ARM® Cortex®-M0

32-位 微控制器

NuMicro® 家族
Mini58DE 系列
技术参考手册

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

目录

1 概述 12
2 特性 13
3 缩写表 16
4 编号信息列表与管脚定义 17
4.1 NuMicro® Mini58 系列产品命名规则 17
4.2 NuMicro® Mini58 系列产品选型指南 18
4.3 管脚配置 19
4.3.1 LQFP 48-pin 19
4.3.2 QFN 33-pin 21
4.3.3 TSSOP 20-pin 23
4.4 管脚描述 24
5 方框图 29
5.1 NuMicro® Mini58 方框图 29
6 功能描述 30
6.1 ARM® Cortex®-M0 内核 30
6.1.1 概述 30
6.1.2 特性 30
6.2 系统管理 32
6.2.1 概述 32
6.2.2 系统复位 32
6.2.3 电源模式和唤醒源 38
6.2.4 系统电源结构 40
6.2.5 系统内存映射 42
6.2.6 内存组织 42
6.2.7 寄存器映射 44
6.2.8 寄存器描述 45
6.2.9 系统定时器(SysTick) 70
6.2.10 嵌套向量中断控制器(NVIC) 74

6.2.11 系统控制寄存器(SCB) 114
6.3 时钟控制器 123
6.3.1 概述 123
6.3.2 自动校准 125
6.3.3 系统时钟和SysTick 时钟 125
6.3.4 外设时钟源选择 126
6.3.5 掉电模式时钟 128
6.3.6 分频器输出 128
6.3.7 寄存器映射 130
6.3.8 寄存器描述 131
6.4 存储控制器 (FMC) 146
6.4.1 概述 146
6.4.2 特性 146
6.4.3 框图 146
6.4.4 功能描述 148
6.4.5 Flash 控制寄存器 164
6.4.6 Flash 控制寄存器 165
6.5 通用I/O (GPIO) 175
6.5.1 概述 175
6.5.2 特性 175
6.5.3 框图 176
6.5.4 基本配置 176
6.5.5 功能描述 177
6.5.6 GPIO 中断和唤醒功能 178
6.5.7 寄存器映射 179
6.5.8 寄存器 描述 183
6.6 定时器控制器(TIMER) 198
6.6.1 概述 198
6.6.2 特性 198
6.6.3 框图 198

6.6.4 基本配置 199
6.6.5 功能描述 199
6.6.6 寄存器映射 204
6.6.7 寄存器描述 205
6.7 增强型PWM 发生器 214
6.7.1 概述 214
6.7.2 特性 214
6.7.3 方块图 216
6.7.4 基本配置 217
6.7.5 功能描述 218
6.7.6 寄存器映射 236
6.7.7 寄存器描述 238
6.8 看门狗定时器 (WDT) 274
6.8.1 概述 274
6.8.2 特性 274
6.8.3 框图 274
6.8.4 时钟控制 274
6.8.5 基本配置 275
6.8.6 功能描述 275
6.8.7 寄存器表 277
6.8.8 寄存器描述 278
6.9 窗口看门狗定时器 (WWDT) 281
6.9.1 概述 281
6.9.2 特性 281
6.9.3 框图 281
6.9.4 时钟控制 282
6.9.5 基本配置 282
6.9.6 功能描述 282
6.9.7 寄存器表 285
6.9.8 寄存器描述 286
6.10 串口控制器 (UART) 291
6.10.1 概述 291

6.10.2 特性 291
6.10.3 框图 291
6.10.4 基本配置 295
6.10.5 功能描述 295
6.10.6 寄存器表 305
6.10.7 寄存器描述 306
6.11 I²C 串行接口控制器 (I²C) 327
6.11.1 概述 327
6.11.2 特征 327
6.11.3 基本配置 327
6.11.4 框图 328
6.11.5 功能描述 328
6.11.6 寄存器映射 343
6.11.7 寄存器描述 344
6.12 串行外围总线 (SPI) 355
6.12.1 概述 355
6.12.2 特征 355
6.12.3 框图 355
6.12.4 基本配置 356
6.12.5 功能描述 356
6.12.6 时序图 362
6.12.7 编程示例 364
6.12.8 寄存器映射 367
6.12.9 寄存器描述 368
6.13 模数转换(ADC) 382
6.13.1 概述 382
6.13.2 特性 382
6.13.3 框图 383
6.13.4 基本配置 383
6.13.5 功能描述 384
6.13.6 寄存器映射 389
6.13.7 寄存器描述 390
6.14 模拟比较器(ACMP) 404

6.14.1 概述 404
6.14.2 **特性** 404
6.14.3 框图 405
6.14.4 基本配置 405
6.14.5 功能描述 406
6.14.6 比较器参考电压(CRV) 407
6.14.7 **寄存器表** 409
6.14.8 寄存器描述 410
7 应用电路 416
8 电气特性 418
9 封装尺寸 419
9.1 48-pin LQFP 419
9.2 33-pin QFN (4 mm x 4 mm) 420
9.3 33-pin QFN (5 mm x 5 mm) 421
9.4 20-pin TSSOP 422
10 修订历史 423

图集

图 4.1-1 NuMicro® Mini58 系列产品命名规则 17
图 4.3-1 NuMicro® Mini58 系列 LQFP 48-pin 管脚图 19
图 4.3-2 NuMicro® Mini58 系列 LQFP 48-pin 多功能管脚图 20
图 4.3-3 NuMicro® Mini58 系列 QFN 33-pin 管脚图 21
图 4.3-4 NuMicro® Mini58 系列 QFN 33-pin 多功能管脚图 22
图 4.3-5 NuMicro® Mini58 系列 TSSOP 20-pin 管脚图 23
图 4.3-6 NuMicro® Mini58 系列 TSSOP 20-pin 多功能管脚图 23
图 5.1-1 NuMicro® Mini58 系列方框图 29
图 6.1-1 功能模块方框图 30
图 6.2-1 系统复位源 33
图 6.2-2 nRESET复位波形 35
图 6.2-3 上电复位(POR)波形 35
图 6.2-4 低电复位 (LVR) 波形 36
图 6.2-5 欠电压检测(BOD)复位波形图 37
图 6.2-6 电源模式状态机 38
图 6.2-7 NuMicro® Mini58 系列电源结构框图 41
图 6.3-1 时钟发生器框图 123
图 6.3-2 时钟发生器全局框图 124
图 6.3-3 系统时钟框图 125
图 6.3-4 SysTick 时钟控制框图 126
图 6.3-5 PCLK外设总线时钟源选择 127
图 6.3-6 分频器的时钟源框图 129
图 6.3-7 分频器框图 129
图 6.4-1 存储控制器框图 147
图 6.4-2 数据FLASH与APROM共享存储空间 149
图 6.4-3 SPROM 安全锁模式 154
图 6.4-4 Flash存储器映射 155
图 6.4-5 支持IAP的系统存储器映射 156
图 6.4-6 支持IAP的LDROM模式 157
图 6.4-7 支持IAP的APROM模式 157
图 6.4-8 不支持IAP功能的系统内存映射 158
图 6.4-9 启动源选择 159
图 6.4-10 ISP流程举例 161

图 6.4-11 CRC-32 Checksum 计算流程 162
图 6.4-12 CRC-32 Checksum计算流程 163
图 6.5-1 GPIO 控制器框图 176
图 6.6-1 定时器控制器框图 198
图 6.6-2 定时器控制器时钟源 199
图 6.6-3 Continuous Counting 模式 200
图 6.6-4 Free-Counting捕捉模式 201
图 6.6-5 外部复位计数模式 202
图 6.7-1 应用电路框图 215
图 6.7-2 PWM 方框图 216
图 6.7-3 PWM CH0/1发生器架构图 216
图 6.7-4 PWM CH2/3发生器架构图 217
图 6.7-5 PWM CH4/5发生器架构图 217
图 6.7-6 边沿对齐模式PWM 219
图 6.7-7 PWM边沿对齐时序图 219
图 6.7-8 边沿对齐流程图 220
图 6.7-9 内部比较器工作原理 221
图 6.7-10 PWM工作时序 221
图 6.7-11 中心对齐模式 222
图 6.7-12 中心对齐模式时序图 223
图 6.7-13 中心对齐模式波形图 223
图 6.7-14 中心对齐模式流程图 224
图 6.7-15 精确中心对齐模式 225
图 6.7-16 精确中心对齐波形图 226
图 6.7-17 精确中心对齐流程图 227
图 6.7-18 PWM中心加载时序 228
图 6.7-19 PWM双缓存图解 228
图 6.7-20 PWM控制器输出占空比 229
图 6.7-21 死区时间示意图 230
图 6.7-22 不对称模式时序图 231
图 6.7-23 上升沿死区PWM极性控制初始化状态 232
图 6.7-24 电机控制PWM中断架构 233
图 6.7-25 屏蔽和屏蔽对齐 234
图 6.8-1 看门狗定时器模块图 274

图 6.8-2 看门狗定时器时钟控制 275
图 6.8-3 看门狗定时器定时溢出间隔和复位周期时序图 276
图 6.9-1 窗口看门狗定时器模块图.....281
图 6.9-2 窗口看门狗定时器时钟控制图.....282
图 6.9-3 窗口看门狗定时器复位和重载过程 283
图 6.10-1 UART0 控制器时钟控制 292
图 6.10-2 UART1 控制器时钟控制 292
图 6.10-3 UART0 控制器框图.....293
图 6.10-4 UART1 控制器框图.....293
图 6.10-5 自动流控模块图 295
图 6.10-6 控制器波特率公式表.....296
图 6.10-7 控制器波特率参数设置表.....296
图 6.10-8 控制器波特率寄存器设置表 297
图 6.10-9 UART 控制器中断源和标志 298
图 6.10-10 UART线控制的数据位和停止位的长度设置.....299
图 6.10-11 UART 线控制校验位的设置 299
图 6.10-12 CTS 自动流控使能.....300
图 6.10-13 RTS 自动流控使能.....300
图 6.10-14 RTS 软件控制的流控 301
图 6.10-15 IrDA 控制器模块框图 301
图 6.10-16 IrDA TX/RX 时序图.....302
图 6.10-17 RS-485 自动方向模式下通过RTS 驱动电平 303
图 6.10-18 RS-485 软件控制下的RTS 驱动电平 304
图 6.10-19 RS-485 帧结构 304
图 6.11-1 I^2C 控制器框图.....328
图 6.11-2 I^2C 总线时序 328
图 6.11-3 I^2C 协议 329
图 6.11-4 起始(START)和停止(STOP)条件.....330
图 6.11-5 I^2C 总线上的位传输 330
图 6.11-6 I^2C 总线上的应答信号.....331
图 6.11-7 主机传送数据到从机 331
图 6.11-8 主机向从机读数据.....331
图 6.11-9 主机写时 二级缓存传输时序 332
图 6.11-10 从机读时二级缓存传输时序 332

图 6.11-11 根据I²C当前状态控制总线 333
图 6.11-12 主机传送模式控制流程 334
图 6.11-13 主机接收模式控制流程 335
图 6.11-14 从机模式控制流程....336
图 6.11-15 GC 模式 337
图 6.11-16 I2C 数据移动方向 339
图 6.11-17 I²C 超时计数器框图 340
图 6.11-18 EEPROM 随机读操作....342
图 6.11-19 随机读协议 342
图 6.12-1 SPI 框图....355
图 6.12-2 SPI 主机模式应用框图....356
图 6.12-3 SPI 从机模式应用框图....357
图 6.12-4 一次传输32-位 357
图 6.12-5 字节重排序 359
图 6.12-6 字节休眠时序波形....359
图 6.12-7 FIFO 模式框图....361
图 6.12-8 SPI 主机模式下的时序....363
图 6.12-9 SPI主机模式下的时序 (交替SPI_CLK相位) 363
图 6.12-10 SPI从机模式下的时序....364
图 6.12-11 SPI从机模式下的时序(交替SPI_CLK相位) 364
图 6.13-1 AD控制器框图 383
图 6.13-2 ADC时钟控制 384
图 6.13-3 单一模式转换时序图 385
图 6.13-4 ADC启动转换条件 385
图 6.13-5 A/D转换结果监控逻辑框图 386
图 6.13-6 A/D控制器中断 386
图 6.13-7 单端输入转换结果映射图....387
图 6.13-8 ADC 连续模式用于 2/3-shunt 388
图 6.13-9 ADC连续模式用于1-shunt 388
图 6.14-1 模拟比较器框图 405
图 6.14-2 比较器控制器中断源....406
图 6.14-3 比较器的迟滞功能....406
图 6.14-4 比较器参考电压框图 408

表集

表 3-1 缩写列表 16
表 4.2-1 NuMicro® Mini58 系列产品选型指南 18
表 4.4-1 NuMicro® Mini58 系列管脚描述 28
表 6.2-1 寄存器复位值 34
表 6.2-2 电源模式详解 38
表 6.2-3 不同电源模式下的时钟情况 39
表 6.2-4 各外设再次进入掉电模式的条件 40
表 6.2-2 内存映射表 42
表 6.2-3 片上模块地址空间分配 43
表 6.2-7 异常模式 75
表 6.2-9 向量表格式 76
表 6.3-1 外设时钟源选择表 128
表 6.3-2 掉电模式控制 132
表 6.6-1 输入捕捉模式操作流程 203
表 6.8-1 看门狗定时器定时溢出间隔周期选择 276
表 6.9-1 窗口看门狗定时器预分频值选择 283
表 6.9-2 CMPDAT 设置限制 284
表 6.11-1 I^2C 状态码描述表 340

1 概述

NuMicro® Mini58 系列32位微控制器内嵌ARM® Cortex®-M0内核，与NuMicro® Mini51 系列管脚、功能兼容。与传统的8位机及16位机相比，Mini58系列单片机有着32机优异的性能、丰富的功能和相当的价格。Mini58系列应用广泛，无论是低端、低成本的设计还是需要复杂计算的应用都能胜任，同时它还可以为经济型的产品提供更多的高级功能。

Mini58 系列最高运行频率可达50MHz，超过Mini51 24MHz的最高运行频率，工作电压为2.5V ~ 5.5V，工作温度-40℃ ~ 105℃。 Mini58 系列程序flash大小从16KB增加成32KB，SRAM大小从2KB增加成4KB。同时，Mini58系列提供大小可配置的数据flash（与程序flash共用）、用于ISP功能的2KB flash。

Mini58 系列为了减少元器件数量、电路板面积以及系统成本，集成了很多高性能的系统级外设功能，如：内置的22.1184MHz RC振荡器(±1%的精度)、多达30个的I/O口、四路32位的Timer、两组可用作RS485和IrDA的UART,、一组SPI,、两组I^2C,、多达三组6通道的16位PWM发生器、一个10位8通道 ADC、看门狗定时器、窗口看门狗定时器、两组模拟比较器及欠压检测。相对于Mini51系列，Mini58系列增加了一组UART和一组I2C，使其可以更方便的应用于通讯应用。

此外，Mini58 系列支持ISP（在系统编程）和ICP（在线电路编程）功能，允许用户不用将芯片从实际产品上取下来就可以更新程序。Mini58系列同样提供在应用编程（IAP）功能，用户可以在flash更新完程序后，直接执行更新的程序，而不用复位芯片。

2 特性

- 内核
  - ARM® Cortex®-M0 内核，最高可运行到 50 MHz
  - 一个 24位系统定时器
  - 支持低功耗空闲模式
  - 单周期32位硬件乘法器
  - NVIC用于32个中断源，每个中断有4级优先级
  - 支持串行调试接口（SWD），支持2个观察点/4个中断点
- 内建LDO，支持宽工作电压：2.5V~5.5V
- 存储器
  - 32 KB Flash 存储器用于存储程序代码 (APROM)
  - 可配置的 Flash存储器用于存储数据 (Data Flash)
  - 2 KB Flash 用于装载程序 (LDROM)
  - 4 KB SRAM 用于内部高速数据缓存(SRAM)
- 时钟控制
  - 可编程系统时钟源
    - 可即时切换系统时钟源
  - 支持4 ~ 24 MHz 外部高速晶振(HXT)用于精准的时间操作
  - 支持32.768 kHz 外部低速晶振 (LXT) 用于空闲模式唤醒和系统运行
  - 内置22.1184 MHz 内部高速振荡器 (HIRC)用于系统操作 (在25^{0}C 、5V环境下，精确度为1%)
    - 在-40℃到105℃环境下，可通过外部32.768K 晶振振荡器(LXT)动态校准HIRC 振荡器频率为22.1184 MHz ±1%
  - 内置10 kHz 内部低速振荡器(LIRC)用于看门狗定时器和唤醒操作
  - PLL允许CPU运行频率最高为50 MHz
- I/O 口
  - LQFP-48封装高达30根通用IO口(GPIO)引脚
  - 四种 I/O 模式:
    - 准双向输入/输出
    - 推挽输出
    - 开漏输出
    - 高阻态输入
  - 可选择施密特触发输入
- 定时器
  - 提供两个32位定时器，每个定时器包括一个8位预分频计数器和一个24位向上计数器

- 支持时间计数模式
- 支持触发输出模式
- 支持外部触发脉冲宽度测量模式
- 支持外部触发脉冲捕捉模式

- WDT (看门狗定时器)
  - 可编程时钟源和时间溢出周期
  - 可从掉电模式或空闲模式下唤醒
  - 可选择在看门狗溢出后中断还是复位
- WWDT (窗口看门狗定时器)
  - 6位向下计数值(CNTDAT)和6位比较值(CMPDAT)使得WWDT溢出窗口周期可灵活设置
- 支持4位数值(PSCSEL)编程WWDT计数器预分频值，最大可达11位。
- PWM
  - 高达3个内置PWM发生器，提供6个PWM输出或3对互补PWM输出
  - 每个PWM发生器拥有独立的时钟源、时钟分频器、8位预分频器和死区时间发生器
  - PWM周期中断
  - 支持边缘对齐或中心对齐
  - 支持错误检测
- UART (通用异步接收器/发射器)
  - 两个 UART 设备
  - 接收器和发射器都有缓冲，均为16字节FIFO
  - 可选流控制功能 (CTSn 和 RTSn)
  - 支持 IrDA (SIR) 功能
  - 可编程波特率发生器，波特率高达系统时钟的1/16
  - 支持 RS-485 功能
- SPI (串行外设接口)
  - 一个SPI设备
  - 主机时钟高达25 MHz,，从机高达10 MHz
  - 支持主机/从机模式
  - 全双工同步数据传输
  - 传输数据长度可配置为1~32位
  - 可选择MSB或LSB优先传输
  - RX引脚可在时钟上升沿或下降沿接收数据
  - TX引脚可在时钟上升沿或下降沿发射数据
  - 在32位传输模式下，支持字节传输延迟模式
- I^2C

- 两路I^2C设备
- 支持主机/从机模式
- 主从机间双向数据传输
- 多主机总线 (无中心主机)
- 总线仲裁，可避免多主机同时传输数据时的冲突
- 串行时钟同步机制允许设备通过一条总线实现设备间不同位速率的通信
- 串行时钟同步可用作为握手机制，控制串行数据传输暂停或恢复
- 可编程时钟允许多种波特率控制
- 支持多地址识别(4个带屏蔽功能的从机地址)

◆ ADC (模数转换器)
- 10位SAR ADC，转换速率可达250 kSPS
- 多达8通道单端输入和1通道来自band-gap的内部输入
- 可通过软件触发或外部引脚触发来启动转换

◆ 模拟比较器
- 两通道模拟比较器，拥有16级内部参考电压
- 内置CRV (比较器参考电压)

◆ 可通过ISP (在系统编程)、ICP (在电路编程)和 IAP (在应用编程) 更新程序

◆ BOD (欠压检测)
- 4个可编程等级： 4.4V/3.7V/2.7V/2.2V
- 可选择欠压中断或复位

◆ 96位唯一的ID

◆ LVR (低电复位)
- 复位门限电压: 2.0V

◆ 运行温度： -40℃~105℃

◆ 可靠性： EFT > ± 4KV, ESD HBM可达4KV

◆ 封装：
- 无铅封装 (RoHS)
- 48引脚LQFP (7x7), 33引脚QFN (5x5) , 33引脚QFN (4x4), 20引脚TSSOP

3 缩写表

缩写	描述
ACMP	Analog Comparator Controller
ADC	Analog-to-Digital Converter
AHB	Advanced High-Performance Bus
APB	Advanced Peripheral Bus
BOD	Brown-out Detection
DAP	Debug Access Port
FIFO	First In, First Out
FMC	Flash Memory Controller
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HIRC	22.1184 MHz Internal High Speed RC Oscillator
HXT	4~24 MHz External High Speed Crystal Oscillator
ICP	In Circuit Programming
ISP	In System Programming
ISR	Interrupt Service Routine
LDO	Low Dropout Regulator
LIRC	10 kHz internal low speed RC oscillator (LIRC)
LXT	32.768 kHz External Low Speed Crystal Oscillator
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PLL	Phase-Locked Loop
PWM	Pulse Width Modulation
SPI	Serial Peripheral Interface
SPS	Samples per Second
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
UCID	Unique Customer ID
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

表 3-1 缩写列表

4 编号信息列表与管脚定义

4.1 NuMicro® Mini58 系列产品命名规则

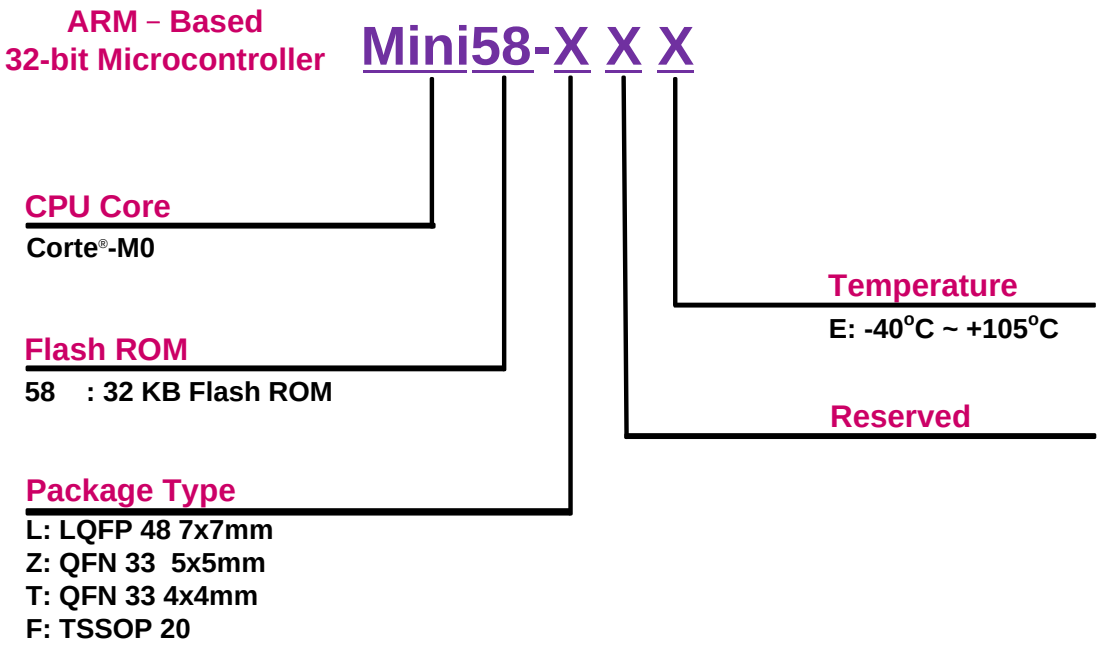


图 4.1-1 NuMicro® Mini58 系列产品命名规则

4.2 NuMicro® Mini58 系列产品选型指南

Part Number	APROM	RAM	Data Flash	ISP Loader ROM	I/O	Timer	Connectivity			Comp.	PWM	ADC	ISP ICP	IRC 22.1184 MHz	Package
							UART	SPI	I^2C						
MINI58LDE	32 KB	4 KB	Configurable	2 KB	up to 30	2x32-bit	2	1	2	2	6	8x10-bit	v	v	LQFP48
MINI58ZDE	32 KB	4 KB	Configurable	2 KB	up to 29	2x32-bit	2	1	2	2	6	8x10-bit	v	v	QFN33(5x5)
MINI58TDE	32 KB	4 KB	Configurable	2 KB	up to 29	2x32-bit	2	1	2	2	6	8x10-bit	v	v	QFN33(4x4)
MINI58FDE	32 KB	4 KB	Configurable	2 KB	up to 17	2x32-bit	2	1	2	-	6	4x10-bit	V	V	TSSOP20

