DP32G030 参考手册

32 位微控制器

Version: 1.23

Action Dynamic Tech.(HK) Trading Co.
深圳市动能世纪科技有限公司
公司地址：深圳市南山区打石一路国际创新谷6期B座1111室
电话：0755-83134419
传真：0755-82519160
公司网址：www.dnsj88.com
EMAIL:dnsj@dn-ic.com
邮编：518031

目录

目录.....1
图目录.....13
表目录.....18
修订记录.....19
1. 简介.....20
1.1 概述.....20
1.2 产品特性.....21
2. 选型指南.....26
3. 芯片结构框图.....27
4. 引脚定义.....28
4.1 封装形式.....28
4.2 复用引脚.....30
4.3 引脚复用功能.....36
5. 功能描述.....39
5.1 地址空间映射.....39
5.2 存储器划分及权限控制.....41
5.3 中断向量表.....41
5.4 ARM Cortex-M0 内核.....43
5.4.1 概述.....43
5.4.2 特性.....43
5.4.3 系统定时器（SYSTICK）.....45
SYST_CTRL 状态寄存器（0x00）.....47
SYST_LOAD 重载寄存器（0x04）.....47
SYST_VAL 当前值寄存器（0x08）.....48
5.5 电源管理（PMU）.....49
5.5.1 概述.....49
5.5.2 特性.....49
5.5.3 模块结构框图.....49
5.5.4 功能描述.....51
电源.....51
复位.....52
低功耗模式.....55
寄存器映射.....62
寄存器描述.....63
LPOW_MD 寄存器（0x00）.....63
LPMD_WKEN 寄存器（0x04）.....64
LPMD_WKST 寄存器（0x08）.....64
CHIP_RST_ST 寄存器（0x0C）.....65
SRC_CFG 寄存器（0x10）.....65
TRIM_POW0 寄存器（0x20）.....66
TRIM_POW1 寄存器（0x24）.....66

TRIM_POW2 寄存器（0x28）.....67
TRIM_POW3 寄存器（0x2C）.....67
TRIM_RCHF 寄存器（0x30）.....67
TRIM_RCLF 寄存器（0x34）.....68
TRIM_OPA 寄存器（0x38）.....68
TRIM_PLL 寄存器（0x3C）.....69
TRIM_LOCK 寄存器（0x80）.....69
DATA_BAK0 寄存器（0x100）.....69
DATA_BAK1 寄存器（0x104）.....69
DATA_BAK2 寄存器（0x108）.....70
DATA_BAK3 寄存器（0x10C）.....70
5.6 系统控制（SYSCON）.....71
5.6.1 概述.....71
5.6.2 特性.....71
5.6.3 模块结构框图.....71
5.6.4 功能描述.....73
RCHF 时钟.....74
PLL 时钟.....75
XTAL 时钟.....76
RCLF 时钟.....77
系统时钟（sys_clk）选择.....78
RTC 时钟.....78
独立看门狗（IWDT）时钟.....79
窗口看门狗（WWDT）时钟.....79
SARADC 采样时钟.....79
低功耗模式唤醒时钟.....79
128 位芯片唯一识别码.....79
寄存器映射.....80
寄存器描述.....81
CLK_SEL 寄存器（0x00）.....81
DIV_CLK_GATE 寄存器（0x04）.....82
DEV_CLK_GATE 寄存器（0x08）.....82
RC_FREQ_DELTA 寄存器（0x78）.....84
VREF_VOLT_DELTA 寄存器（0x7C）.....84
CHIP_ID0 寄存器（0x80）.....85
CHIP_ID1 寄存器（0x84）.....85
CHIP_ID2 寄存器（0x88）.....85
CHIP_ID3 寄存器（0x8C）.....85
PLL_CTRL 寄存器（0x180）.....85
PLL_ST 寄存器（0x184）.....86
5.7 IO 功能配置（PORTCON）.....87
5.7.1 概述.....87
5.7.2 特性.....87
5.7.3 模块结构框图.....88

5.7.4 功能描述.....89
寄存器映射寄存器映射.....94
寄存器描述.....95
PORTA_SEL0 寄存器（0x00）..... 95
PORTA_SEL1 寄存器（0x04）..... 97
PORTB_SEL0 寄存器（0x08）..... 99
PORTB_SEL1 寄存器（0x0C）..... 100
PORTC_SEL0 寄存器（0x10）..... 102
PORTA_IE 寄存器（0x100）..... 104
PORTB_IE 寄存器（0x104）..... 104
PORTC_IE 寄存器（0x108）..... 104
PORTA_PU 寄存器（0x200）..... 105
PORTB_PU 寄存器（0x204）..... 105
PORTC_PU 寄存器（0x208）..... 105
PORTA_PD 寄存器（0x300）..... 105
PORTB_PD 寄存器（0x304）..... 106
PORTC_PD 寄存器（0x308）..... 106
PORTA_OD 寄存器（0x400）..... 106
PORTB_OD 寄存器（0x404）..... 107
PORTC_OD 寄存器（0x408）..... 107
PORTA_WKE 寄存器（0x500）..... 107
PORTB_WKE 寄存器（0x504）..... 107
PORTC_WKE 寄存器（0x508）..... 108
PORT_CFG 寄存器（0x600）..... 108
PORTA_WK_SEL 寄存器（0x700）..... 109
PORTB_WK_SEL 寄存器（0x704）..... 109
PORTC_WK_SEL 寄存器（0x708）..... 109
5.8 通用 IO(GPIO).....110
5.8.1 概述.....110
5.8.2 特性.....110
5.8.3 模块结构框图.....111
5.8.4 功能描述.....111
寄存器映射 GPIODATA 寄存器（0x00）..... 119
GPIODIR 寄存器（0x04）..... 119
INTLVLTRG 寄存器（0x08）..... 119
INTBE 寄存器（0x0C）..... 119
INTRISEEN 寄存器（0x10）..... 120
INTEN 寄存器（0x14）..... 120
INTRAWSTAUS 寄存器（0x18）..... 120
INTSTAUS 寄存器（0x1C）..... 121
INTCLR 寄存器（0x20）..... 121
5.9 基本定时器（TIMERBASE）..... 122
5.9.1 概述..... 122
5.9.2 特性..... 122

5.9.3 模块结构框图.....123
5.9.4 功能描述.....124
寄存器映射.....129
寄存器描述.....130
TIMERBASE_EN 寄存器（0x00）.....130
TIMERBASE_DIV 寄存器（0x04）.....130
TIMERBASE_IE 寄存器（0x10）.....130
TIMERBASE_IF 寄存器（0x14）.....131
HIGH_LOAD 寄存器（0x20）.....131
HIGH_CNT 寄存器（0x24）.....131
LOW_LOAD 寄存器（0x30）.....131
LOW_CNT 寄存器（0x34）.....132
5.10 高级定时器（TIMERPLUS）.....133
5.10.1 概述.....133
5.10.2 特性.....134
5.10.3 模块结构框图.....135
5.10.4 功能描述.....137
计数时钟源选择与预分频.....137
定时模式.....138
计数模式.....140
输入捕获模式.....142
HALL 模式.....144
中断.....146
操作流程.....147
寄存器映射.....158
寄存器描述.....159
TIMERPLUS_EN 寄存器（0x00）.....159
TIMERPLUS_DIV 寄存器（0x04）.....159
TIMERPLUS_CTR 寄存器（0x08）.....159
TIMERPLUS_IE 寄存器（0x10）.....161
TIMERPLUS_IF 寄存器（0x14）.....162
TIMERPLUS_HIGH_LOAD 寄存器（0x20）.....163
TIMERPLUS_HIGH _CNT 寄存器（0x24）.....163
TIMERPLUS_HIGH _CVAL 寄存器（0x28）.....164
TIMERPLUS_LOW_LOAD 寄存器（0x30）.....164
TIMERPLUS_ LOW _CNT 寄存器（0x34）.....164
TIMERPLUS_ LOW _CVAL 寄存器（0x38）.....164
HALL_VAL 寄存器（0x40）.....165
5.11 独立看门狗时钟（IWDT）.....165
5.11.1 概述.....165
5.11.2 特性.....166
5.11.3 模块结构框图.....167
5.11.4 功能描述.....167
寄存器映射.....171

寄存器描述.....171
IWDT_LOAD 寄存器（0x00）.....171
IWDT_CTRL 寄存器（0x08）.....172
IWDT_IF 寄存器（0x0C）.....172
IWDT_FEED 寄存器（0x10）.....172
5.12 窗口看门狗时钟（WWDT）.....173
5.12.1 概述.....173
5.12.2 特性.....173
5.12.3 模块结构框图.....174
5.12.4 功能描述.....175
WWDT 中断产生及喂狗区间.....175
WWDT 喂狗及复位产生.....176
寄存器映射.....177
寄存器描述.....177
WWDT_LOAD 寄存器（0x00）.....177
WWDT_VALUE 寄存器（0x04）.....177
WWDT_CTRL 寄存器（0x08）.....178
WWDT_IF 寄存器（0x0C）.....178
WWDT_FEED 寄存器（0x10）.....179
5.13 基本脉冲宽度调制发生器（PWMBASE）.....180
5.13.1 概述.....180
5.13.2 特性.....180
5.13.3 模块结构框图.....181
5.13.4 功能描述.....183
寄存器映射.....189
寄存器描述.....189
PWMBASE_EN 寄存器（0x00）.....189
PWMBASE_DIV 寄存器（0x04）.....190
PWMBASE_CON 寄存器（0x08）.....190
PWMBASE_PERIOD 寄存器（0x0C）.....191
PWMBASE_INTEN 寄存器（0x10）.....191
PWMBASE_IF 寄存器（0x14）.....192
PWMBASE_CNT 寄存器（0x18）.....192
PWMBASE_CH0_COMP 寄存器（0x20）.....192
PWMBASE_CH1_COMP 寄存器（0x30）.....192
PWMBASE_CH2_COMP 寄存器（0x40）.....193
5.14 高级脉冲宽度调制发生器（PWMPLUS）.....194
5.14.1 概述.....194
5.14.2 特性.....194
5.14.3 模块结构框图.....195
5.14.4 功能描述.....198
寄存器描述.....221
PWMPLUS_CFG 寄存器（0x00）.....221
PWMPLUS_GEN 寄存器（0x04）.....222

PWMPLUS_CLKSRC 寄存器（0x08）.....224
PWMPLUS_BRAKE_CFG 寄存器（0x0c）.....225
PWMPLUS_MASK_LEV 寄存器（0x10）.....227
PWMPLUS_PERIOD 寄存器（0x1C）.....228
PWMPLUS_CH0_COMP 寄存器（0x20）.....228
PWMPLUS_CH1_COMP 寄存器（0x24）.....229
PWMPLUS_CH2_COMP 寄存器（0x28）.....229
PWMPLUS_CH0_DT 寄存器（0x30）.....229
PWMPLUS_CH1_DT 寄存器（0x34）.....230
PWMPLUS_CH2_DT 寄存器（0x38）.....230
PWMPLUS_TRIG_COMP 寄存器（0x40）.....230
PWMPLUS_TRIG_CFG 寄存器（0x44）.....231
PWMPLUS_IE 寄存器（0x60）.....231
PWMPLUS_IF 寄存器（0x64）.....232
PWMPLUS_SWLOAD 寄存器（0x84）.....234
PWMPLUS_MASK_EN 寄存器（0x88）.....235
PWMPLUS_CNT_ST 寄存器（0xE0）.....235
PWMPLUS_BRAKE_ST 寄存器（0xE4）.....236
5.15 实时时钟（RTC）.....237
5.15.1 概述.....237
5.15.2 特性.....237
5.15.3 模块结构图.....238
5.15.4 功能描述.....239
计数时钟源选择.....239
秒标功能.....239
日历功能.....240
闹钟功能.....241
中断功能.....241
低功耗唤醒功能.....242
操作流程.....242
寄存器映射.....244
寄存器描述.....244
RTC_CFG 寄存器（0x00）.....244
RTC_IE 寄存器（0x04）.....245
RTC_IF 寄存器（0x08）.....246
RTC_PRE 寄存器（0x10）.....247
RTC_TR 寄存器（0x14）.....247
RTC_DR 寄存器（0x18）.....248
RTC_AR 寄存器（0x1C）.....249
RTC_TSTR 寄存器（0x20）.....250
RTC_TSDR 寄存器（0x24）.....251
RTC_CNT 寄存器（0x28）.....251
RTC_VALID 寄存器（0x2C）.....251
5.16 UART 控制器（UART）.....253

5.16.1 概述.....253
5.16.2 特性.....254
5.16.3 模块结构框图.....255
功能描述.....256
接收时序及相关状态信号.....256
发送时序及相关状态信号.....257
波特率计算.....257
自动波特率检测.....258
数据采样.....259
硬件自动流控制.....259
接收超时说明.....261
中断.....262
寄存器映射.....263
寄存器描述.....264
UART_CTRL 寄存器（0x00）.....264
UART_BAUD 寄存器（0x04）.....265
UART_TDR 寄存器（0x08）.....266
UART_RDR 寄存器（0x0C）.....266
UART_IE 寄存器（0x10）.....266
UART_IF 寄存器（0x14）.....267
UART_FIFO 寄存器（0x18）.....269
UART_FC 寄存器（0x1C）.....270
UART_RXTO 寄存器（0x20）.....271
5.17 SPI 总线控制器（SPI）.....272
5.17.1 概述.....272
5.17.2 特性.....272
5.17.3 模块结构框图.....273
5.17.4 功能描述.....274
SPI 接口时序.....274
I/O 配置.....275
数据传输配置.....277
中断产生.....278
DMA 控制.....284
操作流程.....285
寄存器映射.....286
寄存器描述.....286
SPICR 寄存器（0x00）.....286
SPIWDR 寄存器（0x04）.....288
SPIRDR 寄存器（0x08）.....289
SPIIE 寄存器（0x10）.....289
SPIIF 寄存器（0x14）.....289
SPIFIFOST 寄存器（0x18）.....290
5.18 IIC 控制器（IIC）.....292
5.18.1 概述.....292

5.18.2 特性.....293
5.18.3 模块结构框图.....294
5.18.4 功能描述.....294
协议介绍.....294
SCL 和 SDA.....298
中断功能.....299
操作流程.....300
寄存器映射.....312
寄存器描述.....312
IIC_CCFG 寄存器（0x00）.....312
IIC_CST 寄存器（0x04）.....313
IIC_CTRANS 寄存器（0x08）.....314
IIC_RXDATA 寄存器（0x0C）.....315
IIC_TXDATA 寄存器（0x10）.....315
IIC_IE 寄存器（0x14）.....316
IIC_IF 寄存器（0x18）.....316
IIC_MCTRL 寄存器（0x20）.....318
IIC_MSPC 寄存器（0x24）.....319
IIC_SCTRL 寄存器（0x30）.....320
IIC_SADDR 寄存器（0x34）.....321
5.19 模数转换器（ADC）.....323
5.19.1 概述.....323
5.19.2 特性.....324
5.19.3 模块结构框图.....326
5.19.4 功能描述.....327
通道选择.....327
采样一次与采样多次取平均.....327
单次采样与连续采样.....328
通道存储与 FIFO.....333
内部采样时钟方式与外部采样时钟方式.....334
CPU 触发与外部信号触发.....334
中断.....335
操作流程.....336
寄存器映射.....337
寄存器描述.....339
ADC_CFG 寄存器（0x00）.....339
ADC_START 寄存器（0x04）.....341
ADC_IE 寄存器（0x08）.....342
ADC_IF 寄存器（0x0C）.....343
ADC_CHx_STAT 寄存器.....343
ADC_CHx_DATA 寄存器.....343
ADC_FIFO_STAT 寄存器（0xA0）.....344
ADC_FIFO_DATA 寄存器（0xA4）.....345
ADC_ EXTTRIG_SEL 寄存器（0xB0）.....345

ADC_CALIB_OFFSET 寄存器（0xF0）.....345
ADC_CALIB_KD 寄存器（0xF4）.....346
5.20 比较器（COMP）.....347
5.20.1 概述.....347
5.20.2 特性.....347
5.20.3 模块结构框图.....348
5.20.4 功能描述.....349
比较器输入引脚及内部输出信号.....349
迟滞.....349
滤波.....350
中断.....351
寄存器映射.....351
寄存器描述.....352
CMP_CFG 寄存器（0x120）.....352
CMP_ST 寄存器（0x124）.....353
5.21 运算放大器（OPAMP）.....354
5.21.1 概述.....354
5.21.2 特性.....354
5.21.3 模块结构图.....355
寄存器映射.....355
寄存器描述.....356
OPA_CFG.....356
5.22 FLASH 控制器（FLASHCTRL）.....357
5.22.1 概述.....357
5.22.2 特性.....357
5.22.3 模块结构框图.....358
5.22.4 功能描述.....359
低功耗模式.....359
读速率配置.....359
NVR 区和 MAIN 区选择.....360
操作模式.....360
FLASH 操作地址.....360
扇区大小.....361
START 操作启动控制.....361
操作锁.....361
MASK 功能.....361
擦除时间和编程时间配置.....362
FLASH 初始化忙标志.....362
控制器忙标志.....362
编程数据缓存寄存器空状态标志.....362
FLASH 程序初始化流程.....363
FLASH 扇区擦操作流程.....363
FLASH 单字编程操作流程.....365
FLASH 多字连续编程操作流程.....367

寄存器映射.....369
寄存器描述.....369
FLASH_CFG 寄存器（0x00）.....369
FLASH_ADDR 寄存器（0x04）.....370
FLASH_WDATA 寄存器（0x08）.....371
FLASH_START 寄存器（0x10）.....371
FLASH_ST 寄存器（0x14）.....371
FLASH_LOCK 寄存器（0x18）.....372
FLASH_UNLOCK 寄存器（0x1C）.....372
FLASH_MASK 寄存器（0x20）.....372
FLASH_ERASETIME 寄存器（0x24）.....373
FLASH_PROGTIME 寄存器（0x28）.....373
5.23 循环冗余校验（CRC）.....375
5.23.1 概述.....375
5.23.2 特性.....375
5.23.3 模块结构图.....376
5.23.4 功能描述.....377
CRC 码生成规则.....377
CRC 多项式.....378
操作流程.....378
寄存器映射.....379
寄存器描述.....379
CRC_CR 寄存器（0x00）.....379
CRC_IV 寄存器（0x04）.....380
CRC_DATAIN 寄存器（0x08）.....381
CRC_DATAOUT 寄存器（0x0C）.....381
5.24 DMA 控制器（DMA）.....382
5.24.1 概述.....382
5.24.2 特性.....382
5.24.3 模块结构示意图.....383
5.24.4 功能描述.....384
DMA 处理流程.....384
DMA 仲裁器处理.....385
DMA 通道配置信息.....386
源地址和目的地址数据格式说明.....388
中断说明.....390
DMA 请求映射关系.....390
DMA 工作流程.....393
寄存器映射.....395
寄存器描述.....396
DMA_CTR 寄存器（0x00）.....396
DMA_INTEN 寄存器（0x04）.....396
DMA_INTST 寄存器（0x08）.....397
DMA_CHnCTR 寄存器（0x100 + 0x20*(n)）.....398

DMA_CHnMOD 寄存器（0x100 + 0x20*(n)+ 0x04）.....399
DMA_CHnMSADDR 寄存器（0x100 + 0x20*(n)+ 0x08）.....401
DMA_CHnMDADDR 寄存器（0x100 + 0x20*(n) + 0x0C）.....401
DMA_CHn_ST 寄存器（0x100 + 0x20*(n) + 0x10）.....401
5.25 AES 运算单元（AES）.....402
5.25.1 概述.....402
5.25.2 特性.....403
5.25.3 模块结构框图.....403
5.25.4 功能描述.....404
电子密码本（ECB）.....404
密码块链接（CBC）.....406
计数器模式（CTR）.....407
数据模式.....409
操作流程.....411
寄存器映射.....414
寄存器描述.....414
AES_CR 寄存器（0x00）.....414
AES_SR 寄存器（0x04）.....416
AES_DINR 寄存器（0x08）.....416
AES_DOUTR 寄存器（0x0C）.....417
AES_KEYR0 寄存器（0x10）.....417
AES_KEYR1 寄存器（0x14）.....418
AES_KEYR2 寄存器（0x18）.....418
AES_KEYR3 寄存器（0x1C）.....418
AES_IVR0 寄存器（0x20）.....419
AES_IVR1 寄存器（0x24）.....419
AES_IVR2 寄存器（0x28）.....419
AES_IVR3 寄存器（0x2C）.....420
6. 电气特性.....421
6.1 测试条件.....421
6.2 绝对最大额定值.....421
6.2.1 电压特性.....421
6.2.2 电流特性.....421
6.2.3 温度特性.....422
6.3 工作条件.....422
6.3.1 通用工作条件.....422
6.3.2 上电和掉电的工作条件.....422
6.3.3 复位和电源控制模块特性.....422
6.3.4 供电电流特性.....423
6.3.5 内嵌参考电压.....424
6.3.6 低功耗模式唤醒时间.....424
6.3.7 内部高频时钟源特性.....425
6.3.8 内部低频时钟源特性.....425
6.3.9 外部高频时钟源特性.....425

6.3.10 外部低频时钟源特性.....426
6.3.11PLL 特性.....426
6.3.12FLASH 存储器特性.....426
6.3.13EMC 特性.....427
6.3.14IO 端口特性.....427
6.3.15EXTRST 端口特性.....428
6.3.16ADC 特性.....429
6.3.17ADC 精度.....429
6.3.18 温度传感器特性.....430
6.3.19 运算放大器特性.....430
6.3.20 比较器特性.....431

图目录

图 5-1 Cortex-M0 处理器功能框图.....43
图 5-2 Systick 模块结构图.....45
图 5-3 systick 计数时序图.....46
图 5-4 电源模块结构框图.....50
图 5-5 上电复位和掉电复位的波形示意图.....53
图 5-6 外部管脚复位波形示意图.....54
图 5-7 SLEEP 模式进入和退出示意图.....59
图 5-8 DEEPSLEEP 模式进入和退出示意图.....61
图 5-9 STOP 模式进入和退出示意图.....62
图 5-10 芯片时钟结构图.....72
图 5-11 外部高频晶振相关硬件配置图.....73
图 5-12 外部时钟相关硬件配置图.....74
图 5-13 外部低频晶振相关硬件配置图.....77
图 5-14 PORTCON 模块结构框图.....88
图 5-15 IO 输入功能选择示意图.....89
图 5-16 IO 输出功能选择示意图.....90
图 5-17 模拟信号连接 IO 示意图.....91
图 5-18 IO 复用为 ana_signal_b 模拟信号的功能示意图.....91
图 5-19 IO 推挽输出模式示意图.....92
图 5-20 IO 开漏输出模式示意图.....93
图 5-21 GPIO 模块系统框图.....110
图 5-22 GPIO 模块结构框图.....111
图 5-23 GPIO 的输出时序图.....112
图 5-24 GPIO 高电平中断时序图.....113
图 5-25 GPIO 低电平触发中断时序图.....114
图 5-26 GPIO 上升沿触发中断时序图.....115
图 5-27 GPIO 下降沿触发中断时序图.....116
图 5-28 GPIO 双边沿触发中断时序图.....117
图 5-29 GPIO 操作流程图.....118
图 5-30 TIMERBASE 模块系统框图.....122
图 5-31 TIMERBASE 模块结构框图.....123
图 5-32 TIMERBASE 高计数器计数时序图.....124
图 5-33 TIMERBASE 低计数器计数时序图.....125
图 5-34 TIMERBASE 计数器分频计数时序图.....126
图 5-35 TIMERBASE 中断标志及中断示意图.....127
图 5-36 TIMERBASE 操作流程.....128
图 5-37 TIMERPLUS 系统框图.....133
图 5-38 TIMERPLUS 结构框图.....136
图 5-39 TIMERPLUS 时钟源选择示意图.....138
图 5-40 TIMERPLUS pclk 分频时钟下计数器时序图.....138
图 5-41 TIMERPLUS 定时模式下计数器时序图.....139

图 5-42 TIMERPLUS 计数模式下上升沿有效时序图.....140
图 5-43 TIMERPLUS 计数模式下下降沿有效时序图.....141
图 5-44 TIMERPLUS 计数模式下双沿有效时序图.....142
图 5-45 TIMERPLUS 输入捕获下上升沿有效时序图.....142
图 5-46 TIMERPLUS 输入捕获下下降沿有效时序图.....143
图 5-47 TIMERPLUS 输入捕获下双沿有效时序图.....144
图 5-48 TIMERPLUS 在 HALL 模式下时序图..... 145
图 5-49 TIMERPLUS 中断标志及中断示意图.....146
图 5-50 TIMERPLUS 定时模式操作流程图.....148
图 5-51 TIMERPLUS 计数模式操作流程图.....151
图 5-52 TIMERPLUS 输入捕获模式操作流程图.....154
图 5-53 TIMERPLUS HALL 模式操作流程图.....157
图 5-54 IWDT 系统框图.....166
图 5-55 IWDT 模块结构框图.....167
图 5-56 IWDT 中断产生图.....168
图 5-57 IWDT 第一次中断前喂狗及复位..... 168
图 5-58 IWDT 第二次中断前喂狗及复位..... 169
图 5-59 IWDT 中断产生操作流程图..... 170
图 5-60 WWDT 模块系统框图.....173
图 5-61 WWDT 模块结构框图.....174
图 5-62 WWDT 中断产生及喂狗区间.....175
图 5-63 WWDT 喂狗及复位.....176
图 5-64 PWMBASE 模块系统框图..... 180
图 5-65 PWMBASE 模块结构框图..... 183
图 5-66 PWMBASE 不同周期和翻转点时序图.....184
图 5-67 PWMBASE 1 分频时序图..... 184
图 5-68 PWMBASE 2 分频时序图..... 185
图 5-69 PWMBASE 6 分频时序图..... 185
图 5-70 PWMBASE 输出使能时序图.....185
图 5-71 PWMBASE 输出翻转时序图.....186
图 5-72 PWMBASE 输出中断时序图.....187
图 5-73 PWMBASE 操作流程图..... 188
图 5-74 PWMPLUS 系统框图..... 194
图 5-75 PWMPLUS 结构框图..... 197
图 5-76 PWMPLUS 边沿对齐模式下，向上计数，单次输出，起始电平为 0 的时序图 199
图 5-77 PWMPLUS 边沿对齐模式下，向上计数，单次输出，起始电平为 1 的时序图 200
图 5-78 PWMPLUS 边沿对齐模式下，向上计数，循环输出，起始电平为 0 的时序图 201
图 5-79 PWMPLUS 边沿对齐模式下，向上计数，循环输出，起始电平为 1 的时序图 201
图 5-80 PWMPLUS 边沿对齐模式下，向下计数，单次输出，起始电平为 0 的时序图 202
图 5-81 PWMPLUS 边沿对齐模式下，向下计数，单次输出，起始电平为 1 的时序图 202
图 5-82 PWMPLUS 边沿对齐模式下，向下计数，循环输出，起始电平为 0 的时序图 203
图 5-83 PWMPLUS 边沿对齐模式下，向下计数，循环输出，起始电平为 1 的时序图 203
图 5-84 PWMPLUS 中心对齐模式下，向上计数，单次输出，起始电平为 0 的时序图 204
图 5-85 PWMPLUS 中心对齐模式下，向上计数，单次输出，起始电平为 1 的时序图 205

图 5-86 PWMPLUS 中心对齐模式下，向上计数，循环输出，起始电平为 0 的时序图 205
图 5-87 PWMPLUS 中心对齐模式下，向上计数，循环输出，起始电平为 0 的时序图 206
图 5-88 PWMPLUS 中心对齐模式下，向下计数，单次输出，起始电平为 0 的时序图 207
图 5-89 PWMPLUS 中心对齐模式下，向下计数，单次输出，起始电平为 1 的时序图 207
图 5-90 PWMPLUS 中心对齐模式下，向下计数，循环输出，起始电平为 0 的时序图 208
图 5-91 PWMPLUS 中心对齐模式下，向下计数，循环输出，起始电平为 1 的时序图 208
图 5-92 PWMPLUS 带死区插入的互补输出时序图.....209
图 5-93 PWMPLUS 死区长度大于正脉冲小于负脉冲时序图.....209
图 5-94 PWMPLUS 死区长度大于负脉冲小于正脉冲时序图.....210
图 5-95 PWMPLUS 边沿对齐模式刹车情况时序图.....210
图 5-96 PWMPLUS 中心对齐模式刹车情况时序图.....211
图 5-97 PWMPLUS 边沿对齐模式 MASK 情况时序图.....211
图 5-98 PWMPLUS 中心对齐模式 MASK 情况时序图.....212
图 5-99 PWMPLUS 边沿对齐模式，向上计数，起始电平为 0 情况下翻转点和周期时序图.....213
图 5-100 PWMPLUS 边沿对齐模式，向上计数，起始电平为 1 情况下翻转点和周期时序图.....213
图 5-101 PWMPLUS 边沿对齐模式，向下计数，起始电平为 0 情况下翻转点和周期时序图.....214
图 5-102 PWMPLUS 边沿对齐模式，向下计数，起始电平为 1 情况下翻转点和周期时序图.....214
图 5-103 PWMPLUS 中心对齐模式，向上计数，起始电平为 0 情况下翻转点和周期时序图.....215
图 5-104 PWMPLUS 中心对齐模式，向上计数，起始电平为 1 情况下翻转点和周期时序图.....215
图 5-105 PWMPLUS 中心对齐模式，向下计数，起始电平为 0 情况下翻转点和周期时序图.....216
图 5-106 PWMPLUS 中心对齐模式，向下计数，起始电平为 1 情况下翻转点和周期时序图.....216
图 5-107 PWMPLUS 输出优先级关系图.....217
图 5-108 PWMPLUS 刹车中断时序图.....217
图 5-109 PWMPLUS 翻转点，特定触发点，周期溢出中断以及触发信号时序图.....218
图 5-110 PWMPLUS 模块操作流程图.....220
图 5-111 RTC 模块系统框图.....237
图 5-112 RTC 模块结构框图.....238
图 5-113 RTCCLK 选择示意图.....239
图 5-114 RTC 中断标志及中断示意图.....241
图 5-115 UART 模块系统框图.....253
图 5-116 UART 模块结构框图.....255
图 5-117 UART 接收时序及相关状态信号图.....256
图 5-118 UART 发送时序及相关状态信号图.....257
图 5-119 UART 自动波特率计算示意图.....258
图 5-120 UART 数据采样示意图.....259
图 5-121 UART 自动流控连接图.....260

图 5-122 UART 流控接收示意图..... 260
图 5-123 UART 流控发送示意图..... 261
图 5-124 UART 第一次接收超时中断..... 261
图 5-125 UART 第二次接收超时中断..... 262
图 5-126 UART 中断标志及中断示意图..... 263
图 5-127 SPI 模块系统框图.....272
图 5-128 SPI 模块结构框图.....273
图 5-129 SPI 数据/时钟时序图（CPHA=0）.....274
图 5-130 SPI 数据/时钟时序图（CPHA=1）.....275
图 5-131 SPI SSN 时序图（CPHA=0）..... 275
图 5-132 SPI Master/SPI Slave 互连..... 277
图 5-133 SPI 中断标志及中断示意图..... 278
图 5-134 SPI 接收 FIFO 满中断..... 279
图 5-135 SPI 接收 FIFO 半满中断..... 281
图 5-136 SPI 发送 FIFO 半满中断..... 282
图 5-137 SPI 发送 FIFO 空中断..... 283
图 5-138 SPI 操作流程.....285
图 5-139 IIC 模块结构框图..... 294
图 5-140 IIC 数据传输图..... 295
图 5-141 IIC 主机发送用 7bit 地址寻址从机接收.....296
图 5-142 IIC 主机在第一个字节后立即读取从机.....297
图 5-143 IIC 组合格式..... 297
图 5-144 IIC 主机发送用 10bit 地址寻址从机接收.....297
图 5-145 IIC 主机接收用 10bit 地址寻址从机发送.....298
图 5-146 IIC SCL 开漏图.....298
图 5-147 IIC 中断标志及中断示意图..... 299
图 5-148 IIC 主机发送操作流程图..... 301
图 5-149 IIC 主机接收操作流程图..... 303
图 5-150 IIC 从机发送操作流程图..... 306
图 5-151 IIC 从机接收操作流程图..... 309
图 5-152 SARADC 系统框图..... 324
图 5-153 SARADC 模块结构图..... 326
图 5-154 SARADC 通道选择示意图..... 327
图 5-155 SARADC 单通道单次采样示意图..... 328
图 5-156 SARADC 单通道连续采样示意图..... 329
图 5-157 SARADC 多通道单次采样示意图..... 330
图 5-158 SARADC 多通道连续采样示意图..... 332
图 5-159 SARADC 外部触发选择..... 335
图 5-160 SARADC 中断标志和中断信号示意图.....335
图 5-161 SARADC 操作流程图..... 336
图 5-162 比较器结构框图..... 348
图 5-163 比较器迟滞示意图..... 350
图 5-164 比较器消除毛刺的滤波功能示意图.....350
图 5-165 比较器中断标志及中断示意图..... 351

图 5-166 运算放大器结构框图..... 355
图 5-167 FLASH 控制器结构图..... 358
图 5-168 扇区擦操作流程图..... 364
图 5-169 单字编程操作流程图..... 366
图 5-170 多字连续编程操作流程图..... 369
图 5-171 CRC 模块系统框图..... 375
图 5-172 CRC 模块结构图..... 376
图 5-173 CRC 码构成示意图..... 377
图 5-174 CRC 操作流程图..... 379
图 5-175 DMA 控制器系统连接示意图..... 382
图 5-176 DMA 模块结构示意图..... 384
图 5-177 DMA 源地址和目的地址的数据格式配置表..... 389
图 5-178 DMA 中断说明表..... 390
图 5-179 DMA 各通道对应外设请求映射关系..... 390
图 5-180 DMA 源侧外设请求映射关系图..... 392
图 5-181 DMA 目的侧外设请求映射关系图..... 393
图 5-182 DMA 工作流程图..... 394
图 5-183 AES 系统框图..... 402
图 5-184 AES 模块结构框图..... 404
图 5-185 AES 电子密码本算法的加密原理图..... 405
图 5-186 AES 电子密码本算法的解密原理图..... 405
图 5-187 AES 密码块链接算法的加密原理图..... 406
图 5-188 AES 密码块链接算法的解密的原理图..... 407
图 5-189 AES 计数器模式链接算法的加密原理图..... 408
图 5-190 AES 计数器模式链接算法的解密原理图..... 409
图 5-191 AES 位移原理图..... 411

表目录

表 2-1 32G030 系列 MCU 选型表.....26
表 4-1 引脚定义.....30
表 4-2 引脚复用功能.....36
表 5-1 地址空间映射.....39
表 5-6 32G030 中断向量表.....41
表 5-8 GPIO 高电平触发中断时序时间参数.....113
表 5-9 GPIO 低电平触发中断时序时间参数.....114
表 5-10 GPIO 上升沿触发中断时序时间参数.....115
表 5-11 GPIO 下降沿触发中断时序时间参数.....116
表 1 电压特性.....421
表 2 电流特性.....421
表 3 温度特性.....422
表 4 共用工作条件表.....422
表 5 上电和掉电的工作条件.....422
表 6 复位和电源控制模块特性.....422
表 7 供电电流特性.....423

修订记录

版本	日期	作者	备注
1.0	2021/11/20	周立业	初版
1.1	2021/12/15	周立业	优化使用说明
1.2	2022/01/20	杨伟佳	进一步优化使用说明及结构图
1.21	2022/02/16	吴伟男	格式修正内容审核
1.22	2022/02/21	孙浩钧	修正格式，更新了 CMP 相关内容
1.23	2022/02/21	孙浩钧	更新正确的寄存器表

1. 简介

1.1概述

本产品内嵌 ARM Cortex M0 内核，最高工作频率可达 72MHz，支持 PLL，内置高速存储器，丰富的增强型 IO 端口和多种外设。本产品包括最多 40 路 IO、1 个（14 通道）12 位的 ADC、4 个 16 位高级定时器，具有输入捕获和周期脉冲输出等功能、6 个 16 位的基本定时器、1 个 20 位的独立看门狗、1 个 7 位的窗口看门狗、6 路独立基本 PWM 波形发生器、6 路独立高级 PWM 波形发生器，支持死区和互补功能、2 个 SPI、2 个 IIC、3 个 UART、1 个标准的 RTC、1 个 CRC、3 个比较器、1 个温度传感器、2 个运算放大器、1 个 128bit 的 AES。

本产品系列工作电压为 2.0V-3.6V，工作温度为-40℃-105℃。多种省电工作模式保证低功耗应用的要求。

本产品适用于以下应用场合：物联网、智能家居、工业控制、仪器仪表、可穿戴设备、小家电、电机控制、医疗及手持设备等产品。

1.2产品特性

- 内核与系统
  - 32 位 ARM Cortex M0 处理器内核
  - 最高工作频率为 72MHz
  - 24 位的系统嘀嗒定时器
  - 集成嵌套向量中断控制器（NVIC），提供最多 32 个中断
  - 通过 SWD 接口烧录程序

- 存储器
  - 内置 64K 字节 FLASH 存储器作为程序存储区
  - 内置 16K 字节 RAM 作为数据存储区

- 时钟、复位及电源管理
  - 2.0V-3.6V 供电电压
  - 上电/断电复位(POR/PDR)、看门狗复位、片外专用引脚复位（EXTRST）
  - 内置 4-32MHz 高频晶体振荡驱动器
  - 内置 32768Hz 低频晶体振荡驱动器
  - 内置经出厂调校的 48MHz 的高频 RC 振荡器
  - 内置经出厂调校的 32768Hz 的低频 RC 振荡器
  - 内置 PLL 电路，支持最高 72MHz 频率

- 低功耗

  支持 SLEEP 模式、DEEPSLEEP 模式以及 STOP 模式，共三种系统低功耗模式

- SARADC
  - 14 通道 12bit SARADC
  - 采样率可以达到 2.4M
  - 用于电源电压、温度和外部信号采样
  - 支持单次模式和连续模式

- 支持硬件求平均功能，可以配置为 1、2、4、8 次求平均
- 支持 ADC 时钟可配置，可以选择系统时钟 1、2、4、8 分频
- 软件触发和硬件触发可选
- 支持多种中断

- GPIO
  - 最多可达 40 个 IO 口
  - 可配置为以下模式：浮空输入、上拉输入、下拉输入、推挽输出、开漏输出、模拟 IO
  - 灵活的中断配置，可配置为电平触发和边沿触发，电平触发可设置为低电平和高电平，边沿触发可设置为上升沿、下降沿和双边沿
  - 每个 GPIO 都支持按键唤醒功能，可配置为上升沿唤醒和下降沿唤醒
  - 大部分 IO 支持 5V 电平输入容忍
- CRC
  - 支持对 8、16、32 位数据进行 CRC 运算
  - 支持对输入数据和输出 CRC 值进行多种数据格式处理
  - 支持 CRC 多项式：

    x^8+x^2+x+1、x^16+x^12+x^5+1、x^16+x^15+x^2+1、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+1
- AES
  - 128bit AES，支持 ECB、CBC 和 CTR 模式
- DMA
  - 最大支持 4 通道
  - 源地址设备和目的地址设备可分别配置传输数据宽度 8、16 和 32bits，源地址和目的地址必须按照数据传输宽度对齐

- 每个通道最多可配置传输 4096 笔数据。各通道每传输完一笔数据，则该通道释放总线，电路将重新判断各通道优先级，优先级高的通道先传输
- 通道可配置为递增或固定地址模式
- 支持存储器到存储器、存储器到外设、外设到存储器
- 支持循环重启 DMA
- 支持每个通道优先级可配置，每个通道独立的中断

- 比较器
  - 3 路比较器
  - Rail to Rail
  - 可以配置不同的迟滞电压
  - 可以组合成窗口比较器
- 运算放大器
  - 2 路运算放大器
  - Rail to Rail
  - 3MHz bandwidth
  - 低偏移电压
  - 输出引脚可以选择配置到 ADC 通道直接采集
- 通信接口
  - 2 个 SPI 接口，可配置主从模式，可编程时钟极性和相位，主模式速率可配置，最高频率为系统时钟 4 分频，数据传输顺序可配置，读写数据寄存器独立，接收和发送分别采用 8 级 FIFO 缓存机制、具有 DMA 传输功能

- 3 个 UART 接口，支持全双工模式、支持 8/9 位数据格式选择、可配置奇偶校验位 奇校验 偶校验 常 0 常 1、支持 1 位停止位、发送延迟时间可配置、支持发送完成中断、接收完成中断、接收超时中断、支持自动波特率检测、具有硬件流控功能、采用 8 级 FIFO 缓存机制、具有 DMA 传输功能
- 2 个 IIC 接口，支持主从模式，支持 3 种模式：Standard-mode（100kbps）、Fast-mode（400kbps）、Fast-mode Plus（1Mbps）、主机模式下支持 clock synchronization、SCL 时钟占空比可配置、从机模式支持从机地址掩码、支持 7 位和 10 位两种地址模式

- 定时器
  - 6 个 16 位基本定时器，计数时钟支持 1-65536 分频，只支持定时功能
  - 4 个 16 位高级定时器，计数时钟支持 1-65536 分频，具有定时、计数、输入捕获和周期脉冲输出功能，支持 HALL 功能
  - 1 个 20 位的独立看门狗定时器 ，1 个 7 位的窗口看门狗定时器
  - 6 路独立的 16 位基本 PWM 波形发生器，计数时钟支持 1-65536 分频，支持翻转点中断和周期溢出中断
  - 6 路独立的 16 位高级 PWM 波形发生器，计数时钟支持 1-65536 分频，支持死区、互补和刹车功能，支持上升沿计数或下降沿计数，支持边沿对齐或中心对齐波形输出，支持空闲电平、计数起始电平和输出电平翻转可配置，支持翻转点中断、周期溢出中断和特定触发点中断
  - 1 个标准的 RTC，具有实时时钟、闹钟和日历功能，并且能自动解决闰年问题，可输出与 RTC 同步的半秒、秒、分、时、日、闹钟等中断，RTC 计数时钟可选择两种时钟源 RCLF 和 XTAL。
- UUID 128 位的芯片唯一识别码

- 环境
  - 工作温度：-40℃-105℃

- 保存温度：-50℃-150℃
- 湿度等级：MSL3

- 封装
  - LQFP48、LQFP32、TSSOP20 等

2. 选型指南

表 2-1 32G030 系列 MCU 选型表

Part Number	Voltage (V)	FLASH (KB)	RAM (KB)	IO	TIMERBASE	TIMERPLUS	PWMBASE	PWMPLUS	IWDT	ADC	UART	SPI	IIC	Package
DP32G030LQ48	2.0V-3.6V	64	16	40	3	2	2	2	1	1	3	2	2	LQFP48
DP32G030LQ32	2.0V-3.6V	64	16	26	3	2	2	2	1	1	3	2	2	LQFP32
DP32G030TS20	2.0V-3.6V	64	16	16	3	2	2	2	1	1	3	2	2	TSSOP20

3. 芯片结构框图

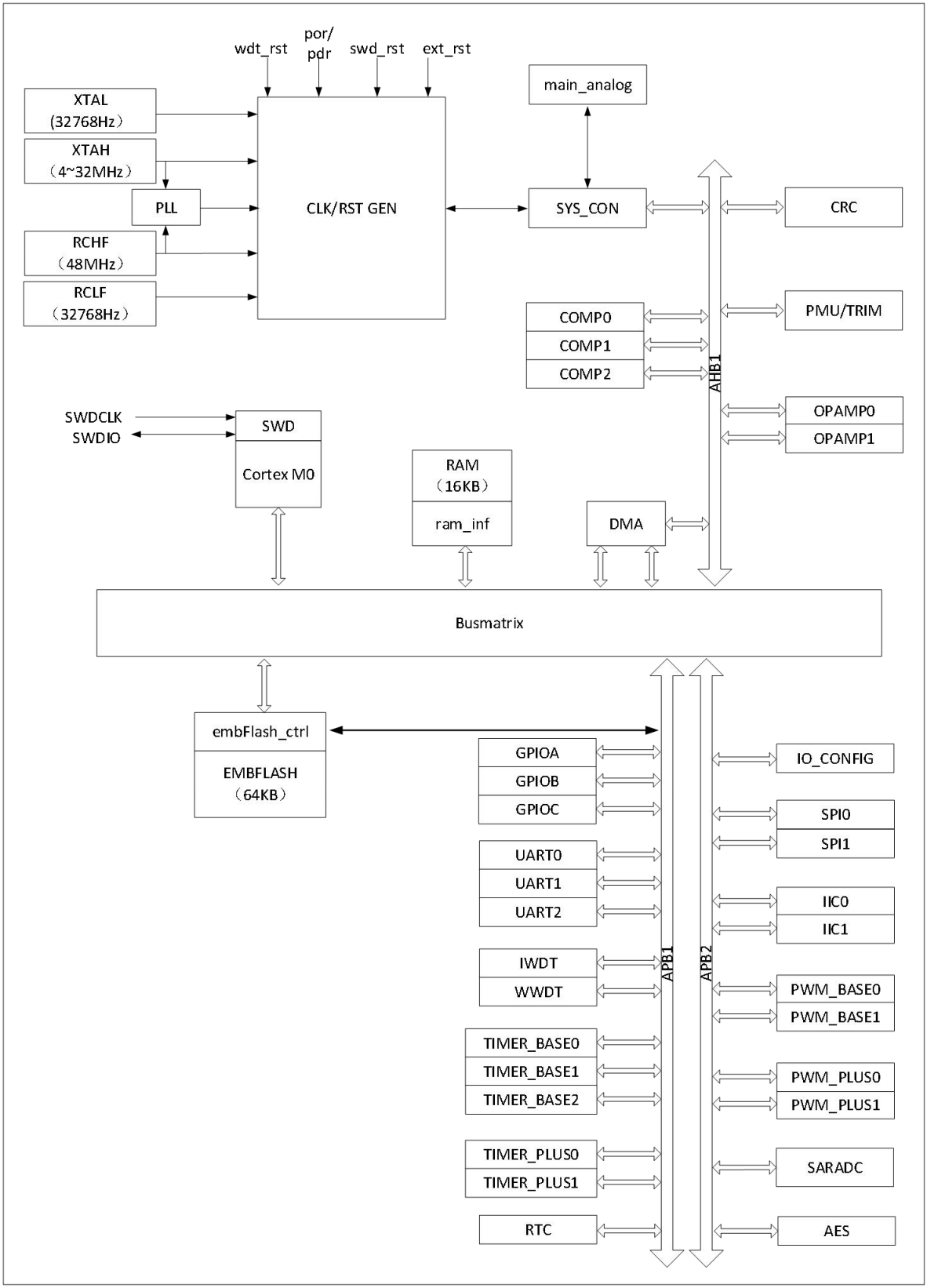
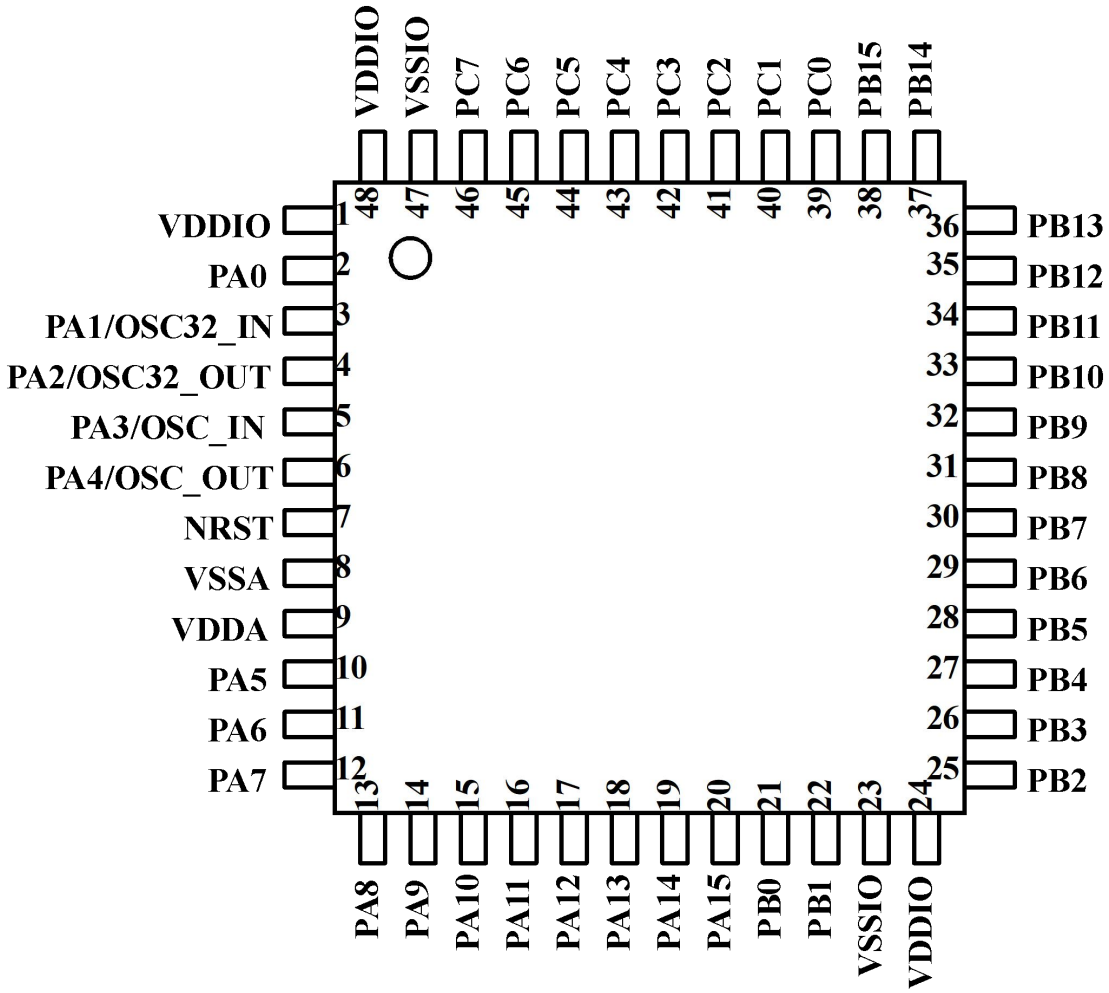


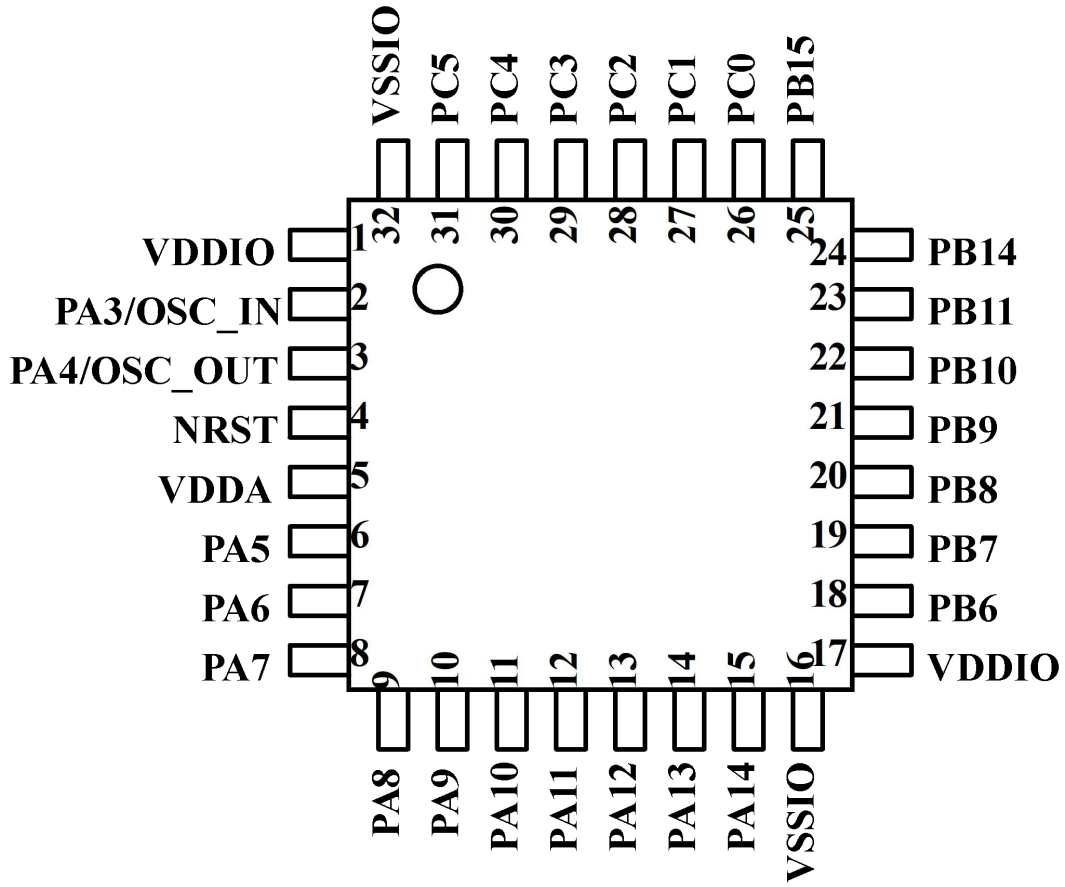
图 3-1 芯片结构图

4. 引脚定义

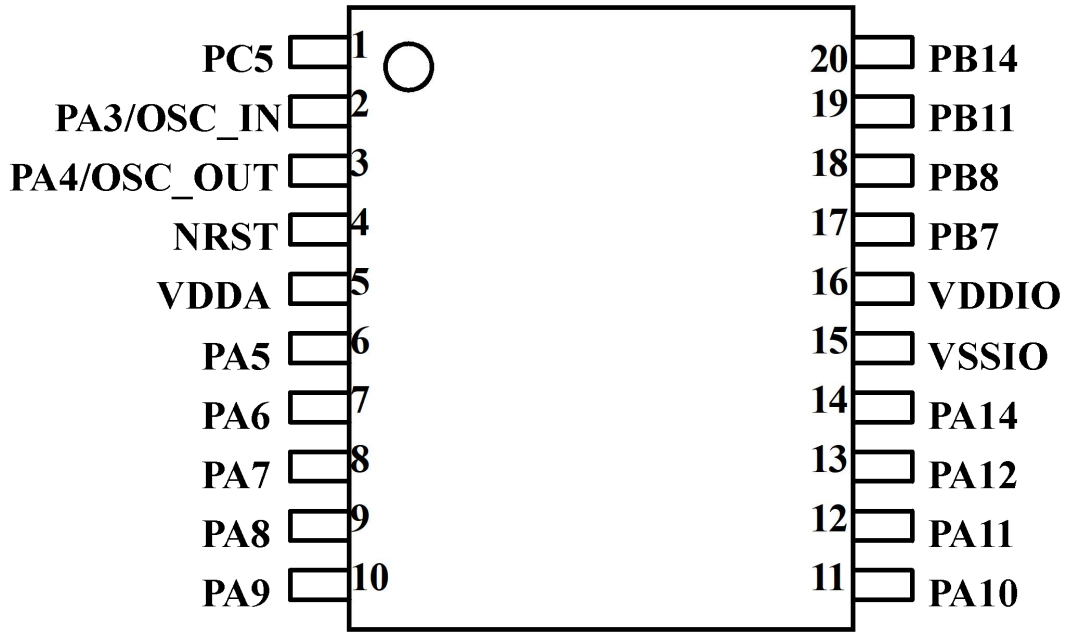
4.1 封装形式



LQFP48(top view)



LQFP32(top view)



TSSOP20(top view)

4.2复用引脚

表 4-1 引脚定义

引脚编号 LQ48	引脚编号 LQ32	引脚编号 TS20	引脚定义	类型	复用功能	描述
1 24 48	1 17	16	VDDIO	S		2.0V~3.6 V 电源输入脚
23 47	16 32	15	VSSIO	S		电源地
9	5	5	VDDA	S		2.3V~3.6 V 电源输入脚
8			VSSA	S		电源地
7	4	4	NRST	I/O		硬件复位引脚
2			PA0	I/O	GPIO_PA0	PA0：数字 GPIO 功能引脚
					PWMP1_PLUS0	PWMP1_PLUS0：高级 PWM1 的输入脉冲引脚 0
					PWMP0_PLUS1	PWMP0_PLUS1：高级 PWM0 的输入脉冲引脚 1
					TM	TM：RTC 1/2 秒标输出引脚
					WAKEUP0	WAKEUP0：唤醒引脚 0
3			PA1	I/O	GPIO_PA1	PA1：数字 GPIO 功能引脚
					XTAL_XI	XTAL_XI：低频晶振的输入引脚
4			PA2	I/O	GPIO_PA2	PA2：数字 GPIO 功能引脚
					XTAL_XO	XTAL_XO：低频晶振的输出引脚
5	2	2	PA3	I/O	GPIO_PA3	PA3：数字 GPIO 功能引脚
					CMP0_VN	CMP0_VN：比较器 0 的 N 输入引脚
					XTAH_XI	XTAH_XI：高频晶振的输入引脚
6	3	3	PA4	I/O	GPIO_PA4	PA4：数字 GPIO 功能引脚
					CMP0_VP	CMP0_VP：比较器 0 的 P 输入引脚
					XTAH_XO	XTAH_XO：高频晶振的输出引脚
10	6	6	PA5	I/O	GPIO_PA5	PA5：数字 GPIO 功能引脚
					UART1_CTS	UART1_CTS：串口 1 的 CTS 引脚
					PWMP1_PLUS1	PWMP1_PLUS1：高级 PWM1 的输入脉冲引脚 1
					TIMERP1_IN0	TIMERP1_IN0：高级定时器 1 的输入 0 引脚
					TIMERP1_OUT_L	TIMERP1_OUT_L：高级定时器 1 的 L 输出引脚
					WAKEUP1	WAKEUP1：唤醒引脚 1

					SARADC_CH0	SARADC_CH0：SARADC 的通道 0 引脚
11	7	7	PA6	I/O	GPIO_PA6	PA6：数字 GPIO 功能引脚
					UART1_RTS	UART1_RTS：串口 1 的 RTS 引脚
					TIMERP1_IN1	TIMERP1_IN1：高级定时器 1 的输入 1 引脚
					TIMERP1_OUT_H	TIMERP1_OUT_H：高级定时器 1 的 H 输出引脚
					SARADC_CH1	SARADC_CH1：SARADC 的通道 1 引脚
					OPA0_OUT	OPA0_OUT：运算放大器 0 的输出引脚
12	8	8	PA7	I/O	GPIO_PA7	PA7：数字 GPIO 功能引脚
					UART1_TX	UART1_TX：串口 1 的发送引脚
					TIMERP0_IN0	TIMERP0_IN0：高级定时器 0 的输入 0 引脚
					TIMERP0_OUT_L	TIMERP0_OUT_L：高级定时器 0 的 L 输出引脚
					SARADC_CH2	SARADC_CH2：SARADC 的通道 2 引脚
					OPA0_VP	OPA0_VP：运算放大器 0 的 P 输入引脚
13	9	9	PA8	I/O	GPIO_PA8	PA8：数字 GPIO 功能引脚
					UART1_RX	UART1_RX：串口 1 的接收引脚
					TIMERP0_IN1	TIMERP0_IN1：高级定时器 0 的输入 1 引脚
					TIMERP0_OUT_H	TIMERP0_OUT_H：高级定时器 0 的 H 输出引脚
					SARADC_CH3	SARADC_CH3：SARADC 的通道 3 引脚
					OPA0_VN	OPA0_VN：运算放大器 0 的 N 输入引脚
14	10	10	PA9	I/O	GPIO_PA9	PA9：数字 GPIO 功能引脚
					SPI0_SSN	SPI0_SSN:SPI0 的片选引脚
					TIMERP1_IN0	TIMERP1_IN0：高级定时器 1 的输入 0 引脚
					TIMERP1_OUT_L	TIMERP1_OUT_L：高级定时器 1 的 L 输出引脚
					TM	TM：RTC 1/2 秒标输出引脚
					SARADC_CH4	SARADC_CH4：SARADC 的通道 4 引脚
					CMP1_VN	CMP1_VN：比较器 1 的 N 输入引脚
15	11	11	PA10	I/O	GPIO_PA10	PA10：数字 GPIO 功能引脚
					SPI0_CLK	SPI0_CLK：SPI0 的时钟引脚
					SARADC_CH5	SARADC_CH5：SARADC 的通道 5 引脚
					CMP1_VP	CMP1_VP：比较器 1 的 P 输入引脚
16	12	12	PA11	I/O	GPIO_PA11	PA11：数字 GPIO 功能引脚
					SPI0_ MISO	SPI0_MISO：SPI0 的主机接收引脚
					PWMB0_CH0	PWMB0_CH0：基本 PWM0 的通道 0 引脚
					PWMP0_BRK0	PWMP0_ BRK0：高级 PWM0 的 BRAKE0 引脚
					TIMERP1_IN1	TIMERP1_IN1：高级定时器 1 的输入 1 引脚
					TIMERP1_OUT_H	TIMERP1_OUT_H：高级定时器 1 的 H 输出引脚
					SARADC_CH6	SARADC_CH6：SARADC 的通道 6 引脚
17	13	13	PA12	I/O	GPIO_PA12	PA12：数字 GPIO 功能引脚
					SPI0_MOSI	SPI0_MOSI：SPI0 的主机发送引脚
					PWMB0_CH1	PWMB0_CH1：基本 PWM0 的通道 1 引脚
					PWMP0_CH0N	PWMP0_CH0N：高级 PWM0 的通道 0N 引脚

					TIMERP0_IN0	TIMERP0_IN0：高级定时器 0 的输入 0 引脚
					TIMERP0_OUT_L	TIMERP0_OUT_L：高级定时器 0 的 L 输出引脚
					SARADC_CH7	SARADC_CH7：SARADC 的通道 7 引脚
18	14		PA13	I/O	GPIO_PA13	PA13：数字 GPIO 功能引脚
					PWMB0_CH2	PWMB0_CH2：基本 PWM0 的通道 2 引脚
					PWMP0_CH1N	PWMP0_CH1N：高级 PWM0 的通道 1N 引脚
					TIMERP0_IN1	TIMERP0_IN1：高级定时器 0 的输入 1 引脚
					TIMERP0_OUT_H	TIMERP0_OUT_H：高级定时器 0 的 H 输出引脚
					SARADC_CH8	SARADC_CH8：SARADC 的通道 8 引脚
19	15	14	PA14	I/O	GPIO_PA14	PA14：数字 GPIO 功能引脚
					PWMB1_CH0	PWMB1_CH0：基本 PWM1 的通道 0 引脚
					PWMP0_CH2N	PWMP0_CH2N：高级 PWM0 的通道 2N 引脚
					TIMERP1_IN0	TIMERP1_IN0：高级定时器 1 的输入 0 引脚
					TIMERP1_OUT_L	TIMERP1_OUT_L：高级定时器 1 的 L 输出引脚
					SARADC_CH9	SARADC_CH9：SARADC 的通道 9 引脚
20			PA15	I/O	GPIO_PA15	PA15：数字 GPIO 功能引脚
					PWMB1_CH1	PWMB1_CH1：基本 PWM1 的通道 1 引脚
					PWMP0_CH0	PWMP0_CH0：高级 PWM0 的通道 0 引脚
					TIMERP1_IN1	TIMERP1_IN1：高级定时器 1 的输入 1 引脚
					TIMERP1_OUT_H	TIMERP1_OUT_H：高级定时器 1 的 H 输出引脚
21			PB0	I/O	GPIO_PB0	PB0：数字 GPIO 功能引脚
					UART2_TX	UART2_TX：串口 2 的发送引脚
					IIC0_SCL	IIC0_SCL：IIC0 的时钟引脚
					PWMB1_CH2	PWMB1_CH2：基本 PWM1 的通道 2 引脚
					PWMP0_CH1	PWMP0_CH1：高级 PWM0 的通道 1 引脚
22			PB1	I/O	GPIO_PB1	PB1：数字 GPIO 功能引脚
					UART2_RX	UART2_RX：串口 2 的接收引脚
					IIC0_SDA	IIC0_SDA：IIC0 的数据引脚
					PWMP0_CH2	PWMP0_CH2：高级 PWM0 的通道 2 引脚
25			PB2	I/O	GPIO_PB2	PB2：数字 GPIO 功能引脚
					SPI1_SSN	SPI1_SSN:SPI1 的片选引脚
					PWMP0_BRK1	PWMP0_ BRK1：高级 PWM0 的 BRAKE1 引脚
					TIMERP1_HALL0	TIMERP1_HALL0：高级定时器 1 的 HALL0 引脚
26			PB3	I/O	GPIO_PB3	PB3：数字 GPIO 功能引脚
					SPI1_CLK	SPI1_CLK：SPI1 的时钟引脚
					IIC1_SDA	IIC1_SDA：IIC1 的数据引脚
					PWMP0_CH0N	PWMP0_CH0N：高级 PWM0 的通道 0N 引脚
					TIMERP1_HALL1	TIMERP1_HALL1：高级定时器 1 的 HALL1 引脚
27			PB4	I/O	GPIO_PB4	PB4：数字 GPIO 功能引脚
					SPI1_ MISO	SPI1_MISO：SPI1 的主机接收引脚
					IIC1_SCL	IIC1_SCL：IIC1 的时钟引脚
					PWMP1_CH0	PWMP1_CH0：高级 PWM1 的通道 0 引脚

					PWMP0_CH1N	PWMP0_CH1N：高级 PWM0 的通道 1N 引脚
					TIMERP1_HALL2	TIMERP1_HALL2：高级定时器 1 的 HALL2 引脚
28			PB5	I/O	GPIO_PB5	PB5：数字 GPIO 功能引脚
					SPI1_MOSI	SPI1_MOSI：SPI1 的主机发送引脚
					PWMP1_CH0N	PWMP1_CH0N：高级 PWM1 的通道 0N 引脚
					PWMP0_CH2N	PWMP0_CH2N：高级 PWM0 的通道 2N 引脚
					TIMERP0_IN0	TIMERP0_IN0：高级定时器 0 的输入 0 引脚
					TIMERP0_OUT_L	TIMERP0_OUT_L：高级定时器 0 的 L 输出引脚
29	18		PB6	I/O	GPIO_PB6	PB6：数字 GPIO 功能引脚
					PWMP0_CH0	PWMP0_CH0：高级 PWM0 的通道 0 引脚
					TIMERP0_IN1	TIMERP0_IN1：高级定时器 0 的输入 1 引脚
					TIMERP0_OUT_H	TIMERP0_OUT_H：高级定时器 0 的 H 输出引脚
30	19	17	PB7	I/O	GPIO_PB7	PB7：数字 GPIO 功能引脚
					SPI0_SSN	SPI0_SSN:SPI0 的片选引脚
					UART0_TX	UART0_TX：串口 0 的发送引脚
					IIC0_SCL	IIC0_SCL：IIC0 的时钟引脚
					PWMP1_BRK0	PWMP1_ BRK0：高级 PWM1 的 BRAKE0 引脚
					PWMP0_CH1	PWMP1_ BRK0：高级 PWM1 的 BRAKE0 引脚
31	20	18	PB8	I/O	GPIO_PB8	PB8：数字 GPIO 功能引脚
					SPI0_CLK	SPI0_CLK：SPI0 的时钟引脚
					UART0_RX	UART0_RX：串口 0 的接收引脚
					IIC0_SDA PWMB0_CH0	IIC0_SDA：IIC0 的数据引脚 PWMB0_CH0：基本 PWM0 的通道 0 引脚
					PWMP1_BRK1	PWMP1_ BRK1：高级 PWM1 的 BRAKE1 引脚
					PWMP0_CH2	PWMP0_CH2：高级 PWM0 的通道 2 引脚
32	21		PB9	I/O	GPIO_PB9	PB9：数字 GPIO 功能引脚
					SPI0_ MISO	SPI0_MISO：SPI0 的主机接收引脚
					UART0_CTS	UART0_CTS：串口 0 的 CTS 引脚
					PWMB0_CH1	PWMB0_CH1：基本 PWM0 的通道 1 引脚
					PWMP1_CH0	PWMP1_CH0：高级 PWM1 的通道 0 引脚
					TIMERP1_IN1	TIMERP1_IN1：高级定时器 1 的输入 1 引脚
					TIMERP1_OUT_H	TIMERP1_OUT_H：高级定时器 1 的 H 输出引脚
33	22		PB10	I/O	GPIO_PB10	PB10：数字 GPIO 功能引脚
					SPI0_MOSI	SPI0_MOSI：SPI0 的主机发送引脚
					UART0_RTS	UART0_RTS：串口 1 的 RTS 引脚
					PWMB0_CH2	PWMB0_CH2：基本 PWM0 的通道 2 引脚
					PWMP1_CH1	PWMP1_CH1：高级 PWM1 的通道 1 引脚
					PWMP0_PLUS0	PWMP0_PLUS0：高级 PWM0 的输入脉冲引脚 0
					TIMERP1_IN0	TIMERP1_IN0：高级定时器 1 的输入 0 引脚
					TIMERP1_OUT_L	TIMERP1_OUT_L：高级定时器 1 的 L 输出引脚
34	23	19	PB11	I/O	GPIO_PB11	PB11：数字 GPIO 功能引脚
					SWDIO	SWDIO：芯片 SW 口的数据引脚

					PWMP1_CH2	PWMP1_CH2：高级 PWM1 的通道 2 引脚
					PWMP0_BRK2	PWMP0_ BRK2：高级 PWM0 的 BRAKE2 引脚
35			PB12	I/O	GPIO_PB12	PB12：数字 GPIO 功能引脚
					UART1_TX	UART1_TX：串口 1 的发送引脚
					IIC1_SCL	IIC1_SCL：IIC1 的时钟引脚
					PWMP1_CH0N	PWMP1_CH0N：高级 PWM1 的通道 0N 引脚
36			PB13	I/O	GPIO_PB13	PB13：数字 GPIO 功能引脚
					UART1_RX	UART1_RX：串口 1 的接收引脚
					IIC1_SDA	IIC1_SDA：IIC1 的数据引脚
					PWMP1_CH1N	PWMP1_CH1N：高级 PWM1 的通道 1N 引脚
37	24	20	PB14	I/O	GPIO_PB14	PB14：数字 GPIO 功能引脚
					SWCLK	SWCLK：芯片 SW 口的时钟引脚
					UART2_TX	UART2_TX：串口 1 的发送引脚
					PWMP1_CH2N	PWMP1_CH2N：高级 PWM1 的通道 2N 引脚
38	25		PB15	I/O	GPIO_PB15	PB15：数字 GPIO 功能引脚
					SPI1_SSN	SPI1_SSN:SPI1 的片选引脚
					UART2_RX	UART2_RX：串口 2 的接收引脚
39	26		PC0	I/O	GPIO_PC0	PC0：数字 GPIO 功能引脚
					SPI1_CLK	SPI1_CLK：SPI1 的时钟引脚
					UART2_CTS	UART2_CTS：串口 2 的 CTS 引脚
					PWMB1_CH0	PWMB1_CH0：基本 PWM1 的通道 0 引脚
40	27		PC1	I/O	GPIO_PC1	PC1：数字 GPIO 功能引脚
					SPI1_ MISO	SPI1_MISO：SPI1 的主机接收引脚
					UART2_RTS	UART2_RTS：串口 2 的 RTS 引脚
					PWMB1_CH1	PWMB1_CH1：基本 PWM1 的通道 1 引脚
					TIMERP0_IN0	TIMERP0_IN0：高级定时器 0 的输入 0 引脚
					TIMERP0_OUT_L	TIMERP0_OUT_L：高级定时器 0 的 L 输出引脚
41	28		PC2	I/O	GPIO_PC2	PC2：数字 GPIO 功能引脚
					SPI1_MOSI	SPI1_MOSI：SPI1 的主机发送引脚
					PWMB1_CH2	PWMB1_CH2：基本 PWM1 的通道 2 引脚
					PWMP1_BRK2	PWMP1_ BRK2：高级 PWM1 的 BRAKE2 引脚
					TIMERP0_IN1	TIMERP0_IN1：高级定时器 0 的输入 1 引脚
					TIMERP0_OUT_H	TIMERP0_OUT_H：高级定时器 0 的 H 输出引脚
42	29		PC3	I/O	GPIO_PC3	PC3：数字 GPIO 功能引脚
					UART0_TX	UART0_TX：串口 0 的发送引脚
					IIC0_SCL	IIC0_SCL：IIC0 的时钟引脚
					PWMP1_CH1N	PWMP1_CH1N：高级 PWM1 的通道 1N 引脚
					TIMERP0_HALL0	TIMERP0_HALL0：高级定时器 0 的 HALL0 引脚
					CMP2_VN	CMP2_VN：比较器 2 的 N 输入引脚
43	30		PC4	I/O	GPIO_PC4	PC4：数字 GPIO 功能引脚
					UART0_RX	UART0_RX：串口 0 的接收引脚
					IIC0_SDA	IIC0_SDA：IIC0 的数据引脚

					PWMP1_CH2N	PWMP1_CH2N：高级 PWM1 的通道 2N 引脚
					TIMERP0_HALL1	TIMERP0_HALL1：高级定时器 0 的 HALL1 引脚
					CMP2_VP	CMP2_VP：比较器 2 的 P 输入引脚
44	31	1	PC5	I/O	GPIO_PC5	PC5：数字 GPIO 功能引脚
					TIMERP0_HALL2	TIMERP0_HALL2：高级定时器 0 的 HALL2 引脚
					OPA1_VP	OPA1_VP：运算放大器 1 的 P 输入引脚
45			PC6	I/O	GPIO_PC6	PC6：数字 GPIO 功能引脚
					IIC1_SCL	IIC1_SCL：IIC1 的时钟引脚
					PWMP1_CH1	PWMP1_CH1：高级 PWM1 的通道 1 引脚
					TIMERP1_IN1	TIMERP1_IN1：高级定时器 1 的输入 1 引脚
					TIMERP1_OUT_H	TIMERP1_OUT_H：高级定时器 1 的 H 输出引脚
					OPA1_VN	OPA1_VN：运算放大器 1 的 N 输入引脚
46			PC7	I/O	GPIO_PC7	PC7：数字 GPIO 功能引脚
					IIC1_SDA	IIC1_SDA：IIC1 的数据引脚
					PWMP1_CH2	PWMP1_CH2：高级 PWM1 的通道 2 引脚
					TIMERP1_IN0	TIMERP1_IN0：高级定时器 1 的输入 0 引脚
					TIMERP1_OUT_L	TIMERP1_OUT_L：高级定时器 1 的 L 输出引脚
					OPA1_OUT	OPA1_ OUT：运算放大器 1 的输出引脚

4.3引脚复用功能

表 4-2 引脚复用功能

LQ48	引脚名称	SEL000	SEL001	SEL010	SEL011	SEL100	SEL101	SEL110	SEL111
2	PA0	GPIOA0	PWMP1_PLUS0	PWMP0_PLUS1	TM	WAKEUP0		---	
3	PA1	GPIOA1	XTAL_XI				---		
4	PA2	GPIOA2	XTAL_XO				---		
5	PA3	GPIOA3	CMP0_VN	XTAH_XI	---	---	---		
6	PA4	GPIOA4	CMP0_VP	XTAH_XO	---	---	---		
10	PA5	GPIOA5	UART1_CTS	PWMP1_PLUS1	TIMERP1_IN0	TIMERP1_OUT_L	WAKEUP1	SARADC_CH0	
11	PA6	GPIOA6	UART1_RTS	TIMERP0_IN0	TIMERP1_OUT_H	SARADC_CH1	OPA0_OUT		
12	PA7	GPIOA7	UART1_TX	BREAK_IN1	TIMERP0_OUT_L	SARADC_CH2	OPA0_VP		
13	PA8	GPIOA8	UART1_RX	TIMERP0_IN1	TIMERP0_OUT_H	SARADC_CH3	OPA0_VN	---	
14	PA9	GPIOA9	SPI0_SSN	TIMERP1_IN0	TIMERP1_OUT_L	TM	SARADC_CH4	CMP1_VN	
15	PA10	GPIOA10	SPI0_CLK	SARADC_CH5	CMP1_VP			---	

16	PA11	GPIOA11	SPI0_MISO	PWMB0_CH0	PWMP0_BRAKE0	TIMERP1_IN1	TIMERP1_OUT_H	SARADC_CH6	
17	PA12	GPIOA12	SPI0_MOSI	PWMB0_CH1	PWMP0_CH0N	TIMERP0_IN0	TIMERP0_OUT_L	SARADC_CH7	
18	PA13	GPIOA13	PWMB0_CH2	PWMP0_CH1N	TIMERP0_IN1	TIMERP0_OUT_H	SARADC_CH8	---	
19	PA14	GPIOA14	PWMB1_CH0	PWMP0_CH2N	TIMERP1_IN0	TIMERP1_OUT_L	SARADC_CH9		
20	PA15	GPIOA15	PWMB1_CH1	PWMP0_CH0	TIMERP1_IN1	TIMERP1_OUT_H	---		
21	PB0	GPIOB0	UART2_TX	IIC0_SCL	PWMB1_CH2	PWMP0_CH1			
22	PB1	GPIOB1	UART2_RX	IIC0_SDA	PWMP0_CH2				
25	PB2	GPIOB2	SPI1_SSN	PWMP0_BRAKE1	TIMERP1_HALL0				
26	PB3	GPIOB3	SPI1_CLK	IIC1_SDA	PWMP0_CH0N	TIMERP1_HALL1			
27	PB4	GPIOB4	SPI1_MISO	IIC1_SCL	PWMP1_CH0	PWMP0_CH1N	TIMERP1_HALL2		
28	PB5	GPIOB5	SPI1_MOSI	PWMP1_CH0N	PWMP0_CH2N	TIMERP0_IN0	TIMERP0_OUT_L		
29	PB6	GPIOB6	PWMP0_CH0	TIMERP0_IN1	TIMERP0_OUT_H				
30	PB7	GPIOB7	SPI0_SSN	UART0_TX	IIC0_SCL	PWMP1_BRAKE0	PWMP0_CH1		
31	PB8	GPIOB8	SPI0_CLK	UART0_RX	IIC0_SDA	PWMB0_CH0	PWMP1_BRAKE1	PWMP0_CH2	
32	PB9	GPIOB9	SPI0_MISO	UART0_CTS	PWMB0_CH1	PWMP1_CH0	TIMERP1_IN1	TIMERP1_OUT_H	
33	PB10	GPIOB10	SPI0_MOSI	UART0_RTS	PWMB0_CH2	PWMP1_CH1	PWMP0_PLUS0	TIMERP1_IN0	TIMERP1_OUT_L
34	PB11	GPIOB11	SWDIO	PWMP1_CH2	PWMP0_BRAKE2				

35	PB12	GPIOB12	UART1_TX	IIC1_SCL	PWMP1_CH0N				
36	PB13	GPIOB13	UART1_RX	IIC1_SDA	PWMP1_CH1N				
37	PB14	GPIOB14	SWCLK	UART2_TX	PWMP1_CH2N				
38	PB15	GPIOB15	SPI1_SSN	UART2_RX					
39	PC0	GPIOC0	SPI1_CLK	UART2_CTS	PWMB1_CH0				
40	PC1	GPIOC1	SPI1_MISO	UART2_RTS	PWMB1_CH1	TIMERP0_IN0	TIMERP0_OUT_L		
41	PC2	GPIOC2	SPI1_MOSI	PWMB1_CH2	PWMP1_BRAKE2	TIMERP0_IN1	TIMERP0_OUT_H		
42	PC3	GPIOC3	UART0_TX	IIC0_SCL	PWMP1_CH1N	TIMERP0_HALL0	CMP2_VN		
43	PC4	GPIOC4	UART0_RX	IIC0_SDA	PWMP1_CH2N	TIMERP0_HALL1	CMP2_VP		
44	PC5	GPIOC5	TIMERP0_HALL2	OPA1_VP					
45	PC6	GPIOC6	IIC1_SCL	PWMP1_CH1	TIMERP1_IN1	TIMERP1_OUT_H	OPA1_VN		
46	PC7	GPIOC7	IIC1_SDA	PWMP1_CH2	TIMERP1_IN0	TIMERP1_OUT_L	OPA1_OUT		

5. 功能描述

5.1地址空间映射

32G030 控制器为 32 位通用控制器，提供了 4G 字节的寻址空间，如下表所示。数据格式仅支持小端模式，各模块具体寄存器排布及操作说明在后面章节有详细的描述。

表 5-1 地址空间映射

起始	结束	模块
存储器		
0x00000000	0x0000FFFF	FLASH 区
0x20000000	0x20003FFF	RAM 区
AHB 总线外设		
0x40000000	0x400007FF	SYSCON
0x40000800	0x40000FFF	PMU
0x40001000	0x400017FF	DMA
0x40003000	0x400037FF	CRC
APB1 总线外设		
0x40060000	0x400607FF	GPIOA
0x40060800	0x40060FFF	GPIOB
0x40061000	0x400617FF	GPIOC
0x40064000	0x400647FF	TIMER_BASE0
0x40064800	0x40064FFF	TIMER_BASE1
0x40065000	0x400657FF	TIMER_BASE2
0x40067000	0x400677FF	TIMER_PLUS0
0x40067800	0x40067FFF	TIMER_PLUS1
0x40069000	0x400697FF	RTC
0x4006A000	0x4006A7FF	IWDT
0x4006A800	0x4006AFFF	WWDT
0x4006B000	0x4006B7FF	UART0
0x4006B800	0x4006BFFF	UART1

0x4006C000	0x4006C7FF	UART2
0x4006F000	0x4006F7FF	FLASH_CTRL
APB2 总线外设		
0x400B0000	0x400B07FF	PORTCON
0x400B1000	0x400B17FF	PWM_BASE0
0x400B1800	0x400B1FFF	PWM_BASE1
0x400B4000	0x400B47FF	PWM_PLUS0
0x400B4800	0x400B4FFF	PWM_PLUS1
0x400B8000	0x400B87FF	SPI0
0x400B8800	0x400B8FFF	SPI1
0x400B9000	0x400B97FF	IIC0
0x400B9800	0x400B9FFF	IIC1
0x400BA000	0x400BA7FF	SARADC
0x400BD000	0x400BD7FF	AES128

5.2存储器划分及权限控制

本芯片共有两种存储区域： 64KB FLASH 和 16KB RAM。

64KB FLASH 作为程序存储区使用，具有 2KB 的 NVR 区和 64KB 的 MAIN 区。NVR 区主要用于存储一些我司特有数据，例如工厂码信息、TRIM 数据以及产品的配置信息等。MAIN 区用于存储用户程序，完全开放给用户使用。

16KB RAM 全部作为数据区使用。

5.3中断向量表

中断向量表如下：

表 5-6 32G030 中断向量表

中断源	外设中断
0	WWDT
1	IWDT
2	RTC
3	DMA
4	SARADC
5	TIMER_BASE0
6	TIMER_BASE1
7	TIMER_PLUS0
8	TIMER_PLUS1
9	PWM_BASE0
10	PWM_BASE1
11	PWM_PLUS0
12	PWM_PLUS1
13	UART0
14	UART1
15	UART2
16	SPI0
17	SPI1
18	IIC0

19	IIC1
20	CMP
21	TIMER_BASE2
22	GPIOA5
23	GPIOA6
24	GPIOA7
25	GPIOB0
26	GPIOB1
27	GPIOC0
28	GPIOC1
29	GPIOA
30	GPIOB
31	GPIOC

5.4 ARM Cortex-M0 内核

5.4.1 概述

Cortex-M0 处理器是 32 位多级可配置的 RISC 处理器。它有 AMBA AHB-Lite 接口和嵌套向量中断控制器(NVIC),具有可选的硬件调试功能,可以执行 Thumb 指令,并与其它 Cortex-M 系列兼容。该系列处理器支持两种操作模式 Thread 模式和 Handler 模式。当有异常发生时,处理器进入 Handler 模式。异常返回只能在 Handler 模式下发生。当复位时,处理器会进入 Thread 模式,处理器也可在异常返回时进入到 Thread 模式。下图显示了处理器内核的各个功能模块。

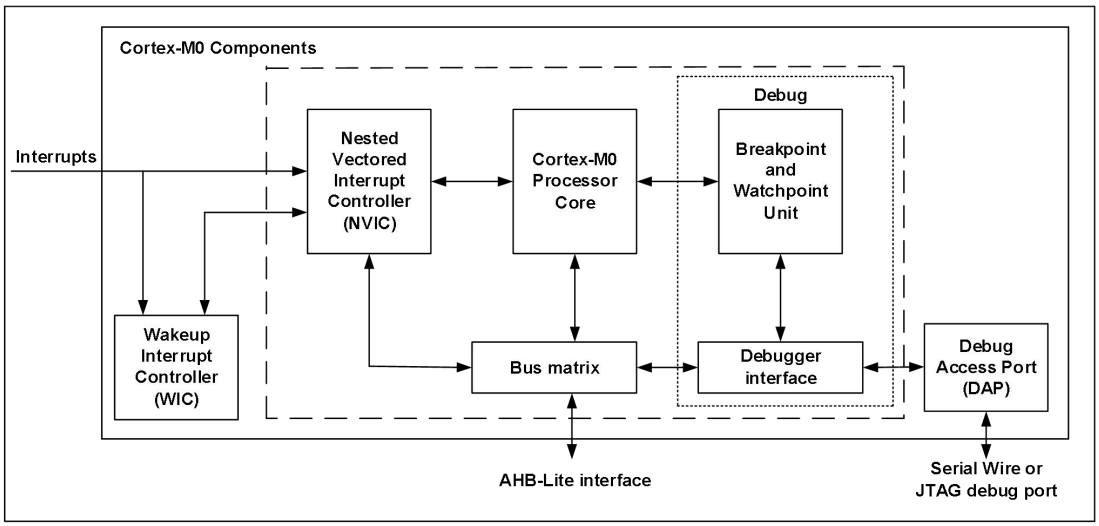


图 5-1 Cortex-M0 处理器功能框图

5.4.2 特性

- 处理器特性:
  - Thumb 指令集
  - Thumb-2 技术
  - 24-bit SYSTICK 定时器
  - 32-bit 硬件乘法器

- 系统接口支持小端（little-endian）数据访问
- 具有确定性，固定延迟的中断处理能力
- 可以丢弃和重新开始多次加载/存储和多周期乘法指令以保证快速中断处理
- 与 C 应用程序二进制接口兼容的异常模式（C-ABI）ARMv6-M（C-ABI） 兼容异常模式允许用户使用纯 C 函数实现中断处理
- 使用等待中断（WFI）从中断返回时直接进入睡眠的 sleepon-exit 特性可以进入低功耗的休眠模式

- NVIC 特性:
  - 32 个外部中断输入， 每个中断具有 4 级优先级
  - 不可屏蔽中断输入（NMI）
  - 支持电平和脉冲触发中断
  - 中断唤醒控制器（WIC）， 支持极低功耗休眠模式
- 调试
  - 四个硬件断点
  - 两个观察点
  - 用于非侵入式代码的程序计数采样寄存器（PCSR）
  - 单步和向量捕获能力
- 总线接口:
  - 单一 32 位的 AMBA-3 AHB-Lite 系统接口， 为所有的系统外设和存储器提供方便的集成
  - 支持 DAP(Debug Access Port)的单一 32 位的从机端

5.4.3 系统定时器（SYSTICK）

概述

Cortex-M0 核内部提供了一个 24 位系统定时器。该定时器使能后装载当前值寄存器（SYST_VAL）内数值并向下递减至 0，并在下个时钟沿重新加载重载寄存器（SYST_LOAD）内数值。计数器再次递减至 0 时，计数器状态寄存器（SYST_CTRL）中标识位 COUNTFLAG 置位，读该位可清零。

复位后，SYST_VAL 寄存器与 SYST_LOAD 寄存器值均未知，因此使用前需初始化，向 SYST_VAL 写入任意值，清零同时复位状态寄存器，保证装载值为 SYST_LOAD 寄存器中数值。

当 SYST_LOAD 寄存器值为 0 时，重新装载后计时器保持为 0，并停止重新装载。

特性

- 24 位系统定时器
- 递减
- 写清零

模块结构框图

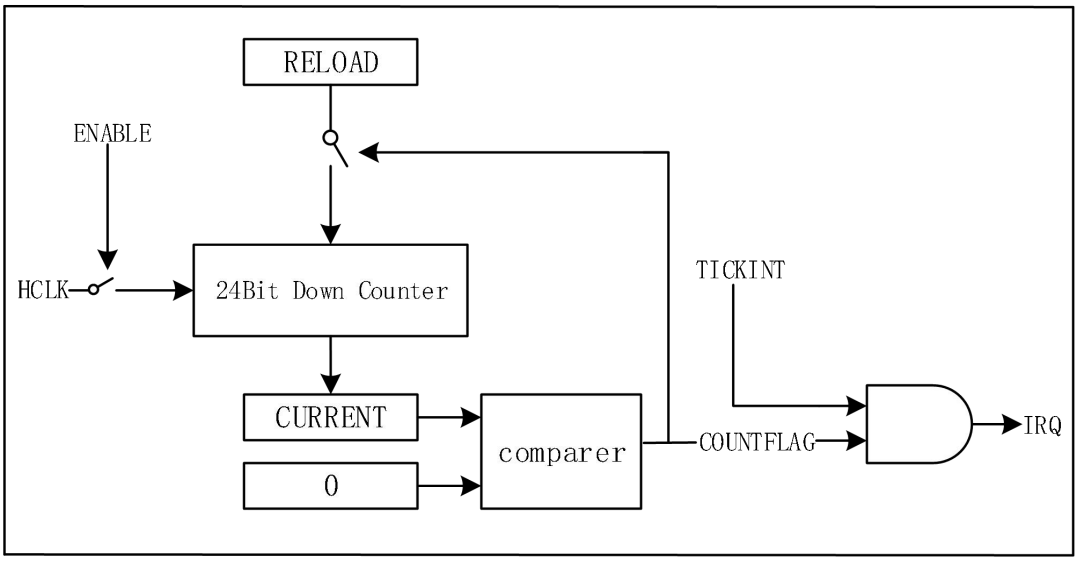


图 5-2　Systick 模块结构图

功能描述

该定时器使能后，装载当前值寄存器（SYST_VAL）内数值并向下递减至 0，并在下个时钟重新加载重载寄存器（SYST_LOAD）内数值。计数器再次递减至 0 时，计数器状态寄存器（SYST_CTRL）中的 标志位 COUNTFLAG 置位，读该位可清零。

复位后，SYST_VAL 寄存器与 SYST_LOAD 寄存器值均未知，因此使用前需初始化，向 SYST_VAL 写入任意值，清零同时复位状态寄存器，保证装载值为 SYST_LOAD 寄存器中数值。

当 SYST_LOAD 寄存器值为 0 时，重新装载后计时器保持为 0，并停止重新装载。该计数器可用作实时系统的滴答定时器或一个简单的计数器。

SysTick 计数时序图如下图所示：

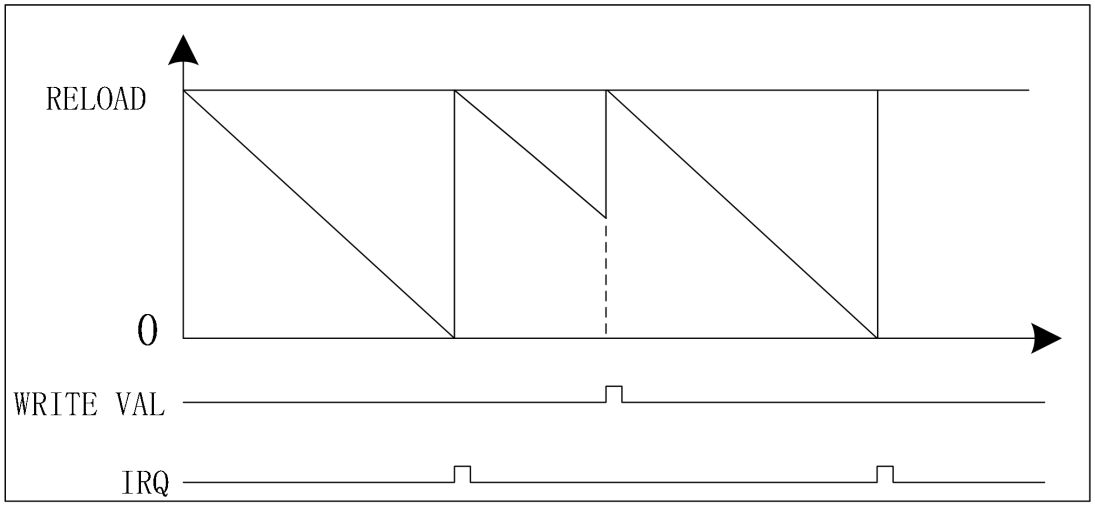


图 5-3 systick 计数时序图

寄存器映射

名称	偏移	类型	复位值	描述
SYSTICK BASE:0xE000E010				
SYST_CTRL	0x00	R/W	0x00	状态寄存器
SYST_LOAD	0x04	R/W	0x00	重载寄存器
SYST_VAL	0x08	R/W	0x00	当前值寄存器

名称	偏移	类型	复位值	描述
SYSTICK	BASE:0xE000E010			
SYST_CTRL	0x00	R/W	0x00	状态寄存器
SYST_LOAD	0x04	R/W	0x00	重载寄存器
SYST_VAL	0x08	R/W	0x00	当前值寄存器

寄存器描述

SYST_CTRL 状态寄存器（0x00）

位域	名称	类型	复位值	描述
31:17	RESERVED	R	0	保留位
16	COUNTFLAG	R	0	计数器计数到 0，此位置 1
15:2	RESERVED	R	0	保留位
1	TINKINT	R/W	0	中断使能位 0：禁能 1：使能
0	ENABLE	R/W	0	定时器使能位 0：禁能 1：使能

SYST_LOAD 重载寄存器（0x04）

位域	名称	类型	复位值	描述
31:24	RESERVED	R	0	保留位
23:0	RELOAD	R/W	0	计数器计数到 0 加载本寄存器的值，重新开始计数

SYST_VAL 当前值寄存器（0x08）

位域	名称	类型	复位值	描述
31:24	RESERVED	R	0	保留位
23:0	VAL	R/W	0	读操作返回当前计数值，写操作会清零该寄存器，同时清除 COUNTFLAG 标志位

5.5电源管理（PMU）

5.5.1 概述

本芯片的供电电压（AVDD）为 2.0-3.6V。通过内置的电压调节器提供给内部数字电路所需的 1.2V 电源。

芯片内部的 A/D 转换器可由独立的 AVDD_ADC 供电，供电电压为 2.0-3.6V，并且可由片外提供独立的 VREF 参考电压。对于不同封装的芯片，A/D 转换器供电电源有可能与芯片主供电电源（AVDD）double bonding 在同一个 PIN 上，具体可参看各封装芯片型号管脚说明。A/D 转换器的 VREF 参考电压在不同封装芯片上也有可能与 AVDD 共同 bonding 在同一个 PIN 上，具体可参看各封装芯片型号管脚说明。

芯片也提供了多种降低功耗的方式，包括：降低系统时钟频率、外设时钟控制以及多种低功耗模式等。

芯片除了上电复位和掉电复位外，还包含看门狗（WDT）复位、管脚复位、CPU 软复位。

5.5.2 特性

- 工作电压是 2.0V-3.6V
- ADC 采用单独的电源供电和参考电源
- 上电复位（POR）和掉电复位(PDR)
- 外部上电复位引脚
- 看门狗复位

工作模式有：正常工作模式（normal）、WFI 模式（WFI）、休眠模式（sleep）、深度休眠模式（deepsleep）、停止模式（stop）

5.5.3 模块结构框图

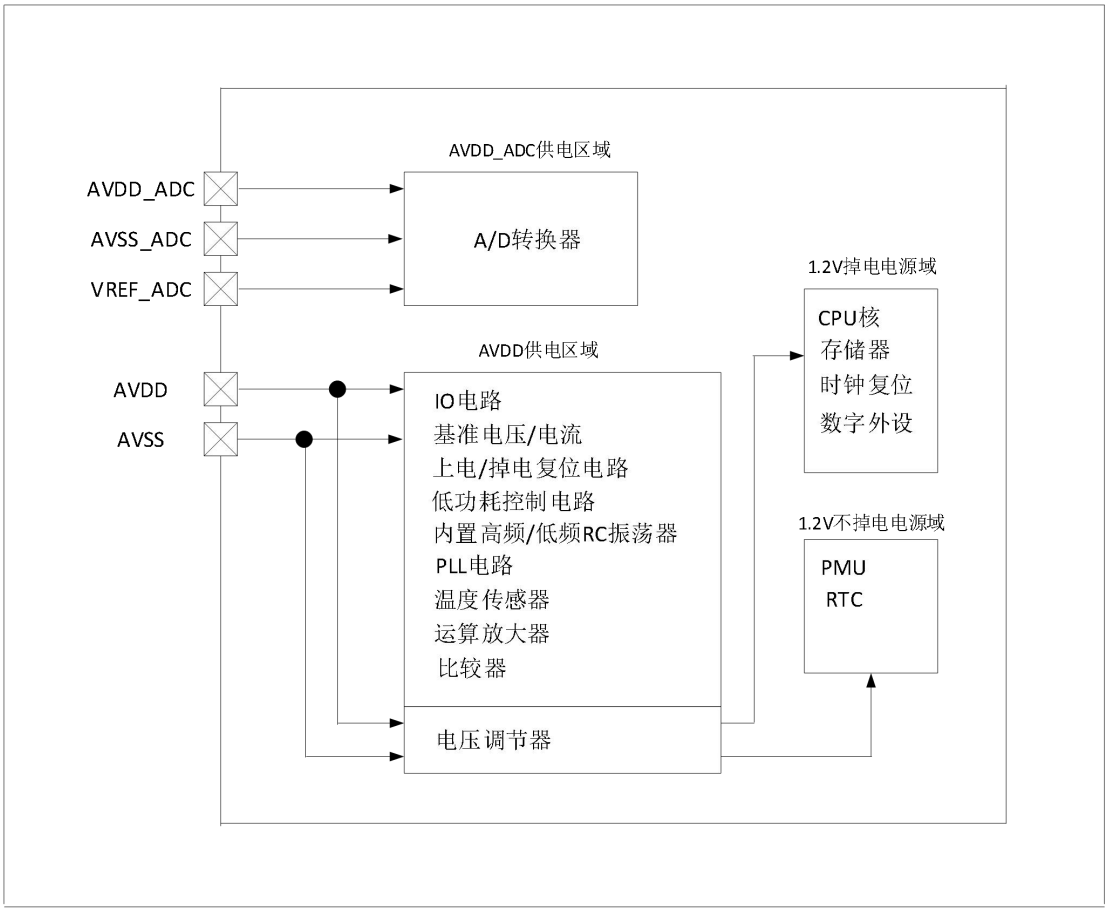


图 5-4　电源模块结构框图

如上图所示，电源/地共有两组：一组是给芯片模拟电路供电的主电源/地（AVDD/AVSS），另外一组是给 A/D 转换器供电的电源/地（AVDD_ADC/AVSS_ADC）。

根据不同的封装，有可能将 AVDD 和 AVDD_ADC 连接在同一个 PIN 上。

A/D 转换器有独立的电源供电（AVDD_ADC/AVSS_ADC），以及独立的外部电压参考输入 PAD（VREF_ADC）。

芯片内部绝大部分模拟电路都由 AVDD 供电，包括：多个电压调节器、IO 电路、基准电压/电流、上电/掉电复位电路、低功耗控制电路、内置高频/低频 RC 振荡器电路、PLL、温度传感器、运算放大器、比较器等。

芯片内数字电路都是工作在 1.2V 电压下，分为两个不同的电源域：1.2V 掉电电源域和 1.2V 不掉电电源域，并且工作电压可由电压调节器调节，以满足不同的低功耗应用场景。

1.2V 掉电电源域主要包括 CPU 核、存储器、时钟复位控制电路以及众多的数字外设等，而 1.2V 不掉电电源域主要包括 PMU 模块、RTC 模块等。

5.5.4 功能描述

电源

HM1030 芯片的工作电压是 2.0V-3.6V。内置电压调节器提供 1.2V 的电源给内部数字电路。为了提高转换精度，ADC 采用单独的电源供电和参考电源。

独立的 A/D 转换器供电电源和参考电压

为了提高 ADC 的转换精度，ADC 使用了独立的电源供电和片外参考电压。单独供电的目的是过滤和屏蔽来自 PCB 板上的干扰和噪声。提供独立的片外参考电压方便在使用过程中提供灵活的参考电压，满足更多的应用场景。

ADC 的电源引脚是 AVDD_ADC。

ADC 的地引脚是 AVSS_ADC。

ADC 的片外参考电压引脚是 VREF_ADC。

假如 VREF_ADC 与 AVDD_ADC 连接在同一个 PIN 上的封装，可以通过内部连接固定电平的方式反推出当前的参考电压源的电压值。

电压调节器

芯片内部电压调节器用于给所有的数字电路提供电源。根据芯片的工作模式它以五种不同的方式工作：

1） 正常工作模式：电压调节器以正常功耗模式提供 1.2V 电源给数字电路，此时所有的数字电路都可正常工作；

2） WFI 模式：只有 CPU 核停止，其它时钟、电源、以及所有外设包括 CPU 核的外设，如 NVIC、嘀嗒时钟（SysTick）等仍然在按照原状态运行。

3） SLEEP 工作模式：电压调节器以低功耗模式提供 1.2V 电源给数字电路，以保证不掉电电源域的数字电路正常工作，掉电电源域的数字电路保存当前状态。

4） DEEPSLEEP 工作模式：电压调节器以低功耗模式提供 1.2V 电源（此电压可通过 TRIM_LPLDO 寄存器做进一步调节降压）给数字电路，以保证不掉电电源域的数字电路正常工作。

5） STOP 模式：电压调节器停止供电。所有的数字电路内容全部丢失。

复位

本芯片共有 6 个复位源：上电复位、掉电复位、管脚复位、看门狗复位以及 CPU 软复位，其中看门狗复位有两种：独立看门狗（IWDT）复位和窗口看门狗（WWDT）复位。

上电复位（POR）和掉电复位（PDR）

芯片内部由一套完整的上电复位（POR）和掉电复位（PDR）电路。当芯片供电电压上升至特定阈值电压时，系统才能正常工作；当低于该特定阈值电压时，芯片保持复位状态，而无需外部复位电路。

上电复位和掉电复位的波形示意图如下所示：

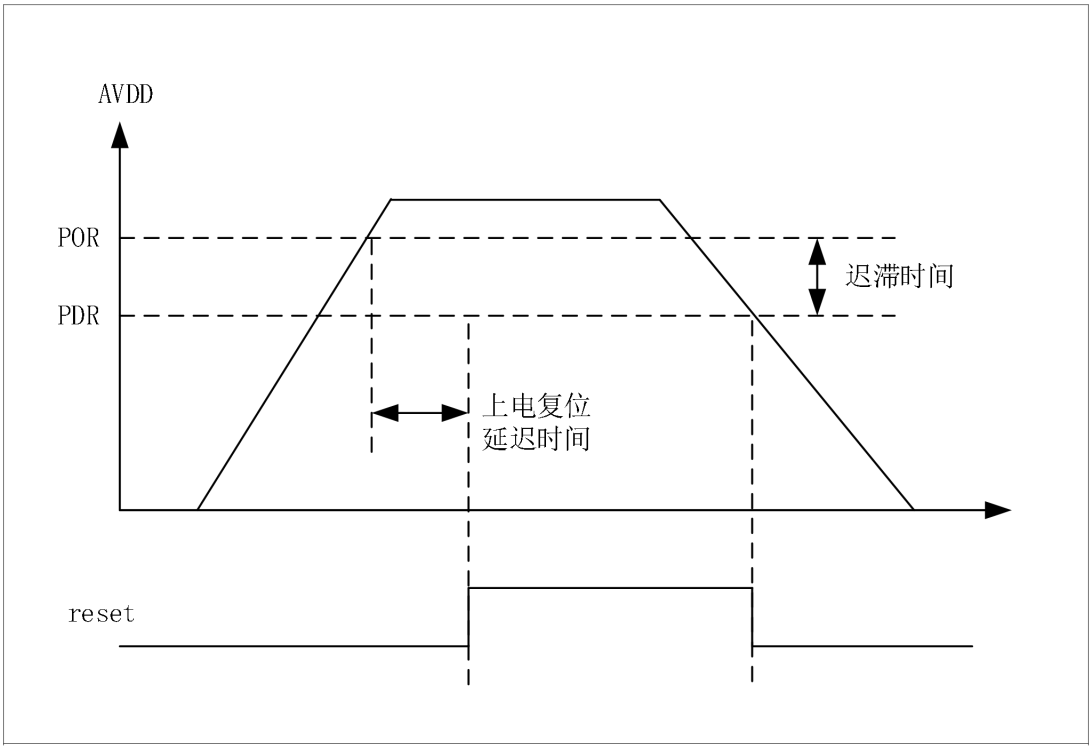


图 5-5　上电复位和掉电复位的波形示意图

其中，reset 为数字电路的总复位信号，复位所有数字电路。

有关上电复位和掉电复位的阈值电压门限、上电复位与掉电复位的迟滞、上电复位延迟时间等细节请参考数据手册电气特性相关章节内容。

管脚复位

芯片支持特定的外部管脚复位，低电平产生复位。

外部管脚复位的波形示意图如下所示：

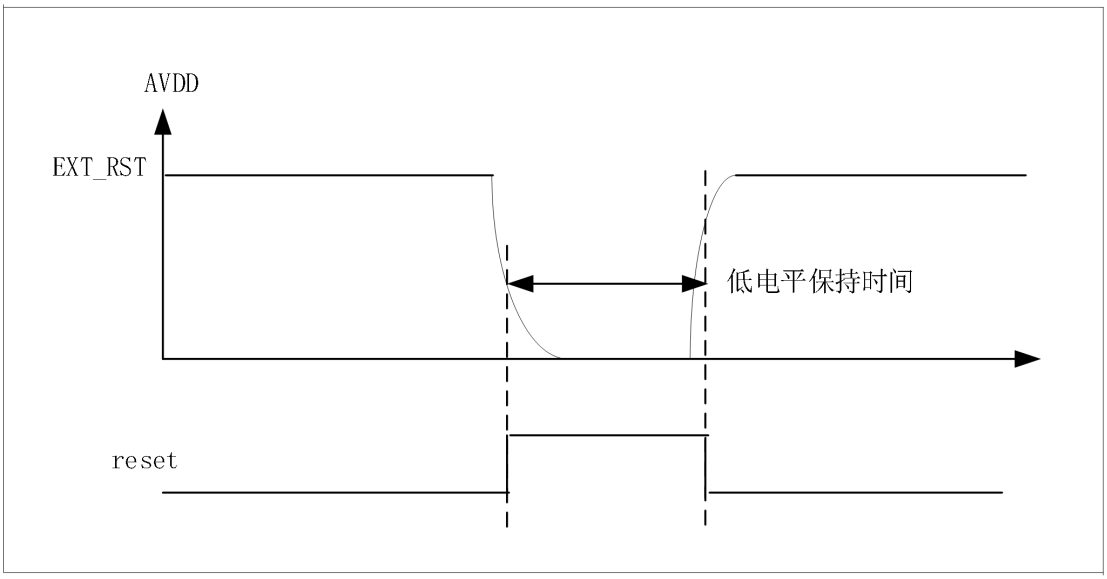


图 5-6　外部管脚复位波形示意图

其中，reset 为数字电路的总复位信号，复位所有数字电路。

有关管脚复位低电平保持时间等细节请参考数据手册电气特性相关章节内容。

看门狗复位

本芯片有两个看门狗：一个是独立看门狗（IWDT），另一个是窗口看门狗（WWDT）。二者都可产生看门狗复位，并且可以复位芯片上的所有数字电路，包括 CPU 核、总线、存储器控制器、外设电路等。

CPU 软复位

通过 ARM CORTEX M0 自有软复位指令，向 0xe000ed0c 地址写入 0x05fa0004，则 CPU 核会发出一个软件复位操作。

该复位会复位绝大多数数字电路。而非掉电域的 PMU/RTC 模块；掉电域的 SYSCON 模块内的 VREF_VOLT_DELTA、RC_FREQ_DELTA、DEVICE_ID0、DEVICE_ID1、DEVICE_ID2、DEVICE_ID3 寄存器，这些电路不会被复位。

复位源复位范围

表 1 复位源复位范围表

复位源	PMU/RTC	绝大多数数字电路	特殊不被 CPU 软复位影响的电路
上电/掉电复位	复位	复位	复位
管脚复位	复位	复位	复位
看门狗复位	复位	复位	复位
CPU 软复位	复位	复位	不复位

其中，特殊不被 CPU 软复位影响的电路指 CPU 软复位章节内容中介绍的电路。绝大多数数字电路指 CPU 核、总线、存储器控制器、时钟复位控制电路、外围设备等。

低功耗模式

在芯片系统电源完成上电，并复位释放以后，芯片处于正常工作状态。此时系统时钟默认采用内部高频 RC 振荡器时钟，输出频率为 24MHz，所有电路可正常工作，CPU 开始进行取指，程序正常运行。

当 CPU 不需要持续运行时，可以通过多种低功耗模式来节省芯片功耗，例如等待某个外部事件时或长期定时等待时。用户可根据最低电源消耗、启动时间要求以及唤醒源等要求，选择一个合适的低功耗模式。

