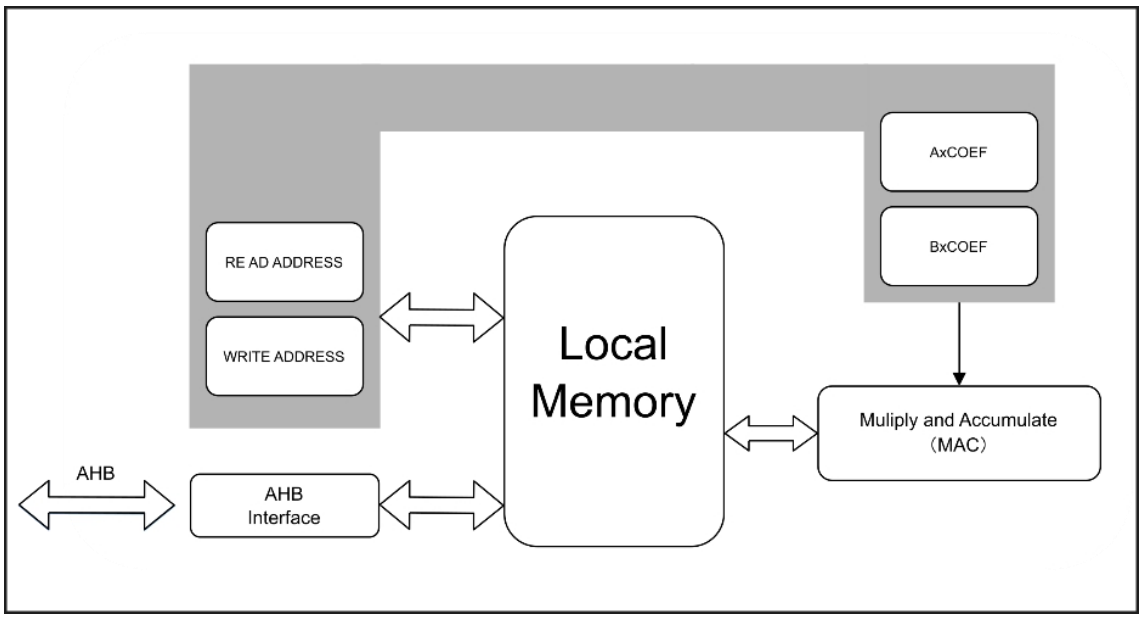


图 13-1 IIR 结构框图

13.3 主要特性

IIR 主要具有以下特性：

- 阶数(1-4)动态可调
- DISCAL 放大支持 2^16，FBSCAL 及 DOSCAL 支持 2^{32}
- 支持影子寄存器及重加载功能
- 支持内部软复

13.4 功能描述

13.4.1 IIR Filter

IIR 滤波器的公式如下：

$$y_n = \sum_{i=0}^{N} b_i x_{n-i} + \sum_{i=1}^{M} a_i y_{n-i}$$

定点化后的公式如下：

$$y'_n = \left(\sum_{i=0}^{N} 2^{scalei} B_i x_{n-i} + \sum_{i=1}^{M} 2^{-fedback} A_i y'_{n-i} \right)$$

$$y_n = 2^{-scaleo} y'_n$$

备注：会提供一个工具将浮点系数b_i和a_i转成定点系数B_i和A_i。

13.4.2 架构描述

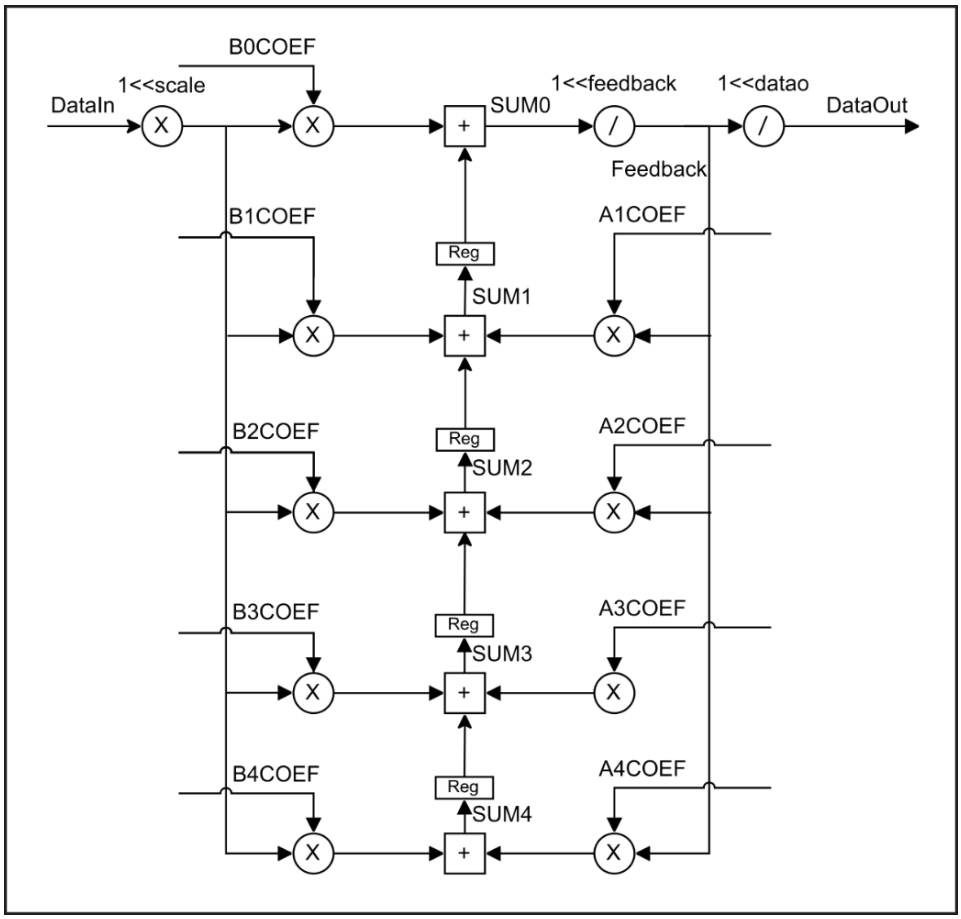


图 13-2 架构描述图（reg->z^-1）

输入：

- 输入数据 DataIn 最大支持长度为 d
- 系数 BxCOEF 最大支持长度为 5
- 系数 AxCOEF 最大支持长度为 4

输出：

- 输出数据 DataOut 与 DataIn 是一一对应的

参数：

- Data In/Out 位宽均为 16bit 位宽
- 参数 AxCOEF 及 BxCOEF 位宽均为 16bit 位宽
- Data In Scale 放大最大支持 2^{16}
- Data Out Scale 缩小及 Feed Back Scale 缩小最大支持 2^{31}

13.5 寄存器描述

13.5.1 寄存器列表

Name	Offset	Width	Description
IIRn_CR0	0x00	32bits	IIRn 控制寄存器 0
IIRn_CR1	0x04	32bits	IIRn 控制寄存器 1
IIRn_IER	0x08	32bits	IIRn 中断使能寄存器
IIRn_ISR	0x0C	32bits	IIRn 中断状态寄存器
IIRn_DOR	0x14	32bits	IIRn 数据输出寄存器
IIRn_DMACR	0x20	32bits	IIRn DMA 控制寄存器
IIRn_DIAR	0x24	32bits	IIRn 数据输入地址寄存器
IIRn_DOAR	0x28	32bits	IIRn 数据输出地址寄存器
IIRn_SCALR	0x2C	32bits	IIRn 缩放寄存器
IIRn_BxCOEFR	0x30/0x34/0x38/0x3C/0x40	32bits	IIRn BxCOEF 寄存器
IIRn_AxCOEFR	0x44/0x48/0x4C/0x50	32bits	IIRn AxCOEF 寄存器
IIRn_DMACSR	0x60	32bits	IIRn DMA 控制影子寄存器
IIRn_DIASR	0x64	32bits	IIRn 数据输入地址影子寄存器
IIRn_DOASR	0x68	32bits	IIRn 数据输出地址影子寄存器
IIRn_SCALS	0x6C	32bits	IIRn 分频影子寄存器
IIRn_BxCOEFSR	0x70/0x74/0x78/0x7C/0x80	32bits	IIRn BxCOEF 影子寄存器
IIRn_AxCOEFSR	0x84/0x88/0x8C/0x90	32bits	IIRn AxCOEF 影子寄存器

13.5.2 寄存器详细描述

13.5.2.1 IIRn 控制寄存器 0 (IIRn_CR0)

- **Name**：IIRn Control Register0 (n = 0 … 4)
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x10

Bit	Field	R/W	Default	Description
[31-6]	Reserved	Reserved	Reserved	Reserved
[5-4]	ORD	R/W	0x1	IIR 阶数选择 (IIR Order selection) 0：1 阶 1：2 阶 2：3 阶 3：4 阶 注意：如果需要在中途切换阶数，则配置时按最大阶数配置，通过调整系数来实现低阶计算；计算延迟按则最大阶数计算。
[3-2]	Reserved	Reserved	Reserved	Reserved
[1]	IBRST	R/W1C	0x0	IIR 内部缓存复位控制 (IIR Internal Buffer Reset control) 内部缓存数据复位控制，用于清除 IIR 累计的计算缓存，重新开始运算。 0：无影响 1：复位缓存
[0]	IIREN	R/W	0x0	IIR 使能控制位 0：不使能 1：使能

13.5.2.2 IIRn 控制寄存器 1 (IIRn_CR1)

- **Name**：IIRn Control Register1 (n = 0 … 4)
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1]	ARE	R/W	0x0	IIR 自动装在使能控制 (IIR Auto-Reload Enable) 使能自动装载功能，可以将影子寄存器(寄存器偏移地址从 0x60 到 0x90)的值，加载到所对应的寄存器中(寄存器偏移地址从 0x20 到 0x50)。 0：不使能 1：使能
[0]	START	R/WAC	0x0	IIR 启动控制位 (IIR Start Control) 0：不启动 1：启动 IIR

13.5.2.3 IIRn 中断使能寄存器 (IIRn_IER)

- **Name**：IIRn Interrupt Enable Register (n = 0 … 4)
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	IE	R/W	0x0	IIR 中断使能位 (IIR Interrupt Enable) 0：不使能 1：使能

13.5.2.4 IIRn 中断状态寄存器 (IIRn_ISR)

- **Name**：IIRn Interrupt Status Register (n = 0 … 4)
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	FDIF	R/W1C	0x0	IIR 的滤波完成中断标志位 (IIR Filter Done Interrupt Flag) 0：No pending 1：Pending 当 IIR 完成一次滤波后，硬件自动置 1，并触发中断。

13.5.2.5 IIRn 数据输出寄存器 (IIRn_DOR)

- **Name**：IIRn Data Output Register (n = 0 … 4)
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	DATA	R	0x0	IIR 数据输出寄存器

13.5.2.6 IIRn DMA 控制寄存器 (IIRn_DMACR)

- **Name**：IIRn DMA Control Register (n = 0 … 4)
- **Size**：32bits
- **Offset**：0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	DMAEN	R/W	0x0	IIR DMA 模式使能控制 (IIR DMA Mode Enable control) 使能 DMA 模式，IIR 滤波完成后，输出数据将自动输出到 IIRn_DOAR 寄存器配置的 RAM 空间地址内； 不使能 DMA 模式，则需通过读取 IIRn_DOR 寄存器方式获取输出的数据。 0：不使能 1：使能

13.5.2.7 IIRn 数据输入地址寄存器 (IIRn_DIAR)

- **Name**：IIRn Data Input Address Register (n = 0...4)
- **Size**：32bits
- **Offset**：0x24
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DIADDR	R/W	0x0	IIR DMA 数据输入地址 (IIR DMA Data Input Address) IIR 输入数据 RAM 空间地址，IIR 将从该 RAM 空间地址直接获取数据。 注意： 1. RAM 空间地址必需按 16bit 对齐； 2. 指定的 RAM 空间必须位于 SRAMB 或 SRAMC 内，具体请参看“系统框架设计 一 总线连接表”。

13.5.2.8 IIRn 数据输出地址寄存器 (IIRn_DOAR)

- **Name**：IIRn Data Output Address Register (n = 0...4)
- **Size**：32bits
- **Offset**：0x28
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DOADDR	R/W	0x0	IIR DMA 数据输出地址 (IIR DMA Data Output Address) IIR 输出数据 RAM 的存放地址，仅用于使能 DMA 模式。 使能 DMA 模式后，IIR 将滤波后的数据直接输出到该 RAM 空间地址内。 注意： 1. RAM 空间地址必需按 16bit 对齐； 2. 指定的 RAM 空间必须位于 SRAMB 或 SRAMC 内，具体请参看“系统框架设计 一 总线连接表”。

13.5.2.9 IIRn 缩放寄存器 (IIRn_SCALR)

- **Name**：IIRn Scale Register (n = 0...4)
- **Size**：32bits
- **Offset**：0x2C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-21]	Reserved	Reserved	Reserved	Reserved
[20-16]	DOSCAL	R/W	0x0	IIR DataOut Scale Register: IIR 数据 DataOut 缩小寄存器，最大支持 2^31 缩小，缩小倍数如下： 0：不缩小 1：2^1 2：2^2 …… 31：2^31
[15-13]	Reserved	Reserved	Reserved	Reserved
[12-8]	FBSCAL	R/W	0x0	IIR FeedBack Scale Register: IIR 数据 FeedBack 缩小寄存器，最大支持 2^31 缩小，缩小倍数如下： 0：不缩小 1：2^1 2：2^2 …… 31：2^31
[7-5]	Reserved	Reserved	Reserved	Reserved
[4-0]	DISCAL	R/W	0x0	IIR DataIn Scale Register: IIR 输入数据放大寄存器，最大支持 2^16 放大，放大倍数如下： 0：不放大 1：2^1 2：2^2 …… 16~31：2^16 注意：16~31 均为放大 2^16，最大支持放大 2^16。

13.5.2.10 IIRn BxCOEF 寄存器 (IIRn_BxCOEFR)

- **Name**：IIRn BxCOEF Register (n = 0...4) (x = 0...4)
- **Size**：32bits
- **Offset**：0x30/0x34/0x38/0x3C/0x40
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	BxCOEF	R/W	0x0	IIR BxCOEF Register (x = 0...4)

13.5.2.11 IIRn AxCOEF 寄存器 (IIRn_AxCOEFR)

- **Name**：IIRn AxCOEF Register (n = 0...4)(x = 1...4)
- **Size**：32bits
- **Offset**：0x44/0x48/0x4C/0x50
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	AxCOEF	R/W	0x0	IIR AxCOEF Register (x = 1...4)

13.5.2.12 IIRn DMA 控制影子寄存器 (IIRn_DMACSR)

- **Name**：IIRn DMA Control Shadow Register (n = 0...4)
- **Size**：32bits
- **Offset**：0x60
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	DMAENS	R/W	0x0	IIRn_DMACR 的影子寄存器配置，位段定义请参考所对应的寄存器描述; 影子寄存器的功能，请参考 IIRn_CR1 寄存器 ARE 位的功能描述。

13.5.2.13 IIRn 数据输入地址影子寄存器 (IIRn_DIASR)

- **Name**：IIRn Data Input Address Shadow Register (n = 0...4)
- **Size**：32bits
- **Offset**：0x64
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DIADDRS	R/W	0x0	IIRn_DIAR 的影子寄存器配置，位段定义请参考所对应的寄存器描述; 影子寄存器的功能，请参考 IIRn_CR1 寄存器 ARE 位的功能描述。

13.5.2.14 IIRn 数据输出地址影子寄存器 (IIRn_DOASR)

- **Name**：IIRn Data Output Address Shadow Register (n = 0...4)
- **Size**：32bits
- **Offset**：0x68
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	DOADDRS	R/W	0x0	IIRn_DOAR 的影子寄存器配置，位段定义请参考所对应的寄存器描述; 影子寄存器的功能，请参考 IIRn_CR1 寄存器 ARE 位的功能描述。

13.5.2.15 IIRn 分频影子寄存器 (IIRn_SCALS)

- **Name**：IIRn Scale Shadow Register (n = 0...4)
- **Size**：32bits
- **Offset**：0x6C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-21]	Reserved	Reserved	Reserved	Reserved
[20-16]	DOSCALS	R/W	0x0	描述同下
[15-13]	Reserved	Reserved	Reserved	Reserved
[12-8]	FBSCALS	R/W	0x0	描述同下
[7-5]	Reserved	Reserved	Reserved	Reserved
[4-0]	DISCALS	R/W	0x0	IIRn_SCALR 的影子寄存器配置，位段定义请参考所对应的寄存器描述; 影子寄存器的功能，请参考 IIRn_CR1 寄存器 ARE 位的功能描述。

13.5.2.16 IIRn BxCOEF 影子寄存器 (IIRn_BxCOEFSR)

- **Name**：IIRn BxCOEF Shadow Register (n = 0...4) (x = 0...4)
- **Size**：32bits
- **Offset**：0x70/0x74/0x78/0x7C/0x80
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	BxCOEFS	R/W	0x0	IIRn_BxCOEF 的影子寄存器配置，位段定义请参考所对应的寄存器描述; 影子寄存器的功能，请参考 IIRn_CR1 寄存器 ARE 位的功能描述。

13.5.2.17 IIRn AxCOEF 影子寄存器 (IIRn_AxCOEFSR)

- **Name**：IIRn AxCOEF Shadow Register (n = 0...4)(x = 1...4)
- **Size**：32bits
- **Offset**：0x84/0x88/0x8C/0x90
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	AxCOEFS	R/W	0x0	IIRn_AxCOEF 的影子寄存器配置，位段定义请参考所对应的寄存器描述; 影子寄存器的功能，请参考 IIRn_CR1 寄存器 ARE 位的功能描述。

14 功率计量单元（ECU）

14.1 简介

ECU 模块支持一路电流有效值、一路电压有效值的测量，支持有功功率、无功功率、视在功率、功率因数以及基波频率等电能参数的测量，可以实现灵活的测量方案。

ECU 模块支持 DMA 功能，通过两个独立的 DMA 通路读取电流/电压数据。

ECU 模块支持全数字的增益校正、偏置校正与相位校正。

ECU 模块具有独立的开方功能，输入数据为 32 位。

14.2 结构框图

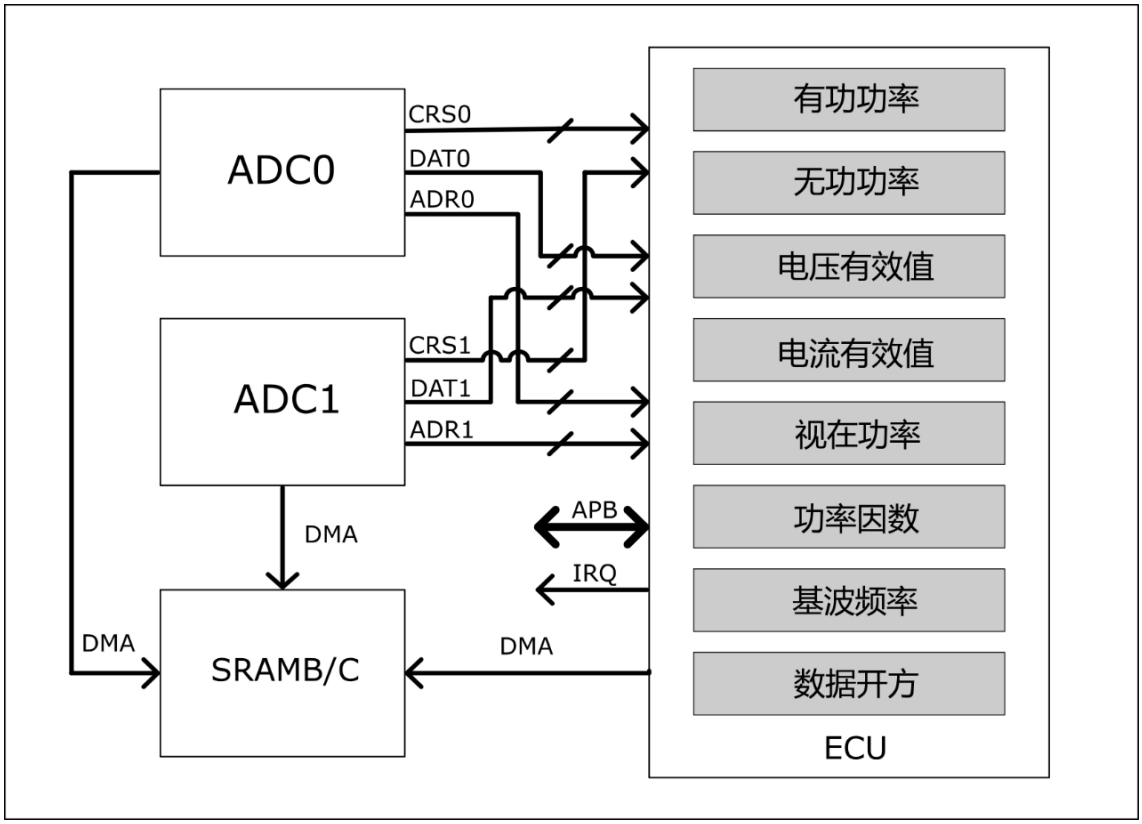


图 14-1 ECU 结构框图

14.3 主要特性

ECU 模块功能包括：

- 提供一路电流和一路电压有效值测量
- 提供有功功率测量，测量精度满足如下精度要求
  - >50%满载时；达到 0.5%精度；满载是指 ADC 满量程幅值*90%
  - 30%~50%，达到 1%精度
  - 10%~30%，达到 2.5%精度
- 提供无功功率测量，测量精度满足如下精度要求
  - >50%满载时；达到 0.5%精度；满载是指 ADC 满量程幅值*90%
  - 30%~50%，达到 1%精度
  - 10%~30%，达到 2.5%精度
- 提供视在功率、功率因数、基波频率等电能参数测量
- 提供增益校正、偏置校正、相位校正功能
- 提供独立开方运算功能

14.4 功能描述

14.4.1 模数转换

ECU 的模数转换功能通过 ADCx 模块实现，ECU 与 ADCx 模块连接关系如图 1 所示。

ADCx 模块将转换后的电流及电压数据通过 DMA 接口存入系统 Memory，与此同时 ADCx 模块为 ECU 提供过零事件、地址事件以及数据事件，ECU 根据 ADCx 模块提供的事件通过 DMA 接口读取系统 Memory，得到转换后的电流及电压数据，随后进行计算。

ADCx 模块总共支持采样 24 路外部信号，每路外部信号支持差分与单端方式输入，最大信号输入幅度为峰值 3.0V，ECU 模块测量所需的一路电流信号与一路电压信号包含在 ADC 的 24 路外部信号之中。

通过配置 ADCx 的采样序列可以得到 ECU 电流/电压信号的采样序列，通过配置 ADCx 的采样时刻可以得到 ECU 电流/电压信号的采样时刻，通过配置 ADCx 的传输事件可以得到 ECU 的传输事件，诸如此类。

ADCx 模块可以协助 ECU 完成增益校正、偏置校正以及相位校正功能，ADCx 模块中电流/电压信号处理流程如图 14-2。

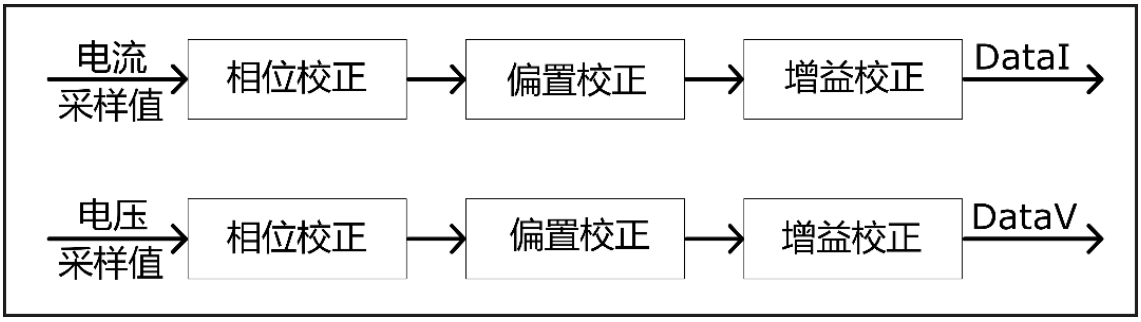


图 14-2 数据前处理框图

14.4.2 有功功率

ECU 模块提供一路有功功率的计算，有功功率的计算框图如图 14-3。

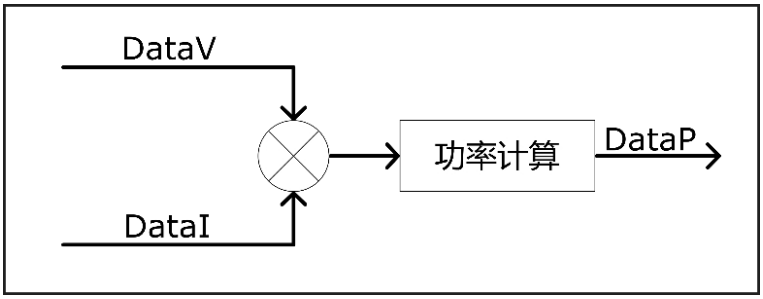


图 14-3 ECU 有功功率计算框图

取平均后的有功功率的计算公式如下：

$$P=\sum_{i=1}^{M}\frac{\sum_{j=1}^{N}V_{ij}*I_{ij}/N}{M}$$

图中 DataI 与 DataV 为 ADCx 模块采样的电流信号与电压信号，相位校正、增益校正、偏置校正部分均在 ADC 模块内完成，经过相位校正、增益校正、偏置校正的电流电压数据为 DataI 与 DataV，ECU 读取 DataI 与 DataV 数据并根据有功功率公式进行计算，得到 DataP。

有功功率 DataP 的计算结果存储在 ECU_P 寄存器中，字长为 32 位，格式为 32 位有符号数。每 M 个半波周期更新一次，M 为 ECU 取平均的次数，通过 ECU_CON 寄存器中的 AVGSEL 位配置。

14.4.3 无功功率

ECU 模块提供一路无功功率的计算，无功功率的计算框图如图 14-4。

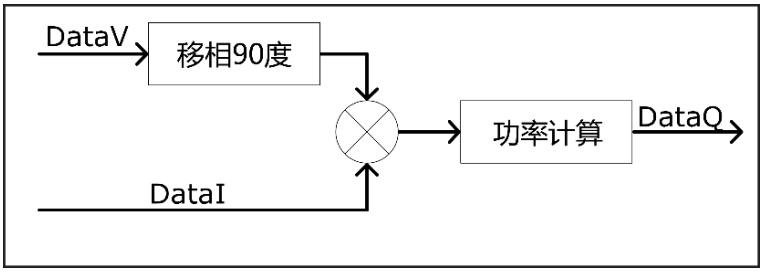


图 14-4 ECU 无功功率计算框图

取平均后的无功功率的计算公式如下：

$$Q=\sum_{i=1}^{M}\frac{\sum_{j=1}^{N}U(t_{ij\text{-}N/2})*I(t_{ij})/N}{M}$$

图中前级部分与有功功率类似，不同的地方在于用于计算的 DataVT 是 DataV 移相 90 度的结果，ECU 读取 DataI 与 DataVT 数据并根据无功功率公式进行计算，得到 DataQ。

无功功率 DataQ 的计算结果存储在 ECU_Q 寄存器中，字长为 32 位，格式为 32 位有符号数。每 M 个半波周期更新一次，M 为 ECU 取平均的次数，通过 ECU_CON 寄存器中的 AVGSEL 位配置。

14.4.4 电流/电压有效值

ECU 提供一路电流有效值和一路电压有效值的计算，电流/电压有效值的计算框图如图 14-5。

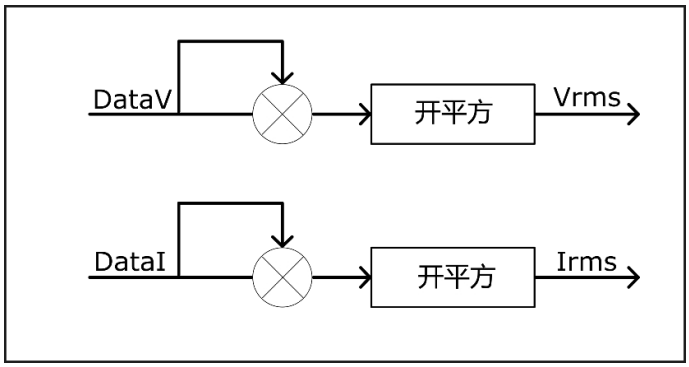


图 14-5 ECU 电流/电压有效值计算框图

取平均后的电流/电压有效值计算公式如下：

$$V_{rms}=\sum_{i=1}^{M}\frac{\sqrt{\sum_{j=1}^{N}V_{ij}^{2}/N}}{M}$$

$$I_{rms}=\sum_{i=1}^{M}\frac{\sqrt{\sum_{j=1}^{N}I_{ij}^{2}/N}}{M}$$

ECU 读取 DataI 与 DataV 数据并根据电流/电压有效值公式进行计算，得到 Irms 与 Urms，注意 ADCx 模块的增益校正/偏置校正/相位校正会影响电流/电压有效值的计算结果。

电流/电压有效值 Irms/Urms 的计算结果存储在 ECU_V/ECU_I 寄存器中，字长为 16 位，格式为 16 位无符号数。每 M 个半波周期更新一次，M 为 ECU 取平均的次数，通过 ECU_CON 寄存器中的 AVGSEL 位配置。

14.4.5 视在功率

ECU 提供一路视在功率的计算，视在功率的计算框图如图 14-6。

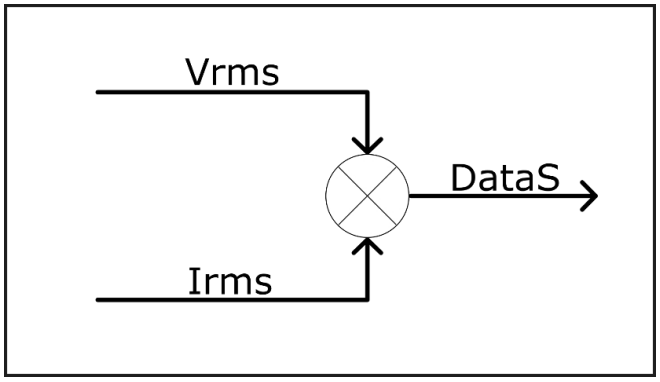


图 14-6 ECU 视在功率计算框图

取平均后的视在功率计算公式如下：

$$S = V_{rms} * I_{rms}$$

ECU 读取 Irms 与 Urms 数据并根据视在功率有效值公式进行计算，得到 DataS，注意由于数据量化误差的存在，可能存在 DataP>DataS 的状况，软件需要对结果进行一定限制。

视在功率 DataS 的计算结果存储在 ECU_S 寄存器中，字长为 32 位，格式为 32 位无符号数。每 M 个半波周期更新一次，M 为 ECU 取平均的次数，通过 ECU_CON 寄存器中的 AVGSEL 位配置。

14.4.6 功率因数

ECU 提供一路功率因数的计算，功率因数的计算框图如图 14-7。

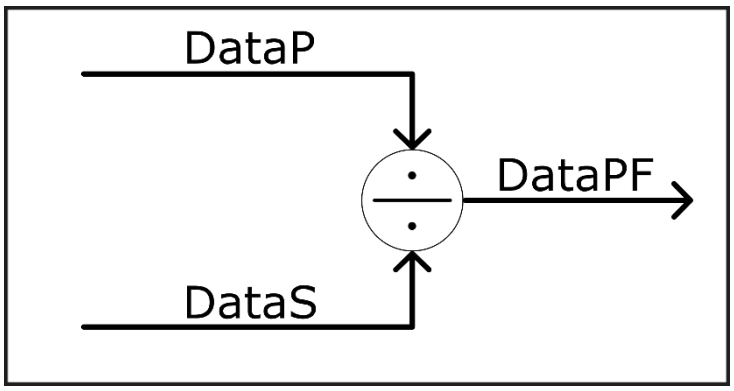


图 14-7 ECU 功率因数计算框图

取平均后的功率因数的计算公式如下：

$$PF = \cos\theta = \frac{P}{S}$$

ECU 读取 DataP 与 DataS 数据并根据功率因数公式进行计算，得到 DataPF，注意由于数据量化误差的存在，可能存在 DataP>DataS 的状况，软件需要对结果进行一定限制。

由于除法运算的特殊性，硬件在进行功率因数计算时，需要对各个数据进行定标，ECU 支持对 DataP 左移 M 位、对 DataS 右移 N 位，则 DataPF 定标为 32.(M+N)定点数。如果希望 DataPF 定标为 32.16 定点数，则需要满足(M+N)=16。

功率因数 DataPF 的计算结果存储在 ECU_PF 寄存器中，字长为 32 位，格式为 32 位有符号数。每 M 个半波周期更新一次，M 为 ECU 取平均的次数，通过 ECU_CON 寄存器中的 AVGSEL 位配置。

14.4.7 频率测量

ECU 提供一路基波频率的计算，取平均后的基波频率的计算公式如下：

$$\begin{cases} \overline{N} = \sum_{i=1}^{M} \frac{N_i}{M} \\ F = \frac{f_s}{2\overline{N}} \end{cases}$$

ECU 根据 ADCx 模块提供的过零事件，读取 ADC 半周期点数，利用 ADC 半周期点数与 ADC 采样频率计算基波频率 DataF。

基波频率 DataF 的计算结果存储在 ECU_F 寄存器中，字长为 16 位，格式为 16 位无符号数，单位为 ADC 采样率。每 M 个半波周期更新一次，M 为 ECU 取平均的次数，通过 ECU_CON 寄存器中的 AVGSEL 位配置。

14.4.8 事件输入&过零检测

ECU 提供一路过零检测功能，通过 ADCx 模块事件输入实现，事件输入框图如图 14-8。

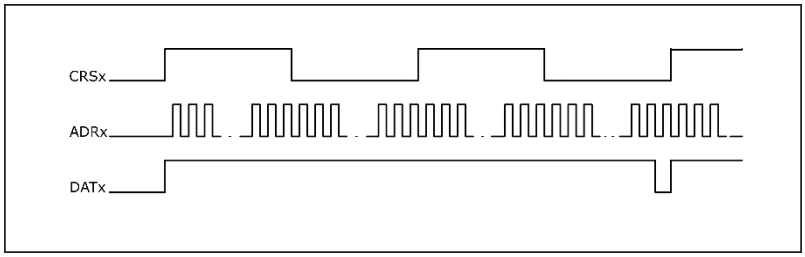


图 14-8 ECU 事件输入框图

ECU 从 ADCx 模块获取过零检测信号，过零检测信号同时会通过 debug 信号送到片外，可以作为过零中断输出到片外。

ECU 的所有外部触发事件如表 14-1 至表 14-3 所示，包括过零触发事件/地址触发事件和数据触发事件。

表 14-1 ECU 过零触发事件

Name	Source	Type	CRSSEL[3:0]
ecu_crs_evt0	ADC0_PPFLAG0	来自片上 ADC 的内部信号	0000
ecu_crs_evt1	ADC0_PPFLAG1	来自片上 ADC 的内部信号	0001
ecu_crs_evt2	ADC0_PPFLAG2	来自片上 ADC 的内部信号	0010
ecu_crs_evt3	ADC0_PPFLAG3	来自片上 ADC 的内部信号	0011
ecu_crs_evt4	ADC1_PPFLAG0	来自片上 ADC 的内部信号	0100
ecu_crs_evt5	ADC1_PPFLAG1	来自片上 ADC 的内部信号	0101
ecu_crs_evt6	ADC1_PPFLAG2	来自片上 ADC 的内部信号	0110
ecu_crs_evt7	ADC1_PPFLAG3	来自片上 ADC 的内部信号	0111
ecu_crs_evt8	TMR0_CC	来自片上定时器的内部信号	1000
ecu_crs_evt9	TMR1_CC	来自片上定时器的内部信号	1001

ecu_crs_evt10	TMR2_CC	来自片上定时器的内部信号	1010
ecu_crs_evt11	TMR3_CC	来自片上定时器的内部信号	1011
ecu_crs_evt12	TMR4_CC	来自片上定时器的内部信号	1100
ecu_crs_evt13	TMR5_CC	来自片上定时器的内部信号	1101
ecu_crs_evt14	TMR6_CC	来自片上定时器的内部信号	1110
ecu_crs_evt15	TMR7_CC	来自片上定时器的内部信号	1111

表 14-2 ECU 地址触发事件

Name	Source	Type	ADRSEL[3:0]
ecu_adr_evt0	ADC0_ADFLAG0	来自片上 ADC 的内部信号	0000
ecu_adr_evt1	ADC0_ADFLAG1	来自片上 ADC 的内部信号	0001
ecu_adr_evt2	ADC0_ADFLAG2	来自片上 ADC 的内部信号	0010
ecu_adr_evt3	ADC0_ADFLAG3	来自片上 ADC 的内部信号	0011
ecu_adr_evt4	ADC1_ADFLAG0	来自片上 ADC 的内部信号	0100
ecu_adr_evt5	ADC1_ADFLAG1	来自片上 ADC 的内部信号	0101
ecu_adr_evt6	ADC1_ADFLAG2	来自片上 ADC 的内部信号	0110
ecu_adr_evt7	ADC1_ADFLAG3	来自片上 ADC 的内部信号	0111
ecu_adr_evt8	TMR0_CC	来自片上定时器的内部信号	1000
ecu_adr_evt9	TMR1_CC	来自片上定时器的内部信号	1001
ecu_adr_evt10	TMR2_CC	来自片上定时器的内部信号	1010
ecu_adr_evt11	TMR3_CC	来自片上定时器的内部信号	1011
ecu_adr_evt12	TMR4_CC	来自片上定时器的内部信号	1100
ecu_adr_evt13	TMR5_CC	来自片上定时器的内部信号	1101
ecu_adr_evt14	TMR6_CC	来自片上定时器的内部信号	1110
ecu_adr_evt15	TMR7_CC	来自片上定时器的内部信号	1111

表 14-3 ECU 数据触发事件

Name	Source	Type	DATSEL[3:0]
ecu_dat_evt0	ADC0_DTFLAG0	来自片上 ADC 的内部信号	0000
ecu_dat_evt1	ADC0_DTFLAG1	来自片上 ADC 的内部信号	0001
ecu_dat_evt2	ADC0_DTFLAG2	来自片上 ADC 的内部信号	0010
ecu_dat_evt3	ADC0_DTFLAG3	来自片上 ADC 的内部信号	0011
ecu_dat_evt4	ADC1_DTFLAG0	来自片上 ADC 的内部信号	0100
ecu_dat_evt5	ADC1_DTFLAG1	来自片上 ADC 的内部信号	0101
ecu_dat_evt6	ADC1_DTFLAG2	来自片上 ADC 的内部信号	0110
ecu_dat_evt7	ADC1_DTFLAG3	来自片上 ADC 的内部信号	0111
ecu_dat_evt8	TMR0_CC	来自片上定时器的内部信号	1000
ecu_dat_evt9	TMR1_CC	来自片上定时器的内部信号	1001
ecu_dat_evt10	TMR2_CC	来自片上定时器的内部信号	1010
ecu_dat_evt11	TMR3_CC	来自片上定时器的内部信号	1011
ecu_dat_evt12	TMR4_CC	来自片上定时器的内部信号	1100
ecu_dat_evt13	TMR5_CC	来自片上定时器的内部信号	1101
ecu_dat_evt14	TMR6_CC	来自片上定时器的内部信号	1110
ecu_dat_evt15	TMR7_CC	来自片上定时器的内部信号	1111

14.4.9 中断

ECU 提供一路计算完成中断，包括一个中断使能位 IE 与一个中断状态位 IS，中断状态位 IS 置起代表 ECU 单次计算完成，IS 写 1 清 0。

ECU 计算完成中断每 M 个半波周期置起一次，M 为 ECU 取平均的次数。

14.4.10 数据开方

ECU 提供一路数据开方功能，包括计算使能位 SQRT、输入数据 INDATA 与输出数据 OUTDATA。软件首先写入待开方数据 INDATA，向使能位 SQRT 写 1 后等待 SQRT 清 0，SQRT 清 0 代表数据开方完成，可以从 OUTDATA 读出平方根数据。

ECU 输入数据格式为 32 位无符号数，输出数据格式为 16 位无符号数。

14.4.11 校表方法

ECU 支持功率校表法，经过校正的仪表，有功和无功精度可以达到指定要求。

ECU 的校正手段包括：

- ADC 采样通道增益校正
- ADC 采样通道偏置校正
- ADC 采样通道相位校正

功率校表法

和传统的脉冲校表法相比，功率校表法具有校表简单、快捷的优点，其校表系统如图 14-9 所示，只需要一台高精度电流电压源，该源精度等级应高于被测表所要求的等级。

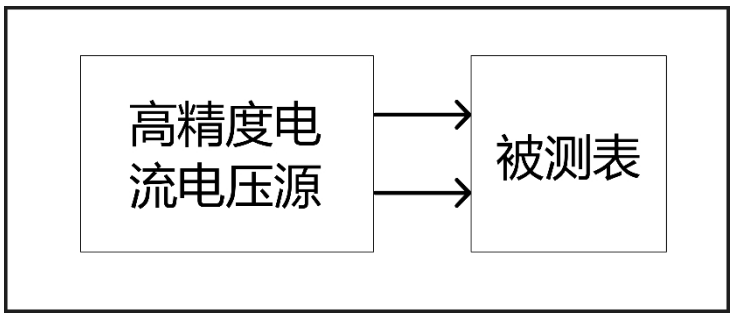


图 14-9 ECU 功率校表框图

14.4.12 功率校表法

功率法校表流程如图 14-10 所示。

1. 计算额定输入时标准 U、I 寄存器值，计算 PF=1.0 和 PF=0.5L 时，标准有功功率值。
2. 搭好校表环境并进行参数配置，如配置好 ECU 事件/地址、ADC 事件/地址、ADC 过零阈值等。
3. 标准源电流/电压额定输入，存储两个基波周期数据，再根据数据最大值最小值计算相应的偏置填入 ADC 偏置寄存器，完成偏置校正。
4. 标准源额定输出，读出实际电压、电流有效值，并计算与理论值的误差，再根据此误差计算出 ADC 电压、电流通道增益寄存器的值，填入并比对校正结果，完成 U、I 通道增益校正。
5. 标准源 PF=0.5L，额定输出，读取有功功率并计算与理论值的误差，再根据此误差计算出通道相位寄存器或功率相位寄存器的值，填入并比对校正结果，完成通道相位校正。

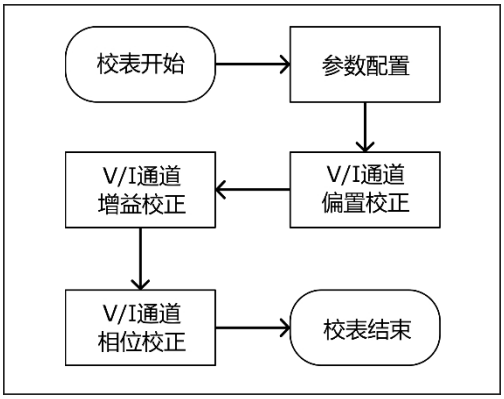


图 14-10 ECU 功率校表流程

14.4.13 参数配置

参数配置流程如图 14-11 所示。

1. ECU 事件/地址选择，填入 ECU 事件/地址寄存器。
2. ADC 事件/地址选择，填入 ADC 事件/地址寄存器。
3. 根据额定电压电流有效值计算过零阈值，并填入相应的阈值寄存器。

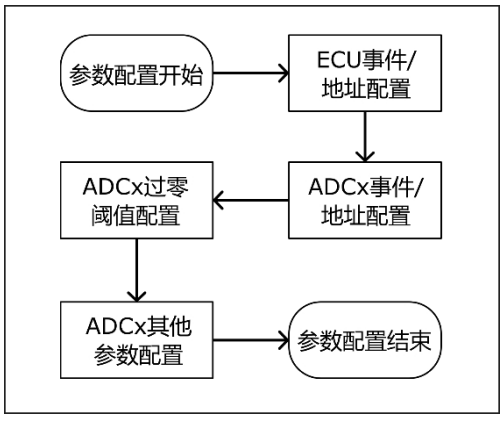


图 14-11 ECU 参数配置流程

14.4.14 偏置校正

电流偏置校正说明：

1. 配置标准源，使得 U、I 都为额定输入。
2. 配置 ADC 采样时序，存储 U、I 通道各自两个全波周期的采样数据。
3. 读取 U、I 通道两个全波周期采样数据的最大/最小值，对最大/最小值求平均值。
4. 将 U、I 通道最大/最小值的平均值写入 UOffset 与 IOffset 寄存器。
5. 电流/电压偏置校正结束。

14.4.15 增益校正

通道增益校正说明：

标准源额定输出，假设标准电压有效值为 U，标准电流有效值为 I，读出实际电压有效值寄存器为 U’，实际电流有效值寄存器为 I’，则：

电压有效值误差 ErrU =（U’-U）/ U

电流有效值误差 ErrI =（I’-I）/ I

U 通道增益校正可通过配置 UCoeff 寄存器实现，UCoeff 的计算方法如下：

$$UGain = \frac{-ErrU}{1+ErrU}$$

如果 UGain>=0，则 UCoeff=INT[UGain*2^12]；

否则 UGain<0，则 UCoeff=INT[2^13+UGain*2^12]。

I 通道功率增益校正可通过配置 ICoeff 寄存器实现，方法同 UCoeff。

通过该方法额定输出的电压电流有效值校正误差可以控制在 0.02%~0.03%，A 相电压有效值及电流有效值校正完成后，有功功率/无功功率/视在功率等增益校正同时完成。

14.4.16 相位校正

电流相位校正说明：

标准源配置改变为 PF=0.5L 即功率因数角为 60°，额定输出，假设理想有功功率为 PA0.5L，读出的实际有功功率为 PA0.5L’，则由相位误差带来的有功功率误差为：

ErrPA=（PA0.5L’-PA0.5L）/ PA0.5L

该误差可以通过通道相位校正方法，即配置 U 通道与 I 通道的采样时间点偏差δ实现校正。

相位补偿公式：若 A 相 U、I 通道间的角差为θ，则

- 若θ>0，表示 UA 超前 IA；
- 若θ<0，表示 UA 滞后 IA。

对 50HZ，相位偏差与采样时间点偏差δ有 0.00675°/ LSB 的关系，即调整 1 个 LSB，功率因数

角为 60°时，相位角度变化 0.00675°。

14.5 寄存器描述

14.5.1 寄存器列表

Name	Offset	Width	Description
ECU_CON	0x00	32bits	ECU 配置寄存器
ECU_PRC	0x04	32bits	ECU 事件处理寄存器
ECU_SQRT_IN	0x08	32bits	ECU Sqrt 输入数据寄存器
ECU_SQRT_OUT	0x0C	32bits	ECU Sqrt 输出数据寄存器
ECU_V_ADDR1	0x10	32bits	ECU 电压其实地址寄存器
ECU_V_ADDR2	0x14	32bits	ECU 电压偏移地址寄存器
ECU_I_ADDR1	0x18	32bits	ECU 电流起始地址寄存器
ECU_I_ADDR2	0x1C	32bits	ECU 电流偏移地址寄存器
ECU_V	0x20	32bits	ECU 电压有效值寄存器
ECU_I	0x24	32bits	ECU 电流有效值寄存器
ECU_P	0x28	32bits	ECU 有功功率寄存器
ECU_Q	0x2C	32bits	ECU 无功功率寄存器
ECU_S	0x30	32bits	ECU 视在功率寄存器
ECU_PF	0x34	32bits	ECU 功率因数寄存器
ECU_F	0x38	32bits	ECU 基波频率寄存器

14.5.2 寄存器详细描述

14.5.2.1 ECU 配置寄存器 (ECU_CON)

- **Name**：ECU Config Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15]	INT	R/W1C	0x0	ECU 结束中断标志位，写 1 清零。 0：ECU 未结束 1：ECU 已经结束
[14]	SQRT	R/WAC	0x0	ECU 开方计算使能位，计算结束后自动清零。 0：不使能 1：使能 ECU 具有独立的开方功能，数据由 SQRT_IN 写入，由 SQRT_OUT 读出
[13-9]	ACSFT	R/W	0x8	ECU 有功功率左移位数（有功功率原始数据 32 位，移位后取高 48 位计算功率因数） 00000：不移位 00001：左移 1 位 …… 01111：左移 15 位 10000：左移 16 位 10001-11111：Reserved
[8-4]	APSFT	R/W	0x8	ECU 视在功率右移位数（视在功率原始数据 32 位，移位后取低 16 位计算功率因数） 00000：不移位 00001：右移 1 位 …… 01111：右移 15 位 10000：右移 16 位 10001-11111：Reserved
[3-2]	AVGSEL	R/W	0x0	对基波周期数据求平均的选择位。 00：不发生求平均 01：对 2 个基波周期求平均 10：对 4 个基波周期求平均 11：对 8 个基波周期求平均
[1]	INTEN	R/W	0x0	ECU 结束中断使能位 0：不使能 1：使能

				ECU 模块使能位
[0]	ENABLE	R/W	0x0	0：不使能 1：使能

14.5.2.2 ECU 事件处理寄存器 (ECU_PRC)

- **Name**：ECU Event Process Register
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11-8]	CRSSEL	R/W	0x0	ECU 过零事件选择位 0000：过零事件 0 0001：过零事件 1 0010：过零事件 2 …… 1111：过零事件 15
[7-4]	ADRSEL	R/W	0x0	ECU 地址事件选择位 0000：地址事件 0 0001：地址事件 1 0010：地址事件 2 …… 1111：地址事件 15
[3-0]	DATSEL	R/W	0x0	ECU 数据事件选择位 0000：数据事件 0 0001：数据事件 1 0010：数据事件 2 …… 1111：数据事件 15

14.5.2.3 ECU Sqrt 输入数据寄存器 (ECU_SQRT_IN)

- **Name**：ECU Sqrt Input Data Register
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	INDATA	R/W	0x0	ECU 被开方数寄存器（32 位）

14.5.2.4 ECU Sqrt 输出数据寄存器 (ECU_SQRT_OUT)

- **Name**：ECU Sqrt Output Data Register
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	OUTDATA	R/W	0x0	ECU 平方根寄存器（32 位）

14.5.2.5 ECU 电压起始地址寄存器 (ECU_V_ADDR1)

- **Name**：ECU Voltage Start Addr Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	STADDR	R/W	0x0	ECU 电压（乒乓）数据存放的起始地址

14.5.2.6 ECU 电压偏移地址寄存器 (ECU_V_ADDR2)

- **Name**：ECU Voltage Offset Addr Register
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFADDR	R/W	0x0	ECU 电压（乒乓）数据存放的偏移地址

14.5.2.7 ECU 电流起始地址寄存器 (ECU_I_ADDR1)

- **Name**：ECU Current Start Addr Register
- **Size**：32bits
- **Offset**：0x18
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ATADDR	R/W	0x0	ECU 电流（乒乓）数据存放的起始地址

14.5.2.8 ECU 电流偏移地址寄存器 (ECU_I_ADDR2)

- **Name**：ECU Current Offset Addr Register
- **Size**：32bits
- **Offset**：0x1C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFADDR	R/W	0x0	ECU 电流（乒乓）数据存放的偏移地址

14.5.2.9 ECU 电压有效值寄存器 (ECU_V)

- **Name**：ECU Voltage RMS Register
- **Size**：32bits
- **Offset**：0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	VRMS	R	0x0	电压平均有效值（Averaged Voltage RMS），位宽 16 位

14.5.2.10 ECU 电流有效值寄存器 (ECU_I)

- **Name**：ECU Current RMS Register
- **Size**：32bits
- **Offset**：0x24
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	IRMS	R	0x0	电流平均有效值（Averaged Current RMS），位宽 16 位

14.5.2.11 ECU 有功功率寄存器 (ECU_P)

- **Name**：ECU Active Power Register
- **Size**：32bits
- **Offset**：0x28
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	PAVG	R	0x0	平均有功功率（Averaged Active Power），位宽 32 位

14.5.2.12 ECU 无功功率寄存器 (ECU_Q)

- **Name**：ECU Reactive Power Register
- **Size**：32bits
- **Offset**：0x2C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	QAVG	R	0x0	平均无功功率（Averaged Reactive Power），位宽 32 位

14.5.2.13 ECU 视在功率寄存器 (ECU_S)

- **Name**：FLASH Timing Register2
- **Size**：32bits
- **Offset**：0x30
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	SCAL	R	0x0	视在功率（Apparent Power），位宽 32 位

14.5.2.14 ECU 功率因数寄存器 (ECU_PF)

- **Name**：ECU Power Factor Register
- **Size**：32bits
- **Offset**：0x34
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	PFCAL	R	0x0	功率因数（Power Factor），位宽 32 位，定标位数由 APSFT 决定

14.5.2.15 ECU 基波频率寄存器 (ECU_F)

- **Name：** ECU Fundamental Frequency Register
- **Size：** 32bits
- **Offset：** 0x38
- **Default：** 0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	FCNT	R	0x0	基波频率（Fundamental Frequency），位宽 16 位

15 模数转换器（ADC）

15.1 简介

芯片支持 2 路 ADC 的实现，每路 ADC 由一个 13 位循环模数转换器组成， ADC0 与 ADC1 具有各自的控制电路，可以在同步模式下运行、也可以在非同步模式下运行。

每路 ADC 具有多达 13 个复用通道，包括 12 个外部通道，还有 1 个内部通道，内部通道与 12 个外部通道共享。各种不同通道的 A/D 转换可在单次、连续、扫描或不连续采样模式下进行。ADC 的结果存储在一个或多个 16 位数据寄存器中。

每路 ADC 具有模拟看门狗功能，允许应用检测输入电压是否超过了用户定义的阈值上限或下限。每路 ADC 内置硬件过采样器，可以提高模拟性能，同时还能减轻 CPU 进行相关计算的负担。

15.2 结构框图

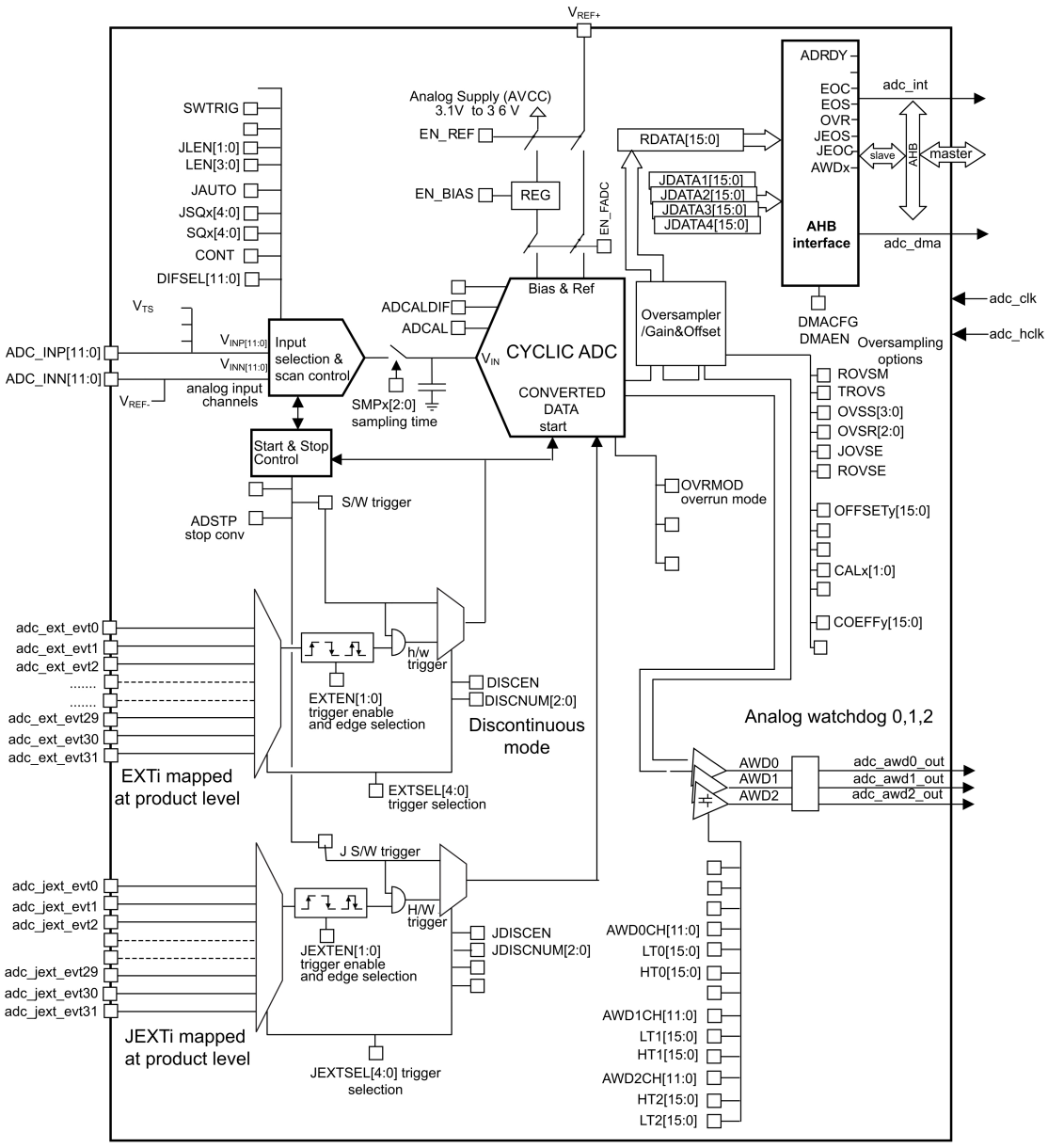


图 15-1 ADC 结构框图

15.3 主要特性

- 高性能特性
  - 支持 2 路 ADC，可以在同步模式下运行
  - 每路 ADC 连接到 12 个外部通道+ 1 个内部通道
  - 每路 ADC 支持独立设置各通道的单端/差分输入
  - 每路 ADC 支持独立设置各通道的采样时间
  - 每路 ADC 支持多达 16 个常规通道
  - 每路 ADC 支持多达 4 个注入通道（模拟输入分配为常规通道或注入通道完全可配置）
  - 每路 ADC 支持 12 路 DMA 搬运数据，供常规通道使用
  - 每路 ADC 支持 4 个专用数据寄存器，供注入通道使用
- 过采样器
  - 每路 ADC 支持 16 位数据寄存器，数据位数最多为 16 位
  - 每路 ADC 支持 2~256 倍过采样
  - 每路 ADC 数据右移可编程
- 数据预处理
  - 每路 ADC 支持增益补偿，最多支持 4 组补偿系数
  - 每路 ADC 支持偏置补偿，最多支持 4 组补偿系数
- 转换启动
  - 每路 ADC 支持通过软件启动常规转换和注入转换
  - 每路 ADC 支持通过具有可配置极性的硬件触发事件（内部定时器事件或 GPIO 输入事件）启动常规转换和注入转换
- 转换模式
  - 每路 ADC 均可转换单个通道，也可扫描一个通道序列
  - 每路 ADC 在单次模式下在每次触发时单次地转换选定的输入
  - 每路 ADC 在连续模式下在每次触发时连续地转换选定的输入
  - 每路 ADC 在不连续模式在每次触发时分组地转换选定的输入
  - 每路 ADC 支持在准备就绪、常规/注入转换结束、常规/注入序列转换结束，模拟看门狗 0/1/2 或数据溢出事件时生成中断
- 每个 ADC 支持 3 个模拟看门狗，模拟看门狗可以过滤并忽略超出范围的数据
- ADC 输入范围：$GND\_ \leq V_{IN} \leq 3V$

15.4 功能描述

15.4.1 ADC 引脚和内部信号

表 15-1 ADC 内部输入/输出信号

内部信号名称	信号类型	说明
adcx_ext_evt[31:0]	输入	共有多达 32 个外部触发输入用于常规转换（可连接至片上定时器），这些输入对 ADC0 和 ADC1 一致。
adcx_jext_evt[31:0]	输入	共有多达 32 个外部触发输入用于注入转换（可连接至片上定时器），这些输入对 ADC0 和 ADC1 一致。
adcx_awd0_out adcx_awd1_out adcx_awd2_out	输出	ADCx 内部模拟看门狗输出信号，连接到片上定时器
adcx_norm_int	输出	ADCx 常规中断
adcx_samp_int	输出	ADCx 采样中断
adcx_half_int	输出	ADCx DMA 半完成中断
adcx_full_int	输出	ADC xDMA 完成中断
adc_hclk	输入	ADCx 总线时钟
adc_clk	输入	ADCx 功能时钟

表 15-2 ADC 输入/输出引脚

引脚名称	信号类型	说明
V_{REF}	输入，参考基准	ADC 的基准电压，典型值为 1.5 V
AVCC	输入，模拟电源	模拟电源，典型值为 3.3 V
AVSS	输入，模拟电源地	模拟电源的接地，典型值为 0 V
ADCx_IN[11:0]	外部模拟输入信号	每路 ADC 多达 12 个模拟输入通道（x＝ADC 编号= 0、1）： ➢ 11 个外部通道（ADCx_INP0 至 INP10） ➢ 1 个内部通道（ADCx_INP11） 输入信号范围为（0,3）V

15.4.2 ADC0/1 连接关系

ADC0 和 ADC1 紧密耦合并共用一些外部通道，如图 15-2 所示。

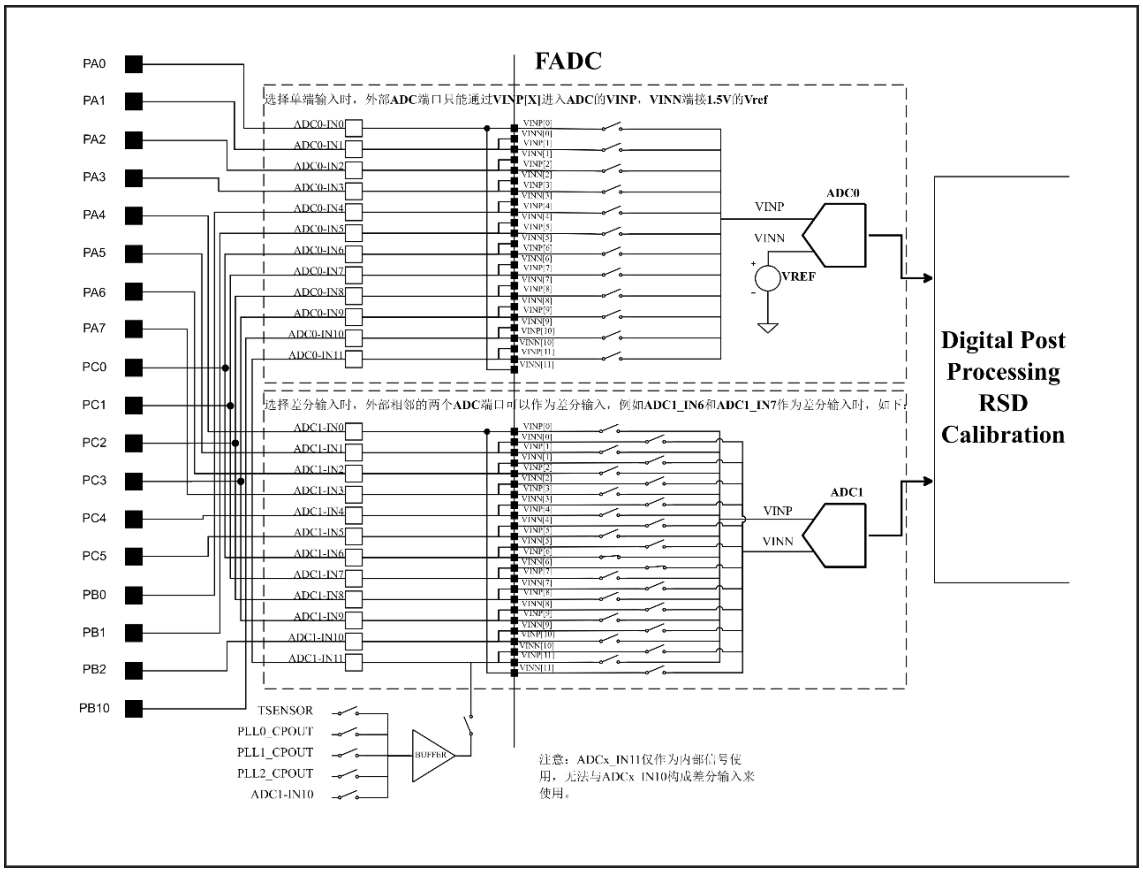


图 15-2 ADC0 / ADC1 连接关系图

15.4.3 单端和差分输入通道

可通过写入 ADC_DIFSEL 寄存器中的 DIFSEL [11:0]位将通道配置为单端输入或差分输入。必须在禁用 ADC（EN_FADC = 0）时写入此配置。

在单端输入模式下，要为通道“i”转换的模拟电压是外部电压 $V_{INP[i]}$（正输入）和 V_{REF-}（负输入）之差。

在差分输入模式下，要为通道“i”转换的模拟电压是外部电压 $V_{INP[i]}$（正输入）和 $V_{INN[i]}$（负输入）之差。

差分模式的输出数据是无符号数据，当 $V_{INP[i]}$为 V_{REF-}、当 $V_{INN[i]}$为 V_{REF+}时，输出数据为 0x0000（16 位分辨率模式）；当 $V_{INP[i]}$为 V_{REF+}、当 $V_{INN[i]}$为 V_{REF-}时，输出数据为 0xFFFF。其中，V_{REF-}为 0V，V_{REF+}为 3V。

$$转换后的值 = \frac{ADC\_Full\_Scale}{2} \times \left[1 + \frac{V_{INP} - V_{INN}}{V_{REF+}}\right]$$

当 ADC 配置为差分模式时，两路输入的偏置电压均应为 $V_{REF+}/2$，输入信号应为差分信号（共

模电压应固定）。

内部通道（例如 V_{TS} 和 V_{REFINT}）仅在单端模式下使用。

注：在差分输入模式下配置通道“i”时，其负输入电压 $V_{INN\,[i]}$ 连接到 $V_{INP\,[i+1]}$，因此，通道“i+1”在单端模式或差分模式下不再可用，不应再次进行转换。

15.4.4 ADC 开关控制

可以使用 EN_FADC 控制位使能或禁止 ADC：

- EN_FADC = 1 使能 ADC。 ADC 准备就绪后，将置位标志 ADRDY。
- EN_FADC = 0 禁用 ADC。

随后可通过将 ADSTART 置 1 开始进行常规转换，如果常规触发事件已使能，也可在外部常规触发事件时开始进行常规转换。

可通过将 JADSTART 置 1 开始进行注入转换，如果注入触发事件已使能，也可在外部注入触发事件时开始进行注入转换。

通过软件使能 ADC 的流程

1. 向 ADC_ISR 寄存器中的 ADRDY 位写入“1”，将其清零；
2. 将 EN_FADC 置 1；
3. 等待直至 ADRDY = 1（ADRDY 会在 ADC 启动时间后置 1）。这可以使用相关的中断来实现（将 ADRDYIE 置 1）；
4. 向 ADC_ISR 寄存器中的 ADRDY 位写入“1”，将其清除（可选）。

通过软件禁止 ADC 的流程

1. 检查 ADSTART 和 JADSTAR 是否均为 0，以确保当前未执行任何转换。如果需要，可将 ADSTP 置 1，并将 JADSTP 置 1，随后等待至 ADSTP = 0 且 JADSTP = 0，以停止任何正在进行的常规转换和注入转换；
2. 将 EN_FADC 置 0。

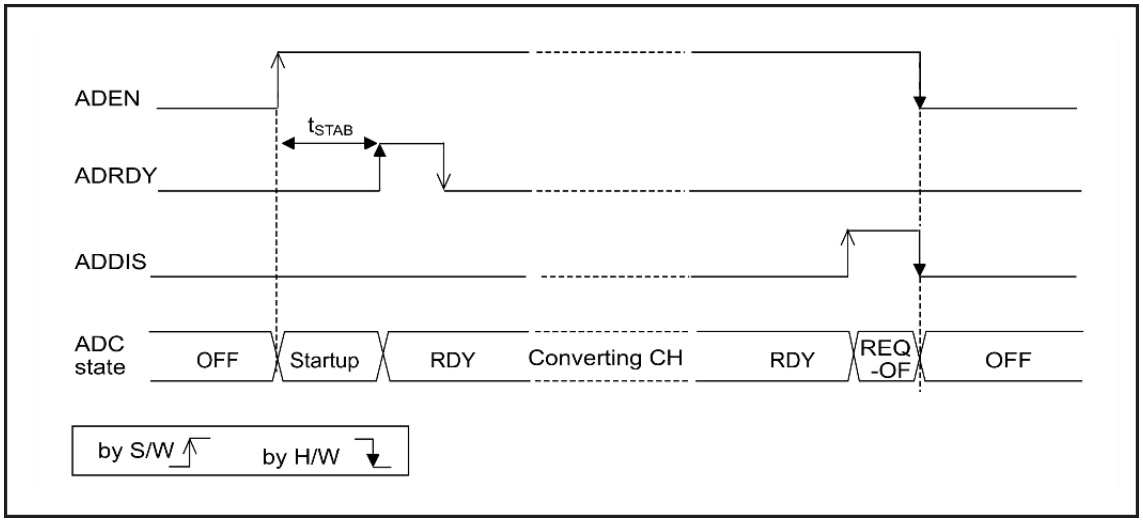


图 15-3 使能/禁止 ADC

15.4.5 写入 ADC 控制位时的限制

仅当 ADC 已禁止时（EN_FADC 必须等于 0），软件才可通过写入 CMU 控制位以配置和启用 ADC 时钟（请参见 CMU 部分），ADC_DIFSEL 寄存器中的控制位 DIFSEL 和 ADC_CR2 寄存器中的控制位 EN_FADC。

只有在 ADC 已使能、并且没有待处理的禁止 ADC 的请求（EN_FADC 必须等于 1）时，才允许软件向 ADC_CR0 寄存器的控制位 ADSTART、JADSTART 执行写操作。

对于 ADC_CR1，ADC_SMPRx，ADC_TRy，ADC_SQRy，ADC_JDRy，ADC_OFRy，ADC_GCRy 和 ADC_IER 寄存器的所有其他控制位：

- 对于与常规转换配置相关的控制位，仅当 ADC 已使能（EN_FADC = 1）且未进行常规转换（ADSTART 必须等于 0）时，才允许软件对这些位执行写操作。
- 对于与注入转换配置相关的控制位，仅当 ADC 已使能（EN_FADC = 1）且未进行注入转换（JADSTART 必须等于 0）时，才允许软件对这些位执行写操作。

仅当 ADC 已使能且没有待处理的禁止 ADC 的请求（ADSTART 或 JADSTART 必须等于 1）时，软件才可以对 ADC_CR0 寄存器中的 ADSTP 或 JADSTP 控制位执行写操作。

如果 ADC 已使能（EN_FADC = 1），软件可随时对 ADC_JSQR 寄存器执行写操作。

注：没有硬件保护可以防止执行此类禁止的写操作，ADC 行为可能进入未知状态。要从此状态恢复，必须禁止 ADC（将 EN_FADC 清 0，并且将 ADC_CR1 寄存器的所有位都清零）。

15.4.6 通道选择

每个 ADC 最多有 13 个多路复用通道：

- 来自 GPIO PAD 的 12 个模拟输入（ADCx_INP / INN [0..11]）
- ADC 连接到以下内部模拟输入
  - 内部基准电压（VREFINT）连接到 ADC0_INP11，ADC1_INP11
  - 内部温度传感器（VTS）连接到 ADC0_INP11 和 ADC1_INP11

可以将转换可以分为两组：常规转换和注入转换。每个组包含一个转换序列，该序列可按任何顺序在任何通道上完成。例如，可以按以下顺序实现转换序列：ADCx_INP / INN3，ADCx_INP / INN8，ADCx_INP / INN2，ADCx / INN2，ADCx_INP / INN0，ADCx_INP / INN2，ADCx_INP / INN2，ADCx_INP / INN11。

- 一个常规转换组最多由 16 个转换构成。必须在 ADC_SQRy 寄存器中选择转换序列的常规通道及其顺序。常规转换组中的转换总数必须写入 ADC_LR 寄存器中的 LEN [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_LR 寄存器的 JLEN [1:0]位。

不可以在常规转换进行中对 ADC_SQRy 寄存器进行修改。如果要修改 ADC_SQRy 寄存器，必须先写入 ADSTP = 1 停止 ADC 常规转换。

进行注入转换时，可实时修改 ADC_JSQR 寄存器。

温度传感器 V_{sense} 和内部参考电压 V_{REFINT} 连接到通道 ADC0 V_{INP}[11]和 ADC1 V_{INP}[11]。

要对其中一条内部模拟通道进行转换，必须先对 SYSCTRL 中的寄存器进行配置，以使能相应的模拟源。

15.4.7 可独立设置各通道采样时间

开始转换之前，ADC 必须在待测电压源与 ADC 内置采样电容之间建立直接连接。该采样时间必须足以使输入电压源为嵌入式电容充电至输入电压水平。

对各通道进行采样时可以使用不同的采样时间，采样时间可通过 ADC_SMPR0 和 ADC_SMPR1 寄存器中的 SMP [2:0]位进行编程。可选采样时间值如下：

- SMP = 000：6 个 ADC 时钟周期
- SMP = 001：18 个 ADC 时钟周期
- SMP = 010：42 个 ADC 时钟周期
- SMP = 011：90 个 ADC 时钟周期
- SMP = 100：186 个 ADC 时钟周期
- SMP = 101：378 个 ADC 时钟周期
- SMP = 110：762 个 ADC 时钟周期
- SMP = 111：1530 个 ADC 时钟周期

总转换时间的计算如下：
TCONV =采样时间+ 30 个 ADC 时钟周期

例：adc_clk = 160 MHz，采样时间为 6 个 ADC 时钟周期：
TCONV =（6 + 30）ADC 时钟周期= 36 个 ADC 时钟周期= 225 ns

15.4.8 单次转换模式

单次转换模式下，ADC 会将通道的所有转换执行一次。CONT 位为 0 时，可通过以下方式启动此模式：

- 将 ADC_CR0 寄存器中的 ADSTART 位置 1（用于常规通道，需要选择软件触发）
- 将 ADC_CR0 寄存器中的 JADSTART 位置 1（用于注入通道，需要选择软件触发）
- 外部硬件触发事件（用于常规通道或注入通道）

触发外部事件之前，ADSTART 位或 JADSTART 位必须置 1。

在常规序列中，每次转换完成后：

- 转换数据存储在 16 位 ADC_DR 寄存器中
- EOC（常规转换结束）标志位置 1
- EOCIE 使能位置 1 时将产生中断

在注入序列中，每次转换完成后：

- 转换数据存储在四个 16 位 ADC_JDRy 寄存器的其中之一中
- JEOC（注入转换结束）标志位置 1
- JEOCIE 使能位置 1 时将产生中断

常规序列完成后：

- EOS（常规序列结束）标志位置 1

- EOSIE 使能位置 1 时将产生中断

注入序列完成后：

- JEOS（注入序列结束）标志位置 1
- JEOSIE 使能位置 1 时将产生中断

随后，ADC 会停止工作，直到发生新的外部常规或注入触发事件，或者 ADSTART 或 JADSTART 位再次置 1，ADC 才会再次工作。

注：如果要转换单个通道，可将序列长度编程为 1。

15.4.9 连续转换模式

连续转换模式仅适用于常规通道。

在连续转换模式下，如果发生软件或硬件的常规触发事件，ADC 将对通道的所有常规转换执行一次，随后会自动重启并持续执行序列的每个转换。CONT 位为 1 时，可通过外部触发事件或将 ADC_CR0 寄存器中的 ADSTART 位置 1 来启动此模式。

在常规转换序列中，每次转换完成后：

- 转换数据存储在 16 位 ADC_DR 寄存器中
- EOC（转换结束）标志位置 1
- EOCIE 使能位置 1 时将产生中断

转换序列完成后：

- EOS（序列结束）标志位置 1
- EOSIE 使能位置 1 时将产生中断

随后，会立即开启新转换序列，ADC 会继续重复执行转换序列。

注：如果要转换单个通道，可将序列长度编程为 1。

不可以同时使能不连续模式和连续模式：禁止同时将 DISCEN 和 CONT 位置 1。

注入通道不能连续转换。唯一例外的是，在连续转换模式下（使用 JAUTO 位）注入通道配置为在常规通道后的自动转换。

15.4.10 开始转换

软件通过将 ADSTART 置 1 的方式开始进行 ADC 常规转换。

ADSTART 置 1 后，会开始进行常规转换：

- EXTEN = 0x0（软件触发事件），常规转换立即开始
- EXTEN != 0x0（硬件触发事件），常规转换在所选常规触发事件的下一个有效沿后开始

软件通过将 JADSTART 置 1 的方式开始进行 ADC 注入转换。

JADSTART 置 1 后，会开始进行注入转换：

- JEXTEN = 0x0（软件触发事件），注入转换立即开始

- JEXTEN != 0x0（硬件触发事件），注入转换在所选注入触发事件的下一个有效沿后开始

注：在自动注入模式（JAUTO = 1）下，使用 ADSTART 位开始常规转换，然后再进行自动注入转换（JADSTART 必须保持为 0）。

ADSTART 和 JADSTART 指示 ADC 当前是否处在工作状态。可以在 ADSTART = 0 且 JADSTART = 0（指示 ADC 处于空闲状态）时重新配置 ADC。

ADSTART 通过硬件清除：

- 在软件触发的单次模式下（CONT = 0，EXTSEL = 0x0，DISCEN = 0）
  - 在转换序列结束后清零（EOS 标志位置 1）
- 在软件触发的单次模式下（CONT = 0，EXTSEL = 0x0，DISCEN = 1）
  - 在转换子组结束后清零（EOS 标志位置 1）
- 在所有其他情况下（CONT = x，EXTSEL = x）
  - 软件配置 ADSTP = 1 并执行之后清零（ADSTP 位清 0）

注：在连续模式下（CONT = 1），由于序列会自动重新启动，因此当 EOS 标志位置 1 时，ADSTART 位不会通过硬件清除。

如果在单次模式下选择了硬件事件触发（CONT = 0 且 EXTSEL != 0x00），当 EOS 标志位置 1 时，ADSTART 位不会通过硬件清除，这样软件无需为下一个硬件触发事件再次将 ADSTART 置 1。这样可以确保不会错过任何后续的硬件触发事件。

JADSTART 由硬件清除：

- 在软件触发的单次模式下（JEXTSEL = 0x0，JDISCEN = 0）
  - 在注入转换序列结束后清零（JEOS 标志位置 1）
- 在软件触发的单次模式下（JEXTSEL = 0x0，JDISCEN = 1）
  - 在注入转换序列结束后清零（JEOS 标志位置 1）
- 在所有其他情况下（JEXTSEL = x）
  - 软件配置 JADSTP = 1 并执行之后清零（JADSTP 位清 0）

注：选择软件触发事件时，如果 EOC 标志位仍为高，则不应将 ADSTART 位置 1。

15.4.11 停止正在进行的转换

软件决定是否停止转换，要停止正在进行的常规转换，应将 ADSTP 置 1，要停止正在进行的注入转换，应将 JADSTP 置 1。

停止转换将复位正在进行的 ADC 操作，随后可以重新配置 ADC（例如：更改通道选择或触发事件选择），为新操作做好准备。

注：可以在常规转换仍在进行时停止注入转换，反之亦然。这样可以在常规转换仍在进行时重新配置注入转换序列及触发事件，反之亦然。

如果 ADSTP 位由软件置 1，则会中止任何正在进行的常规转换，并会丢弃部分转换结果（ADC_DR 寄存器不会更新为当前转换结果）。

如果 JADSTP 位由软件置 1，则会中止任何正在进行的注入转换，并会丢弃部分转换结果（ADC_JDRy 寄存器不会更新为当前转换结果）。扫描序列也会中止并会复位（这意味着重

启 ADC 将重新开始新的序列）。

以上执行完毕后， ADSTP / ADSTART 位（常规转换）或 JADSTP / JADSTART 位（注入转换）会由硬件清除，软件必须查询 ADSTART（或 JADSTART）直到该位为 0，然后才能判定 ADC 已完全停止运行。

注：在自动注入模式下（JAUTO = 1），将 ADSTP 位置 1 会中止常规转换和注入转换（不可以使用 JADSTP 位）。

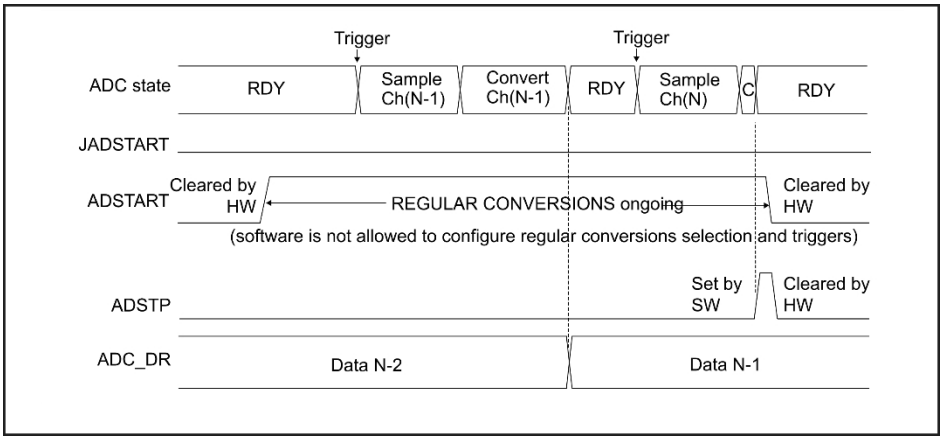


图 15-4 停止正在进行的常规转换

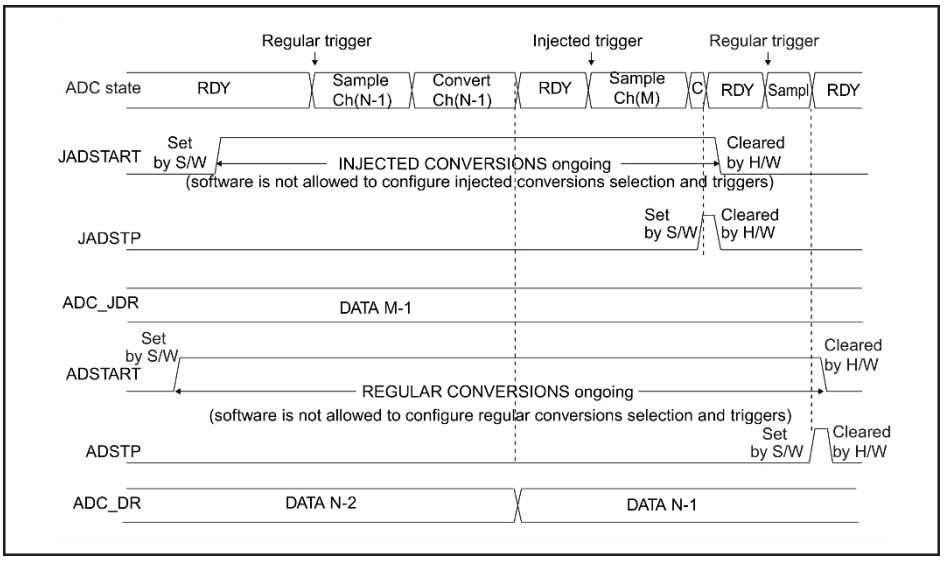


图 15-5 停止正在进行的常规转换和注入转换

15.4.12 外部触发转换和触发极性

可通过软件或外部事件（例如定时器捕获，输入引脚）触发单次转换或转换序列。如果 EXTEN [1:0]控制位（用于常规转换）或 JEXTEN [1:0]位（用于注入转换）不配置为 0x0，则外部事件能够以所选极性触发转换。

软件将 ADSTART 置 1 后，常规触发事件选择生效；软件将 JADSTART 置 1 之后，注入触发事件选择生效。

在转换进行时发生的任何硬件触发事件会被忽略。

- 如果 ADSTART = 0，会忽略任何发生的常规触发事件。
- 如果 JADSTART = 0，会忽略任何发生的注入触发事件。

表 15-3 提供了 EXTEN [1:0] 和 JEXTEN [1:0]值与触发极性之间的对应关系。

表 15-3 配置常规序列外部触发事件的极性

EXTEN[1:0]	来源
00	禁用硬件触发事件，启用软件触发事件
01	硬件触发事件，上升沿有效
10	硬件触发事件，下降沿有效
11	硬件触发事件，上升沿和下降沿均有效

注：不能实时更改常规触发事件的极性。

表 15-4 配置注入序列的外部触发事件的极性

JEXTEN[1:0]	来源
00	禁用硬件触发事件，启用软件触发事件
01	硬件触发事件，上升沿有效
10	硬件触发事件，下降沿有效
11	硬件触发事件，上升沿和下降沿均有效

EL [4:0]和 JEXTSEL [4:0]控制位用于选择常规序列与注入序列的外部触发事件，总共有 32 个事件来源。

常规序列转换可以被注入触发事件中断。

ADC0 与 ADC1 使用相同的外部触发事件，如图 15-6 所示。

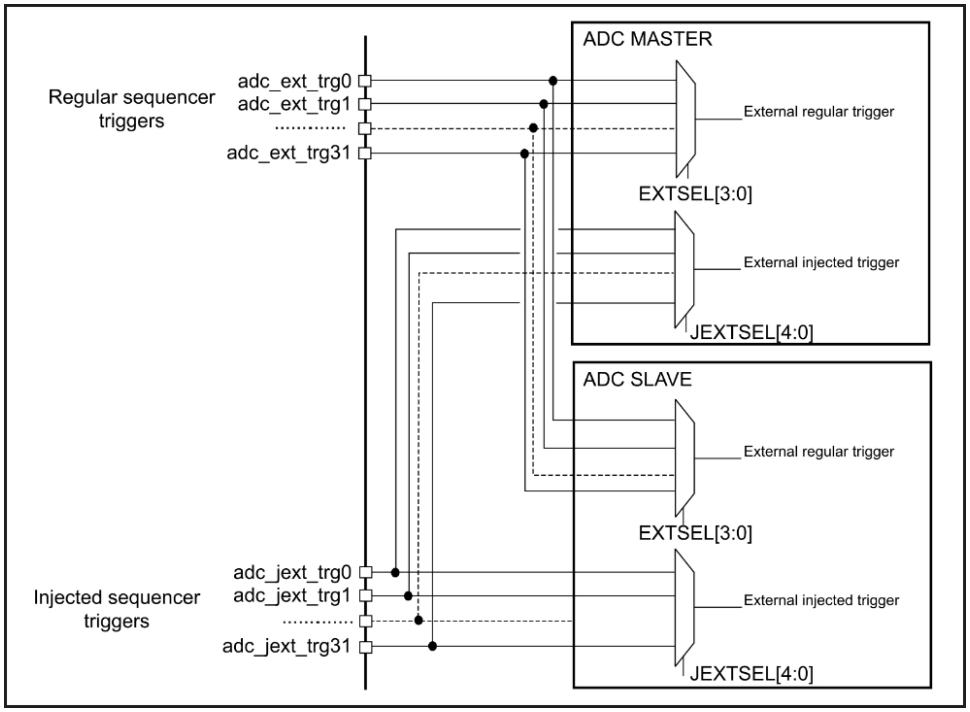


图 15-6 ADC0 与 ADC1 共用触发事件

表 15-5 至表 15-6 列出了 ADC0 / ADC1 的所有外部触发事件，包括常规触发事件和注入触发事件。

表 15-5 ADC0 / ADC1 常规触发事件

Name	Source	Type	EXTSEL[4:0]
adc_ext_evt0	TMR0_CC	来自片上定时器的内部信号	00000
adc_ext_evt1	TMR1_CC	来自片上定时器的内部信号	00001
adc_ext_evt2	TMR2_CC	来自片上定时器的内部信号	00010
adc_ext_evt3	TMR3_CC	来自片上定时器的内部信号	00011
adc_ext_evt4	TMR4_CC	来自片上定时器的内部信号	00100
adc_ext_evt5	TMR5_CC	来自片上定时器的内部信号	00101
adc_ext_evt6	TMR6_CC	来自片上定时器的内部信号	00110
adc_ext_evt7	TMR7_CC	来自片上定时器的内部信号	00111
adc_ext_evt8	TMR0_TRGO	来自片上定时器的内部信号	01000
adc_ext_evt9	TMR1_TRGO	来自片上定时器的内部信号	01001
adc_ext_evt10	TMR2_TRGO	来自片上定时器的内部信号	01010
adc_ext_evt11	TMR3_TRGO	来自片上定时器的内部信号	01011
adc_ext_evt12	TMR4_TRGO	来自片上定时器的内部信号	01100
adc_ext_evt13	TMR5_TRGO	来自片上定时器的内部信号	01101
adc_ext_evt14	TMR6_TRGO	来自片上定时器的内部信号	01110
adc_ext_evt15	TMR7_TRGO	来自片上定时器的内部信号	01111
adc_ext_evt16	HRPWM_ADC_TRG0	来自片上 PWM 的内部信号	10000
adc_ext_evt17	HRPWM_ADC_TRG1	来自片上 PWM 的内部信号	10001
adc_ext_evt18	HRPWM_ADC_TRG2	来自片上 PWM 的内部信号	10010
adc_ext_evt19	HRPWM_ADC_TRG3	来自片上 PWM 的内部信号	10011
adc_ext_evt20	HRPWM_ADC_TRG4	来自片上 PWM 的内部信号	10100
adc_ext_evt21	HRPWM_ADC_TRG5	来自片上 PWM 的内部信号	10101
adc_ext_evt22	HRPWM_ADC_TRG6	来自片上 PWM 的内部信号	10110
adc_ext_evt23	HRPWM_ADC_TRG7	来自片上 PWM 的内部信号	10111
adc_ext_evt24	PA8	外部引脚	11000
adc_ext_evt25	reserved	-	11001
adc_ext_evt26	reserved	-	11010
adc_ext_evt27	reserved	-	11011
adc_ext_evt28	reserved	-	11100
adc_ext_evt29	reserved	-	11101
adc_ext_evt30	reserved	-	11110
adc_ext_evt31	reserved	-	11111

表 15-6 ADC0 / ADC1 注入触发事件

Name	Source	Type	JEXTSEL[4:0]
adc_jext_evt0	TMR0_CC	来自片上定时器的内部信号	00000
adc_jext_evt1	TMR1_CC	来自片上定时器的内部信号	00001
adc_jext_evt2	TMR2_CC	来自片上定时器的内部信号	00010

adc_jext_evt3	TMR3_CC	来自片上定时器的内部信号	00011
adc_jext_evt4	TMR4_CC	来自片上定时器的内部信号	00100
adc_jext_evt5	TMR5_CC	来自片上定时器的内部信号	00101
adc_jext_evt6	TMR6_CC	来自片上定时器的内部信号	00110
adc_jext_evt7	TMR7_CC	来自片上定时器的内部信号	00111
adc_jext_evt8	TMR0_TRGO	来自片上定时器的内部信号	01000
adc_jext_evt9	TMR1_TRGO	来自片上定时器的内部信号	01001
adc_jext_evt10	TMR2_TRGO	来自片上定时器的内部信号	01010
adc_jext_evt11	TMR3_TRGO	来自片上定时器的内部信号	01011
adc_jext_evt12	TMR4_TRGO	来自片上定时器的内部信号	01100
adc_jext_evt13	TMR5_TRGO	来自片上定时器的内部信号	01101
adc_jext_evt14	TMR6_TRGO	来自片上定时器的内部信号	01110
adc_jext_evt15	TMR7_TRGO	来自片上定时器的内部信号	01111
adc_jext_evt16	HRPWM_ADC_TRG0	来自片上 PWM 的内部信号	10000
adc_jext_evt17	HRPWM_ADC_TRG1	来自片上 PWM 的内部信号	10001
adc_jext_evt18	HRPWM_ADC_TRG2	来自片上 PWM 的内部信号	10010
adc_jext_evt19	HRPWM_ADC_TRG3	来自片上 PWM 的内部信号	10011
adc_jext_evt20	HRPWM_ADC_TRG4	来自片上 PWM 的内部信号	10100
adc_jext_evt21	HRPWM_ADC_TRG5	来自片上 PWM 的内部信号	10101
adc_jext_evt22	HRPWM_ADC_TRG6	来自片上 PWM 的内部信号	10110
adc_jext_evt23	HRPWM_ADC_TRG7	来自片上 PWM 的内部信号	10111
adc_jext_evt24	PA9	外部引脚	11000
adc_jext_evt25	reserved	-	11001
adc_jext_evt26	reserved	-	11010
adc_jext_evt27	reserved	-	11011
adc_jext_evt28	reserved	-	11100
adc_jext_evt29	reserved	-	11101
adc_jext_evt30	reserved	-	11110
adc_jext_evt31	reserved	-	11111

15.4.13 注入序列管理

触发注入模式

要使用触发注入，必须将 ADC_CR0 寄存器中的 JAUTO 位清零。

1. 通过外部触发或将 ADC_CR0 寄存器中的 ADSTART 位置 1 来启动常规通道组转换。
2. 如果在常规通道组转换期间出现外部注入触发事件，或者 ADC_CR0 寄存器中的 JADSTART 位置 1，则当前的常规转换会复位，并会启动注入序列转换（所有注入通道都会转换一次）。
3. 然后，常规通道组的常规转换会从上次被中断的常规转换处恢复。
4. 如果在注入转换期间出现常规触发事件，注入转换不会中断，但在注入序列结束时会执行常规序列转换。注入序列相应的时序图如图 15-7 所示。

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长度。例如，如果注入序列长度为 36 个 ADC 时钟周期（即，采样时间为 6 个 ADC 时钟周期的一次转换），则触发事件的最小间隔不能小于 37 个 ADC 时钟周期。

自动注入模式

如果将 ADC_CR1 寄存器中的 JAUTO 位置 1，则注入组中的通道会在常规组通道之后自动转换。这可用于转换最多由 20 个转换构成的序列，这些转换在 ADC_SQRy 和 ADC_JSQR 寄存器中编程。

在该模式下，必须将 ADC_CR0 寄存器中的 ADSTART 位置 1 以开始常规转换，然后再进行注入转换（JADSTART 必须保持清零）。将 ADSTP 位置 1 会中止常规转换和注入转换（不得使用 JADSTP 位）。

在此模式下，必须禁止注入通道上的外部事件触发。

如果 CONT 位和 JAUTO 位均已置 1，则在转换注入通道之后会继续转换常规通道。

注：不能同时使用自动注入和不连续采样模式。

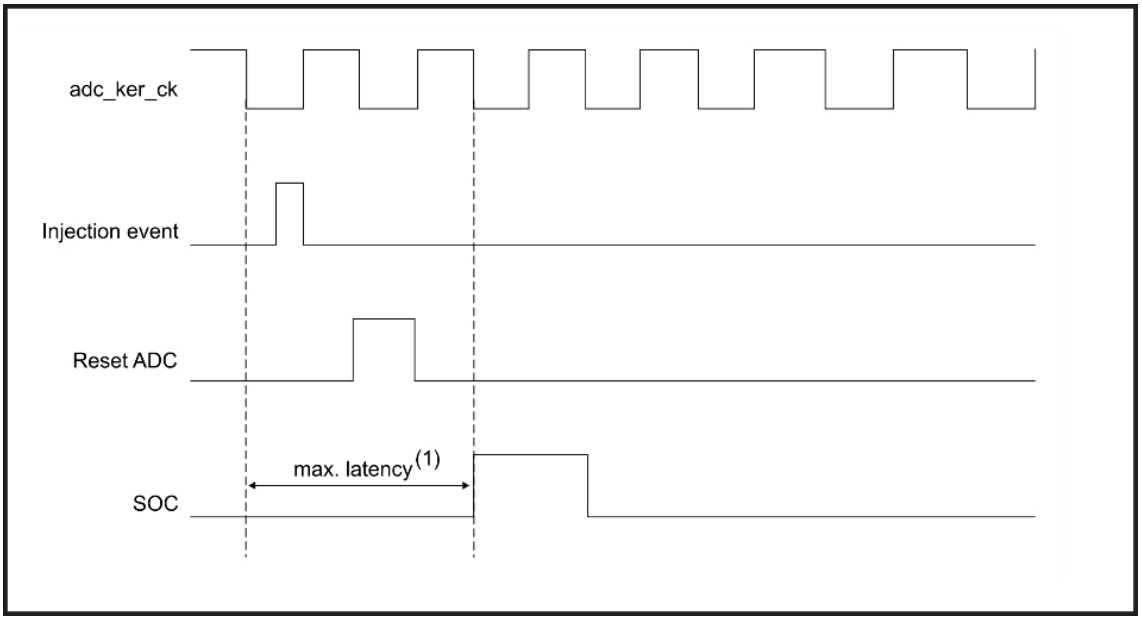


图 15-7 注入转换延迟

15.4.14 不连续模式

常规组模式

可将 ADC_CR1 寄存器中的 DISCEN 位置 1 来使能此模式。

该模式用于转换含有 n（n≤8）个转换的短序列（子组），该短序列是在 ADC_SQRy 寄存器中选择的转换序列的一部分。可通过写入 ADC_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_SQRy 寄存器中选择的接下来 n 个转换，直到序列中的所有

转换均完成为止。通过 ADC_LR 寄存器中的 L [3:0]位定义总序列长度。

示例：

- DISCEN = 1，n = 3，待转换的通道= 1、2、3、6、7、8、9、10、11
  - 第一次触发：转换的通道为 1、2、3，每次转换都会生成 EOC 事件。
  - 第二次触发：转换的通道为 6、7、8，每次转换都会生成 EOC 事件。
  - 第三次触发：转换的通道为 9、10、11，每次转换都会生成 EOC 事件，并会在通道 11 转换完成后生成 EOS 事件。
  - 第四次触发：转换的通道为 1、2、3，每次转换都会生成 EOC 事件。
- DISCEN = 0，待转换的通道= 1，2，3，6，7，8，9，10，11
  - 第一次触发：转换整个序列：通道 1、2、3、6、7、8、9、10 和 11。每次转换都会生成 EOC 事件，最后一次转换还会生成 EOS 事件。
  - 所有后续触发事件都将重启整个序列。

注：在不连续模式下转换常规组时，不会出现翻转（序列的最后一个子组的转换次数少于 n 次）。

转换完所有子组后，下一个触发事件将启动第一个子组的转换。在上述示例中，第四次重新转换了第一个子组中的通道 1、2 和 3。

不可以同时使能不连续模式和连续模式。如果同时使能两种模式（即 DISCEN = 1、CONT = 1），ADC 会认定连续模式已禁止并继续执行相关操作。

注入组模式

可将 ADC_CR1 寄存器中的 JDISCEN 位置 1 来使能此模式。在出现外部注入触发事件之后，该模式下会逐通道转换在 ADC_JSQR 寄存器中选择的序列，相当于不连续模式下常规通道“n”固定为 1 的情况。

出现外部触发事件时，将启动在 ADC_JSQR 寄存器中选择的下一个转换，直到序列中的所有转换均完成为止。通过 ADC_LR 寄存器中的 JLEN [1:0]位定义总序列长度。

示例：

- JDISCEN = 1，待转换的通道= 1、2、3
  - 第一次触发：转换通道 1，生成 JEOC 事件
  - 第二次触发：转换通道 2，生成 JEOC 事件
  - 第三次触发：转换通道 3，生成 JEOC 事件和 JEOS 事件

注：转换完所有注入通道后，下一个触发事件将启动第一个注入通道的转换。在上述示例中，第四次触发重新转换了第一个注入通道。

不可以同时使用自动注入模式和不连续模式：当 JAUTO 置 1 时， DISCEN 和 JDISCEN 位必须通过软件保持清零状态。

15.4.15 转换结束

每次出现常规转换结束（EOC）事件和注入转换结束（JEOC）事件时，ADC 都会通知应用程序。

新的常规转换数据出现在 ADC_DR 寄存器中后，ADC 会立即将 EOC 标志位置 1。如果 EOCIE 使能位置 1，可产生中断。EOC 标志位通过由软件向其写入 1 或读取 ADC_DR 寄存器的方式来清零。

新的注入转换数据出现在相应的 ADC_JDRy 寄存器中后，ADC 会立即将 JEOC 标志位置 1。如果 JEOCIE 使能位置 1，可产生中断。JEOC 标志位可通过由软件向其写入 1 或读取相应 ADC_JDRy 寄存器的方式来清零。

15.4.16 转换序列结束

每次出现常规序列结束（EOS）事件和注入序列结束（JEOS）事件时，ADC 都会通知应用程序。

常规转换序列的最后一个数据出现在 ADC_DR 寄存器中时，ADC 会立即将 EOS 标志位置 1。如果 EOSIE 使能位置 1，可产生中断。EOS 标志位可通过由软件向其写入 1 的方式来清零。

注入转换序列的最后一个数据出现在 ADC_JDRy 寄存器中时，ADC 会立即将 JEOS 标志位置 1。如果 JEOSIE 使能位置 1，可产生中断。JEOS 标志位可通过由软件向其写入 1 的方式来清零。

15.4.17 时序图示例

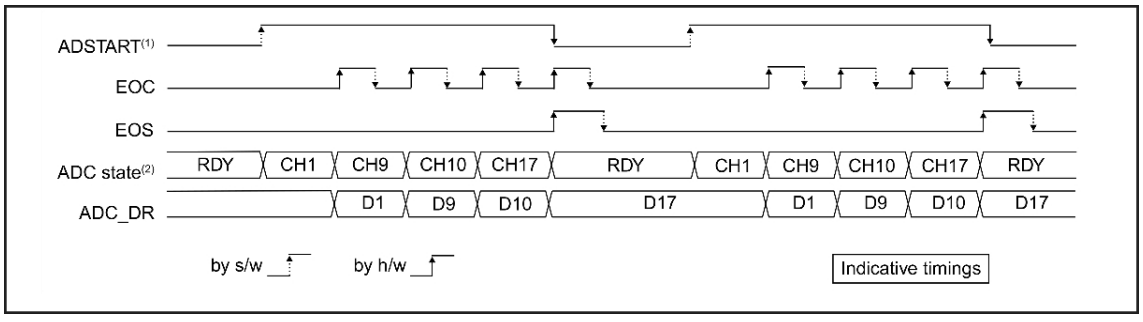


图 15-8 单次序列转换，软件触发

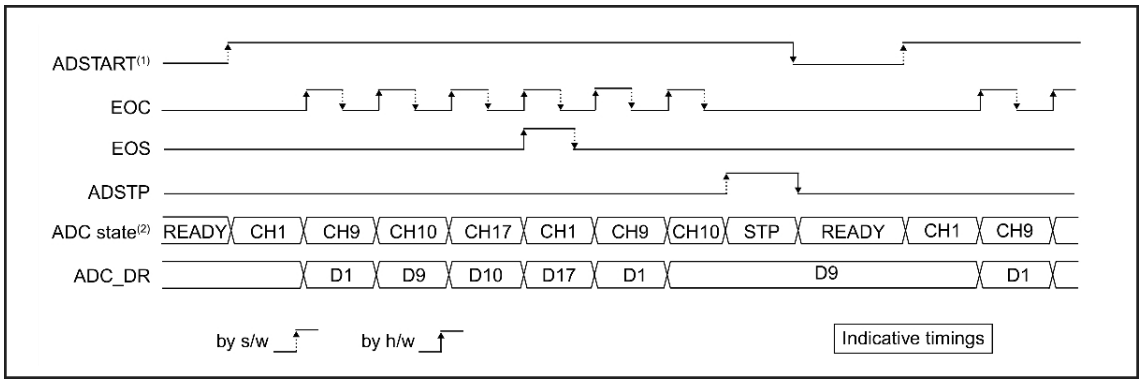


图 15-9 连续序列转换，软件触发

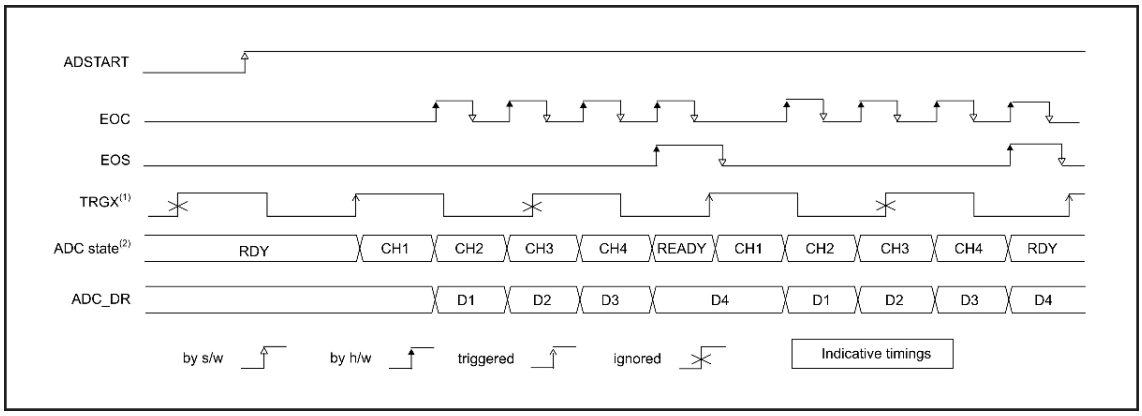


图 15-10 单次序列转换，硬件触发

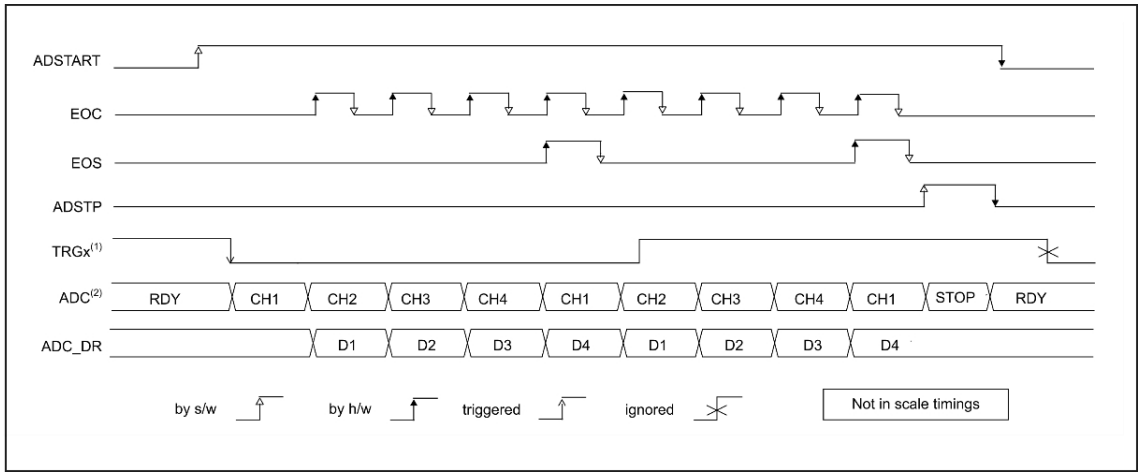


图 15-11 连续序列转换，硬件触发

15.4.18 数据管理

数据与对齐

每次常规转换通道结束时（发生 EOC 事件时），转换后数据的结果都会存储在宽度为 16 位的 ADC_DR 数据寄存器中。

每次注入转换通道结束时（发生 JEOC 事件时），转换后数据的结果都会存储在宽度为 16 位

的相应 ADC_JDRy 数据寄存器中。

正常模式下和过采样模式下，ADC 数据均为右对齐。

偏置补偿

每次转换结束时，转换后的数据都会经过偏置和增益补偿后，存储在相应的数据寄存器。每次转换后，将使用以下公式计算数据。

$$存储数据 = (转换数据 - Offset) \times \frac{Gain}{8192}$$

对各通道进行采样时可以使用不同的偏置补偿，偏置补偿系数总共有有四组，每组包含一个单端偏置与一个差分偏置，可通过 ADC_OFRy 或 ADC_DOFRy 寄存器进行配置。

各通道的补偿系数可通过 ADC_CALR0 和 ADC_CALR1 寄存器中的 CAL[1:0]来选择，可选的补偿系数为 ADC_OFRy/ADC_DOFRy（y=0..3）的其中一组。

这种情况下，转换后的数据会减去写入到 ADC_OFRy 或 ADC_DOFRy 中的用户自定义偏移。结果可能是负值，因此读取的数据为有符号数，SEXT 位代表扩展符号位。

注：过采样模式下也支持偏置补偿。对于过采样模式，会在应用 OVSS 右移之后减去偏置。

模拟看门狗模式支持偏置补偿，模拟看门狗会对偏置补偿之后的数据进行比较。

增益补偿

对各通道进行采样时可以使用不同的增益补偿，增益补偿系数总共有有四组，每组包含一个单端增益与一个差分增益，可通过 ADC_GCRy 和 ADC_DGCRy 寄存器进行配置。

各通道的补偿系数可通过 ADC_CALR0 和 ADC_CALR1 寄存器中的 CAL[1:0]来选择，可选的补偿系数为 ADC_OFRy/ADC_DOFRy（y=0..3）的其中一组。

由于 GAIN 可以编程为 0 至 65535，因此实际增益补偿因子的范围可以为 0 至 7.999878。

在将结果数据存储到 ADC_DR 或 ADC_JDRy 寄存器之前，补偿单元会对 LSB-1 值进行估算，对数据进行四舍五入以最大程度地减少误差。

增益补偿对于过采样也有效。当增益补偿用于过采样模式时，增益计算会在累加和右移操作之后执行以最大程度地减少功耗（增益计算仅执行一次，而不是在每次转换时执行）。

ADC 溢出

如果常规转换后的数据未在新转换数据可用之前（有 CPU 或 DMA）读取，会由溢出标志位（OVR）指示缓冲区溢出事件。

如果新转换完成时 EOC 标志位仍为 1，则 OVR 标志位会置 1。如果 OVRIE 使能位置 1，可产生中断。

如果发生溢出情况，ADC 仍会保持工作状态并可继续进行转换，除非通过软件将 ADSTP 位

置 1，从而停止并复位序列。

OVR 标志位可通过软件向其写入 1 的方式来清零。

可对控制位 OVRMOD 进行编程，从而配置发生溢出事件时是保留数据还是覆盖数据：

- OVRMOD = 0：溢出事件会保留数据寄存器的数据，防止其被覆盖：会保留原数据，并会丢弃而新的转换结果。如果 OVR 保持为 1，可继续正常进行转换，但还是会丢弃结果数据。
- OVRMOD = 1：数据寄存器会由上一次转换结果覆盖，之前未读取的数据会丢失。如果 OVR 保持为 1，可继续正常进行转换，并且 ADC_DR 寄存器将始终包含最新的转换数据。

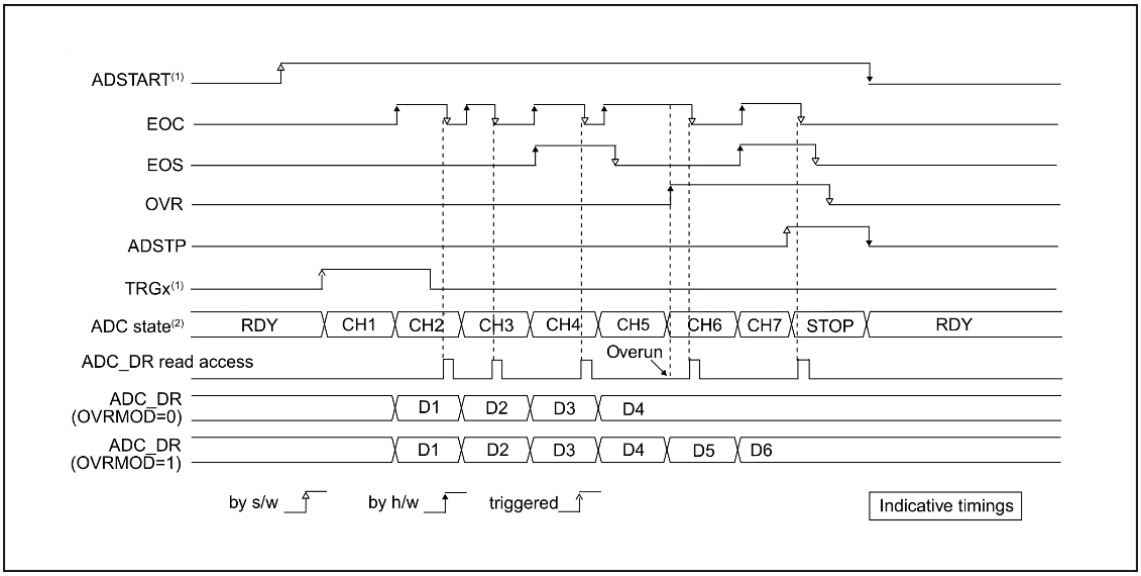


图 15-12 溢出示例

注：由于四条注入通道均有专用的数据寄存器，因此不会对注入通道进行溢出检测。

在不使用 DMA 的情况下管理转换序列

如果转换过程足够慢，则可使用软件处理转换序列。在这种情况下，软件必须使用 EOC 标志位及其相关中断来处理各个数据。每次转换完成时，EOC 都会置 1，并且可以读取 ADC_DR 寄存器。OVRMOD 应配置为 0，以便将溢出事件作为错误进行管理。

在不使用 DMA 且不溢出的情况下管理转换序列

ADC 在转换一个或多个通道时不是每次都读取数据的情况下，这可能会很有用（例如，存在模拟看门狗时）。在这种情况下，OVRMOD 位必须配置为 1，OVR 标志位应被软件忽略。溢出事件不会阻止 ADC 继续进行转换，ADC_DR 寄存器始终包含最新的转换结果。

使用 DMA 管理转换

由于转换得出的数据会存储在唯一的数据寄存器中，因此，对于多个通道的转换，使用 DMA 非常有帮助。这样可以避免丢失在下一次写入之前还未被读出的 ADC_DR 寄存器中的数据。

DMA 模式使能时，会在每次通道转换后发起 DMA 传输。DMA 传输分 12 路通道进行，每路通道可以独立配置传输地址、传输长度、传输模式，12 路通道分时进行传输。这样可以将转换后的数据从 ADC_DR 寄存器传输到用软件选择的目标位置。

尽管如此，如果因 DMA 无法及时处理 DMA 传输请求而发生数据溢出（OVR = 1），ADC 会停止发起 DMA 传输，新转换对应的数据不会通过 DMA 进行传输。这意味着可将传输到 RAM 的所有数据都视为有效数据。

根据 OVRMOD 位的配置，可保留或覆盖数据。

DMA 传输请求会停止，直至软件将 OVR 位清零。

根据应用用途的不同，推荐使用两种不同的 DMA 模式，使用 ADC_TCRx 寄存器的 CIRC 位配置：

- DMA 单次模式（CIRC 位 = 0）
  如果将 DMA 设置为传输固定数目的数据，应选择此模式。
- DMA 循环模式（CIRC 位 = 1）
  如果将 DMA 设置为传输连续数目的数据，应选择此模式。

DMA 单次模式（CIRCx= 0）

在该模式下，每次出现新的转换数据时，ADC 都会进行 DMA 传输，DMA 到达最后一个 DMA 传输操作时，即使转换已再次开始，ADC 也会停止 DMA 传输。

DMA 循环模式（CIRCx = 1）

在该模式下，每次数据寄存器中出现新的转换数据时，ADC 都会进行 DMA 传输，即使 DMA 已到达最后一次 DMA 传输操作也不例外。这样可以处理连续的模拟输入数据流。

ADC 与 ECU 联合使用时，DMA 必须配置为循环模式。

15.4.19 模拟看门狗

三个 AWD 模拟看门狗会监测一些通道是否保持在配置的电压范围（窗口）内。

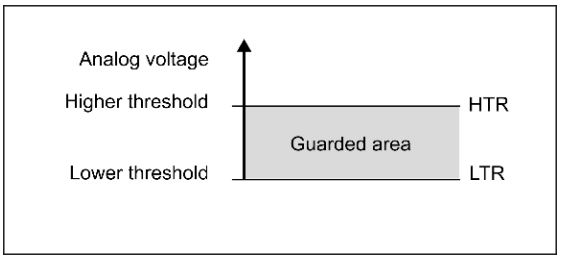


图 15-13 模拟看门狗的保护区域

AWDy 标志和中断

可通过将 ADC_IER 寄存器中的 AWDyIE 置 1 的方式分别为 3 个模拟看门狗使能中断。

AWDy（y = 0,1,2）标志位可通过软件向其写入 1 的方式来清零。

在增益补偿和偏移补偿之后，会将 ADC 转换结果与阈值上限和下限进行比较。

模拟看门狗的说明

模拟看门狗 0~2 非常灵活，可通过编程 AWDyCH [11:0]（y = 0~2）中的相应位来监测多条已选通道。

AWDCHy [11:0]（y = 0~2）的任意位置 1 时，会使能相应的看门狗。

阈值最高可达到 16 位（13 位分辨率，过采样，OSR=256），通过 ADCx_TR0、ADCx_TR1 和 ADCx_TR2 寄存器进行编程。

ADCx_AWDy_OUT 信号输出生成

每个模拟看门狗都关联到一个硬件触发事件 ADCx_AWDy_OUT（x = ADC 编号，y =看门狗编号），该信号直接连接到 HRPWM 的 EVT 输入（外部触发）。

当关联的模拟看门狗使能时，ADCx_AWDy_OUT 会激活：

- 当受监测的转换超出编程阈值时，ADCx_AWDy_OUT 会置 1。
- 在编程阈值范围内的下一受监测转换结束后，ADCx_AWDy_OUT 会复位（如果下一受监测转换仍超出编程阈值范围，该位仍保持置 1）。
- 关闭 ADC 时（将 EN_FADC 置 0 时）ADCx_AWDy_OUT 也保持复位状态。注意停止常规转换或注入转换（将 ADSTP 置 1 或 JADSTP 置 1）对 ADCx_AWDy_OUT 的生成没有任何影响。

注：AWDx 标志位由硬件置 1，并由软件复位；AWDx 标志位对 ADCx_AWDy_OUT 的生成没有影响（例如：如果软件未将 AWDx 标志清除，即 AWDx 标志位会保持为 1，此时 ADCx_AWDy_OUT 仍可翻转）。

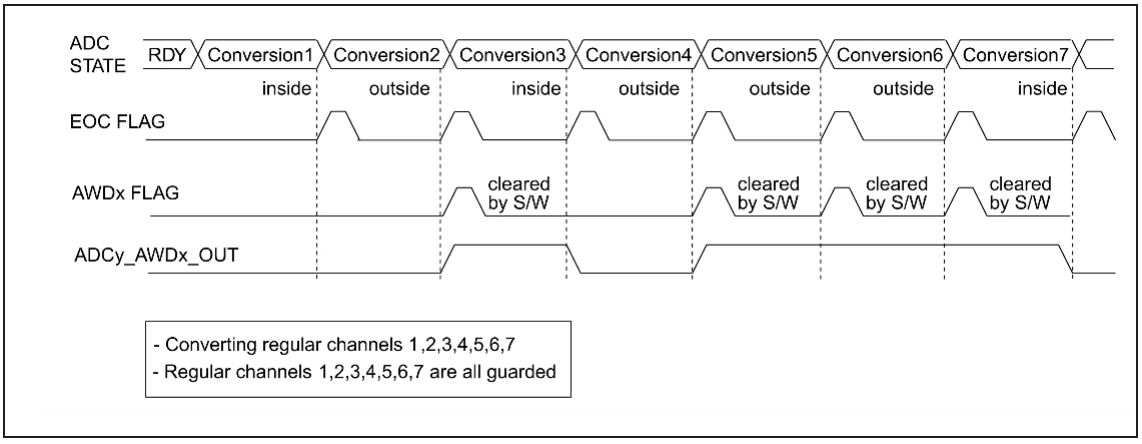


图 15-14　ADCx_AWDy_OUT 事件生成（多个常规通道上）

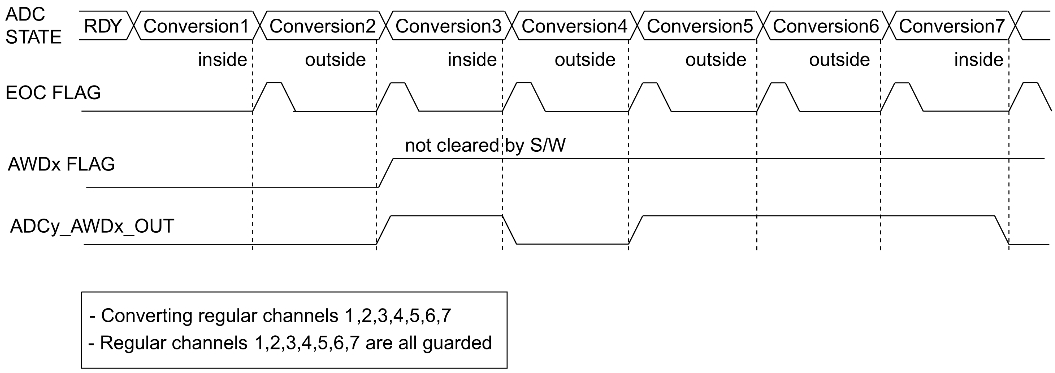


图 15-15　ADCx_AWDy_OUT 事件生成（AWDx 标志位未通过软件清零）

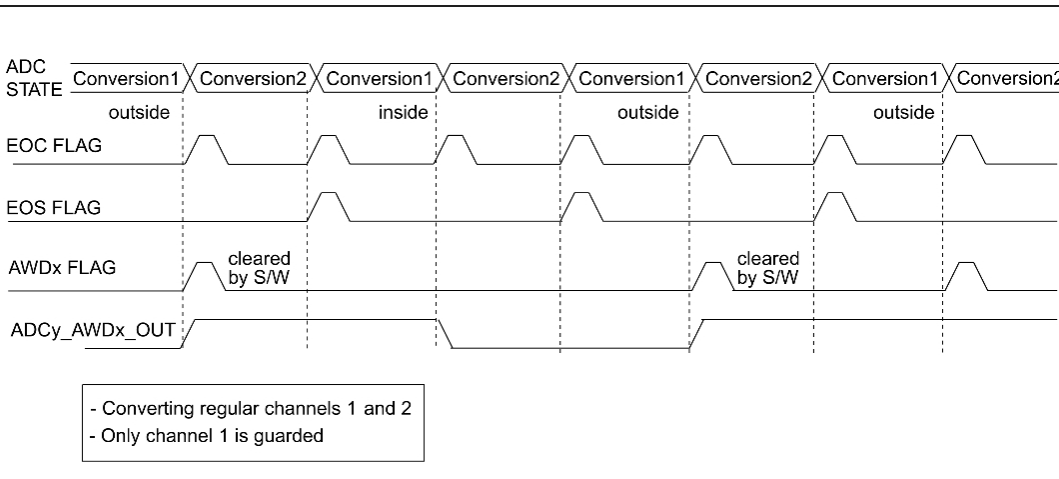


图 15-16　ADCx_AWDy_OUT 事件生成（单个常规通道上）

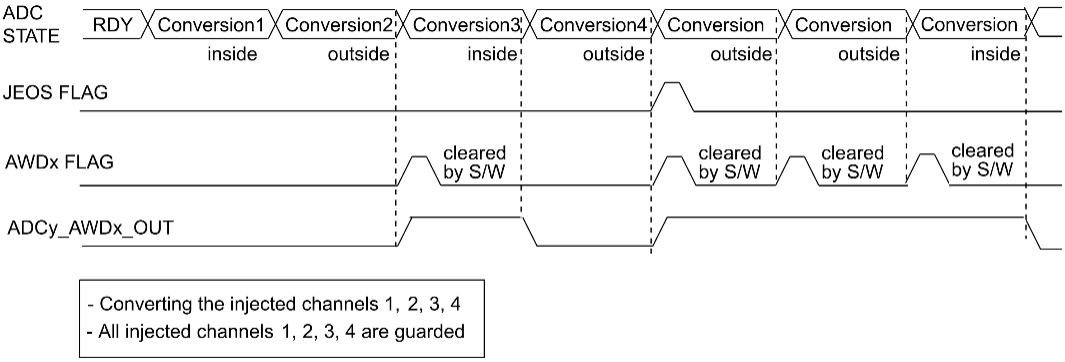


图 15-17　ADCx_AWDy_OUT 事件生成（所有注入通道上）

模拟看门狗阈值控制

ADC 正在进行转换时，同样可以更改 LTx [15:0]和 HTx [15:0]的配置。

加入增益和偏置补偿的模拟看门狗

加入增益和失调补偿后，模拟看门狗会在数据补偿之后再与阈值进行比较。

15.4.20 过采样器

过采样单元会进行数据预处理，以减轻 CPU 的负担。过采样单元还能处理多个转换，并计算多个转换结果的平均值，得到数据宽度增大（多达 16 位）的单个数据。

提供的结果采用以下形式，其中的 N 和 M 可以进行调整：

$$结果 = \frac{1}{M} \times \sum_{n=0}^{n=N-1} 转换(t_n)$$

允许通过硬件执行以下功能：计算平均值，降低数据速率，改善 SNR 以及基本滤波。

过采样率 N 通过 ADC_CR2 寄存器中的 OVRS [2:0]位定义，范围为 2x 到 256x。分频系数 M 通过向右移位来实现（最多可移 8 位），并且通过 ADC_CR2 寄存器中的 OVSS [3:0]位定义。

求和单元可得到最高 20 位（256x 12 位结果）的结果，结果进行右移，会使用移位后剩下的最低有效位四舍五入为最接近的数值，然后将得到的结果传输到 ADC_DR 数据寄存器中。

注：如果移位后的中间结果超过 16 位，则结果将被截断而不进行饱和。

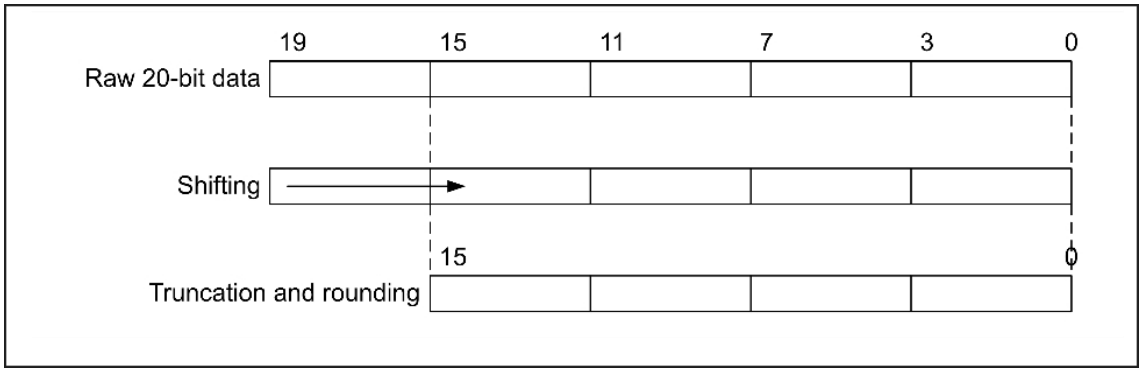


图 15-18　20-bit 到 16-bit 结果饱和处理

原始 20 位累加数据到最终 16 位计算结果的处理示例如图 19 所示。

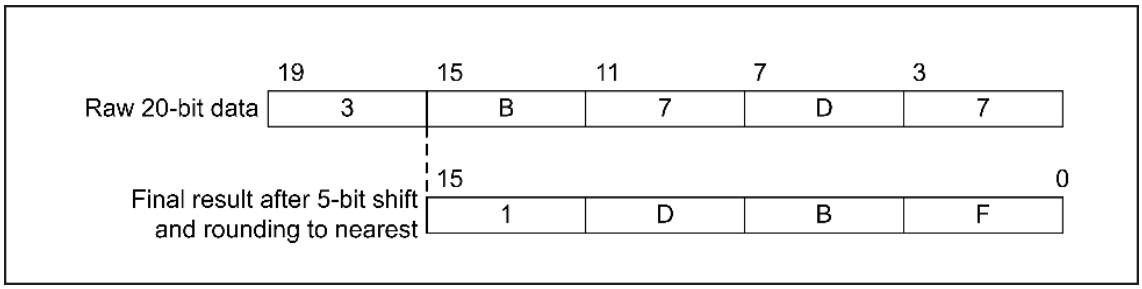


图 15-19　5-bit 移位及取整处理示例

原始转换数据等于 0xFFF 时，各种 N 和 M 组合的数据格式在表 15-7 中所示。

表 15-7 最大输出结果与 N 和 M 的关系（灰色单元格表示截断）

Over sampling ratio	Max Raw data	No-shift OVSS = 0000	1-bit shift OVSS =0001	2-bit shift OVSS =0010	3-bit shift OVSS =0011	4-bit shift OVSS =0100	5-bit shift OVSS =0101	6-bit shift OVSS =0110	7-bit shift OVSS =0111	8-bit shift OVSS =1000
2x	0x1FFE	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040	0x020
4x	0x3FFC	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040
8x	0x7FF8	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080
16x	0xFFF0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100
32x	0x1FFE0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200
64x	0x3FFC0	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400
128x	0x7FF80	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800
256x	0xFFF00	0xFF00	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF

过采样模式下的转换时序不会发生变化：在整个过采样序列中，采样时间保持不变。每完成 N 次转换都会提供一个新数据，等效延迟等于 N x T_{CONV} = N x（t_{SMPL} + t_{SAR}）。各标志位的置位情况如下：

- 如果过采样结果可用，每 N 次转换后都会发生转换结束事件（EOC）
- 过采样数据序列完成后（即 N x 序列长度次转换之后），会发生序列结束事件（EOS）

过采样模式支持的 ADC 工作模式

在过采样模式下，大部分 ADC 工作模式都会保留：

- 单次转换或连续模式转换
- 可由软件或外部事件启动 ADC 转换
- ADC 在转换过程中停止（中止）
- 通过 CPU 或 DMA 在支持溢出检测的情况下读取数据

触发式过采样

均值计算单元还可用于基本滤波，虽然它不是非常强大的滤波器（滚降缓慢、阻带衰减有限），但它可用作陷波滤波器，用于抑制恒定的寄生频率（通常来自电源或开关模式电源）。为此，可以使用 ADC_CR1 中的 TROVS 位使能特定的不连续模式，以获得由用户定义，且与转换时间本身无关的过采样频率。

图 20 显示了在不连续模式下如何响应触发从而开始转换。

如果 TROVS 位置 1，则会忽略 DISCEN 位的内容，并将该位视为 1。

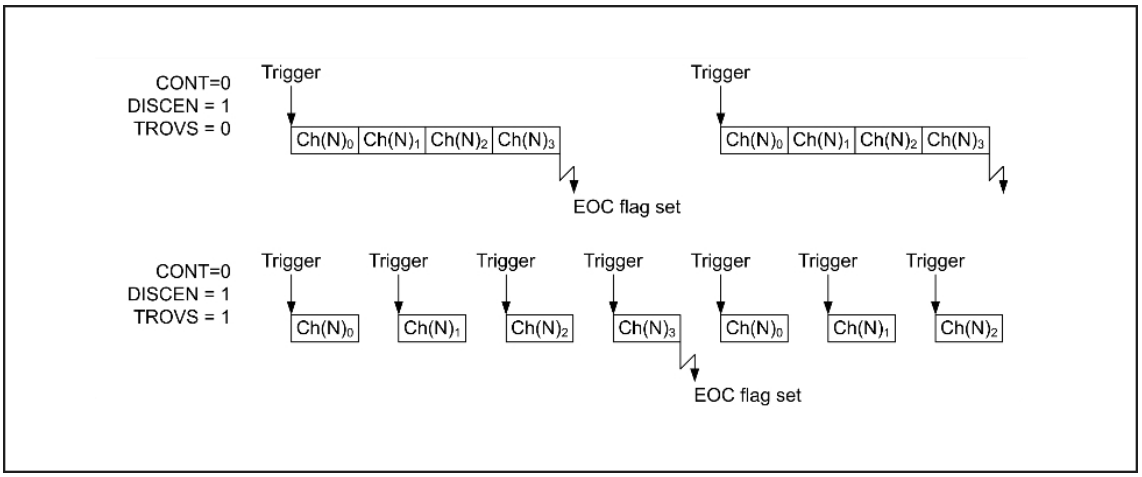


图 15-20 常规序列过采样（事件触发模式，TROVS=1）

过采样时的注入序列和常规序列

在过采样模式下，注入序列和常规序列可以执行不同操作。如果两个序列必须同时使用，则可为它们使能过采样并设定一些限制条件（与唯一的累加单元相关）。

常规序列过采样

常规过采样模式位 ROVSM 定义了常规过采样序列在被注入转换中断的情况下如何恢复：

- 在连续模式下，会从上一有效数据开始重新累加（在由于注入事件而中止常规转换之前）。这样可确保在任何注入频率下均可以完成过采样（假设触发事件之间至少可完成一次常规转换）；
- 在恢复模式下，会从 0 开始重新累加（会忽略之前的转换结果）。该模式可确保所有用于过采样的数据在单个时隙内进行连续地转换。需要注意的是，注入触发事件周期必须超过过采样周期。如果该条件未得到满足，将无法完成过采样，常规序列将被停止。

图 15-21 以 4x 过采样率为例进行了说明。

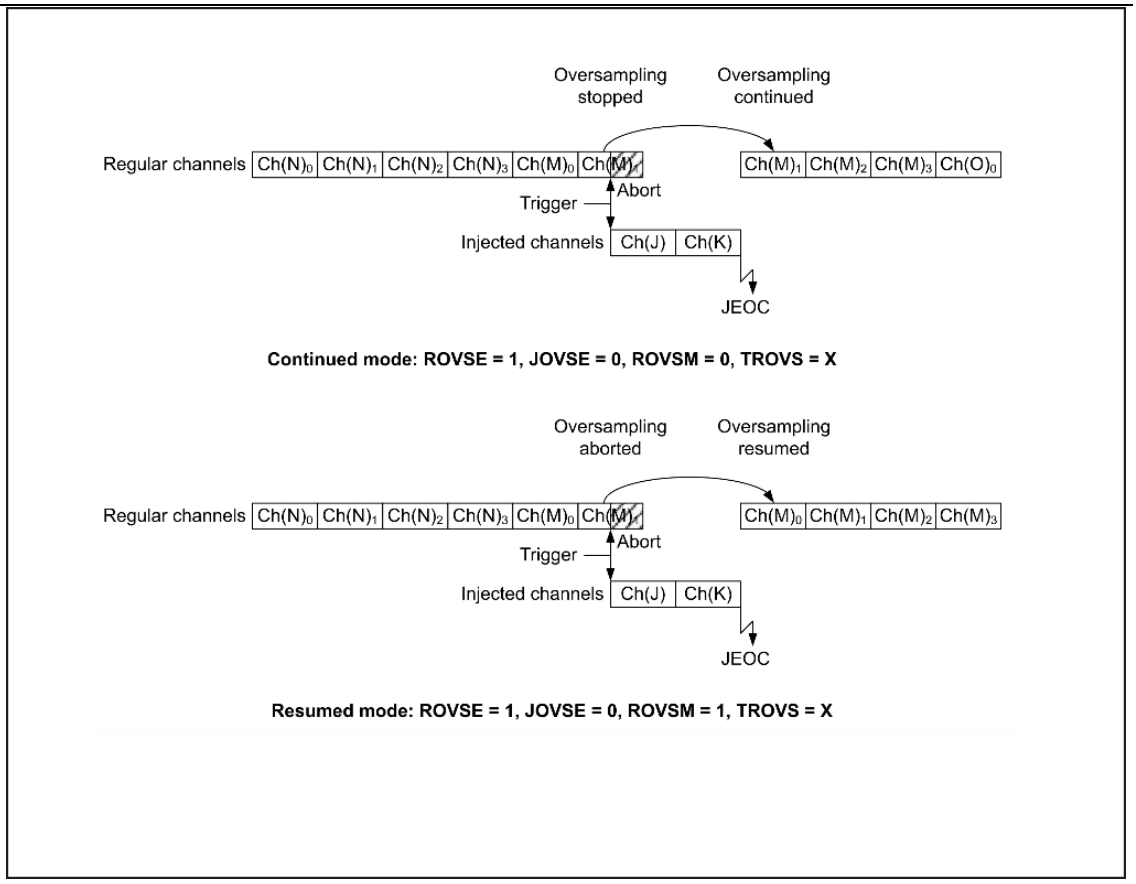


图 15-21　常规过采样模式（4x 过采样率）

仅对注入序列进行过采样

注入序列可以单独启动过采样，通过设置注入过采样位 JOVSE 启动注入序列过采样。

对常规序列和注入序列进行过采样

可以将 ROVSE 和 JOVSE 位都置 1。在这种情况下，常规过采样模式会强制进入恢复模式（忽略 ROVSM 位），如图 15-22 所示。

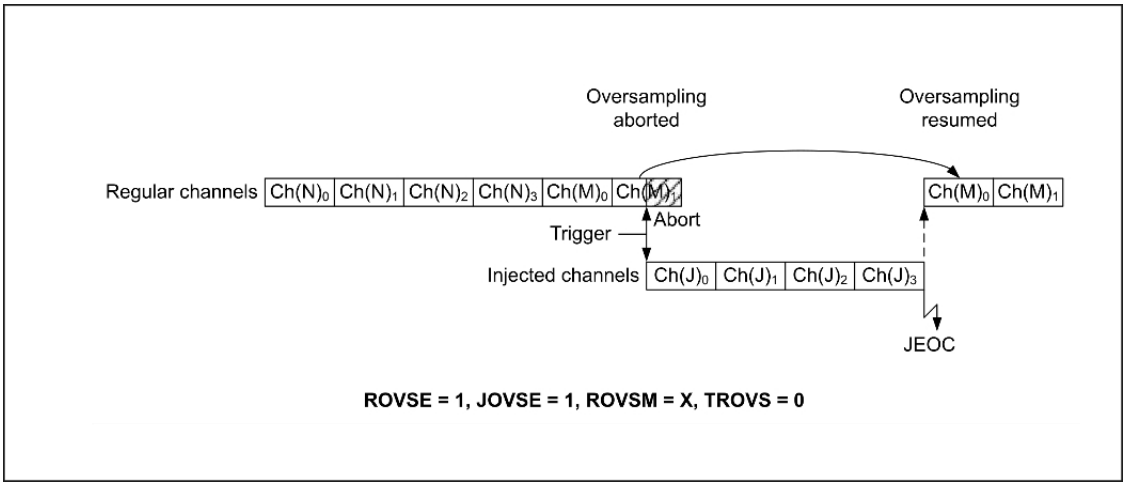


图 15-22　同时使用常规和注入过采样模式

触发式常规过采样支持注入转换

触发式常规过采样支持注入转换。在这种情况下，必须禁止注入过采样模式，并且会忽略 ROVSM 位（强制进入恢复模式），JOVSE 位必须复位。具体时序如图 15-23 所示。

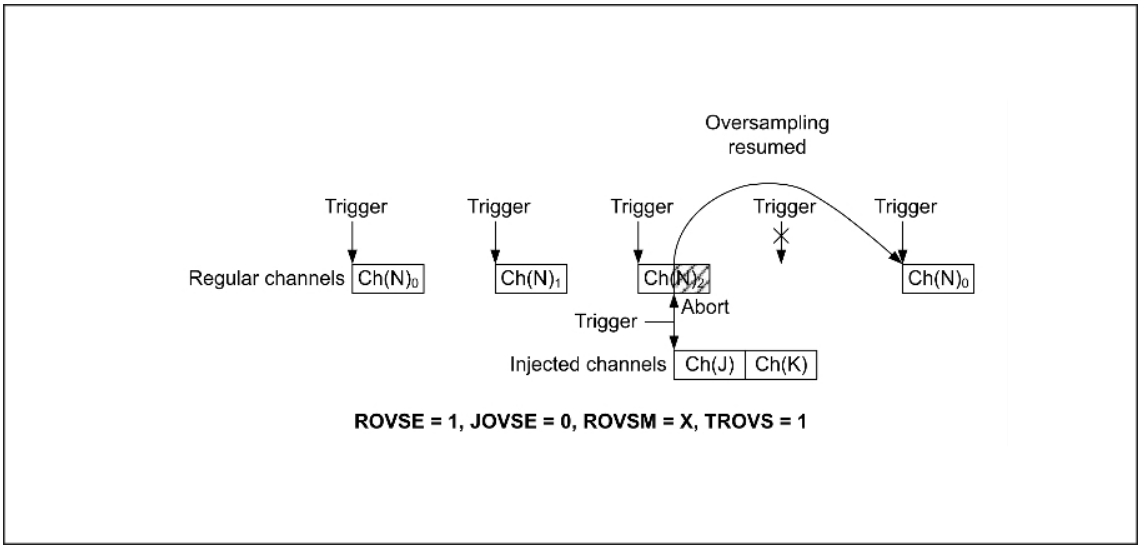


图 15-23 触发式常规过采样支持注入转换

自动注入模式

可以对自动注入序列进行过采样，并将所有转换结果存储在寄存器中，以节省 DMA 资源。使用该模式的前提条件是常规和注入过采样均激活：JAUTO = 1，ROVSE = 1 且 JOVSE = 1，不支持其他组合。在自动注入模式，会忽略 ROVSM 位。

图 15-24 显示了自动注入模式下过采样的转换顺序。

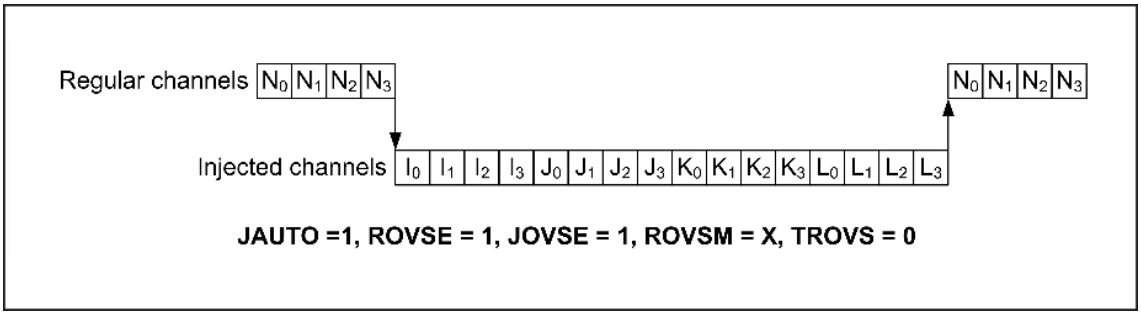


图 15-24 在自动注入模式下进行过采样

此外，还可以使用 TROVS 位使能触发式过采样。在这种情况下，ADC 必须进行如下配置：JAUTO = 1，DISCEN = 0，JDISCEN = 0，ROVSE = 1，JOVSE = 1 且 TROVS = 1。

组合模式汇总

表 15-8 汇总了所有组合，包括不支持的模式。

表 15-8　ADC 采样模式汇总

常规过采样 ROVSE	注入过采样 JOVSE	过采样模式 ROVSM	触发式过采样 TROVS	注释
1	0	0	0	常规序列连续模式
1	0	0	1	不支持
1	0	1	0	常规序列恢复模式
1	0	1	1	触发式常规序列恢复模式
1	1	0	X	不支持
1	1	1	0	注入序列和常规序列恢复模式
1	1	1	1	不支持
0	1	X	X	注入序列过采样

15.4.21 中断

- 每个 ADC 可以产生 8 个常规中断：
  - ADC 使能后准备就绪时（ADRDY 标志）
  - 常规组的任何转换结束时（EOC 标志）
  - 常规组的转换序列结束时（EOS 标志）
  - 注入组的任何转换结束时（JEOC 标志）
  - 注入组的转换序列结束时（JEOS 标志）
  - 发生数据溢出时（OVR 标志）
  - 发生模拟看门狗检测时（AWD1，AWD2 和 AWD3 标志）
- 每个 ADC 可以产生 12 个采样中断：
  - 模拟通道 x 的采样转换完成时（DONx 标志）
- 每个 ADC 可以产生 12 个半完成中断：
  - 模拟通道 x 的 DMA 半传输完成时（HLFx 标志）
- 每个 ADC 可以产生 12 个完成中断：
  - 模拟通道 x 的 DMA 传输完成时（FULx 标志）
- 每个中断都设置有单独的中断使能位，以实现最大的灵活性。

表 15-9 每个 ADC 的中断事件总结

中断向量	中断事件	中断标志	使能控制位	标志清除位
adcx_norm_int	常规转换结束事件	EOC	EOCIE	EOC
	常规序列结束事件	EOS	EOSIE	EOS
	注入转换结束事件	JEOC	JEOCIE	JEOC
	注入序列结束事件	JEOS	JEOSIE	JEOS
	数据溢出事件	OVR	OVRIE	OVR
	模拟看门狗 0 检测事件	AWD0	AWD0IE	AWD0
	模拟看门狗 1 检测事件	AWD1	AWD1IE	AWD1
	模拟看门狗 2 检测事件	AWD2	AWD2IE	AWD2
	ADC 准备就绪事件	ADRDY	ADRDYIE	ADRDY
adcx_samp_int	通道 0~11 采样完成事件	DONx	DONIEx	DONx
adcx_half_int	通道 0~11 DMA 半完成事件	HLFx	HLFIEx	HLFx
adcx_full_int	通道 0~11 DMA 完成事件	FULx	FULIEx	FULx

15.5 寄存器描述

15.5.1 寄存器列表

Name	Offset	Width	Description
ADC_CR0	0x00	32bits	ADC 控制寄存器 0
ADC_CR1	0x04	32bits	ADC 控制寄存器 1
ADC_CR2	0x08	32bits	ADC 控制寄存器 2
ADC_DIFSEL	0x0C	32bits	ADC 差分模式选择寄存器
ADC_IER	0x10	32bits	ADC 中断使能寄存器
ADC_ISR	0x14	32bits	ADC 中断状态寄存器
ADC_SIER	0x18	32bits	ADC 采样中断使能寄存器
ADC_SISR	0x1C	32bits	ADC 采样中断状态寄存器
ADC_SMPR0	0x20	32bits	ADC 采样时间寄存器 0
ADC_SMPR1	0x24	32bits	ADC 采样时间寄存器 1
ADC_CALR0	0x28	32bits	ADC 校准数据寄存器 0
ADC_CALR1	0x2C	32bits	ADC 校准数据寄存器 1
ADC_SQR0	0x30	32bits	ADC 常规序列寄存器 0
ADC_SQR1	0x34	32bits	ADC 常规序列寄存器 1
ADC_LR	0x38	32bits	ADC 常规序列长度寄存器
ADC_DR	0x3C	32bits	ADC 常规序列数据寄存器
ADC_JSQR	0x40	32bits	ADC 注入序列寄存器
ADC_JLR	0x44	32bits	ADC 注入序列长度寄存器
ADC_JDR0	0x50	32bits	ADC 注入序列数据寄存器 0
ADC_JDR1	0x54	32bits	ADC 注入序列数据寄存器 1
ADC_JDR2	0x58	32bits	ADC 注入序列数据寄存器 2
ADC_JDR3	0x5C	32bits	ADC 注入序列数据寄存器 3
ADC_TR0	0x60	32bits	ADC 看门狗 0 阈值寄存器
ADC_TR1	0x64	32bits	ADC 看门狗 1 阈值寄存器
ADC_TR2	0x68	32bits	ADC 看门狗 2 阈值寄存器
ADC_AWD0CR	0x70	32bits	ADC 看门狗 0 控制寄存器
ADC_AWD1CR	0x74	32bits	ADC 看门狗 1 控制寄存器
ADC_AWD2CR	0x78	32bits	ADC 看门狗 2 控制寄存器
ADC_OFR0	0x80	32bits	ADC 单端偏置补偿寄存器 0
ADC_OFR1	0x84	32bits	ADC 单端偏置补偿寄存器 1
ADC_OFR2	0x88	32bits	ADC 单端偏置补偿寄存器 2
ADC_OFR3	0x8C	32bits	ADC 单端偏置补偿寄存器 3
ADC_DOFR0	0x90	32bits	ADC 差分偏置补偿寄存器 0
ADC_DOFR1	0x94	32bits	ADC 差分偏置补偿寄存器 1
ADC_DOFR2	0x98	32bits	ADC 差分偏置补偿寄存器 2
ADC_DOFR3	0x9C	32bits	ADC 差分偏置补偿寄存器 3

ADC_GCR0	0xA0	32bits	ADC 单端增益系数寄存器 0
ADC_GCR1	0xA4	32bits	ADC 单端增益系数寄存器 1
ADC_GCR2	0xA8	32bits	ADC 单端增益系数寄存器 2
ADC_GCR3	0xAC	32bits	ADC 单端增益系数寄存器 3
ADC_DGCR0	0xB0	32bits	ADC 差分增益系数寄存器 0
ADC_DGCR1	0xB4	32bits	ADC 差分增益系数寄存器 1
ADC_DGCR2	0xB8	32bits	ADC 差分增益系数寄存器 2
ADC_DGCR3	0xBC	32bits	ADC 差分增益系数寄存器 3
ADC_ECR0	0xC0	32bits	ADC 事件控制寄存器 0
ADC_ECR1	0xC4	32bits	ADC 事件控制寄存器 1
ADC_ECR2	0xC8	32bits	ADC 事件控制寄存器 2
ADC_ECR3	0xCC	32bits	ADC 事件控制寄存器 3
ADC_CDR0	0xD0	32bits	ADC 通道数据寄存器 0
ADC_CDR1	0xD4	32bits	ADC 通道数据寄存器 1
ADC_CDR2	0xD8	32bits	ADC 通道数据寄存器 2
ADC_CDR3	0xDC	32bits	ADC 通道数据寄存器 3
ADC_CDR4	0xE0	32bits	ADC 通道数据寄存器 4
ADC_CDR5	0xE4	32bits	ADC 通道数据寄存器 5
ADC_CDR6	0xE8	32bits	ADC 通道数据寄存器 6
ADC_CDR7	0xEC	32bits	ADC 通道数据寄存器 7
ADC_CDR8	0xF0	32bits	ADC 通道数据寄存器 8
ADC_CDR9	0xF4	32bits	ADC 通道数据寄存器 9
ADC_CDR10	0xF8	32bits	ADC 通道数据寄存器 10
ADC_CDR11	0xFC	32bits	ADC 通道数据寄存器 11
ADC_HIER	0x100	32bits	ADC 半完成中断使能寄存器
ADC_HISR	0x104	32bits	ADC 半完成中断状态寄存器
ADC_FIER	0x108	32bits	ADC 完成中断使能寄存器
ADC_FISR	0x10C	32bits	ADC 完成中断状态寄存器
ADC_TCR0	0x110	32bits	ADC 传输控制寄存器 0
ADC_TAR0	0x114	32bits	ADC 传输地址寄存器 0
ADC_TLR0	0x11C	32bits	ADC 传输长度寄存器 0
ADC_TCR1	0x120	32bits	ADC 传输控制寄存器 1
ADC_TAR1	0x124	32bits	ADC 传输地址寄存器 1
ADC_TLR1	0x12C	32bits	ADC 传输长度寄存器 1
ADC_TCR2	0x130	32bits	ADC 传输控制寄存器 2
ADC_TAR2	0x134	32bits	ADC 传输地址寄存器 2
ADC_TLR2	0x13C	32bits	ADC 传输长度寄存器 2
ADC_TCR3	0x140	32bits	ADC 传输控制寄存器 3
ADC_TAR3	0x144	32bits	ADC 传输地址寄存器 3
ADC_TLR3	0x14C	32bits	ADC 传输长度寄存器 3
ADC_TCR4	0x150	32bits	ADC 传输控制寄存器 4
ADC_TAR4	0x154	32bits	ADC 传输地址寄存器 4
ADC_TLR4	0x15C	32bits	ADC 传输长度寄存器 4

ADC_TCR5	0x160	32bits	ADC 传输控制寄存器 5
ADC_TAR5	0x164	32bits	ADC 传输地址寄存器 5
ADC_TLR5	0x16C	32bits	ADC 传输长度寄存器 5
ADC_TCR6	0x170	32bits	ADC 传输控制寄存器 6
ADC_TAR6	0x174	32bits	ADC 传输地址寄存器 6
ADC_TLR6	0x17C	32bits	ADC 传输长度寄存器 6
ADC_TCR7	0x180	32bits	ADC 传输控制寄存器 7
ADC_TAR7	0x184	32bits	ADC 传输地址寄存器 7
ADC_TLR7	0x18C	32bits	ADC 传输长度寄存器 7
ADC_TCR8	0x190	32bits	ADC 传输控制寄存器 8
ADC_TAR8	0x194	32bits	ADC 传输地址寄存器 8
ADC_TLR8	0x19C	32bits	ADC 传输长度寄存器 8
ADC_TCR9	0x1A0	32bits	ADC 传输控制寄存器 9
ADC_TAR9	0x1A4	32bits	ADC 传输地址寄存器 9
ADC_TLR9	0x1AC	32bits	ADC 传输长度寄存器 9
ADC_TCR10	0x1B0	32bits	ADC 传输控制寄存器 10
ADC_TAR10	0x1B4	32bits	ADC 传输地址寄存器 10
ADC_TLR10	0x1BC	32bits	ADC 传输长度寄存器 10
ADC_TCR11	0x1C0	32bits	ADC 传输控制寄存器 11
ADC_TAR11	0x1C4	32bits	ADC 传输地址寄存器 11
ADC_TLR11	0x1CC	32bits	ADC 传输长度寄存器 11

15.5.2 寄存器详细描述

15.5.2.1 ADC 控制寄存器 0 (ADC_CR0)

- **Name**：ADC Control Register0
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-4]	Reserved	Reserved	Reserved	Reserved
[3]	JADSTP	R/WAC	0x0	ADC 停止注入转换命令，该位通过硬件清零 0：ADC 注入模式已停止 1：ADC 注入模式停止中
[2]	ADSTP	R/WAC	0x0	ADC 停止常规转换命令，该位通过硬件清零 0：ADC 常规模式已停止 1：ADC 常规模式停止中
[1]	JADSTART	R/WAC	0x0	ADC 开始注入转换命令，该位通过硬件清零 0：ADC 注入模式不运行 1：ADC 注入模式运行中
[0]	ADSTART	R/WAC	0x0	ADC 开始常规转换命令，该位通过硬件清零 0：ADC 常规模式不运行 1：ADC 常规模式运行中

15.5.2.2 ADC 控制寄存器 1 (ADC_CR1)

- **Name**：ADC Control Register1
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	Reserved	Reserved	Reserved	Reserved
[23]	SYNCEN	R/W	0x0	ADC0/ADC1 同步模式（仅 ADC1 配置有效） 0：非同步模式 1：同步模式
[22]	JAUTO	R/W	0x0	注入序列自动转换 0：自动注入不使能 1：自动注入使能
[21]	Reserved	Reserved	Reserved	Reserved
[20]	JDISCEN	R/W	0x0	注入序列的不连续采样模式 0：单次转换模式 1：非连续转换模式
[19-17]	DISCNUM	R/W	0x0	不连续采样模式通道计数 000：1 次转换 001：2 次转换 010：3 次转换 011：4 次转换 100：5 次转换 101：6 次转换 110：7 次转换 111：8 次转换
[16]	DISCEN	R/W	0x0	常规序列的单次/不连续转换模式 0：单次转换模式 1：不连续转换模式
[15-14]	Reserved	Reserved	Reserved	Reserved
[13]	CONT	R/W	0x0	常规序列的单次/连续转换模式 0：单次转换模式 1：连续转换模式
[12]	OVRMOD	R/W	0x0	数据溢出模式 0：数据保留模式 1：数据覆盖模式
[11]	Reserved	Reserved	Reserved	Reserved
[10]	ROVSM	R/W	0x0	常规通道的过采样模式 0：采样继续模式 1：采样恢复模式

[9]	TROVS	R/W	0x0	触发式常规过采样使能 0：1 次触发实现 1 次过采样 1：N 次触发实现 1 次过采样
[8-5]	OVSS	R/W	0x0	过采样数据移位位数 0000：不右移 0001：右移 1 位 0010：右移 2 位 0011：右移 3 位 0100：右移 4 位 0101：右移 5 位 0110：右移 6 位 0111：右移 7 位 1000：右移 8 位 1001-1111：Reserved
[4-2]	OVSR	R/W	0x0	过采样数据叠加比率 000：2x 过采样 001：4x 过采样 010：8x 过采样 011：16x 过采样 100：32x 过采样 101：64x 过采样 110：128x 过采样 111：256x 过采样
[1]	JOVSE	R/W	0x0	注入序列过采样使能 0：不使能 1：使能
[0]	ROVSE	R/W	0x0	常规序列过采样使能 0：不使能 1：使能

15.5.2.3 ADC 控制寄存器 2 (ADC_CR2)

- **Name**：ADC Control Register2
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-14]	Reserved	Reserved	Reserved	Reserved
[13-12]	ISEL	R/W	0x0	偏置电流档位（仅 ADC0 配置有效） 00：10 uA 01：12 uA 10：14 uA 11：8 uA
[11]	TBOMOD	R/W	0x0	Test Buffer 输出模式（仅 ADC0 配置有效） 0：输出开关断开 1：输出开关闭合
[10-8]	TBIMOD	R/W	0x0	Test Buffer 输入模式（仅 ADC0 配置有效） 000-011：输入 MUX 全部关闭 100：A 路 SH 输出 101：A 路 ADC 输出 110：B 路 SH 输出 111：B 路 ADC 输出
[7-5]	Reserved	Reserved	Reserved	Reserved
[4]	EN_TB	R/W	0x0	Test Buffer P 极和 N 极运放使能信号（仅 ADC0 配置有效） 0：不使能 1：使能
[3]	EN_CH	R/W	0x0	通道使能信号 0：不使能 1：使能
[2]	EN_FADC	R/W	0x0	FADC 模块使能信号 0：不使能 1：使能
[1]	EN_REF	R/W	0x0	Reference 模块使能信号 0：不使能 1：使能
[0]	EN_BIAS	R/W	0x0	内部偏置模块使能信号（仅 ADC0 配置有效） 0：不使能 1：使能

15.5.2.4 ADC 差分模式选择寄存器 (ADC_DIFSEL)

- **Name**：ADC Differential Select Register
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11-0]	DIFSEL	R/W	0x0	ADC 通道差分模式选择 DIFSEL[i] = 0x1：第 i 路通道为差分模式 DIFSEL[i] = 0x0：第 i 路通道为单端模式

15.5.2.5 ADC 中断使能寄存器 (ADC_IER)

- **Name**：ADC Interrupt Enable Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-9]	Reserved	Reserved	Reserved	Reserved
[8]	ADRDYIE	R/W	0x0	ADC 就绪中断使能 0：中断不使能 1：中断使能
[7]	AWD2IE	R/W	0x0	模拟看门狗 2 中断使能 0：中断不使能 1：中断使能
[6]	AWD1IE	R/W	0x0	模拟看门狗 1 中断使能 0：中断不使能 1：中断使能
[5]	AWD0IE	R/W	0x0	模拟看门狗 0 中断使能 0：中断不使能 1：中断使能
[4]	OVRIE	R/W	0x0	ADC 数据溢出中断使能 0：中断不使能 1：中断使能
[3]	JEOSIE	R/W	0x0	注入序列结束中断使能 0：中断不使能 1：中断使能
[2]	JEOCIE	R/W	0x0	注入转换结束中断使能 0：中断不使能 1：中断使能
[1]	EOSIE	R/W	0x0	常规序列结束中断使能 0：中断不使能 1：中断使能
[0]	EOCIE	R/W	0x0	常规转换结束中断使能 0：中断不使能 1：中断使能

15.5.2.6 ADC 中断状态寄存器(ADC_ISR)

- **Name**：ADC Interrupt Status Register
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-9]	Reserved	Reserved	Reserved	Reserved
[8]	ADRDY	R/W1C	0x0	ADC 就绪中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[7]	AWD2	R/W1C	0x0	模拟看门狗 2 中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[6]	AWD1	R/W1C	0x0	模拟看门狗 1 中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[5]	AWD0	R/W1C	0x0	模拟看门狗 0 中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[4]	OVR	R/W1C	0x0	ADC 数据溢出中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[3]	JEOS	R/W1C	0x0	注入序列结束中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[2]	JEOC	R/W1C	0x0	注入转换结束中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[1]	EOS	R/W1C	0x0	常规序列结束中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生

[0]	EOC	R/W1C	0x0	常规转换结束中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生

15.5.2.7 ADC 采样中断使能寄存器 (ADC_SIER)

- **Name**：ADC Sample Interrupt Enable Register
- **Size**：32bits
- **Offset**：0x18
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	DONIE11	R/W	0x0	通道 11 转换完成中断使能 0：中断不使能 1：中断使能
[10]	DONIE10	R/W	0x0	通道 10 转换完成中断使能 0：中断不使能 1：中断使能
[9]	DONIE9	R/W	0x0	通道 9 转换完成中断使能 0：中断不使能 1：中断使能
[8]	DONIE8	R/W	0x0	通道 8 转换完成中断使能 0：中断不使能 1：中断使能
[7]	DONIE7	R/W	0x0	通道 7 转换完成中断使能 0：中断不使能 1：中断使能
[6]	DONIE6	R/W	0x0	通道 6 转换完成中断使能 0：中断不使能 1：中断使能
[5]	DONIE5	R/W	0x0	通道 5 转换完成中断使能 0：中断不使能 1：中断使能
[4]	DONIE4	R/W	0x0	通道 4 转换完成中断使能 0：中断不使能 1：中断使能
[3]	DONIE3	R/W	0x0	通道 3 转换完成中断使能 0：中断不使能 1：中断使能
[2]	DONIE2	R/W	0x0	通道 2 转换完成中断使能 0：中断不使能 1：中断使能
[1]	DONIE1	R/W	0x0	通道 1 转换完成中断使能 0：中断不使能 1：中断使能

[0]	DONIE0	R/W	0x0	通道 0 转换完成中断使能 0：中断不使能 1：中断使能

15.5.2.8 ADC 采样中断状态寄存器 (ADC_SISR)

- **Name**：ADC Sample Interrupt Status Register
- **Size**：32bits
- **Offset**：0x1C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	DONIE11	R/W1C	0x0	通道 11 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[10]	DONIE10	R/W1C	0x0	通道 10 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[9]	DONIE9	R/W1C	0x0	通道 9 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[8]	DONIE8	R/W1C	0x0	通道 8 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[7]	DONIE7	R/W1C	0x0	通道 7 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[6]	DONIE6	R/W1C	0x0	通道 6 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[5]	DONIE5	R/W1C	0x0	通道 5 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[4]	DONIE4	R/W1C	0x0	通道 4 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生

[3]	DONIE3	R/W1C	0x0	通道 3 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[2]	DONIE2	R/W1C	0x0	通道 2 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[1]	DONIE1	R/W1C	0x0	通道 1 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[0]	DONIE0	R/W1C	0x0	通道 0 转换完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生

15.5.2.9 ADC 采样时间寄存器 0 (ADC_SMPR0)

- **Name**：ADC Sample Time Register 0
- **Size**：32bits
- **Offset**：0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	Reserved	Reserved	Reserved	Reserved
[30-28]	SMP7	R/W	0x0	通道 7 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[27]	Reserved	Reserved	Reserved	Reserved
[26-24]	SMP6	R/W	0x0	通道 6 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[23]	Reserved	Reserved	Reserved	Reserved
[22-20]	SMP5	R/W	0x0	通道 5 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[19]	Reserved	Reserved	Reserved	Reserved

[18-16]	SMP4	R/W	0x0	通道 4 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[15]	Reserved	Reserved	Reserved	Reserved
[14-12]	SMP3	R/W	0x0	通道 3 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[11]	Reserved	Reserved	Reserved	Reserved
[10-8]	SMP2	R/W	0x0	通道 2 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[7]	Reserved	Reserved	Reserved	Reserved
[6-4]	SMP1	R/W	0x0	通道 1 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[3]	Reserved	Reserved	Reserved	Reserved

[2-0]	SMP0	R/W	0x0	通道 0 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期

15.5.2.10 ADC 采样时间寄存器 1 (ADC_SMPR1)

- **Name**：ADC Sample Time Register 1
- **Size**：32bits
- **Offset**：0x24
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-15]	Reserved	Reserved	Reserved	Reserved
[14-12]	SMP11	R/W	0x0	通道 11 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[11]	Reserved	Reserved	Reserved	Reserved
[10-8]	SMP10	R/W	0x0	通道 10 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[7]	Reserved	Reserved	Reserved	Reserved
[6-4]	SMP9	R/W	0x0	通道 9 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期
[3]	Reserved	Reserved	Reserved	Reserved

[2-0]	SMP8	R/W	0x0	通道 8 采样时间选择位。 000：6 个 ADC_CLK 周期 001：18 个 ADC_CLK 周期 010：42 个 ADC_CLK 周期 011：90 个 ADC_CLK 周期 100：186 个 ADC_CLK 周期 101：378 个 ADC_CLK 周期 110：762 个 ADC_CLK 周期 111：1530 个 ADC_CLK 周期

15.5.2.11 ADC 校准数据寄存器 0 (ADC_CALR0)

- **Name**：ADC Calibration Data Register 0
- **Size**：32bits
- **Offset**：0x28
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-30]	Reserved	Reserved	Reserved	Reserved
[29-28]	CAL7	R/W	0x0	通道 7 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3
[27-26]	Reserved	Reserved	Reserved	Reserved
[25-24]	CAL6	R/W	0x0	通道 6 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3
[23-22]	Reserved	Reserved	Reserved	Reserved
[21-20]	CAL5	R/W	0x0	通道 5 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3
[19-18]	Reserved	Reserved	Reserved	Reserved
[17-16]	CAL4	R/W	0x0	通道 4 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3
[15-14]	Reserved	Reserved	Reserved	Reserved
[13-12]	CAL3	R/W	0x0	通道 3 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3
[11-10]	Reserved	Reserved	Reserved	Reserved
[9-8]	CAL2	R/W	0x0	通道 2 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3

[7-6]	Reserved	Reserved	Reserved	Reserved
[5-4]	CAL1	R/W	0x0	通道 1 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3
[3-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	CAL0	R/W	0x0	通道 0 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3

15.5.2.12 ADC 校准数据寄存器 1 (ADC_CALR1)

- **Name**：ADC Calibration Data Register 1
- **Size**：32bits
- **Offset**：0x2C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-14]	Reserved	Reserved	Reserved	Reserved
[13-12]	CAL11	R/W	0x0	通道 11 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3
[11-10]	Reserved	Reserved	Reserved	Reserved
[9-8]	CAL10	R/W	0x0	通道 10 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3
[7-6]	Reserved	Reserved	Reserved	Reserved
[5-4]	CAL9	R/W	0x0	通道 9 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3
[3-2]	Reserved	Reserved	Reserved	Reserved
[1-0]	CAL8	R/W	0x0	通道 8 校准系数选择位。 00：OFFSET0 / GAIN COEFF0 01：OFFSET1 / GAIN COEFF1 10：OFFSET2 / GAIN COEFF2 11：OFFSET3 / GAIN COEFF3

15.5.2.13 ADC 常规序列寄存器 0 (ADC_SQR0)

- **Name**：ADC Regular Sequence Register0
- **Size**：32bits
- **Offset**：0x30
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-28]	SQ8	R/W	0x0	常规序列中的第 8 次转换
[27-24]	SQ7	R/W	0x0	常规序列中的第 7 次转换
[23-20]	SQ6	R/W	0x0	常规序列中的第 6 次转换
[19-16]	SQ5	R/W	0x0	常规序列中的第 5 次转换
[15-12]	SQ4	R/W	0x0	常规序列中的第 4 次转换
[11-8]	SQ3	R/W	0x0	常规序列中的第 3 次转换
[7-4]	SQ2	R/W	0x0	常规序列中的第 2 次转换
[3-0]	SQ1	R/W	0x0	常规序列中的第 1 次转换

15.5.2.14 ADC 常规序列寄存器 1 (ADC_SQR1)

- **Name**：ADC Regular Sequence Register1
- **Size**：32bits
- **Offset**：0x34
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-28]	SQ16	R/W	0x0	常规序列中的第 16 次转换
[27-24]	SQ15	R/W	0x0	常规序列中的第 15 次转换
[23-20]	SQ14	R/W	0x0	常规序列中的第 14 次转换
[19-16]	SQ13	R/W	0x0	常规序列中的第 13 次转换
[15-12]	SQ12	R/W	0x0	常规序列中的第 12 次转换
[11-8]	SQ11	R/W	0x0	常规序列中的第 11 次转换
[7-4]	SQ10	R/W	0x0	常规序列中的第 10 次转换
[3-0]	SQ9	R/W	0x0	常规序列中的第 9 次转换

15.5.2.15 ADC 常规序列长度寄存器 (ADC_LR)

- **Name**：ADC Regular Length Register
- **Size**：32bits
- **Offset**：0x38
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11-8]	LEN	R/W	0x0	常规序列的长度 0x0：1 次转换 0x1：2 次转换 …… 0xf：16 次转换
[7]	Reserved	Reserved	Reserved	Reserved
[6-5]	EXTEN	R/W	0x0	常规序列的硬件触发使能和极性选择 0x0：硬件触发不使能 0x1：硬件触发上升沿有效 0x2：硬件触发下降沿有效 0x3：硬件触发上升 / 下降沿有效
[4-0]	EXTSEL	R/W	0x0	常规序列的硬件触发选择 0x1：Event 1 …… 0x1f：Event 31

15.5.2.16 ADC 常规序列数据寄存器 (ADC_DR)

- **Name**：ADC Regular Data Register
- **Size**：32bits
- **Offset**：0x3C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R/W	0x0	常规序列已转换的数据

15.5.2.17 ADC 注入序列寄存器 (ADC_JSQR)

- **Name**：ADC Injected Sequence Register
- **Size**：32bits
- **Offset**：0x40
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-12]	JSQ4	R/W	0x0	注入序列中的第 4 次转换
[11-8]	JSQ3	R/W	0x0	注入序列中的第 3 次转换
[7-4]	JSQ2	R/W	0x0	注入序列中的第 2 次转换
[3-0]	JSQ1	R/W	0x0	注入序列中的第 1 次转换

15.5.2.18 ADC 注入序列长度寄存器 (ADC_JLR)

- **Name**：ADC Injected Length Register
- **Size**：32bits
- **Offset**：0x44
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-10]	Reserved	Reserved	Reserved	Reserved
[9-8]	JLEN	R/W	0x0	注入序列的长度 0x0：1 次转换 0x1：2 次转换 0x2：3 次转换 0x3：4 次转换
[7]	Reserved	Reserved	Reserved	Reserved
[6-5]	JEXTEN	R/W	0x0	注入序列的硬件触发使能和极性选择 0x0：硬件触发不使能 0x1：硬件触发上升沿有效 0x2：硬件触发下降沿有效 0x3：硬件触发上升 / 下降沿有效
[4-0]	JEXTSEL	R/W	0x0	注入序列的硬件触发选择 0x0：Event 0 0x1：Event 1 …… 0x1f：Event 31

15.5.2.19 ADC 注入序列数据寄存器 0 (ADC_JDR0)

- **Name**：ADC Injected Data Register0
- **Size**：32bits
- **Offset**：0x50
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R/W	0x0	注入序列的第 0 个已转换数据

15.5.2.20 ADC 注入序列数据寄存器 1 (ADC_JDR1)

- **Name**：ADC Injected Data Register1
- **Size**：32bits
- **Offset**：0x54
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R/W	0x0	注入序列的第 1 个已转换数据

15.5.2.21 ADC 注入序列数据寄存器 2 (ADC_JDR2)

- **Name**：ADC Injected Data Register2
- **Size**：32bits
- **Offset**：0x58
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R/W	0x0	注入序列的第 2 个已转换数据

15.5.2.22 ADC 注入序列数据寄存器 3 (ADC_JDR3)

- **Name**：ADC Injected Data Register3
- **Size**：32bits
- **Offset**：0x5C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R/W	0x0	注入序列的第 3 个已转换数据

15.5.2.23 ADC 看门狗 0 阈值寄存器 (ADC_TR0)

- **Name**：ADC Watchdog0 Threshold Register
- **Size**：32bits
- **Offset**：0x60
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	HT0	R/W	0x0	模拟看门狗 0 阈值上限 模拟看门狗 0 监测的通道转换值高于 HT0 时产生事件 / 中断 格式为 16 位有符号数，范围[-32768, 32767]
[15-0]	LT0	R/W	0x0	模拟看门狗 0 阈值下限 模拟看门狗 0 监测的通道转换值低于 LT0 时产生事件 / 中断 格式为 16 位有符号数，范围[-32768, 32767]

15.5.2.24 ADC 看门狗 1 阈值寄存器 (ADC_TR1)

- **Name**：ADC Watchdog1 Threshold Register
- **Size**：32bits
- **Offset**：0x64
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	HT1	R/W	0x0	模拟看门狗 1 阈值上限 模拟看门狗 1 监测的通道转换值高于 HT1 时产生事件 / 中断 格式为 16 位有符号数，范围[-32768, 32767]
[15-0]	LT1	R/W	0x0	模拟看门狗 1 阈值下限 模拟看门狗 1 监测的通道转换值低于 LT1 时产生事件 / 中断 格式为 16 位有符号数，范围[-32768, 32767]

15.5.2.25 ADC 看门狗 2 阈值寄存器 (ADC_TR2)

- **Name**：ADC Watchdog2 Threshold Register
- **Size**：32bits
- **Offset**：0x68
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	HT2	R/W	0x0	模拟看门狗 2 阈值上限 模拟看门狗 2 监测的通道转换值高于 HT2 时产生事件 / 中断 格式为 16 位有符号数，范围[-32768, 32767]
[15-0]	LT2	R/W	0x0	模拟看门狗 2 阈值下限 模拟看门狗 2 监测的通道转换值低于 LT2 时产生事件 / 中断 格式为 16 位有符号数，范围[-32768, 32767]

15.5.2.26 ADC 看门狗 0 控制寄存器 (ADC_AWD0CR)

- **Name**：ADC Watchdog0 Control Register
- **Size**：32bits
- **Offset**：0x70
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19-16]	AWDFILT	R/W	0x0	模拟看门狗 0 滤波点数 0x0：模拟看门狗 0 不做滤波 0x1：模拟看门狗 0 监测 2 次超过阈值的转换产生事件 / 中断 …… 0xf：模拟看门狗 0 监测 16 次超过阈值的转换产生事件 / 中断
[15-12]	Reserved	Reserved	Reserved	Reserved
[11-0]	AWD0CH	R/W	0x0	模拟看门狗 0 通道选择 AWD0CH[x] ＝0：ADC 通道 x 不受模拟看门狗 0 监测 AWD0CH[x] ＝1：ADC 通道 x 受模拟看门狗 0 监测 AWD0CH＝0x0，模拟看门狗 0 等同于被关闭

15.5.2.27 ADC 看门狗 1 控制寄存器 (ADC_AWD1CR)

- **Name**：ADC Watchdog1 Control Register
- **Size**：32bits
- **Offset**：0x74
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19-16]	AWDFILT	R/W	0x0	模拟看门狗 1 滤波点数 0x0：模拟看门狗 1 不做滤波 0x1：模拟看门狗 1 监测 2 次超过阈值的转换产生事件 / 中断 …… 0xf：模拟看门狗 1 监测 16 次超过阈值的转换产生事件 / 中断
[15-12]	Reserved	Reserved	Reserved	Reserved
[11-0]	AWD1CH	R/W	0x0	模拟看门狗 1 通道选择 AWD1CH[x] ＝0：ADC 通道 x 不受模拟看门狗 1 监测 AWD1CH[x] ＝1：ADC 通道 x 受模拟看门狗 1 监测 AWD1CH＝0x0，模拟看门狗 1 等同于被关闭

15.5.2.28 ADC 看门狗 2 控制寄存器 (ADC_AWD2CR)

- **Name**：ADC Watchdog2 Control Register
- **Size**：32bits
- **Offset**：0x78
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19-16]	AWDFILT	R/W	0x0	模拟看门狗 2 滤波点数 0x0：模拟看门狗 2 不做滤波 0x1：模拟看门狗 2 监测 2 次超过阈值的转换产生事件 / 中断 …… 0xf：模拟看门狗 2 监测 16 次超过阈值的转换产生事件 / 中断
[15-12]	Reserved	Reserved	Reserved	Reserved
[11-0]	AWD2CH	R/W	0x0	模拟看门狗 2 通道选择 AWD2CH[x] ＝0：ADC 通道 x 不受模拟看门狗 2 保护 AWD2CH[x] ＝1：ADC 通道 x 受模拟看门狗 2 保护 AWD2CH＝0x0，模拟看门狗 2 等同于被关闭

15.5.2.29 ADC 单端偏置补偿寄存器 0 (ADC_OFR0)

- **Name**：ADC Single-End Offset Register0
- **Size**：32bits
- **Offset**：0x80
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFFSET	R/W	0x0	单端模式下第 0 组 ADC 偏置补偿系数，ADC 转换后的数据将会减去该系数，完成去偏置操作。

15.5.2.30 ADC 单端偏置补偿寄存器 1 (ADC_OFR1)

- **Name**：ADC Single-End Offset Register1
- **Size**：32bits
- **Offset**：0x84
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFFSET	R/W	0x0	单端模式下第 1 组 ADC 偏置补偿系数，ADC 转换后的数据将会减去该系数，完成去偏置操作。

15.5.2.31 ADC 单端偏置补偿寄存器 2 (ADC_OFR2)

- **Name**：ADC Single-End Offset Register2
- **Size**：32bits
- **Offset**：0x88
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFFSET	R/W	0x0	单端模式下第 2 组 ADC 偏置补偿系数，ADC 转换后的数据将会减去该系数，完成去偏置操作。

15.5.2.32 ADC 单端偏置补偿寄存器 3 (ADC_OFR3)

- **Name**：ADC Single-End Offset Register3
- **Size**：32bits
- **Offset**：0x8C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFFSET	R/W	0x0	单端模式下第 3 组 ADC 偏置补偿系数，ADC 转换后的数据将会减去该系数，完成去偏置操作。

15.5.2.33 ADC 差分偏置补偿寄存器 0 (ADC_DOFR0)

- **Name**：ADC Differential Offset Register0
- **Size**：32bits
- **Offset**：0x90
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFFSET	R/W	0x0	差分模式下第 0 组 ADC 偏置补偿系数，ADC 转换后的数据将会减去该系数，完成去偏置操作。

15.5.2.34 ADC 差分偏置补偿寄存器 1 (ADC_DOFR1)

- **Name**：ADC Differential Offset Register1
- **Size**：32bits
- **Offset**：0x94
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFFSET	R/W	0x0	差分模式下第 1 组 ADC 偏置补偿系数，ADC 转换后的数据将会减去该系数，完成去偏置操作。

15.5.2.35 ADC 差分偏置补偿寄存器 2 (ADC_DOFR2)

- **Name**：ADC Differential Offset Register2
- **Size**：32bits
- **Offset**：0x98
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFFSET	R/W	0x0	差分模式下第 2 组 ADC 偏置补偿系数，ADC 转换后的数据将会减去该系数，完成去偏置操作。

15.5.2.36 ADC 差分偏置补偿寄存器 3 (ADC_DOFR3)

- **Name**：ADC Differential Offset Register3
- **Size**：32bits
- **Offset**：0x9C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	OFFSET	R/W	0x0	差分模式下第 3 组 ADC 偏置补偿系数，ADC 转换后的数据将会减去该系数，完成去偏置操作。

15.5.2.37 ADC 单端增益系数寄存器 0 (ADC_GCR0)

- **Name**：ADC Single-End Gain Coeff Register0
- **Size**：32bits
- **Offset**：0xA0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	COEFF	R/W	0x0	单端模式下第 0 组 ADC 增益补偿系数，ADC 去偏置后的数据乘上该系数，结果右移 13 位。

15.5.2.38 ADC 单端增益系数寄存器 1 (ADC_GCR1)

- **Name**：ADC Single-End Gain Coeff Register1
- **Size**：32bits
- **Offset**：0xA4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	COEFF	R/W	0x0	单端模式下第 1 组 ADC 增益补偿系数，ADC 去偏置后的数据乘上该系数，结果右移 13 位。

15.5.2.39 ADC 单端增益系数寄存器 2 (ADC_GCR2)

- **Name**：ADC Single-End Gain Coeff Register2
- **Size**：32bits
- **Offset**：0xA8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	COEFF	R/W	0x0	单端模式下第 2 组 ADC 增益补偿系数，ADC 去偏置后的数据乘上该系数，结果右移 13 位。

15.5.2.40 ADC 单端增益系数寄存器 3 (ADC_GCR3)

- **Name**：ADC Single-End Gain Coeff Register3
- **Size**：32bits
- **Offset**：0xAC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	COEFF	R/W	0x0	单端模式下第 3 组 ADC 增益补偿系数，ADC 去偏置后的数据乘上该系数，结果右移 13 位。

15.5.2.41 ADC 差分增益系数寄存器 0 (ADC_DGCR0)

- **Name**：ADC Differential Gain Coeff Register0
- **Size**：32bits
- **Offset**：0xB0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	COEFF	R/W	0x0	差分模式下第 0 组 ADC 增益补偿系数，ADC 去偏置后的数据乘上该系数，结果右移 13 位。

15.5.2.42 ADC 差分增益系数寄存器 1 (ADC_DGCR1)

- **Name**：ADC Differential Gain Coeff Register1
- **Size**：32bits
- **Offset**：0xB4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	COEFF	R/W	0x0	差分模式下第 1 组 ADC 增益补偿系数，ADC 去偏置后的数据乘上该系数，结果右移 13 位。

15.5.2.43 ADC 差分增益系数寄存器 2 (ADC_DGCR2)

- **Name**：ADC Differential Gain Coeff Register2
- **Size**：32bits
- **Offset**：0xB8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	COEFF	R/W	0x0	差分模式下第 2 组 ADC 增益补偿系数，ADC 去偏置后的数据乘上该系数，结果右移 13 位。

15.5.2.44 ADC 差分增益系数寄存器 3 (ADC_DGCR3)

- **Name**：ADC Differential Gain Coeff Register3
- **Size**：32bits
- **Offset**：0xBC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	COEFF	R/W	0x0	差分模式下第 3 组 ADC 增益补偿系数，ADC 去偏置后的数据乘上该系数，结果右移 13 位。

15.5.2.45 ADC 事件控制寄存器 0 (ADC_ECR0)

- **Name**：ADC Event Control Register0
- **Size**：32bits
- **Offset**：0xC0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19-16]	ADSRC	R/W	0x0	Addr-Data Flag 来源选择位。 0000：Channel 0 0001：Channel 1 …… 1011：Channel 11 1100-1111：Reserved
[15-14]	PSRCU	R/W	0x0	Ping-Pong Flag 上升过零选择位。 00：Analog Watchdog 0 01：Analog Watchdog 1 10：Analog Watchdog 2 11：Reserved
[13-12]	PSRCD	R/W	0x0	Ping-Pong Flag 下降过零选择位。 00：Analog Watchdog 0 01：Analog Watchdog 1 10：Analog Watchdog 2 11：Reserved
[11-8]	AW2SEL	R/W	0x0	Watchdog 2 来源选择位。 0000：Watchdog 2 Channel 0 0001：Watchdog 2 Channel 1 …… 1011：Watchdog 2 Channel 12 1100-1111：Reserved
[7-4]	AW1SEL	R/W	0x0	Watchdog 1 来源选择位。 0000：Watchdog 1 Channel 0 0001：Watchdog 1 Channel 1 …… 1011：Watchdog 1 Channel 12 1100-1111：Reserved
[3-0]	AW0SEL	R/W	0x0	Watchdog 0 来源选择位。 0000：Watchdog 0 Channel 0 0001：Watchdog 0 Channel 1 …… 1011：Watchdog 0 Channel 12 1100-1111：Reserved

15.5.2.46 ADC 事件控制寄存器 1 (ADC_ECR1)

- **Name**：ADC Event Control Register1
- **Size**：32bits
- **Offset**：0xC4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19-16]	ADSRC	R/W	0x0	Addr-Data Flag 来源选择位。 0000：Channel 0 0001：Channel 1 …… 1010：Channel 11 1100-1111：Reserved
[15-14]	PSRCU	R/W	0x0	Ping-Pong Flag 上升过零选择位。 00：Analog Watchdog 0 01：Analog Watchdog 1 10：Analog Watchdog 2 11：Reserved
[13-12]	PSRCD	R/W	0x0	Ping-Pong Flag 下降过零选择位。 00：Analog Watchdog 0 01：Analog Watchdog 1 10：Analog Watchdog 2 11：Reserved
[11-8]	AW2SEL	R/W	0x0	Watchdog 2 来源选择位。 0000：Watchdog 2 Channel 0 0001：Watchdog 2 Channel 1 …… 1011：Watchdog 2 Channel 12 1100-1111：Reserved
[7-4]	AW1SEL	R/W	0x0	Watchdog 1 来源选择位。 0000：Watchdog 1 Channel 0 0001：Watchdog 1 Channel 1 …… 1011：Watchdog 1 Channel 12 1100-1111：Reserved
[3-0]	AW0SEL	R/W	0x0	Watchdog 0 来源选择位。 0000：Watchdog 0 Channel 0 0001：Watchdog 0 Channel 1 …… 1011：Watchdog 0 Channel 12 1100-1111：Reserved

15.5.2.47 ADC 事件控制寄存器 2 (ADC_ECR2)

- **Name**：ADC Event Control Register2
- **Size**：32bits
- **Offset**：0xC8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19-16]	ADSRC	R/W	0x0	Addr-Data Flag 来源选择位。 0000：Channel 0 0001：Channel 1 …… 1010：Channel 11 1100-1111：Reserved
[15-14]	PSRCU	R/W	0x0	Ping-Pong Flag 上升过零选择位。 00：Analog Watchdog 0 01：Analog Watchdog 1 10：Analog Watchdog 2 11：Reserved
[13-12]	PSRCD	R/W	0x0	Ping-Pong Flag 下降过零选择位。 00：Analog Watchdog 0 01：Analog Watchdog 1 10：Analog Watchdog 2 11：Reserved
[11-8]	AW2SEL	R/W	0x0	Watchdog 2 来源选择位。 0000：Watchdog 2 Channel 0 0001：Watchdog 2 Channel 1 …… 1011：Watchdog 2 Channel 12 1100-1111：Reserved
[7-4]	AW1SEL	R/W	0x0	Watchdog 1 来源选择位。 0000：Watchdog 1 Channel 0 0001：Watchdog 1 Channel 1 …… 1011：Watchdog 1 Channel 12 1100-1111：Reserved
[3-0]	AW0SEL	R/W	0x0	Watchdog 0 来源选择位。 0000：Watchdog 0 Channel 0 0001：Watchdog 0 Channel 1 …… 1011：Watchdog 0 Channel 12 1100-1111：Reserved

15.5.2.48 ADC 事件控制寄存器 3 (ADC_ECR3)

- **Name**：ADC Event Control Register3
- **Size**：32bits
- **Offset**：0xCC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	Reserved	Reserved	Reserved
[19-16]	ADSRC	R/W	0x0	Addr-Data Flag 来源选择位。 0000：Channel 0 0001：Channel 1 …… 1010：Channel 11 1100-1111：Reserved
[15-14]	PSRCU	R/W	0x0	Ping-Pong Flag 上升过零选择位。 00：Analog Watchdog 0 01：Analog Watchdog 1 10：Analog Watchdog 2 11：Reserved
[13-12]	PSRCD	R/W	0x0	Ping-Pong Flag 下降过零选择位。 00：Analog Watchdog 0 01：Analog Watchdog 1 10：Analog Watchdog 2 11：Reserved
[11-8]	AW2SEL	R/W	0x0	Watchdog 2 来源选择位。 0000：Watchdog 2 Channel 0 0001：Watchdog 2 Channel 1 …… 1011：Watchdog 2 Channel 12 1100-1111：Reserved
[7-4]	AW1SEL	R/W	0x0	Watchdog 1 来源选择位。 0000：Watchdog 1 Channel 0 0001：Watchdog 1 Channel 1 …… 1011：Watchdog 1 Channel 12 1100-1111：Reserved
[3-0]	AW0SEL	R/W	0x0	Watchdog 0 来源选择位。 0000：Watchdog 0 Channel 0 0001：Watchdog 0 Channel 1 …… 1011：Watchdog 0 Channel 12 1100-1111：Reserved

15.5.2.49 ADC 通道数据寄存器 0 (ADC_CDR0)

- **Name**：ADC Channel Data Register0
- **Size**：32bits
- **Offset**：0xD0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 0 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.50 ADC 通道数据寄存器 1 (ADC_CDR1)

- **Name**：ADC Channel Data Register1
- **Size**：32bit
- **Offset**：0xD4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 1 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.51 ADC 通道数据寄存器 2 (ADC_CDR2)

- **Name**：ADC Channel Data Register2
- **Size**：32bits
- **Offset**：0xD8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 2 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.52 ADC 通道数据寄存器 3 (ADC_CDR3)

- **Name**：ADC Channel Data Register3
- **Size**：32bits
- **Offset**：0xDC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 3 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.53 ADC 通道数据寄存器 4 (ADC_CDR4)

- **Name**：ADC Channel Data Register4
- **Size**：32bits
- **Offset**：0xE0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 4 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.54 ADC 通道数据寄存器 5 (ADC_CDR5)

- **Name**：ADC Channel Data Register5
- **Size**：32bits
- **Offset**：0xE4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 5 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.55 ADC 通道数据寄存器 6 (ADC_CDR6)

- **Name**：ADC Channel Data Register6
- **Size**：32bits
- **Offset**：0xE8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 6 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.56 ADC 通道数据寄存器 7 (ADC_CDR7)

- **Name**：ADC Channel Data Register7
- **Size**：32bits
- **Offset**：0xEC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 7 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.57 ADC 通道数据寄存器 8 (ADC_CDR8)

- **Name**：ADC Channel Data Register8
- **Size**：32bits
- **Offset**：0xF0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 8 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.58 ADC 通道数据寄存器 9 (ADC_CDR9)

- **Name**：ADC Channel Data Register9
- **Size**：32bits
- **Offset**：0xF4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 9 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.59 ADC 通道数据寄存器 10 (ADC_CDR10)

- **Name**：ADC Channel Data Register10
- **Size**：32bits
- **Offset**：0xF8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 10 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.60 ADC 通道数据寄存器 11 (ADC_CDR11)

- **Name**：ADC Channel Data Register11
- **Size**：32bits
- **Offset**：0xFC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RDATA	R	0x0	通道 11 ADC 转换数据结果（若 OVR 位置 1 则该结果不可用）

15.5.2.61 ADC 半完成中断使能寄存器 (ADC_HIER)

- **Name**：ADC Half Interrupt Enable Register
- **Size**：32bits
- **Offset**：0x100
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	HLFIE11	R/W	0x0	通道 11 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[10]	HLFIE10	R/W	0x0	通道 10 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[9]	HLFIE9	R/W	0x0	通道 9 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[8]	HLFIE8	R/W	0x0	通道 8 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[7]	HLFIE7	R/W	0x0	通道 7 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[6]	HLFIE6	R/W	0x0	通道 6 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[5]	HLFIE5	R/W	0x0	通道 5 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[4]	HLFIE4	R/W	0x0	通道 4 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[3]	HLFIE3	R/W	0x0	通道 3 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[2]	HLFIE2	R/W	0x0	通道 2 DMA 传输半完成中断使能 0：中断不使能 1：中断使能
[1]	HLFIE1	R/W	0x0	通道 1 DMA 传输半完成中断使能 0：中断不使能 1：中断使能

[0]	HLFIE0	R/W	0x0	通道 0 DMA 传输半完成中断使能 0：中断不使能 1：中断使能

15.5.2.62 ADC 半完成中断状态寄存器 (ADC_HISR)

- **Name**：ADC Half Interrupt Status Register
- **Size**：32bits
- **Offset**：0x104
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	HLF11	R/W	0x0	通道 11 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[10]	HLF10	R/W	0x0	通道 10 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[9]	HLF9	R/W	0x0	通道 9 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[8]	HLF8	R/W	0x0	通道 8 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[7]	HLF7	R/W	0x0	通道 7 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[6]	HLF6	R/W	0x0	通道 6 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[5]	HLF5	R/W	0x0	通道 5 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[4]	HLF4	R/W	0x0	通道 4 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生

[3]	HLF3	R/W	0x0	通道 3 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[2]	HLF2	R/W	0x0	通道 2 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[1]	HLF1	R/W	0x0	通道 1 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[0]	HLF0	R/W	0x0	通道 0 DMA 传输半完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生

15.5.2.63 ADC 完成中断使能寄存器 (ADC_FIER)

- **Name**：ADC Full Interrupt Enable Register
- **Size**：32bits
- **Offset**：0x108
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	FULIE11	R/W	0x0	通道 11 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[10]	FULIE10	R/W	0x0	通道 10 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[9]	FULIE9	R/W	0x0	通道 9 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[8]	FULIE8	R/W	0x0	通道 8 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[7]	FULIE7	R/W	0x0	通道 7 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[6]	FULIE6	R/W	0x0	通道 6 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[5]	FULIE5	R/W	0x0	通道 5 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[4]	FULIE4	R/W	0x0	通道 4 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[3]	FULIE3	R/W	0x0	通道 3 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[2]	FULIE2	R/W	0x0	通道 2 DMA 传输完成中断使能 0：中断不使能 1：中断使能
[1]	FULIE1	R/W	0x0	通道 1 DMA 传输完成中断使能 0：中断不使能 1：中断使能

[0]	FULIE0	R/W	0x0	通道 0 DMA 传输完成中断使能 0：中断不使能 1：中断使能

15.5.2.64 ADC 完成中断状态寄存器 (ADC_FISR)

- **Name**：ADC Full Interrupt Status Register
- **Size**：32bits
- **Offset**：0x10C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	FUL11	R/W	0x0	通道 11 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[10]	FUL10	R/W	0x0	通道 10 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[9]	FUL9	R/W	0x0	通道 9 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[8]	FUL8	R/W	0x0	通道 8 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[7]	FUL7	R/W	0x0	通道 7 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[6]	FUL6	R/W	0x0	通道 6 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[5]	FUL5	R/W	0x0	通道 5 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[4]	FUL4	R/W	0x0	通道 4 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生

[3]	FUL3	R/W	0x0	通道 3 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[2]	FUL2	R/W	0x0	通道 2 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[1]	FUL1	R/W	0x0	通道 1 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生
[0]	FUL0	R/W	0x0	通道 0 DMA 传输完成中断标志 中断标志写 1 清 0。 0：中断标志未产生 1：中断标志产生

15.5.2.65 ADC 传输控制寄存器 0 (ADC_TCR0)

- **Name**：ADC Transfer Control Register0
- **Size**：32bits
- **Offset**：0x110
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.66 ADC 传输地址寄存器 0 (ADC_TAR0)

- **Name**：ADC Transfer Address Register0
- **Size**：32bits
- **Offset**：0x114
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 0 的 DMA 传输的首地址。

15.5.2.67 ADC 传输长度寄存器 0 (ADC_TLR0)

- **Name**：ADC Transfer Length Register0
- **Size**：32bits
- **Offset**：0x11C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 0 的 DMA 传输的长度（单位为 byte）。

15.5.2.68 ADC 传输控制寄存器 1 (ADC_TCR1)

- **Name**：ADC Transfer Control Register1
- **Size**：32bits
- **Offset**：0x120
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.69 ADC 传输地址寄存器 1 (ADC_TAR1)

- **Name**：ADC Transfer Address Register1
- **Size**：32bits
- **Offset**：0x124
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 1 的 DMA 传输的首地址。

15.5.2.70 ADC 传输长度寄存器 1 (ADC_TLR1)

- **Name**：ADC Transfer Length Register1
- **Size**：32bits
- **Offset**：0x12C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 1 的 DMA 传输的长度（单位为 byte）。

15.5.2.71 ADC 传输控制寄存器 2 (ADC_TCR2)

- **Name**：ADC Transfer Control Register2
- **Size**：32bits
- **Offset**：0x130
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.72 ADC 传输地址寄存器 2 (ADC_TAR2)

- **Name**：ADC Transfer Address Register2
- **Size**：32bits
- **Offset**：0x134
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 2 的 DMA 传输的首地址。

15.5.2.73 ADC 传输长度寄存器 2 (ADC_TLR2)

- **Name**：ADC Transfer Length Register2
- **Size**：32bits
- **Offset**：0x13C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 2 的 DMA 传输的长度（单位为 byte）。

15.5.2.74 ADC 传输控制寄存器 3 (ADC_TCR3)

- **Name**：ADC Transfer Control Register3
- **Size**：32bits
- **Offset**：0x140
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.75 ADC 传输地址寄存器 3 (ADC_TAR3)

- **Name**：ADC Transfer Address Register3
- **Size**：32bits
- **Offset**：0x144
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 3 的 DMA 传输的首地址。

15.5.2.76 ADC 传输长度寄存器 3 (ADC_TLR3)

- **Name**：ADC Transfer Length Register3
- **Size**：32bits
- **Offset**：0x14C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 3 的 DMA 传输的长度（单位为 byte）。

15.5.2.77 ADC 传输控制寄存器 4 (ADC_TCR4)

- **Name**：ADC Transfer Control Register4
- **Size**：32bits
- **Offset**：0x150
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.78 ADC 传输地址寄存器 4 (ADC_TAR4)

- **Name**：ADC Transfer Address Register4
- **Size**：32bits
- **Offset**：0x154
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 4 的 DMA 传输的首地址。

15.5.2.79 ADC 传输长度寄存器 4 (ADC_TLR4)

- **Name**：ADC Transfer Length Register4
- **Size**：32bits
- **Offset**：0x15C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 4 的 DMA 传输的长度（单位为 byte）。

15.5.2.80 ADC 传输控制寄存器 5 (ADC_TCR5)

- **Name**：ADC Transfer Control Register5
- **Size**：32bits
- **Offset**：0x160
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.81 ADC 传输地址寄存器 5 (ADC_TAR5)

- **Name**：ADC Transfer Address Register5
- **Size**：32bits
- **Offset**：0x164
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 5 的 DMA 传输的首地址。

15.5.2.82 ADC 传输长度寄存器 5 (ADC_TLR5)

- **Name**：ADC Transfer Length Register5
- **Size**：32bits
- **Offset**：0x16C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 5 的 DMA 传输的长度（单位为 byte）。

15.5.2.83 ADC 传输控制寄存器 6 (ADC_TCR6)

- **Name**：ADC Transfer Control Register6
- **Size**：32bits
- **Offset**：0x170
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.84 ADC 传输地址寄存器 6 (ADC_TAR6)

- **Name**：ADC Transfer Address Register6
- **Size**：32bits
- **Offset**：0x174
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 6 的 DMA 传输的首地址。

15.5.2.85 ADC 传输长度寄存器 6 (ADC_TLR6)

- **Name**：ADC Transfer Length Register6
- **Size**：32bits
- **Offset**：0x17C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 6 的 DMA 传输的长度（单位为 byte）。

15.5.2.86 ADC 传输控制寄存器 7 (ADC_TCR7)

- **Name**：ADC Transfer Control Register7
- **Size**：32bits
- **Offset**：0x180
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.87 ADC 传输地址寄存器 7 (ADC_TAR7)

- **Name**：ADC Transfer Address Register7
- **Size**：32bits
- **Offset**：0x184
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 7 的 DMA 传输的首地址。

15.5.2.88 ADC 传输长度寄存器 7 (ADC_TLR7)

- **Name**：ADC Transfer Length Register7
- **Size**：32bits
- **Offset**：0x18C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 7 的 DMA 传输的长度（单位为 byte）。

15.5.2.89 ADC 传输控制寄存器 8 (ADC_TCR8)

- **Name**：ADC Transfer Control Register8
- **Size**：32bits
- **Offset**：0x190
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.90 ADC 传输地址寄存器 8 (ADC_TAR8)

- **Name**：ADC Transfer Address Register8
- **Size**：32bits
- **Offset**：0x194
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 8 的 DMA 传输的首地址。

15.5.2.91 ADC 传输长度寄存器 8 (ADC_TLR8)

- **Name**：ADC Transfer Length Register8
- **Size**：32bits
- **Offset**：0x19C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 8 的 DMA 传输的长度（单位为 byte）。

15.5.2.92 ADC 传输控制寄存器 9 (ADC_TCR9)

- **Name**：ADC Transfer Control Register9
- **Size**：32bits
- **Offset**：0x1A0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.93 ADC 传输地址寄存器 9 (ADC_TAR9)

- **Name**：ADC Transfer Address Register9
- **Size**：32bits
- **Offset**：0x1A4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 9 的 DMA 传输的首地址。

15.5.2.94 ADC 传输长度寄存器 9 (ADC_TLR9)

- **Name**：ADC Transfer Length Register9
- **Size**：32bits
- **Offset**：0x1AC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 9 的 DMA 传输的长度（单位为 byte）。

15.5.2.95 ADC 传输控制寄存器 10 (ADC_TCR10)

- **Name**：ADC Transfer Control Register10
- **Size**：32bits
- **Offset**：0x1B0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.96 ADC 传输地址寄存器 10 (ADC_TAR10)

- **Name**：ADC Transfer Address Register10
- **Size**：32bits
- **Offset**：0x1B4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 10 的 DMA 传输的首地址。

15.5.2.97 ADC 传输长度寄存器 10 (ADC_TLR10)

- **Name**：ADC Transfer Length Register10
- **Size**：32bits
- **Offset**：0x1BC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 10 的 DMA 传输的长度（单位为 byte）。

15.5.2.98 ADC 传输控制寄存器 11 (ADC_TCR11)

- **Name**：ADC Transfer Control Register11
- **Size**：32bits
- **Offset**：0x1C0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	CIRC	R/W	0x0	DMA 循环模式选择位。 0：DMA 模式不开启 1：DMA 模式循环模式开启（地址：ADC_TARx）
[1]	STP	R/WAC	0x0	DMA 模式停止位。 0：DMA 模式不停止 1：DMA 模式写 1 停止，传输结束后自动清零
[0]	START	R/WAC	0x0	DMA 模式启动位。 0：DMA 模式不启动 1：DMA 模式写 1 启动，传输结束后自动清零

15.5.2.99 ADC 传输地址寄存器 11 (ADC_TAR11)

- **Name**：ADC Transfer Address Register11
- **Size**：32bits
- **Offset**：0x1C4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	ADDR	R/W	0x0	在此位配置通道 11 的 DMA 传输的首地址。

15.5.2.100 ADC 传输长度寄存器 11 (ADC_TLR11)

- **Name**：ADC Transfer Length Register11
- **Size**：32bits
- **Offset**：0x1CC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12-0]	LENG	R/W	0x0	在此位配置通道 11 的 DMA 传输的长度（单位为 byte）。

16 数模转换器（DAC）

16.1 简介

DAC 模块是 12 位电压输出数模转换器。DAC 数据为 12 位模式，在 12 位模式下，数据可以左对齐或右对齐。DAC 支持多达四个输出通道，每个输出通道都有自己的转换器。DAC 每个输出通道的转换独立进行，支持独立的三角波与锯齿波发生器。

16.2 结构框图

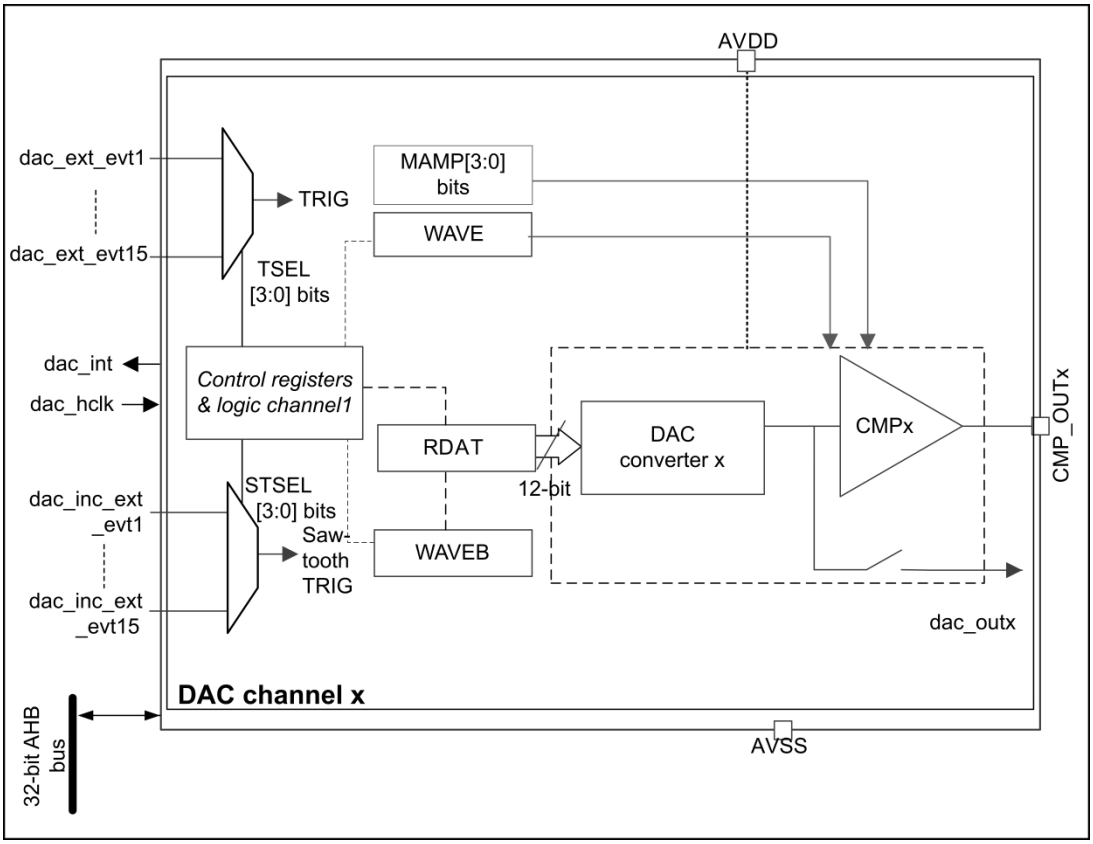


图 16-1 DAC 结构框图

16.3 主要特性

DAC 的主要功能如下：

- 四个 DAC 接口，每个接口对应一个输出通道
- 支持三角波生成模式
- 支持锯齿波生成模式
- 支持软件触发转换
- 支持外部事件触发转换
- 支持连接到片上外设（比较器）

16.4 功能描述

16.4.1 DAC 引脚和内部信号

表 16-1 DAC 输入/输出引脚

Pin name	Signal type	Remarks
AVCC	模拟电源输入	模拟电源
AVSS	模拟电源地输入	模拟地
VCC	模拟电源输入	模拟电源
VDD	数字电源输入	数字电源
VSS	数字地输入	数字地

表 16-2 DAC 输入/输出信号

Internal signal name	Signal type	Description
dac0_ext_evt[15:0]	输入	DAC 通道 0 外部事件输入
dac1_ext_evt[15:0]	输入	DAC 通道 1 外部事件输入
dac2_ext_evt[15:0]	输入	DAC 通道 2 外部事件输入
dac3_ext_evt[15:0]	输入	DAC 通道 3 外部事件输入
dac0_inc_ext_evt[15:0]	输入	DAC 通道 0 增量外部事件输入
dac1_inc_ext_evt[15:0]	输入	DAC 通道 1 增量外部事件输入
dac2_inc_ext_evt[15:0]	输入	DAC 通道 2 增量外部事件输入
dac3_inc_ext_evt[15:0]	输入	DAC 通道 3 增量外部事件输入
dac_int	输出	DAC 中断
dac_hclk	输入	DAC 外设时钟
dac_out0	模拟输出	DAC 通道 0 输出
dac_out1	模拟输出	DAC 通道 1 输出
dac_out2	模拟输出	DAC 通道 2 输出
dac_out3	模拟输出	DAC 通道 3 输出

16.4.2 DAC 通道使能

将 DACx_CR 寄存器中的 PEN 位置 1，即可使能对应 DAC 通道。通道使能后需经过一段时间，完成 DAC 的转换。如果将 DACx_CR 寄存器中的 OEN 位置 1，则 DAC 转换后将输出到相应的 IO 上。

- PEN 位只会使能模拟 DAC 通道。即使将 PEN 位复位，DAC 通道 x 数字接口仍处于使能状态。
- 即便不使能 DAC 输出功能（OEN=0），DAC 内部模拟电路同样会根据配置进行转换，并可以通过内部信号传递到相关的外设（例如 CMP 模块）。

16.4.3 DAC 转换输出模式

DAC 模块支持两种输出模式：

- 直接输出模式
- 波形输出模式

16.4.3.1 DAC 直接输出模式

DAC 配置使能(PEN=1)后，直接输出模式下，将待转换数据写入到 DACx_WDR 寄存器内，在 1 个 HCLK 时钟周期后，将写入的数据自动转移到 DACx_RDR 寄存器内，同时 DAC 模块内部立即开始进行数模转换。经过数模转换时间 $t_{SETTING}$（详细见数据手册）后，如果使能了 DAC 输出功能(OEN=1)并配置了相应的引脚复用，则 DAC 模块将转换后的电平输出到相应的复用引脚上。

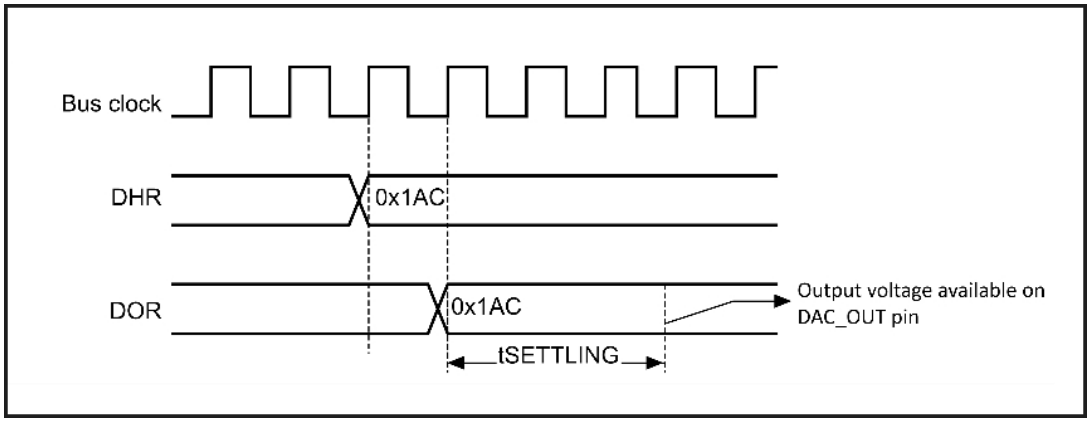


图 16-2 DAC 转换时序图

注意：

- *直接输出模式不支持外部触发，数据写入 DACx_WDR 寄存器后，DAC 内部将立即开始数模转换*
- *直接输出模式不支持中断配置，数模转换完成后不会触发中断事件*
- *DAC 要经过数模转换时间 $t_{SETTING}$ 才完成数模转换，具体时间还取决于电源电压和模拟输出负载*
- *引脚复用请参见“10.4.7 I/O 复用功能”章节*

16.4.3.2 DAC 波形输出模式

DAC 的波形输出模式，目前支持指定两种波形输出：

- 三角波输出(Triangle Wave，简称 TG)
- 锯齿波输出(Sawtooth Wave，简称 ST)

通过配置 DACx_CR 寄存器中的 TGE 或 STE 置 1，使能三角波或锯齿波输出模式。

DAC 的波形输出模式下，波形的输出支持软件(SWT)或外部事件(EXT)两种触发方式，通过配置 DAC 控制寄存器(DACx_CR)中的 TGTRIG[3:0](三角波触发源选择)位域，或 STINCTRIG[3:0]/STRSTTRIG[3:0](锯齿波步进/复位触发源选择)位域进行触发事件的选择。每次软件或外部事件的触发事件都将引发一次 DAC 触发事件，并触发相应的中断(如果使能中断)。

在 DAC 波形输出模式下，每次触发事件(软件或外部事件触发)发生后，经过 3 个 HCLK 后将新的数据自动转移到 DACx_RDR 寄存器，同时 DAC 模块内部根据波形输出的配置，立即对波形补偿后新数据进行数模转换，经过数模转换时间 $t_{SETTING}$（详细见数据手册）完成转换后，如果使能了 DAC 输出功能(OEN=1)并配置了相应的引脚复用，则 DAC 模块将转换后的电平输出到相应的复用引脚上。

注意：

- *三角波和锯齿波更详细的介绍和配置方式请参看“16.4.6 DAC 三角波生成”、“16.4.7DAC 锯齿波生成”。*
- *同一个 DAC 模块内，仅能配置三角波或者锯齿波输出模式，不可同时配置，如果同时配置的 TGE=1 以及 STE=1，则三角波相关的配置无效。*
- *波形输出模式通过触发事件来触发下一次 DAC 转换，更多触发事件描述，请参看“16.4.5 DAC 触发事件选择”。*
- *DAC 要经过数模转换时间 $t_{SETTING}$ 才完成数模转换，具体时间还取决于电源电压和模拟输出负载。*
- *引脚复用请参见”10.4.7 I/O 复用功能”章节。*

16.4.4 DAC 输出电压

经过线性转换后，数字输入会转换为 0 到 V_{AVCC} 之间的输出电压。

各个 DAC 通道引脚的模拟输出电压通过以下公式确定：

$$DACoutput = V_{AVCC} \times \frac{RDARx[11:0]}{4096}$$

16.4.5 DAC 触发事件选择

DAC 的波形输出模式下，触发事件源支持以下两种触发事件：

- SWT 软件触发(Software Trigger)
  通过配置 DAC 软件触发寄存器(DAC_SWTR)中相应位 1 触发相应的触发事件，该位自动清 0。
- EXT 外部事件(External event Trigger)，每个触发源都支持 15 种外部事件选择
  DAC 内部以外部事件的上升沿进行触发新的触发事件。

三角波输出模式时(TGE=1& STE=0)，通过配置 TGTRIG[3:0]选择三角波的触发事件，每次触发事件都将三角波步进输出。触发事件源选择见“表 3/5/7/9. DACx 三角波触发事件&锯齿波复位触发事件选择”。

锯齿波输出模式时(STE=1)，通过配置 STRSTTRIG[3:0]选择锯齿波的复位触发事件，配置 STINCTRIG[3:0]选择锯齿波的步进触发事件，每次触发事件都将触发锯齿波步进或复位输出。触发事件源选择见“表 3/5/7/9. DACx 三角波触发事件&锯齿波复位触发事件选择”和“表 4/6/8/10. DACx 锯齿波步进触发事件选择”。

注意：触发源配置必须在 DAC 使能前(PEN=0)，使能后不可再改变。

表 16-3 DAC0 三角波触发事件&锯齿波复位触发事件选择

Source	Type	TGTRIG0[3:0]/STRSTTRIG0[3:0]
SWTRIG	软件控制位	0000
TMR0_TRGO	来自片上定时器的内部信号	0001
TMR1_TRGO	来自片上定时器的内部信号	0010
TMR2_TRGO	来自片上定时器的内部信号	0011
TMR4_TRGO	来自片上定时器的内部信号	0100
TMR5_TRGO	来自片上定时器的内部信号	0101
TMR6_TRGO	来自片上定时器的内部信号	0110
TMR7_TRGO	来自片上定时器的内部信号	0111
HRPWM_DAC_RESET_TRG0	来自片上 PWM 的内部信号	1000
HRPWM_DAC_RESET_TRG1	来自片上 PWM 的内部信号	1001
HRPWM_DAC_RESET_TRG2	来自片上 PWM 的内部信号	1010
HRPWM_DAC_RESET_TRG3	来自片上 PWM 的内部信号	1011
HRPWM_DAC_RESET_TRG4	来自片上 PWM 的内部信号	1100
HRPWM_DAC_RESET_TRG5	来自片上 PWM 的内部信号	1101
HRPWM_ADC_TRG4	来自片上 PWM 的内部信号	1110
PA10	外部引脚	1111

表 16-4 DAC0 锯齿波步进触发事件选择

Source	Type	STINCTRIG0[3:0]
SWTRIGB	软件控制位	0000
TMR0_TRGO	来自片上定时器的内部信号	0001
TMR1_TRGO	来自片上定时器的内部信号	0010
TMR3_TRGO	来自片上定时器的内部信号	0011
TMR4_TRGO	来自片上定时器的内部信号	0100
TMR5_TRGO	来自片上定时器的内部信号	0101
TMR6_TRGO	来自片上定时器的内部信号	0110
TMR7_TRGO	来自片上定时器的内部信号	0111
HRPWM_DAC_STEP_TRG0	来自片上 PWM 的内部信号	1000
HRPWM_DAC_STEP_TRG1	来自片上 PWM 的内部信号	1001
HRPWM_DAC_STEP_TRG2	来自片上 PWM 的内部信号	1010
HRPWM_DAC_STEP_TRG3	来自片上 PWM 的内部信号	1011
HRPWM_DAC_STEP_TRG4	来自片上 PWM 的内部信号	1100
HRPWM_DAC_STEP_TRG5	来自片上 PWM 的内部信号	1101
HRPWM_ADC_TRG0	来自片上 PWM 的内部信号	1110
PA11	外部引脚	1111

表 16-5 DAC1 通道触发事件&锯齿波复位触发事件选择

Source	Type	TGTRIG1[3:0]/STRSTTRIG1[3:0]
SWTRIG	软件控制位	0000
TMR0_TRGO	来自片上定时器的内部信号	0001
TMR1_TRGO	来自片上定时器的内部信号	0010
TMR2_TRGO	来自片上定时器的内部信号	0011
TMR4_TRGO	来自片上定时器的内部信号	0100
TMR5_TRGO	来自片上定时器的内部信号	0101
TMR6_TRGO	来自片上定时器的内部信号	0110
TMR7_TRGO	来自片上定时器的内部信号	0111
HRPWM_DAC_RESET_TRG0	来自片上 PWM 的内部信号	1000
HRPWM_DAC_RESET_TRG1	来自片上 PWM 的内部信号	1001
HRPWM_DAC_RESET_TRG2	来自片上 PWM 的内部信号	1010
HRPWM_DAC_RESET_TRG3	来自片上 PWM 的内部信号	1011
HRPWM_DAC_RESET_TRG4	来自片上 PWM 的内部信号	1100
HRPWM_DAC_RESET_TRG5	来自片上 PWM 的内部信号	1101
HRPWM_ADC_TRG5	来自片上 PWM 的内部信号	1110
PA10	外部引脚	1111

表 16-6 DAC1 锯齿波步进触发事件选择

Source	Type	STINCTRIG1[3:0]
SWTRIGB	软件控制位	0000
TMR0_TRGO	来自片上定时器的内部信号	0001
TMR1_TRGO	来自片上定时器的内部信号	0010
TMR3_TRGO	来自片上定时器的内部信号	0011
TMR4_TRGO	来自片上定时器的内部信号	0100
TMR5_TRGO	来自片上定时器的内部信号	0101
TMR6_TRGO	来自片上定时器的内部信号	0110
TMR7_TRGO	来自片上定时器的内部信号	0111
HRPWM_DAC_STEP_TRG0	来自片上 PWM 的内部信号	1000
HRPWM_DAC_STEP_TRG1	来自片上 PWM 的内部信号	1001
HRPWM_DAC_STEP_TRG2	来自片上 PWM 的内部信号	1010
HRPWM_DAC_STEP_TRG3	来自片上 PWM 的内部信号	1011
HRPWM_DAC_STEP_TRG4	来自片上 PWM 的内部信号	1100
HRPWM_DAC_STEP_TRG5	来自片上 PWM 的内部信号	1101
HRPWM_ADC_TRG1	来自片上 PWM 的内部信号	1110
PA11	外部引脚	1111

表 16-7 DAC2 通道触发事件&锯齿波复位触发事件选择

Source	Type	TGTRIG2[3:0]/STRSTTRIG2[3:0]
SWTRIG	软件控制位	0000
TMR0_TRGO	来自片上定时器的内部信号	0001
TMR1_TRGO	来自片上定时器的内部信号	0010
TMR2_TRGO	来自片上定时器的内部信号	0011
TMR4_TRGO	来自片上定时器的内部信号	0100
TMR5_TRGO	来自片上定时器的内部信号	0101
TMR6_TRGO	来自片上定时器的内部信号	0110
TMR7_TRGO	来自片上定时器的内部信号	0111
HRPWM_DAC_RESET_TRG0	来自片上 PWM 的内部信号	1000
HRPWM_DAC_RESET_TRG1	来自片上 PWM 的内部信号	1001
HRPWM_DAC_RESET_TRG2	来自片上 PWM 的内部信号	1010
HRPWM_DAC_RESET_TRG3	来自片上 PWM 的内部信号	1011
HRPWM_DAC_RESET_TRG4	来自片上 PWM 的内部信号	1100
HRPWM_DAC_RESET_TRG5	来自片上 PWM 的内部信号	1101
HRPWM_ADC_TRG6	来自片上 PWM 的内部信号	1110
PA10	外部引脚	1111

表 16-8 DAC2 锯齿步进触发事件选择

Source	Type	STINCTRIG2[3:0]
SWTRIGB	软件控制位	0000
TMR0_TRGO	来自片上定时器的内部信号	0001
TMR1_TRGO	来自片上定时器的内部信号	0010
TMR3_TRGO	来自片上定时器的内部信号	0011
TMR4_TRGO	来自片上定时器的内部信号	0100
TMR5_TRGO	来自片上定时器的内部信号	0101
TMR6_TRGO	来自片上定时器的内部信号	0110
TMR7_TRGO	来自片上定时器的内部信号	0111
HRPWM_DAC_STEP_TRG0	来自片上 PWM 的内部信号	1000
HRPWM_DAC_STEP_TRG1	来自片上 PWM 的内部信号	1001
HRPWM_DAC_STEP_TRG2	来自片上 PWM 的内部信号	1010
HRPWM_DAC_STEP_TRG3	来自片上 PWM 的内部信号	1011
HRPWM_DAC_STEP_TRG4	来自片上 PWM 的内部信号	1100
HRPWM_DAC_STEP_TRG5	来自片上 PWM 的内部信号	1101
HRPWM_ADC_TRG2	来自片上 PWM 的内部信号	1110
PA11	外部引脚	1111

表 16-9 DAC3 通道触发事件&锯齿波复位触发事件选择

Source	Type	TGTRIG3[3:0]/STRSTTRIG3[3:0]
SWTRIG	软件控制位	0000
TMR0_TRGO	来自片上定时器的内部信号	0001
TMR1_TRGO	来自片上定时器的内部信号	0010
TMR2_TRGO	来自片上定时器的内部信号	0011
TMR4_TRGO	来自片上定时器的内部信号	0100
TMR5_TRGO	来自片上定时器的内部信号	0101
TMR6_TRGO	来自片上定时器的内部信号	0110
TMR7_TRGO	来自片上定时器的内部信号	0111
HRPWM_DAC_RESET_TRG0	来自片上 PWM 的内部信号	1000
HRPWM_DAC_RESET_TRG1	来自片上 PWM 的内部信号	1001
HRPWM_DAC_RESET_TRG2	来自片上 PWM 的内部信号	1010
HRPWM_DAC_RESET_TRG3	来自片上 PWM 的内部信号	1011
HRPWM_DAC_RESET_TRG4	来自片上 PWM 的内部信号	1100
HRPWM_DAC_RESET_TRG5	来自片上 PWM 的内部信号	1101
HRPWM_ADC_TRG7	来自片上 PWM 的内部信号	1110
PA10	外部引脚	1111

表 16-10 DAC3 锯齿波步进触发事件选择

Source	Type	STINCTRIG3[3:0]
SWTRIGB	软件控制位	0000
TMR0_TRGO	来自片上定时器的内部信号	0001
TMR1_TRGO	来自片上定时器的内部信号	0010
TMR3_TRGO	来自片上定时器的内部信号	0011
TMR4_TRGO	来自片上定时器的内部信号	0100
TMR5_TRGO	来自片上定时器的内部信号	0101
TMR6_TRGO	来自片上定时器的内部信号	0110
TMR7_TRGO	来自片上定时器的内部信号	0111
HRPWM_DAC_STEP_TRG0	来自片上 PWM 的内部信号	1000
HRPWM_DAC_STEP_TRG1	来自片上 PWM 的内部信号	1001
HRPWM_DAC_STEP_TRG2	来自片上 PWM 的内部信号	1010
HRPWM_DAC_STEP_TRG3	来自片上 PWM 的内部信号	1011
HRPWM_DAC_STEP_TRG4	来自片上 PWM 的内部信号	1100
HRPWM_DAC_STEP_TRG5	来自片上 PWM 的内部信号	1101
HRPWM_ADC_TRG3	来自片上 PWM 的内部信号	1110
PA11	外部引脚	1111

16.4.6 DAC 三角波生成

可以用于在直流信号或慢变信号上叠加一个小幅三角波。配置三角波输出可按以下步骤：

1. 配置 DACx_CR 寄存器中的 TGDIR，选择三角波的起始步进方向；
2. 配置 DACx_CR 寄存器中的 TGAMP[3:0]，选择三角波的最大幅度；
3. 配置 DACx_CR 寄存器中的 TGTRIG[3:0]，选择三角波步进的触发源；
4. 配置 DACx_CR 寄存器中的 TGE 位置 1，使能三角波输出功能；
5. 向 DACx_WDR 寄存器写入数据，作为三角波输出的基准电压；
6. 配置 DACx_CR 寄存器中的 DIE 位置 1（如果需要中断）；
7. 配置 DACx_CR 寄存器中的 OEN 位置 1（如果需要输出到 IO）；
8. 最后配置 DACx_CR 寄存器中的 PEN 位置 1，使能 DAC 模块。

配置完成后，DAC 模块每次发生触发事件后，经过三个 HCLK 时钟周期，内部三角波计数器将会递增。在不发生溢出的情况下，该计数器的值将与 DACx_WDR 寄存器内容相加，所得总和将传输到 DACx_RDR 寄存器中。只要小于 TGAMP[3:0]位定义的最大幅度，三角波计数器就会一直递增。一旦达到配置的幅度后，计数开始递减至 0，然后再次递增，以此类推。

注意：

- *要使能三角波输出，必须保证锯齿波输出使能位置 0(STE=0)，否则将三角波的配置无效；*
- *应当在配置完三角波相关的设置后，最后再使能 DAC 输出功能(OEN=1)和 DAC 模块(PEN=1)。*

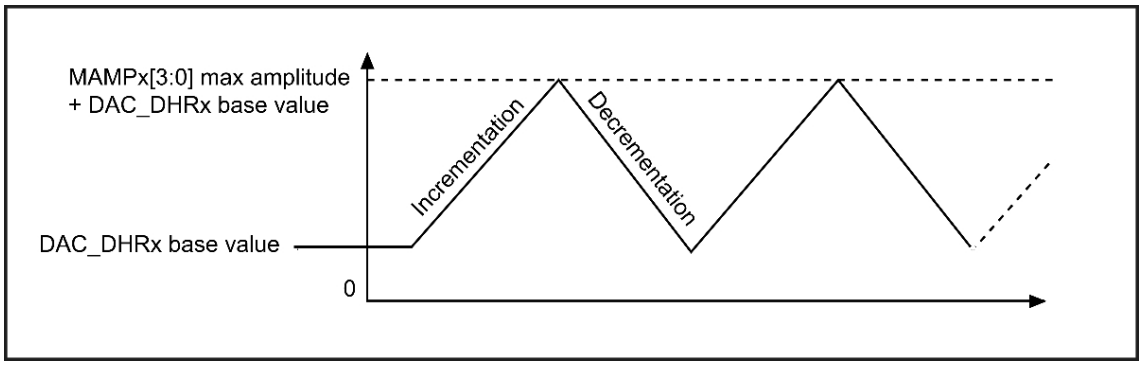


图 16-3 DAC 三角波生成

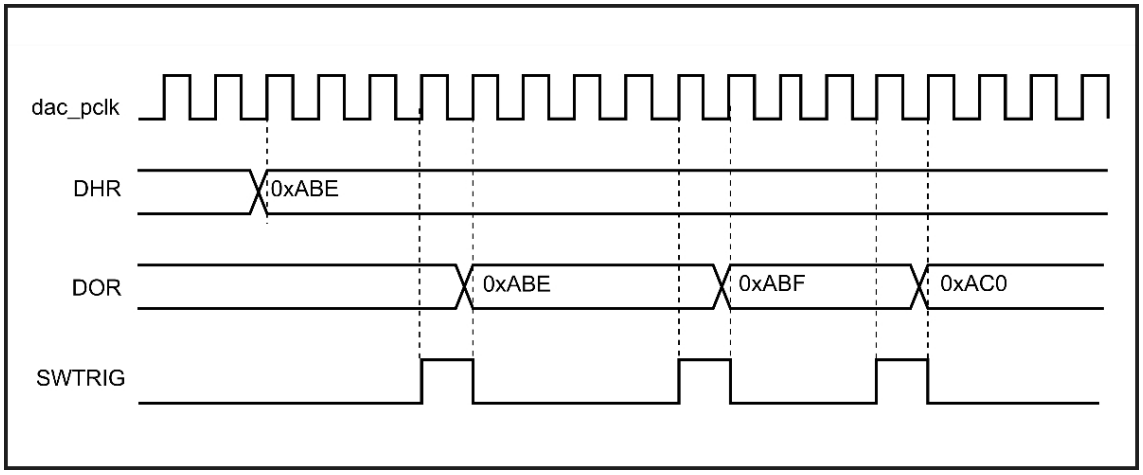


图 16-4 DAC 三角波生成的转换（软件触发模式为例）

16.4.7 DAC 锯齿波生成

DAC 可以生成锯齿波，软件可按以下步骤对初始值、步进值和方向进行特定的寄存器配置：

1. 配置 DACx_CR 寄存器中的 STDIR，选择锯齿波的步进方向；
2. 配置 DACx_CR 寄存器中的 STINCTRIG[3:0]，选择锯齿波的步进触发源；
3. 配置 DACx_CR 寄存器中的 STRSTTRIG[3:0]，选择锯齿波的复位触发源；
4. 配置 DACx_SIDR 寄存器中的 SIDx[15:0]，设置锯齿波的步进步长（12.4 位格式）；
5. 配置 DACx_SRDR 寄存器中的 SRDx[15:4]，设置锯齿波的复位值；
6. 配置 DACx_CR 寄存器中的 DIE 和 DBIE 位置 1（如果需要中断）；
7. 配置 DACx_CR 寄存器中的 OEN 位置 1（如果需要输出到 IO）；
8. 最后配置 DACx_CR 寄存器中的 PEN 位置 1，使能 DAC 模块。

配置完成后，DAC 模块发生复位触发事件后锯齿波计数从 SRDx [15:4]开始，后续每个步进触发事件之后向制定的方向步进 SIDx [15:0]值（12.4 位格式，见后续的格式描述）。

DAC 使用锯齿波计数值的 12 位 MSB（位[15：4]）输出。当计数达到 0x0000 或 0xFFFF 时，计数发生饱和。锯齿波复位触发事件将计数值初始化为 SRDx [15：4]的值。

注意：

- *应当在配置完锯齿波相关的设置后，最后再使能 DAC 输出功能(OEN=1)和 DAC 模块(PEN=1)；*

- *锯齿波的复位触发优先级高于步进优先级，如果复位与步进同时触发，则只处理复位事件，步进触发无效；*
- *如果复位触发和步进触发非同时触发，则 DAC 模块按触发信号到达的顺序进行处理。*

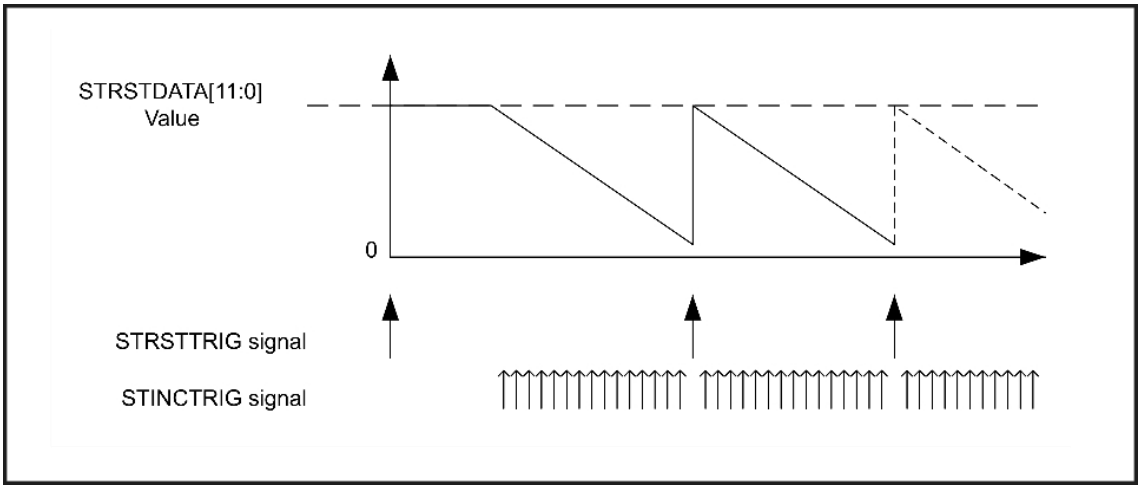


图 16-5 DAC 锯齿波生成示例（STDIR=0）

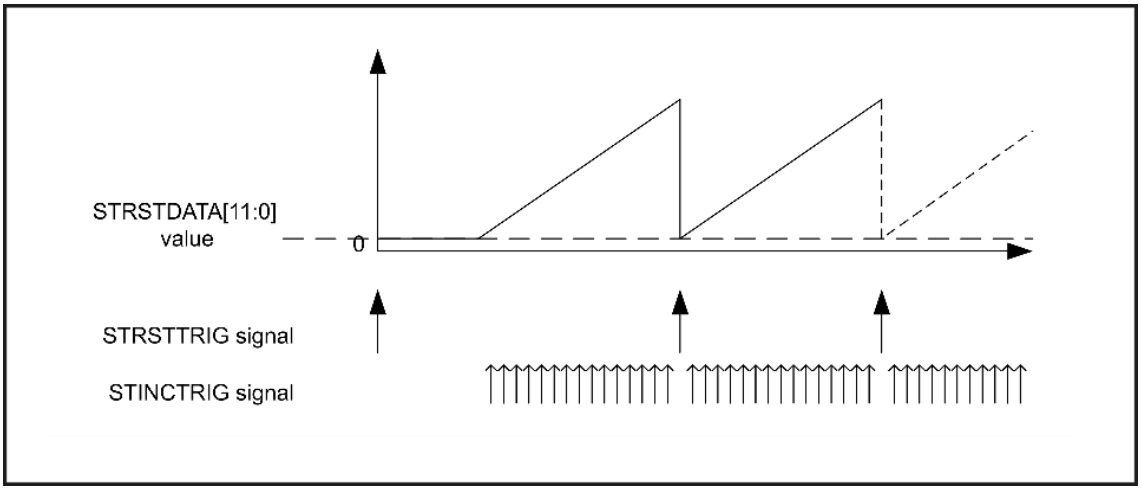


图 16-6 DAC 锯齿波生成示例（STDIR=1）

16.4.8 中断

中断向量	中断事件	中断标志	使能控制位	标志清除方式
dac_int	DACx 三角波触发事件	DxIF	DIEx	DxIF 写 1 清除
	DACx 锯齿波触发事件	DBxIF	DBIEx	DBxIF 写 1 清除

16.5 寄存器描述

16.5.1 寄存器列表

Name	Offset	Width	Description
DACx_CR	0x00/0x04/0x08/0x0C	32bits	DACx 配置寄存器
DAC_SR	0x10	32bits	DAC 中断状态寄存器
DAC_SWTR	0x14	32bits	DAC 软件触发寄存器
DACx_WDR	0x20/0x24/0x28/0x2C	32bits	DACx 写入数据寄存器
DACx_RDAT	0x30/0x34/0x38/0x3C	32bits	DACx 读取数据寄存器
DACx_SIDR	0x40/0x44/0x48/0x4C	32bits	DACx 锯齿波形步进数据寄存器
DACx_SRDR	0x50/0x54/0x58/0x0C	32bits	DACx 锯齿波形复位数据寄存器

16.5.2 寄存器详细描述

16.5.2.1 DACx 配置寄存器 (DACx_CR)

- **Name**：DACx Config Register(x = 0 .. 3)
- **Size**：32bits
- **Offset**：0x00/0x04/0x08/0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-26]	Reserved	Reserved	Reserved	Reserved
[25]	STE	R/W	0x0	锯齿波产生使能位 (Sawtooth Enable) 0：不使能 1：使能 注意：如果三角波产生同时被使能(TGE=1)，则生成锯齿波，三角波配置无效。
[24]	STDIR	R/W	0x0	锯齿波步进方向选择位 (Sawtooth Direction select) 0：向下(幅值减小) 1：向上(幅值增大)
[23-20]	STINCTRIG	R/W	0x0	锯齿波步进触发源选择位 (Sawtooth Increment Trigger select) 更详细说明请参考“16.4.5 DAC 触发事件选择”章节中的说明。
[19-16]	STRSTTRIG	R/W	0x0	锯齿波复位触发源选择位 (Sawtooth Reset Trigger select) 更详细说明请参考“16.4.5 DAC 触发事件选择”章节中的说明。
[15-14]	Reserved	Reserved	Reserved	Reserved

[13]	TGE	R/W	0x0	三角波产生使能位 (Triangle Enable) 0：不使能 1：使能 注意：如果锯齿波产生同时被使能(STE=1)，则生成锯齿波，三角波配置无效。
[12]	TGDIR	R/W	0x0	三角波起始方向选择位 (Triangle Direction select) 0：向下(幅值减小) 1：向上(幅度增大)
[11-8]	TGAMP	R/W	0x0	三角波增长最大幅值选择位 (Triangle Amplitude select) 计算公式：幅值 = $2^{(TGAMP+1)}$ - 1 0000：幅度为 1 0001：幅度为 3 0010：幅度为 7 0011：幅度为 15 ..... 1011：幅度为 4095 其他值：保留
[7-4]	TGTRIG	R/W	0x0	三角波触发源选择 (Triangle Trigger select) 更详细说明请参考“16.4.5 DAC 触发事件选择”章节中的说明。
[3]	DBIE	R/W	0x0	DAC DONEB 中断使能位 (DAC DONEB Interrupt Enable) 0：不使能 1：使能 说明：以下操作将会引发 DONEB 中断： ● 输出锯齿波(STE=1)，触发锯齿波**步进**（软件或事件触发），且 DAC 输出完成后。
[2]	DIE	R/W	0x0	DAC DONE 中断使能位 (DAC DONE Interrupt Enable) 0：不使能 1：使能 说明：以下操作将引发 DONE 中断： ● 输出锯齿波(STE=1)，触发锯齿波**复位**（软件或事件触发），且 DAC 输出完成后； ● 输出三角波(TGE=1)，触发三角波**步进**(软件或事件触发)，且 DAC 输出完成以后。
[1]	OEN	R/W	0x0	DAC 转换输出使能位 (DAC Output Enable) 0：不使能 1：使能 注意：如果不使能转换输出(OEN=0)，则 DAC 的转换不会被输出到相应的 IO 上，但并不影响 DAC 作为内部信号输出到其他外设(例如 CMP)。
[0]	PEN	R/W	0x0	DAC 外设使能位 (DAC Peripheral Enable) 0：不使能 1：使能

16.5.2.2 DAC 中断状态寄存器 (DAC_ISR)

- **Name**：DAC Interrupt Status Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7]	DB3IF	R/W1C	0x0	DAC3 DONEB 中断标志位 (DAC3 DONEB Interrupt Flag) 描述同 DB0IF。
[6]	DB2IF	R/W1C	0x0	DAC2 DONEB 中断标志位 (DAC2 DONEB Interrupt Flag) 描述同 DB0IF。
[5]	DB1IF	R/W1C	0x0	DAC1 DONEB 中断标志位 (DAC1 DONEB Interrupt Flag) 描述同 DB0IF。
[4]	DB0IF	R/W1C	0x0	DAC0 DONEB 中断标志位 (DAC0 DONEB Interrupt Flag) 0：未发生 DONEB 中断 1：发生 DONEB 中断 说明：以下操作将会引发 DBxIF 中断标志位置位： ● 输出锯齿波(STE=1)，触发锯齿波**复位**（软件或事件触发），且 DAC 输出完成后。
[3]	D3IF	R/W1C	0x0	DAC3 DONE 中断标志位 (DAC3 DONE Interrupt Flag) 描述同 D0IF。
[2]	D2IF	R/W1C	0x0	DAC2 DONE 中断标志位 (DAC2 DONE Interrupt Flag) 描述同 D0IF。
[1]	D1IF	R/W1C	0x0	DAC1 DONE 中断标志位 (DAC1 DONE Interrupt Flag) 描述同 D0IF。
[0]	D0IF	R/W1C	0x0	DAC0 DONE 中断标志位 (DAC0 DONE Interrupt Flag) 0：未发生 DONE 中断 1：发生 DONE 中断 说明：以下操作将引发 DxIF 中断标志位置位： ● 输出锯齿波(STE=1)，触发锯齿波**步进**（软件或事件触发），且 DAC 输出完成后； ● 输出三角波(TGE=1)，触发三角波**步进**(软件或事件触发)，且 DAC 输出完成以后。

16.5.2.3 DAC 软件触发寄存器 (DAC_SWTR)

- **Name**：DAC Software Trigger Register
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7]	SWTB3	R/WAC	0x0	DAC3 软件触发 SWTRIGB 事件 (DAC3 Software Trigger B) 描述同 SWTB0。
[6]	SWTB2	R/WAC	0x0	DAC2 软件触发 SWTRIGB 事件 (DAC2 Software Trigger B) 描述同 SWTB0。
[5]	SWTB1	R/WAC	0x0	DAC1 软件触发 SWTRIGB 事件 (DAC1 Software Trigger B) 描述同 SWTB0。
[4]	SWTB0	R/WAC	0x0	DAC0 软件触发 SWTRIGB 事件 (DAC0 Software Trigger B) 0：不触发 1：触发 说明：触发 SWTRIGB 信号用途： ● 输出锯齿波(STE=1)，并且锯齿波复位选择 SWTRIGB 作为触发事件(STRSTTRIG[3:0]选择 SWTRIGB)，将会触发锯齿波**复位** 更多触发事件说明请参考“16.4.5 DAC 触发事件选择”章节。
[3]	SWT3	R/WAC	0x0	DAC3 软件触发 SWTRIG 事件 (DAC3 Software Trigger) 描述同 SWT0。
[2]	SWT2	R/WAC	0x0	DAC2 软件触发 SWTRIG 事件 (DAC2 Software Trigger) 描述同 SWT0。
[1]	SWT1	R/WAC	0x0	DAC1 软件触发 SWTRIG 事件 (DAC1 Software Trigger) 描述同 SWT0。
[0]	SWT0	R/WAC	0x0	DAC0 软件触发 SWTRIG 事件 (DAC0 Software Trigger) 0：不触发 1：触发 说明：触发 SWTRIG 信号用途： ● 输出锯齿波(STE=1)，并且锯齿波步进选择 SWTRIG 作为触发事件(STINCTRIG[3:0]选择 SWTRIG)，将会触发锯齿波**步进**； ● 输出三角波(TGE=1)，并且三角波步进选择 SWTRIG 作为触发事件（TGTRIG[3:0]选择 SWTRIG），将会触发三角波**步进**。 更多触发事件说明请参考“16.4.5 DAC 触发事件选择”章节

16.5.2.4 DACx 写入数据寄存器 (DACx_WDR)

- **Name**：DACx Write Data Register(x = 0 ... 3)
- **Size**：32bits
- **Offset**：0x20/0x24/0x28/0x2C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11-0]	WDATx	R/W	0x0	DACx 写入输出数据 (Write Data for DACx) 注意： ● 输出锯齿波(STE=1)波形时，写入该寄存器**无效**； ● 输出三角波(TGE=1)波形时，写入该寄存器将作为三角波的起始数据，三角波输出以此数据对应电平作为基准，触发进行幅值增长或减少； ● 非波形输出模式下，写入该寄存器将立即进行 DAC 转换。 更多信息参看“16.4.3 DAC 转换输出模式”、“16.4.6 DAC 三角波生成”、“16.4.7 DAC 锯齿波生成”章节。

16.5.2.5 DACx 读取数据寄存器 (DACx_RDR)

- **Name**：DACx Read Data Register(x = 0 ... 3)
- **Size**：32bits
- **Offset**：0x30/0x34/0x38/0x3C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11-0]	RDATx	R	0x0	DAC 读取输出数据(斜坡补偿后) (Read Data from DACx) 说明：输出波形模式下(三角波/锯齿波)，读取该寄存器的值位经过波形补偿后的数值。

16.5.2.6 DACx 锯齿波形步进数据寄存器 (DACx_SIDR)

- **Name**：DACx Sawtooth Increment Data Register(x = 0 ... 3)
- **Size**：32bits
- **Offset**：0x40/0x44/0x48/0x4C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	SIDx	R/W	0x0	DACx 锯齿波形步进数据 (DACx Sawtooth Increment Data) 12.4 位格式（12 位整数部分，4 位小数部分） 更多详细信息参考“16.4.7 DAC 锯齿波生成”章节。

16.5.2.7 DACx 锯齿波形复位数据寄存器 (DACx_SRDR)

- **Name**：DACx Sawtooth Reset Data Register(x = 0 ... 3)
- **Size**：32bits
- **Offset**：0x50/0x54/0x58/0x5C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-4]	SRDx	R/W	0x0	DACx 锯齿波形复位数据(初始数据) (DACx Sawtooth Reset Data) 锯齿波形复位初始数据（整数部分）
[3-0]	Reserved	Reserved	Reserved	Reserved

17 比较器（CMP）

17.1 简介

芯片内置四路模拟比较器通道(CMP0/1/2/3)，可用于各种功能，包括：

- 模拟信号调理
- 与定时器的 PWM 输出结合使用时，构成逐周期电流控制环路

17.2 结构框图

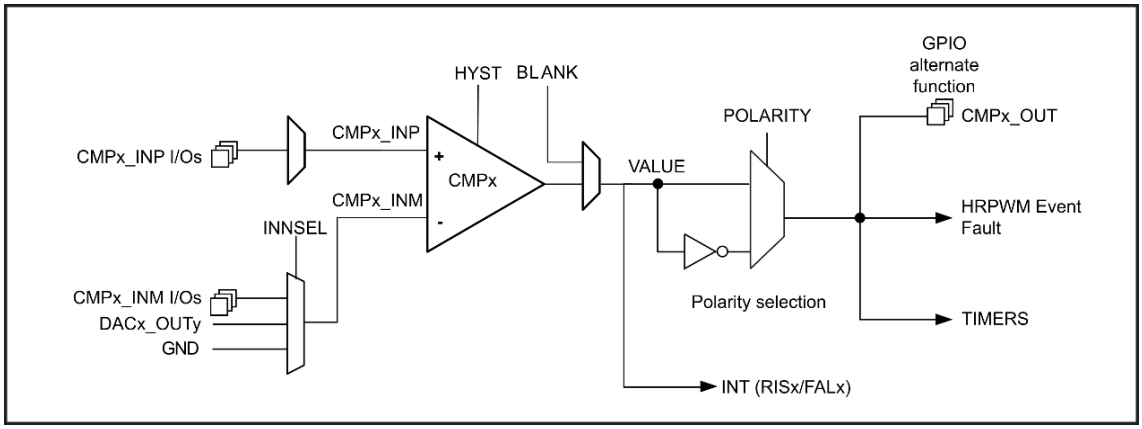


图 17-1 CMP 结构框图

17.3 主要特性

- 可选的负端模拟输入
  - I/O 引脚输入(每个比较通道均有两个引脚可选)
  - DAC 通道输出(内部信号)
  - 内部接至 GND
- 可编程迟滞
- 将输出映射到 I/O，并可配消抖后输出
- 将输出重定向到用于触发以下事件的定时器输入
  - 捕获事件
  - 断路事件（用于快速 PWM 关断）
- 消隐比较器输出
- 上升沿和下降沿中断

17.4 功能描述

17.4.1 引脚和内部信号

表 17-1 CMPx 正端输入引脚

-	CMP0_INP	CMP1_INP	CMP2_INP	CMP3_INP
-	PA7	PB0	PB11	PC4

表 17-2 CMPx 负端输入引脚

INMSEL[1:0]	CMP0_INM	CMP1_INM	CMP2_INM	CMP3_INM
00	GND	GND	GND	GND
01	PA2	PB2	PB15	PB14
10	PA4	PA4	PA4	PA4
11	DAC0_OUT	DAC1_OUT	DAC2_OUT	DAC3_OUT

表 17-3 CMPx 数字输出引脚

-	CMP0_OUT	CMP1_OUT	CMP2_OUT	CMP3_OUT
-	PA2、PA12、PB9	PB1、PB7	PA10、PA13、PD0	PA0、PA5、PC4、PB5

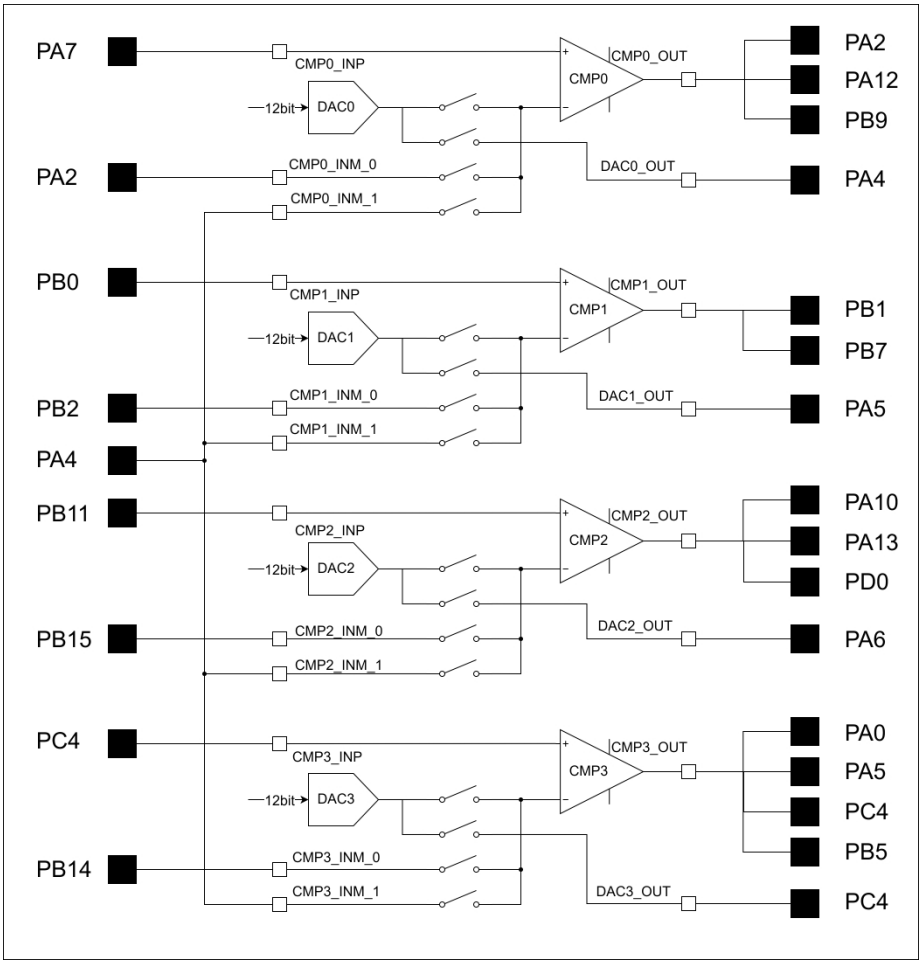


图 17-2 CMPx 与 GPIO 映射关系

17.4.2 迟滞

比较器具有可编程迟滞，可在有信号噪声时避免发生意外输出转换。比较器迟滞是非对称的，仅作用于比较器输出的下降沿。迟滞可在不需要时禁止，以便使用外部组件强制迟滞功能。

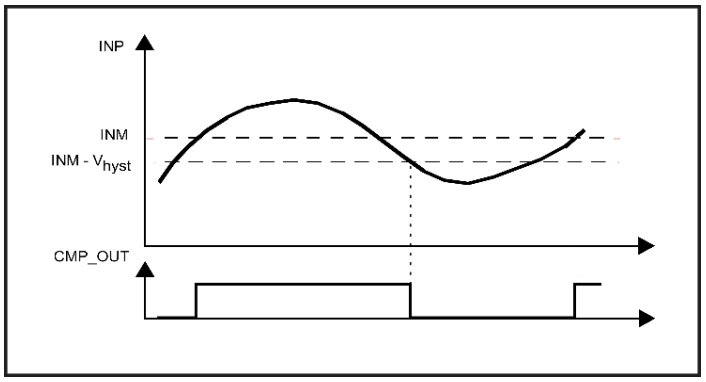


图 17-3 比较器迟滞

17.4.3 输出消隐

消隐功能的目的是防止电流调节在 PWM 周期开始处出现短暂电流尖峰（通常是功率开关反向并联二极管中的恢复电流）时发生跳闸。该功能使用通过定时器输出比较信号定义的消隐窗口。比较器消隐源通过软件配置相应的 CMPx_CR 寄存器中的 CxBLANK[2:0]位来选择，如表 17-4 所示。消隐信号对内部比较器输出进行门控，以便使 cmp_out 免受因电流尖峰而导致的寄生脉冲的干扰，如图 17-4 所示。

表 17-4 CMP 输入消隐源

BLANK [2:0]	CMP0	CMP1	CMP2	CMP3
000	无消隐	无消隐	无消隐	无消隐
001	TMR0_PWM	TMR1_ PWM	TMR0_ PWM	TMR1_ PWM
010	TMR1_ PWM	TMR2_ PWM	TMR1_ PWM	TMR2_ PWM
011	TMR2_ PWM	TMR3_ PWM	TMR2_ PWM	TMR3_ PWM
100	TMR3_ PWM	TMR4_ PWM	TMR3_ PWM	TMR4_ PWM
101	TMR4_ PWM	TMR5_ PWM	TMR4_ PWM	TMR5_ PWM
110	TMR5_ PWM	TMR6_ PWM	TMR5_ PWM	TMR6_ PWM
111	TMR6_ PWM	TMR7_ PWM	TMR6_ PWM	TMR7_ PWM

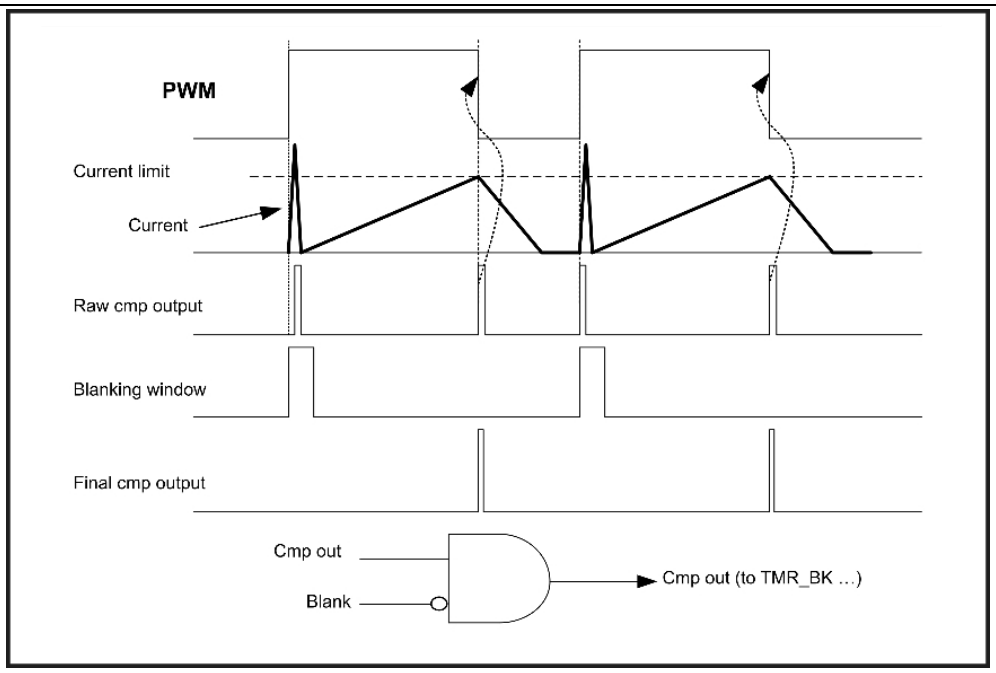


图 17-4 比较器输出消隐

注意：TMRx_PWM 指的是 TMRx 模块输出 PWM 内部信号，需在相应的 TMRx 模块内进行相应的配置，详请参考 TMR 模块中的“定时器对外触发和 PWM 内部信号输出”。

