



HC32L07x Series

32-bit ARM® Cortex®-M0+ microcontrollers

Reference Manual

Rev1.72 March 2025

Statement

- ★ Xiaohua Semiconductor Co., Ltd. (hereinafter referred to as "XHSC") reserves the right to change, correct, enhance, and modify Xiaohua Semiconductor products and/or this document at any time without prior notice. Users can obtain the latest relevant information before placing an order. XHSC products are sold in accordance with the sales terms and conditions stated in the basic purchase and sales contract.

- ★ Customers are solely responsible for selecting appropriate XHSC products for their applications, and for designing, validating, and testing their applications to ensure that they meet the applicable standards and any safety, security, or other requirements.

- ★ XHSC hereby acknowledges that no license to any intellectual property rights is granted, either expressly or impliedly.

- ★ Resale of XHSC products on terms different from those set forth herein will void any warranty provided by XHSC with respect to such products.

- ★ Any graphics or words with "®" or "™" logo are trademarks of XHSC. All other product or service names displayed on XHSC products are the property of their respective owners.

- ★ The information in this notice supersedes and replaces that in previous versions.

©2025 Xiaohua Semiconductor Co., Ltd. All rights reserved

Table of contents

声 明.....	2
目 录.....	3
表索引.....	29
图索引.....	32
简介 (Overview)	39
1 系统结构	40
1.1 概述	40
1.2 系统地址划分	41
1.3 存储器和模块地址分配	42
2 工作模式	43
2.1 运行模式	45
2.2 休眠模式	46
2.3 深度休眠模式	47
3 系统控制器 (SYSCTRL)	50
3.1 系统时钟介绍	50
3.1.1 内部高速 RC 时钟 RCH	51
3.1.2 内部低速 RC 时钟 RCL	51
3.1.3 外部低速晶振时钟 XTL	52
3.1.4 外部高速晶振时钟 XTH	52
3.1.5 锁相环时钟 PLL	52
3.1.6 内部高速时钟 RCH48M	52
3.1.7 时钟启动过程	53
3.2 系统时钟切换	53
3.2.1 标准的时钟切换流程	53
3.2.2 RCH 不同振荡频率间切换流程	54
3.2.3 从其它时钟切换到 XTL 示例	55
3.2.4 从其它时钟切换到 XTH 示例	55
3.2.5 从其它时钟切换到 RCL 示例	56
3.2.6 从其它时钟切换到 RCH 示例	56
3.2.7 PLL 与 RCH 相互切换示例，参考时钟为 RCH	56
3.2.8 PLL 与 XTH 相互切换示例，参考时钟为 XTH	58
3.3 时钟校准模块	60
3.4 中断唤醒控制	60

3.4.1	从深度休眠模式唤醒后执行中断服务程序的方法.....	61
3.4.2	从深度休眠模式唤醒后不执行中断服务程序的方法.....	61
3.4.3	使用退出休眠特性	62
3.5	寄存器	63
3.5.1	系统控制寄存器 0 (SYSCTRL0)	64
3.5.2	系统控制寄存器 1 (SYSCTRL1)	66
3.5.3	系统控制寄存器 2 (SYSCTRL2)	68
3.5.4	RCH 控制寄存器 (RCH_CR)	69
3.5.5	XTH 控制寄存器 (XTH_CR)	70
3.5.6	RCL 控制寄存器 (RCL_CR)	71
3.5.7	XTL 控制寄存器 (XTL_CR)	72
3.5.8	PLL 控制寄存器 (PLL_CR)	73
3.5.9	外围模块时钟控制寄存器 (PERI_CLKEN0)	75
3.5.10	外围模块时钟控制寄存器 (PERI_CLKEN1)	77
3.5.11	定时器时钟倍频控制 (OVCK_CR)	78
3.5.12	RC48M 控制寄存器 (RC48M_CR)	78
4	复位控制器 (RESET)	79
4.1	复位控制器介绍.....	79
4.1.1	上电下电复位 POR	80
4.1.2	外部复位管脚复位	80
4.1.3	WDT 复位	80
4.1.4	PCA 复位	80
4.1.5	LVD 低电压复位	80
4.1.6	Cortex-M0+ SYSRESETREQ 复位	80
4.1.7	Cortex-M0+ LOCKUP 复位	80
4.2	寄存器.....	81
4.2.1	复位标识寄存器 (RESET_FLAG)	81
4.2.2	外围模块复位控制寄存器 (PERI_RESET0)	83
4.2.3	外围模块复位控制寄存器 (PERI_RESET1)	85
5	中断控制器 (NVIC)	86
5.1	概述	86
5.2	中断优先级.....	86
5.3	中断向量表.....	87
5.4	中断输入和挂起行为.....	88
5.5	中断等待	90

5.6 中断源.....	91
5.7 中断结构图.....	92
5.8 寄存器.....	94
5.8.1 中断使能设置寄存器 (NVIC_ISER)	95
5.8.2 中断使能清除寄存器 (NVIC_ICER)	95
5.8.3 中断挂起状态设置寄存器 (NVIC_ISPR)	96
5.8.4 中断挂起状态清除寄存器 (NVIC_ICPR)	96
5.8.5 中断优先级寄存器 (NVIC_IPR0)	97
5.8.6 中断优先级寄存器 (NVIC_IPR1)	97
5.8.7 中断优先级寄存器 (NVIC_IPR2)	98
5.8.8 中断优先级寄存器 (NVIC_IPR3)	98
5.8.9 中断优先级寄存器 (NVIC_IPR4)	99
5.8.10 中断优先级寄存器 (NVIC_IPR5)	99
5.8.11 中断优先级寄存器 (NVIC_IPR6)	100
5.8.12 中断优先级寄存器 (NVIC_IPR7)	100
5.8.13 中断屏蔽特殊寄存器 (PRIMASK)	101
5.9 软件基本操作	102
5.9.1 外部中断使能	102
5.9.2 NVIC 中断使能和清除使能.....	102
5.9.3 NVIC 中断挂起和清除挂起.....	102
5.9.4 NVIC 中断优先级.....	102
5.9.5 NVIC 中断屏蔽	102
6 端口控制器 (GPIO)	104
6.1 端口控制器简介.....	104
6.2 端口控制器主要特性.....	104
6.3 端口控制器功能描述.....	105
6.3.1 端口配置功能	105
6.3.2 端口的写入.....	108
6.3.3 端口的读取.....	108
6.3.4 端口数字复用功能	110
6.3.5 端口中断功能	113
6.4 端口配置操作流程	113
6.4.1 端口复用配置为模拟端口操作流程	113
6.4.2 端口复用配置为数字通用端口操作流程	113
6.4.3 端口复用配置为数字功能端口操作流程	113

6.4.4	端口复用配置为调试测试端口操作流程.....	113
6.4.5	端口复用配置为红外输出信号操作流程.....	113
6.4.6	端口高电平中断操作流程.....	114
6.4.7	端口低电平中断操作流程.....	114
6.4.8	端口上升沿中断操作流程.....	114
6.4.9	端口下降沿中断操作流程.....	114
6.4.10	端口上拉使能配置操作流程	115
6.4.11	端口下拉使能配置操作流程	115
6.4.12	端口增强驱动配置操作流程	115
6.4.13	端口开漏输出配置操作流程	115
6.4.14	端口位置位操作流程	115
6.4.15	端口位清零操作流程	115
6.4.16	端口位置位清零操作流程.....	115
6.5	端口控制器寄存器描述	116
6.5.1	端口通用寄存器.....	122
6.5.2	端口辅助寄存器.....	131
6.5.3	端口 PA 功能选择寄存器.....	139
6.5.4	端口 PB 功能选择寄存器.....	148
6.5.5	端口 PC 功能选择寄存器	157
6.5.6	端口 PD 功能选择寄存器	166
6.5.7	端口 PE 功能选择寄存器	175
6.5.8	端口 PF 功能选择寄存器.....	184
7	I2C 总线 (I2C)	190
7.1	简介	190
7.2	主要特性	190
7.3	协议描述	191
7.3.1	I2C 总线上数据传输	191
7.3.2	I2C 总线上的应答.....	192
7.3.3	I2C 总线上的仲裁.....	193
7.4	功能描述	194
7.4.1	串行时钟发生器.....	195
7.4.2	输入滤波器	195
7.4.3	地址比较器	195
7.4.4	应答标志位	195
7.4.5	中断产生器	196

7.4.6	工作模式	196
7.4.7	状态码表述	201
7.5	编程示例	204
7.5.1	主机发送示例	204
7.5.2	主机接收示例	205
7.5.3	从机接收示例	206
7.5.4	从机发送示例	207
7.6	寄存器描述	208
7.6.1	I2C 波特率计数器使能寄存器(I2Cx_TMRUN)	209
7.6.2	I2C 波特率计数器配置寄存器(I2Cx_TM)	209
7.6.3	I2C 配置寄存器(I2Cx_CR)	210
7.6.4	I2C 数据寄存器(I2Cx_DATA)	211
7.6.5	I2C 地址寄存器(I2Cx_ADDR)	211
7.6.6	I2C 状态寄存器(I2Cx_STAT)	212
8	串行外设接口 (SPI)	213
8.1	SPI 简介	213
8.2	SPI 主要特性	213
8.3	SPI 功能描述	214
8.3.1	SPI 主机模式	214
8.3.2	SPI 从机模式	215
8.3.3	SPI 数据帧格式	216
8.3.4	SPI 状态标志及中断	218
8.3.5	SPI 多机系统配置说明	218
8.3.6	SPI 管脚配置说明	220
8.4	SPI 编程示例	221
8.4.1	SPI 主机发送示例	221
8.4.2	SPI 主机接收示例	221
8.4.3	SPI 从机发送示例	222
8.4.4	SPI 从机接收示例	222
8.5	SPI 寄存器描述	224
8.5.1	SPI 配置寄存器(SPIx_CR)	225
8.5.2	SPI 片选配置寄存器(SPIx_SSN)	226
8.5.3	SPI 状态寄存器(SPIx_STAT)	227
8.5.4	SPI 数据寄存器(SPIx_DATA)	228
8.5.5	SPI 配置寄存器 2(SPIx_CR2)	229

8.5.6	SPI 中断清除寄存器 2(SPIx_ICLR).....	230
9	时钟校准模块 (CLKTRIM)	231
9.1	CLKTRIM 简介.....	231
9.2	CLKTRIM 主要特性.....	231
9.3	CLKTRIM 功能描述.....	231
9.3.1	CLKTRIM 校准模式	231
9.3.2	CLKTRIM 监测模式	234
9.4	CLKTRIM 寄存器描述	236
9.4.1	配置寄存器(CLKTRIM_CR).....	237
9.4.2	参考计数器初值配置寄存器(CLKTRIM_REFCON).....	238
9.4.3	参考计数器值寄存器(CLKTRIM_REFCNT).....	238
9.4.4	校准计数器值寄存器(CLKTRIM_CALCNT).....	239
9.4.5	中断标志位寄存器(CLKTRIM_IFR)	239
9.4.6	中断标志位清除寄存器(CLKTRIM_ICLR).....	240
9.4.7	校准计数器溢出值配置寄存器(CLKTRIM_CALCON)	240
10	硬件除法器模块 (HDIV)	241
10.1	HDIV 简介	241
10.2	HDIV 主要特性	241
10.3	HDIV 功能描述	242
10.3.1	HDIV 操作流程.....	242
10.4	HDIV 寄存器描述.....	243
10.4.1	被除数寄存器(HDIV_DIVIDEND).....	244
10.4.2	除数寄存器(HDIV_DIVISOR)	244
10.4.3	商寄存器(HDIV_QUOTIENT)	245
10.4.4	余数寄存器(HDIV_REMAINDER)	245
10.4.5	符号寄存器(HDIV_SIGN)	246
10.4.6	状态寄存器(HDIV_STAT).....	246
11	FLASH 控制器 (FLASH)	247
11.1	概述	247
11.2	FLASH 容量划分	247
11.3	功能描述	247
11.3.1	页擦除 (Sector Erase)	248
11.3.2	全片擦除 (Chip Erase)	248
11.3.3	写操作 (Program)	249
11.3.4	读操作 (Read)	251

11.4 擦写时间	252
11.5 读等待周期	253
11.6 擦写保护	253
11.6.1 擦写保护位	253
11.6.2 PC 地址擦写保护	253
11.7 寄存器写保护	254
11.8 寄存器	255
11.8.1 TNVS 参数寄存器 (FLASH_TNVS)	256
11.8.2 TPGS 参数寄存器 (FLASH_TPGS)	256
11.8.3 TPROG 参数寄存器 (FLASH_TPROG)	257
11.8.4 TSERASE 寄存器 (FLASH_TSERASE)	257
11.8.5 TMERASE 参数寄存器 (FLASH_TMERASE)	258
11.8.6 TPRCV 参数寄存器 (FLASH_TPRCV)	258
11.8.7 TSRCV 参数寄存器 (FLASH_TSRCV)	259
11.8.8 TMRCV 参数寄存器 (FLASH_TMRCV)	259
11.8.9 CR 寄存器 (FLASH_CR)	260
11.8.10 IFR 寄存器 (FLASH_IFR)	260
11.8.11 ICLR 寄存器 (FLASH_ICLR)	261
11.8.12 BYPASS 寄存器 (FLASH_BYPASS)	261
11.8.13 SLOCK0 寄存器 (FLASH_SLOCK0)	262
11.8.14 SLOCK1 寄存器 (FLASH_SLOCK1)	262
12 RAM 控制器 (RAM)	263
12.1 概述	263
12.2 功能描述	263
12.2.1 RAM 地址范围	263
12.2.2 读写位宽	263
12.2.3 奇偶校验	263
12.3 寄存器	264
12.3.1 控制寄存器 (RAM_CR)	265
12.3.2 奇偶校验出错地址寄存器 (RAM_ERRADDR)	265
12.3.3 出错中断标志寄存器 (RAM_IFR)	266
12.3.4 出错中断标志清除寄存器 (RAM_ICLR)	266
13 DMA 控制器 (DMAC)	267
13.1 DMAC 简介	267
13.2 DMAC 主要特性	267

13.3 功能框图	268
13.4 功能描述	268
13.4.1 DMAC 传输模式概述	268
13.4.2 DMA 软件 Block 传输模式	269
13.4.3 DMA 软件 Burst 传输模式	269
13.4.4 DMA 硬件 Block 传输模式	271
13.4.5 DMA 硬件 Burst 传输模式	272
13.4.6 DMA 传输暂停	273
13.4.7 DMA 其它配置	273
13.4.8 DMA 中断	274
13.5 寄存器	275
13.5.1 DMAC_CONF	276
13.5.2 DMAC_CONFA0、DMAC_CONFA1	277
13.5.3 DMAC_CONFB0、DMAC_CONFB1	279
13.5.4 DMAC_SRCADR0、DMAC_SRCADR1	281
13.5.5 DMAC_DSTADR0、DMAC_DSTADR1	281
14 通用定时器 (TIM0/1/2/3)	282
14.1 通用定时器简介	282
14.1.1 基本特性(TIM0/1/2)	282
14.1.2 基本特性(TIM3)	283
14.2 Timer 功能描述	284
14.2.1 定时器时钟	284
14.2.2 定时计数器	284
14.2.3 定时器预除频	284
14.2.4 模式 0 计数定时器功能	285
14.2.5 模式 1 脉宽测量 PWC	288
14.2.6 模式 2/3 比较捕获模式	292
14.2.7 模式 2/3 从模式	314
14.2.8 正交编码计数功能	315
14.2.9 Timer 触发 ADC	318
14.2.10 刹车控制	319
14.2.11 Timer 互联	319
14.2.12 GATE 输入互联	320
14.2.13 ETR 输入互联	320
14.2.14 CHx 捕获输入互联	320

14.2.15 DMA	321
14.3 Timer 寄存器描述	323
14.4 模式 0 定时器寄存器描述	324
14.4.1 16 位模式重载寄存器 (TIMx_ARR)	324
14.4.2 16 位模式计数寄存器 (TIMx_CNT)	324
14.4.3 32 位模式计数寄存器 (TIMx_CNT32)	325
14.4.4 控制寄存器 (TIMx_M0CR)	326
14.4.5 中断标志寄存器 (TIMx_IFR)	327
14.4.6 中断标志清除寄存器 (TIMx_ICLR)	327
14.4.7 死区时间寄存器 (TIMx_DTR)	328
14.5 脉冲宽度测量 PWC 寄存器描述	329
14.5.1 16 位模式计数寄存器 (TIMx_CNT)	329
14.5.2 控制寄存器 (TIMx_M1CR)	330
14.5.3 中断标志寄存器 (TIMx_IFR)	331
14.5.4 中断标志清除寄存器 (TIMx_ICLR)	331
14.5.5 主从模式控制寄存器 (TIMx_MSCR)	332
14.5.6 输出控制滤波 (TIMx_FLTR)	333
14.5.7 控制寄存器 (TIMx_CR0)	334
14.5.8 比较捕获寄存 (TIMx_CCR0A)	334
14.6 模式 2,3 寄存器描述	335
14.6.1 16 位模式重载寄存器 (TIMx_ARR)	335
14.6.2 16 位模式计数寄存器 (TIMx_CNT)	335
14.6.3 控制寄存器 (TIMx_M23CR)	336
14.6.4 中断标志寄存器 (TIMx_IFR)	339
14.6.5 中断标志清除寄存器 (TIMx_ICLR)	341
14.6.6 主从模式控制寄存器 (TIMx_MSCR)	342
14.6.7 输出控制/输入滤波 (TIMx_FLTR)	344
14.6.8 ADC 触发控制寄存器 (TIMx_ADTR)	347
14.6.9 通道 0 控制寄存器 (TIMx_CRCH0)	348
14.6.10 通道 1/2 控制寄存器 (TIM3_CRCH1/2) (仅 TIM3 存在)	350
14.6.11 死区时间寄存器 (TIMx_DTR)	352
14.6.12 重复周期设置值 (TIMx_RCR)	353
14.6.13 通道 0 比较捕获寄存器 (TIMx_CCR0A/B)	353
14.6.14 通道 1/2 比较捕获寄存器 (TIM3_CCR1/2 A/B) (仅 TIM3 存在)	354
15 低功耗定时器 (LPTIM)	355

15.1 LPTimer 简介.....	355
15.2 LPTimer 功能描述.....	355
15.2.1 计数功能	356
15.2.2 定时功能	356
15.3 LPTimer 互连.....	357
15.3.1 GATE 互联	357
15.3.2 EXT 互联.....	357
15.3.3 Toggle 输出互联	357
15.4 LPTimer 寄存器描述.....	357
15.4.1 计数器计数值寄存器 (LPTIMx_CNT)	359
15.4.2 重载寄存器 (LPTIMx_ARR)	359
15.4.3 控制寄存器 (LPTIMx_CR)	360
15.4.4 中断标志寄存器 (LPTIMx_IFR)	361
15.4.5 中断标志清除寄存器 (LPTIMx_ICLR)	361
16 可编程计数阵列 (PCA)	362
16.1 PCA 简介	362
16.2 PCA 功能描述.....	362
16.2.1 PCA 定时/计数器.....	363
16.2.2 PCA 捕获功能	365
16.2.3 PCA 比较功能	366
16.3 PCA 模块与其他模块互连及控制.....	371
16.4 PCA 寄存器描述	372
16.4.1 控制寄存器 (PCA_CCON)	373
16.4.2 模式寄存器 (PCA_CMOD)	374
16.4.3 计数寄存器 (PCA_CNT)	375
16.4.4 中断清除寄存器 (PCA_ICLR)	375
16.4.5 比较捕获模式寄存器 (PCA_CCAPM0~4)	376
16.4.6 比较捕获数据寄存器高 8 位 (PCA_CCAP0~4H)	377
16.4.7 比较捕获数据寄存器低 8 位 (PCA_CCAP0~4L)	377
16.4.8 比较捕获 16 位寄存器 (PCA_CCAP0~4)	378
16.4.9 比较高速输出标志寄存器 (PCA_CCAPO)	378
16.4.10 周期寄存器 (PCA_CARR)	379
16.4.11 增强 PWM 控制 (PCA_EPWM)	379
17 高级定时器 (TIM4/5/6)	380
17.1 Advanced Timer 简介.....	380

17.2 Advanced Timer 功能描述.....	382
17.2.1 基本动作	382
17.2.2 时钟源选择.....	384
17.2.3 计数方向	385
17.2.4 数字滤波	385
17.2.5 软件同步	386
17.2.6 硬件同步	386
17.2.7 缓存功能	389
17.2.8 通用 PWM 输出.....	390
17.2.9 正交编码计数	395
17.2.10 周期间隔响应	399
17.2.11 保护机制	400
17.2.12 中断说明	400
17.2.13 DMA.....	401
17.2.14 刹车保护	401
17.2.15 内部互连	403
17.3 寄存器描述.....	406
17.3.1 通用计数基准值寄存器 (TIMx_CNTER).....	408
17.3.2 通用周期基准值寄存器 (TIMx_PERAR)	408
17.3.3 通用周期缓存寄存器 (TIMx_PERBR)	409
17.3.4 通用比较基准值寄存器 (TIMx_GCMAR-GCMDR)	409
17.3.5 专用比较基准值寄存器 (TIMx_SCMAR-SCMBR)	410
17.3.6 死区时间基准值寄存器 (TIMx_DTUAR- DTDAR)	410
17.3.7 通用控制寄存器 (TIMx_GCONR)	411
17.3.8 中断控制寄存器 (TIMx_ICONR)	412
17.3.9 端口控制寄存器 (TIMx_PCONR)	413
17.3.10 缓存控制寄存器 (TIMx_BCONR)	416
17.3.11 死区控制寄存器 (TIMx_DCONR)	417
17.3.12 滤波控制寄存器 (TIMx_FCONR)	418
17.3.13 有效周期寄存器 (TIMx_VPERR)	419
17.3.14 状态标志寄存器 (TIMx_STFLR)	420
17.3.15 硬件启动事件选择寄存器 (TIMx_HSTAR)	422
17.3.16 硬件停止事件选择寄存器 (TIMx_HSTPR)	424
17.3.17 硬件清零事件选择寄存器 (TIMx_HCELR)	426
17.3.18 硬件捕获 A 事件选择寄存器 (TIMx_HCPAR)	428

17.3.19 硬件捕获 B 事件选择寄存器 (TIMx_HCPBR)	430
17.3.20 硬件递加事件选择寄存器 (TIMx_HCUPR)	432
17.3.21 硬件递减事件选择寄存器 (TIMx_HCDOR)	434
17.3.22 软件同步启动寄存器 (TIMx_SSTAR)	436
17.3.23 软件同步停止寄存器 (TIMx_SSTPR)	437
17.3.24 软件同步清零寄存器 (TIMx_SCLRR)	438
17.3.25 中断标志寄存器 (TIMx_IFR)	439
17.3.26 中断标志清除寄存器 (TIMx_ICLR)	440
17.3.27 展频及中断触发选择 (TIMx_CR)	441
17.3.28 AOS 选择控制寄存器 (TIMx_AOSSR)	443
17.3.29 AOS 选择控制寄存器标志清除 (TIMx_AOSCL)	444
17.3.30 端口刹车控制寄存器 (TIMx_PTBSK)	445
17.3.31 端口触发控制寄存器 (TIMx_TTRIG)	446
17.3.32 AOS 触发控制寄存器 (TIMx_ITRIG)	447
17.3.33 端口刹车极性控制寄存器 (TIMx_PTBKP)	448
18 实时时钟 (RTC)	449
18.1 实时时钟简介	449
18.2 实时时钟功能描述	450
18.2.1 上电设定	450
18.2.2 RTC 计数开始设定	450
18.2.3 系统低功耗模式切换	450
18.2.4 读出计数寄存器	451
18.2.5 写入计数寄存器	451
18.2.6 闹钟设定	452
18.2.7 1Hz 输出	452
18.2.8 时钟误差补偿	453
18.3 RTC 中断	454
18.3.1 RTC 闹钟中断	454
18.3.2 RTC 周期中断	454
18.4 RTC 寄存器描述	455
18.4.1 控制寄存器 0 (RTC_CRO)	456
18.4.2 控制寄存器 1 (RTC_CR1)	458
18.4.3 秒计数寄存器 (RTC_SEC)	459
18.4.4 分计数寄存器 (RTC_MIN)	459
18.4.5 时计数寄存器 (RTC_HOUR)	460

18.4.6	日计数寄存器 (RTC_DAY)	462
18.4.7	周计数寄存器 (RTC_WEEK)	463
18.4.8	月计数寄存器 (RTC_MON)	464
18.4.9	年计数寄存器 (RTC_YEAR)	464
18.4.10	秒闹钟寄存器 (RTC_ALMSEC)	465
18.4.11	分闹钟寄存器 (RTC_ALMMIN)	465
18.4.12	时闹钟寄存器 (RTC_ALMHOUR)	466
18.4.13	周闹钟寄存器 (RTC_ALMWEEK)	466
18.4.14	时钟误差补偿寄存器 (RTC_COMPEN)	467
19	看门狗定时器 (WDT)	469
19.1	WDT 简介.....	469
19.2	WDT 功能描述.....	470
19.2.1	WDT 溢出后产生中断.....	470
19.2.2	WDT 溢出后产生复位.....	470
19.3	WDT 寄存器描述	471
19.3.1	WDT 清除控制寄存器 (WDT_RST)	472
19.3.2	WDT_CON 寄存器	473
20	脉冲计数器 (PCNT)	474
20.1	脉冲计数器简介.....	474
20.2	脉冲计数器主要特性.....	474
20.3	脉冲计数器功能描述.....	475
20.3.1	整体框图	475
20.3.2	信号说明	475
20.3.3	计数模式	475
20.3.4	脉冲宽度滤波	478
20.3.5	超时	479
20.3.6	脉冲输入选择	479
20.3.7	Debug 观测输出	479
20.3.8	低功耗模式下的自动唤醒定时器	480
20.4	PCNT 寄存器描述	481
20.4.1	PCNT 启动寄存器(PCNT_RUN).....	482
20.4.2	PCNT 控制寄存器(PCNT_CTRL)	483
20.4.3	PCNT 滤波控制寄存器(PCNT_FLT)	484
20.4.4	PCNT 超时控制寄存器(PCNT_TOCR).....	484
20.4.5	PCNT 命令寄存器(PCNT_CMD)	485

20.4.6	PCNT 状态寄存器 1(PCNT_SR1)	485
20.4.7	PCNT 计数寄存器(PCNT_CNT)	486
20.4.8	PCNT 计数溢出寄存器(PCNT_TOP)	486
20.4.9	PCNT 计数溢出缓存寄存器(PCNT_BUF)	487
20.4.10	PCNT 中断标识寄存器(PCNT_IFR)	488
20.4.11	PCNT 中断清除寄存器(PCNT_ICR)	489
20.4.12	PCNT 中断使能寄存器(PCNT_IEN)	489
20.4.13	PCNT 同步状态寄存器(PCNT_SR2)	490
20.4.14	PCNT debug 观测输出选择寄存器(PCNT_DBG)	490
21	通用同步异步收发器 (UART)	491
21.1	简介	491
21.2	主要特性	491
21.3	功能描述	492
21.3.1	工作模式	492
21.3.2	波特率生成	497
21.3.3	帧错误检测	500
21.3.4	多机通讯	500
21.3.5	DMAC 硬件握手	501
21.3.6	硬件流控	502
21.3.7	收发端缓存	504
21.4	寄存器	505
21.4.1	数据寄存器 (UARTx_SBUF)	506
21.4.2	控制寄存器 (UARTx_SCON)	507
21.4.3	地址寄存器 (UARTx_SADDR)	508
21.4.4	地址掩码寄存器 (UARTx_SADEN)	508
21.4.5	标志位寄存器 (UARTx_ISR)	509
21.4.6	标志位清除寄存器 (UARTx_ICR)	509
21.4.7	波特率寄存器 (UARTx_SCNT)	510
22	低功耗同步异步收发器 (LPUART)	511
22.1	简介	511
22.2	主要特性	512
22.3	功能描述	512
22.3.1	配置时钟和传输时钟	512
22.3.2	工作模式	513
22.3.3	波特率生成	518

22.3.4 帧错误检测	521
22.3.5 多机通讯	521
22.3.6 DMAC 硬件握手	522
22.3.7 硬件流控	523
22.3.8 收发端缓存	525
22.4 寄存器	526
22.4.1 数据寄存器 (LPUARTx_SBUF)	527
22.4.2 控制寄存器 (LPUARTx_SCON)	528
22.4.3 地址寄存器 (LPUARTx_SADDR)	529
22.4.4 地址掩码寄存器 (LPUARTx_SADEN)	529
22.4.5 标志位寄存器 (LPUARTx_ISR)	530
22.4.6 标志位清除寄存器 (LPUARTx_ICR)	530
22.4.7 波特率寄存器 (LPUARTx_SCNT)	531
23 循环冗余校验 (CRC)	532
23.1 概述	532
23.2 主要特性	532
23.3 功能描述	532
23.3.1 工作模式	532
23.3.2 编码方式	532
23.3.3 写入位宽	532
23.4 编程示例	533
23.4.1 CRC-16 编码模式	533
23.4.2 CRC-16 检验模式	533
23.4.3 CRC-32 编码模式	533
23.4.4 CRC-32 检验模式	533
23.5 寄存器描述	534
23.5.1 寄存器列表	534
23.5.2 控制寄存器 (CRC_CR)	535
23.5.3 结果寄存器 (CRC_RESULT)	535
23.5.4 数据寄存器 (CRC_DATA)	536
24 真随机数发生器 (TRNG)	537
24.1 概述	537
24.2 功能框图	537
24.3 功能描述	537
24.4 寄存器	538

24.4.1	控制寄存器 (TRNG_CR)	539
24.4.2	模式寄存器 (TRNG_MODE)	540
24.4.3	数据寄存器 0 (TRNG_DATA0)	541
24.4.4	数据寄存器 1 (TRNG_DATA1)	541
24.5	软件基本操作	542
24.5.1	生成 64bits 真随机数的操作流程 (上电第一次)	542
24.5.2	生成 64bits 真随机数的操作流程 (非上电第一次生成)	543
25	高级加密标准模块 (AES)	544
25.1	功能定义	544
25.1.1	AES 算法简述	544
25.1.2	AES 模块功能描述	545
25.2	模块寄存器说明	546
25.2.1	控制寄存器 (AES_CR)	547
25.2.2	数据寄存器 (AES_Data)	548
25.2.3	密钥寄存器 (AES_Key)	549
25.3	异常机制	550
25.4	本模块操作说明	550
25.4.1	IP 操作的共同点	550
25.4.2	加密操作流程	550
25.4.3	解密操作流程	550
25.4.4	数据示例	551
25.5	运行时间说明	553
26	液晶控制器 (LCD)	554
26.1	LCD 简介	554
26.2	LCD 主要特性	554
26.3	LCD 框图	555
26.4	LCD 驱动波形	556
26.4.1	静态驱动波形	556
26.4.2	1/2Duty 1/2Bias 驱动波形	557
26.4.3	1/2Duty 1/3Bias 驱动波形	558
26.4.4	1/3Duty 1/2Bias 驱动波形	559
26.4.5	1/3Duty 1/3Bias 驱动波形	560
26.4.6	1/4Duty 1/2Bias 驱动波形	561
26.4.7	1/4Duty 1/3Bias 驱动波形	562
26.4.8	1/6Duty 1/3Bias 驱动波形	563

26.4.9	1/8Duty 1/3Bias 驱动波形	564
26.5	LCD Bias 产生电路	565
26.5.1	内部电阻模式	565
26.5.2	外部电容模式	566
26.5.3	外部电阻模式	566
26.6	DMA	567
26.7	中断	567
26.8	LCD 显示模式	567
26.8.1	LCD 显示模式 1 (MODE = 1)	568
26.8.2	LCD 显示模式 0 (MODE = 0)	570
26.9	LCD 寄存器	571
26.9.1	配置寄存器 0 (LCD_CRO)	572
26.9.2	配置寄存器 1 (LCD_CR1)	574
26.9.3	中断清除寄存器 (LCD_INTCLR)	575
26.9.4	输出配置寄存器 0 (LCD_POEN0)	575
26.9.5	输出配置寄存器 1 (LCD_POEN1)	576
26.9.6	LCD_RAM0~7	578
26.9.7	LCD_RAM8~F	578
27	Crystal-less USB 时钟校准器 (CTS)	579
27.1	48M 时钟校准	579
27.1.1	系统特性	579
27.1.2	框图	580
27.1.3	基准参考源同步输入	580
27.1.4	频率误差测量	580
27.1.5	频率误差微调	581
27.1.6	CTS 初始化和配置	582
27.2	其他振荡校准	583
27.2.1	系统特性	583
27.2.2	框图	583
27.2.3	校准说明	583
27.3	定时器功能说明	584
27.3.1	框图	584
27.3.2	定时器时钟选择	584
27.3.3	定时器	584
27.4	CTS 寄存器	585
27.4.1	CTS 控制寄存器 (CTS_CR)	586

27.4.2	CTS 配置寄存器 (CTS_CFGR)	587
27.4.3	CTS 中断和状态寄存器 (CTS_ISR)	588
27.4.4	CTS 中断状态清除寄存器 (CTS_ICR)	589
28	音频接口 (I2S)	590
28.1	功能说明	590
28.1.1	管脚复用	591
28.1.2	I2S 全双工	591
28.2	I2S 音频协议说明	592
28.2.1	I2S Philips 标准.....	593
28.2.2	MSB 对齐标准.....	595
28.2.3	LSB 对齐标准.....	597
28.2.4	PCM 标准.....	599
28.3	I2S 时钟发生器.....	601
28.4	I2S 工作模式	605
28.4.1	I2S 主模式	605
28.4.2	I2S 从模式	607
28.5	I2S 状态标志	608
28.5.1	忙标志 (BSY).....	608
28.5.2	发送缓冲区为空 (TXE_L TXE_R)	609
28.5.3	接收缓冲区非空 (RXNE_L RXNE_R)	609
28.5.4	下溢标志 (UDR)	609
28.5.5	上溢标志 (OVR)	609
28.5.6	帧错误标志 (FRE).....	609
28.6	I2S 中断 DMA	610
28.7	I2S 寄存器	611
28.7.1	I2Sx 控制寄存器 (I2Sx_CR)	612
28.7.2	I2Sx 状态寄存器 (I2Sx_SR)	613
28.7.3	I2Sx 配置寄存器 (I2Sx_CFGR)	615
28.7.4	I2Sx 预分频寄存器 (I2Sx_PR)	617
28.7.5	I2Sx 中断状态清除寄存器 (I2Sx_ICR)	618
28.7.6	I2Sx 数据寄存器 L (I2Sx_DRL)	618
28.7.7	I2Sx 数据寄存器 R (I2Sx_DRD)	619
29	USB 2.0 全速模块 (USBFS)	620
29.1	USBFS 简介	620
29.2	USBFS 主要特性	620

29.2.1	通用特性	620
29.2.2	设备模式特性	620
29.3	USBFS 系统框图	621
29.4	USBFS 管脚说明	621
29.5	USBFS 功能说明	622
29.5.1	USBFS 时钟以及工作模式	622
29.5.2	USBFS 设备功能	622
29.5.3	USBFS 功耗控制	625
29.5.4	USBFS 数据 FIFO.....	626
29.5.5	USBFS 设备 FIFO 架构.....	626
29.5.6	USBFS FIFO RAM 分配	627
29.5.7	USBFS 系统性能	628
29.5.8	USBFS 中断和事件	628
29.6	USBFS 编程模型	629
29.6.1	USBFS 模块初始化.....	629
29.6.2	USBFS 设备初始化.....	630
29.6.3	USBFS 设备软断开操作.....	631
29.6.4	USB 复位时的端点初始化.....	632
29.6.5	USB 枚举完成时的端点初始化.....	633
29.6.6	收到 SetAddress 命令时的端点初始化	633
29.6.7	收到 SetConfiguration/SetInterface 命令时的端点初始化	633
29.6.8	端点激活(Active)	634
29.6.9	端点不激活(Deactive).....	634
29.6.10	禁止(Disable)输出端点	635
29.6.11	暂停(Stall)非同步输出端点.....	635
29.6.12	全局输出端点 NAK 模式	636
29.6.13	输出端点传输停止	636
29.6.14	配置中断输入端点为周期或非周期端点 (共享 FIFO 模式)	636
29.6.15	全局非周期输入端点 NAK 模式 (共享 FIFO 模式)	637
29.6.16	发生输入端点 NAK.....	637
29.6.17	禁止非周期输入端点 (共享 FIFO 模式)	638
29.6.18	禁止周期输入端点 (共享 FIFO 模式)	638
29.6.19	非周期输入数据传输超时 (共享 FIFO 模式)	638
29.6.20	非同步输入端点暂停 (共享 FIFO 模式)	639
29.6.21	输入端点传输停止 (共享 FIFO 模式)	639

29.6.22 端点不匹配（共享 FIFO 模式）	640
29.6.23 控制传输	640
29.6.24 输出数据传输	648
29.6.25 非周期批量输入数据传输无阈值	648
29.6.26 非周期批量输入数据传输有阈值	648
29.6.27 非同步输出数据传输无阈值	649
29.6.28 非同步输出数据传输有阈值	649
29.6.29 同步输出数据传输有阈值	649
29.6.30 未完成同步输出数据传输	649
29.6.31 周期输入数据传输无阈值	650
29.6.32 周期输入数据传输有阈值	651
29.7 寄存器说明	652
29.7.1 USBFS 全局寄存器	654
29.7.2 USBFS 设备模式寄存器	670
29.7.3 USBFS 时钟门控控制寄存器	703
30 控制器局域网（CAN）	704
30.1 简介	704
30.2 CAN 系统框图	706
30.3 管脚说明	706
30.4 功能说明	707
30.4.1 波特率设定	707
30.4.2 发送缓冲器	708
30.4.3 接收缓冲器	709
30.4.4 接收筛选寄存器组	710
30.4.5 数据发送	711
30.4.6 单次数据发送	711
30.4.7 取消数据发送	711
30.4.8 数据接收	712
30.4.9 错误处理	712
30.4.10 节点关闭	713
30.4.11 仲裁失败位置捕捉	713
30.4.12 回环模式	713
30.4.13 静默模式	714
30.4.14 软件复位功能	715
30.4.15 向上兼容 CAN-FD 功能	716

30.4.16 时间触发 TTCAN	716
30.4.17 中断	719
30.5 寄存器说明.....	720
30.5.1 CAN 接收 BUF 寄存器 (CAN_RBUF)	722
30.5.2 CAN 发送 BUF 寄存器 (CAN_TBUF)	724
30.5.3 CAN 配置和状态寄存器 (CAN_CFG_STAT)	726
30.5.4 CAN 命令寄存器 (CAN_TCMD)	727
30.5.5 CAN 发送控制寄存器 (CAN_TCTRL)	729
30.5.6 CAN 接收控制寄存器 (CAN_RCTRL)	731
30.5.7 CAN 接收和发送中断使能寄存器 (CAN_RTIE)	732
30.5.8 CAN 接收和发送中断状态寄存器 (CAN_RTIF)	733
30.5.9 CAN 错误中断使能和标志寄存器 (CAN_ERRINT)	735
30.5.10 CAN 位时序寄存器 (CAN_BT)	736
30.5.11 CAN 错误和仲裁失败捕捉寄存器 (CAN_EALCAP)	737
30.5.12 CAN 警告限定寄存器 (CAN_LIMIT)	737
30.5.13 CAN 接收错误计数器寄存器 (CAN_RECNT)	738
30.5.14 CAN 发送错误计数器寄存器 (CAN_TECNT)	738
30.5.15 CAN 筛选器组控制寄存器 (CAN_ACFCTRL)	739
30.5.16 CAN 筛选器组使能寄存器 (CAN_ACFEN)	740
30.5.17 CAN 筛选器组 code 和 mask 寄存器 (CAN_ACF)	741
30.5.18 TTCAN TB slot 指针寄存器 (CAN_TBSLOT)	742
30.5.19 TTCAN 时间触发配置寄存器 (CAN_TTCFG)	743
30.5.20 TTCAN 参考消息寄存器 (CAN_REF_MSG)	744
30.5.21 TTCAN 触发配置寄存器 (CAN_TRG_CFG)	745
30.5.22 TTCAN 触发时间寄存器 (CAN_TRG_TRIG)	746
30.5.23 TTCAN 触发看门时间寄存器 (TRG_WTRIG)	746
30.6 使用注意事项	747
30.6.1 CAN 总线抗干扰措施	747
30.6.2 CAN 控制器噪声制约	747
31 模数转换器 (ADC)	748
31.1 模块简介	748
31.2 ADC 框图	748
31.3 转换时序及转换速度.....	749
31.4 单次转换模式	749
31.5 扫描转换模式	751

31.5.1	顺序扫描转换模式	751
31.5.2	插队扫描转换模式	753
31.5.3	扫描转换触发 DMA 读取	755
31.6	连续转换累加模式	756
31.7	ADC 转换外部触发源	758
31.8	ADC 转换结果比较	759
31.9	ADC 中断	760
31.10	使用温度传感器测量环境温度	761
31.11	ADC 模块寄存器	762
31.11.1	ADC 基本配置寄存器 0 (ADC_CR0)	764
31.11.2	ADC 基本配置寄存器 1 (ADC_CR1)	767
31.11.3	ADC 顺序扫描转换通道配置寄存器 0 (ADC_SQR0)	769
31.11.4	ADC 顺序扫描转换通道配置寄存器 1 (ADC_SQR1)	769
31.11.5	ADC 顺序扫描转换通道配置寄存器 2 (ADC_SQR2)	770
31.11.6	ADC 插队扫描转换通道配置寄存器 (ADC_JQR)	771
31.11.7	ADC 顺序扫描转换通道 x 转换结果 (ADC_SqrResult0 - 15)	771
31.11.8	ADC 插队扫描转换通道 x 转换结果 (ADC_JqrResult0 - 3)	772
31.11.9	ADC 转换结果 (ADC_Result)	772
31.11.10	ADC 转换结果累加值 (ADC_ResultAcc)	773
31.11.11	ADC 比较上阈值 (ADC_HT)	773
31.11.12	ADC 比较下阈值 (ADC_LT)	774
31.11.13	ADC 中断标志寄存器 (ADC_IFR)	775
31.11.14	ADC 中断清除寄存器 (ADC_ICR)	776
31.11.15	ADC 单次转换或顺序扫描转换外部中断触发源配置寄存器 (ADC_ExtTrigger0)	777
31.11.16	ADC 插队扫描转换外部中断触发源配置寄存器 (ADC_ExtTrigger1)	779
31.11.17	ADC 单次转换启动控制寄存器 (ADC_SglStart)	781
31.11.18	ADC 顺序扫描转换启动控制寄存器 (ADC_SqrStart)	781
31.11.19	ADC 插队扫描转换启动控制寄存器 (ADC_JqrStart)	782
31.11.20	ADC 一直转换启动停止控制寄存器 (ADC_allStart)	782
32	数模转换器 (DAC)	783
32.1	DAC 特性	783
32.2	功能说明	784
32.2.1	框图	784
32.2.2	DAC 输出缓冲器使能	785
32.2.3	DAC 通道使能	785

32.2.4	DAC 输出电压	785
32.2.5	DAC 触发选择	786
32.2.6	单模式功能说明	787
32.2.7	双模式功能说明	788
32.2.8	噪音生成	792
32.2.9	三角波生成	793
32.2.10	DMA 请求	794
32.3	DAC 寄存器	796
32.3.1	DAC 控制寄存器 (DAC_CR0)	797
32.3.2	DAC 软件触发寄存器 (DAC_SWTRIGR)	800
32.3.3	DAC0 12 位右对齐数据保持寄存器 (DAC_DHR12R0)	800
32.3.4	DAC0 12 位左对齐数据保持寄存器 (DAC_DHR12L0)	801
32.3.5	DAC0 8 位右对齐数据保持寄存器 (DAC_DHR8R0)	801
32.3.6	DAC1 12 位右对齐数据保持寄存器 (DAC_DHR12R1)	802
32.3.7	DAC1 12 位左对齐数据保持寄存器 (DAC_DHR12L1)	802
32.3.8	DAC1 8 位右对齐数据保持寄存器 (DAC_DHR8R1)	803
32.3.9	双 DAC 12 位右对齐数据保持寄存器 (DAC_DHR12RD)	803
32.3.10	双 DAC 12 位左对齐数据保持寄存器 (DAC_DHR12LD)	804
32.3.11	双 DAC 8 位右对齐数据保持寄存器 (DAC_DHR8RD)	804
32.3.12	DAC0 数据输出寄存器 (DAC_DOR0)	805
32.3.13	DAC1 数据输出寄存器 (DAC_DOR1)	805
32.3.14	DAC 状态寄存器 (DAC_SR)	806
32.3.15	DAC 触发选择寄存器 (DAC_ETRS)	807
33	模拟比较器 (VC)	808
33.1	模拟电压比较器 VC 简介	808
33.2	电压比较器框架图	809
33.3	建立/响应时间	810
33.4	滤波时间	810
33.5	迟滞功能	810
33.6	VC 寄存器	811
33.6.1	VC 配置寄存器 (VC_CR)	812
33.6.2	VC0 配置寄存器 (VC0_CR)	814
33.6.3	VC1 配置寄存器 (VC1_CR)	816
33.6.4	VC0 输出配置寄存器 (VC0_OUT_CFG)	818
33.6.5	VC1 输出配置寄存器 (VC1_OUT_CFG)	819

33.6.6	VC 中断寄存器 (VC_IFR)	820
33.6.7	VC2 配置寄存器 (VC2_CR)	821
33.6.8	VC2 输出配置寄存器 (VC2_OUT_CFG)	823
34	低电压检测器 (LVD)	824
34.1	LVD 简介	824
34.2	LVD 框图	824
34.3	迟滞功能	825
34.4	数字滤波	825
34.5	配置示例	826
34.5.1	LVD 配置为低电压复位	826
34.5.2	LVD 配置为电压变化中断	826
34.6	LVD 寄存器	827
34.6.1	LVD 配置寄存器 (LVD_CR)	828
34.6.2	LVD 中断寄存器 (LVD_IFR)	830
35	运算放大器 (OPA)	831
35.1	OPA 特性	831
35.2	OPA 功能描述	831
35.2.1	OPA0/1/2 (通用 OPA)	831
35.2.2	OPA3/4 --- (可复用为 DAC buffer*2)	832
35.3	OPA 校零	833
35.3.1	软件控制	833
35.3.2	软件触发控制	833
35.3.3	ADC 触发控制	833
35.4	OPA 寄存器	834
35.4.1	OPA 输出使能寄存器 (OPA_CR0)	835
35.4.2	OPA 控制寄存器 (OPA_CR1)	836
35.4.3	OPA 配置寄存器 (OPA_CR2)	837
35.4.4	OPA 校零控制寄存器 (OPA_CR)	838
36	模拟其它寄存器	840
36.1	BGR 配置寄存器 (BGR_CR)	840
37	SWD 调试接口	841
37.1	SWD 调试附加功能	841
37.2	ARM® 参考文档	842
37.3	调试端口引脚	842
37.3.1	SWD 端口引脚	842

37.3.2	SW-DP 引脚分配	842
37.3.3	SWD 引脚上的内部上拉	843
37.4	SWD 端口	843
37.4.1	SWD 协议简介	843
37.4.2	SWD 协议序列	843
37.4.3	SW-DP 状态机（复位、空闲状态、ID 代码）	844
37.4.4	DP 和 AP 读/写访问	844
37.4.5	SW-DP 寄存器	845
37.4.6	SW-AP 寄存器	845
37.5	内核调试	846
37.6	BPU (断点单元)	846
37.6.1	BPU 功能	847
37.7	DWT (数据观察点)	847
37.7.1	DWT 功能	847
37.7.2	DWT 程序计数器采样寄存器	847
37.8	MCU 调试组件 (DBG)	847
37.8.1	对低功耗模式的调试支持	847
37.8.2	对定时器、看门狗的调试支持	847
37.9	调试模式模块工作状态控制 (DEBUG_ACTIVE)	848
38	器件电子签名	850
38.1	产品唯一身份标识 (UID) 寄存器 (80 位)	850
38.2	产品型号寄存器	850
38.3	FLASH 容量寄存器	850
38.4	RAM 容量寄存器	851
38.5	管脚数量寄存器	852
39	附录 A SysTick 定时器	853
39.1	SysTick 定时器简介	853
39.2	设置 SysTick	853
39.3	SysTick 寄存器	853
39.3.1	SysTick 控制和状态寄存器 (CTRL)	854
39.3.2	SysTick 重载寄存器 (LOAD)	854
39.3.3	SysTick 当前值寄存器 (VAL)	854
39.3.4	SysTick 校准值寄存器 (CALIB)	854
40	附录 B 文档约定	856
40.1	寄存器相关缩写词列表	856

40.2 词汇表.....	856
版本修订记录	857

List of tables

表 1-1 地址划分表	42
表 2-1 运行模式下可运行模块图	45
表 2-2 休眠模式下可运行模块图	47
表 2-3 深度休眠模式下可运行模块图	48
表 5-1 Cortex-M0+ 处理器中断一览	86
表 5-2 外部中断与 NVIC 中断输入对应关系	91
表 6-1 端口状态真值表	107
表 6-2 端口复用表	110
表 7-1 I2C 时钟信号波特率	195
表 7-2 I2C 状态码表述	202
表 7-3 寄存器列表	208
表 8-1 SPI 管脚配置说明表	220
表 8-2 SPI 寄存器列表	224
表 8-3 主机模式波特率选择	225
表 9-1 寄存器列表	236
表 10-1 寄存器列表	243
表 11-1 FLASH 容量划分	247
表 11-2 不同频率下 FLASH 擦写时间参数	252
表 12-1 RAM 地址映射	263
表 12-2 寄存器基地址	264
表 14-1 Timer 寄存器列表	323
表 15-1 LPTimer 寄存器列表	358
表 16-1 PCA 比较捕获功能模块设置	371
表 16-2 PCA 寄存器列表	372
表 17-1 Advanced Timer 基本特性	380
表 17-2 Advanced Timer 端口列表	380
表 17-3 AOS 源选择	404
表 17-4 端口触发选择	405
表 17-5 Advanced Timer 寄存器列表	406
表 18-1 RTC 的基本特性	449
表 18-2 RTC 寄存器列表	455
表 19-1 WDT 寄存器列表	471
表 20-1 寄存器列表	481

表 21-1 Mode0/1/2/3 数据结构	492
表 21-2 B8 数据含义	492
表 21-3 PCLK=4MHz 波特率计算表	497
表 21-4 PCLK=8MHz 波特率计算表	498
表 21-5 PCLK=16MHz 波特率计算表	498
表 21-6 PCLK=24MHz 波特率计算表	499
表 21-7 PCLK=32MHz 波特率计算表	499
表 21-8 PCLK=48MHz 波特率计算表	500
表 22-1 Mode0/1/2/3 数据结构	513
表 22-2 B8 数据含义	514
表 22-3 SCL 为 4MHz 波特率计算表	518
表 22-4 SCLK 为 8MHz 波特率计算表	519
表 22-5 SCLK 为 16MHz 波特率计算表	519
表 22-6 SCLK 为 24MHz 波特率计算表	520
表 22-7 SCLK 为 32MHz 波特率计算表	520
表 22-8 SCLK 为 48MHz 波特率计算表	521
表 25-1 寄存器列表	546
表 25-2 128 位操作寄存器示例	551
表 25-3 192 位操作寄存器示例	552
表 25-4 192 位操作寄存器示例	553
表 25-5 AES 加解密运行时间	553
表 26-1 LCD 寄存器	571
表 27-1 CTS 寄存器	585
表 28-1 I2Sx 寄存器	611
表 29-1 USBFS 管脚说明	621
表 29-2 USBFS 中断事件表	628
表 29-3 USBFS 寄存器一览表	652
表 30-1 CAN 管脚说明	706
表 30-2 CAN 位时间设定规则	708
表 30-3 软件复位范围表	715
表 30-4 CAN 中断表	719
表 30-5 CAN 寄存器一览表	720
表 30-6 CAN 寄存器 BYTE/HALFWORD/WORD 访问排布表	721
表 30-7 标准格式 CAN 接收邮箱格式	722
表 30-8 扩展格式 CAN 接收邮箱格式	723

表 30-9 标准格式 CAN 发送邮箱格式	724
表 30-10 扩展格式 CAN 发送邮箱格式	725
表 31-1 ADC 寄存器.....	762
表 32-1 DAC 寄存器.....	796
表 33-1 VC 寄存器	811
表 34-1 LVD 寄存器	827
表 35-1 OPA 寄存器	834

List of figures

图 1-1 系统架构示意图	40
图 1-2 地址区域划分示意图	41
图 2-1 控制模式框图.....	43
图 3-1 时钟控制模块框图.....	51
图 3-2 晶振时钟启动示意图	53
图 3-3 时钟切换示意图	59
图 3-4 时钟校准原理图	60
图 4-1 复位来源示意图	79
图 5-1 只使用了高两位的优先级寄存器.....	86
图 5-2 中断向量表	87
图 5-3 中断激活和挂起状态	88
图 5-4 中断挂起状态被清除然后被重新确认.....	88
图 5-5 中断退出时若中断请求保持高电平就会引起中断处理的再次执行.....	89
图 5-6 中断处理中产生的中断挂起也可以被确认.....	89
图 5-7 中断结构图	92
图 6-1 端口电路图	105
图 6-2 AHB/FASTIO 总线端口随系统时钟的变化	108
图 6-3 读取端口引脚数据同步图	109
图 7-1 I2C 传输协议.....	191
图 7-2 START 和 STOP 条件	191
图 7-3 I2C 总线上位传输.....	192
图 7-4 I2C 总线上应答信号	192
图 7-5 I2C 总线上的仲裁	193
图 7-6 I2C 功能模块图	194
图 7-7 主发送模式数据同步图.....	196
图 7-8 I2C 主机发送状态图	197
图 7-9 主接收模式数据同步图.....	198
图 7-10 I2C 主机接收状态图	198
图 7-11 从接收模式数据同步图.....	199
图 7-12 从机接收状态图	199
图 7-13 从发送模式数据同步图.....	200
图 7-14 I2C 从机发送状态图	200
图 7-15 I2C 广播呼叫状态图	201

图 8-1 从机接收示意图	215
图 8-2 从机发送示意图	216
图 8-3 主机模式帧格式	216
图 8-4 从机 CPHA 为 0 时数据帧格式	217
图 8-5 从机 CHPA 为 1 时数据帧格式	217
图 8-6 SPI 多主机/多从机系统的示意图	219
图 9-1 时钟源选择示意图	232
图 9-2 时钟校准模块硬件示意图	232
图 9-3 时钟校准波形示意图	233
图 9-4 CLKTRIM 时钟选择	234
图 9-5 时钟监控波形示意图	235
图 13-1 功能框图	268
图 14-1 TIM0/1/2 框图	283
图 14-2 TIM3 框图	284
图 14-3 自由计数框图	285
图 14-4 重载计数波形	286
图 14-5 16 位重载计数波形	286
图 14-6 32 位自由计数波形	286
图 14-7 自由计数时序图	287
图 14-8 重载计数时序图 (预分频设置为 2)	287
图 14-9 PWC 测量框图	289
图 14-10 高电平脉冲宽度测量	289
图 14-11 下降沿到下降沿周期测量	290
图 14-12 上升沿到上升沿周期测量	290
图 14-13 上升沿到上升沿周期测量单次模式	291
图 14-14 计数器框图	292
图 14-15 无预分频的向上计数	293
图 14-16 带预分频的上计数	294
图 14-17 不带预分频的下计数	294
图 14-18 带预分频的下计数	295
图 14-19 带预分频的上下计数	295
图 14-20 边沿对齐计时器波形(DIR =1)	295
图 14-21 边沿对齐计时器波形(DIR =0)	296
图 14-22 中心对齐计数器波形	296
图 14-23 重复计数器产生更新时序	297

图 14-24 三角波模式下缓存使能.....	297
图 14-25 三角波模式下缓存无效.....	298
图 14-26 锯齿波模式下上计数缓存使能	298
图 14-27 锯齿波模式下上计数缓存无效	299
图 14-28 锯齿波模式下计数缓存使能.....	299
图 14-29 锯齿波模式下计数缓存无效.....	300
图 14-30 锯齿波模式下计数比较缓存使能.....	300
图 14-31 OCREF 输出框图	301
图 14-32 锯齿波计数单点比较 OCREF 输出波形 (OCMx=111)	302
图 14-33 三角波计数单点比较 OCREF 输出波形 (OCMx=111)	302
图 14-34 锯齿波计数双点比较 OCREF 输出 (OCMx=111)	303
图 14-35 三角波计数双点比较 OCREF 输出 (OCMx=111)	303
图 14-36 独立 PWM 输出框图	304
图 14-37 CCPx=0 时 PWM 输出波形	304
图 14-38 CCPx=1 时 PWM 输出波形	304
图 14-39 互补 PWM 输出框图	305
图 14-40 互补 PWM 输出波形图.....	305
图 14-41 互补 PWM 输出波形图.....	306
图 14-42 死区时间.....	306
图 14-43 三角波模式单脉冲计数.....	307
图 14-44 锯齿波上计数单脉冲模式	308
图 14-45 锯齿波下计数单脉冲模式	308
图 14-46 中断示意图	309
图 14-47 捕获功能框图.....	310
图 14-48 捕获时序图	310
图 14-49 CHA 端口选择	310
图 14-50 CHB 端口选择	311
图 14-51 从模式示意图	314
图 14-52 门控功能	314
图 14-53 触发和复位功能	315
图 14-54 编码计数	317
图 14-55 ADC 触发	318
图 15-1 LPTimer 框图	355
图 15-2 LPTimer 模式 1	356
图 15-3 LPTimer 模式 2	356

图 16-1 PCA 整体框图.....	362
图 16-2 PCA 计数器框图	364
图 16-3 PCA 捕获功能框图.....	366
图 16-4 PCA 比较功能框图.....	367
图 16-5 PCA 16 位 PWM 功能框图.....	368
图 16-6 PCA WDT 功能框图	369
图 16-7 PCA PWM 功能框图	370
图 16-8 PCA PWM 输出波形	371
图 17-1 Advanced Timer 框图	381
图 17-2 锯齿波波形（递加计数）	382
图 17-3 三角波波形	382
图 17-4 比较输出动作.....	383
图 17-5 捕获输入动作.....	384
图 17-6 捕获输入端口的滤波功能	385
图 17-7 软件同步动作.....	386
图 17-8 硬件同步动作.....	388
图 17-9 单缓存方式比较输出时序	389
图 17-10 PWM 展频输出示意图.....	390
图 17-11 CHA 输出 PWM 波	391
图 17-12 三角波 A 模式时软件设定 GCMBR 互补 PWM 波输出.....	391
图 17-13 三角波 B 模式时硬件设定 GCMBR 互补 PWM 波输出（对称死区）	392
图 17-14 6 相 PWM 波.....	393
图 17-15 三角波 A 模式时带死区时间三相互补 PWM 波输出.....	394
图 17-16 位置模式时基本计数动作	395
图 17-17 位置模式时相位差计数动作设定(1 倍).....	396
图 17-18 位置模式时相位差计数动作设定(2 倍).....	396
图 17-19 位置模式时相位差计数动作设定(4 倍).....	396
图 17-20 位置模式时方向计数动作	397
图 17-21 公转模式时 Z 相计数动作.....	397
图 17-22 公转模式时位置计数器输出计数动作.....	398
图 17-23 公转模式时 Z 相计数和位置计数器输出混合计数动作.....	398
图 17-24 公转计数模式-混合计数 Z 相屏蔽动作例 1.....	399
图 17-25 公转计数模式-混合计数 Z 相屏蔽动作例 2.....	399
图 17-26 周期间隔有效请求信号动作.....	400
图 17-27 端口刹车与软件刹车示意图.....	402

图 17-28 输出同高同低刹车示意图	402
图 17-29 VC 刹车控制示意图	403
图 17-30 Timer4/5/6 中断选择	403
图 18-1 RTC 框图	449
图 19-1 WDT 整体框图	469
图 20-1 整体框图	475
图 20-2 单通道脉冲记数模式加计数波形	476
图 20-3 单通道脉冲计数模式减计数波形	476
图 20-4 双通道非交脉冲计数模式加计数波形	477
图 20-5 双通道非交脉冲计数模式减计数波形	477
图 20-6 双通道正交脉冲计数模式加计数波形	478
图 20-7 双通道正交脉冲计数模式减计数波形	478
图 20-8 脉冲宽度滤波波形	479
图 21-1 结构框图	491
图 21-2 Mode0 发送数据	493
图 21-3 Mode0 接收数据	494
图 21-4 Mode1 发送数据	494
图 21-5 Mode1 接收数据	495
图 21-6 Mode2 发送数据	495
图 21-7 Mode2 接收数据	495
图 21-8 Mode3 发送数据	496
图 21-9 Mode3 接收数据	496
图 21-10 UART 硬件流控	502
图 21-11 nRTS 硬件流控信号	502
图 21-12 nCTS 硬件流控信号	503
图 21-13 接收缓存	504
图 21-14 发送缓存	504
图 22-1 结构框图	511
图 22-2 Mode0 发送数据	514
图 22-3 Mode0 接收数据	515
图 22-4 Mode1 发送数据	515
图 22-5 Mode1 接收数据	515
图 22-6 Mode2 发送数据	516
图 22-7 Mode2 接收数据	516
图 22-8 Mode3 发送数据	516

图 22-9 Mode3 接收数据	517
图 22-10 LPUART 硬件流控	523
图 22-11 nRTS 硬件流控信号	524
图 22-12 nCTS 硬件流控信号	524
图 22-13 接收缓存	525
图 22-14 发送缓存	525
图 24-1 TRNG 数据流	537
图 25-1 AES 的加解密示意图	544
图 25-2 AES 的加密流程图	545
图 27-1 CTS 计数器	581
图 28-1 功能框图	590
图 28-2 Philip 协议波形 (16/32 位全精度)	593
图 28-3 Philip 协议波形 (24 位帧)	593
图 28-4 24 位数据发送写操作	594
图 28-5 24 位数据模式接收读操作	594
图 28-6 Philip 协议 16 位扩展到 32 位数据帧	594
图 28-7 16 位数据读写操作	595
图 28-8 MSB 协议波形 (16/32 位全精度)	595
图 28-9 MSB 协议波形 (24 位帧)	596
图 28-10 MSB 协议 16 位扩展到 32 位数据帧	596
图 28-11 LSB 协议波形 (16/32 位全精度)	597
图 28-12 LSB 协议波形 (24 位帧)	597
图 28-13 LSB 24 位数据发送写操作	598
图 28-14 LSB 24 位数据模式接收读操作	598
图 28-15 LSB 协议 16 位扩展到 32 位数据帧	598
图 28-16 16 位数据读写操作	599
图 28-17 PCM 协议波形 (16 位)	599
图 28-18 PCM 协议波形 (16 位数据扩展到 32 位)	600
图 28-19 音频采样频率定义	601
图 28-20 时钟产生	601
图 29-1 USBFS 系统框图	621
图 29-2 USBFS 设备模式系统构建图	622
图 29-3 USBFS 设备模式下 FIFO 架构示意图	626
图 30-1 CAN 系统框图	706
图 30-2 CAN 位时间定义图	707

图 30-3 CAN TBUF 寄存器写发送缓冲器和示意图	708
图 30-4 CAN RBUF 寄存器读接收缓冲器示意图	709
图 30-5 CAN ACF 寄存器访问筛选器组示意图	710
图 30-6 CAN LBMI 和 LBME 示意图	714
图 31-1 ADC 示意框图	748
图 31-2 ADC 转换时序图	749
图 31-3 ADC 顺序扫描转换过程示例	751
图 31-4 ADC 插队扫描转换过程示例	753
图 31-5 ADC 顺序扫描过程中进行插队扫描示例	755
图 31-6 ADC 连续转换累加过程示例	756
图 31-7 ADC 单次转换或顺序扫描转换外部触发源示意图	758
图 31-8 ADC 插队扫描转换外部触发源示意图	759
图 31-9 ADC 转换结果	760
图 32-1 触发不使能时 DAC 转换时序图	787
图 32-2 LFSR 产生波形的 DAC 转换（使用软件触发）	793
图 32-3 生成 DAC 三角波	794
图 32-4 生成三角波波形的 DAC 转换（使能软件触发）	794
图 33-1 VC 框架图	809
图 33-2 VC 滤波响应时间	810
图 33-3 VC 迟滞功能	810
图 34-1 LVD 框图	824
图 37-1 调试支持框图	841

Overview

The HC32L072/HC32L073 series is an ultra-low power, wide voltage range MCU designed to extend the battery life of portable measurement systems. It integrates a 12-bit 1Msps high-precision SARADC, 2 12-bit DACs, and integrates comparators, op amps, built-in high-performance PWM timers, LCD displays, multiple UARTs, SPI, I2C, I2S, USB, CAN and other rich communication peripherals, built-in AES, TRNG and other information security modules, with high integration, high anti-interference, high reliability and ultra-low power consumption. The core of this product adopts the Cortex-M0+ core, with mature Keil & IAR debugging and development software, supports C language and assembly language, assembly instructions.

Typical applications of ultra-low power MCU

- Sensor applications, IoT applications
- Smart meters, wireless modules, thermostats, shelf labels
- Smart transportation, alarm systems
- Smart homes, medical equipment

About this manual

This manual mainly introduces the functions, operation matters and usage methods of the chip. For the specifications of the chip, please refer to the corresponding "Data Sheet".

1 System Structure

1.1 Overview

This product system consists of the following parts:

- 2 AHB bus Masters:
 - Cortex-M0+
 - DMA controller
- 4 AHB bus Slaves:
 - FLASH memory
 - SRAM memory
 - AHB0, AHB to APB Bridge, including all APB interface peripherals
 - AHB1, including all AHB interface peripherals

The entire system bus structure is implemented using multi-level AHB-lite bus interconnection, as shown in the following figure:

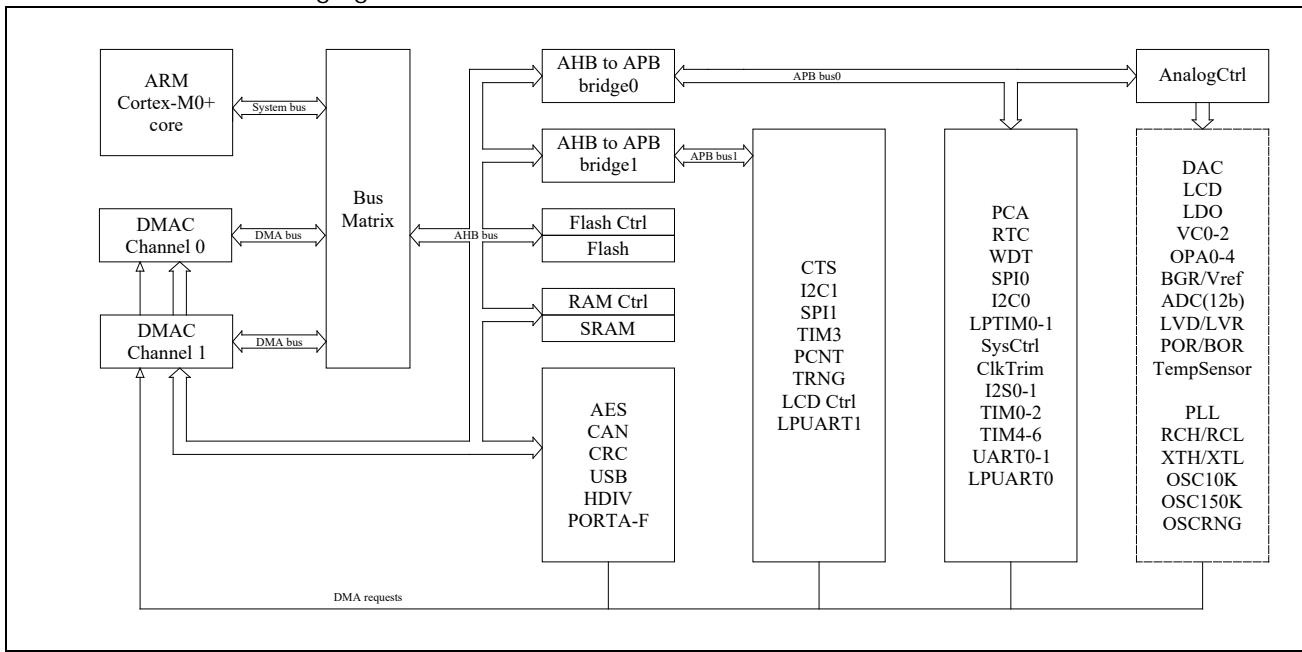


Figure 1-1 System architecture diagram

1.2 System memory map

The address area division of the entire series system is shown in the following figure:

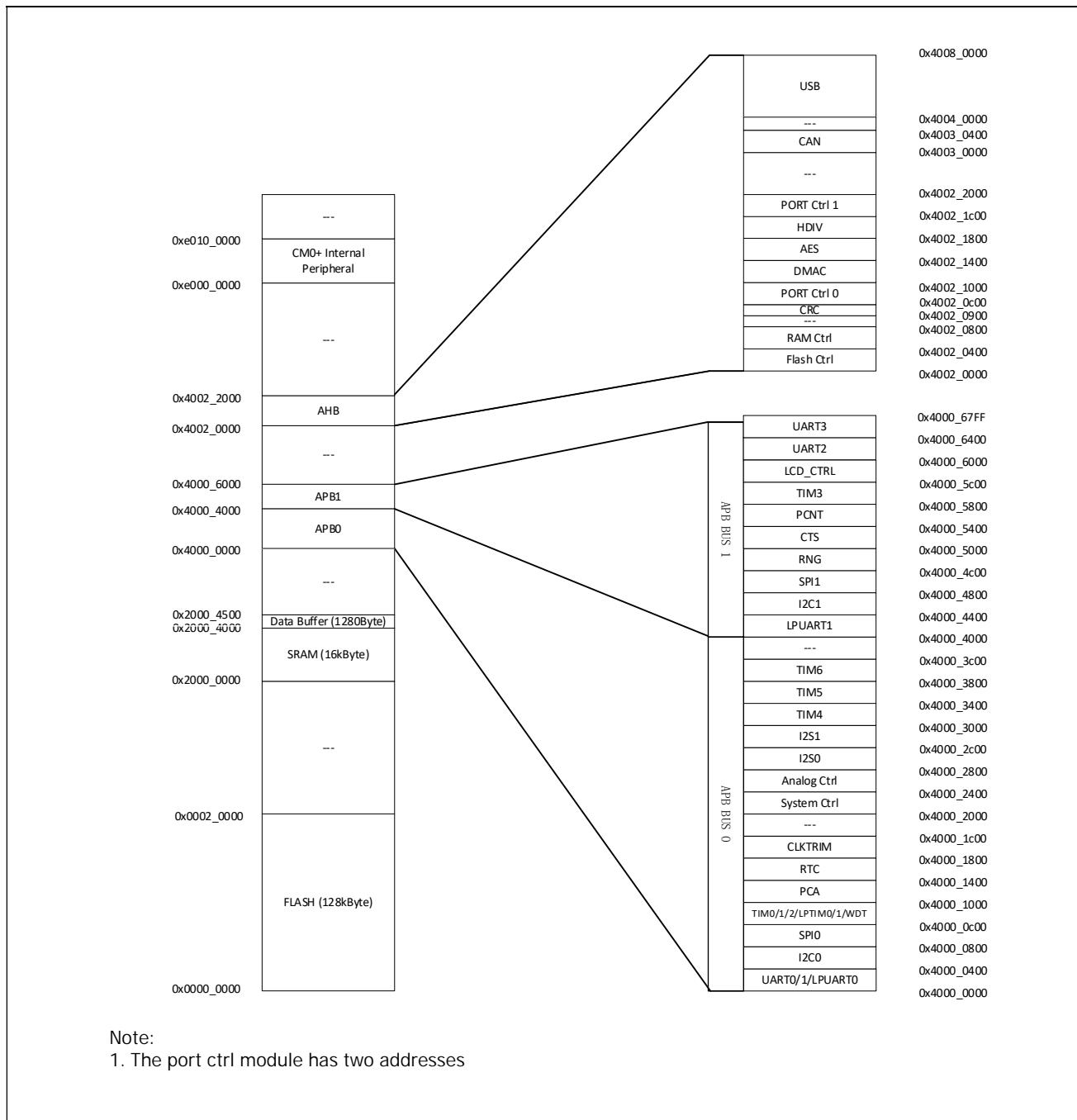


Figure 1-2 Schematic diagram of address area division

1.3 Memory and modules address allocation

Table 1-1 Address allocation table

Boundary Address	Size	Memory Area	Description
0x0000_0000 - 0x0000_FFFF	128KByte	FLASH	
0x0001_0000 - 0x1FFF_FFFF	-	Reserved	
0x2000_0000 - 0x2000_1FFF	16KByte	SRAM	
0x2000_2000 - 0x3FFF_FFFF	-	Reserved	
0x4000_0000 - 0x4000_00FF	256Byte	UART0	
0x4000_0100 - 0x4000_01FF	256Byte	UART1	
0x4000_0200 - 0x4000_02FF	256Byte	LPUART0	
0x4000_0300 - 0x4000_03FF	-	Reserved	
0x4000_0400 - 0x4000_07FF	1KByte	I2C0	
0x4000_0800 - 0x4000_0BFF	1KByte	SPI0	
0x4000_0C00 - 0x4000_0CFF	256Byte	TIM0	
0x4000_0D00 - 0x4000_0DFF	256Byte	TIM1	
0x4000_0E00 - 0x4000_0EFF	256Byte	TIM2	
0x4000_0F00 - 0x4000_0F7F	128Byte	LPTIMO-1	
0x4000_0F80 - 0x4000_0FFF	128Byte	WDT	
0x4000_1000 - 0x4000_13FF	1KByte	PCA	
0x4000_1400 - 0x4000_17FF	1KByte	RTC	
0x4000_1800 - 0x4000_1BFF	1KByte	CLKTRIM	
0x4000_1C00 - 0x4000_1FFF	-	Reserved	
0x4000_2000 - 0x4000_23FF	1KByte	SYSCTRL	
0x4000_2400 - 0x4000_27FF	1KByte	ANALOGCTRL	
0x4000_2800 - 0x4000_2BFF	1KByte	I2S0	
0x4000_2C00 - 0x4000_2FFF	1KByte	I2S1	
0x4000_3000 - 0x4000_33FF	1KByte	TIM4	
0x4000_3400 - 0x4000_37FF	1KByte	TIM5	
0x4000_3800 - 0x4000_3BFF	1KByte	TIM6	
0x4000_3C00 - 0x4000_3FFF	-	Reserved	
0x4000_4000 - 0x4000_43FF	1KByte	LPUART1	
0x4000_4400 - 0x4000_47FF	1KByte	I2C1	
0x4000_4800 - 0x4000_4BFF	1KByte	SPI1	
0x4000_4C00 - 0x4000_4FFF	1KByte	TRNG	
0x4000_5000 - 0x4000_53FF	1KByte	CTS	
0x4000_5400 - 0x4000_57FF	1KByte	PCNT	
0x4000_5800 - 0x4000_5BFF	1KByte	TIM3	
0x4000_5C00 - 0x4000_5FFF	1KByte	LCD	
0x4000_6000 - 0x4000_63FF	1KByte	UART2	
0x4000_6400 - 0x4000_67FF	1KByte	UART3	
0x4000_6800 - 0x4001_FFFF	-	Reserved	
0x4002_0000 - 0x4002_03FF	1KByte	FLASH CTRL	
0x4002_0400 - 0x4002_07FF	1KByte	RAM CTRL	
0x4002_0800 - 0x4002_08FF	256Byte	Reserved	
0x4002_0900 - 0x4002_0BFF	768Byte	CRC	
0x4002_0C00 - 0x4002_0FFF	1KByte	PORT CTRL	
0x4002_1000 - 0x4002_13FF	1KByte	DMAC	
0x4002_1400 - 0x4002_17FF	1KByte	AES	
0x4002_1800 - 0x4002_1BFF	1KByte	HDIV	
0x4002_1C00 - 0x4002_1FFF	1KByte	PORT CTRL1	
0x4003_0000 - 0x4003_03FF	1KByte	CAN	
0x4004_0000 - 0x4007_FFFF	256KByte	USB	

2 Operating modes

The power management module of this product is responsible for managing the switching between various operating modes of this product and controlling the working status of each functional module in each operating mode. The operating voltage (VCC) of this product is 1.8V ~ 5.5V.

This product has the following working modes:

- 1) Running mode: CPU running, peripheral functional modules running.
- 2) Sleep mode: CPU stops running, peripheral functional modules running.
- 3) Deep sleep mode: CPU stops running, high-speed clock stops running.

From the running mode, by executing the software program, you can enter other low-power modes. From other various low-power modes, you can return to the running mode through interrupt triggering.

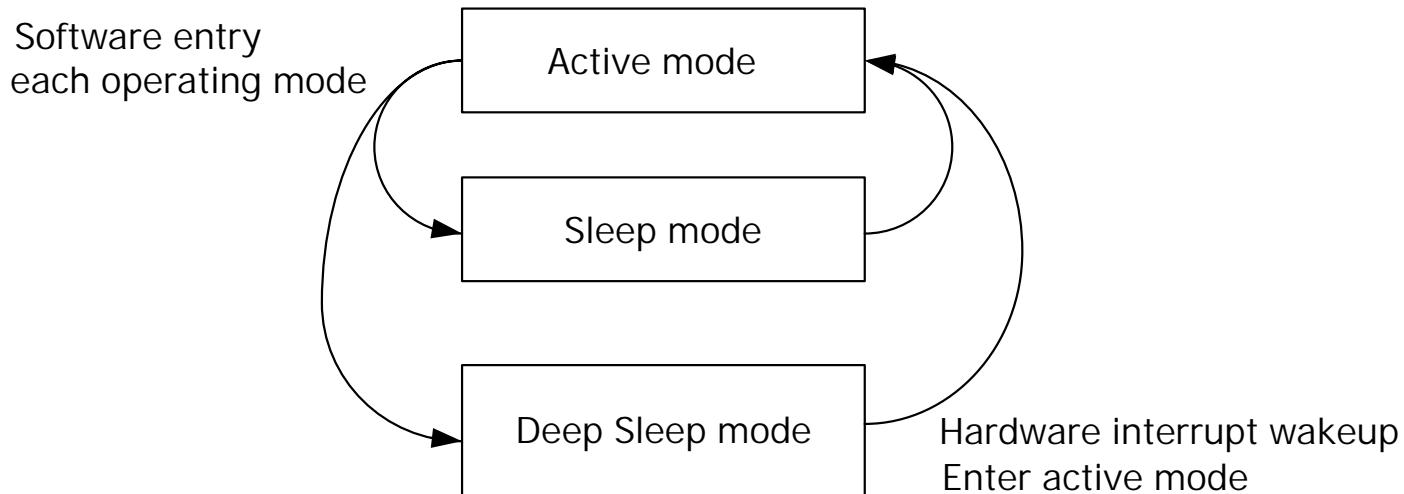


Figure 2-1 Control mode block diagram

In each mode, the CPU can respond to all interrupt types:

Interrupt vector	Interrupt source	Active mode	Sleep Mode	Deep Sleep Mode
[0]	GPIO_PA	✓	✓	✓
[1]	GPIO_PB	✓	✓	✓
[2]	GPIO_PC/GPIO_PE	✓	✓	✓
[3]	GPIO_PD/GPIO_PF	✓	✓	✓
[4]	DMA	✓	✓	
[5]	TIM3	✓	✓	
[6]	UART0	✓	✓	
[7]	UART1	✓	✓	
[8]	LPUART0	✓	✓	✓
[9]	LPUART1	✓	✓	✓
[10]	SPI0/I2S0	✓	✓	
[11]	SPI1/I2S1	✓	✓	
[12]	I2C0	✓	✓	
[13]	I2C1	✓	✓	
[14]	TIM0	✓	✓	
[15]	TIM1	✓	✓	
[16]	TIM2	✓	✓	
[17]	LPTIM0/LPTIM1	✓	✓	✓

[18]	TIM4	✓	✓	
[19]	TIM5	✓	✓	
[20]	TIM6	✓	✓	
[21]	PCA	✓	✓	
[22]	WDT	✓	✓	✓
[23]	RTC	✓	✓	✓
[24]	ADC/DAC	✓	✓	
[25]	PCNT	✓	✓	✓
[26]	VC0/VC1/VC2/LVD	✓	✓	✓
[27]	USB	✓	✓	
[28]	CAN	✓	✓	
[29]	LCD	✓	✓	✓
[30]	FLASH/RAM	✓	✓	
[31]	CLKTRIM/CTS	✓	✓	✓

In each mode, the product can respond to all reset types:

	Reset source	Active mode	Sleep Mode	Deep Sleep Mode
[0]	Power-on reset POR	✓	✓	✓
[1]	External reset pin reset	✓	✓	✓
[2]	LVD reset	✓	✓	✓
[3]	WDT reset	✓	✓	✓
[4]	PCA reset	✓	✓	
[5]	Cortex-M0+ LOCKUP hardware reset	✓		
[6]	Cortex-M0+ SYSRESETREQ software reset	✓		

2.1 Active Mode

Active Mode:

After the system is reset at power-on or wakes up from low power mode, the microcontroller MCU is in operation. When the CPU does not need to continue running, it can use a variety of low power modes to save energy, such as when waiting for an external event. Users need to select the best low power mode based on the lowest energy consumption, fastest startup time, available wake-up sources, etc.

Table 2-1 Diagram of modules that can be run in active mode

Active Mode		
Cortex-M0+	SWD	XTH
FLASH	CAN	RCH
RAM	USB	PLL
DMAC	UART0-3	ADC
TIM0-3	SPI0-1	DAC
TIM4-6	I2C0-1	TRNG
PCA	I2S0-1	OPA0-4
HDIV	CRC	LVD
AES	XTL	RESET
LPUART0-1	RCL	POR/BOR
LPTIM0-1	RTC	VC0-2
PCNT	LCD	WDT
GPIO	CLKTRIM	CTS

Several ways to reduce chip power consumption in active mode:

- 1) In active mode, by programming the prescaler register (SYSCTRL0.HCLK_PRS, SYSCTRL0.PCLK_PRS), the speed of any system clock (HCLK, PCLK) can be reduced. The prescaler can also be used to reduce the clock of peripherals before entering sleep mode.
- 2) In active mode, turn off the clock of unused peripherals (PERIx_CLKx) to reduce power consumption.
- 3) In active mode, turn off the clock of unused peripherals (PERIx_CLKx) to reduce power consumption, and let the system enter sleep mode to reduce power consumption more, and turn off the clock of unused peripherals (PERIx_CLKEN.x) before executing the WFI instruction.
- 4) Use low power mode instead of sleep mode, because the wake-up time of this product is very short (~4us), which can also meet the real-time response requirements of the system.

2.2 Sleep Mode

Sleep Mode of this product

The WFI instruction can be used to enter sleep mode. In sleep mode, the CPU stops running, but the clock module, system clock, NVIC interrupt processing and peripheral functional modules can still work.

When the system enters sleep state, the port status will not be changed. Before entering sleep, the IO status will be changed to the sleep state as needed.

- How to enter sleep mode:

Enter sleep state by executing the WFI instruction. According to the value of the SLEEPONEXIT bit in the Cortex-M0+ system control register, there are two options for selecting the sleep mode entry mechanism:

SLEEP-NOW: If the SLEEPONEXIT bit is cleared, the microcontroller immediately enters sleep mode when WFI or WFE is executed.

SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the microcontroller immediately enters sleep mode when the system exits from the lowest priority interrupt handler.

- How to exit sleep mode:

If the WFI instruction is executed to enter sleep mode, any peripheral interrupt responded by the high-priority nested vectored interrupt controller can wake up the system from sleep mode.

Notes:

- 1) SLEEP-ON-EXIT: Set this bit to 1, and the program will automatically enter sleep after executing the interrupt. The program does not need to write `_wfi()`;
- 2) SLEEP-ON-EXIT: Set this bit to 0, and `main()` will enter sleeping after executing `_wfi()`. After the interrupt is triggered and the interrupt program is executed and returned to `main()`, it will enter sleeping after executing the WFI instruction. Wait for subsequent interrupt triggers.
- 3) The SLEEP-ON-EXIT bit does not affect the execution of the `_wfi()` instruction. SLEEP-ON-EXIT = 0: `main()` will enter sleeping after executing `wfi()`. After the interrupt is triggered and the interrupt program is executed and returned to `main()`, it will continue to execute;
- 4) If you enter sleep during an interrupt, only interrupts with a higher priority than this interrupt can wake you up. The higher priority will be executed first, and then the lower priority will be executed; interrupts with a priority lower than or equal to this interrupt cannot wake you up.

Table 2-2 Diagram of modules that can be run in sleep mode

Sleep Mode		
Cortex-M0+	SWD	XTH
FLASH	CAN	RCH
RAM	USB	PLL
DMAC	UART0-3	ADC
TIM0-3	SPI0-1	DAC
TIM4-6	I2C0-1	TRNG
PCA	I2S0-1	OPA0-4
HDIV	CRC	LVD
AES	XTL	RESET
LPUART0-1	RCL	POR/BOR
LPTIM0-1	RTC	VC0-2
PCNT	LCD	WDT
GPIO	CLKTRIM	CTS

Note: The modules in the gray dashed box do not work in the current state.

2.3 Deep Sleep Mode

This product has deep sleep mode.

Use SLEEPDEEP with the WFI instruction to enter deep sleep mode. In deep sleep mode, the CPU stops running, the high-speed clock is turned off, the low-speed clock can be configured to run or not, and some low-power peripheral modules can be configured to allow or not. NVIC interrupt processing can still work.

- The system enters deep sleep mode from the high-speed clock, the high-speed clock is automatically turned off, and the low-speed clock remains in the state before entering deep sleep.
- The system enters deep sleep mode from the low-speed clock. Since the low-speed clock will not be turned off automatically, it remains running and enters sleep mode. Only ARM Cortex-M0+ does not run, and other modules are running.
- When the system clock switches, all clocks will not be turned off automatically. The corresponding clocks need to be turned off and on by software according to power consumption and system requirements.
- When the system enters deep sleep state, the port state will not be changed. Before entering sleep, change the IO state to the sleep state as needed.

How to enter deep sleep mode:

First, set the SLEEPDEEP bit in the Cortex-M0+ system control register and enter sleep mode by executing the WFI instruction. Depending on the value of the SLEEPONEXIT bit in the Cortex™-M0+ system control register, there are two options for selecting the deep sleep mode entry mechanism:

SLEEP-NOW: If the SLEEPONEXIT bit is cleared, the microcontroller enters sleep mode immediately when WFI or WFE is executed.

SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the microcontroller enters sleep mode immediately when the system exits from the lowest priority interrupt handler.

How to exit deep sleep mode:

If the WFI instruction is executed to enter sleep mode, any peripheral interrupt responded by the nested vector interrupt controller (peripheral module interrupt that can be run in Deep Sleep) can wake up the system from sleep mode.

For wake-up settings, refer to 3.4 Interrupt wake-up control WIC.

Table 2-3 Diagram of modules that can be run in deep sleep mode

DeepSleep Mode		
Cortex-M0+	SWD	XTH
FLASH	CAN	RCH
RAM	USB	PLL
DMAC	UART0-3	ADC
TIM0-3	SPI0-1	DAC
TIM4-6	I2C0-1	TRNG
PCA	I2S0-1	OPA0-4
HDIV	CRC	LVD
AES	XTL	RESET
LPUART0-1	RCL	POR/BOR
LPTIM0-1	RTC	VC0-2
PCNT	LCD	WDT
GPIO	CLKTRIM	CTS

Note: The modules in the gray dashed box do not work in the current state.

System Control Register (Cortex-M0+ Core System Control Register)

Address: 0xE000ED10

Reset value: 0x0000 0000

Bit	Function	Description	R/W
31:5	RESERVED	Reserved	
4	SEVONPEND	When set to 1, each new interrupt will generate an event, which can be used to wake up the processor if WFE sleep is used.	RW
3	RESERVED	Reserved	
2	SLEEPDEEP	When set to 1, execute WFI to enter deep sleep, and the product enters Deep sleep mode. When set to 0, execute WFI to enter sleep, and the product enters sleep/Idle mode.	RW
1	SLEEPONEXIT	When set to 1, when exiting exception handling and returning to the program thread, the processor automatically enters sleep mode (WFI). When set to 0, this feature will be automatically disabled.	RW
0	RESERVED	Reserved	

After entering deep sleep, there are two options for the system clock after waking up. The default clock used when entering deep sleep is used. After the configuration register SYSCTRL0.wakeup_byRCH is set to 1, no matter what clock was used before entering deep sleep, the internal high-speed clock RCH is used after waking up. If an external crystal oscillator is used, this setting can speed up the system wake-up.

3 System Controller (SYSCTRL)

3.1 System Clock Introduction

The clock control module mainly controls the system clock and peripheral clock. It can configure different clock sources as system clocks, configure different system clock frequency divisions, and enable or disable peripheral clocks. To ensure the accuracy of the oscillator, the internal clocks have a calibration function.

This product supports the following five different clock sources as system clocks:

- Internal high-speed RC clock RCH (output frequency is 4~24MHz)
- Internal low-speed RC clock RCL (38.4K and 32.8K configurable)
- External high-speed crystal clock XTH
- External low-speed crystal clock XTL
- Phase-locked loop clock PLL

Note 1: When switching the clock source of the system clock, please strictly follow the operating steps to switch, see Section 3.2 for details.

Note 2: XTL can be connected without a crystal oscillator, and a 32.768kHz clock signal can be directly input from the PC14 pin. XTH can be connected without a crystal oscillator, and a 4~32MHz clock signal can be directly input from the PF00 pin.

This product also contains the following three auxiliary clocks:

- Internal low-speed 10K clock, only used by the watchdog and CLKTRIM modules.
- Internal 150K clock, only used by the LVD and VC modules.
- Internal 48M clock, only used by the USB module, can be automatically calibrated using the CTS module.

The figure below shows the clock architecture of this product.

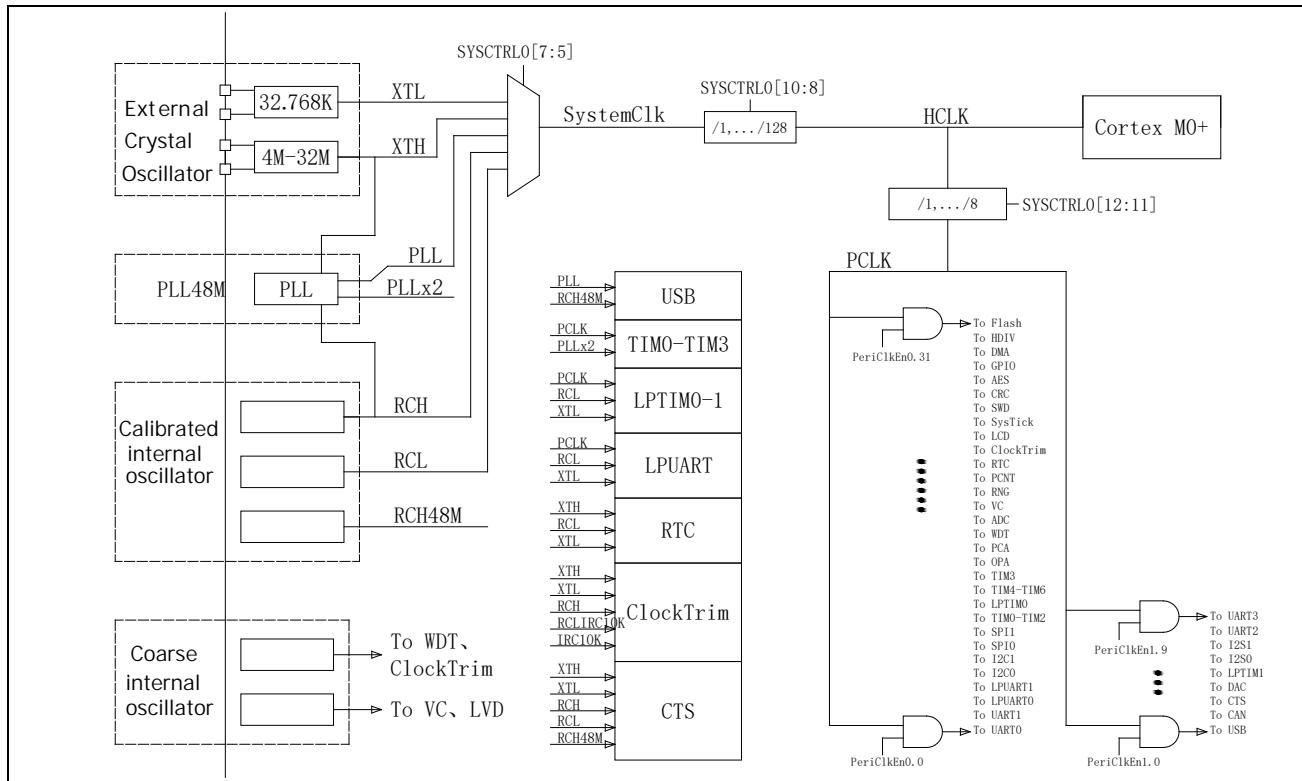


Figure 3-1 Clock control module block diagram

3.1.1 Internal high-speed RC clock RCH

The default clock source after the chip is powered on or reset is the internal high-speed clock with a frequency of 4MHz; when the system enters Deep Sleep, this high-speed clock will be automatically turned off.

The output frequency of RCH can be adjusted by changing the value of register RCH_CR[10:0]. The output frequency of RCH increases by about 0.2% for each increase of the register value, and the total adjustment range is 4~24MHz.

The 5 frequencies that have been pre-adjusted at the factory are 4MHz, 8MHz, 16MHz, 22.12MHz, and 24MHz; if other frequencies are required, please manually adjust the value of this register.

Changing the RCH output frequency requires a specific change sequence, see the system clock switching section for details.

The internal high-speed clock only takes 4us from startup to stability. In order to respond to interrupts quickly in deep sleep mode, it is recommended to switch the system clock to RCH before entering deep sleep mode.

3.1.2 Internal low speed RC clock RCL

The internal low speed clock can select its output frequency through register RCL_CR[9:0]. The available frequencies are 38.4KHz and 32.768KHz. When the system enters DeepSleep, this low speed clock will not be automatically turned off. Ultra-low power peripheral modules can select RCL as their clock.

3.1.3 External low speed crystal clock XTL

The external low speed crystal clock needs to be connected to a 32.768KHz low power crystal oscillator, which has ultra-high precision and ultra-low power consumption. When the system enters Deep Sleep, this low speed clock will not be automatically turned off. Peripheral modules working in ultra-low power mode can choose XTL as their clock.

XTL can also be connected without a crystal oscillator, and directly input a 32.768KHz clock signal from the PC14 pin. The method to input the clock signal from PC14 is: configure the PC14 pin as a GPIO input; set SYSCTRL1. EXTL_EN to 1.

Note:

- The crystal and its matching devices must meet the relevant requirements of the low speed external clock XTL in the electrical characteristics of the data sheet.

3.1.4 External high-speed crystal clock XTH

The external high speed crystal clock needs to be connected to a 4~32MHz high speed crystal oscillator. When the system enters Deep Sleep, this high speed clock will be automatically turned off.

XTH can also be connected without a crystal oscillator, and directly input a 4~32MHz clock signal from the PF00 pin. The method to input a clock signal from PF00 is: configure the PF00 pin as a GPIO input; set SYSCTRL1. EXTH_EN to 1.

Notes:

- The crystal and its matching devices must meet the relevant requirements of the high-speed external clock XTH in the electrical characteristics of the data sheet.
- To ensure the high reliability of XTH operation, it is strongly recommended to use an external active crystal oscillator.

3.1.5 Phase-locked loop clock PLL

The built-in PLL supports 8~48MHz clock output. The reference clock sources of PLL are: RCH, XTH crystal clock, PF00 pin input clock.

PLL has a power consumption of about tens of microamperes in deep sleep mode. It is recommended to configure the system clock to RCH/XTH before entering deep sleep mode and turn off PLL and BGR, and reconfigure the system clock to PLL after waking up.

3.1.6 Internal high-speed clock RCH48M

The internal high-speed clock RCH48M can be used as the clock of crystall-less USB, and can be automatically calibrated using the module CTS. It cannot be used as the system clock.

3.1.7 Clock startup process

The above five clock sources all require startup stabilization time. The following figure takes external XTH as an example to illustrate the clock startup stabilization process.

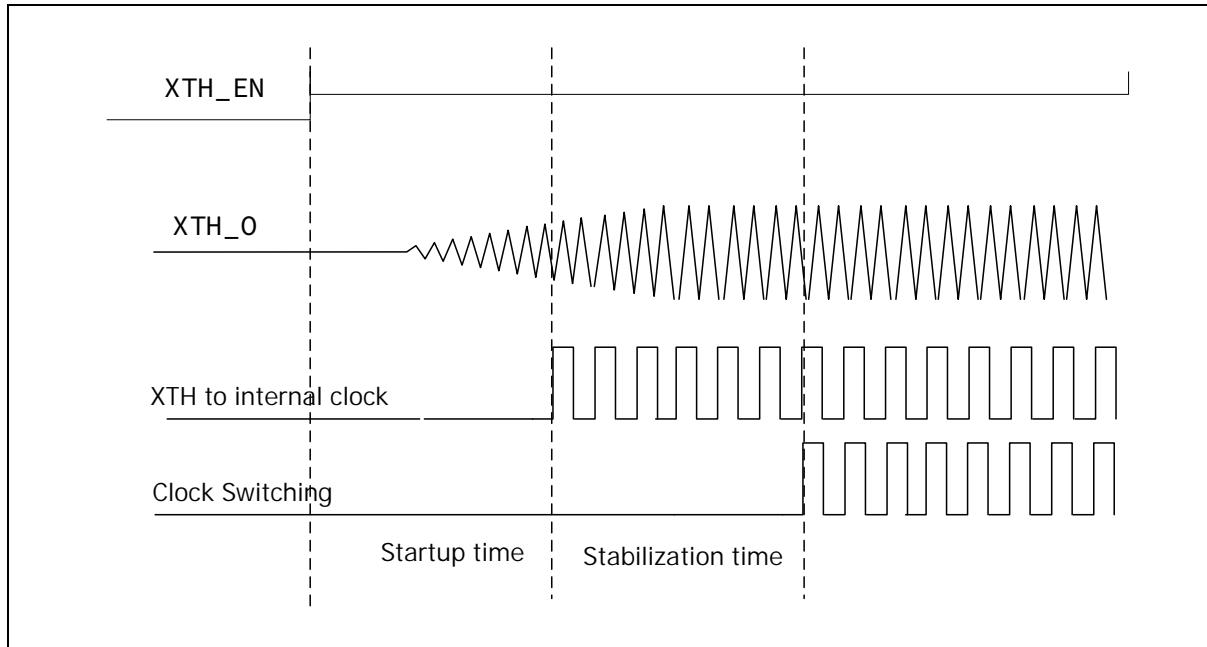


Figure 3-2 Crystal Oscillator Clock Startup Diagram

3.2 System Clock Switching

The switching of the clock source is controlled by register SYSCTRL0[7:0]. The clock source of the system clock can be switched between RCH, RCL, XTH, XTL, and PLL through SYSCTRL0[7:5]. During the switching, any two of the four clock sources RCH, RCL, XTH, and XTL can be switched with each other; PLL can only switch with the two clocks RCH and XTH. The clock switching operation must be performed according to the 7 clock switching processes described below, otherwise anomalies may occur.

When switching the clock, FLASH_CR.WAIT needs to be configured synchronously. If the clock frequency is not greater than 24MHz, FLASH_CR.WAIT should be set to 0; if the clock frequency is greater than 24MHz, FLASH_CR.WAIT should be set to 1; if the clock frequency is greater than 48MHz, FLASH_CR.WAIT should be set to 2.

Note: To set the value of FLASH_CR.WAIT, you need to first write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence and then assign the value to FLASH_CR.WAIT. For details, see the FLASH controller section.

3.2.1 Standard Clock Switching Process

The operation process is as follows:

Step 1: If the new clock source requires an external pin, set the pin to the appropriate mode.

Note: When connecting to an external crystal oscillator, an analog pin is required; when connecting to an external clock input, a GPIO input and enabling the external clock input are required.

Step 2: Configure the oscillation parameters of the new clock source.

Step3: Enable the oscillator of the new clock source.

Step4: According to the higher frequency of the current clock source and the new clock source, configure FLASH_CR. WAIT according to the process of the Flash Controller section.

Step5: Wait for the new clock source to output a stable frequency.

Step6: Configure SYSCTRL0. CLKSW and select the source of the system clock as the new clock source.

Step7: According to the frequency of the new clock source, configure FLASH_CR. WAIT according to the process of the Flash Controller section.

Step8: Turn off the clock source that is no longer used.

3.2.2 RCH switching process between different oscillation frequencies

There are two solutions for switching between different oscillation frequencies of RCH.

Solution 1:

Step 1: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step 2: Set SYSCTRL0.HCLK_PRS to 0x7.

Step 3: Adjust the output frequency of RCH step by step up or down, 4M -> 8M -> 16M -> 24M/22.12M or 24M/22.12M ->16M -> 8M -> 4M.

Step 4: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step 5: Set SYSCTRL0.HCLK_PRS to 0x0.

The sample code for switching from 4M to 24M is as follows:

```
M0P_SystemCtrl->SYSCTRL2 = 0X5A5A;  
M0P_SystemCtrl->SYSCTRL2 = 0XA5A5;  
M0P_SystemCtrl->SYSCTRL0_f.HCLK_PRS = 7;  
M0P_SystemCtrl->RCH_CR = *((uint16 *) ( 0X00100C08 )); //4M  
M0P_SystemCtrl->RCH_CR = *((uint16 *) ( 0X00100C06 )); //8M  
M0P_SystemCtrl->RCH_CR = *((uint16 *) ( 0X00100C04 )); //16M  
M0P_SystemCtrl->RCH_CR = *((uint16 *) ( 0X00100C00 )); //24M  
M0P_SystemCtrl->SYSCTRL2 = 0X5A5A;  
M0P_SystemCtrl->SYSCTRL2 = 0XA5A5;  
M0P_SystemCtrl->SYSCTRL0_f.HCLK_PRS = 0
```

Solution 2:

Step 1: Switch the system clock to RCL, see 3.2.5 Switching from other clocks to RCL example.

Step 2: Switch the system clock to RCH, see 3.2.6 Switching from other clocks to RCH example.

3.2.3 Switching from other clocks to XTL example

The operation flow is as follows:

- Step1: Set PCADS. 14 and PCADS. 15 to 1, and configure the PC14/PC15 pins as analog ports.
- Step2: Configure XTL_CR[5:0] according to the crystal oscillator characteristics.
- Step3: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.
- Step4: Set SYSCTRL0. XTL_EN to 1 to enable the crystal oscillator circuit.
- Step5: Query and wait for the XTL_CR. Stable flag to become 1, and the crystal oscillator outputs a stable clock.
- Step6: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.
- Step7: Set SYSCTRL0. CLKSW to 3 to switch the system clock to XTL.
- Step8: Set FLASH_CR. WAIT to 0 according to the Flash Controller section flow.
- Step 9: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.
- Step 10: Set SYSCTRL0.xxx_EN to 0 to turn off the original clock.

3.2.4 Example of switching from other clocks to XTH

- Step1: Set PFADS. 0 and PFADS. 1 to 1, and configure the PF00/PF01 pins as analog ports.
- Step2: Configure XTH_CR[3:0] according to the crystal characteristics.
- Step3: Set XTH_CR. Startup to 3 and select the longest crystal oscillator stabilization time.
- Step4: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.
- Step5: Set SYSCTRL0. XTH_EN to 1 to enable the crystal oscillator circuit.
- Step6: According to the higher frequency of the current clock and XTH, configure FLASH_CR.WAIT according to the Flash controller chapter process.
- Step7: After querying and waiting for the XTH_CR. Stable flag to become 1, the software delays for more than 10ms, and the crystal oscillator outputs a stable clock.
- Step8: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.
- Step9: Set SYSCTRL0.CLKSW to 1 and switch the system clock to XTH.
- Step10: According to the frequency of XTH, configure FLASH_CR.WAIT according to the process in the Flash controller section.
- Step11: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.
- Step12: Set SYSCTRL0.xxx_EN to 0 to turn off the original clock.

3.2.5 Example of switching from other clocks to RCL

The operation flow is as follows:

Step1: Configure RCL_CR.TRIM and RCL_CR. Startup.

Step2: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step3: Set SYSCTRL0. RCL_EN to 1 to enable the RCL oscillation circuit.

Step4: Query and wait for the RCL_CR. Stable flag to become 1, and RCL outputs a stable clock.

Step5: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step6: Set SYSCTRL0. CLKSW to 2 to switch the system clock to RCL.

Step7: Set FLASH_CR. WAIT to 0 according to the Flash Controller section flow.

Step8: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step9: Set SYSCTRL0.xxx_EN to 0 to turn off the original clock.

3.2.6 Example of switching from other clocks to RCH

The operation flow is as follows:

Step1: Configure RCH_CR.TRIM.

Step2: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step3: Set SYSCTRL0. RCH_EN to 1 to enable the RCH oscillation circuit.

Step4: Query and wait for the RCH_CR. Stable flag to become 1, and RCH outputs a stable clock.

Step5: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step6: Set SYSCTRL0. CLKSW to 0 to switch the system clock to RCH.

Step7: Set FLASH_CR. WAIT to 0 according to the Flash Controller section process.

Step8: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step9: Set SYSCTRL0.xxx_EN to 0 to turn off the original clock.

3.2.7 Example of switching between PLL and RCH, the reference clock is RCH

The operation flow of switching from RCH to PLL is as follows:

Step 1: Set BGR_CR to 0x01 and delay 20us, waiting for BGR to start and stabilize.

Step 2: Set PLL_CR.REFSEL to 3 and select PLL clock source as RCH.

Step3: Configure PLL_CR.FRSEL according to the RCH frequency.

Step4: Configure PLL_CR.DIVN and PLL_CR.FOSC according to the PLL output frequency.

Step5: Set PLL_CR. Startup to 7 and select the longest PLL stabilization time.

Step6: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step7: Set SYSCTRL0.PLL_EN to 1 to enable the PLL oscillation circuit.

Step8: Configure FLASH_CR. WAIT according to the Flash controller chapter flow according to the PLL output frequency.

Step9: Query and wait for the PLL_CR. Stable flag to become 1, and the PLL outputs a stable clock.

Step10: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step11: Set SYSCTRL0.CLKSW to 4 to switch the system clock to PLL.

Warning: RCH_CR.TRIM cannot be changed during the switching process.

The operation flow of switching from PLL to RCH is as follows:

Step1: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step2: Set SYSCTRL0. CLKSW to 0 to switch the system clock to RCH.

Step3: Set FLASH_CR. WAIT to 0 according to the process in the Flash Controller section.

Step4: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step5: Set SYSCTRL0.PLL_EN to 0 to turn off PLL.

Step6: Set BGR_CR to 0x01 to 0 to turn off BGR.

Warning: RCH_CR.TRIM cannot be changed during the switching process.

3.2.8 Example of switching between PLL and XTH, the reference clock is XTH

The operation flow of switching from RCH to PLL is as follows:

Step1: Set BGR_CR to 0x01 and delay 20us, waiting for BGR to start and stabilize.

Step2: Set PLL_CR.REFSEL to 0 and select PLL clock source as XTH.

Step3: Configure PLL_CR.FRSEL according to XTH frequency.

Step4: Configure PLL_CR.DIVN and PLL_CR.FOSC according to PLL output frequency.

Step5: Set PLL_CR. Startup to 7 and select the longest PLL stabilization time.

Step6: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step7: Set SYSCTRL0.PLL_EN to 1 to enable PLL oscillation circuit.

Step8: Configure FLASH_CR. WAIT according to the process of Flash controller chapter according to PLL output frequency.

Step 9: Query and wait for the PLL_CR. Stable flag to become 1, and the PLL outputs a stable clock.

Step 10: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step 11: Set SYSCTRL0. CLKSW to 4 to switch the system clock to PLL.

The operation flow of switching from PLL to XTH is as follows:

Step1: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step2: Set SYSCTRL0. CLKSW to 1 to switch the system clock to XTH.

Step3: According to the frequency of XTH, configure FLASH_CR. WAIT according to the process of the Flash controller section.

Step4: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewrite.

Step5: Set SYSCTRL0.PLL_EN to 0 to turn off PLL.

Step6: Set BGR_CR to 0x01 to 0 to turn off BGR.

The following figure is the clock switching timing diagram:

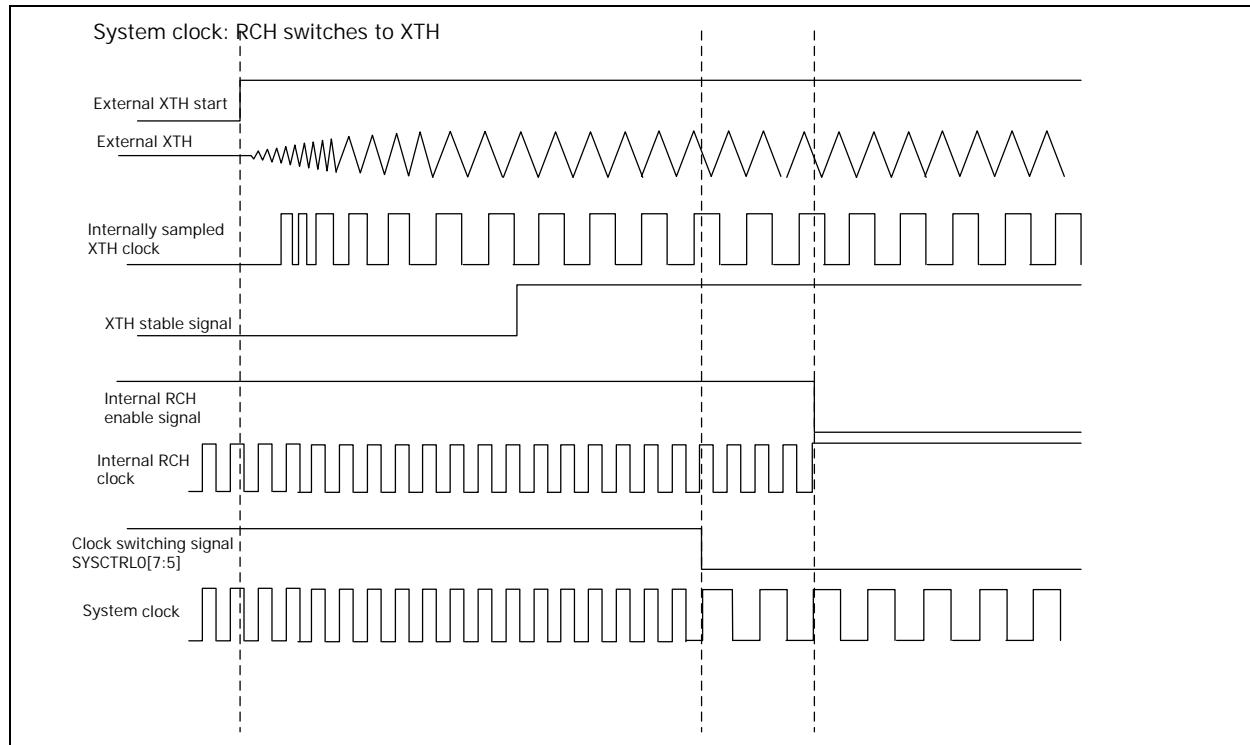


Figure 3-3 Clock switching diagram

3.3 Clock Calibration Module

This product has a built-in clock calibration circuit, as shown in the figure below. The five sources of the system clock can be calibrated with each other. After selecting the reference clock and the calibrated clock, set the register REFCNT value and set cali.start to start the clock calibration circuit. At this time, the two 32-bit counters (incrementing and decrementing) work simultaneously. When the decrement counter is equal to 0, cali.finish is set, indicating that the calibration is completed. At this time, the software can read the CALCNT value, so it is easy to get the frequency relationship between the reference clock and the calibrated clock.

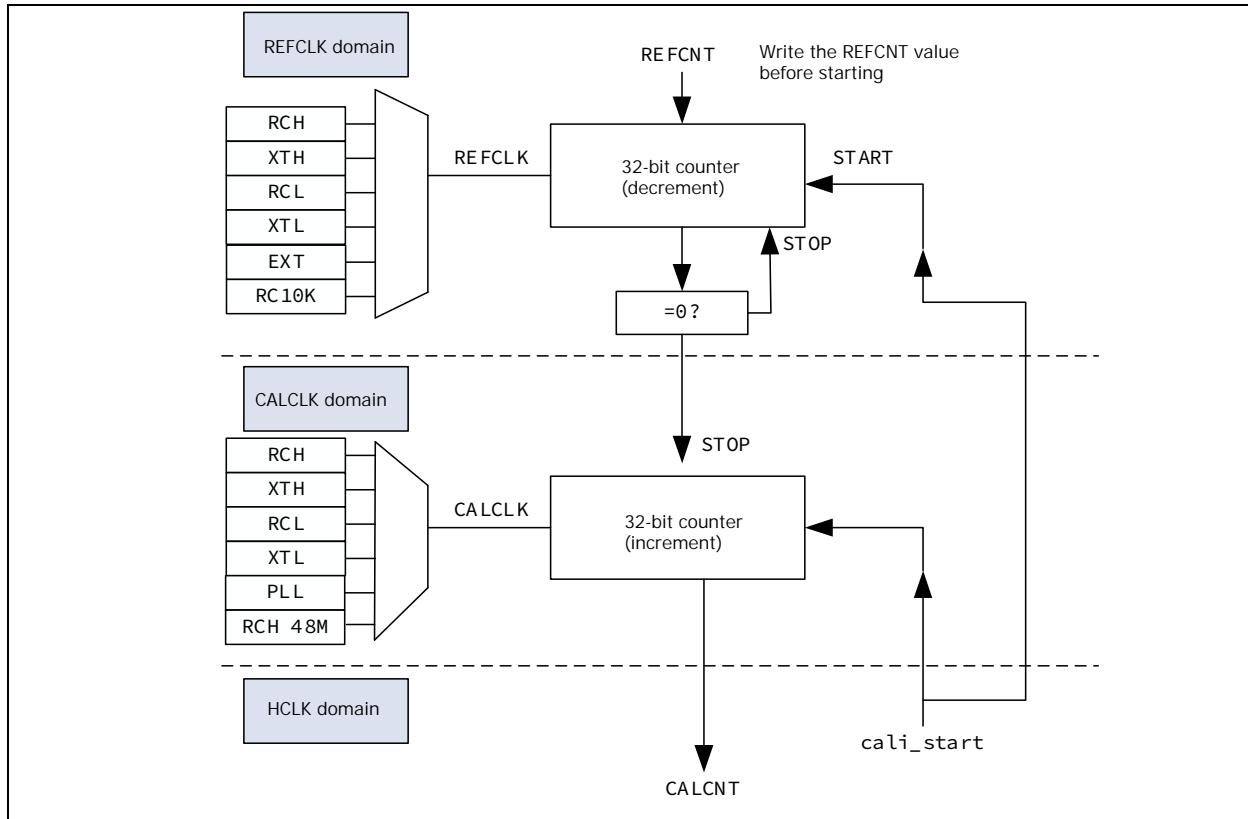


Figure 3-4 Clock calibration schematic

3.4 Interrupt wake-up control

When the processor executes the WFI instruction to enter the sleep state, it will stop executing instructions. When an interrupt request (higher priority) occurs in the sleep state and needs to be processed, the processor will be woken up.

The behavior of the processor in the sleep state when receiving an interrupt request is shown in the following table:

PRIMASK Status	WFI Behavior	Wake up	ISR Execution
0	IRQ priority > current level	Y	Y
0	IRQ priority = current level	N	N
1	IRQ priority > current level	Y	N
1	IRQ priority = current level	N	N

3.4.1 Waking up from deep sleep mode with interrupt

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set SCB->SCR.SLEEPDEEP to 1
4. Execute the WFI instruction to enter deep sleep mode
5. The system enters deep sleep mode and waits for interrupt wake-up. After wake-up, execute the interrupt service routine

Routine:

```
SCB->SCR |= 0x00000004u;  
while(1)  
{  
    __asm("WFI");  
}
```

3.4.2 Waking up from deep sleep mode without interrupt

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set PRIMASK to 1
4. Set SCB->SCR.SLEEPDEEP to 1
5. Execute the WFI instruction to enter deep sleep mode
6. The system enters deep sleep mode and waits for interrupt wake-up. After wake-up, execute the next instruction
7. Clear the interrupt flag and clear the interrupt pending state
8. Perform user-defined operations

Routine:

```
__asm("CPSID I"); //Set PRIMASK  
SCB->SCR |= 0x00000004u;  
while(1)  
{  
    __asm("WFI");  
    M0P_TIMx->ICLR = 0x00;           //Clear Int Flag  
    NVIC_ClearPendingIRQ(TIMERLP IRQn); //Clear Pending Flag  
    ... //Execute user-defined actions  
}
```

3.4.3 Using the Exit Hibernation Feature

Sleep-on-exit is very suitable for interrupt-driven applications. When this feature is enabled, the processor will enter sleep mode as soon as the exception handling is completed and the thread mode is returned. The sleep-on-exit feature allows the processor to stay in sleep mode as much as possible.

The Cortex-M0 uses the sleep-on-exit feature to enter sleep mode, which is similar to executing WFI immediately after executing an exception exit. However, in order to avoid the stack push operation the next time the exception is entered, the processor will not perform the pop process.

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set SCB->SCR.SLEEPDEEP to 1
4. Set SCB->SCR.SLEEPONEXIT to 1
5. Execute the WFI instruction to enter deep sleep mode
6. The system enters deep sleep mode and waits for interrupt wake-up. After wake-up, the interrupt service routine is executed
7. Automatically enter sleep mode when exiting interrupt service

Routine:

```
SCB_SCR |= 0x00000004u;  
SCB_SCR |= 0x00000002u;  
while(1)  
{  
    __asm("WFI");  
}
```

3.5 Registers

Base address 0x40002000

Table 3-1 System Control Registers

Register	Address offset	Description
SYSCTRL0	0x000	System Control Register 0
SYSCTRL1	0x004	System Control Register 1
SYSCTRL2	0x008	System Control Register 2
RCH_CR	0x00C	RCH Control Register
XTH_CR	0x010	XTH Control Register
RCL_CR	0x014	RCL Control Register
XTL_CR	0x018	XTL Control Register
PLL_CR	0x03C	PLL Control Register
PERI_CLKEN0	0x020	Peripheral Module Clock Control Register
PERI_CLKEN1	0x024	Peripheral Module Clock Control Register
OVCK_CR	0x054	Timer Clock Multiplication Control
RC48M_CR	0x058	RC48M Control Register

3.5.1 System Control Register 0 (SYSCTRL0)

Offset address: 0x000

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Wakeup ByRCH	Reserved	PclkPrs	HclkPrs		CLKSW		PLLE N	XTL EN	RCL EN	XTH EN	RCH EN				
RW		RW	RW		RW		RW	RW	RW	RW	RW				

Bit	Mark	Functional Description
31:16	Reserved	Reserved
15	Wakeup ByRCH	1: After waking up from Deep Sleep, the system clock source is RCH, and the original clock continues to be enabled. 0: After waking up from Deep Sleep, the system clock source does not change.
14:13	Reserved	Reserved
12:11	PclkPrs	PCLK clock source selection 00: HCLK 01: HCLK/2 10: HCLK/4 11: HCLK/8
10:8	HclkPrs	HCLK clock source selection 000: SystemClk 001: SystemClk/2 010: SystemClk/4 011: SystemClk/8 100: SystemClk/16 101: SystemClk/32 110: SystemClk/64 111: SystemClk/128
7:5	CLKSW	SystemClk clock source selection 000: Internal high-speed clock RCH 001: External high-speed crystal oscillator XTH 010: Internal low-speed clock RCL 011: External low-speed crystal oscillator XTL 100: Internal PLL
4	PLLEN	PLL enable control 0: Disable 1: Enable Note: PLL can only be enabled after BGR is enabled and stable.
3	XTLEN	XTLEN External low-speed crystal oscillator XTL enable control

		<p>0: Disable 1: Enable</p> <p>Note: PC14 and PC15 need to be set as analog ports.</p>
2	RCLEN	<p>Internal low-speed clock RCL enable control</p> <p>0: Disable 1: Enable</p>
1	XTHEN	<p>External high-speed crystal oscillator XTH enable control</p> <p>0: Disable 1: Enable</p> <p>Note: When the system enters DeepSleep, this high-speed clock will be automatically turned off</p>
0	RCHEN	<p>Internal high-speed clock RCH enable signal.</p> <p>0: Disable 1: Enable</p> <p>Note: When the system enters DeepSleep, this high-speed clock will be automatically turned off</p>

Note:

- Each time you rewrite the values of SYSCTRL0 and SYSCTRL1, you need to first write 0x5A5 and 0xA5A5 to SYSCTRL2 in sequence. Such steps can effectively prevent misoperation of the SYSCTRL0 and SYSCTRL1 registers.

3.5.2 System Control Register 1 (SYSCTRL1)

Offset address: 0x004

Reset value: 0x00000008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserve d	USB48MS EL	Re s	RTC_FREQ_ADJUST	SWD US E IO	LOC KUP _EN	RTC LPW	Res.	XTL_AL_WAYS_ ON	EXT L_E_N	EXT H_E_N	Res.	RW	RW	RW	Re s.
	RW		RW	RW		RW									

Bit	Mark	Functional Description
31:14	Reserved	Reserved
13	USB48MSEL	USB clock selection 0: RCH48M 1: PLL multiplier clock
12	Res.	Reserved
11:9	RTC_FREQ_ADJUST	RTC high-speed clock compensation clock frequency selection 000 4M; 001 6M; 010 8M; 011 12M 100 16M; 101 20M; 110 24M; 111 32M;
8	SWD USE IO	SWD port function configuration 0: SWD port 1: GPIO port
7	Res.	Reserved
6	LOCKUP_EN	Cortex-M0+ LockUp function configuration 0: Disable 1: Enable Note: After enabling, the CPU will reset the MCU when reading an invalid instruction, which can enhance system reliability.
5	RTC_LPW	RTC module low power control 1: Low power mode enabled 0: Low power mode disabled Note: After enabling, the RTC module enters low power mode and its registers cannot be read or written.
4	Res.	Reserved
3	XTL_ALWAYS_ON	XTL Advanced Enable Control 1: SYSCTRL0.XTL_EN can only be set. 0: SYSCTRL0.XTL_EN can be set or cleared.
2	EXTL_EN	External XTL clock input control 1: XTL output clock is input from PC14. 0: XTL output clock is generated by crystal oscillator. Note: When using PC14 input clock, SYSCTRL0.XTL_EN needs to be set to 1; when using PC14 external input clock, PC15 pin cannot be used as GPIO, and PC15 port needs to be configured as analog state.

1	EXTH_EN	External XTH input control 1: XTH output clock is input from PF00. 0: XTH output clock is generated by crystal oscillator. Note: When using PF00 input clock, SYSCTRL0.XTH_EN needs to be set to 1; when using PF00 external input clock, PF01 pin cannot be used as GPIO, and PF01 port needs to be configured as analog state.
0	Reserved	Reserved

Note:

- Each time you rewrite the values of SYSCTRL0 and SYSCTRL1, you need to first write 0x5A5 and 0xA5A5 to SYSCTRL2 in sequence. Such steps can effectively prevent misoperation of the SYSCTRL0 and SYSCTRL1 registers.

3.5.3 System Control Register 2 (SYSCTRL2)

Offset address: 0x0008

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYSCTRL2 WO															

Bit	Mark	Functional Description
31:16	Reserved	Reserved
15:0	SYSCTRL2	Registers SYSCTL0 and SYSCTRL1 protect the series control registers. For SYSCTRL2, first write 0x5A5A and then write 0xA5A5 to start the write operation for registers SYSCTL0 and SYSCTRL1. As long as the write operation is performed on registers SYSCTL0 and SYSCTRL1, this protection bit automatically returns to the protection state and needs to be rewritten to open the series protection.

3.5.4 RCH Control Register (RCH_CR)

Offset address: 0x00C

Reset value: 0x00000126

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		Stable		TRIM											
		RO		RW											

Bit	Mark	Functional Description
31:12	Reserved	Reserved
11	stable	RCH clock stable flag. 1: Indicates that RCH is stable and can be used by internal circuits. 0: Indicates that RCH is not stable and cannot be used by internal circuits.
10:0	TRIM	Clock frequency adjustment, change the value of this register to adjust the output frequency of RCH. Every time the register value increases by 1, the output frequency of RCH increases by about 0.2%, and the total adjustment range is 4~24MHz. Five sets of frequency calibration values have been saved in Flash. Read the calibration values in Flash and write them to RCH_CR.TRIM to obtain accurate frequency. 24M calibration value address: 0x00100C00 - 0x00100C01 22.12M calibration value address: 0x00100C02 - 0x00100C03 16M calibration value address: 0x00100C04 - 0x00100C05 8M calibration value address: 0x00100C06 - 0x00100C07 4M calibration value address: 0x00100C08 - 0x00100C09

3.5.5 XTH Control Register (XTH_CR)

Offset address: 0x010

Reset value: 0x00000022

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Stable	Startup	xth_fsel	Driver				
								RO	RW	RW	RW				

Bit	Mark	Functional Description
31:7	Reserved	Reserved
6	stable	<p>External high-speed clock XTH stable flag. 1: Indicates that XTH is stable and can be used by internal circuits. 0: Indicates that XTH is not stable and cannot be used by internal circuits.</p> <p>Note: In order to increase system reliability, after querying this flag, the software needs to delay for more than 10ms before switching the system clock to XTH.</p>
5:4	Startup	<p>External high-speed clock XTH stabilization time selection 00: 256 cycles; 01: 1024 cycles; 10: 4096 cycles; 11: 16384 cycles</p> <p>Note: It is strongly recommended to set the XTH stabilization time to 11. If the XTH stabilization time is insufficient, the system cannot work stably when switching clocks or waking up from deep sleep.</p>
3:2	xth_fsel	<p>External crystal oscillator operating frequency selection 11: 24M~32M 10: 16M~24M 01: 8M~16M 00: 4M~8M</p>
1:0	Driver	<p>External crystal oscillator drive capability selection 11: Strongest drive capability 10: Default drive capability (recommended value) 01: Weak drive capability 00: Weakest drive capability</p> <p>Note: It is necessary to select the appropriate drive capability based on the crystal oscillator characteristics, load capacitance, and parasitic parameters of the circuit board. The greater the drive capability, the greater the power consumption; the weaker the drive capability, the lower the power consumption.</p>

3.5.6 RCL Control Register (RCL_CR)

Offset address: 0x014

Reset value: 0x00000033F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			Stable	Startup		TRIM									
			RO		RW	RW									

Bit	Mark	Functional Description
31:13	Reserved	Reserved
12	stable	Internal low-speed clock RCL stable flag. 1: Indicates that RCL is stable and can be used by internal circuits. 0: Indicates that RCL is not stable and cannot be used by internal circuits.
11:10	Startup	Internal low-speed clock RCL stable time selection 11: 256 cycles; 10: 64 cycles; 01: 16 cycles; 00: 4 cycles
9:0	TRIM	Internal low-speed clock frequency adjustment, 2 sets of frequency calibration values are stored in Flash. Read the calibration value in Flash and write it to RCL_CR.TRIM to obtain accurate frequency. 38.4K calibration value address: 0x00100C20 - 0x00100C21 32.768K calibration value address: 0x00100C22 - 0x00100C23

3.5.7 XTL Control Register (XTL_CR)

Offset address: 0x018

Reset value: 0x000000021

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Stab le	Startup	Amp_sel	Driver				
								RO	RW	RW	RW				

Bit	Mark	Functional Description
31:7	Reserved	
6	stable	External low-speed crystal oscillator XTL stability flag. 1: Indicates that XTL is stable and can be used by the internal circuit. 0: Indicates that XTL is not stable and cannot be used by the internal circuit.
5:4	Startup	External low-speed crystal oscillator XTL stabilization time selection 00: 256 cycles; 01: 1024 cycles; 10: 4096 cycles; 11: 16384 cycles
3:2	amp_sel	Adjustment of XTL crystal oscillator oscillation amplitude. 11: Maximum amplitude 10: Large amplitude (recommended value) 01: Normal amplitude 00: Minimum amplitude
1:0	Driver	XTL crystal oscillator drive capability selection 11: Strongest drive capability 10: Stronger drive capability 01: General drive capability (recommended value) 00: Weakest drive capability Note: It is necessary to select the appropriate drive capability based on the crystal oscillator characteristics, load capacitance and parasitic parameters of the circuit board. The greater the drive capability, the greater the power consumption; the weaker the drive capability, the lower the power consumption.

3.5.8 PLL Control Register (PLL_CR)

Offset address: 0x03C

Reset value: 0x0000010B0F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														Stable	Startup
														RO	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Stableo pt	FRSEL	LFSEL	IBSEL	DVIN	FOSC	REFSEL									
RW	RW	RW	RW	RW	RW	RW									

Bit	Mark	Functional Description
31:19	Reserved	Reserved
18	Stable	PLL stability flag 0: PLL is not stable 1: PLL is stable
17:15	Startup	PLL stabilization time selection 000: 128 PLL cycles 001: 256 PLL cycles 010: 512 PLL cycles 011: 1024 PLL cycles 100: 2048 PLL cycles 101: 4096 PLL cycles 110: 8192 PLL cycles 111: 16384 PLL cycles
14:13	FRSEL	PLL input frequency selection, configure as follows according to the frequency of the PLL input clock 00: 4M~6M 01: 6M~12M 10: 12M~20M 11: 20M~24M
12:11	LFSEL	PLL filter control bits, please keep the default value
10:9	IBSEL	PLL bias current selection, please keep the default value
8:5	DIVN	PLL output clock multiplier PLL output frequency is a multiple of the input frequency. Output frequency = DIVN*input frequency. The allowed range of DIVN is 0X2~0XC
4:2	FOSC	PLL output frequency range selection. Configure as follows according to the PLL output clock frequency: 000: 8M~12M 001: 12M~18M

		010: 18M~24M 011: 24M~36M 1xx: 36M~48M
1:0	REFSEL	Input clock selection x0: Clock generated by XTH crystal oscillator 01: XTH clock input from pin PF00. (See 3.1.4 for details) 11: RCH clock

3.5.9 Peripheral module clock control register (PERI_CLKENO)

Offset address: 0x020

Reset value: 0x80800000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLASH	HDI V	DMA	GPIO	AES	CRC	SWD	TICK	Res.	LCD	Trim	RTC	PCNT	TRNG	VC	ADC
RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDT	PCA	OPA	Res.	TIM3	ADV TIM	LP TIM0	BASE TIM	SPI1	SPI0	I2C1	I2C0	LPUART1	LPUART0	UART1	UART0
RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Mark	Functional Description
31	FLASH	FLASH controller module clock is enabled. After it is turned off, the FLASH configuration register cannot be written, but the program in FLASH can still run. 1: Enable; 0: Disable
30	HDIV	HDIV module clock enable. 1: Enable; 0: Disable
29	DMA	DMAC module clock enable. 1: Enable; 0: Disable
28	GPIO	GPIO module clock enable. 1: Enable; 0: Disable
27	AES	AES module clock enable. 1: Enable; 0: Disable
26	CRC	CRC module clock enable. 1: Enable; 0: Disable
25	SWD	SWD module clock enable. 1: Enable; 0: Disable
24	TICK	SysTick module clock enable. 1: Enable; 0: Disable
23	Res.	Reserved
22	LCD	LCD module clock enable. 1: Enable; 0: Disable
21	TRIM	CLKTRIM module clock enable. 1: Enable; 0: Disable
20	RTC	RTC module clock enable. 1: Enable; 0: Disable
19	PCNT	PCNT module clock enable. 1: Enable; 0: Disable
18	TRNG	TRNG module clock enable.

		1: Enable; 0: Disable
17	VC	VC, LVD modules clock enable. 1: Enable; 0: Disable
16	ADC	ADC, BGR modules clock enable. 1: Enable; 0: Disable
15	WDT	WDT module clock enable. 1: Enable; 0: Disable
14	PCA	PCA module clock enable. 1: Enable; 0: Disable
13	OPA	OPA module clock enable. 1: Enable; 0: Disable
12	Res.	Reserved
11	TIM3	TIM3 module clock enable. 1: Enable; 0: Disable
10	ADVTIM	TIM456 module clock enable. 1: Enable; 0: Disable
9	LPTIM0	LPTIM0 module clock enable. 1: Enable; 0: Disable
8	BASETIM	TIM012 module clock enable. 1: Enable; 0: Disable
7	SPI1	SPI1 module clock enable. 1: Enable; 0: Disable
6	SPI0	SPI0 module clock enable. 1: Enable; 0: Disable
5	I2C1	I2C1 module clock enable. 1: Enable; 0: Disable
4	I2C0	I2C0 module clock enable. 1: Enable; 0: Disable
3	LPUART1	LPUART1 1: Enable; 0: Disable
2	LPUART0	LPUART0 module clock enable. 1: Enable; 0: Disable
1	UART1	UART1 module clock enable. 1: Enable; 0: Disable
0	UART0	UART0 module clock enable. 1: Enable; 0: Disable

3.5.10 Peripheral module clock control register (PERI_CLKEN1)

Offset address: 0x024

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						UAR T3	UAR T2	RES	I2S1	I2S0	LPTI M1	DAC	CTS	CAN	USB
						R/W	R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Mark	Functional Description
31:10	Reserved	Reserved
9	UART3	UART3 module clock enable. 1: Enable; 0: Disable
8	UART2	UART2 module clock enable. 1: Enable; 0: Disable
7	Reserved	Reserved
6	I2S1	I2S1 module clock enable. 1: Enable; 0: Disable
5	I2S0	I2S0 module clock enable. 1: Enable; 0: Disable
4	LPTIM1	LPTIM1 module clock enable. 1: Enable; 0: Disable
3	DAC	DAC module clock enable. 1: Enable; 0: Disable
2	CTS	CTS module clock enable. 1: Enable; 0: Disable
1	CAN	CAN module clock enable. 1: Enable; 0: Disable
0	USB	USB module clock enable. 1: Enable; 0: Disable

3.5.11 Timer Clock Multiplication Control (OVCK_CR)

Offset address: 0x054

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															OVC K RW

Bit	Mark	Functional Description
31:1	Reserved	Reserved
0	OVCK	When the timer uses the PLL clock, the timer uses the clock selection control. 0: The timer uses the system clock with the same frequency; 1: The timer uses the system clock with 2 times the frequency. When the timer switches the clock (when changing OVCK), the Peri_clken of the corresponding timer needs to be turned off

3.5.12 RC48M Control Register (RC48M_CR)

Offset address: 0x058

Reset value: 0x0280

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		EN	Stab le	Res.	TRIM										
RW		R	R/W												

Bit	Mark	Functional Description
31:13	Reserved	Reserved
12	EN	RCH48M module enable. 1: Enable; 0: Disable
11	Stable	RCH48M clock stability flag 0: Stable; 1: Not stable
10	Reserved	Reserved
9:0	TRIM	RCH48M calibration value

4 Reset controller (RESET)

4.1 Reset Controller Introduction

This product has 7 reset signal sources, each of which can restart the CPU. Most registers will be reset to reset values, and the program will start executing from the reset vector.

- Power-on and power-off reset POR for digital area
- External Reset pin, low level is reset signal
- WDT reset
- PCA reset
- LVD low voltage reset
- Cortex-M0+ SYSRESETREQ software reset
- Cortex-M0+ LOCKUP hardware reset

Each reset source is indicated by a corresponding reset flag. The reset flags are set by hardware and need to be cleared by user software. When the chip is reset, if Reset_flag. POR15V or Reset_flag. POR5V is 1, it is a power-on reset. When the user program resets the register Reset_flag, the reset source can be determined by the relevant bits of Reset_flag at the next reset.

The following figure describes the reset sources of each area.

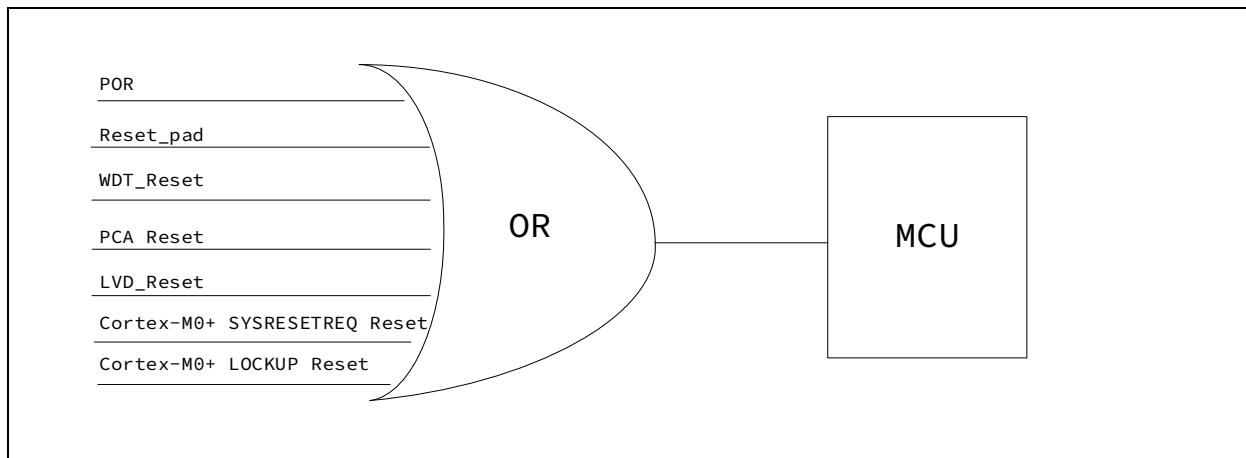


Figure 4-1 Reset sources diagram

4.1.1 Power-On Reset (POR)

This product has two power supply areas: VCC area and VCAP area. All analog modules and IO work in the VCC area; other modules work in the VCAP area.

When the VCC area is powered on, when the VCC voltage is lower than the POR threshold voltage (typical value is 1.65V), a POR5V signal will be generated; when the VCC area is powered off, when the VCC voltage is lower than the BOR threshold voltage (typical value is 1.5V), a POR5V signal will be generated.

When the VCAP area is powered on, when the VCAP voltage is lower than the POR threshold voltage, a POR15V signal will be generated; when the VCAP area is powered off, when the VCAP voltage is lower than the BOR threshold voltage, a POR15V signal will be generated.

Both the POR5V signal and the POR15V signal will reset the chip registers to the initialization state.

4.1.2 External reset pin reset

When the external reset pin detects a low level, a system reset will be generated. The reset pin has a built-in pull-up resistor and an integrated glitch filter circuit. The glitch filter circuit will filter out glitch signals less than 20us (typical value). Therefore, the low-level signal added to the reset pin must be greater than 20us to ensure reliable reset of the chip.

4.1.3 WDT reset

For watchdog reset, please refer to the description in the WDT chapter.

4.1.4 PCA Reset

For PCA reset, please refer to the description in the PCA chapter.

4.1.5 LVD Low Voltage Reset

For LVD reset, please refer to the description in the LVD chapter.

4.1.6 Cortex-M0+ SYSRESETREQ reset

Cortex-M0+ software reset.

4.1.7 Cortex-M0+ LOCKUP reset

When the Cortex-M0+ encounters a serious exception, it will stop its PC pointer at the current address, lock itself, and reset the entire CORE area after a few clock cycle delays.

4.2 Registers

4.2.1 Reset Flag Register (RESET_FLAG)

Address: 0x4000201C

Reset value: 00000000_00000000_00000000_xxxxxx11b

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RSTB	sysreq	lock up	PCA	WDT	LVD	Por15	Por5v
								RW0	RW0	RW0	RW0	RW0	RW0	RW0	RW0

Bit	Mark	Functional Description
31:8	Reserved	Reserved
7	RSTB	RESETB Port reset flag, requires software initialization and clearing, power-on state uncertain 1: Pin reset occurs 0: No pin reset occurs Write 0 to clear, write 1 is invalid
6	Sysreq	Cortex-M0+ CPU software reset flag, requires software initialization and clearing, power-on state uncertain 1: Cortex-M0+ CPU software reset occurs 0: No Cortex-M0+ CPU software reset occurs Write 0 to clear, write 1 is invalid
5	Lockup	Cortex-M0+ CPU Lockup reset flag, requires software initialization and clearing, power-on state uncertain 1: Cortex-M0+ CPU Lockup reset occurs 0: No Cortex-M0+ CPU Lockup reset occurs Write 0 to clear, write 1 is invalid
4	PCA	PCA reset flag, requires software initialization and clearing, power-on state uncertain 1: PCA reset occurs 0: No PCA reset occurs Write 0 to clear, write 1 is invalid
3	WDT	WDT reset flag, requires software initialization and clearing, power-on state uncertain 1: WDT reset occurs 0: No WDT reset occurs Write 0 to clear, write 1 is invalid
2	LVD	LVD reset flag, requires software initialization and clearing, power-on state is uncertain 1: LVD reset occurs 0: No LVD reset occurs Write 0 to clear, write 1 is invalid
1	POR15V	VCAP domain reset flag 1: VCAP domain reset occurs

		0: No reset occurred in VCAP domain Write 0 to clear, write 1 is invalid
0	POR5V	VCC power domain reset flag 1: VCC power domain reset occurred 0: No reset occurred in VCC power domain Write 0 to clear, write 1 is invalid

4.2.2 Peripheral module reset control register (PERI_RESET0)

Address: 0x40002028

Reset value: 0x7F7F6FFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	HDI V	DMA	GPIO	AES	CRC	SW D	TICK	Res.	LCD	Trim	RTC	PCN T	TRN G	VC	ADC	
	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	
Res	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PCA	OPA	Res.	TIM3	ADV TIM	LP TIM 0	BASE TIM	SPI1	SPI0	I2C1	I2C0	LPU ART 1	LPU ART 0	UAR T1	UAR T0	
	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

Bit	Mark	Functional Description
31	Res.	Reserved
30	HDIV	HDIV module reset enable. 1: normal operation; 0: reset module.
29	DMA	DMAC module reset enable. 1: normal operation; 0: reset module.
28	GPIO	GPIO module reset enable. 1: normal operation; 0: reset module.
27	AES	AES module reset enable. 1: normal operation; 0: reset module.
26	CRC	CRC module reset enable. 1: normal operation; 0: reset module.
25	SWD	SWD module reset enable. 1: normal operation; 0: reset module.
24	TICK	SYSTICK module reset enable. 1: normal operation; 0: reset module.
23	Res.	Reserved
22	LCD	LCD module reset enable. 1: normal operation; 0: reset module.
21	TRIM	CLKTRIM module reset enable. 1: 正常工作; 0: 模块处于复位状态
20	RTC	RTC module reset enable. 1: normal operation; 0: reset module.
19	PCNT	PCNT module reset enable. 1: normal operation; 0: reset module.
18	TRNG	TRNG module reset enable. 1: normal operation; 0: reset module.
17	VC	VC, LVD modules reset enable.

		1: normal operation; 0: reset module.
16	ADC	ADC module reset enable. 1: normal operation; 0: reset module.
15	Res.	Reserved
14	PCA	PCA module reset enable. 1: normal operation; 0: reset module.
13	OPA	OPA module reset enable. 1: normal operation; 0: reset module.
12	Res.	Reserved
11	TIM3	TIM3 module reset enable. 1: normal operation; 0: reset module.
10	ADVTIM	TIM456 module reset enable. 1: normal operation; 0: reset module.
9	LPTIMO	LPTIMO module reset enable. 1: normal operation; 0: reset module.
8	BASETIM	TIM012 module reset enable. 1: normal operation; 0: reset module.
7	SPI1	SPI1 module reset enable. 1: normal operation; 0: reset module.
6	SPI0	SPI0 module reset enable. 1: normal operation; 0: reset module.
5	I2C1	I2C1 module reset enable. 1: normal operation; 0: reset module.
4	I2C0	I2C0 module reset enable. 1: normal operation; 0: reset module.
3	LPUART1	LPUART1 module reset enable. 1: normal operation; 0: reset module.
2	LPUART0	LPUART0 module reset enable. 1: normal operation; 0: reset module.
1	UART1	UART1 module reset enable. 1: normal operation; 0: reset module.
0	UART0	UART0 module reset enable. 1: normal operation; 0: reset module.

4.2.3 Peripheral module reset control register (PERI_RESET1)

Address: 0x4000202C

Reset value: 0x000003FF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						UAR T3	UAR T2	RES.	I2S1	I2S0	LPTI M1	DAC	CTS	CAN	USB
						R/W	R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Mark	Functional Description
31:10	Reserved	Reserved
9	UART3	UART3 module reset enable. 1: normal operation; 0: reset module.
8	UART2	UART2 module reset enable. 1: normal operation; 0: reset module.
7	RES.	Reserved
6	I2S1	I2S1 module reset enable. 1: normal operation; 0: reset module.
5	I2S0	I2S0 module reset enable. 1: normal operation; 0: reset module.
4	LPTIM1	LPTIM1 module reset enable. 1: normal operation; 0: reset module.
3	DAC	DAC module reset enable. 1: normal operation; 0: reset module.
2	CTS	CTS module reset enable. 1: normal operation; 0: reset module.
1	CAN	CAN module reset enable. 1: normal operation; 0: reset module.
0	USB	USB module reset enable. 1: normal operation; 0: reset module.

5 Interrupt Controller (NVIC)

5.1 Overview

The Cortex-M0+ processor has a built-in nested vector interrupt controller (NVIC), which supports up to 32 interrupt request (IRQ) inputs and 1 non-maskable interrupt (NMI) input (not used in this product system). In addition, the processor also supports multiple internal exceptions.

Each exception source has a separate exception number, and each exception type has a corresponding priority. Some exceptions have fixed priorities, while others are programmable. The details are shown in the following table:

Table 5-1 Cortex-M0+ processor interrupt overview

Exception number	Exception type	Priority	Description
1	Reset	-3 (highest)	Reset
2	NMI	-2	Non-maskable interrupt (not used in this system)
3	Hardware error	-1	Error handling exception
4-10	Reserved	NA	...
11	SVC	Programmable	Calling the supervisor through the SVC instruction
12-13	Reserved	NA	...
14	PendSV	Programmable	System service maskable request
15	SysTick	Programmable	SysTick timer
16	Interrupt #0	Programmable	External interrupt #0
17	Interrupt #1	Programmable	External interrupt #1
...
47	Interrupt #31	Programmable	External interrupt #31

This chapter only introduces the 32 external interrupt requests (interrupt #0 to interrupt #31) of the processor in detail. For the specific situation of the internal exception of the processor, please refer to other related documents. At the same time, this chapter only discusses the interrupt handling mechanism of the NVIC in the processor core, and the interrupt generation mechanism of the peripheral module itself is not discussed here.

5.2 Interrupt Priority

Each external interrupt corresponds to a priority register. Each priority is 2 bits wide and uses the highest two bits of the interrupt priority register. Each register occupies 1 byte (8 bits). Under this setting, the available priorities are 0x00 (highest), 0x40, 0x80 and 0xc0 (lowest).

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
Used		Not used, read as 0						

Figure 5-1 Only the higher two bits of the priority register are used

If the processor is already running another interrupt handler, and the priority of the new interrupt is higher than the currently executing one, a preemption will occur. The currently running interrupt handler will be suspended and the new interrupt will be executed instead. This process is usually called interrupt nesting. After the new interrupt is executed, the previous interrupt handler will continue to execute and return to the program thread after it is completed.

If the processor is already running another interrupt handler with the same or higher priority, the new interrupt will wait and enter the pending state. The pending interrupt will wait until the current interrupt level changes, for example, the current priority level drops to a lower level than the pending interrupt after the currently running interrupt handler completes and returns.

If two interrupts occur at the same time and they have the same priority, the interrupt with the lower interrupt number will be executed first. For example, if interrupt #0 and interrupt #1 are enabled and have the same priority, when they are triggered at the same time, interrupt #0 will be executed first.

5.3 Interrupt Vector Table

When the Cortex-M0+ processor is processing an interrupt service request, it needs to first determine the starting address of the exception processing. The required information is called the vector table, as shown in Figure 5-2. The vector table is stored at the beginning of the memory space and contains the exception (interrupt) vectors of the available exceptions (interrupts) in the system, as well as the initial value of the main stack pointer (MSP).

Memory address	Exception number
0x0000004C	19
0x00000048	18
0x00000044	17
0x00000040	16
0x0000003C	15
0x00000038	14
0x00000034	13
0x00000030	12
0x0000002C	11
0x00000028	10
0x00000024	9
0x00000020	8
0x0000001C	7
0x00000018	6
0x00000014	5
0x00000010	4
0x0000000C	3
0x00000008	2
0x00000004	1
0x00000000	0

Figure 5-2 Interrupt vector table

The storage order of the interrupt vectors is consistent with the interrupt number. Since each vector is 1 word (4 bytes), the address of the interrupt vector is the interrupt number multiplied by 4. Each interrupt vector is the starting address of the interrupt processing.

5.4 Interrupt Input and Suspend Behavior

In the NVIC module of the Cortex-M0+ processor, each interrupt input corresponds to a pending status register, and each register has only 1 bit, which is used to save the interrupt request, regardless of whether the request is confirmed or not. When the processor starts to process the interrupt, the hardware will automatically clear the pending status bit.

The peripherals of this system use level-triggered interrupt outputs. When an interrupt event occurs, the interrupt signal will be confirmed because the peripheral is connected to the NVIC. The signal will remain high before the processor executes the interrupt service and clears the interrupt signal of the peripheral. Inside the NVIC, when an interrupt is detected, the pending status of the interrupt will be set. When the processor receives the interrupt and starts to execute the interrupt service routine, the pending status will be cleared. The process is shown in Figure 5-3:

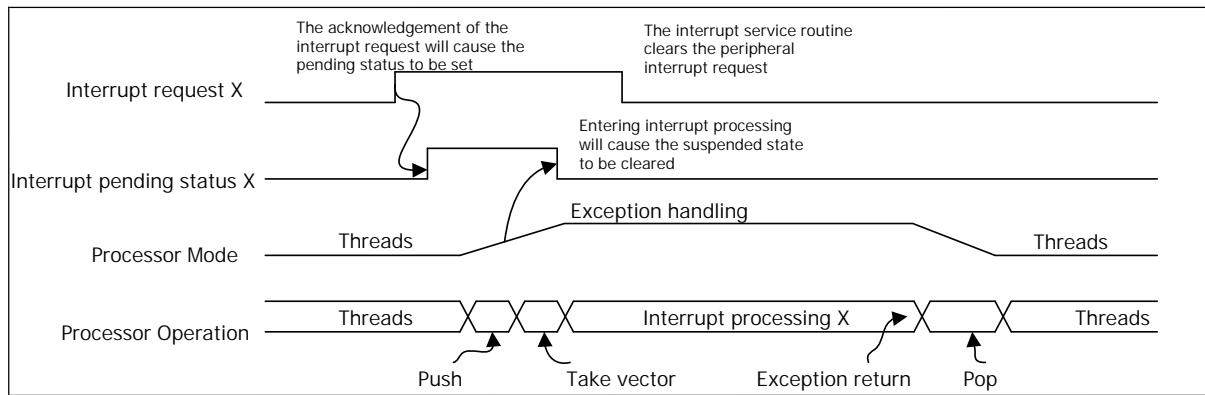


Figure 5-3 Interrupt activation and suspension status

If the interrupt request is not executed immediately and is cleared by software before confirmation, the processor will ignore the request and will not perform interrupt processing. The interrupt suspension status can be cleared by writing the NVIC_ICPR register. This process is very useful when setting up peripherals, because the peripheral may have generated an interrupt request before setting.

If the peripheral still holds the interrupt request when the software clears the suspension status, the suspension status will be generated immediately. The process is shown in Figure 5-4:

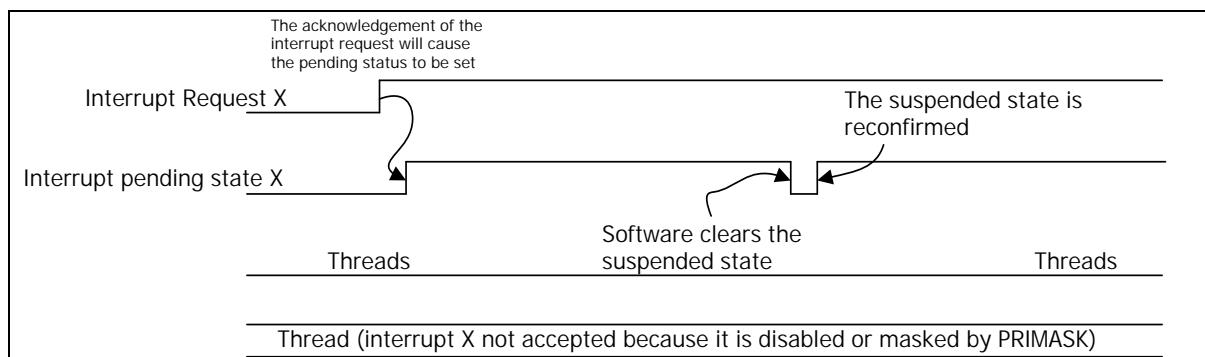


Figure 5-4 The interrupt pending state is cleared and then reasserted

If the interrupt request generated by the peripheral is not cleared during exception processing, the suspended state will be activated again after the exception is returned, so that the interrupt service routine will be executed again. The process is shown in Figure 5-5:

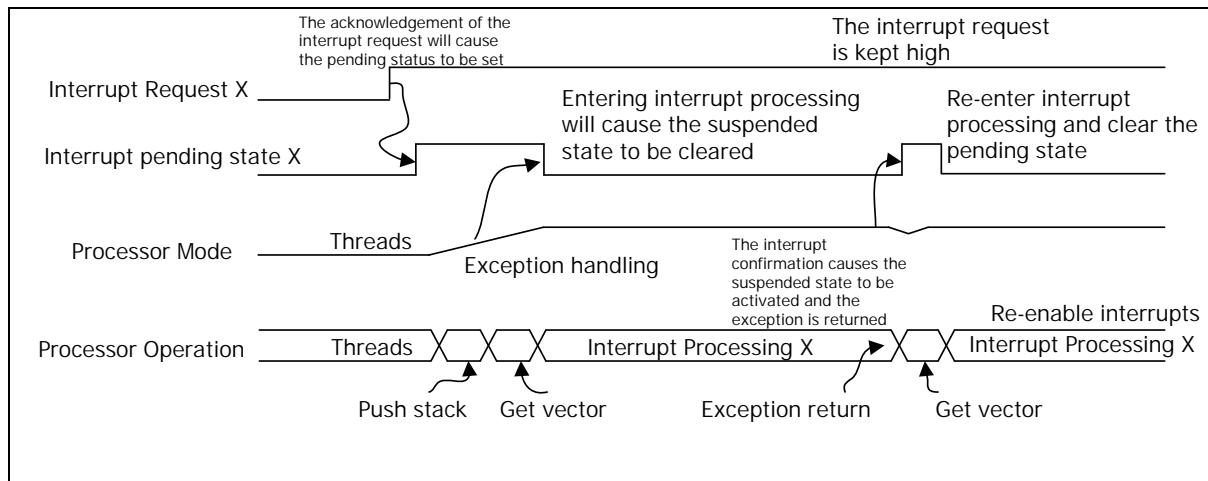


Figure 5-5 If the interrupt request remains high when the interrupt exits, the interrupt processing will be executed again

If a peripheral interrupt request is generated during the execution of the terminal service program, the request will be treated as a new interrupt request, and after the current interrupt exits, it will cause the interrupt service program to be executed again. This process is shown in Figure 5-6:

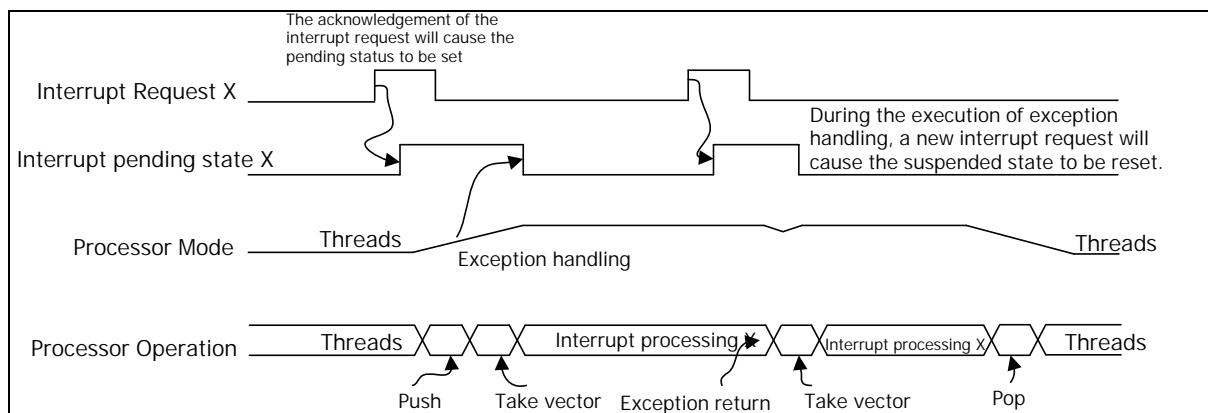


Figure 5-6 Interrupt pending generated during interrupt processing can also be confirmed

5.5 Interrupt Wait

Normally, the interrupt waiting time of NVIC is 16 cycles. This waiting time starts from the processor clock cycle when the interrupt is confirmed and ends when the interrupt processing starts. The following prerequisites are required to calculate the interrupt waiting time:

- The interrupt is enabled and is not masked by PRIMASK or other exception processing being executed.
- The memory system does not have any wait state. Bus transfer is used in interrupt processing, stack push, vector fetch or instruction fetch at the beginning of interrupt processing. If the memory system needs to wait, the wait state generated when the bus transfer occurs may cause the interrupt to be delayed. The following situations may cause different interrupt waiting times:
 - The end of the interrupt is chained. If another interrupt request is generated when the interrupt returns, the processor will skip the stack pop and stack push process, thus reducing the interrupt waiting time.
 - Delayed arrival. If another low-priority interrupt is being pushed when the interrupt occurs, due to the existence of the delayed arrival mechanism, the high-priority interrupt will be executed first, which will also reduce the waiting time of the high-priority interrupt.

5.6 Interrupt Sources

Because the NVIC of the Cortex-M0+ processor supports up to 32 external interrupts, and in this system, the number of external interrupt sources is greater than 32, so some external interrupts are reused on the same NVIC interrupt input, and NMI (non-maskable interrupt) is not used. The correspondence between all external interrupt sources and NVIC interrupt inputs in this system is shown in the following table:

Table 5-2 Correspondence between external interrupts and NVIC interrupt inputs

NVIC interrupt input	External interrupt source	Active mode	Sleep mode	DeepSleep mode
Interrupt #0	PORTA	✓	✓	✓
Interrupt #1	PORTB	✓	✓	✓
Interrupt #2	PORTC/PORTE	✓	✓	✓
Interrupt #3	PORTE/PORTE	✓	✓	✓
Interrupt #4	DMAC	✓	✓	-
Interrupt #5	TIM3	✓	✓	-
Interrupt #6	UART0/2	✓	✓	-
Interrupt #7	UART1/3	✓	✓	-
Interrupt #8	LPUART0	✓	✓	✓
Interrupt #9	LPUART1	✓	✓	✓
Interrupt #10	SPI0/I2S0	✓	✓	-
Interrupt #11	SPI1/I2S1	✓	✓	-
Interrupt #12	I2C0	✓	✓	-
Interrupt #13	I2C1	✓	✓	-
Interrupt #14	TIM0	✓	✓	-
Interrupt #15	TIM1	✓	✓	-
Interrupt #16	TIM2	✓	✓	-
Interrupt #17	LPTIM0/1	✓	✓	✓
Interrupt #18	TIM4	✓	✓	-
Interrupt #19	TIM5	✓	✓	-
Interrupt #20	TIM6	✓	✓	-
Interrupt #21	PCA	✓	✓	-
Interrupt #22	WDT	✓	✓	✓
Interrupt #23	RTC	✓	✓	✓
Interrupt #24	ADC/DAC	✓	✓	-
Interrupt #25	PCNT	✓	✓	✓
Interrupt #26	VC0/VC1/VC2/LVD	✓	✓	✓
Interrupt #27	USB	✓	✓	-
Interrupt #28	CAN	✓	✓	-
Interrupt #29	LCD	✓	✓	✓
Interrupt #30	FLASH/RAM	✓	✓	-

NVIC interrupt input	External interrupt source	Active mode	Sleep mode	DeepSleep mode
Interrupt #31	CLKTRIM/CTS	✓	✓	✓

Note:

- Since some module interrupts are reused in the same interrupt source, when entering the interrupt operation, you must first determine which module generated the interrupt, and then perform the corresponding interrupt operation.

5.7 Interrupt Structure Diagram

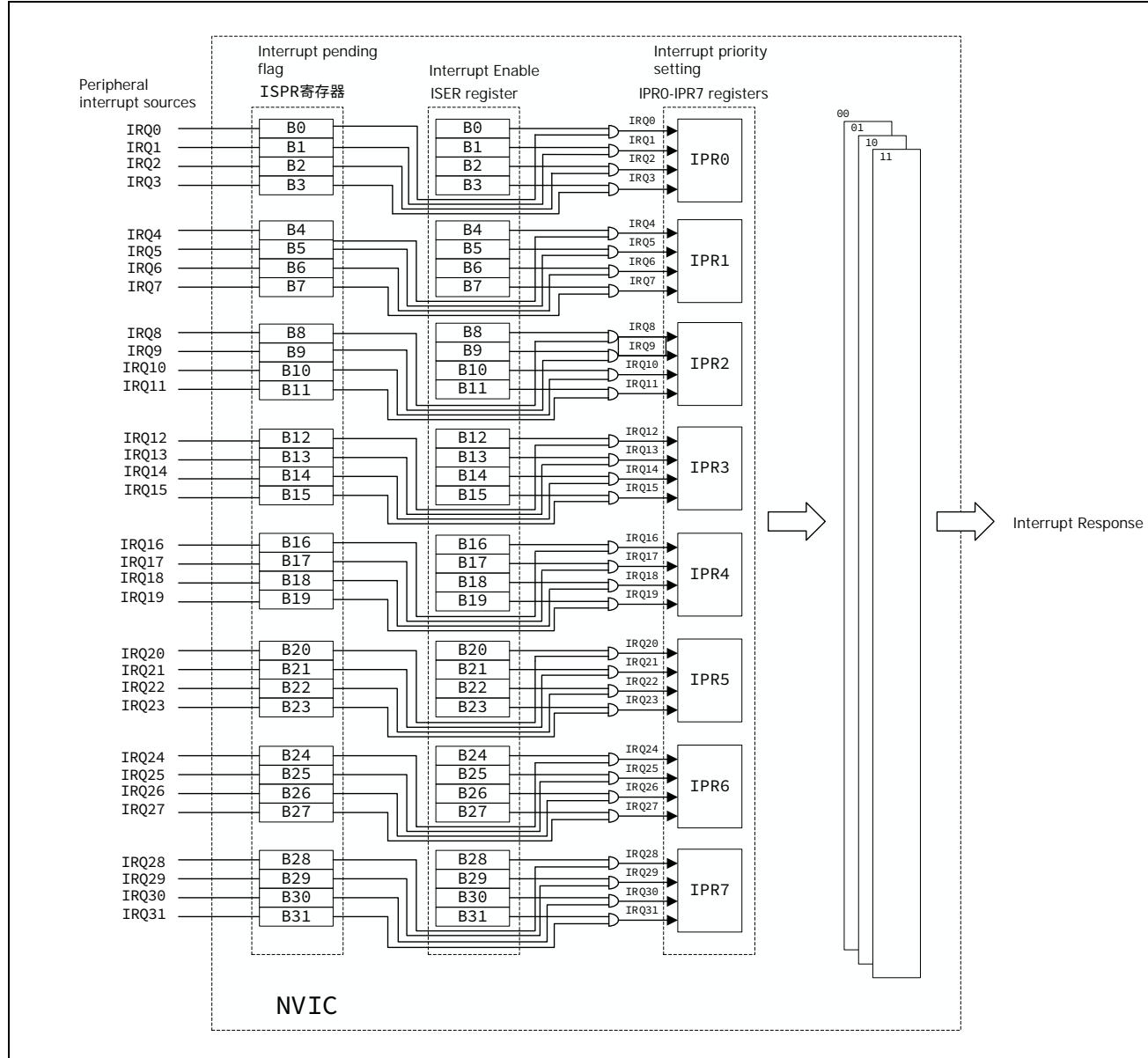


Figure 5-7 Interrupt structure diagram

The interrupt structure block diagram of this system is shown in Figure 5-7. Several points need to be noted:

- The interrupt enable of each peripheral interrupt source is not marked in the figure. Here only the interrupt signal logic block diagram after the peripheral interrupt is generated is included.
- IRQ has more than one peripheral interrupt input multiplexing, and the interrupt flag bits of these peripherals must be read separately to determine which peripheral interrupt is.
- If a high level is generated in the peripheral interrupt source, regardless of whether the NVIC interrupt enable register SCS_SETENA is set or not, the interrupt suspension register SCS_SEPEND will be set, indicating that the corresponding peripheral interrupt source has an interrupt.
- Only when the interrupt enable register NVIC_ISER is set, the corresponding interrupt IRQ will respond to the processor and execute the corresponding interrupt program.
- The high-level interrupt signal of the peripheral interrupt source must be cleared in the interrupt program, and the interrupt suspension register NVIC_ISPR is automatically cleared by the hardware.
- The interrupt priority registers NVIC_IPR0-NVIC_IPR7 set the priority of 32 interrupt sources, 00 is the highest priority, and 11 is the lowest priority. When the priority is the same, the priority is determined by the interrupt number, and the smaller the number, the higher the priority.

5.8 Registers

Base address: 0xE000 E000

Register	Offset Address	Description
NVIC_ISER	0x100	Interrupt Request Enable Register
NVIC_ICER	0x180	Interrupt Request Clear Enable Register
NVIC_ISPR	0x200	Interrupt Set Pending Register
NVIC_ICPR	0x280	Interrupt Clear Pending Register
NVIC_IPR0	0x400	Interrupt #0-Interrupt #3 Priority Register
NVIC_IPR1	0x404	Interrupt #4-Interrupt #7 Priority Register
NVIC_IPR2	0x408	Interrupt #8-Interrupt #11 Priority Register
NVIC_IPR3	0x40C	Interrupt #12-Interrupt #15 Priority Register
NVIC_IPR4	0x410	Interrupt #16-Interrupt #19 Priority Register
NVIC_IPR5	0x414	Interrupt #20-Interrupt #23 Priority Register
NVIC_IPR6	0x418	Interrupt #24-Interrupt #27 Priority Register
NVIC_IPR7	0x41C	Interrupt #28-Interrupt #31 Priority Register
PRIMASK	-	Interrupt Mask Special Register

5.8.1 Interrupt Enable Setting Register (NVIC_ISER)

Offset address: 0x100

Reset value: 0x0000 0000

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

Bit	Mark	Functional Description
31:0	SETENA [31:0]	Set to enable interrupt #0 to interrupt #31; write "1" to set, write "0" to have no effect [0]:IRQ0 [1]:IRQ1 [2]:IRQ2 [31]:IRQ31

5.8.2 Interrupt Enable Clear Register (NVIC_ICER)

Offset address: 0x180

Reset value: 0x0000 0000

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

Bit	Mark	Functional Description
31:0	CLRENA	Clear enable interrupt #0 to interrupt #31; write "1" to clear, write "0" to have no effect

5.8.3 Interrupt Pending Status Set Register (NVIC_ISPR)

Offset address: 0x200

Reset value: 0x0000 0000

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

Bit	Mark	Functional Description
31:0	SETPEND	Set the pending status of interrupt #0 to interrupt #31; write "1" to set, write "0" to have no effect [0]:IRQ0 [1]:IRQ1 [2]:IRQ2 [31]:IRQ31

5.8.4 Interrupt Pending Status Clear Register (NVIC_ICPR)

Offset address: 0x280

Reset value: 0x0000 0000

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

Bit	Mark	Functional Description
31:0	CLRPEND	Clear the pending status of interrupt #0 to interrupt #31; write "1" to clear, write "0" to have no effect [0]:IRQ0 [1]:IRQ1 [2]:IRQ2 [31]:IRQ31

5.8.5 Interrupt Priority Register (NVIC_IPR0)

Offset address: 0x400

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR0[31:16]								RW							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR0[15:0]]								RW							

Bit	Mark	Functional Description
31:0	IPR0[31:0]	The priority of interrupt #0 to interrupt #3 [31:30]: priority of interrupt #3 [23:22]: priority of interrupt #2 [15:14]: priority of interrupt #1 [7:6]: priority of interrupt #0 Among them, 00 is the highest priority and 11 is the lowest priority

5.8.6 Interrupt Priority Register (NVIC_IPR1)

Offset address: 0x404

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR1[31:16]								RW							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR1[15:0]]								RW							

Bit	Mark	Functional Description
31:0	IPR1[31:0]	Priority of interrupt #4 to interrupt #7 [31:30]: Priority of interrupt #7 [23:22]: Priority of interrupt #6 [15:14]: Priority of interrupt #5 [7:6]: Priority of interrupt #4 Among them, 00 is the highest priority and 11 is the lowest priority

5.8.7 Interrupt Priority Register (NVIC_IPR2)

Offset address: 0x408

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR2[31:16]															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR2[15:0]]															RW

Bit	Mark	Functional Description
31:0	IPR2[31:0]	Priority of interrupt #8 to interrupt #11 [31:30]: Priority of interrupt #11 [23:22]: Priority of interrupt #10 [15:14]: Priority of interrupt #9 [7:6]: Priority of interrupt #8 Among them, 00 is the highest priority and 11 is the lowest priority

5.8.8 Interrupt Priority Register (NVIC_IPR3)

Offset address: 0x40C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR3[31:16]															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR3[15:0]]															RW

Bit	Mark	Functional Description
31:0	IPR3[31:0]	Priority of interrupt #12 to interrupt #15 [31:30]: Priority of interrupt #15 [23:22]: Priority of interrupt #14 [15:14]: Priority of interrupt #13 [7:6]: Priority of interrupt #12 Among them, 00 is the highest priority and 11 is the lowest priority

5.8.9 Interrupt Priority Register (NVIC_IPR4)

Offset address: 0x410

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR4[31:16]															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR4[15:0]]															RW

Bit	Mark	Functional Description
31:0	IPR4[31:0]	Priority of interrupt #16 to interrupt #19 [31:30]: Priority of interrupt #19 [23:22]: Priority of interrupt #18 [15:14]: Priority of interrupt #17 [7:6]: Priority of interrupt #16 Among them, 00 is the highest priority and 11 is the lowest priority

5.8.10 Interrupt Priority Register (NVIC_IPR5)

Offset address: 0x414

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR5[31:16]															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR5[15:0]]															RW

Bit	Mark	Functional Description
31:0	IPR5[31:0]	Priority of interrupt #20 to interrupt #23 [31:30]: Priority of interrupt #23 [23:22]: Priority of interrupt #22 [15:14]: Priority of interrupt #21 [7:6]: Priority of interrupt #20 Among them, 00 is the highest priority and 11 is the lowest priority

5.8.11 Interrupt Priority Register (NVIC_IPR6)

Offset address: 0x418

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR6[31:16]															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR6[15:0]]															RW

Bit	Mark	Functional Description
31:0	IPR6[31:0]	Priority of interrupt #24 to interrupt #27 [31:30]: Priority of interrupt #27 [23:22]: Priority of interrupt #26 [15:14]: Priority of interrupt #25 [7:6]: Priority of interrupt #24 Among them, 00 is the highest priority and 11 is the lowest priority

5.8.12 Interrupt Priority Register (NVIC_IPR7)

Offset address: 0x41C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR7[31:16]															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR7[15:0]]															RW

Bit	Mark	Functional Description
31:0	IPR7[31:0]	Priority of interrupt #28 to interrupt #31 [31:30]: Priority of interrupt #31 [23:22]: Priority of interrupt #30 [15:14]: Priority of interrupt #29 [7:6]: Priority of interrupt #28 Among them, 00 is the highest priority and 11 is the lowest priority

5.8.13 Interrupt Mask Special Register (PRIMASK)

Offset address: ---

Reset value: 0x0000 0000

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

Bit	Mark	Functional Description
31:1	Reserved	
0	PRIMASK	When set, all interrupts except NMI and hardware error exceptions will be masked. When cleared, all exceptions and interrupts will not be masked. This special register needs to be accessed through the MSR and MRS special register operation instructions, and can also be accessed using the change processor state instruction CPS. When processing time-sensitive applications, it is necessary to operate the PRIMASK register.

5.9 Basic Software Operations

5.9.1 External interrupt enable

Each peripheral module has its own interrupt enable register. When an interrupt operation is required, the peripheral's own interrupt enable must be turned on first. The operation of this enable bit is not discussed in this chapter. Please refer to the description of each peripheral module.

5.9.2 NVIC interrupt enable and clear enable

The Cortex-M0+ processor supports up to 32 interrupt sources, each of which has an interrupt enable bit and a clear enable bit. In this way, there are 32-bit interrupt enable register NVIC_ISER and 32-bit clear enable register NVIC_ICER. If you want to enable an interrupt, set the corresponding position of the NVIC_ISER register to 1. If you want to clear an interrupt, set the corresponding position of the NVIC_ICER register to 1.

Note that the interrupt enable mentioned here is only for the processor NVIC. Whether each peripheral interrupt is generated or not is determined by the peripheral interrupt control register and has nothing to do with NVIC_ISER and NVIC_ICER.

5.9.3 NVIC interrupt pending and clearing pending

If an interrupt occurs but cannot be serviced immediately, the interrupt request will be pending. The pending state is stored in a register and remains valid until the processor's current priority level has been lowered to a level that allows the pending request to be serviced and the pending state is not cleared manually.

When the processor begins interrupt processing, the hardware automatically causes the pending state to be cleared.

The interrupt pending state can be accessed or modified by manipulating the interrupt set pending NVIC_ISPR and interrupt clear pending NVIC_ICPR registers. The interrupt pending state registers allow software to trigger interrupts.

5.9.4 NVIC Interrupt Priority

The NVIC_IPR0- NVIC_IPR7 registers determine the priority of SCS IRQ0- SCS IRQ32. Programming of the interrupt priority registers should be done before interrupts are enabled, which is usually done at the start of the program. Changing the interrupt priority after interrupts are enabled should be avoided as the results are unpredictable and are not supported by the Cortex-M0+ processor.

5.9.5 NVIC interrupt mask

Some time-sensitive applications need to disable all interrupts for a short period of time, which can be achieved by using the interrupt mask register SCS_PRIMASK. The special register SCS_PRIMASK has only 1 bit valid and defaults to 0 after reset. When this register is 0, all interrupts and exceptions are enabled; when it is set to 1, only NMI (not supported by this system) and hardware error exceptions are enabled. In fact, when SCS_PRIMASK is set to 1, the current priority of the processor is reduced to 0 (the highest priority that can be set).

The SCS_PRIMASK register can be programmed in a variety of ways. Using assembly language, the SCS_PRIMASK register can be set and cleared using the MS R instruction. If using C language and the CMSIS device driver library, users can use the following functions to set and clear PRIMASK.

```
void __enable_irq(void); //Clear PRIMASK
```

```
void __disable_irq(void); //Set PRIMASK
```

6 Port controller (GPIO)

6.1 Introduction to Port Controller

This series has 86 digital general-purpose input and output ports PA[15:0], PB[15:0], PC[15:0], PD[15:0], PE[15:0], PF[11:9], PF[7:0]. All GPIOs that are not packaged should also be configured as inputs and have pull-ups enabled.

The input and output signals of the analog modules ADC/VC/LVD/LCD, the input and output signals of various functional modules (such as SPI, UART, I2C, Timer, etc.) and the digital general-purpose input and output ports are multiplexed.

Each port can be configured as an internal pull-up/pull-down input, a high-impedance input (floating input), a push-pull output (CMOS output), an open drain output (open drain output), and a two-level drive capability output. After the digital port is configured as an analog port, the digital function is isolated and cannot output digital "1" and "0". The result of the CPU reading the port input value register is "0".

When each digital port is configured as an input, it can provide an external interrupt. The interrupt type can be configured as high-level trigger, low-level trigger, rising edge trigger, and falling edge trigger. The corresponding interrupt trigger port can be found by querying the interrupt flag bit of Px_STAT[n]. In addition, the interrupt of each digital port can wake up the chip from sleep mode/deep sleep mode to working mode.

After the chip is reset, the port is a high-impedance input (floating input) to prevent abnormal action on external devices when the chip is reset abnormally. However, in order to avoid leakage caused by high-impedance input, the user must configure the port accordingly after the chip is started (configured as internal pull-up/pull-down input or output).

6.2 Port Controller Main Features

The port controller supports the following features:

- Port input value/output value register supports FAST IO/AHB bus read and write
- Other registers support AHB bus interface read and write
- Analog function pin/digital general pin/digital function pin multiplexing
- Support pull-up/pull-down/two-speed drive/open-drain output function selection
- Support interrupts in active mode/sleep mode/deep sleep mode
- Support high level/low level/rising edge/falling edge trigger interrupts
- Support bit setting, bit clearing, bit setting and clearing functions

Note: For an introduction to FAST IO, please refer to ARM Cortex-M0+_IntegrationAndImplementation-Manual.pdf.

6.3 Port Controller Function Description

6.3.1 Port Configuration Function

Each port can be configured as an analog port or a digital port through the configuration register (PxADS) according to system requirements. When PxADS is '1', the port is configured as an analog port, and when PxADS is '0', the port is configured as a digital port. The port circuit structure is shown in the figure below:

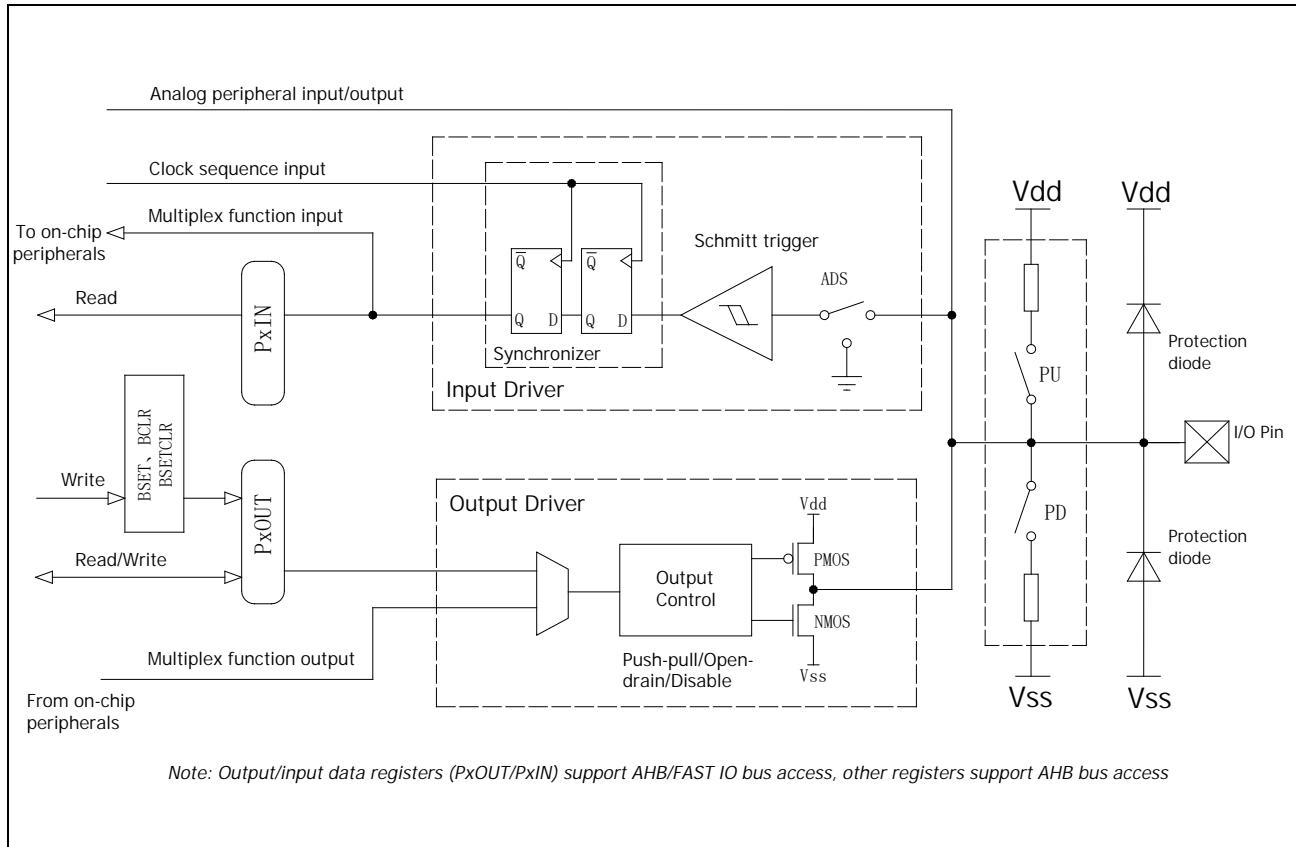


Figure 6-1 Port circuit diagram

When the port is configured as a digital port, it can accept digital general-purpose input and output signals (set the Px_SEL register to '0'), or accept the input and output signals of various functional modules (such as SPI, UART, I2C, Timer, etc.) by configuring the register Px_SEL. For details, see the port digital multiplexing function module.

The corresponding registers can also be configured to implement the following features:

1) Internal pull-up (PxPU)/pull-down (PxPD)

The pull-up register (PxPU) and the pull-down register (PxPD) correspond to the port pull-up enable and port pull-down enable respectively. When the corresponding bit is '1', the pull-up/pull-down enable of the corresponding bit pin is set. When it is '0', the pull-up/pull-down of the corresponding bit pin is disabled.

2) Two-speed drive output (PxDR)

The drive capability can be changed through the PxDR register. When PxDR is '1', it is low drive capability, and when PxDR is '0', it is high drive capability.

3) Open-drain output (PxOD)

The output state of the pin is set through the PxOD register. When PxOD is '1', the port open-drain output is enabled, and when it is '0', the port push-pull output is enabled. When the open-drain pin is not connected to an external pull-up resistor, it can only output a low level. If it is required to have the function of outputting a high level at the same time, a pull-up resistor is required.

4) Direction selection (PxDIR)

It is used to set the direction of the port pin. When PxDIR is '0', the port is output, and when PxDIR is '1', the port is input.

5) Input level status (PxIN)

The synchronized pin level can be obtained by reading the PxIN register. When PxIN is '1', it is a high level, and when PxIN is '0', it is a low level. It can be accessed through the AHB/FAST IO bus.

6) Output high and low level selection (PxOUT)

When the port pin is configured as output, if PxOUT is '1', the port pin outputs a high level, and if PxOUT is '0', it outputs a low level. It can be accessed through the AHB/FAST IO bus.

Position bit (PxBSSET), bit clear (PxBCLR), position bit clear (PxBSSETCLR)

The position bit and bit clear are used to set the bit that the user wants to change without changing the value of other bits to get the corresponding value. Positioning the bit is to set the corresponding value to '1', and clearing the bit is to set the corresponding value to '0'.

Note:

- *The above features are invalid when configured as an analog port.*

The relationship between the port status and register configuration is shown in the following table:

Table 6-1 Port status truth table

IO status	I O mode	PxA DS	PxDI R	PxO UT	PxIN	PxB S ET	PxB C LR	PxBSETC LR	PxP U	PxP D	PxO D	PxD R	Px_S EL
Analog	Input/output	1	W	W	0	W	W	W	W	W	W	W	W
Floating	Input	0	1	W	X	W	W	W	0	0	W	W	0
Pull-down	Input	0	1	W	0	W	W	W	0	1	W	W	0
Pull-up	Input	0	1	W	1	W	W	W	1	0	W	W	0
Pull-up	Input	0	1	W	1	W	W	W	1	1	W	W	0
1	Input	0	1	W	1	W	W	W	W	W	W	W	0
0	Input	0	1	W	0	W	W	W	W	W	W	W	0
1	Output	0	0	1	1	0	0	0	W	W	0	W	0
0	Output	0	0	0	0	0	0	0	W	W	0	W	0
1	Output	0	0	W	1	1	0	0	W	W	0	W	0
0	Output	0	0	W	0	0	1	0	W	W	0	W	0
(SET) 1	Output	0	0	W	(SET) 1	0	0	1	W	W	0	W	0
(CLR) 0	Output	0	0	W	(CLR) 0	0	0	1	W	W	0	W	0
0	Output	0	0	0	0	0	0	0	W	W	1	W	0
z	Output	0	0	1	X	0	0	0	0	0	1	W	0

0	Output	0	0	1	0	0	0	0	0	1	1	W	0
1	Output	0	0	1	1	0	0	0	1	0	1	W	0
1	Output	0	0	1	1	0	0	0	1	1	1	W	0
Note:		0 - Logic low	1 - Logic high	W - Whatever 0 or 1	X - unknown state	Z - high impedance							

6.3.2 Writing to the port

The port input value/output value register (PxIN/PxOUT) supports the reading and writing of AHB bus and FAST IO bus (controlled by register GPIO_CTRL2.ahb_sel bit), and other registers only support the reading and writing of AHB bus. For these two different buses, the system clock (HCLK) has different processing cycles for these two buses. The following two figures show the fastest timing for the two bus port flips:

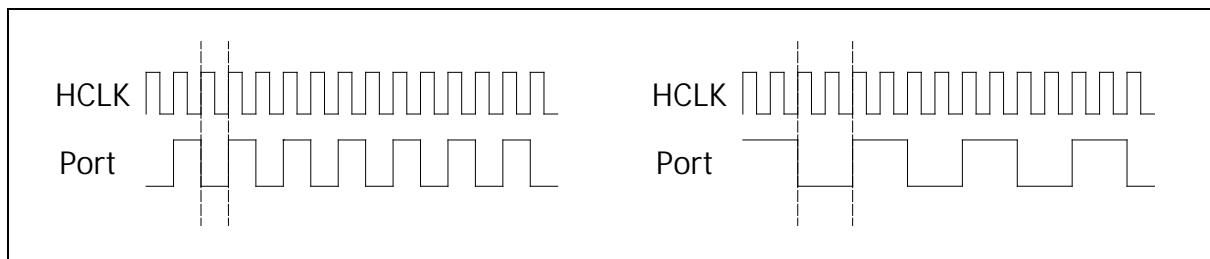


Figure 6-2 AHB/FASTIO bus port changes with system clock (FAST IO on the left, AHB on the right)

For AHB bus, IO flips once every two HCLK cycles, while for FAST IO bus, IO flips once every HCLK cycle.

6.3.3 Reading the port

Each port can obtain the port pin level by reading the PxIN register. As shown in Figure 6-1, the bits of the PxIN register and the latch in front of it form a synchronizer to avoid signal instability caused by pin level changes in a short time when the system clock state changes, but it also introduces delays. The synchronization diagram for reading port pin data is as follows:

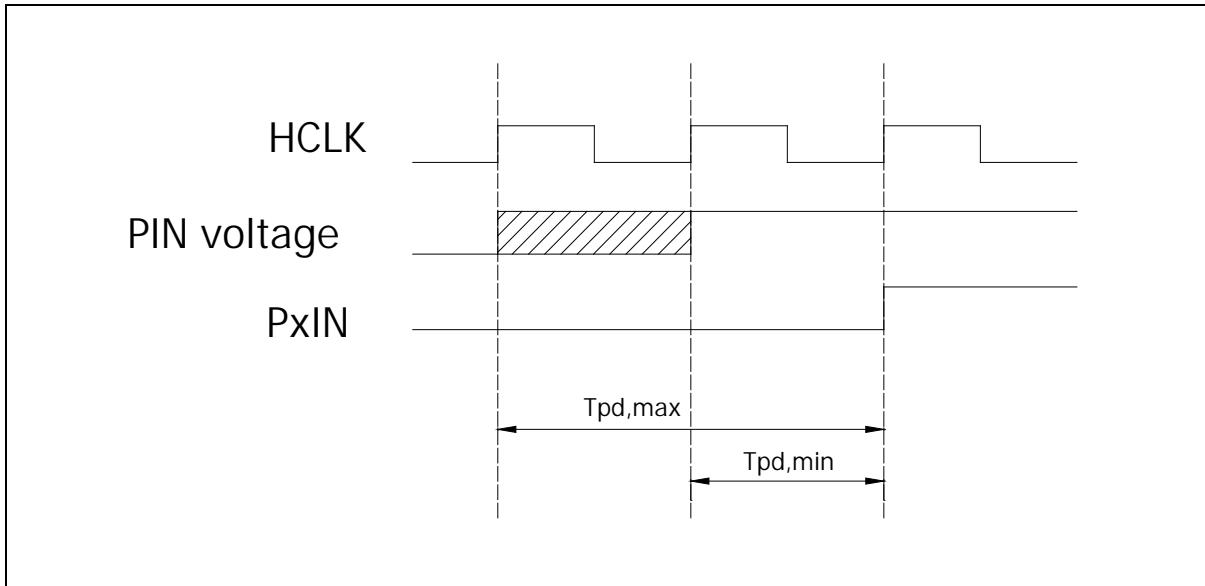


Figure 6-3 Synchronous diagram of reading port pin data

In the clock period after the rising edge of the system clock, the pin level signal will be latched in the internal register, as shown in the shaded part. After the next rising edge of the system clock, the stable pin level signal can be read. Then, at the rising edge of the system clock, the data is latched into the PxIN register. The signal conversion delay Tpd is 1-2 system clocks.

Note:

- *Handling of unconnected pins:*

If there are unconnected pins during use, it is recommended to give the pins a certain level to avoid floating current consumption due to the pins not having a certain level in other digital input enable modes.

6.3.4 Port digital multiplexing function

Port digital multiplexing is one of the main functions of the port controller. By configuring the register, the port can be flexibly configured as a test debug port/digital general port/digital function port.

As shown in the following table: The Px_SEL register is used for digital general port/digital function port switching, and each port can be independently configured as the function port required by the system. For the configuration information of the debug port, please refer to the relevant chapters.

Table 6-2 Port multiplexing table

PxSEL							
0	1	2	3	4	5	6	7
PA0_0	UART1_CTS	LPUART1_TXD	TIM0_ETR	VC0_OUT	TIM1_CHA	TIM3_ETR	TIM0_CHA
PA0_1	UART1_RTS	LPUART1_RXD	TIM0_CHB	TIM1_ETR	TIM1_CHB	HCLK_OUT	SPI1_MOSI
PA0_2	UART1_TXD	TIM0_CHA	VC1_OUT	TIM1_CHA	TIM2_CHA	PCLK_OUT	SPI1_MISO
PA0_3	UART1_RXD	TIM0_GATE	TIM1_CHB	TIM2_CHB	SPI1_CS	TIM3_CH1A	TIM5_CHA
PA0_4	SPI0_CS	UART1_TXD	PCA_CH4	TIM2_ETR	TIM5_CHA	LVD_OUT	TIM3_CH2B
PA0_5	SPI0_SCK	TIM0_ETR	PCA_ECI	TIM0_CHA	TIM5_CHB	XTL_OUT	XTH_OUT
PA0_6	SPI0_MISO	PCA_CH0	TIM3_BK	TIM1_CHA	VC0_OUT	TIM3_GATE	LPUART0_CTS
PA0_7	SPI0_MOSI	PCA_CH1	HCLK_OUT	TIM3_CH0B	TIM2_CHA	VC1_OUT	TIM4_CHB
PA0_8	UART0_TXD	TIM3_CH0A	CRS_SYNC	CAN_STBY	TIM1_GATE	TIM4_CHA	TIM3_BK
PA0_9	UART0_TXD	TIM3_CH1A	TIM0_BK	I2C0_SCL		HCLK_OUT	TIM5_CHA
PA1_0	UART0_RXD	TIM3_CH2A	TIM2_BK	I2C0_SDA	TIM2_GATE	PCLK_OUT	TIM6_CHA
PA1_1	UART0_CTS	TIM3_GATE	I2C1_SCL	CAN_RX	VC0_OUT	SPI0_MISO	TIM4_CHB
PA1_2	UART0_RTS	TIM3_ETR	I2C1_SDA	CAN_TX	VC1_OUT	SPI0_MOSI	PCNT_S0
PA1_3	IR_OUT	UART0_RXD	LVD_OUT	TIM3_ETR	RTC_1HZ	PCNT_S1	VC2_OUT
PA1_4	UART1_TXD	UART0_TXD	TIM3_CH2A	LVD_OUT	RCH_OUT	RCL_OUT	PLL_OUT
PA1_5	SPI0_CS	UART1_RXD	LPUART1 RTS	TIM0_ETR	TIM0_CHA	TIM3_CH1A	
PB0_0	PCA_CH2	TIM3_CH1B	LPUART0_TXD	TIM5_CHB	RCH_OUT	RCL_OUT	PLL_OUT
PB0_1	PCA_CH3	PCLK_OUT	TIM3_CH2B	TIM6_CHB	LPUART0_RT S	VC2_OUT	TCLK_OUT
PB0_2	LPTIMO_TOG	PCA_ECI	LPUART1_TXD	TIM4_CHA	TIM1_BK	TIM0_BK	TIM2_BK
PB0_3	SPI0_SCK	TIM0_CHB	TIM1_GATE	TIM3_CH0A	LPTIMO_GATE	XTL_OUT	XTH_OUT
PB0_4	SPI0_MISO	PCA_CH0	TIM2_BK	UART0_CTS	TIM2_GATE	TIM3_CH0B	LPTIMO_EXT
PB0_5	SPI0_MOSI		TIM1_BK	PCA_CH1	LPTIMO_GATE	PCNT_S0	UART0_RTS
PB0_6	I2C0_SCL	UART0_TXD	TIM1_CHB	TIM0_CHA	LPTIMO_EXT	TIM3_CH0A	LPTIMO_TOG
PB0_7	I2C0_SDA	UART0_RXD	TIM2_CHB	LPUART1_CTS	TIM0_CHB	LPTIMO_TOG N	PCNT_S1

PxSEL							
0	1	2	3	4	5	6	7
PB0_8	I2C0_SCL	TIM1_CHA	CAN_RX	TIM2_CHA	TIM0_GATE	TIM3_CH2A	UART0_TXD
PB0_9	I2C0_SDA	IR_OUT	SPI1_CS	TIM2_CHA	CAN_TX	TIM2_CHB	UART0_RXD
PB1_0	I2C1_SCL	SPI1_SCK	TIM1_CHA	LPUART0_RXD	TIM3_CH1A	LPUART1 RTS	UART1_RTS
PB1_1	I2C1_SDA	TIM1_CHB	LPUART0_RXD	TIM2_GATE	TIM6_CHA	LPUART1_CTS	UART1_CTS
PB1_2	SPI1_CS	TIM3_BK	LPUART0_RXD	TIM0_BK		LPUART0 RTS	TIM6_CHA
PB1_3	SPI1_SCK	I2C1_SCL	TIM3_CH0B	LPUART0_CTS	TIM1_CHA	TIM1_GATE	TIM6_CHB
PB1_4	SPI1_MISO	I2C1_SDA	TIM3_CH1B	TIM0_CHA	RTC_1HZ	LPUART0 RTS	TIM1_BK
PB1_5	SPI1_MOSI	TIM3_CH2B	TIM0_CHB	TIM0_GATE			LPUART1_RXD
PC0_0	LPTIM0_GATE	PCNT_S0	UART1_CTS	UART2_RTS	I2S0_MCK		
PC0_1	LPTIM0_TOG	TIM5_CHB	UART1_RTS	PCNT_S0FO	I2S0_SD	UART2_CTS	
PC0_2	SPI1_MISO	LPTIM0_TOGN	PCNT_S1	UART2_RXD			
PC0_3	SPI1_MOSI	LPTIM0_EXT	LPTIM0_TOGN	PCNT_S1FO	UART2_RXD		
PC0_4	LPUART0_RXD	TIM2_ETR	IR_OUT	VC2_OUT	I2S0_WS		
PC0_5	LPUART0_RXD	TIM6_CHB	PCA_CH4		I2S0_SDIN		
PC0_6	PCA_CH0	TIM4_CHA	TIM2_CHA	LPTIM1_GATE	I2S1_SCK	UART3_RXD	
PC0_7	PCA_CH1	TIM5_CHA	TIM2_CHB	LPTIM1_EXT	I2S1_MCK	UART3_RXD	
PC0_8	PCA_CH2	TIM6_CHA	TIM2_ETR	LPTIM1_TOG	I2S1_SD	UART3_CTS	
PC0_9	PCA_CH3	TIM4_CHB	TIM1_ETR	LPTIM1_TOGN	I2S1_WS	UART3_RTS	
PC1_0	LPUART1_RXD	LPUART0_RXD	PCA_CH2				
PC1_1	LPUART1_RXD	LPUART0_RXD	PCA_CH3	PCNT_S0FO			
PC1_2	LPUART0_RXD	LPUART1_RXD	PCA_CH4	PCNT_S1FO			
PC1_3		RTC_1HZ	TIM3_CH1B		I2S0_SCK		
PC1_4							
PC1_5							
PD0_0	CAN_RX	SPI1_CS					
PD0_1	CAN_TX	SPI1_SCK					
PD0_2	PCA_ECI	LPUART0_RTS	TIM1_ETR				
PD0_3	UART1_CTS	SPI1_MISO	LPTIM1_TOG	I2S1_SCK			
PD0_4	UART1_RTS	SPI1_MOSI	LPTIM1_TOGN	I2S1_MCK			
PD0_5	UART1_RXD	LPTIM1_GATE	CAN_STBY	I2S1_SD			
PD0_6	UART1_RXD	LPTIM1_EXT		I2S1_WS			

PxSEL							
0	1	2	3	4	5	6	7
PD0_7	UART1_TXD			I2S1_SDIN			
PD0_8	LPUART0_RXD	I2S0_SCK					
PD0_9	LPUART0_RXD	I2S0_MCK					
PD1_0	LPUART0_RXD	I2S0_SD					
PD1_1	LPUART0_CTS	I2S0_WS					
PD1_2	LPUART0_RTS	UART2_RTS					
PD1_3	UART2_RXD	I2S0_SDIN					
PD1_4	UART2_TXD						
PD1_5	CRS_SYNC	UART2_CTS					
PE0_0	TIM1_CHA						
PE0_1	TIM2_CHA						
PE0_2	PCA_ECI						
PE0_3	PCA_CH0						
PE0_4	PCA_CH1						
PE0_5	PCA_CH2						
PE0_6	PCA_CH3						
PE0_7	TIM3_ETR	LPTIM1_GATE					
PE0_8	TIM3_CH0B	LPTIM1_EXT					
PE0_9	TIM3_CH0A	LPTIM1_TOG					
PE1_0	TIM3_CH1B	LPTIM1_TOGN					
PE1_1	TIM3_CH1A						
PE1_2	TIM3_CH2B	SPI0_CS	UART3_CTS				
PE1_3	TIM3_CH2A	SPI0_SCK	UART3_RTS				
PE1_4	TIM3_CH0B	SPI0_MISO	UART3_RXD				
PE1_5	TIM3_BK	SPI0_MOSI	UART3_TXD				
PF00	I2C0_SDA	CRS_SYNC	UART1_RXD				
PF01	I2C0_SCL		UART1_RXD				
PF02							
PF03							
PF04							
PF05							
PF06	I2C1_SCL	LPUART1_CTS	UART0_CTS				
PF07	I2C1_SDA	LPUART1_RTS	UART0_RTS				
PF09	TIM0_CHA						
PF10	TIM0_CHB						

PxSEL							
0	1	2	3	4	5	6	7
PF11							

6.3.5 Port interrupt function

Each digital general port can generate an interrupt from an external signal source. The external signal source can be four types of signals: high level/low level/rising edge/falling edge. The corresponding interrupt enable registers are high level interrupt enable register (PxHIE)/low level interrupt enable register (PxLIE)/rising edge interrupt enable register (PxRIE)/falling edge interrupt enable register (PxFIE). When an interrupt is triggered, you can determine which port triggered the interrupt by querying the interrupt status register (Px_STAT), and clear the corresponding interrupt status flag by clearing the interrupt clear register (Px_ICLR).