芯片低功耗工作模式主要有以下四种：WFI 模式、SLEEP 模式、DEEPSLEEP 模式核 STOP 模式。

- WFI 模式：只有 CPU 核停止，其它时钟、电源、以及所有外设包括 CPU 核的外设，如 NVIC、嘀嗒时钟（SYSTICK）等仍然在按照原状态运行。
- SLEEP 模式：所有数字电路不掉电。电压调节器（LDO）也将工作在低功耗模式下，并且输出电压可调节，可做到进一步降低整体功耗。此时内部高频 RC 振荡器时钟源关闭，系统时钟关闭，只保留内部低频 RC 振荡器时钟在工作，用于 RTC 和唤醒电路。

- DEEPSLEEP 模式：1.2V 掉电域数字电路掉电，包括 CPU 核、存储器以及大部分外设等数字电路。1.2V 非掉电域数字电路处于有电状态，并且此时的供电电压调节器（LDO）也将工作在低功耗模式下，并且输出电压可调节，可做到进一步降低整体功耗。此时内部高频 RC 振荡器时钟源关闭，只有内部低频 RC 振荡器时钟在工作，用于 RTC 和唤醒电路。
- STOP 模式：1.2V 电源关闭，所有时钟源关闭。

此外，在正常工作状态下，可以通过以下方式降低功耗：

- 降低系统时钟。
- 关闭总线上未被使用的外设时钟。

下表给出了低功耗模式下详细信息：

表 2 低功耗模式详细信息表

模式	进入	唤醒	对数字域的影响	对时钟源的影响	对电压调节器的影响
WFI 模式	WFI	CPU 中断	CPU 时钟关，对系统时钟及其它外设时钟无影响	无	无
SLEEP	将 LPOW_MD 寄存器的 SLEEP 位置 1	所有 IO 或 RTC	关闭系统时钟和外设时钟	RCHF 关闭	开启低功耗模式，并且可以调节输出电压(TRIM_LPLDO)
DEEPSLEEP	将 LPOW_MD 寄存器的 DEEPSLEEP 位置 1	特定 IO 或 RTC	1.2V 掉电域没电。关闭系统时钟和外设时钟	RCHF 关闭	开启低功耗模式，并且可以调节输出电压(TRIM_LPLDO)
STOP	将 LPOW_MD 寄存器的 STOP 位置 1	特定 IO	所有数字域没电	RCHF 和 RCLF 关闭	关闭

各低功耗模式详细说明如下：

降低系统时钟

在正常工作模式下，通过对 CLK_SEL 寄存器中的 DIV_CLK_SEL 进行编程，可以降低系统时钟的频率。详见 11.6.1 章节：时钟选择寄存器（CLK_SEL）。

外设时钟控制

芯片为每个外设提供独立的外设时钟控制，在正常工作模式下，任何时候都可以通过关闭外设时钟来减少功耗。

通过对 DEV_CLK_GATE 寄存器中的各外设时钟控制位进行编程，来开关各个外设模块的时钟。

WFI 模式

通过执行 ARM CPU 核的 WFI 指令，可直接使 CPU 核进入待机状态。在该模式下，只有 CPU 是停止状态，其它所有的 IO 引脚及外设模块都保持它们在正常工作模式时的状态。任意一个被嵌套向量中断控制器响应的外设中断都能将 CPU 核从待机状态下唤醒，继续正常执行指令。

该模式唤醒所需时间最短，由于没有时间损失在中断的进入或退出上，只需要几个系统时钟周期即可完成。

SLEEP 模式

通过软件将 LPOW_MD 寄存器中的 SLEEP 位置 1，可让芯片主动进入 SLEEP 模式。为了使芯片整体功耗在 SLEEP 模式尽量低，可以在进入之前将不需要工作的模块关闭（例如 ADC 模块、运算放大器模块、以及不需要工作的数字功能外设等），节省功耗，另外还可以通过 TRIM_POW3 寄存器的 LPLDO 电压调整位（TRIM_LPLDO）将电压调节器在低功耗模式下的输出电压调低，从而进一步降低功耗。

在 SLEEP 模式下，模拟功耗控制电路会将功耗消耗比较大的模拟模块全部关闭（例如 RCHF 等），只保留基本的 BG、电压调节器以及内置低频 RC 振荡器（RCLF）时钟，并且使电压调节器工作在低功耗模式下，芯片中的高频时钟以及系统时钟也由于内置高频 RC 振荡器（RCHF）的关闭而停止。此时整个数字电路只有 PMU 模块、RTC 模块中的工作在 RCLF 时钟下的电路在正常工作外，其它所有数字电路由于没有时钟而停止。

SLEEP 模式的退出可通过外部 IO 信号唤醒或通过 RTC 定时信号唤醒。PMU 检测到相应的 IO 或 RTC 有效唤醒信号后，将 SLEEP_MODE 信号清为 0，表示退出 SLEEP 模式，模拟功耗控制电路检测到 SLEEP_MODE 信号变为 0 后，将开启唤醒流程，使芯片从 SLEEP 模式恢复到正常工作模式。

其中，所有 IO 都支持 SLEEP 模式的唤醒功能。

该模式唤醒所需时间大约 100us 左右。

SLEEP 模式进入和退出示意图如下所示：

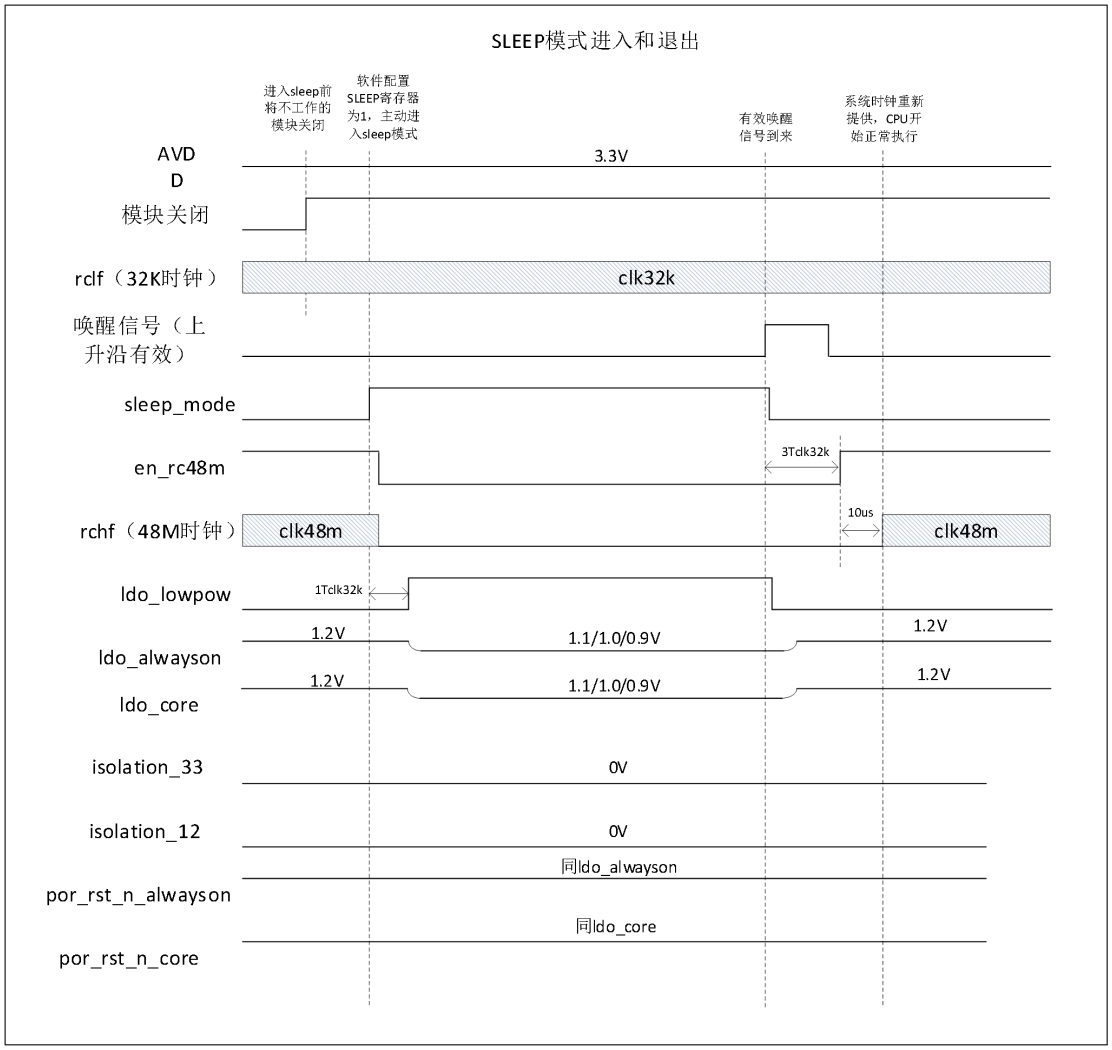


图 5-7　SLEEP 模式进入和退出示意图

DEEPSLEEP 模式

通过软件将 LPOW_MD 寄存器中的 DEEPSLEEP 位置 1，可让芯片主动进入 DEEPSLEEP 模式。为了使芯片整体功耗在 DEEPSLEEP 模式尽量低，可以在进入之前将不需要工作的模块关闭（例如 ADC 模块、运算放大器模块、以及不需要工作的数字功能外设等），节省功耗，另外还可以通过 TRIM_LPLDO 寄存器将电压调节器在低功耗模式下的输出电压调低，从而进一步降低功耗。

在 DEEPSLEEP 模式下，模拟功耗控制电路会将功耗消耗比较大的模拟模块全部关闭（例如 RCHF 等），只保留基本的 BG、电压调节器以及内置低频 RC 振荡器（RCLF）时钟，并且

使电压调节器工作在低功耗模式下，芯片中的高频时钟以及系统时钟也由于内置高频 RC 振荡器（RCHF）的关闭而停止。

1.2V 掉电域数字电路供电也将被断开，使得掉电域的电路全部掉电而达到进一步节省功耗的目的。

1.2V 非掉电域仍然由电压调节器供电，该电源域内的 PMU、RTC 模块中工作在 RCLF 时钟下的电路正常工作外，其它数字电路由于没有时钟而停止。

本芯片，RAM 在掉电域内，数据由于掉电而不会保存。

DEEPSLEEP 模式的退出可通过外部 IO 信号唤醒或通过 RTC 定时信号唤醒。PMU 检测到相应的 IO 或 RTC 有效唤醒信号后，将 DEEPSLEEP_MODE 信号清为 0，表示退出 DEEPSLEEP 模式，模拟功耗控制电路检测到 DEEPSLEEP_MODE 信号变为 0 后，将开启唤醒流程，使芯片从 DEEPSLEEP 模式恢复到正常工作模式

本芯片只有特定 IO 支持 DEEPSLEEP 模式的唤醒功能。

该模式唤醒所需时间大约 140us 左右。

DEEPSLEEP 模式进入和退出示意图如下：

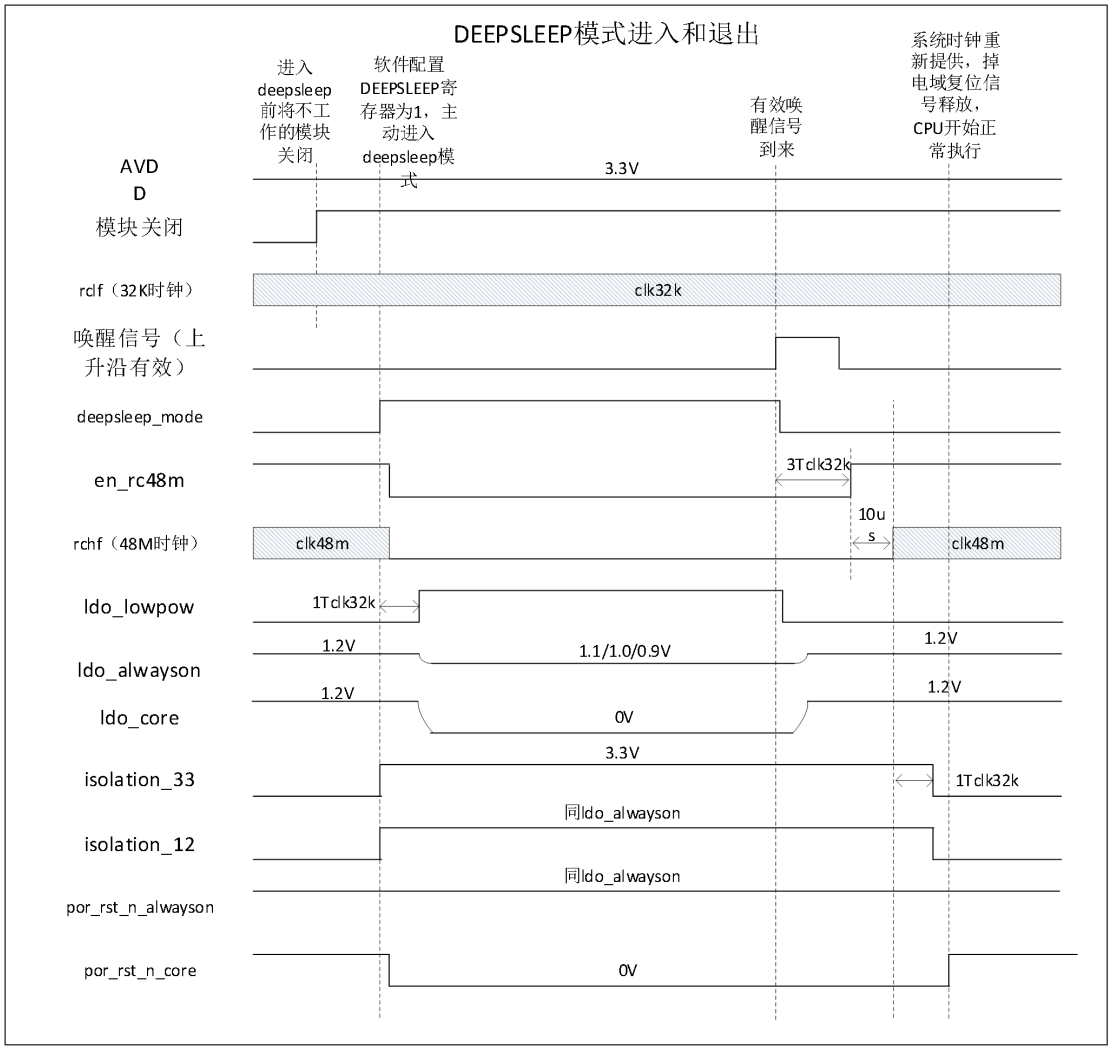


图 5-8　DEEPSLEEP 模式进入和退出示意图

STOP 模式

通过软件将 LPOW_MD 寄存器中的 STOP 位置 1，可让芯片主动进入 STOP 模式。

在 STOP 模式下，电压调节器被关闭，1.2V 数字电路全部掉电，所有的 AVDD 电源域下的模拟模块全部关闭，只保留极少部分模拟唤醒电路在工作。

STOP 模式的退出可通过外部 IO 信号唤醒。当模拟唤醒电路检测到相应唤醒信号后，将开启唤醒流程，使芯片从 STOP 模式恢复到正常工作模式。

本芯片只有特定 IO 支持 STOP 模式的唤醒功能。

该模式唤醒所需时间大约 350us 左右。

STOP 模式进入和退出示意图如下：

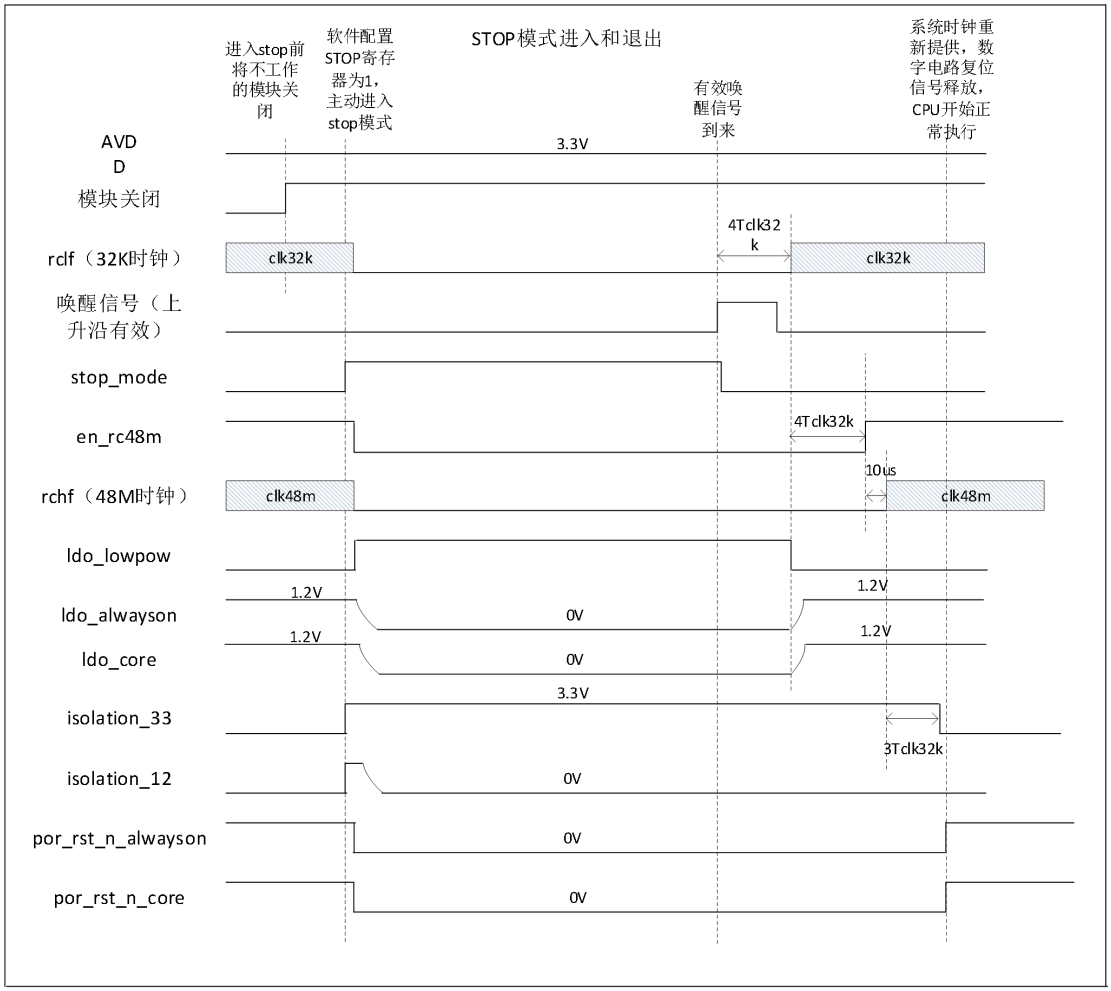


图 5-9　STOP 模式进入和退出示意图

寄存器映射

控制整个芯片进入何种低功耗模式：sleep、deepsleep 和 stop。

本模块为功耗控制模块，包括产生各种进入低功耗控制电路、唤醒电路，以及需要掉电保持的数据（包括模拟 TRIM 值）等。

名称	偏移	类型	复位值	描述
PMU BASE:0x40000800				
LPOW_MD	0x00	R/W	0x00	低功耗模式选择寄存器
LPMD_WKEN	0x04	R/W	0x00	低功耗唤醒源使能寄存器
LPMD_WKST	0x08	R/W	0x00	低功耗唤醒源状态寄存器
CHIP_RST_ST	0x0C	R/W	0x01	芯片复位状态寄存器
SRC_CFG	0x10	R/W	0x03	时钟源配置寄存器
TRIM _POW0	0x20	R/W	0x00	POW0 相关模拟模块 TRIM 寄存器
TRIM _POW1	0x24	R/W	0x00	POW1 相关模拟模块 TRIM 寄存器
TRIM _POW2	0x28	R/W	0x00	POW2 相关模拟模块 TRIM 寄存器
TRIM _POW3	0x2C	R/W	0x00	POW3 相关模拟模块 TRIM 寄存器
TRIM_RCHF	0x30	R/W	0x808	RCHF 时钟模块 TRIM 寄存器
TRIM_RCLF	0x34	R/W	0x810	RCLF 时钟模块 TRIM 寄存器
TRIM_OPA	0x38	R/W	0x00	OPA 模块 TRIM 寄存器
TRIM_PLL	0x3c	R/W	0x00	PLL 模块 TRIM 寄存器
TRIM_LOCK	0x80	R/W	0x00	TRIM 锁定寄存器
DATA_BAKE0	0x100	R/W	0x00	不掉电域数据备份寄存器 0
DATA_BAKE1	0x104	R/W	0x00	不掉电域数据备份寄存器 1
DATA_BAKE2	0x108	R/W	0x00	不掉电域数据备份寄存器 2
DATA_BAKE3	0x10C	R/W	0x00	不掉电域数据备份寄存器 3

寄存器描述

LPOW_MD 寄存器（0x00）

位域	名称	类型	复位值	描述

31:4	RESERVED	R	0	保留位
3	STOP	R/W	0	向该寄存器写 1，芯片进入 STOP 模式 软件写 1，硬件自动清零
2	DEEPSLEEP	R/W	0	向该寄存器写 1，芯片进入 DEEPSLEEP 模式 软件写 1，硬件自动清零
1	SLEEP	R/W	0	向该寄存器写 1，芯片进入 SLEEP 模式 软件写 1，硬件自动清零
0	RESERVED	R	0	保留位

注：芯片从正常工作模式进入低功耗模式，每次只能进入一种低功耗模式，唤醒后将从该低功耗模式退出回到正常工作模式，由软件再配置进入另一种低功耗模式。

LPMD_WKEN 寄存器（0x04）

位域	名称	类型	复位值	描述
31:3	RESERVED	R	0	保留位
2	IO_WKEN	R/W	0	低功耗模式下，IO 唤醒使能 0：禁能　1：使能 注 1：具体哪个 IO 具有唤醒功能可通过 PORTA_WKE、PORTB_WKE、PORTC_WKE 寄存器进行配置
1	RTC_TIM_WKEN	R/W	0	低功耗模式下，RTC 时间信号唤醒使能 1：RTC 时间信号具有低功耗唤醒功能
0	RTC_ALA_WKEN	R/W	0	低功耗模式下，RTC 闹钟信号唤醒使能 1：RTC 闹钟信号具有低功耗唤醒功能

LPMD_WKST 寄存器（0x08）

位域	名称	类型	复位值	描述
31:3	RESERVED	R	0	保留位

2	IO_WKST	R/W	0	低功耗模式下，IO 唤醒标志 1：发生 IO 事件唤醒 0：未发生 IO 事件唤醒 硬件置 1，软件写 1 清除
1	RTC_TIM_WKST	R/W	0	低功耗模式下，RTC 时间唤醒标志 1：发生 RTC 时间事件唤醒 硬件置 1，软件写 1 清除
0	RTC_ALA_WKST	R/W	0	低功耗模式下，RTC 闹钟唤醒标志 1：发生 RTC 闹钟事件唤醒 硬件置 1，软件写 1 清除

CHIP_RST_ST 寄存器（0x0C）

位域	名称	类型	复位值	描述
31:3	RESERVED	R	0	保留位
2	WWDT_RST_ST	R/W	0	WWDT 复位状态标志寄存器 0：表示没有出现 WWDT 复位 1：表示出现 WWDT 复位 写 1 清零
1	IWDT_RST_ST	R/W	0	IWDT 复位状态标志寄存器 0：表示没有出现 IWDT 复位 1：表示出现 IWDT 复位 写 1 清零
0	POR_RST_ST	R/W	1	上电复位状态标志寄存器 0：表示没有出现上电复位 1：表示出现上电复位 写 1 清零

SRC_CFG 寄存器（0x10）

位域	名称	类型	复位值	描述

31:5	RESERVED	R	0	保留位
4	RTC_CLK_SEL	R/W	0	RTC 时钟选择 0：RCLF 1：XTAL
3	XTAL_EN	R/W	0	XTAL 使能控制位 0：关闭 XTAL 1：开启 XTAL
2	XTAH_EN	R/W	0	XTAH 使能控制位 0：关闭 XTAH 1：开启 XTAH
1	RCHF_FSEL	R/W	1	RCHF 频率选择控制位 0：48MHz 1：24MHz
0	RCHF_EN	R/W	1	RCHF 使能控制位 0：关闭 RCHF 1：开启 RCHF

TRIM_POW0 寄存器（0x20）

位域	名称	类型	复位值	描述
31:11	RESERVED	R	0	保留位
10:8	TRIM_TEMPCO_HPBG	R/W	0	HPBG 温度 trim 位
7:4	TRIM_I_HP	R/W	0	HPBG 电流 trim 位
3:0	TRIM_V_HP	R/W	0	HPBG 电压 trim 位

TRIM_POW1 寄存器（0x24）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位

7:4	TRIM_V_LP	R/W	0	LPBG 电压 trim 位
3:0	TRIM_TEMPCO_LPBG	R/W	0	LPBG 温度 trim 位

TRIM_POW2 寄存器（0x28）

位域	名称	类型	复位值	描述
31:0	RESERVED	R	0	保留位

TRIM_POW3 寄存器（0x2C）

位域	名称	类型	复位值	描述
31:4	RESERVED	R	0	保留位
3	TRIM_HPLDO_H	R/W	0	HPLDO 电压调整到 1.264v 0：不变 1：向上调整到 1.264v
2:1	TRIM_LPLDO	R/W	0	LPLDO 电压输出 trim 位 00：1.1V 01：1.0V 10：0.9V 11：0.8V
0	TRIM_PD_UVLO	R/W	0	UVLO33 trim 位 0：在 SLEEP 下，电源电压掉至 1.8V 时，芯片复位；此情况下模拟电路会额外消耗 0.6uA 的功耗； 1：在 SLEEP 下，电源电压掉至 1.3V（+-500mV 的偏差）时，芯片复位；模拟电路会节省 0.6uA 的功耗；

TRIM_RCHF 寄存器（0x30）

位域	名称	类型	复位值	描述

31:12	RESERVED	R	0	保留位
11:8	TRIM_N	R/W	0x8	RCHF N trim 位
7:4	RESERVED	R	0	保留位
3:0	TRIM_P	R/W	0x8	RCHF P trim 位

TRIM_RCLF 寄存器（0x34）

位域	名称	类型	复位值	描述
31:12	RESERVED	R	0	保留位
11:8	TRIM_CS	R/W	0x8	RCLF CS trim 位（粗调位）
7:5	RESERVED	R	0	保留位
4:0	TRIM_FINE	R/W	0x10	RCLF FINE trim 位（精调位）

TRIM_OPA 寄存器（0x38）

位域	名称	类型	复位值	描述
31:20	RESERVED	R	0	保留位
19:15	OPA1_TRIMP	R/W	0	OPA1 的 P 端 TRIM 位
14:10	OPA1_TRIMN	R/W	0	OPA1 的 N 端 TRIM 位
9:5	OPA0_TRIMP	R/W	0	OPA0 的 P 端 TRIM 位
4:0	OPA0_TRIMN	R/W	0	OPA0 的 N 端 TRIM 位

TRIM_PLL 寄存器（0x3C）

位域	名称	类型	复位值	描述
31:4	RESERVED	R	0	保留位
3:0	PLL_R_TRSIM	R/W	0	PLL 的 R 值 TRIM 位

TRIM_LOCK 寄存器（0x80）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:8	TRIM_UNLOCK	W	0	将该寄存器写入 0xAA 后，相应的 TRIM 配置寄存器可以被改写。
7:0	TRIM_LOCK	W	0	将该寄存器写入 0x55 后，相应的 TRIM 配置寄存器不能被改写，用于保护对 TRIM 寄存器的误改写。

DATA_BAK0 寄存器（0x100）

位域	名称	类型	复位值	描述
31:0	DATA_BAK0	R/W	0	数据备份 0

DATA_BAK1 寄存器（0x104）

位域	名称	类型	复位值	描述
31:0	DATA_BAK1	R/W	0	数据备份 1

DATA_BAK2 寄存器（0x108）

位域	名称	类型	复位值	描述
31:0	DATA_BAK2	R/W	0	数据备份 2

DATA_BAK3 寄存器（0x10C）

位域	名称	类型	复位值	描述
31:0	DATA_BAK3	R/W	0	数据备份 3

5.6系统控制（SYSCON）

5.6.1 概述

本芯片通过系统控制单元管理和控制着整体芯片系统的时钟网络，为芯片中的各模块及功能提供时钟，包括系统时钟、各模块外设时钟、各特殊功能时钟。通过该单元还可以独立控制各模块的时钟开或关、系统时钟源的选择及分频、特殊功能时钟分频来达到合理的功耗控制。

5.6.2 特性

- 系统时钟源选择、分频及控制
- 每个外设都有单独的开关
- 具有 128bit 芯片唯一识别码

5.6.3 模块结构框图

共有 5 个时钟源：RCHF（48MHz 内部 RC 振荡器）、RCLF（32768Hz 内部 RC 振荡器）、PLL（最高 72MHz）、XTAH（4-32MHz 晶振）、XTAL（32768Hz 晶振）。

芯片时钟网络结构如下

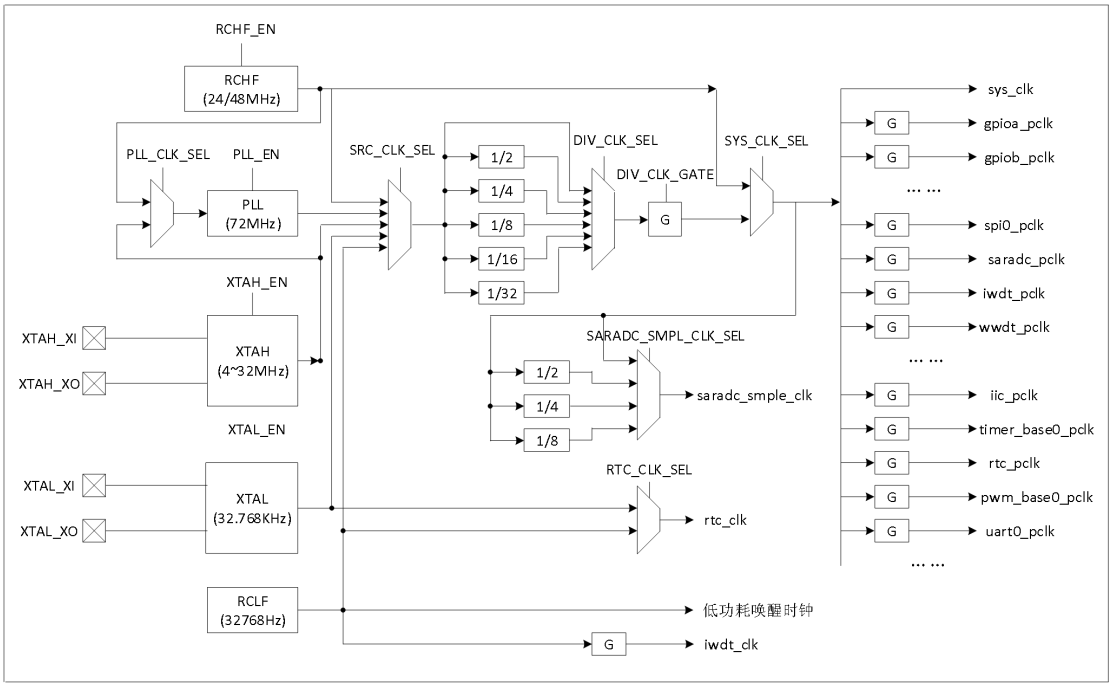


图 5-10 芯片时钟结构图

sys_clk（系统时钟）可选择五个时钟源的其中一个，并且还可以进行 1/2/4/8/16/32 分频。

CPU Cortex-M0 的 FCLK、HCLK 以及 SCLK 都由 sys_clk 供给时钟，所以 M0 内部的 SYSTICK 的时钟源为 sys_clk。

所有的外围设备总线时钟都有各自的时钟控制寄存器来控制其开或关，从而在不使用该模块时可将其时钟关闭，达到节省功耗的目的；并且其总线时钟频率与系统时钟频率一致。

SARADC 的采样时钟通过 CLK_SEL 寄存器的 SARADC_SMPL_CLK_SEL 位可以选择为 sys_clk 的 1/2/4/8 分频，根据采样信号的使用场景不同，可以选择不同的采样时钟频率。

独立看门狗（IWDT）有一个低频的时钟源，用于计数使用，该低频时钟为固定的 RCLF 时钟供给，并且受 DEV_CLK_GATE 寄存器的 IWDT_CLK_GATE 位控制该时钟的开或关。

窗口看门狗（WWDT）采用系统时钟作为计数使用，并且受 DEV_CLK_GATE 寄存器的 WWDT_CLK_GATE 位控制该时钟的开或关。

RTC 的计数时钟可选择为 RCLF 或 XTAL 两个时钟源，通过 DEV_CLK_GATE 寄存器的 RTC_CLK_SEL 位来配置选择。

SLEEP 模式和 DEEPSLEEP 模式固定使用 RCLF 作为唤醒时钟，并且 RCLF 时钟不可关闭。

5.6.4 功能描述

外部高频晶振时钟（XTAH）可由两种时钟源产生：外部晶体/陶瓷谐振器或用户的外部时钟。

为了减少时钟输出的失真和缩短启动稳定时间，晶体/陶瓷谐振器和负载电容必须尽可能地靠近振荡器引脚。并且负载电容必须根据所选择的振荡器来匹配调整。

1、外部晶体/陶瓷谐振器

XTAH 支持外挂 4-32MHz 晶振/陶瓷谐振器，可为芯片提供非常精确的时钟。相关硬件配置可参考如下图所示，并且进一步信息可参考数据手册中的电气特性部分内容。

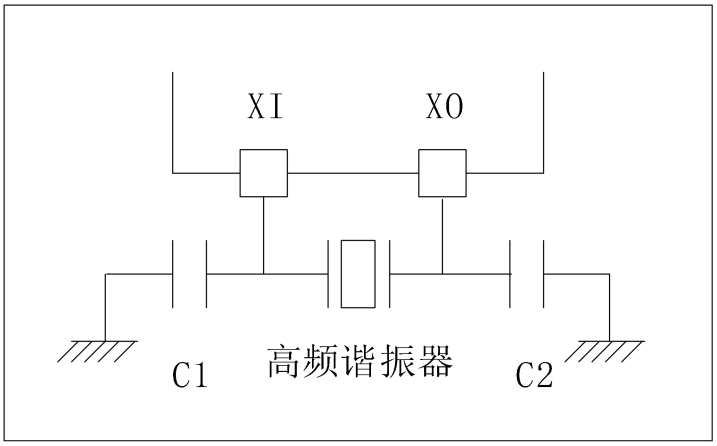


图 5-11 外部高频晶振相关硬件配置图

2、外部时钟

XTAH 也支持外部时钟直接灌入的连接方式，其频率最高可达 48MHz。外部时钟信号必须连接到 XI 引脚，同时要保证 XO 引脚悬空。相关硬件配置可参考下图所示。

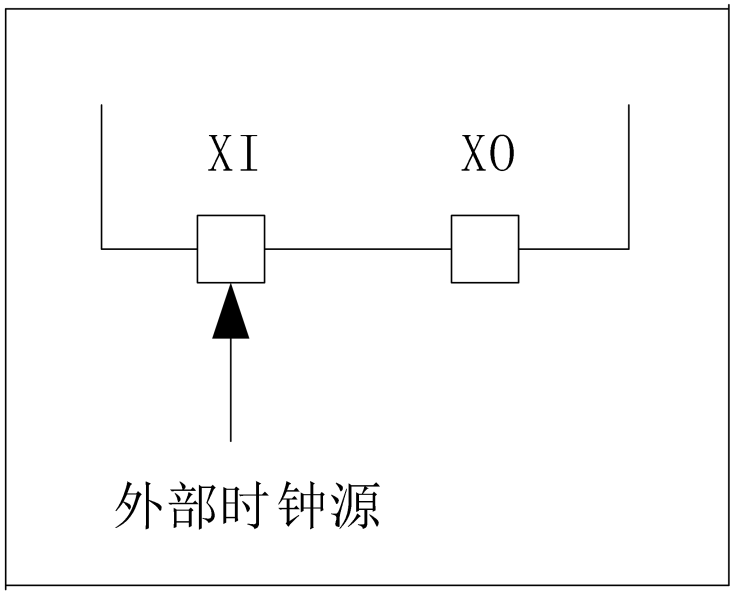


图 5-12 外部时钟相关硬件配置图

XTAH 时钟可以通过配置 PMU 中的 SRC_CFG 寄存器的 XTAH_EN 位来启动或关闭。启动后，至少要等待 10ms 才能正常使用。

RCHF 时钟

RCHF 时钟信号由芯片内部高频 RC 振荡器（48MHz）产生。可以通过配置 PMU 中的 SRC_CFG 寄存器的 RCHF_EN 位来启动或关闭，并且可以通过配置 PMU 中的 SRC_CFG 寄存器的 RCHF_FSEL 位选择 48MHz 输出还是 24MHz 输出。

RCHF RC 振荡器可以在不需要任何外部器件的条件下提供系统所需时钟。它的启动时间比 XTAH 晶体振荡器要短，但即使在校准之后它的时钟频率精度仍然较差。进一步信息可参考数据手册中的电气特性部分内容。

校准

制造工艺决定了不同芯片的 RC 振荡器频率会有不同，这就是为什么每颗芯片的 RCHF 时钟频率在出厂前已经被校准到±1%（25℃）以内的原因。系统复位后，通过软件可将生产测试中写入存储器特定地址的校准值装载到 TRIM_RCHF 寄存器的相应位中，并且还可以

将生产测试中写入存储器特定地址的实际频率值装载到 RC_FREQ_DELTA 寄存器的 RCHF_DELTA 和 RCHF_SIG 位中，在使用时可以获知当前频率与理想频率的偏差，从而获取更精准的时钟频率。

如果用户在使用场景基于不同的电压或环境温度，这将会影响 RC 振荡器的精度。用户也可以通过 TRIM_RCHF 寄存器中的 TRIM_P 位来调整 RCHF 频率。

PLL 时钟

PLL 可以用两种时钟源（XTAH 或 RCHF）之一作为参考时钟输入，产生倍频的时钟输出。参见时钟网络结构图和 CLK_SEL 寄存器中的 PLL_CLK_SEL 位说明。

PLL 在使能之前，必须完成 PLL 的相关各项配置（时钟源选择、预分频系数和倍频系数等），同时应该在它们的输入时钟稳定后才能使能。并且一旦使能 PLL，这些参数将不能再被改变。

当需要改变 PLL 的配置时，必须先将已启动的 PLL 关闭后再对相应配置进行改变，然后再重新使能 PLL。

PLL 使能后，可通过 PLL_ST 寄存器的 PLL_LOCK 状态位判断 PLL 是否已完成锁定，PLL 时钟只有在锁定后才能使用。PLL 使能后大约需要 30us 才能锁定。进一步信息可参考数据手册中的电气特性部分内容。

PLL 时钟频率主要由输入时钟通过 PLL_CTRL 寄存器的 PLL_M 位和 PLL_N 位这两个参数使用倍频的方式获得。其计算公式如下：

$$f_{pll} = \frac{f_{in}}{M} \times N$$

其中 fpll 为 PLL 输出时钟频率，fin 为输入时钟频率。M 为寄存器 PLL_M 配置值，N 为寄存器 PLL_N 配置值。

本芯片 PLL 模块可通过寄存器 CLK_SEL 的 PLL_CLK_SEL 位选择两种输入时钟源，分别为 RCHF 或 XTAH。其中 PLL 的参考时钟频率通过输入时钟/PLL_M 获得，并且参考时钟频率范围为 3MHz-6MHz。

本芯片 PLL 输出的最高频率为 72MHz，输入时钟、PLL_M 以及 PLL_N 的配置关系推荐使用下表方式：

输入信号源	输入频率	PLL_M	PLL 参考时钟频率	PLL_N	PLL 时钟
RCHF	24	8	3	24	72
RCHF	24	6	4	18	72
RCHF	24	4	6	12	72
RCHF	24	6	4	16	64
RCHF	24	6	4	14	56
RCHF	24	6	4	12	48
XTAH	4	1	4	18	72
XTAH	8	2	4	18	72
XTAH	12	2	6	12	72
XTAH	12	4	3	24	72
XTAH	16	4	4	18	72
XTAH	24	4	6	12	72
XTAH	24	6	4	18	72
XTAH	24	8	3	24	72
XTAH	32	8	4	18	72
XTAH	32	8	4	16	64

XTAL 时钟

外部低频晶振时钟（XTAL）可外挂一个 32768Hz 的晶体/陶瓷谐振器。它可为实时时钟或者其它电路提供一个低功耗并且精准的 32768Hz 的时钟源。

XTAL 时钟可以通过配置 SRC_CFG 中的 XTAL_EN 位来启动或关闭。启动后，至少要等待 2s 才能正常使用。

XTAL 支持外挂 32768Hz 晶振/陶瓷谐振器，可为芯片提供非常精确的时钟。相关硬件配置可参考下图所示，并且进一步信息可参考数据手册中的电气特性部分内容。

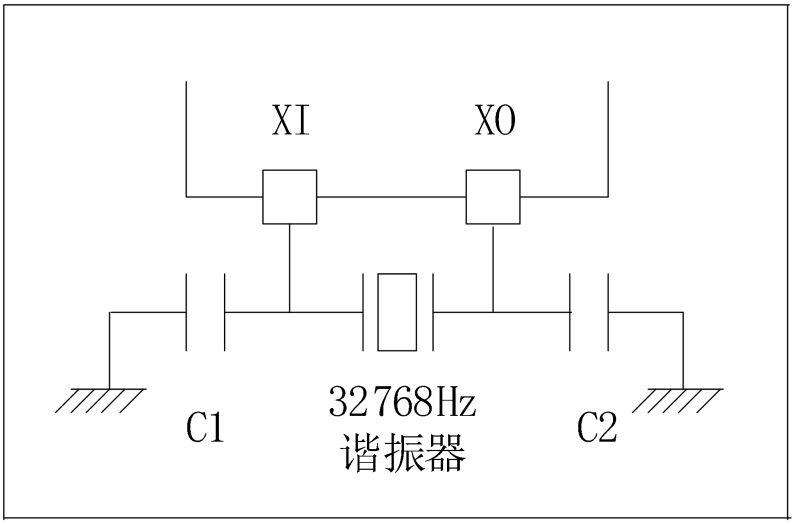


图 5-13 外部低频晶振相关硬件配置图

RCLF 时钟

RCLF 时钟信号由芯片内部低频 RC 振荡器（32768Hz）产生，担当一个低功耗低频时钟源的角色，它可以在 SLEEP 和 DEEPSLEEP 模式下保持运行，为 IWDT、实时时钟和唤醒电路提供时钟。

RCLF 时钟频率为 32768Hz，并且该时钟不可关闭。进一步信息可参考数据手册中的电气特性部分内容。

校准

制造工艺决定了不同芯片的 RC 振荡器频率会有不同，这就是为什么每颗芯片的 RCLF 时钟频率在出厂前已经被校准到±1%（25℃）以内的原因。系统复位后，通过软件可将生产测试中写入存储器特定地址的校准值装载到 TRIM_RCLF 寄存器的相应位中，并且还可以将生产测试中写入存储器特定地址的实际频率值装载到 RC_FREQ_DELTA 寄存器的 RCLF_DELTA 和 RCLF_SIG 位中，在使用时可以获知当前频率与理想频率的偏差，从而获取更精准的时钟频率。

如果用户在使用场景基于不同的电压或环境温度，这将会影响 RC 振荡器的精度。用户也可以通过 TRIM_RCLF 寄存器中的 TRIM_FINE 位来调整 RCLF 频率。

系统时钟（sys_clk）选择

系统复位后，系统时钟（sys_clk）以 RCHF 作为上电后默认的时钟源，并且 RCHF 时钟输出频率为 24MHz。

参看时钟网络结构，系统时钟可通过 CLK_SEL 寄存器中的 SYS_CLK_SEL 位在 RCHF 时钟和 DIV_CLK 时钟之间进行随时切换选择。

而 DIV_CLK 时钟的时钟源和分频如果有更改，则系统时钟一定要先将 SYS_CLK_SEL 配置为 0，系统时钟选择为 RCHF 时钟，以此来保证在切换时钟源过程中对系统时钟不会产生影响。接下来通过将 DIV_CLK_GATE 寄存器配置为 0 关闭 DIV_CLK 时钟，然后再通过 CLK_SEL 寄存器中的 SRC_CLK_SEL 位和 DIV_CLK_SEL 位选择 DIV_CLK 时钟将要输出的频率，选择后再将 DIV_CLK_GATE 配置为 1，打开 DIV_CLK 时钟输出，最后再配置 SYS_CLK_SEL 为 1，使系统时钟选择为 DIV_CLK 时钟输出。

RTC 时钟

通过设置 SRC_CFG 寄存器中的 RTC_CLK_SEL 位，RTCCLK 时钟源可以由 XTAL 或 RCLF 时钟提供。

在 RTC 使能之前，必须完成 RTC 的相关各项配置（时钟源选择等），同时应在改变 RTC_CLK_SEL 位至少 70us 后才能使能。

无论在正常工作模式、SLEEP 模式或 DEEPSLEEP 模式下，都可以根据应用进行灵活选择。

独立看门狗（IWDT）时钟

独立看门狗的计数时钟源由 RCLF 时钟提供，该时钟不能被关闭，并且受到 DEV_CLK_GATE 寄存器中的 IWDT_CLK_GATE 位控制，只有当该位配置为 1 时，IWDTCLK 才能正常使用。

窗口看门狗（WWDT）时钟

窗口看门狗的计数时钟与该模块总线时钟和系统时钟同频率，并且受到 DEV_CLK_GATE 寄存器中的 WWDT_CLK_GATE 位控制，只有当该位配置为 1 时，WWDTCLK 才能正常使用。

SARADC 采样时钟

SARADC 采样时钟可以通过 CLK_SEL 寄存器中的 SARADC_SMPL_CLK_SEL 位选择以下四种频率：系统时钟、系统时钟 2 分频、系统时钟 4 分频和系统时钟 8 分频。

低功耗模式唤醒时钟

在 SLEEP 和 DEEPSLEEP 这两个低功耗模式下，只有 RCLF 时钟在工作，并且低功耗模式的唤醒也通过该时钟驱动。

128 位芯片唯一识别码

本芯片具有 128bit 的芯片唯一识别码，每颗具有唯一性。

产品唯一的身份标识非常适合：

- 用来作为序列号(例如 USB 字符序列号或者其他的终端应用)；
- 用来作为密码，在编写闪存时，将此唯一标识与软件加解密算法结合使用，提高代码在闪存存储器内的安全性；
- 用来激活带安全机制的自举过程；

该 128bit 识别码可通过寄存器 CHIP_ID0、CHIP_ID1、CHIP_ID2 和 CHIP_ID3 这四个寄存器读出使用。

寄存器映射

名称	偏移量	类型	复位值	描述
SYSCON BASE: 0x40000000				
CLK_SEL	0x00	R/W	0x02	时钟选择寄存器
DIV_CLK_GATE	0x04	R/W	0x01	分频时钟门控寄存器
DEV_CLK_GATE	0x08	R/W	0x00	外设时钟门控寄存器
RC_FREQ_DELTA	0x78	R/W	0x00	RCHF/RCLF 真实频率值差值寄存器
VREF_VOLT_DELTA	0x7C	R/W	0x00	VREF 真实电压值差值寄存器
CHIP_ID0	0x80	R/W	0x00	设备 ID 寄存器 0
CHIP _ID1	0x84	R/W	0x00	设备 ID 寄存器 1
CHIP _ID2	0x88	R/W	0x00	设备 ID 寄存器 2
CHIP _ID3	0x8C	R/W	0x00	设备 ID 寄存器 3
PLL_CTRL	0x180	R/W	0x2c8	PLL 控制寄存器
PLL_ST	0x184	R/W	0x00	PLL 状态寄存器

寄存器描述

CLK_SEL 寄存器（0x00）

位域	名称	类型	复位值	描述
31:12	RESERVED	R	0	保留位
11	PLL_CLK_SEL	R	0	PLL 输入时钟选择 0：RCHF 1：XTAH
11:10	SARADC_SMPL_CLK_SEL	W	0	SARADC 采样时钟选择 00：系统时钟的 1 分频 01：系统时钟的 2 分频 10：系统时钟的 4 分频 11：系统时钟的 8 分频
10:9	SARADC_SMPL_CLK_SEL	R	0	SARADC 采样时钟选择 00：系统时钟的 1 分频 01：系统时钟的 2 分频 10：系统时钟的 4 分频 11：系统时钟的 8 分频
8	RESERVED	R	0	保留位
7	RESERVED	R	0	保留位
7	PLL_CLK_SEL	W	0	PLL 输入时钟选择 0：RCHF 1：XTAH
6:4	SRC_CLK_SEL	R/W	0	源时钟（SRC_CLK）选择 000：RCHF 001：RCLF 010：XTAH 011：XTAL 100：PLL 其他：保留

3:1	DIV_CLK_SEL	R/W	01	分频时钟（DIV_CLK）选择 000：SRC_CLK 的 1 分频 001：SRC_CLK 的 2 分频 010：SRC_CLK 的 4 分频 011：SRC_CLK 的 8 分频 100：SRC_CLK 的 16 分频 101：SRC_CLK 的 32 分频 其他：保留
0	SYS_CLK_SEL	R/W	0	系统时钟选择 0：RCHF 时钟 1：DIV_CLK 时钟

DIV_CLK_GATE 寄存器（0x04）

位域	名称	类型	复位值	描述
31:1	RESERVED	R	0	保留位
0	DIV_CLK_GATE	R/W	0x1	分频时钟门控 1：分频时钟输出 0：分频时钟禁止 注：当需要改变 DIV_CLK_SEL 或 SRC_CLK_SEL 寄存器时，需要先将系统时钟切换至 RCHF，然后将该寄存器置为 0，使 DIV_CLK 关闭，最后再改变 DIV_CLK_SEL 或 SRC_CLK_SEL 的值，从而保证时钟的可靠性。

DEV_CLK_GATE 寄存器（0x08）

位域	名称	类型	复位值	描述
31:29	RESERVED	R	0	保留位
28	AES_CLK_GATE	R/W	0	AES128 模块时钟门控

27	CRC_CLK_GATE	R/W	0	CRC 模块时钟门控
26	RESERVED	R	0	保留位
25	SARADC_CLK_GATE	R/W	0	SARADC_CTRL 模块时钟门控
24	WWDT_CLK_GATE	R/W	0	WWDT 模块时钟门控
23	IWDT_CLK_GATE	R/W	0	IWDT 模块时钟门控
22	RTC_CLK_GATE	R/W	0	RTC 模块时钟门控
21	PWM_PLUS1_CLK_GATE	R/W	0	PWM_PLUS1 模块时钟门控
20	PWM_PLUS0_CLK_GATE	R/W	0	PWM_PLUS0 模块时钟门控
19	RESERVED	R	0	保留位
18	PWM_BASE1_CLK_GATE	R/W	0	PWM_BASE1 模块时钟门控
17	PWM_BASE0_CLK_GATE	R/W	0	PWM_BASE0 模块时钟门控
16	TIMER_PLUS1_CLK_GATE	R/W	0	TIMER_PLUS1 模块时钟门控
15	TIMER_PLUS0_CLK_GATE	R/W	0	TIMER_PLUS0 模块时钟门控
14	TIMER_BASE2_CLK_GATE	R/W	0	TIMER_BASE2 模块时钟门控
13	TIMER_BASE1_CLK_GATE	R/W	0	TIMER_BASE1 模块时钟门控
12	TIMER_BASE0_CLK_GATE	R/W	0	TIMER_BASE0 模块时钟门控
11	SPI1_CLK_GATE	R/W	0	SPI1 模块时钟门控
10	SPI0_CLK_GATE	R/W	0	SPI0 模块时钟门控
9	RESERVED	R	0	保留位
8	UART2_CLK_GATE	R/W	0	UART2 模块时钟门控
7	UART1_CLK_GATE	R/W	0	UART1 模块时钟门控
6	UART0_CLK_GATE	R/W	0	UART0 模块时钟门控
5	IIC1_CLK_GATE	R/W	0	IIC1 模块时钟门控

4	IIC0_CLK_GATE	R/W	0	IIC0 模块时钟门控
3	RESERVED	R	0	保留位
2	GPIOC_CLK_GATE	R/W	0	GPIOC 模块时钟门控
1	GPIOB_CLK_GATE	R/W	0	GPIOB 模块时钟门控
0	GPIOA_CLK_GATE	R/W	0	GPIOA 模块时钟门控

RC_FREQ_DELTA 寄存器（0x78）

位域	名称	类型	复位值	描述
31	RCHF_SIG	R/W	0	1：表示 RCHF_DELTA 为正值 0：表示 RCHF_DELTA 为负值
30:11	RCHF_DELTA	R/W	0	RCHF 实际测试频率与 48MHz 的差值 注：真实频率为 48MHz 与差值的和。
10	RCLF_SIG	R/W	0	1：表示 RCLF_DELTA 为正值 0：表示 RCLF_DELTA 为负值
9:0	RCLF_DELTA	R/W	0	RCLF 实际测试频率与 32.768KHz 的差值 注：真实频率为 32.768KHz 与差值的和。

VREF_VOLT_DELTA 寄存器（0x7C）

位域	名称	类型	复位值	描述
31:7	RESERVED	R	0	保留位
6	VREF_SIG	R/W	0	1：表示 VREF_DELTA 为正值 0：表示 VREF_DELTA 为负值
5:0	VREF_DELTA	R/W	0	VREF 实际测试参考电压值与理论值的差值（单位为 mv） 注：真实电压值为理论值与差值的和。

CHIP_ID0 寄存器（0x80）

位域	名称	类型	复位值	描述
31:0	CHIP_ID0	R/W	0	设备 ID 寄存器 0

CHIP_ID1 寄存器（0x84）

位域	名称	类型	复位值	描述
31:0	CHIP_ID1	R/W	0	设备 ID 寄存器 1

CHIP_ID2 寄存器（0x88）

位域	名称	类型	复位值	描述
31:0	CHIP_ID2	R/W	0	设备 ID 寄存器 2

CHIP_ID3 寄存器（0x8C）

位域	名称	类型	复位值	描述
31:0	CHIP_ID3	R/W	0	设备 ID 寄存器 3

PLL_CTRL 寄存器（0x180）

位域	名称	类型	复位值	描述

31:11	RESERVED	R	0	保留位
10:6	PLL_M	R/W	0xb	PLL 参考时钟分频 00000：1 分频 00001：2 分频 00010：3 分频 00011：4 分频 …… 11110：31 分频 11111：32 分频
5:1	PLL_N	R/W	0x4	PLL Feedback 时钟分频 00000：2 分频 00001：4 分频 00010：6 分频 00011：8 分频 …… 11110：62 分频 11111：64 分频
0	PLL_EN	R/W	0	PLL 使能控制位 0：关闭 PLL 1：开启 PLL 注：PLL 开启后，至少 30us 才能锁定。

PLL_ST 寄存器（0x184）

位域	名称	类型	复位值	描述
31:1	RESERVED	R	0	保留位
0	PLL_LOCK	R	0	PLL 锁定状态位 0：未锁定 1：锁定，可以使用 PLL 时钟

5.7IO 功能配置（PORTCON）

5.7.1 概述

每个 IO 都可独立配置其相应功能，通过 PORTx_SELx 寄存器可选择该 IO 复用为何种功能，并且根据手册中列出的每个 IO 端口的特定硬件特性，由软件设置相应寄存器分别配置成多种模式。

5.7.2 特性

- 每个 IO 根据其复用的数字功能不同，可配置为特定数字功能
- 每个 IO 根据其复用的模拟功能不同，可配置为特定模拟功能
- 独立的输入上拉使能控制
- 独立的输入下拉使能控制
- 独立的开漏输出或推挽输出模式
- 独立的输入使能控制
- 独立的 IO 唤醒使能
- 特定的 IO 唤醒有效沿选择控制
- 输入迟滞选择控制
- 输出驱动能力选择控制
- 输入上拉电阻阻值选择控制

以上功能，大部分功能每个 IO 端口都可以独立自由编程，而有些功能控制是全局性的，对所有 IO 起作用，因此在配置时根据相应功能寄存器描述进行选择使用。

5.7.3 模块结构框图

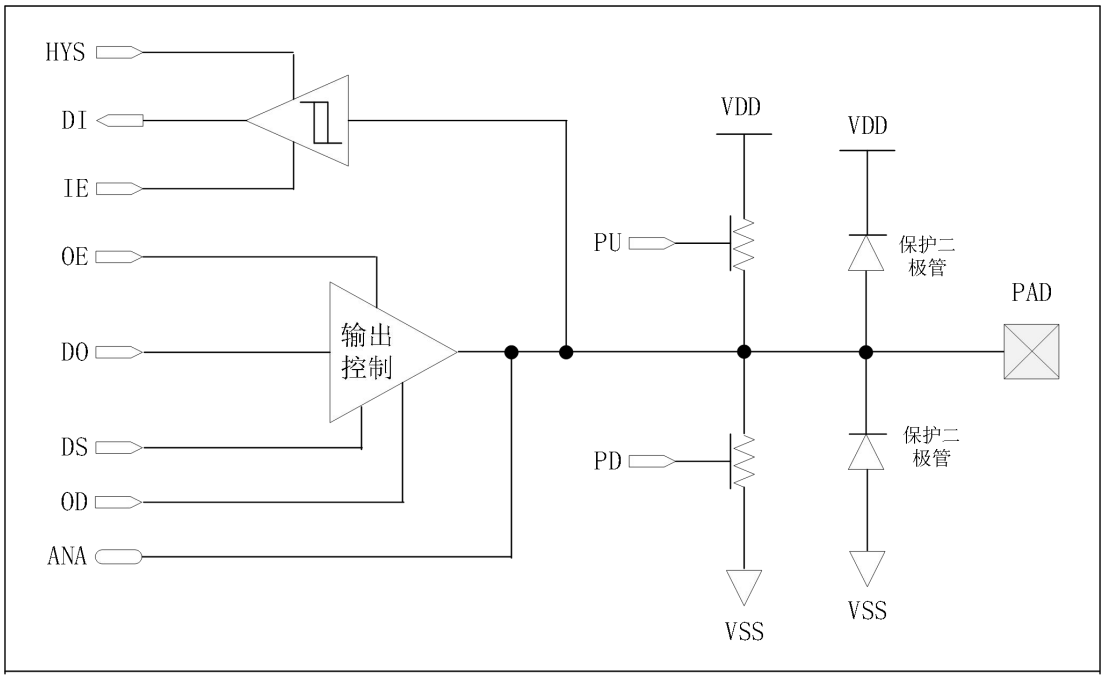


图 5-14　PORTCON 模块结构框图

其中：

HYS 为输入迟滞选择控制端，可由特定寄存器统一配置；

IE 为输入使能控制端，可由 IO 各自独立的寄存器直接配置；

DS 为输出驱动能力控制端，可由特定寄存器统一配置；

OD 为输出模式控制端，可由 IO 各自独立的寄存器直接配置；

PU 为输入上拉使能控制端，可由 IO 各自独立的寄存器直接配置；

PD 为输入下拉使能控制端，可由 IO 各自独立的寄存器直接配置；

ANA 为模拟信号通道，直接与由 IO 各自定义模拟信号连通；

DI 为由 PAD 输入给芯片内部的输入信号端。通过数字功能复用选择，可作为相应功能的输入信号；

DO 为由芯片内部输出给 PAD 的输出信号端。通过数字功能复用选择，可作为相应功能的输出信号；

OE 为输出使能控制端。其与 DO 共同存在，当该 IO 配置的功能为输出时，则硬件会将 OE 自动打开，否则将 OE 关闭；

5.7.4 功能描述

引脚输入使能

本芯片引脚作为输入或需要输入的外设时， 则必须要将相应 IO 的输入使能寄存器（PORTx_IE）配置为有效。当 IO 对应的输入使能寄存器配置为 1 时，即 IE 有效，输入使能打开，PAD 上的电平状态可输入进芯片内部至 DI 端，获取芯片当前外部状态。

数字复用功能选择

不同的 IO 可复用成不同的数字功能，例如 GPIO、SPI、UART 等，IO 的方向取决于复用的具体功能要求。可由 PORTx_SELx 寄存器直接配置各 IO 要实现的功能。

- 输入功能描述：

输入功能选择示意图如下图所示：

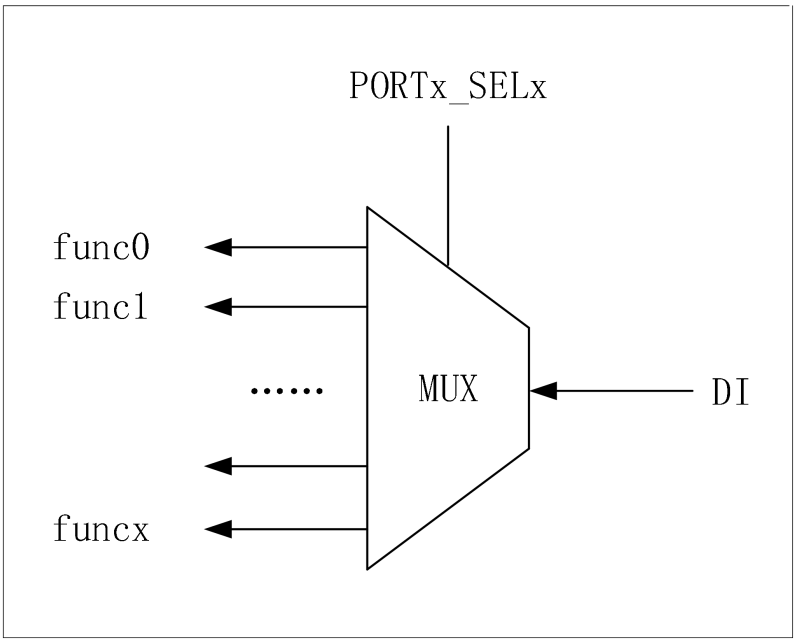


图 5-15 IO 输入功能选择示意图

func0、func1、……、funcx 等为向芯片内部输入的数字信号。

通过 PORTx_SELx 寄存器将某 IO 配置为特定的数字功能，若该功能为输入，则通过 PORTx_SELx 选择将 DI 信号输入给其中一个 func 信号，其它 func 信号为电平 0。通过 PORTx_SELx 可以打通从 DI 到具体数字信号的通路，对应 IO 的 IE 打开后，则实现输入信号从 PAD 端直接进入芯片内部给到选中的数字信号。

- 输出功能描述：

输出功能选择示意图如下图所示：

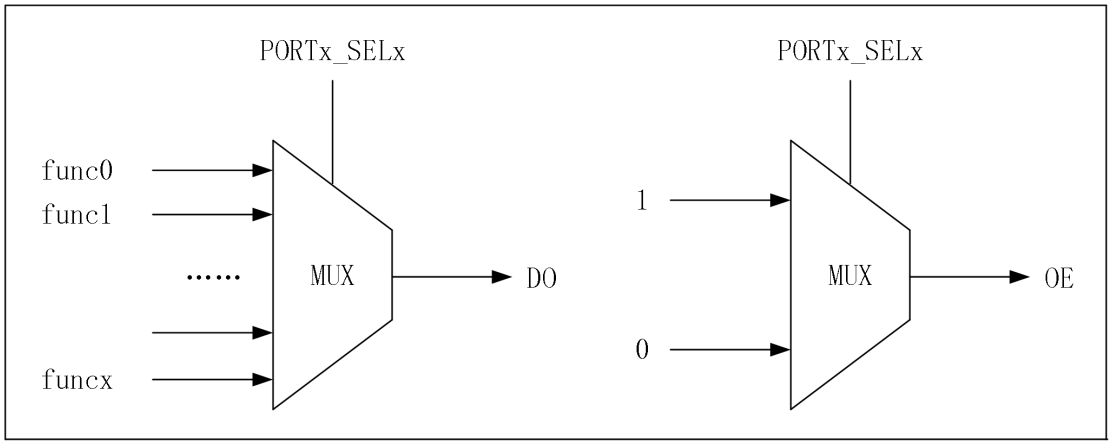


图 5-16 IO 输出功能选择示意图

func0、func1、……、funcx 等为待输出的芯片内部数字信号。

通过 PORTx_SELx 寄存器将某 IO 配置为特定的数字功能，若该功能为输出，则通过 PORTx_SELx 选择将选中的数字信号输出给 DO，其它 func 信号不会选通输出，并且对应 IO 的 OE 端配置为有效。可实现将芯片内部数字信号电平输出至片外的 PAD，供片外获取。

模拟复用功能选择

当通过 PORTx_SELx 寄存器将特定 IO 选择为模拟功能时，IO 的 OE 端会直接被关闭，同时需要将对应的输入使能寄存器（PORTx_IE）配置为 0，将 IO 的 IE 端关闭。另外，为了保证连至 PAD 的模拟信号完整性并不被干扰，软件需要保证该 IO 的上拉电阻和下拉电阻无效（通过关闭相应 IO 的上拉使能（PORTx_PU）和下拉使能（PORTx_PD）来控制。

下面通过一个示例说明模拟信号与 IO 之间的连接及控制关系。

模拟信号连接 IO 的示意图如下所示：

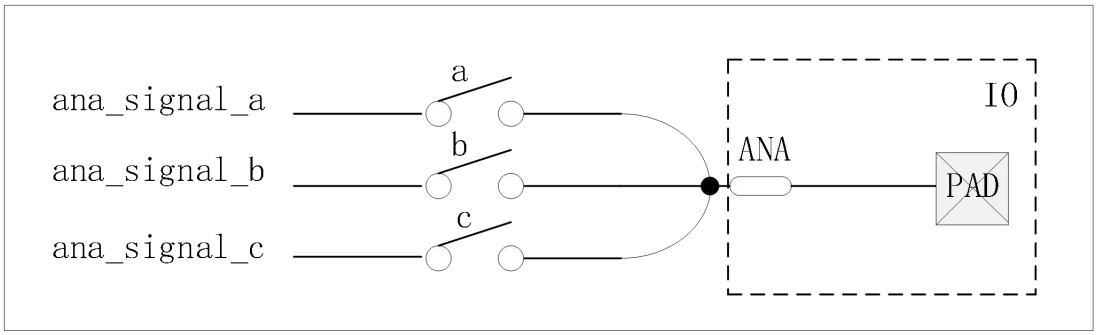


图 5-17 模拟信号连接 IO 示意图

上图中 ana_signal_a、ana_signal_b、ana_signal_c 为复用到同一 IO 上的三个模拟信号，三个模拟信号分别由各自的模拟开关 a、b、c，这些开关在各自不同的模块内控制，并不在该模块内统一控制。当该 IO 不复用模拟信号时，则开关断开，模拟信号不会连通至 ANA 端，从而 PAD 上的电平及行为也不会影响到芯片内部的模拟信号。

当需要将 IO 复用为其中一个模拟功能时，则将相应的开关闭合，使该模拟信号可以连通至 IO 的 ANA 端。如下图所示，IO 复用为 ana_signal_b 模拟信号的功能。

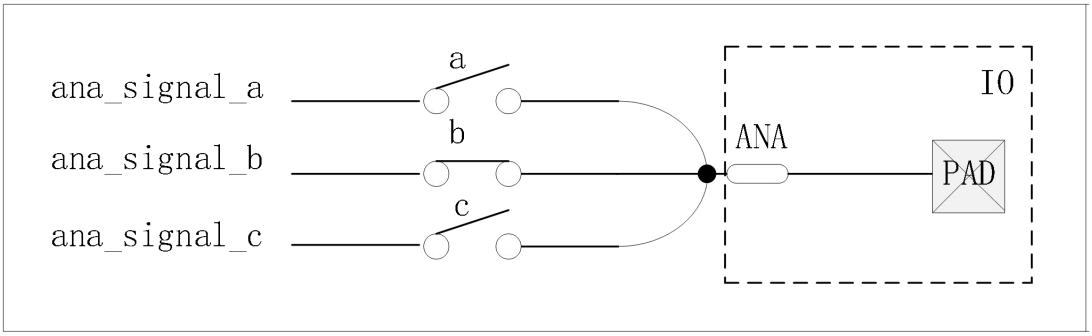


图 5-18 IO 复用为 ana_signal_b 模拟信号的功能示意图

上拉/下电阻使能

本芯片所有 IO 都具有独立输入上拉或输入下拉可配置的功能。

如图 IO 端口基本结构示意图所示，上拉电阻通过 PU 端口控制，下拉电阻通过 PD 端口控制。

当 IO 的 IE 端通过输入使能寄存器（PORTx_IE）配置有效后，通过将相应的上拉电阻使能寄存器（PORTx_PU）配置为 1 后，该 IO 的 PU 端口被置为 1，则即使 PAD 为悬空态，DI 端也为逻辑电平 1。

当 IO 的 IE 端通过输入使能寄存器（PORTx_IE）配置有效后，通过将相应的下拉电阻使能寄存器（PORTx_PD）配置为 1 后，该 IO 的 PD 端口被置为 1，则即使 PAD 为悬空态，DI 端也为逻辑电平 0。

注 0：同一 IO 的上拉使能和下拉使能不能同时配置为有效。本芯片中若同时配置为有效，则下拉电阻起作用，上拉电阻无效。

输出模式选择

本芯片所有 IO 具有独立的输出模式选择控制，可独立配置为输出开漏模式或输出推挽模式。

当 IO 作为输出使用时，OE 为高，可通过配置开漏使能寄存器（PORTx_OD）为 0 或 1，使得 PAD 的输出状态可配置为推挽输出或开漏输出两种模式。

当 IO 的开漏使能寄存器（PORTx_OD）配置为 0，则为推挽输出模式，此时 IO 具有拉电流/灌电流的能力。如下图所示：

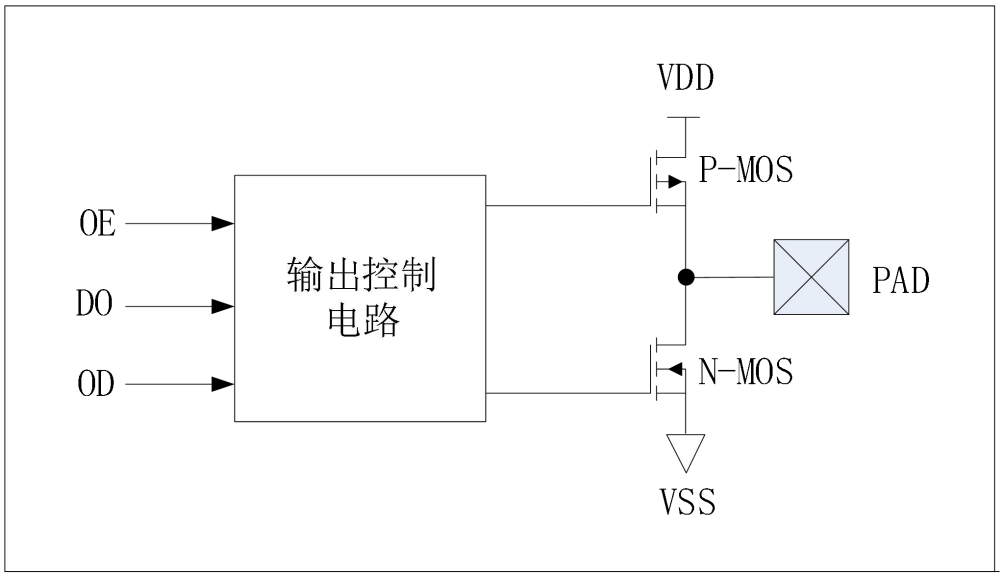


图 5-19　IO 推挽输出模式示意图

当 OD = 0，且 OE = 1 时，当 DO 为 0 时，PAD 为强 0，当 DO 为 1 时，PAD 为强 1。

当 IO 的开漏使能寄存器（PORTx_OD）配置为 1，则为开漏输出模式，此时 IO 只具有灌电流能力，不具有拉电流能力。如下图所示：

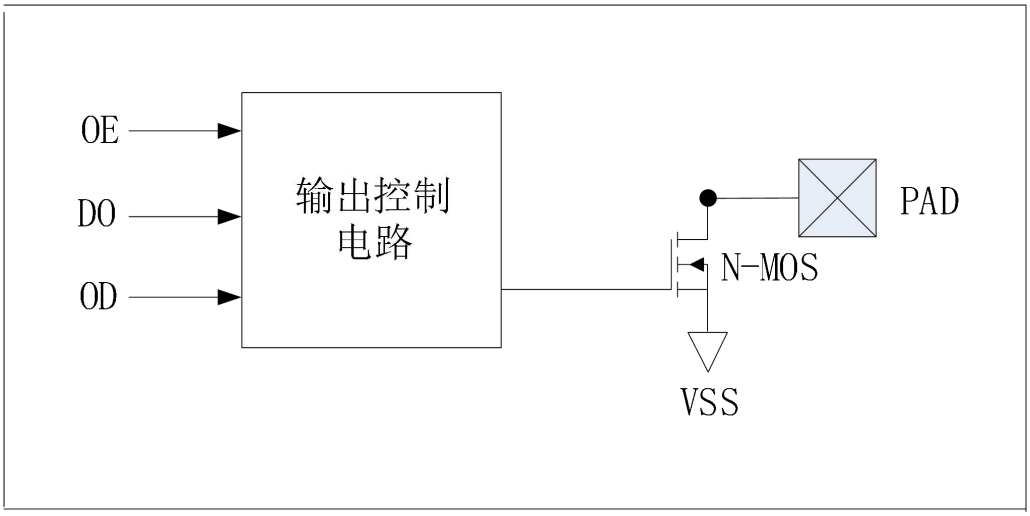


图 5-20 IO 开漏输出模式示意图

当 OD = 1，且 OE = 1 时，当 DO 为 0 时，PAD 为强 0，当 DO 为 1 时，PAD 为高阻态，若需要输出高电平时，需要将外部引脚连接上拉电阻到电源，通过外部上拉实现高电平输出。

引脚唤醒功能

本芯片在低功耗模式下，支持引脚唤醒功能。提供了每个 IO 独立的唤醒控制，用于芯片在使用中提供灵活方便的 IO 唤醒方式。

每个 IO 都可以通过唤醒使能寄存器（PORTx_WKE）独立配置为是否作为唤醒引脚。并且可以支持唤醒沿或电平可配置。

通过 PORT_RIS 寄存器可对所有 IO 独立配置为上升沿唤醒或下降沿唤醒。

注 0：本芯片中，RIS 的定义为当配置为 0 时，为低电平有效；当配置为 1 时，为上升沿有效；

输出驱动能力选择

不同组的 IO 具有独立的输出驱动能力配置控制，可通过各自的 PORT_DS 寄存器来选择。

一般来说 IO 的输出驱动能力可配置为以下四种模式：5mA、14mA、22mA 和 30mA。

注意：该寄存器为 IO 的全局控制寄存器。

输入迟滞选择

本 IO 提供了两种输入迟滞挡位选择控制，用户可根据系统情况及外接信号的电平情况进行合理选择配置。

输入迟滞可通过 PORT_HYS 寄存器控制。当该寄存器配置为 0 时，可选择低输入迟滞配置；当该寄存器配置为 1 时，可选择高输入迟滞配置。

注意：该寄存器为 IO 的全局控制寄存器。

上拉电阻阻值选择

为了给客户提供方便，本 IO 提供了上拉电阻阻值可配置的选项，以便用户可以选择更加合理的上拉电阻。

上拉电阻阻值可通过 PORT_PUR 寄存器配置。有三种阻值可选：32kΩ、40kΩ和 150kΩ。

注意：该寄存器为 IO 的全局控制寄存器。

寄存器映射寄存器映射

名称	偏移量	位宽	类型	复位值	描述
PORT BASE: 0x400B0000					
PORTA_SEL0	0x00	32	R/W	0x00	PORTA 功能选择寄存器 0
PORTA_SEL1	0x04	32	R/W	0x00	PORTA 功能选择寄存器 1
PORTB_SEL0	0x08	32	R/W	0xffff0000	PORTB 功能选择寄存器 0
PORTB_SEL1	0x0C	32	R/W	0x01001000	PORTB 功能选择寄存器 1
PORTC_SEL0	0x10	32	R/W	0x00	PORTC 功能选择寄存器 0
PORTA_IE	0x100	32	R/W	0x00	PORTA 输入使能寄存器
PORTB_IE	0x104	32	R/W	0x4800	PORTB 输入使能寄存器
PORTC_IE	0x108	32	R/W	0x20	PORTC 输入使能寄存器
PORTA_PU	0x200	32	R/W	0x00	PORTA 上拉使能寄存器
PORTB_PU	0x204	32	R/W	0x00	PORTB 上拉使能寄存器

PORTC_PU	0x208	32	R/W	0x00	PORTC 上拉使能寄存器
PORTA_PD	0x300	32	R/W	0x00	PORTA 下拉使能寄存器
PORTB_PD	0x304	32	R/W	0x00	PORTB 下拉使能寄存器
PORTC_PD	0x308	32	R/W	0x20	PORTC 下拉使能寄存器
PORTA_OD	0x400	32	R/W	0x00	PORTA 开漏使能寄存器
PORTB_OD	0x404	32	R/W	0x00	PORTB 开漏使能寄存器
PORTC_OD	0x408	32	R/W	0x00	PORTC 开漏使能寄存器
PORTA_WKE	0x500	32	R/W	0x00	PORTA 唤醒使能寄存器
PORTB_WKE	0x504	32	R/W	0x00	PORTB 唤醒使能寄存器
PORTC_WKE	0x508	32	R/W	0x00	PORTC 唤醒使能寄存器
PORT_CFG	0x600	32	R/W	0x15	PORT 配置寄存器
PORTA_WK_SEL	0x700	32	R/W	0x00	PORTA 唤醒 IO 沿选择寄存器
PORTB_WK_SEL	0x704	32	R/W	0x00	PORTB 唤醒 IO 沿选择寄存器
PORTC_WK_SEL	0x708	32	R/W	0x00	PORTC 唤醒 IO 沿选择寄存器

寄存器描述

PORTA_SEL0 寄存器（0x00）

位域	名称	类型	复位值	描述
31:28	PORTA7	R/W	0	0000：GPIOA7 0001：UART1_TX 0010：TIMERP0_IN0 0011：TIMERP0_OUT_L 0100：SARADC_CH2 0101：OPA0_VP 其它：保留

27:24	PORTA6	R/W	0	0000：GPIOA6 0001：UART1_RTS 0010：TIMERP1_IN1 0011：TIMERP1_OUT_H 0100：SARADC_CH1 0101：OPA0_OUT 其它：保留
23:20	PORTA5	R/W	0	0000：GPIOA5 0001：UART1_CTS 0010：PWMP1_PLUS1 0011：TIMERP1_IN0 0100：TIMERP1_OUT_L 0101：WAKEUP1 0110：SARADC_CH0 其它：保留
19:16	PORTA4	R/W	0	000：GPIOA4 001：CMP0_VP 010：XTAH_XO 其它：保留
15:12	PORTA3	R/W	0	000：GPIOA3 001：CMP0_VN 010：XTAH_XI 其它：保留
11:8	PORTA2	R/W	0	000：GPIOA2 001：XTAL_XO 其它：保留
7:4	PORTA1	R/W	0	000：GPIOA1 001：XTAL_XI 其它：保留

3:0	PORTA0	R/W	0	0000：GPIOA0 0001：PWMP1_PLUS0 0010：PWMP0_PLUS1 0011：TM 0100：WAKEUP0 其它：保留

PORTA_SEL1 寄存器（0x04）

位域	名称	类型	复位值	描述
31:28	PORTA15	R/W	0	0000：GPIOA15 0001：PWMB1_CH1 0010：PWMP0_CH0 0011：TIMERP1_IN1 0100：TIMERP1_OUT_H 其它：保留
27:24	PORTA14	R/W	0	0000：GPIOA14 0001：PWMB1_CH0 0010：PWMP0_CH2N 0011：TIMERP1_IN0 0100：TIMERP1_OUT_L 0101：SARADC_CH9 其它：保留
23:20	PORTA13	R/W	0	0000：GPIOA13 0001：PWMB0_CH2 0010：PWMP0_CH1N 0011：TIMERP0_IN1 0100：TIMERP0_OUT_H 0101：SARADC_CH8 其它：保留

19:16	PORTA12	R/W	0	0000：GPIOA12 0001：SPI0_MOSI 0010：PWMB0_CH1 0011：PWMP0_CH0N 0100：TIMERP0_IN0 0101：TIMERP0_OUT_L 0110：SARADC_CH7 其它：保留
15:12	PORTA11	R/W	0	0000：GPIOA11 0001：SPI0_MISO 0010：PWMB0_CH0 0011：PWMP0_BRAKE0 0100：TIMERP1_IN1 0101：TIMERP1_OUT_H 0110：SARADC_CH6 其它：保留
11:8	PORTA10	R/W	0	0000：GPIOA10 0001：SPI0_CLK 0010：SARADC_CH5 0011：CMP1_VP 其它：保留
7:4	PORTA9	R/W	0	0000：GPIOA9 0001：SPI0_SSN 0010：TIMERP1_IN0 0011：TIMERP1_OUT_L 0100：TM 0101：SARADC_CH4 0110：CMP1_VN 其它：保留

3:0	PORTA8	R/W	0	0000：GPIOA8 0001：UART1_RX 0010：TIMERP0_IN1 0011：TIMERP0_OUT_H 0100：SARADC_CH3 0101：OPA0_VN 其它：保留

PORTB_SEL0 寄存器（0x08）

位域	名称	类型	复位值	描述
31:28	PORTB7	R/W	0xf	0000：GPIOB7 0001：SPI0_SSN 0010：UART0_TX 0011：IIC0_SCL 0100：PWMP1_BRAKE0 0101：PWMP0_CH1 其它：保留
27:24	PORTB6	R/W	0xf	0000：GPIOB6 0001：PWMP0_CH0 0010：TIMERP0_IN1 0011：TIMERP0_OUT_H 其它：保留
23:20	PORTB5	R/W	0xf	0000：GPIOB5 0001：SPI1_MOSI 0010：PWMP1_CH0N 0011：PWMP0_CH2N 0100：TIMERP0_IN0 0101：TIMERP0_OUT_L 其它：保留

19:16	PORTB4	R/W	0xf	0000：GPIOB4 0001：SPI1_MISO 0010：IIC1_SCL 0011：PWMP1_CH0 0100：PWMP0_CH1N 0101：TIMERP1_HALL2 其它：保留
15:12	PORTB3	R/W	0	0000：GPIOB3 0001：SPI1_CLK 0010：IIC1_SDA 0011：PWMP0_CH0N 0100：TIMERP1_HALL1 其它：保留
11:8	PORTB2	R/W	0	0000：GPIOB2 0001：SPI1_SSN 0010：PWMP0_BRAKE1 0011：TIMERP1_HALL0 其它：保留
7:4	PORTB1	R/W	0	0000：GPIOB1 0001：UART2_RX 0010：IIC0_SDA 0011：PWMP0_CH2 其它：保留
3:0	PORTB0	R/W	0	0000：GPIOB0 0001：UART2_TX 0010：IIC0_SCL 0011：PWMB1_CH2 0100：PWMP0_CH1 其它：保留

PORTB_SEL1 寄存器（0x0C）

位域	名称	类型	复位值	描述

31:28	PORTB15	R/W	0	0000：GPIOB15 0001：SPI1_SSN 0010：UART2_RX 其它：保留
27:24	PORTB14	R/W	0x1	0000：GPIOB14 0001：SWCLK 0010：UART2_TX 0011：PWMP1_CH2N 其它：保留
23:20	PORTB13	R/W	0	0000：GPIOB13 0001：UART1_RX 0010：IIC1_SDA 0011：PWMP1_CH1N 其它：保留
19:16	PORTB12	R/W	0	0000：GPIOB12 0001：UART1_TX 0010：IIC1_SCL 0011：PWMP1_CH0N 其它：保留
15:12	PORTB11	R/W	0x1	0000：GPIOB11 0001：SWDIO 0010：PWMP1_CH2 0011：PWMP0_BRAKE2 其它：保留
11:8	PORTB10	R/W	0	0000：GPIOB10 0001：SPI0_MOSI 0010：UART0_RTS 0011：PWMB0_CH2 0100：PWMP1_CH1 0101：PWMP0_PLUS0 0110：TIMERP1_IN0 0111：TIMERP1_OUT_L 其它：保留

7:4	PORTB9	R/W	0	0000：GPIOB9 0001：SPI0_MISO 0010：UART0_CTS 0011：PWMB0_CH1 0100：PWMP1_CH0 0101：TIMERP1_IN1 0110：TIMERP1_OUT_H 其它：保留
3:0	PORTB8	R/W	0	0000：GPIOB8 0001：SPI0_CLK 0010：UART0_RX 0011：IIC0_SDA 0100：PWMB0_CH0 0101：PWMP1_BRAKE1 0110：PWMP0_CH2 其它：保留

PORTC_SEL0 寄存器（0x10）

位域	名称	类型	复位值	描述
31:28	PORTC7	R/W	0	0000：GPIOC7 0001：IIC1_SDA 0010：PWMP1_CH2 0011：TIMERP1_IN0 0100：TIMERP1_OUT_L 0101：OPA1_OUT 其它：保留
27:24	PORTC6	R/W	0	0000：GPIOC6 0001：IIC1_SCL 0010：PWMP1_CH1 0011：TIMERP1_IN1 0100：TIMERP1_OUT_H 0101：OPA1_VN 其它：保留

23:20	PORTC5	R/W	0	0000：GPIOC5 0001：TIMERP0_HALL2 0010：TM 0011：OPA1_VP 其它：保留
19:16	PORTC4	R/W	0	0000：GPIOC4 0001：UART0_RX 0010：IIC0_SDA 0011：PWMP1_CH2N 0100：TIMERP0_HALL1 0101：CMP2_VP 其它：保留
15:12	PORTC3	R/W	0	0000：GPIOC3 0001：UART0_TX 0010：IIC0_SCL 0011：PWMP1_CH1N 0100：TIMERP0_HALL0 0101：CMP2_VN 其它：保留
11:8	PORTC2	R/W	0	0000：GPIOC2 0001：SPI1_MOSI 0010：PWMB1_CH2 0011：PWMP1_BRAKE2 0100：TIMERP0_IN1 0101：TIMERP0_OUT_H 其它：保留
7:4	PORTC1	R/W	0	0000：GPIOC1 0001：SPI1_MISO 0010：UART2_RTS 0011：PWMB1_CH1 0100：TIMERP0_IN0 0101：TIMERP0_OUT_L 其它：保留

3:0	PORTC0	R/W	0	0000：GPIOC0 0001：SPI1_CLK 0010：UART2_CTS 0011：PWMB1_CH0 其它：保留

PORTA_IE 寄存器（0x100）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PORTA_IE	R/W	0x00	PORTA 输入使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 A0,bit1 对应 A1）

PORTB_IE 寄存器（0x104）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PORTB_IE	R/W	0x4800	PORTB 输入使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 B0,bit1 对应 B1）

PORTC_IE 寄存器（0x108）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	PORTC_IE	R/W	0x20	PORTC 输入使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 C0,bit1 对应 C1）

PORTA_PU 寄存器（0x200）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PORTA_PU	R/W	0	PORTA 上拉使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 A0,bit1 对应 A1）

PORTB_PU 寄存器（0x204）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PORTB_PU	R/W	0	PORTB 上拉使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 B0,bit1 对应 B1）

PORTC_PU 寄存器（0x208）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	PORTC_PU	R/W	0	PORTC 上拉使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 C0,bit1 对应 C1）

PORTA_PD 寄存器（0x300）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位

15:0	PORTA_PD	R/W	0	PORTA 下拉使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 A0,bit1 对应 A1）

PORTB_PD 寄存器（0x304）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PORTB_PD	R/W	0	PORTB 下拉使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 B0,bit1 对应 B1）

PORTC_PD 寄存器（0x308）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	PORTC_PD	R/W	0x20	PORTC 下拉使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 C0,bit1 对应 C1）

PORTA_OD 寄存器（0x400）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PORTA_OD	R/W	0	PORTA 开漏使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 A0,bit1 对应 A1）

PORTB_OD 寄存器（0x404）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PORTB_OD	R/W	0	PORTB 开漏使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 B0,bit1 对应 B1）

PORTC_OD 寄存器（0x408）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	PORTC_OD	R/W	0	PORTC 开漏使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 C0,bit1 对应 C1）

PORTA_WKE 寄存器（0x500）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PORTA_WKE	R/W	0	PORTA 唤醒使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 A0,bit1 对应 A1）

PORTB_WKE 寄存器（0x504）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位

15:0	PORTB_WKE	R/W	0	PORTB 唤醒使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 B0,bit1 对应 B1）

PORTC_WKE 寄存器（0x508）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	PORTC_WKE	R/W	0	PORTC 唤醒使能寄存器 0：禁能　1：使能 （每个 bit 对应 1 个 IO，bit0 对应 C0,bit1 对应 C1）

PORT_CFG 寄存器（0x600）

位域	名称	类型	复位值	描述
31:11	RESERVED	R	0	保留位
10	PORT_HYS	R/W	0	PORT 输入迟滞等级选择 0：低输入迟滞（输入信号大于 0.7VDD 和小于 0.3VDD） 1：高输入迟滞（输入信号大于 0.85VDD 和小于 0.15VDD）
9:6	RESERVED	R	0	保留位
5:4	PORTC_DS	R/W	0x01	PORTC 驱动能力选择寄存器 00：5mA 01：10mA 10：15mA 11：20mA
3:2	PORTB_DS	R/W	0x01	PORTB 驱动能力选择寄存器 00：5mA 01：10mA 10：15mA 11：20mA

1:0	PORTA_DS	R/W	0x01	PORTA 驱动能力选择寄存器 00：5mA 01：10mA 10：15mA 11：20mA

PORTA_WK_SEL 寄存器（0x700）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15：0	PORTA_WK_SEL	R/W	0	PORTA 唤醒功能沿配置 0：PORTA 唤醒功能下降沿有效 1：PORTA 唤醒功能上升沿有效

PORTB_WK_SEL 寄存器（0x704）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15：0	PORTB_WK_SEL	R/W	0	PORTB 唤醒功能沿配置 0：PORTB 唤醒功能下降沿有效 1：PORTB 唤醒功能上升沿有效

PORTC_WK_SEL 寄存器（0x708）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7：0	PORTC_WK_SEL	R/W	0	PORTC 唤醒功能沿配置 0：PORTC 唤醒功能下降沿有效 1：PORTC 唤醒功能上升沿有效

5.8通用 IO(GPIO)

5.8.1 概述

GPIO 模块实现了通用的可编程 IO 接口，支持可配置输入输出两种模式，可用于实现串行数据通讯。使用前需使能对应 GPIO 模块的时钟。系统示意图如下：

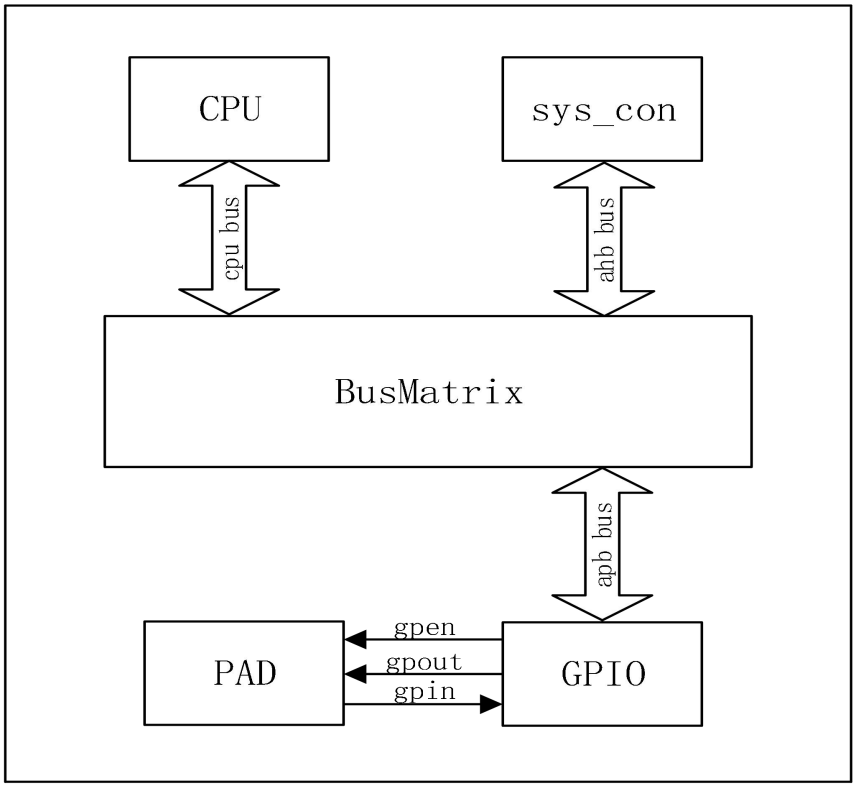


图 5-21 GPIO 模块系统框图

5.8.2 特性

- 最多 40 个独立 IO
- 每个 IO 具有中断入口
- 中断触发条件可配置，支持电平触发和边沿触发
- 电平触发支持高电平和低电平
- 边沿触发支持上升沿、下降沿和双边沿触发

每个 IO 均支持上拉、下拉、推挽、开漏功能

5.8.3 模块结构框图

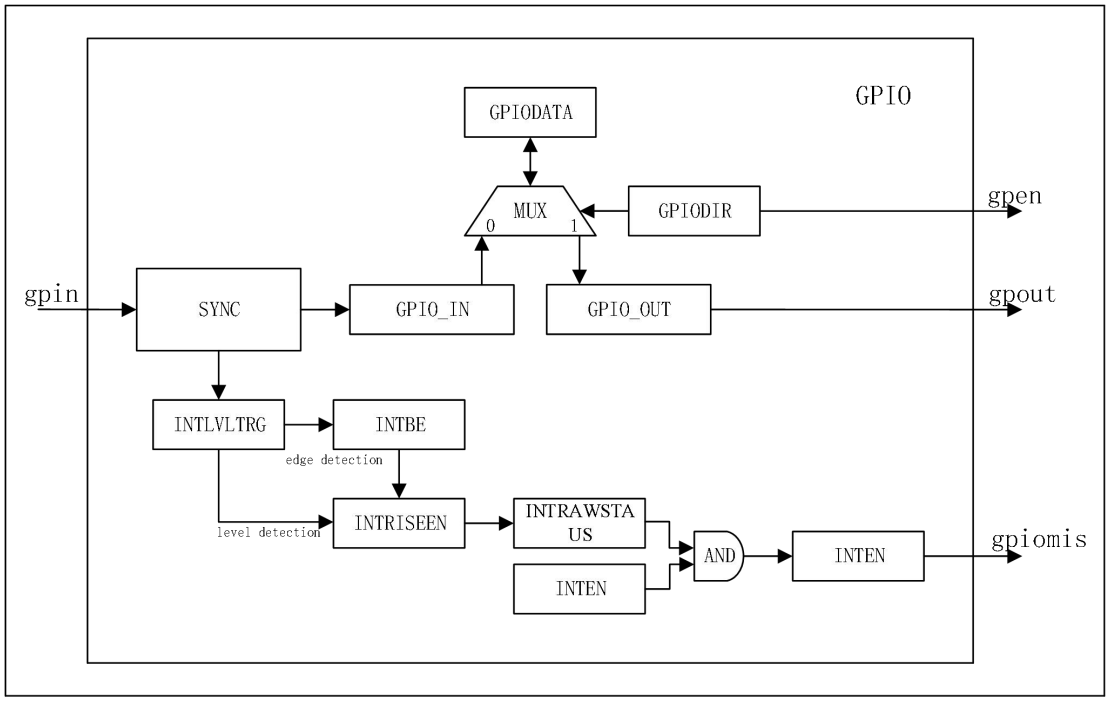


图 5-22 GPIO 模块结构框图

上图为 GPIO 模块内部结构示意图。如上图所示，外部输入数据 gpin 经过同步处理电路 sync 处理，来沿触发中断的检测同时也用于 APB 总线对输入数据的读取。gpin 信号经过不同中断设置产生中断状态 INTRAWSTAUS。INTRAWSTAUS 信号经过中断使能寄存器控制后输出中断标志信号 gpiomis 给系统。本模块中的时钟源为 pclk，与系统时钟同源，可通过 SYSCON 模块中的 DEV_CLK_GATE 寄存器来配置本模块时钟的使能。

5.8.4 功能描述

方向控制

- 除 SWD 引脚外，所有引脚上电后默认状态均为 GPIO 浮空输入（DIR = 0）。
- GPIO 方向寄存器（DIRx）用来将每个独立的管脚配置为输入模式或者输出模式
- 当数据方向设为 0 时，GPIO 对应引脚配置为输入

通过读取相应数据寄存器（DATA）对应位获取指定 GPIO 端口当前状态值

- 当数据方向设为 1 时，GPIO 对应引脚配置为输出

  通过向对应端口数据寄存器（DATA）对应位写入值改变指定引脚输出，0 输出低电平，1 输出高电平。

  GPIO 的输出时序示意图如下所示：

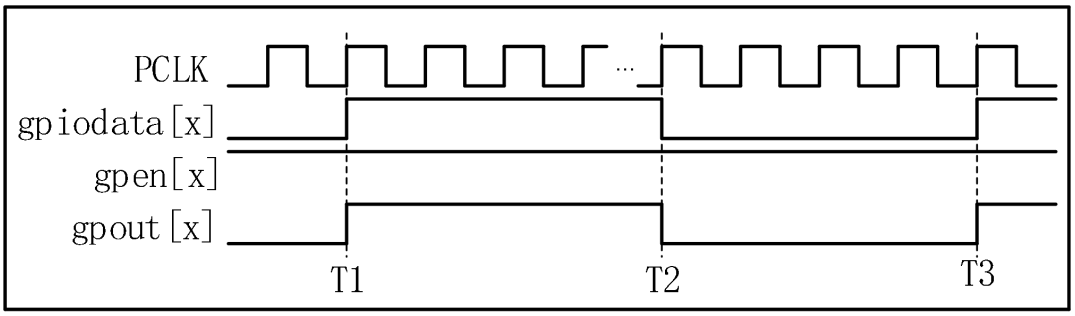


图 5-23 GPIO 的输出时序图

设置 GPIO 模块相应 GPIO 位为输出管脚。

T1 时刻，写入 gpiodata[x]值为 1，GPIO 输出 gpout[x]变为高电平。

T2 时刻，写入 gpiodata[x]值为 0，GPIO 输出 gpout[x]变为低电平。

T3 时刻，写入 gpiodata[x]值为 1，GPIO 输出 gpout[x]变为高电平。

GPIO 输出 gpout[x]随 gpiodata[x]的变化而变化。

中断配置和清除

可根据需求将 GPIO 端口对应引脚配置为中断模式，并通过相关寄存器配置中断极性及触发方式。

触发方式分为边沿触发和电平触发两种模式。

- 对于边沿触发中断，可以设置为上升沿触发，下降沿触发或双边沿触发。中断发生后，标志位具备保持特性，必须通过软件对中断标志位进行清除
- 对于电平触发中断，当外部引脚输入为指定电平时，中断发生。当电平翻转后，中断信号消失，无需软件进行清除。使用电平触发中断，需保证外部信号源保持电平稳定，以便有效中断电平能被端口识别

使用以下寄存器来对产生中断触发方式和极性进行定义：

- GPIO 中断触发方式寄存器（INTLVLTRG），用于配置电平触发或边沿触发

- GPIO 中断触发极性寄存器（INTRISEEN），用于配置电平或边沿触发极性
- GPIO 中断边沿触发配置寄存器（INTBE），选择为边沿触发后，用于配置单边沿触发或双边沿触发

通过 GPIO 中断使能寄存器（INTEN）可以使能或者禁止相应端口对应位中断，GPIO 原始中断状态（INTRAWSTAUS）不受使能位影响。当产生中断时，可以在 GPIO 原始中断状态（RAWINTSTAUS）获取中断信号的状态。当中断使能寄存器（INTEN）对应位为 1 时，中断状态（INTSTAUS）寄存器可读取到对应中断信号，且中断信号会进入中断配置模块及 NVIC 模块，执行中断程序。

通过写 1 到 GPIO 中断清除寄存器（INTCLR）指定位可以清除相应位中断。

1、 高电平触发中断时序

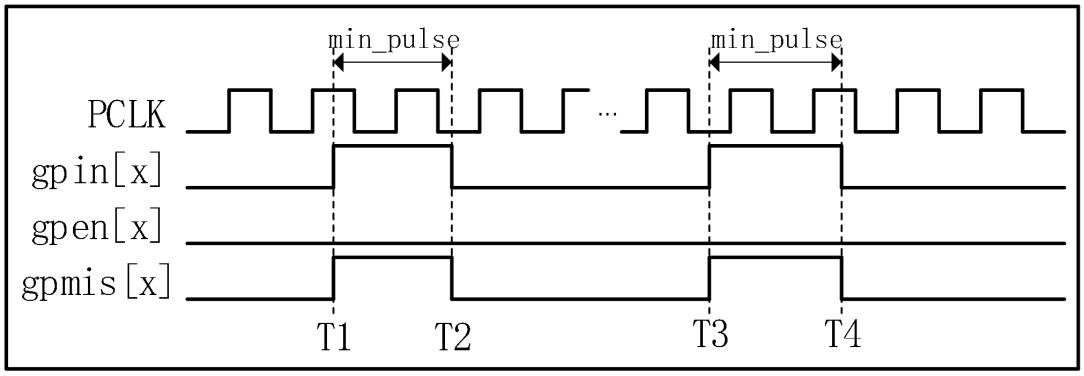


图 5-24 GPIO 高电平中断时序图

图中时间参数说明如下：

表 5-8 GPIO 高电平触发中断时序时间参数

Parameter	描述	min
min_PLUS	中断模块能够处理的中断信号 INTRAWSTAUS 的最小脉冲宽度	>=1pclk

设置 GPIO 模块相应 GPIO 位为输入管脚、高电平检测并且使能中断。当 T1 时刻，gpin[x]输入高电平信号，INTSTAUS 寄存器的 INTSTAUS [x]会被硬件置 1，同时向系统输出 INTSTAUS [x]中断信号。当 T2 时刻，gpin[x]输入低电平信号，INTSTAUS 寄存器的 INTSTAUS [x]被硬件清 0，硬件清除 INTSTAUS [x]中断信号。

需要注意的是，由于中断信号 INTSTAUS 需要由系统中的中断模块进行处理，因此对 gpin 信号的高电平脉冲宽度有最小限制：min_PLUS 宽度至少需要一个 pclk 时钟周期。如果在高电平出现时中断被屏蔽，则该中断会被忽略，不会产生相应的中断标志信号 INTSTAUS。另外，由于 GPIO 模块对高电平触发的中断不锁存，如果由高电平中断事件产生的中断信号在中断模块能够识别之前高电平消失，则该中断标志会被忽略。

2、 低电平触发中断时序

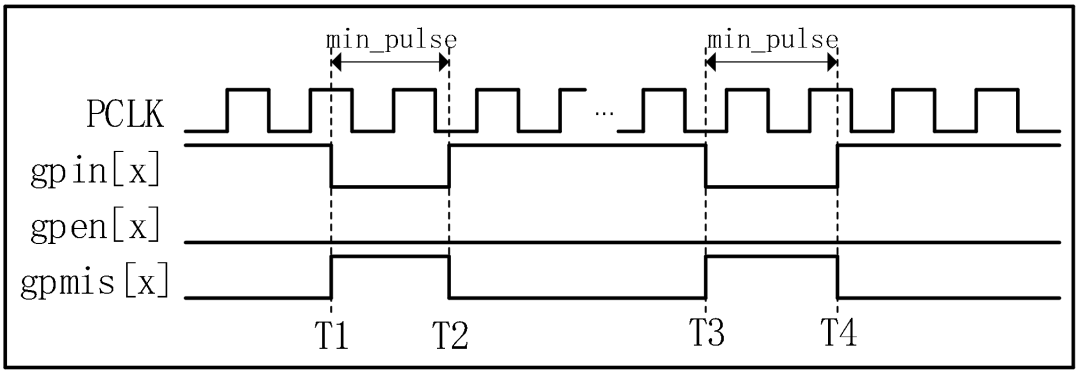


图 5-25 GPIO 低电平触发中断时序图

图中时间参数说明如下：

表 5-9 GPIO 低电平触发中断时序时间参数

Parameter	描述	min
min_PLUS	中断模块能够处理的中断信号 INTRAWSTAUS 的最小脉冲宽度	>=1pclk

设置 GPIO 模块相应 GPIO 位为输入管脚、低电平检测并且使能中断。当 T1 时刻，gpin[x]输入低电平信号，INTSTAUS 寄存器的 INTSTAUS [x]会被硬件置 1，同时向系统输出 INTSTAUS[x]中断信号。当 T2 时刻，gpin[x]输入高电平信号，INTSTAUS 寄存器的 INTSTAUS [x]被硬件清 0，硬件清除 INTSTAUS[x]中断信号。

需要注意的是，由于中断信号 INTSTAUS 需要由系统中的中断模块进行处理，因此对 gpin 信号的高电平脉冲宽度有最小限制：min_PLUS 宽度至少需要一个 pclk 时钟周期。如果在低电平出现时中断被屏蔽，则该中断会被忽略，不会产生相应的中断标志信号 INTSTAUS。另外，由于 GPIO 模块对低电平触发的中断不锁存，如果由低电平中断事件产生的中断信号在中断模块能够识别之前低电平消失，则该中断标志会被忽略。

3、 上升沿触发中断时序

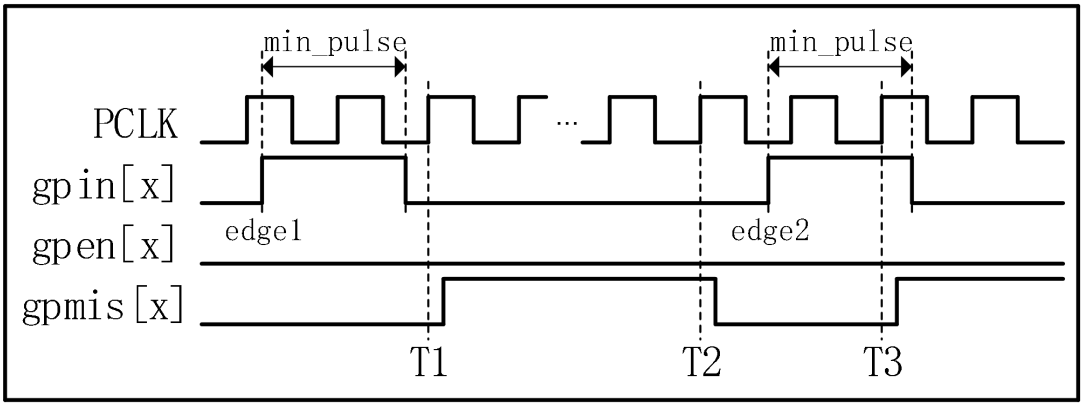


图 5-26 GPIO 上升沿触发中断时序图

图中时间参数说明如下：

表 5-10 GPIO 上升沿触发中断时序时间参数

Parameter	描述	min
min_PLUS	中断模块能够处理的中断信号 INTSTAUS 的最小脉冲宽度	>=1pclk

设置 GPIO 模块相应 GPIO 位为输入管脚、上升沿检测并且使能中断。edge1 时刻，gpin[x]输入一个上升沿，硬件检测上升沿并处理，距离 edge1 上升沿至少 2 个 pclk 周期后，输出 INTSTAUS[x]中断信号，见 T1 时刻。在 T2 时刻系统通过软件方式清除该中断。在 edge2 时刻，gpin[x]又输入一个上升沿，硬件检测上升沿并处理，距离 edge2 上升沿至少 2 个 pclk 周期后，输出由 edge2 时刻的上升沿事件产生的 INTSTAUS[x]中断信号，见 T3 时刻。

需要注意的是，由于 GPIO 模块需要进行上升沿检测，因此对 gpin 信号的高电平脉冲宽度有最小限制：min_PLUS 宽度至少需要一个 pclk 时钟周期，以保证 GPIO 模块能检测到上升沿。如果在上升沿出现时中断被屏蔽，则该中断会被忽略，不会产生相应的中断标志信号 INTSTAUS。另外由于在 T2 时刻才清除当前中断，因此在 T2-2*Tpclk 时刻之前不能出现新的中断事件（上升沿），否则新的中断事件产生的中断标志信号 INTSTAUS 在 T2 时刻也会被同时清除，则新的中断标志信号会被丢失。