表 4.2-1 NuMicro® Mini58 系列产品选型指南

4.3 管脚配置

4.3.1 LQFP 48-pin

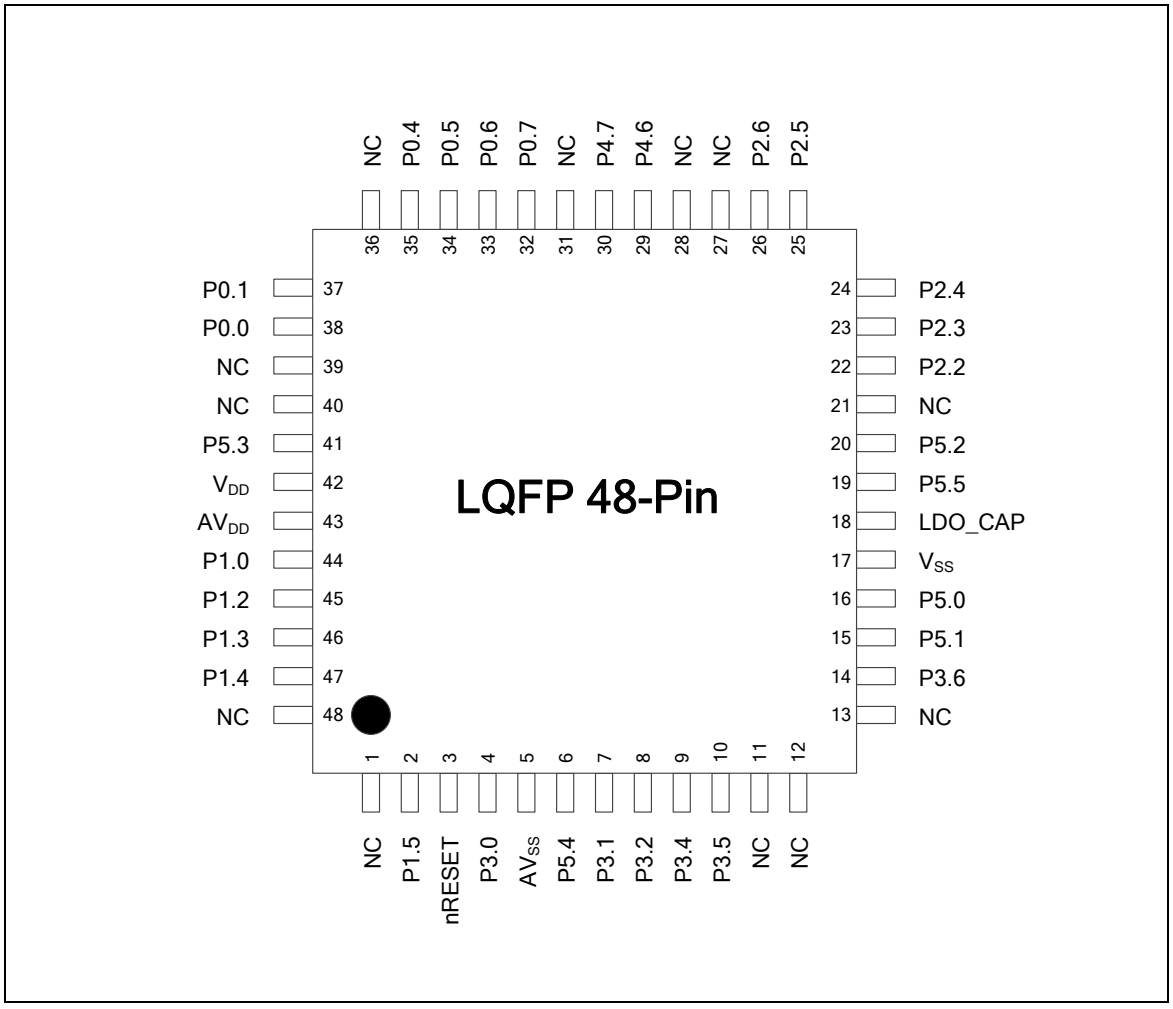


图 4.3-1 NuMicro® Mini58 系列 LQFP 48-pin 管脚图

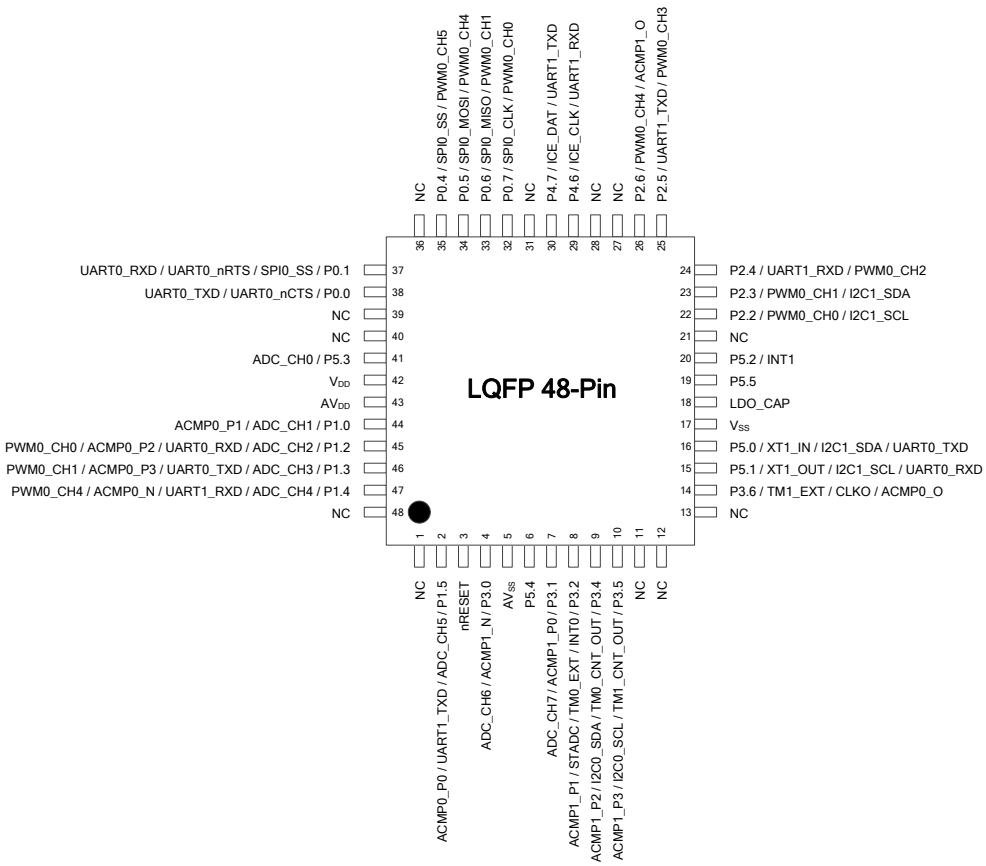


图 4.3-2 NuMicro® Mini58 系列 LQFP 48-pin 多功能管脚图

4.3.2 QFN 33-pin

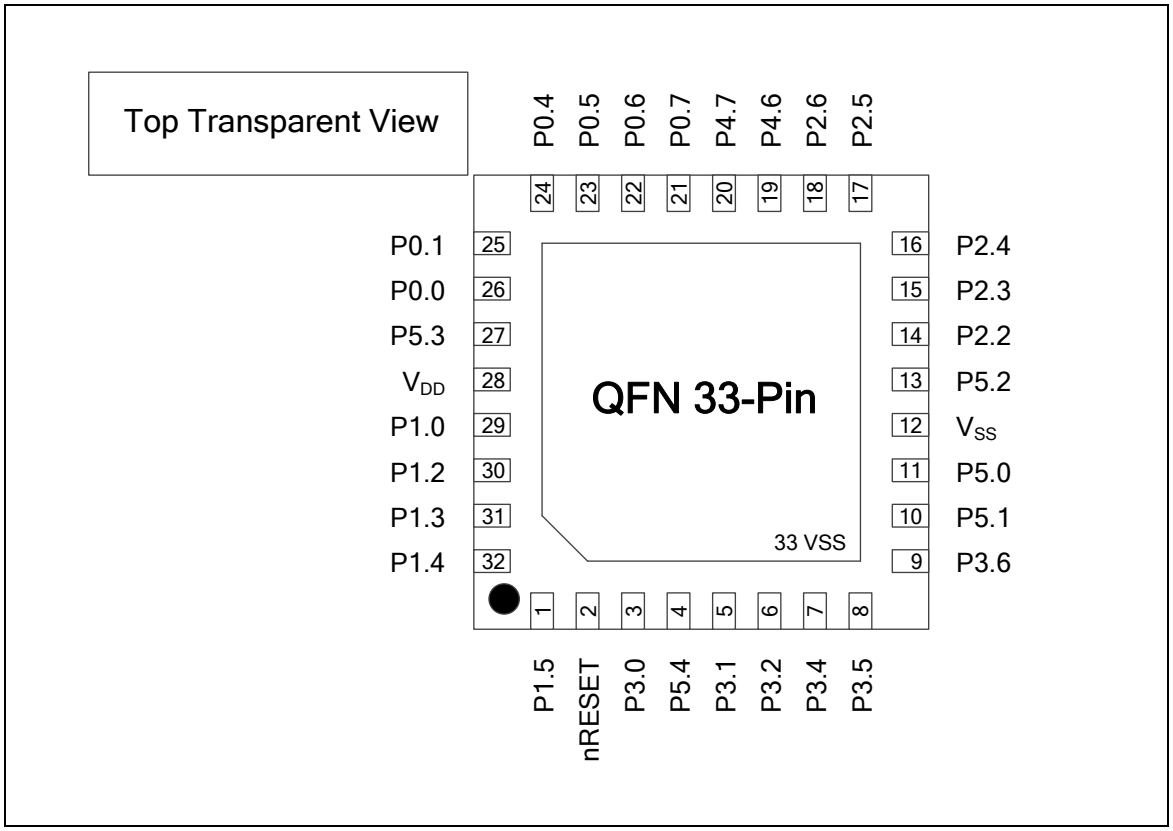


图 4.3-3 NuMicro® Mini58 系列 QFN 33-pin 管脚图

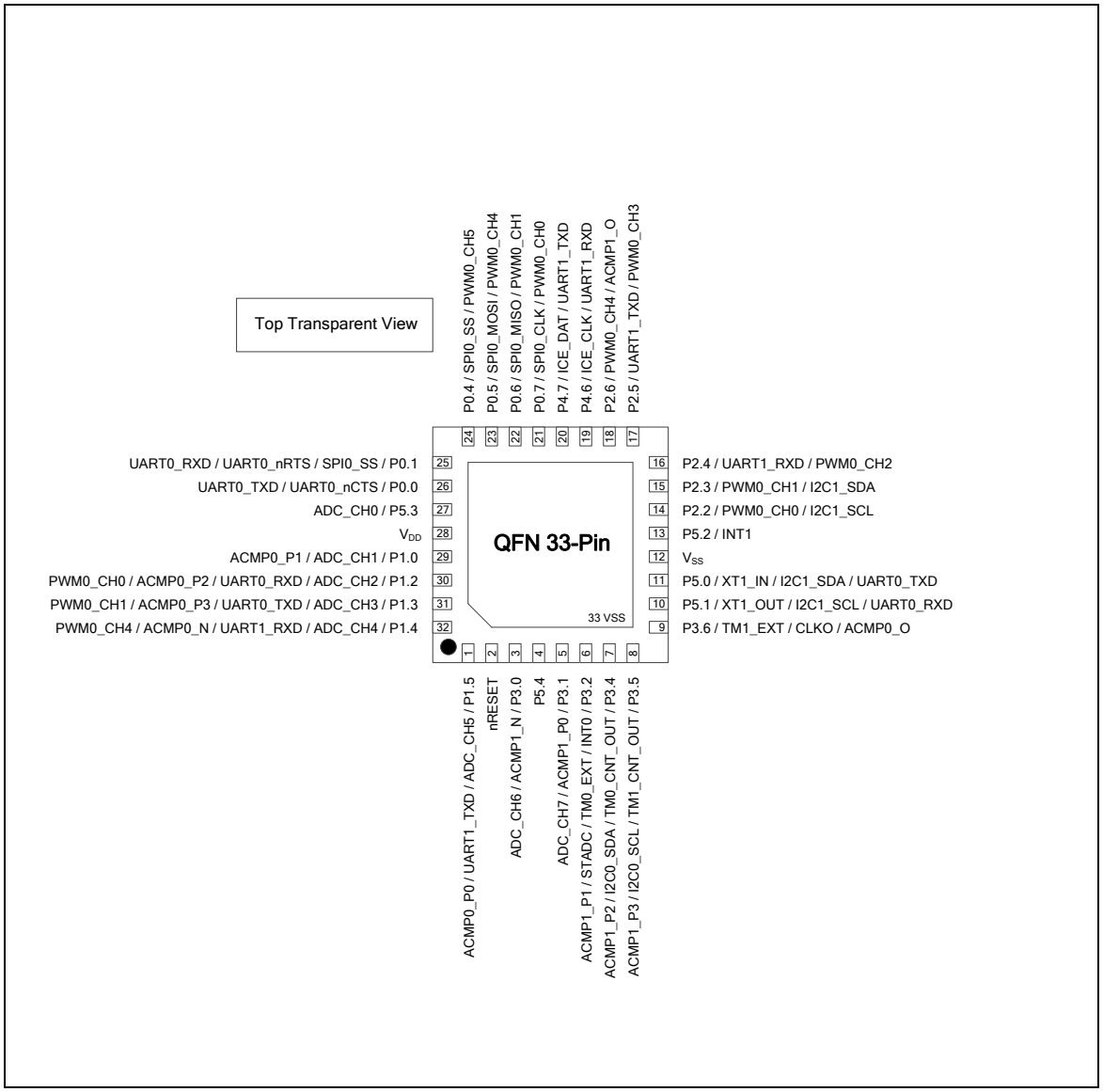


图 4.3-4 NuMicro® Mini58 系列 QFN 33-pin 多功能管脚图

4.3.3 TSSOP 20-pin

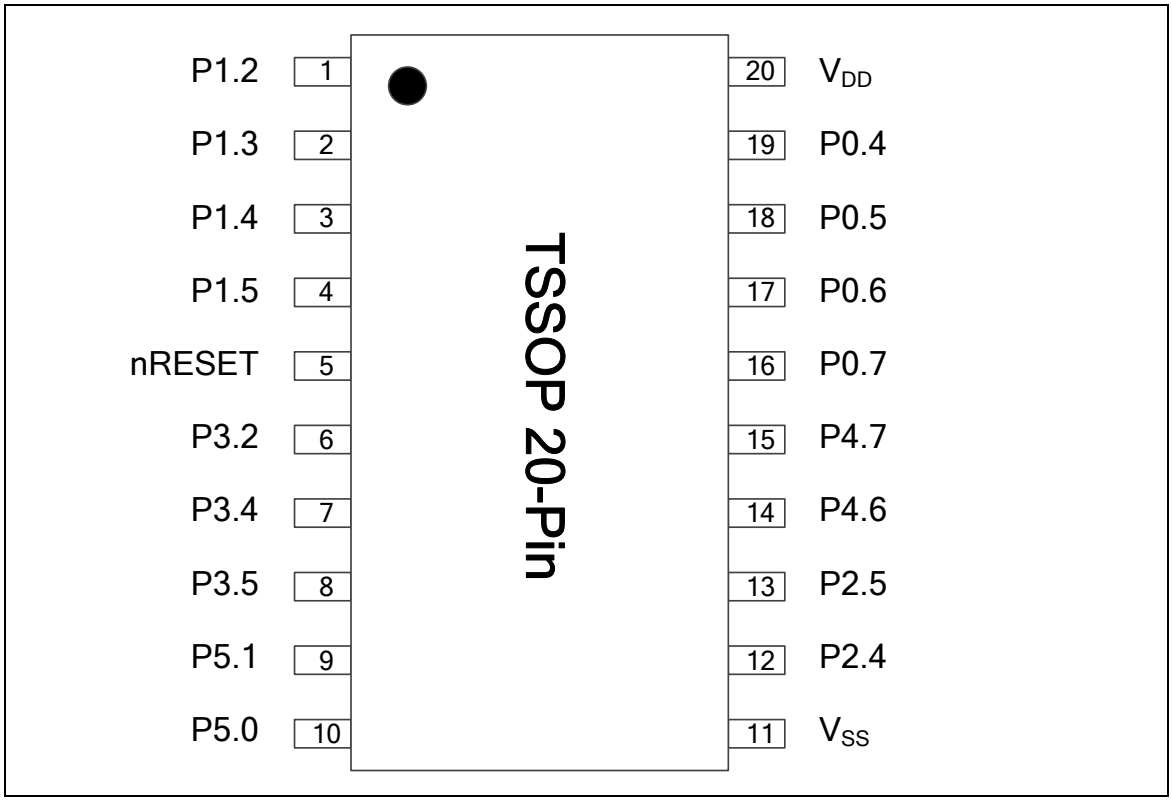


图 4.3-5 NuMicro® Mini58 系列 TSSOP 20-pin 管脚图

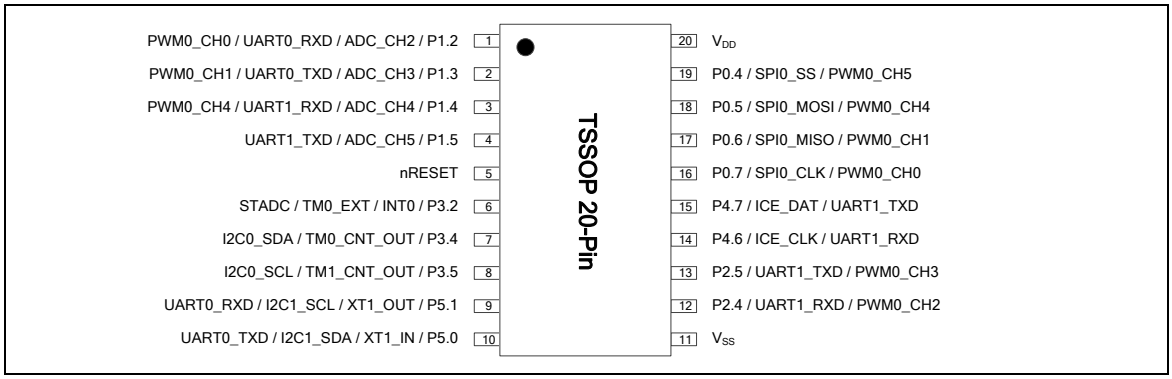


图 4.3-6 NuMicro® Mini58 系列 TSSOP 20-pin 多功能管脚图

4.4 管脚描述

管脚号			管脚名称	管脚类型	描述
LQFP 48-pin	QFN 33-pin	TSSOP 20-pin			
1	---	---	NC	---	无连接
2	1	4	P1.5	I/O	通用数字输入/输出管脚
			ADC_CH5	AI	ADC模拟输入管脚
			UART1_TXD	O	UART1 数据输出管脚
			ACMP0_P0	AI	模拟比较器正输入管脚
3	2	5	nRESET	I(ST)	该施密特输入管脚用于复位硬件设备。当系统时钟运行时，只要该引脚保持低电平768个clock（clock为内部RC振荡器22.1184 MHz），设备将会复位。nRESET管脚有一个内部上拉电阻，因此只要连接一个外部电容到地就可以实现上电复位。
4	3	---	P3.0	I/O	通用数字输入/输出管脚
			ADC_CH6	AI	ADC 模拟输入管脚
			ACMP1_N	AI	模拟比较器负输入管脚
5	---	---	AV_{SS}	AP	模拟地
6	4	---	P5.4	I/O	通用数字输入/输出管脚
7	5	---	P3.1	I/O	通用数字输入/输出管脚
			ADC_CH7	AI	ADC 模拟输入管脚
			ACMP1_P0	AI	模拟比较器正输入管脚
8	6	6	P3.2	I/O	通用数字输入/输出管脚
			INT0	I	外部中断0输入管脚
			STADC	I	ADC 外部触发输入管脚
			TMR0_EXT	I/O	Timer 0外部捕捉 / 复位触发输入管脚 / 翻转输出管脚
			ACMP1_P1	AI	模拟比较器正输入管脚（TSSOP20封装不支持）
9	7	7	P3.4	I/O	通用数字输入/输出管脚
			TMR0_CNT_OUT	I/O	Timer 0 外部事件计数器输入管脚 / 翻转输出管脚
			I2C0_SDA	I/O	I^2C0 数据输入/输出管脚
			ACMP1_P2	AI	模拟比较器正输入管脚
10	8	8	P3.5	I/O	通用数字输入/输出管脚
			TMR1_CNT_OUT	I/O	Timer 1 外部事件计数器输入管脚 / 翻转输出管脚
			I2C0_SCL	I/O	I^2C0 时钟输入/输出管脚
			ACMP1_P3	AI	模拟比较器正输入管脚

管脚号			管脚名称	管脚类型	描述
LQFP 48-pin	QFN 33-pin	TSSOP 20-pin			
11	---	---	NC	---	无连接
12	---	---	NC	---	无连接
13	---	--	NC	---	无连接
14	9	---	P3.6	I/O	通用数字输入/输出管脚
			ACMP0_O	O	模拟比较器输出管脚
			CLKO	O	频率分频输出管脚
			TMR1_EXT	I/O	Timer 1 外部捕捉 / 复位触发输入管脚 / 翻转输出管脚
15	10	9	P5.1	I/O	通用数字输入/输出管脚
			XT1_OUT	O	外部4~24MHz晶振输出管脚
			I2C1_SCL	I/O	I^2C1 时钟输入/输出
			UART0_RXD	I	UART0 数据接收输入管脚
16	11	10	P5.0	I/O	通用数字输入/输出管脚
			XT1_IN	I	外部4~24MHz晶振输入管脚
			I2C1_SDA	I/O	I^2C1 数据输入/输出管脚
			UART0_TXD	O	UART0 数据输出管脚
17	12	11	V_{SS}	P	数字地
	33				
18	---	---	LDO_CAP	P	LDO 输出管脚
19	---	---	P5.5	I/O	通用数字输入/输出管脚 在QFN-33封装中，用户程序必须使能上拉电阻
20	13	---	P5.2	I/O	通用数字输入/输出管脚
			INT1	I	外部中断1输入管脚
21	---	---	NC	---	无连接
22	14	---	P2.2	I/O	通用数字输入/输出管脚
			PWM0_CH0	O	PWM0 输出管脚
			I2C1_SCL	I/O	I^2C1 时钟输入/输出管脚
23	15	---	P2.3	I/O	通用数字输入/输出管脚
			PWM0_CH1	O	PWM1 输出管脚
			I2C1_SDA	I/O	I^2C1 数据输入/输出管脚
24	16	12	P2.4	I/O	通用数字输入/输出管脚
			UART1_RXD	I	UART1数据接收输入管脚

管脚号			管脚名称	管脚类型	描述
LQFP 48-pin	QFN 33-pin	TSSOP 20-pin			
			PWM0_CH2	O	PWM2 输出管脚
25	17	13	P2.5	I/O	通用数字输入/输出管脚
			UART1_TXD	O	UART1数据输出管脚
			PWM0_CH3	O	PWM3 输出管脚
26	18	---	P2.6	I/O	通用数字输入/输出管脚
			PWM0_CH4	O	PWM4 输出管脚
			ACMP1_O	O	模拟比较器输出管脚
27	---	---	NC	---	无连接
28	---	---	NC	---	无连接
29	19	14	P4.6	I/O	通用数字输入/输出管脚
			ICE_CLK	I	串行调试器时钟管脚
			UART1_RXD	I	UART1 数据接收输入管脚
30	20	15	P4.7	I/O	通用数字输入/输出管脚
			ICE_DAT	I/O	串行调试器数据管脚
			UART1_TXD	O	UART1 数据输出管脚
31	---	---	NC	---	无连接
32	21	16	P0.7	I/O	通用数字输入/输出管脚
			SPI0_CLK	I/O	SPI 时钟管脚
			PWM0_CH0	O	PWM0 输出管脚
33	22	17	P0.6	I/O	通用数字输入/输出管脚
			SPI0_MISO	I/O	SPI MISO(主机输入/从机输出) 管脚
			PWM0_CH1	O	PWM1 输出管脚
34	23	18	P0.5	I/O	通用数字输入/输出管脚
			SPI0_MOSI	O	SPI MOSI(主机输出/从机输入)管脚
			PWM0_CH4	O	PWM4 输出管脚
35	24	19	P0.4	I/O	通用数字输入/输出管脚
			SPI0_SS	I/O	SPI 从机选择管脚
			PWM0_CH5	O	PWM5 输出引脚
36	---	---	NC	---	无连接
37	25	---	P0.1	I/O	通用数字输入/输出管脚
			UART0_nRTS	O	UART0 RTS 管脚

管脚号			管脚名称	管脚类型	描述
LQFP 48-pin	QFN 33-pin	TSSOP 20-pin			
			UART0_RXD	**I**	UART0 数据接收输入管脚
			SPI0_SS	**I/O**	SPI 从机选择管脚
38	26	---	P0.0	**I/O**	通用数字输入/输出管脚
			UART0_nCTS	**I**	UART0 CTS 管脚
			UAR0_TXD	**O**	UART0 数据输出管脚
39	---	---	NC	**---**	无连接
40	---	---	NC	**---**	无连接
41	27	---	P5.3	**I/O**	通用数字输入/输出管脚
			ADC_CH0	**AI**	ADC 模拟输入管脚
42	28	20	V_{DD}	**P**	数字电源
43			AV_{DD}	**P**	模拟电源
44	29	---	P1.0	**I/O**	通用数字输入/输出管脚
			ADC_CH1	**AI**	ADC 模拟输入管脚
			ACMP0_P1	**AI**	模拟比较器正输入管脚
45	30	1	P1.2	**I/O**	通用数字输入/输出管脚
			ADC_CH2	**AI**	ADC 模拟输入管脚
			UART0_RXD	**I**	UART0 数据接收输入管脚
			ACMP0_P2	**AI**	模拟比较器正输入管脚（TSSOP20封装不支持）
			PWM0_CH0	**O**	PWM0 输出管脚
46	31	2	P1.3	**I/O**	通用数字输入/输出管脚
			ADC_CH3	**AI**	ADC 模拟输入管脚
			UART0_TXD	**O**	UART0 数据输出管脚
			ACMP0_P3	**AI**	模拟比较器正输入管脚（TSSOP20封装不支持）
			PWM0_CH1	**O**	PWM1 输出管脚
47	32	3	P1.4	**I/O**	通用数字输入/输出管脚
			ADC_CH4	**I/O**	ADC 模拟输入管脚
			UART1_RXD	**I**	UART1 数据接收输入管脚
			ACMP0_N	**AI**	模拟比较器负输入管脚（TSSOP20封装不支持）
			PWM0_CH4	**O**	PWM4 输出管脚
48	---	--	NC	**---**	无连接

表 4.4-1 NuMicro® Mini58 系列管脚描述

[1] I/O 类型描述. I: 输入, O: 输出, I/O: 准双向, D: 开漏, P: 电源管脚, ST: 施密特触发, A: 模拟输入.