6.4 Port Configuration Operation Flow

6.4.1 Port multiplexing configuration as analog port operation flow

- a) Set register PxADS[n] to 1

6.4.2 Port multiplexing configuration as digital universal port operation flow

- a) Set register PxADS[n] to 0

- b) Set register Px_SEL to 0

Set register PxDIR[n] to 1: port direction is input, CPU can read port status PxIN[n]

- d) Set register PxDIR[n] to 0: port direction is output

- e) Set register PxOUT[n] to 1: port output high level

- f) Set register PxOUT[n] to 0: port output low level

6.4.3 Port multiplexing configuration as digital function port operation flow

- a) Set register PxADS[n] to 0

- b) Set register Px_SEL to 1~7 (according to system requirements, refer to port multiplexing table)

- c) Set register PxDIR[n] (according to system requirements)

- d) Set register PxPU[n]/PxPD[n]/PxOD[n] (according to system requirements)

6.4.4 Port multiplexing configuration as debugging test port operation flow

Refer to the relevant chapters of test debugging.

6.4.5 Port multiplexing configuration for infrared output signal operation flow

Ports PA13, PB09, PC04 can modulate the internal clock signal with a frequency of 38K into infrared signal output.

- a) Set register PAADS[13]/PBADS[9]/PCADS[4] to 0

- b) Set register PA13_SEL = 1/PB09_SEL = 2/PC04_SEL = 3

- c) Set register PADIR[13] = 0/PBDIR[9] = 0/PCDIR[4] = 0: port direction is output
- d) Set bit14 of register GPIO_CTRL1 to select infrared signal output polarity
- e) Set register PAOUT[13]/PBOUT[9]/PCOUT[4] to gate infrared signal output

6.4.6 Port high level interrupt operation flow

- a) Set register PxADS[n] to 0
- b) Set register Px_SEL to 0
- c) Set register PxDIR[n] to 1
- d) Set register Px_HIE[n] to 1
- e) Read interrupt status register Px_STAT[n] after interrupt is triggered
- f) Set register Px_ICLR[n] to 0 to clear interrupt status register Px_STAT[n]

6.4.7 Port low level interrupt operation flow

- a) Set register PxADS[n] to 0
- b) Set register Px_SEL to 0
- c) Set register PxDIR[n] to 1
- d) Set register Px_LIE[n] to 1
- e) Read interrupt status register Px_STAT[n] after interrupt is triggered
- f) Set register Px_ICLR[n] to 0 Clear interrupt status register Px_STAT[n]

6.4.8 Port rising edge interrupt operation flow

- a) Set register PxADS[n] to 0
- b) Set register Px_SEL to 0
- c) Set register PxDIR[n] to 1
- d) Set register Px_RIE[n] to 1
- e) Read interrupt status register Px_STAT[n] after interrupt is triggered
- f) Set register Px_ICLR[n] to 0 Clear interrupt status register Px_STAT[n]

6.4.9 Port falling edge interrupt operation flow

- a) Set register PxADS[n] to 0
- b) Set register Px_SEL to 0
- c) Set register PxDIR[n] to 1
- Set register Px_FIE[n] to 1
- e) Read interrupt status register Px_STAT[n] after interrupt is triggered
- f) Set register Px_ICLR[n] to 0 Clear interrupt status register Px_STAT[n]

6.4.10 Port pull-up enable configuration operation flow

- a) Set register PxPU[n] to 1

6.4.11 Port pull-down enable configuration operation flow

- a) Set register PxPU[n] to 0

Set register PxPD[n] to 1

Note: When PxPU[n] and PxPD[n] are set to 1 at the same time, PxPU[n] has a higher priority and PxPD[n] is invalid

6.4.12 Port Enhanced Driver Configuration Operation Flow

- a) Set register PxDR[n] to 0

6.4.13 Port open-drain output configuration operation flow

- a) Set register PxOD[n] to 1

6.4.14 Port bit setting operation flow

- a) Set register PxBSET[n] to 1

6.4.15 Port bit clearing operation flow

- a) Set register PxBCLR[n] to 1

6.4.16 Port bit setting and clearing operation flow

- a) Set register PxBSETCLR[n] to 1

6.5 Port Controller Register Description

Register List 1

Base Address 1: 0x40020C00

Offset	Register Name	Access	Register Description
0x00	PA00_SEL	RW	Port PA00 Function Configuration Register
0x04	PA01_SEL	RW	Port PA01 Function Configuration Register
0x08	PA02_SEL	RW	Port PA02 Function Configuration Register
0x0c	PA03_SEL	RW	Port PA03 Function Configuration Register
0x10	PA04_SEL	RW	Port PA04 Function Configuration Register
0x14	PA05_SEL	RW	Port PA05 Function Configuration Register
0x18	PA06_SEL	RW	Port PA06 Function Configuration Register
0x1c	PA07_SEL	RW	Port PA07 Function Configuration Register
0x20	PA08_SEL	RW	Port PA08 Function Configuration Register
0x24	PA09_SEL	RW	Port PA09 Function Configuration Register
0x28	PA10_SEL	RW	Port PA10 Function Configuration Register
0x2c	PA11_SEL	RW	Port PA11 Function Configuration Register
0x30	PA12_SEL	RW	Port PA12 Function Configuration Register
0x34	PA13_SEL	RW	Port PA13 Function Configuration Register
0x38	PA14_SEL	RW	Port PA14 Function Configuration Register
0x3c	PA15_SEL	RW	Port PA15 Function Configuration Register
0x40	PB00_SEL	RW	Port PB00 Function Configuration Register
0x44	PB01_SEL	RW	Port PB01 Function Configuration Register
0x48	PB02_SEL	RW	Port PB02 Function Configuration Register
0x4c	PB03_SEL	RW	Port PB03 Function Configuration Register
0x50	PB04_SEL	RW	Port PB04 Function Configuration Register
0x54	PB05_SEL	RW	Port PB05 Function Configuration Register
0x58	PB06_SEL	RW	Port PB06 Function Configuration Register
0x5c	PB07_SEL	RW	Port PB07 Function Configuration Register
0x60	PB08_SEL	RW	Port PB08 Function Configuration Register
0x64	PB09_SEL	RW	Port PB09 Function Configuration Register
0x68	PB10_SEL	RW	Port PB10 Function Configuration Register
0x6c	PB11_SEL	RW	Port PB11 Function Configuration Register
0x70	PB12_SEL	RW	Port PB12 Function Configuration Register
0x74	PB13_SEL	RW	Port PB13 Function Configuration Register
0x78	PB14_SEL	RW	Port PB14 Function Configuration Register
0x7c	PB15_SEL	RW	Port PB15 Function Configuration Register
0x80	PC00_SEL	RW	Port PC00 Function Configuration Register

Offset	Register Name	Access	Register Description
0x84	PC01_SEL	RW	Port PC01 Function Configuration Register
0x88	PC02_SEL	RW	Port PC02 Function Configuration Register
0x8c	PC03_SEL	RW	Port PC03 Function Configuration Register
0x90	PC04_SEL	RW	Port PC04 Function Configuration Register
0x94	PC05_SEL	RW	Port PC05 Function Configuration Register
0x98	PC06_SEL	RW	Port PC06 Function Configuration Register
0x9c	PC07_SEL	RW	Port PC07 Function Configuration Register
0xa0	PC08_SEL	RW	Port PC08 Function Configuration Register
0xa4	PC09_SEL	RW	Port PC09 Function Configuration Register
0xa8	PC10_SEL	RW	Port PC10 Function Configuration Register
0xac	PC11_SEL	RW	Port PC11 Function Configuration Register
0xb0	PC12_SEL	RW	Port PC12 Function Configuration Register
0xb4	PC13_SEL	RW	Port PC13 Function Configuration Register
0xb8	PC14_SEL	RW	Port PC14 Function Configuration Register
0xbc	PC15_SEL	RW	Port PC15 Function Configuration Register
0xc0	PD00_SEL	RW	Port PD00 Function Configuration Register
0xc4	PD01_SEL	RW	Port PD01 Function Configuration Register
0xc8	PD02_SEL	RW	Port PD02 Function Configuration Register
0xcc	PD03_SEL	RW	Port PD03 Function Configuration Register
0xd0	PD04_SEL	RW	Port PD04 Function Configuration Register
0xd4	PD05_SEL	RW	Port PD05 Function Configuration Register
0xd8	PD06_SEL	RW	Port PD06 Function Configuration Register
0xdc	PD07_SEL	RW	Port PD07 Function Configuration Register
0xe0	PD08_SEL	RW	Port PD08 Function Configuration Register
0xe4	PD09_SEL	RW	Port PD09 Function Configuration Register
0xe8	PD10_SEL	RW	Port PD10 Function Configuration Register
0xec	PD11_SEL	RW	Port PD11 Function Configuration Register
0xf0	PD12_SEL	RW	Port PD12 Function Configuration Register
0xf4	PD13_SEL	RW	Port PD13 Function Configuration Register
0xf8	PD14_SEL	RW	Port PD14 Function Configuration Register
0xfc	PD15_SEL	RW	Port PD15 Function Configuration Register
0x100	PADIR	RW	Port PA input and output configuration register
0x104	PAIN	RO	Port PA input value register
0x108	PAOUT	RW	Port PA output value configuration register
0x10c	PAADS	RW	Port PA digital-analog configuration register
0x110	PABSET	RW	Port PA bit set register
0x114	PABCLR	RW	Port PA bit clear register
0x118	PABSETCLR	RW	Port PA bit set/clear register

Offset	Register Name	Access	Register Description
0x11c	PADR	RW	Port PA drive capability configuration register
0x120	PAPU	RW	Port PA pull-up enable configuration register
0x124	PAPD	RW	Port PA pull-down enable configuration register
0x12c	PAOD	RW	Port PA open-drain output configuration register
0x130	PAHIE	RW	Port PA high-level interrupt enable configuration register
0x134	PALIE	RW	Port PA low-level interrupt enable configuration register
0x138	PARIE	RW	Port PA rising edge interrupt enable configuration register
0x13c	PAFIE	RW	Port PA falling edge interrupt enable configuration register
0x200	PA_STAT	RO	Port PA interrupt status register
0x210	PA_ICLR	RW	Port PA interrupt clear register
0x140	PBDIR	RW	Port PB input and output configuration register
0x144	PBIN	RO	Port PB input value register
0x148	PBOUT	RW	Port PB output value configuration register
0x14c	PBADS	RW	Port PB digital-analog configuration register
0x150	PBBSET	RW	Port PB bit set register
0x154	PBBCLR	RW	Port PB bit clear register
0x158	PBBSETCLR	RW	Port PB bit set/clear register
0x15c	PBDR	RW	Port PB drive capability configuration register
0x160	PBPU	RW	Port PB pull-up enable configuration register
0x164	PBPD	RW	Port Pb pull-down enable configuration register
0x16c	PBOD	RW	Port PB open-drain output configuration register
0x170	PBHIE	RW	Port PB high-level interrupt enable configuration register
0x174	PBLIE	RW	Port PB low-level interrupt enable configuration register
0x178	PBRIE	RW	Port PB rising edge interrupt enable configuration register
0x17c	PBFIE	RW	Port PB falling edge interrupt enable configuration register
0x240	PB_STAT	RO	Port PB interrupt status register
0x250	PB_ICLR	RW	Port PB interrupt clear register
0x180	PCDIR	RW	Port PC input and output configuration register
0x184	PCIN	RO	Port P input value register
0x188	PCOUT	RW	Port PC output value configuration register
0x18c	PCADS	RW	Port PC digital-analog configuration register
0x190	PCBSET	RW	Port PC bit set register
0x194	PCBCLR	RW	Port PC bit clear register
0x198	PCBSETCLR	RW	Port PC bit set/clear register
0x19c	PCDR	RW	Port PC drive capability configuration register
0x1a0	PCPU	RW	Port PC pull-up enable configuration register
0x1a4	PCPD	RW	Port PC pull-down enable configuration register
0x1ac	PCOD	RW	Port PC open-drain output configuration register

Offset	Register Name	Access	Register Description
0x1b0	PCHIE	RW	Port PC high-level interrupt enable configuration register
0x1b4	PCLIE	RW	Port PC low-level interrupt enable configuration register
0x1b8	PCRIE	RW	Port PC rising edge interrupt enable configuration register
0x1bc	PCFIE	RW	Port PC falling edge interrupt enable configuration register
0x280	PC_STAT	RO	Port PC interrupt status register
0x290	PC_ICLR	RW	Port PC interrupt clear register
0x1c0	PDDR	RW	Port PD input and output configuration register
0x1c4	PDIN	RO	Port PD input value register
0x1c8	PDOOUT	RW	Port PD output value configuration register
0x1cc	PDADS	RW	Port PD digital-analog configuration register
0x1d0	PDBSET	RW	Port PD bit set register
0x1d4	PDBCLR	RW	Port PD bit clear register
0x1d8	PDBSETCLR	RW	Port PD bit set/clear register
0x1dc	PDDR	RW	Port PD drive capability configuration register
0x1e0	PDPU	RW	Port PD pull-up enable configuration register
0x1e4	PDPD	RW	Port PD pull-down enable configuration register
0x1ec	PDOD	RW	Port PD open-drain output configuration register
0x1f0	PDHIE	RW	Port PD high-level interrupt enable configuration register
0x1f4	PDLIE	RW	Port PD low-level interrupt enable configuration register
0x1f8	PDRIE	RW	Port PD rising edge interrupt enable configuration register
0x1fc	PDFIE	RW	Port PD falling edge interrupt enable configuration register
0x2c0	PD_STAT	RO	Port PD interrupt status register
0x2d0	PD_ICLR	RW	Port PD interrupt clear register
0x304	GPIO_CTRL1	RW	Port auxiliary function configuration register 1
0x308	GPIO_CTRL2	RW	Port auxiliary function configuration register 2
0x30c	GPIO_TIMGS	RW	Port auxiliary function timer gate selection
0x310	GPIO_TIMES	RW	Port auxiliary function timer ETR selection
0x314	GPIO_TIMCPS	RW	Port auxiliary function timer capture input selection
0x318	GPIO_PCAS	RW	Port auxiliary function PCA capture selection
0x31c	GPIO_PCNT	RW	Port auxiliary function PCNT input selection
0x1000	PE00_SEL	RW	Port PE00 Function Configuration Register
0x1004	PE01_SEL	RW	Port PE01 Function Configuration Register
0x1008	PE02_SEL	RW	Port PE02 Function Configuration Register
0x100c	PE03_SEL	RW	Port PE03 Function Configuration Register
0x1010	PE04_SEL	RW	Port PE04 Function Configuration Register
0x1014	PE05_SEL	RW	Port PE05 Function Configuration Register
0x1018	PE06_SEL	RW	Port PE06 Function Configuration Register
0x101c	PE07_SEL	RW	Port PE07 Function Configuration Register

Offset	Register Name	Access	Register Description
0x1020	PE08_SEL	RW	Port PE08 Function Configuration Register
0x1024	PE09_SEL	RW	Port PE09 Function Configuration Register
0x1028	PE10_SEL	RW	Port PE10 Function Configuration Register
0x102c	PE11_SEL	RW	Port PE11 Function Configuration Register
0x1030	PE12_SEL	RW	Port PE12 Function Configuration Register
0x1034	PE13_SEL	RW	Port PE13 Function Configuration Register
0x1038	PE14_SEL	RW	Port PE14 Function Configuration Register
0x103c	PE15_SEL	RW	Port PE15 Function Configuration Register
0x1040	PF00_SEL	RW	Port PF00 Function Configuration Register
0x1044	PF01_SEL	RW	Port PF01 Function Configuration Register
0x1048	PF02_SEL	RW	Port PF02 Function Configuration Register
0x104c	PF03_SEL	RW	Port PF03 Function Configuration Register
0x1050	PF04_SEL	RW	Port PF04 Function Configuration Register
0x1054	PF05_SEL	RW	Port PF05 Function Configuration Register
0x1058	PF06_SEL	RW	Port PF06 Function Configuration Register
0x105c	PF07_SEL	RW	Port PF07 Function Configuration Register
0x1064	PF09_SEL	RW	Port PF09 Function Configuration Register
0x1068	PF10_SEL	RW	Port PF10 Function Configuration Register
0x106c	PF11_SEL	RW	Port PF11 Function Configuration Register
0x1100	PEDIR	RW	Port PE input and output configuration register
0x1104	PEIN	RO	Port PE input value register
0x1108	PEOUT	RW	Port PE output value configuration register
0x110c	PEADS	RW	Port PE digital-analog configuration register
0x1110	PEBSET	RW	Port PE bit set register
0x1114	PEBCLR	RW	Port PE bit clear register
0x1118	PEBSETCLR	RW	Port PE bit set/clear register
0x111c	PEDR	RW	Port PE drive capability configuration register
0x1120	PEPU	RW	Port PE pull-up enable configuration register
0x1124	PEPD	RW	Port PE pull-down enable configuration register
0x112c	PEOD	RW	Port PE open-drain output configuration register
0x1130	PEHIE	RW	Port PE high-level interrupt enable configuration register
0x1134	PELIE	RW	Port PE low-level interrupt enable configuration register
0x1138	PERIE	RW	Port PE rising edge interrupt enable configuration register
0x113c	PEFIE	RW	Port PE falling edge interrupt enable configuration register
0x1200	PE_STAT	RO	Port PE interrupt status register
0x1210	PE_ICLR	RW	Port PE interrupt clear register
0x1140	PFDIR	RW	Port PF input and output configuration register
0x1144	PFIN	RO	Port PF input value register

Offset	Register Name	Access	Register Description
0x1148	PFOUT	RW	Port PF output value configuration register
0x114c	PFADS	RW	Port PF digital-analog configuration register
0x1150	PFBSET	RW	Port PF bit set register
0x1154	PFBCLR	RW	Port PF bit clear register
0x1158	PFBSETCLR	RW	Port PF bit set/clear register
0x115c	PFDR	RW	Port PF drive capability configuration register
0x1160	PFPU	RW	Port PF pull-up enable configuration register
0x1164	PFPD	RW	Port PF pull-down enable configuration register
0x116c	PFOD	RW	Port PF open-drain output configuration register
0x1170	PFHIE	RW	Port PF high-level interrupt enable configuration register
0x1174	PFLIE	RW	Port PF low-level interrupt enable configuration register
0x1178	PFRIE	RW	Port PF rising edge interrupt enable configuration register
0x117c	PFFIE	RW	Port PF falling edge interrupt enable configuration register
0x1240	PF_STAT	RO	Port PF interrupt status register
0x1250	PF_ICLR	RW	Port PF interrupt clear register

6.5.1 Port General Registers

6.5.1.1 Port Px Input/Output Configuration Register (PxDIR) (x = A,B,C,D,E,F)

Address offset:

0x100(PA),0x140(PB),0x180(PC),0x1C0(PD),0x1100(PE),0x1140(PF)

Reset value: 0xffff ffff

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxDIR	Port Px input and output configuration register (corresponding to Px15-Px00) 1: Configure as input 0: Configure as output <i>Note: Each bit corresponds to a port, for example: PxDIR[15] corresponds to port Px15</i>

6.5.1.2 Port Px Input Value Register (PxIN) (x = A,B,C,D,E,F)

Address offset:

0x104(PA),0x144(PB),0x184(PC),0x1C4(PD),0x1104(PE),0x1144(PF)

Reset value: NA

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxIN	Port Px input value register (corresponding to Px15-Px00) 1: Input is high level 0: Input is low level

6.5.1.3 Port Px Output Value Configuration Register (PxOUT) ($x = A, B, C, D, E, F$)

Address offset:

0x108(PA),0x148(PB),0x188(PC),0x1C8(PD),0x1108(PE),0x1148(PF)

Reset value: NA

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxOUT	Port Px output value configuration register (corresponding to Px15-Px00) 1: Output high level. If configured as open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level.

6.5.1.4 Port Px Digital-to-Analog Configuration Register (PxADS) ($x = A, B, C, D, E, F$)

Address offset:

0x10C(PA),0x14C(PB),0x18C(PC),0x1CC(PD),0x110C(PE),0x114C(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxADS	Port Px digital-analog configuration register (corresponding to Px15-Px00) 1: Configure as analog port 0: Configure as digital port

6.5.1.5 Port Px Bit Set Register (PxBSSET) ($x = A, B, C, D, E, F$)

Address offset:

0x110(PA),0x150(PB),0x190(PC),0x1D0(PD),0x1110(PE),0x1150(PF)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxBSSET[15:0]															
W1															

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxBSET	Port Px bit set register (corresponding to Px15-Px00) 1: set 0: keep

6.5.1.6 Port Px Bit Clear Register (PxBCCLR) ($x = A, B, C, D, E, F$)

Address offset:

0x114(PA),0x154(PB),0x194(PC),0x1D4(PD),0x1114(PE),0x1154(PF)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxBCCLR[15:0]															
W1															

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxBCLR	Port Px bit clear register (corresponding to Px15-Px00) 1: clear 0: keep

6.5.1.7 Port Px Bit Set Clear Register (PxBSSETCLR) (x = A,B,C,D,E,F)

Address offset:

0x118(PA),0x158(PB),0x198(PC),0x1D8(PD),0x1118(PE),0x1158(PF)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PxBSSET[15:0]															
W1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxBCCLR[15:0]															
W1															

Bit	Mark	Function Description
31:16	PxBSET	Port Px bit set register 1: set 0: keep
15:0	PxBCLR	Port Px bit clear register 1: clear 0: keep

Note:

- When the same bit of PxBSSET and PxBCCLR is set to 1 at the same time, PxBCCLR has higher priority. That is, the port is cleared.

6.5.1.8 Port Px Pull-up Enable Configuration Register (PxPU) (x = A,B,C,D,E,F)

Address offset:

0x120(PA),0x160(PB),0x1A0(PC),0x1E0(PD),0x1120(PE),0x1160(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxPU	Port Px pull-up enable configuration register 1: Enable 0: Disable

6.5.1.9 Port Px Drive Capability Configuration Register (PxDR) ($x = A, B, C, D, E, F$)

Address offset:

0x11C(PA),0x15C(PB),0x19C(PC),0x1DC(PD),0x111C(PE),0x115C(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxDR	Port Px drive capability configuration register 1: low drive capability 0: high drive capability

6.5.1.10 Port Px Pull-down Enable Configuration Register (PxPD) ($x = A, B, C, D, E, F$)

Address offset:

0x124(PA),0x164(PB),0x1A4(PC),0x1E4(PD),0x1124(PE),0x1164(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxPD	Port Px pull-down enable configuration register 1: Enable 0: Disable

6.5.1.11 Port Px Open-Drain Output Configuration Register (PxOD) ($x = A, B, C, D, E, F$)

Address offset:

0x12C(PA),0x16C(PB),0x1AC(PC),0x1EC(PD),0x112C(PE),0x116C(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxOD	Port Px open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output

6.5.1.12 Port Px High Interrupt Enable Configuration Register (PxHIE) ($x = A, B, C, D, E, F$)

Address offset:

0x130(PA),0x170(PB),0x1B0(PC),0x1F0(PD),0x1130(PE),0x1170(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxHIE	Port Px high level interrupt enable configuration register 1: Enable 0: Disable

6.5.1.13 Port Px Low Level Interrupt Enable Configuration Register (PxLIE) ($x = A, B, C, D, E, F$)

Address offset:

0x134(PA),0x174(PB),0x1B4(PC),0x1F4(PD),0x1134(PE),0x1174(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxLIE	Port Px low level interrupt enable configuration register 1: Enable 0: Disable

6.5.1.14 Port Px rising edge interrupt enable configuration register (PxRIE) ($x = A, B, C, D, E, F$)

Address offset:

0x138(PA),0x178(PB),0x1B8(PC),0x1F8(PD),0x1138(PE),0x1178(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxRIE	Port Px rising edge interrupt enable configuration register 1: Enable 0: Disable

6.5.1.15 Port Px Falling Edge Interrupt Enable Configuration Register (PxFIE) (x = A,B,C,D,E,F)

Address offset:

0x13C(PA),0x17C(PB),0x1BC(PC),0x1FC(PD),0x113C(PE),0x117C(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	PxFIE	Port Px falling edge interrupt enable configuration register 1: Enable 0: Disable

6.5.1.16 Port Px Interrupt Status Register (Px_STAT) (x = A,B,C,D,E,F)

Address offset:

0x200(PA),0x240(PB),0x280(PC),0x2C0(PD),0x1200(PE),0x1240(PF)

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	Px_STAT	Port Px interrupt status register 1: interrupt triggered 0: no interrupt triggered

6.5.1.17 Port Px Interrupt Clear Register (Px_ICLR) ($x = A, B, C, D, E, F$)

Address offset:

0x210(PA),0x250(PB),0x290(PC),0x2D0(PD),0x1210(PE),0x1250(PF)

Reset value: 0xffff ffff

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

Bit	Mark	Function Description
31:16	Reserved	Reserved
15:0	Px_ICLR	Port Px interrupt clear register 1: keep interrupt flag bit 0: clear interrupt flag bit

6.5.2 Port Auxiliary Registers

6.5.2.1 Port auxiliary function configuration register 1 (GPIO_CTRL1)

Address offset: 0x304

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	ir_po I	hclk_en	pclk_en	hclk_sel	pclk_sel		ssn0_sel					ext_clk_sel			
	rw	rw	rw	rw	rw		rw					rw			

Bit	Mark	Function Description
31:15	Reserved	Reserved
14	ir_pol	IR output polarity selection. 0 - Forward output 1 - Reverse output
13	hclk_en	HCLK output control 0 - Output 0 1 - Output HCLK
12	pclk_en	PCLK output control 0 - Output 0 1 - Output PCLK
11:10	hclk_sel	HCLK output frequency division selection 00: HCLK 01: HCLK/2 10: HCLK/4 11: HCLK/8
9:8	pclk_sel	PCLK output frequency division selection 00: PCLK 01: PCLK/2 10: PCLK/4 11: PCLK/8
7:4	ssn0_sel	SPI0 SSN signal source selection 0000: 高电平 1000: PB01 0001: PA03 1001: PB02 0010: PA04 1010: PB05 0011: PA06 1011: PB06 0100: PA08 1100: PB09 0101: PA09 1101: PB10 0110: PA12 1110: PB12 0111: PA15 1111: PB14
3:0	ext_clk_sel	External clock signal source selection 0000: 高电平 1000: PB01

		0001: PA03	1001: PB02
		0010: PA04	1010: PB05
		0011: PA06	1011: PB06
		0100: PA08	1100: PB09
		0101: PA09	1101: PB10
		0110: PA12	1110: PB12
		0111: PA15	1111: PB14

6.5.2.2 Port auxiliary function configuration register 2 (GPIO_CTRL2)

Address offset: 0x308

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ahb_sel	Res						tclk_div	tclk_sel	ssn1_sel						
rw							rw	rw							rw

Bit	Mark	Function Description																																
31:4	Reserved	Reserved																																
15	ahb_sel	AHB bus access mode 0 - Use FASTIO bus to access PxIN, PxOUT registers 1 -- Use AHB bus to access PxIN, PxOUT registers																																
14:8	Reserved	Reserved																																
7:6	tclk_div	TCLK output frequency division selection 00: TCLK 01: TCLK/2 10: TCLK/4 11: TCLK/8																																
5:4	tclk_sel	TCLK output signal source selection <table border="1" style="margin-left: 10px;"> <tr><td>0 0</td><td>RCH48M</td></tr> <tr><td>0 1</td><td>RCH</td></tr> <tr><td>1 0</td><td>XTH</td></tr> <tr><td>1 1</td><td>PLL</td></tr> </table>	0 0	RCH48M	0 1	RCH	1 0	XTH	1 1	PLL																								
0 0	RCH48M																																	
0 1	RCH																																	
1 0	XTH																																	
1 1	PLL																																	
3:0	ssn1_sel	SPI1 SSN signal source selection <table border="1" style="margin-left: 10px;"> <tr><td>0000</td><td>高电平</td></tr> <tr><td>0001</td><td>PA03</td></tr> <tr><td>0010</td><td>PA04</td></tr> <tr><td>0011</td><td>PA06</td></tr> <tr><td>0100</td><td>PA08</td></tr> <tr><td>0101</td><td>PA09</td></tr> <tr><td>0110</td><td>PA12</td></tr> <tr><td>0111</td><td>PA15</td></tr> <tr><td>1000</td><td>PB01</td></tr> <tr><td>1001</td><td>PB02</td></tr> <tr><td>1010</td><td>PB05</td></tr> <tr><td>1011</td><td>PB06</td></tr> <tr><td>1100</td><td>PB09</td></tr> <tr><td>1101</td><td>PB10</td></tr> <tr><td>1110</td><td>PB12</td></tr> <tr><td>1111</td><td>PB14</td></tr> </table>	0000	高电平	0001	PA03	0010	PA04	0011	PA06	0100	PA08	0101	PA09	0110	PA12	0111	PA15	1000	PB01	1001	PB02	1010	PB05	1011	PB06	1100	PB09	1101	PB10	1110	PB12	1111	PB14
0000	高电平																																	
0001	PA03																																	
0010	PA04																																	
0011	PA06																																	
0100	PA08																																	
0101	PA09																																	
0110	PA12																																	
0111	PA15																																	
1000	PB01																																	
1001	PB02																																	
1010	PB05																																	
1011	PB06																																	
1100	PB09																																	
1101	PB10																																	
1110	PB12																																	
1111	PB14																																	

6.5.2.3 Port Auxiliary Function Timer Gating Selection (GPIO_TIMGS)

Address offset: 0x30C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		LPTIM0_G		TIM3_G		TIM2_G		TIM1_G		TIM0_G					
		R/W		R/W		R/W		R/W		R/W					

Bit	Mark	Function Description
31:15	Reserved	Reserved
14:12	LPTIM0_G	LPTimer0 timer GATE input selection, see the table below for selection
11:9	TIM3_G	Timer3 timer GATE input selection, see the table below for selection
8:6	TIM2_G	Timer2 timer GATE input selection, see the table below for selection
5:3	TIM1_G	Timer1 timer GATE input selection, see the table below for selection
2:0	TIM0_G	Timer0 timer GATE input selection, see the table below for selection

	TIM0_g	TIM1_g	TIM2_g	TIM3_g	LPTIM0_g
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART0_RXD	LPUART0_RXD	UART0_RXD	UART0_RXD	LPUART0_RXD
010	UART1_RXD	LPUART1_RXD	UART1_RXD	UART1_RXD	LPUART1_RXD
011	VC0_OUT	VC0_OUT	VC0_OUT	UART2	VC0_OUT
100	VC1_OUT	VC1_OUT	VC1_OUT	UART3	VC1_OUT
101	PA03	PA08	PA10	VC0_OUT	PB03
110	PB08	PB03	PB04	PA06	PB05
111	PB15	PB13	PB11	PA11	PC00

6.5.2.4 Port auxiliary function timer ETR selection (GPIO_TIMES)

Address offset: 0x310

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		LPTIMO_E		TIM3_E		TIM2_E		TIM1_E		TIMO_E					
		R/W		R/W		R/W		R/W		R/W					

Bit	Mark	Function Description
31:15	Reserved	Reserved
14:12	LPTIMO_E	LPTimer0 timer EXT input selection, see the table below for selection
11:9	TIM3_E	Timer3 timer ETR input selection, see the table below for selection
8:6	TIM2_E	Timer2 timer ETR input selection, see the table below for selection
5:3	TIM1_E	Timer1 timer ETR input selection, see the table below for selection
2:0	TIMO_E	Timer0 timer ETR input selection, see the table below for selection

	TIM0_e	TIM1_e	TIM2_e	TIM3_e	LPTIMO_E
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	LPUART0_RXD	UART0_RXD	LPUART0_RXD	UART0_RXD	PCNT_S0
010	LPUART1_RXD	UART1_RXD	LPUART1_RXD	UART1_RXD	LVD_OUT
011	VCO_OUT	VC1_OUT	VCO_OUT	VC1_OUT	VC0_OUT
100	LVD_OUT	LVD_OUT	PCNT_S1	PCNT_S0	VC1_OUT
101	PA00	PA01	PA04	PA00	PB04
110	PA05	PC09	PC04	PA12	PB06
111	PA15	PD02	PC08	PA13	PC03

6.5.2.5 Port Auxiliary Function Timer Capture Input Select (GPIO_TIMCPS)

Address offset: 0x314

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		TIM3_CB		TIM3_CA		TIM2_CA		TIM1_CA		TIM0_CA					
		R/W													

Bit	Mark	Function Description
31:15	Reserved	Reserved
14:12	TIM3_CB	Timer3 timer CH0B input selection, see the table below for selection
11:9	TIM3_CA	Timer3 timer CH0A input selection, see the table below for selection
8:6	TIM2_CA	Timer2 timer CHA input selection, see the table below for selection
5:3	TIM1_CA	Timer1 timer CHA input selection, see the table below for selection
2:0	TIM0_CA	Timer0 timer CHA input selection, see the table below for selection

	TIM0_CHA	TIM1_CHA	TIM2_CHA	TIM3_CH0A	TIM3_CH0B
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART0_RXD	UART1_RXD	LPUART0_RXD	LPUART1_RXD	UART0_RXD
010	PA00	PA00	VC0_OUT	LPUART0_RXD	UART1_RXD
011	PA02	PA02	PA02	PCNT_S0	PCNT_S1
100	PA05	PA06	PA07	VC0_OUT	VC1_OUT
101	PA15	PB08	PB08	PA08	PA07
110	PB06	PB10	PB09	PB03	PB04
111	PB14	PB13	PC06	PB06	PB13

6.5.2.6 Port Auxiliary Function PCA Capture Select (GPIO_PCAS)

Address offset: 0x318

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				LPTIM1_E		LPTIM1_G		PCA_CH0		PCA_ECI				R/W	

Bit	Mark	Function Description
31:12	Reserved	Reserved
11:9	LPTIM1_E	LPTimer1 timer EXT input selection, see the table below for selection
8:6	LPTIM1_G	LPTimer1 timer GATE input selection, see the table below for selection
5:3	PCA_ECI	PCA_ECI PCA ECI clock input selection, see the table below for selection
2:0	PCA_CH0	PCA CH0 capture port input selection, see the table below for selection

	LPTIM1_E	LPTIM1_G	pca_eci	pca_ch0
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART1	UART0	PCNT_S1	PCNT_S0
010	LPUART1_RXD	LPUART0_RXD	LVD_OUT	PCNT_S1
011	VC0_OUT	VC1_OUT	VC0_OUT	LVD_OUT
100	VC2_OUT	VC2_OUT	VC1_OUT	VC1_OUT
101	LPTIM0_TOG	PD5	PA05	PA06
110	PE8	PE7	PB02	PB04
111	PC7	PC6	PD02	PC06

6.5.2.7 Port auxiliary function PCNT pulse input selection (GPIO_PCNT)

Address offset: 0x31C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
												PCNT_S1	PCNT_S0		
												R/W	R/W		

Bit	Mark	Function Description
31:4	Reserved	Reserved
3:2	PCNT_S1	PCNT_S1 input selection, see the table below for selection
1:0	PCNT_S0	PCNT_S0 input selection, see the table below for selection

	PCNT_S1	PCNT_S0
00	PX_SEL	PX_SEL
01	VC0_OUT	VC0_OUT
10	VC1_OUT	VC1_OUT
11	VC2_OUT	VC2_OUT

6.5.3 Port PA function selection registers

6.5.3.1 Port PA00 Function Configuration Register (PA00_SEL)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PA00_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA00_sel	Port PA00 function selection. 000 ---- GPIO PA00 001 ---- UART1_CTS UART1 module CTS signal 010 ---- LPUART1_TXD LPUART1 module TXD signal 011 ---- TIM0_ETR Timer0 module external clock input signal 100 ---- VC0_OUT VC0 module output/reverse output signal 101 ---- TIM1_CHA Timer1 module channel A signal 110 ---- TIM3_ETR Timer3 module external clock input signal 111 ---- TIM0_CHA Timer0 module channel A signal

6.5.3.2 Port PA01 Function Configuration Register (PA01_SEL)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA01_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA01_sel	Port PA01 function selection. 000 ---- GPIO PA01 001 ---- UART1_RTS UART1 module RTS signal 010 ---- LPUART1_RXD LPUART1 module RXD signal 011 ---- TIM0_CHB Timer0 module channel B signal 100 ---- TIM1_ETR Timer1 module external clock input signal 101 ---- TIM1_CHB Timer1 module channel B signal 110 ---- HCLK_OUT AHB bus clock output signal 111 ---- SPI1_MOSI SPI1 module master output slave input data signal

6.5.3.3 Port PA02 Function Configuration Register (PA02_SEL)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA02_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA02_sel	Port PA02 function selection. 000 ---- GPIO PA02 001 ---- UART1_TXD UART1 module TXD signal 010 ---- TIM0_CHA Timer0 module channel A signal 011 ---- VC1_OUT VC1 module output/inverse output signal 100 ---- TIM1_CHA Timer1 module channel A signal 101 ---- TIM2_CHA Timer2 module channel A signal 110 ---- PCLK_OUT APB bus clock output signal 111 ---- SPI1_MISO SPI1 module master input slave output data signal

6.5.3.4 Port PA03 Function Configuration Register (PA03_SEL)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA03_sel	rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA03_sel	Port PA03 function selection. 000 ---- GPIO PA03 001 ---- UART1_RXD UART1 module RXD signal 010 ---- TIM0_GATE Timer0 module gate signal 011 ---- TIM1_CHB Timer1 module channel B signal 100 ---- TIM2_CHB Timer2 module channel B signal 101 ---- SPI1_CS SPI1 module host mode chip select signal 110 ---- TIM3_CH1A Timer3 module channel 1A signal 111 ---- TIM5_CHA Timer6 module channel 1A signal

6.5.3.5 Port PA04 Function Configuration Register (PA04_SEL)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA04_sel	rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA04_sel	Port PA04 function selection. 000 ---- GPIO PA04 001 ---- SPI0_CS SPI0 module host mode chip select signal 010 ---- UART1_TXD UART1 module TXD signal 011 ---- PCA_CH4 PCA module channel 4 capture/compare signal 100 ---- TIM2_ETR Timer2 module external clock input signal 101 ---- TIM5_CHA Timer6 module channel 1A signal 110 ---- LVD_OUT LVD module output signal 111 ---- TIM3_CH2B Timer3 module channel 2B signal

6.5.3.6 Port PA05 Function Configuration Register (PA05_SEL)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA05_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA05_sel	Port PA05 function selection. 000 ---- GPIO PA05 001 ---- SPI0_SCK SPI0 module clock signal 010 ---- TIM0_ETR Timer0 module external clock input signal 011 ---- PCA_ECI PCA module external clock input signal 100 ---- TIM0_CHA Timer0 module channel A signal 101 ---- TIM5_CHB Timer6 module channel 1B signal 110 ---- XTL_OUT external 32K crystal output signal 111 ---- XTH_OUT external 32M crystal output signal

6.5.3.7 Port PA06 Function Configuration Register (PA06_SEL)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA06_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA06_sel	Port PA06 function selection. 000 ---- GPIO PA06 001 ---- SPI0_MISO SPI0 module master input slave output data signal 010 ---- PCA_CH0 PCA module channel 0 capture/compare signal 011 ---- TIM3_BK Timer3 module brake signal 100 ---- TIM1_CHA Timer1 module channel A signal 101 ---- VC0_OUT VC0 module output/reverse output signal 110 ---- TIM3_GATE Timer3 module gate signal 111 ---- LPUART0_CTS LPUART0 module CTS signal

6.5.3.8 Port PA07 Function Configuration Register (PA07_SEL)

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA07_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA07_sel	Port PA07 function selection. 000 ---- GPIO PA07 001 ---- SPI0_MOSI SPI0 module master output slave input data signal 010 ---- PCA_CH1 PCA module channel 1 capture/compare signal 011 ---- HCLK_OUT AHB bus clock output signal 100 ---- TIM3_CH0B Timer3 module channel 0B signal 101 ---- TIM2_CHA Timer2 module channel A signal 110 ---- VC1_OUT VC1 module output/reverse output signal 111 ---- TIM4_CHB Timer6 module channel 0B signal

6.5.3.9 Port PA08 Function Configuration Register (PA08_SEL)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA08_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA08_sel	Port PA08 function selection. 000 ---- GPIO PA08 001 ---- UART0_TXD UART0 module TXD signal 010 ---- TIM3_CH0A Timer3 module channel 0A signal 011 ---- CRS_SYNC TBD 100 ---- CAN_STBY TBD 101 ---- TIM1_GATE Timer1 module gate signal 110 ---- TIM4_CHA Timer6 module channel 0A signal 111 ---- TIM3_BK Timer3 module brake signal

6.5.3.10 Port PA09 Function Configuration Register (PA09_SEL)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA09_sel rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA09_sel	Port PA09 function selection. 000 ---- GPIO PA09 001 ---- UART0_RXD UART0 module RXD signal 010 ---- TIM3_CH1A Timer3 module channel 1A signal 011 ---- TIM0_BK Timer0 module brake signal 100 ---- I2C0_SDA I2C0 module data signal 101 ---- Reserved 110 ---- HCLK_OUT AHB bus clock output signal 111 ---- TIM5_CHA Timer6 module channel 1A signal

6.5.3.11 Port PA10 Function Configuration Register (PA10_SEL)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA10_sel rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA10_sel	Port PA10 function selection. 000 ---- GPIO PA10 001 ---- UART0_RXD UART0 module RXD signal 010 ---- TIM3_CH2A Timer3 module channel 2A signal 011 ---- TIM2_BK Timer2 module brake signal 100 ---- I2C0_SDA I2C0 module data signal 101 ---- TIM2_GATE Timer2 module gate signal 110 ---- PCLK_OUT APB bus clock output signal 111 ---- TIM6_CHA Timer6 module channel 2A signal

6.5.3.12 Port PA11 Function Configuration Register (PA11_SEL)

Address offset: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA11_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA11_sel	Port PA11 function selection. 000 ---- GPIO PA11 001 ---- UART0_CTS UART0 module CTS signal 010 ---- TIM3_GATE Timer3 module gate signal 011 ---- I2C1_SCL I2C1 module clock signal 100 ---- CAN_RX CAN module receive signal 101 ---- VC0_OUT VC0 module output/reverse output signal 110 ---- SPI0_MISO SPI0 module master input slave output data signal 111 ---- TIM4_CHB Timer6 module channel 0B signal

6.5.3.13 Port PA12 Function Configuration Register (PA12_SEL)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA12_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA12_sel	Port PA12 function selection. 000 ---- GPIO PA12 001 ---- UART0_RTS UART0 module RTS signal 010 ---- TIM3_ETR Timer3 module external clock input signal 011 ---- I2C1_SDA I2C1 module data signal 100 ---- CAN_TX CAN module send signal 101 ---- VC1_OUT VC1 module output/inverse output signal 110 ---- SPI0_MOSI SPI0 module master output slave input data signal 111 ---- PCNT_S0 PCNT module input signal 0

6.5.3.14 Port PA13 Function Configuration Register (PA13_SEL)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA13_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA13_sel	Port PA13 function selection. 000 ---- GPIO PA13 001 ---- IR_OUT Infrared output signal 010 ---- UART0_RXD UART0 module RXD signal 011 ---- LVD_OUT LVD module output signal 100 ---- TIM3_ETR Timer3 module external clock input signal 101 ---- RTC_1HZ RTC module 1Hz output signal 110 ---- PCNT_S1 PCNT module input signal 1 111 ---- VC2_OUT VC2 module output/reverse output signal

6.5.3.15 Port PA14 Function Configuration Register (PA14_SEL)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA14_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA14_sel	Port PA14 function selection. 000 ---- GPIO PA14 001 ---- UART1_TXD UART1 module TXD signal 010 ---- UART0_TXD UART0 module TXD signal 011 ---- TIM3_CH2A Timer3 module channel 2A signal 100 ---- LVD_OUT LVD module output signal 101 ---- RCH_OUT Internal 24M RC clock output signal 110 ---- RCL_OUT Internal 38K RC clock output signal 111 ---- PLL_OUT Internal PLL clock output signal

6.5.3.16 Port PA15 Function Configuration Register (PA15_SEL)

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PA15_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PA15_sel	Port PA15 function selection. 000 ---- GPIO PA15 001 ---- SPI0_CS SPI0 module master mode chip select signal 010 ---- UART1_RXD UART1 module RXD signal 011 ---- LPUART1_RTS LPUART1 module RTS signal 100 ---- TIM0_ETR Timer0 module external clock input signal 101 ---- TIM0_CHA Timer0 module channel A signal 110 ---- TIM3_CH1A Timer3 module channel 1A signal 111 ---- Reserved

6.5.4 Port PB function selection registers

6.5.4.1 Port PB00 Function Configuration Register (PB00_SEL)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB00_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB00_sel	Port PB00 function selection. 000 ---- GPIO PB00 001 ---- PCA_CH2 PCA module channel 2 capture/compare signal 010 ---- TIM3_CH1B Timer3 module channel 1B signal 011 ---- LPUART0_TXD LPUART0 module TXD signal 100 ---- TIM5_CHB Timer6 module channel 1B signal 101 ---- RCH_OUT Internal 24M RC clock output signal 110 ---- RCL_OUT Internal 38K RC clock output signal 111 ---- PLL_OUT Internal PLL clock output signal

6.5.4.2 Port PB01 Function Configuration Register (PB01_SEL)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PB01_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB01_sel	Port PB01 function selection. 000 ---- GPIO PB01 001 ---- PCA_CH3 PCA module channel 3 capture/compare signal 010 ---- PCLK_OUT APB bus clock output signal 011 ---- TIM3_CH2B Timer3 module channel 2B signal 100 ---- TIM6_CHB Timer6 module channel 2B signal 101 ---- LPUART0_RTS LPUART0 module RTS signal 110 ---- VC2_OUT VC2 module output/inverse output signal 111 ---- TCLK_OUT internal clock output signal

6.5.4.3 Port PB02 Function Configuration Register (PB02_SEL)

Address offset: 0x48

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PB02_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB02_sel	Port PB02 function selection. 000 ---- GPIO PB02 001 ---- LPTIM0_TOG Low power timer module flip signal 010 ---- PCA_ECI PCA module external clock input signal 011 ---- LPUART1_RXD LPUART1 module RXD signal 100 ---- TIM4_CHA Timer6 module channel 0A signal 101 ---- TIM1_BK Timer1 module brake signal 110 ---- TIM0_BK Timer0 module brake signal 111 ---- TIM2_BK Timer2 module brake signal

6.5.4.4 Port PB03 Function Configuration Register (PB03_SEL)

Address offset: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB03_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB03_sel	Port PB03 function selection. 000 ---- GPIO PB03 001 ---- SPI0_SCK SPI0 module clock signal 010 ---- TIM0_CHB Timer0 module channel B signal 011 ---- TIM1_GATE Timer1 module gate signal 100 ---- TIM3_CH0A Timer3 module channel 0A signal 101 ---- LPTIM0_GATE low power timer module gate signal 110 ---- XTL_OUT external 32K crystal output signal 1111 ---- XTH_OUT external 32M crystal output signal

6.5.4.5 Port PB04 Function Configuration Register (PB04_SEL)

Address offset: 0x50

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB04_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB04_sel	Port PB04 function selection. 000 ---- GPIO PB04 001 ---- SPI0_MISO SPI0 module master input slave output data signal 010 ---- PCA_CH0 PCA module channel 0 capture/compare signal 011 ---- TIM2_BK Timer2 module brake signal 100 ---- UART0_CTS UART0 module CTS signal 101 ---- TIM2_GATE Timer2 module gate signal 110 ---- TIM3_CH0B Timer3 module channel 0B signal 111 ---- LPTIM0_EXT low power timer module external clock input signal

6.5.4.6 Port PB05 Function Configuration Register (PB05_SEL)

Address offset: 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB05_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB05_sel	Port PB05 function selection. 000 ---- GPIO PB05 001 ---- SPI0_MOSI SPI0 module master output slave input data signal 010 ---- Reserved 011 ---- TIM1_BK Timer1 module brake signal 100 ---- PCA_CH1 PCA module channel 1 capture/compare signal 101 ---- LPTIM0_GATE low power timer module gate signal 110 ---- PCNT_S0 PCNT module input signal 0 111 ---- UART0_RTS UART0 module RTS signal

6.5.4.7 Port PB06 Function Configuration Register (PB06_SEL)

Address offset: 0x58

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB06_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB06_sel	Port PB06 function selection. 000 ---- GPIO PB06 001 ---- I2C0_SCL I2C0 module clock signal 010 ---- UART0_TXD UART0 module TXD signal 011 ---- TIM1_CHB Timer1 module channel B signal 100 ---- TIM0_CHA Timer0 module channel A signal 101 ---- LPTIM0_EXT Low power timer module external clock input signal 110 ---- TIM3_CH0A Timer3 module channel 0A signal 111 ---- LPTIM0_TOG Low power timer module flip signal

6.5.4.8 Port PB07 Function Configuration Register (PB07_SEL)

Address offset: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB07_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB07_sel	Port PB07 function selection. 000 ---- GPIO PB07 001 ---- I2C0_SDA I2C0 module data signal 010 ---- UART0_RXD UART0 module RXD signal 011 ---- TIM2_CHB Timer2 module channel B signal 100 ---- LPUART1_CTS LPUART1 module CTS signal 101 ---- TIM0_CHB Timer0 module channel B signal 110 ---- LPTIM0_TOGN low power timer module flip reverse signal 111 ---- PCNT_S1 PCNT module input signal 1

6.5.4.9 Port PB08 Function Configuration Register (PB08_SEL)

Address offset: 0x60

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB08_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB08_sel	Port PB08 function selection. 000 ---- GPIO PB08 001 ---- I2C0_SCL I2C0 module clock signal 010 ---- TIM1_CHA Timer1 module channel A signal 011 ---- CAN_RX CAN module receive signal 100 ---- TIM2_CHA Timer2 module channel A signal 101 ---- TIM0_GATE Timer0 module gate signal 110 ---- TIM3_CH2A Timer3 module channel 2A signal 111 ---- UART0_TXD UART0 module TXD signal

6.5.4.10 Port PB09 Function Configuration Register (PB09_SEL)

Address offset: 0x64

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB09_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB09_sel	Port PB09 function selection. 000 ---- GPIO PB09 001 ---- I2C0_SDA I2C0 module data signal 010 ---- IR_OUT infrared output signal 011 ---- SPI1_CS SPI1 module master mode chip select signal 100 ---- TIM2_CHA Timer2 module channel A signal 101 ---- CAN_TX CAN module transmit signal 110 ---- TIM2_CHB Timer2 module channel B signal 111 ---- UART0_RXD UART0 module RXD signal

6.5.4.11 Port PB10 Function Configuration Register (PB10_SEL)

Address offset: 0x68

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB10_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB10_sel	Port PB10 function selection. 000 ---- GPIO PB10 001 ---- I2C1_SCL I2C1 module clock signal 010 ---- SPI1_SCK SPI1 module clock signal 011 ---- TIM1_CHA Timer1 module channel A signal 100 ---- LPUART0_TXD LPUART0 module TXD signal 101 ---- TIM3_CH1A Timer3 module channel 1A signal 110 ---- LPUART1_RTS LPUART1 module RTS signal 111 ---- UART1_RTS UART1 module RTS signal

6.5.4.12 Port PB11 Function Configuration Register (PB11_SEL)

Address offset: 0x6C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PB11_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB11_sel	Port PB11 function selection. 000 ---- GPIO PB11 001 ---- I2C1_SDA I2C1 module data signal 010 ---- TIM1_CHB Timer1 module channel B signal 011 ---- LPUART0_RXD LPUART0 module RXD signal 100 ---- TIM2_GATE Timer2 module gate signal 101 ---- TIM6_CHA Timer6 module channel 2A signal 110 ---- LPUART1_CTS LPUART1 module CTS signal 111 ---- UART1_CTS UART1 module CTS signal

6.5.4.13 Port PB12 Function Configuration Register (PB12_SEL)

Address offset: 0x70

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PB12_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB12_sel	Port PB12 function selection. 000 ---- GPIO PB12 001 ---- SPI1_CS SPI1 module master mode chip select signal 010 ---- TIM3_BK Timer3 module brake signal 011 ---- LPUART0_TXD LPUART0 module TXD signal 100 ---- TIM0_BK Timer0 module brake signal 101 ---- Reserved 110 ---- LPUART0_RTS LPUART0 module RTS signal 111 ---- TIM6_CHA Timer6 module channel 2A signal

6.5.4.14 Port PB13 Function Configuration Register (PB13_SEL)

Address offset: 0x74

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB13_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB13_sel	Port PB13 function selection. 000 ---- GPIO PB13 001 ---- SPI1_SCK SPI1 module clock signal 010 ---- I2C1_SCL I2C1 module clock signal 011 ---- TIM3_CH0B Timer3 module channel 0B signal 100 ---- LPUART0_CTS LPUART0 module CTS signal 101 ---- TIM1_CHA Timer1 module channel A signal 110 ---- TIM1_GATE Timer1 module gate signal 111 ---- TIM6_CHB Timer6 module channel 2B signal

6.5.4.15 Port PB14 Function Configuration Register (PB14_SEL)

Address offset: 0x78

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB14_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB14_sel	Port PB14 function selection. 000 ---- GPIO PB14 001 ---- SPI1_MISO SPI1 module master input slave output data signal 010 ---- I2C1_SDA I2C1 module data signal 011 ---- TIM3_CH1B Timer3 module channel 1B Signal 100 ---- TIM0_CHA Timer0 module channel A signal 101 ---- RTC_1HZ RTC module 1Hz output signal 110 ---- LPUART0_RTS LPUART0 module RTS signal 111 ---- TIM1_BK Timer1 module brake signal

6.5.4.16 Port PB15 Function Configuration Register (PB15_SEL)

Address offset: 0x7C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16							
									Reserved													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
									Reserved													
															PB15_sel							
																rw						

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PB15_sel	Port PB15 function selection. 000 ---- GPIO PB15 001 ---- SPI1_MOSI SPI1 module master output slave input data signal 010 ---- TIM3_CH2B Timer3 module channel 2B signal 011 ---- TIM0_CHB Timer0 module channel B signal 100 ---- TIM0_GATE Timer0 module gate signal 101 ---- Reserved 110 ---- Reserved 111 ---- LPUART1_RXD LPUART1 module RXD signal

6.5.5 Port PC function selection registers

6.5.5.1 Port PC00 Function Configuration Register (PC00_SEL)

Address offset: 0x80

Reset value: 0x0000 0000

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

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC00_sel	Port PC00 function selection. 000 ---- GPIO PC00 001 ---- LPTIM0_GATE Low power timer module gate signal 010 ---- PCNT_S0 PCNT module input signal 0 011 ---- UART1_CTS UART1 module CTS signal 100 ---- UART2_RTS UART2 module RTS signal 101 ---- I2S0_MCK I2S0 module MCK signal 110 ---- Reserved 111 ---- Reserved

6.5.5.2 Port PC01 Function Configuration Register (PC01_SEL)

Address offset: 0x84

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC01_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC01_sel	Port PC01 function selection.: 000 ---- GPIO PC01 001 ---- LPTIM0_TOG Low power timer module flip signal 010 ---- TIM5_CHB Timer6 module channel 1B signal 011 ---- UART1_RTS UART1 module RTS signal 100 ---- PCNT_S0FO PCNT module input signal 0 filter output 101 ---- I2S0_SD I2S0 module SD signal 110 ---- UART2_CTS UART2 module CTS signal 111 ---- Reserved

6.5.5.3 Port PC02 Function Configuration Register (PC02_SEL)

Address offset: 0x88

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC02_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC02_sel	Port PC02 function selection. 000 ---- GPIO PC02 001 ---- SPI1_MISO SPI1 module master input slave output data signal 010 ---- LPTIM0_TOGN Low power timer module flip reverse signal 011 ---- PCNT_S1 PCNT module input signal 1 100 ---- UART2_RXD UART2 module RXD signal 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.5.4 Port PC03 Function Configuration Register (PC03_SEL)

Address offset: 0x8C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC03_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC03_sel	Port PC03 function selection. 000 ---- GPIO PC03 001 ---- SPI1_MOSI SPI1 module master output slave input data signal 010 ---- LPTIMO_EXT Low power timer module external clock input signal 011 ---- LPTIMO_TOGN Low power timer module flip reverse signal 100 ---- PCNT_S1FO PCNT module input signal 1 filter output 101 ---- UART2_TXD UART2 module TXD signal 110 ---- Reserved 111 ---- Reserved

6.5.5.5 Port PC04 Function Configuration Register (PC04_SEL)

Address offset: 0x90

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC04_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC04_sel	Port PC04 function selection. 000 ---- GPIO PC04 001 ---- LPUART0_RXD LPUART0 module RXD signal 010 ---- TIM2_ETR Timer2 module external clock input signal 011 ---- IR_OUT Infrared output signal 100 ---- VC2_OUT VC2 module output/reverse output signal 101 ---- I2S0_WS I2S0 module WS signal 110 ---- Reserved 111 ---- Reserved

6.5.5.6 Port PC05 Function Configuration Register (PC05_SEL)

Address offset: 0x94

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC05_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC05_sel	Port PC05 function selection. 000 ---- GPIO PC05 001 ---- LPUART0_RXD LPUART0 module RXD signal 010 ---- TIM6_CHB Timer6 module channel 2B signal 011 ---- PCA_CH4 PCA module channel 4 capture/compare signal 100 ---- Reserved 101 ---- I2S0_SDIN I2S0 module SDIN signal 110 ---- Reserved 111 ---- Reserved

6.5.5.7 Port PC06 Function Configuration Register (PC06_SEL)

Address offset: 0x98

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC06_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC06_sel	Port PC06 function selection. 000 ---- GPIO PC06 001 ---- PCA_CH0 PCA module channel 0 capture/compare signal 010 ---- TIM4_CHA Timer6 module channel 0A signal 011 ---- TIM2_CHA Timer2 module channel A signal 100 ---- LPTIM1_GATE Low power timer1 module gate signal 101 ---- I2S1_SCK I2S1 module SCK signal 110 ---- UART3_RXD UART3 module RXD signal 111 ---- Reserved

6.5.5.8 Port PC07 Function Configuration Register (PC07_SEL)

Address offset: 0x9C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC07_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC07_sel	Port PC07 function selection. 000 ---- GPIO PC07 001 ---- PCA_CH1 PCA module channel 1 capture/compare signal 010 ---- TIM5_CHA Timer6 module channel 1A signal 011 ---- TIM2_CHB Timer2 module channel B signal 100 ---- LPTIM1_EXT Low power timer1 module external clock input signal 101 ---- I2S1_MCK I2S1 module MCK signal 110 ---- UART3_TXD UART3 module TXD signal 111 ---- Reserved

6.5.5.9 Port PC08 Function Configuration Register (PC08_SEL)

Address offset: 0xA0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC08_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC08_sel	Port PC08 function selection. 000 ---- GPIO PC08 001 ---- PCA_CH2 PCA module channel 2 capture/compare signal 010 ---- TIM6_CHA Timer6 module channel 2A signal 011 ---- TIM2_ETR Timer2 module external clock input signal 100 ---- LPTIM1_TOG Low power timer1 module flip signal 101 ---- I2S1_SD I2S1 module SD signal 110 ---- UART3_CTS UART3 module CTS signal 111 ---- Reserved

6.5.5.10 Port PC09 Function Configuration Register (PC09_SEL)

Address offset: 0xA4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PC09_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC09_sel	Port PC09 function selection. 000 ---- GPIO PC09 001 ---- PCA_CH3 PCA module channel 3 capture/compare signal 010 ---- TIM4_CHB Timer6 module channel 0B signal 011 ---- TIM1_ETR Timer1 module external clock input signal 100 ---- LPTIM1_TOGN Low power timer1 module flip reverse signal 101 ---- I2S1_WS I2S1 module WS signal 110 ---- UART3_RTS UART3 module RTS signal 111 ---- Reserved

6.5.5.11 Port PC10 Function Configuration Register (PC10_SEL)

Address offset: 0xA8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PC10_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC10_sel	Port PC10 function selection. 000 ---- GPIO PC10 001 ---- LPUART1_TXD LPUART1 module TXD signal 010 ---- LPUART0_TXD LPUART0 module TXD signal 011 ---- PCA_CH2 PCA module channel 2 capture/compare signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.5.12 Port PC11 Function Configuration Register (PC11_SEL)

Address offset: 0xAC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC11_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC11_sel	Port PC11 function selection. 000 ---- GPIO PC11 001 ---- LPUART1_RXD LPUART1 module RXD signal 010 ---- LPUART0_RXD LPUART0 module RXD signal 011 ---- PCA_CH3 PCA module channel 3 capture/compare signal 100 ---- PCNT_S0FO PCNT module input signal 0 filter output 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.5.13 Port PC12 Function Configuration Register (PC12_SEL)

Address offset: 0xB0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC12_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC12_sel	Port PC12 function selection. 000 ---- GPIO PC12 001 ---- LPUART0_RXD LPUART0 module RXD signal 010 ---- LPUART1_RXD LPUART1 module RXD signal 011 ---- PCA_CH4 PCA module channel 4 capture/compare signal 100 ---- PCNT_S1FO PCNT module input signal 1 filter output 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.5.14 Port PC13 Function Configuration Register (PC13_SEL)

Address offset: 0xB4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PC13_sel rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC13_sel	Port PC13 function selection. 000 ---- GPIO PC13 001 ---- Reserved 010 ---- RTC_1HZ RTC module 1Hz output signal 011 ---- TIM3_CH1B Timer3 module channel 1B signal 100 ---- Reserved 101 ---- I2S0_SCK I2S0 module SCK signal 110 ---- Reserved 111 ---- Reserved

6.5.5.15 Port PC14 Function Configuration Register (PC14_SEL)

Address offset: 0xB8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PC14_sel rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC14_sel	Port PC14 function selection. 000 ---- GPIO PC14 001 ---- Reserved 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved Reserved 111 ---- Reserved

6.5.5.16 Port PC15 Function Configuration Register (PC15_SEL)

Address offset: 0xBC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC15_sel rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PC15_sel	Port PC15 function selection. 000 ---- GPIO PC15 001 ---- Reserved 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

Port PD function selection registers

6.5.6.1 Port PD00 Function Configuration Register (PD00_SEL)

Address offset: 0xC0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD00_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD00_sel	Port PD00 function selection. 000 ---- GPIO PD00 001 ---- CAN_RX CAN module receive signal 010 ---- SPI1_CS SPI1 module master mode chip select signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.2 Port PD01 Function Configuration Register (PD01_SEL)

Address offset: 0xC4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PD01_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD01_sel	Port PD01 function selection. 000 ---- GPIO PD01 001 ---- CAN_TX CAN module transmit signal 010 ---- SPI1_SCK SPI1 module clock signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.3 Port PD02 Function Configuration Register (PD02_SEL)

Address offset: 0xC8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PD02_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD02_sel	Port PD02 function selection. 000 ---- GPIO PD02 001 ---- PCA_ECI PCA module external clock input signal 010 ---- LPUART0_RTS LPUART0 module RTS signal 011 ---- TIM1_ETR Timer1 module external clock input signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.4 Port PD03 Function Configuration Register (PD03_SEL)

Address offset: 0xCC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD03_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD03_sel	Port PD03 function selection. 000 ---- GPIO PD03 001 ---- UART1_CTS UART1 module CTS signal 010 ---- SPI1_MISO SPI1 module master input slave output data signal 011 ---- LPTIM1_TOG low power timer1 module flip signal 100 ---- I2S1_SCK I2S1 module SCK signal 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.5 Port PD04 Function Configuration Register (PD04_SEL)

Address offset: 0xD0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD04_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD04_sel	Port PD04 function selection. 000 ---- GPIO PD04 001 ---- UART1_RTS UART1 module RTS signal 010 ---- SPI1_MOSI SPI1 module master output slave input data signal 011 ---- LPTIM1_TOGN low power timer1 module flip reverse signal 100 ---- I2S1_MCK I2S1 module MCK signal 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.6 Port PD05 Function Configuration Register (PD05_SEL)

Address offset: 0xD4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PD05_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD05_sel	Port PD05 function selection. 000 ---- GPIO PD05 001 ---- UART1_RXD UART1 module RXD signal 010 ---- LPTIM1_EXT low power timer1 module external clock input signal 011 ---- CAN_STBY CAN module STBY signal 100 ---- I2S1_WS I2S1 module WS signal 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.7 Port PD06 Function Configuration Register (PD06_SEL)

Address offset: 0xD8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
															PD06_sel
															rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD06_sel	Port PD06 function selection. 000 ---- GPIO PD06 001 ---- UART1_RXD UART1 module RXD signal 010 ---- LPTIM1_EXT low power timer1 module external clock input signal 011 ---- Reserved 100 ---- I2S1_WS I2S1 module WS signal 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.8 Port PD07 Function Configuration Register (PD07_SEL)

Address offset: 0xDC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD07_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD07_sel	Port PD07 function selection. 000 ---- GPIO PD07 001 ---- UART1_TXD UART1 module TXD signal 010 ---- Reserved 011 ---- Reserved 100 ---- I2S1_SDIN I2S1 module SDIN signal 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.9 Port PD08 Function Configuration Register (PD08_SEL)

Address offset: 0xE0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD08_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD08_sel	Port PD08 function selection. 000 ---- GPIO PD08 001 ---- LPUART0_TXD LPUART0 module TXD signal 010 ---- I2S0_SCK I2S0 module SCK signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.10 Port PD09 Function Configuration Register (PD09_SEL)

Address offset: 0xE4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD09_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD09_sel	Port PD09 function selection. 000 ---- GPIO PD09 001 ---- LPUART0_RXD LPUART0 module RXD signal 010 ---- I2S0_MCK I2S0 module MCK signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.11 Port PD10 Function Configuration Register (PD10_SEL)

Address offset: 0xE8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD10_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD10_sel	Port PD10 function selection. 000 ---- GPIO PD10 001 ---- LPUART0_TXD LPUART0 module TXD signal 010 ---- I2S0_SD I2S0 module SD signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.12 Port PD11 Function Configuration Register (PD11_SEL)

Address offset: 0xEC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD11_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD11_sel	Port PD11 function selection. 000 ---- GPIO PD11 001 ---- LPUART0_CTS LPUART0 module CTS signal 010 ---- I2S0_WS I2S0 module WS signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.13 Port PD12 Function Configuration Register (PD12_SEL)

Address offset: 0xF0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD12_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD12_sel	Port PD12 function selection. 000 ---- GPIO PD12 001 ---- LPUART0_CTS LPUART0 module CTS signal 010 ---- I2S0_WS I2S0 module WS signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.14 Port PD13 Function Configuration Register (PD13_SEL)

Address offset: 0xF4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD13_sel	rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD13_sel	Port PD13 function selection. 000 ---- GPIO PD13 001 ---- UART2_RXD UART2 module RXD signal 010 ---- I2S0_SDIN I2S0 module SDIN signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.15 Port PD14 Function Configuration Register (PD14_SEL)

Address offset: 0xF8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD14_sel	rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD14_sel	Port PD14 function selection. 000 ---- GPIO PD14 001 ---- UART2_TXD UART2 module TXD signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.6.16 Port PD15 Function Configuration Register (PD15_SEL)

Address offset: 0xFC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PD15_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PD15_sel	Port PD15 function selection. 000 ---- GPIO PD15 001 ---- CRS_SYNC 010 ---- UART2_CTS UART2 module CTS signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7 Port PE function selection registers

6.5.7.1 Port PE00 Function Configuration Register (PE00_SEL)

Address offset: 0x1000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE00_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE00_sel	Port PE00 function selection. 000 ---- GPIO PE00 001 ---- TIM1_CHA Timer1 module channel A signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.2 Port PE01 Function Configuration Register (PE01_SEL)

Address offset: 0x1004

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE01_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE01_sel	Port PE01 function selection. 000 ---- GPIO PE01 001 ---- TIM2_CHA Timer2 module channel A signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.3 Port PE02 Function Configuration Register (PE02_SEL)

Address offset: 0x1008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE02_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE02_sel	Port PE02 function selection. 000 ---- GPIO PE02 001 ---- PCA_ECI PCA module external clock input signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.4 Port PE03 Function Configuration Register (PE03_SEL)

Address offset: 0x100C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE03_sel	rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE03_sel	Port PE03 function selection. 000 ---- GPIO PE03 001 ---- PCA_CH0 PCA module channel 0 capture/compare signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.5 Port PE04 Function Configuration Register (PE04_SEL)

Address offset: 0x1010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE04_sel	rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE04_sel	Port PE04 function selection. 000 ---- GPIO PE04 001 ---- PCA_CH1 PCA module channel 1 capture/compare signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.6 Port PE05 Function Configuration Register (PE05_SEL)

Address offset: 0x1014

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE05_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE05_sel	Port PE05 function selection. 000 ---- GPIO PE05 001 ---- PCA_CH2 PCA module channel 2 capture/compare signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.7 Port PE06 Function Configuration Register (PE06_SEL)

Address offset: 0x1018

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE06_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE06_sel	Port PE06 function selection. 000 ---- GPIO PE06 001 ---- PCA_CH3 PCA module channel 3 capture/compare signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.8 Port PE07 Function Configuration Register (PE07_SEL)

Address offset: 0x101C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE07_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE07_sel	Port PE07 function selection. 000 ---- GPIO PE07 001 ---- TIM3_ETR Timer3 module external clock input signal 010 ---- LPTIM1_GATE Low power timer1 module gate signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.9 Port PE08 Function Configuration Register (PE08_SEL)

Address offset: 0x1020

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE08_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE08_sel	Port PE08 function selection. 000 ---- GPIO PE08 001 ---- TIM3_CH0B Timer3 module channel 0B signal LPTIM1_GATE Low power timer1 module gate signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.10 Port PE09 Function Configuration Register (PE09_SEL)

Address offset: 0x1024

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE09_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE09_sel	Port PE09 function selection. 000 ---- GPIO PE09 001 ---- TIM3_CH0A Timer3 module channel 0A signal 010 ---- LPTIM1_TOG Low power timer1 module flip signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.11 Port PE10 Function Configuration Register (PE10_SEL)

Address offset: 0x1028

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE10_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE10_sel	Port PE10 function selection. 000 ---- GPIO PE10 001 ---- TIM3_CH1B Timer3 module channel 1B signal 010 ---- LPTIM1_TOGN Low power timer1 module flip reverse signal 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.12 Port PE11 Function Configuration Register (PE11_SEL)

Address offset: 0x102C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE11_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE11_sel	Port PE11 function selection. 000 ---- GPIO PE11 001 ---- TIM3_CH1A Timer3 module channel 1A signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.13 Port PE12 Function Configuration Register (PE12_SEL)

Address offset: 0x1030

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE12_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE12_sel	Port PE12 function selection. 000 ---- GPIO PE12 001 ---- TIM3_CH2B Timer3 module channel 2B signal 010 ---- SPI0_CS SPI0 module host mode chip select signal 011 ---- UART3_CTS UART3 module CTS signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.14 Port PE13 Function Configuration Register (PE13_SEL)

Address offset: 0x1034

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE13_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE13_sel	Port PE13 function selection. 000 ---- GPIO PE13 001 ---- TIM3_CH2A Timer3 module channel 2A signal 010 ---- SPI0_SCK SPI0 module clock signal 011 ---- UART3_RTS UART3 module RTS signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.15 Port PE14 Function Configuration Register (PE14_SEL)

Address offset: 0x1038

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE14_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE14_sel	Port PE14 function selection. 000 ---- GPIO PE14 001 ---- TIM3_CH0B Timer3 module channel 0B signal 010 ---- SPI0_MISO SPI0 module master input slave output data signal 011 ---- UART3_RXD UART3 module RXD signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.7.16 Port PE15 Function Configuration Register (PE15_SEL)

Address offset: 0x103C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PE15_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PE15_sel	Port PE15 function selection. 000 ---- GPIO PE15 001 ---- TIM3_BK Timer3 module brake signal 010 ---- SPI0_MOSI SPI0 module master output slave input data signal 011 ---- UART3_TXD UART3 module TXD signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8 Port PF function selection registers

6.5.8.1 Port PF00 Function Configuration Register (PF00_SEL)

Address offset: 0x1040

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PF00_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF00_sel	Port PF00 function selection. 000 ---- GPIO PF00 001 ---- I2C0_SDA I2C0 module data signal 010 ---- CRS_SYNC 011 ---- UART1_TXD UART1 module TXD signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.2 Port PF01 Function Configuration Register (PF01_SEL)

Address offset: 0x1044

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PF01_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF01_sel	Port PF01 function selection. 000 ---- GPIO PF01 001 ---- I2C0_SCL I2C0 module clock signal 010 ---- TIM4_CHB Timer6 module channel 0B signal 011 ---- UART1_RXD UART1 module RXD signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.3 Port PF02 Function Configuration Register (PF02_SEL)

Address offset: 0x1048

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PF02_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF02_sel	Port PF02 function selection. 000 ---- GPIO PF02 001 ---- Reserved 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.4 Port PF03 Function Configuration Register (PF03_SEL)

Address offset: 0x104C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PF03_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF03_sel	Port PF03 function selection. 000 ---- GPIO PF03 001 ---- Reserved 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.5 Port PF04 Function Configuration Register (PF04_SEL)

Address offset: 0x1050

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
PF04_sel rw															

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF04_sel	Port PF04 function selection. 000 ---- GPIO PF04 001 ---- Reserved 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.6 Port PF05 Function Configuration Register (PF05_SEL)

Address offset: 0x1054

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
PF05_sel rw															

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF05_sel	Port PF05 function selection. 000 ---- GPIO PF05 001 ---- Reserved 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.7 Port PF06 Function Configuration Register (PF06_SEL)

Address offset: 0x1058

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PF06_sel	rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF06_sel	Port PF06 function selection. 000 ---- GPIO PF06 001 ---- I2C1_SCL I2C1 module clock signal 010 ---- LPUART1_CTS LPUART1 module CTS signal 011 ---- UART0_CTS UART0 module CTS signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.8 Port PF07 Function Configuration Register (PF07_SEL)

Address offset: 0x105C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PF07_sel	rw

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF07_sel	Port PF07 function selection. 000 ---- GPIO PF07 001 ---- I2C1_SDA I2C1 module data signal 010 ---- LPUART1_RTS LPUART1 module RTS signal 011 ---- UART0_RTS UART0 module RTS signal 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.9 Port PF09 Function Configuration Register (PF09_SEL)

Address offset: 0x1064

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PF09_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF09_sel	Port PF09 function selection. 000 ---- GPIO PF09 001 ---- TIM0_CHA Timer0 module channel A signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.10 Port PF10 Function Configuration Register (PF10_SEL)

Address offset: 0x1068

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PF10_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF10_sel	Port PF10 function selection. 000 ---- GPIO PF10 001 ---- TIM0_CHB Timer0 module channel B signal 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

6.5.8.11 Port PF11 Function Configuration Register (PF11_SEL)

Address offset: 0x106C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PF11_sel	
														rw	

Bit	Mark	Function Description
31:3	Reserved	Reserved
2:0	PF11_sel	Port PF11 function selection. 000 ---- GPIO PF11 001 ---- Reserved 010 ---- Reserved 011 ---- Reserved 100 ---- Reserved 101 ---- Reserved 110 ---- Reserved 111 ---- Reserved

7 I2C bus (I2C)

7.1 Introduction

I2C is a two-wire bidirectional synchronous serial bus. It uses a clock line and a data line to transfer information between two devices connected to the bus, providing a simple and efficient method for data exchange between devices. Each device connected to the bus has a unique address. Any device can be either a host or a slave, but only one host is allowed at the same time. The I2C standard is a true multi-host bus with conflict detection and arbitration mechanisms. It can use arbitration mechanisms to avoid data conflicts and protect data when multiple hosts request control of the bus at the same time.

I2C bus controllers can meet various specifications of the I2C bus and support all transmission modes that communicate with the I2C bus. The I2C logic can handle the transmission of bytes autonomously. It can keep track of serial transmissions, and there is also a status register (I2Cx_STAT) that reflects the status of the I2C bus controller and the I2C bus.

7.2 Main Features

The I2C controller supports the following features:

- Supports four working modes: master transmit/receive, slave transmit/receive
- Supports three working rates: standard (100Kbps) / fast (400Kbps) / high speed (1Mbps)
- Supports 7-bit addressing function
- Supports noise filtering function
- Supports broadcast address
- Supports interrupt status query function

7.3 Protocol Description

The I2C bus uses "SCL" (serial clock bus) and "SDA" (serial data bus) to transmit information. The host outputs the serial clock signal on the SCL line, and the data is transmitted on the SDA line. Each byte is transmitted (MSB starts the transmission), followed by an acknowledge bit. One SCL clock pulse transmits one data bit.

7.3.1 Data transmission on I2C bus

Usually, the standard I2C transmission protocol contains four parts: start (S) or repeated start signal (Sr), slave address and read/write bit, transmission data, and stop signal (P).

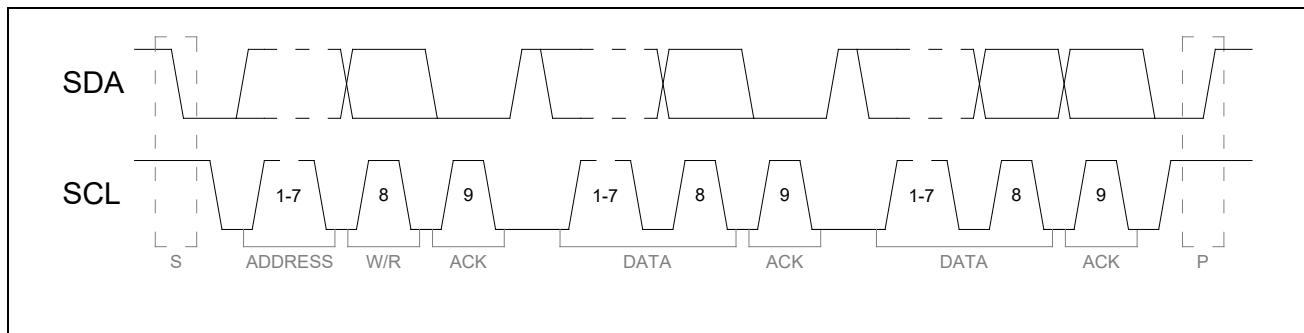


Figure 7-1 I2C transmission protocol

- Start signal, repeated start signal, stop signal
 - When the bus is in idle state (SCL and SDA lines are high at the same time), a signal from high to low appears on the SDA line, indicating that a start signal has been generated on the bus.
 - When there is no stop signal between two start signals, a repeated start signal is generated. The host uses this method to communicate with another slave or the same slave in a different transmission direction (for example: from writing to a device to reading from a device) without releasing the bus.
 - When the SCL line is high, a signal from low to high appears on the SDA line, which is defined as a stop signal. The host sends a stop signal to the bus to end data transmission.

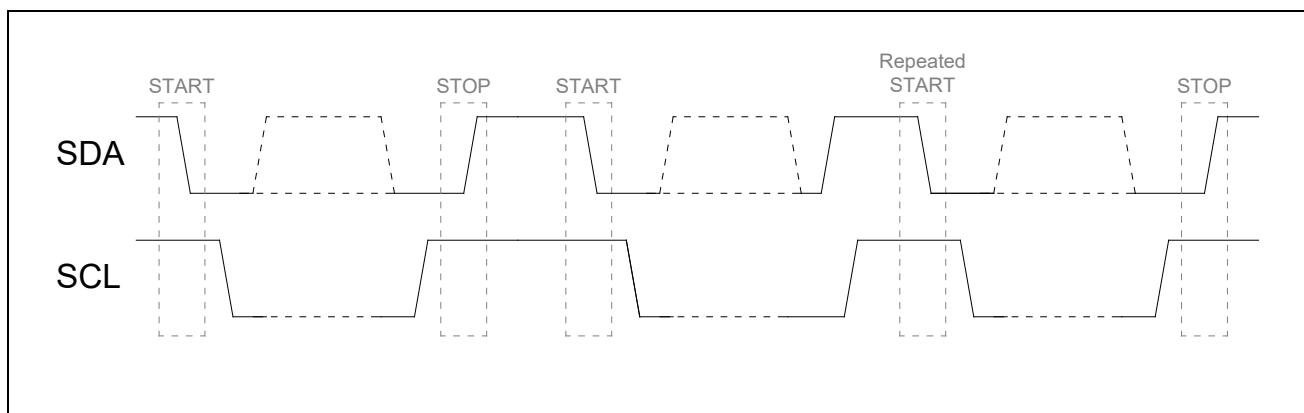


Figure 7-2 START and STOP conditions

- Slave address and read/write bit
 - When the start signal is generated, the host immediately transmits the first byte of data: 7-bit slave address + read/write bit, the read/write bit controls the data transmission direction of the slave (0: write; 1: read). The slave addressed by the host will respond by setting SDA to low level in the 9th SCL clock cycle.
- Transmitting data
 - During data transmission, one SCL clock pulse transmits one data bit, and the SDA line can only change when SCL is low.

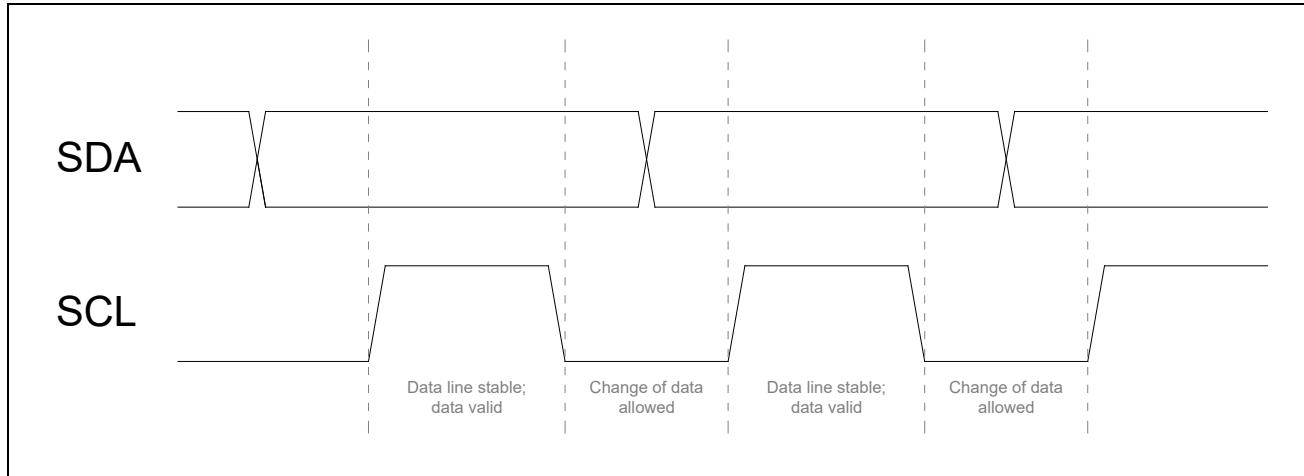


Figure 7-3 I2C bus upper transmission

7.3.2 Acknowledgement on the I2C bus

Each byte is followed by an acknowledge bit. By pulling the SDA line low, the receiver is allowed to respond to the transmitter. ACK is a low-level signal. When the clock signal is high, SDA remains low, indicating that the receiver has successfully received the data from the transmitter.

When the host is used as a transmitting device, if a no response signal (NACK) is generated on the slave, the host can generate a stop signal to exit the data transmission, or generate a repeated start signal to start a new round of data transmission. When the host is used as a receiving device, a no response signal (NACK) occurs, and the slave releases the SDA line, causing the host to generate a stop signal or a repeated start signal.

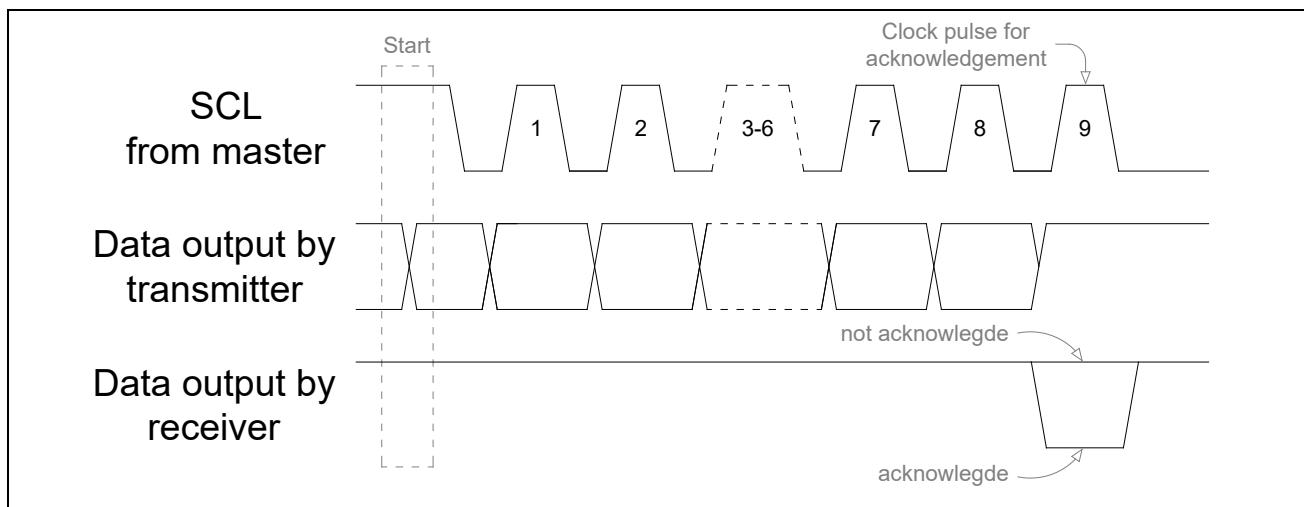


图 7-4 I2C 总线上应答信号

7.3.3 Arbitration on the I2C bus

Arbitration on the I2C bus is divided into two parts: synchronization on the SCL line and arbitration on the SDA line

- **Synchronization on the SCL line (clock synchronization)**

- Since the I2C bus has the logic function of line "AND", as long as one node on the SCL line sends a low level, the bus will show a low level. When all nodes send a high level, the bus can show a high level. Therefore, the time of the clock low level is determined by the device with the longest clock level period, and the time of the clock high level is determined by the device with the shortest clock high level period. Due to this characteristic of I2C, when multiple hosts send clock signals at the same time, a unified clock signal is represented on the bus. If the slave wants the host to reduce the transmission speed, it can actively pull SCL low to extend its low level time to notify the host. When the host finds that the SCL level is pulled low when preparing for the next transmission, it waits until the slave completes the operation and releases the control of the SCL line.

- **Arbitration on the SDA line**

- Arbitration on the SDA line is also due to the fact that the I2C bus has the logic function of line "AND". After sending data, the host decides whether to withdraw from the competition by comparing the data on the bus. The host that loses arbitration immediately switches to the unaddressed slave state to ensure that it can be addressed by the host that wins arbitration. The host that fails arbitration continues to output clock pulses (on SCL) until the current serial byte is sent. This principle can ensure that the I2C bus does not lose data when multiple hosts attempt to control the bus.

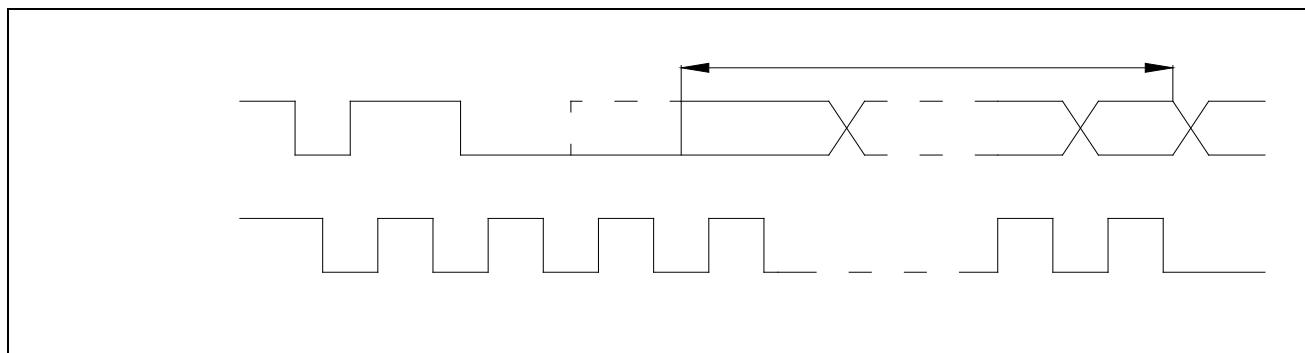


Figure 7-5 Arbitration on the I2C bus

- a) Another device sends serial data;
- b) Another device cancels a logic 1 sent by the I2C master by pulling SDA low (dashed line). Arbitration is lost and I2C enters slave receive mode;
- c) At this time, I2C is in slave receive mode, but still generates clock pulses until the current byte is sent. I2C will not generate clock pulses for the next byte transmission. Once arbitration is won, data transmission on SDA is initiated by the new master.

7.4 Functional Description

The I2C bus uses two wires to transfer information between devices connected to the bus, "SCL" (serial clock line) and "SDA" (serial data line). Filtering logic can filter out glitches on the data bus to protect the integrity of the data. Since there are only non-directional ports, the I2C component requires the use of open-drain buffers on the pins. Each device connected to the bus can be addressed by a specific address using software. The I2C standard is a true multi-master bus with conflict detection and arbitration mechanisms. It prevents data conflicts when two or more masters start transmitting data at the same time. The I2C bus status can be queried in the status register.

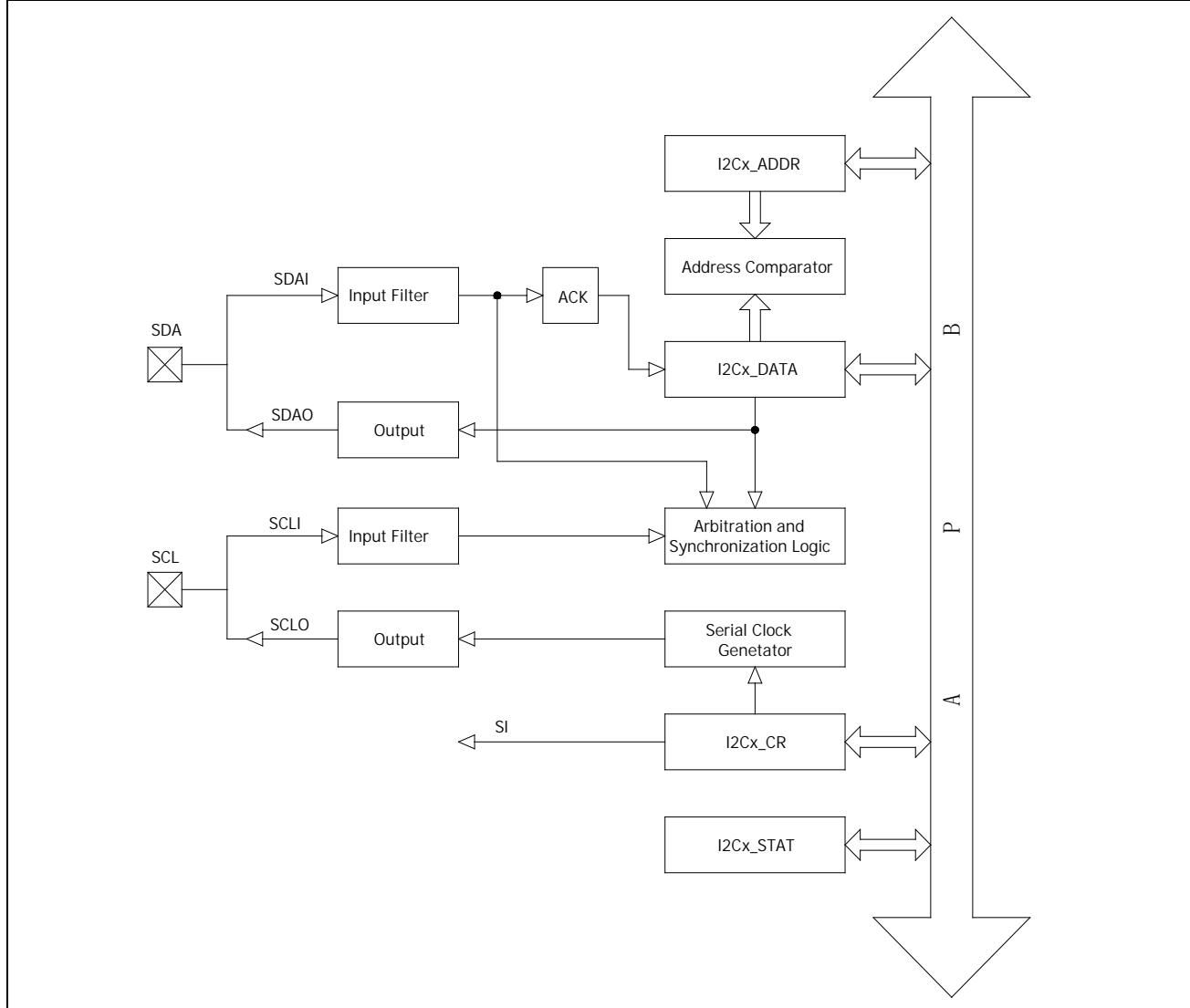


Figure 7-6 I2C functional module diagram

7.4.1 Serial Clock Generator

The serial clock generator uses an 8-bit counter as the baud rate generator. The frequency relationship between the SCL signal and the PCLK signal is $F_{SCL} = F_{PCLK} / 8 / (I2Cx_TM.tm + 1)$, where $I2Cx_TM.tm$ should be greater than 0.

The following table lists the output frequency values of the SCL signal when the PCLK frequency is combined with $I2Cx_TM.tm$.

Table 7-1 I2C clock signal baud rate

PCLK (KHz)	I2Cx_TM.tm						
	1	2	3	4	5	6	7
1000	62	41	31	25	20	17	15
2000	125	83	62	50	41	35	31
4000	250	166	125	100	83	71	62
6000	375	250	187	150	125	107	93
8000	500	333	250	200	166	142	125
10000	625	416	312	250	208	178	156
12000	750	500	375	300	250	214	187
14000	875	583	437	350	291	250	218
16000	1000	666	500	400	333	285	250

7.4.2 Input filter

The input signal is synchronized with PCLK, and the spike pulse signal below the PCLK period will be filtered out.

When this module is used as a host, if the value of $I2C_TM$ is less than or equal to 9, $I2C_CR.H1M$ should be set to 1; if the value of $I2C_TM$ is greater than 9, $I2C_CR.H1M$ should be set to 0.

When this module is used as a slave, if the ratio of PCLK to SCL frequency is less than or equal to 30, $I2C_CR.H1M$ should be set to 1; if the ratio of PCLK to SCL frequency is greater than 30, $I2C_CR.H1M$ should be set to 0.

7.4.3 Address Comparator

The I2C comparator compares its own slave address with the received 7-bit slave address. It can program its own slave address using the "I2Cx_ADDR" register. And it will be compared with the first received 8-bit byte or broadcast address (0x00) according to the "i2cad" bit of the "I2Cx_ADDR" register. If any one of them is the same, the "si" bit of the "I2Cx_CR" register will be set to 1 and an interrupt request will be generated.

7.4.4 Response Flag

The "aa" flag bit of the "I2Cx_CR" register is the response flag bit. When the "aa" bit is 1, the I2C module responds to the response bit after receiving the data, and when the "aa" bit is 0, the I2C module responds to the non-response bit after receiving the data.

7.4.5 Interrupt Generation

The "SI" flag of the "I2Cx_CR" register is the interrupt flag. Whenever the value of the status register (I2Cx_STAT) changes (except changing to 0xF8), the "si" flag will be set to 1. When an interrupt occurs, the status of the I2C bus can be obtained by querying the status register (I2Cx_STAT) to determine the actual source of the interrupt. In order to proceed to the next step, the "si" flag must be cleared by software.

7.4.6 Operation Modes

The I2C component can achieve 8-bit bidirectional data transmission, with a transmission rate of up to 100Kbps in standard mode and 400Kbps in high-speed mode, and up to 1Mbps in ultra-high-speed mode, and can operate in four modes: master transmission mode, master reception mode, slave reception mode, and slave transmission mode. There is also a special mode, broadcast call mode, which operates similarly to the slave reception mode.

■ Master transmission mode

The master sends multiple bytes to the slave, and the master generates a clock, so the set value needs to be filled in I2Cx_TM. In the master transmission mode

It is necessary to I2Cx_CR.sta is set to 1. When the bus is idle, the host initiates a start bit START. If successful, I2Cx_CR.si is set to 1. Next, write the slave address and write bit (SLA +W) into I2Cx_DATA, clear the "si" bit, and send SLA+W on the bus. After the host sends SLA +W and receives the slave's ACK bit, "si" is set to 1. Next, send data according to the user-defined format. After all data is sent, set I2Cx_CR.sto to 1, clear the "si" bit, and send a STOP signal. You can also send a repeated start signal for a new round of data transmission.

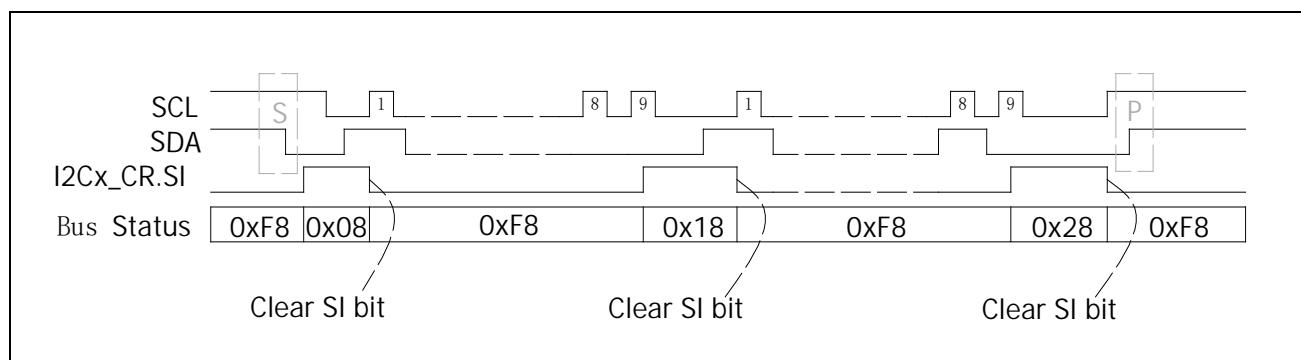
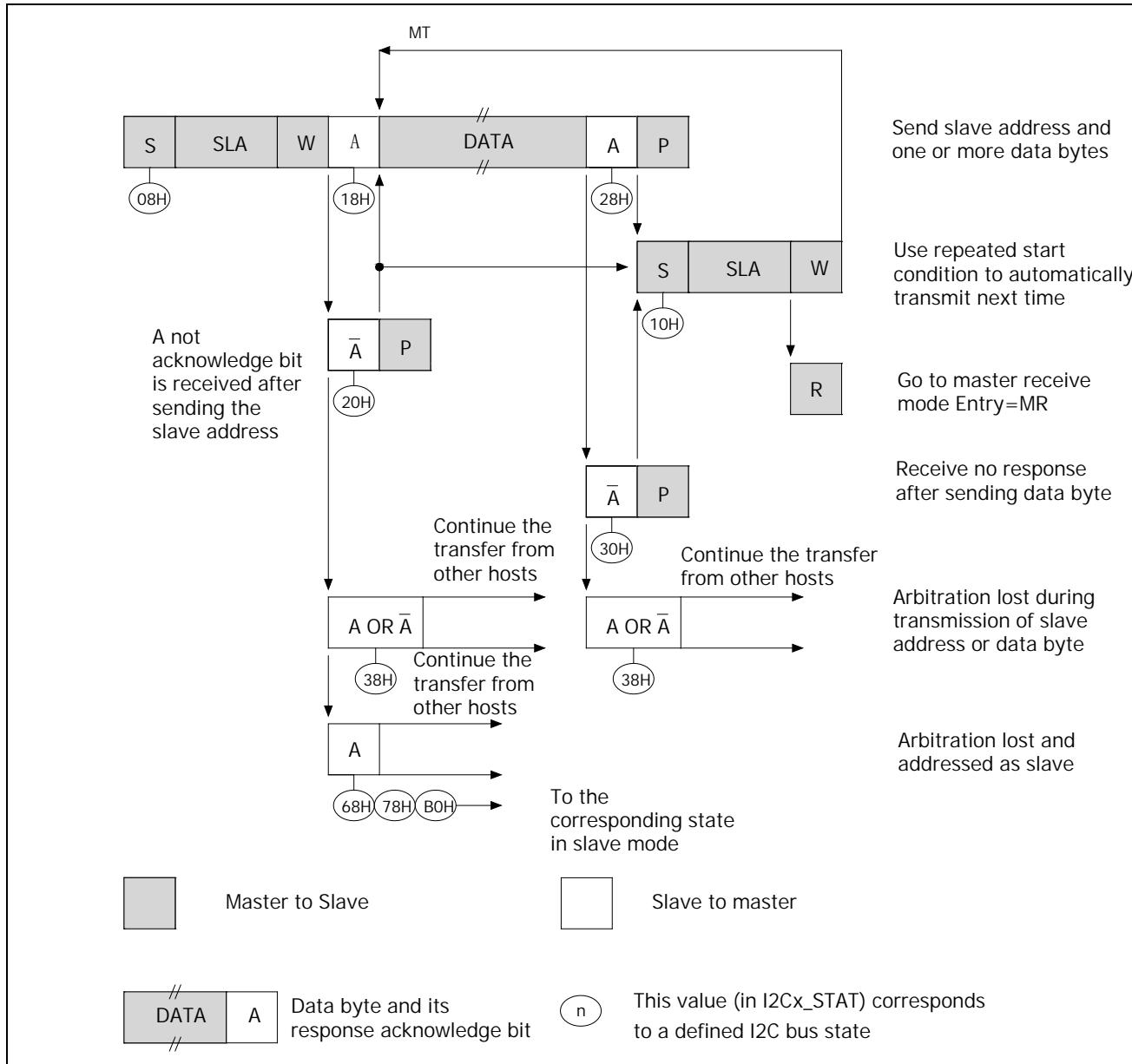


Figure 7-7 Master transmission mode data synchronization diagram

**Figure 7-8 I2C master transmitter status diagram**

■ Master receiving mode

In master receiving mode, the slave transmits data. The initialization setting is the same as the master transmitting mode. After the master sends the start bit, I2Cx_DATA should be written with the slave address and the "read bit" (SLA+R). After receiving the slave's ACK, I2Cx_CR.SI is set to 1. After "SI" is cleared to 0, the slave data is received. If I2Cx_CR.aa is 1, the master responds with an ACK after receiving the data; if it is 0, the master does not respond with NACK after receiving the data. Then the master can send a stop signal or repeat the start signal to start the next round of data transmission.

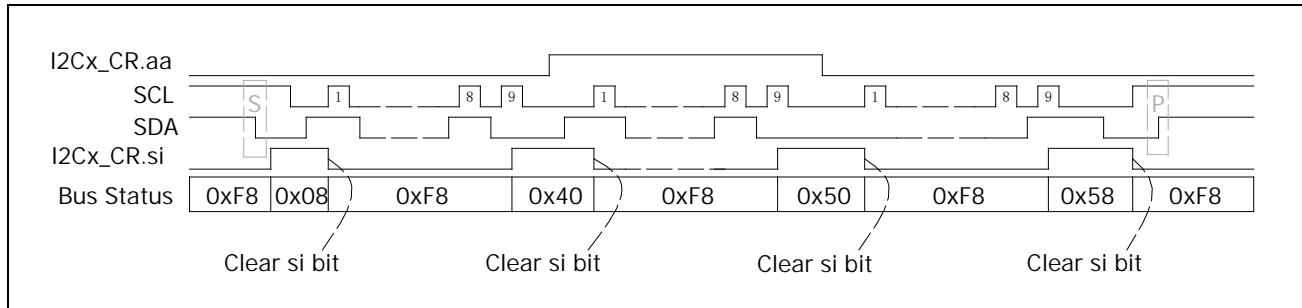


Figure 7-9 Master receiving mode data synchronization diagram

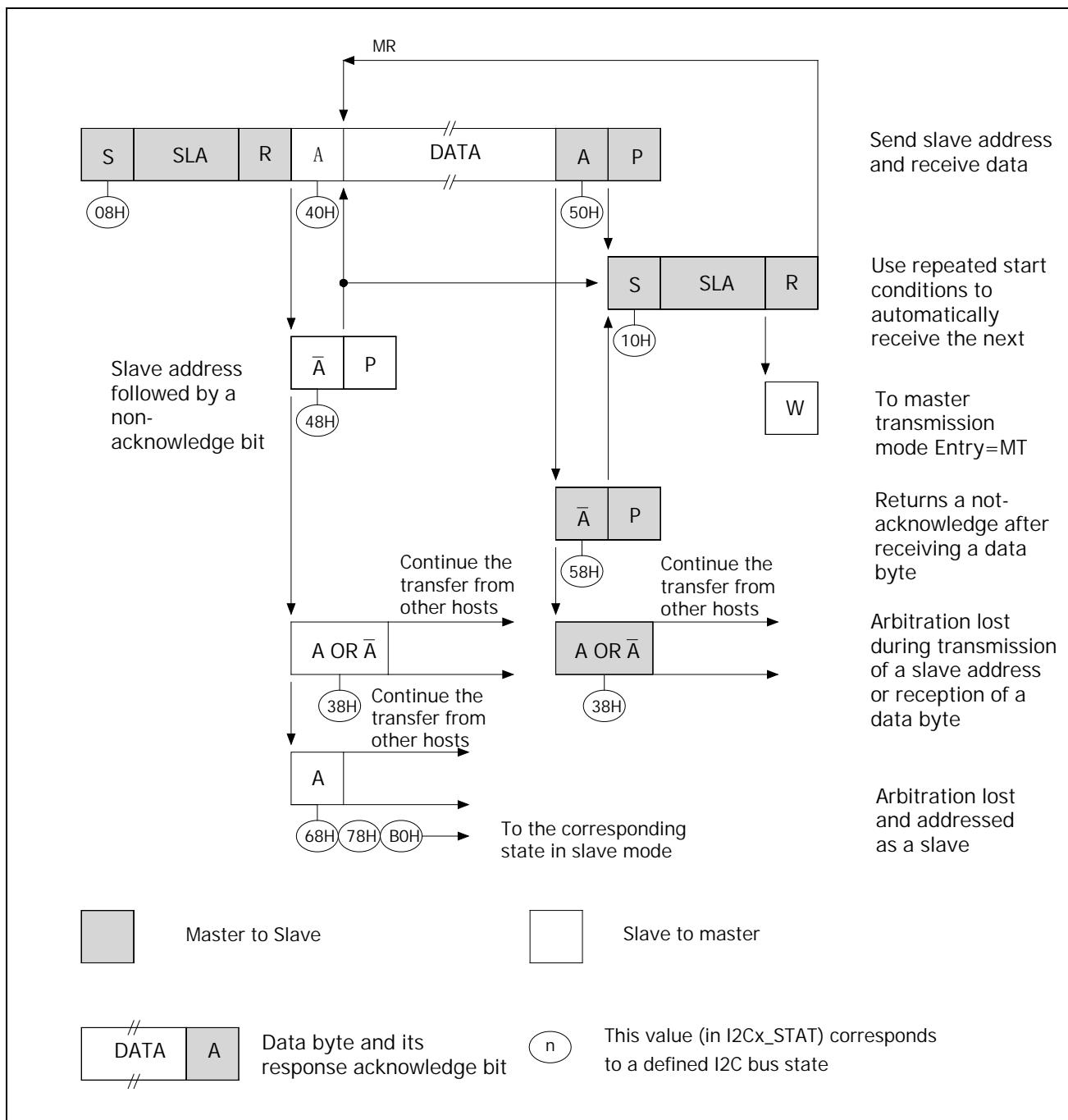


Figure 7-10 I2C master receiving status diagram

■ Slave Receive Mode

In slave receive mode, the slave receives data from the master. Before the transmission starts, I2Cx_ADDR should be written with the slave address, I2Cx_CR.aa is set to 1 to respond to the master's addressing. After the above initialization, the slave enters idle mode and waits for the "write" signal (SLA+W). If the master fails in arbitration, it will directly enter the slave receive mode.

When the slave is addressed by the "write" signal SLA+W, it needs to clear the "si" bit to receive data from the master. If I2Cx_CR.aa=0 during the transmission, the slave will return the non-acknowledge bit NACK in the next byte, and the slave will also become an unaddressed slave, terminate the connection with the master, no longer receive data, and I2C_DATA holds the previously received data. Slave address recognition can be restored by setting "aa", which means that the "aa" bit can temporarily separate the I2C module from the I2C bus.

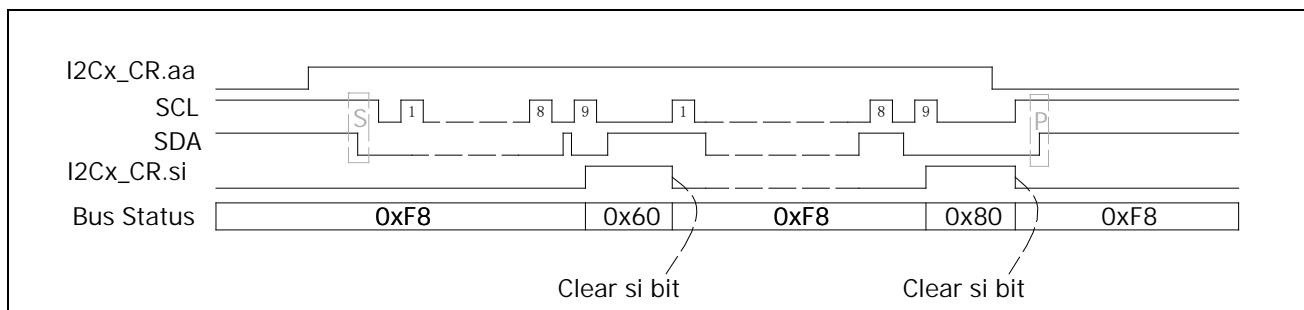


Figure 7-11 Slave receiving mode data synchronization diagram

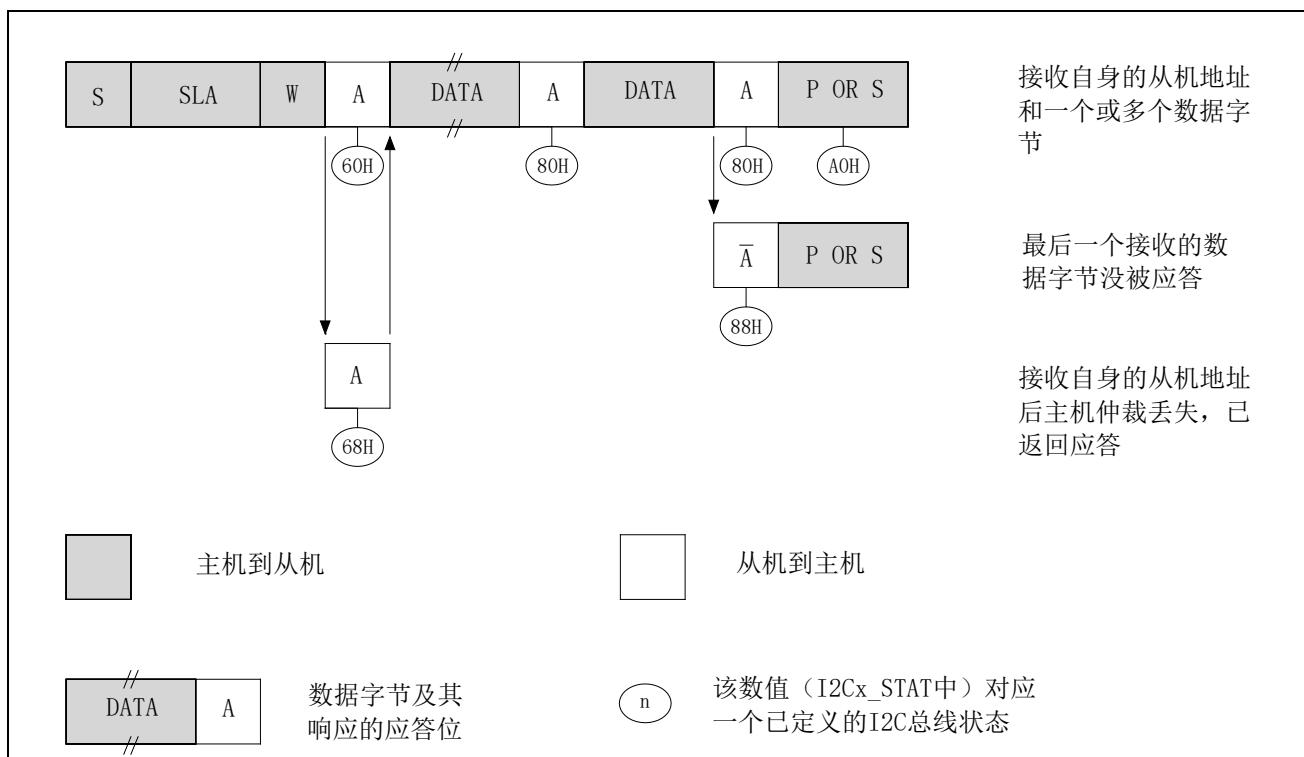


Figure 7-12 Slave receiving state diagram

■ Slave Transmit Mode

In slave transmit mode, data is sent from the slave to the master. After initializing the values of I2Cx_ADDR and I2Cx_CR.aa, the device waits until its own address is addressed by the "read" signal (SLA+R). If the master fails in arbitration, it can also enter the slave transmit mode.

When the slave is addressed by the "read" signal SLA+R, it is necessary to clear "si" to send data to the master. Usually the master returns an acknowledge bit after receiving each byte of data.

If I2Cx_CR.aa is cleared during the transmission, the slave will send the last byte of data, and send all 1 data in the next transmission, and become an unaddressed slave.

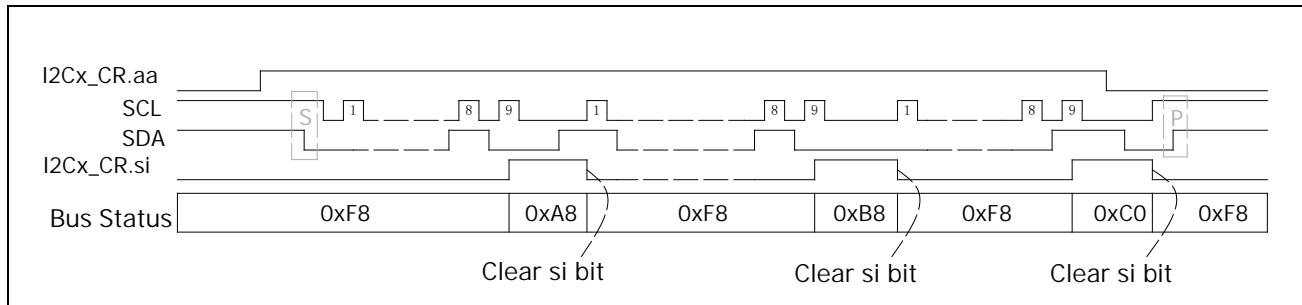


Figure 7-13 Data synchronization diagram in slave transmission mode

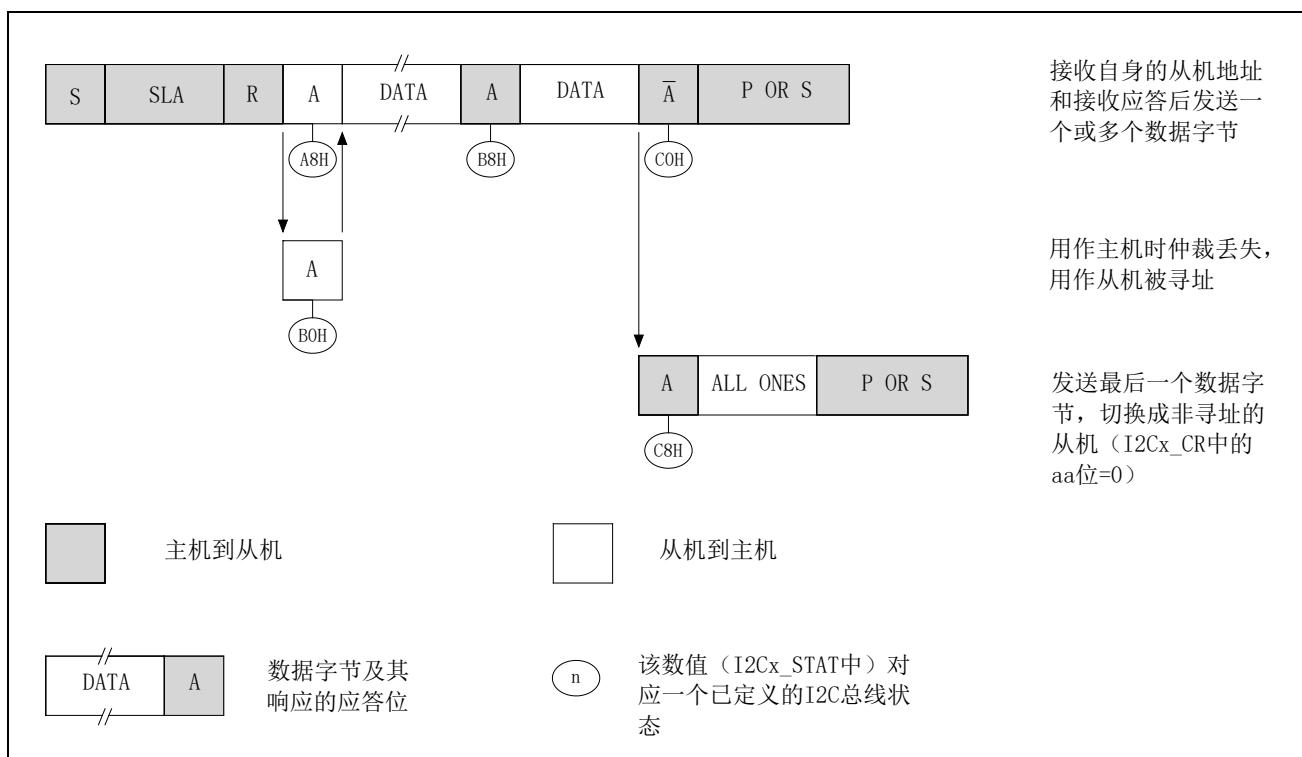


Figure 7-14 I2C slave transmitter status diagram

■ Broadcast call mode

Broadcast call mode is a special slave receiving mode, the addressing mode is 0x00, the slave address and read and write are both 0. When I2Cx_ADDR.GC and I2Cx_CR.aa are both set to 1, the broadcast call mode is enabled. In this mode, the I2Cx_STAT value is different from the I2Cx_STAT value in the normal slave receiving mode. The broadcast call mode may also be entered if arbitration fails.

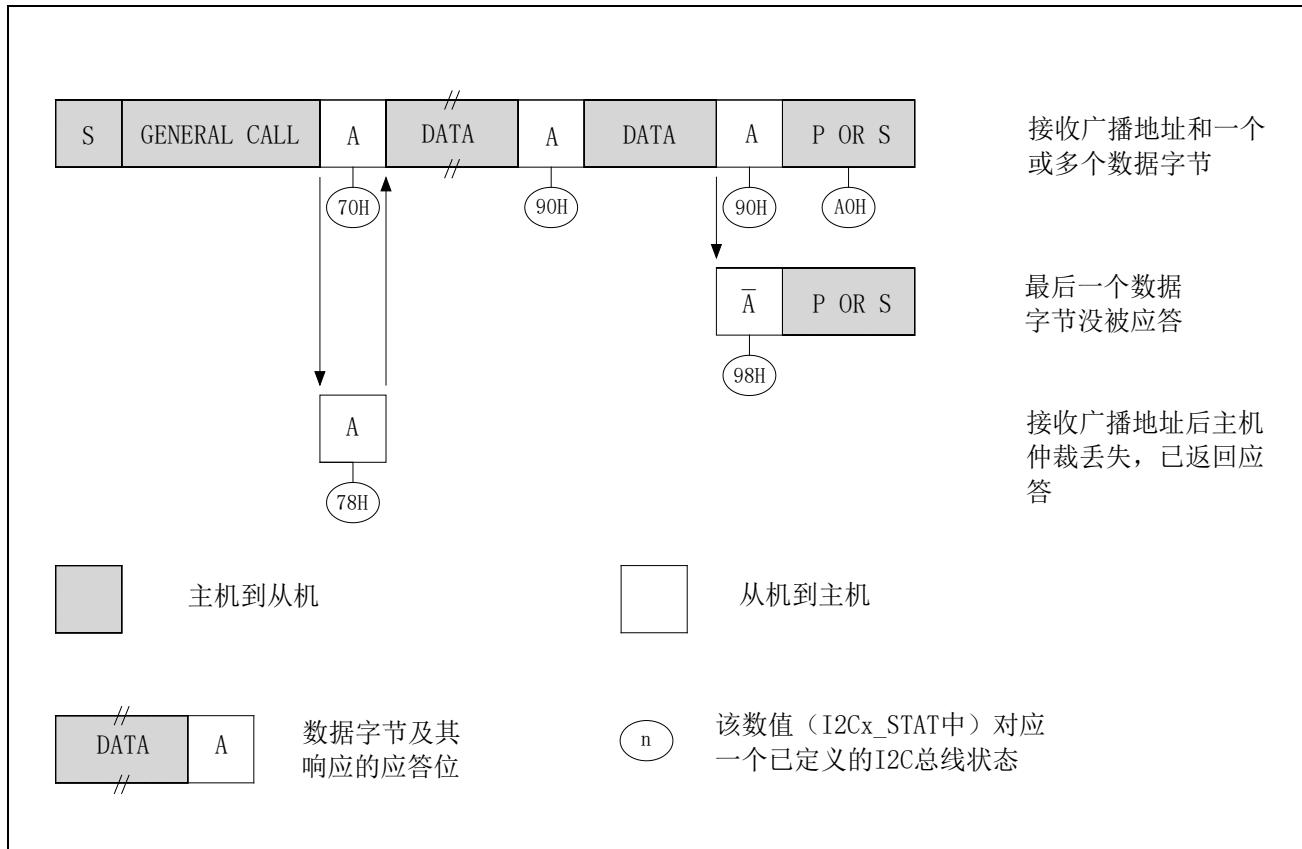


Figure 7-15 I2C Broadcast Call Status Diagram

7.4.7 Status code description

There are two special states in the I2C status register: F8H and 00H.

F8H: This status code indicates that no relevant information is available because the serial interrupt flag "si" has not been set. This situation occurs between other states and the I2C module has not started to perform serial transmission.

00H: This status code indicates that a bus error has occurred during the I2C serial transmission. Bus errors occur when a start or stop condition appears at an illegal position in the format frame. These illegal positions are address bytes, data bytes or acknowledge bits during serial transmission. Bus errors also occur when external interference affects the internal I2C module signals. "si" is set when a bus error occurs.

Table 7-2 I2C status code description

Status code	Description
Master transmission mode	
08H	Start condition sent
10H	Repeated start condition sent
18H	Sent SLA+W, received ACK
20H	Sent SLA+W, received NACK
28H	Sent data in I2Cx_DATA, received ACK
30H	Sent data in I2Cx_DATA, received NACK
38H	Lost arbitration during SLA+ read or write data byte
Master receiving mode	
08H	Start condition sent
10H	Repeated start condition sent
38H	Lost arbitration in non-ACK
40H	SLA+R sent, ACK received
48H	SLA+R sent, NACK received
50H	Data byte received, ACK returned
58H	58H Data byte received, NACK returned
Slave receiving mode	
60H	Received its own SLA+W, ACK returned
68H	Lost arbitration in SLA+ read/write when master, received its own SLA+W, ACK returned
80H	The previous addressing used its own slave address, received data byte, ACK returned
88H	The previous addressing used its own slave address, received data byte, NACK returned
A0H	Stop condition or repeated start condition received during static addressing
Slave transmitting mode	
A8H	Received its own SLA+R, ACK returned
B0H	Lost arbitration when master, received its own SLA+R, ACK returned
B8H	Data sent, ACK received
C0H	Data byte sent, NACK received
C8H	Loaded data byte sent, ACK received
Broadcast call mode	
70H	Received broadcast address (0x00), ACK returned
78H	Master lost arbitration in SLA+ read/write, received broadcast address, ACK returned

Status code	Description
90H	The previous addressing used the broadcast address, received data byte, ACK returned
98H	The previous addressing used the broadcast address, received data byte, NACK returned
A0H	When static addressing, a stop condition or repeated start condition was received
Other miscellaneous status	
F8H	No relevant status information is available, si=0
00H	A bus error occurred during the transmission process, or external interference caused I2C to enter an undefined state

7.5 Programming Examples

7.5.1 Master transmitter example

Step1: Set PERI_CLKEN0.I2Cx to 1 to enable the I2Cx module clock.

Step2: Write 0 and 1 to PERI_RESET.I2Cx in sequence to reset the I2Cx module.

Step3: Configure the ports used by SCL and SDA of I2C to open-drain mode. According to the mode of I2C, the appropriate port direction needs to be configured (for example, if the port direction is configured as output, the corresponding output register needs to be initialized to 1 before the corresponding bit of DIR is cleared), and finally map SCL and SDA to the corresponding pins.

Step4: Configure I2Cx_TM to make the clock rate of SCL meet the application requirements.

Step5: Set I2Cx_TMRUN to 1 to enable the SCL clock generator.

Step6: Set I2Cx_CR.ens to 1 to enable the I2C module.

Step7: Set I2Cx_CR.sta to 1, and the bus attempts to send a Start signal.

Step8: Wait for I2Cx_CR.si to become 1, and the Start signal has been sent to the bus.

Step9: Query I2Cx_STAT. If the register value is 0x08 or 0x10, proceed to the next step, otherwise perform error processing.

Step10: Write SLA+W to I2Cx_DATA, set I2Cx_CR.sta to 0, set I2Cx_CR.si to 0, and send SLA+W.

Step11: Wait for I2Cx_CR.si to become 1, SLA+W has been sent to the bus.

Step12: Query I2Cx_STAT, if the register value is 0x18, proceed to the next step. Otherwise, error handling is performed.

Step13: Write the data to be sent to I2Cx_DATA, set I2Cx_CR.si to 0, and send the data.

Step14: Wait for I2Cx_CR.si to become 1, the data has been sent to the bus.

Step15: Query I2Cx_STAT, if the register value is 0x28, proceed to the next step. Otherwise, error handling is performed.

Step16: If the data to be sent is not completed, jump to Step13 and continue execution.

Step17: Set I2Cx_CR.sto to 1, set I2Cx_CR.si to 0, and the bus attempts to send a Stop signal.

7.5.2 Master receiver example

Step1: Set PERI_CLKEN0.I2Cx to 1 to enable the I2Cx module clock.

Step2: Write 0 and 1 to PERI_RESET.I2Cx in sequence to reset the I2Cx module.

Step3: Configure the ports used by SCL and SDA of I2C to open-drain mode. According to the mode of I2C, the appropriate port direction needs to be configured (for example, if the port direction is configured as output, the corresponding output register needs to be initialized to 1 before the corresponding bit of DIR is cleared), and finally map SCL and SDA to the corresponding pins.

Step4: Configure I2Cx_TM to make the clock rate of SCL meet the application requirements.

Step5: Set I2Cx_TMRUN to 1 to enable the SCL clock generator.

Step6: Set I2Cx_CR.ens to 1 to enable the I2C module.

Step7: Set I2Cx_CR.sta to 1, and the bus attempts to send a Start signal.

Step8: Wait for I2Cx_CR.si to become 1, and the Start signal has been sent to the bus.

Step9: Query I2Cx_STAT. If the register value is 0x08 or 0x10, proceed to the next step, otherwise perform error processing.

Step10: Write SLA+R to I2Cx_DATA, set I2Cx_CR.sta to 0, set I2Cx_CR.si to 0, and send SLA+R.

Step11: Wait for I2Cx_CR.si to become 1, SLA+R has been sent to the bus.

Step12: Query I2Cx_STAT, if the register value is 0x40, continue to the next step, otherwise perform error processing.

Step13: Set I2Cx_CR.aa to 1 to enable the response flag.

Step14: Set I2Cx_CR.si to 0, the slave sends data, and the host sends ACK or NACK according to I2Cx_CR.aa.

Step15: Wait for I2Cx_CR.si to become 1, and read the received data from I2Cx_DATA.

Step16: Query I2Cx_STAT, if the register value is 0x50 or 0x58, continue to the next step, otherwise perform error processing.

Step17: If the data to be received is only the last byte away, set I2Cx_CR.aa to 0 and enable the non-response flag.

Step 18: If the data to be received is not completed, jump to Step 14 to continue execution.

Step 19: Set I2Cx_CR.sto to 1, set I2Cx_CR.si to 0, and the bus attempts to send a Stop signal.

7.5.3 Slave receiver example

Step1: Set PERI_CLKEN0.I2Cx to 1 to enable the I2Cx module clock.

Step2: Write 0 and 1 to PERI_RESET.I2Cx in sequence to reset the I2Cx module.

Step3: Configure the ports used by SCL and SDA of I2C to open-drain mode. According to the mode of I2C, the appropriate port direction needs to be configured (for example, if the port direction is configured as output, the corresponding output register needs to be initialized to 1 before the corresponding bit of DIR is cleared), and finally map SCL and SDA to the corresponding pins.

Step4: Set I2Cx_CR.ens to 1 to enable the I2C module.

Step5: Configure I2Cx_ADDR as the slave address.

Step6: Set I2Cx_CR.aa to 1 to enable the response flag.

Step7: Wait for I2Cx_CR.si to become 1 and be addressed by SLA+W.

Step8: Query I2Cx_STAT. If the register value is 0x60, proceed to the next step, otherwise perform error processing.

Step9: Set I2Cx_CR.si to 0. The host sends data, and the slave returns ACK or NACK according to I2Cx_CR.aa.

Step10: Wait for I2Cx_CR.si to become 1, and read the received data from I2Cx_DATA.

Step11: Query I2Cx_STAT. If the register value is 0x80, proceed to the next step, otherwise perform error processing

Step12: If the data to be received is not completed, jump to Step9 and continue execution.

Step13: Set I2Cx_CR.aa to 0 and set I2Cx_CR.si to 0.

7.5.4 Slave transmitter example

Step1: Set PERI_CLKEN0.I2Cx to 1 to enable the I2Cx module clock.

Step2: Write 0 and 1 to PERI_RESET.I2Cx in sequence to reset the I2Cx module.

Step3: Configure the ports used by SCL and SDA of I2C to open-drain mode. According to the mode of I2C, the appropriate port direction needs to be configured (for example, if the port direction is configured as output, the corresponding output register needs to be initialized to 1 before the corresponding bit of DIR is cleared), and finally map SCL and SDA to the corresponding pins.

Step4: Set I2Cx_CR.ens to 1 to enable the I2C module.

Step5: Configure I2Cx_ADDR as the slave address.

Step6: Set I2Cx_CR.aa to 1 to enable the response flag.

Step7: Wait for I2Cx_CR.si to become 1 and be addressed by SLA+R.

Step8: Query I2Cx_STAT. If the value of the register is 0xA8, proceed to the next step, otherwise perform error processing.

Step9: Write the data to be sent to I2Cx_DATA, set I2Cx_CR.si to 0, and send the data.

Step10: Wait for I2Cx_CR.si to become 1, and the data has been sent to the bus.

Step11: Query I2Cx_STAT. If the value of the register is 0xB8 or 0xC0, proceed to the next step, otherwise perform error processing.

Step12: If the data to be sent is not completed, jump to Step9 and continue execution.

Step13: Set I2Cx_CR.aa to 0 and set I2Cx_CR.si to 0.

7.6 Register Description

Register List

I2C0 base address: 0x40000400

I2C1 base address: 0x40004400

Table 7-3 I2C Register List

Offset	Register Name	Access	Register Description
0x00	I2Cx_TMRUN	RW	I2C baud rate counter enable register.
0x04	I2Cx_TM	RW	I2C baud rate counter configuration register.
0x08	I2Cx_CR	RW	I2C configuration register.
0x0c	I2Cx_DATA	RW	I2C data register.
0x10	I2Cx_ADDR	RW	I2C address register.
0x14	I2Cx_STAT	RO	I2C status register.

7.6.1 I2C Baud Rate Counter Enable Register (I2Cx_TMRUN)

Address offset: 0x00

Reset value: 0x00000000

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

Bit	Mark	Function Description
31: 1	Reserved	
0	tme	Baud rate counter enable. 0 – Disable 1 – Enable

7.6.2 I2C Baud Rate Counter Configuration Register (I2Cx_TM)

Address offset: 0x04

Reset value: 0x00000000

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

Bit	Mark	Function Description
31:8	Reserved	
7:0	tm	tm: Baud rate counter configuration value. Fscl = Fpclk / 8 / (tm+1), where tm >0

7.6.3 I2C Configuration Register (I2Cx_CR)

Address offset: 0x08

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ens	sta	sto	si	aa	Res	h1m	RW

Bit	Mark	Function Description
31:7	Reserved	
6	ens	I2C module enable control 0 - Disable 1 - Enable
5	sta	I2C bus control 0 - No function 1 - Send START to the bus
4	sto	I2C bus control 0 - No function 1 - Send STOP to the bus
3	si	I2C interrupt flag Read 1, I2C interrupt has occurred Write 0, I2C performs the next operation
2	aa	Acknowledge control bit 0 - Send NAK 1 - Send ACK
1	Reserved	
0	h1m	I2C filter parameter configuration 0 - Advanced filtering, higher anti-interference performance 1 - Simple filtering, faster communication rate <i>Note: See the [Input filter] section for details.</i>

7.6.4 I2C Data Register (I2Cx_DATA)

Address offset: 0x0C

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										i2cdat				RW	

Bit	Mark	Function Description
31:8	Reserved	
7:0	i2cdat	I2C 数据寄存器 在 I2C 发送模式下, 写入待发送的数据 在 I2C 接收模式下, 读出收到的数据

7.6.5 I2C Address Register (I2Cx_ADDR)

Address offset: 0x10

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										i2cadr				GC	
										RW				RW	

Bit	Mark	Function Description
31:8	Reserved	
7:1	i2cadr	I2C slave mode address.
0	GC	Broadcast address response enable 0 - Disable 1 - Enable

7.6.6 I2C Status Register (I2Cx_STAT)

Address offset: 0x14

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								i2csta RO							

Bit	Mark	Function Description
31:8	Reserved	
7:0	i2csta	I2C Status Register For the specific definition of the status value, please refer to the [Status Code Description] section

8 Serial Peripheral Interface (SPI)

8.1 SPI Introduction

The SPI interface is a synchronous serial data communication interface operating in full-duplex mode, using 4 pins for communication: MISO, MOSI, SCK, CS/SSN. When SPI is used as a master, it outputs CS and SCK signals to control the communication process. When SPI is used as a slave, it communicates under the control of SSN and SCK signals.

8.2 SPI Main Features

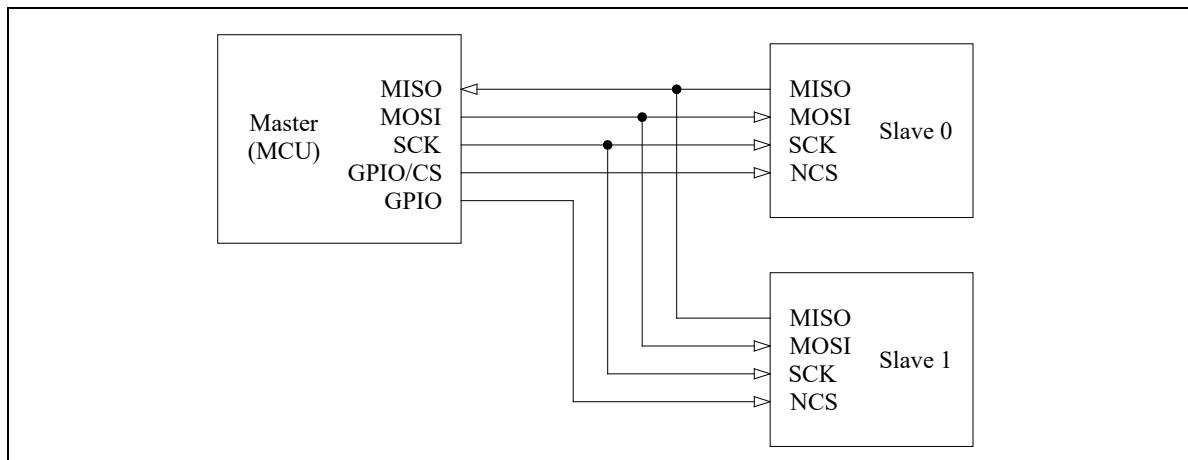
- Supports SPI master mode and SPI slave mode
- Supports standard four-wire full-duplex communication
- Supports configuration of serial clock polarity and phase
- The master mode supports 7 communication rates
- The maximum division factor of the master mode is PCLK/2
- The maximum division factor of the slave mode is PCLK/4
- The frame length is fixed to 8 bits, and the MSB is transmitted first
- Supports DMA software/hardware access

8.3 SPI Function Description

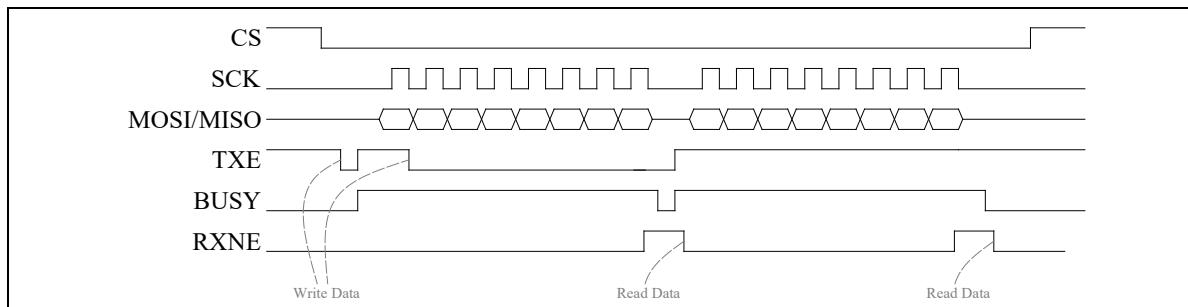
8.3.1 SPI Master Mode

The length of each data frame is fixed to 8 bits, and the first bit of the transmitted data is fixed to MSB. Set SPIx_CR.mstr to 1, and the SPI interface works in the host mode. Write data to the SPIx_Data register to start the SPI transmission, and the SCK pin will automatically generate a serial clock; at the serial clock edge, the data in the shift register is sent to the MOSI pin, and the data on the MISO pin is received in the shift register. The frequency of the SCK pin output clock is controlled by SPIx_CR[spr2:spr0], and its output frequency range is PCLK /2~PCLK/128; the output level of the CS pin is controlled by SPIx_SSN.ssn, and the output level of the GPIO pin is controlled by the GPIO related register.

The typical application block diagram of the host mode is shown below.

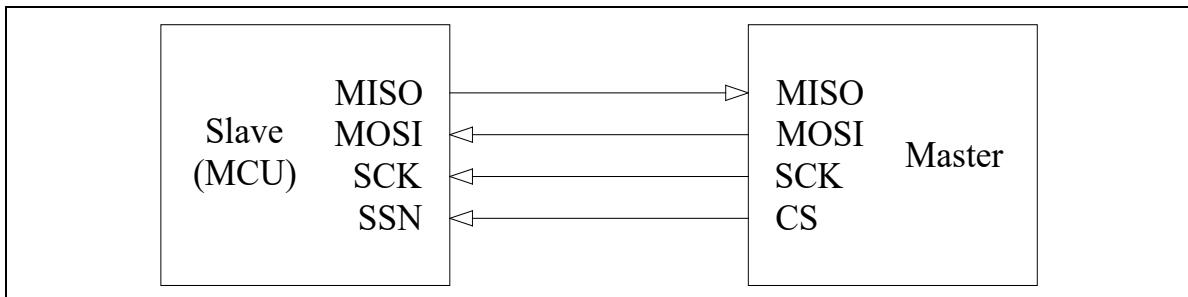


The communication diagram of host mode is shown in the figure below, where CPOL=0 and CPHA=0.



8.3.2 SPI Slave Mode

The length of each data frame is fixed to 8 bits, and the first bit of the received data is fixed to MSB. Set SPIx_CR.mstr to 0, and the SPI interface works in slave mode. In this mode, the SCK pin is used as an input pin and the serial clock comes from the external host; the SSN pin is used as an input pin and the chip select signal comes from the external host or is fixed to low level. For details about the SSN pin, see the auxiliary register in the GPIO chapter. The typical application block diagram of the slave mode is shown below.



When the SPI slave receives a byte of data from the SPI master, the RXNE bit will be set high; the user program should read the received data as soon as possible. The communication timing of receiving data in slave mode is shown below, where CPOL=0, CPHA=0.

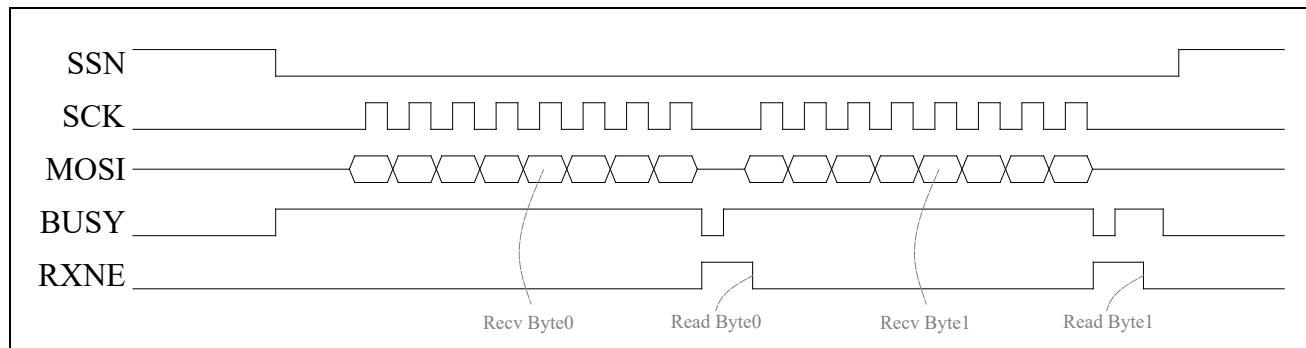


Figure 8-1 Schematic diagram of slave reception

When the SPI slave needs to send data to the host, the following five steps should be performed in sequence before the host pulls NSS low: set the PERI_RESET0.SPIx bit to 0, set the PERI_RESET0.SPIx bit to 1, configure the SPI communication parameters, and write the first byte of data to be sent to the SPIx_DATA register; after NSS is pulled low, whenever the TXE flag is queried to be 1, the subsequent data to be sent should be written to the SPIx_DATA register as soon as possible. The communication timing of sending data in slave mode is as follows, where CPOL=0 and CPHA=0.

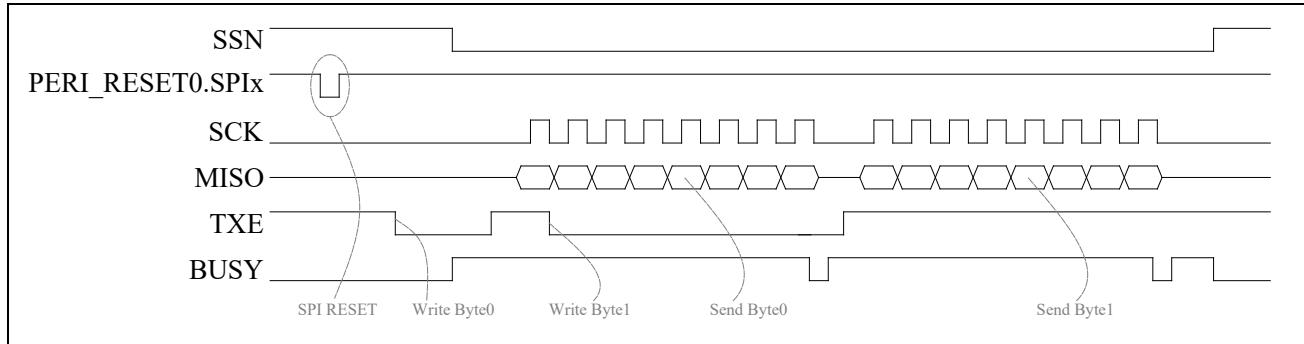


Figure 8-2 Schematic diagram of slave transmission

8.3.3 SPI data frame format

The SPI interface frame format depends on the configuration of the clock polarity bit CPOL and the clock phase bit CPHA.

When CPOL is 0, the idle state of the SCK line is low. When CPOL is 1, the idle state of the SCK line is high. When CPHA is 0, the data will be sampled when the first SCK clock conversion signal jumps. When CPHA is 1, data will be sampled when the second SCK clock signal jumps.

The SPI interface master frame format is shown in the figure below.

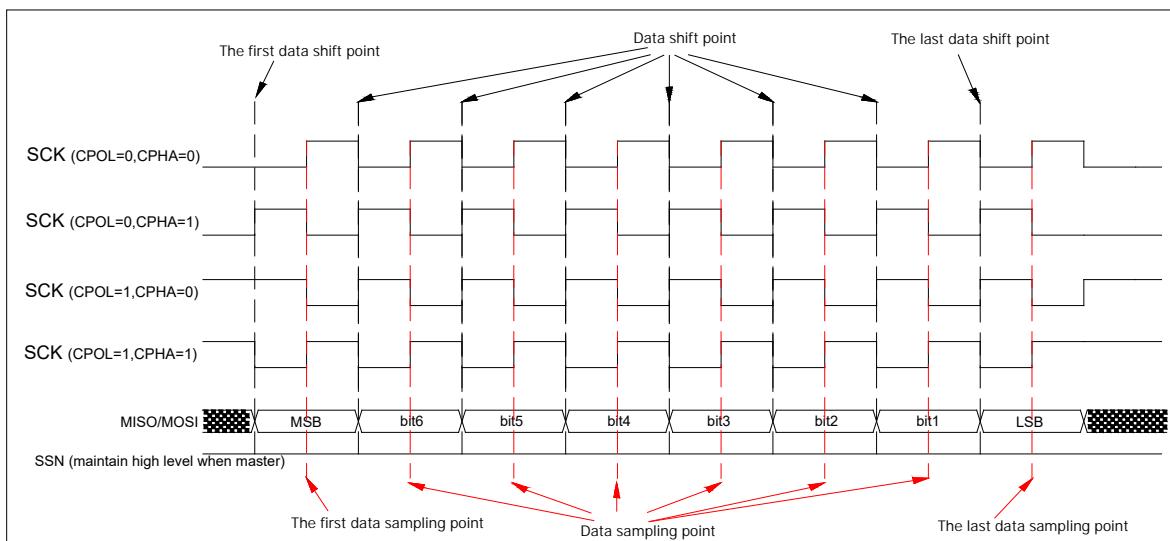


Figure 8-3 Master mode frame format

The SPI interface slave frame format is shown in the figure below.

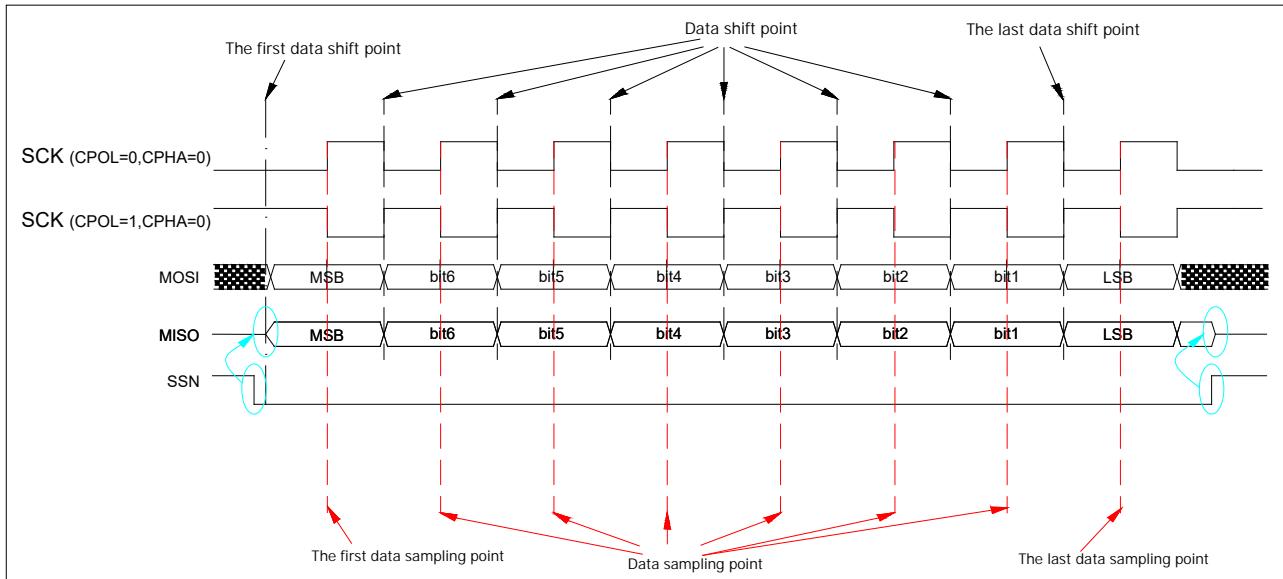


Figure 8-4 Data frame format when slave CPHA is 0

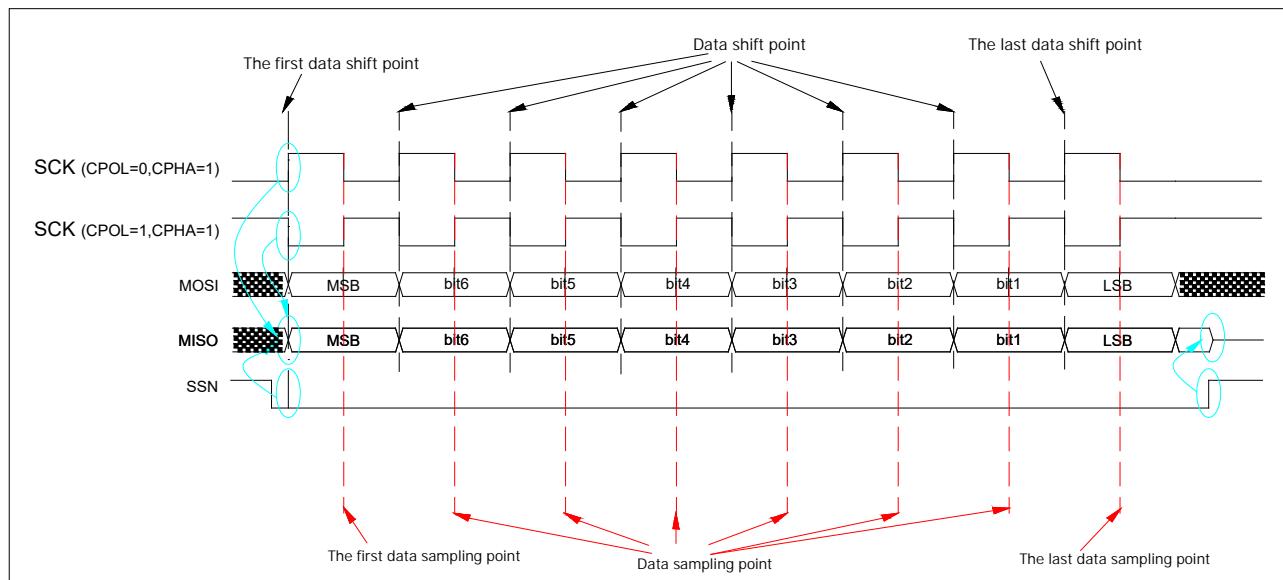


Figure 8-5 Data frame format when slave CPHA is 1

8.3.4 SPI Status Flags and Interrupts

SPI will generate the following four status flags during operation. The conditions for their generation and clearing methods are as follows.

- When the data in the transmit buffer (SPIx_Data) is transferred to the transmit shift register, SPIx_STAT.txe will be set by hardware, indicating that the transmit buffer is empty and the next data can be written. Writing data to SPIx_DATA can clear this flag.
- When the data in the receive shift register is transferred to the receive buffer (SPIx_Data), SPIx_STAT.rxne will be set by hardware, indicating that the receive buffer is not empty and the user needs to read the data as soon as possible. Reading data from SPIx_DATA can clear this flag.
- When SPI works in master mode and the external SSN input is low, SPIx_STAT.mdf will be set by hardware, indicating that other SPI hosts are occupying the bus. When the SSN input is high, SPIx_STAT.mdf will be automatically cleared by hardware.
- When SPI is in slave mode, if the SSN pin is pulled high during data transmission, SPIx_STAT.sserr will be set. Setting SPIx_CR.spen to 0 can clear this flag.

If SPI interrupts are enabled (SPIx_CR2.int_en=1), the following three situations can generate interrupts:

- SPI transmit buffer is empty, that is, SPIx_STAT.txe is 1
- SPI receive buffer is not empty, that is, SPIx_STAT.rxne is 1
- SPI master mode error, that is, SPIx_STAT.mdf is 1

In the interrupt service routine, SPIx_ICLR needs to be written with 0x00 to clear the internal interrupt flag.

8.3.5 SPI multi-machine system configuration description

- When the SPI module is used as a host and works in a single-host system, the slave can be controlled through the SPI_CS pin or the GPIO pin. When the SPI_CS pin is selected as the chip select signal of the slave, the slave can be selected by setting SPIx_SSNS.ssn to 0, and the slave can be released by setting SPIx_SSNS.ssn to 1. When the GPIO pin is selected as the chip select signal of the slave, the slave can be selected by setting the corresponding bit of the GPIOx_OUT register to 0, and the slave can be released by setting the corresponding bit of the GPIOx_OUT register to 1.
- When the SPI module is used as a slave, the source of SPI_SSNS can be configured as needed (see GPIO port auxiliary controller for details). When SSN is low, the slave can be selected for communication; when SSN is high, the slave is in an unselected state.
- When the SPI mode works in multi-master and multi-slave mode, all slave chip select signals are connected through GPIO pins. The master must also connect to the SSN signals of other masters through GPIO pins to monitor whether the bus is occupied. The operation method that Master0 needs to communicate as shown in the figure below is: wait for Master0.SSN to become high; output low from GPIO0 to notify Master1 to release the SPI bus; output low from GPIO2 to select Slave1; communicate with Slave1; output high from GPIO2 to release Slave1; output high from GPIO0 to release Master1.

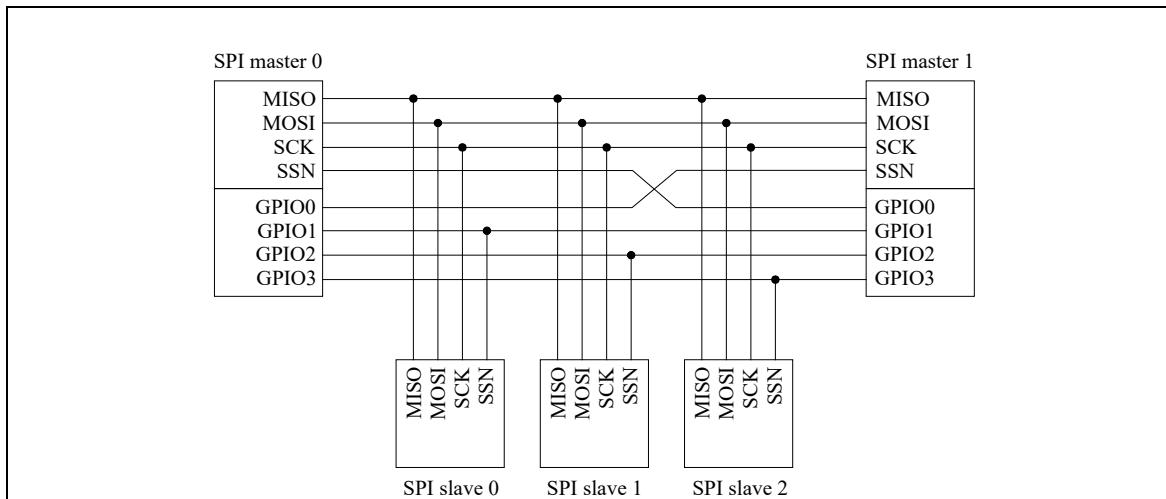


Figure 8-6 Schematic diagram of SPI multi-master/multi-slave system

8.3.6 SPI pin configuration description

SPI can maintain some or all functions under some special pin configurations.

The specific situation is as follows ("√" means the pin is configured and used, blank means the pin is not configured):

Table 8-1 SPI pin configuration description table

	SPI_CS (master)/	SCK	MOSI	MISO	Functional description
Master Mode	√	√	√	√	General configuration All master functions are normal
		√	√	√	All master functions are normal
	√	√	√		Master transmitting function is normal
	√	√		√	Master receiving function is normal
		√	√		Master transmitting function is normal
		√		√	Master receiving function is normal
Slave Mode	√	√	√	√	General configuration All slaves function are normal
	√	√	√		Slave receiving function is normal
	√	√		√	Slave transmitting function is normal
	√ Fixed low level	√	√	√	All slaves function are normal
	√ Fixed low level	√	√		Slave receiving function is normal
	√ Fixed low level	√		√	Slave transmitting function is normal

Notes:

- Situations not listed in the table are not supported yet.
- In master mode, even if the SPIx_CS chip select output is not used, SPIx.SSN needs to be set to 1 before sending data, and SPIx.SSN needs to be set to 0 after sending data.
- In slave mode and when the chip select input is fixed at a low level, SPIx_CR.cpha=1 must be satisfied to maintain normal function.

8.4 SPI Programming Examples

8.4.1 SPI master transmitter example

Step1: Map CS/SCK/MISO/MOSI to the required pins according to the description of the digital multiplexing function of the pins in the GPIO chapter; and configure the CS/SCK/MOSI pins to output mode and the MISO pin to input mode.

Step2: Set SPIx_CR.mstr to 1 to make the SPI work in master mode.

Step3: Configure SPIx_CR[spr2:spr0] so that the clock rate of the SCK output meets the application requirements.

Step4: Configure SPIx_CR.cpol and SPIx_CR.cpha so that the data frame format meets the application requirements.

Step5: Set SPIx_CR.spen to 1 to enable the SPI interface.

Step6: Set SPIx_SSN.ssn to 0 so that the CS pin outputs a low level to select the slave.

Step7: When it is found that SPIx_STAT.txe is 1, write the data to be sent to SPIx_DATA as soon as possible.

Step8: If the data to be sent is not completed, jump to Step7 to continue execution.

Step9: Query and wait for SPIx_STAT.txe to become 1, and the last byte of data has begun to be sent.

Step10: Query and wait for SPIx_STAT.busy to become 0, and the SPI bus is idle.

Step11: Set SPIx_SSN.ssn to 1, so that the CS pin outputs a high level to release the slave.

Notes:

- GPIO can be used instead of CS to achieve chip select output, which is mostly used in multi-machine communication systems.

- During the transmission process, SPIx_SSN.ssn must be set to 0, and after the transmission is completed, SPIx_SSN.ssn must be set to 1.

8.4.2 SPI master receiver example

Step1: Map CS/SCK/MISO/MOSI to the required pins according to the description of the digital multiplexing function of the pins in the GPIO chapter; and configure the CS/SCK/MOSI pins to output mode and the MISO pin to input mode.

Step2: Set SPIx_CR.mstr to 1 to make the SPI work in master mode.

Step3: Configure SPIx_CR[spr2:spr0] so that the clock rate of the SCK output meets the application requirements.

Step4: Configure SPIx_CR.cpol and SPIx_CR.cpha so that the data frame format meets the application requirements.

Step5: Set SPIx_CR.spen to 1 to enable the SPI interface.

Step6: Set SPIx_SSN.ssn to 0 so that the CS pin outputs a low level to select the slave.

Step7: Write any data to SPIx_DATA to trigger the host to send SCK.

Step 8: Query and wait for SPIx_STAT.rxne to become 1, indicating that the data sent by the slave has been received.

Step9: Read the received data from SPIx_DATA.

Step10: If the data to be received is not completed, jump to Step7 to continue execution.

Step11: Set SPIx_SSN.ssn to 1, so that the CS pin outputs a high level to release the slave.

Notes:

- GPIO can be used instead of CS to implement chip select output, which is mostly used in multi-machine communication systems.
- During the transmission process, SPIx_SSN.ssn must be set to 0, and after the transmission is completed, SPIx_SSN.ssn must be set to 1.

8.4.3 SPI slave transmitter example

Step1: Map SSN/SCK/MISO/MOSI to the required pins according to the description of the digital multiplexing function of the pins in the GPIO chapter; and configure the SSN/SCK/MOSI pins to input mode and the MISO pin to output mode. For details on the source of the SSN pin, see the GPIO port auxiliary controller.

Step2: Set PERI_RESET0.SPIx to 0 to reset the SPI module.

Step3: Set PERI_RESET0.SPIx to 1 to put the SPI module in working state.

Step4: Set SPIx_CR.mstr to 0 to make the SPI work in slave mode.

Step5: Configure SPIx_CR.cpol and SPIx_CR.cpha to make the data frame format meet the application requirements.

Step6: Set SPIx_CR.spen to 1 to enable the SPI interface.

Step7: Write the first byte of data to be sent to SPIx_DATA.

Step8: Query and wait for the SSN pin to be pulled low, and the host selects the SPI slave.

Step9: When the query SPIx_STAT.txe is 1, write the data to be sent to SPIx_DATA as soon as possible.

Step10: When the query SSN pin is low and the data to be sent has not been completed, jump to Step9.

Step11: Query and wait for the SSN pin to be pulled high, and the host releases the SPI slave.

Note:

- Whenever the SPI slave needs to send data, it is necessary to perform initialization and sending operations from Step2.

8.4.4 SPI slave receiver example

Step1: Map SSN/SCK/MISO/MOSI to the required pins according to the description of the digital multiplexing function of the pins in the GPIO chapter; and configure the SSN/SCK/MOSI pins to input mode and the MISO pin to output mode. For details on the source of the SSN pin, see the GPIO port auxiliary controller.

Step2: Set SPIx_CR.mstr to 0 to make the SPI work in slave mode.

Step3: Configure SPIx_CR.cpol and SPIx_CR.cpha to make the data frame format meet the application requirements.

Step4: Set SPIx_CR.spen to 1 to enable the SPI interface.

Step5: Query and wait for the SSN pin to be pulled low, and the master selects the SPI slave.

Step6: Query and wait for SPIx_STAT.rxne to become 1, and the data sent by the master has been received.

Step7: Read the received data from SPIx_DATA.

Step8: If the data to be received is not completed, jump to Step6 to continue execution.

Step9: Query and wait for the SSN pin to be pulled high, and the host releases the SPI slave.

8.5 SPI Register Description

Register list

SPI0 base address: 0x40000800

SPI1 base address: 0x40004800

Table 8-2 SPI register list

Offset	Register Name	Access	Register Description
0x00	SPIx_CR	RW	SPIx Configuration Register
0x04	SPIx_SSN	RW	SPIx Chip Select Configuration Register
0x08	SPIx_STAT	RO	SPIx Status Register
0x0c	SPIx_DATA	RW	SPIx Data Register
0x10	SPIx_CR2	RW	SPIx Configuration Register 2
0x14	SPIx_ICLR	WO	SPIx Interrupt Clear Register

8.5.1 SPI Configuration Register (SPIx_CR)

Address offset: 0x000

Reset value: 0x0000 0014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								spr2	spen	Res	mstr	cpol	cpha	spr1	spr0
								RW	RW		RW	RW	RW	RW	RW

Bit	Mark	Function Description																																				
31:8	Reserved																																					
7	spr2	Baud rate selection bit 2 Refer to spr0.																																				
6	spen	SPI module enable control 0 - Disable 1 - Enable																																				
5	Reserved																																					
4	mstr	SPI operation mode configuration 0 - Slave mode 1 - Master mode																																				
3	cpol	SCK line idle state configuration 0 - Low level 1 - High level																																				
2	cpha	Clock phase configuration 0 - First edge 1 - Second edge																																				
1	spr1	Baud rate selection bit 1 Refer to spr0																																				
0	spr0	Baud rate selection bit 0 Table 8-3 Master mode baud rate selection <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>spr2</th> <th>spr1</th> <th>spr0</th> <th>SCK Rate</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>PCLK /2</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>PCLK /4</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>PCLK /8</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>PCLK /16</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>PCLK /32</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>PCLK /64</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>PCLK /128</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>Reserved</td> </tr> </tbody> </table>	spr2	spr1	spr0	SCK Rate	0	0	0	PCLK /2	0	0	1	PCLK /4	0	1	0	PCLK /8	0	1	1	PCLK /16	1	0	0	PCLK /32	1	0	1	PCLK /64	1	1	0	PCLK /128	1	1	1	Reserved
spr2	spr1	spr0	SCK Rate																																			
0	0	0	PCLK /2																																			
0	0	1	PCLK /4																																			
0	1	0	PCLK /8																																			
0	1	1	PCLK /16																																			
1	0	0	PCLK /32																																			
1	0	1	PCLK /64																																			
1	1	0	PCLK /128																																			
1	1	1	Reserved																																			

8.5.2 SPI Chip Select Configuration Register (SPIx_SSN)

Address offset: 0x04

Reset value: 0x000000FF

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

Bit	Mark	Function Description
31:1	Reserved	
0	ssn	SPI_CS output level configuration in host mode 0: SPI_CS port outputs low level 1: SPI_CS port outputs high level

8.5.3 SPI Status Register (SPIx_STAT)

Address offset: 0x08

Reset value: 0x00000004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								spif	Res.	sserr	mdf	busy	txe	rxne	Res.
								RO	RO	RO	RO	RO	RO	RO	Res.

Bit	Mark	Function Description
31:8	Reserved	
7	spif	Receive completion flag 1: A byte has been received from the SPI bus 0: Receiving in progress
6	Reserved	
5	sserr	SSN error flag in slave mode
4	mdf	Conflict flag in master mode 1: SSN pin level is low 0: SSN pin level is high
3	busy	SPI bus transmission status flag 1: SPI bus is transmitting data 0: SPI bus is idle
2	txe	Transmit buffer status flag 1: Transmit buffer is empty 0: Transmit buffer is not empty
1	rxne	Receive buffer status flag 1: Receive buffer is not empty 0: Receive buffer is empty
0	Reserved	

8.5.4 SPI Data Register (SPIx_DATA)

Address offset: 0x0C

Reset value: 0x00000000

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

Bit	Mark	Function Description
31:8	Reserved	
7:0	spdat	<p>Data Register</p> <p>In transmit mode, write the byte to be sent to this register</p> <p>In receive mode, read the received byte from this register</p>

8.5.5 SPI Configuration Register 2 (SPIx_CR2)

Address offset: 0x10

Reset value: 0x0000 0003

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								rxne ie	txei e	hdma_tx	hdma_rx	int_en	Reserved		
								RW	RW	RW	RW	RW			

Bit	Mark	Function Description
31:7	Reserved	
6	rxneie	Receive buffer not empty interrupt enable 0 - Disable 1 - Enable
5	txeie	Transmit buffer empty interrupt enable 0 - Disable 1 - Enable
4	hdma_tx	DMA hardware access transmit enable. 0 - Disable 1 - Enable
3	hdma_rx	DMA hardware access receive enable. 0 - Disable 1 - Enable
2	int_en	SPI interrupt.enable. 0 - Disable 1 - Enable
1:0	Reserved	Please keep the values of these two bits both 1.

8.5.6 SPI Interrupt Clear Register 2 (SPIx_ICLR)

Address offset: 0x14

Reset value: 0x0000 00FF

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

Bit	Mark	Function Description
31:1	Reserved	
0	int_clr	SPI interrupt clear 0 - clear 1 - hold

9 Clock Calibration Module (CLKTRIM)

9.1 CLKTRIM Introduction

The CLKTRIM (Clock Trimming) module is a circuit dedicated to calibrating/monitoring clocks. In calibration mode, select an accurate clock source to calibrate an inaccurate clock source, calibrate repeatedly, and adjust the parameters of the inaccurate clock source until the frequency of the calibrated clock source meets the accuracy requirements. In calibration mode, the count value will have a certain error, but it is within the allowable accuracy error range. In monitoring mode, select a stable clock source to monitor the system working clock. In the set monitoring cycle, monitor whether the system working clock fails and generate an interrupt. In calibration mode and monitoring mode, the required clock sources must be initialized and enabled. For the specific configuration process, please refer to the [System Controller (SYSCTRL)] section.

9.2 CLKTRIM Main Features

CLKTRIM supports the following features:

- Calibration mode
- Monitoring mode
- 32-bit reference clock counter can load initial value
- 32-bit calibrated clock counter can configure overflow value
- 6 reference clock sources
- 6 calibrated clock sources
- Support interrupt mode

9.3 CLKTRIM Function Description

9.3.1 CLKTRIM Calibration Mode

9.3.1.1 Clock Calibration Principle

When calibrating a clock that is not accurate but has parameters to adjust the frequency (clock to be calibrated CAL_CLK), an accurate clock is required as the reference clock (REF_CLK). Set the calibration time Ttrim, the frequency of the clock to be calibrated Fcal, and the reference clock frequency Fref. Use the clock to be calibrated and the reference clock to count simultaneously, and stop counting at the end of the calibration time. Read the value M of the counter of the clock to be calibrated and the value N of the reference clock counter, and get the equation

$$T_{\text{trim}} = M/F_{\text{cal}} = N/F_{\text{ref}}$$

$$\text{Derived } F_{\text{cal}} = F_{\text{ref}} * M / N.$$

According to the formula, the smaller the frequency error rate of the reference clock, the smaller the error rate of the clock to be calibrated.

After calculating the frequency of the clock to be calibrated, if the error rate exceeds the range required by the system, you can adjust the parameters of the clock to be calibrated and repeat the above steps to calibrate again until the error rate of the clock frequency to be calibrated meets the system requirements.

9.3.1.2 Clock Calibration Module Hardware Structure

The time calibration module has 6 clock sources for the clock to be calibrated and 6 clock sources for the reference clock, as shown in Figure 9-1:

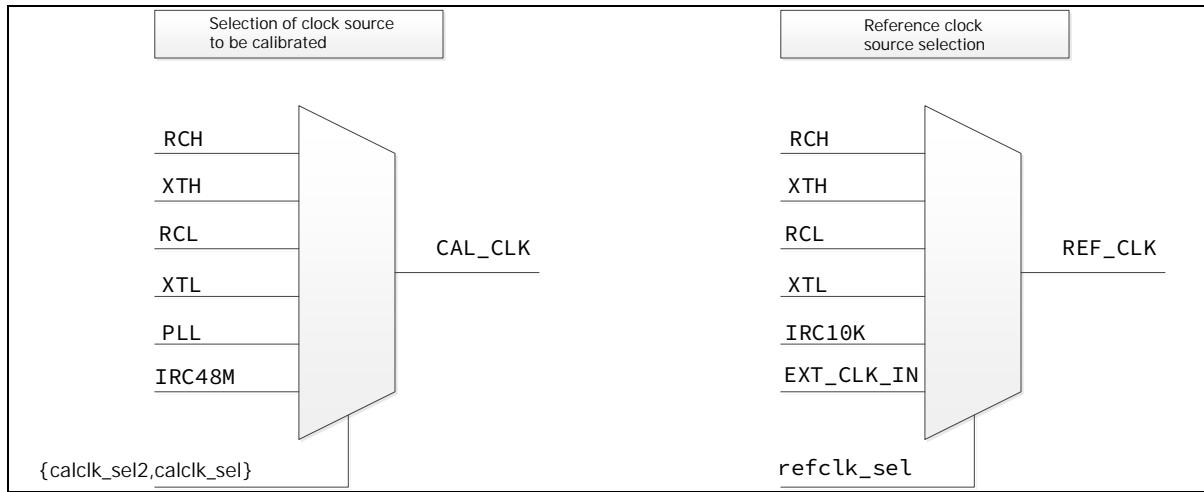


Figure 9-1 Clock source selection diagram

The selection of the clock to be calibrated is configured by registers CLKTRIM_CR.calclk_sel2 and CLKTRIM_CR.calclk_sel.

The selection of the reference clock is configured by register CLKTRIM_CR.refclk_sel.

As shown in Figure 9-2, the clock calibration module has two 32-bit counters, one of which is a down counter with a configurable initial value based on the reference clock. The initial value is configured by register CLKTRIM_REFCON.rcntval. The counter value can be read from register CLKTRIM_REFCNT. When the counter counts down to 0, it stops counting and generates an interrupt. One is an up counter with a configurable overflow value based on the clock to be calibrated. The overflow value is configured by register CLKTRIM_CALCON.ccntval. The counter value can be read from register CLKTRIM_CALCNT. When the counter counts up to the overflow value, it stops counting and generates an interrupt.

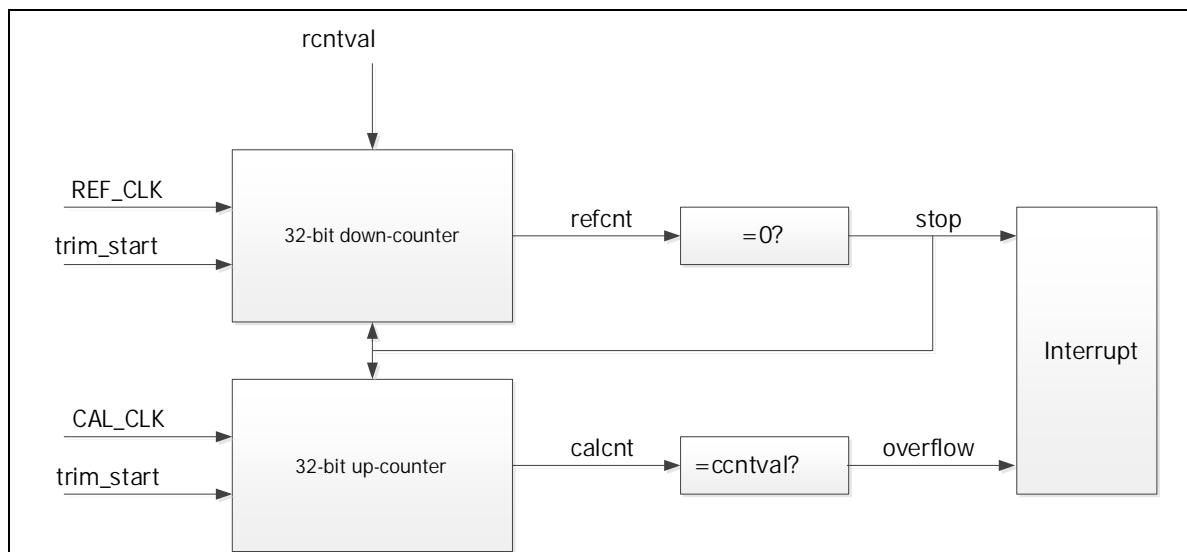


Figure 9-2 Clock calibration module hardware diagram

9.3.1.3 Clock Calibration Software Process

1. Set the CLKTRIM_CR.refclk_sel register to select the reference clock.
2. Set the CLKTRIM_CR.calclk_sel register to select the calibrated clock.
3. Set the CLKTRIM_REFCON.rcntval register to the calibration time.
4. Set the CLKTRIM_CR.IE register to enable interrupts.
5. Set the CLKTRIM_CR.trim_start register to start calibration.
6. The reference clock counter and the clock counter to be calibrated start counting.
7. When the reference clock counter counts down from the initial value to 0, CLKTRIM_IFR.stop is set to 1, triggering an interrupt.
8. The interrupt service subroutine determines that CLKTRIM_IFR.stop is 1, reads the values of registers CLKTRIM_REFCNT and CLKTRIM_CALCNT.
9. Clear the CLKTRIM_CR.trim_start register to end calibration.
10. Calculate the frequency of the clock to be calibrated according to the formula mentioned in the clock calibration principle section,

where M = the value of register CLKTRIM_CALCNT,

N = the value of register CLKTRIM_REFCON.rcntval

When the frequency of the clock to be calibrated does not meet the error rate requirements, adjust the parameters of the clock to be calibrated, re-execute steps 5 to 10 until the clock to be calibrated meets the error rate requirements.

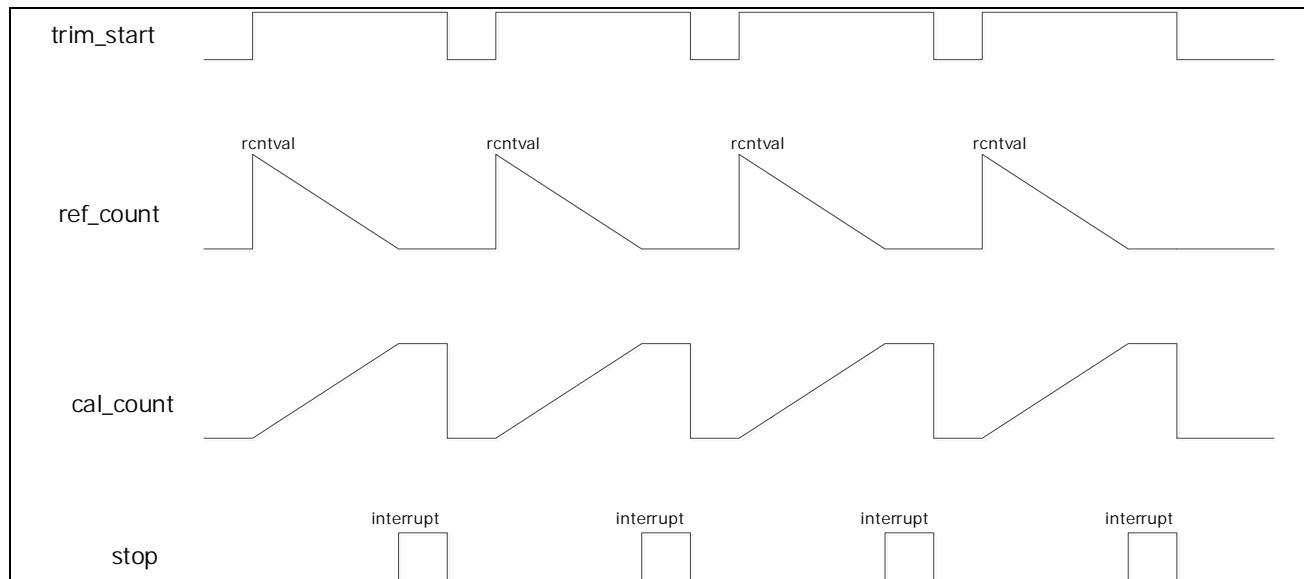


图 9-3 时钟校准波形示意图

Note:

- During the calibration process in calibration mode, the calibration time may be too long, causing the clock counter to be calibrated to overflow before CLKTRIM_IFR.stop is set to 1, and CLKTRIM_IFR.calcnt_of is set to 1, triggering an interrupt. When the interrupt service subroutine finds that CLKTRIM_IFR.calcnt_of is set to 1, it clears the CLKTRIM_CR.trim_start register to end the calibration.

In this case, the calibration cannot be performed correctly, and the calibration time must be adjusted and recalibrated.

The specific steps are:

1. Set the CLKTRIM_REFCON .rcntval register to adjust the calibration time.
2. Set the CLKTRIM_CR .trim_start register to restart the calibration.

9.3.2 CLKTRIM monitoring mode

9.3.2.1 Clock Monitoring Principle

In order to monitor whether the system working clock (monitored clock CAL_CLK) is working properly, a stable clock is required as the reference clock (REF_CLK). Set the monitoring time Ttrim, the monitored clock frequency Fcal, the reference clock frequency Fref, and the monitored clock counter overflow value CALVAL. Use the monitored clock and the reference clock to count at the same time. Stop counting at the end of the monitoring time to determine whether the monitored clock counter is in an overflow state. If the monitored clock counter has overflowed, it means that the monitored clock is working normally during the monitoring time, and continue to the next monitoring. If the monitored clock counter has not overflowed, it means that the monitored clock is working abnormally during the monitoring time. The monitored clock may have stopped or the frequency has changed significantly, resulting in an abnormal interrupt of the monitored clock, and the system working clock is switched by hardware.

9.3.2.2 Clock monitoring hardware structure

The monitored clock of the time calibration module has 3 clock sources and the reference clock has 6 clock sources, as shown in the following figure:

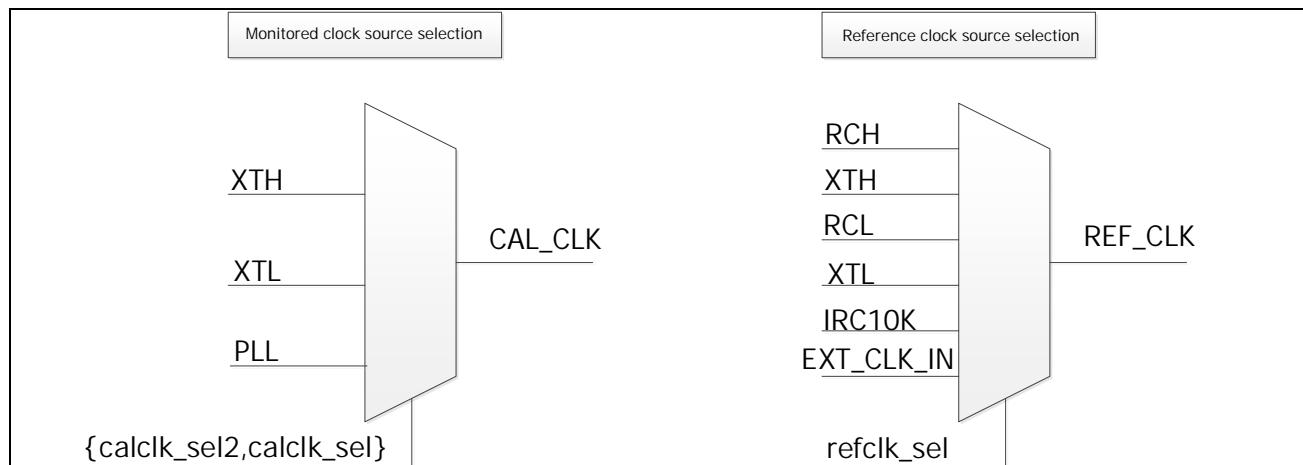


Figure 9-4 CLKTRIM clock selection

The selection of the monitored clock is configured by registers CLKTRIM_CR.calclk_sel2 and CLKTRIM_CR.calclk_sel, and the selection of the reference clock is configured by register CLKTRIM_CR.refclk_sel.

9.3.2.3 Clock Monitoring Software Process

1. Set the CLKTRIM_CR .refclk_sel register to select the reference clock.
2. Set the CLKTRIM_CR .calclk_sel register to select the monitored clock.
3. Set the CLKTRIM_REFCON .rcntval register to the monitoring interval time.
4. Set the CLKTRIM_CALCON. ccntval register to the monitored clock counter overflow time.
5. Set the CLKTRIM_CR .mon_en register to enable the monitoring function.
6. Set the CLKTRIM_CR .IE register to enable interrupts.
7. Set the CLKTRIM_CR .trim_start register to start monitoring.
8. The reference clock counter and the monitored clock counter start counting.
9. When the reference clock counter reaches the monitoring interval, CLKTRIM_IFR.stop is set to 1, and at the same time, it is determined whether the monitored clock counter overflows, that is, whether CLKTRIM_IFR.calcnt_of is set to 1. If it overflows, that is, CLKTRIM_IFR.calcnt_of is 1, indicating that the monitored clock is working normally. If it does not overflow, that is, CLKTRIM_IFR.calcnt_of is 0, it means that the monitored clock fails, LKTRIM_IFR.xtl_fault/xth_fault is set to 1, and an interrupt is triggered.
10. Process the interrupt service subroutine, clear the interrupt flag CLKTRIM_IFR.xtl_fault/xth_fault, and clear the CLKTRIM_CR.trim_start register to end monitoring.

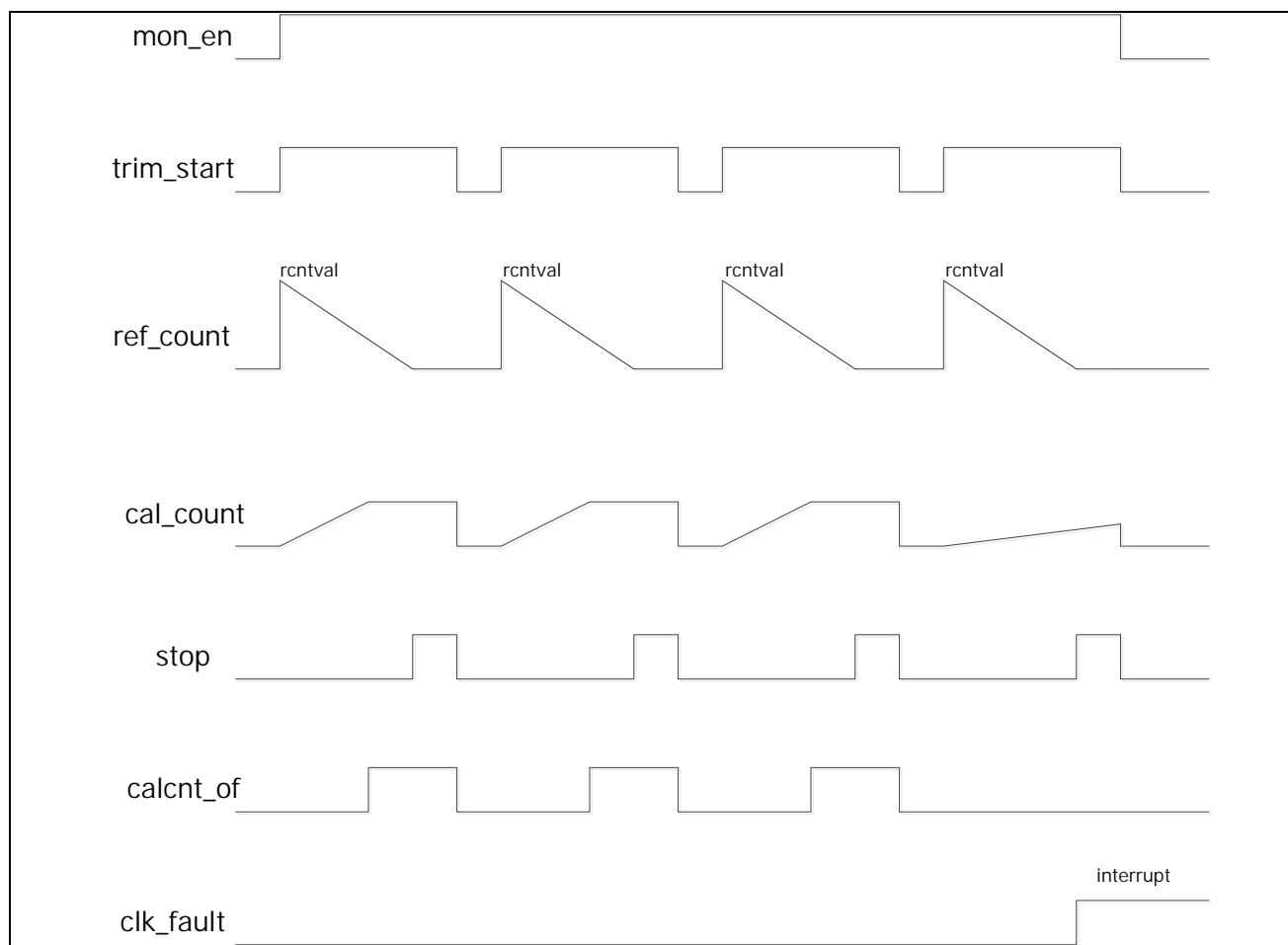


Figure 9-5 Clock monitoring waveform diagram

9.4 CLKTRIM Register Description

Register list

Base address: 0x40001800

Table 9-1 Register List

Offset	Register Name	Access	Register Description
0x00	CLKTRIM_CR	RW	Configuration register
0x04	CLKTRIM_REFCON	RW	Reference counter initial value configuration register
0x08	CLKTRIM_REFCNT	RO	Reference counter value register
0x0c	CLKTRIM_CALCNT	RO	Calibration counter value register
0x10	CLKTRIM_IFR	RO	Interrupt flag register
0x14	CLKTRIM_ICLR	RW	Interrupt flag clear register
0x18	CLKTRIM_CALCON	RW	Calibration counter overflow value configuration register

9.4.1 Configuration Register (CLKTRIM_CR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								calclk_sel2	IE	mon_en	calclk_sel	refclk_sel	trim_start		
								RW	RW	RW	RW	RW			RW

Bit	Mark	Function description														
31:9	Reserved															
8	calclk_sel2	Calibration/Monitoring Clock Select High Register														
7	IE	Interrupt Enable 0 – Disable 1 – Enable														
6	mon_en	Monitor Mode Enable 0 – Disable 1 – Enable														
5:4	calclk_sel	Clock selection for calibration/monitoring low bits <table border="1" style="margin-left: 20px;"> <tr> <td>calclk_sel2, calclk_sel</td> <td></td> </tr> <tr> <td>000</td> <td>RCH</td> </tr> <tr> <td>001</td> <td>XTH</td> </tr> <tr> <td>010</td> <td>RCL</td> </tr> <tr> <td>011</td> <td>XTL</td> </tr> <tr> <td>100</td> <td>PLL</td> </tr> <tr> <td>101</td> <td>IRC48M</td> </tr> </table>	calclk_sel2, calclk_sel		000	RCH	001	XTH	010	RCL	011	XTL	100	PLL	101	IRC48M
calclk_sel2, calclk_sel																
000	RCH															
001	XTH															
010	RCL															
011	XTL															
100	PLL															
101	IRC48M															
3:1	refclk_sel	Reference clock selection register 000 ---- RCH 001 ---- XTH 010 ---- RCL 011 ---- XTL 100 ---- IRC10K 101 ---- EXT_CLK_IN														
0	trim_start	Calibration/monitoring start register 0 – stop 1 – start														

9.4.2 Reference Counter Initial Value Configuration Register (CLKTRIM_REFCON)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:0	rcntval	Reference counter initial value

9.4.3 Reference Counter Value Register (CLKTRIM_REFCNT)

Address offset: 0x08

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:0	refcnt	Reference counter value

9.4.4 Calibration Counter Value Register (CLKTRIM_CALCNT)

Address offset: 0x0C

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:0	calcnt	Calibration counter value

9.4.5 Interrupt Flag Register (CLKTRIM_IFR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								RO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								pll_fault xth_fault xtl_fault calcnt_of stop							

Bit	Mark	Function description
31:5	Reserved	
4	pll_fault	PLL failure flag. CLKTRIM_ICLR pll_fault_clr Write zero to clear this flag
3	xth_fault	XTH failure flag. CLKTRIM_ICLR xth_fault_clr Write zero to clear this flag
2	xtl_fault	XTL failure flag. CLKTRIM_ICLR xtl_fault_clr Write zero to clear this flag
1	calcnt_of	Calibration counter overflow flag. CLKTRIM_CR.start Write zero to clear this flag
0	stop	Reference counter stop flag. CLKTRIM_CR.start Write zero to clear this flag

9.4.6 Interrupt Flag Clear Register (CLKTRIM_ICLR)

Address offset: 0x14

Reset value: 0x1111 1111

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										pll_fault_clr	xth_fault_clr	xtl_fault_clr	Reserved		
										R1W 0	R1W 0	R1W 0			

Bit	Mark	Function description
31:5	Reserved	
4	pll_fault_clr	Clears the PLL fault flag. Write zero to clear.
3	xth_fault_clr	Clears the XTH fault flag. Write zero to clear.
2	xtl_fault_clr	Clears the XTL fault flag. Write zero to clear.
1:0	Reserved	

9.4.7 Calibration Counter Overflow Value Configuration Register (CLKTRIM_CALCON)

Address offset: 0x18

Reset value: 0xffff ffffff

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

Bit	Mark	Function description
31:0	ccntval	Calibrate counter overflow value

10 Hardware Divider Module (HDIV)

10.1 HDIV Introduction

HDIV (Hardware Divider) is a 32-bit signed/unsigned integer hardware divider.

10.2 HDIV Main Features

The HDIV hardware divider supports the following features:

- Configurable signed/unsigned integer division calculation
- 32-bit dividend, 16-bit divisor
- Output 32-bit quotient and 32-bit remainder
- Zero divisor warning flag, division operation end flag
- 10 clock cycles to complete a division operation
- Writing the divisor register triggers the start of the division operation
- Automatically wait for the calculation to end when reading the quotient register/remainder register

10.3 HDIV Functional Description

10.3.1 HDIV operation process

1. Open the clock enable register of the hardware divider in the system controller.
2. Configure register HDIV_SIGN to set signed/unsigned division operation.
3. Configure register HDIV_DIVIDEND to set the dividend.
4. Configure register HDIV_DIVISOR to set the divisor.
5. The division operation starts, query register HDIV_STAT to check the operation end flag bit div_end, and div_end is 1 to mark the end of the operation. Read register HDIV_QUOTIENT to get the quotient, and read register HDIV_REMAINDER to get the remainder.
6. When the divisor is zero, the division operation ends immediately, and the operation result remains the result of the previous operation. At the same time, the divisor is zero warning flag bit div_zero is set.
7. Before the division operation ends, when reading registers HDIV_QUOTIENT/HDIV_REMAINDER, the CPU will be held until the operation ends.

Example: Calculate an unsigned division, the dividend is 1917887483 (0x7250A3FB), and the divisor is 9597 (0x257D)

Step 1: Set the register HDIV_SIGN to 0, i.e., unsigned division operation

Step 2: Set the register HDIV_DIVIDEND to 0x7250A3FB, i.e., set the dividend

Step 3: Set the register HDIV_DIVISOR to 0x257D, i.e., set the divisor and start the calculation

Step 4: Query the register HDIV_STAT operation end flag div_end, div_end is 1, and the operation is over.

Read the register HDIV_QUOTIENT to get the quotient 199842 (0x30CA2)

Read the register HDIV_REMAINDER to get the remainder 3809 (0xEE1)

10.4 HDIV Register Description

Register List

Base Address: 0x40021800

Table 10-1 Register List

Offset	Register Name	Access	Register Description
0x00	HDIV_DIVIDEND	RW	Dividend register
0x04	HDIV_DIVISOR	RW	Divisor register
0x08	HDIV_QUOTIENT	RO	Quotient register
0x0c	HDIV_REMAINDER	RO	Remainder register
0x10	HDIV_SIGN	RW	Sign register
0x14	HDIV_STAT	RO	Status register

10.4.1 Dividend Register (HDIV_DIVIDEND)

Address offset: 0x00

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:0	DIVIDEND	Dividend value register

10.4.2 Divisor Register (HDIV_DIVISOR)

Address offset: 0x04

Reset value: 0x0000 0001

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

Bit	Mark	Function description
31:16	Reserved	
15:0	DIVISOR	Divisor value register (writing this register automatically triggers the division operation)

10.4.3 Quotient Register (HDIV_QUOTIENT)

Address offset: 0x08

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:0	QUOTIENT	Quotient result register

10.4.4 Remainder Register (HDIV_REMAINDER)

Address offset: 0x0C

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:0	REMAINDER	Remainder result register

10.4.5 Sign Register (HDIV_SIGN)

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:1	Reserved	
0	sign	Sign selection register. 0 --- unsigned division operation 1 --- signed division operation

10.4.6 Status Register (HDIV_STAT)

Address offset: 0x14

Reset value: 0x0000 0001

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

Bit	Mark	Function description
31:2	Reserved	
1	div_zero	The divisor is zero warning flag. 0 --- The divisor is not zero 1 --- The divisor is zero
0	div_end	The division operation ends flag. 0 --- The operation is in progress 1 --- The operation ends

11 FLASH controller (FLASH)

11.1 Overview

This system contains a 16k/32k/64k/128k byte FLASH memory, which is divided into 32/64/128/256 sectors, and each sector has a capacity of 512 bytes. The FLASH controller supports erase, program, and read operations on the FLASH memory. This controller also supports erase and write protection of the FLASH memory, as well as write protection of the control register.

11.2 FLASH Capacity Division

Table 11-1 FLASH capacity division

Address	Sector number		Address	Sector number
0x0E00 - 0x0FFF	Sector7	0x1FE00 - 0x1FFFF	Sector255
0x0C00 - 0x0DFF	Sector6	0x1FC00 - 0x1FDFF	Sector254
0x0A00 - 0x0BFF	Sector5	0x1FA00 - 0x1FBFF	Sector253
0x0800 - 0x09FF	Sector4	0x1F800 - 0x1F9FF	Sector252
0x0600 - 0x07FF	Sector3	0x1F600 - 0x1F7FF	Sector251
0x0400 - 0x05FF	Sector2	0x1F400 - 0x1F5FF	Sector250
0x0200 - 0x03FF	Sector1	0x1F200 - 0x1F3FF	Sector249
0x0000 - 0x01FF	Sector0	0x1F000 - 0x1F1FF	Sector248

Note:

- 16KByte FLASH consists of Sector0-32;
- 32KByte FLASH consists of Sector0-63;
- 64KByte FLASH consists of Sector0-127;
- 128KByte FLASH consists of Sector0-255.

11.3 Functional Description

This controller supports data read and write operations on FLASH with three bit widths: Byte (8 bits), half-word (16 bits) and word (32 bits). Note that the target address of Byte operation must be byte-aligned, the target address of half-word operation must be half-word-aligned (the lowest address bit is 1'b0), and the address of word operation must be word-aligned (the lowest two bits of the address are 2'b00). If the target address is not aligned according to the bit width, the operation is invalid and the CPU will enter the Hard Fault error interrupt.

This controller adopts a high-security hardware design and has a FLASH operation source defense function: only when the address of the FLASH operation function is located in 0~32K, can the FLASH erase and write operation be performed correctly.

The FLASH address 0~32K has higher security, and important functions must be placed in this area; for example, important program entry, interrupt entry function, high-security algorithm module, UID, AES, true random number, RTC algorithm cooperation, to form a high-security authentication system.

11.3.1 Sector Erase

Page erase can erase a page (Sector) specified by the user each time. After the erase operation is completed, the data in the page (Sector) is 0xFF. If the erase operation is executed from FLASH, the CPU will stop fetching instructions, and the hardware will automatically wait for the operation to complete (FLASH_CR. BUSY becomes 0); if the erase operation is executed from RAM, the CPU will not stop fetching instructions, and the user software should wait for the operation to complete (FLASH_CR. BUSY becomes 0).

The steps for sector erase operation are as follows:

Step 1: Configure FLASH erase parameters, see Section 11.4 for details.

Step 2: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step 3: Configure FLASH_CR. OP to 2 and set the Flash operation mode to sector erase.

Step 4: Check whether FLASH_CR. OP is 2. If not, jump to Step 2.

Step5: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step6: Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection of the sector.

Step7: Check whether the corresponding bit of FLASH_SLOCK is 1. If not, jump to Step5.

Step8: Write any data to any address in the sector to be erased to trigger sector erase.

Example: *((unsigned char *)0x00000200) = 0x00.

Step9: Wait for FLASH_CR. BUSY to become 0, and the sector erase operation is completed.

Step10: If you need to erase other sectors, repeat Step5 – Step9.

Note:

- The address of the code for erasing the FLASH page must be less than 32768.

11.3.2 Chip Erase

Full chip erase can erase all pages (Sector) at one time. After the erase operation is completed, the data in all pages (Sector) is 0xFF. If the erase operation is executed from FLASH, the operation will be prohibited. Because this operation will erase the program segment where the current PC is located. If this happens, the error flag will be set; if the erase operation is executed from RAM, the CPU will not stop fetching instructions, and the user software should wait for the operation to complete (FLASH_CR. BUSY becomes 0).

The chip erase operation steps are as follows:

Step 1: Configure FLASH erase parameters, see Section 11.4 for details.

Step 2: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step 3: Configure FLASH_CR. OP to 3 and set the Flash operation mode to chip erase.

Step4: Check if FLASH_CR. OP is 3, if not, jump to Step2.

Step5: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step6: Set FLASH_SLOCK0 to 0xFFFFFFFF to remove the erase and write protection of all sectors.

Step7: Check if FLASH_SLOCK0 is 0xFFFFFFFF, if not, jump to Step5.

Step8: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step9: Set FLASH_SLOCK1 to 0xFFFFFFFF to remove the erase and write protection of all sectors.

Step10: Check if FLASH_SLOCK1 is 0xFFFFFFFF. If not, jump to Step8.

Step11: Write to any address in the chip to be erased to trigger chip erase.

Example: *((unsigned char *)0x00000000) = 0x00.

Step12: Wait for FLASH_CR. BUSY to become 0, and the chip erase operation is completed.

11.3.3 Write Operation (Program)

The write operation can only change the bit data in FLASH from 1 to 0, so before writing data, make sure that the data in the address to be written is 0xFF.