4、 下降沿触发中断时序

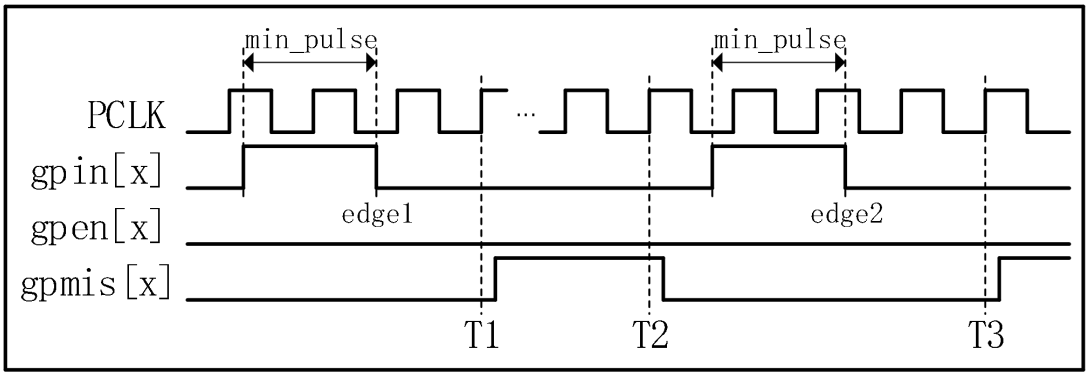


图 5-27 GPIO 下降沿触发中断时序图

图中时间参数说明如下：

表 5-11 GPIO 下降沿触发中断时序时间参数

Parameter	描述	min
min_PLUS	中断模块能够处理的中断信号 INTSTAUS 的最小脉冲宽度	>=1pclk

设置 GPIO 模块相应 GPIO 位为输入管脚、下降沿检测并且使能中断。edge1 时刻，gpin[x]输入一个下降沿，硬件检测下降沿并处理，距离 edge1 上升沿至少 2 个 pclk 周期后，输出 INTSTAUS[x]中断信号，见 T1 时刻。在 T2 时刻系统通过软件方式清除该中断。在 edge2 时刻，gpin[x]又输入一个下降沿，硬件检测下降沿并处理，距离 edge2 上升沿至少 2 个 pclk 周期后，输出由 edge2 时刻的下降沿事件产生的 INTSTAUS[x]中断信号，见 T3 时刻。

需要注意的是，由于 GPIO 模块需要进行下降沿检测，因此对 gpin 信号的低电平脉冲宽度有最小限制：min_PLUS 宽度至少需要一个 pclk 时钟周期，以保证 GPIO 模块能检测到下降沿。如果在下降沿出现时中断被屏蔽，则该中断会被忽略，不会产生相应的中断标志信号 INTSTAUS。另外由于在 T2 时刻才清除当前中断，因此在 T2-2*Tpclk 时刻之前不能出现新的中断事件（下降沿），否则新的中断事件产生的中断标志信号 INTSTAUS 在 T2 时刻也会被同时清除，则新的中断标志信号会被丢失。

5、 双边沿触发中断时序

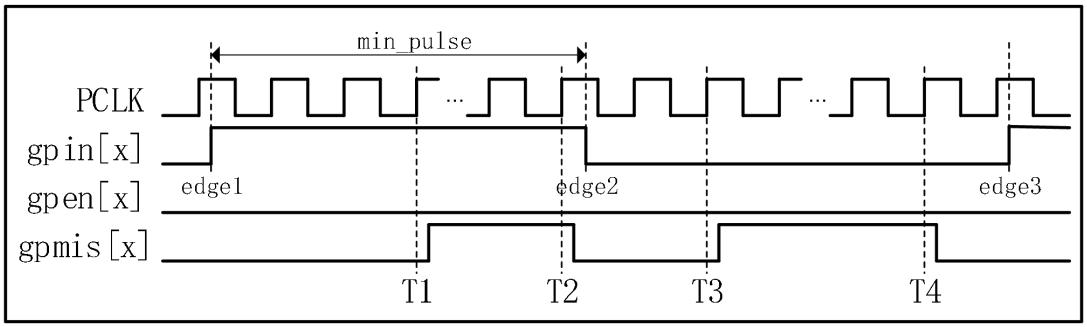


图 5-28 GPIO 双边沿触发中断时序图

设置 GPIO 模块相应 GPIO 位为输入管脚、双沿检测并且使能中断。

edge1 时刻，gpin[x]输入一个上升沿，硬件检测上升沿并处理，距离 edge1 上升沿至少 2 个 pclk 周期后，在 T1 时刻输出 INTSTAUS[x]中断标志信号。系统在 T2 时刻软件清除 edge1 时刻上升沿产生的中断。

edge2 时刻，gpin[x]输入一个下降沿，硬件检测下降沿并处理，距离 edge2 下降沿至少 2 个 pclk 周期后，在 T3 时刻输出 INTSTAUS[x]中断标志信号。系统在 T4 时刻软件清除 edge2 时刻下降沿产生的中断。

在 edge3 时刻，gpin[x]又输入一个上升沿，GPIO 模块对该上升沿以上述同样的方式进行处理并产生中断。

需要注意的是，如果在中断事件（上升沿或下降沿）出现时中断被屏蔽，则该中断会被忽略，不会产生相应的中断标志信号 INTSTAUS。另外，如时序图所示，由于在 T2 时刻才清除 edge1 产生的中断，因此在 T2-2*Tpclk 时刻之前不能出现新的中断事件（下降沿），否则新的中断事件产生的中断标志信号 INTSTAUS 在 T2 时刻也会被同时清除，则新的中断标志信号会被丢失。对于 edge2 事件产生的中断标志信号，在 T4 时刻才被清除，因此在 T4-2*Tpclk 时刻之前不能出现新的中断事件（上升沿），否则新的中断事件产生的中断标志信号 INTSTAUS 在 T4 时刻也会被同时清除，则新的中断标志信号会被丢失。

操作流程

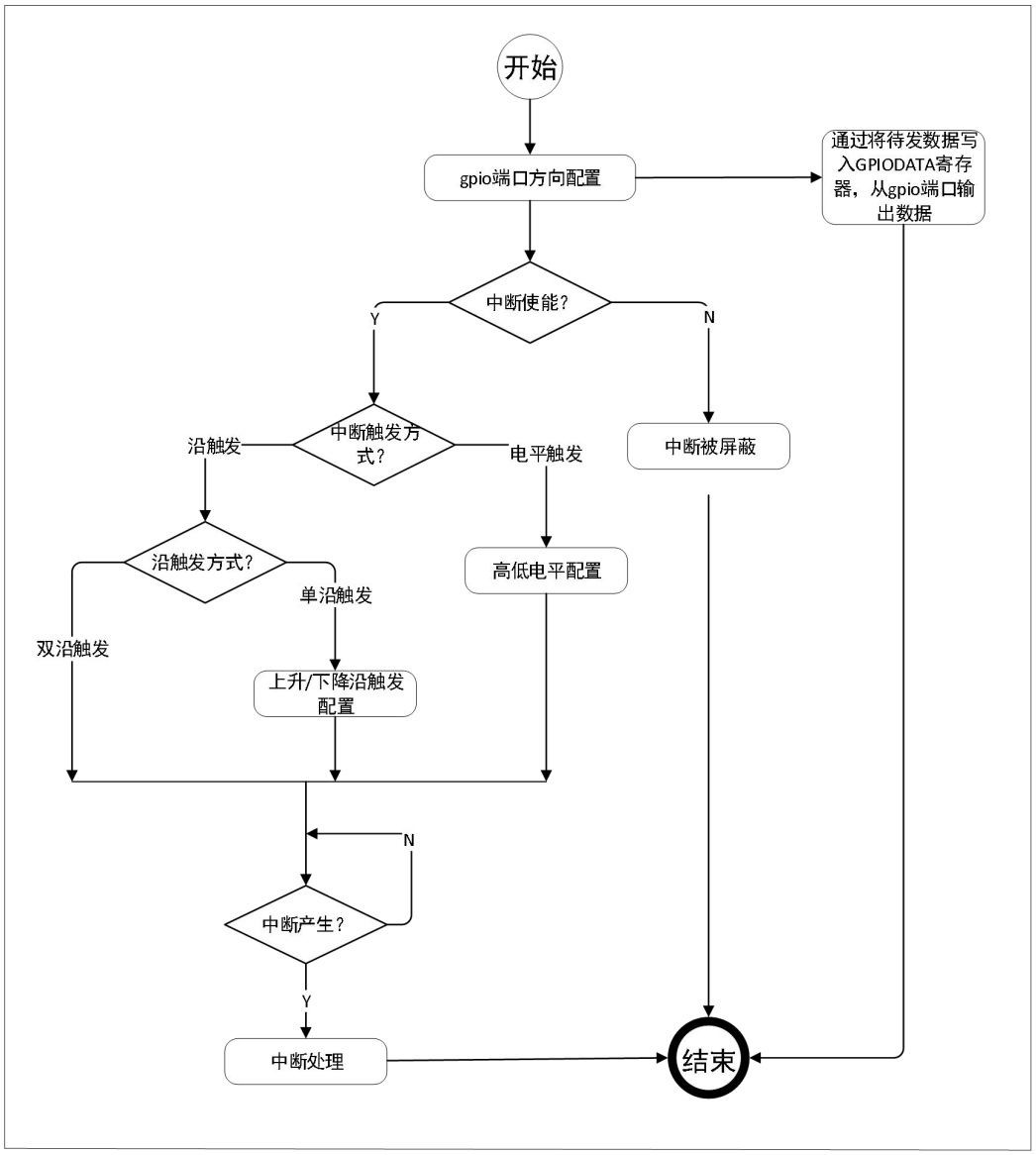


图 5-29 GPIO 操作流程图

- GPIO 模块时钟使能
- PORT 端口配置为 GPIO 功能
- 配置端口方向（GPIODIR）寄存器
- 如果端口配置为输出，则通过将待发数据写入 GPIODATA 寄存器，从 gpio 端口输出数据
- 如果端口配置为输入，则配置中断使能以及中断触发方式，等待中断并处理中断

寄存器映射 GPIODATA 寄存器（0x00）

位域	名称	类型	复位值	描述
31:GPIO_WIDTH	RESERVED	RO	0	保留位
GPIO_WIDTH	GPIODATA	R/W	0	数据寄存器

GPIODIR 寄存器（0x04）

位域	名称	类型	复位值	描述
31:GPIO_WIDTH	RESERVED	RO	0	保留位
GPIO_WIDTH	GPIODIR	R/W	0	设置 GPIO 管脚方向： 1：设置相应位的 GPIO 管脚为输出管脚 0：设置相应位的 GPIO 管脚为输入管脚

INTLVLTRG 寄存器（0x08）

位域	名称	类型	复位值	描述
31:GPIO_WIDTH	RESERVED	RO	0	保留位
GPIO_WIDTH	INTLVLTRG	R/W	0	设置 GPIO 管脚中断敏感条件： 1：设置相应位的 GPIO 管脚为电平检测 0：设置相应位的 GPIO 管脚为沿检测

INTBE 寄存器（0x0C）

位域	名称	类型	复位值	描述
31:GPIO_WIDTH	RESERVED	RO	0	保留位

GPIO_WIDTH	INTBE	R/W	0	设置 GPIO 管脚沿触发方式： 1：设置相应位的 GPIO 管脚为双沿触发中断，即上升沿和下降沿都会触发中断 0：设置相应位的 GPIO 管脚为单沿触发中断，由 INTRISEEN 寄存器相应位确定是上升沿/下降沿触发

INTRISEEN 寄存器（0x10）

位域	名称	类型	复位值	描述
31:GPIO_WIDTH	RESERVED	RO	0	保留位
GPIO_WIDTH	INTRISEEN	R/W	0	设置 GPIO 管脚中断事件方式： 1：设置相应位的 GPIO 管脚为上升沿/高电平触发中断 0：设置相应位的 GPIO 管脚为下降沿/低电平触发中断

INTEN 寄存器（0x14）

位域	名称	类型	复位值	描述
31:GPIO_WIDTH	RESERVED	RO	0	保留位
GPIO_WIDTH	INTEN	R/W	0	设置 GPIO 管脚中断使能： 1：设置相应位的 GPIO 管脚中断使能 0：设置相应位的 GPIO 管脚中断禁止

INTRAWSTAUS 寄存器（0x18）

位域	名称	类型	复位值	描述
31:GPIO_WIDTH	RESERVED	RO	0	保留位

GPIO_WIDTH	INTRAWSTAUS	R	0	当 GPIO 被配置为输入模式时，根据设置的触发条件产生中断标志，不受中断使能寄存器的影响。由硬件置位，软件向 GPIOIC 写 1 清除。 1：表示检测到相应位的 GPIO 中断触发条件(原始，掩码之前) 0：表示没有检测到相应位的 GPIO 中断触发条件

INTSTAUS 寄存器（0x1C）

位域	名称	类型	复位值	描述
31:GPIO_WIDTH	RESERVED	RO	0	保留位
GPIO_WIDTH	INTSTAUS	R	0	当 GPIO 被配置为输入模式，且相应位的中断被使能时，根据设置的触发条件产生中断标志。由硬件置位，软件向 GPIOIC 写 1 清除。 1：表示检测到相应位的 GPIO 管脚产生的中断(掩码之后) 0：表示没有检测到相应位的 GPIO 管脚产生的中断

INTCLR 寄存器（0x20）

位域	名称	类型	复位值	描述
31:GPIO_WIDTH	RESERVED	RO	0	保留位
GPIO_WIDTH	INTCLR	W	0	写 1：清除相应位 GPIO 管脚沿触发中断标志 INTRAWSTAUS 和 INTSTAUS。清除中断标志后，INTCLR 相应位硬件自动恢复为 0。 写 0：没有影响。 对该寄存器进行读操作，返回值为 0。

5.9 基本定时器（TIMERBASE）

5.9.1 概述

基本定时器模块具备定时功能，具有一个 16 位预分频器，支持中断，使用前需要使能定时器模块时钟。基本定时器模块系统框图如下所示：

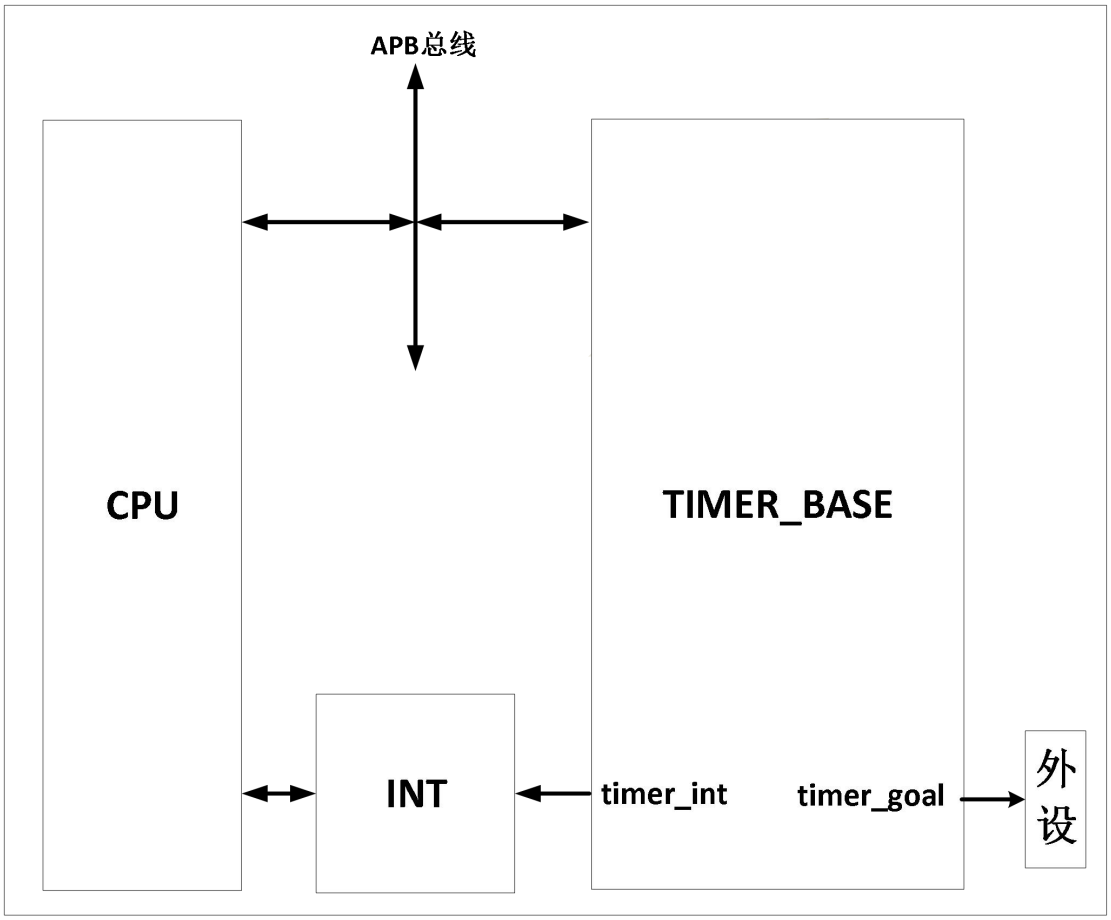


图 5-30　TIMERBASE 模块系统框图

5.9.2 特性

- 支持 2 个独立的向上计数的 16bit 计数器（HIGH、LOW）
- 支持 16bit 预分频

- 可输出中断以及达到目标值标志信号

5.9.3 模块结构框图

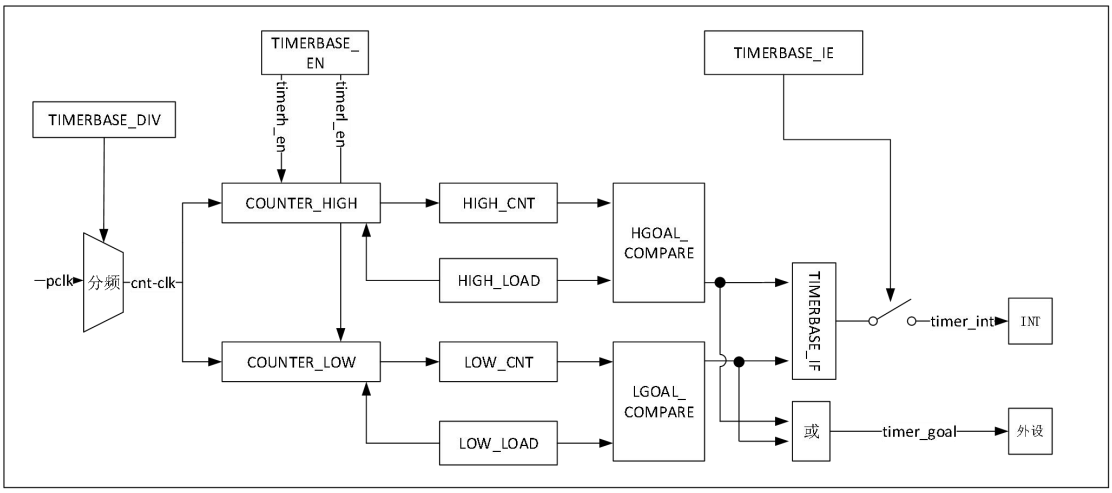


图 5-31 TIMERBASE 模块结构框图

本模块中两个独立的高低计数器共用一个 16bit 分频计数时钟，分频值范围 1-65536，通过分频寄存器 TIMERBASE_DIV 的配置来为计数器提供计数时钟。通过高低定时器目标寄存器 HIGH_LOAD/LOW_LOAD 可以为高低计数器配置计数目标值，然后打开计数使能后计数器开始计数，当计数器的计数值到达目标值时会产生相应的中断状态，并输出计数器目标值标志信号 timer_goal，当中断使能打开时，也会产生相应的中断信号 timer_int。

5.9.4 功能描述

高位计数器计数

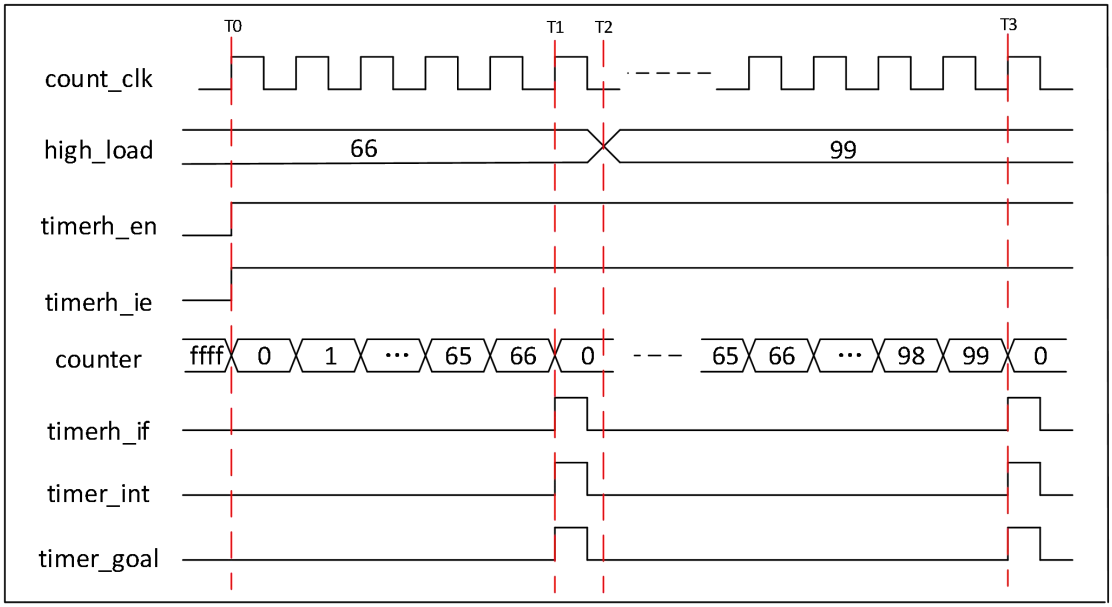


图 5-32 TIMERBASE 高计数器计数时序图

如上图所示为高位计数器计数时序图，当配置计数目标值后，在 T0 时刻打开高位计数器计数使能和中断使能，在 T1 时刻计数器到达目标值后产生高位计数器中断状态，中断信号和达到目标值标志信号。T2 时刻改变目标值，计数器将计数到新的目标值并产生高位计数器中断状态，中断信号和达到目标值标志信号。本模块中的时钟源为 pclk，与系统时钟同源，可通过 SYSCON 模块中的 DEV_CLK_GATE 寄存器来配置本模块时钟的使能。

低位计数器计数

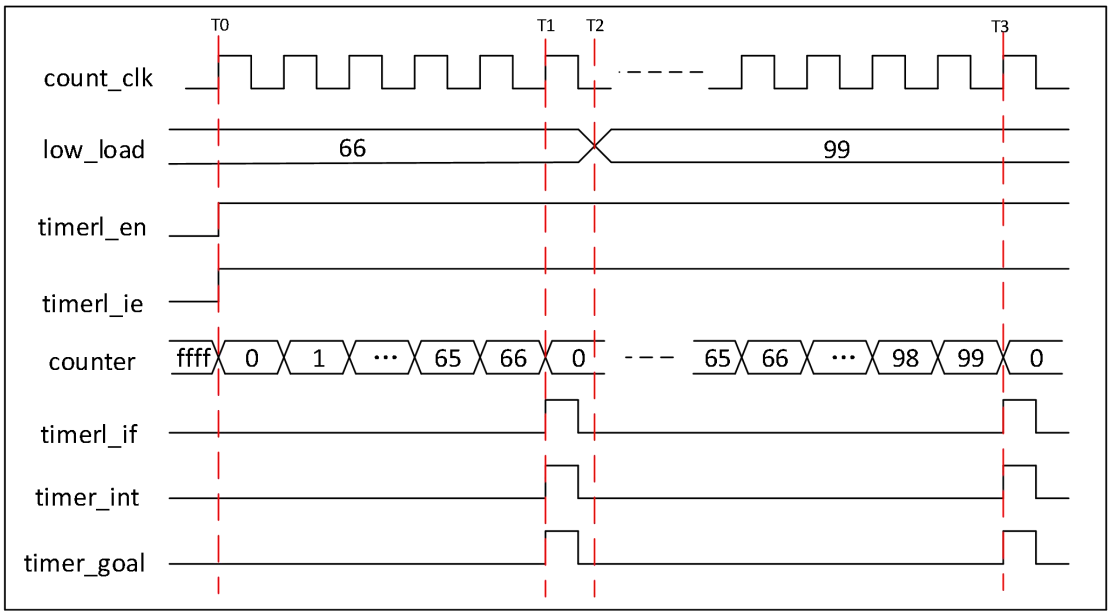


图 5-33　TIMERBASE 低计数器计数时序图

如上图所示为低位计数器计数时序图，当配置计数目标值后，在 T0 时刻打开低位计数器计数使能和中断使能，在 T1 时刻计数器到达目标值后产生低位计数器中断状态，中断信号和达到目标值标志信号。T2 时刻改变目标值，计数器将计数到新的目标值并产生低位计数器中断状态，中断信号和达到目标值标志信号。

分频计数

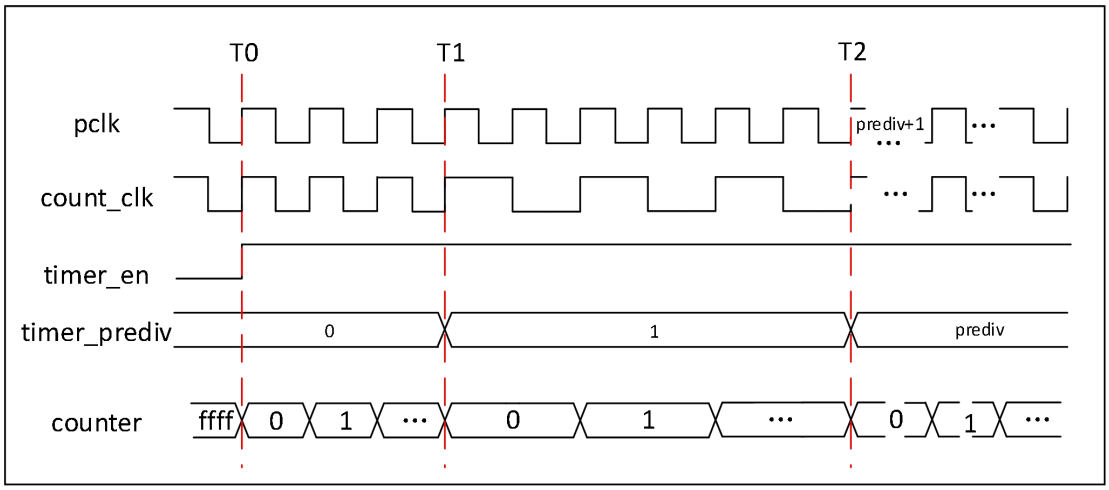


图 5-34 TIMERBASE 计数器分频计数时序图

如上图所示为计数器在配置分频寄存器 TIMERBASE_DIV 分别为 0，1 和 prediv 时的分频计数时序图。

定时时长计算公式

基本定时器作为独立的两个 16 位定时器时，向上计数，计数源为系统时钟 Sys。

定时时长 Tout 计算公式如下：

Tout = (Tpre + 1)*(Tload + 1)/Sys。

注：Tpre 为分频系数，Tload 为装载值的高 16 位或者低 16 位，Sys 为系统时钟。

中断

TIMERBASE 提供了 2 种中断源，它们的关系如下图：

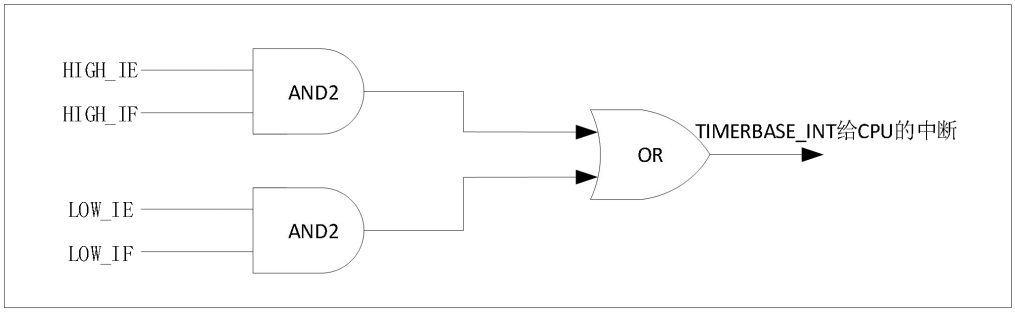


图 5-35　TIMERBASE 中断标志及中断示意图

操作流程

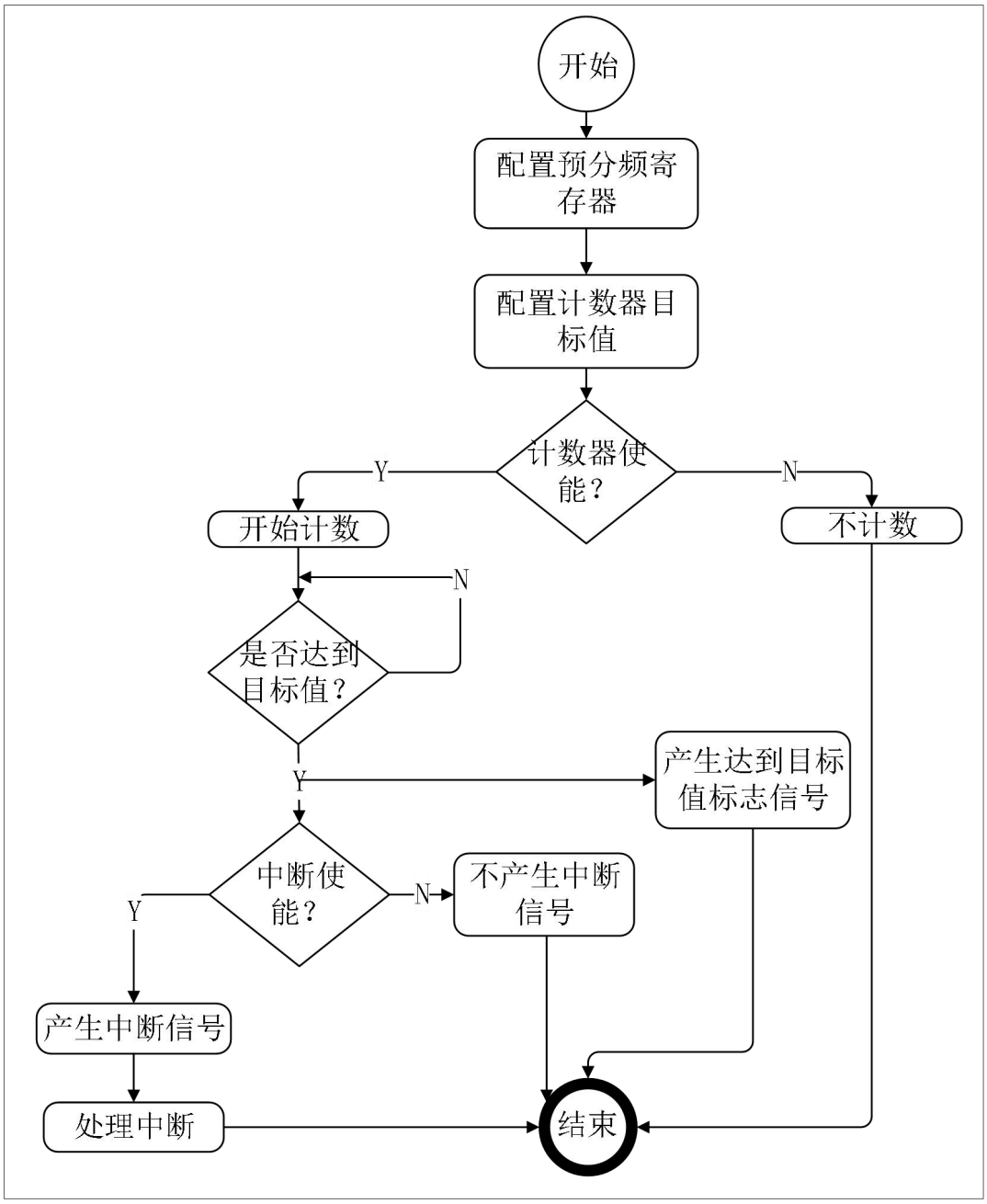


图 5-36 TIMERBASE 操作流程

- 通过预分频寄存器（TIMERBASE_DIV）设置预分频目标值（1-65536）,对系统时钟进行分频。

- 通过装载值寄存器（HIGH_LOAD 或者 LOW_LOAD）设置计数目标值（16bit）。这里分为高 16 位和低 16 位两个定时器，可以根据相应的需求进行配置。
- 通过中断使能寄存器（TIMERBASE_IE）配置中断使能。
- 通过使能寄存器（TIMERBASE_EN）进行相应定时器的使能。
- 对应的定时器开始从 0 向上计数，当计数到装载值时，产生中断，同时重新从 0 开始计数，进入下一个周期的计数。
- 中断可以通过中断状态寄存器（TIMERBASE_IF）进行查询（中断使能的情况下），同时可以对寄存器进行写 1 操作清除中断状态。
- 在计数过程中，可以通过对当前计数值寄存器（HIGH_CNT 或者 LOW_CNT）进行读取，获取当前计数值。

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
TIMERBASE0 TIMERBASE1	BASE: 0x40064000 BASE: 0x40064800				
TIMERBASE_EN	0x00	32	R/W	0x00	TIMER 使能寄存器
TIMERBASE_DIV	0x04	32	R/W	0x00	TIMER 计数时钟分频寄存器
TIMERBASE_IE	0x10	32	R/W	0x00	TIMER 中断使能寄存器
TIMERBASE_IF	0x14	32	R/W	0x00	TIMER 中断状态寄存器
HIGH_LOAD	0x20	32	R/W	0xffff	TIMER HIGH 目标配置寄存器
HIGH_CNT	0x24	32	R	0x00	TIMER HIGH 当前计数值寄存器
LOW_LOAD	0x30	32	R/W	0xffff	TIMER LOW 目标配置寄存器
LOW_CNT	0x34	32	R	0x00	TIMER LOW 当前计数值寄存器

寄存器描述

TIMERBASE_EN 寄存器（0x00）

位域	名称	类型	复位值	描述
31:2	RESERVED	R	0	保留
1	HIGH_EN	R/W	0	TIMERBASE HIGH 定时器使能寄存器
0	LOW_EN	R/W	0	TIMERBASE LOW 定时器使能寄存器

TIMERBASE_DIV 寄存器（0x04）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留
15:0	DIV	R/W	0	TIMERBASE 计数时钟预分频寄存器 0x0000：表示 1 分频 0x0001：表示 2 分频 …… 0xFFFF：表示 65536 分频

TIMERBASE_IE 寄存器（0x10）

位域	名称	类型	复位值	描述
31:2	RESERVED	R	0	保留位
1	HIGH_IE	R/W	0	TIMERBASE HIGH 定时器中断使能
0	LOW_IE	R/W	0	TIMERBASE LOW 定时器中断使能

TIMERBASE_IF 寄存器（0x14）

位域	名称	类型	复位值	描述
31:2	RESERVED	R	0	保留位
1	HIGH_IF	R/W	0	TIMERBASE HIGH 定时器中断状态 写 1 清零
0	LOW_IF	R/W	0	TIMERBASE LOW 定时器中断状态 写 1 清零

HIGH_LOAD 寄存器（0x20）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	HIGH_LOAD	R/W	0xffff	TIMERBASE HIGH 定时器目标配置寄存器 当高 16bit 计数器向上计数达到该设定值后，会产生相应状态信号

HIGH_CNT 寄存器（0x24）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	HIGH_CNT	R	0	TIMERBASE HIGH 定时器当前计数值

LOW_LOAD 寄存器（0x30）

位域	名称	类型	复位值	描述

31:16	RESERVED	R	0	保留位
15:0	LOW_LOAD	R/W	0xffff	TIMERBASE LOW 定时器目标配置寄存器 当低 16bit 计数器向上计数达到该设定值后，会产生相应状态信号

LOW_CNT 寄存器（0x34）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	LOW_CNT	R	0	TIMERBASE LOW 定时器当前计数值

5.10高级定时器（TIMERPLUS）

5.10.1概述

高级定时器模块具备定时、计数、捕获、周期脉冲输出等功能，具有一个 16 位预分频器，支持中断，2 个独立的 16 位的定时器（HIGH 和 LOW），其中低 16 位定时器还支持 HALL 功能，使用前需要使能高级定时器模块时钟。

高级定时器模块的系统框图如下图所示：

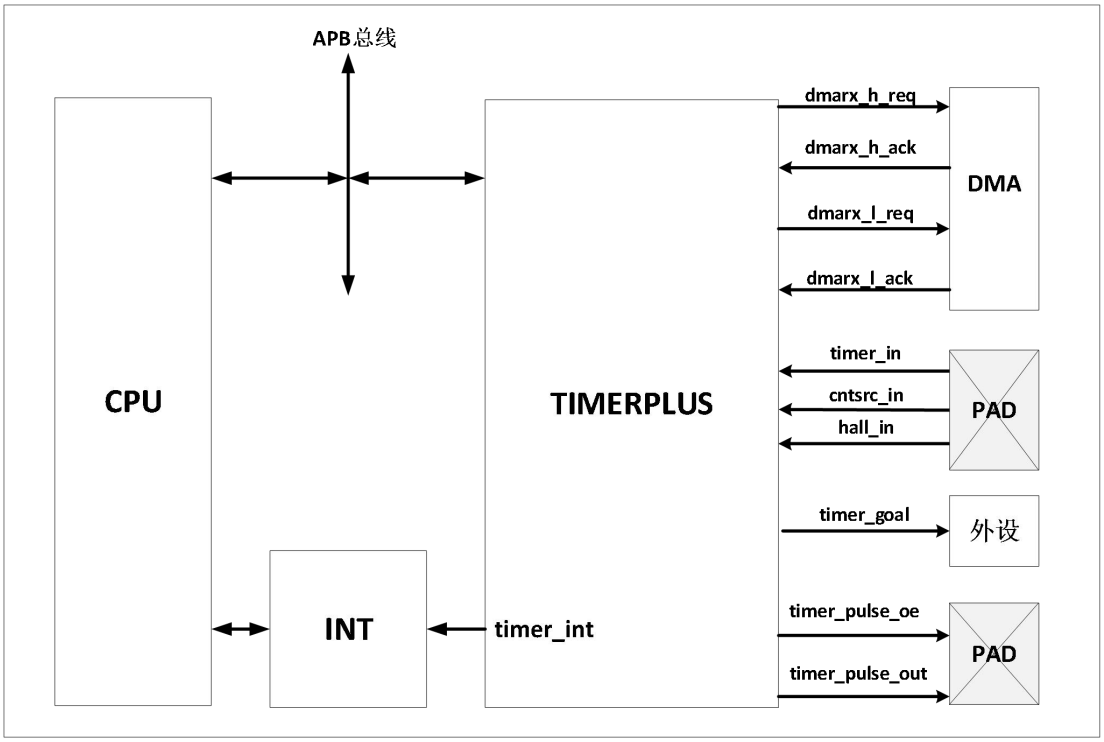


图 5-37 TIMERPLUS 系统框图

如上图所示，本芯片中 TIMERPLUS 模块通过 APB 总线与 CPU 完成通信，并支持与 DMA 的通信，可产生多种中断信号，还可以将周期脉冲输出信号，周期脉冲信号输出使能输出给 PAD 端口，将计数器计数达到目标值标志信号传输给其他外设；同时，TIMERPLUS 还可以通过 PAD 的端口输入片外 timer_in，cntsrc_in，hall_in 等信号。

5.10.2特性

- 2 个独立的 16bit 计数器（HIGH 和 LOW）
- 具有 16bit 预分频计数器
- 两个定时器分别具有定时、计数、输入捕获和周期脉冲输出功能
- LOW 计数器支持 HALL 功能
- 分别支持计数时钟选择
- 支持多种中断功能
- 两个定时器具有各自的目标值影子寄存器，新的目标值将下周期生效
- 支持 DMA 接口读取输入脉冲捕获值

5.10.3模块结构框图

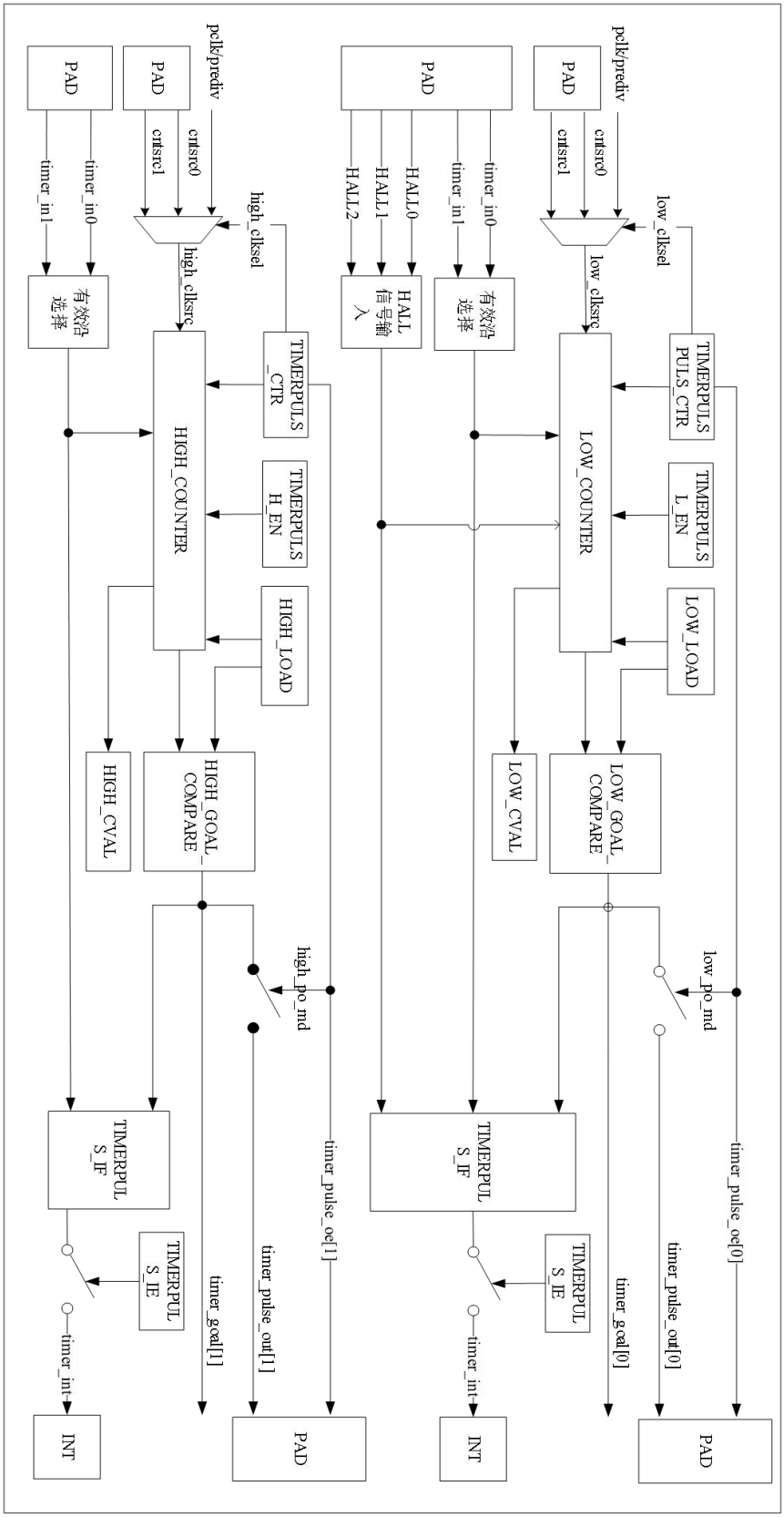


图 5-38 TIMERPLUS 结构框图

上图为 TIMERPLUS 模块的结构图，本模块包含两个独立的 16bit 计数器，两个计数器的计数时钟都可以选择内部预分频时钟或片外计数时钟 cntsrc，其中内部预分频时钟可以通过 pclk 时钟配置预分频寄存器 TIMERPLUS_PREDIV 的 PCLK_PREDIV 位进行分频，分频值范围 1-65536，片外计数时钟 cntsrc 是通过 PAD 的端口输入的。本模块的时钟源为 pclk，与系统时钟同源，可通过 SYSCON 模块中的 DEV_CLK_GATE 寄存器来配置本模块时钟的使能。

高位计数器和低位计数器都支持定时功能，计数功能和输入捕获功能，低位计数器还支持 HALL 功能。高低计数器的功能以及相关配置可以通过寄存器 TIMERPLUS_CTR 进行配置。

本模块中不同状态的中断信号受中断使能寄存器 TIMERPLUS_IE 的控制，只有在相应的中断使能打开的情况下，相应的中断状态产生后才会产生中断信号。

本模块还支持 DMA 对输入捕获的读取，在输入捕获模式下，当 DMA 读取使能为 1 时，DMA 将代替 CPU 从计数器捕获值寄存器中读取捕获值。

本模块产生的 timer_goal[1:0]两个信号可作为 SARADC 模块的触发源。

5.10.4功能描述

计数时钟源选择与预分频

本模块中内部两个定时器都可以分别选择计数时钟源，并且共用一个预分频器。计数器计数时钟源选择示意图如下，其中片外时钟 cntsrc0 和 cntsrc1 是通过 PAD 端口输入的，它与片外输入信号 timer_in 共用相同的 PAD 端口，在 IO 功能配置中选择 PAD 端口为 timer_in 时，PAD 端口也可以作为外时钟 cntsrc0 和 cntsrc1 的输入端口。

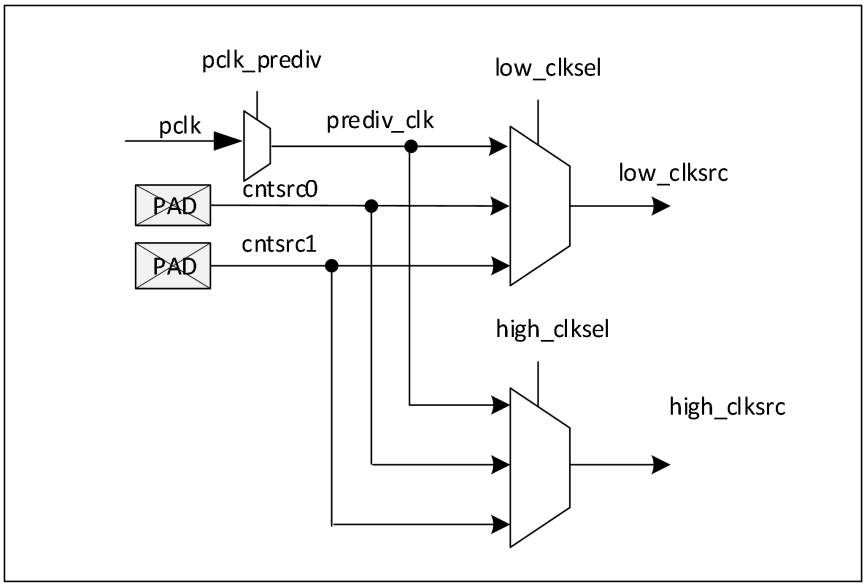


图 5-39　TIMERPLUS 时钟源选择示意图

预分频只用于 pclk，预分频器在预分频寄存器 PCLK_PREDIV 位的控制下可将 pclk 时钟进行分频，分频值范围为 1-65536，分频后的时钟和片外时钟 cntsrc0 和 cntsrc1 通过计数器时钟寄存器的选择为计数器提供计数时钟。计数器在 pclk 分频时钟下的时序示意图如下：

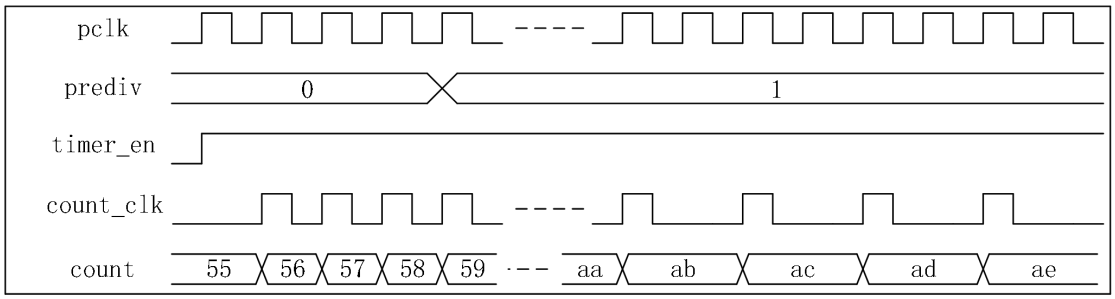


图 5-40　TIMERPLUS pclk 分频时钟下计数器时序图

定时模式

本模块定时模式是指在计数时钟可选择为内部预分频时钟、片外 cntsrc[0]和片外 cntsrc[1]三种时钟源情况下，计数器进行计数，并通过计数器目标值配置寄存器配置定时时间，当计

数到目标值后产生计数器达到目标值标志信号。定时模式下可以配置周期脉冲输出到引脚上，可以作为简单的方波输出。在定时器工作过程中，修改周期值，不会立即生效，而是在周期结束后下一周期生效。

高级定时器作为独立的两个 16 位定时器时，向上计数，以计数时钟源为系统时钟 pclk 为例，定时时长 Tout 计算公式如下：

Tout = (Tpre + 1)*(Tload + 1)/pclk。

注：Tpre 为分频系数，Tload 为装载值的高 16 位或者低 16 位，pclk 为系统时钟。

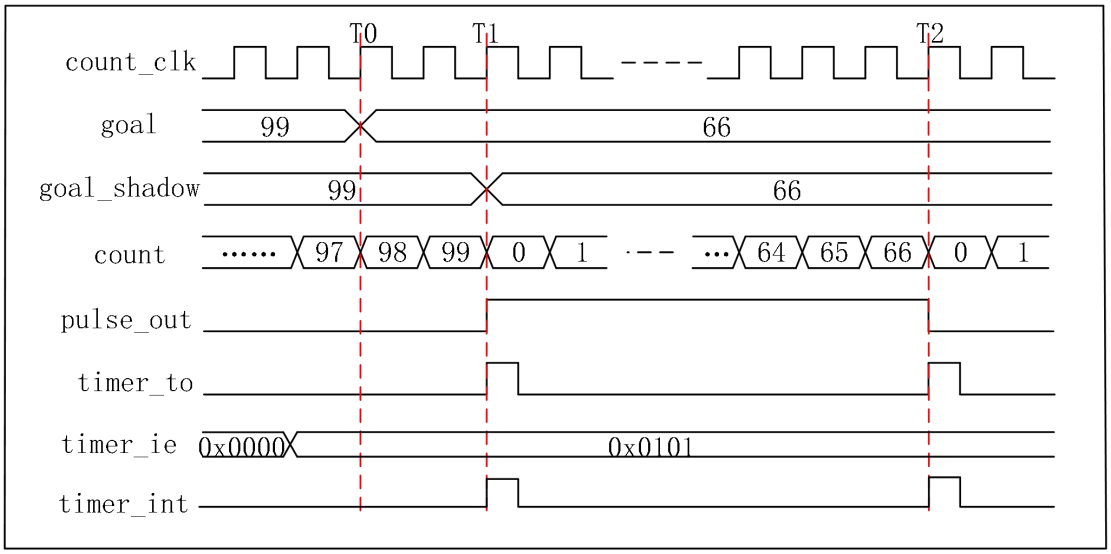


图 5-41 TIMERPLUS 定时模式下计数器时序图

其中，count_clk 为计数器的计数时钟，goal 为 CPU 配置的目标值，goal_shadow 为影子寄存器值，count 为计数值，PLUS_out 为周期脉冲输出的原值，timer_to 为计数值达到目标值时的标志信号。在 T0 时刻改变目标值，影子寄存器的目标值并没有立刻改变，而是在计数器到达之前的目标值之后才改变（T1 时刻），T1 时刻到达目标值之后会使周期脉冲输出信号翻转，产生达到目标值的标志信号，计数器重新从 0 开始计数。T2 时刻计数器再次到达新的目标值后，会再次产生达到目标值的标志信号，周期脉冲输出信号再次翻转，计数器再次重新计数，并且在达到目标值中断使能打开的情况下 T1，T2 会产生达到目标值中断信号。TIMERPLUS_CTR 寄存器的高低计数器周期脉冲输出使能位 HIGH_PO_MD，LOW_PO_MD 分别

控制周期脉冲信号不同 bit 位对 PAD 的输出，当使能打开时，周期脉冲信号才会输出到相应的 PAD 端口。

计数模式

本模块计数模式是指对片外 timer_in[0]或片外 timer_in[1]信号的上升沿，下降沿或双边沿进行计数，当计数到目标值后产生计数器达到目标值标志信号。通过 pclk 检测上升沿，下降沿或双边沿来进行计数器计数，具体时序图如下所示：

上升沿有效

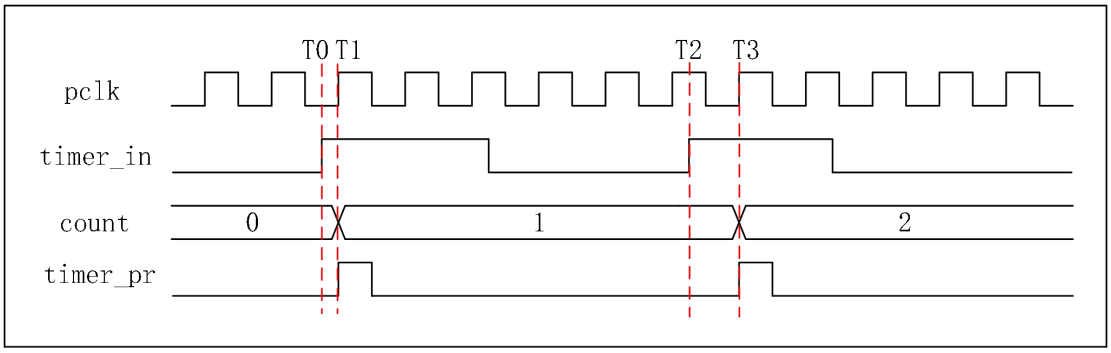


图 5-42　TIMERPLUS 计数模式下上升沿有效时序图

其中计数模式下采用 pclk 为计数器的计数时钟，timer_in 为片外输入信号，count 为计数值，timer_pr 为检测到 timer_in 上升沿标志信号。在 T0 时刻输入信号 timer_in 产生上升沿，在计数时钟的下一个上升沿 T1 时刻计数器加一，并产生上升沿标志信号，T2 和 T3 时刻亦是如此。

下降沿有效

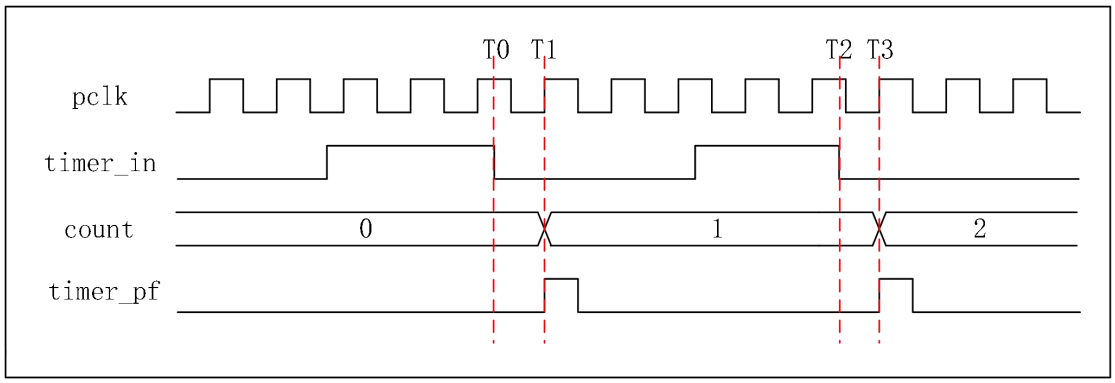


图 5-43　TIMERPLUS 计数模式下下降沿有效时序图

其中 pclk 为计数器的计数时钟，timer_in 为片外输入信号，count 为计数值，timer_pf 为检测到 timer_in 下降沿标志信号。在 T0 时刻输入信号 timer_in 产生下降沿，在计数时钟的下一个上升沿 T1 时刻计数器加一，并产生下降沿标志信号，T2 和 T3 时刻亦是如此。

双沿有效

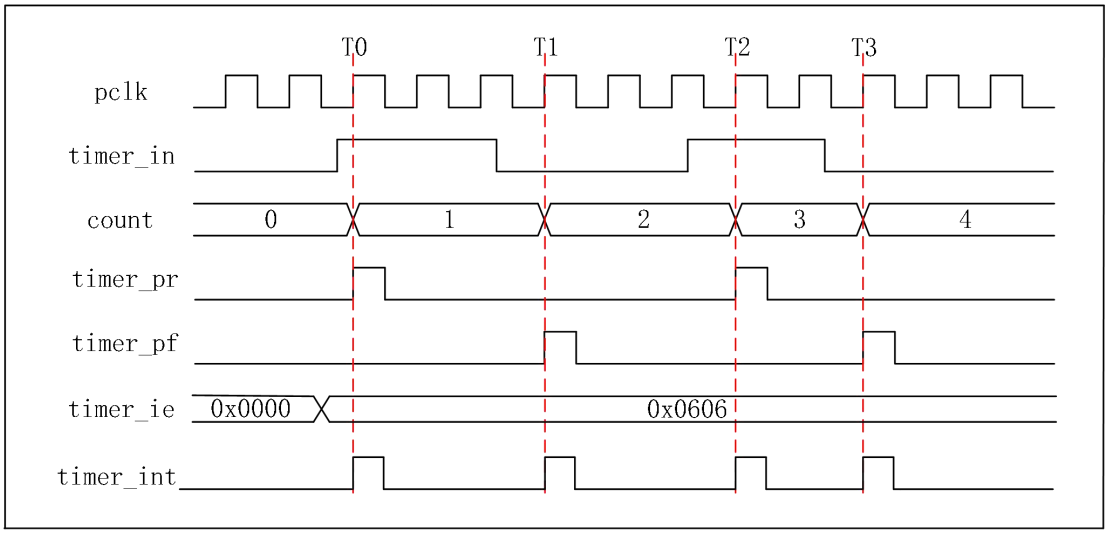


图 5-44　TIMERPLUS 计数模式下双沿有效时序图

其中 pclk 为计数器的计数时钟，timer_in 为片外输入信号，count 为计数值，timer_pr 为检测到 timer_in 上升沿标志信号，timer_pf 为检测到 timer_in 下降沿标志信号。双沿有效的情况与上升沿和下降沿有效的情况下计数器计数一样，即有效沿到来后，计数器的计数将在计数时钟的下一个上升沿有效，当相应的中断使能打开以后，会产生相应的中断信号，如上图中的 T0，T1，T2，T3 时刻。

输入捕获模式

输入捕获模式是指计数时钟可选择为内部预分频时钟、片外 cntsrc[0]和片外 cntsrc[1]三种时钟源情况下，计数器进行计数。当检测到片外 timer_in[0]或片外 timer_in[1]信号的上升沿或下降沿时通过寄存器 cval 保存当前计数器计数值。

不同有效沿时序图如下所示：

上升沿有效

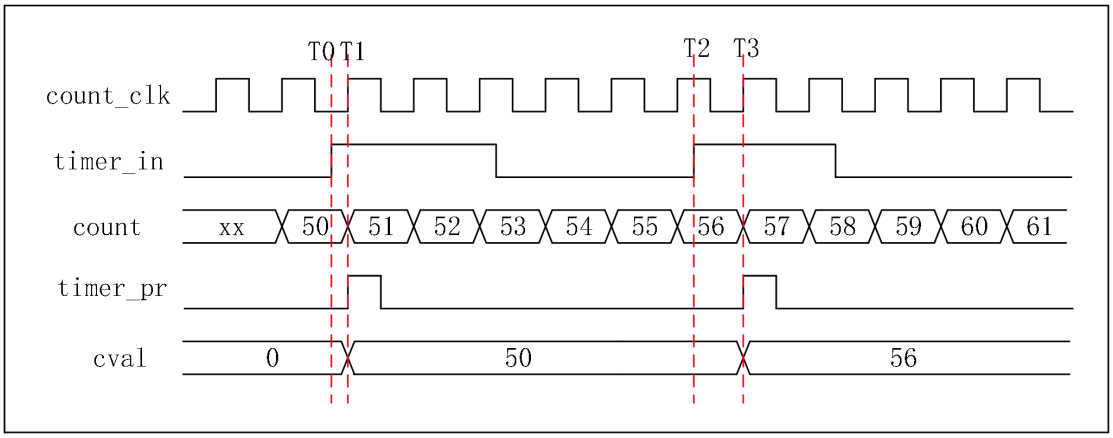


图 5-45　TIMERPLUS 输入捕获下上升沿有效时序图

其中 count_clk 为计数器的计数时钟，timer_in 为片外输入信号，count 为计数值，timer_pr 为检测到 timer_in 上升沿标志信号，cval 为捕获的计数器计数值。在上升沿有效情况下，T0 时刻片外输入信号 timer_in 的上升沿到来，T1 时刻将计数器的当前计数值保存到寄存器 CVAL，并产生相应的检测到上升沿标志信号。T2 时刻片外输入信号 timer_in 的上升沿再次到来，在 T3 时刻计数器的值将再次保存到寄存器 CVAL。

下降沿有效

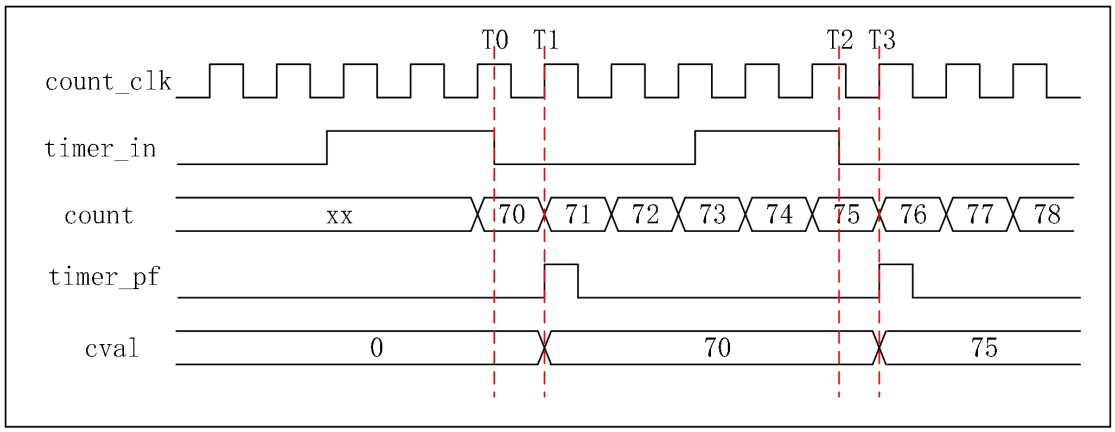


图 5-46 TIMERPLUS 输入捕获下下降沿有效时序图

其中 count_clk 为计数器的计数时钟，timer_in 为片外输入信号，count 为计数值，timer_pf 为检测到 timer_in 下降沿标志信号，cval 为捕获的计数器计数值。在下降沿有效情况下，T0 时刻片外输入信号 timer_in 的下降沿到来，T1 时刻将计数器的当前计数值保存到寄存器 cval，产生相应的检测到下降沿标志信号。T2 时刻片外输入信号 timer_in 的下降沿再次到来，在 T3 时刻计数器的值将再次保存到寄存器 cval。

双沿有效

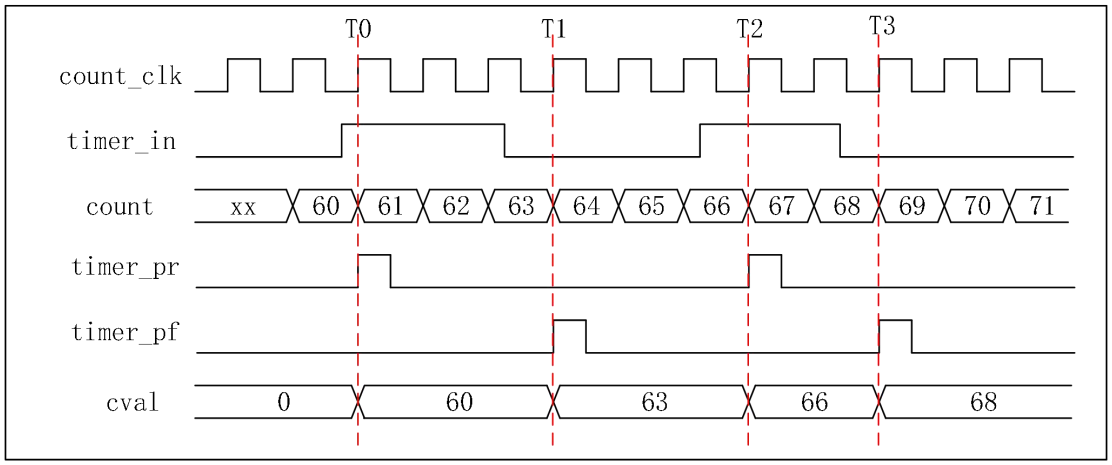


图 5-47 TIMERPLUS 输入捕获下双沿有效时序图

其中 count_clk 为计数器的计数时钟，timer_in 为片外输入信号，count 为计数值，timer_pr 为检测到 timer_in 上升沿标志信号，timer_pf 为检测到 timer_in 下降沿标志信号，cval 为捕获的计数器计数值。在双沿有效的情况下，当上升沿或下降沿到来后，会将计数器当前计数值保存到寄存器 cval，并产生相应的标志信号，如图 T0，T1，T2，T3 时刻所示。

HALL 模式

本模块中，只有 LOW 定时器具有 HALL 模式功能，可用于霍尔信号采集。该模式是指计数时钟可选择为内部预分频时钟、片外 cntsrc[0]和片外 cntsrc[1]三种时钟源情况下，计数器进行计数。当检测到片外 hall_in[0]、片外 hall_in[1]和片外 hall_in[2]信号的上升沿和下降沿时通过寄存器 cval 保存当前计数器计数值，并且将 LOW 计数器清零。

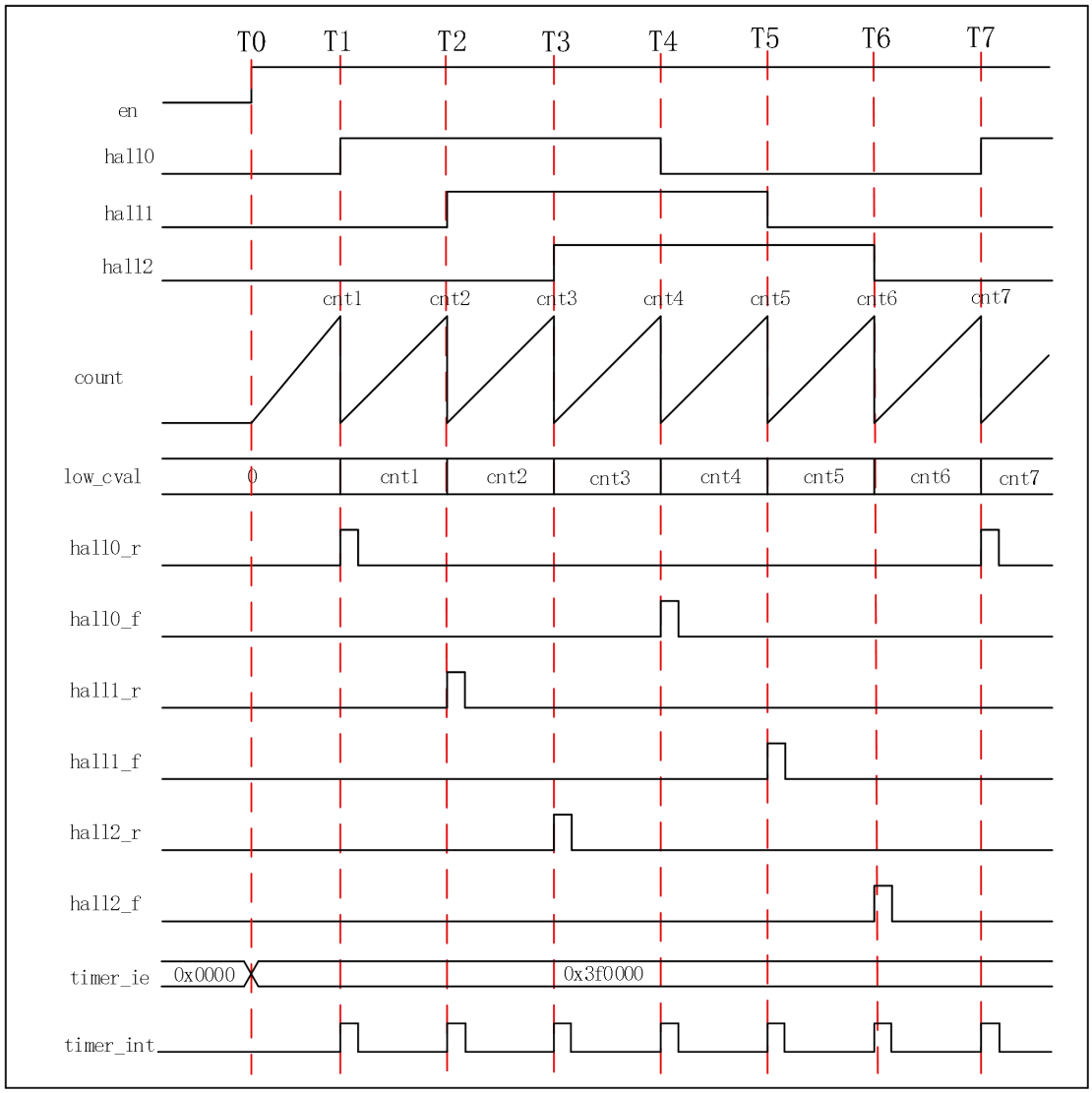


图 5-48　TIMERPLUS 在 HALL 模式下时序图

其中 hall0、hall1 和 hall2 为三路输入的 HALL 信号，count 为计数值，hall0_r、hall1_r 和 hall2_r 为检测到 HALL 信号上升沿标志信号，hall0_f、hall1_f 和 hall0_f 为检测到 HALL 信号下降沿标志信号，low_cval 为捕获的计数器计数值，timer_ie 为中断使能信号，timer_int 为产生的中断信号。电路每检测到上升沿和下降沿都会将计数器清零，并且将清零前的计数值保存到 low_cval 寄存器中，同时会产生各个 HALL 信号的上升沿或下降沿标志信号。当开启相应的中断时，在相应的状态标志信号产生后也会产生相应的中断信号，如上图中 T1-T7 时刻所示。

中断

TIMERPLUS 提供了 12 中断源，其示意图如下：

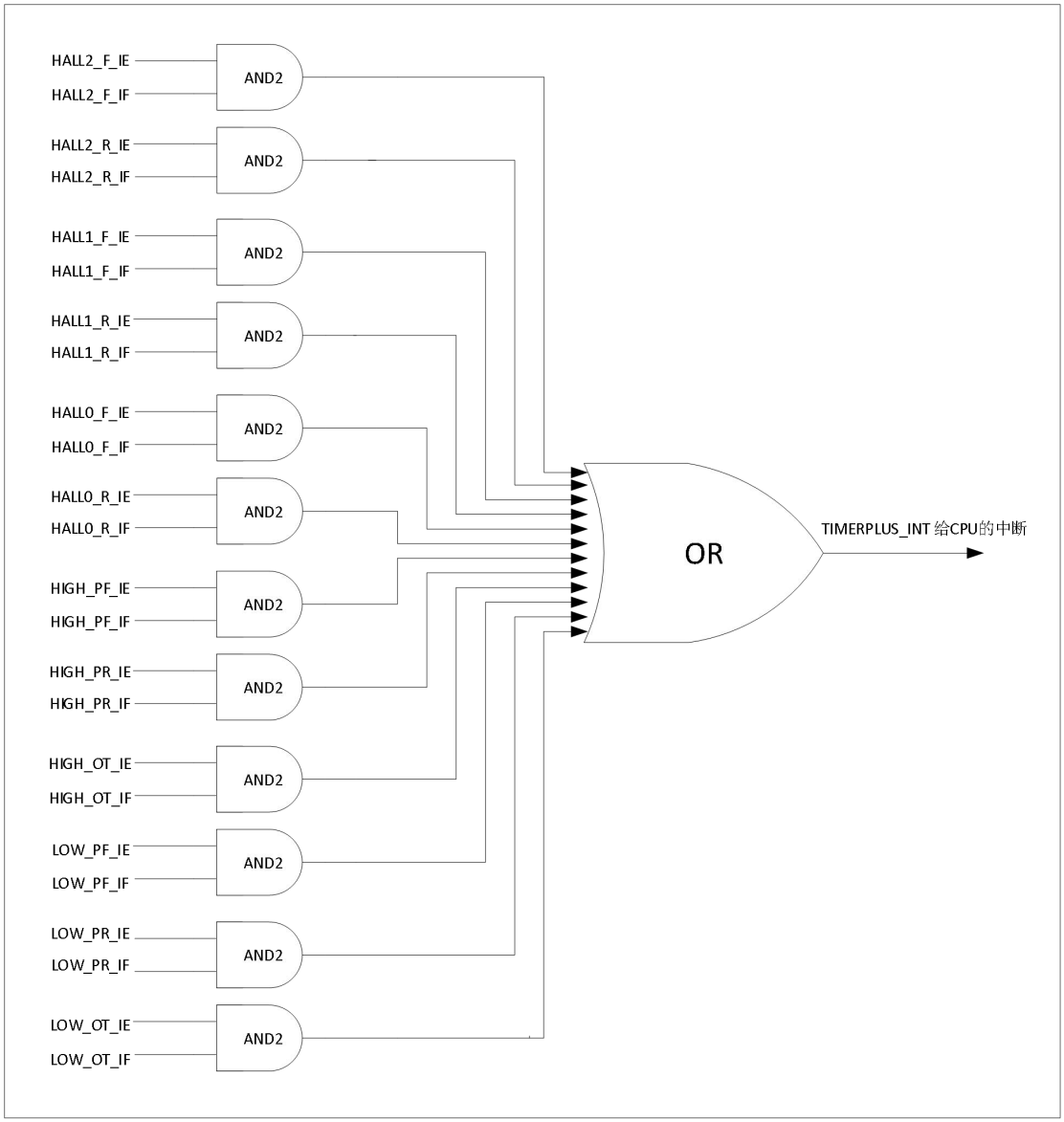


图 5-49　TIMERPLUS 中断标志及中断示意图

操作流程

定时模式操作流程

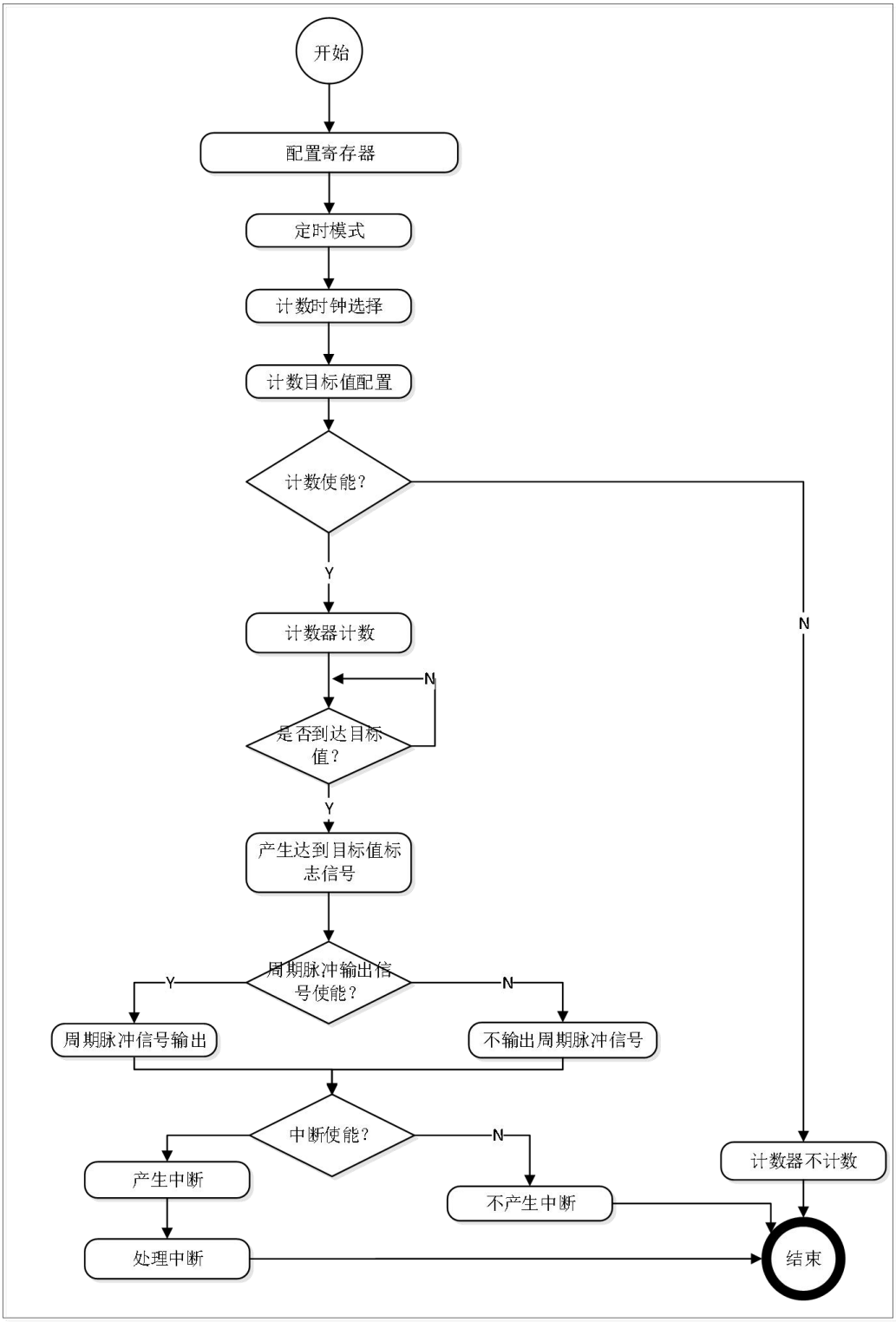


图 5-50 TIMERPLUS 定时模式操作流程图

- TIMER PLUS 时钟使能
- PORT 端口配置为 TIMER PLUS 功能
- 配置计数器工作模式寄存器，选择定时模式
- 配置计数器时钟
- 如果选择内部 pclk 时钟为计数器时钟，则需配置预分频寄存器对时钟进行分频
- 配置计数器计数目标值
- 配置中断使能
- 配置周期脉冲信号使能
- 配置计数器使能，开始计数

计数模式操作流程

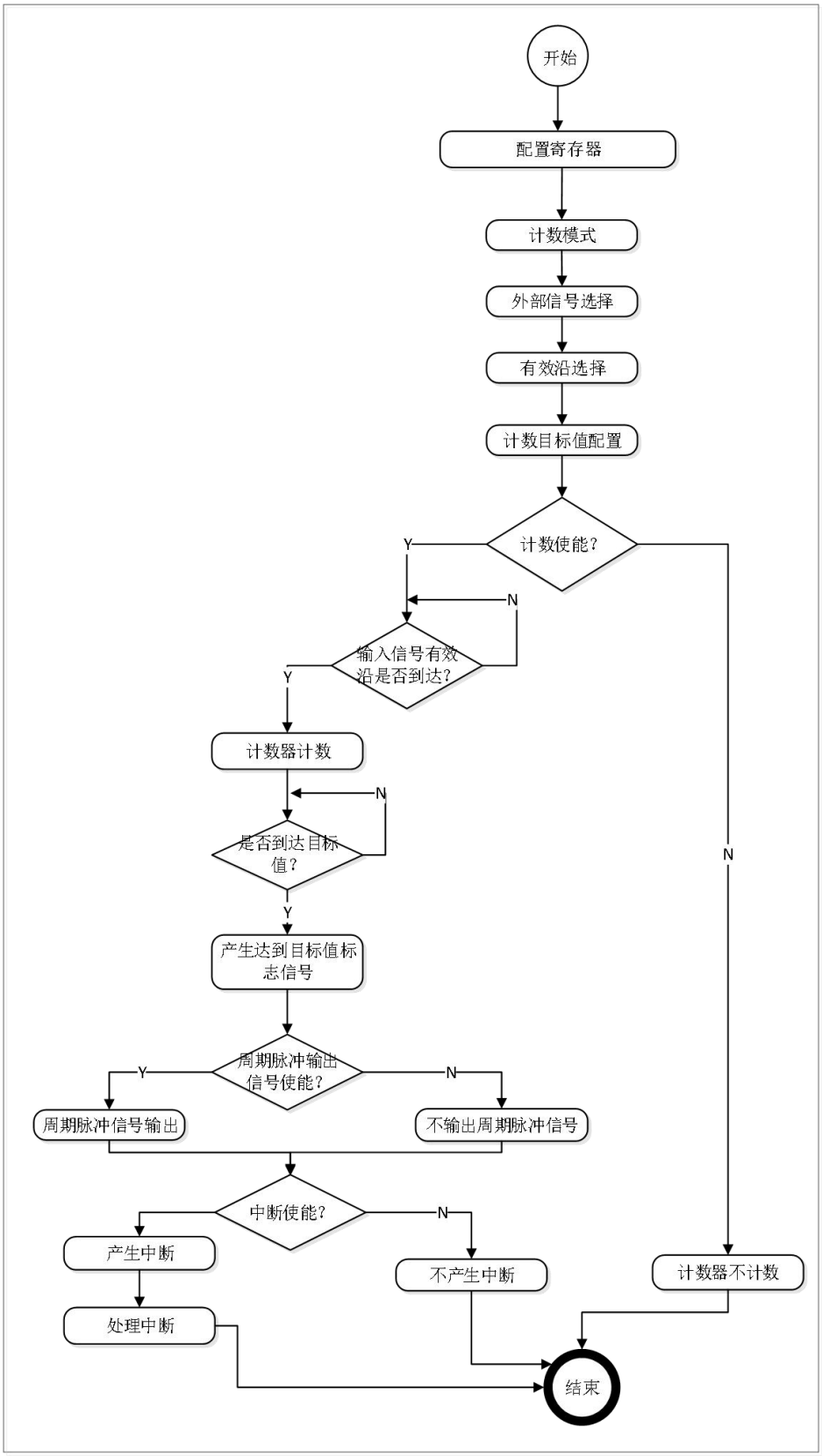


图 5-51 TIMERPLUS 计数模式操作流程图

- TIMERPLUS 时钟使能
- PORT 端口配置为 TIMERPLUS 功能
- 配置计数器工作模式寄存器，选择计数模式
- 配置片外输入信号以及有效沿
- 配置预分频寄存器对时钟进行分频
- 配置计数器计数目标值
- 配置中断使能
- 配置周期脉冲信号使能
- 配置计数器使能，开始计数

输入捕获模式操作流程

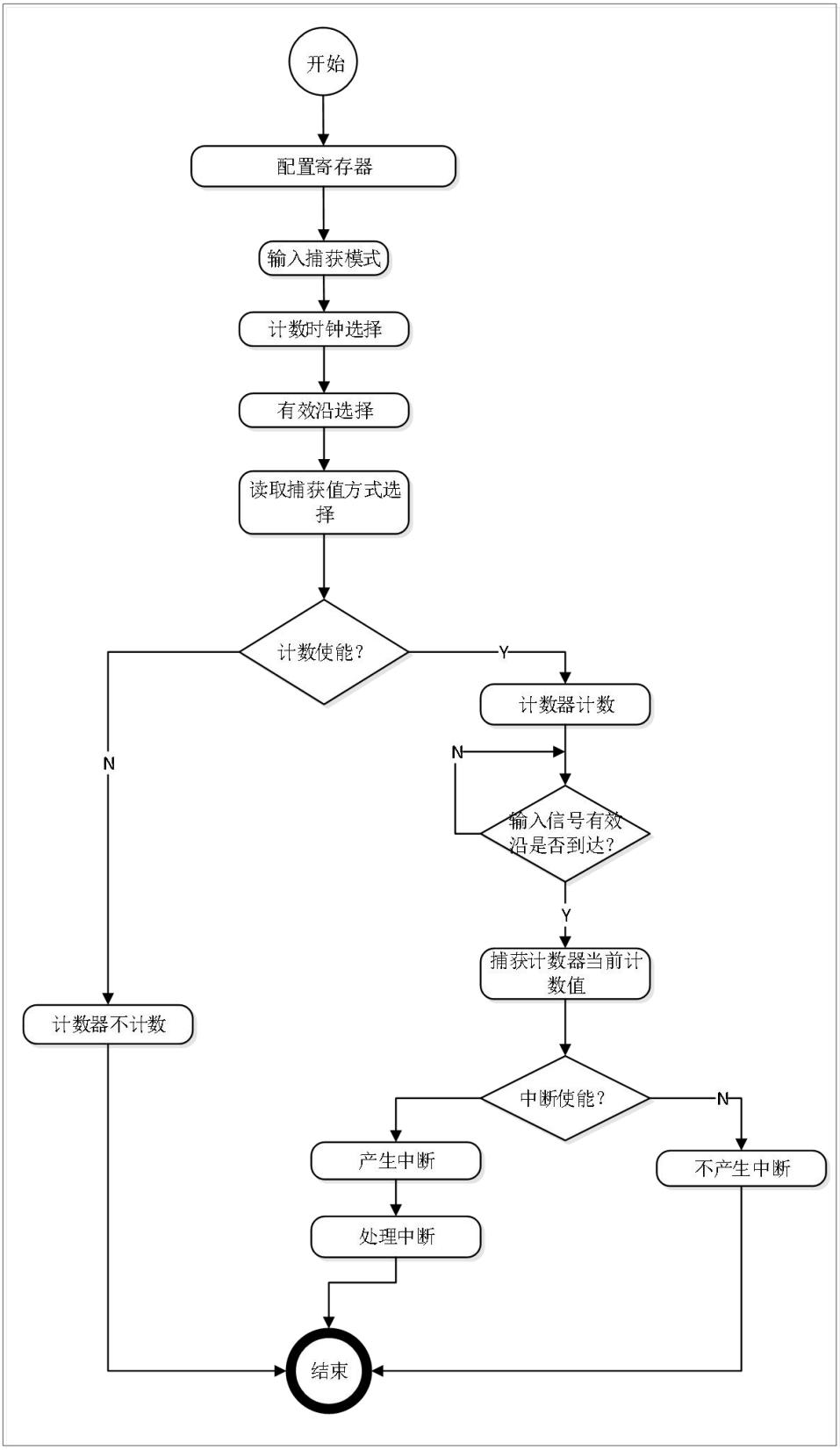


图 5-52　TIMERPLUS 输入捕获模式操作流程图

- TIMER PLUS 时钟使能
- PORT 端口配置为 TIMER PLUS 功能
- 配置计数器工作模式寄存器，选择输入捕获模式
- 配置计数器时钟
- 如果选择内部 pclk 时钟为计数器时钟，则需配置预分频计数器对时钟进行分频
- 配置输入信号以及输入信号有效沿
- 配置中断使能
- 配置计数器使能，开始计数

HALL 模式操作流程

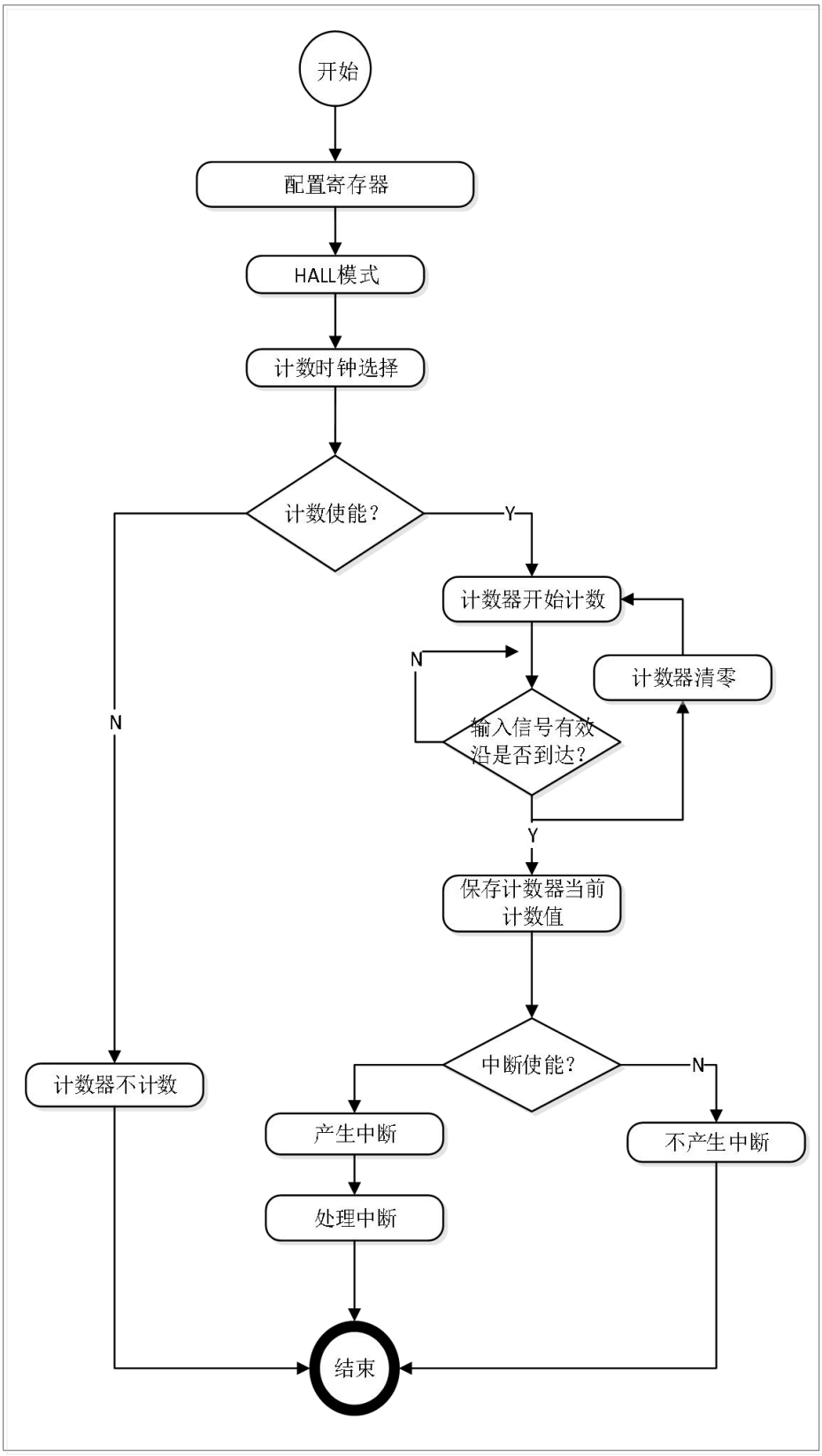


图 5-53 TIMERPLUS HALL 模式操作流程图

- TIMER PLUS 时钟使能
- PORT 端口配置为 TIMER PLUS 功能
- 选择低 16 位计数器
- 配置计数器工作模式寄存器，选择 HALL 模式
- 配置计数器时钟
- 如果选择内部 pclk 时钟为计数器时钟，则需配置预分频计数器对时钟进行分频
- 配置中断使能
- 配置计数器使能，开始计数

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
TIMERPLUS0 BASE: 0x40067000 TIMERPLUS1 BASE: 0x40067800					
TIMERPLUS_EN	0x00	32	R/W	0x00	TIMERPLUS 使能寄存器
TIMERPLUS_DIV	0x04	32	R/W	0x00	TIMERPLUS 计数时钟预分频寄存器
TIMERPLUS_CTR	0x08	32	R/W	0x600060	TIMERPLUS 配置寄存器
TIMERPLUS_IE	0x10	32	R/W	0x00	TIMERPLUS 中断使能寄存器
TIMERPLUS_IF	0x14	32	R/W	0x00	TIMERPLUS 中断状态寄存器
HIGH_GOAL	0x20	32	R/W	0xffff	TIMERPLUS HIGH 目标配置寄存器
HIGH_CNT	0x24	32	R	0x00	TIMERPLUS HIGH 当前计数值寄存器
HIGH_CVAL	0x28	32	R	0x00	TIMERPLUS HIGH 捕获值寄存器
LOW_GOAL	0x30	32	R/W	0xffff	TIMERPLUS LOW 目标配置寄存器
LOW_CNT	0x34	32	R	0x00	TIMERPLUS LOW 当前计数值寄存器
LOW_CVAL	0x38	32	R	0x00	TIMERPLUS HIGH 捕获值寄存器
HALL_VAL	0x40	32	R	0x00	HALL 信号原始值寄存器

寄存器描述

TIMERPLUS_EN 寄存器（0x00）

位域	名称	类型	复位值	描述
31:2	RESERVED	R	0	保留位
1	TIMERPLUS_HIGH_EN	R/W	0	TIMERPLUS 高 16bit 定时器使能寄存器 0：禁能 1：使能
0	TIMERPLUS_LOW_EN	R/W	0	TIMERPLUS 低 16bit 定时器使能寄存器 0：禁能 1：使能

TIMERPLUS_DIV 寄存器（0x04）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	TIMERPLUS_DIV	R/W	0	TIMERPLUS 计数时钟分频寄存器 0x0000：表示 1 分频 0x0001：表示 2 分频 …… 0xFFFF：表示 65536 分频

TIMERPLUS_CTR 寄存器（0x08）

位域	名称	类型	复位值	描述
31:25	RESERVED	R	0	保留

24	HIGH_DMA_EN	R/W	0	DMA 读取 TIMER HIGH 捕获值使能 0：CPU 读取捕获值 1：DMA 读取捕获值
23	HIGH_PO_MD	R/W	0	TIMER HIGH 周期脉冲输出使能 0：输出关闭 1：输出使能
22:21	HIGH_EXT_EDGE	R/W	0x3	TIMER HIGH 计数模式或输入捕获模式输入信号有效沿选择 00：上升沿有效 01：下降沿有效 10：上升沿或下降沿有效 11：保留
20	HIGH_EXT_SEL	R/W	0	TIMER HIGH 计数模式或输入捕获模式输入信号选择 0：timer_in0 1：timer_in1
19:18	HIGH_CLKSEL	R/W	0	TIMER HIGH 计数时钟源选择 00：PCLK/PREDIV（选择 pclk 预分频后的时钟） 01：CNTSRC0 10：CNTSRC1 11：保留
17:16	HIGH_MODE	R/W	0	TIMER HIGH 工作模式寄存器 00：定时模式（可产生周期脉冲输出信号） 01：计数模式（计数时钟只能选择 pclk） 10：输入捕获模式 11：保留
15:9	RESERVED	R	0	保留
8	LOW_DMA_EN	R/W	0	DMA 读取 TIMER LOW 捕获值使能 0：CPU 读取捕获值 1：DMA 读取捕获值
7	LOW_PO_MD	R/W	0	TIMER LOW 周期脉冲输出使能 0：输出关闭 1：输出使能

6:5	LOW_EXT_EDGE	R/W	0x3	TIMER LOW 计数模式或输入捕获模式输入信号有效沿选择 00：上升沿有效 01：下降沿有效 10：上升沿或下降沿有效 11：保留
4	LOW_EXT_SEL	R/W	0	TIMER LOW 计数模式或输入捕获模式输入信号选择 0：timer_in0 1：timer_in1
3:2	LOW_CLKSEL	R/W	0	TIMER LOW 计数时钟源选择 00：PCLK/PREDIV（选择 pclk 预分频后的时钟） 01：CNTSRC0 10：CNTSRC1 11：保留
1:0	LOW_MODE	R/W	0	TIMER LOW 工作模式寄存器 00：定时模式（可产生周期脉冲输出信号） 01：计数模式（计数时钟只能选择 pclk） 10：输入捕获模式 11：HALL 模式

TIMERPLUS_IE 寄存器（0x10）

位域	名称	类型	复位值	描述
31:22	RESERVED	R	0	保留位
21	HALL2_F_IE	R/W	0	HALL2 下降沿中断使能
20	HALL2_R_IE	R/W	0	HALL2 上升沿中断使能
19	HALL1_F_IE	R/W	0	HALL1 下降沿中断使能
18	HALL1_R_IE	R/W	0	HALL1 上升沿中断使能
17	HALL0_F_IE	R/W	0	HALL0 下降沿中断使能

16	HALL0_R_IE	R/W	0	HALL0 上升沿中断使能
15:11	RESERVED	R	0	保留位
10	HIGH_PF_IE	R/W	0	TIMERPLUS HIGH 输入脉冲下降沿中断使能
9	HIGH _PR_IE	R/W	0	TIMERPLUS HIGH 输入脉冲上升沿中断使能
8	HIGH _TO_IE	R/W	0	TIMERPLUS HIGH 达到目标值中断使能
7:3	RESERVED	R	0	保留位
2	LOW_PF_IE	R/W	0	TIMERPLUS LOW 输入脉冲下降沿中断使能
1	LOW _PR_IE	R/W	0	TIMERPLUS LOW 输入脉冲上升沿中断使能
0	LOW _TO_IE	R/W	0	TIMERPLUS LOW 达到目标值中断使能

TIMERPLUS_IF 寄存器（0x14）

位域	名称	类型	复位值	描述
31:22	RESERVED	R	0	保留位
21	HALL2_F_IF	R/W	0	HALL2 下降沿中断状态 写 1 清零
20	HALL2_R_ IF	R/W	0	HALL2 上升沿中断状态 写 1 清零
19	HALL1_F_ IF	R/W	0	HALL1 下降沿中断状态 写 1 清零
18	HALL1_R_ IF	R/W	0	HALL1 上升沿中断状态 写 1 清零
17	HALL0_F_ IF	R/W	0	HALL0 下降沿中断状态 写 1 清零
16	HALL0_R_ IF	R/W	0	HALL0 上升沿中断状态 写 1 清零
15:11	RESERVED	R	0	保留位

10	HIGH _PF_ IF	R/W	0	TIMERPLUS HIGH 输入脉冲下降沿中断状态 写 1 清零
9	HIGH _PR_ IF	R/W	0	TIMERPLUS HIGH 输入脉冲上升沿中断状态 写 1 清零
8	HIGH _TO_ IF	R/W	0	TIMERPLUS HIGH 达到目标值中断状态 写 1 清零
7:3	RESERVED	R	0	保留位
2	LOW _PF_ IF	R/W	0	TIMERPLUS LOW 输入脉冲下降沿中断状态 写 1 清零
1	LOW _PR_ IF	R/W	0	TIMERPLUS LOW 输入脉冲上升沿中断状态 写 1 清零
0	LOW _TO_ IF	R/W	0	TIMERPLUS LOW 达到目标值中断状态 写 1 清零

TIMERPLUS_HIGH_LOAD 寄存器（0x20）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	HIGH_LOAD	R/W	0xffff	TIMERPLUS HIGH 定时器目标配置寄存器 当高 16bit 计数器向上计数达到该设定值后，会产生相应状态信号

注：HIGH 计数器达到该寄存器配置的值后可以产生相应的中断状态（HIGH _TO_ IF）和外部触发信号（如模块结构框图中的 timer_goal[1]信号）。

TIMERPLUS_HIGH _CNT 寄存器（0x24）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	HIGH_CNT	R	0	TIMERPLUS HIGH 定时器当前计数值

TIMERPLUS_HIGH _CVAL 寄存器（0x28）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	HIGH_CVAL	R	0	TIMERPLUS HIGH 捕获值计数值

TIMERPLUS_LOW_LOAD 寄存器（0x30）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	LOW_LOAD	R/W	0xffff	TIMERPLUS LOW 定时器目标配置寄存器 当低 16bit 计数器向上计数达到该设定值后，会产生相应状态信号

注：LOW 计数器达到该寄存器配置的值后可以产生相应的中断状态（LOW_TO_ IF）和外部触发信号（如模块结构框图中的 timer_goal[0]信号）。

TIMERPLUS_ LOW _CNT 寄存器（0x34）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	LOW _CNT	R	0	TIMERPLUS LOW 定时器当前计数值

TIMERPLUS_ LOW _CVAL 寄存器（0x38）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	LOW _CVAL	R	0	TIMERPLUS LOW 捕获值计数值

HALL_VAL 寄存器（0x40）

位域	名称	类型	复位值	描述
31:3	RESERVED	R	0	保留位
2	HALL2_VAL	R	0	HALL2 的原始信号电平
1	HALL1_VAL	R	0	HALL1 的原始信号电平
0	HALL0_VAL	R	0	HALL0 的原始信号电平

5.11 独立看门狗时钟（IWDT）

5.11.1 概述

独立看门狗定时器（IWDT）主要用于控制程序流程正确，在程序流长时间未按既定流程执行指定程序的情况下复位芯片。使用前需使能对应 IWDT 模块时钟。其系统框图如下图所示：

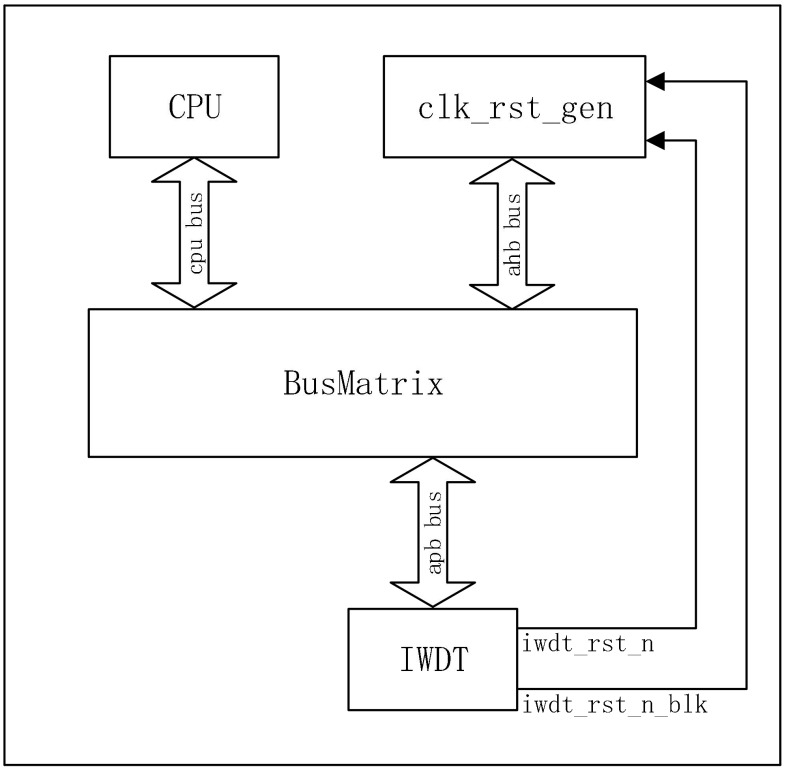


图 5-54 IWDT 系统框图

5.11.2特性

- 产生计数器溢出复位信号，复位信号使能可配
- 具有 32 位计数位宽，可配置灵活、宽范围的溢出周期
- 具有中断功能，提供周期性的计数溢出中断信号，如果该中断信号在下一个中断信号产生之前仍没有被清除，则产生 Watchdog 复位信号。
- 具有喂狗功能

5.11.3模块结构框图

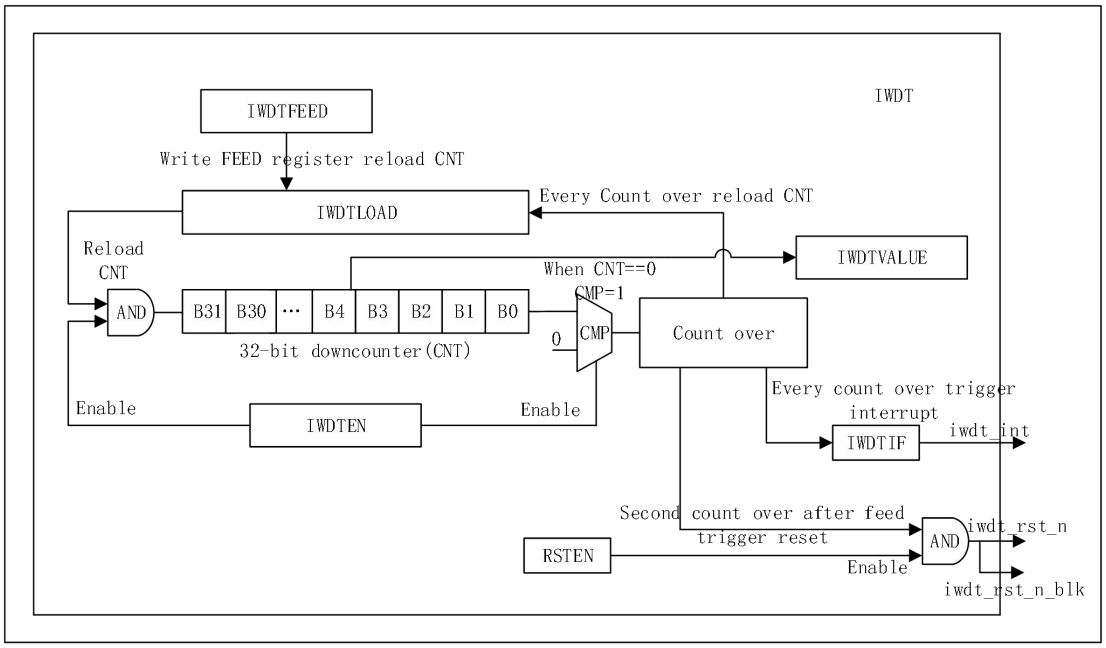


图 5-55 IWDT 模块结构框图

上图为独立看门狗时钟模块的结构框图，本模块有一个 32bit 的向下计数的计数器，计数器初始值可以通过寄存器 IWDTLOAD 配置，并通过计数器使能 IWDTEN 控制计数器的开始和停止，当计数器计数到 0 时，会产生中断信号，iwdt_int 中断信号产生后，如果一直没有得到响应，那么当计数器再次计数到 0，如果 IWDTControl 寄存器中 RSTEN 位为 1 时，复位信号有效，否则当 RSTEN 为 0 时，复位信号无效，此信号可用于产生系统复位。

5.11.4功能描述

中断产生

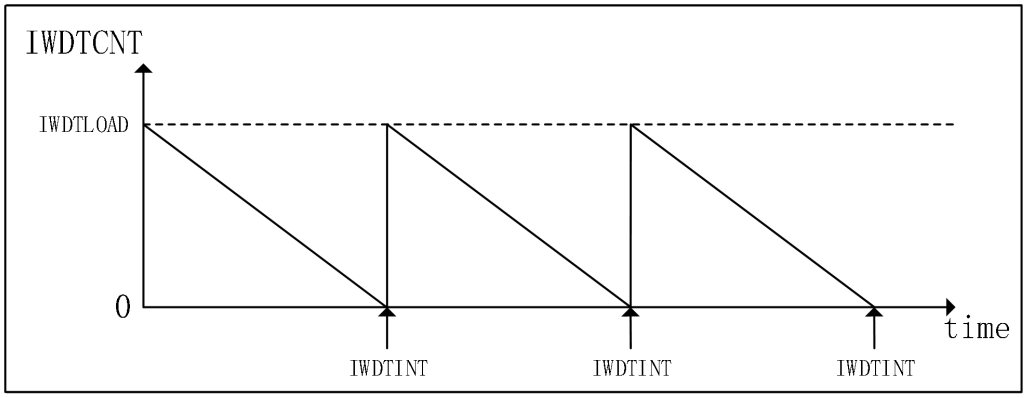


图 5-56 IWDT 中断产生图

IWDT 内置 32BIT 计数器 IWDTCNT，当看门狗使能 IWDTEN 有效时，IWDTCNT 会载入设置好的 IWDTLOAD 的值，并向下开始递减计数，每当 IWDTCNT 计数值到达 0 时，将会产生 IWDT 中断状态 WDT_IF 信号，并且会直接产生系统中断。

注意，中断状态的产生只与 IWDTCNT 的当前计数值有关，每当计数值到达 0 时便会产生中断状态。

当中断一直不清除，等到计数值再次计到 0 时，若此时 RSTEN 位有效，则将会产生看门狗复位，此时 IWDTCNT 会被清 0。

IWDT 喂狗及复位产生

1、第一次中断前喂狗

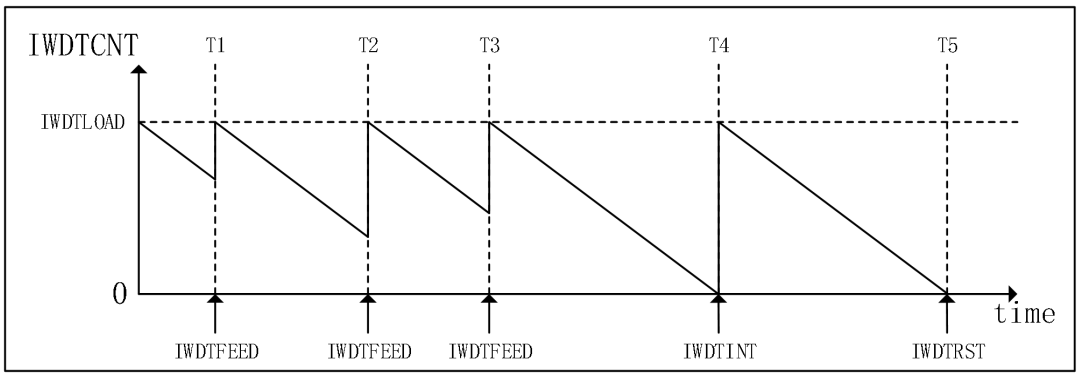


图 5-57 IWDT 第一次中断前喂狗及复位

如上图所示，当 IWDTCNT 计数器还未第一次到达 0 时进行看门狗喂狗，计数器会重新从 IWDTLOAD 开始计数（T1、T2、T3 时刻）。当最后一次喂狗后（T3 时刻），IWDTCNT 产生了两次中断时（T4、T5 时刻），并且 RSTEN 有效，则会产生看门狗复位 IWDTRST（T5 时刻）。