17.4.4 输出重定向

任一 CMP 通道的输出均可重定向到 HRPWM 的事件输入（HRPWM_EVT、HRPWM_FLT），也可重定向作为通用定时器(TIMER)的输入捕获源。

17.4.5 中断

中断向量	中断事件	中断标志	使能控制位	标志清除位
cmp_int	CMPx 上升沿检测事件	RISx	RISIEx	RISx
	CMPx 下降沿检测事件	FALx	FALIEx	FALx

17.5 寄存器描述

17.5.1 寄存器列表

Name	Offset	Width	Description
CMPx_CR	0x00/0x04/0x08/0x0C	32bits	CMPx 控制寄存器
CMP_SR	0x10	32bits	CMP 输出状态寄存器
CMPx_DEBR	0x14/0x18/0x1C/0x20	32bits	CMPx 消抖寄存器

17.5.2 寄存器详细描述

17.5.2.1 CMPx 控制寄存器 (CMPx_CR)

- **Name**：CMPx Control Register(x = 0...3)
- **Size**：32bits
- **Offset**：0x00/0x04/0x08/0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12]	CxOPOL	R/W	0x0	CMPx(x = 0...3)输出极性选择 (CMPx Output Polarity selection(x = 0...3)) 0：输出不翻转 1：输出翻转
[11]	CxDEBE	R/W	0x0	CMPx(x = 0...3)输出源模式选择 (CMPx Input Debunce Enable(x = 0...3)) 0：禁止输出消抖 1：使能输出消抖 注：当配置为消抖后输出，还需要配置 CMPx_DEBR 寄存器，设定消抖的宽度。
[10-8]	CxBLANK	R/W	0x0	CMPx(x = 0...3)消隐事件选择 (CMPx Blanking source selection) 具体配置值请参考“17.4.3 输出消隐”章节中说明。
[7-6]	CxINMSEL	R/W	0x0	CMPx(x = 0...3)负端输入源选择 (CMPx Inverting input selection(x = 0...3)) 00：GND 01：PAD_INM1_0 10：PAD_INM2_0 11：DACn_OUT 注：PAD 和 DAC 作为负端源的相关配置请参看“17.4.1 引脚和内部信号”章节说明。

[5-4]	CxHYST	R/W	0x0	CMPx(x = 0...3)迟滞选择 (CMPx Hysteresis selection(x = 0...3)) 00：无迟滞(0 mV) 01：低迟滞(10 mV) 10：中等迟滞(20 mV) 11：高迟滞(30 mV)
[3]	FALxIE	R/W	0x0	CMPx(x = 0...3)下降沿中断使能 (CMPx Falling Interrupt Enable(x = 0...3)) 0：不使能 1：使能
[2]	RISxIE	R/W	0x0	CMPx(x = 0...3)上升沿中断使能 (CMPx Rising Interrupt Enable(x = 0...3)) 0：不使能 1：使能
[1]	Reserved	Reserved	Reserved	Reserved
[0]	CxEN	R/W	0x0	CMPx(x = 0...3)使能 (CMPx Enable(x = 0...3)) 0：不使能 1：使能

17.5.2.2 CMP 输出状态寄存器 (CMP_SR)

- **Name**：CMP Out Status Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	C3VAL	R	0x0	CMP3 输出状态 (CMP3 Output Status) 描述参见 C0VAL 位。
[10]	C2VAL	R	0x0	CMP2 输出状态 (CMP2 Output Status) 描述参见 C0VAL 位。
[9]	C1VAL	R	0x0	CMP1 输出状态 (CMP1 Output Status) 描述参见 C0VAL 位。
[8]	C0VAL	R	0x0	CMP0 输出状态 (CMP0 Output Status) 0：输出为低（输出极性变化之前，消隐处理之后） 1：输出为高（输出极性变化之前，消隐处理之后）
[7]	FAL3IF	R/W1C	0x0	CMP3 下降沿中断标志位 (CMP3 Falling Interrupt Flag) 描述参见 FA0IF 位。
[6]	FAL2IF	R/W1C	0x0	CMP2 下降沿中断标志位 (CMP2 Falling Interrupt Flag) 描述参见 FA0IF 位。
[5]	FAL1IF	R/W1C	0x0	CMP1 下降沿中断标志位 (CMP1 Falling Interrupt Flag) 描述参见 FA0IF 位。
[4]	FAL0IF	R/W1C	0x0	CMP0 下降沿中断标志位 (CMP0 Falling Interrupt Flag) 0：无下降沿（输出极性变化之前，消隐处理之后） 1：有下降沿（输出极性变化之前，消隐处理之后）
[3]	RIS3IF	R/W1C	0x0	CMP3 上升沿中断标志位 (CMP3 Rising Interrupt Flag) 描述参见 RIS0IF。
[2]	RIS2IF	R/W1C	0x0	CMP2 上升沿中断标志位 (CMP2 Rising Interrupt Flag) 描述参见 RIS0IF。
[1]	RIS1IF	R/W1C	0x0	CMP1 上升沿中断标志位 (CMP1Rising Interrupt Flag) 描述参见 RIS0IF。
[0]	RIS0IF	R/W1C	0x0	CMP0 上升沿中断标志位 (CMP0 Rising Interrupt Flag) 0：无上降沿（输出极性变化之前，消隐处理之后） 1：有上降沿（输出极性变化之前，消隐处理之后）

17.5.2.3 CMPx 消抖寄存器 (CMPx_DEBR)

- **Name**：CMPx Debounce Register(x = 0...3)
- **Size**：32bits
- **Offset**：0x14/0x18/0x1C/0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11-0]	CxDEB	R/W	0x0	CMPx 输出消抖值 (CMPx Input Debounce Value) CMPx(x = 0...3)输出消抖设置- 最大 4095 0：无消抖 X：CMPx 输出变化的持续时间需要大于 X+1 个系统时钟，才会影响到输出

18 通用定时器（TIMER）

18.1 简介

TAE32F5300 包含高达 8 个通用定时器(TMR0/TMR1/.../TMR7)，每个定时器内包含一个 16 位或 32 位自动重载计数器，计数器由可编程的 16 位或 32 位预分频器驱动，并都包含 16 位或 32 位起始/终止寄存器，用于配置计数器的计数起始/终止值。

通用定时器可用于多种用途，包括最基本的定时功能(基础计时)以及测量输入信号的脉冲宽度(输入捕获)、生成 PWM 输出波形(输出比较)、外部脉冲计数(ETR)等多种灵活配置的功能。

所有通用定时器模块之间是彼此独立，不共享任何资源。并且所有定时器具有相同的功能，只是自动重载递增计数器(TMRx_CNTR)、计数起始值寄存器(TMRx_CSVR)、计数终止值寄存器(TMRx_CEVR) 、预分频寄存器(TMRx_PSCR)、捕获/比较寄存器(TMRx_CCR)的有效位宽不同：

- TMR0/1/2/3 的有效位宽为 16 位
- TMR4/5/6/7 的有效位宽为 32 位

TMR0/1/2/3 四个定时器为一组，具有独立可配的寄存器同步功能；TMR4/5/6/7 四个定时器为一组，具有独立可配的寄存器同步功能。详情请参考“18.4.7 定时器同步”章节。

18.2 结构框图

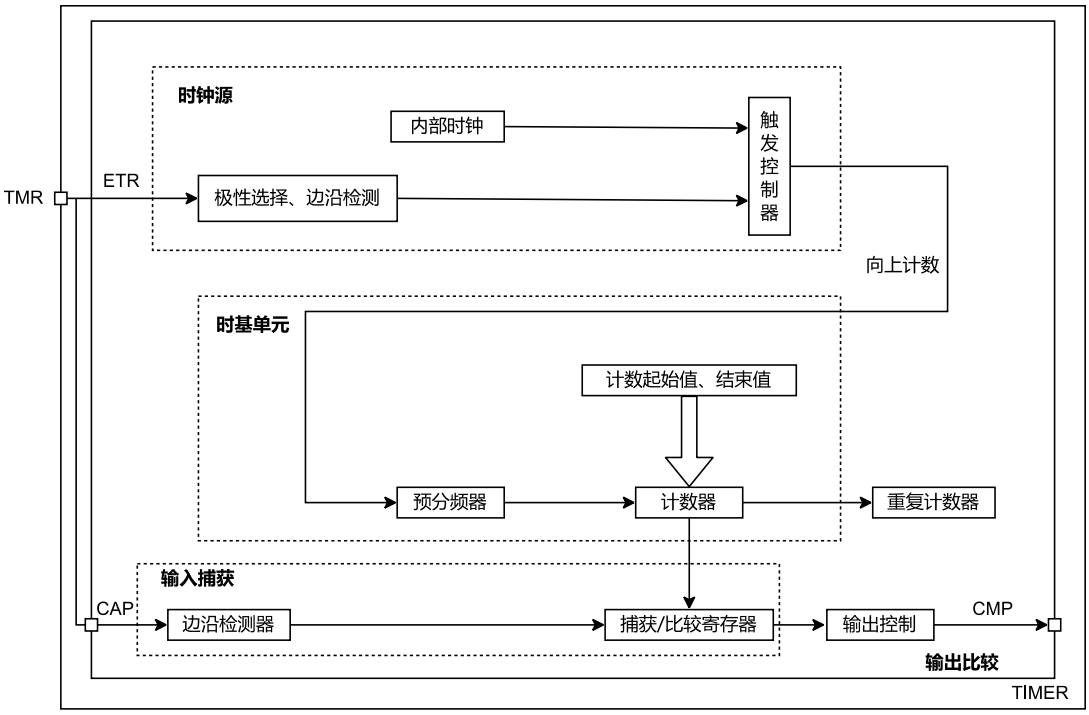


图 18-1 Timer 结构框图

18.3 主要特性

通用定时器 TMRx(x = 0 ... 7)具有以下特性：

- 16 位(TMR0/1/2/3)或 32 位(TMR4/5/6/7)的自动重载递增计数器。
- 16 位(TMR0/1/2/3)或 32 位(TMR4/5/6/7)的预分频器，用于对计数器的时钟源进行预分频。其中 TMR0/1/2/3 以 APB0CLK 作为时钟源，TMR4/5/6/7 以 AHB0CLK 作为时钟源。
- 16 位(TMR0/1/2/3)或 32 位(TMR4/5/6/7)的计数起始值和终止值寄存器，用于灵活设定计数器的开始计数值和终止计数值。
- 两种工作模式：
  - 循环模式，计数完成后自动重载并继续计数
  - 单次模式，计数完成后自动关闭定时器的使能
- 独立通道，可用于：
  - 输入捕获
  - 输出比较
  - PWM 生成
  - ETR 外部触发输入
- 两组定时器同步功能
- 发生如下事件时产生中断：
  - 计数器上溢(Overflow Event)
  - 更新事件(Update Event) (需使能)：计数器上溢产生更新事件，也可软件产生(UG)
  - 输入捕获(Capture Event)、重复捕获(Over Capture Event)
  - 输出比较(Compare Event)

18.4 功能描述

18.4.1 时基单元

通用定时器的主要模块由一个 16/32 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。

通用定时器的自动重载计数器(TMRx_CNTR)、计数起始值寄存器(TMRx_CSVR)、计数终止值寄存器(TMRx_CEVR)和预分频寄存器(TMRx_PSCR)可通过软件进行读写，即使在计数器运行中也可执行读写操作。

时基单元包含：

- 自动重载计数器(TMRx_CNTR)
- 预分频寄存器(TMRx_PSCR)
- 计数起始值寄存器(TMRx_CSVR)
- 计数终止值寄存器(TMRx_CEVR)

其中预分频寄存器(TMRx_PSCR)、计数起始值寄存器(TMRx_CSVR)及计数终止值寄存器(TMRx_CEVR)都是预装载的。对预装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立即生效，也可以在每次发生更新事件时

更新到影子寄存器，这取决于定时器控制寄存器(TMRx_CR)中的自动重载预装载使能位(ARPE)。

定时器计数器达到上溢值，且控制寄存器(TMRx_CR)中的 UDIS 为 0 时，将产生更新事件。此外，该更新事件也可通过软件发起，通过对定时器事件生成寄存器(TMRx_EGR)中的 UG 位置 1。

更新事件相关的详细介绍，请参看“18.4.9.2 更新事件”章节。

定时器的计数器由时钟源经过预分频器分频后提供的时钟驱动，且仅当控制寄存器(TMRx_CR)中的计数器使能位(CEN)置 1 时，才会启动计数器计数。

注意：计数器将在控制寄存器(TMRx_CR)中的计数器使能位(CEN)置 1 时刻后的一个时钟周期开始计数。

预分频器说明

预分频器用于对计数器的计数时钟频率进行分频，通过配置预分频器寄存器(TMRx_PSCR)设定预分频的分频系数。计数器计数频率公式如下：

$$f_{CK\_CNT} = f_{CK\_SRC} / (PSC + 1)$$

该寄存器具有预加载缓存功能，因此可以对预分频器实现实时修改，而新的预分频比将在下一个更新事件发生时被采用。也可通过软件设置 UG 位，立即产生一个更新事件（UEV）。

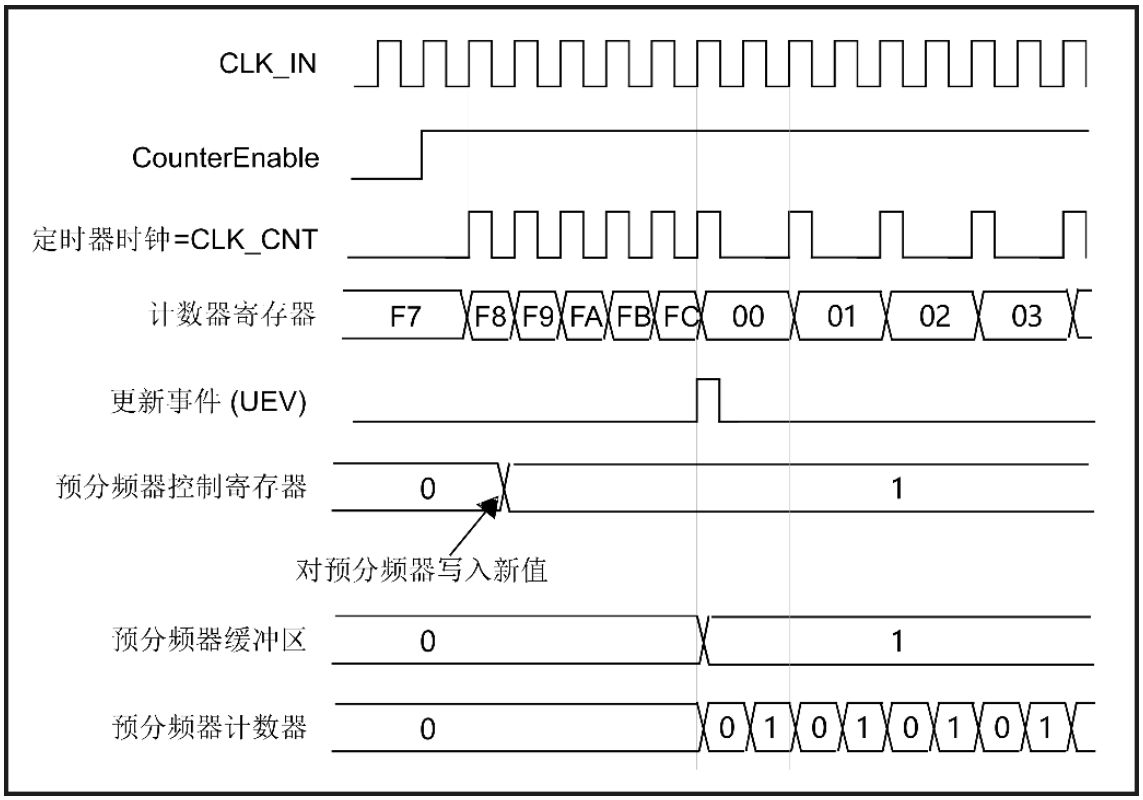


图 18-2 实时修改预分频器的分频值从 1 变为 2 的时序图

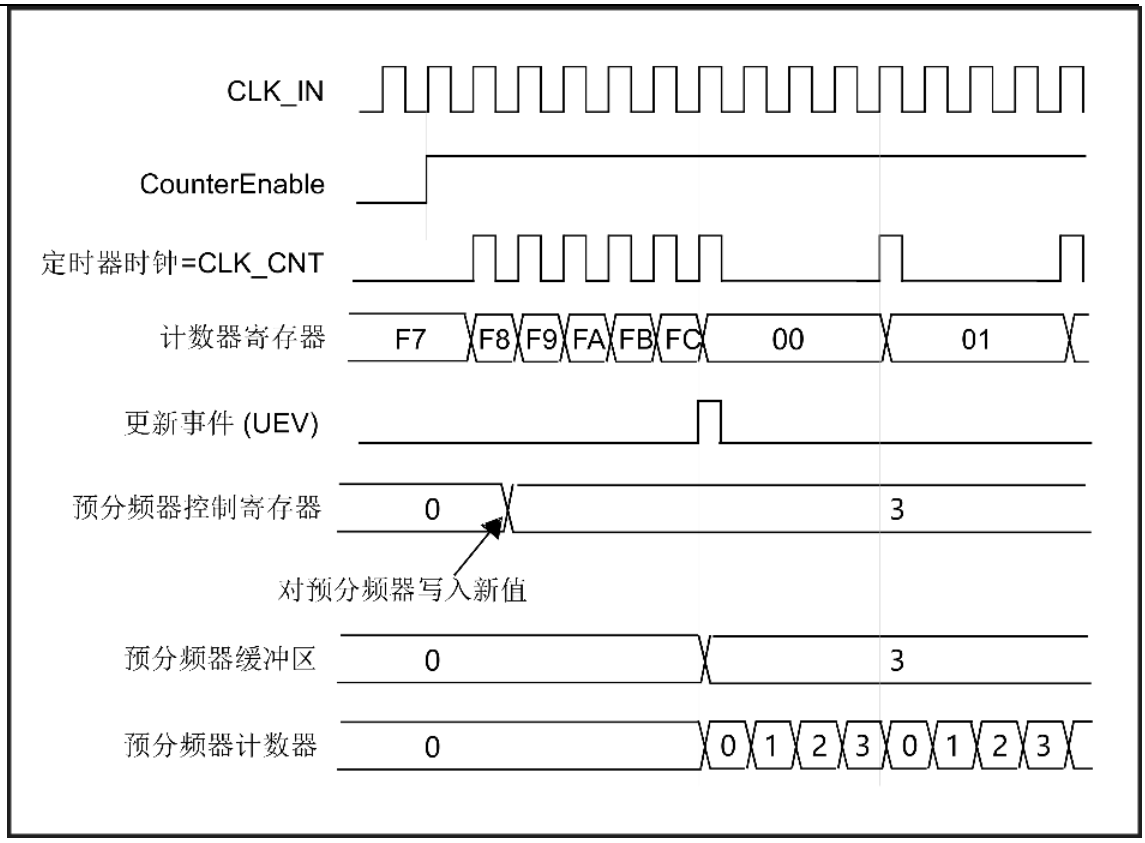


图 18-3 实时修改预分频器的分频值从 1 变为 4 的时序图

18.4.2 计数器模式

通用定时器仅支持递增计数模式，计数器从起始值(TMRx_CSVR)计数到结束值(TMRx_CEVR - 1)，支持单次和连续计数模式：单次模式下，计数完成后将自动关闭计数器使能(CEN 自动置 0)停止计数；连续模式下，每次计数完成后，自动从起始值(TMRx_CSVR)开始重新计数，并生成计数器上溢事件 OVEV 和更新事件 UEV(如果使能更新事件)。

通过设置定时器控制寄存器(TMRx_CR)中的禁止更新位(UDIS)置 1 可禁止更新事件 UEV，禁止之后将不会产生任何的更新事件，直到禁止更新位重新置 0。这样可避免向预装载寄存器写入新值时更新影子寄存器，不过计数仍然能够在上溢事件 OVEV 后，重新从起始值开始重新计数。

此外，如果定时器控制寄存器(TMRx_CR)中更新请求源位(URS)置 1，禁止更新位(UDIS)置 0，然后通过软件将定时器事件产生寄存器(TMRx_EGR)中的更新产生位(UG)置 1，则只会生成更新事件(UEV)，但是不会将更新中断标志位(UIF)置 1，因此不会发起任何中断。

定时器发生更新事件(UEV)时，将更新相关寄存器的值且将更新中断标志位(UIF)置 1(这同时取决于 URS 位)：

- 自动重载影子寄存器将更新为预装载寄存器(TMRx_CEVR)的值
- 预分频器影子寄存器将更新为预装载寄存器(TMRx_PSCR)的值
- 捕获/比较寄存器更新为预装载寄存器(TMRx_CCR)的值(非通过软件设置 UG 位更新事件时)

以下图 18-4～ 图 18-9 将通过一些示例说明当计数器等于 0x36 时，不同时钟频率下表现的行为。

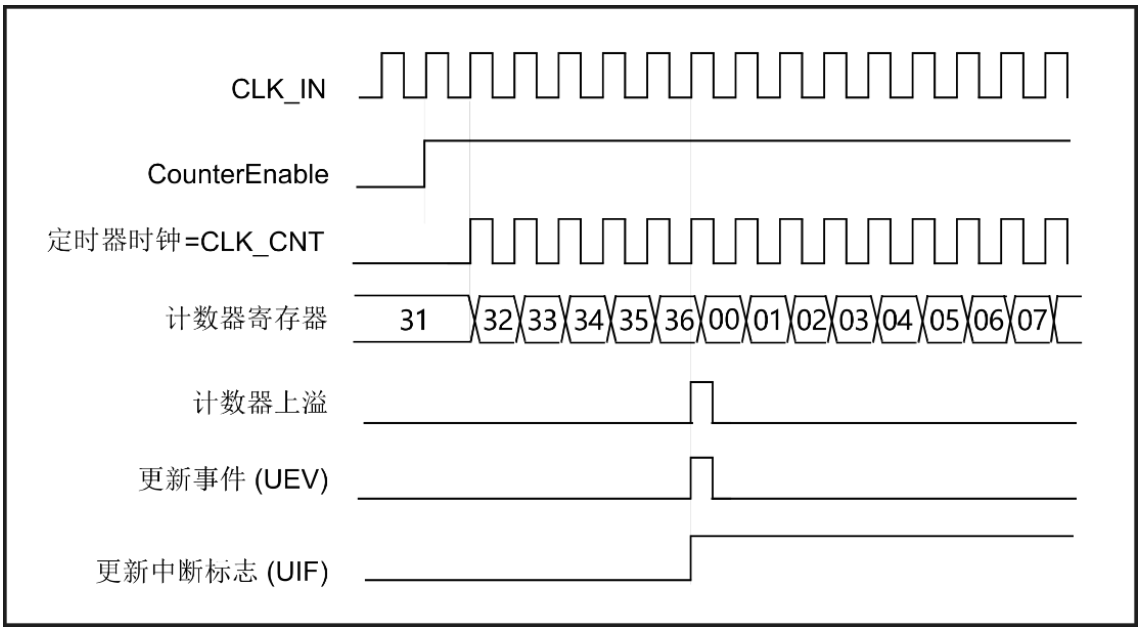


图 18-4 计数器时序图，1 分频内部时钟

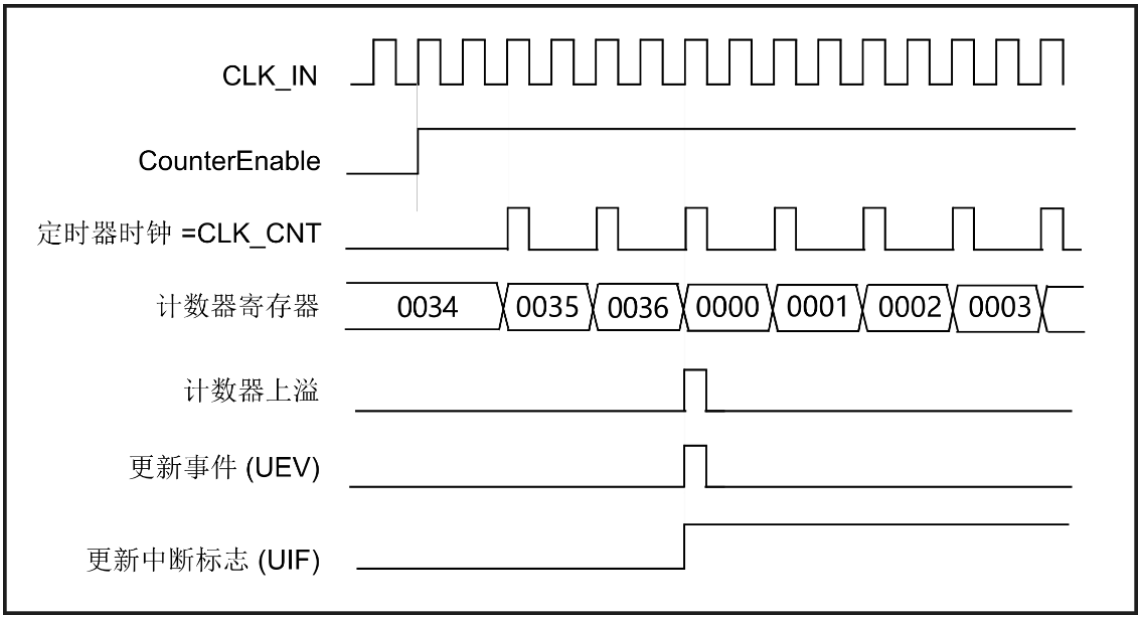


图 18-5 计数器时序图，2 分频内部时钟

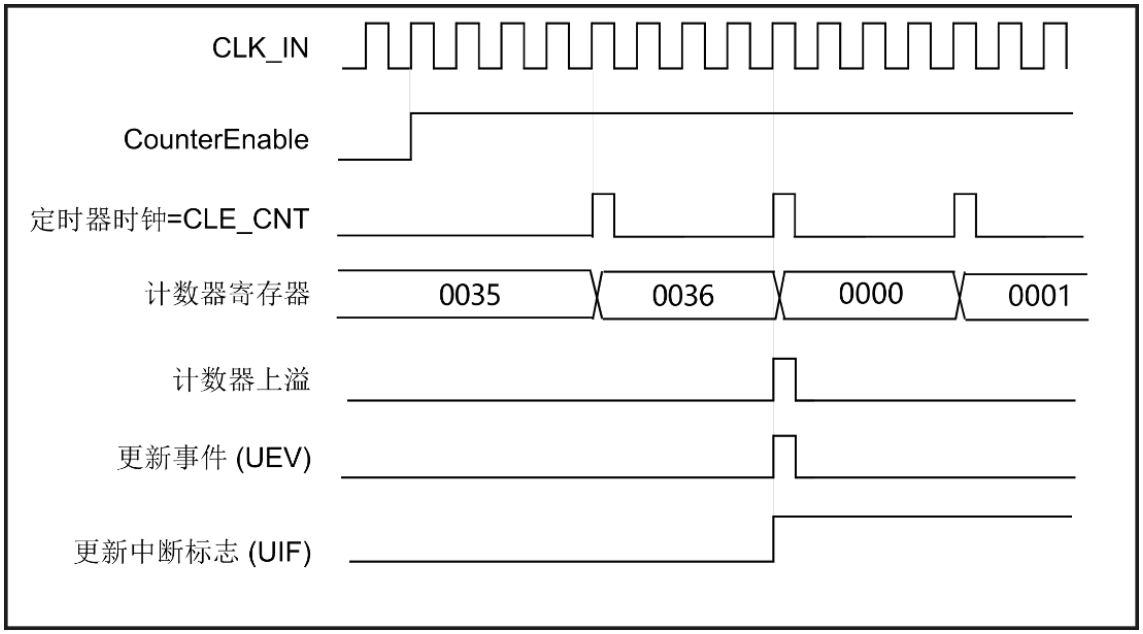


图 18-6 计数器时序图，4 分频内部时钟

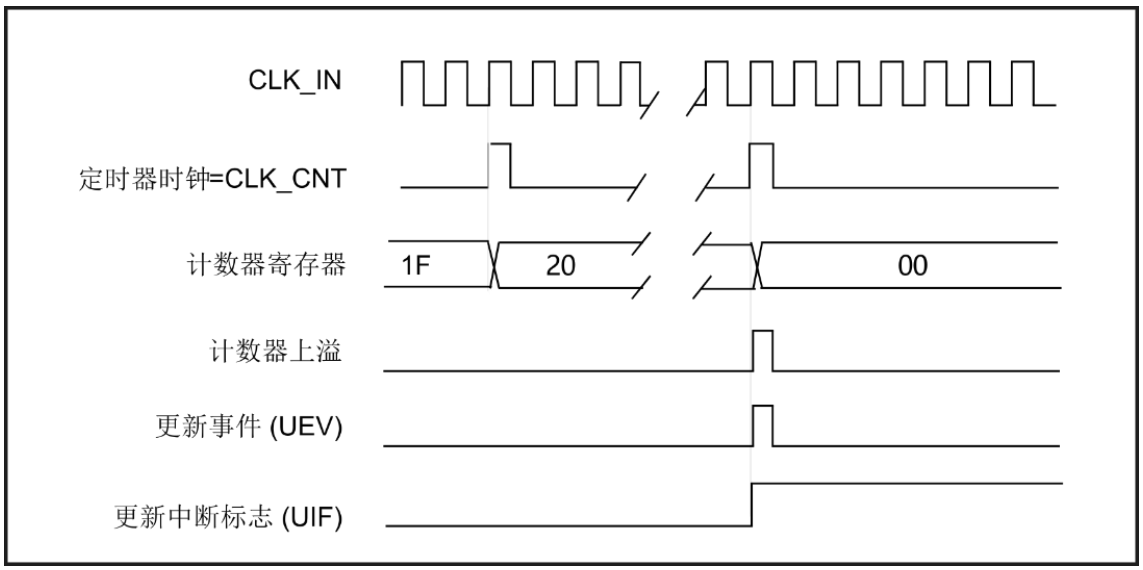


图 18-7 计数器时序图，N 分频内部时钟

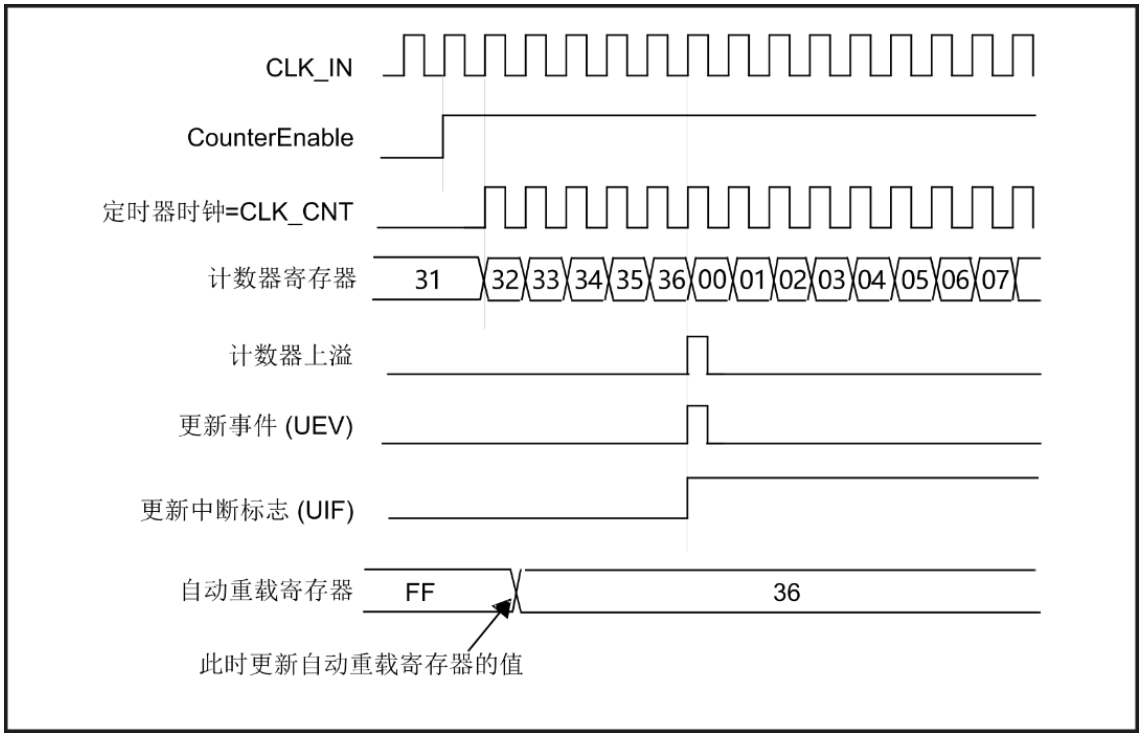


图 18-8 计数器时序图，运行中更新重载值（自动重载使能关闭-实时生效）

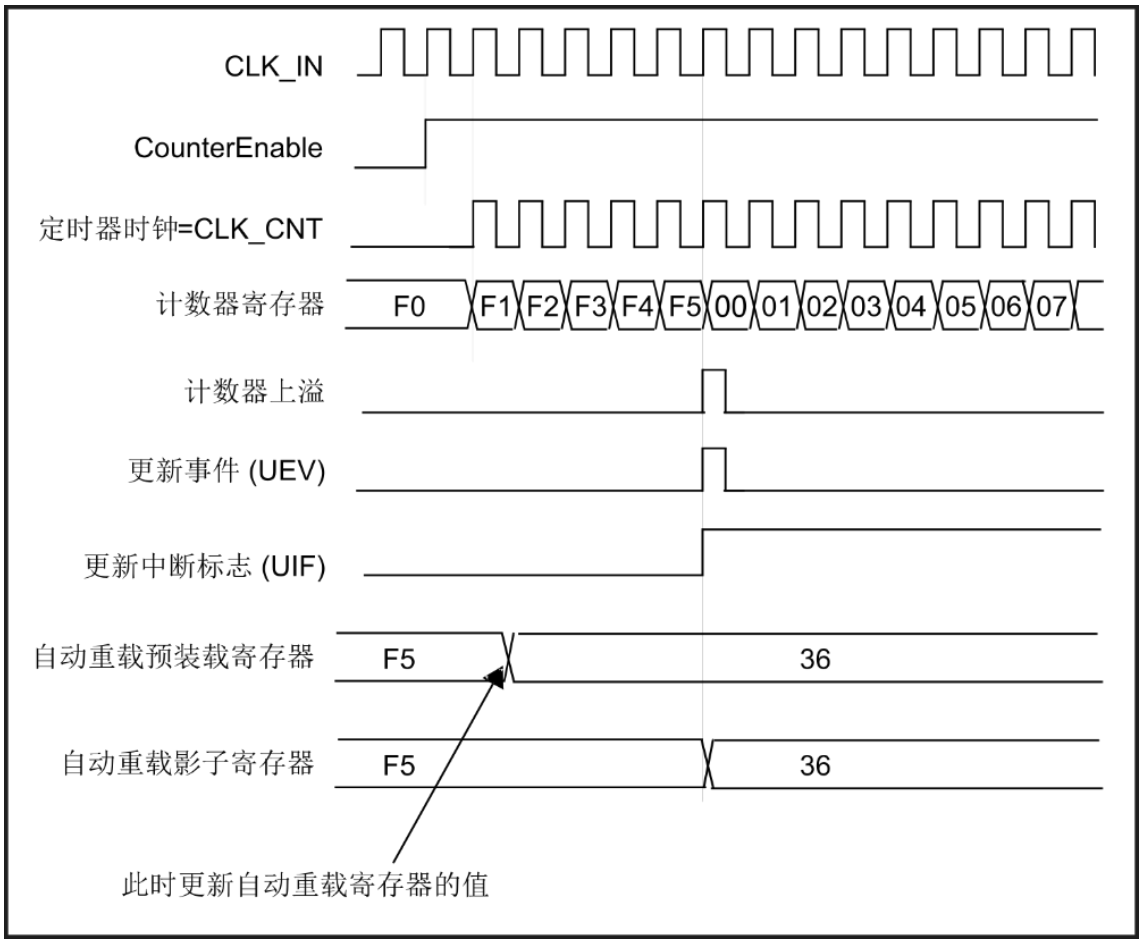


图 18-9 计数器时序图，运行中更新重载值（自动重载使能使能-更新事件生效）

18.4.3 时钟选择

定时器的计数器时钟源分为以下两类：

- 内部时钟源（CK_IN）：系统时钟上升沿（APB0CLK）
- 外部时钟（ETR）：通过外部引脚输入时钟，并可配置触发边沿（上升/下降/双边沿）

内部时钟源（CK_IN）

下图为正常模式下内部控制逻辑及计数器计数的行为（预分频设置为 0，即在没有预分频情况下）。

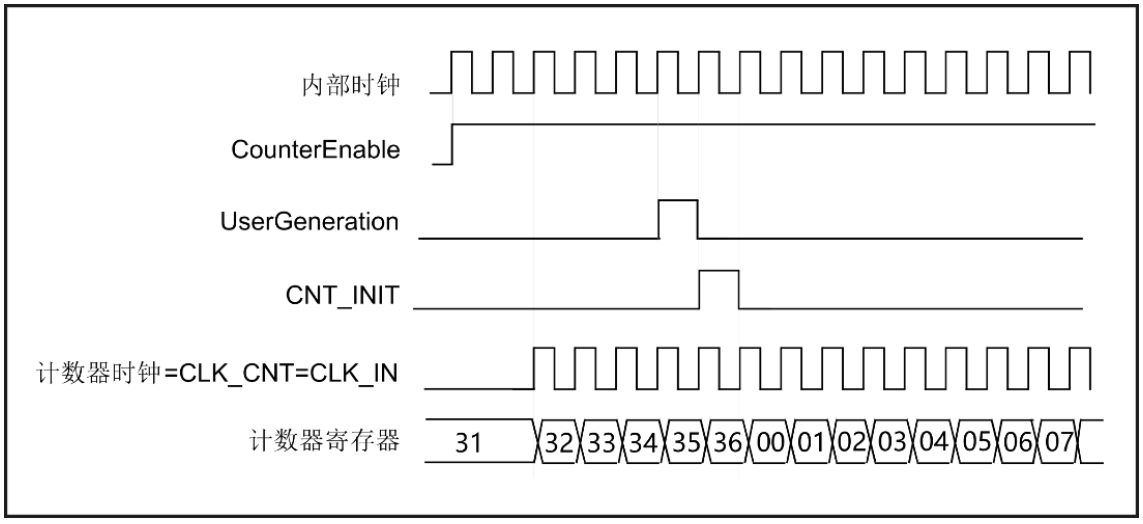


图 18-10 正常模式下的控制时序图（没有预分频情况）

外部时钟源（ETR）

定时器可选取外部时钟源作为计数器的时钟源。外部时钟源由定时器外设的功能引脚灌入，通过内部边沿检测电路触发计数器计数，可利用外部时钟源分为三种边沿，作为计数器时钟的触发：

- 上升沿计数
- 下降沿计数
- 双边沿计数

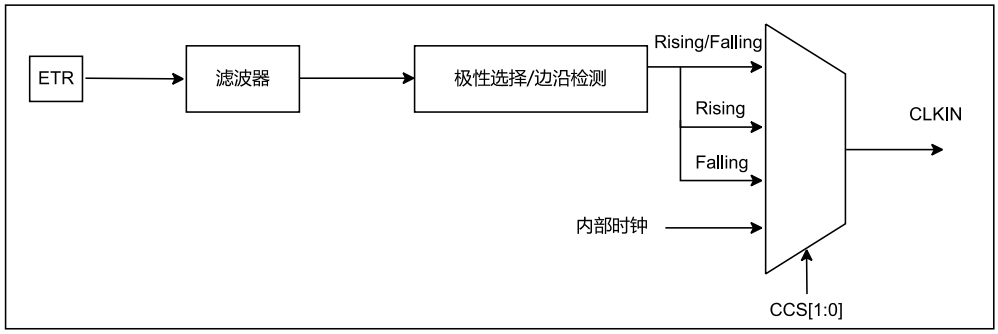


图 18-11 外部时钟连接示意图

注意：外部输入时钟源从功能引脚输入会经过一个滤波器消抖，该滤波器的消抖使能在 GPIO 模块中的“同步/消抖使能寄存器(GPIOx_SDER)”中设置。开启滤波器功能后，只有当输入电平宽度至少达到 4 个时钟周期（时钟源可配，具体参考 GPIO 文档）才会被送出，否则会被当作信号抖动被滤波器滤掉。

18.4.4 捕获/比较通道

每个通用定时器均带有一个独立的捕获/比较通道，捕获/比较通道均基于一个捕获/比较寄存器(包括一个影子寄存器)、一个捕获输入(滤波器、极性判断和预分频器)和一个比较输出(比较器和输出控制)组成。

输入捕获，是对相应的端口输入信号进行采样，经过滤波器处理后输出，再通过极性选择和边沿检测生成一个触发信号作为输入捕获触发信号，用于及时捕获到指定边沿发生时，计数器的准确数值。通过这样可实现对外部信号的精准捕获和测量，由此可实现用于对信号进行脉宽测量、周期测量、计算占空比等功能。

输出比较，是利用计数器计数，并与预先配置好的比较值进行对比，当计数值与比较值匹配时，根据所设定的比较输出模式，相应的端口进行相应的输出，从而实现例如 PWM、反转、强制极性输出等功能。

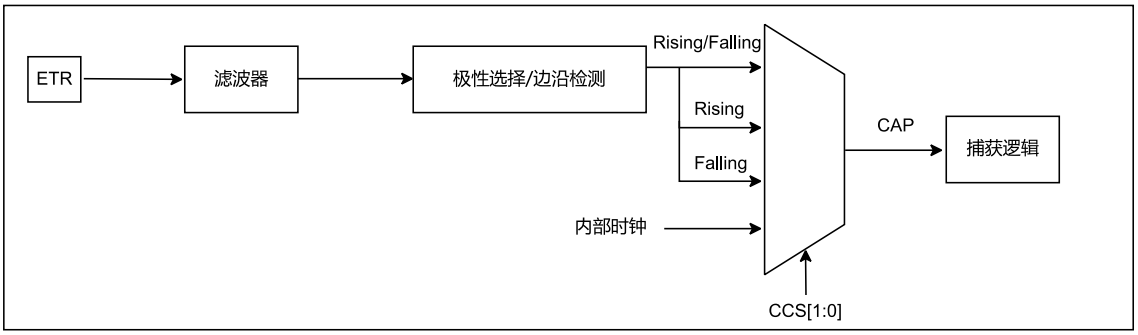


图 18-12 输入捕获通道示意图

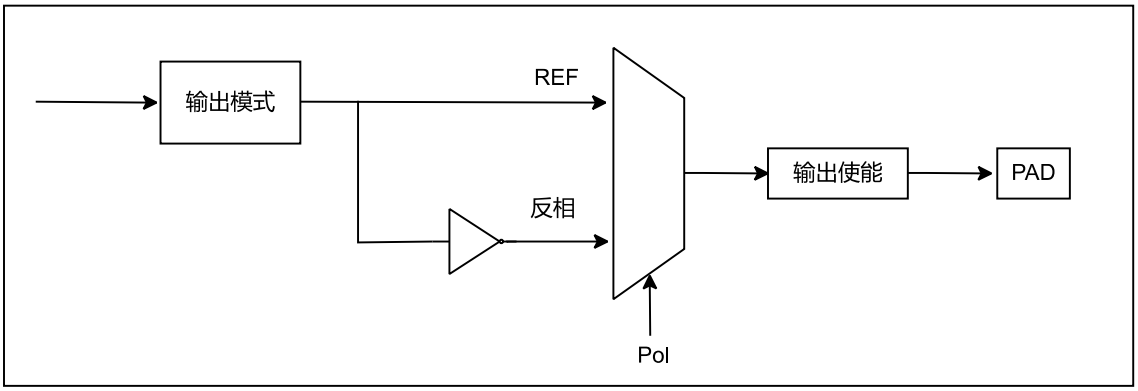


图 18-13 比较输出通道示意图

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。

18.4.5 输入捕获

定时器可通过配置捕获/比较控制寄存器(TMRx_CCCR)中的模式选择位(CCS=1)，并设置使能捕获/比较功能(CCE=1)，使定时器工作在输入捕获模式下。

此外，通过配置捕获/比较控制寄存器(TMRx_CCCR)中的极性选择位域(CCP[1:0])可以选择捕获模式下的触发边沿，配置输入捕获源选择位域(ICSRC[2:0])可选择捕获的目标功能引脚或使用比较器模块(CMPx)的内部信号。

定时器处于输入捕获模式下后，当检测到捕获源产生了目标跳变沿之后，定时器立刻将当前计数器的值锁存捕获到捕获/比较寄存器(TMRx_CCR)内，同时设置输入捕获中断标志位(ICIF)置 1，并触发定时器中断(如果 ICIE=1)，软件上需要及时对输入捕获中断标志位进行清除(ICIF 写 1 清 0，或通过读取捕获/比较寄存器，硬件将自动清除 ICIF 位)。如果在清除该标志位之前，再次捕获到目标跳变沿，会将新的计数值锁存并覆盖捕获/比较寄存器(TMRx_CCR)内的值，同时将会产生重复采样导致输入捕获重复采样标志位(ICOIF)将被置 1，并触发定时器中断(如果 ICOIE=1)。因此在处理输入捕获时，建议在捕获状态标志位置起后应当及时读取数据。否则一旦发生重复捕获，新捕获的数据将覆盖掉之前捕获的数据，导致捕获信息丢失。

18.4.6 输出比较

定时器可通过配置捕获/比较控制寄存器(TMRx_CCCR)中的模式选择位(CCS=0)，并设置使能捕获/比较功能(CCE=1)，使定时器工作在输出比较模式下。

此外，通过配置捕获/比较控制寄存器(TMRx_CCCR)中的比较输出模式位域(OCM[2:0])选择输出比较的工作模式、极性选择位域(CCP[1:0])可以选择比较模式下有效电平的极性、设置输出比较预装载使能位(OCPE)可使能输出比较的预装载功能。

当定时器的计数器的值(TMRx_CNTR)与捕获/比较寄存器(TMRx_CCR)的值相匹配时，将根据目前设置的输出比较工作模式对功能脚输出相应的信号，同时输出比较中断标志位(OCIF)将被置 1，并触发定时器中断(如果 OCIE=1)，软件上可通过对输出比较中断标志位进行清除（OCIF 写 1 清 0）。

输出比较工作模式

目前定时器支持的输出比较工作模式有以下几种，通过 OCM[2:0]位域进行设置：

- 000 ：冻结，比较匹配后不对功能脚产生变化，保持原有输出。
- 001 ：匹配后输出有效电平（有效电平极性通过 CCP[1:0]进行设置）
- 010 ：匹配后输出无效电平（与有效电平相反）
- 011 ：匹配后翻转电平
- 100 ：强制变为无效电平
- 101 ：强制变为有效电平
- 110 ：PWM 模式 1。计数器值 <= 比较值时，输出有效电平，否则为无效电平。
- 111 ：PWM 模式 2。当计数器值 <= 比较值时，输出无效电平，否则为有效电平。

配合定时器中的单次/连续计数模式，还可实现更多组合功能，例如利用单次计数模式 + 输出

比较 PWM 模式可实现单脉冲输出功能。更多特别模式说明将在后续章节进行更详细介绍。

输出比较的预装载功能

在输出比较的过程中，软件可通过修改捕获/比较寄存器(TMRx_CCR)来修改用于比较的值，通过设置输出比较预装载使能控制位(OCPE)， 可配置是否开启上述寄存器的预装载功能。预装载的作用如下：

- 不开启预装载功能，向 TMRx_CCR 写入新值，将立即更新到内部影子寄存器内并立即生效。
- 开启预装载功能，向 TMRx_CCR 写入新值，并不会立即生效，而是在得到比较成功或软件产生捕获/比较事件(CCEV)之后，再将数值更新至内部影子寄存器内生效。更详细的捕获/比较事件描述，请参考“18.4.9.3 捕获/比较事件”章节说明。

18.4.6.1 强制输出

在输出模式下，可直接通过软件将输出比较信号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比较结果。

配置强制输出模式，只需将 TMRx_CCCR 寄存器中比较输出模式选择位域(OCM[2:0])配置为 100 或 101，在配置极性位域(CCP[1:0])选择合适的有效电平。

在配置为强制输出模式后，虽然定时的功能引脚输出将会根据配置强制输出指定的电平，但定时器的计数器值与比较值的匹配行为依然在执行，匹配时依然会引发相应的标志位(OCIF)并触发中断处理。

- 配置强制输出无效电平模式(OCM=100)时：
  - CCP=0（有效电平为高电平），将比较强制输出为低电平
  - CCP=1（有效电平为低电平），将比较强制输出为高电平
- 配置强制输出有效电平模式(OCM=101)时：
  - CCP=0（有效电平为高电平），将比较强制输出为高电平
  - CCP=1（有效电平为低电平），将比较强制输出为低电平

18.4.6.2 PWM 输出

输出比较的 PWM 模式，可以用于生成一个调制信号，该信号频率周期由计数器配置的基础定时周期决定(也就是由 TMRx_PSCR、TMRx_CSVR、TMRx_CEVR 寄存器共同决定)，而占空比则由捕获/比较寄存器(TMRx_CCR)设定的值决定。PWM 模式有两种：

- PWM 模式 1（OCM=110）

该模式下，当计数器值 < 比较值时输出有效电平，否则输出无效电平。

- PWM 模式 2（OCM=111）

该模式下，当计数器值 < 比较值时输出无效电平，否则输出有效电平。

有效电平通过捕获/比较控制寄存器(TMRx_CCCR)中的极性设置位域(CCP[1:0])选择合适的有效电平。

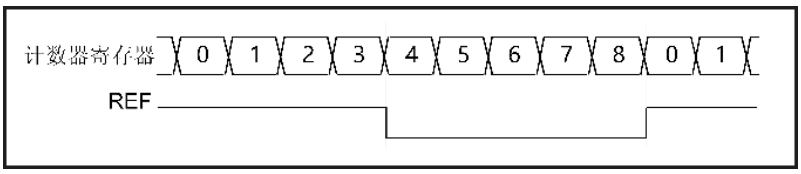


图 18-14 PWM 输出（TMRx_CEVR=8，TCCRR=4，Polarity=0 高电平有效）

18.4.6.3 单脉冲模式

单脉冲模式 (One-Pulse Mode) 是单次计数模式+PWM 模式的一个特例。在这种模式下，可以配置输出一个可控脉宽的单脉冲信号。

首先将定时器模块配置为单次计数模式(MS=1)，配置输出比较工作模式为 PWM 模式 1/2，配置捕获/比较寄存器(TMRx_CCR)写入合适的比较值，再根据电平需求配置有效电平的极性(CCP[1:0])。这样启动定时器计数后，将会输出一段可控宽度的单脉冲。

18.4.7 定时器同步

每个定时器都是完全独立的，定时器之间没有共享任何资源。为了实现定时器同步启动功能，对通用定时器进行了分组并提供了同组内定时器实现同步启动的功能。其中：

- TMR0/1/2/3 为一组，称为 TMRGRP0
- TMR4/5/6/7 为一组，称为 TMRGRP1

通过定时器组同步控制寄存器(TMRGRPx_SYNCR)，实现同组定时器的同步启动功能。

18.4.8 调试模式

当微控制器进入调试模式（Cortex™-M3 内核停止）时，定时器内部计数器会根据 DBG 模块中的 TMRGRP0_DBG_EN/TMRGRP1_DBG_EN 配置位选择继续正常工作或者停止工作。其中：

- TMRGRP0 包括 TMR0/1/2/3 四个定时器
- TMRGRP1 包括 TMR4/5/6/7 四个定时器

18.4.9 事件与中断

通用定时器在多种情况下会产生相应的事件，事件主要用于更新相应的影子寄存器、执行特定功能以及产生相应中断。

通用定时器内主要有以下几种事件：

- 定时器上溢事件，简称上溢事件 OVEV(Overflow Event)
- 定时器更新事件，简称更新事件 UEV (Update Event)

需要使用更新事件，必须将定时器控制寄存器(TMRx_CR)中的禁止更新位(UDIS)置 0。

- 定时器输入捕获/输出比较事件，简称捕获/比较事件 CCEV (Capture/Compare Event)

中断主要包含以下五种中断触发：

- 定时器上溢中断，发生上溢事件后，上溢中断标志位(OVIF)同时被置位。
- 定时器更新中断，发生更新事件后，更新中断标志位(UIF)同时被置位。
  特别说明：如果更新请求源选择位(URS)置 1，即仅上溢事件会产生更新事件，那么软件上如果通过设置更新产生位(UG)，将只会产生更新事件用于更新预加载相关的寄存器，但不会触发定时器更新中断。
- 定时器输入捕获中断，当定时器为输入捕获模式并捕获到相应的边沿时，输入捕获中断标志位（ICIF）将被置位。
- 定时器重复捕获中断，当输入捕获中断标志位已经置位(ICIF=1)，再次捕获到相应边沿，则输入捕获重复捕获中断标志位（ICOIF）将被置位。
- 定时器输出比较中断，当定时器位输出比较模式，并且计数值与比较值相匹配时，输出比较中断标志位(OCIF)将被置位。

18.4.9.1 上溢事件（Overflow Event）

上溢事件(OVEV)由以下情况产生：

- 计数器寄存器(TMRx_CNTR)计数值到达计数终止值寄存器(TMRx_CEVR)设定的值。

上溢事件的作用

计数器到达上溢后，计数器寄存器会立即重新加载计数值起始寄存器(TMRx_CSVR)的值，重新开始计数(如果为连续模式)。

事件产生后，中断状态寄存器(TMRx_ISR)中的上溢中断标志位(OVIF)将会置 1，并且会触发定时器中断(如果 OVIE = 1)。

18.4.9.2 更新事件（Update Event）

更新事件(UEV)可以由以下两种情况产生：

- **计数器上溢**：计数器上溢事件产生的同时会产生更新事件。
- **软件置位**：软件将定时器事件产生寄存器(TMRx_EGR)中的更新标志位(UG) 置 1。

更新事件使用的注意事项

- 如果需要使用更新事件(UEV)，必须将定时器控制寄存器(TMRx_CR)中的更新禁止位(UDIS)置 0，使能更新事件功能，否则更新事件将被禁止。
- 如果需要**软件置位**功能，还必须要配置定时器控制寄存器(TMRx_CR)中的更新请求位(URS)置 0。否则**只有计数器上溢**才会产生更新事件。

更新事件的作用

事件产生后，中断状态寄存器(TMRx_ISR)中的更新中断标志位(UIF)将会置 1，并且会触发定时器中断 (如果 UIE = 1)。

特别的，如果在计数器计数的过程中(未达到计数器上溢)，通过软件设置 UG 位产生更新事件后，计数器寄存器也会立即重新加载计数值起始寄存器(TMRx_CSVR)的值，重新开始计数(如果为连续模式)。不同之处在于：通过软件更新标志位(UG)设置产生的更新事件，不会触发上溢事件的中断。

更新事件对自动装载预加载(Auto-Reload Preload)功能的作用

当定时器**开启自动装载预加载功能**（TMRx_CR 寄存器内 ARPE=1），更新事件产生后，才会将以下几个寄存器的值，更新到内部影子寄存器内使之生效：

- 计数终止值寄存器(TMRx_CEVR)
- 预分频寄存器(TMRx_PSCR)
- 捕获/比较寄存器(TMRx_CCR)

注意：软件置位(UG=1)产生的更新事件，不会影响捕获/比较寄存器(TMRx_CCR)。

如果定时器**未开启自动装载预加载功能**（TMRx_CR 寄存器内 ARPE=0），则上述寄存器写入新值将立即生效，更新事件将不影响上述寄存器的配置。

18.4.9.3 捕获/比较事件（Capture/Compare Event）

捕获/比较事件(CCEV)，可以由以下三种情况产生：

- **比较成功**：计数器寄存器(TMRx_CNTR)的计数值与捕获/比较寄存器(TMRx_CCR)的值比较成功时(仅针对**输出比较模式**下)。
- **捕获成功**：外部输入一个与捕获/比较控制寄存器(TMRx_CCCR)中所设置的极性(CCP 位)相符的边沿信号时（仅针对**输入捕获模式**下）。
- **软件产生**：定时器事件产生寄存器(TMRx_EGR)中相应的事件产生标志位(CCG)置 1。

捕获/比较事件的作用

在不同模式下捕获/比较事件的作用不相同：

- 对于输出比较模式：
  - 捕获/比较事件产生后，中断状态寄存器(TMRx_ISR)中的比较中断(OCIF)将会置 1，并触发定时器中断(如果比较中断使能，OCIE 位置 1)。
  - 如果使能了**比较器预加载功能**(捕获/比较控制寄存器 TMRx_CCCR 中的 OCPE 位置 1)，事件产生后(这里特指**比较成功**所产生事件)，定时器将会执行以下动作：
    - 定时器的输出，将根据捕获/比较控制寄存器(TMRx_CCCR)中设置的比较输出模式(OCM[2:0])，对输出状态进行相应的改变；
    - 将捕获/比较寄存器(TMRx_CCR)的值重新载入内部影子寄存器，使之生效。
    - 将定时器的计数器值(TMRx_CNTR)重新加载为计数起始值寄存器(TMRx_CSVR)的值，并重新开始计数(循环模式)。

- 如果没有使能**比较器预加载功能**(捕获/比较控制寄存器 TMRx_CCCR 中的 OCPE 位置 0)，事件产生后，定时器的动作**与上述情况不同之处在于**：对捕获/比较寄存器(TMRx_CCR)写入**新值将立即生效**，捕获/比较事件不会影响该寄存器值何时生效。

*注意：如果在到达比较值之前(未达到**比较成功**)，通过软件对 CCG 置 1 产生事件，与上述描述基本一致，**不同之处在于：不会导致比较输出模式(OCM[2:0])所设定的输出变化**，因为本次计数器值并没有与触发捕获/比较事件前所设定的比较值相匹配。*

- 对于输入捕获模式：
  - 捕获/比较事件产生后，定时器将当前计数器寄存器(TMRx_CNTR)的值捕获到捕获/比较寄存器(TMRx_CCR)内；
  - 接着中断状态寄存器(TMRx_ISR)中的捕获中断标志(ICIF)将会置 1，并触发定时器中断(如果 ICIE=1)。如果本次捕获成功产生捕获/比较事件时，前一次捕获中断标志未及时清除(ICIF 未清 0)，则中断状态寄存器(TMRx_ISR)中的重复捕获中断标志(ICOIF)将会置 1，并且触发定时器中断(如果使能 ICOIE 置 1)。

*注意：如果通过软件对 CCG 置 1 产生的捕获/比较事件，则**立刻执行上述动作**，而不用外部产生符合捕获成功的信号。*

18.4.9.4 中断

中断标志位与中断使能之间的关系如下表描述所示，详细内容请参考“18.4.9 事件与中断”章节描述：

表 18-1 中断标志位与中断使能之间的关系

中断名称	中断使能	中断标志	清除方式	备注
上溢中断	OVIE	OVIF	W1C	计数器计数上溢后产生
更新中断	UIE	UIF	W1C	计数器更新事件触发更新中断
输入捕获中断	ICIE	ICIF	W1C 或读 CCR	输入捕获模式下，捕获到目标边沿
重复捕获中断	ICOIE	ICOIF	W1C	输入捕获中断未及时清除，再次发生捕获
输出比较中断	OCIE	OCIF	W1C	输出比较模式下，计数值与比较值匹配

18.5 寄存器描述

18.5.1 寄存器列表

表 2. 定时器寄存器列表

Name	Offset	Width	Description
TMRx_CR	0x00	32bits	TMRx 控制寄存器
TMRx_CCCR	0x04	32bits	TMRx 捕获/比较控制寄存器
TMRx_EGR	0x08	32bits	TMRx 事件生成寄存器
TMRx_ICFR	0x0C	32bits	TMRx 输入捕获滤波寄存器
TMRx_ISR	0x10	32bits	TMRx 中断状态寄存器
TMRx_CSVR	0x20	32bits	TMRx 计数起始值寄存器
TMRx_CEVR	0x24	32bits	TMRx 计数终止值寄存器
TMRx_CCR	0x28	32bits	TMRx 捕获/比较寄存器
TMRx_PSCR	0x2C	32bits	TMRx 预分频寄存器
TMRx_CNTR	0x30	32bits	TMRx 计数寄存器
TMRx_ETER	0x34	32bits	TMRx 输出触发事件寄存器

表 3. 定时器分组寄存器列表

Name	Offset	Width	Description
TMRGRPx_SYNCR	0x00	32bits	定时器组同步寄存器

18.5.2 寄存器详细描述

18.5.2.1 TMRx 控制寄存器 (TMRx_CR)

- **Name**：TMRx Control Register (x = 0 ... 7)
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-14]	Reserved	Reserved	Reserved	Reserved
[13-12]	CKSRC	R/W	0x0	定时器时钟源 (Timer Clock Source) 00：系统时钟上升沿 01：外部时钟上升沿 10：外部时钟下降沿 11：外部时钟上下沿
[11-10]	Reserved	Reserved	Reserved	Reserved
[9]	UIE	R/W	0x0	更新事件中断使能控制 (Timer Update Interrupt Enable) 1：使能中断 0：禁止中断 注意：如果需要使用更新事件，则要配置更新事件功能开启(TMRx_CR 寄存器的 UDIS 位置 0)。更详细介绍请参看“18.4.9.2 更新事件”章节。
[8]	OVIE	R/W	0x0	计数器上溢中断使能控制 (Timer Counter Overflow Interrupt Enable) 1：使能中断 0：禁止中断
[7-5]	Reserved	Reserved	Reserved	Reserved
[4]	ARPE	R/W	0x0	定时器自动重载预装载使能 (Timer Auto-Reload Preload Enable) 此位用于开启/关闭影子寄存器预加载功能。 0：寄存器不进行预加载 1：寄存器进行预加载 注意： 1. 预加载功能涉及的寄存器有： – 计数终止值寄存器(TMRx_CEVR) – 预分频寄存器(TMRx_PSCR) – 捕获/比较寄存器(TMRx_CCR) 但他们的更新事件分别由不同的情况产生。 2. 如果不开启预加载功能，则相关寄存器写入新值将立刻生效 3. 如果开启预加载功能，则相关寄存器写入新值后，在下一个更新事件(UEV)到来后生效。 更详细信息请参看“18.4.9.2 更新事件”章节。

[3]	MS	R/W	0x0	定时器工作模式选择 (Timer Mode Selection) 0：连续模式 1：单次模式 注意：配置为单次模式后，计数器溢出后，将自动停止计数。
[2]	URS	R/W	0x0	定时器更新事件源选择 (Timer Update Request Source) 0：使能后，所有以下事件都会生成更新事件： – 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事件。
[1]	UDIS	R/W	0x0	定时器更新事件禁止位 (Timer Update Disable) 此位由软件置 1 和清零，用以使能/禁止更新事件生成。 0：更新使能，更新 (UEV) 事件可通过以下事件之一生成： – 计数器上溢 – 将 UG 位置 1 1：更新禁止 注意：配置为 1 不会生成更新事件(不会产生 Counter 更新中断)，各影子寄存器的值保持不变，如果 UserGenerate 位置 1，则会重新初始化计数器和预分频器。
[0]	CEN	R/W	0x0	计数器使能位 (Timer Counter Enable) 1：开始计数器 0：关闭计数器

18.5.2.2 TMRx 捕获/比较控制寄存器 (TMRx_CCCR)

- **Name**：TMRx Capture/Compare Control Register(x = 0 ... 7)
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-15]	Reserved	Reserved	Reserved	Reserved
[14-12]	OCM	R/W	0x0	定时器输出比较器模式 (Timer Output Compare Mode) 000：冻结，输出比较寄存器与计数器进行比较不会对输出造成任何影响(保持原有的输出状态) 001：当计数器与捕获/ 比较寄存器匹配后，输出有效电平—参考有效电平极性 010：当计数器与捕获/ 比较寄存器匹配后，输出无效电平—参考有效电平极性 011：当计数器与捕获/ 比较寄存器匹配后，发生翻转 100：强制变为无效电平—参考有效电平极性 101：强制变为有效电平—参考有效电平极性 110：PWM 模式 1—当计数 Counter 值 <= Compare 值时，输出有效状态，否则输出无效状态（有效电平取决于极性寄存器） 111：PWM 模式 2—当计数 Counter 值 <= Compare 值时，输出无效状态，否则输出有效状态（有效电平取决于极性寄存器）
[11]	Reserved	Reserved	Reserved	Reserved
[10]	OCIE	R/W	0x0	比较器中断使能位 (Timer Compare Interrupt Enable) 1：开启中断使能 0：关闭中断使能
[9]	ICOIE	R/W	0x0	定时器重复捕获中断使能控制 (Timer OverCapture Interrupt Enable) 1：开启中断使能 0：关闭中断使能
[8]	ICIE	R/W	0x0	定时器捕获中断使能控制 (Timer Capture Interrupt Enable) 1：开启中断使能 0：关闭中断使能

[7-5]	ICSRC	R/W	0x0	TMRx 的输入捕获源选择 (Timer Input Capture Source Selection) 0x0：TMR0/TMR4 的功能引脚： TMRx(x = 0...3 时)，选择的是 TMR0 的功能引脚； TMRx(x = 4...7 时)，选择的是 TMR4 的功能引脚。 0x1：TMR1/TMR5 的功能引脚： TMRx(x = 0...3 时)，选择的是 TMR1 的功能引脚； TMRx(x = 4...7 时)，选择的是 TMR5 的功能引脚。 0x2：TMR2/TMR6 的功能引脚： TMRx(x = 0...3 时)，选择的是 TMR2 的功能引脚； TMRx(x = 4...7 时)，选择的是 TMR6 的功能引脚。 0x3：TMR3/TMR7 的功能引脚： TMRx(x = 0...3 时)，选择的是 TMR3 的功能引脚； TMRx(x = 4...7 时)，选择的是 TMR7 的功能引脚。 0x4：比较器模块 0 的内部输出 (CMP0 Internal Output) 0x5：比较器模块 1 的内部输出 (CMP1 Internal Output) 0x6：比较器模块 2 的内部输出 (CMP2 Internal Output) 0x7：比较器模块 3 的内部输出 (CMP3 Internal Output) 注意：功能引脚的使用，请参考数据手册中的引脚复用(Pinmux)介绍。
[4]	OCPE	R/W	0x0	输出比较器预加载使能控制 (Timer Output Compare Preload Enable) 0：禁止比较器预加载功能 1：使能比较器预加载功能 更详细介绍请参考“18.4.9.3 捕获/比较事件”章节
[3]	CCS	R/W	0x0	定时器捕获/比较模式选择 (Timer Capture /Compare Select) 0：配置为输出比较模式（Output Compare） 1：配置为输入捕获模式（Input Capture）
[2-1]	CCP	R/W	0x0	捕获/比较极性配置位 (Timer Capture /Compare Polarity) **配置为输出比较时(CCS=0)：** 0：高电平有效 1：低电平有效 **配置为输入捕获时(CCS=1)：** 00：上升沿触发 01：下降沿触发 1x：上升沿和下降沿均触发
[0]	CCE	R/W	0x0	定时器捕获/比较使能控制 (Timer Capture/Compare Enable) **配置为输出比较时(CCS=0)：** 0：关闭输出 - 引脚将处于悬空状态 1：开启输出 - 在相应引脚上输出 Compare 信号 **配置为输入捕获时(CCS=1):** 该位将决定在捕获发生后，计数器值是否能被存储到捕获/比较寄存器内 (TMRx_CCR): 0：禁止捕获 1：使能捕获

18.5.2.3 TMRx 事件生成寄存器 (TMRx_EGR)

- **Name**：TMRx Event Generation Register(x = 0 ... 7)
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-9]	Reserved	Reserved	Reserved	Reserved
[8]	CCG	R/W1C	0x0	定时器捕获/比较事件生成 (Timer Capture/Compare Generation) 此位由软件置 1 以生成一次软件的捕获/比较事件，并由硬件自动清零。 0：不执行任何操作 1：软件重生捕获/比较事件 详细说明请参考“18.4.9.3 捕获/比较事件”章节。
[7-1]	Reserved	Reserved	Reserved	Reserved
[0]	UG	R/W1C	0x0	定时器计数更新事件生成 (Timer Counter Update Generation) 此位由软件置 1 以生成一次更新事件，并由硬件自动清零 0：不执行任何操作 1：软件重生更新事件 详细说明请参考“18.4.9.2 更新事件”章节。

18.5.2.4 TMRx 输入捕获滤波寄存器 (TMRx_ICFR)

- **Name**：TMRx Input Capture Filter Register (x = 0 ... 7)
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-0]	ICF	R/W	0x0	定时器输入捕获滤波器值 (Timer Input Capture Filter value) 输入捕获滤波器，该位域可用户定义外部输入信号的数字滤波带宽。通过设定该位域，可实现每 N 个连续事件才视为一个有效的边沿输出，其中： N = 滤波器值(ICF) + 1 例如：输入捕获模式下，配置极性为上升沿捕获(CCP=00)，滤波器值为 3(ICF=3)，则 N = 3 + 1 = 4，意味着每 4 个连续的上升沿才视为一个有效的上升沿，触发捕获/比较事件。 更详细介绍可参考“18.4.5 输入捕获”和“18.4.9.3 捕获/比较事件”章节。

18.5.2.5 TMRx 中断状态寄存器 (TMRx_ISR)

- **Name**：TMRx Interrupt Status Register(x = 0 ... 7)
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-5]	Reserved	Reserved	Reserved	Reserved
[4]	OVIF	R/W1C	0x0	计数器上溢中断标志位 (Timer Counter Overflow Interrupt Flag) 0：未发生上溢 1：计数器上溢
[3]	ICOIF	R/W1C	0x0	定时器输入捕获重复捕获中断标志位 (Timer Input Capture Over-capture Interrupt Flag) 输入捕获模式下有效(CCS=1)，当捕获标志位(ICIF)已经置 1，再次发生输入捕获事件，则称为重复捕获，重复捕获标志位将被置 1。 0：未检测到重复捕获 1：发生重复捕获 更详细描述请参考“18.4.5 输入捕获”章节。
[2]	ICIF	R/W1C	0x0	定时器输入捕获标志位 (Timer Input Capture Interrupt Flag) 输入捕获模式下有效(CCS=1)： 0：未发生输入捕获 1：发生输入捕获（检测到与极性配置位 CCP 设置匹配的边沿），同时当前计数器的值将会被捕获到捕获/比较寄存器（TMRx_CCR）内。 注意：读取捕获/比较寄存器(TMRx_CCR)同样导致该位自动清 0。 更详细描述请参考“18.4.5 输入捕获”章节。
[1]	OCIF	R/W1C	0x0	输出比较匹配标志位 (Timer Output Compare Interrupt Flag) 输出比较模式下有效(CCS=0)： 0：输出比较未匹配 1：输出比较匹配（当前计数器的值与捕获/比较寄存器 TMRx_CCR 内设定的值相匹配）。 更详细描述请参考“18.4.6 输出比较”章节
[0]	UIF	R/W1C	0x0	定时器计数更新中断标志位 (Timer Counter Update Interrupt Flag) 0：未发生更新事件 1：产生更新事件 注意：使用更新事件还需要将定时器控制寄存器(TMRx_CR)中的禁止更新位(UDIS)置 0，否则更新事件将被屏蔽。 更多详细描述请参考“18.4.9.2 更新事件”章节。

18.5.2.6 TMRx 计数起始值寄存器 (TMRx_CSVR)

- **Name**：TMRx Counter Start Value Register (x = 0 ... 7)
- **Size**：32bits
- **Offset**：0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	CSVR[31:16]	R/W	0x0	计数起始值 (Timer Counter Start Value) 计数起始值寄存器高 16 位，仅 TMRx(x = 4...7)有效。 功能描述同下
[15-0]	CSVR[15:0]	R/W	0x0	计数起始值 (Timer Counter Start Value) 计数起始值寄存器，通过配置此寄存器设置定时器开始计数的值。

18.5.2.7 TMRx 计数终止值寄存器 (TMRx_CEVR)

- **Name**：TMRx Counter End Value Register (x = 0 ... 7)
- **Size**：32bits
- **Offset**：0x24
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	CEVR[31:16]	R/W	0x0	计数终止值 (Timer Counter End Value) 计数终止值寄存器高 16 位，仅 TMRx(x = 4...7)有效。 功能描述同下.
[15-0]	CEVR[15:0]	R/W	0xFFFF	计数终止值 (Timer Counter End Value) 通过配置此寄存器设置定时器结束计数的值。此外，该寄存器支持自动预装载(Auto-Reload)功能： **不使能自动预装载功能(ARPE=0)时**，对该寄存器的设置将立即作用到内部影子寄存器内，本次计数周期内即生效； **使能自动预装载功能(ARPE=1)时**，对该寄存器的设置将在更新事件(UEV)产生后，再更新至内部影子寄存器内生效。 更详细介绍请参考“18.4.1 时基单元”和“18.4.9.2 更新事件”章节。

18.5.2.8 TMRx 捕获/比较寄存器 (TMRx_CCR)

- **Name**：TMRx Capture/Compare Register(x = 0 ... 7)
- **Size**：32bits
- **Offset**：0x28
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	CCR[31:16]	R/W	0x0	定时器捕获/比较值 (Timer Capture/Compare value) 仅 TMRx(x = 4...7)有效。 功能描述同下。
[15-0]	CCR[15:0]	R/W	0x0	定时器捕获/比较值 (Timer Capture/Compare value) 不同的工作模式下该寄存器的功能描述如下： **配置为输出比较时(CCS=0)** 该寄存器写入计数器比较值，用于当前的定时器计数器值(TMRx_CNTR)与该寄存器比较，比较一致时产生捕获/比较事件(CCEV)。在该工作模式下时，该寄存器支持输出比较预装载功能； **不使能输出比较预装载功能时(OCPE=0)**，对该寄存器写入的值将立即生效，作为新的比较值用于输出比较。 **使能输出比较预装载功能时(OCPE=1)**，对该寄存器写入的值将在以下几种事件触发后，才会使新的值真正加载进入内部影子寄存器生效： ● 下一次更新事件(UEV，且仅限于计数器上溢产生的更新事件) ● 之前的比较值所产生的捕获/比较事件(CCEV) ● 软件通过设置 CCG 位产生的捕获/比较事件(CCEV) 更详细描述请参考“18.4.6 输出比较”、“18.4.9.2 更新事件”和“18.4.9.3 捕获/比较事件”等相关章节。 **配置为输入捕获时(CCS=1)** 该寄存器用于当外部输入信号产生与极性(CCP)所设置一样的边沿时，将当前定时器计数器的值捕获到该寄存器内，产生捕获/比较事件(CCEV)。 此外，通过软件设置 CCG 位，可以立即产生捕获/比较事件(CCEV)，而无需外部输入信号产生相应的边沿。 更多描述请参考“18.4.5 输入捕获”和“18.4.9.3 捕获/比较事件”等相关章节。

18.5.2.9 TMRx 预分频寄存器 (TMRx_PSCR)

- **Name**：TMRx Prescaler Register (x = 0...7)
- **Size**：32bits
- **Offset**：0x2C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	PSC [31:16]	R/W	0x0	定时器预分频值 (Timer Prescaler value) 定时器预分频寄存器高 16 位，仅 TMRx(x = 4...7)有效。 功能描述同下。
[15-0]	PSC[15:0]	R/W	0x0	定时器预分频值 (Timer Prescaler value) 定时器计数器的计数频率(CK_CNT)通过此预分频器对时钟源的频率(CK_SRC)进行分频得到： $f_{CK\_CNT} = f_{CK\_SRC} / (PSC + 1)$ 此外，该寄存器支持自动预装载(Auto-Reload)功能： **不使能自动预装载功能(ARPE=0)时**，对该寄存器的设置将立即作用到内部影子寄存器内，本次计数周期内即生效； **使能自动预装载功能(ARPE=1)时**，对该寄存器的设置将在更新事件(UEV)产生后，再更新至内部影子寄存器内生效。 更详细介绍请参考“18.4.1 时基单元”和“18.4.9.2 更新事件”章节。

18.5.2.10 TMRx 计数寄存器 (TMRx_CNTR)

- **Name**：TMRx Counter Register (x = 0...7)
- **Size**：32bits
- **Offset**：0x30
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	CNT[31:16]	R/W	0x0	定时器计数值 (Timer Counter Value) 定时器计数器高 16 位，仅 TMRx(x = 4...7)有效。 功能描述同下。
[15-0]	CNT[15:0]	R/W	0x0	定时器计数值 (Timer Counter Value) 通过该寄存器可对定时器计数器的值进行读取/写入。需注意，因为系统与外设之间存在一定的总线延迟，对该寄存器的读取/写入操作可能存在一定的偏差。

18.5.2.11 TMRx 输出触发事件寄存器 (TMRx_ETER)

- **Name**：TMRx Export Trigger Event Register (x = 0...7)
- **Size**：32bits
- **Offset**：0x34
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-12]	CCTPW	R/W	0x0	捕获/比较事件(CCEV)作为触发信号的输出脉冲宽度 (Timer Capture/Compare event Trigger Pulse Width) 配置该数值用于避免接收触发信号外设的时钟大于定时器的时钟，导致未能正确采样触发信号而导致触发失败。 $t_{PW} = (CCTPW + 1) * t_{CK\_TMRx} = (CCTPW + 1) / f_{CK\_TMRx}$ 0：1 个定时器时钟 1：2 个定时器时钟 ..... 15：16 个定时器时钟 更详细描述，请参看“定时器的对外触发和 PWM 信号输出”章节。
[11-10]	Reserved	Reserved	Reserved	Reserved
[9]	PWMOE	R/W	0x0	PWM 信号输出使能控制 (Timer PWM signal Output Enable) 0：禁止输出 PWM 信号 1：允许输出 PWM 信号 **特别注意：** PWM 信号输出仅当定时器被配置为输出比较模式(CCS=0)，并且输出模式选择为 PWM1/PWM2 时(OCM=110 或 111)有效。 如果配置为捕获模式或其他输出模式，将始终无信号输出。 更详细描述，请参看“定时器的对外触发和 PWM 信号输出”章节。
[8]	CCTE	R/W	0x0	捕获/比较事件(CCEV)触发信号的输出使能控制 (Timer Capture/Compare event Trigger Enable) 0：禁止输出捕获/比较事件(CCEV)触发信号 1：允许输出捕获/比较事件(CCEV)触发信号 更详细描述，请参看“定时器的对外触发和 PWM 信号输出”章节。
[7-4]	UTPW	R/W	0x0	更新事件(UEV)作为触发信号的输出脉冲宽度 (Timer Update event Trigger Pulse Width) 配置该数值用于避免接收触发信号外设的时钟大于定时器的时钟，导致未能正确采样触发信号而导致触发失败。 $t_{PW} = (UTPW + 1) * t_{CK\_TMRx} = (UTPW + 1) / f_{CK\_TMRx}$ 0：1 个定时器时钟 1：2 个定时器时钟 ..... 15：16 个定时器时钟 更详细描述，请参看“定时器的对外触发和 PWM 信号输出”章节。
[3-1]	Reserved	Reserved	Reserved	Reserved

[0]	UTE	R/W	0x0	更新事件(UEV)触发信号的输出使能控制 (Timer Update event Trigger Enable) 0：禁止输出更新事件(UEV)触发信号 1：允许输出更新事件(UEV)触发信号 更详细描述，请参看“定时器的对外触发和 PWM 信号输出”章节。

18.5.2.12 定时器组同步寄存器 (TMRGRPx_SYNCR)

- **Name**：Timer Group x Sync Register (x = 0/1)
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

说明：

TMRGRP0 分组包括 TMR0/1/2/3 四个定时器，TMRGPR1 分组包括 TMR4/5/6/7 四个定时器。对该寄存器相应位置同时写 1，则表明所对应的定时器将会同时启动。

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7]	SYNC3DIS	R/W	0x0	定时器 TMR3 或 TMR7 的同步禁止位 (Timer Group x sync 3 disable) ➢ TMRGRP0 时，配置的是 TMR3 ➢ TMRGRP1 时，配置的是 TMR7
[5]	SYNC2DIS	R/W	0x0	定时器 TMR2 或 TMR6 的同步禁止位 (Timer Group x sync 2 disable) ➢ TMRGRP0 时，配置的是 TMR2 ➢ TMRGRP1 时，配置的是 TMR6
[5]	SYNC1DIS	R/W	0x0	定时器 TMR1 或 TMR5 的同步禁止位 (Timer Group x sync 1 disable) ➢ TMRGRP0 时，配置的是 TMR1 ➢ TMRGRP1 时，配置的是 TMR5
[4]	SYNC0DIS	R/W	0x0	定时器 TMR0 或 TMR4 的同步禁止位 (Timer Group x sync 0 disable) ➢ TMRGRP0 时，配置的是 TMR0 ➢ TMRGRP1 时，配置的是 TMR4
[3]	SYNC3EN	R/W	0x0	定时器 TMR3 或 TMR7 的同步使能位 (Timer Group x sync 3 enable) ➢ TMRGRP0 时，配置的是 TMR3 ➢ TMRGRP1 时，配置的是 TMR7
[2]	SYNC2EN	R/W	0x0	定时器 TMR2 或 TMR6 的同步使能位 (Timer Group x sync 2 enable) ➢ TMRGRP0 时，配置的是 TMR2 ➢ TMRGRP1 时，配置的是 TMR6

[1]	SYNC1EN	R/W	0x0	定时器 TMR1 或 TMR5 的同步使能位 (Timer Group x sync 1 enable) ➢ TMRGRP0 时，配置的是 TMR1 ➢ TMRGRP1 时，配置的是 TMR5
[0]	SYNC0EN	R/W	0x0	定时器 TMR0 或 TMR4 的同步使能位 (Timer Group x sync 0 enable) ➢ TMRGRP0 时，配置的是 TMR0 ➢ TMRGRP1 时，配置的是 TMR4

19 高精度脉宽调制器（HRPWM）

19.1 简介

高分辨率脉宽调制器（HRPWM）可生成多达 12 路高精度定时的数字信号，主要用于驱动开关模式电源或照明系统等电源转换系统，但也可用于，一般对时间分辨率有极高要求的应用。

该调制器采用模块化架构，可生成独立波形或耦合波形。波形由独立式定时信号（使用计数器和比较单元）以及多种外部事件（如模拟或数字反馈以及同步信号）确定，因此可生成大量不同的控制信号（PWM、相移、恒定 Ton…），从而满足大部分转换拓扑的需求。

为实现控制和监测用途，该调制器还具有事件触发功能，并连接到内置的 ADC 和 DAC 转换器。此外，该调制器能够处理各种故障机制，从而实现安全关断。

19.2 结构框图

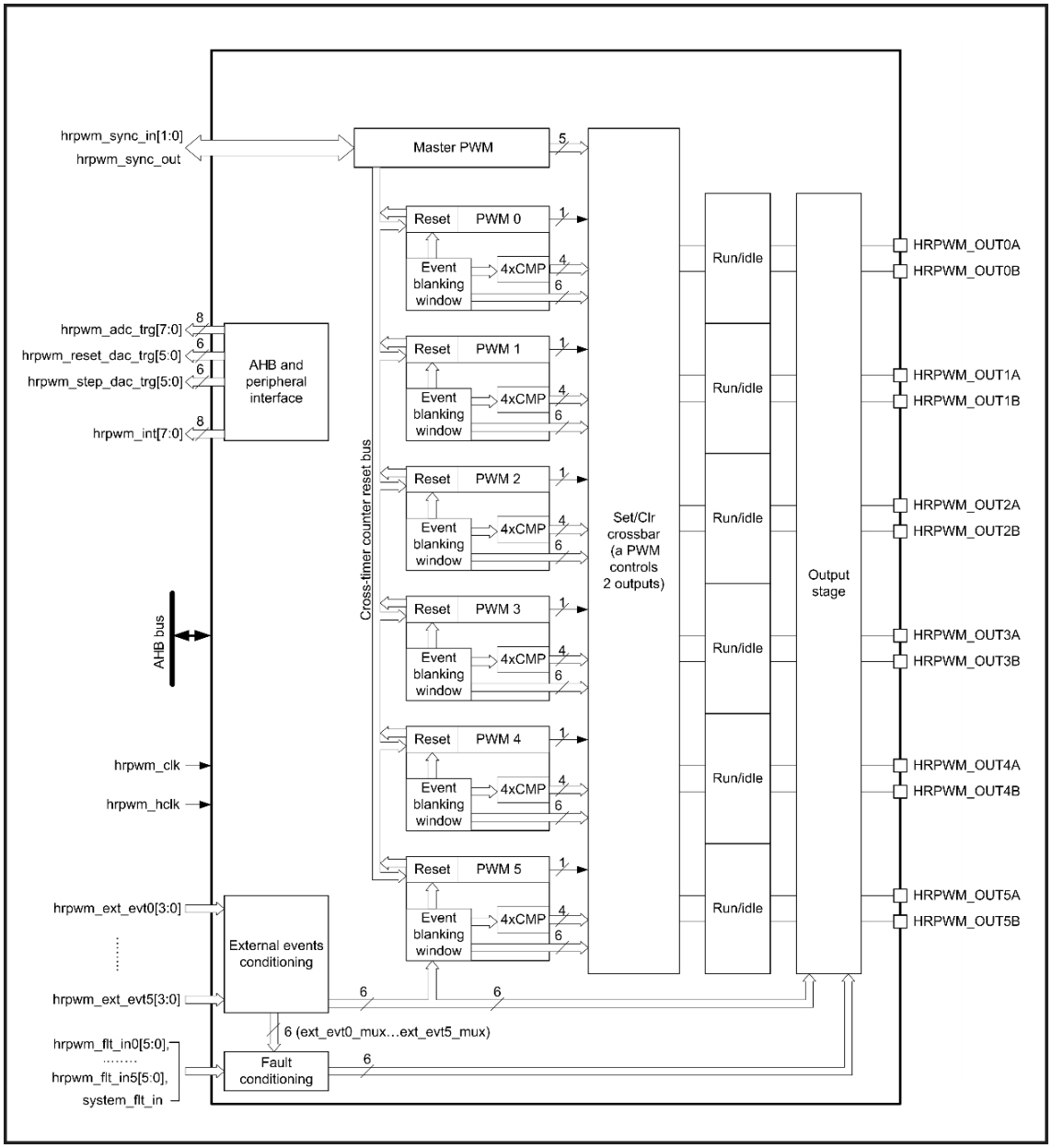


图 19-1 HRPWM 结构框图

19.3 主要特性

- 多个定时单元
  - 195 ps 分辨率，已针对电压和温度变化进行补偿
  - 所有输出均支持高分辨率，可在触发单脉冲模式下调整占空比，频率和脉宽
  - 6 个 16 位定时单元（每个定时单元包含一个独立计数器和 4 个比较单元）
  - 12 路输出可通过定时单元控制，每条通道多达 18 个置位/复位源
  - 模块化结构可满足多种配有 1 或 2 个开关的独立转换器的需求，也可满足少数大型多开关拓扑的需求
- 多达 6 个外部事件，可用于任何定时单元
  - 可编程极性和有效边沿
  - 快速异步模式
  - 可编程数字滤波器
  - 使用消隐和加窗模式实现伪事件过滤
- 多条通道可连接到内置模拟外设
  - 8 个连接到 ADC 转换器的触发事件
  - 8 个连接到 DAC 转换器（三角波补偿）的触发事件
  - 12 个连接到 DAC 转换器（锯齿波补偿）的触发事件（复位和步进）
- 全面的保护机制
  - 6 个故障输入可组合使用并关联到任何定时单元
  - 可编程极性和有效边沿
  - 可编程数字滤波器
- 多个 HRPWM 单元可与外部同步输入/输出同步
- 多功能输出级
  - 高分辨率的死区插入
  - 可编程输出极性
  - 斩波模式
- 8 个中断向量，每个向量最多具有 12 个源

19.4 功能描述

19.4.1 常规功能说明

HRPWM 可分为以下几个子实体：

- 主定时器（Master PWM）
- 定时单元（PWM0 至 PWM5）
- 输出级
- 外部事件和故障信号调节逻辑，由所有定时器共用
- 系统接口

主定时器基于一个 16 位计数器。它可以通过 4 个比较值来置位/复位 12 路输出中的任何一个，并向 6 个定时单元提供同步信号。其主要用途是使各个定时器单元受唯一的时钟源控制。交错降压转换器是一个典型的应用示例，主定时器在其中管理着多个定时单元的相移。

定时器单元既可以独立工作，也可以与其他定时器（包括主定时器）配合工作。每个定时器都可控制两路输出。输出置位/复位事件可以由定时单元的比较寄存器触发，或者由来自主定时器的事件或外部事件触发。

输出级有多种用途：

- 在互补输出模式下为两路输出添加死区
- 将载波频率添加在调制信号上
- 通过将输出异步置为预定义的安全电平来管理故障事件

外部事件和外部故障调节逻辑包括：

- 输入选择 MUX（例如，为给定外部事件通道选择数字输入或片上时钟源）
- 极性和有效边沿编程
- 数字滤波和边沿计数

系统接口允许 HRPWM 与 MCU 的其余部分进行交互：

- 向 CPU 发出中断请求
- 触发 ADC 和 DAC 转换器
- 触发 DAC 转换器进行斜坡补偿

HRPWM 寄存器分为 8 组：

- 主定时器寄存器
- 定时器 0 至定时器 5 寄存器
- 通用寄存器，用于所有定时器单元共用的功能

注意：根据文档编写约定，在文本和寄存器中对 6 个定时单元的引用统一用“x”字母表示（x 可以是 0 到 5 的任意值）。

19.4.2 HRPWM 引脚和内部信号

片上和片外的 HRPWM 输入和输出总结在本节的表 19-1 中。

表 19-1 HRPWM 输入/输出总结

信号名称	信号类型	信号描述
HRPWM_OUT0A HRPWM_OUT0B HRPWM_OUT1A HRPWM_OUT1B HRPWM_OUT2A HRPWM_OUT2B HRPWM_OUT3A HRPWM_OUT3B HRPWM_OUT4A HRPWM_OUT4B HRPWM_OUT5A HRPWM_OUT5B	Outputs	外部输出，总共支持 6 对外部输出，每个输出可以独立工作，也可以通过插入死区（HRPWM_OUTxA 和 HRPWM_OUTxB）来成对工作，这些输出都会连接到片外。
hrpwm_flt_in1[3:0] hrpwm_flt_in2[3:0] hrpwm_flt_in3[3:0] hrpwm_flt_in4[3:0] hrpwm_flt_in5[3:0] hrpwm_flt_in6[3:0]	Digital input	故障事件，总共支持 6 组故障事件，每组故障事件可以从 4 个来源选择，这些来源可以是外部事件，可以是片内比较器输出、也可以是 HRPWM_FLTx 输入引脚的输入，故障事件置位后立即停止 HRPWM 输出并切换为安全电平
system_flt_in	Digital input	系统故障主要指 MCU 内部故障事件，包括 XOSC 时钟丢失，CPU 锁定，LVD 输出等事件。
hrpwm_sync_in[1:0]	Digital Input	同步输入，用于将整个 HRPWM 与其他内部或外部计时器资源同步： hrpwm_sync_in[0]：来源是 TMR0_TRGO 输出 hrpwm_sync_in[1]：来源是 HRPWM_SCIN 输入引脚
hrpwm_sync_out	Digital output	同步输出，用于级联或同步片上或片外的多个 HRPWM 实例： hrpwm_sync_out：可以通过 HRPWM_SCOUT 输出引脚连接到片外 HRPWM 或其他外设
hrpwm_ext_evt0[3:0] hrpwm_ext_evt1[3:0] hrpwm_ext_evt2[3:0] hrpwm_ext_evt3[3:0] hrpwm_ext_evt4[3:0] hrpwm_ext_evt5[3:0]	Digital input	外部事件，总共支持 6 组外部事件，每组外部事件可以从 4 个来源中选择 1 个，来源可以是片内其他外设事件，如比较器输出、ADC 模拟看门狗事件、TIMx 定时器触发事件，也可以是片外事件，如 HRPWM_EVTx 引脚的输入

hrpwm_adc_trg0 hrpwm_adc_trg1 hrpwm_adc_trg2 hrpwm_adc_trg3 hrpwm_adc_trg4 hrpwm_adc_trg5 hrpwm_adc_trg6 hrpwm_adc_trg7	Digital output	ADC 触发事件，主要用于 ADC 的转换开启以及 DAC 的三角波生成。后面 hrpwm_adc_trg0 至 hrpwm_adc_trg7 简称为 ADC 事件 0 至 ADC 事件 7
hrpwm_reset_dac_trg0 hrpwm_reset_dac_trg1 hrpwm_reset_dac_trg2 hrpwm_reset_dac_trg3 hrpwm_reset_dac_trg4 hrpwm_reset_dac_trg5	Digital output	双通道 DAC 触发事件，主要用于 DAC 锯齿波的复位
hrpwm_step_dac_trg0 hrpwm_step_dac_trg1 hrpwm_step_dac_trg2 hrpwm_step_dac_trg3 hrpwm_step_dac_trg4 hrpwm_step_dac_trg5	Digital output	双通道 DAC 触发事件，主要用于 DAC 锯齿波的步进
hrpwm_mst_int hrpwm_slv0_int hrpwm_slv1_int hrpwm_slv2_int hrpwm_slv3_int hrpwm_slv4_int hrpwm_slv5_int hrpwm_flt_int	Digital output	中断请求，每个中断请求对应一个中断入口地址
hrpwm_hclk	-	总线时钟（来自 AHB2 时钟）
hrpwm_clk	-	功能时钟（来自 HSI/HSE/PLL0/PLL1 时钟）

表 19-2 HRPWM 与 ADC 事件连接

ADC 触发事件	ADC0	ADC1
hrpwm_adc_trg0	Yes	Yes
hrpwm_adc_trg1	Yes	Yes
hrpwm_adc_trg2	Yes	Yes
hrpwm_adc_trg3	Yes	Yes
hrpwm_adc_trg4	Yes	Yes
hrpwm_adc_trg5	Yes	Yes
hrpwm_adc_trg6	Yes	Yes
hrpwm_adc_trg7	Yes	Yes

表 19-3 HRPWM 与 DAC 事件连接

DAC 触发事件	DAC0	DAC1	DAC2	DAC3
hrpwm_adc_trg0	Yes	No	No	No
hrpwm_adc_trg1	No	Yes	No	No
hrpwm_adc_trg2	No	No	Yes	No
hrpwm_adc_trg3	No	No	No	Yes
hrpwm_adc_trg4	Yes	No	No	No
hrpwm_adc_trg5	No	Yes	No	No
hrpwm_adc_trg6	No	No	Yes	No
hrpwm_adc_trg7	No	No	No	Yes
hrpwm_reset_dac_trg0 hrpwm_step_dac_trg0	Yes	Yes	Yes	Yes
hrpwm_reset_dac_trg1 hrpwm_step_dac_trg1	Yes	Yes	Yes	Yes
hrpwm_reset_dac_trg2 hrpwm_step_dac_trg2	Yes	Yes	Yes	Yes
hrpwm_reset_dac_trg3 hrpwm_step_dac_trg3	Yes	Yes	Yes	Yes
hrpwm_reset_dac_trg4 hrpwm_step_dac_trg4	Yes	Yes	Yes	Yes
hrpwm_reset_dac_trg5 hrpwm_step_dac_trg5	Yes	Yes	Yes	Yes

19.4.3 时钟

HRPWM 必须由 PLL1/PLL2 提供时钟才能实现高分辨率。f_{HRPWM} 时钟周期被均匀分为多达 32 个中间步长，通过边沿定位逻辑实现。HRPWM 中的所有时钟均由该参考时钟生成。

术语定义（Definition of terms）

f_{HRPWM} ： 主 HRPWM 时钟（hrpwm_clk）。所有后续时钟均由该时钟源生成，并与该时钟源同步。

f_{HRCK} ： 高分辨率等效时钟。
考虑 f_{HRCK} 周期为 f_{HRPWM} 时钟周期除以 32，f_{HRCK} 等效频率为 160 x 32 = 5.12 GHz。

f_{DTG} ： 死区发生器时钟。

f_{CHPFRQ} ： 斩波级时钟源。

f_{1STPW} ： 定义斩波模式下初始脉冲长度的时钟源。

$f_{SAMPLING}$ ： 采样故障输入或外部事件输入所需的时钟。

f_{FLTS} ： 派生自 f_{HRPWM} 的时钟，用于对故障事件进行过滤。

f_{EEVS} ： 派生自 f_{HRPWM} 的时钟，用于对外部事件进行过滤。

定时器时钟和预分频（Timer clock and prescaler）

HRPWM 中的每个定时器都有独立的时钟预分频，以供用户调整定时器分辨率，如表 19-4 所示。

表 19-4 定时器分辨率和最低 PWM 频率（f_{HRPWM}=160MHz）

CKPSC[2:0]	预分频比例	f_{HRCK} 等效频率	分辨率	最低 PWM 频率
000	1	160 x 32 MHz = 5.12 GHz	195 ps	78.1 kHz
001	2	160 x 16 MHz = 2.56 GHz	390 ps	39.1 kHz
010	4	160 x 8 MHz = 1.28 GHz	781 ps	19.5 kHz
011	8	160 x 4 MHz = 640 MHz	1.56 ns	9.76 kHz
100	16	160 x 2 MHz = 320 MHz	3.12 ns	4.88 kHz
101	32	160 MHz	6.25 ns	2.44 kHz
110	64	160/2 MHz = 80 MHz	12.5 ns	1.22 kHz
111	128	160/4 MHz = 40 MHz	25 ns	0.61 kHz

高分辨率可用于边沿定位，PWM 周期调整以及外部触发的脉冲持续时间。

高分辨率不适用于以下功能：

- 定时器计数值的读写访问

对于时钟预分频比低于 32（CKPSC [2:0] <5）的情形，计数器的最低有效位无意义。最低有效位无法被写入，读取时返回 0。

例如，如果 CKPSC [2:0] = 2（预分频比例为 4），写入 0xFFFF 到计数值寄存器产生的有效值为 0xFFF8。相反地，介于 0xFFFF 和 0xFFF8 之间的任何计数值都读取为 0xFFF8。

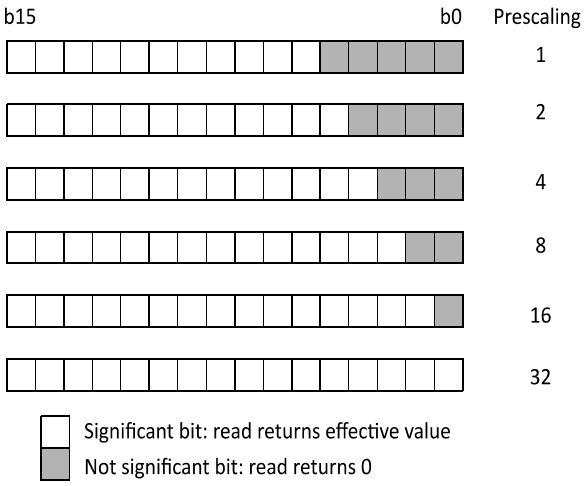


图 19-2 计数寄存器格式与时钟预分频系数

初始化（Initialization）

启动时，必须先初始化预分频位，然后再写入比较值和周期值。定时器使能后（MCR 寄存器中的 MCEN 或 CENx 位置 1），不能修改预分频。

警告：仅当计数器和输出行为与其他定时器的信息和信号无关时，主定时器和定时器 0~5 中

才可以配置不同的预分频比。如果以下某个事件从一个定时单元（或主定时器）传播到另一个定时单元，必须在这些定时单元中配置相同的预分频比：输出置位/复位事件，计数器复位事件，更新事件。预分频系数不相等会导致结果无法预测。

死区发生器时钟（Deadtime generator clock）

死区时间预分频器由（f_{HRPWM} x 8）/ $2^{(CKPSC[2:0])}$提供，通过 PWMxCR0 寄存器中的 CKPSC [2:0]位进行编程。

对于 f_{HRPWM} = 160 MHz，t_{DTG} 的范围为 781 ps 至 100 ns。

斩波级时钟（Chopper stage clock）

斩波级时钟源 f_{CHPFRQ} 由 f_{HRPWM} 生成，使用 16 到 256 的分频系数，对于 f_{HRPWM} = 160 MHz，f_{CHPFRQ} 的范围为 625 kHz≤f_{CHPFRQ}≤10 MHz。

t_{1STPW} 是斩波模式下的初始脉冲长度，通过 CHPxR 寄存器中的 STRPW[3:0]位进行配置，具体计算公式如下：

$$t_{1STPW} = （STRPW [3：0] +1）x 16 x t_{HRPWM}。$$

计算时使用 f_{HRPWM} / 16 作为时钟源（对于 f_{HRPWM} = 160 MHz 为 10 MHz）。

故障输入采样时钟（Fault input sampling clock）

故障输入噪声抑制滤波器的时间常数由 $f_{SAMPLING}$ 定义，可以为 f_{HRPWM} 或 f_{FLTS}。

f_{FLTS} 生成自 f_{HRPWM}，对于 f_{HRPWM} = 160 MHz，f_{FLTS} 的范围为 160 MHz 至 20 MHz。

外部事件输入采样时钟（External event input sampling clock）

故障输入噪声抑制滤波器的时间常数由 $f_{SAMPLING}$ 定义，可以为 f_{HRPWM} 或 f_{EEVS}。

f_{EEVS} 生成自 f_{HRPWM}，对于 f_{HRPWM} = 160 MHz，f_{EEVS} 的范围为 160 MHz 至 20 MHz。

19.4.4 定时器 0~5 定时单元

HRPWM 嵌入了 6 路完全相同的定时单元，这些定时单元由采用自动重载机制的 16 位递增计数器组成，用于定义计数周期和 4 个比较单元，如图 19-3 所示。每路单元都包含对 2 路输出的所有控制功能，因此可作为独立定时器工作。

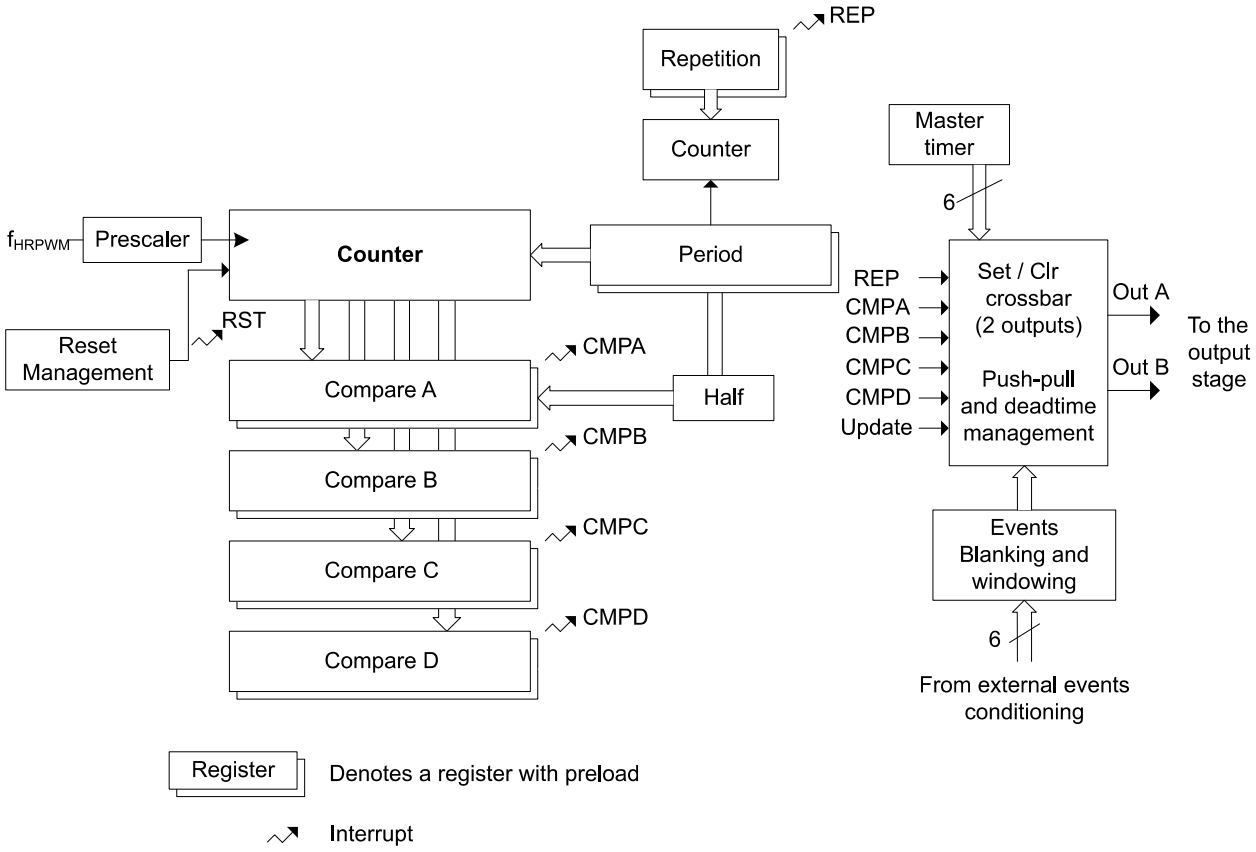


图 19-3 定时器 0~5 概览

周期值和比较值必须在指定的上下限范围内，上下限范围与高分辨率实现相关，具体数值如表 19-5 所示：

- 最小值必须大于或等于 3 个 f_{HRPWM} 时钟周期。
- 最大值必须小于或等于 0xFFFF - 1 个 f_{HRPWM} 时钟周期。

表 19-5 周期值和比较值寄存器的最小值和最大值

CKPSC[2:0] value	Min	Max
0	0x0060	0xFFDF
1	0x0030	0xFFEF
2	0x0018	0xFFF7
3	0x000C	0xFFFB
4	0x0006	0xFFFD
≥ 5	0x0003	0xFFFD

注：如果比较值大于周期值，则不会生成比较匹配事件。

计数器工作模式（Counter operating mode）

定时器 0~5 可在连续模式（自由运行）下工作，也可以单次模式工作，此时会由复位事件触发开始计数，工作模式通过 PWMxCR0 控制寄存器中的 CONT 位设置。另外一个 RETRG 位可用于选择单次模式时可重触发的还是不可重触发的。表 19-6 和图 19-4 以及图 19-5 总结了工作模式的详细信息。

表 19-6 定时器工作模式

CONT	RETRG	工作模式	启动/停止条件&计时/事件生成
0	0	单次不可重触发	将 CENx 位置 1 会使能定时器，但不会启动计数。第一个复位事件启动计数，随后直到计数达到 PER 值为止的任何复位事件都将被忽略。然后计数停止并生成 PER 事件。复位事件重新从 0x0000 开始启动计数。
0	1	单次可重触发	将 CENx 位置 1 会使能定时器，但不会启动计数。如果计数没有启动，则复位事件将启动计数，否则复位事件将计数清零。当计数达到 PER 值时计数停止并生成 PER 事件。复位事件重新从 0x0000 开始启动计数。
1	X	连续模式	将 CENx 位置 1 会使能定时器，并同时启动计数。当计数达到 PER 值时，它将翻转到 0x0000 并恢复计数。任何时刻都可以复位计数。

可以随时清零 CENx 位，以禁止定时器并停止计数。

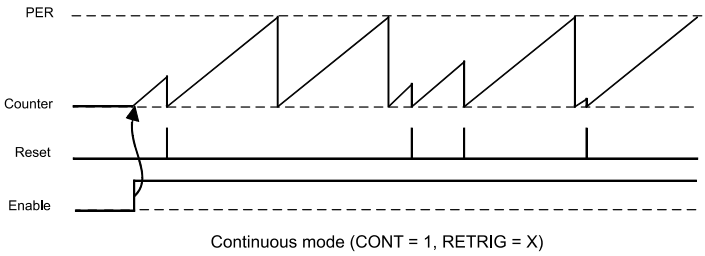


图 19-4 定时器连续工作模式

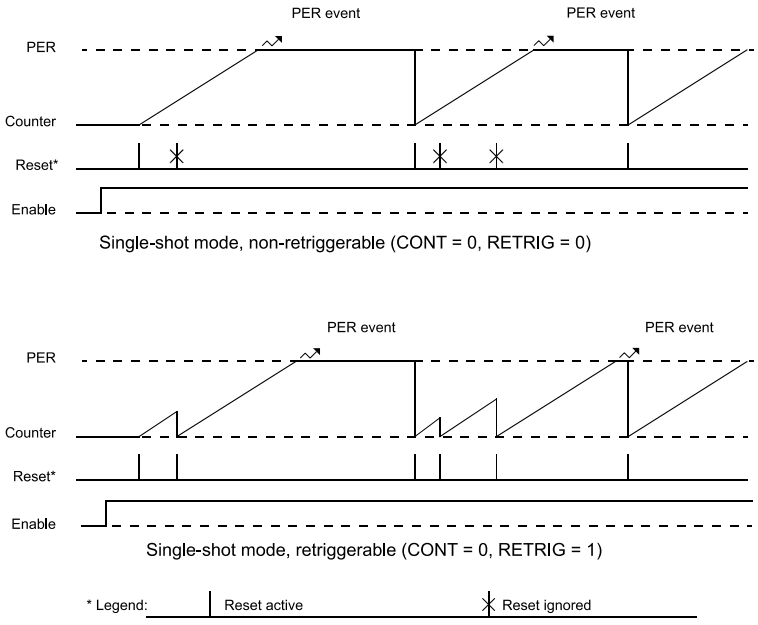


图 19-5 PWM 单次模式

翻转事件（Roll-over event）

在连续模式下，如果计数器在达到 PERxR 寄存器中设置的周期值后恢复为 0，则会生成计数器 roll-over 翻转事件。

该事件在 HRPWM 中有多种用途：

- 置位/复位输出
- 触发寄存器内容更新（从预装载寄存器加载到活动寄存器）
- 产生中断请求
- 产生 ADC 触发事件
- 使重复计数器递减

如果初始计数器值大于定时器启动时的周期值，或者在计数器已超过该值时设置了新周期值，计数器在连续模式下会复位、并且重新开始计数，在单次模式下会继续增加，将在达到最大周期值时溢出。

定时器复位（Timer reset）

定时单元计数器的复位可通过多达 29 个事件触发，这些事件可同时在 RSTxR 寄存器中选择，具体包括以下复位源：

- 定时单元：CMPB，CMPD 和寄存器更新（3 个事件）
- 主定时器：计数器周期事件，CMPA~CMPD（5 个事件）
- 外部事件：EXTEVNT0~5（6 个事件）
- 所有其他定时单元：CMPA、CMPB 和 CMPD（15 个事件）

可同时选择多个事件作为复位源。在这种情况下，会对多个复位请求进行或运算。如果在同一 f_{HRPWM} 时钟周期内生成 2 个计数器复位事件，则会考虑后一个定时器复位事件。

此外，还可以使用 CR2 寄存器中的 RSTx 位对计数器执行软件复位。这些控制位分组到一个寄存器中，从而可同时复位多个计数器。

仅当相关计数器已使能（CENx 位置 1）后，才会考虑复位请求。

如果 f_{HRPWM} 时钟预分频比大于 32 时，计数器复位事件会延迟到预分频时钟的下一有效边沿，这样可确保在输出跳变同步到复位事件（通常是恒定 Ton 时间转换器）时生成的波形无抖动。

图 19-6 显示了时钟预分频比为 128（f_{HRPWM} 除以 4）时的复位处理方式。

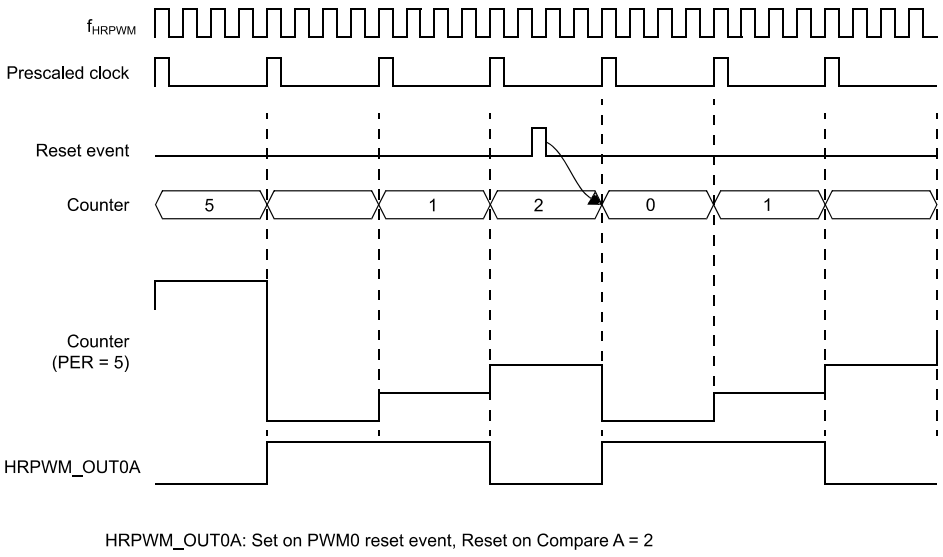


图 19-6 定时器复位重新同步（预分频比大于 32）

重复计数器（Repetition counter）

重复计数器的主要用途是通过分离开关频率和中断频率，来调整周期中断频率并减轻 CPU 的负荷。

定时单元包含重复计数器。该计数器无法读取，只能使用 REPxR 寄存器进行编程，计数器可以自动重载。

定时器使能后（CENx 位置 1），重复计数器会初始化为 REPxR 寄存器的内容。定时器使能后，每次计数器由于复位事件或计数器翻转而清零时，重复计数器都会减 1。当重复计数达到 0 后，会发出 REP 中断请求（若使能 IER 寄存器中的 REPIE 位）。

如果 REPxR 寄存器设为 0，会在每个周期都产生中断。如果值大于 0，则会在（REPxR + 1）个周期后产生 REP 中断。图 19-7 显示了连续模式下重复计数器取不同值时的操作。

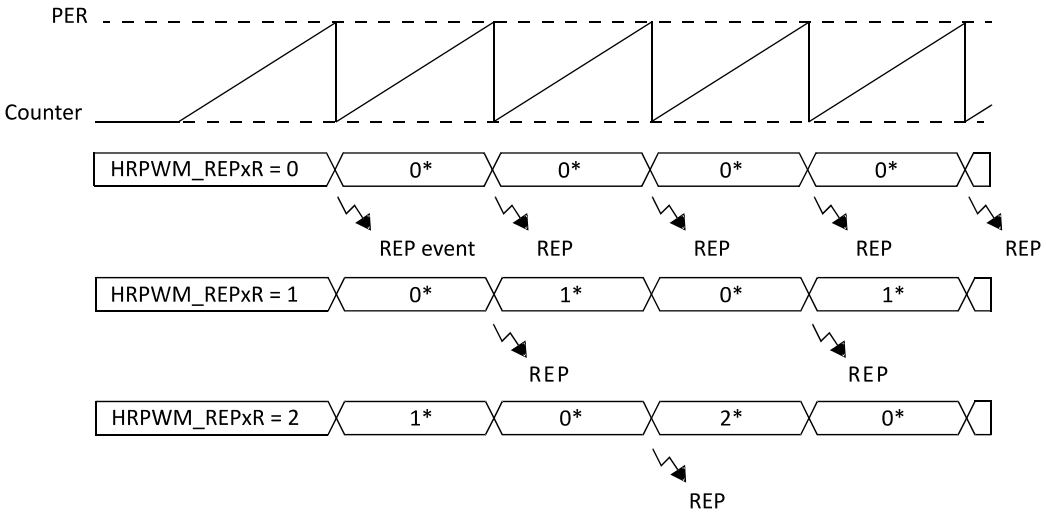


图 19-7 连续模式下的重复事件

无论在连续模式还是单次模式下，如果计数器在达到周期值（可变频率操作）之前复位，则也可使用重复计数器（如图 19-8 所示）。复位会使重复计数器在计数器使能后（CENx 位置 1）执行第一次启动时递减。

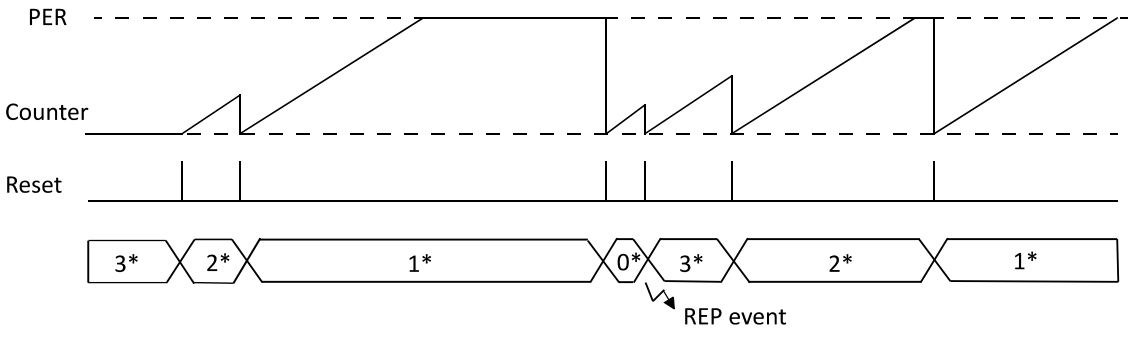


图 19-8 单次模式下的周期重复事件

来自 hrpwm_sync_in 源的复位或启动事件会像其他任何复位事件一样使重复计数器递减。但在通过 SYNCIN 启动的单次模式下（PWMxCR0 寄存器中的 SYNCSTRTx 位置 1），重复计数器仅会在周期后出现第一个复位事件时递减。任何后续的复位事件都不会更改重复计数器的值，直至计数器通过新的 hrpwm_sync_in 输入请求重启。

置位/复位交错配置矩阵（Set / Clear crossbar）

置位事件相当于输出从无效状态跳变为有效状态，而复位事件相当于输出从有效状态转换到无效状态。

波形的极性在输出级中定义，以适应正逻辑或负逻辑外部组件：对于正极性（POLx = 0），有效电平对应于逻辑电平 1，而对于负极性（POLx = 1），有效电平对应于逻辑电平 0。

每个定时单元都会控制两路输出的置位/复位交错配置矩阵，这两路输出可通过多达 19 种事件置位、复位或翻转，这些事件可从以下源中选择：

- 定时单元：周期，CMPA~CMPD，寄存器更新（6 个事件）
- 主定时器：周期，CMPA~CMPD，HRPWM 同步（6 个事件）
- 外部事件：EXTEVNT0~5（6 个事件）
- 软件强制（1 个事件）

注意：在 Updown 模式（UDM 位设置为 1）中，计数周期事件是根据 OUTROM [1：0]位的设置生成的。

事件源会进行或运算，可同时选择多个事件。

每路输出均由两个 32 位寄存器控制，一个寄存器用于输出置位(SETxyR)，一个寄存器用于输出复位(CLRxyR)，其中 x 表示定时单元 0~5，y 表示输出 A 或 B（例如 SET1AR，CLR1AR ...）。

如果为置位和复位选择了相同事件，则会翻转输出状态。每个 t_{HRPWM} 周期输出状态的翻转次数不能超过 1 次：如果同一周期中有两个连续的翻转事件，则仅会考虑第一个翻转事件。

仅当计数器使能后（CENx 位置 1），才会考虑置位和复位请求，但软件在定时器启动时强制请求允许预置输出的情况除外。

使用两个比较事件生成 PWM 波形的示例如图 19-9 所示。

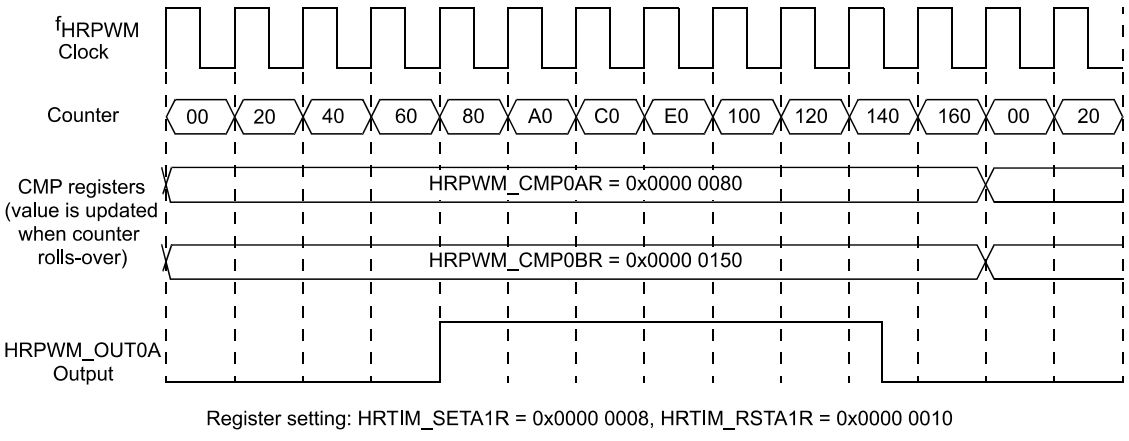


图 19-9 输出因比较事件而动作：CMPA 置位，CMPB 复位

更新事件置位/复位输出（Set / reset on update events）

更新事件可以将输出置位/复位，不过置位/复位事件是在低分辨率下完成的。当 CKPSC [2：0] <5 时，高分辨率延迟被设置为其最大值，因此与其他比较置位/复位事件相比，更新时的置位/复位事件总是滞后，抖动在 0 到 31/32 个 f_{HRPWM} 时钟周期的时间之间变化。

半模式（Half mode）

此模式用于生成占空比固定 50%、频率可变的方波信号（通常用于使用谐振拓扑的转换器）。允许在设定新周期值时自动将占空比强制设为周期值的一半。

要使能此模式，应向 PWMxCR0 寄存器中的 HALF 位写入 1。PERxR 寄存器写入数值后，会自动将 CMPA 值更新为 PERxR / 2 值。

生成方波的输出必须编程为在发生比较 A 事件时进行一次跳变，在发生周期事件进行一次跳变，具体如下：

- HRPWM_SETxyR = 0x0000 0008，HRPWM_CLRxyR = 0x0000 0004
- HRPWM_SETxyR = 0x0000 0004，HRPWM_CLRxyR = 0x0000 0008

半模式会覆盖 CMPAxR 寄存器的内容。访问 PERxR 寄存器不仅会更新比较 A 内部寄存器。用户可访问的 CMPAxR 寄存器也会更新为 PERxR / 2 值。

当使能预装载功能（PREEN = 1）时，则会在发生更新事件时刷新 CMPA 活动寄存器。如果禁止预装载功能（PREEN = 0），则在 PERxR 写入数值后，CMPA 活动寄存器会立即更新。

如果使能了半模式，周期必须大于或等于 6 个 f_{HRPWM} 时钟周期（如果 CKPSC [2:0] = 0，则为 0xC0；如果 CKPSC [2:0] = 1，则为 0x60；如果 CKPSC [2:0] = 2，则为 0x30...）。

交错模式（Interleaved mode）

交错模式补充完善了半模式，可以帮助实现一些交错的拓扑。

当 PERxR 寄存器更新时，会自动重新计算各个比较寄存器的内容。

使用 MCR 和 PWMxCR0 寄存器中的 HALF 位和 INTLVD[1:0]位使能交错模式，如下表 19-7 所示。

表 19-7 交错模式选择

HALF 位	INTLVD [1:0] 位	交错模式
0	00	无影响
1	00	双相交错 180°
0	01	三相交错 120°
0	10	四相交错 90°

表 19-8 给出了三种交错模式下的比较值。交错模式下相应比较寄存器的内容会被覆盖。相应的比较事件可用于触发输出置位/复位或复位定时器。

表 19-8 交错模式下的 CMPxAR~CMPxCR 值

模式	双相交错 180°	三相交错 120°	四相交错 90°
CMPxAR	PERxR/2	PERxR/3	PERxR/4
CMPxBR	无影响	2x (PERxR/3)	PERxR/2
CMPxCR	无影响	无影响	3x (PERxR/4)

注：在半模式和交错模式下，比较值寄存器由硬件控制，寄存器写入无效。不过写入寄存器的值存储在预加载寄存器中，在退出这些模式后下个更新事件后生效。

交换模式（Swap mode）

交换模式允许配置一个寄存器位交换两个输出：输出 A 信号连接到输出 B 引脚，输出 B 信号连接到输出 A 引脚。输出交换由 CR2 寄存器中的 SWPx 位触发，并在下一个更新事件处生效。

输出在进入置位/复位交错配置矩阵之前已经发生交换，具体如下：

- 如果 SWPx = 0，SETxAR 和 CLRxAR 编码控制输出 A，SETxBR 和 CLRxBR 编码控制输出 B
- 如果 SWPx = 1，SETxAR 和 CLRxAR 编码控制输出 B，SETxBR 和 CLRxBR 编码控制输出 A

交换模式只影响预装载寄存器，而不影响活动寄存器。

注：使用交换模式时，必须使能预加载模式。

交换模式不会修改 SETxy 和 RSTxy 状态标志，以及相应的中断请求。例如，当 SWP = 0 时，SETxA 标志与输出 A 相关；当 SWPx = 1 时，SETxA 标志与输出 B 相关。

类似地，交换模式不会更改 OUTxR 寄存器中的控制位（CHPx，FAULTx[1:0]，IDLESx，POLx 位）的属性。例如，无论 SWP 位的值如何，POLA 位都控制输出 A 的极性。

注：在推挽模式下（PWMxCR0 寄存器中的 PSHPLL = 1），SWPx 位将被忽略。

推挽模式（Push-pull mode）

该模式的主要目的是使用推挽拓扑驱动转换器。

通过将 PWMxCR0 寄存器中的 PSHPLL 位置 1，即可使能推挽模式。

在该模式下，会按照周期将交错配置矩阵生成的信号交替地施加到输出 A 和输出 B 上（如果信号施加到输出 A 上，则输出 B 保持在无效状态，反之亦然）。重定向速率（推挽频率）由定时器的周期事件定义，如图 19-10 所示。推挽周期是定时器计数周期的两倍。

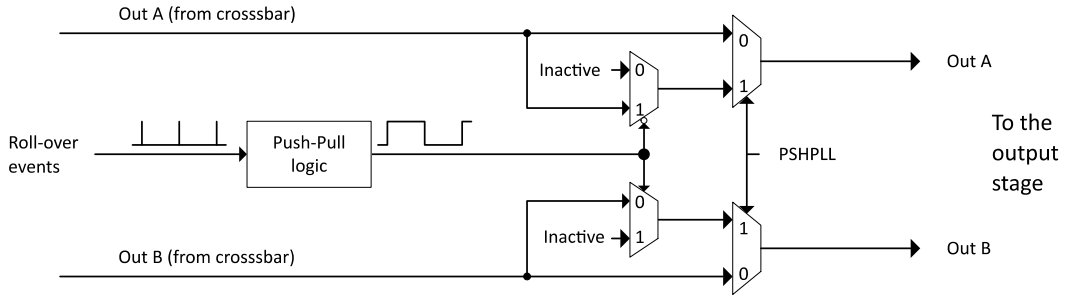


图 19-10 推挽模式框图

定时器在连续模式和单次模式（可重触发/不可重触发）下工作时，可以使用推挽模式；需要禁止定时器以停止推挽操作，并且重新使能推挽操作之前需要将计数器复位。

两路输出的信号波形由 SETxyR 和 CLRxyR 决定。如果希望两路输出的波形完全相同，并实现平衡的操作，需要配置 SETxAR = SETxBR 和 CLRxAR = CLRxBR。不过，仍可对两路输出进行不同的编程，以实现其他用途。