Supports writing 3 data lengths: Byte (8 bits), Half-word (16 bits), Word (32 bits), and the written data is stored in FLASH in little-endian mode, that is, the low address stores the low byte of the data. If the write operation is executed from FLASH, the CPU will stop fetching instructions, and the hardware will automatically wait for the operation to complete (FLASH_CR. BUSY becomes 0); if the write operation is executed from RAM, the CPU will not stop fetching instructions, and the user software should wait for the operation to complete (FLASH_CR. BUSY becomes 0).

The steps for byte write operation are as follows:

Step 1: Configure FLASH erase parameters, see Section 11.4 for details.

Step 2: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step3: Configure FLASH_CR. OP to 1 and set the Flash operation mode to write.

Step4: Check if FLASH_CR. OP is 1. If not, jump to Step2.

Step5: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step6: Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase and write protection.

Step7: Check if the corresponding bit of FLASH_SLOCK is 1. If not, jump to Step5.

Step8: Perform Byte write operation on the target address to be written to trigger the write operation.

Example: *((unsigned char *)0x00001231) = 0x5A.

Step9: Wait for FLASH_CR. BUSY to become 0, and the write operation is completed.

Step10: If you need to write Byte to other addresses, repeat Step8 - Step9.

The steps of half-word write operation are as follows:

Step1: Configure FLASH erase and write time, see Section 11.4 for details.

Step2: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step3: Configure FLASH_CR. OP to 1 and set the Flash operation mode to write.

Step4: Check whether FLASH_CR. OP is 1. If not, jump to Step2.

Step5: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step6: Set the corresponding bit of FLASH_SLOCK to 1 to remove erase and write protection.

Step7: Check whether the corresponding bit of FLASH_SLOCK is 1. If not, jump to Step5.

Step8: Perform half-word write operation on the target address to be written to trigger the write operation.

Example: *((unsigned short int *)0x00001232) = 0xABCD.

Step9: Wait for FLASH_CR. BUSY to become 0, and the write operation is completed.

Step10: If you need to write Half-word to other addresses, repeat Step8 - Step9.

The steps of Word write operation are as follows:

Step1: Configure FLASH erase and write parameters, see Section 11.4 for details.

Step2: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step3: Configure FLASH_CR. OP to 1 and set the Flash operation mode to write.

Step4: Check whether FLASH_CR. OP is 1. If not, jump to Step2.

Step5: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step6: Set the corresponding bit of FLASH_SLOCK to 1 to remove erase and write protection.

Step7: Check whether the corresponding bit of FLASH_SLOCK is 1. If not, jump to Step5.

Step8: Perform Word write operation on the target address to be written to trigger the write operation.

Example: *((unsigned int *)0x00001234) = 0x55667788.

Step9: Wait for FLASH_CR. BUSY to become 0, and the write operation is completed.

Step10: If you need to write Word to other addresses, repeat Step8 - Step9.

Note:

- The address of the code for writing to FLASH must be less than 32768.

11.3.4 Read Operation

Supports reading 3 data lengths: Byte (8 bits), Half-word (16 bits), Word (32 bits), and the read data is in little-endian mode, that is, the low address stores the low byte of the data. The read operation does not require any operation steps, and the data in FLASH can be read at any time.

Example:

Byte read operation: temp = *((unsigned char *)0x00001231)

Half-word read operation: temp = *((unsigned short int *)0x00001232)

Word read operation: temp = *((unsigned int *)0x00001234)

11.4 Erase and Write Time

FLASH memory has strict timing requirements for the control signals of erase and programming operations. If the timing of the control signals is not qualified, the erase and programming operations will fail. When powered on, the erase and programming parameters when HCLK is 4MHz are loaded by default; if the HCLK frequency when erasing Flash is not 4MHz, the user program should load the erase and programming parameters corresponding to the HCLK frequency. When operating FLASH, the highest supported HCLK is 48MHz.

The registers related to the erase and programming timing parameters are: FLASH_TNVS, FLASH_TPGS, FLASH_TPROG, FLASH_TSERASE, FLASH_TMERASE, FLASH_TPRCV, FLASH_TSRCV, FLASH_TMRCV. If the HCLK is increased from the default 4MHz to 8MHz, the value of the above FLASH_Tx register should be set to twice the default value, that is, keep the current Tsysclk*FLASH_Tx result equal to the default value.

The following table shows the corresponding FLASH erase time parameters at different frequencies:

Table 11-2 FLASH erase time parameters at different frequencies

Parameter	4M	8M	16M	24M	32M	48M
TNVS	0x20	0x40	0x80	0xC0	0x100	0x180
TPGS	0x17	0x2E	0x5C	0x8A	0xB8	0xFF
TPROG	0x1B	0x36	0x6C	0xA2	0xD8	0x144
TSERASE	0x4650	0x8CA0	0x11940	0x1A5E0	0x23280	0x34BC0
TMERASE	0x222E0	0x445C0	0x88B80	0xCD140	0x111700	0x19A280
TPRCV	0x18	0x30	0x60	0x90	0xC0	0x120
TSRCV	0xF0	0x1E0	0x3C0	0x5A0	0x780	0xB40
TMRCV	0x3E8	0x7D0	0xFA0	0x1770	0x1F40	0x2EE0

The operation steps for erase and write parameters when the system frequency is configured to 8MHz are as follows:

- Step1: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.
- Step2: Write 0x40 to the FLASH_TNVS register. If the read register value is not 0x40, jump to the previous step.
- Step3: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.
- Step4: Write 0x2E to the FLASH_TPGS register. If the read register value is not 0x2E, jump to the previous step.
- Step5: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.
- Step6: Write 0x36 to the FLASH_TPROG register. If the read register value is not 0x36, jump to the previous step.
- Step7: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.
- Step8: Write 0x8CA0 to the FLASH_TSERASE register. If the read register value is not 0x8CA0, jump to the previous step.
- Step9: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.
- Step10: Write 0x445C0 to the FLASH_TMERASE register. If the read register value is not 0x445C0, jump to the previous step.
- Step11: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step12: Write 0x30 to the FLASH_TPRCV register. If the read register value is not 0x30, jump to the previous step.

Step13: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step14: Write 0x1E0 to the FLASH_TSRCV register. If the read register value is not 0x1E0, jump to the previous step.

Step15: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewrite.

Step16: Write 0x7D0 to the FLASH_TMRCV register. If the read register value is not 0x7D0, jump to the previous step.

11.5 Read Wait Cycles

The fastest instruction fetch frequency supported by the built-in FLASH of this device is 24MHz. When the HCLK frequency exceeds 24MHz and is less than 48MHz, a waiting cycle must be inserted for the FLASH read time, that is, FLASH_CR.WAIT is set to 1. When the waiting cycle is inserted, the FLASH will complete a read operation every two cycles.

11.6 Erase and write protection

11.6.1 Erase and write protection bits

The entire 16k/32k/64k/128k byte FLASH memory is divided into 32/64/128/256 pages, and every 4 pages share an erase protection bit. When a page is protected, the erase and write operations on the page are invalid and an alarm flag and interrupt signal are generated. When any page in the FLASH memory is protected, the full chip erase and write of the FLASH is invalid, and an alarm flag and interrupt signal are generated.

11.6.2 PC address erase and write protection

When the CPU runs a program in FLASH, if the current PC pointer happens to fall within the address range of the page to be erased, the erase operation is invalid and an alarm flag and interrupt signal are generated.

11.7 Write Protection Register

The important controllers of this module shield the normal write operation and must be modified by the write sequence method.

The registers that need to be modified by the write sequence method are as follows:

FLASH_TNVS, FLASH_TPGS, FLASH_TPROG, FLASH_TSERASE, FLASH_TMERASE, FLASH_TPRCV,
FLASH_TSRCV, FLASH_TMRCV, FLASH_CR, FLASH_SLOCK0, FLASH_SLOCK1.

The registers that can be modified without the write sequence method are as follows:

FLASH_ICLR, FLASH_BYPASS.

The specific steps to modify the register value by the write sequence method are as follows:

Step1: Write 0x5A5A to the FLASH_BYPASS register.

Step2: Write 0xA5A5 to the FLASH_BYPASS register.

Step3: Write the target value to the register to be modified.

Step4: Verify whether the current value of the register to be modified is the same as the target, if not, jump to Step1.

Step5: Perform other operations.

Note:

- Write 0x5a5a, 0xa5a5, and write the target register. Do not insert any write operations (write ROM, RAM, REG) between these three write operations, otherwise the value of the target register cannot be rewritten. If the rewrite fails, you need to repeat these three steps.

11.8 Registers

Base address: 0x4002 0000

Register	Offset Address	Description
FLASH_TNVS	0x00	Tnvs timing parameter
FLASH_TPGS	0x04	Tpgs timing parameter
FLASH_TPROG	0x08	Tprog timing parameter
FLASH_TSERASE	0x0C	Tserase timing parameter
FLASH_TMERASE	0x10	Tmerase timing parameter
FLASH_TPRCV	0x14	Tprcv timing parameter
FLASH_TSRCV	0x18	Tsrcv timing parameter
FLASH_TMRCV	0x1C	Tmrcv timing parameter
FLASH_CR	0x20	Control register
FLASH_IFR	0x24	Interrupt flag register
FLASH_ICLR	0x28	Interrupt flag clear register
FLASH_BYPASS	0x2C	0x5a5a-0xa5a5 Bypass sequence register
FLASH_SLOCK0	0x30	Sector0-127 erase protection register
FLASH_SLOCK1	0x34	Sector128-255 erase protection register

11.8.1 TNVS parameter register (FLASH_TNVS)

Offset address: 0x00

Reset value: 0x0000 0020

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

Bit	Mark	Function description
31:9	Reserved	
8:0	TNVS	Calculation formula: TNVS = 8*HCLK, where HCLK is in MHz. Example for 4MHz: TNVS = 8*4 = 32.

11.8.2 TPGS parameter register (FLASH_TPGS)

Offset address: 0x04

Reset value: 0x0000 0017

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

Bit	Mark	Function description
31:8	Reserved	
7:0	TPGS	Calculation formula: TPGS = 5.75*HCLK, HCLK is in MHz. 4MHz example: TPGS = 5.75*4 = 23. <i>Note: When the calculated value is greater than 0xFF, TPGS should be assigned 0xFF.</i>

11.8.3 TPROG parameter register (FLASH_TPROG)

Offset address: 0x08

Reset value: 0x0000 001B

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

Bit	Mark	Function description
31:9	Reserved	
8:0	TPROG	Calculation formula: TPROG = 6.75*HCLK, HCLK is in MHz. 4MHz example: TPROG = 6.75*4 = 27.

11.8.4 TSERASE Register (FLASH_TSERASE)

Offset address: 0x0C

Reset value: 0x0000 4650

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

Bit	Mark	Function description
31:18	Reserved	
17:0	TSERASE	Calculation formula: TSERASE = 4500*HCLK, HCLK is in MHz. 4MHz example: TSERASE = 4500*4 = 18000.

11.8.5 TMERASE parameter register (FLASH_TMERASE)

Offset address: 0x10

Reset value: 0x0002 22E0

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

Bit	Mark	Function description
31:21	Reserved	
20:0	TMERASE	Calculation formula: TMERASE = 35000*HCLK, HCLK is in MHz. 4MHz example: TMERASE = 35000*4 = 140000.

11.8.6 TPRCV parameter register (FLASH_TPRCV)

Offset address: 0x14

Reset value: 0x0000 0018

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

Bit	Mark	Function description
31:12	Reserved	
11:0	TPRCV	Calculation formula: TPRCV = 6*HCLK, HCLK is in MHz. 4MHz example: TPRCV = 6*4 = 24.

11.8.7 TSRCV parameter register (FLASH_TSRCV)

Offset address: 0x18

Reset value: 0x0000 00F0

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

Bit	Mark	Function description
31:12	Reserved	
11:0	TSRCV	Calculation formula: TSRCV = 60*HCLK, HCLK is in MHz. 4MHz example: TSRCV = 60*4 = 240.

11.8.8 TMRCV parameter register (FLASH_TMRCV)

Offset address: 0x1C

Reset value: 0x0000 03E8

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

Bit	Mark	Function description
31:14	Reserved	
13:0	TMRCV	Calculation formula: TMRCV = 250*HCLK, HCLK is in MHz. 4MHz example: TMRCV = 250*4 = 1000.

11.8.9 CR Register (FLASH_CR)

Offset address: 0x20

Reset value: 0x0000 0200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						DPS TB_E N	Reserved		IE	BUS Y	WAIT	OP			
						RW			RW	RO	RW	RW			

Bit	Mark	Function description
31:10	Reserved	
9	DPSTB_EN	FLASH dpstb enable Mask bit 0: When the system enters deepsleep mode, FLASH does not enter low power mode 1: When the system enters deepsleep mode, FLASH enters low power mode
8:7	Reserved	
6:5	IE	IE[6]: FLASH erase protected address interrupt enable; 0: Disable; 1: Enable IE[5]: FLASH erase PC value interrupt enable; 0: Disable; 1: Enable
4	BUSY	Idle/busy flag; 0: Idle state; 1: Busy state;
3:2	WAIT	FLASH read cycles 0~24MHz: 00/11, 1 cycle 24~48MHz: 01: 2 cycles 48~72MHz: 10: 3 cycles; (The highest clock of this series of products is 48MHz)
1:0	OP	FLASH operation 00: Read; 01: Write; 10: Page erase; 11: Full chip erase

11.8.10 IFR Register (FLASH_IFR)

Offset address: 0x24

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:2	Reserved	
1	IF1	Erase protection alarm interrupt flag
0	IF0	Erase PC address alarm interrupt flag

11.8.11 ICLR Register (FLASH_ICLR)

Offset address: 0x28

Reset value: 0x0000 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
												ICLR 1	ICLR 0		
												R1W 0	R1W 0		

Bit	Mark	Function description
31:4	Reserved	
3:2	Reserved	Write invalid, read as 0x3
1	ICLR1	Clear protection alarm interrupt flag; write 0 to clear; write 1 is invalid
0	ICLR0	0 ICLRO Clear PC address alarm interrupt flag; write 0 to clear; write 1 is invalid

11.8.12 BYPASS Register (FLASH_BYPASS)

Offset address: 0x2C

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:16	Reserved	
15:0	BYSEQ	Before modifying the registers of this module, you must write the 0x5a5a-0xa5a5 sequence to the BYSEQ[15:0] register. Each time you write the Bypass sequence, you can only modify the register once. If you need to modify the register again, you must enter the 0x5a5a-0xa5a5 sequence again.

11.8.13 SLOCK0 Register (FLASH_SLOCK0)

Offset address: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLOCK0[31:16]															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLOCK0[15:0]															RW

Bit	Mark	Function description
31:0	SLOCK	Sector Erase/Write Protection Bit; 0: Erase/Write Not Allowed; 1: Erase/Write Allowed Bit [0] corresponds to: Sector0-1-2-3 Bit [1] corresponds to: Sector4-5-6-7 Bit [2] corresponds to: Sector8-9-10-11 Bit [3] corresponds to: Sector12-13-14-15 Bit [31] corresponds to: Sector124-125-126-127

11.8.14 SLOCK1 Register (FLASH_SLOCK1)

Offset address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLOCK1[31:16]															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLOCK1[15:0]															RW

Bit	Mark	Function description
31:0	SLOCK	Sector Erase/Write Protection Bit; 0: Erase/Write Not Allowed; 1: Erase/Write Allowed Bit [0] corresponds to: Sector128-129-130-131 Bit [1] corresponds to: Sector132-133-134-135 Bit [2] corresponds to: Sector136-137-138-139 Bit [3] corresponds to: Sector140-141-142-143 Bit [31] Corresponding: Sector 252-253-254-255

12 RAM controller (RAM)

12.1 Overview

This system contains a 2k/4k/8k/16k byte SRAM, which supports byte (8 bits), half word (16 bits), and word (32 bits) read and write operations. Read and write operations can be performed at the system clock frequency without waiting for a cycle. In addition, this controller also supports parity check, which can perform parity check on each byte of SRAM data and generate a parity check error interrupt.

12.2 Functional Description

12.2.1 RAM Address Range

The address range of RAM in the system mapping is shown in the following table:

Table 12-1 RAM address mapping

Address range	Size	Memory type
0x2000_0000 - 0x2000_3FFF	16KByte	SRAM

12.2.2 Read and write bit width

This controller supports three bit width read and write operations: byte (8 bits), half word (16 bits), and word (32 bits). The address of byte operation must be byte aligned, the target address of half word operation must be half word aligned (the lowest address bit is 1'b0), and the address of word operation must be word aligned (the lowest two addresses are 2'b00). If the target address of the read and write operation is not aligned according to the bit width, the operation is invalid and the system will generate a Hard Fault error interrupt.

12.2.3 Parity Check

This controller supports parity check of SRAM data. When writing data to SRAM, a parity check is performed on each byte of data, and the 1-bit check value is stored in SRAM together with the 8-bit data. When reading data from SRAM, the controller reads the 8-bit data and the 1-bit check value, and performs a parity check. If the check fails, the parity check error flag is set, and an error interrupt is generated when the interrupt is enabled.

Note:

- When parity check is enabled, the SRAM must be initialized before reading SRAM data, otherwise the parity check alarm flag or interrupt may be triggered by mistake.

12.3 Registers

Base address: 0x4002 0400

Table 12-2 Registers

Register	Offset address	Description
RAM_CR	0x00	Control register
RAM_ERRADDR	0x04	Error address register
RAM_IFR	0x08	Error interrupt flag register
RAM_ICLR	0x0C	Error interrupt flag clear register

12.3.1 Control Register (RAM_CR)

Offset address: 0x00

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:2	Reserved	Reserved
1	IE	Error alarm interrupt enable signal; 1: Enable alarm interrupt, 0: Disable alarm interrupt
0	Reserved	Reserved

12.3.2 Parity Error Address Register (RAM_ERRADDR)

Offset address: 0x04

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:14	Reserved	
13:0	ERRADDR	Parity error Byte address; after the interrupt flag is cleared, the address is cleared at the same time.

12.3.3 Error Interrupt Flag Register (RAM_IFR)

Offset address: 0x08

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:1	Reserved	
0	ERR	Parity error flag

12.3.4 Error Interrupt Flag Clear Register (RAM_ICLR)

Offset address: 0x0C

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ERR CLR R1W 0

Bit	Mark	Function description
31:1	Reserved	
0	ERRCLR	Error interrupt flag clear bit; write 1: invalid, write 0: clear

13 DMA controller (DMAC)

13.1 DMAC Introduction

Direct memory access (DMA) is used to provide high-speed data transfer between peripherals and memory or between memory and memory; the CPU does not need to participate in the transfer process; so the CPU can perform other operations simultaneously. DMAC has two independent DMA channels, each channel is dedicated to managing requests from peripherals or memory access. There is also an arbitrator to coordinate the priority of each DMA request.

13.2 DMAC Main Features

- 2 independent DMA channels, support priority configuration
- 3 data transfer widths: 8-bit, 16-bit, 32-bit
- 4 transfer modes: software block, software burst, hardware block, hardware burst
- 2 source address types: peripherals, memory
- 2 target address types: peripherals, memory
- 2 address change modes: fixed, auto-increment
- Transfer address addressing range: 0x00000000 ~ 0xFFFFFFFF
- Configurable number of data blocks to be transferred: 1~65536
- Configurable size of data blocks to be transferred: 1~16
- Support address and transfer number reload function

13.3 Functional Block Diagram

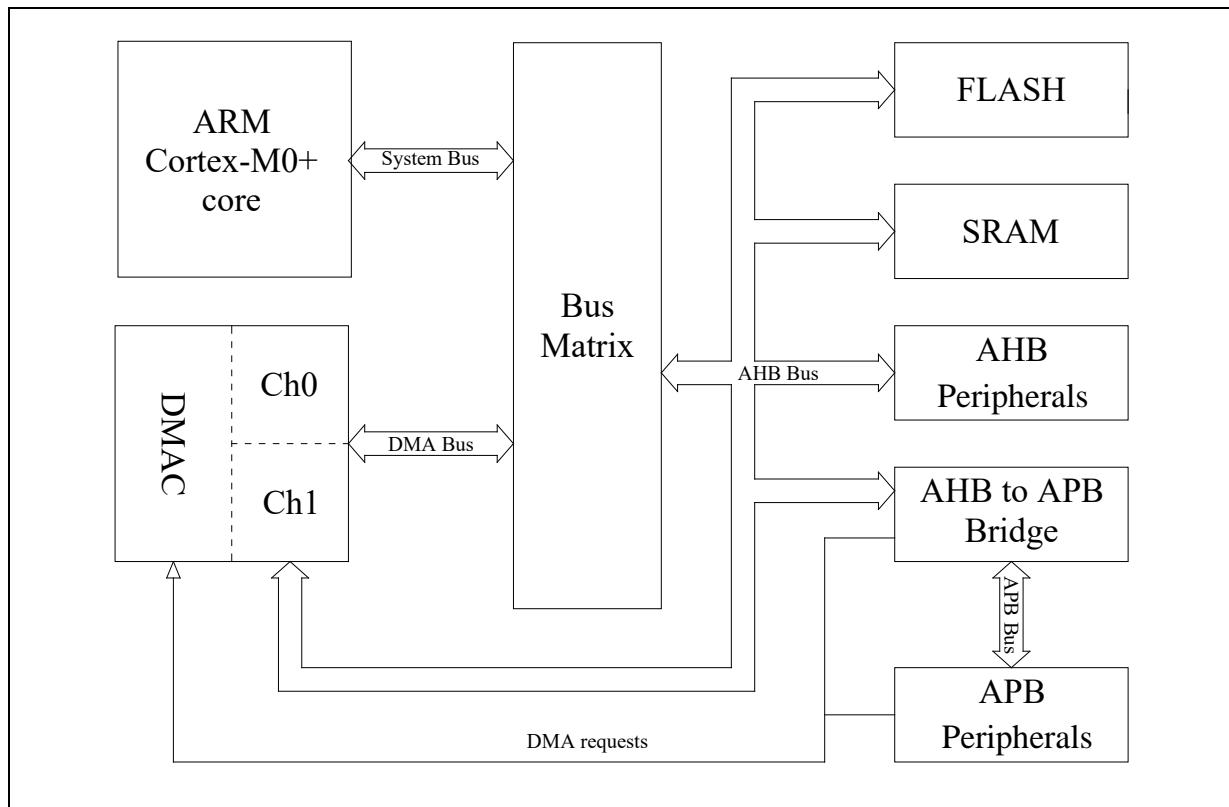


图 13-1 功能框图

- **DMAC**
DMAC has two DMA channels. When conflicts occur between channels, the priority controller will arbitrate.
- **Bus Matrix**
Both the CPU and DMAC are connected to the bus matrix. When the CPU releases the AHB bus, the DMAC can access the AHB bus devices and APB bus devices through the Bus Matrix. The CPU has a higher priority than the DMAC in accessing the bus.
- **DMA requests**
If the peripheral supports DMA request, the DMA channel can be configured as hardware trigger, otherwise it can only be configured as software trigger.

13.4 Functional Description

13.4.1 DMAC Transfer Modes Overview

This DMA controller supports 4 transfer modes: software block transfer mode, software burst transfer mode, hardware block transfer mode, and hardware burst transfer mode.

The comparison of the four transmission modes is shown in the table below:

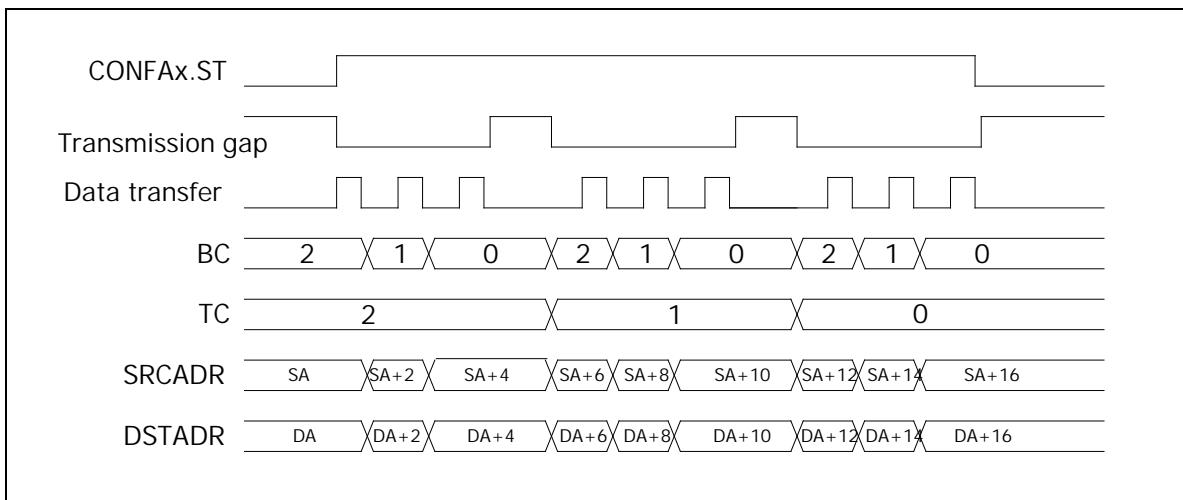
Compare items	Software Block Transfer	Software Burst Transfer	Hardware Block Transfer	Hardware Burst Transfer
Can be interrupted by a higher priority DMA channel or CPU	Yes	No	Yes	No
Start trigger condition	Write DMAC register		Peripheral interrupt flag	
The amount of data that triggers a transfer	$(BC+1) * (TC+1)$		BC+1	$(BC+1) * (TC+1)$
The total amount of data transferred after configuration			$(BC+1) * (TC+1)$	
Main application scenarios	<ul style="list-style-type: none"> Memory to memory Memory to peripherals without DMA request 	<ul style="list-style-type: none"> Memory to peripheral with DMA request Peripheral with DMA request to memory 	<ul style="list-style-type: none"> Memory to Memory Memory to peripherals without DMA request 	

13.4.2 DMA Software Block Transfer Mode

When DMAC_CONFAx.TRI_SEL=0 and DMAC_CONFbx.MODE=0 are configured, DMAC works in software block transfer mode.

When the user code writes 1 to DMAC_CONFAx.ST, DMAC is triggered to start software block transfer mode; DMA transfer stops after completing $(BC+1) * (TC+1)$ data. After each BC+1 data is transferred, DMAC inserts a transfer gap, and the priority arbitrator performs priority arbitration in this gap. If the CPU or a higher priority DMA channel requests the bus in this gap, the bus ownership is transferred to the CPU or a higher priority DMA channel; when the CPU or a higher priority DMA channel releases the bus, the unfinished data transfer will continue to be completed.

The software block transfer diagram is shown below, where SA represents the source address and DA represents the destination address; the data block size is 3 (BC=2), the number of data blocks is 3 (TC=2), the data width is 16bit, and the address is incremented.

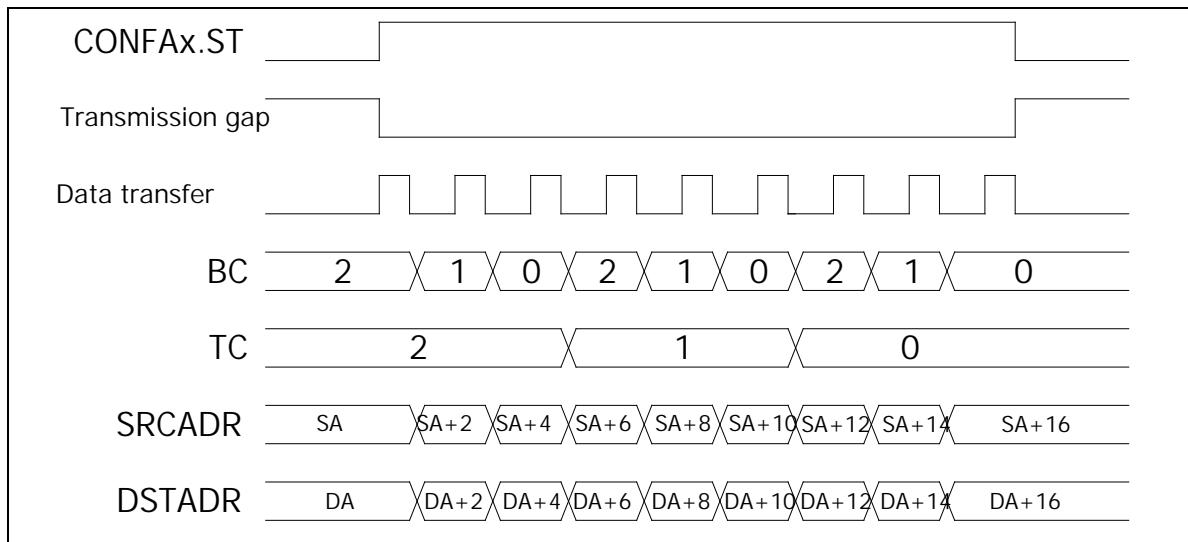


13.4.3 DMA Software Burst Transfer Mode

When DMAC_CONFAx.TRI_SEL=0 and DMAC_CONFbx.MODE=1 are configured, DMAC works in software burst transfer mode.

When the user code writes 1 to DMAC_CONFAx.ST, the DMAC is triggered to start the software Burst transfer mode; the DMA transfer stops after completing $(BC+1) * (TC+1)$ data. There is no transmission gap in this transfer. Before all data transfers are completed, the DMA will occupy the bus all the time. The CPU or higher priority DMA channel can only access the bus after the operation is completed. It is recommended not to transfer too much data each time, otherwise the CPU may not work at all for a period of time.

The software Burst transfer is shown below, where SA represents the source address and DA represents the destination address; the data block size is 3 (BC=2), the number of data blocks is 3 (TC=2), the data width is 16bit, and the address is incremented.



13.4.4 DMA Hardware Block Transfer Mode

When DMAC_CONFAx.TRI_SEL=non-zero and DMAC_CONFBx.MODE=0 are configured, DMAC works in hardware block transfer mode.

Every time the peripheral DMA request signal appears, DMAC is triggered to start a hardware block transfer and transfer (BC+1) data. (TC+1) peripheral DMA request signals are required to complete the transfer of all DMA data. After completing BC+1 data, DMAC inserts a transfer gap, and the priority arbitrator performs priority arbitration in this gap. If the CPU or a higher priority DMA channel is also requesting the bus in this gap, the bus ownership is transferred to the CPU or a higher priority DMA channel; when the CPU or a higher priority DMA channel releases the bus, the unfinished data transfer will continue to be completed.

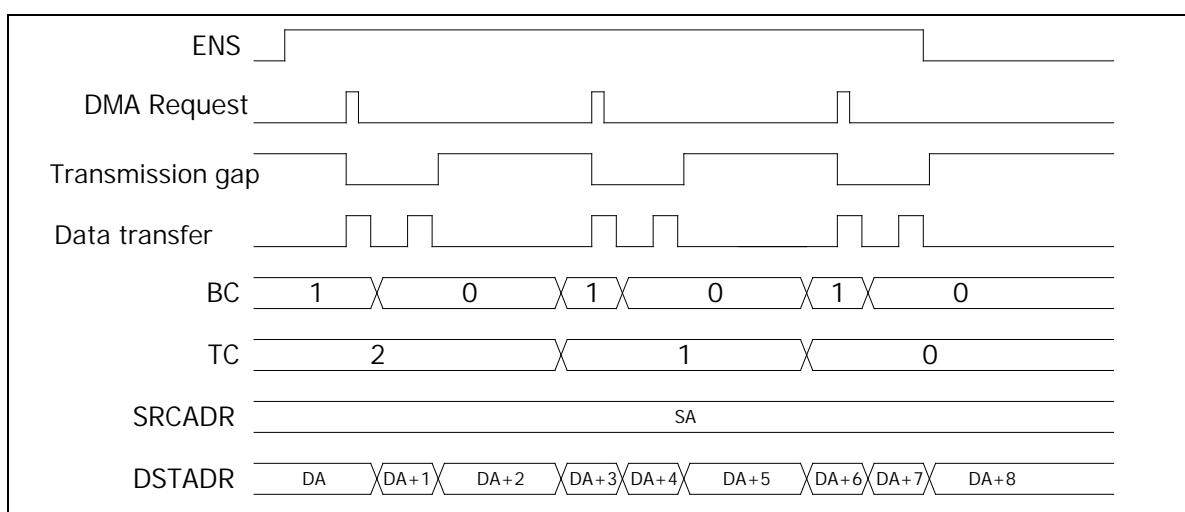
Note:

- In this mode, BC is generally set to 0 to obtain a data from the peripheral or provide a data to the peripheral.

The hardware request signals supported by DMA are as follows:

- LPUARTx / UARTx receive buffer is not empty, transmit buffer is empty
- SPIx receive buffer is not empty, transmit buffer is empty
- I2Sx receive buffer is not empty, transmit buffer is empty
- TIMx capture is complete, compare match is complete
- ADC queue conversion is complete
- ADC sequential conversion is complete
- LCD timer interrupt
- DAC conversion is complete

The hardware block transmission diagram is as follows, where SA represents the source address and DA represents the target address; the data block size is 2 (BC=1), the number of data blocks is 3 (TC=2), the data width is 8bit, the target address is self-incrementing, the source address is fixed, and the transmission is automatically disabled after completion.



13.4.5 DMA Hardware Burst Transfer Mode

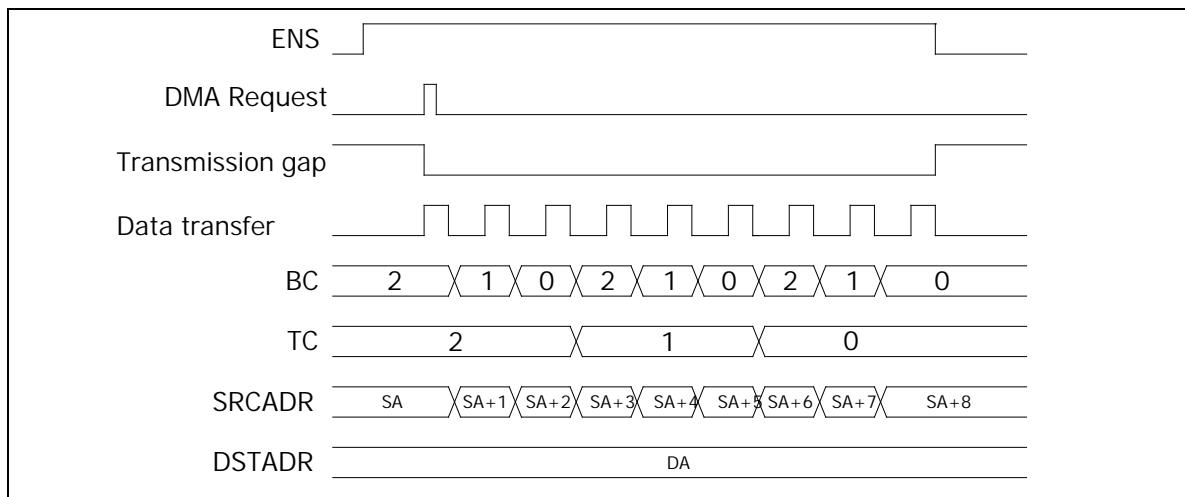
When DMAC_CONFAx.TRI_SEL=non-zero and DMAC_CONFBx.MODE=1 are configured, DMAC works in hardware Burst transfer mode.

When the peripheral DMA request signal appears, DMAC is triggered to start the hardware Burst transfer mode, and stops after the transfer of $(BC+1) * (TC+1)$ data. There is no transmission gap in this transmission. Before all data transfers are completed, DMA will occupy the bus all the time. The CPU or higher priority DMA channel can only access the bus after the operation is completed. It is recommended not to transfer too much data each time, otherwise it may cause the CPU to be completely unable to work for a period of time.

The hardware request signals supported by DMAC are as follows:

- LPUARTx / UARTx receive buffer is not empty, transmit buffer is empty
- SPIx receive buffer is not empty, transmit buffer is empty
- I2Sx receive buffer is not empty, transmit buffer is empty
- TIMx capture is complete, compare match is complete
- ADC queue conversion is complete
- ADC sequential conversion is complete
- LCD timer interrupt
- DAC conversion is complete

The hardware Burst transmission is shown below, where SA represents the source address and DA represents the target address; the data block size is 3 (BC=2), the number of data blocks is 3 (TC=2), the data width is 8bit, the source address is self-incrementing, the target address is fixed, and the transmission is automatically disabled after completion.



13.4.6 DMA transfer pause

When DMAC is working in Block transfer mode, if DMAC_CONF.HALT is configured as a non-zero value, all channels that are transmitting will enter the pause state during the transfer interval and no longer transmit data. After DMAC_CONF.HALT is configured as 0, DMAC needs to receive the next trigger signal to continue the unfinished transfer. When DMAC_CONFAx.PAS is configured as 1, the corresponding channel that is transmitting will enter the pause state during the transfer interval and no longer transmit data. After DMAC_CONFAx.PAS is configured as 0, DMAC needs to receive the next trigger signal to continue the unfinished transfer.

When DMAC is working in Burst transfer mode and no data is being transmitted, if DMAC_CONF.HALT is configured as a non-zero value, the DMA channel will enter the pause state and no longer transmit data. After DMAC_CONF.HALT is configured as 0, DMAC will start a new complete transfer when it receives the next trigger signal. When DMAC is working in Burst transfer mode and no data is being transferred, if DMAC_CONFAx.PAS is configured to 1, the DMA channel enters the pause state and no data is transferred. After DMAC_CONFAx.PAS is configured to 0, DMAC will start a new complete transfer only when it receives the next trigger signal.

When DMAC is working in Burst transfer mode and data is being transferred, writing a non-zero value to DMAC_CONF.HALT or writing 1 to DMAC_CONFAx.PAS will not pause the ongoing transfer; the DMA channel will not enter the pause state after the current transfer is completed.

13.4.7 Other DMA configurations

13.4.7.1 Data Width

By configuring DMAC_CONFBx.WIDTH, the width of each data to be transmitted can be configured to 8bit, 16bit, or 32bit. Note that the bit width should be exactly the same as the bit width of the source address and the destination address of the DMA.

13.4.7.2 Data Block Size

By configuring DMAC_CONFAx.BC, the number of data to be transmitted in each data block can be configured to be 1 to 16. Note that the value of BC should match the number of consecutive operations provided by the source address and the destination address.

If the source address or the destination address can only provide or receive one data at a time, BC can only be configured to 0. For example, UART/LPUART/SPI can only provide one data to DMA at a time, and UART/LPUART/SPI/DAC can only receive one data from DMA at a time, so BC can only be configured to 0.

If both the source address and the destination address can provide or receive multiple data, BC can be configured to a larger value to speed up the transmission. For example, when transferring from FLASH to RAM or from FLASH to CRC, the source address and the destination address can be operated continuously, so BC can be configured to 15 to achieve fast transfer.

13.4.7.3 Number of data blocks

Configure DMAC_CONFAx.TC to configure the number of data blocks transferred each time DMA is started to be 1~65536.

13.4.7.4 Channel Priority

When DMAC_CONF.PRIO=0, DMAC works in fixed priority mode. At this time, CH0 has a higher priority than CH1. Only when CH0 is not performing data transmission, CH1 can perform data transmission.

When DMAC_CONF.PRIO=1, DMAC works in cyclic priority mode. At this time, CH0 and CH1 have higher priority in turn, and the two channels will take turns to occupy the bus for data transmission.

13.4.7.5 Automatic Reloading

If the automatic reloading function is enabled, the data block size, number of data blocks, source address, and destination address are automatically loaded with the last configured values after the transmission is completed, and there is no need to configure each time.

13.4.8 DMA Interrupt

DMA supports two types of interrupts: transmission error and transmission completion. When an interrupt is triggered, the condition that triggered the interrupt can be determined by querying the status register DMAC_CONFBx.STAT, and the built-in interrupt status flag can be cleared by clearing the register DMAC_CONFBx.STAT.

The 5 conditions that trigger DMA interrupts are as follows:

- The transfer address exceeds the addressing range
- The peripheral requests to stop DMA
- An error occurs when accessing the transfer source address
- An error occurs when accessing the transfer destination address
- The transfer is completed

13.5 Registers

DMAC base address: 0x4002 1000

Register	Offset Address	Control Channel	Description
DMAC_CONF	0x00	ALL	Common channel configuration register
DMAC_CONFA0	0x10	CH0	Channel 0 Configuration A Register
DMAC_CONFB0	0x14	CH0	Channel 0 Configuration B Register
DMAC_SRCADR0	0x18	CH0	Channel 0 Transmission Source Address Register
DMAC_DSTADR0	0x1C	CH0	Channel 0 Transmission Destination Address Register
DMAC_CONFA1	0x20	CH1	Channel 1 Configuration A Register
DMAC_CONFB1	0x24	CH1	Channel 1 Configuration B Register
DMAC_SRCADR1	0x28	CH1	Channel 1 Transmission Destination Address Register
DMAC_DSTADR1	0x2C	CH1	Channel 1 Transmission Destination Address Register

13.5.1 DMAC_CONF

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN	Res		PRI0		HALT										Reserved
RW		R	RW		RW										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bit	Mark	Function description
31	EN	DMA controller enable 1: Enable DMA controller 0: Disable DMA controller <i>Note: When the DMA controller is disabled, the ongoing transfer will be forced to stop, causing unpredictable results.</i>
30:29	Reserved	
28	PRI0	DMA channel priority configuration 1: Cyclic priority mode, CH0 and CH1 take turns to obtain bus access rights 0: Fixed priority mode, CH0 priority is higher than CH1 priority
27:24	HALT	DMA all channel transfer pause control 0000: Resume all channel data transfer xxxx: Pause all channel data transfer
23:0	Reserved	

13.5.2 DMAC_CONFA0, DMAC_CONFA1

Offset address: 0x10 (DMAC_CONFA0)

0x20 (DMAC_CONFA1)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENS	PAS	ST				TRI_SEL				Reserved		BC			
RW	RW	RW				RW						RW			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TC[15:0]															
RW															

Bit	Mark	Function description
31	ENS	DMA channel enable control 1: Enable the current DMA channel; if CONFBx.MSK is 0, this bit is automatically cleared when the transfer is completed 0: Disable the current DMA channel <i>Note: When the DMA channel is disabled, the ongoing transfer will be forced to stop, causing unpredictable results.</i>
30	PAS	DMA channel data transfer pause control 1: Pause DMA channel data transfer 0: Resume DMA channel data transfer
29	ST	DMA channel software trigger control 1: Trigger the DMA channel to start software Block / Burst transfer, and this bit is automatically cleared when the transfer is completed 0: Stop DMA channel software Block / Burst transfer
28:22	TRI_SEL	DMA channel trigger source configuration 0x00: Software trigger 0x40: SPI0 receive buffer is not empty 0x41: SPI0 transmit buffer is empty 0x42: SPI1 receive buffer is not empty 0x43: SPI1 transmit buffer is empty 0x44: ADC queue conversion is completed 0x45: ADC sequential conversion is completed 0x46: LCD timer interrupt 0x47: Reserved 0x48: UART0 receive buffer is not empty 0x49: UART0 transmit buffer is empty 0x4A: UART1 receive buffer is not empty 0x4B: UART1 transmit buffer is empty 0x4C: LPUART0 receive buffer is not empty 0x4D: LPUART0 transmit buffer is empty 0x4E: LPUART1 receive buffer is not empty 0x4F: LPUART1 transmit buffer is empty

		0x50: DAC0 conversion completed 0x51: DAC1 conversion completed 0x52: TIM0 channel A, capture event or comparison event occurred 0x53: TIM0 channel B, capture event or comparison event occurred 0x54: TIM1 channel A, capture event or comparison event occurred 0x55: TIM1 channel B, capture event or comparison event occurred 0x56: TIM2 channel A, capture event or comparison event occurred 0x57: TIM2 channel B, capture event or comparison event occurred 0x58: TIM3 channel A, capture event or comparison event occurred 0x59: TIM3 channel B, capture event or comparison event occurred 0x5A: TIM4 channel A, capture event or comparison event occurred 0x5B: TIM4 channel B, capture event or comparison event occurred 0x5C: TIM5 channel A, capture event or comparison event occurred 0x5D: TIM5 Channel B, capture event or compare event occurs 0x5E: TIM6 Channel A, capture event or compare event occurs 0x5F: TIM6 Channel B, capture event or compare event occurs 0x60: UART2 receive buffer is not empty 0x61: UART2 transmit buffer is empty 0x62: UART3 receive buffer is not empty 0x63: UART3 transmit buffer is empty 0x64: I2S0 left channel transmit buffer is empty or receive buffer is not empty 0x65: I2S0 right channel transmit buffer is empty or receive buffer is not empty 0x66: I2S1 left channel transmit buffer is empty or receive buffer is not empty 0x67: I2S1 right channel transmit buffer is empty or receive buffer is not empty
21:20	Reserved	Reserved
19:16	BC	Configure the size of the data block to be transmitted by the DMA channel to BC + 1 <i>Note: When the source address or the target address can only provide or receive one data at a time, the BC value can only be set to 0</i>
15:0	TC	Configure the number of data blocks to be transmitted by the DMA channel to TC + 1 During the transmission process, each time a data block is transmitted, the register value is decremented <i>Note: The total amount of data transmitted by DMA is (TC + 1) * (BC + 1)</i>

13.5.3 DMAC_CONF0, DMAC_CONF1

Offset address: 0x14 (DMAC_CONF0)

0x24 (DMAC_CONF1)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	MODE		WIDTH		FS	FD	RC	RS	RD	ERR IE	FIS_I E	STAT			
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															MSK
															RW

Bit	Mark	Function description
31:30	Reserved	
29:28	MODE	DMA channel transfer mode configuration 00: Block transfer 01: Burst transfer 10: Reserved 11: Reserved
27:26	WIDTH	DMA channel transfer data width configuration 00: 8bit 01: 16bit 10: 32bit 11: Reserved
25	FS	DMA channel source address auto-increment enable 1: The source address remains unchanged during the transfer process 0: The source address is auto-incremented after each data transfer; the data width is 8bit / 16bit / 32bit, and the corresponding auto-increment is 1 / 2 / 4 respectively
24	FD	DMA channel target address auto-increment enable 1: The target address remains unchanged during the transfer process 0: The target address is auto-incremented after each data transfer; the data width is 8bit / 16bit / 32bit, and the corresponding auto-increment is 1 / 2 / 4 respectively
23	RC	DMA channel BC and TC reload enable 1: Enable BC/TC reload, the BC/TC value will automatically return to the initial value written after the transfer is completed 0: Disable BC/TC reload, the BC/TC value will be 0 after the transfer is completed
22	RS	DMA channel source address reload enable 1: Enable source address reload, the source address value will be the initial value written after the transfer is completed 0: Disable source address reload, the source address value will be uncertain after the transfer is completed
21	RD	DMA channel target address reload enable 1: Enable target address reload, the target address value will automatically return to the initial value written after the transfer is completed 0: Disable target address reload, the target address value will be uncertain after the transfer is completed

20	ERR_IE	DMA channel transmission error interrupt enable 1: Generate an interrupt when a transmission error occurs 0: Disable an interrupt when a transmission error occurs <i>Note: In the interrupt service program, you need to write 0x00 to STAT to clear the interrupt status</i>
19	FIS_IE	DMA channel transmission completion interrupt enable 1: Generate an interrupt when the transmission is completed 0: Disable an interrupt when the transmission is completed <i>Note: In the interrupt service program, you need to write 0x00 to STAT to clear the interrupt status</i>
18:16	STAT	DMA channel current transmission status 000: Initial value 001: Transmission error, the transmission address exceeds the addressing range 010: Transmission error, the peripheral requests to stop DMA 011: Transmission error, error in accessing the transmission source address 100: Transmission error, error in accessing the transmission destination address 101: Transmission completed 110: Reserved 111: Transmission paused
15:1	Reserved	
0	MSK	DMA channel transmission is automatically disabled when the transmission is completed 1: When the DMA channel transmission is completed, CONFAX.ENS remains unchanged 0: When the DMA channel transmission is completed, CONFAX.ENS is automatically cleared

13.5.4 DMAC_SRCADR0, DMAC_SRCADR1

Offset address: 0x18 (DMAC_SRCADR0)

0x28 (DMAC_SRCADR1)

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:0	SRCADR	DMA channel source address configuration You can configure its automatic reload through CONFBx.RS <i>Note: The address alignment should match the data width, otherwise it will cause address access errors</i>

13.5.5 DMAC_DSTADR0, DMAC_DSTADR1

Offset address: 0x1C (DMAC_DSTADR0)

0x2C (DMAC_DSTADR1)

Reset value: 0x0000 0000

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

Bit	Mark	Function description
31:0	DSTADR	DMA channel target address configuration You can configure its automatic reload through CONFBx.RD <i>Note: The address alignment should match the data width, otherwise it will cause address access errors</i>

14 General timers (TIM0/1/2/3)

14.1 General Timers Introduction

TIM0/1/2 is a timer composed of 1 counting unit and 2 comparison units. Each timer supports 2 independent PWM outputs or 1 pair of complementary PWM outputs and 1 independent PWM output. Supports 2 capture inputs. TIM0/1/2 can form 3 pairs of complementary PWM outputs.

TIM3 is a timer composed of 1 counting unit and 6 comparison units, supporting 6 independent PWM outputs or 3 pairs of complementary PWM outputs. Supports 6 capture inputs.

Using timer pre-scaling, system pre-scaling and system clock selection, the pulse width and waveform period can be flexibly adjusted; the pulse width can be conveniently measured.

14.1.1 Basic Characteristics (TIM0/1/2)

- 2 independent PWM outputs CHA, CHB
- 1 complementary PWM output (CHA, CHB) + 1 independent PWM output (gate)
- 1 complementary PWM output (CHA, CHB) + 1 capture function (gate)
- Up to 2 capture inputs
- Pulse width measurement
- Dead zone control
- Brake control
- Edge-aligned, symmetrical center-aligned and asymmetrical center-aligned PWM outputs
- Quadrature encoding counting function
- Single pulse mode
- External counting function
- DMA trigger
- When the system clock selects PLL, the timer clock can be configured as 2 times the system clock

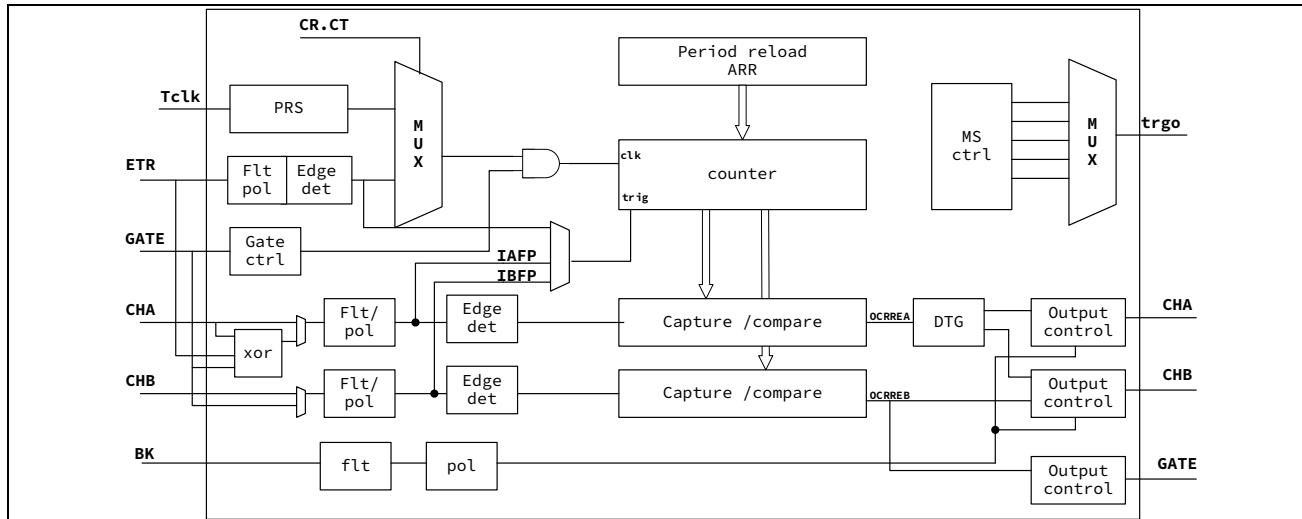


Figure 14-1 TIM0/1/2 block diagram

14.1.2 Basic characteristics (TIM3)

- 6 independent PWM outputs CH0A, CH0B, CH1A, CH1B, CH2A, CH2B
- 3 complementary PWM outputs (CHxA, CHxB) + 1 independent PWM output (gate)
- 3 complementary PWM outputs (CHxA, CHxB) + 1 capture function (gate)
- Up to 6 capture inputs
- Pulse width measurement
- Dead zone control
- Brake control
- Edge-aligned, symmetrical center-aligned and asymmetrical center-aligned PWM outputs
- Quadrature encoding counting function
- Single pulse mode
- External counting function
- DMA trigger
- When the system clock selects PLL, the timer clock can be configured as 2 times the system clock

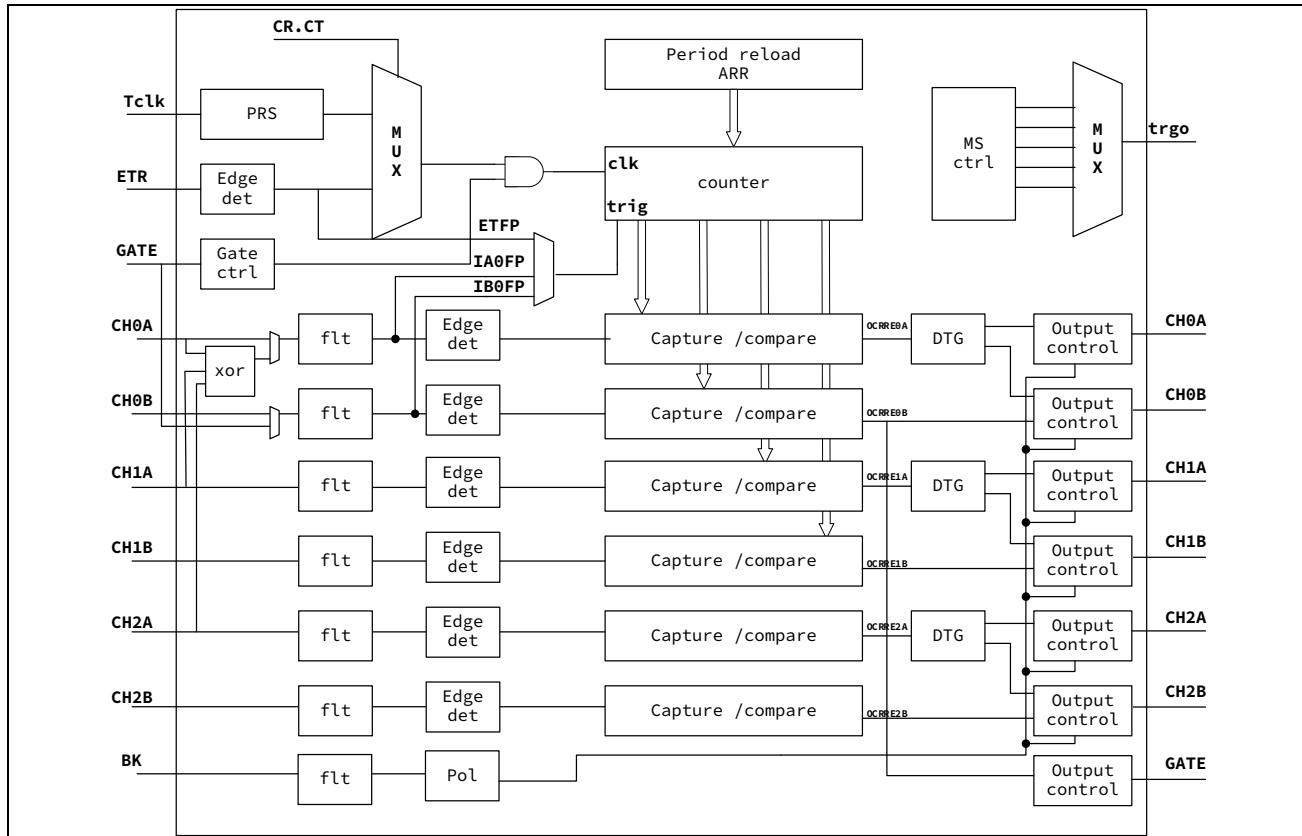


Figure 14-2 TIM3 Block Diagram

14.2 Timer Function Description

14.2.1 Timer Clock

The timer uses PCLK as the timer clock. The timer clock is indicated by TCLK below.

14.2.2 Timing Counter

The main module of the programmable general-purpose timer is a 16-bit counter and its associated auto-reload register. The counter can count up, count down, or count up and down alternately. The clock of the counter can be divided by the prescaler.

The counter, auto-reload register, and prescaler register can be read and written by software. Read and write operations can be performed even when the counter is running.

14.2.3 Timer Prescaler

When using TCLK as the Timer clock, you can use prescaler. The prescaler settings are as follows:

PRS	000	001	010	011	100	101	110	111
Division ratio	1	2	4	8	16	32	64	256

The prescaler does not have a preload buffer, so when the prescaler is changed, the prescaler will take effect immediately.

14.2.4 Mode 0 Counting Timer Function

In this mode, the counter counts up. The counter supports two counting modes, reload mode and free counting mode. You can choose to count with external clock ETR or system clock. Gate can control the counting mask. An interrupt is generated when the count reaches the maximum value. Flip the output ports CHA and CHB. In this mode, CHA and CHB are inverted.

The counting range of the reload mode is from ARR to 0xFFFF overflow, and then counts from ARR again. The counting cycle is 0Xffff-ARR+1; the free counting mode counts from the set count value to 0xFFFFFFFF and then overflows. After the overflow, the count value restarts from 0x0.

The counter can be used to control whether the counter counts through the external gate function. The control relationship is as follows:

MOCR.GATE	MOCR.GATEP	Port GATE Input	MOCR.CTEN	Counter
x	x	x	0	Do not count
0	x	x	1	Count
1	0	High level	1	Count
1	0	Low level	1	Do not count
1	1	High level	1	Do not count
1	1	Low level	1	Count

14.2.4.1 Functional Block Diagram

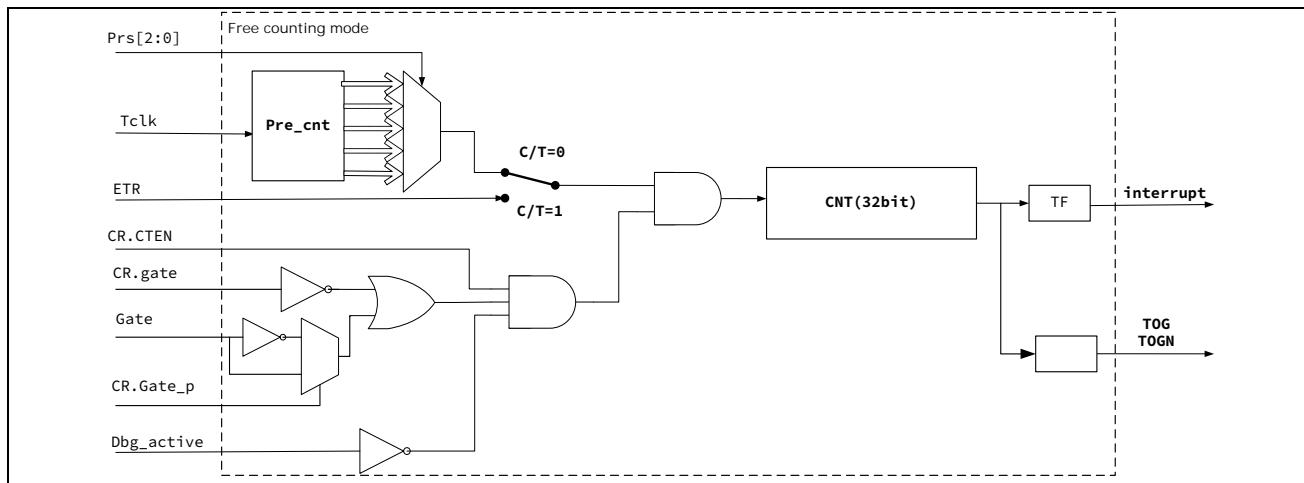


Figure 14-3 Free counting block diagram

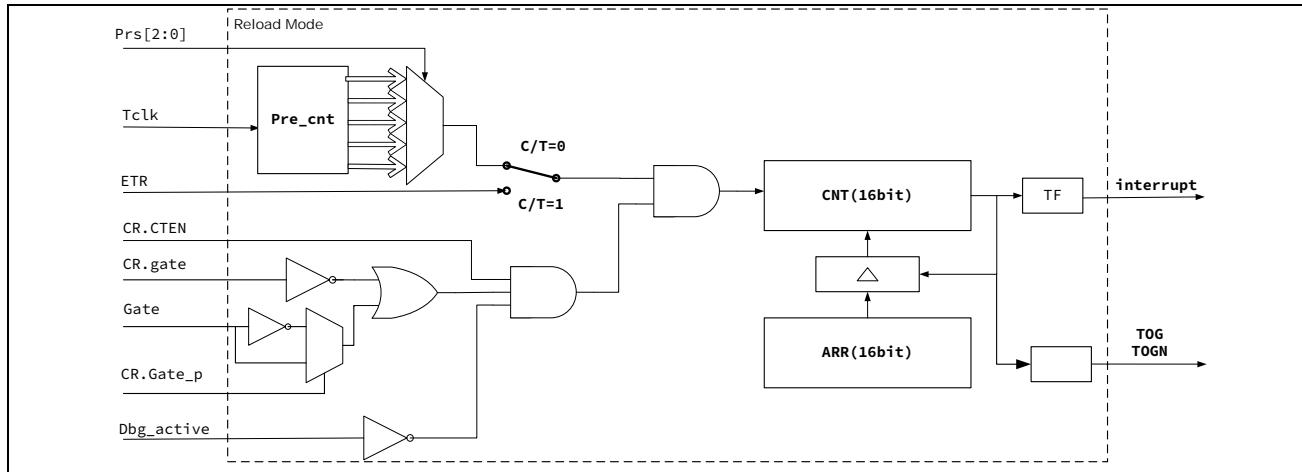


Figure 14-4 Heavy load counting waveform

14.2.4.2 Counting Waveform

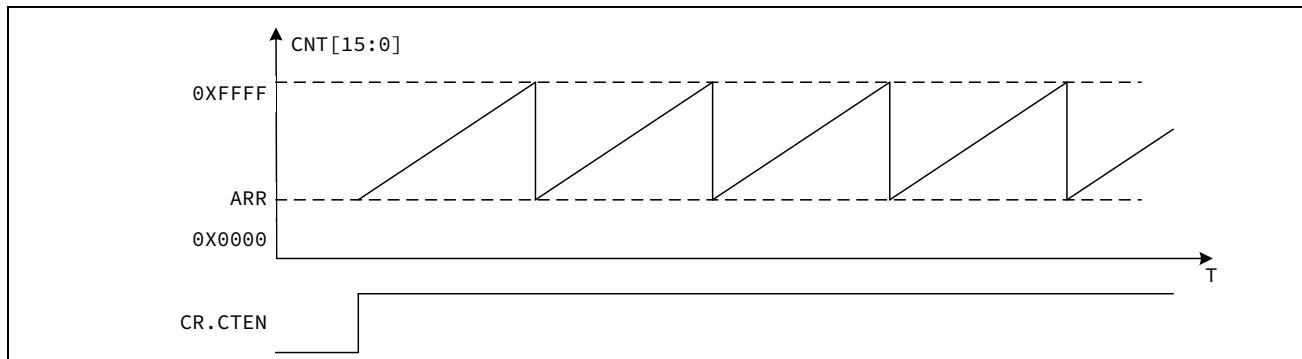


Figure 14-5 16-bit reload count waveform

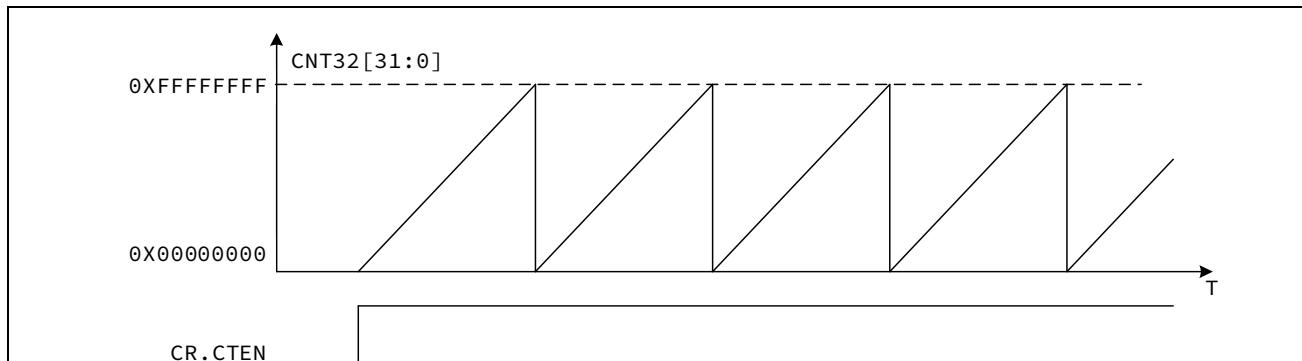


Figure 14-6 32-bit free counting waveform

14.2.4.3 Counting function

The counting function is used to measure the number of times an event occurs. In the counting function, the counter accumulates once on each falling edge of the corresponding input clock. The input signal is sampled by the internal tclk, so the external input clock frequency cannot exceed the system tclk clock. When the count reaches the maximum value, it will overflow and generate an interrupt. The interrupt flag needs to be cleared by software.

14.2.4.4 Timing function

The timing function is used to generate interval timing. In the timing function, the timer has a pre-division frequency. The timer accumulates once for each pre-division clock. When the count reaches the maximum value, it overflows and generates an interrupt. The interrupt flag needs to be cleared by software.

14.2.4.5 Timing diagram

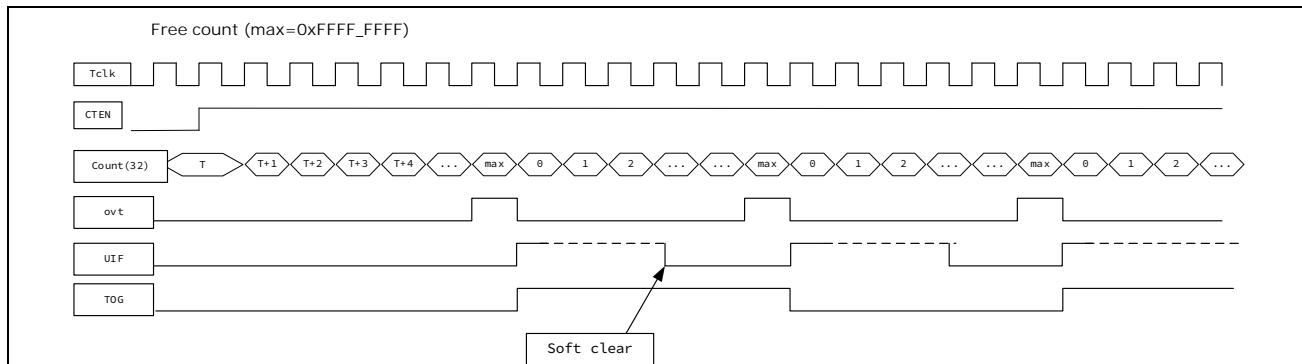


Figure 14-7 Free counting timing diagram

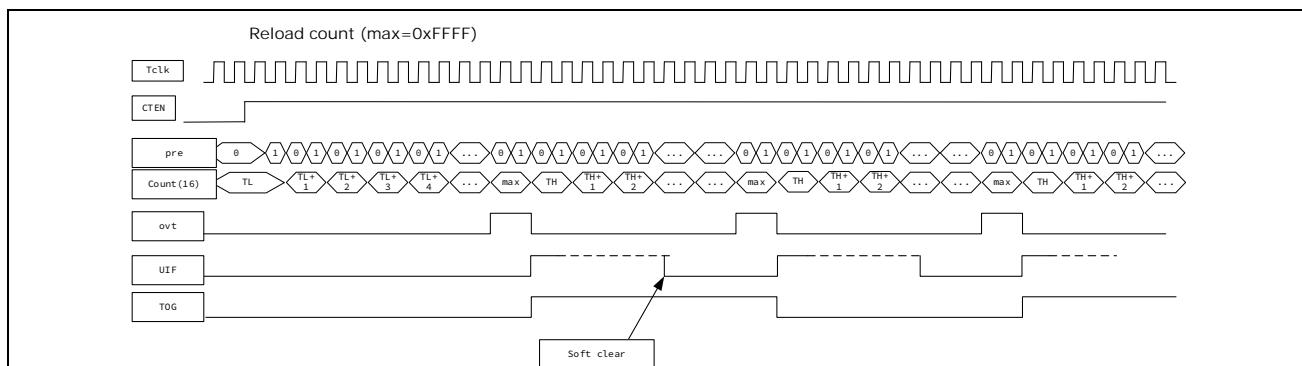


Figure 14-8 Reload count timing diagram (prescaler is set to 2)

14.2.4.6 Buzzer Function

The function of driving the buzzer can be realized through the flip output function of the timer. To use the toggle output, the DTR.MOE control bit needs to be enabled. Setting CR.TOG_EN to 0 can simultaneously set the port CHA and CHB outputs to 0. When the counting clock is 4M, the timer reload mode configuration of the buzzer output with different frequencies is as follows:

Buzzer frequency	Counter period	Counter count value	Counter reload value	CNTL initial value	CNTH reload value
1000Hz	0.5ms	2000	63536	0XF830	0XF830
2000Hz	0.25ms	1000	64536	0xFC18	0xFC18
4000 Hz	0.125ms	500	65036	0xFE0C	0xFE0C

14.2.4.7 Setting Example

Reload timer setting

1. Set timer mode MOCR.MODE=0
2. Set load value ARR
3. Set counter initial value CNT
4. Clear interrupt flag
5. Enable interrupt MOCR.UIE
6. Enable reload mode MOCR.MD
7. Enable timer MOCR.CTEN

Gated external clock free counting setting

1. Set timer mode MOCR.MODE=0
2. Set counter initial value CNT
3. Enable gate function MOCR.GATE
4. Select gate valid level MOCR.GATEP
5. Clear interrupt flag
6. Enable interrupt MOCR.UIE
7. Enable external clock mode MOCR.CT
8. Enable timer MOCR.CTEN

BUZZER output control

1. Set the appropriate ARR value according to the output frequency
2. Set the timer to reload mode, refer to the reload timer setting
3. Enable output enable DTR.MOE
4. Start another timer control MOCR.TOGEN to achieve frequency interval output.

14.2.5 Mode 1 Pulse Width Measurement PWC

In this mode, the high level, low level or cycle width of the input pulse can be automatically measured.

The first valid edge counter is initialized to 0x0001, the second valid edge will stop counting, and the current count value will be stored in CMAR, and a capture interrupt CAF will be generated. If the counter overflows, an overflow flag will be generated. Setting the overflow interrupt enable will generate an overflow interrupt.

M1CR.edg1st	0	0	1	1
M1CR.edg2nd	0	1	0	1
Pulse width measurement	Upward edge~upward edge cycle width	Upward edge~downward edge high level width	Downward edge~upward edge low level width	Downward edge~downward edge cycle width

When measuring cycles, one cycle is measured at intervals.

14.2.5.1 PWC Functional Block Diagram

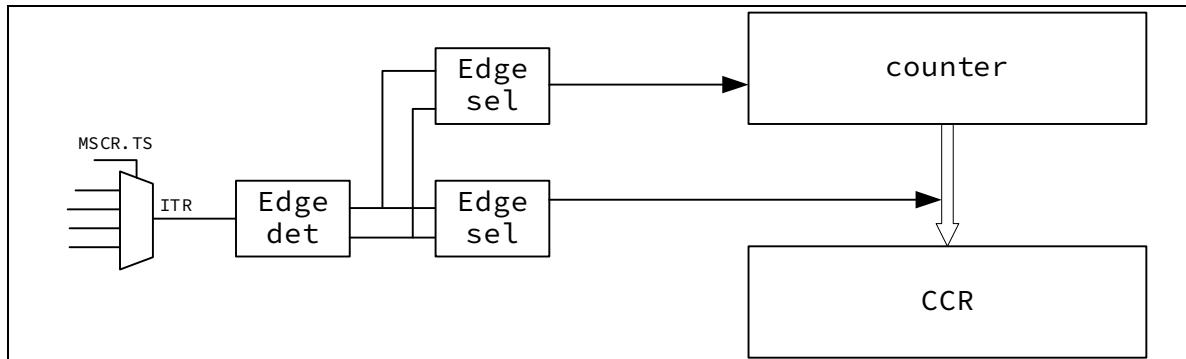


图 14-9 PWC 测量框图

MSCR.TS	触发选择 000: 端口ETR的滤波相位选择后的信号ETFP ; 001: 内部互联信号 ITR0 010: 内部互联信号 ITR1; 011: 内部互联信号 ITR2; 100: 内部互联信号 ITR3; 101: 无效 110: 端口CH0A的滤波后的信号IAFP (极性选择在脉宽测量模式下无效) 111: 端口CH0B的滤波后的信号IBFP (极性选择在脉宽测量模式下无效)
---------	---

14.2.5.2 PWC waveform measurement timing diagram

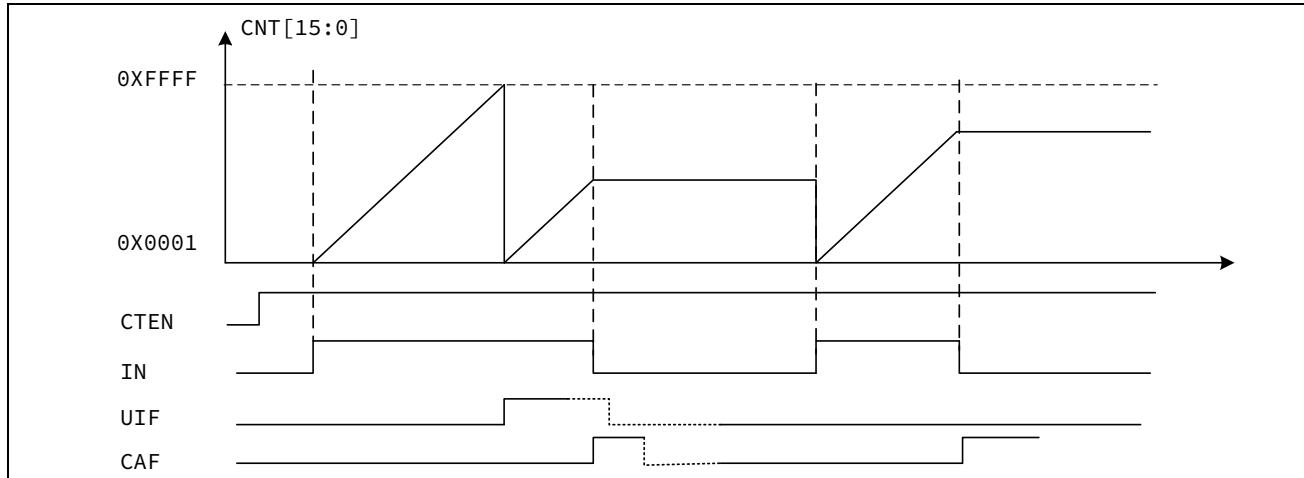


图 14-10 高电平脉冲宽度测量

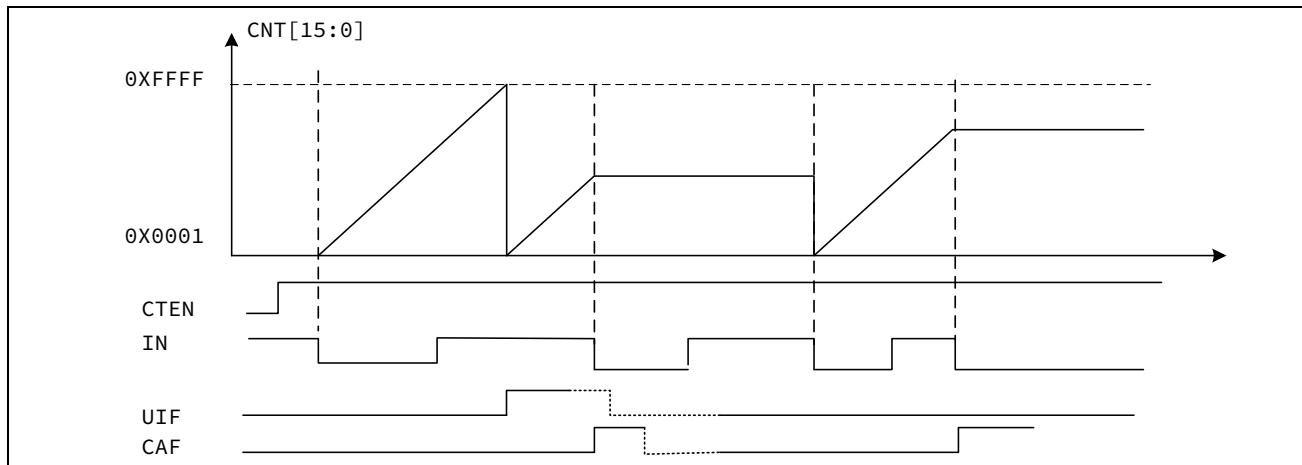


图 14-11 下降沿到下降沿周期测量

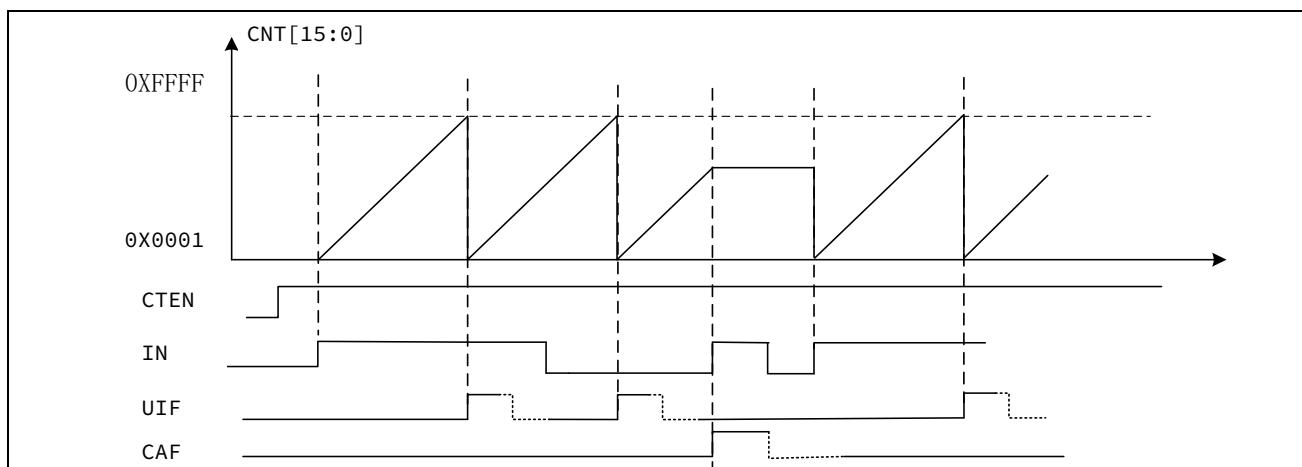


图 14-12 上升沿到上升沿周期测量

通过寄存 MSCR.TS 选择测量信号源。

000 ETFP: ETR 外部输入滤波后的相位选择信号, 可选择外部滤波与输入反向

001 ITR0: Timer 内部互联信号 0, 其他 timer 的 TRGO 输出

010 ITR1: Timer 内部互联信号 1, 其他 timer 的 TRGO 输出

011 ITR2: Timer 内部互联信号 2, 其他 timer 的 TRGO 输出

100 ITR3: Timer 内部互联信号 3, 其他 timer 的 TRGO 输出

101 IA0ED: 无效

110 IAfp: CH0A 外部输入滤波后的相位选择信号, 可选择外部滤波与输入反向

111 IBfp: CH0B 外部输入滤波后的相位选择信号, 可选择外部滤波与输入反向

	ITR0	ITR1	ITR2	ITR3
Timer0	-	TIM1_TRGO	TIM2_TRGO	TIM3_TRGO
Timer1	TIM0_TRGO	-	TIM2_TRGO	TIM3_TRGO
Timer2	TIM0_TRGO	TIM1_TRGO	-	TIM3_TRGO
Timer3	TIM0_TRGO	TIM1_TRGO	TIM2_TRGO	-

注：关于 TRGO 输出参考寄存器描述。

14.2.5.3 PWC one-shot mode

设置 M1CR.ONESHOT=1 可以设置 PWC 单次测量，测量完成后 CTEN 将被清除。

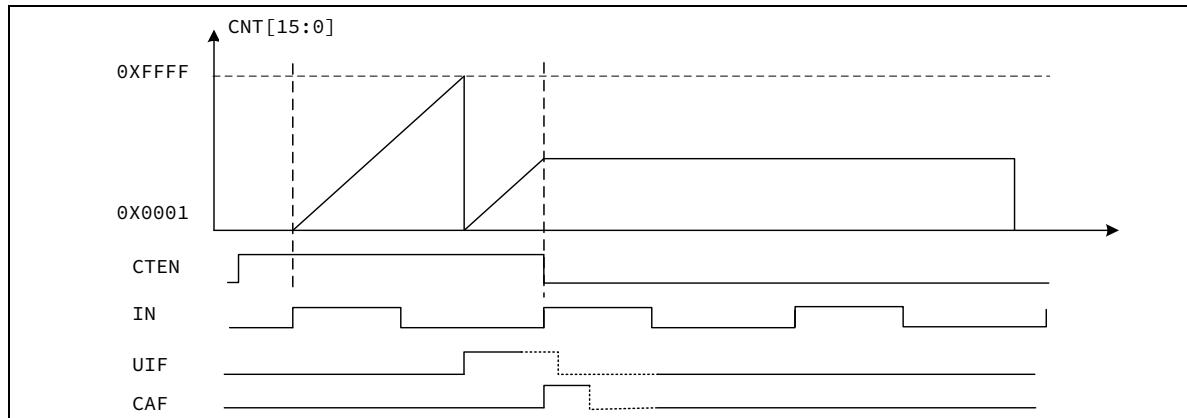


图 14-13 上升沿到上升沿周期测量单次模式

14.2.5.4 Setting Example

脉冲低电平测量设置

1. 设置为脉冲测量模式 M1CR.MODE=1
2. 设置 MSCR.TS 选择测量的信号
3. 设置 M1CR.edg2dn=0,M1CR.edg1st=1 选择测量低电平
4. 清除中断标志
5. 使能溢出中断 M1CR.UIE
6. 使能测量结束中断 CR0.CIEA
7. 使能定时器 M1CR.CTEN
8. 中断服务程序中读取 CCR0A 及溢出次数并清除中断标志
9. 等待下次测量

脉冲高电平单次测量设置

1. 设置为脉冲测量模式 M1CR.MODE=1
2. 设置 MSCR.TS 选择测量的信号
3. 设置脉冲单次测量模式 M1CR.ONESHOT=1
4. 设置 M1CR.edg2dn=1, M1CR.edg1st=0 选择测量低电平
5. 清除中断标志
6. 使能溢出中断 M1CR.UIE

7. 使能测量结束中断 CR0.CIEA
8. 使能定时器 M1CR.CTEN
9. 中断服务程序中读取 CCR0A 及溢出次数并清除中断标志
10. 测量结束

14.2.6 Mode 2/3 Compare Capture Mode

14.2.6.1 Counters

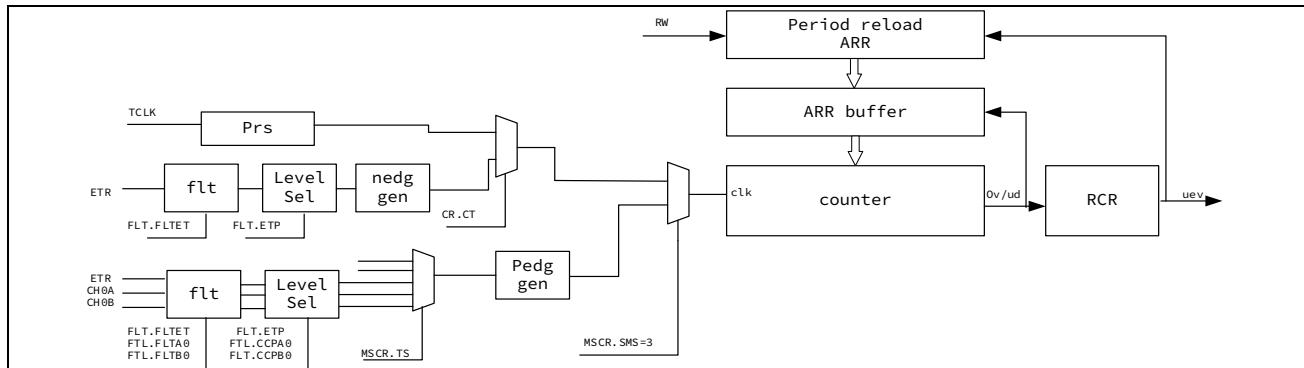


图 14-14 计数器框图

计数器主要部分是一个 16 位计数器与相关的自动装载寄存器。这个计数器可以向上计数（模式 2），向下计数（模式 2）或向上向下双向计数（模式 3）。计数器的时钟可以由预分频器 PRS 分频得到，也可以选择 ETR 输入外部时钟或者通过 MSCR.TS 选择的外部输入信号和内部互联信号。

- 计数器基本单元包括：
- 计数器寄存器 CNT
- 预分频寄存器 CR.PRS
- 自动装载寄存器 ARR
- 重复次数寄存器 RCR
- 时钟选择控制寄存器 FLT,CR0,MSCR,CR

自动装载寄存器具有缓存功能，计数器产生事件更新后重载值从缓存寄存器更新到计数器。当计数器停止状态或者缓存功能关闭状态，自动装载寄存器立刻更新到缓存寄存器。当定时器处于运行状态并行缓存功能有效时，写入到自动装载寄存器的值不会立刻更新到缓存寄存器，当事件更新后才有自动装载寄存器更新到缓存寄存器。

时钟选择及门控功能，触发功能，复位功能参考模式 2/3 从模式章节。

14.2.6.2 Counter Waveform

模式 2 为锯齿波计数波形，通过设置 CR.DIR 可以更改计数方向。

设置 CR.DIR 为 0 时，计数器为递增计数模式，这种模式下，计数器从 0 计数到自动重载值 (TIMx_ARR)，然后重新从 0 开始计数并生成计数器上溢事件。如果使用重复计数器，则当递增计数的重复次数达到重复计数器寄存器中编程的次数加一次 (TIMx_RCR+1) 后，将生成更新事件 (UEV)。否则，将在每次计数器上溢时产生更新事件。

将 TIMx_CR 寄存器的 UG 位置 1（通过软件或使用从模式控制器）时，也将产生更新事件。

发生更新事件时，将更新所有寄存器且将更新标志 (TIMx_IFR 寄存器中的 UIF 位) 置 1 (取决于 URS 位)：

- 自动重载缓存值将以 ARR 寄存器值进行更新
- 比较缓存值将以比较寄存器 CCRxy 进行更新

以下图示显示 ARR=0X2C 时不同计数方向的计数器波形

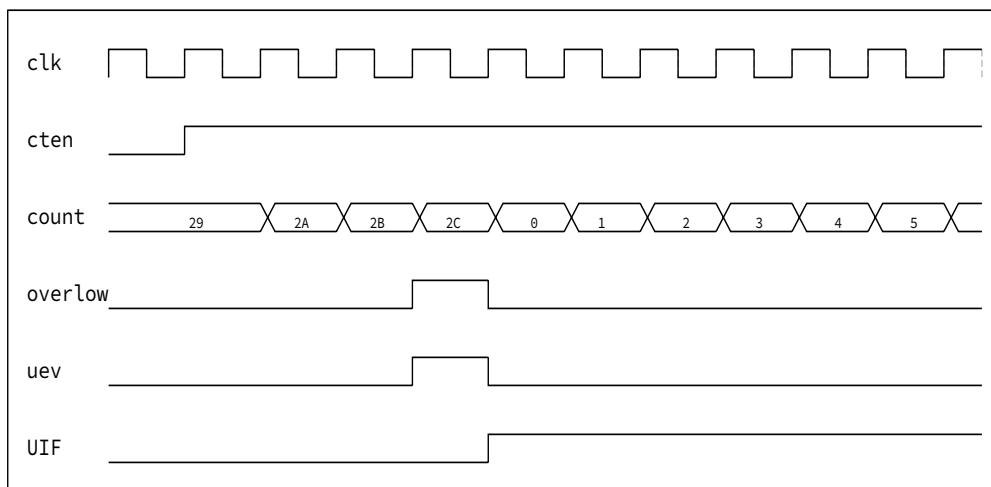


图 14-15 无预分频的向上计数

计数溢出周期时钟个数为 ARR+1，

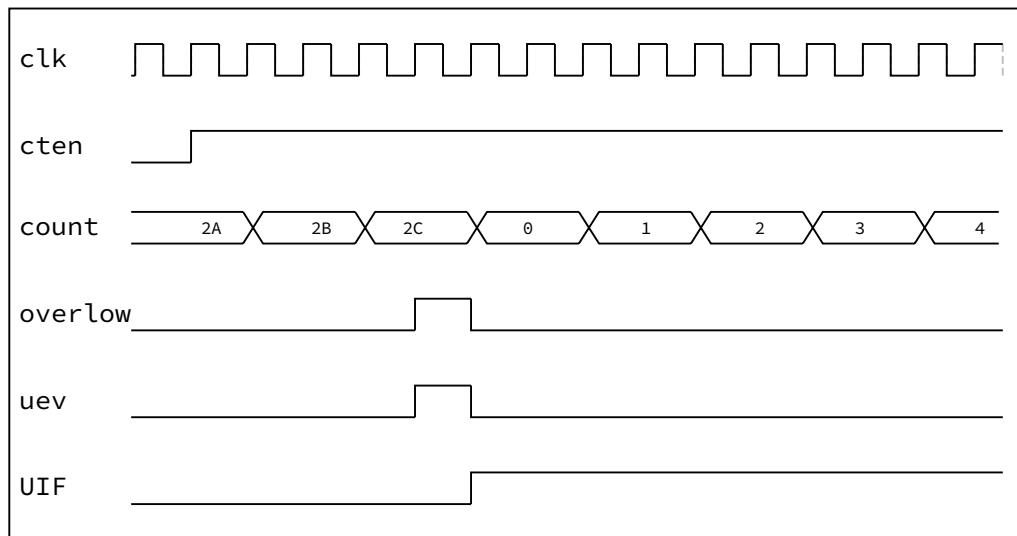


图 14-16 带预分频的上计数

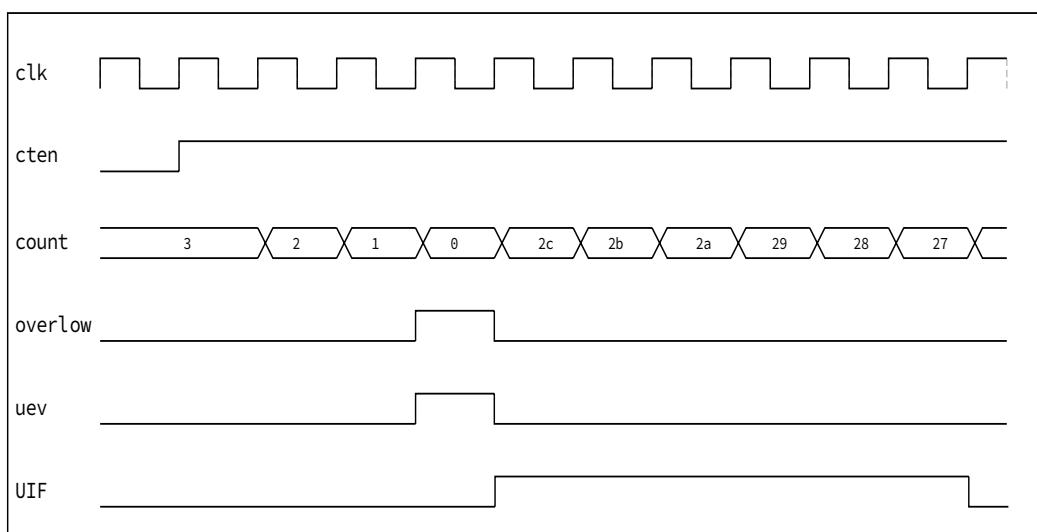


图 14-17 不带预分频的下计数

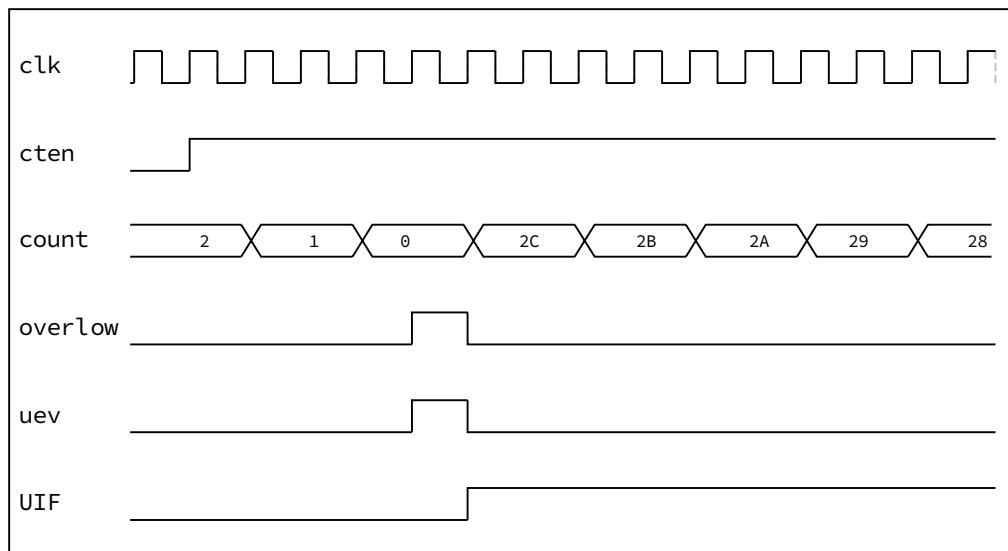


图 14-18 带预分频的下计数

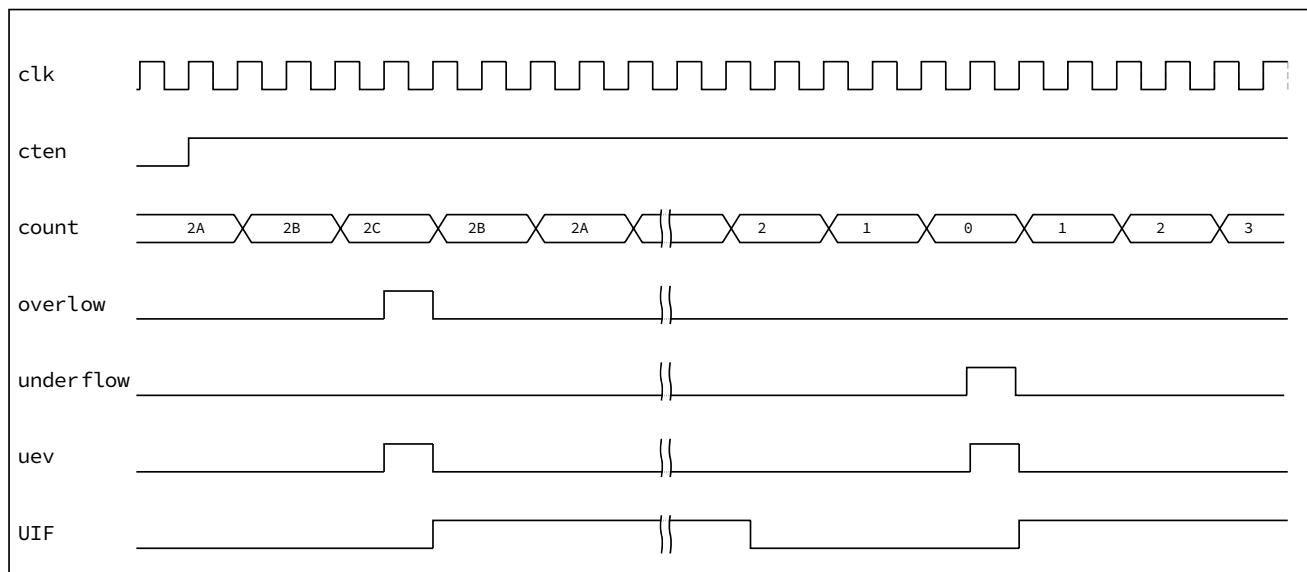


图 14-19 带预分频的上下计数

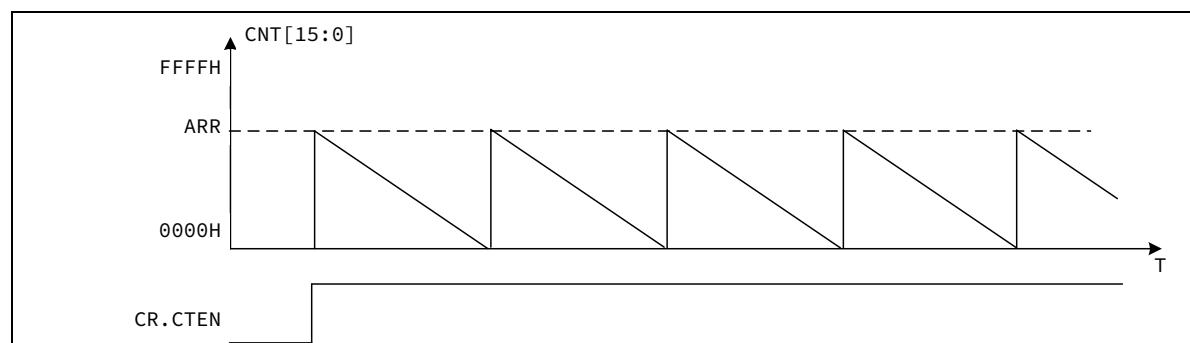


图 14-20 边沿对齐计时器波形(DIR =1)

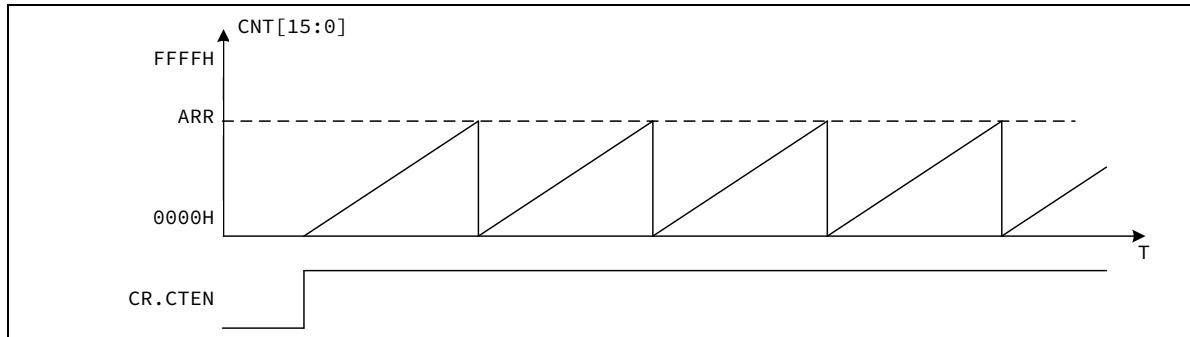
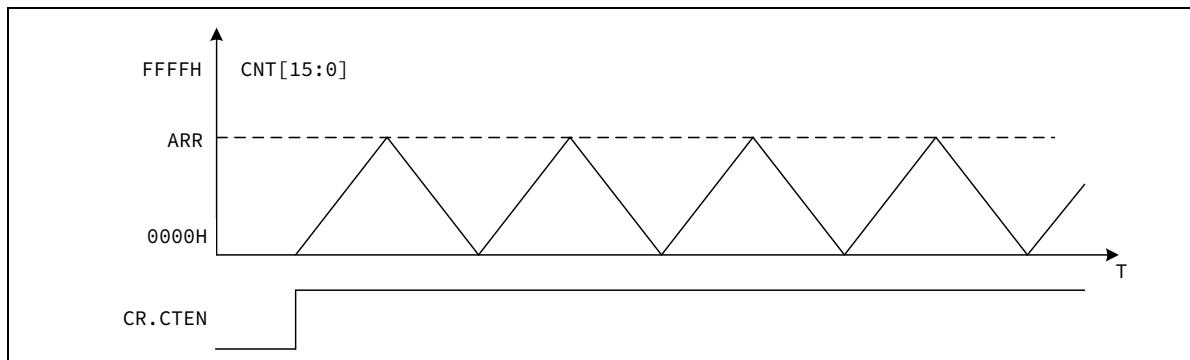


图 14-21 边沿对齐计时器波形 (DIR = 0)

模式 3 为三角波计数波形，计数方向控制位只读，不可以更改计数方向。

中心对齐（三角波）模式下 CR.DIR 方向位是只读的。写值无效。从其他模式切换到中心对齐模式 DIR 自动清 0。软件事件更新和从模式外部触发复位模式 DIR 自动清零。



14.2.6.3 Repeat Count

重复计数器使用计数器的溢出进行向下计数。计数到 0 时，即计数器发生重复寄存器设置的值加一次溢出时。当缓存寄存器使能时，周期重载寄存器更新到周期缓存寄存器。比较模式下比较寄存的值更新到比较缓存寄存中。

重复计数器在下面条件成立时递减

- 上计数模式下每次计数器溢出时
- 下计数模式下每次计数器下溢时
- 三角波模式下每次上溢出和每次下溢出时

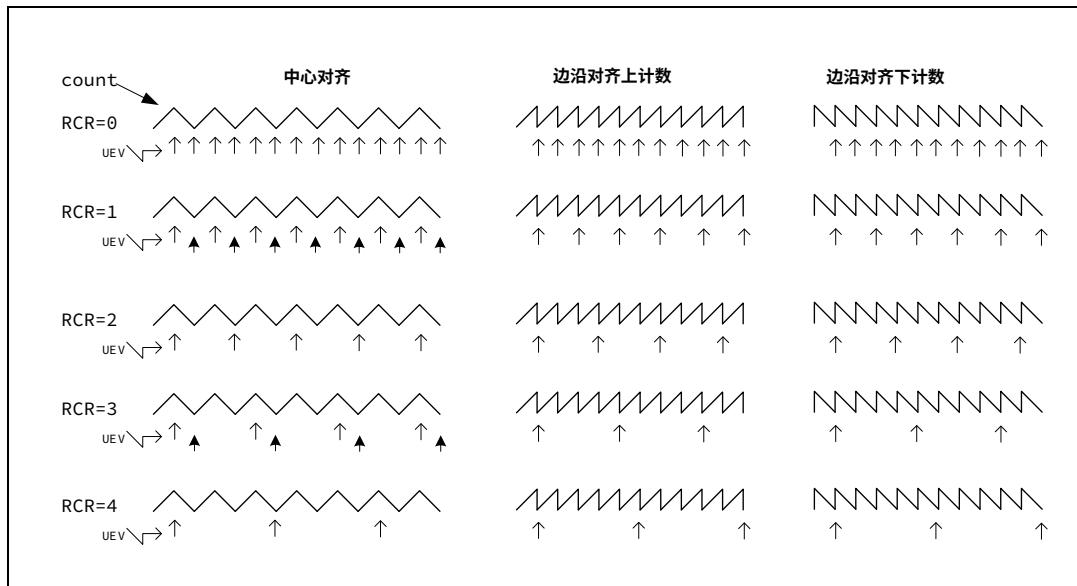


图 14-23 重复计数器产生更新时序

除了上下溢出通过重复计数器可以产生事件更新 UEV 外，还可以通过写寄存器 CR.UG 产生软件及从模式复位事件更新 UEV；这时需要配置 CR.URS。

14.2.6.4 Data Cache

自动重载数据 ARR 与比较寄存器都可以配置缓存功能，当缓存功能有效时，当发生 UEV 事件更新时，写入的周期值 ARR 与比较值 CCR 才会生效。

自动重载值在不同计数模式下的更新时序图如下：

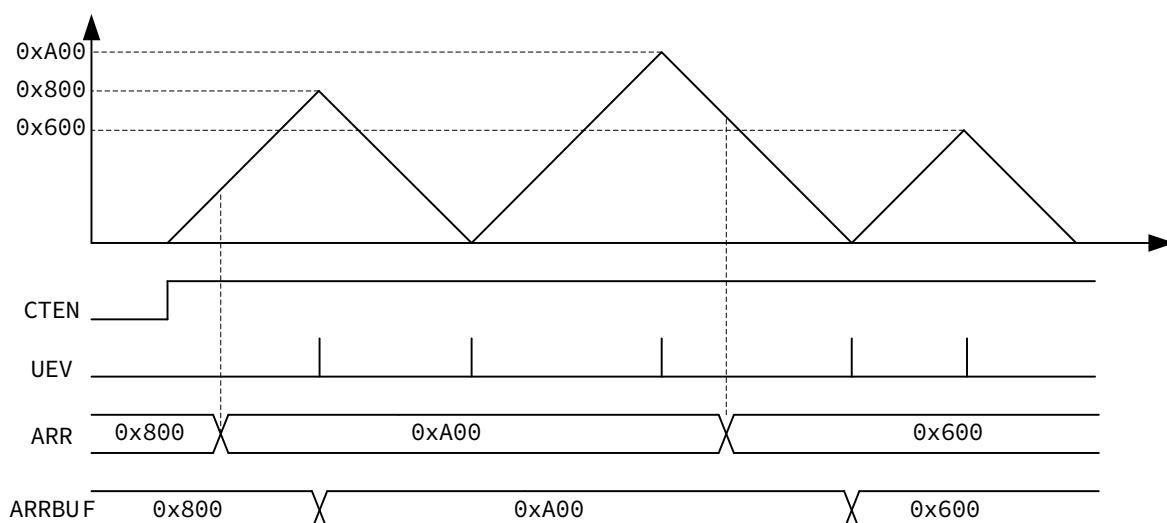


图 14-24 三角波模式下缓存使能

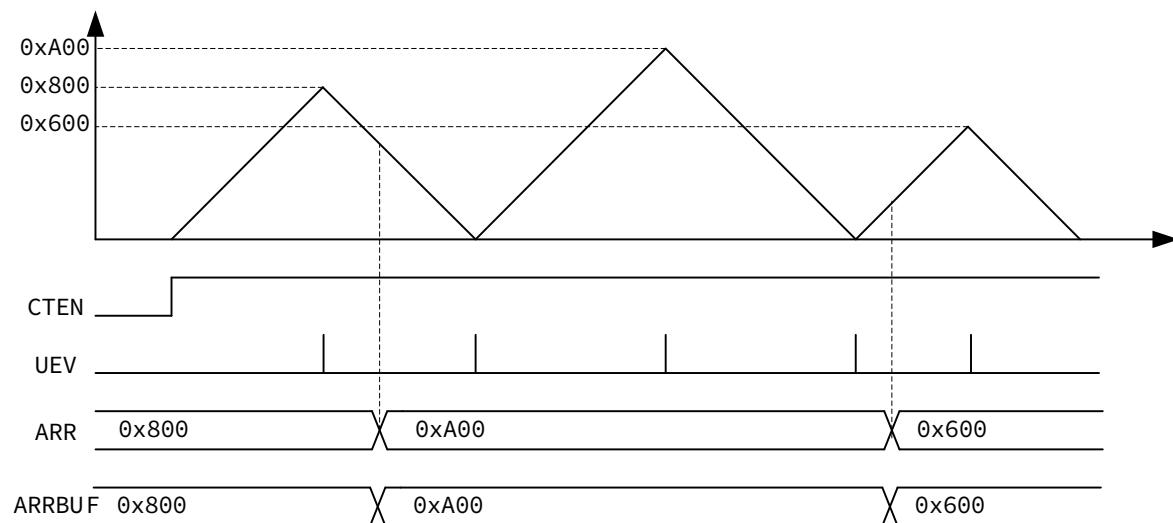


图 14-25 三角波模式下缓存无效

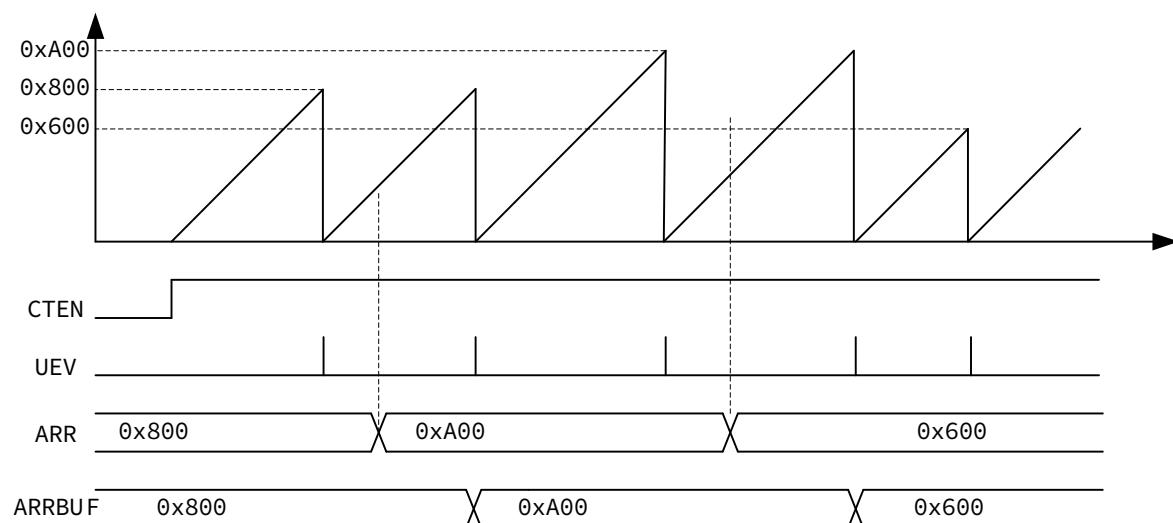


图 14-26 锯齿波模式下上计数缓存使能

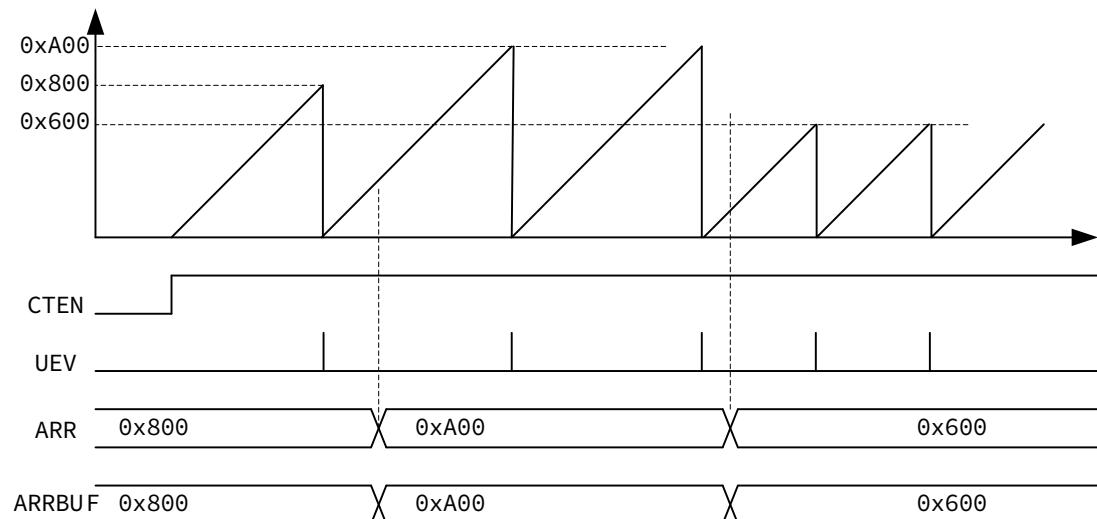


图 14-27 锯齿波模式下上计数缓存无效

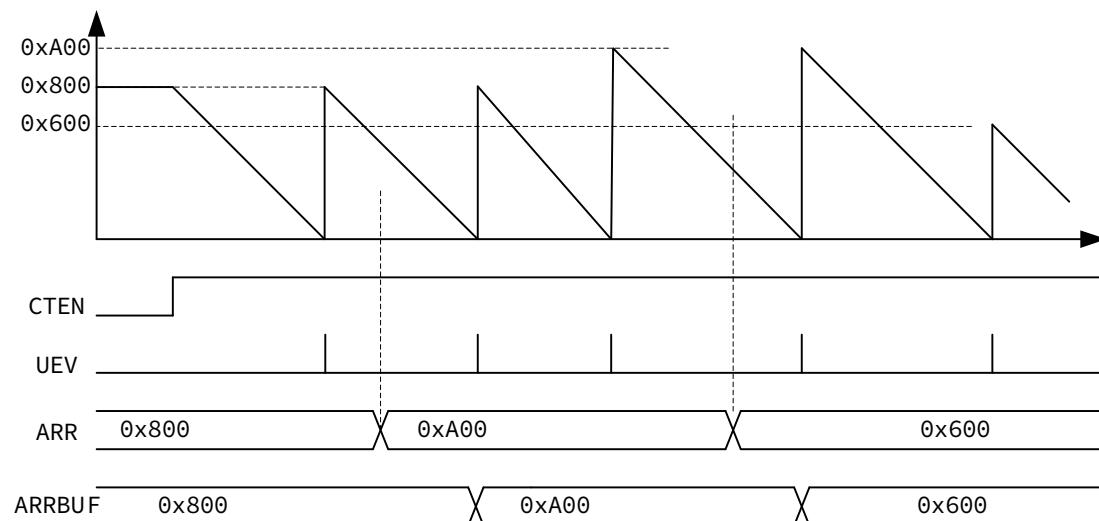


图 14-28 锯齿波模式下计数缓存使能

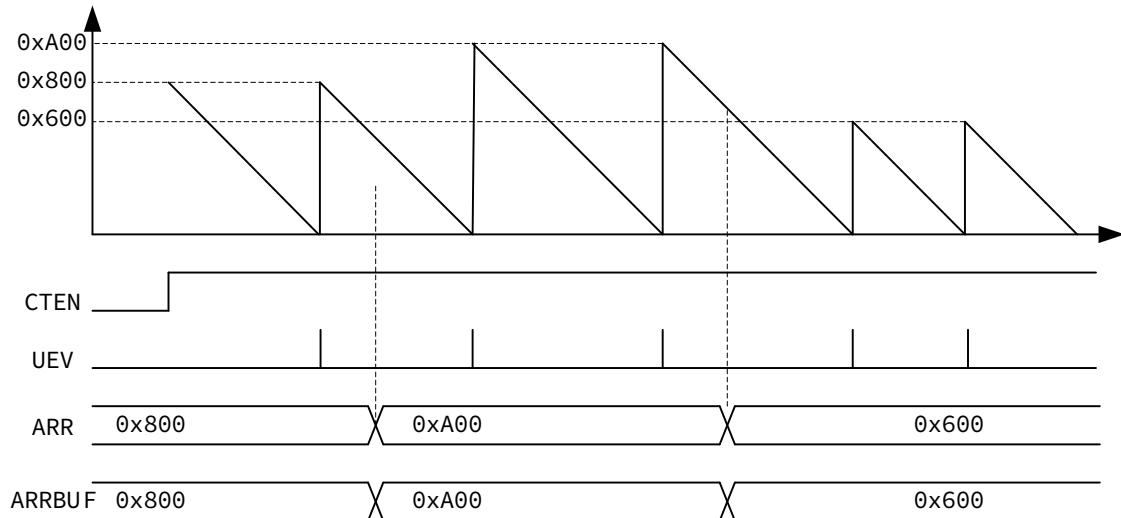


图 14-29 锯齿波模式下计数缓存无效

在三角波模式与锯齿波上计数模式时，如果缓存不使能，更改的 ARR 时，当前计数器的值要小于要更改的 ARR 周期值，否则当前周期会计数到 0xFFFF。

比较缓存与周期缓存更新状态一致，这里不一一列出时序图。

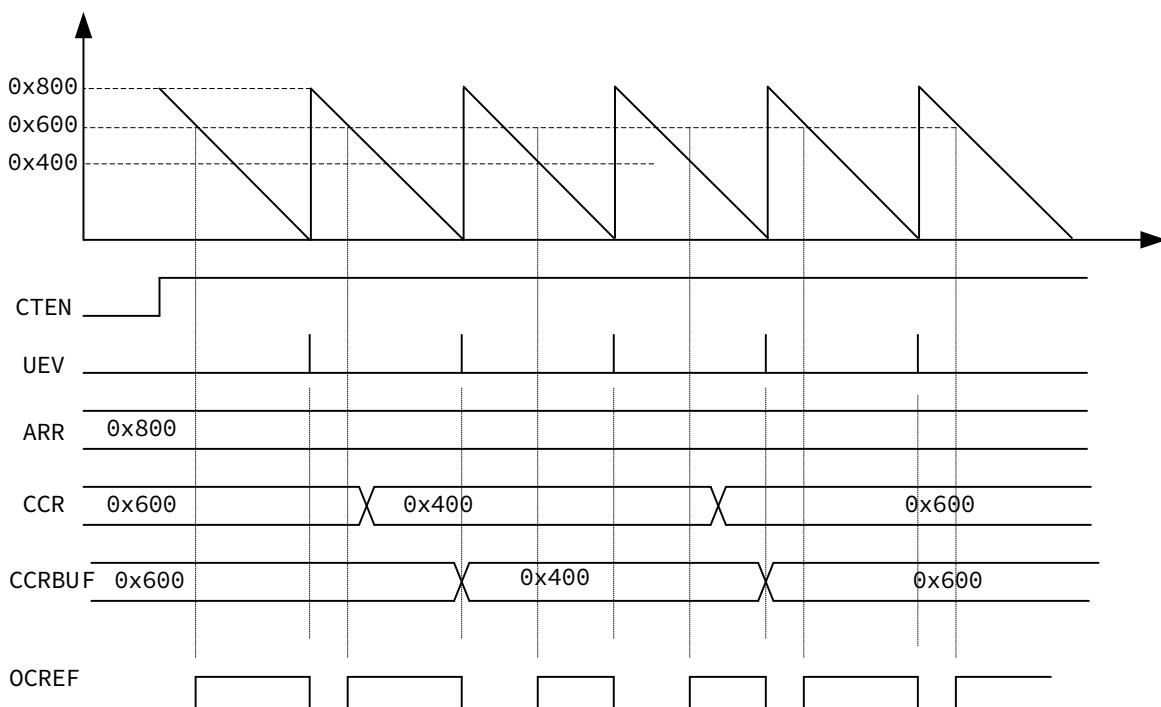


图 14-30 锯齿波模式下计数比较缓存使能

14.2.6.5 Comparison Output OCREF

比较输出 OCREFA 可以配置为单点比较，使用比较寄存器 CCRA 控制 OCREFA 的输出；OCREFA 的比较输出也可以配置为双点比较，使用比较寄存器 CCRA,CCRB 一起控制 OCREFA 的比较输出。

OCREFB 的比较输出只能使用单点比较，使用比较寄存器 CCRB 控制 OCREFB 的比较输出。

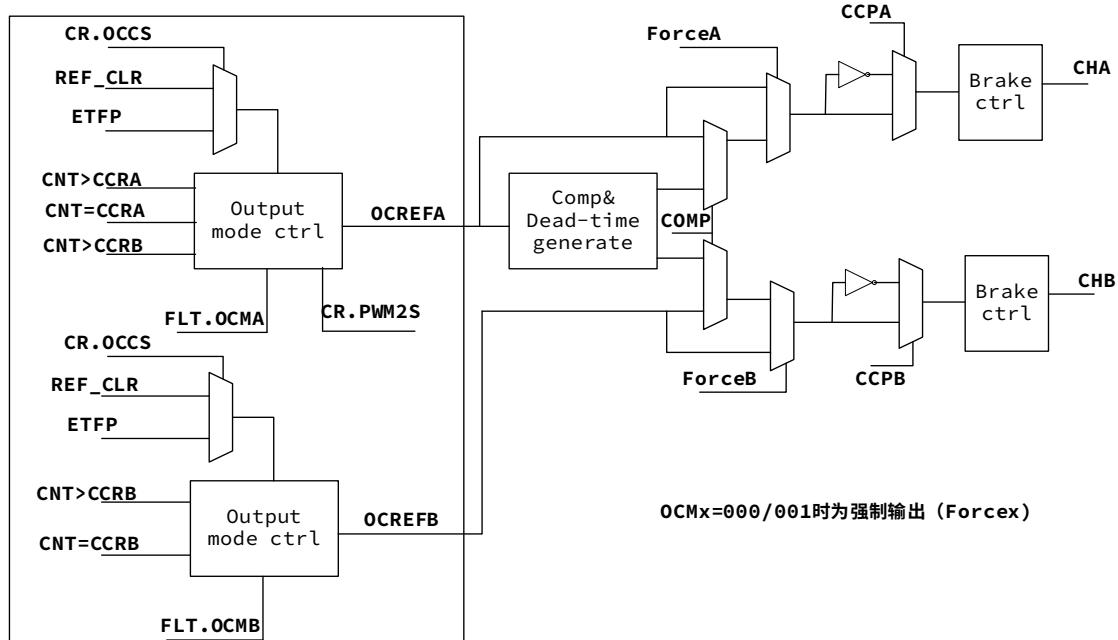


图 14-31 OCREF 输出框图

OCREF 输出使用 OCMx 选择

000：强制为 0

001：强制为 1

010：比较匹配时强制为 0

011：比较匹配时强制为 1

100：比较匹配时翻转

101：比较匹配时输出一个计数周期的高电平

110：PWM 模式 1

单点比较：

上计数时 CNT<CCRxy 输出高，下计数时 CNT>CCRxy 输出为低电平

双点比较：

- 1) 锯齿波上计数 CCRxA<CNT≤CCRxB 输出为低电平
- 2) 锯齿波下计数 CCRxA<CNT≤CCRxB 输出为高电平

3) 三角波上计数 CNT<CCRxA 输出高, 下计数 CNT>CCRxB 为低电平

111: PWM 模式 2

单点比较:

上计数时 CNT<CCRxy 输出低, 下计数时 CNT>CCRxy 输出为高 电平

双点比较:

1) 锯齿波上计数 CCRxA≤CNT<CCRxB 输出为高电平

2) 锯齿波下计数 CCRxA≤CNT<CCRxB 输出为低电平

3) 三角波上计数 CNT<CCRxA 输出低, 下计数 CNT>CCRxB 为高电平

注: 强制输出有高优先级, 当强制输出有效时, 互补输出控制无效。

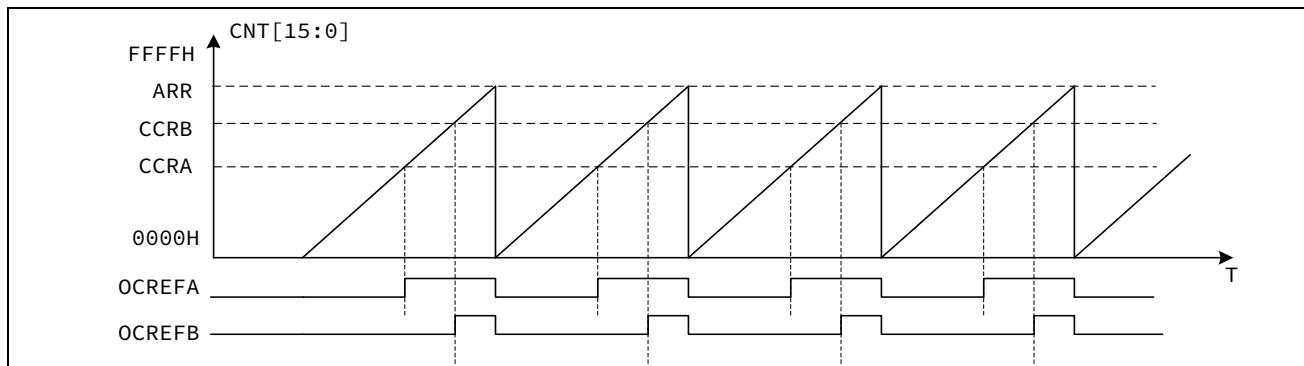


图 14-32 锯齿波计数单点比较 OCREF 输出波形 (OCMx=111)

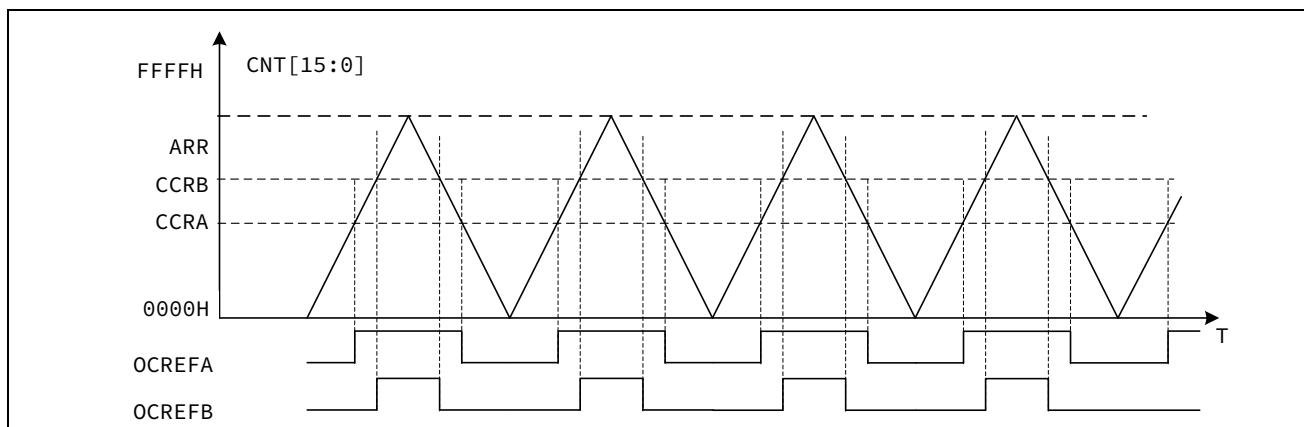


图 14-33 三角波计数单点比较 OCREF 输出波形 (OCMx=111)

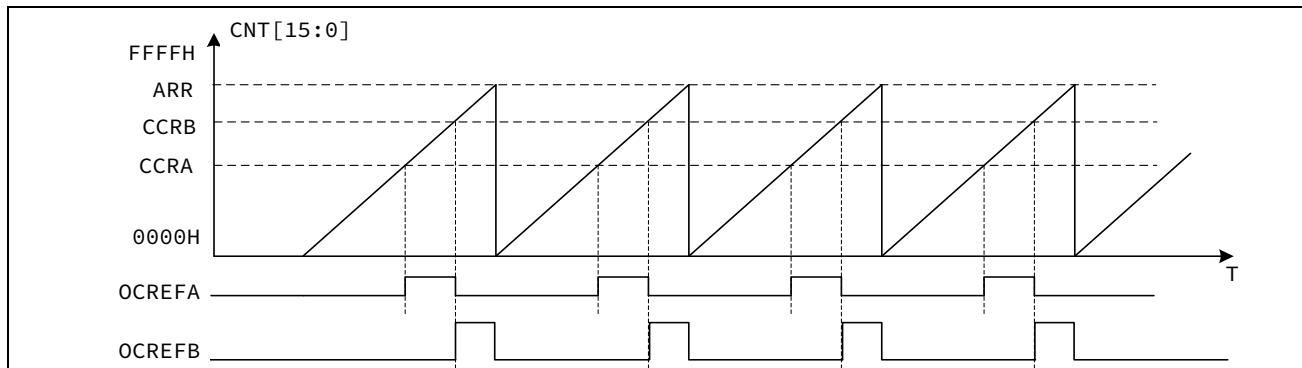


图 14-34 锯齿波计数双点比较 OCREF 输出 (OCMx=111)

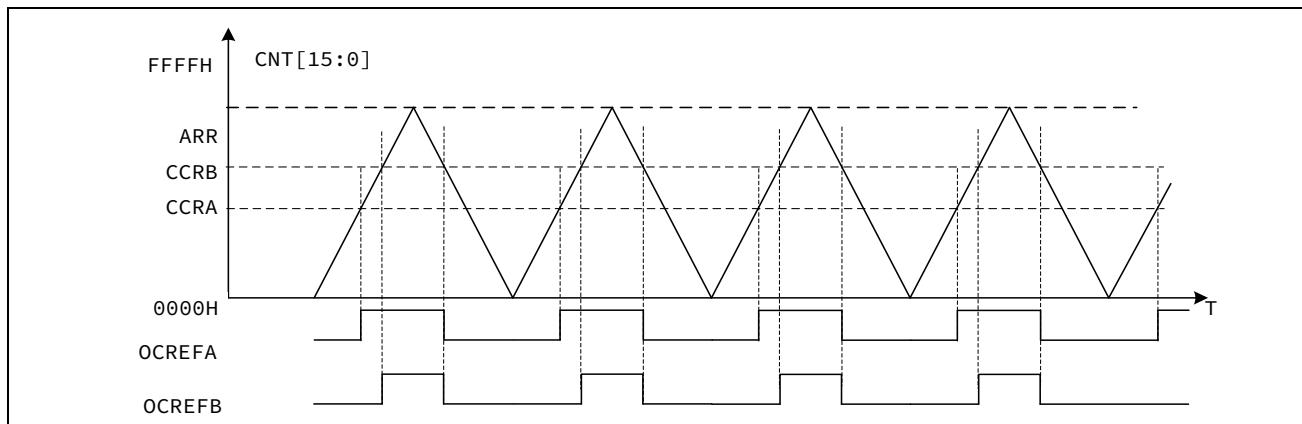


图 14-35 三角波计数双点比较 OCREF 输出 (OCMx=111)

14.2.6.6 Independent PWM Output

由 OCREFA 控制 CHA 的输出，OCREFB 控制 CHB 的输出。通过 CRCHx.CCPA, CRCHx.CCPB 可以控制 CHA, CHB 输出的反向。

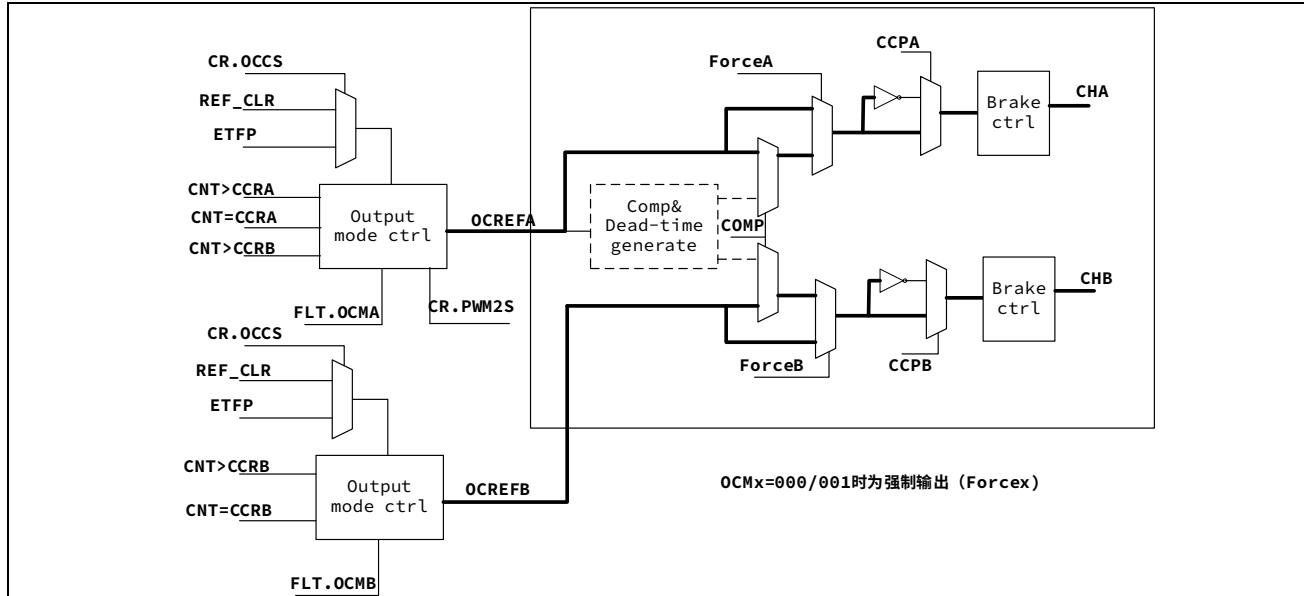


图 14-36 独立 PWM 输出框图

PWM 输出与 OCREF 关系

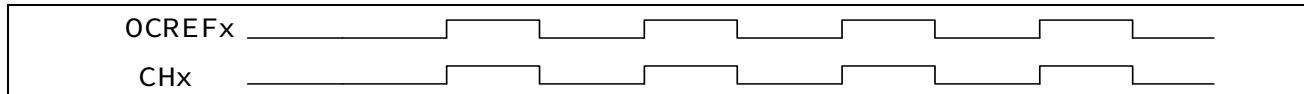


图 14-37 CCPx=0 时 PWM 输出波形

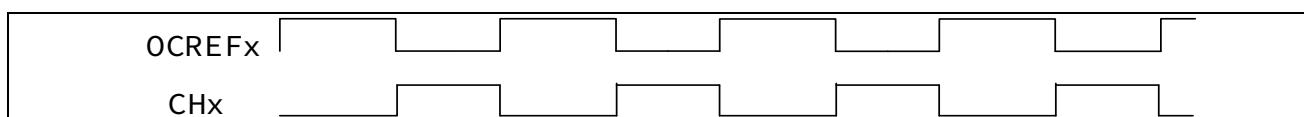


图 14-38 CCPx=1 时 PWM 输出波形

14.2.6.7 Complementary PWM Output

由 OCREF_A 控制 CHA 的输出，OCREF_A 同时控制 CHB 的输出。比较寄存器 CCRxB 可以作为专用比较控制 ADC 触发。

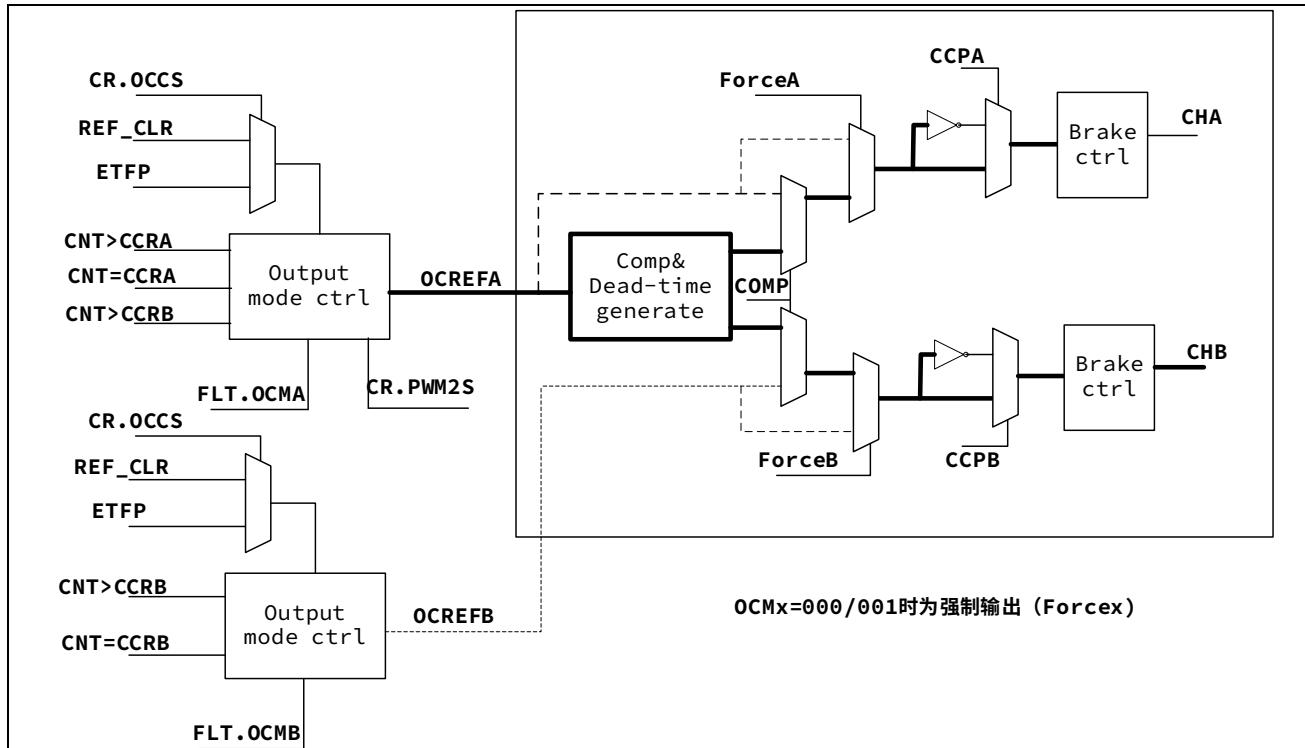


图 14-39 互补 PWM 输出框图

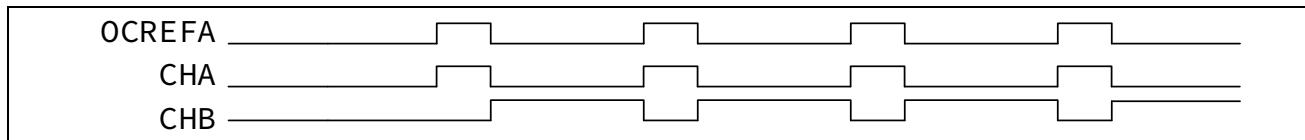


图 14-40 互补 PWM 输出波形图

14.2.6.8 PWM output with dead zone

在互补 PWM 输出模式下可以设置死区功能。

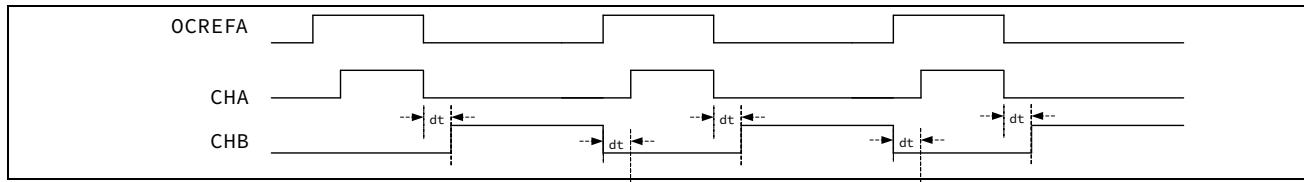
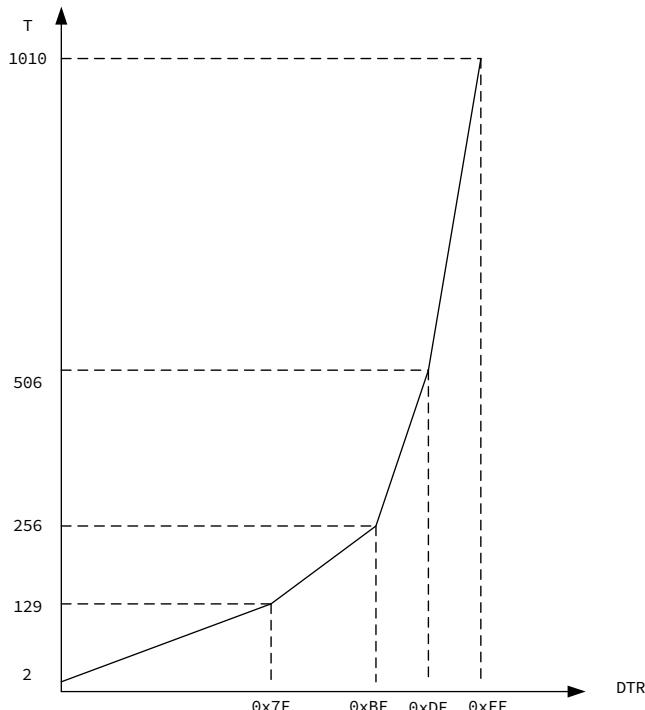


图 14-41 互补 PWM 输出波形图

死区时间使用 8 位 DTR 控制，死区时间 dt 与 DTR 的关系如下

DTR[7] = 0	T = DTR[6:0] + 2	2-129	step=1
DTR[7:6] = 10	T = {DTR[5:0] + 64} * 2 + 2	130-256	step=2
DTR[7:5] = 110	T = {DTR[4:0] + 32} * 8 + 2	258-506	step=8
DTR[7:5] = 111	T = {DTR[4:0] + 32} * 16 + 2	514-1010	step=16



100M clock 死区时间由2.56us调整为10.1us

图 14-42 死区时间

14.2.6.9 Single Pulse Output

单脉冲模式 (ONE SHOT) 是上述 PWM 模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。将

TIMx_M23CR 寄存器中的 ONESHOT 位置 1，即可选择单脉冲模式。这样，发生下一更新事件 UEV 时，计数器将自动停止。

单脉冲模式在锯齿波下计数模式计数器初值不要设置为 0，上计数模式计数值不要设置到大于等于 ARR。

只有当比较值与计数器初始值不同时，才能正确产生输出脉冲。启动前（定时器等待触发时），必须进行如下配置：

- 递增计数时： $CNT < CCRxy \leq ARR$ （特别注意， $0 < CCRxy$ ），
- 递减计数时： $CNT > CCRxy$ 。

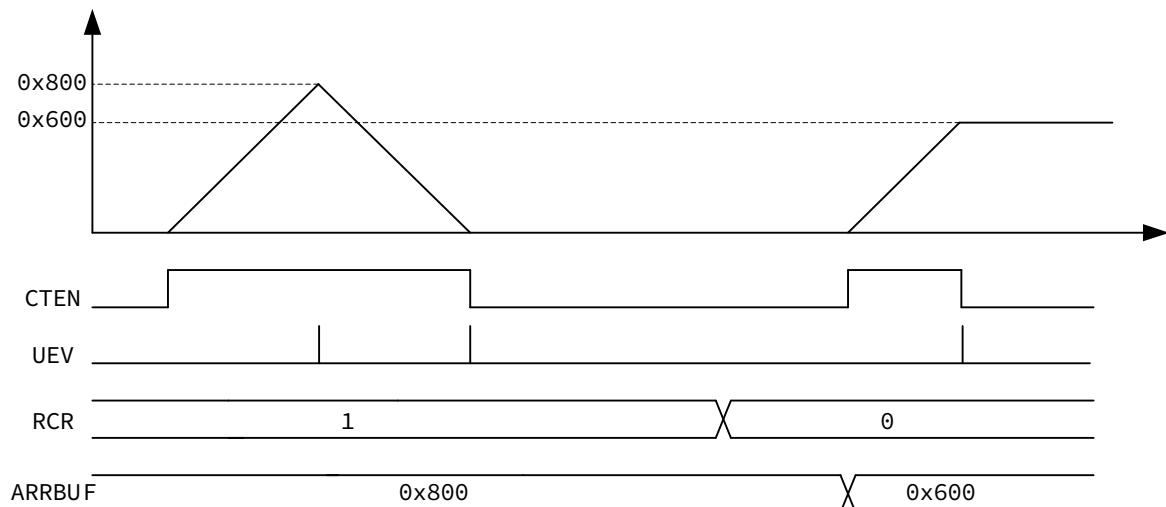


图 14-43 三角波模式单脉冲计数

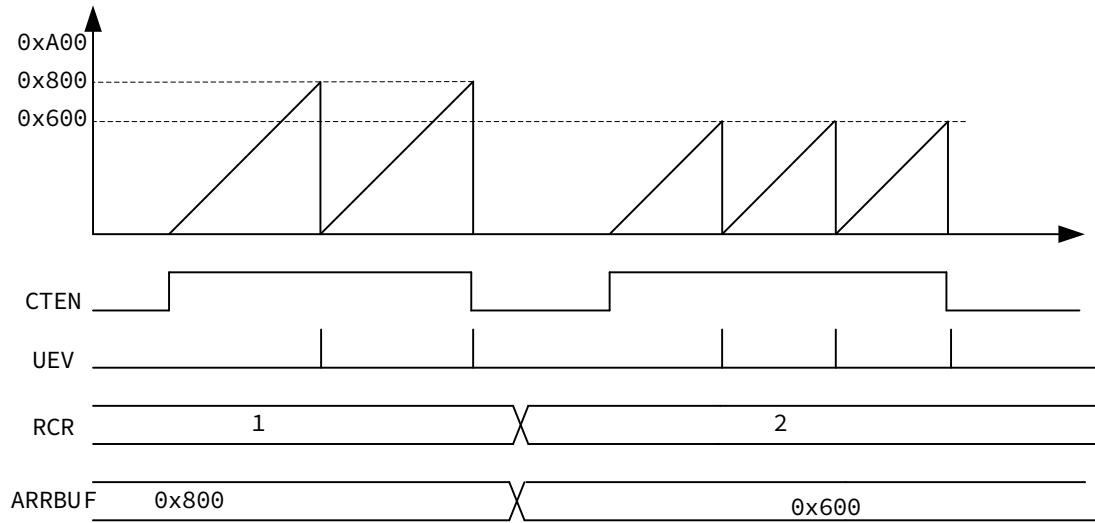


图 14-44 锯齿波上计数单脉冲模式

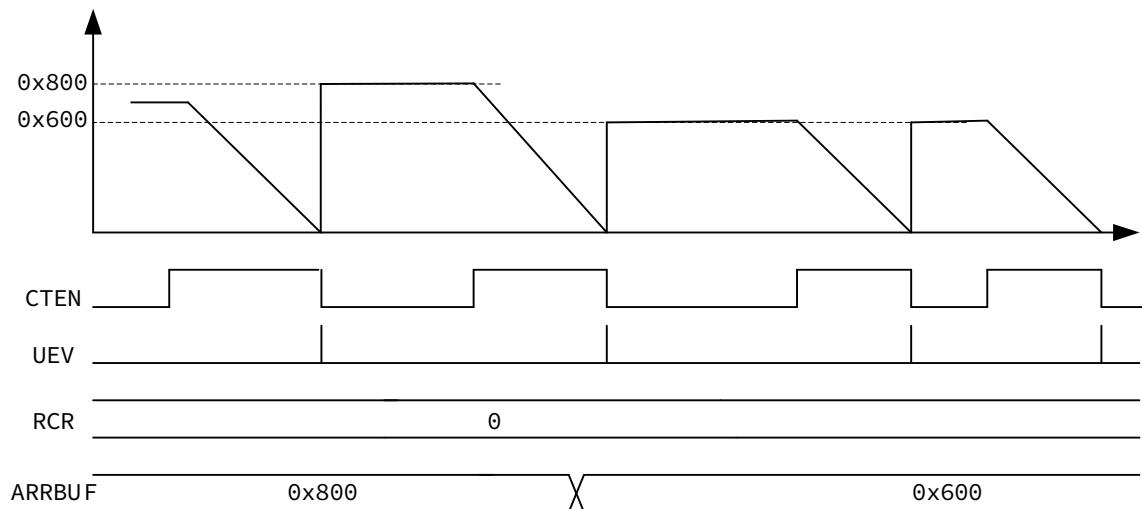


图 14-45 锯齿波下计数单脉冲模式

14.2.6.10 Compare Interrupt

锯齿波比较匹配会将 IFR 寄存器相应的标志置 1，如果中断使能 CRCHx.CIEy ($x=0,1,2; y=A,B$)将会触发中断。

三角波比较匹配可单独选择上升计数比较匹配，下降计数比较匹配或者两者比较都匹配。

比较 A 比较匹配当计数值与 CCRxA 相等时，统一使用 CR.CIS 控制。

CR.CIS=2'b00 时 比较匹配无输出

CR.CIS=2'b01 时 上计数时比较匹配

CR.CIS=2'b10 时 下计数时比较匹配

CR.CIS=2'b11 时 上下计数时比较都匹配

比较 B 比较匹配当计数值与 CCRxB 相等时，不同通道可以使用 CRCHx.CISB 单独控制。

CRCHx.CISB=2'b00 时 通道 x 比较匹配无输出

CRCHx.CISB=2'b01 时 通道 x 上计数时比较匹配

CRCHx.CISB=2'b10 时 通道 x 下计数时比较匹配

CRCHx.CISB=2'b11 时 通道 x 上下计数时比较都匹配

B 通道比较匹配单独控制为了更灵活的触发 ADC。详见 Timer 触发 ADC 章节。

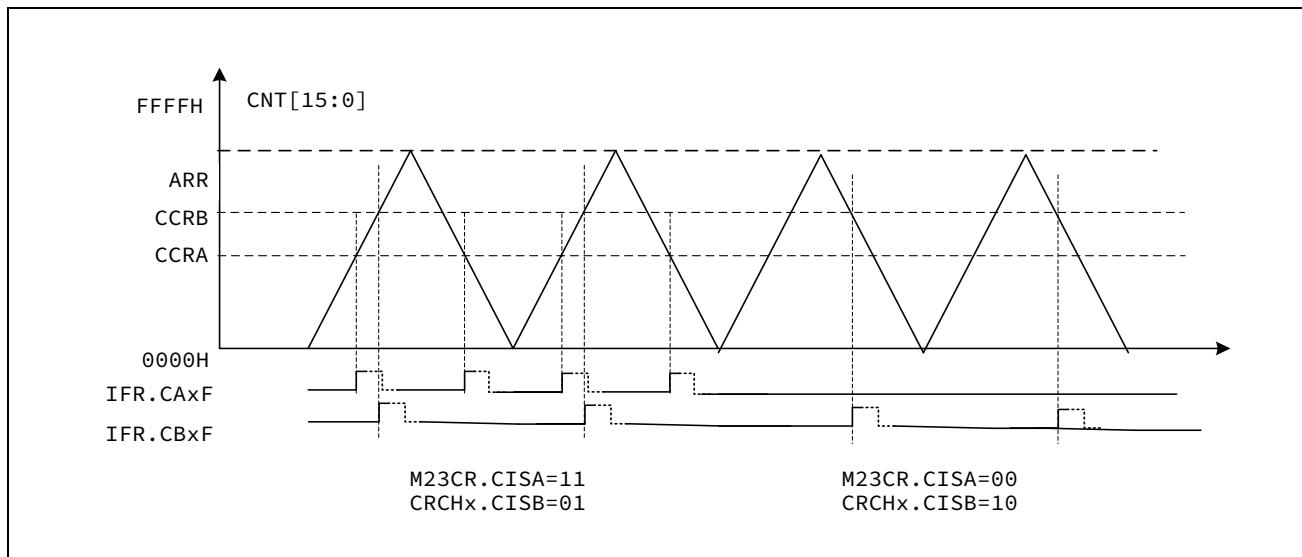


图 14-46 中断示意图

14.2.6.11 Capturing Input

CR.MODE=2/3

在三角波计数或者锯齿波计数模式下都可以设置捕获功能，可以设置捕获的电平边沿，当发生捕获时捕获的值存入比较捕获寄存器并产出捕获中断。

每个通道的比较捕获功能可以单独设置，通过寄存器 CRCHx.CSA/CSB 选择。

每个通道的捕获边沿可以单独设置，通过寄存器 CRCHx.CFy/CRy(x=0/1/2;y=A/B)选择捕获触发的边沿。

当捕获发生后捕获标志未清除前再次发生捕获动作，会产生捕获数据覆盖标志。

定时器未启动时如果有效的捕获边沿也会产生捕获标志及捕获动作。

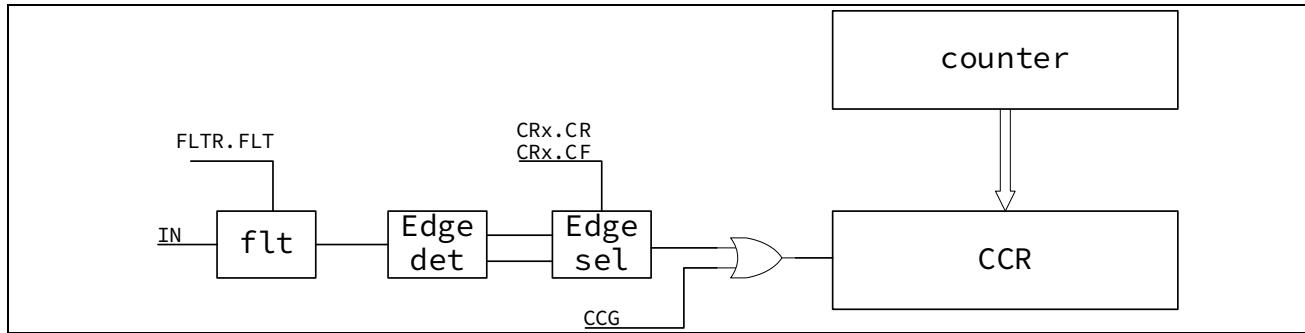


图 14-47 捕获功能框图

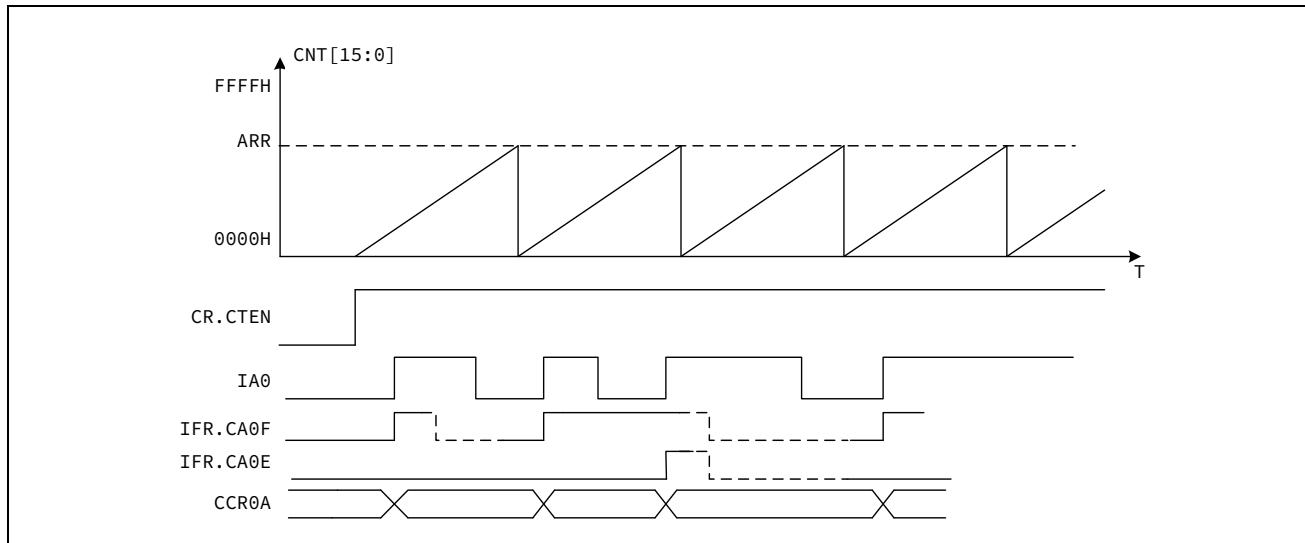


图 14-48 捕获时序图

CH0A 的捕获输入通过 MSCR.IA0S 选择 CH0A 输入还是 CH0A, CH1A, CH2A 的异或输入。

IA0S	0	1
Timer0/1/2	CHA0	CHA0 ETR GATE异或输入
Timer3	CHA0	CHA0 CHA CHA2异或输入

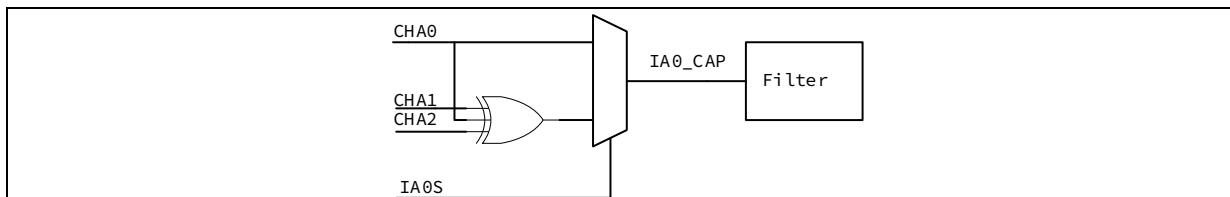


图 14-49 CHA 端口选择

CH0B 的捕获输入通过 MSCR.IB0S 选择 CH0B 输入还是内容 MSCR.TS 选择的信号。

IB0S	0	1																		
Timer0/1/2/3	CHB0	内部触发 MSCR.TS 选择信号 <table border="1"> <tr> <td>MSCR.TS</td><td>通道 B 捕获输入</td></tr> <tr> <td>000</td><td>ETR 滤波相位选择输出信号, 滤波相位选择可配</td></tr> <tr> <td>001</td><td>内部互联 ITR0</td></tr> <tr> <td>010</td><td>内部互联 ITR1</td></tr> <tr> <td>011</td><td>内部互联 ITR2</td></tr> <tr> <td>100</td><td>内部互联 ITR3</td></tr> <tr> <td>101</td><td>CH0A 边沿</td></tr> <tr> <td>110</td><td>CH0A 滤波输出信号, 滤波功能可配</td></tr> <tr> <td>111</td><td>CH0B 滤波输出信号, 滤波功能可配</td></tr> </table>	MSCR.TS	通道 B 捕获输入	000	ETR 滤波相位选择输出信号, 滤波相位选择可配	001	内部互联 ITR0	010	内部互联 ITR1	011	内部互联 ITR2	100	内部互联 ITR3	101	CH0A 边沿	110	CH0A 滤波输出信号, 滤波功能可配	111	CH0B 滤波输出信号, 滤波功能可配
MSCR.TS	通道 B 捕获输入																			
000	ETR 滤波相位选择输出信号, 滤波相位选择可配																			
001	内部互联 ITR0																			
010	内部互联 ITR1																			
011	内部互联 ITR2																			
100	内部互联 ITR3																			
101	CH0A 边沿																			
110	CH0A 滤波输出信号, 滤波功能可配																			
111	CH0B 滤波输出信号, 滤波功能可配																			

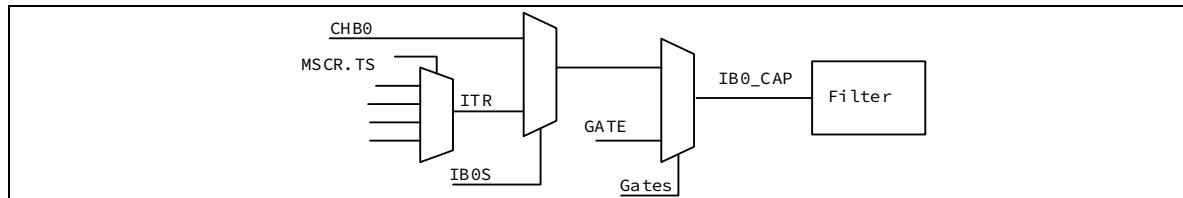


图 14-50 CHB 端口选择

Gates 信号为选择 PWM 互补输出功能生成的信号, 当 PWM 互补输出时, 比较捕获寄存器 CCR0B 没有使用, 自动切换到使用 GATE 作为捕获/比较输出。

14.2.6.12 Setting example

边沿对齐独立 PWM 输出设置

1. 设置模式 CR.MODE=2
2. 设置锯齿波计数方向 CR.DIR
3. 设置 PWM 周期值 ARR
4. 设置计数器初值 (初值必须小于周期值)
5. 设置 PWM 比较值 CCRxA,CCRxB
6. 设置 PWM 输出比较模式 FLT.OCMA,FLT.OCMB 为 6 或 7
7. 清除相关中断标志
8. 使能相应的中断
9. 设置输出极性 FLT.CCPAx,FLT.CCPBx
10. 使能输出 DTR.MOE
11. 使能定时器 CR.CTEN

中心对齐互补 PWM 输出设置

1. 设置模式 CR.MODE=3
2. 设置互补输出 CR.COMP
3. 设置 PWM 周期值 ARR
4. 设置计数器初值 (初值必须小于周期值)
5. 设置 PWM 比较值 CCRxA(CCRxB 不需要设置, PWM 输出与这个寄存器无关)
6. 设置 PWM 输出比较模式 FLT.OCMA FLT.OCMB 为 6 或 7
7. 清除相关中断标志
8. 使能相应的中断
9. 设置输出极性 FLT.CCPAx FLT.CCPBx
10. 使能输出 DTR.MOE
11. 使能定时器 CR.CTEN

三角波非中心对齐带死区互补 PWM 输出设置

1. 设置模式 CR.MODE=3
2. 设置互补输出 CR.COMP
3. 设置两个比较使能 CR.PWM2S
4. 设置 PWM 周期值 ARR
5. 设置计数器初值 (初值必须小于周期值)
6. 设置 PWM 比较值 CCRxA,CCRxB,上计数比较点为 CCRxA,下计数比较点为 CCRxB
7. 设置 PWM 输出比较模式 FLT.OCMA FLT.OCMB 为 6 或 7
8. 清除相关中断标志
9. 使能相应的中断
10. 设置输出极性 FLT.CCPAx FLT.CCPBx
11. 设置死区使能 DTR.DTEN
12. 设置死区时间 DTR.DT
13. 使能输出 DTR.MOE
14. 使能定时器 CR.CTEN

捕获功能设置

设置 CH0B 为上升沿捕获功能

1. 选择计数器计数方式，设置 mode=2
2. 选择 CH0B 为捕获模式 CRCHx.CSB=1
3. 根据需要选择输入滤波 FLT.FLTB0
4. 选择捕获的边沿 CRCHx.CRB=1
5. 设置 ARR 改变周期值
6. 启动定时器 CR.CTEN
7. 清除相关中断标志
8. 清除捕获标志，使能相应中断
9. 查询到捕获标志后，读取 CCR0B 获得捕获值

注：定时器未启动时如果有效的捕获边沿也会产生捕获标志及捕获动作。

互补 PWM 输出时使用 GATE 作为 PWM 输出功能

1. 设置为 PWM 互补输出模式，参考“中心对齐互补 PWM 输出设置”
2. 设置 CCR0B 设置 GATE PWM 比较值
3. 设置 FLT.OCMB0=6/7 设置 PWM 输出模式
4. 使能 PWM 输出 DTR.MOE

互补 PWM 输出时使用 GATE 下降沿作为捕获输入功能

1. 设置为 PWM 互补输出模式，参考“中心对齐互补 PWM 输出设置”
2. 设置 GATE 作为捕获输入 CR.CSG
3. 选择 GATE 捕获输入边沿设置 CR.CFG(GATE 作为捕获输入 B 通道滤波设置无效)
4. 启动定时器 CR.CTEN
5. 清除相关中断标志
6. 清除捕获标志，使能相应中断
7. 查询到捕获标志后，读取 CCR0B 获得捕获值

14.2.7 Mode 2/3 Slave Mode

定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

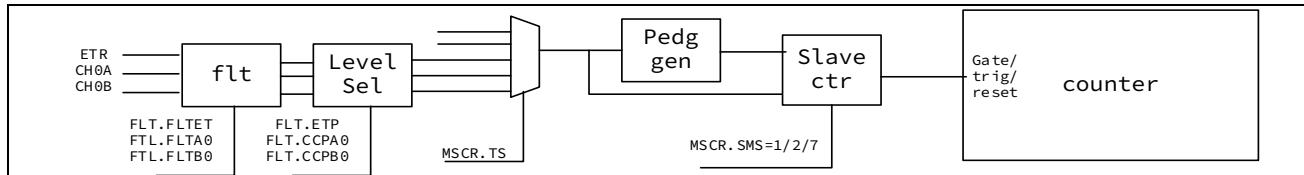


图 14-51 从模式示意图

从模式功能选择通过 MSCR.SMS 选择；

MSCR.SMS	从模式功能选择 001: 复位功能; 010: 触发模式; 111: 门控功能
MSCR.TS	触发选择 000: 端口ETR的滤波相位选择后的信号ETFP, 滤波功能可配置 001: 内部互联信号 ITR0 010: 内部互联信号 ITR1; 011: 内部互联信号 ITR2; 100: 内部互联信号 ITR3; 101: 端口CH0A的边沿信号; 110: 端口CH0A的滤波相位选择后的信号IAFP, 滤波功能可配置 111: 端口CH0B的滤波相位选择后的信号IBFP, 滤波功能可配置

14.2.7.1 Gate Count

按照选中的输入端电平使能计数器。在如下的例子中，计数器只在 CH0A 为低电平时向上计数：

- 配置通道 CH0A 以检测 CH0A 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 FLT.FLTA0=000)。置 FLTR 寄存器中 CCPA0=1 以确定极性(只检测低电平)。
- 置 SMCR 寄存器中 SMS=111，配置定时器为门控模式；置 SMCR 寄存器中 TS=110，选择 CH0A 作为输入源。
- 置 CR 寄存器中 CTEN=1，启动计数器。在门控模式下，如果 CTEN=0，则计数器不能启动，不论触发输入电平如何。只要 CH0A 为低，计数器开始依据内部时钟计数，一旦 CH0A 变高则停止计数。

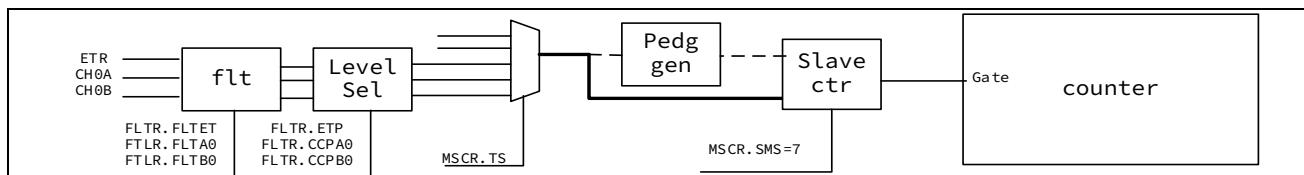


图 14-52 门控功能

14.2.7.2 Trigger function

使用外部触发 (CH0A、CH0B、ETR) 可以使定时器同步启动，使用定时器内部互联信号结合 MSCR.MSM 也可以配置定时器同步启动。触发信号为输入信号的上升沿。也可以使用软件写 CR.TG 启动软件触发功能。

输入端上选中的事件使能计数器。在下面的例子中，计数器在 CH0B 输入的上升沿开始向上计数：

- 配置通道 CH0B 检测 CH0B 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 FLTR.FLTB0=000)。设 FLTR 寄存器中 CCPB0=0 以确定极性(不反向)。
- 置 SMCR 寄存器中 SMS=010，配置定时器为触发模式；置 SMCR 寄存器中 TS=111，选择 CH0B 作为输入源。当 CH0B 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。CH0B 上升沿和计数器启动计数之间的延时，取决于 CH0B 输入端的重同步电路。

注意：如果使用下降沿触发，先选择触发极性，然后再选择模式，否则会产生误触发。

14.2.7.3 Reset Count

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

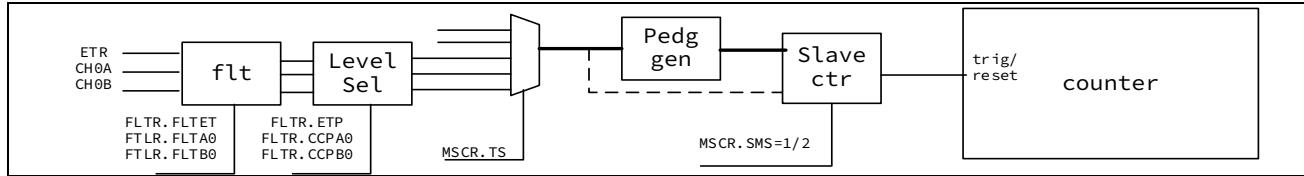


图 14-53 触发和复位功能

14.2.8 Quadrature Encoding Counting Function

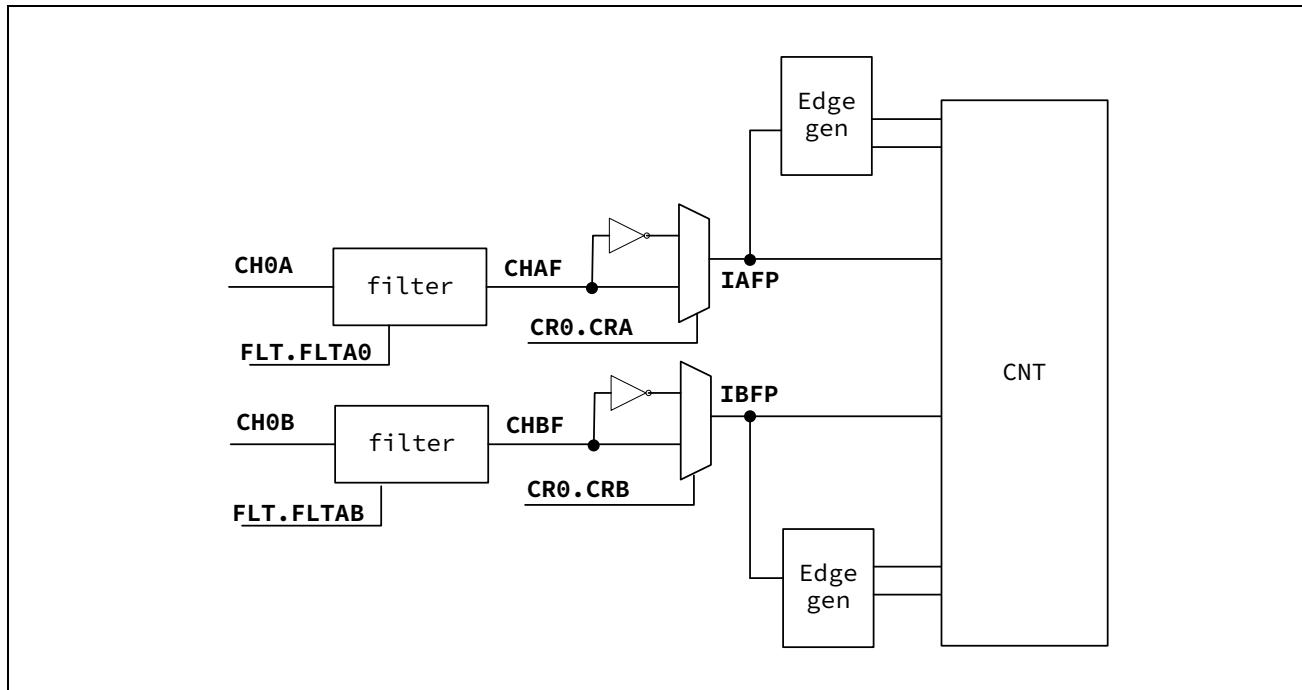
MSCR.SMS=4/5/6，对应正交编码模式的模式 1/2/3。这时计数器根据 IAFP, IBFP 的相位关系进行编码计数。IAFP,IBFP 为端口输入 CH0A, CH0B 的滤波相位选择的信号。

模式 1 使用 CH0A 的边沿计数。模式 2 使用 CH0B 的边沿计数。模式 3 使用 CH0A,CH0B 的边沿都计数。

为了保证计数相位的正确性，需要保证 A/B 输入的相位大于一个脉冲宽度的相位差，A/B 输入脉冲宽度需要大于两个脉冲宽度。

			IAFP		IBFP	
	IBFP	IAFP	Rising	falling	Rising	falling
MOD1	High		Down	Up	-	-
	Low		up	Down	-	-
MOD2		High	-	-	up	Down
		Low	-	-	Down	Up
MOD3	High	High	Down	Up	up	Down
	Low	Low	up	Down	Down	Up

CHAF/CHBF 为端口 CH0A/CH0B 滤波的信号，IAFP/IBFP 为端口滤波相位选择后的信号。



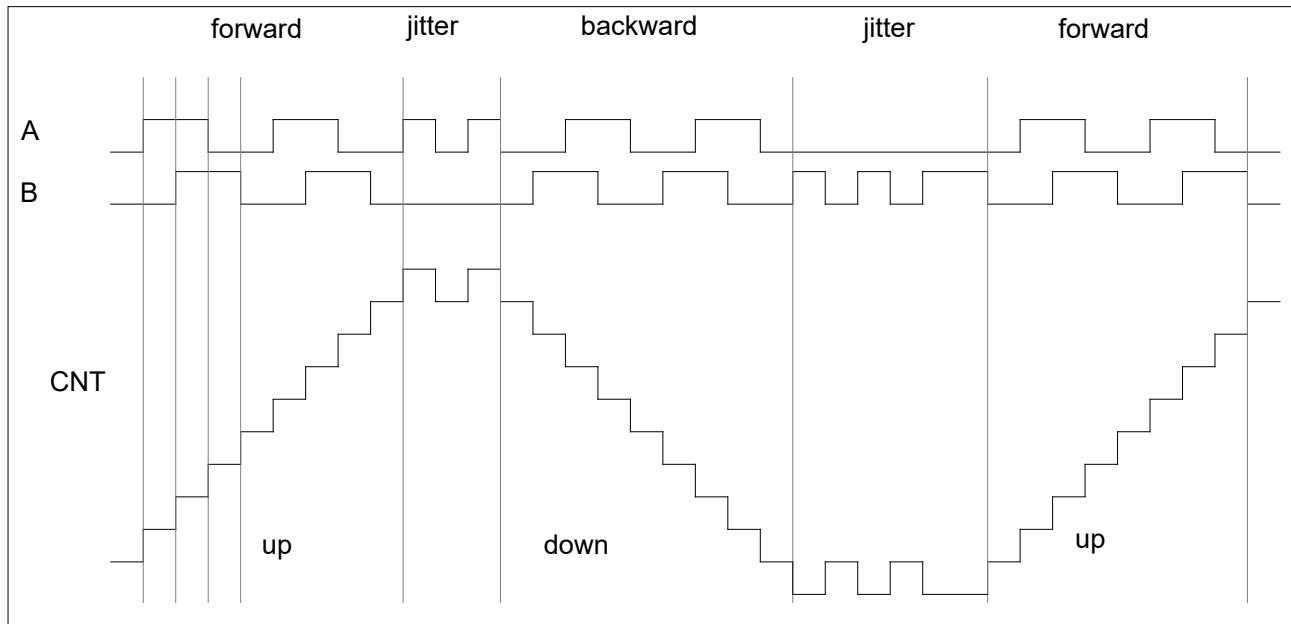


图 14-54 编码计数

14.2.9 ADC triggering

CCMR0A,CCMR1A,CCMR2A 比较匹配可以配置触发 ADC，中心对齐 PWM 时只能通过控制寄存器 CR.CIS 可以选择在上升匹配触发或者下降匹配触发。

CCMR0B,CCMR1B,CCMR2B 比较匹配可以配置触发 ADC,中心对齐 PWM 时,可以通过寄存器 CRx.CISB 分别控制三个匹配触发点 (上升, 下降, 上升下降)。

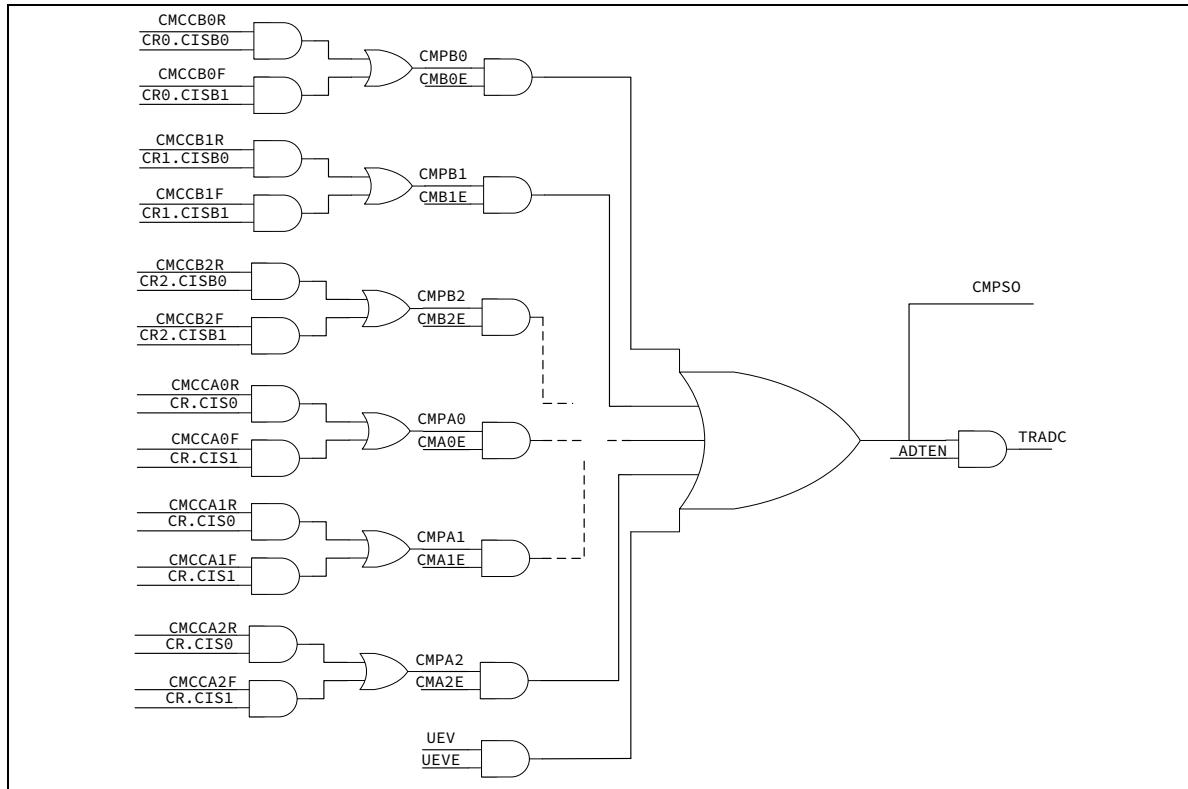


图 14-55 ADC 触发

互补 PWM 输出时 B 比较作为 ADC 触发功能

1. 设置为 PWM 互补输出模式, 参考“中心对齐互补 PWM 输出设置”
2. 设置 CCRxB 设置 ADC 触发比较点
3. 设置 CRCHx.CISB 选择上计数下计数比较匹配 (锯齿波不需要选择)
4. 选择 ADC 触发比较的源 ADTR
5. 使能 ADC 触发 ADTR.ADTE

14.2.10 Brake Control

VC 比较输出可以控制刹车功能，外部 BK 端口输入可控制刹车，系统 fail 可以控制刹车功能。通过 CR.BG 可以实现软件刹车功能，控制输出端口到设定的状态。

两通道的输出的 Tim0/1/2 可以选择使用 Tim0 的刹车端口控制 Tim1/2 的刹车功能，也可以使用各自的刹车输入控制各自的刹车功能。当 DTR.BKSEL 设置为 1 时，Tim1/2 共用 Tim0 的刹车输入。

14.2.11 Timer Interconnection

TRGO 输出信号可以连接到其他定时器的 ITR 信号。连接关系如下：

	ITR0	ITR1	ITR2	ITR3
Timer0	-	TIM1_TRGO	TIM2_TRGO	TIM3_TRGO
Timer1	TIM0_TRGO	-	TIM2_TRGO	TIM3_TRGO
Timer2	TIM0_TRGO	TIM1_TRGO	-	TIM3_TRGO
Timer3	TIM0_TRGO	TIM1_TRGO	TIM2_TRGO	-

定时器 Tim0/1/2 同时启动 PWM 输出设置（使用 Tim0 的 CTEN 触发 Tim1/2）示例

1. 参考 PWM 输出设置，设置定时器 0/1/2 的脉冲调整输出
2. 设置 Tim0 为主模式 MSCR.MSM=1
3. 设置 Tim0 的 CTEN 触发另外两个定时器 MSCR.MMS=1
4. 保持 Tim1/2 的 MSCR.MSM=0
5. 设置 Tim1/2 为触发模式 MSCR.SMS=2
6. 选择 Tim1/2 的触发源为 Tim0 的 TRGO, MSCR.TS=1
7. 最后使能定时器 Tim0，启动定时器 CR.CTEN=1

定时器 Tim0/1/2 同时启动 PWM 输出设置（使用 Tim1 的 UG 触发 Tim0/2）示例

1. 参考 PWM 输出设置，设置定时器 0/1/2 的脉冲调整输出
2. 设置 Tim1 为触发主模式 MSCR.MSM=1
3. 设置 Tim1 的 UG 触发另外两个定时器 MSCR.MMS=0
4. 保持 Tim0/2 的 MSCR.MSM=0
5. 设置 Tim0/2 为触发模式 MSCR.SMS=2
6. 选择 Tim0/2 的触发源为 Tim1 的 TRGO, MSCR.TS=2
7. 最后使能定时器 Tim1，启动定时器 CR.UG=1

14.2.12 GATE Input Interconnection

GATE 输入可以从端口通过 PX_SEL 选择直接输入，也可通过端口功能寄存器 GPIO_TIMGS 选择可以连通到其他模块或端口。

当 TIMx_G=0x0 时，门控 GATE 输入为 PX_SEL 选择的端口输入，当 TIMx_G=0x1~0x7 时，连接其他模块的输入或输出。

	TIM0_g	TIM1_g	TIM2_g	TIM3_g
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART0_RXD	LPUART0_RXD	UART0_RXD	UART0_RXD
010	UART1_RXD	LPUART1_RXD	UART1_RXD	UART1_RXD
011	VC0_OUT	VC0_OUT	VC0_OUT	LPUART0
100	VC1_OUT	VC1_OUT	VC1_OUT	LPUART1
101	PA03	PA08	PA10	VC0_OUT
110	PB08	PB03	PB04	PA06
111	PB15	PB13	PB11	PA11

14.2.13 ETR Input Interconnection

ETR 输入可以从端口直接输入，也可通过端口功能寄存器 GPIO_TIMES 选择可以连通到其他模块或端口。

当 TIMx_E=0x0 时，外部时钟 EXT 输入为 PX_SEL 选择的端口输入，当 TIMx_E=0x1~0x7 时，连接其他模块的输入或输出。

	TIM0_e	TIM1_e	TIM2_e	TIM3_e
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	LPUART0_RXD	UART0_RXD	LPUART0_RXD	UART0_RXD
010	LPUART1_RXD	UART1_RXD	LPUART1_RXD	UART1_RXD
011	VC0_OUT	VC1_OUT	VC0_OUT	VC1_OUT
100	LVD_OUT	LVD_OUT	PCNT_S1	PCNT_S0
101	PA00	PA01	PA04	PA00
110	PA05	PC09	PC04	PA12
111	PA15	PD02	PC08	PA13

14.2.14 CHx capture input interconnection

Timer0/1/2 的 CHA，Timer3 的 CH0A/CH0B 输入可以从端口直接输入，也可通过端口功能寄存器 GPIO_TIMCPS 选择可以连通到其他模块或端口。

当 TIMx_CHy=0x0 时，捕获输入是 PX_SEL 选择的端口输入，当 TIMx_CHy=0x1~0x7 时，连接其他模块的输入或输出。

	TIM0_CHA	TIM1_CHA	TIM2_CHA	TIM3_CH0A	TIM3_CH0B
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART0_RXD	UART1_RXD	LPUART0_RXD	LPUART1_RXD	UART0_RXD
010	PA00	PA00	VC0_OUT	LPUART0_RXD	UART1_RXD
011	PA02	PA02	PA02	PCNT_S0	PCNT_S1
100	PA05	PA06	PA07	VC0_OUT	VC1_OUT
101	PA15	PB08	PB08	PA08	PA07
110	PB06	PB10	PB09	PB03	PB04
111	PB14	PB13	PC06	PB06	PB13

14.2.15 DMA

Timer 支持软件和硬件触发 DMA 进行数据传输。支持数据从其他位置写入定时器，或从定时器读出写入其他位置。可应用于数据捕获后数据的自动搬运和更改周期值或占空比的脉冲宽度的自动调整。每个定时器有两个 DMA 请求，A 请求触发源可选择比较捕获 A，外部触发，B 请求触发源可选择比较捕获 B，事件更新。

IDREQ	Interrupt Signal of Peripheral
18	TIM0A
19	TIM0B
20	TIM1A
21	TIM1B
22	TIM2A
23	TIM2B
24	TIM3A
25	TIM3B

TIMxA 的触发源可选择比较捕获 A,ETRIG

TIMxB 的触发源可选择比较捕获 B,UEV

14.2.15.1 Setting example

捕获数据 DMA 数据传输

1. 使能 DMA
2. 使能 DMA 通道使能
3. 选择定时器 DMA 的通道
4. 设置传输类型、传输长度、传输方式
5. 设置源起始地址，目标起始地址
6. 设置源地址、目标地址的递增方式
7. 根据需要使能 DMA 中断
8. 参考数据捕获流程设置数据捕获
9. 使能 CRCHx.CDy 使能捕获数据触发 DAM 传输。

Note：源地址设置为捕获通道寄存器，源地址固定，目标地址加。

脉冲调整 DMA 数据传输

1. 使能 DMA
2. 使能 DMA 通道使能
3. 选择定时器 DMA 的通道
4. 设置传输类型、传输长度、传输方式
5. 设置源起始地址，目标起始地址
6. 设置源地址、目标地址的递增方式
7. 根据需要使能 DMA 中断
8. 参考脉冲调整输出设置脉冲调制输出
9. 根据需要选择硬件触发 DMA 的条件（UEV 触发或者比较匹配触发）

Note：目标地址设置为捕获通道寄存器，源地址变化，目标地址固定。

14.3 Timer Register Description

Timer	基地址	描述
Timer0	0x40000C00	Timer0基地址
Timer1	0x40000D00	Timer1基地址
Timer2	0x40000E00	Timer2基地址
Timer3	0X40005800	Timer3基地址

寄存器	偏移地址	描述
TIMx_ARR	0X000	Timer重载寄存器/周期
TIMx_CNT	0X004	Timer 16位模式计数寄存器
TIMx_CNT32	0X008	Timer 32位模式计数寄存器 注: TIMx_CNT32是TIMx_ARR与TIMx_CNT的重映射寄存器。为了方便32位定时器写操作。
TIMx_M0CR	0X00C	Timer模式0控制寄存器 (按不同模式进行描述)
TIMx_M1CR	0X00C	Timer模式1控制寄存器 (按不同模式进行描述)
TIMx_M23CR	0X00C	Timer模式23控制寄存器 (按不同模式进行描述)
TIMx_IFR	0X010	Timer中断标志
TIMx_ICLR	0X014	Timer中断清除寄存器
TIMx_MSCR	0X018	主从模式控制
TIMx_FLTR	0X01C	滤波控制
TIMx_ADTR	0X020	ADC触发控制
TIMx_CRCH0	0x024	比较单元0控制寄存器
TIMx_CRCH1	0X028	比较单元1控制寄存器
TIMx_CRCH2	0x02C	比较单元2控制寄存器
TIMx_DTR	0X030	死区寄存器
TIMx_RCR	0X034	重复计数寄存器
TIMx_ARRDM	0X038	Timer重载寄存器/周期映射地址
TIMx_CRO	0x024	控制寄存器
TIMx_CCR0A	0X03C	比较0A寄存器
TIMx_CCR0B	0x040	比较0B寄存器
TIMx_CCR1A	0X044	比较1A寄存器
TIMx_CCR1B	0X048	比较1B寄存器
TIMx_CCR2A	0X04C	比较2A寄存器
TIMx_CCR2B	0X050	比较2B寄存器

表 14-1 Timer 寄存器列表

14.4 Mode 0 Timer Register Description

14.4.1 16-bit Mode Reload Register (TIMx_ARR)

偏移地址：0x000

复位值：0x0000 0000

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

位	符号	描述
31:16	Reserved	保留位
15:0	ARR	16位重载定时器重载值寄存器

14.4.2 16-bit Mode Count Register (TIMx_CNT)

偏移地址：0x004

复位值：0x0000 0000

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

位	符号	描述
31:16	Reserved	保留位
15:0	CNT	16位重载定时器 计数值寄存器

14.4.3 32-bit Mode Count Register (TIMx_CNT32)

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT32[31:16]								RW							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT32[15:0]								RW							

位	符号	描述
31:0	CNT32	32位定时器 计数值寄存器 注：仅在模式0 32位定时器自由计数模式下有效，其他模式禁止写该寄存器

14.4.4 Control Register (TIMx_M0CR)

偏移地址: 0x00C

复位值: 0x0060 0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8								
Reserved		Mode		Reserved		UIE		GATEP		GATE					
		RW				RW		RW		RW					
7	6	5	4	3	2	1	0								
Reserved		PRS		TOGEN		CT		MD		CTEN					
		RW		RW		RW		RW		RW					

位	符号	描述
31:14	Reserved	保留位
13:12	MODE	工作模式 00 定时器模式0; 01 PWC模式 10 锯齿波模式; 11三角波模式
11	Reserved	保留位
10	UIE	中断使能控制, 写1后使能中断
9	GATEP	端口 GATE 极性控制 0: 端口GATE高电平有效 1: 端口GATE低电平有效
8	GATE	定时器门控使能 0: 无门控, CTEN=1 时定时器工作; 1: 门控使能, 端口 GATE 有效并且CTEN=1时定时器才工作;
7	Reserved	保留位
6:4	PRS	内部时钟分频选择 000: 1; 001: 2; 010: 4; 011: 8; 100: 16; 101: 32; 110: 64; 111: 256;
3	TOGEN	模式0下翻转输出使能 1: 翻转输出使能 0: 翻转输出关闭CHA,CHB输出为低电平
2	CT	计数时钟选择 0: 内部计数时钟TCLK 1: 外部计数时钟ETR;
1	MD	模式选择32定时/16定时模式选择 0: 32位自由计数 1: 16位重载计数
0	CTEN	定时器使能

		0: 定时器停止; 1: 定时器使能
--	--	-----------------------

14.4.5 Interrupt Flag Register (TIMx_IFR)

偏移地址: 0x010

复位值: 0x0000 0000

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

位	符号	描述
31:1	Res.	保留位
0	UIF	溢出中断

14.4.6 Interrupt Flag Clear Register (TIMx_ICLR)

偏移地址: 0x014

复位值: 0x0000 FFFF

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

位	符号	描述
31:1	REV	保留位
0	UIF	溢出中断清除, 写0清除

14.4.7 死区时间寄存器 (TIMx_DTR)

偏移地址: 0x030

复位值: 0x0000 0000

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

位	符号	描述
31:13	Reserved	保留位
12	MOE	翻转输出使能 0: 翻转输出为输入状态 1: 翻转端口为输出状态
11:0	Reserved	保留位

14.5 Pulse Width Measurement PWC Register Description

14.5.1 16-bit Mode Count Register (TIMx_CNT)

偏移地址：0x004

复位值：0x0000 0000

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

位	符号	描述
31:16	Reserved	保留位
15:0	CNT	计数值寄存器

14.5.2 Control Register (TIMx_M1CR)

偏移地址: 0x00C

复位值: 0x0060 0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8								
Reserved	Oneshot	Mode		Reserved	UIE	Edg2nd	Edg1st								

15	14	13	12	11	10	9	8
Reserved	Oneshot	Mode		Reserved	UIE	Edg2nd	Edg1st
	RW	RW			RW	RW	RW
7	6	5	4	3	2	1	0
Reserved		PRS		Reserved	CT	Reserved	CTEN
		RW			RW		RW

位	符号	描述												
31:15	Reserved	保留位												
14	Oneshot	单次触发模式选择 1: 完成一次脉冲测量自动结束, 再次测量需要重新使能CTEN 0: 循环测量												
13:12	MODE	工作模式 00 定时器模式0; 01 PWC模式 10 锯齿波模式; 11三角波模式												
11	Reserved	保留位												
10	UIE	中断使能控制, 写1后使能中断 计数到0xFFFF才会溢出并产生溢出标志												
9	Edg2nd	脉冲宽度测量结束边沿选择												
8	Edg1st	脉冲宽度测量开始边沿选择 <table border="1" style="margin-left: auto; margin-right: auto;"><tr> <td>edg2nd</td><td>Edg1st</td><td>00</td><td>01</td><td>10</td><td>11</td></tr> <tr> <td>测量</td><td></td><td>上沿-上沿周期</td><td>低电平宽度</td><td>高电平宽度</td><td>下沿-下沿周期</td></tr> </table>	edg2nd	Edg1st	00	01	10	11	测量		上沿-上沿周期	低电平宽度	高电平宽度	下沿-下沿周期
edg2nd	Edg1st	00	01	10	11									
测量		上沿-上沿周期	低电平宽度	高电平宽度	下沿-下沿周期									
7	Reserved	保留位												
6:4	PRS	内部时钟分频选择 000: 1; 001: 2; 010: 4; 011: 8; 100: 16; 101: 32; 110: 64; 111: 256;												
3	Reserved	保留位												
2	CT	计数时钟选择 0: 内部计数时钟TCLK 1: 外部计数时钟ETR;												
1	Reserved	保留位												
0	CTEN	脉冲宽度测量使能 0: 禁止; 1: 使能												

14.5.3 Interrupt Flag Register (TIMx_IFR)

偏移地址: 0x010

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													CA0F RO	Res. Res.	UIF RO

位	符号	描述
31:3	Res	保留位
2	CA0F	脉冲宽度测量中断标志
1	Res.	保留
0	UIF	溢出中断标志

14.5.4 Interrupt Flag Clear Register (TIMx_ICLR)

偏移地址: 0x014

复位值: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													CA0F R1W0	Res. Res.	UIF R1W0

位	符号	描述
31:3	Res.	保留位
2	CA0F	脉冲宽度测量中断标志清除, 写0清除
1	Res.	保留
0	UIF	溢出中断清除, 写0清除

14.5.5 Master-Slave Mode Control Register (TIMx_MSCR)

偏移地址：0x018

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		IB0S IA0S		Reserved		TS		RW Reserved							

位	符号	描述
31:13	Res.	保留位
12	IB0S	CH0B 输入选择 0: CH0B; 1: 内部触发TS选择信号; 注: 当PWM互补输出时自动选择GATE端口作为CH0B的输入
11	IA0S	IA0 输入选择 0: CH0A; 1: CH0A CH1A CH2A XOR(TIM3) 0: CH0A; 1: CH0A ETR GATE XOR(TIM0) 注: 设置为1后端口的任意一个端口变化将导致输入变化
10:8	Res.	保留位
7:5	TS	触发选择 000: 端口ETR的滤波相位选择后的信号ETFP ; 001: 内部互联信号 ITR0 010: 内部互联信号 ITR1; 011: 内部互联信号 ITR2; 100: 内部互联信号 ITR3; 101: 端口CH0A的边沿信号; 110: 端口CH0A的滤波相位选择后的信号IAFP 111: 端口CH0B的滤波相位选择后的信号IBFP;
4:0	Res.	保留位

14.5.6 Output Control Filter (TIMx_FLTR)

偏移地址：0x01C

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETP		FLTET													Reserved
RW		RW													RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									FLTBO		Resvered		FLTA0		RW

位	符号	描述
31	ETP	ETR 输入相位选择 0：同相位； 1：反向输入；
30:28	FLTET	ETR滤波控制 滤波设置 0xx: 滤波无效 100: pclk 3个连续有效; 101: pclk/4 3个连续有效 110: pclk/16 3个连续有效; 111: pclk/64 3个连续有效
27:7	Resvered	保留位
6:4	FLTB0	CHB 输入滤波控制； 滤波设置 0xx: 滤波无效 100: pclk 3个连续有效; 101: pclk/4 3个连续有效 110: pclk/16 3个连续有效; 111: pclk/64 3个连续有效
3	Resvered	保留位
2:0	FLTA0	CHA 输入滤波控制 滤波设置 0xx: 滤波无效 100: pclk 3个连续有效; 101: pclk/4 3个连续有效 110: pclk/16 3个连续有效; 111: pclk/64 3个连续有效

14.5.7 Control Register (TIMx_CRO)

偏移地址：0x024；

复位值：0x0000 3000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CIEA	Reserved	CSB	CSA	Reserved			
RW								RW		RW		Reserved			

位	符号	描述
31:9	Resvered	保留位
8	CIEA	CIEA 脉冲宽度测量完成中断使能 0: 禁止 1: 使能
7:6	Resvered	保留位
5	CSB	B通道捕获/比较功能选择，当脉冲宽度测量使用通道B滤波时，需要设置CSB为1 0: 比较模式 1: 捕获模式
4	CSA	A通道捕获/比较功能选择，当脉冲宽度测量使用通道A滤波时，需要设置CSA为1 0: 比较模式 1: 捕获模式
3:0	Resvered	保留位

14.5.8 Compare Capture Register (TIMx_CCR0A)

偏移地址：0x03C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR0A															
R															

位	符号	描述
31:16	Reserved	保留位
15:0	CCR0A	脉冲宽度测量结果

14.6 Mode 2, 3 Register Description

14.6.1 16-bit Mode Reload Register (TIMx_ARR)

偏移地址：0x000(0X038 dummy address)

复位值：0x0000 0000

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

位	符号	描述
31:16	Reserved	保留位
15:0	ARR	重载寄存器/周期寄存器，具有缓存功能 在计数器未使能或者缓存没有使能时，缓存寄存器可以立刻更新

14.6.2 16-bit Mode Count Register (TIMx_CNT)

偏移地址：0x004

复位值：0x0000 0000

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

位	符号	描述
31:16	Reserved	保留位
15:0	CNT	重载定时器 计数值寄存器

注：使用 PWM 比较输出时，初始化 CNT 值需要小于 ARR 的值。

14.6.3 Control Register (TIMx_M23CR)

偏移地址: 0x00C

复位值: 0x0060 0008

31	30	29	28	27	26	25	24
Reserved				DIR	BG	UG	TG
				RW/RO	W1	W1	W1
23	22	21	20	19	18	17	16
OCCE		CIS	BIE	TIE	TDE	URS	OCCS
RW		RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8
CSG	Oneshot	Mode		UDE	UIE	CFG	CRG
RW	RW	RW		RW	RW	RW	RW
7	6	5	4	3	2	1	0
BUFPEN		PRS		Pwm2s	CT	Comp	CTEN
RW		RW		RW	RW	RW	RW

位	符号	描述
31:28	Reserved	保留位
27	DIR	计数方向, 只有在锯齿波模式下可以写。其他模式下只读, 写无效 0: 上计数 1: 下计数 注: 从其他模式切换到中心对齐模式DIR自动清0。软件事件更新和从模式外部触发复位模式DIR自动清零。
26	BG	软件刹车, 自动清零 写1产生软件刹车; 写0无效
25	UG	软件更新, 自动清零 写1产生软件更新; 写0无效 初始化计数器并更新缓存寄存器到相应寄存器(缓存使能), 预除频计数器也会被清零。
24	TG	软件触发, 自动清零, 需要在触发模式SMS=2且mode=2/3下都可以触发。 写1产生软件触发; 写0无效
23	OCCE	OCREF 清除使能 1: OCREF_CLR信号可以清除OCREF输出 0: OCREF输出不受OCREF_CLR影响
22:21	CIS	中心对齐A比较中断模式(B比较中断单独控制在CRx寄存器中CISB) 00: 无中断, 01: 上沿中断, 10: 下沿中断, 11: 上下沿都中断

20	BIE	刹车中断使能 1: 中断使能 0: 中断禁止
19	TIE	触发中断使能 1: 中断使能 0: 中断禁止
18	TDE	触发DMA使能 1: 中断使能 0: 中断禁止
17	URS	更新源 0: 上溢出/下溢出/软件更新UG/从模式复位; 1: 上溢出/下溢出
16	OCCS	OCREF 清除源选择 0: 电压比较器VC输出, VC选择在VCx_OUTCFG寄存器设置 1: ETR端口滤波相位选择后的信号 当OCCE有效时, OC_clr可以清除OCREF的比较输出信号为零, (当OCMx>1时有效), 下一个uev事件后继续比较输出
15	CSG	GATE 在PWM互补模式下捕获/比较选择; (只有在PWM互补输出时有效) 使用CCR0B作为GATE的比较或捕获通道 1: 捕获; 0: 比较
14	Oneshot	单次触发模式选择 1: 发生事件更新后定时器停止。 0: 循环计数
13:12	MODE	工作模式 00 定时器模式0; 01 PWC模式 10 锯齿波模式; 11三角波模式
11	UDE	更新DMA使能 1: 使能更新触发DMA 0: 禁止更新触发DMA
10	UIE	UIE 更新中断使能 1: 使能更新中断 0: 禁止更新中断
9	CFG	GATE作为捕获输入时, 下沿捕获有效控制 (只有在PWM互补输出时有效) 1: 下沿捕获有效 0: 下沿捕获无效
8	CRG	GATE作为捕获输入时, 上沿捕获有效控制 (只有在PWM互补输出时有效) 1: 上沿捕获有效 0: 上沿捕获无效