看门狗复位仅在喂狗后的第二次中断产生时产生，若第二次中断产生之前又进行喂狗操作，则重新记录中断个数。

2、第二次中断前喂狗

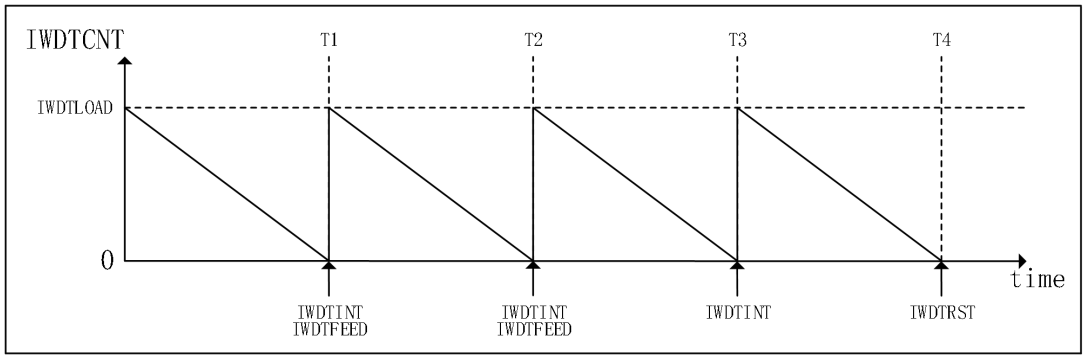


图 5-58 IWDT 第二次中断前喂狗及复位

如上图所示，当 IWDTCNT 计数器一次到达 0 后进行看门狗喂狗，计数会重新从 IWDTLOAD 开始计数（T1、T2 时刻）。当最后一次喂狗后（T2 时刻），IWDTCNT 产生了两次中断时（T3、T4 时刻），并且 RSTEN 有效，则产生看门狗复位 IWDTRST（T4 时刻）。

操作流程

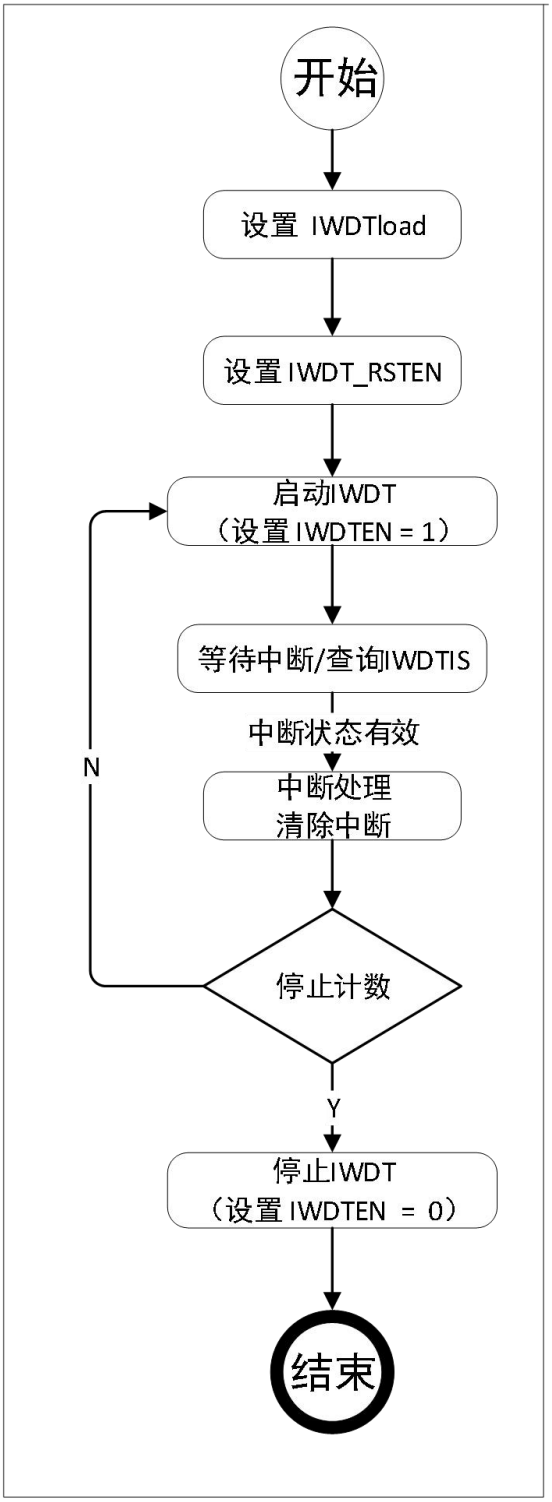


图 5-59 IWDT 中断产生操作流程图

- IWDT 时钟使能
- 配置初始值寄存器（IWDTLOAD）
- 配置复位使能寄存器（RSTEN）
- 配置使能寄存器（IWDTEN）
- 等待中断产生并处理中断
- 如果中断在 IWDTValue 再次计数到 0 时仍未被处理，则会产生看门狗复位

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
IWDT	BASE: 0x4006A000				
IWDT_LOAD	0x00	32	R/W	0xfffff	IWDT 初值寄存器
IWDT_CTRL	0x08	32	R/W	0x00	IWDT 控制寄存器
IWDT_IF	0x0C	32	R/W	0x00	IWDT 状态寄存器
IWDT_FEED	0x10	32	R/W	0x00	IWDT 喂狗寄存器

寄存器描述

IWDT_LOAD 寄存器（0x00）

位域	名称	类型	复位值	描述
31:20	RESERVED	RO	0	保留位
19:0	IWDTLOAD	R/W	0xfffff	IWDT 计数器的初始值配置寄存器。 IWDT 启动时，计数器自动装载 IWDTLOAD 值，开始递减计数。当计数器值计到 0 时，硬件自动将 IWDTLOAD 寄存器中的值重新装载到计数器中，继续进行减计数。 当第一次计数到 0 时，产生中断，若不喂狗，再次减计数到 0 时，产生复位。 该寄存器必须在 IWDTEN 无效时配置。

IWDT_CTRL 寄存器（0x08）

位域	名称	类型	复位值	描述
31:2	RESERVED	RO	0	保留位
1	INTEN	R/W	0	IWDT 中断使能位 1：使能 0：禁止
0	IWDTEN	R/W	0	IWDT 启动位 1：启动 IWDT 计数 0：停止计数

IWDT_IF 寄存器（0x0C）

位域	名称	类型	复位值	描述
31:1	RESERVED	RO	0	保留位
0	IWDT_IF	R/W	0	IWDT 状态位，计数到 0，高有效 硬件置位，软件写 1 清除

IWDT_FEED 寄存器（0x10）

位域	名称	类型	复位值	描述
31:8	RESERVED	RO	0	保留位
7:0	FEED	R/W	0	IWDT 重启计数器寄存器 向该寄存器写入 0x55 后会重启 IWDT 计数器（喂狗操作）

5.12 窗口看门狗时钟（WWDT）

5.12.1概述

窗口看门狗定时器（WWDT）主要用于控制程序流程正确，在程序流长时间未按既定流程执行指定程序的情况下复位芯片。使用前需使能对应 WWDT 模块时钟。

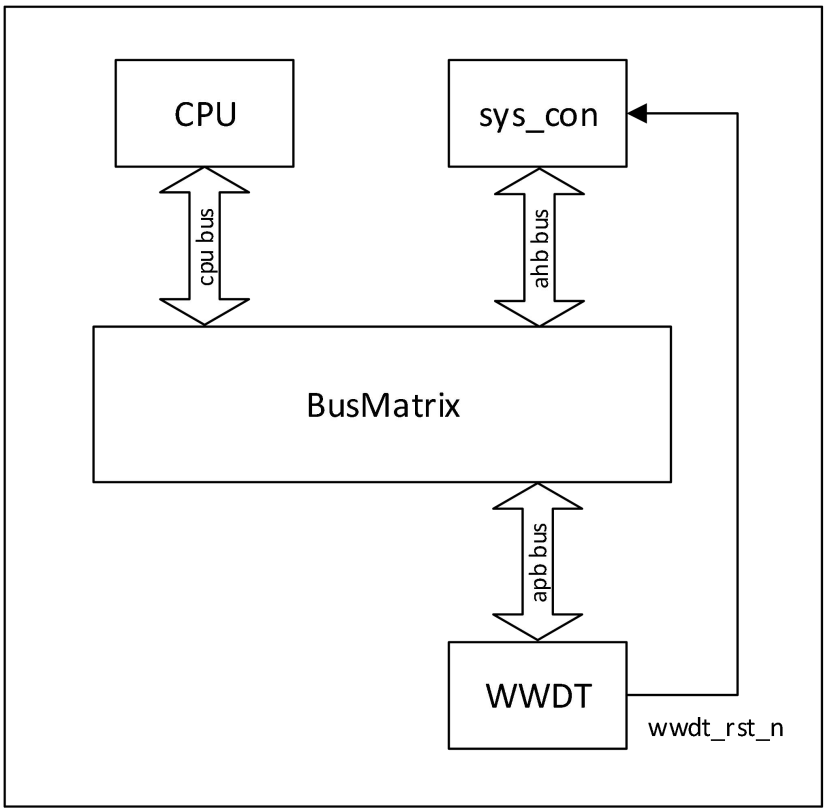


图 5-60 WWDT 模块系统框图

5.12.2 特性

- 看门狗计数时钟源为系统时钟的 4096 分频
- 功能特性：支持 7 位减计数，提供窗口中断与预复位中断，如果在预复位中断后的一个看门狗计数周期内仍然不进行喂狗操作，则产生复位信号。
- 分别支持中断窗口值及复位窗口值独立可配置
- 看门狗计数时钟频率可配置

- 支持预复位中断报警功能

5.12.3 模块结构框图

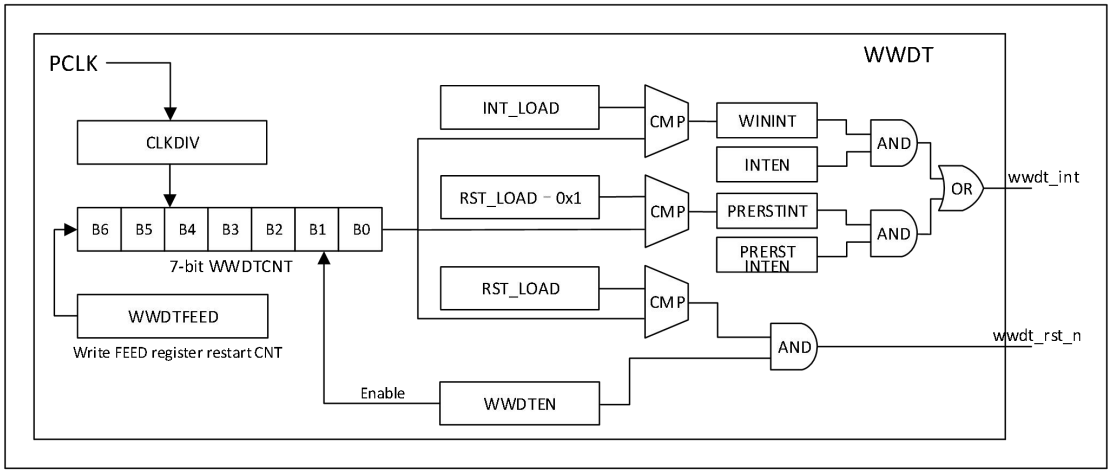


图 5-61 WWDT 模块结构框图

WWDT 内置 7 位计数器，计数时钟为 PCLK 经过 CLKDIV 分频后的时钟，当 WWDTEN 置 1，使能计数器，计数器从寄存器载入计数初始值 0x7f，开启减计数。

当计数值到达 INT_LOAD 设置值时，WWDT 产生 WININT 中断状态，若开启中断使能 INTEN，则会发生系统中断 wwdt_int。

当计数值到达 RST_LOAD-1 设置值时，WWDT 产生 PRERSTINT 中断状态，若开启中断使能 PRERSTINTEN，则会发生系统中断 wwdt_int。

当计数值到达 RST_LOAD 设置值时，产生看门狗复位 wwdt_rst_n。

写入 WWDTFEED 寄存器 0x55 发生看门狗喂狗，使得计数器从 0x7f 值重新开启计数。

5.12.4 功能描述

WWDT 中断产生及喂狗区间

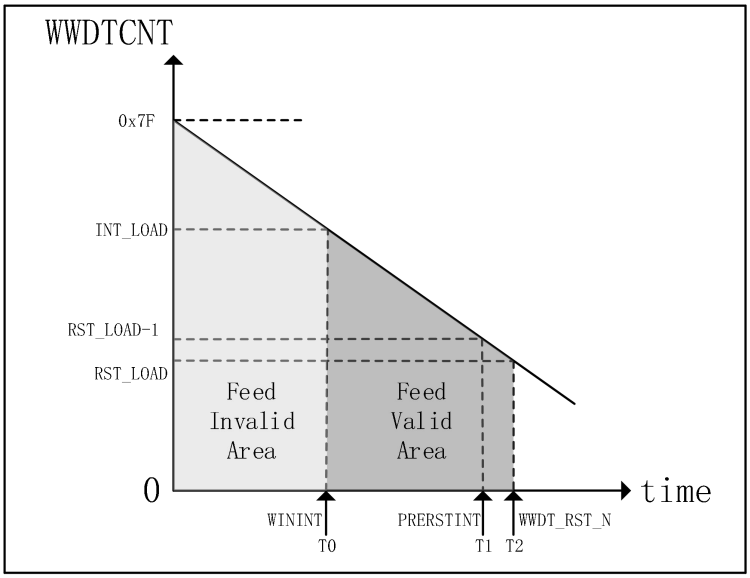


图 5-62　WWDT 中断产生及喂狗区间

如上图所示：

当 WWDTCNT 计数值到达 INT_LOAD 时（T0 时刻），产生 WININT 中断状态；

当 WWDTCNT 计数值到达 RST_LOAD-1 时（T1 时刻），产生 PRERSTINT 中断状态；

当 WWDTCNT 计数值到达 RST_LOAD 时（T2 时刻），产生 WWDT_RST_N 看门狗复位；

WWDT 中断产生只与当前计数值有关。

从 0 时刻到 T0 时刻为无效喂狗区间，若此时喂狗将会立刻产生看门狗复位；

从 T0 时刻到 T2 时刻为有效喂狗区间，若此时喂狗将会重置 WWDTCNT 看门狗计数器。

WWDT 喂狗及复位产生

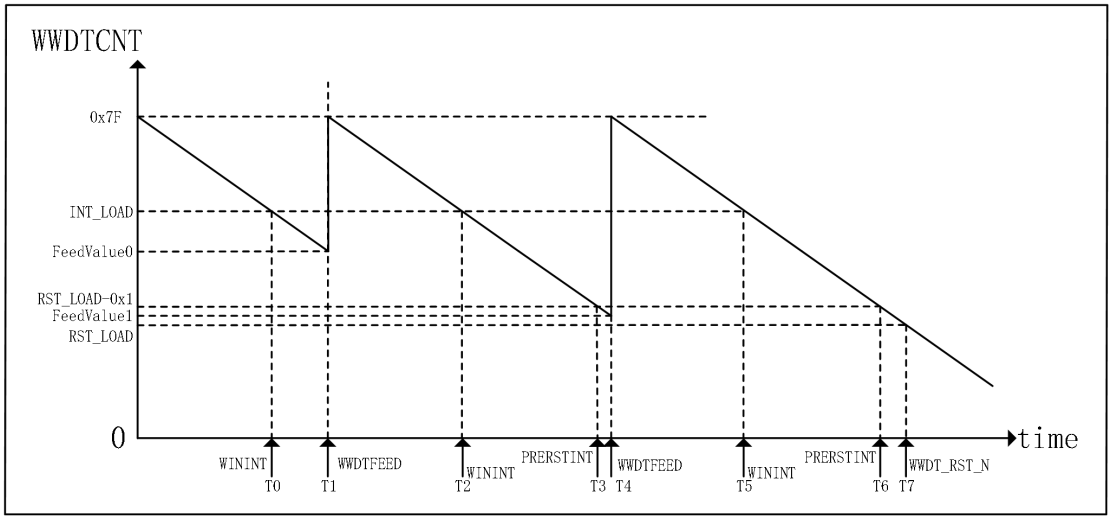


图 5-63　WWDT 喂狗及复位

如上图所示：

当 WWDTCNT 计数值到达 INT_LOAD 时（T0 时刻），产生 WININT 中断状态；

当 WWDTCNT 计数值到达 FeedValue0 时（T1 时刻），进行看门狗喂狗，计数器重新开始计数；

当 WWDTCNT 计数值到达 INT_LOAD 时（T2 时刻），产生 WININT 中断状态；

当 WWDTCNT 计数值到达 RST_LOAD-1 时（T3 时刻），产生 PRERSTINT 中断状态；

当 WWDTCNT 计数值到达 FeedValue1 时（T4 时刻），进行看门狗喂狗，计数器重新开始计数；

当 WWDTCNT 计数值到达 INT_LOAD 时（T5 时刻），产生 WININT 中断状态；

当 WWDTCNT 计数值到达 RST_LOAD-1 时（T6 时刻），产生 PRERSTINT 中断状态；

当 WWDTCNT 计数值到达 RST_LOAD 时（T7 时刻），产生 WWDT_RST_N 看门狗复位。

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
WWDT	BASE: 0x4006A800				
WWDT_LOAD	0x00	32	R/W	0x40	WWDT 初值寄存器
WWDT_VALUE	0x04	32	R/W	0x7f	WWDT 当前计数值寄存器
WWDT_CTRL	0x08	32	R/W	0x00	WWDT 控制寄存器
WWDT_IF	0x0C	32	R/W	0x00	WWDT 中断状态寄存器
WWDT_FEED	0x10	32	R/W	0x00	WWDT 喂狗寄存器

寄存器描述

WWDT_LOAD 寄存器（0x00）

位域	名称	类型	复位值	描述
31:14	RESERVED	RO	0	保留位
13:8	RST_LOAD	R/W	0x0	窗口复位比较值寄存器 注 1：看门狗向下计数到该值时，可产生复位 注 2：看门狗向下计数到该值加 1 时，可产生中断
7	RESERVED	RO	0	保留位
6:0	INT_LOAD	R/W	0x40	窗口中断比较值寄存器 注 1：配置时，窗口中断比较值一定要大于窗口复位比较值 注 2：看门狗向下计数到该值时，可产生中断

WWDT_VALUE 寄存器（0x04）

位域	名称	类型	复位值	描述
31:7	RESERVED	RO	0	保留位

6:0	VALUE	R	0x7f	该寄存器为只读寄存器，复位值为 0x7f。读该寄存器时，返回计数器的当前计数值。

WWDT_CTRL 寄存器（0x08）

位域	名称	类型	复位值	描述
31:5	RESERVED	RO	0	保留位
4	PRERSTINTEN	R/W	0	预复位中断使能位 1：使能 0：禁止
3	INTEN	R/W	0	窗口中断使能位 1：使能 0：禁止
2:1	CLKDIV	R/W	0	看门狗计数器计数时钟预分频 00：看门狗计数时钟 1 分频 01：看门狗计数时钟 2 分频 10：看门狗计数时钟 4 分频 11：看门狗计数时钟 8 分频 注：看门狗计数时钟为系统时钟的 4096 分频
0	EN	R/W	0	WWDT 启动位 1：启动 WWDT 计数 0：停止计数

WWDT_IF 寄存器（0x0C）

位域	名称	类型	复位值	描述
31:2	RESERVED	RO	0	保留位
1	PRERSTINT	R/W	0	预复位中断标志位（即 VALUE = RST_LOAD + 1 时），高有效 硬件置位，软件写 1 清除
0	WININT	R/W	0	窗口中断标志位（即 VALUE = INT_LOAD 时），高有效 硬件置位，软件写 1 清除

WWDT_FEED 寄存器（0x10）

位域	名称	类型	复位值	描述
31:8	RESERVED	RO	0	保留位
7:0	FEED	W	0	WWDT 重启计数器寄存器 向该寄存器写入 0x55 后会重启 WWDT 计数器（喂狗操作） 注：只有在窗口期（INT_LOAD >VALUE > RST_LOAD）内喂狗才能生效，否则喂狗操作将产生看门狗复位。

5.13 基本脉冲宽度调制发生器（PWMBASE）

5.13.1 概述

PWMBASE 为基本脉冲宽度调制发生器，本芯片 PWMBASE 模块提供 3 路（CH0、CH1、CH2）独立通道，支持预分频功能，支持输出电平翻转，支持到达翻转点中断和周期结束中断。使用 PWMBASE 模块之前需要使能 PWMBASE 时钟。PWMNASE 模块是通过 APB 总线来实现与 CPU 连接的，并可以产生中断和输出 PWM 波形到 PAD 端口，其系统框图如下图所示：

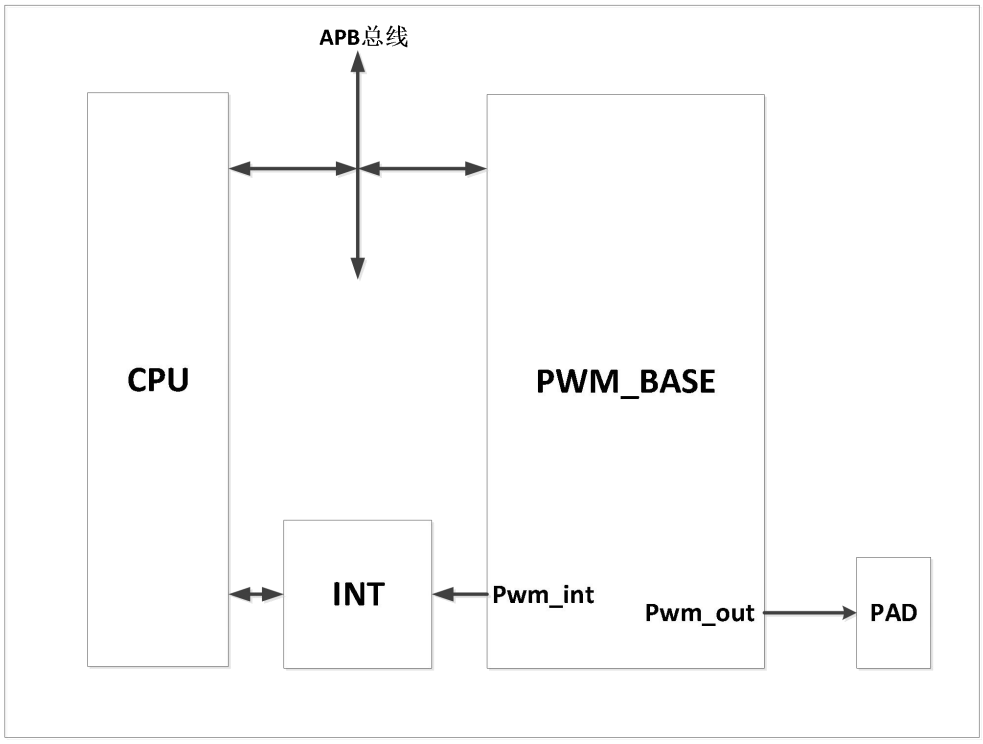


图 5-64 PWMBASE 模块系统框图

5.13.2 特性

- 3 个 16bit PWM，可输出不同占空比的 PWM 波形
- 8bit 预分频计数器
- 输出电平是否翻转
- 支持到达翻转点中断和周期结束中断

- 内部为一个向下计数的计数器

5.13.3模块结构框图

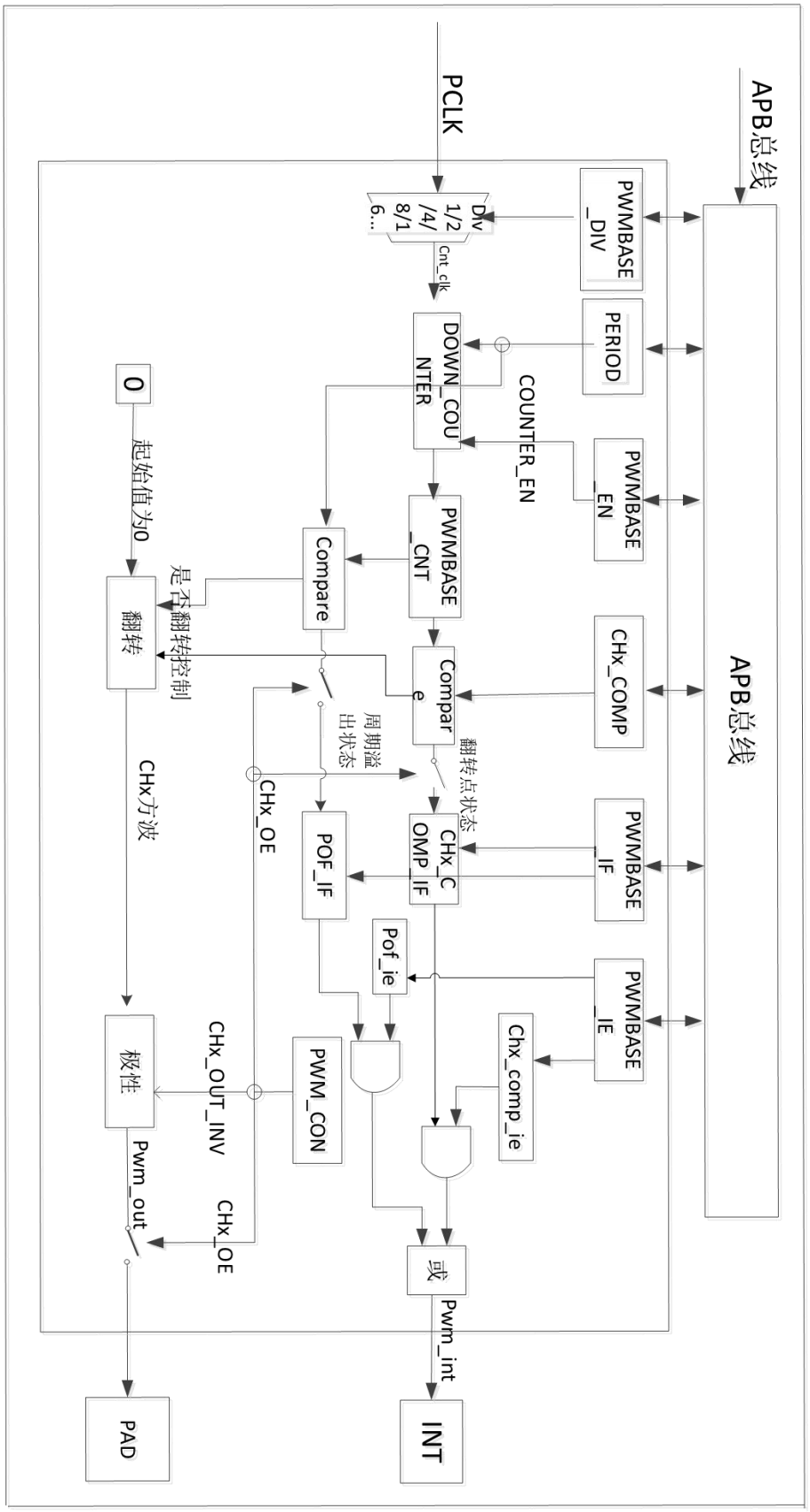

APB总线
PCLK
PWMBASE_DIV
Div 1/2/4/8/16...
Cnt_clk
PERIOD
DOWN_COUNTER
COUNTER_EN
PWMBASE_EN
0
起始值为0
PWMBASE_CNT
Compare
是否翻转控制
翻转
CHx_COMP
Compare
周期溢出状态
翻转点状态
CHx方波
CHx_OE
POF_IF
CHx_COMP_IF
PWMBASE_IF
Pof_ie
PWMBASE_IE
Chx_comp_ie
极性
CHx_OUT_INV
PWM_CON
Pwm_out
CHx_OE
或
Pwm_int
PAD
INT
APB总线

图 5-65 PWMBASE 模块结构框图

上图为 PWMBASE 模块结构框架图，从图中可以看出，系统时钟 pclk 可以通过 8bit 的预分频寄存器 div 进行分频，分频后为向下计数的计数器提供计数时钟。通过周期配置寄存器 period 给计数器配置计数周期，计数器使能 counter_en 控制计数的开始和结束。计数开始以后，当前计数值会存在 PWM_CNT 寄存器，当前计数值会与配置的翻转点和周期值比较，因为通道输出的起始电平为 0，且为向下计数的计数器，所以当计数值小于翻转点值时输出为 1，大于等于翻转点值时输出为 0，当计数满周期后，波形再次翻转。本模块通过 PWMBASE 输出配置寄存器来配置 PWMBASE 波形的输出，其中 chx_out_inv 控制通道波形的极性，chx_out_inv 为 1 时输出与原始波形相反，chx_out_inv 为 0 时输出与原始波形一致；chx_oe 控制通道波形输出的使能，当 chx_oe 为 1 时，PWMBASE 输出通道波形，当 chx_oe 为 0 时 PWMBASE 输出高阻态。

中断信号的产生受中断状态和中断使能的控制，中断状态分为翻转点状态和周期溢出状态，并且中断状态只有在 chx_oe 为 1 时才能产生，在有中断状态产生的情况下，相应的中断使能打开即各通道的翻转点中断使能和周期溢出中断使能，才产生中断信号。

5.13.4功能描述

周期值与翻转点

当通道输出使能寄存器 chx_oe 为 1 时，通过寄存器 PWMBASE_PERIOD 和 PWM_CHX_COMP 可以配置输出波形的不同的周期值和翻转点值，如下图所示，周期值配置为 7，不同翻转点值的时序图。

配置周期值与翻转点值的关系举例（空闲起始电平为 0）：

1） 用户想要产生的 PWM 波形为：周期 8，占空比为 25%，则需要配置为：PERIOD=0x07，CHx_COMP=0x02。

2） 用户想要产生的 PWM 波形为：周期 8，占空比为 87.5%，则需要配置为：PERIOD =0x07，CHx_COMP=0x07。

3） 用户想要产生的 PWM 波形为：周期 8，占空比为 100%，则需要配置为：PERIOD =0x07，CHx_COMP=0x08（大于 7 即可）。

4） 用户想要产生的 PWM 波形为：周期 8，占空比为 0%，则需要配置为：PERIOD =0x07，CHx_COMP=0x00。

以下示意图实例中实现了上述周期值和翻转点值关系下的输出波形。

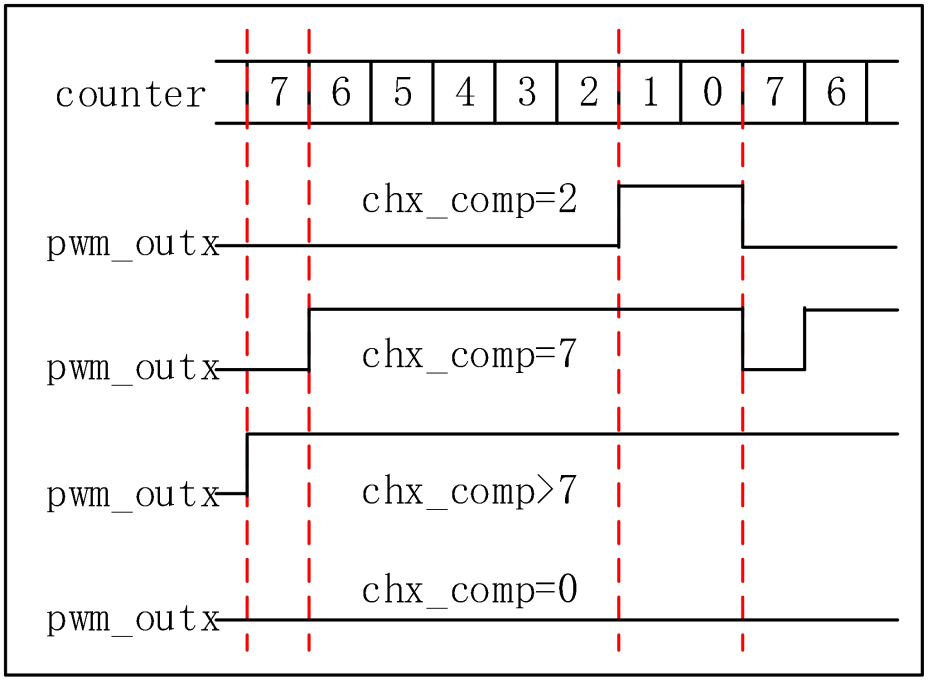


图 5-66 PWMBASE 不同周期和翻转点时序图

预分频计数

本模块可以通过 8bit 预分频计数寄存器 pwmbase_div 配置不同时钟分频值，实现分频计数，分频值范围为 1-256，在下图中，举例了翻转点值为 3，周期值为 6，分频值为 1，2，6 时的时序图。

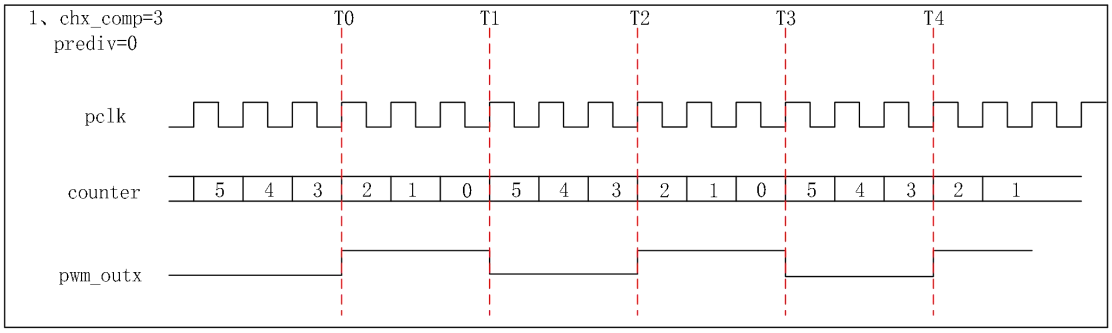


图 5-67 PWMBASE 1 分频时序图

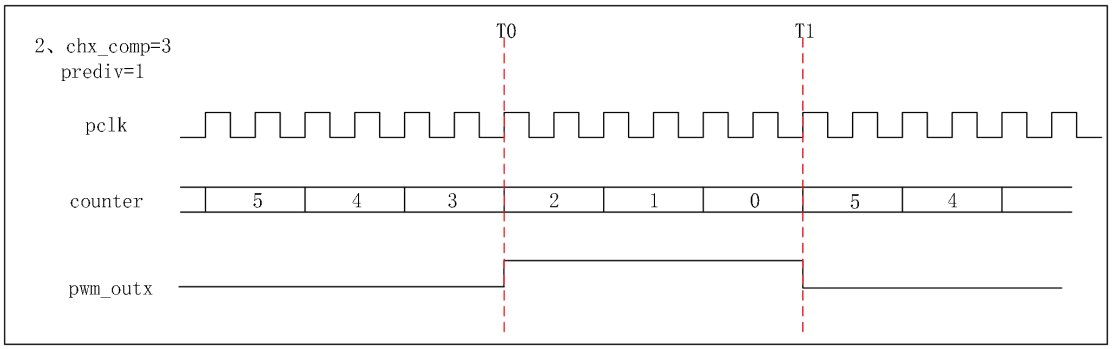


图 5-68　PWMBASE 2 分频时序图

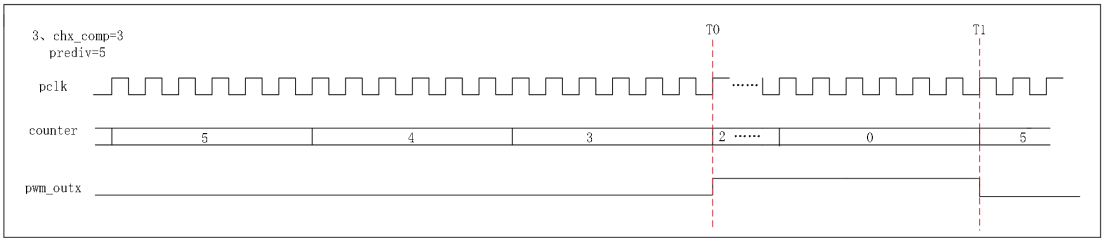


图 5-69　PWMBASE 6 分频时序图

输出使能

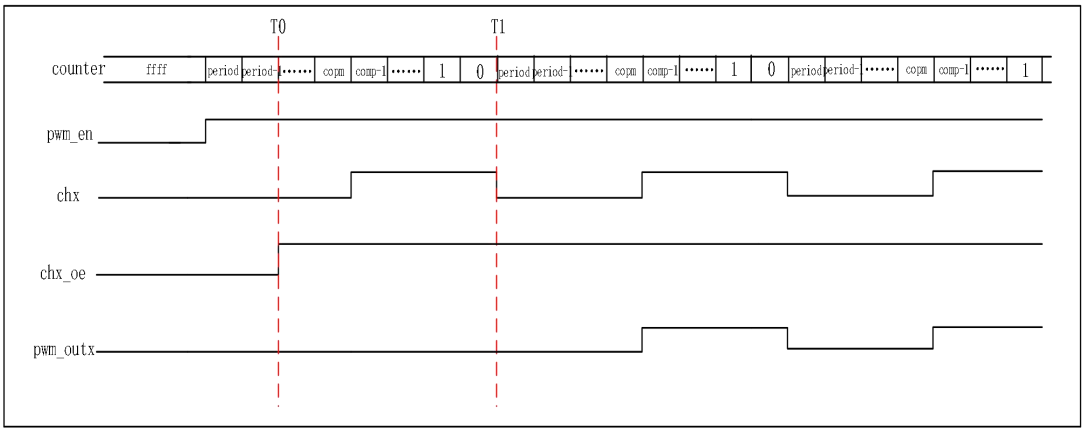


图 5-70　PWMBASE 输出使能时序图

上图为 PWMBASE 的输出使能时序图，其中 chx 为原始波形，chx_oe 为通道波形输出使能，pwm_outx 为 pwm 输出波形，通道波形使能之前即 T0 时刻之前，pwm 输出起始电平 0，当 T0 时刻通道波形使能寄存器 chx_oe 变为 1 时，pwm 并没有立刻输出波形，而是输出通

道的初始电平 0，从下一周期 T1 时刻才开始输出通道波形，以保证输出完整波形。需要注意的是通道波形使能打开前相应 PAD 的端口输出为高阻态，只有通道波形使能打开后，相应 PAD 的端口才输出 pwm_outx 的波形。

输出翻转

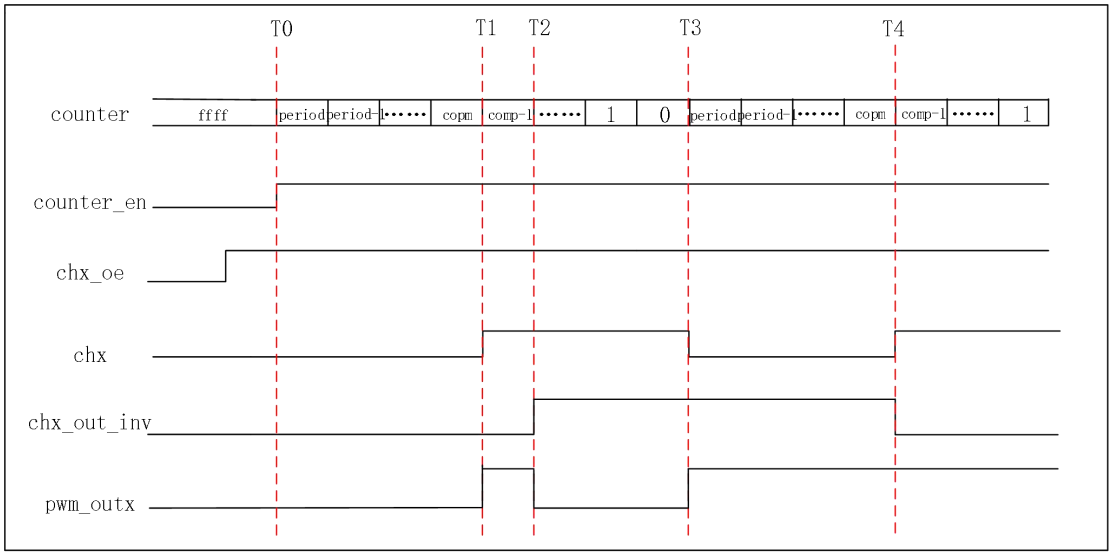


图 5-71　PWMBASE 输出翻转时序图

如上图所示，在计数器使能和通道波形输出使能同时打开的情况下，当配置极性翻转寄存器 chx_out_inv 为 0 时，可以看到在 T0-T2 时间段内 pwm 输出波形与原始波形一致，当 T2 时刻配置 chx_out_inv 为 1 后，pwm 输出的波形将立刻发生极性转换，与原始波形相反，当 T4 时刻再次将 chx_out_inv 配置为 0 后，pwm 输出波形将再次与原始波形一致。

中断

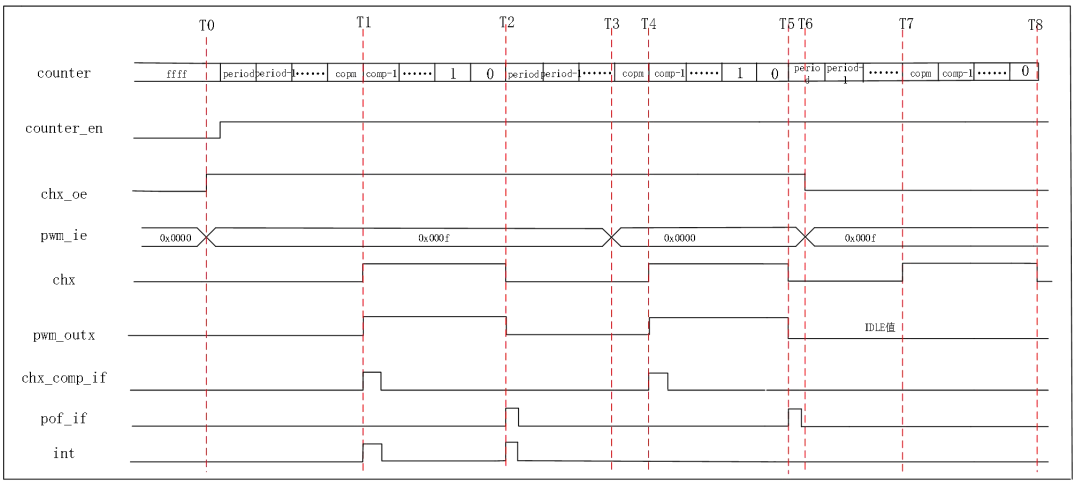


图 5-72　PWMBASE 输出中断时序图

本模块支持多种中断，如上图所示，在 T0 时刻将通道波形输出使能 chx_oe 配置为 1，同时将中断使能寄存器 pwm_ie 配置为 0x000f，即将各个通道的翻转点中断和周期溢出中断都打开。打开计数器使能，在 T1 时刻，当计数值到达翻转时，翻转点中断状态寄存器 chx_comp_if 将会拉高，同时产生中断信号，随后处理中断，中断状态寄存器写 1 清零；当计数值为 0 时，表示计数一个周期，此时周期溢出状态变为 1，如图中 T2 时刻，并产生中断信号，随后处理中断，清除中断状态。当在 T3 时刻，将 pwm_ie 配置为 0x0000，关闭翻转点和周期溢出中断使能，在 T4 和 T5 时刻只会产生相应的中断状态，而不会产生中断信号。在 T6 时刻再次打开翻转点和周期溢出中断使能，而将通道波形输出使能 chx_oe 配置为 0 时，即使开启了中断使能，但在 T7，T8 时刻翻转点和周期溢出中断状态并不会产生，因此，不会产生中断信号。

操作流程

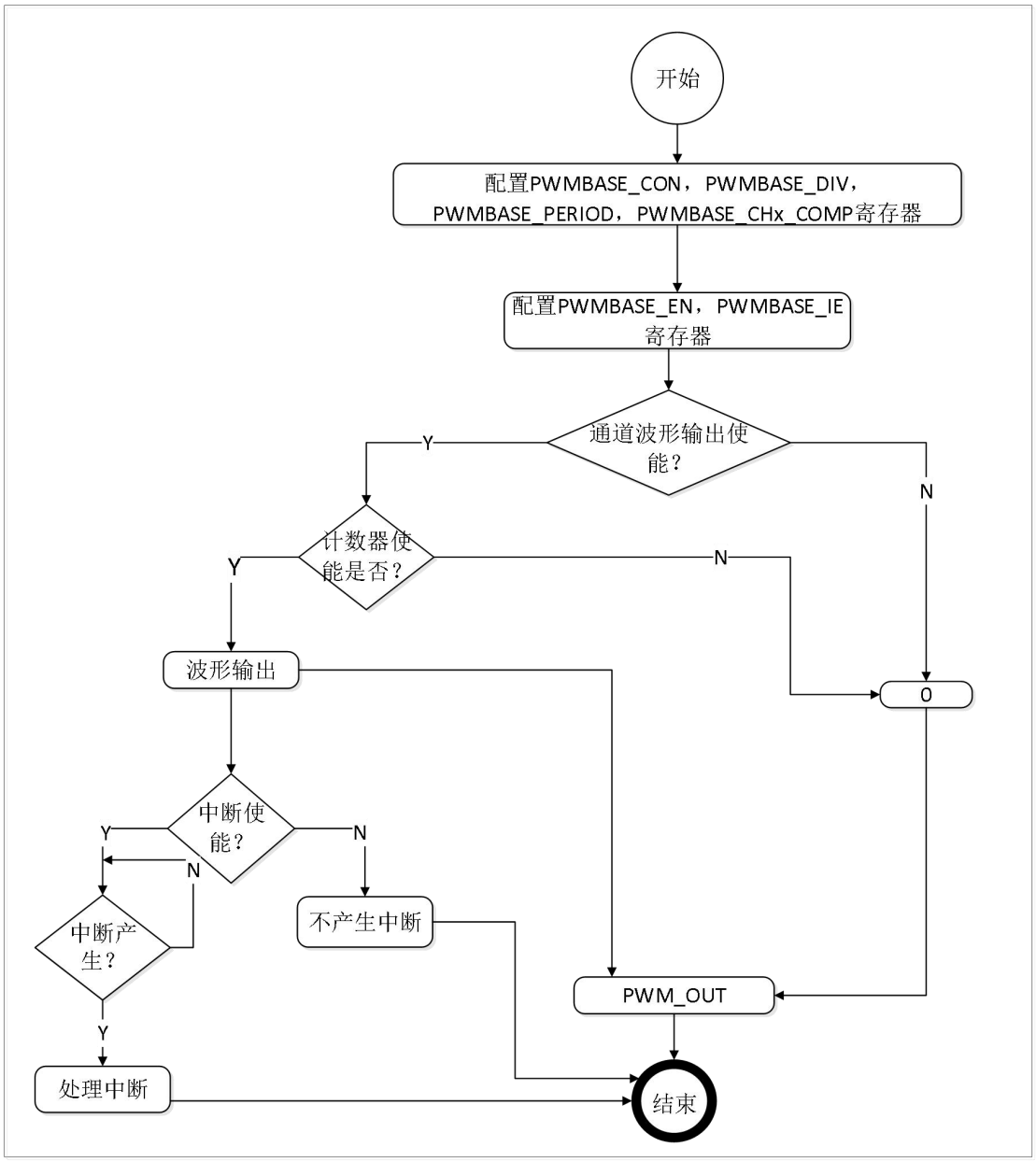


图 5-73 PWMBASE 操作流程图

- 配置 PWMBASE 时钟使能
- PORT 端口配置为 PWMBASE 功能
- 配置预分频（PWMBASE_DIV）寄存器
- 配置输出翻转（PWMBASE_CON）寄存器
- 配置中断使能（PWMBASE_INT_EN）寄存器

- 配置周期（PWMBASEx_PERIOD）和翻转点（PWMBASEx_CHx_COMP）寄存器
- 如果配置了中断，使能 PWMBASE 中断
- 配置波形输出使能（PWMBASE_CON）寄存器
- 配置 PWMBASE 使能位（PWMBASE_EN），开启 PWMBASE

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
PWMBASE0 BASE: 0x400B1000 PWMBASE1 BASE: 0x400B1800					
PWMBASE_EN	0x00	32	R/W	0x00	PWMBASE 使能寄存器
PWMBASE_DIV	0x04	32	R/W	0x00	PWMBASE 时钟预分频寄存器
PWMBASE_CON	0x08	32	R/W	0x00	PWMBASE 输出配置寄存器
PWMBASE_PERIOD	0x0C	32	R/W	0x00	PWMBASE 周期配置寄存器
PWMBASE_IE	0x10	32	R/W	0x00	PWMBASE 中断使能寄存器
PWMBASE_IF	0x14	32	R/W	0x00	PWMBASE 中断状态寄存器
PWMBASE_CNT	0x18	32	R/W	0xffff	PWMBASE 当前计数值寄存器
PWMBASE_CH0_COMP	0x20	32	R/W	0x00	PWMBASE 通道 0 翻转点配置寄存器
PWMBASE_CH1_COMP	0x30	32	R/W	0x00	PWMBASE 通道 1 翻转点配置寄存器
PWMBASE_CH2_COMP	0x40	32	R/W	0x00	PWMBASE 通道 2 翻转点配置寄存器

寄存器描述

PWMBASE_EN 寄存器（0x00）

位域	名称	类型	复位值	描述
31:1	RESERVED	R	0	保留位

0	COUNTER_EN	R/W	0	PWMBASE 计数器使能寄存器 当该位配置为 1，则表示计数器开始计数，CH0/CH1/CH2 按照事先配置好的周期值、比较值产生相应通道的 PWM 输出波形。 当该位配置为 0，则表示停止计数器计数，当计满一个完整计数周期后输出波形电平恢复到初始 0 电平状态

PWMBASE_DIV 寄存器（0x04）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PWMBASE_DIV	R/W	0	PWMBASE 计数时钟预分频寄存器 0x0000：表示 1 分频 0x0001：表示 2 分频 0x0002：表示 3 分频 …… 0xFFFF：表示 65536 分频

PWMBASE_CON 寄存器（0x08）

位域	名称	类型	复位值	描述
31:7	RESERVED	R	0	保留位
6	CH2_OE	R/W	0	CH2 通道波形输出使能 0：输出关闭，管脚上为高阻状态 1：输出 CH2 方波
5	CH1_OE	R/W	0	CH1 通道波形输出使能 0：输出关闭，管脚上为高阻状态 1：输出 CH1 方波
4	CH0_OE	R/W	0	CH0 通道波形输出使能 0：输出关闭，管脚上为高阻状态 1：输出 CH0 方波
3	RESERVED	RO	0	保留位

2	CH2_OUT_INV	R/W	0	CH2 输出极性是否翻转寄存器 0：不改变，CH2 输出波形为原始波形 1：极性翻转，CH2 输出波形为原始波形翻转
1	CH1_OUT_INV	R/W	0	CH1 输出极性是否翻转寄存器 0：不改变，CH1 输出波形为原始波形 1：极性翻转，CH1 输出波形为原始波形翻转
0	CH0_OUT_INV	R/W	0	CH0 输出极性是否翻转寄存器 0：不改变，CH0 输出波形为原始波形 1：极性翻转，CH0 输出波形为原始波形翻转

PWMBASE_PERIOD 寄存器（0x0C）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PWMx_PERIOD	R/W	0	PWMx 输出周期配置寄存器 实际计数周期为此寄存器配置周期值加 1。 注 0：周期不能配置为 0。 例如：配置为十进制 199，则认为 PWM 波形周期为 200。

PWMBASE_INTEN 寄存器（0x10）

位域	名称	类型	复位值	描述
31:4	RESERVED	R	0	保留位
3	POF_IE	R/W	0	周期溢出中断使能
2	CH2_COMP_IE	R/W	0	CH2 到达翻转点中断使能
1	CH1_COMP_IE	R/W	0	CH1 到达翻转点中断使能
0	CH0_COMP_IE	R/W	0	CH0 到达翻转点中断使能

PWMBASE_IF 寄存器（0x14）

位域	名称	类型	复位值	描述
31:4	RESERVED	R	0	保留位
3	POF_IF	R/W	0	周期溢出中断状态　　写 1 清零
2	CH2_COMP_IF	R/W	0	CH2 到达翻转点状态 写 1 清零
1	CH1_COMP_IF	R/W	0	CH1 到达翻转点状态 写 1 清零
0	CH0_COMP_IF	R/W	0	CH0 到达翻转点状态 写 1 清零

PWMBASE_CNT 寄存器（0x18）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	PWMBASE_CNT	R	0xffff	PWMBASE 计数器当前计数值寄存器

PWMBASE_CH0_COMP 寄存器（0x20）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	CH1_COMP	R/W	0	CH0 翻转点配置寄存器 注：计数值小于翻转点值，输出 1；大于等于翻转点值，输出 0。

PWMBASE_CH1_COMP 寄存器（0x30）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位

15:0	CH1_COMP	R/W	0	CH1 翻转点配置寄存器 注：计数值小于翻转点值，输出 1；大于等于翻转点值，输出 0。

PWMBASE_CH2_COMP 寄存器（0x40）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	CH2_COMP	R/W	0	CH2 翻转点配置寄存器 注：计数值小于翻转点值，输出 1；大于等于翻转点值，输出 0。

5.14 高级脉冲宽度调制发生器（PWMPLUS）

5.14.1 概述

PWMPLUS 为高级的 PWM 模块，本芯片的 PWMPLUS 支持死区长度可配、刹车、屏蔽、极性翻转、计数方式和对称方式等多种功能，可以输出灵活的不同占空比的波形，以实现控制外部器件。使用 PWMPLUS 模块之前需要使能 PWMPLUS 时钟。

其系统框架图如下图所示：

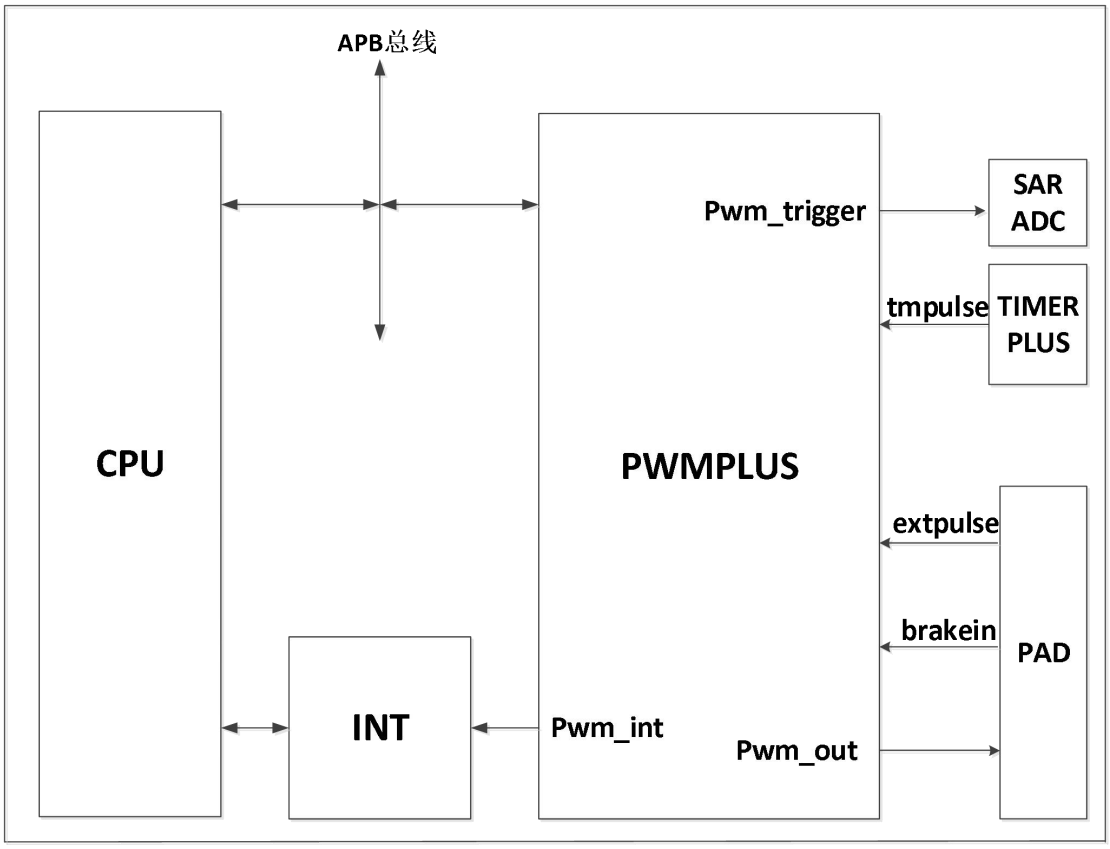


图 5-74 PWMPLUS 系统框图

5.14.2 特性

- 支持 3 个独立通道的 16bit PWM 通道输出（CH0/CH1/CH2），可输出不同占空比的 PWM 波形
- 16bit 预分频计数器，支持片内 timer 或外部信号作为计数时钟功能

- 每个通道支持死区长度可配置
- 每个通道 PWM 都支持自己的互补输出（CH0/CH0N、CH1/CH1N、CH2/CH2N）
- PWM 计数器支持向上计数或向下计数，支持边沿对齐模式或中心对齐模式可配置
- 所有通道支持配置空闲电平、计数起始电平、输出是否翻转
- 支持刹车功能，刹车有效电平可配置
- 支持软件强制输出固定电平功能，电平极性可配置
- 支持内部特定触发机制，可对外输出产生周期结束、通道翻转点、特定触发点三种脉冲信号
- 支持单次或循环方式选择
- 可配置产生边沿对齐波形或中心对齐波形
- 支持周期值、通道翻转点值、特定触发点值固定周期自动加载或软件加载功能

5.14.3模块结构框图

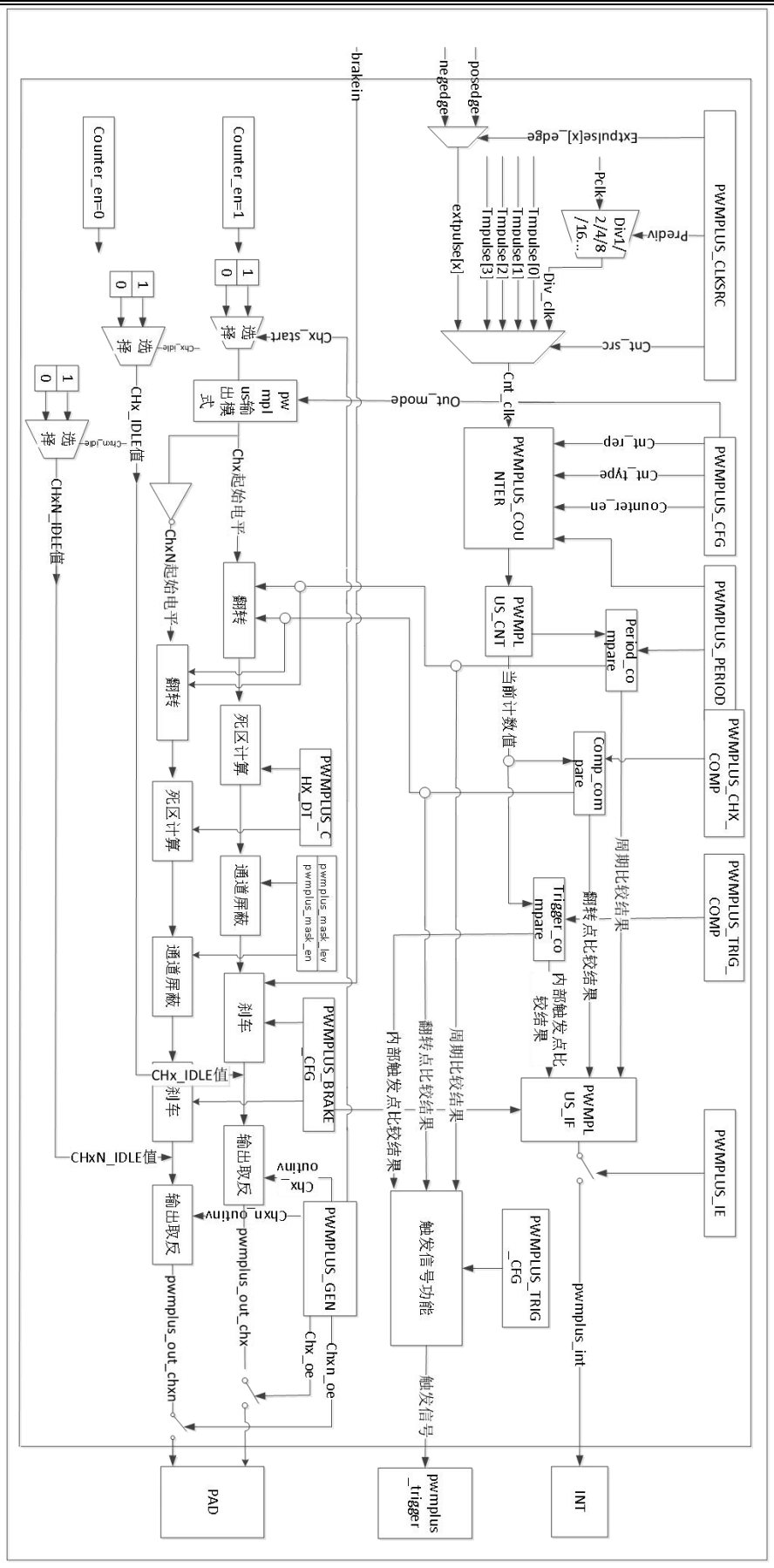

brakein
negedge
posedge
Extpulse[x]_edge
PWMPLUS_CLKSRC
Pclk
Div1/2/4/8/16...
Prediv
extpulse[x]
Tmpulse[0]
Tmpulse[1]
Tmpulse[2]
Tmpulse[3]
Div_clk
Cnt_src
Counter_en=0
Counter_en=1
选择
ChX_start
pwmplus输出模式
Out_mode
Cnt_clk
PWMPLUS_CFG
PWMPLUS_COUNTER
Cnt_rep
Cnt_type
Counter_en
PWMPLUS_PERIOD
PWMPLUS_CNT
Period_compare
当前计数值
PWMPLUS_CHX_COMP
comp_compare
周期比较结果
翻转点比较结果
PWMPLUS_TRIG_COMP
Trigger_compare
内部触发点比较结果
ChX_IDLE值
ChxN_IDLE值
ChxN起始电平
Chx起始电平
翻转
死区计算
PWMPLUS_CHX_DT
通道屏蔽
pwmplus_mask_lev
pwmplus_mask_en
刹车
PWMPLUS_BRAKE_CFG
输出取反
outinv
ChxN_outinv
PWMPLUS_GEN
Chx_oe
Chxn_oe
pwmplus_out_chx
pwmplus_out_chxn
PWMPLUS_IF
PWMPLUS_IE
触发信号功能
PWMPLUS_TRIG_CFG
pwmplus_int
触发信号
PAD
pwmplus_trigger
INT

图 5-75 PWMPLUS 结构框图

如上图所示为 PWMPLUS 模块的结构框架图，其输入信号主要有时钟 pclk，外部刹车输入信号 brakein，外部输入脉冲信号 extPLUS，内部 timer 输入脉冲信号 tmPLUS，其中 pclk，extPLUS，tmPLUS 三个信号都可以选择作为 PWMPLUS 计数时钟使用。输出信号主要有中断信号 pwmplus_int，触发信号 pwmplus_trigger，以及通道波形的输出 pwmplus_out。

本模块中计数时钟有三种可选，分别为内部系统时钟 pclk，外部输入脉冲信号 extPLUS 和内部 timer 输入脉冲信号 tmPLUS，三个时钟可以通过寄存器 pwmplus_clksrc 来配置选择，其中内部时钟 pclk 可以预分频，分频范围为 1-256，而外部输入脉冲信号 extPLUS 则可以选择上升沿有效或下降沿有效。此外，pwmplus 的输出模式，计数器的计数行为方式、计数循环方式和计数使能都可以通过寄存器 pwmplus_cfg 来配置。

通道波形的输出受通道波形使能寄存器的控制，当 chx_oe 为 1 时，向 PAD 的端口输出通道波形，chx_oe 为 0 时向 PAD 的端口输出高阻态。通道波形的输出分为空闲状态输出和通道方波的输出。当计数使能 counter_en 为 0 时 pwm 输出空闲状态 idle 值，idle 值可以通过寄存器 pwmplus_gen 配置为 1 或 0，也可以进一步控制其输出时是否翻转。当计数使能 counter_en 为 1 时 pwm 输出通道方波，而 PWMPLUS 的输出方式可以通过寄存器 pwmplus_cfg 来配置，通道方波的起始电压也可配为 1 或 0。然后，可以通过软件配置的计数周期值，翻转点值与当前计数值的比较来控制波形的翻转，还可以配置自动加载寄存器，表示周期溢出多少次后自动装载一次周期值、比较值、死区值和 TRIGGER 值。之后通过配置死区长度，通道屏蔽，刹车和翻转来产生理想的波形。

中断信号的产生受中断使能和中断状态的控制，在中断使能 pwmplus_ie 为 1 时，在相应的中断状态产生后会产生相应的中断信号，当中断使能 pwmplus_ie 为 0 时，即使有中断状态也不会产生中断信号。触发信号受内部触发配置寄存器 pwmplus_trig_cfg 的控制，可以配置触发信号的功能选择，只有配置相应的触发功能且有相应的触发状态后才会产生触发信号，并且触发信号为一个时钟周期的高电平。

本模块还支持周期值、通道翻转点值、特定触发点值固定周期自动加载或软件加载功能，其中自动装载功能是表示周期溢出多少次后自动装载一次周期值、比较值、死区值和 TRIGGER 值，通过自动装载寄存器 auto_reload 来配置周期溢出次数(auto_reload+1 次)；软件加载功能可以通过 pwmplus 配置寄存器软件 load 控制位来实现。本模块中 pwm_period、

pwm_ch0_comp、pwm_ch1_comp、pwm_ch2_comp、pwm_ch0_dt、pwm_ch1_dt、pwm_ch2_dt、trig_comp 这 8 个寄存器具有各自的影子寄存器。软件向该位写 1 后，硬件在本周期结束后将这 8 个寄存器中保存的最新值 load 到各自的影子寄存器中，并在接下来的周期生效。本模块还可以通过计数器工作状态寄存器读取计数器的工作状态，通过刹车输入信号状态寄存器读取刹车输入信号的当前状态。

5.14.4 功能描述

边沿对齐模式输出

本模块计数器的 PWM 输出模式可配置为边沿对齐模式和中心对齐模式两种，首先在边沿对齐输出模式下，计数器向上或向下计数，单次输出，循环输出和不同起始电平等情况下的时序示意图如下所示。

- 计数器向上计数

向上计数即计数器从零开始计数，每计数时钟周期加一，一直加到配置的周期值。以下图中 chx，chxn，chx_inv，chxn_inv 和灰色部分表示的含义都如下：chx 表示输出状态寄存器 chx_outinv 为 0 时，chx 通道的 pwmplus 输出波形，chx_inv 表示输出状态寄存器 chx_outinv 为 1 时，chx 通道的 pwmplus 输出波形，此时通道的输出波形与通道的原始波形相反，chxn 和 chxn_inv 表示其互补输出。图中灰色部分代表死区，且死区长度可配。

1、单次输出，start=0 的情况

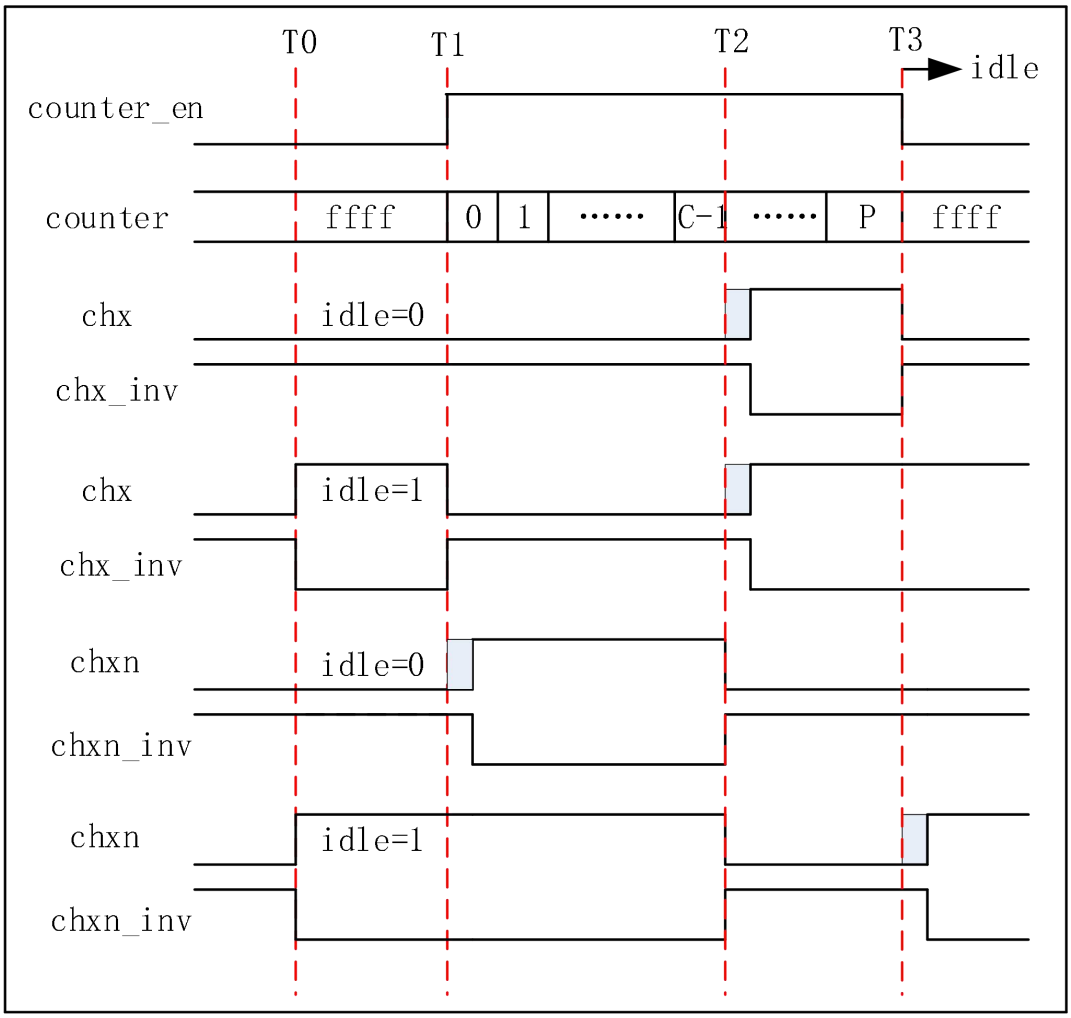


图 5-76 PWMPLUS 边沿对齐模式下，向上计数，单次输出，起始电平为 0 的时序图

如上图所示，在打开计数使能 counter_en 使能之前（T1 时刻之前），pwmplus 输出 IDLE 值，当 T0 时刻改变 IDLE 值时 pwmplus 输出随 IDLE 值的变化而变化，当 T1 时刻打开计数器计数使能，pwmplus 输出通道波形。当配置相应的中断使能后，在 T2 时刻产生翻转点中断，在 T3 时刻产生周期溢出中断。

2、单次输出，start=1 的情况

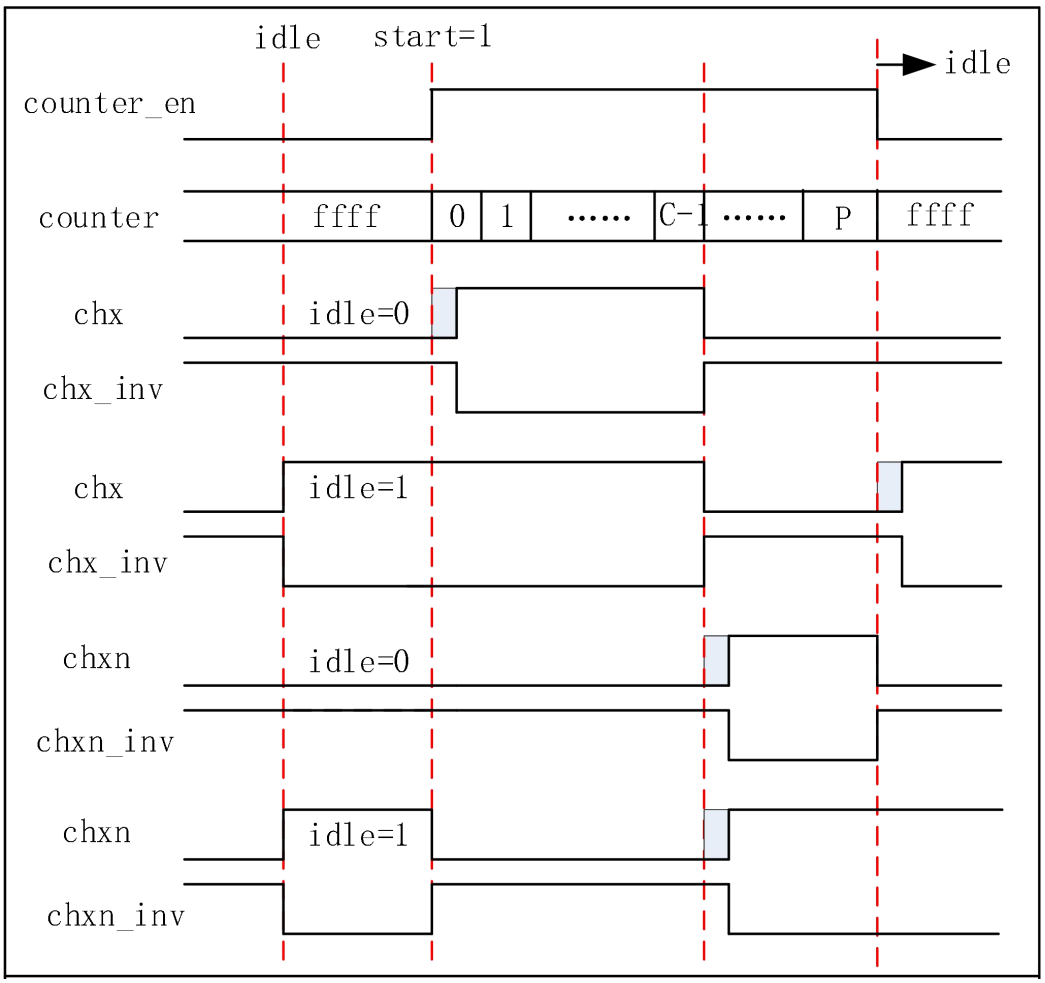


图 5-77 PWMPLUS 边沿对齐模式下，向上计数，单次输出，起始电平为 1 的时序图

如上图所示，配置翻转点中断使能和周期结束中断使能，可在第三个竖线处产生翻转点中断，在第四个竖线处产生周期结束中断。

3、循环输出，start=0 的情况

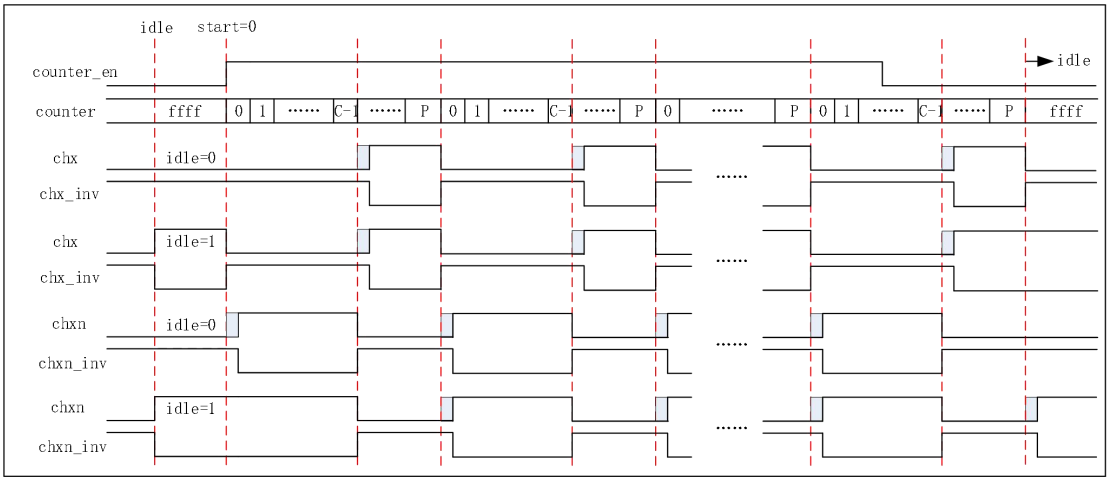


图 5-78 PWMPLUS 边沿对齐模式下，向上计数，循环输出，起始电平为 0 的时序图

如上图所示，配置翻转点中断使能和周期结束中断使能，可在每个“C-1”的位置都可产生翻转点中断，在每个“P”的位置都可产生周期结束中断。

4、循环输出，start=1 的情况

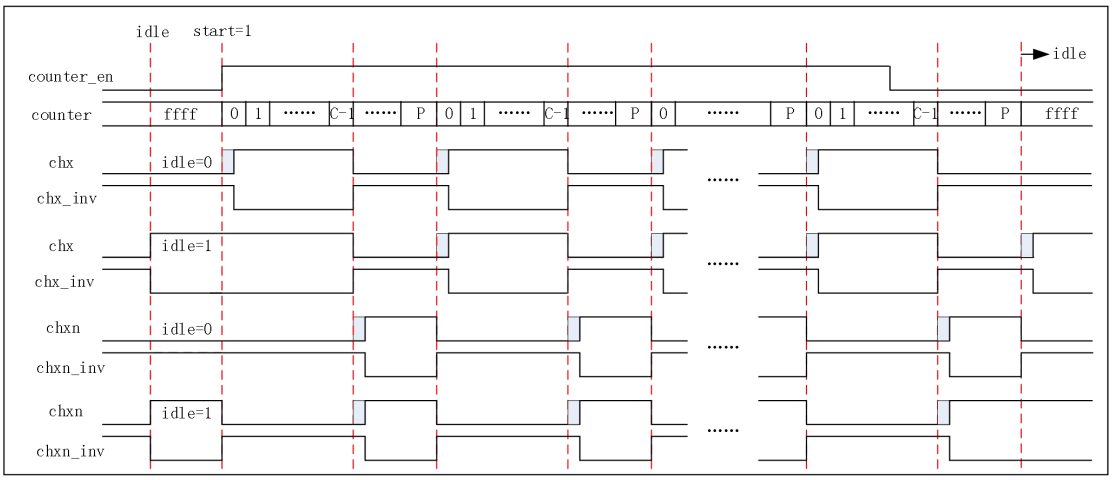


图 5-79 PWMPLUS 边沿对齐模式下，向上计数，循环输出，起始电平为 1 的时序图

如上图所示，配置翻转点中断使能和周期结束中断使能，可在每个“C-1”的位置都可产生翻转点中断，在每个“P”的位置都可产生周期结束中断。

- 计数器向下计数

向下计数即计数器从配置的周期值开始计数，每个计数时钟周期减一，一直减到零。以下图中 chx，chxn，chx_inv，chxn_inv 和灰色部分表示的含义都如下：chx 表示输出状态寄存器 chx_outinv 为 0 时，chx 通道的 pwm 输出波形，chx_inv 表示输出状态寄存器 chx_outinv 为 1 时，chx 通道的 pwmplus 输出波形，此时通道的输出波形与通道的原始波形相反，chxn 和 chxn_inv 表示其互补输出。图中灰色部分代表死区，且死区长度可配。

1、 单次输出，start=0 的情况

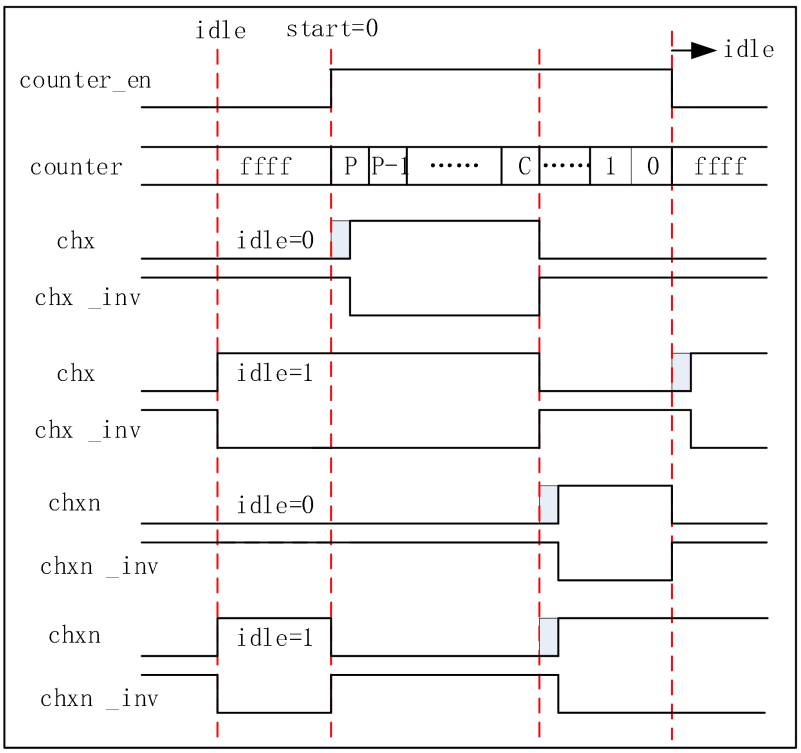


图 5-80 PWMPLUS 边沿对齐模式下，向下计数，单次输出，起始电平为 0 的时序图

2、 单次输出，start=1 的情况

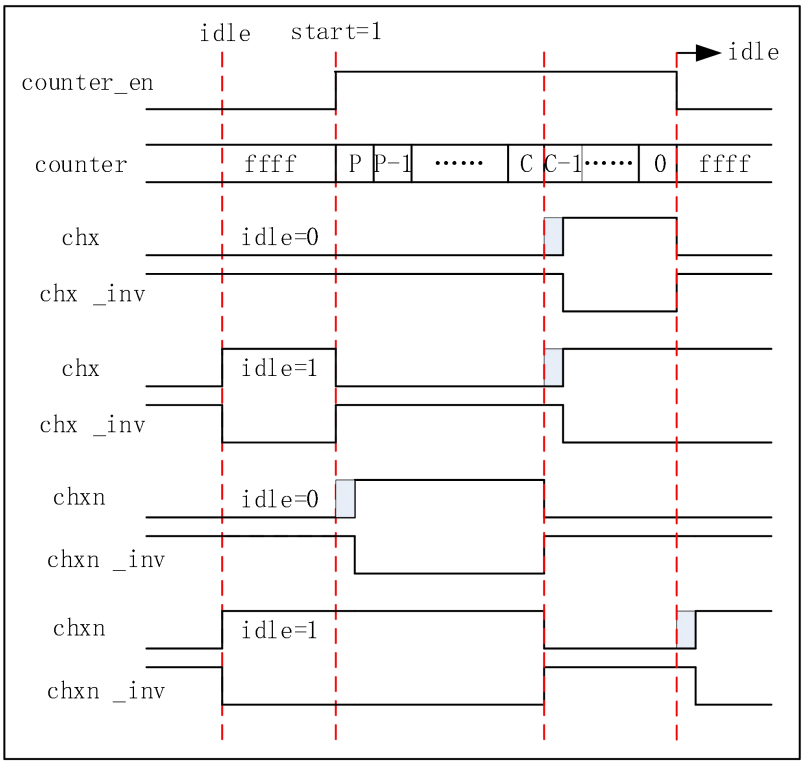


图 5-81 PWMPLUS 边沿对齐模式下，向下计数，单次输出，起始电平为 1 的时序图

3、 循环输出，start=0 的情况

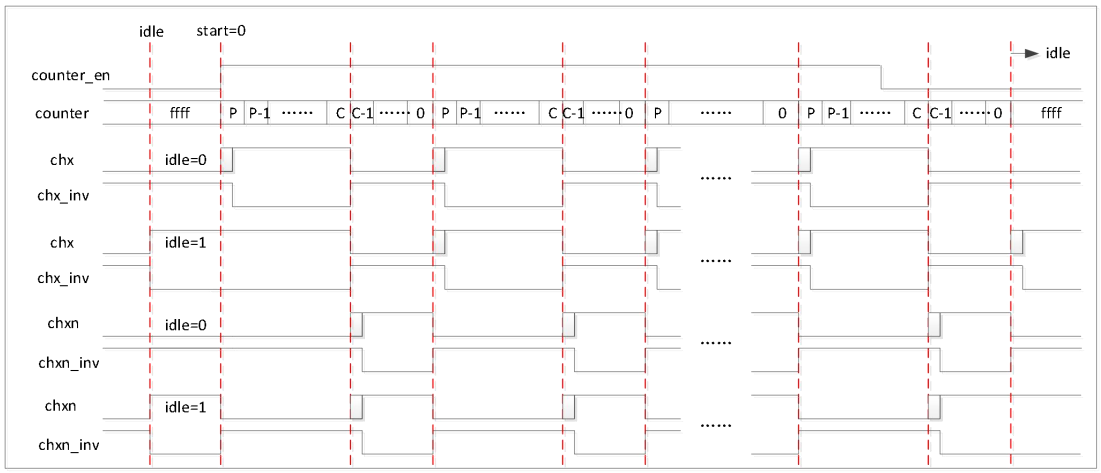


图 5-82 PWMPLUS 边沿对齐模式下，向下计数，循环输出，起始电平为 0 的时序图

4、 循环输出，start=1 的情况

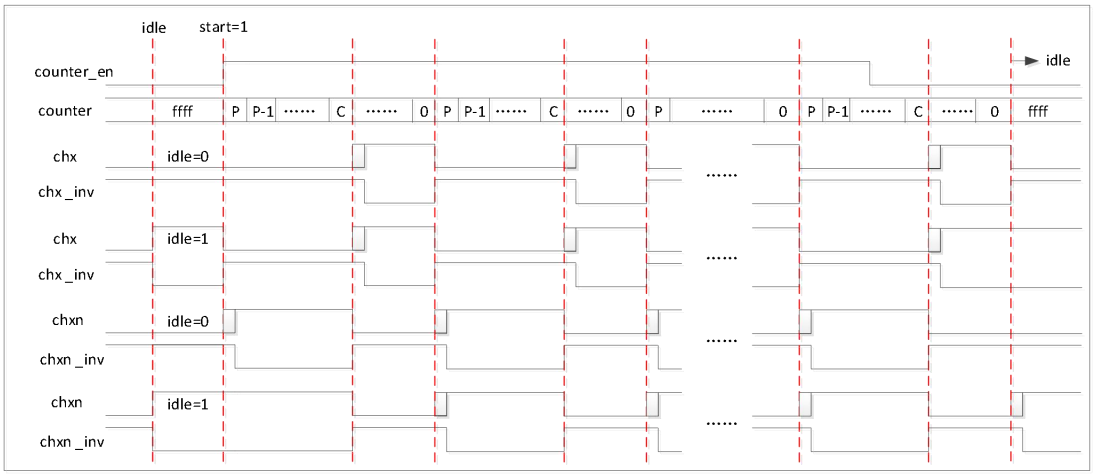


图 5-83 PWMPLUS 边沿对齐模式下，向下计数，循环输出，起始电平为 1 的时序图

中心对齐模式输出

在中心对齐模式下，计数器向上或向下计数，单次输出，循环输出和不同起始电平等情况下的时序示意图如下所示：

- 计数器向上计数

向上计数即计数器从零开始计数，每个计数时钟周期加一，一直加到配置的周期值。以下图中 chx，chxn，chx_inv，chxn_inv 和灰色部分表示的含义都如下：chx 表示输出状态寄存

器 chx_outinv 为 0 时，chx 通道的 pwmplus 输出波形，chx_inv 表示输出状态寄存器 chx_outinv 为 1 时，chx 通道的 pwmplus 输出波形，此时通道的输出波形与通道的原始波形相反，chxn 和 chxn_inv 表示其互补输出。图中灰色部分代表死区，且死区长度可配

1、单次输出，start=0 的情况

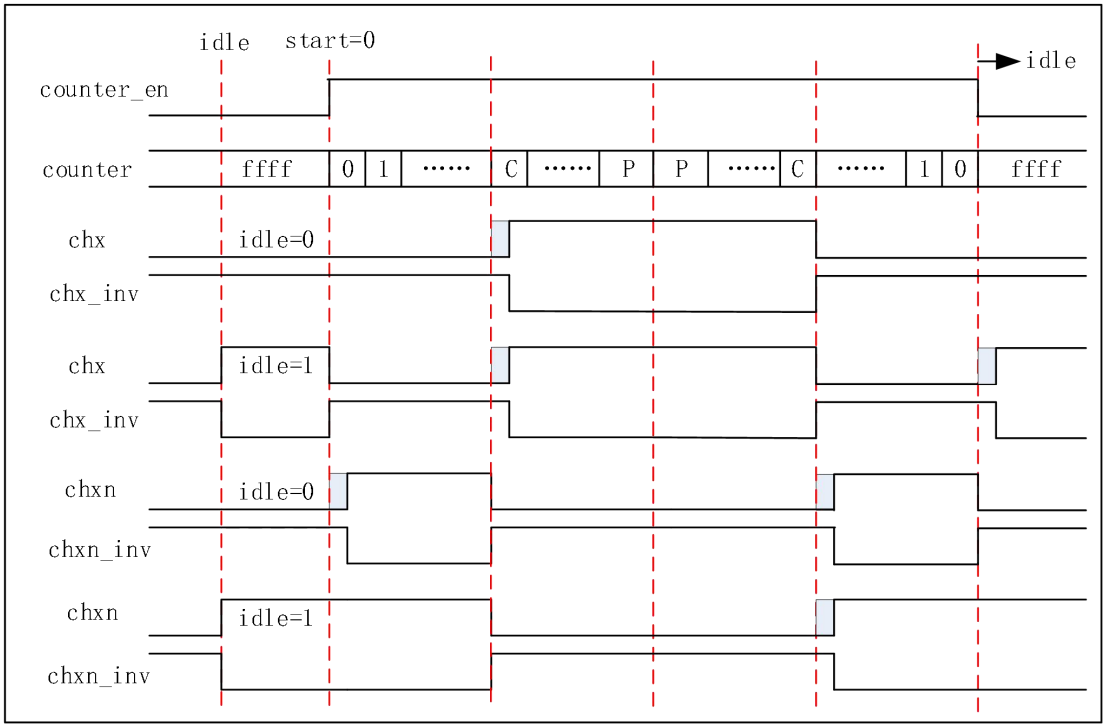


图 5-84 PWMPLUS 中心对齐模式下，向上计数，单次输出，起始电平为 0 的时序图

从上图中可以看出，中心对齐模式下 pwmplus 输出波形以 p 点结束的位置为中心对齐，上半周期为向上计数，下半周期为向下计数。可配置在第三和第五个竖线处产生翻转点中断，在第四和第六个竖线处产生周期结束中断。

2、单次输出，start=1 的情况

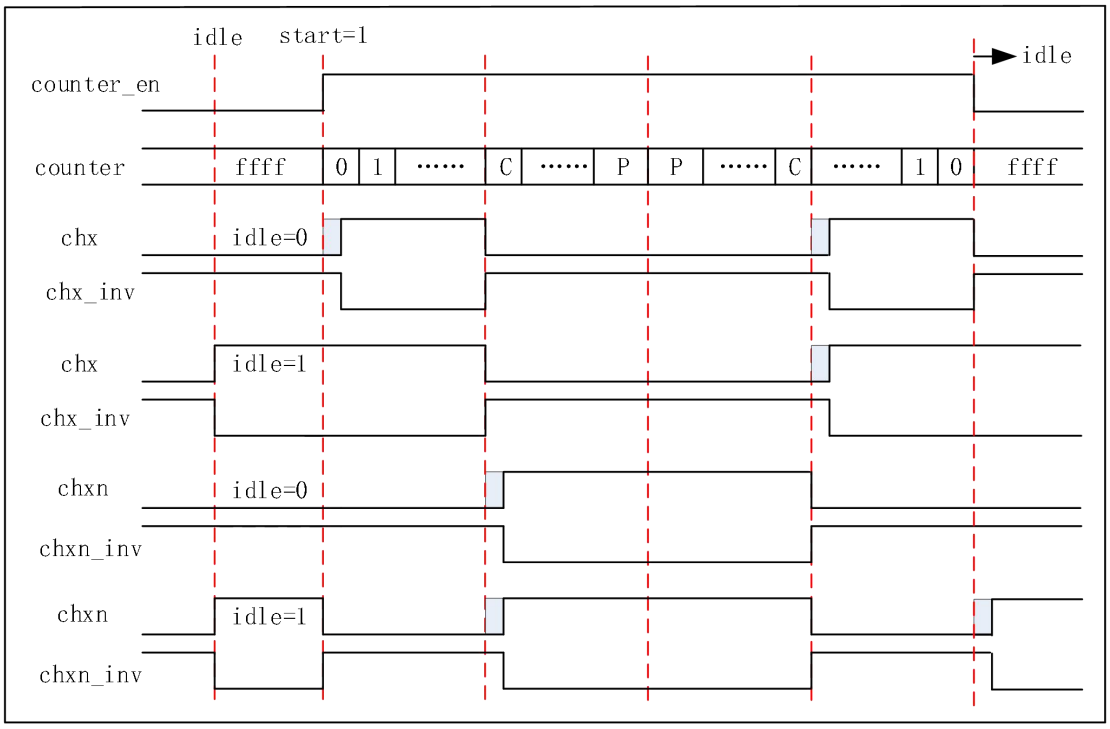


图 5-85 PWMPLUS 中心对齐模式下，向上计数，单次输出，起始电平为 1 的时序图

如上图所示，配置翻转点中断使能和周期结束中断使能可在第三和第五个竖线处产生翻转点中断，在第四和第六个竖线处产生周期结束中断。

3、循环输出，start=0 的情况

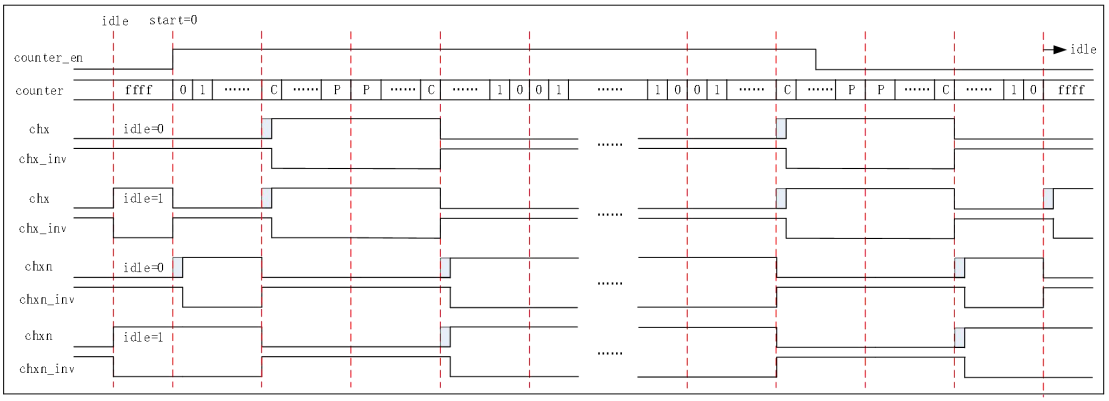


图 5-86 PWMPLUS 中心对齐模式下，向上计数，循环输出，起始电平为 0 的时序图

如上图所示，配置翻转点中断使能和周期结束中断使能，可在每个“C”的位置都可产生翻转点中断，在每两个“P”的位置或每两个“0”的位置都可产生周期结束中断。

4、循环输出，start=1 的情况

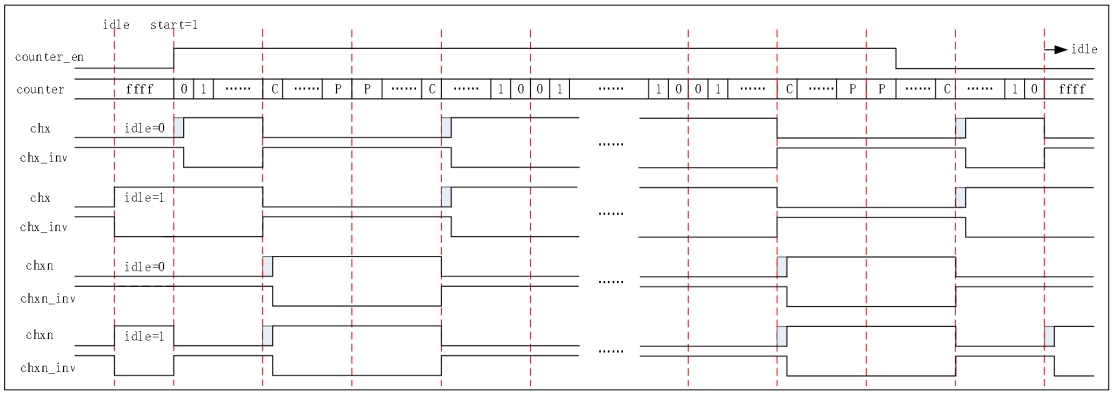


图 5-87 PWMPLUS 中心对齐模式下，向上计数，循环输出，起始电平为 0 的时序图

如上图所示，配置翻转点中断使能和周期结束中断使能可在每个“C”的位置都可产生翻转点中断，在每两个“P”的位置或每两个“0”的位置都可产生周期结束中断。

- 计数器向下计数

向下计数即计数器从配置的周期值开始计数，每个计数时钟周期减一，一直减到零。以下图中 chx，chxn，chx_inv，chxn_inv 和灰色部分表示的含义都如下：chx 表示输出状态寄存器 chx_outinv 为 0 时，chx 通道的 pwmplus 输出波形，chx_inv 表示输出状态寄存器 chx_outinv 为 1 时，chx 通道的 pwmplus 输出波形，此时通道的输出波形与通道的原始波形相反，chxn 和 chxn_inv 表示其互补输出。图中灰色部分代表死区，且死区长度可配

1、 单次输出，start=0 的情况

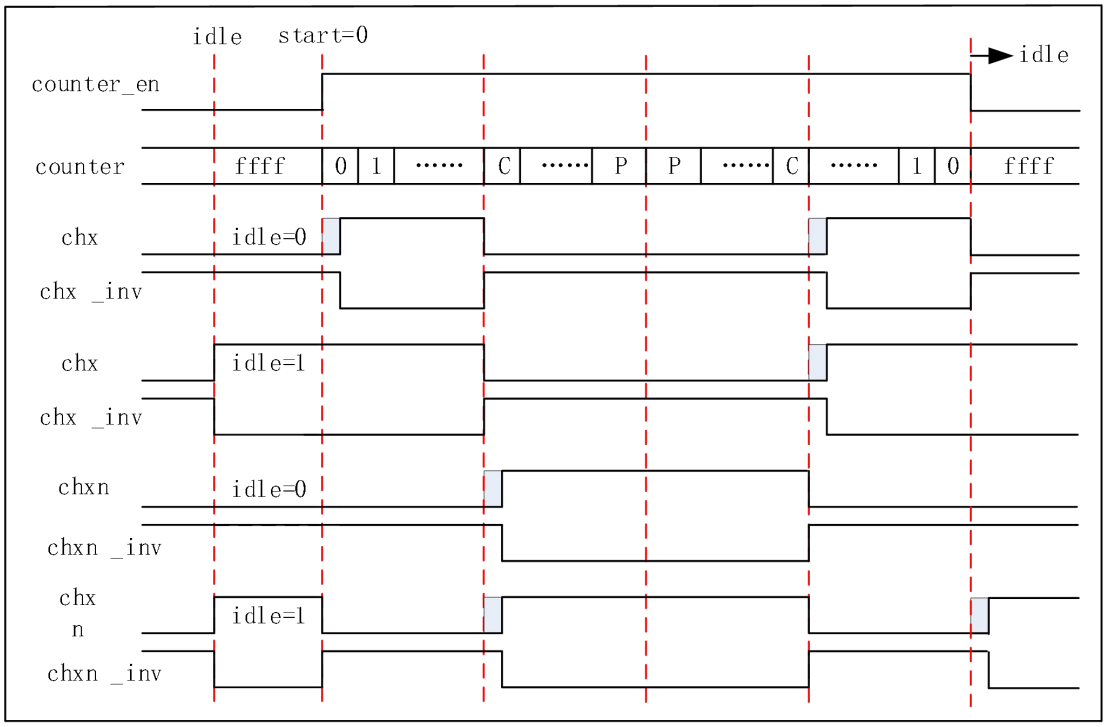


图 5-88 PWMPLUS 中心对齐模式下，向下计数，单次输出，起始电平为 0 的时序图

2、 单次输出，start=1 的情况

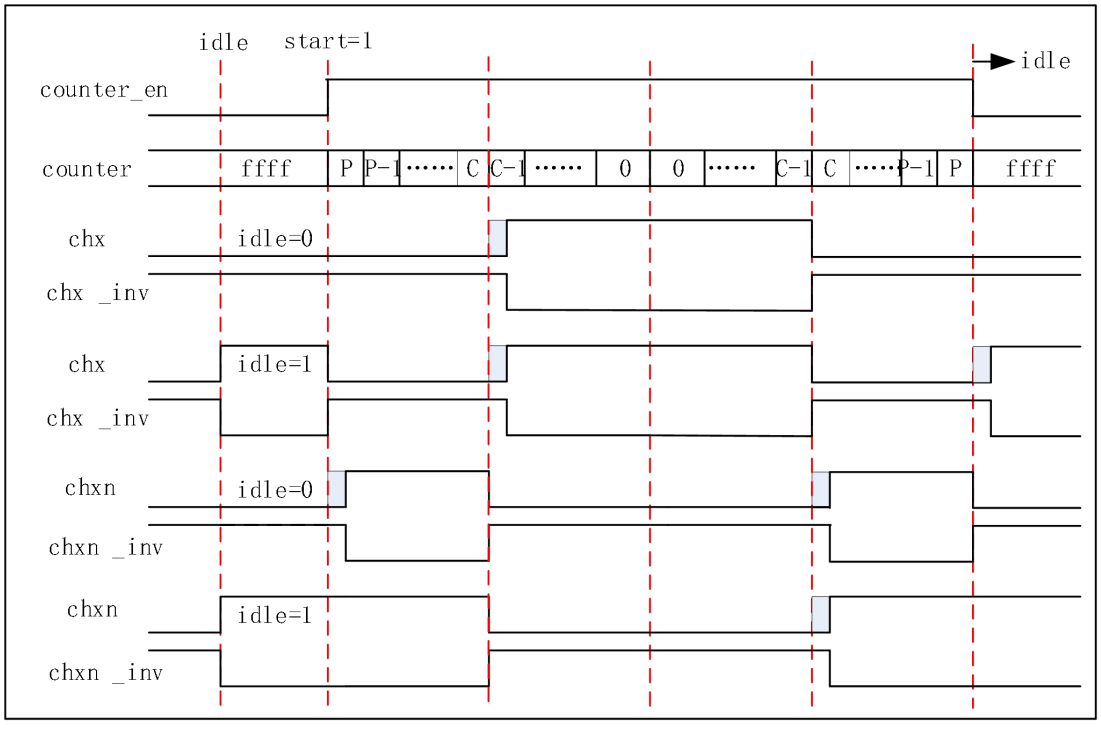


图 5-89 PWMPLUS 中心对齐模式下，向下计数，单次输出，起始电平为 1 的时序图

3、 循环输出，start=0 的情况

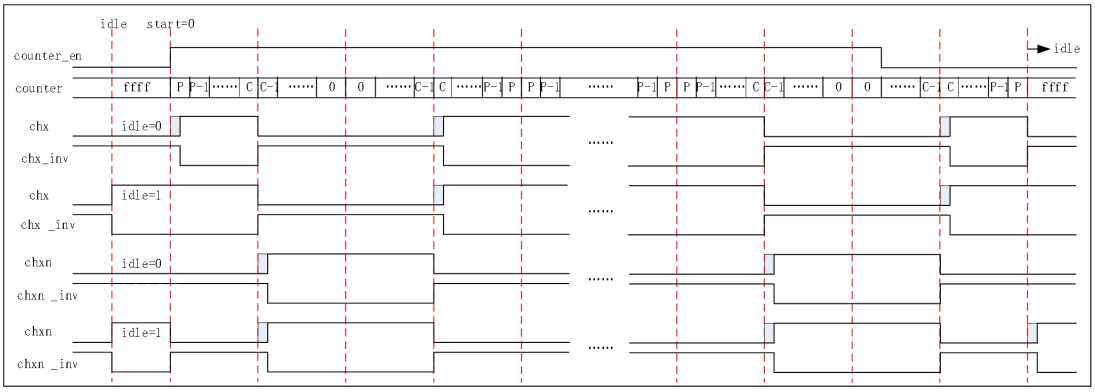


图 5-90 PWMPLUS 中心对齐模式下，向下计数，循环输出，起始电平为 0 的时序图

4、 循环输出，start=1 的情况

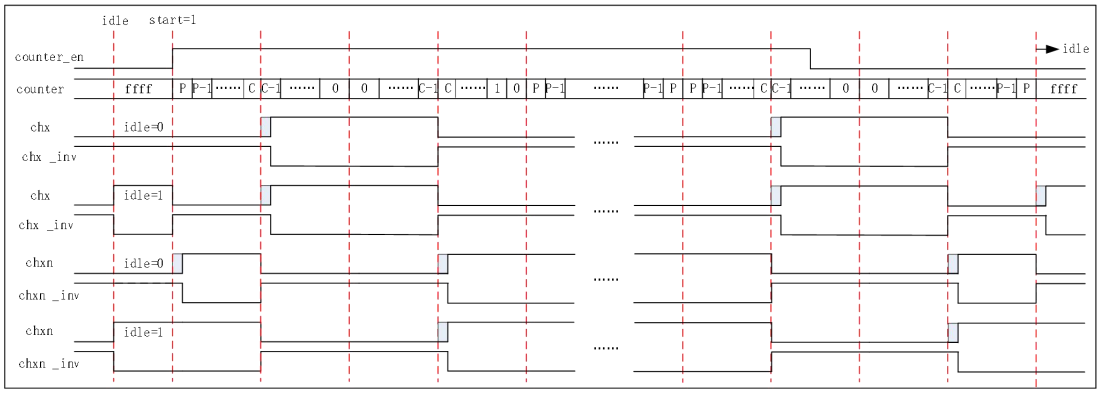


图 5-91 PWMPLUS 中心对齐模式下，向下计数，循环输出，起始电平为 1 的时序图

带死区插入的互补输出

PWMPLUS 能够输出两路互补信号， 并且能够管理输出的瞬时关断和接通，这段时间通常被称为死区， 用户应该根据连接的输出器件和它们的特性（ 电平转换的延时、电源开关的延时等 ）来调整死区时间。

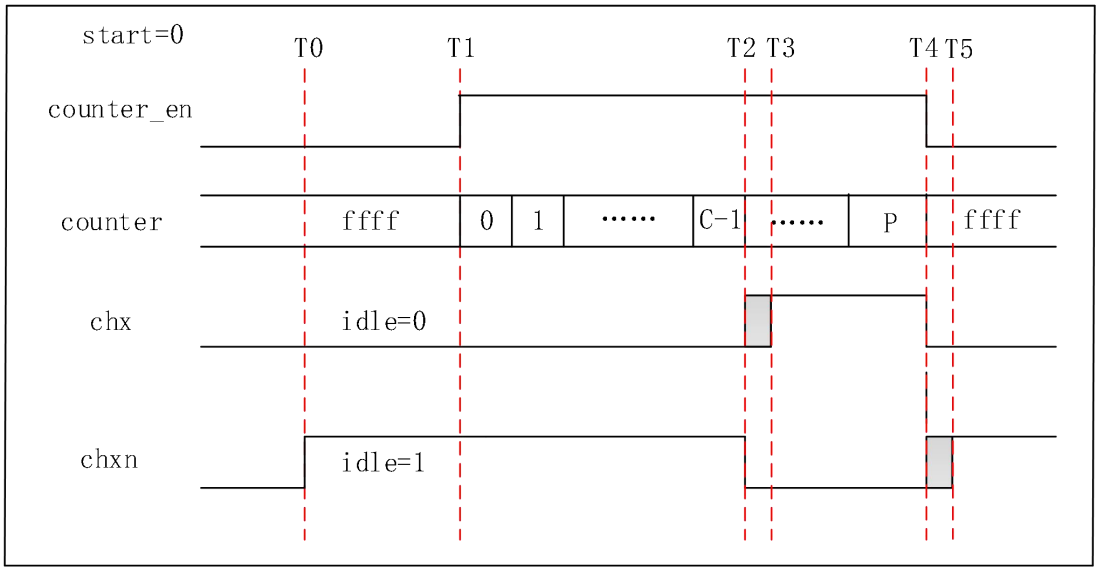


图 5-92　PWMPLUS 带死区插入的互补输出时序图

上图为 PWMPLUS 带死区插入的互补输出，其中 chxn 为 chx 的互补输出，图中 T2-T3 和 T4-T5 间的灰色部分为死区，死区的长度可以通过 CHx_DT 寄存器来配置，配置死区长度时需要与周期值、CHx_START 值、CHx_COMP 值相匹配，否则输出波形可能达不到预期效果。