5 方框图

5.1 NuMicro® Mini58 方框图

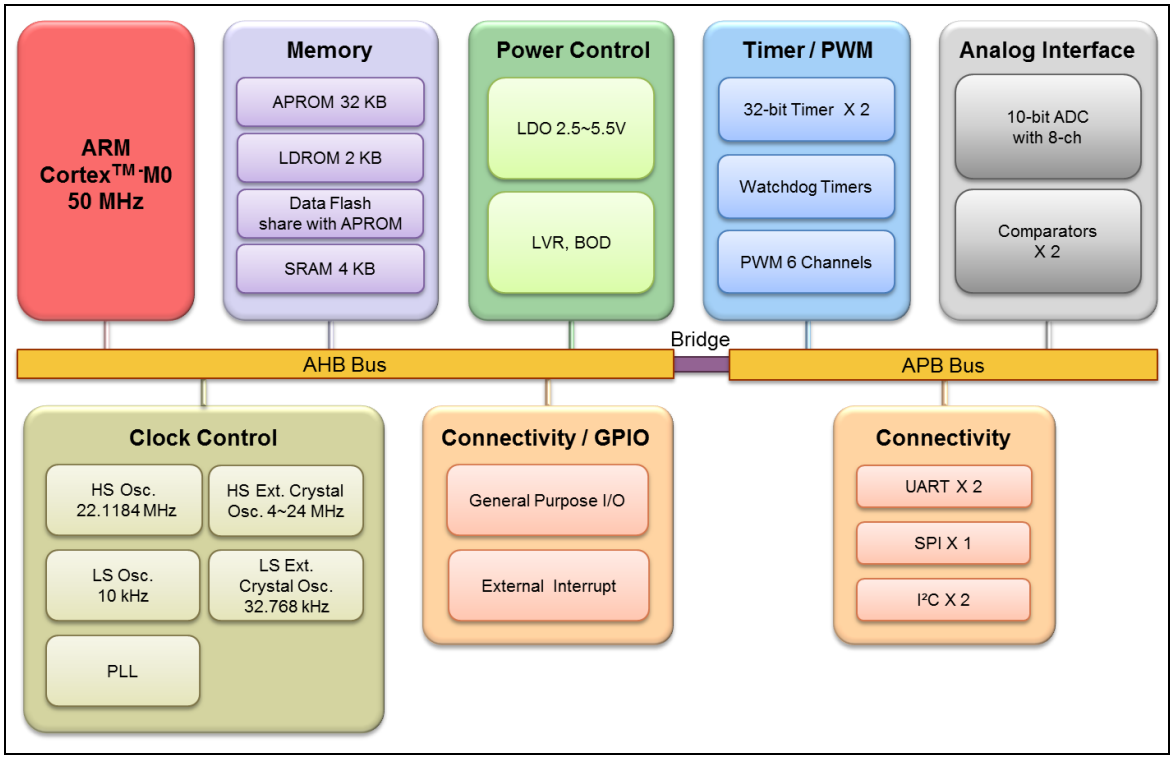


图 5.1-1 NuMicro® Mini58 系列方框图

6 功能描述

6.1 ARM® Cortex®-M0 内核

6.1.1 概述

Cortex®-M0 处理器是一个可配置，具有多级流水线的32位RISC处理器。它拥有一个 AMBA AHB-Lite 接口并包含NVIC组件，同时有可选的硬件调试功能。该处理器可以执行Thumb 指令，并与其他Cortex®-M 系列处理器兼容。该处理器支持两种模式——Thread模式和Handler模式。异常时系统进入Handler模式，异常返回只能在Handler模式下执行。系统复位及异常返回后均可进入Thread模式。 图 6.1-1 展示处理器的功能控制器

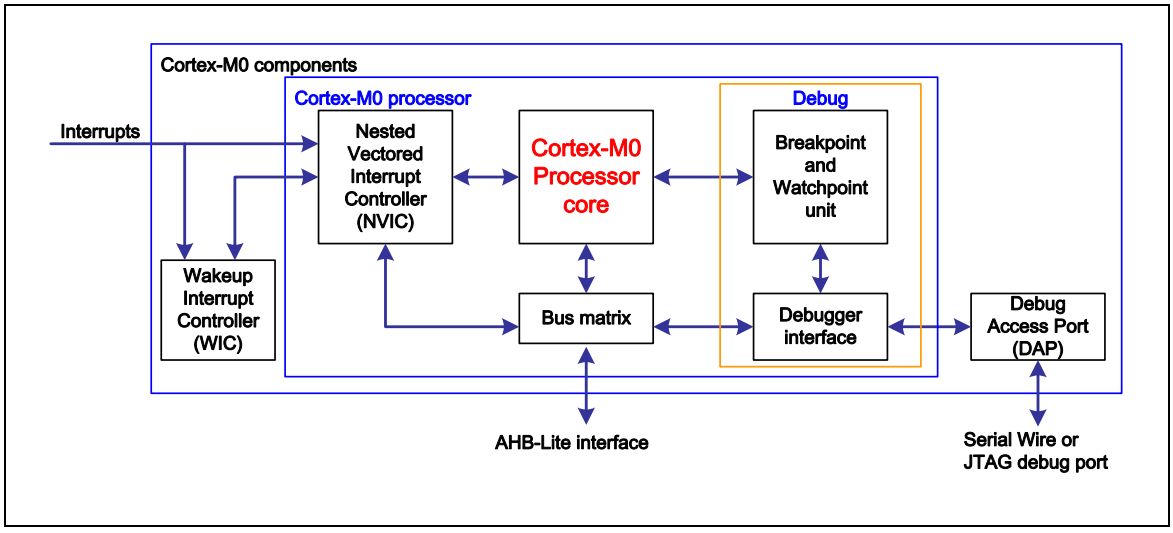


图 6.1-1 功能模块方框图

6.1.2 特性

- 低门数处理器
  - ARMv6-M Thumb® 指令集
  - Thumb-2 技术
  - ARMv6-M 兼容24位系统定时器
  - 一个32位硬件乘法器
  - 系统接口支持小端数据访问
  - 准确而及时的中断处理能力
  - 加载/存储多个数据和多周期乘法指令可被终止，然后重新开始，从而实现快速中断处理
  - C 应用程序二进制接口异常兼容模式：

    ARMv6-M的C应用程序二进制接口(C-ABI)异常兼容模式允许用户在中断处理中使用纯C函数。
  - 使用WFI指令进入低功耗空闲模式，使用WFE指令或从中断退出低功耗空闲模式
- NVIC

- 32个外部中断，每个中断有4级优先级
- 专用的不可屏蔽中断 (NMI)
- 同时支持电平和脉冲触发中断
- 支持中断唤醒控制器(WIC)， 提供极低功耗空闲模式

◆ 调试支持

- 四个硬件断点
- 两个观察点
- 用于非侵入代码分析的程序计数采样寄存器(PCSR)
- 单步和向量捕捉能力

◆ 总线接口

- 为所有的系统接口及存储器提供简单集成的单一的32位AMBA-3 AHB-Lite 系统接口
- 支持DAP（调试使用端口）单一的32位从机端口

6.2 系统管理

6.2.1 概述

系统管理包含以下几个部分：

- 系统复位
- 系统电源分配
- 系统内存映射
- 用于产品ID、芯片复位、片上控制器复位和多功能管脚控制的系统管理寄存器
- 系统定时器 (SysTick)
- 嵌套中断向量控制器 (NVIC)
- 系统控制寄存器

6.2.2 系统复位

系统复位可以由下面列出的任一事件触发。这些复位事件标志可以通过读取SYS_RSTSTS 寄存器来判断复位源。硬件复位可以通过外部的复位信号来复位芯片。软件复位则是通过控制寄存器来触发复位。

- 硬件复位源
  - 上电复位 (POR)
  - nRESET引脚低电平复位 (nRESET)
  - 看门狗定时器溢出复位和窗口看门狗复位 (WDT/WWDT复位)
  - 低电复位(LVR)
  - 欠压检测复位(BOD)
  - CPU锁住复位
- 软件复位源
  - 写1到CHIPRST位(SYS_IPRST0[0])会复位整个芯片
  - 写 1 到 SYSRESETREQ(SCS_AIRCR[2]) 会 触 发 MCU 复 位 重 启， 但 不 影 响 APROM 和 LDROM的启动选项
  - 仅当写1到CPURST(SYS_IPRST0[1])才会触发Cortex®-M0 CPU复位

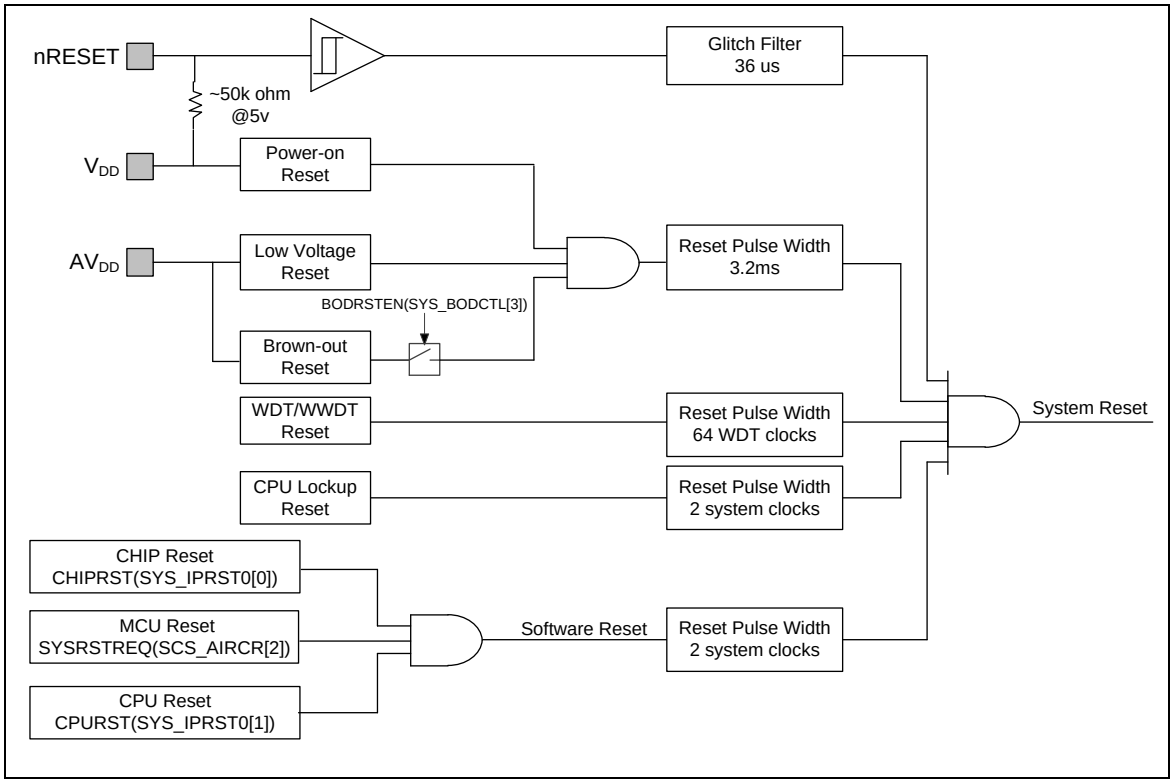


图 6.2-1 系统复位源

NuMicro®系列单片机总共有9个复位源。通常，CPU复位仅用于复位Cortex®-M0内核；其他复位源则会复位Cortex®-M0内核及所有外设。其中，每种复位之间仍有细微的差别，具体可以参见表6.2-5。

复位源 / 寄存器	POR	nRESET	WDT	LVR	BOD	Lockup	CHIP	MCU	CPU
SYS_RSTSTS	0x001	Bit 1 = 1	Bit 2 = 1	0x001	Bit 4 = 1	Bit 8 = 1	Bit 0 = 1	Bit 5 = 1	Bit 7 = 1
CHIPRST (SYS_IPRST0[0])	0x0	-	-	-	-	-	-	-	-
BODEN (SYS_BODCTL[0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-
BODVL (SYS_BODCTL[2:1])									
BODRSTEN (SYS_BODCTL[3])									
XTLEN (CLK_PWRCTL[1:0])	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	
WDTCKEN (CLK_APBCLK0[0])	0x1	-	0x1	-	-	-	0x1	-	-
HCLKSEL (CLK_CLKSEL0[2:0])	0x8	0x8	0x8	0x8	0x8	0x8	0x8	0x8	-
WDTSEL	0x3	0x3	-	-	-	-	-	-	-

(CLK_CLKSEL1[1:0])									
XLTSTB (CLK_STATUS[0])	0x0	-	-	-	-	-	-	-	-
PLLSTB (CLK_STATUS[2])	0x0	-	-	-	-	-	-	-	-
LIRCSTB (CLK_STATUS[3])	0x0								
HIRCSTB (CLK_STATUS[4])	0x0	-	-	-	-	-	-	-	-
CLKSFAIL (CLK_STATUS[7])	0x0	0x0	-	-	-	-	-	-	-
WDT_CTL	0x0700	0x0700	0x0700	0x0700	0x0700	-	0x0700	-	-
WDT_ALTCTL	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_RLDCNT	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CTL	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	-	0x3F0800	-	-
WWDT_STATUS	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CNT	0x3F	0x3F	0x3F	0x3F	0x3F	-	0x3F	-	-
BS (FMC_ISPCTL[1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
ISPEN (FMC_ISPCTL[16])									
FMC_DFBA	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	-	Reload from CONFIG1	-	-
CBS (FMC_ISPSTS[2:1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
VECMAP (FMC_ISPSTS[20:9])	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	-	Reload base on CONFIG0	-	-
Other Peripheral Registers	Reset Value								-
FMC Registers	Reset Value								
Note: '-' means that the value of register keeps original setting.									

表 6.2-1 寄存器复位值

6.2.2.1 nRESET 复位

nRESET复位时指拉低nRESET引脚产生的复位。nRESET复位是一个异步的复位，可以在任何时间用于系统复位。当nRESET引脚电压低于0.2V_{DD}超过36us(消抖时间)，芯片将会复位。nRESET复位将会保持芯片的复位状态直到nRESET引脚电压上升超过0.7V_{DD}且时长超过36us(消抖时间)。当上一个复位源是nRESET复位时，寄存器PINRF(SYS_RSTSTS[1])会被置1。如

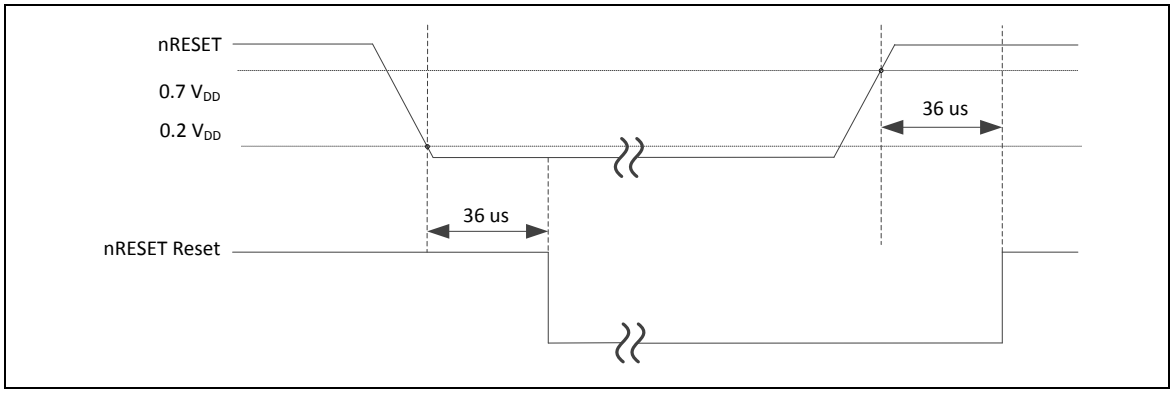


图 6.2-2 是nRESET复位的波形。

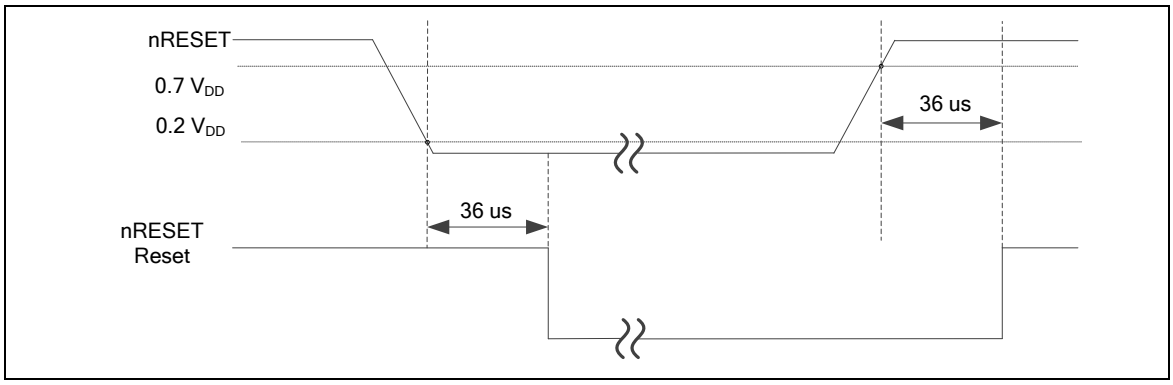


图 6.2-2 nRESET复位波形

6.2.2.2 上电复位 (POR)

上电复位可以产生可靠的系统复位信号从而强制系统复位，以避免单片机异常行为干扰复位。单片机上电时，POR模块会检测上升电压，当电压达到单片机工作电压时，上电复位模块会产生一个复位信号来复位系统。上电复位时，寄存器PORF(SYS_RSTSTS[0])会被置1以显示有一个上电复位事件。向寄存器PORF(SYS_RSTSTS[0])写1可以清除该位。如图 6.2-3 是上电复位时的波形。

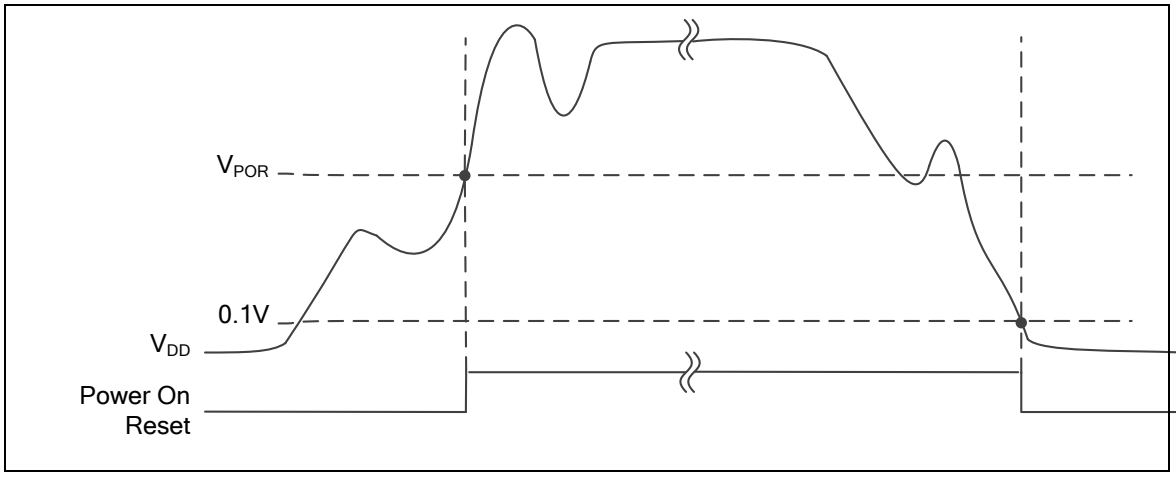


图 6.2-3 上电复位(POR)波形

6.2.2.3 低电复位(LVR)

系统工作时低电复位模块监测AV_{DD}的电压，当AV_{DD}的电压低于V_{LVR}且该状态持续超过消抖时长(16个系统时钟周期(HCLK))，芯片将会复位。低电复位将会保持芯片的复位状态直到AV_{DD}的电压上升超过V_{LVR}且该状态持续超过消抖时长。当上一个复位源是低电复位时，寄存器PINRF (SYS_RSTSTS[1])将会被置1。图 6.2-4 是低电复位的波形图。

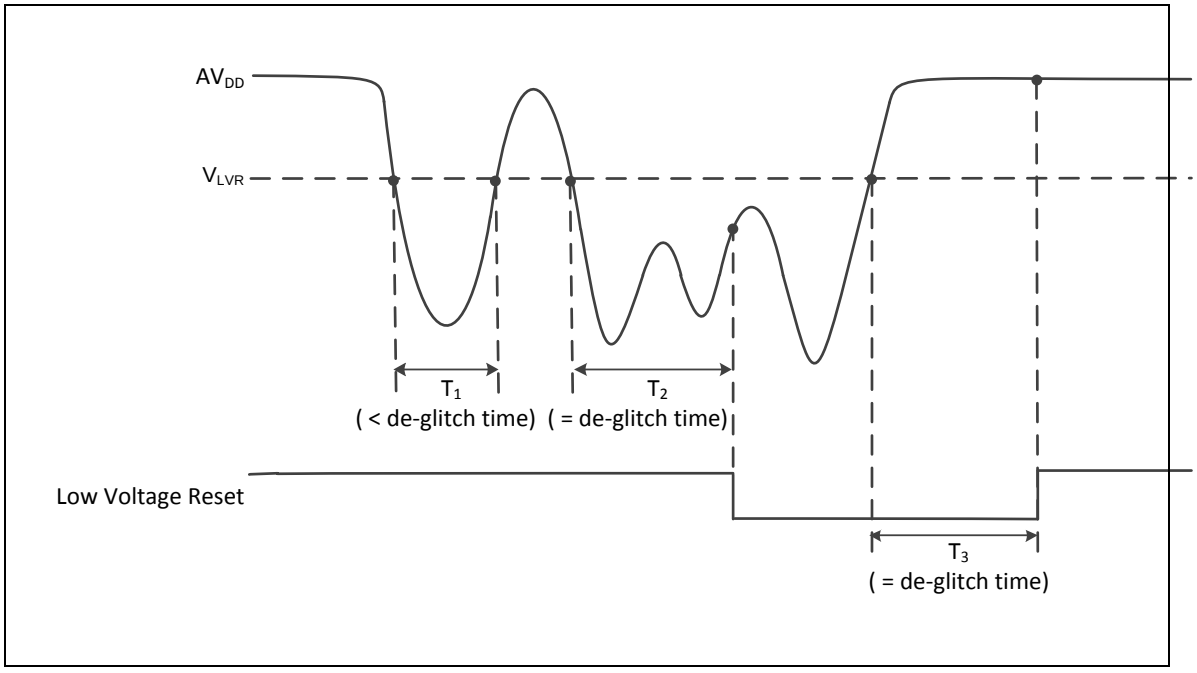


图 6.2-4 低电复位 (LVR) 波形

6.2.2.4 欠压检测复位 (BOD复位)

设置欠电压检测使能位BODEN(SYS_BODCTL[0])打开欠电压检测功能后，系统运行过程中BOD模块会一直检测AV_{DD}电压。当AV_{DD}电压低于预设的V_{BOD}[由寄存器SYS_BODCTL[0]) 及 BODVL (SYS_BODCTL[2:1])决定]且持续超过消抖时间(20个系统时钟(HCLK)周期时长、1个内部低速晶振(LIRC)周期时长间的最大值)，芯片会被复位。BOD复位将会保持芯片复位状态直到AV_{DD}的电压上升超过V_{BOD}并且该状态持续超过消抖时间。BODEN、BODVL和BODRSTEN三个寄存器的默认值分别由存储控制器里的用户配置寄存器CBOVEXT (CONFIG0[23]), CBOV (CONFIG0[22:21]) 和 CBORST (CONFIG0[20])决定。如图图 6.2-5 是BOD复位的波形图。

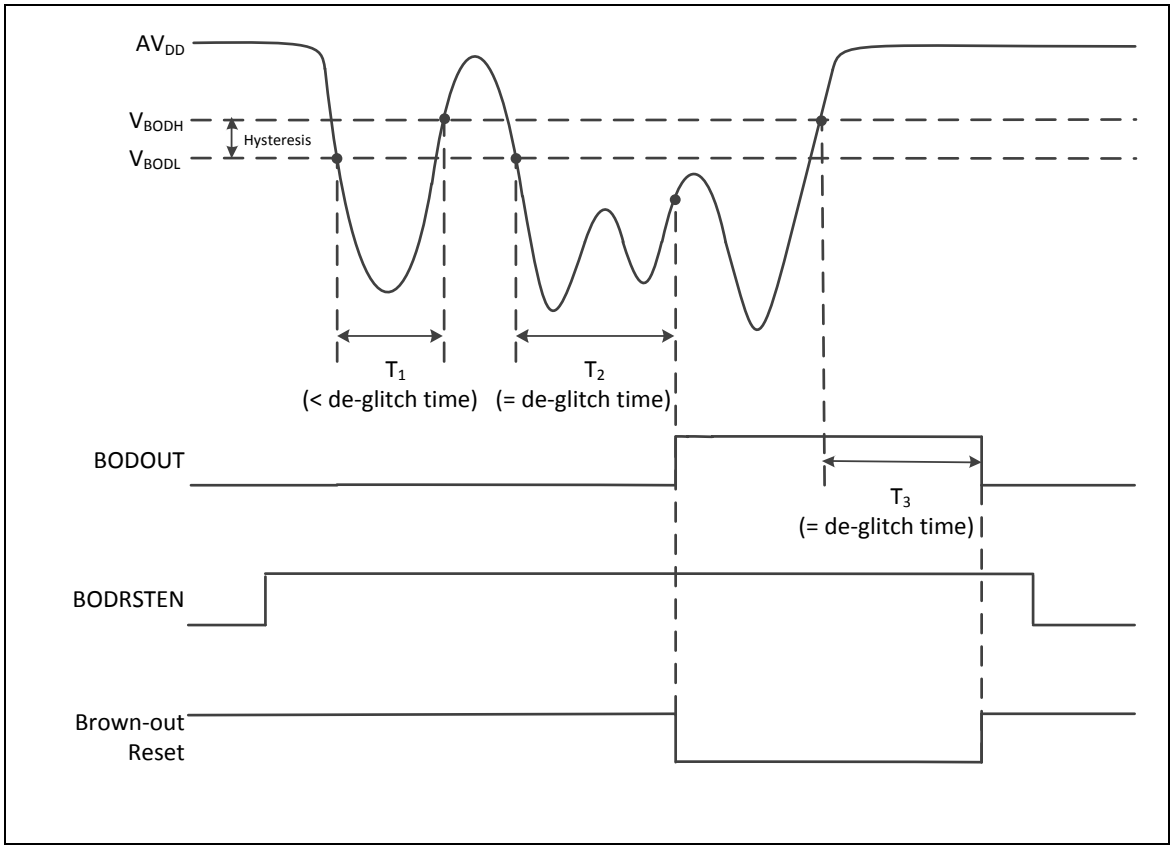


图 6.2-5 欠电压检测(BOD)复位波形图

6.2.2.5 看门狗定时器复位

在大部分工业应用中，系统可靠性是非常重要的。让单片机能从故障状态自动恢复是一个常见的提高系统可靠性的方法。看门狗定时器(WDT)被广泛用来检测系统是否正常工作。当单片机崩溃或者失控时将会引起看门狗定时器溢出。看门狗定时器溢出后，用户可以使能系统复位以使系统恢复正常，并在复位后针对之前的崩溃/失控状态做相应处理。

软件可以通过寄存器WDTRF (SYS_RSTSTS[2])来判断复位是否由看门狗定时器溢出触发，并在复位后针对单片机的错误做相应操作。

6.2.2.6 CPU 锁住复位

在CPU在硬件故障处理(hardfault handler)阶段产生硬件故障(hard fault)之后，CPU会进入锁住状态以表示内核软件正处于严重不稳定的状态。所以遇到不可修复的异常之后，处理器内置的系统状态保护硬件将会激活并锁住CPU。值得注意的是，当芯片处于调试模式下，CPU锁住复位将会被忽略。

6.2.2.7 CPU 复位、 芯片(CHIP)复位 和 系统(SYSTEM)复位

CPU复位是指仅Cortex®-M0内核复位，单片机其他外设的状态在CPU复位之后保持不变。用户可以通过设置寄存器CPURST (SYS_IPRST0[1])位为1来触发CPU复位信号。

芯片(CHIP)复位于上电复位一致。CPU内核和所有外设都会复位，寄存器BS(FMC_ISPCTL[1])会自动装载用户配置里的值。用户可以通过设置寄存器CHIPRST (SYS_IPRST0[0])位为1来触发CHIP复位信号。

单片机(MCU)复位与芯片(CHIP)复位类似。两者的不同点是单片机(MCU)复位中寄存器BS(FMC_ISPCTL[1])不会自动装载用户配置里的值，而会保持之前软件里设定的从APROM或是从LDROM启动。用户可以通过设置寄存器SYSRESETREQ (SCS_AIRCR[2])位为1来触发单片机(MCU)复位信号。

6.2.3 电源模式和唤醒源

当单片机处于空闲模式或者掉电模式下可以由多种唤醒源来唤醒单片机。表 6.2-2 列出了不同电源模式下可用的时钟。

电源模式	正常模式	空闲模式	掉电模式
定义	CPU处于激活的状态	CPU处于休眠的状态	CPU处于休眠的状态。除LXT 、 LIRC和SRAM 保留，所有时钟停止
进入条件	系统复位完成之后芯片将会处于正常模式	CPU执行WFI命令	CPU使能睡眠模式和掉电模式后执行WFI指令
唤醒源	N/A	所有中断	WDT, I²C, 定时器(Timer), UART, BOD 和 GPIO
可用的时钟	所有时钟	除了CPU时钟外的所有时钟	外部低速晶振(LXT)和内部低速振荡器(LIRC)
唤醒后	N/A	CPU恢复到正常模式	CPU恢复到正常模式

表 6.2-2 电源模式详解

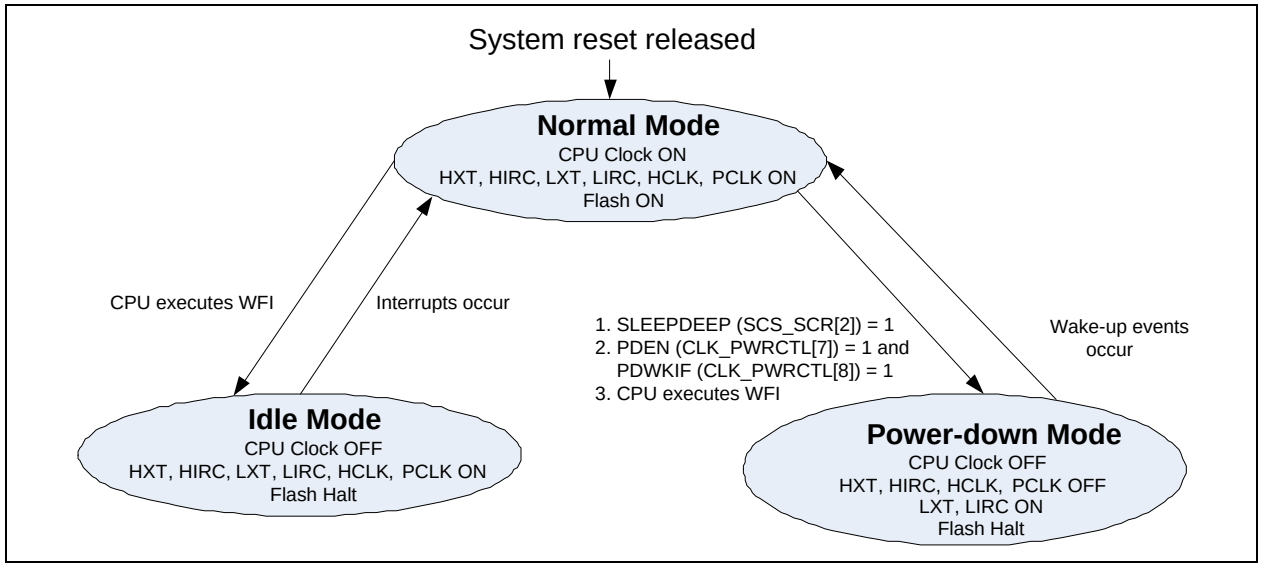


图 6.2-6 电源模式状态机

	正常模式	空闲模式	掉电模式
HXT (4~20 MHz XTL)	ON	ON	Halt
HIRC (12/16 MHz OSC)	ON	ON	Halt
LXT (32768 Hz XTL)	ON	ON	ON/OFF[1]
LIRC (10 kHz OSC)	ON	ON	ON/OFF[2]
PLL	ON	ON	Halt
LDO	ON	ON	ON
CPU	ON	Halt	Halt
HCLK/PCLK	ON	ON	Halt
SRAM retention	ON	ON	ON
FLASH	ON	ON	Halt
GPIO	ON	ON	Halt
TIMER	ON	ON	ON/OFF[3]
PWM	ON	ON	Halt
WDT	ON	ON	ON/OFF[4]
WWDT	ON	ON	Halt
UART	ON	ON	Halt
I^2C	ON	ON	Halt
SPI	ON	ON	Halt
ADC	ON	ON	Halt
ACMP	ON	ON	Halt

表 6.2-3 不同电源模式下的时钟情况

1. LXT (32768 Hz XTL) ON 或 OFF 取决于软件设置的运行模式.
2. LIRC (10 kHz OSC) ON or OFF 取决于软件设置的运行模式.
3. 如果TIMER的时钟源是LIRC/LXT那么LIRC/LXT ON.
4. 如果WDT的时钟源是LIRC那么LIRC ON.