7	BUFPEN	重载缓存使能 1: 周期缓存使能, 写入后下个事件更新后才影响到周期值。 0: 周期缓存无效, 写入后立刻影响周期值
6:4	PRS	内部时钟分频选择 000: 1; 001: 2; 010: 4; 011: 8; 100: 16; 101: 32; 110: 64; 111: 256;
3	PWM2S	OCREFA双点比较选择(缺省值为1) 0: 双点比较使能, 使用CCRA,CCRB比较控制OCREFA输出 1: 单点比较使能, 只使用CCRA比较控制OCREFA输出 注: OCREFB不受影响, 仍然使用CCRB控制OCREFB输出
2	CT	计数时钟选择 0: 内部计数时钟TCLK 1: 外部计数时钟ETR;
1	Comp	PWM互补输出模式选择 0: 独立PWM输出 1: 互补PWM输出
0	CTEN	定时器使能 0: 禁止; 1: 使能 可以外部触发使能, 在oneshot模式下结束时该位自动清零

14.6.4 Interrupt Flag Register (TIMx_IFR)

偏移地址：0x010

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIF	BIF	CB2E	CB1E	CB0E	CA2E	CA1A	CA0E	CB2F	CB1F	CB0F	CA2F	CA1F	CA0F	Res.	UIF
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位	符号	描述
31:16	Reserved	保留位
15	TIF	触发中断标志
14	BIF	刹车中断标志
13	CB2E	通道CH2B捕获数据丢失标志 0: 无数据丢失; 1: 数据丢失 (仅TIM3存在)
12	CB1E	通道CH1B捕获数据丢失标志 0: 无数据丢失; 1: 数据丢失 (仅TIM3存在)
11	CB0E	通道CH0B捕获数据丢失标志 0: 无数据丢失; 1: 数据丢失
10	CA2E	通道CH2A捕获数据丢失标志 0: 无数据丢失; 1: 数据丢失 (仅TIM3存在)
9	CA1E	通道CH1A捕获数据丢失标志 0: 无数据丢失; 1: 数据丢失 (仅TIM3存在)
8	CA0E	通道CH0A捕获数据丢失标志 0: 无数据丢失; 1: 数据丢失
7	CB2F	通道CH2B发生捕获/比较匹配标志 0: 没有发生 1: 发生 (仅TIM3存在)
6	CB1F	通道CH1B发生捕获/比较匹配标志 0: 没有发生 1: 发生 (仅TIM3存在)
5	CB0F	通道CH0B发生捕获/比较匹配标志 0: 没有发生 1: 发生
4	CA2F	通道CH2A发生捕获/比较匹配标志 0: 没有发生 1: 发生 (仅TIM3存在)
3	CA1F	通道CH1A发生捕获/比较匹配标志 0: 没有发生 1: 发生 (仅TIM3存在)
2	CA0F	通道CH0A发生捕获/比较匹配标志 0: 没有发生 1: 发生
1	Res.	保留位

0	UIF	事件更新中断标志 0: 没有发生 1: 发生
---	-----	---------------------------

14.6.5 Interrupt Flag Clear Register (TIMx_ICLR)

偏移地址：0x014

复位值：0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIF	BIF	CB2E	CB1E	CB0E	CA2E	CA1E	CA0E	CB2F	CB1F	CB0F	CA2F	CA1F	CA0F	Res.	UIF
R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	R1W 0

位	符号	描述
31:16	REV	保留位
15	TIF	触发中断标志清除，写0清除
14	BIF	刹车中断标志清除，写0清除
13	CB2E	通道CH2B捕获数据丢失标志清除，写0清除（仅TIM3存在）
12	CB1E	通道CH1B捕获数据丢失标志清除，写0清除（仅TIM3存在）
11	CB0E	通道CH0B捕获数据丢失标志清除，写0清除
10	CA2E	通道CH2A捕获数据丢失标志清除，写0清除（仅TIM3存在）
9	CA1E	通道CH1A捕获数据丢失标志清除，写0清除（仅TIM3存在）
8	CA0E	通道CH0A捕获数据丢失标志清除，写0清除
7	CB2F	通道CH2B捕获/比较匹配标志清除，写0清除（仅TIM3存在）
6	CB1F	通道CH1B捕获/比较匹配标志清除，写0清除（仅TIM3存在）
5	CB0F	通道CH0B捕获/比较匹配标志清除，写0清除
4	CA2F	通道CH2A捕获/比较匹配标志清除，写0清除（仅TIM3存在）
3	CA1F	通道CH1A捕获/比较匹配标志清除，写0清除（仅TIM3存在）
2	CA0F	通道CH0A捕获/比较匹配标志清除，写0清除
1	Res.	保留
0	UIF	事件更新中断清除，写0清除

14.6.6 Master-Slave Mode Control Register (TIMx_MSCR)

偏移地址：0x018

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		IBOS	IAOS	SMS	TS	MSM	CCDS	MMS							
RW	RW	RW	RW	RW	RW	RW	RW	RW							

位	符号	描述
31:13	Res	保留位
12	IBOS	CH0B 输入选择 0: CH0B; 1: 内部触发TS选择信号; 注: 当PWM互补输出时自动选择GATE端口作为CH0B的输入
11	IAOS	IA0 输入选择 0: CH0A; 1: CH0A CH1A CH2A XOR(TIM3) 0: CH0A; 1: CH0A ETR GATE XOR(TIM0) 注: 设置为1后端口的任意一个端口变化将导致输入变化
10:8	SMS	从模式功能选择 000: 使用内部时钟; 001: 复位功能; 010: 触发模式; 011: 外部时钟模式 100: 正交编码计数模式1; 101: 正交编码计数模式2; 110: 正交编码计数模式3; 111: 门控功能
7:5	TS	触发选择 000: 端口ETR的滤波相位选择后的信号ETFP ; 001: 内部互联信号 ITR0 010: 内部互联信号 ITR1; 011: 内部互联信号 ITR2; 100: 内部互联信号 ITR3; 101: 端口CH0A的边沿信号; 110: 端口CH0A的滤波相位选择后的信号IAFP 111: 端口CH0B的滤波相位选择后的信号IBFP;
4	MSM	主从选择 0: 无延时 1: 延时使能, 使主送计数器同时启动. 注: 使用触发模式时, 从模式设置为0, 主模式设置为1, 可以使主从计数同时启动

3	CCDS	比较模式下 DMA 比较触发选择; 0 : 比较匹配触发DMA; 1: 比较匹配不触发DMA, 事件更新代替比较匹配触发DMA																		
2:0	MMS	<p>主模式输出选择, 用于内部互联, 连接到其他定时器的ITRx</p> <table border="1"> <tr> <td>定时器0/1/2</td> <td>定时器3</td> </tr> <tr> <td>000: 软件更新UG, 写CR.UG</td> <td>000: 软件更新UG, 写CR.UG</td> </tr> <tr> <td>001: 定时器使能CTEN</td> <td>001: 定时器使能CTEN</td> </tr> <tr> <td>010: 定时器事件更新UEV;</td> <td>010: 定时器事件更新UEV;</td> </tr> <tr> <td>011: 比较匹配选择输出CMPSO;</td> <td>011: 比较匹配选择输出CMPSO;</td> </tr> <tr> <td>100: 定时器比较参数输出OCREF0A</td> <td>100: 定时器比较参数输出OCREF0A</td> </tr> <tr> <td>101: 定时器比较参数输出OCREF0B</td> <td>101: 定时器比较参数输出OCREF1A</td> </tr> <tr> <td>110: 定时器比较参数输出OCREF0B</td> <td>110: 定时器比较参数输出OCREF2A</td> </tr> <tr> <td>111: 定时器比较参数输出OCREF0B</td> <td>111: 定时器比较参数输出OCREF0B</td> </tr> </table>	定时器0/1/2	定时器3	000: 软件更新UG, 写CR.UG	000: 软件更新UG, 写CR.UG	001: 定时器使能CTEN	001: 定时器使能CTEN	010: 定时器事件更新UEV;	010: 定时器事件更新UEV;	011: 比较匹配选择输出CMPSO;	011: 比较匹配选择输出CMPSO;	100: 定时器比较参数输出OCREF0A	100: 定时器比较参数输出OCREF0A	101: 定时器比较参数输出OCREF0B	101: 定时器比较参数输出OCREF1A	110: 定时器比较参数输出OCREF0B	110: 定时器比较参数输出OCREF2A	111: 定时器比较参数输出OCREF0B	111: 定时器比较参数输出OCREF0B
定时器0/1/2	定时器3																			
000: 软件更新UG, 写CR.UG	000: 软件更新UG, 写CR.UG																			
001: 定时器使能CTEN	001: 定时器使能CTEN																			
010: 定时器事件更新UEV;	010: 定时器事件更新UEV;																			
011: 比较匹配选择输出CMPSO;	011: 比较匹配选择输出CMPSO;																			
100: 定时器比较参数输出OCREF0A	100: 定时器比较参数输出OCREF0A																			
101: 定时器比较参数输出OCREF0B	101: 定时器比较参数输出OCREF1A																			
110: 定时器比较参数输出OCREF0B	110: 定时器比较参数输出OCREF2A																			
111: 定时器比较参数输出OCREF0B	111: 定时器比较参数输出OCREF0B																			

14.6.7 Output Control/Input Filtering (TIMx_FLTR)

偏移地址: 0x01C

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETP	FLTET	BKP		FLTBK	CCPB2		OCMB2	CCPA2		OCMA2					
-	-	-		-	-		FLTB2	-		FLTA2					
RW	RW	RW		RW	RW		RW	RW		RW					RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCPB1	OCMB1	CCPA1		OCMA1	CCPB0		OCMB0	CCPA0		OCMA0					
-	FLTB1	-		FLTA1	CCPB0		FLTB0	CCPA0		FLTA0					
RW	RW	RW		RW	RW		RW	RW		RW					RW

位	符号	描述
31	ETP	ETR 输入相位选择 0: 同相位; 1: 反向输入;
30:28	FLTET	ETR滤波控制 滤波设置 0xx: 滤波无效 100: pclk 3个连续有效; 101: pclk/4 3个连续有效 110: pclk/16 3个连续有效; 111: pclk/64 3个连续有效
27	BKP	刹车BK输入相位选择 0: 同相位; 1: 反向输入;
26:24	FLTBK	刹车输入滤波控制 滤波设置 0xx: 滤波无效 100: pclk 3个连续有效; 101: pclk/4 3个连续有效 110: pclk/16 3个连续有效; 111: pclk/64 3个连续有效 注: 为了保证OCMB0的PWM输出设置, GATE 在PWM互补模式下作为捕获输入, 滤波设置无效。
23	CCPB2	比较功能: CH2B通道比较输出相位控制 0: 正常输出; 1: 反向输出
22:20	OCMB2 FLTB2	比较功能: CH2B通道比较控制;参考OCMB0 捕获功能: CH2B输入通道滤波设置, 参考FLTBK
19	CCPA2	比较功能: CH2A通道比较输出相位控制

		0：正常输出; 1：反向输出
18:16	OCMA2 FLTA2	比较功能：CH2A通道比较控制;参考OCMB0 捕获功能：CH2A输入通道滤波设置，参考FLTBK
15	CCPB1	比较功能：CH1B通道比较输出相位控制 0：正常输出; 1：反向输出
14: 12	OCMB1 FLTB1	比较功能：CH1B通道比较控制;参考OCMB0 捕获功能：CH1B输入通道滤波设置，参考FLTBK
11	CCPA1	比较功能：CH1A通道比较输出相位控制 0：正常输出; 1：反向输出
10:8	OCMA1 FLTA1	比较功能：CH1A通道比较控制;参考OCMB0 捕获功能：CH1A输入通道滤波设置，参考FLTBK
7	CCPB0	比较功能：输出比较模式 CCPBx比较输出CHBx端口极性控制 0：正常输出; 1：反向输出 编码计数与从模式门控功能：输入相位控制 CCPB0从模式门控，复位，外部触发，外部时钟使用CH0B端口输入极性控制 0：正常输入; 1：反向输入
6:4	OCMB0 FLTB0	比较功能：CH0B通道比较控制 000：强制为0 001：强制为1 010：比较匹配时强制为0 011：比较匹配时强制为1 100：比较匹配时翻转 101：比较匹配时输出一个计数周期的高电平 110：PWM 模式1 单点比较： 上计数时CNT<CCRxy输出高，下计数时CNT>CCRxy输出为低电平 双点比较： 1) 锯齿波上计数 CCRxA<CNT≤CCRxB输出为低电平 2) 锯齿波下计数 CCRxA<CNT≤CCRxB输出为高电平 3) 三角波上计数CNT<CCRxA输出高，下计数CNT>CCRxB为低电平 111 PWM 模式2 单点比较： 上计数时CNT<CCRxy输出低，下计数时CNT>CCRxy输出为高电平 双点比较：

		<p>1) 锯齿波上计数 $CCRxA \leq CNT < CCRxB$ 输出为高电平 2) 锯齿波下计数 $CCRxA \leq CNT < CCRxB$ 输出为低电平 3) 三角波上计数 $CNT < CCRxA$ 输出低, 下计数 $CNT > CCRxB$ 为高电平 捕获功能: CH0B输入通道滤波设置, 参考FLTBK</p>
3	CCPA0	<p>比较功能: CCPAx比较输出CHAx端口极性控制 0: 正常输出; 1: 反向输出 编码计数与从模式门控功能: 输入相位控制 CCPA0从模式门控, 复位, 外部触发, 外部时钟使用CH0A端口输入极性控制 0: 正常输入; 1: 反向输入</p>
2:0	OCMA0 FLTA0	<p>比较功能: A通道比较控制; 参考OCMB0 捕获功能: CH0A输入通道滤波设置, 参考FLTBK</p>

14.6.8 ADC Trigger Control Register (TIMx_ADTR)

偏移地址: 0x020

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ADTE	CMB2E	CMB1E	CMB0E	CMA2E	CMA1E	CMA0E	UEVE
								RW	RW	RW	RW	RW	RW	RW	RW

位	符号	描述
31:8	Resvered	保留位
7	ADTE	使能ADC触发全局控制 1: 使能 0: 禁止
6	CMB2E	通道2B比较匹配触发ADC使能, 只有TIM3存在 1: 使能 0: 禁止
5	CMB1E	通道1B比较匹配触发ADC使能, 只有TIM3存在 1: 使能 0: 禁止
4	CMB0E	通道0B比较匹配触发ADC使能 1: 使能 0: 禁止
3	CMA2E	通道2A比较匹配触发ADC使能, 只有TIM3存在 1: 使能 0: 禁止
2	CMA1E	通道1A比较匹配触发ADC使能, 只有TIM3存在 1: 使能 0: 禁止
1	CMA0E	通道0A比较匹配触发ADC使能 1: 使能 0: 禁止
0	UEVE	事件更新触发ADC使能 1: 使能 0: 禁止

14.6.9 Channel 0 Control Register (TIMx_CRCH0)

偏移地址: 0x024

复位值: 0x0000 3000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCGB	CCGA	CISB	CDEB	CDEA	CIEB	CIEA	-	-	CSB	CSA	CFB	CRB	CFA	CRA	
CCGB	CCGA	CISB	CDEB	CDEA	CIEB	CIEA	BUFEB	BUFEA	CSB	CSA	bksb	bksa			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			RW

位	符号	描述
31:16	Resvered	保留位
15	CCGB	捕获比较B软件触发，硬件自动清零。 在比较模式下只产生中断； 在捕获模式下产生中断并且捕获计数器的值到捕获寄存器中。
14	CCGA	捕获比较A软件触发，硬件自动清零。 在比较模式下只产生中断； 在捕获模式下产生中断并且捕获计数器的值到捕获寄存器中。
13:12	CISB	B通道比较匹配设置 00 无匹配；01上升匹配；10 下降匹配；11双匹配
11	CDEB	B捕获比较触发DMA使能 0: 禁止 1: 使能
10	CDEA	A捕获比较触发DMA使能 0: 禁止 1: 使能
9	CIEB	B捕获比较触发中断使能 0: 禁止 1: 使能
8	CIEA	A捕获比较触发中断使能 0: 禁止 1: 使能
7	BUFEB	比较功能：B比较缓存使能控制 0: 禁止 1: 使能
6	BUFEA	比较功能：A比较缓存使能控制 0: 禁止 1: 使能
5	CSB	B通道捕获/比较功能选择 0: 比较模式 1: 捕获模式

4	CSA	A通道捕获/比较功能选择 0：比较模式 1：捕获模式
3	CFB	B通道下降沿捕获使能 0：禁止 1：使能
2	CRB	B通道上升沿捕获使能 0：禁止 1：使能
3:2	BKSB	B通道比较功能输出刹车电平控制 00：高阻输出 01：对输出无影响 10：强制输出低电平 11：强制输出高电平；
1	CFA	A通道下降沿捕获使能 0：禁止 1：使能
0	CRA	A通道上升沿捕获使能 0：禁止 1：使能
1:0	BKSA	A通道比较功能输出刹车电平控制 00：高阻输出 01：对输出无影响 10：强制输出低电平 11：强制输出高电平；

14.6.10 Channel 1/2 Control Register (TIM3_CRCH1/2) (only exists in TIM3)

偏移地址：

TIM3_CRCH1: 0x028;

TIM3_CRCH2: 0x02C

复位值： 0x0000 3000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCGB	CCGA	CISB	CDEB	CDEA	CIEB	CIEA	-	-	CSB	CSA	CFB	CRB	CFA	CRA	
CCGB	CCGA	CISB	CDEB	CDEA	CIEB	CIEA	BUFEB	BUFEA	CSB	CSA	bksb		bksa		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

位	符号	描述
31:16	Resvered	保留位
15	CCGB	捕获比较B软件触发，硬件自动清零。 在比较模式下只产生中断； 在捕获模式下产生中断并且捕获计数器的值到捕获寄存器中。
14	CCGA	捕获比较A软件触发，硬件自动清零。 在比较模式下只产生中断； 在捕获模式下产生中断并且捕获计数器的值到捕获寄存器中。
13:12	CISB	B通道比较匹配设置 00 无匹配；01上升匹配；10 下降匹配；11双匹配
11	CDEB	B捕获比较触发DMA使能 0: 禁止 1: 使能
10	CDEA	A捕获比较触发DMA使能 0: 禁止 1: 使能
9	CIEB	B捕获比较触发中断使能 0: 禁止 1: 使能
8	CIEA	A捕获比较触发中断使能 0: 禁止 1: 使能
7	BUFEB	比较功能: B比较缓存使能控制 0: 禁止 1: 使能
6	BUFEA	比较功能: A比较缓存使能控制 0: 禁止 1: 使能

5	CSB	B通道捕获/比较功能选择 0：比较模式 1：捕获模式
4	CSA	A通道捕获/比较功能选择 0：比较模式 1：捕获模式
3	CFB	B通道下降沿捕获使能 0：禁止 1：使能
2	CRB	B通道上升沿捕获使能 0：禁止 1：使能
3:2	BKSB	B通道比较功能输出刹车电平控制 00：高阻输出 01：保持之前输出 10：强制输出低电平 11：强制输出高电平；
1	CFA	A通道下降沿捕获使能 0：禁止 1：使能
0	CRA	A通道上升沿捕获使能 0：禁止 1：使能
1:0	BKSA	A通道比较功能输出刹车电平控制 00：高阻输出 01：保持之前输出 10：强制输出低电平 11：强制输出高电平；

14.6.11 Dead Time Register (TIMx_DTR)

偏移地址: 0x030

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	VCE	Safeen	MOE	AOE	BKE	DTEN	Bksel				DTR				RW

位	符号	描述
31:15	Reserved	保留位
14	VCE	VC Brake enable 0: 禁止 1: 使能
13	Safeen	Safety 刹车使能 (osc fail,brown down,lockup) 0: 禁止 1: 使能
12	MOE	PWM输出使能 0: 禁止 1: 使能
11	AOE	PWM输出自动使能 0: 禁止 1: 使能
10	BKE	刹车使能 0: 禁止 1: 使能
9	DTEN	死区控制使能 0: 禁止 1: 使能
8	bksel	刹车选择 0: 使用自己的刹车控制; 1: TIM1/2使用 TIM0的刹车控制 注: TIM0/TIM3选择无效。
7:0	DTR	死区时间寄存器 DTR[7] =0 T=DTR[6:0]+2 2-129 step=1 DTR[7:6] =10 T={DTR[5:0]+64}*2 +2 130-256 step=2 DTR[7:5] =110 T={DTR[4:0]+32}*8 +2 258-506 step=8 DTR[7:5] =111 T={DTR[4:0]+32}*16 +2 514-1010 step=16

14.6.12 Repeat period setting value (TIMx_RCR)

偏移地址：0x034

复位值：0x0000 0000

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

位	符号	描述
31:8	Reserved	保留位
7:0	RCR	重复周期计数值 设置RCR+1个周期个上溢出下溢出后产生事件更新，计数器上溢出或下溢出时内部RCR_CNT减1，当计数到零后RCR_CNT重载RCR的值，并且产生事件更新UEV信号

14.6.13 Channel 0 Compare Capture Register (TIMx_CCR0A/B)

偏移地址：

TIMx_CCR0A : 0x03C;

TIMx_CCR0B : 0x040

复位值：0x0000 0000

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

位	符号	描述
31:16	Reserved	保留位
15:0	CCROy	比较捕获寄存器，比较具有缓存功能(y=A,B)

14.6.14 Channel 1/2 Compare Capture Register (TIM3_CCR1/2 A/B) (only available for TIM3)

偏移地址：

TIM3_CCR1A: 0x044

TIM3_CCR1B: 0x048

TIM3_CCR2A: 0x04C

TIM3_CCR2B: 0x050

复位值： 0x0000 0000

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

位	符号	描述
31:16	Reserved	保留位
15:0	CCRxy	比较捕获寄存器，比较具有缓存功能(x=1,2; y=A,B)

15 Low Power Timer (LPTIM)

15.1 LPTimer Introduction

LPTimer 是异步 16 位定时/计数器，在系统时钟关闭后仍然可以通过内部低速 RC 或者外部低速晶体振荡计时/计数。通过中断在低功耗模式下唤醒系统。

LPTimer 支持最大 256 的预分频，系统内有两个低功耗定时器，可以通过内部进行级联。

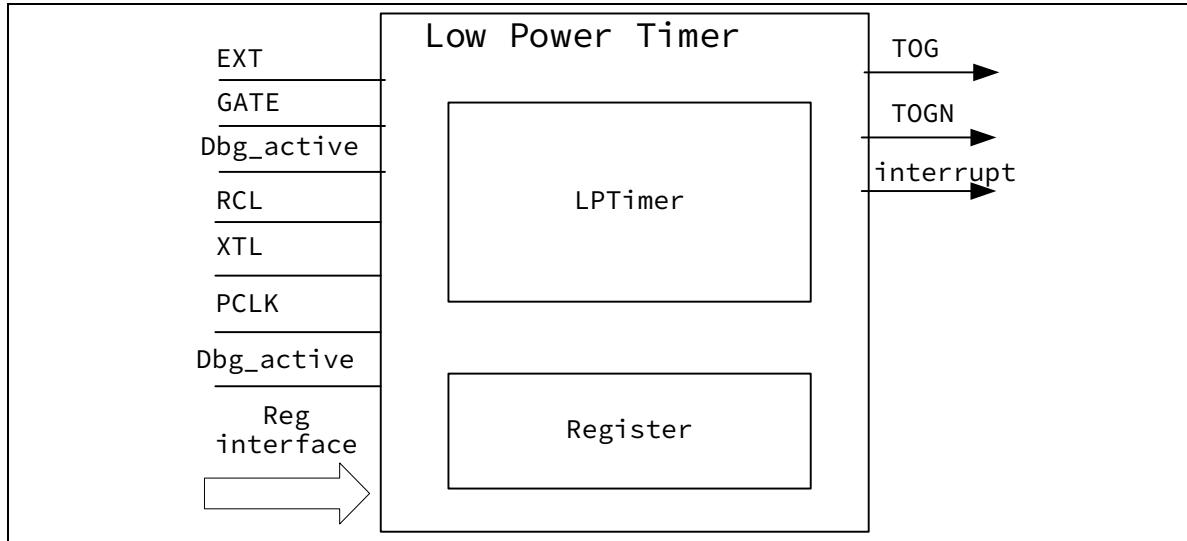


图 15-1 LPTimer 框图

15.2 LPTimer Function Description

LPTimer 支持 2 种工作模式，每个定时器/计数器都有独立的控制启动信号，以及外部输入时钟，门控信号。

LPTimer 使用 EXT, GATE 来进行计数功能，EXT 用于计数器的外部输入时钟信号，Gate 用于有效电平计数使能信号。

LPTimer 的定时器支持两种工作模式，通过设置定时器控制寄存器 (CR) 中 MD 选择工作模式。模式 1 为 16 位自由计数模式。模式 2 是 16 位重载模式。

LPTimer 启动时会自动装载重载寄存器 ARR 的值到计数器中。

LPTimer 可选三种时钟作为定时器时钟，通过控制寄存器 CR.TCK_SEL 选择。默认选择 PCLK。时钟选择如表：

TCK_SEL	00	01	10	11
定时器时钟	PCLK	PCLK	XTL	RCL
读定时器计数值	读经过同步	无同步	读经过同步	读经过同步

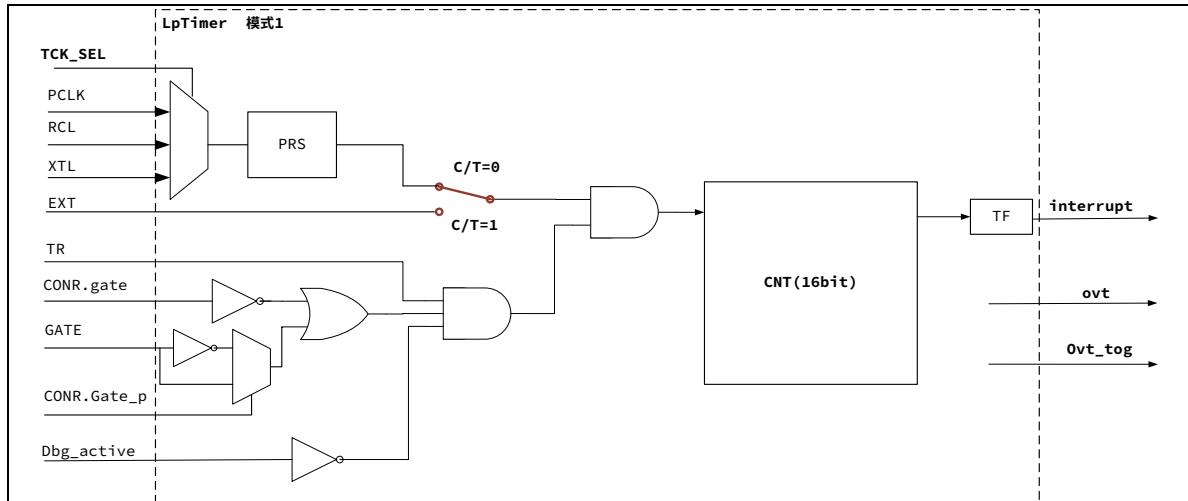


图 15-2 LPTimer 模式 1

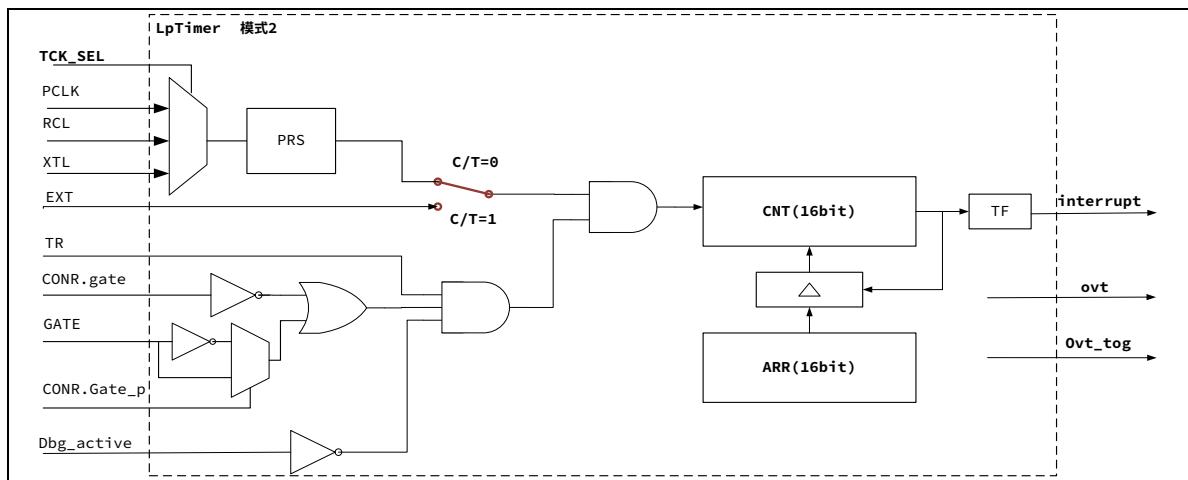


图 15-3 LPTimer 模式 2

当设置对应定时器 TR 为 1 后定时器开始运行。计数器从设定的值开始计数，计数到最大 0xFFFF 后产生溢出中断。模式 1 产生中断后计数器清零继续向上计数，模式 2 产生中断后重载寄存器 ARR 的值到计数器中，向上计数。计数器 CNT 的初值在启动定时器时由 ARR 自动装载。

15.2.1 Counting function

计数功能用于测定某个事件发生的次数。在计数功能中，计数器在每个相应的输入时钟的下降沿累加一次。输入信号被内部的计数时钟采样，因此外部输入时钟频率不能超过系统的计数时钟。计数到最大值会溢出并且产生中断。

15.2.2 Timing function

定时功能用于产生间隔定时。在定时功能中，定时器一个时钟累加一次，计数到最大值会溢出并且产生中断。

定时时钟周期数为(0xFFFF-ARR+0X0001)

15.3 LPTimer Interconnection

15.3.1 GATE Interconnection

GATE 输入可以从端口通过 PX_SEL 选择直接输入，也可通过端口功能寄存器 GPIO_TIMGS 选择可以连通到其他模块或端口。

当 LPTIM_Gx=0x0 时，门控 GATE 输入为 PX_SEL 选择的端口输入，当 LPTIM_Gx=0x1~0x7 时，连接其他模块的输入或输出。

	LPTIM0_G		LPTIM1_G
000	PX_SEL	000	PX_SEL
001	LPUART0_RXD	001	UART0_RXD
010	LPUART1_RXD	010	LPUART0_RXD
011	VCO_OUT	011	VC1_OUT
100	VC1_OUT	100	VC2_OUT
101	PB03	101	PD05
110	PB05	110	PE07
111	PC00	111	PC06

15.3.2 EXT Interconnection

EXT 输入可以从端口直接输入，也可通过端口功能寄存器 GPIO_TIMES 选择可以连通到其他模块或端口。

当 LPTIM_Ex=0x0 时，外部时钟 EXT 输入为 PX_SEL 选择的端口输入，当 LPTIM_Ex=0x1~0x7 时，连接其他模块的输入或输出。

	LPTIM0_E		LPTIM1_E
000	PX_SEL	000	PX_SEL
001	PCNT_S0	001	UART1_RXD
010	LVD_OUT	010	LPUART1_RXD
011	VCO_OUT	011	VCO_OUT
100	VC1_OUT	100	VC2_OUT
101	PB04	101	LPTIM0_TOG
110	PB06	110	PE08
111	PC03	111	PC07

设置 LPTIM1 的定时器的外部输入 LPTIM1_E 为 0x5，选择 LPTIM0 的翻转输出 LPTIM0_TOG 为外部时钟输入，LPTIM1 设置为外部计数模式。这样两个定时器就可以级联使用。

15.3.3 Toggle Output Interconnection

LPTimer 的翻转输出到端口上，可以驱动 Buzzer 实现蜂鸣器的控制。

15.4 LPTimer 寄存器描述

LPTIMER0 基地址 0X40000F00

LPTIMER1 基地址 0X40000F40

表 15-1 LPTimer 寄存器列表

寄存器	偏移地址	描述
LPTIMx_CNT	0X000	LPTimer 计数值只读寄存器
LPTIMx_ARR	0X004	LPTimer 重载寄存器
LPTIMx_CR	0X00C	LPTimer 控制寄存器
LPTIMx_IFR	0X010	LPTimer 中断标志
LPTIMx_ICLR	0X014	LPTimer 中断清除寄存器

15.4.1 Counter Count Value Register (LPTIMx_CNT)

偏移地址: 0x000

复位值: 0x0000 0000

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

位	符号	描述
31:16	Reserved	保留位, 读为0
15:0	CNT	低功耗定时器计数值只读寄存器

15.4.2 Reload Register (LPTIMx_ARR)

偏移地址: 0x004

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW															
位	符号	描述													
31:16	Reserved	保留位, 读为0													
15:0	ARR	低功耗定时器重载寄存器 写 ARR 前需要读取 CR.WT_FLAG, 当且尽当 WT_FLAG 为1时, 写入时才能写入数据。写ARR寄存器后 WT2_FLAG 会变低。 注: 在定时器启动时或者重载时这个值会装载到定时器计数器中。定时器计数器值不可软件更改。													

15.4.3 Control Register (LPTIMx_CR)

偏移地址: 0x00C

复位值: 0x0000 0008

31:11 10 9 8 7 6 4:5 3 2 1 0

Reserved	IE	GATE_P	GATE	WT_FLAG	Reserved	TCK_SEL	TOG_EN	CT	MD	TR
	RW	RW	RW	R		RW	RW	RW	RW	RW

位	符号	描述
31:14	Reserved	保留位, 读为0
13:11	PRS	预除频 (在定时器运行时不可以更改) 000 1 001 2 010 4 011 8 100 16 101 32 110 64 111 256
10	IE	中断使能控制, 写1后使能中断
9	GATE_P	GATE 极性控制, 默认高电平 gate 有效, 设置为1后低电平有效
8	GATE	定时器门控 0: 无门控, TR=1 时定时器工作; 1: 只有门控为1 并且 TR=1 时才工作;
7	WT_FLAG	WT,写同步标志 0: 正在同步, 此时写ARR无效 1: 同步完成, 此时可更改ARR
6	Reserved	保留位, 读为0
5:4	TCK_SEL	LPTimer 时钟选择 00 PCLK; 10:XTL; 11 ,RCL
3	TOG_EN	TOG 输出使能 0: TOG,TOGN 同时输出0 1: TOG,TOGN 输出相位相反的信号。可供 buzzer 使用。
2	CT	计数器/定时器功能选择 0: 定时器功能, 定时器使用 TCK_SEL 选择的时钟进行计数。 1: 计数器功能, 计数器使用外部输入的下降沿进行计数。采样时钟使用 TCK_SEL选择的时钟, 外部输入时钟要低于 1/2 采样时钟。
1	MD	定时器工作模式 0: 模式1无重载模式16位计数器/定时器 1: 模式2自动重装载16位计数器/定时器
0	TR	定时器运行控制位 0: 定时器停止 1: 定时器运行

15.4.4 Interrupt Flag Register (LPTIMx_IFR)

偏移地址: 0x010

复位值: 0x0000 0000

31:8 7 6 5 4 3 2 1 0

Reserved	TF
	RO

位	符号	描述
31:1	Reserved	保留位, 读为0
0	TF	中断标志, 硬件置位。写清除寄存器清零

15.4.5 Interrupt Flag Clear Register (LPTIMx_ICLR)

偏移地址: 0x014

复位值: 0x0000 0001

31:8 7 6 5 4 3 2 1 0

Reserved	TFC
	R1W0

位	符号	描述
31:1	Reserved	保留位, 读为0
0	TFC	中断标志清除, 写0清除, 写1无效

16 Programmable Counter Array (PCA)

16.1 PCA Introduction

PCA(可编程计数器阵列 Programmable Counter Array)支持最多 5 个 16 位的捕获/比较模块。该定时/计数器可用作为一个通用的时钟计数/事件计数器的捕获/比较功能。PCA 的每个模块都可以进行独立编程，以提供输入捕捉，输出比较或脉冲宽度调制。另外模块 4 有额外的看门狗定时器模式。

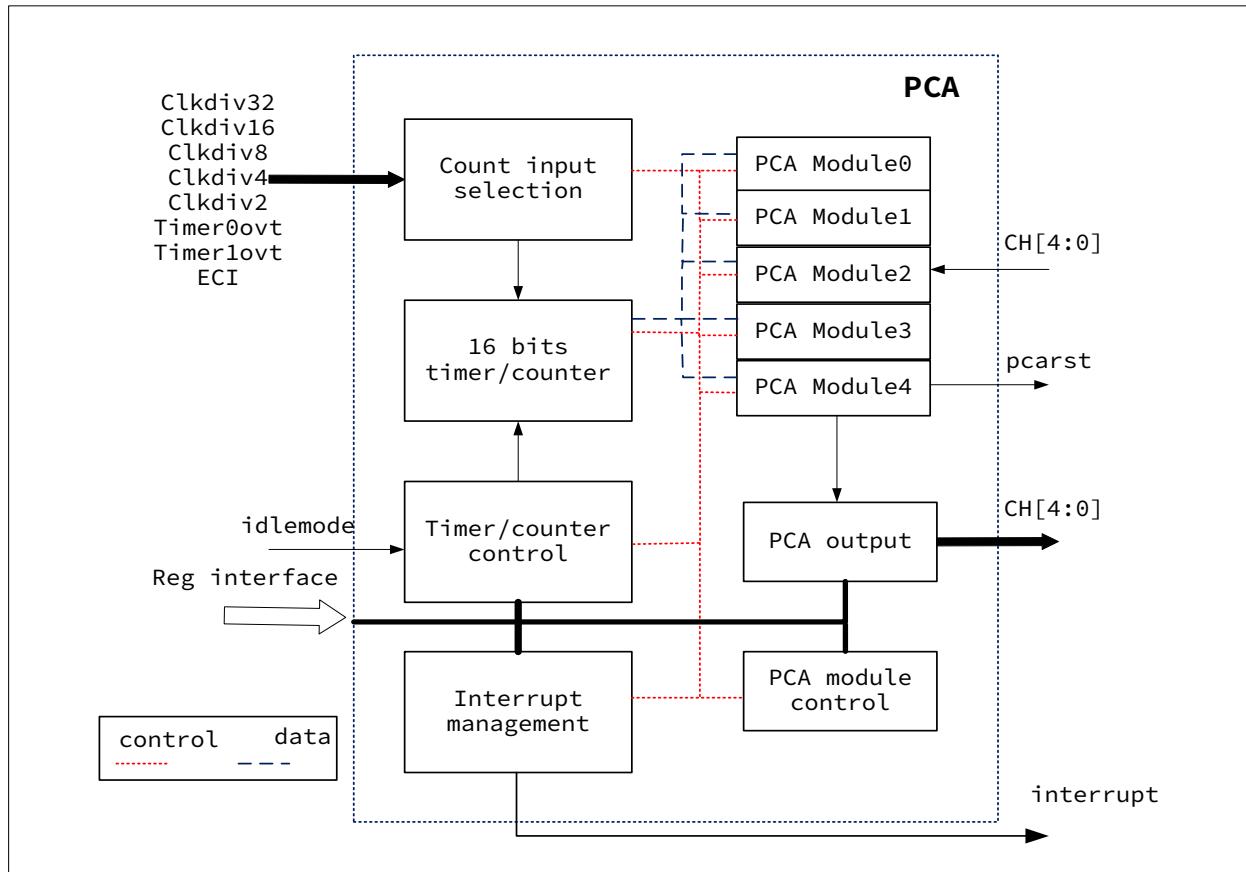


图 16-1 PCA 整体框图

16.2 PCA Functional Description

五个模块都可被配置为独立工作，有三种工作方式：边沿触发捕捉、输出比较、8/16 位脉宽调制。每个模块在系统控制器中都有属于自己的功能寄存器，这些寄存器用于配置模块的工作方式和与模块交换数据。

每组比较/捕获模块是由一个比较/捕获寄存器组(CCAPx)，以及 1 个 16 位比较器，和各种逻辑门控制组成。寄存器组用来存储时间或次数，针对外部触发捕获条件，或内部触发比较条件。在 8 位 PWM 模式下，寄存器(CCAPxL)用来控制输出波形的占空比，CCAPxH 为 8 位比较缓存。16 位 PWM 模式下 CARR 用于控制 PWM 输出的周期，CCAPx 寄存器控制占空比。

每个模块都可以独立编程的操作在任何以下模式：

- 16 位捕获模式的上升沿，下降沿或任意沿触发。
- 比较模式：16 位高速输出，16 位看门狗定时器（模块 4）或 16/8 位脉冲宽度调制。
- 关闭状态。

比较/捕获模块模式寄存器（CCAPMx）确定相应的工作模式。对于比较/捕获模块进行编程时，他们是基于共同的时间计数。定时器/计数器打开和关闭通过 CCON.CR 位即可控制 PCA 定时/计数器的运行。在一个比较/捕获模块捕获，软件定时器，高速输出，设置模块的比较/捕获标志（CCON.CCFx），并产生 PCA 中断请求，如果相应的使能位在 CCAPMx 寄存器设置。CPU 可以在任何时候读写 CCAPx 寄存器。

16.2.1 PCA Timer/Counter

CNT 的这组特殊功能寄存器可用作为一个 16 位定时器/计数器。这是一个 16 位向上计数的计数器。如果 CMOD.CFIE 位被置“1”时，当 CNT 溢出时硬件自动设置 PCA 溢出标志（CCON.CF）并产生 PCA 中断请求。CMOD.CPS[2：0]三位选择八个信号输入到定时器/计数器。

- 系统时钟 PCLK 的 32 分频。
- 系统时钟 PCLK 的 16 分频。
- 系统时钟 PCLK 的 8 分频。
- 系统时钟 PCLK 的 4 分频。
- 系统时钟 PCLK 的 2 分频。
- 定时器 0 的溢出。每次定时器 0 计数溢出后，CNT 就递增，这样提供了 PCA 的可变编程频率输入。
- 定时器 1 的溢出。每次定时器 1 计数溢出后，CNT 就递增，这样提供了 PCA 的可变编程频率输入。
- ECI。CPU 每过 4 个 PCLK 时钟周期就对 PCA ECI 进行采样，当每次采样结果从高变低时，CL 自动加 1，因此最高的 ECI 输入频率不能高于系统时钟 PCLK 的 1/8，以满足采样需求。

设置运行控制器（CCON.CR）启动 PCA 定时/计数器。当 CMOD.CIDL 置“1”后，PCA 定时器/计数器可以继续运行在空闲模式下。CPU 可以随时读取 CNT 的数值，但当计数启动后（CCON.CR=1）时，为了防止计数错误，CNT 是禁止写入的。

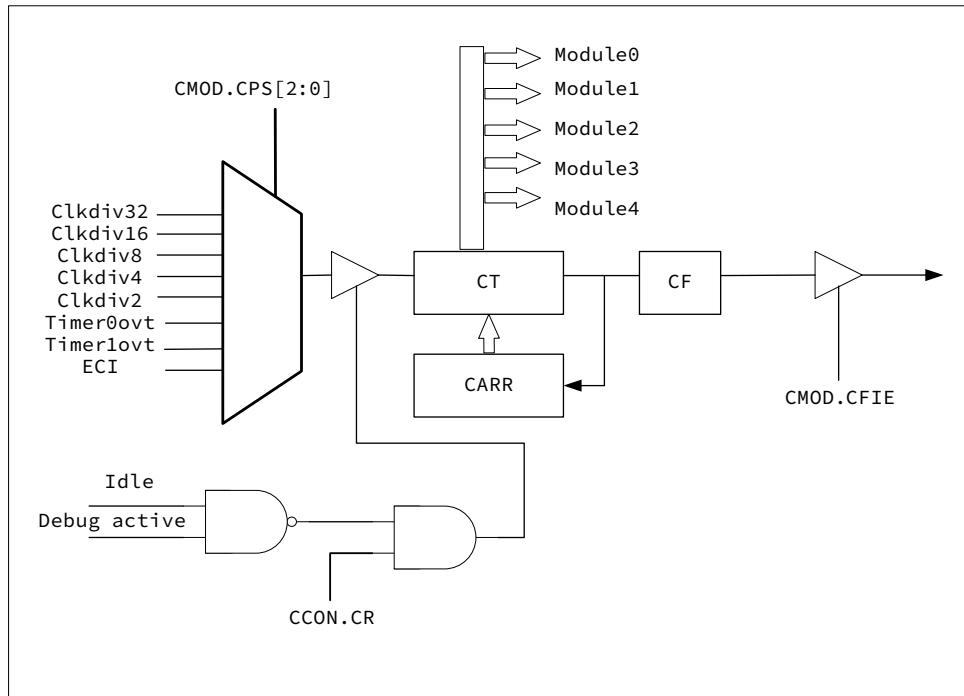


图 16-2 PCA 计数器框图

16.2.1.1 16-bit free counting mode

计数器计数到最大值 0xFFFF 溢出后计数器的值变为 0，重新开始向上计数，如果需要更改计数周期值可以停止 PCA 更改计数器的初值然后继续计数。可用于 PCA 捕获模式，8 位 PWM 模式。

设置流程

1. 保持 PCA_EPWM 的 EPMW 为 0
2. 设置 PCA_CMOD.CPS 选择计数时钟
3. 根据需要设置 PCA_CMOD.CFIE 设置计数溢出中断使能
4. 时钟 PCA_CCON.CR 启动 PCA 计数器
5. 溢出更改计数器初值需要停止 PCA 计数器

16.2.1.2 16-bit reload count mode

计数器计数到与寄存器 CARR 的值相同时溢出后计数器的值变为 0，继续开始向上计数，可用于 PCA 捕获模式，16 位 PWM 模式。

设置流程

1. 设置 PCA_EPWM 的 EPMW 为 1
2. 设置 PCA_CARR 设置计数周期值
3. 设置 PCA_CMOD.CPS 选择计数时钟
4. 根据需要设置 PCA_CMOD.CFIE 设置计数溢出中断使能

5. 时钟 PCA_CCON.CR 启动 PCA 计数器

16.2.2 PCA Capture Function

PCA 捕获模式提供了 5 路 PCA 测量脉冲周期，脉冲宽度，占空比和相位差的功能。

引脚上出现的电平跳变导致 PCA 捕捉 PCA 计数器/定时器的值并将其装入到对应模块的 16 位捕捉/比较寄存器 (CCAPx)。CCAPMx.CAPP 以及 CCAPMx.CAPN 位用于选择触发捕捉的电平变化类型：低电平到高电平（正沿）、高电平到低电平（负沿）或任何变化（正沿或负沿）。当捕捉发生时，CCON 中的捕捉/比较标志 (CCFn) 被置为逻辑‘1’并产生一个中断请求（如果 CCF 中断被允许）。当 CPU 转向中断服务程序时，CCFn 位不能被硬件自动清除，用户软件写 INTCL 寄存器清除此标志位。如果 CCPMx.CAPP 以及 CCAPMx.CAPN 位都被设置为逻辑‘1’，可以通过直接读对应端口引脚的状态来确定本次捕捉是上升沿触发还是由下降沿触发。

分辨率等于定时器/计数器的时钟。输入信号必须在高电平或低电平期间至少保持 2 个时钟周期，以保证输入信号能够被硬件识别。

CPU 可以在任何时候读取或写入 CCAPx 的寄存器。

捕获设置：

- 当需要在外部上升沿进行捕获，CCPMx.CAPP =“1”以及 CCAPMx.CAPN = “0”
- 当需要在外部下降沿进行捕获，CCPMx.CAPP =“0”以及 CCAPMx.CAPN = “1”
- 当需要在外部上升、下降沿进行捕获，CCPMx.CAPP =“1”以及 CCAPMx.CAPN = “1”
- 根据需要配置捕获中断及中断处理程序。
- 按照定时/计数器配置 PCA 计数器启动。

注意：

- 随后由同一模块的捕获值会覆盖现有捕获的值。为了保持捕获的值，在中断服务程序中将它保存在 RAM 里面，这个操作必须在下一次事件出现之前完成，否则就会丢失前面一次捕获采样值。

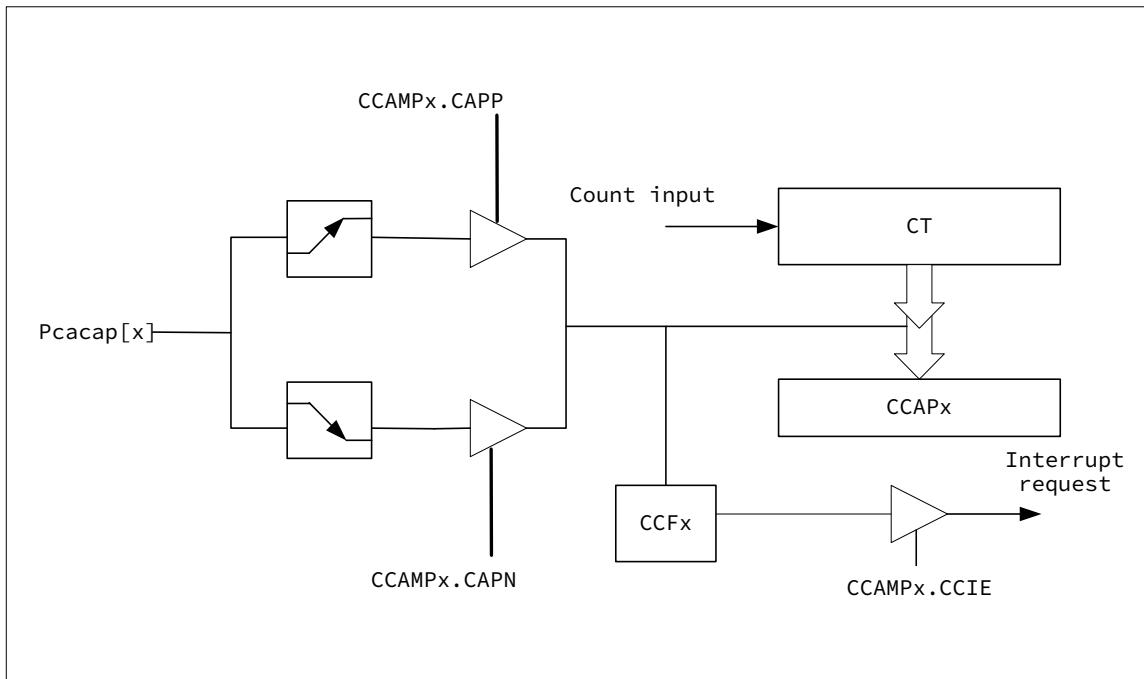


图 16-3 PCA 捕获功能框图

16.2.3 PCA Comparison Function

比较功能提供如下功能，高速输出模式，WDT 模式，16 为 PWM 模式和 8 位 PWM 模式。在前三个功能中，比较/捕获模块比较 16 位 PCA 定时器/计数器的值与预先加载到该模块的 CCAPx 寄存器中的 16 位值。在 8 位 PWM 模式下，PCA 模块不断地将 PCA 定时器/计数器低字节寄存器（CNT）与一个在 CCAPxL 模块寄存器 8 位的值进行比较。每 4 个时钟周期比较一次，即与最快的 PCA 定时器/计数器的时钟速率相匹配。

设置 CCAPMx.ECOM 位选择该模块的比较功能。

若要正确使用在比较模式下的模块，请遵守以下的一般程序：

- 选择 PCA 模块的操作模式。
- 选择 PCA 定时器/计数器的输入信号。
- 比较值加载到模块的比较/捕获寄存器对。
- 设置 PCA 定时器/计数器运行控制位。
- 匹配后产生中断，清除模块的比较/捕获标志。

16.2.3.1 Compare Toggle Output Mode

在比较翻转输出模式，每当 PCA 计数器内的值与模块的 16 位捕捉/比较寄存器（CCAPx）发生匹配时，模块 PCA 的 CH[x]引脚上的逻辑电平将发生变化。这可以提供比切换 IO 输出有更高精度，因为这个输出不会被中断响应而影响输出频率，靠 CPU 来切换 IO 输出的话，功耗，精度都有所欠缺。

要设定一个比较/捕获模块的比较翻转输出模式，设置 CCAPMx.ECOM，CCAPMx.MAT 和 CCAPMx.TOG 位。PCA 定时器/计数器和比较/捕获的寄存器（CCAPx）之间的匹配切换 PCA 的 CH[x]信号，并设置模块的比较/捕获标志（CCON.CCFx）。通过软件设置或清除 PCA 的 CH[x]信号，用户可以选择匹配切换信号从低到高或高到低。

用户也可以选择产生一个中断请求，通过设置相应的中断使能位（CCAPMx.CCIE）当匹配发生时，即可产生中断请求。由于硬件无法清除的比较/捕获标志中断，用户必须在软件中清除这个标志位。如果用户在中断程序中不去改变比较/捕获寄存器，PCA 并重新计数比较值，如相匹配则发生下一次翻转。在中断服务程序中，一个新的 16 位比较值可以被写入比较/捕获的寄存器（CCAPx）。

注意：

- 为了防止无效的匹配，而更新这些寄存器，用户软件应该写 CCAPxL 的首先然后 CCAPxH。写到 CCAPxL 清除禁用比较功能 ECOM 位，而写到 CCAPxH 设置的 ECOM 位，重新启用比较功能。

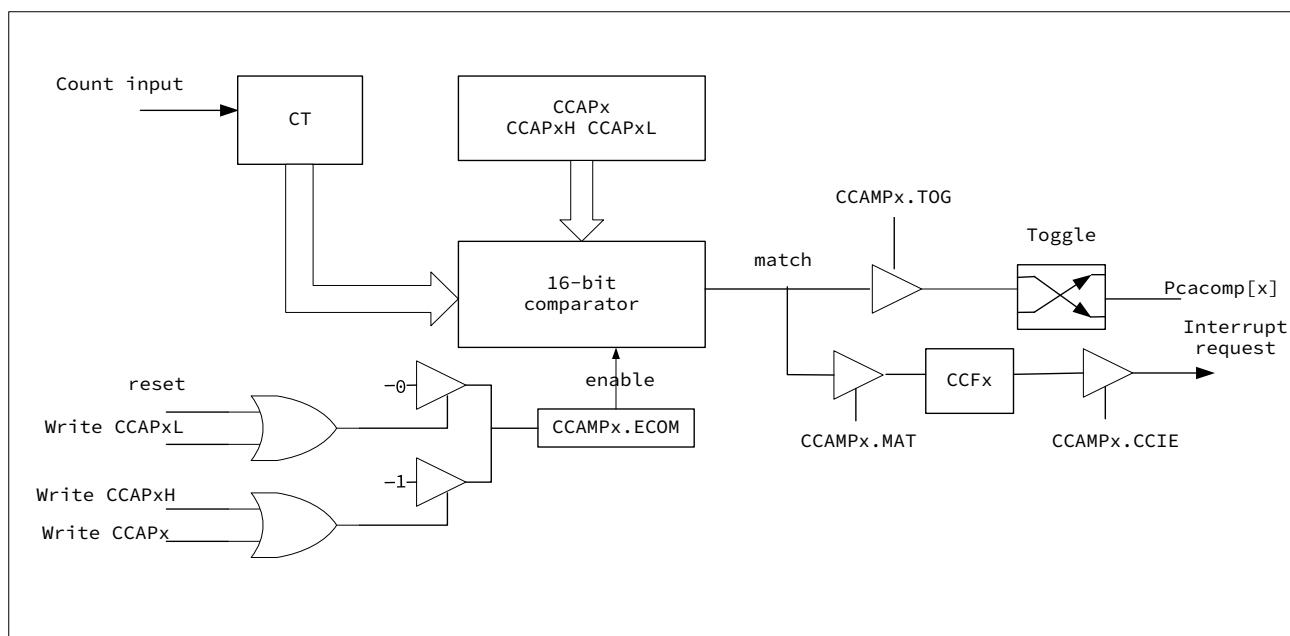


图 16-4 PCA 比较功能框图

16.2.3.2 PCA 16-bit PWM Function

在比较输出 PWM 模式，每当 PCA 计数器内的值与模块的 16 位捕捉/比较寄存器（CCAPx）发生匹配时，模块 PCA 的 CH[x]引脚上的逻辑电平将发生变化。计数器溢出时，CH[x]引脚上的逻辑电平将清零。这可以提供一组 16 位的 PWM 输出。

要设定一个比较/捕获模块的 PWM 模式，设置 CCAPMx.ECOM，CCAPMx.MAT 和 CCAPMx.TOG 位及 EPWM 寄存器。PCA 定时器/计数器和比较/捕获的寄存器（CCAPx）之间的匹配切换 PCA 的 CH[x]信号，并设置模块的比较/捕获标志（CCON.CCFx）。

用户也可以选择产生一个中断请求，通过设置相应的中断使能位（CCAPMx.CCIE）当匹配发生时，即可产生中断请求。由于硬件无法清除的比较/捕获标志中断，用户必须在软件中清除这个标志位。

注：使用 16 位 PWM 时建议不要使用 PCA 的分频，否则占空比输出最大会差别一个周期。

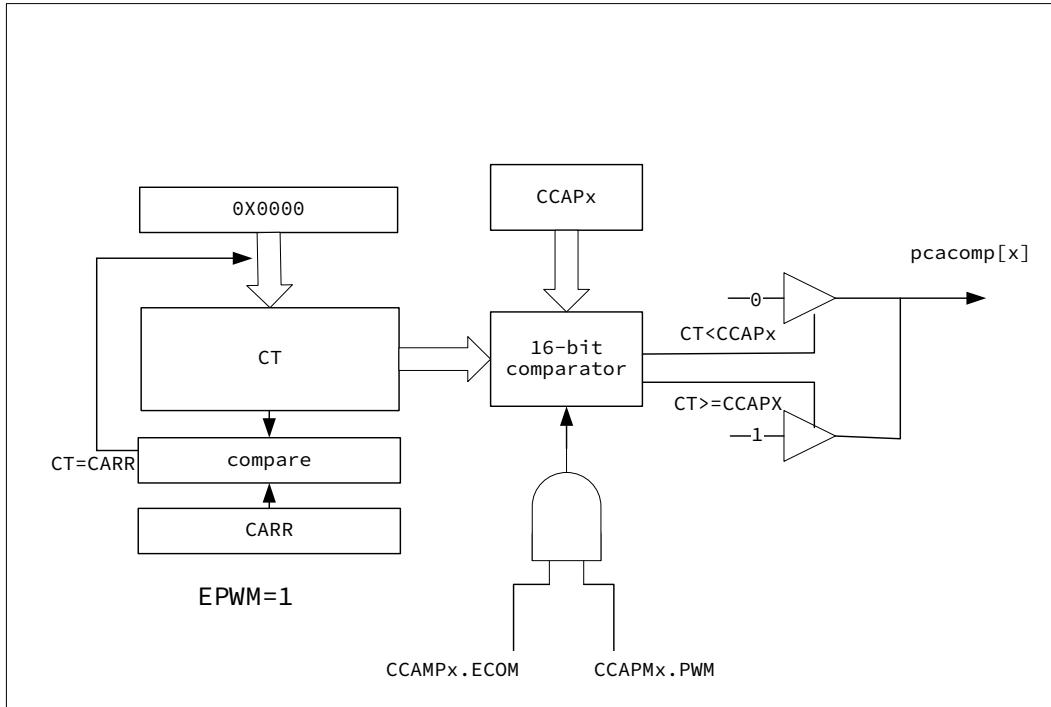


图 16-5 PCA 16 位 PWM 功能框图

16.2.3.3 WDT Function of PCA Module 4

本系列除了一个 WDT 硬件模块，PCA 的模块 4，还提供一个可编程频率的 16 位 WDT。当 PCA 定时器/计数器的计数值与模块 4 中存储的值比较/捕获寄存器（CCAP4）相匹配时，这种模式产生复位信号。PCA 的 WDT 复位信号作为一个独立的复位信号。与外部复位（RST），硬件看门狗复位（WDTRST）和 LVD 低电压复位，POR 上电下电复位相结合。用户可以自由结合或单独使用它们。模块 4 是具有 WDT 模式唯一 PCA 的模块。当不设置为 WDT 时，它可以在其它模式中独立使用。

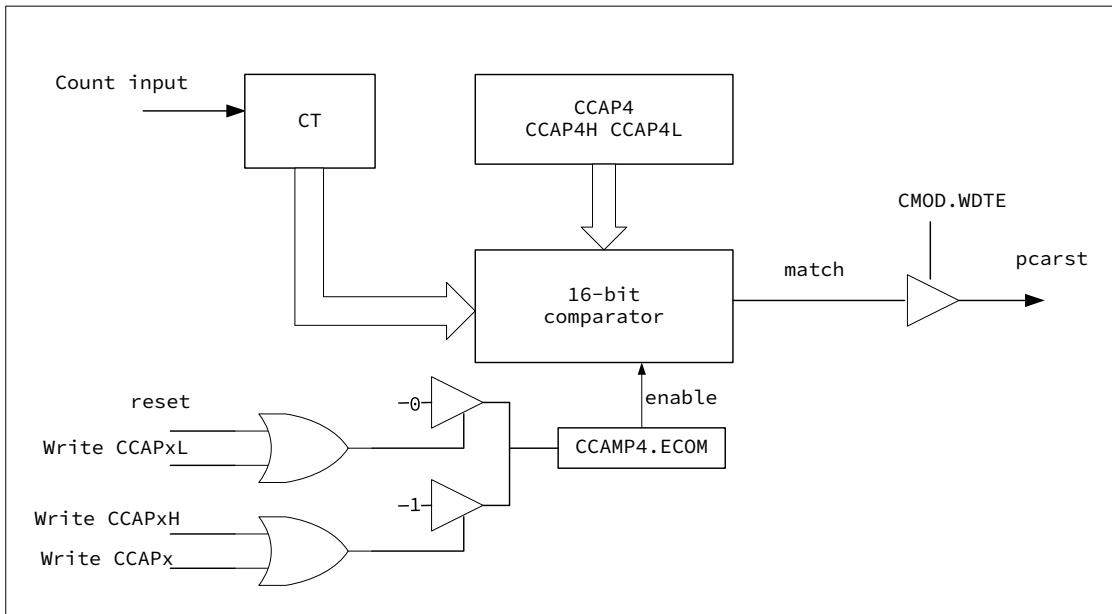


图 16-6 PCA WDT 功能框图

当把 PCA 模块 4 当 WDT 使用时，必须设置 CCAPM4.ECOM4，CCAPM4.MAT4 以及 CMOD.WDTE。另外 PCA 定时器/计数器可以设置 CMOD.CPS 来选择不同的输入计数频率。

在比较/捕获寄存器 (CCAP4) 输入一个 16 位的比较值。在 PCA 定时器/计数器 (CNT)，输入一个 16 位的初始值或使用复位值 (0000h)。这些值乘以的 PCA 输入脉冲率之间的差额确定的 WDT 匹配运行时间。设置定时器/计数器运行控制位 (CCON.CR) 启动 PCA WDT。每次匹配时，PCA 的 WDT 产生复位信号。要防止一个 PCA WDT 复位，用户有三种选择。

- 定期的比较值 CCAP4 的改变，所以匹配永远不会发生。
- 定期更改 PCA 定时器/计数器值 (CNT) 所以匹配永远不会发生。
- 通过在匹配前清除的 CMOD.WDTE 位来禁用模块复位输出信号，后来重新启用它。

前两个选项是更可靠的，因为 WDT 在第三个选项没有被禁用。

第二个选项是不推荐，如果其他 PCA 模块都在使用，因为五个模块共享一个共同的时间基。因此，在大多数应用中的第一个选项是最好的。

PCA WDT 配置流程

- 1) 配置 WDT 比较/捕获寄存器 PCA_CCAP4
- 2) 配置 PCA 计数寄存器 PCA_CNT
- 3) 配置 PCA_CCAMP4 选择比较匹配功能
- 4) 配置 PCA_CMOD 选择输入时钟，使能 WDT 功能
- 5) 启动 PCA
- 6) 选择清除 PCA WDT 清除方式在 PCA WDT 复位前清除 PCA WDT

16.2.3.4 PCA 8-bit PWM Function

脉宽调制是一种使用程序来控制波形占空比，周期，相位的技术。5个PCA模块都可以被独立地用于在对应PCA的CH[x]引脚产生脉宽调制（PWM）输出，脉冲宽度为8位分辨率。PWM输出的频率取决于PCA计数器/定时器的时基。使用模块的捕捉/比较寄存器CCAPxL来改变PWM输出信号的占空比。当PCA计数器/定时器的低字节(CL)与CCAPxL中的值相等时，PCA的CH[x]引脚上的输出被置“1”；当CL中的计数值溢出时，PCA的CH[x]输出被复位“0”。当计数器/定时器的低字节CL溢出时（从0xFF到0x00），保存在CCAPxH中的值被自动装入到CCAPxL，不需软件干预。

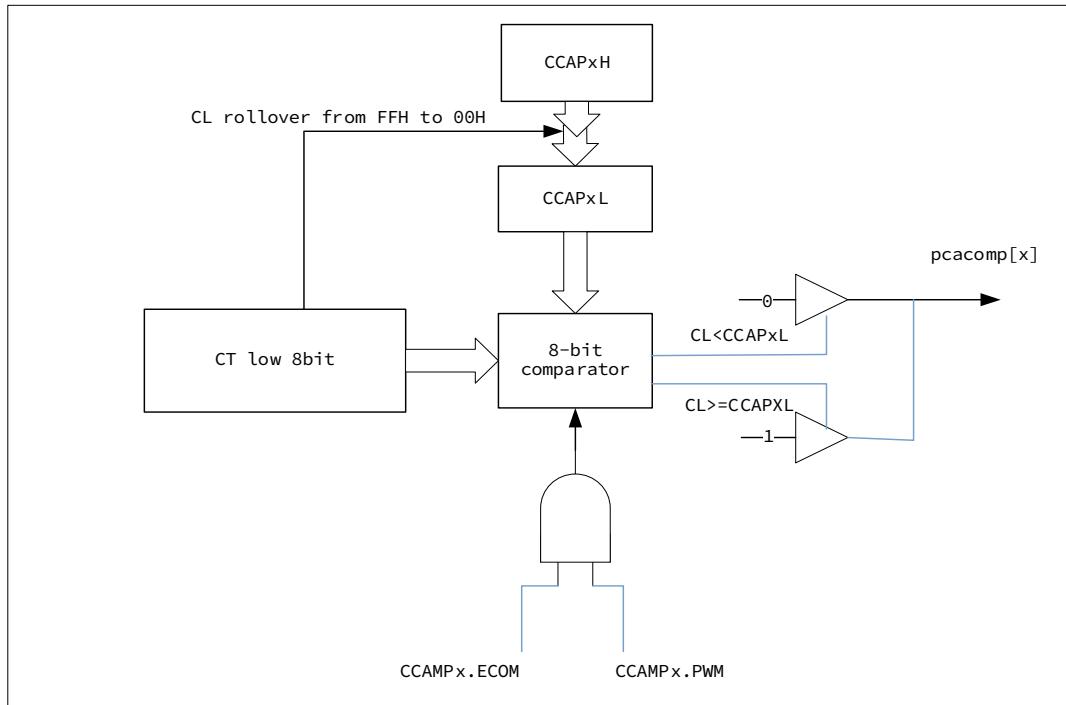


图 16-7 PCA PWM 功能框图

在这种模式下，PCA 定时器/计数器 (CL) 的低字节中的值是不断在低字节比较/捕获寄存器 (CCAPxL) 的值相比。当 $CL < CCAPxL$ ，输出波形为低。当两者匹配时 ($CL = CCAPxL$)，输出波形去到高，直到 CL 溢出从 FFH 到 00H，结束期间仍然很高。在溢出时，在 CCAPxH 的值自动装载到 CCAPxL 内，一个新的周期的开始。

在 CCAPxL 的值决定当前波形的占空比。在 CCAPxH 的值确定下一个波形的占空比。改变 CCAPxL 中的值即可更改的脉冲宽度调制。正如图所示，8位值在 CCAPxL 可以从 0 (100% 占空比)，到 255 (0.4% 占空比)。要改变 CCAPxL 值而不会产生毛刺，需要在高字节寄存器 (CCAPxH) 写入一个新值。当 CL 超过 FFH 滚动到 00h，这个值是由硬件自动加载到 CCAPxL。

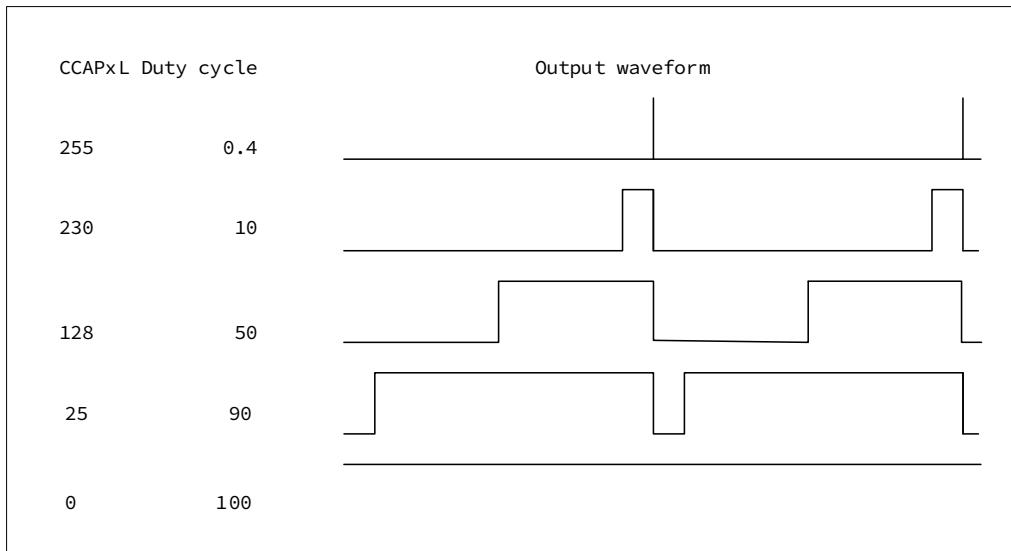


图 16-8 PCA PWM 输出波形

要设定一个比较/捕获模块在 PWM 模式下，需要设置 CCAPMx.ECOM 和 CCAPMx.PWM 位。另外 PCA 定时器/计数器由编程 CMOD.CSP[2:0]可以选择输入计数信号频率。在 CCAPxL 输入一个 8 位的值指定第一个 PWM 波形的占空比。在 CCAPxH 输入一个 8 位的值会指定第二个 PWM 波形的占空比。设置定时器/计数器运行控制位（CCON.CR）启动 PCA 定时器/计数器。

表 16-1 PCA 比较捕获功能模块设置

ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	EPWM	工作方式
X	1	0	0	0	0	X	0	用正沿触发捕获
X	0	1	0	0	0	X	0	用负沿触发捕获
X	1	1	0	0	0	X	0	用跳变沿触发捕获
1	0	0	1	0	0	X	0	软件定时器
1	0	0	1	1	0	X	0	高速输出
1	0	0	0	0	1	X	0	8 位脉冲宽度调制器
1	0	0	1	1	0	X	1	16 位脉冲宽度调制器

16.3 Interconnection and Control between PCA Module and Other Modules

PCA 通过端口功能选择可以连通到其他模块或端口。

		0	1	2	3	4	5	6	7
GPIO_PCAS[2:0]	PCA_ECI	PX_SEL	PCNT1	LVD	VCO	VC1	PA05	PB02	PD02
GPIO_PCAS[5:3]	PCA_CH0	PX_SEL	PCNT0	PCNT1	LVD	VC1	PA06	PB04	PC06

当 GPIO_PCAS[2:0]=0x0 时，PCA_ECI 输入为 PX_SEL 选择的端口输入，当 GPIO_PCAS[2:0]=0x1~0X7 时，连接其他模块的输入或输出。

当 GPIO_PCAS[5:3]=0x0 时，PCA_CH0 捕获输入为 PX_SEL 选择的端口输入，当 GPIO_PCAS[5:3]=0x1~0X7 时，连接其他模块的输入或输出。

16.4 PCA Register Description

基地址 0X40001000

表 16-2 PCA 寄存器列表

寄存器	偏移地址	描述
PCA_CCON	0X000	PCA 控制寄存器
PCA_CMOD	0X004	PCA 模式寄存器
PCA_CNT	0X008	PCA 计数寄存器
PCA_ICLR	0X00C	PCA 中断清除寄存器
PCA_CCAPM0	0x010	PCA 比较/捕获模块 0 模式寄存器
PCA_CCAPM1	0x014	PCA 比较/捕获模块 1 模式寄存器
PCA_CCAPM2	0x018	PCA 比较/捕获模块 2 模式寄存器
PCA_CCAPM3	0x01C	PCA 比较/捕获模块 3 模式寄存器
PCA_CCAPM4	0x020	PCA 比较/捕获模块 4 模式寄存器
PCA_CCAP0H	0X024	PCA 比较/捕获模块 0 高 8 位寄存器
PCA_CCAP0L	0X028	PCA 比较/捕获模块 0 低 8 位寄存器
PCA_CCAP1H	0X02C	PCA 比较/捕获模块 1 高 8 位寄存器
PCA_CCAP1L	0X030	PCA 比较/捕获模块 1 低 8 位寄存器
PCA_CCAP2H	0X034	PCA 比较/捕获模块 2 高 8 位寄存器
PCA_CCAP2L	0X038	PCA 比较/捕获模块 2 低 8 位寄存器
PCA_CCAP3H	0X03C	PCA 比较/捕获模块 3 高 8 位寄存器
PCA_CCAP3L	0X040	PCA 比较/捕获模块 3 低 8 位寄存器
PCA_CCAP4H	0X044	PCA 比较/捕获模块 4 高 8 位寄存器
PCA_CCAP4L	0X048	PCA 比较/捕获模块 4 低 8 位寄存器
PCA_CCAPO	0X04C	PCA PWM 与高速输出标志寄存器
PCA_CCAP0	0X050	PCA 比较/捕获模块 0 的 16 位寄存器
PCA_CCAP1	0X054	PCA 比较/捕获模块 1 的 16 位寄存器
PCA_CCAP2	0X058	PCA 比较/捕获模块 2 的 16 位寄存器
PCA_CCAP3	0X05C	PCA 比较/捕获模块 3 的 16 位寄存器
PCA_CCAP4	0X060	PCA 比较/捕获模块 4 的 16 位寄存器
PCA_CARR	0X064	PCA 周期装载寄存器
PCA_EPWM	0X068	PCA PWM 增强寄存器

16.4.1 Control Register (PCA_CCON)

偏移地址: 0x000

复位值: 0x0000 0000h

	31:8	7	6	5	4	3	2	1	0
	Reserved	CF	CR	Reserved	CCF4	CCF3	CCF2	CCF1	CCF0
		RO	RW		RO	RO	RO	RO	RO

位	符号	描述
31:8	Reserved	保留位
7	CF	PCA 计数器溢出标志 (写无效) 当 PCA 计数溢出时, CF 由硬件置位, 如果 CMOD 寄存器的 CFIE 位为 1, 则 CF 标志可以产生中断 1: 发生计数器溢出; 0: 无溢出;
6	CR	PCA 计数器运行控制位 1: 启动 PCA 计数器计数 0: 关闭 PCA 计数器计数
5	Reserved	保留位
4	CCF4	PCA 计数器模块 4 比较/捕获标志位 当出现匹配或捕获时, 该位由硬件置位。(写无效) 当 CCAPM4.CCIE 置位时, 这个标志位会产生一个 PCA 中断
3	CCF3	PCA 计数器模块 3 比较/捕获标志位 当出现匹配或捕获时, 该位由硬件置位。(写无效) 当 CCAPM3.CCIE 置位时, 这个标志位会产生一个 PCA 中断
2	CCF2	PCA 计数器模块 2 比较/捕获标志位 当出现匹配或捕获时, 该位由硬件置位。(写无效) 当 CCAPM2.CCIE 置位时, 这个标志位会产生一个 PCA 中断
1	CCF1	PCA 计数器模块 1 比较/捕获标志位 当出现匹配或捕获时, 该位由硬件置位。(写无效) 当 CCAPM1.CCIE 置位时, 这个标志位会产生一个 PCA 中断
0	CCF0	PCA 计数器模块 0 比较/捕获标志位 当出现匹配或捕获时, 该位由硬件置位。(写无效) 当 CCAPM0.CCIE 置位时, 这个标志位会产生一个 PCA 中断

16.4.2 Mode Register (PCA_CMOD)

偏移地址: 0x004

复位值: 0x0000 0000h

	31-8	7	6	5	4	3	2	1	0
Reserved		CIDL	WDTE		Reserved		CPS		CFIE
		RW	RW				RW		RW

位	符号	描述
31:8	Reserved	保留位
7	CIDL	空闲模式 IDLE 下, PCA 是否停止工作 1: 休眠模式 (sleep) 下, PCA 停止工作 0: 休眠模式 (sleep) 下, PCA 继续工作
6	WDTE	PCA WDT 功能使能控制位 1: 启动 PCA 模块 4 WDT 功能 0: 关闭 PCA 模块 4 WDT 功能
5:4	Reserved	保留位
3:1	CPS[2:0]	时钟分频选择及时钟源选择 000: PCLK/32 001: PCLK/16 010: PCLK/8 011: PCLK/4 100: PCLK/2 101: timer0 overflow 110: timer1 overflow 111: ECI 外部时钟, 时钟 PCLK 四分频采样
0	CFIE	PCA 计数器中断使能控制信号 1: 使能中断 0: 关闭中断

16.4.3 Count Register (PCA_CNT)

偏移地址: 0x008

复位值: 0x0000 0000h

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

位	符号	描述
31:16	Reserved	保留位
15:0	CNT	定时器计数器的值 只有在 PCA 停止状态 CNT 才可以写入。否则写入无效

16.4.4 Interrupt Clear Register (PCA_ICLR)

偏移地址: 0x00C

复位值: 0x0000 009Fh

31-8	7	6	5	4	3	2	1	0
Reserved	CF	Reserved		CCF4 R1W0	CCF3 R1W0	CCF2 R1W0	CCF1 R1W0	CCF0 R1W0

位	符号	描述
31:8	Reserved	保留位
7	CF	PCA 计数器溢出标志清除 (软件写 0 清零, 写 1 无效), 读出值为 1
6:5	RSV	保留位
4	CCF4	PCA 计数器模块 4 比较/捕获标志位清除 (软件写 0 清零, 写 1 无效), 读出值为 1
3	CCF3	PCA 计数器模块 3 比较/捕获标志位清除 (软件写 0 清零, 写 1 无效), 读出值为 1
2	CCF2	PCA 计数器模块 2 比较/捕获标志位清除 (软件写 0 清零, 写 1 无效), 读出值为 1
1	CCF1	PCA 计数器模块 1 比较/捕获标志位清除 (软件写 0 清零, 写 1 无效), 读出值为 1
0	CCF0	PCA 计数器模块 0 比较/捕获标志位清除 (软件写 0 清零, 写 1 无效), 读出值为 1

16.4.5 Compare Capture Mode Register (PCA_CCAPM0~4)

偏移地址：

CCAPM0: 0x010; CCAPM1: 0x014; CCAPM2: 0x018;

CCAPM3: 0x01C; CCAPM4: 0x020;

复位值: 0x0000 0000h

31-8	7	6	5	4	3	2	1	0
Reserved		ECOM	CAPP	CAPN	MAT	TOG	PWM	CCIE
		RW	RW	RW	RW	RW	RW	RW

位	符号	描述
31:7	Reserved	保留位
6	ECOM	允许比较器功能控制位 1: 允许比较器功能; 0: 禁止比较器功能; 当 PCA 用于软件计数器, 高速输出, PWM 模式, WDT 模式, 要置位 ECOM 写 CCAMPx 或 CCAMPx 寄存器会自动置位 ECOM; 写 CCAMPLx 寄存器会自动清除 ECOM 位
5	CAPP	正沿捕获控制位 1: 允许上升沿捕获; 0: 禁止上升沿捕获
4	CAPN	负沿捕获控制位 1: 允许下降沿捕获; 0: 禁止下降沿捕获
3	MAT	允许匹配控制位 1: PCA 计数值与模块的比较/捕获寄存器的值一旦匹配, 将置位 CCON 寄存的中断标志 CCFx(x=0-4) 0: 禁止匹配功能
2	TOG	翻转控制位 1: 工作在 PCA 高速输出模式, PCA 计数器的值与模块的比较/捕获寄存器的值一旦匹配, CCPx 引脚翻转 0: 禁止翻转功能
1	PWM	脉宽调制控制位 1: 允许 CCPx 引脚作为 PWM 输出 0: 禁止 PWM 脉宽调制功能 只有 CCAPMx[6:0]=100_0010 时, PWM 功能才有效
0	CCIE	PCA 使能中断 1: 使能比较/捕获中断 0: PCA 比较/捕获功能中断禁止

16.4.6 Compare Capture Data Register High 8 Bits (PCA_CCAP0~4H)

偏移地址

CCAP0H: 0x024; CCAP1H: 0x02C; CCAP2H: 0x034;

CCAP3H: 0x03C; CCAP4H: 0x044;

复位值: 0x0000 0000h

	31:8	7	6	5	4	3	2	1	0
Reserved	CCAPx[15: 8]								
	RW								

位	符号	描述
31:8	Reserved	保留位
7:0	CCAPx[15:8]	<p>比较/捕获模式高 8 位寄存器 当 PCA 模式用于比较/捕获模式时，用于保存 16 位捕获计数值的高 8 位；写 CCAPxH 寄存器会自动置位寄存器 CCAPMx 的 ECOM 位。</p> <p>当 PCA 模式用于 PWM 模式时，用于控制输出占空比装载寄存器，在计数器低 8 位溢出时，装载寄存器会自动更新到 PWM 比较寄存器</p>

16.4.7 Compare Capture Data Register Lower 8 Bits (PCA_CCAP0~4L)

偏移地址

CCAP0L: 0x028; CCAP1L: 0x030; CCAP2L: 0x038;

CCAP3L: 0x040; CCAP4L: 0x048;

复位值: 0x0000 0000h

	31:8	7	6	5	4	3	2	1	0
Reserved	CCAPx[7: 0]								
	RW								

位	符号	描述
31:8	Reserved	保留位
7:0	CCAPx[7:0]	<p>比较/捕获模式低 8 位寄存器 当 PCA 模式用于比较/捕获模式时，用于保存 16 位捕获计数值的低 8 位；写 CCAPxL 寄存器会自动清除寄存器 CCAPMx 的 ECOM 位。</p> <p>当 PCA 模式用于 PWM 模式时，用于控制输出占空比比较寄存器，在 PWM 模式，计数器的低 8 位的值小于 CCAPx[7:0]的值 PWM 输出低电平，否则 PWM 输出高电平。</p>

16.4.8 Compare Capture 16-bit Register (PCA_CCAP0~4)

偏移地址

CCAP0: 0x050; CCAP1: 0x054; CCAP2: 0x058;

CCAP3: 0x05C; CCAP4: 0x060;

复位值: 0x0000 0000h

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

位	符号	描述
31:16	Reserved	保留位
15:0	CCAPx	比较/捕获模式 16 位寄存器 当 PCA 模式用于比较/捕获模式时，用于保存 16 位捕获计数值；写 CCAPx 寄存器会置位寄存器 CCAPMx 的 ECOM 位。 写 CCAPX 寄存器相当于写 CCAPxL 及 CCAPxH 这两个 8 位寄存器。在比较/捕获模式下可以直接读写这个寄存器，在 PWM 模式下，使用 CCAPxL 及 CCAPxH 寄存器

16.4.9 Compare High-Speed Output Flag Register (PCA_CCAPO)

偏移地址: 0x04C

复位值: 0x0000 0000h

31:8	7	6	5	4	3	2	1	0	
Reserved					CCAPO4	CCAPO3	CCAPO2	CCAPO1	CCAPO0
					RW	RW	RW	RW	RW

位	符号	描述
31:5	Reserved	保留位
4	CCAPO4	比较模块 4 的输出值
3	CCAPO3	比较模块 3 的输出值
2	CCAPO2	比较模块 2 的输出值
1	CCAPO1	比较模块 1 的输出值
0	CCAPO0	比较模块 0 的输出值

16.4.10 Period Register (PCA_CARR)

偏移地址: 0x064

复位值: 0x0000 0000h

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

位	符号	描述
31:16	Reserved	保留位
15:0	CARR	计数周期重载寄存器

16.4.11 Enhanced PWM Control (PCA_EPWM)

偏移地址: 0x068

复位值: 0x0000 0000h

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

位	符号	描述
31:1	Reserved	保留位
0	EPWM	16 bit PWM 使能

17 Advanced timers (TIM4/5/6)

17.1 Advanced Timer Introduction

Advanced Timer 是一个包含三个定时器 Timer4/5/6。Timer4/5/6 功能相同的高性能计数器，可用于计数产生不同形式的时钟波形，1 个定时器可以产生互补的一对 PWM 或者独立的 2 路 PWM 输出，可以捕获外界输入进行脉冲宽度或周期测量。

Advanced Timer 基本的功能及特性如表所示。

表 17-1 Advanced Timer 基本特性

波形模式	锯齿波、三角波
基本功能	• 递加、递减计数方向
	• 软件同步
	• 硬件同步
	• 缓存功能
	• 正交编码计数
	• 通用 PWM 输出
	• 保护机制
	• AOS 关联动作
中断类型	计数比较匹配中断
	计数周期匹配中断
	死区时间错误中断

表 17-2 Advanced Timer 端口列表

端口名	方向	功能
TIMx_CHA	输入/输出	正交编码计数时钟输入端口或捕获输入端口或比较输出端口 (x=4~6)
TIMx_CHB		2) 硬件启动、停止、清零条件输入端口
TIMTRIA	输入	硬件计数时钟输入端口或捕获输入端口
TIMTRIB		硬件启动、停止、清零条件输入端口
TIMTRIC		
TIMTRID		

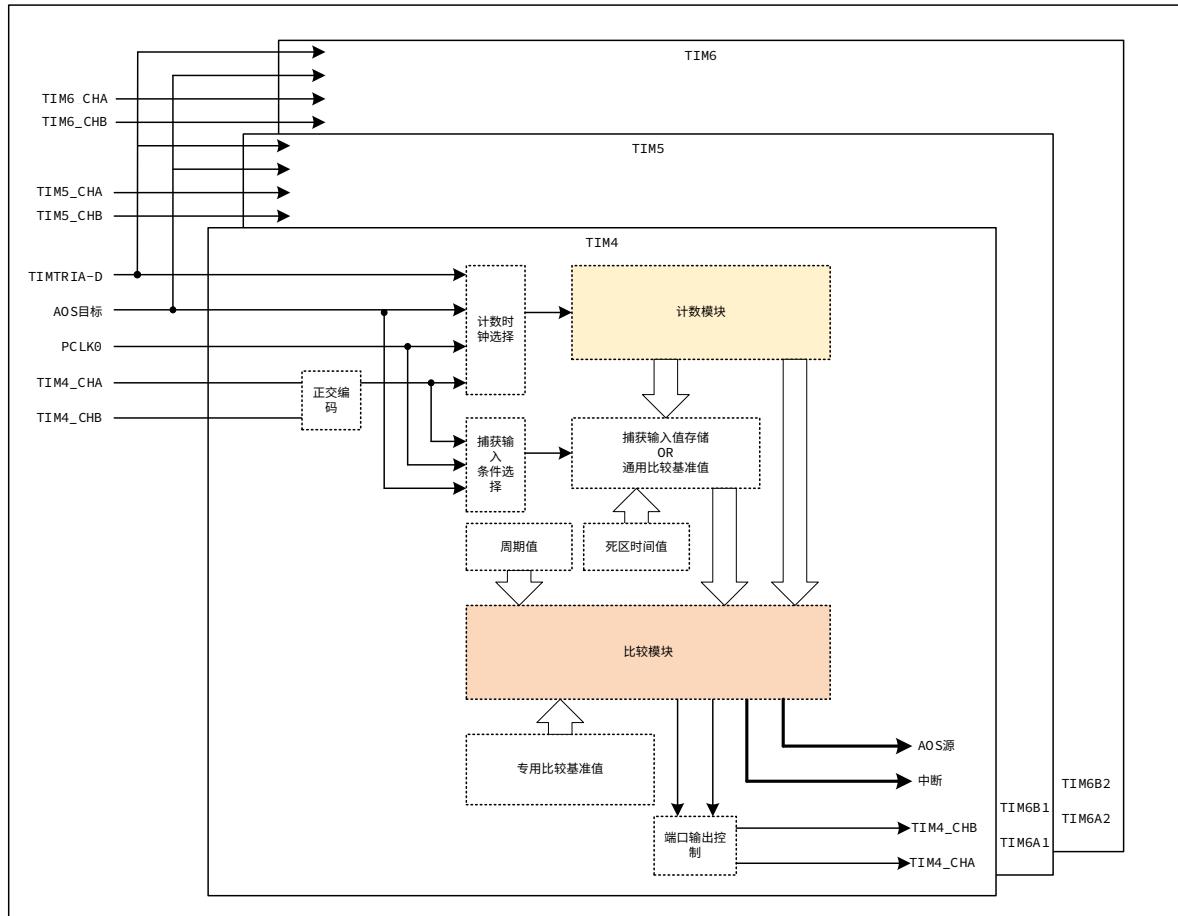


图 17-1 Advanced Timer 框图

17.2 Advanced Timer Function Description

17.2.1 Basic Actions

17.2.1.1 Basic Waveform Mode

Timer4/5/6 有 2 种基本计数波形模式，锯齿波模式和三角波模式。波形模式又由于不同的内部计数动作有所细分，三角波模式分为三角波 A 模式、三角波 B 模式。锯齿波和三角波的基本波形如图 17-2 图 17-3 所示。三角波 A 模式与三角波 B 模式区别在于缓存传送有差别，三角波 A 模式一个周期只发生一次缓存传送（谷点），而三角波 B 模式一个周期发生两次缓存传送（峰点和谷点）。

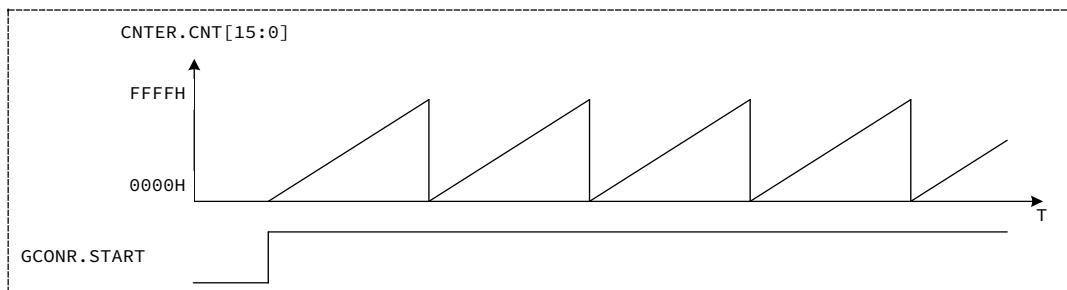


图 17-2 锯齿波波形 (递加计数)

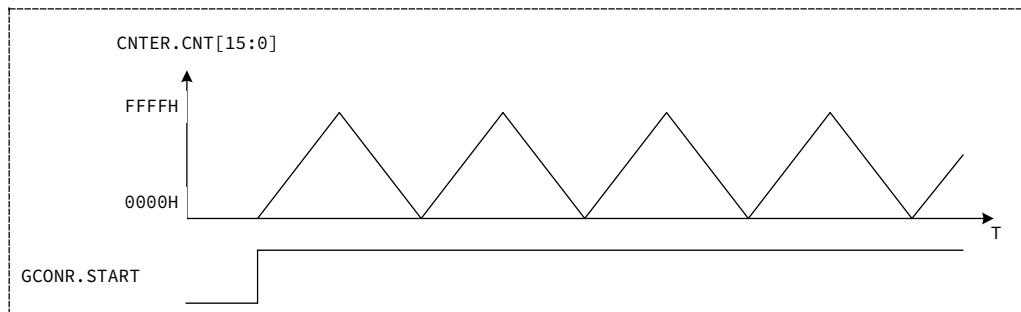


图 17-3 三角波波形

17.2.1.2 Comparison Output

Timer4/5/6 一个定时器有 2 个比较输出端口 (CHxA、CHxB)，可在计数值与计数基准值比较匹配时输出指定的电平。GCMAR、GCMBR 寄存器分别对应了 CHxA、CHxB 的计数比较基准值。当计数器的计数值和 GCMAR 相等时，CHxA 端口输出指定的电平；当计数器的计数值和 GCMBR 相等时，CHxB 端口输出指定电平。

CHxA、CHxB 端口的计数起始电平、停止电平、计数比较匹配时的电平等，可由端口控制寄存器 (PCONR) 的 PCONR.STACA、PCONR.STPCA、PCONR.STASTPSA、PCONR.CMPCA[1:0]、PCONR.PERCA[1:0] 和 PCONR.STACB、PCONR.STPCB、PCONR.STASTPSB、PCONR.CMPCB[1:0]、PCONR.PERCB[1:0] 位设定。图 17-4 为比较输出的动作例。

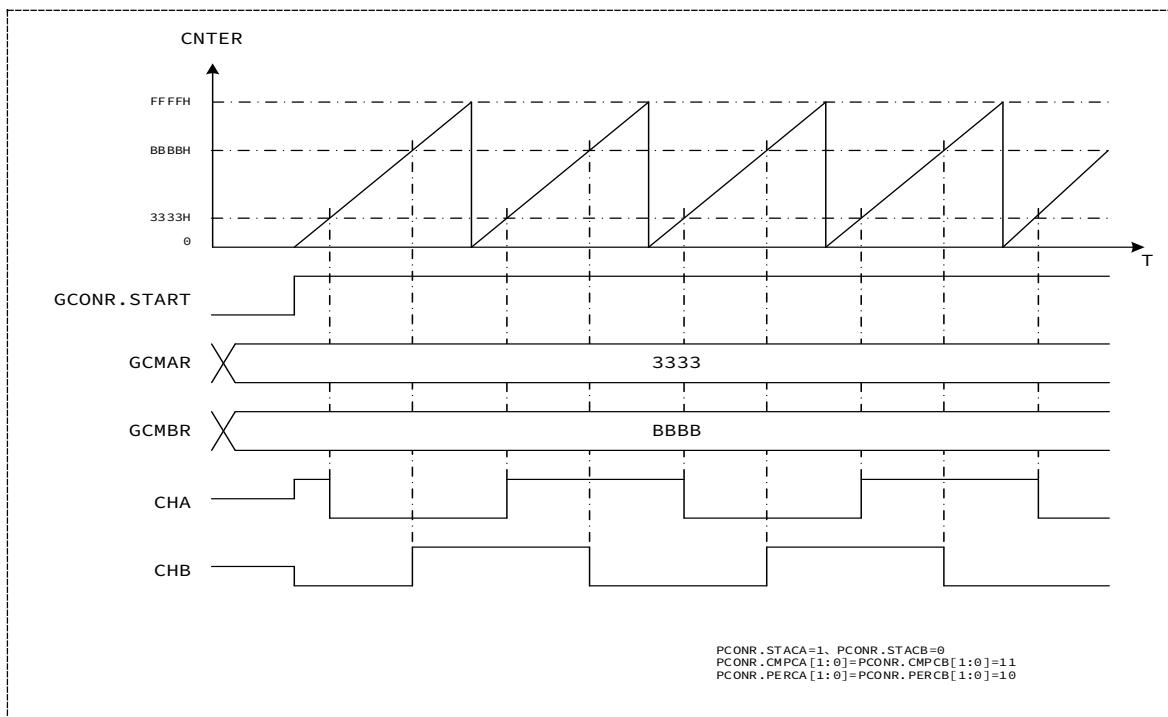


图 17-4 比较输出动作

17.2.1.3 Capturing Input

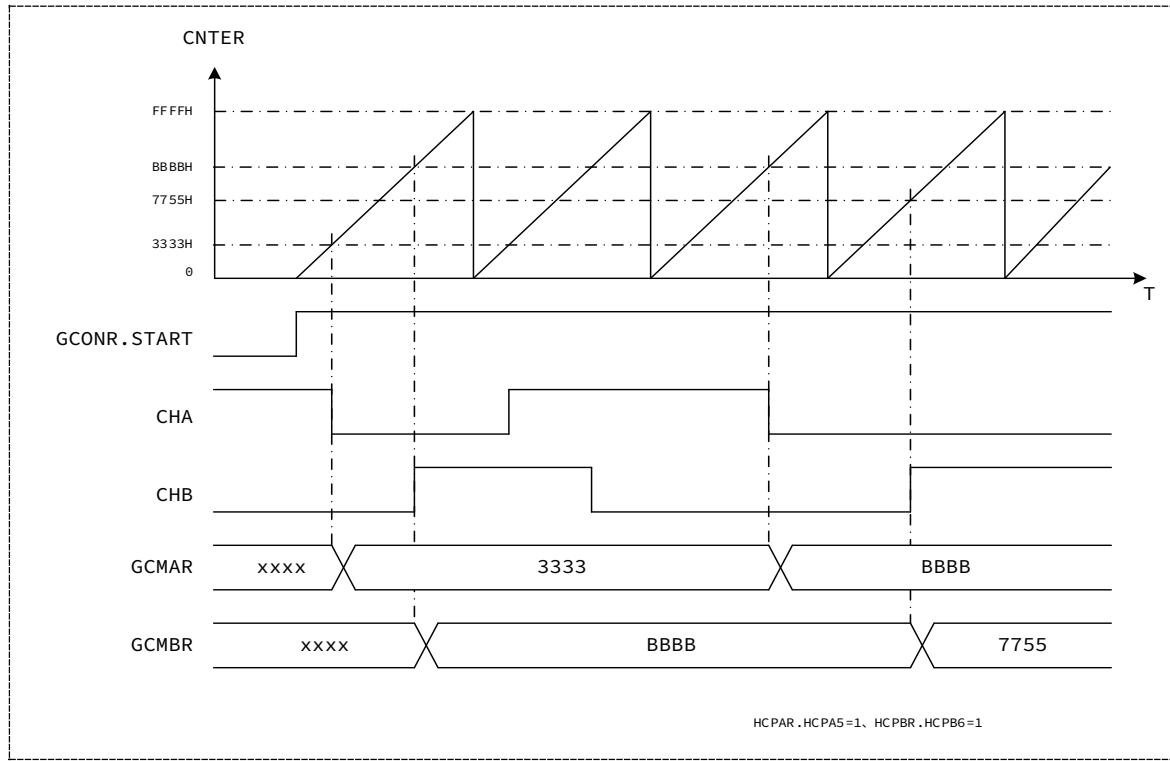


图 17-5 捕获输入动作

Timer4/5/6 都具有捕获输入功能，具备 2 组捕获输入寄存器（GCMAR、GCMBR），用于保存捕获到的计数值。设定端口控制寄存器（PCONR）的 PCONR.CAPCA、PCONR.CAPCB 位为 1，捕获输入功能就有效了。当设定了对应的捕获输入条件且该条件有效时，当前的计数值就被保存到相应的寄存器（GCMAR、GCMBR）中。

每组捕获输入的条件可以是 AOS 事件触发、TIMTRIA-TIMTRID 输入、CHxA 或 CHxB 的输入等，具体的条件选择可通过硬件捕获事件选择寄存器（HCPAR、HCPBR）来设定。图 17-5 为捕获输入的动作例。

17.2.2 Clock Source Selection

Timer4/5/6 的计数时钟可以有以下几种选择：

- PCLK 及 PCLK 的 2、4、8、16、64、256、1024 分频 (GCONR.CKDIV[2:0] 设定)
- AOS 事件触发输入 (HCUPR.HCUP [19:16] 或 HCDOR.HCDO [19:16] 设定)
- CHxA 和 CHxB 的正交编码输入 (HCUPR.HCUP[7:0] 或 HCDOR.HCDO [7:0] 设定)
- TIMTRIA-TIMTRID 的端口输入 (HCUPR.HCUP [15:8] 或 HCDOR.HCDO [15:8] 设定)

从上述描述可以看到，b、c、d 时钟互相独立，可分别设定有效或无效，并且当选择 b、c、d 时钟时，a 时钟自动无效。

17.2.3 Counting direction

Timer4/5/6 的计数器计数方向可通过软件方式改变。不同波形模式时，改变计数方向的方法略有不同。

17.2.3.1 Sawtooth Wave Counting Direction

锯齿波模式时，计数方向可在计数器计数中或停止时设定。

在递加计数中时，设定 GCONR.DIR=0（递减计数），则计数器计数到上溢后变为递减计数模式；在递减计数中时，设定 GCONR.DIR=1（递加计数），则计数器计数到下溢后变为递加计数模式。

在计数停止时，设定 GCONR.DIR 位。则计数开始后直至上溢或下溢时，GCONR.DIR 的设定才会反映到计数中。

17.2.3.2 Triangle wave counting direction

三角波模式时，计数方向只能在计数器停止时设定。在计数中设定计数方向无效。

在计数停止时，设定 GCONR.DIR 位。则计数开始后直至上溢或下溢时，GCONR.DIR 的设定才会反映到计数中。

17.2.4 Digital Filtering

Timer4/5/6 的 CHxA、CHxB、TIMTRIA~D 端口输入都有数字滤波功能。可通过设定滤波控制寄存器 (FCONR) 的相关使能位开启对应端口的滤波功能。滤波用的基准时钟也通过滤波控制寄存器 (FCONR) 设定。

在滤波采样基准时钟采样到端口上 3 次一致的电平时，该电平被当作有效电平传送到模块内部；小于 3 次一致的电平会被当作外部干扰滤掉，不传送到模块内部。其动作例如图 17-6 所示。

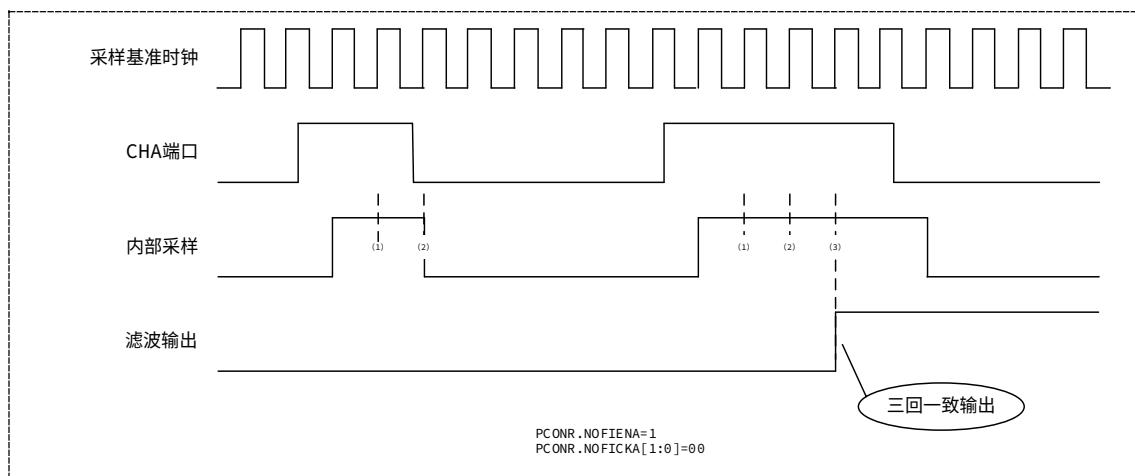


图 17-6 捕获输入端口的滤波功能

TIMTRIA~D 端口是一组 Timer4/5/6 间共用的端口，该组端口的数字滤波功能只在 Timer4 实现，其他定时器 Timer5/6 对该组端口的数字滤波功能设定无效。

17.2.5 Software Synchronization

17.2.5.1 Software Synchronous Startup

Timer4/5/6 可通过设定软件同步启动寄存器 (SSTAR) 的相关位，实现目标 Timer4/5/6 的同步启动。

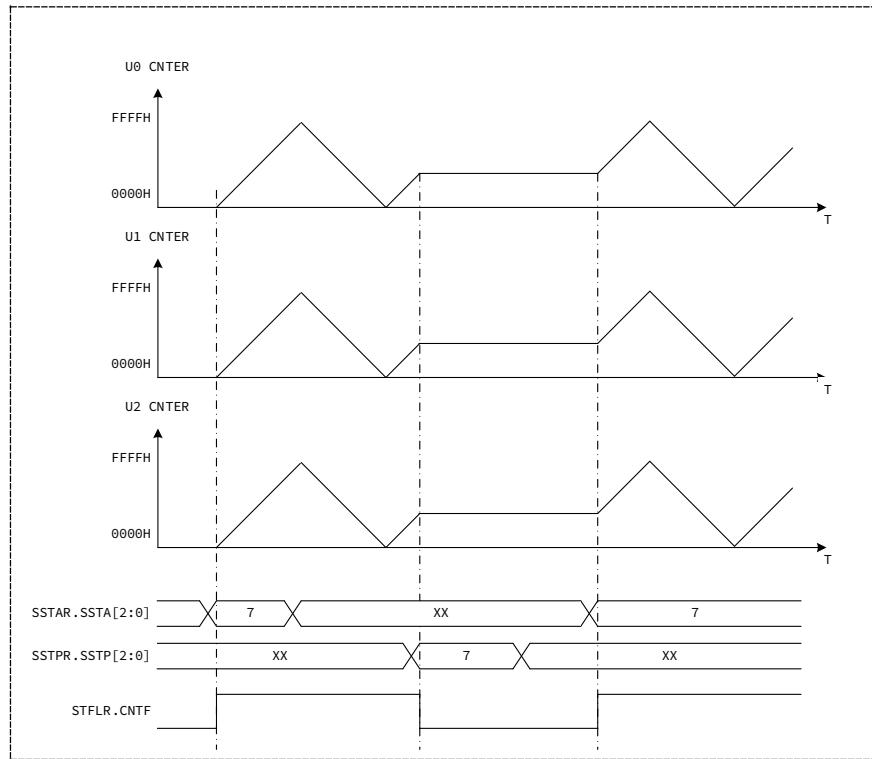


图 17-7 软件同步动作

17.2.5.2 Software Synchronous Stop

Timer4/5/6 可通过设定软件同步停止寄存器 (SSTPR) 的相关位，实现目标 Timer4/5/6 的同步停止。

17.2.5.3 Software Synchronous Clear

Timer4/5/6 可通过设定软件同步清零寄存器 (SCLRR) 的相关位，实现目标 Timer4/5/6 的同步清零。

如图 17-7 所示、若设定 Timer4 的 SSTAR.SSTA0=SSTAR.SSTA1=SSTAR.SSTA2, 即可实现 Timer4/5/6 的软件同步启动。

软件同步动作相关寄存器 (SSTAR、SSTPR、SCLRR) 是一组独立于 Timer4/5/6 外、各个 TIM 间共用的寄存器，这组寄存器的各个位只在写 1 时有效，写 0 无效。在读取 SSTAR 寄存器时，会读出各个定时器的计数器状态，在读取 SSTPR 或 SCLRR 时，会读出 0。

17.2.6 Hardware Synchronization

每个定时器除独立拥有 2 个通用输入端口 (CHxA、CHxB) 外，还共同拥有 4 个外部通用输入端口 (TIMTRIA、TIMTRIB、TIMTRIC、TIMTRID) 及 4 个 AOS 目标，可实现定时器间的硬件同步动作。

17.2.6.1 Hardware Synchronous Startup

各 Timer4/5/6 均可选择用硬件方式启动计数器,选择相同硬件启动条件的定时器即可在启动条件有效时实现同步启动。具体的硬件启动条件由硬件启动事件选择寄存器 (HSTAR) 的设定来决定。

17.2.6.2 Hardware Synchronous Stop

各 Timer4/5/6 均可选择用硬件方式停止计数器,选择相同硬件停止条件的定时器即可在停止条件有效时实现同步停止。具体的硬件停止条件由硬件停止事件选择寄存器 (HSTPR) 的设定来决定。

17.2.6.3 Hardware Synchronous Clear

各 Timer4/5/6 均可选择用硬件方式清零计数器,选择相同硬件清零条件的定时器即可在清零条件有效时实现同步清零。具体的硬件清零条件由硬件清零事件选择寄存器 (HCLRR) 的设定来决定。

17.2.6.4 Hardware Synchronous Capture Input

各 Timer4/5/6 均可选择用硬件方式实现捕获输入功能,选择相同捕获输入功能条件的定时器即可在捕获输入功能条件有效时实现同步捕获输入。具体的硬件捕获输入功能条件由硬件捕获事件选择寄存器 (HCPAR、HCPBR) 的设定来决定。

17.2.6.5 Hardware Synchronous Count

Timer4/5/6 均可选择用硬件输入作为 CLOCK 进行计数，选择相同硬件计数条件的定时器即可在硬件计数 CLOCK 有效时实现同步计数。具体的硬件计数条件由硬件递加事件选择寄存器 (HCUPR) 和硬件递减事件选择寄存器 (HCDOR) 的设定来决定。

选择硬件同步计数功能时，只是选择了外部输入时钟源，不影响计数器的启动、停止、清零动作。计数器的启动、停止、清零等还需要单独设定。

图 17-8 所示、Timer4/5/6 的硬件同步动作例。

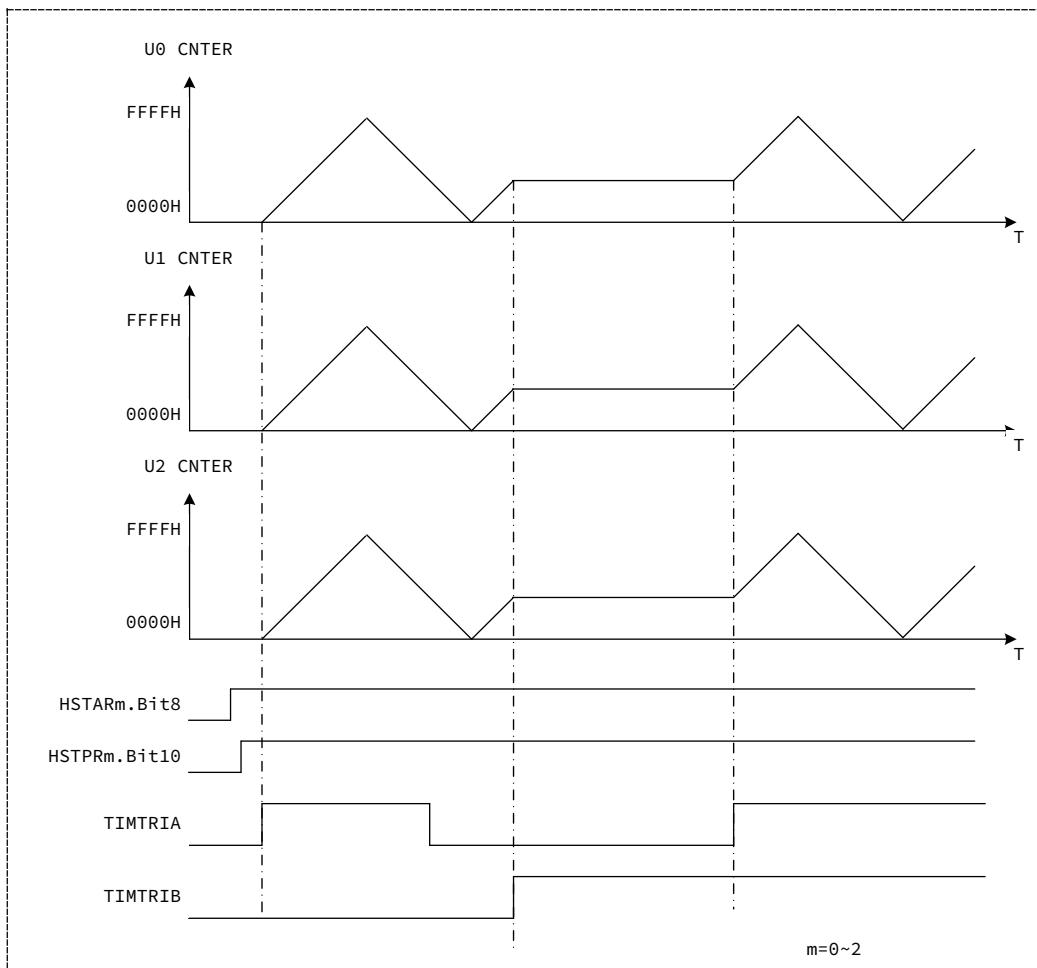


图 17-8 硬件同步动作

17.2.7 Cache Function

缓存动作是指通过设定缓存控制寄存器 (BCONR)，在缓存传送时间点，选择发生以下事件：

- 通用周期基准值缓存寄存器 (PERBR) 的值自动传送到通用周期基准值寄存器 (PERAR) 中
- 通用比较基准值缓存寄存器 (GCMCR、GCMDR) 的值自动传送到通用比较基准值寄存器 (GCMAR、GCMBR) 中 (比较输出时)
- 通用比较基准值寄存器 (GCMAR、GCMBR) 的值自动传送到通用比较基准值缓存寄存器 (GCMCR、GCMDR) 中 (捕获输入时)

图 17-9 所示，是比较输出动作时、通用比较基准值寄存器的单缓存方式的时序图。从图中可以看到，在计数期间改变通用比较基准值寄存器 (GCMAR) 的值可以调整输出占空比，改变通用周期基准值寄存器 (PERAR) 的值可以调整输出周期。

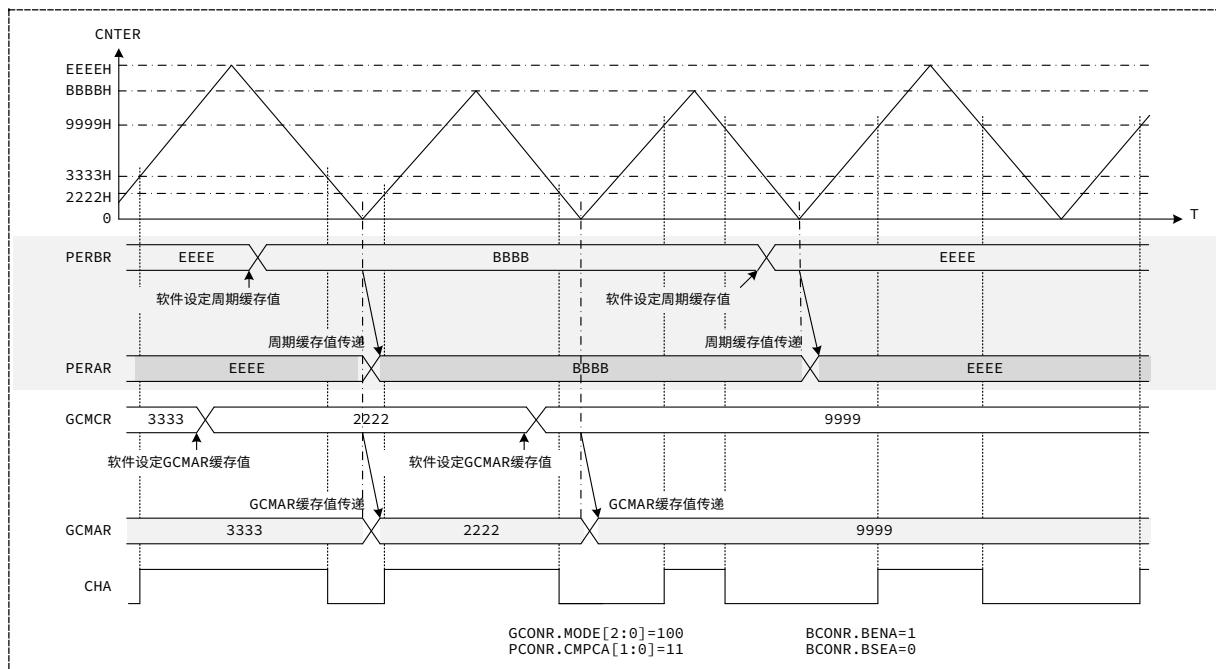


图 17-9 单缓存方式比较输出时序

17.2.7.1 Cache transfer time point

通用周期基准值缓存传送时间点

周期基准值缓存传送时间点为锯齿波时递加计数上溢点或递减计数下溢点、三角波时计数谷点。

通用比较基准值缓存传送时间点

锯齿波 A 模式时，设定 BCONR.BENA=1 或 BCONR.BNEB=1，缓存动作有效。缓存传送发生在上溢点或下溢点。

三角波 A 模式时，设定 BCONR.BENA=1 或 BCONR.BNEB=1，缓存动作有效。缓存传送发生在计数谷点。

三角波 B 模式时，设定 BCONR.BENA=1 或 BCONR.BNEB=1，缓存动作有效。缓存传送发生在计数谷点和计数峰点。

捕获输入值缓存传送时间点

捕获输入动作缓存传送时间点为捕获输入动作时。

清零动作时缓存传送

在锯齿波计数模式或硬件计数模式时，正常的比较输出动作期间若有清零动作产生，通用周期基准值、通用比较基准值、等会根据相应的缓存动作设定状况发生一次缓存传送。

17.2.8 General PWM Output

17.2.8.1 PWM Spread Spectrum Output

为了降低 PWM 输出对外部的干扰，在 PWM 输出级有展频配置。每个 PWM 输出周期会微调 PWM 输出的相位。

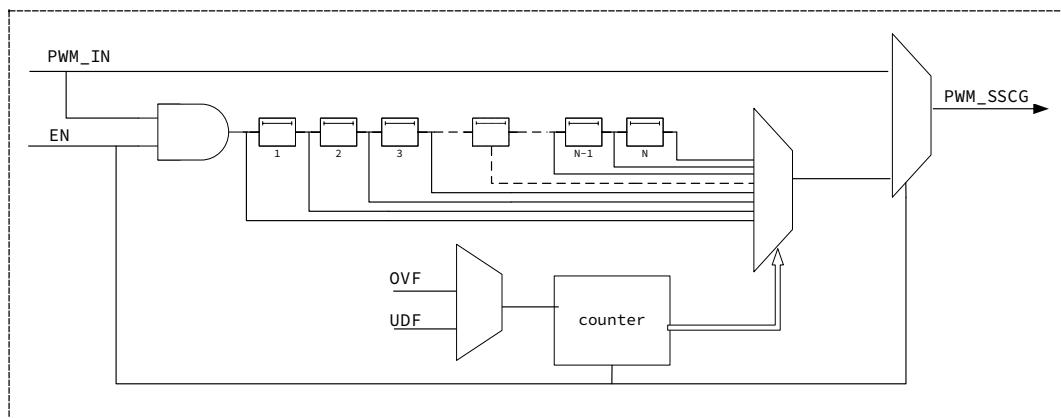


图 17-10 PWM 展频输出示意图

17.2.8.2 Independent PWM Output

每个定时器的 2 个端口 CHxA、CHxB 能独立的输出 PWM 波。如图 17-11，定时器 Timer6 的 CHA 端口输出 PWM 波。

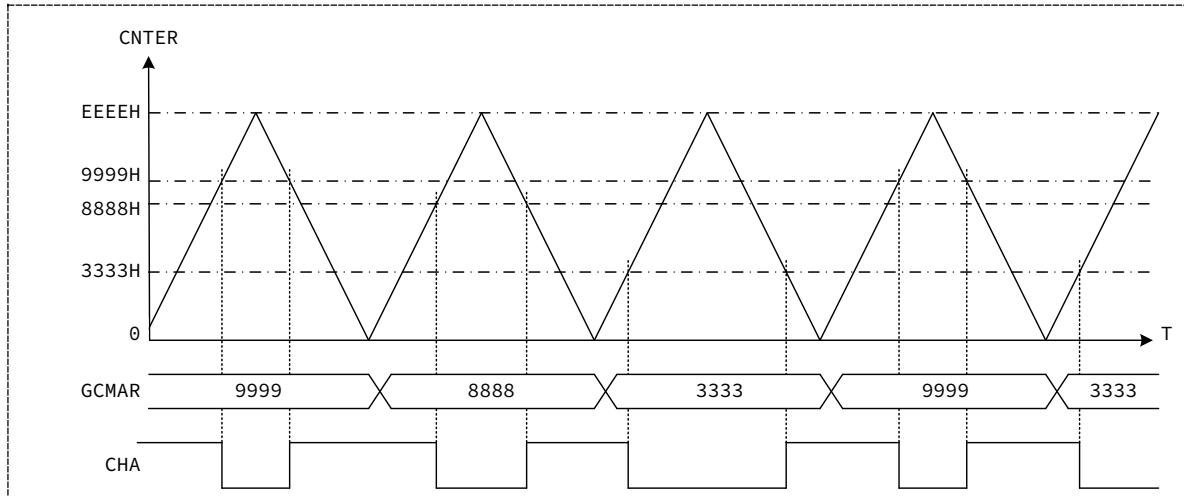


图 17-11 CHA 输出 PWM 波

17.2.8.3 Complementary PWM Output

CHxA 端口和 CHxB 端口，在不同的模式下可组合输出互补 PWM 波形。

软件设定 GCMBR 互补 PWM 输出

软件设定 GCMBR 互补 PWM 输出是指在锯齿波模式和三角波 A 模式、三角波 B 模式下，用于 CHxB 端口波形输出的通用比较基准值寄存器（GCMBR）的值由寄存器直接设定，与通用比较基准值寄存器（GCMAR）的值没有直接关系。

图 17-12 为软件设定 GCMBR 互补 PWM 波的输出例。

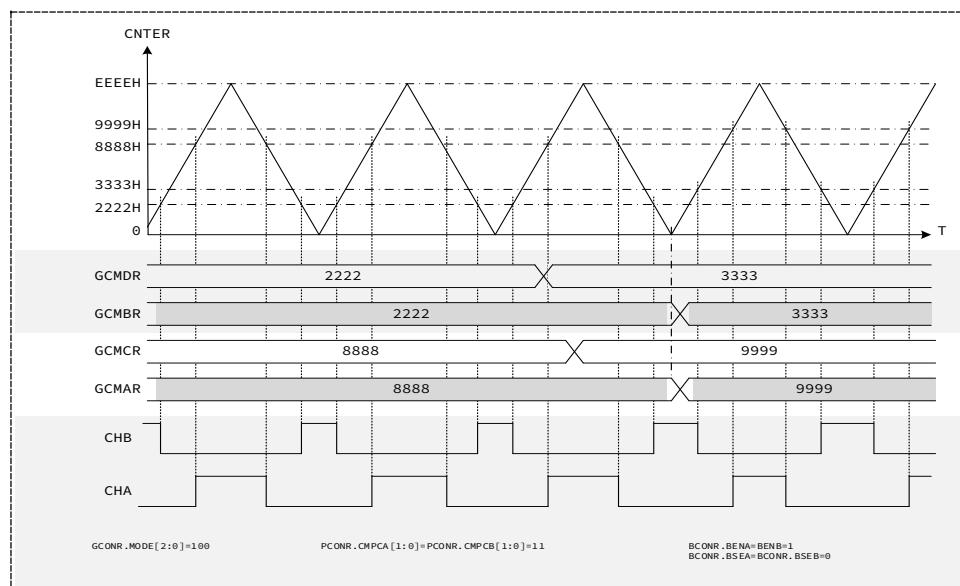


图 17-12 三角波 A 模式时软件设定 GCMBR 互补 PWM 波输出

硬件设定 GCMBR 互补 PWM 输出

硬件设定 GCMBR 互补 PWM 输出是指在三角波 A 模式、三角波 B 模式下，用于 CHxB 端口波形输出的通用比较基准值寄存器（GCMBR）的值由通用比较基准值寄存器（GCMAR）和死区时间基准值寄存器（DTUAR、DTDAR）的值运算决定。

图 17-13 为硬件设定 GCMBR 互补 PWM 波输出例。

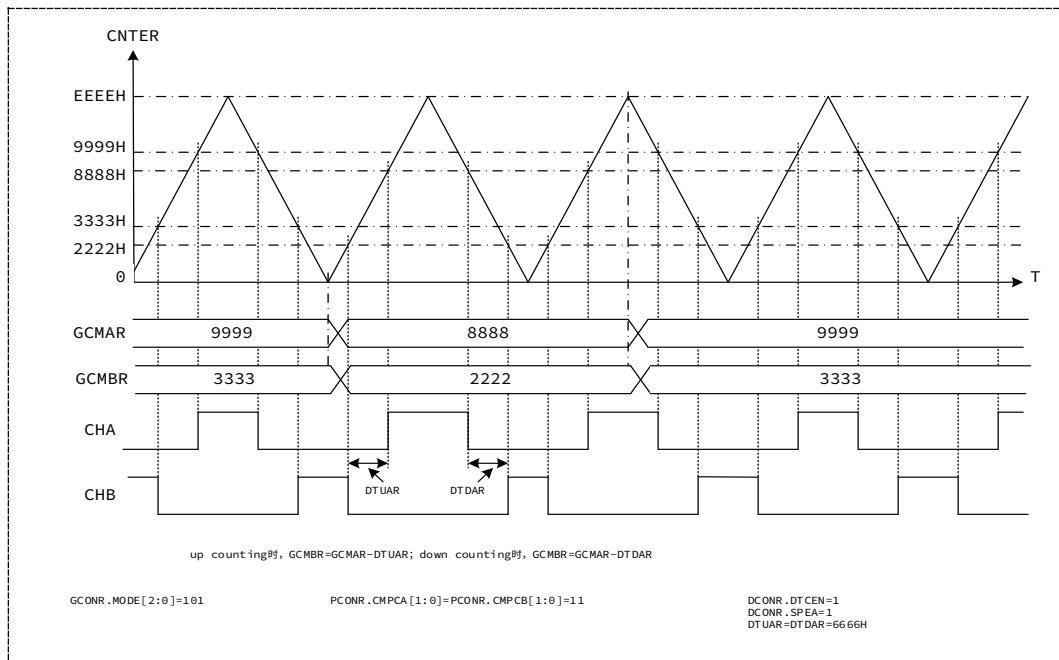


图 17-13 三角波 B 模式时硬件设定 GCMBR 互补 PWM 波输出（对称死区）

17.2.8.4 Multi-Phase PWM Output

每个定时器的 CHxA、CHxB 端口都能输出 2 相独立的 PWM 波或一组互补 PWM 波，多个定时器间组合，同时结合软件、硬件同步动作就可实现多相 PWM 波输出。如图 17-14，Timer4, Timer5, Timer6 组合输出 6 相 PWM 波；如图 17-15，Timer4, Timer5, Timer6 组合输出 3 相互补 PWM 波。

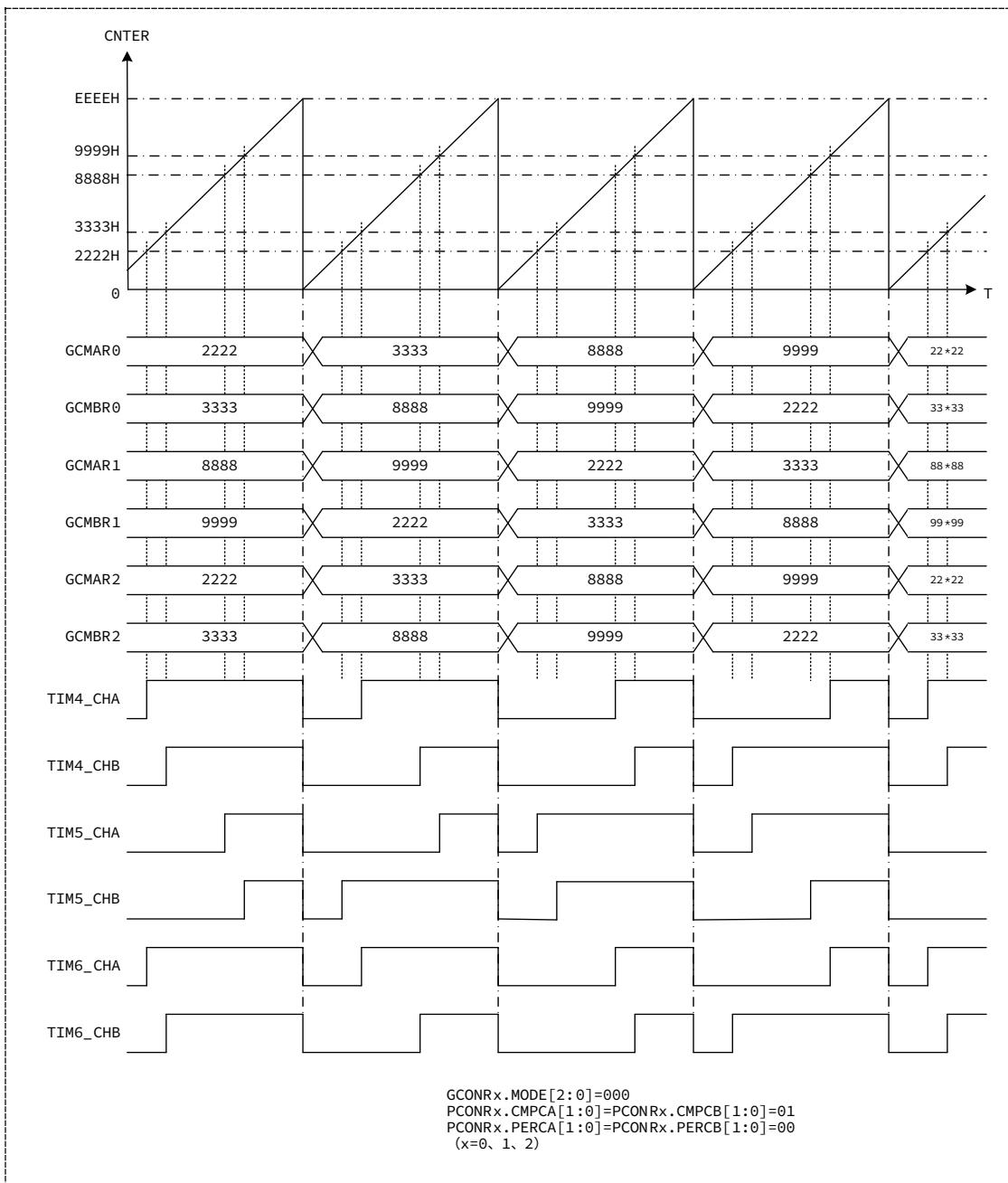


图 17-14 6 相 PWM 波

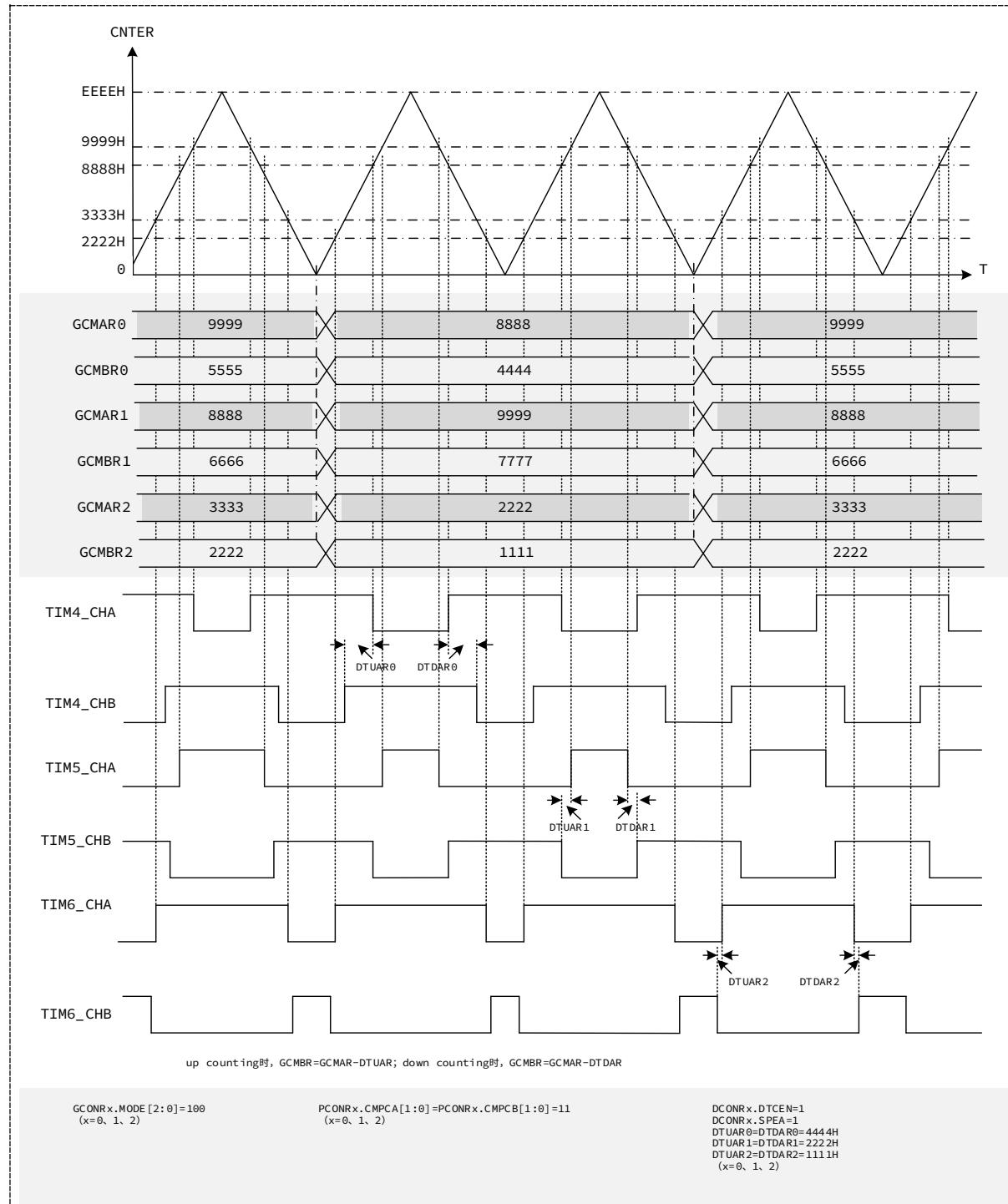


图 17-15 三角波 A 模式时带死区时间三相互补 PWM 波输出

17.2.9 Quadrature Encoding Count

将 CHxA 输入看作 AIN 输入、CHxB 输入看作 BIN 输入、TIMTRIA-D 中的任意一个输入看作 ZIN 输入，Advanced Timer 就可以实现三路输入的正交编码计数。

一个定时器的 AIN、BIN 单独动作可以实现位置计数模式；两个定时器的 AIN、BIN、ZIN 组合动作可以实现公转计数模式，一个定时器用于位置计数，一个定时器用于公转计数。

公转计数模式时，每两个定时器组合（定时器 4、5 组合，定时器 4 作为位置计数单元，定时器 5 作为公转计数单元）分别实现位置计数和公转计数。

AIN 和 BIN 的计数条件通过设定硬件递加事件选择寄存器（HCUPR）和硬件递减事件选择寄存器（HCDOR）中 CHxA 和 CHxB 的正交关系实现；ZIN 的输入动作通过设定位置单元的硬件清零事件选择寄存器（HCLRR）实现位置计数单元的位置计数器清零、通过设定公转单元的硬件递加事件选择寄存器（HCUPR）实现公转计数单元的公转计数器计数。

17.2.9.1 Position Counting Mode

正交编码位置模式，是指根据 AIN、BIN 的输入实现基本计数功能、相位差计数功能和方向计数功能。

基本计数

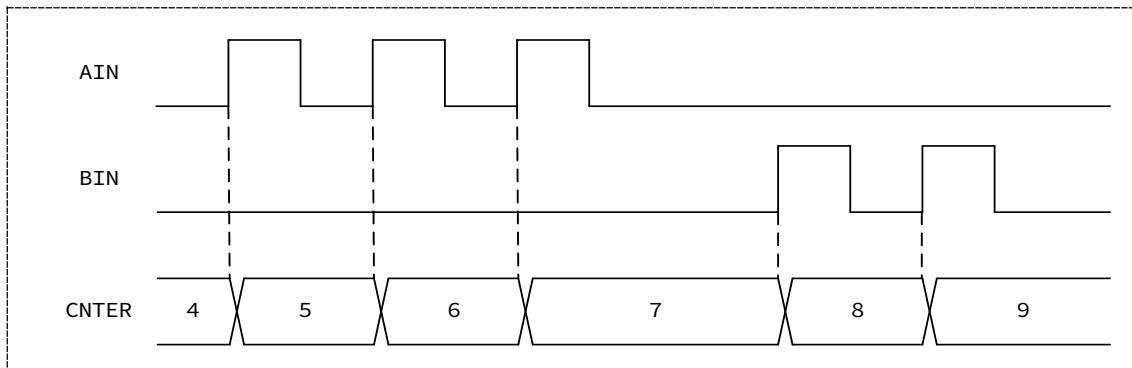


图 17-16 位置模式时基本计数动作

通过设定 HCUPR 与 HCDOR 寄存器，可以灵活的实现各种方式的相位差计数。

相位差计数

相位差计数是指根据 AIN 和 BIN 的相位关系进行计数。根据设定的不同，可以实现 1 倍计数、2 倍计数、4 倍计数等，如下图图 17-17~图 17-19 所示。

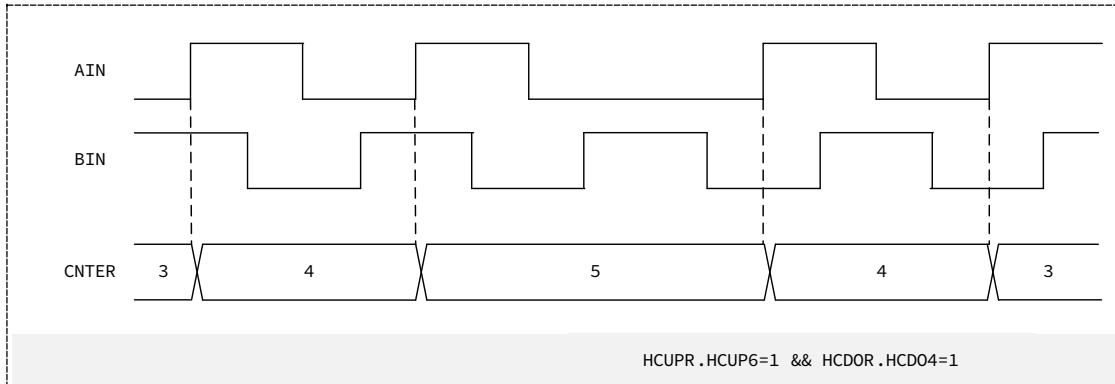


图 17-17 位置模式时相位差计数动作设定(1 倍)

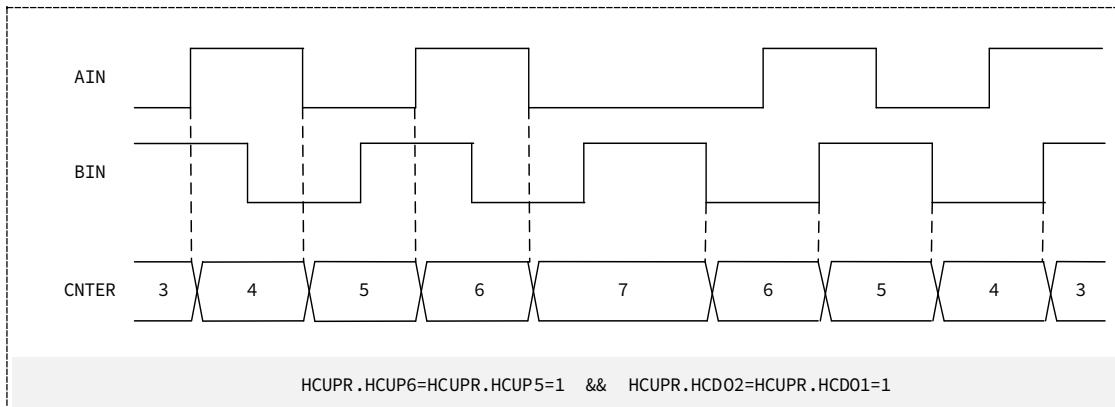


图 17-18 位置模式时相位差计数动作设定(2 倍)

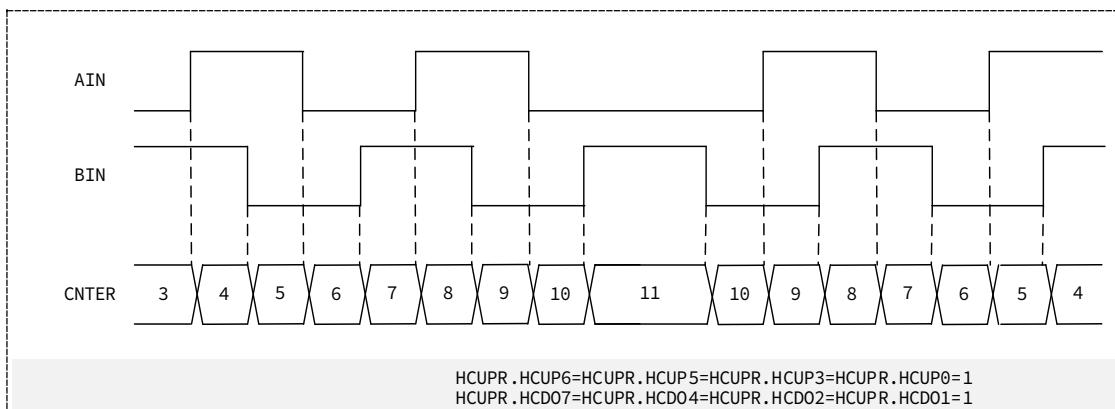


图 17-19 位置模式时相位差计数动作设定(4 倍)

方向计数

方向计数是指将 AIN 的输入状态设定为方向控制，将 BIN 的输入作为时钟计数，如图 17-20 所示。

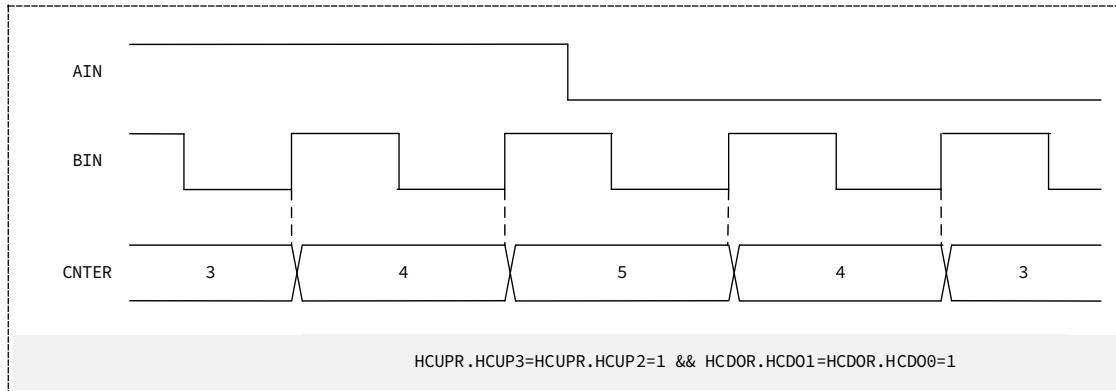


图 17-20 位置模式时方向计数动作

17.2.9.2 Revolution Mode

正交编码公转模式，是指在 AIN、BIN 计数的基础上，加入 ZIN 的输入事件以实现对公转圈数等的判断。公转模式时根据公转计数器的计数方式，可实现 Z 相计数功能、位置计数器输出计数功能和 Z 相计数与位置计数器输出混合计数功能。即使用两个 Advanced Timer 实现此功能。

Z 相计数

Z 相计数是指根据 ZIN 的输入，公转计数单元进行计数，同时将位置计数单元清零的计数动作。

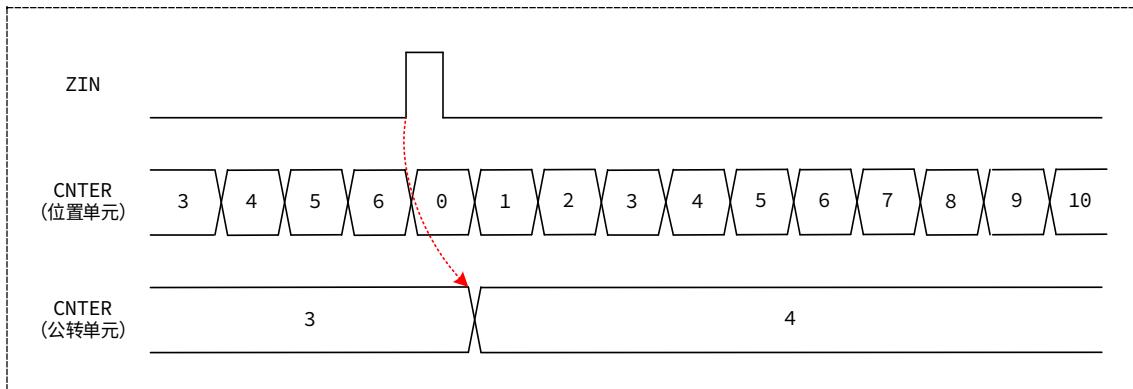


图 17-21 公转模式时 Z 相计数动作

位置溢出计数

位置溢出计数是指位置计数单元计数发生上溢或下溢时，产生一个溢出事件，从而触发公转计数单元的计数器进行一次计数（在该计数方式时 ZIN 的输入不进行公转计数单元的计数动作和位置计数单元的清零动作）。

位置计数单元的溢出事件通过 AOS 模块的联动选通实现公转计数单元计数，即可实现位置溢出计数。公转计数单元的硬件递加（递减）事件选择寄存器（HCUPR 或 HCDOR）的递加（递减）事件选择

Bit16:Bit19 中的 1 位，同时 AOS 模块设定对应的递加（递减）事件的事件源为位置计数单元的计数溢出事件，具体请参考 AOS 章节。如图 17-22 所示。

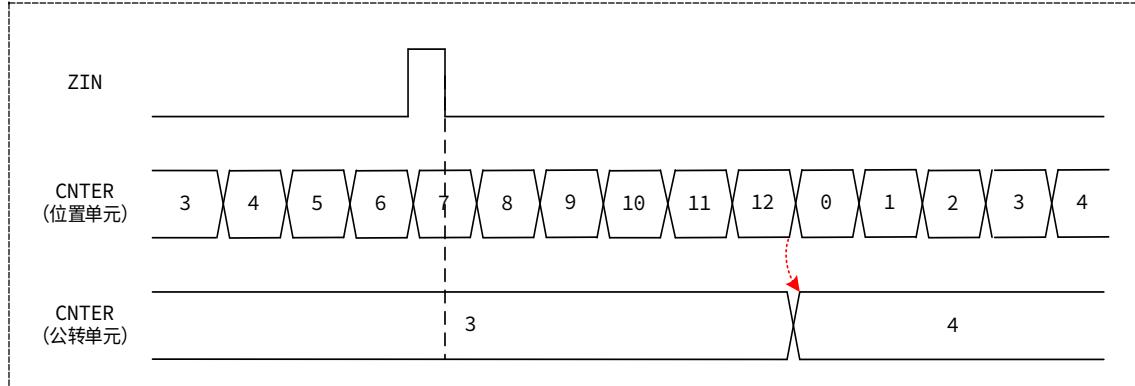


图 17-22 公转模式时位置计数器输出计数动作

混合计数

混合计数是指上述 Z 相计数和位置溢出计数两种计数方式合并起来的计数动作，其实现方式也是上述两种计数方式的组合。如图 17-23 所示。

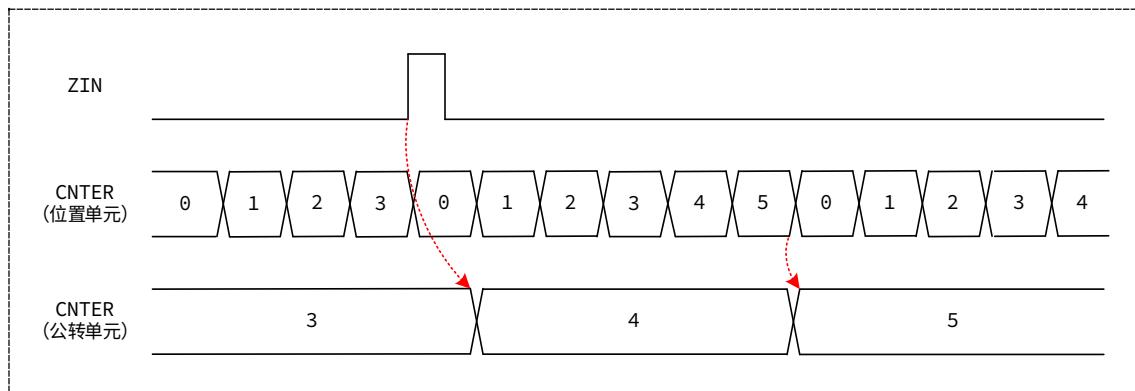


图 17-23 公转模式时 Z 相计数和位置计数器输出混合计数动作

Z 相动作屏蔽

在公转计数模式的 Z 相计数功能或混合计数功能时，可以设定在位置计数器的上溢点或下溢点后的几个周期内 (GCONR.ZMSK[0:1]设定)，将 ZIN 的有效输入屏蔽，不进行公转计数单元的计数和位置计数单元的清零。

位置计数单元的通用控制寄存器 (GCONR) 的 GCONR.ZMSKPOS 为 1 时，位置计数单元的 Z 相屏蔽功能使能，Z 相屏蔽的周期数由 GCONR.ZMSK 设定；公转计数单元的通用控制寄存器 (GCONR) 的 GCONR.ZMSKREV 为 1 时，公转计数单元的 Z 相屏蔽功能使能。

图 17-24 是公转计数模式混合计数时，在位置计数单元计数上溢后的 4 个计数周期内有 ZIN 相输入时，ZIN 相输入的动作无效，即公转计数单元不计数、位置计数单元不清零；之后再来的 ZIN 相输入正常动作。

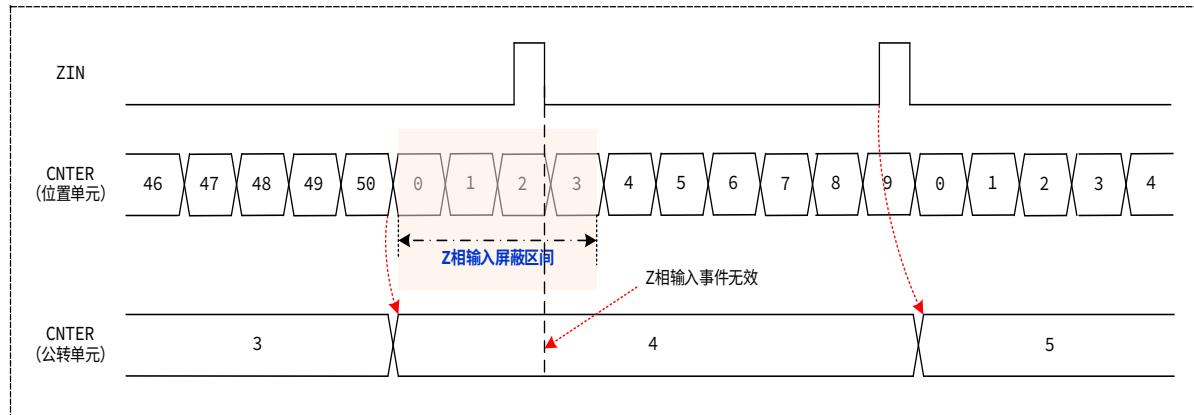


图 17-24 公转计数模式-混合计数 Z 相屏蔽动作例 1

图 17-25 是公转计数模式混合计数时,在位置计数单元计数上溢后的第3个周期,计数方向发生变化,此时设定的4个周期的屏蔽周期变为无效(实际ZIN相屏蔽功能维持了3个周期),开始向下计数。在位置计数单元发生计数下溢后,ZIN相屏蔽功能重新开启,维持4个周期后变为无效。在ZIN相屏蔽期间,ZIN相的输入功能无效,即公转计数单元不计数、位置计数单元不清零;之后再来的ZIN相输入正常动作。

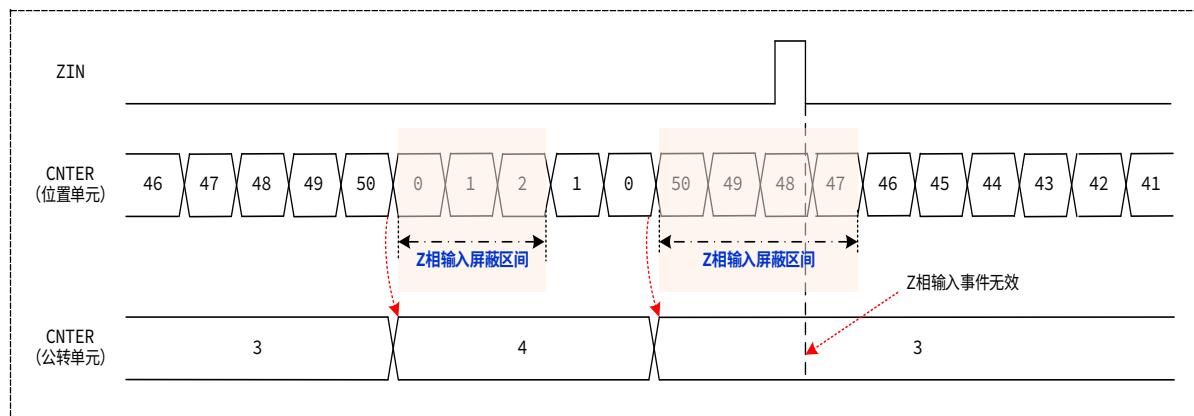


图 17-25 公转计数模式-混合计数 Z 相屏蔽动作例 2

17.2.10 Periodic Interval Response

Timer4/5/6 的通用比较基准值寄存器 (GCMAR~GCMDR), 在计数比较匹配时可分别产生专用有效请求信号, 送到 AOS 模块中用于和其它模块关联动作。

该请求信号可以每间隔几个周期后产生一次有效的请求信号。通过设定有效周期寄存器 (VPERR) 的 VPERR.PCNTS 位来指定每隔多少个周期请求信号有效一次, 其它周期内即使计数值和比较基准值寄存器 GCMAR 或 GCMBR 的值相等, 也不会输出有效的请求信号。

如果在使用周期间隔响应功能时停止并重新启动定时器, 请在停止定时器前配置 VPERR.PCNTE[1:0]=00, 否则在重新启动后第一次产生周期间隔有效请求信号的时刻可能出现偏差。

图 17-26 所示是周期间隔有效请求信号的动作例。

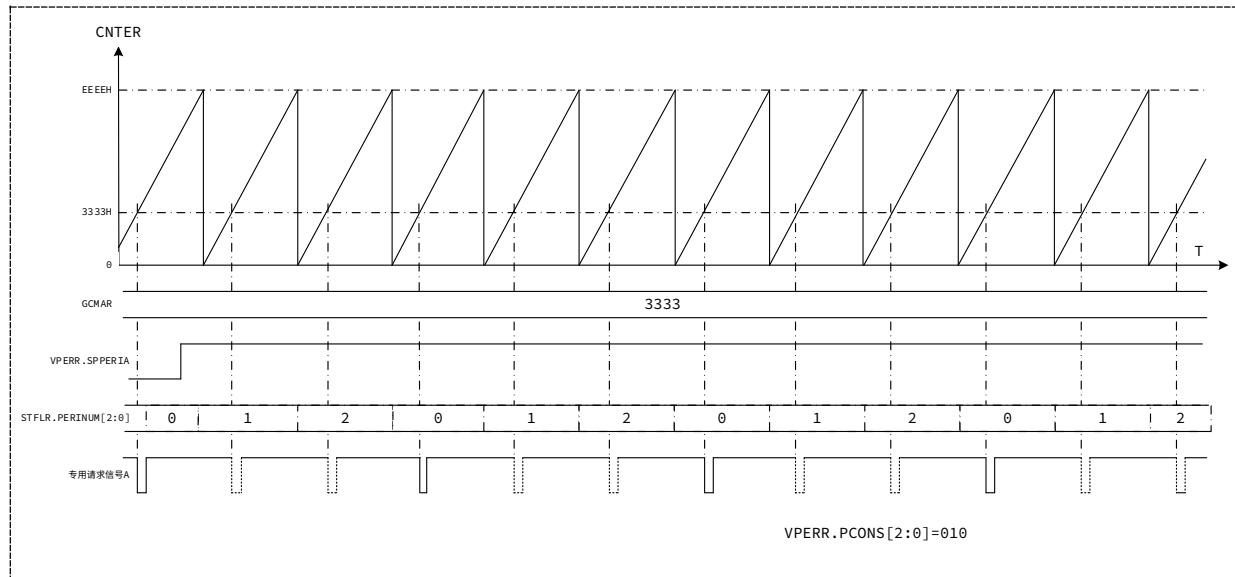


图 17-26 周期间隔有效请求信号动作

17.2.11 Protection Mechanism

Advanced Timer 可以对端口的输出状态进行保护控制。

Advanced Timer 有 4 个共用的端口输出无效事件接口，这 4 个接口连接刹车控制模块输出的 4 组刹车事件。每个接口上选通的异常状况事件可从刹车控制设定，当这些接口上监测到异常状况时，可以实现对通用 PWM 输出的控制。

端口在正常输出期间，若监测到从刹车控制过来的刹车事件，则端口的输出状态可变为预先设定好的状态。通用 PWM 输出端口在刹车控制异常事件发生时，端口状态可以变为输出高阻态、输出低电平或输出高电平（PCONR.DISVALA、PCONR.DISVALB 的设定决定）。

例如，若 PCONR.DISSELA[1:0]=01&PCONR.DISVALA=01 设定时，则在 CHxA 端口正常输出期间，若输出无效条件 1 上产生刹车事件，则 CHxA 端口上输出变为高阻态。

17.2.12 Interrupt Description

Timer4/5/6 各含有 3 类共计 9 个中断。分别是 4 个通用计数比较匹配中断（含 2 个捕获输入中断）、2 个计数周期匹配中断、1 个死区时间错误中断。

17.2.12.1 Count Compare Match Interrupt

通用比较基准值寄存器（GCMAR-GCMDR）共计 4 个，可分别与计数值比较产生比较匹配有效信号。计数比较匹配时，状态标志寄存器（STFLR）中的 STFLR.CMAF~STFLR.CMDF 位分别会被置为 1。此时若设定中断控制寄存器（ICONR）的 ICONR.INTENA~ICONR.INTEND 中相应位为 1 使能中断，则对应的中断请求也会被触发。

在硬件捕获事件选择寄存器 (HCPAR、HCPBR) 选择的捕获输入有效条件产生时，捕获输入动作发生。此时若设置中断控制寄存器 (ICONR) 的 ICONR.INTENA 或 ICONR.INTENB 位为 1 使能中断，则对应的中断请求被触发。

17.2.12.2 Count Period Match Interrupt

锯齿波递加计数至上溢点、锯齿波递减计数至下溢点、三角波计数至谷点或三角波计数至峰点时，状态标志寄存器 (STFLR) 的 STFLR.OVFF 或 STFLR.UDFF 位会被置为 1。此时若设置中断控制寄存器 (ICONR) 的 ICONR.INTENOVF 位与 ICONR.INTENUDF 位使能中断，则在对应的时间点可触发计数周期匹配中断。

17.2.12.3 Dead Time Error Interrupt

将死区时间基准值寄存器 (DTUAR、DTDAR) 的值加载到通用比较基准值寄存器 (GCMBR) 中时，若超过周期限制，则会产生死区时间错误，状态标志寄存器 (STFLR) 的 STFLR.DTEF 位会被置为 1。此时若设置中断控制寄存器 (ICONR) 的 ICONR.INTENDE 位使能中断，则会在该时刻触发死区时间错误中断。

17.2.13 DMA

Timer 支持软件和硬件触发 DMA 进行数据传输。支持数据从其他位置写入定时器，或从定时器读出写入其他位置。可应用于数据捕获后数据的自动搬运和更改周期值或占空比的脉冲宽度的自动调整。每个定时器有两个 DMA 请求，A 请求触发源可选择通用比较捕获 A、C、专用比较 A、计数上溢出，B 请求触发源可选择通用比较捕获 B、D、专用比较 B、计数下溢出。

IDREQ	Interrupt Signal of Peripheral
26	TIM4A
27	TIM4B
28	TIM5A
29	TIM5B
30	TIM6A
31	TIM6B

TIMxA 的 触发源可选择比较捕获 A、C，计数上溢出，专用比较 A

TIMxB 的 触发源可选择比较捕获 B、D，计数下溢出，专用比较 B

17.2.14 Brake protection

当可以设定无效条件 0~3，配置 PCONR.DISVALA, PCONR.DISVALB。无效条件有效时硬件自动将端口状态改变为预设状态（高电平，低电平，高阻态，保持正常输出）。

17.2.14.1 Port brake and software brake

端口经过极性选择控制，有效使能后，经过数字滤波，同步，产生端口刹车标志；端口刹车标志做为 Advanced Timer 的无效条件 3。端口刹车标志需要软件清除。

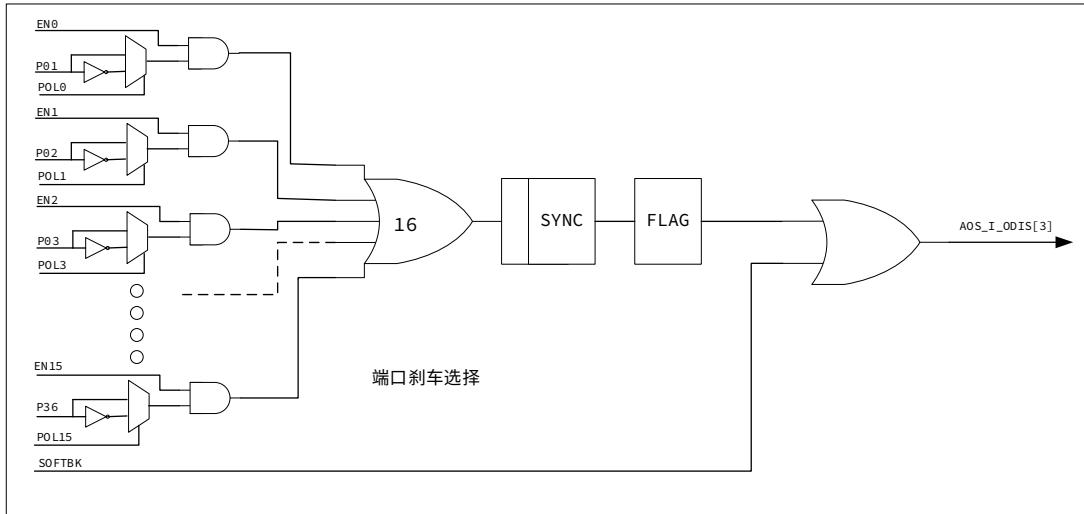


图 17-27 端口刹车与软件刹车示意图

17.2.14.2 Automatic braking in low power mode

系统进入低功耗模式，时钟停止后 PWM 将不能正常工作。低功耗模式作为 Advanced Timer 的无效条件 2 控制 PWM 刹车。

17.2.14.3 Braking when output levels are high or low

输出电平经过电平监测，有效使能后，经过同步，产生同高同低刹车标志；端口刹车标志做为 Advanced Timer 的无效条件 1。同高同低刹车标志需要软件清除。

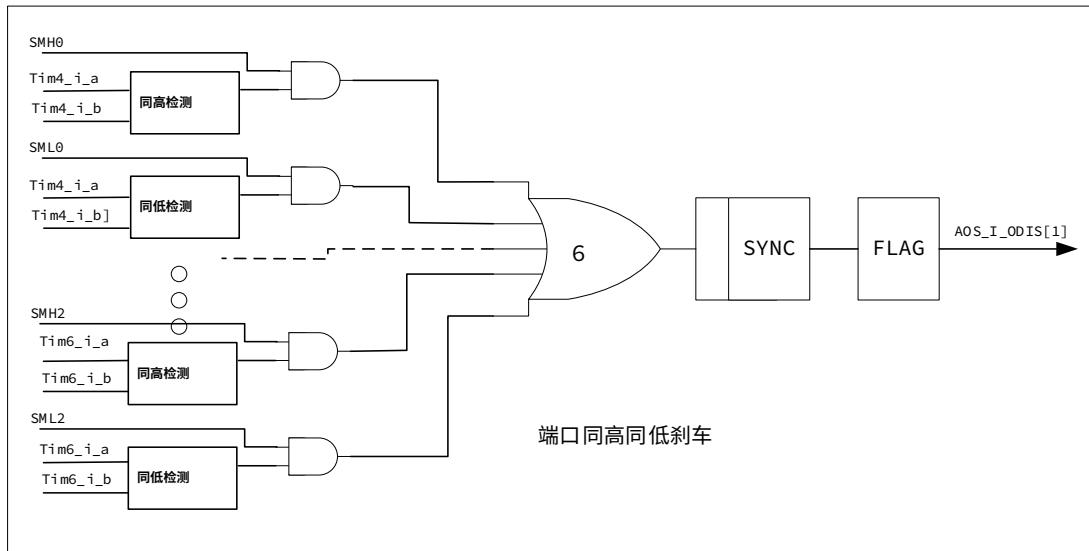


图 17-28 输出同高同低刹车示意图

17.2.14.4 VC Brake

VC0, VC1, VC2 中断标志经过使能后作为 Advanced Timer 的无效条件 0。

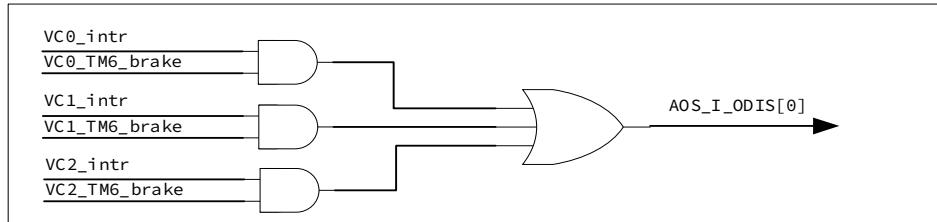


图 17-29 VC 刹车控制示意图

17.2.15 Internal Interconnection

17.2.15.1 Interrupt Trigger Output

由于 Timer4/5/6 的一个中断包含多个中断源。控制触发 ADC 与控制 AOS 的中断信号有单独控制可以选择不同的源，可以选择上溢出、下溢出，4 个比较匹配共 6 个 TIMx 的中断源的任意中断源作为触发条件。

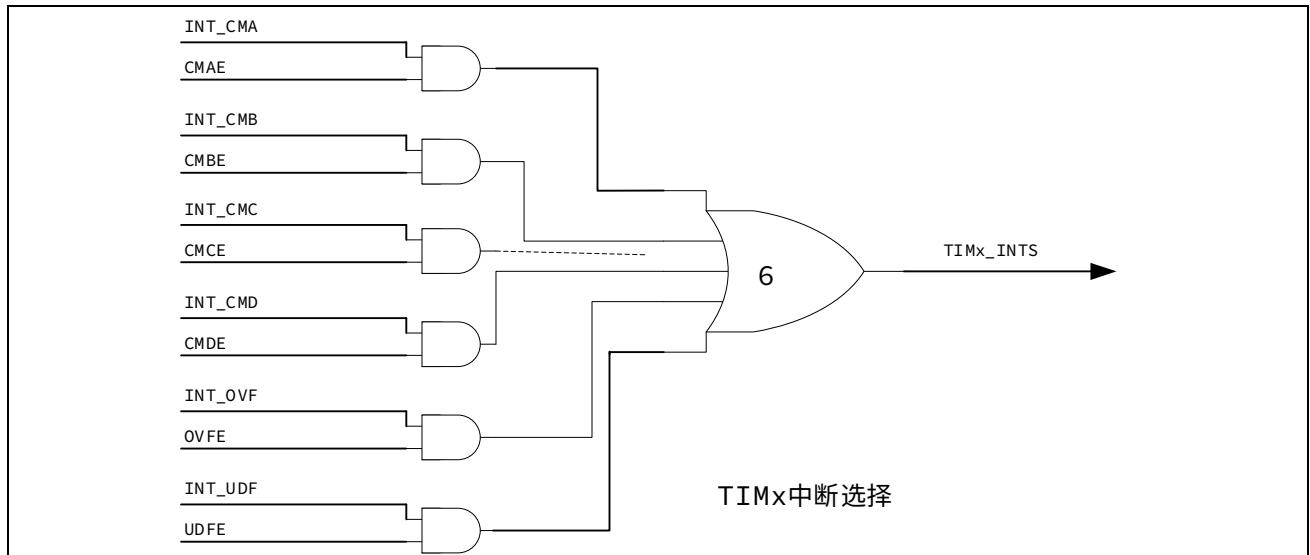


图 17-30 Timer4/5/6 中断选择

17.2.15.2 AOS Trigger

AOS 是系统的内部信号，通过选择控制后可以触发 Advanced Timer 的计数器的开始，停止，清零，加1，减1等功能。Advanced Timer 有4路 AOS 触发，每路触发可以选择不同模块的中断源。选择后的信号产生单脉冲触发输入到 Advanced Timer，控制 Advanced Timer 的计数器的开始，停止，清零。

Timer4/5/6 内部使用寄存器选择不同的 AOS_I_TRIGGER 作为自己的触发信号。如可使用 HSTAR 寄存器可以使用中断触发相应定时器的硬件启动。

	AOS_i_trig0	AOS_i_trig1	AOS_i_trig2	AOS_i_trig3
选择控制信号	ITRIG.IAOS0S	ITRIG.IAOS1S	ITRIG.IAOS2S	ITRIG.IAOS3S
0000	TIM0_INT	TIM0_INT	TIM0_INT	TIM0_INT
0001	TIM1_INT	TIM1_INT	TIM1_INT	TIM1_INT
0010	TIM2_INT	TIM2_INT	TIM2_INT	TIM2_INT
0011	LPTIMER_INT	LPTIMER_INT	LPTIMER_INT	LPTIMER_INT
0100	TIM4_INTS	TIM4_INTS	TIM4_INTS	TIM4_INTS
0101	TIM5_INTS	TIM5_INTS	TIM5_INTS	TIM5_INTS
0110	TIM6_INTS	TIM6_INTS	TIM6_INTS	TIM6_INTS
0111	UART0_INT	UART0_INT	UART0_INT	UART0_INT
1000	UART1_INT	UART1_INT	UART1_INT	UART1_INT
1001	LPUART0_INT	LPUART0_INT	LPUART0_INT	LPUART0_INT
1010	VC0_INT	VC0_INT	VC0_INT	VC0_INT
1011	VC1_INT	VC1_INT	VC1_INT	VC1_INT
1100	RTC_INT	RTC_INT	RTC_INT	RTC_INT
1101	PCA_INT	PCA_INT	PCA_INT	PCA_INT
1110	SPI_INT	SPI_INT	SPI_INT	SPI_INT
1111	ADC_INT	ADC_INT	ADC_INT	ADC_INT

表 17-3 AOS 源选择

17.2.15.3 Port trigger TRIGA-TRIGD

端口触发可以控制 Advanced Timer 的硬件启动、停止、清零、捕获、计数器加减计数等功能，有数字滤波功能可选，端口可以配置为芯片的任意一个端口上。

选择控制信号分别独立控制	TRIGA	TRIGB	TRIGC	TRIGD
0000	PA3	PA3	PA3	PA3
0001	PB3	PB3	PB3	PB3
0010	PC3	PC3	PC3	PC3
0011	PD3	PD3	PD3	PD3
0100	PA7	PA7	PA7	PA7
0101	PB7	PB7	PB7	PB7
0110	PC7	PC7	PC7	PC7
0111	PD7	PD7	PD7	PD7
1000	PA11	PA11	PA11	PA11
1001	PB11	PB11	PB11	PB11
1010	PC11	PC11	PC11	PC11
1011	PD1	PD1	PD1	PD1
1100	PA15	PA15	PA15	PA15
1101	PB15	PB15	PB15	PB15
1110	PC5	PC5	PC5	PC5
1111	PD5	PD5	PD5	PD5

表 17-4 端口触发选择

17.2.15.4 VC Output to Advanced Timer Interconnection

VC 内部可以互连到 Advanced Timer 的捕获输入端，可以对 VC 输出的边沿进行捕获；VC0 与 CHA 连接，VC1 与 CHB 连接；控制在 VC 控制寄存器。

17.3 寄存器描述

CH0 基地址 0x40003000

CH1 基地址 0x40003400

CH2 基地址 0x40003800

表 17-5 Advanced Timer 寄存器列表

寄存器	偏移地址	描述
TIMx_CNTER	0x000	通用计数基准值寄存器
TIMx_PERAR	0x004	通用周期基准值寄存器
TIMx_PERBR	0x008	通用周期基准值缓存寄存器
TIMx_GCMAR	0x010	通用比较 A 基准值寄存器
TIMx_GCMBR	0x014	通用比较 B 基准值寄存器
TIMx_GCMCR	0x018	通用比较 C 基准值寄存器
TIMx_GCMDR	0x01C	通用比较 D 基准值寄存器
TIMx_SCMAR	0x028	专用比较 A 基准值寄存器
TIMx_SCMBR	0x02C	专用比较 B 基准值寄存器
TIMx_DTUAR	0x040	死区时间基准值寄存器
TIMx_DTDAR	0x044	死区时间基准值寄存器
TIMx_GCONR	0x050	通用控制寄存器
TIMx_ICONR	0x054	中断控制寄存器
TIMx_PCONR	0x058	端口控制寄存器
TIMx_BCONR	0x05C	缓存控制寄存器
TIMx_DCONR	0x060	死区控制寄存器
TIMx_FCONR	0x068	滤波控制寄存器
TIMx_VPERR	0x06C	有效周期寄存器
TIMx_STFLR	0x070	状态标志寄存器
TIMx_HSTAR	0x074	硬件启动事件选择寄存器
TIMx_HSTPR	0x078	硬件停止事件选择寄存器
TIMx_HCELR	0x07C	硬件清零事件选择寄存器
TIMx_HCPAR	0x080	硬件捕获事件选择寄存器
TIMx_HCPBR	0x084	硬件捕获事件选择寄存器
TIMx_HCUPR	0x088	硬件递减事件选择寄存器
TIMx_HCDOR	0x08C	硬件递减事件选择寄存器
TIMx_IFR	0x100	中断标志寄存器
TIMx_ICLR	0x104	中断清除寄存器
TIMx_CR	0x108	展频及中断触发选择寄存器
TIMx_AOSSR	0x110	AOS 选择寄存器, 三个通道共用

寄存器	偏移地址	描述
TIMx_AOSCL	0x114	AOS 刹车标志清除寄存器, 三个通道共用
TIMx_PTAKS	0x118	端口刹车控制寄存器, 三个通道共用
TIMx_TTRIG	0x11C	端口触发控制寄存器, 三个通道共用
TIMx_ITRIG	0x120	AOS 触发控制寄存器, 三个通道共用
TIMx_PTAKP	0x124	端口刹车极性控制寄存器, 三个通道共用
TIMx_SSTAR	0x3F4	软件同步启动寄存器
TIMx_SSTPR	0x3F8	软件同步停止寄存器
TIMx_SCLRR	0x3FC	软件同步清零寄存器

17.3.1 Common Count Reference Register (TIMx_CNTER)

地址偏移量: 0x000

复位值: 0x0000 0000

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

位	符号	功能描述
31:16	Reserved	-
15:0	CNT[15:0]	当前计数器的计数值

17.3.2 Universal Period Reference Value Register (TIMx_PERAR)

地址偏移量: 0x004

复位值: 0x0000 FFFF

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

位	符号	功能描述
31:16	Reserved	-
15:0	PERA[15:0]	计数周期值，设定每轮计数的计数周期值

17.3.3 General Period Buffer Register (TIMx_PERBR)

地址偏移量: 0x008

复位值: 0x0000 FFFF

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

位	符号	功能描述
31:16	Reserved	-
15:0	PERB[15:0]	缓存计数周期值，计数周期的缓存值

17.3.4 General Comparison Reference Value Register (TIMx_GCMAR-GCMDR)

地址偏移量: 0x0010, 0x0014, 0x0018, 0x001C

复位值: 0x0000 FFFF

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

位	符号	功能描述
31:16	Reserved	-
15:0	GCMA-D [15:0]	计数比较基准值，比较基准值设定，与计数值相等时匹配信号有效

17.3.5 Dedicated Comparison Reference Value Registers (TIMx_SCMAR-SCMBR)

地址偏移量: 0x0028, 0x002C

复位值: 0x0000 FFFF

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

位	符号	功能描述
31:16	Reserved	-
15:0	SCMA-B [15:0]	计数比较基准值，比较基准值设定，与计数值相等时匹配信号有效

17.3.6 Dead Time Reference Value Register (TIMx_DTUAR-DTDAR)

地址偏移量: 0x040, 0x044

复位值: 0x0000 FFFF

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

位	符号	功能描述
31:16	Reserved	-
15:0	DTUA/DA [15:0]	死区时间值，死区时间设定值

17.3.7 General Control Register (TIMx_GCONR)

地址偏移量: 0x050

复位值: 0x00000100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										ZMSK[1:0]	ZMSK POS	ZMSK REV			
										RW	RW	RW			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										DIR	Res.	CKDIV[2:0]	MODE[2:0]	START	
										RW		RW		RW	

位	标记	功能描述
31:20	Reserved	-
19:18	ZMSK[1:0]	Z 相输入屏蔽周期数 正交编码 Z 相输入被屏蔽的计数周期值 00: Z 相输入屏蔽功能无效 01: 位置计数上溢后或下溢后的 4 个计数周期内的 Z 相输入被屏蔽 10: 位置计数上溢后或下溢后的 8 个计数周期内的 Z 相输入被屏蔽 11: 位置计数上溢后或下溢后的 16 个计数周期内的 Z 相输入被屏蔽
17	ZMSKPOS	Z 相输入位置计数器选择 0: Z 相输入时该定时器作为位置计数器, 在屏蔽周期期间内位置计数器清零功能正常动作 1: Z 相输入时该定时器作为位置计数器, 在屏蔽周期期间内位置计数器清零功能被屏蔽
16	ZMSKREV	Z 相输入公转计数器选择 0: Z 相输入时该定时器作为公转计数器, 在屏蔽周期期间内公转计数器计数功能正常动作 1: Z 相输入时该定时器作为公转计数器, 在屏蔽周期期间内公转计数器计数功能被屏蔽
15:9	Reserved	-
8	DIR	计数方向 0: 递减计数; 1: 递加计数
7	Reserved	-
6:4	CKDIV[2:0]	计数时钟选择 000: PCLK0 001: PCLK0/2 010: PCLK0/4 011: PCLK0/8 100: PCLK0/16 101: PCLK0/64 110: PCLK0/256 111: PCLK0/1024
3:1	MODE[2:0]	计数模式 000: 锯齿波 A 模式 100: 三角波 A 模式 101: 三角波 B 模式 请不要设定其它值
0	START	计数器启动 0: 计数器关闭; 1: 计数器启动 注: 该位在软件停止条件或硬件停止条件有效时, 会自动变为 0

17.3.8 Interrupt Control Register (TIMx_ICONR)

地址偏移量: 0x054

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														INTEN SBD	INTEN SBU
RW														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTEN SAMH	INTEN SAML	Reserved			INTEN DE	INTEN UDF	INTEN OVF	Reserved	INTEN D	INTEN C	INTEN B	INTEN A			
RW	RW				RW	RW	RW		RW	RW	RW	RW			

位	标记	功能
31:20	Reserved	-
19	INTENSBD	专用向下计数触发 ADC 使能 B
18	INTENSBU	专用向上计数触发 ADC 使能 B
17	INTENSAD	专用向下计数触发 ADC 使能 A
16	INTENSAU	专用向上计数触发 ADC 使能 A
15	INTENSAMH	同高中断使能
14	INTENSAML	同低中断使能
13:9	Reserved	-
8	INTENDE	死区时间错误中断使能 0: 死区时间错误时, 该中断无效 1: 死区时间错误时, 该中断使能
7	INTENUDF	下溢中断使能 0: 锯齿波时下溢发生或三角波时计数到谷点, 该中断无效 1: 锯齿波时下溢发生或三角波时计数到谷点, 该中断使能
6	INTENOVF	上溢中断使能 0: 锯齿波时上溢发生或三角波时计数到峰点, 该中断无效 1: 锯齿波时上溢发生或三角波时计数到峰点, 该中断使能
5:4	Reserved	-
3	INTEND	计数匹配中断使能 D 0: GCMDR 寄存器与计数值相等时, 该中断无效 1: GCMDR 寄存器与计数值相等时, 该中断使能
2	INTENC	计数匹配中断使能 C 0: GCMCR 寄存器与计数值相等时, 该中断无效 1: GCMCR 寄存器与计数值相等时, 该中断使能
1	INTENB	计数匹配中断使能 B 0: GCMBR 寄存器与计数值相等时, 或者发生捕获输入事件时, 该中断无效 1: GCMBR 寄存器与计数值相等时, 或者发生捕获输入事件时, 该中断使能
0	INTENA	计数匹配中断使能 A

		0: GCMAR 寄存器与计数值相等时，或者发生捕获输入事件时，该中断无效 1: GCMAR 寄存器与计数值相等时，或者发生捕获输入事件时，该中断使能
--	--	--

17.3.9 Port Control Register (TIMx_PCONR)

地址偏移量: 0x058

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DISVALB	DISSELB	OUTENB	PERCB	CMPCB	STASTPSB	STPCB	STACB	CAPCB				
	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DISVALA	DISSELLA	OUTENA	PERCA	CMPCA	STASTPSA	STPCA	STACA	CAPCA				
	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能
31:29	Reserved	-
28:27	DISVALB	CHxB 输出状态控制 00: 强制输出无效条件 0~3 中被选择的条件成立时，CHxB 端口正常输出 01: 强制输出无效条件 0~3 中被选择的条件成立时，CHxB 端口输出高阻态 10: 强制输出无效条件 0~3 中被选择的条件成立时，CHxB 端口输出低电平 11: 强制输出无效条件 0~3 中被选择的条件成立时，CHxB 端口输出高电平
26:25	DISSELB	强制输出无效条件选择 B 00: 选择强制输出无效条件 0; 01: 选择强制输出无效条件 1 10: 选择强制输出无效条件 2; 11: 选择强制输出无效条件 3
24	OUTENB	输出使能 B 0: Advanced Timer 功能时的 CHxB 端口输出无效 1: Advanced Timer 功能时的 CHxB 端口输出有效
23:22	PERCB	周期值匹配时端口状态设定 B 00: 计数器计数值与周期值相等时，CHxB 端口输出保持为低电平 01: 计数器计数值与周期值相等时，CHxB 端口输出设定为高电平 10: 计数器计数值与周期值相等时，CHxB 端口输出设定为先前状态 11: 计数器计数值与周期值相等时，CHxB 端口输出设定为反转电平
21:20	CMPCB	比较值匹配时端口状态设定 B 00: 计数器计数值与 GCMBR 相等时，CHxB 端口输出保持为低电平 01: 计数器计数值与 GCMBR 相等时，CHxB 端口输出设定为高电平 10: 计数器计数值与 GCMBR 相等时，CHxB 端口输出设定为先前状态 11: 计数器计数值与 GCMBR 相等时，CHxB 端口输出设定为反转电平
19	STASTPSB	计数开始停止端口状态选择 B 0: 计数开始或停止时，CHxB 端口输出由 STACB、STPCB 决定 1: 计数开始或停止时，CHxB 端口输出设定为先前状态 注：此处的计数开始是指初始计数开始或停止再开始；计数停止是指初始时停止或计数开始

		后再停止
18	STPCB	计数停止端口状态设定 B 0：计数停止时，CHxB 端口输出设定为低电平 1：计数停止时，CHxB 端口输出设定为高电平
17	STACB	计数开始端口状态设定 B 0：计数开始时，CHxB 端口输出设定为低电平 1：计数开始时，CHxB 端口输出设定为高电平
16	CAPCB	功能模式选择 B 0：比较输出功能；1：捕获输入功能
15:13	Reserved	-
12:11	DISVALA	CHxA 输出状态控制 00：强制输出无效条件 0~3 中被选择的条件成立时，CHxA 端口正常输出 01：强制输出无效条件 0~3 中被选择的条件成立时，CHxA 端口输出高阻态 10：强制输出无效条件 0~3 中被选择的条件成立时，CHxA 端口输出低电平 11：强制输出无效条件 0~3 中被选择的条件成立时，CHxA 端口输出高电平
10:9	DISSELA	强制输出无效条件选择 A 00：选择强制输出无效条件 0；01：选择强制输出无效条件 1 10：选择强制输出无效条件 2；11：选择强制输出无效条件 3
8	OUTENA	输出使能 A 0：Advanced Timer 功能时的 CHxA 端口输出无效 1：Advanced Timer 功能时的 CHxA 端口输出有效
7:6	PERCA	周期值匹配时端口状态设定 A 00：计数器计数值与周期值相等时，CHxA 端口输出保持为低电平 01：计数器计数值与周期值相等时，CHxA 端口输出设定为高电平 10：计数器计数值与周期值相等时，CHxA 端口输出设定为先前状态 11：计数器计数值与周期值相等时，CHxA 端口输出设定为反转电平
5:4	CMPCA	比较值匹配时端口状态设定 A 00：计数器计数值与 GCMAR 相等时，CHxA 端口输出保持为低电平 01：计数器计数值与 GCMAR 相等时，CHxA 端口输出设定为高电平 10：计数器计数值与 GCMAR 相等时，CHxA 端口输出设定为先前状态 11：计数器计数值与 GCMAR 相等时，CHxA 端口输出设定为反转电平
3	STASTPSA	计数开始停止端口状态选择 A 0：计数开始或停止时，CHxA 端口输出由 STACA、STPCA 决定 1：计数开始或停止时，CHxA 端口输出设定为先前状态 注：此处的计数开始是指初始计数开始或停止再开始；计数停止是指初始时停止或计数开始后再停止
2	STPCA	计数停止端口状态设定 A 0：计数停止时，CHxA 端口输出设定为低电平； 1：计数停止时，CHxA 端口输出设定为高电平
1	STACA	计数开始端口状态设定 A

		0: 计数开始时, CHxA 端口输出设定为低电平 1: 计数开始时, CHxA 端口输出设定为高电平
0	CAPCA	功能模式选择 A 0: 比较输出功能; 1: 捕获输入功能

17.3.10 Cache Control Register (TIMx_BCONR)

地址偏移量: 0x05C

复位值: 0x0000 0000

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

位	标记	功能
31:9	Reserved	-
8	BENP	周期值缓存传送 0: 缓存传送无效 1: 缓存传送使能 (PERBR->PERAR)
7:3	Reserved	-
2	BENB	通用比较值缓存传送 B 0: 缓存传送无效 1: 缓存传送使能 比较输出功能时: (GCMDR->GCMBR); 捕获输入功能时: (GCMBR->GCMDR)
1	Reserved	-
0	BENA	通用比较值缓存传送 A 0: 缓存传送无效 1: 缓存传送使能 比较输出功能时: (GCMCR->GCMAR); 捕获输入功能时: (GCMAR->GCMCR)

17.3.11 Dead Time Control Register (TIMx_DCONR)

地址偏移量: 0x060

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							SEPA	Reserved							DTCEN
							RW								RW

位	标记	功能
31:9	Reserved	-
8	SEPA	分离设定 0: DTUAR 和 DTDAR 分别设定 1: DTDAR 的值和 DTUAR 的值自动相等
7:1	Reserved	-
0	DTCEN	死区功能 0: 死区功能无效 1: 死区功能有效

17.3.12 Filter Control Register (TIMx_FCONR)

地址偏移量: 0x068

复位值: 0x0000 0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.		NOFI CKTD	NOFI ENTD	Res.	NOFI CKTC	NOFI ENTC	Res.	NOFICKTB	NOFIENTB	Res.	NOFICKTA	NOFIENTA				
	RW	RW		RW	RW		RW	RW		RW	RW		RW	RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
									NOFICKGB	NOFIENGB	Res.	NOFICKGA	NOFIENGA			
									RW	RW		RW	RW			

位	标记	功能
31	Reserved	-
30:29	NOFICKTD	TRID 端口滤波采样基准时钟选择 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
28	NOFIENTD	TRID 端口捕获输入滤波使能, 0 无效; 1 使能
27	Reserved	-
26:25	NOFICKTC	TRIC 端口滤波采样基准时钟选择 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
24	NOFIENTC	TRIC 端口捕获输入滤波使能, 0 无效; 1 使能
23	Reserved	-
22:21	NOFICKTB	TRIB 端口滤波采样基准时钟选择 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
20	NOFIENTB	TRIB 端口捕获输入滤波使能, 0 无效; 1 使能
19	Reserved	-
18:17	NOFICKTA	TRIA 端口滤波采样基准时钟选择 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
16	NOFIENTA	TRIA 端口捕获输入滤波使能, 0 无效; 1 使能
15:7	Reserved	-
6:5	NOFICKGB	CHxIB 端口滤波采样基准时钟选择 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
4	NOFIENGB	CHxIB 端口捕获输入滤波使能, 0 无效; 1 使能
3	Reserved	-
2:1	NOFICKGA	CHxIA 端口滤波采样基准时钟选择 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
0	NOFIENGA	CHxIA 端口捕获输入滤波使能, 0 无效; 1 使能

注意:

- TRIGA-D 滤波设置只有在 TIM4 中设置有效, 在 Timer5/6 设置无效。

17.3.13 Valid Period Register (TIMx_VPERR)

地址偏移量: 0x06C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												PCNTS	PCNTE		
												RW	RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												GEPE RID	GEPER IC	GEPE RIB	GEPE RIA
												RW	RW	RW	RW

位	标记	功能
31:21	Reserved	-
20:18	PCNTS	有效周期选择 000: 有效周期选择功能无效 001: 每隔 1 个周期有效一次 010: 每隔 2 个周期有效一次 011: 每隔 3 个周期有效一次 100: 每隔 4 个周期有效一次 101: 每隔 5 个周期有效一次 110: 每隔 6 个周期有效一次 111: 每隔 7 个周期有效一次
17:16	PCNTE	有效周期计数条件选择 00: 有效周期选择功能无效 01: 锯齿波计数上、下溢点或三角波波谷做为计数条件 10: 锯齿波计数上、下溢点或三角波波峰做为计数条件 11: 锯齿波计数上、下溢点或三角波波谷、波峰做为计数条件
15:4	Reserved	-
3	GEPERID	通用信号有效周期选择 D 0: 有效周期选择功能无效; 1: 有效周期选择功能使能
2	GEPERIC	通用信号有效周期选择 C 0: 有效周期选择功能无效; 1: 有效周期选择功能使能
1	GEPERIB	通用信号有效周期选择 B 0: 有效周期选择功能无效; 1: 有效周期选择功能使能
0	GEPERIA	通用信号有效周期选择 A 0: 有效周期选择功能无效; 1: 有效周期选择功能使能

17.3.14 Status Flag Register (TIMx_STFLR)

地址偏移量: 0x070

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
DIRF	Reserved								VPERNUM	Reserved							
R									R								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reserved		CMSBDF	CM SBU F	CMS ADF	CMS AUF	DTE F	UDF F	OVF F	Reserved		CMD F	CMC F	CMB F	CMA F			
		RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW			

位	标记	功能
31	DIRF	计数方向 0: 递减计数 1: 递加计数
30:24	Reserved	-
23:21	VPERNUM	周期次数 有效周期选择功能使能时, 计数后的周期次数
20:13	Reserved	-
12	CMSBDF	向下计数专用比较基准值匹配 B
11	CMSBUF	向上计数专用比较基准值匹配 B
10	CMSADF	向下计数专用比较基准值匹配 A
9	CMSAUF	向上计数专用比较基准值匹配 A
8	DTEF	死区时间错误 0: 未发生死区时间错误; 1: 发生死区时间错误
7	UDFF	下溢匹配 0: 未发生锯齿波下溢或三角波计数到谷点 1: 发生锯齿波下溢或三角波计数到谷点
6	OVFF	上溢匹配 0: 未发生锯齿波上溢或三角波计数到峰点 1: 发生锯齿波上溢或三角波计数到峰点
5:4	Reserved	-
3	CMDF	计数匹配 D 0: GCMDR 寄存器的值与计数值不相等; 1: GCMDR 寄存器的值与计数值相等
2	CMCF	计数匹配 C 0: GCMCR 寄存器的值与计数值不相等; 1: GCMCR 寄存器的值与计数值相等
1	CMBF	计数匹配 B 0: GCMBR 寄存器的值与计数值不相等, 且未发生 CHxB 捕获完成动作 1: GCMBR 寄存器的值与计数值相等, 或发生 CHxB 捕获完成动作
0	CMAF	计数匹配 A 0: GCMAR 寄存器的值与计数值不相等, 且未发生 CHxA 捕获完成动作

		1: GCMAR 寄存器的值与计数值相等，或发生 CHxA 捕获完成动作
--	--	--------------------------------------

17.3.15 Hardware Start Event Select Register (TIMx_HSTAR)

地址偏移量: 0x074

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
START S	Reserved														
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HST A 15	HST A 14	HST A 13	HST A 12	HST A 11	HST A 10	HST A 9	HST A 8	HST A 7	HST A 6	HST A 5	HST A 4	HST A 3	HST A 2	HST A 1	HST A 0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能
31	STARTS	硬件启动使能 0: 硬件启动无效 1: 硬件启动有效 注: 硬件启动有效时, SSTAR 的设定无效
30:16	Reserved	-
15	HSTA15	硬件启动条件 15: TIMTRID 端口上采样到下降沿 0: 条件匹配时, 硬件启动无效 1: 条件匹配时, 硬件启动有效
14	HSTA14	硬件启动条件 14: TIMTRID 端口上采样到上升沿 0: 条件匹配时, 硬件启动无效 1: 条件匹配时, 硬件启动有效
13	HSTA13	硬件启动条件 13: TIMTRIC 端口上采样到下降沿 0: 条件匹配时, 硬件启动无效 1: 条件匹配时, 硬件启动有效
12	HSTA12	硬件启动条件 12: TIMTRIC 端口上采样到上升沿 0: 条件匹配时, 硬件启动无效 1: 条件匹配时, 硬件启动有效
11	HSTA11	硬件启动条件 11: TIMTRIB 端口上采样到下降沿 0: 条件匹配时, 硬件启动无效 1: 条件匹配时, 硬件启动有效
10	HSTA10	硬件启动条件 10: TIMTRIB 端口上采样到上升沿 0: 条件匹配时, 硬件启动无效 1: 条件匹配时, 硬件启动有效
9	HSTA9	硬件启动条件 9: TIMTRIA 端口上采样到下降沿 0: 条件匹配时, 硬件启动无效 1: 条件匹配时, 硬件启动有效
8	HSTA8	硬件启动条件 8: TIMTRIA 端口上采样到上升沿 0: 条件匹配时, 硬件启动无效

		1: 条件匹配时，硬件启动有效
7	HSTA7	硬件启动条件 7: CHxB 端口上采样到下降沿 0: 条件匹配时，硬件启动无效 1: 条件匹配时，硬件启动有效
6	HSTA6	硬件启动条件 6: CHxB 端口上采样到上升沿 0: 条件匹配时，硬件启动无效 1: 条件匹配时，硬件启动有效
5	HSTA5	硬件启动条件 5: CHxA 端口上采样到下降沿 0: 条件匹配时，硬件启动无效 1: 条件匹配时，硬件启动有效
4	HSTA4	硬件启动条件 4: CHxA 端口上采样到上升沿 0: 条件匹配时，硬件启动无效 1: 条件匹配时，硬件启动有效
3	HSTA3	硬件启动条件 3: 从 AOS 来的事件触发 3 有效 0: 条件匹配时，硬件启动无效 1: 条件匹配时，硬件启动有效
2	HSTA2	硬件启动条件 2: 从 AOS 来的事件触发 2 有效 0: 条件匹配时，硬件启动无效 1: 条件匹配时，硬件启动有效
1	HSTA1	硬件启动条件 1: 从 AOS 来的事件触发 1 有效 0: 条件匹配时，硬件启动无效 1: 条件匹配时，硬件启动有效
0	HSTA0	硬件启动条件 0: 从 AOS 来的事件触发 0 有效 0: 条件匹配时，硬件启动无效 1: 条件匹配时，硬件启动有效

17.3.16 Hardware Stop Event Select Register (TIMx_HSTPR)

地址偏移量: 0x078

复位值: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HST P 15	HST A 14	HST P 13	HST P 12	HST P 11	HST P 10	HST P 9	HST P 8	HST P 7	HST P 6	HST P 5	HST P 4	HST P 3	HST P 2	HST P 1	HST P 0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能
31	STOPS	硬件停止使能 0: 硬件停止无效 1: 硬件停止有效 注: 硬件停止有效时, 软件停止的设定无效
30:16	Reserved	-
15	HSTP15	硬件停止条件 15: TIMTRID 端口上采样到下降沿 0: 条件匹配时, 硬件停止无效 1: 条件匹配时, 硬件停止有效
14	HSTP14	硬件停止条件 14: TIMTRID 端口上采样到上升沿 0: 条件匹配时, 硬件停止无效 1: 条件匹配时, 硬件停止有效
13	HSTP13	硬件停止条件 13: TIMTRIC 端口上采样到下降沿 0: 条件匹配时, 硬件停止无效 1: 条件匹配时, 硬件停止有效
12	HSTP12	硬件停止条件 12: TIMTRIC 端口上采样到上升沿 0: 条件匹配时, 硬件停止无效 1: 条件匹配时, 硬件停止有效
11	HSTP11	硬件停止条件 11: TIMTRIB 端口上采样到下降沿 0: 条件匹配时, 硬件停止无效 1: 条件匹配时, 硬件停止有效
10	HSTP10	硬件停止条件 10: TIMTRIB 端口上采样到上升沿 0: 条件匹配时, 硬件停止无效 1: 条件匹配时, 硬件停止有效
9	HSTP9	硬件停止条件 9: TIMTRIA 端口上采样到下降沿 0: 条件匹配时, 硬件停止无效 1: 条件匹配时, 硬件停止有效
8	HSTP8	硬件停止条件 8: TIMTRIA 端口上采样到上升沿 0: 条件匹配时, 硬件停止无效

		1: 条件匹配时，硬件停止有效
7	HSTP7	硬件停止条件 7: CHxB 端口上采样到下降沿 0: 条件匹配时，硬件停止无效 1: 条件匹配时，硬件停止有效
6	HSTP6	硬件停止条件 6: CHxB 端口上采样到上升沿 0: 条件匹配时，硬件停止无效 1: 条件匹配时，硬件停止有效
5	HSTP5	硬件停止条件 5: CHxA 端口上采样到下降沿 0: 条件匹配时，硬件停止无效 1: 条件匹配时，硬件停止有效
4	HSTP4	硬件停止条件 4: CHxA 端口上采样到上升沿 0: 条件匹配时，硬件停止无效 1: 条件匹配时，硬件停止有效
3	HSTP3	硬件停止条件 3: 从 AOS 来的事件触发 3 有效 0: 条件匹配时，硬件停止无效 1: 条件匹配时，硬件停止有效
2	HSTP2	硬件停止条件 2: 从 AOS 来的事件触发 2 有效 0: 条件匹配时，硬件停止无效 1: 条件匹配时，硬件停止有效
1	HSTP1	硬件停止条件 1: 从 AOS 来的事件触发 1 有效 0: 条件匹配时，硬件停止无效 1: 条件匹配时，硬件停止有效
0	HSTP0	硬件停止条件 0: 从 AOS 来的事件触发 0 有效 0: 条件匹配时，硬件停止无效 1: 条件匹配时，硬件停止有效

17.3.17 Hardware Clear Event Select Register (TIMx_HCELR)

地址偏移量: 0x07C

复位值: 0x0000 0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLEAR	Reserved															
RW																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
HCE L 15	HCE L 14	HCE L 13	HCE L 12	HCE L 11	HCE L 10	HCE L 9	HCE L 8	HCE L 7	HCE L 6	HCE L 5	HCE L 4	HCE L 3	HCE L 2	HCE L 1	HCE L 0	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

位	标记	功能
31	CLEAR	硬件清零使能 0: 硬件清零无效 1: 硬件清零有效 注: 硬件清零有效时, 软件清零的设定无效
30:16	Reserved	-
15	HCEL15	硬件清零条件 15: TIMTRID 端口上采样到下降沿 0: 条件匹配时, 硬件清零无效 1: 条件匹配时, 硬件清零有效
14	HCEL14	硬件清零条件 14: TIMTRID 端口上采样到上升沿 0: 条件匹配时, 硬件清零无效 1: 条件匹配时, 硬件清零有效
13	HCEL13	硬件清零条件 13: TIMTRIC 端口上采样到下降沿 0: 条件匹配时, 硬件清零无效 1: 条件匹配时, 硬件清零有效
12	HCEL12	硬件清零条件 12: TIMTRIC 端口上采样到上升沿 0: 条件匹配时, 硬件清零无效 1: 条件匹配时, 硬件清零有效
11	HCEL11	硬件清零条件 11: TIMTRIB 端口上采样到下降沿 0: 条件匹配时, 硬件清零无效 1: 条件匹配时, 硬件清零有效
10	HCEL10	硬件清零条件 10: TIMTRIB 端口上采样到上升沿 0: 条件匹配时, 硬件清零无效 1: 条件匹配时, 硬件清零有效
9	HCEL9	硬件清零条件 9: TIMTRIA 端口上采样到下降沿 0: 条件匹配时, 硬件清零无效 1: 条件匹配时, 硬件清零有效
8	HCEL8	硬件清零条件 8: TIMTRIA 端口上采样到上升沿 0: 条件匹配时, 硬件清零无效