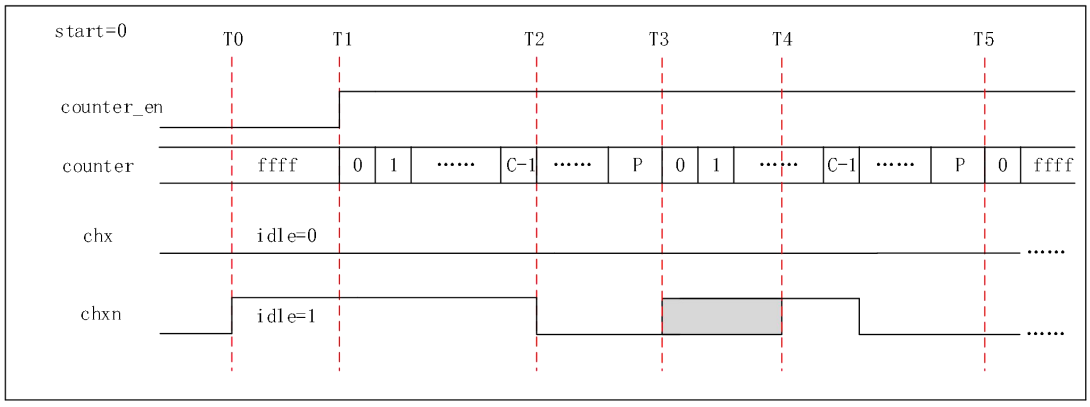


图 5-93　PWMPLUS 死区长度大于正脉冲小于负脉冲时序图

上图为 PWMPLUS 起始电平为 0 时，当配置的死区长度大于正脉冲小于负脉冲的时序图，图中 T3-T4 灰色部分为死区长度。

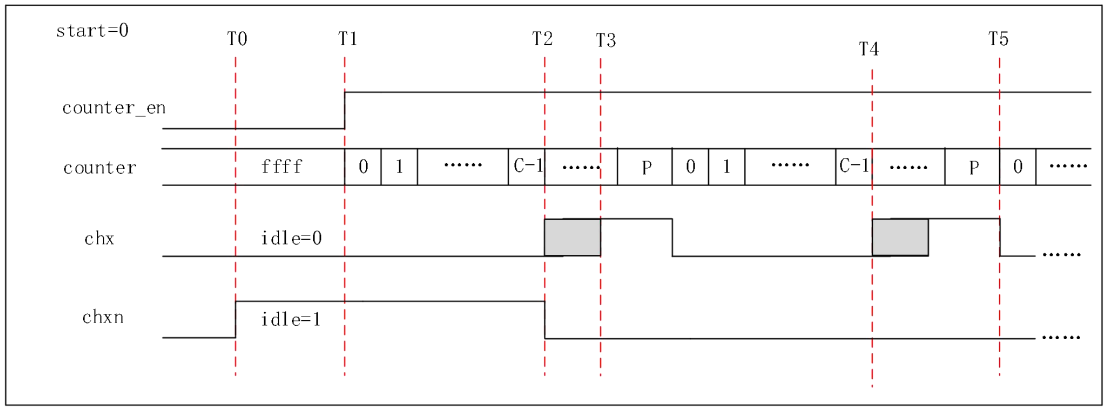


图 5-94　PWMPLUS 死区长度大于负脉冲小于正脉冲时序图

上图为 PWMPLUS 起始电平为 0 时，当配置的死区长度大于正脉冲小于负脉冲的时序图，图中 T2-T3 灰色部分为死区长度。

刹车情况下输出

本模块支持不同通道选择不同刹车信号、有效刹车电平可配置、各通道刹车期间电平值可配置。刹车期间，计数器正常计数，不受刹车影响。

刹车信号到来后，对通道输出信号立刻有效，刹车撤销后，在下一个周期开始才会输出正常波形。

以计数器向上计数、起始电平为 0 时的情况为例：

1、边沿对齐模式

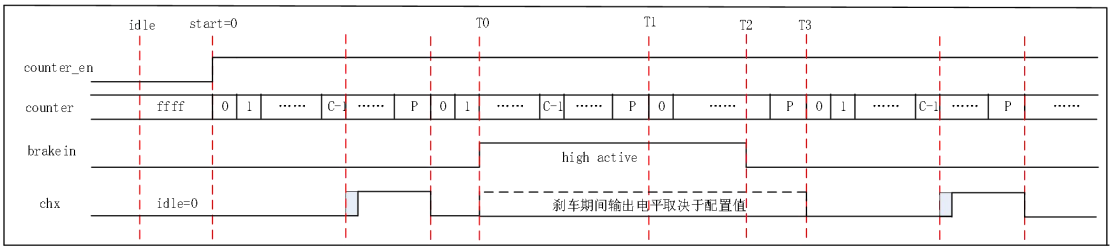


图 5-95　PWMPLUS 边沿对齐模式刹车情况时序图

如上图所示，在边沿对齐模式下，以高有效刹车电平为例，在 T0 时刻输入的刹车电平变为高，信号立刻有效，进入刹车期间，刹车期间各通道不在输出正常波形，如图中 T1 时刻输出波形并不会翻转，而通道波形输出取决于配置值；当 T2 时刻刹车结束后，通道波形并没有立刻恢复正常，而是在下一个周期开始（T3 时刻）才会输出正常波形。

2、中心对齐模式

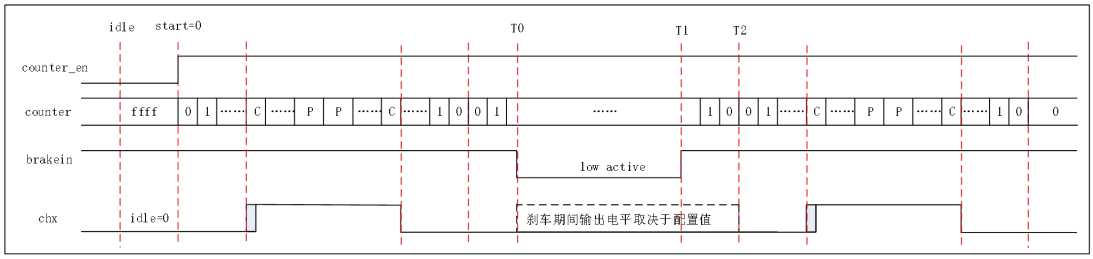


图 5-96 PWMPLUS 中心对齐模式刹车情况时序图

如上图所示，在中心对齐模式下，以低有效刹车电平为例，在 T0 时刻输入的刹车电平变为低，信号立刻有效，进入刹车期间，刹车期间各通道不在输出正常波形，而通道波形输出取决于配置值；当 T1 时刻刹车结束后，通道波形并没有立刻恢复正常，而是在下一个周期开始（T2 时刻）才会输出正常波形。

此外本芯片还支持刹车滤波功能，可以对刹车信号数字滤波控制，可进行 2、4 和 8 个内部预分频时钟滤波，通过寄存器 BRAKE_FILTER 控制。

MASK 情况下输出

本模块支持对各通道独立 MASK 功能配置、各通道 MASK 期间电平值可配置。MASK 期间，计数器正常计数，不受影响。

MASK 功能有效后，对通道输出信号立刻有效，MASK 功能撤销后，立刻输出正常波形。以计数器向上计数、起始电平为 0 时的情况为例：

1、边沿对齐模式

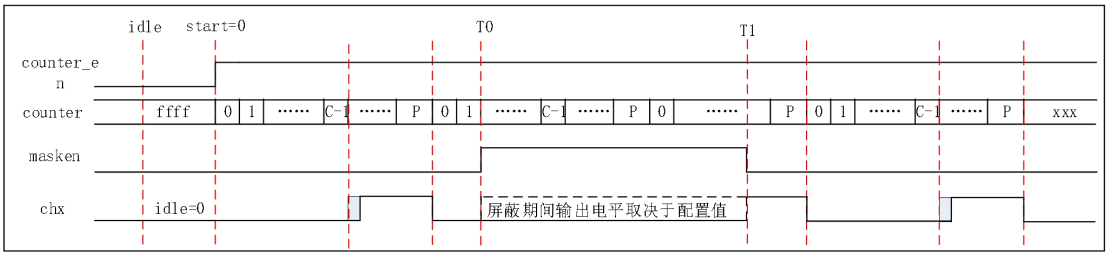


图 5-97 PWMPLUS 边沿对齐模式 MASK 情况时序图

如上图所示，在边沿对齐模式下，在 T0 时刻屏蔽使能变为高，屏蔽使能立刻有效，进入屏蔽期间，屏蔽期间各通道不在输出正常波形，而通道波形输出取决于配置值；当 T1 时刻屏蔽结束后，通道波形立刻恢复正常，输出正常波形。

2、中心对齐模式

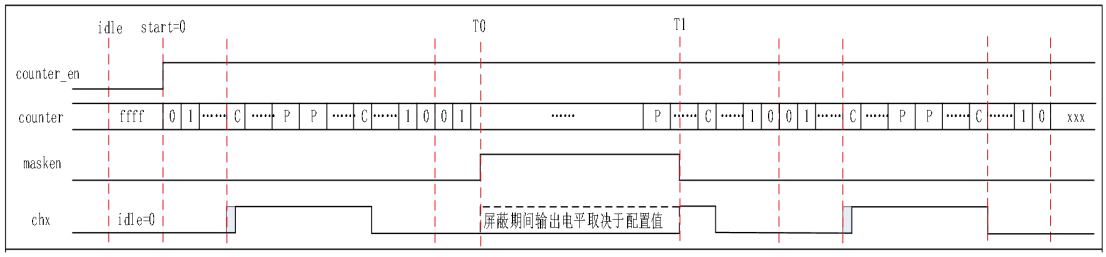


图 5-98 PWMPLUS 中心对齐模式 MASK 情况时序图

如上图所示，在中心对齐模式下，在 T0 时刻屏蔽使能变为高，屏蔽使能立刻有效，进入屏蔽期间，屏蔽期间各通道不在输出正常波形，而通道波形输出取决于配置值；当 T1 时刻屏蔽结束后，通道波形立刻恢复正常，输出正常波形。

周期值与翻转点

配置周期值与翻转点值的关系举例（IDLE 为 0）：

用户想要产生的 PWMPLUS 波形为：周期 8，计数起始电平为 0，占空比为 75%，则需要配置为：PERIOD=0x7，CHx_COMP=0x2。

用户想要产生的 PWMPLUS 波形为：周期 8，计数起始电平为 0，占空比为 12.5%，则需要配置为：PERIOD=0x7，CHx_COMP=0x7。

用户想要产生的 PWMPLUS 波形为：周期 8，计数起始电平为 0，占空比为 0%，则需要配置为：PERIOD=0x7，CHx_COMP=0x8（大于 7 即可）。

用户想要产生的 PWMPLUS 波形为：周期 8，计数起始电平为 0，占空比为 100%，则需要配置为：PERIOD=0x7，CHx_COMP=0x0。

用户想要产生的 PWMPLUS 波形为：周期 8，计数起始电平为 1，占空比为 25%，则需要配置为：PERIOD=0x7，CHx_COMP=0x2。

用户想要产生的 PWMPLUS 波形为：周期 8，计数起始电平为 1，占空比为 87.5%，则需要配置为：PERIOD=0x7，CHx_COMP=0x7。

用户想要产生的 PWMPLUS 波形为：周期 8，计数起始电平为 1，占空比为 100%，则需要配置为：PERIOD=0x7，CHx_COMP=0x8（大于 7 即可）。

用户想要产生的 PWMPLUS 波形为：周期 8，计数起始电平为 1，占空比为 0%，则需要配置为：PERIOD=0x7，CHx_COMP=0x0。

以下示意图实例中周期值配置为 7。

1、边沿对齐模式，向上计数情况：

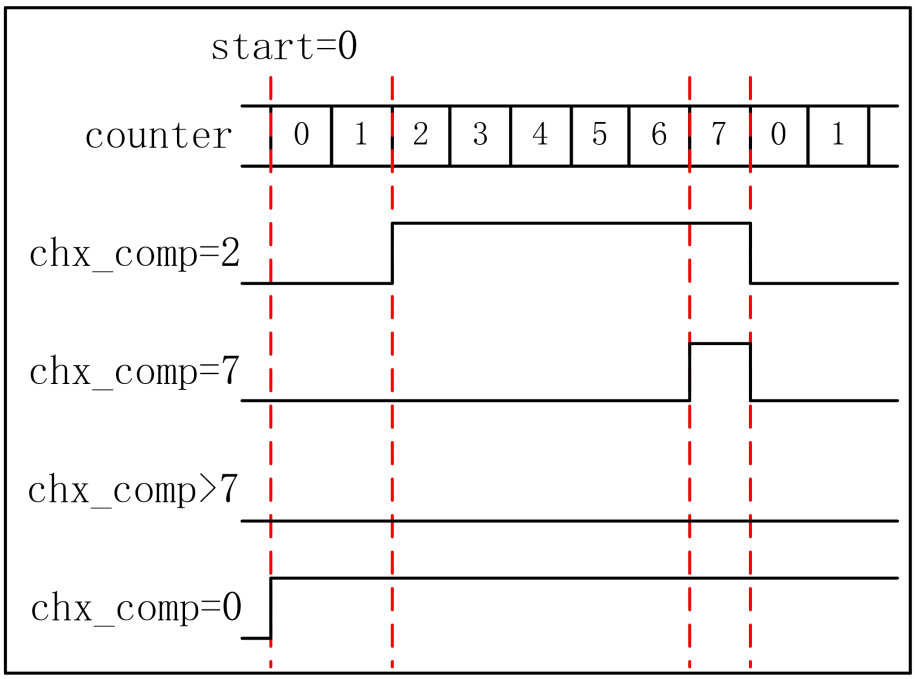


图 5-99 PWMPLUS 边沿对齐模式，向上计数，起始电平为 0 情况下翻转点和周期时序图

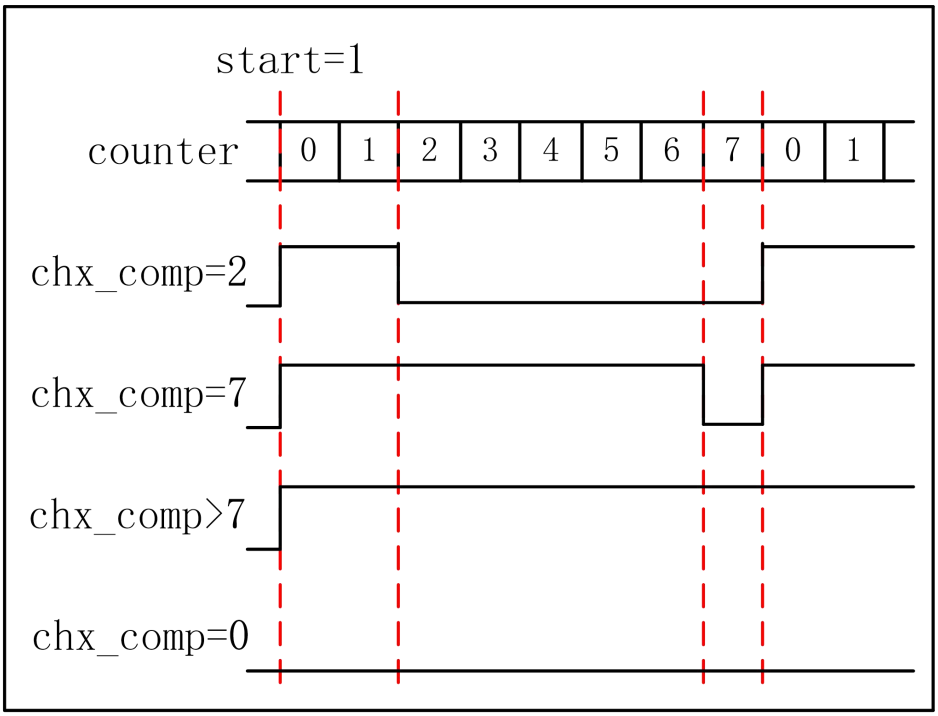


图 5-100 PWMPLUS 边沿对齐模式，向上计数，起始电平为 1 情况下翻转点和周期时序图

2、边沿对齐模式，向下计数情况：

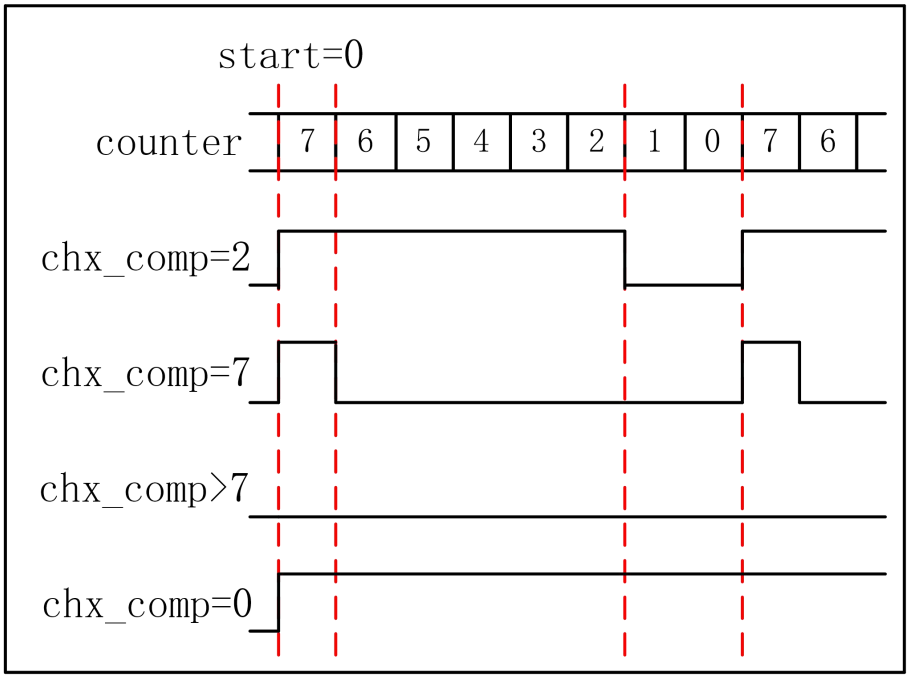


图 5-101 PWMPLUS 边沿对齐模式，向下计数，起始电平为 0 情况下翻转点和周期时序图

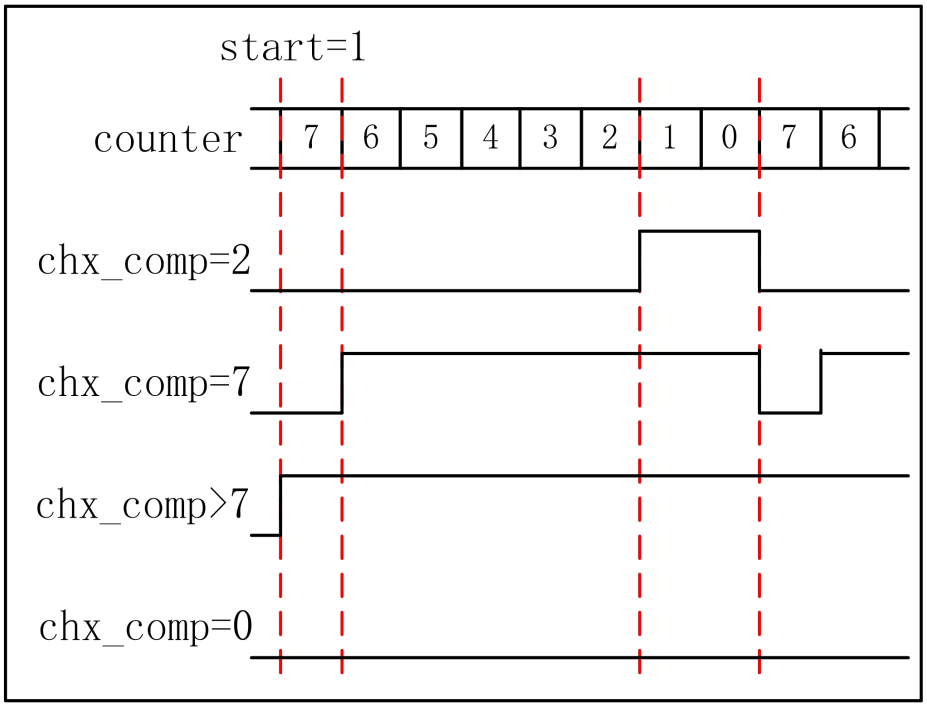


图 5-102 PWMPLUS 边沿对齐模式，向下计数，起始电平为 1 情况下翻转点和周期时序图

3、中心对齐模式，向上计数情况：

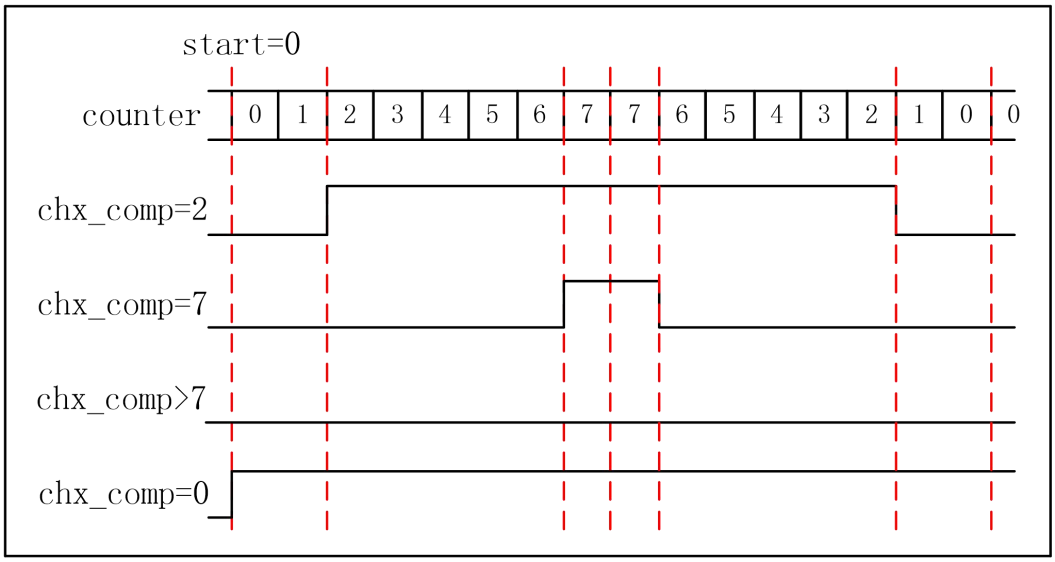


图 5-103 PWMPLUS 中心对齐模式，向上计数，起始电平为 0 情况下翻转点和周期时序图

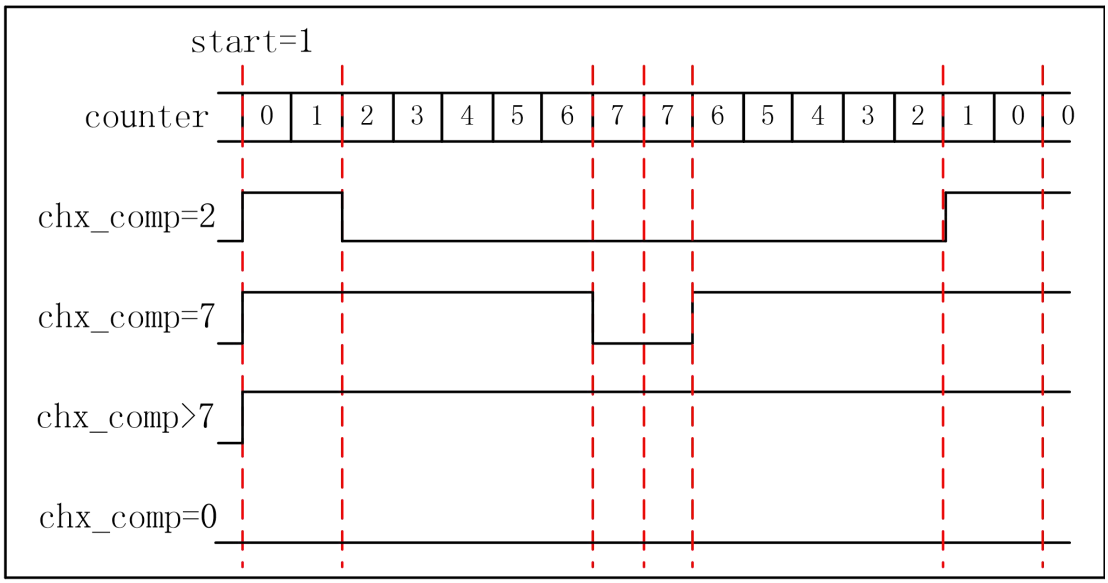


图 5-104 PWMPLUS 中心对齐模式，向上计数，起始电平为 1 情况下翻转点和周期时序图

4、 中心对齐模式，向下计数情况：

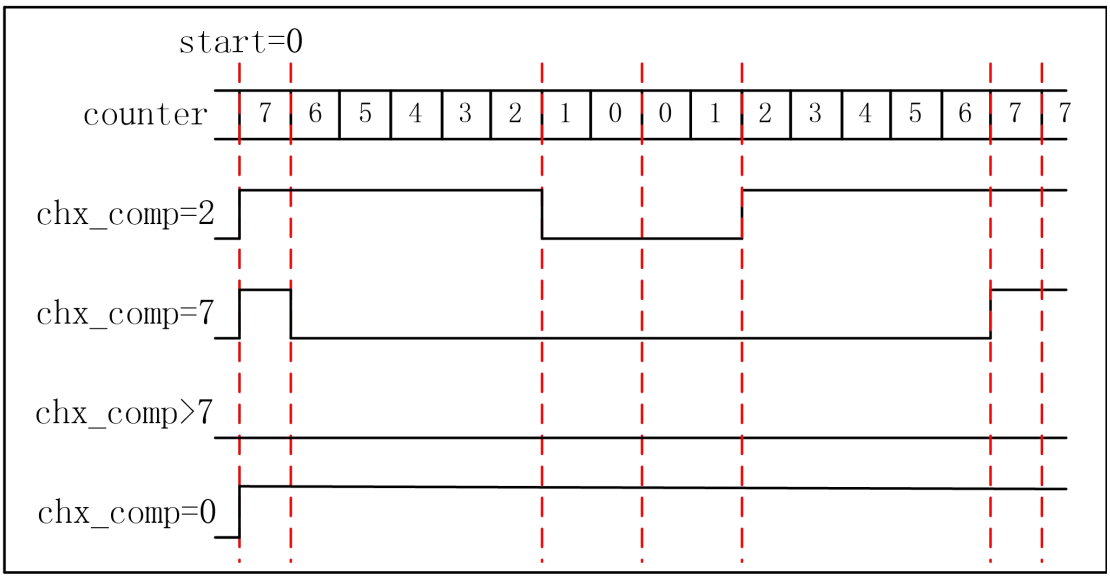


图 5-105 PWMPLUS 中心对齐模式，向下计数，起始电平为 0 情况下翻转点和周期时序图

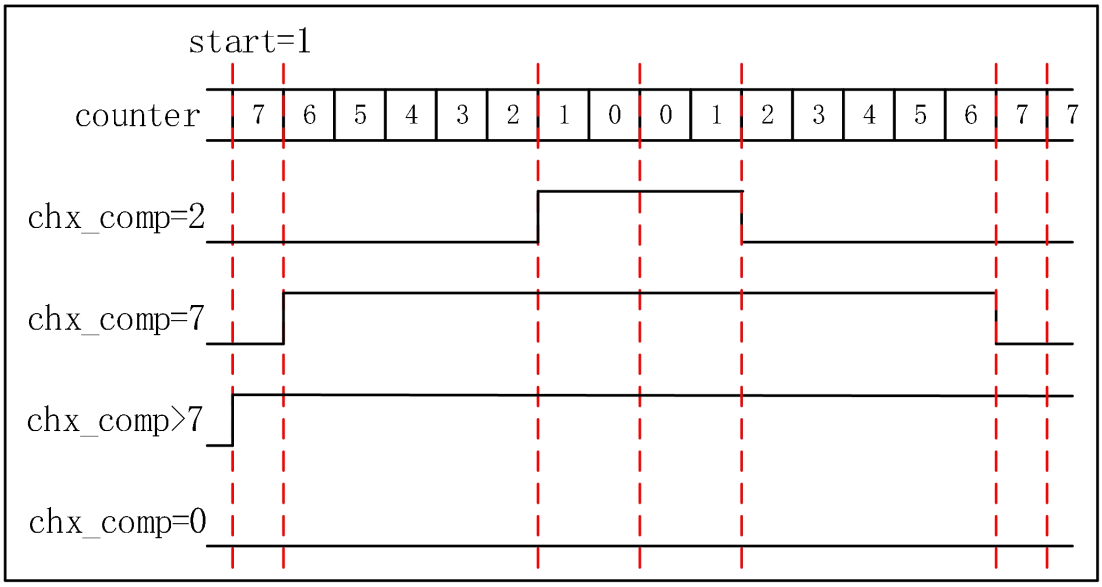


图 5-106 PWMPLUS 中心对齐模式，向下计数，起始电平为 1 情况下翻转点和周期时序图

输出优先级关系

死区值计算、通道屏蔽、刹车信号、输出翻转、输出使能这五种情况会影响各通道 PWM 波形输出，输出波形优先级关系如下所示：

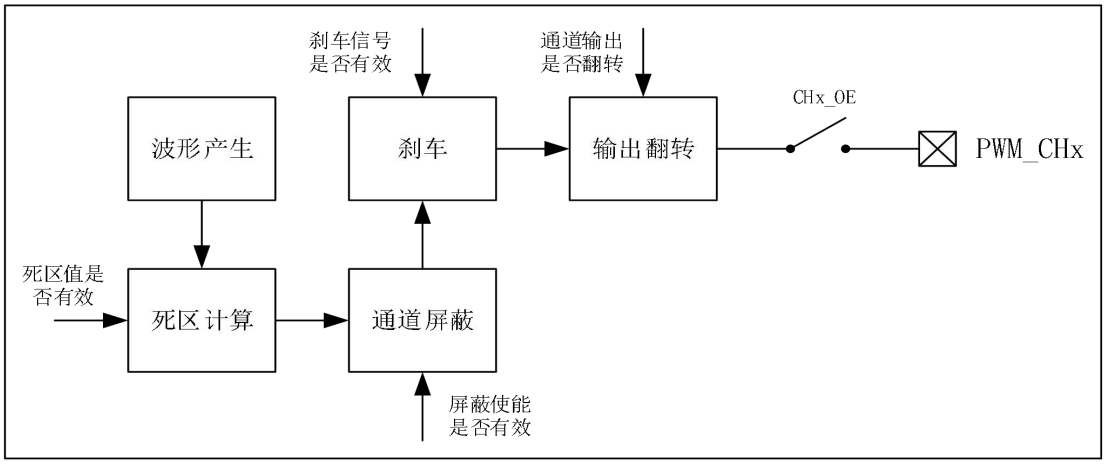


图 5-107 PWMPLUS 输出优先级关系图

触发信号及中断

本模块可实现三个通道翻转点、周期结束和特定触发点三种情况的触发信号，触发信号为 1 个 pclk 时钟的高电平；还可以实现不同计数模式下的翻转点、周期溢出、特点触发点以及刹车中断信号的产生，其中特定触发点只能产生相应的触发状态和中断状态，并不会改变 PWM 输出的波形。下图以边沿对齐模式，起始值为 0，向上计数为例，展示了各种情况下触发信号和中断的产生。

1、 刹车中断

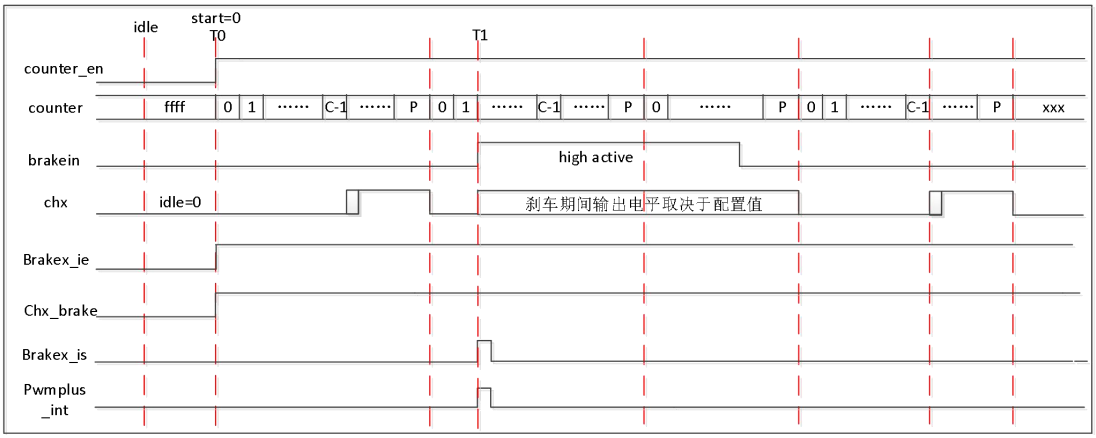


图 5-108 PWMPLUS 刹车中断时序图

如上图所示，刹车中断信号受刹车中断使能寄存器 brake_ie 和刹车信号控制寄存器 ch_brake 的控制，只有任意通道受到任意 bit 的输入刹车信号的控制且当相应 bit 的输入刹

车信号有效电平到来时，才能产生相应的中断状态，且当中断使能为 1 时，才会产生相应的中断信号。如上图所示，在 T0 时刻，打开刹车中断使能，配置刹车信号控制寄存器 ch_brake，使得任意通道受到刹车信号的控制，当 T1 时刻输入刹车信号有效电平到来时，会产生相应的中断状态和中断信号。

2、 翻转点，特定触发点，周期溢出中断以及触发信号

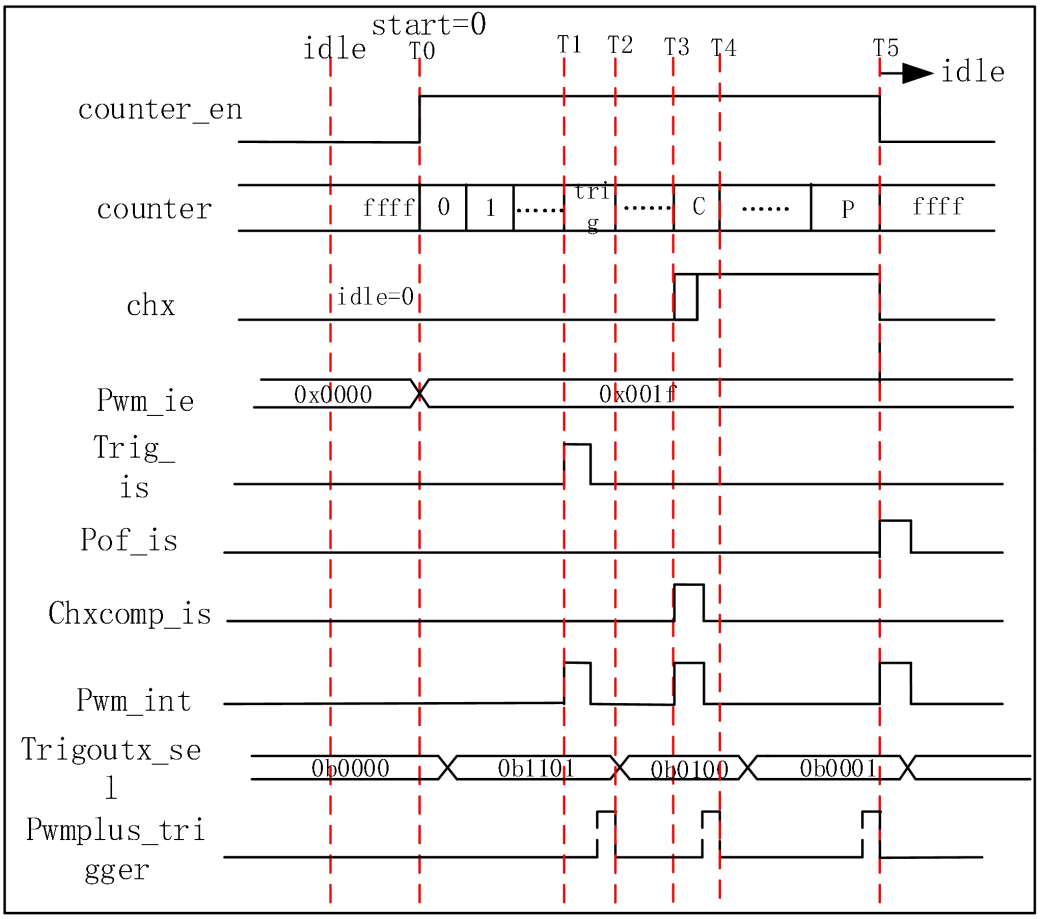


图 5-109 PWMPLUS 翻转点，特定触发点，周期溢出中断以及触发信号时序图

如上图所示，在 T0 时刻打开相应的中断使能，即在向上计数下的内部触发点中断，翻转点中断和周期溢出中断使能。在 T1 时刻到达内部触发点后，会产生内部触发点中断状态，同时产生中断信号，其中内部触发点只产生中断状态，并不会影响 PWM 波形的产生。同理在 T3 和 T5 时刻分别到达翻转点和周期溢出后，会产生相应的翻转点中断状态和周期溢出中断状态，并产生相应的中断信号。通过配置寄存器 PWMPLUS_TRIG_CFG 可以选择 trigger 信号的输出功能，到达相应的状态后会产生相应的 trigger。上图中以向上计数器为例，展示了

在不同 trigger 信号功能下的时序图，在配置相应的输出功能后会在 T2，T4，T5 时刻前产生一个系统时钟周期的输出信号。但是，不同功能的触发信号不能同时选择，因此每 bit 位的触发信号一次只能选择一种输出功能，可以通过不同 bit 位来配置不同的输出功能，本模块 pwm_trigger 共有 4 位。

自动装载

PWMPLUS 模块支持自动装载功能，此功能可以通过配置自动装载寄存器 AUTO_RELOAD 来实现，表示周期溢出多少次后自动装载一次周期值、比较值、死区值和 TRIGGER 值。自动装载动作在（AUTO_RELOAD+1）次周期溢出后进行自动装载。

软件装载

PWMPLUS 模块支持软件装载功能，本模块中 PWM_PERIOD、PWM_CH0_COMP、PWM_CH1_COMP、PWM_CH2_COMP、PWM_CH0_DT、PWM_CH1_DT、PWM_CH2_DT、TRIG_COMP 这 8 个寄存器具有各自的影子寄存器。软件向软件 LOAD 控制位 SWLOAD 写 1 后，硬件在本周期结束后将这 8 个寄存器中保存的最新值 load 到各自的影子寄存器中，并在接下来的周期生效。

操作流程

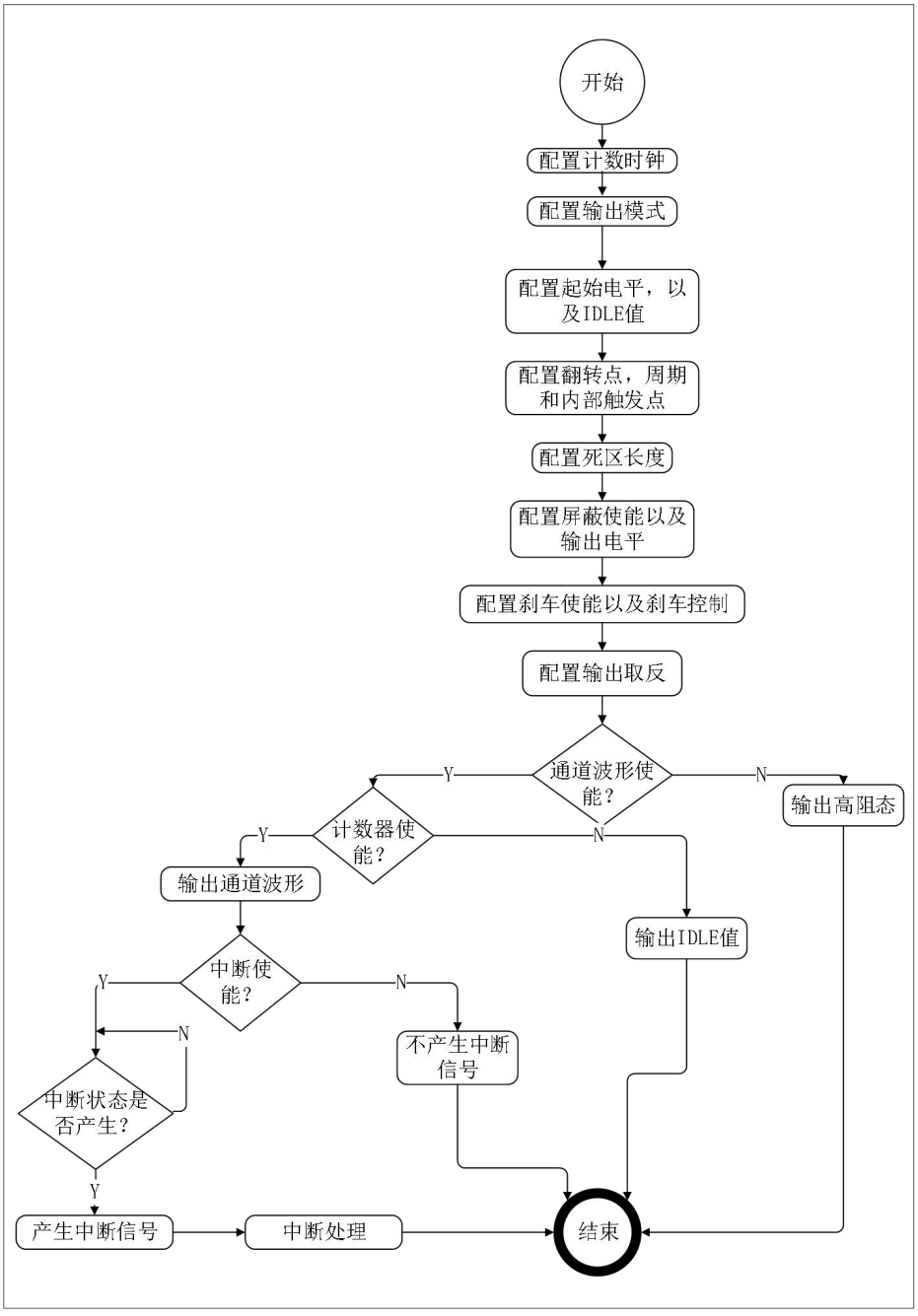


图 5-110 PWMPLUS 模块操作流程图

- 配置 PWMPLUS 时钟使能
- PORT 端口配置为 PWMPLUS 功能
- 配置计数时钟

- 配置输出模式，计数循环方式，计数行为方式
- 配置起始电平和 IDLE 值
- 配置周期值、翻转点值和内部触发点值
- 配置自动装载寄存器
- 配置死区长度
- 配置屏蔽使能以及输出电平
- 配置刹车使能，刹车滤波，刹车信号有效沿以及刹车输出电平
- 配置输出翻转
- 如果配置了中断，使能 PWMPLUS 中断
- 配置通道波形输出使能，如果使能未打开则 PAD 端口输出高阻态
- 配置计数器使能，如果使能未打开则输出 IDLE 值

寄存器描述

PWMPLUS_CFG 寄存器（0x00）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留
15:8	AUTO_RELOAD	R/W	0	自动装载寄存器，表示周期溢出多少次后自动装载一次周期值、比较值、死区值和 TRIGGER 值。 自动装载动作在（AUTO_RELOAD+1）次周期溢出后进行自动装载。
7:4	RESERVED	R	0	保留
3	OUT_MODE	R/W	0	PWM 输出模式 0：边沿对齐模式输出 1：中心对齐模式输出
2	CNT_REP	R/W	0	PWM 计数器循环方式 0：单次，一次计数周期后停止 1：循环，启动后循环计数，直到软件停止

1	CNT_TYPE	R/W	0	PWM 计数器行为方式 0：向上计数 1：向下计数 若为中心对齐模式，则表示计数器前半周期的计数行为。
0	COUNTER_EN	R/W	0	计数器使能寄存器 当该位配置为 1，则表示计数器开始计数，CH0/CH1/CH2 按照事先配置好的周期值、比较值和死区值产生相应通道的 PWM 输出波形。 在配置为单次方式时，该位在一个时钟周期后硬件会自动清零。 在配置为循环方式时，需要将该位软件清零，才能停止计数器计数，当计满一个完整计数周期后输出波形电平恢复到 IDLE 状态。

注：CH0、CH1 和 CH2 共用一个周期配置寄存器。

PWMPLUS_GEN 寄存器（0x04）

位域	名称	类型	复位值	描述
31:30	RESERVED	R	0	保留
29	CH2N_OE	R/W	0	CH2N 通道波形输出使能 0：输出关闭，管脚上为高阻状态 1：输出 CH2N 方波
28	CH2_OE	R/W	0	CH2 通道波形输出使能 0：输出关闭，管脚上为高阻状态 1：输出 CH2 方波
27	CH1N_OE	R/W	0	CH1N 通道波形输出使能 0：输出关闭，管脚上为高阻状态 1：输出 CH1N 方波
26	CH1_OE	R/W	0	CH1 通道波形输出使能 0：输出关闭，管脚上为高阻状态 1：输出 CH1 方波

25	CH0N_OE	R/W	0	CH0N 通道波形输出使能 0：输出关闭，管脚上为高阻状态 1：输出 CH0N 方波
24	CH0_OE	R/W	0	CH0 通道波形输出使能 0：输出关闭，管脚上为高阻状态 1：输出 CH0 方波
23:22	RESERVED	R	0	保留
21	CH2N_OUTINV	R/W	0	CH2N 通道输出状态选择 0：电平无变化 1：输出电平反向
20	CH2_OUTINV	R/W	0	CH2 通道输出状态选择 0：电平无变化 1：输出电平反向
19	CH1N_OUTINV	R/W	0	CH1N 通道输出状态选择 0：电平无变化 1：输出电平反向
18	CH1_OUTINV	R/W	0	CH1 通道输出状态选择 0：电平无变化 1：输出电平反向
17	CH0N_OUTINV	R/W	0	CH0N 通道输出状态选择 0：电平无变化 1：输出电平反向
16	CH0_OUTINV	R/W	0	CH0 通道输出状态选择 0：电平无变化 1：输出电平反向
15:11	RESERVED	R	0	保留
10	CH2_START	R/W	0	原始 CH2 通道计数开始时输出状态值 0：原始 CH2 通道输出 0 电平 1：原始 CH2 通道输出 1 电平 计数开始后，CH2N 为 CH2 的反

9	CH1_START	R/W	0	原始 CH1 通道计数开始时输出状态值 0：原始 CH1 通道输出 0 电平 1：原始 CH1 通道输出 1 电平 计数开始后，CH1N 为 CH1 的反
8	CH0_START	R/W	0	原始 CH0 通道开始计数时输出状态值 0：原始 CH0 通道输出 0 电平 1：原始 CH0 通道输出 1 电平 计数开始后，CH0N 为 CH0 的反
7:6	RESERVED	R	0	保留
5	CH2N_IDLE	R/W	1	原始 CH2N 通道空闲时输出状态值 0：原始 CH2N 通道输出 0 电平 1：原始 CH2N 通道输出 1 电平
4	CH2_IDLE	R/W	0	原始 CH1 通道空闲时输出状态值 0：原始 CH1 通道输出 0 电平 1：原始 CH1 通道输出 1 电平
3	CH1N_IDLE	R/W	1	原始 CH1N 通道空闲时输出状态值 0：原始 CH1N 通道输出 0 电平 1：原始 CH1N 通道输出 1 电平
2	CH1_IDLE	R/W	0	原始 CH1 通道空闲时输出状态值 0：原始 CH1 通道输出 0 电平 1：原始 CH1 通道输出 1 电平
1	CH0N_IDLE	R/W	1	原始 CH0N 通道空闲时输出状态值 0：原始 CH0N 通道输出 0 电平 1：原始 CH0N 通道输出 1 电平
0	CH0_IDLE	R/W	0	原始 CH0 通道空闲时输出状态值 0：原始 CH0 通道输出 0 电平 1：原始 CH0 通道输出 1 电平

PWMPLUS_CLKSRC 寄存器（0x08）

位域	名称	类型	复位值	描述

31:16	PREDIV	R/W	0	内部预分频时钟频率选择。以本模块的 pclk 作为预分频时钟的时钟源。 0x0000：表示 pclk 的 1 分频 0x0001：表示 pclk 的 2 分频 0x0002：表示 pclk 的 3 分频 …… 0xFFFF：表示 pclk 的 65536 分频
15:6	RESERVED	R	0	保留位
5	EXTPLUS1_EDGE	R/W	0	ExtPLUS1 作为计数时钟时边沿选择控制 0：下降沿触发计数 1：上升沿触发计数
4	EXTPLUS0_EDGE	R/W	0	ExtPLUS0 作为计数时钟时边沿选择控制 0：下降沿触发计数 1：上升沿触发计数
3	RESERVED	R	0	保留位
2:0	CNT_SRC	R/W	0	PWM 计数器计数时钟选择 000：选择内部预分频时钟作为计数时钟 001：选择 extPLUS[0]作为计数时钟 010：选择 extPLUS[1]作为计数时钟 011：选择 tmPLUS[0]作为计数时钟 100：选择 tmPLUS[1]作为计数时钟 101：选择 tmPLUS[2]作为计数时钟 110：选择 tmPLUS[3]作为计数时钟 111：保留

PWMPLUS_BRAKE_CFG 寄存器（0x0c）

位域	名称	类型	复位值	描述
31:26	RESERVED	R	0	保留位

25:24	BRAKE_FILTER	R/W	0	刹车信号数字滤波控制 00：不滤波 01：进行 2 个内部预分频时钟滤波 10：进行 4 个内部预分频时钟滤波 11：进行 8 个内部预分频时钟滤波
23:22	RESERVED	R	0	保留位
21	BRAKE_CH2NPOL	R/W	0	刹车时 CH2N 输出电平选择 0：刹车时输出 0 1：刹车时输出 1
20	BRAKE_CH2POL	R/W	0	刹车时 CH2 输出电平选择 0：刹车时输出 0 1：刹车时输出 1
19	BRAKE_CH1NPOL	R/W	0	刹车时 CH1N 输出电平选择 0：刹车时输出 0 1：刹车时输出 1
18	BRAKE_CH1POL	R/W	0	刹车时 CH1 输出电平选择 0：刹车时输出 0 1：刹车时输出 1
17	BRAKE_CH0NPOL	R/W	0	刹车时 CH0N 输出电平选择 0：刹车时输出 0 1：刹车时输出 1
16	BRAKE_CH0POL	R/W	0	刹车时 CH0 输出电平选择 0：刹车时输出 0 1：刹车时输出 1
15	RESERVED	R	0	保留位
14:12	BRAKE_LEV	R/W	0	刹车有效电平选择 BRAKE_LEV 的 Bit2 对应 brakein[2]，Bit1 对应 brakein[1]，Bit0 对应 brakein[0]。 0：表示刹车输入低电平有效 1：表示刹车输入高电平有效
11:9	RESERVED	R	0	保留位

8:6	CH2_BRAKE	R/W	0	CH2/CH2N 刹车控制选择 CH2_BRAKE 的 Bit2 对应 brakein[2]，Bit1 对应 brakein[1]，Bit0 对应 brakein[0]。 0：表示不受刹车信号控制 1：表示受刹车信号控制
5:3	CH1_BRAKE	R/W	0	CH1/CH1N 刹车控制选择 CH1_BRAKE 的 Bit2 对应 brakein[2]，Bit1 对应 brakein[1]，Bit0 对应 brakein[0]。 0：表示不受刹车信号控制 1：表示受刹车信号控制
2:0	CH0_BRAKE	R/W	0	CH0/CH0N 刹车控制选择 CH0_BRAKE 的 Bit2 对应 brakein[2]，Bit1 对应 brakein[1]，Bit0 对应 brakein[0]。 0：表示不受刹车信号控制 1：表示受刹车信号控制

注：刹车开始时，对输出波形立刻生效；刹车过程中，计数器正常计数；刹车撤销后，各通道在完整周期结束后输出波形。

PWMPLUS_MASK_LEV 寄存器（0x10）

位域	名称	类型	复位值	描述
31:6	RESERVED	R	0	保留位
5	CH2N_MASK_LEV	R/W	0	CH2N 通道屏蔽电平选择 0：屏蔽期强制输出 0 1：屏蔽期强制输出 1
4	CH2_MASK_LEV	R/W	0	CH2 通道屏蔽电平选择 0：屏蔽期强制输出 0 1：屏蔽期强制输出 1
3	CH1N_MASK_LEV	R/W	0	CH1N 通道屏蔽电平选择 0：屏蔽期强制输出 0 1：屏蔽期强制输出 1

2	CH1_MASK_LEV	R/W	0	CH1 通道屏蔽电平选择 0：屏蔽期强制输出 0 1：屏蔽期强制输出 1
1	CH0N_MASK_LEV	R/W	0	CH0N 通道屏蔽电平选择 0：屏蔽期强制输出 0 1：屏蔽期强制输出 1
0	CH0_MASK_LEV	R/W	0	CH0 通道屏蔽电平选择 0：屏蔽期强制输出 0 1：屏蔽期强制输出 1

PWMPLUS_PERIOD 寄存器（0x1C）

位域	名称	类型	复位值	描述
31:16	RESERVED	RO	0	保留位
15:0	PERIOD	R/W	0xffff	PWMPLUS 周期配置寄存器。 实际计数周期为此寄存器配置周期值加 1。 注 0：周期不能配置为 0。 例如：配置为十进制 199，则认为 PWM 波形周期为 200。 注 1：在中心对齐模式下，实际周期为配置值加 1 的 2 倍。

PWMPLUS_CH0_COMP 寄存器（0x20）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	CH0_COMP	R/W	0	CH0/CH0N 翻转点配置寄存器。 注：计数值小于翻转点值，输出 start[0]；大于等于翻转点值，输出 start[0]的非。

PWMPLUS_CH1_COMP 寄存器（0x24）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	CH1_COMP	R/W	0	CH1/CH1N 翻转点配置寄存器。 注：计数值小于翻转点值，输出 start[1]；大于等于翻转点值，输出 start[1]的非。

PWMPLUS_CH2_COMP 寄存器（0x28）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	CH2_COMP	R/W	0	CH2/CH2N 翻转点配置寄存器。 注：计数值小于翻转点值，输出 start[2]；大于等于翻转点值，输出 start[2]的非。

PWMPLUS_CH0_DT 寄存器（0x30）

位域	名称	类型	复位值	描述
31:10	RESERVED	R	0	保留位
9:0	CH0_DT	R/W	0	CH0/CH0N 死区长度配置寄存器 注 0：配置为 0 表示无死区；配置为 1 表示死区长度为 1，配置为 2 表示死区长度为 2，以此类推。 注 1：配置该值时需要与周期值、CH0_START 值、CH0_COMP 值相匹配，否则输出波形可能达不到预期效果。

PWMPLUS_CH1_DT 寄存器（0x34）

位域	名称	类型	复位值	描述
31:10	RESERVED	R	0	保留位
9:0	CH1_DT	R/W	0	CH1/CH1N 死区长度配置寄存器 注 0：配置为 0 表示无死区；配置为 1 表示死区长度为 1，配置为 2 表示死区长度为 2，以此类推。 注 1：配置该值时需要与周期值、CH1_START 值、CH1_COMP 值相匹配，否则输出波形可能达不到预期效果。

PWMPLUS_CH2_DT 寄存器（0x38）

位域	名称	类型	复位值	描述
31:10	RESERVED	R	0	保留位
9:0	CH2_DT	R/W	0	CH2/CH2N 死区长度配置寄存器 注 0：配置为 0 表示无死区；配置为 1 表示死区长度为 1，配置为 2 表示死区长度为 2，以此类推。 注 1：配置该值时需要与周期值、CH2_START 值、CH2_COMP 值相匹配，否则输出波形可能达不到预期效果。

PWMPLUS_TRIG_COMP 寄存器（0x40）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位
15:0	TRIG_COMP	R/W	0	内部触发点配置寄存器 注：内部触发点值必须小于周期值

PWMPLUS_TRIG_CFG 寄存器（0x44）

位域	名称	类型	复位值	描述
31:4	RESERVED	R	0	保留位
3:0	TIRGOUT_L_SEL	R/W	0	输出的 trigger0 信号功能选择 0000：无信号输出； 0001：向上计数周期溢出点； 0010：向下计数周期溢出点； 0011：向上或向下计数周期溢出点； 0100：CH0 向上计数翻转点； 0101：CH0 向下计数翻转点； 0110：CH0 向上或向下翻转点； 0111：CH1 向上计数翻转点； 1000：CH1 向下计数翻转点； 1001：CH1 向上或向下翻转点； 1010：CH2 向上计数翻转点； 1011：CH2 向下计数翻转点； 1100：CH2 向上或向下翻转点； 1101：向上计数内部触发点； 1110：向下计数内部触发点； 1111：向上或向下内部触发点；

PWMPLUS_IE 寄存器（0x60）

位域	名称	类型	复位值	描述
31:20	RESERVED	R	0	保留位
19	AUTORELOAD_IE	R/W	0	自动装载中断使能
18	BRAK2_IE	R/W	0	刹车 2 中断使能
17	BRAK1_IE	R/W	0	刹车 1 中断使能

16	BRAK0_IE	R/W	0	刹车 0 中断使能
15:13	RESERVED	R	0	保留位
12	DOWN_TRIG_IE	R/W	0	向下计数达到 TRIGGER 触发点中断使能
11	DOWN_POF_IE	R/W	0	向下计数周期溢出中断使能
10	DOWN_CH2COMP_IE	R/W	0	向下计数 CH2 到达翻转点中断使能
9	DOWN_CH1COMP_IE	R/W	0	向下计数 CH1 到达翻转点中断使能
8	DOWN_CH0COMP_IE	R/W	0	向下计数 CH0 到达翻转点中断使能
7:5	RESERVED	R	0	保留位
4	UP_TRIG_IE	R/W	0	向上计数达到 TRIGGER 触发点中断使能
3	UP_POF_IE	R/W	0	向上计数周期溢出中断使能
2	UP_CH2COMP_IE	R/W	0	向上计数 CH2 到达翻转点中断使能
1	UP_CH1COMP_IE	R/W	0	向上计数 CH1 到达翻转点中断使能
0	UP_CH0COMP_IE	R/W	0	向上计数 CH0 到达翻转点中断使能

PWMPLUS_IF 寄存器（0x64）

位域	名称	类型	复位值	描述
31:20	RESERVED	R	0	保留位
19	AUTORELOAD_IF	R/W	0	自动装载中断状态 写 1 清零
18	BRAK2_IF	R/W	0	刹车 2 中断状态 写 1 清零 注：只有任意通道受刹车 2 信号控制，该状态才会产生。

17	BRAK1_IF	R/W	0	刹车 1 中断状态 写 1 清零 注：只有任意通道受刹车 1 信号控制，该状态才会产生。
16	BRAK0_IF	R/W	0	刹车 0 中断状态 写 1 清零 注：只有任意通道受刹车 0 信号控制，该状态才会产生。
15:13	RESERVED	R	0	保留位
12	DOWN_TRIG_IF	R/W	0	向下计数达到 TRIGGER 触发点中断状态 写 1 清零 注：边沿对齐模式，表示配置为向下计数时到达内部触发点；中心对齐模式，表示配置为向下计数时，在前半周期到达内部触发点或配置为向上计数时，在后半周期达到内部触发点
11	DOWN_POF_IF	R/W	0	向下计数周期溢出中断状态 写 1 清零 注：边沿对齐模式，表示配置为向下计数时到达周期溢出点；中心对齐模式，表示配置为向下计数时，在前半周期到达周期溢出点或配置为向上计数时，在后半周期达到周期溢出点
10	DOWN_CH2COMP_IF	R/W	0	向下计数 CH2 到达翻转点中断状态 写 1 清零
9	DOWN_CH1COMP_IF	R/W	0	向下计数 CH1 到达翻转点中断状态 写 1 清零
8	DOWN_CH0COMP_IF	R/W	0	向下计数 CH0 到达翻转点中断状态 写 1 清零 注：边沿对齐模式，表示配置为向下计数时到达翻转点；中心对齐模式，表示配置为向下计数时，在前半周期到达翻转点或配置为向上计数时，在后半周期达到翻转点。其他通道行为与之一样。
7:5	RESERVED	R	0	保留位

4	UP_TRIG_IF	R/W	0	向上计数达到 TRIGGER 触发点中断状态 写 1 清零 注：边沿对齐模式，表示配置为向上计数时到达内部触发点；中心对齐模式，表示配置为向上计数时，在前半周期到达内部触发点或配置为向下计数时，在后半周期达到内部触发点
3	UP_POF_IF	R/W	0	向上计数周期溢出中断状态 写 1 清零 注：边沿对齐模式，表示配置为向上计数时到达周期溢出点；中心对齐模式，表示配置为向上计数时，在前半周期到达周期溢出点或配置为向下计数时，在后半周期达到周期溢出点
2	UP_CH2COMP_IF	R/W	0	向上计数 CH2 到达翻转点中断状态 写 1 清零
1	UP_CH1COMP_IF	R/W	0	向上计数 CH1 到达翻转点中断状态 写 1 清零
0	UP_CH0COMP_IF	R/W	0	向上计数 CH0 到达翻转点中断状态 写 1 清零 注：边沿对齐模式，表示配置为向上计数时到达翻转点；中心对齐模式，表示配置为向上计数时，在前半周期到达翻转点或配置为向下计数时，在后半周期达到翻转点。其他通道行为与之一样。

PWMPLUS_SWLOAD 寄存器（0x84）

位域	名称	类型	复位值	描述
31:1	RESERVED	R	0	保留位
0	SWLOAD	R/W	0	PWM 配置寄存器软件 LOAD 控制位 本模块中 PWM_PERIOD、PWM_CH0_COMP、PWM_CH1_COMP、PWM_CH2_COMP、PWM_CH0_DT、PWM_CH1_DT、PWM_CH2_DT、TRIG_COMP 这 8 个寄存器具有各自的影子寄存器。软件向该位写 1 后，硬件在本周期结束后将这 8 个寄存器中保存的最新值 load 到各自的影子寄存器中，并在下一周期生效。

PWMPLUS_MASK_EN 寄存器（0x88）

位域	名称	类型	复位值	描述
31:6	RESERVED	R	0	保留位
5	CH2N_MASK_EN	R/W	0	CH2N 通道屏蔽使能 将该位写 1，则启动强制输出屏蔽功能，强制输出电平通过 MASK_CFG 选择。
4	CH2_MASK_EN	R/W	0	CH2 通道屏蔽使能 将该位写 1，则启动强制输出屏蔽功能，强制输出电平通过 MASK_CFG 选择。
3	CH1N_MASK_EN	R/W	0	CH1N 通道屏蔽使能 将该位写 1，则启动强制输出屏蔽功能，强制输出电平通过 MASK_CFG 选择。
2	CH1_MASK_EN	R/W	0	CH1 通道屏蔽使能 将该位写 1，则启动强制输出屏蔽功能，强制输出电平通过 MASK_CFG 选择。
1	CH0N_MASK_EN	R/W	0	CH0N 通道屏蔽使能 将该位写 1，则启动强制输出屏蔽功能，强制输出电平通过 MASK_CFG 选择。
0	CH0_MASK_EN	R/W	0	CH0 通道屏蔽使能 将该位写 1，则启动强制输出屏蔽功能，强制输出电平通过 MASK_CFG 选择。

注：各通道屏蔽功能使能时，强制输出立刻生效。屏蔽功能撤销时，立刻按照原波形正常输出。

PWMPLUS_CNT_ST 寄存器（0xE0）

位域	名称	类型	复位值	描述
31:18	RESERVED	R	0	保留位

17	CNT_ST	R	0	PWM 计数器工作状态 0：表示计数器未工作 1：表示计数器正在计数
16	CNT_DIR	R	0	PWM 计数器当前计数方向 0：表示计数器当前向上计数 1：表示计数器当前向下计数
15:0	PWMPLUS_CNT	R	0	PWM 计数器当前计数值寄存器

PWMPLUS_BRAKE_ST 寄存器（0xE4）

位域	名称	类型	复位值	描述
31:2	RESERVED	R	0	保留位
1:0	BRAKE_ST	R	0	刹车输入信号当前状态 BRAKE_ST[1]对应 brake1，BRAKE_ST[0]对应 brake0。

5.15 实时时钟（RTC）

5.15.1 概述

实时时钟是一个独立的定时器。RTC 模块具有实时时钟功能，自动解决闰年问题，输入时钟源可选择 RCLF 或 XTAL，可输出 1/2 秒及其中断，在使用 RTC 之前需要打开相应的时钟。其系统框图如下所示：

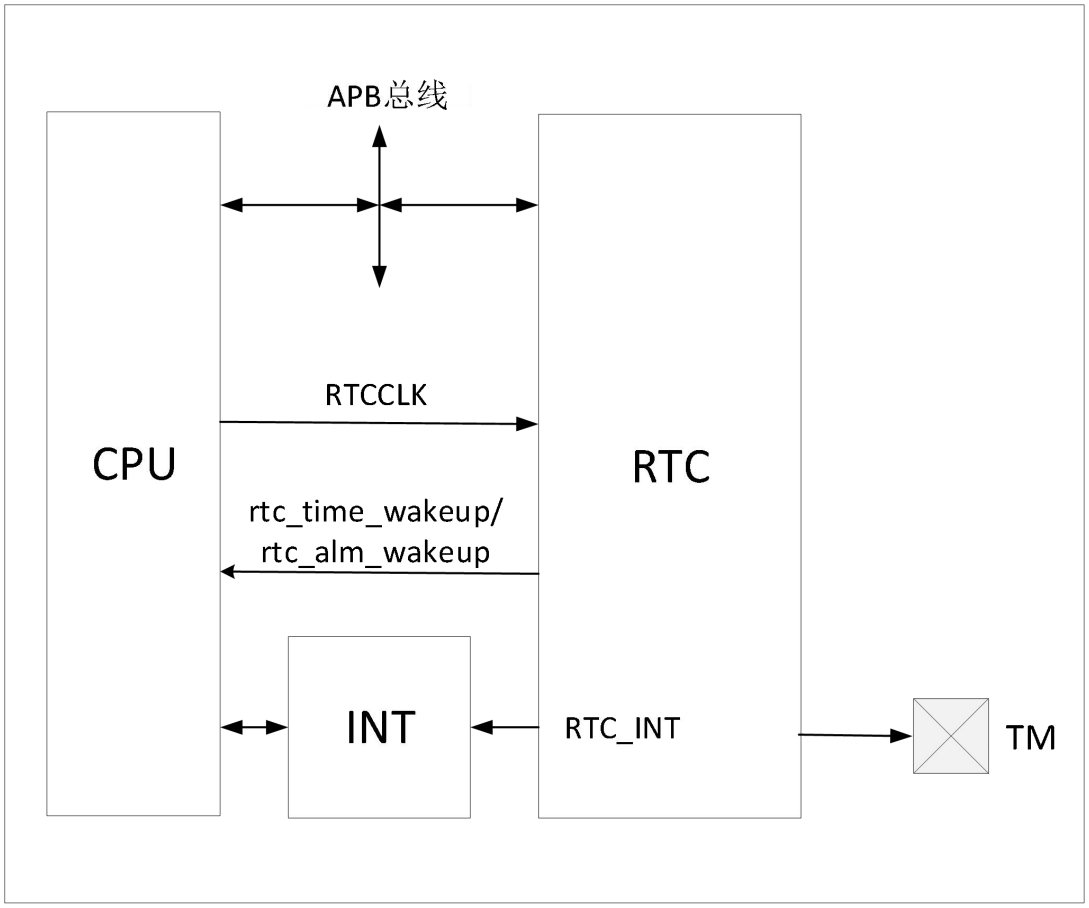


图 5-111 RTC 模块系统框图

5.15.2 特性

- 具有实时时钟功能
- 闹钟功能

- 具有日历功能，计时范围 0-99 年
- 自动解决闰年问题
- 可输出与 RTC 同步的秒、分、时、日、闹钟中断
- RTC 计数时钟可选择两种时钟源
- 可输出 1/2 秒及其中断
- 可输出 RTC 时间唤醒和闹钟唤醒信号用于低功耗唤醒模块
- 时钟设定和闹钟设定的数据有效性检测
- 具有 BCD 编码的闹钟寄存器
- 具有 BCD 编码的时间寄存器

5.15.3 模块结构图

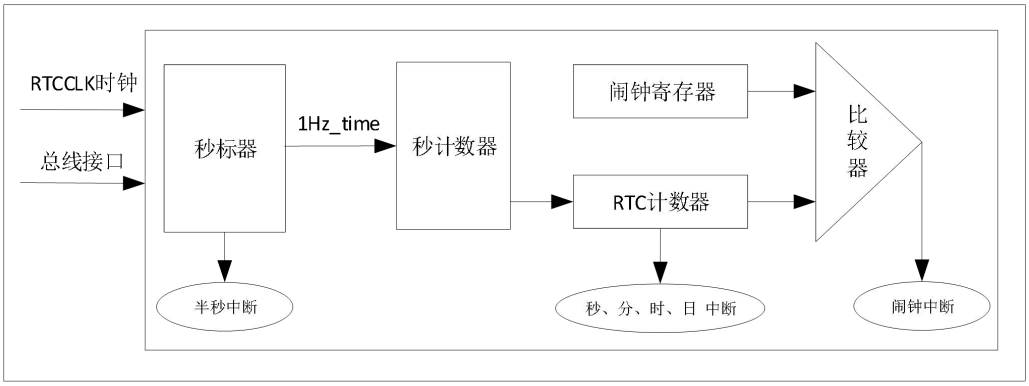


图 5-112　RTC 模块结构框图

5.15.4 功能描述

计数时钟源选择

本芯片 RTC 计数器时钟（RTCCLK）可选择两个时钟源：一个为片内低频 RC 振荡器（RCLF），另一个是片外低频晶体振荡器（XTAL）。

通过设置 SRC_CFG 寄存器中的 RTC_CLK_SEL 位，RTCCLK 时钟源可以由 XTAL 或 RCLF 时钟提供。并且在 RTC 使能之前，必须完成 RTC 的相关各项配置（时钟源选择等）。

RTCCLK 选择示意图如下所示：

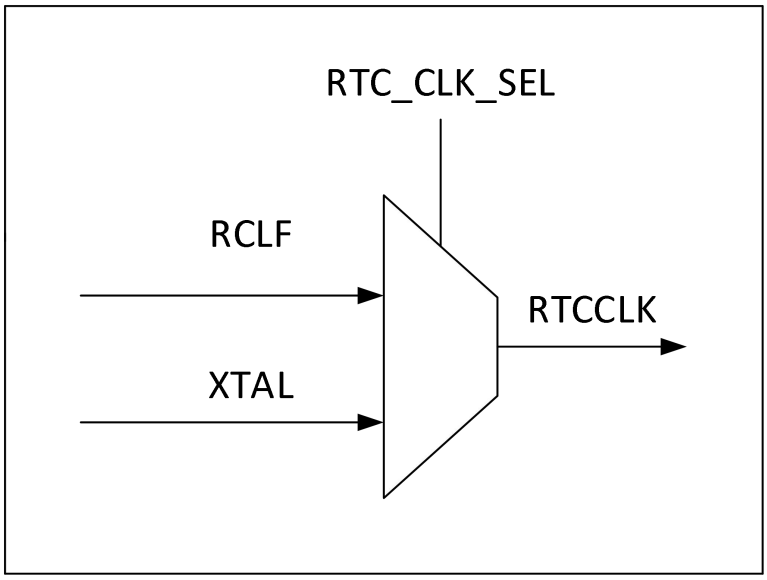


图 5-113 RTCCLK 选择示意图

秒标功能

秒标器产生用于更新日历的秒标时间（1Hz_time），还可通过配置 RTC_PRE 寄存器产生 1/2 秒时间。

RTC_PRE 寄存器的 PRE_ROUND 位是预分频的整数部分，PRE_PERIOD 指每个多少秒进行一次小数部分计算，PRE_DECIMAL 是预分频的小数部分。根据 PRE_PERIOD 和 PRE_DECIMAL 对比哪种精度更优来确定具体配置值，1 秒的误差范围为±1ppm。

例如 1：

RTC 时钟频率为 32767.62Hz，则 PRE_ROUND 配置为 32766；小数部分分别计算 0.62*8=4.96 和 0.62*16=9.92，则选择 8 秒需配置为 5，选择 16 秒需配置 10，可以看出两种方式精度一致，优先选择 8 秒，PRE_DECIMAL 配置为 5，PRE_PERIOD 配置为 0。

则 1 秒的误差为（（32767+5/8）/32767.62）-1 = 0.15ppm

例如 2：

RTC 时钟频率为 32767.71Hz，则 PRE_ROUND 配置为 32766；小数部分分别计算 0.71*8=5.68 和 0.71*16=11.36，则选择 8 秒需配置为 6，选择 16 秒需配置 11，可以看出选择 16 秒精度更高，优先选择 16 秒，PRE_DECIMAL 配置为 11，PRE_PERIOD 配置为 1。

则 1 秒的误差为（（32767+11/16）/32767.71）-1 = -0.68ppm

例如 3：

RTC 时钟频率为 32770.094Hz，则 PRE_ROUND 配置为 32769；小数部分分别计算 0.094*8=0.752 和 0.094*16=1.504，则选择 8 秒需配置为 1，选择 16 秒需配置 1，可以看出选择 8 秒精度更高，优先选择 8 秒，PRE_DECIMAL 配置为 1，PRE_PERIOD 配置为 0。

则 1 秒的误差为（（32770+1/8）/32770.094）-1 = 0.946ppm

日历功能

实时时钟基于秒标器产生的 1Hz_time，对年、月、日、时、分、秒进行精确计时，并输出计时时间。RTC 接受时间设定，可以设置开始计时的时间。

日历功能有一套配置时间寄存器和当前时间寄存器，采用 BCD 码编码格式。配置时间寄存器具备自动检查时间格式功能（除星期外），能够根据用户写入的数值判断设置的时间是否合法。若时间格式不合法（如 1 月 34 日，2 月 30 日，24 时 01 分等），会在状态寄存器 RTC_IF 中的 TIME_ERR 位显示设定时间数据格式错误。此时即使打开 LOAD_EN，配置时间值也不会生效。

闹钟功能

RTC 模块提供闹钟功能，在计数器计到闹钟设定值时，可以产生中断。

闹钟设定通过 RTC_AR 寄存器进行，仅支持星期、时、分、秒。闹钟寄存器具有自动检查时间格式功能（除星期外），能够根据用户写入的数值判断设置的时间是否合法。若写入闹钟寄存器的格式不合法（如 20 时 20 分 63 秒、24 时 01 分等），会在状态寄存器 RTC_IF 中的 ALM_ ERR 位查询到闹钟格式错误，此时即使打开闹钟功能使能位，闹钟也不生效。当星期设置为全 0 时，闹钟仅响应一次，否则闹钟将会循环响应直到关闭。

中断功能

本模块提供了六种中断功能，包括秒、分、时、日、闹钟和半秒中断。可通过 RTC_IE 寄存器配置相应中断使能位，通过 RTC_IF 寄存器查看各相应中断状态位。

中断标志及中断示意图如下：

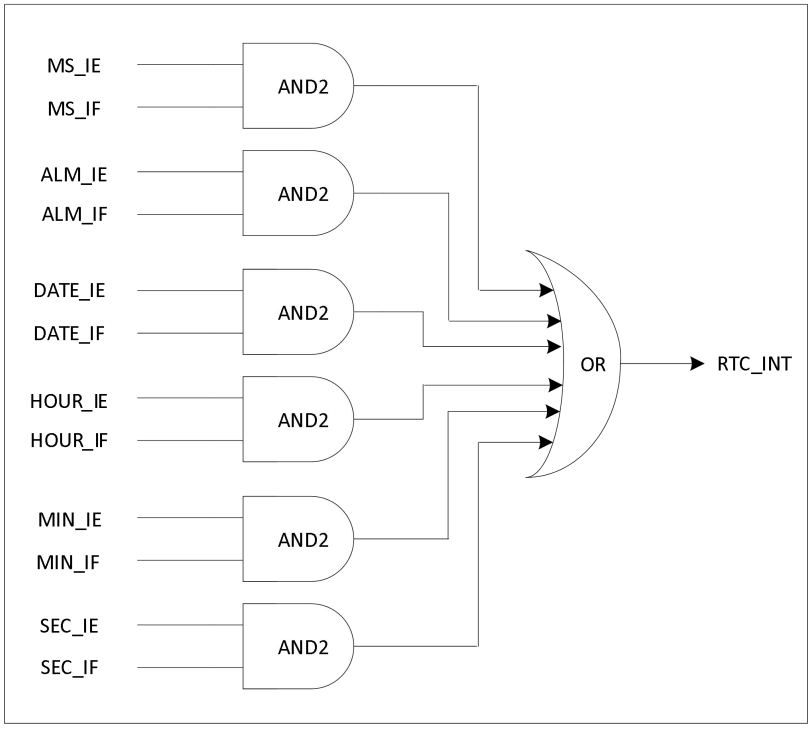


图 5-114 RTC 中断标志及中断示意图

低功耗唤醒功能

本模块提供了两种低功耗唤醒功能，可支持在 SLEEP 模式和 DEEPSLEEP 模式唤醒系统。分别为时间唤醒和闹钟唤醒。

通过配置 RTC 的时间和闹钟可以产生相应的唤醒信号给 PMU 唤醒电路，通过 PMU 唤醒电路配置 LPMD_WKEN 寄存器可以打开相应唤醒使能，通过 LPMD_WKST 寄存器查询唤醒状态。

操作流程

初始化设置

1、根据 RTCCLK，配置 RTC_PRE 寄存器，使输出 1Hz_time；

2、配置 RTC_TR 和 RTC_DR 寄存器；

3、读取状态寄存器 RTC_IF 的 TIME_ERR 位，确认格式正确；

4、配置 RTC_CFG 的 LOAD_EN 位为 1，打开设置时间功能；

5、配置 RTC_AR 寄存器；

6、读取状态寄存器 RTC_IF 的 ALM_ERR 位，确认格式正确；

7、配置 RTC_CFG 的 ALM_EN 位为 1，打开闹钟功能；

8、配置 RTC_CFG 的 RTC_EN 位为 1，打开 RTC，开始计时；

注：若无需时间设定功能跳过 2-4；若无需闹钟功能跳过 5-7。

时间设置

1、配置 RTC_TR 和 RTC_DR 寄存器；

2、读取状态寄存器 RTC_IF 的 TIME_ERR 位，确认格式正确；

3、配置 RTC_CFG 的 LOAD_EN 位为 1，打开设置时间功能；

4、RTC 计时器在下个 RTCCLK 时钟周期载入新值并开始计时。

闹钟设置

1、配置 RTC_AR 寄存器；

2、读取状态寄存器 RTC_IF 的 ALM_ERR 位，确认格式正确；

3、配置 RTC_CFG 的 ALM_EN 位为 1，打开闹钟功能；

4、等待闹钟响应。

读取时间

1、读取并判断 RTC_VALID 为 1；

2、读取寄存器 RTC_TSTR、RTC_TSDR，得到当前时间值；

3、读取 RTC_CNT 得到秒标当前计数值；

中断响应设置

1、配置 RTC_IE，有 6 种中断可分别配置；

2、等待中断响应；

3、读取 RTC_IF 寄存器，查询当前中断状态

4、向 RTC_IF 寄存器的相应位写 1，可清除相应中断。

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
RTC	BASE: 0x40069000				
RTC_CFG	0x00	32	R/W	0x00	RTC 配置寄存器
RTC_IE	0x04	32	R/W	0x00	RTC 中断使能寄存器
RTC_IF	0x08	32	R/W	0x00	RTC 状态寄存器
RTC_PRE	0x10	32	R/W	0x7fff	RTC 预分频寄存器
RTC_TR	0x14	32	R/W	0x00	RTC 时间寄存器
RTC_DR	0x18	32	R/W	0x101	RTC 日期寄存器
RTC_AR	0x1C	32	R/W	0x00	RTC 闹钟寄存器
RTC_TSTR	0x20	32	RO	0x00	RTC 当前时间寄存器
RTC_TSDR	0x24	32	RO	0x101	RTC 当前日期寄存器
RTC_CNT	0x28	32	RO	0x00	RTC 秒标当前计数值
RTC_VALID	0x2C	32	RO	0x00	RTC 当前时间有效标志寄存器

寄存器描述

RTC_CFG 寄存器（0x00）

位域	名称	类型	复位值	描述
31:3	RESERVED	RO	0x0	保留位

2	LOAD_EN	R/W，AC	0x0	RTC 加载寄存器 RTC_TR 和 RTC_DR 时间设定值 1：加载寄存器设定的值，硬件自动清零 0：不加载寄存器设定的值 注：建议 RTC_TR 和 RTC_DR 配置完后，检查 RTC_IF 的 TIME_ ERR 位为 0，再打开此位
1	ALM_EN	R/W	0x0	RTC 闹钟功能使能位 1：打开闹钟功能，仅当 ALM_WEEKDAY 为 0，此位硬件自动清零 0：关闭闹钟功能 注：建议 RTC_AR 配置完后，检查 RTC_IF 的 ALM_ ERR 位为 0，再打开此位
0	RTC_EN	R/W	0x0	RTC 使能位 1：打开 RTC 0：关闭 RTC

RTC_IE 寄存器（0x04）

位域	名称	类型	复位值	描述
31:6	RESERVED	RO	0x0	保留位
5	MS_IE	R/W	0x0	1/2 秒中断使能位 1：使能 0：不使能
4	ALM_IE	R/W	0x0	闹钟中断使能位 1：使能 0：不使能
3	DATE_IE	R/W	0x0	日中断使能位 1：使能 0：不使能
2	HOUR_IE	R/W	0x0	小时中断使能位 1：使能 0：不使能

1	MIN_IE	R/W	0x0	分钟中断使能位 1：使能 0：不使能
0	SEC_IE	R/W	0x0	秒中断使能位 1：使能 0：不使能

RTC_IF 寄存器（0x08）

位域	名称	类型	复位值	描述
31:10	RESERVED	RO	0x0	保留位
9	ALM_ ERR	RO	0x0	闹钟设定有效标志位 1：闹钟设定错误 0：闹钟设定正确 注：此位不判断闹钟的 ALM_WEEKDAY
8	TIME_ERR	RO	0x0	时间设定有效标志位 1：时间设定错误 0：时间设定正确 注：若 BCD_WEEKDAY 配置为 7 此位也有效
7:6	RESERVED	RO	0x0	保留位
5	MS_IF	R，W1C	0x0	1/2 秒中断响应，高有效，写 1 清除
4	ALM_IF	R，W1C	0x0	闹钟中断响应，高有效，写 1 清除
3	DATE_IF	R，W1C	0x0	日中断响应，高有效，写 1 清除
2	HOUR_IF	R，W1C	0x0	小时中断响应，高有效，写 1 清除
1	MIN_IF	R，W1C	0x0	分钟中断响应，高有效，写 1 清除
0	SEC_IF	R，W1C	0x0	秒中断响应，高有效，写 1 清除

RTC_PRE 寄存器（0x10）

位域	名称	类型	复位值	描述
31:25	RESERVED	RO	0x0	保留位
24	PRE_PERIOD	R/W	0x0	小数计算周期选择 0：8 秒 1：16 秒 注：每隔多少秒进行一次小数部分计算
23:20	PRE_DECIMAL	R/W	0x0	预分频小数部分 0：表示无小数 1：表示 1/8 或 1/16 … 7：表示 7/8 或 7/16 8：表示 8/16 … 15：表示 15/16
19:0	PRE_ROUND	R/W	0x7fff	预分频整数部分 计数值为配置值加 1

RTC_TR 寄存器（0x14）

位域	名称	类型	复位值	描述
31:27	RESERVED	RO	0x0	保留位

26:24	BCD_WEEK	R/W	0x0	设定时间所属星期，0-6 为合法数据 0：星期天 1：星期一 2：星期二 3：星期三 4：星期四 5：星期五 6：星期六
23:22	RESERVED	RO	0x0	保留位
21:20	BCD_HOUR_DEC	R/W	0x0	设定时间所属小时的十位数（十进制），0-2 有效
19:16	BCD_HOUR	R/W	0x0	设定时间所属小时的个位数（十进制），0-9 有效
15	RESERVED	RO	0x0	保留位
14:12	BCD_MIN_DEC	R/W	0x0	设定时间所属分钟的十位数（十进制），0-5 有效
11:8	BCD_MIN	R/W	0x0	设定时间所属分钟的个位数（十进制），0-9 有效
7	RESERVED	RO	0x0	保留位
6:4	BCD_SEC_DEC	R/W	0x0	设定时间所属秒钟的十位数（十进制），0-5 有效
3:0	BCD_SEC	R/W	0x0	设定时间所属秒钟的个位数（十进制），0-9 有效

RTC_DR 寄存器（0x18）

位域	名称	类型	复位值	描述
31:24	RESERVED	R/W	0x0	保留位
23:20	BCD_YEAR_DEC	R/W	0x0	设定时间所属纪年的十位数（十进制），0-9 有效
19:16	BCD_YEAR	R/W	0x0	设定时间所属纪年的个位数（十进制），0-9 有效
15:13	RESERVED	RO	0x0	保留位。

12	BCD_MONTH_DEC	R/W	0x0	设定时间所属月份的十位数（十进制），0-1 有效
11:8	BCD_MONTH	R/W	0x1	设定时间所属月份的个位数（十进制），0-9 有效
7:6	RESERVED	RO	0x0	保留位。
5:4	BCD_DATE_DEC	R/W	0x0	设定时间所属日期的十位数（十进制），0-3 有效
3:0	BCD_DATE	R/W	0x1	设定时间所属日期的个位数（十进制），0-9 有效

RTC_AR 寄存器（0x1C）

位域	名称	类型	复位值	描述
31	RESERVED	RO	0x0	保留位
30:24	ALM_WEEKDAY	R/W	0x0	闹钟时间所属星期，每 bit 表示星期的其中一天 bit0=1：星期天 bit1=1：星期一 bit2=1：星期二 bit3=1：星期三 bit4=1：星期四 bit5=1：星期五 bit6=1：星期六 注：为全 0 时，闹钟仅响应一次；为其他值时，闹钟响应多次，直到 ALM_EN 配置关闭
23:22	RESERVED	RO	0x0	保留位
21:20	ALM_HOUR_DEC	R/W	0x0	闹钟时间所属小时的十位数（十进制），0-2 有效
19:16	ALM_HOUR	R/W	0x0	闹钟时间所属小时的个位数（十进制），0-9 有效
15	RESERVED	RO	0x0	保留位
14:12	ALM_MIN_DEC	R/W	0x0	闹钟时间所属分钟的十位数（十进制），0-5 有效
11:8	ALM _MIN	R/W	0x0	闹钟时间所属分钟的个位数（十进制），0-9 有效
7	RESERVED	RO	0x0	保留位

6:4	ALM_SEC_DEC	R/W	0x0	闹钟时间所属秒钟的十位数（十进制），0-5 有效
3:0	ALM _SEC	R/W	0x0	闹钟时间所属秒钟的个位数（十进制），0-9 有效

RTC_TSTR 寄存器（0x20）

位域	名称	类型	复位值	描述
31:27	RESERVED	RO	0x0	保留位。
26:24	WEEKDAY	RO	0x0	当前时间所属星期，0-6 为合法数据 0：星期天 1：星期一 2：星期二 3：星期三 4：星期四 5：星期五 6：星期六
23:22	RESERVED	RO	0x0	保留位。
21:20	HOUR_DEC	RO	0x0	当前时间所属小时的十位数（十进制），0-2 有效
19:16	HOUR	RO	0x0	当前时间所属小时的个位数（十进制），0-9 有效
15	RESERVED	RO	0x0	保留位。
14:12	MIN_DEC	RO	0x0	当前时间所属分钟的十位数（十进制），0-5 有效
11:8	MIN	RO	0x0	当前时间所属分钟的个位数（十进制），0-9 有效
7	RESERVED	RO	0x0	保留位。
6:4	SEC _DEC	RO	0x0	当前时间所属秒钟的十位数（十进制），0-5 有效
3:0	SEC	RO	0x0	当前时间所属秒钟的个位数（十进制），0-9 有效

RTC_TSDR 寄存器（0x24）

位域	名称	类型	复位值	描述
31:25	RESERVED	RO	0x0	保留位
24	LEAPYEAR	RO	0x0	
23:20	YEAR_DEC	RO	0x0	当前时间所属的纪年的十位数（十进制），0-9 有效
19:16	YEAR	RO	0x0	当前时间所属的纪年的个位数（十进制），0-9 有效
15:13	RESERVED	RO	0x0	保留位
12	MONTH_DEC	RO	0x0	当前时间所属的月份的十位数（十进制），0-1 有效
11:8	MONTH	RO	0x1	当前时间所属的月份的个位数（十进制），0-9 有效
7:6	RESERVED	RO	0x0	保留位
5:4	DATE_DEC	RO	0x0	当前时间所属的日期的十位数（十进制），0-3 有效
3:0	DATE	RO	0x1	当前时间所属的日期的个位数（十进制），0-9 有效

RTC_CNT 寄存器（0x28）

位域	名称	类型	复位值	描述
31:20	RESERVED	RO	0x0	保留位
19:0	CNT_20	RO	0x0	20bit 计数位

RTC_VALID 寄存器（0x2C）

位域	名称	类型	复位值	描述

31:1	RESERVED	RO	0x0	保留位
0	CUR_VALID	RO	0x0	当前时间有效标志位 当判断该位为 1 时，可以读取当前时间寄存器（RTC_TSTR、RTC_TSDR、RTC_CNT）

5.16 UART 控制器（UART）

5.16.1 概述

通用异步收发器（Universal Asynchronous Receiver/Transmitter，UART）是串行通信一种通信技术，常用于单片机和电脑之间以及单片机和单片机之间的板级通信，其为异步通讯，端口能够在一根线上发送数据同时在另一根线上接收数据。串口通信最重要的参数是波特率、数据位、停止位和奇偶校验。本芯片具有 3 路串口，支持波特率配置，数据长度、校验位、停止位可配置，支持多种中断，支持 DMA 传输机制，具有硬件自动流控制功能，接收和发送分别独立具有深度为 8 的 FIFO，并且 FIFO 水位可配置，使用前需要使能对应的 UART 时钟。其模块系统连接示意图如下所示：

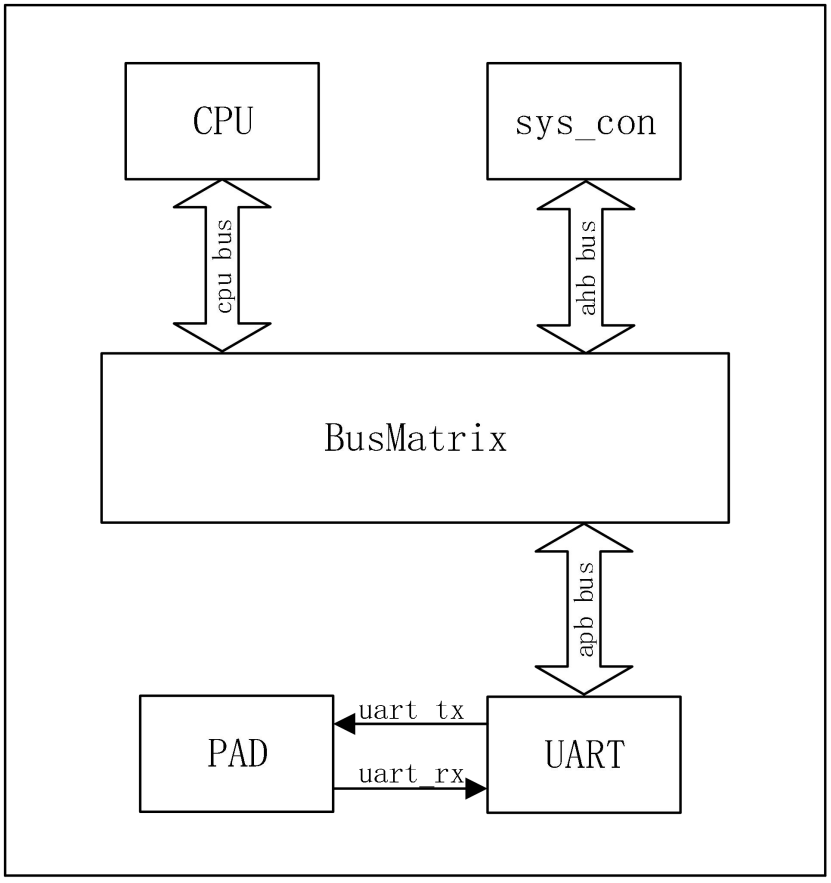


图 5-115 UART 模块系统框图

上图为 UART 模块系统连接示意图。其中 CPU 可通过 APB 总线接口直接控制 UART 模块，UART 模块通过 PAD 实现芯片与外部设备的通讯。

5.16.2特性

- 支持标准的 UART 协议
- 支持全双工模式
- 支持波特率配置
- 可编程数据位长度（8bit 或 9bit）
- 可配置奇偶校验位　奇校验　偶校验　常 0　常 1
- 支持 1 位停止位，并且支持发送延迟时间可配置
- 支持波特率自动侦测功能
- 具有硬件自动流控制功能
- 支持 DMA 传输机制
- 接收和发送分别独立具有深度为 8 的 FIFO，并且 FIFO 水位可配置
- 支持发送完成中断、接收完成中断、接收超时中断功能

5.16.3模块结构框图

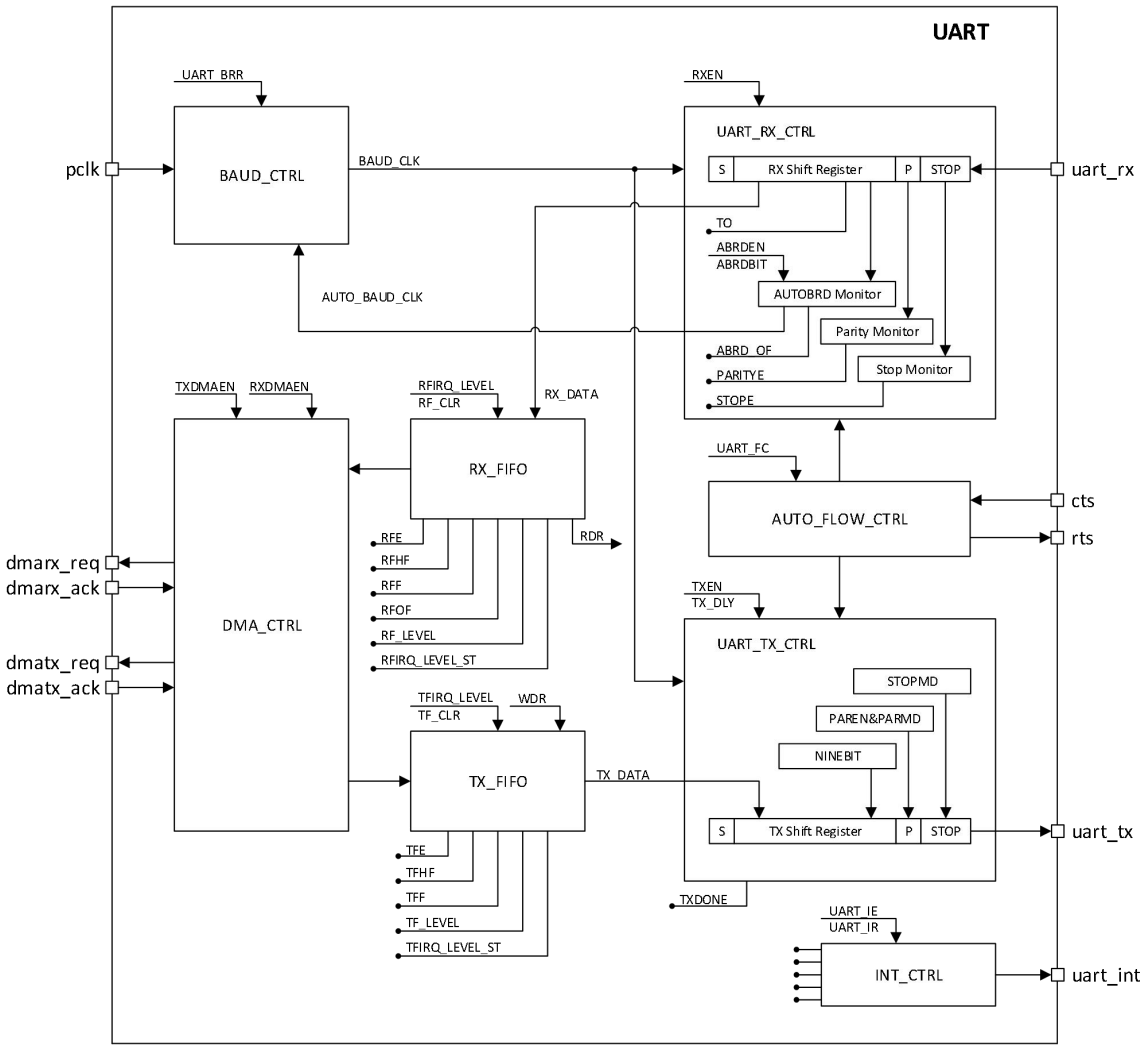


图 5-116 UART 模块结构框图

UART 通过内部波特率控制器 BAUD_CTRL，根据 UART_BAUD 所设置的波特率参数，产生采样时钟 BAUD_CLK，可供接收端 UART_RX_CTRL 进行数据采样、发送端 UART_TX_CTRL 进行数据发送。

UART 内置两个深度为 8 的 FIFO，用于存储 UART_RX_CTRL 采样到的有效数据的 RX_FIFO 和用于存储 CPU 或 DMA 写入的待发送数据的 TX_FIFO。

数据接收端 UART_RX_CTRL 对 UART_RX 的数据进行同步同时采样数据，对采样数据进行分析，检查数据校验位、停止位是否与所设置的模式相同，从而产生相应状态位，并把没有出现校验位错误、停止位错误的有效数据 RX_DATA 存入 RX_FIFO。

数据发送端 UART_RX_CTRL 收到 RX_FIFO 的待发送数据，根据所设置数据位、校验位及停止位按照 BAUD_CLK 逐个 BIT 发送。

UART 支持 DMA 功能，通过内部的 DMA_CTRL 以及 DMA 的对应接口实现 DMA 控制 UART 数据接收与发送。

UART 通过内部 AUTO_FLOW_CTRL 实现硬件流控。

功能描述

接收时序及相关状态信号

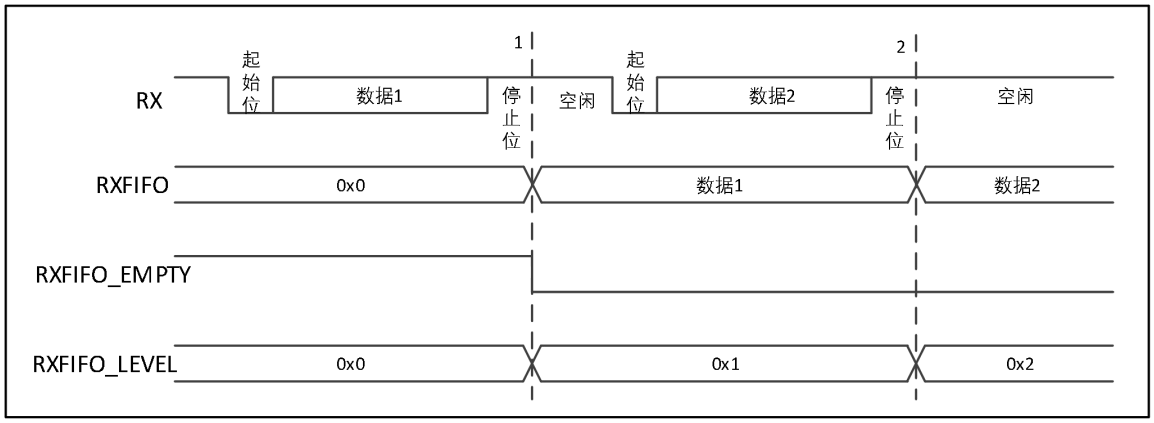


图 5-117 UART 接收时序及相关状态信号图

上图为 UART 接收简单时序图，在标志 1 处 UART 将接收到的数据保存到 RXFIFO 中，RXFIFO 水位增加，RXFIFO 空被置低；标志 2 处，UART 将接收到的数据保存到 RXFIFO 中，RXFIFO 水位增加。

发送时序及相关状态信号

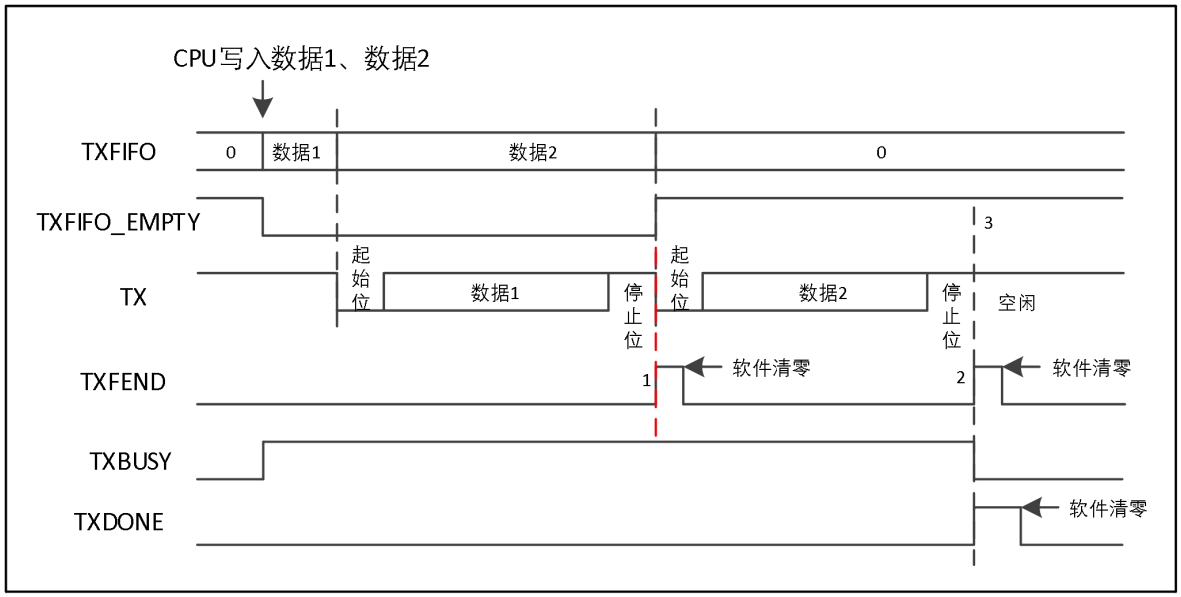


图 5-118 UART 发送时序及相关状态信号图

上图为 UART 发送简单时序图，每发送完一组数据，TXFEND 会被置为 1，见上图 1 和 2 处所示。当全部数据发送完成时，TXDONE 会被置为 1。

当 TXFIFO 中有待发送数据，并且发送移位寄存器正在传输过程中，则 TXBUSY 会被置为有效。

波特率计算

波特率计算公式如下：

$$f_{baudrate} = \frac{f_{pclk}}{UARTDIV}$$

例如：f_{pclk}=48M，波特率设置为 115200，则 UARTDIV=48000000/115200=416.6，根据四舍五入可以选择 417。

自动波特率检测

自动波特率检测功能可以自动测量 uart_rx 管脚输入的数据从而计算波特率。当自动波特率测量结束，测量结果会放入 UART_BAUD 寄存器中。

在自动波特率检测中，将检测 uart_rx 数据从起始位的下降沿开始到第一个上升沿结束的时间，该时间可由 ABRDBIT 来确定。

通过配置 ABRDEN 可以开启自动波特率检测功能。在初始阶段，uart_rx 保持为 1，一段检测到下降沿，即为接收到起始位，自动波特率计数器被启动并开始计数，当检测到第一个上升沿时，自动波特率计数器将停止计数。然后，自动波特率计数器值除以 ABRDBIT 的结果自动存入 UART_BAUD 寄存器中，并且 ABRDEN 会被清零，并产生自动波特率检测结束标志。

自动波特率计算示意图如下：

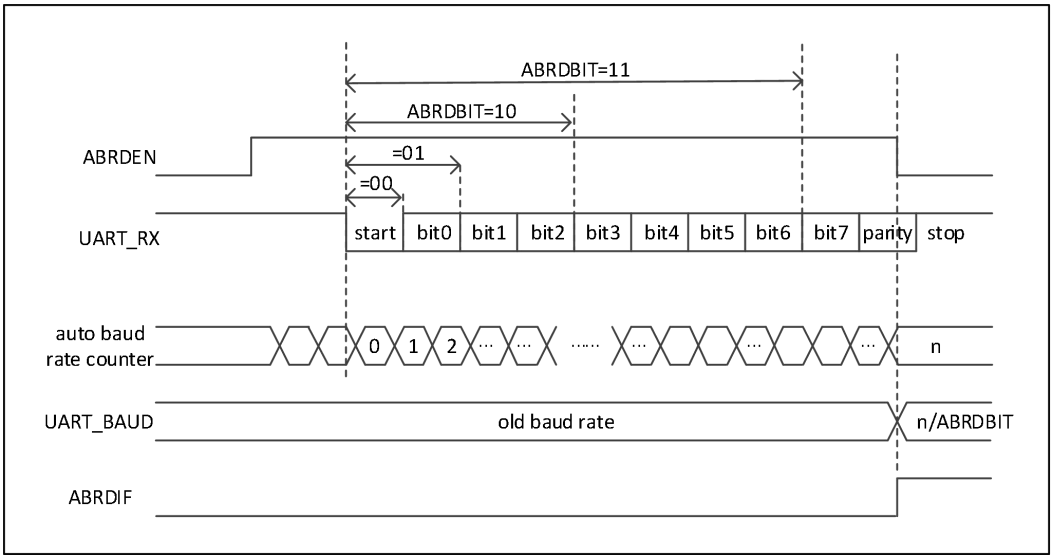


图 5-119　UART 自动波特率计算示意图

数据采样

UART 为异步通信接口，没有独立的操作时钟，因此需要对接收到的数据进行同步处理。为了能够正确获得接收到的字符数据，UART 采用 16 倍于数据波特率的时钟对 UART_RX 进行数据采样，每个 bit 数据位有 16 个采样时钟，取中间第 8，9，10 次采样的数据作为实际收到的数据使用。

数据采样的示意图如下：

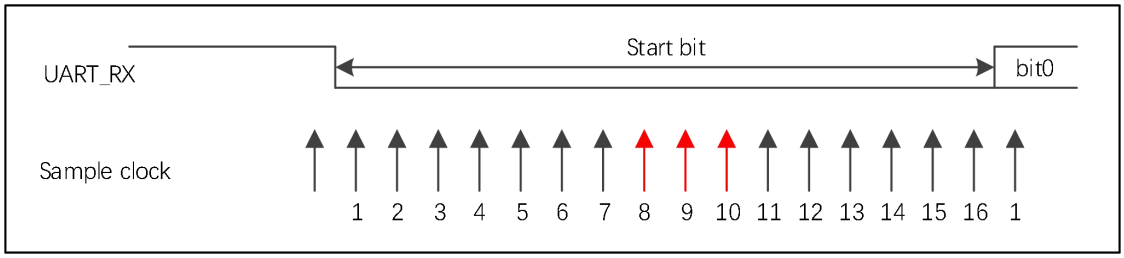


图 5-120 UART 数据采样示意图

对于每 bit 数据来说，都会根据波特率的设置进行 16 次采样，取采样的第 8 次，第 9 次和第 10 次数据进行判断：若三次采样中至少包含两次 0，则该 bit 位为 0；若三次采样中至少包含两次 1，则该 bit 位为 1。

硬件自动流控制

UART 通过 CTS 输入和 RTS 输出可以控制 2 个设备之间的异步串行的数据流。如下图所示：

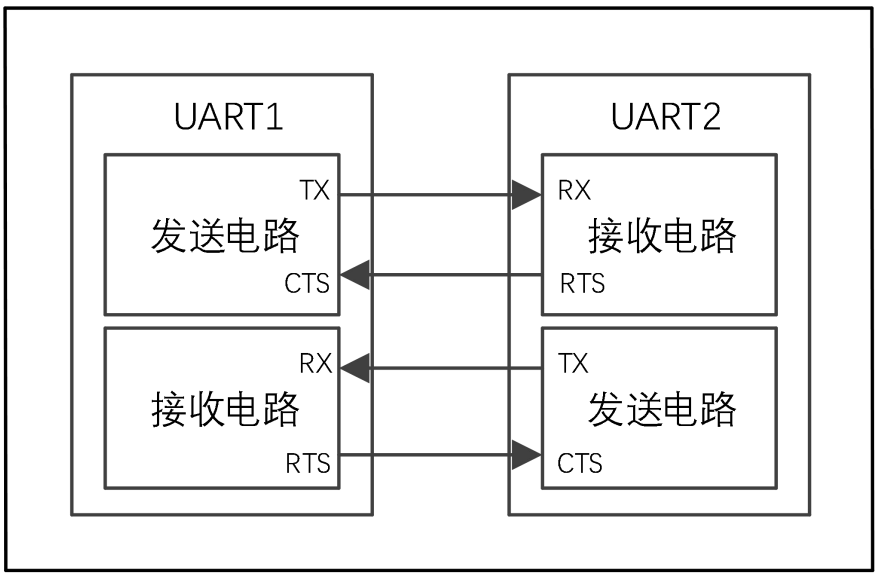


图 5-121　UART 自动流控连接图

RTS 流控制：

当配置 RTSEN 为 1 后，数据接收 FIFO 有空间，则 RTS 则输出有效电平（电平可配置）。当数据接收 FIFO 满时，RTS 则输出为无效电平，以此表示希望在当前组数据结束时停止数据传输。