掉电模式下的唤醒源：

WDT, I^2C, Timer, UART, BOD 及 GPIO

芯片进入掉电模式后表 6.2-4所示的唤醒源可以唤醒芯片到正常模式。表 6.2-4还列出了各外设再次进入掉电模式的条件。

唤醒源	唤醒条件	系统再次进入掉电模式的条件*
BOD	Brown-Out Detector 中断	软件写 1 清除 SYS_BODCTL[BODIF]位后

GPIO	GPIO中断	软件写 1 清除Px_INTSRC[n] 位后
TIMER	Timer中断	软件写 1 清除 TWKF (TIMERx_INTSTS[1]) 和 TIF (TIMERx_INTSTS[0])位后
WDT	WDT中断	软件写 1 清除WKF (WDT_CTL[5])位后(写保护位)
UART	nCTS 唤醒	软件写 1 清除 CTSWKIF (UARTx_INTSTS[16])位后
I^2C	I2C_SDA 或 I2C_CLK的下降沿	软件写 1 清除WKIF (I2C_STATUS1[0])位后

表 6.2-4 各外设再次进入掉电模式的条件

*在设置 PDEN (CLK_PWRCTL[7])位和执行WFI以进入掉电模式前，用户需要等待这些条件完成。

6.2.4 系统电源结构

本芯片的电源分为三个部分。

- 由AV_{DD}和AV_{SS}提供的模拟电源，为芯片模拟部分提供工作电压。AV_{DD} 与V_{DD} 电压值必须相等，以防止电流漏电现象。
- 由V_{DD}和V_{SS}提供的数字电源，为I/O管脚及用于数字运算的1.8V内部线性电源提供工作电压
- 内置一个电容用于内部稳压器

内部稳压器输出管脚LDO_CAP要求外接一个电容，并尽量靠近引脚摆放。模拟电源(AV_{DD})与数字电源(V_{DD})电压等级要相同。图 6.2-77 展示了 Mini58系列的电源分配。

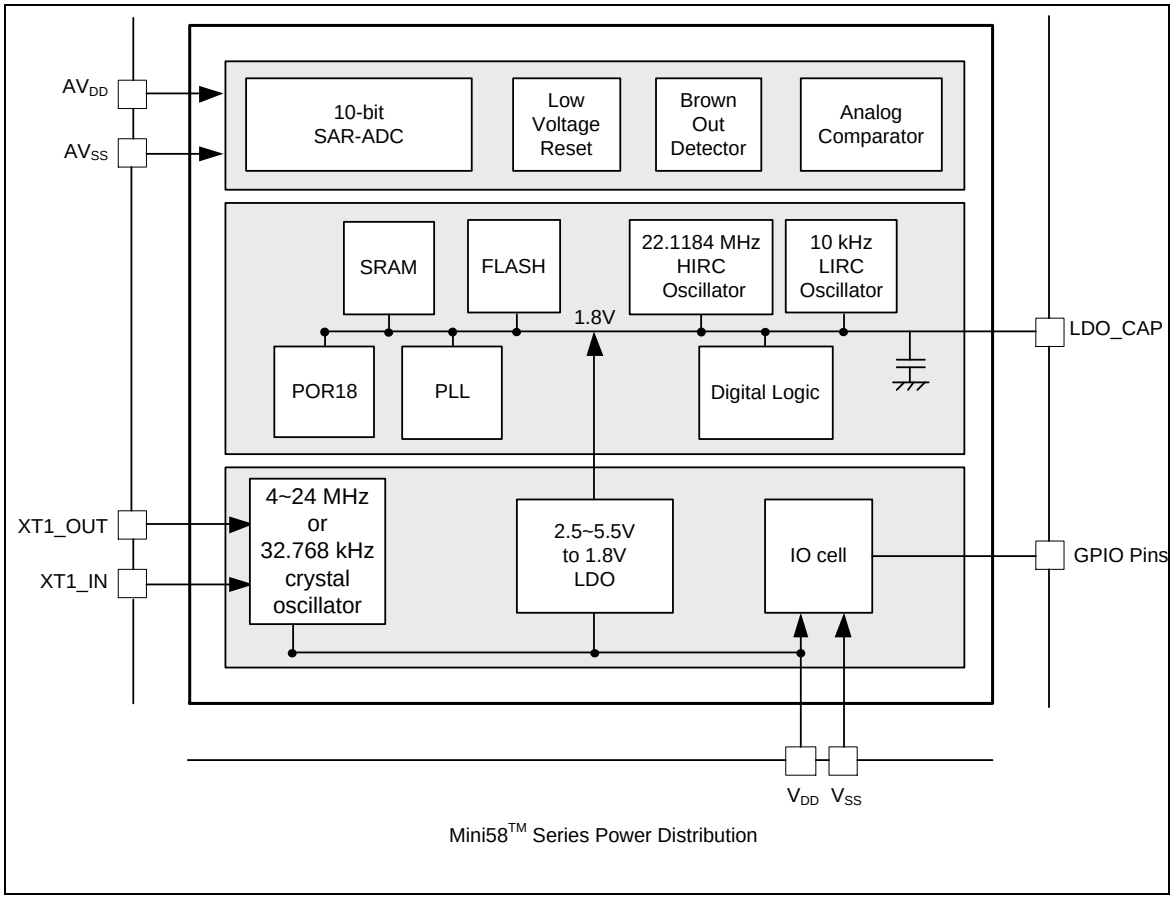


图 6.2-77 NuMicro® Mini58 系列电源结构框图

6.2.5 系统内存映射

	Mini58	
4 GB	Reserved	0xFFFF_FFFF | 0xE000_F000
	System Control	0xE000_EFFF 0xE000_E000
	Reserved	0xE000_E00F | 0x6002_0000
	Reserved	0x6001_FFFF 0x6000_0000
	Reserved	0x5FFF_FFFF | 0x5020_0000
	AHB	0x501F_FFFF 0x5000_0000
	Reserved	0x4FFF_FFFF | 0x4020_0000
	APB	0x401F_FFFF | 0x4000_0000
1 GB	Reserved	0x3FFF_FFFF | 0x2000_1000
	4 KB SRAM	0x2000_0FFF 0x2000_0000
0.5 GB	Reserved	0x1FFF_FFFF | 0x0000_8000
	32 KB on-chip Flash (Mini58)	0x0000_7FFF | 0x0000_0000
0 GB		

System Control

System Control	0xE000_ED00	SCS_BA
External Interrupt Control	0xE000_E100	SCS_BA
System Timer Control	0xE000_E010	SCS_BA

AHB peripherals

FMC	0x5000_C000	FMC_BA
GPIO Control	0x5000_4000	GP_BA
Interrupt Multiplexer Control	0x5000_0300	INT_BA
Clock Control	0x5000_0200	CLK_BA
System Global Control	0x5000_0000	SYS_BA

APB peripherals

UART1 Control	0x4015_0000	UART1_BA
I^2C1 Control	0x4012_0000	I2C1_BA
ADC Control	0x400E_0000	ADC_BA
ACMP Control	0x400D_0000	CMP_BA
UART0 Control	0x4005_0000	UART0_BA
PWM Control	0x4004_0000	PWM_BA
SPI Control	0x4003_0000	SPI_BA
I^2C0 Control	0x4002_0000	I2C0_BA
Timer0/Timer1 Control	0x4001_0000	TMR_BA
WWDT Control	0x4000_4100	WWDT_BA
WDT Control	0x4000_4000	WDT_BA

表 6.2-5 内存映射表

6.2.6 内存组织

6.2.6.1 概述

NuMicro® Mini58 系列提供4GB地址空间，分配给片上控制器的地址空间如下表所示。每个片上外设详细的寄存器定义、地址空间和编程细节将在接下来的章节中描述。Mini58系列只支持小端数据格式。

6.2.6.2 系统内存映射

片上控制器的内存位置分配如下表所示。

地址空间	符号	模块
Flash and SRAM Memory Space		
0x0000_0000 – 0x0000_7FFF	FLASH_BA	Flash存储空间 (32 KB)
0x2000_0000 – 0x2000_0FFF	SRAM_BA	SRAM存储空间 (4 KB)
AHB模块空间 (0x5000_0000 – 0x501F_FFFF)		
0x5000_0000 – 0x5000_01FF	SYS_BA	系统全局控制寄存器
0x5000_0200 – 0x5000_02FF	CLK_BA	时钟控制寄存器
0x5000_0300 – 0x5000_03FF	INT_BA	多路中断控制寄存器
0x5000_4000 – 0x5000_7FFF	GP_BA	GPIO (P0~P5) 控制寄存器
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash内存控制寄存器
APB模块空间 (0x4000_0000 – 0x401F_FFFF)		
0x4000_4000 – 0x4000_00FF	WDT_BA	看门狗定时器控制寄存器
0x4000_4100 – 0x4000_47FF	WWDT_BA	窗口看门狗定时器控制寄存器
0x4001_0000 – 0x4001_3FFF	TMR_BA	Timer0/Timer1控制寄存器
0x4002_0000 – 0x4002_3FFF	I2C0_BA	I^2C0 接口控制寄存器
0x4003_0000 – 0x4003_3FFF	SPI_BA	带主/从机功能的SPI控制寄存器
0x4004_0000 – 0x4004_3FFF	PWM_BA	PWM控制寄存器
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0控制寄存器
0x400D_0000 – 0x400D_3FFF	ACMP_BA	模拟比较器控制寄存器
0x400E_0000 – 0x400E_3FFF	ADC_BA	模数转换器 (ADC)控制寄存器
0x4012_0000 – 0x4012_3FFF	I2C1_BA	I^2C1接口控制寄存器
0x4015_0000 – 0x4015_3FFF	UART1_BA	UART1控制寄存器
系统控制器空间 (0xE000_E000 – 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	系统定时器控制寄存器
0xE000_E100 – 0xE000_ECFF	SCS_BA	嵌套中断向量控制寄存器
0xE000_ED00 – 0xE000_ED8F	SCS_BA	系统控制模块寄存器

表 6.2-6 片上模块地址空间分配

6.2.7 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
SYS 基地址: **SYS_BA = 0x5000_0000**				
SYS_PDID	SYS_BA+0x00	R	器件ID寄存器	0xXXXX_XXXX[1]
SYS_RSTSTS	SYS_BA+0x04	R/W	系统复位状态寄存器	0x0000_00XX
SYS_IPRST0	SYS_BA+0x08	R/W	外设复位控制寄存器0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	外设复位控制寄存器1	0x0000_0000
SYS_BODCTL	SYS_BA+0x18	R/W	掉电检测控制寄存器	0x0000_000X
SYS_P0_MFP	SYS_BA+0x30	R/W	P0 复用功能和输入类型控制寄存器	0x0000_0000
SYS_P1_MFP	SYS_BA+0x34	R/W	P1复用功能和输入类型控制寄存器	0x0000_0000
SYS_P2_MFP	SYS_BA+0x38	R/W	P2 复用功能和输入类型控制寄存器	0x0000_0000
SYS_P3_MFP	SYS_BA+0x3C	R/W	P3 复用功能和输入类型控制寄存器	0x0000_0000
SYS_P4_MFP	SYS_BA+0x40	R/W	P4复用功能和输入类型控制寄存器	0x0000_00C0
SYS_P5_MFP	SYS_BA+0x44	R/W	P5复用功能和输入类型控制寄存器	0x0000_0000
SYS_IRCTCTL	SYS_BA+0x80	R/W	HIRC校准控制寄存器	0x0000_0000
SYS_IRCTIEN	SYS_BA+0x84	R/W	HIRC校准中断使能寄存器	0x0000_0000
SYS_IRCTISTS	SYS_BA+0x88	R/W	HIRC校准中断状态寄存器	0x0000_0000
SYS_REGLCTL	SYS_BA+0x100	R/W	寄存器写保护控制寄存器	0x0000_0000

6.2.8 寄存器描述

器件ID寄存器 (SYS_PDID)

寄存器	偏移地址	R/W	描述	复位值
SYS_PDID	SYS_BA+0x00	R	器件ID寄存器	0xXXXX_XXXX[1]

[1] 每个器件具有一个独一无二的默认复位值

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

位	描述	
[31:0]	**PDID**	**产品器件识别码 (只读)** 该寄存器为设备的器件号码。软件可以通过读取该寄存器来识别所使用的器件。 例如：MINI58LDE PDID 码是 “0x00A05800”.

NuMicro® Mini58 系列	**设备器件识别码**
MINI58LDE	0x00A05800
MINI58ZDE	0x00A05803
MINI58TDE	0x00A05804
MINI58FDE	0x00A05805

系统复位状态寄存器 (SYS_RSTSTS)

该寄存器提供一些信息用于软件识别上次引起芯片复位的复位源。

寄存器	偏移地址	R/W	描述	复位值
SYS_RSTSTS	SYS_BA+0x04	R/W	系统复位状态寄存器	0x0000_00XX

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

位	描述	
[31:9]	**Reserved**	保留
[8]	**CPULKRF**	**Cortex®-M0 锁住标志** 0 = 没有发生Cortex®-M0 锁住而导致复位 1 = Cortex®-M0锁住，芯片复位 **注:** 软件可向该位写1来清0。
[7]	**CPURF**	**CPU复位标志** 如果软件写1到CPURST (SYS_IPRST0[1]) 来复位Cortex®-M0内核和Flash内存控制器(FMC)，硬件会自动置位该CPU复位标志位。 0 = CPU无复位 1 = Cortex®-M0 内核和FMC因为软件置CPURS为1而复位。 **注:** 软件可向该位写1来清0。
[6]	**Reserved**	保留
[5]	**SYSRF**	**系统复位标志** 此系统复位标志由来自Cortex®-M0内核的“复位信号”置位，用于表示导致之前复位的复位源。 0 = Cortex®-M0无复位 1 = Cortex®-M0因为软件向SYSRESETREQ (SCS_AIRCR[2])位写1而发出复位信号，从而导致系统复位。AIRCR是应用中断和复位控制寄存器，其地址为0xE000ED0C，位于Cortex®-M0 内核系统控制寄存器中。 **注:** 软件可向该位写1来清0。

位	描述	
[4]	**BODRF**	**BOD复位标志** BOD复位标志由欠压检测模块的“复位信号”置位，用于表示导致之前复位的信号源。 0 = BOD无复位。 1 = BOD发出复位信号使系统复位。 **注:** 软件可向该位写1来清0。
[3]	**Reserved**	保留
[2]	**WDTRF**	**WDT复位标志** WDT标志由看门狗定时器或窗口看门狗定时器的“复位信号”置位，用于表示导致之前复位的复位源。 0 = 看门狗定时器或窗口看门狗定时器无复位。 1 = 看门狗定时器或窗口看门狗定时器发出复位信号使系统复位。 **注:** 软件可向该位写1来清0。
[1]	**PINRF**	**NRESET管脚复位标志** nRESET管脚复位标志由nRESET管脚的“复位信号”置位，用于表示导致之前复位的复位源。 0 = nRESET管脚无复位 1 = nRESET管脚发出复位信号使系统复位 **注:** 软件可向该位写1来清0。
[0]	**PORF**	**POR复位标志** POR复位标志由上电复位 (POR) 控制器或CHIPRST (SYS_IPRST0[0])位发出“复位信号”置位，用于表示导致之前复位的复位源。 0 = POR或CHIPRST无复位 1 = 上电复位(POR)或CHIPRST发出复位信号使系统复位 **注:** 软件可向该位写1来清0。

外设复位控制寄存器 0 (SYS_IPRST0)

寄存器	偏移地址	R/W	描述	复位值
SYS_IPRST0	SYS_BA+0x08	R/W	外设复位控制寄存器 0	0x0000_0000

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

位	描述	
[31:2]	**Reserved**	保留
[1]	**CPURST**	**处理器内核一次性复位(写保护)** 置该位为1将只复位处理器内核和Flash内存控制器(FMC)，该位将在两个时钟周期后自动清零。 0 = 处理器内核正常运行 1 = 处理器内核一次性复位 **注:** 该位为写写保护位，请参考SYS_REGLCTL寄存器。
[0]	**CHIPRST**	**芯片一次性复位 (写保护)** 置该位为1将复位整个芯片，包括处理器内核和所有外设，该位将在2个时钟周期后自动清零。 CHIPRST与POR复位一样，所有芯片控制器都复位，芯片设置从flash重新加载。 0 = 芯片正常运行 1 = 芯片一次性复位 **注:** 该位为写保护位，请参考SYS_REGLCTL寄存器。

外设复位控制寄存器1 (SYS_IPRST1)

设置这些位为1将产生异步复位信号给相应的模块控制器，用户需要设置这些位为0，才能将相应模块控制器从复位状态中释放。

寄存器	偏移地址	R/W	描述	复位值
SYS_IPRST1	SYS_BA+0x0C	R/W	外设复位控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			ADCRST	Reserved			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	ACMPRST	Reserved	PWM0RST	Reserved		UART1RST	UART0RST
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			SPI0RST	Reserved		I2C1RST	I2C0RST
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				TMR1RST	TMR0RST	GPIORST	Reserved

位	描述	
[31:29]	**Reserved**	保留
[28]	**ADCRST**	**ADC控制器复位** 0 = ADC控制器正常运行 1 = ADC控制器复位
[27:23]	**Reserved**	保留
[22]	**ACMPRST**	**ACMP控制器复位** 0 = ACMP控制器正常运行 1 = ACMP控制器复位
[21]	**Reserved**	保留
[20]	**PWM0RST**	**PWM0控制器复位** 0 = PWM0控制器正常运行 1 = PWM0控制器复位
[19:18]	**Reserved**	保留
[17]	**UART1RST**	**UART1控制器复位** 0 = UART1控制器正常运行 1 = UART1控制器复位
[16]	**UART0RST**	**UART0控制器复位** 0 = UART0控制器正常运行 1 = UART0控制器复位
[15:13]	**Reserved**	保留

位	描述	
[12]	**SPI0RST**	**SPI0控制器复位** 0 = SPI控制器正常运行 1 = SPI控制器复位
[11:10]	**Reserved**	保留
[9]	**I2C1RST**	**I^2C1控制器复位** 0 = I^2C1控制器正常运行 1 = I^2C1控制器复位
[8]	**I2C0RST**	**I^2C0控制器复位** 0 = I^2C0控制器正常运行 1 = I^2C0控制器复位
[7:4]	**Reserved**	保留
[3]	**TMR1RST**	**Timer1控制器复位** 0 = Timer1控制器正常运行 1 = Timer1控制器复位
[2]	**TMR0RST**	**Timer0控制器复位** 0 = Timer0控制器正常运行 1 = Timer0控制器复位
[1]	**GPIORST**	**GPIO (P0~P5)控制器复位** 0 = GPIO控制器正常运行 1 = GPIO控制器复位
[0]	**Reserved**	保留

欠压检测器控制寄存器 (SYS_BODCTL)

SYS_BODCTL控制寄存器部分位的值由flash配置初始化，且为写保护位。假如用户需要编程写保护位的内容，需要一个解锁时序。解锁时序是依次连续的往地址为0x5000_0100的控制器写入0x59、 0x16、0x88。在写以上三个数据的过程中，写入别的数据或其他写操作都将终止解锁时序。

解锁时序后，用户可以查看地址0x5000_0100的位0，1表示没锁定，0表示锁定。没锁定的情况下用户可以更新写保护寄存器的值。写任何其他数据到地址0x5000_0100将再次锁定写保护寄存器。

寄存器	偏移地址	R/W	描述	复位值
SYS_BODCTL	SYS_BA+0x18	R/W	欠压检测器控制寄存器	0x0000_000X

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

位	描述	
[31:7]	**Reserved**	保留
[6]	**BODOUT**	**欠压检测器输出状态** 0 = 欠压检测器输出0，表示检测到的电压比BODVL设置的高 1 = 欠压检测器输出1，表示检测到的电压比BODVL设置的低
[5]	**BODLPM**	**欠压检测器低功耗模式 (写保护)** 0 = BOD运行在正常模式 (默认) 1 = BOD低功耗模式使能 **注:** BOD在正常模式消耗差不多100uA电流，低功耗模式将大约减少电流到1/10，但将减慢BOD的响应速度。
[4]	**BODIF**	**欠压检测器中断标志位** 0 = 欠压检测器没有检测到V_{DD} 曾经下降或上升到BODVL设定的值 1 = 当检测到VDD 下降或者上升到BODVL 设定的值，该位将置1，如果欠压中断使能，则欠压中断将发生

<table>
<tr><th>位</th><th colspan="2">描述</th></tr>
<tr><td>[3]</td><td>BODRSTEN</td><td>欠压复位使能位 (写保护)
默认值由用户配置flash控制寄存器的CBORST(CONFIG0[20]) 位设置。
0 = 欠压“中断”功能使能，当欠压检测器功能使能且检测到电压低于设定的值时，将产生一个中断信号到Cortex®-M0 CPU 。
1 = 欠压“复位”功能使能，当欠压检测器功能使能且检测到电压低于设定的值时，将产生一个信号来复位芯片。
注: 当BOD_EN位使能，中断到来后将一直保持直到将BOD_EN位清0。可以通过失能NVIC或通过失能BOD_EN位来失能BOD中断源，如果需要BOD功能的话，可重新使能BOD_EN位。</td></tr>
<tr><td>[2:1]</td><td>BODVL</td><td>欠压检测器阈电压选择 (写保护)
默认值由用户配置的flash控制寄存器的CBOV (CONFIG0[22:21])位设置。
00 = 保留
01 = 欠压检测器阈电压值为2.7V
10 = 欠压检测器阈电压值为3.7V
11 = 欠压检测器失效</td></tr>
<tr><td>[0]</td><td>BODEN</td><td>欠压检测器选择扩展 (写保护位)
默认值由用户配置的flash控制器的寄存器config0位[23]设置。
如果config0位[23]设置为1， BODEN默认值为0。
如果config0位[23] 设置为0， BODEN默认值为 1。
0 = 欠压检测器阈电压值从BODVL定义的表中选择。
1 = 欠压检测器阈电压值从下表中选择。
<table>
<tr><th>BODVL[1]</th><th>BODVL[0]</th><th>欠压电压值</th></tr>
<tr><td>1</td><td>1</td><td>4.4V</td></tr>
<tr><td>1</td><td>0</td><td>3.7V</td></tr>
<tr><td>0</td><td>1</td><td>2.7V</td></tr>
<tr><td>0</td><td>0</td><td>2.2V</td></tr>
</table></td></tr>
</table>

Port0复用功能控制寄存器 (SYS_P0_MFP)

寄存器	偏移地址	R/W	描述	复位值
SYS_P0_MFP	SYS_BA+0x30	R/W	P0复用功能和输入类型控制寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留
[23:16]	**TYPE[n]**	**P0[7:0]施密特触发输入功能使能位** 0 = P0[7:0] I/O施密特触发输入功能失能 1 = P0[7:0] I/O施密特触发输入功能使能
[15]	**ALT[7]**	**P0.7功能选择位** 位ALT[7] (SYS_P0_MFP[15])和 MFP[7] (SYS_P0_MFP[7]) 决定P0.7 功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择SPI0_CLK功能 (1, 1) = 选择PWM0_CH0功能
[14]	**ALT[6]**	**P0.6 功能选择位** 位ALT[6] (SYS_P0_MFP[14])和MFP[6] (SYS_P0_MFP[6])决定 P0.6功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择SPI0_MISO功能 (1, 1) = 选择PWM0_CH1功能
[13]	**ALT[5]**	**P0.5功能选择位** 位 ALT[5] (SYS_P0_MFP[13])和MFP[5] (SYS_P0_MFP[5])决定P0.5功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择SPI0_MOSI功能 (1, 1) = 选择PWM0_CH4功能

位	描述	
[12]	**ALT[4]**	**P0.4功能选择位** 位ALT[4] (SYS_P0_MFP[12])和 MFP[4] (SYS_P0_MFP[4])决定P0.4功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择SPI0_SS功能 (1, 1) = 选择PWM0_CH5功能
[11:10]	**Reserved**	保留
[9]	**ALT[1]**	**P0.1功能选择位** 位ALT[1] (SYS_P0_MFP[9])和MFP[1] (SYS_P0_MFP[1])决定P0.1功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择 SPI0_SS功能 (1, 0) = 选择UART0_nRTS功能 (1, 1) = 选择UART0_RXD功能
[8]	**ALT[0]**	**P0.0功能选择位** 位ALT[0] (SYS_P0_MFP[8])和 MFP[0] (SYS_P0_MFP[0]) 决定P0.0功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择UART0_nCTS功能 (1, 1) = 选择UART0_TXD功能
[7:0]	**MFP[7:0]**	**P0复用功能选择位** P0的引脚功能取决于MFP和 ALT 详细信息参考ALT描述

Port1复用功能控制寄存器 (SYS_P1_MFP)

寄存器	偏移地址	R/W	描述	复位值
SYS_P1_MFP	SYS_BA+0x34	R/W	P1复用功能和输入类型控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			**P14EXT**	**P13EXT**	**P12EXT**	**Reserved**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TYPE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ALT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFP							

位	描述	
[31:29]	**Reserved**	保留
[28]	**P14EXT**	**P1.4功能选择扩展** 位P14EXT (SYS_P1_MFP[28])、 ALT[4] (SYS_P1_MFP[12])和MFP[4] (SYS_P1_MFP[4])决定 P1.4功能 (0, 0, 0) = 选择 GPIO功能 (0, 0, 1) = 选择ADC_CH4功能 (0, 1, 0) = 选择UART1_RXD功能 (0, 1, 1) =选择 ACMP0_N功能 (1, 0, 0) = 选择 PWM0_CH4功能
[27]	**P13EXT**	**P1.3 功能选择扩展** 位P13EXT (SYS_P1_MFP[27])、 ALT[3] (SYS_P1_MFP[11])和MFP[3] (SYS_P1_MFP[3])决定P1.3功能 (0, 0, 0) = 选择GPIO功能 (0, 0, 1) = 选择ADC_CH3功能 (0, 1, 0) = 选择UART0_TXD功能 (0, 1, 1) = 选择ACMP0_P3功能 (1, 0, 0) = 选择PWM0_CH1功能
[26]	**P12EXT**	**P1.2 功能选择扩展** 位P12EXT (SYS_P1_MFP[26])、ALT[2] (SYS_P1_MFP[10])和MFP[2] (SYS_P1_MFP[2])决定P1.2功能 (0, 0, 0) = 选择GPIO功能 (0, 0, 1) = 选择ADC_CH2功能 (0, 1, 0) = 选择UART0_RXD功能 (0, 1, 1) = 选择ACMP0_P2功能 (1, 0, 0) = 选择PWM0_CH0功能