在图 19-11 中提供的示例中，定时器内部波形的定义如下：

- 发生周期事件时，输出置位
- 发生 CMPA 匹配事件时，输出复位

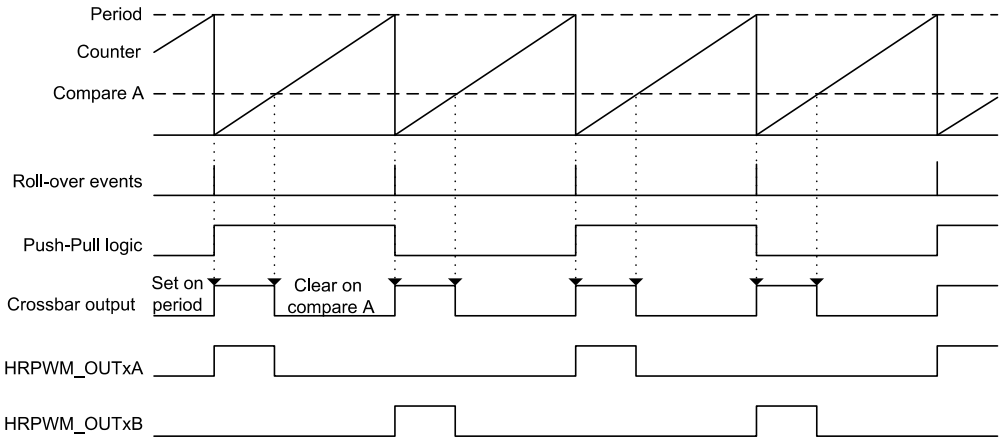


图 19-11 推挽模式示例

推挽模式下的死区插入机制如图 19-12 所示，这里包含了正向死区和负向死区。在这种情况下，输出不再是互补的，两路输出会独立地插入死区时间（交错配置矩阵的 OUTA 和 OUTB 都有效）。

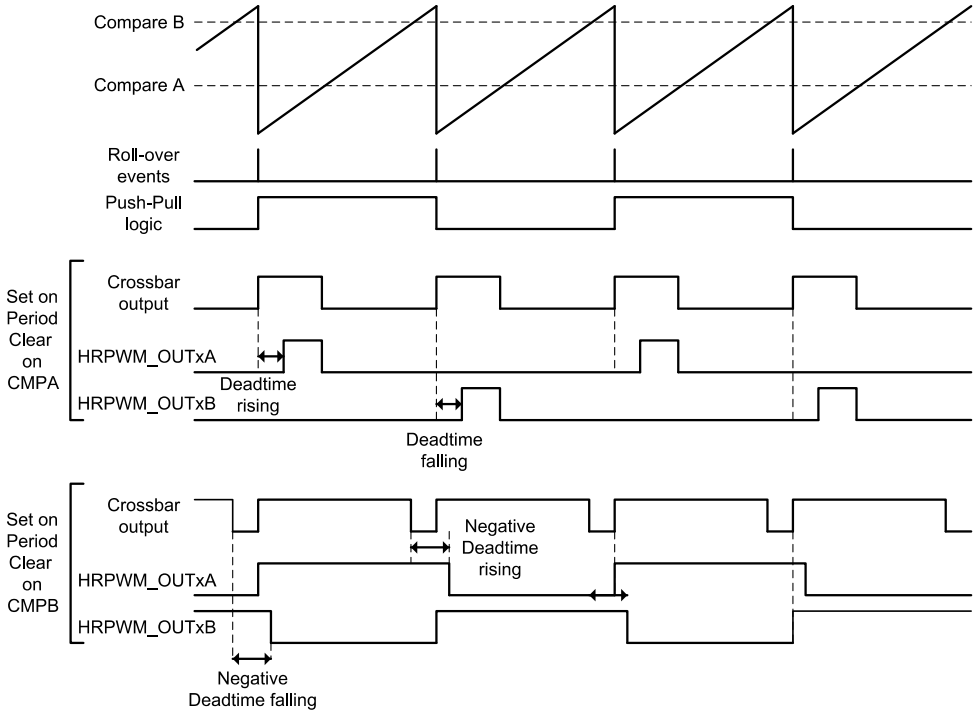


图 19-12 带有死区时间的 PWM 推挽模式

死区（Deadtime）

死区插入单元可通过单个参考波形生成一对互补信号，并且有效状态跳变之间的延迟可编程。这通常用于使用半桥或全桥的拓扑，可简化软件操作流程：只需要对一个波形进行编程和控制即可驱动两路输出。

死区插入通过将 OUTxR 寄存器中 DTEN 位置 1 来使能。互补信号基于输出 A 定义的参考波形构建而成，使用 SETxAR 和 CLRxAR 寄存器；如果 DTEN 位置 1，则 SETxBR 和 CLRxBR 寄存器无意义。

可按照与参考波形上升沿和下降沿的关系定义两个死区，如图 19-13 所示。

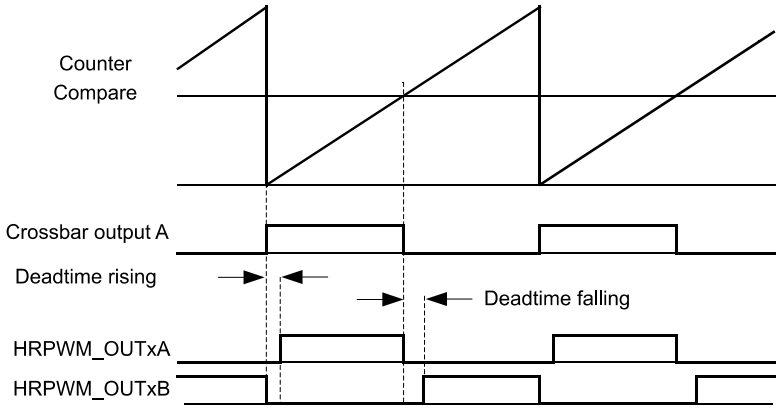


图 19-13 已插入死区的互补输出

如果需要使用一些控制重叠，可定义负死区，此时要使用死区符号位（DTxR 寄存器中的 SDTFx 和 SDTRx 位）。

图 19-14 显示了各符号位对应的互补信号波形。

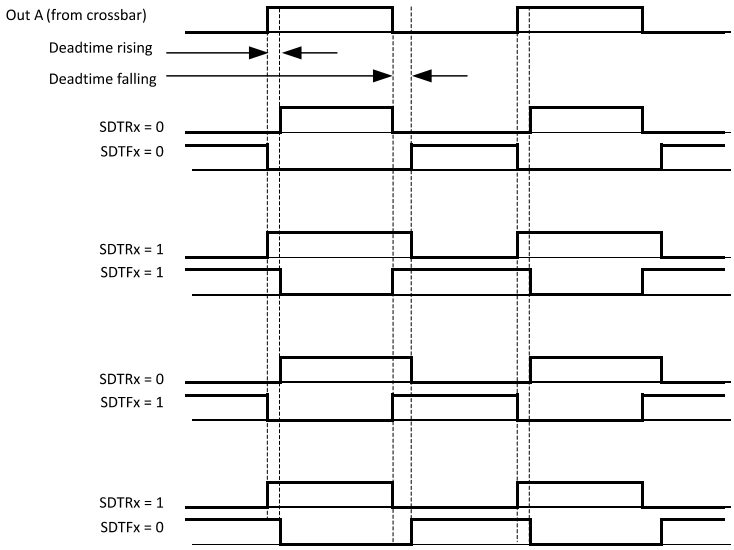


图 19-14 死区插入与死区符号位（符号位为 1 表示负死区）

死区值使用 DTFx[11:0]和 DTRx[11:0]位定义的，基于根据 CKPSC[2:0]位预分频的特定时钟，具体如下：

$$t_{DTx} = +/- DTx[8:0] \times t_{DTG}$$

其中 x 为 R 或 F，t_{DTG} =（$2^{(CKPSC[2:0])}$）x（t_{HRPWM}）。

表 19-9 给出了根据预分频值得出的分辨率和最大绝对值。

表 19-9 死区分辨率和最大死区时间

CKPSC[2:0]	t_{DTG}	T_{DTx} max	f_{HRPWM} = 160 MHz	
			t_{DTG} (ns)	$\|t_{DTx}\|$ max (µs)
000	**t_{HRPWM} / 32**	**4095 * t_{DTG}**	0.195	0.800
001	**t_{HRPWM} / 16**		0.391	1.600
010	**t_{HRPWM} / 8**		0.781	3.199
011	**t_{HRPWM} / 4**		1.563	6.398
100	**t_{HRPWM} / 2**		3.125	12.797
101	**t_{HRPWM}**		6.25	25.594
110	**2 * t_{HRPWM}**		12.5	51.188
111	**4 *t_{HRPWM}**		25	102.375

注：以下情况下，不得更改 DTEN 位：

- *定时器使能时（CENx 位置 1）*
- *定时器输出由另一定时器置位/复位时（CENx 复位时）*

否则会引发不可预测的行为。

因此，需要禁止定时器（CENx 位复位）并禁止相应的输出。

19.4.5 主定时器

主定时器的主要用途是向 6 个定时单元提供公共信号，以便进行同步或置位/复位输出。主定时器不会直接控制任何输出，但可以间接地控制输出，由置位/复位交错配置矩阵实现。

图 19-15 给出了主定时器的概览。

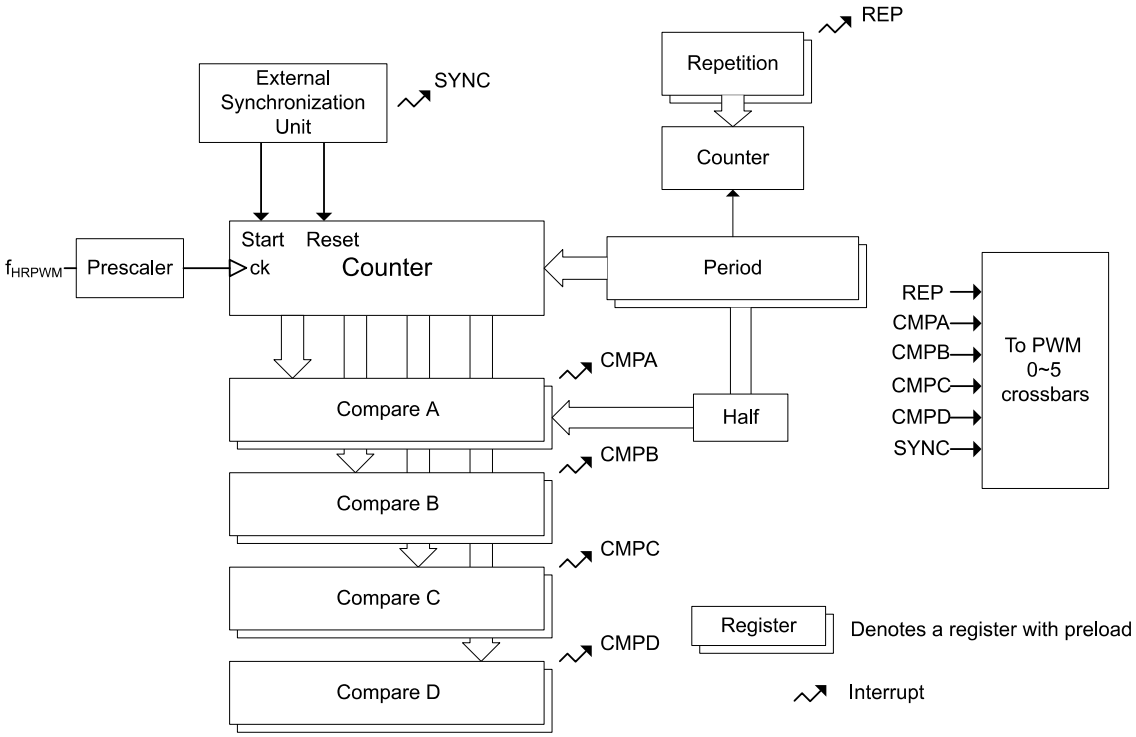


图 19-15 主定时器概览

主定时器采用的架构与定时单元非常相似，二者的不同之处如下：

- 主定时器未关联输出，也没有输出相关的控制
- 主定时器没有自己的交错配置矩阵，也没有推挽或死区模式
- 主定时器只能通过软件或者外部同步电路复位
- 主定时器不包含外部事件消隐和开窗电路
- 主定时器中断请求数量有限：比较 A~比较 D、周期事件、重复事件、寄存器更新事件和同步事件。

主定时器控制寄存器包含主定时器和定时单元 0~5 的所有定时器使能位。这样可通过单次写访问同时启动所有定时器。

主定时器还会利用 MCU 内部和外部（输入/输出）资源，处理整个 HRPWM 定时器的外部同步。

主定时器控制寄存器的映射偏移与定时单元寄存器的偏移相同。

19.4.6 上-下计数模式

HRPWM 默认的计数模式为上计数模式（递增），然后 HRPWM 同样支持上-下计数模式（递增递减），也称为中心对齐模式。

通过 PWMxCR1 寄存器的 UDM 位使能上-下计数模式。一旦定时器开始工作（CENx 位置 1），此位不可以被更改。上-下计数模式只适用于定时器 x，主定时器仅在上计数模式下工作。

本节详细介绍了上-下计数模式与上计数模式的功能差异。

在上-下计数模式下，PERxR 中的周期值必须开启预装载功能（或保持为静态值）。周期值只能在周期事件或者计数复位的情况下更新：

置位/复位交错配置矩阵的不同之处如下：

来自定时单元的事件对输出置位/复位的效果与计数的上-下方向有关：

- 如果事件在 SETxyR 寄存器中使能，上计数期间的事件将输出置位，下计数期间的事件将输出复位。
- 如果事件在 CLRxyR 寄存器中使能，上计数期间的事件将输出复位，下计数期间的事件将输出置位。
- 如果事件在 SETxyR 和 CLRxyR 寄存器中同时使能，事件将输出翻转。

以上机制适用于：

- 定时单元：周期，比较 A~比较 D、寄存器更新（6 个事件）
- 主定时器：周期，比较 A~比较 D、HRPWM 同步（6 个事件）

图 19-16 显示了如何生成基本的波形。

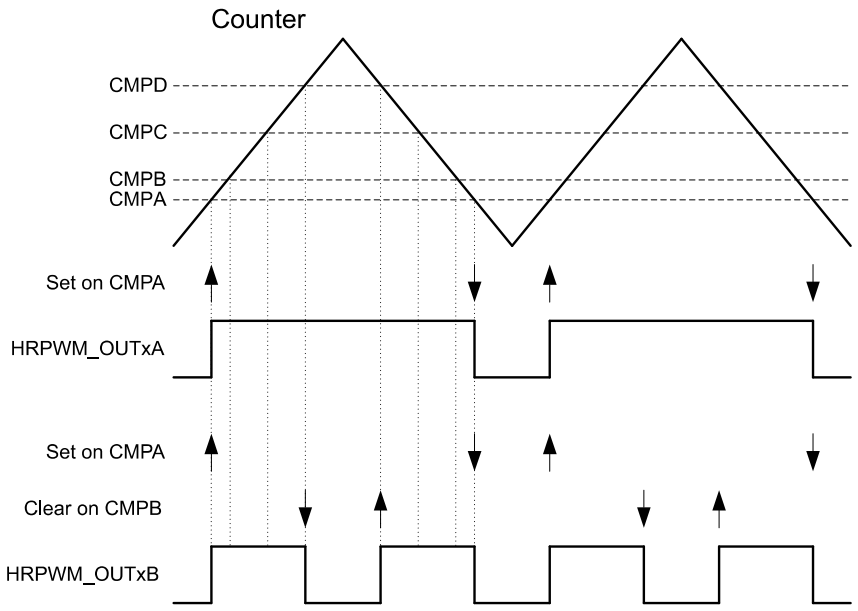


图 19-16 上-下模式下的简单对称波形

图 19-17 显示了如何生成一些更复杂的波形，使用了 4 个比较单元以及输出翻转模式。

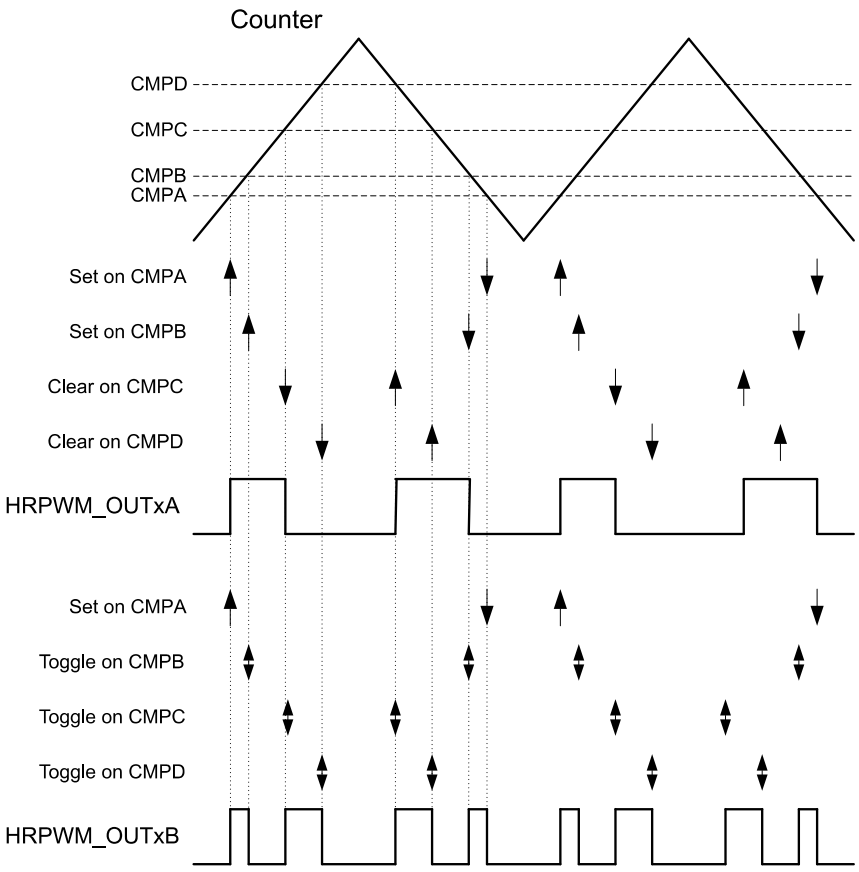


图 19-17 上-下计数模式下的复杂对称波形

图 19-18 显示了如何生成一个非对称的波形。在这种情形下，需要注意 CMPB 值必须大于 CMPA 值才能确保波形不对称。

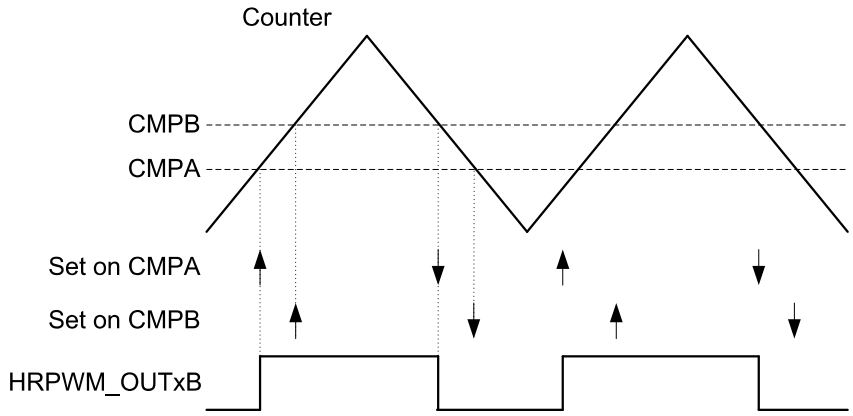


图 19-18 上-下计数模式下的非对称波形

注：对于非对称的情形，需要满足 CMPB> CMPA。

软件强制位和外部事件 Event 0~5 的行为在上计数和上-下计数模式下是相同的。图 19-19 显示了脉冲宽度可以通过外部事件进行缩短。

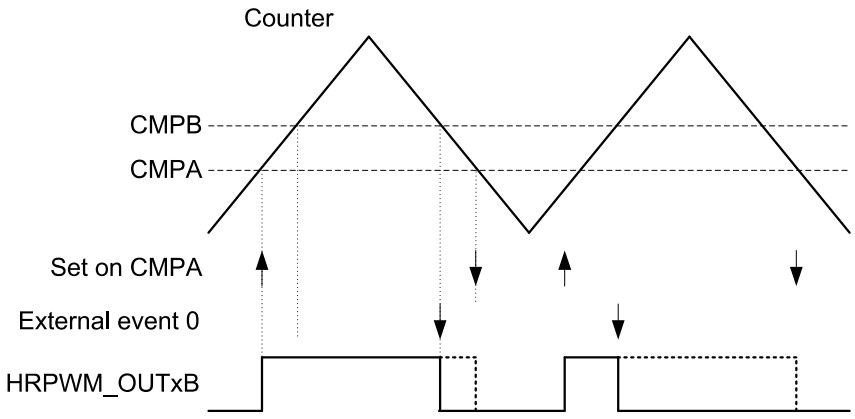


图 19-19 上-下计数模式下的外部事件处理

上-下计数模式适用于连续模式和单次（可重触发与不可重触发）工作模式。复位请求会导致计数器从 0 重新启动。图 19-20、图 19-21 显示了定时器 1 在单次可重触发模式下的计数器行为。

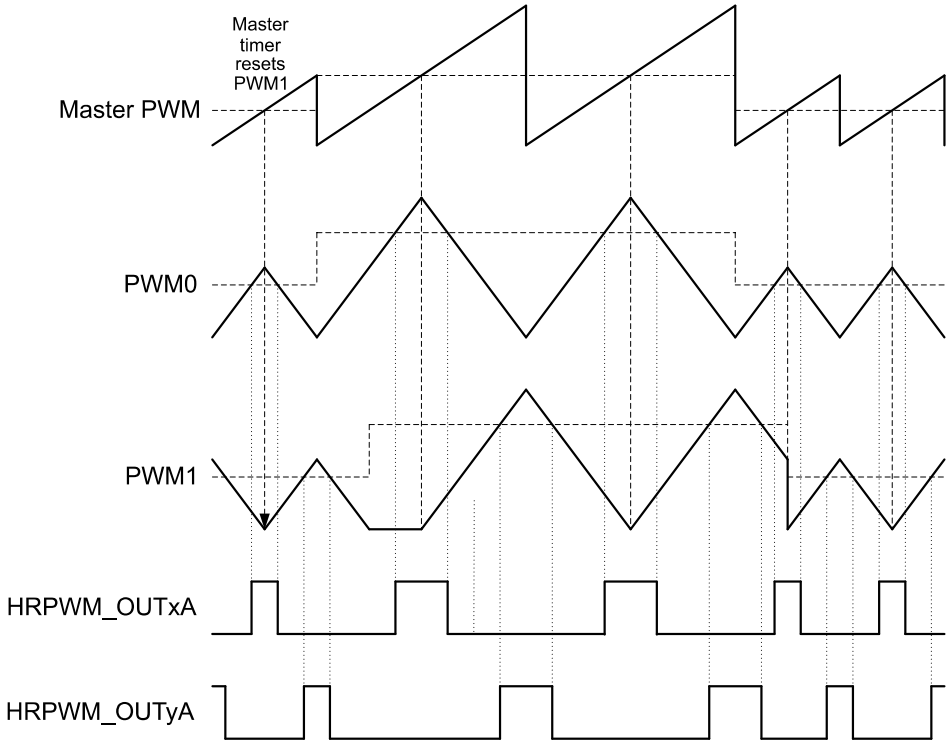


图 19-20 交错计数在上-下模式下的示例

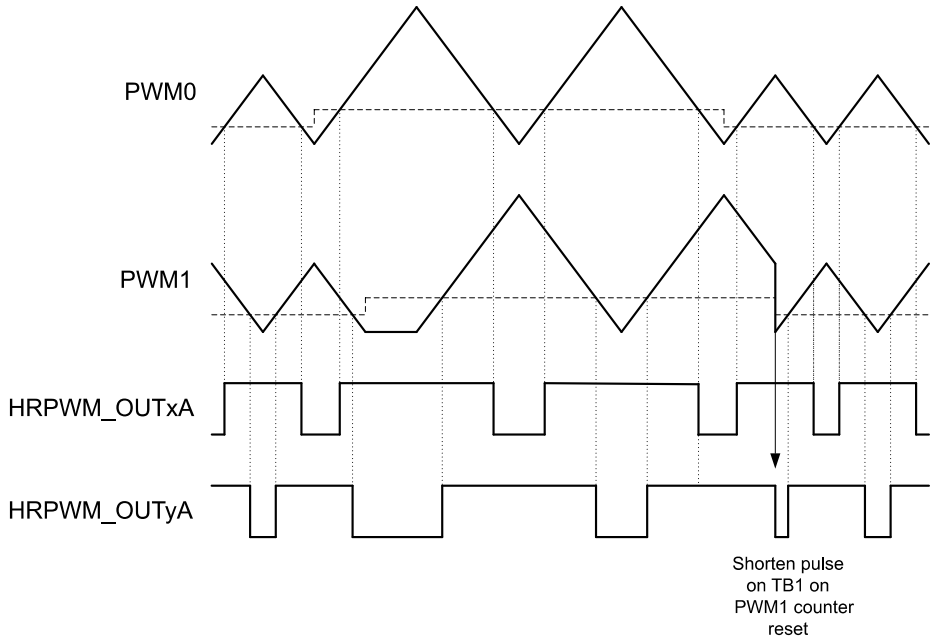


图 19-21 交错计数在上-下模式下的示例

注：在上-下计数模式下，比较值必须小于周期值减去 3×f_{HRPWM} 时钟周期。
若 CKPSC[2:0] = 0，则比较值上限为（PERxR - 0xC0）
若 CKPSC[2:0] = 1，则比较值上限为（PERxR - 0x60）
若 CKPSC[2:0] = 2，则比较值上限为（PERxR - 0x30）

以上适用于在定时单元内部产生的比较事件。

上-下计数模式支持以下功能：
- 交错模式
- 死区插入
- 推挽模式，计数器= 0 时完成推挽交替（如图 19-22 所示）

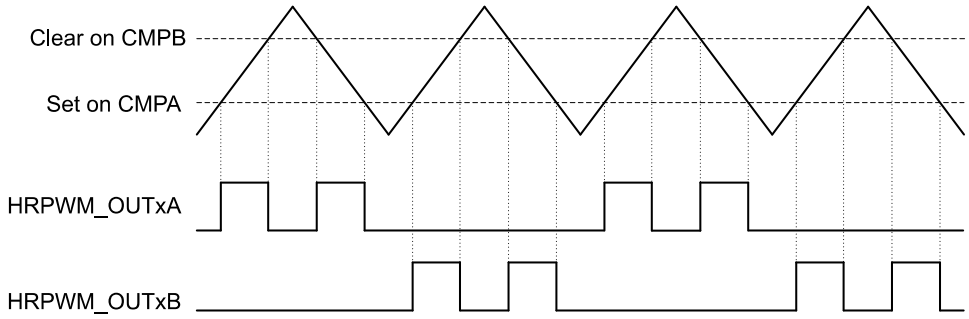


图 19-22 推挽模式在上-下计数模式的示例

计数翻转事件在上-下计数模式下的定义与上计数模式下有所不同，以支持不同的工作条件。
计数翻转事件可以由以下方式生成：
- 在计数值等于 0 时生成（波谷模式）
- 在计数值等于 PERxR 时生成（波峰模式）
- 在计数值等于 0 或 PERxR 时生成（波峰-波谷模式）

计数翻转事件在 HRPWM 中会用在很多地方，根据不同用途可以配置不同的生成方式（波谷、波峰或波峰-波谷）。表 19-10 总结了不同用途下相应的计数翻转模式配置位，xxROM[1:0]位于 PWMx_CR1 寄存器中。

表 19-10 翻转事件用途和生成方式编程

翻转事件用途	编程控制位
输出置位/复位	OUTROM[1:0]
中断请求	ROM[1:0]
ADC 触发事件	ADROM[1:0]
外部事件滤波	EEVROM[1:0]
故障/事件计数器	FLTROM[1:0]

注：对于同时考虑复位以及翻转的事件(IRQ 和 RSTU)，ROM[1:0]只影响翻转事件产生。无论 ROM[1:0]值是多少，复位事件总会被考虑。

计数器翻转事件生成方式由 xxROM[1:0]位的配置定义：

- xxROM[1:0] = 00：在计数值等于 0 或者 PERxR 时生成（波峰-波谷模式）
- xxROM[1:0] = 01：在计数值等于 0 时生成（波谷模式）
- xxROM[1:0] = 10：在计数值等于 PERxR 时生成（波峰模式）

图 19-23 显示了推挽输出在上-下计数模式下的行为，输出在周期（计数器翻转）事件置位，（OUTROM[1:0] =10）。

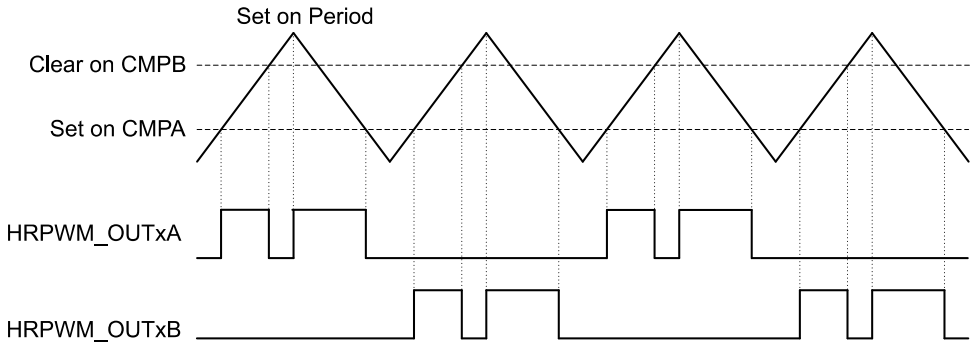


图 19-23 上-下计数模式下输出在周期事件置位（OUTROM[1:0]=10）

图 19-24 显示了在上-下计数模式下重复计数器是如何递减的。

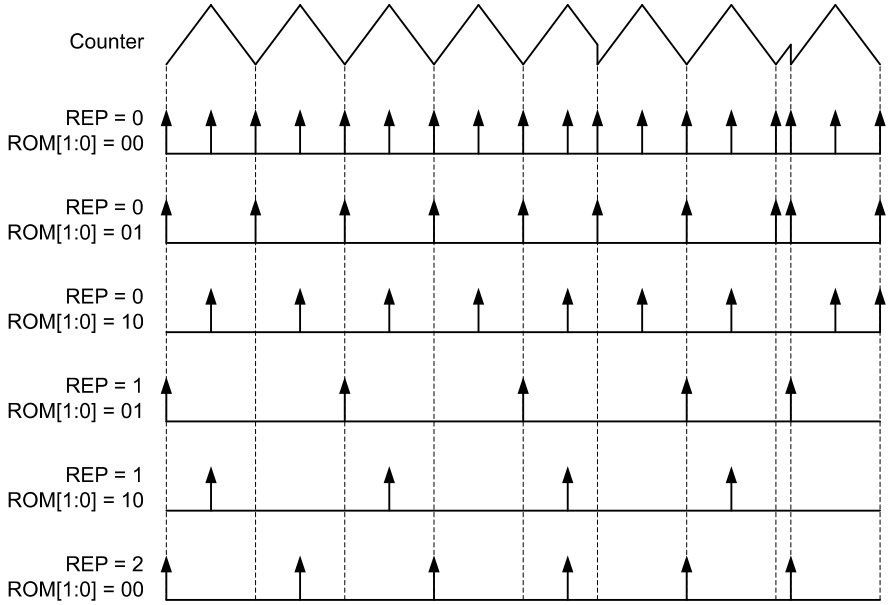


图 19-24 上-下计数模式下的周期重复事件

DAC 触发事件在上-下计数模式下与在上计数模式下一致。

事件消隐和加窗的功能在上-下计数模式下与在上计数模式下不同。EExFLTR[3:0]的功能依赖于 UDM 位的配置，如表 19-11 所示，可以看出消隐/开窗区间根据 UDM 的不同有所不同。无论何时将翻转事件用于事件消隐或开窗时，翻转事件的生成受 EEVROM[1:0]位控制。

表 19-11 EExFLTR[3:0]的功能与 UDM 位设置

EExFLTR[3:0]	上计数模式 (UDM = 0)	上-下计数模式 (UDM = 1)
1111	无消隐与加窗滤波	从上阶段 CMPB 到下阶段 CMPC 加窗
1110	无消隐与加窗滤波	从下阶段 CMPB 到下阶段 CMPC 加窗
1101	无消隐与加窗滤波	从上阶段 CMPB 到上阶段 CMPC 加窗
1100	从复位 / 翻转到 CMPD 加窗	从复位 / 翻转到 CMPD 加窗
1011	从复位 / 翻转到 CMPC 加窗	从复位 / 翻转到 CMPC 加窗
1010	从复位 / 翻转到 CMPB 加窗	从复位 / 翻转到 CMPB 加窗
1001	从复位 / 翻转到 CMPA 加窗	从复位 / 翻转到 CMPA 加窗
1000	无消隐与加窗滤波	从下阶段 CMPC 到下阶段 CMPD 消隐
0111	无消隐与加窗滤波	从下阶段 CMPA 到下阶段 CMPB 消隐
0110	无消隐与加窗滤波	从上阶段 CMPC 到上阶段 CMPD 消隐
0101	无消隐与加窗滤波	从上阶段 CMPA 到上阶段 CMPB 消隐
0100	从复位 / 翻转到 CMPD 消隐	从复位 / 翻转到 CMPD 消隐
0011	从复位 / 翻转到 CMPC 消隐	从复位 / 翻转到 CMPC 消隐
0010	从复位 / 翻转到 CMPB 消隐	从复位 / 翻转到 CMPB 消隐
0001	从复位 / 翻转到 CMPA 消隐	从复位 / 翻转到 CMPA 消隐
0000	无消隐与加窗滤波	无消隐与加窗滤波

19.4.7 置位/复位优先级和窄脉冲管理

本节介绍了在 3 个连续的 f_{HRPWM} 时钟周期内出现多个置位和/或复位请求时，输出波形是如何产生的。

情形 1:时钟预分频 CKPSC[2:0] < 5

每个 f_{HRPWM} 周期内都会执行仲裁，具体分为三步：

1. 对于每个有效事件，会确定所需输出跳变（置位、复位或翻转）。
2. 在有效事件之间执行预定义的仲裁（从最高到最低优先级）：PER → CMPD → CMPC → CMPB → CMPA。
3. 在预定义高分辨率延迟仲裁中，在低分辨率事件和已经获得预定于仲裁的事件中，CLEAR 事件具有最高优先级。

如果 SET 和 CLEAR 请求之间的间隔小于 $2\times f_{HRTIM}$ 周期，则行为取决于时间间隔和与 f_{HRTIM} 时钟的对齐方式，如图 19-25 所示。

注：如果置位和复位请求是同时发生的，并且来自同一个定时单元，则 CMPx 的优先级生效，如上面的步骤 2 所示。

例如，假设 CMPB = CMPD：

- 如果 CMPB 置位，CMPD 复位，输出会复位。
- 如果 CMPB 复位，CMPD 置位，输出会置位。

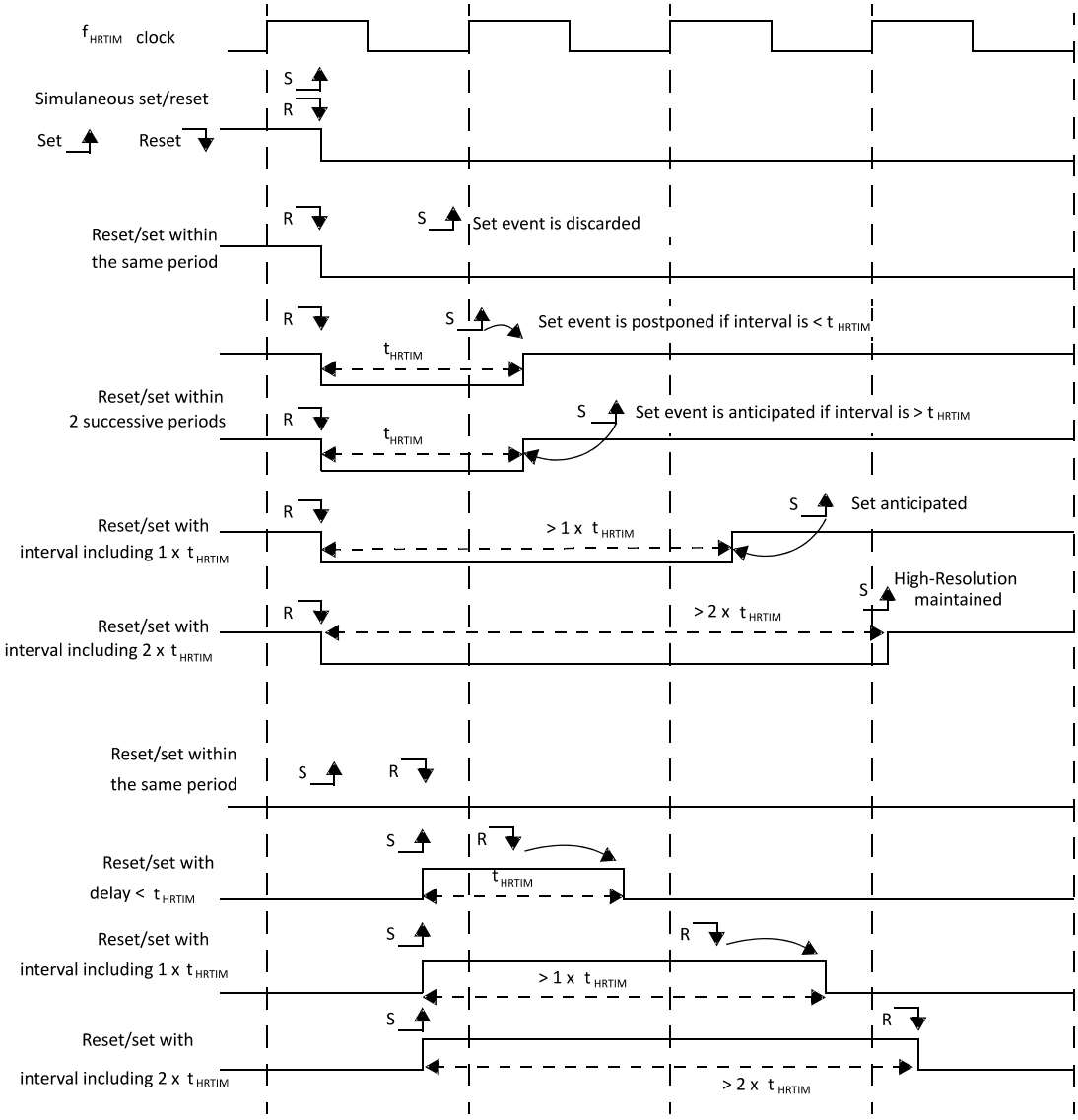


图 19-25 窄脉冲管理：近距离复位/置位事件的管理

如果置位和复位事件在相同的 f_{HRPWM} 周期内产生，则复位事件具有最高优先级，置位事件会被忽略。

如果置位和复位事件生成的间隔小于 1 个 f_{HRPWM} 时钟周期且跨越 2 个周期，会生成宽度为 1 个 f_{HRPWM} 周期的脉冲。

如果置位和复位事件生成的间隔小于 2 个 f_{HRPWM} 时钟周期，会生成宽度为 2 个 f_{HRPWM} 周期的脉冲。

如果置位和复位事件生成的间隔为 2~3 个 f_{HRPWM} 时钟周期，高分辨率在间隔超过 2 个完整 f_{HRPWM} 时钟周期时有效。

如果置位和复位事件生成的间隔大于 3 个 f_{HRPWM} 时钟周期，高分辨率一直有效。

并发置位请求/并发复位请求

如果为置位事件选择了多个源，则在同一 f_{HRPWM} 时钟周期内出现多个置位请求时，会执行仲裁。

如果主定时器在同一 f_{HRPWM} 时钟周期内发出多个请求，则会应用预先定义的仲裁，并会考虑单个有效的请求（优先级从高到低），无论有效的高分辨率设置如何：

MSTCMPER → MSTCMPD → MSTCMPC → MSTCMPB → MSTCMPA

注：建议避免从主定时器产生多组时间间隔低于 3×f_{HRPWM}（CLEAR）请求到给定的定时单元，以保持高分辨率。

如果在同一 f_{HRPWM} 时钟周期内出现多个定时器内部请求，则会应用预先定义的仲裁，并会按照以下优先级顺序考虑请求（从最高到最低），而不考虑有效定时：

PER → CMPD → CMPC → CMPB → CMPA

最后，最高优先级会分配给低分辨率事件：EXTEVNT0~5，RESYNC（来自 SYNC 事件（SYNCRSTx 或 SYNCSTRTx 置 1 时）或软件复位），更新和软件置位（SST）。

总之，对于并发事件（在相同的 f_{HRPWM} 时钟周期内发生的事件），有效置位/复位事件将在以下事件之间进行仲裁：

- 来自主定时器的单个源（按照上文所述的固定仲裁）
- 来自定时单元的单个源
- 低分辨率事件

仲裁原则同样适用于并发复位请求。在这种情况下，复位请求具有最高优先级。

情形 2：时钟预分频 CKPSC[2:0]>=5

窄脉冲管理在高分辨率无效时可以简化。

在预分频时钟周期内发生的置位或者复位事件会延迟到预分频时钟的下一个有效边沿（比如计数器复位时间），即使仲裁仍在每个 f_{HRPWM} 周期内进行。

19.4.8 外部事件全局调节

HRPWM 定时器可处理并非定时器中生成的事件，此类事件被称为外部事件，外部事件来自多个片上或者片外来源：

- 内置比较器
- 数字输入引脚（通常连接到片外比较器和过零检测器）
- 其他外设的片上事件（ADC 的模拟看门狗和通用定时器触发输出）

表 19-12 总结了可用于 6 个外部事件的源。

表 19-12 外部事件输入来源和相关功能

外部事件	外部事件来源（EExSRC [1：0]）				不同封装下的比较器和输入源	
	Src0 (00)	Src1 (01)	Src2 (10)	Src3 (11)	48-pin	64-pin and 100-pin
hrpwm_ext_evt0 [3:0]	HRPWM_ EVT0	CMP0_ OUT	TMR4_ TRGO	ADC0_ AWD0	Comp	Comp & input
hrpwm_ext_evt1 [3:0]	HRPWM_ EVT1	CMP1_ OUT	TMR5_ TRGO	ADC0_ AWD1	Comp	Comp & input
hrpwm_ext_evt2 [3:0]	HRPWM_ EVT2	CMP2_ OUT	TMR6_ TRGO	ADC0_ AWD2	Comp & input	Comp & input
hrpwm_ext_evt3 [3:0]	HRPWM_ EVT3	CMP3_ OUT	TMR7_ TRGO	ADC1_ AWD0	Comp & input	Comp & input
hrpwm_ext_evt4 [3:0]	HRPWM_ EVT4	CMP0_ OUT	TMR0_ TRGO	ADC1_ AWD1	Comp & input	Comp & input
hrpwm_ext_evt5 [3:0]	HRPWM_ EVT5	CMP2_ OUT	TMR1_ TRGO	ADC1_ AWD2	Comp & input	Comp & input

外部事件调节电路可为给定通道选择事件源（使用 4:1 多路复用器），并可将信号源转换为可由交错配置矩阵处理的形式（例如，通过在外部事件通道上检测到的下降沿来触发输出复位）。

最多可调节 6 个外部事件通道，这些外部事件通道可同时用于 6 个定时器中的任何一个。由于这种调节通常取决于外部组件（比如过零检测器）和环境条件（滤波器设置通常与应用噪声级和信号有关），因此对于所有定时器是通用的。图 19-26 显示了单条事件通道的调节逻辑概览。

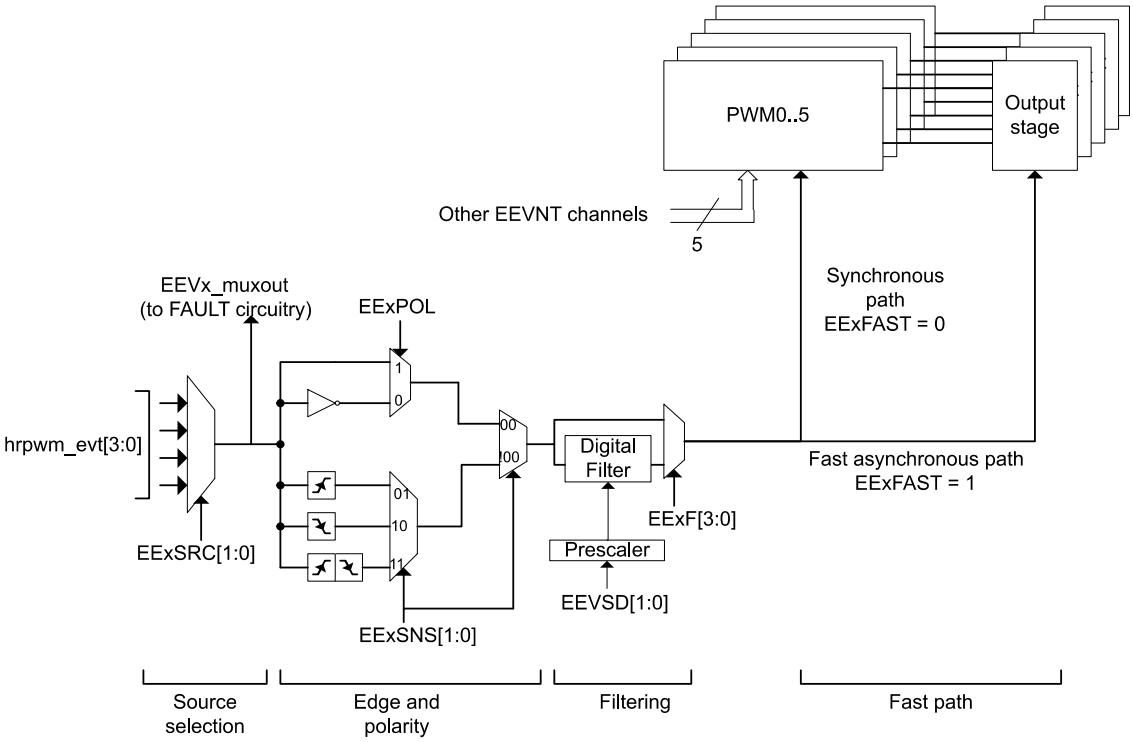


图 19-26 外部事件调节概览（显示了 1 条通道）

6 个外部事件通过 EECR0、EECR1 和 EECR2 寄存器初始化：

- 使用 EExSRC[1:0]位选择事件源，每个外部事件有 4 个源
- 使用 EExSNS[1:0]位选择有效沿，电平有效或者边沿有效（上升沿、下降沿或上升/下降沿），
- 使用 EExPOL 位选择触发电平的极性，极性在电平有效与边沿有效下均生效，
- 使用 EExFAST 位配置外部事件 0~5 的低延迟模式

注：即使 EExSNS[1:0]=0（选择电平有效），用作复位、ADC 触发事件的外部事件也是边沿有效：如果 EExPOL =0，触发事件在外部事件上升沿有效，如果 EExPOL =1，触发事件在外部事件下降沿有效。

计数器禁止后（CENx=0），会立即丢弃外部事件，以防止输出状态更改和计数器复位，外部事件用于 ADC 触发事件时不受影响。

此外，还可以使用 EECR2 寄存器中的 EExF[3:0]位为外部事件使能数字噪声滤波器。

数字滤波器由计数器组成，需要使用 N 个有效采样来验证输出跳变。如果输入值在计数器达到 N 之前发生变化，计数器会复位，跳变会被丢弃（视为伪事件）。如果计数器达到 N，跳变被视为有效，并会作为正确的外部事件进行传输。因此，数字滤波器会向正在进行滤波的外部事件添加延迟，延迟时长视采样时钟和滤波器长度（预期的有效采样数）而定。

采样时钟为 f_{HRPWM} 时钟或由 f_{HRPWM} 预分频而生成的特定时钟 f_{EEVS}，预分频通过 EECR2 寄存器中的 EEVSD[1:0]位定义。

外部事件延迟

外部事件调节能够根据性能预期调整外部事件的处理时间（以及相关延迟）：

- 常规工作模式：在该模式下，会在对交错配置矩阵进行操作之前，使用时钟对外部事件进行重新采样。此过程会增加一些延迟，但可以使用所有交错配置矩阵功能，从而生成外部事件触发的高分辨率脉冲。
- 快速工作模式：在该模式下，外部事件与输出动作之间的延迟得到最大限度地缩短，该模式便于实现超快速的过流保护等功能。

使用 EECR0 寄存器中的 EExFAST 位来定义外部事件 0~5 的工作模式，这会影响输出脉冲上存在的延时和抖动，如下表 19-13 所示。

表 19-13 输出置位/复位的延迟/抖动与外部事件工作模式

EExFAST	反应时间延迟	响应时间抖动	输出脉冲抖动 （由 Ext 事件计数器复位）
0	5~6 个 f_{HRPWM} 时钟周期	1 个 f_{HRPWM} 时钟周期	无抖动，脉冲宽度保持高分辨率
1	最小延迟（取决于使用比较器还是数字输入）	最小抖动	1 个 f_{HRPWM} 时钟周期的抖动 脉宽分辨率下降到 t_{HRPWM}

EExFAST 模式仅适用于电平有效模式（EXSNS[1:0]＝00），不适用于边沿有效模式。

可以对外部事件应用事件过滤（EExFLTR0[3:0]!=0x0，消隐和加窗），在这种情况下，EExLTCH 位必须复位，不支持延迟模式。

注：相关 EExFAST 位置 1 后，不得修改外部事件配置（来源和极性）。

快速外部事件不能用于翻转输出，必须在 SETxyR 或 CLRxyR 寄存器中使能，而不能在两个寄存器中同时使能。

如果置位事件和复位事件（来自 2 个独立的快速外部事件）同时发生，则复位事件在交错配置矩阵中的优先级最高，输出变为无效状态。

如果 EExFAST 位置 1，则在外部事件发生后的 11 个 f_{HRPWM} 时钟周期内，输出都不能更改。

图 19-27 和图 19-28 给出了进行输出置位/复位和计数器复位时对外部事件的实际响应时间示例。

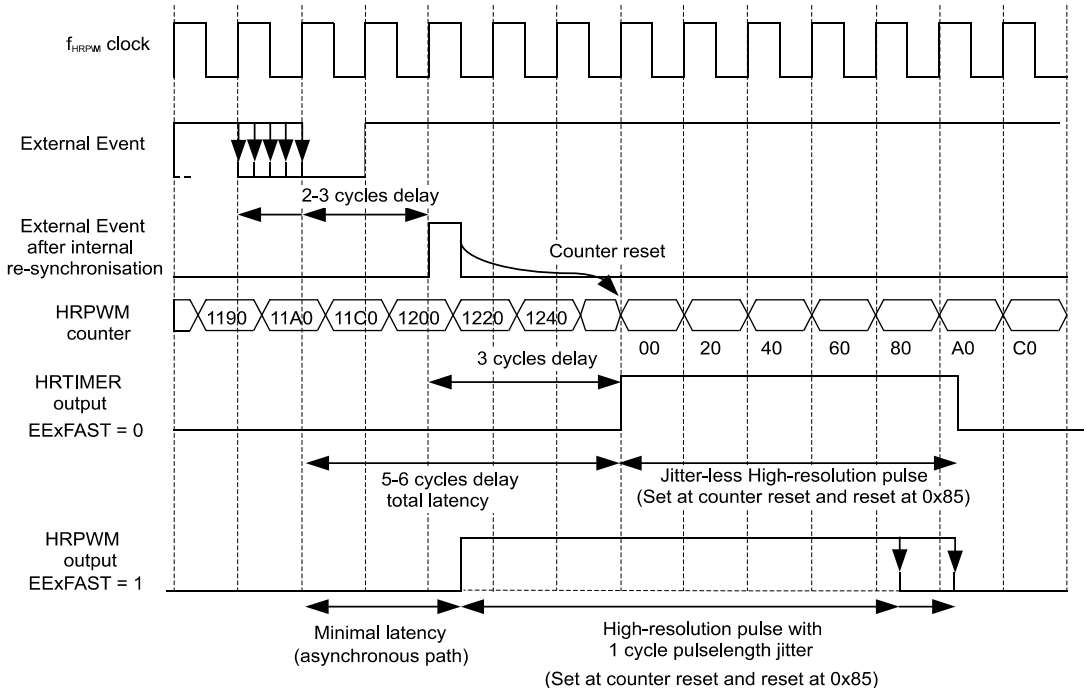


图 19-27 外部事件的延迟（外部事件使计数器复位和输出置位）

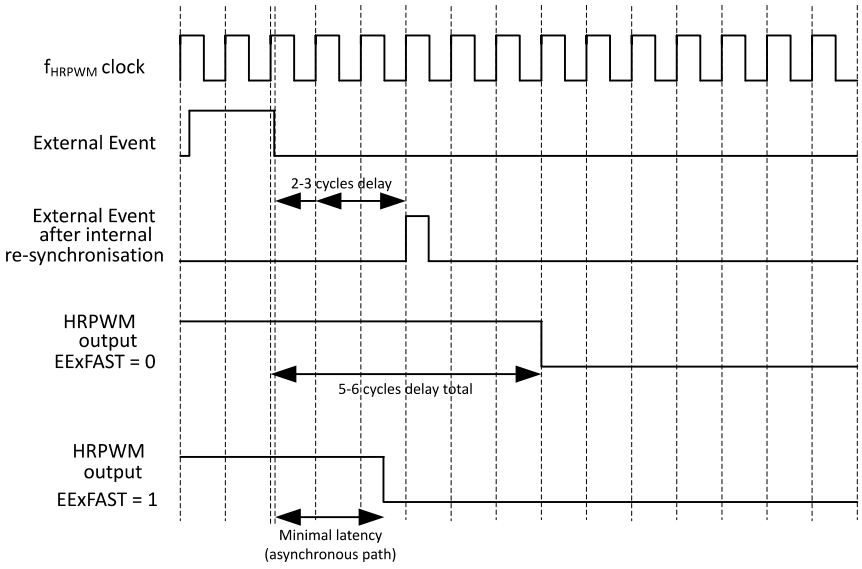


图 19-28 外部事件的延迟（外部事件使输出复位）

19.4.9 定时单元中的外部事件过滤

经过调节后，6 个外部事件可供所有定时单元使用。

这些事件可直接使用，且在定时单元计数器使能后（CENx 位置 1）立即生效。

此外，还可对这些事件进行过滤，以便在限制时间段内执行操作，该时间段通常与计数周期有关。可执行两种操作：

- 消隐模式，在定义的时间段内屏蔽外部事件，
- 加窗模式，仅在定义的时间段内使能外部事件。

这些模式通过 EEFxR0 寄存器中的 EExFLTR 位来使能，6 个定时单元都具有针对这 6 个外部事件的可编程过滤器设置。

消隐模式

在事件消隐模式下，如果外部事件发生在给定的消隐周期内，则会被忽略，如图 19-29 所示。举例来说，为了避免 PWM 周期开始时电流限制因开关噪声而失效，可使用此模式。当 EExFLTR[3:0]位的值在 0001 变化到 1000 范围内时，该模式有效。

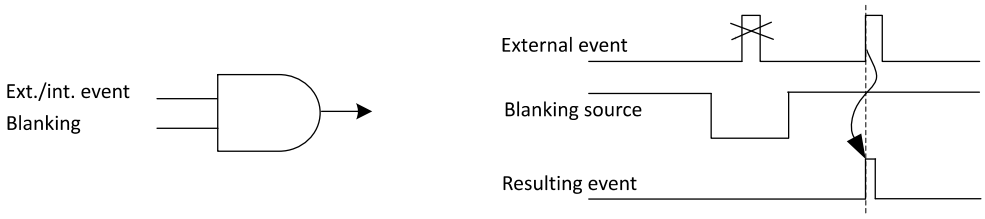


图 19-29 事件消隐模式

在事件延迟模式下，不会立即考虑外部事件，而是会将其保存（锁存）下来，并在消隐周期结束后立即生成外部事件，如图 19-30 所示。通过将 EEFxR0 寄存器中的 EExLTCH 位置 1 来使能此模式。

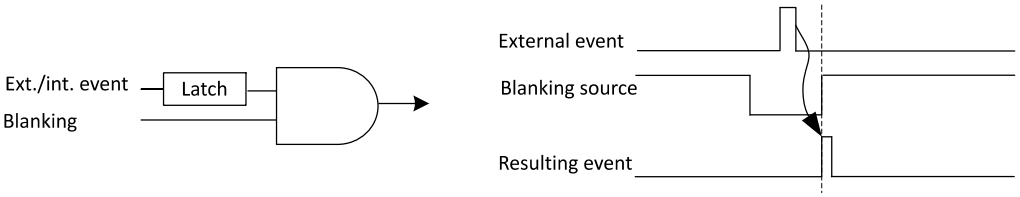


图 19-30 事件延迟模式

消隐信号来自多个来源，编码如下：

- 定时器本身：消隐持续时间从计数器复位开始，到比较值匹配为止（EExFLTR[3:0]＝0001 到 0100），也可以从一个比较值匹配开始，到另一个比较值匹配为止（EExFLTR[3:0]=0101 到 1000）。在 Updown 模式（UDM 位设置为 1）下，计数复位事件与 ROM[1:0]位设置有关。

图 19-31、图 19-32 举例说明了常规模式和延迟模式下，各种边沿有效和电平有效的外部事件消隐。

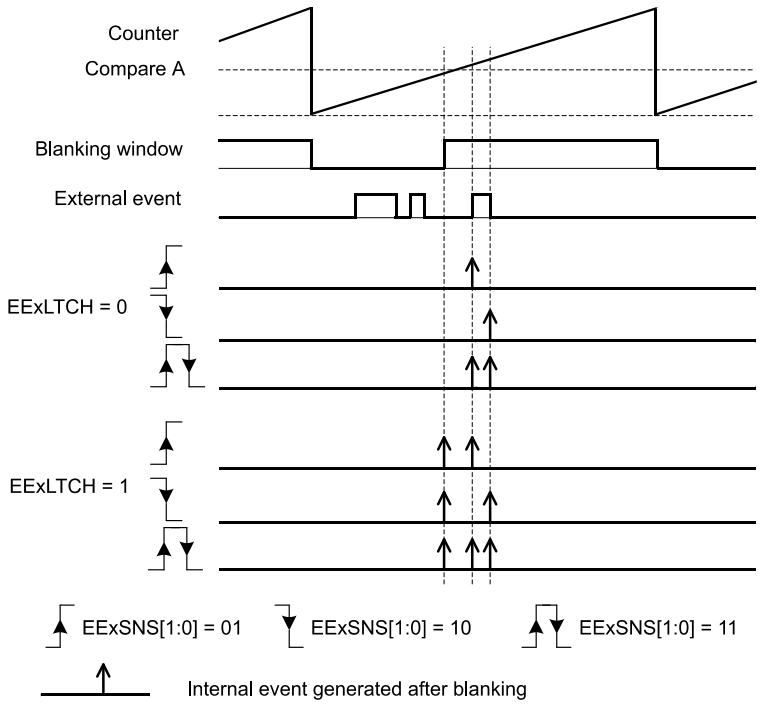


图 19-31 采用边沿有效的外部事件消隐

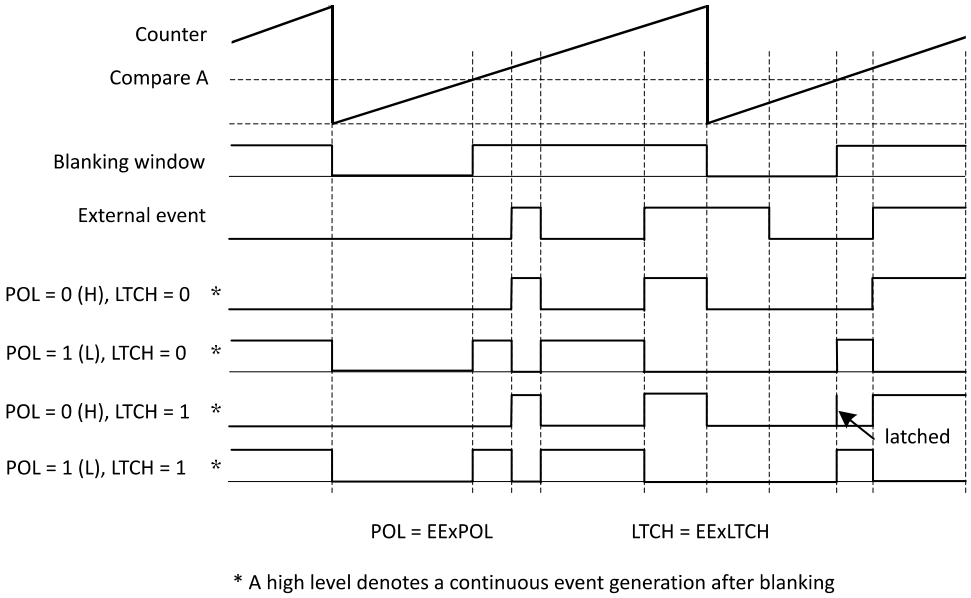


图 19-32 采用电平有效的外部事件消隐

加窗模式

在事件加窗模式中，仅当事件在给定事件窗口内发生时才会被考虑，否则会被忽略。当EExFLTR[3:0]位的值在 1001 到 1111 范围内时，该模式有效。

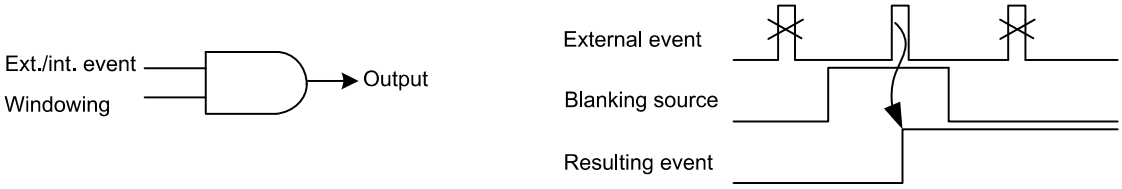


图 19-33 事件加窗模式

使用 EEFxR0 寄存器的 EExLTCH 位开启事件延迟模式，当事件延迟模式有效（EExLTCH=1）时，如果事件在窗口期间发生，该事件延迟到窗口末尾才生效。

- 如果 EExLTCH=0 且事件在窗口期间发生，则事件直接生效。

加窗模式可用于过滤同步事件。

加窗信号来自多个来源，编码如下：

- 定时器本身：加窗持续时间从计数器复位开始，到比较值匹配为止（EExFLTR[3:0]＝1001到 1100），也可以从一个比较值匹配开始，到另一个比较值匹配为止（EExFLTR[3:0]=1101到 1111）。在 Updown 模式（UDM 位设置为 1）下，计数复位事件与 ROM[1:0]位设置有关。

图 19-34 和图 19-35 举例说明了在常规模式和事件延迟模式下，各种边沿有效和电平有效的外部事件加窗。

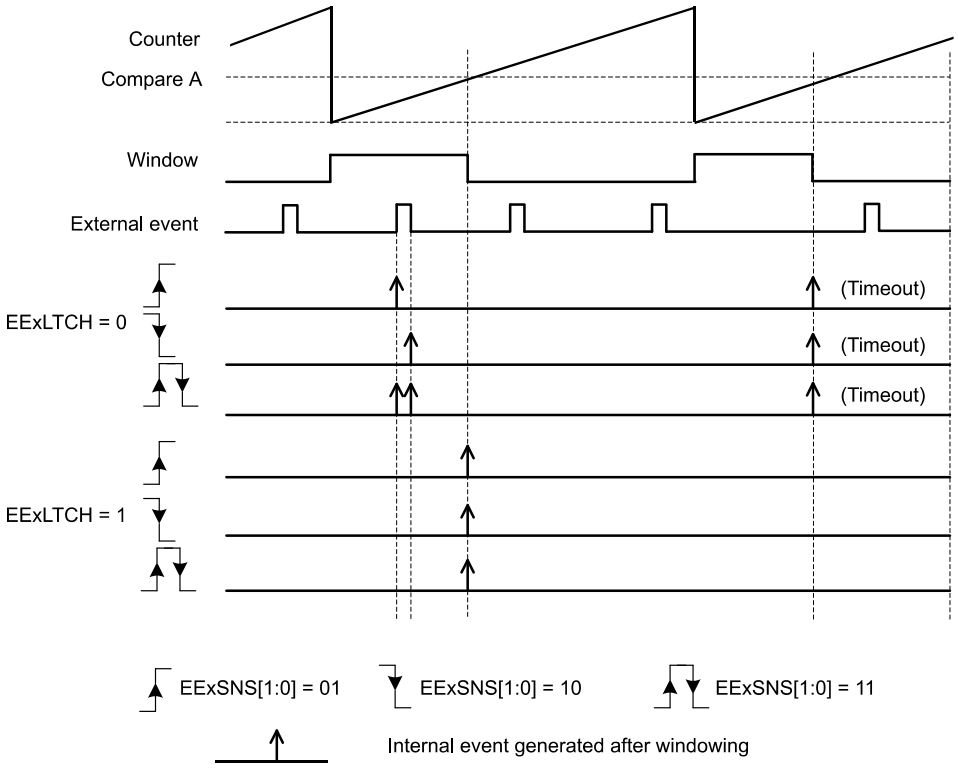


图 19-34 采用边沿有效的外部事件加窗

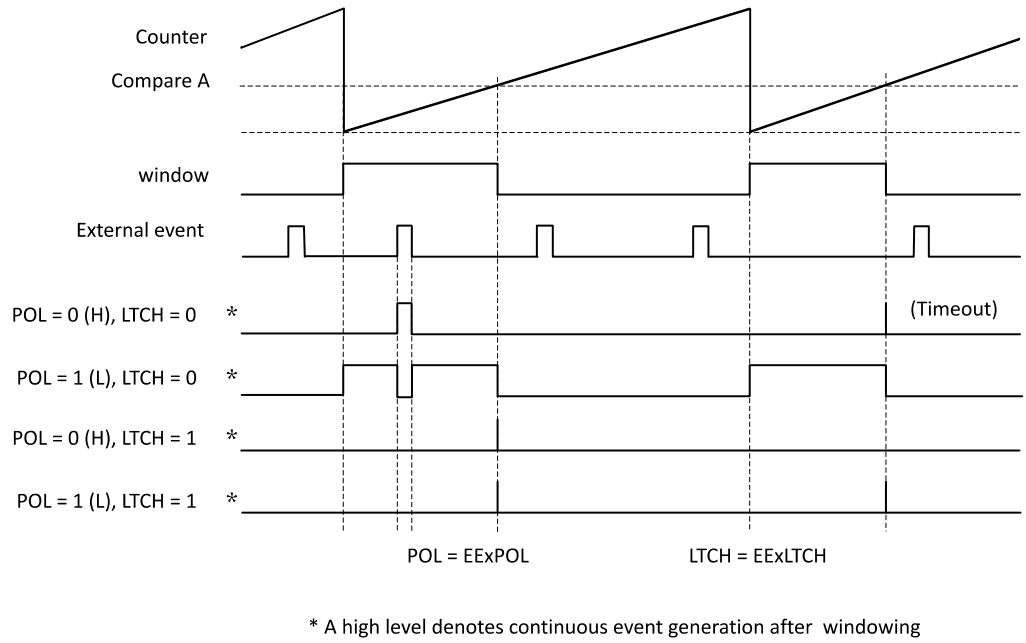


图-19-35 采用电平有效的外部事件加窗

外部事件计数器

每个定时单元在事件过滤之后，有一个外部事件计数器，通常实现波谷信号过滤。

计数器可以对经过事件过滤的 6 个外部事件进行滤波，如图 19-36 所示。

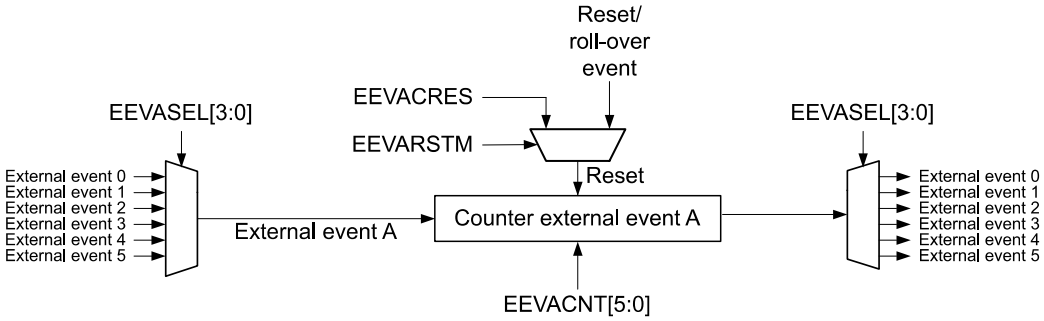


图 19-36 外部事件计数器-通道 A

外部事件计数器通过配置 EEFxR1 寄存器中的 EEVACE 位使能。此模式仅适用于边沿有效的外部事件（EExSNS[1:0]＝01,10 或 11）。

只有当有效边沿的数目大于或等于（EEVACNT[5:0] + 1）编程设定的值时，外部事件才会传输到定时器。

计数器支持两种工作模式：

- 无累积模式：当 EEVARSTM=0 时，外部事件计数器在每一个复位/翻转事件处清零：外部事件只有在给定的 PWM 周期内出现多次时才是有效的。
- 累积模式：当 EEVARSTM=1 时，只有在上一 PWM 周期内没有出现外部事件时，外部事件计数器才会清零。这是一种累积模式，外部事件必须在多个 PWM 周期内都至少出现一次，如下图 19-37 所示。

必须先设定计数器值之后再使能外部事件计数器（设置 EEVACNT[5:0]位之后设置 EVACE 位）。

计数器使能之后，EEVACNT[5:0]位可在任何时间即时改变。EEVACNT[5:0]值在下一复位/翻转事件（根据 EEVARSTM 的配置生成）处生效，或者在软件复位（EEVACRES 置 1）之后生效。

EEVACE 位置 1 之后，不可以修改 EEVASEL[3:0]位。

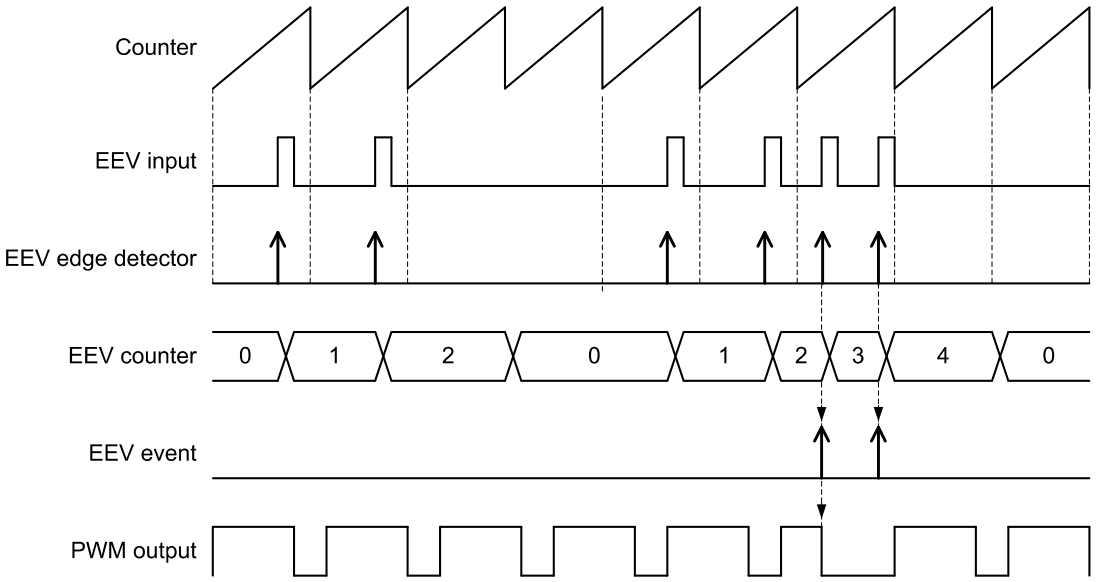


图 19-37 外部事件计数器累积模式（EVEXRSTM＝1，EVEXCNT＝2）

19.4.10 寄存器预装载和更新管理

大部分 HRPWM 寄存器均已缓存，可根据需要进行预装载。这样做通常可避免波形被与有效事件（置位/复位）不同步的寄存器更新更改。

使能预加载模式后，系统访问到的寄存器变为影子寄存器。收到软件发出的更新请求或与事件同步的更新请求之后，影子寄存器的内容会传输到活动寄存器。

默认情况下，MCR 和 PWMxCR0 寄存器中的 PREEN 位会复位，寄存器不会预装载：任何写操作均会直接更新活动寄存器。如果 PREEN 位在定时器运行且预加载已使能时复位，预加载寄存器的内容会直接传输到活动寄存器中。

每个定时单元和主定时器都有自己的 PREEN 位。如果 PREEN 位置 1，仅当发生更新事件时，预加载寄存器才会使能，其内容才会传输到活动寄存器。

如果需要使用预加载功能，可选择两种方式初始化定时器：

- 刚好在定时器初始化结束时使能 PREEN 位，以在定时器使能之前将预装载寄存器内容传输到活动寄存器（通过将 MCEN 和 CENx 位置 1）。
- 在初始化过程中的任何时间使能 PREEN 位，并恰好在启动之前强制进行软件更新。

表 19-14 列出了可预装载的寄存器，并总结了可用更新事件。

表 19-14 HRPWM 可预装载的寄存器和相关的更新源

定时器	可预装载寄存器	预装载使能	更新源
主定时器	MPER.MPER MCNT.MREP MCMPAR.MCMPA MCMPBR.MCMPB MCMPCR.MCMPC MCMPDR.MCMPD	MCR. PREEN	软件更新 周期重复事件 复位/翻转事件
定时器x （x=0~5）	PERxR.PER CNTxR.REP CMPxAR.CMPA CMPxBR.CMPB CMPxCR.CMPC CMPxDR.CMPD DTxR.DTF DTxR.DTR SETxAR.ALL CLRxAR.ALL SETxBR.ALL CLRxBR.ALL RSTxR.ALL	PWMxCR. PREEN	软件更新 周期重复事件 复位翻转事件 硬件更新（来自Master PWM或PWMy）
HRPWM 通用寄存器	ADT0R.ALL ADT1R.ALL ADT2R.ALL ADT3R.ALL ADT4R.ALL ADT5R.ALL ADT6R.ALL ADT7R.ALL	硬件更新（来自主定时器或定时器x） 多个更新源可选，在CR1.USRCx中配置 （所选定时器x的PREEN = 1）	

主定时器有 4 个更新选项：

- 软件更新：将 1 写入 CR2 中的 MSWU 位会立即强制更新寄存器。在这种情况下，会取消任何待定的硬件更新请求。
- 复位事件更新：当主计数器复位或者主计数器翻转时，完成寄存器更新。主定时器默认使能此更新。
- 重复事件更新：当主计数器复位或者主计数器翻转，且主重复计数器等于 0 时，完成寄存器更新。当 MCR 中的 MREPU 位置 1 时，会使能此更新。启用重复事件更新后复位事件更新失效。

主定时器更新事件可以生成中断请求。

各个定时器具有 4 个更新源：

- 软件更新：将 1 写入 CR2 中的 SWUx 位会立即强制更新寄存器。在这种情况下，会取消任何待定的硬件更新请求。
- 复位事件更新：当计数器复位或计数器翻转时，完成寄存器更新。当 PWMxCR0 中的 RSTU 位置 1 时，会使能此更新。
- 重复事件更新：当计数器复位或计数器翻转，且重复计数器等于 0 时，完成寄存器更新。当 PWMxCR0 中的 UPDREP 位置 1 时，会使能此更新。启用重复事件更新后复位事件更新失效。
- 硬件更新：其他定时器或主定时器更新时，完成寄存器更新（例如定时器 0 可与定时器 1 同步更新）。当 PWMxCR0 寄存器中的 MSTU 或 UPDx 位置 1 将启用此更新，适用于需要使用多个定时器的转换器。

来自邻近定时器的更新（当 MUPD，UPD0，UPD1，UPD2，UPD3，UPD4，UPD5 位置 1 时）或来自软件更新的更新（SWUx 位）都可以立即发生，或者与定时器的复位/翻转事件同步发生。这通过 PWMxCR0 寄存器中的 RSYNCU 位实现，如下图 19-38 所示：

- RSYNCU = 0：来自邻近定时器的更新立即发生
- RSYNCU = 1：来自邻近定时器的更新在下一复位/翻转事件处发生。

定时器更新事件可以产生中断请求。

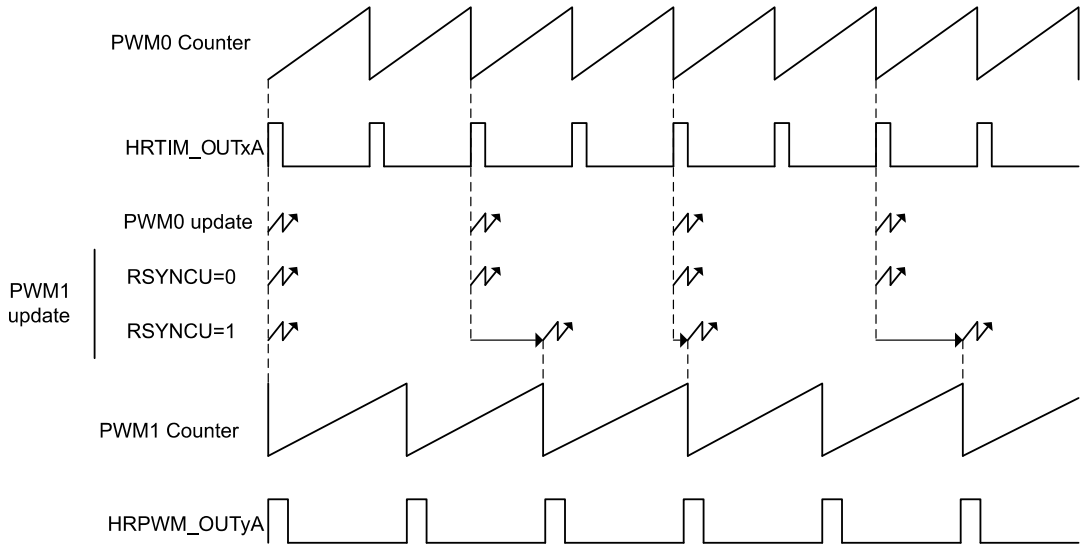


图 19-38 定时器的同步更新（PWM1CR0 的 UPD0=1）

无论选择哪一种更新事件，CR1 寄存器的 MUDIS 和 UDISx 位都可以暂时禁止将预装载寄存器的内容传输至活动寄存器。这样便可以修改并联的定时器中的多个寄存器。这些位复位后，会立即产生常规更新事件。

MUDIS 和 UDISx 位全部分组到同一寄存器中。这样可以同时禁止和恢复更新并联工作的定时器（不需要同步）。

下例为实际用例。第一个电源转换器通过主定时器，定时器 1 和定时器 2 控制。定时器 1 和定时器 2 必须通过主定时器重复事件更新。第二个转换器需要与定时器 0，定时器 3 和定时器 4 同时工作，定时器 3，定时器 4 必须通过定时器 0 重复事件更新。

第一个转换器

在 MCR 中，MREPU 位已置 1：将在主定时器计数器重复周期结束时进行更新。在 PWM1CR0 和 PWM2CR0 中，MSTU 位必须置 1，才能通过主定时器同时更新定时器 1 和定时器 2。

如果必须通过软件调整电源转换器的设定值，则必须在对寄存器执行写访问更新数值之前，将 CR0 寄存器的 MUDIS，UDIS1 和 UDIS2 位置 1。从此刻起，会忽略任何硬件更新请求，并可无风险地访问预装载寄存器，不会将其中的内容传输到活动寄存器中。软件处理结束后，必须将 MUDIS，UDIS1 和 UDIS2 位置 0：发生主定时器重复事件后，会立即将预装载寄存器的内容传输到活动寄存器。

第二个功率转换器

在 PWM0CR0 中，REPU 已位置 1：将在定时器 0 计数器重复周期结束时进行更新。在 PWM3CR0 和 PWM4CR0 中，UPD0 位必须置 1，才能通过定时器 0 同时更新定时器 3 和定时器 4。

如果必须通过软件调整电源转换器的设定值，则必须在对寄存器执行写访问更新数值之前，将 CR0 寄存器的 UDIS0，UDIS3 和 UDIS4 位置 1。从此刻起，会忽略任何硬件更新请求，并可无风险地访问预装载寄存器，不会将其中的内容传输到活动寄存器中。软件处理结束后，必须将 UDIS0，UDIS3 和 UDIS4 位置 0：发生定时器 0 重复事件后，会立即将预装载寄存器的内容传输到活动寄存器。

19.4.11 输出管理

每个定时单元都控制着一对输出，输出有三种工作状态：

- RUN：主要工作状态，在软件启动输出后进入 RUN 状态，在 RUN 模式下，输出根据 CrossBar 的设置变为有效电平或无效电平。
- IDLE：初始工作状态，在复位释放或软件停止输出后进入 IDLE 状态，在 IDLE 模式下，输出维持有效电平或者无效电平。
- FAULT：安全工作状态，在 FAULTx 故障信号产生后进入 FAULT 状态，在 FAULT 模式下，输出维持有效电平、无效电平或高阻态。

输出状态由 OENR 寄存器中的 OENxy 位和 ODISR 寄存器中的 ODISxy 位指示，如表 19-15 所示。

表 19-15 输出状态编程 x = 0~5 , y= A~B

TxyOEN（控制位/状态位）（由软件设置，由硬件清除）	TxyODIS（控制位/状态位）（由软件设置，由硬件清除）	输出工作状态
1	X	运行
0	0	空闲
0	1	故障

OENxy 位与 ODISxy 位既是控制位又是状态位：OENxy 位必须通过软件置 1，使输出进入 RUN 模式。输出恢复到 IDLE 或 FAULT 模式时，OENxy 位会通过硬件清 0。ODISxy 位必须通过软件置 1，使输出返回到 IDLE 模式。OENxy 位为 0 时，ODISxy 位会指示输出是处于 IDLE 还是 FAULT 状态。

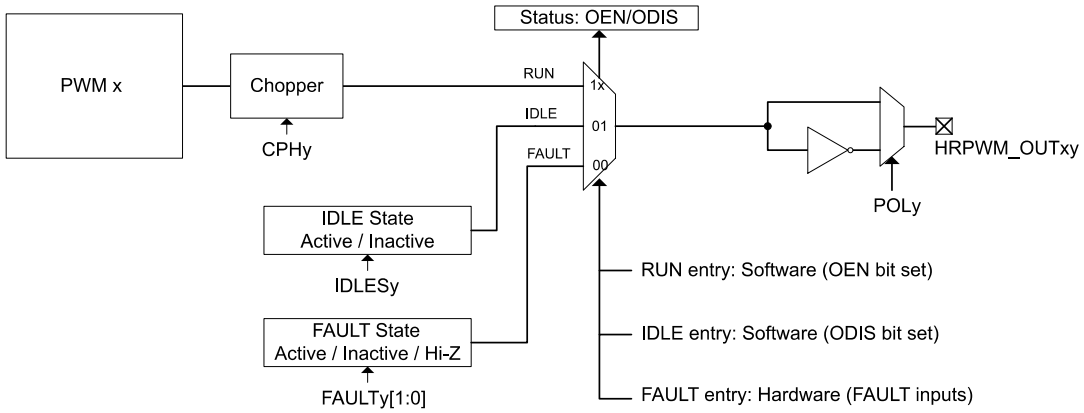
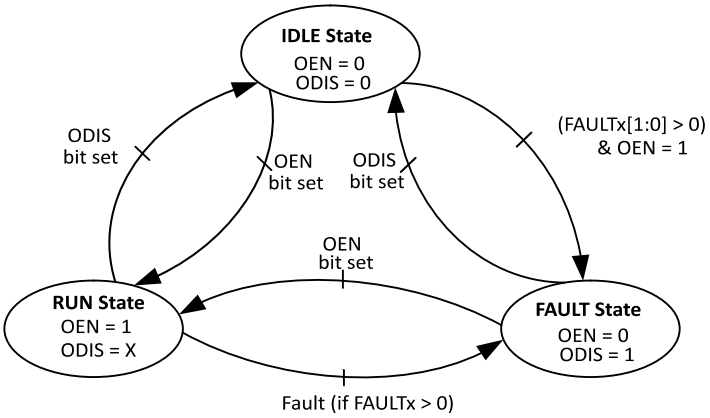


图 19-39 输出管理概览

图 19-40 总结了三种状态对应的位值以及转换的触发方式。任何外部或内部故障源均可触发故障。



Txy prefix is omitted for clarity: (OEN = TxyOEN, ODIS = TxyODIS)

图 19-40 HRPWM 输出状态与转换图

FAULT 和 IDLE 电平定义为有效电平或无效电平。有效电平是指 PWM 输出中引发电源开关闭合的电平，无效电平是指 PWM 输出中引发电源开关关断的电平。

IDLE 状态具有最高优先级：即使 FAULT 条件仍然有效，依然可以实现将 ODIS 位置 1 触发的 FAULT→IDLE 转换。

FAULT 状态的优先级高于 RUN 状态：如果将 OENxy 位置 1 的同时发生了故障事件，则将进入 FAULT 状态。IDLE→FAULT 转换中给出了相应的条件：需要使能故障保护（FAULTx [1：0]位= 01、10、11），且 OEN xy 位在故障有效时置 1，此时由 IDLE 状态进入 FAULT 状态，如图 44 所示。

输出极性可以使用 OUTxR 中的 POLx 位设置。如果 POLx = 0，极性为正（输出高电平有效），如果 POLx = 1，极性为负（输出低电平有效）。实际上，极性的定义取决于要驱动的电源开关（PMOS 与 NMOS）或栅级驱动器的极性。

每路输出 FAULT 状态下的输出电平通过 OUTxR 中的 FAULTx [1：0]位配置，具体如下：

- 00：输出不会进入 FAULT 状态（保持在 RUN 或 IDLE 状态）
- 01：输出在 FAULT 状态下呈现为有效电平
- 10：输出在 FAULT 状态下呈现为无效电平
- 11：输出在 FAULT 状态下呈现为高阻态（浮空状态）。必须通过上拉或下拉电阻在外部强制进入安全状态。

注：一旦输出处于 FAULT 状态，不得更改 FAULTx [1：0]位。

每路输出 IDLE 状态下的输出电平通过 OUTxR 中的 IDLESx 位配置，具体如下：

- 0：输出在 IDLE 状态下呈现为无效电平
- 1：输出在 IDLE 状态下呈现为有效电平

当 OENxy 位置 1 进入 RUN 状态时，输出会立即连接到交错配置矩阵输出。如果定时器时钟停止，电平将为无效电平（HRPWM 复位后）或保持之前电平不变（定时器停止、且输出禁止时）。

在 HRPWM 初始化过程中，在使输出进入 RUN 模式之前，可在 SETxyR 和 RSTxyR 寄存器中使用软件强制输出置位和复位预先设定输出电平。

19.4.12 斩波器

可以在定时单元输出信号上添加高频载波（斩波），以驱动隔离变压器。斩波是在插入极性之前对输出级执行的，如图 19-41 所示，使用 OUTxR 寄存器中的 CHPA 和 CHPB 位分别在 OUTA 和 OUTB 上使能斩波。

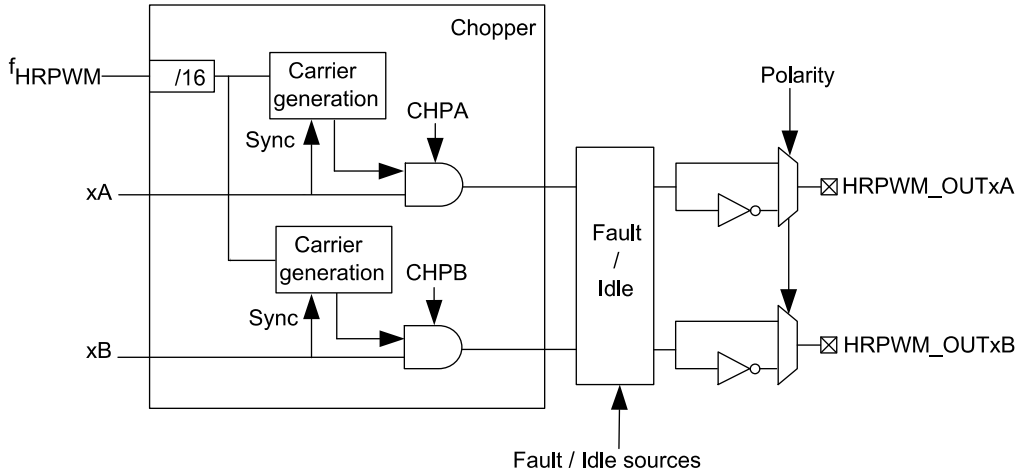


图 19-41 载波频率信号插入

使用 CHPxR 寄存器调整斩波参数，以在斩波开始处定义一个特定脉宽，后跟频率和占空比可编程的载频信号，如图 19-42 所示。

载波频率通过 CARFRQ [3:0]位定义，公式 $F_{CHPFRQ} = f_{HRPWM}/（16 x（CARFRQ [3:0] +1））$，范围从 625 kHz 到 10 MHz（对于 f_{HRPWM} = 160 MHz）。

载波占空比可以通过 CARDTY [2：0]进行调节（步长为 1/8），占空比范围为 0/8 到 7/8。CARDTY [2:0] = 000（占空比= 0/8）时，输出波形仅包含参考波形上升沿之后的启动脉冲，不会添加任何载波。

初始脉冲的脉宽通过 STRPW [3:0]位定义，具体如下：$t_{1STPW} =（STRPW [3:0] +1）x 16 x t_{HRPWM}$，范围从 100 ns 到 1.6 μs（对于 f_{HRPWM} = 160 MHz）。

载波频率参数是根据 f_{HRPWM} 频率定义的，与 CKPSC[2:0]设置无关。

在斩波模式下，载波频率和初始脉宽会通过逻辑与函数与参考波形相结合。初始脉冲结束时会执行同步，以获得重复的信号波形。

斩波信号在输出波形无效状态结束时停止，不会等到当前载波周期结束。因此，斩波信号包含的脉冲可能比编程的脉冲短。

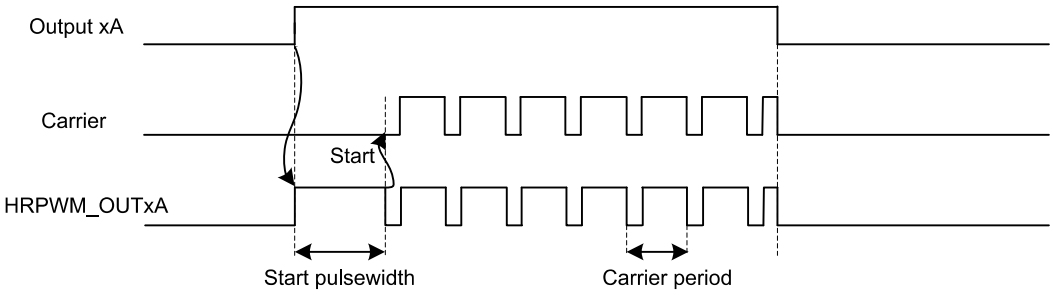


图 19-42 使能斩波模式的 HRPWM 输出

注：必须先将 CHPA 和 CHPB 位置 1，再使能 OENR 寄存器中的 OENxy 位使能输出。

斩波模式有效时（两个 CHPx 位中至少一个置 1），不能修改 CARFRQ[3:0]，CARDTY[2:0]和 STRPW[3:0]位。

19.4.13 故障保护

HRPWM 具有通用的故障保护电路，可在发生异常操作时禁止输出。一旦触发故障，输出便会进入预定义的安全状态。输出通过软件重新使能之前，都会保持该状态。如果是永久故障请求，即使软件尝试重新使能输出，输出也将保持其故障状态，直至故障源消失。

HRPWM 有 6 个故障输入通道，所有通道均可用，并可结合起来用于 6 个定时单元中的每一个，如图 19-43 所示。

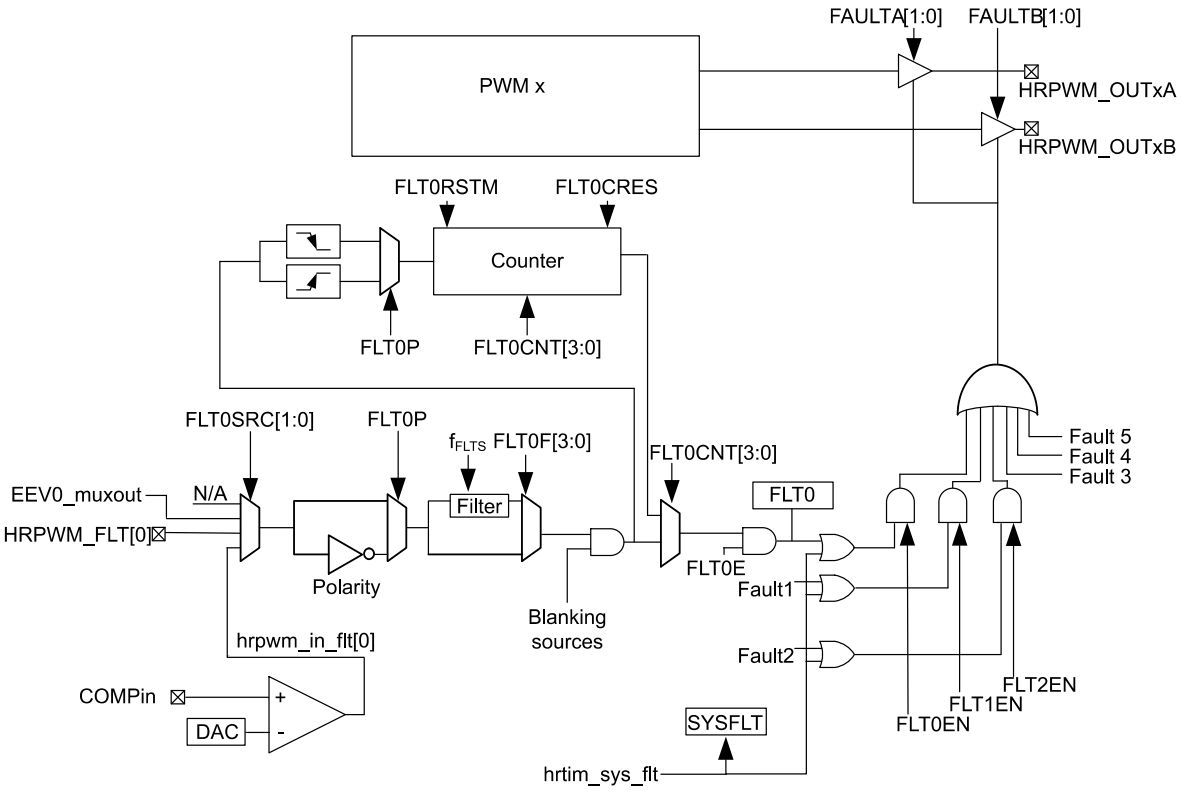


图 19-43 故障保护电路概览（显示了 1 条通道）

在连接到定时单元之前，每条故障通道均可完全通过 FLTINR0、FLTINR1、FLTINR2 和 FLTINR3 寄存器进行配置。FLTxSRC[1:0]位用于选择故障信号源，可以是数字输入或内部事件（内置比较器输出）。

表 19-16 总结了可用于 6 个故障通道的源。

表 19-16 故障事件输入来源

故障事件	FLTxSRC[1:0] = 00	FLTxSRC[1:0] = 01	FLTxSRC[1:0] = 10	FLTxSRC[1:0] = 11
hrpwm_flt_in0[3:0]	HRPWM_FLT0	CMP0_OUT	EEV_mux_out0	N/A
hrpwm_flt_in1[3:0]	HRPWM_FLT1	CMP1_OUT	EEV_mux_out1	N/A
hrpwm_flt_in2[3:0]	HRPWM_FLT2	CMP2_OUT	EEV_mux_out2	N/A
hrpwm_flt_in3[3:0]	HRPWM_FLT3	CMP3_OUT	EEV_mux_out3	N/A
hrpwm_flt_in4[3:0]	HRPWM_FLT4	CMP1_OUT	EEV_mux_out4	N/A
hrpwm_flt_in5[3:0]	HRPWM_FLT5	CMP3_OUT	EEV_mux_out5	N/A

上表 19-16 中提到的 EEVx_muxout 事件，是由 hrpwm_eevx [4：1]输入多路复用器决定（受 EExSRC [1：0]位控制），有关详细信息见图 19-26。

可通过 FLTINRx 寄存器中的 FLTxP 极性位选择信号的极性，以定义有效电平。如果 FLTxP = 0，信号低电平有效；如果 FLTxP = 1，信号高电平有效。

可在极性设置后过滤故障信息。如果 FLTxF[3:0]位设为 0000，则不会对信号进行滤波，信号将独立于 f_{HRPWM} 时钟异步操作。对于所有其他当 FLTxF[3:0]位值，会对信号进行数字滤波。数字滤波器由计数器组成，需要使用 N 个有效采样来验证输出跳变。如果输入值在计数器达到 N 之前发生变化，计数器会复位，跳变会被丢弃（视为伪事件）。如果计数器达到 N，跳变被视为有效，并会作为正确的外部事件进行传输。因此，数字滤波会向正在进行滤波的故障信号添加延迟，延迟时长视采样时钟和滤波器长度（预期的有效采样数）而定。图 19-44 显示了伪故障信号的滤波方式。

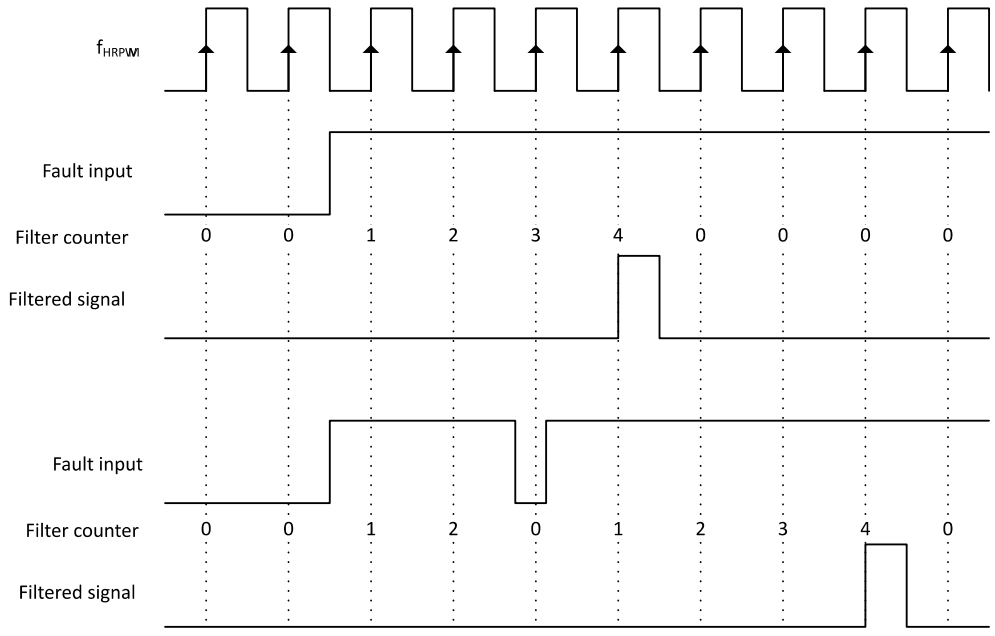


图 19-44 故障信号滤波（FLTxF[3:0] = 0010：fSAMPLING = fHRPWM，N = 4）

滤波周期从 2 个 f_{HRPWM} 时钟周期到 8 个 32 分频的 f_{FLTS} 时钟周期。f_{FLTS} 通过 FLTINR2 寄存器中的 FLTSD[1:0]位定义。

表 19-17 总结了采样速率和滤波长度。考虑到采样引起的不确定性，必须将滤波器长度减去 1 个采样时钟周期的抖动，从而实现有效滤波。

表 19-17 采样速率和滤波长度与 FLTFxF [3:0] 和时钟设置

-	f_{FLTS} vs FLTSD[1:0]				滤波长度 f_{HRPWM} = 160MHz	
FLTxF[3:0]	00	01	10	11	Min	Max
0001	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=6.25ns	N=50ns
0010	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=12.5ns	N=100ns
0011	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=18.75ns	N=150ns
0100	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=25ns	N=200ns
0101	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=31.25ns	N=250ns
0110	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=37.5ns	N=300ns
0111	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=43.75ns	N=350ns
1000	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=50ns	N=400ns
1001	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=56.25ns	N=450ns
1010	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=62.5ns	N=500ns
1011	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=68.75ns	N=550ns
1100	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=75ns	N=600ns
1101	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=81.25ns	N=650ns
1110	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=87.5ns	N=700ns
1111	f_{HRPWM}	$f_{HRPWM}/2$	$f_{HRPWM}/4$	$f_{HRPWM}/8$	N=93.75ns	N=750ns

故障消隐与事件计数

故障输入可以被暂时禁止以消隐伪故障事件，消隐源列在下表 19-18 中。

表 19-18 故障输入消隐事件

-	FLTxBLKS = 0，固定窗口		FLTxBLKS =1，移动窗口	
故障事件	消隐窗口开始	消隐窗口停止	消隐窗口开始	消隐窗口停止
hrpwm_flt_in0[3:0]	PWM0 复位/翻转事件	PWM0 CMPC 事件	PWM0 CMPD 事件	PWM0 CMPC 事件
hrpwm_flt_in1[3:0]	PWM1 复位/翻转事件	PWM1 CMPC 事件	PWM1 CMPD 事件	PWM1 CMPC 事件
hrpwm_flt_in2[3:0]	PWM2 复位/翻转事件	PWM2 CMPC 事件	PWM2 CMPD 事件	PWM2 CMPC 事件
hrpwm_flt_in3[3:0]	PWM3 复位/翻转事件	PWM3 CMPC 事件	PWM3 CMPD 事件	PWM3 CMPC 事件
hrpwm_flt_in4[3:0]	PWM4 复位/翻转事件	PWM4 CMPC 事件	PWM4 CMPD 事件	PWM4 CMPC 事件
hrpwm_flt_in5[3:0]	PWM5 复位/翻转事件	PWM5 CMPC 事件	PWM5 CMPD 事件	PWM5 CMPC 事件

故障计数器也可以过滤伪故障事件，且设置了一种接收条件。

FLTxCNT [3:0]位设置了故障 x 计数器阈值。一个故障仅当事件计数达到 FLTxCNT[3:0]时才会认为有效。

FLTxRSTM 位设置了故障 x 计数器复位模式：

- 0：故障计数器在复位/翻转事件时硬件复位，如下表 24 所示。
- 1：仅当上一计数周期内未发生任何故障事件时，故障计数器在复位/翻转事件时复位，如下图 19-45 所示。

故障计数器可通过软件操作 FLTxCRES 位在任意时刻进行复位。

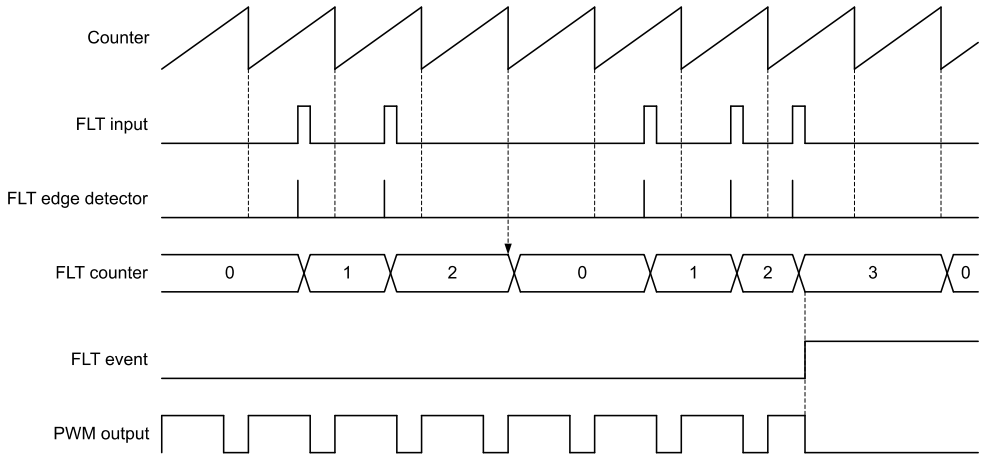


图 19-45 故障计数器累积模式（FLTxRSTM = 1，FLTxCNT [3:0] = 2）

一个给定的 FLTx 故障计数器只能被单一来源复位。表 19-19 描述了给定故障与哪个定时单元相关。这并不影响一个故障事件可以被多个定时器共用（例如，故障计数器使能的 FLT0 可以同时作用于定时器 0、定时器 1 和定时器 2）。

表 19-19 故障事件 0~5 计数复位来源

故障事件输入	故障计数复位来源
hrpwm_flt_in0[3:0]	定时器 0 复位/翻转事件
hrpwm_flt_in1[3:0]	定时器 1 复位/翻转事件
hrpwm_flt_in2[3:0]	定时器 2 复位/翻转事件
hrpwm_flt_in3[3:0]	定时器 3 复位/翻转事件
hrpwm_flt_in4[3:0]	定时器 4 复位/翻转事件
hrpwm_flt_in5[3:0]	定时器 5 复位/翻转事件

系统故障输入（hrpwm_sys_flt）

系统故障由 MCU 提供，对应于来自以下源的系统故障：

- 时钟安全系统
- SRAM 奇偶校验器
- Cortex®-M3-lockup 信号
- PVD 检测器
- FLASH ECC 双重错误检测

此输入会覆盖 FAULT 输入，并禁止所有 FAULTy[1:0] = 01、10、11 的输出。

对于任一定时单元，都会使用 FLTxR 寄存器中的 FLT0EN 至 FLT5EN 位使能 6 条故障通道，并且可同时选择 5 条通道（只要输出受故障机制保护，便会自动使能 sysfault）。这样便可实现：

- 一条故障通道同时禁止多个定时单元
- 多条故障通道进行或运算来禁止单个定时单元

对于每个定时器，故障期间的输出状态是通过 OUTxR 寄存器中的 FaultA[1:0]和 FaultB[1:0]位定义。

19.4.14 将 HRPWM 与其他定时器或 HRPWM 示例同步

HRPWM 可作为主单元（生成同步信号）或从单元（等待触发被同步）来同步多个 HRPWM 实例。此功能也可用于 HRPWM 与其他片外或片上定时器进行同步。同步电路在主定时器内进行控制。

同步输出

本节介绍了如何配置 HRPWM 才能同步外部资源并充当主定时器单元。

可选择四种事件作为要发送到同步输出的源。通过 MCR 寄存器中的 SYNCOUTSRC [1:0]位进行配置，具体如下：

- 00：主定时器启动
  当 MCEN 位置 1、或者定时器在单次模式下达到周期值后重新启动时，会生成该事件。如果计数期间发生复位（CONT 或 RETRG 位置 1）时，也会生成该事件。
- 01：主定时器 CMPA 事件
- 10：定时器 0 启动
  当 CEN0 位置 1 时、或者计数器复位并重新开始计数时，会生成该事件。以下计数器复位事件不会传播到同步输出：连续模式下的计数器翻转、单次不可重触发模式下被丢弃的复位请求。仅当计数期间发生复位时（CONT 或 RETREG 位置 1），才会考虑复位。
- 11：定时器 0 CMPA 事件

MCR 寄存器中的 SYNCOUTEN、SYNCOUTPOL 位指定了同步事件的生成方式。

如果 SYNCOUTEN = 1，则会在 HRPWM_SCOUT 输出引脚上生成同步脉冲。SYNCOUTPOL 位指定了同步信号的极性。如果 SYNCOUTPOL = 0，HRPWM_SCOUT 引脚具有低空闲电平，并会发出长度为 16 个 f_{HRPWM} 时钟周期的正脉冲进行同步。如果 SYNCOUTPOL = 1，空闲电平为高电平，并会生成负脉冲。

注：同步脉冲后会有 16 个 f_{HRPWM} 时钟周期的空闲电平，在此期间，会丢弃任何新的同步请求。因此，最大同步频率为 $f_{HRPWM}/32$。

必须在配置 MCU 输出和计数器使能之前执行同步输出初始化步骤，具体执行步骤如下：

1. 配置 MCR 中的 SYNCOUTEN、SYNCOUTPOL 和 SYNCOUTSRC[1:0]

2. 配置 HRPWM_SCOUT 引脚（参见 GPIO/Pinmux 部分）
3. 使能主定时器或定时器 0 计数器（MCEN 或 CENx 位置 1）

使能同步输入模式并同时启动计数器（使用 SYNCSTRTM / SYNCSTRTx 位）和同步输出模式（SYNCSRC [1:0] = 00 或 10）后，仅当计数器将在运行时启动或复位时，才会生成输出脉冲。如果复位请求将计数器清零而不启动计数器，则不会影响同步输出。

同步输入

HRPWM 可通过外部源进行同步，具体通过对 MCR 寄存器中 SYNCINEN、SYNCINSRC 位进行编程来实现。

如果 SYNCINEN = 0，同步输入功能被禁止。SYNCINSRC 指定同步输入的来源。如果 SYNCINEN = 0，同步输入来源为片上通用定时器（TMR0 TRGO 输出）。如果 SYNCINEN = 1，同步输入来源为 HRPWM_SCIN 输入引脚上的正脉冲。

主定时器或定时单元（MCEN 和/或 CENx 位置 1）使能后，不能更改此位。

HRPWM_SCIN 输入为上升沿有效。定时器行为通过 MCR 和 PWMxCR0 寄存器中的以下位定义（有关详细信息，请参见表 25）：

- 同步启动：输入信号会启动定时器的计数器（SYNCSTRTM 和/或 SYNCSTRTx 位置 1）。CENx（MCEN）位必须置 1 才能使能定时器并使计数器准备号启动。在连续模式下，计数器在收到同步信号之前不会启动。
- 同步复位：输入信号会复位计数器（SYNCRSTM 和/或 SYNCRSTx 位置 1）。该事件会像其他复位事件一样使重复计数器递减。

仅当相关计数器使能后（MCEN 或 CENx 位置 1），才会考虑同步事件。同步事件会触发 SYNCIN 中断。

注：如果当前计数器值大于有效周期值，同步启动事件会复位计数器。

同步事件的作用取决于定时器工作模式，参见表 19-20 中总结的内容。

表 19-20 同步事件的作用与定时器工作模式

PWM 工作模式	SYNC RSTx	SYNC STRTx	同步复位或同步启动的行为
单次模式（不可再触发）	0	1	当计数器停止时启动事件在下列情况下有效: – MCEN 或者 CENx 位置 1 – 计数到达周期值 当计数器在周期值停止时发生启动事件，启动事件会复位计数器。复位请求会清除计数器但不会启动计数器（计数器只能够通过同步事件重新启动）。在计数期间任何复位事件都会被忽略（正如在通常的不可重复触发模式下）

	1	x	同步复位事件可以启动计时器计数。当计数器停止时，复位事件在下列情况下有效： - MCEN 或者 CENx 位置 1 - 计数到达周期值 当有多个复位请求（包括同步复位以及内部复位）时，只考虑第一个复位请求。
单次模式（可再触发）	0	1	仅在计数器没有启动或者到达周期值之后，计数器的同步启动才会生效。在计数器启动之后任何同步事件都不会生效。 当计数器在周期值停止时发生启动事件，启动事件会复位计数器。复位请求会清除计数器但不会启动计数器（计数器只能够通过同步事件重新启动）。在计数期间任何复位事件不会被忽略（正如在通常的可重复触发模式下）
	1	x	来自 HRPWM_SCIN 的复位请求有效，正如 HRPWM 任意一个从内部事件中产生的复位请求一样，这些复位请求可以启动或重新启动计时器计数。 当有多个复位请求时，只考虑第一个复位请求。
连续模式	0	1	计数器使能（MCEN 或者 CENx 置 1）之后会等待同步事件，当同步事件到来后计时器启动计数。计数器启动之后的同步事件都是无效的（计数器只能够被同步事件启动）。复位请求会清除计数器，但不会启动计数器。
	1	x	来自 HRPWM_SCIN 的复位请求有效，正如 HRPWM 任意一个从内部事件中产生的复位请求一样，这些复位请求可以启动或重新启动计时器计数。 当有多个复位请求时，只考虑第一个复位请求。

图 19-46 显示了单次模式下如何进行同步启动。

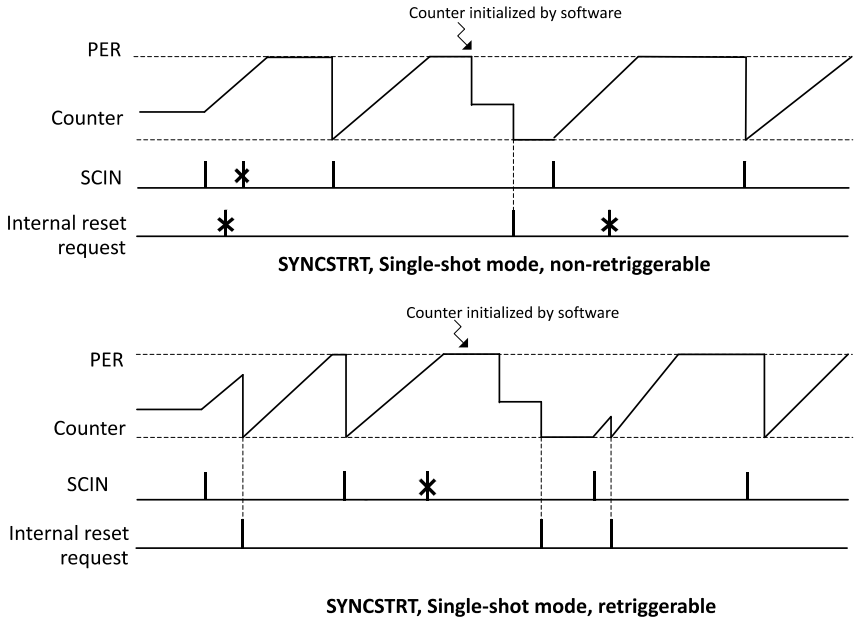


图 19-46 同步启动模式下计数器的行为

19.4.15 ADC/DAC 触发事件

ADC/DAC 转换器可由主定时器和 6 个定时单元触发。

HRPWM 总共支持 8 个 ADC 触发事件，可通过 8 个独立的触发事件同时启动两路 ADC 的常规转换序列和注入转换序列，或者启动 DAC 三角波生成和锯齿波生成。8 个触发事件对应在 ADT0R~ADT7R 寄存器中，最多可为每路触发输出合并 32 个事件（逻辑或运算），如图 19-47 所示。ADC 触发事件 0/2/4/6 和 1/3/5/7 将使用一组相同的源。

外部事件可用作触发信号，这类触发信号会在 EECRx 寄存器中定义的调节结束后立即获取，并且与 EEFxR1 和 EEFxR2 寄存器的配置无关。

可通过同时选择多个源的方式在单个开关周期内进行多次触发。典型案例是非重叠多相转换器，可通过单个 ADC 触发事件对所有相位进行连续采样。

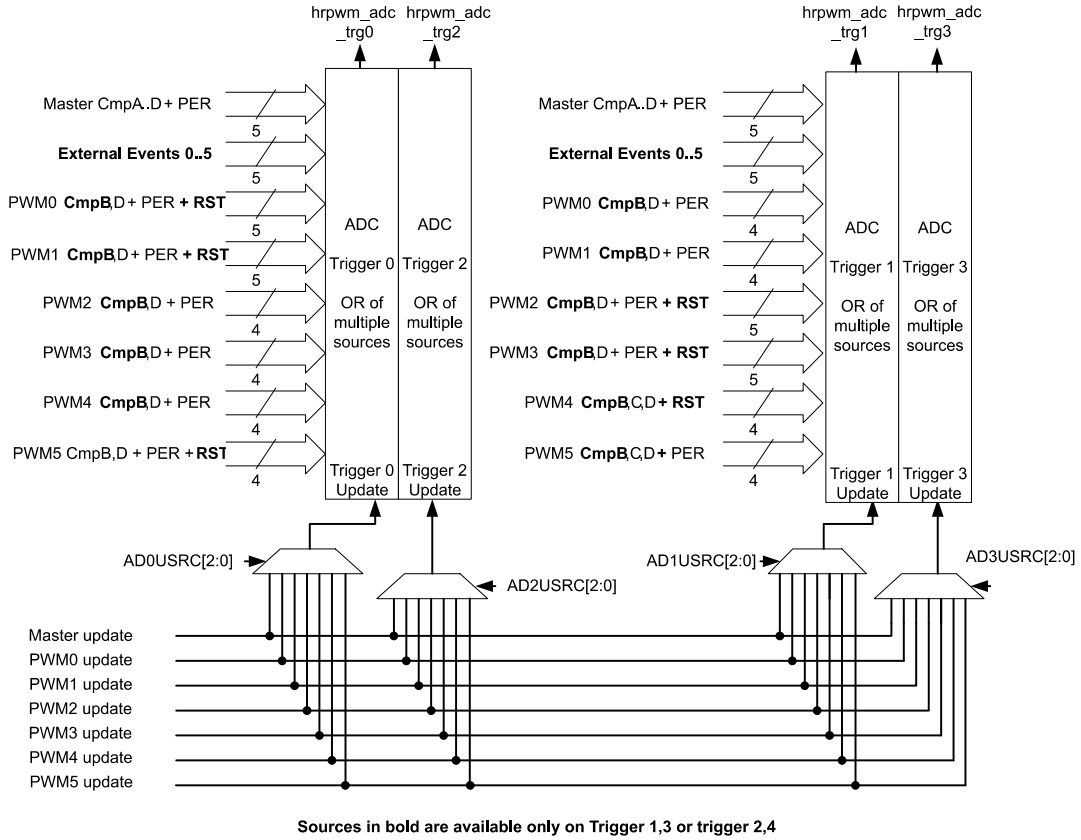


图 19-47 ADC 触发事件概览

ADT0R 到 ADC7R 寄存器会预装载，并可与相关定时器同步更新。更新源是通过 CR1 寄存器中的 USRCx[2:0]位定义的。

举例来说，如果 ADC 触发 1 输出定时器 0 CMPB 事件（ADT0R = 0x00000400），ADT0R 通常将与定时器 0 同步更新（USRC0[2:0] = 001）。

如果源定时器中禁止预装载（PREEN 位置 0），ADTxR 寄存器也不会预装载，写访问会立即更新触发源。

ADC 降采样

降采样单元可以降低 ADC 触发事件的频率，如下图 19-48 所示。

每个 ADC 触发事件的频率可以使用 ADPSR 寄存器中的 PSCx[4:0]位分别进行调整。

在中心对齐模式下，ADC 触发事件频率还取决于各个源定时器中控制的 ADROM[1:0]位，如图 19-49 所示。ADROM[1:0]位可用于任何可以触发 ADC 的事件，包括复位事件，翻转（周期）事件和比较事件：

- ADROM[1:0]＝00：上/下计数阶段都会产生事件
- ADROM[1:0]＝01：下计数阶段会产生事件
- ADROM[1:0]＝10：上计数阶段会产生事件

ADC 降采样寄存器是可预装载寄存器，可以在不停止定时器工作的情况下即时更新。

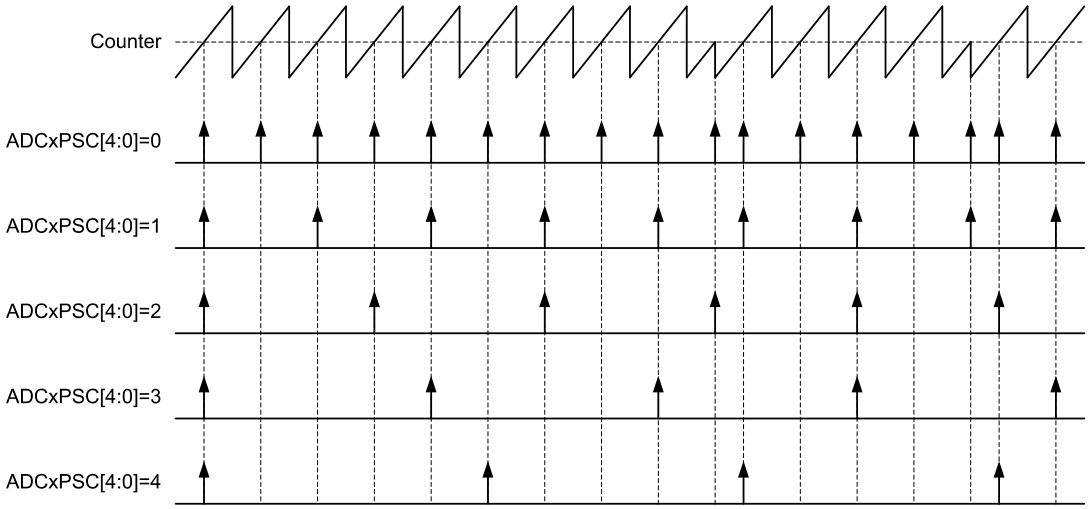


图 19-48 上计数模式下的 ADC 触发事件减采样

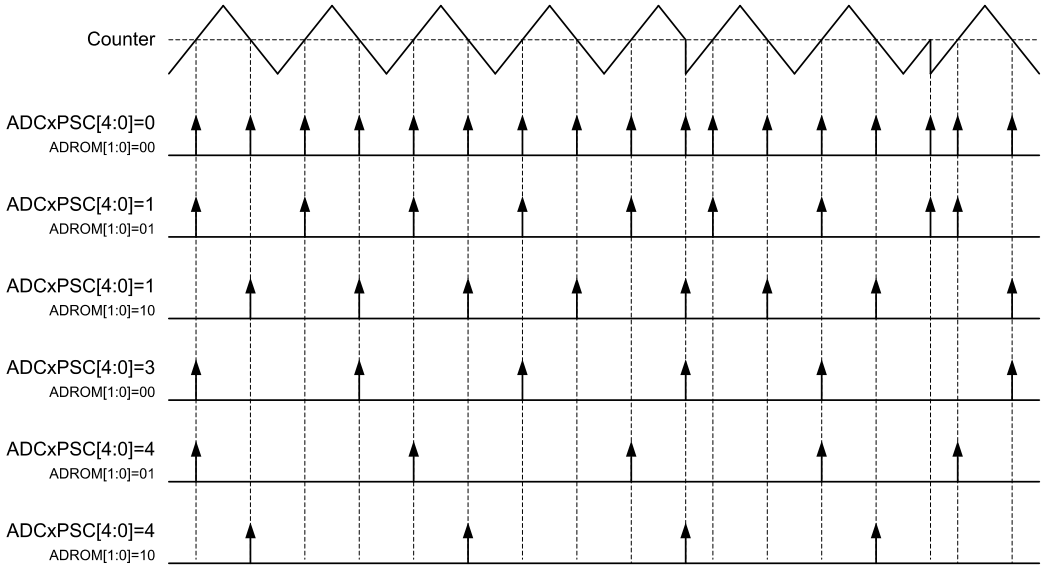


图 19-49 上/下计数模式下的 ADC 触发事件减采样

DAC 锯齿波触发

斜坡补偿技术以及迟滞控制可以使用 HRPWM 和 DAC 锯齿波发生器实现。基本原理是通过 DAC 产生一个递减的与 PWM 周期同步的锯齿波，或者一个与 PWM 信号同步的方波。

此模式通过 PWMxCR1 寄存器中的 DCDE 位使能。定时器开始工作后（CENx 位置 1），这一位不可以更改。

此模式使用了两个触发事件，如下图 19-50 所示：

- hrpwm _reset_dac_trg 产生 DAC 复位/更新事件
- hrpwm_step _dac_trg 产生 DAC 步进事件

PWMxCR1 寄存器中的 DCDR 位决定 hrpwm_reset_dac_trg 触发事件何时生成：

- DCDR = 1：触发事件在计数器复位或翻转事件处产生
- DCDR = 0：触发事件在输出 A 置位事件处产生

注：当 DCDE 位复位时，DCDR 位无意义（DAC 锯齿波触发被禁止）

PWMxCR1 寄存器中的 DCDS 位决定 hrpwm_step_dac_trg 触发事件何时生成：

- DCDS = 1：CMPB 事件产生触发事件
- DCDS = 0：OUTA 复位事件产生触发事件

DCDR 位和 DCDS 位可以覆盖以下的应用场景：

- 边沿对齐的斜坡补偿（DCDR = DCDS = 1）：DAC 的锯齿波在 PWM 周期开始处启动，且在 PWM 周期内会产生多个触发事件
- 中心对齐的斜坡补偿（DCDR = 0 DCDS = 1）：DAC 的锯齿波在输出置位事件处启动，且在 PWM 周期内会产生多个触发事件
- 迟滞控制器：当输出状态更改时，每个周期必须更改 DAC 值两次。每个 PWM 周期产生 2 个触发事件。在边沿对齐模式（DCDR = 1，DCDS = 0）时，触发事件将在计数器复位或翻转时产生。在居中对齐模式下（DCDR = 0，DCDS = 0），触发事件将在输出置位时产生。

当 DCDE 被置位且 DCDS 被复位时，CMPD 会进入一种特定的工作模式。每次生成比较值匹配事件后，有效的比较值就会自动更新，这样触发事件可以周期性地重复产生，其中周期为 CMPD 值，如图 54 所示。

注：CMPD 值可以即时更改，新的值会在下一个比较值匹配事件处更新。

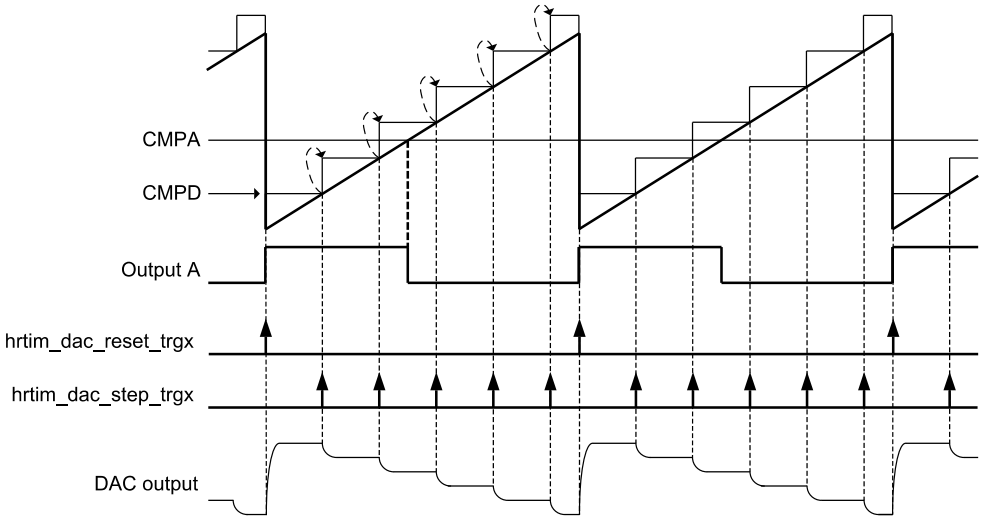
表 19-21 给出了一个例子，在一个 PWM 周期内生成 6 个触发事件。表格说明了有必要对除法结果四舍五入取上舍入值。