启动 RTS 流控制通信示例如下：

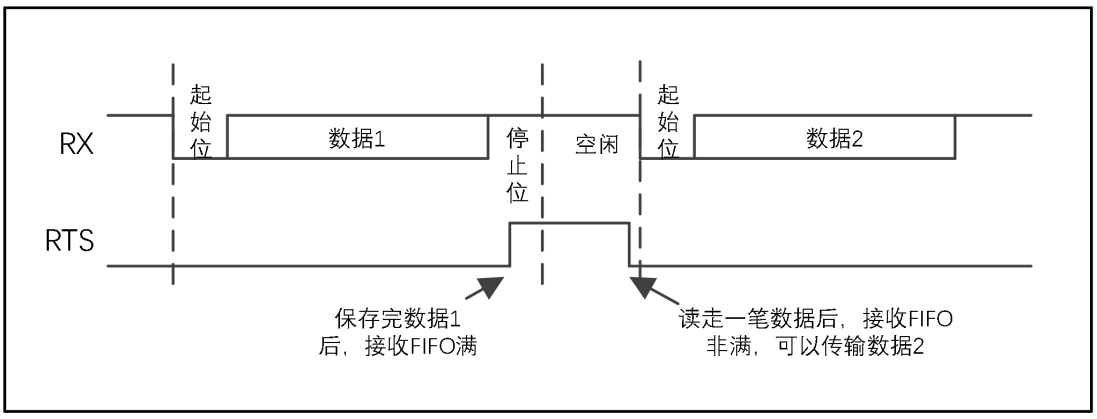


图 5-122 UART 流控接收示意图

CTS 流控制：

当配置 CTSEN 为 1 后，发送电路在发送下一组数据前都要检查输入的 CTS 电平。如果 CTS 有效（电平可配置），则发送 FIFO 中的下一组数据可以被发出，否则一组数据不能被发出去。若 CTS 在数据发送期间变成了无效，则当前传输完成后停止下一组数据发送。

启动 CTS 流控制通信示例如下：

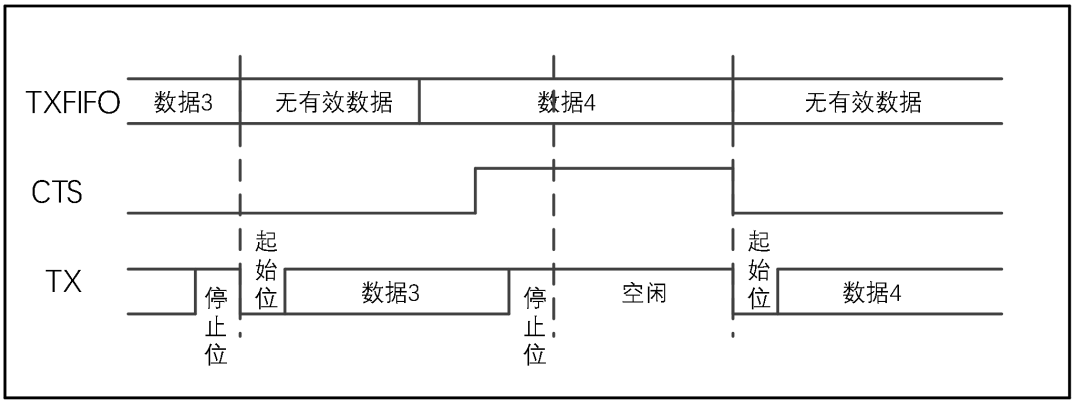


图 5-123 UART 流控发送示意图

接收超时说明

两种情况会产生接收超时：

1、当接收使能有效后，至少收到一个字节数据后超时中断才能生效（超时中断使能打开时）。当时间超过配置的超时时间时，产生超时中断，用户可在中断服务函数中进行处理，清除超时中断标志。

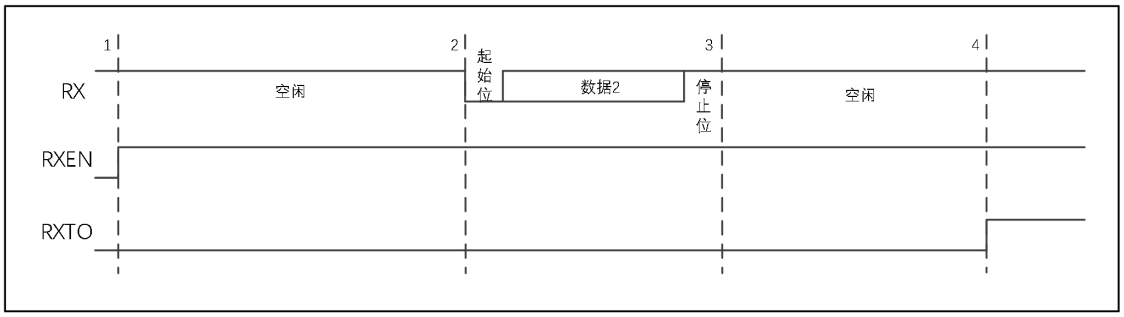


图 5-124 UART 第一次接收超时中断

如上图所示：

1 时刻 UART 接收使能打开，直到 2 时刻前 UART 接收一直空闲，这个区间内是不会产生超时中断的；2 时刻 UART 收到第一个数据，直到 3 时刻数据接收完毕，此后又进入空闲此时超时计数器会开始计数，直到 4 时刻产生超时中断。

2、当发生过接收超时中断后，只有再次至少接收到一个字节数据后，超时中断才能再次生效（超时中断使能打开时）。

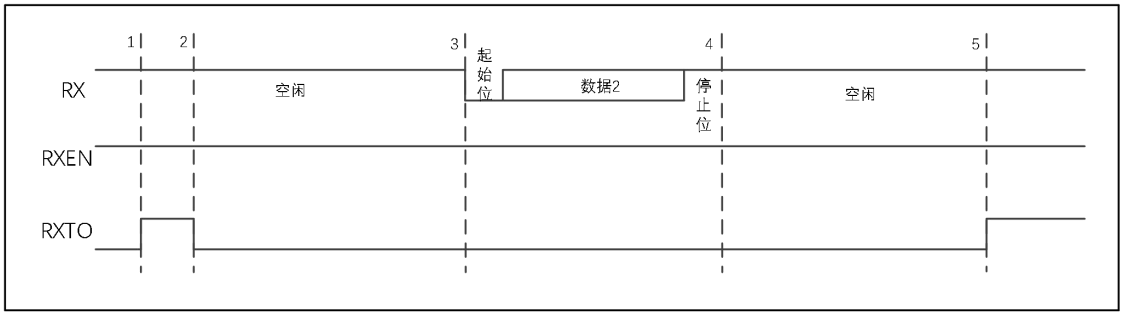


图 5-125　UART 第二次接收超时中断

如上图所示：

1 时刻 UART 产生 RXTO 接收超时中断，并于 2 时刻由 CPU 进行清除，直到 3 时刻前 UART 接收一直空闲，但是这个区间内是不会产生超时中断的；3 时刻 UART 收到第一个数据，直到 4 时刻数据接收完毕，此后又进入空闲此时超时计数器会开始计数，直到 5 时刻产生超时中断。

中断

UART 提供了 8 种中断源，它们的关系如下图：

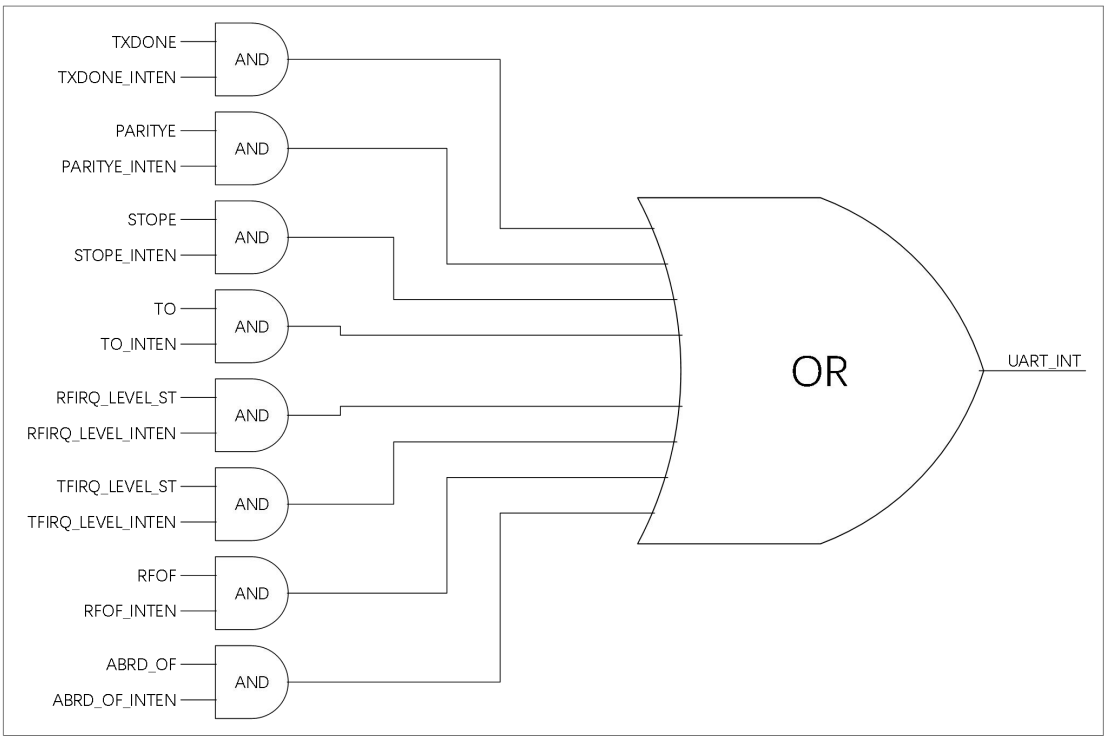


图 5-126　UART 中断标志及中断示意图

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
UART0：　BASE: 0x4006B000 UART1：　BASE: 0x4006B800 UART2：　BASE: 0x4006C000					
UART_CTRL	0x00	32	R/W	0x00	UART 控制寄存器
UART_BAUD	0x04	32	R/W	0x00	UART 波特率配置寄存器
UART_TDR	0x08	32	W	0x00	UART 写数据寄存器
UART_RDR	0x0c	32	R	0x00	UART 读数据寄存器
UART_IE	0x10	32	R/W	0x00	UART 中断使能寄存器
UART_IF	0x14	32	R/W	0x2400	UART 中断状态寄存器
UART_FIFO	0x18	32	R/W	0x07	UART FIFO 控制寄存器
UART_FC	0x1c	32	R/W	0x00	UART 流控制配置寄存器
UART_RXTO	0x20	32	R/W	0xff	UART 接收超时配置寄存器

寄存器描述

UART_CTRL 寄存器（0x00）

位域	名称	类型	复位值	描述
31:17	RESERVED	R	0	保留位
16:14	TX_DLY	R/W	0	发送延迟时间设置（用于设置上一次停止位到下一次开始位之间的传输延迟时间） 000：无延迟 001：1bit 延迟 010：2bit 延迟 …… 111：7bit 延迟
13:12	ABRDBIT	R/W	0	自动波特率检测位长配置 11：8 位时长，从起始位到第一个上升沿。输入数据为 0x80。 10：4 位时长，从起始位到第一个上升沿。输入数据为 0x08。 01：2 位时长，从起始位到第一个上升沿。输入数据为 0x02。 00：1 位时长，从起始位到第一个上升沿。输入数据为 0x01。
11	ABRDEN	R/W	0	自动波特率检测使能 1：自动波特率检测功能使能 0：自动波特率检测功能禁止 自动检测结束后，该位将被自动清零。
10:9	RESERVED	R	0	保留位

8:7	PARMD	R/W	0	奇偶校验模式选择 11：常 0 10：常 1 01：偶校验 00：奇校验
6	PAREN	R/W	0	奇偶校验位使能 1：带奇偶校验位 0：不带奇偶校验位
5	NINEBIT	R/W	0	9bit 数据模式使能 1：9bit 数据模式 0：8bit 数据模式
4	TXDMAEN	R/W	0	发送 DMA 传输使能 1：表示 DMA 操作 UART 的发送数据寄存器 0：表示 CPU 操作 UART 的发送数据寄存器
3	RXDMAEN	R/W	0	接收 DMA 传输使能 1：表示 DMA 操作 UART 的接收数据寄存器 0：表示 CPU 操作 UART 的接收数据寄存器
2	TXEN	R/W	0	发送使能位 1：发送打开。将 tx_fifo 中保存的数据通过 uart_tx 发送出去。 0：发送关闭。不发送数据。uart_tx 信号保持 1。
1	RXEN	R/W	0	接收使能位 1：接收打开。可通过 uart_rx 接收外来的数据 0：接收关闭。不接收 uart_rx 的数据
0	UARTEN	R/W	0	UART 使能位 1：使能 UART 模块 0：关闭 UART 模块

UART_BAUD 寄存器（0x04）

位域	名称	类型	复位值	描述
31:16	RESERVED	R	0	保留位

15:0	BAUD	R/W	0	波特率配置数据

UART_TDR 寄存器（0x08）

位域	名称	类型	复位值	描述
31:9	RESERVED	R	0	保留位
8:0	TDR	W	0	写数据寄存器 将要发送的数据通过该寄存器写入发送 FIFO 中。准备发送数据时，移位寄存器直接读取发送 FIFO 进行数据的发送。

UART_RDR 寄存器（0x0C）

位域	名称	类型	复位值	描述
31:9	RESERVED	R	0	保留
8:0	RDR	R	0	读数据寄存器 通过该寄存器读出接收到的数据。移位寄存器在每次传输完成后，将数据存入接收 FIFO 中，通过该寄存器读出接收 FIFO 中的数据。 注：若发生奇偶校验位错或停止位错，则该组数据不会写入 RXFIFO。若 RXFIFO 不为空，则 RXFIFO 中的数据是有效数据。

UART_IE 寄存器（0x10）

位域	名称	类型	复位值	描述
31:10	RESERVED	R	0	保留位

9	ABRD_OVF	R/W	0	自动波特率检测功能计数器溢出中断使能
8	RXFIFO_OVF	R/W	0	接收 FIFO 溢出中断使能
7	TXFIFO	R/W	0	发送 FIFO 中保存的数据达到设定水位中断使能
6	RXFIFO	R/W	0	接收 FIFO 中接收的数据达到设定水位中断使能
5	RXTO	R/W	0	接收超时中断使能
4	STOPE	R/W	0	接收数据出现停止位错误中断使能
3	PARITYE	R/W	0	接收数据出现奇偶校验错中断使能
2	TXDONE	R/W	0	全部数据发送完成中断使能（表示发送移位寄存器中数据发送完成并且发送数据 FIFO 中也没有待发数据）
1	RESERVED	R/W	0	保留位
0	RESERVED	R/W	0	保留位

UART_IF 寄存器（0x14）

位域	名称	类型	复位值	描述
31:23	RESERVED	R	0	保留位
22:20	TF_LEVEL	R	0	发送 FIFO 水位标志信号 000：未满时表示 FIFO 中有 0 个数据，满时表示 FIFO 有 8 个数据； 001：表示 FIFO 有 1 个数据； …… 111：表示 FIFO 有 7 个数据；

19:17	RF_LEVEL	R	0	接收 FIFO 水位标志信号 000：未满时表示 FIFO 中有 0 个数据，满时表示 FIFO 有 8 个数据； 001：表示 FIFO 有 1 个数据； …… 111：表示 FIFO 有 7 个数据；
16	TXBUSY	R	0	数据发送忙标志 1：发送 FIFO 非空，或有数据正在发送 0：发送 FIFO 空，并且没有正在发送的数据
15	TXFIFO_HFULL	R	0	发送 FIFO 半满标志
14	TXFIFO_FULL	R	0	发送 FIFO 满标志
13	TXFIFO_EMPTY	R	0x1	发送 FIFO 空标志
12	RXFIFO_HFULL	R	0	接收 FIFO 半满标志
11	RXFIFO_FULL	R	0	接收 FIFO 满标志
10	RXFIFO_EMPTY	R	0x1	接收 FIFO 空标志
9	ABRD_OVF	R/W	0	自动波特率检测功能计数器溢出标志 1：计数器溢出，检测失败 0：计数器未溢出 写 1 清零
8	RXFIFO_OVF	R/W	0	接收 FIFO 溢出标志 写 1 清零 注：溢出后的数据将被丢弃
7	TXFIFO	R	0	发送 FIFO 中保存的数据达到设定水位时，该位为 1，否则为 0
6	RXFIFO	R	0	接收 FIFO 中接收到的数据达到设定水位时，该位 1，否则为 0
5	RXTO	R/W	0	接收超时标志 1：发送接收超时 写 1 清零

4	STOPE	R/W	0	接收数据出现停止位错误 写 1 清零 主要指停止位没有在预期的时间段内接收到或识别出来，则认为停止位错误
3	PARITYE	R/W	0	接收数据出现奇偶校验错 写 1 清零
2	TXDONE	R/W	0	全部数据发送完成（表示发送移位寄存器中数据发送完成并且发送数据 FIFO 中也没有待发数据） 写 1 清零
1	RESERVED	R/W	0	保留位
0	RESERVED	R/W	0	保留位

UART_FIFO 寄存器（0x18）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7	TF_CLR	R/W	0	TXFIFO 清除使能 软件写 1 清发送 FIFO，硬件自动清零
6	RF_CLR	R/W	0	RXFIFO 清除使能 软件写 1 清接收 FIFO，硬件自动清零
5:3	TF_LEVEL	R/W	0	用于 TXFIFO 产生中断的水位设置 000：0 001：1 010：2 …… 111：7 表示发送 FIFO 中待发送的数据个数不多于水位设置值 例如设置值为 011，则 TXFIFO 中已写入的数据个数小于等于 3 个时产生相应信号

2:0	RF_LEVEL	R/W	0x7	用于 RXFIFO 产生中断的水位设置 000：1 001：2 010：3 …… 111：8 表示接收 FIFO 中的接收到的数据个数至少达到水位设置值 例如设置值为 011，则 RXFIFO 中数据至少为 4 个时产生相应信号

UART_FC 寄存器（0x1C）

位域	名称	类型	复位值	描述
31:6	RESERVED	R	0	保留位
5	RTS_SIGNAL	R	0	表示线上 RTS 状态 1：RTS 为高电平 0：RTS 为低电平
4	CTS_SIGNAL	R	0	表示线上 CTS 状态 1：CTS 为高电平 0：CTS 为低电平
3	RTSPOL	R/W	0	RTS 信号极性配置 1：RTS 信号输出为高时，UART 可以接收数据；RTS 信号输出为低时，UART 接收 FIFO 已满，不能再接收数据。 0：RTS 信号输出为低时，UART 可以接收数据；RTS 信号输出为高时，UART 接收 FIFO 已满，不能再接收数据。
2	CTSPOL	R/W	0	CTS 信号极性配置 1：CTS 信号输入为高时，UART 可以发送数据；CTS 信号输入为低时，UART 不发出数据。 0：CTS 信号输入为低时，UART 可以发送数据；CTS 信号输入为高时，UART 不发出数据。

1	RTSEN	R/W	0	RTS 流控使能 1：RTS 信号发挥流控作用 0：RTS 信号不起作用
0	CTSEN	R/W	0	CTS 流控使能 1：CTS 信号发挥流控作用 0：CTS 信号不起作用

UART_RXTO 寄存器（0x20）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	RXTO	R/W	0xff	接收数据超时触发比较值 当 RXFIFO 接收到一个新的数据，定时器被清零并重新开始计时（计时时钟为一个数据的时长）。在定时器计时超过该配置值时，还没有接收到有效起始位则产生接收超时标志。

5.17 SPI 总线控制器（SPI）

5.17.1 概述

串行外设接口（Serial Peripheral Interface，SPI）是外部设备通过 2 线交换 8 位数据的串行同步通讯手段。芯片提供了一个 SPI 接口模块，可配置为主设备或从设备，实现与外部的 SPI 通信。

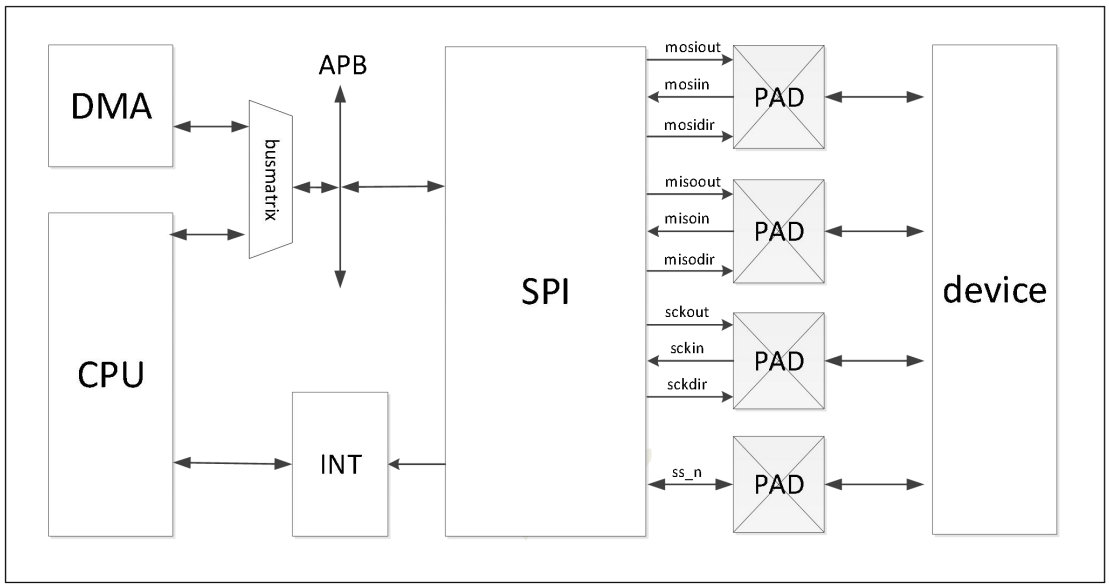


图 5-127 SPI 模块系统框图

上上图为 SPI 模块系统连接示意图。其中 CPU 可通过 APB 总线接口直接控制 SPI 模块，SPI 模块通过 PAD 实现芯片与外部设备的通讯。根据所配置的模式不同，SPI 模块可配置各 PAD 的方向。

5.17.2 特性

- 支持主机模式和从机模式
- 可编程时钟极性和相位
- 主模式速率可配，最大频率为系统时钟 4 分频
- 数据传输顺序可配置

- 传输结束中断标志
- 读数据寄存器和写数据寄存器分开
- 接收和发送分别采用 8 级 FIFO 缓存机制
- 具有 DMA 传输接口

5.17.3模块结构框图

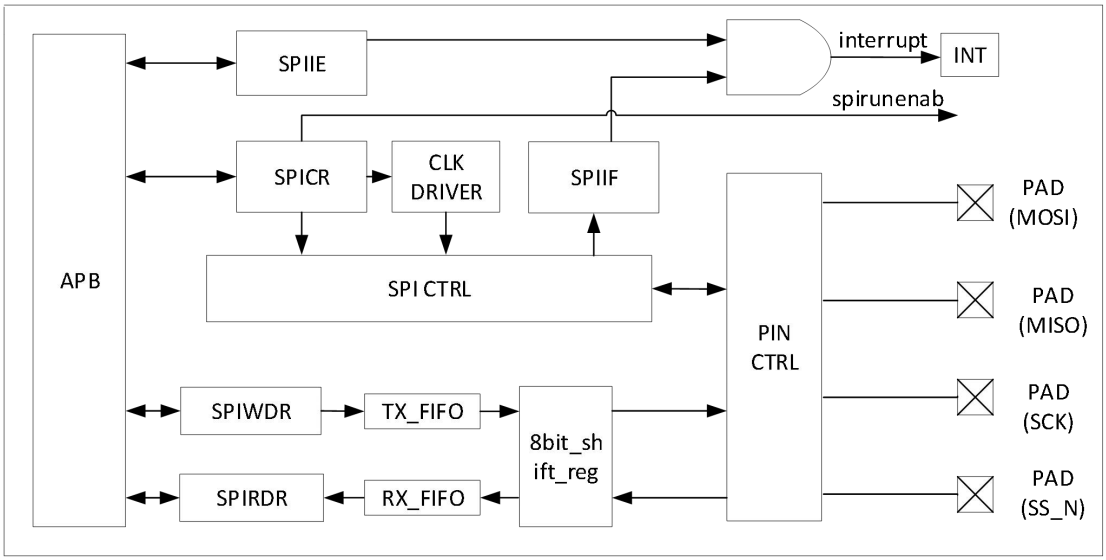


图 5-128 SPI 模块结构框图

如上图所示，为 SPI 模块的结构示意图，可通过 SPI 控制寄存器 SPICR 来配置模块的主从模式，传输方式、FIFO 清除和其他功能等。数据的写通过 SPIWDR 的操作完成，准备发送数据时，SPI 将要发送的数据通过 SPIWDR 寄存器写入发送 FIFO 中，移位寄存器读取 FIFO 的值，进行数据的发送；数据的读通过 SPIRDR 的操作完成，移位寄存器在每次传输完成后，将数据存入接收 FIFO 中，通过 SPIRDR 寄存器读出接收 FIFO 中的数据。模块的中断信号受中断使能寄存器 SPIIE 控制，打开中断使能，当有相应的中断状态后就会产生中断信号。

5.17.4功能描述

SPI 接口时序

不同的 SPI 外设，SPI 串行时钟的时序可以通过时钟相位选择位（SPICR.CPHA）和时钟极性选择位(SPICR.CPOL)设置产生 4 种不同组合。为保证数据正确传输，主从器件的时序配置必须一致。

器件模式或 SPI 系统使能位（SPICR.SPE）位为 0 时，SPI 的 SCK 引脚无串行时钟输出。

1：CPHA=0 时，SPI 模块在串行时钟的第一个跳变沿采样数据，即：

若 CPOL=1，在串行时钟的下降沿采样数据；

若 CPOL=0，在串行时钟的上升沿采样数据。如下图所示：

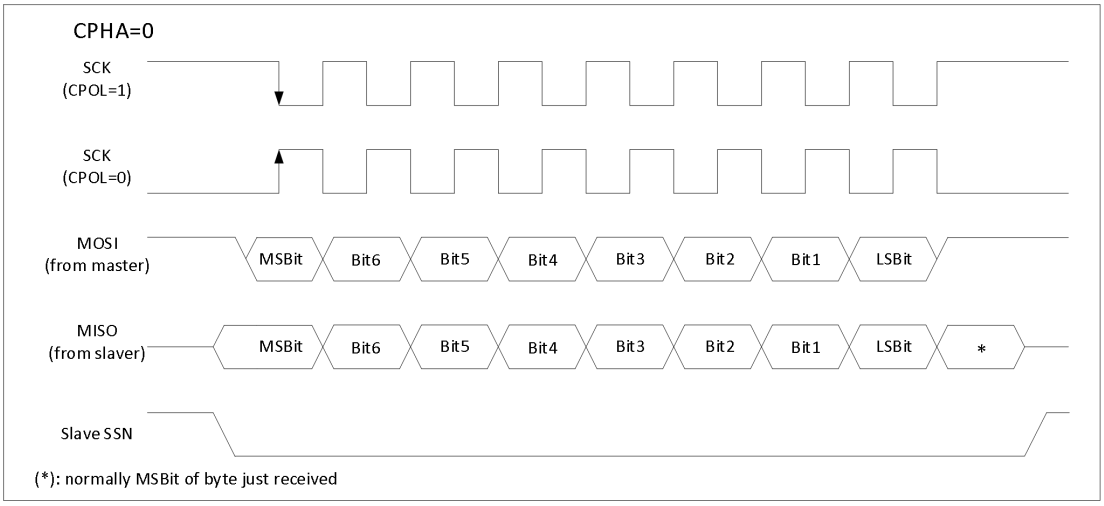


图 5-129　SPI 数据/时钟时序图（CPHA=0）

2：CPHA=1 时，SPI 模块在串行时钟的第二个跳变沿采样数据，即：

若 CPOL=1，在串行时钟的上升沿采样数据；

若 CPOL=0，在串行时钟的下降沿采样数据。如下图所示：

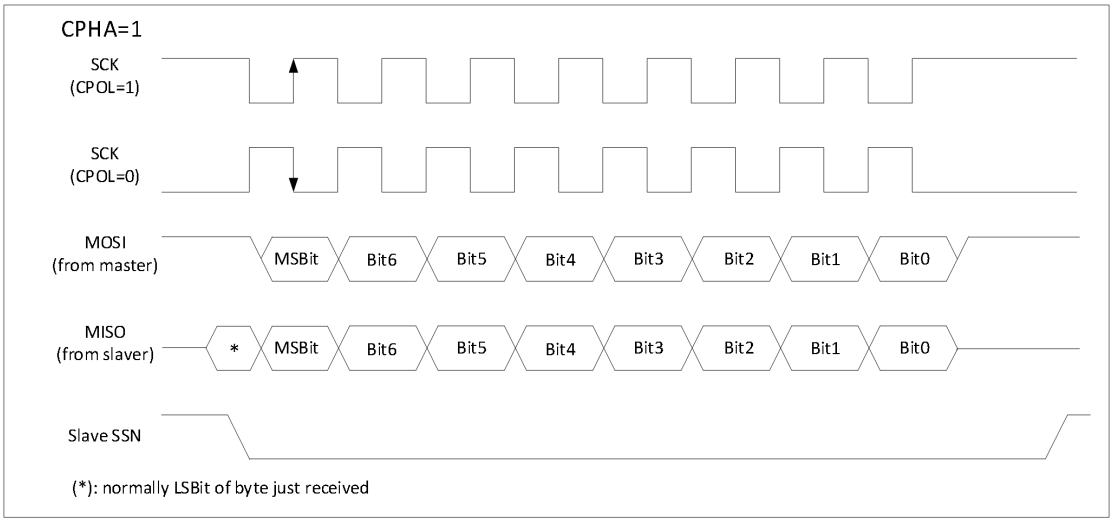


图 5-130 SPI 数据/时钟时序图（CPHA=1）

3：从器件 SSN

若 SPI 为从器件，从器件的 SSN 引脚可以在连续数据传输时一直为低。如下图所示：

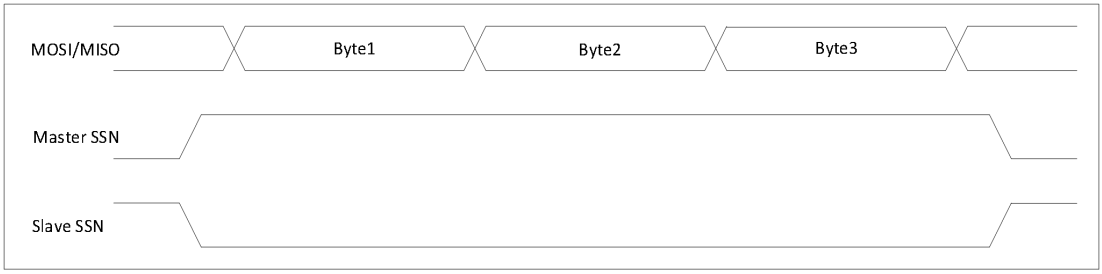


图 5-131 SPI SSN 时序图（CPHA=0）

I/O 配置

主输出、从输入（MOSI）

主出从入（MOSI）引脚是主器件的输出和从器件的输入，用于主器件到从器件的串行数据传输。当 SPI 配置为主器件时，该引脚为输出，当 SPI 配置为从器件时，该引脚为输入。

主输入、从输出（MISO）

主入从出（MISO）引脚是从器件的输出和主器件的输入，用于从器件到主器件的串行数据传输。当 SPI 配置为主器件时，该引脚为输入，当 SPI 配置为从器件时，该引脚为输出。

串行时钟（SCK）

串行时钟（SCK）引脚是主器件的输出和从器件的输入，用于同步主器件和从器件之间在 MOSI 和 MISO 线上的串行数据传输。当 SPI 配置为主器件时，该引脚输出时钟，当 SPI 配置为从器件时，该引脚为输入。

从选择（SSN）

从选择（SSN）引脚用来控制从器件选中，当 SPI 配置为主器件时，SSN 引脚可以通过寄存器的方式控制从器件的选中与否，当 SPI 配置为从器件时，SSN 引脚来源于主器件的控制信号。

SPI 主从器件的连接如图下所示：

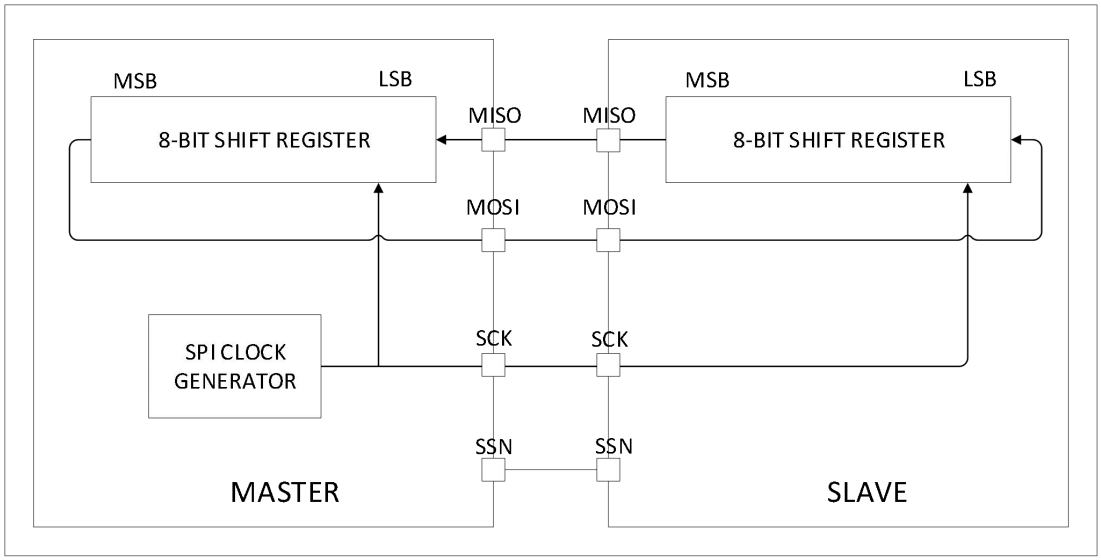


图 5-132　SPI Master/SPI Slave 互连

主从器件的 MOSI、MISO、SCK 和 SSN 分别连在一起。主从器件通过 MOSI、MISO 连成一个环路，主器件输出时钟，数据传输时，主器件通过 MOSI 输出数据，从器件通过 MISO 输出数据。一个字节数据传输完毕，主从器件将交换 8 位移位寄存器数值。

数据传输配置

1：进行数据传输前需先配置 SPICR.SPE 位和 SPICR.MSTR 位，以使能 SPI 和设置主从模式。

2：配置 SPICR.CPHA 位和 SPICR.CPOL 位，以设置串行时钟相位和极性（主从器件需一致）。

3：配置 SPICR.SPR[2:0]位，以设置串行时钟波特率（若为从器件模式则不用设置，串行时钟速率由主器件决定）。

4：配置 SPICR.LSB 位，设置传输顺序，配置 SPICR.CPHA_DATAHOLD_S，来设置从模式下传输数据保持周期数目。

需要时，配置中断，配置 SPIIE 和 SPIIR 位。

主器件模式下数据传输前需先将从器件的 SSN 引脚拉低。主器件模式下 MCU 写 SPIWDR 寄存器的动作启动数据传输，中断标志 SPIIR.SPIF 置起完成数据传输。

从器件模式处理较为特殊，当 CPHA=0 时，从器件的 SSN 引脚拉低启动数据传输，从器件的 SSN 引脚拉高结束数据传输（即使在此之前 SPIIR.SPIF 中断已经产生），因为从器件不知道传输何时开始，当 SSN 引脚拉低后，MISO 引脚立即开始数据 MSB 的传输。

当 CPHA=1 时，从器件在串行时钟的第一个沿启动数据传输，在 SPIIR.SPIF 置位后结束数据传输。

中断产生

SPI 模块提供 5 种中断源，中断源示意图如下:

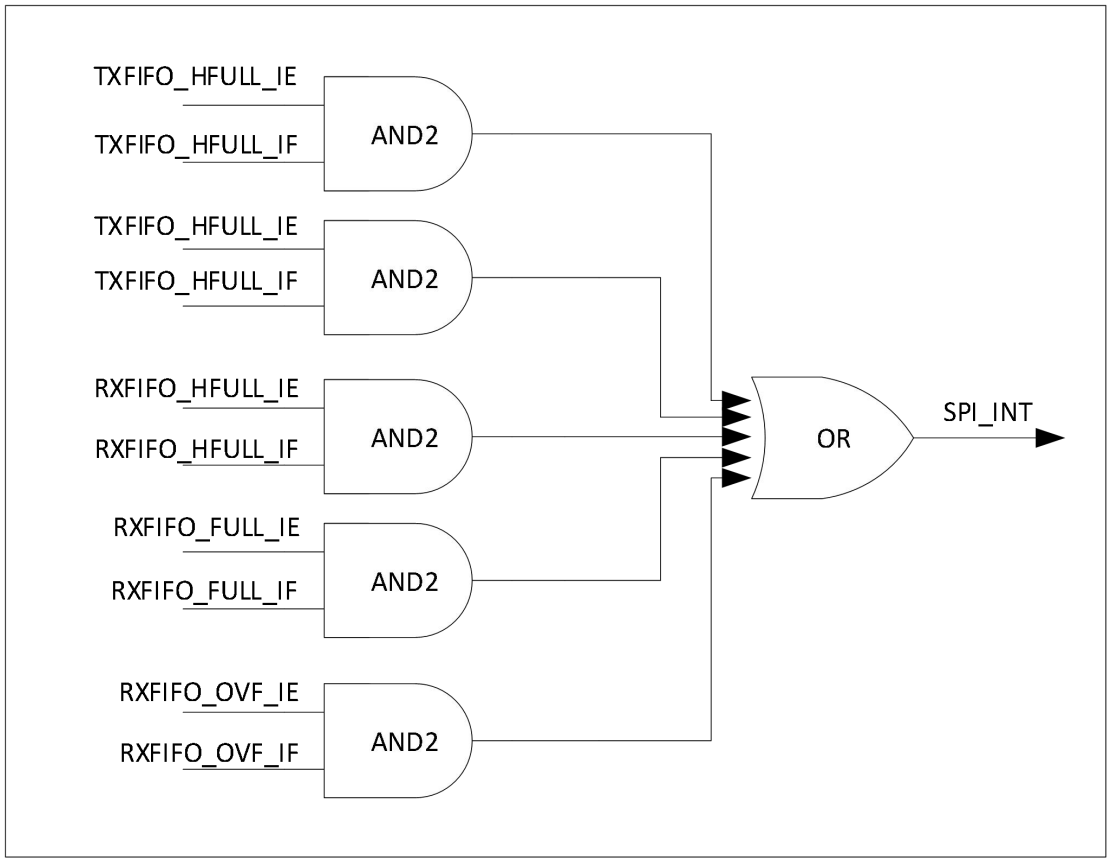


图 5-133　SPI 中断标志及中断示意图

接收 FIFO 满中断

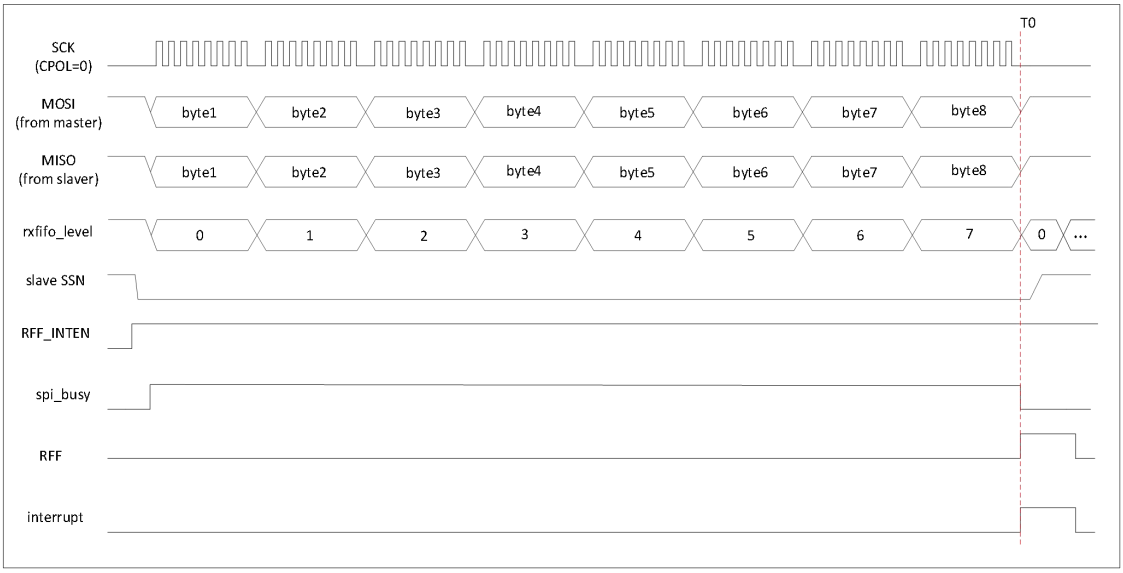
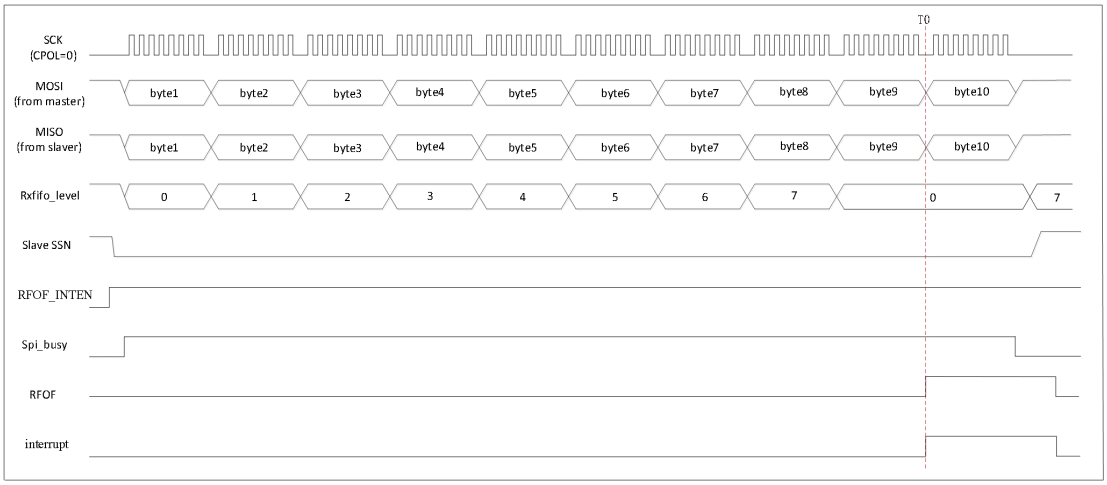


图 5-134 SPI 接收 FIFO 满中断

本模块在数据传输过程中，接收的数据会通过移位寄存器保存到接收 FIFO 中，使能接收 FIFO 满中断使能寄存器 RFF_INTEN，本模块的 FIFO 深度为 8，因此当 FIFO 中的数据到达 8 时，接收 FIFO 满状态信号 RFF 会被置 1 即上图中的 T0 时刻，并产生中断信号。需要注意的是，当 FIFO 中的数据不被读走，FIFO 始终满时，接收满状态信号会一直保持高，即使状态清零后它还会被再次置 1，只有 FIFO 中的数据被读走，FIFO 不满的状态下，满状态才会被清除，并且在 SPI 数据传输过程中，SPI 传输忙标志会一直保持高状态。

接收 FIFO 溢出中断



本模快在数据传输过程中，接收的数据会通过移位寄存器保存到接收 FIFO 中，使能接收 FIFO 溢出中断使能寄存器 RFOF_INTEN，本模块的 FIFO 深度为 8，因此当 FIFO 中的数据到达 8 后，继续向接收 FIFO 中传输数据，在 T0 时刻接收 FIFO 溢出状态信号 RFOF 会被置 1，并产生中断信号。需要注意的是，当 FIFO 中的数据不被读走，FIFO 始终满时，继续向 FIFO 传输数据，数据并不会被写入，会被丢掉，接收 FIFO 溢出状态信号也会一直保持高，即使状态清零后它还会被再次置 1，并且在 SPI 数据传输过程中，SPI 传输忙标志会一直保持高状态。

接收 FIFO 半满中断

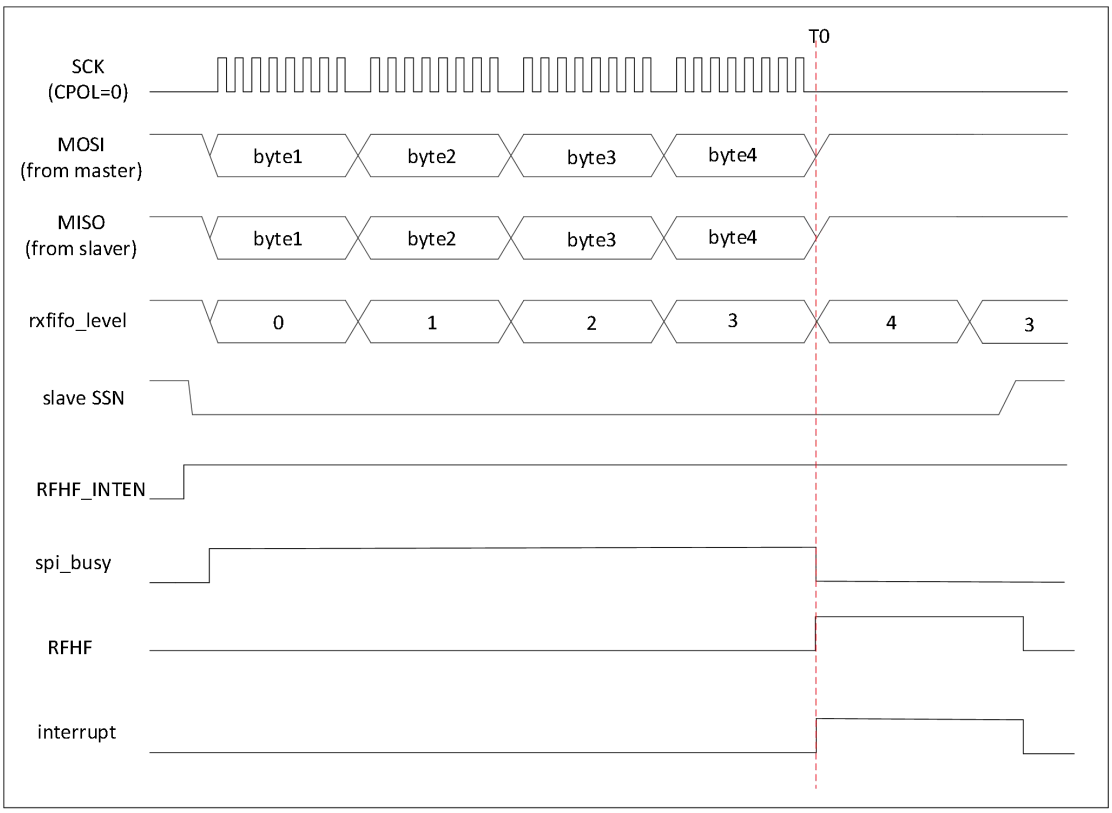


图 5-135　SPI 接收 FIFO 半满中断

本模块在数据传输过程中，接收的数据会通过移位寄存器保存到接收 FIFO 中，使能接收 FIFO 半满中断使能寄存器 RFHF_INTEN，本模块的 FIFO 深度为 8，因此当 FIFO 中的数据到达 4 时，接收 FIFO 半满状态信号 RFF 会被置 1 即上图中的 T0 时刻，并产生中断信号。需要注意的是，当 FIFO 中的数据数大于等于 4 时，接收半满状态信号会一直保持高，即使状态清零后它还会被再次置 1，只有 FIFO 中的数据被读走，FIFO 中数据小于半满的状态下，半满状态才会被清除，并且在 SPI 数据传输过程中，SPI 传输忙标志会一直保持高状态。

发送 FIFO 半满中断

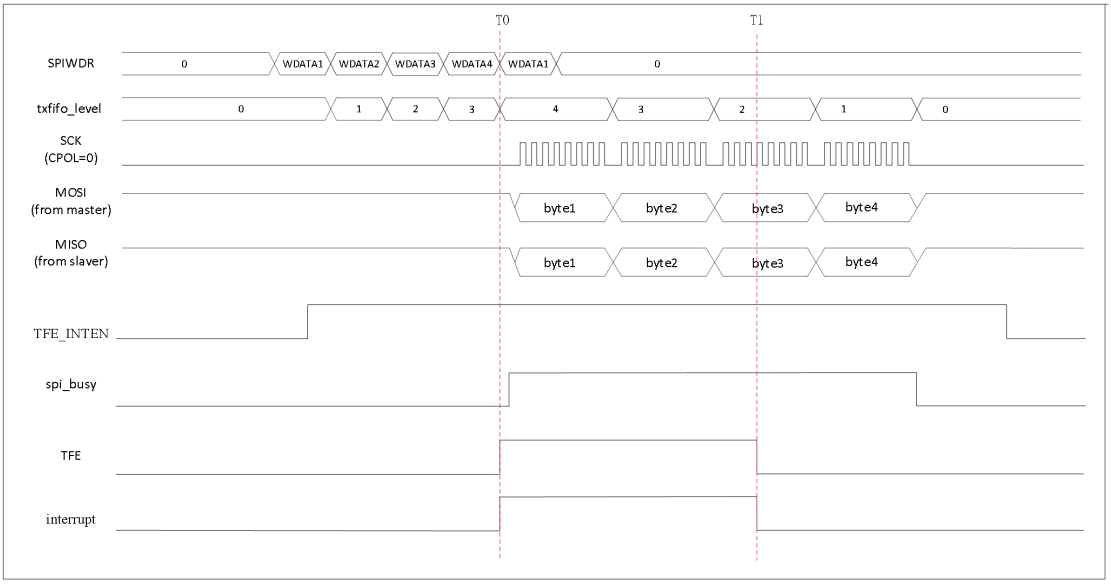


图 5-136 SPI 发送 FIFO 半满中断

本模块在数据传输过程中，发送的数据会通过 SPIWDR 寄存器保存到发送 FIFO 中，使能发送 FIFO 半满中断使能寄存器 TFE_INTEN，本模块的 FIFO 深度为 8，因此当发送 FIFO 中的数据量到达 4 时，发送 FIFO 半满状态信号 TFE 会被置 1 即上图中的 T0 时刻，并产生中断信号。需要注意的是，当发送 FIFO 中的数据不被读走，FIFO 始终大于等于半满时，发送满状态信号会一直保持高，即使状态清零后它还会被再次置 1，只有 FIFO 中的数据被读走，FIFO 在小于半满的状态下，满状态才会被清除，如图中 T1 时刻，并且在 SPI 数据传输过程中，SPI 传输忙标志会一直保持高状态。

发送 FIFO 空中断

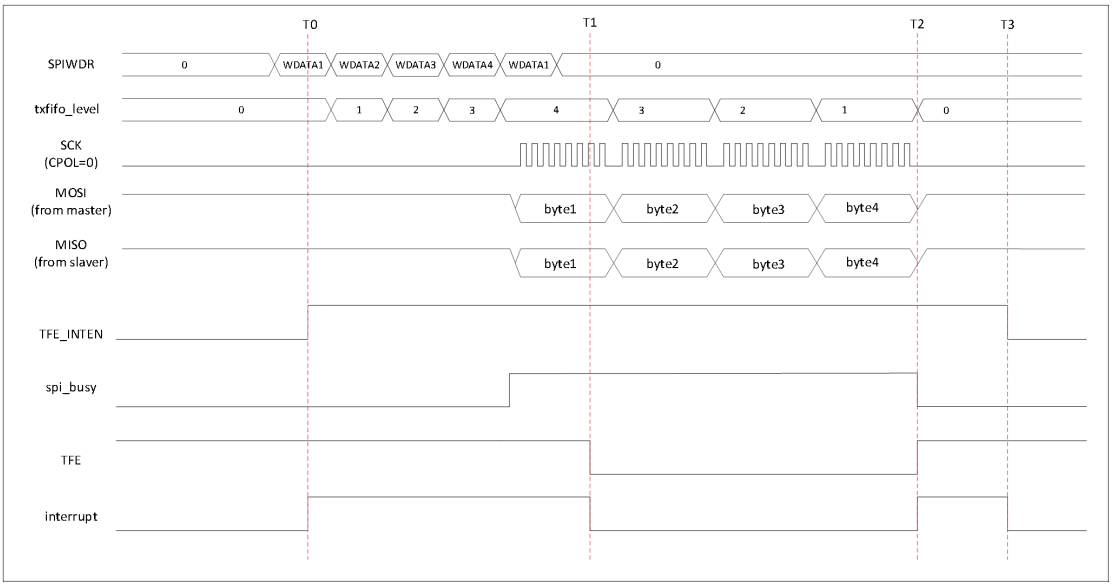


图 5-137　SPI 发送 FIFO 空中断

本模块在数据传输过程中，发送的数据会通过 SPIWDR 寄存器保存到发送 FIFO 中，传输开始时发送 FIFO 中没有数据，因此发送空状态信号 TFE 为 1，T0 时刻使能发送 FIFO 半满中断使能寄存器 TFE_INTEN，会产生中断信号；当向发送 FIFO 中写入数据后，并对空状态进行了清零，T1 时刻空状态和发送空中断信号会被置 0；T2 时刻，发送 FIFO 的数据再次被读空，发送空状态会再次被置 1，并产生中断信号；T3 时刻关闭发送空中断使能，将不会再产生中断信号。需要注意的是，当发送 FIFO 中一致没有数据，FIFO 始终为空时，发送满状态信号会一直保持高，即使状态清零后它还会被再次置 1，只有 FIFO 中有数据写入，FIFO 在非空状态下，空状态才会被清除，并且在 SPI 数据传输过程中，SPI 传输忙标志会一直保持高状态。

DMA 控制

DMA 可以通过 DMA 控制使能寄存器 TXDMAEN 和 RXDMAEN 实现对本模块的控制，TXDMAEN 为 1 时，表示 DMA 操作 SPI 的发送数据寄存器；RXDMAEN 为 1 时，表示 DMA 操作 SPI 的接收数据寄存器。

操作流程

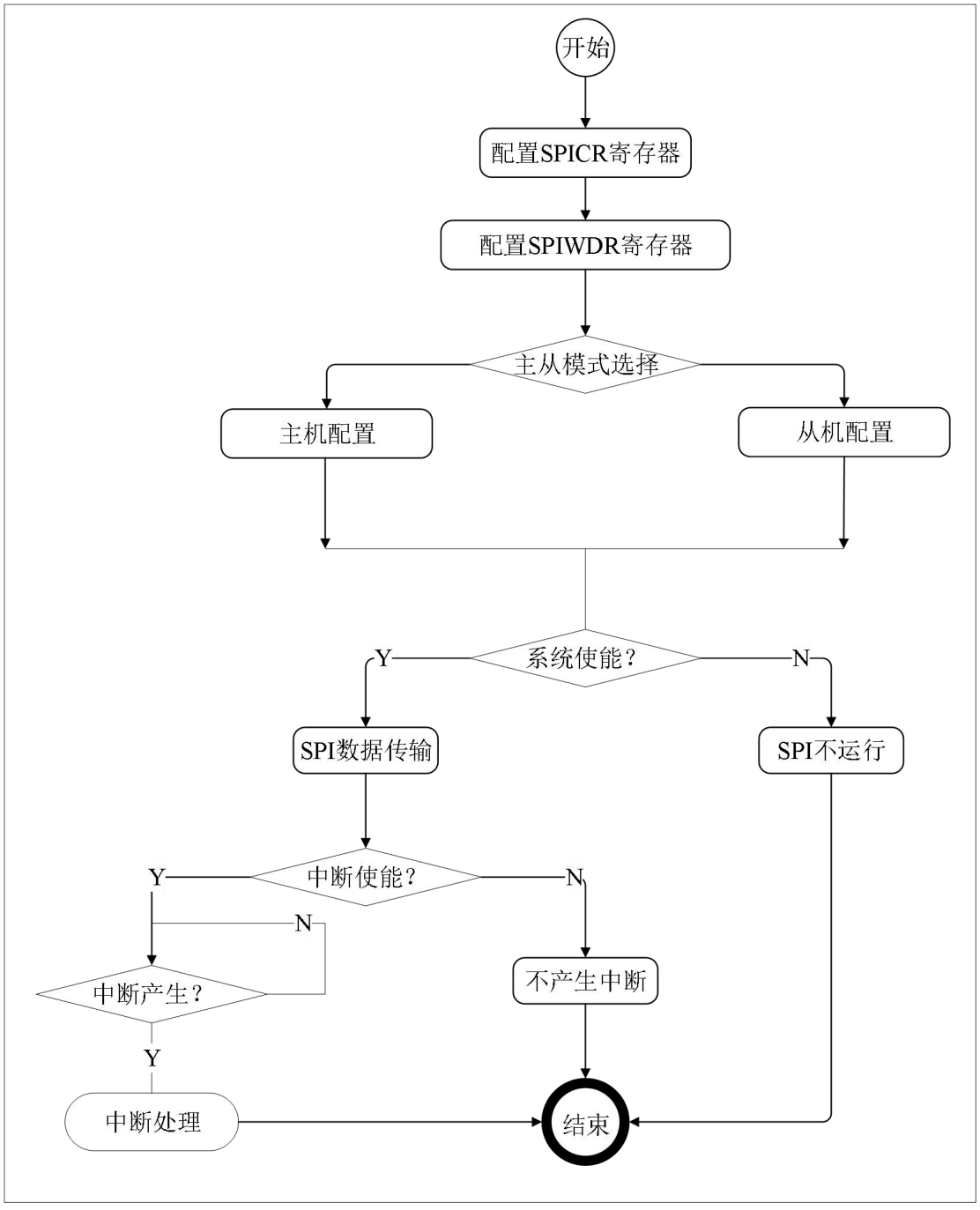


图 5-138　SPI 操作流程

- 配置 SPI 模块时钟
- PORT 端口配置为 SPI 功能

- 配置 SPI 主从模式
- 配置 SPI 时钟的相位和极性
- 配置传输顺序
- 如果为主模式还需配置 SPI 波特率
- 配置中断使能
- 配置 SPI 系统使能，开启 SPI，开始传输数据

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
SPI0: BASE: 0x400B8000 SPI1: BASE: 0x400B8800					
SPICR	0x00	32	R/W	0x1010	SPI 控制寄存器
SPIWDR	0x04	32	R/W	0x00	SPI 写数据寄存器
SPIRDR	0x08	32	R	0x00	SPI 读数据寄存器
SPIIE	0x10	32	RW	0x00	SPI 中断使能寄存器
SPIIF	0x14	32	R/W	0x8	SPI 中断状态寄存器
SPIFIFOST	0x18	32	R/W	0x9	SPIFIFO 状态寄存器

寄存器描述

SPICR 寄存器（0x00）

位域	名称	类型	复位值	描述
31:17	RESERVED	R	0	保留位
16	TF_CLR	R/W	0	发送 FIFO 清除位 软件写 1 清发送 FIFO，硬件自动清零
15	RF_CLR	R/W	0	接收 FIFO 清除位 软件写 1 清接收 FIFO，硬件自动清零
14	TXDMAEN	R/W	0	发送 DMA 控制使能位 1：表示 DMA 操作 SPI 的发送数据寄存器 0：表示 CPU 操作 SPI 的发送数据寄存器
13	RXDMAEN	R/W	0	接收 DMA 控制使能位 1：表示 DMA 操作 SPI 的接收数据寄存器 0：表示 CPU 操作 SPI 的接收数据寄存器
12	MSR_SSN	R/W	1	主模式下 SSN 输出，默认情况下输出 1 该寄存器仅在主模式下有效
11:8	CPHA_DATA HOLD_S	R/W	0	从模式下 CPHA 为 1 时，数据保持时间配置寄存器 0000：1 个 pclk 0001：2 个 pclk …… 1111：16 个 pclk
7	LSB	R/W	0	数据传输顺序选择 0：MSB 1：LSB
6	MSTR	R/W	0	主从模式选择 0 = SPI 系统配置为从器件模式 1 = SPI 系统配置为主器件模式
5	CPOL	R/W	0	时钟极性选择 0 = 串行时钟空闲状态为低电平，有效电平为高电平 1 = 串行时钟空闲状态为高电平，有效电平为低电平
4	CPHA	R/W	1	时钟相位选择 0 = 在串行时钟的第一个跳变沿采样数据 1 = 在串行时钟的第二个跳变沿采样数据

3	SPE	R/W	0	SPI 系统使能 0 = SPI 系统关闭 1 = SPI 系统使能
2	SPR2	R/W	0	SPI 波特率选择位 2
1	SPR1	R/W	0	SPI 波特率选择位 1
0	SPR0	R/W	0	SPI 波特率选择位 0

SPR0，SPR1，SPR2 表示波特率的选择：

SPR2	SPR1	SPR0	Fsck	Fsck(Fcpu=48Mhz)
0	0	0	Fpclk/4	12MHz
0	0	1	Fpclk/8	6MHz
0	1	0	Fpclk/16	3MHz
0	1	1	Fpclk/32	1.5MHz
1	0	0	Fpclk/64	750KHz
1	0	1	Fpclk/128	375KHz
1	1	0	Fpclk/256	187.5KHz
1	1	1	Fpclk/512	93.75KHz

SPIWDR 寄存器（0x04）

位域	名称	类型	复位值	描述
7:0	SPIWDR	R/W	0	SPI 将要发送的数据通过 SPIWDR 寄存器写入发送 FIFO 中。准备发送数据时，移位寄存器直接读取发送 FIFO 进行数据的发送。

SPIRDR 寄存器（0x08）

位域	名称	类型	复位值	描述
7:0	SPIRDR	R	0	SPI 通过 SPIRDR 寄存器读出接收到的数据。移位寄存器在每次传输完成后，将数据存入接收 FIFO 中，通过 SPIRDR 寄存器读出接收 FIFO 中的数据。

SPIIE 寄存器（0x10）

位域	名称	类型	复位值	描述
31:5	RESERVED	R	0	保留位
4	TXFIFO_HFULL	R/W	0	发送 FIFO 半满中断使能
3	TXFIFO_EMPTY	R/W	0	发送 FIFO 空中断使能
2	RXFIFO_HFULL	R/W	0	接收 FIFO 半满中断使能
1	RXFIFO_FULL	R/W	0	接收 FIFO 满中断使能
0	RXFIFO_OVF	R/W	0	接收 FIFO 溢出中断使能

SPIIF 寄存器（0x14）

位域	名称	类型	复位值	描述
31:5	RESERVED	R	0	保留位
4	TXFIFO_HFULL	R/W	0	发送 FIFO 半满标志 写 1 清零
3	TXFIFO_EMPTY	R/W	1	发送 FIFO 空标志 写 1 清零
2	RXFIFO_HFULL	R/W	0	接收 FIFO 半满标志 写 1 清零

1	RXFIFO_FULL	R/W	0	接收 FIFO 满标志 写 1 清零
0	RXFIFO_OVF	R/W	0	接收 FIFO 溢出标志 写 1 清零 注：溢出后的数据将被丢弃。

SPIFIFOST 寄存器（0x18）

位域	名称	类型	复位值	描述
31:12	Reserved	R	0	保留位
11:9	TF_LEVEL	R	0	发送 FIFO 水位状态 000：未满时表示 FIFO 有 0 个数据，满时表示 FIFO 有 8 个数据； 001：表示 FIFO 有 1 个数据； 010：表示 FIFO 有 2 个数据； 011：表示 FIFO 有 3 个数据； 100：表示 FIFO 有 4 个数据； 101：表示 FIFO 有 5 个数据； 110：表示 FIFO 有 6 个数据； 111：表示 FIFO 有 7 个数据；
8:6	RF_LEVEL	R	0	接收 FIFO 水位状态 000：未满时表示 FIFO 有 0 个数据，满时表示 FIFO 有 8 个数据； 001：表示 FIFO 有 1 个数据； 010：表示 FIFO 有 2 个数据； 011：表示 FIFO 有 3 个数据； 100：表示 FIFO 有 4 个数据； 101：表示 FIFO 有 5 个数据； 110：表示 FIFO 有 6 个数据； 111：表示 FIFO 有 7 个数据；
5	TFHF	R	0	发送 FIFO 半满标志
4	TFF	R	0	发送 FIFO 满标志

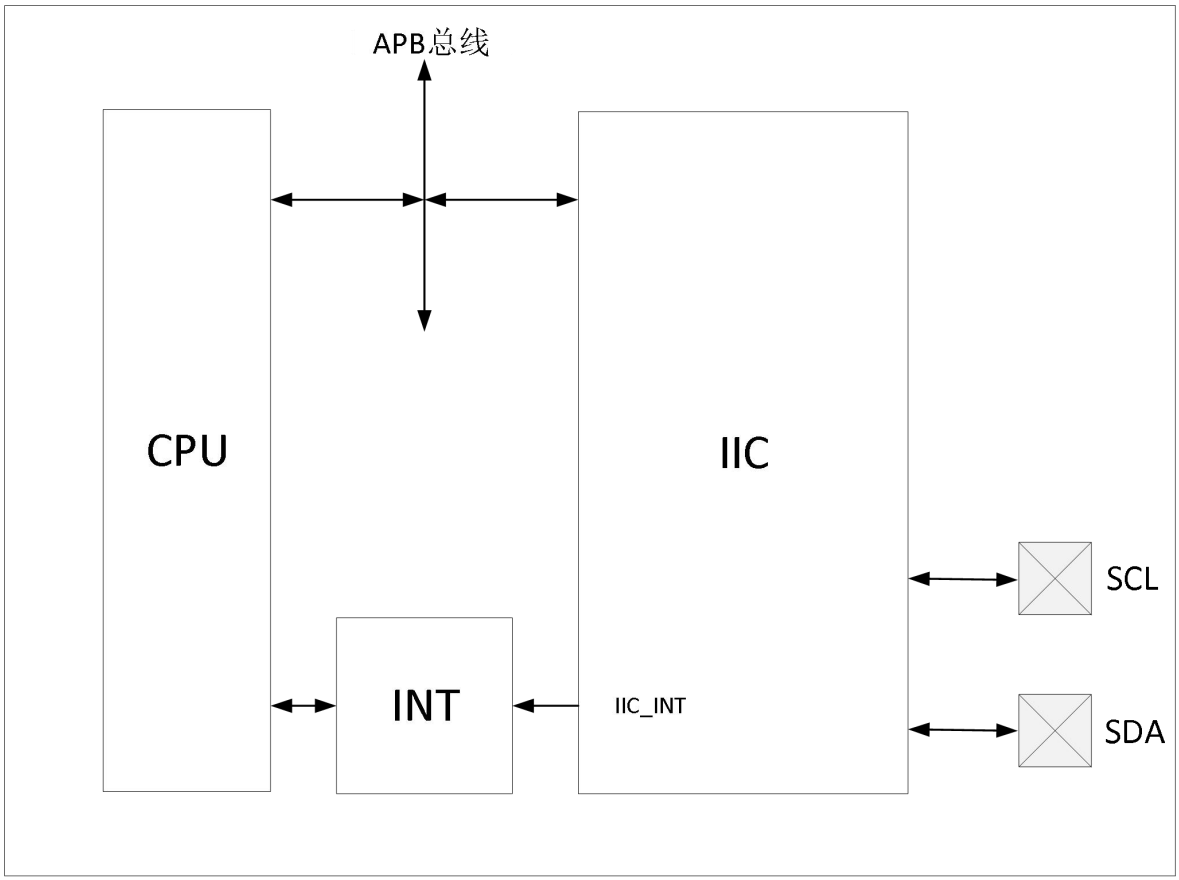
3	TFE	R	1	发送 FIFO 空标志
2	RFHF	R	0	接收 FIFO 半满标志
1	RFF	R	0	接收 FIFO 满标志
0	RFE	R	1	接收 FIFO 空标志

5.18 IIC 控制器（IIC）

5.18.1 概述

IIC(Inter-Integrated Circuit) 是一种串行通信总线，使用多主从架构，IIC 总线在物理连接上非常简单，分别由 SDA(串行数据线)和 SCL(串行时钟线)及上拉电阻组成。通信原理是通过对 SCL 和 SDA 线高低电平时序的控制，来产生 IIC 总线协议所需要的信号进行数据的传递。在总线空闲状态时，这两根线一般被上面所接的上拉电阻拉高，保持着高电平。IIC 通信方式为半双工，只有一根 SDA 线，同一时间只可以单向通信。IIC 总线数据传输速率在标准模式下可达 100kbit/s，快速模式下可达 400kbit/s。一般通过 IIC 总线接口可编程时钟来实现传输速率的调整，同时也跟所接的上拉电阻的阻值有关。芯片提供了一个 IIC 接口模块，实现与外部的 IIC 设备通信。

下图为 IIC 模块的系统示意图，本模块是通过 APB 总线来实现与 CPU 连接的，并可以产生中断。



IIC 模块系统示意图

5.18.2特性

- 支持 master、slave 两种模式
- 支持 IIC 输入信号数字滤波
- 支持 3 种模式：Standard-mode（100kbps）、Fast-mode（400kbps）、Fast-mode Plus（1Mbps）
- SCL/SDA 线上数据可读
- 支持多种中断

Master 模式特性：

- 支持 SCL LOW 超时报警
- 支持发出的 SCL 时钟周期最大为（2^17）*系统时钟
- SCL 时钟占空比可配置

Slave 模式特性：

- 支持 7 位、10 位两种地址模式
- 支持地址 mask，一个 slave 器件可以占用多个地址，7 位地址模式，一个 slave 器件最多可占用 128 个地址；10 位地址模式，一个 slave 器件最多可占用 256 个地址
- 支持 clock stretching，slave 器件可通过拉低 SCL 来 hold 总线

5.18.3模块结构框图

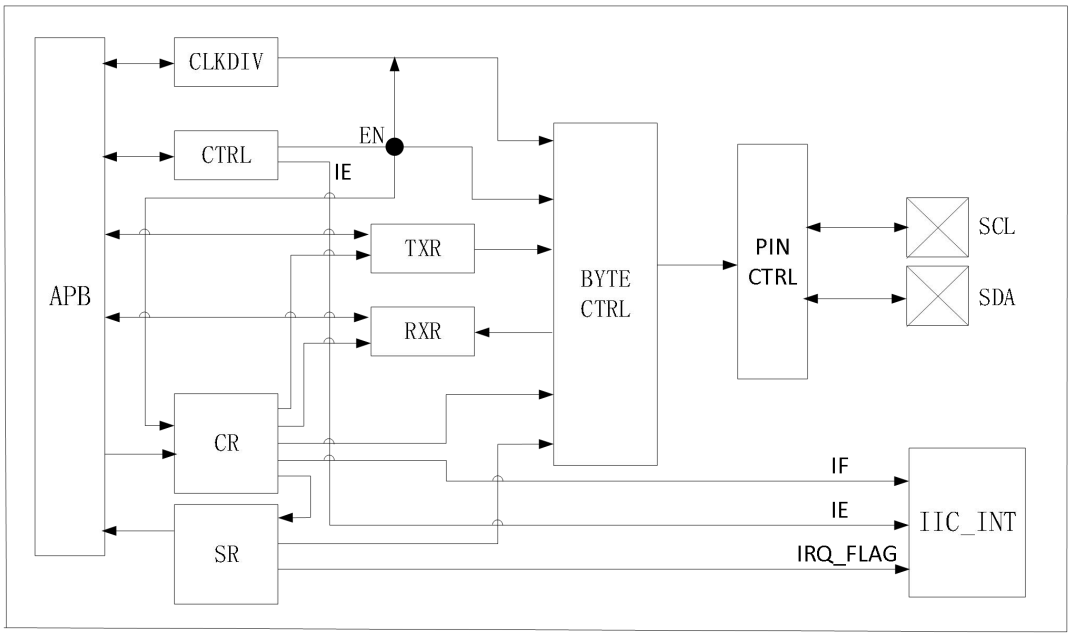


图 5-139 IIC 模块结构框图

IIC 模块的结构如上图所示。REG 模块实现寄存器读写并产生中断；IO 模块对 SDA、SCL 输入进行同步处理和数字滤波，对 SDA、SCL 输出进行选择；MASTER 模块实现主机功能；SLAVE 模块实现从机功能；TIMER 实现定时功能，包括 START SCL 保持时间、SCL 低电平长度计时、SCL 高电平长度计时，发送数据时 SDA 数据保持时间计时、SCL 低电平长度超时计时等。

5.18.4功能描述

协议介绍

IIC 总线采用串行数据线(SDA)和串行时钟线(SCL)传输数据。

数据在主从设备之间通过 SCL 时钟信号在 SDA 数据线上逐字节同步传输。每一个 SCL 时钟脉冲发送一位数据，高位在前。每发送一个字节的数据产生一个应答信号。在时钟线 SCL 高电平期间对数据的每一位进行采样。数据线 SDA 在时钟线 SCL 为低电平时改变，在时钟线 SCL 为高电平时保持稳定。

通常情况下，一个标准的通信包含四个部分：开始信号、从机地址、数据传输、停止信号。如下图所示：

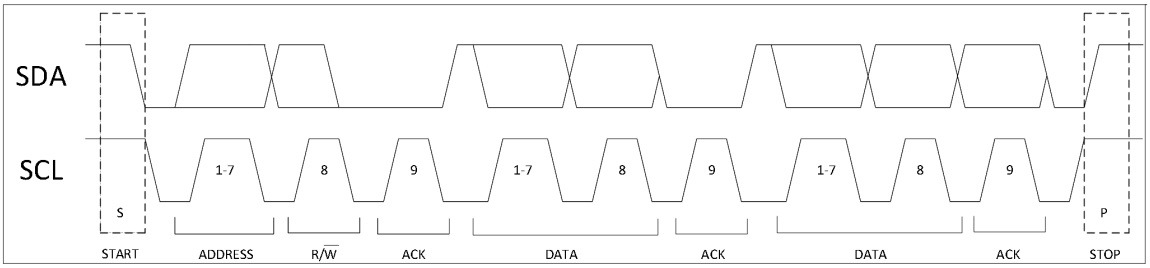


图 5-140　IIC 数据传输图

开始信号发送

当总线空闲时，表示没有主机设备占用总线（SCL 和 SDA 都保持高电平），主机可以通过发送一个开始信号启动传输。启动信号，通常被称为 S 位。SCL 为高电平时，SDA 由高电平向低电平跳变。启动信号表示开始新的数据传输。

从机地址发送

在开始信号后，由主机传输的第一个字节数据是从机地址。包含 7 位的从设备地址和 1 位的 R/W 指示位。R/W 指示位信号表示与从机的数据传输方向，0 表示写操作，1 表示读操作。在系统中的从机不可以具有相同的地址。 只有从机地址和主机发送的地址匹配时才能产生一个应答位（在第九个时钟周期拉低 SDA）进行响应。

数据传输

一旦成功取得了从机地址，主机就可以通过 R/W 位控制逐字节的发送数据。每传输一个字节都需要在第九个时钟周期产生一个应答位。

如果从机信号无效或者从机返回一个 NACK 信号，主机可以生成一个停止信号中止数据传输。

如果主机作为接收设备，没有应答从机，从机就会释放 SDA，主机产生停止信号。

停止信号发送

主机可以通过生成一个停止信号终止通信。停止信号通常被称为 P 位，被定义为 SCL 为高电平时，SDA 由低电平向高电平跳变。

7bit 地址数据传输

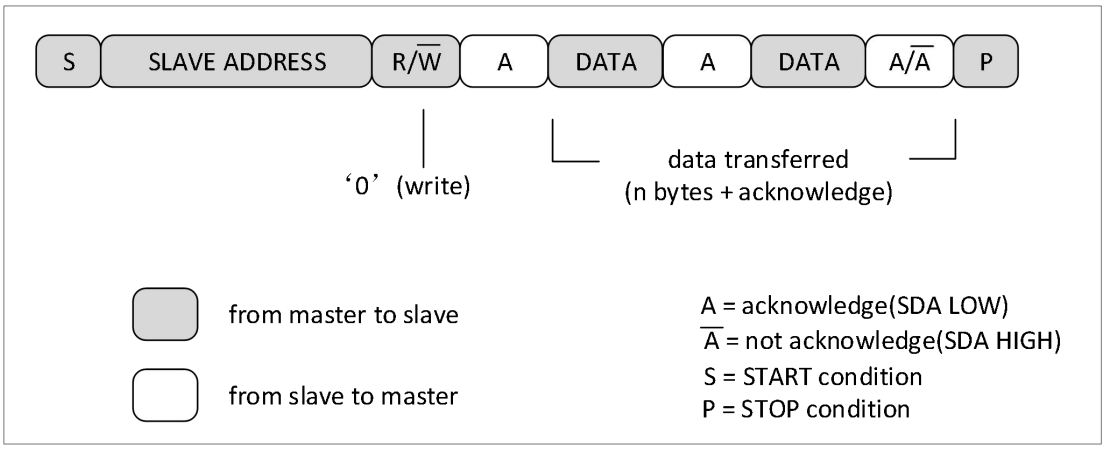


图 5-141　IIC 主机发送用 7bit 地址寻址从机接收

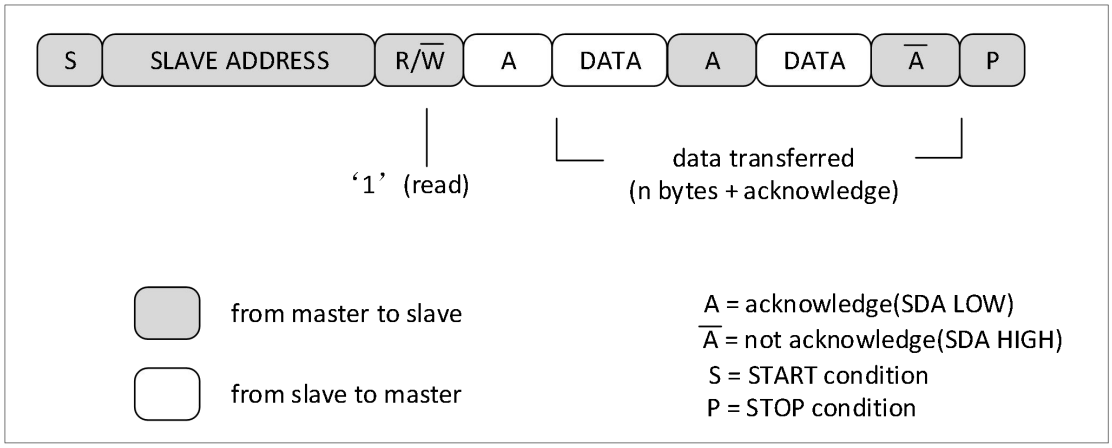


图 5-142　IIC 主机在第一个字节后立即读取从机

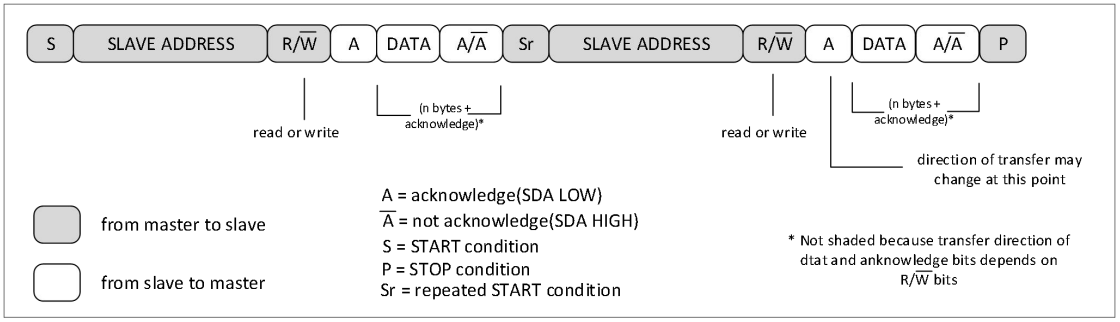


图 5-143 IIC 组合格式

10bit 地址数据传输

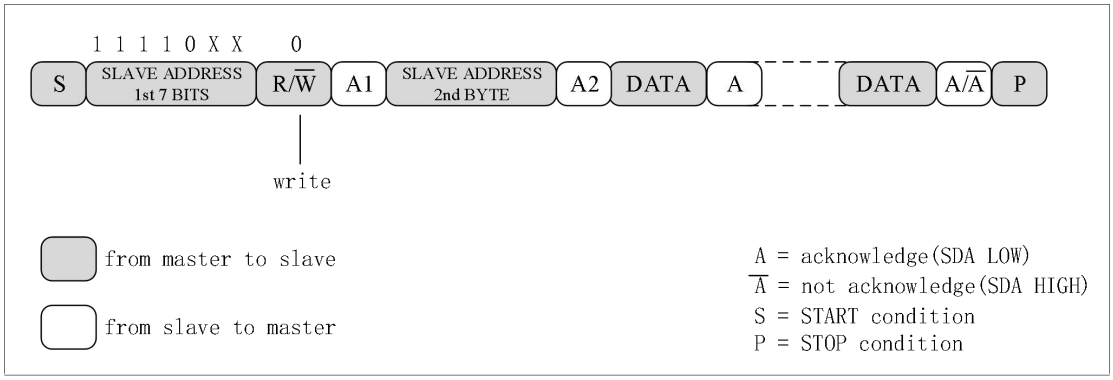


图 5-144 IIC 主机发送用 10bit 地址寻址从机接收

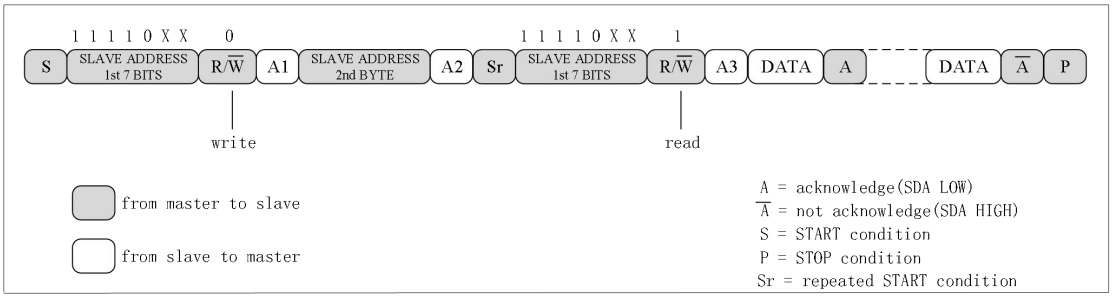


图 5-145 IIC 主机接收用 10bit 地址寻址从机发送

SCL 和 SDA

在 IIC 协议中，端口信号 SCL 和 SDA 为双向信号。对于 Standard-mode、Fast-mode，SCL 和 SDA 为双向开漏 IO，信号由电阻上拉到高电平，上升沿缓慢。

SCL 和 SDA 使用的 IO 为普通的双向 IO，通过对使能信号的处理来实现双向开漏 IO 的功能。通过如下的方式，实现 SCL 和 SDA 的双向开漏 IO：

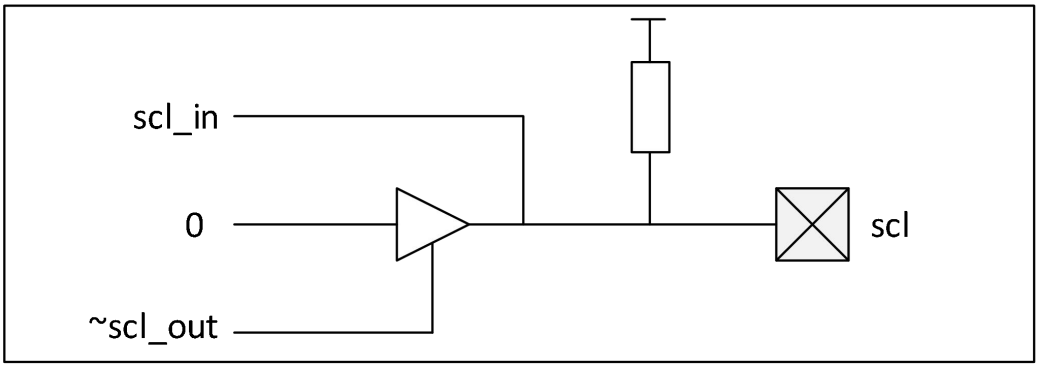


图 5-146 IIC SCL 开漏图

中断功能

本模块提供了六种中断功能，包括接收数据溢出、发送完成、接收完成、SLAVE 检测到 START、SLAVE 检测到 STOP 和 MASTER SCL LOW 超时中断。可通过 IIC_IE 寄存器配置相应中断使能位，通过 IIC_IF 寄存器查看各相应中断状态位。

中断标志及中断示意图如下：

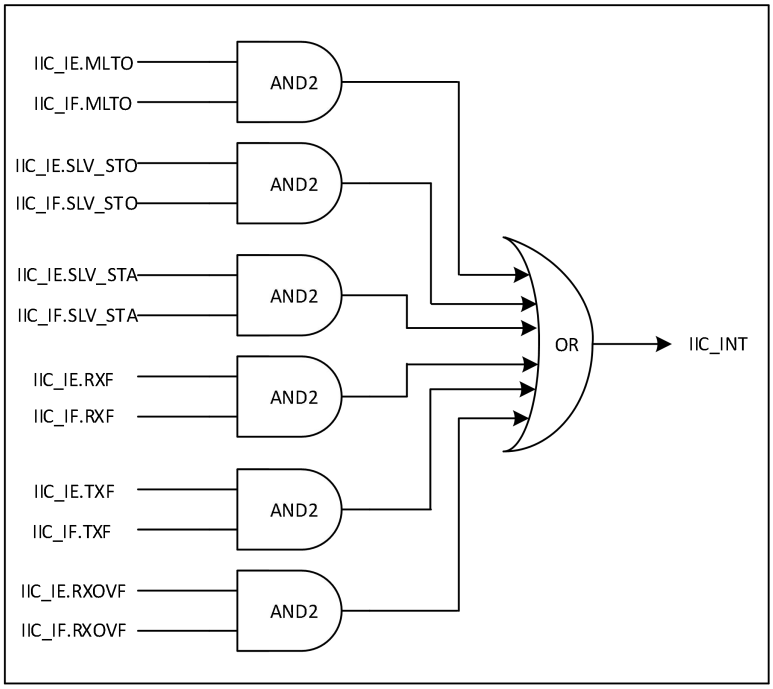


图 5-147　IIC 中断标志及中断示意图

操作流程

master-transmitter

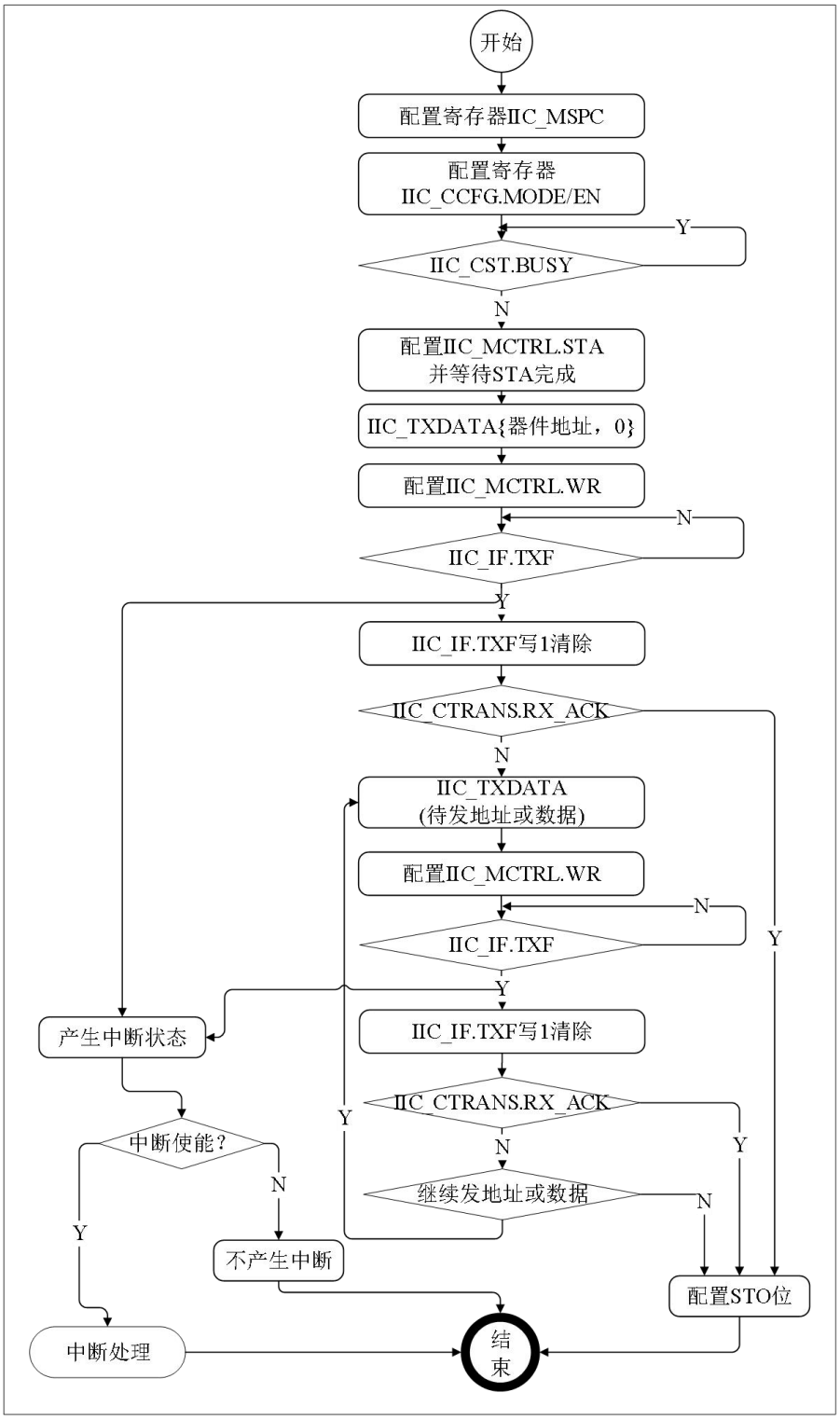


图 5-148 IIC 主机发送操作流程图

1、设置寄存器 IIC_MSPC。假设 pclk=60M，希望 IIC 工作在 Standard-mode（100kbps）速度下，则每个 SCL 600 个 pclk，那么 tLOW 为 400 个 pclk，tHIGH 为 200 个 pclk，这样可以设置 SCL_LOW=0xC4，SCL_HITH=0x60，CPD=0x01。

2、设置寄存器 IIC_CCFG 的 MODE 位为 1，寄存器 IIC_CCFG 的 EN 位为 1。

3、查询寄存器 IIC_CST 的 BUSY 位，如果为 1，则等待直至其变为 0；如果为 0，则进行下一步。

4、发 Start。设置寄存器 IIC_MCTRL 的 STA 位为 1，查询该位，直至其变为 0。

5、发 slave 地址字节，具体步骤如下：

设置寄存器 IIC_TXDATA，其中 bit7~bit1 为从机地址，bit0 为 0 即写命令；

设置寄存器 IIC_MCTRL 的 WR 位为 1，查询该位，直至其变为 0（或查询到寄存器 IIC_IF 的 TXF 位为 1（发送成功），并写 1 清除）；

读寄存器 IIC_CTRANS 的 RX_ACK 位，如果该位为 0，表示 slave 地址匹配成功可以进行下一步，如果该位为 1 转到步骤 8。

6、向 slave 发待写地址或数据，具体步骤如下：

设置寄存器 IIC_TXDATA，准备待写入 slave 的地址或数据；

设置寄存器 IIC_MCTRL 的 WR 位为 1，查询该位，直至其变为 0（或查询到寄存器 IIC_IF 的 TXF 位为 1，并写 1 清除）；

读寄存器 IIC_CTRANS 的 RX_ACK 位，如果该位为 0，表示写地址或数据成功可以进行下一步，如果该位为 1 转到步骤 8。

7、如果想继续发送地址或数据，重复步骤 6，否则进行下一步。

8、发 STOP。设置寄存器 IIC_MCTRL 的 STO 位为 1，查询该位，直至其变为 0。

9、若设置中断使能，则可在 TXF 发送完成时查到中断状态，可进行中断处理。

master-receiver

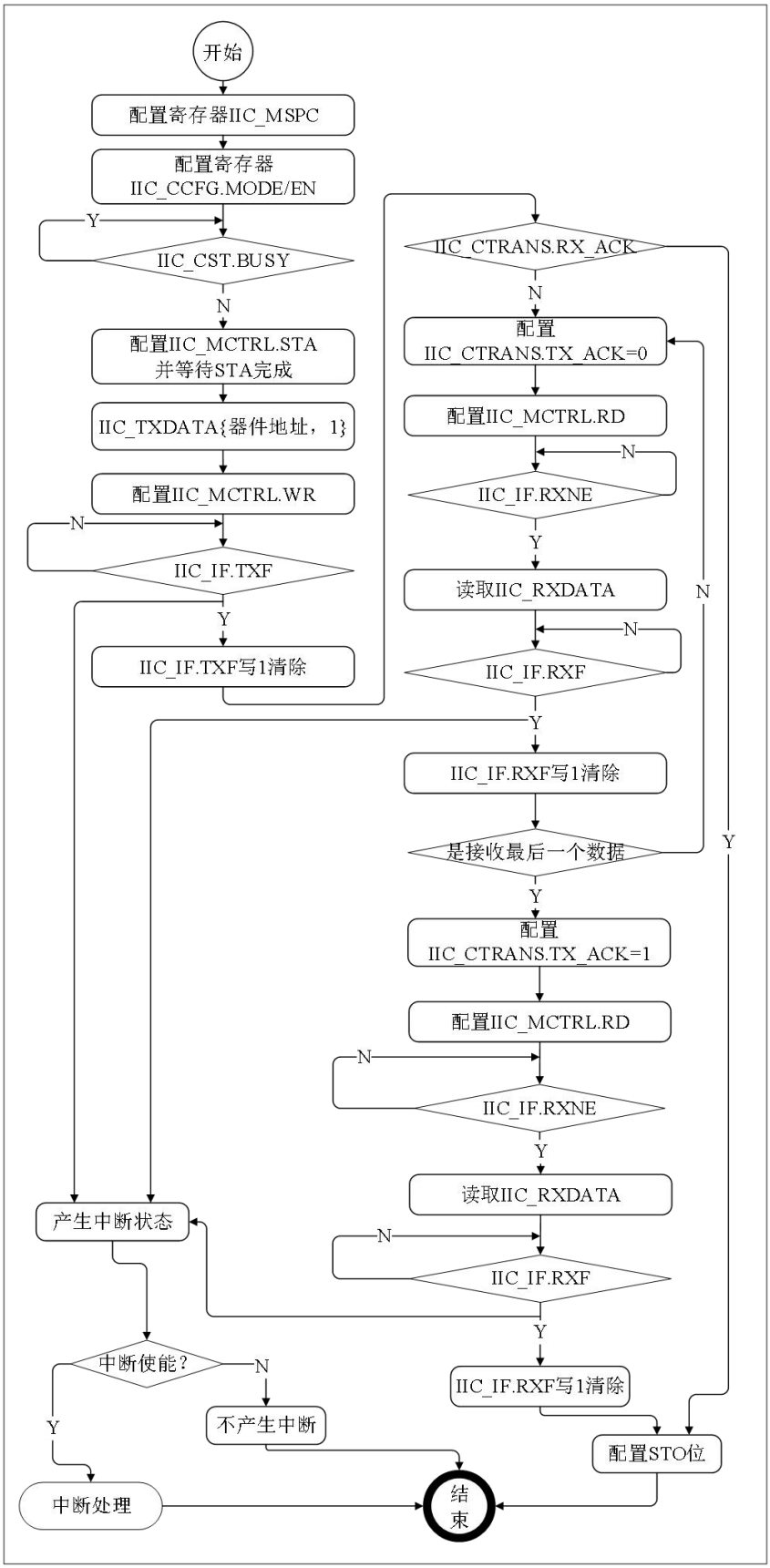


图 5-149 IIC 主机接收操作流程图

1、设置寄存器 IIC_MSPC。假设 pclk=60M，希望 IIC 工作在 Standard-mode（100kbps）速度下，则每个 SCL 600 个 pclk，那么 tLOW 为 400 个 pclk，tHIGH 为 200 个 pclk，这样可以设置 SCL_LOW=0xC4，SCL_HITH=0x60，CPD=0x01。

2、设置寄存器 IIC_CCFG 的 MODE 位为 1，寄存器 IIC_CCFG 的 EN 位为 1。

3、查询寄存器 IIC_CST 的 BUSY 位，如果为 1，则等待直至其变为 0；如果为 0，则进行下一步。

4、发 Start。设置寄存器 IIC_MCTRL 的 STA 位为 1，查询该位，直至其变为 0。

5、发 slave 地址字节，具体步骤如下：

设置寄存器 IIC_TXDATA，其中 bit7~bit1 为从机地址，bit0 为 1 即读命令；

设置寄存器 IIC_MCTRL 的 WR 位为 1，查询该位，直至其变为 0（或查询到寄存器 IIC_IF 的 TXF 位为 1（发送成功），并写 1 清除）；

读寄存器 IIC_CTRANS 的 RX_ACK 位，如果该位为 0，表示 slave 地址匹配成功可以进行下一步，如果该位为 1 转到步骤 8。

6、从 slave 读数据，具体步骤如下：

设置寄存器 IIC_CTRANS 的 TX_ACK 位为 0；

设置寄存器 IIC_MCTRL 的 RD 位为 1，查询直到寄存器 IIC_IF 的 RXNE 位为 1；

读取寄存器 IIC_RXDATA，得到 slave 数据；

查询寄存器 IIC_MCTRL 的 RD 位，直至其变为 0（或查询到寄存器 IIC_IF.RXF 位为 1，并写 1 清除）。

7、继续接收并且不是接收的最后一个数据，重复步骤 6；如果是接收的最后一个数据，设置寄存器 IIC_CTRANS 的 TX_ACK 位为 1，其他同步骤 6，完成后进行下一步。

8、发 STOP。设置寄存器 IIC_MCTRL 的 STO 位为 1，查询该位，直至其变为 0。

9、若设置中断使能，则可在 TXF 发送完成时和 RXF 接收完成时查到中断状态，可进行中断处理。

slave-transmitter

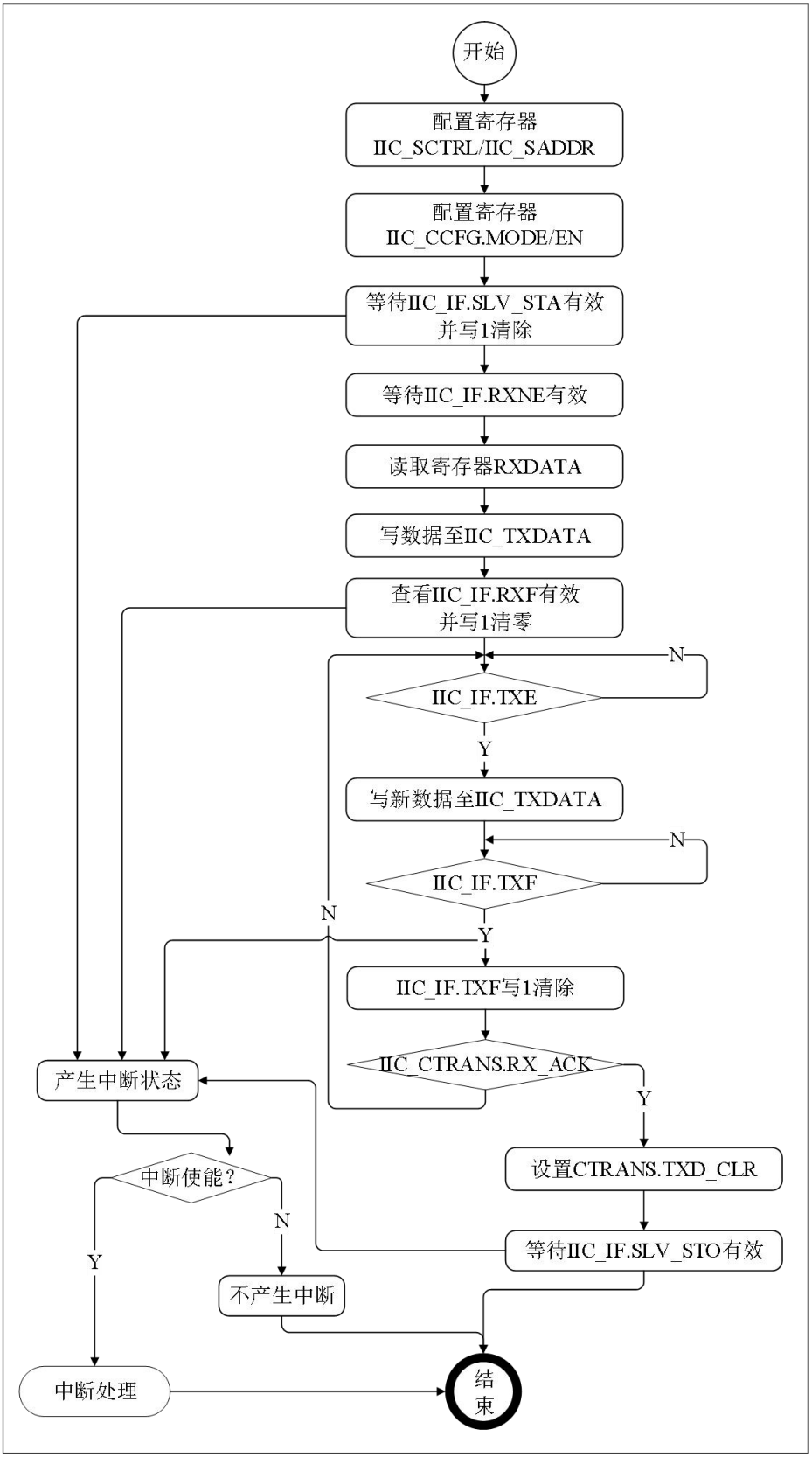


图 5-150　IIC 从机发送操作流程图

1、设置 slave 地址模式。寄存器 IIC_SCTRL 的 ADMD 位为 0。

2、设置 slave 地址寄存器 IIC_SADDR。

3、设置寄存器 IIC_CCFG 的 MODE 位为 0，寄存器 IIC_CCFG 的 EN 位为 1。

4、查询直至寄存器 IIC_IF 的 SLV_STA 位，表示检测到 IIC 总线上有 start 发出，并写 1 清零。

5、查询直至寄存器 IIC_IF 的 RXNE 位为 1。表示有 master 选中本器件。

6、读取寄存器 IIC_RXDATA，如果寄存器 IIC_SADDR 中设置了地址 mask，判断 master 发送的实际地址。

7、准备数据，写寄存器 IIC_TXDATA。

8、查询直到寄存器 IIC_IF 的 RXF 位为 1，表示之前地址匹配后，返回 ACK 结束。

9、查询直到寄存器 IIC_IF 的 TXE 位为 1，就可以向 TXDATA 中写入新数据了。

10、查询直到寄存器 IIC_IF 的 TXF 位为 1，表示数据发送完成。然后写 1 清除。

11、查询寄存器 IIC_CTRANS 的 RX_ACK 位，如果为 0，表示 master 希望继续接收数据，重复 9~11；如果寄存器 IIC_CTRANS 的 RX_ACK 位为 1，表示 master 希望结束读操作，则设置寄存器 IIC_CTRANS 的 TXD_CLR 位，清除之前预准备到寄存器 IIC_TXDATA 中的最后一个数据。转入下一步。

12、查询到寄存器 IIC_IF 的 SLV_STO 位，表示检测到 IIC 总线上有 STOP 发出。本次会话结束。

13、若设置中断使能，则可在 SLAVE 检测到 STA、RXF 接收完成、TXF 发送完成和 SLAVE 检测到 STO 时查到中断状态，可进行中断处理。

slave-receiver

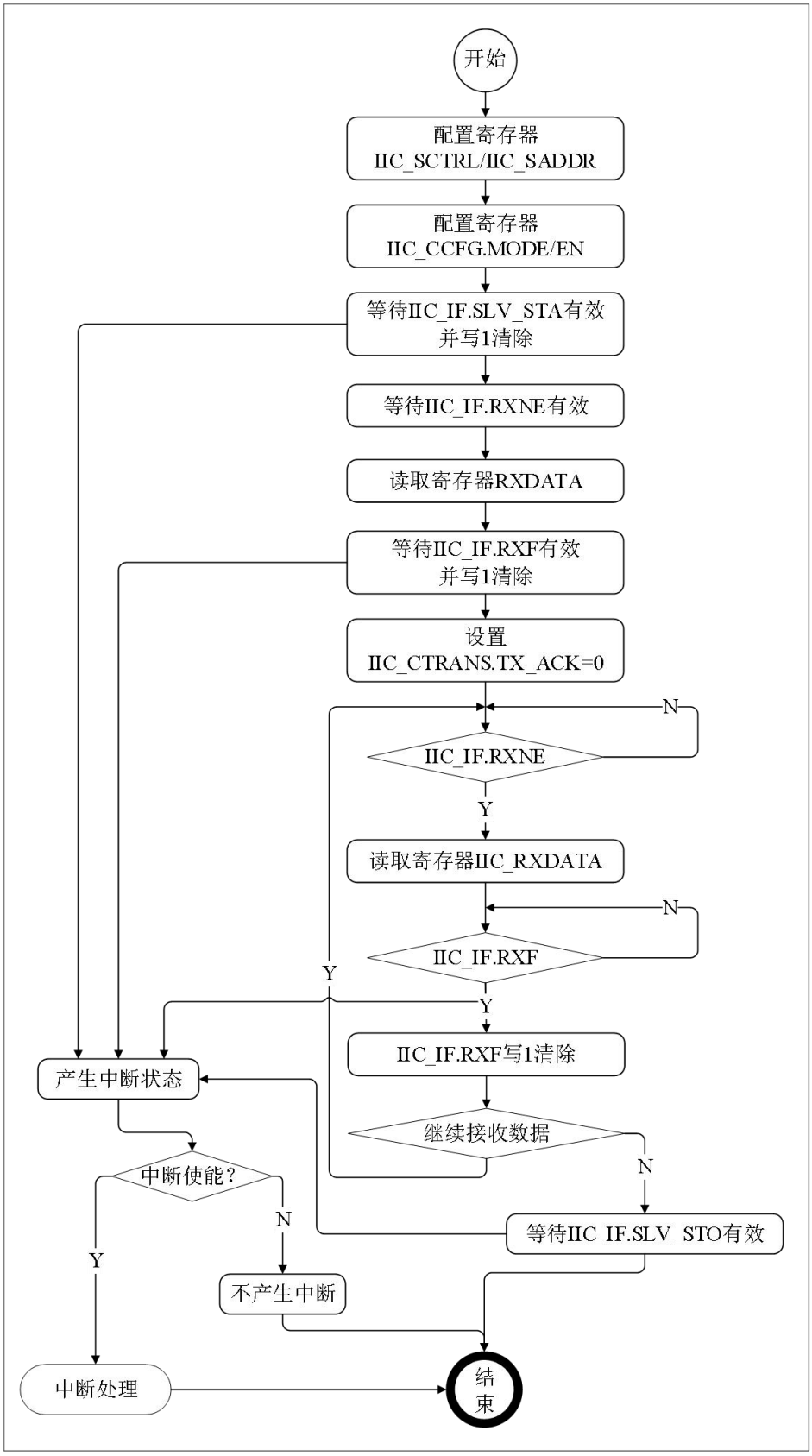


图 5-151 IIC 从机接收操作流程图

1、设置 slave 地址模式。寄存器 IIC_SCTRL 的 ADMD 位为 0。

2、设置 slave 地址寄存器 IIC_SADDR。

3、设置寄存器 IIC_CCFG 的 MODE 位为 0，寄存器 IIC_CCFG 的 EN 位为 1。

4、查询直至寄存器 IIC_IF 的 SLV_STA 位，表示检测到 IIC 总线上有 start 发出，并写 1 清零。

5、等待直至寄存器 IIC_IF 的 RXNE 位为 1。表示有 master 选中本器件。

6、读取寄存器 IIC_RXDATA，如果寄存器 IIC_SADDR 中设置了地址 mask，判断 master 发送的实际地址。

7、查询直到寄存器 IIC_IF 的 RXF 位为 1，表示之前地址匹配后，返回 ACK 结束。然后写 1 清除。

8、设置寄存器 IIC_CTRANS 的 TX_ACK 位为 0。

9、查询直到寄存器 IIC_IF 的 RXNE 位为 1，表示 slave 接收到新数据，读取寄存器 IIC_RXDATA。

10、查询直到寄存器 IIC_IF 的 RXF 位为 1，表示之前接收数据后，返回 ACK 结束。然后写 1 清除。

11、重复 9~10，继续接收数据，直到查询到寄存器 IIC_IF 的 SLV_STO 位，表示本次会话结束。

12、若设置中断使能，则可在 SLAVE 检测到 STA、RXF 接收完成和 SLAVE 检测到 STO 时查到中断状态，可进行中断处理。

clock-stretching

以 master-receiver 为例，具体如下：

1、设置寄存器 IIC_MSPC。假设 pclk=60M，希望 IIC 工作在 Standard-mode（100kbps）速度下，则每个 SCL 600 个 pclk，那么 tLOW 为 400 个 pclk，tHIGH 为 200 个 pclk，这样可以设置 SCL_LOW=0xC4，SCL_HITH=0x60，CPD=0x01。