位	描述	
[25:24]	**Reserved**	保留
[23:16]	**TYPE[n]**	**P1[7:0]施密特触发输入功能使能位** 0 = P1[7:0] I/O施密特触发输入功能失能 1 = P1[7:0] I/O施密特触发输入功能使能
[15:14]	**Reserved**	保留
[13]	**ALT[5]**	**P1.5功能选择位** 位ALT[5] (SYS_P1_MFP[13])和MFP[5] (SYS_P1_MFP[5])决定 P1.5 功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择ADC_CH5功能 (1, 0) = 选择UART1_TXD功能 (1, 1) = 选择ACMP0_P0功能
[12]	**ALT[4]**	**P1.4功能选择位** 位P14EXT (SYS_P1_MFP[28])、ALT[4] (SYS_P1_MFP[12])和MFP[4] (SYS_P1_MFP[4])决定P1.4功能 (0, 0, 0) = 选择GPIO功能 (0, 0, 1) = 选择ADC_CH4功能 (0, 1, 0) = 选择UART1_RXD功能 (0, 1, 1) = 选择ACMP0_N功能 (1, 0, 0) = 选择PWM0_CH4功能
[11]	**ALT[3]**	**P1.3功能选择位** 位P13EXT (SYS_P1_MFP[27])、ALT[3] (SYS_P1_MFP[11])和 MFP[3] (SYS_P1_MFP[3])决定 P1.3 功能 (0, 0, 0) = 选择GPIO功能 (0, 0, 1) = 选择ADC_CH3功能 (0, 1, 0) = 选择UART0_TXD功能 (0, 1, 1) = 选择ACMP0_P3功能 (1, 0, 0) = 选择PWM0_CH1功能
[10]	**ALT[2]**	**P1.2功能选择位** 位P12EXT (SYS_P1_MFP[26])、 ALT[2] (SYS_P1_MFP[10])和MFP[2] (SYS_P1_MFP[2])决定P1.2功能 (0, 0, 0) = 选择GPIO功能 (0, 0, 1) = 选择ADC_CH2功能 (0, 1, 0) = 选择UART0_RXD功能 (0, 1, 1) = 选择ACMP0_P2功能 (1, 0, 0) = 选择PWM0_CH0功能
[9]	**Reserved**	保留

位	描述	
[8]	**ALT[0]**	**P1.0功能选择位** 位ALT[0] (SYS_P1_MFP[8])和MFP[0] (SYS_P1_MFP[0])决定P1.0功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择ADC_CH1功能 (1, 0) = 保留 (1, 1) = 选择ACMP0_P1功能
[7:0]	**MFP[7:0]**	**P1复用功能选择位** P1引脚功能取决于MFP和ALT 细节请参考ALT描述

Port2复用功能控制寄存器(SYS_P2_MFP)

寄存器	偏移地址	R/W	描述	复位值
SYS_P2_MFP	SYS_BA+0x38	R/W	P2复用功能和输入类型控制寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留
[23:16]	**TYPE[n]**	**P2[7:0]施密特触发输入功能使能位** 0 = P2[7:0] I/O施密特触发输入功能失能 1 = P2[7:0] I/O施密特触发输入功能使能
[15]	**Reserved**	保留
[14]	**ALT[6]**	**P2.6功能选择位** 位ALT[6] (SYS_P2_MFP[14])和MFP[6] (SYS_P2_MFP[6]) 决定P2.6功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择PWM0_CH4功能 (1, 1) = 选择ACMP1_O功能
[13]	**ALT[5]**	**P2.5功能选择位** 位ALT[5] (SYS_P2_MFP[13])和 MFP[5] (SYS_P2_MFP[5]) 决定 P2.5功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择UART1_TXD功能 (1, 0) = 选择PWM0_CH3功能 (1, 1) = 保留
[12]	**ALT[4]**	**P2.4功能选择位** 位ALT[4] (SYS_P2_MFP[12])和MFP[4] (SYS_P2_MFP[4]) 决定 P2.4功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择UART1_RXD功能 (1, 0) = 选择PWM0_CH2功能 (1, 1) = 保留

位	描述	
[11]	**ALT[3]**	**P2.3功能选择位** 位ALT[3] (SYS_P2_MFP[11])和MFP[3] (SYS_P2_MFP[3])决定P2.3功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择PWM0_CH1功能 (1, 1) = 选择 I2C1_SDA1功能
[10]	**ALT[2]**	**P2.2功能选择位** 位 ALT[2] (SYS_P2_MFP[10])和MFP[2] (SYS_P2_MFP[2]) 决定P2.2功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择PWM0_CH0功能 (1, 1) = 选择I2C1_SCL功能
[9:8]	**Reserved**	保留
[7:0]	**MFP[7:0]**	**P2复用功能选择位** P2 引脚功能取决于MFP和ALT 细节请参考ALT描述

Port3复用功能控制寄存器(SYS_P3_MFP)

寄存器	偏移地址	R/W	描述	复位值
SYS_P3_MFP	SYS_BA+0x3C	R/W	P3复用功能和输入类型控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					**P32EXT**	**Reserved**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TYPE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ALT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFP							

位	描述	
[31:27]	**Reserved**	保留
[26]	**P32EXT**	**P3.2功能选择扩展** 位P32EXT (SYS_P3_MFP[26])、 ALT[2] (SYS_P3_MFP[10])和MFP[2] (SYS_P3_MFP[2])决定 P3.2功能 (0, 0, 0) = 选择GPIO功能 (0, 0, 1) = 选择 INT0功能 (0, 1, 0) = 选择TM0_EXT功能 (0, 1, 1) = 选择STADC功能 (1, 0, 0) = 选择ACMP1_P1功能
[25:24]	**Reserved**	保留
[23:16]	**TYPE[n]**	**P3[7:0] 施密特触发输入功能使能位** 0 = P3[7:0] I/O施密特触发输入功能失能 1 = P3[7:0] I/O施密特触发输入功能使能
[15]	**Reserved**	保留
[14]	**ALT[6]**	**P3.6功能选择位** 位 ALT[6] (SYS_P3_MFP[14])和MFP[6] (SYS_P3_MFP[6])决定 P3.6 功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择TM®1_EXT功能 (1, 0) = 选择CLKO功能 (1, 1) = 选择ACMP0_O功能

位	描述	
[13]	**ALT[5]**	**P3.5功能选择位** 位 ALT[5] (SYS_P3_MFP[13])和MFP[5] (SYS_P3_MFP[5])决定 P3.5功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择TM®1_CNT_OUT功能 (1, 0) = 选择I2C0_SCL功能 (1, 1) = 选择ACMP1_P3功能
[12]	**ALT[4]**	**P3.4功能选择位** 位ALT[4] (SYS_P3_MFP[12])和MFP[4] (SYS_P3_MFP[4])决定P3.4功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择TM®0_CNT_OUT功能 (1, 0) = 选择I2C0_SDA功能 (1, 1) = 选择ACMP1_P2功能
[11]	**Reserved**	保留
[10]	**ALT[2]**	**P3.2功能选择位** 位P32EXT (SYS_P3_MFP[26])、ALT[2] (SYS_P3_MFP[10])和 MFP[2] (SYS_P3_MFP[2])决定P3.2功能 (0, 0, 0) = 选择GPIO功能 (0, 0, 1) = 选择 INT0 功能 (0, 1, 0) = 选择TM0_EXT功能 (0, 1, 1) = 选择STADC功能 (1, 0, 0) = 选择ACMP1_P1功能
[9]	**ALT[1]**	**P3.1功能选择位** 位 ALT[1] (SYS_P3_MFP[9])和MFP[1] (SYS_P3_MFP[1])决定P3.1功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择ACMP1_P0功能 (1, 1) = 选择ADC_CH7功能
[8]	**ALT[0]**	**P3.0功能选择位** 位ALT[0] (SYS_P3_MFP[8])和MFP[0] (SYS_P3_MFP[0])决定P3.0功能 (0, 0) = 选择GPIO功能 (0, 1) = 保留 (1, 0) = 选择ACMP1_N功能 (1, 1) = 选择ADC_CH6功能
[7:0]	**MFP[7:0]**	**P3复用功能选择位** P3引脚功能取决于MFP和ALT 细节请参考ALT描述

Port4复用功能控制寄存器 (SYS_P4_MFP)

寄存器	偏移地址	R/W	描述	复位值
SYS_P4_MFP	SYS_BA+0x40	R/W	P4复用功能和输入类型控制寄存器	0x0000_00C0

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

位	描述	
[31:24]	**Reserved**	保留
[23:16]	**TYPE[n]**	**P4[7:0] 施密特触发输入功能使能位** 0 = P4[7:0] I/O施密特触发输入功能失能 1 = P4[7:0] I/O施密特触发输入功能使能
[15]	**ALT[7]**	**P4.7功能选择位** 位ALT[7] (SYS_P4_MFP[15])和MFP[7] (SYS_P4_MFP[7])决定P4.7功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择ICE_DAT功能 (1, 0) = 选择 UART1_TXD功能 (1, 1) = 保留
[14]	**ALT[6]**	**P4.6功能选择位** 位 ALT[6] (SYS_P4_MFP[14)和MFP[6] (SYS_P4_MFP[6])决定P4.6功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择 ICE_CLK功能 (1, 0) = 选择UART1_RXD功能 (1, 1) = 保留
[13:8]	**Reserved**	保留
[7:0]	**MFP[7:0]**	**P4复用功能选择位** P4引脚功能取决于MFP和ALT 细节参考ALT描述

Port5复用功能控制寄存器(SYS_P5_MFP)

寄存器	偏移地址	R/W	描述	复位值
SYS_P5_MFP	SYS_BA+0x44	R/W	P5复用功能和输入类型控制寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留
[23:16]	**TYPE[n]**	**P5[7:0] 施密特触发输入功能使能位** 0 = P5[7:0] I/O施密特触发输入功能失能 1 = P5[7:0] I/O施密特触发输入功能使能
[15:14]	**Reserved**	保留
[13]	**ALT[5]**	**P5.5功能选择位** 位ALT[5] (SYS_P5_MFP[13])和MFP[5] (SYS_P5_MFP[5])决定P5.5功能 (0, 0) = 选择GPIO功能 其他 = 保留
[12]	**ALT[4]**	**P5.4功能选择位** 位 ALT[4] (SYS_P5_MFP[12])和 MFP[4] (SYS_P5_MFP[4])决定P5.4功能 (0, 0) = 选择GPIO功能 其他 = 保留
[11]	**ALT[3]**	**P5.3功能选择位** 位ALT[3] (SYS_P5_MFP[11])和MFP[3] (SYS_P5_MFP[3])决定P5.3功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择ADC_CH0功能 其他 = 保留
[10]	**ALT[2]**	**P5.2功能选择位** 位ALT[2] (SYS_P5_MFP[10])和MFP[2] (SYS_P5_MFP[2])决定P5.2功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择INT1功能 其他 = 保留

位	描述	
[9]	**ALT[1]**	**P5.1功能选择位** 位ALT[1] (SYS_P5_MFP[9])和MFP[1] (SYS_P5_MFP[1]) 决定 P5.1 功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择XT1_OUT功能 (1, 0) = 选择I2C1_SCL1功能 (1, 1) = 选择UART0_RXD功能 **注:** 为了使能外部XTAL功能，寄存器CLK_PWRCTL位[1:0] (XTLEN)及外部HXT或LXT晶振控制寄存器也必须被设置。
[8]	**ALT[0]**	**P5.0功能选择位** 位 ALT[0] (SYS_P5_MFP[8])和MFP[0] (SYS_P5_MFP[0])决定 P5.0功能 (0, 0) = 选择GPIO功能 (0, 1) = 选择XT1_IN功能 (1, 0) = 选择I2C1_SDA1功能 (1, 1) = 选择UART0_TXD功能 **注:** 为了使能外部XTAL功能,，寄存器CLK_PWRCTL位 [1:0] (XTLEN)及外部 HXT或 LXT晶振控制寄存器也必须被设置。
[7:0]	**MFP[7:0]**	**P5复用功能选择位** P5引脚功能取决于MFP和ALT 细节请参考ALT描述

HIRC校准控制寄存器 (SYS_IRCTCTL)

寄存器	偏移地址	R/W	描述	复位值
SYS_IRCTCTL	SYS_BA+0x80	R/W	HIRC校准控制寄存器	0x0000_0000

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

位	描述	
[31:6]	**Reserved**	保留
[5:4]	**LOOPSEL**	**校准计算循环** 该部分定义校准值是基于多少个LXT时钟计算出来的。 例如：假如LOOPSEL设置为“00”，自动校准电路将会基于4个LXT时钟偏差的平均值计算校准值。 该部分也定义了自动校准电路会在HIRC频率锁定前多少时间内试图更新HIRC校准值。 一旦HIRC被锁定，内部校准值更新计数器将会被复位。 假如校准值更新计数器到达限制值，但HIRC还没被锁定，那么自动校准操作将被失能，FREQSEL将被清零。 00 = 校准值基于4个LXT时钟偏差的平均值计算出来的且重试校准计数器上限值为64 01 = 校准值基于8个LXT时钟偏差的平均值计算出来的且重试校准计数器上限值为128 10 = 校准值基于16个LXT时钟偏差的平均值计算出来的且重试校准计数器上限值为256 11 = 校准值基于32个LXT时钟偏差的平均值计算出来的且重试校准计数器上限值为512
[3:1]	**Reserved**	保留
[0]	**FREQSEL**	**校准频率选择位** 该位用来使能HIRC自动校准 当该位设置为1时，HIRC自动校准功能会基于LXT参考时钟自动校准HIRC为22.1184 MHz 。 在自动校准过程中，如果检测到LXT时钟错误或者重试校准计数器到达限制值，该位会自动清零。 0 = HIRC自动校准功能失能 1 = HIRC自动校准功能使能且HIRC被校准到22.1184 MHz

HIRC校准中断使能寄存器 (SYS_IRCTIEN)

寄存器	偏移地址	R/W	描述	复位值
SYS_IRCTIEN	SYS_BA+0x84	R/W	HIRC校准中断使能寄存器	0x0000_0000

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

位	描述	
[31:3]	**Reserved**	保留
[2]	**CLKEIEN**	**LXT时钟错误中断使能位** 该位控制着在自动校准运算过程中，LXT时钟错误是否会使CPU产生中断。 如果该位为1，并且在自动校准运算期间，CLKERRIF (SYS_IRCTISTS[2])被置位，将会触发中断，提示LXT时钟不准确。 0 = 失能CLKERRIF (SYS_IRCTISTS[2]) 触发CPU中断 1 = 使能CLKERRIF (SYS_IRCTISTS[2]) 触发CPU中断
[1]	**TFAILIEN**	**校准失败中断使能位** 该位控制当HIRC校准次数达到更新极限值但HIRC频率仍然没有锁定在FREQSEL (SYS_IRCTCTL[1:0])设定的目标频率时，是否产生一个中断。 如果该位为1，并且在自动校准运算期间，TFAILIF (SYS_IRCTISTS[1])被置位，将会触发中断，提示HIRC校准次数已经达到了自动校准更新限制次数。 0 = 失能TFAILIF (SYS_IRCTISTS[1]) 触发CPU中断 1 = 使能TFAILIF (SYS_IRCTISTS[1]) 触发CPU中断
[0]	**Reserved**	保留

HIRC校准中断状态寄存器 (SYS_IRCTISTS)

寄存器	偏移地址	R/W	描述	复位值
SYS_IRCTISTS	SYS_BA+0x88	R/W	HIRC校准中断状态寄存器	0x0000_0000

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

位	描述	
[31:3]	**Reserved**	保留
[2]	**CLKERRIF**	**LXT时钟错误中断状态** 该位表明LXT时钟频率不精确。一旦该位置1，自动校准运算将会停止，且FREQSEL (SYS_IRCTCTL[0]) 会被硬件自动清零。 如果该位被置1且CLKEIEN (SYS_IRCTIEN[2]) 为高，将触发中断来提示LXT时钟频率不精确。软件可以通过向该位写1来清零。 0 = LXT时钟频率是精确的 1 = LXT时钟频率是不精确的
[1]	**TFAILIF**	**校准失败中断状态** 该位表明HIRC校准次数已经达到极限值时，HIRC频率仍然没有锁定。一旦该位置1，自动校准运算将会停止，且FREQSEL (SYS_IRCTCTL[1:0]) 会被硬件自动清零。 如果该位被置位且TFAILIEN (SYS_IRCTIEN[1])为高，将触发中断表明HIRC校准次数已经达到限制值。软件可以通过向该位写1来清零。 0 = 校准值更新限制次数没到达 1 = 校准值更新限制次数到达且HIRC频率仍没锁定
[0]	**FREQLOCK**	**HIRC频率锁定状态** 该位表明HIRC频率锁定在22.1184 MHz. 该位为只读状态位，不会触发任何中断

写保护寄存器控制寄存器(SYS_REGLCTL)

有些寄存器需要被保护起来，以防止误写而影响芯片运行。这些系统寄存器在上电复位后就受保护，直到用户关闭寄存器保护。用户需要编程这些受保护寄存器的话，需要一个解锁时序。解锁时序是依次连续的往地址为0x5000_0100的控制器写入0x59、 0x16、0x88。在写以上三个数据的过程中，写入其他任何数据、不同顺序写、或往别的地址写入数据都将终止解锁时序。

写保护功能失能后，用户可以查看保护失能位（地址0x5000_0100位0），1表示保护失能，0表示保护使能。写保护失能后，用户可以更新目标保护寄存器的值，往地址0x5000_0100写入任何值将使能寄存器保护。

写该寄存器来失能/使能寄存器保护，读该寄存器获取REGLCTL状态

寄存器	偏移地址	R/W	描述	复位值
SYS_REGLCTL	SYS_BA+0x100	R/W	寄存器写保护控制寄存器	0x0000_0000

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

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**REGLCTL**	**写保护代码寄存器(只写)** 一些寄存器具有写保护功能。写这些寄存器前必须按照顺序依次往寄存器写入0x59、0x16、0x88来失能写保护功能。在这个时序结束后，REGLCTL 位会被置1，写保护寄存器可以正常写入数据。

<table>
<tr><th>位</th><th colspan="2">描述</th></tr>
<tr><td>[0]</td><td>REGLCTL</td><td>
写保护寄存器失能标志 (只读)
0 = 写保护寄存器的写保护功能使能，任何写入写保护寄存器的数据将被忽略
1 = 写保护寄存器的写保护功能失能

保护寄存器如下表所示
<table>
<tr><th>寄存器</th><th>地址</th><th>注释</th></tr>
<tr><td>SYS_IPRST0</td><td>0x5000_0008</td><td>外设复位控制寄存器0</td></tr>
<tr><td>SYS_BODCTL</td><td>0x5000_0018</td><td>欠压检测器控制寄存器</td></tr>
<tr><td>CLK_PWRCTL</td><td>0x5000_0200</td><td>位[6]没有受保护，电源唤醒中断将清除该位</td></tr>
<tr><td>CLK_APBCLK bit[0]</td><td>0x5000_0208</td><td>位[0]是看门狗时钟使能位</td></tr>
<tr><td>CLK_CLKSEL0</td><td>0x5000_0210</td><td>HCLK和CPU STCLK 时钟源选择</td></tr>
<tr><td>CLK_CLKSEL1 bit[1:0]</td><td>0x5000_0214</td><td>看门狗时钟源选择</td></tr>
<tr><td>NMI_SEL bit[8]</td><td>0x5000_0380</td><td>NMI中断使能</td></tr>
<tr><td>ISPCON</td><td>0x5000_C000</td><td>Flash ISP控制</td></tr>
<tr><td>ISPTRG</td><td>0x5000_C010</td><td>ISP触发控制</td></tr>
<tr><td>WTCR</td><td>0x4000_4000</td><td>看门狗定时器控制</td></tr>
</table>
注: 写保护位在其描述旁边会标注“(写保护)”。
</td></tr>
</table>

6.2.9 系统定时器(SysTick)

Cortex®-M0内置一个系统定时器SysTick，SysTick提供一个简单的24位写清零、递减、自动装载，同时拥有灵活控制机制的寄存器。该计数器可用作实时操作系统(RTOS)的滴答定时器或一个简单的计数器。

当系统定时器使能后，将从SysTick当前值寄存器(SYST_CVR)的值向下计数到0，并在下一个时钟边缘，重新加载SysTick重新加载值寄存器(SYST_RVR)的值，然后再随时钟递减。当计数器递减到0，COUNTFLAG 状态位就会被设置，读COUNTFLAG 位使其清零。

SYST_CVR的值在复位后是未知的，使能前，软件应该向寄存器写入值来清零。这样确保定时器使能时以SYST_RVR的值开始计数，而非任意值。

如果SYST_RVR的值为0，定时器在重新加载后将保持值为0。这种机制可以在计数器使能后，独立地禁用计数功能。

详情请参考 “ARM® Cortex®-M0技术参考手册”和“ARM® v6-M架构参考手册”.

6.2.9.1 系统定时器控制寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
SCS基地址: **SCS_BA = 0xE000_E000**				
SYST_CSR	SCS_BA+0x10	R/W	SysTick控制和状态寄存器	0x0000_0000
SYST_RVR	SCS_BA+0x14	R/W	SysTick重新加载值寄存器	0x00XX_XXXX
SYST_CVR	SCS_BA+0x18	R/W	SysTick当前值寄存器	0x00XX_XXXX

6.2.9.2 系统定时器控制寄存器

SysTick控制和状态寄存器 (SYST_CSR)

寄存器	偏移地址	R/W	描述	复位值
SYST_CSR	SCS_BA+0x10	R/W	SysTick控制和状态寄存器	0x0000_0000

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

位	描述	
[31:17]	**Reserved**	保留
[16]	**COUNTFLAG**	**系统滴答计数器标志位** 从上次读取该寄存器后，如果定时器计数到0，则返回1 计数从1到0转变时，COUNTFLAG置位 读该位或写当前值寄存器将使COUNTFLAG清零
[15:3]	**Reserved**	保留
[2]	**CLKSRC**	**系统滴答时钟源选择位** 0 = 时钟源是可选的，参考STCLKSEL 1 = 内核时钟用于SysTick 定时器
[1]	**TICKINT**	**系统滴答中断使能位** 0 = 向下计数到0不会引起SysTick 异常而挂起。软件可以根据COUNTFLAG标志位来确定是否已经发生计数到0的情况。 1 = 向下计数到0将引起SysTick异常而挂起。软件清除SysTick当前值寄存器值将不会导致SysTick挂起。
[0]	**ENABLE**	**系统滴答计数器使能位** 0 = 计数器失能 1 = 计数器使能并运行于multi-shot方式

SysTick 重加载值寄存器 (SYST_RVR)

寄存器	偏移地址	R/W	描述	复位值
SYST_RVR	SCS_BA+0x14	R/W	SysTick重加载值寄存器	0x00XX_XXXX

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

位	描述	
[31:24]	**Reserved**	保留
[23:0]	**RELOAD**	**系统滴答重加载值** 当计数器计数到0时，该值将加载到当前值寄存器

SysTick当前值寄存器 (SYST_CVR)

寄存器	偏移地址	R/W	描述	复位值
SYST_CVR	SCS_BA+0x18	R/W	SysTick当前值寄存器	0x00XX_XXXX

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

位	描述	
[31:24]	**Reserved**	保留
[23:0]	**CURRENT**	**系统滴答当前值** 当前计数值。该值为采样时刻的计数器的值。计数器不提供读修改写保护功能。该寄存器有写清除功能，软件写入任何值到该寄存器将清除该寄存器值为0。不支持位RAZ (参看SysTick重新加载值寄存器)。

6.2.10 嵌套向量中断控制器(NVIC)

6.2.10.1 概述

Cortex®-M0 CPU提供一个中断控制器用于异常模式，称之为“嵌套向量中断控制器(NVIC)”。NVIC与处理器内核紧密相连，它提供以下特征：

6.2.10.2 特性

- 支持嵌套向量中断
- 自动保存和恢复处理器状态
- 动态改变优先级
- 简化和确定的中断时间

NVIC按照优先级处理所有支持的异常。所有的异常在“Handler模式”处理。NVIC架构支持32个(IRQ[31:0])离散中断，每个中断支持4级中断优先级。所有的中断和大部分系统异常可以配置成不同的优先级等级。当一个中断发生时，NVIC将比较新中断与当前中断的优先级，如果新中断优先级高，则立即处理新中断。

当一个中断接受后，中断服务程序(ISR)的开始地址可从内存中的向量表取得。软件不需要决定哪个中断被响应，也不用分配相关ISR的开始地址。当开始地址取得后，NVIC将自动保存包含寄存器“PC、PSR、LR、R0~R3、R12”值的处理器状态到栈中。在ISR结束后，NVIC将从栈中恢复相关寄存器的值，并运行在正常状态。因此花费少量且确定的时间处理中断请求。

NVIC支持“末尾连锁”，可以有效的处理背对背中断，即无需保存和恢复当前状态，从而减少结束当前ISR切换到挂起ISR的延迟时间。NVIC还支持“Late Arrival”，因此可以提高并发中断的效率。当较高优先级中断请求发生在当前ISR开始执行之前（保存处理器状态和获取起始地址阶段），NVIC将立即处理更高优先级的中断，从而提高实时性。

更多详细信息，请参考 “ARM® Cortex®-M0 技术参考手册”和 “ARM® v6-M 架构参考手册”。

6.2.10.3 异常模式和系统中断映射

下表列出了NuMicro® Mini58系列支持的异常模式。与所有中断一样，软件可以为其中一些异常设置4级优先级。最高用户可配置优先级为0，最低优先级为3。所有用户可配置中断的默认优先级为0。注：优先级0在系统为第4优先级，排在“Reset”、 “NMI”和“Hard Fault”三个系统异常之后。

异常名称	向量号	优先级
Reset	1	-3
NMI	2	-2
Hard Fault	3	-1
Reserved	4 ~ 10	保留
SVCall	11	可配置

Reserved	12 ~ 13	保留
PendSV	14	可配置
SysTick	15	可配置
Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	可配置

表 6.2-7 异常模式

异常号	中断号 (中断寄存器对应位)	中断名称	源模块	中断描述	掉电唤醒
1 ~ 15	-	**-**	-	系统异常	-
16	0	**BODOUT**	Brown-out	欠压检测中断	Yes
17	1	**WDT_INT**	WDT	看门狗定时器中断	Yes
18	2	**EINT0**	GPIO	P3.2引脚外部信号中断	Yes
19	3	**EINT1**	GPIO	P5.2引脚外部信号中断	Yes
20	4	**GP0/1_INT**	GPIO	P0~P1引脚外部信号中断	Yes
21	5	**GP2/3/4_INT**	GPIO	P2~P4(除P3.2)引脚外部信号中断	Yes
22	6	**PWM_INT**	PWM	PWM中断	No
23	7	**BRAKE_INT**	PWM	PWM刹车中断	No
24	8	**TMR0_INT**	TMR0	Timer 0 中断	Yes
25	9	**TMR1_INT**	TMR1	Timer 1 中断	Yes
26 ~ 27	10 ~ 11	**-**	-	-	
28	12	**UART0_INT**	UART0	UART0 中断	Yes
29	13	**UART1_INT**	UART1	UART1中断	Yes
30	14	**SPI_INT**	SPI	SPI中断	No
31	15	**-**	-	-	
32	16	**GP5_INT**	GPIO	P5(除P5.2)外部信号中断	Yes
33	17	**HIRC_TRIM_INT**	HIRC	HIRC校准中断	No
34	18	**I2C0_INT**	I^2C0	I^2C0中断	Yes
35	19	**I2C1_INT**	I^2C1	I^2C1中断	No
36 ~ 40	20 ~ 24	**-**	-	-	
41	25	**ACMP_INT**	ACMP	模拟比较器0或比较器1中断	Yes
42 ~ 43	26 ~ 27	**-**	-	-	
44	28	**PWRWU_INT**	CLKC	用于芯片掉电唤醒的时钟控制器中断	Yes
45	29	**ADC_INT**	ADC	ADC中断	No