考虑一个计数器周期 PWMxPER =8192。将 8192 除以 6 得到 1365.33。

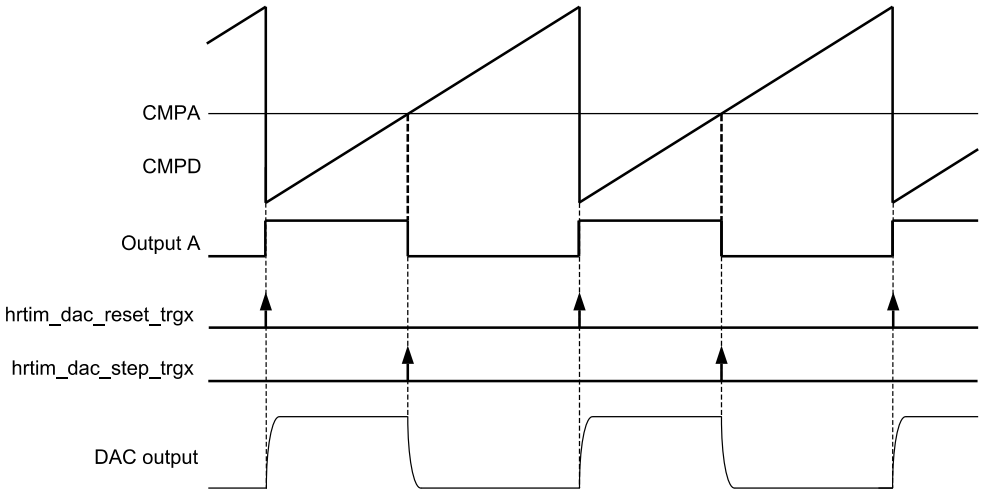
- 下舍入值：1365：会生成 7 个触发事件，第 6 个触发事件和第 7 个触发事件非常接近（分别对应计数值= 8190 和 8192）
- 上舍入值：1366：会生成 6 个触发事件，hrpwm_step_dac_trg 的第 6 个触发事件（对应计数值= 8192）被舍弃，因为计数器从 8192 到 0 发生翻转。

表 19-21 DAC 锯齿波触发事件示例

-	CMPD = 1365	reset_dac_trg	step_dac_trg	CMPD = 1366	reset_dac_trg	step_dac_trg
Counter value	1365	-	1	1366	-	1
	2730	-	2	2732	-	2
	4095	-	3	4098	-	3
	5460	-	4	5464	-	4
	6825	-	5	6830	-	5
	8190	-	6	8192	6	-
	8192	7	-	1366	-	1
	1365	-	1	2732	-	2
	...	-	-	...	-	-



DCDR = 1 (reset on roll-over), DCDT = 1 (step trigger on Compare D)



DCDR = 0 (reset on output A set), DCDT = 0 (step trigger on output A clr)

图 19-50 用于斜坡补偿的 DAC 锯齿波触发事件

19.4.16 HRPWM 中断

主定时器可生成 8 个中断：

- 主定时器寄存器更新
- 接收同步事件
- 主定时器周期事件
- 主定时器重复事件
- 主定时器比较 A~比较 D 事件

每个定时单元可生成 12 个中断：

- 计数器复位事件
- 计数器周期（翻转）事件
- 输出 1 和输出 2 复位（从有效电平跳变为无效电平）
- 输出 1 和输出 2 置位（从无效电平跳变为有效电平）
- 定时单元寄存器更新
- 定时单元重复事件
- 定时单元比较 A~比较 D 事件

整个 HRPWM 生成 7 个故障中断：

- 系统故障事件
- 故障 0~故障 5 事件（不考虑定时单元的因素）

中断请求会分到 8 个向量组中，具体如下：

- hrpwm_mst_int：主定时器中断
- hrpwm_slv0_int：定时器 0 中断
- hrpwm_slv1_int：定时器 1 中断
- hrpwm_slv2_int：定时器 2 中断
- hrpwm_slv3_int：定时器 3 中断
- hrpwm_slv4_int：定时器 4 中断
- hrpwm_slv5_int：定时器 5 中断
- hrpwm_flt_int：HRPWM 故障中断

表 19-22 总结了中断请求及其映射以及相关控制和状态位。

表 19-22 HRPWM 中断汇总

中断向量	中断事件	中断标志	使能控制位	标志清除位
hrpwm_mst_int	同步事件接收事件	SYNC	SYNCIE	SYNC
	主寄存器更新事件	MUPD	MUPDIE	MUPD
	主周期重复事件	MREP	MREPIE	MREP
	主周期事件	MPER	MPERIE	MPER
	主比较 A～D 事件	MCMPA	MCMPAIE	MCMPA
		MCMPB	MCMPBIE	MCMPB
		MCMPC	MCMPCIE	MCMPC
		MCMPD	MCMPDIE	MCMPD
hrpwm_slv0_int hrpwm_slv1_int hrpwm_slv2_int hrpwm_slv3_int hrpwm_slv4_int hrpwm_slv5_int	周期重复事件	REP	REPIE	REP
	计数复位事件	RST	RSTIE	RST
	输出 A 和输出 B 复位事件(有效电平到无效电平)	CLRA	CLRAIE	CLRA
		CLRB	CLRBIE	CLRB
	输出 A 和输出 B 置位事件(无效电平到有效电平)	SETA	SETAIE	SETA
		SETB	SETBIE	SETB
	寄存器更新事件	UPD	UPDIE	UPD
	周期翻转事件	PER	PERIE	PER
	比较 A～D 事件	CMPA	CMPAIE	CMPA
		CMPB	CMPBIE	CMPB
		CMPC	CMPCIE	CMPC
		CMPD	CMPDIE	CMPD
hrpwm_flt_int	系统故障事件	SYSFLT	SYSFLTIE	SYSFLT
	故障 0～5 事件	FLT0	FLT0IE	FLT0
		FLT1	FLT1IE	FLT1
		FLT2	FLT2IE	FLT2
		FLT3	FLT3IE	FLT3
		FLT4	FLT4IE	FLT4
		FLT5	FLT5IE	FLT5

19.5 寄存器描述

19.5.1 寄存器列表

Name	Offset	Width	Description
HRPWM_MCR	0x00	32bits	HRPWM Master PWM Control Register
HRPWM_MISR	0x08	32bits	HRPWM Master PWM Interrupt Status Register
HRPWM_MIER	0x0C	32bits	HRPWM Master PWM Interrupt Enable Register
HRPWM_MCNTR	0x10	32bits	HRPWM Master PWM Counter Register
HRPWM_MPER	0x14	32bits	HRPWM Master PWM Period Register
HRPWM_MCMPAR	0x18	32bits	HRPWM Master PWM Compare A Register
HRPWM_MCMPBR	0x1C	32bits	HRPWM Master PWM Compare B Register
HRPWM_MCMPCR	0x20	32bits	HRPWM Master PWM Compare C Register
HRPWM_MCMPDR	0x24	32bits	HRPWM Master PWM Compare D Register
HRPWM_PWMxCR0	0x80 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Control Register0
HRPWM_PWMxCR1	0x84 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Control Register1
HRPWM_PWMxISR	0x88 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Interrupt Status Register
HRPWM_PWMxIER	0x8C + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Interrupt Enable Register
HRPWM_CNTxR	0x90 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Counter Register
HRPWM_PERxR	0x94 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Period Register
HRPWM_CMPAxR	0x98 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Compare A Register
HRPWM_CMPBxR	0x9C + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Compare B Register
HRPWM_CMPCxR	0xA0 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Compare C Register
HRPWM_CMPDxR	0xA4 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Compare D Register
HRPWM_DTxR	0xA8 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx DeadTime Register
HRPWM_SETxAR	0xAC + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Output A Set Register
HRPWM_CLRxAR	0xB0 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Output A Clear Register
HRPWM_SETxBR	0xB4 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Output B Set Register
HRPWM_CLRxBR	0xB8 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Output B Clear Register
HRPWM_EEFxR0	0xBC + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx External Event Register0
HRPWM_EEFxR1	0xC0 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx External Event Register1
HRPWM_RSTxR	0xC4 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Reset Register
HRPWM_CHPxR	0xC8 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Chopper Register
HRPWM_OUTxR	0xCC + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Output Register
HRPWM_FLTxR	0xD0 + 0x80 * x (x = 0~5)	32bits	HRPWM PWMx Fault Register
HRPWM_CR0	0x380	32bits	HRPWM Control Register0
HRPWM_CR1	0x384	32bits	HRPWM Control Register 1
HRPWM_CR2	0x388	32bits	HRPWM Control Register 2

HRPWM_ISR	0x38C	32bits	HRPWM Interrupt Status Register
HRPWM_IER	0x390	32bits	HRPWM Interrupt Enable Register
HRPWM_OENR	0x394	32bits	HRPWM Output Enable Register
HRPWM_ODISR	0x398	32bits	HRPWM Output Disable Register
HRPWM_EECR0	0x39C	32bits	HRPWM External Event Register0
HRPWM_EECR1	0x3A0	32bits	HRPWM External Event Register1
HRPWM_EECR2	0x3A4	32bits	HRPWM External Event Register2
HRPWM_ADT0R	0x3A8	32bits	HRPWM ADC Trigger 0 Register
HRPWM_ADT1R	0x3AC	32bits	HRPWM ADC Trigger 1 Register
HRPWM_ADT2R	0x3B0	32bits	HRPWM ADC Trigger 2 Register
HRPWM_ADT3R	0x3B4	32bits	HRPWM ADC Trigger 3 Register
HRPWM_ADT4R	0x3B8	32bits	HRPWM ADC Trigger 4 Register
HRPWM_ADT5R	0x3BC	32bits	HRPWM ADC Trigger 5 Register
HRPWM_ADT6R	0x3C0	32bits	HRPWM ADC Trigger 6 Register
HRPWM_ADT7R	0x3C4	32bits	HRPWM ADC Trigger 7 Register
HRPWM_ADPSR	0x3C8	32bits	HRPWM ADC Trigger Post Scaler Register
HRPWM_DLLCR	0x3CC	32bits	HRPWM DLL Control Register
HRPWM_FLTINR0	0x3D0	32bits	HRPWM Fault Input Register0
HRPWM_FLTINR1	0x3D4	32bits	HRPWM Fault Input Register 1
HRPWM_FLTINR2	0x3D8	32bits	HRPWM Fault Input Register 2
HRPWM_FLTINR3	0x3DC	32bits	HRPWM Fault Input Register 3

19.5.2 寄存器详细描述

19.5.2.1 HRPWM Master PWM Control Register (HRPWM_MCR)

- **Name**：HRPWM Master PWM Control Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-26]	Reserved	Reserved	Reserved	Reserved
[25]	PREEN	R/W	0x0	预加载使能 此位定义了是否开启预加载功能，定义了寄存器写访问直接作用于工作寄存器还是作用于影子寄存器。 0：预加载关闭，写访问直接作用于工作寄存器 1：预加载开启，写访问作用于影子寄存器
[24]	MREPU	R/W	0x0	Master PWM Repetition 更新 此位定义了 Master PWM 更新是否由 Master PWM Repetition 事件触发。 0：Master PWM Repetition 事件不触发 Master PWM 更新 1：Master PWM Repetition 事件触发 Master PWM 更新
[23]	Reserved	Reserved	Reserved	Reserved

[22]	CEN5	R/W	0x0	PWM5 使能 此位定义是否使能 PWM5。 0：PWM5 不使能 1：PWM5 使能
[21]	CEN4	R/W	0x0	PWM4 使能 此位定义是否使能 PWM4。 0：PWM4 不使能 1：PWM4 使能
[20]	CEN3	R/W	0x0	PWM3 使能 此位定义是否使能 PWM3。 0：PWM3 不使能 1：PWM3 使能
[19]	CEN2	R/W	0x0	PWM2 使能 此位定义是否使能 PWM2。 0：PWM2 不使能 1：PWM2 使能
[18]	CEN1	R/W	0x0	PWM1 使能 此位定义是否使能 PWM1。 0：PWM1 不使能 1：PWM1 使能
[17]	CEN0	R/W	0x0	PWM0 使能 此位定义是否使能 PWM0。 0：PWM0 不使能 1：PWM0 使能
[16]	MCEN	R/W	0x0	Master PWM 使能 此位定义是否使能 Master PWM。 0：Master PWM 不使能 1：Master PWM 使能
[15-14]	SYNC OUTSRC	R/W	0x0	同步事件输出来源 此位定义了同步事件输出的来源。 00： Master PWM Start 01： Master PWM Compare A 10：PWM0 Start/Reset 11：PWM0 Compare A
[13]	SYNC OUTEN	R/W	0x0	同步事件输出使能 此位定义了是否使能同步事件。 0：同步事件输出不使能 1：同步事件输出使能
[12]	SYNC OUTPOL	R/W	0x0	同步事件输出极性 此位定义了同步事件输出的极性。 0：同步事件输出高有效（输出一个高电平脉冲，16 个 f_{HRPWM} 时钟周期） 1：同步事件输出低有效（输出一个低电平脉冲，16 个 f_{HRPWM} 时钟周期）

[11]	SYNC STRTM	R/W	0x0	同步事件启动 Master PWM 此位定义了同步事件后的 Master PWM 行为。 0：同步事件不启动 Master PWM 1：同步事件启动 Master PWM
[10]	SYNC RSTM	R/W	0x0	同步事件复位 Master PWM 此位定义了同步事件后的 Master PWM 行为。 0：同步事件不复位 Master PWM 1：同步事件复位 Master PWM
[9]	SYNC INEN	R/W	0x0	同步事件输入使能 此位定义了是否使能同步事件输入用以触发 HRPWM。 0：同步事件输入不使能 1：同步事件输入使能，上升沿有效
[8]	SYNC INSRC	R/W	0x0	同步事件输入来源 此位定义了同步事件输入的来源。 0：TIM0_TRGO 1：HRPWM_SCIN
[7-6]	INTLVD	R/W	0x0	Master PWM Interleaved 模式 此位配置 Master PWM Interleaved 模式。 ● 1/3 Interleaved 模式下 Compare A 数值自动被设为 Period 数值的 1/3，Compare B 数值自动被设为 Period 数值的 2/3。 ● 1/4 Interleaved 模式下 Compare A 数值自动被设为 Period 数值的 1/4，Compare B 数值自动被设为 Period 数值的 2/4，Compare C 数值自动被设为 Period 数值的 3/4。 00：Interleaved 模式关闭 01：1/3 Interleaved 模式开启 10：1/4 Interleaved 模式开启 11：Interleaved 模式关闭
[5]	HALF	R/W	0x0	Master PWM Half 模式 此位使能 Master PWM Half 模式，Half 模式下 Compare A 数值自动被设为 Period 数值的 1/2。 0：Half 模式关闭 1：Half 模式开启
[4]	RETRG	R/W	0x0	Master PWM 可重触发模式 此位定义了 PWMx 单次模式下的特性。 0：Master PWM 不可重触发，在计数停止后才可以进行计数复位 1：Master PWM 可重触发，无论计数状态如何都可以进行计数复位
[3]	CONT	R/W	0x0	Master PWM 连续模式 此位定义了 PWMx 计数模式。 0：Master PWM 工作在单次模式，计数到达 Period 值后停止 1：Master PWM 工作在连续模式，计数到达 Period 值后翻转到 0
[2-0]	CKPSC	R/W	0x0	Master PWM 时钟预分频比 此位定义了 Master PWM 高分辨率时钟预分频比。 Master PWM 等效时钟为 f_{HRCK} / 2 CKPSC。

19.5.2.2 HRPWM Master PWM Interrupt Status Register (HRPWM_MISR)

- **Name**：HRPWM Master PWM Interrupt Status Register
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7]	MREP	R/W1C	0x0	Master Repetition 中断标志 写 1 清除中断标志。 0：Repetition 中断标志未产生 1：Repetition 中断标志产生
[6]	MUPD	R/W1C	0x0	Master Update 中断标志 写 1 清除中断标志。 0：Update 中断标志未产生 1：Update 中断标志产生
[5]	SYNC	R/W1C	0x0	Sync Input 中断标志 写 1 清除中断标志。 0：Sync Input 中断标志未产生 1：Sync Input 中断标志产生
[4]	MPER	R/W1C	0x0	Master Period 中断标志 写 1 清除中断标志。 0：Period 中断标志未产生 1：Period 中断标志产生
[3]	MCMPD	R/W1C	0x0	Master Compare D 中断标志 写 1 清除中断标志。 0：Compare D 中断标志未产生 1：Compare D 中断标志产生
[2]	MCMPC	R/W1C	0x0	Master Compare C 中断标志 写 1 清除中断标志。 0：Compare C 中断标志未产生 1：Compare C 中断标志产生
[1]	MCMPB	R/W1C	0x0	Master Compare B 中断标志 写 1 清除中断标志。 0：Compare B 中断标志未产生 1：Compare B 中断标志产生
[0]	MCMPA	R/W1C	0x0	Master Compare A 中断标志 写 1 清除中断标志。 0：Compare A 中断标志未产生 1：Compare A 中断标志产生

19.5.2.3 HRPWM Master PWM Interrupt Enable Register (HRPWM_MIER)

- **Name**：HRPWM Master PWM Interrupt Enable Register
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7]	MREPIE	R/W	0x0	Master Repetition 中断使能 0：Repetition 中断不使能 1：Repetition 中断使能
[6]	MUPDIE	R/W	0x0	Master Update 中断使能 0：Update 中断不使能 1：Update 中断使能
[5]	SYNCIE	R/W	0x0	Sync Input 中断使能 0：Sync Input 中断不使能 1：Sync Input 中断使能
[4]	MPERIE	R/W	0x0	Master Period 中断使能 0：Period 中断不使能 1：Period 中断使能
[3]	MCMPDIE	R/W	0x0	位 3：Master Compare D 中断使能 0：Compare D 中断不使能 1：Compare D 中断使能
[2]	MCMPCIE	R/W	0x0	Master Compare C 中断使能 0：Compare C 中断不使能 1：Compare C 中断使能
[1]	MCMPBIE	R/W	0x0	Master Compare B 中断使能 0：Compare B 中断不使能 1：Compare B 中断使能
[0]	MCMPAIE	R/W	0x0	Master CompareA 中断使能 0：Compare A 中断不使能 1：Compare A 中断使能

19.5.2.4 HRPWM Master PWM Counter Register (HRPWM_MCNTR)

- **Name**：HRPWM Master PWM Counter Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	MREP	R/W	0x0	Master PWM Repetition Period 数值 此位保持 Master PWM Repetition Period 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。
[23-20]	Reserved	Reserved	Reserved	Reserved
[19]	CNTWR	R/WAC	0x0	对此位写 1 的同时写 MCNT，可以将 MCNT 写入到 Master PWM Counter，写入之后此位自动清零
[18]	CNTRD	R/WAC	0x0	对此位写 1，可以将 Master PWM Counter 当前值读取到 MCNT，读取之后此位自动清零
[17-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	MCNT	R/W	0x0	Master PWM Counter 数值 此位保持 Master PWM Counter 数值，当高精度模式（CKPSC＜5）开启时，此位的低有效位无意义，这些位无法被写入，读取时返回为 0。

19.5.2.5 HRPWM Master PWM Period Register (HRPWM_MPER)

- **Name**：HRPWM Master PWM Period Register
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	MPER	R/W	0x0	Master PWM Period 数值 此位保持 Master PWM Period 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Period 数值必须大于或等于 3 个 f_{HRPWM} 周期. ● 若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30 ● 若 CKPSC＝2 为 0x18，Period 最大值为 0xFFDF

19.5.2.6 HRPWM Master PWM Compare A Register (HRPWM_MCMPAR)

- **Name**：HRPWM Master PWM Compare A Register
- **Size**：32bits
- **Offset**：0x18
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	MCMPA	R/W	0x0	Master PWM Compare A 数值 此位保持 Master PWM Compare A 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Compare A 数值必须大于或等于 3 个 f_{HRPWM} 周期，若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30，若 CKPSC＝2 为 0x18。

19.5.2.7 HRPWM Master PWM Compare B Register (HRPWM_MCMPBR)

- **Name**：HRPWM Master PWM Compare B Register
- **Size**：32bits
- **Offset**：0x1C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	MCMPB	R/W	0x0	Master PWM Compare B 数值 此位保持 Master PWM Compare B 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Compare B 数值必须大于或等于 3 个 f_{HRPWM} 周期，若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30，若 CKPSC＝2 为 0x18。

19.5.2.8 HRPWM Master PWM Compare C Register (HRPWM_MCMPCR)

- **Name**：HRPWM Master PWM Compare C Register
- **Size**：32bits
- **Offset**：0x20
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	MCMPC	R/W	0x0	Master PWM Compare C 数值 此位保持 Master PWM Compare C 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Compare C 数值必须大于或等于 3 个 f_{HRPWM} 周期，若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30，若 CKPSC＝2 为 0x18。

19.5.2.9 HRPWM Master PWM Compare D Register (HRPWM_MCMPDR)

- **Name**：HRPWM Master PWM Compare D Register
- **Size**：32bits
- **Offset**：0x24
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	MCMPD	R/W	0x0	Master PWM Compare D 数值 此位保持 Master PWM Compare D 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Compare D 数值必须大于或等于 3 个 f_{HRPWM} 周期，若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30，若 CKPSC＝2 为 0x18。

19.5.2.10 HRPWM PWMx Control Register0 (HRPWM_PWMxCR0)

- **Name**：HRPWM PWMx Control Register0
- **Size**：32bits
- **Offset**：0x80 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-26]	Reserved	Reserved	Reserved	Reserved
[25]	PREEN	R/W	0x0	预加载使能 此位定义了是否开启预加载功能，定义了寄存器写访问直接作用于工作寄存器还是作用于影子寄存器。 0：预加载关闭，写访问直接作用于工作寄存器 1：预加载开启，写访问作用于影子寄存器
[24]	UPDREP	R/W	0x0	PWMx Repetition 更新 此位定义了 PWMx 更新是否由 PWMx Repetition 事件触发。 0：PWMx Repetition 事件不触发 PWMx 更新 1：PWMx Repetition 事件触发 PWMx 更新
[23]	UPDRST	R/W	0x0	PWMx Reset 更新 此位定义了 PWMx 更新是否由 PWMx Reset / Roll-Over 事件触发。 0：PWMx Reset / Roll-Over 事件不触发 PWMx 更新 1：PWMx Reset / Roll-Over 事件触发 PWMx 更新
[22]	UPD5	R/W	0x0	PWM5 更新 此位定义了 PWMx 更新是否由 PWM5 更新事件触发。 0：PWM5 更新事件不触发 PWMx 更新 1：PWM5 更新事件触发 PWMx 更新
[21]	UPD4	R/W	0x0	PWM4 更新 此位定义了 PWMx 更新是否由 PWM4 更新事件触发。 0：PWM4 更新事件不触发 PWMx 更新 1：PWM4 更新事件触发 PWMx 更新
[20]	UPD3	R/W	0x0	PWM3 更新 此位定义了 PWMx 更新是否由 PWM3 更新事件触发。 0：PWM3 更新事件不触发 PWMx 更新 1：PWM3 更新事件触发 PWMx 更新
[19]	UPD2	R/W	0x0	PWM2 更新 此位定义了 PWMx 更新是否由 PWM2 更新事件触发。 0：PWM2 更新事件不触发 PWMx 更新 1：PWM2 更新事件触发 PWMx 更新
[18]	UPD1	R/W	0x0	PWM1 更新 此位定义了 PWMx 更新是否由 PWM1 更新事件触发。 0：PWM1 更新事件不触发 PWMx 更新 1：PWM1 更新事件触发 PWMx 更新

[17]	UPD0	R/W	0x0	PWM0 更新 此位定义了 PWMx 更新是否由 PWM0 更新事件触发。 0：PWM0 更新事件不触发 PWMx 更新 1：PWM0 更新事件触发 PWMx 更新
[16]	MUPD	R/W	0x0	Master PWM 更新 此位定义了 PWMx 更新是否由 Master PWM 更新事件触发。 0：Master PWM 更新事件不触发 PWMx 更新 1：Master PWM 更新事件触发 PWMx 更新
[15-12]	Reserved	Reserved	Reserved	Reserved
[11]	SYNCSTRT	R/W	0x0	同步事件启动 PWMx 此位定义了同步事件后的 PWMx 行为。 0：同步事件不启动 PWMx 1：同步事件启动 PWMx
[10]	SYNCRST	R/W	0x0	同步事件复位 PWMx 此位定义了同步事件后的 PWMx 行为。 0：同步事件不复位 PWMx 1：同步事件复位 PWMx
[9]	RSYNCU	R/W	0x0	PWMx 重同步更新 此位定义了来自 PWMx 外部的 Update 事件是否需要同步。 0：来自 PWMx 外部的 Update 事件等待下个 Reset / Roll-Over 事件后生效 1：来自 PWMx 外部的 Update 事件立即生效
[8]	PSHPLL	R/W	0x0	PWMx Push-pull 模式 此位开启 Push-Pull 模式 0：Push-pull 模式关闭 1：Push-pull 模式开启
[7-6]	INTLVD	R/W	0x0	PWMx Interleaved 模式 此位配置 PWMx Interleaved 模式 ● 1/3 Interleaved 模式下 Compare A 数值自动被设为 Period 数值的 1/3，Compare B 数值自动被设为 Period 数值的 2/3 ● 1/4 Interleaved 模式下 Compare A 数值自动被设为 Period 数值的 1/4，Compare B 数值自动被设为 Period 数值的 2/4，Compare C 数值自动被设为 Period 数值的 3/4 00：Interleaved 模式关闭 01：1/3 Interleaved 模式开启 10：1/4 Interleaved 模式开启 11：Interleaved 模式关闭
[5]	HALF	R/W	0x0	PWMx Half 模式 此位使能 PWMx Half 模式，Half 模式下 Compare A 数值自动被设为 Period 数值的 1/2。 0：Half 模式关闭 1：Half 模式开启

[4]	RETRG	R/W	0x0	PWMx 可重触发模式 此位定义了 PWMx 单次模式下的特性。 0：PWMx 不可重触发，在计数停止后才可以进行计数复位 1：PWMx 可重触发，无论计数状态如何都可以进行计数复位
[3]	CONT	R/W	0x0	PWMx 连续模式 此位定义了 PWMx 计数模式。 0：PWMx 工作在单次模式，计数到达 Period 值后停止 1：PWMx 工作在连续模式，计数到达 Period 值后翻转到 0
[2-0]	CKPSC	R/W	0x0	PWMx 时钟预分频比 此位定义了 PWMx 高分辨率时钟预分频比。 PWMx 等效时钟为 f HRCK / 2 CKPSC.

19.5.2.11 HRPWM PWMx Control Register1 (HRPWM_PWMxCR1)

- **Name**：HRPWM PWMx Control Register1
- **Size**：32bits
- **Offset**：0x84 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-14]	FLTROM	R/W	0x0	故障 Roll-Over 模式选择 此位定义了 Up-Down 模式下何时生成 Roll-Over 事件，这里 Roll-Over 事件对应于 FLTINR2，用于产生 Fault 计数的复位信号。 00：Roll-Over 事件在计数为 Period / 0 时产生 01：Roll-Over 事件在计数为 0 时产生 10：Roll-Over 事件在计数为 Period 时产生 11：Reserved
[13-12]	EEVROM	R/W	0x0	Roll-Over 模式选择 此位定义了 Up-Down 模式下何时生成 Roll-Over 事件，这里 Roll-Over 事件对应于 EEFxR1，用于产生 Event A 计数的复位信号 00：Roll-Over 事件在计数为 Period / 0 时产生 01：Roll-Over 事件在计数为 0 时产生 10：Roll-Over 事件在计数为 Period 时产生 11：Reserved
[11-10]	ADROM	R/W	0x0	ADC Roll-Over 模式选择 此位定义了 Up-Down 模式下何时生成 Roll-Over 事件，这里 Roll-Over 事件对应于 ADT0R-ADT7R，用于产生 Adc Trg 计数的复位信号 00：Roll-Over 事件在计数为 Period / 0 时产生 01：Roll-Over 事件在计数为 0 时产生 10：Roll-Over 事件在计数为 Period 时产生 11：Reserved
[9-8]	OUTROM	R/W	0x0	输出 Roll-Over 模式选择 此位定义了 Up-Down 模式下何时生成 Roll-Over 事件，这里 Roll-Over 事件对应于 SETxyR、CLRxyR，用于产生 PWM 输出。 00：Roll-Over 事件在计数为 Period / 0 时产生 01：Roll-Over 事件在计数为 0 时产生 10：Roll-Over 事件在计数为 Period 时产生 11：Reserved

[7-6]	ROM	R/W	0x0	Roll-Over 模式选择 此位定义了 Up-Down 模式下何时生成 Roll-Over 事件，这里 Roll-Over 事件用于于更新影子寄存器，产生中断标志位，减少重复计数值以及进行外部事件滤波。 00：Roll-Over 事件在计数为 Period / 0 时产生 01：Roll-Over 事件在计数为 0 时产生 10：Roll-Over 事件在计数为 Period 时产生 11：Reserved
[5]	Reserved	Reserved	Reserved	Reserved
[4]	UDM	R/W	0x0	Up-Down 模式选择 此位定义了 PWM 在何种模式下工作。 0：PWM 在 Up 模式下工作 1：PWM 在 Up-Down 模式下工作
[3]	Reserved	Reserved	Reserved	Reserved
[2]	DCDR	R/W	0x0	DAC Reset Trg 来源 此位定义了何时生成 DAC Reset Trg 事件。 0：DAC Reset Trg 产生于 SETA 事件 1：DAC Reset Trg 产生于 RST 事件
[1]	DCDS	R/W	0x0	DAC Step Trg 来源 此位定义了何时生成 DAC Step Trg 事件。 0：DAC Step Trg 产生于 CLRA 事件 1：DAC Step Trg 产生于 CMPD 事件
[0]	DCDE	R/W	0x0	DAC Reset / Step Trg 使能 0：DAC Reset / Step Trg 不使能 1：DAC Reset / Step Trg 使能

19.5.2.12 HRPWM PWMx Interrupt Status Register (HRPWM_PWMxISR)

- **Name**：HRPWM PWMx Interrupt Status Register
- **Size**：32bits
- **Offset**：0x88 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	REP	R/W1C	0x0	Repetition 中断标志 写 1 清除中断标志。 0：Repetition 中断标志未产生 1：Repetition 中断标志产生
[10]	RST	R/W1C	0x0	Reset 中断标志 写 1 清除中断标志。 0：Reset 中断标志未产生 1：Reset 中断标志产生
[9]	CLRB	R/W1C	0x0	Out B Clear 中断标志 写 1 清除中断标志。 0：Out B Clear 中断标志未产生 1：Out B Clear 中断标志产生
[8]	SETB	R/W1C	0x0	Out B Set 中断标志 写 1 清除中断标志。 0：Out B Set 中断标志未产生 1：Out B Set 中断标志产生
[7]	CLRA	R/W1C	0x0	Out A Clear 中断标志 写 1 清除中断标志。 0：Out A Clear 中断标志未产生 1：Out A Clear 中断标志产生
[6]	SETA	R/W1C	0x0	Out A Set 中断标志 写 1 清除中断标志。 0：Out A Set 中断标志未产生 1：Out A Set 中断标志产生
[5]	UPD	R/W1C	0x0	Update 中断标志 写 1 清除中断标志。 0：Update 中断标志未产生 1：Update 中断标志产生
[4]	PER	R/W1C	0x0	Period / Roll-Over 中断标志 写 1 清除中断标志。 0：Period / Roll-Over 中断标志未产生 1：Period / Roll-Over 中断标志产生

[3]	CMPD	R/W1C	0x0	Compare D 中断标志 写 1 清除中断标志。 0：Compare D 中断标志未产生 1：Compare D 中断标志产生
[2]	CMPC	R/W1C	0x0	Compare C 中断标志 写 1 清除中断标志。 0：Compare C 中断标志未产生 1：Compare C 中断标志产生
[1]	CMPB	R/W1C	0x0	Compare B 中断标志 写 1 清除中断标志。 0：Compare B 中断标志未产生 1：Compare B 中断标志产生
[0]	CMPA	R/W1C	0x0	Compare A 中断标志 写 1 清除中断标志。 0：Compare A 中断标志未产生 1：Compare A 中断标志产生

19.5.2.13 HRPWM PWMx Interrupt Enable Register (HRPWM_PWMxIER)

- **Name：** HRPWM PWMx Interrupt Enable Register
- **Size：** 32bits
- **Offset：** 0x8C + 0x80 * x (x = 0~5)
- **Default：** 0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	REPIE	R/W	0x0	Repetition 中断使能 0：Repetition 中断不使能 1：Repetition 中断使能
[10]	RSTIE	R/W	0x0	Reset 中断使能 0：Reset 中断不使能 1：Reset 中断使能
[9]	CLRBIE	R/W	0x0	Out B Clear 中断使能 0：Out B Clear 中断不使能 1：Out B Clear 中断使能
[8]	SETBIE	R/W	0x0	Out B Set 中断使能 0：Out B Set 中断不使能 1：Out B Set 中断使能
[7]	CLRAIE	R/W	0x0	Out A Clear 中断使能 0：Out A Clear 中断不使能 1：Out A Clear 中断使能
[6]	SETAIE	R/W	0x0	Out A Set 中断使能 0：Repetition 中断不使能 1：Repetition 中断使能

[5]	UPDIE	R/W	0x0	Update 中断使能 0：Update 中断不使能 1：Update 中断使能
[4]	PERIE	R/W	0x0	Period / Roll-Over 中断使能 0：Period / Roll-Over 中断不使能 1：Period / Roll-Over 中断使能
[3]	CMPDIE	R/W	0x0	Compare D 中断使能 0：Compare D 中断不使能 1：Compare D 中断使能
[2]	CMPCIE	R/W	0x0	Compare C 中断使能 0：Compare C 中断不使能 1：Compare C 中断使能
[1]	CMPBIE	R/W	0x0	Compare B 中断使能 0：Compare B 中断不使能 1：Compare B 中断使能
[0]	CMPAIE	R/W	0x0	Compare A 中断使能 0：Compare A 中断不使能 1：Compare A 中断使能

19.5.2.14 HRPWM PWMx Counter Register (HRPWM_CNTxR)

- **Name**：HRPWM PWMx Counter Register
- **Size**：32bits
- **Offset**：0x90 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	REP	R/W	0x0	PWMx Repetition Period 数值 此位保持 PWMx Repetition Period 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。
[23-20]	Reserved	Reserved	Reserved	Reserved
[19]	CNTWR	R/W	0x0	对此位写 1 同时写 CNT，可以将 CNT 写入到 PWMx Counter，写入之后此位自动清零
[18]	CNTRD	R/W	0x0	对此位写 1，可以将 PWMx Counter 当前值读取到 CNT，读取之后此位自动清零
[17-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	CNT	R/W	0x0	PWMx Counter 数值 此位保持 PWMx Counter 数值，当高精度模式（CKPSC＜5）开启时，此位的低有效位无意义，这些位无法被写入，读取时返回为 0。

19.5.2.15 HRPWM PWMx Period Register (HRPWM_PERxR)

- **Name**：HRPWM PWMx Period Register
- **Size**：32bits
- **Offset**：0x94 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	PER	R/W	0x0	PWMx Period 数值 此位保持 PWMx Period 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Period 数值必须大于或等于 3 个 f_{HRPWM} 周期，若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30，若 CKPSC＝2 为 0x18，Period 最大值为 0xFFDF。

19.5.2.16 HRPWM PWMx Compare A Register (HRPWM_CMPAxR)

- **Name**：HRPWM PWMx Compare A Register
- **Size**：32bits
- **Offset**：0x98 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	CMPA	R/W	0x0	PWMx Compare A 数值 此位保持 PWMx Compare A 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Compare A 数值必须大于或等于 3 个 f_{HRPWM} 周期，若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30，若 CKPSC＝2 为 0x18。

19.5.2.17 HRPWM PWMx Compare B Register (HRPWM_CMPBxR)

- **Name**：HRPWM PWMx Compare B Register
- **Size**：32bits
- **Offset**：0x9C + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	CMPB	R/W	0x0	PWMx Compare B 数值 此位保持 PWMx Compare B 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Compare B 数值必须大于或等于 3 个 f_{HRPWM} 周期，若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30，若 CKPSC＝2 为 0x18。

19.5.2.18 HRPWM PWMx Compare C Register (HRPWM_CMPCxR)

- **Name**：HRPWM PWMx Compare C Register
- **Size**：32bits
- **Offset**：0xA0 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	CMPC	R/W	0x0	PWMx Compare C 数值 此位保持 PWMx Compare C 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Compare C 数值必须大于或等于 3 个 f_{HRPWM} 周期，若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30，若 CKPSC＝2 为 0x18。

19.5.2.19 HRPWM PWMx Compare D Register (HRPWM_CMPDxR)

- **Name**：HRPWM PWMx Compare D Register
- **Size**：32bits
- **Offset**：0xA4 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	CMPD	R/W	0x0	PWMx Compare D 数值 此位保持 PWMx Compare D 数值，此位为影子寄存器（预加载）数值，如果预加载功能被禁用，则此位同时也是工作寄存器数值。 Compare D 数值必须大于或等于 3 个 f_{HRPWM} 周期，若 CKPSC＝0 为 0x60，若 CKPSC＝1 为 0x30，若 CKPSC＝2 为 0x18。

19.5.2.20 HRPWM PWMx DeadTime Register (HRPWM_DTxR)

- **Name**：HRPWM PWMx DeadTime Register
- **Size**：32bits
- **Offset**：0xA8 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-29]	Reserved	Reserved	Reserved	Reserved
[28]	SDTF	R/W	0x0	下降沿死区符号 该位确定死区时间是正向还是负向 0：下降沿死区为负向（信号重叠） 1：下降沿死区为正向（信号无重叠）
[27-16]	DTF	R/W	0x0	下降沿死区时间 此位定义了在 PWM 参考信号下降沿之后的死区时间 t_{DTR} = DTR × t_{DTG}
[15-13]	Reserved	Reserved	Reserved	Reserved
[12]	SDTR	R/W	0x0	上升沿死区符号 该位确定死区时间是正向还是负向 0：上升沿死区为负向（信号重叠） 1：上升沿死区为正向（信号无重叠）
[11-0]	DTR	R/W	0x0	上升沿死区时间 此位定义了在 PWM 参考信号上升沿之后的死区时间 t_{DTR} = DTR × t_{DTG}

19.5.2.21 HRPWM PWMx Output A Set Register (HRPWM_SETxAR)

- **Name**：HRPWM PWMx Output A Set Register
- **Size**：32bits
- **Offset**：0xAC + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-19]	Reserved	Reserved	Reserved	Reserved
[18]	SST	R/W	0x0	此位使能后 Software Set Trigger 事件使 PWMx Out A 进入有效状态。
[17]	RESYNC	R/W	0x0	此位使能后 Sync Input Event 事件使 PWMx Out A 进入有效状态。
[16]	EXTEVNT5	R/W	0x0	此位使能后 PWMx Event 5 事件使 PWMx Out A 进入有效状态。
[15]	EXTEVNT4	R/W	0x0	此位使能后 PWMx Event 4 事件使 PWMx Out A 进入有效状态。
[14]	EXTEVNT3	R/W	0x0	此位使能后 PWMx Event 3 事件使 PWMx Out A 进入有效状态。
[13]	EXTEVNT2	R/W	0x0	此位使能后 PWMx Event 2 事件使 PWMx Out A 进入有效状态。
[12]	EXTEVNT1	R/W	0x0	此位使能后 PWMx Event 1 事件使 PWMx Out A 进入有效状态。
[11]	EXTEVNT0	R/W	0x0	此位使能后 PWMx Event 0 事件使 PWMx Out A 进入有效状态。
[10]	MSTPER	R/W	0x0	此位使能后 Master PWM Period 事件使 PWMx Out A 进入有效状态。
[9]	MSTCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件使 PWMx Out A 进入有效状态。
[8]	MSTCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件使 PWMx Out A 进入有效状态。
[7]	MSTCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件使 PWMx Out A 进入有效状态。
[6]	MSTCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件使 PWMx Out A 进入有效状态。
[5]	PER	R/W	0x0	此位使能后 PWMx Roll-Over 事件使 PWMx Out A 进入有效状态。
[4]	CMPD	R/W	0x0	此位使能后 PWMx Compare D 事件使 PWMx Out A 进入有效状态。
[3]	CMPC	R/W	0x0	此位使能后 PWMx Compare C 事件使 PWMx Out A 进入有效状态。
[2]	CMPB	R/W	0x0	此位使能后 PWMx Compare B 事件使 PWMx Out A 进入有效状态。
[1]	CMPA	R/W	0x0	此位使能后 PWMx Compare A 事件使 PWMx Out A 进入有效状态。
[0]	UPD	R/W	0x0	此位使能后 PWMx Update 事件使 PWMx Out A 进入有效状态。

19.5.2.22 HRPWM PWMx Output A Clear Register (HRPWM_CLRxAR)

- **Name**：HRPWM PWMx Output A Clear Register
- **Size**：32bits
- **Offset**：0xB0 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-19]	Reserved	Reserved	Reserved	Reserved
[18]	SST	R/W	0x0	此位使能后 Software Set Trigger 事件使 PWMx Out A 进入无效状态。
[17]	RESYNC	R/W	0x0	此位使能后 Sync Input Event 事件使 PWMx Out A 进入无效状态。
[16]	EXTEVNT5	R/W	0x0	此位使能后 PWMx Event 5 事件使 PWMx Out A 进入无效状态。
[15]	EXTEVNT4	R/W	0x0	此位使能后 PWMx Event 4 事件使 PWMx Out A 进入无效状态。
[14]	EXTEVNT3	R/W	0x0	此位使能后 PWMx Event 3 事件使 PWMx Out A 进入无效状态。
[13]	EXTEVNT2	R/W	0x0	此位使能后 PWMx Event 2 事件使 PWMx Out A 进入无效状态。
[12]	EXTEVNT1	R/W	0x0	此位使能后 PWMx Event 1 事件使 PWMx Out A 进入无效状态。
[11]	EXTEVNT0	R/W	0x0	此位使能后 PWMx Event 0 事件使 PWMx Out A 进入无效状态。
[10]	MSTPER	R/W	0x0	此位使能后 Master PWM Period 事件使 PWMx Out A 进入无效状态。
[9]	MSTCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件使 PWMx Out A 进入无效状态。
[8]	MSTCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件使 PWMx Out A 进入无效状态。
[7]	MSTCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件使 PWMx Out A 进入无效状态。
[6]	MSTCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件使 PWMx Out A 进入无效状态。
[5]	PER	R/W	0x0	此位使能后 PWMx Roll-Over 事件使 PWMx Out A 进入无效状态。
[4]	CMPD	R/W	0x0	此位使能后 PWMx Compare D 事件使 PWMx Out A 进入无效状态。
[3]	CMPC	R/W	0x0	此位使能后 PWMx Compare C 事件使 PWMx Out A 进入无效状态。
[2]	CMPB	R/W	0x0	此位使能后 PWMx Compare B 事件使 PWMx Out A 进入无效状态。
[1]	CMPA	R/W	0x0	此位使能后 PWMx Compare A 事件使 PWMx Out A 进入无效状态。
[0]	UPD	R/W	0x0	此位使能后 PWMx Update 事件使 PWMx Out A 进入无效状态。

19.5.2.23 HRPWM PWMx Output B Set Register (HRPWM_SETxBR)

- **Name**：HRPWM PWMx Output B Set Register
- **Size**：32bits
- **Offset**：0xB4 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-19]	Reserved	Reserved	Reserved	Reserved
[18]	SST	R/W	0x0	此位使能后 Software Set Trigger 事件使 PWMx Out B 进入有效状态。
[17]	RESYNC	R/W	0x0	此位使能后 Sync Input Event 事件使 PWMx Out B 进入有效状态。
[16]	EXTEVNT5	R/W	0x0	此位使能后 PWMx Event 5 事件使 PWMx Out B 进入有效状态。
[15]	EXTEVNT4	R/W	0x0	此位使能后 PWMx Event 4 事件使 PWMx Out B 进入有效状态。
[14]	EXTEVNT3	R/W	0x0	此位使能后 PWMx Event 3 事件使 PWMx Out B 进入有效状态。
[13]	EXTEVNT2	R/W	0x0	此位使能后 PWMx Event 2 事件使 PWMx Out B 进入有效状态。
[12]	EXTEVNT1	R/W	0x0	此位使能后 PWMx Event 1 事件使 PWMx Out B 进入有效状态。
[11]	EXTEVNT0	R/W	0x0	此位使能后 PWMx Event 0 事件使 PWMx Out B 进入有效状态。
[10]	MSTPER	R/W	0x0	此位使能后 Master PWM Period 事件使 PWMx Out B 进入有效状态。
[9]	MSTCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件使 PWMx Out B 进入有效状态。
[8]	MSTCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件使 PWMx Out B 进入有效状态。
[7]	MSTCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件使 PWMx Out B 进入有效状态。
[6]	MSTCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件使 PWMx Out B 进入有效状态。
[5]	PER	R/W	0x0	此位使能后 PWMx Roll-Over 事件使 PWMx Out B 进入有效状态。
[4]	CMPD	R/W	0x0	此位使能后 PWMx Compare D 事件使 PWMx Out B 进入有效状态。
[3]	CMPC	R/W	0x0	此位使能后 PWMx Compare C 事件使 PWMx Out B 进入有效状态。
[2]	CMPB	R/W	0x0	此位使能后 PWMx Compare B 事件使 PWMx Out B 进入有效状态。
[1]	CMPA	R/W	0x0	此位使能后 PWMx Compare A 事件使 PWMx Out B 进入有效状态。
[0]	UPD	R/W	0x0	此位使能后 PWMx Update 事件使 PWMx Out B 进入有效状态。

19.5.2.24 HRPWM PWMx Output B Clear Register (HRPWM_CLRxBR)

- **Name**：HRPWM PWMx Output B Clear Register
- **Size**：32bits
- **Offset**：0xB8 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-19]	Reserved	Reserved	Reserved	Reserved
[18]	SST	R/W	0x0	此位使能后 Software Set Trigger 事件使 PWMx Out B 进入无效状态。
[17]	RESYNC	R/W	0x0	此位使能后 Sync Input Event 事件使 PWMx Out B 进入无效状态。
[16]	EXTEVNT5	R/W	0x0	此位使能后 PWMx Event 5 事件使 PWMx Out B 进入无效状态。
[15]	EXTEVNT4	R/W	0x0	此位使能后 PWMx Event 4 事件使 PWMx Out B 进入无效状态。
[14]	EXTEVNT3	R/W	0x0	此位使能后 PWMx Event 3 事件使 PWMx Out B 进入无效状态。
[13]	EXTEVNT2	R/W	0x0	此位使能后 PWMx Event 2 事件使 PWMx Out B 进入无效状态。
[12]	EXTEVNT1	R/W	0x0	此位使能后 PWMx Event 1 事件使 PWMx Out B 进入无效状态。
[11]	EXTEVNT0	R/W	0x0	此位使能后 PWMx Event 0 事件使 PWMx Out B 进入无效状态。
[10]	MSTPER	R/W	0x0	此位使能后 Master PWM Period 事件使 PWMx Out B 进入无效状态。
[9]	MSTCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件使 PWMx Out B 进入无效状态。
[8]	MSTCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件使 PWMx Out B 进入无效状态。
[7]	MSTCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件使 PWMx Out B 进入无效状态。
[6]	MSTCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件使 PWMx Out B 进入无效状态。
[5]	PER	R/W	0x0	此位使能后 PWMx Roll-Over 事件使 PWMx Out B 进入无效状态。
[4]	CMPD	R/W	0x0	此位使能后 PWMx Compare D 事件使 PWMx Out B 进入无效状态。
[3]	CMPC	R/W	0x0	此位使能后 PWMx Compare C 事件使 PWMx Out B 进入无效状态。
[2]	CMPB	R/W	0x0	此位使能后 PWMx Compare B 事件使 PWMx Out B 进入无效状态。
[1]	CMPA	R/W	0x0	此位使能后 PWMx Compare A 事件使 PWMx Out B 进入无效状态。
[0]	UPD	R/W	0x0	此位使能后 PWMx Update 事件使 PWMx Out B 进入无效状态。

19.5.2.25 HRPWM PWMx External Event Register0 (HRPWM_EEFxR0)

- **Name**：HRPWM PWMx External Event Register0
- **Size**：32bits
- **Offset**：0xBC + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-30]	Reserved	Reserved	Reserved	Reserved
[29-26]	EE5FLTR	R/W	0x0	Event 5 滤波位 0000：无消隐 / 加窗滤波功能 0001：从 Reset / Roll-Over 事件到 Compare A 事件期间消隐，消隐期间事件无法通过 0010：从 Reset / Roll-Over 事件到 Compare B 事件期间消隐，消隐期间事件无法通过 0011：从 Reset / Roll-Over 事件到 Compare C 事件期间消隐，消隐期间事件无法通过 0100：从 Reset / Roll-Over 事件到 Compare D 事件期间消隐，消隐期间事件无法通过 0101：从 Up 阶段 Compare A 事件到 Up 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0110：从 Up 阶段 Compare C 事件到 Up 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0111：从 Down 阶段 Compare A 事件到 Down 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1000：从 Down 阶段 Compare C 事件到 Down 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1001：从 Reset / Roll-Over 事件到 Compare A 事件期间加窗，加窗期间以外事件无法通过 1010：从 Reset / Roll-Over 事件到 Compare B 事件期间加窗，加窗期间以外事件无法通过 1011：从 Reset / Roll-Over 事件到 Compare C 事件期间加窗，加窗期间以外事件无法通过 1100：从 Reset / Roll-Over 事件到 Compare D 事件期间加窗，加窗期间以外事件无法通过 1101：从 Up 阶段 Compare B 事件到 Up 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1110：从 Down 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1111：从 Up 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效
[25]	EE5LTCH	R/W	0x0	Event 5 锁存位 0：Event 5 出现在消隐期间，则事件不被锁存 1：Event 5 出现在消隐期间，则事件被锁存

[24-21]	EE4FLTR	R/W	0x0	Event 4 滤波位 0000：无消隐 / 加窗滤波功能 0001：从 Reset / Roll-Over 事件到 Compare A 事件期间消隐，消隐期间事件无法通过 0010：从 Reset / Roll-Over 事件到 Compare B 事件期间消隐，消隐期间事件无法通过 0011：从 Reset / Roll-Over 事件到 Compare C 事件期间消隐，消隐期间事件无法通过 0100：从 Reset / Roll-Over 事件到 Compare D 事件期间消隐，消隐期间事件无法通过 0101：从 Up 阶段 Compare A 事件到 Up 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0110：从 Up 阶段 Compare C 事件到 Up 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0111：从 Down 阶段 Compare A 事件到 Down 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1000：从 Down 阶段 Compare C 事件到 Down 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1001：从 Reset / Roll-Over 事件到 Compare A 事件期间加窗，加窗期间以外事件无法通过 1010：从 Reset / Roll-Over 事件到 Compare B 事件期间加窗，加窗期间以外事件无法通过 1011：从 Reset / Roll-Over 事件到 Compare C 事件期间加窗，加窗期间以外事件无法通过 1100：从 Reset / Roll-Over 事件到 Compare D 事件期间加窗，加窗期间以外事件无法通过 1101：从 Up 阶段 Compare B 事件到 Up 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1110：从 Down 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1111：从 Up 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效
[20]	EE4LTCH	R/W	0x0	Event 4 锁存位 1：Event 4 出现在消隐期间，则事件被锁存 0：Event 4 出现在消隐期间，则事件不被锁存

[19-16]	EE3FLTR	R/W	0x0	Event 3 滤波位 0000：无消隐 / 加窗滤波功能 0001：从 Reset / Roll-Over 事件到 Compare A 事件期间消隐，消隐期间事件无法通过 0010：从 Reset / Roll-Over 事件到 Compare B 事件期间消隐，消隐期间事件无法通过 0011：从 Reset / Roll-Over 事件到 Compare C 事件期间消隐，消隐期间事件无法通过 0100：从 Reset / Roll-Over 事件到 Compare D 事件期间消隐，消隐期间事件无法通过 0101：从 Up 阶段 Compare A 事件到 Up 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0110：从 Up 阶段 Compare C 事件到 Up 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0111：从 Down 阶段 Compare A 事件到 Down 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1000：从 Down 阶段 Compare C 事件到 Down 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1001：从 Reset / Roll-Over 事件到 Compare A 事件期间加窗，加窗期间以外事件无法通过 1010：从 Reset / Roll-Over 事件到 Compare B 事件期间加窗，加窗期间以外事件无法通过 1011：从 Reset / Roll-Over 事件到 Compare C 事件期间加窗，加窗期间以外事件无法通过 1100：从 Reset / Roll-Over 事件到 Compare D 事件期间加窗，加窗期间以外事件无法通过 1101：从 Up 阶段 Compare B 事件到 Up 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1110：从 Down 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1111：从 Up 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效
[15]	EE3LTCH	R/W	0x0	Event 3 锁存位 0：Event 3 出现在消隐期间，则事件不被锁存 1：Event 3 出现在消隐期间，则事件被锁存

[14-11]	EE2FLTR	R/W	0x0	Event 2 滤波位 0000：无消隐 / 加窗滤波功能 0001：从 Reset / Roll-Over 事件到 Compare A 事件期间消隐，消隐期间事件无法通过 0010：从 Reset / Roll-Over 事件到 Compare B 事件期间消隐，消隐期间事件无法通过 0011：从 Reset / Roll-Over 事件到 Compare C 事件期间消隐，消隐期间事件无法通过 0100：从 Reset / Roll-Over 事件到 Compare D 事件期间消隐，消隐期间事件无法通过 0101：从 Up 阶段 Compare A 事件到 Up 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0110：从 Up 阶段 Compare C 事件到 Up 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0111：从 Down 阶段 Compare A 事件到 Down 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1000：从 Down 阶段 Compare C 事件到 Down 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1001：从 Reset / Roll-Over 事件到 Compare A 事件期间加窗，加窗期间以外事件无法通过 1010：从 Reset / Roll-Over 事件到 Compare B 事件期间加窗，加窗期间以外事件无法通过 1011：从 Reset / Roll-Over 事件到 Compare C 事件期间加窗，加窗期间以外事件无法通过 1100：从 Reset / Roll-Over 事件到 Compare D 事件期间加窗，加窗期间以外事件无法通过 1101：从 Up 阶段 Compare B 事件到 Up 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1110：从 Down 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1111：从 Up 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效
[10]	EE2LTCH	R/W	0x0	Event 2 锁存位 0：Event 2 出现在消隐期间，则事件不被锁存 1：Event 2 出现在消隐期间，则事件被锁存

[9-6]	EE1FLTR	R/W	0x0	Event 1 滤波位 0000：无消隐 / 加窗滤波功能 0001：从 Reset / Roll-Over 事件到 Compare A 事件期间消隐，消隐期间事件无法通过 0010：从 Reset / Roll-Over 事件到 Compare B 事件期间消隐，消隐期间事件无法通过 0011：从 Reset / Roll-Over 事件到 Compare C 事件期间消隐，消隐期间事件无法通过 0100：从 Reset / Roll-Over 事件到 Compare D 事件期间消隐，消隐期间事件无法通过 0101：从 Up 阶段 Compare A 事件到 Up 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0110：从 Up 阶段 Compare C 事件到 Up 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0111：从 Down 阶段 Compare A 事件到 Down 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1000：从 Down 阶段 Compare C 事件到 Down 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1001：从 Reset / Roll-Over 事件到 Compare A 事件期间加窗，加窗期间以外事件无法通过 1010：从 Reset / Roll-Over 事件到 Compare B 事件期间加窗，加窗期间以外事件无法通过 1011：从 Reset / Roll-Over 事件到 Compare C 事件期间加窗，加窗期间以外事件无法通过 1100：从 Reset / Roll-Over 事件到 Compare D 事件期间加窗，加窗期间以外事件无法通过 1101：从 Up 阶段 Compare B 事件到 Up 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1110：从 Down 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1111：从 Up 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效
[5]	EE1LTCH	R/W	0x0	Event 1 锁存位 0：Event 1 出现在消隐期间，则事件不被锁存 1：Event 1 出现在消隐期间，则事件被锁存

[4-1]	EE0FLTR	R/W	0x0	Event 0 滤波位 0000：无消隐 / 加窗滤波功能 0001：从 Reset / Roll-Over 事件到 Compare A 事件期间消隐，消隐期间事件无法通过 0010：从 Reset / Roll-Over 事件到 Compare B 事件期间消隐，消隐期间事件无法通过 0011：从 Reset / Roll-Over 事件到 Compare C 事件期间消隐，消隐期间事件无法通过 0100：从 Reset / Roll-Over 事件到 Compare D 事件期间消隐，消隐期间事件无法通过 0101：从 Up 阶段 Compare A 事件到 Up 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0110：从 Up 阶段 Compare C 事件到 Up 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 0111：从 Down 阶段 Compare A 事件到 Down 阶段 Compare B 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1000：从 Down 阶段 Compare C 事件到 Down 阶段 Compare D 事件期间消隐，消隐期间事件无法通过，此位仅在 Updown 模式下有效 1001：从 Reset / Roll-Over 事件到 Compare A 事件期间加窗，加窗期间以外事件无法通过 1010：从 Reset / Roll-Over 事件到 Compare B 事件期间加窗，加窗期间以外事件无法通过 1011：从 Reset / Roll-Over 事件到 Compare C 事件期间加窗，加窗期间以外事件无法通过 1100：从 Reset / Roll-Over 事件到 Compare D 事件期间加窗，加窗期间以外事件无法通过 1101：从 Up 阶段 Compare B 事件到 Up 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1110：从 Down 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效 1111：从 Up 阶段 Compare B 事件到 Down 阶段 Compare C 事件期间加窗，加窗期间以外事件无法通过，此位仅在 Updown 模式下有效
[0]	EE0LTCH	R/W	0x0	Event 0 锁存位 0：Event 0 出现在消隐期间，则事件不被锁存 1：Event 0 出现在消隐期间，则事件被锁存

19.5.2.26 HRPWM PWMx External Event Register1 (HRPWM_EEFxR1)

- **Name**：HRPWM PWMx External Event Register1
- **Size**：32bits
- **Offset**：0xC0 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-14]	Reserved	Reserved	Reserved	Reserved
[13-8]	EEVACNT	R/W	0x0	Event A 计数阈值 当事件总数达到（EEVACNT＋1）时认为事件有效
[7]	Reserved	Reserved	Reserved	Reserved
[6-4]	EEVASEL	R/W	0x0	Event A 来源选择 000：Event 0 为 Event A 001：Event 1 为 Event A 010：Event 2 为 Event A 011：Event 3 为 Event A 100：Event 4 为 Event A 101：Event 5 为 Event A 110-111：Reserved
[3]	Reserved	Reserved	Reserved	Reserved
[2]	EEVARSTM	R/W	0x0	Event A 计数复位模式 0：Event A 计数在 Reset / Roll-Over 事件处复位 1：Event A 计数在 Reset / Roll-Over 事件处复位（仅当上个周期无事件发生时）
[1]	EEVACRES	R/WAC	0x0	Event A 计数复位 此位将 Event A 计数复位，软件写 1 后等复位完成后硬件自动清 0。 0：Event A 计数不复位 1：Event A 计数复位
[0]	EEVACE	R/W	0x0	Event A 计数使能 0：Event A 计数不使能 1：Event A 计数使能

19.5.2.27 HRPWM PWMx Reset Register (HRPWM_RSTxR)

- **Name**：HRPWM PWMx Reset Register
- **Size**：32bits
- **Offset**：0xC4 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-29]	Reserved	Reserved	Reserved	Reserved
[28]	EXTEVT5	R/W	0x0	此位使能后 PWMx Event 5 事件使 PWMx 计数器复位
[27]	EXTEVT4	R/W	0x0	此位使能后 PWMx Event 4 事件使 PWMx 计数器复位
[26]	EXTEVT3	R/W	0x0	此位使能后 PWMx Event 3 事件使 PWMx 计数器复位
[25]	UPDCMPA5	R/W	0x0	此位使能后 PWM5 Update/Compare A 事件使 PWMx 计数器复位
[24]	CMPD4	R/W	0x0	此位使能后 PWM4 Compare D 事件使 PWMx 计数器复位
[23]	CMPB4	R/W	0x0	此位使能后 PWM4 Compare B 事件使 PWMx 计数器复位
[22]	UPDCMPA4	R/W	0x0	此位使能后 PWM4 Update/Compare A 事件使 PWMx 计数器复位
[21]	CMPD3	R/W	0x0	此位使能后 PWM3 Compare D 事件使 PWMx 计数器复位
[20]	CMPB3	R/W	0x0	此位使能后 PWM3 Compare B 事件使 PWMx 计数器复位
[19]	UPDCMPA3	R/W	0x0	此位使能后 PWM3 Update/Compare A 事件使 PWMx 计数器复位
[18]	CMPD2	R/W	0x0	此位使能后 PWM2 Compare D 事件使 PWMx 计数器复位
[17]	CMPB2	R/W	0x0	此位使能后 PWM2 Compare B 事件使 PWMx 计数器复位
[16]	UPDCMPA2	R/W	0x0	此位使能后 PWM2 Update/Compare A 事件使 PWMx 计数器复位
[15]	CMPD1	R/W	0x0	此位使能后 PWM1 Compare D 事件使 PWMx 计数器复位
[14]	CMPB1	R/W	0x0	此位使能后 PWM1 Compare B 事件使 PWMx 计数器复位
[13]	UPDCMPA1	R/W	0x0	此位使能后 PWM1 Update/Compare A 事件使 PWMx 计数器复位
[12]	CMPD0	R/W	0x0	此位使能后 PWM0 Compare D 事件使 PWMx 计数器复位
[11]	CMPB0	R/W	0x0	此位使能后 PWM0 Compare B 事件使 PWMx 计数器复位
[10]	UPDCMPA0	R/W	0x0	此位使能后 PWM0 Update/Compare A 事件使 PWMx 计数器复位
[9]	CMPD5	R/W	0x0	此位使能后 PWM5 Compare D 事件使 PWMx 计数器复位
[8]	CMPB5	R/W	0x0	此位使能后 PWM5 Compare B 事件使 PWMx 计数器复位
[7]	EXTEVT2	R/W	0x0	此位使能后 PWMx Event 2 事件使 PWMx 计数器复位
[6]	EXTEVT1	R/W	0x0	此位使能后 PWMx Event 1 事件使 PWMx 计数器复位
[5]	EXTEVT0	R/W	0x0	此位使能后 PWMx Event 0 事件使 PWMx 计数器复位
[4]	MSTPER	R/W	0x0	此位使能后 Master PWM Period 事件使 PWMx 计数器复位
[3]	MSTCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件使 PWMx 计数器复位
[2]	MSTCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件使 PWMx 计数器复位
[1]	MSTCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件使 PWMx 计数器复位
[0]	MSTCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件使 PWMx 计数器复位

19.5.2.28 HRPWM PWMx Chopper Register (HRPWM_CHPxR)

- **Name**：HRPWM PWMx Chopper Register
- **Size**：32bits
- **Offset**：0xC8 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-11]	Reserved	Reserved	Reserved	Reserved
[10-7]	STRPW	R/W	0x0	PWMx 启动脉冲宽度 此位定义在输出信号的上升沿之后的初始脉冲宽度 t_{ISTPW}＝ (STRPW＋1) × 16 × t_{HRPWM} 0000：100 ns（1/10MHz） …… 1111：1.6 us（16/10MHz）
[6-4]	CARDTY	R/W	0x0	PWMx 斩波占空比值此位定义斩波信号的占空比 000：0/8（仅存在第一个脉冲） …… 1111：7/8
[3-0]	CARFRQ	R/W	0x0	PWMx 载波频率值 此位定义载波频率为 FCHPFRQ＝f_{HRPWM} / （16 × (CARFRQ＋1)) 0000：10 MHz (f_{HRPWM} / 16) …… 1111：625 KHz (f_{HRPWM} / 256)

19.5.2.29 HRPWM PWMx Output Register (HRPWM_OUTxR)

- **Name**：HRPWM PWMx Output Register
- **Size**：32bits
- **Offset**：0xCC + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	DTEN	R/W	0x0	死区使能 0：死区输出关闭此位使得输出 A 和输出 B 之间插入死区 1：死区输出开启（根据输出 A 产生死区输出）
[30-21]	Reserved	Reserved	Reserved	Reserved
[20]	CHPB	R/W	0x0	输出 B 斩波使能 0：输出斩波关闭此位使得输出 B 开启斩波 1：输出斩波开启
[19]	IDLESB	R/W	0x0	输出 B 空闲状态 此位控制输出 B 在空闲状态的输出电平。 0：无效电平 1：有效电平
[18-17]	FAULTB	R/W	0x0	输出 B 故障状态 此位控制输出 B 在故障状态的输出电平。 00：无动作 01：有效电平 10：无效电平 11：高阻态
[16]	POLB	R/W	0x0	输出 B 极性 此位选择输出 B 的极性。 0：正极性（输出高有效） 1：负极性（输出低有效）
[15-5]	Reserved	Reserved	Reserved	Reserved
[4]	CHPA	R/W	0x0	输出 A 斩波使能 0：输出斩波关闭此位使得输出 A 开启斩波 1：输出斩波开启
[3]	IDLESA	R/W	0x0	输出 A 空闲状态 此位控制输出 A 在空闲状态的输出电平。 0：无效电平 1：有效电平
[2-1]	FAULTA	R/W	0x0	输出 A 故障电平 此位控制输出 A 在故障状态的输出电平。 00：无动作 01：有效电平 10：无效电平 11：高阻态

[0]	POLA	R/W	0x0	输出 A 极性 此位选择输出 A 的极性。 0：正极性（输出高有效） 1：负极性（输出低有效）

19.5.2.30 HRPWM PWMx Fault Register (HRPWM_FLTxR)

- **Name**：HRPWM PWMx Fault Register
- **Size**：32bits
- **Offset**：0xD0 + 0x80 * x (x = 0~5)
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-6]	Reserved	Reserved	Reserved	Reserved
[5]	FLT5EN	R/W	0x0	Fault 5 使能 0：Fault 5 关闭 1：Fault 5 开启，可以停止 PWM 输出
[4]	FLT4EN	R/W	0x0	Fault 4 使能 0：Fault 4 关闭 1：Fault 4 开启，可以停止 PWM 输出
[3]	FLT3EN	R/W	0x0	Fault 3 使能 0：Fault 3 关闭 1：Fault 3 开启，可以停止 PWM 输出
[2]	FLT2EN	R/W	0x0	Fault 2 使能 0：Fault 2 关闭 1：Fault 2 开启，可以停止 PWM 输出
[1]	FLT1EN	R/W	0x0	Fault 1 使能 0：Fault 1 关闭 1：Fault 1 开启，可以停止 PWM 输出
[0]	FLT0EN	R/W	0x0	Fault 0 使能 0：Fault 0 关闭 1：Fault 0 开启，可以停止 PWM 输出

19.5.2.31 HRPWM Control Register0 (HRPWM_CR0)

- **Name**：HRPWM Control Register0
- **Size**：32bits
- **Offset**：0x380
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-29]	USRC7	R/W	0x0	Add Trg 7 更新来源 此位定义了触发 ADT7R 寄存器更新的来源，定义了来源未定义更具体的触发条件。 000：Master PWM 001：PWM0 010：PWM1 011：PWM2 100：PWM3 101：PWM4 110：PWM5 111：Reserved
[28-26]	USRC6	R/W	0x0	Add Trg 6 更新来源 此位定义了触发 ADT6R 寄存器更新的来源，定义了来源未定义更具体的触发条件。 000：Master PWM 001：PWM0 010：PWM1 011：PWM2 100：PWM3 101：PWM4 110：PWM5 111：Reserved
[25-23]	USRC5	R/W	0x0	Add Trg 5 更新来源 此位定义了触发 ADT5R 寄存器更新的来源，定义了来源未定义更具体的触发条件。 000：Master PWM 001：PWM0 010：PWM1 011：PWM2 100：PWM3 101：PWM4 110：PWM5 111：Reserved

[22-20]	USRC4	R/W	0x0	Add Trg 4 更新来源 此位定义了触发 ADT4R 寄存器更新的来源，定义了来源未定义更具体的触发条件。 000：Master PWM 001：PWM0 010：PWM1 011：PWM2 100：PWM3 101：PWM4 110：PWM5 111：Reserved
[19-17]	USRC3	R/W	0x0	Add Trg 3 更新来源 此位定义了触发 ADT3R 寄存器更新的来源，定义了来源未定义更具体的触发条件。 000：Master PWM 001：PWM0 010：PWM1 011：PWM2 100：PWM3 101：PWM4 110：PWM5 111：Reserved
[16-14]	USRC2	R/W	0x0	Add Trg 2 更新来源 此位定义了触发 ADT2R 寄存器更新的来源，定义了来源未定义更具体的触发条件。 000：Master PWM 001：PWM0 010：PWM1 011：PWM2 100：PWM3 101：PWM4 110：PWM5 111：Reserved
[13-11]	USRC1	R/W	0x0	Add Trg 1 更新来源 此位定义了触发 ADT1R 寄存器更新的来源，定义了来源未定义更具体的触发条件。 000：Master PWM 001：PWM0 010：PWM1 011：PWM2 100：PWM3 101：PWM4 110：PWM5 111：Reserved

[10-8]	USRC0	R/W	0x0	Add Trg 0 更新来源 此位定义了触发 ADT0R 寄存器更新的来源，定义了来源未定义更具体的触发条件。 000：Master PWM 001：PWM0 010：PWM1 011：PWM2 100：PWM3 101：PWM4 110：PWM5 111：Reserved
[7]	Reserved	Reserved	Reserved	Reserved
[6]	MUDIS	R/W	0x0	Master PWM 更新禁止 此位用于暂时禁止更新，软件需要写入并同时加载多个寄存器时需要暂时禁止。 0：允许更新 1：禁止更新
[5]	UDIS5	R/W	0x0	PWM5 更新禁止 此位用于暂时禁止更新，软件需要写入并同时加载多个寄存器时需要暂时禁止。 0：允许更新 1：禁止更新
[4]	UDIS4	R/W	0x0	PWM4 更新禁止 此位用于暂时禁止更新，软件需要写入并同时加载多个寄存器时需要暂时禁止。 0：允许更新 1：禁止更新
[3]	UDIS3	R/W	0x0	PWM3 更新禁止 此位用于暂时禁止更新，软件需要写入并同时加载多个寄存器时需要暂时禁止。 0：允许更新 1：禁止更新
[2]	UDIS2	R/W	0x0	PWM2 更新禁止 此位用于暂时禁止更新，软件需要写入并同时加载多个寄存器时需要暂时禁止。 0：允许更新 1：禁止更新
[1]	UDIS1	R/W	0x0	PWM1 更新禁止 此位用于暂时禁止更新，软件需要写入并同时加载多个寄存器时需要暂时禁止。 0：允许更新 1：禁止更新

[0]	UDIS0	R/W	0x0	PWM0 更新禁止 此位用于暂时禁止更新，软件需要写入并同时加载多个寄存器时需要暂时禁止。 0：允许更新 1：禁止更新

19.5.2.32 HRPWM Control Register1 (HRPWM_CR1)

- **Name**：HRPWM Control Register 1
- **Size**：32bits
- **Offset**：0x384
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-22]	Reserved	Reserved	Reserved	Reserved
[21]	SWP5	R/W	0x0	PWM5 输出交换 此位用于交换 PWM5 输出，注意当推挽模式开启时（PSHPLL＝1），此位无意义。 0：SET5A->SET5B，SET5B->SET5A，CLR5A->CLR5B，CLR5B->CLR5A 1：SET5A->SET5B，SET5B->SET5A，CLR5A->CLR5B，CLR5B->CLR5A
[20]	SWP4	R/W	0x0	PWM4 输出交换 此位用于交换 PWM4 输出，注意当推挽模式开启时（PSHPLL＝1），此位无意义。 0：SET4A->SET4B，SET4B->SET4A，CLR4A->CLR4B，CLR4B->CLR4A 1：SET4A->SET4B，SET4B->SET4A，CLR4A->CLR4B，CLR4B->CLR4A
[19]	SWP3	R/W	0x0	PWM3 输出交换 此位用于交换 PWM3 输出，注意当推挽模式开启时（PSHPLL＝1），此位无意义。 0：SET3A->SET3B，SET3B->SET3A，CLR3A->CLR3B，CLR3B->CLR3A 1：SET3A->SET3B，SET3B->SET3A，CLR3A->CLR3B，CLR3B->CLR3A
[18]	SWP2	R/W	0x0	PWM2 输出交换 此位用于交换 PWM2 输出，注意当推挽模式开启时（PSHPLL＝1），此位无意义。 0：SET2A->SET2B，SET2B->SET2A，CLR2A->CLR2B，CLR2B->CLR2A 1：SET2A->SET2B，SET2B->SET2A，CLR2A->CLR2B，CLR2B->CLR2A
[17]	SWP1	R/W	0x0	PWM1 输出交换 此位用于交换 PWM1 输出，注意当推挽模式开启时（PSHPLL＝1），此位无意义。 0：SET1A->SET1B，SET1B->SET1A，CLR1A->CLR1B，CLR1B->CLR1A 1：SET1A->SET1B，SET1B->SET1A，CLR1A->CLR1B，CLR1B->CLR1A
[16]	SWP0	R/W	0x0	PWM0 输出交换 此位用于交换 PWM0 输出，注意当推挽模式开启时（PSHPLL＝1），此位无意义。 0：SET0A->SET0B，SET0B->SET0A，CLR0A->CLR0B，CLR0B->CLR0A 1：SET0A->SET0B，SET0B->SET0A，CLR0A->CLR0B，CLR0B->CLR0A
[15]	Reserved	Reserved	Reserved	Reserved
[14]	MRST	R/WAC	0x0	Master PWM 软件复位 此位写 1 强制将计数器复位，复位完成后硬件自动清零，写 0 没有效果。 0：软件不复位计数器 1：软件复位计数器

[13]	RST5	R/WAC	0x0	PWM5 软件复位 此位写 1 强制将计数器复位，复位完成后硬件自动清零，写 0 没有效果。 0：软件不复位计数器 1：软件复位计数器
[12]	RST4	R/WAC	0x0	PWM4 软件复位 此位写 1 强制将计数器复位，复位完成后硬件自动清零，写 0 没有效果。 0：软件不复位计数器 1：软件复位计数器
[11]	RST3	R/WAC	0x0	PWM3 软件复位 此位写 1 强制将计数器复位，复位完成后硬件自动清零，写 0 没有效果。 0：软件不复位计数器 1：软件复位计数器
[10]	RST2	R/WAC	0x0	PWM2 软件复位 此位写 1 强制将计数器复位，复位完成后硬件自动清零，写 0 没有效果。 0：软件不复位计数器 1：软件复位计数器
[9]	RST1	R/WAC	0x0	PWM1 软件复位 此位写 1 强制将计数器复位，复位完成后硬件自动清零，写 0 没有效果。 0：软件不复位计数器 1：软件复位计数器
[8]	RST0	R/WAC	0x0	PWM0 软件复位 此位写 1 强制将计数器复位，复位完成后硬件自动清零，写 0 没有效果。 0：软件不复位计数器 1：软件复位计数器
[7]	Reserved	Reserved	Reserved	Reserved
[6]	MSWU	R/WAC	0x0	Master PWM 软件更新 此位写 1 强制将影子寄存器更新到工作寄存器，更新完成后硬件自动清零，写 0 没有效果。 0：软件不更新工作寄存器 1：软件更新工作寄存器
[5]	SWU5	R/WAC	0x0	PWM5 软件更新 此位写 1 强制将影子寄存器更新到工作寄存器，更新完成后硬件自动清零，写 0 没有效果。 0：软件不更新工作寄存器 1：软件更新工作寄存器
[4]	SWU4	R/WAC	0x0	PWM4 软件更新 此位写 1 强制将影子寄存器更新到工作寄存器，更新完成后硬件自动清零，写 0 没有效果。 0：软件不更新工作寄存器 1：软件更新工作寄存器

[3]	SWU3	R/WAC	0x0	PWM3 软件更新 此位写 1 强制将影子寄存器更新到工作寄存器，更新完成后硬件自动清零，写 0 没有效果。 0：软件不更新工作寄存器 1：软件更新工作寄存器
[2]	SWU2	R/WAC	0x0	PWM2 软件更新 此位写 1 强制将影子寄存器更新到工作寄存器，更新完成后硬件自动清零，写 0 没有效果。 0：软件不更新工作寄存器 1：软件更新工作寄存器
[1]	SWU1	R/WAC	0x0	PWM1 软件更新 此位写 1 强制将影子寄存器更新到工作寄存器，更新完成后硬件自动清零，写 0 没有效果。 0：软件不更新工作寄存器 1：软件更新工作寄存器
[0]	SWU0	R/WAC	Reserved	PWM0 软件更新 此位写 1 强制将影子寄存器更新到工作寄存器，更新完成后硬件自动清零，写 0 没有效果。 0：软件不更新工作寄存器 1：软件更新工作寄存器

19.5.2.33 HRPWM Control Register2 (HRPWM_CR2)

- **Name**：HRPWM Control Register 2
- **Size**：32bits
- **Offset**：0x388
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-28]	TLEN7	R/W	0x0	Adc Trg 事件长度 0000：Adc Trg 为 1 个 hrpwm_clk 周期 0001：Adc Trg 为 2 个 hrpwm_clk 周期 0010：Adc Trg 为 3 个 hrpwm_clk 周期 …… 1110：Adc Trg 为 15 个 hrpwm_clk 周期 1111：Adc Trg 为 16 个 hrpwm_clk 周期
[27-24]	TLEN6	R/W	0x0	Adc Trg 事件长度 0000：Adc Trg 为 1 个 hrpwm_clk 周期 0001：Adc Trg 为 2 个 hrpwm_clk 周期 0010：Adc Trg 为 3 个 hrpwm_clk 周期 …… 1110：Adc Trg 为 15 个 hrpwm_clk 周期 1111：Adc Trg 为 16 个 hrpwm_clk 周期
[23-20]	TLEN5	R/W	0x0	Adc Trg 事件长度 0000：Adc Trg 为 1 个 hrpwm_clk 周期 0001：Adc Trg 为 2 个 hrpwm_clk 周期 0010：Adc Trg 为 3 个 hrpwm_clk 周期 …… 1110：Adc Trg 为 15 个 hrpwm_clk 周期 1111：Adc Trg 为 16 个 hrpwm_clk 周期
[19-16]	TLEN4	R/W	0x0	Adc Trg 事件长度 0000：Adc Trg 为 1 个 hrpwm_clk 周期 0001：Adc Trg 为 2 个 hrpwm_clk 周期 0010：Adc Trg 为 3 个 hrpwm_clk 周期 …… 1110：Adc Trg 为 15 个 hrpwm_clk 周期 1111：Adc Trg 为 16 个 hrpwm_clk 周期
[15-12]	TLEN3	R/W	0x0	Adc Trg 事件长度 0000：Adc Trg 为 1 个 hrpwm_clk 周期 0001：Adc Trg 为 2 个 hrpwm_clk 周期 0010：Adc Trg 为 3 个 hrpwm_clk 周期 …… 1110：Adc Trg 为 15 个 hrpwm_clk 周期 1111：Adc Trg 为 16 个 hrpwm_clk 周期

[11-8]	TLEN2	R/W	0x0	Adc Trg 事件长度 0000：Adc Trg 为 1 个 hrpwm_clk 周期 0001：Adc Trg 为 2 个 hrpwm_clk 周期 0010：Adc Trg 为 3 个 hrpwm_clk 周期 …… 1110：Adc Trg 为 15 个 hrpwm_clk 周期 1111：Adc Trg 为 16 个 hrpwm_clk 周期
[7-4]	TLEN1	R/W	0x0	Adc Trg 事件长度 0000：Adc Trg 为 1 个 hrpwm_clk 周期 0001：Adc Trg 为 2 个 hrpwm_clk 周期 0010：Adc Trg 为 3 个 hrpwm_clk 周期 …… 1110：Adc Trg 为 15 个 hrpwm_clk 周期 1111：Adc Trg 为 16 个 hrpwm_clk 周期
[3-0]	TLEN0	R/W	0x0	Adc Trg 事件长度 0000：Adc Trg 为 1 个 hrpwm_clk 周期 0001：Adc Trg 为 2 个 hrpwm_clk 周期 0010：Adc Trg 为 3 个 hrpwm_clk 周期 …… 1110：Adc Trg 为 15 个 hrpwm_clk 周期 1111：Adc Trg 为 16 个 hrpwm_clk 周期

19.5.2.34 HRPWM Interrupt Status Register (HRPWM_ISR)

- **Name**：HRPWM Interrupt Status Register
- **Size**：32bits
- **Offset**：0x38C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-7]	Reserved	Reserved	Reserved	Reserved
[6]	SYSFLT	R/W1C	0x0	System Fault 中断标志 写 1 清除中断标志。 0：System Fault 中断标志未产生 1：System Fault 中断标志产生
[5]	FLT5	R/W1C	Reserved	Fault 5 中断标志 写 1 清除中断标志。 0：Fault 5 中断标志未产生 1：Fault 5 中断标志产生
[4]	FLT4	R/W1C	0x0	Fault 4 中断标志 写 1 清除中断标志。 0：Fault 4 中断标志未产生 1：Fault 4 中断标志产生
[3]	FLT3	R/W1C	Reserved	Fault 3 中断标志 写 1 清除中断标志。 0：Fault 3 中断标志未产生 1：Fault 3 中断标志产生
[2]	FLT2	R/W1C	0x0	Fault 2 中断标志 写 1 清除中断标志。 0：Fault 2 中断标志未产生 1：Fault 2 中断标志产生
[2]	FLT1	R/W1C	Reserved	Fault 1 中断标志 写 1 清除中断标志。 0：Fault 1 中断标志未产生 1：Fault 1 中断标志产生
[0]	FLT0	R/W1C	0x0	Fault 0 中断标志 写 1 清除中断标志。 0：Fault 0 中断标志未产生 1：Fault 0 中断标志产生

19.5.2.35 HRPWM Interrupt Enable Register (HRPWM_IER)

- **Name**：HRPWM Interrupt Enable Register
- **Size**：32bits
- **Offset**：0x390
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-7]	Reserved	Reserved	Reserved	Reserved
[6]	SYSFLTIE	R/W	0x0	System Fault 中断使能 0：System Fault 中断不使能 1：System Fault 中断使能
[5]	FLT5IE	R/W	0x0	Fault 5 中断使能 0：Fault 5 中断不使能 1：Fault 5 中断使能
[4]	FLT4IE	R/W	0x0	Fault 4 中断使能 0：Fault 4 中断不使能 1：Fault 4 中断使能
[3]	FLT3IE	R/W	0x0	Fault 3 中断使能 0：Fault 3 中断不使能 1：Fault 3 中断使能
[2]	FLT2IE	R/W	0x0	Fault 2 中断使能 0：Fault 2 中断不使能 1：Fault 2 中断使能
[1]	FLT1IE	R/W	0x0	Fault 1 中断使能 0：Fault 1 中断不使能 1：Fault 1 中断使能
[0]	FLT0IE	R/W	0x0	Fault 0 中断使能 0：Fault 0 中断不使能 1：Fault 0 中断使能

19.5.2.36 HRPWM Output Enable Register (HRPWM_OENR)

- **Name**：HRPWM Output Enable Register
- **Size**：32bits
- **Offset**：0x394
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	OEN5B	R/W1S	0x0	PWM5 Out B 启动 此位写 1 启动输出 B，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS5B 读出 0：Out B 停止，可能处在故障状态或空闲状态 1：Out B 启动
[10]	OEN5A	R/W1S	0x0	PWM5 Out A 启动 此位写 1 启动输出 A，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS5A 读出 0：Out A 停止，可能处在故障状态或空闲状态 1：Out A 启动
[9]	OEN4B	R/W1S	0x0	PWM4 Out B 启动 此位写 1 启动输出 B，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS4B 读出 0：Out B 停止，可能处在故障状态或空闲状态 1：Out B 启动
[8]	OEN4A	R/W1S	0x0	PWM4 Out A 启动 此位写 1 启动输出 A，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS4A 读出 0：Out A 停止，可能处在故障状态或空闲状态 1：Out A 启动
[7]	OEN3B	R/W1S	0x0	PWM3 Out B 启动 此位写 1 启动输出 B，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS3B 读出 0：Out B 停止，可能处在故障状态或空闲状态 1：Out B 启动

[6]	OEN3A	R/W1S	0x0	PWM3 Out A 启动 此位写 1 启动输出 A，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS3A 读出 0：Out A 停止，可能处在故障状态或空闲状态 1：Out A 启动
[5]	OEN2B	R/W1S	0x0	PWM2 Out B 启动 此位写 1 启动输出 B，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS2B 读出 0：Out B 停止，可能处在故障状态或空闲状态 1：Out B 启动
[4]	OEN2A	R/W1S	0x0	PWM2 Out A 启动 此位写 1 启动输出 A，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS2A 读出 0：Out A 停止，可能处在故障状态或空闲状态 1：Out A 启动
[3]	OEN1B	R/W1S	0x0	PWM1 Out B 启动 此位写 1 启动输出 B，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS1B 读出 0：Out B 停止，可能处在故障状态或空闲状态 1：Out B 启动
[2]	OEN1A	R/W1S	0x0	PWM1 Out A 启动 此位写 1 启动输出 A，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS1A 读出 0：Out A 停止，可能处在故障状态或空闲状态 1：Out A 启动
[1]	OEN0B	R/W1S	0x0	PWM0 Out B 启动 此位写 1 启动输出 B，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS0B 读出 0：Out B 停止，可能处在故障状态或空闲状态 1：Out B 启动

[0]	OEN0A	R/W1S	0x0	PWM0 Out A 启动 此位写 1 启动输出 A，输出从空闲状态或故障状态进入启动状态，此位写 0 没有效果 读此位返回输出启动状态，输出停止状态可能是故障或空闲状态，由 ODIS0A 读出 0：Out A 停止，可能处在故障状态或空闲状态 1：Out A 启动

19.5.2.37 HRPWM Output Disable Register (HRPWM_ODISR)

- **Name**：HRPWM Output Disable Register
- **Size**：32bits
- **Offset**：0x398
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	ODIS5B	R/W1S	0x0	PWM5 Out B 停止 此位写 1 停止输出 B，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out B 停止，在空闲状态 1：Out B 停止，在故障状态
[10]	ODIS5A	R/W1S	0x0	PWM5 Out A 停止 此位写 1 停止输出 A，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out A 停止，在空闲状态 1：Out A 停止，在故障状态
[9]	ODIS4B	R/W1S	0x0	PWM4 Out B 停止 此位写 1 停止输出 B，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out B 停止，在空闲状态 1：Out B 停止，在故障状态
[8]	ODIS4A	R/W1S	0x0	PWM4 Out A 停止 此位写 1 停止输出 A，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out A 停止，在空闲状态 1：Out A 停止，在故障状态
[7]	ODIS3B	R/W1S	0x0	PWM3 Out B 停止 此位写 1 停止输出 B，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out B 停止，在空闲状态 1：Out B 停止，在故障状态

[6]	ODIS3A	R/W1S	0x0	PWM3 Out A 停止 此位写 1 停止输出 A，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out A 停止，在空闲状态 1：Out A 停止，在故障状态
[5]	ODIS2B	R/W1S	0x0	PWM2 Out B 停止 此位写 1 停止输出 B，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out B 停止，在空闲状态 1：Out B 停止，在故障状态
[4]	ODIS2A	R/W1S	0x0	PWM2 Out A 停止 此位写 1 停止输出 A，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out A 停止，在空闲状态 1：Out A 停止，在故障状态
[3]	ODIS1B	R/W1S	0x0	PWM1 Out B 停止 此位写 1 停止输出 B，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out B 停止，在空闲状态 1：Out B 停止，在故障状态
[2]	ODIS1A	R/W1S	0x0	PWM1 Out A 停止 此位写 1 停止输出 A，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out A 停止，在空闲状态 1：Out A 停止，在故障状态
[1]	ODIS0B	R/W1S	0x0	PWM0 Out B 停止 此位写 1 停止输出 B，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out B 停止，在空闲状态 1：Out B 停止，在故障状态
[0]	ODIS0A	R/W1S	0x0	PWM0 Out A 停止 此位写 1 停止输出 A，输出从运行状态或故障状态进入空闲状态，此位写 0 没有效果 读此位返回输出停止状态，PWM 输出有效时此位无意义 0：Out A 停止，在空闲状态 1：Out A 停止，在故障状态

19.5.2.38 HRPWM External Event Register0 (HRPWM_EECR0)

- **Name**：HRPWM External Event Register0
- **Size**：32bits
- **Offset**：0x39C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-30]	Reserved	Reserved	Reserved	Reserved
[29]	EE4FAST	R/W1S	0x0	Event 4 快速模式 0：Event 4 同步之后作用于输出 1：Event 4 异步作用于输出
[28-27]	EE4SNS	R/W1S	0x0	Event 4 输入有效沿 00：电平有效 01：上升沿有效 10：下降沿有效 11：上升 / 下降沿有效
[26]	EE4POL	R/W1S	0x0	Event 4 输入极性 0： Event 4 输入高有效 1： Event 4 输入低有效
[25-24]	EE3SRC	R/W1S	0x0	Event 4 输入来源 00：来源 0 01：来源 1 10：来源 2 11：来源 3
[23]	EE3FAST	R/W1S	0x0	Event 3 快速模式 0：Event 3 同步之后作用于输出 1：Event 3 异步作用于输出
[22-21]	EE3SNS	R/W1S	0x0	Event 3 输入有效沿 00：电平有效 01：上升沿有效 10：下降沿有效 11：上升 / 下降沿有效
[20]	EE3POL	R/W1S	0x0	Event 3 输入极性 0： Event 3 输入高有效 1： Event 3 输入低有效
[19-18]	EE3SRC	R/W1S	0x0	Event 3 输入来源 00：来源 0 01：来源 1 10：来源 2 11：来源 3
[17]	EE2FAST	R/W1S	0x0	Event 2 快速模式 0：Event 2 同步之后作用于输出 1：Event 2 异步作用于输出

[16-15]	EE2SNS	R/W1S	0x0	Event 2 输入有效沿 00：电平有效 01：上升沿有效 10：下降沿有效 11：上升 / 下降沿有效
[14]	EE2POL	R/W1S	0x0	Event 2 输入极性 0： Event 3 输入高有效 1： Event 3 输入低有效
[13-12]	EE2SRC	R/W1S	0x0	Event 2 输入来源 00：来源 0 01：来源 1 10：来源 2 11：来源 3
[11]	EE1FAST	R/W1S	0x0	Event 1 快速模式 0：Event 1 同步之后作用于输出 1：Event 1 异步作用于输出
[10-9]	EE1SNS	R/W1S	0x0	Event 1 输入有效沿 00：电平有效 01：上升沿有效 10：下降沿有效 11：上升 / 下降沿有效
[8]	EE1POL	R/W1S	0x0	Event 1 输入极性 0： Event 3 输入高有效 1： Event 3 输入低有效
[7-6]	EE1SRC	R/W1S	0x0	Event 1 输入来源 00：来源 0 01：来源 1 10：来源 2 11：来源 3
[5]	EE0FAST	R/W1S	0x0	Event 0 快速模式 0：Event 0 同步之后作用于输出 1：Event 0 异步作用于输出
[4-3]	EE0SNS	R/W1S	0x0	Event 0 输入有效沿 00：电平有效 01：上升沿有效 10：下降沿有效 11：上升 / 下降沿有效
[2]	EE0POL	R/W1S	0x0	Event 0 输入极性 0： Event 3 输入高有效 1： Event 3 输入低有效

				Event 0 输入来源 00：来源 0 01：来源 1 10：来源 2 11：来源 3
[1-0]	EE0SRC	R/W1S	0x0	

19.5.2.39 HRPWM External Event Register1 (HRPWM_EECR1)

- **Name**：HRPWM External Event Register1
- **Size**：32bits
- **Offset**：0x3A0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-6]	Reserved	Reserved	Reserved	Reserved
[5]	EE5FAST	R/W	0x0	Event 5 快速模式 0：Event 5 同步之后作用于输出 1：Event 5 异步作用于输出
[4-3]	EE5SNS	R/W	0x0	Event 5 输入有效沿 00：电平有效 01：上升沿有效 10：下降沿有效 11：上升 / 下降沿有效
[2]	EE5POL	R/W	0x0	Event 5 输入极性 0：Event 5 输入高有效 1：Event 5 输入低有效
[1-0]	EE5SRC	R/W	0x0	Event 5 输入来源 00：来源 0 01：来源 1 10：来源 2 11：来源 3

19.5.2.40 HRPWM External Event Register2 (HRPWM_EECR2)

- **Name**：HRPWM External Event Register2
- **Size**：32bits
- **Offset**：0x3A4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-30]	EEVSD	R/W	0x0	Event 采样时钟分频比例 此位决定 Event 采样时钟与 HRPWM 模块时钟的分频比例。 00：1 分频（f_sample = f_hrpwm/8） 01：2 分频（f_sample = f_hrpwm/4） 10：4 分频（f_sample = f_hrpwm/2） 11：8 分频（f_sample = f_hrpwm）
[29-24]	Reserved	Reserved	Reserved	Reserved
[23-20]	EE5F	R/W	0x0	Event 5 滤波长度 此位定义了用于 Event 5 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Event 信号抓一拍即输出
[19-16]	EE4F	R/W	0x0	Event 4 滤波长度 此位定义了用于 Event 4 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Event 信号抓一拍即输出
[15-12]	EE3F	R/W	0x0	Event 3 滤波长度 此位定义了用于 Event 3 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Event 信号抓一拍即输出
[11-8]	EE2F	R/W	0x0	Event 2 滤波长度 此位定义了用于 Event 2 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Event 信号抓一拍即输出
[7-4]	EE1F	R/W	0x0	Event 1 滤波长度 此位定义了用于 Event 1 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Event 信号抓一拍即输出

[3-0]	EE0F	R/W	0x0	Event 0 滤波长度 此位定义了用于 Event 0 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Event 信号抓一拍即输出

19.5.2.41 HRPWM ADC Trigger 0 Register (HRPWM_ADT0R)

- **Name**：HRPWM ADC Trigger 0 Register
- **Size**：32bits
- **Offset**：0x3A8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	CMPD5	R/W	0x0	此位使能后 PWM5 Compare D 事件会产生 Adc Trg 0
[30]	CMPC4	R/W	0x0	此位使能后 PWM5 Compare C 事件会产生 Adc Trg 0
[29]	PER4	R/W	0x0	此位使能后 PWM4 Period 事件会产生 Adc Trg 0
[28]	CMPD4	R/W	0x0	此位使能后 PWM4 Compare D 事件会产生 Adc Trg 0
[27]	CMPC4	R/W	0x0	此位使能后 PWM4 Compare C 事件会产生 Adc Trg 0
[26]	CMPB4	R/W	0x0	此位使能后 PWM4 Compare B 事件会产生 Adc Trg 0
[25]	RST3PER3	R/W	0x0	此位使能后 PWM3 Reset/Period 事件会产生 Adc Trg 0
[24]	CMPD3	R/W	0x0	此位使能后 PWM3 Compare D 事件会产生 Adc Trg 0
[23]	CMPC3	R/W	0x0	此位使能后 PWM3 Compare C 事件会产生 Adc Trg 0
[22]	CMPB3	R/W	0x0	此位使能后 PWM3 Compare B 事件会产生 Adc Trg 0
[21]	PER2	R/W	0x0	此位使能后 PWM2 Period 事件会产生 Adc Trg 0
[20]	CMPD2	R/W	0x0	此位使能后 PWM2 Compare D 事件会产生 Adc Trg 0
[19]	CMPC2	R/W	0x0	此位使能后 PWM2 Compare C 事件会产生 Adc Trg 0
[18]	CMPB2	R/W	0x0	此位使能后 PWM2 Compare B 事件会产生 Adc Trg 0
[17]	RST1PER1	R/W	0x0	此位使能后 PWM1 Reset/Period 事件会产生 Adc Trg 0
[16]	CMPD1	R/W	0x0	此位使能后 PWM1 Compare D 事件会产生 Adc Trg 0
[15]	CMPC1	R/W	0x0	此位使能后 PWM1 Compare C 事件会产生 Adc Trg 0
[14]	CMPB1	R/W	0x0	此位使能后 PWM1 Compare B 事件会产生 Adc Trg 0
[13]	PER0	R/W	0x0	此位使能后 PWM0 Period 事件会产生 Adc Trg 0
[12]	CMPD0	R/W	0x0	此位使能后 PWM0 Compare D 事件会产生 Adc Trg 0
[11]	CMPC0	R/W	0x0	此位使能后 PWM0 Compare C 事件会产生 Adc Trg 0
[10]	CMPB0	R/W	0x0	此位使能后 PWM0 Compare B 事件会产生 Adc Trg 0
[9]	RST5PER5	R/W	0x0	此位使能后 PWM5 Reset/Period 事件会产生 Adc Trg 0
[8]	CMPB5	R/W	0x0	此位使能后 PWM5 Compare B 事件会产生 Adc Trg 0
[7]	EEV2	R/W	0x0	此位使能后 External Event 2 事件会产生 Adc Trg 0
[6]	EEV1	R/W	0x0	此位使能后 External Event 1 事件会产生 Adc Trg 0
[5]	EEV0	R/W	0x0	此位使能后 External Event 0 事件会产生 Adc Trg 0
[4]	MPER	R/W	0x0	此位使能后 Master PWM Period 事件会产生 Adc Trg 0
[3]	MCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件会产生 Adc Trg 0
[2]	MCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件会产生 Adc Trg 0
[1]	MCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件会产生 Adc Trg 0
[0]	MCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件会产生 Adc Trg 0

19.5.2.42 HRPWM ADC Trigger 1 Register (HRPWM_ADT1R)

- **Name**：HRPWM ADC Trigger 1 Register
- **Size**：32bits
- **Offset**：0x3AC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	CMPD5	R/W	0x0	此位使能后 PWM5 Compare D 事件会产生 Adc Trg 1
[30]	CMPC5	R/W	0x0	此位使能后 PWM5 Compare C 事件会产生 Adc Trg 1
[29]	RST4PER4	R/W	0x0	此位使能后 PWM4 Reset/Period 事件会产生 Adc Trg 1
[28]	CMPD4	R/W	0x0	此位使能后 PWM4 Compare D 事件会产生 Adc Trg 1
[27]	CMPC4	R/W	0x0	此位使能后 PWM4 Compare C 事件会产生 Adc Trg 1
[26]	CMPB4	R/W	0x0	此位使能后 PWM4 Compare B 事件会产生 Adc Trg 1
[25]	PER3	R/W	0x0	此位使能后 PWM3 Period 事件会产生 Adc Trg 1
[24]	CMPD3	R/W	0x0	此位使能后 PWM3 Compare D 事件会产生 Adc Trg 1
[23]	CMPC3	R/W	0x0	此位使能后 PWM3 Compare C 事件会产生 Adc Trg 1
[22]	CMPB3	R/W	0x0	此位使能后 PWM3 Compare B 事件会产生 Adc Trg 1
[21]	RST2PER2	R/W	0x0	此位使能后 PWM2 Reset/Period 事件会产生 Adc Trg 1
[20]	CMPD2	R/W	0x0	此位使能后 PWM2 Compare D 事件会产生 Adc Trg 1
[19]	CMPC2	R/W	0x0	此位使能后 PWM2 Compare C 事件会产生 Adc Trg 1
[18]	CMPB2	R/W	0x0	此位使能后 PWM2 Compare B 事件会产生 Adc Trg 1
[17]	PER1	R/W	0x0	此位使能后 PWM1 Period 事件会产生 Adc Trg 1
[16]	CMPD1	R/W	0x0	此位使能后 PWM1 Compare D 事件会产生 Adc Trg 1
[15]	CMPC1	R/W	0x0	此位使能后 PWM1 Compare C 事件会产生 Adc Trg 1
[14]	CMPB1	R/W	0x0	此位使能后 PWM1 Compare B 事件会产生 Adc Trg 1
[13]	RST0PER0	R/W	0x0	此位使能后 PWM0 Reset/Period 事件会产生 Adc Trg 1
[12]	CMPD0	R/W	0x0	此位使能后 PWM0 Compare D 事件会产生 Adc Trg 1
[11]	CMPC0	R/W	0x0	此位使能后 PWM0 Compare C 事件会产生 Adc Trg 1
[10]	CMPB0	R/W	0x0	此位使能后 PWM0 Compare B 事件会产生 Adc Trg 1
[9]	PER5	R/W	0x0	此位使能后 PWM5 Period 事件会产生 Adc Trg 1
[8]	CMPB5	R/W	0x0	此位使能后 PWM5 Compare B 事件会产生 Adc Trg 1
[7]	EEV5	R/W	0x0	此位使能后 External Event 5 事件会产生 Adc Trg 1
[6]	EEV4	R/W	0x0	此位使能后 External Event 4 事件会产生 Adc Trg 1
[5]	EEV3	R/W	0x0	此位使能后 External Event 3 事件会产生 Adc Trg 1
[4]	MPER	R/W	0x0	此位使能后 Master PWM Period 事件会产生 Adc Trg 1
[3]	MCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件会产生 Adc Trg 1
[2]	MCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件会产生 Adc Trg 1
[1]	MCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件会产生 Adc Trg 1
[0]	MCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件会产生 Adc Trg 1

19.5.2.43 HRPWM ADC Trigger 2 Register (HRPWM_ADT2R)

- **Name**：HRPWM ADC Trigger 2 Register
- **Size**：32bits
- **Offset**：0x3B0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	CMPD5	R/W	0x0	此位使能后 PWM5 Compare D 事件会产生 Adc Trg 2
[30]	CMPC5	R/W	0x0	此位使能后 PWM5 Compare C 事件会产生 Adc Trg 2
[29]	PER4	R/W	0x0	此位使能后 PWM4 Period 事件会产生 Adc Trg 2
[28]	CMPD4	R/W	0x0	此位使能后 PWM4 Compare D 事件会产生 Adc Trg 2
[27]	CMPC4	R/W	0x0	此位使能后 PWM4 Compare C 事件会产生 Adc Trg 2
[26]	CMPB4	R/W	0x0	此位使能后 PWM4 Compare B 事件会产生 Adc Trg 2
[25]	RST3PER3	R/W	0x0	此位使能后 PWM3 Reset/Period 事件会产生 Adc Trg 2
[24]	CMPD3	R/W	0x0	此位使能后 PWM3 Compare D 事件会产生 Adc Trg 2
[23]	CMPC3	R/W	0x0	此位使能后 PWM3 Compare C 事件会产生 Adc Trg 2
[22]	CMPB3	R/W	0x0	此位使能后 PWM3 Compare B 事件会产生 Adc Trg 2
[21]	PER2	R/W	0x0	此位使能后 PWM2 Period 事件会产生 Adc Trg 2
[20]	CMPD2	R/W	0x0	此位使能后 PWM2 Compare D 事件会产生 Adc Trg 2
[19]	CMPC2	R/W	0x0	此位使能后 PWM2 Compare C 事件会产生 Adc Trg 2
[18]	CMPB2	R/W	0x0	此位使能后 PWM2 Compare B 事件会产生 Adc Trg 2
[17]	RST1PER1	R/W	0x0	此位使能后 PWM1 Reset/Period 事件会产生 Adc Trg 2
[16]	CMPD1	R/W	0x0	此位使能后 PWM1 Compare D 事件会产生 Adc Trg 2
[15]	CMPC1	R/W	0x0	此位使能后 PWM1 Compare C 事件会产生 Adc Trg 2
[14]	CMPB1	R/W	0x0	此位使能后 PWM1 Compare B 事件会产生 Adc Trg 2
[13]	PER0	R/W	0x0	此位使能后 PWM0 Period 事件会产生 Adc Trg 2
[12]	CMPD0	R/W	0x0	此位使能后 PWM0 Compare D 事件会产生 Adc Trg 2
[11]	CMPC0	R/W	0x0	此位使能后 PWM0 Compare C 事件会产生 Adc Trg 2
[10]	CMPB0	R/W	0x0	此位使能后 PWM0 Compare B 事件会产生 Adc Trg 2
[9]	RST5PER5	R/W	0x0	此位使能后 PWM5 Reset/Period 事件会产生 Adc Trg 2
[8]	CMPB5	R/W	0x0	此位使能后 PWM5 Compare B 事件会产生 Adc Trg 2
[7]	EEV2	R/W	0x0	此位使能后 External Event 2 事件会产生 Adc Trg 2
[6]	EEV1	R/W	0x0	此位使能后 External Event 1 事件会产生 Adc Trg 2
[5]	EEV0	R/W	0x0	此位使能后 External Event 0 事件会产生 Adc Trg 2
[4]	MPER	R/W	0x0	此位使能后 Master PWM Period 事件会产生 Adc Trg 2
[3]	MCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件会产生 Adc Trg 2
[2]	MCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件会产生 Adc Trg 2
[1]	MCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件会产生 Adc Trg 2
[0]	MCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件会产生 Adc Trg 2

19.5.2.44 HRPWM ADC Trigger 3 Register (HRPWM_ADT3R)

- **Name**：HRPWM ADC Trigger 3 Register
- **Size**：32bits
- **Offset**：0x3B4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	CMPD5	R/W	0x0	此位使能后 PWM5 Compare D 事件会产生 Adc Trg 3
[30]	CMPC5	R/W	0x0	此位使能后 PWM5 Compare C 事件会产生 Adc Trg 3
[29]	RST4PER4	R/W	0x0	此位使能后 PWM4 Reset/Period 事件会产生 Adc Trg 3
[28]	CMPD4	R/W	0x0	此位使能后 PWM4 Compare D 事件会产生 Adc Trg 3
[27]	CMPC4	R/W	0x0	此位使能后 PWM4 Compare C 事件会产生 Adc Trg 3
[26]	CMPB4	R/W	0x0	此位使能后 PWM4 Compare B 事件会产生 Adc Trg 3
[25]	PER3	R/W	0x0	此位使能后 PWM3 Period 事件会产生 Adc Trg 3
[24]	CMPD3	R/W	0x0	此位使能后 PWM3 Compare D 事件会产生 Adc Trg 3
[23]	CMPC3	R/W	0x0	此位使能后 PWM3 Compare C 事件会产生 Adc Trg 3
[22]	CMPB3	R/W	0x0	此位使能后 PWM3 Compare B 事件会产生 Adc Trg 3
[21]	RST2PER2	R/W	0x0	此位使能后 PWM2 Reset/Period 事件会产生 Adc Trg 3
[20]	CMPD2	R/W	0x0	此位使能后 PWM2 Compare D 事件会产生 Adc Trg 3
[19]	CMPC2	R/W	0x0	此位使能后 PWM2 Compare C 事件会产生 Adc Trg 3
[18]	CMPB2	R/W	0x0	此位使能后 PWM2 Compare B 事件会产生 Adc Trg 3
[17]	PER1	R/W	0x0	此位使能后 PWM1 Period 事件会产生 Adc Trg 3
[16]	CMPD1	R/W	0x0	此位使能后 PWM1 Compare D 事件会产生 Adc Trg 3
[15]	CMPC1	R/W	0x0	此位使能后 PWM1 Compare C 事件会产生 Adc Trg 3
[14]	CMPB1	R/W	0x0	此位使能后 PWM1 Compare B 事件会产生 Adc Trg 3
[13]	RST0PER0	R/W	0x0	此位使能后 PWM0 Reset/Period 事件会产生 Adc Trg 3
[12]	CMPD0	R/W	0x0	此位使能后 PWM0 Compare D 事件会产生 Adc Trg 3
[11]	CMPC0	R/W	0x0	此位使能后 PWM0 Compare C 事件会产生 Adc Trg 3
[10]	CMPB0	R/W	0x0	此位使能后 PWM0 Compare B 事件会产生 Adc Trg 3
[9]	PER5	R/W	0x0	此位使能后 PWM5 Period 事件会产生 Adc Trg 3
[8]	CMPB5	R/W	0x0	此位使能后 PWM5 Compare B 事件会产生 Adc Trg 3
[7]	EEV5	R/W	0x0	此位使能后 External Event 5 事件会产生 Adc Trg 3
[6]	EEV4	R/W	0x0	此位使能后 External Event 4 事件会产生 Adc Trg 3
[5]	EEV3	R/W	0x0	此位使能后 External Event 3 事件会产生 Adc Trg 3
[4]	MPER	R/W	0x0	此位使能后 Master PWM Period 事件会产生 Adc Trg 3
[3]	MCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件会产生 Adc Trg 3
[2]	MCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件会产生 Adc Trg 3
[1]	MCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件会产生 Adc Trg 3
[0]	MCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件会产生 Adc Trg 3

19.5.2.45 HRPWM ADC Trigger 4 Register (HRPWM_ADT4R)

- **Name**：HRPWM ADC Trigger 4 Register
- **Size**：32bits
- **Offset**：0x3B8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	CMPD5	R/W	0x0	此位使能后 PWM5 Compare D 事件会产生 Adc Trg 4
[30]	CMPC5	R/W	0x0	此位使能后 PWM5 Compare C 事件会产生 Adc Trg 4
[29]	PER4	R/W	0x0	此位使能后 PWM4 Period 事件会产生 Adc Trg 4
[28]	CMPD4	R/W	0x0	此位使能后 PWM4 Compare D 事件会产生 Adc Trg 4
[27]	CMPC4	R/W	0x0	此位使能后 PWM4 Compare C 事件会产生 Adc Trg 4
[26]	CMPB4	R/W	0x0	此位使能后 PWM4 Compare B 事件会产生 Adc Trg 4
[25]	RST3PER3	R/W	0x0	此位使能后 PWM3 Reset/Period 事件会产生 Adc Trg 4
[24]	CMPD3	R/W	0x0	此位使能后 PWM3 Compare D 事件会产生 Adc Trg 4
[23]	CMPC3	R/W	0x0	此位使能后 PWM3 Compare C 事件会产生 Adc Trg 4
[22]	CMPB3	R/W	0x0	此位使能后 PWM3 Compare B 事件会产生 Adc Trg 4
[21]	PER2	R/W	0x0	此位使能后 PWM2 Period 事件会产生 Adc Trg 4
[20]	CMPD2	R/W	0x0	此位使能后 PWM2 Compare D 事件会产生 Adc Trg 4
[19]	CMPC2	R/W	0x0	此位使能后 PWM2 Compare C 事件会产生 Adc Trg 4
[18]	CMPB2	R/W	0x0	此位使能后 PWM2 Compare B 事件会产生 Adc Trg 4
[17]	RST1PER1	R/W	0x0	此位使能后 PWM1 Reset/Period 事件会产生 Adc Trg 4
[16]	CMPD1	R/W	0x0	此位使能后 PWM1 Compare D 事件会产生 Adc Trg 4
[15]	CMPC1	R/W	0x0	此位使能后 PWM1 Compare C 事件会产生 Adc Trg 4
[14]	CMPB1	R/W	0x0	此位使能后 PWM1 Compare B 事件会产生 Adc Trg 4
[13]	PER0	R/W	0x0	此位使能后 PWM0 Period 事件会产生 Adc Trg 4
[12]	CMPD0	R/W	0x0	此位使能后 PWM0 Compare D 事件会产生 Adc Trg 4
[11]	CMPC0	R/W	0x0	此位使能后 PWM0 Compare C 事件会产生 Adc Trg 4
[10]	CMPB0	R/W	0x0	此位使能后 PWM0 Compare B 事件会产生 Adc Trg 4
[9]	RST5PER5	R/W	0x0	此位使能后 PWM5 Reset/Period 事件会产生 Adc Trg 4
[8]	CMPB5	R/W	0x0	此位使能后 PWM5 Compare B 事件会产生 Adc Trg 4
[7]	EEV2	R/W	0x0	此位使能后 External Event 2 事件会产生 Adc Trg 4
[6]	EEV1	R/W	0x0	此位使能后 External Event 1 事件会产生 Adc Trg 4
[5]	EEV0	R/W	0x0	此位使能后 External Event 0 事件会产生 Adc Trg 4
[4]	MPER	R/W	0x0	此位使能后 Master PWM Period 事件会产生 Adc Trg 4
[3]	MCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件会产生 Adc Trg 4
[2]	MCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件会产生 Adc Trg 4
[1]	MCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件会产生 Adc Trg 4
[0]	MCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件会产生 Adc Trg 4

19.5.2.46 HRPWM ADC Trigger 5 Register (HRPWM_ADT5R)

- **Name**：HRPWM ADC Trigger 5 Register
- **Size**：32bits
- **Offset**：0x3BC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	CMPD5	R/W	0x0	此位使能后 PWM5 Compare D 事件会产生 Adc Trg 5
[30]	CMPC5	R/W	0x0	此位使能后 PWM5 Compare C 事件会产生 Adc Trg 5
[29]	RST4PER4	R/W	0x0	此位使能后 PWM4 Reset/Period 事件会产生 Adc Trg 5
[28]	CMPD4	R/W	0x0	此位使能后 PWM4 Compare D 事件会产生 Adc Trg 5
[27]	CMPC4	R/W	0x0	此位使能后 PWM4 Compare C 事件会产生 Adc Trg 5
[26]	CMPB4	R/W	0x0	此位使能后 PWM4 Compare B 事件会产生 Adc Trg 5
[25]	PER3	R/W	0x0	此位使能后 PWM3 Period 事件会产生 Adc Trg 5
[24]	CMPD3	R/W	0x0	此位使能后 PWM3 Compare D 事件会产生 Adc Trg 5
[23]	CMPC3	R/W	0x0	此位使能后 PWM3 Compare C 事件会产生 Adc Trg 5
[22]	CMPB3	R/W	0x0	此位使能后 PWM3 Compare B 事件会产生 Adc Trg 5
[21]	RST2PER2	R/W	0x0	此位使能后 PWM2 Reset/Period 事件会产生 Adc Trg 5
[20]	CMPD2	R/W	0x0	此位使能后 PWM2 Compare D 事件会产生 Adc Trg 5
[19]	CMPC2	R/W	0x0	此位使能后 PWM2 Compare C 事件会产生 Adc Trg 5
[18]	CMPB2	R/W	0x0	此位使能后 PWM2 Compare B 事件会产生 Adc Trg 5
[17]	PER1	R/W	0x0	此位使能后 PWM1 Period 事件会产生 Adc Trg 5
[16]	CMPD1	R/W	0x0	此位使能后 PWM1 Compare D 事件会产生 Adc Trg 5
[15]	CMPC1	R/W	0x0	此位使能后 PWM1 Compare C 事件会产生 Adc Trg 5
[14]	CMPB1	R/W	0x0	此位使能后 PWM1 Compare B 事件会产生 Adc Trg 5
[13]	RST0PER0	R/W	0x0	此位使能后 PWM0 Reset/Period 事件会产生 Adc Trg 5
[12]	CMPD0	R/W	0x0	此位使能后 PWM0 Compare D 事件会产生 Adc Trg 5
[11]	CMPC0	R/W	0x0	此位使能后 PWM0 Compare C 事件会产生 Adc Trg 5
[10]	CMPB0	R/W	0x0	此位使能后 PWM0 Compare B 事件会产生 Adc Trg 5
[9]	PER5	R/W	0x0	此位使能后 PWM5 Period 事件会产生 Adc Trg 5
[8]	CMPB5	R/W	0x0	此位使能后 PWM5 Compare B 事件会产生 Adc Trg 5
[7]	EEV5	R/W	0x0	此位使能后 External Event 5 事件会产生 Adc Trg 5
[6]	EEV4	R/W	0x0	此位使能后 External Event 4 事件会产生 Adc Trg 5
[5]	EEV3	R/W	0x0	此位使能后 External Event 3 事件会产生 Adc Trg 5
[4]	MPER	R/W	0x0	此位使能后 Master PWM Period 事件会产生 Adc Trg 5
[3]	MCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件会产生 Adc Trg 5
[2]	MCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件会产生 Adc Trg 5
[1]	MCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件会产生 Adc Trg 5
[0]	MCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件会产生 Adc Trg 5

19.5.2.47 HRPWM ADC Trigger 6 Register (HRPWM_ADT6R)

- **Name**：HRPWM ADC Trigger 6 Register
- **Size**：32bits
- **Offset**：0x3C0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	CMPD5	R/W	0x0	此位使能后 PWM5 Compare D 事件会产生 Adc Trg 6
[30]	CMPC5	R/W	0x0	此位使能后 PWM5 Compare C 事件会产生 Adc Trg 6
[29]	PER4	R/W	0x0	此位使能后 PWM4 Period 事件会产生 Adc Trg 6
[28]	CMPD4	R/W	0x0	此位使能后 PWM4 Compare D 事件会产生 Adc Trg 6
[27]	CMPC4	R/W	0x0	此位使能后 PWM4 Compare C 事件会产生 Adc Trg 6
[26]	CMPB4	R/W	0x0	此位使能后 PWM4 Compare B 事件会产生 Adc Trg 6
[25]	RST3PER3	R/W	0x0	此位使能后 PWM3 Reset/Period 事件会产生 Adc Trg 6
[24]	CMPD3	R/W	0x0	此位使能后 PWM3 Compare D 事件会产生 Adc Trg 6
[23]	CMPC3	R/W	0x0	此位使能后 PWM3 Compare C 事件会产生 Adc Trg 6
[22]	CMPB3	R/W	0x0	此位使能后 PWM3 Compare B 事件会产生 Adc Trg 6
[21]	PER2	R/W	0x0	此位使能后 PWM2 Period 事件会产生 Adc Trg 6
[20]	CMPD2	R/W	0x0	此位使能后 PWM2 Compare D 事件会产生 Adc Trg 6
[19]	CMPC2	R/W	0x0	此位使能后 PWM2 Compare C 事件会产生 Adc Trg 6
[18]	CMPB2	R/W	0x0	此位使能后 PWM2 Compare B 事件会产生 Adc Trg 6
[17]	RST1PER1	R/W	0x0	此位使能后 PWM1 Reset/Period 事件会产生 Adc Trg 6
[16]	CMPD1	R/W	0x0	此位使能后 PWM1 Compare D 事件会产生 Adc Trg 6
[15]	CMPC1	R/W	0x0	此位使能后 PWM1 Compare C 事件会产生 Adc Trg 6
[14]	CMPB1	R/W	0x0	此位使能后 PWM1 Compare B 事件会产生 Adc Trg 6
[13]	PER0	R/W	0x0	此位使能后 PWM0 Period 事件会产生 Adc Trg 6
[12]	CMPD0	R/W	0x0	此位使能后 PWM0 Compare D 事件会产生 Adc Trg 6
[11]	CMPC0	R/W	0x0	此位使能后 PWM0 Compare C 事件会产生 Adc Trg 6
[10]	CMPB0	R/W	0x0	此位使能后 PWM0 Compare B 事件会产生 Adc Trg 6
[9]	RST5PER5	R/W	0x0	此位使能后 PWM5 Reset/Period 事件会产生 Adc Trg 6
[8]	CMPB5	R/W	0x0	此位使能后 PWM5 Compare B 事件会产生 Adc Trg 6
[7]	EEV2	R/W	0x0	此位使能后 External Event 2 事件会产生 Adc Trg 6
[6]	EEV1	R/W	0x0	此位使能后 External Event 1 事件会产生 Adc Trg 6
[5]	EEV0	R/W	0x0	此位使能后 External Event 0 事件会产生 Adc Trg 6
[4]	MPER	R/W	0x0	此位使能后 Master PWM Period 事件会产生 Adc Trg 6
[3]	MCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件会产生 Adc Trg 6
[2]	MCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件会产生 Adc Trg 6
[1]	MCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件会产生 Adc Trg 6
[0]	MCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件会产生 Adc Trg 6

19.5.2.48 HRPWM ADC Trigger 7 Register (HRPWM_ADT7R)

- **Name**：HRPWM ADC Trigger 7 Register
- **Size**：32bits
- **Offset**：0x3C4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	CMPD5	R/W	0x0	此位使能后 PWM5 Compare D 事件会产生 Adc Trg 7
[30]	CMPC5	R/W	0x0	此位使能后 PWM5 Compare C 事件会产生 Adc Trg 7
[29]	RST4PER4	R/W	0x0	此位使能后 PWM4 Reset/Period 事件会产生 Adc Trg 7
[28]	CMPD4	R/W	0x0	此位使能后 PWM4 Compare D 事件会产生 Adc Trg 7
[27]	CMPC4	R/W	0x0	此位使能后 PWM4 Compare C 事件会产生 Adc Trg 7
[26]	CMPB4	R/W	0x0	此位使能后 PWM4 Compare B 事件会产生 Adc Trg 7
[25]	PER3	R/W	0x0	此位使能后 PWM3 Period 事件会产生 Adc Trg 7
[24]	CMPD3	R/W	0x0	此位使能后 PWM3 Compare D 事件会产生 Adc Trg 7
[23]	CMPC3	R/W	0x0	此位使能后 PWM3 Compare C 事件会产生 Adc Trg 7
[22]	CMPB3	R/W	0x0	此位使能后 PWM3 Compare B 事件会产生 Adc Trg 7
[21]	RST2PER2	R/W	0x0	此位使能后 PWM2 Reset/Period 事件会产生 Adc Trg 7
[20]	CMPD2	R/W	0x0	此位使能后 PWM2 Compare D 事件会产生 Adc Trg 7
[19]	CMPC2	R/W	0x0	此位使能后 PWM2 Compare C 事件会产生 Adc Trg 7
[18]	CMPB2	R/W	0x0	此位使能后 PWM2 Compare B 事件会产生 Adc Trg 7
[17]	PER1	R/W	0x0	此位使能后 PWM1 Period 事件会产生 Adc Trg 7
[16]	CMPD1	R/W	0x0	此位使能后 PWM1 Compare D 事件会产生 Adc Trg 7
[15]	CMPC1	R/W	0x0	此位使能后 PWM1 Compare C 事件会产生 Adc Trg 7
[14]	CMPB1	R/W	0x0	此位使能后 PWM1 Compare B 事件会产生 Adc Trg 7
[13]	RST0PER0	R/W	0x0	此位使能后 PWM0 Reset/Period 事件会产生 Adc Trg 7
[12]	CMPD0	R/W	0x0	此位使能后 PWM0 Compare D 事件会产生 Adc Trg 7
[11]	CMPC0	R/W	0x0	此位使能后 PWM0 Compare C 事件会产生 Adc Trg 7
[10]	CMPB0	R/W	0x0	此位使能后 PWM0 Compare B 事件会产生 Adc Trg 7
[9]	PER5	R/W	0x0	此位使能后 PWM5 Period 事件会产生 Adc Trg 7
[8]	CMPB5	R/W	0x0	此位使能后 PWM5 Compare B 事件会产生 Adc Trg 7
[7]	EEV5	R/W	0x0	此位使能后 External Event 5 事件会产生 Adc Trg 7
[6]	EEV4	R/W	0x0	此位使能后 External Event 4 事件会产生 Adc Trg 7
[5]	EEV3	R/W	0x0	此位使能后 External Event 3 事件会产生 Adc Trg 7
[4]	MPER	R/W	0x0	此位使能后 Master PWM Period 事件会产生 Adc Trg 7
[3]	MCMPD	R/W	0x0	此位使能后 Master PWM Compare D 事件会产生 Adc Trg 7
[2]	MCMPC	R/W	0x0	此位使能后 Master PWM Compare C 事件会产生 Adc Trg 7
[1]	MCMPB	R/W	0x0	此位使能后 Master PWM Compare B 事件会产生 Adc Trg 7
[0]	MCMPA	R/W	0x0	此位使能后 Master PWM Compare A 事件会产生 Adc Trg 7

19.5.2.49 HRPWM ADC Trigger Post Scaler Register (HRPWM_ADPSR)

- **Name**：HRPWM ADC Trigger Post Scaler Register
- **Size**：32bits
- **Offset**：0x3C8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-28]	PSC7	R/W	0x0	Adc Trg 降采样比例 此位配置 Adc Trg 降采样功能，(PSC+1)个 Adc Trg 7 产生 1 个 Adc Trg 7
[27-24]	PSC6	R/W	0x0	Adc Trg 降采样比例 此位配置 Adc Trg 降采样功能，(PSC+1)个 Adc Trg 6 产生 1 个 Adc Trg 6
[23-20]	PSC5	R/W	0x0	Adc Trg 降采样比例 此位配置 Adc Trg 降采样功能，(PSC+1)个 Adc Trg 5 产生 1 个 Adc Trg 5
[19-16]	PSC4	R/W	0x0	Adc Trg 降采样比例 此位配置 Adc Trg 降采样功能，(PSC+1)个 Adc Trg 4 产生 1 个 Adc Trg 4
[15-12]	PSC3	R/W	0x0	Adc Trg 降采样比例 此位配置 Adc Trg 降采样功能，(PSC+1)个 Adc Trg 3 产生 1 个 Adc Trg 3
[11-8]	PSC2	R/W	0x0	Adc Trg 降采样比例 此位配置 Adc Trg 降采样功能，(PSC+1)个 Adc Trg 2 产生 1 个 Adc Trg 2
[7-4]	PSC1	R/W	0x0	Adc Trg 降采样比例 此位配置 Adc Trg 降采样功能，(PSC+1)个 Adc Trg 1 产生 1 个 Adc Trg 1
[3-0]	PSC0	R/W	0x0	Adc Trg 降采样比例 此位配置 Adc Trg 降采样功能，(PSC+1)个 Adc Trg 0 产生 1 个 Adc Trg 0

19.5.2.50 HRPWM DLL Control Register (HRPWM_DLLCR)

- **Name**：HRPWM DLL Control Register
- **Size**：32bits
- **Offset**：0x3CC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-21]	Reserved	Reserved	Reserved	Reserved
[20-16]	DLLTHRES	R/W	0x0	DLL Clock Delay Threshold. CLKPHASE > DLLTHRES1 : Sample PreDelay CLKPHASE <= DLLTHRES1 : Sample Non-PreDelay
[15-11]	DLLTHRES0	R/W	0x0	DLL Clock Delay Threshold. CLKPHASE <= DLLTHRES0 : Sample hrpwm_clk Pulse CLKPHASE > DLLTHRES0 : Sample hrpwm_dly_clk Pulse
[10-6]	DLLTHRES1	R/W	0x0	DLL Clock Delay Threshold. CLKPHASE >= DLLTHRES1 : Sample hrpwm_clk Pulse CLKPHASE < DLLTHRES1 : Sample hrpwm_dly_clk Pulse
[5-4]	Reserved	Reserved	Reserved	Reserved
[3]	DLLSTART	R/W	0x0	DLL 启动位 DLL 开启至少 20us 后 DLL 启动，等待 1ms 后 DLL 保证稳定。 0：DLL 停止 1：DLL 启动
[2-1]	DLLGCP	R/W	0x0	DLL 电流选择位 00：4uA 01：6uA 10：6uA 11：8uA
[0]	DLLEN	R/W	0x0	DLL 使能位 DLL 开启至少 20us 后 DLL 启动，等待 1ms 后 DLL 保证稳定。 0：DLL 关闭 1：DLL 开启

19.5.2.51 HRPWM Fault Input Register0 (HRPWM_FLTINR0)

- **Name**：HRPWM Fault Input Register0
- **Size**：32bits
- **Offset**：0x3D0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	Reserved	Reserved	Reserved	Reserved
[23-22]	DLLTHRES	R/W	0x0	Fault 5 输入来源 00：HRPWM_FLT5 01：COMP0_OUT 10：HRPWM Event 5 11：Reserved
[21]	DLLTHRES0	R/W	0x0	Fault 5 输入极性 0：Fault 5 输入高有效 1：Fault 5 输入低有效
[20]	DLLTHRES1	R/W	0x0	Fault 5 输入使能 0：Fault 5 关闭 1：Fault 5 开启
[19-18]	Reserved	R/W	0x0	Fault 4 输入来源 00：HRPWM_FLT4 01：COMP0_OUT 10：HRPWM Event 4 11：Reserved
[17]	DLLSTART	R/W	0x0	Fault 4 输入极性 0：Fault 4 输入高有效 1：Fault 4 输入低有效
[16]	DLLGCP	R/W	0x0	Fault 4 输入使能 0：Fault 4 关闭 1：Fault 4 开启
[15-14]	DLLEN	R/W	0x0	Fault 3 输入来源 00：HRPWM_FLT3 01：COMP0_OUT 10：HRPWM Event 3 11：Reserved
[13]	Reserved	R/W	0x0	Fault 3 输入极性 0：Fault 3 输入高有效 1：Fault 3 输入低有效
[12]	DLLTHRES	R/W	0x0	Fault 3 输入使能 0：Fault 3 关闭 1：Fault 3 开启