2、设置寄存器 IIC_CCFG 的 MODE 位为 1，寄存器 IIC_CCFG 的 EN 位为 1。

3、查询寄存器 IIC_CST 的 BUSY 位，如果为 1，则等待直至其变为 0；如果为 0，则进行下一步。

4、发 Start。设置寄存器 IIC_MCTRL 的 STA 位为 1，查询该位，直至其变为 0。

5、发 slave 地址字节，具体步骤如下：

设置寄存器 IIC_TXDATA，其中 bit7~bit1 为从机地址，bit0 为 1 即读命令；

设置寄存器 IIC_MCTRL 的 WR 位为 1，查询该位，直至其变为 0（或查询到寄存器 IIC_IF 的 TXF 位为 1（发送成功），并写 1 清除）；

读寄存器 IIC_CTRANS 的 RX_ACK 位，如果该位为 0，表示 slave 地址匹配成功可以进行下一步，如果该位为 1 转到步骤 8。

6、从 slave 读数据：如果 slave 准备数据的时间很长，master 的 SCL 会被 slave hold 直至 slave 数据准备好，如果 hold 的时间过长，会导致 master 产生寄存器 IIC_IF 的 MLTO 位的中断，具体步骤如下：

设置寄存器 IIC_CTRANS 的 TX_ACK 位为 0；

设置寄存器 IIC_MCTRL 的 RD 位为 1，查询直到寄存器 IIC_IF 的 RXNE 位为 1；

读取寄存器 IIC_RXDATA，得到 slave 数据；

查询寄存器 IIC_MCTRL 的 RD 位，直至其变为 0（或查询到寄存器 IIC_IF.RXF 位为 1，并写 1 清除）。

7、继续接收并且不是接收的最后一个数据，重复步骤 6；如果是接收的最后一个数据，设置寄存器 IIC_CTRANS 的 TX_ACK 位为 1，其他同步骤 6，完成后进行下一步。

8、发 STOP。设置寄存器 IIC_MCTRL 的 STO 位为 1，查询该位，直至其变为 0。

9、若设置中断使能，则可在 TXF 发送完成时和 RXF 接收完成时查到中断状态，可进行中断处理。

以 slave-transmitter 为例，具体如下：

1、设置 slave 地址模式。寄存器 IIC_SCTRL 的 ADMD 位为 0，STRETCH 位为 1。

2、设置 slave 地址寄存器 IIC_SADDR。

3、设置寄存器 IIC_CCFG 的 MODE 位为 0，寄存器 IIC_CCFG 的 EN 位为 1。

4、查询直至寄存器 IIC_IF 的 SLV_STA 位，表示检测到 IIC 总线上有 start 发出，并写 1 清零。

5、查询直至寄存器 IIC_IF 的 RXNE 位为 1。表示有 master 选中本器件。

6、读取寄存器 IIC_RXDATA，如果寄存器 IIC_SADDR 中设置了地址 mask，判断 master 发送的实际地址。

7、准备数据，写寄存器 IIC_TXDATA，如果准备数据的时间很长，则 slave 硬件会自动拉低 SCL，阻止 master 发出新的 SCL。当 slave 完成写寄存器 IIC_TXDATA 动作后，slave 硬件会再等待寄存器 IIC_MSPC 设置的 SCL_LOW 时间后，释放 SCL，进行下一步。

8、查询直到寄存器 IIC_IF 的 RXF 位为 1，表示之前地址匹配后，返回 ACK 结束。

9、查询直到寄存器 IIC_IF 的 TXE 位为 1，就可以向 TXDATA 中写入新数据了。

10、查询直到寄存器 IIC_IF 的 TXF 位为 1，表示数据发送完成。然后写 1 清除。

11、查询寄存器 IIC_CTRANS 的 RX_ACK 位，如果为 0，表示 master 希望继续接收数据，重复 9~11；如果寄存器 IIC_CTRANS 的 RX_ACK 位为 1，表示 master 希望结束读操作，则设置寄存器 IIC_CTRANS 的 TXD_CLR 位，清除之前预准备到寄存器 IIC_TXDATA 中的最后一个数据。转入下一步。

12、查询到寄存器 IIC_IF 的 SLV_STO 位，表示检测到 IIC 总线上有 STOP 发出。本次会话结束。

13、若设置中断使能，则可在 SLAVE 检测到 STA、RXF 接收完成、TXF 发送完成和 SLAVE 检测到 STO 时查到中断状态，可进行中断处理。

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
IIC0 IIC1	BASE: 0x400B9000 BASE: 0x400B9800				
IIC_CCFG	0x0	32	R/W	0x18	通用配置寄存器
IIC_CST	0x4	32	RO	0x06	通用状态寄存器
IIC_CTRANS	0x8	32	R/W	0x02	通用传输寄存器
IIC_RXDATA	0xC	32	RO	0x00	接收数据寄存器
IIC_TXDATA	0x10	32	R/W	0x00	发送数据寄存器
IIC_IE	0x14	32	R/W	0x00	中断使能寄存器
IIC_IF	0x18	32	R/W	0x01	中断标志寄存器
IIC_MCTRL	0x20	32	R/W	0x00	主机模式控制寄存器
IIC_MSPC	0x24	32	R/W	0x33f7f	时序配置寄存器
IIC_SCTRL	0x30	32	R/W	0x08	从机模式控制寄存器
IIC_SADDR	0x34	32	R/W	0x00	从机模式地址寄存器

寄存器描述

IIC_CCFG 寄存器（0x00）

位域	名称	类型	复位值	描述
31:7	RESERVED	RO	0	保留位

6:3	DNF	R/W	0x3	Receive SDA、SCL 数字噪声滤波（Digital Noise Filter） 0000：滤波不使能 0001：滤波使能，且滤波能力最大 1 个系统时钟 …… 1111：滤波使能，且滤波能力最大 15 个系统时钟
2	RESERVED	RO	0	保留位
1	MODE	R/W	0x0	模式控制 0：slave 模式 1：master 模式
0	EN	R/W	0	IIC 总线使能 0：不使能 1：使能

IIC_CST 寄存器（0x04）

位域	名称	类型	复位值	描述
31:14	RESERVED	R	0	保留位
13:12	SLV_RXDT	RO	0	Slave 接收到的数据类型。仅在 Slave 模式有效。 00：RXDATA 为空。 01：接收到的是地址。 10：接收到的是数据。 11：接收到的是 master code。仅当 MCDE=1 时有效。
11	SLV_STRETCH_BUSY	RO	0	Slave clock stretching 忙状态。仅在 slave 模式有效。 0：无 clock stretching。 1：有 clock stretching。
10	SLV_WR	RO	0	Slave 写状态。仅在 slave 模式有效。 1：slave 接收到 master 的写请求后有效。 0：slave 接收到 master 的读请求或 STOP 后，自动清除。

9	SLV_RD	RO	0	Slave 读状态。仅在 slave 模式有效。 1：slave 接收到 master 的读请求后有效。 0：slave 接收到 master 的写请求或 STOP 后，自动清除。
8	SLV_ACTIVE	RO	0	Slave 活跃状态。仅在 slave 模式有效。 0：slave 器件处于非活跃状态 1：slave 器件处于活跃状态。地址匹配成功后本位有效；接收到 STOP，或 Start_repeat 后的地址匹配不成功，自动清除。
7:3	RESERVED	RO	0	保留位。
2	SDA	RO	1	IIC SDA 状态。不受 IIC 总线使能影响 0：IIC SDA 为低 1：IIC SDA 为高
1	SCL	RO	1	IIC SCL 状态。不受 IIC 总线使能影响 0：IIC SCL 为低 1：IIC SCL 为高
0	BUSY	RO	0	总线忙状态。本位不受 IIC_CCFG.EN 位控制，当 EN 不使能时，仍然检测总线忙状态 0：总线不忙 1：总线忙，IIC 总线 START 至 STOP 期间有效

IIC_CTRANS 寄存器（0x08）

位域	名称	类型	复位值	描述
31:3	RESERVED	R	0	保留位
2	TXD_CLR	W, AC	0	发送数据寄存器清空。硬件自动清除。 0：不清空。 1：清 IF.TXE 位。
1	RX_ACK	RO	1	当作为发送方时，接收到的 ACK/NACK。硬件置位，IF.TXF 有效后即可查询此位；接收到 Start_repeat 或 STOP 会将此位置 1。 0：接收到 ACK 1：接收到 NACK

0	TX_ACK	R/W	0	当作为接收方时，发送 ACK/NACK。 0：发送 ACK。 1：发送 NACK。 注：以下情况，ACK/NACK 不由本位决定： A. slave 接收地址时，硬件自动发送 ACK/NACK。 B. slave MCDE 有效，接收到 master code 时，硬件自动返回 NACK。 C. slave 接收溢出时，硬件自动发送 NACK。

IIC_RXDATA 寄存器（0x0C）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	RXDATA	R	0	接收数据寄存器。IF.RXNE 为 1，表示本寄存器中存在有效数据。 注：在完成数据接收（不包含 ACK/NACK 发送）的时刻，更新此寄存器。 slave 接收地址字节情况，参见 RXF 位说明。

IIC_TXDATA 寄存器（0x10）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	TXDATA	R/W	0	发送数据寄存器。IF.TXE 为 0，表示本寄存器中存在待发送数据。

IIC_IE 寄存器（0x14）

位域	名称	类型	复位值	描述
31:18	RESERVED	R	0	保留位
17	MLTO	R/W	0	Master SCL LOW 超时中断使能。 0：不使能。 1：使能。
16:10	RESERVED	RO	0	保留位。
9	SLV_STO	R/W	0	Slave 检测到 STOP 中断使能。 0：不使能。 1：使能。
8	SLV_STA	R/W	0	Slave 检测到 START 中断使能。 0：不使能。 1：使能。
7:5	RESERVED	RO	0	保留位。
4	RXF	R/W	0	接收数据结束中断使能。 0：不使能。 1：使能。
3	TXF	R/W	0	发送数据结束中断使能。 0：不使能。 1：使能。
2	RXOVF	R/W	0	接收数据寄存器溢出中断使能。 0：不使能。 1：使能。
1:0	RESERVED	RO	0	保留位。

IIC_IF 寄存器（0x18）

位域	名称	类型	复位值	描述
31:18	RESERVED	R	0	保留位

17	MLTO	R, W1C	0	Master SCL LOW 超时。写 1 清除。仅在 master 模式有效。 0：未超时。 1：超时。SCL LOW 时间超过 1024 个由 MSPC 寄存器设置的 SCL LOW 时间。
16:10	RESERVED	RO	0	保留位
9	SLV_STO	R, W1C	0	Slave 检测到 STOP。写 1 清除。仅在 slave 模式下有效。 0：slave 未检测到 STOP。 1：slave 检测到 STOP。
8	SLV_STA	R, W1C	0	Slave 检测到 START。写 1 清除。仅在 slave 模式下有效。 0：slave 未检测到 START。 1：slave 检测到 START。
7:5	RESERVED	RO	0	保留位
4	RXF	R, W1C	0	接收结束。写 1 清除，包含 ACK/NACK 时间。 0：接收未结束。 1：接收结束。 注：slave 接收情况说明： 1、 Slave 器件 7 位地址模式下，slave 地址字节（含 R/W 位）接收完成，若地址匹配，则生成此中断。 2、 Slave 器件 10 位地址模式下，slave 地址的第 2 字节（ADDR[7:0]）接收完成，若 10 位地址匹配，则生成此中断；跟在 repeat START 之后的 slave 地址第 1 字节，若地址 8、9 位匹配，则生成此中断；跟在 START 之后的第 1 字节接收完成后，即使 ADDR[9:8]匹配，也不会生成此中断。 3. Slave 模式，SCTRL.MCDE=1，接收到 master code 时，会生成此中断。
3	TXF	R, W1C	0	发送结束。写 1 清除，包含 ACK/NACK 时间。 0：发送未结束，或没有发送。 1：发送结束。

2	RXOVF	R, W1C	0	接收数据寄存器溢出。软件写 1 清除。（更新的时刻点，不包含 ACK/NACK 发送） 0：无溢出。 1：当 RXDATA 非空时，又接收到新的字节，会产生溢出。溢出发生时，新数据丢失。 注：对于 slave 模式，如果 STRETCH 位有效，当接收数据寄存器非空，且又接收到新的字节，slave 器件会拉低 SCL 信号，直到 RXDATA 中的旧数据被读走，再把新数据存到 RXDATA 中，此情况不会产生溢出。
1	RXNE	RO	0	接收数据寄存器非空。 0：接收数据寄存器空，不存在未读取的接收数据。 1：接收数据寄存器非空，存在未读取的接收数据。 注：在接收完数据的时刻更新此位（不包含 ACK/NACK 发送时间）。 如果新数据接收完成时，旧数据未及时读取，分如下几种情况处理： Master 模式： 新数据丢失。同时置位 IF.RXOVF 位。 Slave 模式： A．STRETCH=0：新数据丢失。同时置位 IF. RXOVF 位，硬件自动发送 NACK。 B．STRETCH=1：正常返回 ACK，然后在 master 发送下一个字节前，slave 将 SCL hold 在低电平，直到旧数据被读走后，再将新数据更新到 RXDATA 寄存器中。最后释放 SCL。
0	TXE	RO	1	发送数据寄存器空。 0：发送数据寄存器非空，不允许写 TXDATA 寄存器。 1：发送数据寄存器空，允许写 TXDATA 寄存器。 注：在发送数据开始的时刻，发送数据被硬件读走后，此位被更新为 1（此时 IF.TXF 仍为 0）。 向 TXDATA 寄存器写入新数据，可清除此位。

IIC_MCTRL 寄存器（0x20）

位域	名称	类型	复位值	描述

31:4	RESERVED	RO	0	保留位
3	STO	W, AC	0	写 1，产生 STOP，完成后自动清零。
2	WR	W, AC	0	写 1，发送 TXDATA 中数据，完成后（含 ACK/NACK 时间）自动清零。 向本位写 1 前，要求 TXDATA 不能为空。否则，本位无法设置。 注意：WR 与 RD 位不能同时写 1。
1	RD	W, AC	0	写 1，接收数据到 RXDATA 中，完成后（含 ACK/NACK 时间）自动清零。
0	STA	W, AC	0	写 1，产生 START，完成后自动清零。 注意：允许 STA 和 WR 同时置位，优先发送 START。

IIC_MSPC 寄存器（0x24）

位域	名称	类型	复位值	描述
31:28	RESERVED	RO	0	保留位
27:24	DAT_HD	R/W	0x00	SDA 数据保持时间配置。（对 Master 和 Slave 有效） 对于 master：tHD;DAT=(DAT_HD + 4) * Tpclk 对于 slave：tHD;DAT=(DAT_HD + DNF + 6) * Tpclk 注：如果应用环境比较恶劣，则应注意，出现在 SDA 数据保持期间的毛刺有可能导致 SDA 的变化沿提前毛刺宽度的时间（如果此时 SCL 上无毛刺，则总线上会出现非预期的 STA、STOP）。在此情况下，应设置 DAT_HD 使得 tHD;DAT 大于最大的毛刺宽度。
23:16	CPD	R/W	0x03	时钟预分频，详见 SCL_HI 和 SCL_LOW 描述。（仅对 Master 模式有效）
15:8	SCL_HI	R/W	0x3f	SCL 时钟高电平时间配置。（仅对 Master 模式有效） tHIGH=((SCL_HI+1) * (CPD+1) + DNF + 6) * Tpclk

7:0	SCL_LOW	R/W	0x7f	SCL 时钟低电平时间配置。（对 Master 模式有效；在 slave 模式下，如果使能了 STRETCH 功能，且 SCTRL.ASDS 配置为 0，则需要配置本寄存器。在 slave 写 TXDATA 后，延迟本寄存器设置的时间，再释放 SCL。） tLOW=((SCL_LOW+1) * (CPD+1) + DAT_HD + 5) * Tpclk SCL 的周期为 tHIGH+tLOW。 推荐 tHIGH 与 tLOW 的比例为 1:2。

IIC_SCTRL 寄存器（0x30）

位域	名称	类型	复位值	描述
31:4	RESERVED	RO	0	保留位
3	ASDS	R/W	1	Stretching 后数据建立时间自适应使能。（Adaptive Stretching Data Setup） 0：自适应不使能。由 MSPC 设置。 1：自适应使能。在接收 master 地址时，自动检测 SCL 低电平时间，作为 stretching 后数据建立时间。 注：Slave-transmitter，当 STRETCH 寄存器设置为有效，且发生 stretching 的情况，在新数据准备好后，salve 会继续拉低 SCL 一段时间，以保证 SDA 线上满足数据建立时间的要求。

2	STRETCH	R/W	0	Clock stretching 使能控制。 0：Clock stretching 不使能。 1：Clock stretching 使能。 注：（slave 作为 receiver 时，当接收到新数据，但旧数据未被及时读取（IF.RXNE=1）：CTRANS. STRETCH_BUSY 变有效，在返回 ACK 后，将 SCL hold 在低电平，直到旧数据被读取后，把新数据更新到 RXDATA 中，同时 STRETCH_BUSY 变无效，再释放 SCL，开始下一个数据的接收。 slave 作为 transmitter 时，当发送结束（IF. TXF=1，含接收 ACK/NACK 时间），但新数据未准备好（TXE=1）：CTRANS.STRETCH_BUSY 变有效，将 SCL hold 在低电平，直到新数据准备好，延迟 SCL_LOW 时间后，STRETCH_BUSY 变无效，再释放 SCL，开始新数据的发送。）
1	MCDE	R/W	0	Master Code Detect Enable. 0：不检测 master code。 1：检测 master code。 本位有效时，slave 在 START 之后检测到 master code，会生成 IF.RXF 中断，并硬件设置 CST. SLV_RXDT 为 11。软件应保证 slave 地址设置不与 master code 冲突。
0	ADMD	R/W	0	slave 地址模式控制。 0：7 位地址模式 1：10 位地址模式

IIC_SADDR 寄存器（0x34）

位域	名称	类型	复位值	描述
31:24	RESERVED	RO	0	保留位

23:17	MASK_ADDR[7:1]	R/W	0	Slave 对应地址位掩码。 0：不掩码。 1：掩码对应位地址。掩码后，硬件匹配 slave 地址时，忽略被掩码的地址位。 对于 10 位地址模式，RXDATA 仅保存 ADDR[7:0]，所以不支持对 ADDR[9:8]的 mask。
16	MASK_ADDR0	R/W	0	Slave 对应地址位掩码。
15:10	RESERVED	RO	0	保留位。
9:8	ADDR[9:8]	R/W	0	7 位地址模式：don’t care 10 位地址模式：地址 bit9~bit8
7:1	ADDR[7:1]	R/W	0	地址 bit7~bit1
0	ADDR0	R/W	0	7 位地址模式：don’t care 10 位地址模式：地址 bit0

5.19 模数转换器（ADC）

5.19.1 概述

本芯片 SARADC 采用逐次逼近结构，可对 14 个有效通道的模拟信号进行采样，实现 A/D 转换。

其中通道 0 - 通道 10 用于外部模拟输入信号采样。并且通道 1 还可以选择对 OPA0 模拟输出进行采样，通道 10 还可以选择对 OPA1 模拟输出进行采样。

通道 11 和通道 12 为保留不使用。

通道 13 用于对芯片温度进行采样。

通道 14 用于对 1.2V 参考电压进行采样。

通道 15 用于对 AVDD/3 的电压进行采样。

其系统框图如下图所示：

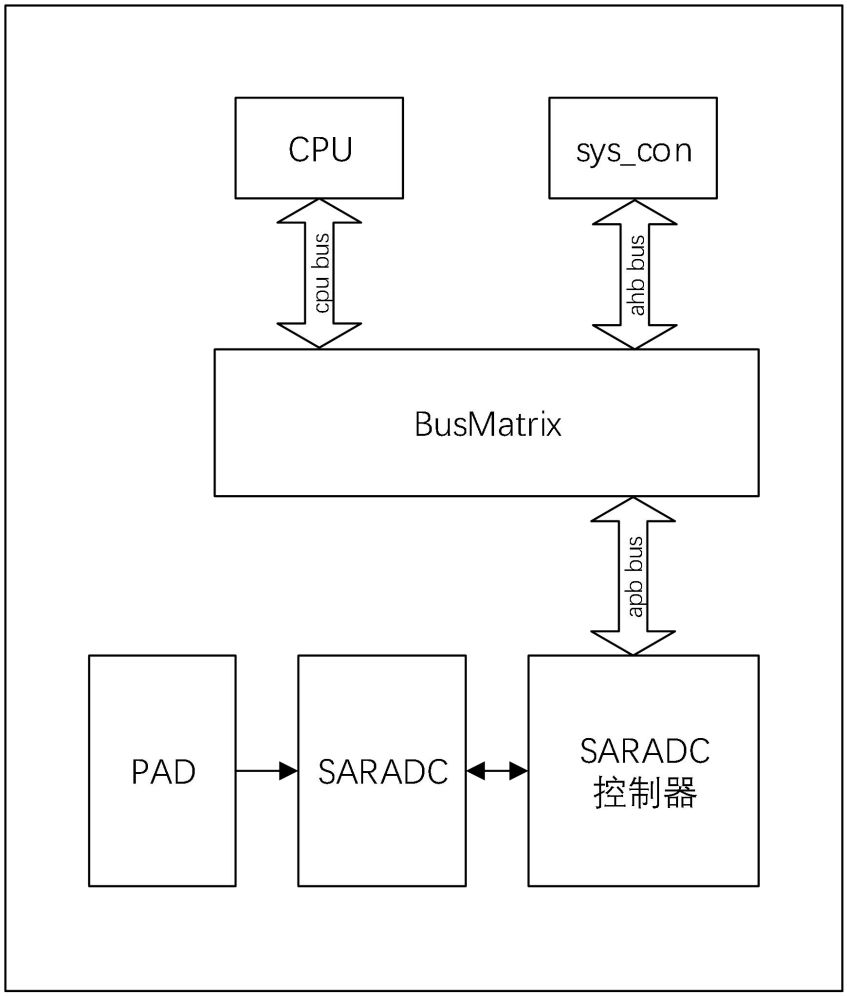


图 5-152 SARADC 系统框图

5.19.2特性

- 从外部 IO 口连接的 11 个模拟通道输入；
- 采样率最高可达 2.4M；
- 0-10 通道为高速通道，13-15 通道为低速通道；高速通道建立时间短，所需采样窗口短，反之低速通道建立时间长，需要适当增加采样窗口，来保证采样电压的准确性；
- 13 通道为片上温度传感器电压输入；
- 14 通道为 1.2V 参考电压输入；
- 15 通道为 AVDD/3 电压输入；

- 外部 ADC 参考电压 PAD_VREF，某些封装 PAD_VREF 与 AVDD_ADC 连接在一起；
- 独立的通道数据寄存器、全通道共用 16 级 FIFO 可选；
- 软件触发和硬件触发，硬件触发可选择 TIMERPLUS 模块的溢出信号和 PMWPLUS 的 trigger 信号；
- 支持采样一次或采样多次取平均可配置；
- 支持单次采样和连续采样可配置；
- 单次采样在启动之后，由低通道到高通道遍历全部所选择的通道一次；
- 连续采样在启动之后，不间断反复由低通道到高通道遍历全部所选择的通道，直到软件停止采样；
- 支持内部采样时钟、外部采样时钟可配置；
- 支持采样窗口可配置；
- 各通道转换完成状态、FIFO 满状态和 FIFO 半满状态都能产生中断；
- ADC 通道输入范围：0≤VIN≤Vref≤AVDD_ADC；
- 支持 DMA 读取 FIFO 采样数据；

5.19.3模块结构框图

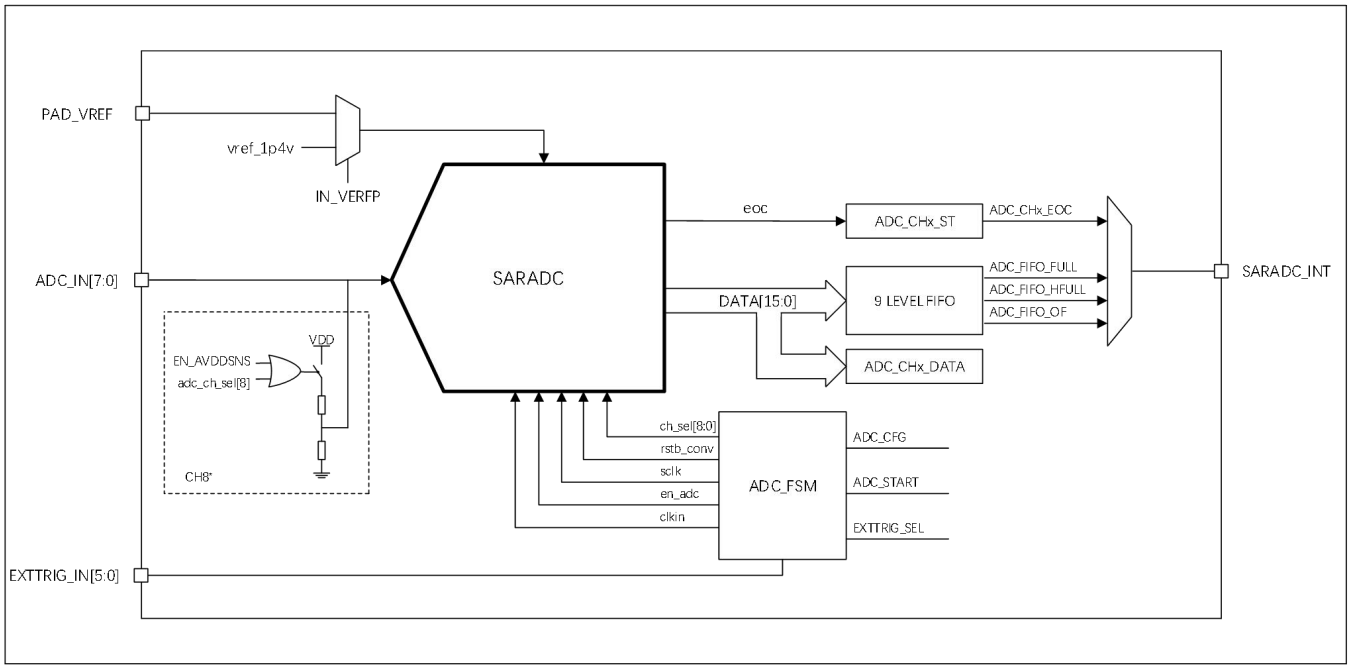


图 5-153 SARADC 模块结构图

SARADC 拥有 11 个外部 ADC 采样通道，外部 ADC 采样通道电压要小于参考电压 PAD_VREF。ADC 控制单元根据 ADC 配置产生相应的控制时序，控制 ADC 进行采样。采样结果可以选择存入各通道寄存器或者存入 FIFO 中，并且产生采样完成状态或 FIFO 半满和满状态。

SARADC 支持 12 个外部触发源，分别是来自 TIMERPLUS 模块的 TIMER_PLUS1_GOAL[1:0]（TIMERPLUS1 模块的高计数器溢出信号和低计数器溢出信号）和 TIMER_PLUS0_GOAL[1:0]（TIMERPLUS0 模块的高计数器溢出信号和低计数器溢出信号），以及来自 PWMPLUS 模块的 PWM_PLUS1_TRIGGER[3:0]（PWMPLUS1 模块的 trigger 信号）和 PWM_PLUS0_TRIGGER[3:0]（PWMPLUS0 模块的 trigger 信号）。

5.19.4功能描述

通道选择

本 SARADC 一共拥有 16 个采样通道，其中 14 为有效通道，2 个为无效通道（通道 11 和通道 12）不要使用。通过控制寄存器 ADC_CFG 的 ADC_CH_SEL 位来控制哪些通道在下一次的采样中生效。

发起采样之后控制单元将会由低通道到高通道遍历全部所被选择的有效通道，完成采样。

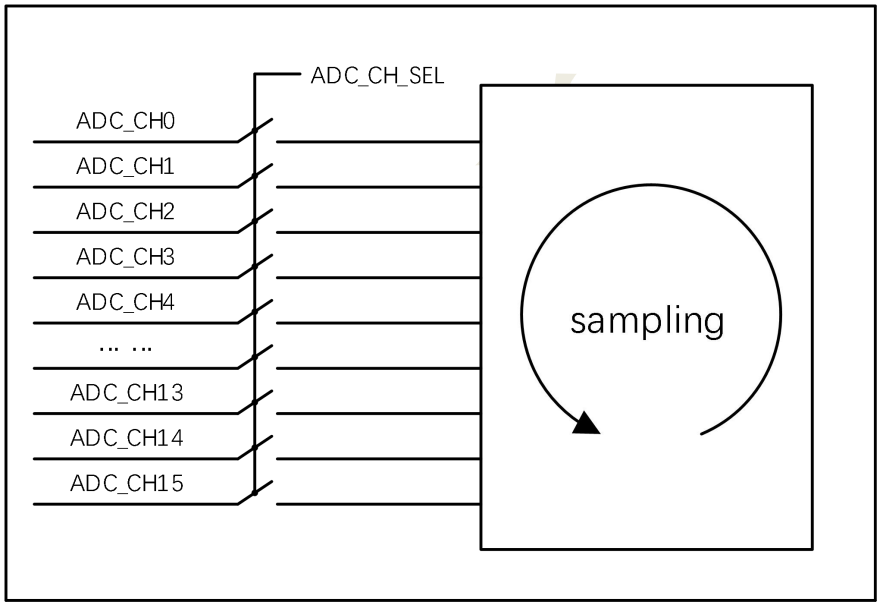


图 5-154　SARADC 通道选择示意图

采样一次与采样多次取平均

通过配置寄存器 SPL_NUM 来设置每个通道每次采样结果是以多少次采样取平均得出的。比如当 SPL_NUM 设置为采样 8 次取平均，那么每个通道在每次采样时，会进行 8 次完整的采样，最后 SARADC_CTRL 会把 8 次采样结果求得平均值，再存入通道寄存器或是 FIFO 里。

通道采样完成标志会在取平均之后产生。

单次采样与连续采样

通过配置寄存器 ADC_MD 来设置 SARADC 采样模式。SARADC 拥有单次采样与连续采样两种模式。单次采样模式时，SARADC 会遍历有效通道采样一次，然后停止等待下次采样；连续采样模式时，SARADC 会循环遍历有效通道进行采样，持续进行采样直到软件停止采样。

单通道单次采样

单通道单次采样：有效通道仅为一个单独通道即 ADC_CH_SEL 仅有一位为高、SARADC 采样模式为单次采样。

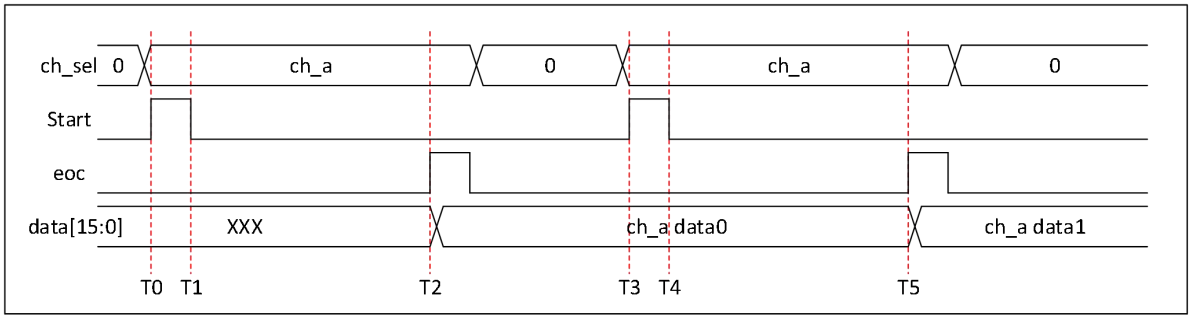


图 5-155 SARADC 单通道单次采样示意图

上图为通道 a 单次采样时序示意图：

- T0 时刻 CPU 编程 ADC_START 寄存器中的 ADC_START 位为 1，发起 SARADC 对通道 a 进行采样；
- T1 时刻 ADC_START 位被数字逻辑硬件清零；

- T2 时刻 SARADC 采样一次完成，输出通道 a 采样数据 ch_a data0，eoc 标志拉高，此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- 若想对该通道进行再次采样时，CPU 需要再次对 ADC_START 寄存器中的 ADC_START 位写 1。如上图 T3 时刻，发起 SARADC 对通道 a 的再次采样；
- T4 时刻 ADC_START 寄存器被数字逻辑硬件清零；
- T5 时刻 SARADC 采样一次完成，输出通道 a 采样数据 ch_a data1，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；

单通道单次采样时，CPU 启动 ADC_START 位会对该通道进行一次采样，ADC_START 会直接被硬件清零，采样完成后 SARADC 停止采样，等待 CPU 发起下次采样。

单通道连续采样

单通道连续采样：有效通道仅为一个单独通道即 ADC_CH_SEL 仅有一位为高、SARADC 采样模式为连续采样。

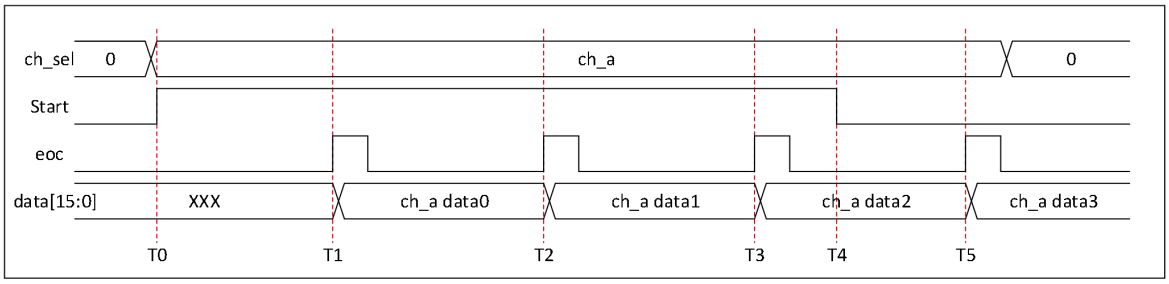


图 5-156 SARADC 单通道连续采样示意图

上图为通道 a 连续采样时序示意图：

- T0 时刻 CPU 编程 ADC_START 寄存器中的 ADC_START 位为 1，发起 SARADC 对通道 a 进行采样；

- T1 时刻 SARADC 采样一次完成，输出通道 a 采样数据 ch_a data0，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- T2 时刻 SARADC 采样一次完成，输出通道 a 采样数据 ch_a data1，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- T3 时刻 SARADC 采样一次完成，输出通道 a 采样数据 ch_a data2，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- 若 CPU 期望获取 N 次采样数据，则需要在获取了 N-1 次数据后通过编程 ADC_START 寄存器中的 ADC_START 位为 0，来停止 CPU 继续采样。见上图 T4 时刻，将 ADC_START 清为 0，则硬件电路在当前采样周期完成后将会停止 SARADC 继续采样；
- T5 时刻 SARADC 采样一次完成，输出通道 a 采样数据 ch_a data3，eoc 标志拉高。此时，CPU 可以获取最后一次采样数据；

单通道连续采样时，ADC_START 置一后 SARADC 会循环对此通道进行采样，直到 ADC_START 被程序清零，清零之后，SARADC 会在当前采样完成之后停止采样。

多通道单次采样

多通道单次采样：有效通道为多个通道即 ADC_CH_SEL 有多位为高、SARADC 采样模式为单次采样。

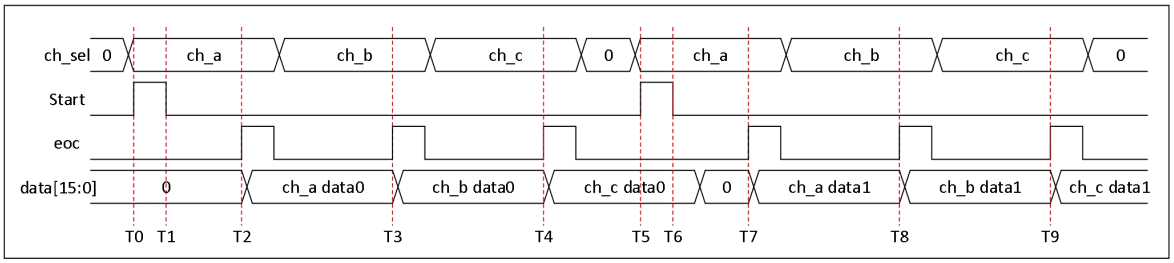


图 5-157 SARADC 多通道单次采样示意图

上图为通道 a、通道 b、通道 c 连续采样时序示意图，其中通道号 a<b<c，例通道 0<通道 5<通道 8：

- T0 时刻 CPU 编程 ADC_START 寄存器中的 ADC_START 位为 1，硬件发起 SARADC 依次对通道 a、b、c 进行采样；
- T1 时刻 ADC_START 寄存器被数字逻辑硬件清零；
- T2 时刻 SARADC 采样通道 a 完成，输出通道 a 采样数据 ch_a data0，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- T3 时刻 SARADC 采样通道 b 完成，输出通道 b 采样数据 ch_b data0，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- T4 时刻 SARADC 采样通道 c 完成，输出通道 c 采样数据 ch_c data0，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- 若想对该序列通道进行再次采样时，CPU 需要再次对 ADC_START 寄存器中的 ADC_START 位写 1。如上图 T5 时刻，发起 SARADC 对通道 a、b、c 的再次采样；
- T6 时刻 ADC_START 寄存器被数字逻辑硬件清零；
- T7 时刻 SARADC 采样通道 a 完成，输出通道 a 采样数据 ch_a data1，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- T8 时刻 SARADC 采样通道 b 完成，输出通道 b 采样数据 ch_b data1，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- T9 时刻 SARADC 采样通道 c 完成，输出通道 c 采样数据 ch_c data1，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；

多通道单次采样时，启动 ADC_START 会对全部所选择的通道进行一次采样，ADC_START 会直接被清零，每个通道采样一次，全部采样完成后 SARADC 停止采样，等待下次发起采样。

多通道连续采样

多通道连续采样：有效通道为多个通道即 ADC_CH_SEL 有多位为高、SARADC 采样模式为连续采样。

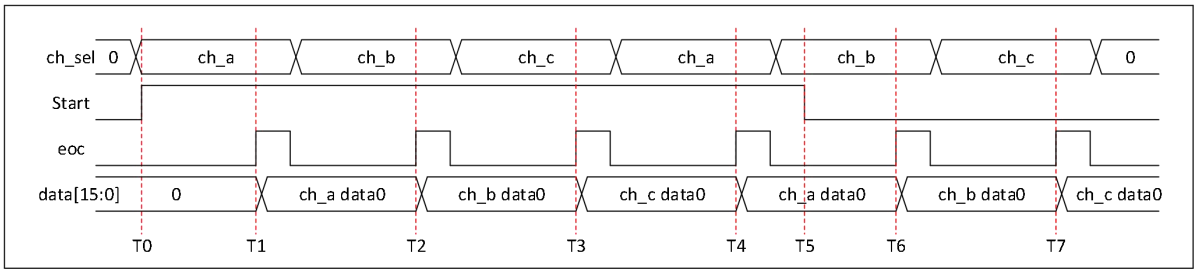


图 5-158 SARADC 多通道连续采样示意图

上图为通道 a、通道 b、通道 c 连续采样时序示意图，其中通道号 a<b<c，例如通道 0<通道 5<通道 8：

- T0 时刻 CPU 编程 ADC_START 寄存器中的 ADC_START 位为 1，发起 SARADC 依次对通道 a、b、c 进行采样；
- T1 时刻 SARADC 采样通道 a 完成，输出通道 a 采样数据 ch_a data0，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- T2 时刻 SARADC 采样通道 b 完成，输出通道 b 采样数据 ch_b data0，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；

- T3 时刻 SARADC 采样通道 c 完成，输出通道 c 采样数据 ch_c data0，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- 若 CPU 期望获取 N 次采样数据，则需要在获取了 N-1 次数据后，并且在第 N 次全部通道数据转换完成之前，通过编程 ADC_START 寄存器中的 ADC_START 位为 0，来停止 CPU 继续采样。见上图 T5 时刻为例，将 ADC_START 清为 0，则硬件电路在当前采样周期全部有效通道转换完成后将会停止 SARADC 采样；
- T4 时刻 SARADC 采样通道 a 完成，输出通道 a 采样数据 ch_a data1，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- T6 时刻 SARADC 采样通道 b 完成，输出通道 b 采样数据 ch_b data1，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；
- T7 时刻 SARADC 采样通道 c 完成，输出通道 c 采样数据 ch_c data1，eoc 标志拉高。此时 CPU 可通过读取相应通道数据寄存器或 FIFO 数据寄存器获取采样数据，以及通过读取相应状态寄存器获取采样完成标志；

多通道连续采样时，ADC_START 置一后 SARADC 会循环对全部所选通道进行遍历采样，直到 ADC_START 被程序清零，清零之后，SARADC 会在当前循环采样完成之后停止采样。

通道存储与 FIFO

SARADC 能够以各通道采样数据寄存器存储采样结果，也可以使用内置深度为 16 字节的 FIFO 来存储全部通道的采样结果。通过配置寄存器 ADC_MEM_MD 来设置采样数据存储位置。

使用通道寄存器时，各通道的通道寄存器会存储自己通道的采样值，若数据没有及时读出，新的采样数据会覆盖旧的数据。

使用 FIFO 存储时，FIFO 会存储选通通道的采样值，若数据没有及时读出，新的采样数据会丢掉。

内部采样时钟方式与外部采样时钟方式

SARADC 有两种采样方式：一种是内部采样时钟方式，另一种是外部采样时钟方式，通过配置寄存器 ADC_IN_SMPL 进行选择。

不同的采样时钟方式需要配置相对应的窗口设置。内部采样时钟方式的窗口设置可通过 ADC_CFG 寄存器中的 IN_SMPL_WIN 位软件编程，可配置参数为 1/3/5/7/9/11/13/15 共八种选择；外部采样时钟方式的窗口设置可通过 ADC_CFG 寄存器中的 SMPL_SETUP 位软件编程，可配置参数为 1/2/4/8/16/32/64/128 共八种选择。

CPU 触发与外部信号触发

SARADC 提供了可选择的外部信号触发源，通过配置寄存器 ADC_TRIG 来选择触发方式。共有 12 个外部触发源信号。

其中 ADC_EXTTRIG_SEL[3:0]分别用于控制 PWM_PLUS0_TRIGGER[3:0]；

ADC_EXTTRIG_SEL[7:4]分别用于控制 PWM_PLUS1_TRIGGER[3:0]；

ADC_EXTTRIG_SEL[9:8]分别用于控制 TIMER_PLUS0_GOAL[1:0]；

ADC_EXTTRIG_SEL[11:10]分别用于控制 TIMER_PLUS1_GOAL[1:0]；

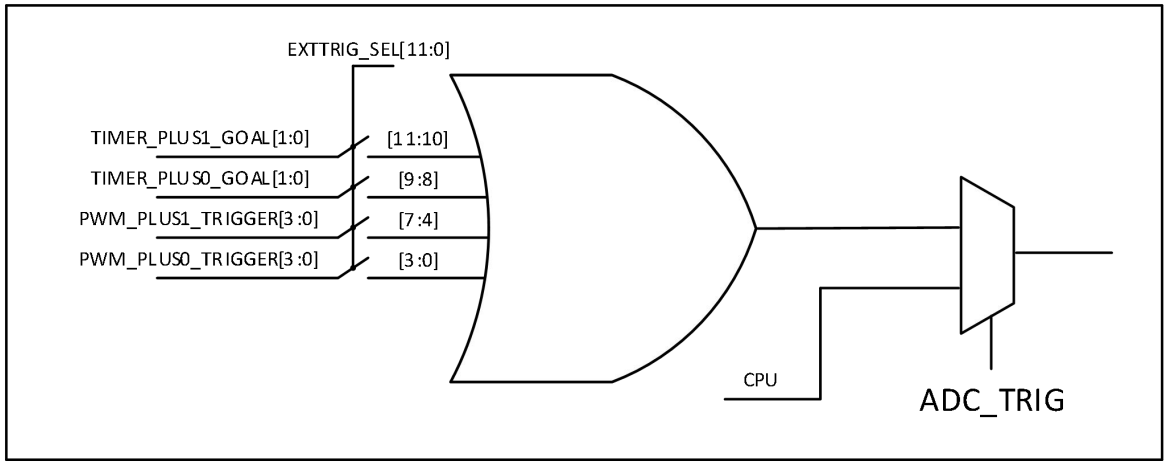


图 5-159 SARADC 外部触发选择

中断

SARADC 提供了 18 个中断，详情如下图。

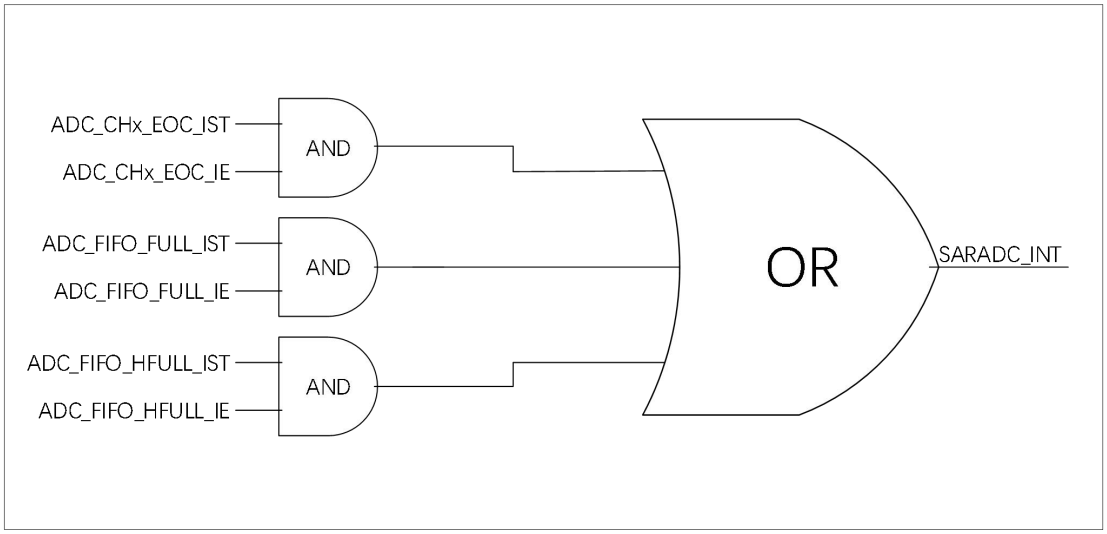


图 5-160 SARADC 中断标志和中断信号示意图

操作流程

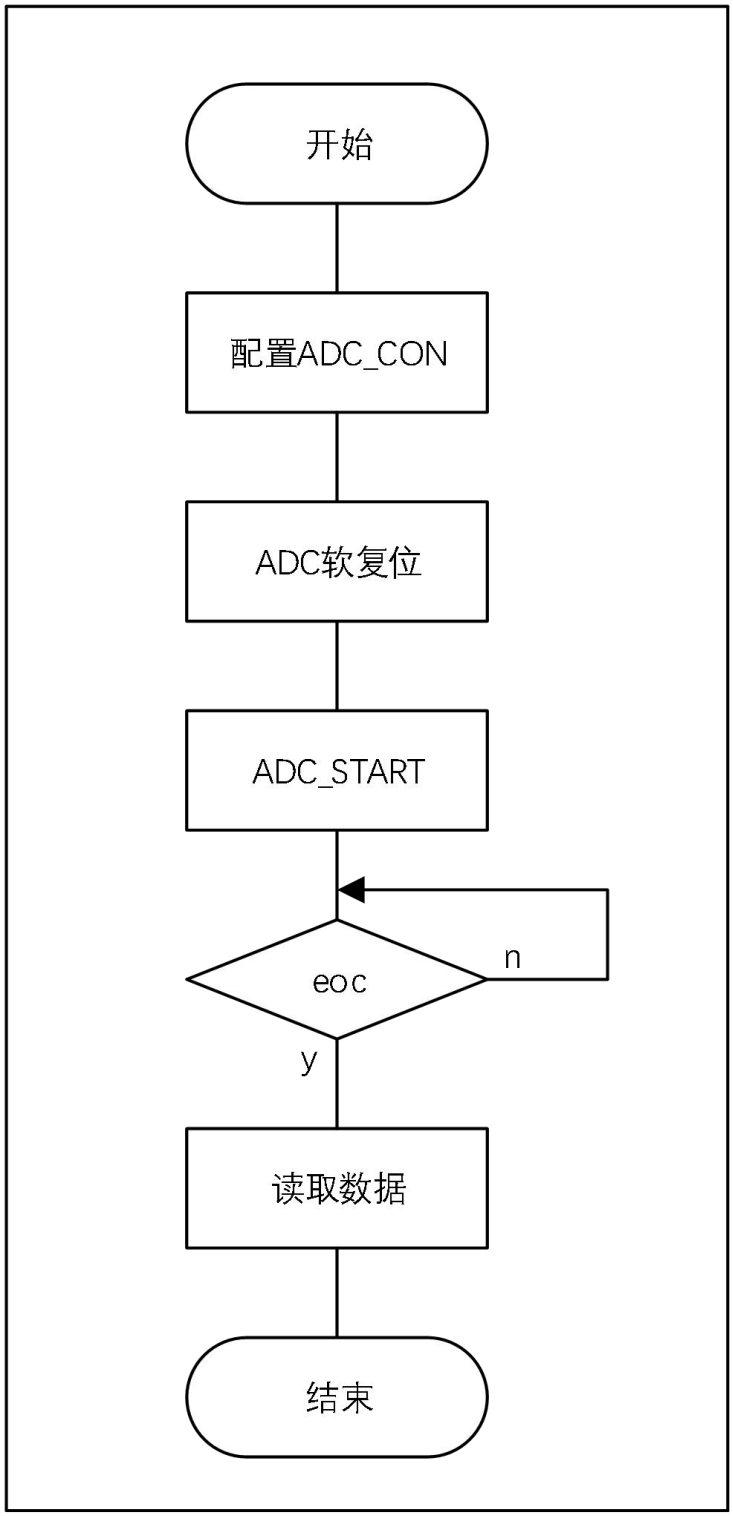


图 5-161 SARADC 操作流程图

- 打开 SARADC 模块时钟
- 配置引脚为 ADC 功能
- 配置 ADC 的转换时钟分频值
- 配置 ADC 的转换通道
- 配置采样取平均
- 配置转换模式为单次模式还是连续模式
- 配置数据存储为 FIFO 还是通道
- 配置采样时钟为内部采样还是外部采样
- 配置内部或者外部采样窗口
- 配置 KD 数据是否有效
- 配置 offset 数据是否有效
- 配置通道转换完成中断是否使能
- 配置 FIFO 满中断是否使能
- 配置 FIFO 半满中断是否使能
- 发起 ADC 软复位
- 开始 ADC 采样
- 等待采样结束
- 读出采样数据

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
SARADC	BASE: 0x400BA000				
ADC_CFG	0x00	32	R/W	0x2100000	ADC 配置寄存器
ADC_START	0x04	32	R/W	0x00	ADC 启动寄存器
ADC_IE	0x08	32	R/W	0x00	ADC 中断使能寄存器
ADC_IF	0x0c	32	RW	0x00	ADC 中断状态寄存器
ADC_CH0_STAT	0x10	32	R/W	0x00	ADC 通道 0 状态寄存器
ADC_CH0_DATA	0x14	32	R/W	0x00	ADC 通道 0 数据寄存器
ADC_CH1_STAT	0x18	32	R/W	0x00	ADC 通道 1 状态寄存器
ADC_CH1_DATA	0x1C	32	R/W	0x00	ADC 通道 1 数据寄存器
ADC_CH2_STAT	0x20	32	R/W	0x00	ADC 通道 2 状态寄存器
ADC_CH2_DATA	0x24	32	R/W	0x00	ADC 通道 2 数据寄存器
ADC_CH3_STAT	0x28	32	R/W	0x00	ADC 通道 3 状态寄存器
ADC_CH3_DATA	0x2C	32	R/W	0x00	ADC 通道 3 数据寄存器
ADC_CH4_STAT	0x30	32	R/W	0x00	ADC 通道 4 状态寄存器
ADC_CH4_DATA	0x34	32	R/W	0x00	ADC 通道 4 数据寄存器
ADC_CH5_STAT	0x38	32	R/W	0x00	ADC 通道 5 状态寄存器
ADC_CH5_DATA	0x3C	32	R/W	0x00	ADC 通道 5 数据寄存器
ADC_CH6_STAT	0x40	32	R/W	0x00	ADC 通道 6 状态寄存器
ADC_CH6_DATA	0x44	32	R/W	0x00	ADC 通道 6 数据寄存器
ADC_CH7_STAT	0x48	32	R/W	0x00	ADC 通道 7 状态寄存器
ADC_CH7_DATA	0x4C	32	R/W	0x00	ADC 通道 7 数据寄存器
ADC_CH8_STAT	0x50	32	R/W	0x00	ADC 通道 8 状态寄存器
ADC_CH8_DATA	0x54	32	R/W	0x00	ADC 通道 8 数据寄存器
ADC_CH9_ST	0x58	32	R/W	0x00	ADC 通道 9 状态寄存器
ADC_CH9_DATA	0x5c	32	R/W	0x00	ADC 通道 9 数据寄存器
ADC_CH10_ST	0x60	32	R/W	0x00	ADC 通道 10 状态寄存器
ADC_CH10_DATA	0x64	32	R/W	0x00	ADC 通道 10 数据寄存器
ADC_CH11_ST	0x68	32	R/W	0x00	ADC 通道 11 状态寄存器

ADC_CH11_DATA	0x6c	32	R/W	0x00	ADC 通道 11 数据寄存器
ADC_CH12_ST	0x70	32	R/W	0x00	ADC 通道 12 状态寄存器
ADC_CH12_DATA	0x74	32	R/W	0x00	ADC 通道 12 数据寄存器
ADC_CH13_ST	0x78	32	R/W	0x00	ADC 通道 13 状态寄存器
ADC_CH13_DATA	0x7c	32	R/W	0x00	ADC 通道 13 数据寄存器
ADC_CH14_ST	0x80	32	R/W	0x00	ADC 通道 14 状态寄存器
ADC_CH14_DATA	0x84	32	R/W	0x00	ADC 通道 14 数据寄存器
ADC_CH15_ST	0x88	32	R/W	0x00	ADC 通道 15 状态寄存器
ADC_CH15_DATA	0x8c	32	R/W	0x00	ADC 通道 15 数据寄存器
ADC_FIFO_STAT	0xa0	32	R/W	0x04	ADC FIFO 状态寄存器
ADC_FIFO_DATA	0xa4	32	R/W	0x00	ADC FIFO 数据寄存器
EXTTRIG_SEL	0xb0	32	R/W	0x00	外部信号触发 ADC 选择寄存器
ADC_CALIB_OFFSET	0xf0	32	R/W	0x00	ADC 校准 OFFSET 寄存器
ADC_CALIB_KD	0xf4	32	R/W	0x00	ADC 校准 KD 寄存器

寄存器描述

ADC_CFG 寄存器（0x00）

位域	名称	类型	复位值	描述
31:30	RESERVED	R	0	保留位
29	DMA_EN	R/W	0	DMA 读取 FIFO 使能 0：CPU 读取 FIFO 1：DMA 读取 FIFO
28	ADC_TRIG	R/W	0	ADC 触发源选择 0：选择 CPU 触发 ADC 采样 1：选择外部信号触发 ADC 采样

27	ADC_EN	R/W	0	ADC 使能位 0：禁能　1：使能
26:24	IN_SMPL_WIN	R/W	010	ADC 内部采样时钟方式采样窗口选择 000：采样建立时间保持 1 个 ADC 时钟周期 001：采样建立时间保持 3 个 ADC 时钟周期 010：采样建立时间保持 5 个 ADC 时钟周期 011：采样建立时间保持 7 个 ADC 时钟周期 100：采样建立时间保持 9 个 ADC 时钟周期 101：采样建立时间保持 11 个 ADC 时钟周期 110：采样建立时间保持 13 个 ADC 时钟周期 111：采样建立时间保持 15 个 ADC 时钟周期
23	ADC_SMPL_CLK	R/W	0	ADC 采样模式选择 1：表示 ADC 采用内部采样时钟方式 0：表示 ADC 采用外部采样时钟方式
22	ADC_MEM_MODE	R/W	0	ADC 数据存储方式选择： 0：ADC 数据存储为 FIFO 模式； 1：ADC 数据存储为通道模式；
21:19	SMPL_SETUP	R/W	010	ADC 外部采样时钟方式采样窗口选择 000：采样建立时间保持 1 个 ADC 时钟周期 001：采样建立时间保持 2 个 ADC 时钟周期 010：采样建立时间保持 4 个 ADC 时钟周期 011：采样建立时间保持 8 个 ADC 时钟周期 100：采样建立时间保持 16 个 ADC 时钟周期 101：采样建立时间保持 32 个 ADC 时钟周期 110：采样建立时间保持 64 个 ADC 时钟周期 111：采样建立时间保持 128 个 ADC 时钟周期
18	CONT	R/W	0	ADC 采样工作模式 0：单次采样 1：连续采样
17:16	AVG	R/W	0	一次启动 ADC 采样取平均次数配置寄存器 00：1 次采样取平均 01：2 次采样取平均 10：4 次采样取平均 11：8 次采样取平均

15:0	ADC_CH_SEL	R/W	0	ADC 通道选择寄存器 Bit15-bit0 分别对应通道 15-通道 0。 对应位配置为 1 则表示该通道有效。

ADC_START 寄存器（0x04）

位域	名称	类型	复位值	描述
30:4	RESERVED	R	0	保留位
3	FIFO_CLR	R/W	0	FIFO 清除使能 0：禁能　1：使能 写 1 清 FIFO
2	SOFT_RESET	R/W	0	ADC 软复位 0 复位有效 在 ADC 使能之后，启动之前必须要复位一次 ADC
1	BUSY	R	0	ADC 工作状态 1 表示忙

0	START	R/W	0	ADC 启动信号 0：禁能　1：使能 该位写 1，则启动一次转换。可以 ADC_ CONT 配合使用 若 ADC_ CONT 处于单次采样模式，则该位置 1 后，将对有效通道依次轮询进行采样转换，并将转换的数据保存在相应通道的 FIFO 或寄存器中。转换完成后硬件会自动清零。 若 ADC_ CONT 处于连续采样模式，则该位置 1 表示启动 ADC 转换，清零后表示停止 ADC 转换。启动 ADC 转换后，将对有效通道依次轮询进行采样转换，并将转换的数据保存在相应通道的 FIFO 或寄存器中。每次转换完成后判断该位是否为 1，若为 1 则继续转换，若为 0 则停止转换。

ADC_IE 寄存器（0x08）

位域	名称	类型	复位值	描述
30:18	RESERVED	R	0	保留位
17	ADC_FIFO_ HFULL_IE	R/W	0	ADC FIFO 半满中断使能 0：禁能　1：使能
16	ADC_FIFO_F ULL_IE	R/W	0	ADC FIFO 满中断使能 0：禁能　1：使能
15:0	ADC_CHx_E OC_IE	R/W	0	ADC 通道 x 数据转换完成中断使能

ADC_IF 寄存器（0x0C）

位域	名称	类型	复位值	描述
30:18	RESERVED	R	0	保留位
17	ADC_FIFO_HFULL_IF	R/W	0	ADC FIFO 半满中断状态 写 1 清零
16	ADC_FIFO_FULL_IF	R/W	0	ADC FIFO 满中断状态 写 1 清零
15:0	ADC_CHx_EOC_IST	R/W	0	ADC 通道 x 数据转换完成中断状态 写 1 清零

ADC_CHx_STAT 寄存器

位域	名称	类型	复位值	描述
30:1	RESERVED	R	0	保留位
0	ADC_CH_EOC	R	0	ADC 通道 x 数据转换完成标志 1：表示 ADC 对通道 x 一次采样转换完成 通过向 ADC_IF 对应位写 1 可清零

ADC_CHx_DATA 寄存器

位域	名称	类型	复位值	描述
30:16	RESERVED	R	0	保留位
15:12	ADC_CH_NUM	R	0	ADC 数据对应的通道编号
11:0	ADC_CH_DATA	R	0	ADC 通道 x 数据寄存器 注：溢出后，再次转换的数据会覆盖旧数据

ADC_FIFO_STAT 寄存器（0xA0）

位域	名称	类型	复位值	描述
30:8	RESERVED	R	0	保留位
7:4	ADC_FIFO_LEVEL	R	0	ADC 数据 FIFO 水位 0000：未满时表示 FIFO 有 0 个数据，满时表示 FIFO 有 16 个数据； 0001：表示 FIFO 有 1 个数据； 0010：表示 FIFO 有 2 个数据； 0011：表示 FIFO 有 3 个数据； 0100：表示 FIFO 有 4 个数据； 0101：表示 FIFO 有 5 个数据； 0110：表示 FIFO 有 6 个数据； 0111：表示 FIFO 有 7 个数据； 1000：表示 FIFO 有 8 个数据； 1001：表示 FIFO 有 9 个数据； 1010：表示 FIFO 有 10 个数据； 1011：表示 FIFO 有 11 个数据； 1100：表示 FIFO 有 12 个数据； 1101：表示 FIFO 有 13 个数据； 1110：表示 FIFO 有 14 个数据； 1111：表示 FIFO 有 15 个数据；
3	RESERVED	R	0	保留位
2	ADC_FIFO_EMPTY	R	1	ADC 数据 FIFO 空标志 1：表示 FIFO 空 0：表示 FIFO 非空
1	ADC_FIFO_HFULL	R	0	ADC 数据 FIFO 半满标志 1：表示 FIFO 半满 0：表示 FIFO 非半满
0	ADC_FIFO_FULL	R	0	ADC 数据 FIFO 满标志 1：表示 FIFO 满 0：表示 FIFO 非满

ADC_FIFO_DATA 寄存器（0xA4）

位域	名称	类型	复位值	描述
30:16	RESERVED	R	0	保留位
15:12	ADC_FIFO_NUM	R	0	ADC 数据对应的通道编号
11:0	ADC_FIFO_DATA	R	0	ADC 数据 FIFO 寄存器 注：溢出后，再次转换的数据会被丢掉

ADC_ EXTTRIG_SEL 寄存器（0xB0）

位域	名称	类型	复位值	描述
30:12	RESERVED	R	0	保留位
11:0	EXTTRIG_SEL	R/W	0	外部触发 ADC 采样信号选择控制 Bit11-bit0：表示分别用于控制 exttrig_in[11:0] 是否触发 ADC 的控制位 0 表示无效；1 表示有效

ADC_CALIB_OFFSET 寄存器（0xF0）

位域	名称	类型	复位值	描述
30:17	RESERVED	R	0	保留位
16	OFFSET_VALID	R/W	0	OFFSET 数据是否有效
15:8	RESERVED	R	0	保留位
7:0	OFFSET	R/W	0	ADC 数据校准的 OFFSET 值。计算出的 OFFSET 需要存入该寄存器。用于在使用 ADC 时进行校准。

ADC_CALIB_KD 寄存器（0xF4）

位域	名称	类型	复位值	描述
30:17	RESERVED	R	0	保留位
16	KD_VALID	R/W	0	KD 数据是否有效
15:10	RESERVED	R	0	保留位
9:0	KD	R/W	0	ADC 数据校准 K 值的小数部分。计算出的 K 值的小数部分需要存入该寄存器。用于在使用 ADC 时进行校准。若 K 大于 1，则符号位为 1，若 K 小于 1，则符号位为 0。

5.20 比较器（COMP）

5.20.1 概述

本芯片提供了三个通用的比较器单元，每个比较器都可独立配置使用。使用特定 IO 端口作为输入端，具体端口如本模块的结构框图所示。它们具有多种功能，包括轨到轨的比较器、可独立编程配置迟滞电压和独立编程配置输出滤波功能等。

5.20.2 特性

- 轨到轨的比较器
- 每个比较器可独立编程配置迟滞电压：0mv 迟滞、24mv 迟滞、40mv 迟滞和 60mv 迟滞四种挡位
- 每个比较器可独立编程配置输出滤波功能：不滤波、2 个系统时钟周期滤波、4 个系统时钟周期滤波和 8 个系统时钟周期滤波四种滤波方式
- 每个比较器都具有中断产生的能力
- 可以组合成窗口比较器

5.20.3 模块结构框图

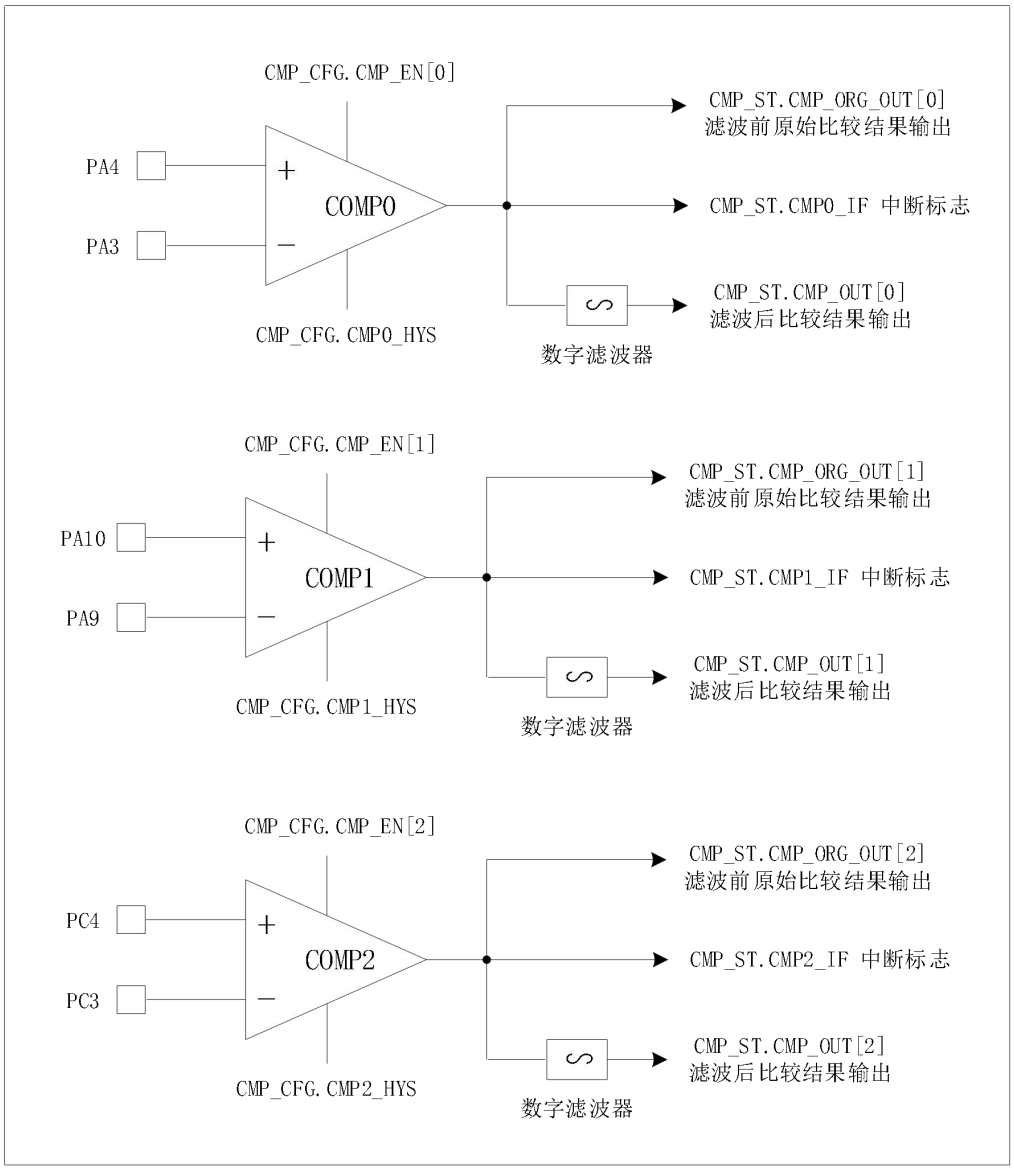


图 5-162 比较器结构框图

如上图所示：

比较器 0 的 VP 端由 PA4 引脚输入，VN 端由 PA3 引脚输入；并且该比较器可由其独立的使能寄存器 CMP_CFG.CMP_EN[0]直接编程。

比较器 1 的 VP 端由 PA10 引脚输入，VN 端由 PA9 引脚输入；并且该比较器可由其独立的使能寄存器 CMP_CFG.CMP_EN[1]直接编程。

比较器 2 的 VP 端由 PC4 引脚输入，VN 端由 PC3 引脚输入；并且该比较器可由其独立的使能寄存器 CMP_CFG.CMP_EN[2]直接编程。

5.20.4 功能描述

比较器输入引脚及内部输出信号

三个独立比较器分别具有独立的输入和输出。

比较器 0（COMP0）的正端通过 PA4 输入，负端通过 PA3 输入。通过 CMP_ST 寄存器的 CMP_ORG_OUT[0]位可以查询原始比较结果，通过 CMP_ST 寄存器的 CMP_OUT[0]位可以查询经过滤波后的比较结果，通过 CMP_ST 寄存器的 CMP_IF[0]位可以查询中断标志。

比较器 1（COMP1）的正端通过 PA10 输入，负端通过 PA9 输入。通过 CMP_ST 寄存器的 CMP_ORG_OUT[1]位可以查询原始比较结果，通过 CMP_ST 寄存器的 CMP_OUT[1]位可以查询经过滤波后的比较结果，通过 CMP_ST 寄存器的 CMP_IF[1]位可以查询中断标志

比较器 2（COMP2）的正端通过 PC4 输入，负端通过 PC3 输入。通过 CMP_ST 寄存器的 CMP_ORG_OUT[2]位可以查询原始比较结果，通过 CMP_ST 寄存器的 CMP_OUT[2]位可以查询经过滤波后的比较结果，通过 CMP_ST 寄存器的 CMP_IF[2]位可以查询中断标志

迟滞

比较器的迟滞功能可通过各自的迟滞寄存器 CMPx_HYS 独立配置，用于避免由于干扰而导致比较输出错误。

通过可编程的迟滞寄存器可以选择四种迟滞挡位：0mV、24mV、40mV 和 60mV。

比较器迟滞示意如图所下：

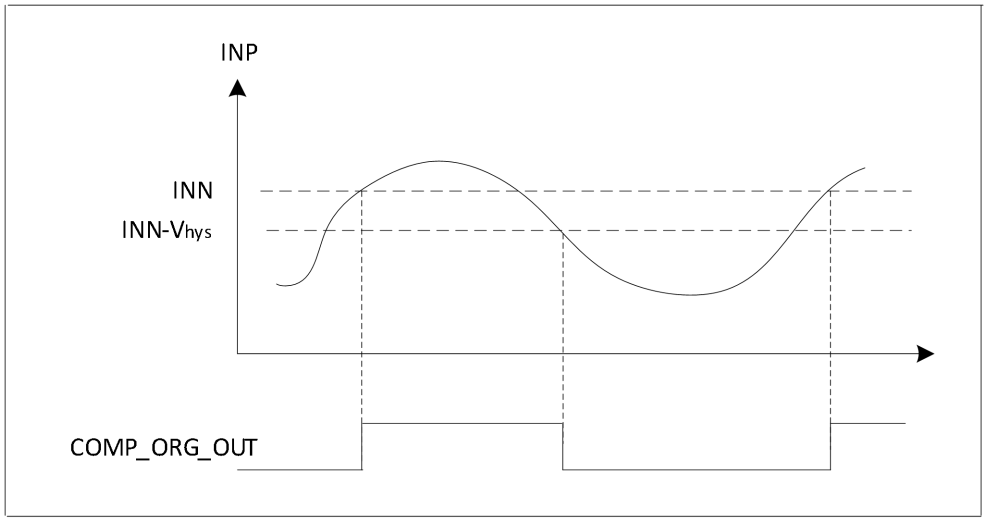


图 5-163 比较器迟滞示意图

滤波

为了提高比较器可靠性，减少由于干扰导致的误判，对比较器的输出结果进行了数字滤波。

通过可编程的滤波寄存器 CMP_CFG.CMP_FILTER 可以选择四种滤波挡位：0 个系统时钟周期滤波、2 个系统时钟周期滤波、4 个系统时钟周期滤波和 8 个系统时钟周期滤波。

用于消除毛刺的滤波功能示意图如下所示：

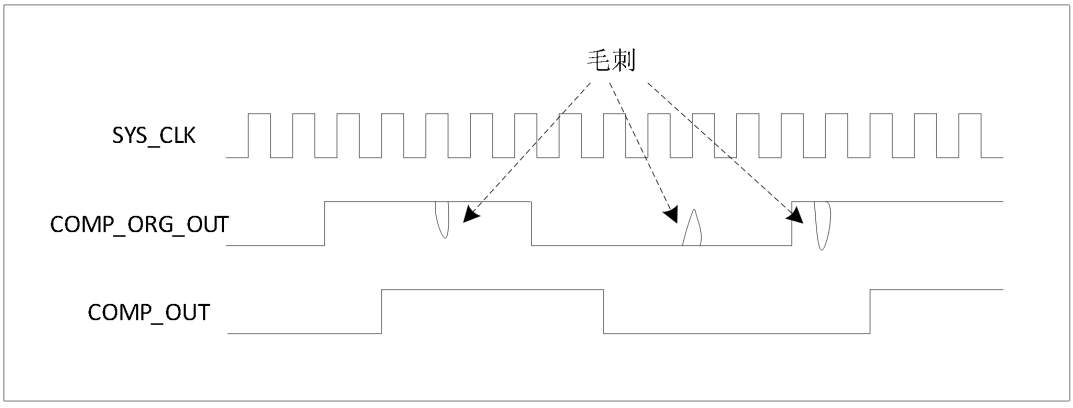


图 5-164 比较器消除毛刺的滤波功能示意图

该配置对所有比较器有效。

中断

每一个比较器都有各自的两个比较器输出状态标志：一个是表示由 0 到 1 的变化状态标志；另一个是表示由 1 到 0 的变化状态标志。

三个独立的比较器由各自的中断使能控制，共同输出给 CPU 的中断控制器产生芯片的比较器中断源。

中断标志及中断示意图如下：

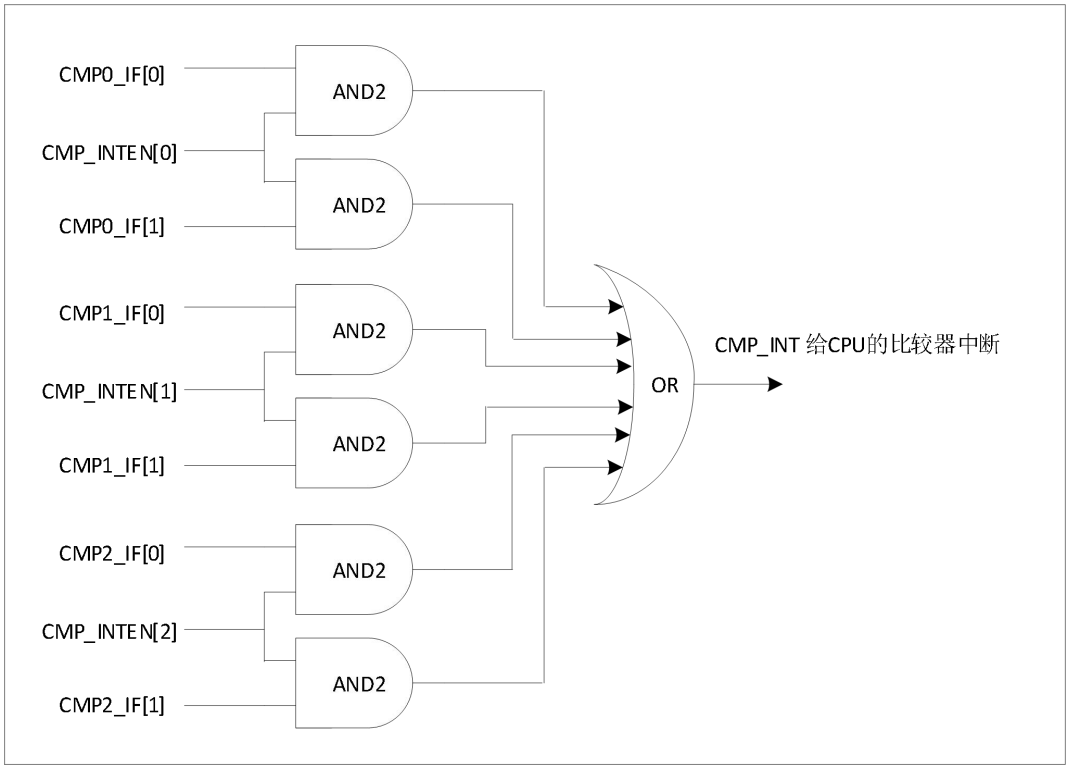


图 5-165 比较器中断标志及中断示意图

寄存器映射

名称	偏移量	类型	复位值	描述
COMPARATOR	BASE: 0x40000000			

CMP_CFG	0x120	R/W	0x200000	比较器配置寄存器
CMP_ST	0x124	R/W	0x00	比较器状态寄存器

寄存器描述

CMP_CFG 寄存器（0x120）

位域	名称	类型	复位值	描述
31:26	RESERVED	R	0	保留位
25:24	CMP_FILTER	R/W	10	比较器输出滤波控制 00：不滤波 01：2 个系统时钟周期滤波 10：4 个系统时钟周期滤波 11：8 个系统时钟周期滤波
23:22	RESERVED	R	0	保留位
21:20	CMP2_HYS	R/W	0	CMP2 迟滞选择 00：24mv 01：40mv 10：60mv 11：0mv
19:18	CMP1_HYS	R/W	0	CMP1 迟滞选择 00：24mv 01：40mv 10：60mv 11：0mv
17:16	CMP0_HYS	R/W	0	CMP0 迟滞选择 00：24mv 01：40mv 10：60mv 11：0mv
15:11	RESERVED	R	0	保留位
10:8	CMP_INTEN	R/W	0	比较器中断使能控制位 1：使能 0：关闭 Bit10-bit8 分别控制 CMP2-CMP0
7:3	RESERVED	R	0	保留位

2:0	CMP_EN	R/W	0	比较器使能控制位 1：使能 0：关闭 Bit2-bit0 分别控制 CMP2-CMP0

CMP_ST 寄存器（0x124）

位域	名称	类型	复位值	描述
31:22	RESERVED	R	0	保留位
21:20	CMP2_IF	R/W	0	比较器 2 的中断标志 Bit21 为 1 表示比较器 2 输出发生从 0 到 1 的变化 Bit20 为 1 表示比较器 2 输出发生从 1 到 0 的变化 二者都为写 1 清零
19:18	CMP1_IF	R/W	0	比较器 1 的中断标志 Bit19 为 1 表示比较器 1 输出发生从 0 到 1 的变化 Bit18 为 1 表示比较器 1 输出发生从 1 到 0 的变化 二者都为写 1 清零
17：16	CMP0_IF	R/W	0	比较器 0 的中断标志 Bit17 为 1 表示比较器 0 输出发生从 0 到 1 的变化 Bit16 为 1 表示比较器 0 输出发生从 1 到 0 的变化 二者都为写 1 清零
15:11	RESERVED	R	0	保留位
10：8	CMP_ORG_OUT	R	0	比较器结果输出（滤波前） 1：P 端>N 端 0：P 端<N 端 Bit10~bit8 分别代表 CMP2~CMP0 的输出
7:3	RESERVED	R	0	保留位
2:0	CMP_OUT	R	0	比较器结果输出（滤波后） 1：P 端>N 端 0：P 端<N 端 Bit2~bit0 分别代表 CMP2~CMP0 的输出

5.21 运算放大器（OPAMP）

5.21.1 概述

本芯片提供了两个独立的运算放大器单元，每个放大器都可独立配置使用。每个运算放大器的输出端在芯片内部已连接至特定的 ADC 输入通道，可实现 ADC 的直接采样测量。使用特定 IO 端口作为输入端，具体端口如本模块的结构框图所示。

5.21.2 特性

- 轨到轨的输入/输出
- 正端/负端输入
- 2 路运算放大器
- 3MHz bandwidth
- 低偏移电压
- 输出信号可以选择配置到 ADC 通道直接采集

5.21.3 模块结构图

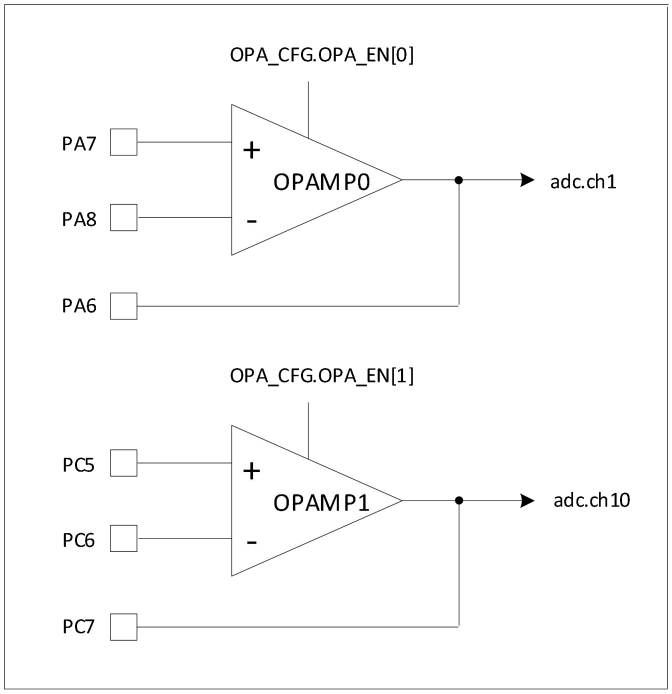


图 5-166 运算放大器结构框图

如上图所示：

OPAMP0 的 VP 端由 PA7 引脚输入，VN 端由 PA8 引脚输入，输出端可直接输出给 PA6 引脚或输出给 SARADC 的通道 1；并且该运算放大器可由其独立使能寄存器 OPA_CFG.OPA_EN[0]直接编程。

OPAMP0 的 VP 端由 PC5 引脚输入，VN 端由 PC6 引脚输入，输出端可直接输出给 PC7 引脚或输出给 SARADC 的通道 10；并且该运算放大器可由其独立使能寄存器 OPA_CFG.OPA_EN[1]直接编程。

寄存器映射

名称	偏移量	类型	复位值	描述
OPERATIONAL AMPLIFIER	BASE: 0x40000000			

OPA_CFG	0x140	R/W	0x00	运算放大器配置寄存器

寄存器描述

OPA_CFG

位域	名称	类型	复位值	描述
31:2	RESERVED	R	0	保留位
1:0	OPA_EN	R	0	OPA 使能控制位 1：使能 0：关闭 Bit1-bit0 分别控制 OPA1-OPA0

5.22 FLASH 控制器（FLASHCTRL）

5.22.1 概述

本芯片内部具有一个 64K 字节的嵌入式 FLASH 存储器，用于存储应用程序和芯片性能调校相关的参数数据。FLASH 具有两块区域：一块为主阵列区，用于存储应用程序或数据；另一块为 NVR 区，用于存储本芯片性能调校相关的参数数据等。

用于控制内部 FLASH 操作的控制器电路位于 CPU 和 FLASH 之间，通过 CPU 的读、写、擦操作实现对 FLASH 的读操作、编程操作以及擦操作等。

5.22.2 特性

- 支持 64K 字节应用程序存储空间
- 支持进入和退出 FLASH 低功耗模式
- 读速率可配置。当系统时钟频率小于或等于 56MHz 时，读速率为 1 个系统时钟周期等待；当系统时钟频率大于 56MHz 小于或等于 84MHz 时，读速率为 2 个系统时钟周期等待
- 配置区（NVR 区）域共 4 个扇区，每个扇区 512 字节，第 1 个扇区存放的是 FLASH 自身的配置信息，用户不能进行操作。其它 3 个扇区用户可以进行擦、读或写操作
- 操作模式支持读操作、编程操作以及扇区擦操作等
- 可以配置操作锁，禁止对 FLASH 进行编程和擦除操作，保护 FLASH 数据不被修改
- 支持地址屏蔽功能，屏蔽地址可以配置为 2K 字节、4K 字节或 8K 字节
- 每个扇区大小是 512 字节，支持扇区擦除操作
- 支持连续编程，在同一扇区内最多 64 个字（256 字节）连续编程

5.22.3 模块结构框图

FLASH 控制器包括 AHB 从接口、APB 从接口、FLASH 控制寄存器、FLASH 初始化控制器、FLASH 操作控制器和片上 FLASH 构成。

FLASH 存储控制器结构示意框图如下图所示：

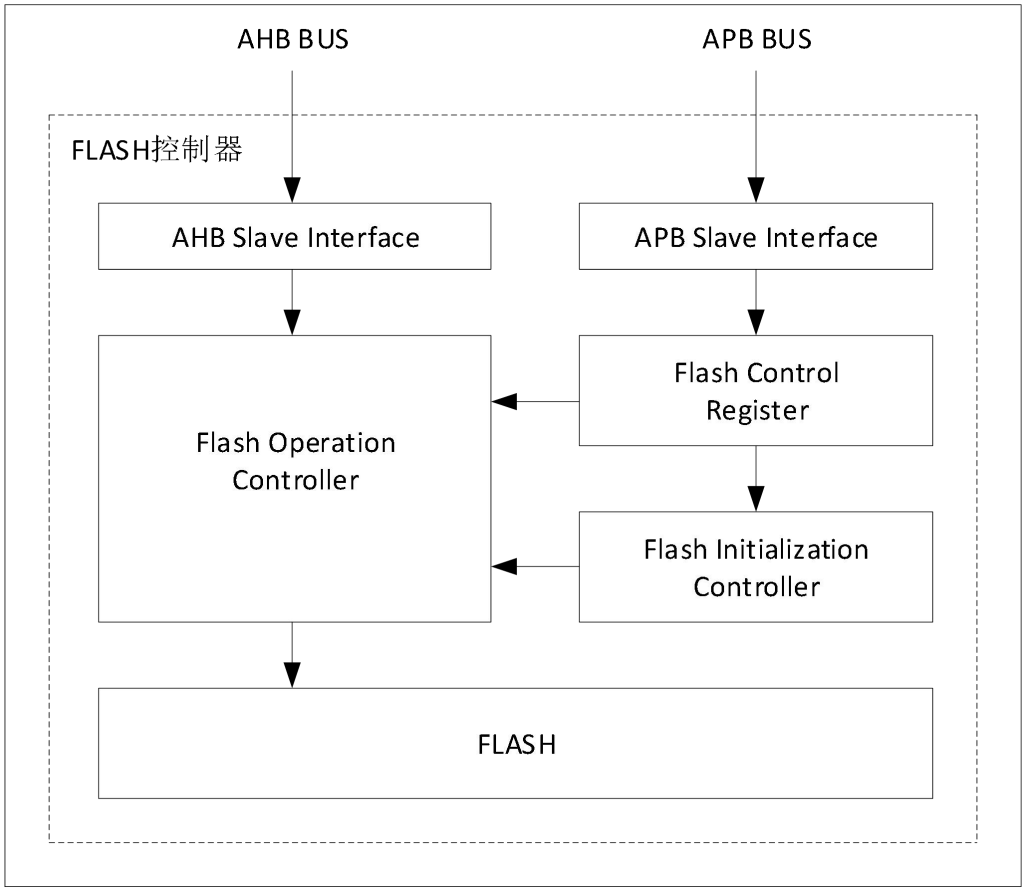


图 5-167 FLASH 控制器结构图

AHB 从接口：用于 CPU 的指令和数据读取。

APB 从接口：用于 CPU 对 FLASH 控制器相关寄存器进行编程配置。

FLASH 控制寄存器：用于实现 FLASH 控制器模块中所需要的各功能寄存器逻辑和各操作模式下相关状态产生逻辑。

FLASH 初始化控制器：当芯片上电或复位，FLASH 初始化控制器电路将开始自动访问 FLASH，并且读取 FLASH 参数数据，装载用户配置信息到 FLASH 寄存器中，完成 FLASH 初始化。FLASH 完成初始化后，可以开始执行正常操作。

FLASH 操作控制：根据 CPU 通过总线操作 FLASH 的功能要求，实现对 FLASH 进行相关操作，包括 FLASH 擦除、FLASH 编程和 FLASH 读，并产生相应的状态信号反馈给控制寄存器逻辑。

片上 FLASH 存储器：片上 FLASH 存储器主要是用于存储用户程序和参数的，它包括了 2K 字节的配置区和 64K 字节的程序区，每个扇区大小是 512 字节，共 128 个扇区，最小可编程数据位大小是 32 位。

5.22.4 功能描述

低功耗模式

FLASH 支持进入低功耗模式，进入低功耗模式后，可以大幅度的降低芯片的工作电流。进入低功耗的方式是在 RAM 中配置 FLASH_CFG 寄存器中的 DEEP_PD 位置 1 即可。退出低功耗的方式是在 RAM 中配置 FLASH_CFG 寄存器中的 DEEP_PD 位置 0 即可。

读速率配置

系统时钟配置为不同的频率时，需要匹配相应的读速率，否则 FLASH 读取会出问题。当系统时钟频率小于或等于 56MHz 时，只需要 1 个系统时钟等待即可，也就是在 RAM 中配置 FLASH_CFG 寄存器中的 READ_MD 位置 0 即可；当系统时钟频率大于 56MHz 并且小于或等于 84MHz 时，需要 2 个系统时钟等待，也就是在 RAM 中配置 FLASH_CFG 寄存器中的 READ_MD 位置 1 即可。

NVR 区和 MAIN 区选择

芯片的电压和时钟等 TRIM 值保存在 NVR 区中，初始化时程序需要从 NVR 区内将 TRIM 值读取并写入相应的寄存器中。

NVR 区的访问方式是在 RAM 中配置 FLASH_CFG 寄存器中的 NVR_SEL 位置 1 即可。NVR 区域共 4 个扇区，每个扇区 512 字节，第 1 个扇区存放的是 FLASH 的配置信息，用户不能进行操作，后面的 3 个扇区用户可以进行擦、读和写操作。

操作模式

FLASH 的操作模式支持以下三种方式：读操作、编程操作、扇区擦操作。配置操作模式时必须在 RAM 中执行。

编程操作：将待编程数据编程进以 FLASH_ADDR 中地址为起始的地址中，并且每次最多编程半个扇区(256 字节)，且只能在半个扇区内进行编程。

扇区擦操作：每个扇区为 512 字节，对应擦除扇区的地址由 FLASH_ADDR 来指定。每次编程或扇区擦操作完成后，必须将操作模式配置回到读操作。

读操作：通过 CPU 直接访问 FLASH 地址的方式即可实现 FLASH 的读操作。

FLASH 操作地址

FLASH_ADDR 寄存器：编程/扇区擦操作的地址寄存器。

启动编程操作时的起始地址，以字为单位。每次连续编程的最大编程数据为半个扇区，即最多编程数据为 64 个字，并且只能在半个扇区内进行编程。

启动扇区擦操作时，bit7-bit13 表示执行擦的扇区号；也就是 1 个扇区 512Byte，地址进行 512 字节对齐。

扇区大小

本芯片 FLASH 每个扇区是 512Byte，主程序区（MAIN 区）一共是 64KBytes，也就是 128 个扇区，配置区（NVR 区）是 2KBytes，也就是 4 个扇区。芯片支持扇区擦除，编程操作时最多只能编程半个扇区(256 字节)，并且只能在半个扇区内进行编程。

START 操作启动控制

当 FLASH_START 寄存器中的 START 位置 1 时，FLASH 将会启动 MODE 寄存器中所配置的命令操作（读、扇区擦、编程），当前命令操作完成后该位自动清零。

操作锁

向 FLASH_LOCK 寄存器写入 0x55，则操作启动被锁定，START 不能写 1。用于保护对 FLASH 的误操作，上电默认为锁定状态。

向 FLASH_UNLOCK 寄存器写入 0xAA，则操作启动被解锁，START 可以被写 1。

MASK 功能

MASK 地址可以配置为无屏蔽、2KBytes、4KBytes 和 8KBytes。

可以配置 MASK 选择锁定，当配置 MASK_LOCK 位配置为 1 时，MASK_SEL 不可更改。当该位配置为 0 时，MASK_SEL 才可以修改。

擦除时间和编程时间配置

根据系统时钟频率的不同，需要对擦除时间和编程时间进行配置使其满足 FLASH 自身时序的相应要求。

具体配置可以参看相应寄存器描述。

FLASH 初始化忙标志

当 FLASH_ST 寄存器的 INIT_BUSY 位为 1 时表示正在进行 flash 初始化过程中，为 0 时表示初始化完成。

当上电、复位和退出低功耗模式后，程序需要判断该位，只有 FLASH 初始化完成后，才能进行正常操作。

控制器忙标志

当 FLASH_ST 寄存器的 BUSY 位为 1 时表示控制器正在进行当前命令操作过程中，为 0 时表示当前操作已完成，控制器处于 READY 状态，等待新命令操作执行。

编程数据缓存寄存器空状态标志

当 FLASH_ST 寄存器的 PROG_BUF_EMPTY 位为 1 时表示编程数据缓存寄存器为空，可以写入下一个字数据；为 0 时表示编程数据缓存寄存器非空，不能写入下一个字数据。在当前字编程过程中（16us）写入下一个字数据则会连续编程，否则终止连续编程操作。

FLASH 程序初始化流程

配置 FLASH 进入正常工作模式，配置为读模式，根据系统时钟配置读速率、擦除时间、编程时间参数，FLASH 上锁，禁止对 FLASH 编程操作。

这些操作必须在 RAM 中执行。

FLASH 扇区擦操作流程

1） 查询 FLASH 控制器忙标志，等待控制器处于 READY 状态；

2） 配置模式为扇区擦操作；

3） 写入扇区擦起始地址，以字为单位；

4） 配置 FLASH 解锁，启动 START 命令；

5） 查询 FLASH 控制器忙标志，等待控制器处于 READY 状态；

6） 配置模式为读操作，FLASH 上锁，完成扇区擦操作。

扇区擦操作流程示意图如下所示：

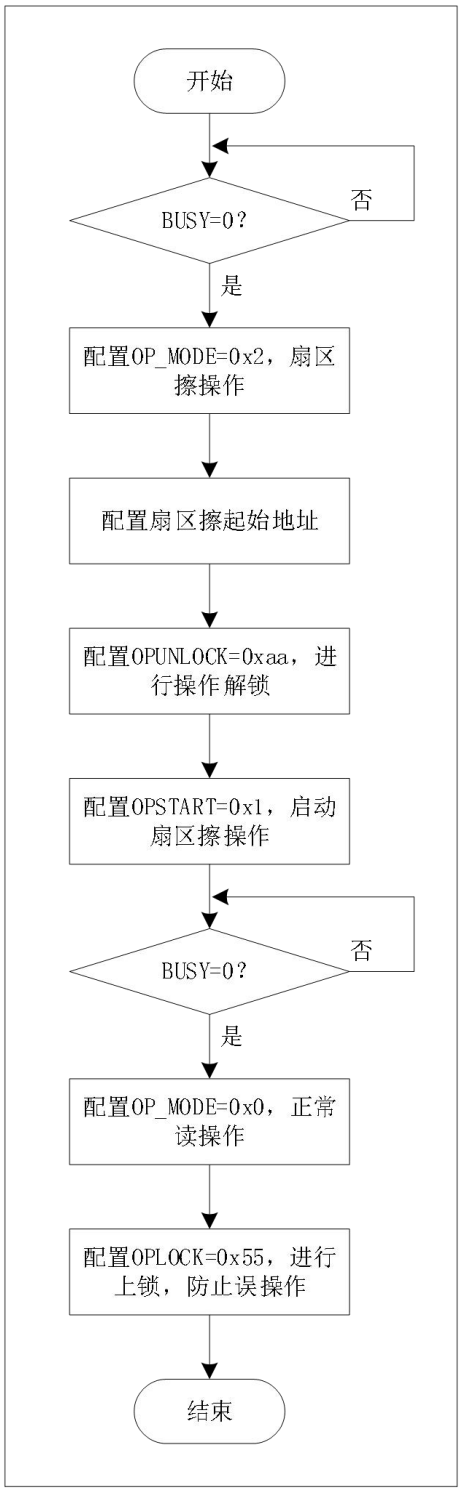


图 5-168 扇区擦操作流程图

这些操作必须在 RAM 中执行。

FLASH 单字编程操作流程

1） 查询 FLASH 控制器忙标志，等待控制器处于 READY 状态；

2） 配置模式为编程操作；

3） 写入编程地址，以字为单位；

4） 将待编程数据放到数据寄存器中；

5） 配置 FLASH 解锁，启动 START 命令；

6） 查询 FLASH 控制器忙标志，等待控制器处于 READY 状态；

7） 配置模式为读操作，FLASH 上锁，完成对该字数据的编程操作。

编程操作流程示意图如下所示：

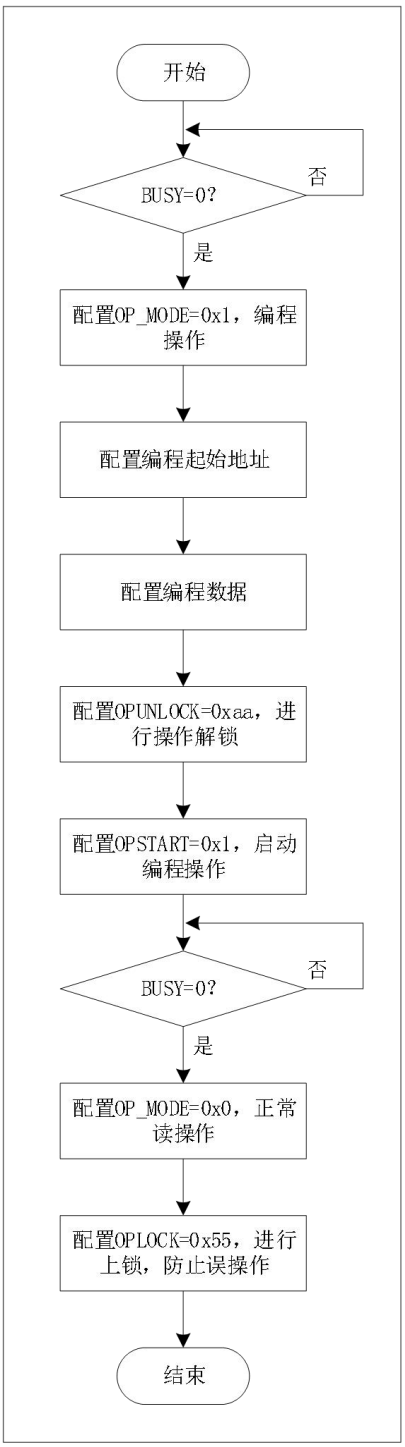


图 5-169 单字编程操作流程图

这些操作必须在 RAM 中执行。

FLASH 多字连续编程操作流程

1） 查询 FLASH 控制器忙标志，等待控制器处于 READY 状态；

2） 配置模式为编程操作；

3） 写入编程地址，以字为单位；

4） 将编程数据的第一个字写入数据寄存器中；

5） 配置 FLASH 解锁，启动 START 命令；

6） 若还有待编程数据，则判断 PROG_BUF_EMPTY 位是否为 0，为 0 时则可以写入下一个待编程的字（为 1 时等待，不能写入），直到所有待编程数据全部写入完成；

7） 查询 FLASH 控制器忙标志，等待控制器处于 READY 状态；

8） 配置模式为读操作，FLASH 上锁，完成对本次所有待编程数据操作，其中多字连续编程数据量最大为 64 个字，并且只能在半个扇区内进行编程。

编程操作流程示意图如下所示：

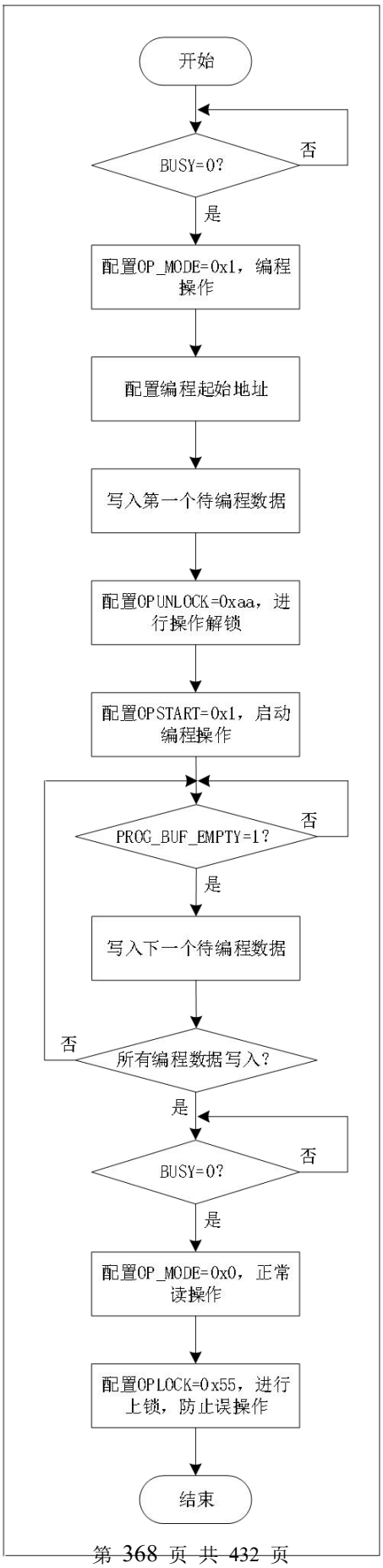

开始
BUSY=0?
否
是
配置OP_MODE=0x1，编程操作
配置编程起始地址
写入第一个待编程数据
配置OPUNLOCK=0xaa，进行操作解锁
配置OPSTART=0x1，启动编程操作
PROG_BUF_EMPTY=1?
否
是
写入下一个待编程数据
所有编程数据写入?
否
是
BUSY=0?
否
是
配置OP_MODE=0x0，正常读操作
配置OPLOCK=0x55，进行上锁，防止误操作
结束

图 5-170 多字连续编程操作流程图

这些操作必须在 RAM 中执行。

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
FLASH _CTRL	BASE: 0x4006F000				
FLASH_CFG	0x00	32	R/W	0x80000000	配置寄存器
FLASH_ADDR	0x04	32	R/W	0x0	地址配置寄存器
FLASH_WDATA	0x08	32	W	0x0	写数据寄存器
FLASH_START	0x10	32	R/W	0x0	操作启动寄存器
FLASH_ST	0x14	32	R	0x5	状态寄存器
FLASH_LOCK	0x18	32	W	0x0	操作锁定控制寄存器
FLASH_UNLOCK	0x1c	32	W	0x0	操作解锁控制寄存器
FLASH_MASK	0x20	32	R/W	0x2	屏蔽控制寄存器
FLASH_ERASETIME	0x24	32	R/W	0x7532a31b	擦除时间参数配置寄存器
FLASH_PROGTIME	0x28	32	R/W	0x1f4360	编程时间参数配置寄存器

寄存器描述

FLASH_CFG 寄存器（0x00）

位域	名称	类型	复位值	描述
31	DEEP_PD	R/W	1	配置 FLASH 进入低功耗模式 1：进入低功耗模式 0：正常工作模式 注：该操作只能在 RAM 中执行。

30:5	RESERVED	R	0	保留位
4:2	MODE	R/W	0	操作模式配置寄存器 000：正常读操作。用于 AHB 读时的操作模式。 001：编程操作。将待编程数据编程进以 FLASH_ADDR 中地址为起始的地址中。并且每次最多编程半个扇区(256 字节)，并且只能在半个扇区内进行编程。 010：扇区擦操作（每个扇区为 512 字节）。对应擦除扇区的地址由 FLASH_ADDR 来指定。 其它：保留 注 1：上述所有操作只能在 RAM 中执行。 注 2：每次其它操作模式（非正常读操作模式）完成后，MODE 一定要配置回 0。
1	NVR_SEL	R/W	0	NVR 区选择 0：选择 Main Array（共 128 个扇区，每个扇区 512 字节） 1：选择 NVR 扇区（共 4 个扇区，每个扇区 512 字节） 注：该寄存器改变必须在 RAM 中执行。
0	READ_MD	R/W	0x0	读速率模式选择 1：2 个系统时钟周期等待 （56MHz<sys_clk<84MHz） 0：1 个系统时钟周期等待 （sys_clk<=56MHz） 注：该寄存器改变必须在 RAM 中执行。

FLASH_ADDR 寄存器（0x04）

位域	名称	类型	复位值	描述
31:14	RESERVED	R	0	保留位
13:0	ADDR	R/W	0	编程/扇区擦操作的地址寄存器 启动编程操作时的起始地址，以字为单位。每次连续编程的最大编程数据为半个扇区，即最多编程数据为 64 个字，并且只能在半个扇区内进行编程； 启动扇区擦操作时，bit7-bit13 表示执行擦的扇区号；也就是 1 个扇区 512Byte

FLASH_WDATA 寄存器（0x08）

位域	名称	类型	复位值	描述
31:0	WDATA	R/W	0	编程操作的数据寄存器 启动编程操作时，将该寄存器中的数据编程写入对应的地址中

FLASH_START 寄存器（0x10）

位域	名称	类型	复位值	描述
31:1	RESERVED	R	0	保留位
0	START	R/W	0	操作启动控制位 将该位写 1，则启动 MODE 寄存器中所配置的命令操作。 当前命令操作完成后该位自动清零

FLASH_ST 寄存器（0x14）

位域	名称	类型	复位值	描述
31:3	RESERVED	R	0	保留位
2	PROG_BUF_EMPTY	R	1	编程数据缓存寄存器空状态标志 1：表示编程数据缓存寄存器为空。可以写入下一个字数据。 0：表示编程数据缓存寄存器非空，不能写入下一个字数据。 在当前字编程过程中写入下一个字数据则会连续编程，否则终止连续编程操作。

1	BUSY	R	0	控制器忙标志 1：表示控制器正在进行当前命令操作过程中。 0：表示控制器正处于 READY 状态，等待命令操作执行。
0	INIT_BUSY	R	1	FLASH 初始化忙标志 1：表示正在进行 flash 初始化过程中。 0：表示初始化完成

FLASH_LOCK 寄存器（0x18）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	LOCK	W	0x0	操作锁控制 向该寄存器写入 0x55，则操作启动被锁定，START 不能写 1。用于保护对 FLASH 的误操作。 上电默认锁定。

FLASH_UNLOCK 寄存器（0x1C）

位域	名称	类型	复位值	描述
31:8	RESERVED	R	0	保留位
7:0	UNLOCK	W	0	操作解锁控制 向该寄存器写入 0xAA，则操作启动被解锁，START 可以被写 1。

FLASH_MASK 寄存器（0x20）

位域	名称	类型	复位值	描述

31:3	RESERVED	R	0	保留位
2	MASK_LOCK	R/W	1	MASK 选择锁定控制 当该位配置为 1 时，MASK_SEL 不可更改。当该位配置为 0 时，MASK_SEL 才可以修改。
1:0	MASK_SEL	R/W	0	MASK 选择 00：无屏蔽 01：最低 2KB（即最低 4 个扇区）被屏蔽 10：最低 4KB（即最低 8 个扇区）被屏蔽 11：最低 8KB（即最低 16 个扇区）被屏蔽

FLASH_ERASETIME 寄存器（0x24）

位域	名称	类型	复位值	描述
31	RESERVED	R	0	保留位
30:19	TRCV	R/W	0xea6	FLASH 扇区擦时，TRCV 时间寄存器 根据系统时钟的频率，需要对 TRCV 进行适当配置，使得其时间至少大于 50us 的要求。 其中，扇区擦操作可在系统时钟最高为 72MHz 情况下工作 注 1：默认值为系统时钟为 48MHz 情况下，扇区擦操作的配置值。
18:0	TERASE	R/W	0x2a31b	FLASH 扇区擦时，TERASE 时间寄存器 根据系统时钟的频率，需要对 TERASE 进行适当配置，使得 TERASE 时间在 3.2ms-4ms 范围内。 其中，扇区擦操作可在系统时钟最高为 72MHz 情况下工作； 注 1：默认值为系统时钟为 48MHz 情况下，扇区擦操作的配置值。

FLASH_PROGTIME 寄存器（0x28）

位域	名称	类型	复位值	描述

31:22	RESERVED	R	0	保留位
21:11	TPGS	R/W	0x3e8	FLASH 编程操作时的 TPGS 时间控制寄存器。 根据系统时钟的频率，需要对 TPGS 进行适当配置，使得其时间至少大于 20us 的要求。 注：默认值为系统时钟在 48MHz 情况下
10:0	TPROG	R/W	0x360	FLASH 编程操作时的 TPROG 时间控制寄存器。 编程时，TPROG 时间需要在 16us-20us 范围； 根据系统时钟的频率，需要对 TPROG 进行适当配置，使得其在合适范围内。 注：默认值为系统时钟在 48MHz 情况下