异常号	中断号 (中断寄存器对应位)	中断名称	源模块	中断描述	掉电唤醒
46 ~ 47	30 ~ 31	-	-	-	

表 6.2-8 系统中断映射向量表

6.2.10.4 向量表

当一个中断被接受时，处理器会自动从内存中的向量表取出中断服务例程(ISR)的起始地址。对于ARMv6-M，向量表基地址固定为0x00000000。向量表包含复位后的栈指针初始值和所有异常处理器的入口地址。前一页的向量号表示前部分处理异常的先后顺序。

向量表字偏移地址 (字节)	描述
0x00	初始化栈指针值
异常号 * 0x04	异常入口指针使用异常号

表 6.2-8 向量表格式

6.2.10.5 操作描述

NVIC中断可以通过写使能中断或清使能中断寄存器相关位来使能或失能。这些寄存器使用写1使能和写1清零机制，通过读取这些寄存器可以读取当前相应中断的使能状态。当中断禁用后，中断声明将使中断挂起，因此中断不被激活。如果在禁用中断时中断被激活，它将保持激活状态直到被复位或异常返回清除。清使能位可以阻止相应中断的新中断被激活。

NVIC中断可以使用互补寄存器对来挂起/取消挂起来使能/禁用中断，这些寄存器分别为Set-Pending寄存器和Clear-Pending寄存器。这些寄存器使用写1使能和写1清零机制，通过读取这些寄存器可以读取当前相应中断的挂起状态。 Clear-Pending寄存器不影响激活中断的执行状态。

NVIC中断的优先级可以通过更新32位寄存器的8个位段来设置（每个寄存器支持4个中断）。

与NVIC相关的通用寄存器在系统控制空间的一块内存中设置，下一节将做出描述。

6.2.10.6 NVIC 控制寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
SCS基地址: **SCS_BA = 0xE000_E000**				
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31 设置使能控制寄存器	0x0000_0000
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 清使能控制寄存器	0x0000_0000
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ31 设置挂起控制寄存器	0x0000_0000
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31 清挂起控制寄存器	0x0000_0000
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 中断优先级控制寄存器	0x0000_0000
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 中断优先级控制寄存器	0x0000_0000
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11 中断优先级控制寄存器	0x0000_0000
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15 中断优先级控制寄存器	0x0000_0000
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19 中断优先级控制寄存器	0x0000_0000
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 中断优先级控制寄存器	0x0000_0000
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ27 中断优先级控制寄存器	0x0000_0000
NVIC_IPR7	SCS_BA+0x41C	R/W	IRQ28 ~ IRQ31 中断优先级控制寄存器	0x0000_0000

IRQ0 ~ IRQ31 设置使能控制寄存器 (NVIC_ISER)

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31设置使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	**SETENA**	**中断使能位** 使能一个或多个中断。每一位代表一个从IRQ0~IRQ31的中断（向量号从16~47）。 写操作: 0 = 无效 1 = 写1使能相关中断 读操作: 0 = 相关中断状态禁止 1 = 相关中断状态使能 **注:** 读该寄存器值表明当前使能状态

IRQ0 ~ IRQ31 清使能控制寄存器(NVIC_ICER)

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 清使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	**CLRENA**	**中断禁用位** 禁用一个或多个中断。每一位表示一个从IRQ0~IRQ31的中断（向量号从16~47）。 写操作: 0 = 无效 1 = 写1禁止相关中断 读操作: 0 = 相关中断状态禁止 1 = 相关中断状态使能 **注:** 读该寄存器值表明当前使能状态

IRQ0 ~ IRQ31 设置挂起控制寄存器 (NVIC_ISPR)

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ31设置挂起控制寄存器	0x0000_0000

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

位	描述	
[31:0]	**SETPEND**	**设置中断挂起位** 写操作: 0 = 无效 1 = 写1置挂起状态。每一位表示一个从IRQ0 ~ IRQ31的中断（向量号从16~47）。 读操作: 0 = 相关中断不在挂起状态 1 = 相关中断在挂起状态 **注:** 读该寄存器值表明当前挂起状态

IRQ0 ~ IRQ31 清挂起控制寄存器 (NVIC_ICPR)

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31清挂起控制寄存器	0x0000_0000

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

位	描述	
[31:0]	**CLRPEND**	**清中断挂起位** 写操作: 0 = 无效 1 = 写1清挂起状态。每一位表示一个从IRQ0 ~ IRQ31的中断（向量号从16~47）。 读操作: 0 = 相关中断不在挂起状态 1 = 相关中断在挂起状态 **注:** 读该寄存器值表明当前挂起状态

IRQ0 ~ IRQ3 中断优先级寄存器 (NVIC_IPR0)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 中断优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_3		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_2		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_1		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_0		**Reserved**					

位	描述	
[31:30]	**PRI_3**	**IRQ3优先级** 0表示最高优先级，3表示最低优先级
[29:24]	**Reserved**	保留
[23:22]	**PRI_2**	**IRQ2优先级** 0表示最高优先级，3表示最低优先级
[21:16]	**Reserved**	保留
[15:14]	**PRI_1**	**IRQ1优先级** 0表示最高优先级，3表示最低优先级
[13:8]	**Reserved**	保留
[7:6]	**PRI_0**	**IRQ0优先级** 0表示最高优先级，3表示最低优先级
[5:0]	**Reserved**	保留

IRQ4 ~ IRQ7 中断优先级寄存器 (NVIC_IPR1)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 中断优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_7		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_6		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_5		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_4		**Reserved**					

位	描述	
[31:30]	**PRI_7**	**IRQ7优先级** 0表示最高优先级，3表示最低优先级
[29:24]	**Reserved**	保留
[23:22]	**PRI_6**	**IRQ6优先级** 0表示最高优先级，3表示最低优先级
[21:16]	**Reserved**	保留
[15:14]	**PRI_5**	**IRQ5优先级** 0表示最高优先级，3表示最低优先级
[13:8]	**Reserved**	保留
[7:6]	**PRI_4**	**IRQ4优先级** 0表示最高优先级，3表示最低优先级
[5:0]	**Reserved**	保留

IRQ8 ~ IRQ11中断优先级寄存器(NVIC_IPR2)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11中断优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_10		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_9		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_8		**Reserved**					

位	描述	
[31:30]	**PRI_11**	**IRQ11优先级** 0表示最高优先级，3表示最低优先级
[29:24]	**Reserved**	保留
[23:22]	**PRI_10**	**IRQ10优先级** 0表示最高优先级，3表示最低优先级
[21:16]	**Reserved**	保留
[15:14]	**PRI_9**	**IRQ9优先级** 0表示最高优先级，3表示最低优先级
[13:8]	**Reserved**	保留
[7:6]	**PRI_8**	**IRQ8优先级** 0表示最高优先级，3表示最低优先级
[5:0]	**Reserved**	保留

IRQ12 ~ IRQ15 中断优先级寄存器 (NVIC_IPR3)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15中断优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_15		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_14		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_13		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_12		**Reserved**					

位	描述	
[31:30]	**PRI_15**	**IRQ15优先级** 0表示最高优先级，3表示最低优先级
[29:24]	**Reserved**	保留
[23:22]	**PRI_14**	**IRQ14优先级** 0表示最高优先级，3表示最低优先级
[21:16]	**Reserved**	保留
[15:14]	**PRI_13**	**IRQ13优先级** 0表示最高优先级，3表示最低优先级
[13:8]	**Reserved**	保留
[7:6]	**PRI_12**	**IRQ12优先级** 0表示最高优先级，3表示最低优先级
[5:0]	**Reserved**	保留

IRQ16 ~ IRQ19中断优先级寄存器(NVIC_IPR4)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19中断优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_19		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_18		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_17		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_16		**Reserved**					

位	描述	
[31:30]	**PRI_19**	**IRQ19优先级** 0表示最高优先级，3表示最低优先级
[29:24]	**Reserved**	保留
[23:22]	**PRI_18**	**IRQ18优先级** 0表示最高优先级，3表示最低优先级
[21:16]	**Reserved**	保留
[15:14]	**PRI_17**	**IRQ17优先级** 0表示最高优先级，3表示最低优先级
[13:8]	**Reserved**	保留
[7:6]	**PRI_16**	**IRQ16优先级** 0表示最高优先级，3表示最低优先级
[5:0]	**Reserved**	保留

IRQ20 ~ IRQ23 中断优先级寄存器 (NVIC_IPR5)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 中断优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_23		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_22		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_21		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_20		**Reserved**					

位	描述	
[31:30]	**PRI_23**	**IRQ23优先级** 0表示最高优先级，3表示最低优先级
[29:24]	**Reserved**	保留
[23:22]	**PRI_22**	**IRQ22优先级** 0表示最高优先级，3表示最低优先级
[21:16]	**Reserved**	保留
[15:14]	**PRI_21**	**IRQ21优先级** 0表示最高优先级，3表示最低优先级
[13:8]	**Reserved**	保留
[7:6]	**PRI_20**	**IRQ20优先级** 0表示最高优先级，3表示最低优先级
[5:0]	**Reserved**	保留

IRQ24 ~ IRQ27 中断优先级寄存器 (NVIC_IPR6)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ27 中断优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_27		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_26		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_25		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_24		**Reserved**					

位	描述	
[31:30]	**PRI_27**	**IRQ27优先级** 0表示最高优先级，3表示最低优先级
[29:24]	**Reserved**	保留
[23:22]	**PRI_26**	**IRQ26优先级** 0表示最高优先级，3表示最低优先级
[21:16]	**Reserved**	保留
[15:14]	**PRI_25**	**IRQ25优先级** 0表示最高优先级，3表示最低优先级
[13:8]	**Reserved**	保留
[7:6]	**PRI_24**	**IRQ24优先级** 0表示最高优先级，3表示最低优先级
[5:0]	**Reserved**	保留

IRQ28 ~ IRQ31 中断优先级寄存器 (NVIC_IPR7)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPR7	SCS_BA+0x41C	R/W	IRQ28 ~ IRQ31中断优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_31		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_30		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_29		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_28		**Reserved**					

位	描述	
[31:30]	**PRI_31**	**IRQ31优先级** 0表示最高优先级，3表示最低优先级
[29:24]	**Reserved**	保留
[23:22]	**PRI_30**	**IRQ30优先级** 0表示最高优先级，3表示最低优先级
[21:16]	**Reserved**	保留
[15:14]	**PRI_29**	**IRQ29优先级** 0表示最高优先级，3表示最低优先级
[13:8]	**Reserved**	保留
[7:6]	**PRI_28**	**IRQ28优先级** 0表示最高优先级，3表示最低优先级
[5:0]	**Reserved**	保留

6.2.10.7 中断源控制寄存器

除了与NVIC相关的中断控制寄存器外，NuMicro® Mini58 系列也提供了一些特殊的控制寄存器来增强中断功能，包含“中断源识别”、“NMI源选择”和“中断测试模式”，如下描述。

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
INT 基地址: **INT_BA = 0x5000_0300**				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) 中断源识别	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) 中断源识别	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) 中断源识别	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) 中断源识别	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GP0/1) 中断源识别	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (GP2/3/4) 中断源识别	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWM) 中断源识别	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (BRAKE) 中断源识别	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) 中断源识别	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) 中断源识别	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	-	保留	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	-	保留	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0) 中断源识别	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (UART1) 中断源识别	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI) 中断源识别	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	-	保留	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (GP5) 中断源识别	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (HIRC trim) 中断源识别	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I^2C0) 中断源识别	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I^2C1) 中断源识别	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	-	保留	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	-	保留	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	-	保留	0xXXXX_XXXX
IRQ23_SRC	INT_BA+0x5C	-	保留	0xXXXX_XXXX

寄存器	偏移地址	R/W	描述	复位值
INT 基地址: **INT_BA = 0x5000_0300**				
IRQ24_SRC	INT_BA+0x60	-	保留	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (ACMP) 中断源识别	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	-	保留	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	-	保留	0xXXXX_XXXX
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) 中断源识别	0xXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) 中断源识别	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	-	保留	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	-	保留	0xXXXX_XXXX
NMI_CON	INT_BA+0x80	R/W	NMI中断源选择控制寄存器	0x0000_0000
MCU_IRQ	INT_BA+0x84	R/W	MCU IRQ号识别寄存器	0x0000_0000

IRQ0 (BOD) 中断源识别 (IRQ0_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) 中断源识别	0xXXXX_XXXX

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

位	描述	
[31:1]	**Reserved**	保留
[0]	**BOD_INT**	**IRQ0 中断源识别** 0 = IRQ0 中断源不是来自BOD中断 (BOD_INT). 1 = IRQ0 中断源来自BOD中断 (BOD_INT).

IRQ1 (WDT) 中断源识别 (IRQ1_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) 中断源识别	0xXXXX_XXXX

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

位	描述	
[31:2]	**Reserved**	保留
[1]	**WWDT_INT**	**IRQ1中断源识别** 0 = IRQ1中断源不是来自窗口看门狗中断 (WWDT _INT). 1 = IRQ1 中断源来自窗口看门狗中断 (WWDT_INT).
[0]	**WDT_INT**	**IRQ1中断源识别** 0 = IRQ1中断源不是来自看门狗中断(WDT _INT). 1 = IRQ1中断源来自看门狗中断 (WDT_INT).

IRQ2 (EINT0) 中断源识别 (IRQ2_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0)中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**EINT0**

位	描述	
[31:1]	**Reserved**	保留
[0]	**EINT0**	**IRQ2 中断源识别** 0 = IRQ2中断源不是来自P3.2引脚的外部中断信号0(EINT0) 1 = IRQ2中断源来自P3.2引脚的外部中断信号0(EINT0)

IRQ3 (EINT1) 中断源识别 (IRQ3_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) 中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**EINT1**

位	描述	
[31:1]	**Reserved**	保留
[0]	**EINT1**	**IRQ3中断源识别** 0 = IRQ3中断源不是来自P5.2引脚的外部中断信号1 (EINT1) 1 = IRQ3中断源是来自P5.2引脚的外部中断信号1 (EINT1)

IRQ4 (GPA/B)中断源识别 (IRQ4_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GP0/1) 中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**GP1_INT**	**GP0_INT**

位	描述	
[31:2]	**Reserved**	保留
[1]	**GP1_INT**	**IRQ4中断源识别** 0 = IRQ4中断源不是来自GP1中断(GP1_INT). 1 = IRQ4中断源来自GP1中断(GP1_INT).
[0]	**GP0_INT**	**IRQ4中断源识别** 0 = IRQ4中断源不是来自GP0中断(GP0_INT). 1 = IRQ4中断源来自GP0中断(GP0_INT).

IRQ5 (GPC/D/F)中断源识别(IRQ5_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (GP2/3/4)中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**GP4_INT**	**GP3_INT**	**GP2_INT**

位	描述	
[31:3]	**Reserved**	保留
[2]	**GP4_INT**	**IRQ5中断源识别** 0 = IRQ5中断源不是来自GP4中断 (GP4_INT). 1 = IRQ5中断源来自GP4中断(GP4_INT).
[1]	**GP3_INT**	**IRQ5中断源识别** 0 = IRQ5中断源不是来自GP3中断(GP3_INT). 1 = IRQ5中断源来自GP3中断(GP3_INT).
[0]	**GP2_INT**	**IRQ5中断源识别** 0 = IRQ5中断源不是来自GP2中断(GP2_INT). 1 = IRQ5来自GP2中断(GP2_INT).

IRQ6 (PWM)中断源识别(IRQ6_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWM) 中断源识别	0xXXXX_XXXX

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

位	描述	
[31:1]	**Reserved**	保留
[0]	**PWM_INT**	**IRQ6中断源识别** 0 = IRQ6中断源不是来自PWM中断(PWM_INT). 1 = IRQ6中断源来自PWM中断(PWM_INT).

IRQ7 (BRAKE)中断源识别 (IRQ7_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (BRAKE) 中断源识别	0xXXXX_XXXX

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

位	描述	
[31:1]	**Reserved**	保留
[0]	**BRAKE_INT**	**IRQ7中断源识别** 0 = IRQ7中断源不是来自Brake中断(BRAKE_INT). 1 = IRQ7中断源来自Brake中断(BRAKE_INT).

IRQ8 (TMR0)中断源识别 (IRQ8_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0)中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**TMR0_INT**

位	描述	
[31:1]	**Reserved**	保留
[0]	**TMR0_INT**	**IRQ8中断源识别** 0 = IRQ8中断源不是来自Timer0中断(TMR0_INT). 1 = IRQ8中断源来自Timer0中断(TMR0_INT).

IRQ9 (TMR1) 中断源识别(IRQ9_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) 中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**TMR1_INT**

位	描述	
[31:1]	**Reserved**	保留
[0]	**TMR1_INT**	**IRQ9中断源识别** 0 = IRQ9中断源不是来自Timer1中断(TMR1_INT). 1 = IRQ9中断源来自Timer1中断(TMR1_INT).

IRQ12 (UART0)中断源识别(IRQ12_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0) 中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**UART0_INT**

位	描述	
[31:1]	**Reserved**	保留
[0]	**UART0_INT**	**IRQ12中断源识别** 0 = IRQ12中断源不是来自UART0中断(UART0_INT). 1 = IRQ12中断源来自UART0中断(UART0_INT).

IRQ13 (UART1)中断源识别(IRQ13_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (UART1) 中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**UART1_INT**

位	描述	
[31:1]	**Reserved**	保留
[0]	**UART1_INT**	**IRQ13中断源识别** 0 = IRQ13中断源不是来自UART1中断(UART1_INT). 1 = IRQ13中断源来自UART1中断(UART1_INT).

IRQ14 (SPI)中断源识别 (IRQ14_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI) 中断源识别	0xXXXX_XXXX

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

位	描述	
[31:1]	**Reserved**	保留
[0]	**SPI_INT**	**IRQ14中断源识别** 0 = IRQ14中断源不是来自SPI中断(SPI_INT). 1 = IRQ14中断源来自SPI中断(SPI_INT).

IRQ16 (GP5)中断源识别(IRQ16_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (GP5) 中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**GP5_INT**

位	描述	
[31:1]	**Reserved**	保留
[0]	**GP5_INT**	**IRQ16中断源识别** 0 = IRQ16中断源不是来自GP5中断(GP5_INT). 1 = IRQ16中断源来自GP5中断(GP5_INT).

IRQ17 (HIRC校准) 中断源识别(IRQ17_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (HIRC校准)中断源识别	0xXXXX_XXXX

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

位	描述	
[31:1]	**Reserved**	保留
[0]	**HIRC_TRIM_INT**	**IRQ17中断源识别** 0 = IRQ17中断源不是来自HIRC校准中断(HIRC_TRIM_INT). 1 = IRQ17来自HIRC校准中断(HIRC_TRIM_INT).

IRQ18 (I^2C0)中断源识别(IRQ18_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I^2C0)中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**I2C0_INT**

位	描述	
[31:1]	**Reserved**	保留
[0]	**I2C0_INT**	**IRQ18 中断源识别** 0 = IRQ18中断源不是来自I^2C0中断(I2C0_INT). 1 = IRQ18中断源来自I^2C0中断(I2C0_INT).

IRQ19 (I^2C1)中断源识别(IRQ19_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I^2C1)中断源识别	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**I2C1_INT**

位	描述	
[31:1]	**Reserved**	保留
[0]	**I2C1_INT**	**IRQ19中断源识别** 0 = IRQ19中断源不是来自I^2C1中断(I2C1_INT). 1 = IRQ19中断源来自I^2C1中断(I2C1_INT).

IRQ25 (ACMP)中断源识别(IRQ25_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (ACMP)中断源识别	0xXXXX_XXXX

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

位	描述	
[31:1]	**Reserved**	保留
[0]	**ACMP_INT**	**IRQ25中断源识别** 0 = IRQ25中断源不是来自ACMP中断(ACMP_INT). 1 = IRQ25中断源来自ACMP中断(ACMP_INT).

IRQ28 (PWRWU)中断源识别(IRQ28_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU)中断源识别	0xXXXX_XXXX

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

位	描述	
[31:1]	**Reserved**	保留
[0]	**PWRWU_INT**	**IRQ28中断源识别** 0 = IRQ28中断源不是来自PWRWU中断(PWRWU_INT). 1 = IRQ28中断源来自PWREU中断(PWRWU_INT).

IRQ29 (ADC)中断源识别(IRQ29_SRC)

寄存器	偏移地址	R/W	描述	复位值
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC)中断源识别	0xXXXX_XXXX

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

位	描述	
[31:1]	**Reserved**	保留
[0]	**ADC_INT**	**IRQ29中断源识别** 0 = IRQ29中断源不是来自ADC中断(ADC_INT). 1 = IRQ29中断源来自ADC中断(ADC_INT).

NMI中断源选择控制寄存器(NMI_SEL)

寄存器	偏移地址	R/W	描述	复位值
NMI_CON	INT_BA+0x80	R/W	NMI中断源选择控制寄存器	0x0000_0000

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

位	描述	
[31:9]	**Reserved**	保留
[8]	**NMI_SEL_EN**	**NMI中断使能位(写保护)** 0 = NMI中断禁止 1 = NMI中断使能 **注:** 该位为写保护位，要编程该位需要依次往地址0x5000_0100写入0x59、0x16和0x88来关闭寄存器保护。详情请参考地址为SYS_BA+0x100的寄存器SYS_REGLCTL 。
[7:5]	**Reserved**	保留
[4:0]	**NMI_SEL**	**NMI中断源选择位** Cortex®-M0的NMI中断可以选择NMI_SEL设置的任何一个外设中断。

MCU 中断请求源寄存器(MCU_IRQ)

寄存器	偏移地址	R/W	描述	复位值
MCU_IRQ	INT_BA+0x84	R/W	MCU IRQ号识别寄存器	0x0000_0000

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

位	描述	
[31:0]	**MCU_IRQ**	**MCU IRQ中断源位** MCU_IRQ收集来自外设的所有中断，然后形成到Cortex®-M0内核的同步中断。有正常和测试两种模式形成到Cortex®-M0的中断。 MCU_IRQ收集来自每个外设的所有中断、并使他们同步，然后中断Cortex®-M0。 当MCU_IRQ[n]为0时，设置MCU_IRQ[n]为1将生成一个到Cortex®-M0 NVIC[n]的中断。 当MCU_IRQ[n]为1（意味着中断被声明）时，设置MCU_bit[n]为1将清除中断，设置MCU_IRQ[n] 为0无影响。

6.2.11 系统控制寄存器(SCB)

Cortex®-M0状态和运行模式控制由系统控制寄存器管理。通过这些系统控制寄存器可以控制CPUID、Cortex®-M0中断优先级和Cortex®-M0电源管理。

更多详细信息请参考“ARM® Cortex®-M0技术参考手册”和“ARM® v6-M架构参考手册”。

6.2.11.1 系统控制块寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
SCS基地址: **SCS_BA = 0xE000_E000**				
SCS_CPUID	SCS_BA+0xD00	R	CPUID寄存器	0x410C_C200
SCS_ICSR	SCS_BA+0xD04	R/W	中断控制状态寄存器	0x0000_0000
SCS_AIRCR	SCS_BA+0xD0C	R/W	应用中断和复位控制寄存器	0xFA05_0000
SCS_SCR	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000
SCS_SHPR2	SCS_BA+0xD1C	R/W	系统处理器优先级寄存器2	0x0000_0000
SCS_SHPR3	SCS_BA+0xD20	R/W	系统处理器优先级寄存器3	0x0000_0000

6.2.11.2 系统控制寄存器

CPUID寄存器(SCS_CPUID)

寄存器	偏移地址	R/W	描述	复位值
SCS_CPUID	SCS_BA+0xD00	R	CPUID寄存器	0x410C_C200

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

位	描述	
[31:24]	**IMPLEMENTER**	**实施代码** 实施代码由 ARM® 分配(ARM® = 0x41)
[23:20]	**Reserved**	保留
[19:16]	**PART**	**处理器的架构** ARMv6-M读出来的值为0xC
[15:4]	**PARTNO**	**处理器产品编号** 读出来的值为0xC20
[3:0]	**REVISION**	**修订号** 读出来的值为0x0

中断控制状态寄存器(SCS_ICSR)

寄存器	偏移地址	R/W	描述	复位值
SCS_ICSR	SCS_BA+0xD04	R/W	中断控制状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
NMIPENDSET	**Reserved**		**PENDSVSET**	**PENDSVCLR**	**PENDSTSET**	**PENDSTCLR**	**Reserved**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISRPREEMPT	**ISRPENDING**	**Reserved**	**VECTPENDING**				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VECTPENDING				**Reserved**			**VECTACTIVE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VECTACTIVE							

位	描述	
[31]	**NMIPENDSET**	**NMI设置挂起位** 写操作: 0 = 无效 1 = 将NMI异常挂起 读操作: 0 = NMI异常没有挂起 1 = NMI异常挂起 **注:** 由于NMI是最高优先级异常，所以通常处理器一检测到该位写1就会进入NMI异常处理。进入异常处理后，处理器会将该位清零。这意味着只有当处理器正在执行NMI异常处理程序时再次产生NMI信号，NMI异常处理程序读取这一位才返回1。
[30:29]	**Reserved**	保留
[28]	**PENDSVSET**	**PendSV设置挂起位** 写操作: 0 = 无效 1 = 将PendSV异常挂起 读操作: 0 = PendSV异常没有挂起 1 = PendSV异常挂起 **注:** 设置该位为1是设置PendSV 异常挂起的唯一方法。
[27]	**PENDSVCLR**	**PendSV 清挂起位** 写操作: 0 = 无效 1 = 清除PendSV异常挂起状态 **注:** 该位为只写位。为了清除PENDSV位，你必须同时往PENDSVSET写0，往PENDSVCLR写1 。

位	描述	
[26]	**PENDSTSET**	**SysTick异常设置挂起位** 写操作: 0 = 无效 1 = 将SysTick异常挂起 读操作: 0 = SysTick异常没有挂起 1 = SysTick异常挂起
[25]	**PENDSTCLR**	**SysTick异常清挂起位** 写操作: 0 = 无效 1 = 清除SysTick异常挂起状态 **注:** 该位为只读位。当你要清除PENDST位时，必须同时往PENDSTSET写0，往PENDSTCLR写1 。
[24]	**Reserved**	保留
[23]	**ISRPREEMPT**	**中断抢先占有位** 如果该位设置，一个挂起的异常将会从调试停止状态退出并进入服务。 **注:** 该位只读
[22]	**ISRPENDING**	**中断挂起标志(不包括 NMI和Faults)** 0 = 中断没有挂起 1 = 中断挂起 **注:** 该位只读
[21]	**Reserved**	保留
[20:12]	**VECTPENDING**	**挂起异常中优先级最高的异常号** 0 = 没有异常挂起 非0 = 挂起异常中优先级最高的异常号 **注:** 这些位只读
[11:9]	**Reserved**	保留
[8:0]	**VECTACTIVE**	**包含当前执行异常号** 0 = Thread模式 非0 = 当前执行异常的异常号 **注:** 这些位为只读位