[11-10]	DLLTHRES0	R/W	0x0	Fault 2 输入来源 00：HRPWM_FLT2 01：COMP0_OUT 10：HRPWM Event 2 11：Reserved
[9]	DLLTHRES1	R/W	0x0	Fault 2 输入极性 0：Fault 2 输入高有效 1：Fault 2 输入低有效
[8]	Reserved	R/W	0x0	Fault 2 输入使能 0：Fault 2 关闭 1：Fault 2 开启
[7-6]	DLLSTART	R/W	0x0	Fault 1 输入来源 00：HRPWM_FLT1 01：COMP0_OUT 10：HRPWM Event 1 11：Reserved
[5]	DLLGCP	R/W	0x0	Fault 1 输入极性 0：Fault 1 输入高有效 1：Fault 1 输入低有效
[4]	DLLEN	R/W	0x0	Fault 1 输入使能 0：Fault 1 关闭 1：Fault 1 开启
[3-2]	Reserved	R/W	0x0	Fault 0 输入来源 00：HRPWM_FLT0 01：COMP0_OUT 10：HRPWM Event 0 11：Reserved
[1]	DLLTHRES	R/W	0x0	Fault 0 输入极性 0：Fault 0 输入高有效 1：Fault 0 输入低有效
[0]	DLLTHRES0	R/W	0x0	Fault 0 输入使能 0：Fault 0 关闭 1：Fault 0 开启

19.5.2.52 HRPWM Fault Input Register1 (HRPWM_FLTINR1)

- **Name**：HRPWM Fault Input Register1
- **Size**：32bits
- **Offset**：0x3D4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-30]	FLTSD	R/W	0x0	Fault 采样时钟分频比例 此位决定 Fault 采样时钟与 HRPWM 模块时钟的分频比例。 00：1 分频（f_sample = f_hrpwm/8） 01：2 分频（f_sample = f_hrpwm/4） 10：4 分频（f_sample = f_hrpwm/2） 11：8 分频（f_sample = f_hrpwm）
[29-24]	Reserved	Reserved	Reserved	Reserved
[23-20]	FLT5F	R/W	0x0	Fault 5 滤波长度 此位定义了用于 Fault 5 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Fault 信号抓一拍即输出
[19-16]	FLT4F	R/W	0x0	Fault 4 滤波长度 此位定义了用于 Fault 4 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Fault 信号抓一拍即输出
[15-12]	FLT3F	R/W	0x0	Fault 3 滤波长度 此位定义了用于 Fault 3 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Fault 信号抓一拍即输出
[11-8]	FLT2F	R/W	0x0	Fault 2 滤波长度 此位定义了用于 Fault 2 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Fault 信号抓一拍即输出
[7-4]	FLT1F	R/W	0x0	Fault 1 滤波长度 此位定义了用于 Fault 1 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Fault 信号抓一拍即输出

[3-0]	FLT0F	R/W	0x0	Fault 0 滤波长度 此位定义了用于 Fault 0 的数字滤波长度，数字滤波由事件计数实现，连续 N 个事件有效则输出有效 假设滤波长度为 N，滤波总时长为 N×1/f_sample，滤波长度为 0 时不滤波，Fault 信号抓一拍即输出

19.5.2.53 HRPWM Fault Input Register2 (HRPWM_FLTINR2)

- **Name**：HRPWM Fault Input Register2
- **Size**：32bits
- **Offset**：0x3D8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	Reserved	Reserved	Reserved	Reserved
[23]	FLT5RSTM	R/W	0x0	Fault 5 复位模式 0：Fault 5 计数在 Reset / Roll-Over 事件处复位 1：Fault 5 计数在 Reset / Roll-Over 事件处复位（仅当上个周期无事件发生时）
[22]	FLT5CRES	R/W	0x0	Fault 5 计数复位 此位将故障计数复位，软件写 1 后等复位完成后硬件自动清 0。 0：Fault 5 计数不复位 1：Fault 5 计数复位
[21]	FLT5BLKS	R/W	0x0	Fault 5 消隐源（窗口大小由 PWM5 决定） 0：Fault 5 固定窗口 1：Fault 5 移动窗口
[20]	FLT5BLKE	R/W	0x0	Fault 5 消隐使能 0：Fault 5 消隐关闭 1：Fault 5 消隐开启
[19]	FLT4RSTM	R/W	0x0	Fault 4 复位模式 0：Fault 4 计数在 Reset / Roll-Over 事件处复位 1：Fault 4 计数在 Reset / Roll-Over 事件处复位（仅当上个周期无事件发生时）
[18]	FLT4CRES	R/W	0x0	Fault 4 计数复位 此位将故障计数复位，软件写 1 后等复位完成后硬件自动清 0。 0：Fault 4 计数不复位 1：Fault 4 计数复位
[17]	FLT4BLKS	R/W	0x0	Fault 4 消隐源（窗口大小由 PWM4 决定） 0：Fault 4 固定窗口 1：Fault 4 移动窗口
[16]	FLT4BLKE	R/W	0x0	Fault 4 消隐使能 0：Fault 4 消隐关闭 1：Fault 4 消隐开启
[15]	FLT3RSTM	R/W	0x0	Fault 3 复位模式 0：Fault 3 计数在 Reset / Roll-Over 事件处复位 1：Fault 3 计数在 Reset / Roll-Over 事件处复位（仅当上个周期无事件发生时）

[14]	FLT3CRES	R/W	0x0	Fault 3 计数复位 此位将故障计数复位，软件写 1 后等复位完成后硬件自动清 0。 0：Fault 3 计数不复位 1：Fault 3 计数复位
[13]	FLT3BLKS	R/W	0x0	Fault 3 消隐源（窗口大小由 PWM3 决定） 0：Fault 3 固定窗口 1：Fault 3 移动窗口
[12]	FLT3BLKE	R/W	0x0	Fault 3 消隐使能 0：Fault 3 消隐关闭 1：Fault 3 消隐开启
[11]	FLT2RSTM	R/W	0x0	Fault 2 复位模式 0：Fault 2 计数在 Reset / Roll-Over 事件处复位 1：Fault 2 计数在 Reset / Roll-Over 事件处复位（仅当上个周期无事件发生时）
[10]	FLT2CRES	R/W	0x0	Fault 2 计数复位 此位将故障计数复位，软件写 1 后等复位完成后硬件自动清 0。 0：Fault 2 计数不复位 1：Fault 2 计数复位
[9]	FLT2BLKS	R/W	0x0	Fault 2 消隐源（窗口大小由 PWM2 决定） 0：Fault 2 固定窗口 1：Fault 2 移动窗口
[8]	FLT2BLKE	R/W	0x0	Fault 2 消隐使能 0：Fault 2 消隐关闭 1：Fault 2 消隐开启
[7]	FLT1RSTM	R/W	0x0	Fault 1 复位模式 0：Fault 1 计数在 Reset / Roll-Over 事件处复位 1：Fault 1 计数在 Reset / Roll-Over 事件处复位（仅当上个周期无事件发生时）
[6]	FLT1CRES	R/W	0x0	Fault 1 计数复位 此位将故障计数复位，软件写 1 后等复位完成后硬件自动清 0。 0：Fault 1 计数不复位 1：Fault 1 计数复位
[5]	FLT1BLKS	R/W	0x0	Fault 1 消隐源（窗口大小由 PWM2 决定） 0：Fault 1 固定窗口 1：Fault 1 移动窗口
[4]	FLT1BLKE	R/W	0x0	Fault 1 消隐使能 0：Fault 1 消隐关闭 1：Fault 1 消隐开启
[3]	FLT0RSTM	R/W	0x0	Fault 0 复位模式 0：Fault 0 计数在 Reset / Roll-Over 事件处复位 1：Fault 0 计数在 Reset / Roll-Over 事件处复位（仅当上个周期无事件发生时）

[2]	FLT0CRES	R/W	0x0	Fault 0 计数复位 此位将故障计数复位，软件写 1 后等复位完成后硬件自动清 0。 0：Fault 0 计数不复位 1：Fault 0 计数复位
[1]	FLT0BLKS	R/W	0x0	Fault 0 消隐源（窗口大小由 PWM2 决定） 0：Fault 0 固定窗口 1：Fault 0 移动窗口
[0]	FLT0BLKE	R/W	0x0	Fault 0 消隐使能 0：Fault 0 消隐关闭 1：Fault 0 消隐开启

19.5.2.54 HRPWM Fault Input Register3 (HRPWM_FLTINR3)

- **Name**：HRPWM Fault Input Register3
- **Size**：32bits
- **Offset**：0x3DC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	Reserved	Reserved	Reserved	Reserved
[23-20]	FLT5CNT	R/W	0x0	Fault 5 计数阈值 当故障计数总数等于（FLT5CNT＋1）时认为故障有效
[19-16]	FLT4CNT	R/W	0x0	Fault 4 计数阈值 当故障计数总数等于（FLT5CNT＋1）时认为故障有效
[15-12]	FLT3CNT	R/W	0x0	Fault 3 计数阈值 当故障计数总数等于（FLT5CNT＋1）时认为故障有效
[11-8]	FLT2CNT	R/W	0x0	Fault 2 计数阈值 当故障计数总数等于（FLT5CNT＋1）时认为故障有效
[7-4]	FLT1CNT	R/W	0x0	Fault 1 计数阈值 当故障计数总数等于（FLT5CNT＋1）时认为故障有效
[3-0]	FLT0CHT	R/W	0x0	Fault 0 计数阈值 当故障计数总数等于（FLT5CNT＋1）时认为故障有效

20 独立看门狗（IWDG）

20.1 简介

独立看门狗具有安全性高、使用灵活的特点。可用于检测并解决软件错误导致的故障，在定时器到达指定的超时值时触发系统复位。

独立看门狗 (IWDG) 由其专用低速时钟 (LSI) 驱动，因此即便在主时钟发生故障时仍然保持工作状态。

IWDG 最适合应用于那些需要看门狗作为一个在主程序之外，能够完全独立工作，并且对时间精度要求较低的场合。

20.2 结构框图

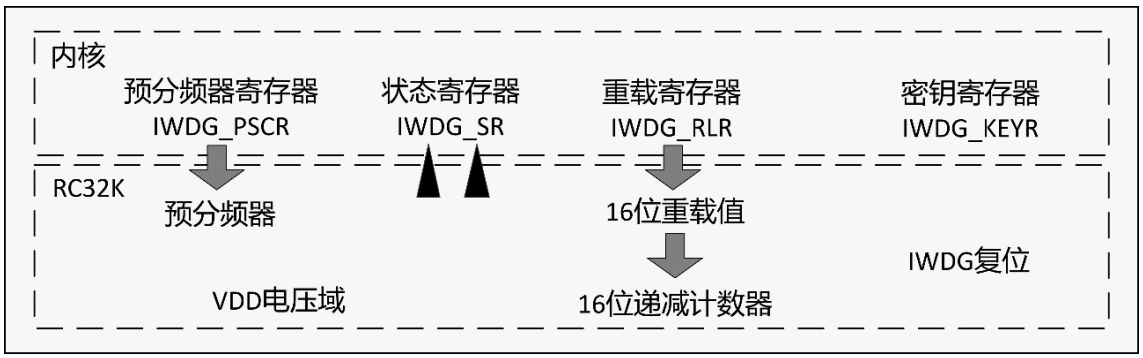


图 20-1 IWDG 架构示意图

20.3 主要特性

IWDG 主要具有以下特性：

- 自由运行递减计数器
- 16 位重载计数值寄存器，计数时钟支持 4 到 512 分频
- 时钟由独立 RC 振荡器(LSI)提供（可在待机模式下独立运行）
- 当递减计数器值达到 0x0 时产生复位（如果看门狗已激活）

20.4 功能描述

20.4.1 IWDG 键值功能

将键值 0xCCCC 写到键寄存器 (IWDG_KEYR[15:0])中，将启动独立看门狗。看门狗内部计数器开始从复位值 0xFFFF 递减计数，当内部计数器计数到终值 (0x0000) 时将会产生一个系统复位或触发中断。

任何时候将键值 0xAAAA 写到键寄存器 (IWDG_KEYR[15:0])中，重载寄存器（IWDG_RLR）的值就会被重载到计数器，从而避免产生看门狗复位。

将键值 0xDDDD 写到键寄存器(IWDG_KEYR[15:0])中，将停止独立看门狗。

将键值 0x5003 写到键寄存器(IWDG_KEYR[15:0])中，将解除 IWDG 的寄存器访问保护，详见“20.4.2IWDG 寄存器访问保护”说明。

注意：重载操作(将键值 0xAAAA 写到 IWDG_KEYR[15:0])必须在启动 IWDG 后，且 IWDG_SR 寄存器中的 RLVUPD 和 PSCUPD 位都为 0 时才可以进行。

20.4.2 IWDG 寄存器访问保护

IWDG 预分频寄存器(IWDG_PSCR)、重加载寄存器(IWDG_RLR)及控制寄存器(IWDG_CR)具有写访问保护。

若要修改上述寄存器，首先需要将 0x5003 写入键寄存器 (IWDG_KEYR[15:0])解除上述寄存器的访问保护；写入其他值，则使寄存器访问保护重新生效，下次修改则需要重新解除访问保护。这意味着重装载操作（写 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。

20.4.3 IWDG 调试模式

当微控制器进入调试模式时（Cortex™-M3 内核停止），IWDG 计数器会根据 DBG 模块中的 IWDG_DBG_EN 配置位选择继续正常工作或者停止工作。详见 DBG 调试模块相关章节。

20.4.4 IWDG 中断模式和复位模式

IWDG 支持两种工作模式：中断模式和复位模式。通过 IWDG_CR 寄存器的 MODE 位选择 IWDG 的工作模式。

中断模式下，IWDG 在未重载导致超时后，IWDG_SR 寄存器中的 TOIF 将会置位，并触发中断(如果使能 IWDG 中断)，且不会导致系统复位。

复位模式下，IWDG 在未重载导致超时后，触发一个系统复位信号，如果 DBG 中的

IWDG_RST_EN 配置使能 IWDG 复位功能，则引发系统复位(如果 IWDG_RST_EN 未使能，则同样不会导致系统复位，直到该位使能)。

20.4.5 IWDG 计时

表 20-1 32 kHz 频率条件下 IWDG 超时周期的最小值/最大值

预分频器	IWDG_PSCR[2:0]	最短超时（ms） IWDG_RLR = 0x0	最长超时（ms） IWDG_RLR = 0xFFFF
/4	0	0.125	8192
/8	1	0.25	16384
/16	2	0.5	32768
/32	3	1	65536
/64	4	2	131072
/128	5	4	262144
/256	6	8	524288
/512	7	16	1048576

注意：

- *这些时间均针对 32kHz 时钟给出。实际上芯片内部的 RC 频率会在 30kHz 到 60kHz 之间变化。此外，即使 RC 振荡器的频率是精确的，确切的时序仍然依赖于 APB 接口时钟与 RC 振荡器时钟之间的相位差，因此总会有一个完整的 RC 周期是不确定的；*
- *配置 IWDG_RLR 及 IWDG_PSCR 都需要从 APB 时钟域同步到 LSI 时钟域下，会一定程度影响 IWDG 的精度。*

20.4.6 IWDG 中断和状态

IWDG_SR 寄存器中定义了下述几个中断标志位和状态标志位。

标志名称	中断使能	标志位	备注
超时中断	TOIE	TOIF	IWDG 处于中断模式下时，未重载导致看门狗超时。
重载值更新状态	-	RLVUPD	置位时表示重载值正在更新，完成后硬件自动清 0。
分频值更新状态	-	PSCUPD	置位时表示分频值正在更新，完成后硬件自动清 0。

注意：IWDG 配置重载值寄存器(IWDG_RLR)和预分频寄存器(IWDG_PSCR)后，应当检查 RLVUPD 和 PSCUPD 标志，只有更新动作完成后，才可对看门狗进行重载操作。

20.5 寄存器描述

20.5.1 寄存器列表

Name	Offset	Width	Description
IWDG_KEYR	0x00	32bits	IWDG 键寄存器
IWDG_CR	0x04	32bits	IWDG 控制寄存器
IWDG_RLR	0x08	32bits	IWDG 重载寄存器
IWDG_PSCR	0x0C	32bits	IWDG 预分频寄存器
IWDG_SR	0x10	32bits	IWDG 状态寄存器

20.5.2 寄存器详细描述

20.5.2.1 IWDG 键寄存器 (IWDG_KEYR)

- **Name**：IWDG Key Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	KEY	R/W	0x0	IWDG 键值（IWDG key value） 详细请参看“20.4.1IWDG 键值功能”。

20.5.2.2 IWDG 控制寄存器 (IWDG_CR)

- **Name**：IWDG Control Register
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1]	TOIE	R/W	0x0	IWDG 超时中断使能位（IWDG Timeout Int Enable） 需要由软件解锁，写访问前对 0x00 寄存器写入值 0x5003； 0：关闭 1：使能 IWDG 超时中断 注意：仅中断模式下有效，详请参考“20.4.4 IWDG 中断模式和复位模式”。
[0]	MODE	R/W	0x0	IWDG 模式配置位（IWDG Mode config） 0：复位模式 1：中断模式 注意：详请参考“20.4.4 IWDG 中断模式和复位模式”章节。

20.5.2.3 IWDG 重载寄存器 (IWDG_RLR)

- **Name**：IWDG Reload Register
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	RLV	R/W	0xFFFF	IWDG 重载值（IWDG Reload Value） 对 IWDG 进行重载操作(IWDG_KEYR 寄存器写 0xAAAA)时，将 RLV 值重载至内部计数器重新开始计数。 注意： 1. 若要对 IWDG 进行重载，IWDG_SR 寄存器的 RLVUPD 和 PSCUPD 位必须都为 0。 2. 若要对 IWDG 的重载值进行修改， RLVUPD 位同样必须为 0。

20.5.2.4 IWDG 预分频寄存器 (IWDG_PSCR)

- **Name**：IWDG Prescaler Register
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2-0]	PSC	R/W	0x0	IWDG 预分频器（IWDG Prescaler Divider） 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 111：512 分频 注意： 1. 若要对 IWDG 进行重载，IWDG_SR 寄存器的 RLVUPD 和 PSCUPD 位必须都为 0。 2. 若要对 IWDG 的分频进行修改，PSCUPD 位同样必须为 0。

20.5.2.5 IWDG 状态寄存器 (IWDG_SR)

- **Name**：IWDG Status Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	Reserved	Reserved	Reserved
[2]	TOIF	R/W	0x0	IWDG 超时中断标志位（IWDG Timeout Interrupt Flag） 0：No Pending 1：Irq Pending 软件写 1 清 0，写 0 无效。
[1]	RLVUPD	R	0x0	IWDG 计数器重载值更新（IWDG Reload Value Update） 通过硬件将该位置 1 以指示重载值正在更新，更新完成后会通过硬件将该位复位。 注意：重载值只有在该位为 0 时才可更新。
[0]	PSCUPD	R	0x0	IWDG 预分频器值更新（IWDG Prescaler Update） 通过硬件将该位置 1 以指示预分频器值正在更新，更新完成后会通过硬件将该位复位。 注意：预分频器值只有在该位为 0 时才可更新。

21 窗口看门狗（WWDG）

21.1 简介

窗口看门狗 (WWDG) 时钟由 APB1 时钟经预分频后提供，窗口看门狗通常被用来监测，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。

由外部干扰或不可预见的逻辑条件造成的应用程序背离正常的运行序列而产生的软件故障。除非递减计数器的值在 T6 位变成 0 前被刷新，看门狗电路在达到预置的时间周期时，会产生一个系统复位。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递减计数器值，也会产生复位。这意味着必须在限定的时间窗口内刷新计数器。

21.2 结构框图

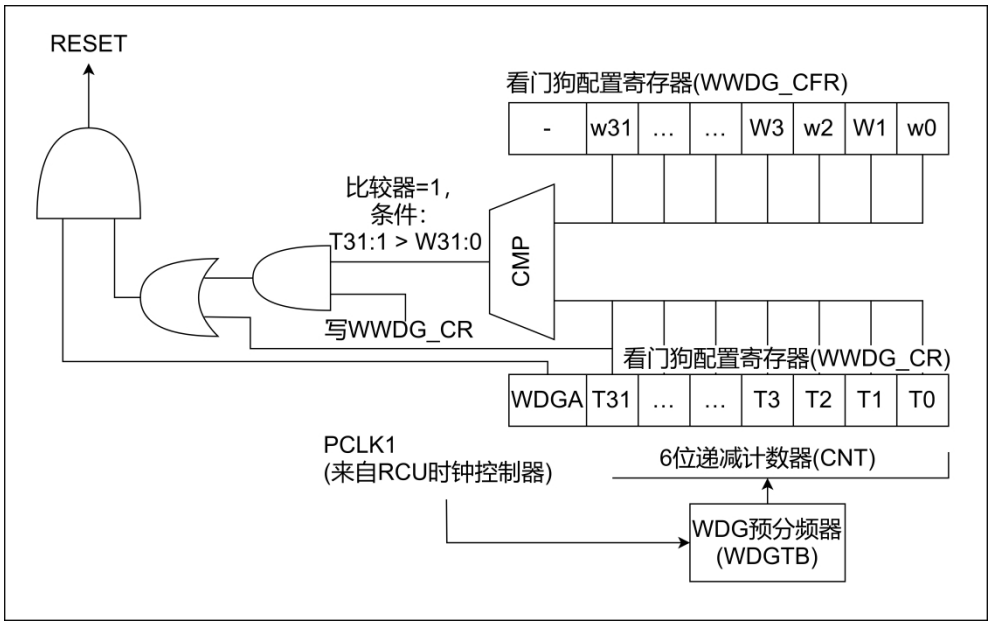


图 21-1 WWDG 架构框图

21.3 主要特性

WWDG 主要具有以下特性：

- 自由运行递减计数器
- 复位条件(DBG 模块的 WWDG_RST_EN 需置 1，允许 WWDG 系统复位，详见 DBG 模块介绍)
  - 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）
- 提前唤醒中断(EWI)：当递减计数器减至 0x40 时触发（如果已使能且看门狗已激活），可用于在中断内重载计数器以避免 WWDG 复位

21.4 功能描述

如果激活看门狗(WWDG_CR 寄存器的 WEN 位置 1)，当 16 位递减计数器从 0x40 滚动到 0x3F 时会引发复位信号；如果软件在计数器值(IWDG_CVR)大于窗口寄存器(IWDG_WVR)中所存储的值时重载计数器，也会产生复位信号。

如果 DBG 模块内的 WWDG_RST_EN 置 1，则 WWDG 的复位信号最终导致系统复位。

应用程序在正常运行过程中必须定期地写 WWDG_CVR 寄存器以防止发生复位。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_CVR 寄存器中的值须介于 0xFFFF 和 0x40 之间(如果设置为 0x40 将无法触发提前唤醒中断 Early Wakeup Interrupt)，用户应当根据实际应用需求合理配置。

21.4.1 使能看门狗

在系统复位后，看门狗处于关闭状态。可通过设置窗口看门狗控制寄存器(WWDG_CR)中的 WEN 位来使能看门狗，也可以按需随时关闭窗口看门狗功能。

21.4.2 控制递减计数器

在使能窗口看门狗之后递减计数器(WWDG_CVR[15:0])就开始运行，每个计数周期执行一次递减操作。使能看门狗前，用户需要根据自己的应用需求确保以下限制：

- 递减计数器的值不应当配置 0x40 以下，否则启动窗口狗后将立即引发系统复位信号。
- 如果需要启用提前唤醒中断(EWI)功能，递减计数器的值不应当配置在 0x41 以下(仅在递减计数器从 0x41 递减至 0x40 时触发提前唤醒中断)

在 WWDG 开始工作后，为了防止系统复位，当递减计数器的值低于窗口寄存器(IWDG_WVR[15:0])的值，且大于 0x3F 时，必须对递减计数器进行重载（向 WWDG_CVR[15:0]写入新值）。重载后递减计数器从新值重新开始按计数周期递减。

21.4.3 看门狗中断高级特性

如果在产生实际复位之前必须执行特定的安全操作或数据记录，则可使用提前唤醒中断(EWI)。通过设置控制寄存器(WWDG_CR)中 EWIE 位使能该中断。当递减计数器的值从 0x41 递减至 0x40 时，将生成中断。在 WWDG 递减计数器继续递减到 0x3F 之前，可以使用相应的中断服务程序来触发特定操作（例如通信或数据记录）或重载递减计数器避免复位。

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可用来重载 WWDG 递减计数器以避免 WWDG 复位，然后再触发所需操作。

通过对中断状态寄存器(WWDG_ISR)写 1 对 EWIF 清 0，即清除提前中断。

注意：在提前唤醒中断的处理中，如果处理时间超过一个计数周期未重载递减计数器，计数

器从 0x40 递减至 0x3F，WWDG 将发出复位信号，引发系统复位(如果使能复位)。因此所有操作必须在这一个计数周期内完成，否则应当先关闭WWDG 或重载递减计数器再执行其他操作。

21.4.4 复位使能和调试模式

WWDG 的复位功能，还需要在 DBG 模块中将 WWDG_RST_EN 位置 1，使能 WWDG 的系统复位功能。否则 WWDG 发出的复位信号并不会引发实际的系统复位，直到 DBG 模块中的 WWDG_RST_EN 位被置 1。

还可以通过将 DBG 模块中的 WWDG_DBG_EN 位置 1，开启 WWDG 的调试模式。开启后，当 MCU 进入调试模式，并且处于 halt 状态（Cortex™-M3 内核停止），WWDG 的递减定时器也会停止工作，直到退出调试模式或调试处于非 halt 状态。

21.4.5 看门狗设置

图 21-2 为 WWDG 计数时序图。

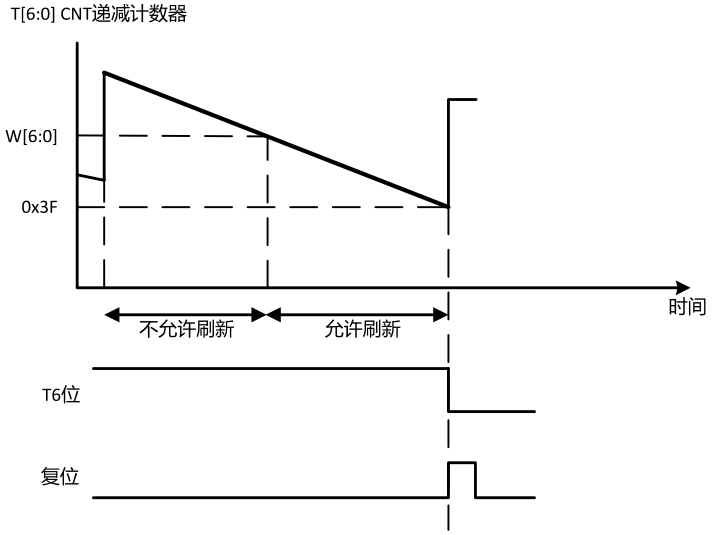


图 21-2 WWDG 计数时序图

超时值的计算公式如下：

$$t_{WWDG} = t_{APB1} \times (PSC + 1) \times (CV + 1) \quad ms$$

$$t_{WWDG} = t_{PCLK} \times (2^{SCALE}) \times (T[15:0] + 1)$$

其中：

PSC ：WWDG_PSCR 窗口看门狗分频寄存器

CV ：WWDG_CVR 窗口看门狗递减计数器

t_{WWDG} ：WWDG 超时时间

t_{APB1} ：APB1 时钟周期，以 ms 为测量单位

例如，假设 APB1 频率为 50MHz(t_{APB1}=1/50000 ms)，PSC 值为 99，CV 值为 999：

$$t_{WWDG} = (1/50000) \times (99 + 1) \times (999 + 1)$$

$$= 2 \text{ ms}$$

21.5 寄存器描述

21.5.1 寄存器列表

Name	Offset	Width	Description
WWDG_CR	0x00	32bits	WWDG 控制寄存器
WWDG_WVR	0x04	32bits	WWDG 窗口值寄存器
WWDG_CVR	0x08	32bits	WWDG 计数器值寄存器
WWDG_PSCR	0x0C	32bits	WWDG 预分频寄存器
WWDG_ISR	0x10	32bits	WWDG 中断状态寄存器

21.5.2 寄存器详细描述

21.5.2.1 WWDG 控制寄存器 (WWDG_CR)

- **Name**：WWDG Control Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-2]	Reserved	Reserved	Reserved	Reserved
[1]	EWIE	R/W	0x0	WWDG 提前唤醒中断使能位（WWDG Early Wakeup Interrupt Enable） 使能提前唤醒中断，在计数减至 0x40 时产生中断，用于提醒即将系统复位，需及时喂狗或执行复位前的必要操作。 0：关闭 1：使能
[0]	WEN	R/W	0x0	WWDG 使能位（WWDG Enable） 0：关闭看门狗 1：使能看门狗

21.5.2.2 WWDG 窗口值寄存器 (WWDG_WVR)

- **Name：** WWDG Window Value Register
- **Size：** 32bits
- **Offset：** 0x04
- **Default：** 0xFFFF

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	WV	R/W	0xFFFF	WWDG 窗口比较值（WWDG Window Value） 注意：用于与递减计数器进行比较的窗口值，只有当计数器值低于窗口寄存器值时，才能重载计数器值，否则也会产生复位操作。

21.5.2.3 WWDG 计数器值寄存器 (WWDG_CVR)

- **Name：** WWDG Counter Value Register
- **Size：** 32bits
- **Offset：** 0x08
- **Default：** 0xFFFF

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	CV	R/W	0xFFFF	WWDG 递减计数值（WWDG Counter Value） 注意： 1. 每个计数周期递减一次，当它从 0x41 递减到 0x40 时引发提前唤醒中断；从 0x40 递减到 0x3F 时会产生复位。 2. 正常运行过程中必须定期地写该寄存器以防止 MCU 发生复位，只有当计数器值低于窗口比较值时，才能执行此操作，否则也会产生复位操作。

21.5.2.4 WWDG 预分频寄存器 (WWDG_PSCR)

- **Name**：WWDG Prescaler Register
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	Reserved	Reserved	Reserved	Reserved
[15-0]	PSC	R/W	0x0	WWDG 预分频器（WWDG Prescaler divider） 用于对递减计数器时钟源(APB1CLK)分频。 分频关系 = PSC + 1 分频 0：不分频 1：2 分频 2：3 分频 3：4 分频 …… 65535： 65536 分频

21.5.2.5 WWDG 中断状态寄存器 (WWDG_ISR)

- **Name**：WWDG Interrupt Status Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	EWIF	R/W1C	0x0	WWDG 提前唤醒中断标志位（WWDG Early Wakeup Interrupt Flag） 0：No Pending 1：Pending

22 内部集成接口（I2C）

22.1 简介

I2C 总线是两线串行接口，由串行数据线（SDA）和串行时钟（SCL）组成。在芯片中 I2C 可作为主机也可以作为从机，数据速率最大可达到 3.4Mb/s。同时，TAE32F5300 的 I2C 还支持 SMBus 协议。

22.2 结构框图

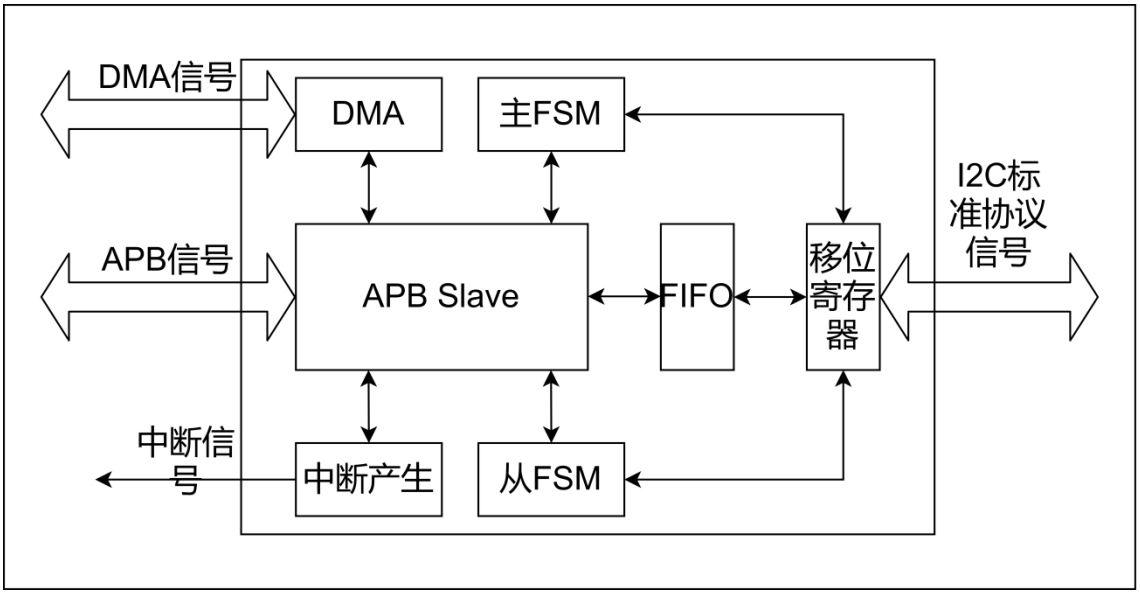


图 22-1 I2C 顶层结构框图

22.3 主要特性

- 两线式 I2C 串行接口–由串行数据线（SDA）和串行时钟（SCL）组成
- 三种速度
  - 标准模式（0 至 100 Kb/s）
  - 快速模式（≤400 Kb/s）或快速模式（≤1000Kb/s）
- 主或从 I2C 操作
- 7 位或 10 位寻址
- 7 位或 10 位组合格式传输
- 发送和接收缓冲区
- 中断或轮询模式操作
- 可编程的 SDA 保持时间
- SMBus / PMBus 支持

22.4 功能描述

22.4.1 I2C 协议

22.4.1.1 I2C 电气结构

SDA 和 SCL 都是双向线路，通过连接到电流源或上拉电阻，产生正电源电压电阻（见图 22-2）。当 I2C 总线处于空闲状态时，SCL 和 SDA 两条线均为高。连接到总线的设备的输出级必须具有漏极开路或集电极开路才能执行线与功能。

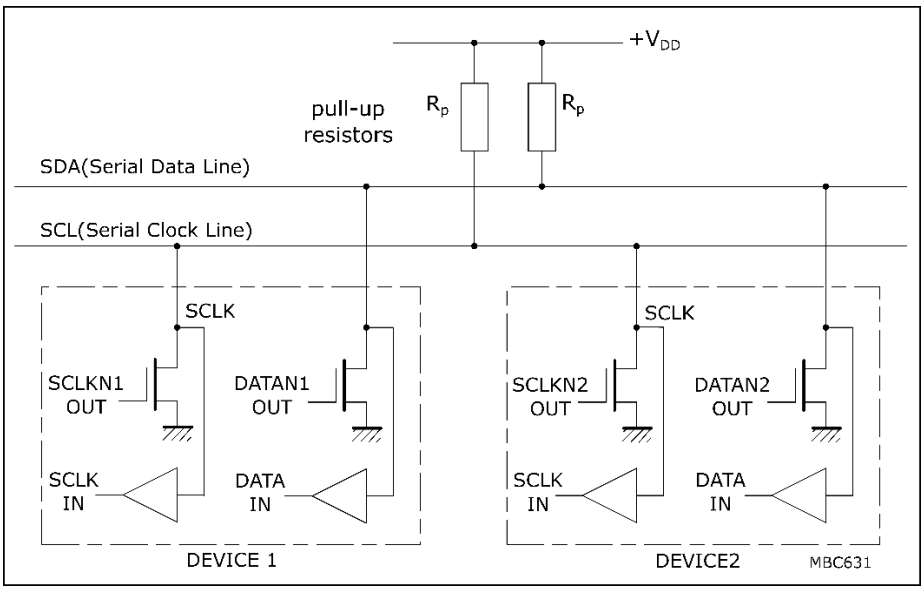


图 22-2　I2C 总线电气结构图

22.4.1.2 I2C 基本结构

I2C 基本结构由下：

- Transmitter：将数据发送到总线的设备。Transmitter 可以是发送数据到总线的主机设备（Master-Transmitter），也可以是响应主机的要求将数据发送到总线的从机设备（Slave-Transmitter）。
- Receiver：从总线接收数据的设备。Receiver 可以是根据自己的请求接收数据的设备（Master-Receiver），也可以是响应于来自主设备的请求的设备（slave-Receiver）。
- Master：初始化传输（START 命令），生成时钟（SCL）信号并终止传输（STOP 命令）的组件。主机可以是 Transmitter 或 Receiver。
- Slave：主站寻址的设备。从机可以是 Transmitter 或 Receiver。
- SDA：数据信号线（串行数据）。
- SCL：时钟信号线（串行时钟）。
- START（RESTART）：数据传输以 START 或 RESTART 条件开始。 SDA 数据线的电平从高电平变为低电平，而 SCL 时钟线保持高电平。当发生这种情况时，总线变得繁忙。
- STOP：数据传输因 STOP 条件而终止。 当 SDA 数据线上的电平从低电平变为高电平，而 SCL 时钟线保持高电平时，就会发生这种情况。数据传输终止后，总线将再次空闲。

上述结构关系如图 22-3 所示。

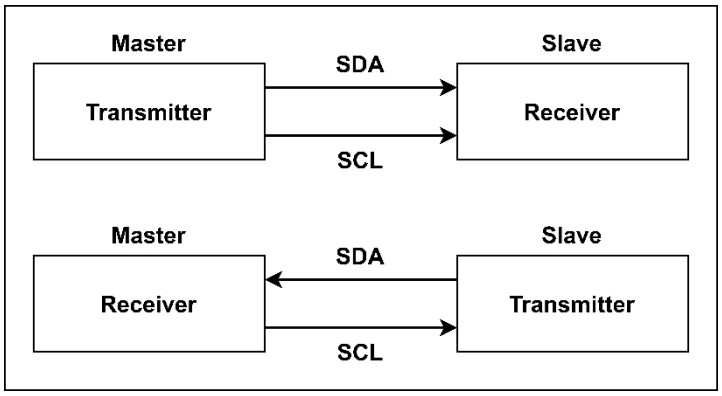


图 22-3 主/从和发送器/接收器的关系

Master 负责产生时钟并控制数据传输。Slave 负责向 Master 发送数据或从 Master 接收数据。数据确认由接收数据的设备发送，该设备可以是 Master，也可以是 Slave。

每个 Slave 都有一个由系统设计人员确定的唯一地址。当 Master 要与从机通信时，Master 发送一个 START / RESTART 命令，然后是 Slave 的地址和一个控制位（R / W）。然后，Slave 在地址之后发送一个确认（ACK）脉冲。

如果 Master（Transmitter）正在写入 Slave（Receiver），则 Receiver 将获得一个字节的数据。此事务将继续进行，直到 Master 以 STOP 条件终止传输为止。如果 Master 正在从 Slave 读取数据，则 Slave（Transmitter）向 Master（Receiver）发送一个字节的数据，然后 Master 用 ACK 脉冲确认该事务。该事务持续进行，直到 Master 在接收到最后一个字节后不确认（NACK）事务来终止传输，然后 Master 发出 STOP 条件或在发出 RESTART 条件后寻址另一个 Slave。上述行为如图 22-4 所示。

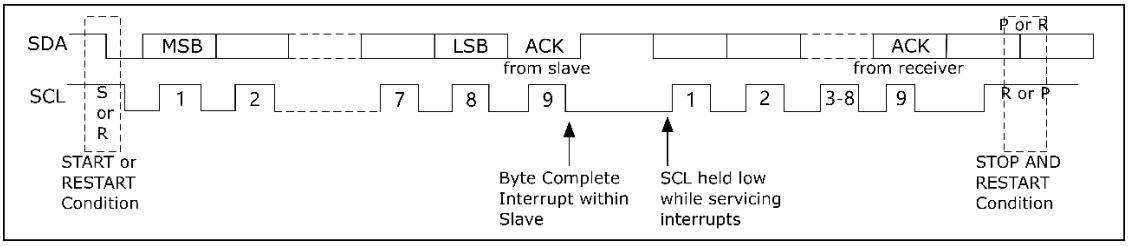


图 22-4　I2C 总线的数据传输

22.4.1.3　STOP 和 START

总线空闲时，SCL 和 SDA 信号均通过总线上的外部上拉电阻上拉。 当 Master 希望在总线上开始传输时，Master 发出 START（SCL 为高时 SDA 信号由高到低的跳变）；当 Master 希望终止发送时，主机发出 STOP（SCL 为高时 SDA 线从低到高的跳变）。当数据在总线上传输时，当 SCL 为高时，SDA 线必须稳定。图 22-5 显示了 START 和 STOP 条件的时序。

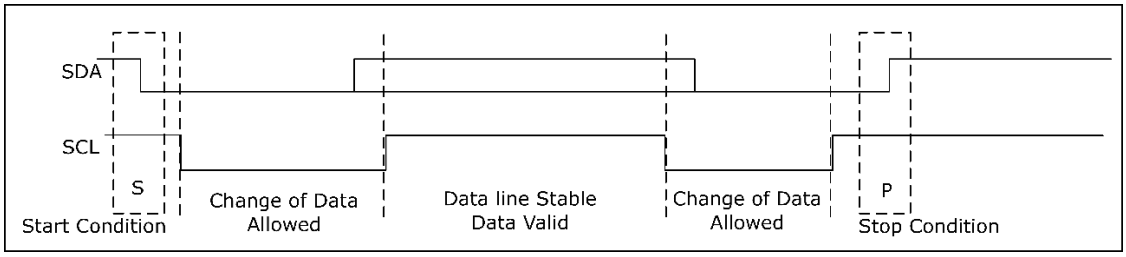


图 22-5　START 和 STOP 的时序

22.4.1.4　寻址从协议

地址格式有两种：7 位地址格式和 10 位地址格式。

7 位地址格式

在 7 位地址格式中，第一个字节的前七个位（位 7：1）设置从机地址，而 LSB 位（位 0）为 R / W 位，如图 22-6 所示。 当位 0（R / W）设置为 0 时，Master 写入 Slave。 当位 0（R / W）设置为 1 时，Master 将从 Slave 读取数据。

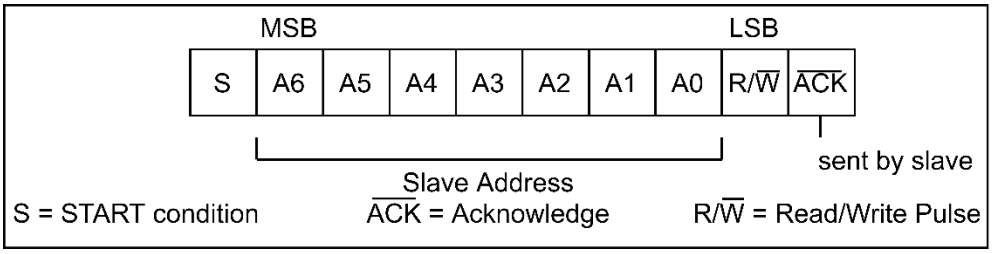


图 22-6　7 位地址格式

10 位地址格式

在 10 位寻址期间，将传输两个字节以设置 10 位地址。第一个字节的传输包含以下位定义。 前五个位（位 7：3）通知 Slave 这是 10 位传输，其后是后两个位（位 2：1），它们是 Slave 地

址位 9：8，LSB 位是 R / W 位。传输的第二个字节设置 Slave 地址的位 7：0。10 位地址格式如图 22-7 所示。

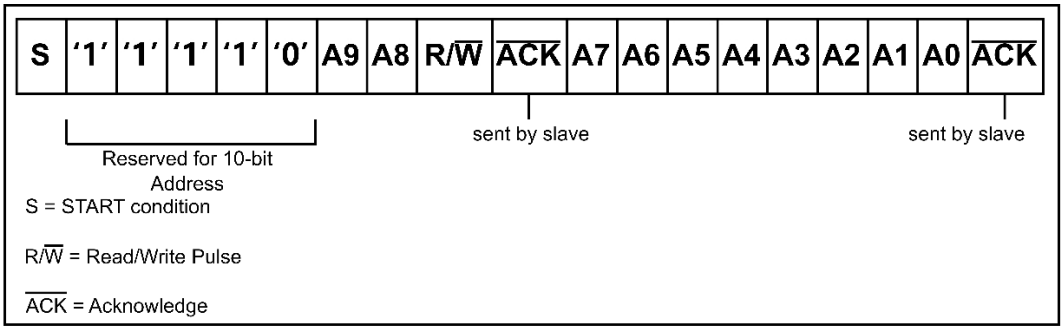


图 22-7　10 位地址格式

表 22-1 定义了特殊用途和保留的第一个字节地址。

表 22-1　I2C / SMBus 第一个字节中的位定义

Slave Address	R/W Bit	Description
0000 000	0	通用呼叫地址。 i2c 将数据放置在接收缓冲区中，并发出常规调用中断。
0000 000	1	START 字节。
0000 001	X	CBUS 地址。 i2c 忽略这些访问。
0000 010	X	Reserved.
0000 011	X	Reserved
0000 1XX	X	高速主模式
1111 1XX	X	Reserved
1111 0XX	X	10 位从机寻址。
0001 000	X	SMbus Host
0001 100	X	SMBus 警报响应地址
1100 001	X	SMBus 设备默认地址

I2C 不会限制使用这些保留的地址。但是，如果使用这些保留的地址，则可能会与其他 I2C 组件不兼容。

22.4.1.5　收发协议

Master 可以作为 Master-Transmitter 或 Master-Receiver。 Slave 响应来自 Master 的请求，以分别向总线发送数据或从总线接收数据，分别充当 Slave-Transmitter 或 Slave-Receiver。

Master-Transmitter 和 Slave-Receiver

所有数据均以字节格式传输，每次数据传输的字节数没有限制。 在 Master 发送地址和 R/W 位或 Master 向 Slave 发送数据字节之后，Slave-Receiver 必须以确认信号（ACK）进行响应。当 Slave-Receiver 未响应 ACK 脉冲时，Master 通过发出 STOP 条件中止传输。如图 22-8 所示，Slave 必须将 SDA 线保持高电平，以便 Master 可以中止传输。如果 Master-Transmitter 正在发送数据， Slave-Receiver 在接收到每个字节的数据后用 ACK 响应 Master-Transmitter。

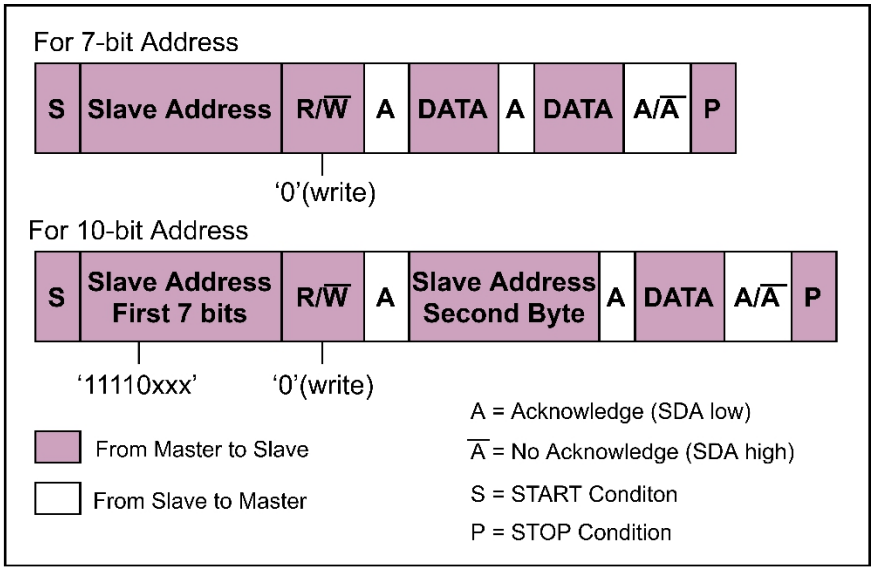


图 22-8 Master-Transmitter 协议

Slave-Transmitter 和 Master-Receiver

如图 22-9 所示，如果 Master 正在接收数据，则在接收到一个字节的数据（最后一个字节除外）之后，Master 以一个 ACK 脉冲响应 Slave-Transmitter，通知 Slave-Transmitter 这是最后一个字节。Slave-Transmitter 检测到无应答（NACK）后放弃 SDA 线， Master 可以发出 STOP。

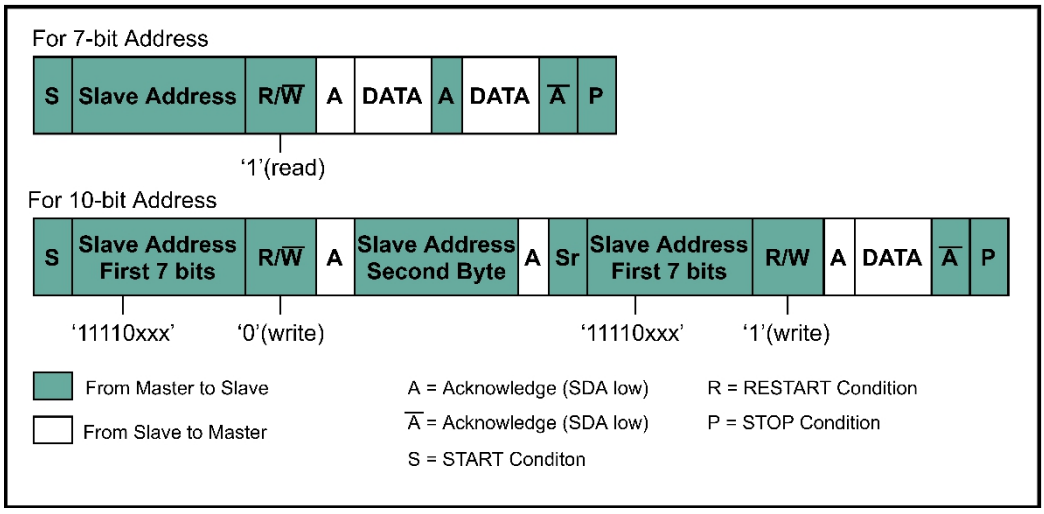


图 22-9 Master-Receiver 协议

当 Master 不希望以 STOP 放弃总线时，Master 可以发出 RESTART。RESTART 波形与 START 相同

22.4.1.6 START BYTE 传输协议

START BYTE 传输协议是为没有板载专用 I2C 硬件模块的系统设置的。当 I2C 作为 Slave 寻址时，它总是以支持的最高速度对 I2C 总线进行采样，因此它永远不需要 START BYTE 传输。但是，当 I2C 是 Master 时，它支持在每次传输开始时生成 START BYTE 传输，以防从机需要它。

该协议由七个零组成，后跟一个 1，如图 22-10 所示。 这允许轮询总线的处理器对地址相位进行欠采样，直到检测到 0。一旦检测到 0，它将从欠采样率切换到正确的主机速率。

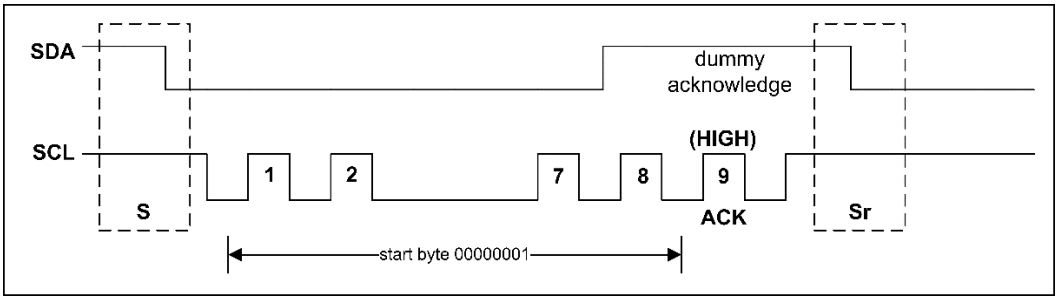


图 22-10 START BYTE 传输

START BYTE 过程如下：

1. 主机产生一个启动条件；
2. 主机发送起始字节（0000 0001）；
3. 主机发送 ACK 时钟脉冲；
4. 没有从机将 ACK 信号设置为 0；
5. 主机产生一个 RESTART（R）条件。

硬件接收器不响应 START BYTE，因为它是保留地址，并在产生 RESTART 条件后复位。

22.4.2 Tx FIFO 管理以及 START, STOP 和 RESTART 生成

当 Tx FIFO 变空时，I2C 不会产生 STOP。此时，I2C 将 SCL 线保持为低电平，从而使总线停顿，直到 Tx FIFO 中有新的数据可用为止。仅当用户通过设置写入 I2C_DATA_CMD 寄存器的命令的第 9 位（STOP）明确要求时，才产生 STOP。

图 22-11 说明了当 Tx FIFO 作为 Master-Transmitter 工作时 Tx FIFO 变空时 I2C 的行为，并显示了 STOP 的产生。

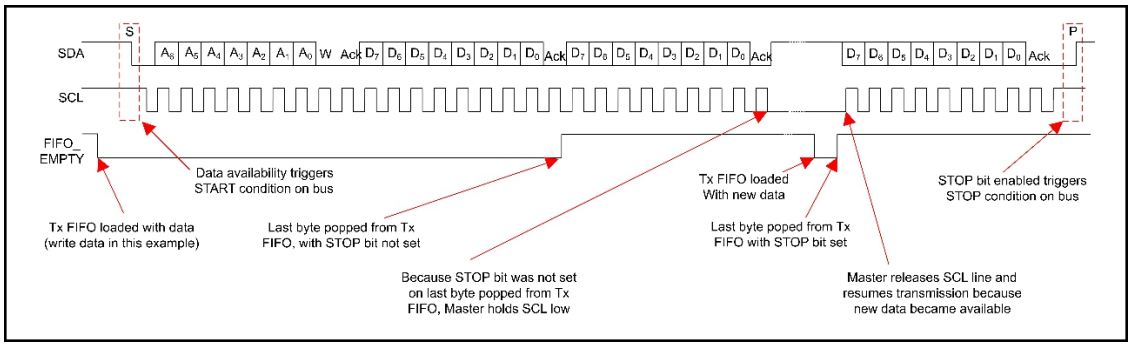


图 22-11 Master-Transmitter—Tx FIFO 变空

图 22-12 说明了当 Tx FIFO 作为 Master-Receiver 工作时 Tx FIFO 变空时 I2C 的行为，并显示了 STOP 的产生。

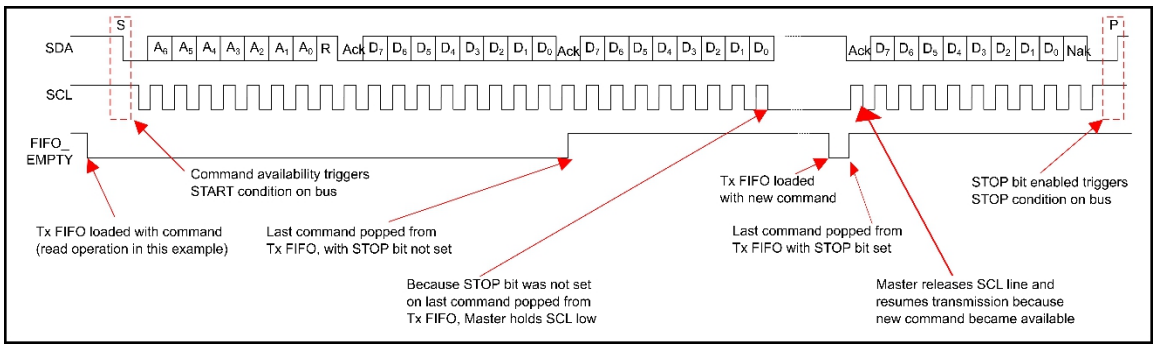


图 22-12　Master-Receiver—Tx FIFO 变空

用户可以控制 I2C 总线上 RESTART 条件的产生。如果设置了 I2C_DATA_CMD 寄存器的第 10 位（重新启动），则在将数据字节写入从设备或从设备读取数据字节之前，将产生 RESTART。

图 22-13 说明了作为 Master-Transmitter 工作期间的这种情况。

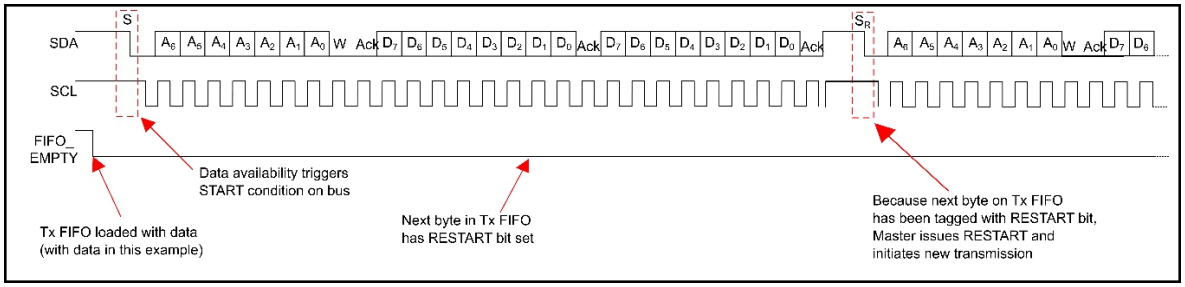


图 22-13　Master-Transmitter 工作

图 22-14 说明了作为 Master-Receiver 工作期间的这种情况。

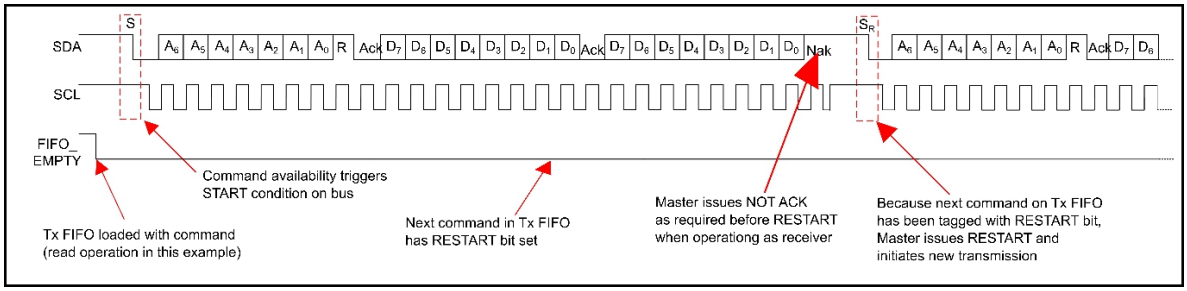


图 22-14　Master-Receiver 工作

图 22-15 说明了作为 Master-Transmitter 的操作，其中 I2C_DATA_CMD 寄存器的停止位置 1/Tx FIFO 不为空。

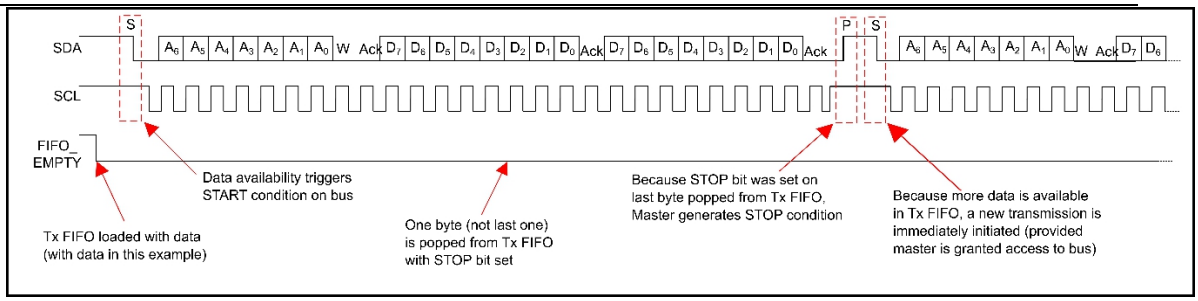


图 22-15　Master-Transmitter——停止位置 1/Tx FIFO 不为空

图 22-16 举例说明了作为 Master-Transmitter 的操作，其中允许在 TX FIFO 中装入的第一个字节变为空，并且将 RESTART 位置 1。

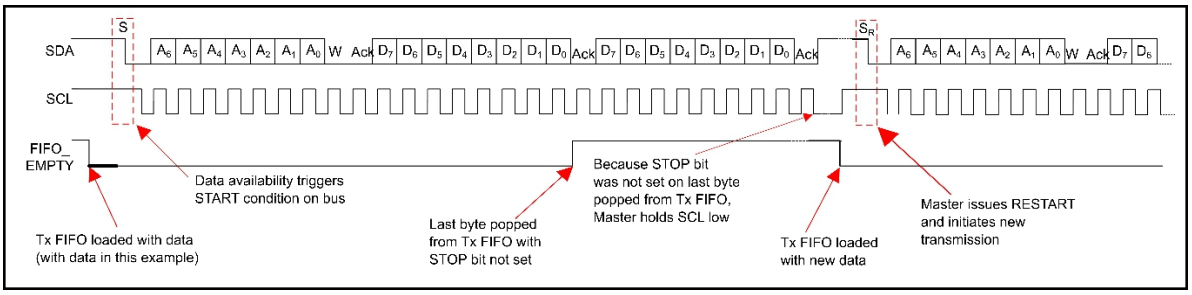


图 22-16　Master-Transmitter——停止位置 1/Tx FIFO 为空

图 22-17 说明了作为 Master-Receiver 的操作，其中 I2C_DATA_CMD 寄存器的 STOP 位置 1/Tx FIFO 不为空。

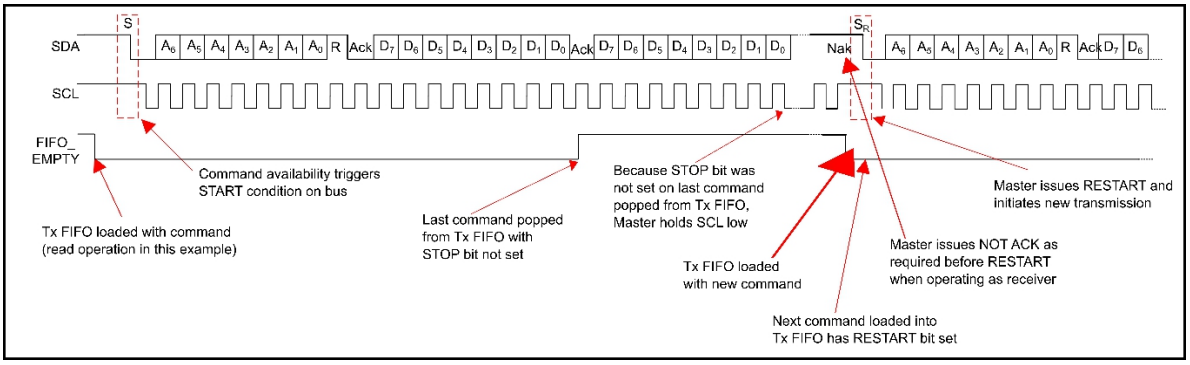


图 22-17　Master-Receiver——停止位置 1/Tx FIFO 不为空

图 22-18 说明了作为 Master-Receiver 的操作，其中允许在 Tx FIFO 之后加载的第一个命令为空，并且将 RESTART 位置 1。

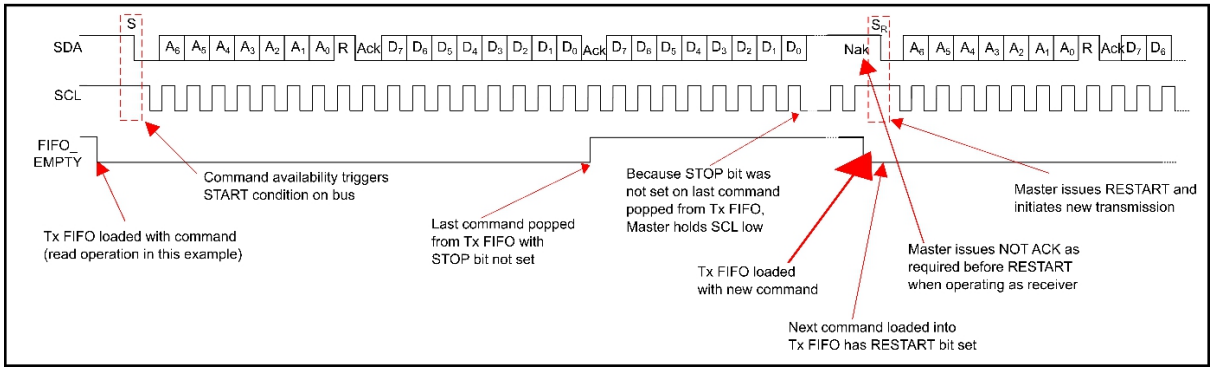


图 22-18　Master-Receiver——停止位置 1/Tx FIFO 为空

22.4.3　操作模式

I2C 在同一时间里只能设置为 I2C Master 模式或者 I2C Slave 模式，不能同时进行。也就是说，要确保 I2C_CON 寄存器的 BIT6（I2C_SLAVE_DISABLE）和 BIT0（I2C_MASTER_MODE）永远不会分别设置为 0 和 1。

22.4.3.1　Slave Mode

初始化配置

要将 I2C 用作 Slave，需要执行以下步骤：

1. 对 I2C_ENABLE 寄存器的 BIT0 写入 0 来关闭 I2C。
2. 写入 I2C_SAR 寄存器（位 9:0）以设置 Slave 地址。这是 I2C 响应的地址。
3. 写入 I2C_CON 寄存器以指定支持哪种寻址类型（通过将 BIT3 设置为 7 位或 10 位）。通过在 BIT6（I2C_SLAVE_DISABLE）中写入“ 0”，在 BIT0（MASTER_MODE）中写入“0”，以仅从机模式启用 I2C。
4. 通过在 I2C_ENABLE 寄存器的 BIT0 中写入“1”来启动 I2C。

单字节的 Slave-Transmitter

当总线上的另一个 I2C 主设备寻址 TAE32F5300 的 I2C 并请求数据时，此时 TAE32F5300 的 I2C 充当从 Transmitter，并发生以下步骤：

1. 外部 I2C 主设备使用与 TAE32F5300 的 I2C 的 I2C_SAR 寄存器中的从设备地址启动 I2C 传输。
2. TAE32F5300 的 I2C 识别地址以及传输方向，并用 ACK 响应。
3. TAE32F5300 的 I2C 产生 RD_REQ 中断（I2C_RAW_INTR_STAT 寄存器的 BIT5）并将 SCL 线保持为低电平。它处于等待状态，直到软件响应为止。

   如果由于 I2C_INTR_MASK 寄存器的 BIT5（M_RD_REQ）被设置为 0 而屏蔽了 RD_REQ 中断，则建议使用硬件或软件定时例程来指示 CPU 进行对 CPU 的定期读取 I2C_RAW_INTR_STAT 寄存器。

   读取 RD_REQ 中断后，软件必须采取行动来满足 I2C 传输要求：使用的时序间隔应为 I2C

可以处理的最快 SCL 时钟周期的 10 倍。例如，对于 400 kb / s，定时间隔为 25us。

4. 如果在接收到读取请求之前，Tx FIFO 中剩余数据，则 I2C 会产生 TX_ABRT 中断（I2C_RAW_INTR_STAT 寄存器的 BIT6）以从 TX FIFO 中清除旧数据。

注意：由于每当 TX_ABRT 中断事件发生时，I2C 的 Tx FIFO 都会被强制进入刷新/复位状态，因此在尝试写入 Tx FIFO 之前，软件必须通过读取 I2C_CLR_TX_ABRT 寄存器并从该状态释放 I2C。

5. 软件将要写入的数据写入 I2C_DATA_CMD 寄存器（通过在 BIT8 中写入“0”）。
6. 在继续之前，软件必须清除 I2C_RAW_INTR_STAT 寄存器的 RD_REQ 和 TX_ABRT 中断（分别为 BIT5 和 BIT6）。

   如果 RD_REQ 和 TX_ABRT 中断已被屏蔽，则当 R_RD_REQ 或 R_TX_ABRT 位读为 1 时，将已经执行 I2C_RAW_INTR_STAT 寄存器的清除。

7. TAE32F5300 的 I2C 释放 SCL 并发送字节。
8. 主机可以通过发出 RESTART 条件来保持 I2C 总线，也可以通过发出 STOP 条件来释放 I2C 总线。

单字节的 Slave-Receiver

当总线上的另一个 I2C 主设备寻址 TAE32F5300 的 I2C 并发送数据时，此时 TAE32F5300 的 I2C 充当从 Receiver，并发生以下步骤：

1. 外部 I2C 主设备启动 I2C 传输，其地址与 I2C_SAR 寄存器中 I2C 的从地址相匹配。
2. TAE32F5300 的 I2C 识别地址以及传输方向，并用 ACK 响应。
3. TAE32F5300 的 I2C 接收发送的字节，并将其放入接收缓冲区。

注意：如果在压入一个字节时 Rx FIFO 完全充满了数据，I2C 会将 SCL 线保持为低电平，直到 Rx FIFO 有一定空间，然后继续下一个读取请求。

4. TAE32F5300 的 I2C 产生 RX_FULL 中断（I2C_RAW_INTR_STAT [2]寄存器）。
5. 软件可以从 I2C_DATA_CMD 寄存器中读取字节（位 7：0）。
6. 另一个主设备可以通过发出 RESTART 条件来保持 I2C 总线，或者通过发出 STOP 条件来释放总线。

22.4.3.2 Master Mode

初始化配置

1. 通过将 0 写入 I2C_ENABLE 寄存器的位 0 来禁用 I2C。
2. 写入 I2C_CON 寄存器，以设置从机操作支持的最大速度模式（BIT2:1），并指定当设备是从机（BIT3）时 I2C 是否以 7/10 位寻址模式开始传输。
3. 将要寻址的 I2C 器件的地址写入 I2C_TAR 寄存器。
4. 通过将 1 写入 I2C_ENABLE 寄存器的位 0 来启用 I2C。
5. 配置传输方向和、I2C_DATA_CMD 寄存器的数据。如果在启用 I2C 之前写入了 I2C_DATA_CMD 寄存器，则当未启用 I2C 时，由于缓冲区保持清除状态，因此数据和命令将丢失。

Master-Transmitter 和 Master-Receiver

I2C 支持动态读写之间的来回切换。要发送数据，请将要写入的数据写入 I2C Rx / Tx 数据缓冲区和命令寄存器（I2C_DATA_CMD）的低字节。对于 I2C 写操作，应将 CMD 位[8]写入 0。随后若要发送命令，可通过将任意无效的数据写入 I2C_DATA_CMD 寄存器的低字节来发出读取命令，并且应将 1 写入 CMD 位。只要在发送 FIFO 中存在命令，I2C 主设备就会继续启动传输。如果发送 FIFO 变空，I2C 检查 I2C_DATA_CMD [9]是否设置为 1：

- 如果设置为 1，它将在完成当前传输后发出 STOP 条件。
- 如果设置为 0，则它将 SCL 保持为低电平，直到将下一个命令写入发送 FIFO。

22.4.3.3 中止 I2C 传输

寄存器 I2C_ENABLE_STATUS 可查询 I2C 状态，将 I2C_ENABLE 寄存器的 BIT0 设置为 0 完全关闭硬件。

I2C_ENABLE 寄存器的 ABORT 控制位允许软件在完成从 Tx FIFO 发出的传输命令之前放弃 I2C 总线。控制器通过 I2C 总线发出 STOP，然后发送 Tx FIFO 刷新。仅在主操作模式下才允许中止传输。

操作流程

1. 停止使用新命令填充 Tx FIFO（I2C_DATA_CMD）。
2. 在 DMA 模式下操作时，通过将 TDMAE 设置为 0 来禁用发送 DMA。
3. 将 I2C_ENABLE 寄存器（ABORT）的 BIT1 设置为 1。
4. 等待 M_TX_ABRT 中断。
5. 读取 I2C_TX_ABRT_SOURCE 寄存器以将源标识为 ABRT_USER_ABRT。

22.4.4 总线清除功能

I2C 支持总线清除功能。该功能可在时钟或数据线卡在 LOW 位置时，使数据（SDA）和时钟（SCL）线恢复正常。

22.4.4.1 SDA 线卡在低电平

如果 SDA 线卡在低状态，则主机执行以下操作以进行恢复，如图 2219、图 22-20 所示：

1. 主机最多发送 9 个时钟脉冲并试图在这 9 个时钟内恢复总线。
   - 时钟脉冲的数量将随从设备要发送的剩余位数而变化。由于最大位数为 9，主机发送最多 9 个时钟脉冲，并允许从机恢复。
   - 主站尝试在 SDA 线上声明逻辑 1，并检查 SDA 是否已恢复。如果未恢复 SDA，它将继续发送最多 9 个 SCL 时钟。

2. 如果 SDA 线在 9 个时钟脉冲内恢复，则主机将发送 STOP 来释放总线。
3. 如果即使在第 9 个时钟脉冲后 SDA 线仍未恢复，则系统需要硬件复位。

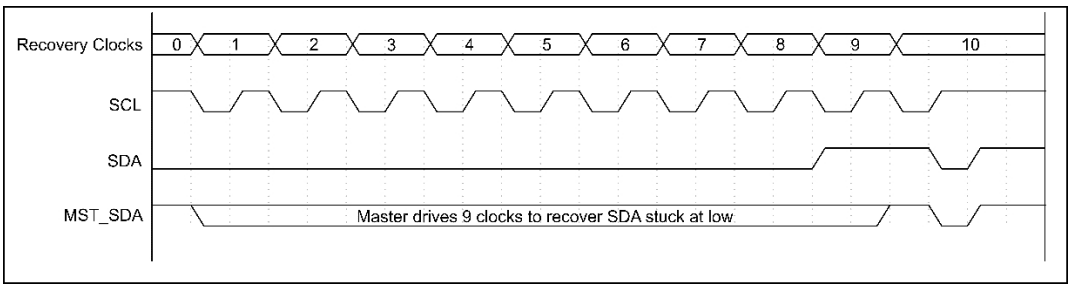


图 22-19　9 个 SCL 时钟的 SDA 恢复

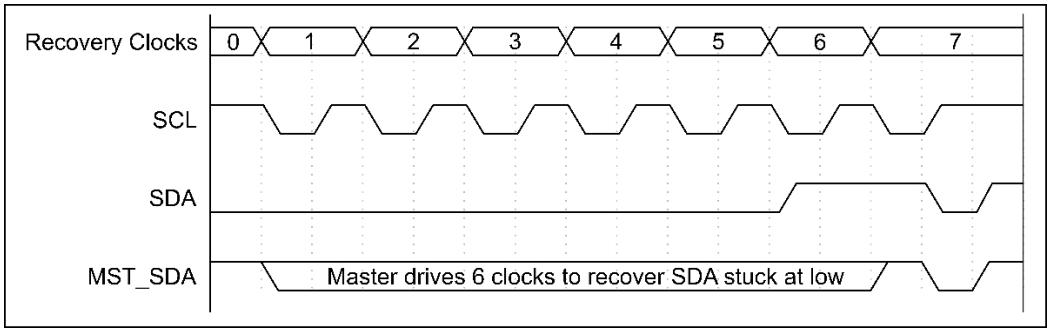


图 22-20　6 个 SCL 时钟的 SDA 恢复

22.4.4.2 SCL 线卡在低电平

电路的电气时钟（SCL）保持为 LOW，没有克服该问题的有效方法，只能使用硬件复位信号来复位总线。

22.4.5 SMBus/PMBus

SMBus 旨在在系统与其设备之间提供可预测的通信线路。它描述了设备超时定义及其条件。

22.4.5.1 $T_{Timeout}$, MIN 参数

该参数允许 Master 或 Slave 得出结论：即有缺陷的设备将时钟无限期保持为低电平，或者主机有意试图将设备从总线上驱除。

I2C 在 SMBus 模式下启用总线清除功能，并且用户可以使用 I2C_SCL_STUCK_TIMEOUT 寄存器对 $T_{TIMEOUT, MIN}$ 值进行编程以检测 SMBCLK 低超时。在检测到 SCL_STUCK_TIMEOUT 中断后，I2C Slave 将重置其通信接口并释放 SCL 和 SDA 线。I2C Master 具有中止功能，该中止可通过配置 I2C_ENABLE [1]寄存器位来完成当前传输并在总线上生成 STOP。

22.4.5.2 Master 时钟扩展

T_{LOW}：MEXT 间隔定义为允许主设备在消息中的一个字节内延长其时钟周期的累积时间，从以下时间开始计算：

- 开始确认
- ACK 到 ACK
- 确认停止。

I2C 主设备使用 I2C_SMBUS_CLOCK_LOW_MEXT 寄存器检测主设备时钟扩展超时并生成 SMBUS_CLK_LOW_MEXT 中断。

22.4.5.3 Slave 时钟扩展

T_{LOW}：SEXT 间隔是允许给定从设备将一条消息中的时钟周期从初始 START 扩展到 STOP 的累积时间。

I2C 主设备使用 I2C_SMBUS_CLOCK_LOW_SEXT 寄存器检测从设备时钟扩展超时并生成 SMBUS_CLK_LOW_SEXT 中断。

通过允许用户中止（I2C_ENABLE [1]），允许主机中止与任何违反 T_{LOW}：SEXT 或 $T_{TIMEOUT, MIN}$ 规范的从机的正在进行的事务。

22.4.5.4 SMBDAT 低电平超时

发生故障的设备将 SMBDAT 线路无限期拉低。这将阻止主机发出 STOP 条件并结束事务。如果在事务结束时 SMBCLK 变为高电平后 SMBDAT 仍为低 $T_{TIMEOUT, MAX}$，则主机应至少在 $T_{TIMEOUT, MAX}$ 上将 SMBCLK 保持低电平，以尝试重置总线上所有设备的 SMBus 接口。

I2C 在 SMBus 模式下启用总线清除功能，并且用户可以使用 I2C_SDA_STUCK_TIMEOUT 寄存器来编程 SMBDAT 超时值以检测 SMBDAT 低电平超时。如果 SMBDAT 线卡在低电平，则会产生 SDA_STUCK_TIMEOUT 中止，并且软件可以启用 I2C_ENABLE 寄存器的 SMBUS_CLK_RESET 寄存器位以将 I2C_SCL_STUCK_TIMEOUT 的 SCL 保持为低电平，从而复位总线上所有设备的 SMBus 接口。

22.4.5.5 总线协议

典型的 SMBus 设备有一组命令，通过这些命令可以读取和写入数据。所有命令的长度均为一字节，而其参数和返回值的长度可以不同。根据 SMBus 规范，最高有效位（MSB）首先被传输。对于任何给定的设备，有十一种命令协议。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、块写入和块写入-块读取过程调用。

用于消息事务的 SMBus 协议通常与 I2C 数据传输命令不同。仍然可以对 SMBus 主设备进行编程以传递 I2C 数据传输命令。下表描述了通过 I2C 中的 Tx-FIFO 命令派生 SMBus 总线协议。

在 SMBus 主模式下，所有接收数据字节将在 Rx-FIFO 中可用。在 SMBus 从模式下，所有总线协议命令代码和数据字节将在 Rx-FIF0 中接收，并且读取请求数据字节必须使用 Tx-FIFO 发送，类似于 I2C 模式。

表 22-2　I2C 中的 SMBus 总线协议用法

Protocol	TxFIFO 命令	命令/数据（I2C_DATA_CMD [7：0]）	CMD 位（I2C_DATA_CMD [8]）	停止位（I2C_DATA_CMD [9]）	备注
快速命令	1	不适用	设置命令[R / W]	置 1	将 I2C_TAR [11]和 I2C_TAR [16]设置为 1
发送字节	1	数据字节	置 0	置 1	
接受字节	1	不适用	置 1	置 1	
写入字节	2	命令码	置 0	置 0	
		数据字节	置 0	置 1	
写入字	3	命令码	置 0	置 0	
		数据字节低位	置 0	置 0	
		数据字节高位	置 0	置 1	
读取字节	2	命令码	置 0	置 0	
		不适用	置 1	置 1	
读取字	3	命令码	置 0	置 0	
		不适用	置 1	置 0	
		不适用	置 1	置 1	
过程调用	5	命令码	置 0	置 0	
		数据字节低位	置 0	置 0	
		数据字节高位	置 0	置 0	
		不适用	置 1	置 0	
		不适用	置 1	置 1	
块写入	N+1	命令码	置 0	置 0	
		数据字节 1	置 0	置 0	
		（N+1）数据字节 N	置 0	置 1	
块读取	N+1	命令码	置 0	置 0	
		不适用	置 0	置 0	
		（N+1）不适用	置 0	置 1	
块写入-块读取过程调用	M+N+1	命令码	置 0	置 0	
		数据字节 1	置 0	置 0	
		（M+1）数据字节 M	置 0	置 0	
		（M+2）不适用	置 1	置 0	
		（M+2）不适用	置 1	置 0	
		（M+N+1）不适用	置 1	置 1	
SMBUS 主机通知协议	3	数据字节低位	置 0	置 0	将 I2C_TAR [6：0]设置为 SMB 主机地址（0001 000）
		数据字节高位	置 0	置 1	

通过使能 I2C_CON 寄存器中的 SLAVE_QUICK_CMD_EN 位，可以使 I2C 从设备仅接收快速命令。每当选择该位时，从站仅接收快速命令，并且不接受其他总线协议。 I2C 从站在接收到 QUICK 命令后发出 SMBUS_QUICK_DET 中断。

SMBus 通过在总线协议末尾附加 PEC 字节，引入了一种包错误检查机制。这可以通过在软件接收时传输和解码时添加一个额外的命令（PEC 字节）来实现。

22.4.5.6 SMBUS 地址解析协议

通过主机为每个从设备动态分配一个新的唯一地址，可以解决 SMBus 从地址冲突。SMBus 为系统中的每个设备引入了一个 128 位的唯一设备 ID（UDID），以隔离每个设备以进行地址分配。I2C 将 I2C_SMBUS_UDID_MSB 参数用于高 96 位常数，将“I2C_SMBUS_ARP_UDID_LSB”寄存器用于 UDID 32 位低变量。I2C 使用 I2C_CON 寄存器中的 PERSISTANT_SLV_ADDR_EN 寄存器位指示 I2C 是否支持持久性从地址。

I2C Master 可以发出通用和定向的地址解析协议（ARP）命令，以为 SMBus 系统中的从站分配动态地址。

表 22-3 描述了通过 i2c 中的 Tx-FIFO 命令派生 SMBus ARP 命令。

表 22-3 通过 I2C 中的 TxFIFO 命令派生 SMBus ARP 命令

ARP 命令	TxFIFO 命令	命令/数据（I2C_DATA_CMD [7：0]）	CMD 位（I2C_DATA_CMD [8]）	停止位（I2C_DATA_CMD [9]）	备注
准备 ARP	2	Command = '0000 0001'	置 0	置 0	将 I2C_TAR [6：0]设置为 SMB 默认地址（1100 001）
		PEC 字节	置 0	置 1	
复位设备（General）	2	Command = '0000 0010'	置 0	置 0	将 I2C_TAR [6：0]设置为 SMB 默认地址（1100 001）
		PEC 字节	置 0	置 1	
获取 UDID（General）	20	Command = '0000 0011'	置 0	置 0	1.将 I2C_TAR [6：0]设置为 SMB 默认地址（1100001）。 2.16 对 128 个 UDID 字节执行读取。 3.从站地址的最后读取命令。
		不适用	置 1	置 0	
		不适用	置 1	置 0	
		不适用	置 1	置 0	
		PEC 字节	置 0	置 1	

分配地址	20	Command = '0000 0011'	置 0	置 0	1.将 I2C_TAR [6：0]设置为 SMB 默认地址（1100001）。 2.16 将对 128 UDID 字节执行写操作。 3.分配从站地址的最后写入命令。
		字节数=17	置 0	置 0	
		UDID 字节 15	置 0	置 0	
		UDID 字节 14	置 0	置 0	
		分配地址	置 0	置 0	
		PEC 字节	置 1	置 1	
获取 UDID（Directed）	19	Command = '0000 0011'	置 0	置 0	1.将 I2C_TAR [6：0]设置为 SMB 默认地址（1100001）。 2.16 对 128 UDID 字节执行读取。 3.从站地址的最后读取命令。
		{从地址[6:0],1}	置 1	置 0	
		不适用	置 1	置 0	
		不适用	置 1	置 0	
		PEC 字节	置 1	置 1	
复位设备（Directed）	2	Command = {从地址[6:0],1}	置 0	置 0	将 I2C_TAR [6:0]设置为 SMB 默认地址（1100001）
		不适用	置 0	置 1	
通知 ARP 主机	3	设备地址 =’11000010’	置 0	置 0	将 I2C_TAR [6：0]设置为 SMB 主机地址（0001000）
		数据字节低位 =’00000000’	置 0	置 0	
		数据字节高位 =’00000000’	置 0	置 1	

22.4.6 I2C_CLK 频率配置

当 I2C 配置为标准（SS），快速（FS）主设备时，必须先设置* CNT 寄存器，然后才能进行任何 I2C 总线事务以确保正确的 I / O 时序。 * CNT 寄存器是：

- I2C_SS_SCL_HCNT
- I2C_SS_SCL_LCNT
- I2C_FS_SCL_HCNT
- I2C_FS_SCL_LCNT

注意：START，STOP 和 RESTART 寄存器的 tBUF 时序和建立/保持时间将 HCNT / * LCNT 寄存器设置用于相应的速度模式。*

下表列出了来自* CNT 编程寄存器的 I2C 时序参数。

表 22-4 从* CNT 寄存器推导 I2C 时序参数

时序参数	符号	标准（SS）	快速（FS）
SCL 时钟的低电平周期	tLOW	I2C_SS_SCL_LCNT	I2C_FS_SCL_LCNT
SCL 时钟的高电平周期	tHIGH	I2C_SS_SCL_HCNT	I2C_FS_SCL_HCNT
重复 START 的建立时间	tSU;STA	I2C_SS_SCL_LCNT	I2C_FS_SCL_HCNT
重复 START 的保持时间	tHD;STA	I2C_SS_SCL_HCNT	I2C_FS_SCL_HCNT
STOP 的建立时间	tSU;STO	I2C_SS_SCL_HCNT	I2C_FS_SCL_HCNT
STOP 和 START 条件之间的总线空闲时间	tBUF	I2C_SS_SCL_LCNT	I2C_FS_SCL_LCNT
Spike length	tSP	I2C_FS_SPKLEN	I2C_FS_SPKLEN
数据保持时间	tHD;DAT	I2C_SDA_HOLD	I2C_SDA_HOLD
数据建立时间	tSU;DAT	I2C_SDA_SETUP	I2C_SDA_SETUP

SS，FS 模式下的计数

- 由 I2C 驱动的总 SCL LOW 周期为 I2C_*_LCNT 寄存器值。硬件不支持将小于 6 的值写入 I2C_*_LCNT 寄存器。此外，I2C 能够支持的最小 SCL 低电平时间为 6 I2C_clk 周期。
- 由 I2C 驱动的总 SCL HIGH 周期为 I2C_*_ HCN 寄存器值+ SPKLEN +3。此外，I2C 能够支持的 SCL 高电平最小时间为 5 I2C_clk 周期[1 + 1 + 3]。

I2C 主设备生成的 SCL 的总高电平时间和低电平时间也受 SCL 线的上升时间和下降时间影响。SCL 上升和下降时间参数取决于外部因素，例如：

- IO 驱动程序的特征
- 上拉电阻值
- SCL 线上的总电容，依此类推

这些特性超出了 I2C 的控制范围。

22.4.7 DMA 控制接口

I2C 具有内置 DMA 功能，它具有与 DMA 控制器的握手接口，以请求和控制传输。 APB 总线用于执行与 DMA 之间的数据传输。

22.5 寄存器描述

22.5.1 寄存器列表

Name	Offset	Width	Description
I2C_CON	0x00	[19:0]	I2C Control
I2C_TAR	0x04	[15:0]	I2C Target Address
I2C_SAR	0x08	[9:0]	I2C Slave Address
I2C_DATA_CMD	0x10	[15:0]	I2C Rx/Tx Data Buffer and Command
I2C_SS_SCL_HCNT	0x14	[15:0]	Standard speed I2C Clock SCL High Count
I2C_SS_SCL_LCNT	0x18	[15:0]	Standard speed I2C Clock SCL Low Count
I2C_FS_SCL_HCNT	0x1C	[15:0]	Fast Mode and Fast Mode Plus I2C Clock SCL High Count
I2C_FS_SCL_LCNT	0x20	[15:0]	Fast Mode and Fast Mode Plus I2C Clock SCL Low Count
I2C_INTR_STAT	0x2C	[15:0]	I2C Interrupt Status
I2C_INTR_MASK	0x30	[15:0]	I2C Interrupt Mask
I2C_RAW_INTR_STAT	0x34	[15:0]	I2C Raw Interrupt Status
I2C_RX_TL	0x38	[7:0]	I2C Receive FIFO Threshold
I2C_TX_TL	0x3C	[7:0]	I2C Transmit FIFO Threshold
I2C_CLR_INTR	0x40	[0:0]	Clear Combined and Individual Interrupts
I2C_CLR_RX_UNDER	0x44	[0:0]	Clear RX_UNDER Interrupt
I2C_CLR_RX_OVER	0x48	[0:0]	Clear RX_OVER Interrupt
I2C_CLR_TX_OVER	0x4C	[0:0]	Clear TX_OVER Interrupt
I2C_CLR_RD_REQ	0x50	[0:0]	Clear RD_REQ Interrupt
I2C_CLR_TX_ABRT	0x54	[0:0]	Clear TX_ABRT Interrupt
I2C_CLR_RX_DONE	0x58	[0:0]	Clear RX_DONE Interrupt
I2C_CLR_ACTIVITY	0x5C	[0:0]	Clear ACTIVITY Interrupt
I2C_CLR_STOP_DET	0x60	[0:0]	Clear STOP_DET Interrupt
I2C_CLR_START_DET	0x64	[0:0]	Clear START_DET Interrupt
I2C_CLR_GEN_CALL	0x68	[0:0]	Clear GEN_CALL Interrupt
I2C_ENABLE	0x6C	[18:0]	I2C Enable
I2C_STATUS	0x70	[20:0]	I2C Status register
I2C_TXFLR	0x74	[3:0]	Transmit FIFO Level Register
I2C_RXFLR	0x78	[3:0]	Receive FIFO Level Register
I2C_TX_ABRT_SOURCE	0x80	[31:0]	I2C Transmit Abort Status Register
I2C_SLV_DATA_NACK_ONLY	0x84	[0:0]	Generate SLV_DATA_NACK Register

I2C_DMA_CR	0x88	[1:0]	DMA Control Register for transmit and receive handshaking interface
I2C_DMA_TDLR	0x8C	[2:0]	DMA Transmit Data Level
I2C_DMA_RDLR	0x90	[2:0]	DMA Receive Data Level
I2C_ACK_GENERAL_CALL	0x98	[0:0]	I2C ACK General Call Register
I2C_ENABLE_STATUS	0x9C	[2:0]	I2C Enable Status Register
I2C_CLR_RESTART_DET	0xA8	[0:0]	Clear RESTART_DET Interrupt
I2C_SCL_STUCK_AT_LOW_TIMEOUT	0xAC	[31:0]	I2C SCL stuck at low timeout register
I2C_SDA_STUCK_AT_LOW_TIMEOUT	0xB0	[31:0]	2C SDA Stuck at Low Timeout
I2C_CLR_SCL_STUCK_DET	0xB4	[0:0]	Clear SCL Stuck at Low Detect Interrupt Register
I2C_SMBUS_CLOCK_LOW_SEXT	0xBC	[31:0]	SMBUS Slave Clock Extend Timeout Register
I2C_SMBUS_CLOCK_LOW_MEXT	0xC0	[31:0]	SMBUS Master extend clock Timeout Register
I2C_SMBUS_THIGH_MAX_IDLE_COUNT	0xC4	[15:0]	SMBus Thigh MAX Bus-Idle count Register
I2C_SMBUS_INTR_STAT	0xC8	[31:0]	I2C SMBUS Interrupt Status Register
I2C_SMBUS_INTR_MASK	0xCC	[31:0]	I2C Interrupt Mask Register
I2C_SMBUS_INTR_RAW_STATUS	0xD0	[31:0]	I2C SMBUS Raw Interrupt Status Register
I2C_CLR_SMBUS_INTR	0xD4	[31:0]	Clear SMBUS Interrupt Register
I2C_OPTIONAL_SAR	0xD8	[6:0]	I2C Optional Slave Address Register
I2C_SMBUS_UDID_LSB	0xDC	[31:0]	SMBUS ARP UDID LSB Register

22.5.2 寄存器详细描述

22.5.2.1 I2C_CON

- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-20]	Reserved	R	0x0	Reserved
[19]	SMBUS_PERSISTANT_SLV_ADDR_EN	R/W	0x0	该位控制将 I2C 从站启用为持久或非持久从站。 如果从设备不是 PSA，则 I2C 从设备将清除 General 和 Directed 重置 ARP 命令的“地址有效”标志，否则该地址有效标志将始终设置为 1。
[18]	SMBUS_ARP_EN	R/W	0x0	该位控制 I2C 是否应在 SMBus 模式下启用地址解析逻辑。 从模式将解码地址解析协议命令并对其作出响应。I2C 从站还包括地址解析协议命令的 PEC 字节的生成/有效性。 该位仅适用于从模式。
[17]	SMBUS_SLAVE_QUICK_CMD_EN	R/W	0x0	如果该位设置为 1，则 I2C 从站仅在 SMBus 模式下接收快速命令。 如果此位设置为 0，则 I2C 从站将接收所有总线协议，但不会接收快速命令。 该位仅适用于从机模式。
[16]	OPTIONAL_SAR_CTRL	R/W	0x0	用户必须在 I2C_OPTIONAL_SAR 中编程一个有效地址，然后才能向该字段写入 1。
[15-12]	Reserved	R	0x0	Reserved
[11]	BUS_CLEAR_FEATURE_CTRL	R/W	0x0	在主机模式下： ● 1'b1：启用了总线清除功能 ● 1'b0：总线清除功能已禁用 在从模式下，该寄存器位不适用。
[10]	STOP_DET_IF_MASTER_ACTIVE	R/W	0x0	在主控模式下 ● 1’b1：仅当主机处于活动状态时才发出 STOP_DET 中断 ● 1’b0：发出 STOP_DET，与主服务器是否处于活动状态无关
[9]	RX_FIFO_FULL_HLD_CTRL	R/W	0x0	当 RXFIFO 已经满到极限（8）时，该位控制 I2C 是否应保持总线： ● 1'b1：保持总线 ● 1'b0：不保持总线
[8]	TX_EMPTY_CTRL	R/W	0x0	如 I2C_RAW_INTR_STAT 寄存器中所述，该位控制 TX_EMPTY 中断的产生。
[7]	STOP_DET_IFADDRESSED	R/W	0x0	在从模式下： ● 1’b1 ：仅在被寻址时发出 STOP_DET 中断。 ● 1’b0 ：发出 STOP_DET，无论它是否被寻址。
[6]	I2C_SLAVE_DISABLE	R/W	0x0	该位控制 I2C 是否禁用其从机 0：启用从站 1：禁用从站

[5]	I2C_RESTART_EN	R/W	0x0	确定充当主机时是否可以发送重新启动条件。 一些较早的从站不支持处理 RESTART 条件； 但是，在几个 I2C 操作中使用了 RESTART 条件。 0：禁用 1：启用 当禁用 RESTART 时，I2C 主站无法执行以下功能： ● 发送开始字节 ● 执行任何高速模式操作 ● 以组合格式模式执行方向更改 ● 使用 10 位地址执行读取操作 通过将 RESTART 条件替换为 STOP 和后续的 START 条件，拆分操作被分解为多个 I2C 传输。如果执行上述操作，将导致 I2C_RAW_INTR_STAT 寄存器的第 6 位（TX_ABRT）置 1
[4]	I2C_10BITADDR_MASTER	R/W	0x0	控制 I2C 充当主机时是以 7 位或 10 位寻址模式开始传输： 0：7 位寻址 1：10 位寻址
[3]	I2C_10BITADDR_SLAVE	R/W	0x0	当作为从设备时，该位控制 I2C 是否响应 7 位或 10 位地址。 0：7 位寻址。 I2C 忽略涉及 10 位寻址的事务。 对于 7 位寻址，仅比较 I2C_SAR 寄存器的低 7 位。 1：10 位寻址。 I2C 仅响应与 I2C_SAR 寄存器的全部 10 位匹配的 10 位寻址传输。
[2-1]	SPEED	R/W	0x0	这些位控制 I2C 的运行速度。 1：标准模式（0 到 100 Kb / s） 2：快速模式（≤400 Kb / s）或快速模式加（≤1000 Kb / s） 3：高速模式（≤3.4 Mb / s）
[0]	MASTER_MODE	R/W	0x0	该位控制是否启用 I2C 主设备。 0：禁用主机 1：启用主机

22.5.2.2 I2C_TAR

- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-17]	Reserved	R	0x0	Reserved
[16]	SMBUS_QUICK_CMD	R/W	0x0	如果位 11（SPECIAL）设置为 1，则该位指示 I2C 是否要执行快速命令。
[15-14]	Reserved	R	0x0	Reserved
[13]	Device_ID	R/W	0x0	如果位 11（SPECIAL）设置为 1，则此位指示 I2C 主设备是否要执行 I2C_TAR [6：0]中提到的特定从设备的 Device-ID。 0：不执行设备 ID，并检查 I2C_tar [10]以执行常规调用或 START 字节命令。 1：执行设备 ID 传输，并且从目标从站接收基于 Tx-FIFO 中读取命令数量的字节，并将其放入 Rx-FIFO 中。
[12]	I2C_10BITADDR_MASTER	R/W	0x0	当充当主机时，该位控制 I2C 是否以 7 位或 10 位寻址模式开始其传输。 0：7 位寻址 1：10 位寻址
[11]	SPECIAL	R/W	0x0	该位指示软件是否执行设备 ID，常规呼叫或 START BYTE 命令。 0：忽略位 10 GC_OR_START 并正常使用 I2C_TAR 1：执行设备 ID 或 GC_OR_START 位中指定的特殊 I2C 命令
[10]	GC_OR_START	R/W	0x0	如果位 11（SPECIAL）设置为 1，位 13（Device-ID）设置为 0，则该位指示 I2C 是执行通用调用还是启动字节命令。 0：通用呼叫地址–发出通用呼叫后，只能执行写操作。 尝试发出读取命令会导致 I2C_RAW_INTR_STAT 寄存器的第 6 位（TX_ABRT）置 1。 I2C 保持在通用调用模式，直到 SPECIAL 位值（位 11）被清除为止。 1：开始字节
[9-0]	I2C_TAR	R/W	0x0	这是任何主通信设备的目标地址。 在发送一般呼叫时，这些位将被忽略。要生成 START BYTE，CPU 只需向这些位写入一次。

22.5.2.3 I2C_SAR

- **Size**：32bits
- **Offset**：0x08
- **Default**：0x55

Bit	Field	R/W	Default	Description
[31-10]	Reserved	R	0x0	Reserved
[9-0]	I2C_SAR	R/W	0x55	当 I2C 作为从设备运行时，I2C_SAR 保留从设备地址。 对于 7 位寻址，仅使用 I2C_SAR [6：0]。 仅当禁用 I2C 接口（对应于将 I2C_ENABLE [0]设置为 0）时，才可以写入该寄存器。在其他时间，写操作无效。

22.5.2.4 I2C_DATA_CMD

- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	R	0x0	Reserved
[11]	FIRST_DATA_BYTE	R	0x0	指示在地址阶段之后在主接收器或从接收器模式下接收传输的第一个数据字节。
[10]	RESTART	W	0x0	该位控制在发送或接收字节之前是否发出 RESTART。 仅当 I2C_EMPTYFIFO_HOLD_MASTER_EN 配置为 1 时，此位才可用。 1：如果 I2C_RESTART_EN 为 1，则在发送/接收数据之前（根据 CMD 的值）将发出 RESTART，无论传输方向是否从上一个命令改变。 如果 I2C_RESTART_EN 为 0，则发出 STOP，然后发出 START。 0：如果 I2C_RESTART_EN 为 1，则仅当传输方向从上一条命令改变时才发出 RESTART；否则，将发出 RESTART。 如果 I2C_RESTART_EN 为 0，则发出 STOP，然后发出 START。
[9]	STOP	W	0x0	该位控制在发送或接收字节之后是否发出 STOP。 1：在该字节之后发出 STOP，无论 Tx FIFO 是否为空。 如果 Tx FIFO 不为空，则主机通过发出 START 并为总线进行仲裁，立即尝试开始新的传输。 0：在该字节之后不发出 STOP，无论 Tx FIFO 是否为空。 如果 Tx FIFO 不为空，则主机通过根据 CMD 位的值发送/接收数据字节来继续当前传输。 如果 Tx FIFO 为空，则主机将 SCL 线保持为低电平并停止总线，直到 Tx FIFO 中有新命令可用为止。
[8]	CMD	W	0x0	该位控制执行读取还是写入。 当 I2C 充当从机时，该位不控制方向。 当它充当主机时，它仅控制方向。 1：读取 0：写入
[7-0]	DAT	R/W	0x0	该寄存器包含要在 I2C 总线上发送或接收的数据。 如果您正在写入该寄存器并要执行读取操作，则 I2C 将忽略位 7：0（DAT）。 但是，当您读取该寄存器时，这些位将返回 I2C 接口上接收到的数据值。

22.5.2.5 I2C_SS_SCL_HCNT

- **Size**：32bits
- **Offset**：0x14
- **Default**：0x28

Bit	Field	R/W	Default	Description
[31-16]	Reserved	R	0x0	Reserved
[15-0]	I2C_SS_SCL_HCNT	W	0x28	必须先设置此寄存器，然后才能进行任何 I2C 总线事务以确保正确的 I/O 时序。 该寄存器为标准速度设置 SCL 时钟的高周期计数。 仅当禁用 I2C 接口（对应于将 I2C_ENABLE [0]设置为 0）时，才可以写入该寄存器。在其他时间，写操作无效。 最小有效值为 6； 硬件会阻止小于此值的值被写入，如果强制写入小于 6 也会被设置为 6。

22.5.2.6 I2C_SS_SCL_LCNT

- **Size**：32bits
- **Offset**：0x18
- **Default**：0x2f

Bit	Field	R/W	Default	Description
[31-16]	Reserved	R	0x0	Reserved
[15-0]	I2C_SS_SCL_LCNT	R	0x2f	必须先设置此寄存器，然后才能进行任何 I2C 总线事务以确保正确的 I/O 时序。 该寄存器设置标准速度的 SCL 时钟低电平周期计数。 仅当禁用 I2C 接口（对应于将 I2C_ENABLE [0]设置为 0）时，才可以写入该寄存器。在其他时间，写操作无效。 最小有效值为 8； 硬件会阻止小于此值的值被写入，如果强制写入小于 8 也会被设置 8.

22.5.2.7 I2C_FS_SCL_HCNT

- **Size**：32bits
- **Offset**：0x1C
- **Default**：0x6

Bit	Field	R/W	Default	Description
[31-16]	Reserved	R	0x0	Reserved
[15-0]	I2C_FS_SCL_HCNT	R	0x6	必须先设置此寄存器，然后才能进行任何 I2C 总线事务以确保正确的 I/O 时序。该寄存器设置快速模式或快速模式加的 SCL 时钟高周期计数。 仅当禁用 I2C 接口（对应于将 I2C_ENABLE [0]设置为 0）时，才可以写入该寄存器。在其他时间，写操作无效。 最小有效值为 6； 硬件会阻止小于此值的值被写入，如果强制写入小于 6 也会被设置为 6。

22.5.2.8 I2C_FS_SCL_LCNT

- **Size**：32bits
- **Offset**：0x20
- **Default**：0xd

Bit	Field	R/W	Default	Description
[31-16]	Reserved	R	0x0	Reserved
[15-0]	I2C_FS_SCL_LCNT	R	0xd	必须先设置此寄存器，然后才能进行任何 I2C 总线事务以确保正确的 I / O 时序。该寄存器设置快速模式或快速模式加的 SCL 时钟低电平周期计数。 仅当禁用 I2C 接口（对应于将 I2C_ENABLE [0]设置为 0）时，才可以写入该寄存器。在其他时间，写操作无效。 最小有效值为 8； 硬件会阻止小于此值的值被写入，如果强制写入小于 8 也会被设置 8。

22.5.2.9 I2C_INTR_STAT

- **Size**：32bits
- **Offset**：0x2C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-15]	Reserved	R	0x0	Reserved
[14]	R_SCL_STUCK_AT_LOW	R	0x0	SCL 低电平 STUCK 控制位。该位指示是否 SCL 线卡在 I2C_SCL_STUCK_LOW_TIMOUT 个 I2C_clk 周期的低电平上。
[13]	R_MST_ON_HOLD	R	0x0	该位指示是否主机保持总线并且 Tx FIFO 为空。
[12]	R_RESTART_DET	R	0x0	该位指示当 IIC 在从机模式下运行且从机是寻址从机时，I2C 接口上是否发生了 RESTART 条件。 注意：但是，在高速模式或 START BYTE 传输期间，根据 I2C 协议，RESTART 位于地址字段之前。在这种情况下，因此，当发出 RESTART 时，从机不是被寻址的从机，因此 IIC 不会生成 RESTART_DET 中断。
[11]	R_GEN_CALL	R	0x0	只有当接收到 Genaral Call 地址并确认时设置该位。通过禁用 IIC 或者 CPU 读取 I2C_CLR_GEN_CALL[0]时清除该位。IIC 将接收的数据存储在 Rx buffe 中。
[10]	R_START_DET	R	0x0	该位指示无论 IIC 工作在主模式还是从模式下，START 或 RESTART 条件是否发生。
[9]	R_STOP_DET	R	0x0	"无论 IIC 工作在主模式还是从模式下，该位指示 IIC 是否发生了 STOP 条件。 从模式下： 如果 CON[7] = 1（STOP_DET_IFADDRESSED），只有当从机被寻址时才会发生 STOP_DET 中断。 注意：在 General Call 地址期间，如果 STOP_DET_IFADDRESSED = 1，即使从机通过产生 ACK 来响应 General Cal 地址，从机不会发出 STOP_DET 中断。 如果 CON[7] = 0（STOP_DET_IFADDRESSED），无论是否被寻址都会发出 STOP_DET 中断。 主模式下： 如果 CON[10] = 1（STOP_DET_IF_MASTER_ACTIVE)），只有当主机激活时才会发出 STOP_DET 中断。 如果 CON[10] = 0（STOP_DET_IFADDRESSED），无论主机是否为激活状态都会发出 STOP_DET 中断。"
[8]	R_ACTIVITY	R	0x0	"该位捕获并保持 IIC 激活状态直到该位被清除。有以下四种方法清除： ● 禁用 IIC ● 读取 CLR_ACTIVITY 寄存器 ● 读取 CLR_INTR 寄存器 ● 系统复位 注意：一旦该位被设置，只能用上述四种方法才能清除该位。即使 IIC 模块空闲，该位仍保持置位状态，直到清零为止，这表明总线上有活动。"

[7]	R_RX_DONE	R	0x0	当 IIC 作为从机发送时，如果主机没有确认最后一个发送字节，该位置 1，传输结束。
[6]	R_TX_ABRT	R	0x0	该位指示作为 I2C 发送器的 IIC 是否无法完成对发送 FIFO 内容的预期操作。 这种情况既可以作为 I2C 主设备发生，也可以作为 I2C 从设备发生，并称为“发送中止”。
[5]	R_RD_REQ	R	0x0	当 IIC 作为从机并且另一个 IIC 主机尝试从该 IIC 从机上读取数据时，该位置 1。在发生中断前 IIC 保持总线等待 SCL = 0，这意味着从机已由要求传输数据的远程主机处寻址。处理器需要必须响应中断并将响应的数据写进 DATA_CMD 寄存器中。当处理器读取 CLR_RD_REQ 寄存器之后该位清 0。
[4]	R_TX_EMPTY	R	0x0	TX_EMPTY 中断响应的行为根据 CON 寄存器中的 TX_EMPTY_CTRL 的选择而不同。 当 TX_EMPTY_CTRL = 0 时：如果传输 buffer 小于或等于 TX_TL 寄存器上的值时该位置 1. 当 TX_EMPTY_CTRL = 1 时：如果传输 buffer 小于或等于 TX_TL 寄存器上的值并且从最近弹出的内部移位寄存器的传输地址/数据的指令完成，则该位置 1。
[3]	R_TX_OVER	R	0x0	在传输过程中，如果传输 buffer 达到 I2C_TX_BUFFER_DEPTH 并且处理器通过向 DATA_CMD 寄存器写入数据来尝试发出另一个 IIC 指令，该位置 1。当禁用 IIC 时，该位保持电平直到主机或从机状态机进入空闲状态，当 I2C_en 变为 0 时，该中断被清除。
[2]	R_RX_FULL	R	0x0	当接收 buffer 达到或超过 RX_TL 寄存器的 RX_TL 阈值时设置该位。但 buffer 电平低于阈值时，该位会被自动清 0. 如果模块被禁用（ENABLE [0] = 0），则 RX FIFO 被刷新并保持复位状态; 因此 RX FIFO 未满。 因此，无论继续的活动如何，一旦 ENABLE [0]设置为 0，该位就会被清除。
[1]	R_RX_OVER	R	0x0	如果接收 buffer 完全填充到 RX_BUFFER+DEPTH 并且接收来自一个外部 IIC 器件的另一字节，该位置 1。IIC 会对此确认，但 FIFO 已满后任何接收到的数据都会丢失。如果禁用 IIC（ENABLE[0] = 0）,该位保持电平直到主机或从机的状态机空闲并且当 ENABLE[0] = 0 时，中断清 0。
[0]	R_RX_UNDER	R	0x0	当读取 DATA_CMD 寄存器时，如果处理器尝试读取接收 buffer，该位置 1。如果禁用该模块（ENABLE[0] = 0），该位会保持电平直到主机或从机状态机空闲，并且清除中断。

22.5.2.10 I2C_INTR_MASK

- **Size**：32bits
- **Offset**：0x30
- **Default**：0x7fff

Bit	Field	R/W	Default	Description
[31-15]	Reserved	R	0x0	Reserved
[14]	M_SCL_STUCK_AT_LOW	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的R_SCL_STUCK_AT_LOW中断位。写0屏蔽。
[13]	M_MST_ON_HOLD	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的R_MST_ON_HOLD 中断位。写0屏蔽。
[12]	M_RESTART_DET	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的R_RESTART_DET 中断位。写0屏蔽。
[11]	M_GEN_CALL	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[10]	M_START_DET	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[9]	M_STOP_DET	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[8]	M_ACTIVITY	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[7]	M_RX_DONE	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[6]	M_TX_ABRT	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[5]	M_RD_REQ	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[4]	M_TX_EMPTY	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[3]	M_TX_OVER	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[2]	M_RX_FULL	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[1]	M_RX_OVER	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。
[0]	M_RX_UNDER	R/W	0x1	该位屏蔽I2C_INTR_STAT寄存器中的相应的中断状态位。写0屏蔽。

22.5.2.11 I2C_RAW_INTR_STAT

- **Size**：32bits
- **Offset**：0x34
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-15]	Reserved	R	0x0	Reserved
[14]	SCL_STUCK_AT_LOW	R	0x0	SCL 低电平 STUCK 控制位。该位指示是否 SCL 线卡在 I2C_SCL_STUCK_LOW_TIMOUT 个 I2C_clk 周期的低电平上。
[13]	MST_ON_HOLD	R	0x0	该位指示是否主机保持总线并且 Tx FIFO 为空。
[12]	RESTART_DET	R	0x0	该位指示当 IIC 在从机模式下运行且从机是寻址从机时，I2C 接口上是否发生了 RESTART 条件。 注意：但是，在高速模式或 START BYTE 传输期间，根据 I2C 协议，RESTART 位于地址字段之前。 在这种情况下，因此，当发出 RESTART 时，从机不是被寻址的从机，因此 IIC 不会生成 RESTART_DET 中断。
[11]	GEN_CALL	R	0x0	只有当接收到 Genaral Call 地址并确认时设置该位。通过禁用 IIC 或者 CPU 读取 I2C_CLR_GEN_CALL[0]时清除该位。IIC 将接收的数据存储在 Rx buffe 中。
[10]	START_DET	R	0x0	该位指示无论 IIC 工作在主模式还是从模式下，START 或 RESTART 条件是否发生。
[9]	STOP_DET	R	0x0	"无论 IIC 工作在主模式还是从模式下，该位指示 IIC 是否发生了 STOP 条件。 从模式下： 如果 CON[7] = 1（STOP_DET_IFADDRESSED），只有当从机被寻址时才会发生 STOP_DET 中断。 注意：在 General Call 地址期间，如果 STOP_DET_IFADDRESSED = 1，即使从机通过产生 ACK 来响应 General Cal 地址，从机不会发出 STOP_DET 中断。 如果 CON[7] = 0（STOP_DET_IFADDRESSED），无论是否被寻址都会发出 STOP_DET 中断。 主模式下： 如果 CON[10] = 1（STOP_DET_IF_MASTER_ACTIVE)），只有当主机激活时才会发出 STOP_DET 中断。 如果 CON[10] = 0（STOP_DET_IFADDRESSED），无论主机是否为激活状态都会发出 STOP_DET 中断。"
[8]	ACTIVITY	R	0x0	"该位捕获并保持 IIC 激活状态直到该位被清除。有以下四种方法清除： ● 禁用 IIC ● 读取 CLR_ACTIVITY 寄存器 ● 读取 CLR_INTR 寄存器 ● 系统复位 注意：一旦该位被设置，只能用上述四种方法才能清除该位。即使 IIC 模块空闲，该位仍保持置位状态，直到清零为止，这表明总线上有活动。"

[7]	RX_DONE	R	0x0	当 IIC 作为从机发送时，如果主机没有确认最后一个发送字节，该位置 1，传输结束。
[6]	TX_ABRT	R	0x0	该位指示作为 I2C 发送器的 IIC 是否无法完成对发送 FIFO 内容的预期操作。 这种情况既可以作为 I2C 主设备发生，也可以作为 I2C 从设备发生，并称为“发送中止”。
[5]	RD_REQ	R	0x0	当 IIC 作为从机并且另一个 IIC 主机尝试从该 IIC 从机上读取数据时，该位置 1。在发生中断前 IIC 保持总线等待 SCL = 0，这意味着从机已由要求传输数据的远程主机处寻址。处理器需要必须响应中断并将响应的数据写进 DATA_CMD 寄存器中。当处理器读取 CLR_RD_REQ 寄存器之后该位清 0。
[4]	TX_EMPTY	R	0x0	"TX_EMPTY 中断响应的行为根据 CON 寄存器中的 TX_EMPTY_CTRL 的选择而不同。 当 TX_EMPTY_CTRL = 0 时：如果传输 buffer 小于或等于 TX_TL 寄存器上的值时该位置 1. 当 TX_EMPTY_CTRL = 1 时：如果传输 buffer 小于或等于 TX_TL 寄存器上的值并且从最近弹出的内部移位寄存器的传输地址/数据的指令完成，则该位置 1。"
[3]	TX_OVER	R	0x0	在传输过程中，如果传输 buffer 达到 I2C_TX_BUFFER_DEPTH 并且处理器通过向 DATA_CMD 寄存器写入数据来尝试发出另一个 IIC 指令，该位置 1。当禁用 IIC 时，该位保持电平直到主机或从机状态机进入空闲状态，当 I2C_en 变为 0 时，该中断被清除。
[2]	RX_FULL	R	0x0	当接收 buffer 达到或超过 RX_TL 寄存器的 RX_TL 阈值时设置该位。但 buffer 电平低于阈值时，该位会被自动清 0. 如果模块被禁用（ENABLE [0] = 0），则 RX FIFO 被刷新并保持复位状态; 因此 RX FIFO 未满。 因此，无论继续的活动如何，一旦 ENABLE [0]设置为 0，该位就会被清除。
[1]	RX_OVER	R	0x0	如果接收 buffer 完全填充到 RX_BUFFER+DEPTH 并且接收来自一个外部 IIC 器件的另一字节，该位置 1。IIC 会对此确认，但 FIFO 已满后任何接收到的数据都会丢失。如果禁用 IIC（ENABLE[0] = 0）,该位保持电平直到主机或从机的状态机空闲并且当 ENABLE[0] = 0 时，中断清 0。
[0]	RX_UNDER	R	0x0	当读取 DATA_CMD 寄存器时，如果处理器尝试读取接收 buffer，该位置 1。如果禁用该模块（ENABLE[0] = 0），该位会保持电平直到主机或从机状态机空闲，并且清除中断。

22.5.2.12 I2C_RX_TL

- **Size**：32bits
- **Offset**：0x38
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	R	0x0	Reserved
[7-0]	RX_TL	R/W	0x0	接收 FIFO 阈值 控制触发 RX_FULL 中断的条目（或更高级别）（I2C_RAW_INTR_STAT 寄存器中的位 2）。 有效范围是 0-255，另外还有硬件限制，即不允许该值设置为大于缓冲区深度的值。 如果尝试这样做，则设置的实际值将是缓冲区的最大深度。 值 0 设置 1 个条目的阈值，值 255 设置 256 个条目的阈值。

22.5.2.13 I2C_TX_TL

- **Size**：32bits
- **Offset**：0x3C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	R	0x0	Reserved
[7-0]	TX_TL	R/W	0x0	发送 FIFO 阈值 控制触发 TX_EMPTY 中断（I2C_RAW_INTR_STAT 寄存器中的位 4）的条目级别（或更低）。 有效范围是 0-255，并有其他限制，即不能将其设置为大于缓冲区深度的值。 如果尝试这样做，则设置的实际值将是缓冲区的最大深度。 值 0 设置 0 个条目的阈值，值 255 设置 255 个条目的阈值。

22.5.2.14 I2C_CLR_INTR

- **Size**：32bits
- **Offset**：0x40
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_INTR	R	0x0	读取该寄存器以清除组合的中断，所有单独的中断以及 I2C_TX_ABRT_SOURCE 寄存器。 该位不会清除硬件可清除中断，但会清除软件可清除中断。

22.5.2.15 I2C_CLR_RX_UNDER

- **Size：** 32bits
- **Offset：** 0x44
- **Default：** 0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_RX_UNDER	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 RX_UNDER 中断（位 0）。

22.5.2.16 I2C_CLR_RX_OVER

- **Size：** 32bits
- **Offset：** 0x48
- **Default：** 0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_RX_OVER	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 RX_OVER 中断（位 0）。

22.5.2.17 I2C_CLR_TX_OVER

- **Size：** 32bits
- **Offset：** 0x4C
- **Default：** 0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_TX_OVER	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 TX_OVER 中断（位 0）。

22.5.2.18 I2C_CLR_RD_REQ

- **Size：** 32bits
- **Offset：** 0x50
- **Default：** 0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_RD_REQ	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 RD_REQ 中断（位 0）。

22.5.2.19 I2C_CLR_TX_ABRT

- **Size**：32bits
- **Offset**：0x54
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_TX_ABRT	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 TX_ABRT 中断（位 0）。

22.5.2.20 I2C_CLR_RX_DONE

- **Size**：32bits
- **Offset**：0x58
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_RX_DONE	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 RX_DONE 中断（位 0）。

22.5.2.21 I2C_CLR_ACTIVITY

- **Size**：32bits
- **Offset**：0x5C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_ACTIVITY	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 ACTIVITY 中断（位 0）。

22.5.2.22 I2C_CLR_STOP_DET

- **Size**：32bits
- **Offset**：0x60
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_STOP_DET	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 STOP_DET 中断（位 0）。

22.5.2.23 I2C_CLR_START_DET

- **Size：**32bits
- **Offset：**0x64
- **Default：**0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_START_DET	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 START_DET 中断（位 0）。

22.5.2.24 I2C_CLR_GEN_CALL

- **Size：**32bits
- **Offset：**0x68
- **Default：**0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_GEN_CALL	R	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 GEN_CALL 中断（位 0）。

22.5.2.25 I2C_ENABLE

- **Size**：32bits
- **Offset**：0x6C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-19]	Reserved	R	0x0	Reserved
[18]	SMBUS_ALERT_EN	R/W	0x0	SMBUS_ALERT_CTRL 寄存器位用于控制 SMBALERT 信号的置位。 1：置位 SMBALERT 信号 在检测到来自主机的警报响应地址的确认后，该寄存器位将自动清除。
[17]	SMBUS_SUSPEND_EN	R/W	0x0	SMBUS_SUSPEND_EN 寄存器位用于控制 SMBSUS 信号的断言和断言。 0：取消置位 SMBSUS 信号 1：置位 SMBSUS 信号
[16]	SMBUS_CLK_RESET	R/W	0x0	在 SMBus 主机模式下，该位用于启动 SMBus 主时钟复位。 仅当主机空闲时才应启用此位。 每当使能此位时，SMBCLK 会在 I2C_SCL_STUCK_TIMEOUT I2C_clk 周期内保持低电平，以复位 SMBus 从器件。
[15-4]	Reserved	R	0x0	Reserved
[3]	SDA_STUCK_RECOVERY_ENABLE	R/W	0x0	如果通过 TX_ABORT 中断（I2C_TX_ABRT_SOURCE [17]）指示 SDA 停留在低电平，则此位用作控制旋钮以启动 SDA 恢复机制（即，最多发送 9 个 SCL 时钟，并停止以释放 SDA 线 ），然后自动清除该位。
[2]	TX_CMD_BLOCK	R/W	0x0	在主控模式下 ● 1’b1：即使 Tx FIFO 有要发送的数据，也会阻止 I2C 总线上的数据发送。 ● 1’b0：Tx FIFO 中的第一个数据可用后，数据自动在 I2C 总线上开始传输。
[1]	ABORT	R/W	0x0	设置后，控制器将启动传输中止。 0：ABORT 未启动或 ABORT 完成 1：正在进行中止操作 通过将该位置 1，软件可以在主机模式下中止 I2C 传输。 仅当已设置 ENABLE 时，软件才能设置此位。 否则，控制器将忽略对 ABORT 位的任何写操作。 设置后，软件无法清除 ABORT 位。 响应于 ABORT，控制器在完成当前传输后发出 STOP 指令并刷新 Tx FIFO，然后在中止操作后设置 TX_ABORT 中断。 中止操作后，ABORT 位自动清除。
[0]	ENABLE	R/W	0x0	控制是否启用 I2C。 0：禁用 I2C（TX 和 RX FIFO 保持擦除状态） 1：启用 I2C

22.5.2.26 I2C_STATUS

- **Size**：32bits
- **Offset**：0x70
- **Default**：0x6

Bit	Field	R/W	Default	Description
[31-21]	Reserved	R	0x0	Reserved
[20]	SMBUS_ALERT_STATUS	R	0x0	该位指示输入信号的状态是否为 I2C_smbus_alert_in_n。当 SMBus 设备声明 SMBus 警报信号时，声明此信号。
[19]	SMBUS_SUSPEND_STATUS	R	0x0	该位指示输入信号的状态是否为 I2C_smbus_sus_in_n。 当 SMBus 主机发出 SMBus 挂起信号时，发出该信号。
[18]	SMBUS_SLAVE_ADDR_RESOLVED	R	0x0	该位指示 ARP 主机是否解析 SMBus 从机地址（I2C_sar [6：0]）。
[17]	SMBUS_SLAVE_ADDR_VALID	R	0x0	该位指示 SMBus 从设备地址（I2C_sar [6：0]）是否有效。
[16]	SMBUS_QUICK_CMD_BIT	R	0x0	该位指示接收到的快速命令的 R / W 位。 用户读取该位后，该位将被清除。
[15-12]	Reserved	R	0x0	Reserved
[11]	SDA_STUCK_NOT_RECOVERED	R	0x0	该位指示恢复机制后，未恢复处于低电平的 SDA。
[10]	SLV_HOLD_RX_FIFO_FULL	R	0x0	由于 Rx FIFO 已满并接收到另外的字节，该位指示从设备处于 BUS 保持状态
[9]	SLV_HOLD_TX_FIFO_EMPTY	R	0x0	当 Tx FIFO 为空时，该位指示针对读取请求的总线从动模式。 总线处于保持状态，直到 Tx FIFO 拥有要发送的读取请求数据为止。
[8]	MST_HOLD_RX_FIFO_FULL	R	0x0	该位表示由于 Rx FIFO 已满并且已接收到附加字节而导致总线处于主机模式
[7]	MST_HOLD_TX_FIFO_EMPTY	R	0x0	当主机由于 Tx FIFO 为空而使主机保持总线时，该位指示 BUS 保持，并且先前传输的命令未设置停止位。
[6]	SLV_ACTIVITY	R	0x0	从站 FSM 活动状态。 当从机有限状态机（FSM）不处于 IDLE 状态时，该位置 1。 0：从站 FSM 处于空闲状态，因此 I2C 的从站部分不活动 1：从站 FSM 不在空闲状态，因此 I2C 的从站部分处于活动状态
[5]	MST_ACTIVITY	R	0x0	主 FSM 活动状态。 当主有限状态机（FSM）不在 IDLE 状态时，该位置 1。 0：主 FSM 处于空闲状态，因此 I2C 的“主”部分不处于活动状态 1：主 FSM 不在空闲状态，因此 I2C 的 Master 部分处于活动状态
[4]	RFF	R	0x0	接收 FIFO 完全满。 当接收 FIFO 完全满时，该位置 1。 当接收 FIFO 包含一个或多个空位置时，该位被清除。 0：接收 FIFO 未满 1：接收 FIFO 已满

[3]	RFNE	R	0x0	接收 FIFO 不为空。当接收 FIFO 包含一个或多个条目时，该位置 1。当接收 FIFO 为空时，该位被清除。 0：接收 FIFO 为空 1：接收 FIFO 不为空
[2]	TFE	R	0x1	发送 FIFO 完全为空。 当发送 FIFO 完全为空时，该位置 1。 当它包含一个或多个有效条目时，该位被清除。 该位字段不请求中断。 0：发送 FIFO 不为空 1：发送 FIFO 为空
[1]	TFNF	R	0x1	发送 FIFO 未满。当发送 FIFO 包含一个或多个空位置时置 1，并在 FIFO 满时将其清除。 0：发送 FIFO 已满 1：发送 FIFO 未满
[0]	ACTIVITY	R	0x0	I2C 活动状态。

22.5.2.27 I2C_TXFLR

- **Size：**32bits
- **Offset：**0x74
- **Default：**0x0

Bit	Field	R/W	Default	Description
[31-4]	Reserved	R	0x0	Reserved
[3-0]	TXFLR	R	0x0	Transmit FIFO Level. 包含发送 FIFO 中有效数据条目的数量。

22.5.2.28 I2C_RXFLR

- **Size：**32bits
- **Offset：**0x78
- **Default：**0x0

Bit	Field	R/W	Default	Description
[31-4]	Reserved	R	0x0	Reserved
[3-0]	RXFLR	R	0x0	Receive FIFO Level. 包含接收 FIFO 中有效数据条目的数量。
[15-0]	**I2C_SDA_TX_HOLD**	**R/W**	**0x1**	**当 I2C 用作发送器时，以 I2C_clk 周期为单位设置所需的 SDA 保持时间。**