		1: 条件匹配时，硬件清零有效
7	HCEL7	硬件清零条件 7: CHxB 端口上采样到下降沿 0: 条件匹配时，硬件清零无效 1: 条件匹配时，硬件清零有效
6	HCEL6	硬件清零条件 6: CHxB 端口上采样到上升沿 0: 条件匹配时，硬件清零无效 1: 条件匹配时，硬件清零有效
5	HCEL5	硬件清零条件 5: CHxA 端口上采样到下降沿 0: 条件匹配时，硬件清零无效 1: 条件匹配时，硬件清零有效
4	HCEL4	硬件清零条件 4: CHxA 端口上采样到上升沿 0: 条件匹配时，硬件清零无效 1: 条件匹配时，硬件清零有效
3	HCEL3	硬件清零条件 3: 从 AOS 来的事件触发 3 有效 0: 条件匹配时，硬件清零无效 1: 条件匹配时，硬件清零有效
2	HCEL2	硬件清零条件 2: 从 AOS 来的事件触发 2 有效 0: 条件匹配时，硬件清零无效 1: 条件匹配时，硬件清零有效
1	HCEL1	硬件清零条件 1: 从 AOS 来的事件触发 1 有效 0: 条件匹配时，硬件清零无效 1: 条件匹配时，硬件清零有效
0	HCELO	硬件清零条件 0: 从 AOS 来的事件触发 0 有效 0: 条件匹配时，硬件清零无效 1: 条件匹配时，硬件清零有效

17.3.18 Hardware Capture A Event Selection Register (TIMx_HCPAR)

地址偏移量: 0x080

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCP A 15	HCP A 14	HCP A 13	HCP A 12	HCP A 11	HCP A 10	HCP A 9	HCP A 8	HCP A 7	HCP A 6	HCP A 5	HCP A 4	HCP A 3	HCP A 2	HCP A 1	HCP A 0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能
31:16	Reserved	-
15	HCRA15	硬件捕获 A 条件 15: TIMTRID 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
14	HCRA14	硬件捕获 A 条件 14: TIMTRID 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
13	HCRA13	硬件捕获 A 条件 13: TIMTRIC 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
12	HCRA12	硬件捕获 A 条件 12: TIMTRIC 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
11	HCRA11	硬件捕获 A 条件 11: TIMTRIB 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
10	HCRA10	硬件捕获 A 条件 10: TIMTRIB 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
9	HCRA9	硬件捕获 A 条件 9: TIMTRIA 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
8	HCRA8	硬件捕获 A 条件 8: TIMTRIA 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
7	HCRA7	硬件捕获 A 条件 7: CHxB 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
6	HCRA6	硬件捕获 A 条件 6: CHxB 端口上采样到上升沿

		0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
5	HCPA5	硬件捕获 A 条件 5: CHxA 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
4	HCPA4	硬件捕获 A 条件 4: CHxA 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
3	HCPA3	硬件捕获 A 条件 3: 从 AOS 来的事件触发 3 有效 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
2	HCPA2	硬件捕获 A 条件 2: 从 AOS 来的事件触发 2 有效 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
1	HCPA1	硬件捕获 A 条件 1: 从 AOS 来的事件触发 1 有效 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效
0	HCPA0	硬件捕获 A 条件 0: 从 AOS 来的事件触发 0 有效 0: 条件匹配时, 硬件捕获 A 无效 1: 条件匹配时, 硬件捕获 A 有效

17.3.19 Hardware Capture B Event Selection Register (TIMx_HCPBR)

地址偏移量: 0x084

复位值: 0x0000 0000

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

位	标记	功能
31:16	Reserved	-
15	HCPB15	硬件捕获 B 条件 15: TIMTRID 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
14	HCPB14	硬件捕获 B 条件 14: TIMTRID 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
13	HCPB13	硬件捕获 B 条件 13: TIMTRIC 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
12	HCPB12	硬件捕获 B 条件 12: TIMTRIC 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
11	HCPB11	硬件捕获 B 条件 11: TIMTRIB 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
10	HCPB10	硬件捕获 B 条件 10: TIMTRIB 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
9	HCPB9	硬件捕获 B 条件 9: TIMTRIA 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
8	HCPB8	硬件捕获 B 条件 8: TIMTRIA 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
7	HCPB7	硬件捕获 B 条件 7: CHxB 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
6	HCPB6	硬件捕获 B 条件 6: CHxB 端口上采样到上升沿

		0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
5	HCPB5	硬件捕获 B 条件 5: CHxA 端口上采样到下降沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
4	HCPB4	硬件捕获 B 条件 4: CHxA 端口上采样到上升沿 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
3	HCPB3	硬件捕获 B 条件 3: 从 AOS 来的事件触发 3 有效 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
2	HCPB2	硬件捕获 B 条件 2: 从 AOS 来的事件触发 2 有效 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
1	HCPB1	硬件捕获 B 条件 1: 从 AOS 来的事件触发 1 有效 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效
0	HCPB0	硬件捕获 B 条件 0: 从 AOS 来的事件触发 0 有效 0: 条件匹配时, 硬件捕获 B 无效 1: 条件匹配时, 硬件捕获 B 有效

17.3.20 Hardware Increment Event Select Register (TIMx_HCUPR)

地址偏移量: 0x088

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														HCU	HCU
												P	P	P	P
												19	18	17	16
												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCU	HCU	HCU	HCU	HCU	HCU	HCU	HCU	HCU	HCU	HCU	HCU	HCU	HCU	HCU	HCU
P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能
31:20	Reserved	-
19	HCUP19	硬件递加条件: 从 AOS 来的事件触发 3 有效 0: 条件匹配时, 硬件递加无效 1: 条件匹配时, 硬件递加有效
18	HCUP18	硬件递加条件: 从 AOS 来的事件触发 2 有效 0: 条件匹配时, 硬件递加无效 1: 条件匹配时, 硬件递加有效
17	HCUP17	硬件递加条件: 从 AOS 来的事件触发 1 有效 0: 条件匹配时, 硬件递加无效 1: 条件匹配时, 硬件递加有效
16	HCUP16	硬件递加条件: 从 AOS 来的事件触发 0 有效 0: 条件匹配时, 硬件递加无效 1: 条件匹配时, 硬件递加有效
15	HCUP15	硬件递加条件: TIMTRID 端口上采样到下降沿 0: 条件匹配时, 硬件递加无效 1: 条件匹配时, 硬件递加有效
14	HCUP14	硬件递加条件: TIMTRID 端口上采样到上升沿 0: 条件匹配时, 硬件递加无效 1: 条件匹配时, 硬件递加有效
13	HCUP13	硬件递加条件: TIMTRIC 端口上采样到下降沿 0: 条件匹配时, 硬件递加无效 1: 条件匹配时, 硬件递加有效
12	HCUP12	硬件递加条件: TIMTRIC 端口上采样到上升沿 0: 条件匹配时, 硬件递加无效 1: 条件匹配时, 硬件递加有效
11	HCUP11	硬件递加条件: TIMTRIB 端口上采样到下降沿 0: 条件匹配时, 硬件递加无效

		1: 条件匹配时，硬件递加有效
10	HCUP10	硬件递加条件：TIMTRIB 端口上采样到上升沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
9	HCUP9	硬件递加条件：TIMTRIA 端口上采样到下降沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
8	HCUP8	硬件递加条件：TIMTRIA 端口上采样到上升沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
7	HCUP7	硬件递加条件：CHxB 端口为高电平时，CHxA 端口上采样到下降沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
6	HCUP6	硬件递加条件：CHxB 端口为高电平时，CHxA 端口上采样到上升沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
5	HCUP5	硬件递加条件：CHxB 端口为低电平时，CHxA 端口上采样到下降沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
4	HCUP4	硬件递加条件：CHxB 端口为低电平时，CHxA 端口上采样到上升沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
3	HCUP3	硬件递加条件：CHxA 端口为高电平时，CHxB 端口上采样到下降沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
2	HCUP2	硬件递加条件：CHxA 端口为高电平时，CHxB 端口上采样到上升沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
1	HCUP1	硬件递加条件：CHxA 端口为低电平时，CHxB 端口上采样到下降沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效
0	HCUP0	硬件递加条件：CHxA 端口为低电平时，CHxB 端口上采样到上升沿 0: 条件匹配时，硬件递加无效 1: 条件匹配时，硬件递加有效

17.3.21 Hardware Decrement Event Select Register (TIMx_HCDOR)

地址偏移量: 0x08C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															HCD O 19 RW
HCD O 15	HCD O 14	HCD O 13	HCD O 12	HCD O 11	HCD O 10	HCD O 9	HCD O 8	HCD O 7	HCD O 6	HCD O 5	HCD O 4	HCD O 3	HCD O 2	HCD O 1	HCD O 0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能
31:20	Reserved	-
19	HCDO19	硬件递减条件: 从 AOS 来的事件触发 3 有效 0: 条件匹配时, 硬件递减无效 1: 条件匹配时, 硬件递减有效
18	HCDO18	硬件递减条件: 从 AOS 来的事件触发 2 有效 0: 条件匹配时, 硬件递减无效 1: 条件匹配时, 硬件递减有效
17	HCDO17	硬件递减条件: 从 AOS 来的事件触发 1 有效 0: 条件匹配时, 硬件递减无效 1: 条件匹配时, 硬件递减有效
16	HCDO16	硬件递减条件: 从 AOS 来的事件触发 0 有效 0: 条件匹配时, 硬件递减无效 1: 条件匹配时, 硬件递减有效
15	HCDO15	硬件递减条件: TIMTRID 端口上采样到下降沿 0: 条件匹配时, 硬件递减无效 1: 条件匹配时, 硬件递减有效
14	HCDO14	硬件递减条件: TIMTRID 端口上采样到上升沿 0: 条件匹配时, 硬件递减无效 1: 条件匹配时, 硬件递减有效
13	HCDO13	硬件递减条件: TIMTRIC 端口上采样到下降沿 0: 条件匹配时, 硬件递减无效 1: 条件匹配时, 硬件递减有效
12	HCDO12	硬件递减条件: TIMTRIC 端口上采样到上升沿 0: 条件匹配时, 硬件递减无效 1: 条件匹配时, 硬件递减有效
11	HCDO11	硬件递减条件: TIMTRIB 端口上采样到下降沿 0: 条件匹配时, 硬件递减无效

		1: 条件匹配时，硬件递减有效
10	HCDO10	硬件递减条件：TIMTRIB 端口上采样到上升沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
9	HCDO9	硬件递减条件：TIMTRIA 端口上采样到下降沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
8	HCDO8	硬件递减条件：TIMTRIA 端口上采样到上升沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
7	HCDO7	硬件递减条件：CHxB 端口为高电平时，CHxA 端口上采样到下降沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
6	HCDO6	硬件递减条件：CHxB 端口为高电平时，CHxA 端口上采样到上升沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
5	HCDO5	硬件递减条件：CHxB 端口为低电平时，CHxA 端口上采样到下降沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
4	HCDO4	硬件递减条件：CHxB 端口为低电平时，CHxA 端口上采样到上升沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
3	HCDO3	硬件递减条件：CHxA 端口为高电平时，CHxB 端口上采样到下降沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
2	HCDO2	硬件递减条件：CHxA 端口为高电平时，CHxB 端口上采样到上升沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
1	HCDO1	硬件递减条件：CHxA 端口为低电平时，CHxB 端口上采样到下降沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效
0	HCDO0	硬件递减条件：CHxA 端口为低电平时，CHxB 端口上采样到上升沿 0: 条件匹配时，硬件递减无效 1: 条件匹配时，硬件递减有效

17.3.22 Software Synchronous Start Register (TIMx_SSTAR)

地址偏移量: 0x3F4

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												SST A2	SST A1	SST A0	RW

位	标记	功能
31:3	Reserved	
2	SSTA2	Timer6 软件启动 0: 软件启动无效 1: 软件启动使能
1	SSTA1	Timer5 软件启动 0: 软件启动无效 1: 软件启动使能
0	SSTA0	Timer4 软件启动 0: 软件启动无效 1: 软件启动使能

17.3.23 Software Synchronous Stop Register (TIMx_SSTPR)

地址偏移量: 0x3F8

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												SST P2	SST P1	SST P0	RW

位	标记	功能
31:3	Reserved	
2	SSTP2	Timer6 软件停止 0: 软件停止无效 1: 软件停止使能
1	SSTP1	Timer5 软件停止 0: 软件停止无效 1: 软件停止使能
0	SSTP0	Timer4 软件停止 0: 软件停止无效 1: 软件停止使能

17.3.24 Software Synchronous Clear Register (TIMx_SCLRR)

地址偏移量: 0x3FC

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
SCL R2 SCL R1 SCL R0 RW RW RW															

位	标记	功能
31:3	Reserved	
2	SCLR2	Timer6 软件清零 0: 软件清零无效 1: 软件清零使能
1	SCLR1	Timer5 软件清零 0: 软件清零无效 1: 软件清零使能
0	SCLR0	Timer4 软件清零 0: 软件清零无效 1: 软件清零使能

17.3.25 Interrupt Flag Register (TIMx_IFR)

地址偏移量: 0x100

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAM HF	SAM LF	Reserved				DTEF	UDFF	OVFF	Reserved	CMD F	CM CF	CM BF	CM AF	RO	RO
RO	RO					RO	RO	RO		RO	RO	RO	RO		

位	标记	功能
31:16	Reserved	-
15	SAMHF	CHxA/B 端口高状态中断标志 0: CHxA 和 CHxB 端口上没有同时出现高电平 1: CHxA 和 CHxB 端口上同时出现高电平
14	SAMLF	CHxA/B 端口低状态中断标志 0: CHxA 和 CHxB 端口上没有同时出现低电平 1: CHxA 和 CHxB 端口上同时出现低电平
13:9	Reserved	-
8	DTEF	死区时间错误中断标志 0: 未发生死区时间错误；1: 发生死区时间错误
7	UDFF	下溢匹配中断标志 0: 未发生锯齿波下溢或三角波计数到谷点 1: 发生锯齿波下溢或三角波计数到谷点
6	OVFF	上溢匹配中断标志 0: 未发生锯齿波上溢或三角波计数到峰点 1: 发生锯齿波上溢或三角波计数到峰点
5:4	Reserved	-
3	CMDF	计数匹配 D 中断标志 0: GCMDR 寄存器的值与计数值不相等；1: GCMDR 寄存器的值与计数值相等
2	CMCF	计数匹配 C 中断标志 0: GCMCR 寄存器的值与计数值不相等；1: GCMCR 寄存器的值与计数值相等
1	CMBF	计数匹配 B 中断标志 0: GCMBR 寄存器的值与计数值不相等，且未发生 CHxB 捕获完成动作 1: GCMBR 寄存器的值与计数值相等，或发生 CHxB 捕获完成动作
0	CMAF	计数匹配 A 中断标志 0: GCMAR 寄存器的值与计数值不相等，且未发生 CHxA 捕获完成动作 1: GCMAR 寄存器的值与计数值相等，或发生 CHxA 捕获完成动作

17.3.26 Interrupt Flag Clear Register (TIMx_ICLR)

地址偏移量: 0x104

复位值: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAM HC	SAML C	Reserved				DTEC	UDFC	OVFC	Reserved		CMD C	CM CC	CM BC	CM AC	
R1W 0	R1W 0					R1W0	R1W0	R1W0			R1W 0	R1 W0	R1 W0	R1 W0	

位	标记	功能
31:16	Reserved	-
15	SAMHC	CHxA/B 端口高状态中断标志清除, 写 1 无效, 写 0 清除对应中断
14	SAMLC	CHxA/B 端口低状态中断标志清除, 写 1 无效, 写 0 清除对应中断
13:9	Reserved	-
8	DTEC	死区时间错误中断标志清除, 写 1 无效, 写 0 清除对应中断
7	UDFC	下溢匹配中断标志清除, 写 1 无效, 写 0 清除对应中断
6	OVFC	上溢匹配中断标志清除, 写 1 无效, 写 0 清除对应中断
5:4	Reserved	-
3	CMDC	计数匹配 D 中断标志清除, 写 1 无效, 写 0 清除对应中断
2	CMCC	计数匹配 C 中断标志清除, 写 1 无效, 写 0 清除对应中断
1	CMBC	计数匹配 B 中断标志清除, 写 1 无效, 写 0 清除对应中断
0	CMAC	计数匹配 A 中断标志清除, 写 1 无效, 写 0 清除对应中断

17.3.27 Spread spectrum and interrupt trigger selection (TIMx_CR)

地址偏移量: 0x108

复位值: 0x0000 0300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									Dma_s_cmb	Dma_s_cma	Dma_g_udf	Dma_g_ovf		Reserved	Dma_g_cmd
									RW	RW	RW	RW			RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Dma_g_cmc	Dma_g_cmb	Dma_g_cma	CMSBE	CMSAE	DITENB	DITENA	DITENS	UDFE	OVFE	Reserved	CMD E	CMCE	CMB E	CMA E	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW

位	标记	功能
31:23	Reserved	-
22	Dma_s_cmb	专用比较 DMA 使能
21	Dma_s_cma	专用比较 DMA 使能
20	Dma_g_udf	下溢出 DMA 使能
19	Dma_g_ovf	上溢出 DMA 使能
18:17	Reserved	
16	Dma_g_cmd	通用比较 DMA 使能
15	Dma_g_cmc	通用比较 DMA 使能
14	Dma_g_cmb	通用比较 DMA 使能
13	Dma_g_cma	通用比较 DMA 使能
12	CMSBE	专用比较基准值匹配 B 使能触发 ADC
11	CMSAE	专用比较基准值匹配 A 使能触发 ADC
10	DITENS	PWM 展频计数选择 0: 选择下溢出, 1: 选择上溢出
9	DITENB	PWM 通道 B 展频使能 0: 使能无效, 1: 使能有效, 每个周期改变 PWM 的输出延时
9	DITENA	PWM 通道 A 展频使能 0: 使能无效, 1: 使能有效, 每个周期改变 PWM 的输出延时
7	UDFE	下溢匹配使能触发 ADC 0: 使能无效, 1: 使能有效, 这个匹配可以控制 ADC/AOS_i_tirg
6	OVFE	上溢匹配使能触发 ADC 0: 使能无效, 1: 使能有效, 这个匹配可以控制 ADC/AOS_i_tirg
5:4	Reserved	-
3	CMDE	计数匹配 D 使能触发 ADC 0: 使能无效, 1: 使能有效, 这个匹配可以控制 ADC/AOS_i_tirg
2	CMCE	计数匹配 C 使能触发 ADC 0: 使能无效, 1: 使能有效, 这个匹配可以控制 ADC/AOS_i_tirg

1	CMBE	计数匹配 B 使能触发 ADC 0: 使能无效, 1: 使能有效, 这个匹配可以控制 ADC/AOS_i_tirg
0	CMAE	计数匹配 A 使能触发 ADC 0: 使能无效, 1: 使能有效, 这个匹配可以控制 ADC/AOS_i_tirg

17.3.28 AOS Selection Control Register (TIMx_AOSSR)

地址偏移量: 0x110

复位值: 0x0000 0000

Timer4/5/6 使用同一个实体寄存器，任意一个定时器其更改后，在另外两个定时器的值会同时更改。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SMH2	SMH1	SMH0	SML2	SML1	SML0	SOF	TBK	Reserved	BFILTEN	BFILTS	FSAMEME	FBRAKE		
	RW	RW	RW	RW	RW	RW				RW	RW	R	R		

位	标记	功能
31:14	Reserved	-
13	SMH2	通道 2 同高选择 0: 选择无效, 1: 选择有效, 出现同高时 AOS_i_odis[1]
12	SMH1	通道 1 同高选择 0: 选择无效, 1: 选择有效, 出现同高时 AOS_i_odis[1]
11	SMH0	通道 0 同高选择 0: 选择无效, 1: 选择有效, 出现同高时 AOS_i_odis[1]
10	SML2	通道 2 同低选择 0: 选择无效, 1: 选择有效, 出现同低时 AOS_i_odis[1]
9	SML1	通道 1 同低选择 0: 选择无效, 1: 选择有效, 出现同低时 AOS_i_odis[1]
8	SML0	通道 0 同低选择 0: 选择无效, 1: 选择有效, 出现同低时 AOS_i_odis[1]
7	SOFTBK	软件刹车: 写 1 实现软件刹车
13	Reserved	-
4	BFILTEN	端口刹车滤波使能
3:2	BFILTS	端口刹车滤波时钟选择
1	FSAME	同高同低刹车标志, 只读
0	FBRAKE	端口刹车标志, 只读

17.3.29 AOS selection control register flag clear (TIMx_AOSCL)

地址偏移量: 0x114

复位值: 0x0000 0000

Timer4/5/6 使用同一个实体寄存器，任意一个定时器其更改后，在另外两个定时器的值会同时更改。

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

位	标记	功能
31:2	Reserved	-
1	FSAME	同高同低刹车标志清除，写 0 清除，写 1 无效，读恒为 1
0	FBRAKE	端口刹车标志清除，写 0 清除，写 1 无效，读恒为 1

17.3.30 Port brake control register (TIMx_PTAKS)

地址偏移量: 0x118

复位值: 0x0000 0000

Timer4/5/6 使用同一个实体寄存器，任意一个定时器其更改后，在另外两个定时器的值会同时更改。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15 4	EN14 3	EN13 2	EN12 1	EN11 0	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	RW

位	标记	功能
31:16	Reserved	-
15	EN15	PD5 刹车端口使能：1 选择，0 无效
14	EN14	PC15 刹车端口使能：1 选择，0 无效
13	EN13	PB15 刹车端口使能：1 选择，0 无效
12	EN12	PA15 刹车端口使能：1 选择，0 无效
11	EN11	PD1 刹车端口使能：1 选择，0 无效
10	EN10	PC11 刹车端口使能：1 选择，0 无效
9	EN9	PB11 刹车端口使能：1 选择，0 无效
8	EN8	PA11 刹车端口使能：1 选择，0 无效
7	EN7	PD7 刹车端口使能：1 选择，0 无效
6	EN6	PC7 刹车端口使能：1 选择，0 无效
5	EN5	PB7 刹车端口使能：1 选择，0 无效
4	EN4	PA7 刹车端口使能：1 选择，0 无效
3	EN3	PD3 刹车端口使能：1 选择，0 无效
2	EN2	PC3 刹车端口使能：1 选择，0 无效
1	EN1	PB3 刹车端口使能：1 选择，0 无效
0	EN0	PA3 刹车端口使能：1 选择，0 无效

17.3.31 Port Trigger Control Register (TIMx_TTRIG)

地址偏移量: 0x11C

复位值: 0x0000 0000

Timer4/5/6 使用同一个实体寄存器，任意一个定时器其更改后，在另外两个定时器的值会同时更改。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGDS				TRIGCS				TRIGBS				TRIGAS			
RW				RW				RW				RW			

位	标记	功能
31:16	Reserved	-
15:12	TRIGDS	TIMx 触发 D 端口选择
11:8	TRIGCS	TIMx 触发 C 端口选择
7:4	TRIGBS	TIMx 触发 B 端口选择
3:0	TRIGAS	TIMx 触发 A 端口选择

控制信号与端口选择如下

0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
PA3	PB3	PC3	PD3	PA7	PB7	PC7	PD7	PA11	PB11	PC11	PD1	PA15	PB15	PC15	PD5

17.3.32 AOS Trigger Control Register (TIMx_ITRIG)

地址偏移量: 0x120

复位值: 0x0000 0000

Timer4/5/6 使用同一个实体寄存器，任意一个定时器其更改后，在另外两个定时器的值会同时更改。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IAOS3S				IAOS2S				IAOS1S				IAOS0S			
RW				RW				RW				RW			

位	标记	功能
31:16	Reserved	-
15:12	IAOS3S	TIMx AOS3 触发源选择
11:8	IAOS2S	TIMx AOS2 触发源选择
7:4	IAOS1S	TIMx AOS1 触发源选择
3:0	IAOS0S	TIMx AOS0 触发源选择

控制信号 (IAOSxS) 与中断源选择如下 (x=0,1,2,3)

0000	0001	0010	0011	0100	0101	0110	0111
TIMO_INT	TIM1_INT	TIM2_INT	LPTIMER_INT	TIM4_INTS	TIM5_INTS	TIM6_INTS	UART0_INT
1000	1001	1010	1011	1100	1101	1110	1111
UART1_INT	LPUART0_INT	VCO_INT	VC1_INT	RTC_INT	PCA_INT	SPI_INT	ADC_INT

17.3.33 Port brake polarity control register (TIMx_PTBKPR)

地址偏移量: 0x124

复位值: 0x0000 0000

Timer4/5/6 使用同一个实体寄存器，任意一个定时器其更改后，在另外两个定时器的值会同时更改。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POL15	POL14	POL13	POL12	POL11	POL10	POL9	POL8	POL7	POL6	POL5	POL4	POL3	POL2	POL1	POL0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能
31:16	Reserved	-
15	POL15	PD5 刹车端口极性选择：1 低电平有效，0 高电平有效
14	POL14	PC15 刹车端口极性选择：1 低电平有效，0 高电平有效
13	POL13	PB15 刹车端口极性选择：1 低电平有效，0 高电平有效
12	POL12	PA15 刹车端口极性选择：1 低电平有效，0 高电平有效
11	POL11	PD1 刹车端口极性选择：1 低电平有效，0 高电平有效
10	POL10	PC11 刹车端口极性选择：1 低电平有效，0 高电平有效
9	POL9	PB11 刹车端口极性选择：1 低电平有效，0 高电平有效
8	POL8	PA11 刹车端口极性选择：1 低电平有效，0 高电平有效
7	POL7	PD7 刹车端口极性选择：1 低电平有效，0 高电平有效
6	POL6	PC7 刹车端口极性选择：1 低电平有效，0 高电平有效
5	POL5	PB7 刹车端口极性选择：1 低电平有效，0 高电平有效
4	POL4	PA7 刹车端口极性选择：1 低电平有效，0 高电平有效
3	POL3	PD3 刹车端口极性选择：1 低电平有效，0 高电平有效
2	POL2	PC3 刹车端口极性选择：1 低电平有效，0 高电平有效
1	POL1	PB3 刹车端口极性选择：1 低电平有效，0 高电平有效
0	POL0	PA3 刹车端口极性选择：1 低电平有效，0 高电平有效

18 Real-time clock (RTC)

18.1 Real-time Clock Introduction

实时时钟/日历提供秒、分、时、日、周、月、年的信息，每月的天数和闰年的天数可自动调整。时钟操作可通过 AM/PM 寄存器位，决定采用 24 或 12 小时格式。表 18-1 所示是其基本特性。具有最小周期为 0.5S 的周期中断，如果需要更短的中断周期请使用 LPTIM 产生中断。

表 18-1 RTC 的基本特性

时钟源	片外低速晶振 XTL (32.768kHz) 片内低速振荡器 RCL(32kHz, 1%精度) 片外高速晶振 XTH
基本功能	可计算 00~99 年之间的秒、分、时、日、周、月、年
	可自动进行闰年调整
	可配置为 24 或 12 小时格式
	可程序控制启动或停止
	具有闹钟功能
	具有高精度 1Hz 方波信号输出
中断	具有最小周期为 0.5S 的周期中断
	具有精确到秒的闹钟中断

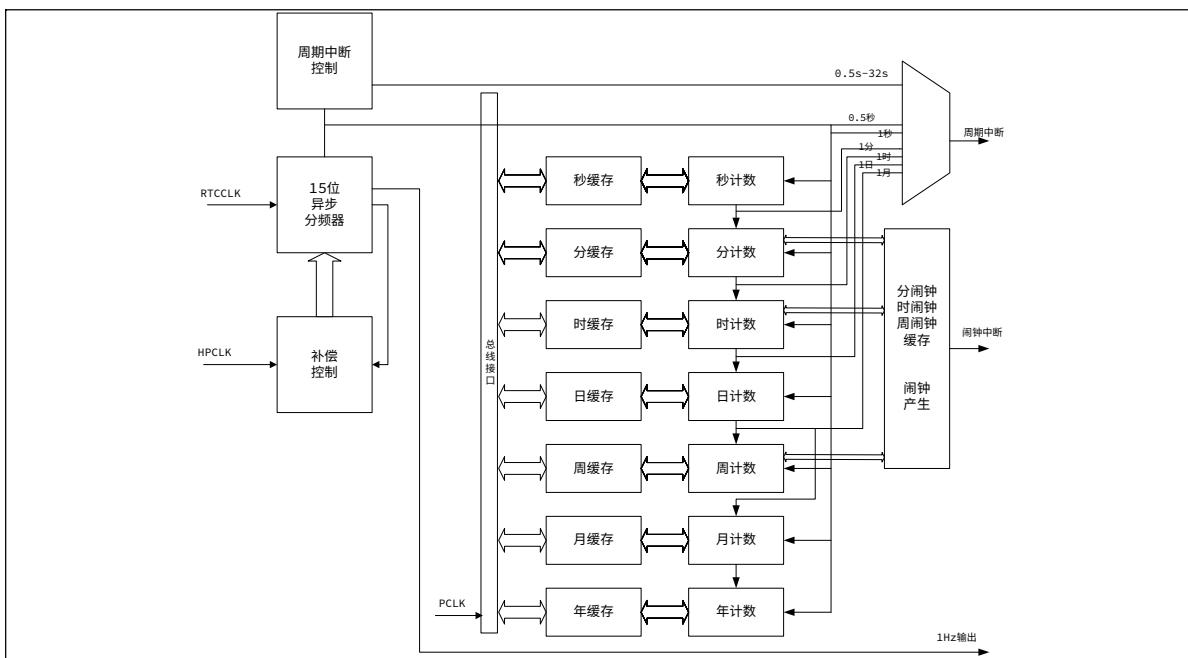


图 18-1 RTC 框图

18.2 Real-time clock function description

实时时钟的时钟源可配置为外部低速晶振，外部高速晶振，内部低速 RC；默认使用外部低速晶振。控制寄存器 RTC_CR0、RTC_CR1 与 RTC_COMPEN 只受上电复位控制，其他复位源不能复位这三个控制寄存器。其他数据寄存器上电状态不定，上电后需要初始化，不受任何复位影响。

所有软件写入和读取的日期时间值都为 BCD 码，无须十六进制转换为十进制。

任何无效的日期时间将无法写入，比如 32 日，25 时，70 秒，B 月等。

18.2.1 Power-on settings

RTC 在上电之后复位一次，在系统不掉电的情况下，外部各种复位请求都不能复位 RTC，RTC 会一直处于计数状态。在上电之后，设定日历初始值、闹钟设置、误差补偿、中断等之后，启动 RTC。

18.2.2 RTC count start setting

1. 设定 RTC_CR0.START=0，计数停止；
2. 设定 RTC_CR0.AMPM 和 RTC_CR0.PRDS，RTC_CR0.PRDX 设定时制和中断周期；
3. 设定 RTC_CR1.CKSEL 选择 RTC 的计时时钟；
4. 设定秒，分，时，周，日，月，年的日历计数寄存器；
5. 需要进行时钟误差补偿时，设定计数时钟误差补偿寄存器 RTC_COMPEN；
6. 清除中断标志位 RTC_CR1.ALMF，RTC_CR1.PRDF，并使能中断；
7. 设定 RTC_CR0.START=1，计数开始。

18.2.3 System Low Power Mode Switching

在 RTC 计数开始后，系统如果立即切换为低功耗模式时，请执行下列任意一种确认后再进行模式切换。

控制寄存器在系统控制寄存器 SYSCTRL1.RTC_LPW

在 RTC_CR0.START=1 设定后，经过 2 个以上的 RTC 计数时钟后再进行模式切换。

在 RTC_CR0.START=1 设定后，设定 RTC_CR1.WAIT=1，查询 RTC_CR1.WAITF=1。

再设定 RTC_CR1.WAIT=0，查询 RTC_CR1.WAITF=0 后再进行模式切换。

在 RTC 低功耗模式下，RTC 寄存器不能读写。在低功耗模式下，RTC 消耗更少的电流。

RTC 运行时切换低功耗模式不需要等待。

18.2.4 Reading the Count Register

有三种读取计数寄存器的方式：

方式 1：任意时刻读取方式 1

1. 设定 RTC_CR1.WAIT=1，停止日历寄存器计数，进入读写模式；
2. 查询直到 RTC_CR1.WAITF=1；
3. 读出秒，分，时，周，日，月，年计数寄存器值；
4. 设定 RTC_CR1.WAIT=0，计数器计数；
5. 查询直到 RTC_CR1.WAITF=0。

方式 2：任意时刻读取方式 2

1. 读出分，时，周，日，月，年计数寄存器值；
2. 读出秒计数寄存器值；
3. 再次读出秒计数寄存器值；
4. 判断两次秒的读出值是否相同，不同重新从第一步开始，相同读取结束。

方式 3：中断读取方式

在 RTC 周期中断服务中读取秒，分，时，周，日，月，年计数寄存器值。因为中断发生后到下次数据改变至少 0.5s 的时间。

18.2.5 Writing to the Count Register

1. 设定 RTC_CR1.WAIT=1，停止日历寄存器计数，进入读写模式；
2. 查询直到 RTC_CR1.WAITF=1；
3. 写入秒，分，时，周，日，月，年计数寄存器值；
4. 设定 RTC_CR1.WAIT=0，计数器重新开始计数。注意：须在 1 秒内完成所有写操作；
5. 查询直到 RTC_CR1.WAITF=0。

在 RTC 未启动模式下写秒，分，时，周，日，月，年计数寄存器不需要等 WAIT。

注意：

- 在计数模式下更改秒寄存器会复位秒计数，写分，时，周，日，月，年计数寄存器值不会影响 RTC 计数。

18.2.6 Alarm Setting

1. 设定 RTC_CR1.ALMEN=0，闹钟禁止；
2. 设定 RTC_CR1.ALMIE=1，闹钟中断许可；
3. 秒闹钟 RTC_ALMSEC，分闹钟 RTC_ALMMIN，时闹钟 RTC_ALMHOUR，周闹钟 RTC_ALMWEEK 设定；
4. 设定 RTC_CR1.ALMEN=1，闹钟许可；
5. 等待发生中断；
6. 由于闹钟中断和定周期中断共用中断请求信号，则当 RTC_CR1.ALMF=1 时，进入闹钟中断处理；否则进入定周期中断处理。

18.2.7 1Hz Output

RTC 可选择输出一般精度，较高精度和高精度 3 种 1Hz 时钟。当时钟误差补偿功能有效时输出较高精度的 1Hz 时钟；当使用不同频率的 PCLK 时输出高精度的 1Hz 时钟。需要根据 PCLK 频率配置系统控制寄存器，其中，

一般精度的 1Hz 输出设定如下：(无时钟补偿)

1. 设定 RTC_CR0.START=0，计数停止；
2. RTC 输出引脚设定；
3. RTC_CR0.1HZOE=1，时钟输出许可；
4. 设定 RTC_CR0.START=1，计数开始；
5. 等待 2 个计数周期以上；
6. 1Hz 输出开始。

较高精度的 1Hz 输出设定如下：(低速补偿)

1. 设定 RTC_CR0.START=0，计数停止；
2. RTC 输出引脚设定；
3. RTC_CR0.1HZOE=1，时钟输出许可；
4. 时钟误差补偿寄存器 RTC_COMPEN.CR 补偿数设定；
5. 时钟误差补偿寄存器 RTC_COMPEN.EN=1，误差补偿有效；
6. 设定 RTC_CR0.START=1，计数开始；
7. 等待 2 个计数周期以上；
8. 1Hz 输出开始。

当高精度的 1Hz 输出时，需要在较高精度输出的基础上，为 RTC 提供 4M,6M,8M,12M,16M,20M,24M,32MHz 的高速 PCLK 时钟，输出设定如下：

1. 设定 RTC_CR0.START=0，计数停止；
2. RTC 输出引脚设定；
3. RTC_CR0.1HZOE=1，时钟输出许可；
4. RTC_CR0.1HZSEL=1，选择输出高精度 1Hz 时钟；
5. 配置高速时钟补偿时钟 SYSCtrl1.RTC_FREQ_Adjust
6. 时钟误差补偿寄存器 RTC_COMPEN.CR[8:0] 补偿数设定；
7. 时钟误差补偿寄存器 RTC_COMPEN.EN=1，精度补偿有效；
8. 设定 RTC_CR0.START=1，计数开始；
9. 等待 2 个计数周期以上；
10. 1Hz 输出开始。

18.2.8 Clock Error Compensation

由于外部晶振存在误差，在需要得到高精度的计数结果时，需要对该误差进行补偿。补偿方法分为两种：第一种，基于自身时钟的误差补偿；第二种，基于高速时钟的误差补偿。

基于自身时钟的误差补偿原理与计算：

由于计数器采用 32.768KHz 的时钟计数，如果需要对每秒精度进行补偿时，只能按照 32.768KHz 的整数周期补偿，则每秒补偿的最小单位为 $(1/32768) * 10^6 = 30.5\text{ppm}$ ，无法满足高精度的要求。

那么要在 32.768KHz 的计数时钟下实现精度较高的时钟补偿时，需要在算法上做调整，将最大补偿周期扩大 32 倍。则在只能补偿的最小单位为 30.5ppm 的情况下，平均到每秒的补偿单位变为 $30.5\text{ppm}/32=0.96\text{ppm}$ 。满足了精度较高的时钟补偿要求。而且补偿发生在每 32 秒内比较均匀的范围内。所以，该寄存器中引入了 5 位小数的设定。

例 1：

当默认状态下直接输出 1Hz 时钟，通过测定该时钟的精度，计算补偿目标值。

假设实际测定值为 0.9999888Hz，则：

$$\text{实际发振频率} = 32768 \times 0.9999888 \approx 32767.63$$

$$\begin{aligned}\text{补偿目标值} &= (\text{实际发振频率} - \text{目标频率}) / \text{目标频率} \times 10^6 \\ &= (32767.96 - 32768) / 32768 \times 10^6 \\ &= -11.29\text{ppm}\end{aligned}$$

根据

$$CR[8:0] = \left(\frac{\text{补偿目标值[ppm]} \times 2^{15}}{10^6} \right) \quad +0001.00000 \text{ B}$$

取 2 的补码

如果补偿目标值为 -11.29ppm，计算相应的寄存器值如下：

$$\begin{aligned} CR[8:0] &= (-11.29 \times 215/106) \text{ 取 2 的补码} + 0001.00000 \text{ B} \\ &= (-0.37) \quad \text{取 2 的补码} + 0001.00000 \text{ B} \\ &= 1111.10101 \text{ B} + 0001.00000 \text{ B} \\ &= 0000.10101 \text{ B} \end{aligned}$$

基于高速 24MHz 时钟的误差补偿原理与计算：

该方式的计算方法与基于自身时钟的误差补偿相同。由于引入了 4M-32MHz 高速时钟，本来需要在最多 32 秒内累计的 1/32768 秒误差可分散到每 1 秒，针对每 1 秒进行最小 0.96ppm (23 个 24MHz 时钟周期) 的补偿，实现平均的每秒高精度 1Hz 时钟输出。

18.3 RTC Interrupt

RTC 支持两种中断类型。闹钟中断、定周期中断。闹钟中断与定周期中断共用一个中断信号。

18.3.1 RTC Alarm Interrupt

当 RTC_CR1.ALMIE=1 时，若当前日历时间与秒闹钟寄存器 (RTC_ALMSEC)、分闹钟寄存器 (RTC_ALMMIN)、时闹钟寄存器 (RTC_ALMHOUR)、周闹钟寄存器 (RTC_ALMWEEK) 相等时，触发闹钟中断。

18.3.2 RTC Periodic Interrupt

控制寄存器 1 (RTC_CR1) 的 ALMIE=1 时，选择的周期发生后，触发定周期唤醒中断，由于闹钟和定周期共用中断，通过标志寄存器位来区分。

18.4 RTC Register Description

基地址 0X40001400

表 18-2 RTC 寄存器列表

寄存器	偏移地址	描述
RTC_CR0	0X000	控制寄存器 0
RTC_CR1	0X004	控制寄存器 1
RTC_SEC	0X008	秒计数寄存器
RTC_MIN	0X00C	分计数寄存器
RTC_HOUR	0X010	时计数寄存器
RTC_WEEK	0X014	周计数寄存器
RTC_DAY	0X018	日计数寄存器
RTC_MON	0X01C	月计数寄存器
RTC_YEAR	0X020	年计数寄存器
RTC_ALMMIN	0X024	分闹钟寄存器
RTC_ALMHOUR	0X028	时闹钟寄存器
RTC_ALMWEEK	0X02C	周闹钟寄存器
RTC_COMPEN	0X030	时钟误差补偿寄存器
RTC_ALMSEC	0X034	秒闹钟寄存器

18.4.1 Control Register 0 (RTC_CR0)

*只有上电对该寄存器复位有效

地址偏移量：0x000

复位值 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13:8		7	6	5	4	3						2:0	
Res.	PRDSEL	PRDX	START	HZ1SEL	HZ1OE	Res.	AMPM	PRDS							
	R/W	R/W	R/W	R/W	R/W		R/W	R/W							

位	符号	功能描述
31:15	Reserved	-
14	PRDSEL	0: 使用 PRDS 所设定的周期中断时间间隔 1: 使用 PRDX 所设定的周期中断时间间隔
13:8	PRDX	设置产生周期中断的时间间隔，可设定的范围为 0.5 秒到 32 秒，步进为 0.5 秒。 000000: 0.5 秒 000001: 1 秒 111110: 31.5 秒 111111: 32 秒
7	START	0: 停止 RTC 计数器 1: 使能 RTC 计数器
6	HZ1SEL	0: 普通精度 1Hz 输出 1: 高精度 1Hz 输出
5	HZ1OE	0: 禁止 1Hz 输出 1: 使能 1Hz 输出
4	Reserved	-
3	AMPM	0: 12 小时制 1: 24 小时制
2:0	PRDS	设置产生中断的时间间隔： 000: 不产生周期中断 001: 0.5 秒 010: 1 秒 011: 1 分钟 100: 1 小时 101: 1 天 11x: 1 月 注意：如需要在 START=1 时写入更改周期中断的时间间隔操作步骤如下： step1, 在 NVIC 中关闭 RTC 中断； step2, 更改周期中断的时间间隔；

		step3, 清除 RTC 中断标志; step4, 使能 RTC 中断。
--	--	--

18.4.2 Control Register 1 (RTC_CR1)

*只有上电对该寄存器复位有效

地址偏移量：0x004

复位值 0X00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15:11	10:8	7	6	5	4	3	2	1	0						
Reserved		CKSEL	ALMEN	ALMIE	Res.	ALMF	PRDF	Res.	WAITF	WAIT					
		R/W	R/W	R/W		R/W	R/W	Res.	R/W	R/W					

位	符号	功能描述
31:11	Reserved	-
10:8	CKSEL	RTC 时钟选择 00x: XTL 32.768k 01x: RCL 32k 100: XTH/128 (晶振为 4M 时选择此项) 101: XTH/256 (晶振为 8M 时选择此项) 110: XTH/512 (晶振为 16M 时选择此项) 111: XTH/1024 (晶振为 32M 时选择此项)
7	ALMEN	0: 禁止闹钟 1: 使能闹钟 注意: 在 START=1 日历计数过程中并且 ALMIE=1 中断许可的情况下使能 ALMEN 时, 为防止误动作请将系统中断关闭。使能后请将 ALMF 标志位清除。
6	ALMIE	0: 禁止闹钟中断 1: 使能闹钟中断
5	Reserved	-
4	ALMF	0: 未发生闹钟中断 1: 已发生闹钟中断 注意: 该位仅在 ALMEN=1 时有效。闹钟匹配时, 32.7689KHz 一个时钟后置 1。写 0 时清除标志, 写 1 无效。
3	PRDF	0: 未发生周期中断 1: 已发生周期中断 注意: 发生周期中断后, 该位置 1。写 0 时清除该标志, 写 1 无效。
2	Reserved	-
1	WAITF	0: 非写入/读出状态 1: 写入/读出状态 注意: WAIT 位设定是否有效标志。在写入/读出前请确认该位是否为“1”。计数过程中, 在 WAIT 位清“0”后等待写入完成后该位才清“0”。
0	WAIT	0: 正常计数模式 1: 写入/读出模式 注意: 在写入/读出时请将该位置“1”, 由于计数器在连续计数, 请在 1 秒的时间内完成写入/读

		出操作并将该位清“0”。
--	--	--------------

18.4.3 Seconds Count Register (RTC_SEC)

地址偏移量: 0x008

复位值: 不定

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

位	符号	功能描述
31:7	Reserved	-
6:4	SECH	秒计数十位值
3:0	SECL	秒计数个位值

表示 0-59 秒，采用十进制计数。请写入十进制 0-59 的 BCD 码，写入错误值时，写入值将被忽略。

18.4.4 Minute Count Register (RTC_MIN)

地址偏移量: 0x00C

复位值: 不定

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

位	符号	功能描述
31:7	Reserved	-
6:4	MINH	分计数十位值
3:0	MINL	分计数个位值

表示 0-59 分，采用十进制计数。请写入十进制 0-59 的 BCD 码，写入错误值时，写入值将被忽略。

18.4.5 Time counter register (RTC_HOUR)

地址偏移量: 0x010

复位值: 不定

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										HOURH	HOURL				R/W

位	符号	功能描述
31:6	Reserved	-
5:4	HOURH	时计数十位值
3:0	HOURL	时计数个位值

24 小时时制时，表示 0-23 小时。12 小时时制时，b5=0 表示 AM，则 01:12 表示上午；b5=1 表示 PM，则 21:32 表示下午。

请根据控制为 AMPM 的值，设定正确十进制的 0:23 或者 01:12,21:32 的 BCD 码。写入超出范围的值将被忽略。

具体时间表示参考下表：

24 小时时制	AMPM=1	12 小时时制	AMPM=0
时间	寄存器表示	时间	寄存器表示
00 时	00H	AM 12 时	12H
01 时	01H	AM 01 时	01H
02 时	02H	AM 02 时	02H
03 时	03H	AM 03 时	03H
04 时	04H	AM 04 时	04H
05 时	05H	AM 05 时	05H
06 时	06H	AM 06 时	06H
07 时	07H	AM 07 时	07H
08 时	08H	AM 08 时	08H
09 时	09H	AM 09 时	09H
10 时	10H	AM 10 时	10H
11 时	11H	AM 11 时	11H
12 时	12H	PM 12 时	32H
13 时	13H	PM 01 时	21H
14 时	14H	PM 02 时	22H
15 时	15H	PM 03 时	23H
16 时	16H	PM 04 时	24H
17 时	17H	PM 05 时	25H
18 时	18H	PM 06 时	26H
19 时	19H	PM 07 时	27H
20 时	20H	PM 08 时	28H
21 时	21H	PM 09 时	29H
22 时	22H	PM 10 时	30H
23 时	23H	PM 11 时	31H

18.4.6 Day Count Register (RTC_DAY)

地址偏移量: 0x018

复位值: 不定

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

位	符号	功能描述
31:6	Reserved	-
5:4	DAYH	日计数十位值
3:0	DAYL	日计数个位值

十进制表示 1:31 日，自动计算闰年和月份。具体表示如下：

月份	日计数表示
2月（普通年）	01:28
2月（闰年）	01:29
4、6、9、11月	01:30
1、3、5、7、8、10、12月	01:31

18.4.7 Week Count Register (RTC_WEEK)

地址偏移量: 0x014

复位值: 不定

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WEEK R/W

位	符号	功能描述
31:3	Reserved	-
2:0	WEEK	周计数值

十进制 0:6 表示周日:周六。请写入正确的十进制 0:6 的 BCD 码，写入其他值，将被忽略。周计数值对应关系如下：

周	周计数表示
周日	00H
周一	01H
周二	02H
周三	03H
周四	04H
周五	05H
周六	06H

18.4.8 Month Count Register (RTC_MON)

地址偏移量: 0x01C

复位值: 不定

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

位	符号	功能描述
31:5	Reserved	-
4:0	MON	月计数值

十进制 1:12 表示 1:12 月。请写入正确的十进制 1:12 的 BCD 码，写入其他值，将被忽略。

18.4.9 Year Count Register (RTC_YEAR)

地址偏移量: 0x020

复位值: 不定

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

位	符号	功能描述
31:8	Reserved	-
7:4	YEARH	年计数十位值
3:0	YEARL	年个位计数值

十进制 0:99 表示 0:99 年。根据月进位计数。自动计算闰年如：00、04、08、...、92、96 等。请写入正确的十进制年计数值，写入错误值将被忽略。

18.4.10 Second Alarm Register (RTC_ALMSEC)

地址偏移量: 0x034

复位值: 不定

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

位	符号	功能描述
31:7	Reserved	-
6:4	ALMSECH	秒闹钟匹配值十位
3:0	ALMSECL	秒闹钟匹配值个位

请设定十进制 0:59 的 BCD 码。写入其他值，不会发生闹钟匹配。

18.4.11 Minute Alarm Register (RTC_ALMMIN)

地址偏移量: 0x024

复位值: 不定

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

位	符号	功能描述
31:7	Reserved	-
6:4	ALMMINH	分闹钟匹配值十位
3:0	ALMMINL	分闹钟匹配值个位

请设定十进制 0:59 的 BCD 码。写入其他值，不会发生闹钟匹配。

18.4.12 Alarm Clock Register (RTC_ALMHOUR)

地址偏移量: 0x028

复位值: 不定

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

位	符号	功能描述
31:6	Reserved	-
5:4	ALMHOURH	时闹钟十位匹配值
3:0	ALMHOURL	时闹钟个位匹配值

请根据时制设定正确的闹钟匹配值，否则不会发生时闹钟匹配。

18.4.13 Weekly Alarm Register (RTC_ALMWEEK)

地址偏移量: 0x02C

复位值: 不定

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

位	符号	功能描述
31:7	Reserved	-
6:0	ALMWEEK	周闹钟匹配值。 b0:b6 分别对应周日:周六，对应为置“1”时，代表每周该日闹钟有效。如，b0=1, b5=1 代表周日和周五闹钟设定有效。

请根据时制设定正确的闹钟匹配值，否则不会发生时闹钟匹配。

18.4.14 Clock Error Compensation Register (RTC_COMPEN)

地址偏移量: 0x030

*只有上电对该寄存器复位有效, 复位值: 0x00000020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN R/W	Reserved								CR R/W						

位	符号	功能描述																																																																																																																									
31:16	Reserved	-																																																																																																																									
15	EN	补偿使能 0: 禁止时钟误差补偿 1: 使能时钟误差补偿																																																																																																																									
14:9	Reserved	-																																																																																																																									
8:0	CR	补偿值 通过补偿值设定, 可针对每秒进行 +/-0.96ppm 的精度补偿。补偿值为 9 位带小数点的 2 的补码, 后 5 位为小数部分。可补偿范围 274.6ppm:212.6ppm。最小微分误差 +/-0.48ppm。 最小分辨率 0.96ppm。具体补偿精度请参考下表: <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="10">补偿值设定</th> <th>补偿数</th> </tr> <tr> <th>EN</th> <th colspan="9">CR[8:0]</th> <th></th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> <td>-274.6ppm</td> </tr> <tr> <td></td> <td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td> <td>-273.7ppm</td> </tr> <tr> <td>:</td> <td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td> <td>:</td> </tr> <tr> <td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td> <td>-0.95ppm</td> </tr> <tr> <td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> <td>0ppm</td> </tr> <tr> <td>:</td> <td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td> <td>:</td> </tr> <tr> <td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td> <td>+211.7ppm</td> </tr> <tr> <td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td> <td>+212.6ppm</td> </tr> <tr> <td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td> <td>无补偿</td> </tr> </tbody> </table>	补偿值设定										补偿数	EN	CR[8:0]										1	1	0	0	0	0	0	0	0	0	-274.6ppm		1	0	0	0	0	0	0	0	1	-273.7ppm	:	:	:	:	:	:	:	:	:	:	:	0	0	0	0	1	1	1	1	1	1	-0.95ppm	0	0	0	1	0	0	0	0	0	0	0ppm	:	:	:	:	:	:	:	:	:	:	:	0	1	1	1	1	1	1	1	1	0	+211.7ppm	0	1	1	1	1	1	1	1	1	1	+212.6ppm	0	X	X	X	X	X	X	X	X	X	无补偿
补偿值设定										补偿数																																																																																																																	
EN	CR[8:0]																																																																																																																										
1	1	0	0	0	0	0	0	0	0	-274.6ppm																																																																																																																	
	1	0	0	0	0	0	0	0	1	-273.7ppm																																																																																																																	
:	:	:	:	:	:	:	:	:	:	:																																																																																																																	
0	0	0	0	1	1	1	1	1	1	-0.95ppm																																																																																																																	
0	0	0	1	0	0	0	0	0	0	0ppm																																																																																																																	
:	:	:	:	:	:	:	:	:	:	:																																																																																																																	
0	1	1	1	1	1	1	1	1	0	+211.7ppm																																																																																																																	
0	1	1	1	1	1	1	1	1	1	+212.6ppm																																																																																																																	
0	X	X	X	X	X	X	X	X	X	无补偿																																																																																																																	

补偿原理说明与计算:

由于计数器采用 32.768KHz 的时钟计数, 如果需要对每秒精度进行补偿时, 只能按照 32.768KHz 的整数周期补偿, 则每秒补偿的最小单位为 $(1/32768) * 10^6 = 30.5\text{ppm}$, 无法满足高精度的要求。

那么要在 32.768KHz 的计数时钟下实现精度较高的时钟补偿时, 需要在算法上做调整, 将最大补偿周期扩大 32 倍。则在只能补偿的最小单位为 30.5ppm 的情况下, 平均每秒的补偿单位变为为 $30.5\text{ppm}/32=0.96\text{ppm}$ 。满足了精度较高的时钟补偿要求。而且补偿发生在每 32 秒内比较均匀的范围内。所以, 该寄存器中引入了 5 位小数的设定。

设定值计算如下:

$$CR[8:0] = \left(\frac{\text{补偿目标值[ppm]} \times 2^{15}}{10^6} \right) \text{ 取 2 的补码} + 0001.00000B$$

如果补偿目标值为 +20.6ppm，计算相应的寄存器值如下：

$$\begin{aligned} CR[8:0] &= (20.6 \times 2^{15}/10^6) \text{ 取 2 的补码} + 0001.00000B \\ &= (0.6651904) \text{ 取 2 的补码} + 0001.00000B \\ &= 0000.10101B + 0001.00000B \\ &= 0001.10101B \end{aligned}$$

如果补偿目标值为-20.6ppm，计算相应的寄存器值如下：

$$\begin{aligned} CR[8:0] &= (-20.6 \times 2^{15}/10^6) \text{ 取 2 的补码} + 0001.00000B \\ &= (-0.6651904) \text{ 取 2 的补码} + 0001.00000B \\ &= 1111.01011B + 0001.00000B \\ &= 0000.01011B \end{aligned}$$

19 Watchdog Timer (WDT)

19.1 WDT Introduction

WDT 可用来检测和解决由软件错误引起的故障。当 WDT 计数器达到设定的溢出时间后，会触发中断或产生系统复位。WDT 由专用的 10KHz 片内振荡器驱动。

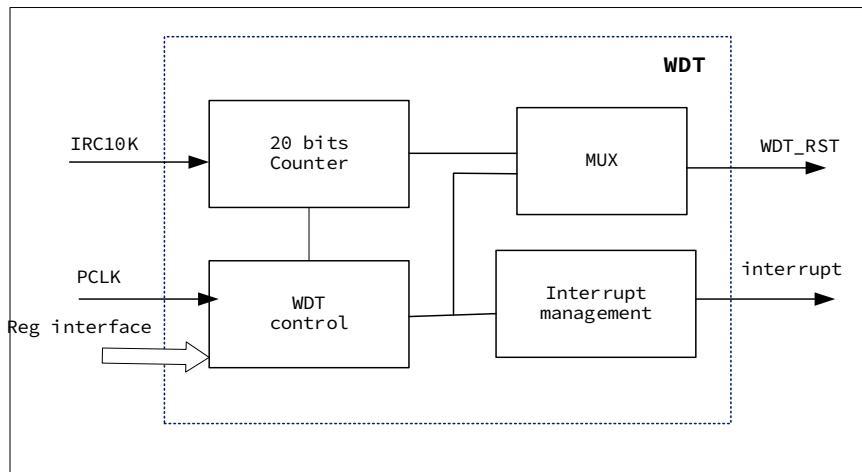


图 19-1 WDT 整体框图

19.2 WDT Function Description

- 20Bit 自由运行的递增计数器，溢出时间可配置为 1.6ms - 50s。
- 溢出后的动作可配置为中断或复位。
- WDT 时钟由独立的 RC 振荡器提供，可在 Sleep 和 DeepSleep 模式下工作。
- WDTCON 寄存器只有在 WDT 未被启动时才能修改，以防止启动后无意之间修改 WDT 的配置。

19.2.1 WDT overflow interrupt

在本模式下，WDT 将按所设定的时间周期性地产生中断。在中断服务程序中需要清除 WDT 中断标志。

配置方法如下所示：

- Step1：配置 WDT_CON. WOV，选择 WDT 计时溢出时间。
- Step2：设置 WDT_CON. WINT_EN 为 1，选择 WDT 溢出后产生中断。
- Step3：使能 NVIC 中断向量表中的 WDT 中断。
- Step4：向 WDT_RST 寄存器依次写入 0x1E、0xE1，启动 WDT 定时器。
- Step5：在中断服务程序中向 WDT_RST 寄存器依次写入 0x1E、0xE1 以清除中断标志。

19.2.2 WDT overflow reset

在本模式下，WDT 计数器溢出后会产生 Reset 信号，该信号会复位 MCU。用户程序需要在 WDT 溢出前清零 WDT 计数器，从而避免产生 WDT 复位。

配置方法如下所示：

- Step1：配置 WDT_CON. WOV，选择 WDT 计数器溢出时间。
- Step2：设置 WDT_CON. WINT_EN 为 0，选择 WDT 溢出后产生复位。
- Step3：向 WDT_RST 寄存器依次写入 0x1E、0xE1，启动 WDT 定时器。
- Step4：在 WDT 溢出前向 WDT_RST 寄存器依次写入 0x1E、0xE1 以清零 WDT 计数器。

注意：

- 由于 WDT 振荡器是低精度的 RC 振荡器，强烈建议在 WDT 计数器计数值到达溢出值的一半之前对 WDT 进行清零。

19.3 WDT Register Description

基地址 0X40000F00

表 19-1 WDT 寄存器列表

寄存器	偏移地址	描述
WDT_RST	0X080	WDT 清除控制寄存器
WDT_CON	0X084	WDT 控制寄存器

19.3.1 WDT Clear Control Register (WDT_RST)

偏移地址: 0x080

复位值: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved				WDTRST				WO

位	符号	描述
31:8	Reserved	保留位, 读为 0
7:0	WDTRST	看门狗启动/清零控制 当看门狗未启动时, 向该寄存器依次写入 0x1E、0xE1, 启动 WDT 定时器。 当看门狗已启动时, 向该寄存器依次写入 0x1E、0xE1, 清零 WDT 定时器及中断标志。

19.3.2 WDT_CON Register

偏移地址: 0x084

复位值: 0x0000 000F

注: 该寄存器只有在 WDT 未运行时才可以写入。

	31:16	15:8	7	6	5	4	3	2	1	0
Reserved	WCNTL	WDTINT	Res.	WINT_EN	WDTR			WOV		
	RO	RO		RW	RO			RW		

位	符号	描述																
31:16	Reserved	保留位, 读为 0																
15:8	WCNTL	WDT 计数器低 8 位																
7	WDTINT	WDT 中断标志 1: 已发生 WDT 中断, 向 WDT_RST 寄存器依次写入 0x1E、0xE1 以清除该中断标志。 0: 未发生 WDT 中断。																
5	WINT_EN	WDT 溢出后的动作配置 1: WDT 溢出后产生中断。 0: WDT 溢出后产生复位。																
4	WDTR	WDT 运行标志 1: WDT 正在运行 0: WDT 停止																
3:0	WOV[3:0]	WDT 计时溢出时间配置 <table border="1" data-bbox="412 1201 1119 1560"> <tbody> <tr><td>0000: 1.6ms</td><td>1000: 500ms</td></tr> <tr><td>0001: 3.2ms</td><td>1001: 820ms</td></tr> <tr><td>0010: 6.4ms</td><td>1010: 1.64s</td></tr> <tr><td>0011: 13ms</td><td>1011: 3.28s</td></tr> <tr><td>0100: 26ms</td><td>1100: 6.55s</td></tr> <tr><td>0101: 51ms</td><td>1101: 13.1s</td></tr> <tr><td>0110: 102ms</td><td>1110: 26.2s</td></tr> <tr><td>0111: 205ms</td><td>1111: 52.4s</td></tr> </tbody> </table>	0000: 1.6ms	1000: 500ms	0001: 3.2ms	1001: 820ms	0010: 6.4ms	1010: 1.64s	0011: 13ms	1011: 3.28s	0100: 26ms	1100: 6.55s	0101: 51ms	1101: 13.1s	0110: 102ms	1110: 26.2s	0111: 205ms	1111: 52.4s
0000: 1.6ms	1000: 500ms																	
0001: 3.2ms	1001: 820ms																	
0010: 6.4ms	1010: 1.64s																	
0011: 13ms	1011: 3.28s																	
0100: 26ms	1100: 6.55s																	
0101: 51ms	1101: 13.1s																	
0110: 102ms	1110: 26.2s																	
0111: 205ms	1111: 52.4s																	

20 Pulse counter (PCNT)

20.1 Pulse Counter Introduction

脉冲计数器（PCNT）模块可以对输入的脉冲进行计数，支持三种脉冲模式：单通道脉冲、双通道正交脉冲、双通道非交脉冲。无需软件参与，可在低功耗模式下正确计数。

20.2 Pulse Counter Main Characteristics

脉冲计数器支持以下特性：

- 支持重载功能的 16 bit 计数器
- 单通道脉冲计数
- 双通道非交脉冲计数
- 双通道正交脉冲计数，不失码
- 加/减计数溢出中断
- 脉冲超时中断
- 4 种解码错误中断，非交脉冲模式
- 1 种方向改变中断，正交脉冲模式
- 多级脉冲宽度滤波
- 输入脉冲极性可配置
- 支持低功耗模式计数
- 支持唤醒低功耗模式下 MCU
- 支持任意脉冲沿间距不小于 1 个计数时钟周期
- 支持 IO 输入或者 VC 输出作为脉冲输入信号
- 支持双路脉冲信号整形之后通过 IO 观测输出 PCNT_S0FO/PCNT_S1FO
- 具备低功耗模式下自动定时唤醒功能，最大定时达 1024 秒

20.3 Pulse Counter Function Description

20.3.1 Overall Block Diagram

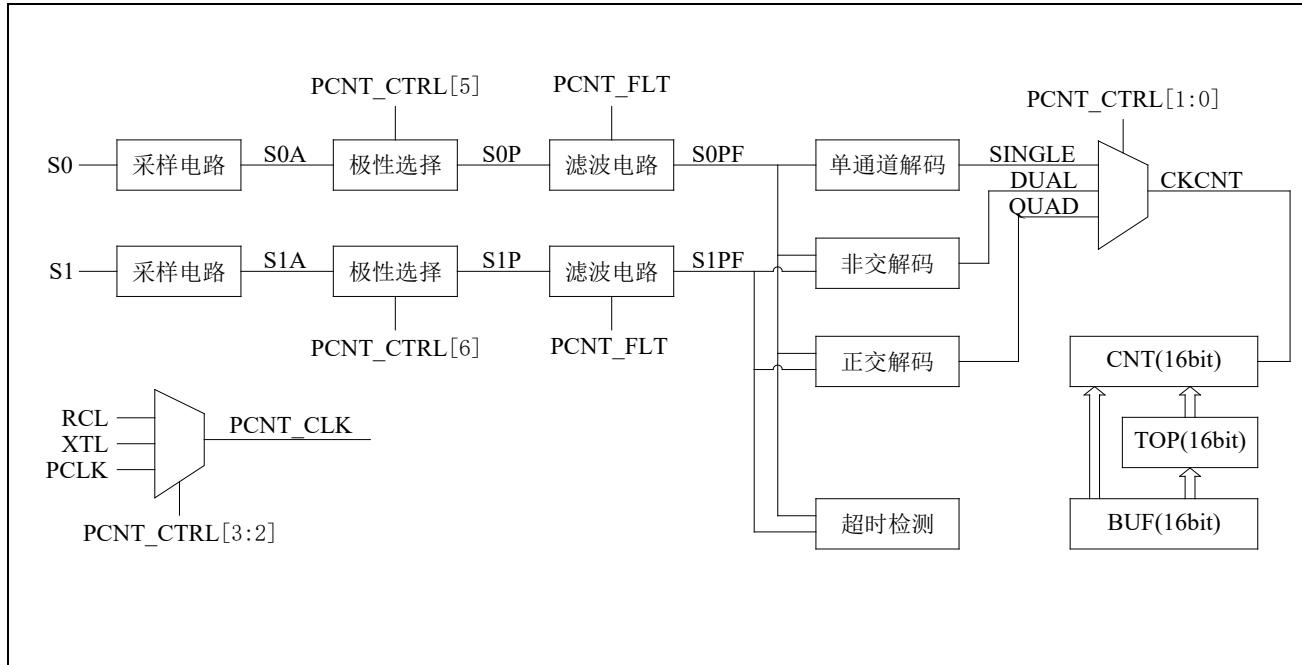


图 20-1 整体框图

20.3.2 Signal Description

以下以输入信号 S0 为例说明信号在计数器模块中的处理流程。

- S0 信号经采样模块后输出 S0A 信号，采样时钟为 PCNT_CLK。
- S0A 信号经极性选择模块后输出 SOP 信号。
- SOP 信号经脉冲宽度滤波模块后输出 SOPF 信号，滤波模块的时钟为分频后的 PCNT_CLK。
- SOPF 信号经单通道解码模块、双通道非交解码模块、双通道正交解码模块解码后输出相应的 SIGNAL、DUAL、QUAD 信号。
- SIGNAL、DUAL、QUAD 经多路选择器后输出 CKCNT 信号以驱动 CNT 模块进行计数。

20.3.3 Counting Modes

本模块支持 3 种计数模式：单通道脉冲、双通道正交脉冲、双通道非交脉冲。

20.3.3.1 Single Channel Pulse Counting Mode (Single Mode)

当配置 PCNT_CTRL.Mode 为 0x00 或 0x01 时，脉冲计数器工作于单通道脉冲计数模式。该模式下解码模块只对 S0 脉冲信号进行计数。当 PCNT_CLK 时钟采样到 SOPF 信号的下降沿时，计数器按 PCNT_CTRL.DIR 的配置进行一次递增或递减操作。

计数器的计数范围为 0x00 至计数上阈值(PCNT_TOP)。

在加计数状态，当计数值等于 PCNT_TOP 且采样到 SOPF 信号的下降沿时，计数器发生上溢出，计数值回到 0 并置位 PCNT_IFR.OV，该中断标志需要使用软件进行清零。下图为 PCNT 加计数的时序图，图中 PCNT_TOP 的值为 99。

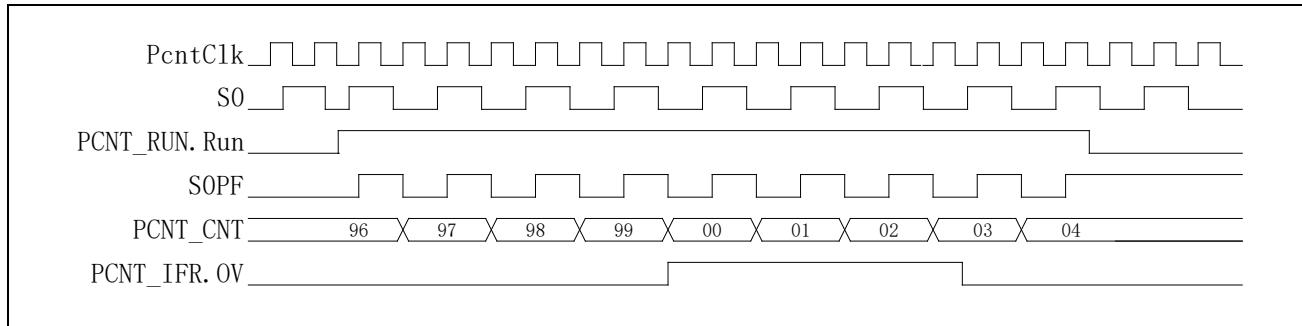


图 20-2 单通道脉冲记数模式加计数波形

在减计数状态，当计数值等于 0x00 且采样到 SOPF 信号的下降沿时，计数器发生下溢出，计数值回到 PCNT_TOP 并置位 PCNT_IFR.UF，该中断标志需要使用软件进行清零。下图为 PCNT 减计数的时序图，图中 PCNT_TOP 的值为 99。

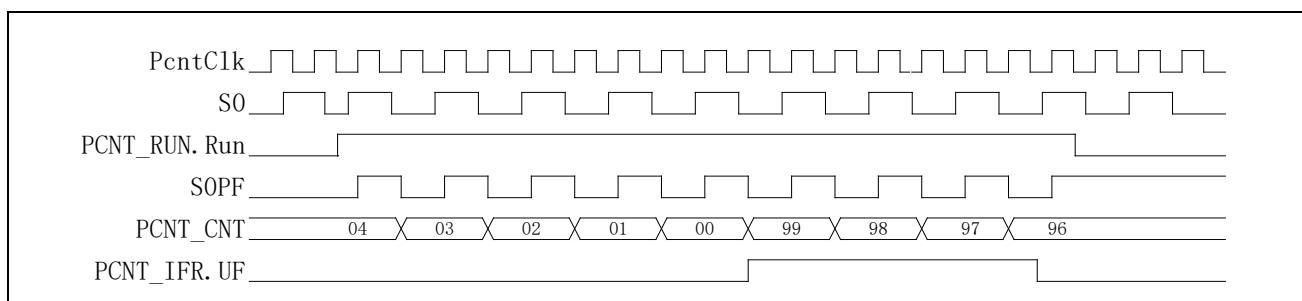


图 20-3 单通道脉冲计数模式减计数波形

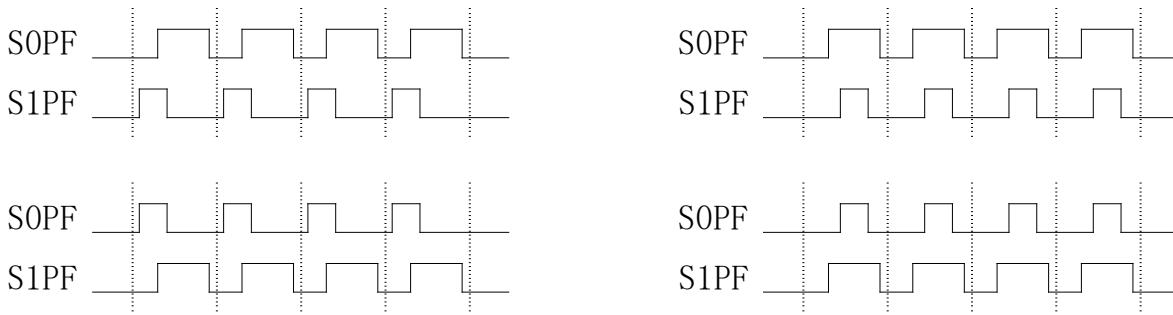
20.3.3.2 Dual Channel Non-intersecting Pulse Counting Mode (Dual Mode)

当配置 PCNT_CTRL.Mode 为 0x02 时，脉冲计数器工作于双通道非交脉冲计数模式。该模式下解码模块对 S0 和 S1 脉冲信号进行解码计数。当 SOPF 和 S1PF 依次出现两个正脉冲，则计数器按 PCNT_CTRL.DIR 的配置进行一次递增或递减操作。

解码模块能正确计数的两种波形如下所示，SOPF 与 S1PF 没有同时为高的情况。



解码模块不能正确计数的几种波形如下所示，SOPF 与 S1PF 出现同时为高的情况。



计数器的计数范围为 0x00 至计数上阈值(PCNT_TOP)。

在加计数状态，当计数值等于 PCNT_TOP 且 SOPF、S1PF 依次出现两个正脉冲时，计数器发生上溢出，计数值回到 0 并置位 PCNT_IFR.OF，该中断标志需要使用软件进行清零。下图为 PCNT 加计数的时序图，图中 PCNT_TOP 的值为 99。

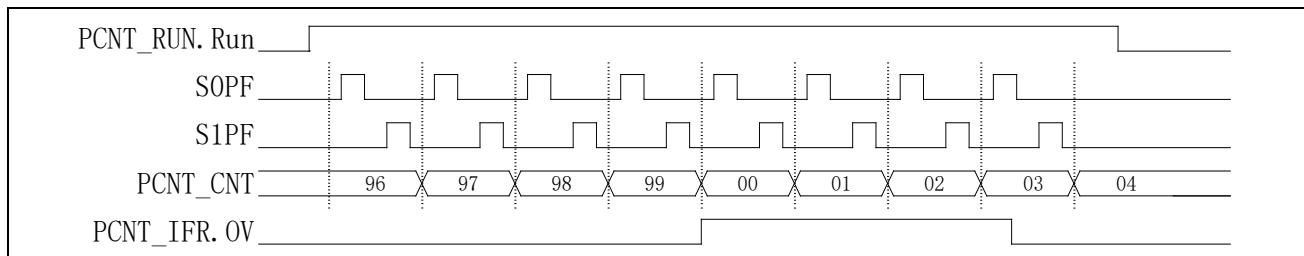


图 20-4 双通道非交脉冲计数模式加计数波形

在减计数状态，当计数值等于 0x00 且 SOPF、S1PF 依次出现两个正脉冲时，计数器发生下溢出，计数值回到 PCNT_TOP 并置位 PCNT_IFR.UF，该中断标志需要使用软件进行清零。下图为 PCNT 减计数的时序图，图中 PCNT_TOP 的值为 99。

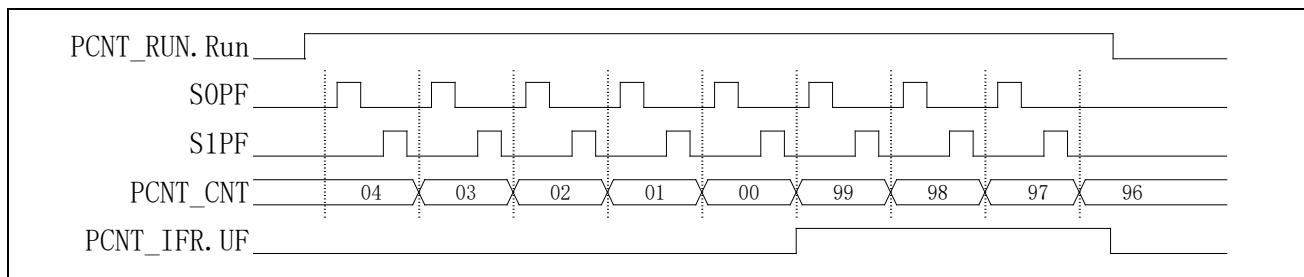


图 20-5 双通道非交脉冲计数模式减计数波形

20.3.3.3 Dual-channel quadrature pulse counting mode (Quad Mode)

当配置 PCNT_CTRL.Mode 为 0x03 时，脉冲计数器工作于双通道正交脉冲计数模式。该模式下解码模块需要 S0 和 S1 两路脉冲信号进行解码计数。计数器根据 SOPF 和 S1PF 脉冲的相位关系自动判断计数的方向，物理设备任意正转、反转均能正确计数。当 SOPF 和 S1PF 完成一个周期的变化时，计数器按计数方向进行一次递增或递减操作。

解码模块判断计数方向的波形如下所示：



计数器的计数范围为 0x00 至计数上阈值(PCNT_TOP)。

在加计数状态,当计数值等于 PCNT_TOP 且 SOPF 和 S1PF 完成一个周期的变化时,计数器发生上溢出,计数值回到 0 并置位 PCNT_IFR.OV, 该中断标志需要使用软件进行清零。下图为 PCNT 加计数的时序图, 图中 PCNT_TOP 的值为 99。

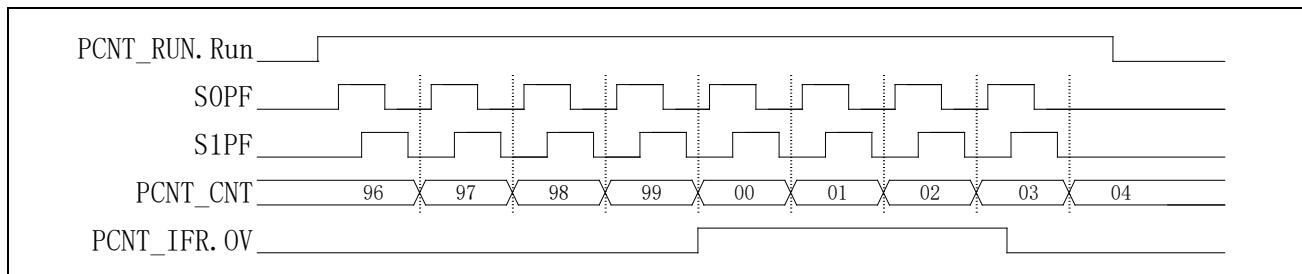


图 20-6 双通道正交脉冲计数模式加计数波形

在减计数状态,当计数值等于 0x00 且 SOPF 和 S1PF 完成一个周期的变化时,计数器发生下溢出,计数值回到 PCNT_TOP 并置位 PCNT_IFR.UF, 该中断标志需要使用软件进行清零。下图为 PCNT 减计数的时序图, 图中 PCNT_TOP 的值为 99。

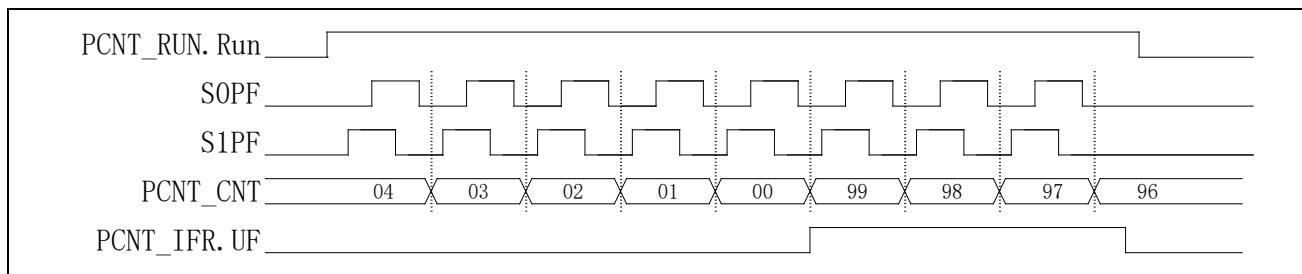


图 20-7 双通道正交脉冲计数模式减计数波形

20.3.4 Pulse Width Filtering

该脉冲宽度滤波器使用计数时钟分频作为滤波时钟,并以此计数从而实现脉冲去抖动目的。其中滤波时钟的分频系数从 1~8096 倍配置寄存器可选 (FLT.ClkDiv), 保持电平超过 (FLT.DebTop) (1~7) 个滤波时钟视为非抖动, 可通过滤波器。

该滤波功能可由配置寄存器 (FLT.EN) 打开或关闭。

同时仅当启动 PCNT 之后 (Run=1), 滤波计数器才启动。

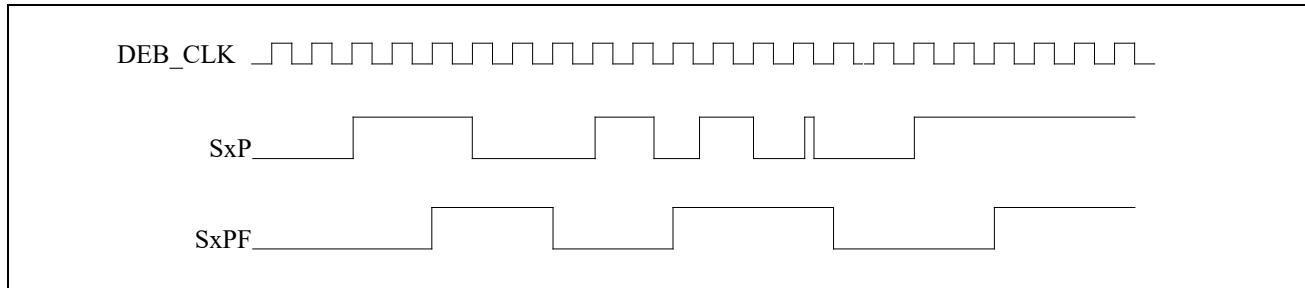


图 20-8 脉冲宽度滤波波形

图示为 $\text{FLT.DebTop} = 2$ 时的滤波波形。

$$f_{\text{DEB_CLK}} = \frac{f_{\text{PCNT_CLK}}}{(\text{FLT.ClkDiv}+1)}$$

20.3.5 Timeout

打开超时功能 (TOCR.EN) 将使用滤波时钟 (见 20.3.4 脉冲宽度滤波) 对脉冲高电平时间由 0 开始进行加计数，采样到低电平将使计数器回到 0，若一个高电平计数值达到 (TOCR.TH) 将产生中断标志位 IF_TO。

超时功能使用滤波时钟，需配置分频系数 (FLT.ClkDiv)，但不依赖于是否打开滤波使能 (FLT.EN)。

同时仅当启动 PCNT 之后 (Run=1)，超时计数器才启动。

溢出时间计算公式如下：

$$T_{\text{TO}} = \frac{(\text{FLT.ClkDiv}+1) \times \text{TOCR.TH}}{f_{\text{PCNT_CLK}}}$$

20.3.6 Pulse input selection

相较于 HC32L130 / HC32L136 系列只能选择 IO 的输入作为 PCNT 的脉冲输入，在本系列中增加了 VC 的输出作为 PCNT 的脉冲输入，通过 GPIO_PCNT 寄存器来进行选择，详见 GPIO 章节端口辅助寄存器相关描述。

20.3.7 Debug Observation Output

为确认进入脉冲解码模块的信号符合预期，可以通过寄存器选择三组整形后的脉冲通过 IO 输出作为 PCNT_S0FO/PCNT_S1FO 观测信号，分别是：

- S0A/S1A
- S0P/S1P
- S0PF/S1PF

通过观测 PCNT_S0FO/PCNT_S1FO 输出脉冲信号，可以更直观的调整极性选择和脉冲宽度滤波器的配置，避免解码电路的输入信号不符合预期。

20.3.8 Auto-wake-up timer in low power mode

PCNT 通过适当的配置，可实现低功耗模式下自动唤醒，其最长定时周期为 1024 秒。

具体操作流程如下：

- Step1：设置 GPIO_PCNT 寄存器为 0，选通 IO 作为 PCNT 的输入源。
- Step2：确保端口功能选择寄存器 (Pxx_SEL) 不要配置任何端口为 PCNT_S0。
- Step3：设置 CTRL.Mode 为 0，选择单通道模式。
- Step4：设置 CTRL.S0P 为 1，对 S0 输入通道选择极性取反。
- Step5：配置 CTRL.ClkSel，选择 XTL 或者 RCL 作为计数器时钟。
- Step6：配置 FLT.ClkDiv，选择滤波器的时钟分频系数。
- Step7：配置 TOCR.TH，选择超时定时器的定时周期。
- Step8：设置 TOCR.EN 为 1，使能超时定时器。
- Step9：使能 PCNT 中断向量表。
- Step10：向 ICR 写 0，清除 PCNT 所有中断标志
- Step11：设置 IEN.TO 为 1，使能超时中断。
- Step12：设置 RUN 为 1，启动 PCNT。
- Step13：进入低功耗模式，等待 PCNT 超时中断唤醒 MCU。
- Step14：在中断服务程序中，依次执行以下步骤：
 - a. 设置 RUN 为 0
 - b. 向 ICR.TO 写 0，清除超时中断标志
 - c. 设置 PCNT_IEN.TO 为 0
 - d. 执行用户需要的执行的功能
 - e. 查询直到 PCNT_RUN 为 0
 - f. 查询直到 PCNT_IFR.TO 为 0
 - g. 退出中断服务程序
- Step15：如需再次启动定时器，重复步骤 11~14。

20.4 PCNT Register Description

寄存器列表

基地址：0x40005400

表 20-1 寄存器列表

偏移量	寄存器名称	访问	同步/异步	寄存器描述
0x00	PCNT_RUN	RW	异步	PCNT 启动寄存器
0x04	PCNT_CTRL	RW	异步长信号	PCNT 控制寄存器
0x08	PCNT_FLT	RW	异步长信号	PCNT 滤波控制寄存器
0x0c	PCNT_TOCR	RW	异步长信号	PCNT 超时控制寄存器
0x10	PCNT_CMD	WO1	异步	PCNT 命令寄存器
0x14	PCNT_SR1	RO	同步	PCNT 状态寄存器 1
0x18	PCNT_CNT	RO	同步	PCNT 计数寄存器
0x1c	PCNT_TOP	RO	同步	PCNT 计数溢出寄存器
0x20	PCNT_BUF	RW	异步	PCNT 计数溢出缓存寄存器
0x24	PCNT_IFR	RO	同步	PCNT 中断标识寄存器
0x28	PCNT_ICR	WO0	异步	PCNT 中断清除寄存器
0x2c	PCNT_IEN	RW	异步长信号	PCNT 中断使能寄存器
0x30	PCNT_SR2	RO	异步	PCNT 状态寄存器 2
0x34	PCNT_DBG	RW	异步长信号	PCNT PCNT_S0FO/PCNT_S1FO 输出选择寄存器

20.4.1 PCNT Start Register (PCNT_RUN)

地址偏移量：0x00

复位值：0x0000 0000

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

位	标记	功能描述
31:1	Reserved	
0	Run	PCNT 启动/停止控制。 0: 停止 1: 启动

注意：

- 当前模块将 *pclk* 时钟域的配置寄存器视作长信号，未进行到 *pcnt_clk* 时钟域的同步，所以当 *Run=1* 启动 PCNT 后，配置寄存器**不允许**进行修改以避免发生未知错误。所有配置寄存器的修改务必在 PCNT 停止的状态 (*run=0*) 下进行。
- 对 *Run* 位写入一个值后，需要经过一个同步的过程才可启动/停止 PCNT 计数，在同步未完成之前读取 *Run* 位将返回写入之前的 *Run* 状态值。所以如果要停止 PCNT 之后再启动 PCNT，需对 *Run* 位写入 0，查询到 *Run* 位=0 之后才可继续后续其他操作（包括对 *Run* 位写入 1 以重新启动 PCNT）。

20.4.2 PCNT Control Register (PCNT_CTRL)

地址偏移量: 0x04

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								S1P	S0P	DIR	ClkSel	Mode			
								rw	rw	rw	rw				rw

位	标记	功能描述
31:7	Reserved	
6	S1P	S1 通道输入极性选择 0: 不取反 1: 取反
5	S0P	S0 通道输入极性选择 0: 不取反 1: 取反
4	DIR	计数方向选择 0: 加计数 1: 减计数 注: 仅对单通道模式和双通道非交模式有效。
3:2	ClkSel	PCNT_CLK 计数时钟选择 0: PCLK 1: PCLK 2: XTL 3: RCL
1:0	Mode	脉冲计数模式选择 0: 单通道脉冲计数模式 1: 单通道脉冲计数模式 2: 双通道非交脉冲计数模式 3: 双通道正交脉冲计数模式

20.4.3 PCNT Filter Control Register (PCNT_FLT)

地址偏移量：0x08

复位值：0x00002000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Reserved															EN	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	RW
DebTop															ClkDiv	
rw															rw	

位	标记	功能描述
31:17	Reserved	
16	EN	脉冲宽度滤波器使能控制。 0: 不使能 1: 使能
15:13	DebTop	计数器阈值 使用滤波时钟对输入脉冲连续采样，当连续电平采样计数值达到阈值则视为正常脉冲，可通过滤波器。 0: 为非法值
12:0	ClkDiv	滤波时钟分频系数 系数 = ClkDiv + 1

20.4.4 PCNT Timeout Control Register (PCNT_TOCR)

地址偏移量：0x0c

复位值：0x00000FFF

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

位	标记	功能描述
31:17	Reserved	
16	EN	超时功能使能控制。 0: 不使能 1: 使能
15:12	Reserved	
11:0	TH	超时阈值。 当脉冲高电平被滤波时钟连续采样计数达到阈值，产生超时中断标识，计数器重新开始计数。

20.4.5 PCNT Command Register (PCNT_CMD)

地址偏移量: 0x10

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
B2C B2T T2C wo1 wo1 wo1															

位	标记	功能描述
31:3	Reserved	
2	B2C	写 1，立即将 BUF 中的值同步到 CNT。 同步过程中 PCNT_SR2 的相应位为 1，此时不应再对 BUF 进行写操作，不应再对 CNT 进行读操作。
1	B2T	写 1，立即将 BUF 中的值同步到 TOP。 同步过程中 PCNT_SR2 的相应位为 1，此时不应再对 BUF 进行写操作，不应再对 TOP 进行读操作。
0	T2C	写 1，立即将 TOP 中的值同步到 CNT。 同步过程中 PCNT_SR2 的相应位为 1，此时不应再对 B2T 写 1，不应再对 CNT 进行读操作。

20.4.6 PCNT Status Register 1 (PCNT_SR1)

地址偏移量: 0x14

复位值: 0x0000 0000

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

位	标记	功能描述
31:1	Reserved	
0	DIR	双通道正交脉冲计数方向指示 0: 加计数 1: 减计数

20.4.7 PCNT Count Register (PCNT_CNT)

地址偏移量：0x18

复位值：0x0000 0000

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

位	标记	功能描述
31:16	Reserved	
15:0	CNT	计数器计数值。

20.4.8 PCNT Count Overflow Register (PCNT_TOP)

地址偏移量：0x1c

复位值：0x0000 00FF

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

位	标记	功能描述
31:16	Reserved	
15:0	TOP	计数器上溢出阈值。 计数器下溢出重载值。

20.4.9 PCNT Count Overflow Buffer Register (PCNT_BUF)

地址偏移量：0x20

复位值：0x0000 00FF

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

位	标记	功能描述
31:16	Reserved	
15:0	BUF	CNT 寄存器、TOP 寄存器的缓存。

20.4.10 PCNT Interrupt Flag Register (PCNT_IFR)

地址偏移量：0x24

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								S1E	S0E	BB	FE	DIR	TO	OV	UF
								ro	ro	ro	ro	ro	ro	ro	ro

位	标记	功能描述
31:8	Reserved	
7	S1E	脉冲解码错误中断标识。 在一个非交编码完整采样周期内，S1 通道未发生变化。 仅在双通道非交编码模式有效。
6	S0E	脉冲解码错误中断标识。 在一个非交编码完整采样周期内，S0 通道未发生变化。 仅在双通道非交编码模式有效。
5	BB	脉冲解码错误中断标识。 双通道脉冲分别背靠背采样为高，中间并无低状态。 仅在双通道非交编码模式有效。
4	FE	脉冲解码错误中断标识。 在一个非交编码完整采样周期内，采样到的不是一个正确的非交编码帧。 仅在双通道非交编码模式有效。
3	DIR	正交脉冲方向改变中断标识。 正交脉冲被解码器判定为计数方向发生改变。 仅在双通道正交脉冲有效。
2	TO	超时中断标识。 在 3 种模式均有效。
1	OV	上溢出中断标识。 在 3 种模式均有效。
0	UF	下溢出中断标识。 在 3 种模式均有效。

20.4.11 PCNT Interrupt Clear Register (PCNT_ICR)

地址偏移量：0x28

复位值：0x000000FF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								S1E	S0E	BB	FE	DIR	TO	OV	UF
								wo0							

对每个 bit 写 0 可对 20.4.10 中 PCNT_IFR 各对应中断标识清除。

每 bit 详细解释参见 20.4.10。

20.4.12 PCNT Interrupt Enable Register (PCNT_IEN)

地址偏移量：0x2c

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								S1E	S0E	BB	FE	DIR	TO	OV	UF
								rw	rw	rw	rw	rw	rw	rw	rw

对每个 bit 对应 20.4.10 中 PCNT_IFR 各对应中断的输出使能位。

1: 使能。 0: 不使能。

每 bit 详细解释参见 20.4.10。

20.4.13 PCNT Synchronization Status Register (PCNT_SR2)

地址偏移量：0x30

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
B2C B2T T2C												ro	ro	ro	

位	标记	功能描述
31:3	Reserved	
2	B2C	将 BUF 中数值同步到 CNT 时的同步状态位。 1: 正在进行同步，不可读取 CNT 寄存器 0: 同步已完成
1	B2T	将 BUF 中数值同步到 TOP 时的同步状态位。 1: 正在进行同步，不可读取 TOP 寄存器 0: 同步已完成
0	T2C	将 TOP 中数值同步到 CTN 时的同步状态位。 1: 正在进行同步，不可读取 CNT 寄存器 0: 同步已完成

20.4.14 PCNT debug observation output selection register (PCNT_DBG)

地址偏移量：0x34

复位值：0x0000 0000

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

位	标记	功能描述
31:2	Reserved	
1:0	DBG	选择脉冲 PCNT_S0FO/PCNT_S1FO 观测输出的来源。 0: 固定为 0 1: 脉冲同步之后的输出 (S0A/S1A) 2: 脉冲极性选择之后的输出 (S0P/S1P) 3: 脉冲滤波之后的输出 (S0PF/S1PF)

21 Universal Synchronous Asynchronous Receiver/Transmitter (UART)

21.1 Introduction

通用同步异步收发器（UART）能够灵活地与外部设备进行全双工数据交换，它支持同步单向通信、单线半双工通信以及多处理器通信。常用于短距离、低速的串行通信中。UART 通过可编程波特率发生器提供了多种波特率。UART 支持多种工作模式。

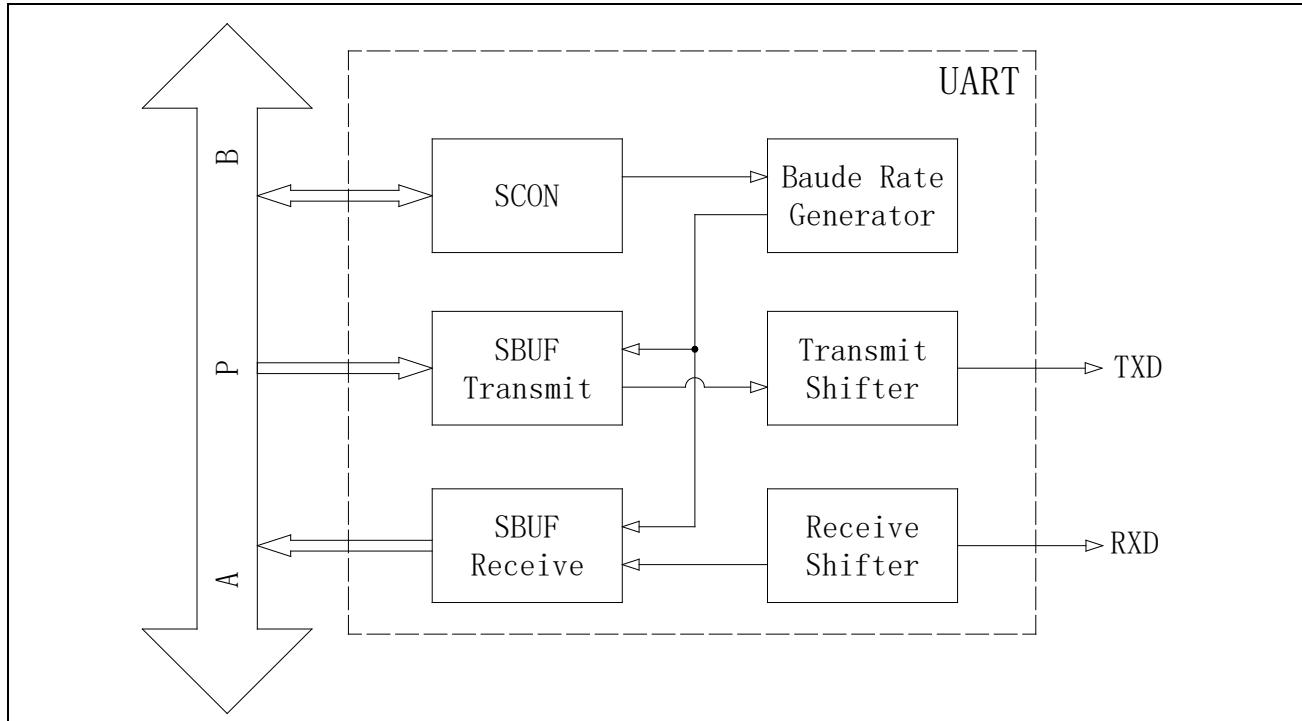


图 21-1 结构框图

21.2 Main Features

通用 UART 模块支持以下基本功能：

- 全双工传输、半双工传输、单线半双工传输
- 可编程串行通信功能
 - 两种字符长度：8 比特、9 比特
 - 三种校验方式：无检验、奇校验、偶校验
 - 三种停止长度：1 比特、2 比特、1.5 比特
- 16 比特波特率发生器
- 多机通讯
- 硬件地址识别
- 硬件流控
- DMA 传输握手

21.3 Functional Description

21.3.1 Operation Modes

UART 支持多种工作模式：同步半双工模式、异步全双工模式、单线半双工模式。通过 `UARTx_SCON.SM` 和 `UARTx_SCON.HDSEL` 搭配，即可配置出所需要的各种工作模式。

21.3.1.1 Mode0~Mode3 Function Comparison

配置 `UARTx_SCON.SM` 可选择不同的传输模式：Mode0~Mode3。这四种工作模式的主要功能对比如下表所示：

表 21-1 Mode0/1/2/3 数据结构

工作模式		传输位宽	数据组成	波特率
Mode0	同步模式 半双工	8bit	Data(8bit)	$BaudRate = \frac{f_{PCLK}}{12}$
Mode1	异步模式 全双工	10~11bit	Start (1bit) + Data(8bit) + Stop(1~2bit)	$BaudRate = \frac{f_{PCLK}}{OVER * SCNT}$
Mode2	异步模式 全双工	11~12bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1~2bit)	$BaudRate = \frac{f_{PCLK}}{OVER}$
Mode3	异步模式 全双工	11~12bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1~2bit)	$BaudRate = \frac{f_{PCLK}}{OVER * SCNT}$

注：

- Mode0 只能作为主机发送 UART 同步移位时钟，不可以作为从机接收外部输入的 UART 同步移位时钟。
- f_{PCLK} 代表当前 PCLK 的频率。
- OVER 的定义详见 `UARTx_SCON`。
- SCNT 的定义详见 `UARTx_SCNT`。
- B8 数据位比较特殊，在不同应用下具有不同的含义，请参考以下表格：

表 21-2 B8 数据含义

应用场景	<code>UARTx_SCON.ADRDET</code>	<code>UARTx_SCON.B8CONT[1:0]</code>	B8 数据含义
奇偶校验	--	01/10	接收时，B8 是所收到的 8-Bit 数据的奇偶校验位； 发送时，B8 是待发送的 8-Bit 数据的奇偶校验位；
多机通讯	1	--	B8=1，代表当前是地址帧； B8=0，代表当前是数据帧；
其他	0	00/11	接收/发送数的第 8 比特

注意：

- 当开启多机通讯模式，接收数据奇偶校验自动关闭；发送数据奇偶校验仍受 `B8CONT` 控制。

21.3.1.2 Mode0 Data Transmission and Reception Description

发送数据时，清除 UARTx_SCON.REN 位，并将数据写入 UARTx_SBUF 寄存器中。此时，发送数据从 RXD 输出（低位在先，高位在后），同步移位时钟从 TXD 输出。

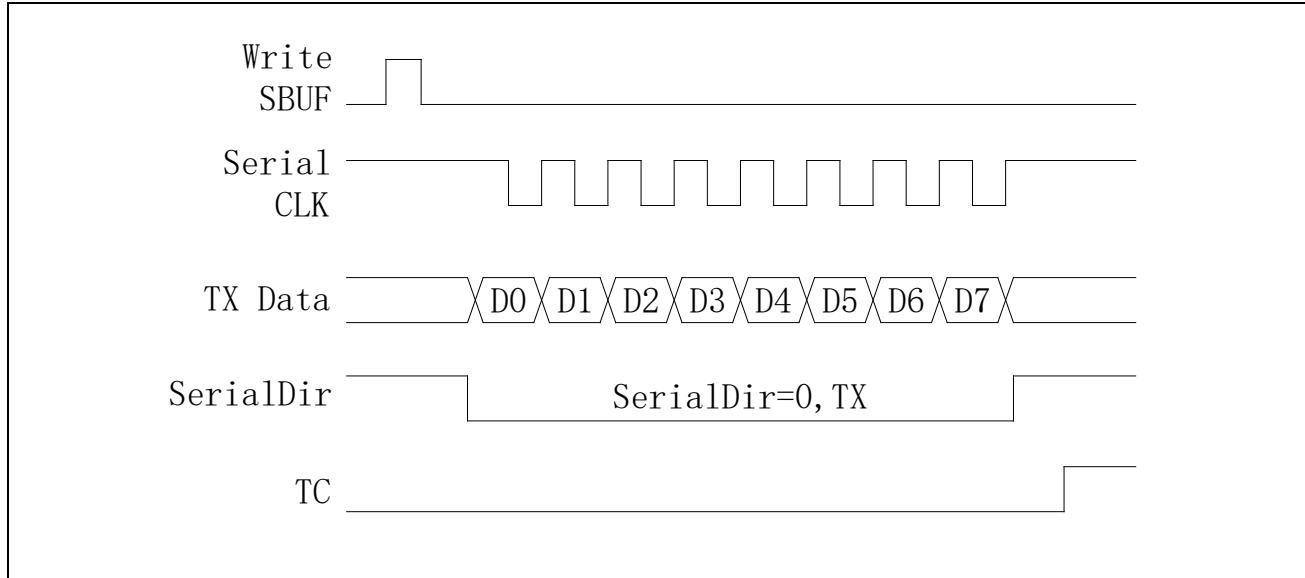


图 21-2 Mode0 发送数据

接收数据时，将 `UARTx_SCON.REN` 位置 1，并将 `UARTx_ISR.RC` 位清零。当接收结束，数据可从 `UARTx_SBUF` 寄存器读取。此时，接收数据从 `RXD` 输入（低位在先，高位在后），同步移位时钟从 `TXD` 输出。

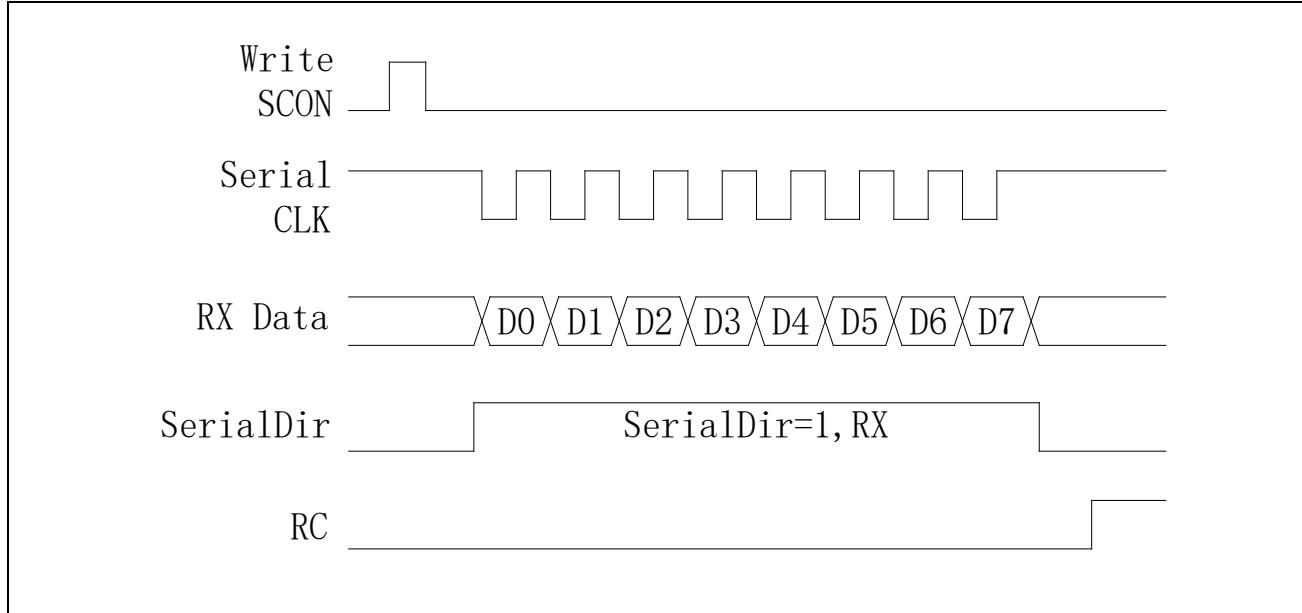


图 21-3 Mode0 接收数据

21.3.1.3 Mode 1 Data Transmission and Reception Description

发送数据时，与 `UARTx_SCON.REN` 的值无关，将所发送数据写入 `UARTx_SBUF` 寄存器中，数据就会从 `TXD` 移出（低位在先，高位在后）。

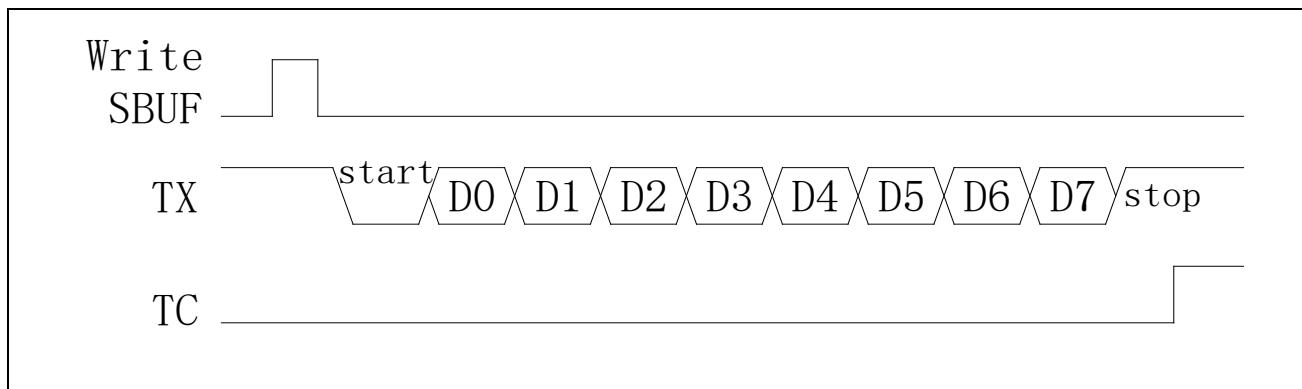


图 21-4 Mode1 发送数据

接收数据时，将 `UARTx_SCON.REN` 位置 1，并将 `UARTx_ISR.RC` 位清 0。开始接收 RXD 上数据（低位在先，高位在后），当接收完毕，可以从 `UARTx_SBUF` 寄存器读出。



RC

图 21-5 Mode1 接收数据

21.3.1.4 Mode 2 Data Transmission and Reception Description

发送数据时，与 `UARTx_SCON.REN` 的值无关，将所发送数据写入 `UARTx_SBUF` 寄存器中，数据就会从 TXD 移出（低位在先，高位在后）。

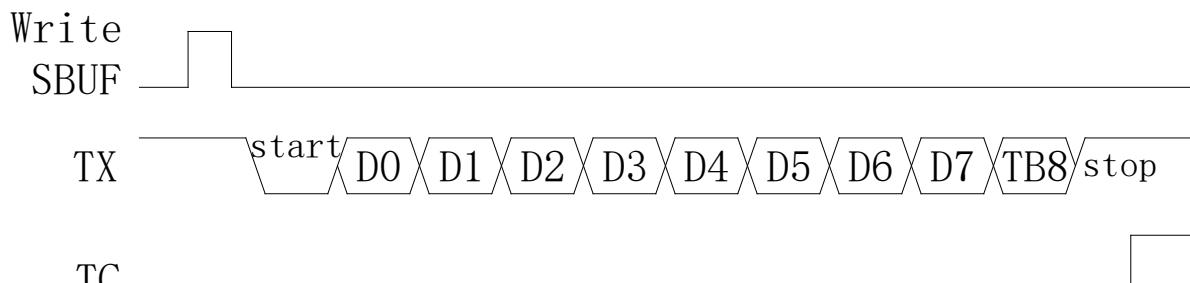


图 21-6 Mode2 发送数据

接收数据时，需将 `UARTx_SCON.REN` 位置 1，并将 `UARTx_ISR.RC` 位清 0。开始接收 RXD 上的数据（低位在先，高位在后），当接收完毕，可以从 `UARTx_SBUF` 寄存器读取。



RC

图 21-7 Mode2 接收数据

21.3.1.5 Mode 3 Data Transmission and Reception Description

发送数据时，与 `UARTx_SCON.REN` 的值无关，将所发送数据写入 `UARTx_SBUF` 寄存器中，数据就会从 `TXD` 移出（低位在先，高位在后）。

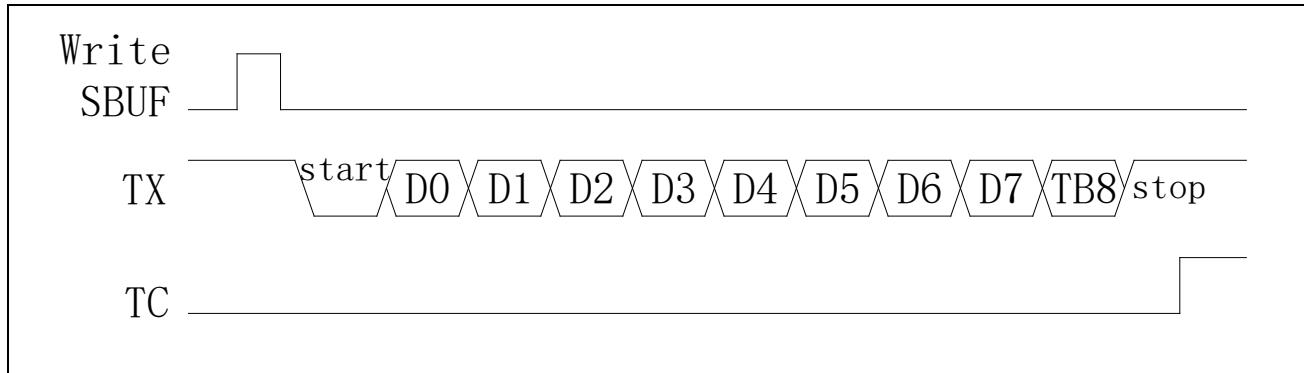


图 21-8 Mode3 发送数据

接收数据时，将 `UARTx_SCON.REN` 位置 1，并将 `UARTx_ISR.RC` 位清 0。开始接收 `RXD` 上数据（低位在先，高位在后），当接收完毕，可以从 `UARTx_SBUF` 寄存器读出。

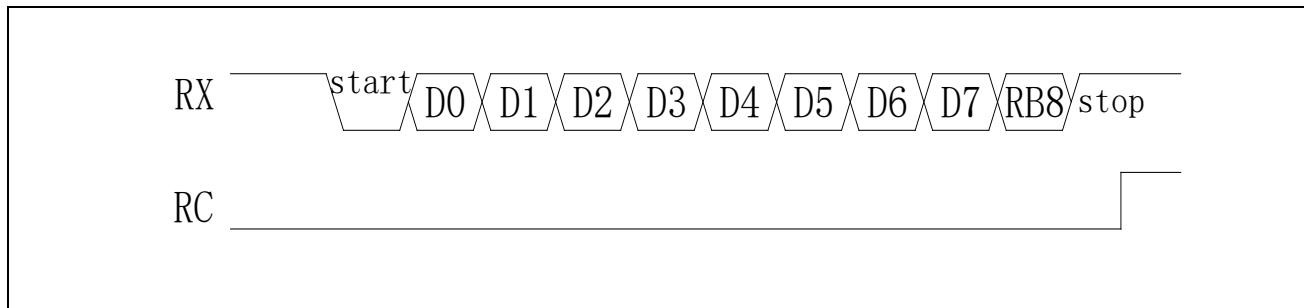


图 21-9 Mode3 接收数据

21.3.1.6 Single-wire half-duplex data transmission and reception description

当 `UARTx_SCON.HDSEL=1'b1`，可进入单线半双工传输模式：

- TX 与 RX 信号线在模块内部相连。
- RX 信号不再使用，接发送的数据都经由 TX 信号完成。TX 信号的方向控制由硬件逻辑完成，无需软件控制。
- 当发送缓存为空时，TX 信号始终为输入（接收状态），一旦向发送缓存填入一个数据，TX 信号变为输出（发送状态），当发送完成后，发送缓存变为空，TX 信号又回到输入。
- 无数据传输时，`TXD` 引脚始终处于输入状态。用户程序需配置该引脚为开漏输出并外接上拉电阻。

注意：

1. 即使模块处于接收过程中，只要发送缓存被填入数据，TX 信号立即变为输出（发送状态），接收过程将被打断，并得到一个错误的接收数据。

2. 单线半双工模式与 Mode0 模式的异同：

同：两者都是用一根信号分时复用地完成数据收发，

异：Mode0 为同步传输，有同步时钟，而单线半双工模式为异步传输，所有的波特率计算还是遵照 Mode1~3 的波特率公式，仅仅是收发功能复用到了 TX 信号上。

21.3.2 Baud Rate Generation

Mode0~Mode3 生成波特率的公式不尽相同，详见下方所示。

$$\text{Mode0 波特率生成公式: } \text{BaudRate} = \frac{f_{\text{PCLK}}}{12}$$

$$\text{Mode1 波特率生成公式: } \text{BaudRate} = \frac{f_{\text{PCLK}}}{\text{OVER} * \text{SCNT}}$$

$$\text{Mode2 波特率生成公式: } \text{BaudRate} = \frac{f_{\text{PCLK}}}{\text{OVER}}$$

$$\text{Mode3 波特率生成公式: } \text{BaudRate} = \frac{f_{\text{PCLK}}}{\text{OVER} * \text{SCNT}}$$

注：

- f_{PCLK} 代表当前 PCLK 的频率。
- OVER 的定义详见 *UARTx_SCON*。
- SCNT 的定义详见 *UARTx_SCNT*

21.3.2.1 Mode1/Mode3 baud rate setting example

表 21-3 PCLK=4MHz 波特率计算表

波特率	PCLK = 4 MHz					
	OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	208	2403.85	0.16%	104	2403.85	0.16%
4800	104	4807.69	0.16%	52	4807.69	0.16%
9600	52	9615.38	0.16%	26	9615.38	0.16%
19200	26	19230.77	0.16%	13	19230.77	0.16%
38400	13	38461.54	0.16%	7	35714.29	-6.99%
57600	9	55555.56	-3.55%	4	62500.00	8.51%
76800	7	71428.57	-6.99%	3	83333.33	8.51%
115200	4	125000.00	8.51%	2	125000.00	8.51%
128000	4	125000.00	-2.34%	2	125000.00	-2.34%
250000	2	250000.00	0.00%	1	250000.00	0.00%

表 21-4 PCLK=8MHz 波特率计算表

波特率	PCLK = 8 MHz					
	OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	417	2398.08	-0.08%	208	2403.85	0.16%
4800	208	4807.69	0.16%	104	4807.69	0.16%
9600	104	9615.38	0.16%	52	9615.38	0.16%
19200	52	19230.77	0.16%	26	19230.77	0.16%
38400	26	38461.54	0.16%	13	38461.54	0.16%
57600	17	58823.53	2.12%	9	55555.56	-3.55%
76800	13	76923.08	0.16%	7	71428.57	-6.99%
115200	9	111111.11	-3.55%	4	125000.00	8.51%
128000	8	125000.00	-2.34%	4	125000.00	-2.34%
256000	4	250000.00	-2.34%	2	250000.00	-2.34%
500000	2	500000.00	0.00%	1	500000.00	0.00%

表 21-5 PCLK=16MHz 波特率计算表

波特率	PCLK = 16 MHz					
	OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	833	2400.96	0.04%	417	2398.08	-0.08%
4800	417	4796.16	-0.08%	208	4807.69	0.16%
9600	208	9615.38	0.16%	104	9615.38	0.16%
19200	104	19230.77	0.16%	52	19230.77	0.16%
38400	52	38461.54	0.16%	26	38461.54	0.16%
57600	35	57142.86	-0.79%	17	58823.53	2.12%
76800	26	76923.08	0.16%	13	76923.08	0.16%
115200	17	117647.06	2.12%	9	111111.11	-3.55%
128000	16	125000.00	-2.34%	8	125000.00	-2.34%
256000	8	250000.00	-2.34%	4	250000.00	-2.34%
500000	4	500000.00	0.00%	2	500000.00	0.00%
1000000	2	1000000.00	0.00%	1	1000000.00	0.00%

表 21-6 PCLK=24MHz 波特率计算表

波特率	PCLK = 24 MHz					
	OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	1250	2400.00	0.00%	625	2400.00	0.00%
4800	625	4800.00	0.00%	313	4792.33	-0.16%
9600	313	9584.66	-0.16%	156	9615.38	0.16%
19200	156	19230.77	0.16%	78	19230.77	0.16%
38400	78	38461.54	0.16%	39	38461.54	0.16%
57600	52	57692.31	0.16%	26	57692.31	0.16%
76800	39	76923.08	0.16%	20	75000.00	-2.34%
115200	26	115384.62	0.16%	13	115384.62	0.16%
128000	23	130434.78	1.90%	12	125000.00	-2.34%
256000	12	250000.00	-2.34%	6	250000.00	-2.34%
1000000	3	1000000.00	0.00%	2	750000.00	-25.00%
1500000	2	1500000.00	0.00%	1	1500000.00	0.00%

表 21-7 PCLK=32MHz 波特率计算表

波特率	PCLK = 32 MHz					
	OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	1667	2399.52	-0.02%	833	2400.96	0.04%
4800	833	4801.92	0.04%	417	4796.16	-0.08%
9600	417	9592.33	-0.08%	208	9615.38	0.16%
19200	208	19230.77	0.16%	104	19230.77	0.16%
38400	104	38461.54	0.16%	52	38461.54	0.16%
57600	69	57971.01	0.64%	35	57142.86	-0.79%
76800	52	76923.08	0.16%	26	76923.08	0.16%
115200	35	114285.71	-0.79%	17	117647.06	2.12%
128000	31	129032.26	0.81%	16	125000.00	-2.34%
256000	16	250000.00	-2.34%	8	250000.00	-2.34%
1000000	4	1000000.00	0.00%	2	1000000.00	0.00%
2000000	2	2000000.00	0.00%	1	2000000.00	0.00%

表 21-8 PCLK=48MHz 波特率计算表

波特率	PCLK = 48 MHz					
	OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	2500	2400.00	0.00%	1250	2400.00	0.00%
4800	1250	4800.00	0.00%	625	4800.00	0.00%
9600	625	9600.00	0.00%	313	9584.66	-0.16%
19200	313	19169.33	-0.16%	156	19230.77	0.16%
38400	156	38461.54	0.16%	78	38461.54	0.16%
57600	104	57692.31	0.16%	52	57692.31	0.16%
76800	78	76923.08	0.16%	39	76923.08	0.16%
115200	52	115384.62	0.16%	26	115384.62	0.16%
128000	47	127659.57	-0.27%	23	130434.78	1.90%
256000	23	260869.57	1.90%	12	250000.00	-2.34%
1000000	6	1000000.00	0.00%	3	1000000.00	0.00%
2000000	3	2000000.00	0.00%	2	1500000.00	-25.00%
3000000	2	3000000.00	0.00%	1	3000000.00	0.00%

21.3.3 Frame Error Detection

当工作在 Mode1/2/3 时，UART 具有帧错误检测功能，如果接收数据时硬件未在预期时间内识别出停止位，从而出现同步失效或过度的噪声，则会检测到帧错误。检测到帧错误时，UARTx_ISR.FE 置 1。UARTx_ISR.FE 要及时软件清零。

21.3.4 Multi-machine Communication

当工作在 Mode2/3 时，将 UARTx_SCON.ADRDET 位置 “1”，可开启多机通讯功能。

主机可以通过 UARTx_SBUF[8] 来区分当前的发送帧是地址帧（UARTx_SBUF[8]=1）还是数据帧（UARTx_SBUF[8]=0）。

- 当为数据帧时，该帧数据不会存入到从机的 UARTx_SBUF 寄存器中，从机也不会产生接收中断。
- 当为地址帧时，由于多机通讯中自动地址识别功能已开启，使得从机可以检测接收到的地址与其自身地址是否相符合。
 - 如果地址相符合，从机会对 UARTx_ISR.RC 置“1”，UARTx_SBUF[8] 置 “1”，同时将该地址帧存入到 UARTx_SBUF 寄存器中。从机软件看到 UARTx_SBUF[8]=1 并且 UARTx_ISR.RC=1 后，将 UARTx_SCON.ADEDET 位清 “0”，接受数据帧。
 - 如果地址不符合，表明主机并不是寻址该从机，从机硬件保持 UARTx_SBUF[8] 和 UARTx_ISR.RC 为“0”，软件保持 UARTx_SCON.ADRDET 位为“1”，继续处于地址监听状态。

注：如果有需要，也可以在 Mode1 下开启多机通讯位，此时 TB8 位由 stop 位代替。当从机接收到匹配的地址帧和有效的 stop 位时，UARTx_ISR.RC 会被置“1”。

21.3.4.1 Given address

UART 设备的 UARTx_SADDR 寄存器用来表示自己的设备给定地址。

UARTx_SADEN 寄存器是地址掩码，当 UARTx_SADEN 的某一位为“0”可定义地址中的无关位，不参与地址匹配。这些无关位增加了寻址的灵活性，使得主机可以同时寻址一个或者多个从机设备。

注意：如果需要给出唯一匹配地址，UARTx_SADEN 寄存器必须设为 8'hFF。给定地址公式如下所示：

$$\text{GivenAddr} = \text{SADDR} \& \text{SADEN}$$

21.3.4.2 Broadcast Address

广播地址是用来同时寻址所有从机设备的，一般广播地址为 8'hFF。

$$\text{BroadCastAddr} = \text{SADDR} | \text{SADEN}$$

21.3.4.3 Examples

假设某从机的 UARTx_SADDR 和 UARTx_SADEN 配置如下：

SADDR: 8'b01101001

SADEN: 8'b11111011

那么其给定地址和广播地址如下：

Given: 8'b01101x01

Broadcast: 8'b11111x11

可见，主机可以用四个地址寻址到本从机，分别是：

8'b01101001 和 8'b01101101 (given address)

8'b11111011 和 8'b11111111 (broadcast address)。

21.3.5 DMAC Hardware Handshake

UART 模块支持 DMAC 的硬件握手逻辑。

- 将 UARTx_SCON.DMACTXEN 设置为 1，可以打开 UART TX 的 DMAC 硬件握手逻辑。当发送缓存为空时，UART 会向 DMAC 发出数据搬运请求 TX REQ。DMAC 收到该信号，则从 DMAC 源地址搬运一个帧的发送数据到 UARTx_SBUF 中。上述步骤重复发生，直到 DMAC 中所配置的数据长度全部搬运完毕。
- 将 UARTx_SCON.DMACRXEN 设置为 1，可以打开 UART RX 的 DMAC 硬件握手逻辑。当一帧接收完成，UART 会向 DMAC 发出数据搬运请求 RX REQ。DMAC 收到该信号，则从 UARTx_SBUF 中把接收数据搬运至 DMAC 的目标地址中。上述步骤重复发生，直到 DMAC 中所配置的数据长度全部搬运完毕。

21.3.6 Hardware Flow Control

通过增加 nCTS 和 nRTS 信号可以实现 UART 硬件流控的功能，即 UART 硬件模块根据 nCTS 和 nRTS 的高低电平自动控制数据的收发，而无需通过软件来判断。两个 UART 模块之间的硬件流控示意图如下所示：

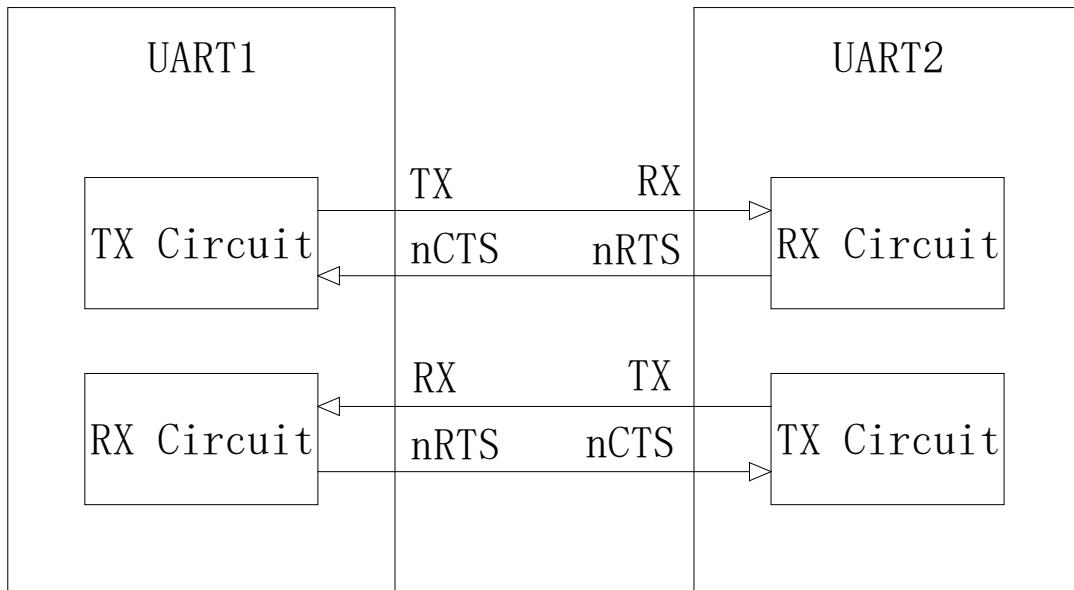


图 21-10 UART 硬件流控

■ nRTS 流控

nRTS 流控使能时 (UARTx_SCON.RTSEN 设置为 1)：

- 当 UART 接收缓存空时，会将 nRTS 变为有效（连接到低电平）。
- 当接收缓存满时，会将 nRTS 变为无效（连接到高电平），表明发送过程会在当前帧结束后停止。

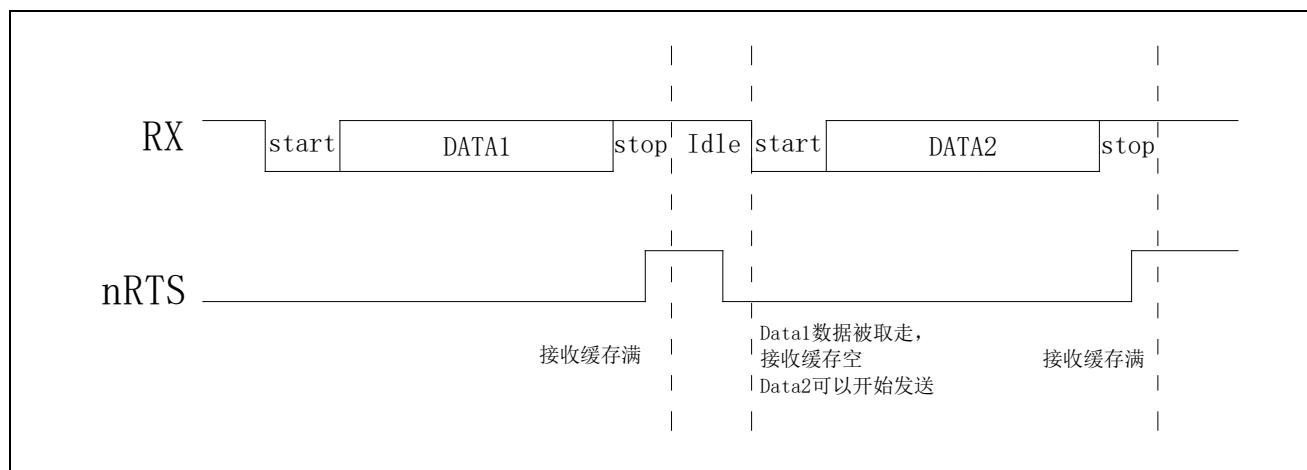


图 21-11 nRTS 硬件流控信号

■ CTS 流控

nCTS 流控使能时 (UARTx_SCON.CTSEN 设置为 1)，在 UART 发送下一帧数据之前，先判断 nCTS 的高低电平：

- 如果 nCTS 为有效（连接到低电平），则 UART 发送下一帧数据。
- 如果 nCTS 为无效（连接到高电平），则 UART 当前帧发送完成后，暂停发送下一帧数据。

当 nCTS 流控使能时，一旦 nCTS 信号发生翻转，UARTx_SFLAG.CTSIE 便会硬件置 1。如果 UARTx_SCON.CTSIE 置 1，则会产生中断，同时，nCTS 信号的高低电平会记录在 UARTx_SFLAG.CTS 标志位中。

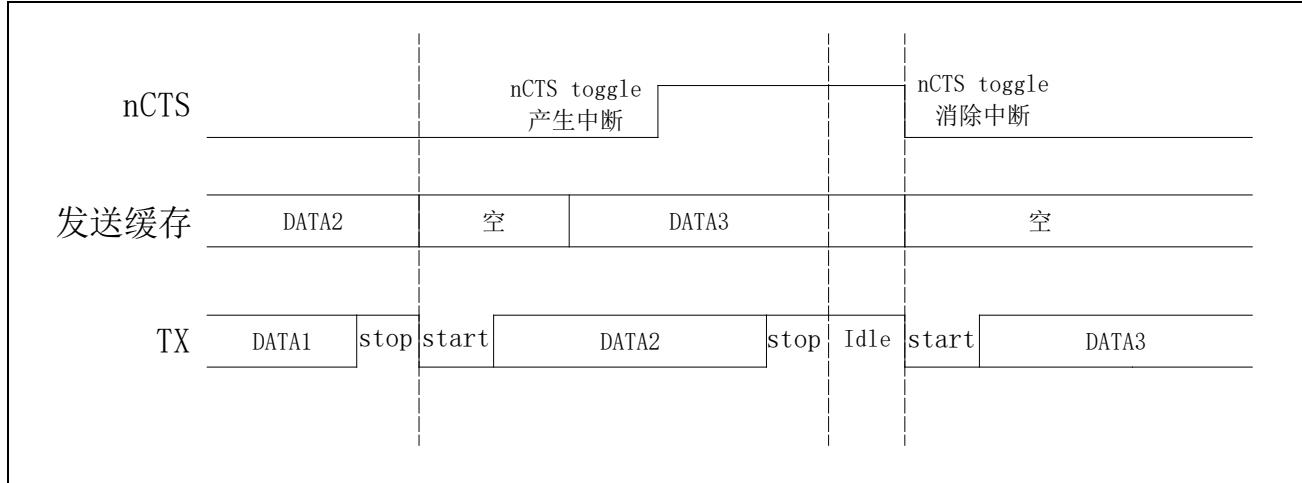


图 21-12 nCTS 硬件流控信号

21.3.7 Transceiver Buffer

■ 接收缓存

UART 模块接收端有一个帧(8/9-Bit)的接收缓存，即保存接收的数据帧直至下一帧数据的 Stop 位被接收才更新该数据帧。

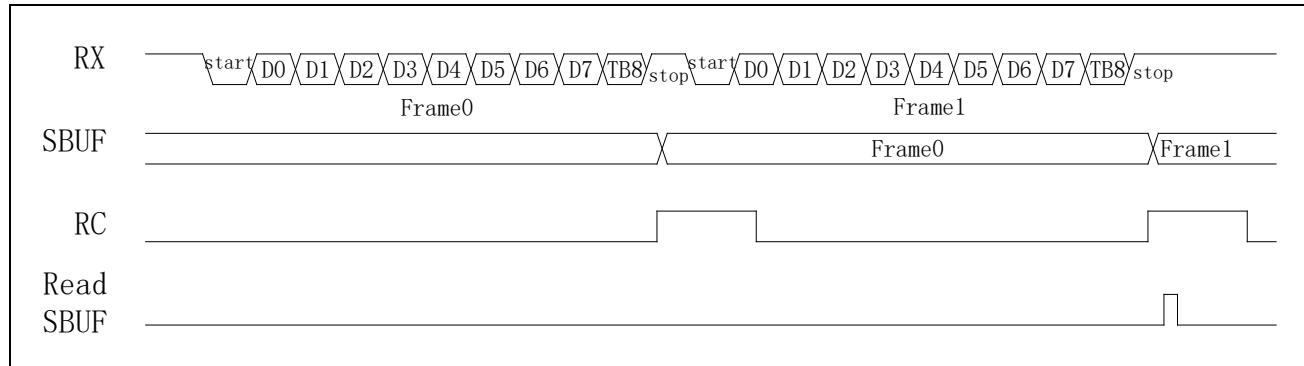


图 21-13 接收缓存

■ 发送缓存

UART 模块发送端有一个帧(8/9-Bit)的发送缓存，当 UART 在发送当前帧时，下一个发送数据软件写入 `UARTx_SBUF`。

当 `UARTx_ISR.TXE=0` 时，表明当前发送缓存满，`UARTx_SBUF` 不能写入下一个发送数据。否则该数据会硬件丢弃。

当 `UARTx_ISR.TXE=1` 时，表明当前发送缓存空，`UARTx_SBUF` 可以写入下一个发送数据，在完成当前数据传输后，硬件自动把发送缓存中的数据装载入移位寄存器中发送出去。

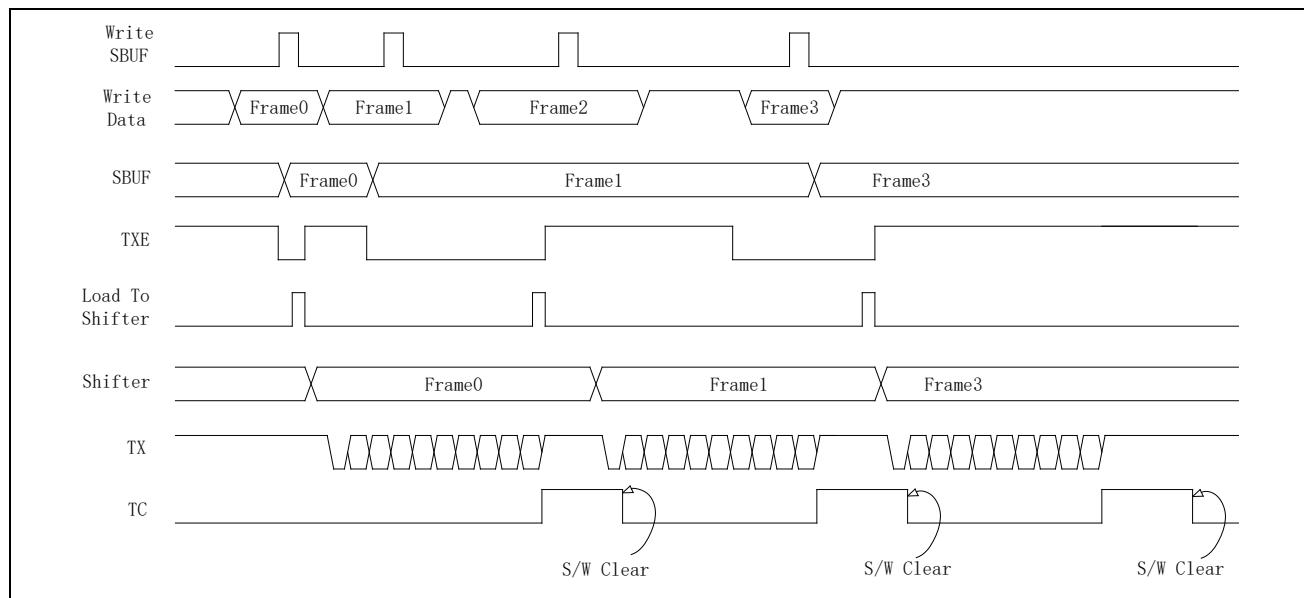


图 21-14 发送缓存

21.4 Registers

UART0 基地址: 0x4000 0000

UART1 基地址: 0x4000 0100

UART2 基地址: 0x4000 6000

UART3 基地址: 0x4000 6400

寄存器	偏移地址	描述
UARTx_SBUF	0x00	数据寄存器
UARTx_SCON	0x04	控制寄存器
UARTx_SADDR	0x08	地址寄存器
UARTx_SADEN	0x0C	地址掩码寄存器
UARTx_ISR	0x10	中断标志位寄存器
UARTx_ICR	0x14	中断标志位清除寄存器
UARTx_SCNT	0x18	波特率寄存器

21.4.1 Data Register (UARTx_SBUF)

偏移地址：0x00

复位值：0x0000 0000

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

位	标记	功能描述
31:9	Reserved	
8	DATA[8]	<p>在 Mode0/1 下，读取该位为 0，写入该位无效；</p> <p>在 Mode2/3 下，该位表示 Bit8 数据位，分以下两种情况：</p> <p>(1) 当硬件奇偶校验位开启时，接收时该位为接收数据奇偶校验位，校验由硬件进行，如校验出错，校验错误标志位 PE 置 1；发送时该位无效，发送数据奇偶校验位由硬件计算并发送；</p> <p>(2) 当硬件奇偶校验位关闭时，接收时该位为接收数据 Bit8；发送时该位为发送数据 Bit8；</p> <p>注意：当开启多机通讯模式，接收数据奇偶校验自动关闭；发送数据奇偶校验仍受 B8CONT 控制；</p>
7:0	DATA[7:0]	发送数据时，将发送数据写入该寄存器；接收数据时，数据接收完毕后，从该寄存器中读取。

21.4.2 Control Register (UARTx_SCON)

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									HDS EL	FEIE	CTSI E	CTS EN	RTS EN	DMA TXE N	DMA RXE N
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOPBIT	PEIE	Reserved	OVR	TXEIE	SM	ADRDET	REN	B8CONT	TCIE	RCIE					
RW	RW		RW	RW		RW	RW				RW	RW	RW	RW	RW

位	标记	功能描述
31:23	Reserved	
22	HDSEL	单线半双工模式使能位；0：关闭单线半双工模式；1：打开单线半双工模式；
21	FEIE	帧错误中断使能位；0：关闭中断；1：打开中断；
20	CTSIE	CTS 信号翻转中断使能位；0：关闭中断；1：打开中断；
19	CTSEN	硬件 flow control 信号使能位；0：关闭 flow control 信号；1：打开 flow control 信号；
18	RTSEN	
17	DMATXEN	TX DMAC 的硬件握手信号使能位；0：关闭硬件握手信号；1：打开硬件握手信号；
16	DMARXEN	RX DMAC 的硬件握手信号使能位；0：关闭硬件握手信号；1：打开硬件握手信号；
15:14	STOPBIT	stop bit 长度选择；00:1-bit; 01:1.5-bit; 10:2-bit; 11: reserved; 注意：Mode0 时虽然没有 Stop Bit，但仍需把 STOPBIT[1:0]保持为 2'b00；
13	PEIE	奇偶校验错误中断使能位；0：关闭奇偶校验错误中断；1：打开奇偶校验错误中断； 数据接收标志产生于数据停止位接收完成，硬件奇偶校验与数据接收中断由于停止位设置不同，奇偶校验错误中断会提前于数据接收中断。使用此中断使能时请注意。 建议如下： 方法 1：关闭 PEIE 中断使能，接收数据中断后软件判断 ISR.PE 奇偶校验是否正确。 方法 2：关闭 PEIE 中断使能，接收数据中断通过接收 SBUF.BIT8 软件判断奇偶校验是否正确。
12:10	Reserved	
9	OVER	Mode0：无效； Mode1/3：0：16 采样分频；1：8 采样分频； Mode2：0：32 采样分频；1：16 采样分频；
8	TXEIE	TX 空中断使能位；0：TX Buffer 空中断关闭；1：TX Buffer 空中断打开；
7:6	SM	工作模式；00：mode0；01：mode1；10：mode2；11：mode3；
5	ADRDET	多机通讯地址自动识别使能位；0：关闭；1：打开；
4	REN	Mode0：0：发送；1：接收； Mode1/2/3：0：发送；1：接收/发送；
3:2	B8CONT	Bit8 数据控制位； 00：由软件读写 SBUF[8]来决定；01：硬件偶校验； 10：硬件奇校验；11：保留；
1	TCIE	发送中断使能位；0：发送中断关闭；1：发送中断打开；

0	RCIE	接收中断使能位；0：接收中断关闭；1：接收中断打开；
---	------	----------------------------

21.4.3 Address Register (UARTx_SADDR)

偏移地址：0x08

复位值：0x0000 0000

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

位	标记	功能描述
31:8	Reserved	
7:0	SADDR	从机设备地址寄存器

21.4.4 Address Mask Register (UARTx_SADEN)

偏移地址：0x0C

复位值：0x0000 0000

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

位	标记	功能描述
31:8	RESERVED	
7:0	SADEN	从机设备地址掩码寄存器

21.4.5 Flag Register (UARTx_ISR)

偏移地址：0x10

复位值：0x0000 0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CTS	CTSIF	PE	TXE	FE	TC	RC	
								RO	RO	RO	RO	RO	RO	RO	RO

位	符号	描述
31:7	Reserved	
6	CTS	CTS 信号标志位；硬件置 1；硬件清零；0: CTS 信号为低电平；1: CTS 信号为高电平；
5	CTSIF	CTS 中断标志位；硬件置 1；软件清零；0: CTS 信号没有发生反转；1: CTS 信号发生反转；
4	PE	奇偶校验错误标志位；硬件置 1；软件清零；0: 无奇偶校验错误；1: 奇偶校验错误；
3	TXE	Tx Buffer 空标志位；硬件置 1；硬件清零；0: Tx Buffer 非空；1: Tx Buffer 空
2	FE	帧错误标志位；0: 硬件置 1；软件清零；
1	TC	发送完毕中断标志位；硬件置 1；软件清零；0: 发送未完成；1: 发送完毕；
0	RC	接收完毕中断标志位；硬件置 1；软件清零；0: 接收未完成；1: 接收完成；

21.4.6 Flag Clear Register (UARTx_ICR)

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CTSIFCF	PECF	Res.	FECF	TCCF	RCCF		
								R1W0	R1W0		R1W0	R1W0	R1W0		

位	标记	功能描述
31:6	Reserved	
5	CTSIFCF	CTSIF 标志清楚位；写 0 清除；写 1 无效；
4	PECF	PE 标志清楚位；写 0 清除；写 1 无效；
3	Reserved	
2	FECF	FE 标志清除位；写 0 清零；写 1 无效；
1	TCCF	TC 标志清除位；写 0 清零；写 1 无效；
0	RCCF	RC 标志清除位；写 0 清零；写 1 无效；

21.4.7 Baud Rate Register (UARTx_SCNT)

偏移地址: 0x18

复位值: 0x0000 0000

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

位	标记	功能描述
31:16	Reserved	
15:0	SCNT	波特率计数器

22 Low Power Synchronous Asynchronous Receiver/Transmitter (LPUART)

22.1 Introduction

低功耗同步异步接收器 (LPUART) 是一种 UART，允许有限功耗下进行全双工 UART 通信。即使当微控制器处于停止模式，能耗极低时，LPUART 也会等待 UART 帧的到来。LPUART 包含所有必要的硬件支持，使在最小功耗下可以进行异步串行通信。

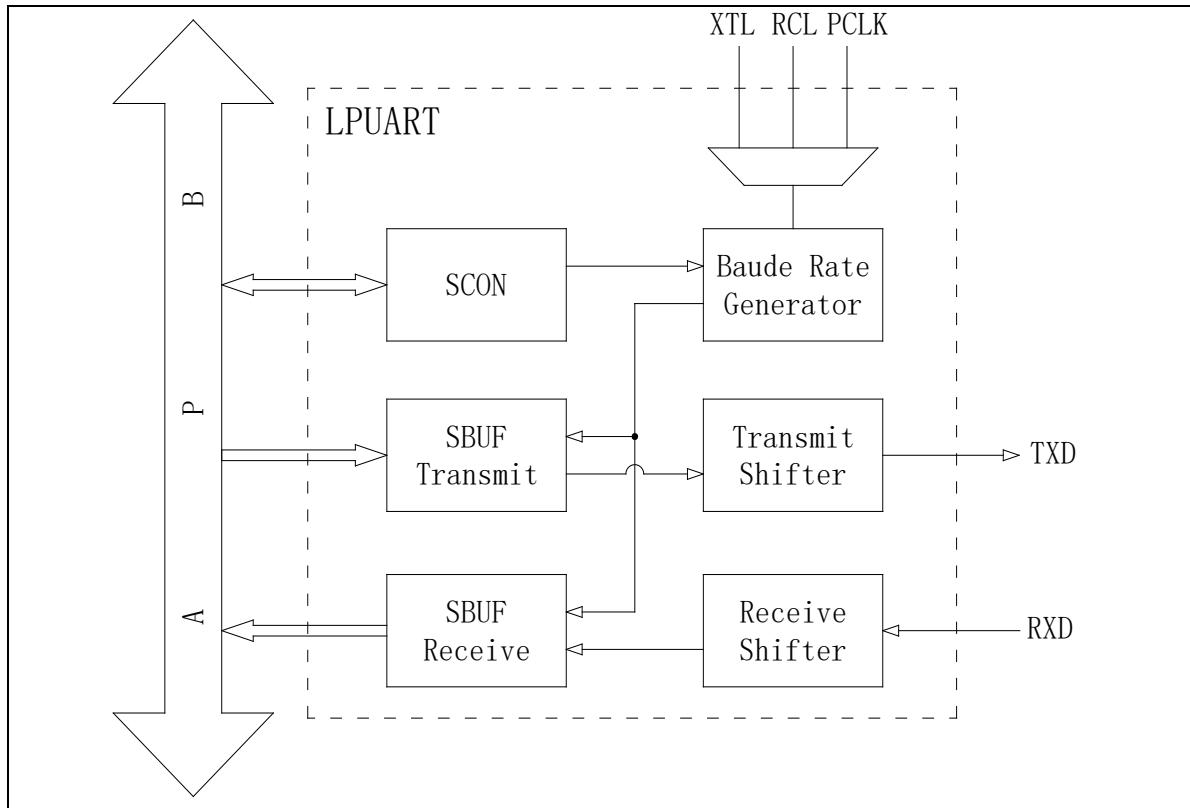


图 22-1 结构框图

22.2 Main Features

LPUART 模块（LPUART0/1）支持以下基本功能：

- 配置时钟 PCLK
- 传输时钟 SCLK（SCLK 可选择 XTL、RCL 以及 PCLK）
- 系统低功耗模式下收发数据
- 全双工传输、半双工传输、单线半双工传输
- 可编程串行通信功能
 - 两种字符长度：8 比特、9 比特
 - 三种校验方式：无检验、奇校验、偶校验
 - 三种停止长度：1 比特、2 比特、1.5 比特
- 16 比特波特率计数器
- 多机通讯
- 硬件地址识别
- 硬件流控
- DMA 传输握手

22.3 Functional Description

22.3.1 Configuration Clock and Transmission Clock

LPUART 模块有两个时钟：配置时钟 PCLK 和传输时钟 SCLK。

■ 配置时钟

配置时钟（PCLK）用于系统 APB 总线对 LPUART 模块进行寄存器配置，配置时钟固定为 APB 总线时钟 PCLK。当系统进入深度休眠（DeepSleep）模式，PCLK 时钟将会停止。

■ 传输时钟

传输时钟（SCLK）用于 LPUART 数据收发逻辑工作，可选择外部低速晶振时钟（XTL）、内部低速 RC 时钟（RCL）以及 PCLK 时钟。

当系统进入深度休眠（DeepSleep）模式，如果 SCLK 选择外部低速晶振时钟（XTL）或者内部低速 RC 时钟（RCL）。LPUART 仍旧可以进行正常的数据收发，而不受系统深度休眠（DeepSleep）模式的影响。

22.3.2 Operating Modes

LPUART 支持多种工作模式：同步半双工模式、异步全双工模式、单线半双工模式。通过 LPUARTx_SCON.SM 和 LPUARTx_SCON.HDSEL 搭配，即可配置出所需要的各种工作模式。

22.3.2.1 Mode0~Mode3 Function Comparison

配置 LPUARTx_SCON.SM 可选择不同的传输模式：Mode0~Mode3。这四种工作模式的主要功能对比如下表所示：

表 22-1 Mode0/1/2/3 数据结构

工作模式		传输位宽	数据组成	波特率
Mode0	同步模式 半双工	8-Bit	Data(8bit)	$BaudRate = \frac{f_{sclk}}{12}$
Mode1	异步模式 全双工	10-Bit	Start(1bit) + Data(8bit) + Stop (1~2bit)	$BaudRate = \frac{f_{sclk}}{OVER * SCNT}$
Mode2	异步模式 全双工	11-Bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop (1~2bit)	$BaudRate = \frac{f_{sclk}}{OVER}$
Mode3	异步模式 全双工	11-Bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop (1~2bit)	$BaudRate = \frac{f_{sclk}}{OVER * SCNT}$

注：

- Mode0 只能作为主机发送 LPUART 同步移位时钟，不可以作为从机接收外部输入的 LPUART 同步移位时钟；
- f_{sclk} 为 SCLK 的时钟频率；
- OVER 的定义详见 LPUARTx_SCON；
- SCNT 的定义详见 LPUARTx_SCNT；
- B8 数据位比较特殊，在不同应用下具有不同的含义，请参考以下表格：

表 22-2 B8 数据含义

应用场景	LPUARTx_SCON. ADRDET	LPUARTx_SCON. B8CONT[1:0]	B8 数据含义
奇偶校验	--	01/10	接收时，B8 是所收到的 8-Bit 数据的奇偶校验位； 发送时，B8 是所发送的 8-Bit 数据的奇偶校验位；
多机通讯	1	--	B8=1，代表当前是地址帧； B8=0，代表当前是数据帧；
其他	0	00/11	接收/发送的 DATA[8]

注意：

- 当开启多机通讯模式，接收数据奇偶校验自动关闭；发送数据奇偶校验仍受 B8CONT 控制。

22.3.2.2 Mode0 Data Transmission and Reception Description

发送数据时，清除 LPUARTx_SCON.REN 位，并将数据写入 LPUARTx_SBUF 寄存器中。此时，发送数据将从 RXD 输出（低位在先，高位在后），同步移位时钟从 TXD 输出。

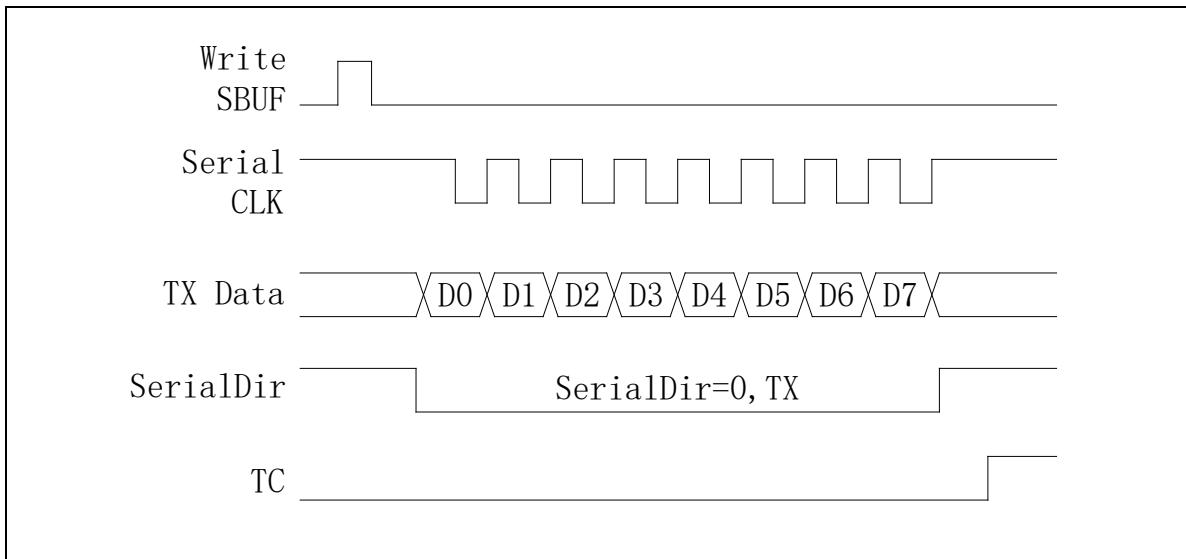


图 22-2 Mode0 发送数据

接收数据时，将 LPUARTx_SCON.REN 位置 1，并将 LPUARTx_ISR.RC 位清零。当接收结束，数据可从 LPUARTx_SBUF 寄存器读取。此时，接收数据从 RXD 输入（低位在先，高位在后），同步移位时钟从 TXD 输出。

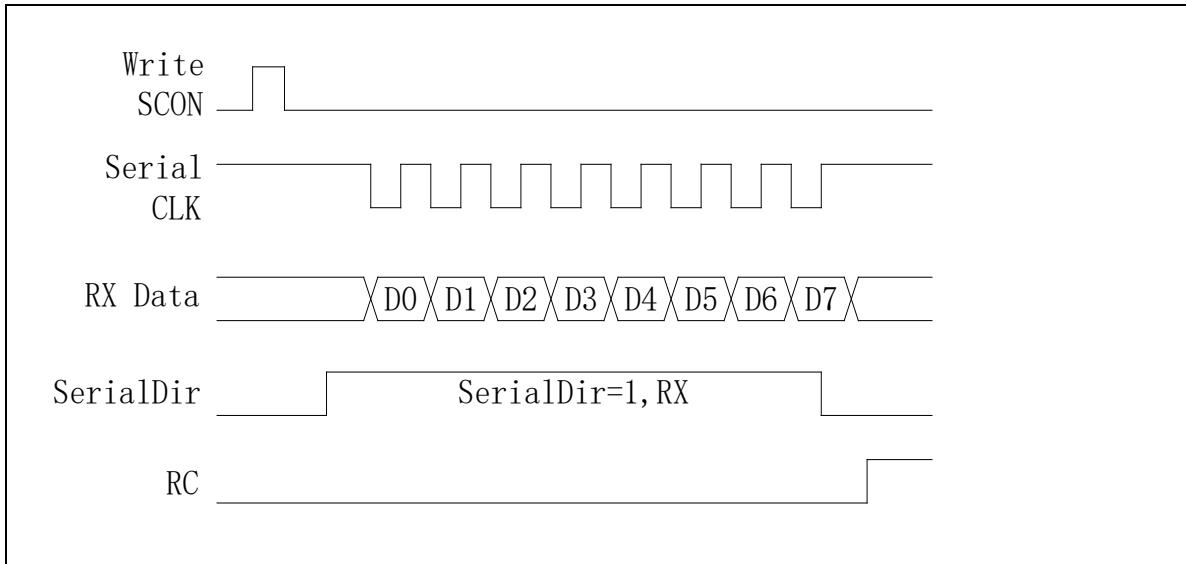


图 22-3 Mode0 接收数据

22.3.2.3 Mode 1 Data Transmission and Reception Description

发送数据时，与 LPUARTx_SCON.REN 的值无关，将所发送数据写入 LPUARTx_SBUF 寄存器中，数据就会从 TXD 移出（低位在先，高位在后）。

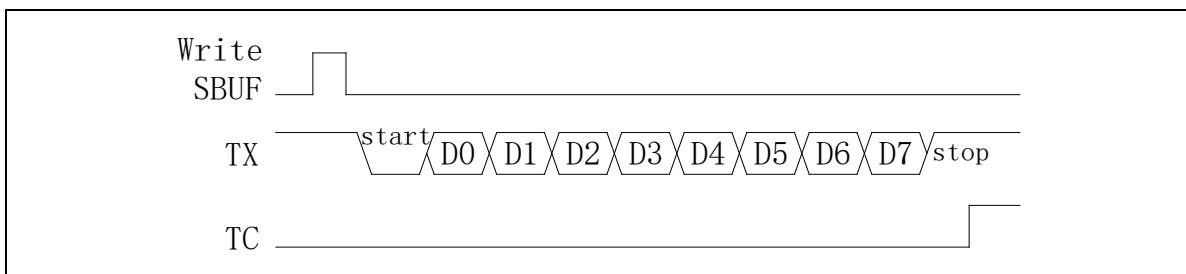


图 22-4 Mode1 发送数据

接收数据时，将 LPUARTx_SCON.REN 位置 1，并将 LPUARTx_ISR.RC 位清 0。开始接收 RXD 上数据（低位在先，高位在后），当接收完毕，可以从 LPUARTx_SBUF 寄存器读出。

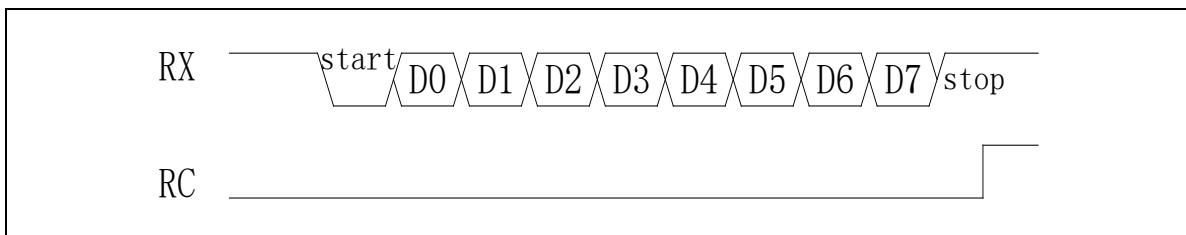


图 22-5 Mode1 接收数据

22.3.2.4 Mode2 Data Transmission and Reception Description

发送数据时，与 LPUARTx_SCON.REN 的值无关，将所发送数据写入 LPUARTx_SBUF 寄存器中，数据就会从 TXD 移出（低位在先，高位在后）。

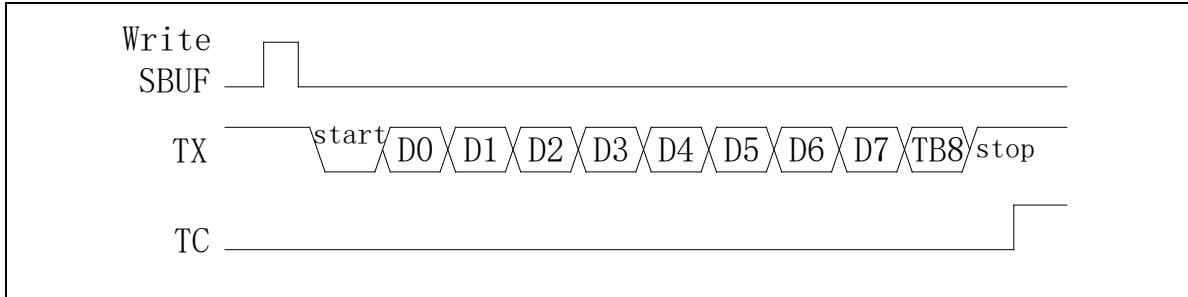


图 22-6 Mode2 发送数据

接收数据时，需将 LPUARTx_SCON.REN 位置 1，并将 LPUARTx_ISR.RC 位清 0。开始接收 RXD 上的数据（低位在先，高位在后），当接收完毕，可以从 LPUARTx_SBUF 寄存器读取。

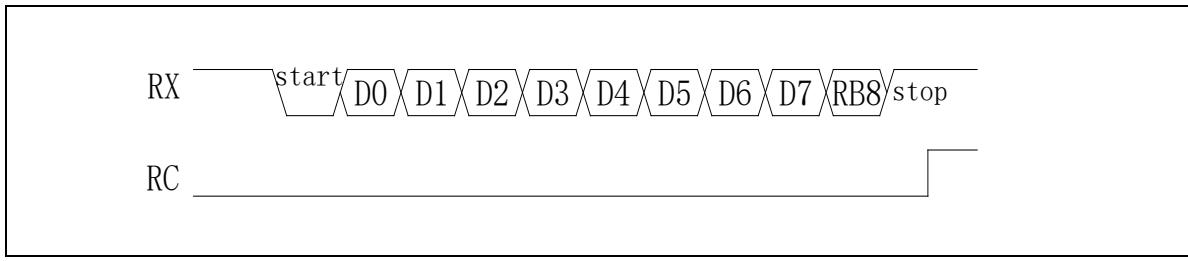


图 22-7 Mode2 接收数据

22.3.2.5 Mode3 Data Transmission and Reception Description

发送数据时，与 LPUARTx_SCON.REN 的值无关，将所发送数据写入 LPUARTx_SBUF 寄存器中，数据就会从 TXD 移出（低位在先，高位在后）。

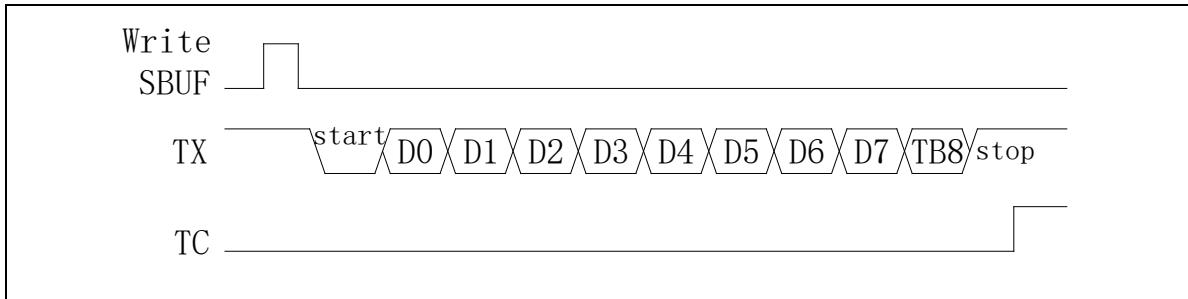


图 22-8 Mode3 发送数据

接收数据时，将 LPUARTx_SCON.REN 位置 1，并将 LPUARTx_ISR.RC 位清 0。开始接收 RXD 上数据（低位在先，高位在后），当接收完毕，可以从 LPUARTx_SBUF 寄存器读出。

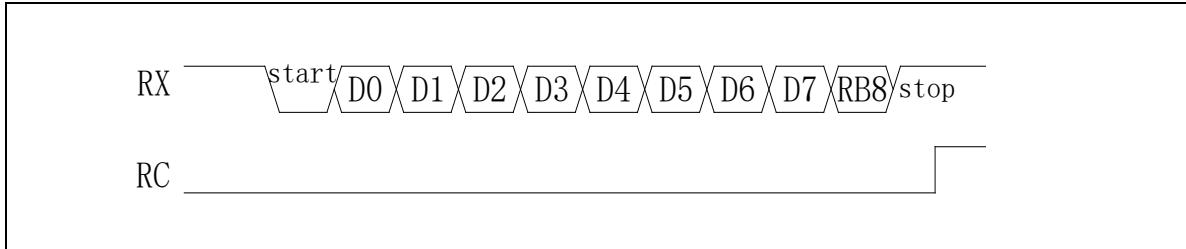


图 22-9 Mode3 接收数据

22.3.2.6 Single-wire half-duplex data transmission and reception description

当 LPUARTx_SCON.HDSEL=1'b1，可进入单线半双工传输模式：

- TX 与 RX 信号线在模块内部相连。
- RX 信号不再使用，接收发送的数据都经由 TX 信号完成。TX 信号的方向控制由硬件逻辑完成，无需软件控制。
- 当发送缓存为空时，TX 信号始终为输入（接收状态）。一旦向发送缓存填入一个数据，TX 信号变为输出（发送状态）。当发送完成，发送缓存变为空，TX 信号又回到输入（接收状态）。
- 无数据传输时，TXD 引脚始终处于输入状态。用户程序需配置该引脚为开漏输出并外接上拉电阻。

注意：

1. 即使模块处于接收过程中，只要发送缓存被填入数据，TX 信号立即变为输出（发送状态）。接收过程将被打断，并得到一个错误的接收数据。

2. 单线半双工模式与 Mode0 模式的异同：

同：两者都是用一根信号分时复用地完成了数据收发，

异：Mode0 为同步传输，有同步时钟。而单线半双工模式为异步传输，所有的波特率计算还是遵照 Mode1~Mode3 的波特率公式，仅仅是收发功能复用到了 TX 这一根信号上面。

22.3.3 Baud Rate Generation

Mode0~Mode3 生成波特率的公式不尽相同，详见下方所示。

$$\text{Mode0 波特率生成公式: } \text{BaudRate} = \frac{f_{SCLK}}{12}$$

$$\text{Mode1 波特率生成公式: } \text{BaudRate} = \frac{f_{SCLK}}{\text{OVER} * \text{SCNT}}$$

$$\text{Mode2 波特率生成公式: } \text{BaudRate} = \frac{f_{SCLK}}{\text{OVER}}$$

$$\text{Mode3 波特率生成公式: } \text{BaudRate} = \frac{f_{SCLK}}{\text{OVER} * \text{SCNT}}$$

注：

- f_{SCLK} 代表当前 SCLK 的 频率；
- OVER 的定义详见 LPUARTx_SCON；
- SCNT 的定义详见 LPUARTx_SCNT。

22.3.3.1 Mode1/Mode3 baud rate setting example

表 22-3 SCL 为 4MHz 波特率计算表

波特率	SCLK = 4 MHz								
	OVER4			OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	417	2398.08	-0.08%	208	2403.85	0.16%	104	2403.85	0.16%
4800	208	4807.69	0.16%	104	4807.69	0.16%	52	4807.69	0.16%
9600	104	9615.38	0.16%	52	9615.38	0.16%	26	9615.38	0.16%
19200	52	19230.77	0.16%	26	19230.77	0.16%	13	19230.77	0.16%
38400	26	38461.54	0.16%	13	38461.54	0.16%	7	35714.29	-6.99%
57600	17	58823.53	2.12%	9	55555.56	-3.55%	4	62500.00	8.51%
76800	13	76923.08	0.16%	7	71428.57	-6.99%	3	83333.33	8.51%
115200	9	111111.11	-3.55%	4	125000.00	8.51%	2	125000.00	8.51%
128000	8	125000.00	-2.34%	4	125000.00	-2.34%	2	125000.00	-2.34%
256000	4	250000.00	-2.34%	2	250000.00	-2.34%	1	250000.00	-2.34%
1000000	1	1000000.00	0.00%	1	500000.00	-50.00%	0	/	/

表 22-4 SCLK 为 8MHz 波特率计算表

波特率	SCLK = 8 MHz								
	OVER4			OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	833	2400.96	0.04%	417	2398.08	-0.08%	208	2403.85	0.16%
4800	417	4796.16	-0.08%	208	4807.69	0.16%	104	4807.69	0.16%
9600	208	9615.38	0.16%	104	9615.38	0.16%	52	9615.38	0.16%
19200	104	19230.77	0.16%	52	19230.77	0.16%	26	19230.77	0.16%
38400	52	38461.54	0.16%	26	38461.54	0.16%	13	38461.54	0.16%
57600	35	57142.86	-0.79%	17	58823.53	2.12%	9	55555.56	-3.55%
76800	26	76923.08	0.16%	13	76923.08	0.16%	7	71428.57	-6.99%
115200	17	117647.06	2.12%	9	111111.11	-3.55%	4	125000.00	8.51%
128000	16	125000.00	-2.34%	8	125000.00	-2.34%	4	125000.00	-2.34%
256000	8	250000.00	-2.34%	4	250000.00	-2.34%	2	250000.00	-2.34%
1000000	2	1000000.00	0.00%	1	1000000.00	0.00%	1	500000.00	-50.00%
2000000	1	2000000.00	0.00%	1	1000000.00	-50.00%	0	/	/

表 22-5 SCLK 为 16MHz 波特率计算表

波特率	SCLK = 16 MHz								
	OVER4			OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	1667	2399.52	-0.02%	833	2400.96	0.04%	417	2398.08	-0.08%
4800	833	4801.92	0.04%	417	4796.16	-0.08%	208	4807.69	0.16%
9600	417	9592.33	-0.08%	208	9615.38	0.16%	104	9615.38	0.16%
19200	208	19230.77	0.16%	104	19230.77	0.16%	52	19230.77	0.16%
38400	104	38461.54	0.16%	52	38461.54	0.16%	26	38461.54	0.16%
57600	69	57971.01	0.64%	35	57142.86	-0.79%	17	58823.53	2.12%
76800	52	76923.08	0.16%	26	76923.08	0.16%	13	76923.08	0.16%
115200	35	114285.71	-0.79%	17	117647.06	2.12%	9	111111.11	-3.55%
128000	31	129032.26	0.81%	16	125000.00	-2.34%	8	125000.00	-2.34%
256000	16	250000.00	-2.34%	8	250000.00	-2.34%	4	250000.00	-2.34%
1000000	4	1000000.00	0.00%	2	1000000.00	0.00%	1	1000000.00	0.00%
2000000	2	2000000.00	0.00%	1	2000000.00	0.00%	1	1000000.00	-50.00%
4000000	1	4000000.00	0.00%	1	2000000.00	-50.00%	0	/	/

表 22-6 SCLK 为 24MHz 波特率计算表

波特率	SCLK = 24 MHz								
	OVER4			OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	2500	2400.00	0.00%	1250	2400.00	0.00%	625	2400.00	0.00%
4800	1250	4800.00	0.00%	625	4800.00	0.00%	313	4792.33	-0.16%
9600	625	9600.00	0.00%	313	9584.66	-0.16%	156	9615.38	0.16%
19200	313	19169.33	-0.16%	156	19230.77	0.16%	78	19230.77	0.16%
38400	156	38461.54	0.16%	78	38461.54	0.16%	39	38461.54	0.16%
57600	104	57692.31	0.16%	52	57692.31	0.16%	26	57692.31	0.16%
76800	78	76923.08	0.16%	39	76923.08	0.16%	20	75000.00	-2.34%
115200	52	115384.62	0.16%	26	115384.62	0.16%	13	115384.62	0.16%
128000	47	127659.57	-0.27%	23	130434.78	1.90%	12	125000.00	-2.34%
256000	23	260869.57	1.90%	12	250000.00	-2.34%	6	250000.00	-2.34%
1000000	6	1000000.00	0.00%	3	1000000.00	0.00%	2	750000.00	-25.00%
2000000	3	2000000.00	0.00%	2	1500000.00	-25.00%	1	1500000.00	-25.00%
6000000	1	6000000.00	0.00%	1	3000000.00	-50.00%	0	/	/

表 22-7 SCLK 为 32MHz 波特率计算表

波特率	SCLK = 32 MHz								
	OVER4			OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	3333	2400.24	0.01%	1667	2399.52	-0.02%	833	2400.96	0.04%
4800	1667	4799.04	-0.02%	833	4801.92	0.04%	417	4796.16	-0.08%
9600	833	9603.84	0.04%	417	9592.33	-0.08%	208	9615.38	0.16%
19200	417	19184.65	-0.08%	208	19230.77	0.16%	104	19230.77	0.16%
38400	208	38461.54	0.16%	104	38461.54	0.16%	52	38461.54	0.16%
57600	139	57553.96	-0.08%	69	57971.01	0.64%	35	57142.86	-0.79%
76800	104	76923.08	0.16%	52	76923.08	0.16%	26	76923.08	0.16%
115200	69	115942.03	0.64%	35	114285.71	-0.79%	17	117647.06	2.12%
128000	63	126984.13	-0.79%	31	129032.26	0.81%	16	125000.00	-2.34%
256000	31	258064.52	0.81%	16	250000.00	-2.34%	8	250000.00	-2.34%
1000000	8	1000000.00	0.00%	4	1000000.00	0.00%	2	1000000.00	0.00%
2000000	4	2000000.00	0.00%	2	2000000.00	0.00%	1	2000000.00	0.00%
4000000	2	4000000.00	0.00%	1	4000000.00	0.00%	1	2000000.00	-50.00%
8000000	1	8000000.00	0.00%	1	4000000.00	-50.00%	0	/	/

表 22-8 SCLK 为 48MHz 波特率计算表

波特率	SCLK = 48 MHz								
	OVER4			OVER8			OVER16		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	5000	2400.00	0.00%	2500	2400.00	0.00%	1250	2400.00	0.00%
4800	2500	4800.00	0.00%	1250	4800.00	0.00%	625	4800.00	0.00%
9600	1250	9600.00	0.00%	625	9600.00	0.00%	313	9584.66	-0.16%
19200	625	19200.00	0.00%	313	19169.33	-0.16%	156	19230.77	0.16%
38400	313	38338.66	-0.16%	156	38461.54	0.16%	78	38461.54	0.16%
57600	208	57692.31	0.16%	104	57692.31	0.16%	52	57692.31	0.16%
76800	156	76923.08	0.16%	78	76923.08	0.16%	39	76923.08	0.16%
115200	104	115384.62	0.16%	52	115384.62	0.16%	26	115384.62	0.16%
128000	94	127659.57	-0.27%	47	127659.57	-0.27%	23	130434.78	1.90%
256000	47	255319.15	-0.27%	23	260869.57	1.90%	12	250000.00	-2.34%
1000000	12	1000000.00	0.00%	6	1000000.00	0.00%	3	1000000.00	0.00%
2000000	6	2000000.00	0.00%	3	2000000.00	0.00%	2	1500000.00	-25.00%
4000000	4	3000000.00	0.00%	2	3000000.00	0.00%	1	3000000.00	0.00%
6000000	3	4000000.00	0.00%	2	3000000.00	/	1	3000000.00	/
12000000	1	12000000.00	0.00%	1	6000000.00	/	0	/	/

22.3.4 Frame Error Detection

当工作在 Mode1/2/3 时，LPUART 具有帧错误检测功能，如果接收数据时未在预期时间内识别出停止位，从而出现同步失效或过度的噪声，则会检测到帧错误。检测帧错误时，LPUARTx_ISR.FE 置 1。LPUARTx_ISR.FE 要及时软件清零。

22.3.5 Multi-machine communication

当工作在 Mode2/3 时，将 LPUARTx_SCON.ADRDET 位置 “1”，可开启多机通讯功能。

主机可以通过 LPUARTx_SBUF[8]来区分当前的发送帧是地址帧（LPUARTx_SBUF[8]=1）还是数据帧（LPUARTx_SBUF[8]=0）。

- 当为数据帧时，该帧数据不会存入到从机的 LPUARTx_SBUF 寄存器中，从机也不会产生接收中断。
- 当为地址帧时，由于多机通讯中自动地址识别功能已开启，使得从机可以检测接收到的地址与其自身地址是否相符合。
 - 如果地址相符合，从机会对 LPUARTx_ISR.RC 置“1”， LPUARTx_SBUF[8]置 “1”，同时将该地址帧存入到 LPUARTx_SBUF 寄存器中。从机软件看到 LPUARTx_SBUF[8]=1 并且 LPUARTx_ISR.RC=1 后，将 LPUARTx_SCON.ADEDET 位清 “0”，接受数据帧。
 - 如果地址不符合，表明主机并不是寻址该从机，从机硬件保持 LPUARTx_SBUF[8]和 LPUARTx_ISR.RC 为“0”，软件保持 LPUARTx_SCON.ADRDET 位为“1”，从机继续处于地址监听状态。

注：

- 如果有必要，也可以在 Mode1 下开启多机通讯位，此时 TB8 位由 stop 位代替。当从机接收到匹配的地址帧和有效的 stop 位时，LPUARTx_ISR.RC 会被置 “1”。

22.3.5.1 Given address

LPUART 设备的 LPUARTx_SADDR 寄存器用来表示自己的设备给定地址, LPUARTx_SADEN 寄存器是地址掩码, 当 LPUARTx_SADEN 的某一位为“0”, 可以用来定义地址中的无关位, 不参与地址匹配。这些无关位增加了寻址的灵活性, 使得主机可以同时寻址一个或者多个从机设备。

注意: 如果需要给出唯一匹配地址, LPUARTx_SADEN 寄存器必须设为 8'hFF。给定地址公式如下所示:

$$\text{GivenAddr} = \text{SADDR} \& \text{SADEN}$$

22.3.5.2 Broadcast Address

广播地址是用来同时寻址所有从机设备的, 一般广播地址为 8'hFF。

$$\text{BroadCastAddr} = \text{SADDR} | \text{SADEN}$$

22.3.5.3 Examples

假设某从机的 LPUARTx_SADDR 和 LPUARTx_SADEN 配置如下:

SADDR: 8'b01101001

SADEN: 8'b11111011

那么其给定地址和广播地址如下:

Given: 8'b01101x01

Broadcast: 8'b11111x11

可见, 主机可以用四个地址寻址到本从机, 分别是:

8'b01101001 和 8'b01101101 (given address)

8'b11111011 和 8'b11111111 (broadcast address)。

22.3.6 DMAC Hardware Handshake

LPUART 模块支持 DMAC 的硬件握手逻辑。

- 将 LPUARTx_SCON.DMACTXEN 设置为 1, 可以打开 LPUART TX 的 DMAC 硬件握手逻辑。当发送缓存为空时, LPUART 会向 DMAC 发出数据搬运请求 TX REQ。DMAC 收到该信号, 则从 DMAC 源地址搬运一个帧的数据到 LPUARTx_SBUF 中。上述步骤重复发生, 直到 DMAC 中所配置的数据长度全部搬运完毕。
- 将 LPUARTx_SCON.DMACRXEN 设置为 1, 可以打开 LPUART RX 的 DMAC 硬件握手逻辑。当一帧接收完成, LPUART 会向 DMAC 发出数据搬运请求 RX REQ。DMAC 收到该信号, 则从 LPUARTx_SBUF 中把接收数据搬运至 DMAC 的目标地址中。上述步骤重复发生, 直到 DMAC 中所配置的数据长度全部搬运完毕。

22.3.7 Hardware Flow Control

通过增加 nCTS 和 nRTS 信号可以实现 LPUART 硬件流控的功能，即 LPUART 硬件模块根据 nCTS 和 nRTS 的高低电平自动控制数据的收发，而无需通过软件来判断。两个 LPUART 模块之间的硬件流控示意图如下所示：

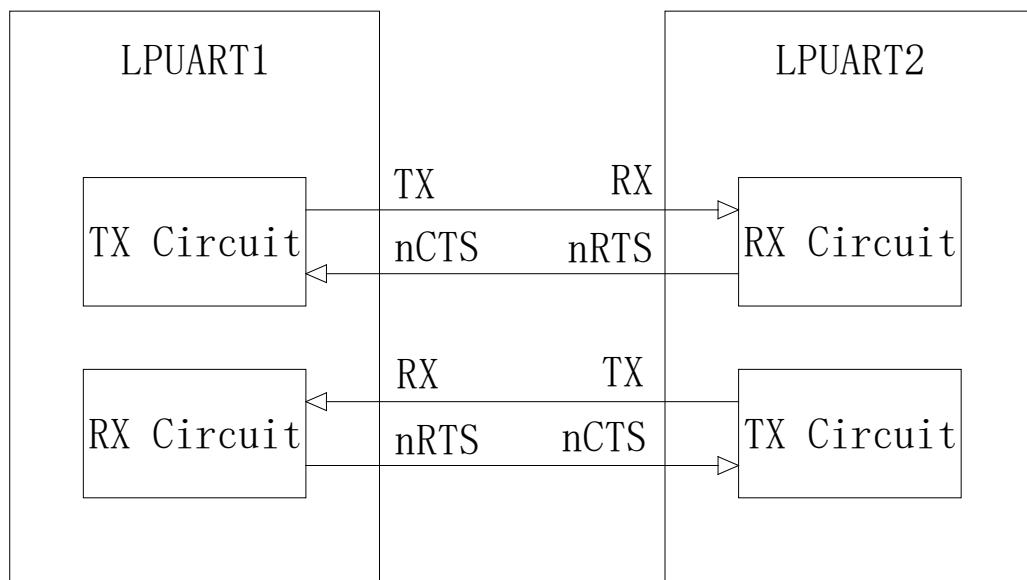


图 22-10 LPUART 硬件流控

■ nRTS 流控

nRTS 流控使能时 (LPUARTx_SCON.RTSEN 设置为 1)：

- 当 LPUART 接收缓存空时，会将 nRTS 变为有效（连接到低电平）。
- 当接收缓存满时，会将 nRTS 变为无效（连接到高电平），表明发送过程会在当前帧结束后停止。

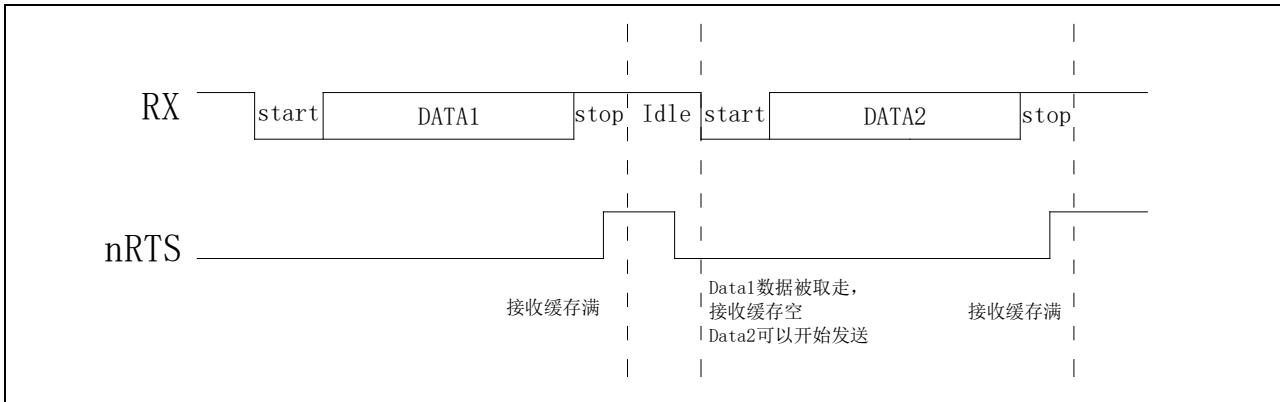


图 22-11 nRTS 硬件流控信号

■ CTS 流控

当 nCTS 流控使能时 (LPUARTx_SCON.CTSEN 设置为 1)，在 LPUART 发送下一帧数据之前，先判断 nCTS 的高低电平：

- 如果 nCTS 为有效（连接到低电平），则 LPUART 发送下一帧数据。
- 如果 nCTS 为无效（连接到高电平），则 LPUART 当前帧发送完成后，暂停下一帧的发送

当 nCTS 流控使能时，一旦 nCTS 信号发生翻转，LPUARTx_SFLAG.CTSIF 便会硬件置 1。如果 LPUARTx_SCON.CTSIE 置 1，则会产生中断。同时，nCTS 信号的高低电平会记录在 LPUARTx_SFLAG.CTS 标志位中。

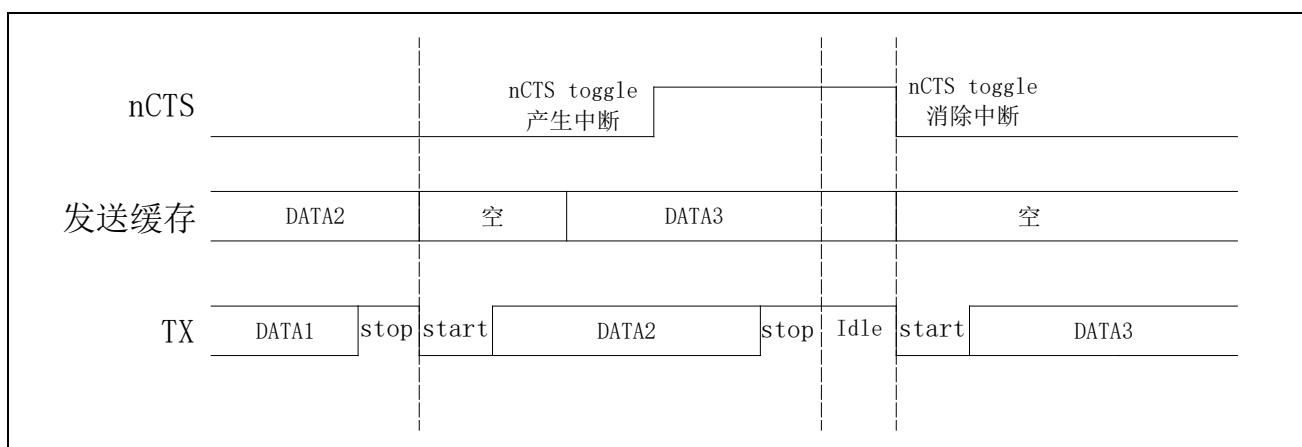


图 22-12 nCTS 硬件流控信号

22.3.8 Transceiver Buffer

■ 接收缓存

LPUART 模块接收端有一个帧 (8/9-Bit) 的接收缓存，即保持接受的数据帧直至下一帧数据的 Stop 位被接才更新该数据帧。

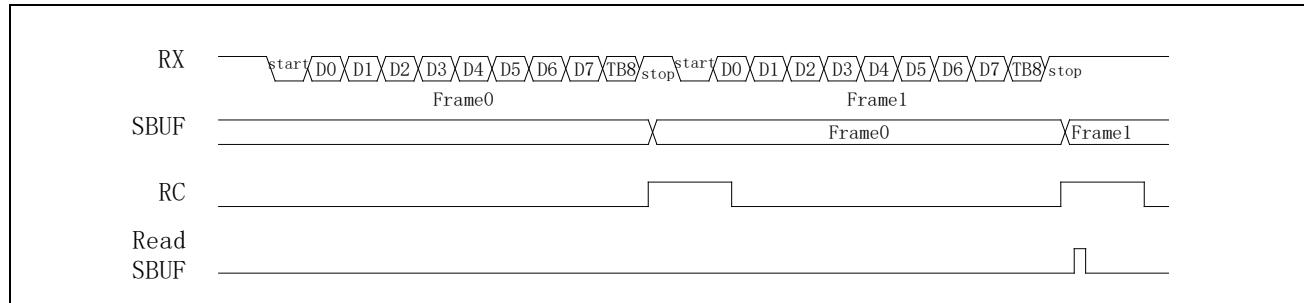


图 22-13 接收缓存

■ 发送缓存

LPUART 模块发送端有一个帧(8/9-Bit)的发送缓存，当 LPUART 在发送当前帧时，软件写入下一个发送数据到 LPUARTx_SBUF。

当 LPUARTx_ISR.TXE=0 时，表明当前发送缓存满，LPUARTx_SBUF 不能写入下一个发送数据。否则该数据会被硬件丢弃。

当 LPUARTx_ISR.TXE=1 时，表明当前发送缓存空，LPUARTx_SBUF 可以写入下一个发送数据，在完成当前数据传输后，硬件自动把发送缓存中的数据装载入移位寄存器中发送出去。

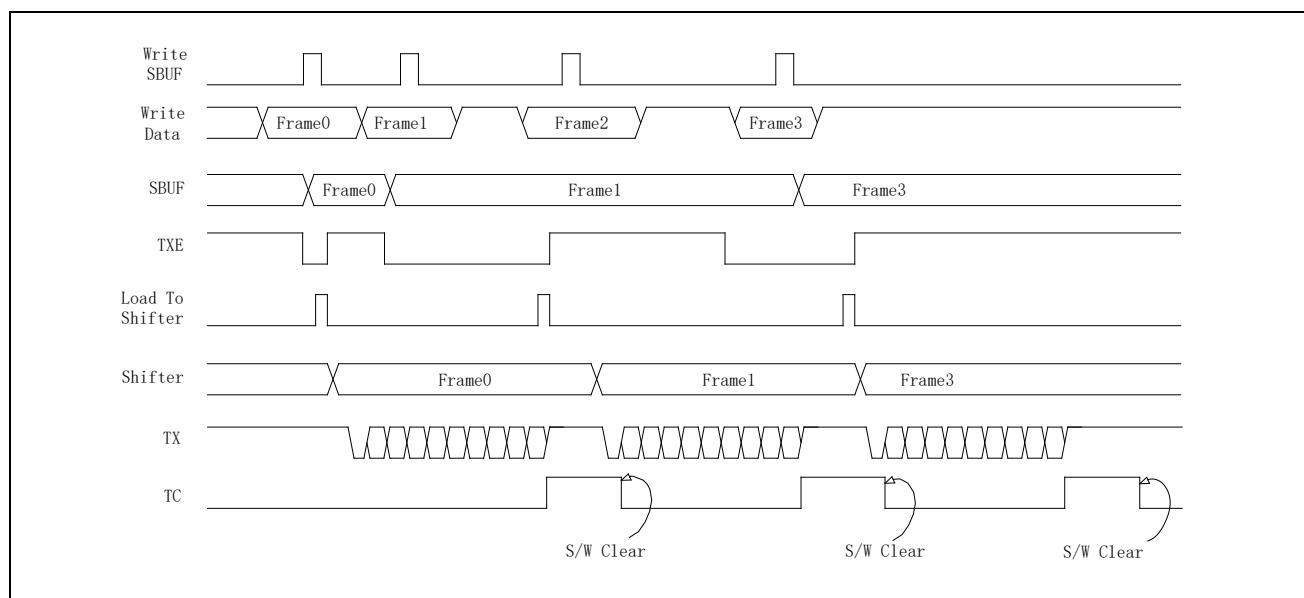


图 22-14 发送缓存

22.4 Registers

LPUART0 基地址: 0x4000 0200

LPUART1 基地址: 0x4000 4000

寄存器	偏移地址	描述
LPUARTx_SBUF	0x00	数据寄存器
LPUARTx_SCON	0x04	控制寄存器
LPUARTx_SADDR	0x08	地址寄存器
LPUARTx_SADEN	0x0C	地址掩码寄存器
LPUARTx_ISR	0x10	中断标志位寄存器
LPUARTx_ICR	0x14	中断标志位清除寄存器
LPUARTx_SCNT	0x18	波特率寄存器

22.4.1 Data Register (LPUARTx_SBUF)

偏移地址：0x00

复位值：0x0000 0000

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

位	标记	功能描述
31:9	Reserved	
8	DATA[8]	<p>在 Mode0/1 下，读取该位为 0，写入该位无效；</p> <p>在 Mode2/3 下，该位表示 Bit8 数据位，分以下两种情况：</p> <p>(1) 当硬件奇偶校验位开启时，接收时该位为接收数据奇偶校验位，校验由硬件进行，如校验出错，校验错误标志位 PE 置 1；发送时该位无效，发送数据奇偶校验位由硬件计算并发送；</p> <p>(2) 当硬件奇偶校验位关闭时，接收时该位为接收数据 Bit8；发送时该位为发送数据 Bit8；</p> <p>注意：当开启多机通讯模式，接收数据奇偶校验自动关闭；发送数据奇偶校验仍受 B8CONT 控制；</p>
7:0	DATA[7:0]	发送数据时，将发送数据写入该寄存器；接收数据时，数据接收完毕后，从该寄存器中读取。

22.4.2 Control Register (LPUARTx_SCON)

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									HDS EL	FEIE	CTSI E	CTS EN	RTS EN	DMA TXE N	DMA RXE N
									RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOPBIT	PEIE	SCLKSEL	OVE R	TXEI E	SM	ADR DET	REN	B8CONT	TCIE	RCIE					
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW					

位	标记	功能描述
31:23	Reserved	
22	HDSEL	单线半双工模式使能；0：关闭单线半双工模式；1：打开单线半双工模式；
21	FEIE	帧错误中断使能；0：关闭中断；1：打开中断；
20	CTSIE	CTS 信号翻转中断使能；0：关闭中断；1：打开中断；
19	CTSEN	硬件 flow control 信号使能位；0：关闭 flow control 信号；1：打开 flow control 信号；
18	RTSEN	
17	DMATXEN	TX DMAC 的硬件握手信号使能位；0：关闭硬件握手信号；1：打开硬件握手信号；
16	DMARXEN	RX DMAC 的硬件握手信号使能位；0：关闭硬件握手信号；1：打开硬件握手信号；
15:14	STOPBIT	stop bit 长度选择；00:1-bit; 01:1.5-bit; 10:2-bit; 11: reserved; 注意：Mode0 时虽然没有 Stop Bit，但仍需把 STOPBIT[1:0]保持为 2'b00；
13	PEIE	奇偶校验错误中断使能位；0：奇偶校验错误中断关闭；1：奇偶校验错误中断打开； 在系统时钟使用高速时钟与 LPUART 使用低速时钟情况下，由于需要不同时钟域的同步，硬件奇偶校验与数据接收中断由于停止位设置不同，奇偶校验错误中断会提前或落后于数据接收中断。使用奇偶校验中断使能时请注意。 建议如下： 关闭 PEIE 中断使能，接收数据中断后通过接收的数据位 SBUF.BIT8 软件判断是否出现奇偶校验错误。
12:11	SCLKSEL	传输时钟选择位：00, 01: PCLK; 10: XTL; 11: RCL;
10:9	OVER	Mode0: 无效； Mode1/3: 00: 16 采样分频； 01: 8 采样分频； 10: 4 采样分频； 11: 保留； Mode2: 00: 32 采样分频； 01: 16 采样分频； 10: 8 采样分频； 11: 保留；
8	TXEIE	TX 空中断使能位；0：TX Buffer 空中断关闭；1：TX Buffer 空中断打开；
7:6	SM	工作模式；00: mode0; 01: mode1; 10: mode2; 11: mode3;
5	ADRDET	多机通讯地址自动识别使能位；0：关闭；1：打开；
4	REN	Mode0: 0: 发送；1: 接收； Mode1/2/3: 0: 发送；1: 接收/发送；
3:2	B8CONT	Bit8 数据控制位； 00：由软件读写 SBUF[8]来决定；01：硬件偶校验；

		10：硬件奇校验；	11：保留；
1	TCIE	发送中断使能位；0：发送中断关闭；1：发送中断打开；	
0	RCIE	接收中断使能位；0：接收中断关闭；1：接收中断打开；	

22.4.3 Address Register (LPUARTx_SADDR)

偏移地址：0x08

复位值：0x0000 0000

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

位	标记	功能描述
31:8	Reserved	
7:0	SADDR	从机设备地址寄存器

22.4.4 Address Mask Register (LPUARTx_SADEN)

偏移地址：0x0C

复位值：0x0000 0000

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

位	标记	功能描述
31:8	Reserved	
7:0	SADEN	从机设备地址掩码寄存器

22.4.5 Flag Register (LPUARTx_ISR)

偏移地址：0x10

复位值：0x0000 0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CTS	CTSIF	PE	TXE	FE	TC	RC	
								RO	RO	RO	RO	RO	RO	RO	RO

位	符号	描述
31:7	Reserved	
6	CTS	CTS 信号标志位；硬件置 1；硬件清零；0: CTS 信号为低电平；1: CTS 信号为高电平；
5	CTSIF	CTS 中断标志位；硬件置 1；软件清零；0: CTS 信号没有发生反转；1: CTS 信号发生反转；
4	PE	奇偶校验错误标志位；硬件置 1；软件清零；0: 无奇偶校验错误；1: 奇偶校验错误；
3	TXE	Tx Buffer 空标志位；硬件置 1；硬件清零；0: Tx Buffer 非空；1: Tx Buffer 空
2	FE	帧错误标志位；0: 硬件置 1；软件清零；
1	TC	发送完毕中断标志位；硬件置 1；软件清零；0: 发送未完成；1: 发送完毕；
0	RC	接收完毕中断标志位；硬件置 1；软件清零；0: 接收未完成；1: 接收完成；

22.4.6 Flag Clear Register (LPUARTx_ICR)

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CTSIFCF	PECF	Res.	FECF	TCCF	RCCF		
								R1W0	R1W0		R1W0	R1W0	R1W0		

位	标记	功能描述
31:6	Reserved	
5	CTSIFCF	CTSIF 标志清楚位；写 0 清除；写 1 无效；
4	PECF	PE 标志清楚位；写 0 清除；写 1 无效；
3	Reserved	
2	FECF	FE 标志清除位；写 0 清零；写 1 无效；
1	TCCF	TC 标志清除位；写 0 清零；写 1 无效；
0	RCCF	RC 标志清除位；写 0 清零；写 1 无效；

22.4.7 Baud Rate Register (LPUARTx_SCNT)

偏移地址：0x18

复位值：0x0000 0000

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

位	标记	功能描述
31:16	Reserved	
15:0	SCNT	波特率计数器

23 Cyclic Redundancy Check (CRC)

23.1 Overview

循环冗余校验 (CRC) 计算单元将数据流或数据块作为输入，在生成多项式的控制下生成一个输出数。该输出数常用于验证数据传输或存储的正确性和完整性。本模块支持计算 CRC 值和检验 CRC 值。

23.2 Main Features

- 一种执行标准：ISO/IEC13239
- 两种编码方式：CRC-16，CRC-32
- 三种写入位宽：8bit, 16bit, 32bit
- 两种工作模式：CRC 编码模式、CRC 校验模式
- CRC-16 多项式： $x^{16} + x^{12} + x^5 + 1$
- CRC-32 多项式： $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

23.3 Functional Description

23.3.1 Operating Mode

本模块支持两种工作模式：CRC 编码模式、CRC 校验模式。

CRC 编码模式是指向 CRC 模块输入一定数量的原始数据，获取 CRC 模块生成的输出值 (CRC_RESULT)。
CRC 校验模式是指向 CRC 模块输入一定数量的原始数据+CRC 校验值，验证原始数据与 CRC 校验值是否匹配 (CRC_CR.FLAG)。

23.3.2 Encoding

本模块支持两种编码方式 CRC-16、CRC-32，其计算结果分别为 16 比特和 32 比特。通过 CRC_CR.CR 配置需要使用的编码方式。

CRC-16 多项式： $x^{16} + x^{12} + x^5 + 1$ 。

CRC-32 多项式： $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ 。

23.3.3 Write Bit Width

本模块支持三种写入位宽：8bit, 16bit, 32bit。不同位宽的写入需要符合“位宽一致、先低后高”的原则，即“每次写入数据，都必须写入到与本次有效数据位宽相等的寄存器中，并且较低位数据的写入先于较高位数据”。

下方展示了同一序列数据采用三种位宽进行写入的方法，其输出结果相同。

- 8bit 位宽写入：0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77

- 16bit 位宽写入: 0x1100, 0x3322, 0x5544, 0x7766
- 32bit 位宽写入: 0x33221100, 0x77665544

23.4 Programming Examples

23.4.1 CRC-16 Encoding Mode

- Step 1: 向 CRC_CR.CR 写入 0x00, 选择 CRC-16。
- Step 2: 向 CRC_RESULT 写入 0xFFFF, 初始化 CRC 计算。
- Step 3: 将待编码的原始数据依次写入 CRC_DATA 寄存器, 写入位宽可选择 8bit、16bit、32bit。
- Step 4: 读取 CRC_RESULT[15:0]以获取 CRC 值。

23.4.2 CRC-16 Check Mode

- Step 1: 向 CRC_CR.CR 写入 0x00, 选择 CRC-16。
- Step 2: 向 CRC_RESULT 写入 0xFFFF, 初始化 CRC 计算。
- Step 3: 将已编码的数据序列依次写入 CRC_DATA 寄存器, 写入位宽可选择 8bit、16bit、32bit。
- Step 4: 根据 CRC_CR.FLAG 的值判定已编码的数据序列是否被篡改。

23.4.3 CRC-32 Encoding Mode

- Step 5: 向 CRC_CR.CR 写入 0x01, 选择 CRC-32。
- Step 6: 向 CRC_RESULT 写入 0xFFFFFFFF, 初始化 CRC 计算。
- Step 7: 将待编码的原始数据依次写入 CRC_DATA 寄存器, 写入位宽可选择 8bit、16bit、32bit。
- Step 8: 读取 CRC_RESULT[31:0]以获取 CRC 值。

23.4.4 CRC-32 Check Mode

- Step 5: 向 CRC_CR.CR 写入 0x01, 选择 CRC-32。
- Step 6: 向 CRC_RESULT 写入 0xFFFFFFFF, 初始化 CRC 计算。
- Step 7: 将已编码的数据序列依次写入 CRC_DATA 寄存器, 写入位宽可选择 8bit、16bit、32bit。
- Step 8: 根据 CRC_CR.FLAG 的值判定已编码的数据序列是否被篡改。