应用中断和复位控制寄存器(SCS_AIRCR)

寄存器	偏移地址	R/W	描述	复位值
SCS_AIRCR	SCS_BA+0xD0C	R/W	应用中断和复位控制寄存器	0xFA05_0000

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

位	描述	
[31:16]	**VECTORKEY**	**寄存器访问密钥** 写操作: 当要写这个寄存器时，VECTORKEY 域必须设置为0x05FA ，否则写操作将被忽略。VECTORKEY 域用于防止系统复位或者清除异常状态时误写该寄存器。 读操作: 读出来的值为0xFA05.
[15:3]	**Reserved**	保留
[2]	**SYSRESETREQ**	**系统复位请求** 往该位写1将引起一个复位信号给芯片，表明有复位请求。 该位为只写位，复位后自动清零。
[1]	**VECTCLRACTIVE**	**异常有效状态清除位** 保留给调试使用。当写这个寄存器时，用户必须往该位写0，否则将出现不可预知的情况。
[0]	**Reserved**	保留

系统控制寄存器 (SCS_SCR)

寄存器	偏移地址	R/W	描述	复位值
SCS_SCR	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000

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

位	描述	
[31:5]	**Reserved**	保留
[4]	**SEVONPEND**	**挂起时发送事件** 0 = 只有使能中断或事件可以唤醒处理器，不包括禁用中断在内。 1 = 使能事件和所有中断（包括禁用的中断），可以唤醒处理器。 当一个事件或中断进入挂起状态时，事件信号从WFE唤醒处理器。如果处理器没在等待事件，事件将会被注册并影响下一个WFE。 执行SEV指令或外部事件同样会唤醒处理器。
[3]	**Reserved**	保留
[2]	**SLEEPDEEP**	**处理器深度睡眠和睡眠模式选择** 控制处理器在低电模式时使用休眠还是深度休眠模式。 0 = 休眠模式 1 = 深度休眠模式
[1]	**SLEEPONEXIT**	**Sleep-On-Exit使能** 该位表明当从Handler 模式返回到Thread模式时，是否退出休眠 0 = 当从Thread 模式返回时，不休眠 1 = 从ISR返回到Thread模式时，进入休眠或深度休眠 置该位为1使能一个中断驱动应用，从而避免返回到一个空的主函数应用。
[0]	**Reserved**	保留

系统处理器优先级寄存器 2 (SCS_SHPR2)

寄存器	偏移地址	R/W	描述	复位值
SCS_SHPR2	SCS_BA+0xD1C	R/W	系统处理器优先级寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:30]	**PRI_11**	**系统处理器11优先级 – SVCall** 0表示最高优先级，3表示最低优先级
[29:0]	**Reserved**	保留

系统处理器优先级寄存器3 (SCS_SHPR3)

寄存器	偏移地址	R/W	描述	复位值
SCS_SHPR3	SCS_BA+0xD20	R/W	系统处理器优先级寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
PRI_15		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_14		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:30]	**PRI_15**	**系统处理器15优先级 – SysTick** 0表示最高优先级，3表示最低优先级
[29:24]	**Reserved**	保留
[23:22]	**PRI_14**	**系统处理器14优先级 – PendSV** 0表示最高优先级，3表示最低优先级
[21:0]	**Reserved**	保留

6.3 时钟控制器

6.3.1 概述

时钟控制器为整个芯片提供时钟源，包括系统时钟和所有外围设备时钟。该控制器还通过单独时钟的开或关，时钟源选择和分频器来进行功耗控制。在CPU使能低功耗PDEN(CLK_PWRCTL[7])位和Cortex®-M0内核执行WFI指令后，芯片才能进入低功耗模式。直到唤醒中断发生，芯片才会退出低功耗模式。在低功耗模式下，时钟控制器会关闭外部4~24MHz高速晶振和内部22.1184MHz高速RC振荡器，以降低整个系统功耗。下图所示各模块时钟发生器和时钟源的简图。

时钟发生器由如下3个时钟源组成：

- 外部4~24 MHz高速晶振(HXT)或外部32.768 kHz低速晶振（LXT）
- 可编程的PLL输出时钟频率(PLLFOUT)，PLL 由外部 4~24 MHz 的高速晶振或内部22.1184 MHz 振荡器提供时钟源)
- 内部22.1184 MHz高速RC振荡器(HIRC)
- 内部10 kHz低速RC振荡器(LIRC)

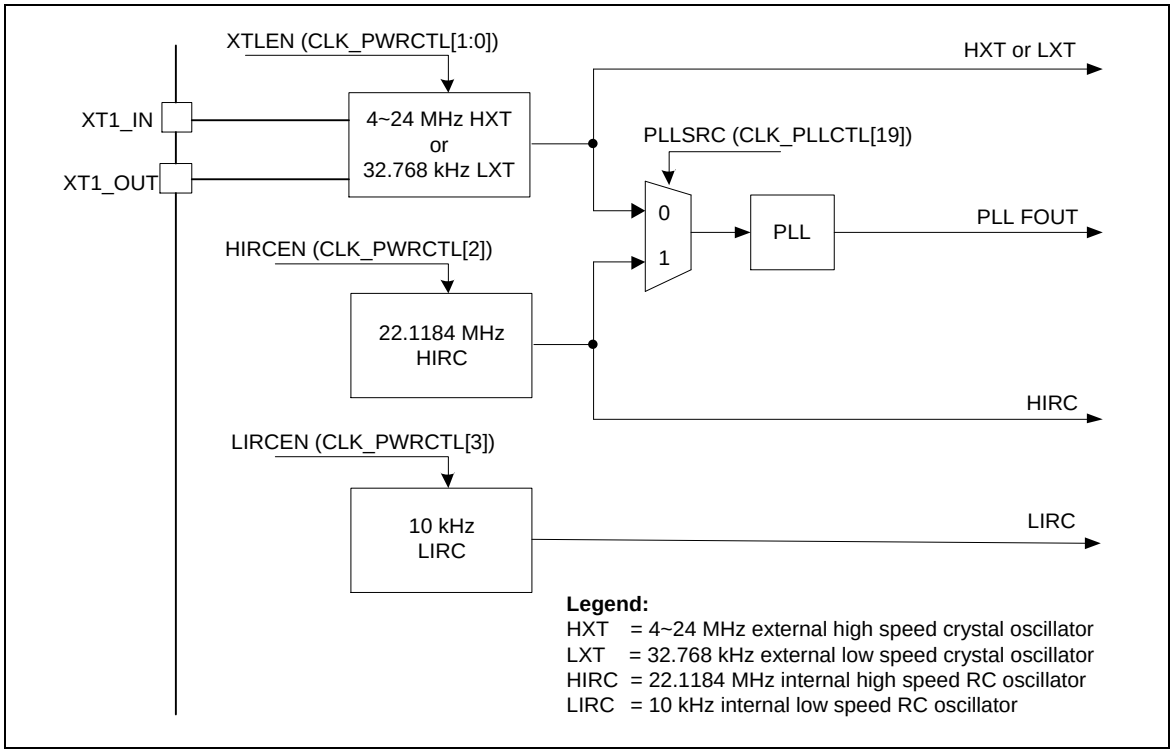


图 6.3-1 时钟发生器框图

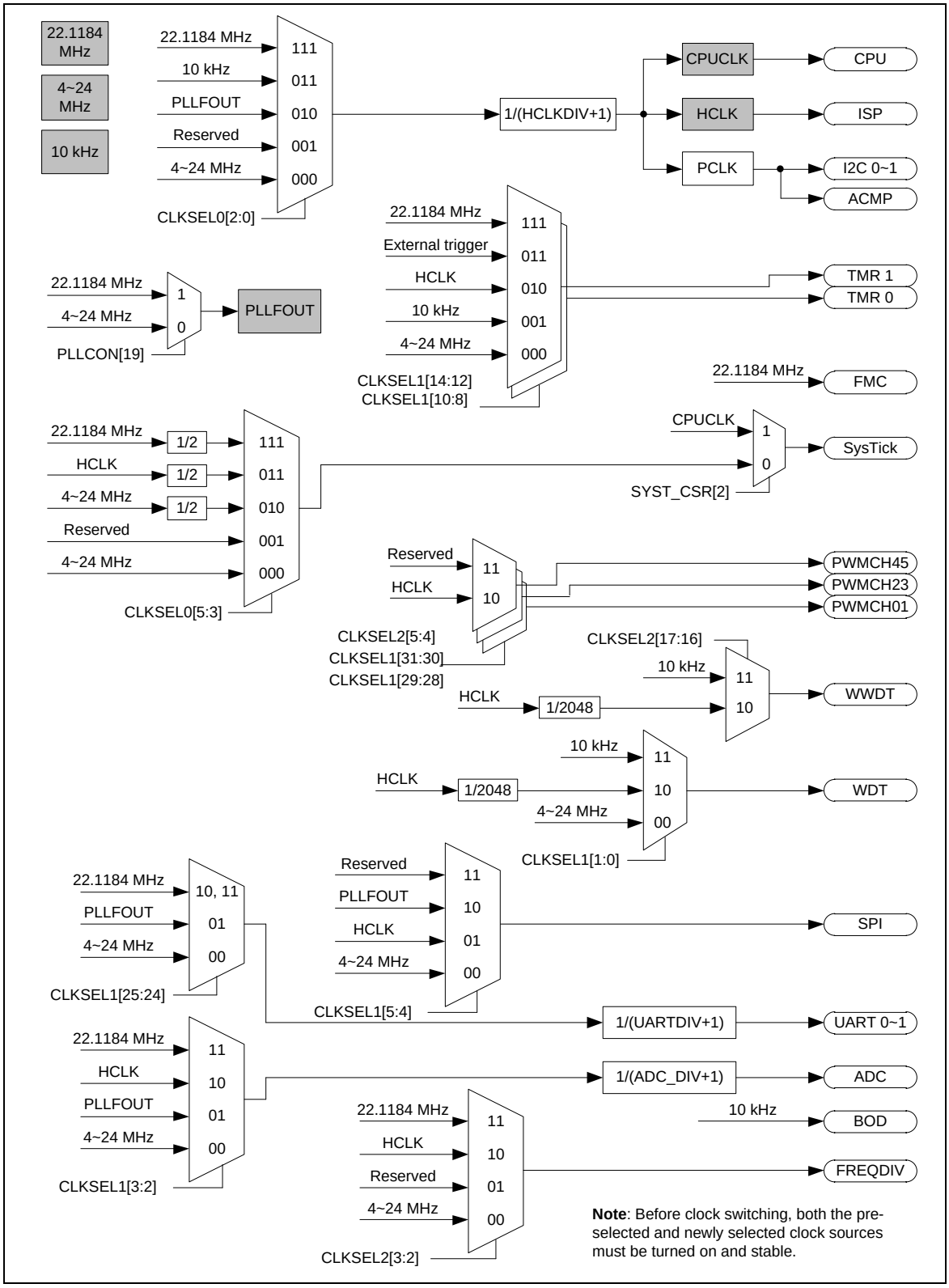


图 6.3-2 时钟发生器全局框图

6.3.2 自动校准

该芯片支持自动频率校准功能：HIRC（内部 22.1184MHz RC 振荡器）可以通过精确的LXT（外部32.768 kHz晶体振荡器）自动调整输出频率在全温度范围里，误差在1%内。例如：系统需要一个精确的22.1184 MHz时钟，但用户不想使用22.1184 MHz HXT作为系统时钟源，他们可以在系统上焊接一个32.768 kHz晶振，然后设置FREQSEL (SYS_IRCTCTL[0] 频率校准选择) 为“1”，这样自动频率校准功能就会被使能。 中断状态标志位FREQLOCK (SYS_IRCTISTS[0] HIRC频率锁定状态)为“1”表明HIRC输出频率误差在1%内。为了取得更好的效果，推荐同时设置LOOPSEL (SYS_IRCTCTL[5:4] 频率校准计算循环) 和RETRYCNT (SYS_IRCTCTL[7:6] 调整值更新限制计数) 为“11”。

6.3.3 系统时钟和SysTick 时钟

系统时钟有4个可选时钟源，由时钟发生器产生。时钟源切换取决于寄存器HCLKSEL (CLK_CLKSEL0[2:0]). 其框图如下所示。

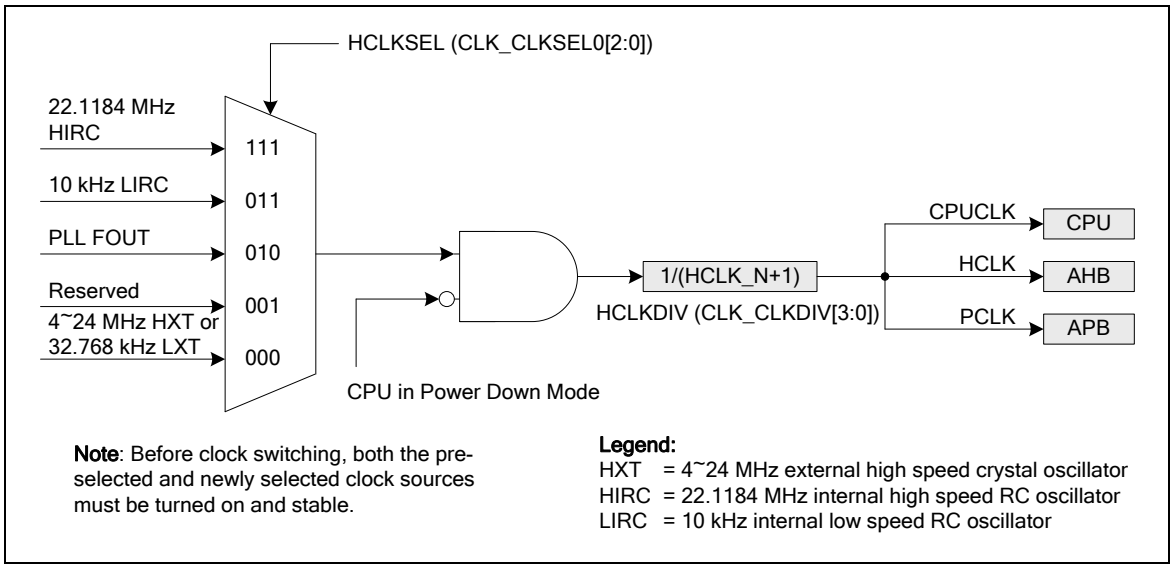


图 6.3-3 系统时钟框图

在系统时钟结构中，PCLK的时钟源等同于HCLK。

Cortex®-M0内核的SysTick 时钟源可以选择CPU时钟或外部时钟(SYST_CTRL[2])。如果使用外部时钟，SysTick 时钟 (STCLK) 有 4 个可选时钟源。时钟源切换取决于寄存器STCLKSEL (CLK_CLKSEL0[5:3])。其框图如下图 6.3-4所示：

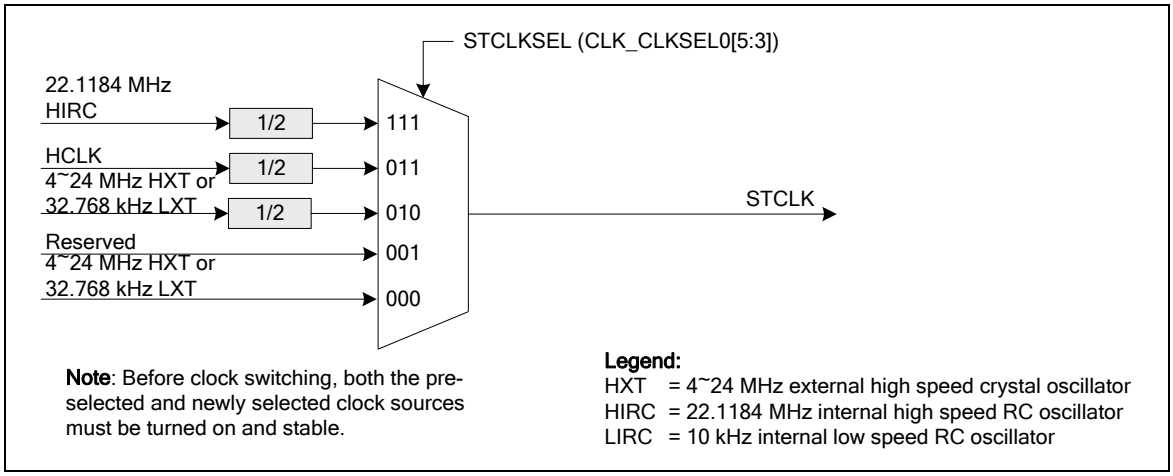


图 6.3-4 SysTick 时钟控制框图

6.3.4 外设时钟源选择

外设时钟可以有不同的时钟源做切换设置。主要取决于不同的外设。请参看寄存器CLK_CLKSEL1 和 CLK_CLKSEL2。在6.3.8章节有详细描述。

请注意, 用户必须等到两个时钟源都处于稳定状态之后，才可从一个时钟源切换到另一个时钟源。

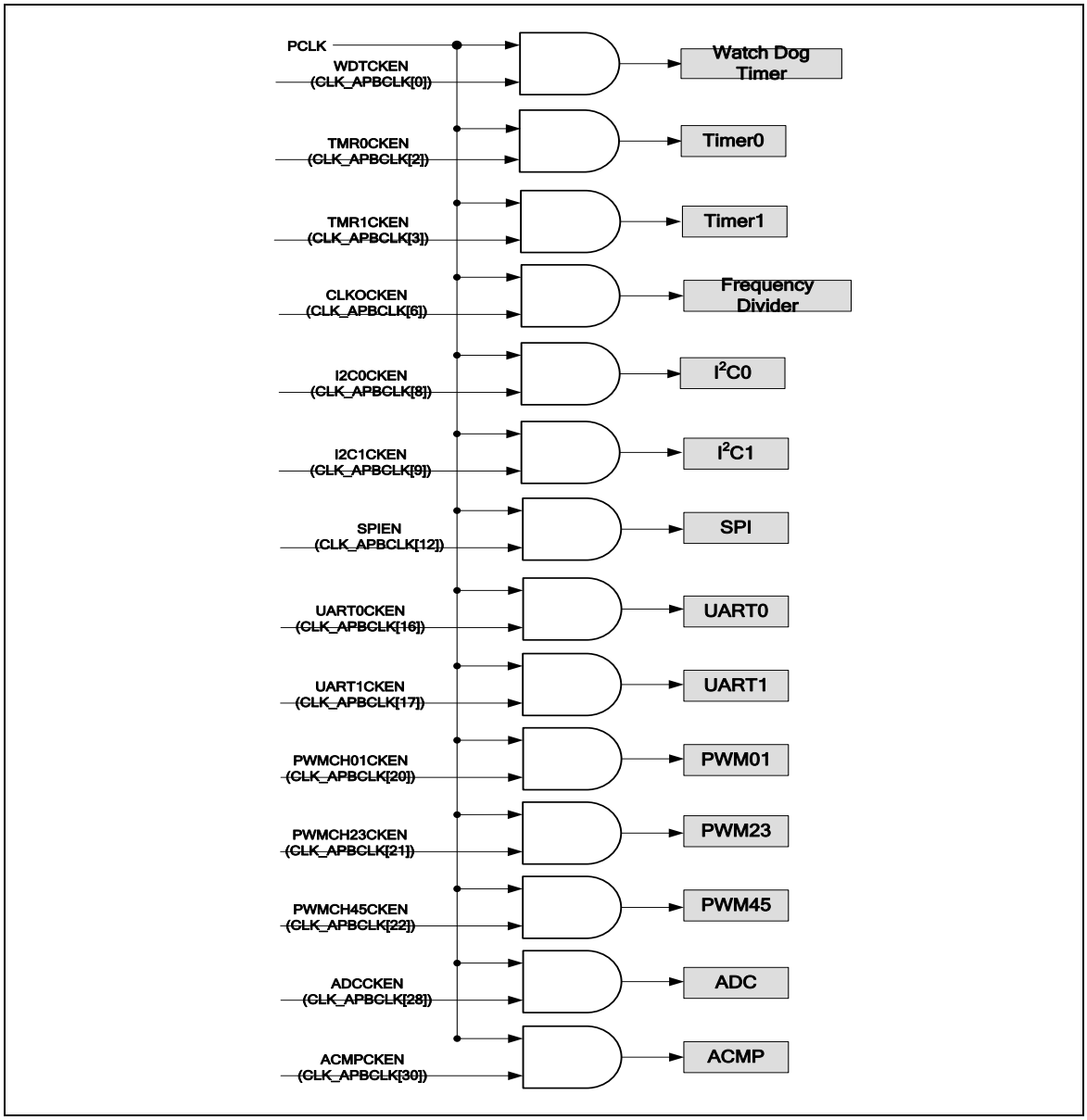


图 6.3-5 PCLK外设总线时钟源选择

	Peripheral Clock 选择表	Ext. CLK(HXT Or LXT)	HIRC	LIRC	HCLK	PLL
WDT	Yes	Yes	No	Yes	Yes	No
WWDT	Yes	Yes	No	Yes	Yes	No
Timer0	Yes	Yes	Yes	Yes	Yes	No
Timer1	Yes	Yes	Yes	Yes	Yes	No
I^2C0	No	-	-	-	-	-
I^2C1	No	-	-	-	-	-
SPI	Yes	Yes	No	No	Yes	Yes
UART0	Yes	Yes	Yes	No	No	Yes
UART1	Yes	Yes	Yes	No	No	Yes
PWM	No	-	-	-	-	-
ADC	Yes	Yes	Yes	No	Yes	Yes
ACMP	No	-	-	-	-	-

表 6.3-1 外设时钟源选择表

注意:对于那些时钟源不可选择的外设，它们的时钟源固定为PCLK。

6.3.5 掉电模式时钟

当芯片进入掉电模式时，系统时钟和一些时钟源以及一些外设时钟将被关闭。也有一些时钟源与外设时钟仍在工作。

如下时钟仍在工作：

- 时钟发生器
- 10 kHz内部低速振荡器时钟（LIRC）
- 32.768 kHz外部低速晶振时钟(LXT)(当PDLXT = 1且XTLEN[1:0]=10)
- 外设时钟 (当模块的时钟源来自10 kHz内部低速振荡器)
  - 看门狗时钟
  - 定时器0/1时钟

6.3.6 分频器输出

该设备带有一个2的若干次幂的频率分频器，该分频器由16个链式的二分频器组成。其中哪一级的值被输出，由一个16选1的多路转换器决定，该多路转换器的输出连接到CLKO管脚上。因此共有16种时钟分频选择，分别是$F_{in}/2^1$、$F_{in}/2^2$ …$F_{in}/2^{16}$，此处Fin是到时钟分频器的时钟输入频率。

输出公式：$\mathbf{F_{out} = F_{in}/2^{(N+1)}}$，其中Fin为输入时钟频率，Fout为时钟分频器输出频率，N为FREQSEL (CLK_CLKOCTL[3:0])的值。

将CLKOEN (CLK_CLKOCTL[4])置1，计数器开始计数。将CLKOEN (CLK_CLKOCTL[4])置0，计数器保持计数状态，直到分频时钟达到低电平并会保持在低电平状态。

若将DIV1EN (CLK_CLKOCTL[5])置1，分频时钟（FRQDIV_CLK）将绕过分频器，使得分频器的时钟在CLKO管脚上直接输出。

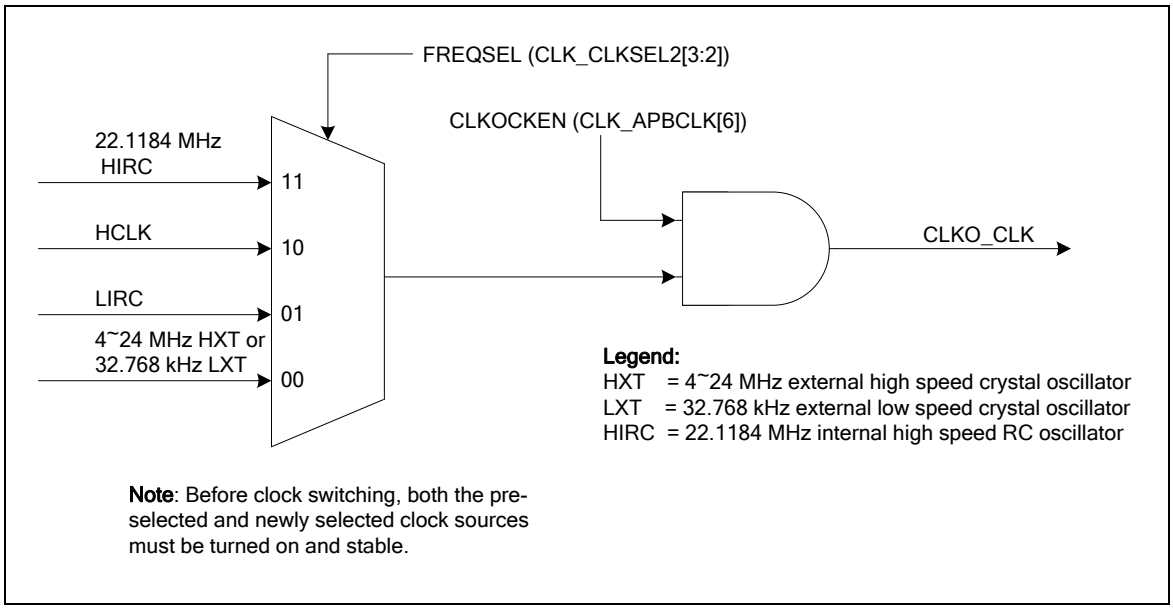


图 6.3-6 分频器的时钟源框图

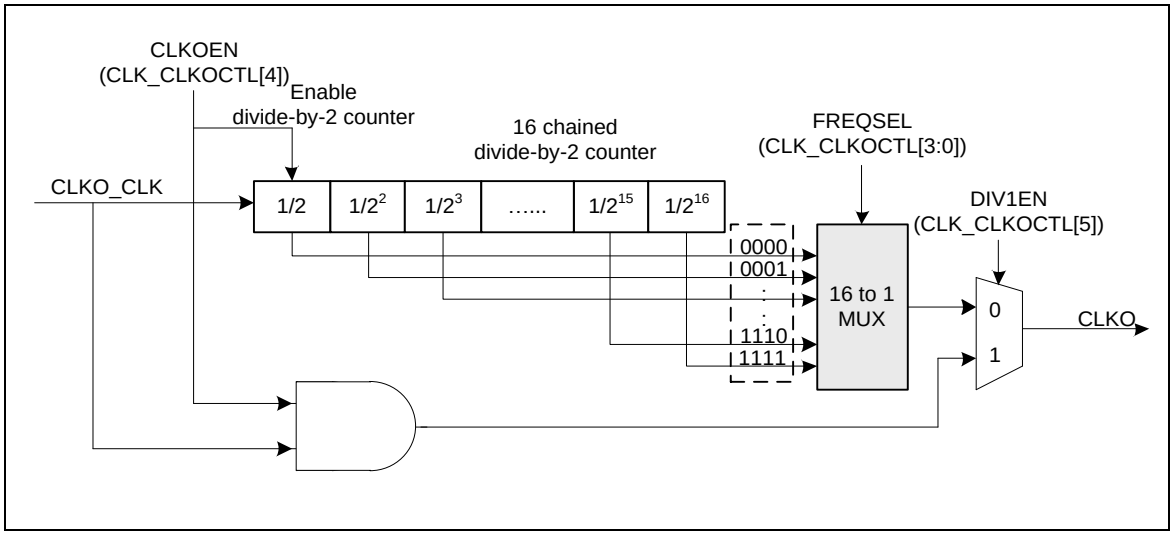


图 6.3-7 分频器框图

6.3.7 寄存器映射

R:只读, **W:**只写, **R/W:**读/写

寄存器	偏移地址	读/写	描述	复位值
CLK Base Address: **CLK_BA = 0x5000_0200**				
CLK_PWRCTL	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x0000_001C
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB设备时钟使能控制寄存器	0x0000_0005
CLK_APBCLK	CLK_BA+0x08	R/W	APB 设备时钟使能控制寄存器	0x0000_0001
CLK_STATUS	CLK_BA+0x0C	R/W	时钟状态监测寄存器	0x0000_0018
CLK_CLKSEL0	CLK_BA+0x10	R/W	时钟源选择控制寄存器0	0x0000_003F
CLK_CLKSEL1	CLK_BA+0x14	R/W	时钟源选择控制寄存器1	0xAFFF_FFFF
CLK_CLKDIV	CLK_BA+0x18	R/W	时钟分频寄存器	0x0000_0000
CLK_CLKSEL2	CLK_BA+0x1C	R/W	时钟源选择控制寄存器2	0x0002_00EF
CLK_PLLCTL	CLK_BA+0x20	R/W	PLL控制寄存器	0x0005_C230
CLK_CLKOCTL	CLK_BA+0x24	R/W	分频器控制寄存器	0x0000_0000