22.5.2.29 I2C_TX_ABRT_SOURCE

- **Size**：32bits
- **Offset**：0x80
- **Default**：0x0

Bit	Field	R/W	Default	Description
[32-23]	TX_FLUSH_CNT	R	0x0	该字段指示由于 TX_ABRT 中断而刷新的 Tx FIFO 数据命令的数量。每当禁用 I2C 时，它将清除。(Master-Transmitter or Slave-Transmitter)
[22-21]	Reserved	R	0x0	Reserved
[20]	ABRT_DEVICE_WRITE	R	0x0	这是仅主模式的位。 主机正在启动 DEVICE_ID 传输，并且 Tx-FIFO 由写命令组成。
[19]	ABRT_DEVICE_SLVADDR_NOACK	R	0x0	这是仅主模式的位。 主机正在启动 DEVICE_ID 传输，并且任何从机均未确认发送的从机地址。
[18]	ABRT_DEVICE_NOACK	R	0x0	这是仅主模式的位。 主设备启动 DEVICE_ID 传输，任何从设备均不确认发送的设备 ID。
[17]	ABRT_SDA_STUCK_AT_LOW	R	0x0	这是仅主模式的位。 主机检测到 I2C_clks 的 I2C_SDA_STUCK_AT_LOW_TIMEOUT 值使 SDA 处于低电平。
[16]	ABRT_USER_ABRT	R	0x0	这是仅主模式的位。 主机已检测到传输中止（I2C_ENABLE [1]）。
[15]	ABRT_SLVRD_INTX	R	0x0	1：当处理器侧响应从机模式请求将数据发送到远程主机时，用户在 I2C_DATA_CMD 寄存器的 CMD（位 8）中写入 1。
[14]	ABRT_SLV_ARBLOST	R	0x0	1：从站在将数据传输到远程主机时丢失了总线。 I2C_TX_ABRT_SOURCE [12]同时设置。
[13]	ABRT_SLVFLUSH_TXFIFO	R	0x0	1：从设备已接收到读取命令，并且 TX FIFO 中存在一些数据，因此从设备发出 TX_ABRT 中断以刷新 TX FIFO 中的旧数据。
[12]	ARB_LOST	R	0x0	1：主机丢失仲裁，或者如果还设置了 I2C_TX_ABRT_SOURCE [14]，则从机发送方丢失仲裁。
[11]	ABRT_MASTER_DIS	R	0x0	1：用户尝试在禁用主机模式的情况下启动主机操作。
[10]	ABRT_10B_RD_NORSTRT	R	0x0	1：禁止重启（I2C_RESTART_EN 位（I2C_CON [5]）= 0），并且主机在 10 位寻址模式下发送读取命令。
[9]	ABRT_SBYTE_NORSTRT	R	0x0	要清除位 9，必须首先固定 ABRT_SBYTE_NORSTRT 的源。 重新启动必须使能（I2C_CON [5] = 1），必须清除 SPECIAL 位（I2C_TAR [11]），或者必须将 GC_OR_START 位清除（I2C_TAR [10]）。 一旦 ABRT_SBYTE_NORSTRT 的源固定，就可以用与该寄存器中其他位相同的方式清除该位。 如果在尝试清除该位之前 ABRT_SBYTE_NORSTRT 的源未固定，则位 9 清除一个周期，然后重新置位。 1：禁止重新启动（I2C_RESTART_EN 位（I2C_CON [5]）= 0），并且用户正在尝试发送 START 字节。
[8]	ABRT_HS_NORSTRT	R	0x0	1：禁止重启（I2C_RESTART_EN 位（I2C_CON [5]）= 0），并且用户正在尝试使用主机在高速模式下传输数据。

[7]	ABRT_SBYTE_ACKDET	R	0x0	1：主设备发送了一个 START 字节，并且确认了 START 字节（错误行为）。
[6]	ABRT_HS_ACKDET	R	0x0	1：主机处于高速模式，并且高速主机代码被确认（错误行为）。
[5]	ABRT_GCALL_READ	R	0x0	1：主模式下的 I2C 发送了通用调用，但用户将通用调用后的字节编程为要从总线读取（I2C_DATA_CMD [9]设置为 1）。
[4]	ABRT_GCALL_NOACK	R	0x0	1：在主模式下的 I2C 发送了通用呼叫，总线上没有从机确认通用呼叫。
[3]	ABRT_TXDATA_NOACK	R	0x0	1：这是仅主模式的位。 主机已收到该地址的确认，但是当它在该地址之后发送数据字节时，它没有收到远程从机的确认。
[2]	ABRT_10ADDR2_NOACK	R	0x0	1：主机处于 10 位地址模式，任何从机均未确认 10 位地址的第二个地址字节。
[1]	ABRT_10ADDR1_NOACK	R	0x0	1：主机处于 10 位地址模式，任何从机均未确认第一个 10 位地址字节。
[0]	ABRT_7B_ADDR_NOACK	R	0x0	1：主设备处于 7 位寻址模式，并且任何从设备均未确认发送的地址。

22.5.2.30 I2C_SLV_DATA_NACK_ONLY

- **Size**：32bits
- **Offset**：0x84
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	NACK	R/W	0x0	生成 NACK。 仅当 I2C 是从机接收器时，才会发生此 NACK。 如果该寄存器的值设置为 1，则只能在接收到数据字节后生成 NACK；否则，它只能生成 NACK。 因此，数据传输将中止，并且接收到的数据不会被推送到接收缓冲区。 当寄存器的值设置为 0 时，它会根据正常条件生成 NACK / ACK。 1：接收到数据字节后生成 NACK 0：正常生成 NACK / ACK

22.5.2.31 I2C_DMA_CR

- **Size**：32bits
- **Offset**：0x88
- **Default**：0x0

Bit	Field	R/W	Default	Description
[32-2]	Reserved	R	0x0	Reserved
[1]	TDMAE	R/W	0x0	发送 DMA 使能。 该位启用/禁用发送 FIFO DMA 通道。 0：禁用发送 DMA 1：启用发送 DMA
[0]	RDMAE	R/W	0x0	接收 DMA 使能。 该位启用/禁用接收 FIFO DMA 通道。 0：禁用接收 DMA 1：启用接收 DMA

22.5.2.32 I2C_DMA_TDLR

- **Size**：32bits
- **Offset**：0x8C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-4]	Reserved	R	0x0	Reserved
[3-0]	DMATDL	R/W	0x0	传输数据 Level。该位字段控制发送逻辑发出 DMA 请求的 Level。也就是说，当发送 FIFO 中的有效数据条目数等于或小于此字段值且 TDMAE = 1 时，将生成 dma_tx_req 信号。

22.5.2.33 I2C_DMA_RDLR

- **Size：**32bits
- **Offset：**0x90
- **Default：**0x0

Bit	Field	R/W	Default	Description
[31-4]	Reserved	R	0x0	Reserved
[3-0]	DMARDL	R/W	0x0	接收数据 Level。该位字段控制接收逻辑发出 DMA 请求的 Level。也就是说，当接收 FIFO 中的有效数据条目数等于或大于此字段值+ 1 并且 RDMAE = 1 时，将生成 dma_rx_req。例如，当 DMARDL 为 0 时，则当 1 或更大时 dma_rx_req 被声明。 数据条目存在于接收 FIFO 中。

22.5.2.34 I2C_ACK_GENERAL_CALL

- **Size：**32bits
- **Offset：**0x98
- **Default：**0x1

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	ACK_GEN_CALL	R/W	0x1	ACK 一般呼叫。 设置为 1 时，I2C 收到常规呼叫时将以 ACK 响应（通过声明 I2C_data_oe）。 设置为 0 时，I2C 不会生成常规调用中断。

22.5.2.35 I2C_ENABLE_STATUS

- **Size：**32bits
- **Offset：**0x9C
- **Default：**0x0

Bit	Field	R/W	Default	Description
[31-3]	Reserved	R	0x0	Reserved
[2]	SLV_RX_DATA_LOST	R	0x0	从站接收数据丢失。 该位表示由于将 I2C_ENABLE [0]从 1 设置为 0，从 I2C 传输接收到的至少一个数据字节是否导致中止从接收器操作。当读为 1 时，I2C 被视为已在主动参与其中。 I2C 传输中止（具有匹配的地址），并且即使已使用 NACK 响应了数据字节，也已进入 I2C 传输的数据阶段。
[1]	SLV_DISABLED_WHILE_BUSY	R	0x0	忙时（发送，接收）禁用从机。 该位指示由于将 I2C_ENABLE 寄存器的位 0 从 1 设置为 0 而导致潜在的或活动的从动操作已中止。当以下情况时，当 CPU 将 I2C_ENABLE 的位 0 写入 0 时，该位置位：（a）接收 I2C 远程主机的从机发送操作的地址字节； 或，（b）来自远程主机的从机接收器操作的地址和数据字节。 读为 1 时，无论 I2C 地址是否与 I2C（I2C_SAR 寄存器）中设置的从机地址匹配，或者是否在 I2C_ENABLE 的位 0 为零之前完成传输，都将 I2C 视为在 I2C 传输的任何部分强制 NACK。 设置为 0，但没有生效。
[0]	I2C_EN	R	0x0	I2C_en 状态。 该位始终反映在输出端口 I2C_en 上驱动的值。 1：I2C 被视为处于启用状态。 0：I2C 被视为完全不活动。

22.5.2.36 I2C_CLR_RESTART_DET

- **Size：**32bits
- **Offset：**0xA8
- **Default：**0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	R	0x0	Reserved
[0]	CLR_RESTART_DET	R/W	0x0	读取该寄存器以清除 I2C_RAW_INTR_STAT 寄存器的 RESTART_DET 中断（位 12）。

22.5.2.37 I2C_SCL_STUCK_AT_LOW_TIMEOUT

- **Size：**32bits
- **Offset：**0xAC
- **Default：**0xffff_ffff

Bit	Field	R/W	Default	Description

[31-0]	I2C_SCL_STUCK_LOW_TIMEOUT	R/W	0xffff_ffff	如果 I2C 以 I2C_clk 周期为单位检测到 I2C_SCL_STUCK_LOW_TIMEOUT 中的 SCL 处于低电平，则生成中断以指示 SCL 处于低电平。

22.5.2.38 I2C_SCL_STUCK_AT_LOW_TIMEOUT

- **Size**：32bits
- **Offset**：0xB0
- **Default**：0xffff_ffff

Bit	Field	R/W	Default	Description
[31-0]	I2C_SCL_STUCK_LOW_TIMEOUT	R/W	0xffff_ffff	如果 I2C 以 I2C_clk 周期为单位检测到 I2C_SCL_STUCK_LOW_TIMEOUT 中的 SCL 处于低电平，则生成中断以指示 SCL 处于低电平。

22.5.2.39 I2C_CLR_SCL_STUCK_DET

- **Size**：32bits
- **Offset**：0xB0
- **Default**：0xffff_ffff

Bit	Field	R/W	Default	Description
[31-0]	I2C_SCL_STUCK_LOW_TIMEOUT	R/W	0xffff_ffff	如果 I2C 以 I2C_clk 周期为单位检测到 I2C_SCL_STUCK_LOW_TIMEOUT 中的 SCL 处于低电平，则生成中断以指示 SCL 处于低电平。

22.5.2.40 I2C_SMBUS_CLOCK_LOW_SEXT

- **Size**：32bits
- **Offset**：0xBC
- **Default**：0xffff_ffff

Bit	Field	R/W	Default	Description
[31-0]	SMBUS_CLK_LOW_SEXT_TIMEOUT	R/W	0xffff_ffff	该字段用于检测主机模式下从设备在从初始 START 到 STOP 的一条消息中扩展的从设备时钟扩展超时（tLOW：SEXT）。 该寄存器中的值以 I2C_clk 周期为单位。

22.5.2.41 I2C_SMBUS_CLOCK_LOW_MEXT

- **Size**：32bits
- **Offset**：0xC0
- **Default**：0xffff_ffff

Bit	Field	R/W	Default	Description

[31-0]	SMBUS_CLK_LOW_MEXT_TIMEOUT	R/W	0xffff_ffff	此字段用于检测在主机模式下从启动到确认，从确认到确认或从确认到停止定义的主机扩展 SMBus 时钟（SCL）超时。 该寄存器中的值以 I2C_clk 周期为单位。

22.5.2.42 I2C_SMBUS_THIGH_MAX_IDLE_COUNT

- **Size**：32bits
- **Offset**：0xC4
- **Default**：0xffff

Bit	Field	R/W	Default	Description
[31-16]	Reserved	R	0x0	Reserved
[15-0]	SMBUS_THIGH_MAX_BUS_IDLE_CNT	R/W	0xffff	该字段用于设置所需的总线空闲时间段，该时间段是当主机已动态添加到总线并且可能未在 SMBCLK 或 SMBDAT 线上检测到状态转换时使用的时间。 在这种情况下，主机必须等待以确保当前未进行传输。 该寄存器中的值以 I2C_clk 周期为单位。

22.5.2.43 I2C_SMBUS_INTR_STAT

- **Size**：32bits
- **Offset**：0xC8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-11]	Reserved	R	0x0	Reserved
[10]	R_SMBUS_ALERT_DET	R	0x0	指示从站是否将 SMBALERT（I2C_smbalert_in_n）信号驱动为低电平。
[9]	R_SMBUS_SUSPEND_DET	R	0x0	指示主机是否将 SMBSUS（I2C_smbsus_in_n）信号驱动为低电平。
[8]	R_SLV_RX_PEC_NACK	R	0x0	指示从设备是否为从设备的 ARP 命令的 PEC 字节生成 NACK。
[7]	R_ARP_ASSGN_ADDR_CMD_DET	R	0x0	指示是否已收到分配地址 ARP 命令。
[6]	R_ARP_GET_UDID_CMD_DET	R	0x0	指示是否已收到“常规”或定向的“获取 UDID ARP”命令。
[5]	R_ARP_RST_CMD_DET	R	0x0	指示是否已收到常规或定向重置 ARP 命令。
[4]	R_ARP_PREPARE_CMD_DET	R	0x0	指示是否已收到“准备 ARP”命令。
[3]	R_HOST_NOTIFY_MST_DET	R	0x0	指示是否已收到主机通知命令。
[2]	R_QUICK_CMD_DET	R	0x0	指示无论 I2C 是在从属模式还是在主控模式下运行，SMBus 接口上是否都已收到快速命令。 仅当 I2C_SMBUS = 1 设置为 1 时，才启用此位。

[1]	R_MST_CLOCK_EXTND_TIMEOUT	R	0x0	指示从 START 到 STOP 的主设备事务（START-to-ACK，ACK-to-ACK 或 ACK-to-STOP）在消息的每个字节中是否超过 I2C_SMBUS_CLOCK_LOW_MEXT 时间。
[0]	R_SLV_CLOCK_EXTND_TIMEOUT	R	0x0	指示从设备（即从 START 到 STOP）的事务是否超过 I2C_SMBUS_CLOCK_LOW_SEXT 时间。

22.5.2.44 I2C_SMBUS_INTR_MASK

- **Size**：32bits
- **Offset**：0xCC
- **Default**：0x7ff

Bit	Field	R/W	Default	Description
[31-11]	Reserved	R	0x0	Reserved
[10]	M_SMBUS_ALERT_DET	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_SMBUS_ALERT_DET 中断位。写 0 表示屏蔽。
[9]	M_SMBUS_SUSPEND_DET	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_SMBUS_SUSPEND_DET 中断位。写 0 表示屏蔽。
[8]	M_SLV_RX_PEC_NACK	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_SLV_RX_PEC_NACK 中断位。写 0 表示屏蔽。
[7]	M_ARP_ASSGN_ADDR_CMD_DET	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_ARP_ASSGN_ADDR_CMD_DET 中断位。写 0 表示屏蔽。
[6]	M_ARP_GET_UDID_CMD_DET	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_ARP_GET_UDID_CMD_DET 中断位。写 0 表示屏蔽。
[5]	M_ARP_RST_CMD_DET	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_ARP_RST_CMD_DET 中断位。写 0 表示屏蔽。
[4]	M_ARP_PREPARE_CMD_DET	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_ARP_PREPARE_CMD_DET 中断位。写 0 表示屏蔽。
[3]	M_HOST_NOTIFY_MST_DET	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_HOST_NOTIFY_MST_DET 中断位。写 0 表示屏蔽。
[2]	M_QUICK_CMD_DET	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_QUICK_CMD_DET 中断位。写 0 表示屏蔽。
[1]	M_MST_CLOCK_EXTND_TIMEOUT	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_MST_CLOCK_EXTND_TIMEOUT 中断位。写 0 表示屏蔽。
[0]	M_SLV_CLOCK_EXTND_TIMEOUT	R/W	0x1	该位屏蔽了 I2C_SMBUS_INTR_STAT 寄存器中的 R_SLV_CLOCK_EXTND_TIMEOUT 中断位。写 0 表示屏蔽。

22.5.2.45 I2C_SMBUS_INTR_STAT

- **Size**：32bits
- **Offset**：0xD0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-11]	Reserved	R	0x0	Reserved
[10]	SMBUS_ALERT_DET	R	0x0	指示从站是否将 SMBALERT（I2C_smbalert_in_n）信号驱动为低电平。
[9]	SMBUS_SUSPEND_DET	R	0x0	指示主机是否将 SMBSUS（I2C_smbsus_in_n）信号驱动为低电平。
[8]	SLV_RX_PEC_NACK	R	0x0	指示从设备是否为从设备的 ARP 命令的 PEC 字节生成 NACK。
[7]	ARP_ASSGN_ADDR_CMD_DET	R	0x0	指示是否已收到分配地址 ARP 命令。
[6]	ARP_GET_UDID_CMD_DET	R	0x0	指示是否已收到“常规”或定向的“获取 UDID ARP”命令。
[5]	ARP_RST_CMD_DET	R	0x0	指示是否已收到常规或定向重置 ARP 命令。
[4]	ARP_PREPARE_CMD_DET	R	0x0	指示是否已收到“准备 ARP”命令。
[3]	HOST_NOTIFY_MST_DET	R	0x0	指示是否已收到主机通知命令。
[2]	QUICK_CMD_DET	R	0x0	指示无论 I2C 是在从属模式还是在主控模式下运行，SMBus 接口上是否都已收到快速命令。 仅当 I2C_SMBUS = 1 设置为 1 时，才启用此位。
[1]	MST_CLOCK_EXTND_TIMEOUT	R	0x0	指示从 START 到 STOP 的主设备事务（START-to-ACK，ACK-to-ACK 或 ACK-to-STOP）在消息的每个字节中是否超过 I2C_SMBUS_CLOCK_LOW_MEXT 时间。
[0]	SLV_CLOCK_EXTND_TIMEOUT	R	0x0	指示从设备（即从 START 到 STOP）的事务是否超过 I2C_SMBUS_CLOCK_LOW_SEXT 时间。

22.5.2.46 I2C_CLR_SMBUS_INTR

- **Size**：32bits
- **Offset**：0xD4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-11]	Reserved	R	0x0	Reserved
[10]	CLR_SMBUS_ALERT_DET	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 SMBUS_ALERT_DET 中断（位 10）。
[9]	CLR_SMBUS_SUSPEND_DET	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 SMBUS_SUSPEND_DET 中断（位 9）。
[8]	CLR_SLV_RX_PEC_NACK	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 SLV_RX_PEC_NACK 中断（位 8）。
[7]	CLR_ARP_ASSGN_ADDR_CMD_DET	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 ARP_ASSGN_ADDR_CMD_DET 中断（位 7）。
[6]	CLR_ARP_GET_UDID_CMD_DET	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 ARP_GET_UDID_CMD_DET 中断（位 6）。
[5]	CLR_ARP_RST_CMD_DET	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 ARP_RST_CMD_DET 中断（位 5）。
[4]	CLR_ARP_PREPARE_CMD_DET	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 ARP_PREPARE_CMD_DET 中断（位 4）。
[3]	CLR_HOST_NOTIFY_MST_DET	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 HOST_NOTIFY_MST_DET 中断（位 3）。
[2]	CLR_QUICK_CMD_DET	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 QUICK_CMD_DET 中断（位 2）。
[1]	CLR_MST_CLOCK_EXTND_TIMEOUT	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 MST_CLOCK_EXTND_TIMEOUT 中断（位 1）。
[0]	CLR_SLV_CLOCK_EXTND_TIMEOUT	W	0x0	写入该寄存器以清除 I2C_SMBUS_RAW_INTR_STAT 寄存器的 SLV_CLOCK_EXTND_TIMEOUT 中断（位 0）。

22.5.2.47 I2C_OPTIONAL_SAR

- **Size**：32bits
- **Offset**：0xD8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-7]	Reserved	R	0x0	Reserved
[6-0]	I2C_OPTIONAL_SAR	R/W	0x0	在 SMBus 模式下作为从设备运行时，I2C 的可选从设备地址。

22.5.2.48 I2C_SMBUS_UDID_LSB

- **Size**：32bits
- **Offset**：0xDC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-0]	I2C_SMBUS_ARP_UDID_LSB	R/W	0x0	该字段用于存储地址解析协议中使用的从属唯一设备标识符的 LSB 32 位值。

23 通用异步收发器（UART）

23.1 简介

UART，全称 Universal asynchronous receiver transmitter，中文翻译“通用异步串行收发器”，它是一种通用串行数据总线。在嵌入式设计中，UART 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。TAE32F5300 的 UART 支持异步单向通信和全双工单线通信；而且，它还支持多处理器通信。UART 通过小数波特率发生器提供了多种波特率。

23.2 结构框图

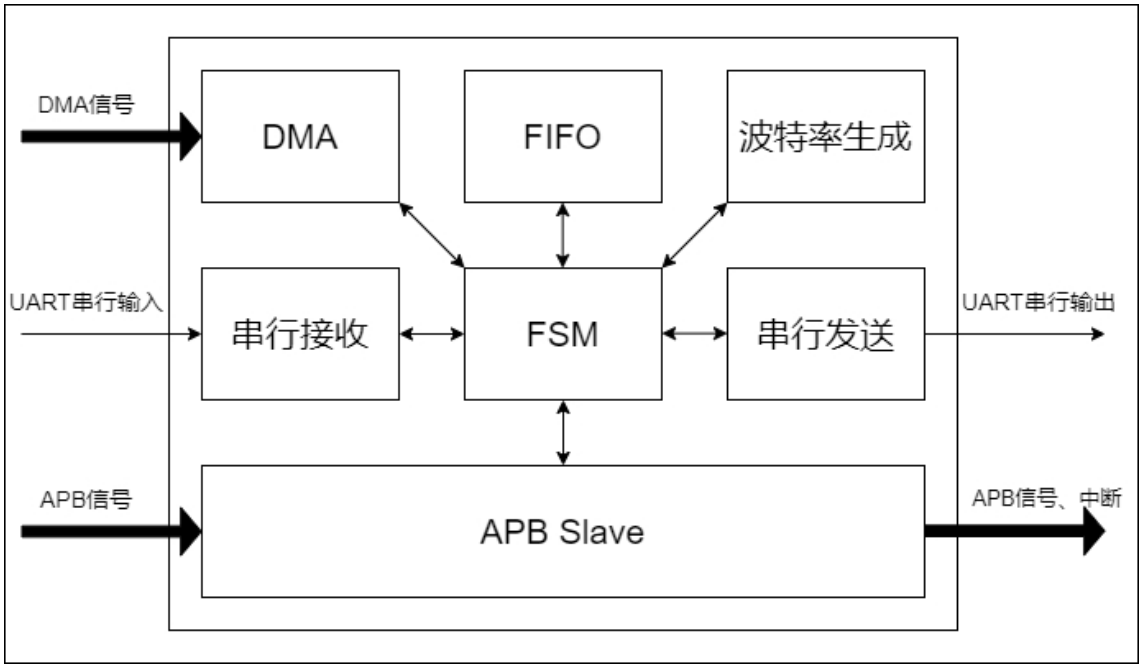


图 23-1 UART 结构框图

23.3 主要特性

- 支持 9 位串行数据传输
- 错误启动位检测
- 支持可编程小数波特率
- 支持多点 RS485 接口
- 可选的奇偶校验位，停止位数
- 16 位的 FIFO 深度
- 支持中断方式
- 由下列公式可计算可编程串行数据波特率：波特率 =(串行时钟频率)/(16×除数)

23.4 功能描述

23.4.1 UART 协议（RS232）

由于 UART 与设备间的通信是异步的，在串行数据中要加入两个数据位（start 和 stop）来标志通信的开始和结束。通过这些数据位可以让两个设备间进行同步。

串行数据的格式如图 23-2 所示：

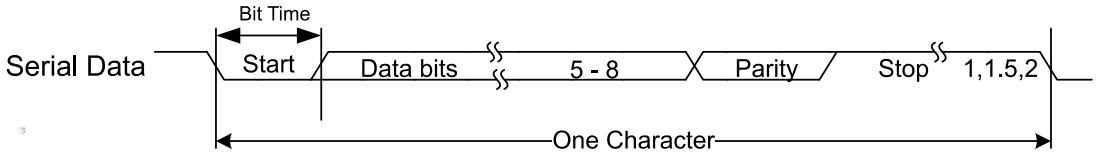


图 23-2　串行数据格式

UART 传输的字符中可以加入校验位，在最后一位数据位之后，位于 stop 位之前。主要用来对 UART 接收的数据进行简单错误检查。UART 的寄存器（LCR）用于控制串行字符特征。数据位在 start 位之后发出，并以 LSB 位开始。紧接着是可选的校验位，stop 位（可以为 1/1.5/2 位）。UART 传输中，每一位的传输时间都是一样的，1.5 bit stop 位中的半 stop 位除外。每一位的传输时间称为位周期，一个位周期等于 16 个波特时钟 cycle。

为了保证线上传输的稳定性，接收端检测到 start 位后，在位周期的中点处开始采集输入的数据。因为每个数据位传输时间的波特时钟 cycle 数量是已知的，不难计算数据采集的中点。即 start 位中点后的每 16 个波特时钟 cycle。除了对输入进行去抖动外，这种采样方式也可以避免检测到错误的 start 位。短暂的不稳定信号可以通过去抖动滤掉，线上不会产生传输。如果一个不稳定信号时间长到可以躲过被去抖动滤掉，那么 start 位只有在经过半个位周期再次采样为低时才算检测有效。

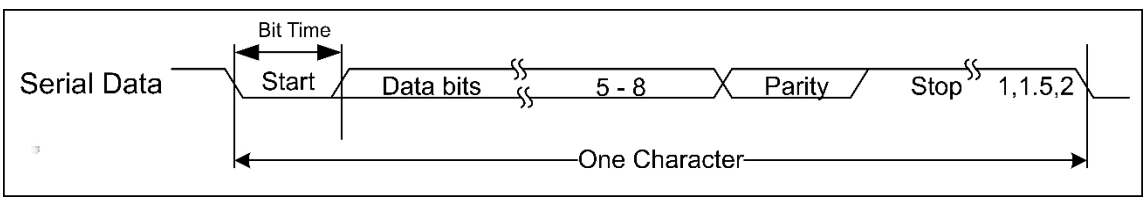


图 23-3　接收数据采样点

作为 16550 标准的一部分，可选的波特时钟参考输出信号可以向接收设备提供所需的时序信息。UART 的波特率是由寄存器（DLH 和 DLL）控制的。

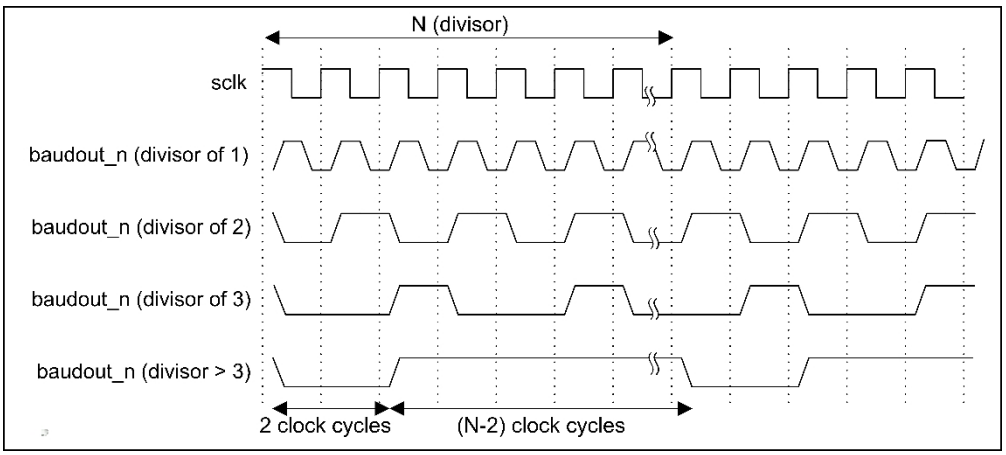


图 23-4　波特率参考时序图

23.4.2　RS485 协议

RS485 协议也可以像 RS232 协议一样支持双线配置进行串行通信。RS485 协议和 RS232 协议的区别在于 RS485 协议中进行传输时用了具有指导传输方向的信号线。这样可以去除大部分噪声，提高信号强度。在这种基础上，RS485 协议可以用于比 RS232 距离更长的场景。

RS485 协议可以使用 RS485 接口进行数据传输。在使能 RS485 接口支持后，会相应加入驱动使能信号（DE）和接收使能信号（RE）。DE 和 RE 信号是由硬件生成的，TAE32F5300 中无 RE 线，需要用户对 DE 进行取反，获得 RE 信号，DE 有效/无效时间是可配置的。

RS485 的配置步骤：

1. 通过接收控制寄存器（TCR）的 bit 0 来使能或取消 RS485 模式；
2. TCR 的 bit1 用于选择 DE 信号的极性；
3. TCR 的 bit[4:3]选择 RS485 模式的传输类型；

23.4.3 9bit 数据传输

在发送和接收模式中，UART 都可以配置成 9bit 数据传输。字符中第 9 个数据位，位于第 8 个数据位和校验位之间。图 23-5 中，D8 表示第 9 个 bit。

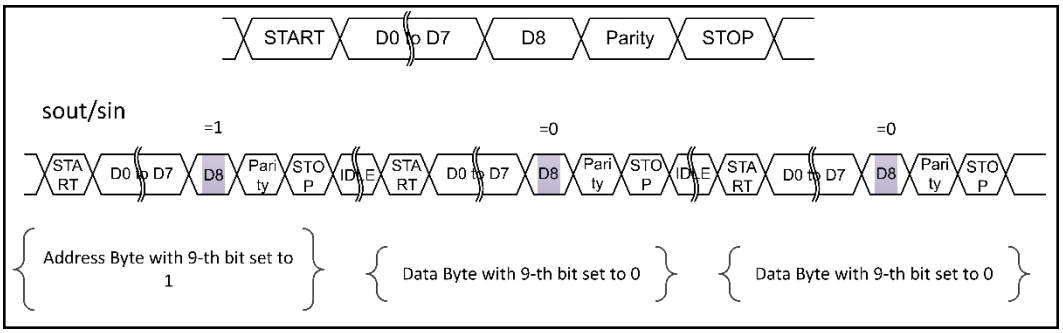


图 23-5 9bit 参数

通过使能 9bit 数据传输模式，UART 可以用于一个 master 和多个 slave 组成的多点系统。当 master 和多个 slave 中的一个进行通信，需要向 slave 发送一堆数据时，master 首先需要发送一个地址字节表明目标 slave。地址字节和数据字节的不同在于：地址字节的第 9 个 bit 为 1。地址字节发出后，所有 slave 都会对比地址字节和自己的地址。只有一个 slave（和地址字节匹配）会被使能来接收 master 发送的数据，之后 master 开始发送数据。没有匹配上的 slaves 会无视发送过来的数据，直到接收到一个新的地址字节。

配置 UART 9bit 数据传输的步骤如下：

1. LCR_EXT[0]是使能或不使能 9bit 数据传输；
2. LCR_EXT[1]是用于选择在接收情况下的地址匹配模式；
3. LCR_EXT[2]是用于使能发送情况下的地址发送；
4. LCR_EXT[3]选择地址发送模式；
5. TAR 和 RAR 寄存器分别用于发送地址和匹配接收到的地址；
6. 9 位的 THR，RBR，STHR 和 SRBR 寄存器是用于执行 9bit 模式下的数据传输；
7. LSR[8]用于表示地址接收中断。

UART 支持两种发送模式：

- 发送模式 0（LCR_EXT[0]为 0）
- 发送模式 1（LCR_EXT[0]为 1）

发送模式 0

在发送模式 0 中，地址是由发送地址寄存器（TAR）配置，数据写入发送寄存器（THR）中。在这个模式下，THR 的第 9 个 bit 是不能用的。

图 23-6 表示了在 SEND_ADDR（LCR_EXT[2]），Halt Tx 和 TX FIFO/THR 为空情况下，地址和数据的传输。

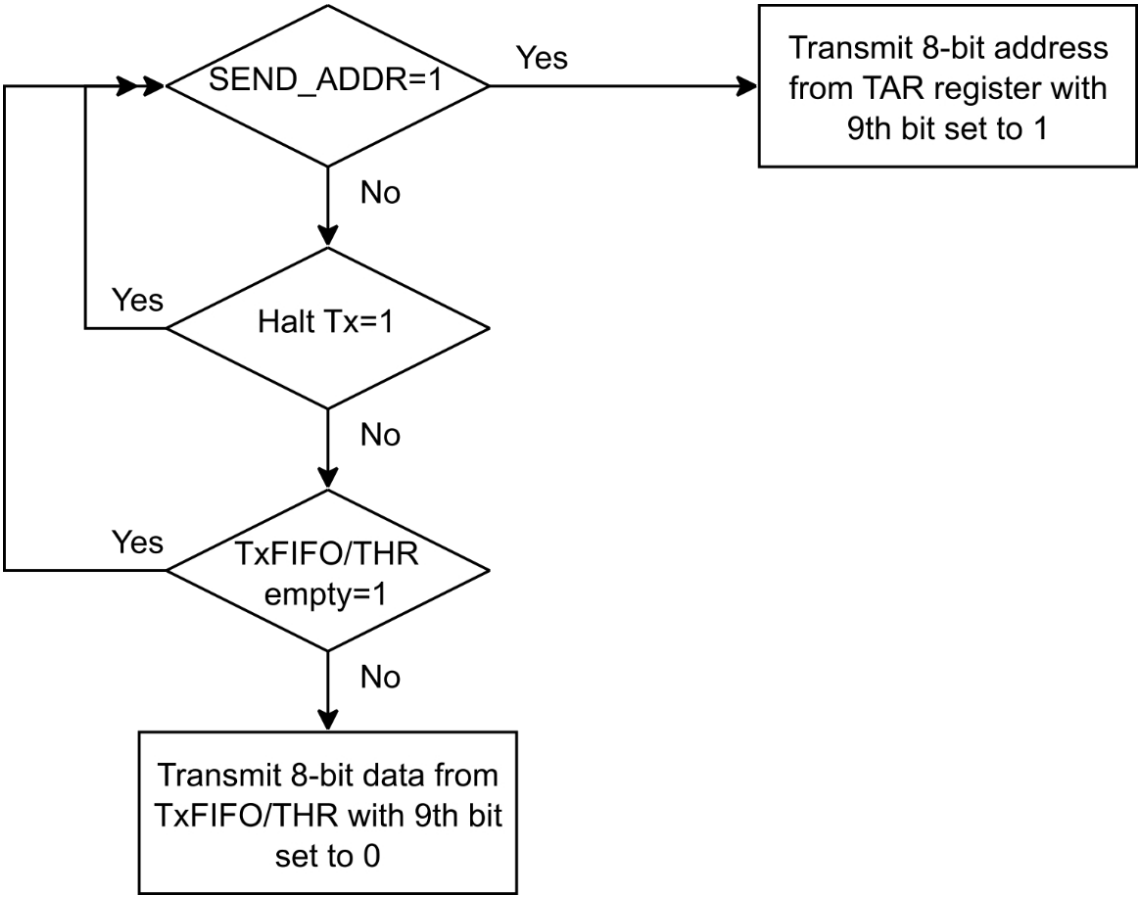


图 23-6 自动地址发送流程图

发送模式 1

在发送模式 1 中，THR 寄存器都是 9 位的，地址和数据都是通过 THR 寄存器来配置的。UART 不区分地址和数据，两者都是从 TX FIFO 中取出来的。在改模式下，SEND_ADDR（LCR_EXT[2]）和 TAR 寄存器都是不可用的。因此，软件上必须配置第 9 个 bit 为 0/1 来决定发送的是数据还是地址。

UART 支持两种接收模式：

- 硬件地址匹配接收模式（ADDR_MATCH（LCR_EXT[1]）为 1）
- 软件地址匹配接收模式（ADDR_MATCH（LCR_EXT[1]）为 0）

硬件地址匹配接收模式

在硬件地址匹配接收模式中，如果接收到的字符的第 9 个数据位为 1，UART 通过接收地址寄存器（RAR）中配置的地址来匹配接收到字符。如果接收到的地址与 RAR 寄存器中配置的地址匹配上，那接下来的数据字节将会放进 RX FIFO 中。如果地址不匹配，UART 会丢掉后续的数据字节，直到接收到一个匹配的地址。在地址匹配下，UART 的数据传输流程图 23-7 所示。

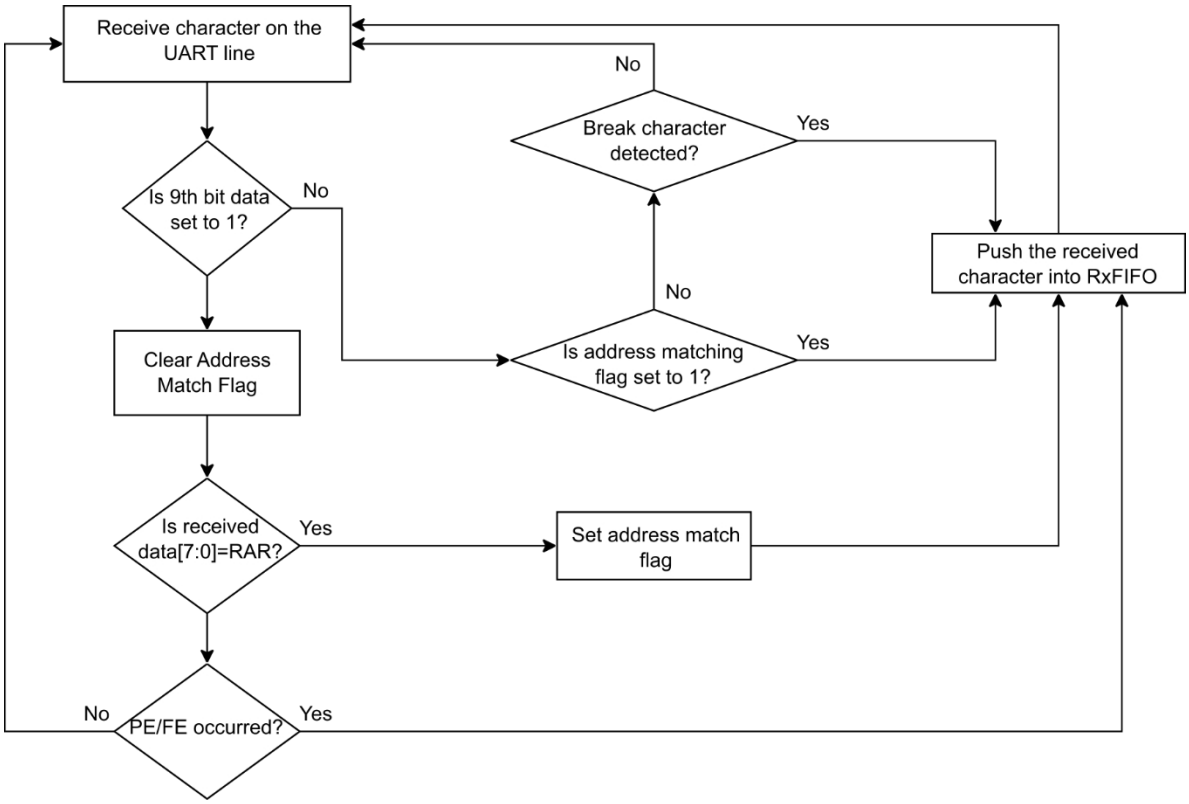


图 23-7 硬件地址匹配接收模式

软件地址匹配接收模式

在这种模式下，UART 不会将接收到的地址和 RAR 寄存器进行匹配。UART 一直接收 9bit 的数据，并将数据放进 RX FIFO（FIFO 模式下）或 RBR 寄存器中（非 FIFO 模式下）。每当接收到地址字节时，用户必须比较地址，并且通过线状态寄存器的 ADDR_RCVD 指示。当地址不匹配时，用户可以通过 FIFO 控制寄存器（FCR）的 RCVR FIFO Reset 位来刷新或重置 RX FIFO。

23.4.4 小数波特率支持

UART 支持小数波特率：可以让用户配置波特率分频值的小数部分由此产生波特率的小数部分，减少频率误差。UART 需要通过软件配置处理让波特率在 2%的频率误差内。UART 的波特率是由 pclk 以及分频锁存寄存器（DLH 和 DLL）共同控制的。波特率由以下因素决定：

- 串行时钟（pclk）频率
- 预期的波特率
- 波特率分频值（DLH 和 DLL）
- 可接受的波特率误差，%ERROR

波特率的计算公式如下：

$$波特率=\frac{串行时钟频率}{(16\times 分频值)}$$

其中，分频值是 DLH 和 DLL 配置的值，串行时钟频率是 UART 中 pclk 的频率。

配置 UART 小数波特率的步骤：

1. 设置 DLF_SIZE 参数——选择保存分频值小数部分寄存器的位宽；
2. 通过分频锁存小数寄存器（DLF）配置分频小数部分的值。

可配置的小数波特率分频值让波特时钟较传统的整数分频有更好的分辨率。小数波特率分频中的小数和整数部分都是可配置的。由此波特率分频值的更准确的公式为：

$$波特率=\frac{串行时钟频率}{(16\times 分频值)}=BRD_I + BRD_F$$

其中，BRD_I 表示分频值的整数部分，BRD_F 表示分频值的小数部分。

23.4.5 FIFO

UART 中可以加入 FIFO 用来缓存发送和接收的数据。UART 中使用的 FIFO 是具有内部 D 触发器结构的 RAM，接收和发送的 FIFO 的深度都为 16 。

有一个特殊的 FIFO 测试访问模式可以用测试目的：

- 接收 FIFO 由 master 来写
- 发送 FIFO 由 master 来读

23.4.6 DMA

UART 具有内置 DMA 功能，它具有与 DMA 控制器的握手接口，以请求和控制传输。 APB 总线用于执行与 DMA 之间的数据传输。

UART 有两个 DMA 通路：一个用于发送数据，一个用于接收数据。

23.5 寄存器描述

23.5.1 寄存器列表

Name	Offset	Width	Description
UART_RBR	0x00	32bit	UART Receive Buffer Register
UART_THR	0x00	32bit	UART Transmit Holding Register
UART_DLL	0x00	32bit	UART Divisor Latch (Low)
UART_DLH	0x04	32bit	UART Divisor Latch (High)
UART_IER	0x04	32bit	UART Interrupt Enable Register
UART_IIR	0x08	32bit	UART Interrupt Identification Register
UART_FCR	0x08	32bit	UART FIFO Control Register
UART_LCR	0x0C	32bit	UART Line Control Register
UART_LSR	0x14	32bit	UART Line Status Register
UART_USR	0x7C	32bit	UART Status Register
UART_TFL	0x80	32bit	UART Transmit FIFO Level
UART_RFL	0x84	32bit	UART Receive FIFO Level
UART_HTX	0xA4	32bit	UART Halt TX
UART_TCR	0xAC	32bit	UART Transceiver Control Register
UART_DE_EN	0xB0	32bit	UART Driver Output Enable Register
UART_RE_EN	0xB4	32bit	UART Receiver Output Enable Register
UART_DET	0xB8	32bit	UART Driver Output Enable Timing Register
UART_TAT	0xBC	32bit	UART TurnAround Timing Register
UART_DLF	0xC0	32bit	UART Divisor Latch Fractional Value
UART_RAR	0xC4	32bit	UART Receive Address Register
UART_TAR	0xC8	32bit	UART Transmit Address Register
UART_LCR_EXT	0xCC	32bit	UART Line Extended Control Register

23.5.2 寄存器详细描述

23.5.2.1 UART Receive Buff Register (UART_RBR)

- **Name**：Receive Buff Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0
- **Condition**：只有当 DLAB 位（LCR[7]）被清除时，才能访问该寄存器。

Bit	Field	R/W	Default	Description
[31-9]	Reserved	Reserved	Reserved	Reserved
[8]	MSB_9th bit	R	0	在 UART 模式下，串行输入端口上的 MSB 第 9 位接收到的数据字节。
[7-0]	LSB_8bits	R	0	在 UART 模式下通过串行输入端口接收的数据字节，或在红外模式下通过串行红外输入接收的数据字节。仅当线路状态寄存器（LSR）中的数据就绪（DR）位置 1 时，此寄存器中的数据才有效。 如果使能 FIFO（FCR [0]设置为 1），则该寄存器访问接收 FIFO 的头部。 如果接收 FIFO 已满，并且在下一个数据字符到达之前未读取该寄存器，则将保留 FIFO 中已经存在的数据，但是所有输入数据都会丢失，并且会发生溢出错误。

23.5.2.2 UART Transmit Holding Register (UART_THR)

- **Name**：Transmit Holding Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0
- **Condition**：只有当 DLAB 位（LCR[7]）被清除时，才能访问该寄存器。

Bit	Field	R/W	Default	Description
[31-9]	Reserved	Reserved	Reserved	Reserved
[8]	MSB_9th bit	W	0	在 UART 模式下，要通过串行输出端口在 MSB 第 9 位上传输的数据。
[7-0]	LSB_8bits	W	0	在 UART 模式下要在串行输出端口上传输的数据，或者在红外模式下要在串行红外输出上传输的数据。仅当将 THR 空（THRE）位（LSR [5]）置 1 时，才应将数据写入 THR。 如果启用了 FIFO（FCR [0] = 1）并且设置了 THRE，则可能在 FIFO 满之前将 x 个字符的数据写入 THR。 x 数（默认值= 16）由您在配置期间设置的 FIFO 深度值确定。当 FIFO 已满时，任何写数据的尝试都会导致写数据丢失。

23.5.2.3 UART Divisor Latch Low (UART_DLL)

- **Name**：Divisor Latch Low
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0
- **Condition**：当 DLAB 位（LCR[7]）被设置时，才能访问该寄存器。

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-0]	DLL	R/W	0	16 位读/写除数锁存寄存器的低 8 位，其中包含 UART 的波特率除数。 输出波特率等于串行时钟频率（pclk）除以波特率除数值的十六倍，如下所示：波特率=（串行时钟频率）/（16 *除数）。 注意：将除数锁存寄存器（DLL 和 DLH）设置为 0 时，波特率时钟被禁用，并且不会发生串行通信。 同样，一旦设置了 DLL，在发送或接收数据之前，应至少允许通过最慢的 uart 总线时钟的 8 个时钟周期。

23.5.2.4 UART Divisor Latch High (UART_DLH)

- **Name**：Divisor Latch High
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0
- **Condition**：当 DLAB 位（LCR[7]）被设置时，才能访问该寄存器。

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	DLH	R/W	0	16 位读/写除数锁存寄存器的高 8 位，其中包含 UART 的波特率除数。 输出波特率等于串行时钟频率（pclk）除以波特率除数值的十六倍，如下所示：波特率=（串行时钟频率）/（16 *除数）。 注意：将除数锁存寄存器（DLL 和 DLH）设置为 0 时，波特率时钟被禁用，并且不会发生串行通信。 同样，一旦设置了 DLH，在发送或接收数据之前，应至少允许通过最慢的 uart 总线时钟的 8 个时钟周期。

23.5.2.5 UART Interrupt Enable Register (UART_IER)

- **Name**：Interrupt Enable Register
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0
- **Condition**：只有当 DLAB 位（LCR[7]）被清除时，才能访问该寄存器。

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7]	PTIME	R/W	0	可编程 THRE 中断模式使能。这用于启用/禁用 THRE 中断的产生。 0：disabled 1：enabled
[6-4]	Reserved	Reserved	Reserved	Reserved
[3]	EDSSI	R/W	0	启用调制解调器状态中断。这用于启用/禁用调制解调器状态中断的生成。这是第四高优先级的中断。 0：disabled 1：enabled
[2]	ELSI	R/W	0	启用接收器线路状态中断。这用于启用/禁用接收器线路状态中断的生成。这是最高优先级的中断 0：disabled 1：enabled
[1]	ETBEI	R/W	0	使能发送保持寄存器清空中断。这用于启用/禁用发送器保持寄存器空中断的生成。这是第三高优先级的中断。 0：disabled 1：enabled
[0]	ERBFI	R/W	0	启用接收数据可用中断。这用于启用/禁用接收数据可用中断和字符超时中断的生成。这些是第二高优先级的中断 0：disabled 1：enabled

23.5.2.6 UART Interrupt Identity Register (UART_IIR)

- **Name**：Interrupt Identity Register
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0
- **Condition**：当 DLAB 位（LCR[7]）被设置时，才能访问该寄存器。

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-6]	FIFOSE	R	0	FIFOs Enabled。用于指示 FIFO 是否已启用或禁用 00：禁用 11：启用
[5-4]	Reserved	Reserved	Reserved	Reserved
[3-0]	IID	R	1	Interrupt ID。这表示最高优先级的挂起中断，可以是以下类型： 0000：调制解调器状态 0001：无中断挂起 0010：THR 空载 0100：接收数据可用 0110：接收线路状态 0111：忙检测 1100：字符超时: 注意：位 3 表示只有当 FIFO 被启用并用于区分字符超时条件中断时，才会发生中断。

23.5.2.7 UART FIFO Control Register (UART_FCR)

- **Name**：FIFO Control Register
- **Size**：32bitsS
- **Offset**：0x08
- **Default**：0x0
- **Condition**：当 DLAB 位（LCR[7]）被清除时，才能访问该寄存器。

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-6]	RT	W	0	RCVR Trigger。这用于选择接收器 FIFO 中的触发电平，在该电平上生成接收到的数据可用中断。 00：FIFO 中 1 个字符 01：FIFO¼已满 10：FIFO½已满 11：比 FIFO 满时少 2 个数
[5-4]	TET	W	0	TX Empty Trigger。这用于选择当模式激活时产生中断的空阈值电平。 00：FIFO 为空 01：FIFO 中的 2 个字符 10：FIFO¼已满 11：FIFO½已满
[3]	MAAM	W	0	DMA Mode 0：dma mode 0 1：dma mode 1
[2]	XFIFOR	W	0	XMIT FIFO Reset。 这将复位发送 FIFO 的控制部分，并将 FIFO 视为空。注意，该位是“自清除”的。无需清除此位。
[1]	RFIFOR	W	0	RCVR FIFO Reset。这将重置接收 FIFO 的控制部分，并将 FIFO 视为空。注意，该位是“自清除”的。无需清除此位。
[0]	FIFOE	W	0	FIFO Enable。这将启用/禁用发送（XMIT）和接收（RCVR）FIFO。每当此位的值改变时，FIFO 的 XMIT 和 RCVR 控制器部分都会复位。

23.5.2.8 UART Line Control Register (UART_LCR)

- **Name**：Line Control Register
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7]	DLAB	R/W	0	除数锁存器访问位。该位用于使能除数锁存寄存器 DLL 和 DLH 的读写，以设置 UART 的波特率。初始波特率设置后，必须清除该位以访问其他寄存器。
[6]	BC	R/W	0	中断控制位。用于将中断条件发送到接收设备。 如果设置为 1，则串行输出被强制为间隔（逻辑 0）状态。
[5]	Stick_ Parity	R/W	0	Stick_Parity。该位用于强制奇偶校验值。 当 PEN，EPS 和 Stick Parity 设置为 1 时，奇偶校验位被发送并检查为逻辑 0。 如果 PEN 和 Stick Parity 被设置为 1 且 EPS 为逻辑 0，则发送奇偶校验位并作为逻辑检查 1。 如果此位设置为 0，则禁用强制奇偶校验。
[4]	EPS	R/W	0	偶校验选择。 启用奇偶校验（PEN 设置为 1）时，该选项用于在偶校验和奇校验之间进行选择。 如果设置为 1，则发送或检查偶数个逻辑 1。 如果设置为 0，则发送或检查奇数个逻辑 1。
[3]	PEN	R/W	0	奇偶校验启用。该位用于分别启用和禁用发送和接收的串行字符中的奇偶校验生成和检测。 0：禁用奇偶校验 1：启用奇偶校验
[2]	STOP	R/W	0	停止位数。这用于选择外设发送和接收的每个字符的停止位数。 如果设置为 0，则在串行数据中发送一个停止位。 如果设置为 1 并且数据位设置为 5（LCR [1：0]设置为 00），则发送一个半停止位。否则，将发送两个停止位。 请注意，无论选择多少停止位，接收器都只会检查第一个停止位。 0：1 个停止位 1：DLS（LCR [1：0]）为 00 时，1–1.5 个停止位，否则为 2 个停止位
[1-0]	DLS	R/W	0	数据长度选择。当 LCR_EXT 中的 DLS_E 设置为 0 时，该寄存器用于选择外设发送和接收的每个字符的数据位数。可以选择的位数如下： 00：5 位 01：6 位 10：7 位 11：8 位

23.5.2.9 UART Line Status Register (UART_LSR)

- **Name**：Line Status Register
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-9]	Reserved	Reserved	Reserved	Reserved
[8]	ADDR_RCVD	R/W	0	地址接收位。 如果启用了 9 位数据模式（LCR_EXT [0] = 1），则该位用于指示接收数据的第 9 位设置为 1。该位还可以用于指示输入字符是否为地址或数据。 1：表示字符是地址。 0：表示字符是数据。 在 FIFO 模式下，由于第 9 位与接收到的字符相关联，因此当第 9 位设置为 1 的字符位于 FIFO 列表的顶部时会显示出来。 读取 LSR 将清除第 9 位。
[7]	RFE	R	0	接收器 FIFO 错误位。当使能了 FIFO（FCR [0]设置为 1）时，此位才相关。这用于指示FIFO中是否至少存在一个奇偶校验错误，成帧错误或中断指示。 0：RX FIFO 中没有错误 1：RX FIFO 中的错误 当读取 LSR 且有错误的字符位于接收器 FIFO 的顶部且 FIFO 中没有后续错误时，该位将被清除。
[6]	TEMT	R	1	发送器空位。如果使能 FIFO（FCR [0]设置为 1），则只要发送移位寄存器和 FIFO 都为空，就设置该位。
[5]	THRE	R	1	THR 为空位。 如果禁用 THRE 模式（IER [7]设置为 0），该位表示 THR 或 TX FIFO 为空。每当数据从 THR 或 TX FIFO 传输到发送移位寄存器并且没有新数据写入 THR 或 TX FIFO 时，该位置 1。 如果启用了 THRE 中断，这也会导致 THRE 中断发生。 如果两种模式都处于活动状态（IER [7]设置为 1，FCR [0]设置为 1），则功能切换为指示发送器FIFO已满，并且不再控制 THRE 中断，然后由 FCR [5:4]阈值设置控制。
[4]	BI	R	0	中断位中断。这用于指示检测到串行输入数据上的中断序列。 如果处于 UART 模式，则只要串行输入 sin 保持逻辑“0”状态的时间长于（开始时间+数据位+奇偶校验+停止位）的总和，就将其置位。串行输入的中断条件导致 UART 接收只有一个字符（由全 0 组成）。 在FIFO模式下，与中断条件相关的字符通过FIFO传送，并在字符位于 FIFO 顶部时显示出来。读取 LSR 将清除 BI 位

[3]	FE	R	0	帧错误位。这用于指示接收器中发生帧错误。 当接收器未在接收到的数据中检测到有效的 STOP 位时，就会发生成帧错误。 0：没有成帧错误 1：成帧错误 读取 LSR 会清除 FE 位。
[2]	PE	R	0	奇偶校验错误位。 如果奇偶校验使能（PEN）位（LCR [3]）被置 1，则用于指示接收器中发生奇偶校验错误。 在 FIFO 模式下，由于奇偶校验错误与接收到的字符相关联，因此当具有奇偶校验错误的字符到达 FIFO 的顶部时会显示出来。 应当注意，如果发生中断位中断，则可以设置奇偶校验错误（PE）位（LSR [2]），如中断位中断（BI）位（LSR [4]）所示。在这种情况下，如果启用了奇偶校验生成和检测（LCR [3] = 1）并且奇偶校验设置为奇数（LCR [4] = 0），则奇偶校验错误位将置 1。 0：无奇偶校验错误 1：奇偶校验错误 读取 LSR 将清除 PE 位。
[1]	OE	R	0	溢出错误位。这用于指示发生超限错误。如果在读取先前的数据之前接收到新的数据字符，则会发生这种情况： 在 FIFO 模式下，当 FIFO 已满并且新字符到达接收器时，会发生溢出错误。FIFO 中的数据将保留，而接收移位寄存器中的数据将丢失。 0：无溢出错误 1：溢出错误 读取 LSR 会清除 OE 位。
[0]	DR	R	0	数据就绪位。这用于指示接收器在 RBR 或接收器 FIFO 中至少包含一个字符。 0：无数据准备 1：数据准备就绪 在 FIFO 模式下接收方 FIFO 为空时，该位清零。

23.5.2.10 UART Status Register (UART_USR)

- **Name**：UART Status Register
- **Size**：32bits
- **Offset**：0x7C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-5]	Reserved	Reserved	Reserved	Reserved
[4]	RFF	R	0	接收 FIFO 已满。这用于指示接收 FIFO 完全满。 0：接收 FIFO 未满 1：接收 FIFO 已满 当 RX FIFO 不再满时，该位被清除
[3]	RFNE	R	0	接收 FIFO 不为空。这用于指示接收 FIFO 包含一个或多个条目。 0：接收 FIFO 为空 1：接收 FIFO 不为空 当 RX FIFO 为空时，该位清零。
[2]	TFE	R	1	发送 FIFO 为空。这用于指示发送 FIFO 完全为空。 0：发送 FIFO 不为空 1：发送 FIFO 为空 当 TX FIFO 不再为空时，该位被清除。
[1]	TFNF	R	1	发送 FIFO 未满。这用于指示发送 FIFO 未满。 0：发送 FIFO 已满 1：发送 FIFO 未满 TX FIFO 已满时，该位被清除。
[0]	Reserved	Reserved	Reserved	Reserved

23.5.2.11 UART Transmit FIFO Level (UART_TFL)

- **Name**：Transmit FIFO Level
- **Size**：32bits
- **Offset**：0x80
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-5]	Reserved	Reserved	Reserved	Reserved
[4-0]	TFL	R	0	发送 FIFO 级别。这指示发送 FIFO 中的数据条目数

23.5.2.12 UART Receive FIFO Level (UART_RFL)

- **Name**：Receive FIFO Level
- **Size**：32bits
- **Offset**：0x84
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-5]	Reserved	Reserved	Reserved	Reserved
[4-0]	RFL	R	0	接收 FIFO 级别。这表示接收 FIFO 中数据条目的数量。

23.5.2.13 UART Halt TX (UART_HTX)

- **Name**：Halt TX
- **Size**：32bits
- **Offset**：0xA4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	HTX	R/W	0	该寄存器用于暂停传输以进行测试，以便在实现和启用 FIFO 时，主机可以填充发送 FIFO。 0：禁用暂停发送 1：启用暂停发送 注意，如果实现了 FIFO 且未启用 FIFO，则暂停 TX 寄存器的设置不会影响操作。

23.5.2.14 UART Transceiver Control Register (UART_TCR)

- **Name**：Transceiver Control Register
- **Size**：32bits
- **Offset**：0xAC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-5]	Reserved	Reserved	Reserved	Reserved
[4-3]	XFER_MODE	R/W	0	传输模式 ● 0：在这种模式下，发送和接收可以同时发生。您可以随时启用 DE_EN 和 RE_EN。TAT 寄存器中编程的周转时序不适用于此模式。 ● 1：在这种模式下，DE 和 RE 是互斥的。从 RE 切换到 DE 或从 DE 切换到 RE 时，硬件会考虑 TAT 寄存器中编程的周转时序。确保编程时应启用 DE 或 RE。对于传输，硬件在开始传输之前会等待它是否正在接收任何传输。 ● 2：在这种模式下，DE 和 RE 是互斥的。对 DE_EN 或 RE_EN 进行编程后，默认情况下将启用“re”，并且 uart 总线控制器将准备好接收。如果用户使用数据对 TX FIFO 进行编程，则 uart 总线在确保没有进行接收之后，将禁用“re”并启用“de”信号。 TX FIFO 变空后，将启用“re”信号，并且“de”信号将被禁用。在这种操作模式下，当从 RE 切换到 DE 或从 DE 切换到 RE 时，硬件会考虑 TAT 寄存器中编程的周转时序。在这种模式下，“de”和“re”信号严格互补。
[2]	DE_POL	R/W	1	驱动程序启用极性 1：DE 信号为高电平有效 0：DE 信号为低电平有效
[1]	RE_POL	R/W	1	接收器使能极性 1：RE 信号为高电平有效 0：RE 信号为低电平有效
[0]	RS485_EN	R/W	0	RS485 传输启用 ● 0：在此模式下，传输仍处于 RS232 模式。 保留该寄存器中的所有其他字段，并保留寄存器 DE_EN，RE_EN，DET 和 TAT。 ● 1：在此模式下，传输将在 RS485 模式下进行。该寄存器的所有其他字段均适用。

23.5.2.15 UART Driver Output Enable Register (UART_DE_EN)

- **Name**：Driver Output Enable Register
- **Size**：32bits
- **Offset**：0xB0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	DE_EN	R/W	0	DE 启用控制 “DE 使能”寄存器位用于控制“de”信号的断言和断言。 0：取消置位“de”信号 1：置位“de”信号

23.5.2.16 UART Receiver Output Enable Register (UART_RE_EN)

- **Name**：Receiver Output Enable Register
- **Size**：32bits
- **Offset**：0xB4
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-1]	Reserved	Reserved	Reserved	Reserved
[0]	RE_EN	R/W	0	RE 启用控制 “RE 使能”寄存器位用于控制“re”信号的断言和断言。 0：取消置位“re”信号 1：置位“re”信号

23.5.2.17 UART Driver Output Enable Timing Register (UART_DET)

- **Name**：Driver Output Enable Timing Register
- **Size**：32bits
- **Offset**：0xB8
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	Reserved	Reserved	Reserved	Reserved
[23-16]	DE de-assertion time	R/W	0	驱动器启用断言时间。 此字段控制串行输出上停止位结束到驱动器输出启用信号下降沿之间的时间量（以串行时钟周期数为单位）。
[15-8]	Reserved	Reserved	Reserved	Reserved
[7-0]	DE assertion time	R	0	驱动器启用声明时间。 该字段控制在驱动器输出使能信号的上升沿有效到串行发送使能之间的时间（以串行时钟周期数为单位）。发送使能后，发送缓冲区中的任何数据将从串行输出开始。

23.5.2.18 UART TurnAround Timing Register (UART_TAT)

- **Name**：TurnAround Timing Register
- **Size**：32bits
- **Offset**：0xBC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-16]	RE to DE	R/W	0	接收器使能到驱动器使能 TurnAround 时间。 RE 取消声明为 DE 断言的周转时间（以串行时钟为单位）。 注意： ● 如果 DET 寄存器中的 DE 声明时间为 0，则实际值为编程值+ 3。 ● 如果 DET 寄存器中的 DE 声明时间为 1，则实际值为编程值+ 2。 ● 如果 DET 寄存器中的 DE 声明时间大于 1，则实际值为编程值+ 1。
[15-0]	DE to RE	R/W	0	驱动器使能到接收器使能 TurnAround 时间。 DE 置低为 RE 断言的周转时间（以串行时钟为单位）。 注意：实际时间是编程值+1。

23.5.2.19 UART Divisor Latch Fraction Register (UART_DLF)

- **Name**：Divisor Latch Fraction Register
- **Size**：32bits
- **Offset**：0xC0
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-4]	Reserved	Reserved	Reserved	Reserved
[3-0]	DLF	R/W	0	除数的小数部分。 分数值加到 DLH，DLL 设置的整数值中。分数值由（除数分数值）/（2 ^ DLF_SIZE）确定。 DLF_SIZE = 4 下表说明了要为 DLF_SIZE = 4 编程的 DLF 值。

DLF Value	Fraction	Fractional Value
0000	0/16	0.0000
0001	1/16	0.0625
0010	2/16	0.125
0011	3/16	0.1825
0100	4/16	0.25
0101	5/16	0.3125
0110	6/16	0.375
0111	7/16	0.4375
1000	8/16	0.5
1001	9/16	0.5625
1010	10/16	0.625
1011	11/16	0.6875
1100	12/16	0.75
1101	13/16	0.8125
1110	14/16	0.875
1111	15/16	0.9375

23.5.2.20 UART Receive Address Register (UART_RAR)

- **Name：** Receive Address Register
- **Size：** 32bits
- **Offset：** 0xC4
- **Default：** 0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-0]	RAR	R/W	0	这是接收模式下的地址匹配寄存器。如果输入字符中设置了第 9 位，则将对照该寄存器值检查其余的 8 位。如果匹配发生，则将第 9 位设置为 0 的后续字符视为数据字节，直到接收到下一个地址字节为止。 注意：仅当“ADDR_MATCH”（LCR_EXT [1]）和“DLS_E”（LCR_EXT [0]）位设置为 1 时，此寄存器才适用。

23.5.2.21 UART Transmit Address Register (UART_TAR)

- **Name：** Transmit Address Register
- **Size：** 32bits
- **Offset：** 0xC8
- **Default：** 0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-0]	TAR	R/W	0	这是发送模式下的地址匹配寄存器。如果启用了 DLS_E（LCR_EXT [0]）位，则 uart 总线发送第 9 位设置为 1 的 9 位字符，如果“SEND_ADDR”（LCR_EXT[2]）位设置为 1，则将从该寄存器发送剩余的 8 位地址。 注意： ● 该寄存器仅用于发送地址。正常数据应通过编程 THR 寄存器发送。 ● 一旦地址开始在 uart 总线的串行通道上发送，那么“SEND_ADDR”位将被硬件自动清除。

23.5.2.22 UART Line Extended Control Register (UART_LCR_EXT)

- **Name**：Line Extended Control Register
- **Size**：32bits
- **Offset**：0xCC
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-4]	Reserved	Reserved	Reserved	Reserved
[3]	TRANSMIT_ MODE	R/W	0	发送模式控制位。该位用于控制 9 位数据传输期间的传输模式类型。 ● 1：在这种操作模式下，发送保持寄存器（THR）为 9 位宽。您必须确保为地址/数据正确写入了 THR 寄存器。 地址：第 9 位设置为 1 数据：第 9 位设置为 0 注意：发送地址寄存器（TAR）在此操作模式下不适用。 ● 0：在这种操作模式下，发送保持寄存器（THR）为 8 位宽。用户需要将地址编程到发送地址寄存器（TAR）中，并将数据编程到 THR 寄存器中。 SEND_ADDR 位用作控制旋钮，用于指示何时发送地址的 uart 总线。
[2]	SEND_ADDR	R/W	0	发送地址控制位。该位用作控制旋钮，供用户确定何时在发送模式下发送地址。 ● 1–9 位字符将在第 9 位设置为 1 的情况下发送，其余 8 位将与“发送地址寄存器（TAR）”中编程的内容匹配。 ● 0–9 位字符将在第 9 位设置为 0 的情况下发送，其余 8 位将从通过 8 位宽的 THR 寄存器编程的 Tx FIFO 中获取。 注意： 1.发送地址字符后，硬件将自动清除该位。 2.仅当 DLS_E 位设置为 1 且 TRANSMIT_MODE 设置为 0 时，此字段才适用。
[1]	ADDR_MATCH	R/W	0	地址匹配模式。该位用于在接收期间启用地址匹配功能。 1：地址匹配模式；uart 总线将等到第 9 位设置为 1 的传入字符。然后，进一步检查该地址是否与“接收地址匹配寄存器（RAR）”中编程的地址匹配。如果找到匹配项，则随后的字符将被视为有效数据，并且 uart 总线开始接收数据。 0：正常模式；uart 总线将开始接收数据，并且将形成 9 位字符写入 Rx FIFO。用户负责读取数据并区分地址和数据。
[0]	DLS_E	R/W	0	DLS 的扩展。该位用于启用 9 位数据以进行发送和接收传输。 1：每个字符 9 位 0：DLS 选择的数据位数

24 控制器局域网络（CAN）

24.1 简介

CAN 通信是按框架组织的。存在两种类型的框架：标准框架和扩展框架。对于 CAN 2.0，最大数据有效载荷最多为 8 个字节。

在总线访问方面，所有 CAN 节点均相等，没有超级节点，因为 CAN 是多主控总线。数据寻址是使用消息标识符完成的。在 CAN 网络中，只有一个节点应发送带有特定标识符的消息。所有节点都接收所有消息，并且节点主机控制器必须决定是否通过适当的消息标识符对其进行寻址。为了减少主机控制器的负载，CAN 内核可以使用接收滤波器。这些滤波器将所有接收到的消息标识符与用户可选的位模式进行比较。仅当消息通过接收滤波器时，才会通过信号发送到主机控制器。

CAN 帧的标识符也用于总线仲裁。当具有更高优先级标识符的消息被另一个 CAN 节点发送时，CAN 协议机器停止发送具有低优先级标识符的消息。CAN 协议机器会自动尝试在下一个可能的发送位置重新发送已停止的消息。

CAN 2.0B 定义了高达 1Mbit / s 的数据比特率。

24.2 结构框图

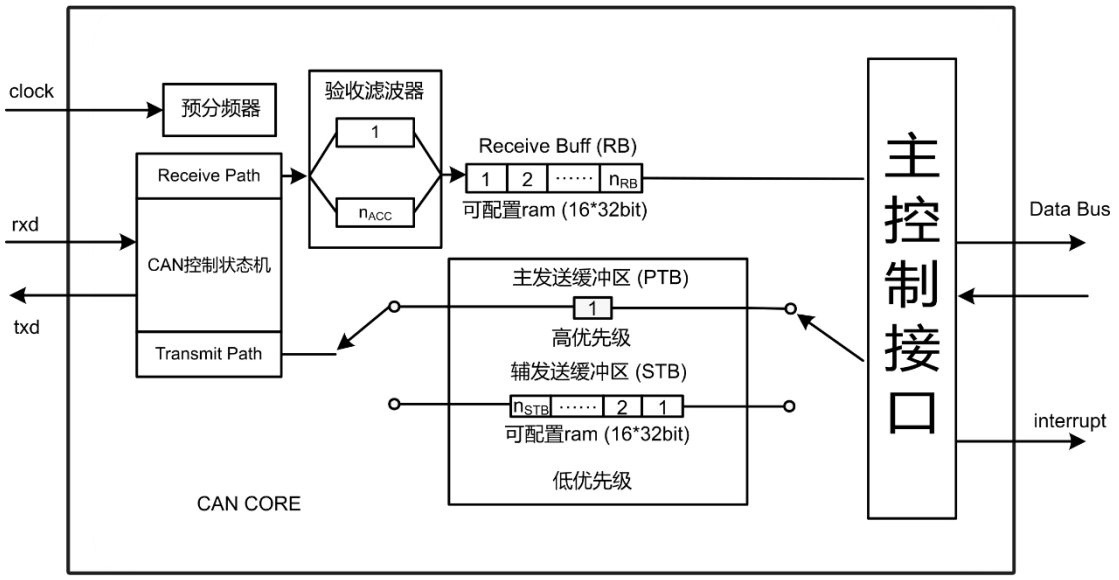


图 24-1 CAN 结构框图

24.3 主要特性

- 支持 CAN 规范
  - CAN 2.0B（最高 8 字节有效负载）
- 自由可编程数据速率
  - CAN 2.0B 定义了高达 1Mbit / s 的数据速率
- 可编程波特率预分频器（1/2 至 1/256）
- 主机接口和 CAN 协议机器的单独时钟域
- 可配置的接收缓冲区（RB）大小
  - 通用参数选择缓冲区插槽的数量
  - 类似于 FIFO 的行为（基于双端口内存）
  - 收到的“不接受”或“不正确”的消息不会覆盖已存储的消息
- 两个发送缓冲区
  - 一个主发送缓冲器（PTB）
  - 可选的可配置二级发送缓冲器（STB）
- 可以包含或不包含 STB。如果包含，则通用参数选择 1 到 16 个缓冲区插槽。
- 类似 FIFO 的行为
  - 一个用于 PTB 和 STB 的双端口存储模块
- 独立的可编程内部 29 位接收滤波器
  - 通用参数可选择的接收滤波器数量，范围为 1 到 16
- 扩展功能
  - 单发传输模式（用于 PTB 和/或 STB）
  - 仅收听模式
  - 回送模式（内部和外部）
  - 收发器待机模式
- 扩展状态和错误报告
  - 捕获上次发生的错误
  - 捕获仲裁丢失的位置
  - 可编程错误警告极限
- 不同的主机控制器接口
  - 32 位同步主机控制器接口；8 位主机的包装器
  - 根据要求，可选的应用程序特定于主机控制器的接口
- 可配置的中断源

24.4 功能描述

英文缩写列表

Abbreviation	Description
ACF	Acceptance Filter
PTB	Primary Transmit Buffer (high priority)
RB	Receive Buffer
RDC	Receiver Delay Compensation (related to TDC)
SP	Sample Point
SSP	Secondary Sample Point (CAN FD)
STB	Secondary Transmit Buffer (low priority)
TDC	Transmitter Delay Compensation (CAN FD specification)
TQ	Time Quanta (CAN specification)

24.4.1 时钟

CAN 使用两个时钟域：一个用于主机接口模块，一个用于 CAN 协议模块。这样，就可以选择以最适合的 CAN 总线数据速率的时钟来操作 CAN，而主机接口则可以自由地以不同的时钟速度进行操作。

时钟域交叉（CDC）通过双缓冲同步器和双向握手机制完成。以下提供有关 CDC 的一些详细信息：

- 控制和状态寄存器分别移至相应的时钟域。例如：中断标志使用双向握手，而状态寄存器使用单向同步器。
- 在启动期间，当 RESET 位变为无效时，验收过滤器将复制到 CAN 时钟域。当 CAN 节点在总线空闲时间之后参与通信时，复制接受过滤器将完成。
- 接收的帧存储在较小的临时接收缓冲区中。该临时缓冲区仅保存部分帧，并在必要时将这些部分复制到 RBUF。
- 将要传输的帧逐步从 TBUF 复制到较小的临时传输缓冲区。

使用同步器和握手机制会导致一些延迟。双向握手在每个域中最多需要 3 个时钟。

24.4.2 Bus-Off 状态

使用寄存器 CFG_STAT 中的状态位 BUSOFF 指示“Bus-Off”状态。如果 CAN 节点的传输错误计数器大于 255，则会自动进入“Bus-Off”状态。然后，它将不再参与进一步的通信，直到它再次返回到错误激活状态。如果启用了 EIE，则将 BUSOFF 设置为 1 也会设置 EIF 中断。如果 CAN 节点通过加电复位进行复位或接收到 128 个 11 位隐性位序列（恢复序列），则它将返回到错误激活状态。

在“Bus-Off”状态下，RECNT 用于计数恢复序列，而 TECNT 保持不变。请注意，进入总线关闭状态时 TECNT 会翻转，因此可能保持较小的值。因此，建议在节点进入总线关闭状态之前使用 TECNT，然后再进入状态位 BUSOFF。

如果节点从“Bus-Off”状态恢复，则 RECNT 和 TECNT 将自动重置为 0。

24.4.3 Acceptance Filters（ACF）

为了减轻主机控制器接收到的帧的负载，CAN 使用了接收滤波器。CANCTRL 内核在接受过滤期间检查消息标识符。因此，每个接收滤波器的长度为 29 位。

如果报文通过了其中一个滤波器，那么该报文将被存储到 RB 中，如果启用了 RIE，RIF 会置 1。如果不接受该报文，则 RIF 不会置 1，并且不增加 RB FIFO 指针。不接受的消息将被丢弃，并被下一条报文覆盖。任何未接受的报文都不会覆盖已存储的有效消息。

与接受过滤的结果无关，CAN 都会检查总线上的每条报文并向总线发送确认或错误帧。

接受掩码定义应比较的位，而接受代码定义适当的值。将屏蔽位设置为 0 可使所选接受代码位与相应的报文标识符位进行比较。设置为 1 的掩码位被禁用以进行接受检查，这将导致接受报文。

标识符位将与相应的接受码位 ACODE 进行如下比较：

- 标准：ID（10：0）和 ACODE（10：0）
- 扩展：ID（28：0）和 ACODE（28：0）

示例：如果 AMASK_x（0）= 0，并且所有其他 AMASK_x 位均为 1，则对于接受的消息，最后一个 ID 位的值必须等于 ACODE（0）。滤波器将忽略所有其他 ID 位。

24.4.4 接收报文

接收到的数据将被存储在 RB 中，RB 其实就是 FIFO。如果启用了 RIE，则每个接收到的有效且已接受的报文都将导致 RIF 置 1。根据填充状态设置 RSTAT。当已填充的缓冲区数等于可编程值 AFWL 时，如果启用了 RAFIE，RAFIF 会置 1。如果所有缓冲区都已满，则如果启用了 RFIE，RFIF 会置 1。

RB 始终将包含最早报文的报文插槽映射到 RBUF 寄存器。

如果 RB 已满，则下一条传入报文将被临时存储，直到通过为止（第 6 个 EOF 位）。然后，如果启用了 ROIE，则最早的报文将被最新报文覆盖，ROIF 会置 1。如果主机控制器读取最旧的报文并在新的传入消息变为有效之前设置 RREL，则不会覆盖任何报文。

处理消息接收

如果没有接受过滤，CAN 将发出信号通知每个帧的接收，并且将要求主机决定是否对其进行寻址。这将导致主机控制器上的负担很大。

可以禁用中断并使用接受过滤器来减轻主机控制器的负载。对于基本操作，如果启用了 RIE 并且 CAN 已收到有效消息，则 RIF 会置 1。为了减少接收中断的数量，可以使用 RAIE/RAIF 或 RFIE/RFIF 代替 RIE / RIF。也可使用 AFWL，不过软件处理速度不够快可能会导致丢失数据。

RB 包含许多 RB 空间，其可以在合成之前使用通用参数来选择。使用 RB 的步骤如下：

1. 使用 RBUF 寄存器从 RB FIFO 中读取最早的报文。
2. 释放 RREL=1 的 RB 插槽。这将选择下一条报文（下一个 FIFO 插槽。RBUF 将自动更新。
3. 重复这些操作，直到 RSTAT 发出空 RB 为止。

如果 RB FIFO 已满，则若新接收到的报文被识别为有效（第 6 个 EOF 位），最早的报文将被覆盖。在此事件之前，不会覆盖任何报文。这将为主机控制器留出足够的时间，以便在 RB FIFO 已满且发生所选中断后，从 RB 读取至少一帧。为了实现此行为，RB 包含比综合参数 RBUF_SLOTS 指定的更多（隐藏）的插槽。该隐藏的插槽用于接收消息，对其进行验证并在覆盖较旧的报文之前检查它是否与接受过滤器匹配。

24.4.5 发送报文

在开始任何传输之前，至少一个传输缓冲区（PTB 或 STB）必须装有一条消息。TPE 发信号通知 PTB 是否锁定，TSSTAT 发信号通知 STB 的填充状态。

TBUF 寄存器提供对 PTB 和 STB 的访问。以下是推荐的流程：

1. 将 TBSEL 设置为所需值以选择 PTB 或 STB。
2. 将帧写入 TBUF 寄存器。
3. 对于 STB，将 TSNEXT = 1 设置为完成此 STB 插槽的加载。

CAN 2.0 消息的最大有效载荷长度为 8 字节，CAN FD 消息的最大有效长度为 64 字节。每个消息的单独长度由 DLC 定义。对于远程帧（RTR 位），DLC 变得毫无意义，因为远程帧的数据长度始终为 0 字节。由于 DLC 和 RTR，要求主机控制器将 TSNEXT 设置为跳到下一个 STB 插槽。所有 TBUF 字节均可按任何顺序写入。

如果 TBSEL = 0 选择了 PTB，则将 TSNEXT = 1 设置是没有意义的。在这种情况下，TSNEXT 会自动清除，不会造成错误。

使用 PTB 时，应将 TPE 位置 1 以开始发送。要使用 STB，必须将 TSONE 设置为开始传输单个消息（最旧的消息），或者将 TSALL 设置为传输所有消息。

PTB 始终具有比 STB 高的优先级。如果两个发送缓冲区都具有发送顺序，则无论帧标识符如何，始终将首先发送 PTB 消息。如果来自 STB 的传输已经处于活动状态，则它将在下一个可能的传输位置（下一个帧间时隙）发送来自 PTB 的消息之前完成。PTB 传输完成或中止之后，CANCTRL 内核返回以处理来自 STB 的其他挂起消息。

传输完成后，将设置以下传输中断：

- 对于 PTB，如果启用 TPIE，则 TPIF 会置 1。
- 对于使用 TSONE 的 STB，如果完成了一条消息并启用了 TSIE，则 TSIF 会置 1。
- 对于使用 TSALL 的 STB，如果所有消息均已完成并且启用了 TSIE，则 TSIF 会置 1。换句话说：如果 STB 为空，则 TSIE 会置 1。因此，如果在开始 TSALL 传输之后，主机控制器将附加消息写入 STB，则 TSIF 置 1 之前，还将发送附加消息。

STB 为空时设置 TSONE 或 TSALL 是没有意义的。在这种情况下，TSONE 或 TSALL 将自动重置。不会设置中断标志，也不会发送任何帧。

消息传输中止

如果出现这种情况，即由于其低优先级而无法发送发送缓冲区中的消息，这将长时间阻塞缓冲区。为了避免这种情况，如果尚未开始传输，则主机控制器可以通过分别设置 TPA 或 TSA 撤回传输请求。

TPA 和 TSA 都提供一个中断标志：AIF。CAN 协议机器仅在不向 CAN 总线传输任何内容时才执行中止操作。因此，以下内容是有效的：

- 总线仲裁期间不中止。
  - 如果节点丢失仲裁，则中止将在之后执行。
  - 如果节点赢得仲裁，则将发送该帧。
- 传输帧时不中止。
  - 如果成功发送了帧，则将成功发送信号通知主机控制器。在这种情况下，不会发出中止的信号。这是通过相应的中断和状态位完成的。
  - 在 CAN 节点未收到确认的传输失败后，错误计数器将递增，并执行中止操作。
  - 如果主机已命令发送所有帧（TSALL = 1），则 STB 中至少剩余一帧，则已完成的帧和中止都将信号发送给主机。

由于这些事实，中止传输可能需要一些时间，具体取决于 CAN 通信速度和帧长度。如果执行中止，则将导致以下操作：

- TPA 释放 PTB，导致 TPE = 0。释放 PTB 之后，帧数据仍存储在 PTB 中。
- TSA 释放一个（最旧的）消息插槽或 STB FIFO 的所有消息插槽。这取决于是使用 TSONE 还是 TSALL 来开始传输。 TSSTAT 将相应更新。

在 STB 中释放帧会导致丢弃帧数据，因为主机由于类似 FIFO 的行为而无法访问它。

不建议同时设置 TPA 和 TSA。如果主机控制器决定仍然执行此操作，则将设置 AIF，并且如果可能的话，来自 PTB 和 STB 的传输都将中止。如已经说明的，如果在中止可以执行之前完成一次传输，这将导致发出成功传输的信号。因此，如果启用，可以设置以下中断标志：

- AIF（一次用于 PTB 和 STB 传输中止）
- TPIF + AIF
- TSIF + AIF
- TPIF + TSIF（很少，仅当主机不立即处理 TPIF 时才会发生）
- TPIF + TSIF + AIF（很少，只有在主机不立即处理 TPIF 和 TSIF 的情况下才会发生）

要清除整个 STB，需要同时设置 TSALL 和 TSA。为了检测是否由于丢失仲裁而无法长时间发送消息，主机可以使用 ALIF / ALIE。

24.4.6 扩展状态和错误报告

在 CAN 总线通信期间，可能会发生传输错误。以下功能支持对其进行检测和分析。

可编程错误警告极限

RECNT 和 TECNT 对接收/发送期间的错误进行计数。主机控制器可使用位于寄存器 LIMIT 中

的可编程错误警告限制 EWL，以对这些事件做出灵活的反应。可以从 8 到 128 的 8 个误差中选择极限值：

错误计数限制=（EWL + 1）* 8

如果在以下情况下由 EIE 启用，则将设置中断 EIF：

- 错误警告限制的边界已被 RECNT 或 TECNT 沿任一方向交叉
- BUSOFF 位已在任一方向上更改。

仲裁丢失捕获（ALC）

内核能够检测仲裁丢失的仲裁字段中的确切位位置。如果已启用此事件，则可以通过 ALIF 中断来发出信号。如果节点能够赢得仲裁，则 ALC 的值保持不变。然后，ALC 保留最后一次仲裁失败的旧值。

ALC 的值定义如下：帧从 SOF 位开始，然后发送 ID 的第一位。该第一 ID 位的 ALC 值为 0，第二 ID 位的 ALC 值为 1，依此类推。

仅在仲裁字段中允许仲裁。因此，ALC 的最大值为 31，这是扩展帧中的 RTR 位。

注：如果标准远程帧与扩展帧进行仲裁，则扩展帧将在 IDE 位丢失仲裁，ALC 将为 12。

错误种类（KOER）

内核识别 CAN 总线上的错误，并将最后的错误事件存储在 KOER 位中。如果使能了 BEIF 中断，则可以指示 CAN 总线错误。每个新的错误事件都会覆盖 KOER 的先前存储的值。因此，主机控制器必须对错误事件做出快速反应。成功发送或接收帧后，将重置 KOER。

24.4.7 扩展功能

单发传输

有时不希望自动重传。因此，可以通过位 TPSS 为发送缓冲区 PTB 和通过位 TSSS 为发送缓冲区 STB 分别设置仅发送一次消息的顺序。在这种情况下，如果选定的传输处于活动状态，则在发生错误或仲裁丢失的情况下，将不会执行任何重新传输。

在立即成功传输的情况下，与正常传输没有区别。但是，如果在传输过程中发生错误，则将 TPIF 置位（如果启用），会清除相应的传输缓冲区，并更新 KOER 和错误计数器。因此，对于单发传输，仅 TPIF 不能指示帧是否已成功传输或发生了错误。

如果将单发传输与 TSALL 一起使用，并且 STB 中有一个以上的帧，则对于每一帧都将进行单发传输。无论是否成功传输了任何帧（例如由于 ACK 错误），CAN 都会前进到下一个帧，如果 STB 为空则停止。

因此，在这种情况下只有错误计数器指示发生了什么。这可能非常复杂，因为如果两个帧之一出现错误，主机将无法检测到哪一个是成功的。

仅收听模式（LOM）

LOM 提供了监视 CAN 总线的能力，而对总线没有任何影响。CAN 的 LOM 与 CAN FD 规范中定义的总线监视功能兼容。

另一个应用是自动比特率检测，其中主机控制器尝试不同的时序设置，直到没有错误发生为止。

在 LOM 中，内核无法将显性位写入总线（没有活动的错误标志或过载标志，没有确认），但是这些显性位在内部环回模式下在内部重新路由，以便内核接收自己的位。将在 LOM 中监视错误（KOER，BEIF）。

对 LOM 重要：

- 无论任何错误情况，错误计数器均保持不变。
- 如果一个节点发送帧，则仅当至少一个附加节点连接到不在 LOM 中的总线时，才会生成 ACK。这样就不会有错误，并且所有节点（包括 LOM 中的那些节点）都将接收该帧。

如果除发送节点之外的所有节点都在 LOM 中，则总线上将没有 ACK，发送器将生成 ACK 错误，并且可能会重新发送帧。因为仅当总线是隐性的，直到 ACK 之后帧比特的第 6 个末尾才接收到帧，因此错误帧将违反此规则，故 LOM 中的节点不会接收到该帧。

传输处于活动状态时，不应启用 LOM。主机控制器必须注意这一点。没有来自 CAN-CTRL 的附加保护。如果启用了 LOM，则无法开始传输。

总线连接测试

要检查节点是否连接到总线，必须执行以下步骤：

- 传送框架。如果节点连接到总线，则其 TX 位在其 RX 输入上可见。
- 如果还有其他节点连接到 CAN 总线，则期望传输成功（包括来自其他节点的确认）。没有错误信号。
- 如果该节点是连接到 CAN 总线的唯一节点（但是总线，收发器和 CAN-CTRL 内核之间的连接很好），则第一个常规错误情况发生在 ACK 插槽中，因为没有来自用户的确认其他节点。如果启用且 KOER =“ 100”（ACK 错误），则将产生 BEIF 错误中断。
- 如果与收发器或总线的连接断开，则在 SOF 位之后立即设置 BEIF 错误中断，并且 KOER =“ 001”（位错误）。

环回模式（LBMI 和 LBME）

CAN 支持两种环回模式：内部（LBMI）和外部（LBME）。两种模式都导致接收到自己发送的帧，这对于自检很有用。

在 LBMI 中，CAN-CTRL 与 CAN 总线断开连接，并且 txd 输出设置为隐性。输出数据流将反馈到输入。

在 LBME 中，CAN-CTRL 保持与收发器的连接，并且已传输的帧在总线上可见。在这种模式下，CAN-CTRL 接收自己的帧。是否存在来自另一个节点的 ACK 无关紧要。

在环回模式下，内核接收自己的消息，将其存储在 RBUF 中，并设置相应的接收中断标志（如果启用）。在回送模式传输期间，将禁用 ACK 错误生成，并且如果启用，将设置适当的传输中断。

LBMI 对于芯片内部和软件测试很有用，而 LBME 可以测试收发器及其连接。

传输处于活动状态时，不应同时激活两种环回模式。主机控制器必须注意这一点。没有来自 CAN-CTRL 的附加保护。

如果该节点已连接到 CAN 总线，则不能仅通过将 LBMI 设置为 0 来完成从 LBMI 切换回正常操作的操作，因为这可能是在另一个 CAN 节点正在传输时发生的。在这种情况下，应通过将 RESET 位设置为 1 来切换回正常工作。这将自动将 LBMI 清零。最后，可以禁止 RESET，并且内核将返回正常工作状态。与此相反，可以每次禁用 LBME。

收发器待机模式

使用寄存器位 STBY 来驱动输出信号 stby。它可用于激活收发器的待机模式。

激活待机后，将无法进行任何传输，因此无法设置 TPE，TSONE 和 TSALL。另一方面，如果传输已经激活，则 CAN 不允许设置 STBY（设置了 TPE，TSONE 或 TSALL）。

如果设置了 STBY，则收发器进入低功耗模式。在这种模式下，它无法全速接收帧，但会监视 CAN 总线是否处于显性状态。如果主导状态在收发器数据手册中定义的时间内处于活动状态，则收发器会将 rxd 信号拉低。如果 rxd 为低，则 CAN 自动将 STBY 清除为 0，这将禁用收发器的待机模式。这是在不中断主机控制器的情况下以静默方式完成的。

从待机模式切换到活动模式需要花费一些时间，因此收发器无法成功接收初始唤醒帧。因此，最近处于待机状态的节点将不会以 ACK 响应。如果总线上没有 CAN 节点通过 ACK 响应唤醒帧，则将导致唤醒帧发送器的 ACK 错误。然后，发送器将自动重复该帧。在重复过程中，收发器将返回活动模式，并且 CAN 将接收该帧并以 ACK 响应。

总结：一个节点发送一帧用于唤醒。如果所有其他节点都处于待机模式，则发送器会收到 ACK 错误，并将自动重复帧。在重复期间，节点返回到活动模式，并将以 ACK 响应。

STBY 不受复位位的影响。

24.4.8 软件复位

如果 CFG_STAT 中的 RESET 位设置为 1，则软件复位有效。 如果 RESET = 1，则多个组件将被强制进入复位状态，但是 RESET 不会触摸所有组件。 某些组件仅对硬件重置敏感。 对于软件和硬件复位，所有位的复位值始终相同。

Register	RESET	Comment
RBUF	Yes	所有 RB 插槽均标记为空。
TBUF	Yes	所有 TB 插槽均标记为空。
LBME	Yes	-
LBMI	Yes	-
TPSS	Yes	-

TSSS	Yes	-
RACTIVE	Yes	即使接收处于活动状态，接收也会立即被取消。 不会产生 ACK。
TACTIVE	Yes	所有传输均立即通过 RESET 终止。 如果传输处于活动状态，则将导致错误的帧。 其他节点将生成错误帧。
BUSOFF	No	-
TBSEL	Yes	TBUF 已固定为指向 PTB。
LOM	Yes	-
STBY	No	-
TPE	Yes	-
TPA	Yes	-
TSONE	Yes	-
TSALL	Yes	-
TSA	Yes	-
TSNEXT	Yes	-
TSSTAT	Yes	所有 STB 插槽均标记为空。
ROV	Yes	所有 RB 插槽均标记为空。
RREL	Yes	-
RSTAT	Yes	所有 RB 插槽均标记为空。
RIE	No	-
ROIE	No	-
RFIE	No	-
RAFIE	No	-
TPIE	No	-
TSIE	No	-
EIE	No	-
TSFF	Yes	所有 STB 插槽均标记为空。
RIF	Yes	-
ROIF	Yes	-
RFIF	Yes	-
RAFIF	Yes	-
TPIF	Yes	-
TSIF	Yes	-
EIF	No	-
AIF	Yes	-
EWARN	No	-
EPASS	No	-
EPIE	No	-
EPIF	Yes	-
ALIE	No	-
ALIF	Yes	-
BEIE	No	-
BEIF	Yes	-

BITTIME_x	No	如果 RESET = 1，则该寄存器是可写的；如果是 0，则该寄存器是写锁定的。
S_PRESC, F_PRESC	No	如果 RESET = 1，则该寄存器是可写的；如果是 0，则该寄存器是写锁定的。
AFWL	No	-
EWL	Yes	-
KOER	Yes	-
ALC	Yes	-
RECNT	No	-
TECNT	No	-
SELMASK	No	-
ACFADR	No	-
AE_x	No	-
ACODE_x	No	如果 RESET = 1，则该寄存器是可写的；如果是 0，则该寄存器是写锁定的。
AMASK_x	No	如果 RESET = 1，则该寄存器是可写的；如果是 0，则该寄存器是写锁定的。

24.5 寄存器描述

24.5.1 寄存器列表

名称	位宽	偏移量	说明
CAN_RBUF	-	0x00	CAN Receive Buffer Registers
CAN_TBUF	-	0x48	CAN Transmit Buffer Registers
CAN_CFG_STAT	[7:0]	0x90	CAN Configuration and Status Register
CAN_TCMD	[7:0]	0x91	CAN Command Register
CAN_TCTRL	[7:0]	0x92	CAN Transmit Control Register
CAN_RCTRL	[7:0]	0x93	CAN Receive Control Register
CAN_RTIE	[7:0]	0x94	CAN Receive and Transmit Interrupt Enable Register
CAN_RTIF	[7:0]	0x95	CAN Receive and Transmit Interrupt Flag Register
CAN_ERRINT	[7:0]	0x96	CAN Error Interrupt Enable and Flag Register
CAN_LIMIT	[7:0]	0x97	CAN Warning Limits Register
CAN_BITTIME	[23:0]	0x98	CAN Bit Timing Register
CAN_S_PRESC	[7:0]	0x9c	CAN Prescaler Registers
CAN_F_PRESC	[7:0]	0x9d	CAN Prescaler Registers
CAN_TDC	[7:0]	0x9e	CAN Transmitter Delay Compensation Register
CAN_EALCAP	[7:0]	0xa0	CAN Error and Arbitration Lost Capture Register
CAN_RECNT	[7:0]	0xa2	CAN Receive Error Count Register
CAN_TECNT	[7:0]	0xa3	CAN Transmit Error Count Register
CAN_ACFCTRL	[7:0]	0xa4	CAN Acceptance Filter Control Register
CAN_ACF_EN	[15:0]	0xa6	CAN Acceptance Filter Enable
CAN_ACF	[32:0]	0xa8	CAN Acceptance Filter

24.5.2 寄存器详细描述

24.5.2.1 CAN_CFG_STAT

- **Name**：CFG_STAT
- **Size**：**8**bits
- **Offset**：0x90
- **Default**：0x80

Bits	Name	Access	Function
[7]	RESET	rw-1	RESET 请求位 1-主机控制器执行 CAN-CTRL 的本地复位。 0-没有本地复位的 CAN-CTRL 仅当 RESET = 1 时才能修改节点配置（BITTIME_x，x_PRESC，接受代码和掩码）。 CAN-CTRL 内核不断监听总线。错误的时间安排和验收过滤器可能会导致不可预测的不良行为。因此，仅当协议机器复位并因此与总线断开连接时，才能更改这些参数。 RESET 位将使多个组件进入复位状态。如果节点进入“总线关闭”状态，则会自动设置 RESET。 请注意，在 11 个 CAN 位时间之后，将 RESET 设置为 0 后，CAN 节点将参与 CAN 通信。 CAN 标准要求此延迟。如果将 RESET 设置为 1 并立即设置为 0，则需要一些时间才能将 RESET 读取为 0 并变为无效状态。原因是时钟域从主机跨越到 CAN 时钟域。根据主机和 CAN 时钟之间的关系，只要需要，RESET 就会保持激活状态。
[6]	LBME	rw-0	环回模式，外部 0-禁用 1-已启用 传输处于活动状态时，不应启用 LBME。
[5]	LBMI	rw-0	内部环回模式 0-禁用 1-已启用 传输处于活动状态时，不应启用 LBMI。
[4]	TPSS	rw-0	PTB 的传输主要单发模式 0-禁用 1-已启用
[3]	TSSS	rw-0	STB 的传输辅助单发模式 0-禁用 1-已启用
[2]	RACTIVE	r-0	接收激活（接收状态位） 1-控制器当前正在接收数据或远程帧。 0-没有接收活动。
[1]	TACTIVE	r-0	传输激活（传输状态位） 1-控制器当前正在传输数据或远程帧。 0-没有发送活动。

[0]	BUSOFF	r-0	总线关闭（总线状态位） 1-控制器状态为“总线关闭”。 0-控制器状态为“ bus on”。

24.5.2.2 CAN_TCMD

- **Name：** TCMD
- **Size：** 8bits
- **Offset：** 0x91
- **Default：** 0x00

Bits	Name	Access	Function
[7]	TBSEL	rw-0	发送缓冲器选择 选择要加载消息的发送缓冲区。 使用 TBUF 寄存器进行访问。 始终在写 TBUF 寄存器以及设置 TSNEXT 时 TBSEL 必须保持稳定。 0-PTB（高优先级缓冲区） 1-STB（类似于 FIFO）
[6]	LOM	rw-0	仅听模式 0-禁用 1-已启用 传输处于活动状态时，不应启用 LOM。 如果启用了 LOM，则无法开始传输。
[5]	STBY	rw-0	收发器待机模式 0-禁用 1-已启用 该寄存器位连接到输出信号 stby，可用于控制收发器的待机模式。 如果 TPE = 1，TSONE = 1 或 TSALL = 1，则 STBY 不能设置为 1。 如果主机将 STBY 设置为 0，则主机需要等待收发器启动所需的时间，然后主机才能请求新的传输
[4]	TPE	rw-0	传输主要启用 1-高优先级 PTB 消息的传输使能 0-PTB 无传输 如果设置了 TPE，则来自 PTB 的消息将在下一个可能的发送位置发送。 从机顶盒开始的传输将在此之前完成，但是待发送的新消息会延迟到 PTB 消息已发送。 TPE 保持设置状态，直到成功发送了消息或使用 TPA 中止消息为止。 仅主机控制器可以设置 TPE。 主机控制器可以将 TPE 设置为 1，但不能将其重置为 0。这只能通过使用 TPA 并中止消息来实现。 在短时间内，CAN-CTRL 内核会重置该位，主机无法设置该位。 如果 RESET = 1，BUSOFF = 1，STBY = 1 或 LOM = 1，则该位将重置为硬件复位值。

[3]	TPA	rw-0	传送主要中止 1 –中止 TPE = 1 请求但尚未开始的来自 TPB 的传输。（消息的数据字节保留在 PTB 中。） 0 –不中止 该位必须由主机控制器设置，并将通过 CAN-CTRL 复位。 设置 TPA 会自动使 TPE 无效。 主机控制器可以将 TPA 设置为 1，但不能将其重置为 0。 在短时间内，CAN-CTRL 内核会重置该位，主机无法设置该位。 如果 RESET = 1 或 BUSOFF = 1，则该位将重置为硬件重置值。 TPA 不应与 TPE 同时设置。
[2]	TSONE	rw-0	传输次一帧 1 –启用 STB 中最早的消息的传输。 消息格式存储在该消息所属的 IDE 缓冲区中。 一旦总线空闲并且没有待处理的 PTB（位 TPE）请求，控制器就开始传输。 0 – STB 没有传输。 TSONE 保持设置状态，直到成功发送了消息或使用 TSA 中止消息为止。 只有主机控制器可以设置 TSONE。 主机控制器可以将 TSONE 设置为 1，但不能将其重置为 0。这只能通过使用 TSA 并中止消息来实现。 在短时间内，CAN-CTRL 内核会重置该位，主机无法设置该位。 如果 RESET = 1，BUSOFF = 1，STBY = 1 或 LOM = 1，则该位将重置为硬件复位值。
[1]	TSALL	rw-0	传输辅助所有帧 1 –启用 STB 中所有消息的传输。消息格式存储在适当的 IDE 缓冲区中。一旦总线空闲并且没有待处理的 PTB（位 TPE）请求，控制器就开始传输。 0 – STB 没有传输。 TSALL 保持设置状态，直到所有消息都已成功发送或使用 TSA 中止为止。仅主机控制器可以设置 TSALL。 主机控制器可以将 TSALL 设置为 1，但不能将其重置为 0。这只能使用 TSA 并中止消息来实现。 在短时间内，CAN-CTRL 内核会重置该位，主机无法设置该位。 如果 RESET = 1，BUSOFF = 1，STBY = 1 或 LOM = 1，则该位将重置为硬件复位值。 如果在传输期间向机顶盒加载了新帧，则新帧也将被传输。 换句话说：当 STB 变空时，由 TSALL 发起的传输完成。 如果 STBY = 1，则 TSALL 不能设置为 1。

			传输二次中止 1 –中止来自 STB 的已请求但尚未开始的传输。 对于 TSONE 传输，仅中止一帧，而对于 TSALL 传输，所有帧均被中止。 将释放一个或所有消息槽，这些消息槽将更新 TSSTAT。 由于 STB 的类似于 FIFO 的行为，所有中止的消息都将丢失，因为它们不再可访问。
[0]	TSA	rw-0	0 –不中止 该位必须由主机控制器设置，并将通过 CAN-CTRL 复位。 设置 TSA，分别自动取消激活 TSONE 或 TSALL。 主机控制器可以将 TSA 设置为 1，但不能将其重置为 0。 如果 RESET = 1 或 BUSOFF = 1，则该位将重置为硬件重置值。 TSA 不应与 TSONE 或 TSALL 同时设置。

24.5.2.3 CAN_TCRTL

- **Name**：TCRTL
- **Size**：8bits
- **Offset**：0x92
- **Default**：0x80

Bits	Name	Access	Function
[7]	FD_ISO	rw-1	CAN FD ISO 模式 0-Bosch CAN FD（非 ISO）模式 1-ISO CAN FD 模式 仅当 RESET = 1 时，该位才可写。 ISO CAN FD 模式具有不同的 CRC 初始化值和其他填充位计数。 两种模式不兼容，并且不能在一个 CAN 网络中混合使用。 该位对 CAN 2.0B 没有影响。
[6]	TSNEXT	rw-0	发送缓冲区次要 NEXT 0-无动作 1-STB 插槽已满，选择下一个 FIFO 插槽。 将所有帧字节写入 TBUF 寄存器后，主机控制器必须将 TSNEXT 设置为信号，表明该插槽已满。 然后，CAN-CTRL 内核将 TBUF 寄存器连接到下一个 FIFO 插槽。 一旦将插槽标记为已填充，就可以使用 TSONE 或 TSALL 开始传输。 可以在一次写访问中一起设置 TSNEXT 和 TSONE 或 TSALL。 TSNEXT 必须由主机控制器设置，并在设置后立即由 CAN-CTRL 内核自动重置。 仅当 TBSEL = 1 选择了 STB 时，设置 TSNEXT 才有意义。如果 TBSEL = 0，则 TSNEXT 被忽略并自动清除。 它没有任何危害。 如果机顶盒的所有插槽均已填满，则 TSNEXT 保持置位状态，直到一个插槽可用为止（第 3.6.7 节）。
[5]	-	r-0	reserved
[4-0]	TSSTAT	r-0	传输辅助状态位已填充消息缓冲区的数量（0…16）。

24.5.2.4 CAN_RCRTL

- **Name**：RCRTL
- **Size**：8bits
- **Offset**：0x93
- **Default**：0x00

Bits	Name	Access	Function
[7-6]	-	r-0	reserved
[5]	ROV	r-0	接收缓冲区 OVerflow 1 –溢出。 至少丢失一条消息。 0 –无溢出。 通过设置 RREL = 1 清除 ROV。
[4]	RREL	rw-0	接收缓冲区释放 主机控制器确认已清空实际的 RB 插槽。 然后，CAN-CTRL 核心指向下一个 RB 插槽。 RSTAT 更新。 1 –release：主机已清空 RB。 0 –no release
[3]	-	r-0	reserved
[1-0]	RSTAT	r-0	接收缓冲区状态 00-空 01->空且<几乎已满（AFWL） 10-≥几乎已满（AFWL 可编程阈值），但未满且没有溢出 11-满（如果发生溢出，则设置停留时间–有关溢出信号，请参阅 ROV）

24.5.2.5 CAN_RTIE

- **Name**：RTIE
- **Size**：8bits
- **Offset**：0x94
- **Default**：0xFE

Bits	Name	Access	Function
[7]	RIE	rw-1	接收中断使能 0 – Disabled, 1 – Enabled
[6]	ROIE	rw-1	RB 溢出中断使能 0 – Disabled, 1 – Enabled
[5]	RFIE	rw-1	RB 全中断使能 0 – Disabled, 1 – Enabled
[4]	RAFIE	rw-1	RB 几乎全中断使能 0 – Disabled, 1 – Enabled
[3]	TPIE	rw-1	传输主中断使能 0 – Disabled, 1 – Enabled
[2]	TSIE	rw-1	传输辅助中断使能 0 – Disabled, 1 – Enabled
[1]	EIE	rw-1	错误中断使能 0 – Disabled, 1 – Enabled
[0]	TSFF	r-0	发送辅助缓冲区满标志 1 - The STB is filled with the maximal count of messages. 0 - The STB is not filled with the maximal count of messages

24.5.2.6 CAN_RTIF

- **Name**：RTIF
- **Size**：8bits
- **Offset**：0x95
- **Default**：0x00

Bits	Name	Access	Function
[7]	RIF	r-0	接收中断标志 1-数据或远程帧已被接收，并且在接收缓冲区中可用。 0-没有接收到帧。
[6]	ROIF	r-0	RB 溢出中断标志 1-RB 中至少覆盖了一条接收到的消息。 0-没有 RB 被覆盖。 如果发生超限，将同时设置 ROIF 和 RFIF。
[5]	RFIF	r-0	RB 全中断标志 1-所有 RB 已满。 如果在接收到下一条有效消息之前没有释放任何 RB，则最早的消息将丢失。 0-RB FIFO 未满。
[4]	RAFIF	r-0	RB 几乎全中断标志 1-已填充的 RB 插槽数量≥AFWL_i 0-已填充的 RB 插槽数<AFWL_i
[3]	TPIF	r-0	传输主中断标志 1-请求的 PTB 传输已成功完成。 0-PTB 的传输尚未完成。
[2]	TSIF	r-0	传输辅助中断标志 1-请求的 STB 传输已成功完成。 0-未成功完成 STB 的传输。
[1]	EIF	r-0	错误中断标志 1-错误警告限制的边界已沿任一方向越过，或者 BUSOFF 位已沿任一方向被改变。 0-不变。
[0]	AIF	r-0	中止中断标志 1-设置 TPA 或 TSA 后，相应的消息已中止。建议不要同时设置 TPA 和 TSA，因为这两个源都是 AIF。 0-没有执行中止。 AIF 没有关联的使能寄存器。

24.5.2.7 CAN_ERRINT

- **Name**：ERRINT
- **Size**：8bits
- **Offset**：0x96
- **Default**：0x00

Bits	Name	Access	Function
[7]	EWARN	r-0	达到错误警告限制 1-错误计数器 RECNT 或 TECNT 之一等于或大于 EWL 0-两个计数器中的值均小于 EWL。
[6]	EPASS	r-0	错误被动模式有效 0-不活动（节点错误活动） 1-主动（节点是错误被动）
[5]	EPIE	rw-0	错误被动中断使能
[4]	EPIF	r-0	错误被动中断标志。 如果错误状态从主动错误变为被动错误，反之亦然，并且允许了该中断，则将激活 EPIF。
[3]	ALIE	rw-0	仲裁丢失中断使能
[2]	ALIF	r-0	仲裁丢失中断标志
[1]	BEIE	rw-0	总线错误中断使能
[0]	BEIF	r-0	总线错误中断标志

24.5.2.8 CAN_LIMIT

- **Name**：LIMIT
- **Size**：8bits
- **Offset**：0x97
- **Default**：0x1B

Bits	Name	Access	Function
[7-4]	AFWL(3:0)	rw-0x1	接收缓冲区几乎满警告限制 AFWL 定义内部警告限制 AFWL_i，其中 n_{RB} 为可用 RB 插槽的数量。 $AFWL\_i = \begin{cases} AFWL \mid n_{RB} < 16 \\ 2 \cdot AFWL \mid 16 \leq n_{RB} < 32 \\ 4 \cdot AFWL \mid 32 \leq n_{RB} < 64 \end{cases}$ 将 AFWL_i 与已填充的 RB 时隙数进行比较，如果相等，则触发 RAFIF。 AFWL_i 的有效范围。 AFWL_i 是没有意义的，会自动视为 0x1。 （请注意，AFWL 在此规则中是指的，而不是 AFWL_i。） AFWL_i > n_{RB} 是没有意义的，会自动视为 nRB。 AFWL_i = n_{RB} 是有效值，但请注意，RFIF 也存在。
[3-0]	EWL(3:0)	rw-0xB	可编程错误警告极限=（EWL + 1）* 8。 可能的极限值：8，16，…128。EWL 的值控制 EIF。 需要使用 CDC 将 EWL 从主机传输到 CAN 时钟域。 在传输过程中，EWL 寄存器的位将为主机写锁定几个时钟，直到 CDC 完成。

24.5.2.9 CAN_BITTIME_0

- **Name**：BITTIME_0
- **Size**：8bits
- **Offset**：0x98
- **Default**：0x23

Bits	Name	Access	Function
[7-6]	F_SJW	rw-02	同步跳转宽度（快速） 同步跳跃宽度 $t_{SJW}=(SJW+1)\cdot TQ$ 是用于缩短或延长重新同步的位时间的最大时间，其中 TQ 是时间量。
[5-0]	S_Seg_1	rw-0x3	位定时段 1（慢速） 比特时间开始后，采样点将设置为 $t_{Seg\_1}=(Seg\_1+2)\cdot TQ$。 Seg_1 = 0 是没有意义的，会自动视为 1。

24.5.2.10 CAN_BITTIME_1

- **Name**：BITTIME_1
- **Size**：8bits
- **Offset**：0x99
- **Default**：0x22

Bits	Name	Access	Function
[7-5]	F_Seg_2	rw-0x2	位定时段 2（快速） 采样点之后的时间 $t_{Seg\_2}=(Seg\_2+1)\cdot TQ$。 Seg_2 = 0 毫无意义，并自动视为 1。
[4-0]	S_Seg_2	rw-0x2	位时序段 2（低速） 采样点之后的时间 $t_{Seg\_2}=(Seg\_2+1)\cdot TQ$。 Seg_2 = 0 毫无意义，并自动视为 1。

24.5.2.11 CAN_BITTIME_2

- **Name**：BITTIME_2
- **Size**：9bits
- **Offset**：0x9a
- **Default**：0x32

Bits	Name	Access	Function
[7-4]	F_Seg_1(3:0)	rw-0x3	位定时段 1（快速） 比特时间开始后，采样点将设置为 $t_{Seg\_1}=(Seg\_1+2)\cdot TQ$。 Seg_1 = 0 是没有意义的，会自动视为 1。
[3-0]	S_SJW(3:0)	rw-02	同步跳跃宽度（慢速） 同步跳转宽度 $t_{SJW}=(SJW+1)\cdot TQ$ 是用于缩短或延长重新同步的位时间的最大时间，其中 TQ 是时间量

24.5.2.12 CAN_S_PRESC

- **Name**：S_PRESC
- **Size**：8bits
- **Offset**：0x9c
- **Default**：0x01

Bits	Name	Access	Function
[7-0]	S_PRESC F_PRESC	rw-0x01	预分频器（慢速和快速） 预分频器将系统时钟分频以获得时间量子时钟 tq_clk。 有效范围 PRESC = [0x01，0xff]导致分频器值为 2 到 256。 禁止 PRESC = 0 并自动将其视为 1。（这是必需的，因为此内核的位时序逻辑需要它。）

24.5.2.13 CAN_F_PRESC

- **Name**：F_PRESC
- **Size**：8bits
- **Offset**：0x9d
- **Default**：0x01

见 5.2.12 S_PRESC 描述。

24.5.2.14 CAN_TDC

- **Name**：TDC
- **Size**：8bits
- **Offset**：0x9e
- **Default**：0x00

Bits	Name	Access	Function
[7]	TDCEN	rw-0	发送器延迟补偿使能 如果 TDCEN = 1，则在 BRS 处于活动状态时，在 CAN FD 帧的数据阶段将激活 TDC。
[6-5]	-	r-0	Reserved
[4-0]	SSPOFF	rw-0x00	二次采样点偏移 发送器延迟加上 SSPOFF 定义了 TDC 辅助采样点的时间。 SSPOFF 作为 TQ 数给出。

24.5.2.15 CAN_EALCAP

- **Name**：EALCAP
- **Size**：8bits
- **Offset**：0xA0
- **Default**：0x00

Bits	Name	Reset Value	Function
[7-5]	KOER(2:0)	r-0x0	一种错误（错误代码） 000-没有错误 001-位错误 010-格式错误 011-资料错误 100-确认错误 101-CRC 错误 110-其他错误 （自己的错误标志后的主要位，接收到的错误标志的时间太长，ACK 错误后在 Passive-Error-Flag 期间的主要位） 111-未使用 成功发送或接收帧后，将重置 KOER。
[4-0]	ALC(4:0)	r-0x0	仲裁丢失捕获（丢失仲裁的帧中的位位置）

24.5.2.16 CAN_RECNT

- **Name**：RECNT
- **Size**：8bits
- **Offset**：0xA2
- **Default**：0x00

Bits	Name	Access	Function
[7-0]	RECNT	r-0x00	接收错误系数（接收期间的错误数） RECNT 按照 CAN 规范中的定义递增和递减。 RECNT 不会溢出。 RECNT 信号 0xff＝255 作为最大值。 见第二章。

24.5.2.17 CAN_TECNT

- **Name**：TECNT
- **Size**：8bits
- **Offset**：0xA3
- **Default**：0x00

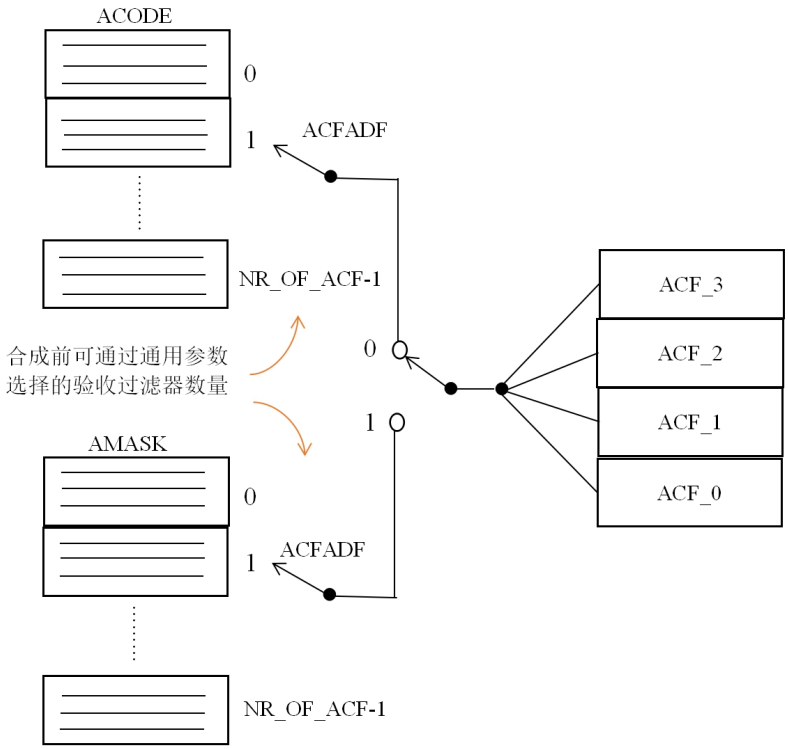
Bits	Name	Access	Function
7:0	TECNT	r-0x00	传输错误系数（传输期间的错误数） TECNT 按照 CAN 规范中的定义递增和递减。 TECNT 不溢出。 TECNT 信号 0xff＝255 作为最大值。

24.5.2.18 CAN_ACFCTRL

- **Name：** ACFCTRL
- **Size：** 8bits
- **Offset：** 0xA4
- **Default：** 0x00

Bits	Name	Access	Function
7:6	-	r-0	reserved
5	SELMASK	rw-0	选择接受面具 0-将 ACF_x 寄存器注册为接受代码 1-将 ACF_x 点注册到接受掩码。 ACFADR 选择一种特定的验收过滤器。
4	-	r-0	Reserved
3:0	ACFADR	rw-0	验收过滤器地址 ACFADR 指向特定的接受过滤器。 所选过滤器可通过寄存器 ACF_x 访问。 SELMASK 位在接受代码和所选接受过滤器的掩码之间进行选择。 ACFADR> NR_OF_ACF-1 的值是没有意义的，并自动视为值 NR_OF_ACF-1。（有关 NR_OF_ACF 的详细信息，请参见第 0 章和图 3-4）

接受过滤器寄存器 ACF_x 提供对接受过滤器代码 ACODE_x 和接受过滤器掩码 AMASK_x 的访问，具体取决于 SELMASK 的设置。 仅当 RESET = 1 时，才可以对 ACF_x 进行读/写访问。 如果 RESET = 0，则从 ACF_x 读取结果为 0。



24.5.2.19 CAN_ACF_EN_0

- **Size**：8bits
- **Offset**：0xa6
- **Default**：0x01

Bits	Name	Access	Function
7:0	AE_x	rw-0x01	验收过滤器启用 1-启用接受过滤器 0-禁用过滤器 每个验收过滤器（AMASK / ACODE）可以单独启用或禁用。 硬件重置后，默认情况下仅启用过滤器编号 0。 禁用的过滤器拒绝消息。如果适当的 AMASK / ACODE 配置匹配，则只有启用的过滤器才能接受消息。 要接受所有消息，必须通过设置 AE_x = 1，AMASK_x = 0xff 和 ACODE_x = 0x00 启用一个过滤器 x。 这是过滤器 x = 0 进行硬件重置后的默认配置，同时禁用了所有其他过滤器。

24.5.2.20 CAN_ACF_EN_1

- **Size**：8bits
- **Offset**：0xa7
- **Default**：0x00

Bits	Name	Access	Function
7:0	AE_x	rw-0x00	验收过滤器启用 1-启用接受过滤器 0-禁用过滤器 每个验收过滤器（AMASK / ACODE）可以单独启用或禁用。 禁用的过滤器拒绝消息。 如果适当的 AMASK / ACODE 配置匹配，则只有启用的过滤器才能接受消息。

24.5.2.21 CAN_ACF

- **Size**：8bits
- **Offset**：0xa8
- **Default**：不确定性

Bits	Name	Access	Function
31	-	r-0	Reserved
30	AIDEE	rw-0 rw-u	接受掩码 IDE 位检查使能 1-接受过滤器接受 AIDE 定义的标准或扩展 0-接受过滤器接受标准或扩展帧 上电复位仅影响滤波器 0。 所有其他过滤器保持未初始化状态。
29	AIDE	rw-0 rw-u	接受掩码 IDE 位值 如果 AIDEE = 1，则： 1-接受过滤器仅接受扩展帧 0-接受过滤器仅接受标准帧 上电复位仅影响滤波器 0。 所有其他过滤器保持未初始化状态。
[28:0]	ACF	-	ACODE_x 或者 AMASK_x；见下两表

Bits	Name	Access	Function
7:0	ACODE_0 ACODE_x	rw-0x00 rw-u	验收 CODE 1-ACC 位值与接收到的消息的 ID 位进行比较 0-ACC 位值与接收到的消息的 ID 位进行比较 ACODE_x（10：0）将用于扩展帧。 ACODE_x（28：0）将用于扩展帧。 上电复位仅影响滤波器 0。 所有其他过滤器保持未初始化状态。 见第二章。 3.6.2 了解更多详情。

Bits	Name	Access	Function
7:0	AMASK_0 AMASK_x	rw-0xFF rw-u	验收 MASK 1-禁用接收标识符的这些位的接受检查 0-接受检查以使能接收标识符的这些位 AMASK_x（10：0）将用于扩展帧。 AMASK_x（28：0）将用于扩展帧。 禁用的位将导致接受消息。 因此，重置过滤器 0 后的默认配置将接受所有消息。 上电复位仅影响滤波器 0。 所有其他过滤器保持未初始化状态。 见第二章。 3.6.2 了解更多详情。

25 通用串行总线（USB）

25.1 简介

USB 控制器是 USB 传输过程中的控制器，作为设备，CPU 可以通过 AHB 总线对 USB 的访问配置 USB 和主机进行数据交互。

25.2 结构框图

图 25-1 为系统架构内 USB 连接示意图。

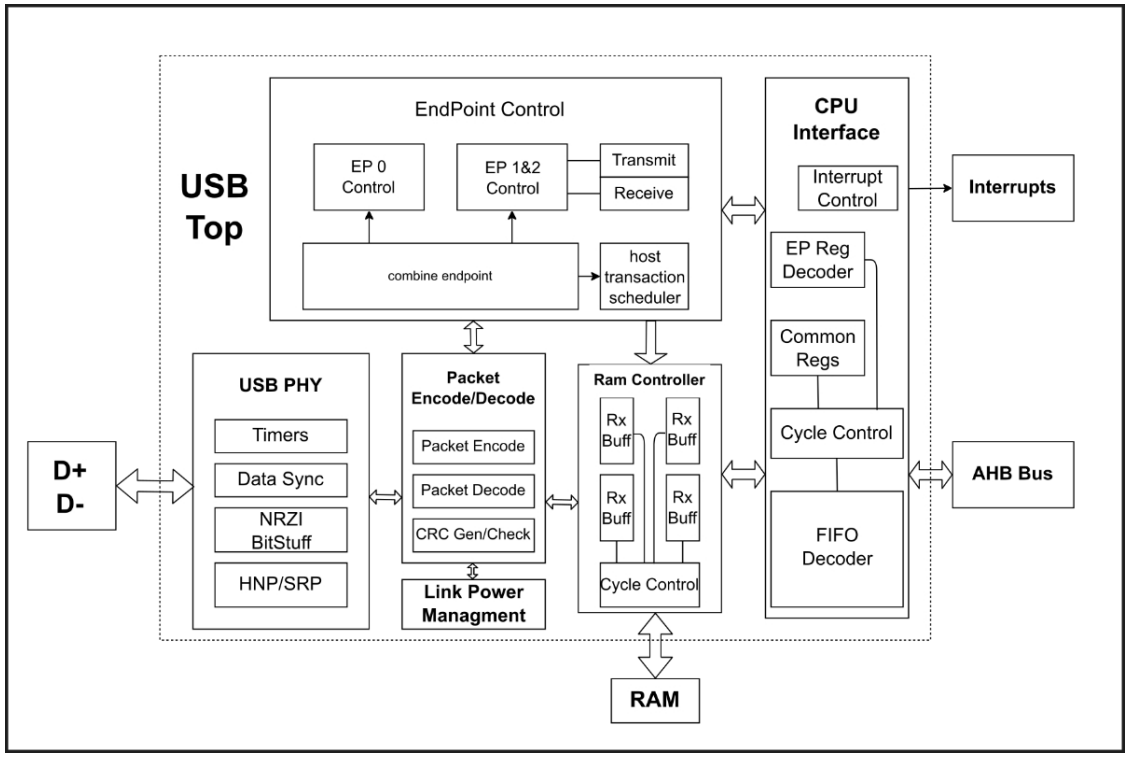


图 25-1　USB 连接框图

25.3 主要性能

- USB2.0 协议，支持全速模式（不支持高速和低速模式）
- 包括端点 0 一共有 3 个端点，都支持 RX 和 TX
- 256 bytes 的专用数据包缓冲存储器
- 循环冗余码校验（CRC）生成/校验，不归零反相（NRZI）编码/解码和位填充
- 支持控制、ISO、中断和 bulk 传输
- USB 插入/拔出检测
- USB 挂起/唤醒功能
- 支持 DMA，端点 1 和端点 2 各配备一个 DMA

25.4 功能描述

25.4.1 插入拔出检测

25.4.1.1 功能描述

图 25-2 为一个 USB 2.0 协议中全速识别方式。

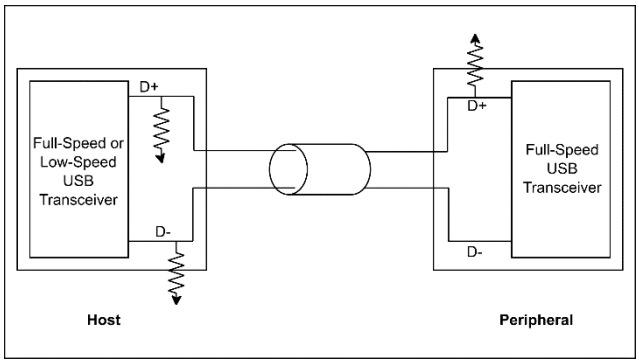


图 25-2 全速识别方式（设备的 D-悬空）

如图 25-2 所示，USB 作为 Host 时，会在 D+和 D-分别施加 15k 的下拉电阻；USB 作为 Peripheral 时，会在 D+上施加一个 1.5k 的上拉电阻。当设备没有连入主机时，主机 D+和 D-都为 0，设备的 D+为 1，D-为高阻态；当设备连入主机时，主机 D+的 15k 下拉相对于设备 D+的 1.5K 上拉为弱下拉电阻，因此此时 D+为 1；设备的 D-悬空，因此 D-为 0。

如果使用图 2 中的全速识别方式，作为设备来说在不插入时的 D-位不定态，无法确定 D-的值，而插入后 D-为 0，在没有 vbus 的前提下比较难去做一个设备检测是否插入主机，因此，在不影响主机的判断的情况下，在我们芯片对 D-做了一个弱上拉，如图 25-3 所示：

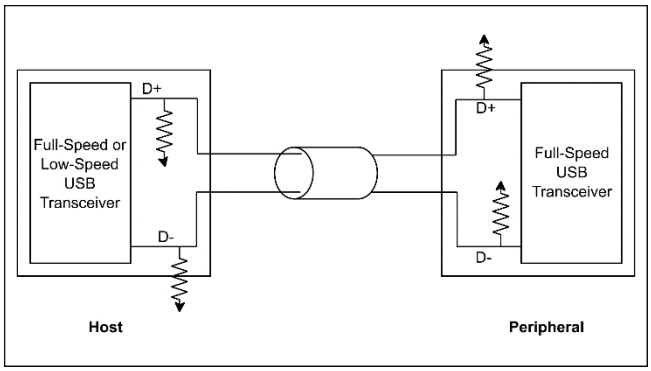


图 25-3 全速识别方式（设备的 D-加弱上拉）

如图 25-3 所示，在设备端加入 D-的 300k 上拉电阻，当设备没有连入主机时，主机 D+和 D-都为 0，设备的 D+为 1，D-不再为高阻态，而是 1；当设备连入主机时，主机 D+的 15k 下拉相对于设备 D+的 1.5k 上拉为弱下拉电阻，因此此时 D+为 1；主机 D-的 15k 下拉相对于设备 D-的 300k 上拉电阻为强下拉电阻，因此此时 D-为 0；