23.5 Register Description

23.5.1 Register List

基地址：0x4002 0900

寄存器	偏移地址	描述
CRC_CR	0x00	CRC 控制寄存器
CRC_RESULT	0x04	CRC 结果寄存器
CRC_DATA	0x80	CRC 数据寄存器

23.5.2 Control Register (CRC_CR)

偏移地址：0x00

复位值：0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
R															
FLA G	CR														
RO															

位	符号	功能描述
31:2	Reserved	
1	FLAG	CRC 校验结果 0: 当前 CRC 校验错误 1: 当前 CRC 校验正确
0	CR	CRC 编码方式选择 0: CRC-16 编码 1: CRC-32 编码

23.5.3 Result Register (CRC_RESULT)

偏移地址：0x04

复位值：0x0000 0000

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

位	符号	描述
31:0	RESULT	CRC 计算结果 读取 RESULT[15:0]以获取 CRC-16 的计算结果 读取 RESULT[31:0]以获取 CRC-32 的计算结果 向 RESULT[15:0]写入 0xFFFF 以初始化 CRC-16 计算 向 RESULT[31:0]写入 0xFFFFFFFF 以初始化 CRC-32 计算

23.5.4 Data Register (CRC_DATA)

偏移地址：0x80

复位值：0x0000 0000

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

位	符号	功能描述
31:0	DATA	<p>本寄存器用于写入需要运算的数据，支持 3 种写入位宽</p> <p>8bit 写入方式： * ((uint8_t *)0x40020980) = 0XX</p> <p>16bit 写入方式： * ((uint16_t *)0x40020980) = 0XXXX</p> <p>32bit 写入方式： * ((uint32_t *)0x40020980) = 0XXXXXXXX</p>

24 True Random Number Generator (TRNG)

24.1 Overview

真随机数模块产生 64 位真随机数。

24.2 Functional Block Diagram

以下示意了 TRNG 模块的数据流：

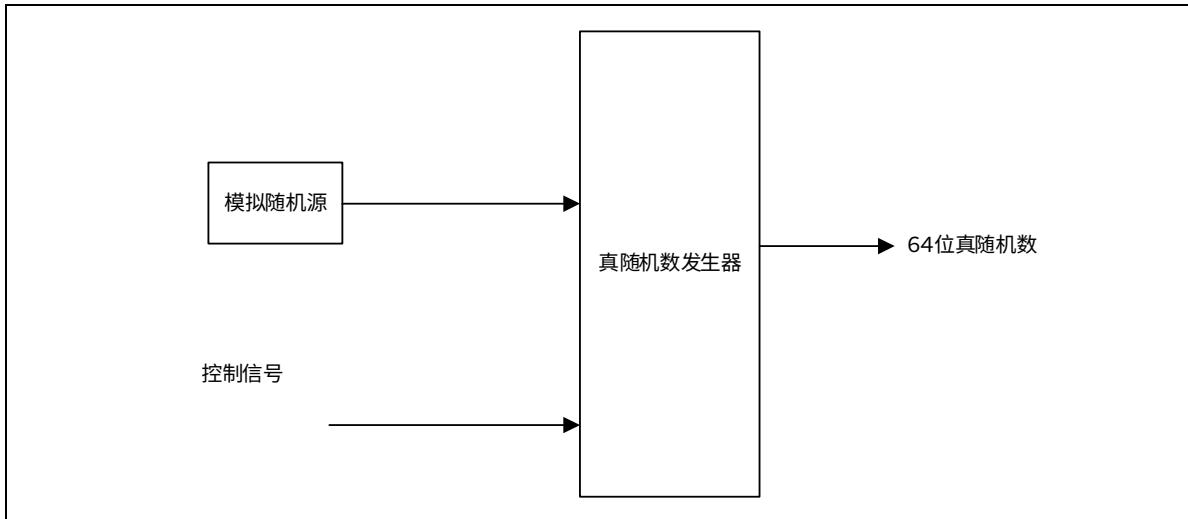


图 24-1 TRNG 数据流

24.3 功能描述

本模块采用内部的模拟随机源，每次启动都可以产生 64bits 真随机数。此外，还可以对真随机数生成的方式进行软件配置，详细内容可查看寄存器描述章节。生成的 64 位真随机数分别存放在 DATA0 和 DATA1 寄存器中。

24.4 Registers

基地址: 0x4000 4C00

寄存器	偏移地址	描述
TRNG_CR	0x00	控制寄存器
TRNG_MODE	0x04	模式寄存器
TRNG_DATA0	0x0C	数据寄存器 0
TRNG_DATA1	0x10	数据寄存器 1

24.4.1 Control Register (TRNG_CR)

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
														TRNG_RUN	TRNGcir_EN
														RW	RW

位	标记	功能描述
31:2	Reserved	
1	TRNG_RUN	软件写入“1”，开始产生新的 64bits 随机数；运行完毕后，硬件清零； 0：随机数产生完成； 1：写 1 启动随机数产生，读 1 表示随机数正在产生；
0	TRNGcir_EN	随机源电路使能位： 0：关闭随机源； 1：打开随机源；

24.4.2 Mode Register (TRNG_MODE)

偏移地址：0x04

复位值：0x0000 0010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TRNG_CNT	TRNG_FDBK	TRNG_LOAD					
								RW	RW	RW					

位	标记	描述
31:5	Reserved	
4:2	TRNG_CNT	64bits TRNG 的反馈移位次数 3'b000:移位 0 次（即输出随机源的采样值） 3'b001:移位 8 次 3'b010:移位 16 次 3'b011:移位 32 次 3'b100:移位 64 次 3'b101:移位 128 次 3'b110:移位 256 次 3'b111:Reserved
1	TRNG_FDBK	在移位操作时，64bits TRNG 的反馈信号是否与随机源进行异或操作 0: 不进行异或操作； 1: 进行异或操作；
0	TRNG_LOAD	在产生新的随机数时，64bits TRNG 是否从随机源获得新的初始值 0: 不装载新的初始值（产生伪随机数）； 1: 装载新的初始值（产生真随机数）；

24.4.3 Data Register 0 (TRNG_DATA0)

偏移地址：0x0C

复位值：---

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA0[31:16]															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA0[15:0]															
RO															

位	标记	功能描述
31:0	DATA0	软件对本寄存器读取将得到低 32 位的随机数

24.4.4 Data Register 1 (TRNG_DATA1)

偏移地址：0x10

复位值：---

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA1[31:16]															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA1[15:0]															
RO															

位	标记	功能描述
31:0	DATA1	软件对本寄存器读取将得到高 32 位的随机数

24.5 Basic Software Operations

24.5.1 Operation flow of generating 64-bit true random numbers (first time after power-on)

上电第一次生成 64bits 真随机数，需要经过以下操作：

Step1：打开随机源电路：对真随机数控制寄存器的 Bit0 (TRNG_CR.TRNGcir_En) 写入 “1”，启动随机源电路，开始输出串行真随机数。

Step2：选择重新装载初始值：将真随机数模式寄存器的 Bit0 (TRNG_MODE.TRNG_LOAD) 设置为 “1”，使新产生的真随机数的初始值从随机源获得。

Step3：选择 PRNG64 的直接反馈的方式：将真随机数模式寄存器的 Bit1 (TRNG_MODE.TRNG_FDBK) 设置为 “1”，将反馈信号与随机源异或后输入 PRNG 中。

Step4：选择 PRNG64 的移位次数：设置真随机数模式寄存器的 Bit4—Bit2 (TRNG_MODE.TRNG_CNT) 为 “110”，选择移位 256 次。

Step5：生成真随机数：软件将 “1” 写入真随机数控制寄存器的 Bit1 (TRNG_CR.TRNG_RUN)，硬件根据真随机数生成配置进行操作，在操作完成后，硬件自动将 Bit1 清为 “0”。

Step6：选择不重新装载初始值：将真随机数模式寄存器的 Bit0 (TRNGModeReg.TRNG_Load) 设置为 “0”。

Step7：选择 PRNG64 的直接反馈的方式：将真随机数模式寄存器的 Bit1 (TRNG_MODE.TRNG_FDBK) 设置为 “0”，将反馈信号直接输入 PRNG 中。

Step8：选择 PRNG64 的移位次数：设置真随机数模式寄存器的 Bit4—Bit2 (TRNG_MODE.TRNG_CNT) 为 “100”，选择移位 64 次。

Step9：生成真随机数：软件将 “1” 写入真随机数控制寄存器的 Bit1 (TRNG_CR.TRNG_RUN)，硬件根据真随机数生成配置进行操作，在操作完成后，硬件自动将 Bit1 清为 “0”。

Step10：读取真随机数：软件在查询到真随机数控制寄存器的 Bit1 (TRNG_CR.TRNG_RUN) 变为 “0” 后，通过读取真随机数数据寄存器 0(TRNG_DATA0)和真随机数数据寄存器 1(TRNG_DATA1)，得到 64Bits 真随机数。

Step11：完成真随机数的生成后，推荐选择关闭随机源电路，节省功耗：对真随机数控制寄存器的 Bit0 (TRNG_CR.TRNGcir_En) 写入 “0”，关闭随机源电路。

24.5.2 Operational procedures for generating 64-bit true random numbers (not first time after power-on)

非上电第一次生成 64bits 真随机数，需要经过以下操作：

Step1：打开随机源电路：对真随机数控制寄存器的 Bit0 (TRNG_CR.TRNGcir_En) 写入 “1”，启动随机源电路，开始输出串行真随机数。

Step2：选择不重新装载初始值：将真随机数模式寄存器的 Bit0 (TRNG_MODE.TRNG_LOAD) 设置为“0”。

Step3：选择 PRNG64 的直接反馈的方式：将真随机数模式寄存器的 Bit1 (TRNG_MODE.TRNG_FDBK) 设置为 “1”，将反馈信号与随机源异或后输入 PRNG 中。

Step4：选择 PRNG64 的移位次数：设置真随机数模式寄存器的 Bit4—Bit2 (TRNG_MODE.TRNG_CNT) 为 “110”，选择移位 256 次。

Step5：生成真随机数：软件将 “1” 写入真随机数控制寄存器的 Bit1 (TRNG_CR.TRNG_RUN)，硬件根据真随机数生成配置进行操作，在操作完成后，硬件自动将 Bit1 清为 “0”。

Step6：选择 PRNG64 的直接反馈的方式：将真随机数模式寄存器的 Bit1 (TRNG_MODE.TRNG_FDBK) 设置为 “0”，将反馈信号直接输入 PRNG 中。

Step7：选择 PRNG64 的移位次数：设置真随机数模式寄存器的 Bit4—Bit2 (TRNG_MODE.TRNG_CNT) 为 “100”，选择移位 64 次。

Step8：读取真随机数：软件在查询到真随机数控制寄存器的 Bit1 (TRNG_CR.TRNG_RUN) 变为 “0” 后，通过读取真随机数数据寄存器 0 (TRNG_Data0) 和真随机数数据寄存器 1 (TRNG_Data1)，得到 64Bits 真随机数。

如果需要继续生成新的真随机数，那么回到 Step2，直到满足要求。

Step9：完成真随机数的生成后，推荐选择关闭随机源电路，节省功耗：对真随机数控制寄存器的 Bit0 (TRNG_CR.TRNGcir_En) 写入 “0”，关闭随机源电路。

25 Advanced Encryption Standard Module (AES)

25.1 Function Definition

25.1.1 AES Algorithm Overview

AES (The Advanced Encryption Standard) 是美国国家标准技术研究所 (NIST) 在 2000 年 10 月 2 日正式宣布的新的数据加密标准。

AES 的分组长度固定为 128 位，而密钥长度支持 128、192 和 256 位。对于加密来说，其输入是一个明文分组和一个密钥，输出是一个密文分组；对解密而言，输入是一个密文分组和一个密钥，而输出是一个明文分组。此过程如图 25-1 所示：

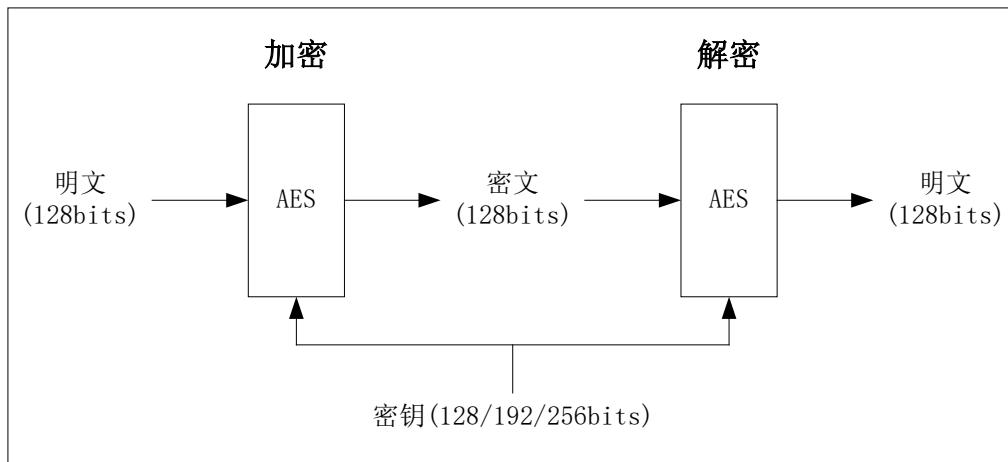


图 25-1 AES 的加解密示意图

AES 算法处理的基本单位是字节，128 位信息被分成 16 个字节，按顺序复制到一个 4×4 的矩阵中，称为状态 (state)，AES 的所有变换都是基于状态矩阵的变换，该矩阵上保存着计算的中间结果。

AES 是一个密钥迭代分组密码，包含了轮变换对状态的重复作用。AES 的轮变换由四个操作组成：SubBytes、ShiftRows、MixColumns、AddRoundKey。其中，SubBytes 包括求每个字节在 $GF(2^8)$ 中的模逆元和一个仿射变换；ShiftRows 是一个字节换位，它将状态中的行按照不同的偏移量进行循环移位；MixColumns 对状态各列进行线性变换；AddRoundKey，状态中的各字节与轮密钥进行逐位异或操作。AES 的加密流程如图 25-2 所示：

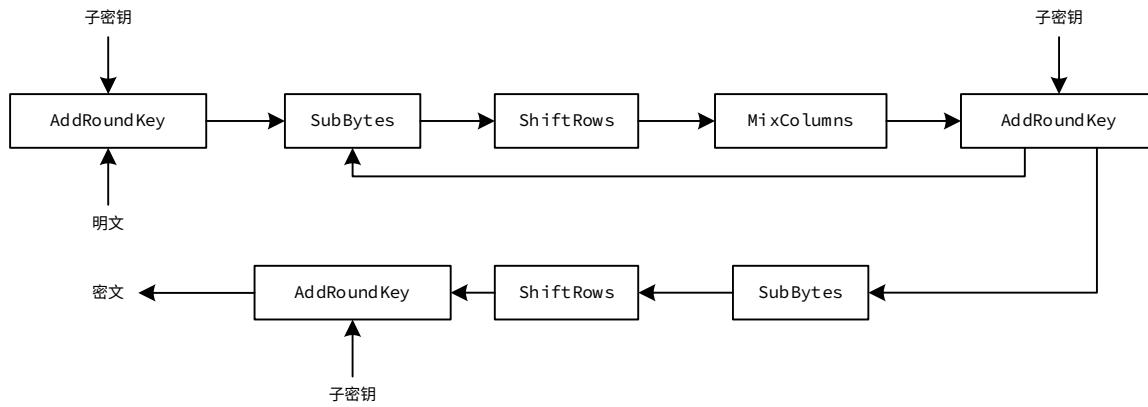


图 25-2 AES 的加密流程图

图中所用子密钥需要由初始密钥扩展而来，且密钥的扩展过程和加密过程是同步进行的。

由于明文固定为 128 位，加密过程运行的轮数就取决于密钥的长度。比如，密钥为 128 位时，运行轮数为 10 轮；密钥为 192 位时，运行轮数为 12 轮；密钥为 256 位时，运行轮数为 14 轮。除了最后一轮缺少 MixColumns 变换，其余各轮均进行完整的轮变换操作。

解密流程与加密流程有所区别，首先必须完成所有密钥的扩展，解密过程从扩展的最后一轮子密钥往回使用；然后是轮变换的四个操作变成了相应的逆运算：InvSubBytes、InvShiftRows、InvMixColumns、AddRoundKey。InvSubBytes 中的模逆运算仍然保持，但仿射变换改为逆变换；InvShiftRows 和 InvMixColumns 变成相应的逆变换；AddRoundKey 保持不变。

直接解密流程的轮变换对四个操作的调用顺序为：InvShiftRows、InvSubBytes、AddRoundKey、InvMixColumns，与加密流程的调用顺序不一致，但使用的密钥与加密流程一致；等价解密流程的轮变换对四个操作的调用顺序为：InvSubBytes、InvShiftRows、InvMixColumns、AddRoundKey，与加密流程的调用顺序完全一致，只是每一轮的子密钥需要进行 InvMixColumns 运算。

关于详细的算法表述，可以参见标准《FIPS PUB 197》。

25.1.2 AES module function description

- 执行 AES 算法标准的加密流程和解密流程，其执行结果完全符合《FIPS PUB 197》对算法原理的描述；
- 支持 128、192 和 256 位密钥。

25.2 Register Description

AES 基地址 0x40021400

表 25-1 寄存器列表

寄存器	偏移地址	描述
AES_CR	0x00 或 0x40	控制寄存器
AES_Data	0x10~0x1C	数据寄存器
AES_Key	0x20~0x3C	密钥寄存器

25.2.1 Control Register (AES_CR)

偏移地址：0x00 或 0x40

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										Keysize	Reserved	Mode	Start		
										RW		RW	RW		

位	符号	描述
31:5	Reserved	保留位，读为 0
4:3	Keysize	2'b00: 密钥长度为 128 位 2'b01: 密钥长度为 192 位 2'b10: 密钥长度为 256 位 2'b11: 密钥长度为 128 位
2	Reserved	保留位，读为 0
1	Mode	0: 加密运算 1: 解密运算
0	Start	0: 本模块运算结束或未被启动 1: 启动本模块进行运算

说明：

1. AES_CR.Start 位的操作方法是：软件对本位写入 1 后，本模块将启动运行，本次运行结束后本模块硬件会自动将本位清 0，软件查询到本位为 0 即表示本次运行完成。
2. 对本寄存器的写入操作只能在本模块不处于运算状态时（即 AES_CR.Start = 0 时）才能进行，否则硬件将自动忽略写操作。读操作则不受此限制。

25.2.2 Data Register (AES_Data)

偏移地址：0x10~0x1C

复位值：0x0000 0000

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

位	符号	描述
31:0	Data	存放 AES 算法的 128 比特明文/密文

说明：

1. 数据寄存器由四个 32 位的寄存器组成 128 位数据，用于在模块运算前存放需要被加密的明文或者需要被解密的密文，并且运算完成后存放加密后的密文或者解密后的明文。

加密运算		解密运算	
运算前	运算后	运算前	运算后
128 位明文	128 位密文	128 位密文	128 位明文

四个 32 位寄存器连接在一起组成一个 128 位的数据，读写操作时需要分别对四个寄存器进行操作。数据寄存器对应的操作顺序如下：

数据举例：0xFFEEDDCCBAA99887766554433221100

偏移地址	寄存器名称	填入数据
0x10	AES_Data0	0x33221100
0x14	AES_Data1	0x77665544
0x18	AES_Data2	0xBBAA9988
0x1C	AES_Data3	0xFFEEDDCC

2. 对于本寄存器的写入只能在本模块没有处于运算状态时（即 AES_CR.Start = 0 时）才能进行，否则硬件将自动忽略对本寄存器的写操作。
3. 对于本寄存器的读取只能在本模块没有处于运算状态时（即 AES_CR.Start = 0 时）才能进行，否则对本寄存器的读取将得到全 0。

25.2.3 Key Register (AES_Key)

偏移地址：0x20~0x3C

复位值：0x0000 0000

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

位	符号	描述
31:0	Key	存放 AES 算法的 128、192、256 比特密钥

1. 密钥寄存器由 8 个 32 位的寄存器组成，存放输入的初始密钥。写操作时需要分别对 8 个 32 位的寄存器进行操作。对应的操作顺序如下：

128 位密钥举例：0x0F0E0D0C_0B0A0908_07060504_03020100

偏移地址	寄存器名称	填入数据
0x20	AES_Key0	0x03020100
0x24	AES_Key1	0x07060504
0x28	AES_Key2	0x0B0A0908
0x2C	AES_Key3	0x0F0E0D0C

192 位密钥举例：0x17161514_13121110_0F0E0D0C_0B0A0908_07060504_03020100

偏移地址	寄存器名称	填入数据
0x20	AES_Key0	0x03020100
0x24	AES_Key1	0x07060504
0x28	AES_Key2	0x0B0A0908
0x2C	AES_Key3	0x0F0E0D0C
0x30	AES_Key4	0x13121110
0x34	AES_Key5	0x17161514

256 位密钥举例：0x1F1E1D1C_1B1A1918_17161514_13121110_0F0E0D0C_0B0A0908_07060504_03020100

偏移地址	寄存器名称	填入数据
0x20	AES_Key0	0x03020100
0x24	AES_Key1	0x07060504
0x28	AES_Key2	0x0B0A0908
0x2C	AES_Key3	0x0F0E0D0C
0x30	AES_Key4	0x13121110
0x34	AES_Key5	0x17161514
0x38	AES_Key6	0x1B1A1918
0x3C	AES_Key7	0x1F1E1D1C

- 对于本寄存器的写入只能在本模块没有处于运算状态时（即 AES_CR.Start = 0 时）才能进行，否则硬件将自动忽略对本寄存器的写操作。
- 对于本寄存器的读取只能在本模块没有处于运算状态时（即 AES_CR.Start = 0 时）才能进行，否则对本寄存器的读取将得到全 0。

25.3 Exception Mechanism

- 只支持 32 位访问，其它位宽的访问会导致系统异常，进入硬件异常中断。
- 访问 AES 模块的偏移地址大于等于 0x80 的地址，会导致系统异常，进入硬件异常中断。

25.4 Operation Instructions

本模块共有两个功能：加密、解密。对两个功能的操作有一些共同的特点，下面先介绍其共同点，再分别介绍每个功能的标准操作流程。

25.4.1 Common IP Operations

1. 在 AES 加解密过程中，数据寄存器会改变，如果下次运算的被操作数据就是本次运算的结果，那么就无需重新写入数据了。
2. 支持 128、192 和 256 位密钥，128 位密钥写入偏移地址 0x20~0x2C，192 位密钥写入偏移地址 0x20~0x34，256 位密钥写入偏移地址 0x20~0x3C。
3. 判断模块运算结束的方法：不断读取 AES_CR.Start，如果其值变为 0，则表示运算结束。

25.4.2 Encryption Operation Process

- Step 1：将待加密的 128 位数据写入数据寄存器（AES_DATA）中。
Step 2：将加密密钥写入密钥寄存器（AES_KEY）中。
Step 3：根据密钥长度设置 AES_CR.Keysize
Step 4：将 AES_CR.Mode 设置为 0，启动加密模式。
Step 5：向控制寄存器中的 AES_CR.Start 写入 1，启动模块进行运算。
 Step 3、Step 4 和 Step 5 可同时进行。
Step 6：等待 AES_CR.Start 的值恢复为 0，模块运算结束。
Step 7：读取数据寄存器（AES_DATA），获得 128 位密文。

25.4.3 Decryption Operation Process

- Step 1：将待解密的 128 位数据写入数据寄存器（AES_DATA）中。
Step 2：将解密密钥写入密钥寄存器（AES_KEY）中。
Step 3：根据密钥长度设置 AES_CR.Keysize
Step 4：将 AES_CR.Mode 设置为 1，启动解密模式。
Step 5：向控制寄存器中的 AES_CR.Start 写入 1，启动模块进行运算。
 Step 3、Step 4 和 Step 5 可同时进行。
Step 6：等待 AES_CR.Start 的值恢复为 0，模块运算结束。
Step 7：读取数据寄存器（AES_DATA），获得 128 位明文。

25.4.4 Data Examples

128 位明文: 0xFFEEDDCCBAA99887766554433221100

128 位密钥: 0x0F0E0D0C0B0A09080706050403020100

128 位密文: 0x5AC5B47080B7CDD830047B6AD8E0C469

表 25-2 128 位操作寄存器示例

加密前			
寄存器	值 (密钥)	寄存器	值 (明文)
Key0	0x03020100	Data0	0x33221100
Key1	0x07060504	Data1	0x77665544
Key2	0x0B0A0908	Data2	0xBBAA9988
Key3	0x0F0E0D0C	Data3	0xFFEEDDCC

加密后			
寄存器	值 (密钥)	寄存器	值 (密文)
Key0	0x03020100	Data0	0xD8E0C469
Key1	0x07060504	Data1	0x30047B6A
Key2	0x0B0A0908	Data2	0x80B7CDD8
Key3	0x0F0E0D0C	Data3	0x5AC5B470

128 位明文: 0xFFEEDDCCBAA99887766554433221100

192 位密钥: 0x17161514131211100F0E0D0C0B0A09080706050403020100

128 位密文: 0x5AC5B47080B7CDD830047B6AD8E0C469

表 25-3 192 位操作寄存器示例

加密前			
寄存器	值 (密钥)	寄存器	值 (明文)
Key0	0x03020100	Data0	0x33221100
Key1	0x07060504	Data1	0x77665544
Key2	0x0B0A0908	Data2	0xBBAA9988
Key3	0x0F0E0D0C	Data3	0xFFEEDDCC
Key4	0x13121110		
Key5	0x17161514		

加密后			
寄存器	值 (密钥)	寄存器	值 (密文)
Key0	0x03020100	Data0	0xA47CA9DD
Key1	0x07060504	Data1	0xE0DF4C86
Key2	0x0B0A0908	Data2	0xA070AF6E
Key3	0x0F0E0D0C	Data3	0x91710DEC
Key4	0x13121110		
Key5	0x17161514		

128 位明文：0xFFEEDDCCBAA99887766554433221100

256 位密钥：

0x1F1E1D1C1B1A191817161514131211100F0E0D0C0B0A09080706050403020100

128 位密文：0x5AC5B47080B7CDD830047B6AD8E0C469

表 25-4 192 位操作寄存器示例

加密前			
寄存器	值 (密钥)	寄存器	值 (明文)
Key0	0x03020100	Data0	0x33221100
Key1	0x07060504	Data1	0x77665544
Key2	0x0B0A0908	Data2	0xBBAA9988
Key3	0x0F0E0D0C	Data3	0xFFEEDDCC
Key4	0x13121110		
Key5	0x17161514		
Key6	0x1B1A1918		
Key7	0x1F1E1D1C		

加密后			
寄存器	值 (密钥)	寄存器	值 (密文)
Key0	0x03020100	Data0	0xCAB7A28E
Key1	0x07060504	Data1	0xBF456751
Key2	0x0B0A0908	Data2	0x9049FCAE
Key3	0x0F0E0D0C	Data3	0x8960494B
Key4	0x13121110		
Key5	0x17161514		
Key6	0x1B1A1918		
Key7	0x1F1E1D1C		

25.5 Runtime Description

本模块从启动一次运算 (AES_CR.Start 写入 1) 到该次运算结束 (AES_CR.Start 恢复到 0) 所需时间如表 25-5 所示：

表 25-5 AES 加解密运行时间

	128 位密钥	192 位密钥	256 位密钥
加密	220 cycles	260 cycles	300 cycles
解密	290 cycles	332 cycles	398 cycles

26 Liquid crystal controller (LCD)

26.1 LCD Introduction

LCD 控制器是一款适用于单色无源液晶显示器(LCD)的数字控制器/驱动器，最多具有 8 个公用端子 (COM) 和 48 个区段端子 (SEG)，用以驱动 208 (4x52)或 384 (8x48)个 LCD 图像元素。端子的确切数量取决于数据手册中所述的器件引脚。

LCD 由若干区段（像素或完整符号）组成，这些区段均可点亮或熄灭。每个区段都包含一层在两根电极之间对齐的液晶分子。当向液晶施加高于阈值电压的电压时，相应的区段可见。区段电压必须为交流，以避免液晶中出现电泳效应（这将影响显示效果）。之后，必须在区段两端生成波形以避免出现直流。

词汇表

液晶 (LCD)：无源显示面板，带有直接引向区段的端子。

公用 (COM)：连接到多个区段的电气连接端子。

偏置 (BIAS)：驱动 LCD 时使用的电压等级，定义为 1/(驱动 LCD 显示的电压等级数-1)。

区段 (SEG)：最小可视单元 (LCD 显示器上的最小组成元素，线条或点)。

占空比 (DUTY)：定义为 1/(LCD 显示器上的公用端子数)的数字。

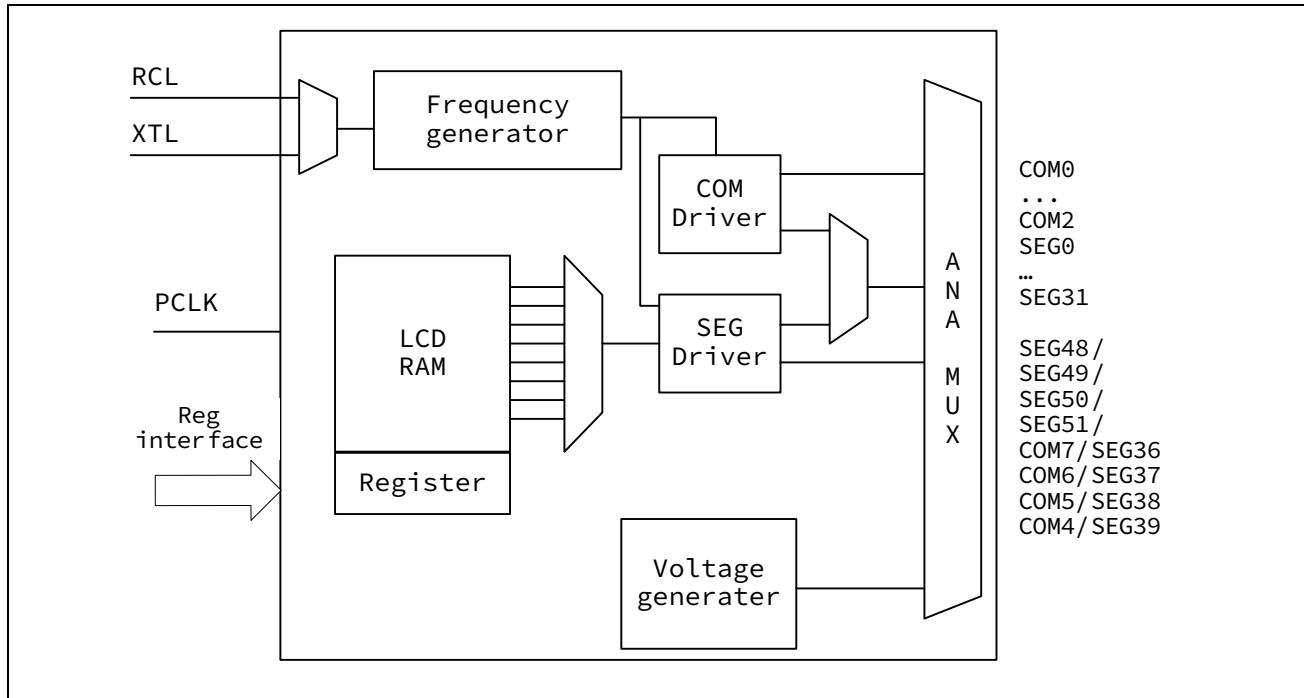
帧：写入区段的波形的一个周期。

帧速率：每秒帧数，即每秒激励 LCD 区段的次数。

26.2 LCD Main Characteristics

- 高度灵活的帧速率控制。
- 支持静态、1/2、1/3、1/4、1/6 和 1/8 占空比。
- 支持 1/2、1/3 偏置。
- 多达 16 个寄存器的 LCD 数据 RAM。
- 可通过软件配置 LCD 的对比度。
- 3 种驱动波形生成方式
 - 内部电阻分压、外部电阻分压，外部电容分压方式
 - 可通过软件配置内部电阻分压方式的功耗，从而匹配 LCD 面板所需的电容电荷
- 支持低功耗模式：LCD 控制器可在 Active、Sleep、DeepSleep 模式下进行显示。
- 可配置帧中断。
- 支持 LCD 闪烁功能且可配置多种闪烁频率。
- 未使用的 LCD 区段和公共引脚可配置为数字或模拟功能。

26.3 LCD Block Diagram



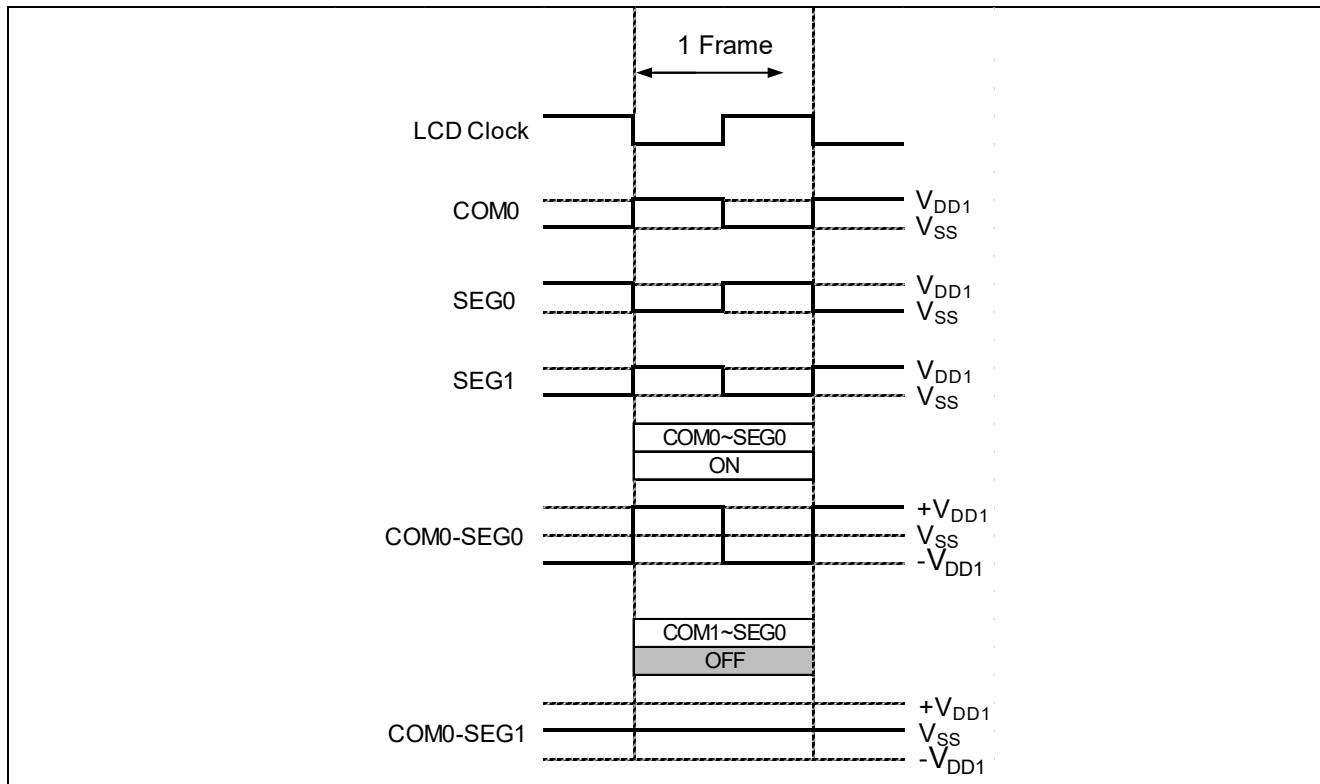
26.4 LCD Driving Waveform

LCD 支持 5 种占空比 (Duty) 的驱动波形：静态、1/2、1/3、1/4、1/6 和 1/8，由 LCD_CR0.Duty 进行设置。LCD 支持 2 种偏置 (Bias) 的驱动波形：1/2、1/3，由 LCD_CR0.Bias 进行设置。建议的组合方式如下表所示：

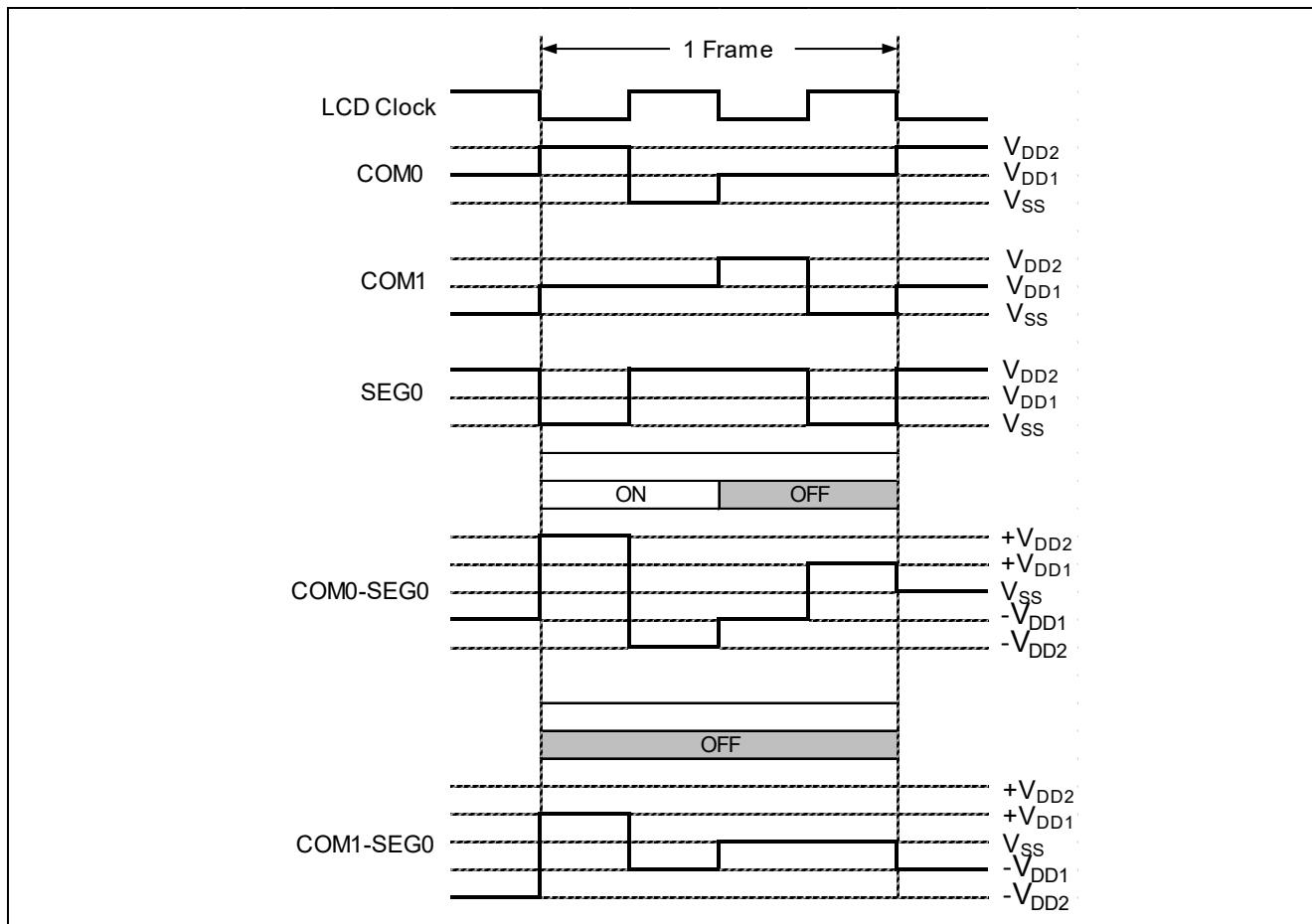
	1/2 Duty	1/3 Duty	1/4 Duty	1/6 Duty	1/8 Duty
1/2 Bias	✓	✓	不推荐	不推荐	不推荐
1/3 Bias	不推荐	不推荐	✓	✓	✓

各模式下的驱动波形如下方所示：

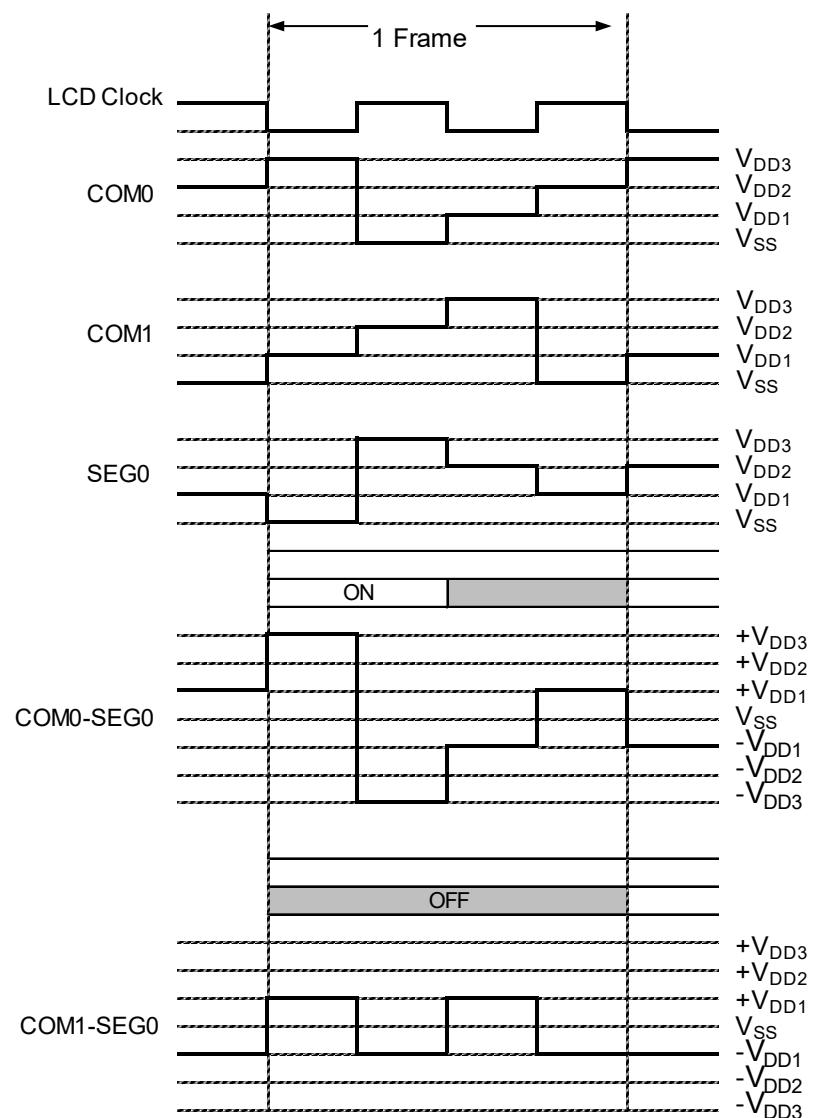
26.4.1 Static drive waveform



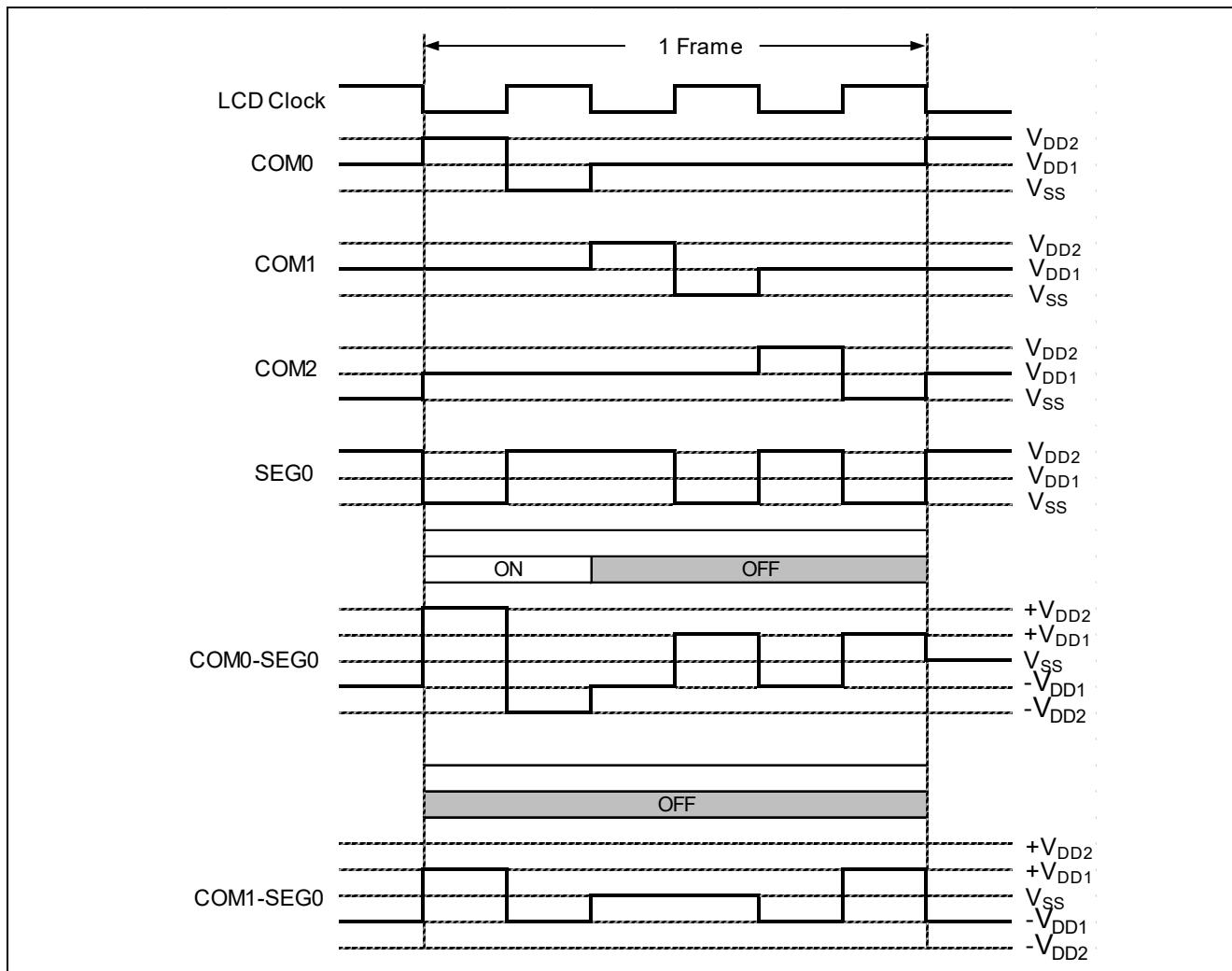
26.4.2 1/2Duty 1/2Bias Drive Waveform



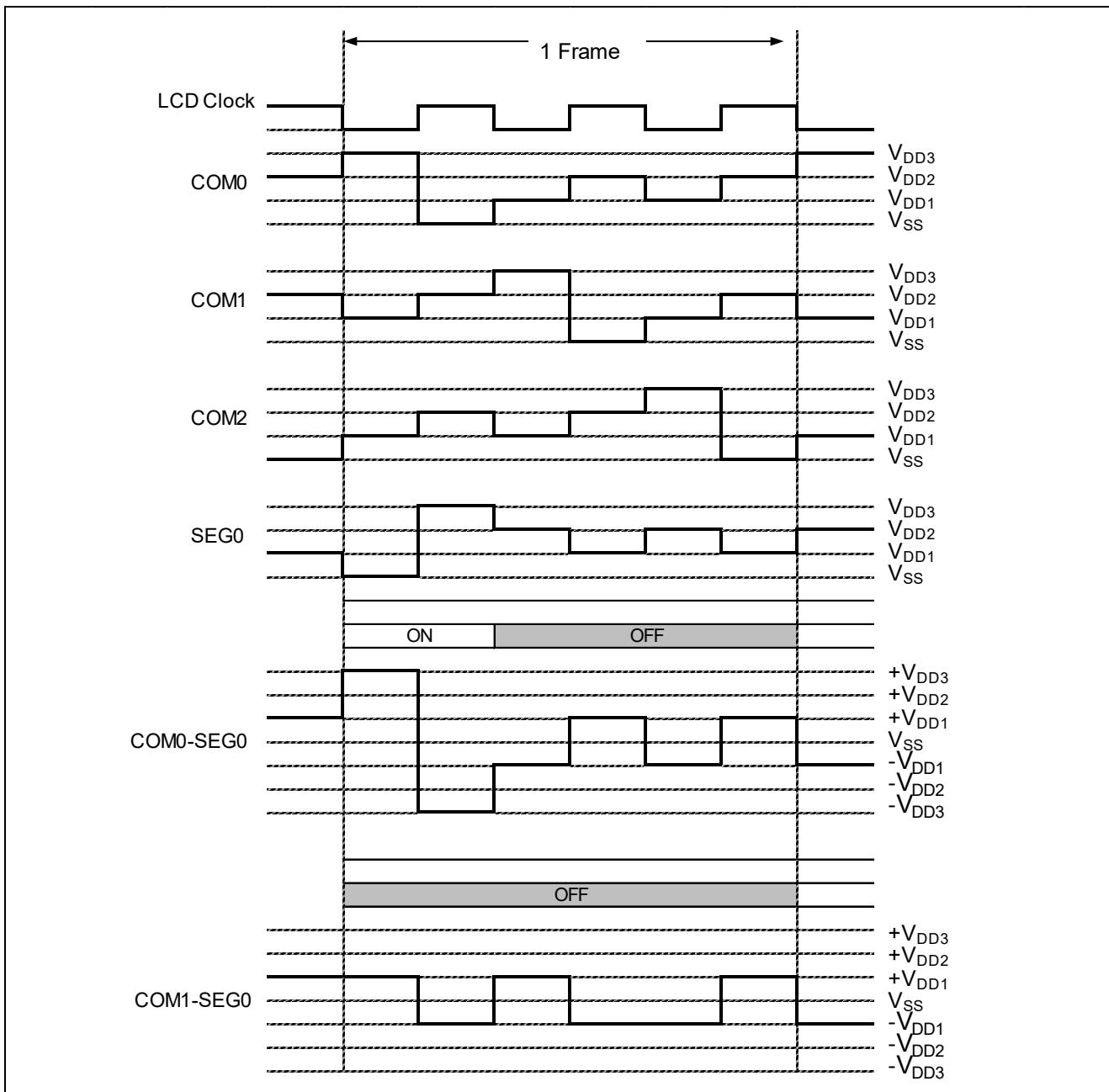
26.4.3 1/2Duty 1/3Bias Drive Waveform



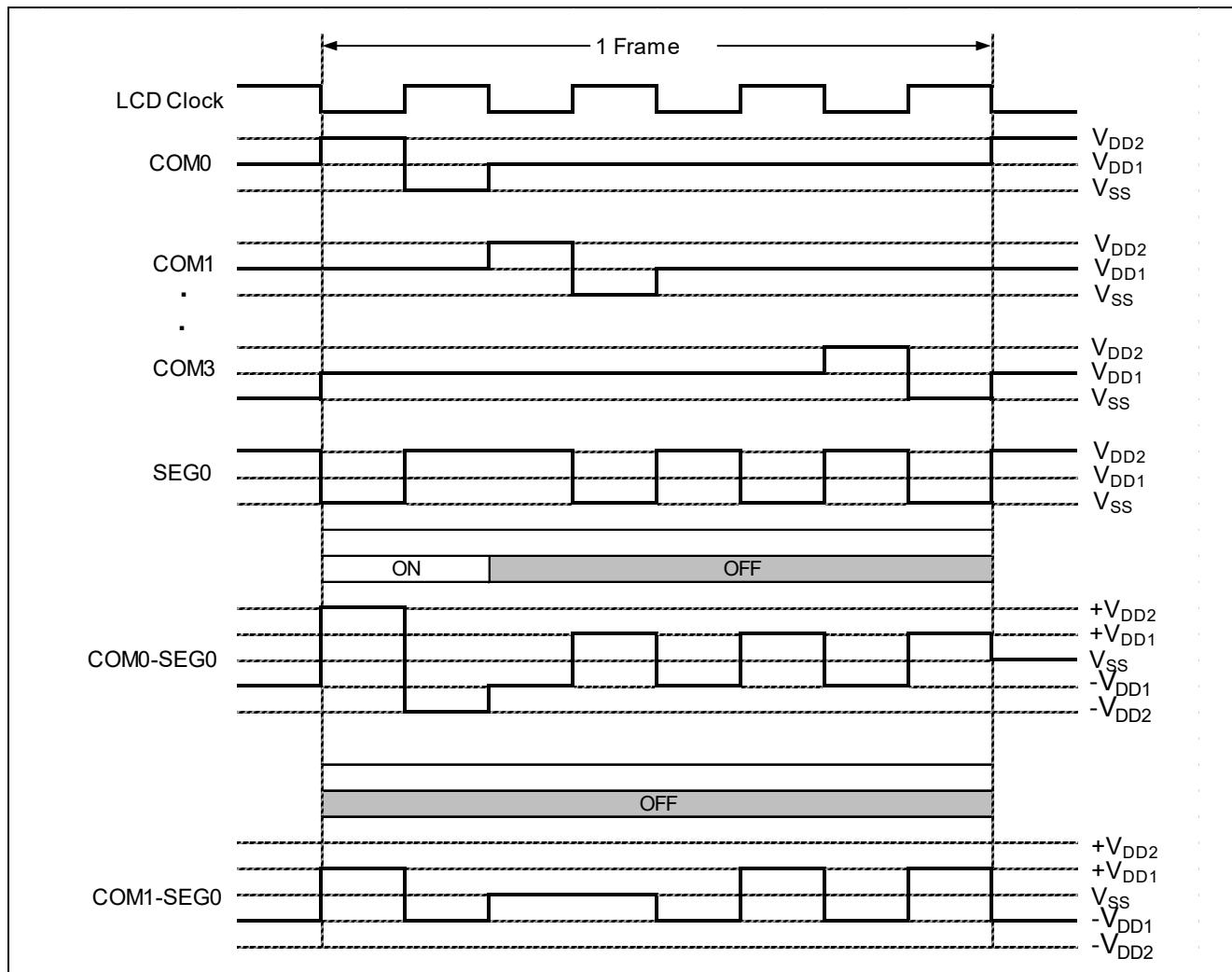
26.4.4 1/3Duty 1/2Bias Drive Waveform



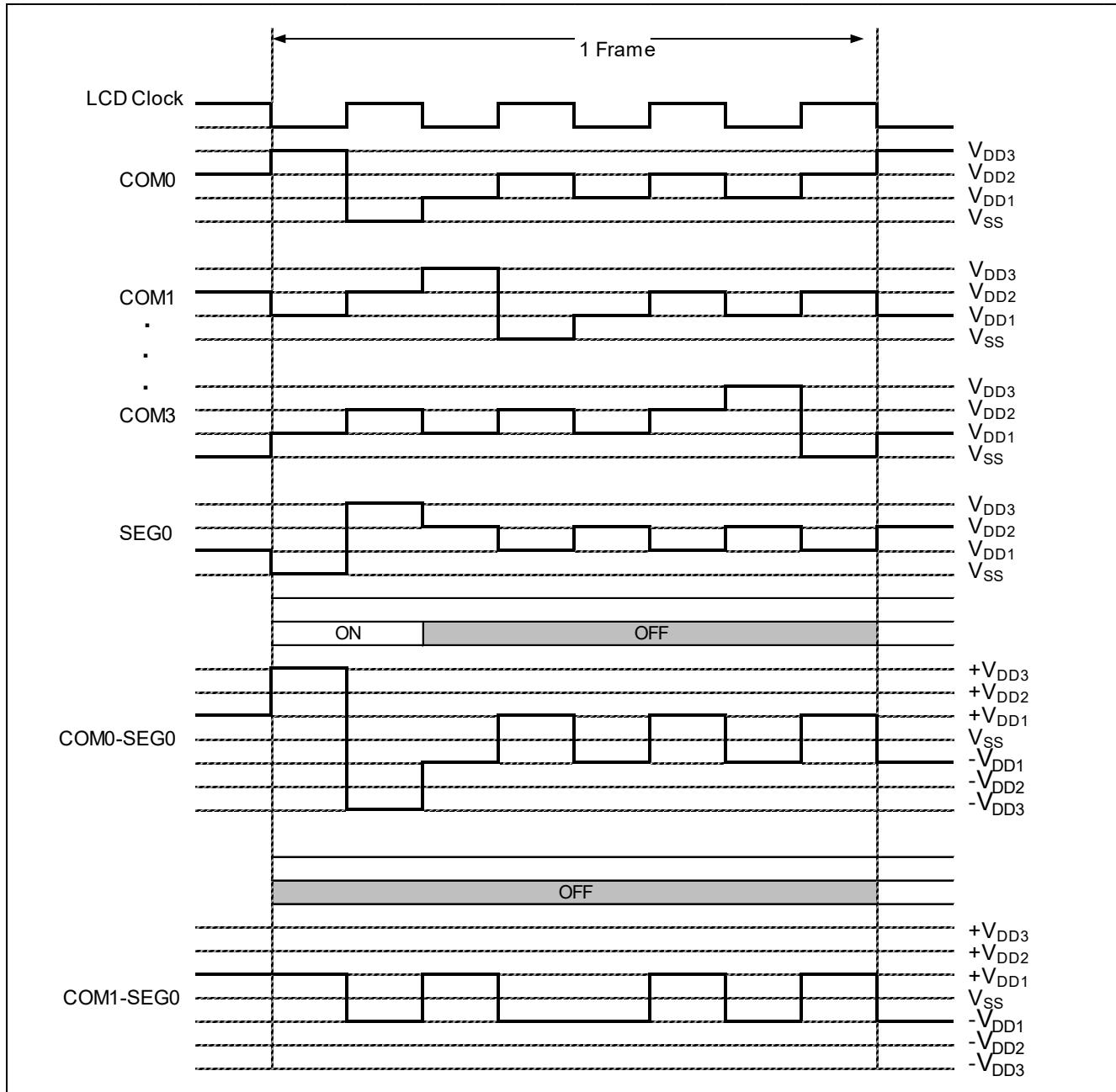
26.4.5 1/3Duty 1/3Bias Drive Waveform



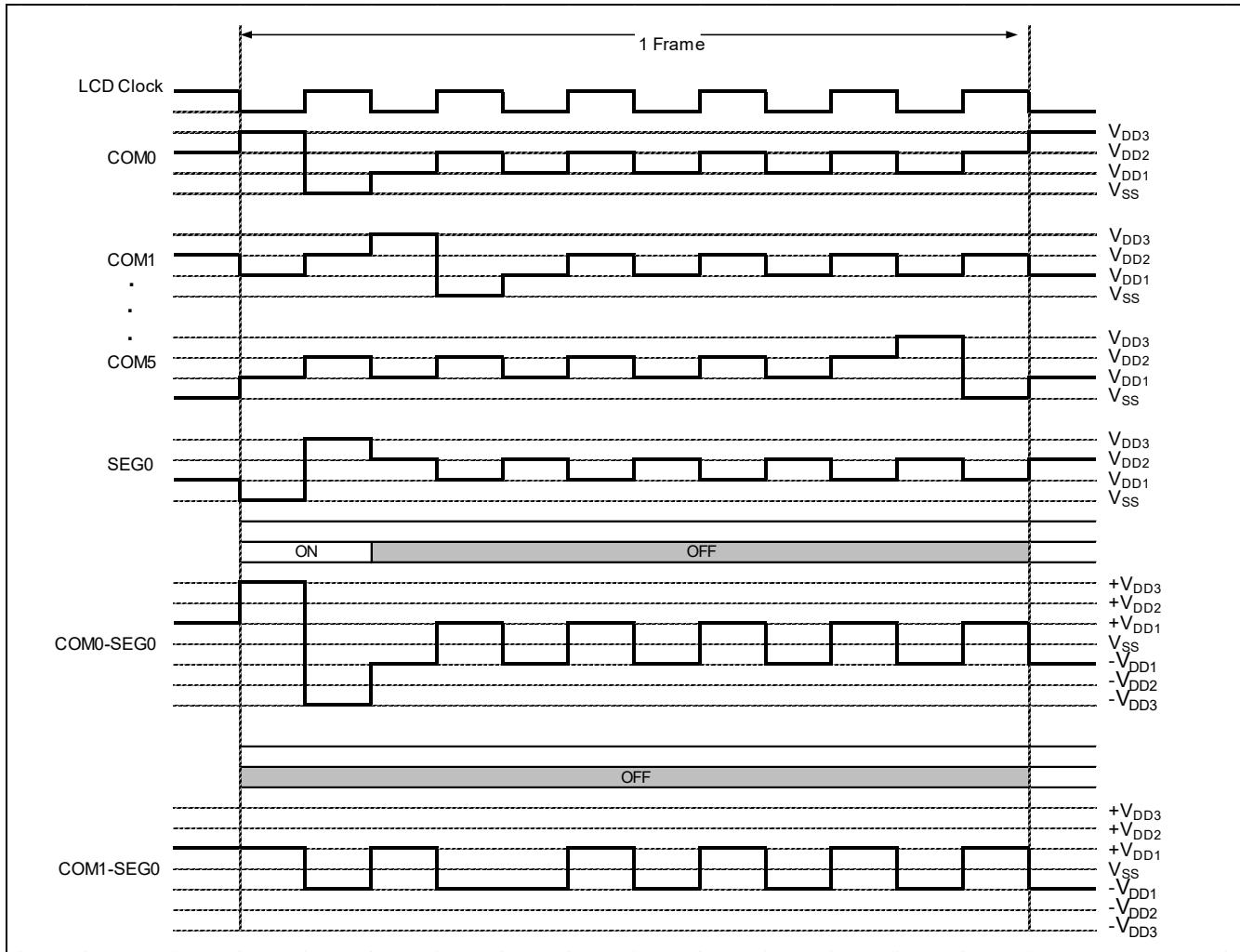
26.4.6 1/4Duty 1/2Bias Drive Waveform



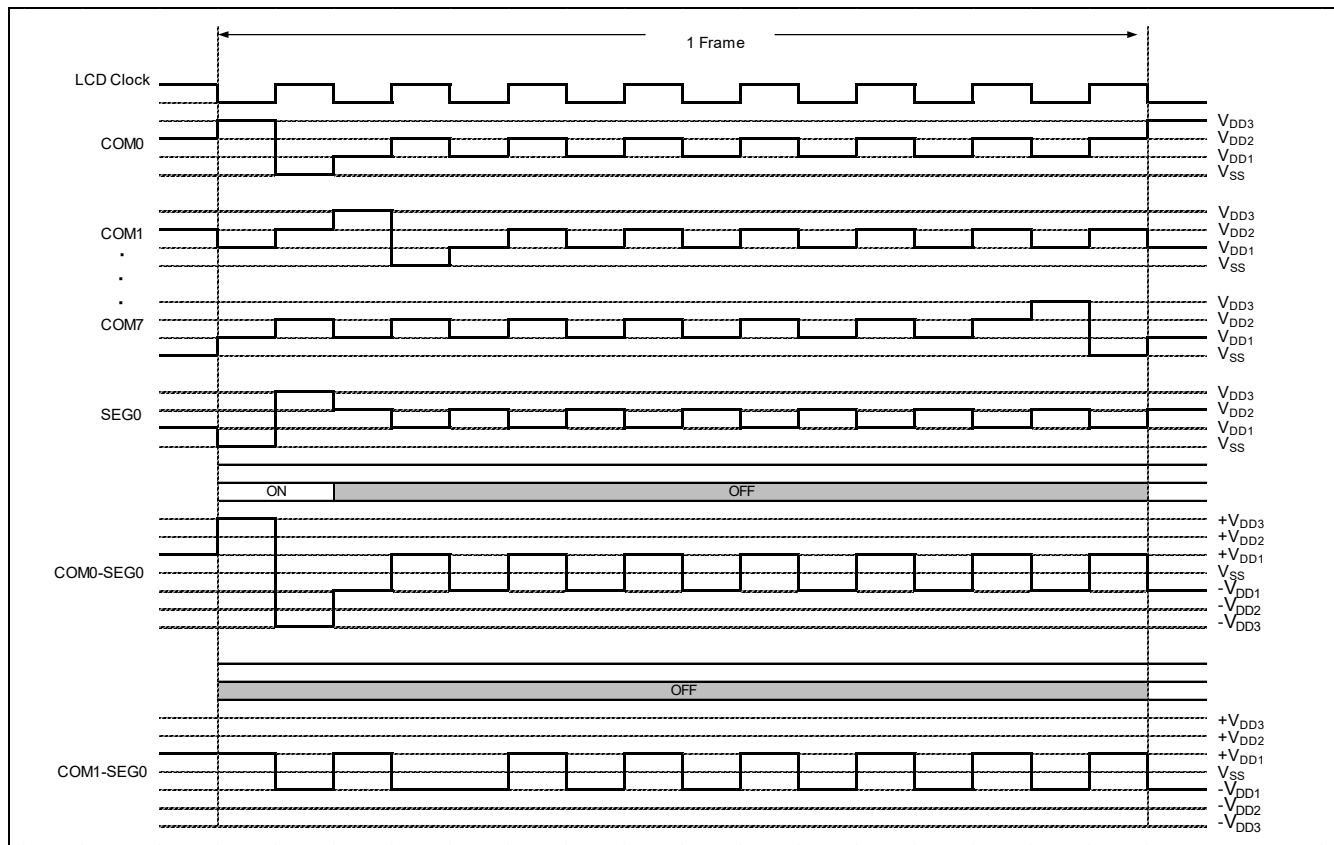
26.4.7 1/4Duty 1/3Bias Drive Waveform



26.4.8 1/6Duty 1/3Bias Drive Waveform



26.4.9 1/8Duty 1/3Bias Drive Waveform



26.5 LCD Bias Generation Circuit

LCD 的 Bias 电压具有 3 种来源：内部电阻分压、外部电阻分压、外部电容分压。

- 1) 当选择内部电阻分压时，芯片会自动切换到内部的电路，这种模式驱动能力较弱。当选择 $1/2bias$ 时，大功耗/中功耗/小功耗电阻分别为 $240K/360K/720K$ ；当选择 $1/3bias$ 时，大功耗/中功耗/小功耗电阻分别为 $360K/540K/1080K$ 。
- 2) 当选择外部电阻分压时，根据系统功耗和 LCD 驱动能力来调整阻值，推荐使用 $500K$ 电阻。
当选择外部电容分压时，驱动能力较强，推荐使用 $100nF$ 电容。但这两种电路需要用户在芯片的外围引脚搭建相关电路，并占用 4 个 IO 口。

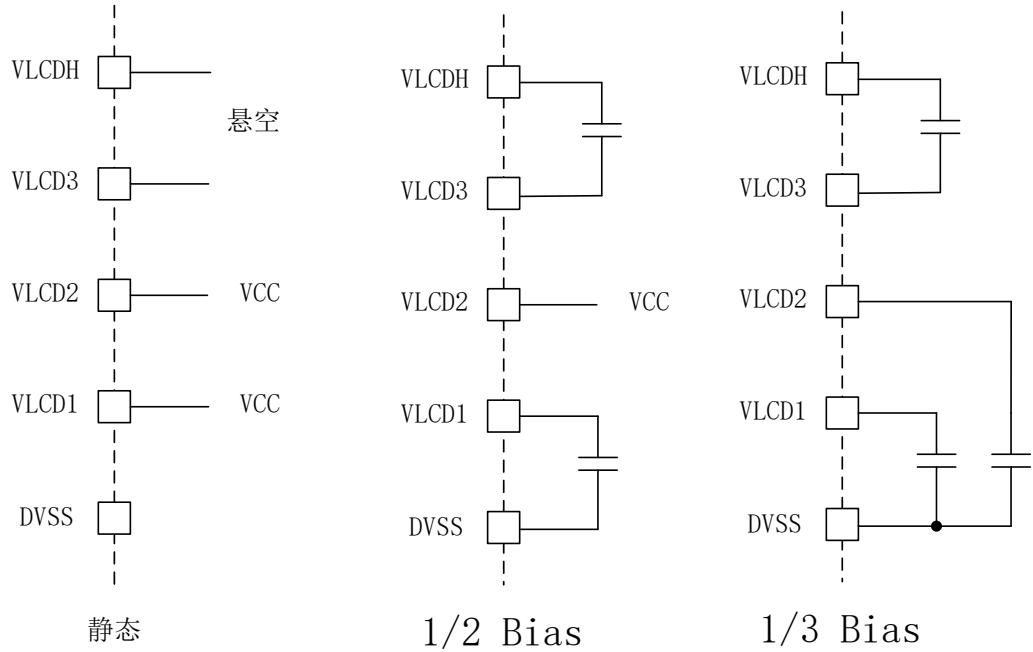
26.5.1 Internal Resistor Mode

内部电阻模式 VLCDH,VLCD1~VLCD3 可以作为 LCD SEG 输出或者 IO 端口使用。

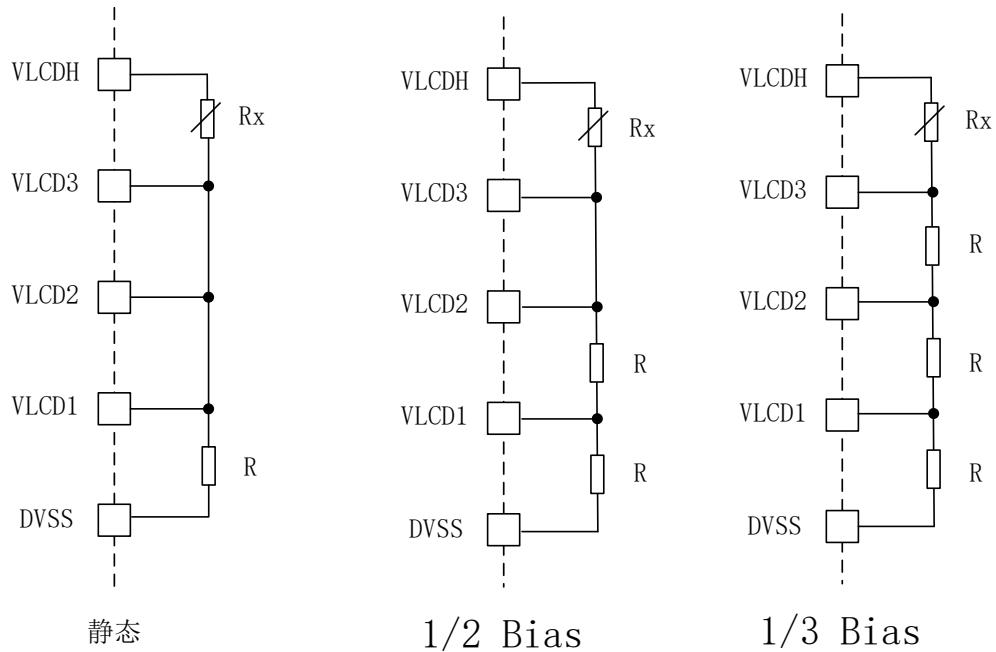
内部电阻模式，LCD 的驱动电压由 CR0.Contrast 控制，如下表所示：

CR0.Contrast	VLCD(1/3 bias)	VLCD(1/2 bias)
0	$1.00 * VCC$	$1.00 * VCC$
1	$0.94 * VCC$	$0.92 * VCC$
2	$0.9 * VCC$	$0.85 * VCC$
3	$0.85 * VCC$	$0.8 * VCC$
4	$0.81 * VCC$	$0.75 * VCC$
5	$0.78 * VCC$	$0.70 * VCC$
6	$0.75 * VCC$	$0.66 * VCC$
7	$0.72 * VCC$	$0.63 * VCC$
8	$0.70 * VCC$	$0.61 * VCC$
9	$0.67 * VCC$	$0.58 * VCC$
10	$0.65 * VCC$	$0.55 * VCC$
11	$0.63 * VCC$	$0.53 * VCC$
12	$0.61 * VCC$	$0.51 * VCC$
13	$0.59 * VCC$	$0.48 * VCC$
14	$0.57 * VCC$	$0.47 * VCC$
15	$0.55 * VCC$	$0.45 * VCC$

26.5.2 External Capacitor Mode



26.5.3 External Resistor Mode



注意：

- R_x 为可调电阻，用于调节 LCD 显示对比度
- 根据使用 LCD 屏幕选择合适的电阻 R

26.6 DMA

LCD 支持软件和硬件触发 DMA 数据传输，可以将需要显示的内容从 RAM 或 ROM 中自动搬到 LCD 显示 RAM 中。硬件触发使用的是帧中断信号。LCD 使用的 DMA 通道号为[6]。

DMA 数据传送配置流程：

1. 使能 DMA
2. 使能 DMA 通道使能
3. 选择 LCD DMA
4. 设置传输类型、传输长度、传输方式
5. 设置源起始地址，目标起始地址
6. 设置源地址、目标地址的递增方式
7. 根据需要使能 DMA 中断
8. 使能 LCD DMA 触发

26.7 Interrupts

当 LCD 设置有效时，LCD 中断可以配置为帧数产生中断。

26.8 LCD Display Modes

LCD 支持两种显示模式。一种以 COM 为显示单元，同一个 SEG 的所有 COM 段在同一字节中（模式 0）。另外一种为同一个 COM 的不同 SEG 在同一个字节中（模式 1）。根据 LCD 面板选择合适的显示方式可以简化程序操作。

26.8.1 LCD Display Mode 1 (MODE = 1)

1/8 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	SEG24	SEG25	SEG26	SEG27	SEG28	SEG29	SEG30	SEG31	
COM0																															LCDRAM0	
COM1																															LCDRAM1	
COM2																															LCDRAM2	
COM3																															LCDRAM3	
COM4																															LCDRAM4	
COM5																															LCDRAM5	
COM6																															LCDRAM6	
COM7																															LCDRAM7	
	SEG32	SEG33	SEG34	SEG35						SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47	SEG48	SEG49	SEG50	SEG51											
COM0																															LCDRAM8	
COM1																															LCDRAM9	
COM2																															LCDRAMA	
COM3																															LCDRAMB	
COM4																															LCDRAMC	
COM5																															LCDRAMD	
COM6																															LCDRAME	
COM7																															LCDRAMF	

1/6 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	SEG24	SEG25	SEG26	SEG27	SEG28	SEG29	SEG30	SEG31	
COM0																															LCDRAM0	
COM1																															LCDRAM1	
COM2																															LCDRAM2	
COM3																															LCDRAM3	
COM4																															LCDRAM4	
COM5																															LCDRAM5	
	SEG32	SEG33	SEG34	SEG35	SEG36	SEG37	SEG38	SEG39	SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47	SEG48	SEG49	SEG50	SEG51												
COM0																															LCDRAM6	
COM1																															LCDRAM7	
COM2																															LCDRAM8	
COM3																															LCDRAM9	
COM4																															LCDRAMA	
COM5																															LCDRAMB	
COM6																															LCDRAMC	
COM7																															LCDRAMD	
	SEG32	SEG33	SEG34	SEG35	SEG36	SEG37	SEG38	SEG39	SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47	SEG48	SEG49	SEG50	SEG51												

1/4 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	SEG24	SEG25	SEG26	SEG27	SEG28	SEG29	SEG30	SEG31	
COM0																															LCDRAM0	
COM1																															LCDRAM1	
COM2																															LCDRAM2	
COM3																															LCDRAM3	
COM4																															LCDRAM4	
COM5																															LCDRAM5	
COM6																															LCDRAM6	
COM7																															LCDRAM7	
	SEG32	SEG33	SEG34	SEG35	SEG36	SEG37	SEG38	SEG39	SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47	SEG48	SEG49	SEG50	SEG51												
COM0																															LCDRAM8	
COM1																															LCDRAM9	
COM2																															LCDRAMA	
COM3																															LCDRAMB	
COM4																															LCDRAMC	
COM5																															LCDRAMD	
COM6																															LCDRAME	
COM7																															LCDRAMF	

1/3 Duty 1/2 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	SEG24	SEG25	SEG26	SEG27	SEG28	SEG29	SEG30	SEG31	
COM0																															LCDRAM0	
COM1																															LCDRAM1	
COM2																															LCDRAM2	
																															LCDRAM3	
																															LCDRAM4	
																															LCDRAM5	
																															LCDRAM6	
																															LCDRAM7	
	SEG32	SEG33	SEG34	SEG35	SEG36	SEG37	SEG38	SEG39	SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47	SEG48	SEG49	SEG50	SEG51												
COM0																															LCDRAM8	
COM1																															LCDRAM9	
COM2																															LCDRAMA	
																															LCDRAMB	
																															LCDRAMC	
																															LCDRAMD	
																															LCDRAWE	
																															LCDRAMF	

26.8.2 LCD Display Mode 0 (MODE = 0)

1/8 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	
SEG0								SEG1								SEG2								SEG3								LCDRAM0
SEG4								SEG5								SEG6								SEG7								LCDRAM1
SEG8								SEG9								SEG10								SEG11								LCDRAM2
SEG12								SEG13								SEG14								SEG15								LCDRAM3
SEG16								SEG17								SEG18								SEG19								LCDRAM4
SEG20								SEG21								SEG22								SEG23								LCDRAM5
SEG24								SEG25								SEG26								SEG27								LCDRAM6
SEG28								SEG29								SEG30								SEG31								LCDRAM7
SEG32								SEG40								SEG48								SEG50								LCDRAM8
SEG33								SEG41								SEG49								SEG51								LCDRAM9
SEG34								SEG42								SEG50								SEG51								LCDRAMA
SEG35								SEG43								SEG51								SEG53								LCDRAMB
SEG36								SEG44								SEG52								SEG54								LCDRAMC
SEG37								SEG45								SEG53								SEG55								LCDRAMD
SEG38								SEG46								SEG54								SEG56								LCDRAWE
SEG39								SEG47								SEG55								SEG57								LCDRAMF

1/6 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	
SEG0								SEG1								SEG2								SEG3								LCDRAM0
SEG4								SEG5								SEG6								SEG7								LCDRAM1
SEG8								SEG9								SEG10								SEG11								LCDRAM2
SEG12								SEG13								SEG14								SEG15								LCDRAM3
SEG16								SEG17								SEG18								SEG19								LCDRAM4
SEG20								SEG21								SEG22								SEG23								LCDRAM5
SEG24								SEG25								SEG26								SEG27								LCDRAM6
SEG28								SEG29								SEG30								SEG31								LCDRAM7
SEG32								SEG40								SEG48								SEG50								LCDRAM8
SEG33								SEG41								SEG49								SEG51								LCDRAM9
SEG34								SEG42								SEG50								SEG51								LCDRAMA
SEG35								SEG43								SEG51								SEG53								LCDRAMB
SEG36								SEG44								SEG52								SEG54								LCDRAMC
SEG37								SEG45								SEG53								SEG55								LCDRAMD
SEG38								SEG46								SEG54								SEG56								LCDRAWE
SEG39								SEG47								SEG55								SEG57								LCDRAMF

1/4 Duty 1/2 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	
SEG0								SEG1								SEG2								SEG3								LCDRAM0
SEG4								SEG5								SEG6								SEG7								LCDRAM1
SEG8								SEG9								SEG10								SEG11								LCDRAM2
SEG12								SEG13								SEG14								SEG15								LCDRAM3
SEG16								SEG17								SEG18								SEG19								LCDRAM4
SEG20								SEG21								SEG22								SEG23								LCDRAM5
SEG24								SEG25								SEG26								SEG29								LCDRAM6
SEG28								SEG29								SEG30								SEG31								LCDRAM7
SEG32								SEG40								SEG48								SEG50								LCDRAM8
SEG33								SEG41								SEG49								SEG51								LCDRAM9
SEG34								SEG42								SEG50								SEG51								LCDRAMA
SEG35								SEG43								SEG51								SEG53								LCDRAMB
SEG36								SEG44								SEG52								SEG54								LCDRAMC
SEG37								SEG45								SEG53								SEG55								LCDRAMD
SEG38								SEG46								SEG54								SEG56								LCDRAWE
SEG39				</td																												

26.9 LCD Registers

基址地址 0x40005C00

表 26-1 LCD 寄存器

寄存器	偏移地址	描述
LCD_CR0	0x000	LCD 配置寄存器 0
LCD_CR1	0x004	LCD 配置寄存器 1
LCD_INTCLR	0x008	LCD 中断清除寄存器
LCD_POEN0	0x00C	LCD 输出配置寄存器
LCD_POEN1	0x010	LCD 输出配置寄存器
LCD_RAM0	0x040	LCD RAM0
LCD_RAM1	0x044	LCD RAM1
LCD_RAM2	0x048	LCD RAM2
LCD_RAM3	0x04C	LCD RAM3
LCD_RAM4	0x050	LCD RAM4
LCD_RAM5	0x054	LCD RAM5
LCD_RAM6	0x058	LCD RAM6
LCD_RAM7	0x05C	LCD RAM7
LCD_RAM8	0x060	LCD RAM8
LCD_RAM9	0x064	LCD RAM9
LCD_RAMA	0x068	LCD RAM10
LCD_RAMB	0x06C	LCD RAM11
LCD_RAMC	0x070	LCD RAM12
LCD_RAMD	0x074	LCD RAM13
LCD_RAME	0x078	LCD RAM14
LCD_RAMF	0x07C	LCD RAM15

26.9.1 Configuration Register 0 (LCD_CRO)

偏移地址 0x000

复位值 0x000000DA

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Contrast				BSEL		Duty		Bias		CpClk		LcdClk		EN	
RW				RW		RW		RW		RW		RW		RW	

位	标记	功能描述																	
31:16	Reserved	保留																	
15:12	Contrast	LCD 对比度调整 注：仅当 Bias 电压来源选择内部电阻分压时有效。 Contrast 值越大，LCD 波形的幅度越小。 0X0 时，LCD 波形幅度最大，对比度最大； 0XF 时，LCD 波形幅度最小，对比度最小；																	
11:9	BSEL	Bias 电压来源选择 111: Reserved 110: 内部电阻分压，大功耗模式 101: Reserved 100: 内部电阻分压，小功耗模式 011: Reserved 010: 内部电阻分压，中功耗模式 001: 电容分压模式，需要外部电路配合 000: 外部电阻模式，需要外部电路配合																	
8:6	Duty	LCD duty 配置 000: 静态 001: 1/2 duty 010: 1/3 duty 011: 1/4 duty 100: Reserved 101: 1/6 duty 110: Reserved 111: 1/8 duty <table border="1" style="margin-top: 10px;"> <tr> <th>duty</th> <th>PAD</th> <th>FUNCTION</th> </tr> <tr> <td rowspan="4">1/4 duty</td> <td>COM4SEG39PAD</td> <td>SEG39</td> </tr> <tr> <td>COM5SEG38PAD</td> <td>SEG38</td> </tr> <tr> <td>COM6SEG37PAD</td> <td>SEG37</td> </tr> <tr> <td>COM7SEG36PAD</td> <td>SEG36</td> </tr> <tr> <td rowspan="2">1/6 duty</td> <td>COM4SEG39PAD</td> <td>COM4</td> </tr> <tr> <td>COM5SEG38PAD</td> <td>COM5</td> </tr> </table>	duty	PAD	FUNCTION	1/4 duty	COM4SEG39PAD	SEG39	COM5SEG38PAD	SEG38	COM6SEG37PAD	SEG37	COM7SEG36PAD	SEG36	1/6 duty	COM4SEG39PAD	COM4	COM5SEG38PAD	COM5
duty	PAD	FUNCTION																	
1/4 duty	COM4SEG39PAD	SEG39																	
	COM5SEG38PAD	SEG38																	
	COM6SEG37PAD	SEG37																	
	COM7SEG36PAD	SEG36																	
1/6 duty	COM4SEG39PAD	COM4																	
	COM5SEG38PAD	COM5																	

			COM6SEG37PAD	SEG37	
			COM7SEG36PAD	SEG36	
1/8 duty			COM4SEG39PAD	COM4	
			COM5SEG38PAD	COM5	
			COM6SEG37PAD	COM6	
			COM7SEG36PAD	COM7	
			LCD Bias 配置 0: 1/3 bias 1: 1/2 bias		
5	Bias		电压泵时钟频率选择 00: 2k Hz 01: 4k Hz 10: 8k Hz 11: 16k Hz		
4:3	CpClk		LCD 扫描频率选择 00: 64Hz 01: 128Hz 10: 256Hz 11: 512Hz 注: LCD 帧频率 = LCD 扫描频率×Duty		
2:1	LcdClk		LCD 使能控制 1: 使能 0: 禁止		
0	EN				

26.9.2 Configuration Register 1 (LCD_CR1)

偏移地址 0x004

复位值 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				INTF	DmaEn	IE	Mode	ClkSrc	BlinkEn					BlinkCnt	
				RO	RW	RW	RW	RW	RW					RW	

位	标记	功能描述
31:11	Reserved	保留
11	INTF	中断标志 1: 中断 0: 无中断
10	DmaEn	DMA 硬件触发使能 1: 使能 LCD 中断触发 DMA, 0: 禁止 LCD 中断触发 DMA
9	IE	中断使能 1: 使能 0: 禁止
8	Mode	LCD RAM 显示模式选择 0 模式 0 1 模式 1
7	ClkSrc	LCD 时钟源选择 1: XTL 0: RCL
8	BlinkEn	LCD 闪屏配置 1: 使能 0: 禁止
5:0	BlinkCnt	闪屏频率与 LCD 中断间隔设置 注: LCD 闪烁频率 = LCD 帧频率 / (BlinkCnt+1) LCD 中断间隔 = (BlinkCnt+1)*(1/LCD 帧频率)

26.9.3 Interrupt Clear Register (LCD_INTCLR)

偏移地址 0x008

复位值 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				INTF ROW0		Reserved									

位	标记	功能描述
31:11	Reserved	保留
10	INTF	中断标志清除, 写 0 清除, 写 1 无效
9:0	Reserved	保留

26.9.4 Output Configuration Register 0 (LCD_POEN0)

偏移地址 0x00C

复位值 0xFFFFFFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
S31	S30	S29	S28	S27	S26	S25	S24	S23	S22	S21	S20	S19	S18	S17	S16
RW															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0
RW															

位	标记	功能描述
31:0	Sx	Segx 输出控制位 0: SEG 输出使能 1: SEG 输出关闭, 可以使用其他功能, 如 IO, 模拟输入输出

26.9.5 Output Configuration Register 1 (LCD_POEN1)

偏移地址 0x010

复位值 0x01FF_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved						COM MUX	C3	C2	C1	C0	S51	S50	S49	S48	S47
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
S46	S45	S44	MUX	S43	S42	S41	S40	S39 C4	S38 C5	S37 C6	S36 C7	S35	S34	S33	S32
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能描述
3:0	Sx	Segx输出控制位 0: SEG输出使能 1: SEG输出关闭，可以使用其他功能，如 IO,模拟输入输出
7:4	SxCy	Segx/COMy输出控制位 0: SEG/COM输出使能 1: SEG/COM输出关闭，可以使用其他功能，如 IO,模拟输入输出 SEG COM 管脚功能选择由CR0.DUTY决定
11:8	Sx	Segx输出控制位 0: SEG输出使能 1: SEG输出关闭，可以使用其他功能，如 IO,模拟输入输出
12	MUX	SEG32~SEG35端口功能选择，详细参考选择表格
20:13	Sx	SEG40-SEG51端口功能选择
24:21	Cx	COMx输出控制位 0: COM输出使能 1: COM输出关闭，可以使用其他功能，如 IO,模拟输入输出
25	COMMUX	COM2/3管脚不同位置输出。 0: COM2/3 在PA11/PA12管脚输出 1: COM2/3 在PF6/ PA15管脚输出

VLCDXSEGXPAD 状态描述		寄存器如何配置						
		MUX	S<35: 32>	BSEL				
VLCDXSEGXPAD 选择 GPIO, LCD disable(LCD_ON=0)时		VLCDHSEG35=IO VLCD3SEG34=IO VLCD2SEG33=IO VLCD1SEG32=IO	1	1111	X	X	X	
		VLCDHSEG35=IO VLCD3SEG34=IO VLCD2SEG33=IO VLCD1SEG32=IO			1	1	0	
		VLCDHSEG35=IO VLCD3SEG34=IO VLCD2SEG33=IO VLCD1SEG32=IO			0	1	0	
		VLCDHSEG35=IO VLCD3SEG34=IO VLCD2SEG33=IO VLCD1SEG32=IO			1	0	0	
VLCDXSEGXPAD 选择 IO, LCD enable(LCD_ON=1)时, 这时只能选择内部电阻工作模式		VLCDHSEG35=IO VLCD3SEG34=IO VLCD2SEG33=IO VLCD1SEG32=IO	1	1111	1	0	0	
		VLCDHSEG35=IO VLCD3SEG34=IO VLCD2SEG33=IO VLCD1SEG32=IO			0	1	0	
		VLCDHSEG35=IO VLCD3SEG34=IO VLCD2SEG33=IO VLCD1SEG32=IO			1	0	0	
		VLCDHSEG35=IO VLCD3SEG34=IO VLCD2SEG33=IO VLCD1SEG32=IO			0	0	0	
选择外部电阻工作模式, 内部电阻断路		VLCDHSEG35=VOUT VLCD3SEG34=VLCD3 VLCD2SEG33=VLCD2 VLCD1SEG32=VLCD1	0	0000	0	0	0	
		VLCDHSEG35=SEG35/IO VLCD3SEG34=SEG34/IO VLCD2SEG33=SEG33/IO VLCD1SEG32=SEG32/IO			XXXX	1	1	0
		VLCDHSEG35=SEG35/IO VLCD3SEG34=SEG34/IO VLCD2SEG33=SEG33/IO VLCD1SEG32=SEG32/IO			XXXX	0	1	0
		VLCDHSEG35=SEG35/IO VLCD3SEG34=SEG34/IO VLCD2SEG33=SEG33/IO VLCD1SEG32=SEG32/IO			XXXX	1	0	0
选择内部电阻工作模式		VLCDHSEG35=DH1 VLCD3SEG34=DH2 VLCD2SEG33=VLCD2 VLCD1SEG32=VLCD1	0	0000	0	0	1	

26.9.6 LCD_RAM0~7

偏移地址 0x040~0x5c

复位值：无

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
RW															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
RW															

位	标记	功能描述
31:0	Dx	LCD 点输出，显示参考 LCD 显示模式 0 对应的 SEG COM 交叉点不亮；1 对应的 SEG COM 交叉点亮；

26.9.7 LCD_RAM8~F

偏移地址 0x060~0x7C

复位值：无

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								D23	D22	D21	D20	D19	D18	D17	D16
								RW							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能描述
31:24	Reserved	保留
23:0	Dx	LCD 点输出，显示参考 LCD 显示模式 0 对应的 SEG COM 交叉点不亮；1 对应的 SEG COM 交叉点亮；

27 Crystal-less USB Clock Tuner (CTS)

时钟校准器可以调整校准 RCH48M 时钟频率，也可以调整校准其他 RC 振荡的时钟频率，还可以作为一个通用定时器来使用。本模块主要功能为 RCH48M 自动校准，其他功能为不使用 48M 自动校准时的附加功能。

调整校准 RCH48M 通过可选基准参考信号进行测量待校准振荡器的频率，可以根据测得频率误差自动调整振荡器的频率，也可以使用手动调整。

RCH48M 为 USB 外设提供精准时钟，在这种情况下，基准参考信号可由 USB 总线上的帧起始(SOF)数据包信号提供，USB 主机以 1ms 的时间间隔发送该信号。USB 模块会产生一个 2ms 为周期的翻转信号，CTS 系统时钟基准参考信号也可以由 XTL 振荡器输出或者外部管脚提供，或者由用户软件产生。

RCH24M 与 RCL 振荡校准不支持自动校准，只支持手动调整。参考时钟源可以灵活选择。

定时器是重载模式计数定时器，可以产生捕获中断。支持低功耗唤醒。

27.1 48M Clock Calibration

27.1.1 System Features

具有可编程预分频和极性的可选基准参考源

- USB SOF 数据包接收
- XTL 振荡器输出
- 外部引脚
- LPTIMER 翻转

可由软件生成基准参考脉冲

振荡器自动微调功能，无需 CPU 参与

手动软件控制选项，可加快校准启动

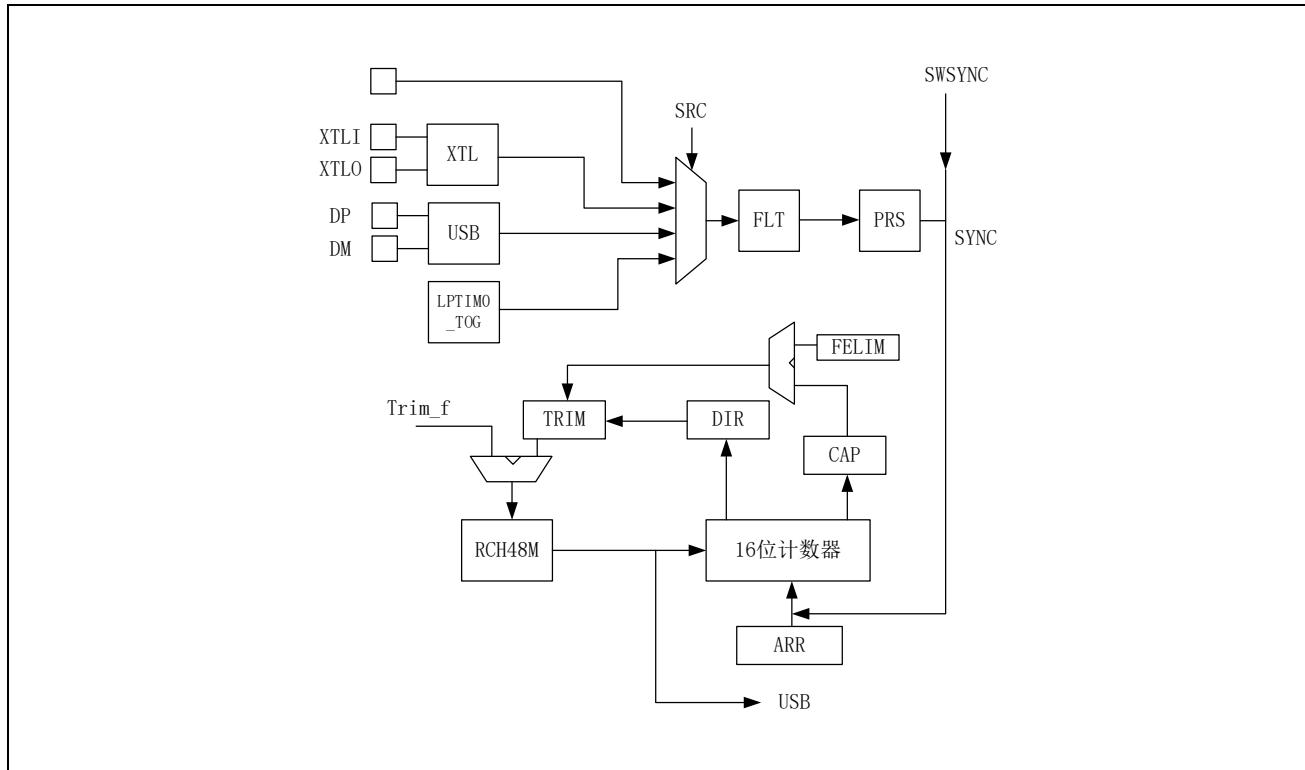
16 位频率误差计数器，用于自动误差捕获和重载

可编程误差限值，用于自动频率误差值和状态标志

可屏蔽中断

- 计数器计数下溢出 (UDF)
- 时钟校准正常 (OK)
- 时钟校准报警 (WARN)
- 时钟校准错误 (ERR)

27.1.2 Block Diagram



27.1.3 Reference Source Synchronous Input

时钟校准系统基准参考源可通过寄存器 CTS_CFGR 选择，它可以是来自外部的引脚信号、XTL 时钟、USB_SOF 信号或 LPTIMER 的溢出信号。为使基准参考信号输入稳定，可配置是否使用一个简单的数字滤波器来滤除任何干扰。该信号源还可以配置极性，可由预分频器分频，以获取合理频率范围内的同步信号（通常为 1mS 左右）。

还可以通过软件将 CTS_CR 寄存器中的 SW_SYNC 位置 1 的方法生成同步信号。使用 SW_SYNC 时，将参考时钟选择为引脚信号，并且要保证引脚输入电平为低电平。

27.1.4 Frequency Error Measurement

频率误差计数器是一个 16 位递减/递增计数器，会在每个基准参考源的 SYNC 事件发生时重载 ARR 值。它开始递减计数，直至达到零值，此时会生成 UDF（预期下溢出）事件。随后它将递增计数到 OUTRANGE 限值，这种情况下最终将停止计数（如果未接收到 SYNC 事件）并生成 MISS 事件。OUTRANGE 限值定义为频率误差限值（CTS_CFGR 寄存器的 FELIM 字段）乘以 128。

当检测到 SYNC 事件时，频率误差计数器的实际值及其方向存储在 CTS_ISR 寄存器的 FECAP（频率误差捕捉）字段和 FEDIR（频率误差方向）位中。当在递减计数阶段（达到零值之前）期间检测到 SYNC 事件时，意味着实际频率小于目标频率（因此，TRIM 值应递增）；当在递增计数期间检测到 SYNC 事件时，意味着实际频率大于目标频率（因此，TRIM 值应递减）。

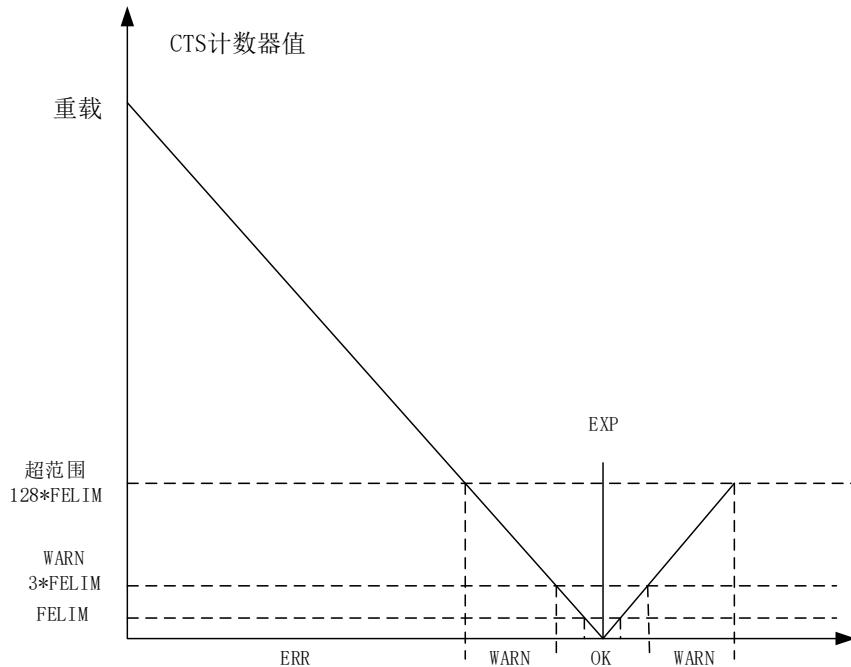


图 27-1 CTS 计数器

27.1.5 Frequency Error Fine-tuning

测得的频率误差通过将频率值与一组限值进行比较进行估算：

- *TOLERANCE LIMIT* (容差限值)，在 *CTS_CFGR* 寄存器的 *FELIM* 字段中直接给出
- *WARNING LIMIT* (警告限值)，定义为 $3 * FELIM$ 值
- *OUTRANGE* (误差限值)，定义为 $128 * FELIM$ 值

该比较结果用于生成状态指示以及控制自动微调，自动微调通过将 *CTS_CR* 寄存器中的 *AUTO_TRIMEN* 位置 1 来使能。

- 当频率误差低于容差限值时，意味着 TRIM 字段中的实际微调值是最优值，因此无需任何微调操作。
 - 指示 OK 状态
 - AUTOTRIM 模式下的 TRIM 值无变化
- 当频率误差低于警告限值但高于或等于容差限值时，意味着需要某种微调操作，但只需一步微调便可达到最优 TRIM 值。
 - 指示 OK 状态
 - 在 AUTOTRIM 模式下通过 ± 1 微调来调整 TRIM 值

- 当频率误差高于或等于警告限值但低于误差限值时，意味着需要更强的微调操作，并且存在下一周期无法达到最优 TRIM 值的风险。
 - 指示 *WARN* 状态
 - 在 *AUTOTRIM* 模式下通过±2 微调来调整 *TRIM* 值
- 当频率误差高于或等于误差限值时，意味着频率超出微调范围。当 *SYNC* 输入不干净或某个 *SYNC* 脉冲丢失时（例如，当一个 *USB SOF* 被破坏时）。也会发生这种情况。
 - 指示 *ERR* 或 *MISS* 状态
 - *AUTOTRIM* 模式下的 *TRIM* 值无变化

注意：

- 如果 *TRIM* 字段的实际值十分接近其限值，如果需要向其限值方向调整，只会调整到其限值，之后 *TRIM* 值保持为限值，此时将指示 *OVF* 状态。
- 在 *AUTOTRIM* 模式 (*CTS_CR* 寄存器中的 *AUTO_TRIMEN* 位置 1) 下，*CTS_CR* 的 *TRIM* 字段由硬件调整并且是只读的。

27.1.6 CTS Initialization and Configuration

ARR 值

ARR 值应根据目标频率与预分频后参考同步源频率之比进行选择。该值随后会减 1，以在零值时达到预期同步。具体公式如下：

$$\text{ARR} = (\text{fTARGET} / \text{fSYNC}) - 1$$

ARR 字段的复位值对应于 48 MHz 的目标频率和 1 kHz 的同步信号频率(来自 USB 的 SOF 信号)。

FELIM 值

FELIM 值的选择与 RCH48M 振荡器的特性及其典型微调步长紧密相关。最优值对应于微调步长的一半，以 RCH48M 振荡器时钟节拍数表示。可使用以下公式：

$$\text{FELIM} = (\text{fTARGET} / \text{fSYNC}) * \text{STEP}[\%] / 100\% / 2$$

结果应始终舍入为最接近的整数值以获得最佳微调响应。如果应用中不需要频繁的微调操作，可通过稍微增大 FELIM 值来增加微调滞后。

FELIM 字段的复位值对应于 $(\text{fTARGET} / \text{fSYNC}) = 48000$ 以及 0.14% 的典型微调步长。

注意：

- 错误配置 ARR 和 FELIM 字段无法实现硬件保护，从而导致不定微调响应。预期工作模式需要正确设置 ARR 值（根据同步源频率），该值还大于 $128 * \text{FELIM}$ 值 (*OUTRANGE* 限值)。

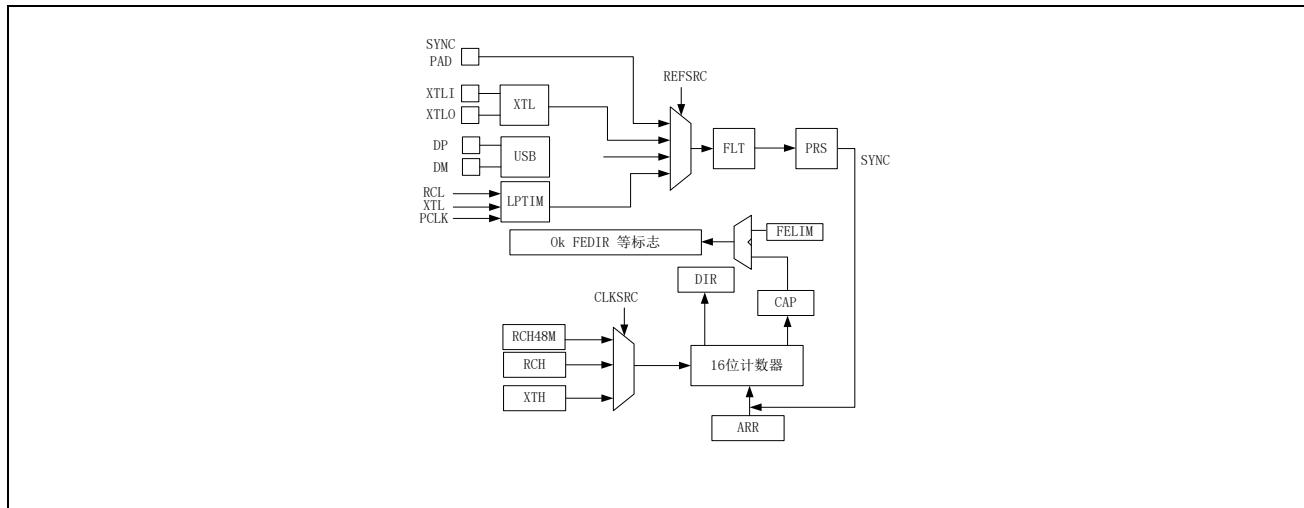
27.2 Other Oscillators Calibrations

27.2.1 System Features

校准高速时钟 RCH 参考时钟可以选择端口输入，或者内部其他精准时钟源的通过 LPTIMER 的输出。

校准低速时钟 RCL 时，将 RCL 作为参考时钟，计数时钟选择为精准的内部时钟或外部输入精准时钟。

27.2.2 Block Diagram



27.2.3 Calibration Instructions

RCH 振荡校准：

1. 设置参考时钟为端口输入或者精准的 XTL，设置合适的参考时钟分频
2. 设置校准时钟为 RCH
3. 设置 CFGR 设置要校准的目标频率与误差范围
4. 在中断服务程序中判断 FEDIR，如果为 1，将频率值提高，增大 TRIM 值；反之减小

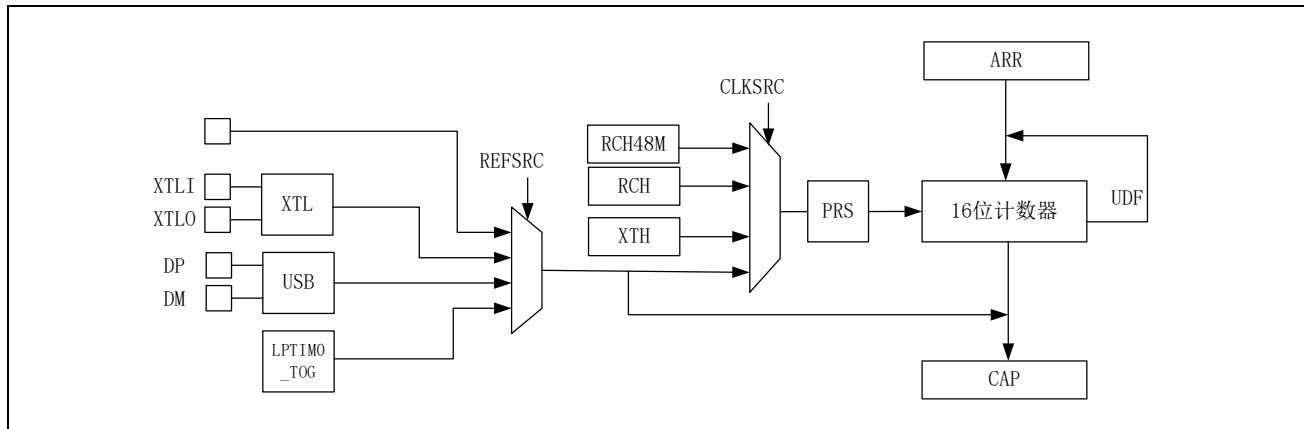
TRIM 值初始设置为中值，下次更改的值的上次更改值的 1/2，即 2 分法。9 位的 TRIM 值 8 次即可校准。

RCL 振荡校准：

1. 设置参考时钟 RCL 作为 LPTIMER 定时器器的时钟产生参数的 LPTIMO 翻转信号
 2. 设置校准时钟为 XTH 或校准后的 RCH 时钟
 3. 设置 CFGR 设置要校准的目标频率与误差范围
 4. 在中断服务程序中判断 FEDIR，如果为 0，将频率值提高，增大 TRIM 值；反之减小
- TRIM 值初始设置为中值，下次更改的值的上次更改值的 1/2，即 2 分法。9 位的 TRIM 值 8 次即可校准。

27.3 Timer Function Description

27.3.1 Block Diagram



- 16 位下计数自动重载低功耗定时器
- 多种时钟源可配
- 1/2/4/8/16/32/64/256 预分频可配置
- 输入捕获功能
- 定时器唤醒功能
- 支持溢出中断，捕获中断

27.3.2 Timer Clock Selection

定时器可以选择 RCH48M,RCH,XTH 等内部高速时钟，也可以选择外部端口输入，低速 XTL 和 LPTIMER 定时器翻转信号做为计数时钟。当选择高速时钟时，外部输入可以实现上沿或下沿的边沿捕获功能。

27.3.3 Timers

定时器计数值从 ARR 减计数，当计数到 0 后产生下溢出，如果中断使能会产生下溢出中断，计数周期为 $ARR+1$ 。

TIM_EN 只有在 CEN 为 0 时才可以更改，TIM_EN 为 1 并且 CEN 写 1 时定时器会重新装载重载值到定时器。

27.4 CTS Registers

基地址 0x40005000

27-1 CTS

CTS_CR	0x000	控制寄存器
CTS_CFGR	0x004	配置寄存器
CTS_ISR	0x008	中断和状态寄存器
CTS_ICR	0x00C	中断标志清除寄存器

27.4.1 CTS Control Register (CTS_CR)

偏移地址 0x00

复位值 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIM_EN					TRIM			SW_SYNC	AUTO_T_RIM_EN	CEN	CAPI_E	UDFIE	ERRIE	WARNIE	OKIE
RW					RW			RW	RW	RW	RW	RW	RW	RW	RW

&#a	RkCEO	,w*\$K)š
31:16	Reserved	保留
15	TIM_EN	定时器功能（只有在 CEN 为 0 时，才可以改写）
14:8	TRIM	RCH48M 振荡微调值，当 AUTO_TRIMEN 为 1 时，由硬件控制且只读；为有符号数，0x40 最小，0x3F 最大，0x00 是中值；自动校准时值有上下限保护。 手动 TRIM 时，这个值可以读写。
7	SW_SYNC	软件同步，软件写 1，硬件清零。 注：如果 trim 的目标时钟频率小于系统时钟，不可以使用软件同步
6	AUTO_TRIM_EN	自动微调使能 1：使能；0：禁止 注：使能时会将写入的 TRIM 值同步到硬件自动 TRIM 值中，如果需要保持自动 TRIM 值为中值，必须在手动 TRIM 模式下将 TRIM 值清零。
5	CEN	频率误差计数器/定时器使能
4	CAPIE	定时器模式捕获中断使能
3	UDFIE	计数器下溢出中断使能
2	ERRIE	校准错误中断使能
1	WARNIE	校准警告中断使能
0	OKIE	校准正常中断使能

27.4.2 CTS Configuration Register (CTS_CFGR)

偏移地址 0x04

复位值 0x0022BB7F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLKSRC	REFSRC	POL		DIV							FELIM				
RW	RW	RW		RW							RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ARR							
								RW							

位	标记	功能描述
31:30	CLKSRC	校准模式下校准时钟选择 定时器模式下计数时钟选择 保留 默认 RCH48M 00: RCH48M 01: RCH 10: XTH 11: RFFCLK (仅限定时器模式下使用)
29:28	REFSRC	校准同步信号源选择, 定时器模式时钟选择或捕获源选择 00: USB SOF 01: XTL 10: GPIO 11: LPTIM TOG 翻转输出
27	POL	同步极性选择 0: 上升沿 1: 下降沿 注: 使用 USB_SOF 时无效, USB_SOF 为周期 2ms 的翻转信号, 需要将其转换为周期 1ms 的信号
26:24	PRS	TRIM 同步信号分频器 /定时器时钟预分频 000: 不分频 001: 2 分频 010: 4 分频 011: 8 分频 100: 16 分频 101: 32 分频 110: 64 分频 111: 256 分频
23:16	FELIM	频率误差限值
15:0	ARR	计数器重载值

注: 这个寄存器只有在 CEN 为 0 时才可以更改。

27.4.3 CTS Interrupt and Status Register (CTS_ISR)

偏移地址 0x08

复位值 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FECAP															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR	Reserved			OVF	MISS	ERR	Reserved				UDF_F	ERR_F	WARNF	OKF	
RO				RO	RO	RO					RO	RO	RO	RO	

位	标记	功能描述
31:16	FECAP	频率误差捕获值
15	DIR	频率误差方向，显示实际频率低于目标频率还是高于目标频率 0：递增计数方向，实际频率高于目标频率 1：递减计数方向，实际频率低于目标频率
14:11	Reserved	保留
10	OVF	TRIM 值上溢或下溢时，该标志由硬件置 1
9	MISS	校准参考同步丢失(频率过快并超范围)，计数到 0 后计数值递增计数到 FELIM*128 没有检测到同步信号。
8	ERR	校准错误(频率过慢并超范围)，参考同步信号在计数器溢出之前且误差大于 FELIM*128，
7:5	Reserved	保留
4	CAPF	捕获标志
3	UDFF	计数器下溢出标志，频率误差计数器计数到 0 时，该位硬件置 1。
2	ERRF	校准错误标志，OVF,MISS,ERR 逻辑或的结果
1	WARNF	校准警告标志，频率误差大于等于 FELIM*3 且小于 FELIM*128
0	OKF	校准正常标志。测量频率误差小于 FELIM*3 时，该标志硬件置 1。

27.4.4 CTS Interrupt Status Clear Register (CTS_ICR)

偏移地址 0x0C

复位值 0x0000001F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										CAP C	UDF C	ERR C	WAR NC	OKC	
R1W 0										R1W 0	R1W 0	R1W 0	R1W 0	R1W 0	

位	标记	功能描述
31:5	Reserved	保留
4	CAPC	定时器模式捕获标志清除，写 0 清除，写 1 无效。
3	UDFC	预期同步标志清除。写 0 清除
2	ERRC	校准错误标志清除，同时清除 OVF,MISS,ERR 标志。写 0 清除
1	WARNC	校准警告标志清除。写 0 清除
0	OKC	校准正常标志清除。写 0 清除

28 Audio interface (I2S)

I2S 特性

- 支持 Philip / MSB / LSB / PCM 模式
- 支持 MCK 输出
- 支持 48K, 44.1K, 32 K, 16K, 8K 等不同的音频采样率
- 支持 数据长度 16 位,24 位,32 位
- 支持 帧长度 16 位/32 位
- 支持 DMA 数据传输
- 支持全双工收发 (2 个 I2S 配合)
- 支持 master 发送、接收
- 支持 slave 发送、接收
- 与 SPI 共用中断与端口

28.1 Functional Description

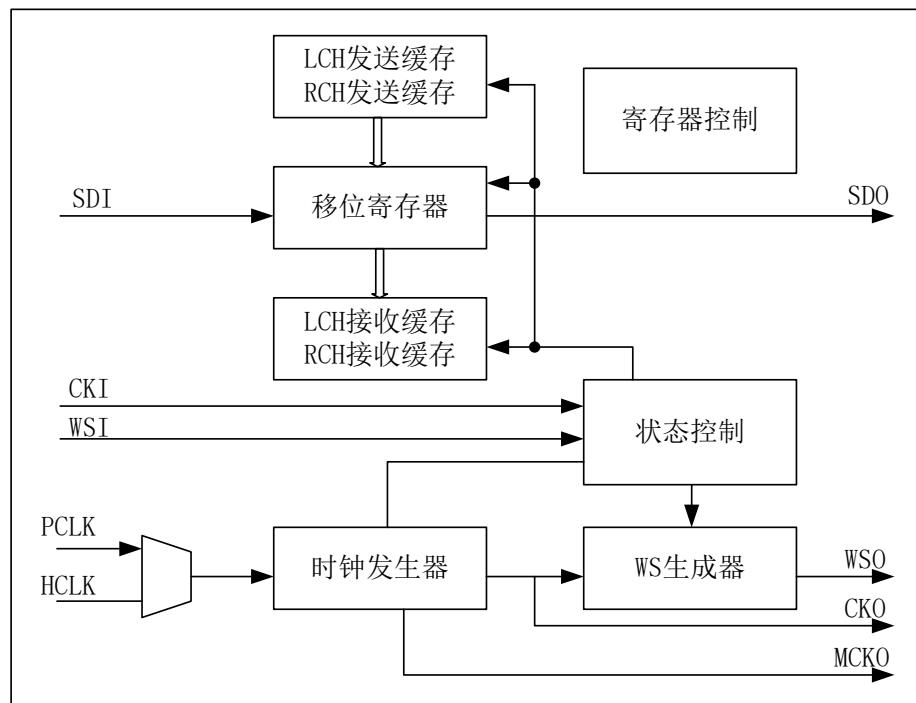


图 28-1 功能框图

28.1.1 Pin Multiplexing

I2S 与 SPI IO 复用，复用关系如下：

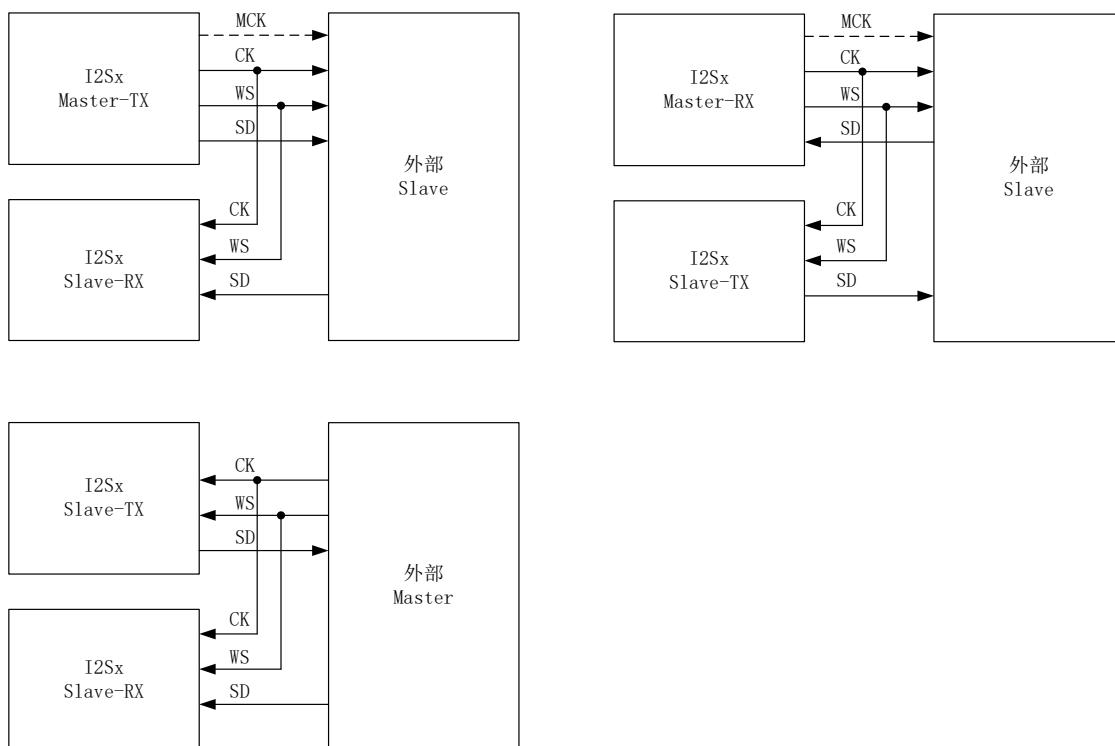
当使能 I2S 功能（将 SYSCTRL_PERI 寄存器中的 I2S 位置 1 并且 SPI 无效）后，SPI 与 I2S 复用端口用作音频 I2S 接口。

I2S	SPI	I2S 管脚说明
SD	MOSI	串行数据（映射到 MOSI 引脚），用于发送或接收两个时分复用的数据通道上的数据（仅半双工模式）
WS	NSS	字选择（映射到 NSS 引脚），是主模式下的数据控制信号输出以及从模式下的数据控制信号输入。
CK	SCK	串行时钟（映射到 SCK 引脚），是主模式下的串行时钟输出以及从模式下的串行时钟输入。
MCK	MISO	主时钟输出（MCK 映射到 MISO 引脚），当 I2S 配置为主模式（并且 I2Sx_PR 寄存器中的 MCKOE 位置 1）时，使用主时钟（单独映射）输出此附加时钟，该时钟以 $256 \times fs$ 的预配置频率生成，其中 fs 为音频信号采样频率。

I2S 另外有一组独立的 IO 设置，客户可以同时使用 I2S 与 SPI。在这种模式下中断与 SPI 共用同一个中断入口地址，需要根据各自标志判断处理相应的中断服务程序。

28.1.2 I2S Full-Duplex

每个 I2S 都支持主模式与从模式，支持半双工的发送与接收，如果需要全双工模式，需要使用两个 I2S 进行配合。



28.2 I2S Audio Protocol Description

I2S 需要处理通常在左右两个通道上时分复用的音频数据。但是，有两个 16 位寄存器进行发送或接收。所以，需由软件将与相应通道对应的值写入通道数据寄存器，以及从相应通道数据寄存器中读取相应数据。始终先发送左通道数据，而后再发送右通道数据（对于 PCM 协议来说，右通道数据没有意义）。

数据和帧格式组合有四种，可采用下列格式发送数据：

- 将 16 位数据封装在 16 位帧中
- 将 16 位数据封装在 32 位帧中
- 将 24 位数据封装在 32 位帧中
- 将 32 位数据封装在 32 位帧中

当使用 32 位数据包中的 16 位数据时，只需要读写 16 位有效数据，16 位无效数据被强制清零，无需任何软件操作或 DMA 请求（只需一个读/写操作）。

24 位和 32 位数据帧需要对 I2Sx_DR 寄存器执行两次 CPU 读取或写入操作，或者需要两次 DMA 操作。对于 24 位数据帧，硬件会在自动填充 8 个 0 扩展到 32 位。

对于所有数据格式和通信标准而言，始终会先发送最高有效位（MSB 优先）。

I2S 接口支持四种音频标准，可使用 I2Sx_CFGR 寄存器中的 I2SSTD[1:0] 和 PCMSYNC 位对其进行配置。

28.2.1 I2S Philips Standard

使用 WS 信号来指示当前正在发送的数据所属的通道。该信号从当前通道数据的第一个位(MSB) 之前的一个时钟开始有效。

发送方在时钟信号 (CK) 的下降沿改变数据，接收方在上升沿读取数据。WS 信号也在 CK 的下降沿变化。

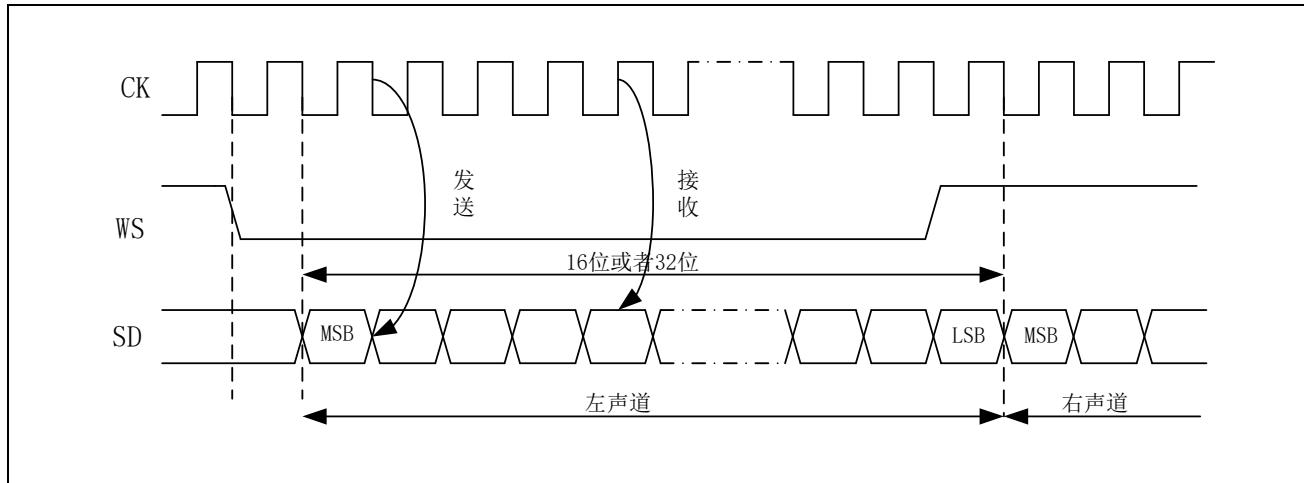


图 28-2 Philip 协议波形 (16/32 位全精度)

该模式需要对 I2Sx_DRL/ I2Sx_DRR 寄存器执行两次写入或读取操作。

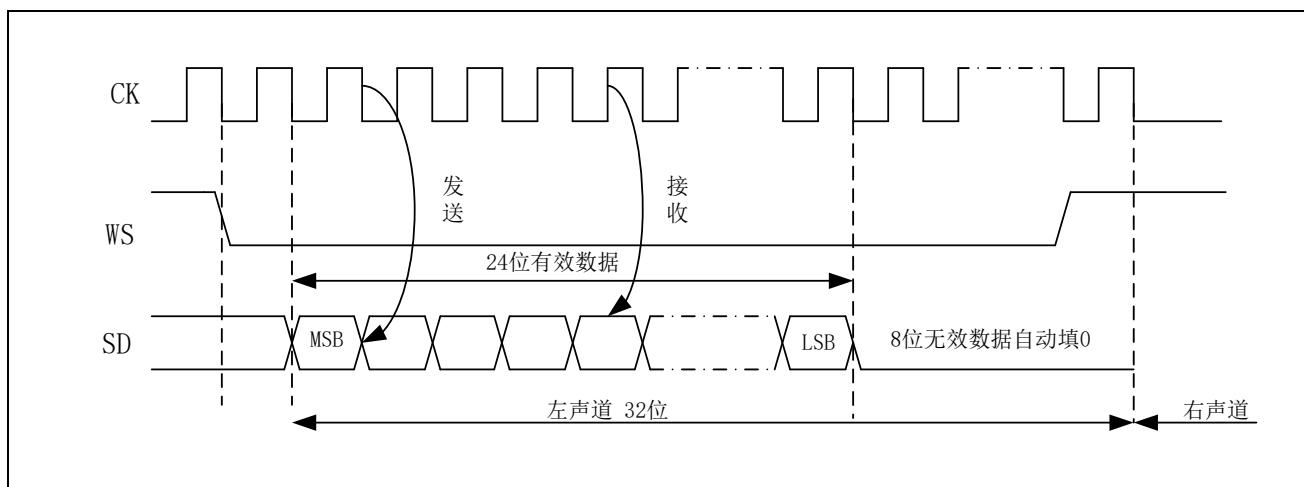


图 28-3 Philip 协议波形 (24 位帧)

该模式需要对 I2Sx_DRL/ I2Sx_DRR 寄存器执行两次写入或读取操作。

在发送模式下，如果需要发送 0x89ABCD,0x8FEDCA (24 位)：

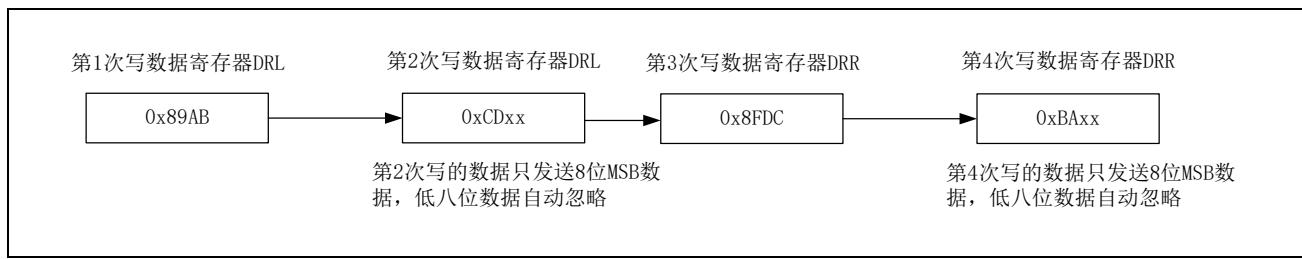


图 28-4 24 位数据发送写操作

在接收模式下，如果接收数据 0x123456,0x89ABCD：

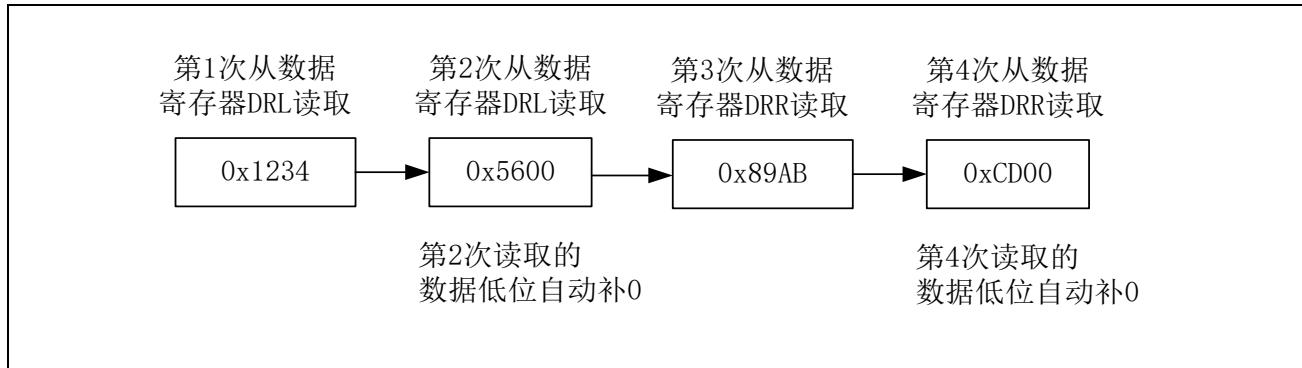


图 28-5 24 位数据模式接收读操作

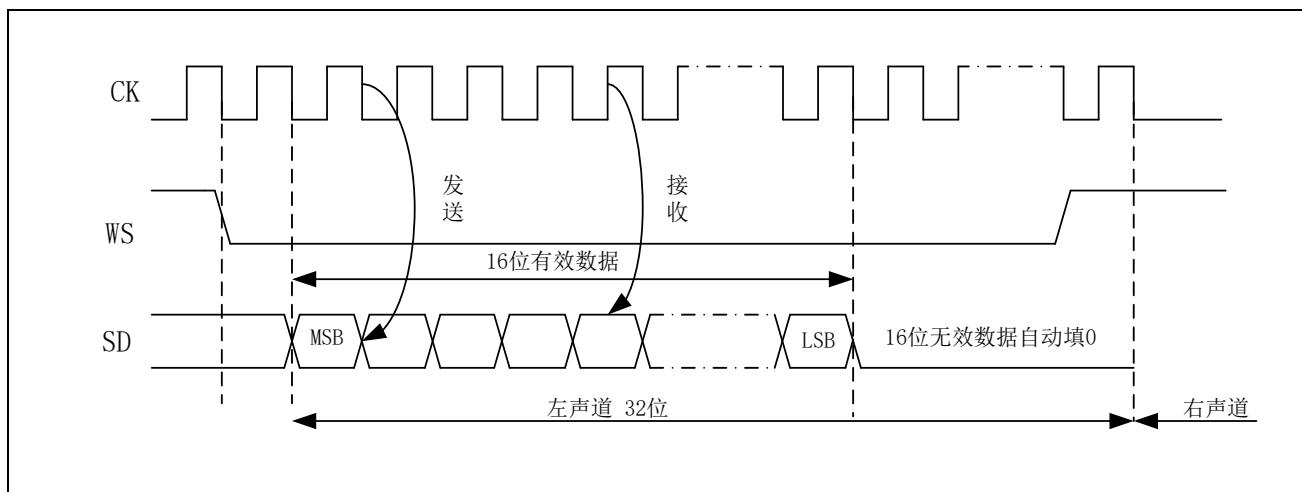


图 28-6 Philip 协议 16 位扩展到 32 位数据帧

如果在 I2S 配置阶段选择将 16 位数据帧扩展到 32 位通道帧，则只需要访问一次 I2Sx_DR 寄存器。另外 16 位将有硬件自动填 0，不需要软件参与。

如果要发送的数据或已接收的数据为 0xCDEF (0xCDEF0000 扩展为 32 位)，则需要执行下图中显示的操作。

只需要一次数据寄存器 DRL/DRR 读写

0xCDEF

图 28-7 16 位数据读写操作

发送时，每次将 MSB 写入数据寄存器 I2Sx_DRL/R，数据转移到移位寄存器后，相应通道的 TXE 标志就会置 1，并在中断使能的情况下触发中断，以将要发送的新数据加载到 I2Sx_DR 寄存器。即使硬件填充的低 16 位 0x0000 还未发送，也会如此，因为低 16 位是由硬件发送。

接收时，接收到第一个半字（高 16 位），则硬件将相应通道的 RXNE 标志置 1，并在中断使能的情况下触发中断。

28.2.2 MSB Alignment Standard

此标准同时生成 WS 信号和第一个数据位，WS 左声道对应高电平，右声道对应高电平。

发送方在时钟信号 (CK) 的下降沿改变数据，接收方在上升沿读取数据。WS 信号也在 CK 的下降沿变化。

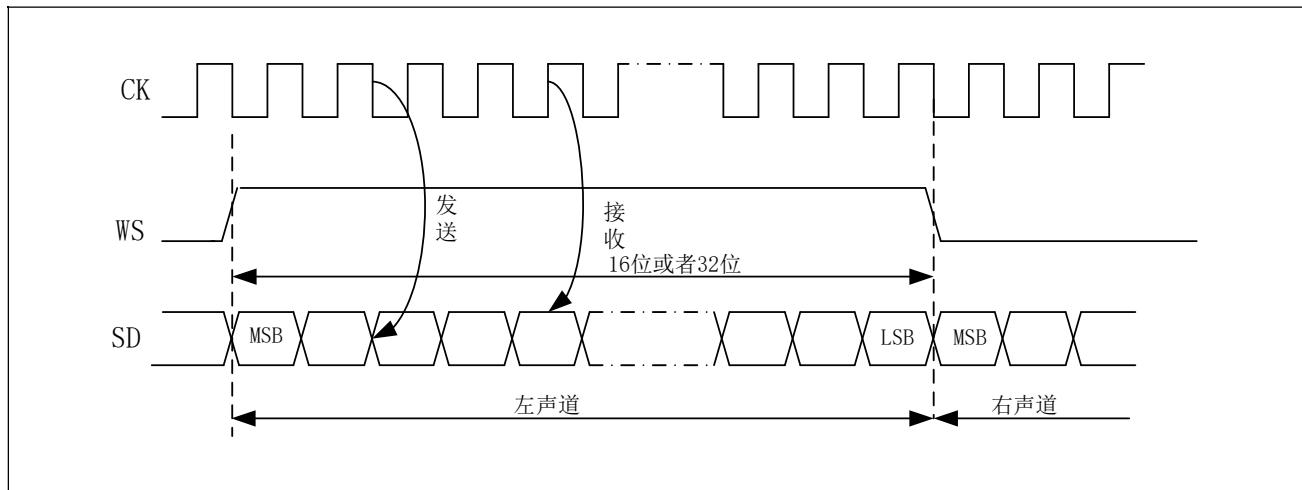


图 28-8 MSB 协议波形 (16/32 位全精度)

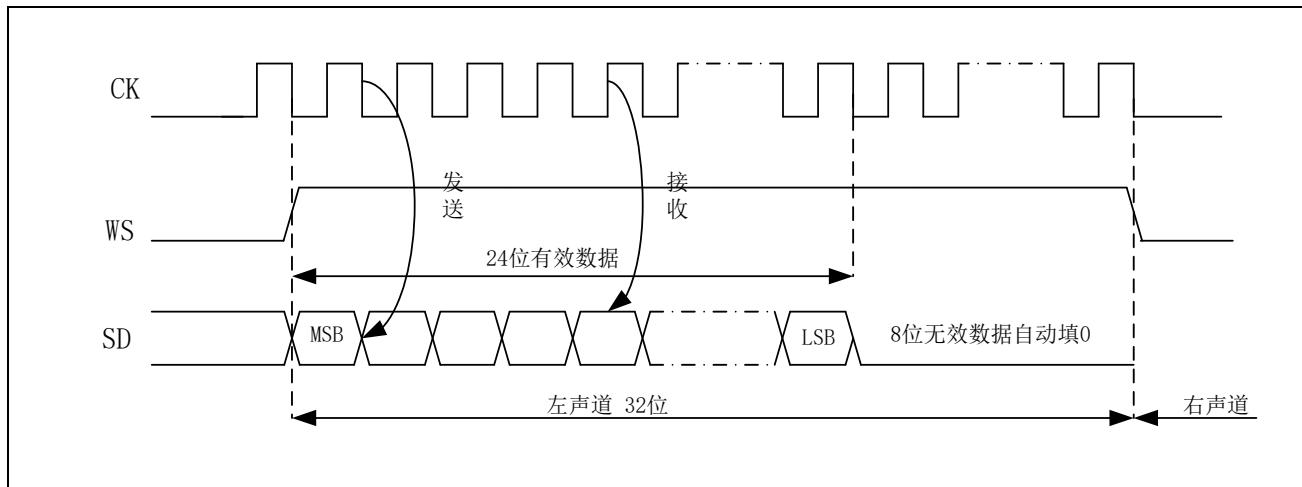


图 28-9 MSB 协议波形 (24 位帧)

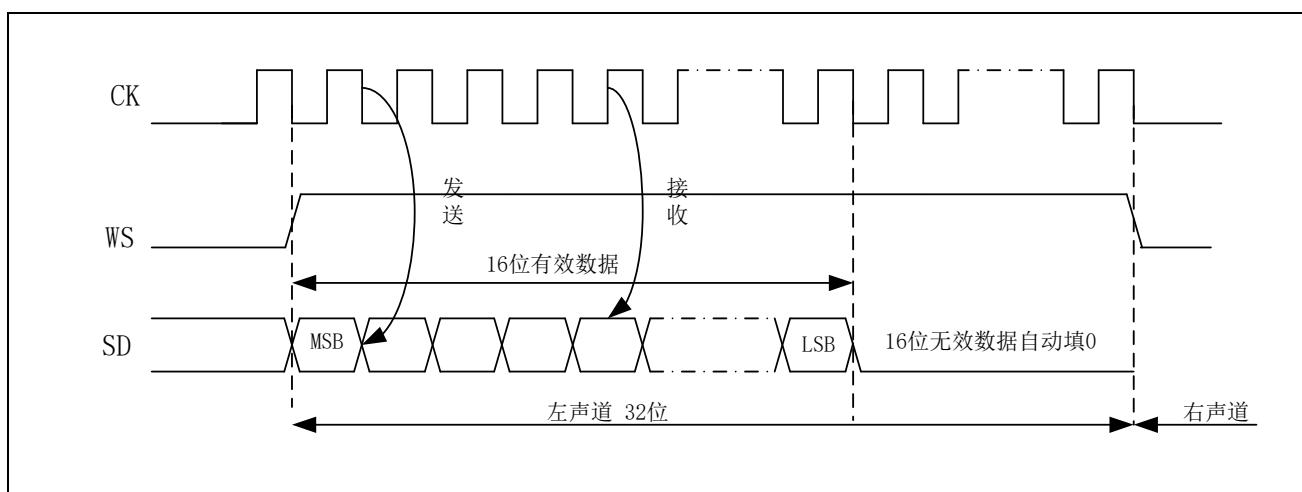


图 28-10 MSB 协议 16 位扩展到 32 位数据帧

28.2.3 LSB Alignment Standard

该标准与 MSB 对齐标准类似（对于 16 位和 32 位全精度帧格式，没有任何不同）。

发送方在时钟信号 (CK) 的下降沿改变数据，接收方在上升沿读取数据。WS 信号也在 CK 的下降沿变化。

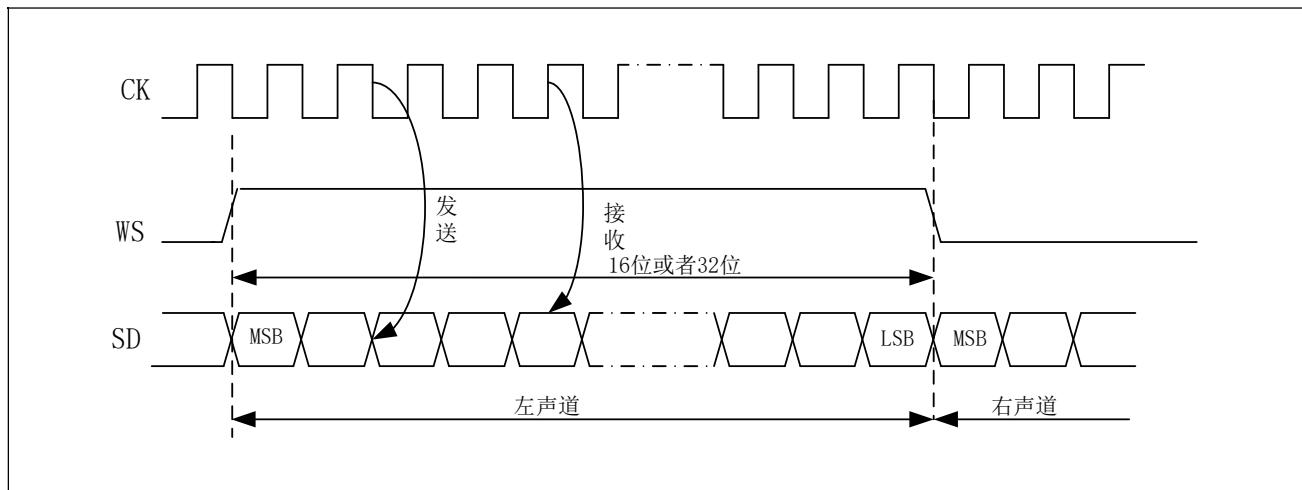


图 28-11 LSB 协议波形 (16/32 位全精度)

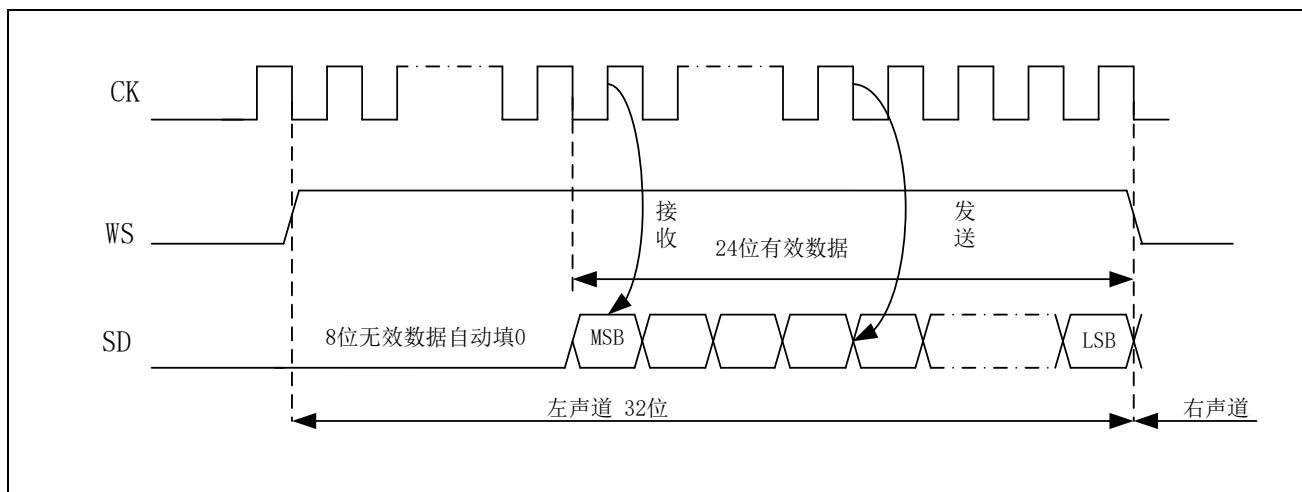


图 28-12 LSB 协议波形 (24 位帧)

在发送模式下，如果需要发送数据 0x89ABCD，则需要通过软件或 DMA 对 I2Sx_DRL/I2Sx_DRD 寄存器执行两次写入操作。下面给出了这些操作。

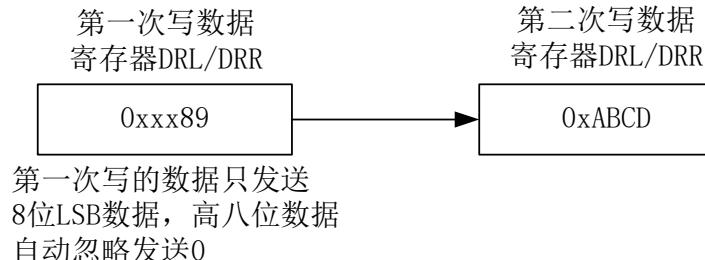


图 28-13 LSB 24 位数据发送写操作

在接收模式下，如果接收到数据 0x123456，则在每个 RXNE 事件时需要对 I2Sx_DR 寄存器执行两次连续的读取操作。

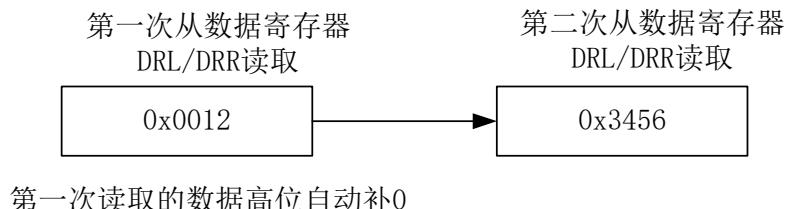


图 28-14 LSB 24 位数据模式接收读操作

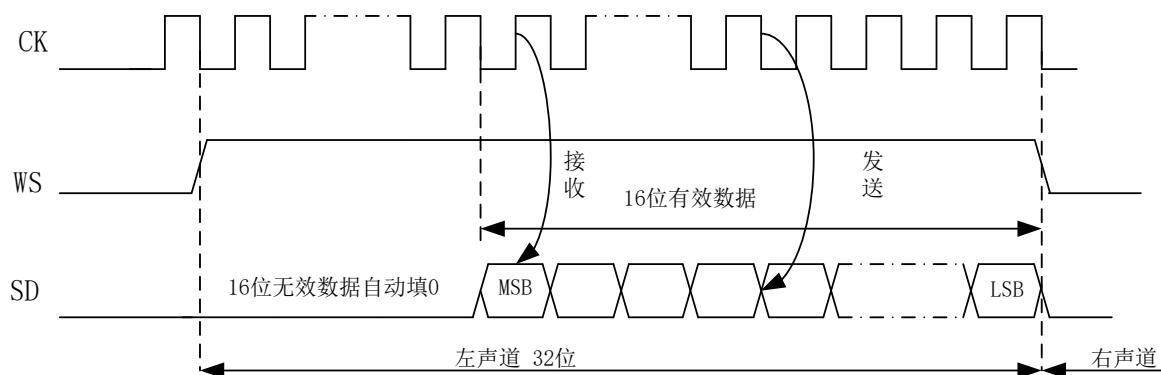


图 28-15 LSB 协议 16 位扩展到 32 位数据帧

如果在 I2S 配置阶段选择将 16 位数据帧扩展到 32 位通道帧，则只需要访问一次 I2Sx_DRL/I2Sx_DRD 寄存器。扩展到 32 位中，高半字（16 位 MSB）被硬件置为 0x0000。

如果要发送的数据或已接收的数据为 0x7893 (0x0000 7983 是扩展 32 位的数据)，则需要执行下中显示的操作。

只需要一次数据
寄存器 DRL/DRR
读写
0x7893

图 28-16 16 位数据读写操作

在发送模式下，相应通道发生 TXE 事件时，应用程序需要在相应通道写入要发送的数据（此例中，为 0x7893）。

首先发送 0x0000 字段（扩展到 32 位）。相应通道的有效数据（0x7893）转移到移位寄存器后，相应通道的 TXE 标志会被再次置 1。

在接收模式下，当接收到有效半字后（而非 0x0000 字段），即会置位相应通道的 RXNE。

28.2.4 PCM Standard

对于 PCM 标准，无需使用通道信息。可使用两种 PCM 模式（短帧和长帧），并且可使用 I2Sx_CFGR 寄存器中的 PCMSYNC 位来配置。PCM 发送数据使用上升沿，接收数据使用下降沿。

发送数据与接收数据只需要使用 I2Sx_DRL 数据寄存器，I2Sx_DRD 数据寄存器在这种模式下无效。

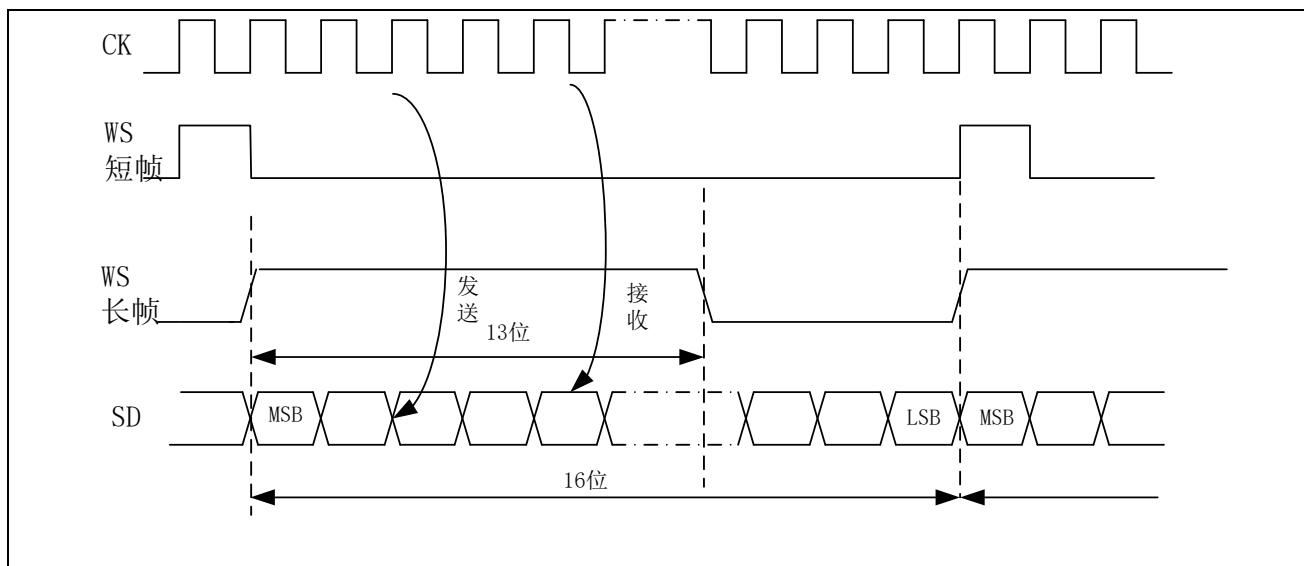


图 28-17 PCM 协议波形 (16 位)

对于长帧同步，在主模式下会将 WS 信号持续 13 个周期。

对于短帧同步，WS 同步信号的持续时间仅为一个周期。

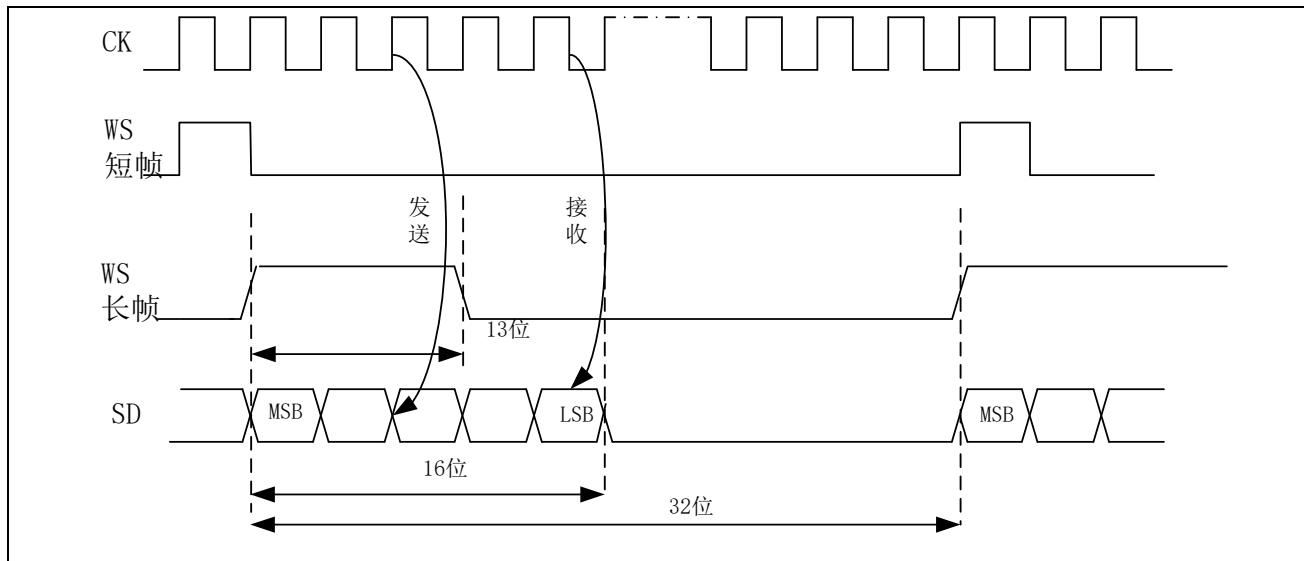


图 28-18 PCM 协议波形（16 位数据扩展到 32 位）

注意：

- 对于两种模式（主/从模式）和两种同步（短/长同步），即使在从模式下，也需要指定两组连续数据（以及两个同步信号）之间位的个数（I2Sx_CFGR 寄存器中的 DATLEN 位和 CHLEN 位）。

28.3 I2S Clock Generator

I2S 比特率用来确定 I2S 数据线上的数据流和 I2S 时钟信号频率。

I2S 比特率 = 每个通道的位数 × 通道数 × 音频采样频率

对于 16 位双通道音频，I2S 比特率的计算公式如下：

I2S 比特率 = $16 \times 2 \times f_S$

如果数据包为 32 位宽，则 I2S 比特率 = $32 \times 2 \times f_S$ 。

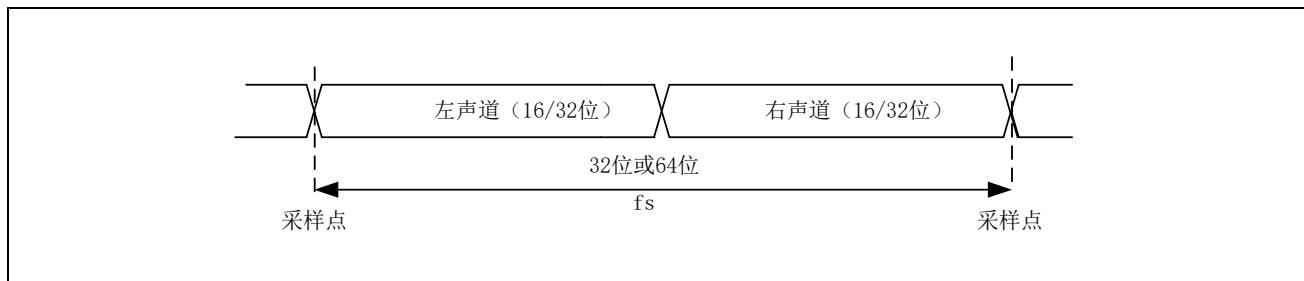


图 28-19 音频采样频率定义

配置主模式时，需要正确地对线性分频器进行设置，以便采用所需的音频频率进行通信。

下图展示了通信时钟架构。I2Sx 时钟可选择 PCLK 或者 HCLK。

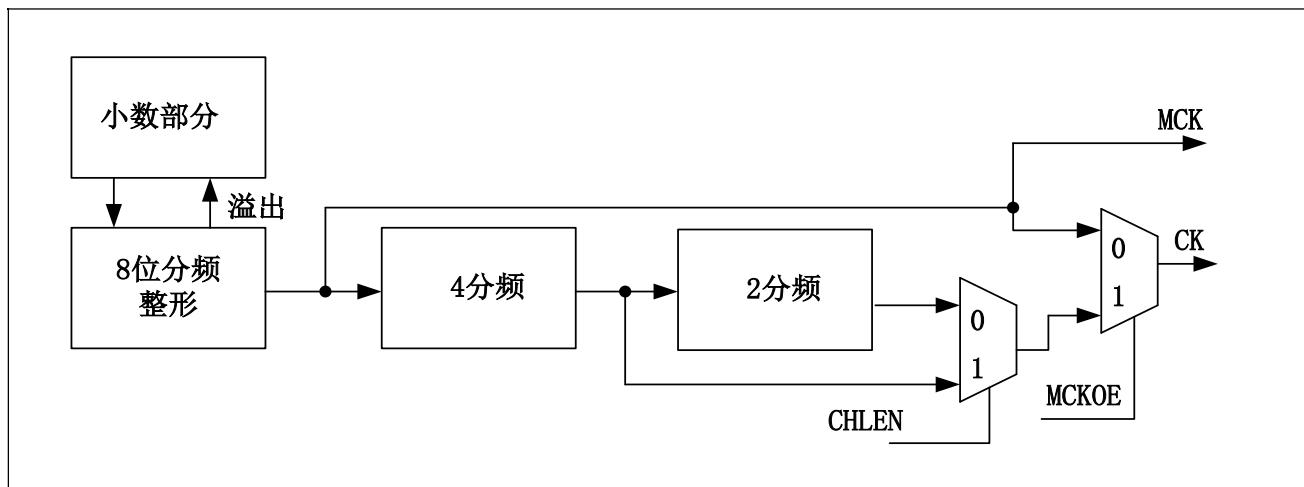


图 28-20 时钟产生

音频采样频率可以是 96 kHz、48 kHz、44.1 kHz、32 kHz、22.05 kHz、16 kHz、11.025 kHz 或 8 kHz（或此范围内的任何其它值）。为达到所需频率，需要根据以下公式对线性分频器进行编程：

小数分频 FRACT 等于 0x00 时：

输出主时钟 (I2Sx_PR 寄存器中的 MCKOE 置 1) 时：

$$f_S = I2SxCLK / [(16*2)*((2*I2SDIV)+ODD)*8] \text{ (通道帧宽度为 16 位时)}$$

$$f_S = I2SxCLK / [(32*2)*((2*I2SDIV)+ODD)*4] \text{ (通道帧宽度为 32 位时)}$$

禁止主时钟输出 (MCKOE 位清零) 时：

$$f_S = I2SxCLK / [(16*2)*((2*I2SDIV)+ODD)] \text{ (通道帧宽度为 16 位时)}$$

$$f_S = I2SxCLK / [(32*2)*((2*I2SDIV)+ODD)] \text{ (通道帧宽度为 32 位时)}$$

小数分频 FRACT 不等于 0x00 时：

输出主时钟 (I2Sx_PR 寄存器中的 MCKOE 置 1) 时：

$$f_S = I2SxCLK / \{ (16*2)* [2*(I2SDIV+FRACT/64)]*8 \} \text{ (通道帧宽度为 16 位时)}$$

$$f_S = I2SxCLK / \{ [(32*2)*[2*(I2SDIV+FRACT/64)]*4 \} \text{ (通道帧宽度为 32 位时)}$$

禁止主时钟输出 (MCKOE 位清零) 时：

$$f_S = I2SxCLK / \{ (16*2)* [2*(I2SDIV+FRACT/64)] \} \text{ (通道帧宽度为 16 位时)}$$

$$f_S = I2SxCLK / \{ (32*2)* [2*(I2SDIV+FRACT/64)] \} \text{ (通道帧宽度为 32 位时)}$$

下表提供了时钟 48M 时钟针对不同采样率的设置及精度值（不使用小数分频）。

I2S CLK	通道 长度	I2SDIV	I2SODD	MCLK	采样率	实际 频率	误差%	freq(mck)
48000000	16	8	0	无输出	96000	93750	-2.34	0
48000000	32	4	0	无输出	96000	93750	-2.34	0
48000000	16	15	1	无输出	48000	48387	0.81	0
48000000	32	8	0	无输出	48000	46875	-2.34	0
48000000	16	17	0	无输出	44100	44118	0.04	0
48000000	32	8	1	无输出	44100	44118	0.04	0
48000000	16	23	1	无输出	32000	31915	-0.27	0
48000000	32	11	1	无输出	32000	32609	1.9	0
48000000	16	34	0	无输出	22050	22059	0.04	0
48000000	32	17	0	无输出	22050	22059	0.04	0
48000000	16	47	0	无输出	16000	15957	-0.27	0
48000000	32	23	1	无输出	16000	15957	-0.27	0
48000000	16	68	0	无输出	11025	11029	0.04	0
48000000	32	34	0	无输出	11025	11029	0.04	0
48000000	16	94	0	无输出	8000	7979	-0.26	0
48000000	32	47	0	无输出	8000	7979	-0.26	0
48000000	16	2	0	输出	48000	46875	-2.34	12000000
48000000	32	2	0	输出	48000	46875	-2.34	12000000
48000000	16	2	0	输出	44100	46875	6.29	12000000
48000000	32	2	0	输出	44100	46875	6.29	12000000
48000000	16	3	0	输出	32000	31250	-2.34	8000000
48000000	32	3	0	输出	32000	31250	-2.34	8000000
48000000	16	4	1	输出	22050	20833	-5.52	5333248
48000000	32	4	1	输出	22050	20833	-5.52	5333248
48000000	16	6	0	输出	16000	15625	-2.34	4000000
48000000	32	6	0	输出	16000	15625	-2.34	4000000
48000000	16	8	1	输出	11025	11029	0.04	2823424
48000000	32	8	1	输出	11025	11029	0.04	2823424
48000000	16	11	1	输出	8000	8152	1.9	2086912
48000000	32	11	1	输出	8000	8152	1.9	2086912

下表提供了时钟 48M 时钟使用小数分频针对不同采样率的设置及精度值。

I2S CLK	通道 长度	I2SDIV	FRACT	MCLK	采样率	实际 频率	误差%	freq(mck)
48000000	16	7	52	无输出	96000	96000	0	0
48000000	32	3	58	无输出	96000	96000	0	0
48000000	16	15	40	无输出	48000	48000	0	0
48000000	32	7	52	无输出	48000	48000	0	0
48000000	16	17	0	无输出	44100	44118	0.04	0
48000000	32	8	32	无输出	44100	44118	0.04	0
48000000	16	23	28	无输出	32000	32000	0	0
48000000	32	11	46	无输出	32000	32000	0	0
48000000	16	34	1	无输出	22050	22049	0	0
48000000	32	17	0	无输出	22050	22059	0.04	0
48000000	16	46	56	无输出	16000	16000	0	0
48000000	32	23	28	无输出	16000	16000	0	0
48000000	16	68	2	无输出	11025	11024	-0.01	0
48000000	32	34	1	无输出	11025	11024	-0.01	0
48000000	16	93	48	无输出	8000	8000	0	0
48000000	32	46	56	无输出	8000	8000	0	0
48000000	16	1	61	输出	48000	48000	0	12288000
48000000	32	1	61	输出	48000	48000	0	12288000
48000000	16	2	8	输出	44100	44118	0.04	11294208
48000000	32	2	8	输出	44100	44118	0.04	11294208
48000000	16	2	60	输出	32000	31915	-0.27	8170240
48000000	32	2	60	输出	32000	31915	-0.27	8170240
48000000	16	4	16	输出	22050	22059	0.04	5647104
48000000	32	4	16	输出	22050	22059	0.04	5647104
48000000	16	5	55	输出	16000	16000	0	4096000
48000000	32	5	55	输出	16000	16000	0	4096000
48000000	16	8	32	输出	11025	11029	0.04	2823424
48000000	32	8	32	输出	11025	11029	0.04	2823424
48000000	16	11	46	输出	8000	8000	0	2048000
48000000	32	11	46	输出	8000	8000	0	2048000

28.4 I2S Operating Modes

28.4.1 I2S Master Mode

I2S 可配置为主模式。这意味着将在 CK 引脚输出串行时钟，在 WS 引脚生成字选信号。主时钟 (MCK) 可以输出，也可以不输出，具体由 I2Sx_PR 寄存器中的 MCKOE 位控制。

步骤：

1. 设置 I2Sx_PR 寄存器的 I2SDIV[7:0] 位，以定义串行时钟波特率，从而达到相应的音频采样频率。I2Sx_PR 寄存器的 ODD 与 FRACT 位根据需要进行设置。
2. 如果需要为外部 DAC/ADC 音频组件提供主时钟 MCK，则将 I2Sx_PR 寄存器的 MCKOE 位置 1 (I2SDIV 和 ODD (FRACT) 值应根据 MCK 输出的状态进行计算。有关详细信息，请参见时钟发生器。
3. 通过 I2SSTD[1:0] 和 PCMSYNC 位选择 I2S 标准，通过 DATLEN[1:0] 位选择数据长度并通过配置 CHLEN 位选择每个通道的位数。此外，通过 I2Sx_CFGR 寄存器的 I2SCFG[1:0] 位选择 I2S 主模式和方向（发送器或接收器）。
4. 如果需要，通过对 I2Sx_CR2 寄存器执行写操作来选择所有可能的中断源和 DMA 功能。
5. I2Sx_CFGR 寄存器的 I2SE 位必须置 1。

WS 和 CK 的 IO 配置为输出模式。如果 I2Sx_PR 的 MCKOE 位置 1，则 MCK 也需要配置为输出。

发送序列

将半字写入发送缓冲区后，发送序列随即开始。

假设写入发送缓冲区的左声道第一个数据写入数据寄存器 L；写入发送缓冲区的右声道第一个数据写入数据寄存器 R。数据从发送缓冲区传输到移位寄存器时，相应通道的 TXE 置 1，并且必须将对应于相应通道的数据写入相应通道的发送缓冲区。

一个完整帧表示先进行左通道数据发送再进行右通道数据发送。不存在仅发送左通道的部分帧。

首位发送期间，数据按半字并行加载到 16 位移位寄存器中，然后以串行方式移位并输出到 MOSI/SD 引脚 (MSB 在前)。每次数据从发送缓冲区传输到移位寄存器后，相应通道的 TXE 标志都将置 1，如果 I2Sx_CR2 寄存器的 TXEIE 位置 1，将产生中断。

有关各种 I2S 标准模式的写操作的更多详细信息，请参见 I2S 音频协议说明。

为确保连续进行音频数据发送，必须在当前数据发送结束前将下一个要发送的数据写入 I2Sx_DRL/DRR 寄存器。

要通过将 I2SE 清零来关闭 I2S，必须等待 TXE = 1 且 BSY = 0。

接收序列

此工作模式与发送模式相同，只有第 3 点存在不同（请参见 I2S 主模式所述的步骤），即通过 I2SCFG[1:0] 位设置主器件接收模式。

无论数据或通道长度如何，音频数据始终按 16 位数据包进行接收。这意味着，每当接收相应通道缓冲区有接收到数据时，相应通道的 RXNE 标志即置 1，并且如果 I2Sx_CR2 寄存器的 RXNEIE 位置 1，还将产生中断。所接收的右通道或左通道的音频值可通过一次或两次接收操作进入相应通道的接收缓冲区，具体取决于数据和通道长度配置。

读取 I2Sx_DRL/DRR 寄存器即会使相应通道的 RXNE 位清零。

有关各种 I2S 标准模式中读操作的更多详细信息，请参见 I2S 音频协议说明。

如果在先前收到的数据尚未读取时又接收到新数据，将产生上溢错误并将 OVR 标志置 1。数据接收寄存的值保持先前的值，当前接收的数据丢失。

如果 I2Sx_CR2 寄存器的 ERRIE 位置 1，将产生中断以指示该错误。

要关闭 I2S，需要执行特定操作来确保 I2S 正确完成传输周期而不启动新的数据传输。该序列取决于数据和通道长度的配置，以及所选的音频协议模式。在以下情况下：

32 位通道长度上扩展的 16 位数据长度 (DATLEN = 00 且 CHLEN = 1)，使用 LSB 对齐模式 (I2SSTD = 10)

- a) 等待倒数第二个 RXNE = 1 ($n - 1$)
- b) 然后等待 17 个 I2S 时钟周期（使用软件循环）
- c) 禁止 I2S (I2SE = 0)

32 位通道长度上扩展的 16 位数据长度 (DATLEN = 00 且 CHLEN = 1)，使用 MSB 对齐、I2S 或 PCM 模式（分别为 I2SSTD = 00、I2SSTD = 01 或 I2SSTD = 11）

- a) 等待最后一个 RXNE
- b) 然后等待 1 个 I2S 时钟周期（使用软件循环）
- c) 禁止 I2S (I2SE = 0)

对于 DATLEN 和 CHLEN 的所有其它组合，无论通过 I2SSTD 位选择何种音频模式，都将执行以下序列来关闭 I2S：

- a) 等待倒数第二个 RXNE = 1 ($n - 1$)
- b) 然后等待一个 I2S 时钟周期（使用软件循环）
- c) 禁止 I2S (I2SE = 0)

注：传输期间，BSY 标志保持低电平。

28.4.2 I2S Slave Mode

对于从配置而言，I2S 可配置为发送模式或接收模式。

此工作模式所遵循的规则与 I2S 主模式配置基本相同。在从模式下，I2S 接口不产生时钟。

时钟和 WS 信号从 I2S 接口所连接的外部主器件输入。这样，用户便不需要配置时钟。

应遵循如下配置步骤：

1. 通过 I2SSTD[1:0] 位选择 I2S 标准，通过 DATLEN[1:0] 位选择数据长度并通过配置 CHLEN 位选择帧中每个通道的位数。此外，通过 I2Sx_CFGR 寄存器的 I2SCFG[1:0] 位选择从器件的模式（发送或接收）。
2. 如果需要，通过对 I2Sx_CR2 寄存器执行写操作来选择所有可能的中断源和 DMA 功能。
3. I2Sx_CFGR 寄存器的 I2SE 位必须置 1。

发送序列

当外部主器件发送时钟并且通过 I2Sx_WS 信号请求传输数据时，发送序列开始。必须首先使能从器件，然后外部主器件才能开始通信。主器件开始通信前，还必须加载 I2S 数据寄存器。

对于 I2S、MSB 对齐和 LSB 对齐模式，要发送的第一个数据为左声道数据，分别在左右声道数据寄存器写入相应通道的数据。通信开始时，数据从发送缓冲区传输到移位寄存器。相应通道 TXE 标志随即置 1，以请求将相应通道的下一个数据写入 I2S 相应通道数据寄存器。

从器件需要首先为发送第一个数据做好准备，然后主器件才能产生时钟。WS 置位意味着首先发送左通道数据。

注：必须要在主器件发出的第一个时钟出现在 CK 线上至少 2 个 PCLK 周期之前置位 I2SE。

首位发送期间，数据按半字从内部总线并行加载到 16 位移位寄存器中，然后以串行方式移位并输出到 MOSI/SD 引脚（MSB 在前）。每次数据从发送缓冲区传输到移位寄存器后，相应通道的 TXE 标志都将置 1，如果 I2Sx_CR2 寄存器的 TXEIE 位置 1，将产生中断。

请注意，仅当 TXE 标志为 1 时，才可以尝试向发送缓冲区写入数据。

有关各种 I2S 标准模式中写操作的更多详细信息，请参见 I2S 音频协议说明。

为确保连续进行音频数据发送，必须在当前数据发送结束前将下一个要发送数据写入 I2Sx_DRL/R 寄存器。如果在数据尚未写入 I2Sx_DR 寄存器时下一个数据通信的首个时钟边沿到来，下溢标志将置 1 并可能产生中断。通过这种方式，软件可以获知所传输的数据不正确。如果 I2Sx_CR2 寄存器的 ERRIE 位置 1，则当 I2Sx_SR 寄存器中的 UDR 标志变为 1 时，将产生中断。这种情况下，必须关闭 I2S 并从左通道开始重新启动数据传输。

要通过将 I2SE 位清零来关闭 I2S，必须等待 TXE = 1 且 BSY = 0。

接收序列

此工作模式与发送模式相同，只有第 1 点存在不同（请参见 I2S 从模式所述的步骤），即通过 I2Sx_CFG 寄存器的 I2SCFG[1:0] 位设置从器件接收模式。

无论数据长度或通道长度如何，音频数据始终按 16 位数据包进行接收。这意味着，每当接收缓冲区有数据时，I2Sx_SR 寄存器中的相应通道的 RXNE 标志即置 1，并且如果 I2Sx_CR2 寄存器的 RXNEIE 位置 1，还将产生中断。所接收的右通道或左通道的音频值可能通过一次或两次接收操作进入相应通道的接收缓冲区，具体取决于数据长度和通道长度配置。

每次接收的数据要从相应通道的数据寄存器读取，读取 I2Sx_DR 寄存器即会使 RXNE 位清零。

有关各种 I2S 标准模式的读操作的更多详细信息，请参见 I2S 音频协议说明。

如果在先前收到的数据尚未读取时又接收到新数据，将产生上溢错误，OVR 标志将置 1。如果 I2Sx_CR2 寄存器的 ERRIE 位置 1，将产生中断以指示该错误。

要在接收模式下关闭 I2S，必须在接收到最后一个 RXNE = 1 后立即将 I2SE 清零。

注：外部主器件应能够通过音频通道以 16 位或 32 位数据包发送/接收数据。

28.5 I2S Status Flags

应用程序可通过三个状态标志来全面监视 I2S 总线的状态。

28.5.1 Busy Flag (BSY)

BSY 标志由硬件置 1 和清零（写入此标志没有任何作用）。该标志表示 I2S 通信层的状态。

BSY 置 1 时，表示 I2S 正在忙于通信。在主接收模式 (I2SCFG = 11) 中，BSY 标志的情况例外，该标志在接收期间仍保持低电平。

如果软件需要禁止 I2S，可使用 BSY 标志检测传输是否结束。这可以避免损坏最后传输的数据。为此，必须严格遵循下述步骤。

在传输开始时（I2S 处于主接收器模式时除外），将 BSY 标志置 1

出现以下情况时，BSY 标志被硬件清零：

- 传输完成时（主发送模式除外，在该模式下通信是连续的）
- 禁止 I2S 时

当通信连续时：

- 在主发送模式下，BSY 标志在所有传输期间均保持高电平
- 在从模式下，BSY 标志在每次传输之间变为低电平并持续一个 I2S 时钟周期

注：请勿使用 BSY 标志处理每次数据发送或接收，最好改用 TXE 标志和 RXNE 标志。

28.5.2 Transmit Buffer Empty (TXE_L TXE_R)

如果此标志置 1，表示发送缓冲区为空，可将要发送的数据加载到其中。发送缓冲区已包含要发送的数据时，TXE 标志复位。禁止 I2S (I2SE 位复位) 时，该标志也会复位。

28.5.3 Receive Buffer Not Empty (RXNE_L RXNE_R)

此标志置 1 时，表示接收缓冲区中存在有效的已接收数据。读取 I2Sx_DRL/R 寄存器时，该标志复位。

28.5.4 Underflow Flag (UDR)

在发送模式下，如果在软件尚未将任何值加载到 I2Sx_DR 之前出现第一个数据发送时钟，此标志将置 1。如果 I2Sx_CR2 寄存器中的 ERRIE 位置 1，可产生中断。

UDR 位通过 I2Sx_ICLR 进行清零。

28.5.5 Overflow Flag (OVR)

如果在尚未从 I2Sx_DR 寄存器读取上一个数据时又接收到新数据，此标志将置 1。因此，传入的数据将丢失。如果 I2Sx_CR2 寄存器中的 ERRIE 位置 1，可产生中断。

这种情况下，将不会用接收到的新数据更新接收缓冲区的内容。对 I2Sx_DR 寄存器执行的读操作将返回先前正确接收的数据。主器件后续发送的所有其它半字都将丢失。

要将 OVR 位清零，应首先对 I2Sx_DR 寄存器执行读操作，然后再对 I2Sx_SR 寄存器进行读访问。

28.5.6 Frame Error Flag (FRE)

仅当 I2S 配置为从模式时，此标志才可由硬件置 1。如果外部主器件没有按照从器件期望的那样改变 WS 线，则此标志将置 1。如果失去同步，要从此状态中恢复并将外部主器件与

I2S 从器件重新同步，需执行下列步骤：

1. 禁止 I2S。
2. 在 WS 线上检测到正确的电平时将其重新使能 (WS 线在 I2S 模式下为高电平，在 MSB 对齐、LSB 对齐或 PCM 模式下为低电平)。

主器件与从器件之间的同步失效可能是由于 SCK 通信时钟或 WS 帧同步信号线上存在噪音干扰。如果 ERRIE 位置 1，可产生错误中断。同步失效标志 (FRE) 由软件操作 I2Sx_ICLR 进行清零。

28.6 I2S Interrupt DMA

下表为 I2S 中断列表

中断事件	事件标志	使能控制
发送缓存空	TXE_L TXE_R	TXEIE
接收缓存非空	RXNE RXNE_R	RXNEIE
上溢错误	OVR	ERRIE
下溢错误	UDR	
帧错误	FRE	

DMA 特性

在 I2S 模式下，发送数据和接收的数据可以使用 DMA 进行搬运。

28.7 I2S Registers

基地址

I2S0: 0x4000 2800

I2S1: 0x4000 2C00

表 28-1 I2Sx 寄存器

寄存器	偏移地址	描述
I2Sx_CR	0x004	控制寄存器
I2Sx_SR	0x008	中断和状态寄存器
I2Sx_CFGR	0x01C	配置寄存器
I2Sx_PR	0x020	主模式分频寄存器
I2Sx_ICR	0x024	中断标志清除寄存器
I2Sx_DRL	0x028	左声道发送数据与接收数据寄存器
I2Sx_DRD	0x02C	右声道发送数据与接收数据寄存器

28.7.1 I2Sx Control Register (I2Sx_CR)

偏移地址 0x04

复位值 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								TXEIE	RXNEIE	ERRIE	Res.		RDMAEN	LDMAEN	
								RW	RW	RW			RW	RW	

位	标记	功能描述
31:8	Reserved	保留
7	TXEIE	发送缓冲区空中断使能 (Tx buffer empty interrupt enable) 0: 屏蔽 TXE 中断 1: 使能 TXE 中断。TXE 标志置 1 时产生中断请求。
6	RXNEIE	接收缓冲区非空中断使能 (RX buffer not empty interrupt enable) 0: 屏蔽 RXNE 中断 1: 使能 RXNE 中断 RXNE 标志置 1 时产生中断请求。
5	ERRIE	错误中断使能 (Error interrupt enable) 此位用于在出现错误条件(UDR、OVR 和 FRE) 时控制中断的生成。 0: 屏蔽错误中断 1: 使能错误中断
4:2	Reserved	保留
1	RDMAEN	右声道缓冲区 DMA 数据发送接使能 (Right buffer DMA enable) 当此位置 1 时，每当右声道有数据发送接收标志置 1 时，即产生 DMA 请求。 0: 禁止右声道缓冲区 DMA 1: 使能右声道缓冲区 DMA
0	LDMAEN	左声道缓冲区 DMA 声道数据发送接使能 (Left buffer DMA enable) 当此位置 1 时，每当左声道有数据发送接标志置 1 时，即产生 DMA 请求。 0: 禁止左声道缓冲区 DMA 1: 使能左声道缓冲区 DMA

28.7.2 I2Sx Status Register (I2Sx_SR)

偏移地址 0x08

复位值 0x0000 8002

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE_R RO	RXNE_R RO	OVR_R RO	Res			FRE RO	BSY RO	OVR_L RO	Res.		UDR_R RO	UDR_L RO	TXE_L RO	RXNE_L RO	

位	标记	功能描述
31:16	Reserved	保留
15	TXE_R	右通道发送缓冲区为空 (Transmit buffer empty) 0: 发送缓冲区非空 1: 发送缓冲区为空
14	RXNE_R	右通道接收缓冲区非空 (Receive buffer not empty) 0: 发送缓冲区为空 1: 接收缓冲区非空
13	OVR_R	右通道上溢标志 (Overrun flag) 0: 未发生上溢 1: 发生上溢 此标志由硬件置 1, 由软件清零
12:9	Reserved	保留
8	FRE	帧错误 (Frame Error) 0: 无帧错误 1: 发生帧错误 该位由硬件置 1, 由软件清零。 无论选择哪种音频协议, 它会在从模式下的非预期时间检测 WS 线的变化, 通知外部主器件和从器件之间的同步失效情况。
7	BSY	BSY: 忙标志 (Busy flag) 0: I2S 不繁忙 1: I2S 忙于通信或者发送缓冲区不为空 此标志由硬件置 1 和清零。
6	OVR_L	左通道上溢标志 (Overrun flag) 0: 未发生上溢 1: 发生上溢 此标志由硬件置 1, 由软件清零
5:4	Reserved	保留
3	UDR_R	右通道下溢标志 (Underrun flag) 0: 未发生下溢 1: 发生下溢

		此标志由硬件置 1，由软件清零
2	UDR_L	左通道下溢标志 (Underrun flag) 0: 未发生下溢 1: 发生下溢 此标志由硬件置 1，由软件清零
1	TXE_L	左通道发送缓冲区为空 (Transmit buffer empty) 0: 发送缓冲区非空 1: 发送缓冲区为空
0	RXNE_L	左通道接收缓冲区非空 (Receive buffer not empty) 0: 发送缓冲区为空 1: 接收缓冲区非空

28.7.3 I2Sx Configuration Register (I2Sx_CFGR)

偏移地址 0x1C

复位值 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					I2SE	I2SCFG	PCMSYNC	CKSEL	I2SSTD	Res	DATLEN	CHLEN			
					RW	RW	RW	RW	RW	Res	RW	RW			

位	标记	功能描述
31:11	Reserved	保留
10	I2SE	I2S 使能 (I2S Enable) 0: 禁止 I2S 1: 使能 I2S
9:8	I2SCFG	I2S 配置模式 (I2S configuration mode) 00: 从模式 - 发送 01: 从模式 - 接收 10: 主模式 - 发送 11: 主模式 - 接收
7	PCMSYNC	PCM 帧同步 (PCM frame synchronization) 0: 短帧同步 1: 长帧同步 注: 只有在 I2SSTD = 11 (使用 PCM 标准) 时, 此位才有意义
6	CKSEL	主模式下 I2S 时钟选择 0 PCLK 1 HCLK
5:4	I2SSTD	I2S 标准选择 (I2S standard selection) 00: I2S Philips 标准。 01: MSB 对齐标准 (左对齐) 10: LSB 对齐标准 (右对齐) 11: PCM 标准 注: 为确保正确运行, 应在 I2S 禁止时配置这些位
3	Reserved	保持值为 0
2:1	DATLEN	要传输的数据长度 (Data length to be transferred) 00: 16 位数据长度 01: 24 位数据长度 10: 32 位数据长度 11: 不允许 注: 为确保正确运行, 应在 I2S 禁止时配置这些位。
0	CHLEN	CHLEN: 通道长度 (每个音频通道的位数) (Channel length (number of bits per audio channel)) 0: 16 位

		<p>1: 32 位</p> <p>只有在 DATLEN = 00 时，此位的写 0 操作才有意义，在其他数据长度下此位被强制写 1。</p> <p>注：为确保正确运行，应在 I2S 禁止时配置此位。</p>
--	--	--

28.7.4 I2Sx Prescaler Register (I2Sx_PR)

偏移地址 0x20

复位值 0x0000 0002

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACT				MCKOE		ODD		I2SDIV							
RW				RW		RW		RW							

位	标记	功能描述
31:16	Reserved	保留
15:10	FRACT	预分频器小数分频设置，0x00 时 ODD 设置有效，小数分频为 0，非零时，ODD 设置无效 实际分频值为 = (I2SDIV + FRACT/64)*2 注： 应在 I2S 禁止时配置此位。
9	MCKOE	主时钟输出使能 (Master clock output enable) 0：禁止主时钟输出 1：使能主时钟输出 注： 应在 I2S 禁止时配置此位。
8	ODD	预分频器的奇数因子 (Odd factor for the prescaler) 0：实际分频值为 = I2SDIV *2 1：实际分频值为 = (I2SDIV * 2)+1 详细参见预分频设置 注： 应在 I2S 禁止时配置此位。
7:0	I2SDIV	I2SDIV: I2S 线性预分频器 (I2S Linear prescaler) I2SDIV [7:0] = 0 为禁用值。 详细参见预分频设置 注： 应在 I2S 禁止时配置这些位。

28.7.5 I2Sx Interrupt Status Clear Register (I2Sx_ICR)

偏移地址 0x24

复位值 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							FRE W0	Res	OVR W0	Res		UDF W0	Res		

位	标记	功能描述
31:9	Reserved	保留
8	FRE	帧错误 (Frame Error) 标志清除 写 0 清除 写 1 无效
7	Reserved	保留
6	OVR	上溢标志 (Overrun flag) 写 0 清除 写 1 无效
5:4	Reserved	保留
3	UDF	下溢标志 (Underrun flag) 写 0 清除 写 1 无效
2:0	Reserved	保留

28.7.6 I2Sx Data Register L (I2Sx_DRL)

偏移地址 0x28

复位值 0x0000 0000

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

位	标记	功能描述
31:16	Reserved	保留
15:0	DRL	数据寄存器 (Data register) 已接收或者要发送的左声道数据与 PCM 的数据。 数据寄存器分为 2 个缓冲区，一个用于写入（发送缓冲区），一个用于读取（接收缓冲区）。对数据寄存器执行写操作时，数据将写入发送缓冲区，从数据寄存器执行读取

		时，将返回接收缓冲区中的值。
--	--	----------------

28.7.7 I2Sx Data Register R (I2Sx_DRR)

偏移地址 0x2C

复位值 0x0000 0000

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

位	标记	功能描述
31:16	Reserved	保留
15:0	DRR	数据寄存器 (Data register) 已接收或者要发送的右声道数据。PCM 模式下无效。 数据寄存器分为 2 个缓冲区，一个用于写入（发送缓冲区），一个用于读取（接收缓冲区）。对数据寄存器执行写操作时，数据将写入发送缓冲区，从数据寄存器执行读取时，将返回接收缓冲区中的值。

29 USB 2.0 full speed module (USBFS)

29.1 USBFS Introduction

USB 全速 (USBFS) 控制器为便携式设备提供了一套 USB 通信解决方案。USBFS 控制器支持设备模式，且芯片内部集成全速 PHY。设备模式下支持全速 (FS, 12Mb/s) 收发器。USBFS 控制器支持 USB 2.0 协议所定义的所有四种传输方式 (控制传输、批量传输、中断传输和同步传输)。

29.2 USBFS Main Features

主要分为两类：通用特性和设备模式特性。

29.2.1 General features

- 内置片上 USB 2.0 全速 PHY
- 支持设备模式
- 模块内嵌 DMA，并可软件配置 AHB 突发传输类型
- 具备省电功能，例如 USB 挂起，停止 RAM 时钟，停止 PHY 域时钟
- 具有采用高级 FIFO 控制的 1.25KB 专用 RAM
- 可以将 RAM 空间划分为不同的 FIFO，以便灵活有效的使用 RAM
- 每个 FIFO 可存储多个数据包
- 动态分配存储区
- FIFO 的大小可配置成为非 2 的幂次方值，以便连续使用存储单元
- 一帧之内可以不需要应用程序干预，以达到最大 USB 带宽

29.2.2 Device Mode Features

- 设备模式支持 USB 2.0 全速 (FS, 12Mb/s) 传输。
- 1 个双向控制端点 0
- 4 个 IN 端点 (1, 3, 5, 7)，可以配置为批量传输、中断传输或同步传输
- 4 个 OUT 端点 (2, 4, 6, 8)，可以配置为批量传输、中断传输或同步传输
- 8 个端点只能选择一个配置为同步传输 (周期传输) 模式
- 包含一个非周期发送 FIFO (由所有非周期 IN 端点共享)，一个周期发送 FIFO (由周期 IN 端点独享) 和一个接收 FIFO (由所有 OUT 端点共享)
- 支持远程唤醒功能。
- 支持软断开功能

29.3 USBFS System Block Diagram

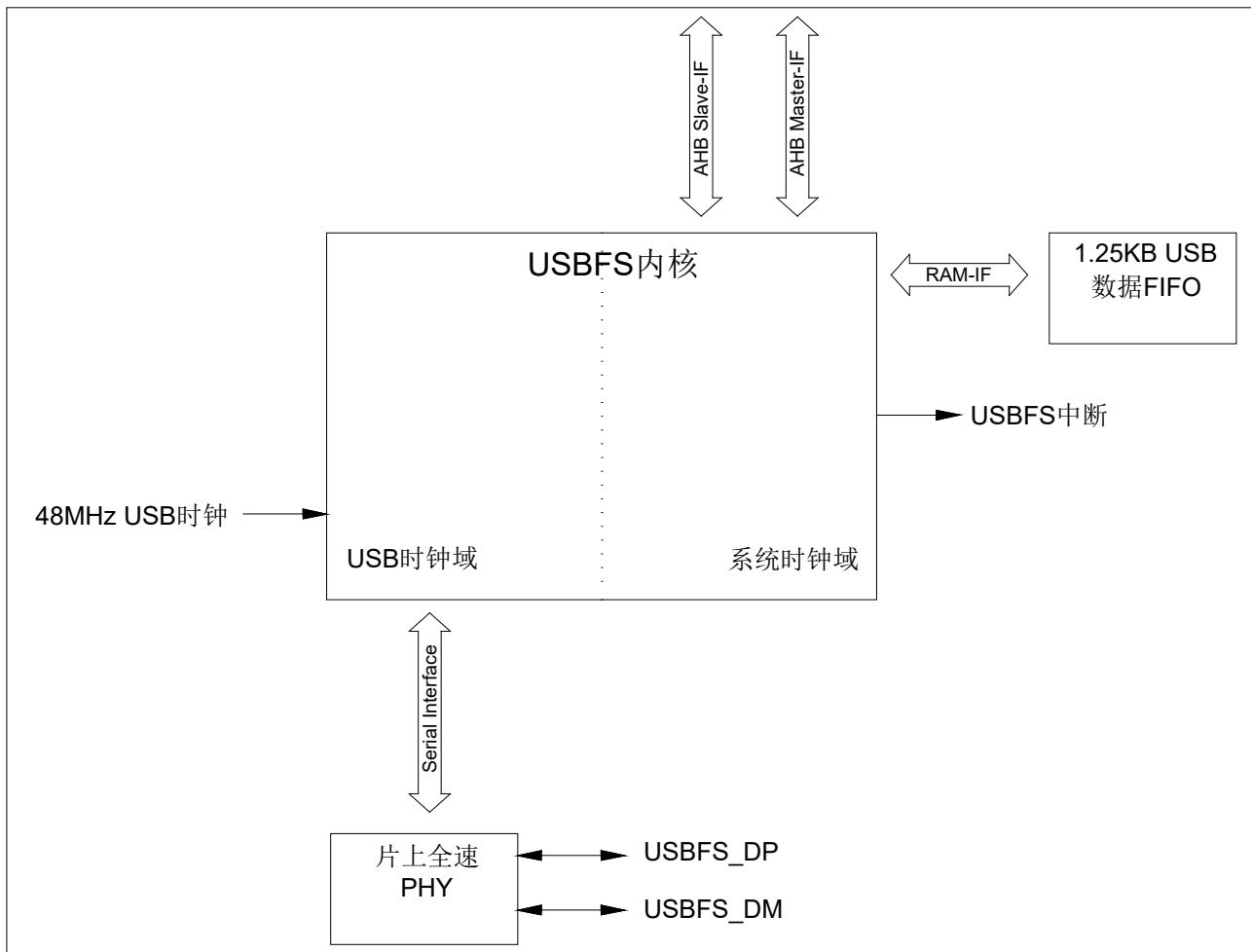


图 29-1 USBFS 系统框图

29.4 USBFS Pin Description

表 29-1 USBFS 管脚说明

管脚名	方向	功能描述
USBFS_DP	输入/输出	差分数据D+信号
USBFS_DM	输入/输出	差分数据D-信号

由于 USBFS_DP 和 USBFS_DM 管脚和通用 GPIO 复用，在使用 USBFS 时，建议关闭其对应管脚的数字功能，具体请参考端口控制器（GPIO）章节。另外 USBFS 功能不使用时，USBFS_DP 和 USBFS_DM 管脚对应的数字功能管脚翻转时，会产生额外的电流消耗。

29.5 USBFS Functional Description

29.5.1 USBFS clock and operating mode

USBFS 所使用的时钟有 PHY 时钟和系统时钟，其中 PHY 时钟频率需要配置为 48MHz，该 48MHz 时钟可由内部 PLL 电路产生，也可由内部高速 48M RC 时钟产生，使用 USBFS 模块前，需要在系统控制器模块内配置好 USBFS 时钟。

USBFS 作为设备使用，并且包含一个片上全速 PHY。

上拉和下拉电阻已经集成在片上全速 PHY 的内部，并且 USBFS 可以根据当前模式和连接状态自动选择。

USBFS 工作时，VCC 电压范围为 3.0~3.6V。

29.5.2 USBFS Device Functionality

29.5.2.1 Introduction to device functions

典型的 USB 设备模式系统构建图如下：

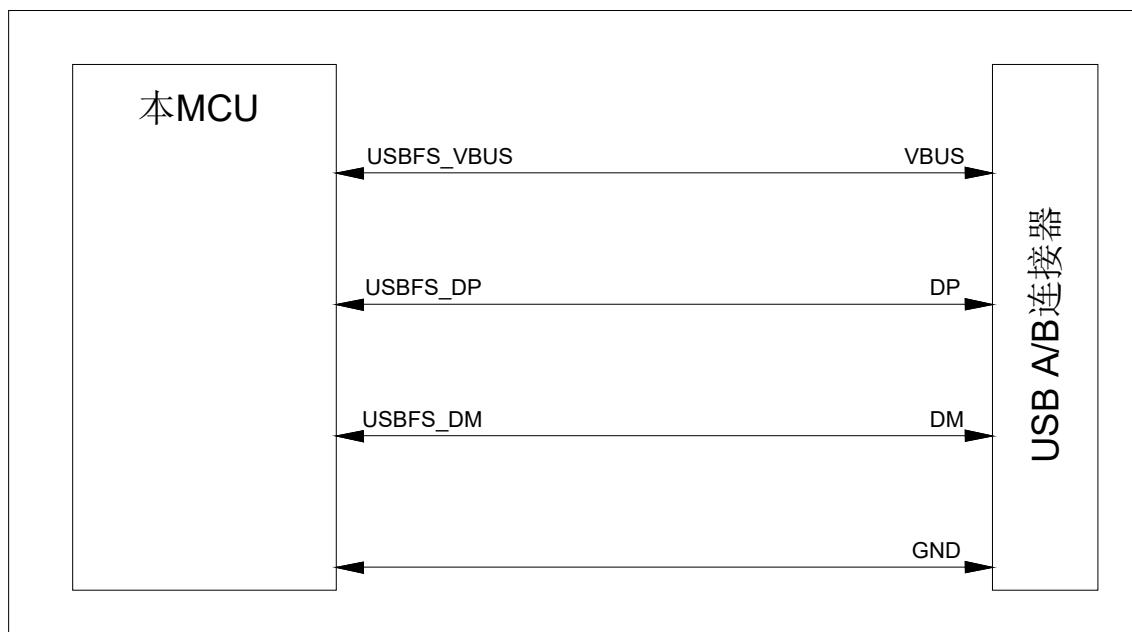


图 29-2 USBFS 设备模式系统构建图

29.5.2.2 Device Power Supply Status

模块检测到 USBFS_VBUS 为高电平时，就会使 USB 设备进入供电状态（请参见 USB2.0 第 9.1 节）。然后，USBFS 自动连接 DP 上拉电阻，发出全速设备与主机相连的信号并生成会话请求中断（USBFS_GINTSTS 中的 VBUSVINT 位），指示进入供电状态。

供电状态下，USBFS 期望收到来自主机的复位信号。其它 USB 操作则无法执行。收到复位信号后，立即生成检测到复位中断（USBFS_GINTST 中的 USBRST）。复位信号结束后，将生成枚举完成中断（USBFS_GINTSTS 中的 ENUMDNE 位），USBFS 随即进入默认状态。

29.5.2.3 Device Default Status

默认状态下，USBFS 期望从主机收到 SET_ADDRESS 命令。其它 USB 操作则无法执行。当 USB 上解码出有效 SET_ADDRESS 命令时，应用程序会将相应的地址值写入设备配置寄存器中的设备地址字段（USBFS_DCFG 中的 DAD 位）。USBF 随即进入地址状态，并准备好以所配置的 USB 地址对主机事务进行应答。

29.5.2.4 Device Suspended State

USBFS 设备持续监视 USB 活动。在 USB 空闲时间达到 3ms 后，将发出早期挂起中断（USBFS_GINTSTS 中 ESUSP 位），并在 3ms 后由挂起中断（USBFS_GINTSTS 中的 USBSUSP 位）确认设备进入挂起状态。然后，设备状态寄存器中的设备挂起位（USBFS_DSTS 中的 SUSPSTS 位）自动置 1，USBFS 随即进入挂起状态。

可通过设备本身退出挂起状态。这种情况下，应用程序会将设备控制寄存器中的远程唤醒信号位（USBFS_DCTL 中的 RWUSIG 位）置 1，并在 1ms 到 15ms 后将其清零。

但若设备检测到主机发出的恢复信号，将生成恢复中断（USBFS_GINTSTS 中的 WKUPINT 位），设备挂起位自动清零。

29.5.2.5 Device Soft Disconnect

供电状态可借助软断开功能通过软件退出。将设备控制寄存器中的软断开位（USBFS_DCTL 中的 SDIS 位）置 1 即可移除 DP 上拉电阻，此时尽管没有从主机端口实际拔出 USB 电缆，但主机端仍会发生设备断开检测中断。

29.5.2.6 Device Endpoints

端点类别

USBFS 模块实现了以下 USB 端点：

- 控制端点 0：
 - 双向且仅处理控制消息
 - 使用一组单独的寄存器来处理 IN 和 OUT 事务
 - 专用控制（USBFS_DIEPCTL0/USBFS_DOEPCTL0）寄存器、传输配置（USBFS_DIEPTSIZ0/USBFS_DOEPTSIZ0）寄存器和状态中断（USBFS_DIEPINT0/USBFS_DOEPINT0）寄存器。控制和传输配置寄存器中可用的位组与其它端点中稍有不同

- 4 个 IN 端点 ($n = 1,3,5,7$)
 - 每个端点都可配置为支持同步传输、批量传输或中断传输类型
 - 每个端点都有专用控制 (*USBFS_DIEPCTLn*) 寄存器、传输配置 (*USBFS_DOEPTSIzn*) 寄存器和状态中断(*USBFS_DIEPINTn*) 寄存器
 - 设备 IN 端点通用中断屏蔽寄存器 (*USBFS_DOEPMSK*) 可用于使能/禁止所有 IN 端点(包括 EP0)上的同一类端点中断源
 - 支持未完成的同步 IN 传输中断 (*USBFS_GINTSTS* 中的 *IISOIXFR* 位)，该中断将在当前帧中至少有一个同步 IN 端点上的传输未完成时触发。该中断和周期性帧中断 (*USBFS_GINTSTS/EOPF*) 一起触发
- 4 个 OUT 端点($n = 2,4,6,8$)
 - 每个端点都可配置为支持同步传输、批量传输或中断传输类型
 - 每个端点都有专用控制 (*USBFS_DOEPCCTLn*) 寄存器、传输配置(*USBFS_DOEPTSIzn*) 寄存器和状态中断 (*USBFS_DOEPINTn*) 寄存器
 - 设备 OUT 端点通用中断屏蔽寄存器 (*USBFS_DOEPMSK*) 可用于使能/禁止所有 OUT 端点(包括 EP0) 上的同一类端点中断源
 - 支持未完成的同步 OUT 传输中断 (*USBFS_GINTSTS* 中的 *INCOMPISOOUT* 位)，该中断将在当前帧中至少有一个同步 OUT 端点上的传输未完成时触发。该中断和周期性帧中断 (*USBFS_GINTSTS/EOPF*) 一起触发

端点控制

应用程序可通过设备端点 n IN/OUT 控制寄存器 (DIEPCTLn/DOEPCCTLn) 对端点采取以下控制：

- 端点使能/禁止
- 在当前配置下激活端点
- 设置 USB 传输类型 (同步、批量和中断)
- 设置支持的数据包大小
- 设置与 IN 端点相关的 Tx-FIFO 编号
- 设置希望收到的或发送时要使用到的 data0/data1 PID (仅限批量/中断传输)
- 设置接收或发送事务时所对应的奇/偶帧 (仅限同步传输)
- 可以设置 NAK 位，从而不论此时 FIFO 的状态如何，都对主机的请求回复 NAK
- 可以设置 STALL 位，使得主机对该端点的令牌都被硬件回复 STALL
- 可以将 OUT 端点设置为侦听模式，即对接收到的数据不进行 CRC 检查