5.23 循环冗余校验（CRC）

5.23.1 概述

本芯片 CRC 主要用于在数据通信过程中降低通信线路的误码率。可通过 AHB 总线接口将传输的数据经过本模块运算，通过循环冗余检验结果来确保数据传输的可靠性。

其系统框图如下：

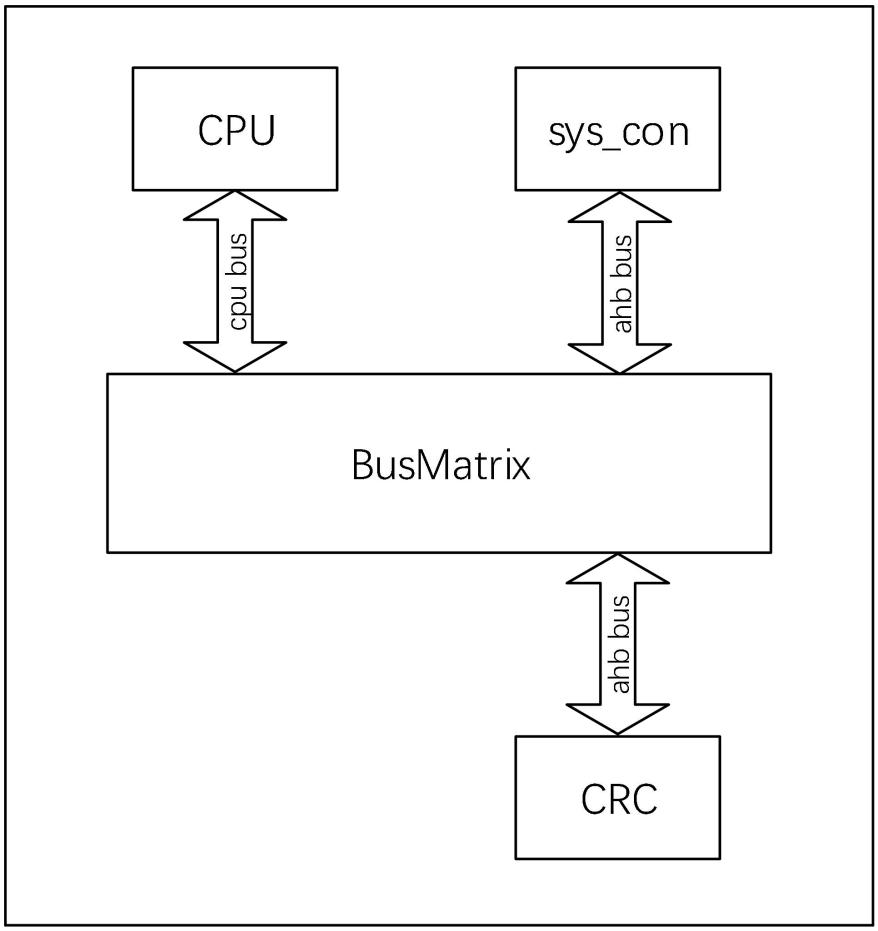


图 5-171 CRC 模块系统框图

5.23.2 特性

- 支持对 8、16、32 位数据进行 CRC 运算

- 支持输入数据、输出数据取反；
- 支持输入数据、输出数据翻转；
- 支持以下四种 CRC 多项式：

  x^16+x^12+x^5+1、

  x^8+x^2+x+1、

  x^16+x^15+x^2+1、

  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+1。

5.23.3 模块结构图

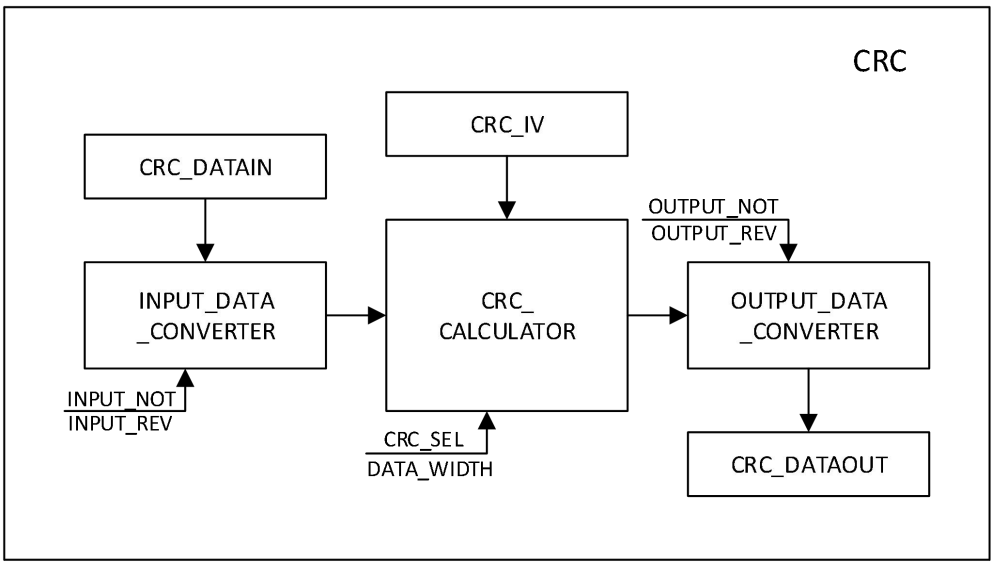


图 5-172 CRC 模块结构图

上图为 CRC 模块内部结构示意图。输入数据 CRC_DATAIN 经过取反和翻转之后进入 CRC 计算器，计数器根据 CRC 初始值、有效数据位设置和所选 CRC 多项式来进行 CRC 计算，计算结果在经过取反和翻转之后输出至 CRC_DATAOUT。

5.23.4 功能描述

CRC 码生成规则

CRC 码由两部分组成的，前部分是信息码，就是需要校验的信息，后部分是校验码。在发送端（存储器写）根据要传送的 k 位二进制码序列，以一定的规则产生一个校验用的 r 位监督码(CRC 码)，附在原始信息后边，构成一个新的二进制码序列数共 k+r 位，然后发送出去。在接收端（存储器读），根据信息码和 CRC 码之间所遵循的规则进行检验，以确定传送中是否出错。

如果 CRC 码 T 长共 n bit，信息码 D 长 k bit， 剩余的 r bit 的 F 即为校验位。如(7,3)码：110 1001，前三位 110 为信息码，1001 为校验码。如下所示，是一个 CRC 码的构成示意图。

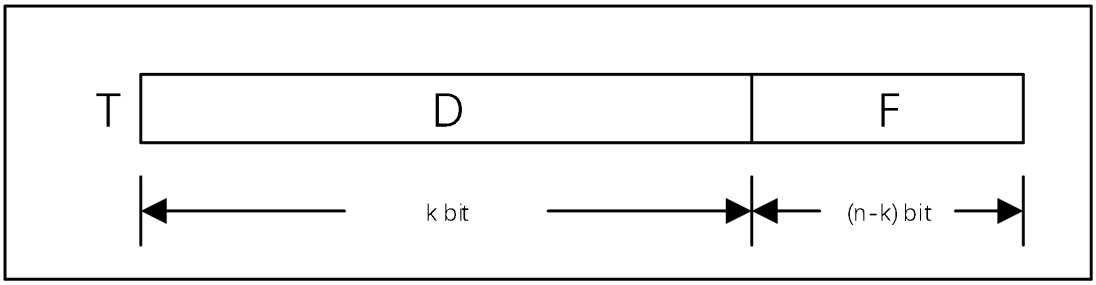


图 5-173 CRC 码构成示意图

CRC 校验码的生成规则可以概括为以下步骤：

1. 将原信息码左移 CRC 多项式位，右侧补零；如：二进制数据 110 进行 4 位 CRC 校验码生成 －> 110 0000；

2. 用 110 0000 除以 CRC 多项式 (注意，使用的是模 2 除法)，得到的余数即为 CRC 校验码；

3. 将校验码续接到信息码的尾部，形成 CRC 码。

CRC 多项式

CRC 模式	声明名称	CRC 校验码位数	对应多项式
CRC-CCITT	CRC_CCITT	16	x^16+x^12+x^5+1
CRC-8	CRC_8	8	x^8+x^2+x+1
CRC-16	CRC_16	16	x^16+x^15+x^2+1
CRC-32	CRC_32	32	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+1

操作流程

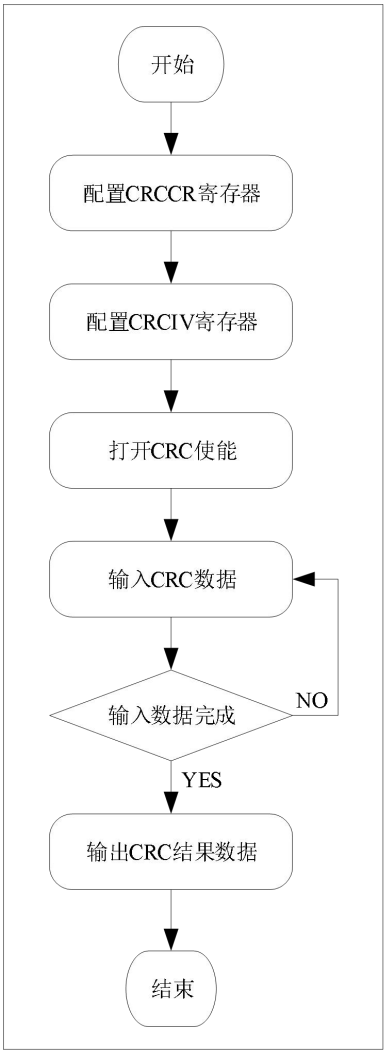


图 5-174 CRC 操作流程图

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
CRC	BASE: 0x40003000				
CRC_CR	0x00	32	R/W	0x00	CRC 控制寄存器
CRC_IV	0x04	32	W	0x00	CRC 初始值寄存器
CRC_DATAIN	0x08	32	R/W	0x00	CRC 输入数据寄存器
CRC_DATAOUT	0x0c	32	R	0x00	CRC 输出数据寄存器

寄存器描述

CRC_CR 寄存器（0x00）

位域	名称	类型	复位值	描述
31:11	RESERVED	RO	0	保留位
10:9	CRC_SEL	R/W	0	CRC 算法选择寄存器 00： x^16+x^12+x^5+1 01： x^8+x^2+x+1 10： x^16+x^15+x^2+1 11： 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
8:7	DATA_WIDTH	R/W	0	CRC 输入数据有效位数寄存器 00：32 位输入数据有效 01：低 16 位输入数据有效 10：低 8 位输入数据有效 11：保留

6:5	OUTPUT_INV	R/W	0	输出数据翻转寄存器 翻转规则同 INPUT_INV
4	OUTPUT_REV	R/W	0	输出数据是否取反 1：输出数据取反 0：输出数据不取反
3:2	INPUT_INV	R/W	0	输入数据翻转寄存器 00：bit 顺序不变 01：bit 顺序完全翻转 32 位数据宽度 31:0 -> 0:31； 16 位数据宽度 15:0 -> 0:15； 8 位数据宽度 7:0 -> 0:7 10：bit 顺序在字节范围内翻转 32 位数据宽度 31:0 -> 24:31, 16:23, 8:15, 0:7； 16 位数据宽度 15:0 -> 8:15, 0:7； 8 位数据宽度同 01 11：仅字节顺序翻转 32 位数据宽度 31:0 -> 7:0,15:8,23:16,31:24； 16 位数据宽度 15:0 -> 7:0,15:8； 8 位数据宽度同 00
1	INPUT_REV	R/W	0	输入数据是否取反 1：输入数据取反 0：输入数据不取反
0	CRC_EN	R/W	0	CRC 使能位 1：CRC 使能 0：CRC 不使能

CRC_IV 寄存器（0x04）

位域	名称	类型	复位值	描述
31:0	CRC_IV	R/W	0	CRC 初始值寄存器

CRC_DATAIN 寄存器（0x08）

位域	名称	类型	复位值	描述
31:0	CRC_DATAIN	W	0	CRC 输入数据寄存器 注：当 DATA_WIDTH 为 00：32 位有效 01：低 16 位有效 10：低 8 位有效 11：保留

CRC_DATAOUT 寄存器（0x0C）

位域	名称	类型	复位值	描述
31:0	CRC_DATAOUT	W	0	CRC 输出数据寄存器 注：当 CRC_SEL 为 00：低 16 位有效 01：低 8 位有效 10：低 16 位有效 11：32 位有效

5.24 DMA 控制器（DMA）

5.24.1 概述

该模块为 DMA 控制器，其位于两个外设之间，主要用于实现系统存储设备与外设之间的数据搬运。其系统连接示意图如下所示：

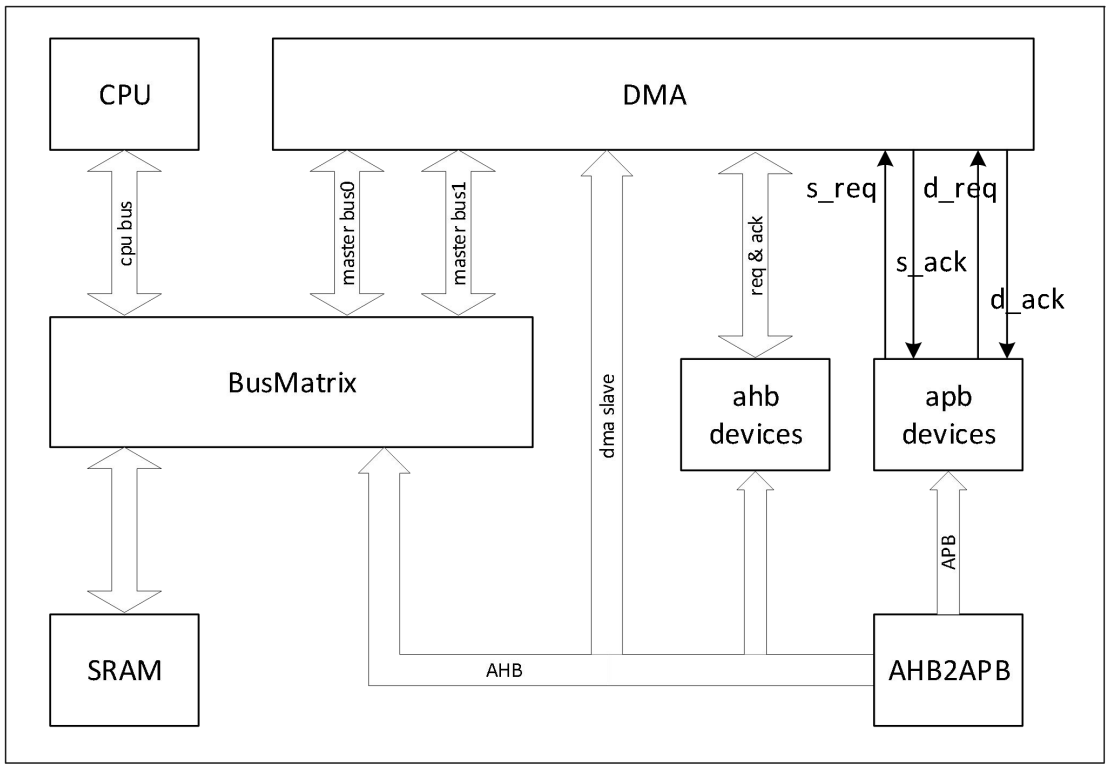


图 5-175 DMA 控制器系统连接示意图

5.24.2 特性

- 本芯片 DMA 通道数为 4。
- 具有两个 AHB 主控制端接口：一个用于从源地址设备读取数据，另一个用于向目的地址设备写入数据；

- 源地址设备和目的地址设备可分别配置传输数据宽度 8、16 和 32bits，源地址和目的地址必须按照数据传输宽度对齐；
- 每个通道最多可配置传输 4096 笔数据。各通道每传输完一笔数据，则该通道释放总线，电路将重新判断各通道优先级，优先级高的通道先传输。
- 支持通道可配置为递增或固定地址模式。
- 支持存储器到存储器、存储器到外设、外设到存储器之间三种方式
- 支持循环重启 DMA
- 支持每个通道优先级可配置
- 支持每个通道独立的中断

5.24.3 模块结构示意图

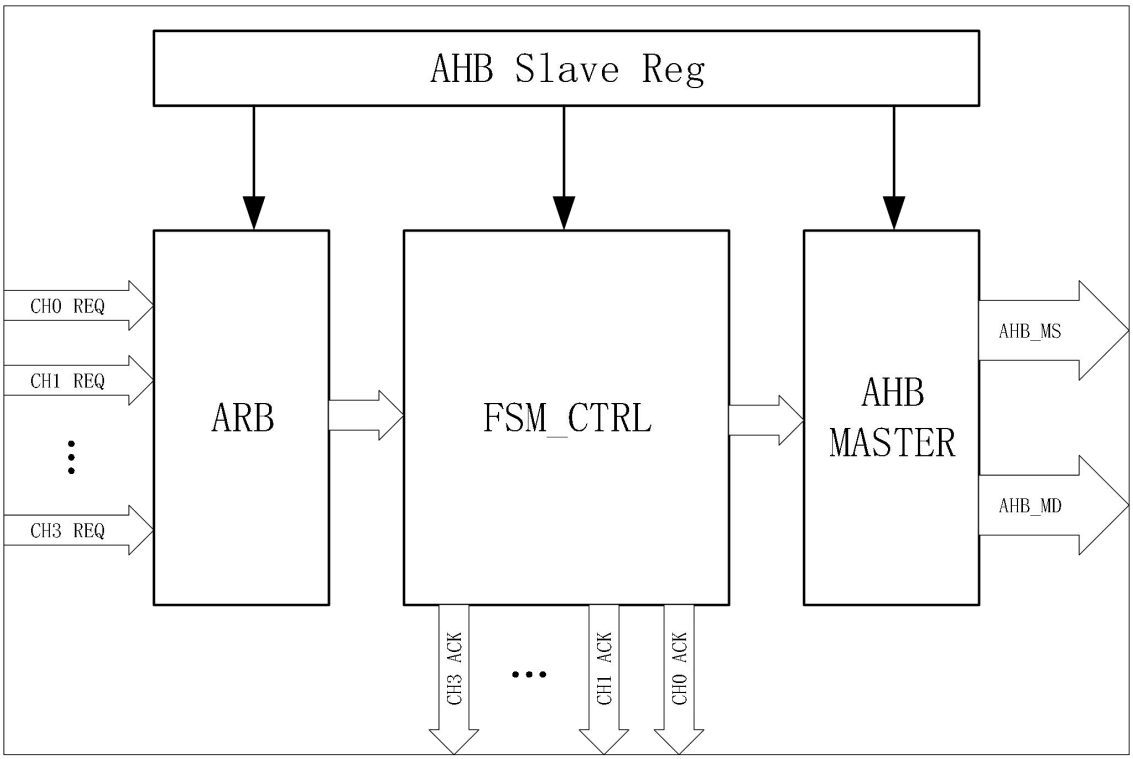


图 5-176 DMA 模块结构示意图

5.24.4 功能描述

DMA 控制器与 CPU 内核共享系统总线矩阵，可以直接执行存储器与存储器、存储器与外设、外设与存储器之间的数据传输。当 CPU 和 DMA 同时访问相同的目标（RAM 或外设）时，DMA 请求会暂停 CPU 访问总线达若干个周期，总线仲裁器将执行仲裁操作，以保证 CPU 至少可以得到一半的系统总线带宽。

DMA 处理流程

存储器（RAM）到存储器（RAM）

若期望 DMA 使用 CHn 通道将数据从 RAM 的一个空间搬运到 RAM 的另一个空间。则配置该通道的源侧将 CHnMOD.MS_SEL 寄存器配置为 0，目的侧将 CHnMOD.MD_SEL 寄存器也配置为 0，并且源地址和目的地址都配置为 RAM 的地址空间，该通道的其它配置项目配置完毕后，将 CHnCON.SWREQ 配置为 1，则开始启动数据的搬运，每笔数据按照数配置的 SIZE 大小 DMA 通过源侧主控制器从 RAM 的源地址读出，经过 DMA 处理后，通过目的侧主控制器写入 RAM 的目的地址中。

存储器（RAM）到外设

若希望 DMA 使用 CHn 通道将数据从 RAM 搬运到外设。

则配置该通道的源侧将 CHnMOD.MS_SEL 寄存器配置为 0，目的侧将 CHnMOD.MD_SEL 寄存器配置为相应的外设，并且该通道的其它配置项目配置完毕后，将 CHnCON.SWREQ 配置为 1，则开始启动数据的搬运，每笔数据按照数配置的 SIZE 大小 DMA 通过源侧主控制器

从 RAM 的源地址读出并经过 DMA 处理，等待目的侧发过来请求信号后，将处理后的数据通过总线发送给外设，同时给外设发送本次外设请求所对应的应答信号，用于外设清除本次的有效请求，以便正常发起下次请求。

外设到存储器（RAM）

若希望 DMA 使用 CHn 通道将数据从外设搬运到 RAM。

则配置该通道的源侧将 CHnMOD.MS_SEL 寄存器配置为相应的外设，目的侧将 CHnMOD.MD_SEL 寄存器配置为 0，并且该通道的其它配置项目配置完毕后，使能该通道，等待源侧外设发过来的请求信号，收到请求信号后，则开始启动数据的搬运，每笔数据按照数配置的 SIZE 大小 DMA 通过源侧主控制器从外设的源地址读出，经过 DMA 处理后，通过目的侧主控制器写入 RAM 的目的地址中，同时给源侧外设返回本次请求的应答信号，用于源侧外设清除本次请求，以便正常发起下次请求。

DMA 仲裁器处理

每个通道都可以独立配置优先权等级，仲裁器根据各通道的请求优先级来启动存储器或外设的访问传输。

DMA 的各通道优先级通过 CHnCON.PRI 来配置，共有 4 个等级：

---低优先级

---中优先级

---高优先级

---最高优先级

如果在仲裁的同一时刻有两个或两个以上通道都发起请求，则本笔数据的传输先对优先级高的通道进行数据搬运，若优先级相同，则编号较低的通道比编号较高的通道有优先权。例如：通道 2 优先于通道 3。

在 DMA 正在进行某通道数据搬运期间，收到其它通道发来的请求，则搁置这些请求，直到该通道本笔数据搬运完毕，才会释放一次 DMA 的控制权。下次传输通过重新冲裁来选择哪个通道被传输。

DMA 通道配置信息

每个通道都有独立的各自通道相关配置寄存器和状态寄存器，都可以独立的在有固定地址的外设寄存器和存储器之间进行 DMA 数据传输。DMA 传输的数据量是可编程的，最大可支持 4096 笔数据的传输，每笔数据位大小也编程。具有数据传输次数寄存器可以知道当前已传输了多少次数据，在每次传输完成后该寄存器递增。

源地址

通过 CHnMSADDR 寄存器可以配置 DMA 传输的外设或存储器的源地址。发生数据传输时，这个地址将作为数据传输的源。

目的地址

通过 CHnMDADDR 寄存器可以配置 DMA 传输的外设或存储器的目的地址。发生数据传输时，这个地址将作为数据传输的目的。

通道优先级

通过 CHnCON.PRI 寄存器可配置当前通道的优先权等级，共有 4 个等级可选择。在每次仲裁周期对比各个通道的优先权等级，等级高的优先传输。

循环模式

通过 CHnCON.CIRMD 寄存器配置当前通道的循环模式。该寄存器配置为 0 时，则不开启循环模式，该通道配置的数据量全部传输完毕后，则会停止传输；该寄存器配置为 1 时，则开启循环模式，该通道配置的数据量全部传输完毕后，将自动的按照该通道的配置信息，DMA 将重新启动数据传输，直到该通道被关闭。

传输的数据量

通过 CHnCON.LENTH 寄存器可配置当前通道的传输数据量，最大可支持 4096 笔数据。通过 CHnMOD.MD_SIZE 寄存器配置目的侧总线写数据格式大小，通过 CHnMOD.MS_SIZE 寄存器配置源侧总线读数据格式大小，这两个寄存器可分别编程为 8bit、16bit 和 32bit。

通道使能

通过 CHnCON.CH_EN 寄存器可配置当前通道开启或关闭。当通道的相关配置信息写入相关配置寄存器后，将该寄存器 配置为 1，则配置信息生效，等待源侧的有效请求信号。

当该寄存器配置为 1 后，若循环模式未开启，当收到源侧的有效请求后，开始进行数据传输，当配置的数据量全部传输完毕后，硬件自动将 CHnCON.CH_EN 寄存器清除为 0，关闭该通道，此时即使再收到源侧有效请求也不会启动该通道。若希望再次启用该通道传输，则需软件将该寄存器再编程为 1。

当该寄存器配置为 1 后，如循环模式开启，当收到源侧的有效请求后，开始进行数据传输，当配置的数据量全部传输完毕后，传输数据量计数器重新开始计数，DMA 按照之前的配置信息再进行数据传输。若不再希望数据传输，则需软件编程将该寄存器 CHnCON.CH_EN 配置为 0，关闭该通道，则即使收到源侧请求也不会再进行数据传输。需要注意，当软件将 CHnCON.CH_EN 寄存器清除为 0 时，并不会立刻关闭该通道，硬件将在当前进行的数据量全部传输完毕后才会自动关闭该通道。

通道映射选择

每个通道都可以独立选择源侧和目的侧的所对应的存储器或外设。

源侧可通过 CHnMOD.MS_SEL 寄存器编程选择，目的侧可通过 CHnMOD.MD_SEL 寄存器编程选择。这两个寄存器都分别可以选择多种设备（存储器或外设），其中配置为 0x00 选择为存储器，通常为芯片内的 RAM 存储器。也可以用于无握手无需等待可访问的其他外设，配置为 0x01-0x07 时，选择为具有需要握手功能的外设。

通道地址变化方式

每个通道都可以独立选择源侧和目的侧的地址变化方式。有两种地址变化方式选择：地址递增和地址不变。

其中，源侧可通过 CHnMOD.MS_ADDMOD 寄存器编程选择，目的侧可通过 CHnMOD.MD_ADDMOD 寄存器编程选择。

源地址和目的地址数据格式说明

源地址和目的地址的数据格式可分别配置，对应关系见下表示例：

其中 F0/F1/F2/F3/F4/F5/F6/F7/F8/F9/FA/FB/FC/FD/FE/FF 都表示一个字节数据。

图 5-177 DMA 源地址和目的地址的数据格式配置表

源宽度	目的宽度	传输长度	源（地址/数据）	传输操作	目的（地址/数据）
8	8	4	0x0/F0 0x1/F1 0x2/F2 0x3/F3	1:在 0x0 读 F0[7:0],在 0x0 写 F0[7:0] 2:在 0x1 读 F1[7:0],在 0x1 写 F1[7:0] 3:在 0x2 读 F2[7:0],在 0x2 写 F2[7:0] 4:在 0x3 读 F3[7:0],在 0x3 写 F3[7:0]	0x0/F0 0x1/F1 0x2/F2 0x3/F3
8	16	4	0x0/F0 0x1/F1 0x2/F2 0x3/F3	1:在 0x0 读 F0[7:0],在 0x0 写 00F0[15:0] 2:在 0x1 读 F1[7:0],在 0x2 写 00F1[15:0] 3:在 0x2 读 F2[7:0],在 0x4 写 00F2[15:0] 4:在 0x3 读 F3[7:0],在 0x6 写 00F3[15:0]	0x0/00F0 0x2/00F1 0x4/00F2 0x6/00F3
8	32	4	0x0/F0 0x1/F1 0x2/F2 0x3/F3	1:在 0x0 读 F0[7:0],在 0x0 写 000000F0[31:0] 2:在 0x1 读 F1[7:0],在 0x4 写 000000F1[31:0] 3:在 0x2 读 F2[7:0],在 0x8 写 000000F2[31:0] 4:在 0x3 读 F3[7:0],在 0xC 写 000000F3[31:0]	0x0/000000F0 0x4/000000F1 0x8/000000F2 0xC/000000F3
16	8	4	0x0/F1F0 0x2/F3F2 0x4/F5F4 0x6/F7F6	1:在 0x0 读 F1F0[15:0],在 0x0 写 F0[7:0] 2:在 0x2 读 F3F2[15:0],在 0x1 写 F2[7:0] 3:在 0x4 读 F5F4[15:0],在 0x2 写 F4[7:0] 4:在 0x6 读 F7F6[15:0],在 0x3 写 F6[7:0]	0x0/F0 0x1/F2 0x2/F4 0x3/F6
16	16	4	0x0/F1F0 0x2/F3F2 0x4/F5F4 0x6/F7F6	1:在 0x0 读 F1F0[15:0],在 0x0 写 F1F0[15:0] 2:在 0x2 读 F3F2[15:0],在 0x2 写 F3F2[15:0] 3:在 0x4 读 F5F4[15:0],在 0x4 写 F5F4[15:0] 4:在 0x6 读 F7F6[15:0],在 0x6 写 F7F6[15:0]	0x0/F1F0 0x2/F3F2 0x4/F5F4 0x6/F7F6
16	32	4	0x0/F1F0 0x2/F3F2 0x4/F5F4 0x6/F7F6	1:在 0x0 读 F1F0[15:0],在 0x0 写 0000F1F0[31:0] 2:在 0x2 读 F3F2[15:0],在 0x4 写 0000F3F2[31:0] 3:在 0x4 读 F5F4[15:0],在 0x8 写 0000F5F4[31:0] 4:在 0x6 读 F7F6[15:0],在 0xC 写 0000F7F6[31:0]	0x0/0000F1F0 0x4/0000F3F2 0x8/0000F5F4 0xC/0000F7F6
32	8	4	0x0/ F3F2F1F0 0x4/ F7F6F5F4 0x8/FBFAF9F8 0xC/FFFEFDFC	1:在 0x0 读 F3F2F1F0[31:0],在 0x0 写 F0[7:0] 2:在 0x4 读 F7F6F5F4[31:0],在 0x1 写 F4[7:0] 3:在 0x8 读 FBFAF9F8 [31:0],在 0x2 写 F8[7:0] 4:在 0xc 读 FFFEFDFC [31:0],在 0x3 写 FC[7:0]	0x0/F0 0x1/F4 0x2/F8 0x3/FC

32	16	4	0x0/ F3F2F1F0 0x4/ F7F6F5F4 0x8/FBFAF9F8 0xC/FFFEFDFC	1:在 0x0 读 F3F2F1F0[31:0],在 0x0 写 F1F0[15:0] 2:在 0x4 读 F7F6F5F4[31:0],在 0x2 写 F5F4[15:0] 3:在 0x8 读 FBFAF9F8 [31:0],在 0x4 写 F9F8[15:0] 4:在 0xc 读 FFFEFDFC [31:0],在 0x6 写 FDFC[15:0]	0x0/F1F0 0x2/F5F4 0x4/F9F8 0x6/FDFC
32	32	4	0x0/ F3F2F1F0 0x4/ F7F6F5F4 0x8/FBFAF9F8 0xC/FFFEFDFC	1:在 0x0 读 F3F2F1F0[31:0],在 0x0 写 F3F2F1F0[31:0] 2:在 0x4 读 F7F6F5F4[31:0],在 0x4 写 F7F6F5F4[31:0] 3:在 0x8 读 FBFAF9F8 [31:0],在 0x8 写 FBFAF9F8 [31:0] 4:在 0xc 读 FFFEFDFC [31:0],在 0xC 写 FFFEFDFC [31:0]	0x0/ F3F2F1F0 0x4/ F7F6F5F4 0x8/FBFAF9F8 0xC/FFFEFDFC

中断说明

每个通道都可以在 DMA 传输过半和传输完成时产生中断。为了满足应用的灵活性和实用性考虑，可通过编程寄存器的不同位来打开这些中断。

图 5-178 DMA 中断说明表

中断事件	使能控制位	事件标志位
传输过半	HTC_INTEN	HTC_INTST
传输完成	TC_INTEN	TC_INTST

DMA 请求映射关系

本芯片 DMA 共有 4 个通道，通道 0-通道 3。支持从 UART0、UART1、UART2、SPI0、SPI1、ADC、TIMER_PLUS0 和 TIMER_PLUS1 这八个外设产生的请求。

每个通道对应的哪些外设请求可参见以下表及图所示相应关系。

各通道对应外设请求映射关系如下：

图 5-179 DMA 各通道对应外设请求映射关系

对应 bit	通道 0		通道 1	
	源侧	目的侧	源侧	目的侧
000	UART0_RX	UART2_TX	UART1_RX	UART0_TX
001	UART1_RX	UART0_TX	UART2_RX	UART1_TX
010	SPI0_RX	SPI1_TX	SPI1_RX	SPI0_TX
011	SARADC	N/A	SARADC	N/A
100	TIMER_PLUS0_L	N/A	TIMER_PLUS0_H	N/A
101	TIMER_PLUS1_L	N/A	TIMER_PLUS1_H	N/A
对应 bit	通道 2		通道 3	
	源侧	目的侧	源侧	目的侧
000	UART2_RX	UART1_TX	UART0_RX	UART2_TX
001	UART0_RX	UART2_TX	UART1_RX	UART0_TX
010	SPI1_RX	SPI0_TX	SPI0_RX	SPI1_TX
011	SARADC	N/A	SARADC	N/A
100	TIMER_PLUS1_L	N/A	TIMER_PLUS1_H	N/A
101	TIMER_PLUS0_L	N/A	TIMER_PLUS0_H	N/A

源侧外设请求映射关系图如下：

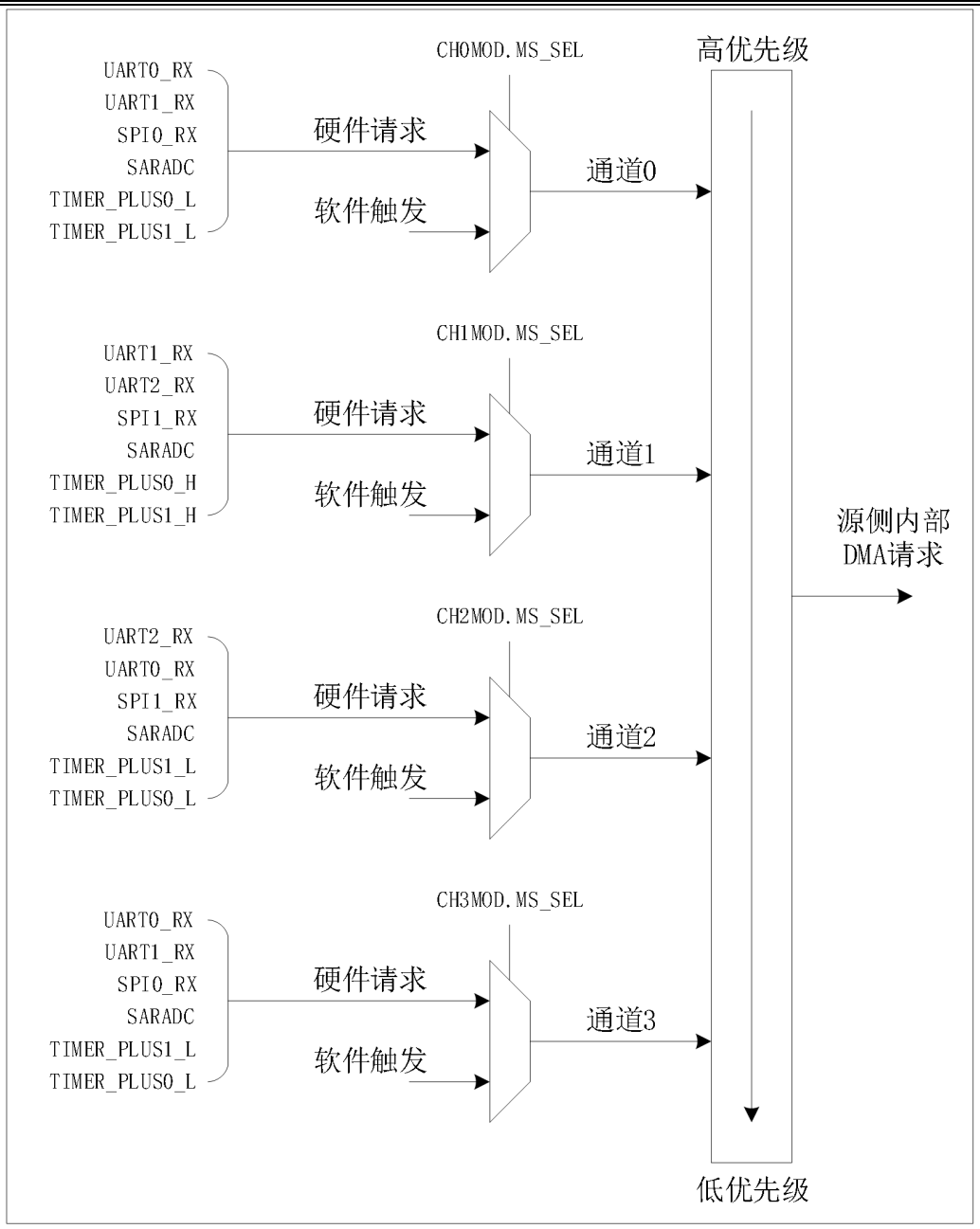


图 5-180 DMA 源侧外设请求映射关系图

目的侧外设请求映射关系图如下：

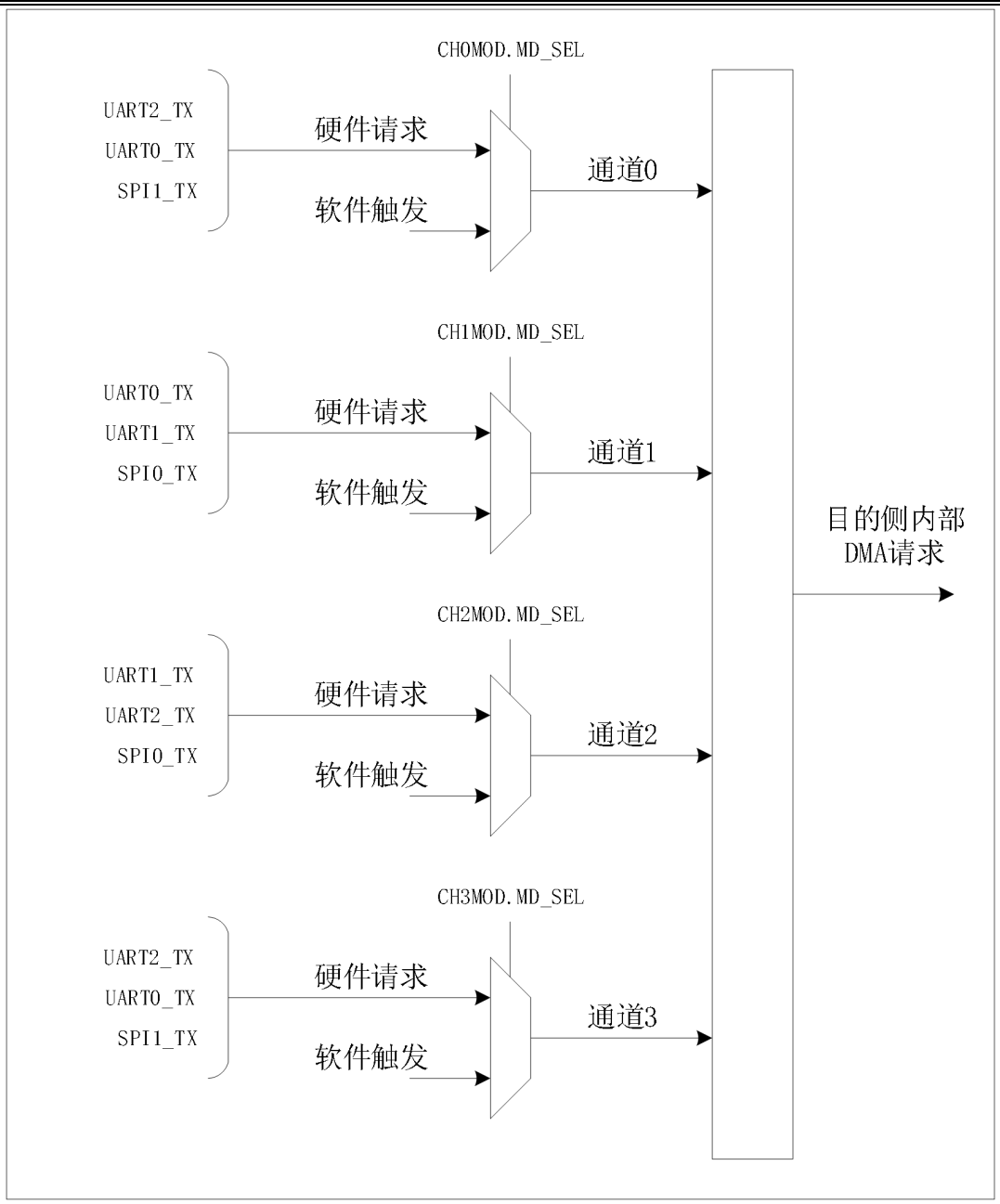


图 5-181 DMA 目的侧外设请求映射关系图

DMA 工作流程

DMA 的工作流程见图如下图所示：

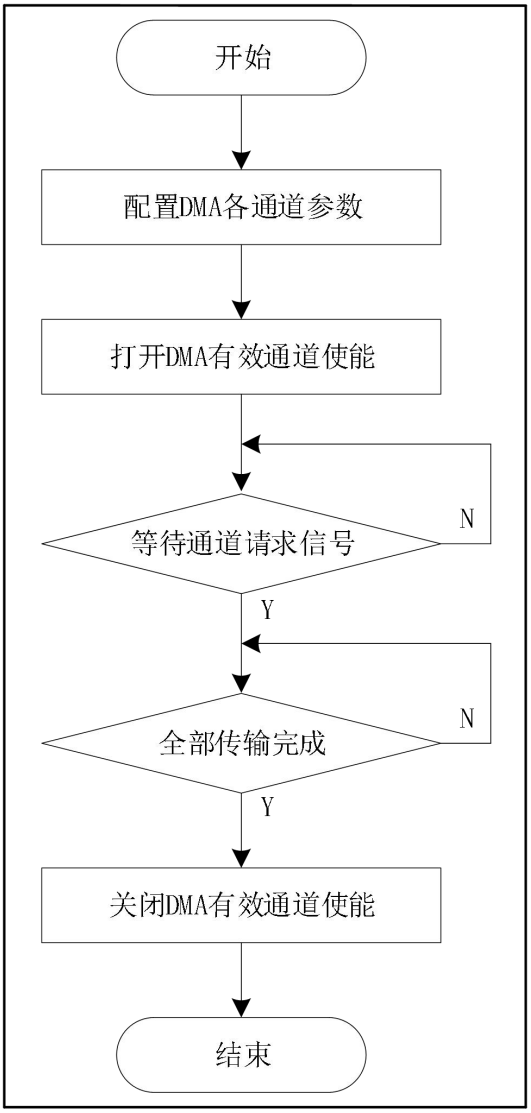


图 5-182 DMA 工作流程图

上图简要描述了 DMA 的工作过程。主要有以下一些操作：

1）将寄存器 DMA_CON 中的 DMA_EN 位配置为 1，将 DMA 打开；

2）打开各通道相应中断使能；

3）通过各通道的寄存器 DMA_CHnCON、DMA_CHnMOD、DMA_CHnMSADDR 和 DMA_CHnMADDR 将各自通道需要配置的参数信息编程进相应的寄存器位中；

4）配置相应通道 DMA_CHnCON.CH_EN 寄存器为 1，打开通道；

5）等待该通道源侧的请求信号。DMA 收到源侧请求信号后，通过总线从对应外设所设定的源地址读出数据，并进行保存处理；

6）从源侧读回所需数据后，等待目的侧的请求信号。当 DMA 收到目的侧请求信号后，通过总线向对应外设所设定的目的地址写入处理后的数据；

7）根据配置的传输数据量，当所有数据传输了一半或全部传输完毕后，会产生传输一半中断标志或传输完成中断标志；

8）当所有数据传输完毕后，将该通道 CH_EN 配置为 0，关闭该通道。此时可以修改该通道的配置寄存器值，等待下次数据传输使用。

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
DMA	BASE: 0x40001000				
DMA_CTR	0x00	32	R/W	0x00	DMA 控制寄存器
DMA_INTEN	0x04	32	R/W	0x00	DMA 中断使能寄存器
DMA_INTST	0x08	32	R/W	0x00	DMA 中断状态寄存器
DMA_CH0CTR	0x100	32	R/W	0x1ffe	通道 0 控制寄存器
DMA_CH0MOD	0x104	32	R/W	0x00	通道 0 模式寄存器
DMA_CH0MSADDR	0x108	32	R/W	0x00	通道 0 源地址寄存器
DMA_CH0MDADDR	0x10c	32	R/W	0x00	通道 0 目的地址寄存器
DMA_CH0_ST	0x110	32	R	0x00	通道 0 状态寄存器
DMA_CH1CTR	0x120	32	R/W	0x1ffe	通道 1 控制寄存器
DMA_CH1MOD	0x124	32	R/W	0x00	通道 1 模式寄存器
DMA_CH1MSADDR	0x128	32	R/W	0x00	通道 1 源地址寄存器
DMA_CH1MDADDR	0x12c	32	R/W	0x00	通道 1 目的地址寄存器

DMA_CH1_ST	0x130	32	R	0x00	通道 1 状态寄存器
DMA_CH2CTR	0x140	32	R/W	0x1ffe	通道 2 控制寄存器
DMA_CH2MOD	0x144	32	R/W	0x00	通道 2 模式寄存器
DMA_CH2MSADDR	0x148	32	R/W	0x00	通道 2 源地址寄存器
DMA_CH2MDADDR	0x14c	32	R/W	0x00	通道 2 目的地址寄存器
DMA_CH2_ST	0x150	32	R	0x00	通道 2 状态寄存器
DMA_CH3CTR	0x160	32	R/W	0x1ffe	通道 3 控制寄存器
DMA_CH3MOD	0x164	32	R/W	0x00	通道 3 模式寄存器
DMA_CH3MSADDR	0x168	32	R/W	0x00	通道 3 源地址寄存器
DMA_CH3MDADDR	0x16c	32	R/W	0x00	通道 3 目的地址寄存器
DMA_CH3_ST	0x170	32	R	0x00	通道 3 状态寄存器

寄存器描述

DMA_CTR 寄存器（0x00）

位域	名称	类型	复位值	描述
31: 1	RESERVED	R	0	保留位
0	DMA_EN	R/W	0	DMA 使能 0：DMA 关闭 1：DMA 使能

DMA_INTEN 寄存器（0x04）

位域	名称	类型	复位值	描述
31:12	RESERVED	R	0	保留位

11	CH3_THC_INTEN	R/W	0	通道 3 传输一半完成中断使能寄存器
10	CH2_THC_INTEN	R/W	0	通道 2 传输一半完成中断使能寄存器
9	CH1_THC_INTEN	R/W	0	通道 1 传输一半完成中断使能寄存器
8	CH0_THC_INTEN	R/W	0	通道 0 传输一半完成中断使能寄存器
7:4	RESERVED	R	0	保留位
3	CH3_TC_INTEN	R/W	0	通道 3 传输完成中断使能寄存器
2	CH2_TC_INTEN	R/W	0	通道 2 传输完成中断使能寄存器
1	CH1_TC_INTEN	R/W	0	通道 1 传输完成中断使能寄存器
0	CH0_TC_INTEN	R/W	0	通道 0 传输完成中断使能寄存器

DMA_INTST 寄存器（0x08）

位域	名称	类型	复位值	描述
31:12	RESERVED	R	0	保留位
11	CH3_THC_INTST	R/W	0	通道 3 传输一半完成中断状态寄存器 写 1 清零。
10	CH2_THC_INTST	R/W	0	通道 2 传输一半完成中断状态寄存器 写 1 清零。
9	CH1_THC_INTST	R/W	0	通道 1 传输一半完成中断状态寄存器 写 1 清零。
8	CH0_THC_INTST	R/W	0	通道 0 传输一半完成中断状态寄存器 写 1 清零。
7:4	RESERVED	R	0	保留位
3	CH3_TC_INTST	R/W	0	通道 3 传输完成中断状态寄存器 写 1 清零。
2	CH2_TC_INTST	R/W	0	通道 2 传输完成中断状态寄存器 写 1 清零。

1	CH1_TC_INTST	R/W	0	通道 1 传输完成中断状态寄存器 写 1 清零。
0	CH0_TC_INTST	R/W	0	通道 0 传输完成中断状态寄存器 写 1 清零。

DMA_CHnCTR 寄存器（0x100 + 0x20*(n)）

位域	名称	类型	复位值	描述
31:17	RESERVED	R	0	保留位
16	SWREQ	R/W	0	通过软件方式请求本通道开始传输 写 1 启动传输 注 1：仅用于源地址侧配置为存储设备时发起的请求。 注 2：LOOP 如果配置为 0 时，则完成一次传输，硬件自动清零；LOOP 如果配置为 1 时，则由软件控制将其清零。
15:14	PRI	R/W	0	通道优先级 00：低 01：中 10：高 11：最高 注：如果不同通道配置了相同的优先级，则通道号小的优先级更高。
13	LOOP	R/W	0	循环方式控制位 0：不执行循环方式。传输完成后停止 1：执行循环方式。传输完成后自动按照原有配置重新传输
12:1	LENTH	R/W	0xfff	传输计数寄存器，表示本通道传输的请求数。 实际的传输数为（LENTH+1）次。 每次传输的数据单位为：源地址侧数据单位由 MS_SIZE 决定；目的地址侧数据单位由 MD_SIZE 决定。

0	CH_EN	R/W	0	通道使能控制 0：通道关闭 1：通道有效 注 1：通道使能由软件写 1，启动传输。LOOP 如果配置为 0 时，则完成一次传输，硬件自动清零；LOOP 如果配置为 1 时，则由软件控制将其清零。 注 2：所有的配置需要在使能关闭的情况下进行。该位为 1 时，不能更改相关配置。

DMA_CHnMOD 寄存器（0x100 + 0x20*(n)+ 0x04）

位域	名称	类型	复位值	描述
31:14	RESERVED	R	0	保留位
13:11	MD_SEL	R/W	0	MD 侧外设选择 000：选择存储外设 001：选择产生 hsreq_md[0]所对应的外设 010：选择产生 hsreq_md[1]所对应的外设 011：选择产生 hsreq_md[2]所对应的外设 100：选择产生 hsreq_md[3]所对应的外设 101：选择产生 hsreq_md[4]所对应的外设 110：选择产生 hsreq_md[5]所对应的外设 111：选择产生 hsreq_md[6]所对应的外设 注 1：配置为 000 时，表示选择存储外设，通常表示芯片内的 SRAM。也可以用于无握手无等待可访问的其他外设。 注 2：配置为 001-111 时，不同芯片所连接外设不同，对应外设需要根据具体芯片实际连接外设所决定。

10:9	MD_SIZE	R/W	0	MD 侧的总线传输宽度 00：8bits 01：16bits 10：32bits 11：保留
8	MD_ADDMOD	R/W	0	MD 侧的地址变化方式选择 0：地址不变 1：地址递增
7:6	RESERVED	R	0	保留位
5:3	MS_SEL	R/W	0	MS 侧外设选择 000：选择存储外设 001：选择产生 hsreq_ms[0]所对应的外设 010：选择产生 hsreq_ms[1]所对应的外设 011：选择产生 hsreq_ms[2]所对应的外设 100：选择产生 hsreq_ms[3]所对应的外设 101：选择产生 hsreq_ms[4]所对应的外设 110：选择产生 hsreq_ms[5]所对应的外设 111：选择产生 hsreq_ms[6]所对应的外设 注 1：配置为 000 时，表示选择存储外设，通常表示芯片内的 SRAM。也可以用于无握手无等待可访问的其他外设。 注 2：配置为 001-111 时，不同芯片所连接外设不同，对应外设需要根据具体芯片实际连接外设所决定。
2:1	MS_SIZE	R/W	0	MS 侧的总线传输宽度 00：8bits 01：16bits 10：32bits 11：保留

0	MS_ADDMOD	R/W	0	MS 侧的地址变化方式选择 0：地址不变 1：地址递增

DMA_CHnMSADDR 寄存器（0x100 + 0x20*(n)+ 0x08）

位域	名称	类型	复位值	描述
31:0	MS_ADDR	R/W	0	MS 侧地址（源地址） 32 位操作时，字对齐；16 位操作时，半字对齐。8 位操作时，字节对齐。

DMA_CHnMDADDR 寄存器（0x100 + 0x20*(n) + 0x0C）

位域	名称	类型	复位值	描述
31:0	MD_ADDR	R/W	0	MD 侧地址（目的地址） 32 位操作时，字对齐；16 位操作时，半字对齐。8 位操作时，字节对齐。

DMA_CHn_ST 寄存器（0x100 + 0x20*(n) + 0x10）

位域	名称	类型	复位值	描述
31:12	RESERVED	R	0	保留位
11:0	CUR_LENTH	R	0	当前已传输个数

5.25 AES 运算单元（AES）

5.25.1 概述

本芯片具有一个 AES 模块，支持 128 位密钥及初始向量，可用于使用 AES 算法对数据进行加密和解密。

使用 128 位密钥长度对 128 位块进行加密和解密。 它还可以执行密钥派生、 加密或解密密钥存储在内部寄存器中，以便在使用同一密钥处理多个数据块时最大程度地减少 CPU 的写操作。

默认情况下，选择电子密码本模式（ECB）， 硬件还支持密码块链接（CBC）或计数器（CTR 模式）链接算法。

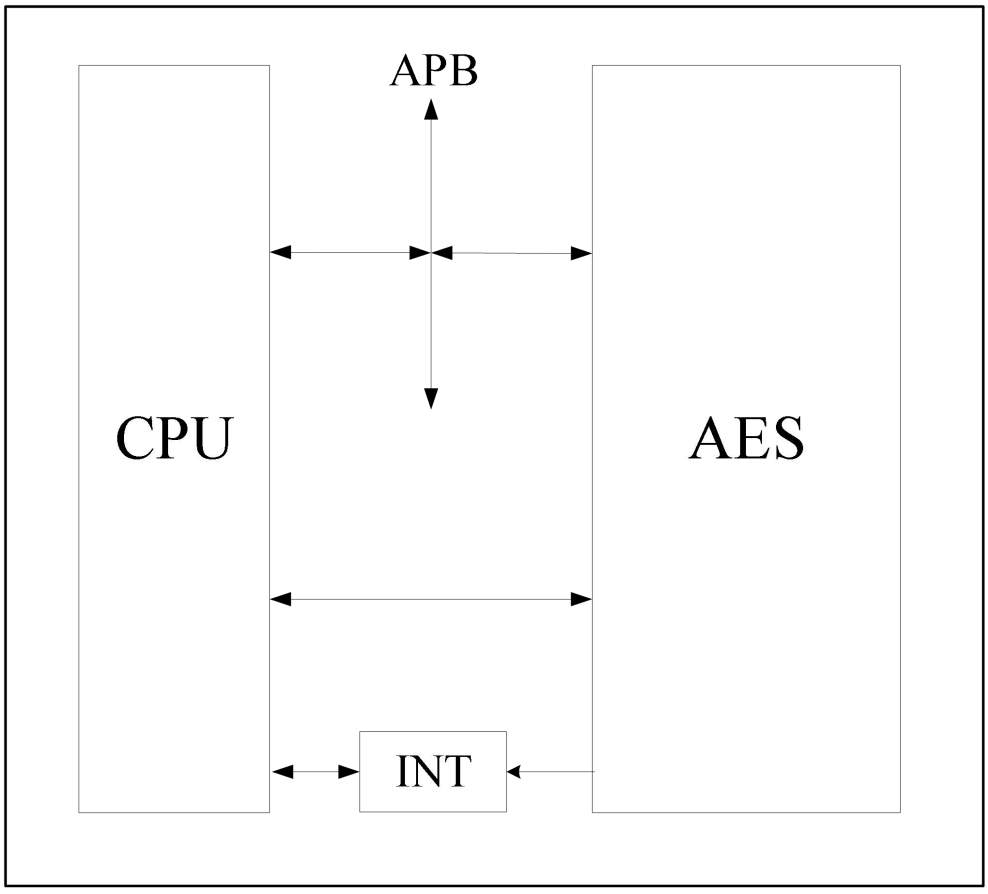


图 5-183 AES 系统框图

5.25.2 特性

- 使用 AES Rijndael 块密码算法进行加密/解密
- 内部 128 位寄存器，用于存储加密或派生密钥（4 个 32 位寄存器）
- 支持电子密码本（ECB），密码块链接（CBC）和计数器模式（CTR）
- 解密的派生密钥推导
- 128 位数据块处理
- 128 位密钥长度
- 213 个时钟周期来加密或解密一个 128 位块（包括输入和输出阶段）
- 1 个 32 位 INPUT 数据 buffer 和 1 个 32 位 OUTPUT 数据 buffer
- 仅支持 32 位数据宽度的寄存器访问
- 一个 128 位寄存器，用于在 CBC 模式下配置 AES 时用于初始化向量，或在选择 CTR 模式时用于 32 位计数器初始化

5.25.3 模块结构框图

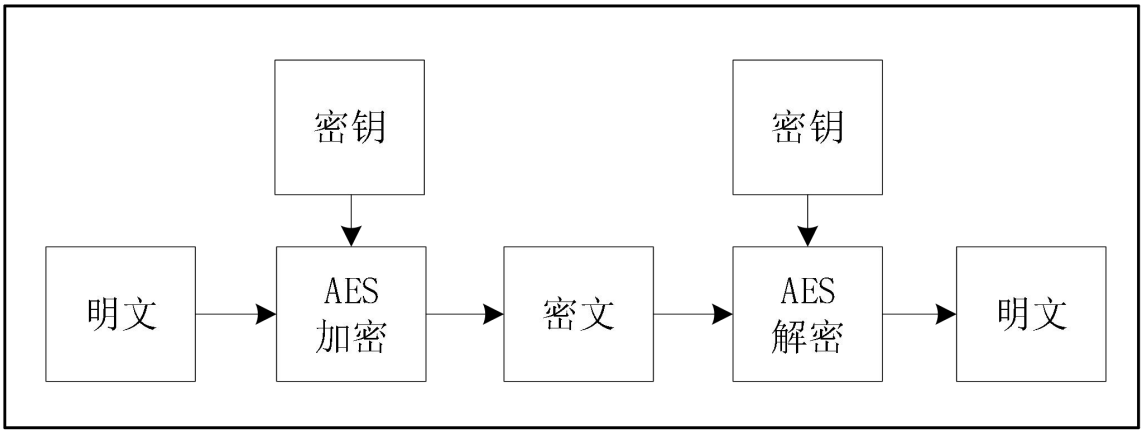


图 5-184 AES 模块结构框图

上图为 AES 模块的结构框图，图中只是简单的描述了加密和解密的过程，本模块中各个模式下不同算法的加密和解密过程以及原理图将在功能描述章节具体说明。

5.25.4 功能描述

AES 硬件支持三种算法，当禁用 AES（位 EN = 0）时，可以通过 AES_CR 寄存器中的 CHMOD [1：0]位进行选择：

•电子密码本（ECB）

•密码块链接（CBC）

•计数器模式（CTR）

电子密码本（ECB）

这是默认模式。 此模式下不使用 AES_IVR 寄存器。 没有链接操作。 数据分为多个块，每个块分别进行加密。

电子密码本算法的加密和解密的原理图如下：

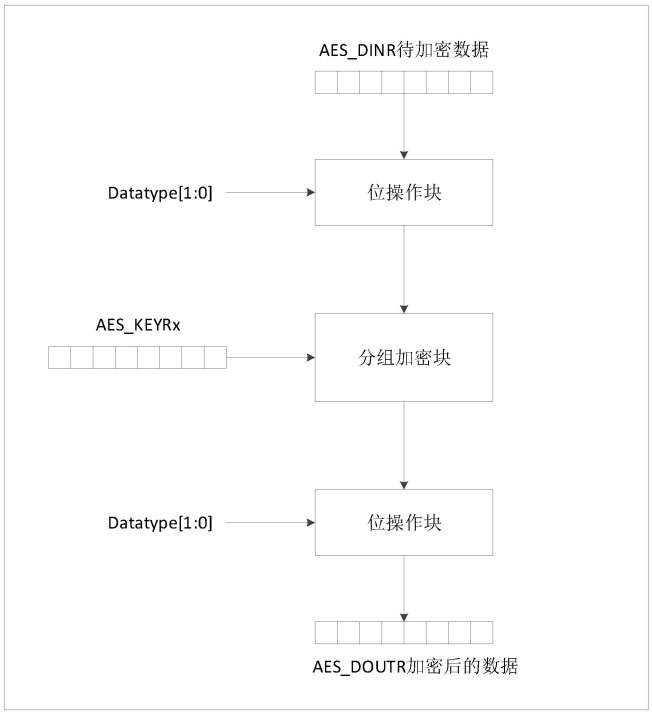


图 5-185 AES 电子密码本算法的加密原理图

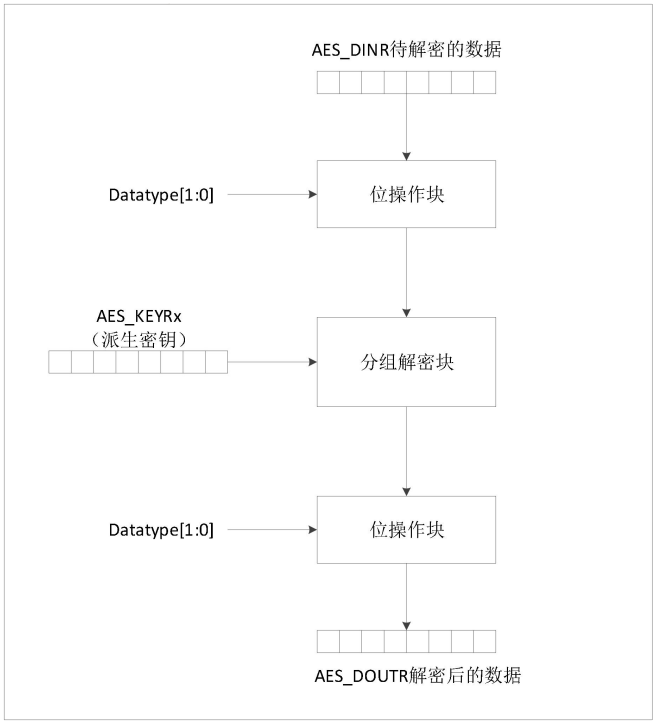


图 5-186 AES 电子密码本算法的解密原理图

密码块链接（CBC）

在密码块链接（CBC）模式下，每个待加密数据在加密之前都与前一个加密密文进行异或。为了使每个消息都是唯一的，在处理第一个加密块时将使用初始化向量（AES_IVRx）。加密模式下 IV 在加密块之前参与异或，解密模式下 IV 在解密块之后参与异或。

密码块链接算法的加密和解密的原理图如下：

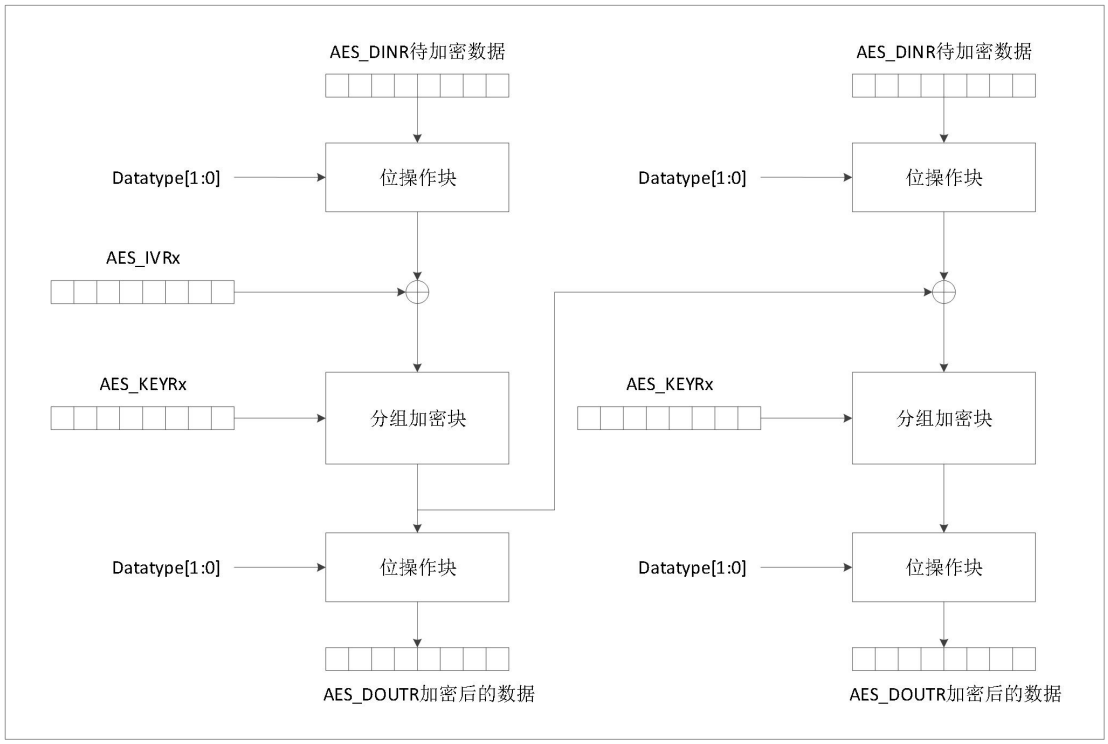


图 5-187 AES 密码块链接算法的加密原理图

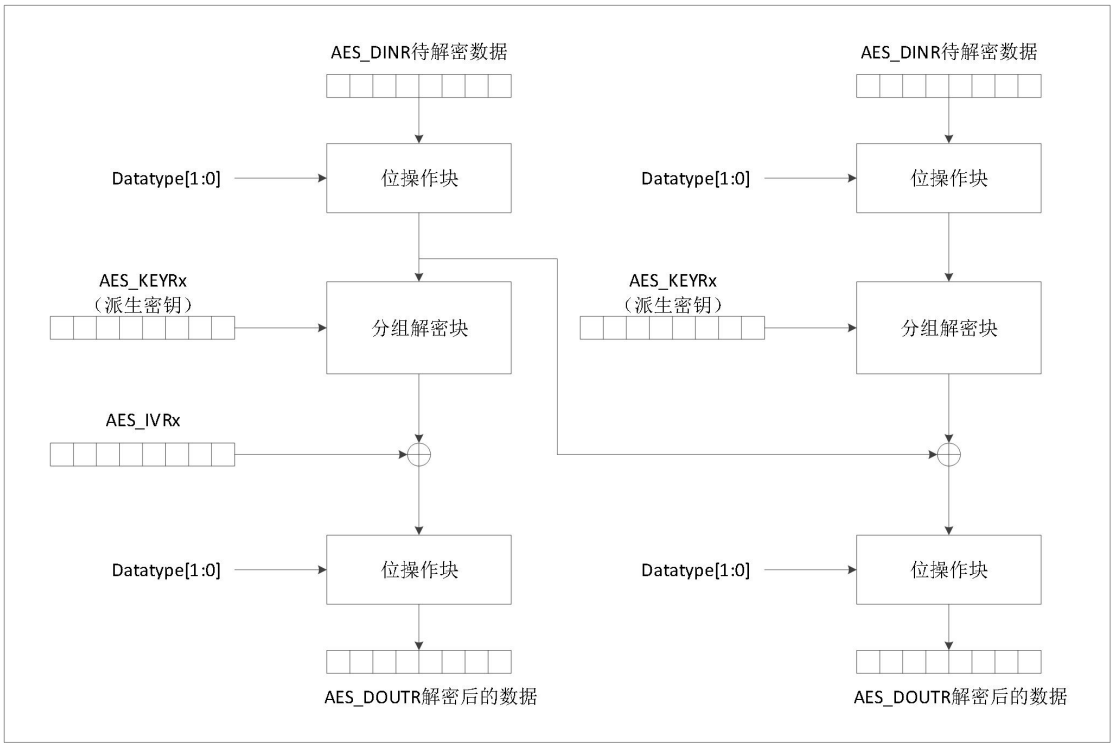


图 5-188 AES 密码块链接算法的解密的原理图

计数器模式（CTR）

在计数器模式下，除了的随机数值外，还使用一个 32 位计数器用于与密文或待加密数据进行 XOR 操作。

计数器模式链接算法的加密和解密的原理图如下：

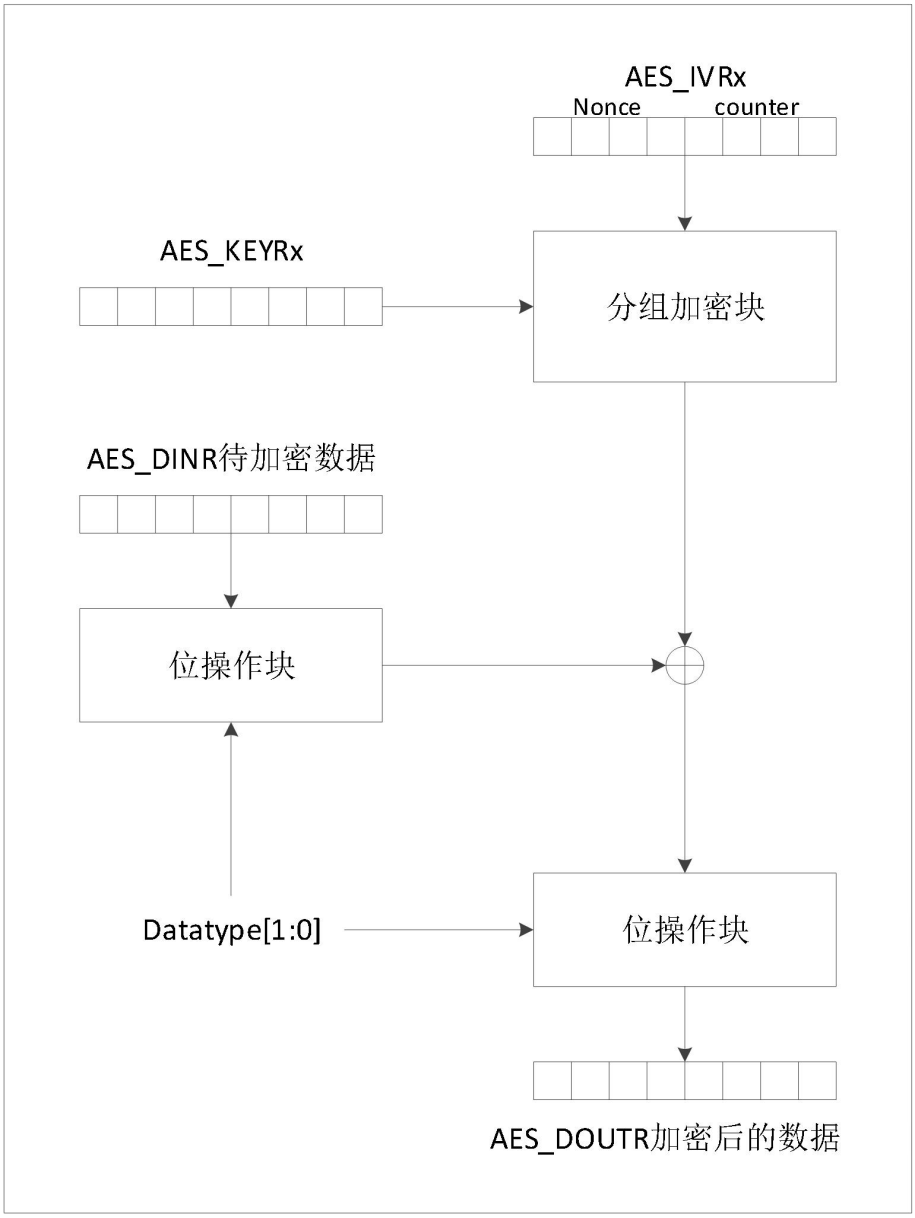


图 5-189　AES 计数器模式链接算法的加密原理图

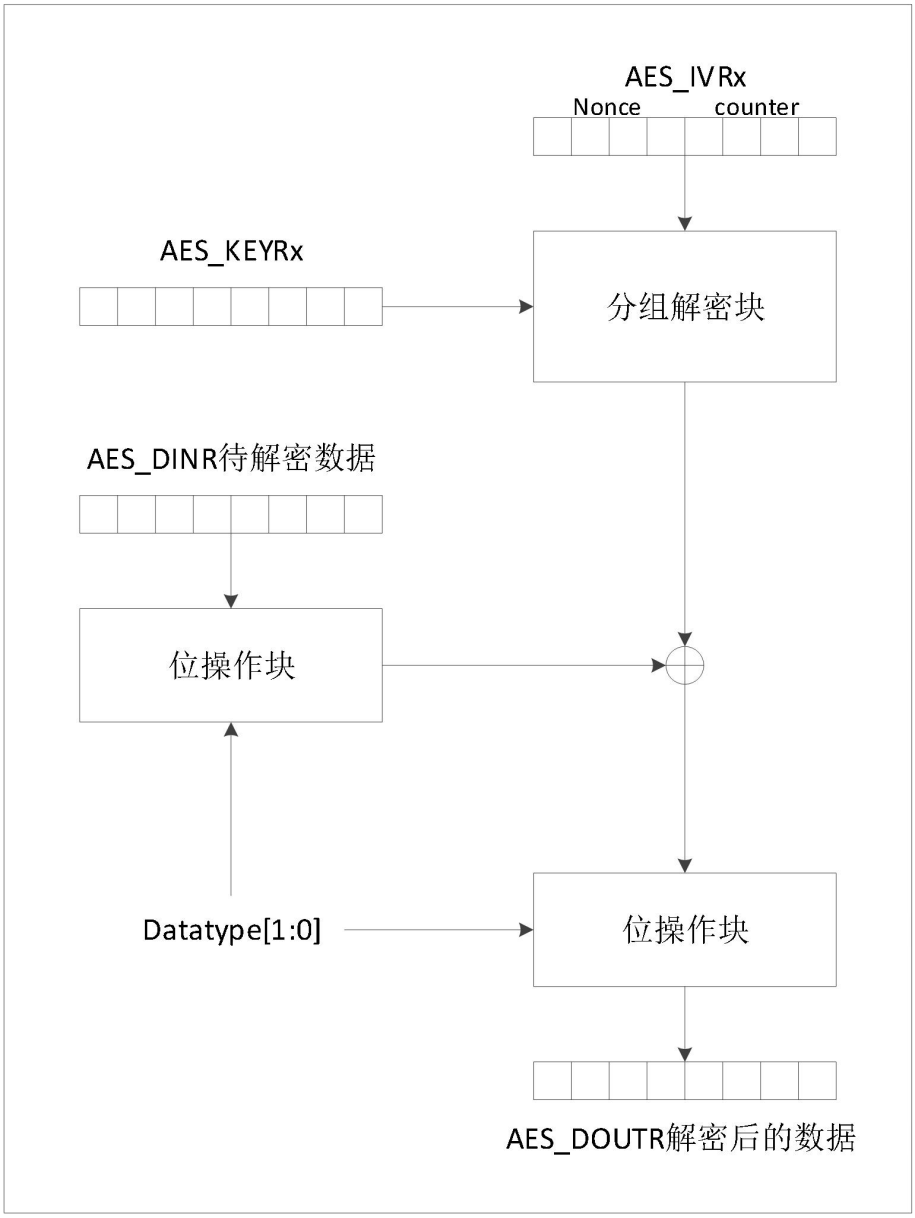
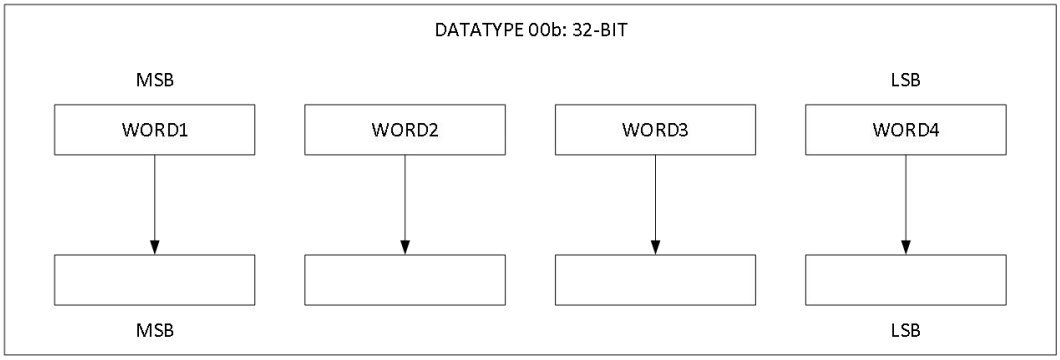


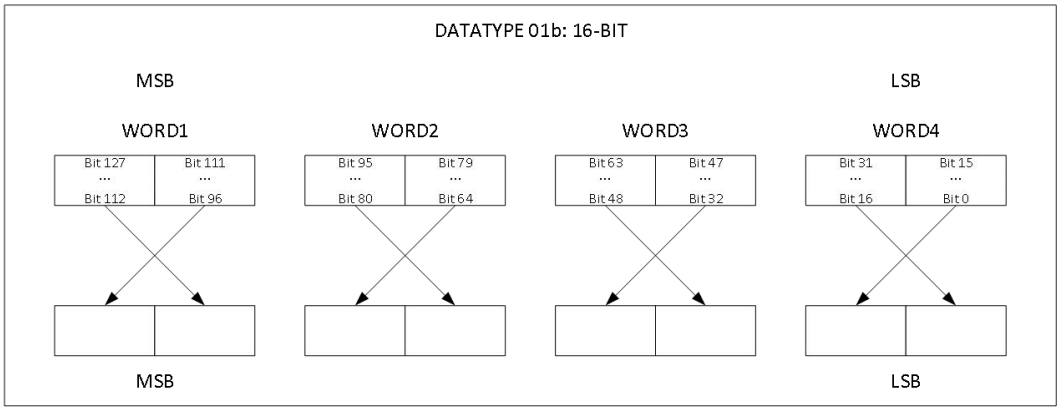
图 5-190 AES 计数器模式链接算法的解密原理图

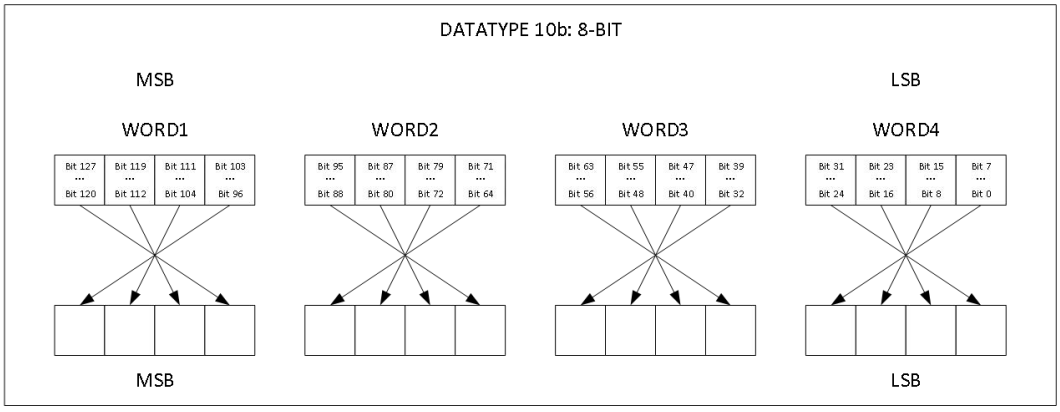
数据模式

AES 提供了四种数据格式来对输入输出数据进行移位操作，具体移位操作请参照上面章节各个模块的算法说明，可以通过修改 AES_CR 寄存器的 DATATYPE 位进行设置。

具体位移原理如下图：


DATATYPE 00b: 32-BIT
MSB
LSB
WORD1
WORD2
WORD3
WORD4
MSB
LSB


DATATYPE 01b: 16-BIT
MSB
LSB
WORD1
WORD2
WORD3
WORD4
Bit 127 ... Bit 112
Bit 111 ... Bit 96
Bit 95 ... Bit 80
Bit 79 ... Bit 64
Bit 63 ... Bit 48
Bit 47 ... Bit 32
Bit 31 ... Bit 16
Bit 15 ... Bit 0
MSB
LSB


DATATYPE 10b: 8-BIT
MSB
LSB
WORD1
WORD2
WORD3
WORD4
Bit 127 ... Bit 120
Bit 119 ... Bit 112
Bit 111 ... Bit 104
Bit 103 ... Bit 96
Bit 95 ... Bit 88
Bit 87 ... Bit 80
Bit 79 ... Bit 72
Bit 71 ... Bit 64
Bit 63 ... Bit 56
Bit 55 ... Bit 48
Bit 47 ... Bit 40
Bit 39 ... Bit 32
Bit 31 ... Bit 24
Bit 23 ... Bit 16
Bit 15 ... Bit 8
Bit 7 ... Bit 0
MSB
LSB

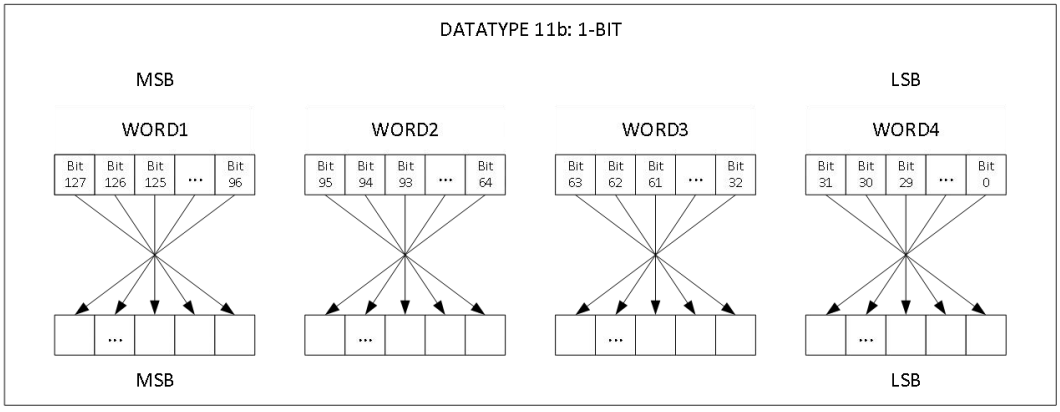

DATATYPE 11b: 1-BIT
MSB
LSB
WORD1
WORD2
WORD3
WORD4
Bit 127
Bit 126
Bit 125
...
Bit 96
Bit 95
Bit 94
Bit 93
...
Bit 64
Bit 63
Bit 62
Bit 61
...
Bit 32
Bit 31
Bit 30
Bit 29
...
Bit 0
MSB
LSB

图 5-191 AES 位移原理图

操作流程

加密

加密是指 AES 利用密钥寄存器（KEY）和初始向量寄存器（IV）在不同模式下对原数据进行加密处理，从而得到一系列密文的过程。加密流程如下：

1. 设置 AES_CR·EN=0 禁用 AES；
2. 设置 AES_CR·MODE[1:0]=00 配置为加密模式，设置 AES_CR·CHMOD[1:0]配置链接模式；
3. ECB 模式时配置 AES_KEYRx 寄存器，CTR 或 CBC 模式时还要配置 AES_IVRx 寄存器；
4. 设置 AES_CR·EN=1 开启 AES；
5. 把待加密的明文数据分四次连续写入 AES_DINR 寄存器（先写入 MSB）；
6. 等待 AES_SR 寄存器中 CCF 标志产生；
7. 连续读取 AES_DOUTR 寄存器四次来获取加密后的密文数据（先读出 MSB）；
8. 重复 5、6、7 完成所有数据的加密。

注：加密时使用的 KEY 为数据加密的初始密钥；

加密时 KEY 寄存器不改变；

CTR 模式加密时 IV 会改变；

密钥派生

密钥派生是指 AES 通过利用密钥寄存器（KEY）在不同模式下进行密钥派生处理，从而得到一系列派生密钥的过程。派生密钥流程如下：

1. 设置 AES_CR • EN=0 禁用 AES；

2. 设置 AES_CR • MODE[1:0]=01 配置为密钥派生模式，派生密钥时无需设置 AES_CR • CHMOD[1:0]，因为密钥派生模式时与所选的链接算法无关；

3. 配置 AES_KEYRx 寄存器，配置 AES_IVRx 寄存器无效；

4. 设置 AES_CR • EN=1 开启 AES；

5. 等待 AES_SR 寄存器中 CCF 标志产生；

6. 此时派生的密钥会储存在 AES_KEYRx 寄存器中，如果需要请读取 AES_KEYRx 寄存器来获得派生密钥；

7. 如果需要重新开始密钥派生操作，请重复步骤 3、4、5、6。

注：密钥派生与链接算法无关，只和原始密钥 KEY 有关。

解密

解密是指 AES 通过利用密钥寄存器（KEY）和初始向量寄存器（IV）在不同模式下对输入的密文数据进行解密处理，从而得到一系列明文数据的过程。解密流程如下：

1. 设置 AES_CR • EN=0 禁用 AES；

2. 设置 AES_CR • MODE[1:0]=10 配置为解密模式，设置 AES_CR • CHMOD[1:0]配置链接模式；

3. ECB 模式时配置 AES_KEYRx 寄存器，CTR 或 CBC 模式时还要配置 AES_IVRx 寄存器；

4. 设置 AES_CR • EN=1 开启 AES；

5. 把待解密的密文数据分四次连续写入 AES_DINR 寄存器（先写入 MSB）；

6. 等待 AES_SR 寄存器中 CCF 标志产生；

7. 连续读取 AES_DOUTR 寄存器四次来获取解密后的明文数据（先读出 MSB）；

8. 重复 5、6、7 完成所有数据的解密。

注：解密模式下使用的 KEY 为派生密钥，派生密钥可以通过密钥派生模式生成，然后再用于解密模式；

CTR 模式下的 KEY 为初始 KEY，CTR 解密模式下的 IV 为初始 IV。

密钥派生+解密

派生密钥+解密是指 AES 先后进行密钥派生和解密过程，因此这里使用的密钥寄存器（KEY）为初始密钥。流程如下：

1. 设置 AES_CR·EN=0 禁用 AES；

2. 设置 AES_CR·MODE[1:0]=11 配置为密钥派生+解密模式，设置 AES_CR·CHMOD[1:0] 配置链接模式（CTR 模式不支持密钥派生+解密模式，若配置此模式，则会硬件强制返回解密模式）；

3. ECB 模式时配置 AES_KEYRx 寄存器，CTR 或 CBC 模式时还要配置 AES_IVRx 寄存器；

4. 设置 AES_CR·EN=1 开启 AES；

5. 把待解密的密文数据分四次连续写入 AES_DINR 寄存器（先写入 MSB）；

6. 等待 AES_SR 寄存器中 CCF 标志产生；

7. 连续读取 AES_DOUTR 寄存器四次来获取解密后的明文数据（先读出 MSB）；

8. 重复 5、6、7 完成所有数据的解密。

注：此模式下使用的 KEY 为原始密钥，AES 工作过程中会进行密钥派生，并且把派生密钥存放在 AES_KEY 寄存器中；

注意 CTR 模式下不支持此类操作，强行配置会被硬件返回至解密模式；

寄存器映射

名称	偏移量	位宽	类型	复位值	描述
AES	BASE: 0x400BD000				
AES_CR	0x00	32	R/W	0x00	AES 控制寄存器
AES_SR	0x04	32	R	0x00	AES 状态寄存器
AES_DINR	0x08	32	R/W	0x00	AES 输入数据寄存器
AES_DOUTR	0x0c	32	R	0x00	AES 输出数据寄存器
AES_KEYR0	0x10	32	R/W	0x00	AES 密钥寄存器 0
AES_KEYR1	0x14	32	R/W	0x00	AES 密钥寄存器 1
AES_KEYR2	0x18	32	R/W	0x00	AES 密钥寄存器 2
AES_KEYR3	0x1C	32	R/W	0x00	AES 密钥寄存器 3
AES_IVR0	0x20	32	R/W	0x00	AES 加密起始点寄存器 0
AES_IVR1	0x24	32	R/W	0x00	AES 加密起始点寄存器 1
AES_IVR2	0x28	32	R/W	0x00	AES 加密起始点寄存器 2
AES_IVR3	0x2C	32	R/W	0x00	AES 加密起始点寄存器 3

寄存器描述

AES_CR 寄存器（0x00）

位域	名称	类型	复位值	描述
31:9	RESERVED	R	0	保留位
8	ERRC	R/W	0	错误标志清除 将 1 写入此位将清除 RDERR 和 WRERR 标志。 此位读出为低。

7	CCFC	W	0	计算完成标志清除。 此位读出为低。
6:5	CHMOD	R/W	0	AES 链接模式选择。 00：电子码本（ECB） 01：密码块链接（CBC） 10：计数器模式（CTR） 11：保留。 只有在禁用 AES 时才能更改 AES 链接模式。 禁止在启用 AES 时写入这些位，以避免不可预知的 AES 行为。
4:3	MODE	R/W	0	AES 模式选择。 00：模式一：加密 01：模式二：密钥派生 10：模式三：解密 11：模式四：密钥派生+解密 只有禁用 AES 时，才能更改操作模式。 禁止在启用 AES 时写入这些位，以避免不可预知的 AES 行为。 如果选择 CTR 模式，则禁止模式 4。如果软件试图为 CTR 模式配置模式 4，则将强制进入模式 3。
2:1	DATATYPE	R/W	0	数据类型选择。 00：32 位数据，不进行位交换。 01：16 位数据或半字，进行半字交换。例如，原数据 0x764356AB，给加密块的值是 0x56AB7643 10：8 位数据或字节，所有字节进行交换。例如，原数据是 0x764356AB，给加密块的值是 0xAB564376。 11：1 位数据。所有位都进行交换。例如，原数据为 0x00112233，给加密块的值是 0xCC448800 只有禁用 AES 时，才能更改操作模式。 禁止在启用 AES 时写入这些位，以避免一些不可预知的 AES 行为。
0	EN	R/W	0	AES 使能信号。 AES 随时可以通过重置此位来初始化： 当 EN 设置之后 AES 准备开始处理新块。 当 AES 在模式 2（密钥派生）完成操作时，该位将由硬件自动清除

AES_SR 寄存器（0x04）

位域	名称	类型	复位值	描述
31:3	RESERVED	R	0	保留位。
2	WRERR	R	0	写入错误标志。 当检测到来自 AES_DINR 寄存器的意外读取操作（在计算或数据输入阶段）时，该位由硬件设置。 软件通过在 AES_CR 寄存器中设置 ERRC 位来清除。 0：未检测到写入错误 1：检测到写入错误 这个标志对 AES 没有影响，即使产生了 WRERR 中断，AES 也会继续运行。
1	RDERR	R	0	读取错误标志。 当检测到来自 AES_DOUTR 寄存器的意外读取操作（在计算或数据输入阶段）时，该位由硬件设置。 软件通过在 AES_CR 寄存器中设置 ERRC 位来清除。 0：未检测到读取错误 1：检测到读取错误 这个标志对 AES 没有影响，即使产生了 RDERR 中断，AES 也会继续运行。
0	CCF	R	0	计算完成标志。 如果 CCFIE 位先前已在 AES_CR 寄存器中设置。 软件通过在 AES_CR 寄存器中设置 CCFC 位来清除。 1：计算完成 0：计算未完成

AES_DINR 寄存器（0x08）

位域	名称	类型	复位值	描述

31:0	DINR	R/W	0	输入数据寄存器。 在输入阶段，必须将该寄存器写入 4 次： - 在模式 1（加密）中，必须写入 4 个字节，表示从 MSB 到 LSB 的数据。 - 在模式 2（密钥派生）中，不使用此寄存器，因为此模式仅涉及从 AES_KEYRx 寄存器开始的密钥派生计算。 - 在模式 3（解密）和模式 4（密钥派生+解密）中，必须写入 4 个字节，表示密码文本 MSB 到 LSB。 注：此寄存器必须以 32 位数据宽度访问。

AES_DOUTR 寄存器（0x0C）

位域	名称	类型	复位值	描述
31:0	DOUTR	R	0	数据输出寄存器 这个寄存器是只读的。 一旦产生了 CCF 标志（计算完成标志），读取该数据寄存器 4 次，即可访问 128 位输出结果： -在模式 1（加密）中，读取的 4 个字节表示从 MSB 到 LSB 的加密数据。 -在模式 2（密钥派生）中，不需要读取该寄存器，因为派生密钥位于 AES_KEYRx 寄存器中。 -在模式 3（解密）和模式 4（密钥派生+解密）中，读取的 4 个字表示从 MSB 到 LSB 的纯文本。 注：此寄存器必须以 32 位数据宽度访问。

AES_KEYR0 寄存器（0x10）

位域	名称	类型	复位值	描述

31:0	KEYR	R/W	0	密钥寄存器[31:0] 必须在打开 AES 使能之前写入此寄存器： 在模式 1（加密）、模式 2（密钥派生）和模式 4（密钥派生+解密）中，要写入的值表示来自 LSB 的加密密钥，这意味着密钥[31:0]。 在模式 3（解密）中，要写入的值表示来自 LSB 的解密密钥 [31:0]。当在这种解密模式下用加密密钥写入寄存器时，在启用 AES 之前读取它将返回加密值。设置 CCF 标志后读取它将返回派生密钥。 启用 AES 时读取此寄存器将返回一个不可预测的值。 注意：此寄存器不包含模式 4 中的派生密钥（派生密钥+解密）。它始终包含加密密钥值。

AES_KEYR1 寄存器（0x14）

位域	名称	类型	复位值	描述
31:0	KEYR	R/W	0	密钥寄存器[63:32]

AES_KEYR2 寄存器（0x18）

位域	名称	类型	复位值	描述
31:0	KEYR	R/W	0	密钥寄存器[95:64]

AES_KEYR3 寄存器（0x1C）

位域	名称	类型	复位值	描述
31:0	KEYR	R/W	0	密钥寄存器[127:96]

AES_IVR0 寄存器（0x20）

位域	名称	类型	复位值	描述
31:0	IVR	R/W	0	初始化向量寄存器[31：0] 必须在 AES_CR 寄存器的 EN 位置 1 之前写入此寄存器： 在以下情况下，寄存器值没有意义： - ECB 模式（电子密码本）。 - CTR 或 CBC 模式下的密钥派生。 在 CTR 模式（计数器模式）下，该寄存器包含 32 位计数器值。 在启用 AES 的同时读取该寄存器将返回值 0x00000000。

AES_IVR1 寄存器（0x24）

位域	名称	类型	复位值	描述
31:0	IVR	R/W	0	初始化向量寄存器[63:32] 在 CTR 模式（计数器模式）下，该寄存器包含随机数值。 在启用 AES 的同时读取该寄存器将返回值 0x00000000。

AES_IVR2 寄存器（0x28）

位域	名称	类型	复位值	描述
31:0	IVR	R/W	0	初始化向量寄存器[95:64] 在 CTR 模式（计数器模式）下，该寄存器包含随机数值。 在启用 AES 的同时读取该寄存器将返回值 0x00000000。

AES_IVR3 寄存器（0x2C）

位域	名称	类型	复位值	描述
31:0	IVR	R/W	0	初始化向量寄存器（MSB IVR [127：96]） 在 CTR 模式（计数器模式）下，该寄存器包含随机数值。 在启用 AES 的同时读取该寄存器将返回值 0x00000000。

6. 电气特性

本章说明芯片的电气参数，包括工作电压、工作温度、功耗、模拟特性参数及 IO 的特性参数等。

6.1 测试条件

除非特别说明，所有电压以 3.3V 为基准。除非特别说明，最大和最小参数是在环境温度 T_A = 25 °C， V_{DD} =3.3V 下进行测试的。

6.2 绝对最大额定值

6.2.1 电压特性

表 1 电压特性

符号	描述	最小值	典型值	最大值	单位
$V_{DD} - V_{SS}$	外部主供电电压	-0.3		3.6	V
V_{IN}	引脚的输入电压	VSS-0.3		3.6	V
Δ_{VDD}	不同供电引脚之间的电压差			50	mV
Δ_{VSS}	不同接地引脚之间的电压差			50	mV

6.2.2 电流特性

表 2 电流特性

符号	描述	最小值	典型值	最大值	单位
I_{DD}	经过电源线的总电流			120	*mA*
I_{OUT}	单个引脚上的输出电流	-30		30	*mA*
I_{IN}	单个引脚上的注入电流	-5		5	*mA*
$\sum I_{IN}$	所有引脚上的总注入电流	-25		25	*mA*

6.2.3 温度特性

表 3 温度特性

符号	描述	最小值	典型值	最大值	单位
T_{STG}	储存温度范围	-45		150	℃
T_J	最大结温度			125	℃

6.3 工作条件

6.3.1 通用工作条件

表 4 共用工作条件表

符号	描述	最小值	典型值	最大值	单位
V_{DD}	标准工作电压	2.0	3.3	3.6	V
F_{CLK}	系统时钟频率	-	-	72	MHz
T_A	工作温度	-40	-	85	℃

6.3.2 上电和掉电的工作条件

表 5 上电和掉电的工作条件

符号	描述	最小值	典型值	最大值	单位
T_{VR}	VDD 上升速率	0.3	-	∞	uS/V
T_{VF}	VDD 下降速率	0.3	-	∞	uS/V

6.3.3 复位和电源控制模块特性

表 6 复位和电源控制模块特性

符号	描述	最小值	典型值	最大值	单位
V_{POR}	上电复位阈值	1.87	1.9	1.96	V
V_{PDR}	掉电复位阈值	1.76	1.8	1.85	V

V_{HYS}	PDR 迟滞	90	100	110	mV
T_{RST}	复位持续时间	-	0.6	-	ms

注：此为电路仿真理论值。

6.3.4 供电电流特性

表 7 供电电流特性

符号	描述	条件	最小值	典型值	最大值	单位
I_{DD}	运行模式下供电电流	3.3V@48MHz Run in RAM All Peripherals clock OFF				*mA*
I_{DD}	运行模式下供电电流	3.3V@24MHz Run in RAM All Peripherals clock OFF				*mA*
I_{DD}	运行模式下供电电流	3.3V@12MHz Run in RAM All Peripherals clock OFF				μA
I_{DD}	运行模式下供电电流	3.3V@32KHz Run in RAM All Peripherals clock OFF				μA
I_{DD}	运行模式下供电电流	3.3V@48MHz All Peripherals clock ON				*mA*
I_{DD}	运行模式下供电电流	3.3V@24MHz All Peripherals clock ON				*mA*
I_{DD}	运行模式下供电电流	3.3V@12MHz All Peripherals clock ON				μA
I_{DD}	运行模式下供电电流	3.3V@48MHz All Peripherals clock OFF				*mA*
I_{DD}	运行模式下供电电流	3.3V@24MHz All Peripherals clock OFF				*mA*
I_{DD}	运行模式下供电电流	3.3V@12MHz All Peripherals clock OFF				μA
I_{DD}	sleep 模式下的供电电流	3.3V@48MHz	-	12	-	μA
I_{DD}	sleep 模式下的供电电流	3.3V@48MHz All Peripherals clock ON				μA
I_{DD}	sleep 模式下的供电电流	3.3V@24MHz All Peripherals clock ON				μA

I_{DD}	sleep 模式下的供电电流	3.3V@12MHz All Peripherals clock ON				μA
I_{DD}	sleep 模式下的供电电流	3.3V@48MHz All Peripherals clock OFF				μA
I_{DD}	sleep 模式下的供电电流	3.3V@24MHz All Peripherals clock OFF				μA
I_{DD}	sleep 模式下的供电电流	3.3V@12MHz All Peripherals clock OFF				μA
I_{DD}	deepsleep 模式下的供电电流	3.3V@48MHz	-	5	-	μA
I_{DD}	stop 模式下的供电电流	3.3V@48MHz	-	0.5	-	μA
I_{DD}	stop 模式下的供电电流	3.3V@48MHz All Peripherals clock OFF				μA
I_{DD}	stop 模式下的供电电流	3.3V@24MHz All Peripherals clock OFF				μA
I_{DD}	stop 模式下的供电电流	3.3V@12MHz All Peripherals clock OFF				μA

注：所有带 All Peripherals clock 条件项电流为链接部分基本外设后测量得出。

6.3.5 内嵌参考电压

符号	描述	条件	最小值	典型值	最大值	单位
V_{REFINT}	内部参考电压	-40℃~85℃	1.19	1.2	1.21	V
tS_vrefint	ADC sampling time when reading the internal reference voltage	-	-	5	-	us

注：此为电路仿真理论值。

6.3.6 低功耗模式唤醒时间

符号	描述	条件	最小值	典型值	最大值	单位
$T_{wusleep}$	sleep 模式唤醒	-	-	100	-	us
$T_{wudeepsleep}$	deepsleep 模式唤醒	-	-	140	-	us
T_{wustop}	stop 模式唤醒	-	-	350	-	us

注：此为电路仿真理论值。

6.3.7 内部高频时钟源特性

符号	描述	条件	最小值	典型值	最大值	单位
f_{HSI}	时钟频率	-	-	48	-	MHz
TRIM	Trim step	-	-	0.8	-	%
ACC	振荡器精度	VDD=2.0V~3.6 V -40℃~85℃	-1	-	1	%
DC	占空比	VDD=3.3V	45	50	55	%
T_{SU}	振荡器启动时间	-	-	10	-	μS
I_{DD}	功耗	-	-	100	85	μA

注：此为电路仿真理论值。

6.3.8 内部低频时钟源特性

符号	描述	条件	最小值	典型值	最大值	单位
F_{LSI}	时钟频率	-	-	32.768	-	KHz
TRIM	Trim step	-	-	-	1.8	%
ACC	振荡器精度	VDD=2.0V~3.6 V -40℃~85℃	-3		3	%
DC	占空比	VDD=3.3V	45	50	55	%
T_{SU}	振荡器启动时间	-	-	-	80	μS
I_{DD}	功耗	-	-	-	1	μA

注：此为电路仿真理论值。

6.3.9 外部高频时钟源特性

符号	描述	条件	最小值	典型值	最大值	单位
F_{HSE}	时钟频率	-	4	-	32	MHz
T_{SU}	振荡器启动时间	-	0.3	-	1	ms

I_{DD}	功耗	VDD = 3.3V, Rm = 45Ω, CL = 10pF@8MHz	-	0.86	1.45	uA
		VDD = 3.3V, Rm = 30Ω, CL = 20pF@16MHz	-	0.8	1.2	uA

注：此为电路仿真理论值。

6.3.10 外部低频时钟源特性

符号	描述	条件	最小值	典型值	最大值	单位
F_{LSE}	时钟频率	-	-	32.768	-	KHz
T_{SU}	振荡器启动时间	CL=20pF	-	600	-	ms
I_{DD}	功耗	CL = 20pF@8MHz	1.2	-	2	uA

注：此为电路仿真理论值。

6.3.11PLL 特性

符号	描述	条件	最小值	典型值	最大值	单位
f_{PLL_IN}	输入时钟频率	-	4	-	48	MHz
	输入时钟占空比	-	40	-	60	%
f_{PLL_OUT}	输出时钟频率	-	-	72	-	MHz
T_{lock}	PLL 锁定时间	-	-	-	35	us
Jitter	Cycle-to-cycle jitter	-	-		200	ps

注：此为电路仿真理论值。

6.3.12FLASH 存储器特性

符号	描述	条件	最小值	典型值	最大值	单位
T_{prog}	32-bit 编程时间	-40℃~85℃	-	-	20	us
T_{erase}	扇区（0.5KB）擦时间	-40℃~85℃	-	-	4	ms

IDD	编程电流	-40℃~85℃	-	-	0.9	mA
	扇区擦电流	-40℃~85℃	-	-	0.9	mA
	Standby 电流	常温	-	90	-	uA
	Deep power down 电流	常温	-	3	-	uA
N_{cycle}	编程/擦次数	85℃	10	-	-	万次
T_{ret}	数据保持	85℃	10	-	-	年

注：设计保证，不在产品中测试。

6.3.13EMC 特性

符号	描述	条件	最大值	单位
ESD(HBM)	静电放电人体模型	TA = 25◦C， 符合 JEDEC JS-001-2017	2000	V
ESD(CDM)	静电放电充电设备模型	TA = 25◦C， 符合 JEDEC JS-002-2014	500	V
LatchUp	静态闩锁类	TA = 25◦C， 符合 JEDEC78D	100	*mA*

注：此为电路仿真理论值。

6.3.14IO 端口特性

符号	描述	条件	最小值	典型值	最大值	单位
V_{IL}	输入低电平电压	TC（3.3V IO）	-	-	0.39VDD	V
		FT（5V tolerance IO）	-	-	0.47VDD	V
V_{IH}	输入高电平电压	TC（3.3V IO）	0.55VDD	-	-	V
		FT（5V tolerance IO）	0.51VDD	-	-	V
V_{hys}	施密特迟滞电压	-	-	240	-	mV
I_{IH}	输入漏电流	-	-1	-	1	uA

I_{IL}		-	-1	-	1	uA
R_{PU}	弱上拉等效电阻	-	-	40	-	K Ω
R_{PD}	弱下拉等效电阻	-	-	40	-	K Ω
V_{OL}	输出低电平	TC（3.3V IO） |IIO| = 8mA VDD ≥ 2.7V	-	-	0.27	V
V_{OH}	输出高电平		VDD-0.29	-	-	V
V_{OL}	输出低电平	TC（3.3V IO） |IIO| = 20mA VDD ≥ 2.7V	-	-	1	V
V_{OH}	输出高电平		VDD-0.87	-	-	V
V_{OL_5}	输出低电平	FT（5V tolerance IO） |IIO| = 8mA VDD ≥ 2.7V	-	-	0.28	V
V_{OH_5}	输出高电平		VDD-0.3	-	-	V
V_{OL_5}	输出低电平	FT（5V tolerance IO） |IIO| = 20mA VDD ≥ 2.7V	-	-	0.9	V
V_{OH_5}	输出高电平		VDD-0.91	-	-	V

注：此为电路仿真理论值。

6.3.15EXTRST 端口特性

符号	描述	条件	最小值	典型值	最大值	单位
V_{IL}	输入低电平电压	-	-	-	0.3VDD	V
V_{IH}	输入高电平电压	-	0.7VDD	-	-	V
V_{hys}	施密特迟滞电压	-	0.71	1.19	1.33	V
R_{PU}	弱上拉等效电阻	$V_{IN} = V_{SS}$	-	40	-	K Ω
V_F	输入滤波脉冲电压	-	-	-	70	ns
V_{NF}	输入非滤波脉冲电压	-	-	350	-	ns

注：此为电路仿真理论值。

6.3.16ADC 特性

符号	描述	条件	最小值	典型值	最大值	单位
V_{DD}	ADC 输入电压	-	2	-	3.6	V
I_{DD}	ADC 电流	VDD = 3.3V，常温	-	0.5	-	mV
f_{ADC}	ADC 采样时钟频率	-	-	48	-	MHz
fs	采样率	-	-	2.4	-	MHz
V_{AIN}	转换电压范围	-	0	-	VDD	V
R_{AIN}	外部输入阻抗	VIN = VSS	-	-	50	KΩ
R_{ADC}	采样开关电阻	-	-	-	0.65	KΩ
C_{ADC}	内部采样及保持电阻	-	-	-	3.8	pF
T_s	采样时间	-	1	-	128	$1/f_{ADC}$
T_{CONV}	总转换时间（包括采样时间、转换时间、数据运算及存储）	$f_{ADC} = f_{SYS} = 48MHz$	20	-	147	$1/f_{ADC}$
$V_{extvref}$	外部参考电压	-	-	-	VDD	V

注：此为电路仿真理论值。

6.3.17ADC 精度

符号	描述	条件	最小值	典型值	最大值	单位
ET	Total unadjusted error	fADC = 48M, RAIN＜3kΩ, VDDA=2.7V to 3.6V TA = -40 to 85℃	-	-	+/-4	LSB
EO	Offset error		-	-	+/-1.5	LSB
EG	Gain error		-	-	+/-2.5	LSB
ED	Differential linearity error		-	-	+/-1	LSB
EL	Integral linearity error		-	-	+/-1.7	LSB

注：此为电路仿真理论值。

6.3.18 温度传感器特性

符号	描述	条件	最小值	典型值	最大值	单位
T_L	线性精度	-	-	-	+/-2.7	℃
Avg_Slope	Slope 均值	-	3.05	3.13	3.2	mv/℃
V_{25}	25℃时电压	-	1.34	1.543	1.52	V
T_{start}	启动时间	-	4	-	10	us
T_{s_temp}	ADC 测量温度时的采样时间	-	5	-	-	us

注：此为电路仿真理论值。

6.3.19 运算放大器特性

符号	描述	条件	最小值	典型值	最大值	单位
VDD	电压范围	-	2.0	-	3.6	V
CMIR	输入范围	-	0	-	VDD	V
V_{opa_offset}	Input offset voltage maximum trim range	25℃，No load output	-	-	8	mV
	Input offset voltage after offset trim	25℃，No load output	-	-	0.4	mV
		2.0V~3.6V，-40℃~85℃	-	-	0.8	mV
I_{load}	驱动电流	-	-	-	500	uA
IDD	功耗	无负载，静态模式	-	-	320	uA
CMRR	Common mode rejection ratio	-	54	-	196	dB
PSRR	Power supply rejection ratio	-	72	-	138	dB
GBW	带宽	-	2.1	4.7	12.6	MHz
SR	Slew Rate	-	2.5	4.1	5.4	V/us
R_{load}	电阻负载	-	4	-	-	KΩ

C_{load}	电容负载	-	-	-	50	pF
VOH_{sat}	High saturation voltage	-	VDD-30	-	-	mV
VOL_{sat}	Low saturation voltage	-	-	-	31	mV
PM	Phase Margin	-	46	67	97	°
GM	Gain Margin	-	6.6	17	30	dB
THD	Total Harmonic Distortion	-	3.1	3.2	3.3	%

注：此为电路仿真理论值。

6.3.20 比较器特性

符号	描述	条件	最小值	典型值	最大值	单位
VDD	电压范围	-	2.0	-	3.6	V
IDD	功耗	-	-	30	45	uA
V_{CM}	输入电压范围	-	0.35	0.5VDD	VDD-0.3	V
V_{offset}	输入 offset 电压	迟滞关闭时	-	10	20	mV
V_{hys}	迟滞窗口	-	-	60	140	mV
T_{start}	比较器启动时间	-	-	-	400	ns
A_V	DC Voltage Gain	-	45	65	75	dB

注：此为电路仿真理论值。