上述不会破坏主机的判断，同时从机也可以更好的做插入拔出检测，无插入时，设备的 D+和

D-都为 1；插入后，设备的 D+为 1 和 D-为 0；再拔出时，设备的 D+和 D-都为 1。插入检测就是 D+和 D-全是 1 的状态转换到 D+和 D-不全是 1 的状态；拔出检测就是 D+和 D-不全是 1 的状态转换到 D+和 D-全是 1 的状态。

25.4.1.2 配置方式

插入检测

1. 启动 USB，配置 UCFG0 为 0x44003d；
2. 配置 UCFG2[15:4]的 Debouce_Max，配置去抖动的阈值；
3. 将 UCFG2[1]的 DET_INT_EN 置 1，开启插入检测中断；
4. 收到中断后检测 UCFG2[0]的 DET_INT 是否为 1，并写 1 清 0，此时设备已经连入主机。

拔出检测

1. 配置 UCFG2[15:4]的 Debouce_Max，配置去抖动的阈值；
2. 将 UCFG2[3]的 DISCONN_INT_EN 置 1，开启拔出检测中断；
3. 收到中断后检测 UCFG2[2]的 DISCONN_INT 是否为 1，并写 1 清 0，此时设备已经从主机拔出；
4. 此时 USB 是复位状态，要重新启动 USB，需要重新配置除 UCFG0、UCFG1 和 UCFG2 外其他的 USB 寄存器。

25.4.2 USB Reset（Reset 中断）

当主机向 USB 发送 reset 信号时，USB 会禁用所有的端点寄存器的通讯，仅仅会开放端点 0 寄存器的通讯；并且 FADDR 的设备地址会被清 0；此时会起一个 RESET 中断告知软件主机已经和 USB 设备连接成功了，可以着手准备枚举了。

25.4.3 IN 事务

当接收到 IN 令牌包时，如果接收到的地址与已配置的有效端点匹配，USB 将访问与寻址端点相关的缓冲区并获取数据。再次访问数据包存储器以读取要发送的第一个字节，并开始发送 DATA0 或 DATA1 PID。PID 完成后，将从缓冲存储器读取的第一个字节加载到输出移位寄存器中，以在 USB 总线上传输。发送完最后一个数据字节后，将发送计算出的 CRC。如果寻址的端点无效，则根据 TXCSRN 寄存器中的 SendStall 位发送 NAK 或 STALL 握手数据包，而不是数据包。

25.4.4 OUT/SETUP 事务

在芯片设计中，OUT 和 SETUP 包以相同的方式处理；当接收到 OUT / SETUP PID 时，如果地址与有效端点匹配，USB 随后将接收到的数据字节被打包为字节，然后传输到指定的数据包缓冲区，同时每收到一个字节 RXCOUNT 将会自增 1。当检测到 DATA 数据包的末尾时，将测试接收到的 CRC 的正确性，并且只有在接收过程中没有发生错误时，才会将 ACK 握手

数据包发送回发送主机。此时起一个中断，可以通过读取 UCFG2 寄存器中的 USBSendState 位去知道当前收到的是 SETUP 包还是 OUT 包。

如果出现错误的 CRC 或其他类型的错误（位冲突，帧错误等），数据字节仍会复制到数据包存储缓冲区中，至少直到错误检测点为止，但不会发送 ACK 数据包。但是，在这种情况下，通常不需要采取任何软件操作：USB 外设从接收错误中恢复并为下一次交易做好准备。如果寻址的端点无效，则根据 TXCSRN 寄存器中的 SendStall 位发送 NAK 或 STALL 握手数据包，而不是 ACK，并且不将任何数据写入接收存储器缓冲区。

25.4.5 控制传输

端点 0 是核心的主要控制端点，需要该软件来处理可能通过端点 0 发送或接收的所有标准设备请求。USB 外围设备收到的标准设备请求可以分为三类：零数据请求（命令中包括所有信息），写入请求（命令后跟其他数据）和读取请求（要求设备将数据发送回主机）。

25.4.5.1 中断

端点 0 中断生成：

- 接收到有效令牌并将数据写入 FIFO 后，内核将 RxPktRdy 位（TXCSRN.D0）置 1 时。
- 当 FIFO 中的数据包已成功发送到主机后，内核清除 TxPktRdy 位（TXCSRN.D1）。
- 由于协议冲突导致控制交易结束后，内核将 SentStall 位（TXCSRN.D2）置 1 时。
- CPU 将 SetupEnd 位（TXCSRN.D4）置 1 时，因为在设置 DataEnd（TXCSRN.D3）之前控制传输已结束。

25.4.5.2 零数据请求

零数据请求的所有信息都包含在 8 字节命令中，不需要传输其他数据。 “零数据”标准设备请求的示例包括： SET_FEATURE ， CLEAR_FEATURE ， SET_ADDRESS ，SET_CONFIGURATION，SET_INTERFACE。

与所有请求一样，事件序列将在软件收到端点 0 中断时开始。 RxPktRdy 位（TXCSRN.D0）也将被置位。然后应从端点 0 FIFO 中读取 8 字节命令，对其进行解码并采取适当的措施。例如，如果命令为 SET_ADDRESS，则命令中包含的 7 位地址值应写入 FAddr 寄存器。然后，应写入 TXCSRN 寄存器以设置 ServicedRxPktRdy 位（D6）（指示已从 FIFO 读取命令）并设置 DataEnd 位（D3）（指示此请求不需要其他数据）。当主机移至请求的状态阶段时，将生成第二个 Endpoint 0 中断以指示请求已完成。该软件无需采取进一步措施：第二个中断只是对请求成功完成的确认。

如果该命令是无法识别的命令，或者由于某些其他原因而无法执行，则在对其进行解码后，应写入 CSR0 寄存器以设置 ServicedRxPktRdy 位（D6）和设置 SendStall 位（D5）。当主机移至请求的状态阶段时，USB 将发送一个 STALL 通知主机未执行请求。将产生第二个端点 0 中断，并将 SentStall 位（TXCSRN.D2）置 1。如果在将 DataEnd 位置 1 之后主机发送更多数据，则 USB 将发送一个 STALL。将产生一个端点 0 中断，并且 SentStall 位（TXCSRN.D2）将被

置 1。

25.4.5.3 写入请求

写请求涉及在 8 字节命令之后从主机发送的一个或多个附加数据包。 “写入”标准设备请求的一个示例是：SET_DESCRIPTOR。

与所有请求一样，事件序列将在软件收到端点 0 中断时开始。RxPktRdy 位（TXCSRN.D0）也将被置位。然后应从端点 0 FIFO 中读取 8 字节命令并进行解码。与零数据请求一样，然后应写入 TXCSRN 寄存器以设置 ServicedRxPktRdy 位（D6）（指示已从 FIFO 中读取命令），但在这种情况下，不应设置 DataEnd 位（D3）（指示预计会有更多数据）。收到第二个端点 0 中断时，应读取 CSR0 寄存器以检查端点状态。 RxPktRdy 位（CSR0L.D0）应设置为指示已接收到数据包。然后应读取 RXCOUNT 寄存器以确定此数据包的大小。然后可以从端点 0 FIFO 中读取数据包。

如果与请求关联的数据长度（由命令中的 wLength 字段指示）大于端点 0 的最大数据包大小，则将发送其他数据包。在这种情况下，应写入 TXCSRN 以将 ServicedRxPktRdy 位置 1，但不应将 DataEnd 位置 1。当接收到所有预期的数据包时，应写入 TXCSRN 寄存器以将 ServicedRxPktRdy 位置 1 并将 DataEnd 位置 1（指示不需要更多数据）。当主机移至请求的状态阶段时，将生成另一个端点 0 中断以指示请求已完成。该软件不需要采取进一步的措施，中断只是对请求成功完成的确认。

如果该命令是无法识别的命令，或者由于某些其他原因而无法执行，则在对其进行解码后，应写入 TXCSRN 寄存器以设置 ServicedRxPktRdy 位（D6）和设置 SendStall 位（D5）。当主机发送更多数据时，USB 将发送一个 STALL 通知主机未执行该请求。将产生一个端点 0 中断，并且 SentStall 位（TXCSRN.D2）将被置 1。如果主机在设置了 DataEnd 之后发送更多数据，则 USB 将发送一个 STALL。将产生一个端点 0 中断，并且 SentStall 位（TXCSRN.D2）将被置 1。

25.4.5.4 读取请求

读请求具有一个 8 字节命令后从功能发送到主机的数据包。“读取”标准设备请求的示例包括：GET_CONFIGURATION ， GET_INTERFACE ， GET_DESCRIPTOR ， GET_STATUS ， SYNCH_FRAME。

与所有请求一样，事件序列将在软件收到端点 0 中断时开始。 RxPktRdy 位（TXCSRN.D0）也将被置位。然后应从端点 0 FIFO 中读取 8 字节命令并进行解码。然后应写入 TXCSRN 寄存器以设置 ServicedRxPktRdy 位（D6）（指示该命令已从 FIFO 中读取）。然后应将要发送到主机的数据写入端点 0 FIFO。如果要发送的数据大于端点 0 的最大数据包大小，则应仅将最大数据包大小写入 FIFO。然后，应写入 TXCSRN 寄存器以设置 TxPktRdy 位（D1）（指示 FIFO 中有要发送的数据包）。将数据包发送到主机后，将生成另一个端点 0 中断，并且可以将下一个数据包写入 FIFO。将最后一个数据包写入 FIFO 后，应写入 TXCSRN 寄存器以设置 TxPktRdy 位和设置 DataEnd 位（D3）（指示此数据包之后没有更多数据）。当主机移至请求的状态阶段时，将生成另一个端点 0 中断以指示请求已完成。该软件不需要采取进一步的措

施：中断只是对请求成功完成的确认。

如果该命令是无法识别的命令，或者由于某些其他原因而无法执行，则在对其进行解码后，应写入 TXCSRN 寄存器以设置 ServicedRxPktRdy 位（D6）和设置 SendStall 位（D5）。当主机请求数据时，USB 将发送一个 STALL 来通知主机该请求未执行。将产生一个端点 0 中断，并且 SentStall 位（TXCSRN.D2）将被置 1。如果在设置了 DataEnd（D3）之后主机请求更多数据，则 USB 将发送一个 STALL。将产生一个端点 0 中断，并且 SentStall 位（TXCSRN.D2）将被置 1。

25.4.6 BULK 传输

25.4.6.1 BULK IN 传输

批量输入事务用于将非周期性数据从功能控制器传输到主机。三个可选功能可用于批量 IN 事务的 Tx 端点一起使用：

- 双包缓冲
  当写入 TxMaxP 寄存器的值小于或等于分配给端点的 FIFO 大小的一半时，将自动启用双包缓冲。启用后，最多可在 FIFO 中存储两个包，等待传输到主机。
- DMA
  如果为端点启用了 DMA，则只要端点能够在其 FIFO 中接受另一个数据包，就会生成 DMA 请求。此功能可用于允许 DMA 控制器将数据包加载到 FIFO 中，而无需处理器干预。如果使用 DMA 模式 1，则必须将 TxMaxP [D10：0]设置为偶数，以正确产生中断。
- AutoSet
  启用 AutoSet 后，当将一个 TxMaxP 字节的数据包加载到 FIFO 中时，TxPktRdy 位（TXCSRN.D0）将被自动设置。当使用 DMA 加载 FIFO 时，这特别有用，因为在大型批量传输期间加载单个数据包时，无需任何处理器干预。

设置

在为批量事务配置 Tx 端点时，必须使用端点的最大数据包大小（以字节为单位）写入 TxMaxP 寄存器。此值应与端点的标准端点描述符的 wMaxPacketSize 字段相同。此外，应将 TXCSRN 寄存器的高字节按如下表所示设置（位 D9–D8 未使用）：

D15	AutoSet	0/1	如果需要 AutoSet 功能，则设置为 1。
D14	ISO	0	设置为 0 以启用批量传输。
D13	Mode	1	设置为 1 以确保启用 FIFO。
D12	DMAReqEnab	0/1	如果需要 DMA 请求，则设置为 1。 注意：如果设置为 1，则还需要选择所选的 DMAReqMode（TXCSRN.D10）。
D11	FrcDataTog	0	设置为 0 允许正常的数据切换操作。

首次配置端点时（遵循端点 0 上的 SET_CONFIGURATION 或 SET_INTERFACE 命令），应写入 TXCSRN 的低字节以将 ClrDataTog 位（D6）置 1。 这将确保数据切换（由 USB 自动处

理）以正确的状态启动。 同样，如果 FIFO 中有任何数据包（通过设置 FIFONotEmpty 位（TXCSRN.D1）指示），则应通过将 FlushFIFO 位（TXCSRN.D3）置位来清除它们。 注意：如果启用了双重缓冲，则可能有必要连续两次将该位置位。

操作

当要通过 Bulk IN 管道传输数据时，需要将数据包加载到 FIFO 中，并写入 TXCSRN 寄存器以设置 TxPktRdy 位（D0）。发送数据包后，USB 将清除 TxPktRdy 位，并产生一个中断，以便可以将下一个数据包加载到 FIFO 中。如果启用了双重数据包缓冲，则在第一个数据包已装入并且 TxPktRdy 位置 1 后，USB 将立即清除 TxPktRdy 位，并产生一个中断，以便将第二个数据包装入 FIFO。该软件应以相同的方式运行，在接收到中断时加载数据包，而不管是否启用了双重数据包缓冲。

通常情况下，数据包的大小不得超过 TxMaxP 寄存器的低 11 位所指定的大小。寄存器的这一部分定义了通过 USB 传输的有效负载（数据包大小），并且 USB 规范要求其为 8、16、32、64（全速或高速）或 512 字节（高速）。如果要传输的数据量超过此数量，则需要将其作为多个 USB 数据包发送，大小应均为 TxMaxP [D10：D0]，但最后一个保留残差的数据包除外。

主机可以通过知道期望的数据总量来确定用于传输的所有数据。可替代地，当它接收到小于规定的数据包（TxMaxP [D10：D0]）时，它可以推断出所有数据已被发送。在后一种情况下，如果数据块的总大小是此有效负载的倍数，则该功能将有必要在所有数据都已发送之后发送空数据包。通过在接收到下一个中　　断时将 TxPktRdy 设置为完成，而不将任何数据加载到 FIFO 中。

如果软件要关闭 Bulk IN 管道，则应将 SendStall 位（TxCSRL.D4）置 1。当 USB 收到下一个 IN 令牌时，它将向主机发送一个 STALL，将 SentStall 位（TXCSRN.D5）置 1 并产生一个中断。当软件收到已将 SentStall 位（TXCSRN.D5）置 1 的中断时，应清除 SentStall 位。但是，它应该将 SendStall 位（TXCSRN.D4）置 1，直到准备重新启用 Bulk IN 管道为止。注意：如果主机由于某种原因未能接收到 STALL 数据包，它将发送另一个 IN 令牌，因此建议将 SendStall 位置 1，直到软件准备好重新启用 Bulk IN 管道为止。重新启用管道后，应通过将 TXCSRN 寄存器（D6）中的 ClrDataTog 位置 1，重新启动数据切换序列。

25.4.6.2 BULK OUT 传输

Bulk OUT 事务用于将非定期数据从主机传输到功能控制器。三个可选功能可用于大容量 OUT 事务的 Rx 端点一起使用：

- 双包缓冲
  当写入 RxMaxP 寄存器的值小于或等于分配给端点的 FIFO 大小的一半时，将自动启用双包缓冲。启用后，FIFO 中最多可以存储两个包。
- DMA
  如果为端点启用了 DMA，则只要端点在其 FIFO 中有数据包，就会生成 DMA 请求。此功能可用于允许 DMA，而无需处理器干预。 。如果使用 DMA 模式 1，则必须将 RxMaxP [D10：0]设置为偶数，以产生适当的中断。
- AutoClear

启用 AutoClear 功能后，当已从 FIFO 中卸载 RxMaxP 字节的数据包时，RxPktRdy 位（RxCSRL.D0）将被自动清除（例外，请参见寄存器说明）。当使用 DMA 卸载 FIFO 时，这特别有用，因为它避免了在大批量传输期间卸载单个数据包时需要任何处理器干预。

设置

在为批量输出事务配置 Rx 端点时，必须使用该端点的最大数据包大小（以字节为单位）写入 RxMaxP 寄存器。 该值应与端点的标准端点描述符的 wMaxPacketSize 字段相同。 此外，应将 RXCSRN 寄存器的高字节应如下所示设置（位 D10–D8 未使用/只读）：

D15	AutoClear	0/1	如果需要自动清除功能，则设置为 1。
D14	ISO	0	设置为 0 以启用批量传输。
D13	DMAReqEnab	0/1	如果此端点需要 DMA 请求，则设置为 1。 注意：如果设置为 1，则还需要选择所选的 DMAReqMode（RxCSRN.D11）。
D12	DisNyet	0	设置为 0 以允许正常的 PING 流控制。

首次配置端点时（遵循端点 0 上的 SET_CONFIGURATION 或 SET_INTERFACE 命令），应写入 RXCSRN 的低字节以设置 ClrDataTog 位（D7）。 这将确保数据切换（由 USB 自动处理）以正确的状态启动。 同样，如果 FIFO 中有任何数据包（由 RxPktRdy 位（RXCSRN.D0 表示）指示），则应通过将 FlushFIFO 位（RXCSRN.D4）置位来刷新它们。 注意：如果启用了双重缓冲，则可能有必要连续两次将该位置位。

操作

当 Bulk Rx 端点接收到数据包时，RxPktRdy 位（RXCSRN.D0）被置位并产生中断。软件应读取端点的 RxCount 寄存器，以确定数据包的大小。应从 FIFO 中读取数据包，然后应清除 RxPktRdy。如果在清除 RxPktRdy 时将 FIFOFull 位设置为 1，则 USB 将首先清除 FIFOFull 位。然后它将再次设置 RxPktRdy，以指示 FIFO 中有另一个要卸载的数据包。

收到的数据包不应超过 RxMaxP 寄存器中指定的大小（因为这应该是在发送给主机的端点描述符的 wMaxPacketSize 字段中设置的值）。当需要将大于 wMaxPacketSize 的数据块发送给函数时，它将作为多个数据包发送。除最后一个包含残差的数据包外，所有数据包的大小均为 wMaxPacketSize。该软件可以使用特定于应用的方法来确定块的总大小，并因此确定何时已接收到最后一个数据包。可替代地，当其接收到大小小于 wMaxPacketSize 的分组时，可以推断出整个块已经被接收。 （如果数据块的总大小是 wMaxPacketSize 的倍数，则将在数据之后发送一个空数据包以表示传输已完成。）

如果要传输大块数据，则可以使用 DMA 避免调用中断服务程序来卸载每个数据包的开销。

25.4.7 中断传输

中断 IN 事务使用与批量 IN 事务相同的协议，并且可以以相同的方式使用。同样，中断 OUT 事务使用与 Bulk OUT 事务几乎相同的协议，并且可以以相同的方式使用。

Tx 端点支持 Bulk IN 事务中不支持的 Interrupt IN 事务的一项功能——支持数据切换位的连续切换。通过将 TxCSR 寄存器（D11）中的 FrcDataTog 位置 1，可以启用此功能。当该位设置为“ 1”时，无论是否从主机接收到 ACK，USB 都会将数据包视为已成功发送，并切换端点的数据位。

另一个区别是中断端点不支持 PING 流控制。这意味着 USB 绝不应该响应 NYET 握手，而只能响应 ACK / NAK / STALL。为确保这一点，应将 RXCSRN 寄存器（D12）中的 DisNyet 位设置为 1，以禁止传输 NYET 握手。

尽管 DMA 可以与 Interrupt OUT 端点一起使用，但是由于中断端点通常希望在单个数据包中传输所有数据，因此它通常没有什么好处。

25.4.8 ISO 传输

25.4.8.1 ISO IN 传输

同步 IN 事务用于将周期性数据从功能控制器传输到主机。本节描述了 USB 全速同步 Tx 端点的使用。可将三种可用于同步 IN 事务的 Tx 端点一起使用：

- 双包缓冲
  当写入 TxMaxP 寄存器的值小于或等于分配给端点的 FIFO 大小的一半时，将自动启用双包缓冲。启用后，最多可在 FIFO 中存储两个包，等待传输到主机。注意：通常建议对同步事务使用双包缓冲，以避免运行不足错误（请参见下面的“操作”）。
- DMA
  如果为端点启用了 DMA，则只要端点能够在其 FIFO 中接受另一个数据包，就会生成 DMA 请求。此功能可用于允许 DMA 控制器在无需处理器干预的情况下将数据包加载到 FIFO 中。但是，此功能对于同步端点不是特别有用，因为传输的数据包通常不是最大数据包大小，并且在每个数据包之后都需要访问 TxCSR 寄存器以检查运行不足错误。
- AutoSet
  当为低带宽同步端点启用自动设置功能时，当将 TxMaxP 字节的数据包加载到 FIFO 中时，TxPktRdy 位（TxCSRL.D0）将自动设置。但是，此功能对于同步端点不是特别有用，因为传输的数据包通常不是最大数据包大小，并且在每个数据包之后都需要访问 TXCSRN 寄存器以检查运行不足错误。

设置

在为同步 IN 事务配置 Tx 端点时，必须使用端点的最大数据包大小（以字节为单位）写入 TxMaxP 寄存器。 此值应与端点的标准端点描述符的 wMaxPacketSize 字段相同。 此外，应将 TXCSRN 寄存器的高字节按如下表所示设置（位 D9–D8 未使用）：

D15	AutoSet	0/1	如果需要 AutoSet 功能，则设置为 1。
D14	ISO	1	设置为 1 以启用 ISO 传输。
D13	Mode	1	设置为 1 以确保启用 FIFO。
D12	DMAReqEnab	0/1	如果需要 DMA 请求，则设置为 1。 注意：如果设置为 1，则还需要选择所选的 DMAReqMode（TXCSRN.D10）。
D11	FrcDataTog	0	设置为 0 允许正常的数据切换操作。

操作

同步端点不支持数据重试，因此，如果要避免运行中的数据，必须在接收 IN 令牌之前将要发送到主机的数据加载到 FIFO 中。主机每帧将发送一个 IN 令牌，但是帧（或微帧）内的时序可能会有所不同。如果在一个帧的结尾附近接收到 IN 令牌，然后在下一帧的开始处接收到 IN 令牌，则将没有时间重新加载 FIFO。因此，通常需要对端点进行双重缓冲。

AutoSet 功能可与低带宽同步 Tx 端点一起使用，其方式与批量 Tx 端点相同。但是，除非数据以绝对一致的速率从源到达，并且与主机的帧时钟同步，否则发送到主机的数据包的大小必须逐帧增加或减少（或从微帧到微帧）才能匹配源数据速率。这意味着实际的数据包大小将不总是 TxMaxP，从而使自动设置功能无效。

每当将数据包发送到主机时，都会产生一个中断，并且软件可以使用该中断将下一个数据包加载到 FIFO 中，并以与批量 Tx 端点相同的方式将 TXCSRN 寄存器（D0）中的 TxPktRdy 位置 1。由于中断可能几乎在一帧（/微帧）内的任何时间发生，具体取决于主机安排事务的时间，因此这可能会导致 FIFO 加载请求的时序不规则。如果端点的数据源来自某些外部硬件，则在加载 FIFO 之前等到每个帧（/微帧）结束可能会更方便，因为这将最大程度地减少对额外缓冲的需求。这可以通过使用 SOF 中断（IntrUSB.D3）触发下一个数据包的加载来完成。中断仍可用于设置 TXCSRN（D0）中的 TxPktRdy 位并检查数据超限/不足运行（请参见下面的“错误处理”）。

启动双缓冲同步 IN 管道可能会引起问题。双缓冲要求数据包只有在加载后的帧（/微帧）之后才发送。如果函数在主机设置管道之前（因此开始发送 IN 令牌），至少将第一个数据包加载到一个帧（/微帧），则没有问题。但是，如果主机在第一个数据包加载时已经开始发送 IN 令牌，则取决于在 IN 之前还是之后加载，可以在加载该数据包的同一帧（/微帧）中传输该数据包。令牌已收到。可以通过将电源寄存器（D7）中的 ISO 更新位置 1 来避免此潜在问题。当该位设置为 1 时，加载到同步 Tx 端点 FIFO 中的任何数据包都不会被发送，直到接收到下一个 SOF 包之后，从而确保该数据包不会过早发送。

错误处理

如果端点在接收到 IN 令牌后在其 FIFO 中没有数据，它将向主机发送一个空数据包，并将 TXCSRN 寄存器（D2）中的 UnderRun 位置 1。这表明软件无法为主机提供足够快的数据。由应用程序确定如何处理此错误情况。

如果软件正在每帧（/微帧）加载一个数据包，并且发现它想加载下一个数据包时 TXCSRN 寄存器（D0）中的 TxPktRdy 位被置位，则表明尚未发送数据包（也许 因为来自主机的 IN 令牌

已损坏）。 取决于应用程序如何处理这种情况：它可以选择通过将 TXCSRN 寄存器（D3）中的 FlushFIFO 位置 1 来刷新未发送的数据包，或者可以选择跳过当前数据包。

25.4.8.2 ISO OUT 传输

同步 OUT 事务用于将周期性数据从主机传输到功能控制器。本节描述了在外围模式下全速同步 Rx 端点的使用三种可选功能可用于同步 OUT 事务的 Rx 端点一起使用：

- 双包缓冲
  当写入 RxMaxP 寄存器的值小于或等于分配给端点的 FIFO 大小的一半时，将自动启用双包缓冲。启用后，最多可在 FIFO 中存储两个包，等待传输到主机。注意：通常建议对同步事务使用双包缓冲，以避免溢出错误（请参见下面的“操作”）。
- DMA
  如果为端点启用了 DMA，则只要端点在其 FIFO 中有数据包，就会生成 DMA 请求。此功能可用于允许 DMA 控制器在无需处理器干预的情况下从 FIFO 卸载数据包。但是，此功能在同步端点上不是特别有用，因为传输的数据包通常不是最大数据包大小，并且在每个数据包之后都需要访问 RxCSR 寄存器以检查溢出或 CRC 错误。
- AutoClear
  启用自动清除功能后，当已从 FIFO 中卸载 RxMaxP 字节的数据包时，RxPktRdy 位（RxCSRL.D0）将被自动清除（例外，请参见寄存器说明）。但是，此功能在同步端点上不是特别有用，因为传输的数据包通常不是最大数据包大小，并且在每个数据包之后都需要访问 RxCSR 寄存器以检查溢出或 CRC 错误。

设置

在为同步 OUT 事务配置 Rx 端点时，必须使用该端点的最大数据包大小（以字节为单位）写入 RxMaxP 寄存器。 此值应与端点的标准端点描述符的 wMaxPacketSize 字段相同。 此外，应将 RXCSRN 寄存器的高字节按如下表所示设置（位 D10–D8 未使用/只读）：

D15	AutoClear	0/1	如果需要自动清除功能，则设置为 1。
D14	ISO	1	设置为 1 以启用 ISO 传输。
D13	DMAReqEnab	0/1	如果此端点需要 DMA 请求，则设置为 1。 注意：如果设置为 1，则还需要选择所选的 DMAReqMode（RxCSRN.D11）。
D12	DisNyet	0	在同步模式下被忽略。

操作

同步端点不支持数据重试，因此，如果要避免数据溢出，则在接收数据包时，FIFO 中必须有空间来接受数据包。主机每帧（或高速模式下的微帧）将发送一个数据包，但是帧内的时间可能会有所不同。如果在一个帧（/微帧）的末尾附近接收到一个数据包，而另一个在下一帧的开始处到达，则将没有时间卸载 FIFO。因此，通常需要对端点进行双重缓冲。

AutoClear 功能可与同步 Rx 端点一起使用，方式与批量 Rx 端点相同。但是，除非数据接收器以绝对一致的速率接收数据并同步到主机的帧时钟，否则从主机发送的数据包的大小必须逐

帧增加或减少（或从微帧到微帧）才能匹配所需的数据速率。这意味着实际的数据包大小并不总是 RxMaxP，从而使 AutoClear 功能失效。

每当从主机接收到数据包时，都会产生一个中断，并且软件可以使用此中断从 FIFO 卸载数据包并以与批量 Rx 端点相同的方式清除 RxCSR 寄存器（D0）中的 RxPktRdy 位。由于中断几乎可以在一帧（/微帧）内的任何时间发生，具体取决于主机安排事务的时间，因此 FIFO 卸载请求的时间可能不规则。如果端点的数据接收器要连接到某些外部硬件，则最好等到每个帧（/微帧）结束后再卸载 FIFO，以最大程度地减少对额外缓冲的需求。这可以通过使用 SOF 中断（IntrUSB.D3）。中断仍可用于清除 RXCSRN 中的 RxPktRdy 位并检查数据是否超限/不足运行（请参见下面的“错误处理”）。

错误处理

如果从主机接收到数据包时，如果 FIFO 中没有空间存储数据包，则 RxCSR 寄存器（D2）中的溢出位将被置 1。 这表明该软件无法为主机快速卸载数据。 由应用程序确定如何处理此错误情况。

如果 USB 发现接收到的数据包存在 CRC 错误，它将仍然将数据包存储在 FIFO 中，并将 RxPktRdy 位（RXCSRN.D0）和 DataError 位（RXCSRN.D3）置 1。 由应用程序决定如何处理此错误情况。

25.5 寄存器描述

25.5.1 寄存器列表

名称	位宽	偏移量	说明
Common USB registers（00h~0Fh）：提供对 USB 的控制和状态			
USB_FAddr	[7:0]	0x00	Function address register
USB_Power	[7:0]	0x01	Power management register.
USB_IntrTx	[15:0]	0x02	Interrupt register for Endpoint 0 plus TX Endpoints 1 to 15
USB_IntrRx	[15:0]	0x04	Interrupt register for Rx Endpoints 1 to 15
USB_IntrTxE	[15:0]	0x06	Interrupt enable register for IntrTx.
USB_IntrRxE	[15:0]	0x08	Interrupt enable register for IntrRx.
USB_IntrUSB	[7:0]	0xA	Interrupt register for common USB interrupts
USB_IntrUSBE	[7:0]	0xB	Interrupt enable register for IntrUSB
USB_Frame	[15:0]	0xC	Frame number.
USB_Index	[7:0]	0xE	Index register for selecting the endpoint status and control registers.
USB_Testmode	[7:0]	0xF	Enables the USB 2.0 test modes
Indexed Endpoint Control/Status registers（10h~1Fh）：这些寄存器为当前选定的端点提供控制和状态。（这一部分的寄存器是分为是否为端点 0，是否为 host，同一个寄存器在不同的配置映射的内容是不一样的）			
USB_TxMaxP	[15:0]	0x10	Maximum packet size for TX endpoint.
USB_TxCSRN	[15:0]	0x12	Control Status register for Endpoint 0 or TX endpoint ;
USB_RxMaxP	[15:0]	0x14	Maximum packet size for Rx endpoint.
USB_RxCSRN	[15:0]	0x16	Control Status register for Rx endpoint.
USB_RxCount	[15:0]	0x18	Number of bytes to be read from Rx endpoint FIFO.
USB_TxType	[7:0]	0x1A	(Host Only) Sets the transaction protocol, speed and peripheral endpoint number for the host TX endpoint.
USB_TxInterval	[7:0]	0x1B	(Host Only)Sets the polling interval for Interrupt/ISOC transactions or the NAK response timeout on Bulk transactions for host TX endpoint.
USB_RxType	[7:0]	0x1C	(Host Only)Sets the transaction protocol, speed and peripheral endpoint number for the host Rx endpoint.
USB_RxInterval	[7:0]	0x1D	(Host Only) Sets the polling interval for Interrupt/ISOC transactions or the NAK response timeout on Bulk transactions for host Rx endpoint.
USB_FIFOSize	[7:0]	0x1F	Returns the configured size of the selected Rx FIFO and TX FIFOs.
FIFOs（20h–5Fh）：此地址范围提供对端点 FIFO 的访问			
USB_FIFO0	[31:0]	0x20	FIFOs for Endpoints 0
USB_FIFO1	[31:0]	0x24	FIFOs for Endpoints 1
USB_FIFO2	[31:0]	0x28	FIFOs for Endpoints 2
Additional Control and Configuration registers（60h–7Fh）：这些寄存器提供其他设备状态和控制。			
USB_DevCtl	[7:0]	0x60	OTG device control register
USB_MISC	[7:0]	0x61	Miscellaneous Register

USB_TxFIFOsz	[7:0]	0x62	TX Endpoint FIFO size（no use）
USB_RxFIFOsz	[7:0]	0x63	Rx Endpoint FIFO size（no use）
USB_TxFIFOadd	[15:0]	0x64	TX Endpoint FIFO address（no use）
USB_RxFIFOadd	[15:0]	0x66	RX Endpoint FIFO address（no use）
USB_HWVers	[15:0]	0x6C	Hardware Version Number Register
USB_EPInfo	[7:0]	0x78	Information about numbers of TX and Rx endpoints
USB_RAMInfo	[7:0]	0x79	Information about the width of the RAM and the number of DMA channels
USB_LinkInfo	[7:0]	0x7A	Information about delays to be applied
USB_VPLen	[7:0]	0x7B	Duration of the VBus pulsing charge
USB_HS_EOF1	[7:0]	0x7C	Time buffer available on High-Speed transactions.
USB_FS_EOF1	[7:0]	0x7D	Time buffer available on Full-Speed transactions.
USB_LS_EOF1	[7:0]	0x7E	Time buffer available on Low-Speed transactions.
USB_SOFT_RST	[7:0]	0x7F	Soft Reset.
Extended Registers（300h – 347h）			
USB_RqPktCount1	[15:0]	0x304	Number of requested packets for Receive Endpoint 1
USB_RqPktCount2	[15:0]	0x308	Number of requested packets for Receive Endpoint 2
USB_RxDPktBufDis	[15:0]	0x340	Double Packet Buffer Disable register for Rx Endpoints 1 to 15
USB_TXDPktBufDis	[15:0]	0x342	Double Packet Buffer Disable register for TX Endpoints 1 to 15
USB_C_T_UCH	[15:0]	0x344	This register sets the Chirp Timeout Timer
USB_C_T_HSRTN	[15:0]	0x346	This register sets the delay from the end of High Speed resume signaling to enable UTM normal operating mode.
LPM Registers (360h – 365h)			
USB_LPM_ATTR	[15:0]	0x360	LPM Attribute Register
USB_LPM_CNTRL	[7:0]	0x362	LPM Control Register
USB_LPM_INTREN	[7:0]	0x363	LPM Interrupt Enable Register
USB_LPM_INTR	[7:0]	0x364	LPM Interrupt Register
USB_LPM_FADDR	[7:0]	0x365	LPM Function Address Register
User Registers(400h – 407h): 用户自定义的寄存器			
USB_Ucfg0	[31:0]	0x400	包含了模拟的寄存器、OTG 相关的配置
USB_Ucfg1	[31:0]	0x404	包含了各个中断是否送出到 CPU 的 mask
USB_Ucfg2	[31:0]	0x408	包含了插入拔出检测的中断信号

25.5.2 寄存器详细描述

25.5.2.1 USB_FADDR

- **Description：** Function address register
- **Size：** 8bits
- **Offset：** 0x00
- **Default：** 0x0

BIT	Name	Reset	R/W	Description
[7]	Reserved	0	R	Reserved
[6-0]	FADDR	0	R/W	设备地址

25.5.2.2 USB_POWER

- **Description：** Power management register
- **Size：** 8bits
- **Offset：** 0x01
- **Default：** 0x20

BIT	Name	Reset	R/W	Description
[7]	ISO Updata	0	R/W	当由 CPU 设置时，MUSBMHDRC 将等待从设置 TxPktRdy 开始的 SOF 令牌，然后再发送数据包。 如果在 SOF 令牌之前收到 IN 令牌，则将发送零长度的数据包。 注：只会影响执行同步传输的端点
[6]	Soft Conn	0	R/W	如果启用了“Soft Connect/Disconnect”功能，则当 CPU 将该位置 1 时将启用 USB D + / D-线，而当 CPU 将该位置 1 时将其变为三态。
[5]	HS Enab	1	R/W	当由 CPU 设置时，当集线器重置设备时，MUSBMHDRC 将协商为高速模式。 如果未设置，则设备将仅在全速模式下运行。 1：高速模式 0：全速模式
[4]	HS Mode	0	R	置位时，此只读位指示 USB Reset 期间成功协商了高速模式。 在外设模式下，当 USB Reset 完成时（检测 USB Reset 中断），此选项才有效。
[3]	Reset	0	R	当总线上存在复位信号时，该位置 1。
[2]	Resume	0	R/W	当设备处于挂起模式时，由 CPU 设置以生成恢复信令。 在外设模式下，CPU 应在 10 毫秒（最多 15 毫秒）后清除此位，以结束恢复信令。
[1]	Suspend Mode	0	R/W	在外设模式下，此位设置为进入挂起模式。 当 CPU 读取中断寄存器或将上面的 Resume 位设置为 1 时，该位被清除。
[0]	Enable SuspendM	0	R/W	由 CPU 设置以启用 SUSPENDM 输出

25.5.2.3 USB_INTRTX

- **Description：** Interrupt register for Endpoint 0 plus TX Endpoint 1 to 15
- **Size：** 16bits
- **Offset：** 0x02
- **Default：** 0x0

BIT	Name	Reset	R/W	Description
[15-3]	Reserved	0	R	Reserved
[2]	EP2 TX INT	0	R	TX Endpoint 2 interrupt
[1]	EP1 TX INT	0	R	TX Endpoint 1 interrupt
[0]	EP0 INT	0	R	Endpoint 0 interrupt

25.5.2.4 USB_INTRRX

- **Description：** Interrupt register for RX Endpoint 1 to 15
- **Size：** 16bits
- **Offset：** 0x04
- **Default：** 0x0

BIT	Name	Reset	R/W	Description
[15-3]	Reserved	0	R	Reserved
[2]	EP2 RX INT	0	R	RX Endpoint 2 interrupt
[1]	EP1 RX INT	0	R	RX Endpoint 1 interrupt
[0]	Reserved	0	R	Reserved

25.5.2.5 USB_INTRTXE

- **Description：** Interrupt enable register for IntrTx
- **Size：** 16bits
- **Offset：** 0x06
- **Default：** 0x07

BIT	Name	Reset	R/W	Description
[15-3]	Reserved	0	R/W	Reserved
[2]	EP2 TX INT EN	1	R/W	TX Endpoint 2 interrupt enable
[1]	EP1 TX INT EN	1	R/W	TX Endpoint 1 interrupt enable
[0]	EP0 INT EN	1	R/W	Endpoint 0 interrupt enable

25.5.2.6 USB_INTRRXE

- **Description：** Interrupt enable register for IntrRx
- **Size：** 16bits
- **Offset：** 0x08
- **Default：** 0x06

BIT	Name	Reset	R/W	Description
[15-3]	Reserved	0	R/W	Reserved
[2]	EP2 RX INT EN	0	R/W	RX Endpoint 2 interrupt enable
[1]	EP1 RX INT EN	0	R/W	RX Endpoint 1 interrupt enable
[0]	Reserved	0	R/W	Reserved

25.5.2.7 USB_INTRUSB

- **Description：** Interrupt register for common USB interrupts
- **Size：** 8bits
- **Offset：** 0x0A
- **Default：** 0x0

BIT	Name	Reset	R/W	Description
[7]	Vbus Error（no use）	0	R	在会话期间 VBus 降至 VBus Valid 阈值以下时设置
[6]	Sess Req（no use）	0	R	在检测到会话请求信令时设置
[5]	Discon	0	R	在会话结束后设置为 1
[4]	Conn（no use）	0	R	当检测到设备连接时设置为 1
[3]	SOF	0	R	当一个新的帧开始时设置为 1
[2]	Reset	0	R	当在总线上检测到复位信号时设置为 1
[1]	Resume	0	R	当 USB 处于挂起模式时，在总线上检测到继续信令时设置。
[0]	Suspend	0	R	在总线上检测到挂起信号时设置为 1

25.5.2.8 USB_INTRUSBE

- **Description：** Interrupt enable register for IntrUSB
- **Size：** 8bits
- **Offset：** 0x0B
- **Default：** 0x0

BIT	Name	Reset	R/W	Description
[7]	Vbus Error EN	0	R/W	Vbus Error Interrut Enable
[6]	Sess Req EN	0	R/W	Sess Req Interrut Enable
[5]	Discon EN	0	R/W	Discon Interrut Enable
[4]	Conn EN	0	R/W	Conn Interrut Enable
[3]	SOF EN	0	R/W	SOF Interrut Enable
[2]	Reset EN	1	R/W	Reset Interrut Enable
[1]	Resume EN	1	R/W	Resume Interrut Enable
[0]	Suspend EN	0	R/W	Suspend Interrut Enable

25.5.2.9 USB_FRAME

- **Description**：Frame number
- **Size**：16bits
- **Offset**：0x0C
- **Default**：0x0

BIT	Name	Reset	R/W	Description
[15-11]	Reserved	0	R	Reserved
[10-0]	FADDR	0	R	保存最后收到的帧号

25.5.2.10 USB_INDEX

- **Description**：Index register for selecting the endpoint status and control registers
- **Size**：8bits
- **Offset**：0x0E
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31]	EMODE	R/W	0x0	FLASH Erase Mode: 1：Chip Erase 0：Sector Erase
[30-9]	Reserved	Reserved	Reserved	Reserved
[8-0]	ESNB	R/W	0x0	FLASH Erase Sector Number select： 仅当 EMODE=0 时有效。 擦除扇区号选择：0 – 159（共计 160 个扇区，每个扇区 512 字节）。

25.5.2.11 USB_TESTMODE

- **Description：** Enables the USB 2.0 test modes
- **Size：** 8bits
- **Offset：** 0x0F
- **Default：** 0x0

BIT	Name	Reset	R/W	Description
[7]	Force Host	0	R/W	当将该位置 1 时，CPU 将该位置 1 以指示内核进入主机模式，而不管其是否连接至任何外设。
[6]	FIFO Access	0	R/W	CPU 将该位置 1，以将端点 0 TX FIFO 中的数据包传输到端点 0 Rx FIFO。 它会自动清除。
[5]	Force FS	0	R/W	
[4]	Force HS	0	R/W	
[3]	Test Packet（no use）	0	R/W	CPU 将该位置 1 以进入 Test_Packet 测试模式。 在这种模式下，USB 在总线上重复发送一个 53 字节的测试数据包
[2]	Test K（no use）	0	R/W	CPU 将该位置 1 以进入 Test_K 测试模式。 在这种模式下，USB 在总线上传输连续的 K。
[1]	Test J（no use）	0	R/W	CPU 将该位置 1 以进入 Test_J 测试模式。 在这种模式下，USB 在总线上传输连续的 J。
[0]	Test SE0 NAK（no use）	0	R/W	CPU 将该位置 1 以进入 Test_SE0_NAK 测试模式。 在此模式下，USB 保持高速模式，但使用 NAK 响应任何有效的 IN 令牌。

Force FS	Force HS	Speed
0	0	低速
0	1	高速
1	0	全速
1	1	Reserved

25.5.2.12 USB_TXMAXP

- **Description：** Maximum packet size for TX endpoint
- **Size：** 16bits
- **Offset：** 0x10
- **Default：** 0x0

BIT	Name	Reset	R/W	Description
[15-0]	TXMAXP	0	R/W	定义单个事务中传输的最大有效负载。该值最多可以为 1024 字节，受 USB 规范中对 FS 和 HS 操作中的 Bulk，中断和同步传输的数据包大小的约束。

25.5.2.13 USB_TXCSRN (端点 0)

- **Description**：Control Status register for Endpoint 0 or TX endpoint
- **Size**：16bits
- **Offset**：0x12
- **Default**：0x0

BIT	Name	Reset	R/W	Description
[8]	FlushFIFO	0	R/W	CPU 向该位写入 1，以刷新要从端点 0 FIFO 读取/发送的下一个数据包。 FIFO 指针被复位并且 TxPktRdy / RxPktRdy 位被清除。 **注：仅当设置了 TxPktRdy / RxPktRdy 时，才应使用 FlushFIFO。在其他时候，这可能会导致数据损坏。**
[7]	ServicedSetupEnd	0	R/W	CPU 将 1 写入该位以清除 SetupEnd 位。 它会自动清除。
[6]	ServicedRxPktRdy	0	R/W	CPU 将 1 写入该位以清除 RxPktRdy 位。 自动清除
[5]	SendStall	0	R/W	CPU 向该位写入 1 以终止当前事务。 将发送 STALL 握手信号，然后自动清除该位。
[4]	SetupEnd	0	R	当控制事务在 DataEnd 位置 1 之前结束时，该位置 1。 此时将产生一个中断并刷新 FIFO。 CPU 通过将 1 写入 ServicedSetupEnd 位来清除该位。
[3]	DataEnd	0	R/W	CPU 设置此位： 1.在为最后一个数据包设置 TxPktRdy 时。 2.卸载最后一个数据包后清除 RxPktRdy。 3.为零长度的数据包设置 TxPktRdy 时。 自动清除
[2]	SentStall	0	R/W	发送停止握手时，该位置 1。 CPU 应该清除该位。
[1]	TxPktRdy	0	R/W	将数据包加载到 FIFO 后，CPU 将该位置 1。 传输数据包后，它将自动清除。 此时也会生成一个中断（如果启用）
[0]	RxPktRdy	0	R	当接收到数据包时该位置位。 该位置 1 时产生中断。 CPU 通过将 ServicedRxPktRdy 位置 1 清除该位。

25.5.2.14 USB_TXCSRN (其他端点)

- **Description**：Control Status register for TX endpoint
- **Size**：16bits
- **Offset**：0x12
- **Default**：0x0

BIT	Name	Reset	R/W	Description
[15]	AutoSet	0	R/W	如果 CPU 将该位置 1，则将最大数据包大小（TxMaxP 中的值）的数据加载到 TX FIFO 中时，TxPktRdy 将自动设置。 如果加载的数据包小于最大数据包大小，则必须手动设置 TxPktRdy。 **注：高带宽同步端点或高带宽中断端点不应该设置此位**
[14]	ISO	0	R/W	CPU 将该位置 1 以使 TX 端点启用同步传输，并清除该位以使 TX 端点启用批量或中断传输。
[13]	Mode	0	R/W	CPU 将该位置 1 以将端点方向启用为 TX，将其清除以将其启用为 Rx。 **注：该位仅在将相同的端点 FIFO 用于 TX 和 Rx 事务时才有效。**
[12]	DMAReqEnab	0	R/W	CPU 将该位置 1 以使能 TX 端点的 DMA 请求。
[11]	FrcDataTog	0	R/W	CPU 将该位置 1，以强制端点数据切换切换，并强制从 FIFO 中清除数据包，而不管是否收到 ACK。 可以用于中断 TX 端点，用于为同步端点传递速率反馈。
[10]	DMAReqMode	0	R/W	CPU 将该位置 1 以选择 DMA 请求模式 1，并将其清除以选择 DMA 请求模式 0。 **注：在清除上述 DMAReqEnab 位之前或与之相同的周期内，不得清除该位。**
[9-8]	Reserved	0	R	Reserved
[7]	IncompTx	0	R/W	当端点用于高带宽同步时，此位设置为指示将大数据包拆分为 2 或 3 个数据包进行传输，但未接收到足够的 IN 令牌来发送所有部分的位置。 注意：在同步传输以外的任何方式下，该位将始终返回 0。
[6]	ClrDataTog	0	R/W	CPU 在该位写入 1 以将端点数据触发器重置为 0
[5]	SendStall	0	R/W	发送停止握手时，此位置 1。 刷新 FIFO 并清除 TxPktRdy 位（见下文）。 CPU 应该清除该位。
[4]	SendStall	0	R/W	CPU 对此位写入 1，以向 IN 令牌发出 STALL 握手信号。 CPU 清除该位以终止停止条件。 注意：在将端点用于同步传输的位置，此位无效。
[3]	FlushFIFO	0	R/W	CPU 对此位写入 1，以刷新来自端点 TX FIFO 的最新数据包。 复位 FIFO 指针，清除 TxPktRdy 位（下），并产生中断。 可以与 TxPktRdy 同时设置以中止当前正在加载到 FIFO 中的数据包。 注意：仅当设置了 TxPktRdy 时，才应使用 FlushFIFO。 在其他时候，这可能会导致数据损坏。 另请注意，如果 FIFO 是双缓冲的，则可能需要将 FlushFIFO 设置两次以完全清除 FIFO。
[2]	UnderRun	0	R/W	如果未设置 TxPktRdy 时接收到 IN 令牌，则 USB 会将该位置 1。 CPU 应该清除该位。

[1]	FIFONoEmpty	0	R/W	当 TX FIFO 中至少有 1 个数据包时，USB 将该位置 1
[0]	TxpktRdy	0	R/W	将数据包加载到 FIFO 后，CPU 将该位置 1。 传输数据包后，它将自动清除。 此时也会产生一个中断（如果允许）。 在将第二个数据包加载到双缓冲 FIFO 中之前，TxPktRdy 也会自动清除。

25.5.2.15 USB_RXMAXP

- **Description**：Maximum packet size for RX endpoint
- **Size**：16bits
- **Offset**：0x14
- **Default**：0x0

BIT	Name	Reset	R/W	Description
[15-0]	RXMAXP	0	R/W	定义（以字节为单位）在单个事务中传输的最大有效负载。 该值集的最大长度为 1024 个字节，但受 USB 规范对全速和高速操作中的批量，中断和同步传输的数据包大小的约束。

25.5.2.16 USB_RXCSRN

- **Description**：Control Status register for RX endpoint
- **Size**：16bits
- **Offset**：0x16
- **Default**：0x0

BIT	Name	Reset	R/W	Description
[15]	AutoClear	0	R/W	如果 CPU 将该位置 1，则当已从 Rx FIFO 卸载 RxMaxP 字节的数据包时，RxPktRdy 位将被自动清除。 当小于最大数据包大小的数据包被卸载时，必须手动清除 RxPktRdy。 使用 DMA 卸载 RxFIFO 时，无论 RxMaxP 是多少，都以 4 字节块的形式从 RxFIFO 读取数据。 因此，RxPktRdy 位将被清除，如下所示： **余（RXMAXP/4）** | **实际读取的字节** | **清除 RxPktRdy 的数据包大小。** **0 (i.e. RXMaxP = 64 bytes)** | RXMAXP | RXMAXP, RXMAXP-1, RXMAXP-2, RXMAXP-3 **3 (i.e. RXMaxP = 63 bytes)** | RXMAXP+1 | RXMAXP, RXMAXP-1, RXMAXP-2 **2 (i.e. RXMaxP = 62 bytes)** | RXMAXP+2 | RXMAXP, RXMAXP-1 **1 (i.e. RXMaxP = 61 bytes)** | RXMAXP+3 | RXMAXP **注：高带宽同步端点不应该设置**
[14]	ISO	0	R/W	CPU 将该位置 1 以使 Rx 端点启用同步传输，并清除该位以使 Rx 端点启用批量/中断传输。
[13]	DMAReqEnab	0	R/W	CPU 将该位置 1 以启用 Rx 端点的 DMA 请求。
[12]	DisNyet（no use） PID Error	0	R/W	批量/中断事务：CPU 将该位置 1 以禁止发送 NYET 握手。 置位时，所有成功接收的 Rx 数据包都将被 ACK，包括在 FIFO 满时。 注：该位仅在高速模式下才有效，在该模式下应为所有中断端点设置该位。 ISO 事务处理：核心将该位置 1 以指示接收到的数据包中的 PID 错误。
[11]	DMAReqMode	0	R/W	CPU 将该位置 1 以选择 DMA 请求模式 1，并将其清除以选择 DMA 请求模式 0
[10-9]	Reserved	0	R	Reserved
[8]	IncomRx	0	R/W	如果由于未接收到部分数据而导致 Rx FIFO 中的数据包不完整，则在高带宽同步/中断传输中将该位置 1。 清除 RxPktRdy 时清除。 **注：在同步传输以外的任何方式下，该位将始终返回 0。**

[7]	ClrDataTog	0	R/W	CPU 将 1 写入该位以将端点数据触发器重置为 0。
[6]	SentStall	0	R/W	发送停止握手时，该位置 1。 CPU 应该清除该位。
[5]	SendStall	0	R/W	CPU 向该位写入 1 以发出 STALL 握手信号。 CPU 清除该位以终止停止条件。 **注：当端点用于同步传输时，此位无效**
[4]	FlushFIFO	0	R/W	CPU 向该位写入 1，以刷新要从端点 Rx FIFO 读取的下一个数据包。 FIFO 指针被复位并且 RxPktRdy 位（如下）被清除。 注意：仅当设置了 RxPktRdy 时，才应使用 FlushFIFO。 在其他时候，这可能会导致数据损坏。 另请注意，如果 FIFO 是双缓冲的，则可能需要将 FlushFIFO 设置两次以完全清除 FIFO。
[3]	DataError	0	R	如果数据包具有 CRC 或位填充错误，则在设置 RxPktRdy 时设置此位。 清除 RxPktRdy 时清除。 注意：仅当端点在 ISO 模式下运行时，此位才有效。 在批量模式下，它始终返回零。
[2]	OverRun	0	R/W	如果无法将 OUT 数据包加载到 Rx FIFO 中，则该位置 1。 CPU 应该清除该位。 **注：仅当端点在 ISO 模式下运行时，此位才有效。 在批量模式下，它始终返回零。**
[1]	FIFOFull	0	R	当没有更多的数据包可以加载到 Rx FIFO 中时，该位置 1
[0]	RxPktRdy	0	R/W	当接收到数据包时该位置位。 从 Rx FIFO 卸载数据包后，CPU 应清除该位。 当该位置 1 时，产生一个中断。

25.5.2.17 USB_RXCOUNT

- **Description**：Number of bytes to be read from RX endpoint FIFO
- **Size**：16bits
- **Offset**：0x18
- **Default**：0x0

BIT	Name	Reset	R/W	Description
[15-14]	Reserved	0	R	Reserved
[13-0]	RXCOUNT	0	R	RxCount 是一个 14 位只读寄存器，其中包含当前要从 Rx FIFO 中读取的行中数据包中的数据字节数。 如果数据包是作为多个批量数据包发送的，则给出的数字将用于组合数据包。

25.5.2.18 USB_TxType (No Use)

- **Description**：(Host Only) Sets the transaction protocol, speed and peripheral endpoint number for the host TX endpoint
- **Size**：8bits
- **Offset**：0x1A
- **Default**：0x0

25.5.2.19 USB_TxInterval (No Use)

- **Description**： (Host Only) Sets the polling interval for Interrupt / SOC transactions or the NAK response timeout on Bulk transactions for host TX endpoint
- **Size**：8bits
- **Offset**：0x1B
- **Default**：0x00

25.5.2.20 USB_RxType (No Use)

- **Description**：(Host Only) Sets the transaction protocol, speed and peripheral endpoint number for the host RX endpoint
- **Size**：8bits
- **Offset**：0x1C
- **Default**：0x00

25.5.2.21 USB_RxInterval (No Use)

- **Description**： (Host Only) Sets the polling interval for Interrupt / SOC transactions or the NAK response timeout on Bulk transactions for host RX endpoint
- **Size**：8bits
- **Offset**：0x1D
- **Default**：0x00

25.5.2.22 USB_FIFOSize

- **Description：** Returns the configured size of the selected RX FIFO and TX FIFOs
- **Size：** 8bits
- **Offset：** 0x1F
- **Default：** 0x00

BIT	Name	Reset	R/W	Description
[15-11]	Rx FIFO Size	0	R	返回与所选附加 TX / Rx 端点关联的 FIFO 的大小。 下半字节对所选 TX 端点 FIFO 的大小进行编码。 高半字节对所选 Rx 端点 FIFO 的大小进行编码。值 3 – 13 对应于 2n 字节（8 – 8192 字节）的 FIFO 大小。
[10-0]	Tx FIFO Size	0	R	

25.5.2.23 USB_FIFO0

- **Description：** FIFOs for Endpoints 0
- **Size：** 32bits
- **Offset：** 0x20
- **Default：** 0x00

BIT	Name	Reset	R/W	Description
[31-0]	FIFO0	0	R/W	端点 0 的 TXFIFO 和 RXFIFO

25.5.2.24 USB_FIFO1

- **Description：** FIFOs for Endpoints 1
- **Size：** 32bits
- **Offset：** 0x28
- **Default：** 0x00

BIT	Name	Reset	R/W	Description
[31-0]	FIFO2	0	R/W	端点 2 的 TXFIFO 和 RXFIFO

25.5.2.25 USB_FIFO2

- **Description：** FIFOs for Endpoints 2
- **Size：** 32bits
- **Offset：** 0x28
- **Default：** 0x00

BIT	Name	Reset	R/W	Description
[31-0]	FIFO2	0	R/W	端点 2 的 TXFIFO 和 RXFIFO

25.5.2.26 USB_DEVCTL

- **Description**：OTG device control register
- **Size**：32bits
- **Offset**：0x60
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[7]	B-Device	1	R	该只读位指示 USB 是作为“ A”设备还是“ B”设备运行。 0⇒“ A”设备； 1⇒“ B”设备。 仅在会话进行期间有效。 要确定没有会话正在进行时的角色，请设置会话位并读取该位。 注意：如果内核处于 Force_Host 模式（即会话已以 Testmode.D7 = 1 启动），则该位将指示来自 PHY 的 HOSTDISCON 输入信号的状态。
[6]	FSDev（no use）	0	R	当检测到全速或高速设备连接到端口时，此只读位置 1。 （通过在重置设备时检查高速 DPDM 来区分高速设备与全速设备。）仅在主机模式下有效
[5]	LSDev	0	R	当检测到低速设备连接到端口时，此只读位置 1。 仅在主机模式下有效。
[4-3]	Vbus	0	R	这些只读位将当前的 VBus 级别编码如下： **数值** / **描述** **00** / Below SessionEnd **01** / Above SessionEnd,Below AValid **10** / Above AValid,Below VBusValid **11** / Above VBusValid
[2]	Host Mode	0	R	当 USB 充当主机时，此只读位置 1
[1]	Host Req	0	R/W	设置后，进入挂起模式时， USB 将启动主机协商。 主机协商完成后清除。（仅适用于“ B”设备）
[0]	Session	0	R/W	当作为“ A”设备运行时，CPU 将该位置 1 或清除该位以开始或结束会话。 当作为“ B”设备运行时，会话开始/结束时， USB 会将其设置/清除。它也由 CPU 设置为启动会话请求协议。 当 USB 处于挂起模式时，该位可以由 CPU 清除以执行软件断开连接。 注意：当 Core 未挂起时清除该位将导致不确定的行为。

25.5.2.27 USB_MISC

- **Description**：Miscellaneous Register
- **Size**：8bits
- **Offset**：0x61
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[7-2]	Reserved	0	R	Reserved
[1]	Tx_edma	0	R/W	1'b0：将所有 IN 端点的 DMA_REQ 信号置为无效时，将 MAXP 字节写入端点。 这是延迟模式。 1'b1：将所有 IN 端点的 DMA_REQ 信号置为无效时，将 MAXP-8 字节写入端点。 这是早期模式。
[0]	Rx_edma	0	R/W	1'b0：将所有 OUT 端点的 DMA_REQ 信号置为无效时，已将 MAXP 字节读取到端点。 这是延迟模式。 1'b1：将所有 OUT 端点的 DMA_REQ 信号置为无效时，已将 MAXP-8 字节读取到端点。 这是早期模式。

25.5.2.28 USB_TxFIFOsz (No Use)

- **Description**：TX Endpoint FIFO size (no use)
- **Size**：8bits
- **Offset**：0x62
- **Default**：0x00

25.5.2.29 USB_RxFIFOsz (No Use)

- **Description**：RX Endpoint FIFO size (no use)
- **Size**：8bits
- **Offset**：0x63
- **Default**：0x00

25.5.2.30 USB_TxFIFOadd (No Use)

- **Description**：TX Endpoint FIFO address (no use)
- **Size**：16bits
- **Offset**：0x64
- **Default**：0x00

25.5.2.31 USB_RxFIFOadd (No Use)

- **Description**：RX Endpoint FIFO address (no use)
- **Size**：16bits
- **Offset**：0x66
- **Default**：0x00

25.5.2.32 USB_HWVers

- **Description**：Hardware Version Number Register
- **Size**：16bits
- **Offset**：0x6C
- **Default**：0x800

BIT	Name	Reset	R/W	Description
[15]	RC	0	R	如果 RTL 是从发布候选版本而不是内核的完整版本使用的，则设置为“ 1”。
[14-10]	xx	0x2	R	主要版本号（范围 0 – 31）
[9-0]	yyy	0	R	次要版本号（范围 0 – 999）。

25.5.2.33 USB_EPInfo

- **Description：** Information about numbers of TX and RX endpoints
- **Size：** 8bits
- **Offset：** 0x78
- **Default：** 0x22

BIT	Name	Reset	R/W	Description
[7-4]	RxEndPoints	0x2	R	设计中实现的 Rx 端点数。
[3-0]	TxEndPoints	0x2	R	设计中实现的 Tx 端点数。

25.5.2.34 USB_RAMInfo

- **Description：** Information about the width of the RAM and the number of DMA channels
- **Size：** 8bits
- **Offset：** 0x79
- **Default：** 0x26

BIT	Name	Reset	R/W	Description
[7-4]	DMAChan	2	R	设计中实现的 DMA 通道数。
[3-0]	RamBits	0x6	R	RAM 地址总线的宽度

25.5.2.35 USB_LINKInfo

- **Description：** Information about delays to be applied
- **Size：** 8bits
- **Offset：** 0x7A
- **Default：** 0x5c

BIT	Name	Reset	R/W	Description
[7-4]	WTCON	0x5	R	设置要应用的等待时间，以允许用户连接/断开过滤器，以 533.3ns 为单位。（默认设置对应于 2.667μs。）
[3-0]	WTID	0xc	R	设置从声明 IDPULLUP 到认为有效的 IDDIG 所应用的延迟，以 4.369ms 为单位。 （默认设置对应于 52.43ms。）

25.5.2.36 USB_VPLEN

- **Description：** Duration of the VBus pulsing charge
- **Size：** 8bits
- **Offset：** 0x7B
- **Default：** 0x3c

BIT	Name	Reset	R/W	Description
[7-0]	VPLEN	0x3c	R	以 546.1 μs 为单位设置 VBus 脉冲充电的持续时间。 （默认设置对应于 32.77ms。）

25.5.2.37 USB_HS_EOF1

- **Description：** Time buffer available on High-Speed transactions
- **Size：** 8bits
- **Offset：** 0x7C
- **Default：** 0x80

BIT	Name	Reset	R/W	Description
[7-0]	HS_EOF1	0x80	R	为 EOF 停止开始新事务之前的时间设置高速事务，以 133.3ns 为单位。 （默认设置对应于 17.07μs。）

25.5.2.38 USB_FS_EOF1

- **Description：** Time buffer available on Full-Speed transactions
- **Size：** 8bits
- **Offset：** 0x7D
- **Default：** 0x77

BIT	Name	Reset	R/W	Description
[7-0]	FS_EOF1	0x77	R	为全速事务设置 EOF 停止开始新事务之前的时间，单位为 533.3ns。（默认设置对应于 63.46μs。）

25.5.2.39 USB_LS_EOF1

- **Description：** Time buffer available on Low-Speed transactions
- **Size：** 8bits
- **Offset：** 0x7E
- **Default：** 0x72

BIT	Name	Reset	R/W	Description
[7-0]	LS_EOF1	0x72	R	为低速事务设置 EOF 停止开始新事务之前的时间，单位为 1.067μs。（默认设置对应于 121.6μs。）

25.5.2.40 USB_SOFT_RST

- **Description：** Soft Reset
- **Size：** 8bits
- **Offset：** 0x7F
- **Default：** 0x00

BIT	Name	Reset	R/W	Description
[7-2]	Reserved	0	R	Reserved
[1]	NRSTX	0	R/W	该位的默认值为 1'b0； 向该位写入 1 时，将在 CLK 输入的 7 个周期的最小延迟内将输出 NRSTXO 置为有效（低电平）。 相对于 XCLK，输出 NRSTXO 将被异步置为有效，并被同步取消置为有效。该寄存器是自清除的，将通过输入 NRST 复位。
[0]	NRST	0	R/W	该位的默认值为 1'b0； 向该位写入 1 时，将在 CLK 输入的 7 个周期的最小延迟内将输出 NRSTO 置为有效（低电平）。相对于 CLK，输出 NRSTO 将被异步置为有效，同时同步为无效。 该寄存器是自清除的，将通过输入 NRST 复位。

25.5.2.41 USB_DMA_INTR

- **Description：**
- **Size：** 8bits
- **Offset：** 0x200
- **Default：** 0x00

BIT	Name	Reset	R/W	Description
[7-1]	Reserved	0	R	Reserved
[1]	DMA_INTR2	0	R/W	该寄存器为 DMA 通道 2 提供一个中断。 读取时清除此中断寄存器。
[0]	DMA_INTR1	0	R/W	该寄存器为 DMA 通道 1 提供一个中断。 读取时清除此中断寄存器。

25.5.2.42 USB_DMA_CNTL

- **Description：**
- **Size：**16bits
- **Offset：**0x204
- **Default：**0x00

BIT	Name	Reset	R/W	Description
[15-11]	Reserved	0	R	Reserved
[10-9]	DMA_BRSTM	0	R/W	突发模式 00 =突发模式 0：未指定长度的突发传输 01 =突发模式 1：INCR4 或未指定长度 10 =突发模式 2：INCR8，INCR4 或未指定长度 11 =突发模式 3：INCR16，INCR8，INCR4 或未指定长度
[8]	DMA_ERR	0	R/W	总线错误位。 指示在输入 AHB_HRESPM [1：0]上观察到总线错误。 该位由软件清除。
[7-4]	DMAEP	0	R/W	此通道分配给的端点号。
[3]	DMAIE	0	R/W	DMA 中断使能
[2]	DMAMODE	0	R/W	该位选择 DMA 传输模式。 0 = DMA 模式 0 传输 1 = DMA 模式 1 传输
[1]	DMA_DIR	0	R/W	该位选择 DMA 传输方向。 0 = DMA 写（RX 端点） 1 = DMA 读（TX 端点）
[0]	DMA_EN	0	R/W	该位使能 DMA 传输，并将导致传输开始。

25.5.2.43 USB_DMA_ADDR

- **Description：**
- **Size：**32bits
- **Offset：**0x208
- **Default：**0x00

BIT	Name	Reset	R/W	Description
[31-0]	DMA_ADDR	0	R/W	DMA 通道 2 内存地址。 **注意：写入该寄存器的初始内存地址必须具有一个值，以使其模 4 值等于 0。也就是说，DMA_ADDR [1：0]必须等于 2'b00。 该寄存器的低两位是只读的，不能由软件设置。**

25.5.2.44 USB_DMA_COUNT

- **Description：**
- **Size：**32bits
- **Offset：**0x20C
- **Default：**0x00

BIT	Name	Reset	R/W	Description
[31-0]	DMA_COUNT	0	R/W	该寄存器标识通道 2 传输的当前 DMA 计数。 软件将设置传输的初始计数，以标识整个传输长度。 随着计数的进行，此计数随着字节的传输而递减。

25.5.2.45 USB_DMA_CNTL2

- **Description：**
- **Size：**16bits
- **Offset：**0x214
- **Default：**0x00

BIT	Name	Reset	R/W	Description
[15-11]	Reserved	0	R	Reserved
[10-9]	DMA_BRSTM	0	R/W	突发模式 00 =突发模式 0：未指定长度的突发传输 01 =突发模式 1：INCR4 或未指定长度 10 =突发模式 2：INCR8，INCR4 或未指定长度 11 =突发模式 3：INCR16，INCR8，INCR4 或未指定长度
[8]	DMA_ERR	0	R/W	总线错误位。 指示在输入 AHB_HRESPM [1：0]上观察到总线错误。 该位由软件清除。
[7-4]	DMAEP	0	R/W	此通道分配给的端点号。
[3]	DMAIE	0	R/W	DMA 中断使能
[2]	DMAMODE	0	R/W	该位选择 DMA 传输模式。 0 = DMA 模式 0 传输 1 = DMA 模式 1 传输
[1]	DMA_DIR	0	R/W	该位选择 DMA 传输方向。 0 = DMA 写（RX 端点） 1 = DMA 读（TX 端点）
[0]	DMA_EN	0	R/W	该位使能 DMA 传输，并将导致传输开始。

25.5.2.46 USB_DMA_ADDR2

- **Description：**
- **Size：**32bits
- **Offset：**0x218
- **Default：**0x00

BIT	Name	Reset	R/W	Description
[31-0]	DMA_ADDR2	0	R/W	DMA 通道 2 内存地址。 **注意：写入该寄存器的初始内存地址必须具有一个值，以使其模 4 值等于 0。也就是说，DMA_ADDR [1：0]必须等于 2'b00。 该寄存器的低两位是只读的，不能由软件设置。**

25.5.2.47 USB_DMA_COUNT2

- **Description：**
- **Size：**32bits
- **Offset：**0x21C
- **Default：**0x00

BIT	Name	Reset	R/W	Description
[31-0]	DMA_COUNT2	0	R/W	该寄存器标识通道 2 传输的当前 DMA 计数。 软件将设置传输的初始计数，以标识整个传输长度。 随着计数的进行，此计数随着字节的传输而递减。

25.5.2.48 USB_RqPktCount1 (No Use)

- **Description：**Number of requested packets for Receive Endpoint 1
- **Size：**16bits
- **Offset：**0x304
- **Default：**0x00

25.5.2.49 USB_RqPktCount2 (No Use)

- **Description：**Number of requested packets for Receive Endpoint 2
- **Size：**16bits
- **Offset：**0x308
- **Default：**0x00

25.5.2.50 USB_RxDPktBufDis

- **Description**：Double Packet Buffer Disable register for RX Endpoints 1 to 15
- **Size**：16bits
- **Offset**：0x340
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[15-3]	Reserved	0	R	Reserved
[2]	EP2 RxDis	0	R	端点 2 的 Rx Double Packet Buffer 禁用
[1]	EP1 RxDis	0	R	端点 1 的 Rx Double Packet Buffer 禁用
[0]	Reserved	0	R	Reserved

25.5.2.51 USB_TxDPktBufDis

- **Description**：Double Packet Buffer Disable register for TX Endpoints 1 to 15
- **Size**：16bits
- **Offset**：0x342
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[15-3]	Reserved	0	R	Reserved
[2]	EP2 TxDis	0	R	端点 2 的 Tx Double Packet Buffer 禁用
[1]	EP1 TxDis	0	R	端点 1 的 Tx Double Packet Buffer 禁用
[0]	Reserved	0	R	Reserved

25.5.2.52 USB_C_T_UCH

- **Description**：This register sets the Chirp Timeout Timer
- **Size**：16bits
- **Offset**：0x344
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[15-0]	C_T_UCH	0x4074	R	该寄存器设置线性调频超时。 该数字乘以 4 表示发生超时之前的 XCLK 周期数。 也就是说，如果 XCLK 为 30MHz，则该数字表示发生超时之前的 133ns 时间间隔数。 如果 XCLK 为 60MHz，则此数字表示发生超时之前 67ns 的时间间隔数。

25.5.2.53 USB_C_T_HHSRTN

- **Description**：This register sets the delay from the end of High Speed resume signaling to enable UTM normal operating mode
- **Size**：16bits
- **Offset**：0x346
- **Default**：0x5e6

BIT	Name	Reset	R/W	Description
[15-0]	C_T_ HHSRTN	0x5e6	R	从高速恢复信号结束到启用 UTM 正常操作模式的延迟。如果主机 PHY 数据宽度为 16 位（XCLK 为 30MHz），则默认值为 2F3h;如果 PHY 数据宽度为 8 位（XCLK 为 60Mhz），则默认值为 5E6h，对应于 100us 的延迟

25.5.2.54 USB_LPM_ATTR

- **Description**：LPM Attribute Register
- **Size**：16bits
- **Offset**：0x360
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[15-12]	EndPoint	0	R	这是 LPM 事务的令牌包中的 EndPnt。
[11-9]	Reserved	0	R	Reserved
[8]	RmtWak	0	R	该位是远程唤醒使能位。 RmtWak = 1’b0：未启用远程唤醒。 RmtWak = 1’b1：启用远程唤醒。 该位仅在临时的基础上应用，并且仅适用于当前的挂起状态。 在当前的挂起周期之后，枚举协商的远程唤醒功能将适用
[7-4]	HIRD	0	R	这是主机启动的恢复持续时间。 该值是主机在总线上驱动恢复的最短时间。 该寄存器中的值对应于实际的恢复时间： 恢复时间= 50us + HIRD * 75us。 结果为 50us 至 1200us
[3-0]	LinkState	0	R	该值由主机提供给外围设备，以指示外围设备在接收和接受 LPM 事务后必须转变为什么状态。 LinkState = 4’h0-保留 LinkState = 4’h1-睡眠状态（L1） LinkState = 4’h2-保留 LinkState = 4’h3-保留

25.5.2.55 USB_LPM_CNTRL

- **Description**：LPM Control Register
- **Size**：8bits
- **Offset**：0x362
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[7-5]	Reserved	0	R	Reserved
[4]	LPMNAK	0	R	该位用于将所有端点置于某种状态，以使对除 LPM 事务以外的所有事务的响应为 NAK。 仅在 USB LPM 挂起后，此位才生效。 在这种情况下，USB 将继续为 NAK，直到该位被软件清除为止。
[3-2]	LPMEN	0	R	该寄存器用于启用 MUSBMHDRC 中的 LPM。 可以在三个级别启用 LPM，这三个级别将确定 MUSBMHDRC 对 LPM 事务的响应。 这三个级别是： 2’b00、2’b10 - 不支持 LPM 和扩展交易。 在这种情况下，USB 将不响应 LPM 事务，并且事务将超时 •2’b01 –不支持 LPM，但支持扩展事务。 在这种情况下，USB 将使用 STALL 响应 LPM 事务。 •2’b11 – USB 支持 LPM 扩展事务。 在这种情况下，USB 将响应由 LPMXMT 的值和其他条件确定的 NYET 或 ACK。
[1]	LPMRES	0	R	该位由软件用来启动恢复（远程唤醒）。 该位与 POWER 寄存器中的传统 RESUME 位（地址 0x01.2）的不同之处在于，RESUME 信号的时序由硬件控制。 当软件写入该位时，恢复信号将保持 50us。 这一点是自我清除。
[0]	LPMXMT	0	R	该位由软件置 1，以指示 USB 在接收到下一个 LPM 事务时转换为 L1 状态。 仅当 LPMEN 设置为 2’b11 时，此位才有效。 可以在与 LPMEN 相同的周期中设置该位。 如果将此位设置为 1’b1 且 LPMEN = 2’b11，则 USB 可以通过以下方式进行响应： •如果没有数据待处理（所有 TX FIFO 为空），则 USB 将以 ACK 响应。 在这种情况下，该位将自动清零，并且将产生软件中断。 •如果数据待处理（数据驻留在至少一个 TX FIFO 中），则 USB 将以 NYET 进行响应。 在这种情况下，该位不会自动清零，但是会产生软件中断。

25.5.2.56 USB_LPM_INTREN

- **Description**：LPM Interrupt Enable Register
- **Size**：8bits
- **Offset**：0x363
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[7-6]	Reserved	0	R	Reserved
[5]	LPMERREN	0	R	0：禁用 LPMERR 中断 1：启用 LPMERR 中断
[4]	LPMRESEN	0	R	0：禁用 LPMRES 中断 1：启用 LPMRES 中断
[3]	LPMNCEN	1	R/W	0：禁用 LPMNC 中断 1：启用 LPMNC 中断
[2]	LPMACKEN	0	R/W	0：禁用 LPMACK 中断 1：启用 LPMACK 中断
[1]	LPMNYEN	1	R/W	0：禁用 LPMNY 中断 1：启用 LPMNY 中断
[0]	LPMSTEN	1	R/W	0：禁用 LPMST 中断 1：启用 LPMST 中断

25.5.2.57 USB_LPM_INTR

- **Description**：LPM Interrupt Register
- **Size**：8bits
- **Offset**：0x364
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[7-6]	Reserved	0	R	Reserved
[5]	LPMERR	0	R	如果收到的 LPM 事务具有不支持的 LinkState 字段，则设置此位。在这种情况下，对事务的响应将是 STALL。 但是，将更新 LPM_ATTR 寄存器，以便软件可以观察到不兼容的 LPM 数据包有效负载
[4]	LPMRES	0	R	如果由于任何原因恢复了 USB，则该位置 1。 该位与 POWER 寄存器的 RESUME 位（地址 0x01）互斥。
[3]	LPMNC	1	R	当接收到 LPM 事务并且由于 RX FIFO 中有待处理的数据而使 USB 响应 NYET 时，该位置 1。 这只能在以下情况下发生： •LMRESP 寄存器中的 LPMRESP 字段设置为 2’b11，LPMXMT 字段设置为 1’b1，并且 USB TX FIFO 中有待处理的数据。
[2]	LPMACK	0	R	当接收到 LPM 事务并且 USB 响应 ACK 时，该位置 1。 这只能在以下情况下发生： •LMRESP 寄存器中的 LPMRESP 字段设置为 2’b11，LPMXMT 字段设置为 1’b1，并且 USB TX FIFO 中没有待处理的数据
[1]	LPMNY	1	R	当接收到 LPM 事务并且 USB 响应 NYET 时，该位置 1。 这只能在以下情况下发生： •LMRESP 寄存器中的 LPMRESP 字段设置为 2’b11，而 LPMXMT 字段设置为 1’b0。
[0]	LPMST	1	R	当接收到 LPM 事务并且 MUSBMDHRC 响应 STALL 时，该位置 1。这只能在以下情况下发生： •LMRESP 寄存器中的 LPMRESP 字段设置为 2’b01。

25.5.2.58 USB_LPM_FADDR

- **Description**：LPM Function Address Register
- **Size**：8bits
- **Offset**：0x365
- **Default**：0x00

BIT	Name	Reset	R/W	Description
[7]	Reserved	0	R	Reserved
[6-0]	LPM_ADDR	0	R/W	LPM 功能地址

25.5.2.59 USB_UCFG0

- **Description：**
- **Size：**32bits
- **Offset：**0x400
- **Default：**0x55445504

BIT	Name	Reset	R/W	Description
[31]	DM_OE	0	R/W	DM 输出使能 1：enable 0：disable
[30]	DM_OE_EN	1	R/W	DM 输出使能 1：使用 bit[31] 0：使用硬件生成的 DM_OE
[29]	DP_OE	0	R/W	DP 输出使能 1：enable 0：disable
[28]	DP_OE_EN	1	R/W	DP 输出使能 1：使用 bit[29] 0：使用硬件生成的 DP_OE
[27]	DM_IE	0	R/W	DM 输入使能 1：enable 0：disable
[26]	DM_IE_EN	1	R/W	DM 输入使能 1：使用 bit[27] 0：使用硬件生成的 DM_IE
[25]	DP_IE	0	R/W	DP 输入使能 1：enable 0：disable
[24]	DP_IE_EN	1	R/W	DP 输入使能 1：使用 bit[25] 0：使用硬件生成的 DP_IE
[23]	DMSR	0	R/W	DM PAD 输出 slewrate 增强 1：enable 0：disable
[22-20]	DMTRIM	3’100	R/W	DM 端电阻修调 3'b000：MIN …… 3'b111：MAX
[19]	DPSR	0	R/W	DP PAD 输出 slewrate 增强 1：enable 0：disable

[18-16]	DPTRIM	3’100	R/W	DP 端电阻修调 3'b000：MIN …… 3'b111：MAX
[15]	DMPD	0	R/W	DM 下拉使能 1：enable 0：disable
[14]	DMPD_EN	1	R/W	DM 下拉使能 1：使用 bit[15] 0：使用硬件生成的 DMPD
[13]	DMPU	0	R/W	DM 上拉使能 1：enable 0：disable
[12]	DMPU_EN	1	R/W	DM 上拉使能 1：使用 bit[13] 0：使用硬件生成的 DMPU
[11]	DPPD	0	R/W	DP 下拉使能 1：enable 0：disable
[10]	DPPD_EN	1	R/W	DP 下拉使能 1：使用 bit[11] 0：使用硬件生成的 DPPD
[9]	DPPU	0	R/W	DP 上拉使能 1：enable 0：disable
[8]	DPPU_EN	1	R/W	DP 上拉使能 1：使用 bit[9] 0：使用硬件生成的 DPPU
[7]	Reserved	0	R/W	Reserved
[6]	Test_En	0	R/W	设置为 1 时开启 USB 测试模式，DPDM 会持续输出 6MHz 的时钟
[5]	VBUSVALID	0	R/W	Vbus above VBusValid threshold
[4]	AVALID	0	R/W	Vbus above Vbus A-device session threshold
[3]	VBUSLO	0	R/W	Vbus above session end threshold
[2]	VID	1	R/W	Mini-AB connector ID pin
[1]	TM1	0	R/W	Test Mode
[0]	PHY_EN	0	R/W	PHY enable

25.5.2.60 USB_UCFG1

- **Description：**
- **Size：**32bits
- **Offset：**0x404
- **Default：**0x00

BIT	Name	Reset	R/W	Description
[31-19]	Reserved	0	R/W	Reserved
[18]	lpmerr_mask	0	R/W	中断使能
[17]	lpmres_mask	0	R/W	0：发送到 CPU
[16]	lpmnc_mask	0	R/W	1：不发送到 CPU
[15]	lpmack_mask	0	R/W	
[14]	lpmny_mask	0	R/W	
[13]	lpmst_mask	0	R/W	
[12]	ep2_rxint_mask	0	R/W	
[11]	ep1_rxint_mask	0	R/W	
[10]	ep2_txint_mask	0	R/W	
[9]	ep1_txint_mask	0	R/W	
[8]	ep0_int_mask	0	R/W	
[7]	vbuserr_mask	0	R/W	
[6]	sessreq_mask	0	R/W	
[5]	diconn_mask	0	R/W	
[4]	conn_mask	0	R/W	
[3]	sof_mask	0	R/W	
[2]	babble_reset_mask	0	R/W	
[1]	resume_mask	0	R/W	
[0]	suspend_mask	0	R/W	

25.5.2.61 USB_UCFG2

- **Description：**
- **Size：**32bits
- **Offset：**0x408
- **Default：**0x00

BIT	Name	Reset	R/W	Description
[31-30]	Usbsendstate	0	R	00:reserved 01:setup 10:out 11:in
[29-27]	Reserved	0	R/W	Reserved
[26]	InState	0	R	收到 IN 包
[25]	OutState	0	R	收到 OUT 包
[24]	SetupState	0	R	收到 Setup 包
[23]	DPOE			
[22]	DMOE			
[21]	DPIE			
[20]	DMIE			
[19]	DIP			
[18]	DIM			
[17]	DOP			
[16]	DOM			
[15-4]	Debouce_Max	0x40	R/W	去抖动最大计数值
[3]	DISCONN_INT_EN	0	R/W	USB 拔出检测中断使能
[2]	DISCONN_INT	0	R/W	USB 拔出检测中断 1：有 usb 插入 0：无 usb 插入 写 1 清 0；
[1]	DET_INT_EN	0	R/W	USB 插入检测中断使能
[0]	DET_INT	0	R/W	USB 插入检测中断 1：有 usb 插入 0：无 usb 插入 写 1 清 0；

26 数字可寻址照明接口（DALI）

26.1 简介

DALI(Digital Addressable Lighting Interface)即数字可寻址照明接口，是一种数据传输协议，它定义了电子镇流器与设备控制器的通讯方式。

DALI 通讯模块是为可控电子镇流器设计的串行通信电路，镇流器是电路拓扑的统称，其为荧光灯、汞灯或其他电子灯提供所需的启动电压及工作电流。

26.2 结构框图

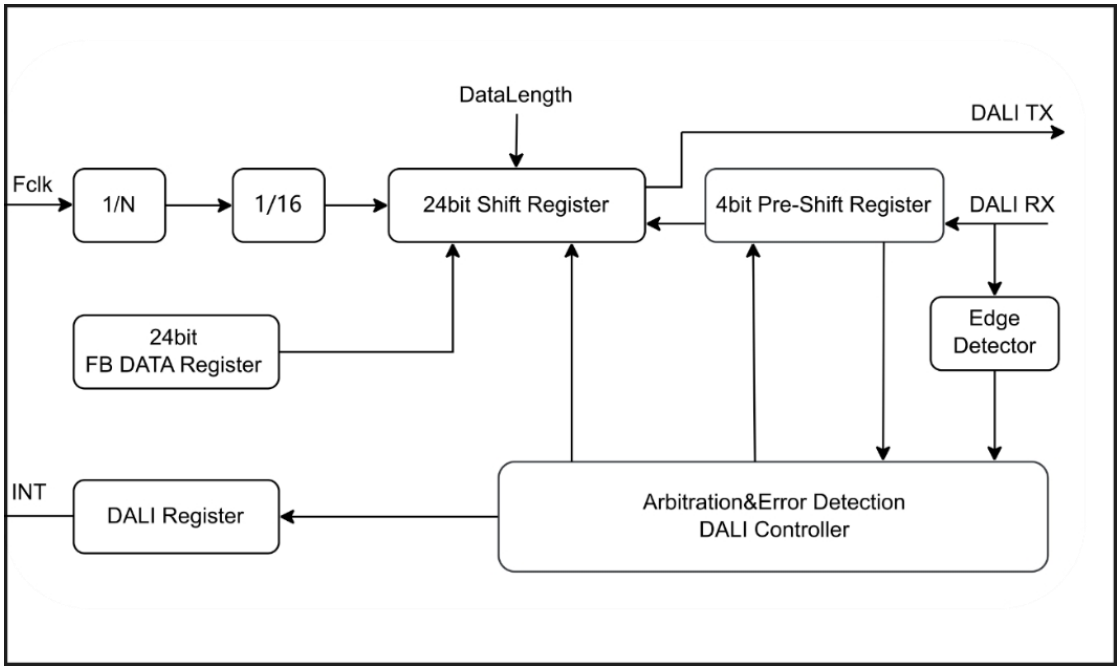


图 26-1 DALI 结构框图

26.3 主要特性

DALI 主要具有以下特性：

- 支持主模式和从模式
- 支持曼彻斯特编码及解码
- 支持数据位 16/17/18/24Bit
- 支持 1.2K/2.4K/4.8K 波特率
- 支持前向帧及后向帧延迟参数可配
- 支持滤波，滤波长度可配
- 支持波特率容错，容错机制在每次跳变沿更新
- 支持 RX、TX 极性可配

26.4 功能描述

26.4.1 DALI 传输协议

DALI 协议采用曼彻斯特编码数据格式，一帧的所有位除了两个停止位之外，其他所有 bit 位都是二相编码。其中默认标准传输波特率为 1.2KHz，两相位周期为 833.33us±10%。

前向帧及后向帧在输出过程中，如果某一个帧出现编码错误，则该帧应在出现编码错误后被忽略，需等待该帧传输完毕才能再次回到接收帧的准备状态。

26.4.2 DALI 前/后向帧

前向帧可由 19/20/21/27 个二相编码位组成，具体取决于数据长度配置位：

- 1 个起始位(0->1：逻辑 1)
- 根据消息长度，数据位长度为 16/17/18/24 位
- 2 个高位停止位(相位不变)

后向帧由 11 个双相位编码为组成：

- 1 个起始位(0->1：逻辑 1)
- 1 个数据字节(8 位数据)
- 2 个高位停止位(相位不变)

DALI 传输帧格式及传输序列如图 26-2 所示(极性低有效，POL=1)。

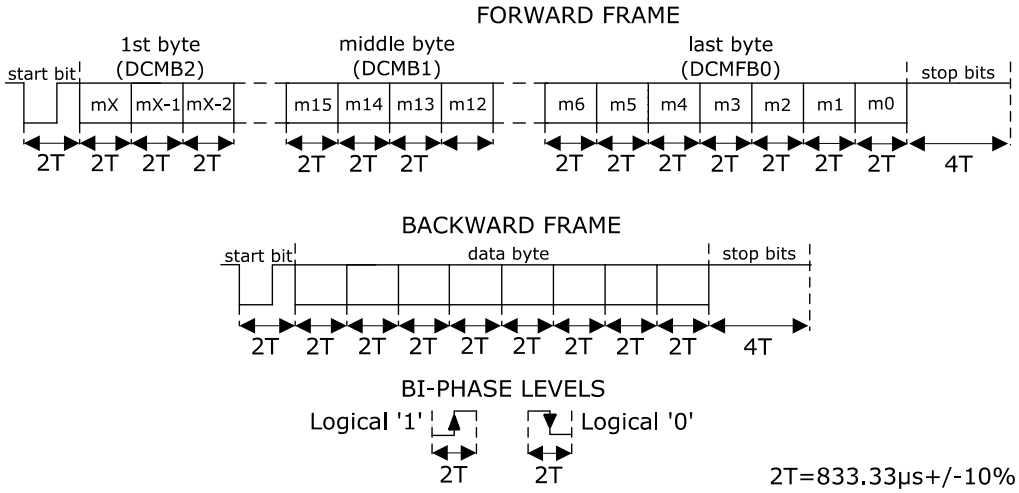


图 26-2 DALI 传输帧格式及传输序

26.4.3 DALI 波特率

DALI 起始位及信息位均采用二相编码，其中标准波特率为 1.2Khz，曼彻斯特二相编码电平“0”和“1”所对应波形跳变如图 26-3(1.2K 波特率，极性低有效，POL=1)。

图 26-3 二相电平“1”和“0”所用符号

T = 1 / (2 × 1200 bps) = 416.67us

DALI 波特率容错范围为±%10，则时间 T 限值为：334us < T < 500us。

前向帧定时

以数据长度 16bit（ML[1:0] = 00），极性低有效(POL=1)，波特率 1.2K 为例：

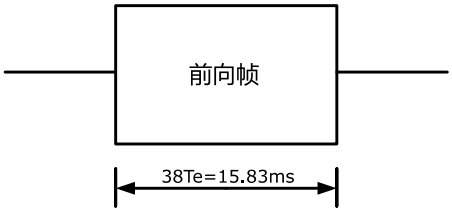


图 26-4 前向帧

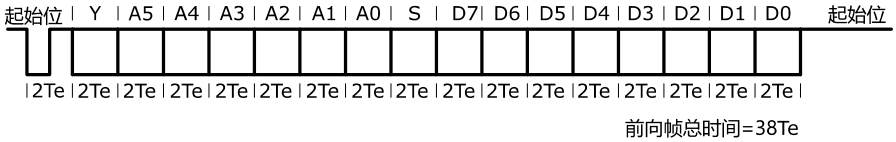


图 26-5 前向帧定时

后向帧定时

以极性低有效(POL=1)，波特率 1.2K 为例：

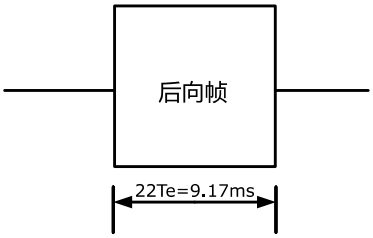


图 26-6 后向帧

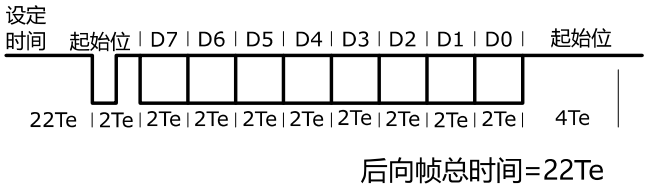


图 26-7 后向帧定时

帧流程定时

帧延迟时间定义：当前帧的最后一个停止位结束后，到下一帧的起始位之前的延迟时间

- 两个连续前向帧之间的延迟时间至少为 22T；
- 前向帧和后向帧之间的延迟时间至少为 7T-22T；
- 后向帧和前向帧之间的延迟时间至少为 22T；

图 26-8 为帧流程实例图。

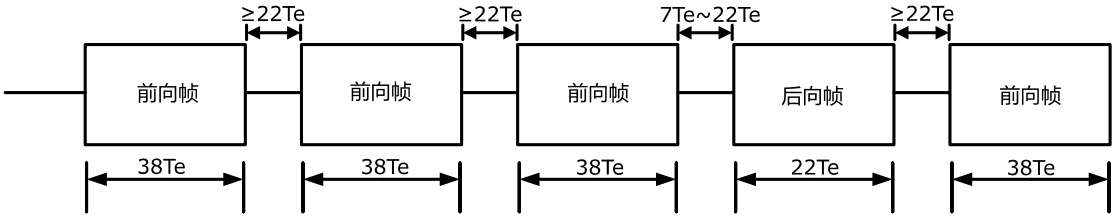


图 26-8 帧流程定时示例

图 26-9 为前向帧到后向帧的过渡。

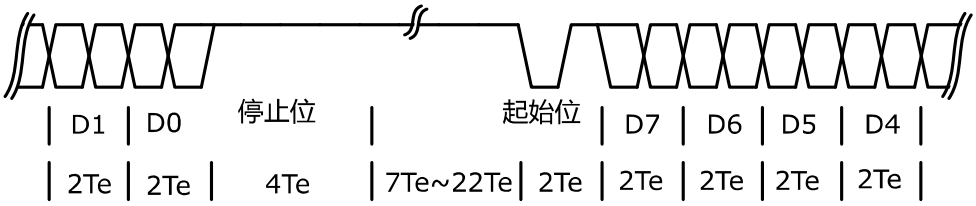


图 26-9 前向帧到后向帧的过渡

图 26-10 为后向帧到前向帧和前向帧到前向帧的过渡。

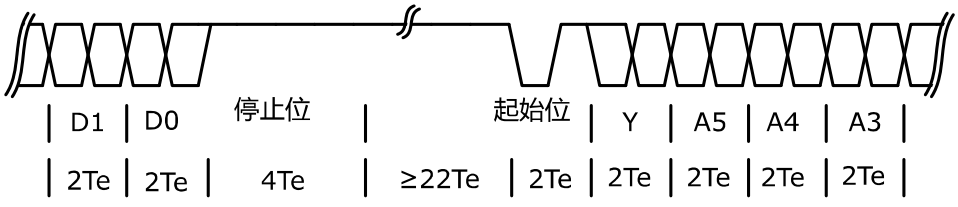


图 26-10 后向帧到前向帧和前向帧到前向帧的过渡

26.4.4 DALI 接口信号

DALI RX/TX 为复用功能 I/O，可根据产品 Datasheet 中 I/O 复用描述进行对应的配置。

DALI RX/TX 信号线的极性可根据实际 DALI 协议进行调整，但是复位后的默认状态依据 DALI 标准协议，其中发送和接收信号线在空闲状态下均为 1。

26.4.5 DALI 中断

DALI 模块可在前向/后向帧传输完成、解码/时序错误时产生中断。可灵活通过设置寄存器内相应的位打开/检查相应的中断事件。

中断事件	中断使能	中断标志
后向帧解码/时序错误	BEIE	BEIF
前向帧解码/时序错误	FEIE	FEIF
后向帧传输完成	BDIE	BDIF
前向帧传输完成	FDIE	FDIF

26.5 寄存器描述

26.5.1 寄存器列表

Name	Offset	Width	Description
DALI_CR	0x00	32bits	DALI 控制寄存器
DALI_ISR	0x04	32bits	DALI 中断状态寄存器
DALI_FDR	0x08	32bits	DALI 前向帧数据寄存器
DALI_BDR	0x0C	32bits	DALI 后向帧数据寄存器
DALI_FCR	0x10	32bits	DALI 滤波器控制寄存器
DALI_PSCR	0x14	32bits	DALI 预分频寄存器
DALI_TCR	0x18	32bits	DALI 定时控制寄存器

26.5.2 寄存器详细描述

26.5.2.1 DALI 控制寄存器 (DALI_CR)

- **Name**：DALI Control Register
- **Size**：32bits
- **Offset**：0x00
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[11]	BEIE	R/W	0x0	后向帧错误中断使能 (DALI Backward Error Interrupt Enable) 1：使能 0：不使能
[10]	FEIE	R/W	0x0	前向帧错误中断使能 (DALI Forward Error Interrupt Enable) 1：Enable 0：Disable
[9]	BDIE	R/W	0x0	后向帧完成中断使能 (DALI Backward Done Interrupt Enable) 1：使能 0：不使能
[8]	FDIE	R/W	0x0	前向帧完成中断使能 (DALI Forward Done Interrupt Enable) 1：使能 0：不使能
[7-6]	Reserved	Reserved	Reserved	Reserved
[5-4]	ML	R/W	0x0	DALI 帧长度 (DALI Message Length)：用于定义前向帧的长度。 00：16 位 01：17 位 10：18 位 11：24 位 注意：只用于定义前向帧长度，后向帧长度固定为 8 位。
[3]	POL	R/W	0x0	DALI 极性选择 (DALI Polarity Select) 1：低有效 0：高有效
[2]	TS	R/WAC	0x0	DALI 传输起始 (DALI Transmit Start) 1：传输起始 0：无影响 注意： 作为主模式时(MODE=1)，发送前向帧， 作为从模式时(MODE =0)，发送后向帧。
[1]	MODE	R/W	0x0	DALI 模式选择 (DALI Mode Selection) 1：Master Mode 0：Slave Mode 注意：DALI 使能后不可再改变模式。

[0]	PEN	R/W	0x0	DALI 外设使能控制 (DALI Peripheral Enable) 1：使能 0：不使能 注意：如果中途 Disable，硬件会自动清除 Couter 及 FSM。

26.5.2.2 DALI 中断状态寄存器 (DALI_ISR)

- **Name**：DALI Interrupt Status Register
- **Size**：32bits
- **Offset**：0x04
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-5]	Reserved	Reserved	Reserved	Reserved
[4]	BSY	R	0x0	DALI 忙状态 (DALI Busy Status) 0：空闲状态 1：忙状态 注意：DALI 模块忙状态时，不发送发起下次传输。
[3]	BEIF	R/W1C	0x0	DALI 后向帧错误中断标志位 (DALI Backward Error Interrupt Flag) 针对 Master 接收后向帧出现解码错误。 0：无错误 1：后向帧错误标志挂起 注意： 1、作为主机时，后向帧错误指的是接收解码中出现解码错误； 2、作为从机时，后向帧错误指的是传输中再次发起后向帧操作。
[2]	FEIF	R/W1C	0x0	前向帧错误中断标志位 (DALI Forward Error Interrupt Flag) 0：无错误 1：前向帧错误标志挂起 注意： 1、作为主机时，前向帧错误指的是传输中再次发起前向帧操作； 2、作为从机时，前向帧错误指的是接收解码中出现解码错误。
[1]	BDIF	R/W1C	0x0	后向帧完成中断标志位 (DALI Backward Done Interrupt Flag) 0：未完成 1：传输完成 注意： 1、作为主机时，后向帧完成指的是后向帧**接收**完成； 2、作为从机时，后向帧完成指的是后向帧**发送**完成。
[0]	FDIF	R/W1C	0x0	前向帧完成中断标志位 (DALI Forward Done Interrupt Flag) 0：未完成 1：传输完成 注意： 1、作为主机时，前向帧完成指的是前向帧**发送**完成； 2、作为从机时，前向帧完成指的是前向帧**接收**完成。

26.5.2.3 DALI 前向帧数据寄存器 (DALI_FDR)

- **Name**：DALI Forward Data Register
- **Size**：32bits
- **Offset**：0x08
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-24]	Reserved	Reserved	Reserved	Reserved
[23-0]	FDR	R/W	0x0	前向帧数据寄存器 (DALI Forward Data Register) 该寄存器的有效长度由 DALI_CR 寄存器中的 ML 决定。 注意： 1、作为主机时，作为前向帧发送数据存放位置； 2、作为从机时，作为前向帧接收数据存放位置。

26.5.2.4 DALI 后向帧数据寄存器 (DALI_BDR)

- **Name**：DALI Backward Data Register
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-8]	Reserved	Reserved	Reserved	Reserved
[7-0]	DFBR	R/W	0x0	后向帧数据寄存器 (DALI Backward Data Register) 注意：后向帧固定为 8 位 1、作为主机时，作为后向帧接收存放位置； 2、作为从机时，作为后向帧发送存放位置。

26.5.2.5 DALI 滤波器控制寄存器 (DALI_FCR)

- **Name**：DALI Filter Control Register
- **Size**：32bits
- **Offset**：0x10
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-17]	Reserved	Reserved	Reserved	Reserved
[16]	FE	R/W	0x0	滤波器使能控制 (DALI Filter Enable) 1：使能滤波器 0：禁止滤波器 注意：如果使用 Filter 功能，请在使能 DALI Enable 前使能 Fileter Enable。
[15-0]	FCNT	R/W	0x0	滤波器计数器 (DALI Filter Counter) DALI 滤波周期 Counter 值配置，以 APB1 时钟周期为单位。

26.5.2.6 DALI 预分频寄存器 (DALI_PSCR)

- **Name**：DALI Prescaler Register
- **Size**：32bits
- **Offset**：0x14
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-13]	Reserved	Reserved	Reserved	Reserved
[12]	FTR	R/W	0x0	DALI 容错范围配置 (DALI Fault Tolerance Range) 0：±%10 1：±%20
[11-0]	PSC	R/W	0x0	DALI 分频值 (DALI Division Value) 波特率 = f_{APB1} / ((PSC + 1) * 16)

26.5.2.7 DALI 定时控制寄存器 (DALI_TCR)

- **Name**：DALI Timing Control Register
- **Size**：32bits
- **Offset**：0x18
- **Default**：0x0

Bit	Field	R/W	Default	Description
[31-12]	Reserved	Reserved	Reserved	Reserved
[22-16]	BDLY	R/W	0x0	后向帧的延时设置 (DALI Backward Frame Delay) 注意：Delay 时间=(BDLY + 7) * T，例如(以 1.2K 波特率为例)： 0：7 * T = 7 * 0.41667 = 2.92ms …… 127：(127 + 7) * T = 134 * 0.41667 = 55.8ms 注意： 1. 针对从机模式下，返回后向帧前的延时 2. 延时配置必须符合 DALI 协议要求
[15-9]	Reserved	Reserved	Reserved	Reserved
[8-0]	FDLY	R/W	0x0	前向帧延时设置 (DALI Forward Frame Delay) 注意：Delay 时间=(FDLY +22) * T，例如(1.2K 波特率为例)： 0：22 * T = 22 * 0.41667 = 9.17ms …… 511：(511 + 22) * T = 533 * 0.41667 = 222.1ms 注意： 1. 针对主机模式下，发送前向帧的延时 2. 延时配置必须符合 DALI 协议要求

27 调试接口

27.1 主要特性

芯片使用 Cortex™-M3 的内核，该内核内含硬件调试模块，支持复杂的调试操作。硬件调试模块允许内核在取指(指令断点)或访问数据(数据断点)时停止，当内核停止时，内核的内部状态和系统的外部状态都是可以查询的。完成查询后，内核和外设可以被复原，程序将继续执行。

芯片支持 SWD 调试接口，当芯片连接到调试器并开始调试时，调试器将使用内核的硬件调试模块进行调试操作。

ARM Cortex™-M3 内核提供集成的片上调试功能。它由以下部分组成：

- SWD-DP：串行调试端口
- ITM：执行跟踪单元
- FPB：闪存指令断点
- DWT：数据触发

有关调试接口的更多信息请参考 ARM 官方手册”Cortex™-M3 Technical Reference Manual - Revision r2p1”，（汉译版本为“Cortex™-M3 技术参考手册”），以及 ARM 开发工具集技术参考手册。

27.2 SWD 调试接口

芯片内核集成了串行调试接口(SWD)，串行调试接口(SWD)为 AHP-AP 模块提供 2 脚接口(时钟＋数据)：

- SWCLK：从主机到目标的时钟
- SWDIO：双向数据接口

对于 SWDIO，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。

芯片的的 2 个普通 I/O 口用作为 SWD 接口的引脚，这些引脚在所有的封装里都存在。

表 27-1 SWD 调试端口引脚

SWD 端口引脚名称	SWD 端口类型	SWD 端口描述	芯片内部状态	引脚分配
SWCLK	输入	时钟	集成下拉	PA14
SWDIO	输入/输出	数据	集成上拉	PA13
SWO	输出		-	

28 设备电子签名

电子签名存储在 FLASH 内部，可以使用 SWD 或 CPU 对其进行读取。它包含出厂前编程的标识数据。

28.1 唯一设备标识（128 位）

唯一设备标标识最适合：

- 用于产品序列号（例如 USB 字符串序列号或其它终端应用程序）
- 用于安全密钥以提高 FLASH 中代码的安全性
- 激活安全自举过程等

128 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。

128 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。

28.2 唯一设备标识寄存器

基址: 0x4002 9080

- **Description**：UID Control register0
- **Size**：32bits
- **Offset**：0x00
- **Default**：0xXXXX XXXX（只读，其中 X 是出厂前编程的）

BIT	Name	Reset	R/W	Description
[31-0]	UID0	0	R/W	设备标识 0

- **Description**：UID Control register1
- **Size**：32bits
- **Offset**：0x04
- **Default**：0xXXXX XXXX（只读，其中 X 是出厂前编程的）

BIT	Name	Reset	R/W	Description
[31-0]	UID1	0	R/W	设备标识 1

- **Description**：UID Control register2
- **Size**：32bits
- **Offset**：0x08
- **Default**：0xXXXX XXXX（只读，其中 X 是出厂前编程的）

BIT	Name	Reset	R/W	Description
[31-0]	UID2	0	R/W	设备标识 2

- **Description**：UID Control register3
- **Size**：32bits
- **Offset**：0x0C
- **Default**：0xXXXX XXXX（只读，其中 X 是出厂前编程的）

BIT	Name	Reset	R/W	Description
[31-0]	UID3	0	R/W	设备标识 3

29 电气特性

29.1 参数条件

除非特别说明，所有电压都是以 VSS 作为参考。

29.2 最大值和最小值

除非特别说明，最大值和最小值是在最差环境温度、供电电压以及频率下的测试结果。

29.3 绝对最大额定值

超过下表所示绝对最大额定值的电压值、电流或温度可能导致芯片永久性的损坏。

29.3.1 电压特性

符号	说明	Min	Max	单位
V_{CC} -V_{SS}	片外供电电压，包括 VCC、AVCC。	-0.3	4.0	V
V_{CC} - AV_{CC}	允许的 VCC 和 AVCC 电压差值	-0.4	0.4	V
VIN	GPIO 的输入电压	VSS-0.3	4.0	V
	晶振接口的输入电压	VSS-0.3	1.8	V
ΔV_{CCx}	不同 VCC 引脚的电压差值	-	50	mV
ΔV_{SSx}	不同 VSS 引脚的电压差值(包括 AVSS)	-	50	mV
ESD(HBM)	静电放电人体模型电压	TBD	-	V

29.3.2 电流特性

符号	描述	最大值	单位
ΣI_{VCC}	所有 VCC 引脚输入的总电流	200	mA
ΣI_{AVCC}	所有 AVCC 引脚输入的总电流	100	mA
I_{VCC}	单个 VCC 引脚输入的总电流	100	mA
I_{AVCC}	单个 AVCC 引脚输入的总电流	100	mA
I_{IO}	单个 GPIO 输入/输出的电流	24	mA
ΣI_{IO}	所有 GPIO 输入/输出的总电流	80	

29.3.3 温度特性

符号	描述	最大值	单位
T_{STRG}	芯片储存温度范围	TBD	℃
T_j	芯片结温	125	℃

29.4 典型工作条件

General Operating Conditions						
Symbol	**Parameters**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
F_{fclk}	芯片内部 AHB0 时钟频率		0		90	MHz
F_{pclk1}	芯片内部APB0时钟频率		0			MHz
F_{pclk2}	芯片内部APB1时钟频率		0			MHz
VCC	数字 IO 工作电压		2.97	3.3	3.63	V
AVCC	模拟供电电压	与 VCC 电压相同	2.97	3.3	3.63	V
VDD	数字 Core 供电电压，片内 LDO		1.35	1.5	1.65	V
V_{REF}	参考电压	校正后		1.5		
V_{IN}	IO 输入电压	GPIO	-0.3		VCC+0.3	V
		HXI、HXO	-0.3		VDD+0.3	V
PD	整体功耗	温度 TA=85℃，工作频率 90MHz，无 IO 负载			400	mW
TJ	芯片结温		-40		125	℃

29.5 芯片上电/掉电特性

（VCC=AVCC=3.3V，VDD=1.5V，TA=25℃，除非另有说明）

Operating conditions at power up / power down						
Symbol	**Parameters**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
VCC_{POR}	AVCC/VCC 上电、掉电复位阈值电压	下降沿	1.54	1.75	1.91	V
		上升沿	1.97	2.12	2.3	V
$VCC_{PORhyst}$	AVCC/VCC POR 迟滞			400		mV
VCC_{OK}	AVCC/VCC 上电 OK 阈值	下降沿		2.25		V
		上升沿		2.7		V
VDD_{POR}	VDD 上电、掉电复位阈值电压	下降沿	0.54	0.66	0.7	V
		上升沿	0.82	0.87	0.92	V
$VDD_{PORhyst}$	VDD POR 迟滞			100		mV
VCC_{LVD}	VCC、AVCC 低压检测阈值	2bit 可配置为 2.4V~3V，偏差与 BGR 精度一致	2.4		3	V
T_{rst}	上电复位时间		1.3	2	2.9	ms

29.6 片内参考电压

（AVCC=3.3V，TA=25℃）

Internal Reference Voltage						
Symbol	**Parameters**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
V_{REF}	内部参考电压	-40℃ < TA < 105℃		1.5		V
T_{coeff}*	温度系数			50	100	ppm/℃
VRP	内部正参考电压	-40℃ < TA < 105℃，TRIM 后		3		V
VRP_{no}*	内部正参考电压输出噪声	片外滤波电容 2.2uF，积分带宽 1Hz~1GHz		110		uVrms

29.7 时钟特性

29.7.1 片外晶振特性

（VCC=AVCC=3.3V，AVDD=1.5V，TA=25℃）

External Crystal Oscillator Characteristics						
Symbol	**Parameter**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
F_{xosc}	外部输入时钟频率或晶振频率		4	8	26	MHz
VXI_H	XI 输入高电平	VDD=1.5V，如果输入外部时钟，XO 浮空，从 XI 输入，不同于 3.3V IO，该 IO 电平为 0~1.5V。	1.05	-	1.65	V
VXI_L	XI 输入低电平		0	-	0.45	V
Duty	外部输入时钟占空比		45	50	55	%
RFB	反馈电阻		40	50	60	kΩ
IVCC	起振电路功耗	起振期间			2	mA
		Rs=30，CL=10pF @ 8MHz		0.4		
		Rs=45，CL=10pF @ 8MHz		0.5		
		Rs=30，CL=5pF @ 32MHz		0.8		
		Rs=30，CL=10pF @ 32MHz		1		
		Rs=30，CL=20pF @ 32MHz		1.5		
gm*	振荡器跨导	起振期间，DR 可调	2		20	mA/V
T_{su}*	启动时间	从软件使能到输出稳定的时钟		0.5	2	ms

注：*-由设计保证，实际芯片不测试。

29.7.2 片内高速 RC 振荡器特性

（VCC=AVCC=3.3V，VDDRC=1.5V，TA=25℃）

High-Speed Internal RC oscillator Characteristics						
symbol	**Parameter**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
F_{HSI}**	Frequency		5	8	13	MHz
DuCy	Duty Cycle		45	50	55	%
ACC*	振荡器的频率精度	TA=-40 ~ 105℃		±1		%
		TA=-10 ~ 85℃		±0.6		
		TA=0 ~ 85℃				
T_{su}	启动时间	从软件使能到震荡到目标频率			1	us
IVCC	振荡器功耗	TA=25℃ @ 8MHz		1.2		mA

注：*-振荡器频率在 TA=25℃测量并存储到 eFLASH 中，振荡器的频率稳定性由设计保证，量产芯片中并未全部测量。

注：**-振荡器频率不校正，通过调节 PLL 分频系数进行 PLL 输出频率校正，绝对精度<±1%。

29.7.3 片内低速 RC 振荡器特性

（VCC=AVCC=3.3V，TA=25℃）

Low-Speed Internal RC oscillator Characteristics						
symbol	**Parameter**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
F_{LSI}**	Frequency		24	32	40	kHz
DuCy	Duty Cycle		45	50	55	%
ACC*	振荡器的频率精度	TA=-40 ~ 105℃		±0.35		%

注：*-由设计保证，实际芯片不测试。

注：**-振荡器频率无校正，FLASH 存储其 TA=25℃的频率值，可通过查询 FLASH NVR 区域获得实际振荡频率。

29.7.4 片内锁相环特性

（VCC=AVCC=3.3V，TA=25℃）

PLL Characteristics					
Symbol	**Parameter**	**Min.**	**Typ.**	**Max.**	**Unit**
F_{PLL_IN}	PLL 输入时钟	4		26	MHz
F_{PLL_OUT}	PLL 输出时钟	60	160	220	MHz
T_{LOCK}*	PLL 锁定时间		0.5	2	ms
T_{Jitter}*	Cycle to Cycle Jitter			300	ps

注：*-由设计保证，实际芯片不测试。

29.7.5 高精度 PWM 特性

（VCC=AVCC=3.3V，TA=25℃）

HRPWM Characteristics						
Symbol	**Parameter**	**Condition**	**Min.**	**Typ.**	**Max.**	**Unit**
F_{HRPWM}	高精度PWM时钟频率		120		160	MHz
T_{RES}*	时钟分辨率	F_{HRPWM}=160MHz, TA=-40~105℃		195	400	ps
ResTIMER	定时器分辨率				16	bit
$F_{CHOPFREQ}$	斩波时钟频率		1/256		1/16	F_{HRPWM}
T_{1STPW}	斩波第一个脉冲宽度		16		256	T_{HRPWM}

注：*-由设计保证，实际芯片不测试。

29.8 存储器特性

FLASH Memory Characteristics						
Symbol	**Parameter**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
T_{PROG}	Byte Program time			6		us
T_{SECTOR_ERAZE}	Sector Erase time			4		ms
T_{CHIP_ERAZE}	Chip Erase time			20		ms
T_{ACCESS}	Fast read access time			30		ns
NEND	EFLASH Endurance		20			kCycles
	DFLASH Endurance		100			
T_{RET}	Data Retention	TA=25℃	100			Years
		TA=105℃	20			
		TA=125℃	10			

29.9 通用输入/输出端口特性

IO static Characteristics						
Symbol	**Parameter**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
VIL	Low level input voltage	GPIO Hysteresis On	0.43*VCC-0.15	0.45*VCC-0.12	0.48*VCC-0.12	V
		HXI Hysteresis On	0.665	0.714	0.771	
		GPIO Hysteresis Off	0.455*VCC	0.5*VCC	0.54*VCC	
VIH	High level input voltage	GPIO Hysteresis On	0.54*VCC+0.08	0.545*VCC+0.165	0.56*VCC+0.2	
		HXI Hysteresis On	0.732	0.784	0.846	
		GPIO Hysteresis Off	0.455*VCC	0.5*VCC	0.54*VCC	
CIO	IO pin capacitance	Total Capacitance should add Package Cap:1.5pF	1.39	1.65	1.75	pF
RPU	weak pull-up equivalent resistor		8.3	10	12	kΩ
RPD	weak pull-down equivalent resistor		8.3	10	12	kΩ
VOL	Low level output voltage	IIO=+6mA, DR=0	-	-	0.57	V
		IIO=+20mA, DR=1	-	-	0.66	

VOH	High level output voltage	IIO=+6mA, DR=0	VCC-0.59	-	-	V
		IIO=+20mA, DR=1	VCC-0.68	-	-	
RPU_MCLR	MCLR PIN Pull-up resistor		75	100	125	kΩ
RPU_BOOT	BOOT PIN Pull-up resistor		75	100	125	kΩ

29.10 模拟外设特性

29.10.1 模数转换器特性

（测试条件为：AVCC=3.3V，TA=25℃，F_ADC=160MHz，除非另有说明）

ADC Characteristics						
Symbol	**Parameter**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
AVCC	模拟电源电压		3.1	3.3	3.63	V
Resolution	ADC 分辨率			13		bit
V_{ref+}	ADC 正参考电压			3		V
V_{ref-}	ADC 负参考电压			0		V
V_{AIN}	ADC 输入电压范围		GND		3	V
C_{IN}	ADC 输入电容			1.6		pF
F_{ADC}	ADC 工作时钟			160		MHz
F_s	ADC 采样率				2.2	MSps
T_{sample}	ADC 采样时间，ADC 时钟周期数		6		1530	Tadc
T_{con}	总转换时间	Tcon=Tsample(6~1530Tadc)+Tconv(30Tadc)	36		1560	Tadc
DNL*	微分非线性		-1			LSB
INL*	积分非线性		-5		16	LSB
I_AVCC	ADC 功耗	单端/差分，2.2MSPS		2		mA
DR*	ADC 动态范围	差分输入，-40dBFS 输入测量，10kHz 正弦信号		70		dB
		单端输入，-40dBFS 输入测量，10kHz 正弦信号		65		dB
SNDR*	ADC 信号与噪声失真比	差分输入，-6dBFS 输入测量，10kHz 正弦信号		68		dB
		单端输入，-6dBFS 输入测量，10kHz 正弦信号		65		dB

注：*-由设计保证，实际芯片不测试。

29.10.2 数模转换器特性

（测试条件为：AVCC=3.3V，TA=25℃，除非另有说明）

DAC Characteristics						
Symbol	**Parameter**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
AVCC	模拟电源电压		3.1	3.3	3.6	V
Resolution	DAC 分辨率			12		bit
V_{ref+}	DAC 正参考电压			AVCC		V
V_{ref-}	DAC 负参考电压			GND		V
V_{DACOUT}	DAC 输出电压范围		GND		AVCC-1LSB	V
DNL*	微分非线性				±1	LSB
INL*	积分非线性				±2	LSB
I_AVCC	DAC 功耗			2		mA
Update Rate	DAC 数据更新速率				2	MS/s

注：*-由设计保证，实际芯片不测试。

29.10.3 比较器特性

（AVCC=3.3V，TA=25℃）

Comparator Characteristics						
Symbol	**Parameter**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
AVCC	模拟电源电压		3.1	3.3	3.6	V
V_{in}	比较器输入电压范围		GND		AVCC	V
V_{offset}*	比较器失调电压			±5		mV
V_{hyst}*	比较器迟滞电压	0~30mV 可调	0		30	mV
T_{DLY}*	比较器延时	从比较器输入引脚到输出引脚，CL=30pF		15	20	ns

注：*-由设计保证，实际芯片不测试。

29.10.4 温度传感器特性

（AVCC=3.3V，TA=25℃）

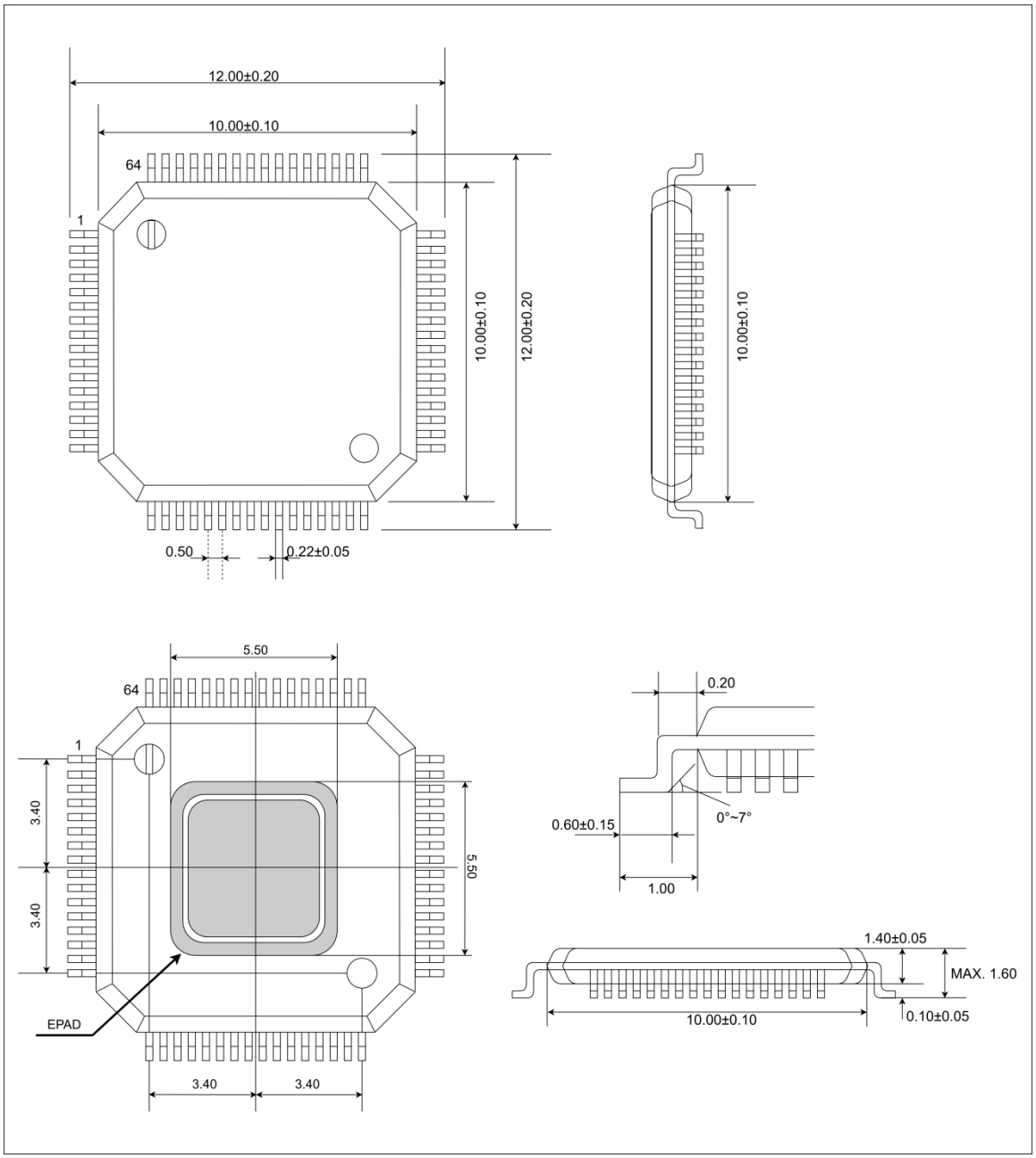
Temperature Sensor Characteristics						
Symbol	**Parameter**	**Conditions**	**Min.**	**Typ.**	**Max.**	**Unit**
Slope*	温度曲线斜率		4.6	4.81	5	mV/℃
V_0	0 摄氏度电压		1.26	1.34	1.42	V
T_{START}	启动时间		4		10	us

注：为了保证温度传感器测量精度，可开启 ADC 过采样。

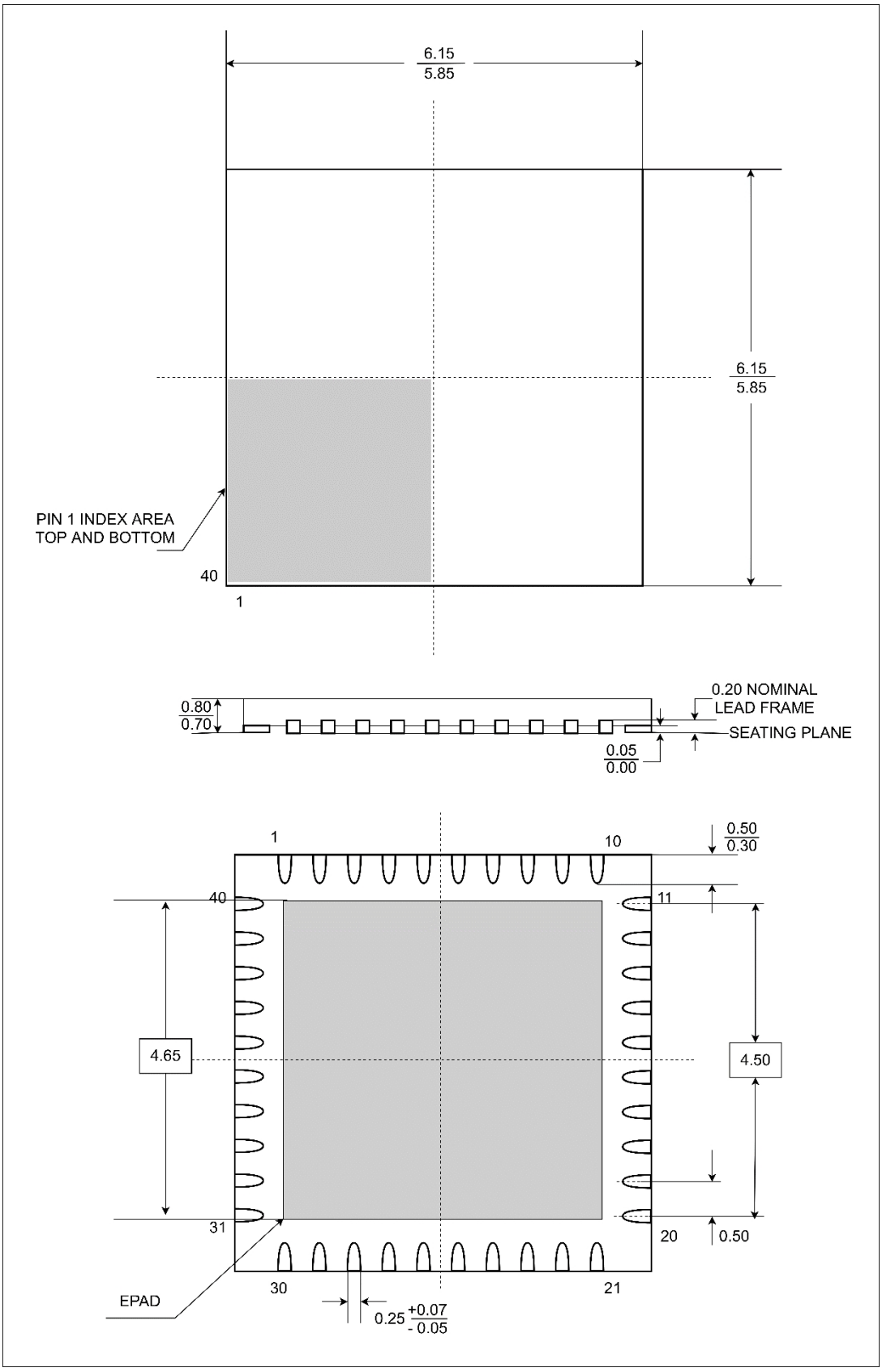
注：*-由设计保证，实际芯片不测试。

30 封装信息

LQFP 64L：



WQFN 40L：



31 订购信息

Example: TAE32 F 5300 A L F 80

产品序列

TAE30 = ARM Cortex-M0 32bit 系列
TAE32 = ARM Cortex-M3 32bit 系列
TAE33 = ARM Cortex-M4 32bit 系列
TAE34 = ARM Cortex-M7 32bit 系列

产品类型

F = Flash类型，无特殊功耗需求
L = 低功耗类型

产品型号

5300 = 产品型号

温度范围

G = 工作温度范围为-40℃至+105℃
A = 工作温度范围为-40℃至+125℃

封装类型

L = 封装类型： LQFP
Q = 封装类型：WQFN

引脚数

C = 引脚数36引脚
D = 引脚数40引脚
F = 引脚数64引脚

Flash 大小

80= 80KB

版本历史

日期	版本	版本记录
2021/03/20	v1.0	初始版本