端点传输

设备端点 n 传输尺寸寄存器 (DIEPTSIZn/DOEPTSIZn) 允许应用程序对传输尺寸参数进行编程并读取传输状态。必须在端点控制寄存器中的端点使能位置 1 之前完成对此寄存器的设置。使能端点后，这些字段立即变为只读状态，同时 USBFS 模块根据当前传输状态对这些字段进行更新。

可对以下传输参数进行编程：

- 以字节为单位的传输大小
- 构成整个传输的数据包个数

端点状态/状态

设备端点 n 中断寄存器 (DIEPINTn/DOEPINTn) 指示端点在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的 OUT 端点中断位或 IN 端点中断位（分别为 USBFS_GINTSTS 中 OEPINT 位或 USBFS_GINTSTS 中的 IEPINT 位）置 1 时，应用程序必须读取这些寄存器以获得详细信息。在应用程序读取这些寄存器之前，必须先读取设备全体端点中断(USBFS_DAINT) 寄存器，以获取设备端点 n 中断寄存器的端点编号。应用程序必须将此寄存器中的相应位清零，才能将 DAINT 和 GINTSTS 寄存器中的相应位清零。

- 模块提供以下状态检查和中断产生功能：
- 传输完成中断，指示应用程序 AHB 和 USB 端均已完成数据传输
- Setup 阶段已完成（仅针对控制传输类型的 OUT 端点）
- 相关的发送 FIFO 为半空或全空状态（IN 端点）
- NAK 应答已发送到主机（仅针对同步传输的 IN 端点）
- TxFIFO 为空时接收到 IN 令牌（仅针对批量和中断传输类型的 IN 端点）
- 尚未使能端点时接收到 OUT 令牌
- 检测到 babble 错误
- 应用程序关闭端点生效
- 应用程序对端点设置 NAK 生效（仅针对同步传输类型的 IN 端点）
- 接收到 3 个以上连续 setup 数据包（仅针对控制类型的 OUT 端点）
- 检测到超时状况（仅针对控制传输类型的 IN 端点）
- 同步传输类型的数据包未产生中断而丢失

29.5.3 USBFS Power Consumption Control

不使用 USBFS 模块时，可以通过系统控制器模块停止 USBFS 模块的 HCLK 和 PHY 时钟，从而降低功耗。

使用 USB 模块，但设备 USB 会话未开始或设备未连接时，可以在 USB 挂起状态下使用功率降低技术。

- 停止 PHY 时钟 (USBFS_GCCTL 中的 STPPCLK 位)

将时钟门控控制寄存器中的停止 PHY 时钟位置 1 时，USBFS 全速模块的大多数 48 MHz 内部时钟域均由时钟门控关闭。即使应用程序仍提供时钟输入，也会节省掉模块由于时钟信号翻转带来的动态功耗还会关掉收发器的大部分单元，只有负责检测异步恢复事件或远程唤醒事件的部分还保持工作状态。

- HCLK 门控 (USBFS_GCCTL 中的 GATEHCLK 位)

将时钟门控控制寄存器中的 GATEHCLK 位置 1 时，USBFS 模块内部的大多数系统时钟域均由时钟门控关闭。只有寄存器读取和写入接口保持活动状态。即使应用程序仍提供时钟输入，也会节省掉模块由于时钟信号翻转带来的动态功耗。

为了节省动态功耗，只在 USB 数据 FIFO 被 USBFS 模块访问时为其提供时钟。

29.5.4 USBFS Data FIFO

USBFS 系统具有 1.25KB 专用 RAM，采用高效的 FIFO 控制机制。USBFS 模块中的数据包 FIFO 控制器模块将 RAM 空间划分为一个非周期 TxFIFO，一个周期 TxFIFO（USB 传输前，应用程序将数据压入其中进行短暂停储）和一个 Rx FIFO（从 USB 接收到的数据被应用程序读取之前，在其中进行短暂停储）。

其中非周期 TxFIFO 由所有的非周期 IN 端点共享，Rx FIFO 由所有的 OUT 端点共享。FIFO 的大小均由软件配置，以更好地满足应用要求。

29.5.5 USBFS Device FIFO Architecture

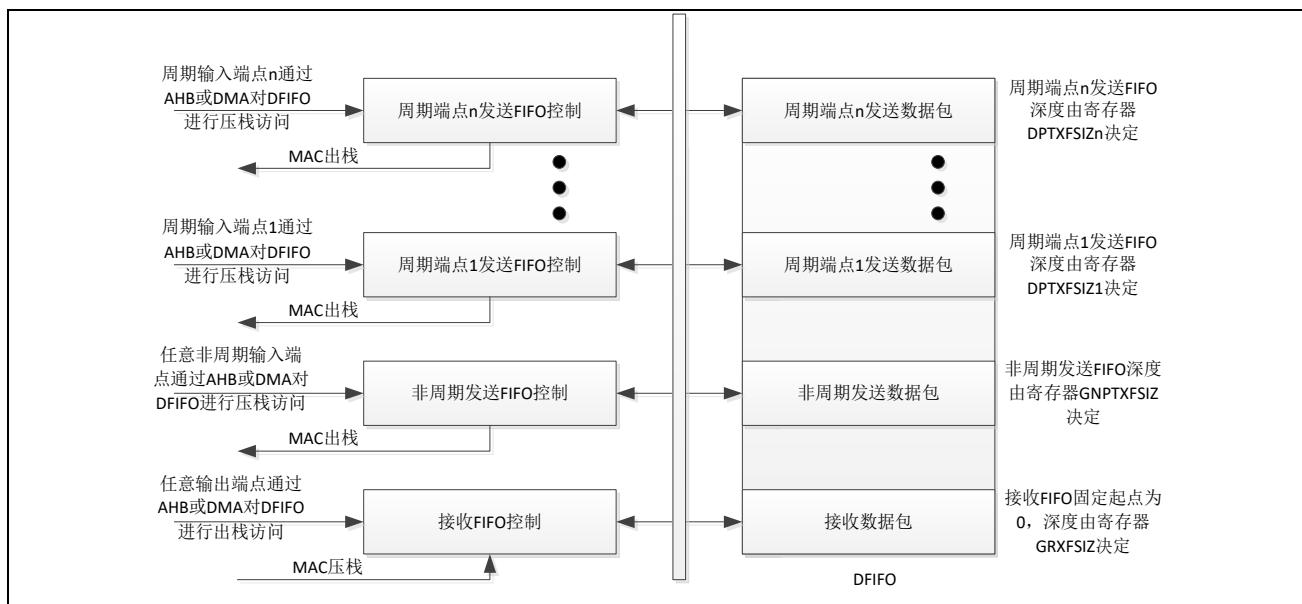


图 29-3 USBFS 设备模式下 FIFO 架构示意图

29.5.5.1 Device RxFIFO

USBFS 设备使用单个接收 FIFO 接收发送到所有 OUT 端点的数据。只要 RxFIFO 中有空余空间，收到的数据包就挨个填入 RxFIFO。除了有效数据外，接收到的数据包状态（包含 OUT 端点目标编号、字节数、数据 PID 和对所接收数据的验证）也由模块进行存储。没有可用空间时，设备会回复主机事务 NAK 应答并在被寻址的端点上触发中断。接收 FIFO 的大小在接收 FIFO 大小寄存器 (GRXFSIZ) 中配置。

单个接收 FIFO 架构使得 USB 设备更高效地填充接收 RAM 缓冲区：

- 所有 OUT 端点共享同一个 RAM 缓冲区（共享 FIFO）
- 对于任意主机序列 OUT 令牌，USBFS 模块可将接收 FIFO 填充至限值

只要至少有一个数据包在 RxFIFO 中可供读取，应用程序就会一直接收 RxFIFO 非空中断 (USBFS_GINTSTS 中的 RXFNE 位)。应用程序从接收状态读取和出栈寄存器(GRXSTSP)中读取数据包信息，最后通过读取与端点相关的出栈地址从接收 FIFO 读出相应数据。

29.5.5.2 Device TxFIFO

所有的非周期 IN 端点共享一个 TxFIFO。周期 IN 端点独享一个 TxFIFO。应用程序通过非周期发送 FIFO 大小寄存器 (USBFS_GNPTXFSIZ) 为非周期 IN 端点配置 FIFO 大小；通过周期 IN 端点发送 FIFO 大小寄存器 (DPTXFSIZn) 为周期 IN 端点 n 配置 FIFO 大小。

29.5.6 USBFS FIFO RAM Allocation

接收 FIFO RAM 分配

应用程序应为 SETUP 数据包分配 RAM：接收 FIFO 中必须保留 11 个位置以在控制端点上接收 SETUP 数据包。USBFS 模块不会向这些为 SETUP 数据包保留的位置写入任何其它数据。将会为全局 OUT NAK 分配一个位置。状态信息随各个接收数据包写入 FIFO。因此，必须至少为接收数据包分配（最大数据包大小/4）+1 的空间。如果使能了多个同步端点，则为接收连续数据包分配的空间必须至少为（最大数据包大小/4）的两倍+1。通常，推荐的空间为（最大数据包/4 + 1）的两倍，这样当上一个数据包向 CPU 传送时，USB 可同时接收后续的数据包。

传输完成状态信息和该端点收到的最后一个数据包会一起被推入 FIFO。通常情况下，推荐为每个 OUT 端点分配一个位置。

发送 FIFO RAM 分配

各个 IN 端点发送 FIFO 所需的最小 RAM 空间为该特定 IN 端点的最大数据包大小。

29.5.7 USBFS System Performance

凭借大容量 RAM 缓冲区、高度可配置的 FIFO 大小、通过 AHB 压栈/出栈寄存器进行 32 位 FIFO 快速访问，尤其是高级 FIFO 控制机制可获得最佳 USB 和系统性能。实际上，无论当前 USB 序列如何，USBFS 均可通过该机制高效填充可用的 RAM 空间。借助这些特性：

- 应用程序有足够的裕量来计算并校正 CPU 的负载，从而优化 CPU 带宽利用率：
 - 应用程序可先积累大量发送数据，再通过发送出去
 - 可带来足够的时间裕量，以从接收读取数据
- USB 模块能够保持全速工作状态，也就是提供最大的全速带宽（尽量多的硬件自动运行，尽量少的软件参与）
 - USB 模块可提前积累大量发送数据供其支配，从而可对数据发送进行自主管理
 - 接收缓冲区中有大量空白空间，可通过中的数据自动填满

由于 USBFS 模块能够高效填充 1.25KB RAM 缓冲区且 1.25KB 发送/接收数据足以满足一个全速帧所能容纳的数据量，因此 USB 系统在一帧之内可以无需应用程序干预达到最大 USB 带宽。

29.5.8 USBFS Interrupts and Events

全局中断是软件需要处理的主要中断，全局中断的标志位可在 USBFS_GINTSTS 寄存器读取。

表 29-2 USBFS 中断事件表

中断标志	描述
WKUPINT	远程唤醒中断
DATAFSUSP	数据获取挂起
INCOMPISOOUT	未完成OUT同步传输
IISOIXFR	未完成IN同步传输
OEPINT	OUT端点中断
IEPINT	IN端点中断
EOPF	周期性帧结束中断
ISOODRP	丢弃同步OUT数据包中断
ENUMDNE	枚举完成
USBRST	USB复位中断
USBSUSP	USB挂起中断
ESUSP	早期挂起中断
GONAKEFF	全局OUT NAK有效中断
GINAKEFF	全局非周期性IN NAK有效中断
RXFNE	RxFIFO非空中断
SOF	帧起始中断
MMIS	模式不匹配中断

29.6 USBFS Programming Model

29.6.1 USBFS module initialization

USBFS 模块工作需要两个时钟，系统时钟和 PHY 时钟，其中 PHY 时钟频率必须配置为 48Mhz。PHY 时钟可以选择 PLL 时钟或者内部高速 RC48M 时钟。

1. 配置 PLL 时钟或者 RC48M 时钟，等待时钟输出稳定(操作流程请参考系统控制器(SYSCtrl)章节)。
2. 配置系统控制器寄存器 SYSCtrl1.USB48MSEL，置 0 选择 RC48M 时钟，置 1 选择 PLL 时钟。
3. 配置系统控制器寄存器 PERI_CLKEN1.USB 置 1 使能 USBFS 模块时钟。

29.6.2 USBFS Device Initialization

应用程序必须执行下列步骤来将模块作为设备进行初始化。

1. Slave 模式下, USBFS_GINTMSK.NPTxFEmpMsk 和 USBFS_GINTMSK.RxFLvIMsk 置 1, DMA 模式下, USBFS_GINTMSK.NPTxFEmpMsk 和 USBFS_GINTMSK.RxFLvIMsk 置 0
2. 在 USBFS_DCFG 寄存器中编程以下字段:
 - 设备速度 (DevSpd)
 - 非零长度状态输出握手信号 (NZStsOUTHSht)
 - 周期帧间隔 (PerFrInt)
3. 清除 USBFS_DCTL.SftDiscon 位, 让控制器执行连接到主机。
4. 编程 USBFS_GINTMSK 寄存器以取消屏蔽以下中断:
 - USB复位 (USBRstMsk)
 - 枚举完成(EnumDoneMsk)
 - 早期挂起(ErlySuspMsk)
 - USB挂起(USBSuspMsk)
 - SOF(SofMsk)
5. 等待 USBFS_GINTSTS.USBRst 中断。这表示已在 USB 上检测到复位信号, 复位过程自接收到此中断后约持续 10ms。应用程序必须执行章节 USB 复位时的端点初始化中所列的步骤。
6. 等待 USBFS_GINTSTS.EnumDone 中断。此中断指示 USB 上复位过程结束。接收到此中断时, 应用程序必须读取 USBFS_DSTS 寄存器以确定枚举速度并执行 USB 枚举完成时的端点初始化中所列的步骤。

此时, 设备已准备好接受 SOF 数据包并在控制端点 0 上执行控制传输。

29.6.3 USBFS device soft disconnect operation

应用程序可以置位寄存器

执行一个软断开操作。当设备处于数据传输状态时，

执行以下步骤：

置位寄存器

置位寄存器

等待寄存器

位被控制器清零

重新执行

步骤

29.6.4 Endpoint Initialization During USB Reset

把所有输出端点 位置

(对于所有输出端点)

取消对以下中断位的屏蔽

(控制 输入端点)

(控制 输出端点)

弃

为每个 设置数据

编程 寄存器，以能够接收控制传输的输出数据和 数据。该寄存器值必须至少等于控制端点 的 个最大数据包大小 个字（用于控制输出数据包的状态） 个字（用于 数据包）。

编程 寄存器，以能够发送控制输入数据。该寄存器值必须至少等于控制端点 的 个最大数据包大小。

对端点相关寄存器中的以下字段进行编程，以使控制输出端点 接收 数据包。

- USBFS_DOEPTSI0.SUPCnt=3 (接收最多 个连续的 数据包)

- DMA 模式下， 寄存器指定内存地址存放收到的 数据包

此时，接收 数据包所需的所有初始化工作便已完成。

29.6.5 Endpoint Initialization When USB Enumeration is Complete

1. 在枚举完成中断 (USBFS_GINTSTS.EnumDone) 发生后, 读取寄存器 USBFS_DSTS.EnumSpd 以确定设备的枚举速度。
2. 对 USBFS_DIEPCTL0 中的 MPS 字段进行编程以设置最大数据包大小。该步骤配置控制端点 0。控制端点的最大数据包大小取决于枚举速度。
3. DMA 模式下, 编程 USBFS_DOEPCTL0 寄存器使能控制输出端点 0。
`USBFS_DOEPCTL0.EPEna = 1`
此时, 设备已准备好接收 SOF 数据包并配置为在控制端点 0 执行控制传输。

29.6.6 Endpoint Initialization When a SetAddress Command is Received

本节介绍了应用程序在 SETUP 数据包中接收到 SetAddress 命令时必须执行的操作。

1. 使用在 SetAddress 命令中接收到的设备地址来对 USBFS_DCFG 寄存器进行编程。
2. 对模块进行编程以发出状态阶段的输入数据包。

29.6.7 Endpoint Initialization Upon Receiving SetConfiguration/SetInterface Command

本节介绍了应用程序在 SETUP 包中接收 SetConfiguration 或 SetInterface 命令时必须执行的操作。

1. 接收到 SetConfiguration 命令时, 应用程序必须对端点寄存器进行编程, 以使用新配置中有效端点的特性来配置这些端点寄存器。
2. 接收到 SetInterface 命令时, 应用程序必须对命令指定的端点寄存器进行编程。
3. 在先前配置或其它设置中有效的端点在新的配置或其它设置中无效。必须不激活这些无效端点。
4. 使用 USBFS_DAINTMSK 寄存器使能有效端点的中断, 屏蔽无效端点的中断。
5. 为每个 FIFO 设置数据 FIFO RAM。
6. 配置完所有必需的端点后, 应用程序必须对模块进行编程以发送状态阶段的输入数据包。
此时, 设备模块已可以接收和发送任何类型的数据包

29.6.8 Endpoint Activation

本节介绍激活设备端点或者将现有设备端点配置为新类型所需的步骤。

1. 在 USBFS_DIEPCTL_n 寄存器（对于输入端点）或 USBFS_DOEPCTL_n 寄存器（对于输出端点）的以下字段中，对所需端点的特性进行编程。
 - 最大数据包大小（MPS）
 - USB 活动端点位（USBActEP）置 1
 - 端点初始数据同步位（DPID）（对于中断和批量端点）
 - 端点类型（EPType）
 - Tx FIFO 编号(TxFNum)
2. 激活端点后，模块便开始解码发送到该端点的令牌，并在收到的令牌有效的情况下回复有效握手信号。

29.6.9 Endpoint Deactivation

本节介绍不激活现有端点所需的步骤。

1. 在要不激活的端点中，将 USBFS_DIEPCTL_n 寄存器（对于输入端点）或 USBFS_DOEPCTL_n 寄存器（对于输出端点）中的 USB 活动端点位（USBActEP）清零。
2. 停用端点后，模块便会忽略发送到该端点的令牌，从而导致 USB 超时。

29.6.10 Disable Out Endpoint

应用程序必须按照以下步骤禁止一个已经使能的输出端点。

1. USBFS_DCTL.SGOUTNak=1, 进入全局输出端点 NAK 模式
2. 等待中断 USBFS_GINTSTS.GOUTNakEff
3. USBFS_DOEPCTLn.EPDDisable=1, USBFS_DOEPCTLn.SNAK=1
4. 等待中断 USBFS_DOEPINTx.EPDisbld,
当中断发生时, 控制器会清除以下寄存器
USBFS_DOEPCTLx.EPDis = 0
USBFS_DOEPCTLn.EPEna = 0
5. 程序必须退出全局输出端点 NAK 模式
USBFS_DCTL.CGOUTNak = 1
USBFS_DCTL.SGOUTNak=0

29.6.11 Stall Asynchronous Out Endpoint

应用程序必须按照以下步骤暂停非同步输出端点。

1. USBFS_DCTL.SGOUTNak=1, 进入全局输出 NAK 模式
2. 等待中断 USBFS_GINTSTS.GOUTNakEff
3. USBFS_DOEPCTLn.EPDDisable=1, USBFS_DOEPCTLn.Stall=1
4. 等待中断 USBFS_DOEPINTx.EPDisbld,
当中断发生时, 控制器会清除以下寄存器
USBFS_DOEPCTLx.EPDis = 0
USBFS_DOEPCTLn.EPEna = 0
5. 程序必须退出全局输出端点 NAK 模式
USBFS_DCTL.CGOUTNak = 1
USBFS_DCTL.SGOUTNak=0

29.6.12 Global Out Endpoint NAK Mode

程序必须进入全局输出端点 NAK 模式以停止接收接收 FIFO 里的任何数据。

1. USBFS_DCTL.SGOUTNak=1，进入全局输出 NAK 模式。
2. 等待中断 USBFS_GINTSTS.GOUTNakEff，中断发生表示控制器已停止接收除 SETUP 数据包以外的数据。
3. 程序可以写寄存器屏蔽这个中断 USBFS_GINTMSK.GOUTNakEffMsk=0。
4. 如果程序要退出全局输出 NAK 模式，执行以下操作同时可以清除中断。

```
USBFS_GINTSTS.GOUTNakEff  
USBFS_DCTL.CGOUTNak = 1  
USBFS_DCTL.SGOUTNak=0
```

29.6.13 Out Endpoint Transfer Stop

1. 使能所有的输出端点 USBFS_DOEPCTLn.EPEna = 1。
2. 等待 USBFS_GRSTCTL.AHBIdle 置位，表示控制器的 AHB 总线控制器空闲。
3. USBFS_GRSTCTL.RxFFlsh =1
4. 等待 USBFS_GRSTCTL.RxFFlsh 清零,表示接收 FIFO 成功清空。
5. 执行章节禁止(Disable)输出端点。

29.6.14 Configuring an Interrupt Input Endpoint as a Periodic or Aperiodic Endpoint (Shared FIFO Mode)

共享 FIFO 模式下，可以通过设置 USBFS_DIEPCTLn.TxFNum 值，把中断输入端点配置成周期输入端点或者非周期输入端点。

当配置成非周期输入端点时，程序必须遵照非周期输入端点的操作流程。

当配置成周期输入端点时，程序必须遵照周期输入端点的操作流程。

29.6.15 Global Aperiodic In Endpoint NAK Mode (Shared FIFO Mode)

1. 设置 USBFS_DCTL.SGNPInNak=1，通知控制器停止向非周期输入端点发送数据，进入全局非周期输入端点 NAK 模式。
2. 等待中断标志位 USBFS_GINTSTS.GINNakEff 置位，该标志位置位表示控制器已停止向非周期输入端点发送数据，但是在置位前控制器依然可以向非周期输入端点发送数据。
3. 中断标志位 USBFS_GINTSTS.GINNakEff 可以由
USBFS_GINTMSK.GINTNakEffMsk 屏蔽，
设置 USBFS_GINTMSK.GINTNakEffMsk=0，
屏蔽中断标志位 USBFS_GINTSTS.GINNakEff
设置 USBFS_GINTMSK.GINTNakEffMsk=1，
取消屏蔽中断标志位 USBFS_GINTSTS.GINNakEff
4. 编程 USBFS_DCTL.CGNPInNak=1
USBFS_DCTL.SGNPInNak=0，
退出全局非周期输入端点 NAK 模式，
同时清除中断标志位 USBFS_GINTSTS.GINNakEff。

29.6.16 In Endpoint NAK Occurs

1. 停止向一个特别的输入端点发送数据，程序必须置位 NAK 位
USBFS_DIEPCTLn.SNAK=1，进入输入端点 NAK 模式。
2. 等待中断标志位 USBFS_DIEPINTn.NAKIntrpt 置位，表示控制器已经停止向这个端点发送数据。
3. USBFS_DIEPMSK.NAKMsk = 0，可以屏蔽中断 USBFS_DIEPINTn.NAKIntrpt，
USBFS_DIEPMSK.NAKMsk = 1，取消屏蔽中断 USBFS_DIEPINTn.NAKIntrpt。
4. 如果要退出输入端点 NAK 模式
USBFS_DIEPCTLn.CNAK=1

29.6.17 Disabling Non-Periodic Input Endpoints (Shared FIFO Mode)

禁止端点操作必须在该端点被使能之后。禁止一个非周期输入端点，程序必须禁止所有非周期端点。

1. 设置 USBFS_DCTL.SGNPInNak=1，进入全局非周期输入端点 NAK 模式。
2. 等待中断标志位 USBFS_GINTSTS.GINNakEff 置位。
3. 设置 USBFS_DIEPCTLn.EPDis=1, USBFS_DIEPCTLn.SNAK=1, 一个接一个，直到所有非周期端点被禁止。
4. 等待中断标志位 USBFS_DIEPINTn.EPDisbld 置位，该标志位置位表示控制器已经禁止对应的端点，同时清除寄存器位 USBFS_DIEPCTLn.EPEna 和 USBFS_DIEPCTLn.EPDis。依次判断所有的非周期端点已经被禁止。
5. 程序必须读出所有被禁止的非周期输入端点寄存器 USBFS_DIEPTSIzn 的值，计算出每个端点已经传输的数据量。
6. 程序必须清除非周期发送 FIFO，操作如下：
USBFS_GRSTCTL.TxFNum=0
USBFS_GRSTCTL.TxFFlsh=1
等待控制器清除 USBFS_GRSTCTL.TxFFlsh， 表示非周期发送 FIFO 清除已结束。
7. 编程 USBFS_DCTL.CGNPInNak=1 USBFS_DCTL.SGNPInNak=0，退出全局非周期输入端点 NAK 模式。

29.6.18 Disabling the Periodic Input Endpoint (Shared FIFO Mode)

禁止端点操作必须在该端点被使能之后。

1. 设置 USBFS_DIEPCTLn.SNAK=1，
等待中断标志位 USBFS_DIEPINTn.INEPNakEff 置位，
该标志位置位表示控制器已停止向该端点发送数据。
2. 设置 USBFS_DIEPCTLn.EPDis=1，
等待中断标志位 USBFS_DIEPINTn.EPDisbld 置位，
该标志位置位表示控制器已经禁止对应的端点，同时清除寄存器位
USBFS_DIEPCTLn.EPEna 和 USBFS_DIEPCTLn.EPDis。
3. 程序必须读出被禁止的周期输入端点寄存器 USBFS_DIEPTSIzn 的值，计算出该端点已经传输的数据量。
4. 程序必须清除非周期发送 FIFO，操作如下：
USBFS_GRSTCTL.TxFNum= Periodic FIFO Number
USBFS_GRSTCTL.TxFFlsh=1
等待控制器清除 USBFS_GRSTCTL.TxFFlsh， 表示周期发送 FIFO 清除已结束。

29.6.19 Non-Periodic Input Data Transfer Timeout (Shared FIFO Mode)

1. 当某个非周期输入端点在传输数据时，

中断标志位 USBFS_DIEPCTLn.TimeOut 置位，该标志位置位表示该端点数据传输超时。

2. 按照禁止非周期输入端点（共享 FIFO 模式）步骤执行操作。
3. 重新使能端点并传送超时的数据包。

29.6.20 Asynchronous Input Endpoint Pause (Shared FIFO Mode)

1. 设置 USBFS_DCTL.SGNPInNak=1，进入全局非周期输入端点 NAK 模式。
2. 等待中断标志位 USBFS_GINTSTS.GINNakEff 置位。
3. 设置 USBFS_DIEPCTLn.EPDIs=1, USBFS_DIEPCTLn.Stall=1, 一个接一个，直到所有非周期端点被禁止。
4. 等待中断标志位 USBFS_DIEPINTn.EPDIsbld 置位，该标志位置位表示控制器已经禁止对应的端点，同时清除寄存器位 USBFS_DIEPCTLn.EPEna 和 USBFS_DIEPCTLn.EPDIs。依次判断所有的非周期端点已经被禁止。
5. 程序必须读出所有被禁止的非周期输入端点寄存器 USBFS_DIEPTSIzn 的值，计算出每个端点已经传输的数据量。
6. 程序必须清除非周期发送 FIFO，操作如下：
USBFS_GRSTCTL.TxFNum=0
USBFS_GRSTCTL.TxFFlsh=1
等待控制器清除 USBFS_GRSTCTL.TxFFlsh，表示非周期发送 FIFO 清除已结束。
7. 设置 USBFS_DCTL.SGNPInNak=0，退出全局非周期输入端点 NAK 模式。
8. 重新使能其他不需要暂停的端点
9. 如果要退出暂停模式，设置 USBFS_DIEPCTLn.Stall=0。

29.6.21 Input Endpoint Transfer Stop (Shared FIFO Mode)

1. 设置 USBFS_DIEPCTLn.EPDIs=1。
2. 等待中断标志位 USBFS_DIEPINTn.EPDIsbld 置位，该标志位置位表示控制器已经禁止对应的端点，同时清除寄存器位 DIEPCTLn.EPEna 和 DIEPCTLn.EPDIs。
3. 程序必须清除非周期发送 FIFO，操作如下：
USBFS_GRSTCTL.TxFNum=0
USBFS_GRSTCTL.TxFFlsh=1
等待控制器清除 USBFS_GRSTCTL.TxFFlsh， 表示非周期发送 FIFO 清除已结束。

29.6.22 Endpoint Mismatch (Shared FIFO Mode)

共享 FIFO 模式下，所有的非周期输入端点共用一个发送 FIFO, 程序必须预测主机访问非周期输入端点的顺序，当顺序预测错误时，就会发生端点不匹配情况，中断标志位 USBFS_DIEPINTn.INTknEPMis 置位。

29.6.23 Control Transfer

控制传输有三种类型：

- 控制写传输
- 控制读传输
- 两阶段控制传输

每一个控制传输包含两个或者三个阶段：

- Setup 阶段
- 数据阶段
- 状态阶段

29.6.23.1 Controlling Write Transfer Programming Flow

控制写传输 Setup 阶段

1. 在系统内存里创建一个缓存用来保存 Setup 数据包
2. 编程寄存器 USBFS_DOEPDMA0 来配置缓存列表的基地址
3. 编程寄存器 USBFS_DOEPTSI0 的 XferSize,PktCnt 和 SUPCnt 字段
4. 编程寄存器 USBFS_DOEPCTL0.MPS 来配置控制端点最大数据包大小，编程寄存器 USBFS_DOEPCTL0.EPEna 置 1 使能控制输出端点
5. 等待输出端点中断
6. 当控制器收到 Setup 令牌包时，会置位 USBFS_DOEPINT0 中断
 - a) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=0，程序必须读寄存器 USBFS_DOEPINT0 的 StupPktRcvd 位。如果 StupPktRcvd=1，表示 Setup 令牌包已经收到。程序必须解码主机发送的控制传输类型。然后程序必须检查 USBFS_DOEPINT0.StsPhaseRcvd 是否置 1，如果置 1，表示进入状态阶段，如果没有置 1，表示进入数据阶段。
 - b) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=1，程序必须读 USBFS_DOEPINT0 寄存器的 StupPktRcvd 位。如果 StupPktRcvd=1，表示 Setup 令牌包已经收到。程序必须解码主机发送的控制传输类型。然后程序必须进入数据阶段。USBFS_DOEPINT0.SetUp=1 表示主机在控制写传输的数据阶段发送了输出令牌包。

控制写传输数据阶段

1. 创建 n 个缓存接收 `wlength` 字节数据确保每个缓存包含一个数据包 [n=wlength/MPS+wlengthMOD(MPS)]
2. 编程寄存器 USBFS_DOEPDMA0 来配置描述符列表的基址
3. 编程寄存器 USBFS_DOEPTSI0 的 XferSize,PktCnt 和 SUPCnt 字段
4. 编程寄存器 USBFS_DOEPCTL0.MPS 来配置控制端点最大数据包大小，编程寄存器 USBFS_DOEPCTL0.EPEna 置 1 使能控制输出端点
5. 等待输出端点中断
6. 根据收到令牌的类型，控制器会产生下面的中断
 - a) 如果 USBFS_DOEPINT0.XferComp=0 并且 USBFS_DOEPINT0.SetUp=1, 控制器对主机发送的第一个输出令牌回复 NAK。然后编程 USBFS_DOEPCTL0.CNAK=1 清除 NAK，然后跳转到第 4 步。
 - b) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=0，读寄存器 USBFS_DOEPINT0 的 StupPktRcvd 位如果 StupPktRcvd=0，表示输出令牌已收到，如果这个令牌不是最后一个，就跳转到第 4 步接收下一个令牌。如果这是最后一个令牌，配置 USBFS_DOEPCTL0.Stall=1, 这样控制器会暂停接收主机发送的输出令牌包。数据阶段已完成，程序将跳转到控制写传输的状态阶段。如果 StupPktRcvd=1，表示一个新的控制传输 Setup 令牌已收到。程序必须解码主机发送的控制传输类型。
 - c) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.StsPhseRcvd=1，中断标志位 USBFS_DOEPINT0.XferComp 表示控制器在数据阶段收到了输出令牌而中断标志位 USBFS_DOEPINT0.StsPhseRcvd 表示状态阶段开始。在处理完最后一个输出令牌后，程序进入控制写传输的状态阶段。

控制写传输状态阶段

1. 在系统内存里创建缓存列表包含一个输入缓存
2. 编程寄存器 USBFS_DIEPDMA0 来配置缓存列表的基地址
3. 编程寄存器 USBFS_DIEPCTL0 的 XferSize,PktCnt 字段
4. 编程寄存器 USBFS_DIEPCTL0.MPS 来配置控制端点最大数据包大小，编程寄存器 USBFS_DIEPCTL0.EPEna 置 1 使能控制输入端点
5. 在系统内存里创建缓存列表包含一个输出缓存
6. 编程寄存器 USBFS_DOEPDMA0 来配置缓存列表的基地址
7. 编程寄存器 USBFS_DOEPTSI0 的 XferSize,PktCnt 和 SUPCnT 字段
8. 编程寄存器 USBFS_DOEPTSI0.MPS 来配置控制端点最大数据包大小，编程寄存器 USBFS_DOEPTSI0.EPEna 置 1 使能控制输出端点
9. 等待控制端点输入和输出中断同时发生
10. 根据收到令牌的类型，控制器会产生下面的中断

如果 USBFS_DIEPINT0.XferComp=1，只有输入端点产生中断，表示 DMA 已经处理输入描述符的零长度数据包并压栈到对应的发送 FIFO。当前控制传输的状态阶段已完成，程序可以处理下一个控制传输。

如果 USBFS_DOEPINT0.StsPhseRcvd=1，只有输出端点产生中断，编程寄存器 USBFS_DIEPCTL0.CNAK=1，接受输入状态令牌，并跳转到第 8 步。

如果 USBFS_DOEPINT0.XferComp=1，只有输出端点产生中断，表示主机提前发送一个新的 Setup 令牌。

如果 USBFS_DOEPINT0.SetUp=0 并且 USBFS_DOEPINT0.StupPktRcvd=1，表示新的控制传输 Setup 令牌已经收到。程序必须解码主机发送的控制传输类型。然后程序必须进入下一个阶段。

如果 USBFS_DOEPINT0.SetUp=1 且 USBFS_DOEPINT0.StupPktRcvd=1，表示新的控制传输 Setup 令牌已经收到。

DOEPINT0.SetUp=1 表示主机可能发送以下令牌包：

- 控制写传输数据阶段输出令牌包
- 控制读传输数据阶段输入令牌包
- 两阶段控制传输状态阶段输入令牌包

29.6.23.2 Controlling Read Transfer Programming Flow

控制读传输 Setup 阶段

1. 在系统内存里创建一个缓存列表包含一个输出缓存
2. 编程寄存器 USBFS_DOEPDMA0 来配置缓存列表的基地址
3. 编程寄存器 USBFS_DOEPTSI20 的 XferSize,PktCnt 和 SUPCnt 字段
4. 编程寄存器 USBFS_DOEPCTL0.MPS 来配置控制端点最大数据包大小，编程寄存器 USBFS_DOEPCTL0.EPEna 置 1 使能控制输出端点
5. 等待输出端点中断
6. 当控制器收到 Setup 数据包时，会置位 USBFS_DOEPINT0 中断
 - a) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=0，程序必须读 USBFS_DOEPINT0 寄存器的 StupPktRcvd 位。如果 StupPktRcvd=1，表示 Setup 令牌已经收到。程序必须解码主机发送的控制传输类型。然后程序必须进入控制传输的数据阶段。
 - b) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=1，程序必须读 USBFS_DOEPINT0 寄存器的 StupPktRcvd 位。如果 StupPktRcvd=1，表示 Setup 令牌已经收到。USBFS_DOEPINT0.SetUp=1 表示主机在控制读传输的数据阶段发送了输入令牌包。然后程序必须进入下一个阶段。

控制读传输数据和状态阶段

1. 创建 n 个缓存接收 `wlength` 字节数据确保每个缓存包含一个数据包 [n=wlength/MPS+wlengthMOD(MPS)]
2. 编程寄存器 USBFS_DIEPTSIZE0 的 XferSize,PktCnt 字段
3. 编程寄存器 USBFS_DIEPDMA0 来配置缓存列表的基地址
4. 编程寄存器 USBFS_DIEPCTL0.MPS 来配置控制端点最大数据包大小，编程控制输入端点寄存器 USBFS_DIEPCTL0.EPEna 置 1 使能控制输入端点
5. 编程寄存器 USBFS_DOEPTSIZE0 的 XferSize,PktCnt 和 SUPCnt 字段来接收一个提前的状态阶段或者一个新的 Setup 令牌包。
6. 编程寄存器 USBFS_DOEPDMA0 来配置缓存列表的基地址
7. 编程寄存器 USBFS_DOEPCTL0.MPS 来配置控制端点最大数据包大小，编程寄存器 USBFS_DOEPCTL0.EPEna 置 1 使能控制输出端点编程寄存器 USBFS_DOEPCTL0.CNAK=1 清除 NAK
8. 等待控制端点的输入输出中断。
9. 根据控制器接收到的令牌包的类型，会产生以下中断的组合：
如果 USBFS_DOEPINT0.SetUp=1，并且只有输出端点产生中断，表示主机发送输入令牌开始这次传输的数据阶段。
编程 USBFS_DIEPCTL0.CNAK=1 清除 NAK 并接收这次传输数据阶段的输入令牌包。编程 USBFS_DOEPCTL0.CNAK=1 清除 NAK 并接收这次传输的状态阶段的输出令牌包或者 Setup 阶段的 SETUP 令牌包。跳转到第 9 步。
如果 USBFS_DIEPINT0.XferComp=1，并且只有输入端点产生中断，表示 DMA 已经处理输入缓存数据包并压栈到对应的发送 FIFO。当最后一个缓存的 XferCompl 标志位产生，编程 USBFS_DIEPCTL0.Stall=1 当发送 FIFO 里所有的数据已经发送给主机并且发送 FIFO 为空时暂停响应主机发送的额外的输入令牌包。跳转到第 9 步，等待输出端点中断。如果 USBFS_DOEPINT0.XferComp=1，并且只有输出端点产生中断，表示主机发送控制传输的状态阶段或者一个新的控制传输的 Setup 令牌包。
 - a) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=0，程序必须读 USBFS_DOEPINT0 寄存器的 StupPktRcvd 位。如果 StupPktRcvd=1，表示 Setup 令牌已经收到。程序必须解码主机发送的控制传输类型。如果 StupPktRcvd=0，表示这次传输的状态阶段的输出令牌已经收到。程序必须为下一次控制传输准备一个缓存。
 - b) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=1，程序必须读 USBFS_DOEPINT0 寄存器的 StupPktRcvd 位，如果 StupPktRcvd=1，表示 Setup 令牌已经收到。程序必须解码主机发送的控制传输类型。

29.6.23.3 Two-Phase Control Transfer

两阶段控制传输 Setup 阶段

1. 在系统内存里创建一个缓存列表包含一个输出缓存
2. 编程寄存器 USBFS_DOEPTSIZ0 的 XferSize,PktCnt 和 SUPCnt 字段
3. 编程寄存器 USBFS_DOEPDMA0 来配置缓存列表的基地址
4. 编程寄存器 USBFS_DOEPCTL0.MPS 来配置控制端点最大数据包大小，编程寄存器 USBFS_DOEPCTL0.EPEna 置 1 使能控制输出端点
5. 等待输出端点中断
6. 当控制器收到 Setup 令牌包时，会置位 USBFS_DOEPINT0 中断
 - a) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=0，程序必须读 USBFS_DOEPINT0 寄存器的 StupPktRcvd 位。如果 StupPktRcvd=1，表示 Setup 令牌已经收到。程序必须解码主机发送的控制传输类型。根据 Wlength 字段，程序可以判断当前传输是否是一个两阶段控制传输。如果是两阶段控制传输，程序必须进入这次传输的状态阶段。
 - b) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=1，程序必须读 USBFS_DOEPINT0 寄存器的 StupPktRcvd 位。如果 StupPktRcvd=1，表示 Setup 令牌已经收到。USBFS_DOEPINT0.SetUp=1 表示主机在这次控制读传输的数据阶段发送了输入令牌包。

两阶段控制传输状态阶段

1. 在系统内存里创建一个缓存列表包含一个输入缓存。
2. 编程寄存器 USBFS_DIEPTSI20 的 XferSize,PktCnt 字段。
3. 编程寄存器 USBFS_DIEPDMA0 来配置缓存列表的基地址。
4. 编程寄存器 USBFS_DIEPCTL0.MPS 来配置控制端点最大数据包大小，编程寄存器 USBFS_DIEPCTL0.EPEna 置 1 使能控制输出端点。
5. 编程输出端点，在系统内存里创建一个缓存列表包含一个输出缓存保证在状态阶段可以接收一个新的 Setup 令牌包。
6. 编程寄存器 USBFS_DOEPTSI20 的 XferSize,PktCnt 和 SUPCnt 字。
7. 编程寄存器 USBFS_DOEPDMA0 来配置缓存列表的基地址。
8. 编程寄存器 USBFS_DOEPCTL0.MPS 来配置控制端点最大数据包大小，编程寄存器 USBFS_DOEPCTL0.EPEna 置 1 使能控制输出端。
9. 等待控制端点的输入输出中断。
10. 根据控制器接收到的令牌包的类型，会产生以下中断的组合：

如果 USBFS_DOEPINT0.Setup=1，并且只有输出端点产生中断，编程 USBFS_DIEPCTL0.CNAK=1 清除 NAK 并接收这次传输状态阶段的输入令牌包。跳转到第 8 步。

如果 USBFS_DIEPINT0.XferComp=1，只有输入端点产生中断，表示 DMA 已经处理输入缓存的零长度数据包并压栈到对应的发送 FIFO。两阶段控制传输完成，程序可以处理新的控制传输。

如果 USBFS_DOEPINT0.XferComp=1，并且只有输出端点产生中断。

- a) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=0，程序必须读 USBFS_DOEPINT0 寄存器的 StupPktRcvd 位。如果 StupPktRcvd=1，表示新的控制传输的 Setup 令牌已经收到。程序必须解码主机发送的控制传输类型。程序必须处理新的控制传输。
- b) 如果 USBFS_DOEPINT0.XferComp=1 并且 USBFS_DOEPINT0.SetUp=1，程序必须读 USBFS_DOEPINT0 寄存器的 StupPktRcvd 位。如果 StupPktRcvd=1，表示新的控制传输的 Setup 令牌已经收到。

USBFS_DOEPINT0.SetUp=1 表示主机可能发送以下令牌包：

- 控制写传输数据阶段输出令牌包
- 控制读传输数据阶段输入令牌包
- 两阶段控制传输状态阶段输入令牌包

29.6.24 Output Data Transfer

29.6.24.1 Controlling Setup Transmission

1. 编程寄存器 USBFS_DOEPTSIZ0
USBFS_DOEPTSIZ0.SUPCnt = 3
2. 根据端点特性编程寄存器 USBFS_DOEPDMA0 和寄存器 USBFS_DOEPCTL0，并使能端点 USBFS_DOEPCTL0.EPEna = 1
3. 等待 USBFS_DOEPINT0.SETUP 中断标志位置位表示 SETUP 数据传输完成。
4. 程序必须读寄存器 USBFS_DOEPTSIZ0 判定已收到的 SETUP 数据包个数并处理最后收到的 SETUP 数据包。
5. 在 DMA 模式下，程序必须判定寄存器 USBFS_DOEPINT0.Back2BackSETup 是否置位。如果置位表示控制器收到超过 3 个背靠背 SETUP 数据包。在这种情况下，程序必须忽略 USBFS_DOEPTSIZ0.SUPCnt 值，而使用寄存器 USBFS_DOEPDMA0 直接读出最后一个收到的 SETUP 数据包。

29.6.25 Non-periodic batch input data transfer without threshold

1. 编程寄存器 USBFS_DIEPTSIZn 的 XferSize 和 PktCnt 字段，编程寄存器 USBFS_DIEPDMA_n。
2. 根据端点特性编程寄存器 USBFS_DIEPCTL_n，置位 CNAK 位和 EPEna 位。
3. 在 DMA 模式，编程 NextEp 字段以确保控制器按照正确的顺序抓取输入端点数据。
4. USBFS_DIEPINTn.TimeOut 中断标志位置位表示在这个端点检测到一个超时情况。处理方法请参考章节非周期输入数据传输超时（共享 FIFO 模式）。

29.6.26 Non-periodic bulk input data transfer with threshold

1. 编程寄存器 USBFS_DIEPTSIZn 的 XferSize 和 PktCnt 字段，编程寄存器 USBFS_DIEPDMA_n。
2. 根据端点特性编程寄存器 USBFS_DIEPCTL_n，置位 CNAK 位和 EPEna 位，编程 USBFS_DIEPCTL_n.TXFNum 字段。
3. USBFS_DIEPINTn.XferCompl 中断标志位置位表示非周期输入传输完成。读取寄存器 USBFS_DIEPTSIZn 的 XferSize 和 PktCnt 字段，值都为 0，表示所有的数据已经传输完成。

29.6.27 Asynchronous Output Data Transfer Without Threshold

1. 编程寄存器 USBFS_DOEPTSIZn 的 XferSize 和 PktCnt 字段, 编程寄存器 USBFS_DOEPDMA_n。
2. 根据端点特性编程寄存器 USBFS_DOEPCTL_n, 置位 CNAK 位和 EPEna 位。
3. USBFS_DOEPINTn.XferCompl 中断标志位置位表示非同步输出传输完成。读取寄存器 USBFS_DOEPTSIZn 来判定收到的数据包大小。

29.6.28 Asynchronous Output Data Transfer with Threshold

操作流程请参考章节非同步输出数据传输无阈值。

29.6.29 Synchronous Output Data Transfer with Threshold

操作流程请参考章节非同步输出数据传输无阈值。

29.6.30 Incomplete Synchronous Output Data Transfer

1. 中断标志位 USBFS_GINTSTS.incomplSOIN 置位表示当前帧至少有一个同步输出端点有一个未完成的传输。
2. 如果该端点未完成的传输是因为同步输出数据没有完全被清空, 程序必须清空接收 FIFO 内所有同步输出数据。
3. 当接收 FIFO 内的所有数据被清空, 程序可以检测到 USBFS_DOEPINTn.XferCompl 中断。在这种情况下, 程序必须重新使能端点在下一帧接收同步输出数据。
4. 当中断标志位 USBFS_GINTSTS.incomplSOIN 置位, 程序必须读所有同步输出端点的控制寄存器(USBFS_DOEPCTL_n)判定哪一个端点在当前帧产生了未完成传输。
5. 前面的步骤必须在 USBFS_GINTSTS.SOF 中断标志位置位前完成以保证当前帧数字没有改变。
6. 当同步输出端点发生了未完成传输, 程序必须丢弃内存的数据并且禁止该端点。
`USBFS_DOEPCTLn.EPDis = 1.`
7. 等待中断标志位 USBFS_DOEPINTn.EPDisbld 置位, 然后重新使能端点接收下一帧的新数据。

29.6.31 Cycle Input Data Transfer Without Threshold

1. 编程寄存器 USBFS_DIEPTSIZn 的 XferSize 和 PktCnt 字段，编程寄存器 USBFS_DIEPDMA_n。
2. 根据端点特性编程寄存器 USBFS_DIEPCTL_n，置位 CNAK 位和 EPEna 位。
3. USBFS_DIEPINTn.INTknTXFEmp 中断标志位置位表示 DMA 还没有把所有的数据传输到发送 FIFO。
4. 如果中断端点早已使能，忽略第 3 步里的中断。如果同步端点早已使能，而发生第 3 步里的中断。
5. 当中断标志位 USBFS_DIEPINTn.XferCompl 置位，而 USBFS_DIEPINTn.INTknTXFEmp 中断标志位没有置位时，表示同步输入传输成功完成。读寄存器 USBFS_DIEPTSIZn 的 XferSize 和 PktCnt 字段，值都为 0，表示所有的数据已经传输完成。
6. 当中断标志位 USBFS_DIEPINTn.XferCompl 置位，而 USBFS_DIEPINTn.INTknTXFEmp 中断标志位置位或者没有置位时，表示中断输入传输成功完成。读寄存器 USBFS_DIEPTSIZn 的 XferSize 和 PktCnt 字段，值都为 0，表示所有的数据已经传输完成。
7. 中断标志位 USBFS_GINTSTS.incomplSOIN 置位表示控制器在当前帧没有收到至少一个周期输入令牌包。

29.6.32 Cycle Input Data Transfer with Threshold

1. 编程寄存器 USBFS_DIEPTSIZn 的 XferSize 和 PktCnt 字段，编程寄存器 USBFS_DIEPDMA_n。
2. 根据端点特性编程寄存器 USBFS_DIEPCTL_n，置位 CNAK 位和 EPEna 位。同时在寄存器 USBFS_DIEPCTL_n.TXFNum 字段指定发送 FIFO 数字。
3. USBFS_DIEPINTn.INTknTxFEmp 中断标志位置位表示 DMA 还没有把所有的数据传输到发送 FIFO。
4. 如果中断端点早已使能，忽略第 3 步里的中断。如果同步端点早已使能，而发生第 3 步里的中断。
5. 当中断标志位 USBFS_DIEPINTn.XferCompl 置位，而 USBFS_DIEPINTn.INTknTxFEmp 中断标志位没有置位时，表示同步输入传输成功完成。读寄存器 USBFS_DIEPTSIZn 的 XferSize 和 PktCnt 字段，值都为 0，表示所有的数据已经传输完成。
6. 当中断标志位 USBFS_DIEPINTn.XferCompl 置位，而 USBFS_DIEPINTn.INTknTxFEmp 中断标志位置位或者没有置位时，表示中断输入传输成功完成。读寄存器 USBFS_DIEPTSIZn 的 XferSize 和 PktCnt 字段，值都为 0，表示所有的数据已经传输完成。
7. 中断标志位 USBFS_GINTSTS.incomplSOIN 置位表示控制器在当前帧没有收到至少一个周期输入令牌包。
8. 中断标志位 USBFS_DIEPINTn.TxFifoUndrn 置位表示当前帧的同步传输出现低流量情况，程序可以选择忽略这个中断，因为这个中断经常由于 USBFS_GINTSTS.incomplSOIN 在周期帧的结尾引发。

29.7 Register Description

应用程序通过 AHB 从接口对控制和状态寄存器进行读写操作，以此来控制 USBFS 模块。USBFS 模块所有寄存器为 32 位寄存器，其地址按 32 位对齐，因此只能以 32 位的方式访问。

控制和状态寄存器分为以下几类：

- 模块全局寄存器
- 设备模式寄存器
- 电源和时钟门控控制寄存器

USBFS 模块寄存器列表以及基地址请参考表 29-3 USBFS 寄存器一览表。

USBFS 模块寄存器基地址：0x40040000

表 29-3 USBFS 寄存器一览表

偏移量	寄存器名称	访问	寄存器描述
0x08	USBFS_GAHBCFG	RW	USBFS AHB控制寄存器
0x0c	USBFS_GUSBCFG	RW	USBFS USB配置寄存器
0x10	USBFS_GRSTCTL	RW	USBFS复位寄存器
0x14	USBFS_GINTSTS	RW	USBFS模块中断寄存器
0x18	USBFS_GINTMSK	RW	USBFS中断屏蔽寄存器
0x1c	USBFS_GRXSTSR	RW	USBFS接收状态调试读取寄存器
0x20	USBFS_GRXSTSP	RW	USBFS接收状态读取和出栈寄存器
0x24	USBFS_GRXFSIZ	RW	USBFS接收FIFO大小寄存器
0x28	USBFS_GNPTXFSIZ	RW	USBFS非周期发送FIFO大小寄存器
0x2c	USBFS_GNPTXSTS	R	USBFS非周期发送FIFO/队列状态寄存器
0x3c	USBFS_CID	RW	USBFS模块ID寄存器
0x104	USBFS_DPTXFSIZ	RW	USBFS设备周期IN端点发送FIFO大小寄存器
0x800	USBFS_DCFG	RW	USBFS设备配置寄存器
0x804	USBFS_DCTL	RW	USBFS设备控制寄存器
0x808	USBFS_DSTS	R	USBFS设备状态寄存器
0x810	USBFS_DIEPMSK	RW	USBFS设备IN端点通用中断屏蔽寄存器
0x814	USBFS_DOEPMSK	RW	USBFS设备OUT端点通用中断屏蔽寄存器
0x818	USBFS_DAINT	R	USBFS设备全体端点中断寄存器
0x81c	USBFS_DAINTMSK	RW	USBFS设备全体端点中断屏蔽寄存器
0x820	USBFS_DTKNQR1	R	USBFS设备输入令牌顺序队列读寄存器1
0x824	USBFS_DTKNQR2	R	USBFS设备输入令牌顺序队列读寄存器2
0x830	USBFS_DTKNQR3	R	USBFS设备输入令牌顺序队列读寄存器3
0x834	USBFS_DTKNQR4	R	USBFS设备输入令牌顺序队列读寄存器4
0x900	USBFS_DIEPCTL0	RW	USBFS设备IN端点0控制寄存器

0x900+n*0x20(n=1,3,5,7)	USBFS_DIEPCTLn	RW	USBFS设备IN端点n控制寄存器
0x908+n*0x20(n=0,1,3,5,7)	USBFS_DIEPINTn	RW	USBFS设备IN端点n中断寄存器
0x910	USBFS_DIEPSIZ0	RW	USBFS设备IN端点0传输大小寄存器
0x910+n*0x20(n=1,3,5,7)	USBFS_DIEPSIZn	RW	USBFS设备IN端点n传输大小寄存器
0x914+n*0x20(n=1,3,5,7)	USBFS_DIEPDMAm	RW	USBFS设备IN端点n DMA地址寄存器
0xb00	USBFS_DOEPCTL0	RW	USBFS设备OUT端点0控制寄存器
0xb00+n*0x20(n=2,4,6,8)	USBFS_DOEPCTLn	RW	USBFS设备OUT端点n控制寄存器
0xb08+n*0x20(n=0,2,4,6,8)	USBFS_DOEPINTn	RW	USBFS设备OUT端点n中断寄存器
0xb10	USBFS_DOEPTSIZ0	RW	USBFS设备OUT端点0传输大小寄存器
0xb10+n*0x20(n=2,4,6,8)	USBFS_DOEPTSIZn	RW	USBFS设备OUT端点n传输大小寄存器
0xb14+n*0x20(n=2,4,6,8)	USBFS_DOEPDMAm	RW	USBFS设备OUT端点n DMA地址寄存器
0xe00	USBFS_GCCTL	RW	USBFS电源和门控时钟控制寄存器

29.7.1 USBFS Global Registers

29.7.1.1 USBFS AHB Control Register (USBFS_GAHBCFG)

AHB Configuration Register

偏移地址：0x08

复位值：0x0000 0000

该寄存器可用于在上电后或更改角色模式时对模块进行配置。该寄存器主要包含 AHB 系统相关的配置参数。

应用程序必须在开始任何 AHB 或 USB 事务前对该寄存器进行编程。请勿在初始编程后更改该寄存器。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								Reserve d	Reserve d	Reserve d	DMA EN	HBSTLEN[3:0]			GIN TMS K

位	标记	位名	功能	读写
b31~b9	Reserved	-	必须保持复位值。	R
b8	Reserved	-	必须保持复位值。	R
b7	Reserved	-	必须保持复位值。	R
b6	Reserved	-	必须保持复位值。	R
b5	DMAEN	DMA 使能	DMA 使能 (DMA enable) 0: 模块以从模式运行 1: 模块以DMA模式运行	R/W
b4~b1	HBSTLEN	批量长度/类型	批量长度/类型 (Burst length/type) 0000b:单次 0001b: INCR 0011:INCR4 0101:INCR8 0111:INCR16 其它值: 保留	R
b0	GINMSK	全局中断屏蔽	全局中断屏蔽 (Global interrupt mask) 该位用于屏蔽全局中断或对全局中断取消屏蔽。中断状态寄存器由模块进行更新，与此位的设置无关。 0: 屏蔽应用程序触发的中断 1: 取消对应用程序触发的中断的屏蔽	R/W

29.7.1.2 USBFS USB Configuration Register (USBFS_GUSBCFG)

USBFS USB configuration register

偏移地址：0x00C

复位值：0x0000 0A00

该寄存器可对模块进行配置。其中包含与 USB 和 USB-PHY 相关的配置参数。

应用程序必须在开始任何 AHB 或 USB 事务前对该寄存器进行编程。请勿在初始编程后更改该寄存器。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved	TRDT[3:0]				Reserved			PHY SEL	Reserved			TOCAL[2:0]			

位	标记	位名	功能	读写
b31~b14	Reserved	-	必须保持复位值。	R
b13~b10	TRDT	USB周转时间	USB周转时间 (USB turnaround time) 以PHY时钟数为单位设置周转时间。 要计算TRDT的值，请使用如下公式： $TRDT = 4 \times AHB\text{ 时钟} + 1\text{ 个PHY时钟}$ 例如： 1. 如果AHB时钟频率 = 84 MHz (PHY时钟频率 = 48 MHz)，则 TRDT设置为9。 2. 如果AHB时钟频率 = 48 MHz (PHY时钟频率 = 48 MHz)，则 TRDT设置为 5。	R/W
b9~b7	Reserved	-	必须保持复位值。	R
b6	PHYSEL	全速系列收发器选择	全速系列收发器选择 (Full Speed serial transceiver select) 此位为只写位且始终为1。	W
b5~b3	Reserved	-	必须保持复位值。	R
b2~b0	TOCAL	FS 超时校准	FS 超时校准 (FS timeout calibration) PHY引入的额外延迟包括应用程序在该字段中设置的PHY时钟数，以及模块的全速数据包间超时间隔。不同PHY引入的延迟对数据线状态的影响是不同的。全速操作的USB标准超时值为16到18（含）个位时间。应用程序必须根据枚举速度编程该字段。每个PHY时钟增加的位时间数为0.25个位时间。	

29.7.1.3 USBFS Reset Register (USBFS_GRSTCTL)

USBFS reset register

偏移地址：0x10

复位值：0x8000 0000

应用程序通过此寄存器复位模块中的各项硬件特性。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AHBI DL	DMA REQ														Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				TXFNUM[4:0]				TXF FLS H	RXF FLS H	Res erve d	Res erve d	HSR ST	CSR ST		

位	标记	位名	功能	读写
b31	AHBIDL	AHB主器件空闲	AHB主器件空闲(AHB master idle) 指示AHB主器件状态机处于空闲情况。	R
b30	DMAREQ	AHB主器件空闲	DMA请求信号 (DMA request signal) 该位指示DMA请求正在进行中。用于调试。	R
b29~b11	Reserved	-	读出时为“0”,写入时写“0”	R
b10~b6	TXFNUM	TxFIFO编号	TxFIFO编号(TxFIFO number) 使用TxFIFO刷新位进行FIFO刷新的FIFO编号。只有在模块将TxFIFO刷新位清零后，方可更改此字段。 <ul style="list-style-type: none"> ● 00000: 刷新Tx FIFO 0 ● 00001: 刷新TxFIFO 1 ● 00010: 刷新TxFIFO 2 ... ● 00101: 刷新TxFIFO 15 ● 10000: 刷新所有的发送FIFO 	R/W
b5	TXFFLSH	TxFIFO刷新	TxFIFO刷新 (TxFIFO flush) 此位选择性地刷新一个或所有的发送FIFO，但当模块处理通信事务时无法执行该操作。 只有在确认模块当前未对TxFIFO 执行读写操作后，应用程序方可对此位执行写操作。使用以下寄存器进行确认： <ul style="list-style-type: none"> — 读：NAK有效中断可确保模块当前未对FIFO执行读操作 — 写：USBFS_GRSTCTL中的 AHBIDL位可确保模块当前未对FIFO 执行任何写操作 	R/W
b4	RXFFLSH	RxFIFO刷新	RxFIFO刷新(RxFIFO flush) 应用程序可使用此位刷新整个RxFIFO，但必须首先确保模块当前未在处理通信事务。只有在确认模块当前未对RxFIFO执行读写操	R/W

			作后，应用程序方可对此位执行写操作。 应用程序必须等到此位清零后，方可执行其它操作。通常需要等待8个时钟周期（以PHY或AHB时钟中最慢的为准）。	
b3~b2	Reserved	-	必须保持复位值。	R
b1	HSRST	HCLK域逻辑软复位	<p>HCLK域逻辑软复位 (HCLK soft reset)</p> <p>应用程序使用此位来刷新 AHB 时钟域中的控制逻辑。仅复位 AHB 时钟域流水线。</p> <p>FIFO 不通过此位来刷新。</p> <p>遵照协议终止 AHB 上的事务后，AHB 时钟域中的所有状态机均复位至空闲状态。</p> <p>AHB 时钟域状态机所使用的 CSR 控制位清零。</p> <p>要清零该中断，需要将由 AHB 时钟域状态机生成并用于控制中断状态的状态屏蔽位清零。</p> <p>由于中断状态位并未清零，因此应用程序可以获取在该位置 1 后所发生的所有模块事件的状态。</p> <p>此位为自清零位，模块将在其中所有必要逻辑复位后将该位清零。该过程需要若干个时钟的时间，具体取决于模块的当前状态。</p>	R/W
b0	CSRST	模块软复位	<p>模块软复位 (Core soft reset)</p> <p>按如下所述将HCLK和PCLK域复位：</p> <p>除以下各位外，将各个中断和所有CSR寄存器位清零：</p> <ul style="list-style-type: none"> – USBFS_PCGCCTL中的RSTPDMODL位 – USBFS_PCGCCTL中的GAYEHCLK位 – USBFS_PCGCCTL中的PWRCLMP位 – USBFS_PCGCCTL中的STPPCLK位 – USBFS_HCFG 中的FSLSPCS位 – USBFS_DCFG 中的DSPD位 <p>将所有模块状态机 (AHB从器件除外) 复位至空闲状态，并清空所有发送FIFO和接收FIFO。</p> <p>在AHB传输的最后数据阶段结束后，尽快终止AHB主器件上的所有事务。立即终止USB上的所有事务。</p> <p>应用程序可在需要复位模块时随时对该位执行写操作。该位为自清零位，模块将在其中所有必要逻辑复位后将该位清零，该过程需要若干个时钟的时间，具体取决于模块的当前状态。该位一旦清零，软件必须等待至少3个PHY时钟后才可以访问PHY域（同步延迟）。此外，软件还必须在确定该寄存器中的位31置 1 (AHB主器件空闲) 后方可开始运行。</p> <p>软件复位通常在两种情况下使用，一是软件开发期间，二是用户动</p>	R/W

			态更改以上所列USB配置寄存器中的PHY选择位后。用户更改PHY时，将为PHY选择相应的时钟并用于PHY域中。一旦选择了新的时钟，则必须复位PHY域，才能保证正常运行。	
--	--	--	--	--

29.7.1.4 USBFS Global Interrupt Status Register (USBFS_GINTSTS)

USBFS interrupt status register

偏移地址：0x14

复位值：0x14000020

该寄存器用于借助系统级别的事件来中断应用程序。

FIFO 状态中断为只读；如果软件在处理这些中断期间对 FIFO 执行读写操作，则 FIFO 中断标志将自动清零。

使能中断位前，应用程序必须在初始化时将 USBFS_GINTSTS 寄存器清零，才可以避免在初始化前产生任何中断。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKU INT	VBU SVIN T	Rese rved	DAT AFS USP	INC OMP ISO UT	IISOI XFR	OEPI NT	IEPI NT	Rese rved	Rese rved						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EOP F	ISO ODR P	ENU MDN E	USB RST	USB SUS P	ESU SP	Rese rved	Rese rved	GOU TNA KEF F	GIN AKE FF	Res erve d	RXF LNE	SOF	Res erve d	Res erve d	Rese rved

位	标记	位名	功能	读写
b31	WKUPINT	检测到恢复/远程唤醒中断	检测到恢复/远程唤醒中断 (Resume/remote wakeup detected interrupt) 当USB总线上检测到恢复信号时，将触发该中断通过软件对该位写1清零。	R/W
b30	VBU SVINT	VBU SV有效中断	VBU SV有效中断 (VBU SV valid interrupt) 设备模式下，当检测到USBFS_VBU SV管脚由低变高时，将触发该中断。 通过软件对该位写1清零。	R/W
b29~b23	Reserved	-	必须保持复位值。	R
b22	DATAFSUS P	数据获取挂起	数据获取挂起 (Data fetch suspended) 该中断仅在DMA模式下有效。该中断指示，模块因Tx FIFO空间或请求队列空间不可用而停止为IN端点获取数据。应用程序将该中断用于端点不匹配算法中。例如，在检测到端点不匹配后，应用程序将执行以下操作： — 将全局非周期性IN NAK握手信号置1 — 禁止IN端点 — 清空FIFO — 根据IN令牌序列学习队列确定令牌序列 — 重新使能端点 — 如果全局非周期性IN NAK已清零但模块尚未为IN端点获取数	R/W

			<p>据，同时又已接收到IN令牌，则清零全局非周期性IN NAK握手信号：模块将产生“FIFO为空时接收到IN令牌”中断。然后，USBFS将NAK响应发送到主机。为避免这种情况的发生，应用程序可以检查USBFS_GINTSTS 中的DATAFSUSP中断，该中断可确保在FIFO存满后再将全局 NAK 握手信号清零。或者，应用程序可以在将全局IN NAK握手信号清零时屏蔽“当 FIFO为空时接收到IN令牌中断”。</p> <p>通过软件对该位写1清零。</p>	
b21	INCOMPISOOUT	未完成OUT同步传输	<p>INCOMPISOOUT: 未完成 OUT 同步传输 (Incomplete isochronous OUT transfer)</p> <p>模块将该中断置 1 时，指示当前帧中至少有一个同步OUT端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断 (EOPF) 位一同触发。</p> <p>通过软件对该位写1清零。</p>	R/W
b20	IISOIXFR	未完成IN同步传输	<p>未完成IN同步传输 (Incomplete isochronous IN transfer)</p> <p>模块将该中断置1时，指示当前帧中至少有一个同步IN端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断 (EOPF) 位一同触发。</p> <p>通过软件对该位写1清零。</p>	R/W
b19	OEPINT	OUT端点中断	<p>OUT端点中断(OUT endpoint interrupt)</p> <p>模块将该位置1时，指示模块中一个OUT端点上存在挂起的中断（在设备模式下）。应用程序必须读取主机USBFS_DAINT寄存器，以确定发生中断的OUT端点的准确编号，然后读取相应的USBFS_DOEPINTx寄存器，以确定引发中断的确切原因。应用程序必须先将相应USBFS_DOEPINTx 寄存器的相应状态位清零，之后才能将该位清零。</p>	R
b18	IEPINT	IN端点中断	<p>IN端点中断 (IN endpoint interrupt)</p> <p>模块将该位置1时，指示模块中一个IN端点上存在挂起的中断（在设备模式下）。应用程序必须读取主机USBFS_DAINT寄存器，以确定发生中断的IN端点的准确编号，然后读取相应的USBFS_DIEPINTx寄存器，以确定引发中断的确切原因。应用程序必须先将相应USBFS_DIEPINTx 寄存器的相应状态位清零，之后才能将该位清零。</p>	R
b17~b16	Reserved	-	必须保持复位值。	R
b15	EOPF	周期性帧结束中断	<p>周期性帧结束中断 (End of periodic frame interrupt)</p> <p>指示当前帧已达到USBFS_DCFG 寄存器中周期性帧间隔字段 (USBFS_DCFG 中的PFIVL位) 所指定的周期。</p> <p>通过软件对该位写1清零。</p>	R/W
b14	ISOODRP	丢弃同步OUT数	丢弃同步OUT数据包中断(Isochronous OUT packet dropped interrupt)	R/W

		据包中断	如果由于Rx FIFO空间不足，无法容纳同步OUT端点的最大数据包，从而导致模块无法向Rx FIFO写入同步OUT数据包，模块会将该位置1。 通过软件对该位写1清零。	
b13	ENUMDNE	枚举完成中断	枚举完成中断 (Enumeration done interrupt) 模块将该位置1时，指示速度枚举已完成。应用程序必须读取USBFS_DSTS寄存器来获取枚举速度。 通过软件对该位写1清零。	R/W
b12	USBRST	USB复位中断	USB复位中断(USB reset interrupt) 模块将该位置1时，指示在USB 上检测到复位信号。 通过软件对该位写1清零。	R/W
b11	USBSUSP	USB挂起中断	USB挂起中断(USB suspend interrupt) 模块将该位置1时，指示在USB上检测到挂起状态。当USB总线上的空闲状态保持3ms，模块便会进入挂起状态。通过软件对该位写1清零。	R/W
b10	ESUSP	早期挂起中断	早期挂起中断(Early suspend interrupt) 模块将该位置1时，指示已检测到USB处于空闲状态的时间达到3ms。	R/W
b9~b8	Reserved	-	必须保持复位值。	R
b7	GONAKEFF	全局OUT NAK有效中断	全局OUT NAK有效中断(Global OUT NAK effective interrupt) 指示USBFS_DCTL寄存器中由应用程序设置的“将全局OUT NAK置 1”位 (USBFS_DCTL中的SGONAK位) 已在模块中生效。通过写入 USBFS_DCTL 寄存器中的“将全局 OUT NAK 清零”位 (USBFS_DCTL中的CGONAK位)，可将该位清零。	R
b6	GINAKEFF	全局非周期性IN NAK有效中断	全局非周期性IN NAK有效中断(Global IN nonperiodic NAK effective interrupt) 指示USBFS_DCTL寄存器中由应用程序设置的“将全局非周期性IN NAK置 1”位 (USBFS_DCTL中的SGINAK位) 已在模块中生效。也就是说，模块已对应用程序设置的全局IN NAK位进行采样，结果已生效。通过清零USBFS_DCTL寄存器中的“将全局非周期性IN NAK清零”位 (USBFS_DCTL中的CGINAK位)，可将该位清零。此中断不一定表示USB上已发送了一个NAK 握手信号。STALL位优先级高于NAK 位。	R
b5	Reserved	-	必须保持复位值。	R
b4	RXFNE	RxFIFO非空中断	RxFIFO非空中断(RxFIFO non-empty interrupt) 指示RxFIFO中至少有一个数据包等待读取。	R
b3	SOF	帧起始中断	帧起始中断 (Start of frame interrupt) 模块将该位置1时，指示 USB 上已接收到一个SOF令牌。应用程序可通过读取设备状态寄存器来获得当前的帧编号。只有在模块以 FS 模式运行时，才会出现此中断。	R/W

			通过软件对该位写1清零。	
b2~b0	Reserved	-	必须保持复位值。	R

29.7.1.5 USBFS Global Interrupt Mask Register (USBFS_GINTMSK)

USBFS interrupt mask register

偏移地址：0x18

复位值：0x00000000

该寄存器与模块中断寄存器结合使用，以中断应用程序。如果将某个中断位屏蔽，则不会产生与该位相关的中断。

但是，与该中断相对应的模块中断 (USBFS_GINTSTS) 寄存器位仍会置 1。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKU IM	VBU SVIM	Reserred	Reserred	Reserred	Reserred	Reserred	Reserred	Reserred	DAT AFS USP M	INC OMP ISO OUT M	IISOI XFR M	OEPI M	IEPI M	Reserred	Reserred
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EOP FM	ISO ODR PM	ENU MDN EM	USB RSTM	USB SUS PM	ESU SPM	Reserred	Reserred	GOU TNA KEF FM	GIN AKE FFM	Reserve d	RXF NEM	SOF M	Reserve d	Reserred	Reserred

位	标记	位名	功能	读写
b31	WKUPIM	检测到恢复/远程唤醒中断屏蔽	检测到恢复/远程唤醒中断屏蔽 (Resume/remote wakeup detected interrupt mak) 0: 屏蔽中断 1: 使能中断	R/W
b30	VBUSVM	VBUS有效中断屏蔽	VBUS有效中断屏蔽 (VBUS valid interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b29~b23	Reserved	-	必须保持复位值。	R
b22	DATAFSUSPM	数据获取挂起中断屏蔽	数据获取挂起中断屏蔽 (Data fetch suspended interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b21	INCOMPISO OUTM	未完成OUT同步传输中断屏蔽	INCOMPISOOUT: 未完成 OUT 同步传输中断屏蔽 (Incomplete isochronous OUT transfer interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b20	IISOIXFRM	未完成IN同步传输中断屏蔽	未完成IN同步传输中断屏蔽 (Incomplete isochronous IN transfer interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b19	OEPIIM	OUT端点中断屏蔽	OUT端点中断屏蔽(OUT endpoint interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W

b18	IEPIM	IN端点中断屏蔽	IN端点中断屏蔽(IN endpoint interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b17~b16	Reserved	-	必须保持复位值。	R
b15	EOPFM	周期性帧结束中断屏蔽	周期性帧结束中断屏蔽(End of periodic frame interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b14	ISOODRPM	丢弃同步OUT数据包中断屏蔽	丢弃同步OUT数据包中断屏蔽(Isochronous OUT packet dropped interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b13	ENUMDNE M	枚举完成中断屏蔽	枚举完成中断屏蔽(Enumeration done interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b12	USBRSTM	USB复位中断屏蔽	USB复位中断屏蔽(USB reset interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b11	USBSUSPM	USB挂起中断屏蔽	USB挂起中断屏蔽(USB suspend interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b10	ESUSPM	早期挂起中断屏蔽	早期挂起中断屏蔽(Early suspend interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b9~b8	Reserved	-	必须保持复位值。	R
b7	GONAKEFF M	全局OUT NAK有效中断屏蔽	全局OUT NAK有效中断屏蔽(Global OUT NAK effective interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b6	GINAKEFF M	全局非周期性IN NAK有效中断屏蔽	全局非周期性IN NAK有效中断屏蔽(Global IN nonperiodic NAK effective interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b5	Reserved	-	必须保持复位值。	R
b4	RXFNEM	RxFIFO 非空中断屏蔽	RxFIFO非空中断屏蔽(RxFIFO non-empty interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b3	SOFM	帧起始中断屏蔽	帧起始中断屏蔽(Start of frame interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b2~b0	Reserved	-	必须保持复位值。	R

29.7.1.6 USBFS Receive Status Debug Read/USBFS Status Read and Pop Registers (USBFS_GRXSTS/USBFS_GRXSTSP)

USBFS Receive status debug read/USBFS status read and pop registers

读取的偏移地址：0x01C

出栈的偏移地址：0x020

复位值：0x0000 0000

读取接收状态调试读取寄存器将返回接收 FIFO 顶部的内容。读取接收状态读取和出栈寄存器将额外弹出 RxFIFO 顶部的数据条目。

当接收 FIFO 为空时，模块会忽略对该寄存器的读取或出栈操作，并返回值 0x0000 0000。当模块中断寄存器的接收 FIFO 非空位（USBFS_GINTSTS 中的 RXFNE 位）置位时，应用程序必须仅弹出接收状态 FIFO。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										PKTSTS[3:0]			DPID[1]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DPID[0]										EPNUM[3:0]					

位	标记	位名	功能	读写
b31~b21	Reserved	-	必须保持复位值。	R
b20~b17	PKTSTS	数据包状态	数据包状态(Packet status) 指示接收的数据包的状态 0001: 全局OUT NAK (触发中断) 0010: 接收到OUT数据包 0011: OUT传输完成 (触发中断) 0100: SETUP事务完成 (触发中断) 0110: 接收到SETUP数据包 其它值: 保留	R
b16~b15	DPID	数据PID	数据PID(Data PID) 指示接收的OUT数据包的数据PID 00: DATA0 10: DATA1 01: DATA2 11: MDATA	R
b14~b4	BCNT	字节计数	字节计数(Byte count) 指示接收的数据包的字节数。	R
b3~b0	EPNUM	端点编号	端点编号(Endpoint number) 指示当前接收的数据包所属的端点编号。	R

29.7.1.7 USBFS Receive FIFO Size Register (USBFS_GRXFSIZ)

USBFS Receive FIFO size register

偏移地址：0x024

复位值：0x0000 0140

此应用程序可以对必须分配给 RxFIFO 的 RAM 大小进行编程。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					RXFD[10:0]										

位	标记	位名	功能	读写
b31~b11	Reserved	-	必须保持复位值。	R
b10~b0	RXFD	RxFifo深度	RXFD: RxFIFO深度 (RxFIFO depth) 以32位字为单位。 最小值为16 最大值为256 上电复位值为最大Rx数据FIFO深度。	R/W

29.7.1.8 USBFS Non-Periodic Transmit FIFO Size Register (USBFS_GNPTXFSIZ)

USBFS Non-Periodic Transmit FIFO size register

偏移地址：0x028

复位值：可配置

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
NPTxFDep[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

位	标记	位名	功能	读写
b31~b16	NPTxFDep	TxFIFO深度	非周期发送FIFO大小设定 最小值16 最大值32768	R/W
b15~b0	Reserved	-	必须保持复位值。	R

29.7.1.9 USBFS Non-Periodic Transmit FIFO/Queue Status Register (USBFS_GNPTXSTS)

USBFS Non-Periodic Transmit FIFO/Queue Status register

偏移地址：0x02c

复位值：可配置

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Res.	NPTxQTop[6:0]								NPTxQSpAvail[7:0]						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
NPTxFSpAvail															

位	标记	位名	功能	读写
b31	Reserved		必须保持复位值。	R
b30~b24	NPTxQTop	非周期发送请求队列	MAC正在处理的非周期发送请求队列	R
b23~b16	NPTxQSpAvail	非周期发送请求队列可用空间	表示非周期发送请求队列可用空间值 0 : 非周期发送请求队列满 1 : 1个位置可用 2 : 2个位置可用 N : n个位置可用(n <= 8)	R
b15~b0	NPTxFSpAvail	- 非周期发送 FIFO 可用空间	表示非周期发送FIFO可用空间值 0 : 非周期发送FIFO满 1 : 1个字可用 2 : 2个字可用 N : n个字可用(n <= 1024)	R

29.7.1.10 USBFS Module ID Register (USBFS_CID)

USBFS core ID register

偏移地址：0x03C

复位值：0x12345678

该寄存器为可编程用户配置 ID 寄存器。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PRODUCT_ID[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PRODUCT_ID[15:0]															

位	标记	位名	功能	读写
b31~b0	PRODUCT_ID	产品ID字段	产品ID字段(Product ID field) 可通过应用程序编程的ID字段。	R/W

29.7.1.11 USBFS Device Periodic IN Endpoint Transmit FIFO Size Register (USBFS_DPTXFSIZx) (x = 1..4)

USBFS Device Periodic Transmit FIFO size register

偏移地址：0x104+(x-1)*0x4

此应用程序可以对必须分配给设备 TxFIFO 的大小进行编程。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DPTxFSIZE[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DPTxFStAddr[11:0]															

位	标记	位名	功能	读写
b31~b16	DPTxFSIZE	周期 IN 端点 TxFIFO 大小	设备周期IN端点TxFIFO深度 (Device Period Transmit TxFIFO size) 以32位字为单位。 最小值为4 最大值为256	R
b15~b0	DPTxFStAddr	周期 IN 端点 TxFIFOx RAM 起始地址	(Period IN endpoint FIFOx transmit RAM start address) 此字段包含周期IN端点发送FIFOx的存储器起始地址。 该地址必须与32位存储器位置对齐。	R/R_W

29.7.2 USBFS Device Mode Register

设备模式寄存器会影响设备模式下的模块操作。

除非特别说明，否则寄存器描述中的位值以二进制表示。

29.7.2.1 USBFS Device Configuration Register (USBFS_DCFG)

USBFS Device configuration register

偏移地址：0x800

复位值：0x0820 0000

此寄存器在上电、执行某些控制命令或枚举后，会将模块配置为设备模式。请勿在初始编程后更改该寄存器。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved	PFIVL[1:0]			DAD[6:0]			Rese rv ed	NZL SOH SK		DSPD[1:0]					

位	标记	位名	功能	读写
b31~b13	Reserved	-	必须保持复位值。	R
b12~b11	PFIVL	周期性帧间隔	周期性帧间隔(Periodic frame interval) 指示一帧内必须使用周期性帧中断通知应用程序的时间点。此功能可用于确定该帧的所有同步通信是否完成。 00: 80%帧间隔 01: 85%帧间隔 10: 90%帧间隔 11: 95%帧间隔	R/W
b10~b4	DAD	设备地址	设备地址(Device address) 应用程序必须在执行每个SetAddress控制命令后根据命令参数对该字段进行设置。	R/W
b3	Reserved	-	必须保持复位值。	R
b2	NZLSOHSK	非零长度状态 OUT握手信号	非零长度状态OUT握手信号 (Non-zero-length status OUT handshake) 在控制传输状态阶段的OUT事务期间，当模块收到非零长度数据包后，应用程序可以使用此字段选择要发送的握手信号。 1: 收到非零长度状态OUT事务时，回复STALL握手信号，收到的OUT数据包不发送给应用程序。 0: 将收到的OUT数据包（零长度或非零长度）发送给应用程序，并基于设备端点控制寄存器中端点的NAK和STALL位回复握手信号。	R/W

b1~b0	DSPD	设备速度	<p>设备速度 (Device speed) 指示应用程序要求模块进行枚举所采用的速度，或应用程序支持的最大速度。但是，实际总线速度只有在完成chirp序列后才能确定，同时此速度基于与模块连接的USB主机的速度。</p> <p>00: 保留 01: 保留 10: 保留 11: 全速 (USB 2.0收发器时钟为48 MHz)</p>	R/W
-------	------	------	---	-----

29.7.2.2 USBFS Device Control Register (USBFS_DCTL)

USBFS Device control register

偏移地址：0x804

复位值：0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved	POP RGD NE	CGO NAK	SGO NAK	CGI NAK	SGI NAK	TCTL[2:0]	GON STS	GIN STS	SDIS	RWU SIG					

位	标记	位名	功能	读写
b31~b12	Reserved	-	必须保持复位值。	R
b11	POPRGDNE	上电编程完成	上电编程完成(Power-on programming done) 应用程序使用此位指示寄存器从掉电模式唤醒后已完成编程。	R/W
b10	CGONAK	清零全局OUT NAK	清零全局OUT NA (Clear global OUT NAK) 对此位执行写操作会将全局OUT NAK清零。	W
b9	SGONAK	置位全局OUT NAK	置位全局OUT NAK(Set global OUT NAK) 对此位执行写操作会将全局OUT NAK置 1。 应用程序使用此位在所有OUT端点发送NAK握手信号。 应用程序只有确定模块中断寄存器中全局OUT NAK有效位(USBFS_GINTSTS中GONAKEFF位)已清零时，才可以将此位置1。	W
b8	CGINAK	清零全局IN NAK	清零全局IN NAK (Clear global IN NAK) 对此位执行写操作会将全局IN NAK清零。	W
b7	SGINAK	置位全局IN NAK	置位全局IN NAK (Set global IN NAK) 对此字段执行写操作会将全局非周期性IN NAK置 1。应用程序使用此位使所有非周期性IN端点发送NAK握手信号。 应用程序只有确定模块中断寄存器中全局IN NAK有效位(USBFS_GINTSTS中GINAKEFF位)已清零时，才可以将此位置1。	W
b6~b4	Reserved	-	必须保持复位值。	R
b3	GONSTS	全局OUT NAK状态	全局OUT NAK状态(Global OUT NAK status) 0: 将根据FIFO状态和NAK和STALL位设置发送握手信号。 1: 无论Rx FIFO中是否还有空闲空间都不接收数据。除 SETUP事务之外，对所有收到的数据包回复NAK握手信号。所有同步类型的OUT数据包都将被丢弃。	R
b2	GINSTS	全局IN NAK状态	全局IN NAK状态(Global IN NAK status) 0: 将根据发送FIFO中的数据可用性回复握手信号。 1: 使所有非周期性IN端点回复NAK握手信号，无需考虑发送FIFO中的数据可用性。	R

b1	SDIS	软断连	<p>软断连(Soft disconnect)</p> <p>应用程序使用该位向USBFS模块发出执行软断开的信号。该位置1时，主机不会看到设备已连接，且该设备也不会接收USB上的信号。在应用程序将此位清零之前，模块会保持断开状态。</p> <p>0：正常工作。此位在软断连之后清零，会使主机收到设备已连接的事件。重新连接设备之后，USB主机会重新启动设备枚举。</p> <p>1：使主机收到设备断开连接的事件。</p> <p>全速时，软断连的最长时间规定如下：</p> <p>挂起状态：最长时间为1ms+2.5us</p> <p>空闲状态：2.5us</p> <p>非空闲或挂起状态：2.5us</p>	R/W
b0	RWUSIG	发送远程唤醒信号	<p>发送远程唤醒信号(Remote wakeup signaling)</p> <p>应用程序将此位置1时，模块会启动远程唤醒信号，以唤醒USB主机。应用程序必须将此位置1以使模块退出挂起状态。根据USB 2.0规范，应用程序必须在将此位置1之后的1 ms到15 ms内将其清零。</p>	R/W

29.7.2.3 USBFS Device Status Register (USBFS_DSTS)

USBFS Device status register

偏移地址：0x808

复位值：0x0000 0002

此寄存器指示模块在出现 USB 相关事件时的状态。发生中断时，必须从设备全体中断 (USBFS_DAINT) 寄存器读取发生中断的端点信息。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										FNSOF[13:8]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FNSOF[7:0]								Reserved				EERR	ENUMSPD[1:0]	SUSPSTS	

位	标记	位名	功能	读写
b31~b22	Reserved	-	必须保持复位值。	R
b21~b8	FNSOF	接收SOF的帧编号	接收SOF的帧编号 (Frame number of the received SOF)	R
b7~b4	Reserved	-	必须保持复位值。	R
b3	EERR	不定错误	不定错误(Erratic error) 模块将该位置1以报告任何不定错误。 由于不定错误，USBFS控制器会进入挂起状态，并且会USBFS_GINTSTS寄存器的早期挂起位 (USBFS_GINTSTS中的ESUSP位) 生成一个中断。如果早期挂起中断是由不定错误触发，则应用程序只能执行软断开以恢复通信。	R
b2~b1	ENUMSPD	枚举速度	枚举速度(Enumerated speed) 指示USBFS控制器通过chirp序列检测速度后被枚举成的速度。 01: 保留 10: 保留 11: 全速 (PHY 时钟运行频率为 48 MHz) 其它值: 保留	R
b0	SUSPSTS	挂起状态	挂起状态(Suspend status) 在设备模式下，只要在USB上检测到挂起状态，该位就会置1。当USB总线上的空闲状态保持3ms，模块便会进入挂起状态。出现以下情况时，模块会退出挂起状态： — USB 数据线上有活动 — 应用程序对USBFS_DCTL寄存器的远程唤醒信号位 (USBFS_DCTL中的RWUSIG位) 执行写操作。	R

29.7.2.4 USBFS Device IN Endpoint General Interrupt Mask Register (USBFS_DIEPMSK)

USBFS Device IN endpoint common interrupt mask register

偏移地址：0x810

复位值：0x0000 0000

此寄存器与全体端点的各个 USBFS_DIEPINTx 寄存器配合使用，以便在每个 IN 端点上生成中断。通过对此寄存器的相应位执行写操作，可屏蔽 USBFS_DIEPINTx 寄存器中的 IN 端点中断。默认情况下，状态中断都被屏蔽。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								INEP NEM	INEP NM M	ITTX FEM SK	TOM	Rese rved	EPD M	XFR CM	

位	标记	位名	功能	读写
b31~b7	Reserved	-	必须保持复位值。	R
b6	INEPNEM	IN端点NAK有效中断屏蔽	IN端点NAK有效中断屏蔽 (IN endpoint NAK effective mask) 0: 屏蔽中断 1: 使能中断	R/W
b5	INEPNMM	EP不匹配时接收到IN令牌中断屏蔽	EP不匹配时接收到IN令牌中断屏蔽 (IN token received with EP mismatch mask) 0: 屏蔽中断 1: 使能中断	R/W
b4	ITTXFEMSK	TxFIFO为空时接收到IN令牌中断屏蔽	TxFIFO为空时接收到IN令牌中断屏蔽 (IN token received when TxFIFO empty mask) 0: 屏蔽中断 1: 使能中断	R/W
b3	TOM	超时中断屏蔽 (非同步端点)	超时中断屏蔽 (非同步端点) (Timeout condition mask (Non-isochronous endpoints)) 0: 屏蔽中断 1: 使能中断	R/W
b2	Reserved	-	必须保持复位值。	R
b1	EPDM	端点禁止中断屏蔽	端点禁止中断屏蔽(Endpoint disabled interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b0	XFRCM	传输完成中断屏蔽	传输完成中断屏蔽(Transfer completed interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W

29.7.2.5 USBFS Device OUT Endpoint General Interrupt Mask Register (USBFS_DOEPMSK)

USBFS Device OUT endpoint common interrupt mask register

偏移地址：0x814

复位值：0x0000 0000

此寄存器与全体端点的各个 USBFS_DOEPINTx 寄存器配合使用，以便在每个 OUT 端点上生成中断。通过对此寄存器的相应位执行写操作，可屏蔽 USBFS_DOEPINTx 寄存器中的 OUT 端点中断。默认情况下，状态中断都被屏蔽。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
OTE PDM	STU PM	Rese rved	EPD M	XFR CM											

位	标记	位名	功能	读写
b31~b5	Reserved	-	必须保持复位值。	R
b4	OTE PDM	端点禁止时接收到OUT令牌中断屏蔽 (OUT token received when endpoint disabled mask) 仅适用于控制 OUT 端点。 0: 屏蔽中断 1: 使能中断	端点禁止时接收到OUT令牌中断屏蔽 (OUT token received when endpoint disabled mask) 仅适用于控制 OUT 端点。 0: 屏蔽中断 1: 使能中断	R/W
b3	STU PM	SETUP 阶段完成 中断屏蔽	SETUP阶段完成中断屏蔽(SETUP phase done mask) 仅适用于控制端点。 0: 屏蔽中断 1: 使能中断	R/W
b2	Reserved	-	必须保持复位值。	R
b1	EPDM	端点禁止中断屏 蔽	端点禁止中断屏蔽(Endpoint disabled interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W
b0	XFR CM	传输完成中断屏 蔽	传输完成中断屏蔽(Transfer completed interrupt mask) 0: 屏蔽中断 1: 使能中断	R/W

29.7.2.6 USBFS Device Global Endpoint Interrupt Register (USBFS_DAINT)

USBFS Device OUT endpoint common interrupt mask register

偏移地址：0x818

复位值：0x0000 0000

当端点上发生有效事件时，USBFS_DAINT 寄存器将通过 USBFS_GINTSTS 寄存器中的设备 OUT 端点中断位或设备 IN 端点中断位（分别为 USBFS_GINTSTS 中的 OEPINT 或 IEPINT 位）来中断应用程序。每个端点对应一个中断位，OUT 端点和 IN 端点均最多有 16 个中断位。双向端点将使用相应的 IN 和 OUT 中断位。当应用程序将相应设备端点 x 中断寄存器(USBFS_DIEPINTx/USBFS_DOEPINTx)中的位置 1 和清零时，此寄存器中的相应位也将置 1 和清零。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OEPINT[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IEPINT[15:0]															

位	标记	位名	功能	读写
b31~b16	OEPINT	OUT端点中断位	OUT端点中断位(OUT endpoint interrupt bits) 每个OUT端点对应一位： OUT端点0对应位16，而OUT端点4对应位20。	R/W
b15~b0	IEPINT	IN端点中断位	IN端点中断位(IN endpoint interrupt bits) 每个IN端点对应一位： IN端点0对应位0，而IN端点4对应位4。	R/W

29.7.2.7 USBFS Device Global Endpoint Interrupt Mask Register (USBFS_DAINTMSK)

USBFS Device all endpoints interrupt mask register

偏移地址：0x81C

复位值：0x0000 0000

USBFS_DAINTMSK 寄存器与设备端点中断寄存器结合使用，在设备端点上发生事件时中断应用程序。但是，与该中断相对应的 USBFS_DAINT 寄存器位仍会置 1。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OEPINTM[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IEPINTM[15:0]															

位	标记	位名	功能	读写
b31~b16	OEPINTM	OUT端点中断屏蔽位	OUT端点中断平局位(OUT endpoint interrupt mask bits) 每个OUT端点对应一位： OUT端点0对应位16，而OUT端点4对应位20。 0: 屏蔽中断 1: 使能中断	R/W
b15~b0	IEPINTM	IN端点中断屏蔽位	IN端点中断位(IN endpoint interrupt mask bits) 每个IN端点对应一位： IN端点0对应位0，而IN端点4对应位4。 0: 屏蔽中断 1: 使能中断	R/W

29.7.2.8 USBFS Device IN Command Sequence Learning Queue Read Register 1 (USBFS_DTKNQR1)

USBFS Device IN Token Sequence Learning Queue Read register 1

偏移地址：0x820

复位值：0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPTkn[23:8]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EPTkn[7:0]								Wra pBit	Reserved		INTknWPtr[4:0]				

位	标记	位名	功能	读写
b31~b8	EPTkn	端点指令	每4位表示指令的端点值 [31:28] 指令5的端点值 [27:24] 指令4的端点值 [23:20] 指令3的端点值 [19:16] 指令2的端点值 [15:12] 指令1的端点值 [11:8] 指令0的端点值	R
b7	WrapBit	缠绕位	这位置1表示写指针缠绕。这位会在学习队列清除的时候被清零。	R
b6~b5	Reserved		必须保持复位值。	R
b4~b0	INTknWPtr	IN指令队列写指 针	IN指令队列写指针	R

29.7.2.9 USBFS Device IN Command Sequence Learning Queue Read Register 2 (USBFS_DTKNQR2)

USBFS Device IN Token Sequence Learning Queue Read register 2

偏移地址：0x824

复位值：0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPTkn[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EPTkn[15:0]															

位	标记	位名	功能	读写
b31~b0	EPTkn	端点指令	每4位表示指令的端点值 [31:28] 指令13的端点值 [27:24] 指令12的端点值 [23:20] 指令11的端点值 [19:16] 指令10的端点值 [15:12] 指令9的端点值 [11:8] 指令8的端点值 [7:4] 指令7的端点值 [3:0] 指令6的端点值	R

29.7.2.10 USBFS Device IN Command Sequence Learning Queue Read Register 3 (USBFS_DTKNQR3)

USBFS Device IN Token Sequence Learning Queue Read register 3

偏移地址：0x830

复位值：0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPTkn[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EPTkn[15:0]															

位	标记	位名	功能	读写
b31~b0	EPTkn	端点指令	每4位表示指令的端点值 [31:28] 指令21的端点值 [27:24] 指令20的端点值 [23:20] 指令19的端点值 [19:16] 指令18的端点值 [15:12] 指令17的端点值 [11:8] 指令16的端点值 [7:4] 指令15的端点值 [3:0] 指令14的端点值	R

29.7.2.11 USBFS Device IN Command Sequence Learning Queue Read Register 4 (USBFS_DTKNQR4)

USBFS Device IN Token Sequence Learning Queue Read register 4

偏移地址：0x834

复位值：0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPTkn[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EPTkn[15:0]															

位	标记	位名	功能	读写
b31~b0	EPTkn	端点指令	每4位表示指令的端点值 [31:28] 指令29的端点值 [27:24] 指令28的端点值 [23:20] 指令27的端点值 [19:16] 指令26的端点值 [15:12] 指令25的端点值 [11:8] 指令24的端点值 [7:4] 指令23的端点值 [3:0] 指令22的端点值	R

29.7.2.12 USBFS Device Control IN Endpoint 0 Control Register (USBFS_DIEPCTL0)

USBFS Device control IN endpoint 0 control register

偏移地址：0x900

复位值：0x0000 8000

此寄存器用于控制控制传输端点 0。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPE NA	EPDI S	Reserved	SNA K	CNA K		TXFNUM[3:0]		STAL L	Rese rved	EPTYP[1:0]	NAK STS	Rese rved			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USB AEP															MPSIZ[1:0]

位	标记	位名	功能	读写
b31	EPENA	端点使能	端点使能(Endpoint enable) 应用程序将此位置1以在端点0上启动数据发送。 在此端点上触发以下任一中断之前，模块会将此位清零： — 端点禁止 — 传输完成	R/W
b30	EPDIS	端点禁止	端点禁止(Endpoint disable) 即使在该端点上的传输完成之前，应用程序也可将此位置1，以停止端点上的数据发送。应用程序必须等到发生端点禁止中断后，才能将端点视为禁止端点。在端点禁止中断位置1前，模块会将此位清零。只有在该端点的端点使能位置1后，应用程序才可将该位置1。	R/W
b29~b28	Reserved	-	必须保持复位值。	R
b27	SNAK	置位NAK位	置位NAK位(Set NAK) 对此位进行写操作会将端点的NAK位置1。 通过此位，应用程序可以控制端点上NAK握手信号的发送。模块也可在端点接收到SETUP数据包后将该端点的此位置1。	R/W
b26	CNAK	清零NAK位	清零NAK位(Clear NAK) 对此位进行写操作会将端点的NAK位清零。	R/W
b25~b22	TXFNUM	TxFIFO编号	TxFIFO编号(TxFIFO number) 该值设置为分配给IN端点0的FIFO编号。	R/W
b21	STALL	STALL握手	STALL握手(STALL handshake) 应用程序只能将此位置1，端点接收到SETUP令牌时，模块会将此位清零。如果NAK位、全局IN NAK或全局OUT NAK与此位均置1，则STALL位优先。	R/W
b20	Reserved	-	必须保持复位值。	R
b19~b18	EPTYP	端点类型	端点类型(Endpoint type) 硬件设置为‘00’，表示控制类型的端点。	R

b17	NAKSTS	NAK状态	NAK状态(NAK status) 指示以下结果： 0：模块根据FIFO状态回复非NAK握手。 1：模块在此端点上回复NAK握手。 当此位置1时（无论是被应用程序还是被模块），即使TxFIFO中仍有数据可用，模块也会停止发送数据。无论此位如何设置，模块总是通过ACK握手响应SETUP数据包。	R
b16	Reserved	-	必须保持复位值。	R
b15	USBAEP	USB活动端点	USB活动端点(USB active endpoint) 此位总是置1，指示在所有配置和接口中控制端点0始终处于激活状态。	R
b14~b2	Reserved	-	必须保持复位值。	R
b1~b0	MPSIZ	最大数据包大小	最大数据包大小(Maximum packet size) 应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。 00: 64字节 01: 32字节 10: 16字节 11: 8字节	R/W

29.7.2.13 USBFS Device IN Endpoint x Control Register (USBFS_DIEPCTLx) (x=1..4)

USBFS Device IN endpoint x control register

偏移地址: 0x900 + (端点编号 × 0x20)

复位值: 0x0000 0080

应用程序使用此寄存器控制各个逻辑端点（端点 0 除外）的行为。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPE NA	EPDI S	SOD DFR M	SD0 PID/ SEV NFR M	SNA K	CNA K	TXFNUM[3:0]				STAL L	Rese rved	EPTYP[1:0]		NAK STS	EON UM/ DPI D
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USB AEP	Reserved			MPSIZ[10:0]											

位	标记	位名	功能	读写
b31	EPENA	端点使能	端点使能(Endpoint enable) 应用程序将此位置1以在端点上启动数据发送。 在此端点上触发以下任一中断之前，模块会将此位清零： — SETUP阶段完成 — 端点禁止 — 传输完成	R/W
b30	EPDIS	端点禁止	端点禁止(Endpoint disable) 即使在该端点上的传输完成之前，应用程序也可将此位置1，以停止端点上的数据发送。应用程序必须等到发生端点禁止中断后，才能将端点视为禁止端点。在端点禁止中断位置1前，模块会将此位清零。只有在该端点的端点使能位置1后，应用程序才可将该位置1。	R/W
b29	SODDFRM	设置奇数帧	设置奇数帧(Set odd frame) 仅适用于同步IN和OUT端点。 对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为奇数帧。	R/W
b28	SD0PID/ SEVNFRM	设置DATA0 PID/ SEVNFRM	设置DATA0 PID (Set DATA0 PID) 仅适用于中断/批量IN 端点。 对此字段进行写操作会将此寄存器中的端点数据PID(DPID)字段设置为 DATA0。 SEVNFRM: 设置偶数帧 (Set even frame) 仅适用于同步IN端点。 对此字段进行写操作会将偶数/奇数帧(EONUM)字段设置为偶数帧。	R/W
b27	SNAK	置位NAK位	置位NAK位(Set NAK)	R/W

			<p>对此位进行写操作会将端点的NAK位置 1。</p> <p>通过此位，应用程序可以控制端点上NAK握手信号的发送。发生传输完成中断时或端点上接收到SETUP后，模块也可以将OUT端点的这个位置 1</p>	
b26	CNAK	清零NAK位	<p>清零NAK位(Clear NAK)</p> <p>对此位进行写操作会将端点的NAK位清零。</p>	R/W
b25~b22	TXFNUM	TxFIFO编号	<p>TxFIFO编号(TxFIFO number)</p> <p>这些位用于指定与此端点相关联的 FIFO 编号。必须为每个有效的IN端点设置单独的 FIFO编号。</p> <p>此字段仅针对IN端点有效。</p>	R/W
b21	STALL	STALL握手	<p>STALL握手(STALL handshake)</p> <p>应用程序将此位置1使得设备对来自 USB 主机的所有令牌都回复STALL。如果NAK位、全局IN NAK或全局OUT NAK与此位同时置1，则STALL位优先。只有应用程序能够将此位清零，而模块则不能。</p>	R/W
b20	Reserved	-	必须保持复位值。	R
b19~b18	EPTYP	端点类型	<p>端点类型(Endpoint type)</p> <p>以下是这个逻辑端点支持的传输类型。</p> <p>00: 控制 01: 同步 10: 批量 11: 中断</p>	R
b17	NAKSTS	NAK状态	<p>NAK状态(NAK status)</p> <p>指示以下结果：</p> <p>0: 模块根据FIFO状态回复非NAK握手。</p> <p>1: 模块在此端点上回复NAK握手。</p> <p>当应用程序或模块将此位置1时：</p> <p>对于非同步IN端点：即使TxFIFO中存在可用数据，模块也会停止通过 IN端点发送任何数据。</p> <p>对于同步IN端点：即使TxFIFO中存在可用数据，模块也会发送长度为零的数据包。</p> <p>无论此位如何设置，模块总是通过ACK握手响应SETUP数据包。</p>	R
b16	EONUM/DPID	偶数/奇数帧/ 端点数据PID	<p>偶数/奇数帧 (Even/odd frame)</p> <p>仅适用于同步IN端点。</p> <p>指示模块为此端点发送/接收同步的数据所在的帧的编号。应用程序必须通过此寄存器中的SEVNFRM和SODDFRM字段对偶数/奇数帧编号进行编程，以便此端点发送/接收同步数据。</p> <p>0: 偶数帧 1: 奇数帧</p>	R/W

			DPID：端点数据PID (Endpoint data PID) 仅适用于中断/批量IN端点。 包含此端点上将要接收或发送的数据包的PID。端点激活后，应用程序必须对要在此端点上接收或发送的首个数据包的PID进行编程。应用程序使用SD0PID寄存器字段对DATA0或DATA1 PID进行编程。 0: DATA0 1: DATA1	
b15	USBAEP	USB活动端点	USB活动端点(USB active endpoint) 指示此端点在当前配置和接口中是否激活。检测到USB复位后，模块会为所有端点（端点0除外）将此位清零。接收到SetConfiguration和SetInterface命令后，应用程序必须相应地对端点寄存器进行编程并将此位置1。	R/W
b14~b11	Reserved	-	必须保持复位值。	R/W
b10~b0	MPSIZ	最大数据包大小	最大数据包大小(Maximum packet size) 应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。 此值以字节为单位。	

29.7.2.14 USBFS Device IN Endpoint x Interrupt Register (USBFS_DIEPINTx) (x=0..4)

USBFS Device IN endpoint x interrupt register

偏移地址：0x908 + (端点编号 × 0x20)

复位值：0x0000 0000

此寄存器指示端点在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的 IN 端点中断位 (USBFS_GINTSTS 中的 IEPINT 位) 置 1 时，应用程序必须读取此寄存器。在应用程序能够读取此寄存器之前，必须先读取设备全体端点中断(USBFS_DAINT)寄存器，以获取设备端点 x 中断寄存器的准确端点编号。应用程序必须将此寄存器中的相应位清零，才能将 USBFS_DAINT 和 USBFS_GINTSTS 寄存器中的对应位清零。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TXFE	NEPNE	Reserved	TTXFE	TOC	Reserved	EPDISD	XFR C

位	标记	位名	功能	读写
b31~b8	Reserved	-	必须保持复位值。	R
b7	TXFE	发送FIFO为空	发送FIFO为空 (Transmit FIFO empty) 当此端点的TxFIFO为半空或全空时，此中断被置位。TxFIFO为半空还是全空状态由USBFS_GAHBCFG寄存器中的TxFIFO空白别位 (USBFS_GAHBCFG中的TXFELVL位) 决定。	R
b6	INEPNE	IN端点NAK有效	INEPNE: IN端点NAK有效 (IN endpoint NAK effective) 当应用程序通过向USBFS_DIEPCTLx 中的CNAK位写入数据来将IN端点 NAK清零时，此位可被清零。 该中断指示模块已对 (由应用程序或模块) 置1的NAK采样，结果已生效。该中断指示由应用程序置1的IN端点NAK位已在模块中起作用。 此中断不保证在USB上发送了NAK握手信号。 STALL位的优先级高于 NAK 位。 软件写1也可将此位清零。	R/W
b5	Reserved	-	必须保持复位值。	R
b4	ITTXFE	TxFIFO为空时接收到IN令牌	TxFIFO为空时接收到IN令牌 (IN token received when TxFIFO is empty) 仅适用于非周期性 IN 端点。 当和该端点相对应的TxFIFO (周期性/非周期性) 为空时，接收到IN令牌，从而产生中断。 通过软件写1清零。	
b3	TO	超时	超时条件(Timeout condition) 仅适用于控制IN端点。	R/W

			指示该端点对最近收到的IN令牌响应超时。 通过软件写1清零。	
b2	Reserved	-	必须保持复位值。	R
b1	EPDISD	端点禁止中断	端点禁止中断 (Endpoint disabled interrupt) 此位指示该端点已经由应用程序禁止掉。 通过软件写1清零。	R/W
b0	XFRC	传输完成中断	传输完成中断 (Transfer completed interrupt) 此字段指示在此端点上设置的传输已经在USB和AHB上传输完成。 通过软件写1清零。	R/W

29.7.2.15 USBFS Device IN Endpoint 0 Transfer Size Register (USBFS_DIEPTSIZE0)

USBFS Device IN endpoint 0 transfer size register

偏移地址：0x910

复位值：0x0000 0000

在使能端点 0 之前，应用程序必须修改此寄存器。通过设备控制端点 0 控制寄存器中的端点使能位（USBFS_DIEPCTL0 中的 EPENA）使能端点 0 后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

非零端点使用端点 1~5 的寄存器。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										PKTCNT[1:0]	Reserved				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										XFRSIZ[6:0]					

位	标记	位名	功能	读写
b31~b21	Reserved	-	必须保持复位值。	R
b20~b19	PKTCNT	数据包计数	数据包计数 (Packet count) 指示端点0的一次数据传输包含的数据包个数。 每次从TxFIFO读取数据包 (最大大小或短数据包) 时，此字段将递减。	R/W
b18~b7	Reserved	-	必须保持复位值。	R
b6~b0	XFRSIZ	传输大小	传输大小(Transfer size) 指示端点0的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次向TxFIFO写入来自外部存储器的数据包时，模块会使此字段递减。	R/W

29.7.2.16 USBFS Device IN Endpoint x Transfer Size Register (USBFS_DIEPTSIz) (x=1..4)

USBFS Device IN endpoint x transfer size register

偏移地址：0x910 + (端点编号 × 0x20)

复位值：0x0000 0000

在使能该端点之前，应用程序必须修改此寄存器。通过 USBFS_DIEPCTLx 寄存器中的端点使能位（USBFS_DIEPCTLx 中的 EPENA 位）使能该端点后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved		PKTCNT[9:0]										XFRSIZ[18:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
XFRSIZ[15:0]															

位	标记	位名	功能	读写
b31~b29	Reserved	-	必须保持复位值。	R
b28~b19	PKTCNT	数据包计数	数据包计数 (Packet count) 指示该端点上的一次数据传输包含的数据包个数。 每次从TxFIFO读取数据包 (最大大小或短数据包) 时，此字段将递减。	R/W
b18~b0	XFRSIZ	传输大小	传输大小(Transfer size) 此字段包含当前端点的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。 传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次向TxFIFO写入来自外部存储器的数据包时，模块会使此字段递减。	R/W

29.7.2.17 USBFS Device IN Endpoint x DMA Address Register (USBFS_DIEPDMAx) (x=0..4)

USBFS Device IN endpoint x transfer size register

偏移地址: 0x914 + (端点编号 × 0x20)

复位值: 0x0000 0000

该寄存器用于设定设备端点 DMA 模式时 DMA 地址。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DMAADDR[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMAADDR[15:0]															

位	标记	位名	功能	读写
b31~b0	DMAADDR	DMA地址	DMA地址(DMA address) 该位包含使用DMA进行端点上数据存储时的外部存储区起始地址。 注意：对于控制端点，该字段所指向的存储区也用于存储控制 OUT 数据包以及 SETUP 事务数据包。连续接收到三个以上的 SETUP 数据包时，存储器中的 SETUP 数据包将被覆盖。每次进行 AHB 传输，该寄存器都会递增。应用程序必须设定一个双字对齐地址。	R/W

注意：

- 当 *USBFS_GAHBCFG.DMAEN* 为 1 时，该寄存器的值应小于 0x20002000(RAM)/0x00010000(FLASH)。

29.7.2.18 USBFS Device IN Endpoint Transmit FIFO Status Register (USBFS_DTXFSTSx) (x=0..4)

USBFS Device IN endpoint transmit FIFO status register

偏移地址：0x918 + (端点编号 × 0x20)

复位值：0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
INEPTFSAV[15:0]															

位	标记	位名	功能	读写
b31~b16	Reserved	-	必须保持复位值。	R
b15~b0	INEPTFSAV	IN端点Tx FIFO可用空间	IN端点Tx FIFO可用空间 (IN endpoint Tx FIFO space available) 指示端点 Tx FIFO 中的可用空闲空间大小。 以32位字为单位： 0x0：端点Tx FIFO已满 0x1：1个字可用 0x2：2个字可用 0xn：n个字可用	R

29.7.2.19 USBFS Device Control OUT Endpoint 0 Control Register (USBFS_DOEPCTL0)

USBFS Device control OUT endpoint 0 control register

偏移地址：0xB00

复位值：0x0000 8000

此寄存器用于控制控制传输端点 0。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPE NA	EPDI S	Reserved	SNA K	CNA K	Reserved	Reserved	STAL L	SNP M	EPTYP[1:0]	NAK STS	Rese rved				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USB AEP	Reserved													MPSIZ[1:0]	

位	标记	位名	功能	读写
b31	EPENA	端点使能	端点使能(Endpoint enable) 应用程序将此位置1以在端点0上启动数据发送。 在此端点上触发以下任一中断之前，模块会将此位清零： — SETUP 阶段完成 — 端点禁止 — 传输完成	R/W
b30	EPDIS	端点禁止	端点禁止(Endpoint disable) 应用程序无法禁止控制OUT端点0。	R
b29~b28	Reserved	-	必须保持复位值。	R
b27	SNAK	置位NAK位	置位NAK位(Set NAK) 对此位进行写操作会将端点的NAK位置1。 通过此位，应用程序可以控制端点上NAK握手信号的发送。模块也可在端点接收到SETUP数据包后将该端点的此位置1。	R/W
b26	CNAK	清零NAK位	清零NAK位(Clear NAK) 对此位进行写操作会将端点的NAK位清零。	R/W
b25~b22	Reserved	-	必须保持复位值。	R
b21	STALL	STALL握手	STALL握手(STALL handshake) 此端点接收到SETUP令牌时，应用程序只能将此位置1，而模块会将其清零。如果NAK位、全局OUT NAK与此位同时置1，则STALL位优先。无论此位如何设置，模块总是通过ACK握手响应SETUP数据包。	R/W
b20	SNPM	监听模式	监听模式 (Snoop mode) 此位用于将端点配置为监听模式。在监听模式下，模块不会在将OUT数据包传输到应用存储区前检查其是否正确。	R/W
b19~b18	EPTYP	端点类型	端点类型(Endpoint type) 硬件设置为'00'，表示控制类型的端点。	R/W
b17	NAKSTS	NAK状态	NAK状态(NAK status)	R

			指示以下结果： 0：模块根据FIFO状态回复非NAK握手。 1：模块在此端点上回复NAK握手。 当应用程序或模块将此位置1时，即使Rx FIFO中存在空间可继续容纳收到的数据包，模块也会停止接收数据。无论此位如何设置，模块总是通过ACK握手响应SETUP数据包。	
b16	Reserved	-	必须保持复位值。	R
b15	USBAEP	USB活动端点	USB活动端点(USB active endpoint) 此位总是置1，指示在所有配置和接口中控制端点0始终处于激活状态。	R
b14~b2	Reserved	-	必须保持复位值。	R
b1~b0	MPSIZ	最大数据包大小	最大数据包大小(Maximum packet size) 控制OUT端点0的最大数据包大小与在控制IN端点0中进行编程的值相同。 00: 64 字节 01: 32 字节 10: 16 字节 11: 8 字节	R/W

29.7.2.20 USBFS Device OUT Endpoint x Control Register (USBFS_DOEPCTLx) (x=1..4)

USBFS Device OUT endpoint x control register

偏移地址: 0xB00 + (端点编号 × 0x20)

复位值: 0x0000 0000

应用程序使用此寄存器控制各个逻辑端点（端点 0 除外）的行为。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPE NA	EPDI S	SOD DFR M/ SD1 PID	SD0 PID/ SEV NFR M	SNA K	CNA K	Reserved				STAL L	SNP M	EPTYP[1:0]	NAK STS	EON UM/ DPID	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USB AEP	Reserved			MPSIZ[10:0]											

位	标记	位名	功能	读写
b31	EPENA	端点使能	端点使能(Endpoint enable) 软件置位, USBFS清零 0: 端点除能 1: 端点使能	R/W
b30	EPDIS	端点禁止	端点禁止(Endpoint disable) 即使在该端点上的传送完成之前, 应用程序也可将此位置1, 以停止端点上的数据发送/接收。应用程序必须等到发生端点禁止中断后, 才能将端点视为禁止端点。在端点禁止中断位置1前, 模块会将此位清零。只有在该端点的端点使能位置1后, 应用程序才可将该位置1。	R/W
b29	SD1PID/ SODDFRM	设 置 DATA1 PID/ 设置奇数 帧	设置DATA1 PID (Set DATA1 PID) 仅适用于中断/批量OUT端点。对此字段进行写操作会将此寄存器中的端点数据PID(DPID) 字段设置为DATA1。 SODDFRM: 设置奇数帧 (Set odd frame) 仅适用于同步OUT端点。对此字段进行写操作会将偶数/奇数帧(EONUM) 字段设置为奇数帧。	R
b28	SD0PID/ SEVNFRM	设 置 DATA0 PID/ SEVNFRM	设置DATA0 PID (Set DATA0 PID) 仅适用于中断/批量OUT端点。 对此字段进行写操作会将此寄存器中的端点数据PID(DPID)字段设置为 DATA0。 SEVNFRM: 设置偶数帧 (Set even frame) 仅适用于同步OUT端点。 对此字段进行写操作会将偶数/奇数帧(EONUM)字段设置为偶数帧。	R
b27	SNAK	置位NAK位	置位NAK位(Set NAK) 对此位进行写操作会将端点的NAK位置1。	R/W

			通过此位，应用程序可以控制端点上NAK握手信号的发送。发生传输完成中断时或端点上接收到SETUP后，模块也可以将OUT端点的这个位置1。	
b26	CNAK	清零NAK位	清零NAK位(Clear NAK) 对此位进行写操作会将端点的NAK位清零。	R/W
b25~b22	Reserved	-	必须保持复位值。	R
b21	STALL	STALL握手	STALL握手(STALL handshake) 此端点接收到SETUP令牌时，应用程序只能将此位置1，而模块会将其清零。如果NAK位、全局OUT NAK与此位同时置1，则STALL位优先。只有应用程序能够将此位清零，而模块则不能。	R/W
b20	SNPM	监听模式	监听模式 (Snoop mode) 此位用于将端点配置为监听模式。在监听模式下，模块不会再检查接收数据的正确性。	R/W
b19~b18	EPTYP	端点类型	端点类型(Endpoint type) 以下是这个逻辑端点支持的传输类型。 00: 控制 01: 同步 10: 批量 11: 中断	R/W
b17	NAKSTS	NAK状态	NAK状态 (NAK status) 指示以下结果： 0: 模块根据FIFO状态回复非NAK握手。 1: 模块在此端点上回复NAK握手。 当应用程序或模块将此位置1时： 即使Rx FIFO存在空间可容纳传入数据包，模块也会停止在OUT端点上接收任何数据。 无论此位如何设置，模块总是通过ACK握手响应SETUP数据包。	R
b16	EONUM/ DPID	偶数/奇数帧/ 端点数据PID	偶数/奇数帧 (Even/odd frame) 仅适用于同步OUT 端点。 指示模块为此端点发送/接收同步的数据所在的帧的编号。应用程序必须通过此寄存器中的SEVNFRM和SODDFRM字段对偶数/奇数帧编号进行编程，以便此端点发送/接收同步数据。 0: 偶数帧 1: 奇数帧 DPID: 端点数据PID (Endpoint data PID) 仅适用于中断/批量OUT端点。 包含此端点上将要接收或发送的数据包的PID。端点激活后，应用程序必须对要在此端点上接收或发送的首个数据包的PID进行编	R/W

			程。应用程序使用 SD0PID 寄存器字段对 DATA0或DATA1 PID 进行编程。 0: DATA0 1: DATA1	
b15	USBAEP	USB活动端点	USB活动端点(USB active endpoint) 指示此端点在当前配置和接口中是否激活。检测到USB复位后，模块会为所有端点（端点0除外）将此位清零。接收到SetConfiguration和SetInterface命令后，应用程序必须相应地对端点寄存器进行编程并将此位置1。	R/W
b14~b11	Reserved	-	必须保持复位值。	R/W
b10~b0	MPSIZ	最大数据包大小	最大数据包大小(Maximum packet size) 应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。 此值以字节为单位。	

29.7.2.21 USBFS Device OUT Endpoint x Interrupt Register (USBFS_DOEPINTx) (x=0..4)

USBFS Device OUT endpoint x interrupt register

偏移地址：0xb08 + (端点编号 × 0x20)

复位值：0x0000 0080

此寄存器指示端点在出现 USB 和 AHB 相关事件时的状态。当 USBFS_GINTSTS 寄存器中的 OUT 端点中断位 (USBFS_GINTSTS 中的 OEPINT 位) 置 1 时，应用程序必须读取此寄存器。在应用程序能够读取此寄存器之前，必须先读取 USBFS_DAINT 寄存器，以获取 USBFS_DOEPINTx 寄存器的准确端点编号。应用程序必须将此寄存器中的相应位清零，才能将 USBFS_DAINT 和 USBFS_GINTSTS 寄存器中的对应位清零。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
B2B STU P	Rese rved	OTE PDIS	STU P	Rese rved	EPDI SD	XFR C									

位	标记	位名	功能	读写
b31~b7	Reserved	-	必须保持复位值。	R
b6	B2BSTUP	接收到连续的 SETUP数据包	接收到连续的 SETUP 数据包 (Back-to-back SETUP packets received) 仅适用于控制OUT端点。此位指示该端点已接收到三个以上的连续 SETUP数据包。软件写1也可将此位清零。	R/W
b5	Reserved	-	必须保持复位值。	R
b4	OTEPDIS	端点禁止时接收到OUT令牌	端点禁止时接收到OUT令牌 (OUT token received when endpoint disabled) 仅适用于控制OUT端点。指示在尚未使能端点时接收到OUT令牌，从而产生中断。通过软件写1清零。	R/W
b3	STUP	SETUP阶段完成	SETUP 阶段完成 (SETUP phase done) 仅适用于控制OUT端点。指示控制端点的SETUP阶段已完成，当前控制传输中不再接收到连续的 SETUP数据包。在此中断上，应用程序可以对接收到的SETUP数据包进行解码。 通过软件写1清零。	
b2	Reserved	-	必须保持复位值。	R
b1	EPDISD	端点禁止中断	端点禁止中断 (Endpoint disabled interrupt) 此位指示该端点已经由应用程序禁止掉。 通过软件写1清零。	R/W
b0	XFRC	传输完成中断	传输完成中断 (Transfer completed interrupt) 此字段指示在此端点上设置的传输已经在USB和AHB上传输完成。 通过软件写1清零。	R/W

29.7.2.22 USBFS Device OUT Endpoint 0 Transfer Size Register (USBFS_DOEPTSIZ0)

USBFS Device OUT endpoint 0 transfer size register

偏移地址：0xB10

复位值：0x0000 0000

在使能端点 0 之前，应用程序必须修改此寄存器。通过设备控制端点 0 控制寄存器中的端点使能位 (USBFS_DIEPCTL0 中的 EPENA) 使能端点 0 后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

非零端点使用端点 1~4 的寄存器。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved	STUPCNT[1:0]		Reserved				PKTCNT	Reserved								
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	Reserved								XFRSIZ[6:0]							

位	标记	位名	功能	读写
b31	Reserved	-	必须保持复位值。	R
b30~b29	STUPCNT	SETUP数据包计数	SETUP数据包计数 (SETUP packet count) 此字段指定端点能连续接收的SETUP数据包数量。 01: 1个数据包 10: 2个数据包 11: 3个数据包	
b28~b20	Reserved	-	必须保持复位值。	R
b19	PKTCNT	数据包计数	数据包计数 (Packet count) 一次传输中应该接收到数据包数量。 在端点使能前，软件设置该位，在传输开始后，每当数据包接收到后，该域数值自动减少。	R/W
b18~b7	Reserved	-	必须保持复位值。	R
b6~b0	XFRSIZ	传输大小	传输大小(Transfer size) 指示端点0的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次从Rx FIFO读取数据包并将其写入外部存储器时，模块会使此字段递减。	R/W

29.7.2.23 USBFS Device OUT Endpoint x Transfer Size Register (USBFS_DOEPTSIZx) (x=1..4)

USBFS Device OUT endpoint x transfer size register

偏移地址：0xB10 + (端点编号 × 0x20)

复位值：0x0000 0000

在使能该端点之前，应用程序必须修改此寄存器。通过 USBFS_DOEPCTLx 寄存器中的端点使能位（USBFS_DOEPCTLx 中的 EPENA 位）使能该端点后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved		PKTCNT[9:0]										XFRSIZ[18:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
XFRSIZ[15:0]															

位	标记	位名	功能	读写
b31~b29	Reserved	-	必须保持复位值。	R
b28~b19	PKTCNT	数据包计数	数据包计数 (Packet count) 指示该端点上的一次数据传输包含的数据包个数。 每次向Rx FIFO写入数据包 (最大大小或短数据包) 后，此字段将递减。	R/W
b18~b0	XFRSIZ	传输大小	传输大小(Transfer size) 此字段包含当前端点的一次数据传输包含的数据量，以字节为单位。仅当些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次从Rx FIFO读取数据包并将其写入外部存储器时，模块会使此字段递减。	R/W

29.7.2.24 USBFS Device OUT Endpoint x DMA Address Register (USBFS_DOEPDMAx) (x=0..4)

USBFS Device OUT endpoint x transfer size register

偏移地址：0xB14 + (端点编号 × 0x20)

复位值：0xFFFF XXXX

该寄存器用于设定设备端点 DMA 模式时 DMA 地址。

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DMAADDR[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMAADDR[15:0]															

位	标记	位名	功能	读写
b31~b0	DMAADDR	DMA地址	DMA地址(DMA address) 该位包含使用DMA进行端点上数据发送时的外部存储区起始地址。 注意：对于控制端点，该字段所指向的存储区也用于存储控制OUT数据包以及SETUP事务数据包。连续接收到三个以上的SETUP数据包时，存储器中的SETUP数据包将被覆盖。每次进行AHB传输，该寄存器都会递增。应用程序必须设定一个双字对齐地址。	R/W

注意：

- 当 *USBFS_GAHBCFG.DMAEN* 为 1 时，该寄存器的值应小于 0x20002000(RAM)/0x00010000(FLASH)。

29.7.3 USBFS Clock Gating Control Register

通过门控时钟控制寄存器控制 HCLK 和 PHY 时钟从而降低功耗。除非特别说明，否则寄存器描述中的位值以二进制表示。

29.7.3.1 USBFS Clock Gating Control Register (USBFS_GCCTL)

偏移地址：0xE00

复位值：0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved														GATEHCLK	STPPCLK

位	标记	位名	功能	读写
b31~b2	Reserved	-	必须保持复位值。	R
b1	GATEHCLK	门控HCLK	门控HCLK (Gate HCLK) 当USB通信挂起或会话无效时，应用程序会将此位置 1，以停止对除AHB总线从接口、主接口和唤醒逻辑之外的模块提供时钟。当USB恢复通信或新会话启动时，应用程序将此位清零。	R/W
b0	STPPCLK	停止PHY时钟	停止PHY时钟(Stop PHY clock) 当USB通信挂起、会话无效或设备断开连接时，应用程序将此位置 1以停止PH 时钟。当USB恢复通信时，应用程序将此位清零。	R/W

30 Controller Area Network (CAN)

30.1 Introduction

CAN (Controller Area Network) 总线是一种可以在无主机情况下实现微处理器或者设备之间相互通信的总线标准。本模块遵循 CAN 总线协议 2.0A 和 2.0B 协议并向上兼容 CAN-FD。CAN 总线控制器可以处理总线上的数据收发，在本产品中，CAN 具有 8 组筛选器。筛选器用于为应用程序选择要接收的消息。

应用程序通过 1 个高优先级的主发送缓冲器（Primary Transmit Buffer，以下简称 PTB）和 4 个辅发送缓冲器（Secondary Transmit Buffer，以下简称 STB）将发送数据送至总线，由发送调度器决定邮箱发送顺序。通过 10 个接收缓冲器（Receive Buffer，以下简称 RB）获取总线数据。4 个 STB 以及 10 个 RB 可以理解为一个 4 级 FIFO 和一个 10 级 FIFO，FIFO 完全由硬件控制。

CAN 总线控制器同时也可以支持时间触发 CAN 通信（Time-trigger communication）。

CAN 主要特性：

- 完全支持 CAN2.0A/CAN2.0B 协议。
- 向上兼容 CAN-FD 协议。
- 支持最高通信波特率 1Mbit/s
- 支持 1~1/256 的波特率预分频，灵活配置波特率。
- 10 个接收缓冲器
 - FIFO 方式
 - 错误或者不被接收的数据不会覆盖存储的消息
- 1 个高优先主发送缓冲器 PTB
- 4 个副发送缓冲器 STB
 - FIFO 方式
 - 优先级仲裁方式
- 8 组独立的筛选器
 - 支持 11 位标准 ID 和 29 位扩展 ID
 - 可编程 ID CODE 位以及 MASK 位
- PTB/STB 均支持支持单次发送模式
- 支持静默模式
- 支持回环模式
- 支持捕捉传输的错误种类以及定位仲裁失败位置
- 可编程的错误警告值
- 支持 ISO11898-4 规定时间触发 CAN 以及接收时间戳

30.2 CAN System Block Diagram

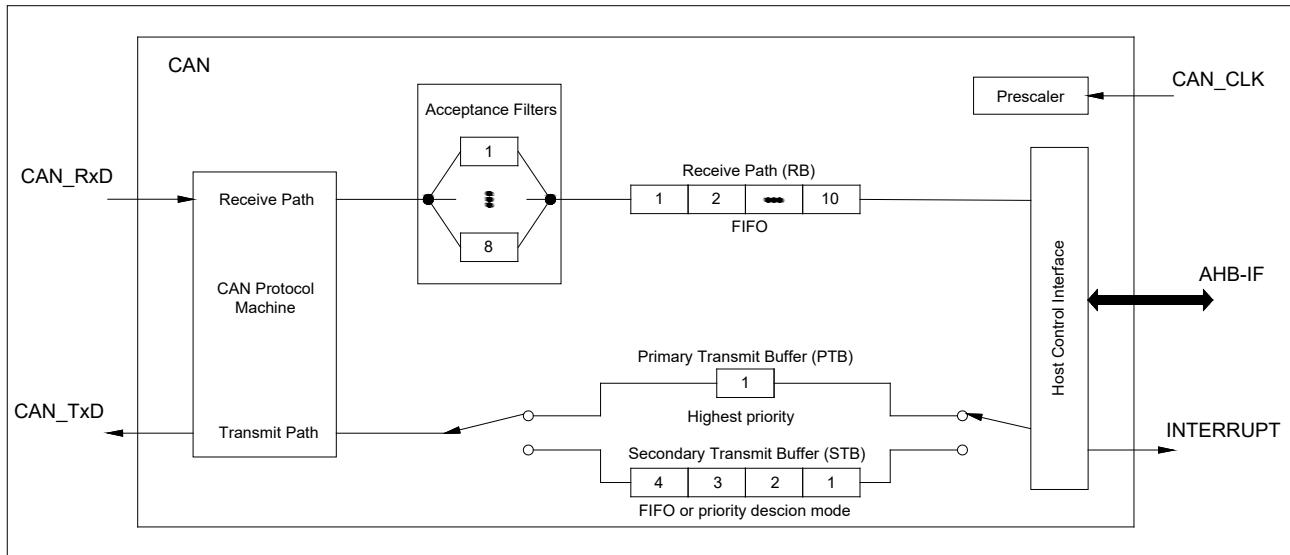


图 30-1 CAN 系统框图

30.3 Pin Description

表 30-1 CAN 管脚说明

管脚名	方向	功能描述
CAN_RxD	输入	CAN接收数据信号
CAN_TxD	输出	CAN发送数据信号

30.4 Functional Description

本章将对 CAN 功能详细说明。

30.4.1 Baud rate setting

下图给出 CAN 位时间定义图，虚线上部分为 CAN 协议规定的位时间，虚线下部分为本 CAN 控制器 CAN-CTRL 定义的位时间。其中 segment1 和 segment2 可以通过寄存器 BT 设定。BT 寄存器只能在 CFG_STAT.RESET=1 即 CAN 软件复位时设定。

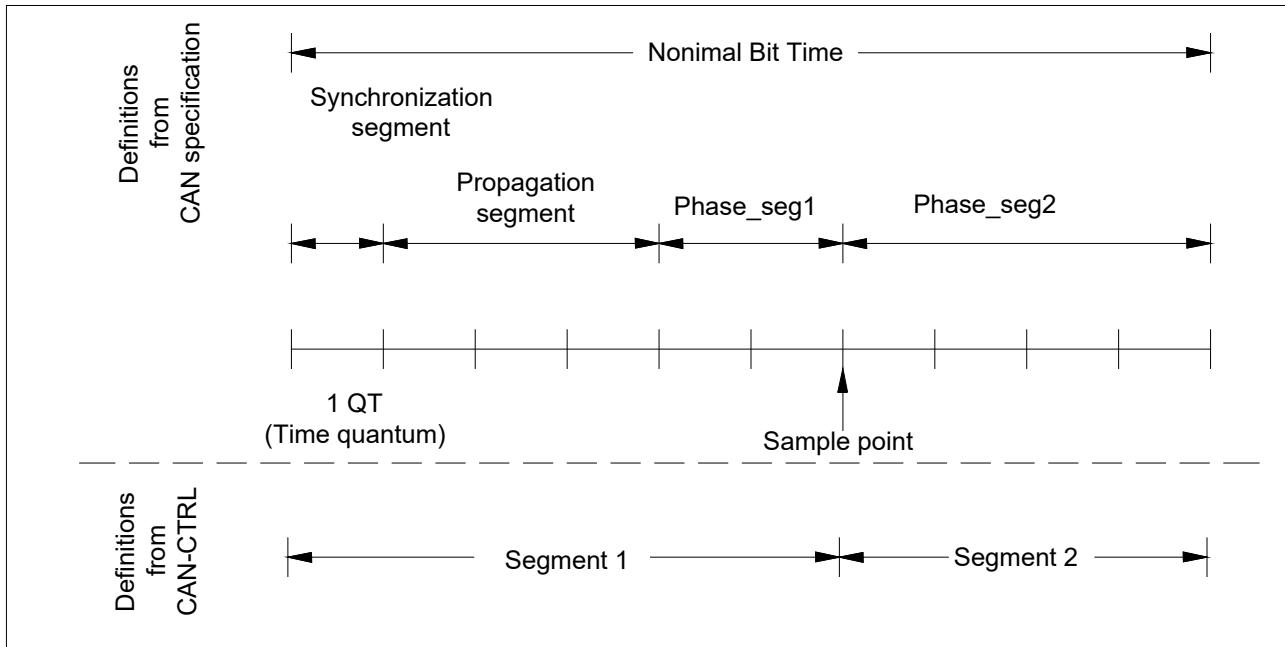


图 30-2 CAN 位时间定义图

TQ 计算方法请参考以下公式，其中 PRESC 通过 BT 寄存器的 PRESC 位设定。 f_{can_clk} 为 CAN 通信时钟频率。

$$TQ = \frac{PRESC+1}{f_{can_clk}}$$

采样点的计算方法请参考下列说明：

- 1) 若 BT 寄存器的 PRESC 位设定 ≥ 1 时，参考图 30-2 的 sample point 所示，采样点位于 segment1 和 segment2 的分界点上；
- 2) 若 BT 寄存器的 PRESC 位设定为 0 时，采样点在 sample point，也即 segment1 和 segment2 分界点前面 2 个 TQ 的位置。

建议将 BT 寄存器的 PRESC 位设定值 ≥ 1 。

位时间计算方法请参考以下公式，其中 SEG_1 和 SEG_2 通过 BT 寄存器的 SEG_1 位和 SEG_2 位设定。

$$BT = t_{SEG1} + t_{SEG2} = ((SEG_1+2) + (SEG_2+1)) \times TQ$$

表 30-2 CAN 位时间设定规则

位	设定范围	规则
BT寄存器的SEG_1位	0~63	SEG_1 ≥ SEG_2 + 1 SEG_2 ≥ SJW
BT寄存器的SEG_2位	0~7	
BT寄存器的SJW位	0~7	

30.4.2 Transmit Buffer

CAN_CTRL 提供两种发送缓冲器用于发送数据，主发送数据缓冲器 PTB 和副发送缓冲器 STB。PTB 具有最高的优先级，但只能缓冲一帧数据。STB 优先级比 PTB 低，但可以缓冲 4 帧数据，且 STB 内 4 帧数据可以工作在 FIFO 模式或者优先级仲裁模式。

STB 中的 4 帧说数据可以通过 TCMD 寄存器的 TSALL 位设定为 1 全部发送，在 FIFO 模式下，最先写入的数据先发送，在优先级模式下，ID 小的数据先发送。

PTB 中的数据具有最高优先级，所以 PTB 发送能推迟 STB 发送，但是已经赢得仲裁并开始发送的 STB 不能够被 PTB 发送推迟。

PTB 和 STB 中的一帧数据需占用 4 个字，可以通过 TBUF 寄存器进行访问。通过 TCMD 寄存器的 TBSEL 位选择 PTB 或者 STB，TBSEL=0，选择 PTB，TBSEL=1，选择 STB。通过 TCTRL 寄存器的 TSNEXT 位选择 STB 中的下一个 SLOT。对应关系如下图所示：

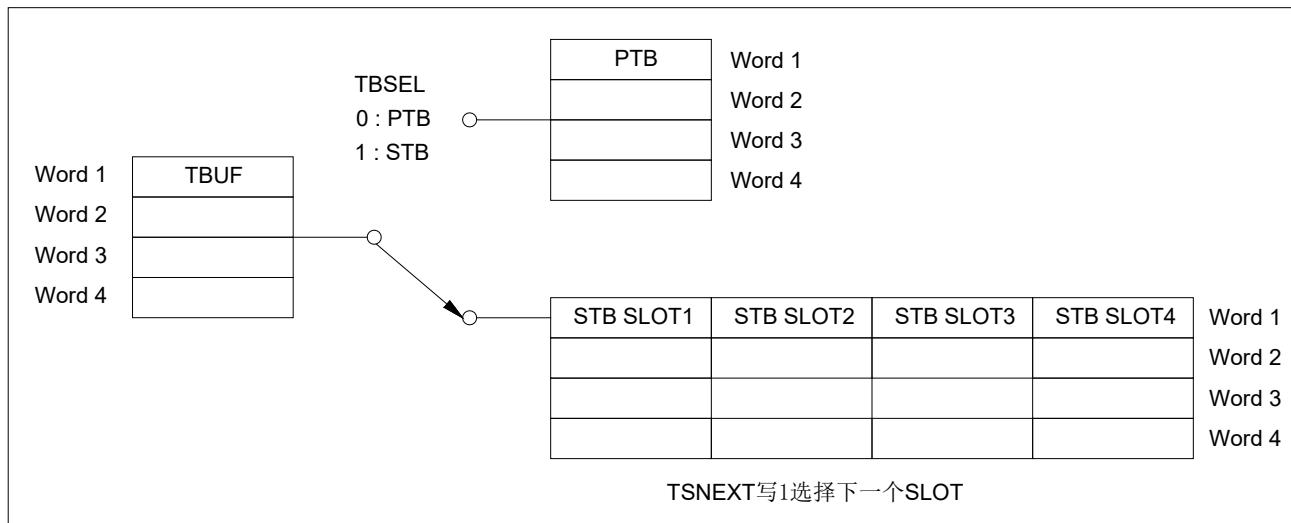


图 30-3 CAN TBUF 寄存器写发送缓冲器和示意图

30.4.3 Receive Buffer

CAN_CTRL 提供 10 个 SLOT 的接收缓冲器用于存储接收到的数据，该 10SLOT 的接收缓冲器工作在 FIFO 模式。每个 RB SLOT 需占用 4 个字，通过 RBUF 寄存器来读取接收到的数据，总是最先读取最早接收到数据，并通过 RCTRL 寄存器的 RREL 设置为 1 释放已经读取的 RB SLOT，并指向下一个 RB SLOT。

通过 RBUF 读取 RB SLOT 示意图如下。

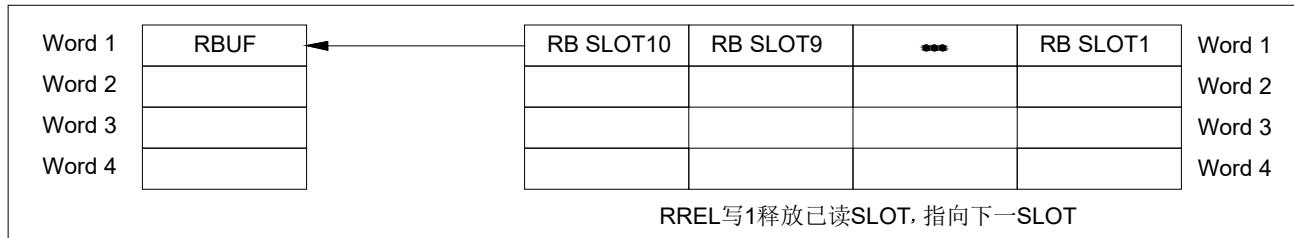


图 30-4 CAN RBUF 寄存器读接收缓冲器示意图

30.4.4 Receive Filter Register Group

CAN_CTRL 提供 8 组 32 位筛选器用于过滤接收到的数据从而降低 CPU 负荷，筛选器可以支持标准格式 11 位 ID 或者扩展格式 29 位 ID。每组筛选器有一个 32 位 ID CODE 寄存器和一个 32 位 ID MASK 寄存器，ID CODE 寄存器用于比较接收到 CAN ID，而 ID MASK 寄存器用于选择比较的 CAN ID 位。对应的 ID MASK 位为 1 时，不比较该位的 ID CODE。

接收到的数据只要通过 8 组筛选器的任意一组，则被接收，接收到的数据存储在 RB 中，否则数据不被接收，也不被存储。

每组筛选器通过 ACFEN 寄存器使能或者禁止。ID CODE 和 ID MASK 通过 ACFCTRL 寄存器的 SELMASK 位设定，SELMASK=0 时，指向 ID CODE，SELMASK=1 时，指向 ID MASK。筛选器通过 ACFCTRL 寄存器的 ACFADR 位选择。ID CODE 和 ID MASK 通过 ACF 寄存器访问且只能在 CFG_STAT.RESET=1 即 CAN 软件复位时设定。ACF 寄存器访问筛选寄存器组的方式请参考下图。

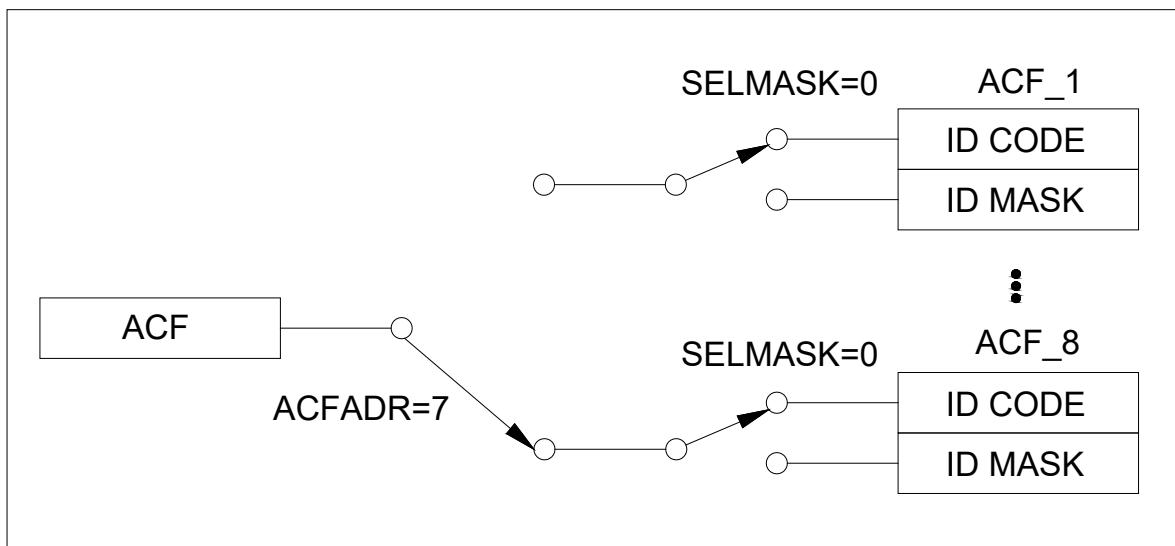


图 30-5 CAN ACF 寄存器访问筛选器组示意图

30.4.5 Data Transmission

在开始发送前必须保证 PTB 或者 STB 中至少有一帧数据已被装载，PTB 发送过程中 TPE 被锁定，STB 的填充情况可以通过 TSSTAT 位确认。发送数据设定步骤如下：

1. 设定 TBSEL 从 PTB 和 STB 中选择发送 BUF
2. 通过 TBUF 寄存器写需要发送的数据。
3. 如果选择的时 STB，设置 TSNEXT=1 以完成 STB SLOT 的装载。
4. 发送使能
 - PTB 发送使用 TPE
 - STB 发送使用 TSALL 或者 TSONE
5. 发送完成状态确认
 - PTB 发送完成使用 TPIF，TPIE 用于使能 TPIF
 - STB 采用 TSONE 发送完成时使用 TSIF，TSIE 用于使能 TSIF
 - STB 采用 TSALL 发送完成时使用 TSIF，此时需要设定的全部 STB SLOT 数据发送完成后，TSIF 才置位，TSIE 用于使能 TSIF

30.4.6 Single Data Transmission

不需要使用自动重新发送功能时，可以通过寄存器设定为单次发送模式，CFG_STAT 寄存器的 TPSS 位用于 PTB 的单次发送模式设定，TSSS 位用于 STB 的单次发送模式。数据成功发送时单次发送和正常发送模式时动作相同。但是数据没有成功发送时会出现以下结果：

- TPIF 置位 ($TPIE=1$)，对应的 BUF SLOT 数据会被清除。
- 有错误发送时，KOER 更新，BEIF 置位 ($BEIE=1$)。
- 仲裁失败，ALIF 置位 ($ALIE=1$)。

单次发送模式，不能单独依靠 TPIF 来判断发送完成，需要同 BEIF 和 ALIF 一起判断发送是否完成。

30.4.7 Cancel data sending

可以通过 TPA 或者 TSA 取消已请求但还没有被执行的数据发送。取消数据发送会出现以下几种情况：

- 仲裁中
 - 节点仲裁失败，则取消数据发送。
 - 节点仲裁成功，则继续发送。
- 数据发送中
 - 成功发送数据且收到 ACK，对应的标志和状态正常置位。数据发送不取消。
 - 成功发送数据但没有收到 ACK，数据发送取消，错误计数器增加。
 - TSALL=1 设定的发送数据，正在发送的 STB SLOT 数据正常发送，没有开始发送的 STB SLOT 被取消。

取消数据发送的结果有以下两种情况。

- TPA 释放 PTB，且使 TPE=0。
- TSA 释放一个 STB SLOT 或者全部 STB SLOT 取决是 TSONE 还是 TSALL 使能的发送。

30.4.8 Data Reception

接收筛选器组可以过滤掉不需要的接收数据，减少中断的发生和 RB 的读取，从而降低 CPU 负荷。接收数据设定步骤如下：

1. 设定筛选器组。
2. 设定 RFIE, RAIIE 和 AFWL。
3. 等待 RFIF 或者 RAIIF。
4. 通过 RBUF 从 RB FIFO 中读取最早接收到的数据。
5. 设置 RREL=1，选择下一个 RB SLOT。
6. 重复 4, 5 直到通过 RSTAT 确认 RB 为空。

30.4.9 Error Handling

CAN_CTRL 一方面可以自动处理部分错误，比如自动重发数据或者丢弃接收到含有错误的帧，另一方面通过中断将错误向 CPU 报告。

CAN 节点有以下三种错误状态：

- 错误主动：节点检测到错误时自动发送主动错误标志。
- 错误被动：节点检测到错误时自动发送被动错误标志。
- 节点关闭：关闭状态下此节点不再影响整个 CAN 网络。

CAN_CTRL 提供 TECNT 和 RECNT 两个计数器用于计数错误。TECNT 和 RECNT 计数器按照 CAN2.0B 协议规定的规则进行增减。另外提供可编程的 CAN 错误警告 LIMIT 寄存器用于产生错误中断通知 CPU。

CAN 通信过程中有以下 5 种错误类型，错误类型可以通过 EALCAP 寄存器的 KOER 位识别。

- 位错误
- 形式错误
- 填充错误
- 应答错误
- CRC 错误

30.4.10 Node shutdown

当发送错误数大于 255 时，CAN 节点自动进入节点关闭状态从而不参与 CAN 通信，直到返回到错误主动状态。可以通过 CFG_STAT 寄存器的 BUSOFF 位确认 CAN 节点关闭状态。BUSOFF 被置位的同时 EIF 中断产生。

CAN 从节点关闭状态恢复到错误主动状态有以下两种方法：

- 上电复位
- 接收到连续 128 个 11 位的隐形位序列（恢复序列）

节点关闭状态下，TECNT 值保持不变，RECNT 用于计数恢复序列。从节点关闭状态恢复后，TECNT 和 RECNT 被复位为 0。

节点关闭标志 BUSOFF 置位的同时，CFG_STAT 寄存器的 RESET 位也被置位。

30.4.11 Arbitration Failure Position Capture

CAN_CTRL 能够精确捕捉到仲裁失败位的位置并反映到 ALC 寄存器中。ALC 寄存器中保存着最近一次仲裁失败位的位置，如果节点赢得仲裁，则 ALC 位不更新。

ALC 值定义如下：

SOF 位后，第一个 ID 数据位 ALC 为 0，第二个 ID 数据位 ALC 为 1，依次类推。因为仲裁只发生在仲裁场内，所以 ALC 的最大值为 31。比如一个标准格式远程帧和一个扩展帧仲裁，扩展帧在 IDE 位失败，则 ALC=12。

30.4.12 Loopback Mode

CAN_CTRL 支持以下两种回环模式：

- 内部回环
- 外部回环

两种回环模式都可以接收自己发出的数据帧，主要用于测试用途。

内部回环模式，模块内部将接收数据线连接到发送数据线，并且发送数据不输出。内部回环模式下，节点会生成自应答信号以避免 ACK 错误。

外部回环模式保持和收发器的连接因此发送的数据仍能出现在 CAN 总线上，在收发器的帮助下，CAN 能收到自己发送的数据。外部回环模式可以通过 RCTRL 寄存器的 SACK 位来决定是否生成自应答信号，SACK=0 时，不生成自应答信号，SACK=1 时，生成自应答信号。

外部回环模式，SACK=0 时，会出现以下两种情况：

- 其它节点也收到本节点发送的数据帧并发送应答信号，该情况下本节点能够成功收发数据。
- 如果没有其它节点返回应答信号，则会产生应答错误，会重新发送数据并增加错误计数器。此时推荐采用单次发送模式。

从回环模式返回到正常模式时，除了清除模式位以外，还需要软件复位 CAN_CTRL。

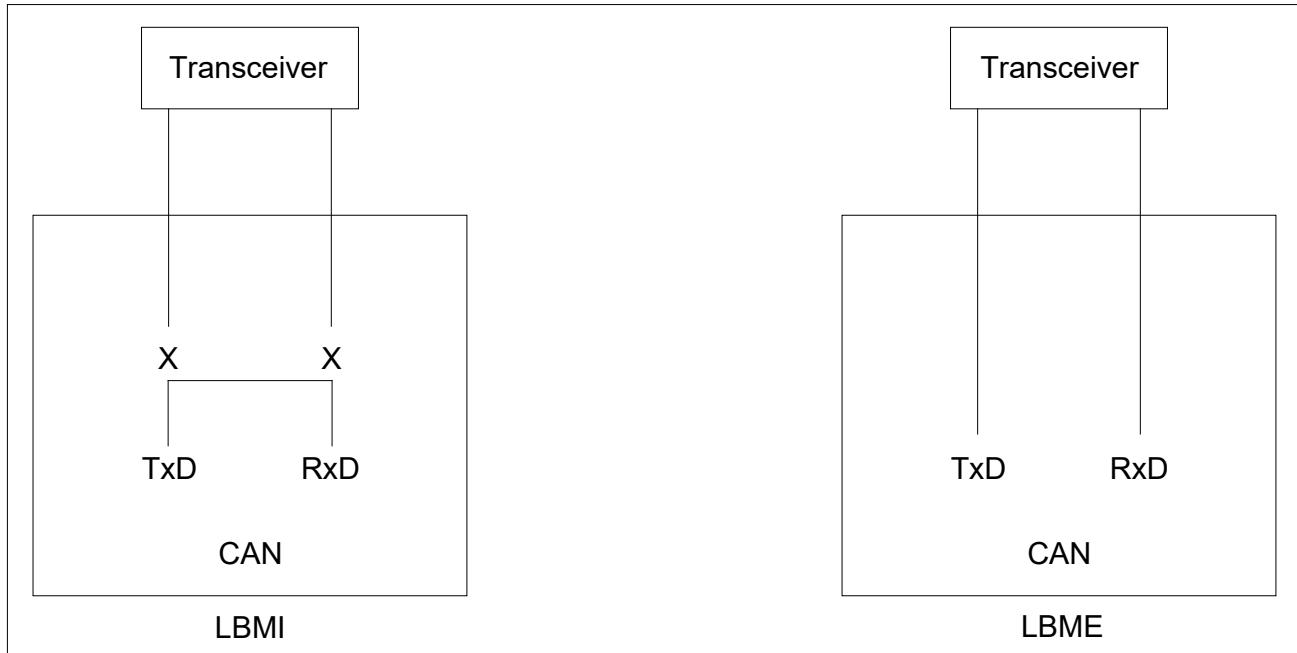


图 30-6 CAN LBMI 和 LBME 示意图

30.4.13 Silent Mode

静默模式可以用来监控 CAN 网络数据。在静默模式下，可以从 CAN 总线接收数据，不向总线发送任何数据。将 TCMD 寄存器中的 LOM 置 1，使 CAN 总线控制器进入静默模式，将其清 0 可以离开静默模式。

外部回环模式可以和静默模式组合成外部回环静默模式，此时 CAN 可以认为一个安静的接收者，但在必要的时候可以发送数据。外部回环静默模式下，帧包含自应答信号允许被发送，但是该节点不会产生错误标志和过载帧。

30.4.14 Software reset function

通过设定寄存器 CFG_STAT 寄存器的 RESET 位为 1, 实现软件复位功能, 软件复位功能的复位范围如下表所示。

表 30-3 软件复位范围表

寄存器位名	软件复位	备注	寄存器位名	软件复位	备注
ACFADR	否	-	EWL	是	-
ACODE	否	只能在软件复位时可以写	KOER	是	-
AE_x	否	-	LBME	是	-
AFWL	否	-	LBMI	是	-
AIF	是	-	RACTIVE	是	接收立即停止, 并不生成ACK
ALC	是	-	RAFIE	否	-
ALIE	否	-	RAFIF	是	-
ALIF	是	-	RBALL	是	-
AMASK	否	只能在软件复位时可以写	RBUF	是	所有RB被标记为空, 数值不定
BEIE	否	-	REF_ID	否	-
BEIF	是	-	REF_IDE	否	-
BUSOFF	否	通过写1清除	RFIE	否	-
EIE_F	否	-	RFIF	是	-
EIF	否	-	RIE	否	-
EPASS	否	-	RIF	是	-
EPIE	否	-	ROIE	否	-
EPIF	是	-	ROIF	是	-
EWARN	否	否	ROM	否	-

寄存器位名	软件复位	备注	寄存器位名	软件复位	备注
ROV	是	-	TSNEXT	是	-
RREL	是	-	TSONE	是	-
PRESC	否	只能在软件复位时可以写	TPIE	否	-
RSTAT	是	-	TPIF	是	-
SACK	是	-	TPSS	是	-
SELMASK	否	-	TSFF	是	所有STB SLOT被标记为空
SEG_1	否	只能在软件复位时可以写	TSIE	否	-
SEG_2	否	只能在软件复位时可以写	TSIF	是	-
SJW	否	只能在软件复位时可以写	TSSS	是	-
TACTIVE	是	发送立即停止	TSSTAT	是	所有STB SLOT被标记为空
TBE	是	-	TTEN	是 是	-
TBF	否	-	TTIF	是	-
TBPTR	否	-	TTIE	否	-
TBSEL	是	-	TTPTR	否	-
TBUF	是	所有的STB被标记为空，指向PTB	TTTBM	否	-
TECNT	否	通过BUSOFF=1清除	TTYPE	否	-
TEIF	是	-	TT_TRIG	否	-
TPA	是	-	TT_WTRIG	否	-
TPE	是	-	T_PRESC	否	-
TSA	是	-	WTIE	否	-
TSALL	是	-	WTIF	是	
TSMODE	否	-			

30.4.15 Upward compatibility with CAN-FD functions

CAN-CTRL 即使在包含 CAN-FD 网络中收到 CAN-FD 的帧，接收器会自动忽略这些帧，不返回 ACK，等到总线空闲时，再发送或者接收下一个 CAN2.0B 的帧。

30.4.16 Time-triggered TTCAN

CAN-CTRL 为 ISO11898-4 规定的时间触发通信方式提供部分(lever 1)硬件支持。本章节从以下 5 个部分介绍 TTCAN 功能。

30.4.16.1 TBUF Behavior in TTCAN Mode

TTTBM=1

TTTBM=1 时，PTB 和 STB SLOT 一样组成 TB SLOT，通过 TB PTR 寄存器指定发送 BUF，其中 TB PTR=0 时，指向 PTB，TB PTR=1 是指向 STB SLOT1，依次类推。主机可以通过 TPE 和 TPF 寄存器来标记发送 BUF SLOT。此时 TB SEL 和 TS NEXT 寄存器无任何意义从而可以被忽略。

TTTBM=1 时，PTB 不具有任何特殊的属性，和 STB SLOT 一样，传送完成标志也采用 TSIF。

TTCAN 模式时，发送 BUF 没有 FIFO 模式和优先级仲裁模式，同时也只有一个选定的 SLOT 可以发送数据。

TTCAN 模式下，传输开始需要采用时间触发方式，TPE，TS ONE，TS ALL，TPSS 和 TPA 被固定为 0 且被忽略。

TTTBM=0

TTTBM=0 时，组合使用事件驱动通信和接收时间戳功能。在该模式下，PTB 和 STB 的功能和 TTEN=0 时一致，因此 PTB 始终具有最高的优先级，而 STB 可以工作在 FIFO 模式或者仲裁模式。

30.4.16.2 TTCAN functional description

上电后，Time Master 需要根据 ISO 11898-4 协议进行初始化。一个 CAN 网络中，最多可以有 8 个潜在的 Time Master。每一个 Time Master 都具有自己的参考消息 ID（ID 最后 3 位）。这些潜在的 Time Master 根据自己的优先级发送各自的参考消息。

TTEN=1 后，16 位的计数器开始工作，当参考消息被成功接收或者 Time Master 成功发送参考消息时，CAN 控制器将 Sync_Mark 拷贝给 Ref_Mark，Ref_Mark 将 cycle time 设置为 0。成功接收参考消息置位 RIF 标志而成功发送参考消息置位 TPIF 标志或者 TSIF 标志。此时主机需要准备下一个动作的触发条件。

触发条件可以是接收触发。该触发仅触发中断可用于检测期待的消息没有被收到。

触发条件也可以是发送触发。该触发开始发送通过 TT PTR 寄存器指定的 TBUF SLOT 里的数据。如果选定的 TBUF SLOT 被标记为空，则不开始发送，但置位中断标志。

30.4.16.3 TTCAN Timing

CAN_CTRL 支持 ISO11898-4 level 1。包含的一个 16 位计数器工作在为 PRESC，SEG_1，SET_2 定义的位时间下。如果 TTEN=1，则有一个额外的预分频器 T_PRESC。

一帧数据的 SOF 时，计数器的值为 Sync_Mark。如果该帧数据为参考消息，则将 Sync_Mark 拷贝给 Ref_Mark。cycle time 等于计数器的值减去 Ref_Mark。该时间用作为接收消息的时间戳或者发送消息的触发时间基准。

30.4.16.4 TTCAN trigger mode

通过 TTTYPE 寄存器定义 TTCAN 的触发方式，TTPTR 寄存器指定发送 SLOT，而 TT_TRIG 指定触发器的 cycle time。

包含以下五种触发方式：

- 立即触发
- 时间触发
- 单次发送触发
- 发送开始触发
- 发送停止触发

除了立即触发方式外，所有的触发器都使用 TTIF 标志。TTTBM=1 时，只支持时间触发方式。

立即触发

通过写 TT_TRIG 的高位（不在意写入的值），启动触发器。此模式下，TTPTR 选定的 TBUF SLOT 内的数据会立即发送。TTIF 不置位。

时间触发

时间触发方式仅通过置位 TTIF 标志产生中断，并无其他功能。如果一个节点期待在特定的时间窗口内收到期待的数据，则可以使用时间触发方式。如果 TT_TRIG 值小于实际的 cycle time，则 TEIF 置位且无其它动作。

单次发送触发

单次发送触发方式用于在执行时间窗口内发送数据。此时，忽略 TSSS 位。

通过 TEW 位设定 ISO11898-4 规定的最多 16 个 cycle time 的 Tick，设定范围为 1~16。如果在规定的发送使能时间窗口内数据没有开始发送，则帧被丢弃。对应的发送 BUF SLOT 被标记为空，并且置位 AIF，对应的发送 BUF 内的数据不会被改写，因为可以通过置位 TPF 再次发送。

如果 TT_TRIG 值小于实际的 cycle time，则 TEIF 置位且无其它动作。

发送开始触发

发送开始触发方式用于仲裁时间窗口内，参与仲裁。TSSS 用于决定是否自动重发或者单次发送模式。如果指定的消息没有被成功发送，可以使用发送停止触发来停止该发送。

如果 TT_TRIG 值小于实际的 cycle time，则 TEIF 置位且无其它动作。

发送停止触发

发送停止触发方式用于停止通过发送开始触发方式已经开始的发送。如果发送被停止，则发送帧被舍弃，置位 AIF 并将选定的 TBUF SLOT 标记为空，但 TBUF SLOT 内的数据不会被改写，可以通过置位 TPF 就可以再次发送。

如果 TT_TRIG 值小于实际的 cycle time 则 TEIF 置位且执行停止。

30.4.16.5 TTCAN trigger watchdog time

TTCAN 触发看门时间功能类似于看门狗功能，在 TTTBM=1 时使用。用来看门从上次成功接收到参考消息开始的时间。参考消息可以在周期 cycle time 中或者一个事件后被接收，应用程序应该根据具体情况设定合适的看门时间。

如果 cycle count 等于 TT_WTRIG，则置位 WTIF。通过 WTIE 写 0，关闭看门触发。

如果 TT_WTRIG 比实际的 cycle time 小，则 TEIF 置位。

30.4.17 Interrupts

表 30-4 CAN 中断表

中断标志	描述
RIF	接收中断
ROIF	接收上溢中断
ROIF	接收BUF满中断
RAFIF	接收BUF将满中断
TPIF	PTB发送中断
TSIF	STB发送中断
EIF	错误中断
AIF	取消发送中断
EPIE	错误被动中断
ALIF	仲裁失败中断
BEIF	总线错误中断
WTIF	触发看门中断
TEIF	触发错误中断
TTIF	时间触发中断

30.5 Register Description

CAN_BASE_ADDR: 0x4003_0000

表 30-5 CAN 寄存器一览表

寄存器名	符号	偏移地址	位宽	复位值
CAN接收BUF寄存器	CAN_RBUF	0x00~0x0F	128	0XXXXX XXXXX
CAN发送BUF寄存器	CAN_TBUF	0x50~0x5F	128	0XXXXX XXXXX
CAN配置和状态寄存器	CAN_CFG_STAT	0xA0	8	0x80
CAN命令寄存器	CAN_TCMD	0xA1	8	0x00
CAN发送控制寄存器	CAN_TCTRL	0xA2	8	0x90
CAN接收控制寄存器	CAN_RCTRL	0xA3	8	0x00
CAN接收和发送中断使能寄存器	CAN_RTIE	0xA4	8	0xFE
CAN接收和发送中断标志寄存器	CAN_RTIF	0xA5	8	0x00
CAN错误中断使能和标志寄存器	CAN_ERRINT	0xA6	8	0x00
CAN警告限定寄存器	CAN_LIMIT	0xA7	8	0x1B
CAN位时序寄存器	CAN_BT	0xA8	32	0x0102 0203
CAN错误和仲裁失败捕捉寄存器	CAN_EALCAP	0xB0	8	0x00
CAN接收错误计数器寄存器	CAN_RECNT	0xB2	8	0x00
CAN发送错误计数器寄存器	CAN_TECNT	0xB3	8	0x00
CAN筛选器组控制寄存器	CAN_ACFCTRL	0xB4	8	0x00
CAN筛选器组使能寄存器	CAN_ACFEN	0xB6	8	0x01
CAN筛选器组code和mask寄存器	CAN_ACF	0xB8	32	0XXXXX XXXXX
TTCAN TB slot指针寄存器	CAN_TBSLOT	0xBE	8	0x00
TTCAN 时间触发配置寄存器	CAN_TTCFG	0xBF	8	0x90
TTCAN 参考消息寄存器	CAN_REF_MSG	0xC0	32	0XXXXX XXXXX
TTCAN 触发配置寄存器	CAN_TRG_CFG	0xC4	16	0x0000
TTCAN 触发时间寄存器	CAN_TRG_TRIG	0xC6	16	0x0000
TTCAN 触发看门时间寄存器	CAN_TRG_WTRIG	0xC8	16	0x0000

表 30-6 CAN 寄存器 BYTE/HALFWORD/WORD 访问排布表

地址	BYTE访问	HALFWORD访问		WORD访问								
0x00~0x0F	CAN_RBUF	CAN_RBUF		CAN_RBUF								
0x50~0x5F	CAN_TBUF	CAN_TBUF		CAN_TBUF								
0xA0	CAN_CFG_STAT	CAN_TCMD	CAN_CFG_STAT	CAN_RCTRL	CAN_TCTRL	CAN_TCMD	CAN_CFG_STAT					
0xA1	CAN_TCMD	-		-								
0xA2	CAN_TCTRL	CAN_RCTRL	CAN_TCTRL	-								
0xA3	CAN_RCTRL	-		-								
0xA4	CAN_RTIE	CAN_RTIF	CAN_RTIE	CAN_LIMIT	CAN_ERRINT	CAN_RTIF	CAN_RTIE					
0xA5	CAN_RTIF	-		-								
0xA6	CAN_ERRINT	CAN_LIMIT	CAN_ERRINT	-								
0xA7	CAN_LIMIT	-		-								
0xA8	CAN_BT[7:0]	CAN_BT[15:0]		CAN_BT								
0xA9	CAN_BT[15:8]	-		-								
0xAA	CAN_BT[23:16]	-		-								
0xAB	CAN_BT[31:24]	CAN_BT[31:16]		-								
0xB0	-	-		CAN_TECNT	CAN_RECNT	-						
0xB1	-	-		-								
0xB2	CAN_RECNT	CAN_TECNT	CAN_RECNT	-								
0xB3	CAN_TECNT	-		-								
0xB4	CAN_ACFCTRL[7:0]	CAN_ACFCTRL		CAN_ACFEN	CAN_ACFCTRL							
0xB5	CAN_ACFCTRL[15:8]	-		-								
0xB6	CAN_ACFEN[7:0]	CAN_ACFEN		-								
0xB7	CAN_ACFEN[15:8]	-		-								
0xB8	CAN_ACF	CAN_ACF		CAN_ACF								
0xBC	-	-		CAN_TTCFG	CAN_TBSLOT	-	-					
0xBD	-	-		-								
0xBE	CAN_TBSLOT	CAN_TTCFG	CAN_TBSLOT	-								
0xBF	CAN_TTCFG	-		-								
0xC0	CAN_REF_MSG[7:0]	CAN_REF_MSG[15:0]		CAN_REF_MSG								
0xC1	CAN_REF_MSG[15:8]	-		-								
0xC2	CAN_REF_MSG[23:16]	CAN_REF_MSG[31:16]		-								
0xC3	CAN_REF_MSG[31:24]	-		-								
0xC4	CAN_TRG_CFG[7:0]	CAN_TRG_CFG		CAN_TRG_TRIG	CAN_TRG_CFG							
0xC5	CAN_TRG_CFG[15:8]	-		-								
0xC6	CAN_TRG_TRIG[7:0]	CAN_TRG_TRIG		-								
0xC7	CAN_TRG_TRIG[15:8]	-		-								
0xC8	CAN_TRG_WTRIG[7:0]	CAN_TRG_WTRIG		-	CAN_TRG_WTRIG							
0xC9	CAN_TRG_WTRIG[15:8]	-		-								

30.5.1 CAN Receive BUF Registers (CAN_RBUF)

CAN Receive Buffer Registers

偏移地址：0x00~0x0F

复位值：0XXXX XXXX

RBUF 寄存器指向最早接收到的 CAN 邮箱的 RB SLOT 地址，RBUF 寄存器可以按照任意顺序读取。

KOER 位即为寄存器 EALCAP.KOER，仅在 RBALL=1 时有意义。

TX 位表示在回环模式下接受到自己发送的邮箱。

CYCLE_TIME 位仅在 TTCAN 模式时有效，表示 SOF 开始时的 cycle time。

CAN 接收邮箱的数据格式如下：

表 30-7 标准格式 CAN 接收邮箱格式

地址	b7	b6	b5	b4	b3	b2	b1	b0	功能
RBUF	ID[7:0]								ID
RBUF+1	-				ID[10:8]				ID
RBUF+2	-								ID
RBUF+3	-								ID
RBUF+4	IDE=0	RTR	0	0	DLC[3:0]				Control
RBUF+5	KOER[2:0]			TX	-				Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN
RBUF+7	CYCLE_TIME[15:8]								TTCAN
RBUF+8	DATA1								Data
RBUF+9	DATA2								Data
RBUF+10	DATA3								Data
RBUF+11	DATA4								Data
RBUF+12	DATA5								Data
RBUF+13	DATA6								Data
RBUF+14	DATA7								Data
RBUF+15	DATA8								Data

表 30-8 扩展格式 CAN 接收邮箱格式

地址	b7	b6	b5	b4	b3	b2	b1	b0	功能
RBUF	ID[7:0]								ID
RBUF+1	ID[15:8]								ID
RBUF+2	ID[10:8]								
RBUF+2	ID[23:16]								ID
RBUF+3	-			ID[28:24]					ID
RBUF+4	IDE=1	RTR	0	0	DLC[3:0]				Control
RBUF+5	KOER[2:0]			TX	-				Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN
RBUF+7	CYCLE_TIME[15:8]								TTCAN
RBUF+8	DATA1								Data
RBUF+9	DATA2								Data
RBUF+10	DATA3								Data
RBUF+11	DATA4								Data
RBUF+12	DATA5								Data
RBUF+13	DATA6								Data
RBUF+14	DATA7								Data
RBUF+15	DATA8								Data

控制位含义如下：

IDE(IDentifier Extension):

0：标准格式

1：扩展格式

RTR(Remote Transmission Request)

0：数据帧

1：远程帧

DLC(Data Length Code):

数据长度码，设定范围为 0~8，对应数据长度为 0Byte~8Byte

30.5.2 CAN Transmit BUF Registers (CAN_TBUF)

CAN Transmit Buffer Registers

偏移地址：0x50~0x5F

复位值：0xFFFF XXXX

TBUF 寄存器指向下一个空的 CAN 发送 BUF SLOT，TBUF 寄存器可以按照任意顺序读取。通过软件将 TSNEXT 写 1 来标记对应的 TBUF SLOT 已经写入数据，从而指向下一个 TBUF SLOT。

TBUF 只能 WORD 访问。

CAN 发送邮箱的数据格式如下：

表 30-9 标准格式 CAN 发送邮箱格式

地址	b7	b6	b5	b4	b3	b2	b1	b0	功能
TBUF	ID[7:0]								ID
TBUF+1	-				ID[10:8]				ID
TBUF+2	-								ID
TBUF+3	-								ID
TBUF+4	IDE=0	RTR	0	0	DLC[3:0]				Control
TBUF+5	-	TX	TX	-					-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
TBUF+11	DATA4								Data
TBUF+12	DATA5								Data
TBUF+13	DATA6								Data
TBUF+14	DATA7								Data
TBUF+15	DATA8								Data

表 30-10 扩展格式 CAN 发送邮箱格式

地址	b7	b6	b5	b4	b3	b2	b1	b0	功能
TBUF	ID[7:0]								ID
TBUF+1	ID[15:8] ID[10:8]								ID
TBUF+2	ID[23:16]								ID
TBUF+3	-	ID[28:24]							ID
TBUF+4	IDE=0	RTR	0	0	DLC[3:0]				Control
TBUF+5	-	TX	TX	-	-	-	-	-	-
TBUF+6	-	-	-	-	-	-	-	-	-
TBUF+7	-	-	-	-	-	-	-	-	-
TBUF+8	DATA1	-	-	-	-	-	-	-	Data
TBUF+9	DATA2	-	-	-	-	-	-	-	Data
TBUF+10	DATA3	-	-	-	-	-	-	-	Data
TBUF+11	DATA4	-	-	-	-	-	-	-	Data
TBUF+12	DATA5	-	-	-	-	-	-	-	Data
TBUF+13	DATA6	-	-	-	-	-	-	-	Data
TBUF+14	DATA7	-	-	-	-	-	-	-	Data
TBUF+15	DATA8	-	-	-	-	-	-	-	Data

控制位含义如下：

IDE(IDentifier Extension):

0：标准格式

1：扩展格式

RTR(Remote Transmission Request)

0：数据帧

1：远程帧

DLC(Data Length Code):

数据长度码，设定范围为 0~8，对应数据长度为 0Byte~8Byte

30.5.3 CAN Configuration and Status Register (CAN_CFG_STAT)

CAN Configuration and Status Register

偏移地址：0xA0

复位值：0x80

b7	b6	b5	b4	b3	b2	b1	b0
RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF

位	标记	位名	功能	读写
b7	RESET	复位请求	复位请求位 0：不请求局部复位 1：请求局部复位 部分寄存器只能在RESET=1时进行写操作，具体请参考软件复位功能，当该节点进入BUS OFF状态时，硬件自动将RESET位置1。请注意，当RESET=0后需要11个CAN bit time该节点才能参与通信。	R/W
b6	LBME	外部回环模式使能位	外部回环模式使能位 0：禁止外部回环模式 1：使能外部回环模式 注意：通信中禁止设定该位。	R/W
b5	LBMI	内部回环模式使能位	内部回环模式使能位 0：禁止内部回环模式 1：使能内部回环模式 注意：通信中禁止设定该位。	R/W
b4	TPSS	PTB单次传输模式	PTB单次传输模式 0：禁止PTB单次传输模式 1：使能PTB单次传输模式	R/W
b3	TSSS	STB单次传输模式	STB单次传输模式 0：禁止STB单次传输模式 1：使能STB单次传输模式	R/W
b2	RACTIVE	接收中状态信号	接收中状态信号 0：非接收中 1：接收中	R
b1	TACTIVE	发送中状态信号	发送中状态信号 0：非发送中 1：发送中	R
b0	BUSOFF	总线关闭状态	总线关闭状态 0：总线有效状态 1：总线关闭状态 注意：写1可以清零TECNT和RECNT寄存器，但仅限用于调试用途。	R/W

30.5.4 CAN Command Register (CAN_TCMD)

CAN Command Register

偏移地址：0xA1

复位值：0x00

b7	b6	b5	b4	b3	b2	b1	b0
TBSEL	LOM	STBY	TPE	TPA	TSONE	TSALL	TSA

位	标记	位名	功能	读写
b7	TBSEL	发送BUF选择位	发送BUF选择位 (Transmit Buffer Select) 0: PTB 1: STB 当TTEN=1&TTTBM=1时，TBSEL被复位成复位值。 注意：写TBUF寄存器或者TSNEXT位时，此位需要保持定值。	R/W
b6	LOM	静默模式使能位	静默模式使能位 (Listen Only Mode) 0: 禁止静默模式 1: 使能静默模式 LOM=1&LBME=0时禁止发送。 LOM=1&LBME=1时禁止应答相应接收到的帧以及错误帧，但可以发送数据。 注意：通信中禁止设定该位。	R/W
b5	STBY	STBY设定位	STBY设定位 0 : 清除STBY 1 : 置位STBY (置位前必须保证TPE,TSONE,TSALL都为0)	R/W
b4	TPE	PTB发送使能位	PTB发送使能位 (Transmit Primary Enable) 0: 禁止PTB发送 1: 使能PTB发送 此位使能后，PTB中的Mailbox将在下一个可以发送的位置被发送。已经开始的STB发送将继续，但是下一个等待的STB发送会被延迟到PTB发送完成后再进行。 该位写1后将保持为1直到PTB发送完成或者通过TPA取消发送。软件不能通过写0清除该位。 以下情况TPE被硬件复位成复位值： - RESET=1 - BUSOFF=1 - LOM=1&LBME=0 - TTEN=1&TTTBM=1	R/W

位	标记	位名	功能	读写
b3	TPA	PTB 发送取消位	<p>PTB发送取消位 (Transmit Primary Abort)</p> <p>0: 不取消 1: 取消已经通过TPE置1请求但还未开始的PTB发送</p> <p>该位软件写1但是通过硬件清零。通过写1可以清零TPE位，因此和TPE同时写1。</p> <p>以下情况TPE被硬件复位成复位值：</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 - TTEN=1&TTTB=1 	R/W
b2	TSONE	发送一帧STB数据设定位	<p>发送一帧STB数据设定位 (Transmit Secondary ONE frame)</p> <p>0: 不发送 1: 发送一帧STB数据</p> <p>FIFO模式中，发送最早写入的数据，优先级模式里发送最高优先级的数据</p> <p>该位写1后将保持为1直到STB发送完成或者通过TSA取消发送。软件不能通过写0清除该位。</p> <p>以下情况TSONE被硬件复位成复位值：</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 - LOM=1&LBME=0 - TTEN=1&TTTB=1 	R/W
b1	TSALL	发送所有的STB数据设定位	<p>发送所有的STB数据设定位 (Transmit Secondary ALL frame)</p> <p>0: 不发送 1: 发送STB中所有的数据</p> <p>该位写1后将保持为1直到STB发送完成或者通过TSA取消发送。软件不能通过写0清除该位。</p> <p>以下情况TSALL被硬件复位成复位值：</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 - LOM=1&LBME=0 - TTEN=1&TTTB=1 	R/W
b0	TSA	STB发送取消位	<p>STB发送取消位 (Transmit Secondary Abort)</p> <p>0: 不取消 1: 取消已经通过TSONE或者TSALL置1请求但还未开始的STB发送</p> <p>该位通过软件写1但是通过硬件清零。写1可以清零TSONE或者TSALL位。</p> <p>以下情况TSA被硬件复位成复位值：</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 	R/W

30.5.5 CAN Transmit Control Register (CAN_TCTRL)

CAN Transmit Control Register

偏移地址：0xA2

复位值：0x90

b7	b6	b5	b4	b3	b2	b1	b0
-	TSNEXT	TSMODE	TTTBM	-	-	TSSTAT[1:0]	

位	标记	位名	功能	读写
b7	Reserved	-	必须保持复位值。	R
b6	TSNEXT	下一个 STB SLOT	下一个STB (Transmit buffer Secondary NEXT) 0: 无动作 1: 当前STB SLOT已填充，指向下一个SLOT 应用程序将TBUF中的数据写完后，通过置位TSNEXT位标识当前STB SLOT已经被填充，从而硬件将TBUF指向下一个STB SLOT。 被TSNEXT位标识的STB SLOT中的数据可以通过TSONE或者TSALL位发送。 该位通过应用程序写1，硬件清零。 所有的STB SLOT被填满后，TSNEXT保持为1直到有STB SLOT被释放。 注意：TTCAN模式时此位固定为0。	R/W
b5	TSMODE	STB发送模式	STB发送模式 (Transmit buffer Secondary operation MODE) 0: FIFO模式 1: 优先级模式 FIFO模式根据数据帧写入的先后顺序发送。 优先级模式根据ID自动判断，ID越小，优先级越高。 无论何种模式，PTB具有最高的优先级。 注意：TSMODE位只能在STB空时设定。	R/W
b4	TTTBM	TTCAN BUF 模式	TTCAN BUF模式 (TTCAN Transmit Buffer Mode) TTEN=0时，TTTBM被忽略。 0: TSMODE决定，PTB和STB 1: 通过TBPTR和TTPTR设定 TTCAN模式时，只需要接收时间戳功能时，此位可以设置为0，通过TSMODE决定使用PTB还是STB。 注意：TSMODE位只能在STB空时设定。	R/W
b3~b2	Reserved	-	必须保持复位值。	R

位	标记	位名	功能	读写
b1~b0	TSSTAT	STB状态	STB状态 (Transmission Secondary STATus bits) TTEN=0 00: STB空 01: STB小于等于半满 10: STB大于半满 11: STB满 TTEN=0&TTTBM=1 00: PTB和STB空 01: PTB和STB非满 10: 保留 11: PTB和STB满	R

30.5.6 CAN Receive Control Register (CAN_RCTRL)

CAN Receive Control Register

偏移地址：0xA3

复位值：0x00

b7	b6	b5	b4	b3	b2	b1	b0
SACK	ROM	ROV	RREL	RBALL	-	RSSTAT[1:0]	

位	标记	位名	功能	读写
b7	SACK	自应答	自应答 (Self-ACKnowledge) 0: 无自应答 1: LBME=1时, 使能自应答功能	R/W
b6	ROM	接收BUF上溢模式设定位	接收BUF上溢模式设定位 (Receive buffer Overflow Mode) 0: 最早接收到的数据被覆盖 1: 新接收到的数据不被存储	R/W
b5	ROV	接收BUF上溢标志位	接收BUF上溢标志位 (Receive buffer OVerflow) 0: 无上溢 1: 上溢, 最少有一个数据丢失 通过写RREL为1清零。	R
b4	RREL	释放接收BUF	释放接收BUF (Receive buffer RELease) 0: 不释放 1: 表示该接收BUF已经被读取过, RBUF寄存器指向下一个RB SLOT。	R/W
b3	RBALL	接收BUF数据存储所有数据帧	接收BUF数据存储所有数据帧 (Receive Buffer stores ALL data frames) 0: 正常模式 1: 存储所有的数据包括有错误的数据。	R/W
b2	Reserved	-	必须保持复位值。	R
b1~b0	RSTAT	接收BUF状态	接收BUF状态 (Receive buffer STATus) 00: RBUF空 01: RBUF非空但小于AFWL编程值 10: RBUF大于等于AFWL编程值但未满 11: 满 (上溢时保持此值)	R

30.5.7 CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE)

CAN Receive and Transmit Interrupt Enable Register

偏移地址：0xA4

复位值：0xFE

b7	b6	b5	b4	b3	b2	b1	b0
RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF

位	标记	位名	功能	读写
b7	RIE	接收中断使能	接收中断使能 (Receive Interrupt Enable) 0: 禁止 1: 使能	R/W
b6	ROIE	接收上溢中断使能	接收上溢中断使能 (Receive Overrun Interrupt Enable) 0: 禁止 1: 使能	R/W
b5	RFIE	接收BUF满中断使能	接收BUF满中断使能 (RB Full Interrupt Enable) 0: 禁止 1: 使能	R/W
b4	RAFIE	接收BUF将满中断使能	接收BUF将满中断使能 (RB Almost Full Interrupt Enable) 0: 禁止 1: 使能	R/W
b3	TPIE	PTB发送中断使能	PTB发送中断使能 (Transmission Primary Interrupt Enable) 0: 禁止 1: 使能	R/W
b2	TSIE	STB发送中断使能	STB发送中断使能 (Transmission Secondary Interrupt Enable) 0: 禁止 1: 使能	R/W
b1	EIE	错误中断使能	错误中断使能 (Error Interrupt Enable) 0: 禁止 1: 使能	R/W
b0	TSFF	发送BUF满标志	TTEN=0 or TTTBM=0: STB满标志 (Transmit Secondary buffer Full Flag) 0: STB SLOT没有被全部填充 1: STB SLOT被全部填充 TTEN=1 and TTTBM=1: TB满标志 (Transmit buffer Full Flag) 0: TBPTR选择的发送BUF没有被全部填充 1: TBPTR选择的发送BUF被全部填充	

30.5.8 CAN Receive and Transmit Interrupt Status Register (CAN_RTIF)

CAN Receive and Transmit Interrupt Status Register

偏移地址：0xA5

复位值：0x00

b7	b6	b5	b4	b3	b2	b1	b0
RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF

位	标记	位名	功能	读写
b7	RIF	接收中断标志	接收中断标志 (Receive Interrupt Flag) 0: 未收到数据帧 1: 接收到有效的数据帧或者远程帧 通过应用程序写1清0。	R/W
b6	ROIF	接收上溢中断标志	接收上溢中断标志 (Receive Overrun Interrupt Flag) 0: 未发生接收上溢 1: 发生接收上溢 上溢时ROIF和RFIF同时置1。 通过应用程序写1清0。	R/W
b5	RFIF	接收BUF满中断标志	接收BUF满中断标志 (RB Full Interrupt Flag) 0: RB FIFO未满 1: RB FIFO满 通过应用程序写1清0。	R/W
b4	RAFIF	接收BUF将满中断标志	接收BUF将满中断标志 (RB Almost Full Interrupt Flag) 0: 被填充的RB SLOT数目小于AFWL设定值 1: 被填充的RB SLOT数目大于等于AFWL设定值 通过应用程序写1清0。	R/W
b3	TPIF	PTB发送中断标志	PTB发送中断标志 (Transmission Primary Interrupt Flag) 0: 没有PTB发送完成 1: 请求的PTB发送成功完成 通过应用程序写1清0。 注意：TTCAN模式时，TPIF无效，仅适用TSIF标志	R/W
b2	TSIF	STB发送中断标志	STB发送中断标志 (Transmission Secondary Interrupt Flag) 0: 没有STB发送完成 1: 请求的STB发送成功完成 通过应用程序写1清0。 注意：TTCAN模式时，TPIF无效，仅使用TSIF标志	R/W

位	标记	位名	功能	读写
b1	EIF	错误中断标志	<p>错误中断标志 (Error Interrupt Flag)</p> <p>0: BUSOFF位未发生变化，或者错误计数器的值与ERROR warning limit设定值的相对关系未发生变化。</p> <p>1: BUSOFF位发生变化，或者错误计数器的值与ERROR warning limit设定值的相对关系发生变化。比如错误计数器的值从小于设定值变为大于设定值，或者从大于设定值变为小于设定值。</p> <p>通过应用程序写1清0。</p>	R/W
b0	AIF	取消发送中断标志	<p>取消发送中断标志 (Abort Interrupt Flag)</p> <p>0: 未取消发送数据</p> <p>1: 通过TPA和TSA请求的发送消息被成功取消。</p> <p>通过应用程序写1清0。</p>	R/W

30.5.9 CAN Error Interrupt Enable and Flag Register (CAN_ERRINT)

CAN ERRor INTerrupt Enable and Flag Register

偏移地址：0xA6

复位值：0x00

b7	b6	b5	b4	b3	b2	b1	b0
EWARN	EPASS	WPIE	EPIF	ALIE	ALIF	BEIE	BEIF

位	标记	位名	功能	读写
b7	EWARN	到达设定的 ERROR WARNING LIMIT	到达设定的ERROR WARNING LIMIT (Error WARNING limit reached) 0: RECNT或者TECNT小于EWL设定值 1: RECNT或者TECNT大于等于EWL设定值 通过应用程序写1清0。	R/W
b6	EPASS	错误被动	错误被动 (Error Passive mode active) 0: 节点是主动错误节点 1: 节点时被动错误节点 通过应用程序写1清0。	R
b5	EPIE	错误被动中 断使能	错误被动中断使能 (Error Passive Interrupt Enable) 0: 禁止 1: 使能	R/W
b4	EPIF	错误被动中 断标志	错误被动中断标志 (Error Passive Interrupt Flag) 0: 未发生错误主动到错误被动或者错误被动到错误主动的变化 1: 发生错误主动到错误被动或者错误被动到错误主动的变化 通过应用程序写1清0。	R/W
b3	ALIE	仲裁失败中 断使能	仲裁失败中断使能 (Arbitration Lost Interrupt Enable) 0: 禁止 1: 使能	R/W
b2	ALIF	仲裁失败中 断标志	仲裁失败中断标志 (Arbitration Lost Interrupt Flag) 0: 仲裁成功 1: 仲裁失败 通过应用程序写1清0。	R/W
b1	BEIE	总线错误中 断使能	总线错误中断使能 (Bus Error Interrupt Enable) 0: 禁止 1: 使能	R/W
b0	BEIF	总线错误中 断标志	总线错误中断标志 (Bus Error Interrupt Flag) 0: 无总线错误 1: 总线错误 通过应用程序写1清0。	R/W

30.5.10 CAN Bit Timing Register (CAN_BT)

CAN Bit Timing Register

偏移地址：0xA8

复位值：0x0102 0203

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
				PRESC[7:0]				-				SJW[6:0]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-				SEG_2[6:0]								SEG_1[7:0]			

位	标记	位名	功能	读写
b31~b24	PRESC	预分频设定	预分频设定 (Prescaler) 此寄存器设定值时将模块BCLK设定位 (PRESC+1) 分频作为TQ的时钟。	R/W
b23	Reserved	-	必须保持复位值。	R
b22~b16	SJW	再同步补偿宽度时间设定	再同步补偿宽度时间设定 (Bit Timing Segment 2) 再同步补偿宽度时间=(SJW+1)*TQ	R/W
b15	Reserved	-	必须保持复位值。	R
b14~b8	SEG_2	位段2时间设定	位段2时间单元设定 (Bit Timing Segment 2) 位段2时间=(SEG_2+1)*TQ	R/W
b7~b0	SEG_1	位段1时间设定	位段1时间单元设定 (Bit Timing Segment 1) 位段1时间=(SEG_1+2)*TQ	R/W

30.5.11 CAN Error and Arbitration Loss Capture Register (CAN_EALCAP)

CAN Error and Arbitration Lost Capture Register

偏移地址：0xB0

复位值：0x00

b7	b6	b5	b4	b3	b2	b1	b0
KOER[2:0]						ALC[4:0]	

位	标记	位名	功能	读写
b7~b5	KOER	错误类别	错误类别 (Kind Of Error) 000: 无错误 001: 位错误 010: 形式错误 011: 填充错误 100: 应答错误 101: CRC错误 110: 其他错误 111: 保留 有错误时KOER位更新，正常发送接收时KOER位保持不变。	R
b4~b0	ALC	仲裁失败位置 捕捉	仲裁失败位置捕捉 (Arbitration Lost Capture) 仲裁失败时ALC记录一帧数据中仲裁失败时的位置。	R/W

30.5.12 CAN Warning Limit Register (CAN_LIMIT)

CAN Warning Limits Register

偏移地址：0xA7

复位值：0x1B

b7	b6	b5	b4	b3	b2	b1	b0
AFWL[3:0]				EWL[3:0]			

位	标记	位名	功能	读写
b7~b4	AFWL	接收BUF将满 Warning Limit	接收BUF将满Warning Limit (receive buffer Almost Full Warning Limit) 设定值范围为1~10。 AFWL=0无意义，当做AFWL=1处理。	R/W
b3~b0	EWL	Error Waring Limit编程值	Error Waring Limit编程值 (Programmable Error Warning Limit) Error Waring Limit= (EWL+1) *8。 该寄存器设定值影响EIF标志。	R/W

30.5.13 CAN Receive Error Counter Register (CAN_RECNT)

CAN Receive Error CouNT Register

偏移地址：0xB2

复位值：0x00

b7	b6	b5	b4	b3	b2	b1	b0
RECNT[7:0]							

位	标记	位名	功能	读写
b7~b0	RECNT	接收错误计数器	接收错误计数器 (Receive Error CouNT) 接收错误计数器根据CAN协议规定的错误计数增加或者减少。该计数器不存在上溢，255为最大值。	R/W

30.5.14 CAN Transmit Error Counter Register (CAN_TECNT)

CAN Transmit Error CouNT Register

偏移地址：0xB3

复位值：0x00

b7	b6	b5	b4	b3	b2	b1	b0
TECNT[7:0]							

位	标记	位名	功能	读写
b7~b0	TECNT	发送错误计数器	发送错误计数器 (Transmit Error CouNT) 发送错误计数器根据CAN协议规定的错误计数增加或者减少。该计数器不存在上溢，255为最大值。	R/W

30.5.15 CAN Filter Group Control Register (CAN_ACFCTRL)

CAN Acceptance Filter Control Register

偏移地址：0xB4

复位值：0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	SELMASK	-	-	-	ACFADR	-	-

位	标记	位名	功能	读写
b7~b6	Reserved	-	必须保持复位值。	R
b5	SELMASK	选择筛选器的屏蔽寄存器	选择筛选器的屏蔽寄存器 (SElect acceptance MASK) 0: ACF指向筛选器ID寄存器 1: ACF指向筛选器MASK寄存器 通过ACFADR选择具体的筛选寄存器组	R/W
b4	Reserved	-	必须保持复位值。	R
b3~b0	ACFADR	筛选器地址	筛选器地址 (acceptance filter address) ACFADR指向具体的筛选器，通过SELMASK去区分ID和MASK。 0000: 指向ACF_1 0001: 指向ACF_2 0010: 指向ACF_3 0011: 指向ACF_4 0100: 指向ACF_5 0101: 指向ACF_6 0110: 指向ACF_7 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111: 指向ACF_8	

30.5.16 CAN Filter Group Enable Register (CAN_ACFEN)

CAN Acceptance Filter Enable Register

偏移地址：0xB6

复位值：0x01

b7	b6	b5	b4	b3	b2	b1	b0
AE_8	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1

位	标记	位名	功能	读写
b7	AE_8	ACF_8使能	ACF_8使能 (Acceptance Filter 8 Enable) 0: 禁止 1: 使能	R/W
b6	AE_7	ACF_7使能	ACF_7使能 (Acceptance Filter 7 Enable) 0: 禁止 1: 使能	R/W
b5	AE_6	ACF_6使能	ACF_6使能 (Acceptance Filter 6 Enable) 0: 禁止 1: 使能	R/W
b4	AE_5	ACF_5使能	ACF_5使能 (Acceptance Filter 5 Enable) 0: 禁止 1: 使能	R/W
b3	AE_4	ACF_4使能	ACF_4使能 (Acceptance Filter 4 Enable) 0: 禁止 1: 使能	R/W
b2	AE_3	ACF_3使能	ACF_3使能 (Acceptance Filter 3 Enable) 0: 禁止 1: 使能	R/W
b1	AE_2	ACF_2使能	ACF_2使能 (Acceptance Filter 2 Enable) 0: 禁止 1: 使能	R/W
b0	AE_1	ACF_1使能	ACF_1使能 (Acceptance Filter 1 Enable) 0: 禁止 1: 使能	R/W

30.5.17 CAN filter group code and mask register (CAN_ACF)

CAN Acceptance Filter code and mask Register

偏移地址：0xB8

复位值：0xFFFF XXXX

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	AIDEE	AIDE													ACODE[28:16] or AMASK[28:16]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ACODE[15:0] or AMASK[15:0]															

位	标记	位名	功能	读写
b31	Reserved	-	读出值为不定。	R
b30	AIODEE	IDE位比较使能	IDE位比较使能 (Acceptance mask IDE bit check enable) 尽在SELMASK=1时有效 0: 筛选器接收标准格式和扩展格式帧 1: 筛选器接收AIDE位定义的标准格式或者扩展格式帧	R/W
b29	AIDE	IDE位MASK	IDE位MASK 0: 筛选器仅接收标准格式 1: 筛选器仅接收扩展格式	R/W
b28~b0	ACODE/ AMASK	筛选器CODE/ 筛选器MASK	筛选器CODE (acceptance filter code) 通过ACFADR指向具体的筛选器。 SELMASK=0时表示筛选器的CODE。 标准格式时使用位10~位0, 扩展格式时使用位29~位0。 筛选器CODE (acceptance filter mask) 通过ACFADR指向具体的筛选器。 SELMASK=1时表示筛选器的MASK。 标准格式时使用位10~位0, 扩展格式时使用位29~位0。	R/W

30.5.18 TTCAN TB slot pointer register (CAN_TBSLOT)

TTCAN TB Slot Pointer Register

偏移地址：0xBE

复位值：0x00

b7	b6	b5	b4	b3	b2	b1	b0
TBE	TBF	-	-	-	TBPTR[2:0]		

位	标记	位名	功能	读写
b7	TBE	设置TB为空	设置TB为空 (set TB slot to "empty") 0: 无操作 1: 被TB PTR选择的SLOT被标记为空 当SLOT被标记为空并且TSFF=0时, TBE自动复位为0。 如果设定此位为1时, 被选定的SLOT中存在数据正在发送状态则 TBE=1, 则等到发送完成、发送错误或者发送取消后TBE复位为0。 TBE优先级高于TBF。	R/W
b6	TBE	设置TB为已填充	设置TB已填充 (set TB slot to "Filled") 0: 无操作 1: 被TB PTR选择的SLOT被标记为已填充 当SLOT被标记为已填充并且TSFF=1时, TBE自动复位为0。	R/W
b5~b3	Reserved	-	必须保持复位值。	R
b2~b0	TB PTR	TB SLOT指针	TB SLOT指针 (Pointer to a TB message slot) 000: 指向PTB 001: 指向STB SLOT1 010: 指向STB SLOT2 011: 指向STB SLOT3 100: 指向STB SLOT4 其他: 设定禁止 被指向的TB SLOT可以通过TBUF进行读写访问, 并且可以通过TBE和 TBF来标记是否已经被填充。 TTCAN模式时, TBSEL和TSNEXT寄存器无效。 注意: 仅可以在TSFF=0时对该位进行写操作。	R/W

30.5.19 TTCAN Time Trigger Configuration Register (CAN_TTCFG)

TTCAN TB Slot Pointer Register

偏移地址：0xBF

复位值：0x90

b7	b6	b5	b4	b3	b2	b1	b0
WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC[1:0]		TTEN

位	标记	位名	功能	读写
b7	WTIE	触发看门中断使能	触发看门中断使能 (Watch Trigger Interrupt Enable) 0: 禁止 1: 使能	R/W
b6	WTIF	触发看门中断标志	触发看门中断标志 (Watch Trigger Interrupt Flag) 当CYCLE COUNT值=TT_WTRIG设定值时且WTIE=1时，WTIF置位。 通过应用程序写1清0。	R/W
b5	TEIF	触发错误中断标志	触发错误中断标志 (Trigger Error Interrupt Flag) TT_TTIG设定值小于实际的CYCLE_TIME时，TEIF置位。	R
b4	TTIE	时间触发中断使能	时间触发中断使能 (Time Trigger Interrupt Enable) 0: 禁止 1: 使能	R/W
b3	TTIF	时间触发中断标志	时间触发中断标志 (Time Trigger Interrupt Flag) 当CYCLE COUNT值=TT_TRIG设定值时且TTIE=1时，TTIF置位。 如果TT_TRIG没有更新，则TTIF只置位1次，下一个基本CYCLE不置位。 通过应用程序写1清0。	R/W
b2~b1	T_PRESC	TTCAN 计数器预分频	TTCAN计数器预分频 (TTCAN Timer PRESCaler) 00: BT寄存器设定的位时间的1分频 01: BT寄存器设定的位时间的2分频 10: BT寄存器设定的位时间的4分频 11: BT寄存器设定的位时间的8分频 注意：T_PRESC可在TTEN=0时进行写操作或者写TTEN=1时同时操作。	R/W
b0	TTEN	TTCAN使能	TTCAN使能 (Time Trigger Enable) 0: 禁止 1: 使能TTCAN，计数器开始计数。	R/W

30.5.20 TTCAN Reference Message Register (CAN_REF_MSG)

TTCAN Reference Message Register

偏移地址：0xC0

复位值：0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
REF_IDE	-	REF_ID[28:16]													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
REF_ID[15:0]															

位	标记	位名	功能	读写
b31	REF_IDE	参考消息的IDE位	参考消息的IDE位 (REFERENCE message IDE bit) 0: 标准格式 1: 扩展格式	R/W
b30~b29	Reserved	-	读出值为不定。	R
b28~b0	REF_ID	参考消息的ID位	参考消息的ID位 (REFERENCE message IDentifier) REF_ID=0: REF_ID[28:0]有效 REF_ID=1: REF_ID[28:0]有效 REF_ID用于检测参考消息，适用于发送和接收。 检测到参考消息后，当前帧的Sync_Mark则变成Ref_Mark。 REF_ID[2:0]固定为0，并不检查其值，这样最多可以支持8个潜在的time master。 当REF_MSG的最高字节写操作后，则需要等待6个CAN时钟周期以完成REF_MSG向CAN时钟域的传递。	R/W

30.5.21 TTCAN Trigger Configuration Register (CAN_TRG_CFG)

TTCAN Reference Message Register

偏移地址：0xC4

复位值：0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TEW[3:0]	-			TTTYPE[2:0]				-				TTPTR[2:0]			

位	标记	位名	功能	读写
b15~b12	TEW	发送使能窗口	发送使能窗口 (Transmit Enable Window) 用于TTCAN的单次发送触发模式 (Single Shot Transmit Trigger) ,可以设定TEW+1个cycle time的窗口，发送仅在此窗口内被允许。	R/W
b11	Reserved	-	必须保持复位值。	R
b10~b8	TTTYPE	触发类型	触发类型 (Trigger Type) 000: 立即触发 (Immediate Trigger for immediate transmission) 001: 时间触发 (Time Trigger for receive triggers) 010: 单次发送触发 (Single Shot Transmit Trigger for exclusive time windows) 011: 发送开始触发 (Transmit Start Trigger for merged arbitrating time windows) 100: 发送停止触发 (Transmit Stop Trigger for merged arbitrating time windows) 其他: 保留 触发时间通过TT_TRIGGER寄存器设定，TB Slot通过TTPTR选择。	R/W
b7~b3	Reserved	-	必须保持复位值。	R
b2~b0	TTPTR	发送触发器 TB slot指针	发送触发器TB slot指针 (Transmit Trigger TB slot Pointer) 000: 指向PTB 001: 指向STB SLOT1 010: 指向STB SLOT2 011: 指向STB SLOT3 100: 指向STB SLOT4 其他: 设定禁止 如果指向的TB SLOT被标记为空，当到达触发时间后，TEIF置位。	R/W

30.5.22 TTCAN Trigger Time Register (CAN_TRG_TRIG)

TTCAN Reference Message Register

偏移地址：0xC6

复位值：0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TT_TRIG[15:0]															

位	标记	位名	功能	读写
b15~b0	TT_TRIG	触发时间	触发时间 (Trigger Time) 用于指定触发器的cycle time, 对于发送触发器来说发送SOF时间大约是TT_TRIG设定值+1 当TT_TRIG的最高字节写操作后, TT_TRIG值开始向CAN时钟域的传递。 因此如果BYTE操作, 需先写低字节再写高字节。	R/W

30.5.23 TTCAN Trigger Watchdog Time Register (TRG_WTRIG)

TTCAN Watch Trigger Time Register

偏移地址：0xC8

复位值：0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TT_WTRIG[15:0]															

位	标记	位名	功能	读写
b15~b0	TT_TRIG	触发时间	触发时间 (Trigger Time) 用于指定看门触发器的cycle time。 当TT_WTRIG的最高字节写操作后, TT_WTRIG值开始向CAN时钟域的传递。因此如果BYTE操作, 需先写低字节再写高字节。	R/W

30.6 Precautions for use

30.6.1 CAN 总线抗干扰措施

CAN 总线广泛应用于汽车、工业控制等行业，如果 CAN 应用现场电磁环境比较恶劣，存在电路不平衡性、空间电磁场、电网进线等因素，会导致 CAN 总线因辐射、传导干扰而产生大量通信噪声，致使总线错误帧增加、重发频繁，正确数据不能及时到达等情况，严重影响数据通信质量。所以实际应用中应该致力于消除噪声干扰，保障 CAN 总线网络稳定工作。

以下是几类常用的 CAN 总线抗干扰措施（包括但不仅限于）

- 1) 增加 CAN 总线接口电气隔离
- 2) 共收发器的信号地
- 3) 使用屏蔽双绞线缆并正确接地
- 4) 提高 CAN 传输线双绞程度
- 5) 增加信号保护器
- 6) 改进网络拓扑
- 7) 应用层软件抗干扰机制

30.6.2 CAN Controller Noise Limitation

在 CAN 总线网络中应确保通信的位时间满足标准协议的要求，若引入不满足位时间宽度的噪声干扰，可能引起 CAN 控制器异常动作。

31 Analog-to-digital converter (ADC)

31.1 Module Introduction

外部的模拟信号需要转变成数字信号才能由 MCU 进一步处理。内部集成了一个 12 位高精度、高转换速率的逐次逼近型模数转换器(SAR ADC)模块。具有以下特性：

- 12 位转换精度；
- 1M SPS 转换速度；
- 40 个输入通道，包括 36 路外部管脚输入、1 路内部温度传感器电压、1 路 1/3 AVCC 电压、2 路 DAC 输出；
- 4 种参考源：AVCC 电压、ExRef 引脚、内置 1.5v 参考电压、内置 2.5v 参考电压；
- ADC 的电压输入范围：0~Vref；
- 4 种转换模式：单次转换、顺序扫描连续转换、插队扫描连续转换、连续转换累加；
- 输入通道电压阈值监测；
- 软件可配置 ADC 的转换速率；
- 内置信号跟随器，可转换高阻信号；
- 支持片内外设自动触发 ADC 转换，有效降低芯片功耗并提高转换的实时性。

31.2 ADC Block Diagram

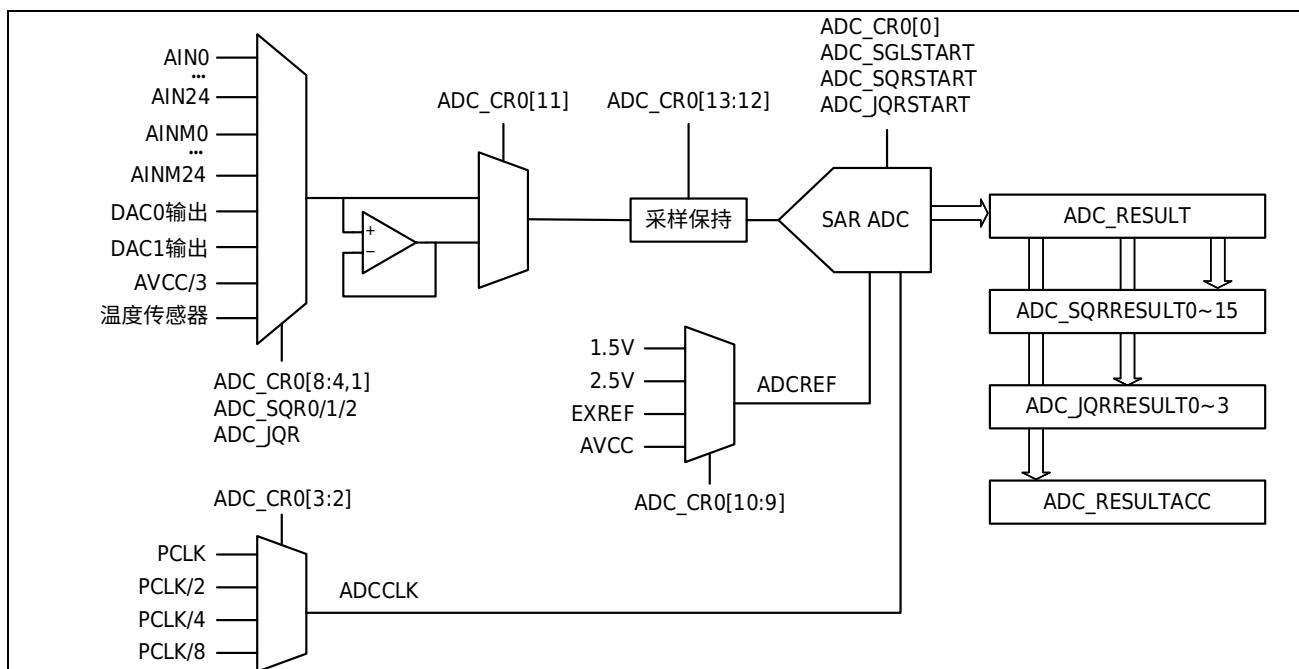


图 31-1 ADC 示意图

31.3 Conversion Timing and Conversion Speed

ADC 的转换时序如下图所示：一次完整的 ADC 转换由转换过程及逐次比较过程组成。其中转换过程需要 4~12 个 ADCCLK，由 ADC_CR0.SAM 配置；逐次比较过程需要 16 个 ADCCLK。所以，一次 ADC 转换共需要 20~28 个 ADCCLK。

ADC 转换速度的单位为 SPS，即每秒进行多少次 ADC 转换。ADC 转换速度的计算方法为：ADCCLK 的频率 / 一次 ADC 转换所需要的 ADCCLK 的个数。

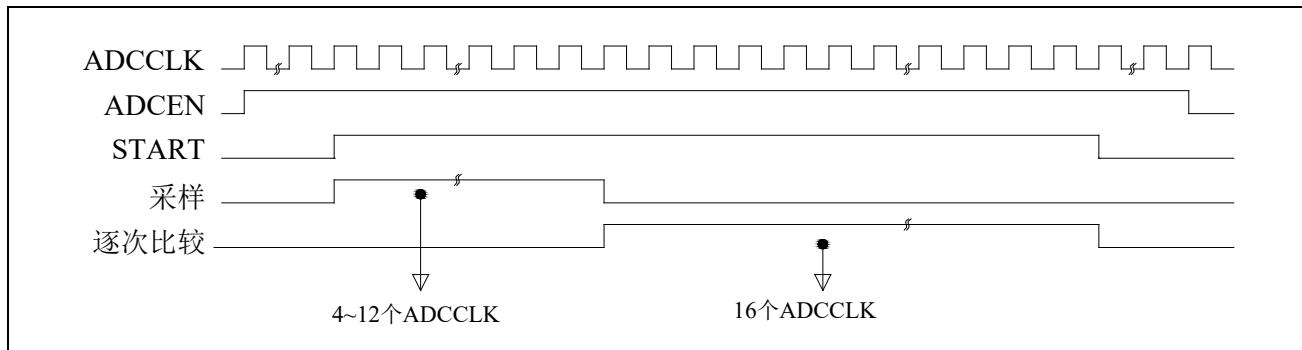


图 31-2 ADC 转换时序图

ADC 转换速度与 ADC 参考电压及 AVCC 电压相关，最高转换速度如下表所示：

ADC 参考电压	AVCC 电压	最高转换速度	最大 ADCCLK 频率
内部 1.5V	1.8V~5.5V	200K SPS	4MHz
内部 2.5V	2.8V~5.5V	200K SPS	4MHz
AVCC / ExRef	1.8V~2.4V	200K SPS	4MHz
AVCC / ExRef	2.4V~2.7V	500K SPS	16MHz
AVCC / ExRef	2.7V~5.5V	1M SPS	24MHz

31.4 Single Conversion Mode

在单次转换模式下，ADC 启动后只执行一次转换，可对所有的 30 路 ADC 通道进行转换。该模式既可通过设置 ADC_SglStart.Start 位启动也可通过设置 ADC_ExtTrigger0 的外部触发启动。一旦选定通道的 ADC 转换完成，ADC_IFR.SGLIF 位会自动置 1，转换结果保存在 ADC_Result 寄存器中。

通过 ADC_SglStart.Start 位启动 ADC 单次转换操作流程：

Step1：配置 PAADS~PEADS 相应的位，将待转换的 ADC 通道配置为模拟端口。

Step2：设置 PBADS.bit1 为 1，将 ADC 外部参考电压引脚配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 BGR_CR.BGR_EN 为 1，使能 BGR 模块。

Step4：设置 ADC_CR0.En 为 1，使能 ADC 模块。

Step5：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step6：设置 ADC_CR1.Mode 为 0，选择单次转换模式。

Step7：配置 ADC_CR0.Ref，选择 ADC 的参考电压。

Step8：设置 ADC_CR0.InRefEn 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step9：配置 ADC_CR0.SAM 及 ADC_CR0.CkDiv，设置 ADC 的转换速度。

Step10：配置 ADC_CR0.SGLMux，选择待转换的通道。

Step11：设置 ADC_ICR.SGLIC 为 0，清除 ADC_IFR.SGLIF 标志。

Step12：设置 ADC_SglStart.Start 为 1，启动 ADC 单次转换。

Step13：等待 ADC_IFR.SGLIF 变为 1，读取 ADC_Result 寄存器以获取 ADC 转换结果。

Step14：如需对其他通道进行转换，重复执行 Step10~Step13。

Step15：设置 ADC_CR0.En 及 BGR_CR.BGR_EN 为 0，关闭 ADC 模块、BGR 模块。

通过外部触发启动 ADC 单次转换操作流程：

Step1：配置 PAADS~PEADS 相应的位，将待转换的 ADC 通道配置为模拟端口。

Step2：设置 PBADS.bit1 为 1，将 ADC 外部参考电压引脚配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 BGR_CR.BGR_EN 为 1，使能 BGR 模块。

Step4：设置 ADC_CR0.En 为 1，使能 ADC 模块。

Step5：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step6：设置 ADC_CR1.Mode 为 0，选择单次转换模式。

Step7：设置 ADC_CR0.IE 为 1，使能 ADC 中断。

Step8：使能 NVIC 中断向量表中的 ADC 中断。

Step9：配置 ADC_CR0.Ref，选择 ADC 的参考电压。

Step10：设置 ADC_CR0.InRefEn 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step11：配置 ADC_CR0.SAM 及 ADC_CR0.CkDiv，设置 ADC 的转换速度。

Step12：配置 ADC_CR0.SGLMux，选择待转换的通道。

Step13：设置 ADC_IFR 为 0x0，清除 ADC 中断标志。

Step14：配置 ADC_ExtTrigger0，选择外部触发条件。

Step15：当外部触发条件触发 ADC 完成转换时，ADC 模块会产生中断。用户可在 ADC 中断服务程序中读取 ADC_Result 寄存器以获取 ADC 转换结果。

Step16：如需对其他通道进行转换，重复执行 Step12~Step15。

Step17：设置 ADC_CR0.En 及 BGR_CR.BGR_EN 为 0，关闭 ADC 模块、BGR 模块。

31.5 Scan Conversion Mode

扫描转换模式分为顺序扫描转换和插队扫描转换两种模式。两种模式各自单独工作时，均可连续对多个通道进行多次转换；当两种模式同时工作时，则优先对插队扫描配置的通道进行转换。

31.5.1 Sequential Scan Conversion Mode

顺序扫描转换模式最多可进行 16 次连续转换，转换的总次数由 ADC_SQR2.CNT 进行配置；可配置所有通道进行转换，待转换通道由 ADC_SQRx.CHxMux 进行配置。该模式既可通过设置 ADC_SqrStart.Start 位启动也可通过设置 ADC_ExtTrigger0 的外部触发启动。启动转换后，ADC 模块依次转换 CHxMux~CH0Mux 中配置的通道直到总转换次数完成。ADC 模块完成总转换次数后，ADC_IFR.SQRIF 位会自动置 1，转换结果保存在转换通道所对应的 ADC_SqrResultx~ADC_SqrResult0 寄存器中。下图演示了对 AIN0、AIN1、AIN5 进行 8 次转换的顺序扫描转换过程。其中顺序扫描转换通道 7, 4, 1 配置为 AIN1，转换通道 6, 3, 0 配置为 AIN0，转换通道 5, 2 配置为 AIN5。ADC_SqrStart.Start 置 1 后，ADC 模块会依次对顺序扫描转换通道 7~0 进行转换。

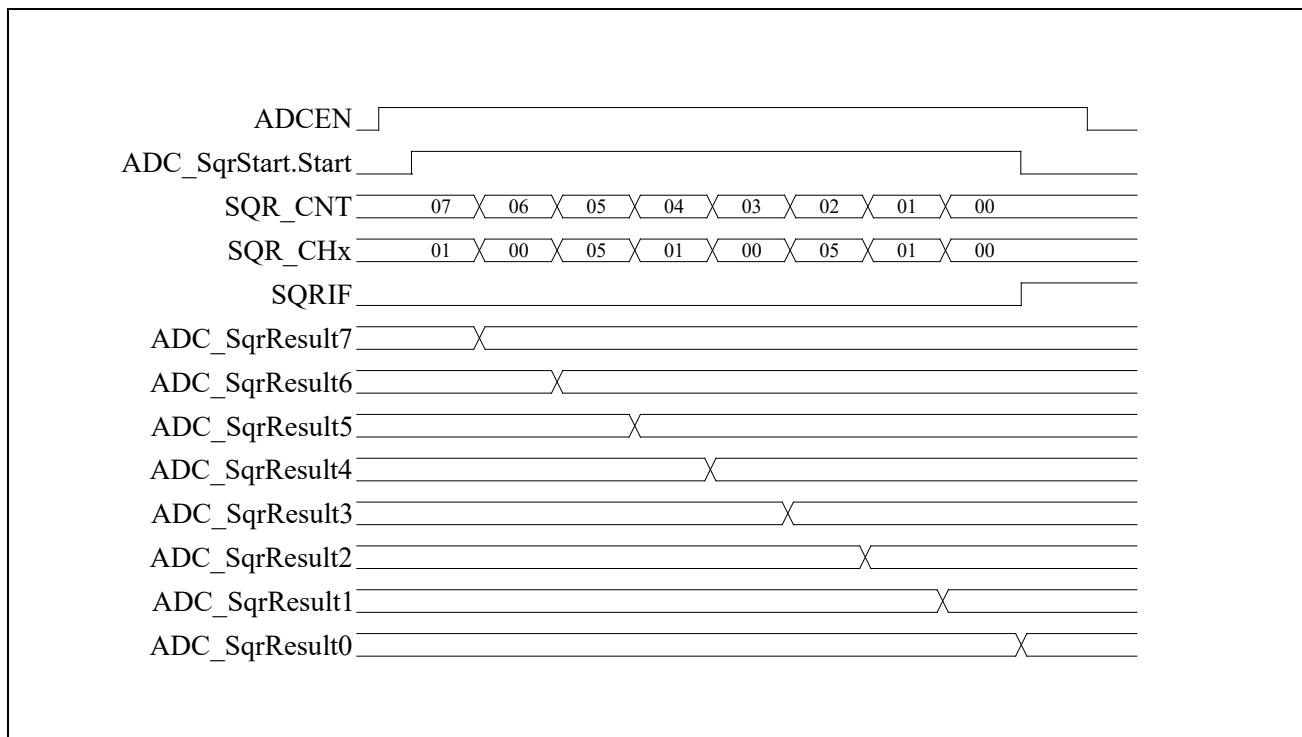


图 31-3 ADC 顺序扫描转换过程示例

通过 **ADC_SqrStart.Start** 位启动 ADC 顺序扫描转换操作流程：

Step1：配置 PAADS~PEADS 相应的位，将待转换的 ADC 通道配置为模拟端口。

Step2：设置 PBADS.bit1 为 1，将 ADC 外部参考电压引脚配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 BGR_CR.BGR_EN 为 1，使能 BGR 模块。

Step4：设置 ADC_CR0.En 为 1，使能 ADC 模块。

Step5：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step6：设置 ADC_CR1.Mode 为 1，选择扫描转换模式。

Step7：配置 ADC_CR0.Ref，选择 ADC 的参考电压。

Step8：设置 ADC_CR0.InRefEn 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step9：配置 ADC_CR0.SAM 及 ADC_CR0.CkDiv，设置 ADC 的转换速度。

Step10：配置 ADC_SQRx.CHxMux，选择顺序扫描转换通道。

Step11：配置 ADC_SQR2.CNT，选择顺序扫描转换的总转换次数。

注：总转换次数需与 Step9 中配置的转换通道数保持一致。

Step12：设置 ADC_ICR.SQRIC 为 0，清除 ADC_IFR.SQRIF 标志。

Step13：设置 ADC_SqrStart.Start 为 1，启动 ADC 顺序扫描转换。

Step14：等待 ADC_IFR.SQRIF 变为 1，读取 ADC_SqrResultx ~ ADC_SqrResult0 寄存器以获取相应通道的转换结果。

Step15：如需对其他通道进行转换，重复执行 Step10~Step14。

Step16：设置 ADC_CR0.En 及 BGR_CR.BGR_EN 为 0，关闭 ADC 模块、BGR 模块。

通过外部触发启动 ADC 顺序扫描转换操作流程：

Step1：配置 PAADS~PEADS 相应的位，将待转换的 ADC 通道配置为模拟端口。

Step2：设置 PBADS.bit1 为 1，将 ADC 外部参考电压引脚配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 BGR_CR.BGR_EN 为 1，使能 BGR 模块。

Step4：设置 ADC_CR0.En 为 1，使能 ADC 模块。

Step5：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step6：设置 ADC_CR1.Mode 为 1，选择扫描转换模式。

Step7：设置 ADC_CR0.IE 为 1，使能 ADC 中断。

Step8：使能 NVIC 中断向量表中的 ADC 中断。

Step9：配置 ADC_CR0.Ref，选择 ADC 的参考电压。

Step10：设置 ADC_CR0.InRefEn 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step11：配置 ADC_CR0.SAM 及 ADC_CR0.CkDiv，设置 ADC 的转换速度。

Step12：配置 ADC_SQRx.CHxMux，选择顺序扫描转换通道。

Step13：配置 ADC_SQR2.CNT，选择顺序扫描转换的总转换次数。

注：总转换次数需与 Step11 中配置的转换通道数保持一致。

Step14：设置 ADC_IFR 为 0x0，清除 ADC 中断标志。

Step15：配置 ADC_ExtTrigger0，选择外部触发条件。

Step16：当外部触发条件触发 ADC 完成转换时，ADC 模块会产生中断。用户可在 ADC 中断服务程序中读取 ADC_SqrResultx ~ ADC_SqrResult0 寄存器以获取相应通道的转换结果。

Step17：如需对其他通道进行转换，重复执行 Step12~Step16。

Step18：设置 ADC_CR0.En 及 BGR_CR.BGR_EN 为 0，关闭 ADC 模块、BGR 模块。

31.5.2 Queue-jumping scan conversion mode

插队扫描转换模式最多可进行 4 次连续转换，转换的总次数由 ADC_JQR.CNT 进行配置；可配置所有通道进行转换，待转换通道由 ADC_JQR.CHxMux 进行配置。该模式既可通过设置 ADC_JqrStart.Start 位启动也可通过设置 ADC_ExtTrigger1 的外部触发启动。启动转换后，ADC 模块依次转换 CHxMux~CH0Mux 中配置的通道直到总转换次数完成。ADC 模块完成总转换次数后，ADC_IFR.JQRIIF 位会自动置 1，转换结果保存在转换通道所对应的 ADC_JqrResultx~ ADC_JqrResult0 寄存器中。下图演示了对 AIN0、AIN1、AIN5 进行 4 次转换的插队扫描转换的过程。其中插队扫描转换通道 3，2，1，0 分别设置为 AIN5，AIN0，AIN1，AIN5。ADC_JqrStart.Start 置 1 后，ADC 模块会依次对插队扫描转换通道 3~0 进行转换。

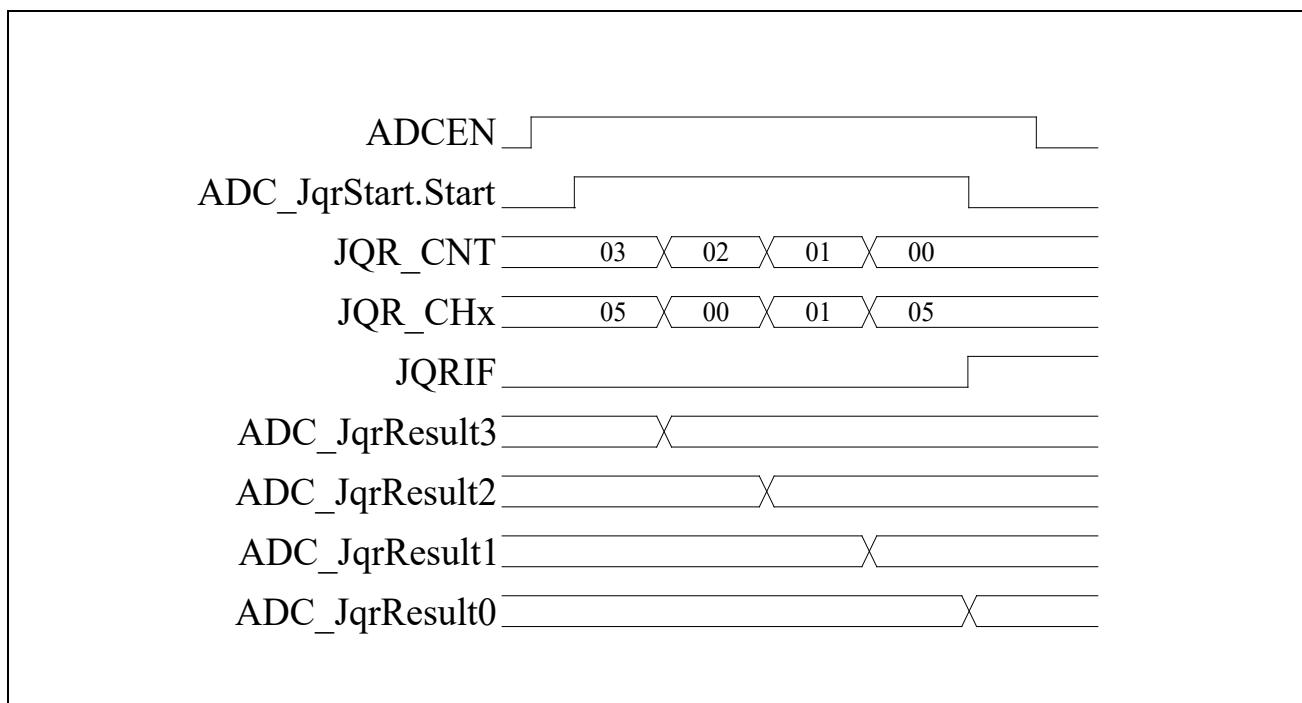


图 31-4 ADC 插队扫描转换过程示例

通过 ADC_JqrStart.Start 位启动 ADC 插队扫描转换操作流程：

Step1：配置 PAADS~PCADS 相应的位，将待转换的 ADC 通道配置为模拟端口。

Step2：设置 PBADS.bit1 为 1，将 ADC 外部参考电压引脚配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 BGR_CR.BGR_EN 为 1，使能 BGR 模块。

Step4：设置 ADC_CR0.En 为 1，使能 ADC 模块。

Step5：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step6：设置 ADC_CR1.Mode 为 1，选择扫描转换模式。

Step7：配置 ADC_CR0.Ref，选择 ADC 的参考电压。

Step8：设置 ADC_CR0.InRefEn 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step9：配置 ADC_CR0.SAM 及 ADC_CR0.CkDiv，设置 ADC 的转换速度。

Step10：配置 ADC_JQR.CHxMux，选择插队扫描转换通道。

Step11：配置 ADC_JQR.CNT，选择插队扫描转换的总转换次数。

注：总转换次数需与 Step9 中配置的转换通道数保持一致。

Step12：设置 ADC_ICR.JQRIC 为 0，清除 ADC_IFR.JQRIF 标志。

Step13：设置 ADC_JqrStart.Start 为 1，启动 ADC 插队扫描转换。

Step14：等待 ADC_IFR.JQRIF 变为 1，读取 ADC_JqrResultx~ ADC_JqrResult0 寄存器以获取相应通道的转换结果。

Step15：如需对其他通道进行转换，重复执行 Step10~Step14。

Step16：设置 ADC_CR0.En 及 BGR_CR.BGR_EN 为 0，关闭 ADC 模块、BGR 模块。

通过外部触发启动 ADC 插队扫描转换操作流程：

Step1：配置 PAADS~PCADS 相应的位，将待转换的 ADC 通道配置为模拟端口。

Step2：设置 PBADS.bit1 为 1，将 ADC 外部参考电压引脚配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 BGR_CR.BGR_EN 为 1，使能 BGR 模块。

Step4：设置 ADC_CR0.En 为 1，使能 ADC 模块。

Step5：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step6：设置 ADC_CR1.Mode 为 1，选择扫描转换模式。

Step7：设置 ADC_CR0.IE 为 1，使能 ADC 中断。

Step8：使能 NVIC 中断向量表中的 ADC 中断。

Step9：配置 ADC_CR0.Ref，选择 ADC 的参考电压。

Step10：设置 ADC_CR0.InRefEn 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step11：配置 ADC_CR0.SAM 及 ADC_CR0.CkDiv，设置 ADC 的转换速度。

Step12：配置 ADC_JQR.CHxMux，选择插队扫描转换通道。

Step13：配置 ADC_JQR.CNT，选择插队扫描转换的总转换次数。

注：总转换次数需与 Step11 中配置的转换通道数保持一致。

Step14：设置 ADC_IFR 为 0x0，清除 ADC 中断标志。

Step15：配置 ADC_ExtTrigger1，选择外部触发条件。

Step16：当外部触发条件触发 ADC 完成转换时，ADC 模块会产生中断。用户可在 ADC 中断服务程序中读取 ADC_JqrResultx~ADC_JqrResult0 寄存器以获取相应通道的转换结果。

Step17：如需对其他通道进行转换，重复执行 Step12~Step16。

Step18：设置 ADC_CR0.En 及 BGR_CR.BGR_EN 为 0，关闭 ADC 模块、BGR 模块。

插队扫描转换的执行优先级高于顺序扫描转换，当顺序扫描转换正在进行时，如果插队扫描转换启动，则顺序扫描在完成当前通道的转换后暂停，等到插队扫描转换完成后，再继续执行剩下的通道转换。下图演示了对 AIN0, AIN1, AIN5, AIN8 进行顺序扫描转换时，启动对 AIN2, AIN6 进行插队扫描转换的过程。其中顺序扫描转换通道 3, 2, 1, 0 分别设置为 AIN1, AIN0, AIN5, AIN8，插队扫描转换通道 1, 0 分别设置为 AIN6, AIN2。在顺序扫描进行 AIN0 转换的过程中，启动了插队扫描，顺序扫描会将当前 AIN0 的转换完成然后暂停，等到插队扫描完成 AIN6 和 AIN2 的转换后，顺序扫描再进行 AIN5 和 AIN8 的转换。

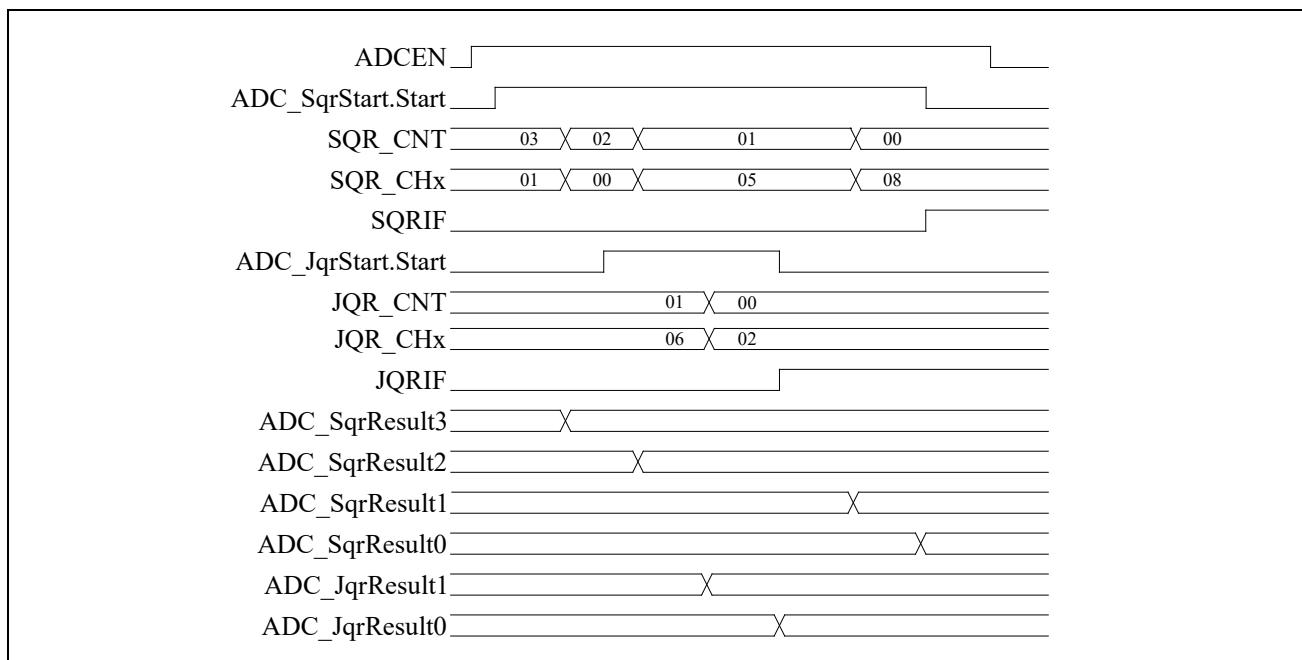


图 31-5 ADC 顺序扫描过程中进行插队扫描示例

31.5.3 Scan Conversion Trigger DMA Read

顺序扫描和插队扫描转换完成后，可自动触发 DMA 读取转换结果。顺序扫描模式下，通过配置 ADC_CR1.DmaSqr 为 1 使能该功能；插队扫描模式下，通过配置 ADC_CR1.DmaJqr 为 1 使能该功能。

31.6 Continuous Conversion Accumulation Mode

在连续转换累加模式下，启动一次 ADC 可对多个通道进行多次转换并对每次转换的结果进行累加；可配置所有通道进行转换。ADC 转换的总次数由 ADC_SQR2.CNT 进行配置；待转换通道由 ADC_SQRx.CHxMux 进行配置。该模式既可通过设置 ADC_SqrStart.Start 位启动也可通过设置 ADC_ExtTrigger0 的外部触发启动。启动连续转换后，ADC 模块依次转换 CHxMux~CH0Mux 中配置的通道直到总转换次数完成。ADC 模块完成总转换次数后，ADC_IFR.SQRIF 位会自动置 1，转换结果的累加值保存在 ADC_ResultAcc 寄存器中。

下图演示了对 AIN0、AIN1、AIN5 进行 10 次连续转换累加的过程。顺序扫描转换通道 9, 6, 3, 0 配置为 AIN0，转换通道 8, 5, 2 配置为 AIN1，转换通道 7, 4, 1 配置为 AIN5。ADC_SqrStart.Start 置 1 后，ADC 模块会按照顺序扫描转换通道配置依次进行转换，直到 SQR_CNT 的计数值变为 0。每次转换完成时，ADC_ResultAcc 寄存器都会自动进行累加。图中给定的 AIN0、AIN1、AIN5 的转换结果依次为 0x010、0x020、0x040。

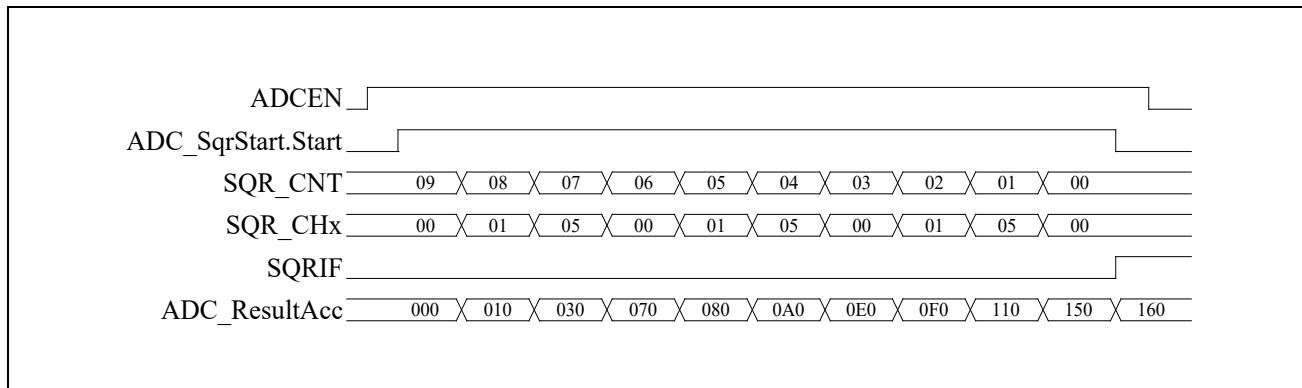


图 31-6 ADC 连续转换累加过程示例

通过 ADC_SqrStart.Start 位启动 ADC 连续转换累加操作流程：

Step1：配置 PAADS~PEADS 相应的位，将待转换的 ADC 通道配置为模拟端口。

Step2：设置 PBADS.bit1 为 1，将 ADC 外部参考电压引脚配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 BGR_CR.BGR_EN 为 1，使能 BGR 模块。

Step4：设置 ADC_CR0.En 为 1，使能 ADC 模块。

Step5：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step6：设置 ADC_CR1.Mode 为 1，选择扫描转换模式。

Step7：设置 ADC_CR1.RAccEn 为 1，使能 ADC 转换自动累加功能。

Step8：配置 ADC_CR0.Ref，选择 ADC 的参考电压。

Step9：设置 ADC_CR0.InRefEn 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step10：配置 ADC_CR0.SAM 及 ADC_CR0.CkDiv，设置 ADC 的转换速度。

Step11：配置 ADC_SQRx.CHxMux，选择顺序扫描转换通道。

Step12：配置 ADC_SQR2.CNT，选择顺序扫描转换的总转换次数。

注：总转换次数需与 Step10 中配置的转换通道数保持一致。

Step13：设置 ADC_ICR.SQRIC 为 0，清除 ADC_IFR.SQRIF 标志。

Step14：设置 ADC_CR1.RAccClr 为 0，清零 ADC_ResultAcc 寄存器。

Step15：设置 ADC_SqrStart.Start 为 1，启动 ADC 顺序扫描转换。

Step16：等待 ADC_IFR.SQRIF 变为 1，读取 ADC_ResultAcc 寄存器以获取转换结果累加值。

Step17：如需对其他通道进行转换，重复执行 Step11~Step16。

Step18：设置 ADC_CR0.En 及 BGR_CR.BGR_EN 为 0，关闭 ADC 模块、BGR 模块。

通过外部触发启动 ADC 连续转换累加操作流程：

Step1：配置 PAADS~PEADS 相应的位，将待转换的 ADC 通道配置为模拟端口。

Step2：设置 PBADS.bit1 为 1，将 ADC 外部参考电压引脚配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 BGR_CR.BGR_EN 为 1，使能 BGR 模块。

Step4：设置 ADC_CR0.En 为 1，使能 ADC 模块。

Step5：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step6：设置 ADC_CR1.Mode 为 1，选择扫描转换模式。

Step7：设置 ADC_CR0.IE 为 1，使能 ADC 中断。

Step8：使能 NVIC 中断向量表中的 ADC 中断。

Step9：设置 ADC_CR1.RAccEn 为 1，使能 ADC 转换自动累加功能。

Step10：配置 ADC_CR0.Ref，选择 ADC 的参考电压。

Step11：设置 ADC_CR0.InRefEn 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step12：配置 ADC_CR0.SAM 及 ADC_CR0.CkDiv，设置 ADC 的转换速度。

Step13：配置 ADC_SQRx.CHxMux，选择顺序扫描转换通道。

Step14：配置 ADC_SQR2.CNT，选择顺序扫描转换的总转换次数。

注：总转换次数需与 Step12 中配置的转换通道数保持一致。

Step15：设置 ADC_IFR 为 0x0，清除 ADC 中断标志。

Step16：设置 ADC_CR1.RAccClr 为 0，清零 ADC_ResultAcc 寄存器。

Step17：配置 ADC_ExtTrigger0，选择外部触发条件。

Step18：当外部触发条件触发 ADC 完成转换时，ADC 模块会产生中断。用户可在 ADC 中断服务程序中读取 ADC_ResultAcc 寄存器以获取转换结果累加值。

Step19：如需对其他通道进行转换，重复执行 Step13~Step18。

Step20：设置 ADC_CR0.En 及 BGR_CR.BGR_EN 为 0，关闭 ADC 模块、BGR 模块。

31.7 ADC Conversion External Trigger Source

ADC 转换既可通过软件配置启动，也可通过外部触发启动。

配置 ADC_ExtTrigger0 寄存器可设置 ADC 单次转换或顺序扫描转换的外部触发源。

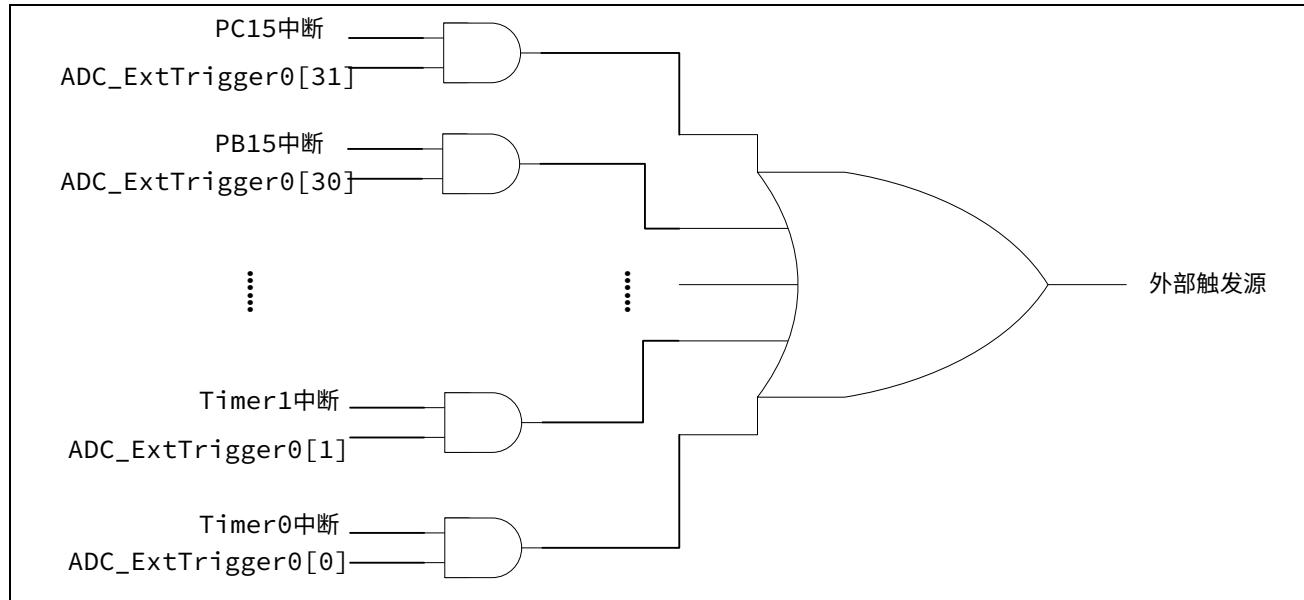


图 31-7 ADC 单次转换或顺序扫描转换外部触发源示意图

配置 ADC_ExtTrigger1 寄存器可设置 ADC 插队扫描转换的外部触发源。

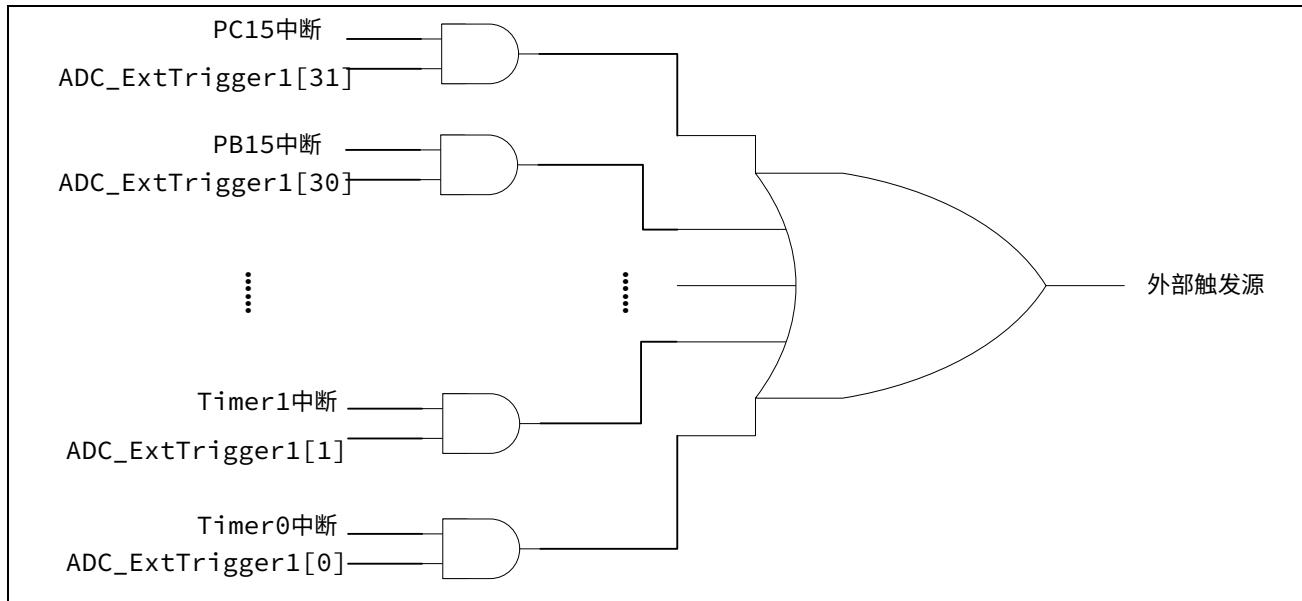


图 31-8 ADC 插队扫描转换外部触发源示意图

31.8 ADC Conversion Result Comparison

ADC 转换完成时，ADC 转换结果可以与用户设定的阈值进行比较，支持上阈值比较、下阈值比较、区间值比较。该功能需要将相应的控制位 HtCmp、LtCmp、RegCmp 置 1。该功能可实现对模拟量的自动监测，直到 ADC 转换结果符合用户预期时才产生中断申请用户程序介入。监测通道选择通过 ADC_CR1.ThCh 进行配置。

上阈值比较：当 ADC 转换结果位于[ADC_HT, 4095]区间内则 ADC_IFR.HTIF 置 1；向 ADC_ICR.HTIC 写入 0 则清零 ADC_IFR.HTIF。

下阈值比较：当 ADC 转换结果位于[0,ADC_LT)区间内则 ADC_IFR.LTIF 置 1；向 ADC_ICR.LTIC 写入 0 则清零 ADC_IFR.LTIF。

区间值比较：当 ADC 转换结果位于[ADC_LT,ADC_Ht)区间内则 ADC_IFR.REGIF 置 1；向 ADC_ICR.REGIC 写入 0 则清零 ADC_IFR.REGIF。

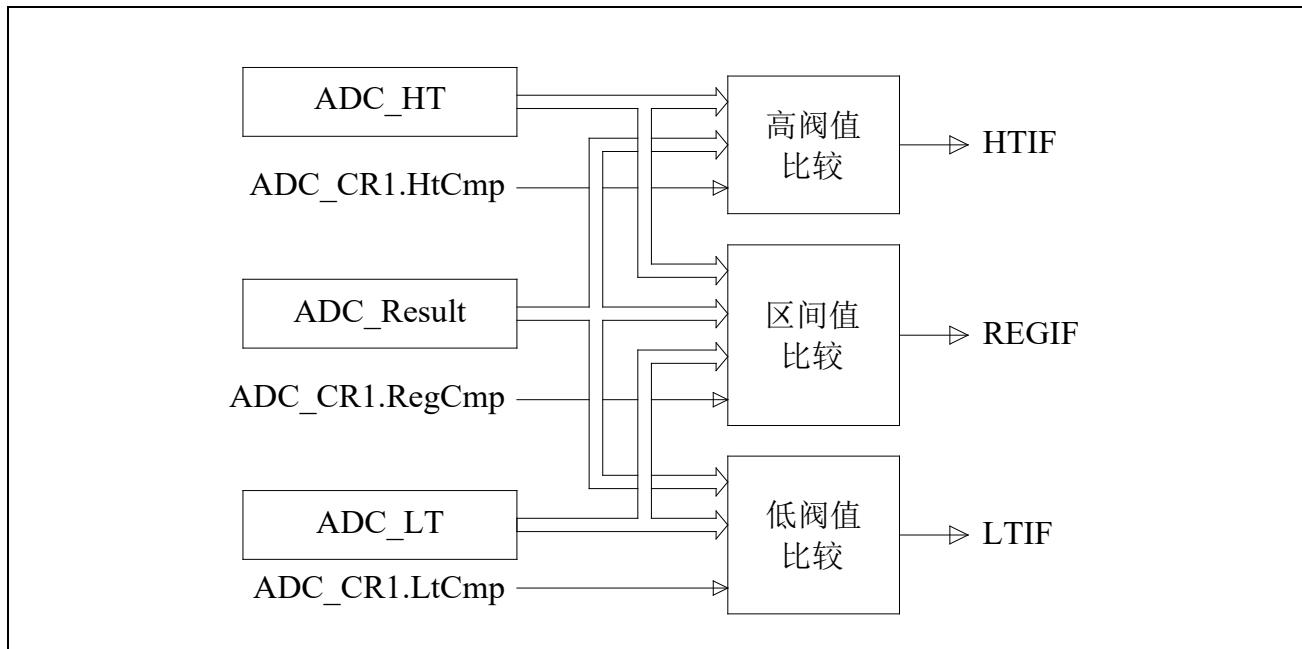


图 31-9 ADC 转换结果

31.9 ADC Interrupt

ADC 中断请求如下表所示：

中断源	中断标志	中断使能
ADC 插队扫描转换完成	ADC_IFR.JQRIF	ADC_CR0.IE
ADC 顺序扫描转换完成	ADC_IFR.SQRIF	
ADC 转换结果位于区间值区域 (大于等于下阈值小于上阈值)	ADC_IFR.REGIF	
ADC 转换结果位于上阈值区域 (大于等于上阈值)	ADC_IFR.HTIF	
ADC 转换结果比较下阈值区域 (小于下阈值)	ADC_IFR.LTIF	
ADC 单次转换完成	ADC_IFR.SGLIF	

31.10 Measuring the Ambient Temperature Using a Temperature Sensor

温度传感器的输出电压会随环境温度的改变而变化，故根据温度传感器的输出电压即可计算出相应的环境温度。当 ADC 模块的测量通道选择温度传感器的输出电压时，即可测量环境温度。

计算公式如下：

$$\text{环境温度} = 25 + 0.0854 \times V_{ref} \times (\text{AdcValue} - \text{Trim})$$

其中： V_{ref} 为当前 ADC 模块的参考电压，取值为 1.5 或 2.5。

AdcValue 为 ADC 模块测量温度传感器输出电压的结果，取值为 0~4095。

Trim 为 16Bits 的校准值，计算时需要从 Flash 存储器中读出，其存放地址详见下表。

ADC 参考电压	校准值存放地址	校准值精度
内部 1.5V	0x00100C34	$\pm 3^{\circ}\text{C}$
内部 2.5V	0x00100C36	$\pm 3^{\circ}\text{C}$

计算示例如下：

条件 1: $V_{ref}=2.5$ 、 $\text{AdcValue}=0x7E5$ 、 $\text{Trim}=0x76C$:

$$\text{温度 } 1: 25 + 0.0854 \times 2.5 \times (0x7E5 - 0x76C) = 50.8^{\circ}\text{C}.$$

条件 2: $V_{ref}=1.5$ 、 $\text{AdcValue}=0x72D$ 、 $\text{Trim}=0x76C$:

$$\text{温度 } 2: 25 + 0.0854 \times 1.5 \times (0x72D - 0x76C) = 16.9^{\circ}\text{C}.$$

通过 ADC 测量环境温度操作流程：

Step1: 设置 BGR_CR 为 3，使能 BGR 模块和温度传感器模块。

Step2: 设置 ADC_CR0.En 为 1，使能 ADC 模块。

Step3: 延时 20us，等待 ADC 及 BGR 模块启动完成。

Step4: 设置 ADC_CR1.Mode 为 0，选择单次转换模式。

Step5: 设置 ADC_CR0.InRefEn 为 1，使能 ADC 内部参考电压。

Step6: 配置 ADC_CR0.Ref，选择 ADC 的参考电压为内部 1.5V 或内部 2.5V。

Step7: 配置 ADC_CR0.SAM 及 ADC_CR0.CkDiv，设置 ADC 的转换速度。

Step8: 设置 ADC_CR0.SGLMux 为 0x1C，选择待转换的通道为温度传感器的输出。

Step9: 设置 ADC_CR0.Buf 为 1，使能输入信号放大器。

Step10: 设置 ADC_ICR.SGLIC 为 0，清除 ADC_IFR.SGLIF 标志。

Step11: 设置 ADC_SglStart.Start 为 1，启动 ADC 单次转换。

Step12: 等待 ADC_IFR.SGLIF 变为 1，读取 ADC_Result 寄存器以获取 ADC 转换结果。

Step13: 设置 ADC_CR0.En 及 BGR_CR 为 0，关闭 ADC 模块、BGR 模块、温度传感器模块。

Step14: 读取温度传感器校准值，根据公式计算当前的环境温度。

31.11 ADC Registers

基地址 0x40002400

表 31-1 ADC 寄存器

寄存器	偏移地址	描述
ADC_CR0	0x004	ADC 基本配置寄存器 0
ADC_CR1	0x008	ADC 基本配置寄存器 1
ADC_SQR0	0x040	ADC 顺序扫描转换通道配置寄存器 0
ADC_SQR1	0x044	ADC 顺序扫描转换通道配置寄存器 1
ADC_SQR2	0x048	ADC 顺序扫描转换通道配置寄存器 2
ADC_JQR	0x04C	ADC 插队扫描转换通道配置寄存器
ADC_SqrResult0	0x050	ADC 顺序扫描转换通道 0 转换结果
ADC_SqrResult1	0x054	ADC 顺序扫描转换通道 1 转换结果
ADC_SqrResult2	0x058	ADC 顺序扫描转换通道 2 转换结果
ADC_SqrResult3	0x05C	ADC 顺序扫描转换通道 3 转换结果
ADC_SqrResult4	0x060	ADC 顺序扫描转换通道 4 转换结果
ADC_SqrResult5	0x064	ADC 顺序扫描转换通道 5 转换结果
ADC_SqrResult6	0x068	ADC 顺序扫描转换通道 6 转换结果
ADC_SqrResult7	0x06C	ADC 顺序扫描转换通道 7 转换结果
ADC_SqrResult8	0x070	ADC 顺序扫描转换通道 8 转换结果
ADC_SqrResult9	0x074	ADC 顺序扫描转换通道 9 转换结果
ADC_SqrResult10	0x078	ADC 顺序扫描转换通道 10 转换结果
ADC_SqrResult11	0x07C	ADC 顺序扫描转换通道 11 转换结果
ADC_SqrResult12	0x080	ADC 顺序扫描转换通道 12 转换结果
ADC_SqrResult13	0x084	ADC 顺序扫描转换通道 13 转换结果
ADC_SqrResult14	0x088	ADC 顺序扫描转换通道 14 转换结果
ADC_SqrResult15	0x08C	ADC 顺序扫描转换通道 15 转换结果
ADC_JqrResult0	0x090	ADC 插队扫描转换通道 0 转换结果
ADC_JqrResult1	0x094	ADC 插队扫描转换通道 1 转换结果
ADC_JqrResult2	0x098	ADC 插队扫描转换通道 2 转换结果
ADC_JqrResult3	0x09C	ADC 插队扫描转换通道 3 转换结果
ADC_Result	0x0A0	ADC 转换结果
ADC_ResultAcc	0x0A4	ADC 转换结果累加值
ADC_LT	0x0A8	ADC 比较上阈值
ADC_LT	0x0AC	ADC 比较下阈值
ADC_IFR	0x0B0	ADC 中断标志寄存器
ADC_ICR	0x0B4	ADC 中断清除寄存器
ADC_ExtTrigger0	0x0B8	ADC 单次转换或顺序扫描转换外部中断触发源配置寄存器

寄存器	偏移地址	描述
ADC_ExtTrigger1	0x0BC	ADC 插队扫描转换外部中断触发源配置寄存器
ADC_SglStart	0x0C0	ADC 单次转换启动控制寄存器
ADC_SqrStart	0x0C4	ADC 顺序扫描转换启动控制寄存器
ADC_JqrStart	0x0C8	ADC 插队扫描转换启动控制寄存器
ADC_allstart	0x0CC	ADC 一直转换启动停止控制寄存器

31.11.1 ADC Basic Configuration Register 0 (ADC_CR0)

偏移地址 0x004

复位值 0x000067F0

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	InRefEn	SAM	Buf	Ref	SGLMux				CkDiv	CHMAP	En				
RW	RW	RW	RW	RW	RW				RW	RW	RW				

位	标记	功能描述														
31:16	Reserved	保留														
15	IE	ADC中断控制 1: 使能中断 0: 禁止中断														
14	InRefEn	ADC内部参考电压使能 1: 使能内部参考电压 0: 禁止内部参考电压														
13:12	SAM	ADC采样周期选择 00: 4个转换周期 01: 6个转换周期 10: 8个转换周期 11: 12个转换周期														
11	Buf	内置跟随器使能控制 0: 关闭跟随器, 外部输入信号与ADC直接相连。 1: 打开跟随器, 外部输入信号通过跟随器后与ADC相连, 用于测量高阻信号。 注1: 使能该功能时, 最大速率为200Ksps 注2: 以下条件必须使能该功能 待转换的信号输出阻抗较高 待转换的信号为: AVCC/3、内置温度传感器输出电压、DAC输出														
10:9	Ref	ADC参考电压选择 00: 内部1.5V 01: 内部2.5V 10: 外部参考电压ExRef (PB01) 11: AVCC电压														
8:4	SGLMux	单次转换模式转换通道选择 (ADC_CR0.CHMAP=0) 00000: AIN0 (PA00) 00001: AIN1 (PA01) 00010: AIN2 (PA02)				单次转换模式转换通道选择 (ADC_CR0.CHMAP=1) 00000: AIN0 (PD08) 00001: AIN1 (PD09) 00010: AIN2 (PD10)										

		00011: AIN3 (PA03) 00100: AIN4 (PA04) 00101: AIN5 (PA05) 00110: AIN6 (PA06) 00111: AIN7 (PA07) 01000: AIN8 (PB00) 01001: AIN9 (PB01) 01010: AIN10 (PC00) 01011: AIN11 (PC01) 01100: AIN12 (PC02) 01101: AIN13 (PC03) 01110: AIN14 (PC04) 01111: AIN15 (PC05) 10000: AIN16 (PB02) 10001: AIN17 (PB10) 10010: AIN18 (PB11) 10011: AIN19 (PB12) 10100: AIN20 (PB13) 10101: AIN21 (PB14) 10110: AIN22 (PB15) 10111: AIN23 (PE15) 11000: AIN24 (PC07)	00011: AIN3 (PD11) 00100: AIN4 (PA04) 00101: AIN5 (PA05) 00110: AIN6 (PE08) 00111: AIN7 (PE09) 01000: AIN8 (PE10) 01001: AIN9 (PB01) 01010: AIN10 (PE11) 01011: AIN11 (PE12) 01100: AIN12 (PE13) 01101: AIN13 (PE14) 01110: AIN14 (PC04) 01111: AIN15 (PC05) 10000: AIN16 (PB02) 10001: AIN17 (PB10) 10010: AIN18 (PB11) 10011: AIN19 (PB12) 10100: AIN20 (PB13) 10101: AIN21 (PB14) 10110: AIN22 (PB15) 10111: AIN23 (PE15) 11000: AIN24 (PC07)
		11001: DAC0输出 11010: DAC1输出 11011: AVCC/3 11100: 内置温度传感器输出电压 11101: Reserved	注: ADC_CR0.Buf必须为1 注: ADC_CR0.Buf必须为1 注: ADC_CR0.Buf必须为1 注: ADC_CR0.Buf必须为1
3:2	CkDiv	ADC时钟选择 00: PCLK时钟 01: PCLK时钟2分频 10: PCLK时钟4分频 11: PCLK时钟8分频	
1	CHMAP	OPA到DAC 输入映射 0: 使用SGLMUX选择的端口通道 1: 部分通道映射到OPA输出的相关端口	
0	En	ADC使能控制 1: 使能ADC 0: 禁止ADC	

CHMAP 为 1 时，以下 ADC 通道端口映射到其他端口。

AIN6M	PE08
AIN7M	PE09
AIN8M	PE10
AIN10M	PE11
AIN11M	PE12
AIN12M	PE13
AIN13M	PE14
AIN0M	PD08
AIN1M	PD09
AIN2M	PD10
AIN3M	PD11

31.11.2 ADC Basic Configuration Register 1 (ADC_CR1)

偏移地址 0x008

复位值 0x00008000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAcc Clr	Reg Cmp	HtC mp	LtC mp	RAcc En	Mod e	Dma Jqr	Dma Sqr			ThCh		Alig n		Reserved	

位	标记	功能描述
31:16	Reserved	保留
15	RAccClr	ADC 转换结果累加寄存器清零 1: 无作用; 0: ADC 转换结果累加寄存器 (ADC_ResultAcc) 清零。
14	RegCmp	ADC 区间比较控制 1: 使能区间比较 0: 禁止区间比较
13	HtCmp	ADC 高阈值比较控制 1: 使能高阈值比较 0: 禁止高阈值比较
12	LtCmp	ADC 低阈值比较控制 1: 使能低阈值比较 0: 禁止低阈值比较
11	RAccEn	ADC 转换结果自动累加控制 1: 使能 ADC 转换结果自动累加功能 0: 禁止 ADC 转换结果自动累加功能
10	Mode	ADC 转换模式选择 1: 扫描转换模式 0: 单次转换模式
9	Dmajqr	插队扫描触发 DMA 读取控制 1: 使能插队扫描转换触发 DMA 读取 0: 禁止插队扫描转换触发 DMA 读取
8	DmaSqr	顺序扫描触发 DMA 读取控制 1: 使能顺序扫描转换触发 DMA 读取 0: 禁止顺序扫描转换触发 DMA 读取
7:3	ThCh	阈值比较通道选择 00000: 选择通道 0 进行阈值比较 00001: 选择通道 1 进行阈值比较 00010: 选择通道 2 进行阈值比较

	 11101：选择通道 29 进行阈值比较
2	Align	转换结果对齐控制 1：转换结果 16Bits 左对齐存储 0：转换结果 16Bits 右对齐存储
1:0	Reserved	保留

31.11.3 ADC Sequential Scan Conversion Channel Configuration Register 0 (ADC_SQR0)

偏移地址 0x040

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		CH5Mux				CH4Mux				CH3Mux					
		RW				RW				RW					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		CH2MUX				CH1Mux				CH0Mux					
		RW				RW				RW					

位	标记	功能描述
31:30	Reserved	保留
29:25	CH5Mux	顺序扫描转换通道 5 选择, 设置参见 ADC_CR0.SGLMux
24:20	CH4Mux	顺序扫描转换通道 4 选择, 设置参见 ADC_CR0.SGLMux
19:15	CH3Mux	顺序扫描转换通道 3 选择, 设置参见 ADC_CR0.SGLMux
14:10	CH2Mux	顺序扫描转换通道 2 选择, 设置参见 ADC_CR0.SGLMux
9:5	CH1Mux	顺序扫描转换通道 1 选择, 设置参见 ADC_CR0.SGLMux
4:0	CH0Mux	顺序扫描转换通道 0 选择, 设置参见 ADC_CR0.SGLMux

31.11.4 ADC Sequential Scan Conversion Channel Configuration Register 1 (ADC_SQR1)

偏移地址 0x044

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		CH11Mux				CH10Mux				CH9Mux					
		RW				RW				RW					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		CH8Mux				CH7Mux				CH6Mux					
		RW				RW				RW					

位	标记	功能描述
31:30	Reserved	保留
29:25	CH11Mux	顺序扫描转换通道 11 选择, 设置参见 ADC_CR0.SGLMux
24:20	CH10Mux	顺序扫描转换通道 10 选择, 设置参见 ADC_CR0.SGLMux
19:15	CH9Mux	顺序扫描转换通道 9 选择, 设置参见 ADC_CR0.SGLMux
14:10	CH8Mux	顺序扫描转换通道 8 选择, 设置参见 ADC_CR0.SGLMux
9:5	CH7Mux	顺序扫描转换通道 7 选择, 设置参见 ADC_CR0.SGLMux
4:0	CH6Mux	顺序扫描转换通道 6 选择, 设置参见 ADC_CR0.SGLMux

31.11.5 ADC Sequential Scan Conversion Channel Configuration Register 2 (ADC_SQR2)

偏移地址 0x048

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CNT	CH15Mux						
								RW							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH14Mux				CH13Mux				CH12Mux							

位	标记	功能描述
31:24	Reserved	保留
23:20	CNT	顺序扫描转换次数
19:15	CH15Mux	顺序扫描转换通道 15 选择, 设置参见 ADC_CR0.SGLMux
14:10	CH14Mux	顺序扫描转换通道 14 选择, 设置参见 ADC_CR0.SGLMux
9:5	CH13Mux	顺序扫描转换通道 13 选择, 设置参见 ADC_CR0.SGLMux
4:0	CH12Mux	顺序扫描转换通道 12 选择, 设置参见 ADC_CR0.SGLMux

31.11.6 ADC Jump Queue Scan Conversion Channel Configuration Register (ADC_JQR)

偏移地址 0x04C

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										CNT	CH3Mux				
										RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2MUX					CH1Mux					CH0Mux					

位	标记	功能描述
31:24	Reserved	保留
21:20	CNT	插队扫描转换次数
19:15	CH3Mux	插队扫描转换通道 3 选择, 设置参见 ADC_CR0.SGLMux
14:10	CH2Mux	插队扫描转换通道 2 选择, 设置参见 ADC_CR0.SGLMux
9:5	CH1Mux	插队扫描转换通道 1 选择, 设置参见 ADC_CR0.SGLMux
4:0	CH0Mux	插队扫描转换通道 0 选择, 设置参见 ADC_CR0.SGLMux

31.11.7 ADC Sequential Scan Conversion Channel x Conversion Result (ADC_SqrResult0 - 15)

偏移地址 0x050 ~ 0x08C

复位值 0x00000000

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

位	标记	功能描述
31:12	Reserved	保留
11:0	Result	ADC 顺序扫描转换通道 x 转换结果

31.11.8 ADC Queue Scan Conversion Channel x Conversion Result (ADC_JqrResult0 - 3)

偏移地址 0x090 ~ 0x9C

复位值 0x00000000

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

位	标记	功能描述
31:12	Reserved	保留
11:0	Result	ADC 插队扫描转换通道 x 转换结果

31.11.9 ADC Conversion Result (ADC_Result)

偏移地址 0x0A0

复位值 0x00000000

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

位	标记	功能描述
31:12	Reserved	保留
11:0	Result	ADC 转换结果

31.11.10 ADC Conversion Result Accumulated Value (ADC_ResultAcc)

偏移地址 0x0A4

复位值 0x00000000

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

位	标记	功能描述
31:20	Reserved	保留
19:0	ResultAcc	ADC 转换累加值

31.11.11 ADC Compare Upper Threshold (ADC_HT)

偏移地址 0x0A8

复位值 0x00000FFF

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

位	标记	功能描述
31:12	Reserved	保留
11:0	HT	ADC 转换结果比较上阈值

31.11.12 ADC Compare Lower Threshold (ADC_LT)

偏移地址 0x0AC

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
Reserved																	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
Reserved				<table border="1" style="width: 100%;"><tr><td style="width: 50%;">LT</td><td style="width: 50%;">RW</td></tr></table>												LT	RW
LT	RW																

位	标记	功能描述
31:12	Reserved	保留
11:0	LT	ADC 转换结果比较下阈值

31.11.13 ADC Interrupt Flag Register (ADC_IFR)

偏移地址 0x0B0

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								JQRIF	SQRIF	REGIF	HHT ₋ INTF	LLT ₋ INTF	S ₋ INTF		
								RO	RO	RO	RO	RO	RO	RO	RO

位	标记	功能描述
31:6	Reserved	保留
5	JQRIF	ADC 插队扫描转换完成标志 1: ADC 插队扫描转换完成 0: ADC 插队扫描转换未完成
4	SQRIF	ADC 顺序扫描转换完成标志 1: ADC 顺序扫描转换完成 0: ADC 顺序扫描转换未完成
3	REGIF	ADC 转换结果比较区间标志 1: ADC 转换结果位于[ADC_LT , ADC_HT)区间内 0: ADC 转换结果位于[ADC_LT , ADC_HT)区间外
2	HTIF	ADC 转换结果比较上阈值标志 1: ADC 转换结果位于[ADC_HT, 4095]区间内 0: ADC 转换结果位于[ADC_HT, 4095]区间外
1	LTIF	ADC 转换结果比较下阈值标志 1: ADC 转换结果位于[0, ADC_LT)区间内 0: ADC 转换结果位于[0, ADC_LT)区间外
0	SGLIF	ADC 单次转换完成标志 1: ADC 单次转换完成 0: ADC 单次转换未完成

31.11.14 ADC Interrupt Clear Register (ADC_ICR)

偏移地址 0x0B4

复位值 0x0000003F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										JQRIC	SQRIC	REGIC	HHT-INTC	LLT-INTC	S-INTC
										R1W0	R1W0	R1W0	R1W0	R1W0	R1W0

位	标记	功能描述
31:6	Reserved	保留
5	JQRIC	写 0 清除 ADC 插队扫描转换完成标志 写 1 无作用
4	SQRIC	写 0 清除 ADC 顺序扫描转换完成标志 写 1 无作用
3	REGIC	写 0 清除 ADC 转换结果比较区间标志 写 1 无作用
2	HTIC	写 0 清除 ADC 转换结果比较上阈值 写 1 无作用
1	LTIC	写 0 清除 ADC 转换结果比较下阈值标志 写 1 无作用
0	SGLIC	写 0 清除 ADC 单次转换完成标志 写 1 无作用

31.11.15 ADC Single Conversion or Sequential Scan Conversion External Interrupt Trigger

Source Configuration Register (ADC_ExtTrigger0)

偏移地址 0x0B8

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC1 5	PB1 5	PA1 5	PC1 1	PB1 1	PA1 1	PD0 7	PC0 7	PB0 7	PA0 3	PD0 3	PC0 3	PB0 3	PA0 3	DMA	SPI1
RW	RW	RW													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI0	PCA	RTC	VC1	VC0	LPUART1	LPUART0	UAR T1	UAR T0	TIM6	TIM5	TIM4	TIM3	TIM2	TIM1	TIMO
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能描述
31	PC15	PC15 中断触发 ADC 转换
30	PB15	PB15 中断触发 ADC 转换
29	PA15	PA15 中断触发 ADC 转换
28	PC11	PC11 中断触发 ADC 转换
27	PB11	PB11 中断触发 ADC 转换
26	PA11	PA11 中断触发 ADC 转换
25	PD07	PD07 中断触发 ADC 转换
24	PC07	PC07 中断触发 ADC 转换
23	PB07	PB07 中断触发 ADC 转换
22	PA07	PA07 中断触发 ADC 转换
21	PD03	PD03 中断触发 ADC 转换
20	PC03	PC03 中断触发 ADC 转换
19	PB03	PB03 中断触发 ADC 转换
18	PA03	PA03 中断触发 ADC 转换
17	DMA	DMA 中断触发 ADC 转换
16	SPI1	SPI1 中断触发 ADC 转换
15	SPI0	SPI0 中断触发 ADC 转换
14	PCA	PCA 中断触发 ADC 转换
13	RTC	RTC 中断触发 ADC 转换
12	VC1	VC1 中断触发 ADC 转换
11	VC0	VC0 中断触发 ADC 转换
10	LPUART1	LPUART1 中断触发 ADC 转换
9	LPUART0	LPUART0 中断触发 ADC 转换
8	UART1	UART1 中断触发 ADC 转换
7	UART0	UART0 中断触发 ADC 转换
6	TIM6	Timer6 中断触发 ADC 转换

5	TIM5	Timer5 中断触发 ADC 转换
4	TIM4	Timer4 中断触发 ADC 转换
3	TIM3	Timer3 中断触发 ADC 转换
2	TIM2	Timer2 中断触发 ADC 转换
1	TIM1	Timer1 中断触发 ADC 转换
0	TIM0	Timer0 中断触发 ADC 转换

Notes:

- 1) TIM4/5/6 中断触发 ADC 自动转换，除了需要使能 TIM4/5/6 的相应中断外，还需要配置 Advanced Timer 的展频及中断触发选择寄存器 TIMX_CR 选择可以触发 ADC 的中断源。
- 2) 触发 ADC 使用的是各中断标志位的上升沿。如果需要重复触发，需要清除中断标志。如果不进入中断服务程序，请不要使能 NVIC 的中断使能。

31.11.16 ADC Queue Scan Conversion External Interrupt Trigger Source Configuration

Register (ADC_ExtTrigger1)

偏移地址 0x0BC

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC1 5	PB1 5	PA1 5	PC1 1	PB1 1	PA1 1	PD0 7	PC0 7	PB0 7	PA0 3	PD0 3	PC0 3	PB0 3	PA0 3	DMA	SPI1
RW	RW	RW													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI0	PCA	RTC	VC1	VC0	LPUART1	LPUART0	UAR T1	UAR T0	TIM6	TIM5	TIM4	TIM3	TIM2	TIM1	TIMO
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位	标记	功能描述
31	PC15	PC15 中断触发 ADC 转换
30	PB15	PB15 中断触发 ADC 转换
29	PA15	PA15 中断触发 ADC 转换
28	PC11	PC11 中断触发 ADC 转换
27	PB11	PB11 中断触发 ADC 转换
26	PA11	PA11 中断触发 ADC 转换
25	PD07	PD07 中断触发 ADC 转换
24	PC07	PC07 中断触发 ADC 转换
23	PB07	PB07 中断触发 ADC 转换
22	PA07	PA07 中断触发 ADC 转换
21	PD03	PD03 中断触发 ADC 转换
20	PC03	PC03 中断触发 ADC 转换
19	PB03	PB03 中断触发 ADC 转换
18	PA03	PA03 中断触发 ADC 转换
17	DMA	DMA 中断触发 ADC 转换
16	SPI1	SPI1 中断触发 ADC 转换
15	SPI0	SPI0 中断触发 ADC 转换
14	PCA	PCA 中断触发 ADC 转换
13	RTC	RTC 中断触发 ADC 转换
12	VC1	VC1 中断触发 ADC 转换
11	VC0	VC0 中断触发 ADC 转换
10	LPUART1	LPUART1 中断触发 ADC 转换
9	LPUART0	LPUART0 中断触发 ADC 转换
8	UART1	UART1 中断触发 ADC 转换
7	UART0	UART0 中断触发 ADC 转换
6	TIM6	Timer6 中断触发 ADC 转换

5	TIM5	Timer5 中断触发 ADC 转换
4	TIM4	Timer4 中断触发 ADC 转换
3	TIM3	Timer3 中断触发 ADC 转换
2	TIM2	Timer2 中断触发 ADC 转换
1	TIM1	Timer1 中断触发 ADC 转换
0	TIM0	Timer0 中断触发 ADC 转换

Notes:

- 1) TIM4/5/6 中断触发 ADC 自动转换，除了需要使能 TIM4/5/6 的相应中断外，还需要配置 Advanced Timer 的展频及中断触发选择寄存器 TIMX_CR 选择可以触发 ADC 的中断源。
- 2) 触发 ADC 使用的是各中断标志位的上升沿。如果需要重复触发，需要清除中断标志。如果不进入中断服务程序，请不要使能 NVIC 的中断使能。

31.11.17 ADC Single Conversion Start Control Register (ADC_SglStart)

偏移地址 0x0C0

复位值 0x00000000

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

位	标记	功能描述
31:1	Reserved	保留
0	Start	ADC 单次转换启动控制 1: 启动 ADC 单次转换 0: 停止 ADC 单次转换

31.11.18 ADC Sequential Scan Conversion Start Control Register (ADC_SqrStart)

偏移地址 0x0C4

复位值 0x00000000

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

位	标记	功能描述
31:1	Reserved	保留
0	Start	ADC 顺序扫描转换启动控制 1: 启动 ADC 顺序扫描转换 0: 停止 ADC 顺序扫描转换

31.11.19 ADC Jump Scan Conversion Start Control Register (ADC_JqrStart)

偏移地址 0x0C8

复位值 0x00000000

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

位	标记	功能描述
31:1	Reserved	保留
0	Start	ADC 插队扫描转换启动控制 1: 启动 ADC 插队扫描转换 0: 停止 ADC 插队扫描转换

31.11.20 ADC All-Time Conversion Start and Stop Control Register (ADC_allStart)

偏移地址 0x0CC

复位值 0x00000000

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

位	标记	功能描述
31:1	Reserved	保留
0	Start	ADC 插队扫描转换启动控制 1: 启动 ADC 一直转换 0: 停止 ADC 一直转换,需要等待到当次转换结束 读 0 无转换; 1 正在转换

32 Digital-to-Analog Converter (DAC)

DAC 模块是 12 位电压输出数模转换器。DAC 可以按 8 位或 12 位模式进行配置，并且可与 DMA 控制器配合使用。在 12 位模式下，数据可以采用左对齐或右对齐。可使用输入参考电压 VREF+（与 ADC 共享）。可以有选择地缓冲输出以实现更高的电流驱动。

32.1 DAC Characteristics

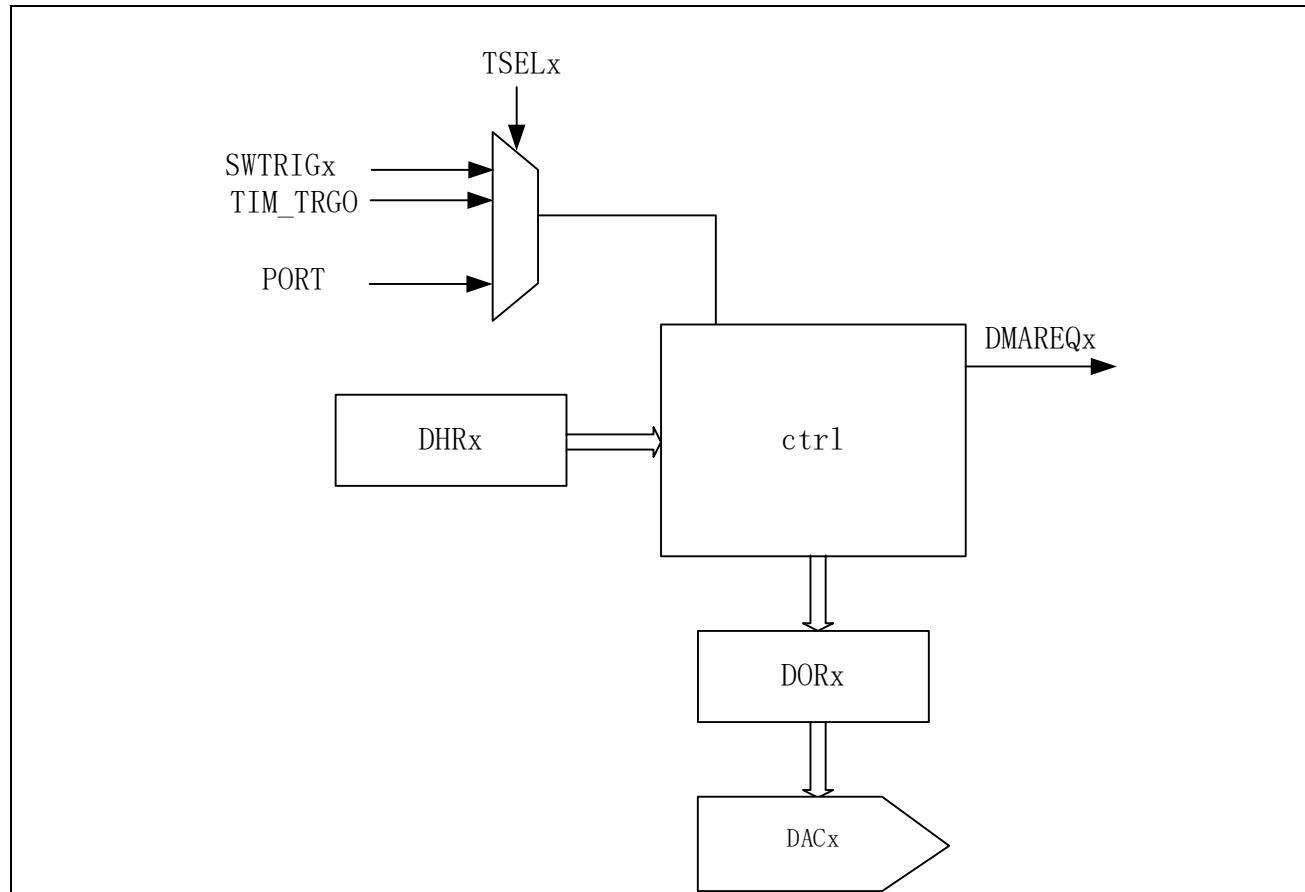
器件集成了两个 DAC 转换器，各具有一个输出通道：DAC0_OUT 和 DAC1_OUT。

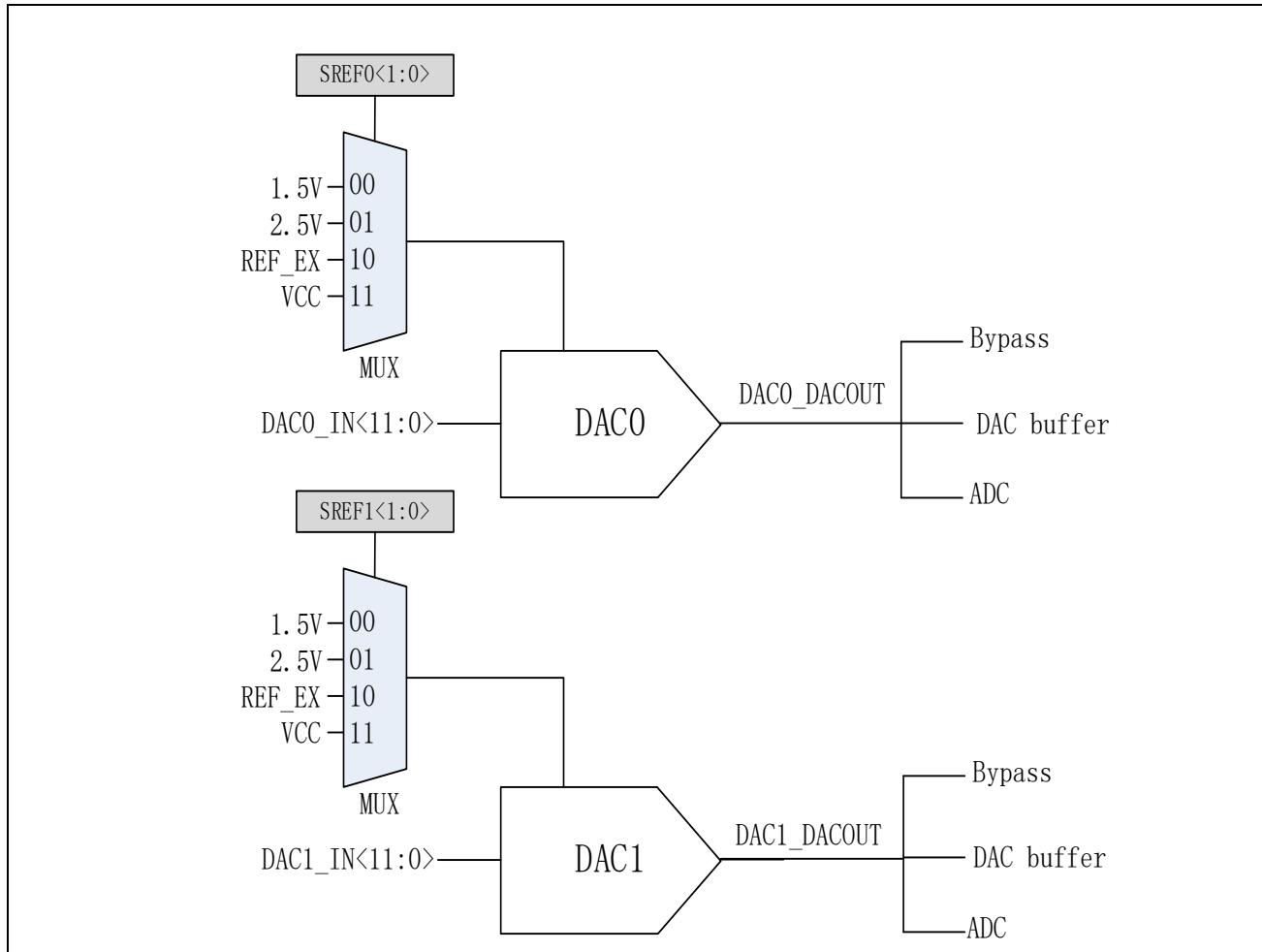
DAC 主要特性如下：

- 一个数据保持寄存器
- 12 位模式下数据采用左对齐或右对齐
- 同步更新功能
- 生成噪声波
- 生成三角波
- 双 DAC 通道，支持单独或同时转换
- DMA 功能（包括下溢检测）
- 通过外部触发信号进行转换
- 4 种参考源：AVCC 电压、ExRef 引脚、内置 1.5v 参考电压、内置 2.5v 参考电压；

32.2 Functional Description

32.2.1 Block Diagram.1





32.2.2 DAC Output Buffer Enable

DAC 可以使用输出缓冲器，可用来降低输出阻抗并在不增加外部运算放大器的情况下直接驱动外部负载。关于输出缓存器的设置参考<运算放大器 OPA>章节

通过 DAC_CR 寄存器中的 BOFF1 位，可使能或禁止 DAC 通道输出缓冲器。

32.2.3 DAC Channel Enable

将 DAC_CR 寄存器中的相应 ENx 位置 1，即可使能对应 DAC 通道。经过一段启动时间 tWAKEUP 后，各 DAC 通道被真正使能。

ENx 位只会使能模拟 DAC 通道 x 单元。即使 ENx 位复位，DAC 通道 x 数字接口仍处于使能状态。

32.2.4 DAC Output Voltage

经过线性转换后，数字输入会转换为 0 到 VREF+ 之间的输出电压。各 DAC 通道引脚的模拟输出电压通过以下公式确定：

$$DAC_{out} = V_{ref} * DOR / 4096$$

32.2.5 DAC Trigger Selection

如果 $TENx$ 控制位置 1，可通过外部事件（定时计数器、外部中断线）或软件触发转换。 $TSELx[2:0]$ 控制位将决定哪个可能事件将触发转换，如下表所示。

TSEL	000	001	010	011	100	101	110	111
触发源	TIM0 TRADC	TIM1 TRADC	TIM2 TRADC	TIM3 TRADC	TIM4 TRADC	TIM5 TRADC	SW TRIG	EXTI

每当 DAC 所选定时器 TRGO 输出或所选外部端口触发上检测到上升沿时，DAC_DHRx 寄存器中存储的最后一个数据即会转移到 DAC_DORx 寄存器中。发生触发后再经过三个 APB1 周期，DAC_DORx 寄存器将会得到更新。

如果选择软件触发，一旦 SWTRIG 位置 1，转换即会开始。DAC_DHRx 寄存器内容加载到 DAC_DORx 寄存器中后，SWTRIG 即由硬件复位。

注意：

- ENx 位置 1 时，无法更改 $TSELx[2:0]$ 位。如果选择软件触发，DAC_DHRx 寄存器的内容只需一个 APB1 时钟周期即可转移到 DAC_DORx 寄存器。

32.2.6 Single Mode Functional Description

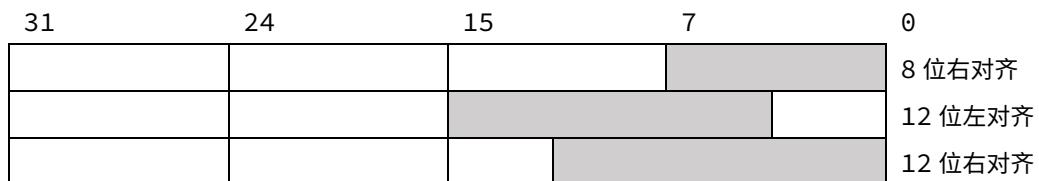
32.2.6.1 DAC Data Format

根据所选配置模式，数据必须按如下方式写入指定寄存器：

存在三种可能：

- 8 位右对齐：软件必须将数据加载到 DAC_DHR8Rx [7:0] 位（存储到 DHRx[11:4] 位）
- 12 位左对齐：软件必须将数据加载到 DAC_DHR12Lx [15:4] 位（存储到 DHRx[11:0] 位）
- 12 位右对齐：软件必须将数据加载到 DAC_DHR12Rx [11:0] 位（存储到 DHRx[11:0] 位）

根据加载的 DAC_DHRyyx 寄存器，用户写入的数据将移位并存储到相应的 DHRx（数据保持寄存器 x，即内部非存储器映射寄存器）。之后，DHRx 寄存器将被自动加载，或者通过软件或外部事件触发加载到 DORx 寄存器。



32.2.6.2 DAC Conversion

DAC_DORx 无法直接写入，任何数据都必须通过加载 DAC_DHRx 寄存器（写入 DAC_DHR8Rx、DAC_DHR12Lx、DAC_DHR12Rx）才能传输到 DAC 通道 x。

如果未选择触发（DAC_CR 寄存器中的 TENx 位复位），那么经过一个 APB1 时钟周期后，DAC_DHRx 寄存器中存储的数据将自动转移到 DAC_DORx 寄存器。但是，如果选择硬件触发（置位 DAC_CR 寄存器中的 TENx 位）且触发条件到来，将在三个 PCLK1 时钟周期后进行转移。

当 DAC_DORx 加载了 DAC_DHRx 内容时，模拟输出电压将在一段时间 tSETTLING 后可用，具体时间取决于电源电压和模拟输出负载。

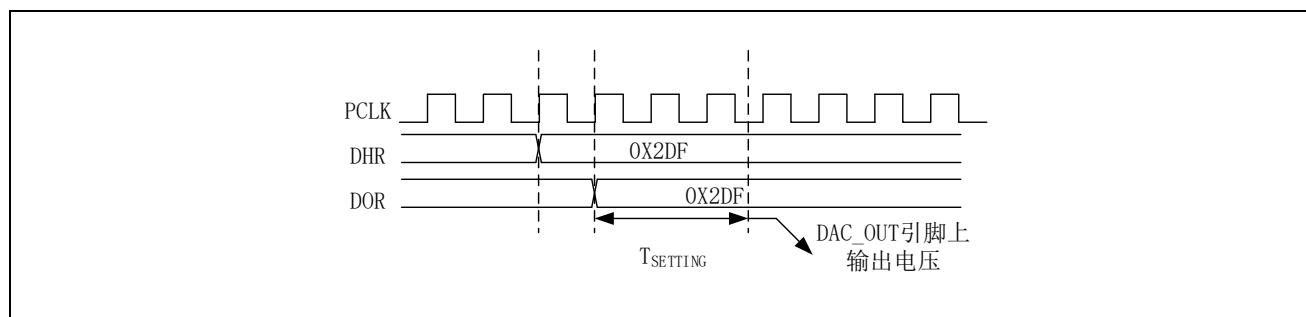


图 32-1 触发不使能时 DAC 转换时序图

生成单个 LFSR（线性反馈移位寄存器）的独立触发

要将 DAC 配置为此转换模式（请参见 32.2.8 噪音生成），需要遵循以下顺序：

1. 将 DAC 通道触发使能位 TENx 置 1。
 2. 通过设置 TSELx[2:0] 位配置触发源。
 3. 将 DAC 通道的 WAVEx[1:0] 位配置为“01”，并在 MAMPx[3:0] 位中配置相同的 LFSR 掩码值
 4. 将 DAC 通道数据加载到所需 DAC DHRx 寄存器 (DHR12RD、DHR12LD 或 DHR8RD)。

DAC 通道 x 触发信号到达时，LFSRx 计数器内容（使用相同的掩码）与 DHRx 寄存器内容相加，所得总和转移到 DAC DORx 中（三个 APB 时钟周期之后）。LFSRx 计数器随即更新。

生成单个三角波的独立触发

要将 DAC 配置为此转换模式（请参见 32.2.9 三角波生成），需要遵循以下顺序：

1. 将 DAC 通道 x 触发使能位 TENx 置 1。
 2. 通过设置 TSELx[2:0] 位配置触发源。
 3. 将 DAC 通道 x 的 WAVEx[1:0] 位配置为“1x”，并在 MAMPx[3:0] 位中配置相同的最大振幅值
 4. 将 DAC 通道 x 数据加载到所需 DAC DHRx 寄存器 (DHR12RD、DHR12LD 或 DHR8RD)。

DAC 通道 x 触发信号到达时, DAC 通道 x 三角波计数器内容(使用相同的三角波振幅)与 DHRx 寄存器内容相加, 所得总和转移到 DAC_DORx 中(三个 APB 时钟周期之后)。DAC 通道 x 三角波计数器随即更新。

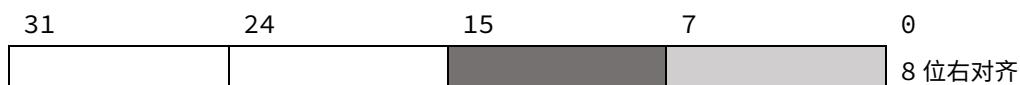
32.2.7 Dual Mode Function Description

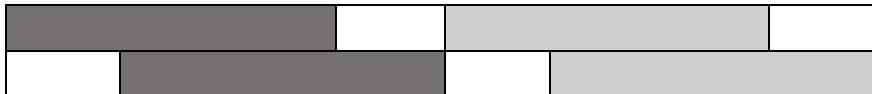
32.2.7.1 DAC Data Format

在 DAC 双通道模式下，有三种可能的方式：

- 8 位右对齐: 将 DAC0 通道的数据加载到 DAC_DHR8RD [7:0] 位 (存储到 DHR0[11:4]位), 将 DAC1 通道的数据加载到 DAC_DHR8RD [15:8] 位 (存储到 DHR1[11:4] 位)
 - 12 位左对齐: 将 DAC0 通道的数据加载到 DAC_DHR12RD [15:4] 位 (存储到 DHR0[11:0]位), 将 DAC1 通道的数据加载到 DAC_DHR12RD [31:20] 位 (存储到 DHR1[11:0] 位)
 - 12 位右对齐: 将 DAC0 通道的数据加载到 DAC_DHR12RD [11:0] 位 (存储到 DHR0[11:0]位), 将 DAC1 通道的数据加载到 DAC_DHR12RD [27:16] 位 (存储到 DHR1[11:0] 位)

根据加载的 DAC_DHRyyD 寄存器，用户写入的数据将移位并存储到 DHR0 和 DHR1（数据保持寄存器，即内部非存储器映射寄存器）。之后，DHR0 和 DHR1 寄存器将被自动加载，或者通过软件或外部事件触发分别被加载到 DOR0 和 DOR1 寄存器。





12 位左对齐

12 位右对齐

32.2.7.2 DAC Conversion

双模式下的 DAC 通道转换执行方式与单模式下基本相同，必须通过写入 DAC_DHR8Rx、DAC_DHR12Lx、DAC_DHR12Rx、DAC_DHR8RD、DAC_DHR12LD 或 DAC_DHR12RD 加载数据。

32.2.7.3 DAC Dual Conversion Mode Description

为了在同时需要两个 DAC 通道的应用中有效利用总线带宽，DAC 模块有三个双寄存器可供操作：DHR8RD、DHR12RD 和 DHR12LD。这样，只需一个寄存器访问即可同时驱动两个 DAC 通道。

通过两个 DAC 通道和这三个双寄存器可以实现 11 种转换模式。但如果需要，所有这些转换模式也都可以通过单独的 DHRx 寄存器来实现。

下面几段内容将介绍所有这些模式。

独立触发（不产生波形）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN0 和 TEN1 置 1
2. 将 TSEL1[2:0] 和 TSEL0[2:0] 设置为不同的值，以配置不同的触发源
3. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）

DAC0 通道触发信号到达时，DHR0 寄存器的内容转移到 DAC_DOR9（三个 APB 时钟周期之后）。

DAC 1 通道触发信号到达时，DHR1 寄存器的内容转移到 DAC_DOR1（三个 APB 时钟周期之后）。

独立触发（生成单个 LFSR）

要将 DAC 配置为此转换模式（请参见 32.2.8 噪音生成），需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN0 置 1
2. 将 TSEL1[2:0] 和 TSEL0[2:0] 设置为不同的值，以配置不同的触发源
3. 将两个 DAC 通道的 WAVE[1:0] 设置为“01”，并在 MAMPX[3:0] 位中配置相同的 LFSR 掩码值
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DHR12RD、DHR12LD 或 DHR8RD）

DAC 通道 0 触发信号到达时，LFSR0 计数器内容（使用相同的掩码）与 DHR0 寄存器内容相加，所得总和转移到 DAC_DOR0 中（三个 APB 时钟周期之后）。LFSR0 计数器随即更新。

DAC 通道 1 触发信号到达时，LFSR1 计数器内容（使用相同的掩码）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个 APB 时钟周期之后）。LFSR1 计数器随即更新。

独立触发（生成不同 LFSR）

要将 DAC 配置为此转换模式（请参见 32.2.8 噪音生成），需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN0 置 1
2. 将 TSEL1[2:0] 和 TSEL0[2:0] 设置为不同的值，以配置不同的触发源
3. 将两个 DAC 通道的 WAVE[1:0] 设置为“01”，并在 MAMP1[3:0] 和 MAMP0[3:0] 位中设置不同的 LFSR 掩码值
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）

DAC 通道 0 触发信号到达时，LFSR0 计数器内容（使用 MAMP0[3:0] 配置的掩码）与 DHR0 寄存器内容相加，所得总和转移到 DAC_DOR0 中（三个 APB 时钟周期之后）。LFSR0 计数器随即更新。

DAC 通道 1 触发信号到达时，LFSR1 计数器内容（使用 MAMP1[3:0] 配置的掩码）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个 APB 时钟周期之后）。LFSR1 计数器随即更新。

独立触发（生成单个三角波）

要将 DAC 配置为此转换模式（请参见 32.2.9 三角波生成），需要遵循以下顺序：

1. 将 DAC 通道 x 触发使能位 TENx 置 1。
2. 通过在 TSELx[2:0] 位中设置不同的值，以配置不同的触发源
3. 将 DAC 通道 x 的 WAVE[1:0] 位配置为“1x”，并在 MAMPx[3:0] 位中配置相同的最大振幅值
4. 将 DAC 通道 x 数据加载到所需 DAC_DHRx 寄存器

有关为 DAC 转换计时的 AHB 总线（APB 或 APB1）的详细信息，请参见 第 15.5.2 节：DAC 通道转换。

DAC 通道 x 触发信号到达时，DAC 通道 x 三角波计数器内容（使用相同的三角波振幅）与 DHRx 寄存器内容相加，所得总和转移到 DAC_DORx 中（三个 APB 时钟周期之后）。DAC 通道 x 三角波计数器随即更新。

独立触发（生成不同三角波）

要将 DAC 配置为此转换模式（请参见 32.2.9 三角波生成），需要遵循以下顺序：

1. 将 DAC 通道 x 触发使能位 TENx 置 1。
2. 通过在 TSELx[2:0] 位中设置不同的值，以配置不同的触发源
3. 将 DAC 通道 x 的 WAVE[1:0] 位设置为“1x”，并在 MAMPx[3:0] 位中设置不同的最大振幅值
4. 将 DAC 通道 x 数据加载到所需 DAC_DHRx 寄存器

DAC 通道 x 触发信号到达时，DAC 通道 x 三角波计数器内容（使用 MAMPx[3:0] 配置的三角波振幅）与 DHRx 寄存器内容相加，所得总和转移到 DAC_DORx 中（三个 APB 时钟周期之后）。DAC 通道 x 三角波计数器随即更新。

同步软件启动

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）

该配置用一个 APB 时钟周期。

同步触发（不产生波形）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN0 置 1
2. 将 TSEL1[2:0] 和 TSEL0[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源
3. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）

当触发信号到达时，DHR1 和 DHR0 寄存器内容将分别转移到 DAC_DOR1 和 DAC_DOR0 中（三个 APB 时钟周期之后）。

同步触发（生成单个 LFSR）

要将 DAC 配置为此转换模式（请参见 32.2.8 噪音生成），需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN0 置 1
2. 将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源
3. 将两个 DAC 通道的 WAVEx[1:0] 设置为“01”，并在 MAMPx[3:0] 位中配置相同的 LFSR 掩码值
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DHR12RD、DHR12LD 或 DHR8RD）触发信号到达时，LFSR1 计数器内容（使用相同的掩码）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个 APB 时钟周期之后）。LFSR1 计数器随即更新。同时，LFSR0 计数器内容（使用相同的掩码）与 DHR0 寄存器内容相加，所得总和转移到 DAC_DOR0 中（三个 APB 时钟周期之后）。LFSR0 计数器随即更新。

同步触发（生成不同 LFSR）

要将 DAC 配置为此转换模式（请参见 32.2.8 噪音生成），需要遵循以下顺序：

1. 将两个 DAC 通道触发使能位 TEN1 和 TEN0 置 1
2. 将 TSEL1[2:0] 和 TSEL0[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源

3. 将两个 DAC 通道的 WAVE_x[1:0] 设置为“01”，并在 MAMP1[3:0] 和 MAMP2[3:0] 位中设置不同的 LFSR 掩码值
4. 将 DAC 双通道数据加载到所需 DHR 寄存器（DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD）触发信号到达时，LFSR1 计数器内容（使用 MAMP1[3:0] 配置的掩码）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个 APB 时钟周期之后）。LFSR1 计数器随即更新。

同时，LFSR0 计数器内容（使用 MAMP0[3:0] 配置的掩码）与 DHR0 寄存器内容相加，所得总和转移到 DAC_DOR0 中（三个 APB 时钟周期之后）。LFSR0 计数器随即更新。

同步触发（生成单个三角波）

要将 DAC 配置为此转换模式（请参见 32.2.9 三角波生成），需要遵循以下顺序：

1. 将 DAC 通道 x 触发使能位 TEN1x 置 1。
2. 在 TSEL_x[2:0] 位中设置相同的值，以便为两个 DAC 通道配置相同的触发源。
3. 将 DAC 通道 x 的 WAVE_x[1:0] 位配置为“1x”，并使用 MAMP_x[3:0] 位配置相同最大振幅值
4. 将 DAC 通道 x 数据加载到所需 DAC_DHR_x 寄存器。

触发信号到达时，DAC 通道 x 三角波计数器内容（使用相同的三角波振幅）与 DHR_x 寄存器内容相加，所得总和转移到 DAC_DOR_x 中（三个 APB 时钟周期之后）。DAC 通道 x 三角波计数器随即更新。

同步触发（生成不同三角波）

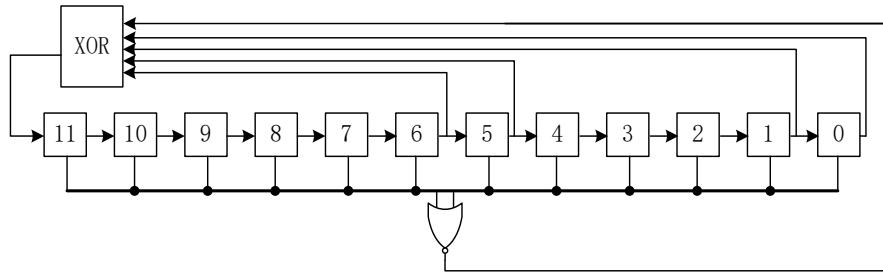
要将 DAC 配置为此转换模式（请参见 32.2.9 三角波生成），需要遵循以下顺序：

1. 将 DAC 通道 x 触发使能位 TEN_x 置 1。
2. 在 TSEL_x[2:0] 位中设置相同的值，以便为 DAC 通道 x 配置相同的触发源。
3. 将 DAC 通道 x 的 WAVE_x[1:0] 位配置为“1x”，并在 MAMP_x[3:0] 位中设置不同的最大振幅值。
4. 将 DAC 通道 x 数据加载到所需 DAC_DHR_x 寄存器。

触发信号到达时，DAC 通道 x 三角波计数器内容（使用 MAMP_x[3:0] 配置的三角波振幅）与 DHR_x 寄存器内容相加，所得总和转移到 DAC_DOR_x 中（三个 APB 时钟周期之后）。DAC 通道 x 三角波计数器随即更新。

32.2.8 Noise Generation

为了生成可变振幅的伪噪声，可使用 LFSR（线性反馈移位寄存器）。将 WAVE_x[1:0] 置为“01”即可选择生成噪声。LFSR 中的预加载值为 0xAAA。在每次发生触发事件后，经过三个 APB 时钟周期，该寄存器会依照特定的计算算法完成更新。



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

如果 LFSR 为 0x0000，将向其注入“1”（防锁定机制）。

可以通过复位 WAVEx[1:0] 位来将 LFSR 波形产生功能关闭。

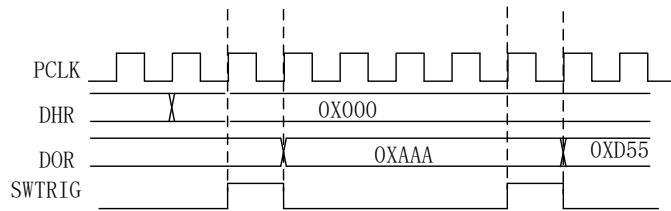


图 32-2 LFSR 产生波形的 DAC 转换（使用软件触发）

32.2.9 Triangle Wave Generation

可以在直流电流或慢变信号上叠加一个小幅三角波。将 WAVEx[1:0] 置为“10”即可选择 DAC 生成三角波。振幅通过 DAC_CR 寄存器中的 MAMPx[3:0] 位进行配置。每次发生触发事件后，经过三个 APB 时钟周期，内部三角波计数器将会递增。在不发生溢出的情况下，该计数器的值将与 DAC_DHRx 寄存器内容相加，所得总和将存储到 DAC_DORx 寄存器中。只要小于 MAMPx[3:0] 位定义的最大振幅，三角波计数器就会一直递增。一旦达到配置的振幅，计数器将递减至零，然后再递增，以此类推。

可以通过复位 WAVEx[1:0] 位来将三角波产生功能关闭。

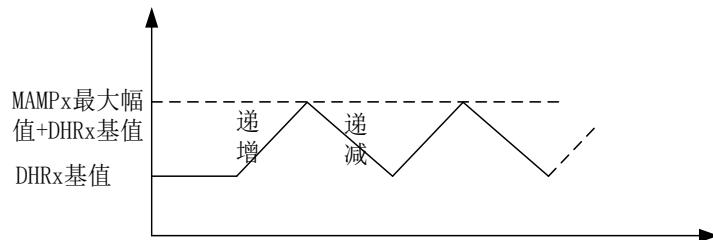


图 32-3 生成 DAC 三角波

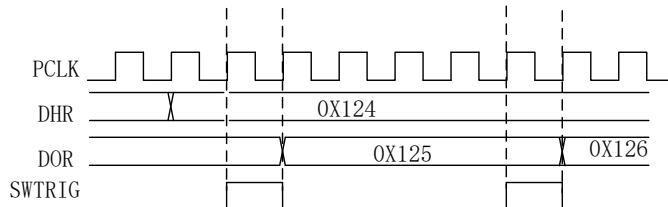


图 32-4 生成三角波波形的 DAC 转换 (使能软件触发)

32.2.10 DMA Request

每个 DAC 通道都具有 DMA 功能。两个 DMA 通道用于处理 DAC 通道的 DMA 请求。

当 DMAENx 位置 1 时，如果发生触发，则将产生 DAC DMA 请求。DAC_DHRx 寄存器的值随后转移到 DAC_DORx 寄存器。

在双通道模式下，如果两个 DMAENx 位均置 1，则将产生两个 DMA 请求。如果只需要一个 DMA 请求，用户应仅将相应 DMAENx 位置 1。这样，应用程序可以在双通道模式下通过一个 DMA 请求和一个特定 DMA 通道来管理两个 DAC 通道。

DMA 下溢

DAC DMA 请求没有缓冲队列。这样，如果第二个外部触发到达时尚未收到第一个外部触发的应答，将不会发出新的请求，并且 DAC_SR 寄存器中的 DAM 通道下溢标志 DMAUDRx 将置 1，以报告这一错误状况。DMA 数据传输随即禁止，并且不再处理其他 DMA 请求。DAC 通道仍将继续转换旧有数据。

软件应通过写入“0”来将 DMAUDRx 标志清零，将所用 DMA 数据流的 DMAEN 位清零，并重新初始化 DMA 和 DAC 通道，以便正确地重新开始 DMA 传输。软件应修改 DAC 触发转换频率或减轻 DMA 工作负载，以避免再次发生 DMA 下溢。最后，可通过使能 DMA 数据传输和转换触发来继续完成 DAC 转换。

对于各 DAC 通道，如果使能 DAC_CR 寄存器中相应的 DMAUDRIEx 位，还将产生中断。

32.3 DAC Registers

基地址 0x40002500

表 32-1 DAC 寄存器

寄存器	偏移地址	描述
DAC_CR0	0x000	控制寄存器
DAC_SWTRIGR	0x004	软件触发寄存器
DAC_DHR12R0	0x008	DAC0 12 位右对齐数据寄存器
DAC_DHR12L0	0x00C	DAC0 12 位左对齐数据寄存器
DAC_DHR8R0	0X010	DAC0 8 位右对齐数据寄存器
DAC_DHR12R1	0x014	DAC1 12 位右对齐数据寄存器
DAC_DHR12L1	0x018	DAC1 12 位左对齐数据寄存器
DAC_DHR8R1	0x01C	DAC1 8 位右对齐数据寄存器
DAC_DHR12RD	0X020	DAC0/1 12 位右对齐数据寄存器
DAC_DHR12LD	0x024	DAC0/1 12 位左对齐数据寄存器
DAC_DHR8RD	0x028	DAC0/1 8 位右对齐数据寄存器
DAC_DOR0	0x02C	DAC0 数据输出寄存器
DAC_DOR1	0X030	DAC1 数据输出寄存器
DAC_SR	0X034	状态寄存器
DAC_ETRS	0X038	DAC 测试寄存器/外部触发选择寄存器

32.3.1 DAC Control Register (DAC_CR0)

偏移地址 0x00

复位值 0xC000 C000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SREF1	DMA UDR1 E1	DMA EN1		MAMP1		WAVE1		TSEL1		TEN 1	BOF F1	EN1			
RW	RW	RW		RW		RW		RW		RW	RW	RW			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SREF0	DMA UDR IE0	DMA EN0		MAMP0		WAVE0		TSEL0		TEN 0	BOF F0	EN0			
RW	RW	RW		RW		RW		RW		RW	RW	RW			

位	标记	功能描述
31:30	SREF1	DAC1 参考电压选择 00: 内部 1.5V 01: 内部 2.5V 10: 外部参考电压 ExRef (PB01) 11: AVCC 电压
29	DMAUDRIE1	DAC1 DMA 下溢中断使能 0: 禁止 DAC1 通道 DMA 下溢中断； 1: 使能 DAC1 通道 DMA 下溢中断
28	DMAEN1	DAC1 通道 DMA 使能 0: 禁止 DAC1 通道 DMA 模式 1: 使能 DAC1 通道 DMA 模式
27:24	MAMP1	DAC1 通道掩码/振幅选择器 0000: 不屏蔽 LFSR 的位 0/三角波振幅等于 1 0001: 不屏蔽 LFSR 的位[1:0]/三角波振幅等于 3 0010: 不屏蔽 LFSR 的位[2:0]/三角波振幅等于 7 0011: 不屏蔽 LFSR 的位[3:0]/三角波振幅等于 15 0100: 不屏蔽 LFSR 的位[4:0]/三角波振幅等于 31 0101: 不屏蔽 LFSR 的位[5:0]/三角波振幅等于 63 0110: 不屏蔽 LFSR 的位[6:0]/三角波振幅等于 127 0111: 不屏蔽 LFSR 的位[7:0]/三角波振幅等于 255 1000: 不屏蔽 LFSR 的位[8:0]/三角波振幅等于 511 1001: 不屏蔽 LFSR 的位[9:0]/三角波振幅等于 1023 1010: 不屏蔽 LFSR 的位[10:0]/三角波振幅等于 2047 ≥1011: 不屏蔽 LFSR 的位[11:0]/三角波振幅等于 4095
23:22	WAVE1	DAC1 通道噪声/三角波生成使能 00: 禁止生成波

		01:使能生成噪声波 1x 使能生成三角波
21:19	TSEL1	DAC1 通道触发器选择 000 TIM0 TRADC 触发 001 TIM1 TRADC 触发 010 TIM2 TRADC 触发 011 TIM3 TRADC 触发 100 TIM4 TRADC 触发 101 TIM5 TRADC 触发 110 软件触发 111 外部端口触发
18	TEN1	DAC1 通道触发使能 0: 禁止 DAC1 通道触发, 写入 DAC_DHR1 寄存器的数据在一个 APB1 时钟周期之后转移到 DAC_DOR1 寄存器 1: 使能 DAC1 通道触发, DAC_DHR1 寄存器的数据在三个 APB1 时钟周期之后转移到 DAC_DOR1 寄存器
17	BOFF1	DAC1 通道输出缓冲器禁止控制, 使能缓冲器的 BUFFEN 在 OPA 控制寄存器中 0: 使能 DAC1 通道输出缓冲器, DAC1 通过缓存器到端口 1: 禁止 DAC1 通道输出缓冲器, DAC1 输出到端口 使用缓存作为输出到端口, 需要使能 OPA4 的 BUFEN 功能, 且不能使能 OPA4
16	EN1	DAC1 通道使能 0: 禁止 DAC1 通道 1: 使能 DAC1 通道
15:14	SREF0	DAC0 参考电压选择 00: 内部 1.5V 01: 内部 2.5V 10: 外部参考电压 ExRef (PB01) 11: AVCC 电压
13	DMAUDRIE0	DAC0 DMA 下溢中断使能 0: 禁止 DAC0 通道 DMA 下溢中断; 1: 使能 DAC0 通道 DMA 下溢中断
12	DMAENO	DAC0 通道 DMA 使能 0: 禁止 DAC0 通道 DMA 模式 1: 使能 DAC0 通道 DMA 模式
11:8	MAMPO	DAC0 通道掩码/振幅选择器 0000: 不屏蔽 LFSR 的位 0/三角波振幅等于 1 0001: 不屏蔽 LFSR 的位[1:0]/三角波振幅等于 3 0010: 不屏蔽 LFSR 的位[2:0]/三角波振幅等于 7 0011: 不屏蔽 LFSR 的位[3:0]/三角波振幅等于 15 0100: 不屏蔽 LFSR 的位[4:0]/三角波振幅等于 31

		0101: 不屏蔽 LFSR 的位[5:0]/三角波振幅等于 63 0110: 不屏蔽 LFSR 的位[6:0]/三角波振幅等于 127 0111: 不屏蔽 LFSR 的位[7:0]/三角波振幅等于 255 1000: 不屏蔽 LFSR 的位[8:0]/三角波振幅等于 511 1001: 不屏蔽 LFSR 的位[9:0]/三角波振幅等于 1023 1010: 不屏蔽 LFSR 的位[10:0]/三角波振幅等于 2047 ≥1011: 不屏蔽 LFSR 的位[11:0]/三角波振幅等于 4095
7:6	WAVE0	DAC0 通道噪声/三角波生成使能 00:禁止生成波 01:使能生成噪声波 1x 使能生成三角波
5:3	TSEL0	DAC0 通道触发器选择 000 TIM0 TRADC 触发 001 TIM1 TRADC 触发 010 TIM2 TRADC 触发 011 TIM3 TRADC 触发 100 TIM4 TRADC 触发 101 TIM5 TRADC 触发 110 软件触发 111 外部端口触发
2	TEN0	DAC0 通道触发使能 0: 禁止 DAC0 通道触发, 写入 DAC_DHR0 寄存器的数据在一个 APB1 时钟周期之后 转移到 DAC_DOR0 寄存器 1: 使能 DAC0 通道触发, DAC_DHR0 寄存器的数据在三个 APB1 时钟周期之后转移 到 DAC_DOR0 寄存器
1	BOFF0	DAC0 通道输出缓冲器禁止控制, 使能缓冲器的 BUFFEN 在 OPA 控制寄存器中 0: 使能 DAC0 通道输出缓冲器, DAC0 通过缓存器到端口 1: 禁止 DAC0 通道输出缓冲器, DAC0 输出到端口 使用缓存作为输出到端口, 需要使能 OPA3 的 BUFEN 功能, 且不能使能 OPA3
0	EN0	DAC0 通道使能 0: 禁止 DAC0 通道 1: 使能 DAC0 通道

32.3.2 DAC Software Trigger Register (DAC_SWTRIGR)

偏移地址 0x04

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														SWT RIG1	SWT RIG0
														W1	W1

位	标记	功能描述
31:2	Reserved	保留
1	SWTRIG1	DAC1 软件触发 写 1 软件触发，自动清零，写 0 无动作
0	SWTRIG0	DAC0 软件触发 写 1 软件触发，自动清零，写 0 无动作

32.3.3 DAC0 12-bit Right-aligned Data Holding Register (DAC_DHR12R0)

偏移地址 0x08

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DHR0 RW											

位	标记	功能描述
31:12	Reserved	保留
11:0	DHR0	DAC 通道 0 12 位右对齐数据 这些位由软件写入，用于为 DAC 通道 0 指定 12 位数据。

32.3.4 DAC0 12-bit Left-aligned Data Holding Register (DAC_DHR12L0)

偏移地址 0x0C

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DHR0 RW															

位	标记	功能描述
31:16	Reserved	保留
15:4	DHR0	DAC 通道 0 12 位左对齐数据 这些位由软件写入，用于为 DAC 通道 0 指定 12 位数据。
3:0	Reserved	保留

32.3.5 DAC0 8-bit Right-aligned Data Holding Register (DAC_DHR8R0)

偏移地址 0x10

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved DHR0 RW															

位	标记	功能描述
31:8	Reserved	保留
7:0	DHR0	DAC 通道 0 8 位右对齐数据 这些位由软件写入，用于为 DAC 通道 0 指定 8 位数据。写入的为 DHR 的高八位。

32.3.6 DAC1 12-bit Right-aligned Data Holding Register (DAC_DHR12R1)

偏移地址 0x14

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DHR1 RW							

位	标记	功能描述
31:12	Reserved	保留
11:0	DHR1	DAC 通道1 12 位右对齐数据 这些位由软件写入，用于为 DAC 通道1指定 12 位数据。

32.3.7 DAC1 12-bit Left-aligned Data Holding Register (DAC_DHR12L1)

偏移地址 0x18

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DHR1 RW															

位	标记	功能描述
31:16	Reserved	保留
15:4	DHR1	DAC 通道1 12 位左对齐数据 这些位由软件写入，用于为 DAC 通道1指定 12 位数据。
3:0	Reserved	保留

32.3.8 DAC1 8-bit Right-aligned Data Holding Register (DAC_DHR8R1)

偏移地址 0x1C

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DHR1 RW							

位	标记	功能描述
31:8	Reserved	保留
7:0	DHR1	DAC 通道 1 8 位右对齐数据 这些位由软件写入，用于为 DAC 通道 1 指定 8 位数据。写入的为 DHR 的高八位。

32.3.9 Dual DAC 12-bit Right Justified Data Holding Register (DAC_DHR12RD)

偏移地址 0x20

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								DHR1 RW							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DHR0 RW							

位	标记	功能描述
31:28	Reserved	保留
27:16	DHR1	DAC 通道 1 12 位左对齐数据 这些位由软件写入，用于为 DAC 通道 1 指定 12 位数据。
15:12	Reserved	保留
11:0	DHR0	DAC 通道 0 12 位右对齐数据 这些位由软件写入，用于为 DAC 通道 0 指定 12 位数据。

32.3.10 Dual DAC 12-bit Left-Justified Data Holding Register (DAC_DHR12LD)

偏移地址 0x24

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DHR1												Reserved			
RW												Reserved			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DHR0												Reserved			
RW												Reserved			

位	标记	功能描述
31:20	DHR1	DAC 通道1 12 位左对齐数据 这些位由软件写入，用于为 DAC 通道 1 指定 12 位数据。
19:16	Reserved	保留
15:4	DHR0	DAC 通道0 12 位左对齐数据 这些位由软件写入，用于为 DAC 通道 0 指定 12 位数据。
3:0	Reserved	保留

32.3.11 Dual DAC 8-bit Right Justified Data Holding Register (DAC_DHR8RD)

偏移地址 0x28

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DHR1								DHR0							
RW								RW							

位	标记	功能描述
31:16	Reserved	保留
15:8	DHR1	DAC 通道1 8 位右对齐数据 这些位由软件写入，用于为 DAC 通道 0 指定 8 位数据。写入的为 DHR 的高八位。
7:0	DHR0	DAC 通道0 8 位右对齐数据 这些位由软件写入，用于为 DAC 通道0指定 8 位数据。写入的为DHR的高八位。

32.3.12 DAC0 Data Output Register (DAC_DOR0)

偏移地址 0x2C

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DOR0 RO											

位	标记	功能描述
31:12	Reserved	保留
11:0	DOR0	DAC 0 通道数据输出 DAC 0 通道的数据输出

32.3.13 DAC1 Data Output Register (DAC_DOR1)

偏移地址 0x30

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DOR1 RO											

位	标记	功能描述
31:12	Reserved	保留
11:0	DOR1	DAC 1 通道数据输出 DAC 1 通道的数据输出

32.3.14 DAC Status Register (DAC_SR)

偏移地址 0x34

复位值 0x00004000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	DMAU	Reserved													
	DR1														
	RW														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	DMAU	Reserved													
	DR0														
	RW														

位	标记	功能描述
31:30	Reserved	保留
29	DMAUDR1	DAC 1 通道 DMA 下溢标志 0: DAC 1 通道未发生 DMA 下溢错误状况 1: DAC 1 通道发生 DMA 下溢错误状况 (当前所选触发源以高于 DMA 服务能力的频率驱动 DAC 1 通道转换)
28:14	Reserved	保留
13	DMAUDR0	DAC 0 通道 DMA 下溢标志 0: DAC0 通道未发生 DMA 下溢错误状况 1: DAC 0 通道发生 DMA 下溢错误状况 (当前所选触发源以高于 DMA 服务能力的频率驱动 DAC 0 通道转换)
12:0	Reserved	保留

32.3.15 DAC Trigger Selection Register (DAC_ETRS)

偏移地址 0x038

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
PTIRGSEL										RSV					
RW										RW					

位	标记	功能描述
31:7	Reserved	保留位
6:4	PTRIGSEL	端口触发选择 000 PA9 001 PB9 010 PC9 011 PD9 100 PE9 101 PF9 11X 保留
3:0	RSV	保留位

33 Analog comparator (VC)

33.1 Analog Voltage Comparator VC Introduction

模拟电压比较器 VC 用于比较两个输入模拟电压的大小，并根据比较结果输出高/低电平。当“+”输入端电压高于“-”输入端电压时，电压比较器输出为高电平；当“+”输入端电压低于“-”输入端电压时，电压比较器输出为低电平。本产品内部集成的模拟电压比较器 VC 具有以下特性：

- 支持电压比较功能；
- 支持内部 64 阶 VCC 分压（使用分压来源电压需要大于 1.8V）；
- 支持 16 个外部输入端口作为电压比较器的输入；
- 支持三种软件可配置的中断触发方式：高电平触发/上升沿触发/下降沿触发；
- 电压比较器的输出可以作为 通用定时器 和 低功耗定时器控制输入；
- 电压比较器的输出可以作为 高级定时器和通用定时器的刹车输入或者捕获输入；
- 支持在超低功耗模式下工作，电压比较器的中断输出可以将芯片从超低功耗模式下唤醒；
- 提供软件可配置的滤波时间以增强芯片的抗干扰能力。

注意：使用 VC 需要使能 BGR,参考 BGR 寄存器描述。

33.2 Voltage Comparator Block Diagram

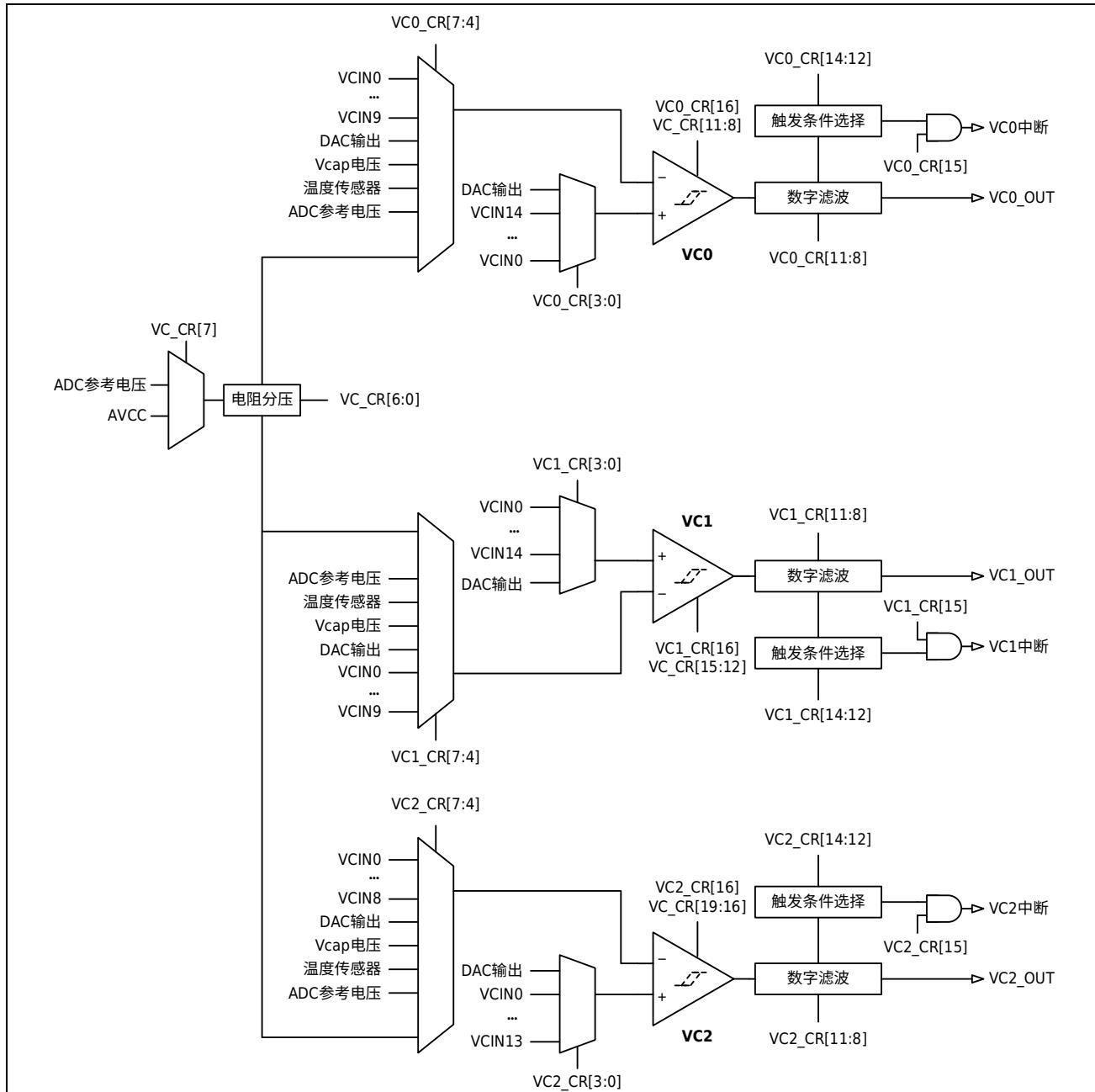


图 33-1 VC 框架图

33.3 Setup/Response Time

当使用电压比较器时，从 VC 使能或者 VC 的两端输入电压变化到输出正确的结果的时间由 VC 控制寄存器 (VC_CR) 中的 BIAS_SEL 控制位决定，电流越大 VC 响应越快，典型值从 200ns 到 20us 四档可调。

如果需要内部 BGR 提供的参考电压 (1.5V 或 2.5V) 作为比较基准电压。内部 BGR 的启动时间大约为 20us，电压比较器需要等待内部 BGR 稳定后才能正常输出。

33.4 Filter Time

在电压比较器固有的建立/响应时间之外，用户可以设置更长的滤波时间来过滤掉系统噪声，比如马达停止时的大电流噪声。

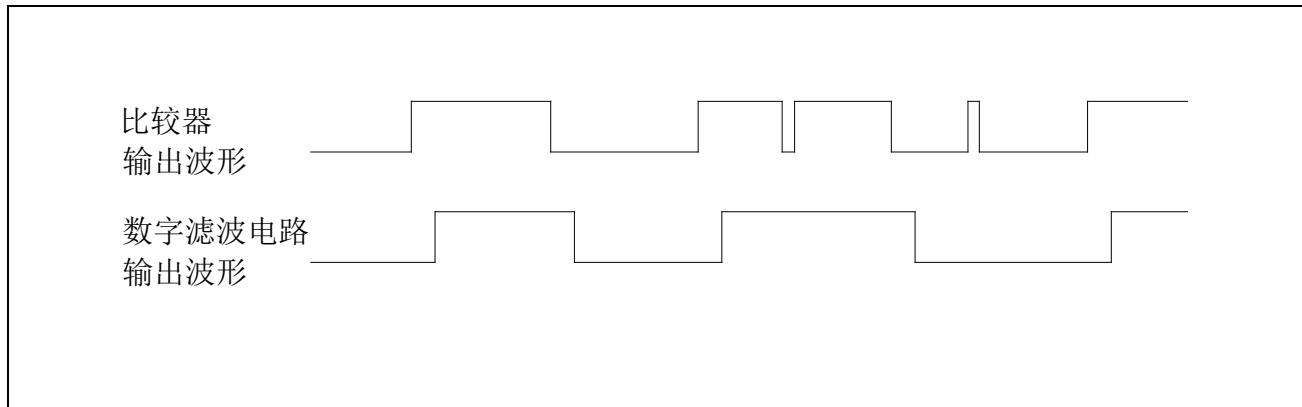


图 33-2 VC 滤波响应时间

33.5 Hysteresis Function

电压比较器可以选择迟滞功能，迟滞功能使能后的图示如下：

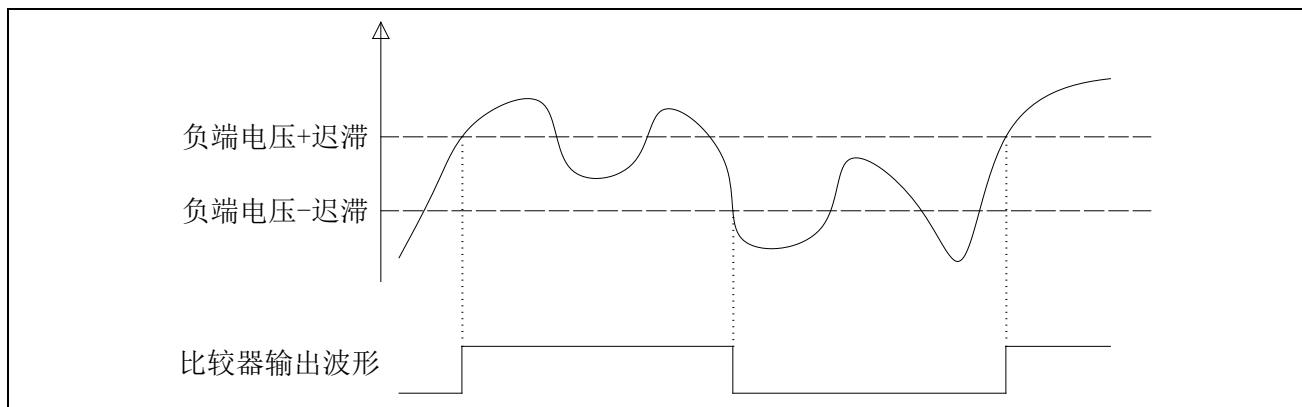


图 33-3 VC 迟滞功能

33.6 VC Registers

基地址 0x40002400

表 33-1 VC 寄存器

寄存器	偏移地址	描述
VC_CR	0x010	VC0/1 配置寄存器 0
VC0_CR	0x014	VC0 配置寄存器
VC1_CR	0x018	VC1 配置寄存器
VC0_OUT_CFG	0x01C	VC0 输出配置寄存器
VC1_OUT_CFG	0x020	VC1 输出配置寄存器
VC_IFR	0x024	VC 中断寄存器
VC2_CR	0x150	VC2 配置寄存器
VC2_OUT_CFG	0x154	VC2 输出配置寄存器

33.6.1 VC Configuration Register (VC_CR)

偏移地址 0x010

复位值 0x00000020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												VC2_HYS_SEL	VC2_BIAS_SEL		
R/W									R/W	R/W				R/W	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VC1_HYS_SEL	VC1_BIAS_SEL	VC0_HYS_SEL	VC0_BIAS_SEL	VC_REF2P5_SEL	VC_DIV_EN	VC_DIV									
R/W	R/W	R/W	R/W	R/W	R/W	R/W									

位	标记	功能描述
31:16	Reserved	保留
19:18	VC2_HYS_SEL	VC2 迟滞选择: 00:没有迟滞 01:迟滞电压大约 10mV 10:迟滞电压大约 20mV 11:迟滞电压大约 30mV
17:16	VC2_BIAS_SEL	VC2 功耗选择 (功耗越大, 响应速度越快) 00:300nA 01:1.2uA 10:10uA(需要开启 BGR, BGR 启动时间大约 20us) 11:20uA(需要开启 BGR, BGR 启动时间大约 20us)
15:14	VC1_HYS_SEL	VC1 迟滞选择: 00:没有迟滞 01:迟滞电压大约 10mV 10:迟滞电压大约 20mV 11:迟滞电压大约 30mV
13:12	VC1_BIAS_SEL	VC1 功耗选择 (功耗越大, 响应速度越快) 00:300nA 01:1.2uA 10:10uA(需要开启 BGR, BGR 启动时间大约 20us) 11:20uA(需要开启 BGR, BGR 启动时间大约 20us)
11:10	VC0_HYS_SEL	VC0 迟滞选择: 00:没有迟滞 01:迟滞电压大约 10mV 10:迟滞电压大约 20mV 11:迟滞电压大约 30mV
9:8	VC0_BIAS_SEL	VC0 功耗选择 (功耗越大, 响应速度越快) 00:300nA 01:1.2uA 10:10uA(需要开启 BGR, BGR 启动时间大约 20us)

		11:20uA(需要开启 BGR, BGR 启动时间大约 20us)
7	VC_REF2P5_SEL	VC_DIV 参考电压 Vref 选择 0:VCC 1:ADC_CR0.SREF 所选择的参考电压
6	VC_DIV_EN	6 位 DAC 使能 1: 使能
5:0	VC_DIV	6 位 DAC 配置 000000: 1/64 Vref 000001:2/64 Vref 000010:3/64 Vref 000011:4/64 Vref ... 111110:63/64 Vref 111111: Vref

33.6.2 VC0 Configuration Register (VC0_CR)

偏移地址 0x014

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															EN RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	level	rising	falling	debounce_time	FLTEN	n_sel	p_sel								
RW	RW	RW	RW	RW	RW	RW	RW								

位	标记	功能描述
31:17	Reserved	保留
16	EN	电压比较器使能 1: 使能电压比较器 0: 关闭电压比较器
15	IE	VC 中断使能 1: 使能; 0: 禁止
14	level	VC 输出信号触发中断选择 1: 使能高电平触发 INT flag 0: 禁止高电平触发 INT flag
13	rising	VC 输出信号触发中断选择 1: 使能上升沿触发 INT flag 0: 禁止上升沿触发 INT flag
12	falling	VC 输出信号触发中断选择 1: 使能下降沿触发 INT flag 0: 禁止下降沿触发 INT flag
11:9	debounce_time	VC 输出滤波时间配置 111: 滤波时间大约为 28.8ms 110: 滤波时间大约为 7.2ms 101: 滤波时间大约为 1.8ms 100: 滤波时间大约为 450us 011: 滤波时间大约为 112us 010: 滤波时间大约为 28us 001: 滤波时间大约为 14us 000: 滤波时间大约为 7us 注意: 滤波时间的配置只有在 FLTEN=1 时才有效。
8	FLTEN	1: 启动 VC 滤波 0: VC 无滤波
7:4	N_SEL	电压比较器“-”端输入选择

		0000: select channel 0 input PA0 0001: select channel 1 input PA1 0010: select channel 2 input PA2 0011: select channel 3 input PA3 0100: select channel 4 input PA4 0101: select channel 5 input PA5 0110: select channel 6 input PA6 0111: select channel 7 input PA7 1000: select channel 8 input PC4 1001: select channel 9 input PC5 1010:DAC0 1011:电阻分压输出电压 1100:内置温度传感器输出电压 1101:Reserved 1110:ADC 的 REF (使用需要使能 ADC) 1111:VCAP 的输出电压
3:0	P_SEL	电压比较器“+”端输入选择 0000: select channel 0 input PC0 0001: select channel 1 input PC1 0010: select channel 2 input PC2 0011: select channel 3 input PC3 0100: select channel 4 input PA0 0101: select channel 5 input PA1 0110: select channel 6 input PA2 0111: select channel 7 input PA3 1000: select channel 8 input PA4 1001: select channel 9 input PA5 1010: select channel 10 input PA6 1011: select channel 11 input PA7 1100: select channel 12 input PB4 1101: select channel 13 input PB5 1110: select channel 14 input PB6 1111: DAC0

33.6.3 VC1 Configuration Register (VC1_CR)

偏移地址 0x018

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															EN RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	level	rising	falling	debounce_time	FLTEN	n_sel	p_sel								
RW	RW	RW	RW	RW	RW	RW	RW								

位	标记	功能描述
31:17	Reserved	保留
16	EN	电压比较器使能 1: 使能电压比较器 0: 关闭电压比较器
15	IE	VC 中断使能 1: 使能; 0: 禁止
14	level	VC 输出信号触发中断选择 1: 使能高电平触发 INT flag 0: 禁止高电平触发 INT flag
13	rising	VC 输出信号触发中断选择 1: 使能上升沿触发 INT flag 0: 禁止上升沿触发 INT flag
12	falling	VC 输出信号触发中断选择 1: 使能下降沿触发 INT flag 0: 禁止下降沿触发 INT flag
11:9	debounce_time	VC 输出滤波时间配置 111: 滤波时间大约为 28.8ms 110: 滤波时间大约为 7.2ms 101: 滤波时间大约为 1.8ms 100: 滤波时间大约为 450us 011: 滤波时间大约为 112us 010: 滤波时间大约为 28us 001: 滤波时间大约为 14us 000: 滤波时间大约为 7us 注意: 滤波时间的配置只有在 FLTEN=1 时才有效。
8	FLTEN	1: 启动 VC 滤波 0: VC 无滤波
7:4	N_SEL	电压比较器“-”端输入选择

		0000: select channel 0 input PC0 0001: select channel 1 input PC1 0010: select channel 2 input PC2 0011: select channel 3 input PC3 0100: select channel 4 input PA0 0101: select channel 5 input PA1 0110: select channel 6 input PB0 0111: select channel 7 input PB1 1000: select channel 8 input PB2 1001: select channel 9 input PB3 1010: DAC1 1011: 电阻分压输出电压 1100: 内置温度传感器输出电压 1101: Reserved 1110: ADC 的 REF (使用需要使能 ADC) 1111: VCAP 引脚电压
3:0	P_SEL	电压比较器“+”端输入选择 0000: select channel 0 input PA0 0001: select channel 1 input PA1 0010: select channel 2 input PA2 0011: select channel 3 input PA3 0100: select channel 4 input PA4 0101: select channel 5 input PA5 0110: select channel 6 input PB1 0111: select channel 7 input PB2 1000: select channel 8 input PB10 1001: select channel 9 input PB12 1010: select channel 10 input PB13 1011: select channel 11 input PB14 1100: select channel 12 input PB4 1101: DAC1 1110: select channel 14 input PB6 1111: select channel 15 input PB7

33.6.4 VC0 Output Configuration Register (VC0_OUT_CFG)

偏移地址 0x01C

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
brake	TIM6	INV_TIM6	TIM5	INV_TIM5	TIM4	INV_TIM4	Reserved		TIM_BK	TIM3_RCLR	TIM2_RCLR	TIM1_RCLR	TIM0_RCLR	INV_Timer	
RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW	

位	标记	功能描述
31:16	Reserved	保留
15	brake	VC0 作为 Advanced Timer 刹车控制 1: 使能; 0: 禁止。
14	TIM6	VC0 filter 结果输出到 TIM6 捕获输入 CHA 使能 1: 使能; 0: 禁止。
13	INV_TIM6	VC0 filter 结果输出到 TIM6 反向使能 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
12	TIM5	VC0 filter 结果输出到 TIM5 捕获输入 CHA 使能 1: 使能; 0: 禁止。
11	INV_TIM5	VC0 filter 结果输出到 TIM5 反向使能 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
10	TIM4	VC0 filter 结果输出到 TIM4 捕获输入 CHA 使能 1: 使能; 0: 禁止。
9	INV_TIM4	VC0 filter 结果输出到 TIM4 反向使能 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
8:6	Res.	保留
5	TIMBK	VC0 filter 结果输出到 Timer0/1/2/3 刹车控制 1: 使能; 0: 禁止。
4	TIM3RCLR	VC0 filter 结果输出到 TIM3 REFCLR 使能控制 1: 使能; 0: 禁止。
3	TIM2RCLR	VC0 filter 结果输出到 TIM2 REFCLR 使能控制 1: 使能; 0: 禁止。
2	TIM1RCLR	VC0 filter 结果输出到 TIM1 REFCLR 使能控制 1: 使能; 0: 禁止。
1	TIM0RCLR	VC0 filter 结果输出到 TIM0 REFCLR 使能控制 1: 使能; 0: 禁止。
0	INV_Timer	VC0 filter 结果输出反向到各 TIM0/1/2/3/REFCLR 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。

33.6.5 VC1 Output Configuration Register (VC1_OUT_CFG)

偏移地址 0x020

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
brake	TIM6	INV_TIM6	TIM5	INV_TIM5	TIM4	INV_TIM4	Reserved		TIMBK	TIM3RCLR	TIM2RCLR	TIM1RCLR	TIMO_RCLR	INV_Timer	
RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW	

位	标记	功能描述
31:16	Reserved	保留
15	brake	VC1 作为 Advanced Timer 刹车控制 1: 使能; 0: 禁止。
14	TIM6	VC1 filter 结果输出到 TIM6 捕获输入 CHB 使能 1: 使能; 0: 禁止。
13	INV_TIM6	VC1 filter 结果输出到 TIM6 反向使能 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
12	TIM5	VC1 filter 结果输出到 TIM5 捕获输入 CHB 使能 1: 使能; 0: 禁止。
11	INV_TIM5	VC1 filter 结果输出到 TIM5 反向使能 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
10	TIM4	VC1 filter 结果输出到 TIM4 捕获输入 CHB 使能 1: 使能; 0: 禁止。
9	INV_TIM4	VC1 filter 结果输出到 TIM4 反向使能 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
8:6	Res.	保留
5	TIMBK	VC1 filter 结果输出到 Timer0/1/2/3 刹车控制 1: 使能; 0: 禁止。
4	TIM3RCLR	VC1 filter 结果输出到 TIM3 REFCLR 使能控制 1: 使能; 0: 禁止。
3	TIM2RCLR	VC1 filter 结果输出到 TIM2 REFCLR 使能控制 1: 使能; 0: 禁止。
2	TIM1RCLR	VC1 filter 结果输出到 TIM1 REFCLR 使能控制 1: 使能; 0: 禁止。
1	TIMO_RCLR	VC1 filter 结果输出到 TIM0 REFCLR 使能控制 1: 使能; 0: 禁止。
0	INV_Timer	VC1 filter 结果输出反向到各 TIM0/1/2/3/REFCLR

		1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
--	--	--------------------------------

33.6.6 VC Interrupt Register (VC_IFR)

偏移地址 0x024

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										VC2_Filter	VC2_INTF	VC1_Filter	VC0_Filter	VC1_INTF	VC0_INTF
										RO	RW0	RO	RO	RW0	RW0

位	标记	功能描述
31:4	Reserved	保留
5	VC2_Filter	VC2 Filter 后的状态
4	VC2_INTF	VC2 中断标志, 1 发生 VC2 中断; 0 未发生中断; 写 0 清除中断标志, 写 1 无效
3	VC1_Filter	VC1 Filter 后的状态
2	VC0_Filter	VC0 Filter 后的状态
1	VC1_INTF	VC1 中断标志, 1 发生 VC1 中断; 0 未发生中断; 写 0 清除中断标志, 写 1 无效
0	VC0_INTF	VC0 中断标志, 1 发生 VC0 中断; 0 未发生中断; 写 0 清除中断标志, 写 1 无效

33.6.7 VC2 Configuration Register (VC2_CR)

偏移地址 0x150

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															EN R/W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	level	rising	falling	debounce_time	FLTEN		n_sel		p_sel						
R/W	R/W	R/W	R/W	R/W	R/W		R/W		R/W						R/W

位	标记	功能描述
31:17	Reserved	保留
16	EN	电压比较器使能 1: 使能电压比较器 0: 关闭电压比较器
15	IE	VC 中断使能 1: 使能; 0: 禁止
14	level	VC 输出信号触发中断选择 1: 使能高电平触发 INT flag 0: 禁止高电平触发 INT flag
13	rising	VC 输出信号触发中断选择 1: 使能上升沿触发 INT flag 0: 禁止上升沿触发 INT flag
12	falling	VC 输出信号触发中断选择 1: 使能下降沿触发 INT flag 0: 禁止下降沿触发 INT flag
11:9	debounce_time	VC 输出滤波时间配置 111: 滤波时间大约为 28.8ms 110: 滤波时间大约为 7.2ms 101: 滤波时间大约为 1.8ms 100: 滤波时间大约为 450us 011: 滤波时间大约为 112us 010: 滤波时间大约为 28us 001: 滤波时间大约为 14us 000: 滤波时间大约为 7us 注意: 滤波时间的配置只有在 FLTEN=1 时才有效。
8	FLTEN	1: 启动 VC 滤波 0: VC 无滤波
7:4	N_SEL	电压比较器“-”端输入选择

		0000: select channel 0 input PA5 0001: select channel 1 input PB1 0010: select channel 2 input PE11 0011: select channel 3 input PE15 0100: select channel 4 input PB11 0101: select channel 5 input PB14 0110: select channel 6 input PD10 0111: select channel 7 input PD11 1000: select channel 8 input PC7 1001: DAC0 1010: DAC1 1011: NC 1100: 内置温度传感器输出电压 1101: Reserved 1110: ADC 的 REF (使用需要使能 ADC) 1111: LDO 的输出电压
3:0	P_SEL	电压比较器“+”端输入选择 0000: select channel 0 input PA5 0001: select channel 1 input PB1 0010: select channel 2 input PE9 0011: select channel 3 input PE10 0100: select channel 4 input PE11 0101: select channel 5 input PE13 0110: select channel 6 input PE14 0111: select channel 7 input PE15 1000: select channel 8 input PB11 1001: select channel 9 input PB14 1010: select channel 10 input PD9 1011: select channel 11 input PD10 1100: select channel 12 input PD11 1101: select channel 13 input PC7 1110: DAC0 1111: DAC1

33.6.8 VC2 Output Configuration Register (VC2_OUT_CFG)

偏移地址 0x154

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
brake	TIM6	INV_Timer6	TIM5	INV_Timer5	TIM4	INV_Timer4	Reserved		TIMB_K	TIM3_RCLR	TIM2_RCLR	TIM1_RCLR	TIM0_RCLR	INV_Timer	

位	标记	功能描述
31:16	Reserved	保留
15	brake	VC2 作为 Advanced Timer 刹车控制 1: 使能; 0: 禁止。
14	TIM6	VC2 filter 结果输出到 TIM6 捕获输入 CHB 使能 1: 使能; 0: 禁止。
13	INV_TIM6	VC2 filter 结果输出到 TIM6 反向使能 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
12	TIM5	VC2 filter 结果输出到 TIM5 捕获输入 CHB 使能 1: 使能; 0: 禁止。
11	INV_TIM5	VC2 filter 结果输出到 TIM5 反向使能 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
10	TIM4	VC2 filter 结果输出到 TIM4 捕获输入 CHB 使能 1: 使能; 0: 禁止。
9	INV_TIM4	VC2 filter 结果输出到 TIM4 反向使能 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。
8:6	Res.	保留
5	TIMBK	VC2 filter 结果输出到 Timer0/1/2/3 刹车控制 1: 使能; 0: 禁止。
4	TIM3RCLR	VC2 filter 结果输出到 TIM3 REFCLR 使能控制 1: 使能; 0: 禁止。
3	TIM2RCLR	VC2 filter 结果输出到 TIM2 REFCLR 使能控制 1: 使能; 0: 禁止。
2	TIM1RCLR	VC2 filter 结果输出到 TIM1 REFCLR 使能控制 1: 使能; 0: 禁止。
1	TIM0RCLR	VC2 filter 结果输出到 TIM0 REFCLR 使能控制 1: 使能; 0: 禁止。
0	INV_Timer	VC2 filter 结果输出反向到各 TIM0/1/2/3/REFCLR 1: 使能反向; 0: 禁止反向, 输入与 VC 输出同向。

34 Low Voltage Detector (LVD)

34.1 LVD Introduction

LVD 可用于监测 VCC 及芯片管脚的电压。当被监测电压与 LVD 阈值的比较结果满足触发条件时，LVD 会产生中断或复位信号，用户可根据该信号执行一些紧急任务。

LVD 具有以下特性：

- 4 路监测源，AVCC、PC13、PB08、PB07；
- 16 阶阈值电压，1.8V~3.3V 可选；
- 8 种触发条件，高电平、上升沿、下降沿组合；
- 2 种触发结果，复位、中断；
- 8 阶滤波配置，防止误触发；
- 具备迟滞功能，强力抗干扰。

34.2 LVD Block Diagram

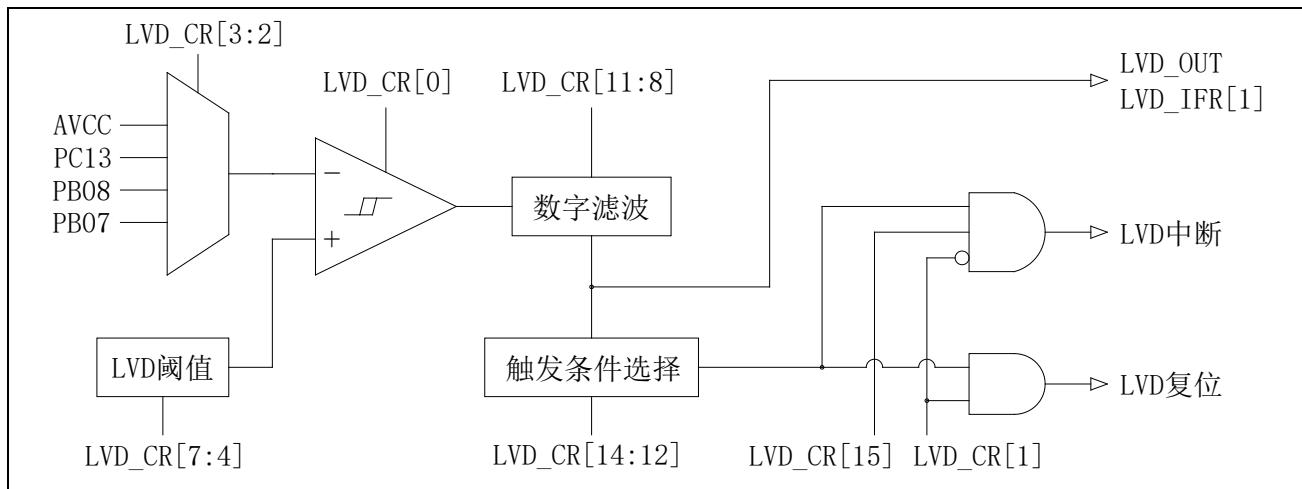


图 34-1 LVD 框图

34.3 Hysteresis Function

LVD 内置的电压比较器具有迟滞功能，可以增强芯片的抗干扰能力。LVD 的输出信号会等到输入信号高于或低于阈值电压 20mV 后才发生翻转，其输入输出信号示意如下：

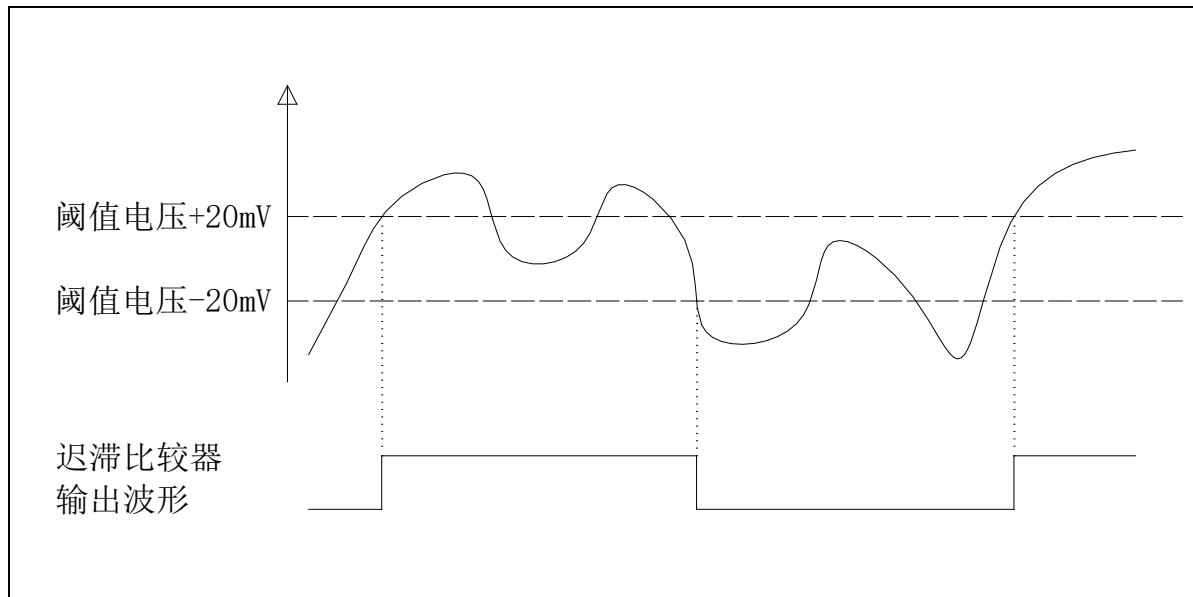


图 34-2 LVD 迟滞响应

34.4 Digital Filtering

如果芯片的工作环境恶劣，迟滞比较器的输出会出现噪声信号。使能数字滤波模块，则迟滞比较器的输出波形中脉宽小于 LVD_CR.Debounce_time 的噪声信号都可以被滤除。禁止数字滤波模块，则数字滤波模块的输入输出信号相同。

从寄存器 LVD_IFR[1]可以读出数字滤波后的信号电平；当 GPIO 功能配置为 LVD_OUT 时，数字滤波后的信号可以从 GPIO 输出以方便测量。

使能数字滤波模块，滤波示意如下所示：

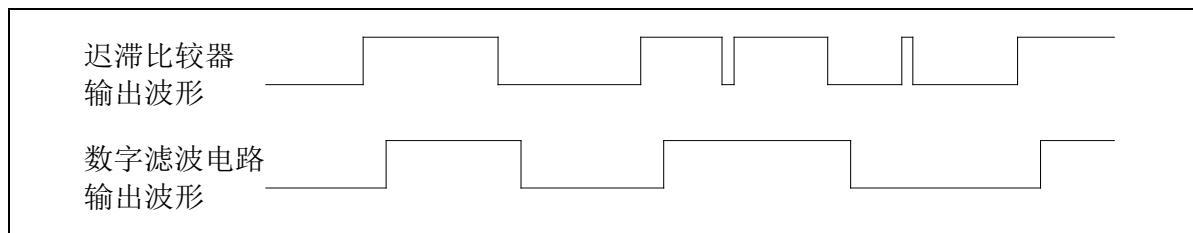


图 34-3 LVD 滤波输出

34.5 Configuration Examples

34.5.1 LVD Configuration for Low Voltage Reset

在本模式下，监测电压低于阈值电压时复位 MCU。

配置方法如下所示：

Step1：配置 LVD_CR.Source_sel，选择待监测的电压来源。

Step2：配置 LVD_CR.VTDS，选择 LVD 的阈值电压。

Step3：配置 LVD_CR.Debounce_time，选择 LVD 滤波时间。

Step4：配置 LVD_CR.FLTEN，使能 LVD 滤波。

Step5：设置 LVD_CR.HTEN 为 1，选择高电平触发 LVD 动作。

Step6：设置 LVD_CR.ACT 为 1，选择 LVD 动作为复位。

Step7：设置 LVD_CR.LVDEN 为 1，使能 LVD。

34.5.2 LVD Configuration for Voltage Change Interrupt

在本模式下，监测电压高于或低于阈值电压时产生中断。

配置方法如下所示：

Step1：配置 LVD_CR.Source_sel，选择待监测的电压来源。

Step2：配置 LVD_CR.VTDS，选择 LVD 的阈值电压。

Step3：配置 LVD_CR.Debounce_time，选择 LVD 滤波时间。

Step4：配置 LVD_CR.FLTEN，使能 LVD 滤波。

Step5：设置 LVD_CR.RTEN 和 LVD_CR.FTEN 为 1，选择电平变化触发 LVD 动作。

Step6：设置 LVD_CR.ACT 为 0，选择 LVD 动作为中断。

Step7：设置 LVD_CR.IE 为 1，使能 LVD 中断。

Step8：使能 NVIC 中断向量表中的 LVD 中断。

Step9：设置 LVD_CR.LVDEN 为 1，使能 LVD。

Step10：在 LVD 的中断服务程序中执行用户需要进行的操作；退出中断服务程序前向 LVD_IFR 写入 0x00 以清除中断标志。

34.6 LVD Registers

基地址 0x40002400

表 34-1 LVD 寄存器

寄存器	偏移地址	描述
LVD_CR	0x028	LVD 配置寄存器
LVD_IFR	0x02C	LVD 中断标志寄存器

34.6.1 LVD Configuration Register (LVD_CR)

偏移地址 0x028

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	HTEN	RREN	FTEN	Debounce_time	FLTEN	VTDS	Source_sel	ACT	LVDEN						
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW						

位	标记	功能描述
31:16	Reserved	保留
15	IE	LVD 中断使能 1: 使能; 0: 禁止。
14	HTEN	高电平触发使能 (被监测电压低于阈值电压) 1: 使能; 0: 禁止。
13	RREN	上升沿触发使能 (被监测电压从高于阈值电压变为低于阈值电压) 1: 使能; 0: 禁止。
12	FTEN	下降沿触发使能 (被监测电压从低于阈值电压变为高于阈值电压) 1: 使能; 0: 禁止。
11:9	Debounce_time	数字滤波时间配置 111: 滤波时间大约为 28.8ms 110: 滤波时间大约为 7.2ms 101: 滤波时间大约为 1.8ms 100: 滤波时间大约为 450us 011: 滤波时间大约为 112us 010: 滤波时间大约为 28us 001: 滤波时间大约为 14us 000: 滤波时间大约为 7us 注: 滤波时间仅在 FLTEN 为 1 时有效。
8	FLTEN	数字滤波使能配置 1: 使能数字滤波 0: 禁止数字滤波
7:4	VTDS	LVD 阈值电压选择 1111: 3.3v 1110: 3.2v

		1101: 3.1v 1100: 3.0v 1011: 2.9v 1010: 2.8v 1001: 2.7v 1000: 2.6v 0111: 2.5v 0110: 2.4v 0101: 2.3v 0100: 2.2v 0011: 2.1v 0010: 2.0v 0001: 1.9v 0000: 1.8v
3:2	Source_sel	LVD 监测来源选择 11: PB07 端口输入电压 10: PB08 端口输入电压 01: PC13 端口输入电压 00: AVCC 电源电压
1	ACT	LVD 触发动作选择 1: 系统复位 0: NVIC 中断
0	LVDEN	LVD 使能控制 1: 使能 LVD 0: 禁止 LVD

34.6.2 LVD Interrupt Register (LVD_IFR)

偏移地址 0x02C

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															LVD_Filter INTF
															RO R/W0

位	标记	功能描述
31:2	Reserved	保留
1	LVD_Filter	LVD Filter 后的状态
0	INTF	LVD 中断标志： 1: 发生 LVD 中断; 0: 未发生中断; 写 0 清除中断标志, 写 1 无效。

35 Operational Amplifier (OPA)

OPA 模块适用于简易放大器和电压跟随器应用。内部的五个运放可以使用外部电阻进行级联。OPA 的输入范围是 0V 到 AVCC，输出范围是 0.1V 到 AVCC-0.2V。

35.1 OPA Features

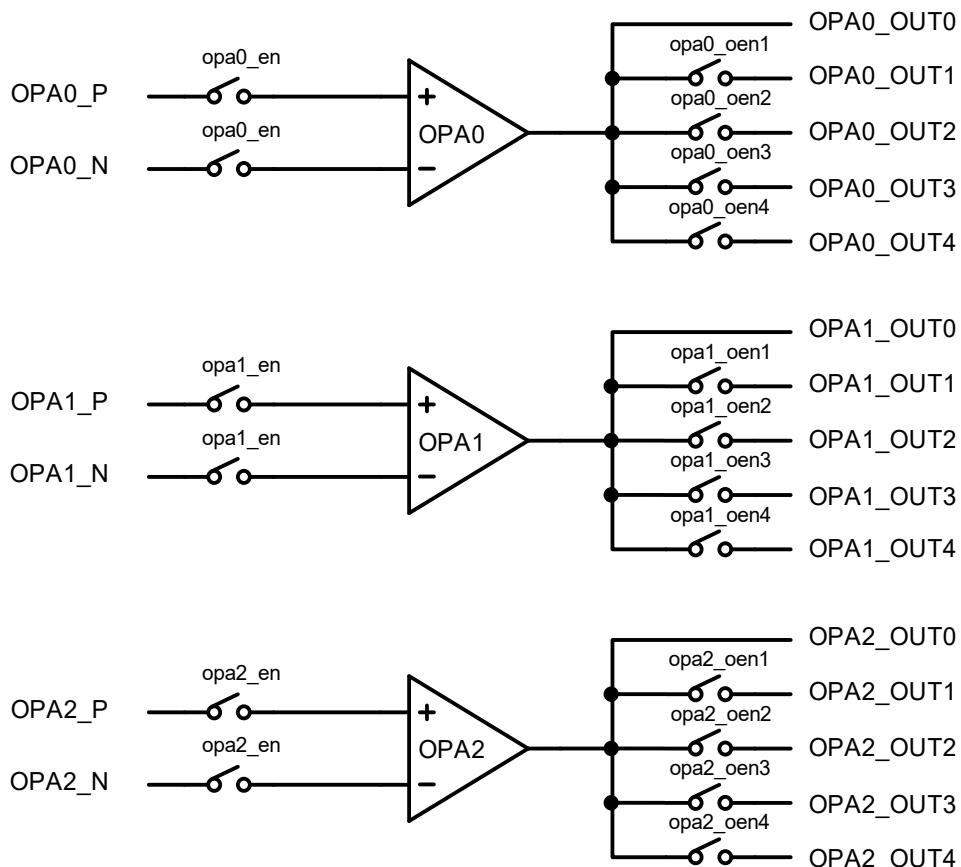
- 5 个独立配置运放
- OPA 的输入范围是 0 到 AVCC，输出范围是 0.1V 到 AVCC-0.2V 可编程增益
- 可配置为以下模式
 - 通用运放模式 (*general purpose OPA*)
 - DAC 电压跟随器

35.2 OPA Functional Description

5 个 OPA 可以通过使用外部元件组成放大器来放大小信号模拟输入信号，输出为放大后的信号。

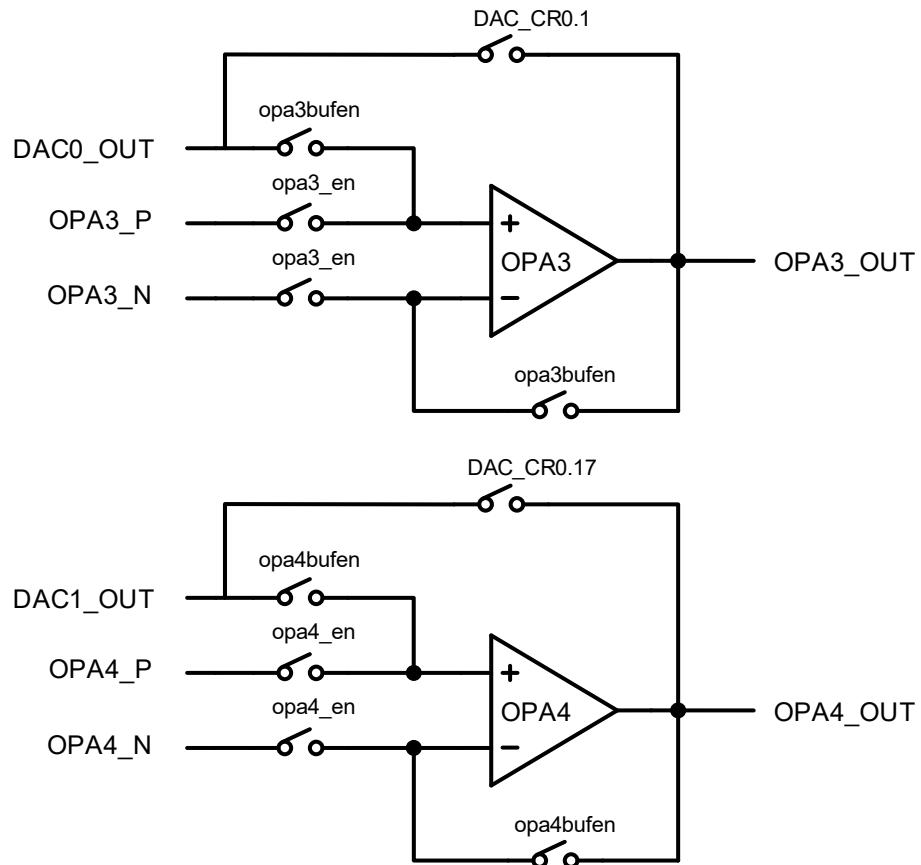
35.2.1 OPA0/1/2 (Generic OPA)

OPA0/1/2 的示意图如下：



35.2.2 OPA3/4 --- (Can be reused as DAC buffer*2)

OPA3/4 的示意图如下：



35.3 OPA Zero Calibration

OPA 支持不同模式的自动校零功能，可以软件自己控制，也可以硬件控制。高精度 OPA 应用，在每次使用 OPA 前，需要 OPA 校零。校零建议时间 10us 以上。校零后请立即采样。

35.3.1 Software Control

- 使能需要校零的 OPA 的 OPA_CRx.AZEN 为 1
- 设置 OPA_CR. CLK_SW_SET 为 1
- 软件写 OPA_CR. AZ_PULSE 为 1
- 设定校零时间，等待校零时间结束，清除 OPA_CR. AZ_PULSE 为 0

35.3.2 Software Trigger Control

- 使能需要校零的 OPA 的 OPA_CRx.AZEN 为 1
- 选择需要校零的时间参数 OPA_CR.CLK_SEL
- 设置软件触发 OPA_CR.TRIGGER

35.3.3 ADC Trigger Control

- 使能需要校零的 OPA 的 OPA_CRx.AZEN 为 1
- 选择需要校零的时间参数 OPA_CR.CLK_SEL
- 使能 ADC 触发 ADCTR_EN

35.4 OPA Registers

基地址 0x40002400

表 35-1 OPA 寄存器

寄存器	偏移地址	描述
OPA_CR0	0x030	OP0 控制寄存器
OPA_CR1	0x034	OP1 控制寄存器
OPA_CR2	0x038	OP2 控制寄存器
OPA_CR	0x03C	OP 控制寄存器

35.4.1 OPA Output Enable Register (OPA_CR0)

偏移地址 0x030

复位值 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	OP2 OEN 4	OP2 OEN 4	OP2 OEN 3	OP2 OEN 2	OP2 OEN 1	OP1 OEN 4	OP1 OEN 3	OP1 OEN 2	OP1 OEN 1	OP0 OEN 4	OP0 OEN 3	OP0 OEN 2	OP0 OEN 1		

位	标记	功能描述
31:12	Reserved	保留
11	OP2OEN4	OP2输出4使能
10	OP2OEN3	OP2输出3使能
9	OP2OEN2	OP2输出2使能
8	OP2OEN1	OP2输出1使能
7	OP1OEN4	OP1输出4使能
6	OP1OEN3	OP1输出3使能
5	OP1OEN2	OP1输出2使能
4	OP1OEN1	OP1输出1使能
3	OP0OEN4	OP0输出4使能
2	OP0OEN3	OP0输出3使能
1	OP0OEN2	OP0输出2使能
0	OP0OEN1	OP0输出1使能

35.4.2 OPA Control Register (OPA_CR1)

偏移地址 0x034

复位值 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		OP4 BUF EN	OP3 BUF EN	EN4	EN3	EN2	EN1	ENO	AZ EN4	AZ EN3	AZ EN2	AZ EN1	AZ ENO		
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

位	标记	功能描述
31:12	Reserved	保留
11	OP4BUFEN	DAC1使用OP4单位增加缓存使能，与OPA4使能不能同时使能
10	OP3BUFEN	DAC0使用OP3单位增加缓存使能，与OPA3使能不能同时使能
9	EN4	OPA4使能
8	EN3	OPA3使能
7	EN2	OPA2使能
6	EN1	OPA1使能
5	ENO	OPA0使能
4	AZEN4	OPA4 自动校零使能
3	AZEN3	OPA3 自动校零使能
2	AZEN2	OPA2 自动校零使能
1	AZEN1	OPA1 自动校零使能
0	AZENO	OPA0 自动校零使能

35.4.3 OPA Configuration Register (OPA_CR2)

偏移地址 0x038

复位值 0x0000 36DB

不需要更改配置，使用缺省值就可以

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		BIASSEL4		BIASSEL3		BIASSEL2		BIASSEL1		BIASSEL0					
		RW													

位	标记	功能描述
31:15	Reserved	保留
14:12	BIASSEL4	OPA4 偏置电流选择 客户不需要更改配置，使用缺省值就可以
11:9	BIASSEL3	OPA3 偏置电流选择 客户不需要更改配置，使用缺省值就可以
8:6	BIASSEL2	OPA2 偏置电流选择 客户不需要更改配置，使用缺省值就可以
5:3	BIASSEL1	OPA1 偏置电流选择 客户不需要更改配置，使用缺省值就可以
2:0	BIASSEL0	OPA0 偏置电流选择 客户不需要更改配置，使用缺省值就可以

35.4.4 OPA Zero Calibration Control Register (OPA_CR)

偏移地址 0x03C

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CLK_SEL	CLK_SW_SET	AZ_PULSE	TRIGGER	ADCTR_EN						
					RW	RW	RW	W1	RW						

位	标记	功能描述
31:8	Reserved	保留
7:4	CLK_SEL	自动校准脉冲宽度设置 0000:1个PCLK周期 0001:2个PCLK周期 0010:4个PCLK周期 0011:8个PCLK周期 0100:16个PCLK周期 0101:32个PCLK周期 0110:64个PCLK周期 0111:128个PCLK周期 1000:256个PCLK周期 1001:512个PCLK周期 1010:1024个PCLK周期 1011:2048个PCLK周期 11xx:4096个PCLK周期 建议自动校准时间为10us
3	CLK_SW_SET	自动校零选择 1: 软件校准使能 0: 软件校准禁止，软件触发校准使能
2	AZ_PULSE	软件校准 1: 校准 0: 无效
1	TRIGGER	软件触发校准设置，自动清零 1: 软件触发校准 0: 无效
0	ADCTR_EN	ADC启动触发OPA自动校准使能 1: 使能 0: 禁止 高精度OPA应用，在每次使用OPA前，需要OPA校零。校零建议时间10us以上。校零后请立即采样。

36 Simulate other registers

地址 0x40002400

寄存器	偏移地址	描述
BGR_CR	0x000	BGR 控制寄存器

36.1 BGR Configuration Register (BGR_CR)

偏移地址 0x000

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TS_EN	BGR_EN
														RW	RW

位	标记	功能描述
31:2	Reserved	保留
1	TS_EN	内置温度传感器使能控制 1: 使能内部温度传感器 0: 禁止内部温度传感器 注意: 温度传感器使能 20us 后, 才能输出稳定信号。
0	BGR_EN	BGR 使能控制 1: 使能 BGR 0: 禁止 BGR 注意: 1) Peri_clken0.ADC 为 1 时, 才可以操作此寄存器。 2) BGR 使能 20us 后, 才能输出稳定的高精度参考电压。BGR 稳定后才可以被其它模块使用, 故用户操作中应加入等待 BGR 稳定的步骤。 3) 当使用 ADC/OPA/PLL 时, 必须使能 BGR。 4) 当使用 VC 时, 需根据 VC 寄存器的配置决定是否使能 BGR。

37 SWD debug interface

本系列使用 ARM Cortex-M0+内核，该内核具有硬件调试模块 SWD，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。内核停止时，内核的内部状态和系统的外部状态都可以在 IDE 中进行查询。完成查询后，内核和外设可以被复原，程序将继续执行。当本系列微控制器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。

注意：

- SWD 在 DeepSleep 模式下不能工作，请在 Active 和 Sleep 模式下进行调试操作。

37.1 SWD Debug Additional Functions

本产品使用了 ARM Cortex-M0+ CPU，该内核包含用于高级调试功能的硬件扩展，因此本产品所拥有的调试功能与 Cortex-M0+一致。调试扩展允许内核可以在取指(指令断点)或取访问数据(数据断点)时停止内核。内核停止时，可以查询内核的内部状态和系统的外部状态。查询完成后，将恢复内核和系统并恢复程序执行。

当调试主机与 MCU 相连并进行调试时，将使用调试功能。

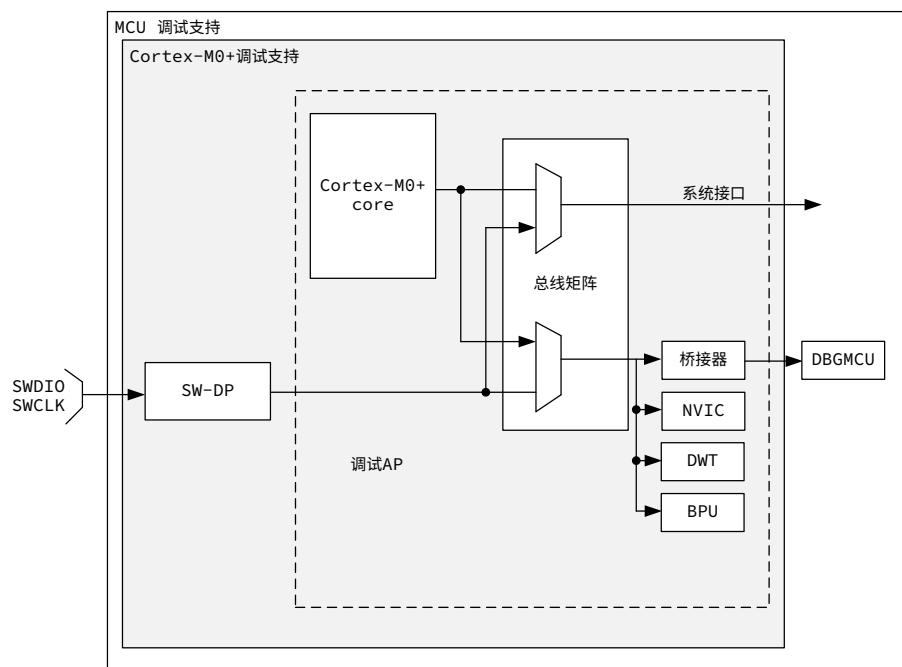


图 37-1 调试支持框图

Cortex®-M0+ 内核中内置的调试功能是 ARM® CoreSight 设计套件的一部分。

ARM® Cortex®-M0+内核提供集成片上调试支持。它包括：

- SW-DP：串行线
- BPU：断点单元
- DWT：数据观察点触发

注：

- 有关 ARM® Cortex®-M0+ 内核支持的调试功能的详细信息，请参见 Cortex®-M0+ 技术参考手册。

37.2 ARM® Reference Documentation

- Cortex®-M0+技术参考手册 (TRM)
可从 www.infocenter.arm.com 获取。
- ARM® 调试接口 V5
- ARM® CoreSight 设计套件版本 r1p1 技术参考手册

37.3 Debug Port Pinouts

37.3.1 SWD Port Pins

本系列的 SWD 接口，需要用到 2 个引脚，如下表所示。

SWD 端口名称	调试功能	引脚分配
SWCLK	串行时钟	PA14
SWDIO	串行数据输入/输出	PA13

37.3.2 SW-DP Pin Assignment

如果烧录程序时使能了【加密芯片】选项，则上电后 SWD 调试功能被禁止。如果烧录程序时没有使能【加密芯片】选项，则上电后 PA13/PA14 引脚均被初始化为可被调试器使用的专用引脚。用户可设置 SYSCTRL1 .SWD_USE_IO 寄存器来禁止 SWD 引脚的调试功能，SWD 引脚将被释放以用作普通 GPIO。SWD 引脚的配置与功能小结如下表所示：

【加密芯片】选项	SWD_USE_IO 配置	PA13/PA14 功能
加密	0	NA
加密	1	GPIO
不加密	0	SWD
不加密	1	GPIO

37.3.3 Internal Pull-up on SWD Pin

用户软件释放 SW I/O 后，GPIO 控制器便会控制这些引脚。GPIO 控制寄存器的复位状态会将 I/O 置于等效的状态：

- SWDIO：输入上拉
- SWCLK：输入上拉

由于内置上拉和下拉电阻，因此无需添加外部电阻。

37.4 SWD Port

37.4.1 Introduction to SWD Protocol

此同步串行协议使用两个引脚：

- SWCLK：从主机到目标的时钟
- SWDIO：双向

利用该协议，可以同时读取和写入两组寄存器组 (DPACC 寄存器组和 APACC 寄存器组)。传输数据时，LSB 在前。

对于 SWDIO 双向管理，必须在电路板上对线路进行上拉 (ARM® 建议采用 100 k)。这些上拉电阻可在内部配置。无需外部上拉电阻。

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。

37.4.2 SWD Protocol Sequence

每个序列包括三个阶段：

1. 主机发送的数据包请求 (8 位)
2. 目标发送的确认响应 (3 位)
3. 主机或目标发送的数据传输阶段 (33 位)

位	名称	说明
0	启动	必须为 1
1	APnDP	0: DP 访问；1: AP 访问
2	RnW	0: 写请求；1: 读请求
4:3	A[3:2]	DP 或 AP 寄存器的地址字段
5	奇偶校验	前面几位的单位奇偶校验
6	停止	0
7	驻留	不受主机驱动。由于存在上拉，因此必须由目标读为 1

有关 DPACC 和 APACC 寄存器的详细说明，请参见 Cortex®-M0+ TRM。

数据包请求后面始终为转换时间（默认 1 位），此时主机和目标都不会驱动。

位	名称	说明
0	ACK	001: FAULT 010: WAIT 100: OK

仅当发生 READ 事务或者接收到 WAIT 或 FAULT 确认时，ACK 响应后才必须是转换时间。

位	名称	说明
0:31	WDATA 或 RDATA	写入或读取数据
32	奇偶校验	32 个数据位的单奇偶校验

仅当发生 READ 事务时，DATA 传输后才必须是转换时间。

37.4.3 SW-DP State Machine (Reset, Idle State, ID Code)

SW-DP 的状态机有一个用于标识 SW-DP 的内部 ID 代码。该代码符合 JEP-106 标准。此 ID 代码是默认的 ARM® 代码，设置为 **0x0BB11477**（相当于 Cortex®-M0+）。

注意：

- 在目标读取此 ID 代码前，SW-DP 状态机是不工作的。
- 在上电复位后或者线路处于高电平超过 50 个周期后，SW-DP 状态机处于复位状态。
- 如果在复位状态后线路处于低电平至少两个周期，SW-DP 状态机处于空闲状态。
- 复位状态后，该状态机必须首先进入空闲状态，然后对 DP-SW ID CODE 寄存器执行读访问。否则，目标将在另一个事务上发出 FAULT 确认响应。

有关 SW-DP 状态机的更多详细信息，请参见 Cortex®-M0+ TRM 和 CoreSight 设计套件 r1p0TRM。

37.4.4 DP and AP Read/Write Access

- 不延迟对 DP 的读访问：可以立即发送目标响应（如果 ACK=OK），也可以延迟发送目标响应（如果 ACK=WAIT）。
- 延迟对 AP 的读访问。这意味着会在下次传输时返回访问结果。如果要执行的下次访问不是 AP 访问，则必须读取 DP-RDBUFF 寄存器来获取结果。
- 每次进行 AP 读访问或 RDBUFF 读请求时都会更新 DP-CTRL/STAT 寄存器的 READOK 标志，以便了解 AP 读访问是否成功。
- SW-DP 有写缓冲区（用于 DP 或 AP 写入），这样即使在其它操作仍未完成时，也可以接受写入操作。如果写缓冲区已满，则目标确认响应为 WAIT。但 IDCODE 读取、CTRL/STAT 读取或 ABORT 写入除外，这几项操作在写缓冲区已满时也会被接受。
- 由于存在异步时钟域 SWCLK 和 HCLK，因此写操作后（奇偶校验位后）还需要两个额外的 SWCLK 周期，以使写入操作在内部生效。应在将线路驱动为低电平时（空闲状态）应用这些周期。

在写 CTRL/STAT 寄存器以提出一个上电请求时，这一点特别重要。否则下一个操作（在内核上电后才有效的操作）会立即执行，这将导致失败。

37.4.5 SW-DP Register

当 APnDP=0 时能够访问这些寄存器：

A[3:2]	RW	SELECT 寄存器的 CTRLSEL 位	寄存器	注释
00	读取		IDCODE	制造商代码设置为 Cortex®-M0+ 的默认 ARM® 代码。 0x0BB11477 (标识 SW-DP)
00	写		ABORT	
01	读 / 写	0	DP-CTRL/STAT	目的： - 请求系统或调试上电 - 配置 AP 访问的传输操作 - 控制比较和验证操作 - 读取一些状态标志 (上溢和上电确认)
01	读 / 写	1	WIRE CONTROL	用于配置物理串行端口协议 (如转换时间的 持续时间)
10	读取		READ RESEND	允许从已损坏的调试软件传输中恢复读取数据，无需重复执行原始 AP 传输。
10	写		SELECT	用于选择当前访问端口和活动的 4 字寄存器 窗口
11	读 / 写		READ BUFFER	由于已发出 AP 访问，因此该读缓冲区非常有用 (在执行下个 AP 事务时提供读取 AP 请求的结果)。此读取缓冲区捕获 AP 中的数据，显示为

前一次读取的结果，无需启动新操作。

37.4.6 SW-AP Register

当 APnDP=1 时能够访问这些寄存器：

有多个 AP 寄存器，这些寄存器按以下组合进行寻址：

- 移位值 A[3:2]
- DP SELECT 寄存器的当前值

地址	A[3:2] 值	说明
0x0	00	保留，必须保持复位值。
0x4	01	DP CTRL/STAT 寄存器。用于： - 请求系统或调试上电 - 配置 AP 访问的传输操作 - 控制比较和验证操作 - 读取一些状态标志（上溢和上电确认）
0x8	10	DP SELECT 寄存器：用于选择当前访问端口和活动的 4 字寄存器窗口。 - 位 31:24：APSEL：选择当前 AP (select the current AP) - 位 23:8：保留 - 位 7:4：APBANKSEL：在当前 AP 上选择活动的 4 字寄存器窗口 - 位 3:0：保留
0xC	11	DP RDBUFF 寄存器：用于通过调试器在执行一系列操作后获取最后结果 (无需请求新的 JTAG-DP 操作)

37.5 Core Debugging

通过内核调试寄存器调试内核。通过调试访问端口调试访问这些寄存器。它由四个寄存器组成：

寄存器	说明
DHCSR	32 位调试停止控制和状态寄存器 此寄存器提供有关处理器状态的信息，能够使内核进入调试停止状态并提供处理器步进功能。
DCRSR	17 位调试内核寄存器选择器寄存器： 此寄存器选择需要进行读写操作的处理器寄存器。
DCRDR	32 位调试内核寄存器数据寄存器： 此寄存器保存在寄存器与 DCRSR（选择器）寄存器选择的处理器之间读取和写入的数据。
DEMCR	32 位调试异常和监视控制寄存器： 此寄存器提供向量捕获和调试监视控制。

这些寄存器在系统复位时不复位。它们只能通过上电复位来复位。有关更多详细信息，请参见 Cortex®-M0+ TRM。

为了在复位后立即使内核进入调试停止状态，必须：

- 使能调试和异常监视控制寄存器的位 0 (VC_CORRESET)
- 使能调试停止控制和状态寄存器的位 0 (C_DEBUGEN)

37.6 BPU (Breakpoint Unit)

Cortex®-M0+ BPU 实现提供四个断点寄存器。

37.6.1 BPU Function

处理器断点实现了基于 PC 的断点功能。

有关 BPU CoreSight 标识寄存器及其地址和访问类型的更多信息，请参见 ARMv6-M ARM® 和 ARM® CoreSight 组件技术参考手册。

37.7 DWT (Data Observation Point)

Cortex®-M0+ DWT 实现提供了两个观察点寄存器组。

37.7.1 DWT Function

处理器观察点实现了数据地址和基于 PC 的观察点功能(即 PC 采样寄存器),并支持比较器地址掩码,如 ARMv6-M ARM® 中所述。

37.7.2 DWT Program Counter Sampling Register

实现数据观察点单元的处理器还实现了 ARMv6-M 可选 DWT 程序计数器采样寄存器(DWT_PCSR)。此寄存器允许调试程序定期采样 PC, 无需停止处理器。这可提供粗略分析。有关更多信息, 请参见 ARMv6-M ARM®。

Cortex®-M0+ DWT_PCSR 记录通过条件代码和指令以及未通过条件代码的指令。

37.8 MCU Debug Group (DBG)

MCU 调试组件帮助调试器为以下各项提供支持:

- 低功耗模式
- 断点期间的定时器、看门狗的时钟控制

37.8.1 Debug Support for Low-Power Modes

要进入低功耗模式, 必须执行指令 WFI 或 WFE。

MCU 支持多个低功耗模式, 这些模式可以禁止 CPU 时钟或降低 CPU 功耗。

内核不允许在调试会话期间关闭 FCLK 或 HCLK。由于调试期间需要使用它们进行调试连接, 因此其必须保持激活状态。MCU 集成了特殊方法, 允许用户在低功耗模式下调试软件。

37.8.2 Debug support for timers and watchdogs

断点期间, 必须选择定时器和看门狗的计数器的行为方式:

- 在产生断点时, 计数器继续计数。例如, 当 PWM 控制电机时, 通常需要这种方式。
- 在产生断点时, 计数器停止计数。用于看门狗时需要这种方式。

37.9 Debug Mode Module Working Status Control (DEBUG_ACTIVE)

复位值 0x00001FFF (仅在 SWD 调试模式下, 此寄存器设置才起作用)

偏移地址: 0x4000_2038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16												
Reserved																											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0												
Reserved		LPTIM1		TIM3		Res.		RTC		WDT		PCA		TIM6		TIM5		TIM4		LPTIMO		TIM2		TIM1		TIM0	
RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW			

位	标记	功能描述
31:13	Reserved	保留
12	LPTIM1	调试时, LpTimer1 计数功能配置 1: 在 SWD 调试界面下, 暂停 LPTimer1 计数功能 0: 在 SWD 调试界面下, LPTimer1 正常计数功能
11	TIM3	调试时, Timer3 计数功能配置 1: 在 SWD 调试界面下, 暂停 Timer3 计数功能 0: 在 SWD 调试界面下, Timer3 正常计数功能
10	Reserved	保留
9	RTC	调试时, RTC 计数功能配置 1: 在 SWD 调试界面下, 暂停 RTC 计数功能 0: 在 SWD 调试界面下, RTC 正常计数功能
8	WDT	调试时, WDT 计数功能配置 1: 在 SWD 调试界面下, 暂停 WDT 计数功能 0: 在 SWD 调试界面下, WDT 正常计数功能
7	PCA	调试时, PCA 计数功能配置 1: 在 SWD 调试界面下, 暂停 PCA 计数功能 0: 在 SWD 调试界面下, PCA 正常计数功能
6	TIM6	调试时, Timer6 计数功能配置 1: 在 SWD 调试界面下, 暂停 Timer 计数功能, 暂停后再次运行需要初始化计数器计数值, 否则 PWM 输出相位可能不对。 0: 在 SWD 调试界面下, Timer 正常计数功能
5	TIM5	调试时, Timer5 计数功能配置 1: 在 SWD 调试界面下, 暂停 Timer 计数功能, 暂停后再次运行需要初始化计数器计数值, 否则 PWM 输出相位可能不对。 0: 在 SWD 调试界面下, Timer 正常计数功能
4	TIM4	调试时, Timer4 计数功能配置 1: 在 SWD 调试界面下, 暂停 Timer 计数功能, 暂停后再次运行需要初始化计数器计数值, 否则 PWM 输出相位可能不对。 0: 在 SWD 调试界面下, Timer 正常计数功能

3	LPTIM0	调试时， LpTimer0 计数功能配置 1: 在 SWD 调试界面下，暂停 LPTimer0 计数功能 0: 在 SWD 调试界面下，LPTimer0 正常计数功能
2	TIM2	调试时， Timer2 计数功能配置 1: 在 SWD 调试界面下，暂停 Timer 计数功能 0: 在 SWD 调试界面下，Timer 正常计数功能
1	TIM1	调试时， Timer1 计数功能配置 1: 在 SWD 调试界面下，暂停 Timer 计数功能 0: 在 SWD 调试界面下，Timer 正常计数功能
0	TIM0	调试时， Timer0 计数功能配置 1: 在 SWD 调试界面下，暂停 Timer 计数功能 0: 在 SWD 调试界面下，Timer 正常计数功能

38 Device electronic signature

电子签名存放在闪存存储器模块的系统存储区域，可以通过 SWD 或者 CPU 读取。它包含的芯片识别信息在出厂时编写，用户固件或者外部设备可以读取电子签名，用以自动匹配不同配置的 HC32Fxxx / HC32Lxxx 微控制器。

38.1 Product Unique Identifier (UID) Register (80 bits)

唯一身份标识符典型应用场景：

- 用作序列号
- 在对内部 Flash 进行编程前将 UID 与软件加密原语和协议结合使用时用作安全密钥以提高 Flash 中代码的安全性
- 激活安全自举过程等

80 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。80 位的唯一设备标识符也可以以单字节/半字/字等不同方式读取，然后使用自定义算法连接起来。

基址：0x0010 0E74

偏移地址	描述	UID Bits (80 bits)							
		7	6	5	4	3	2	1	0
0	Lot Number	UID[7:0]							
1		UID[15:8]							
2		UID[23:16]							
3		UID[31:24]							
4		UID[39:32]							
5		UID[47:40]							
6	X Coordinate on the wafer	UID[55:48]							
7	Y Coordinate on the wafer	UID[63:56]							
8	Wafer Number	UID[71:64]							
9	Rev ID	UID[79:72]							

38.2 Product Model Register

0x0010 0C60 ~ 0x0010 0C6F 存储了产品型号的 ASCII 码。如产品型号不足 16 字节，则以 0x00 进行填充。

例：484333324C3133364B38544100000000 所代表的产品型号为 HC32L136K8TA。

38.3 FLASH Capacity Register

基地址：0x0010 0C70

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FlashSize[31:16]															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FlashSize[15:0]															
R															

位	标记	功能描述
31:0	FlashSize	产品内置 Flash 的容量，以字节为单位 0x00008000 代表 Flash 容量为 32K Byte

38.4 RAM Capacity Register

基地址：0x0010 0C74

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

位	标记	功能描述
31:0	RamSize	产品内置 RAM 的容量，以字节为单位 0x00000800 代表 RAM 容量为 2K Byte

38.5 Pin Number Register

基地址: 0x0010 0C7A

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PinCount[15:0]								R							

位	标记	功能描述
15:0	PinCount	产品管脚的数量, 以只为单位 0x0020 代表产品管脚数量为 32

39 Appendix A SysTick Timer

39.1 SysTick Introduction

OS 要想支持多任务，就需要周期执行上下文切换，这样就需要有定时器之类的硬件资源打断程序执行。当定时器中断产生时，处理器就会在异常处理中进行 OS 任务调度，同时还会进行 OS 维护的工作。Cortex-M0 处理器中有一个称为 SysTick 的简单定时器，用于产生周期性的中断请求。

SysTick 为 24 位的定时器，并且向下计数。定时器的计数减到 0 后，就会重新装载一个可编程的数值，并且同时产生 SysTick 异常（异常编号为 15），该异常事件会引起 SysTick 异常处理的执行，这个过程是 OS 的一部分。

对于不需要 OS 的系统，SysTick 定时器也可以用作其他用途，比如定时、计时或者为需要周期执行的任务提供中断源。SysTick 异常的产生是可控的，如果异常被禁止，仍然可以用轮询的方法使用 SysTick 定时器，比如检查当前的计数值或者轮询计数标志。

39.2 Setting SysTick

由于 SysTick 定时器的重载值和当前值在复位时都是未定义的，为了防止产生异常结果，对 SysTick 的配置需要遵循一定的流程：

- Step1：配置 SysTick->CTRL. ENABLE 为 0，禁止 SysTick。
- Step2：配置 SysTick->CTRL. CLKSOURCE，选择 SysTick 的时钟源。
- Step3：配置 SysTick->LOAD，选择 SysTick 的溢出周期。
- Step4：向 SysTick->VAL 写入任意值，清零 SysTick->VAL 及 SysTick->CTRL. COUNTFLAG。
- Step5：配置 SysTick->CTRL. TICKINT 为 1，使能 SysTick 中断。
- Step6：配置 SysTick->CTRL. ENABLE 为 1，使能 SysTick。
- Step7：在中断服务程序中读取 SysTick->CTRL 以清除溢出标志。

注意：Systick 溢出周期为 SysTick->LOAD+1，配置示例如下：

时钟源	SysTick->LOAD	溢出周期
HCLK 4MHz	3999	1ms
XTL 32.768KHz	327	10.01ms

39.3 SysTick 寄存器

地址	名称	CMSIS 符号	全名
0XE000E010	SYS_CSR	SysTick->CTRL	SysTick 控制和状态寄存器

0XE000E014	SYS_RVR	SysTick->LOAD	SysTick 重装载寄存器
0XE000E018	SYS_CVR	SysTick->VAL	SysTick 当前值寄存器
0XE000E01C	SYS_CALIR	SysTick->CALIB	SysTick 校准值寄存器

39.3.1 SysTick Control and Status Register (CTRL)

位	符号	功能描述	类型	复位值
31:17	Reserved	-	-	-
16	COUNTFLAG	SysTick 定时器溢出标志 1: SysTick 定时器发生下溢出 0: SysTick 定时器未发生溢出 读该寄存器，可清除 COUNTFLAG 标志	RO	0
15:3	Reserved	-	-	-
2	CLKSOURCE	SysTick 时钟源选择 1: 使用内核时钟(HCLK) 0: 使用参考时钟(XTL) 注:当选择参考时钟作为 SysTick 的时钟源时, 需要同步使能 XTL 时钟源及 SysCtrl.PERI_CLKEN0.TICK	RW	0
1	TICKINT	SysTick 中断使能 1: 使能中断 0: 禁止中断	RW	0
0	ENABLE	SysTick 定时器使能 1: 使能 SysTick 0: 禁止 SysTick	RW	0

39.3.2 SysTick Reload Register (LOAD)

位	符号	功能描述	类型	复位值
31:24	Reserved	-	-	-
23:0	RELOAD	SysTick 定时器重载值	RW	未定义

39.3.3 SysTick Current Value Register (VAL)

位	符号	功能描述	类型	复位值
31:24	Reserved	-	-	-
23:0	CURRENT	读取该寄存器，获取 SysTick 定时器的当前计数值 写任意值到该寄存器，清零该寄存器及 COUNTFLAG	RW	未定义

39.3.4 SysTick Calibration Value Register (CALIB)

位	符号	功能描述	类型	复位值
31	NOREF	SysTick 当前计数时钟标志 1: 当前计数时钟为内核时钟 0: 当前计数时钟为外部参考时钟 (XTL)	RO	-
30	SKEW	TENMS 精度指示	RO	

		1: TENMS 值代表粗略的 10ms 0: TENMS 值代表精确的 10ms		
29:24	Reserved	-	-	-
23:0	TENMS	10 毫秒校准值	RO	-

40 Appendix B Document Conventions

40.1 List of register-related abbreviations

The following abbreviations are used in register descriptions:

RW: Read and write, software can read and write these bits.

RO: Read only, software can only read these bits.

WO: Write only, software can only write this bit. Invalid data will be returned when reading this bit.

W1: Write only 1, hardware automatically clears to 0, writing 0 is invalid.

ROW1: Software reads this bit as 0, writes 1 to clear this bit. Writing 0 has no effect on the value of this bit.

RW0: Software can read and write this bit, write 1 is invalid, write 0 to clear.

R1W0: Software reads this bit as 1, writes 0 to clear this bit. Writing 1 has no effect on the value of this bit.

RC: Software can read this bit. When reading this bit, it will be automatically cleared. Writing "0" has no effect on the value of this bit.

Res, Reserved: Reserved bit, must keep the reset value.

40.2 Glossary

This section briefly defines the acronyms and abbreviations used in this document:

Word: 32-bit data.

Half Word: 16-bit data.

Byte: 8-bit data.

IAP (In Application Programming): IAP means that the Flash memory of the microcontroller can be reprogrammed while the user program is running.

ICP (In-Circuit Programming): ICP means that the Flash memory of the microcontroller can be programmed using the JTAG protocol, SWD protocol, or bootloader while the device is mounted on the user's application board.

AHB: Advanced High-Performance Bus.

APB: Low-Speed Peripheral Bus.

DMA: Direct Memory Access.

TIM: Timer.

Version Revision History

Version	Date	Revision Content
Rev1.00	2019/11/20	First draft released.
Rev1.10	2019/12/31	Updated the following information: ① Added QFN32 package; ② Typical application circuit diagram; ③ Device electronic signature; ④ Appendix A SysTick timer; ⑤ Illustrations and precautions in high-speed external clock XTH and low-speed external clock XTL; ⑥ Description of general operating conditions; ⑦ Added precautions to system control register 1 (SYSCTRL1); ⑧ Use temperature sensor to measure ambient temperature; ⑨ Description of DMA controller (DMAC) and cyclic redundancy check (CRC); ⑩ Bit 0 of I2C configuration register (I2Cx_CR).
Rev1.20	2020/04/10	Updated the following information: ① Pin function description; ② Added notes to 29.7.2.17 and 29.7.2.24; ③ Step8 in 7.5.1 and 7.5.2; ④ QFN32 in package size; ⑤ Added Step in 3.2.7 and 3.2.8; ⑥ Added AVCC/3 accuracy in ADC characteristics; ⑦ 44/45/47/48 pins of HC32L073KATA changed; ⑧ 32/33/35/36 pins of HC32L073JATA changed.
Rev1.30	2020/05/29	Updated the following information: ① Deleted the note in 11.3.2; ② Corrected the typo in External input low-speed clock; ③ Added LCD controller; ④ RCL oscillator accuracy in Internal RCL oscillator; ⑤ Added low-power timer description in Product Features Ultra-low Power MCU; ⑥ Added Step2 and Step3 in 7.5.
Rev1.40	2020/06/30	Updated the following information: ① Added I2S information to the pin function description; ② Corrected LPTIM to LPTIMO, LPTIMx_ETR to LPTIMx_EXT; ③ Unified the pin function names and some register names; ④ Updated the descriptions of 21.3.1.6 and 22.3.2.6.
Rev1.50	2020/07/31	Updated the following information: ① Input characteristics - VIH value and VIL value in ports PA, PB, PC, PD, PE, PF; ② Added TIM timer characteristics and communication interface section; ③ EFT characteristic level; ④ Adjusted 29.7.1 and 29.7.2 registers.
Rev1.60	2020/09/30	Updated the following information: ① Functional block diagram in Functional Module; ② Added SPI Characteristics and I2S Characteristics; ③ Updated typos in 11.8.4, 11.8.14 and 35.4.2; ④ Clock system description; ⑤ VIL and VIH of RESETB pin characteristics; ⑥ Added input characteristics - USB_DP, USB_DM; ⑦ Bit3 of 28.7.3.
Rev1.61	2020/12/31	Delete product features, pin configuration, packaging information, etc. (please refer to the latest data sheet for related information), and modify the statement.
Rev1.62	2021/09/17	Replace with a high-resolution vector images.
Rev1.63	2022/03/09	Company Logo updated.
Rev1.64	2022/08/13	Updated the following information: ① In the "I2C Bus" section, the I2C operation process errata was added; ② In the "External High-speed Crystal Clock XTH" section, the description of the recommendation to use an active crystal was added; ③ In the "Port Digital Multiplexing Function" section, the PF01 function mapping of TIM4_CHB was deleted; ④ In the "Periodic Interval Response" section, the description of the initialization method and precautions of this function were added; ⑤ In the "Baud Rate Setting" section, the description of the sampling point was added; ⑥ In the "Controller Area Network" section, precautions for using CAN were added.

Version	Date	Revision Content
Rev1.65	2023/06/21	In the "Pulse Width Measurement PWC Register Description" section, added the unreleased function description of the control register (TIMx_CR0).
Rev1.70	2024/05/23	Updated the following information: ①3.5.2 Modified the precautions for the description of EXTL_EN and EXTH_EN functions of the SYSCTRL1 register; ②7.5 Updated the steps of programming examples; ③Added PLL to the monitored clock source in 9.3.2.2 Clock monitoring hardware structure; ④Added precautions in 16.2.3.2 PCA 16-bit PWM function; ⑤Changed PC5 to PC15 in 17.3.30 TIMx_PTBKS and 17.3.33 TIMx_PTBDK; ⑥Changed "HSEL" to "HDSEL" in 22.4.2 LPUARTx_SCON; ⑦Updated the description of BEIF function of CAN_ERRINT register in 30.5.9; ⑧Deleted the 1.2V related content in the analog-to-digital converter (ADC) section; ⑨Deleted the 1.2V and reference voltage output of the on-chip BGR in the analog voltage comparator (VC) section.
Rev1.71	2025/01/14	3.5.2 Modified the notes of the EXTL_EN and EXTH_EN function description of the SYSCTRL1 register.
Rev1.72	2025/03/28	Deleted the description of the maximum communication rate of the master and slave in 8.2 SPI main features.