6.3.8 寄存器描述

系统掉电控制寄存器(CLK_PWRCTL)

除BIT[6]外,其他位都是写保护的，在编程这些位之前需将0x59, 0x16, 0x88写入到地址0x5000_0100，以解锁寄存器状态。详细内容请参考 SYS_REGLCTL寄存器，其地址为SYS_BA + 0x100。

寄存器	偏移地址	读/写	描述	复位值
CLK_PWRCTL	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x0000_001C

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**PDLXT**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDEN	**PDWKIF**	**PDWKIEN**	**PDWKDLY**	**LIRCEN**	**HIRCEN**	**XTLEN**	

位	描述	
[31:10]	**Reserved**	保留
[9]	**PDLXT**	**掉电模式下使能 LXT** 该位用于控制晶振时钟在掉电模式下是否有效。. 0 = 掉电模式下无效 1 = 当XTLEN[1:0] = 10时，LXT在掉电模式下依旧有效。.
[8]	**Reserved**	保留
[7]	**PDEN**	**系统掉电模式使能位(写保护)** 当芯片从掉电模式唤醒时，该位由硬件清除，用户需要重新设置该位以便进入下一次的掉电模式。 在掉电模式下，外部4~24 MHz 高速晶振(HXT),32.768KHz的外部低速晶振（LXT）与内部22.1184 MHz的高速振荡器将被禁用，但是内部10 kHz的低速振荡器依旧有效。 在掉电模式下，PLL 与系统时钟将被禁用，忽略时钟源选择。如果外设时钟源为内部10 kHz 低速振荡器，则外设的时钟不受掉电模式的控制。 0 = 执行 WFI 命令，芯片工作于正常模式或者进入 idle 模式 1 = 芯片立即进入掉电模式或者等待 CPU 休眠命令
[6]	**PDWKIF**	**芯片掉电模式唤醒中断状态** 该位由掉电唤醒事件置位，表示从掉电模式中恢复。 如果GPIO, UART, WDT,ACMP, TIMER, 或BOD 唤醒事件发生时，该标志位置1。 **注意：**只有在PDWKEN (CLK_PWRCTL[5]) 被置 1 的时候，该位置1。

位	描述	
[5]	**PDWKIEN**	**掉电模式唤醒中断使能位（写保护）** 0 = 禁用掉电模式唤醒中断 1 = 使能掉电模式唤醒中断 **注意:**当PDWKIF 和 PDWKIEN都为1时，该中断将产生.
[4]	**PDWKDLY**	**唤醒延时计数器使能位（写保护）** 当芯片从掉电模式中唤醒时，时钟控制将会延迟一定的时钟周期以保证系统时钟稳定。 当芯片工作在外部4~24 MHz高速晶振，及外部32.768KHz低速晶振时，延迟时钟周期为4096 个时钟周期；当芯片工作在内部22.1184 MHz 高速振荡器时，延迟时钟周期为16 个时钟周期。 0 = 禁用时钟周期的延迟 1 = 使能时钟周期的延迟
[3]	**LIRCEN**	**LIRC使能位（写保护）** 0 = 禁用10 kHz 内部低速 RC 振荡器(LIRC) 1 = 使能10 kHz内部低速 RC 振荡器(LIRC)
[2]	**HIRCEN**	**HIRC使能位 (写保护)** 0 = 禁用22.1184 MHz 内部高速RC振荡器(HIRC) 1 =使能22.1184 MHz 内部高速RC振荡器(HIRC) **注意:** HIRCEN位的默认值为1。
[1:0]	**XTLEN**	**外部晶振HXT或LXT使能位（写保护）** 默认的时钟源为HIRC。XTLEN的默认值为”00”，且XT1_IN、XT1_OUT管脚默认为GPIO功能. 00 = XT1_IN和XT1_OUT为GPIO, 禁止LXT/HXT(默认值)。 01 = HXT使能。 10 = LXT使能。 11 = XT1_IN为外部时钟输入管脚、XT1_OUT为GPIO管脚. 注意:为使能外部XTAL功能，必须在SYS_P5_MFP 中设定ALT[1:0]和MFP[1:0] 。

寄存器模式/指令模式	SLEEPDEEP (SCR[2])	PDEN (CLK_PWRCTL [7])	CPU Run WFI Instruction	时钟禁用
正常运行模式	0	0	NO	所有时钟皆可被禁用
空闲模式 (CPU 进入休眠模式)	0	0	YES	仅禁用 CPU 时钟
掉电模式 (CPU 进入深度休眠模)	1	1	YES	大部分时钟被禁用，仅10KHz时钟源和选用其为时钟源的WDT外设时钟还能工作。

表 6.3-2 掉电模式控制

当芯片进入掉电模式时，用户可以通过一些中断源唤醒芯片。用户必须在设置PDEN位 (CLK_PWRCTL[7]) 之前，使能相关的中断源及相应的NVIC IRQ 使能位(NVIC_ISER) 以保证芯片能进入掉电模式并能够成功被唤醒。

AHB设备时钟使能控制寄存器(CLK_AHBCLK)

该寄存器的各位用于使能/禁用系统时钟.

寄存器	偏移地址	读/写	描述	复位值
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB设备时钟使能控制寄存器	0x0000_0005

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

位	描述	
[31:3]	**Reserved**	保留
[2]	**ISPCKEN**	**Flash ISP 控制器时钟使能位** 0 = 禁用Flash ISP外设时钟 1 = 使能Flash ISP外设时钟
[1:0]	**Reserved**	保留

APB设备时钟使能控制寄存器 (CLK_APBCLK)

该寄存器用于使能/禁用外设控制器时钟.

寄存器	偏移地址	读/写	描述	复位值
CLK_APBCLK	CLK_BA+0x08	R/W	APB设备时钟使能控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved	**ACMPCKEN**	**Reserved**	**ADCCKEN**	**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**PWMCH45CKEN**	**PWMCH23CKEN**	**PWMCH01CKEN**	**Reserved**		**UART1CKEN**	**UART0CKEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**SPICKEN**	**Reserved**		**I2C1CKEN**	**I2C0CKEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CLKOCKEN**	**Reserved**		**TMR1CKEN**	**TMR0CKEN**	**Reserved**	**WDTCKEN**

位	描述	
[31]	**Reserved**	保留
[30]	**ACMPCKEN**	**模拟比较器时钟使能位** 0 = 禁用模拟比较器时钟 1 = 使能模拟比较器时钟
[29]	**Reserved**	保留
[28]	**ADCCKEN**	**模拟数字转换器(ADC)时钟使能位** 0 = 禁用ADC外设时钟 1 = 使能ADC外设时钟
[27:23]	**Reserved**	保留
[22]	**PWMCH45CKEN**	**PWM_45 时钟使能位** 0 = 禁用PWM45时钟 1 = 使能PWM45时钟.
[21]	**PWMCH23CKEN**	**PWM_23 时钟使能位** 0 = 禁用PWM23时钟 1 = 使能PWM23时钟
[20]	**PWMCH01CKEN**	**PWM_01 时钟使能位** 0 = 禁用PWM01时钟 1 = 使能PWM01时钟.
[19:18]	**Reserved**	保留

位	描述	
[17]	**UART1CKEN**	**UART1 时钟使能位** 0 = 禁用UART1时钟 1 = 使能UART1 时钟
[16]	**UART0CKEN**	**UART0 时钟使能位** 0 = 禁用UART0时钟 1 = 使能UART0 时钟
[15:13]	**Reserved**	保留
[12]	**SPICKEN**	**SPI 时钟使能位** 0 = 禁用SPI 时钟 1 = 使能SPI 时钟
[11:10]	**Reserved**	保留
[9]	**I2C1CKEN**	**I2C1 时钟使能位** 0 =禁用 I2C1 时钟 1 = 使能I2C1 时钟
[8]	**I2C0CKEN**	**I2C0 时钟使能位** 0 = 禁用I2C0 时钟 1 = 使能I2C0 时钟
[7]	**Reserved**	保留
[6]	**CLKOCKEN**	**CLKO时钟使能位** 0 = 禁用CLKO时钟 1 = 使能CLKO 时钟
[5:4]	**Reserved**	保留
[3]	**TMR1CKEN**	**Timer1 时钟使能位** 0 = 禁用Timer1 时钟 1 = 使能Timer1 时钟
[2]	**TMR0CKEN**	**Timer0 时钟使能位** 0 = 禁用Timer0 时钟 1 = 使能Timer0 时钟
[1]	**Reserved**	保留
[0]	**WDTCKEN**	**Watchdog 定时器时钟使能位(写保护)** **0 =** 禁用Watchdog 时钟 **1 =** 使能Watchdog 时钟 **注意:** 该位写保护，编程时需将0x59,0x16和0x88写入地址0x5000_0100，以解锁寄存器写保护状态，参看SYS_REGLCTL寄存器，其地址为SYS_BA + 0x100。

时钟状态寄存器 (CLK_STATUS)

该寄存器用于监控芯片时钟源是否稳定，以及时钟源切换是否失败。

寄存器	偏移地址	读/写	描述	复位值
CLK_STATUS	CLK_BA+0x0C	R/W	时钟状态监测寄存器	0x0000_0018

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

位	描述	
[31:8]	**Reserved**	保留
[7]	**CLKSFAIL**	**时钟切换失败标志（只读）** 当用户切换系统时钟源时，该位被更新。如果切换目标时钟稳定，该位将被置0。如果切换目标时钟不稳定，该位将被设置为1。 0 =时钟切换成功 1 =时钟切换失败 **注意：**该位是只读位，在时钟源稳定之后，硬件自动将系统时钟切换至所选时钟源，且该位自动由硬件写1清除。
[6:5]	**Reserved**	保留
[4]	**HIRCSTB**	**HIRC时钟源稳定标志（只读）** 0 = HIRC 时钟不稳定或者禁用 1 = HIRC 时钟使能并稳定
[3]	**LIRCSTB**	**LRC时钟源稳定标志（只读）** 0 = LIRC 时钟不稳定或者禁用 1 = LIRC 时钟使能并稳定
[2]	**PLLSTB**	**内部 PLL 时钟源稳定标志(只读)** 0 = 内部 PLL时钟不稳定或者禁用 1 = 内部PLL 时钟使能并稳定
[1]	**Reserved**	保留
[0]	**XTLSTB**	**HXT或LXT时钟源稳定标志（只读）** 0 = HXT 或LXT 时钟不稳定或者禁用 1 = HXT或 LXT 时钟使能并稳定

时钟源选择控制寄存器0 (CLK_CLKSEL0)

寄存器	偏移地址	读/写	描述	复位值
CLK_CLKSEL0	CLK_BA+0x10	R/W	时钟源选择控制寄存器0	0x0000_003F

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

位	描述	
[31:6]	**Reserved**	保留
[5:3]	**STCLKSEL**	**Cortex®-M0 SysTick时钟源选择(写保护)** 如果CLKSRC (SYST_CSR[2]) = 1, SysTick 时钟源为HCLK. 如果CLKSRC (SYST_CSR[2]) = 0, SysTick 时钟源配置如下： 000 = 时钟源为HXT或LXT 001 = 保留 010 = 时钟源为HXT/2或LXT/2 011 = 时钟源为HCLK/2 111 = 时钟源为HIRC/2 Other = 保留 **注意1:**这些位是写保护位，编程时需将0x59,0x16和0x88写入地址0x5000_0100，以解锁寄存器写保护状态，参看SYS_REGLCTL寄存器。 **注意2：**如果 SysTick 时钟源不是来自 HCLK (例如 SLKCRC(SYST_ CTRL [2] = 0)), SysTick 时钟源必须小于或等于HCLK/2。 **注意3:** 设置CLK_PWRCTL[1:0]，选择HXT或LXT.
[2:0]	**HCLKSEL**	**HCLK时钟源选择(写保护)** 000 = 时钟源为HXT或LXT 001 = 保留 010 = 时钟源为 PLL 011 = 时钟源为LIRC 111 = 时钟源为HIRC 其它 = 保留 **注意1:** 在时钟切换前，相关时钟源（预选和新选）必须打开并保持稳定。 **注意2：**该位写保护，编程时需将0x59,0x16和0x88写入地址0x5000_0100，以解锁寄存器写保护状态，参看SYS_REGLCTL寄存器，地址SYS_BA + 0x100。 **注意3：**设置CLK_PWRCTL[1:0]选择HXT或LXT。

时钟源选择控制寄存器1(CLK_CLKSEL1)

在时钟切换之前，必须打开相关的时钟源（预选和新选)

寄存器	偏移地址	读/写	描述	复位值
CLK_CLKSEL1	CLK_BA+0x14	R/W	时钟源选择控制寄存器 1	0xAFFF_FFFF

31	30	29	28	27	26	25	24
PWMCH23SEL		**PWMCH01SEL**		**Reserved**		**UARTSEL**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**TMR1SEL**			**Reserved**	**TMR0SEL**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**SPISEL**		**ADCSEL**		**WDTSEL**	

位	描述	
[31:30]	**PWMCH23SEL**	**PWM2和 PWM3时钟源选择** PWM2和 PWM3具有相同的外设时钟源和预分频器。. 00 = 保留 01 = 保留 10 = 时钟源为 HCLK. 11 = 保留
[29:28]	**PWMCH01SEL**	**PWM0和 PWM1时钟源选择** PWM0和 PWM1具有相同的外设时钟源和预分频器。. 00 = 保留 01 = 保留 10 = 时钟源为 HCLK. 11 = 保留.
[27:26]	**Reserved**	保留.
[25:24]	**UARTSEL**	**UART时钟源选择** 00 = 时钟源为 HXT或 LXT. 01 = 时钟源为PLL. 10 = 时钟源为HIRC. 11 = 时钟源为 HIRC. **注意:** 设置 CLK_PWRCTL[1:0]选择 HXT或 LXT时钟源。
[23:15]	**Reserved**	保留

位	描述	
[14:12]	**TMR1SEL**	**TIMER1 时钟源选择** 000 = 时钟源为HXT或LXT 001 = 时钟源为LIRC 010 = 时钟源为HCLK 011 = 时钟源为外部时钟管脚. 111 = 时钟源为内部22.1184 MHz 高速振荡器时钟(HIRC) 其它 = 保留 **注意：**设置 CLK_PWRCTL[1:0]选择 HXT 或 LXT晶振时钟。
[11]	**Reserved**	保留
[10:8]	**TMR0SEL**	**TIMER0 时钟源选择** 000 = 时钟源为HXT或LXT 001 = 时钟源为LIRC 010 = 时钟源为HCLK 011 = 时钟源为外部时钟管脚. 111 = 时钟源为内部22.1184 MHz 高速振荡器时钟(HIRC) 其它 = 保留 **注意：**设置 CLK_PWRCTL[1:0]选择 HXT 或 LXT晶振时钟。
[7:6]	**Reserved**	保留
[5:4]	**SPISEL**	**SPI时钟源选择** 00 = 时钟源为HXT或LXT 01 = 时钟源为HCLK 10 = 时钟源为PLL 11 = 保留 **注意:**设置 CLK_PWRCTL[1:0]选择 HXT或 LXT.
[3:2]	**ADCSEL**	**ADC 外设时钟源选择** 00 = 时钟源为 HXT或 LXT. 01 = 时钟源为PLL. 10 = 时钟源为 HCLK. 11 = 时钟源为HIRC. **注意:**设置 CLK_PWRCTL[1:0]选择 HXT或 LXT.
[1:0]	**WDTSEL**	**WDT 时钟源选择（写保护）** 00 = 时钟源为HXT或LXT 01 = 保留 10 = 时钟源为HCLK/2048 11 = 时钟源为LIRC **注意:** 该位写保护，编程时需将0x59,0x16和0x88写入地址0x5000_0100，以解锁寄存器写保护状态，参看SYS_REGLCTL寄存器，地址SYS_BA + 0x100。 **注意：**设置 CLK_PWRCTL[1:0]选择 HXT或 LXT.

时钟分频数寄存器(CLK_CLKDIV)

寄存器	偏移地址	读/写	描述	复位值
CLK_CLKDIV	CLK_BA+0x18	R/W	时钟分频数寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留
[23:16]	**ADCDIV**	**ADC时钟分频数，时钟源来自ADC外设时钟源** ADC 外设时钟频率= (ADC 时钟源频率) / (ADCDIV + 1).
[15:12]	**Reserved**	保留
[11:8]	**UARTDIV**	**UART时钟分频数，时钟源来自UART时钟源** UART时钟频率= (UART时钟源频率) / (UARTDIV + 1).
[7:4]	**Reserved**	保留
[3:0]	**HCLKDIV**	**HCLK时钟分频数，时钟源来自HCLK时钟源** HCLK时钟频率= (HCLK时钟源频率) / (HCLKDIV + 1).

时钟源选择控制寄存器2(CLK_CLKSEL2)

在时钟切换之前，必须打开相关的时钟源（预选和新选）

寄存器	偏移地址	读/写	描述	复位值
CLK_CLKSEL2	CLK_BA+0x1C	R/W	时钟源选择控制寄存器2	0x0002_00EF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**WWDTSEL**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**PWMCH45SEL**		**FREQSEL**		**Reserved**	

位	描述	
[31:18]	**Reserved**	Reserved.
[17:16]	**WWDTSEL**	**窗口看门狗定时器时钟源选择** 00 = 保留 01 = 保留 10 = 时钟源为HCLK/2048 clock. 11 = 时钟源为10 kHz内部低速RC振荡器 (LIRC).
[15:6]	**Reserved**	保留.
[5:4]	**PWMCH45SEL**	**PWM4 和 PWM5时钟源选择** PWM4和 PWM5具有相同的时钟源和预分频器. 00 = 保留. 01 = 保留. 10 = 时钟源为HCLK. 11 = 保留.
[3:2]	**FREQSEL**	**时钟分频器的时钟源选择** 00 = 时钟源为 HXT或 LXT. 01 = 时钟源为LIRC. 10 = 时钟源为 HCLK. 11 = 时钟源为 HIRC. **Note:** 设置 CLK_PWRCTL[1:0]选择 HXT或 LXT.
[1:0]	**Reserved**	保留.

PLL 控制寄存器(CLK_PLLCTL)

PLL 的参考时钟源来自外部 4~24 MHz高速晶振时钟输入或者内部 22.1184 MHz 高速振荡器。该寄存器用于控制 PLL 的输出频率和 PLL 的操作模式

寄存器	偏移地址	读/写	描述	复位值
CLK_PLLCTL	CLK_BA+0x20	R/W	PLL控制寄存器	0x0005_C230

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

位	描述	
[19]	**PLLSRC**	**PLL 时钟源选择** 0 = PLL 时钟源为外部 4~24 MHz 高速晶振(HXT) 1 = PLL 时钟源为内部 22.1184 MHz 高速振荡器(HIRC)
[18]	**OE**	**PLL OE (FOUT 使能) 管脚控制** 0 = PLL FOUT 使能 1 = PLL FOUT 固定为低
[17]	**BP**	**PLL 旁路控制** 0 = PLL 正常模式（默认） 1 = PLL 时钟输出与PLL时钟源输入相同
[16]	**PD**	**掉电模式** 如果设置寄存器CLK_PWRCTL的PDEN位为 1，PLL 将进入掉电模式。 0 = PLL 正常模式 1 = PLL 进入掉电模式（默认）
[15:14]	**OUTDIV**	**PLL 输出分频控制** 参考下表公式
[13:9]	**INDIV**	**PLL 输入分频控制** 参考下表公式
[8:0]	**FBDIV**	**PLL 反馈分频控制** 参考下表公式

输出时钟频率设置

:$$\boxed{FOUT = FIN \times \frac{NF}{NR} \times \frac{1}{NO}}$$

约束条件:

1. $4MHz < FIN < 24MHz$

2. $800KHz < Fref = \frac{FIN}{2 \times NR} < 7.5MHz$

3. $100MHz < FCO = Fref \times 2 \times NF = FIN \times \frac{NF}{NR} < 200MHz$

   $120MHz < FCO\ is\ preferred$

符号	描述
FOUT	输出时钟频率
FIN	输入（参考）时钟频率
NR	输入分频(INDIV + 2)
NF	反馈分频(FBDIV + 2)
NO	OUTDIV = “00” : NO = 1 OUTDIV = “01” : NO = 2 OUTDIV = “10” : NO = 2 OUTDIV = “11” : NO = 4

默认频率设置

默认值: 0xC230
FIN = 12 MHz
NR = (1+2) = 3
NF = (48+2) = 50
NO = 4
FOUT= 12/4 *50* 1/3 = 50 MHz

分频控制寄存器 (CLK_CLKOCTL)

寄存器	偏移地址	读/写	描述	复位值
CLK_CLKOCTL	CLK_BA+0x24	R/W	分频控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**DIV1EN**	**CLKOEN**	**FREQSEL**			

位	描述	
[31:6]	**Reserved**	保留.
[5]	**DIV1EN**	**分频器1使能位** 0 =分频器的输出频率由FREQSEL的值决定 1 = 分频器输出频率与输入时钟频率相同
[4]	**CLKOEN**	**分频器使能位** 0 = 禁用时钟分频器 1 = 使能时钟分频器
[3:0]	**FREQSEL**	**时钟输出频率选择位** 输出频率的公式: $F_{out} = F_{in}/2^{(N+1)}$ F_{in}为输入时钟频率 F_{out}为分频器输出时钟频率 N为FREQSEL[3:0]的值。

6.4 存储控制器 (FMC)

6.4.1 概述

NuMicro® Mini58系列具有32K字节的片上flash，用于存储应用程序和存储大小可配置的数据Flash。一个用户配置区，用于系统初始化。一个2K字节的引导存储器(LDROM)，用于在系统编程(ISP)功能。一个512字节安全保护ROM(SPROM)用来保障用户程序。支持在应用编程(IAP)，更新flash程序后，执行引导程序和用户程序之间切换时，无需复位。

6.4.2 特性

- 支持32K字节应用程序存储空间(APROM).
- 支持2K引导存储器(LDROM).
- 支持与APROM共享空间、大小可配置的数据Flash.
- 支持512字节的安全保护ROM(SPROM)，用于保障用户程序.
- 支持12字节可配置区，用于控制系统初始化.
- 对所有片上Flash操作, 支持512字节页擦除.
- 支持CRC-32 checksum计算功能.
- 支持在系统编程(ISP)/在应用编程(IAP)来更新片上Flash.

6.4.3 框图

存储控制器(FMC)包括AHB从接口， flash控制寄存器，flash初始化控制器，flash操作控制和片上flash.Flash存储控制器框图如下图 6.4-1所示。

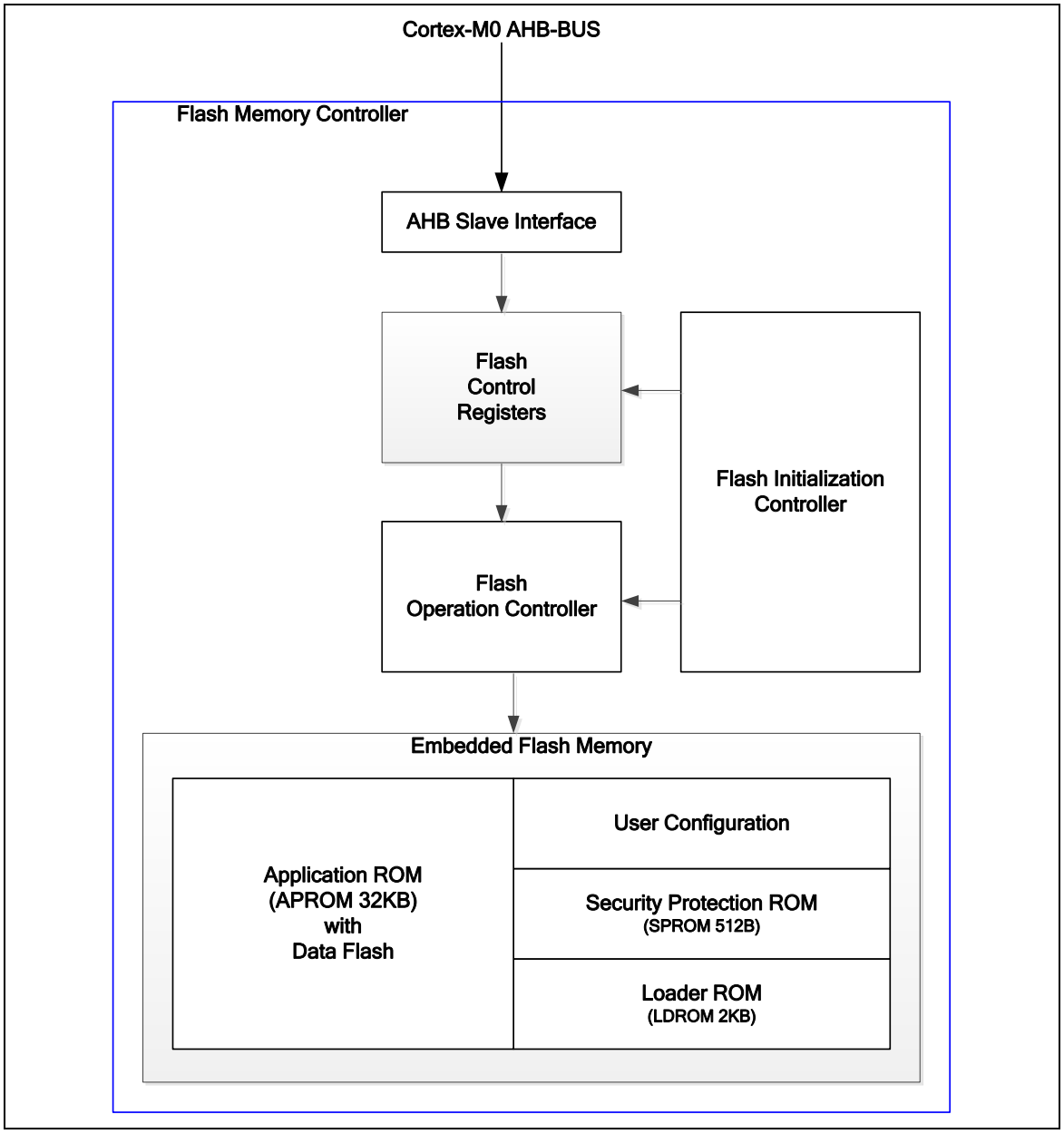


图 6.4-1 存储控制器框图

AHB从接口

在flash存储控制器中有一个AHB从接口，用于指令和数据读取，也用于ISP寄存器的访问。

Flash控制寄存器

所有的ISP控制和状态寄存器都在flash控制寄存器中。详细内容将会在寄存器描述部分介绍。

Flash初始化控制器

当芯片上电或复位，flash初始化控制器将开始自动访问flash，并且检测flash的稳定性,重载用户配置内容到flash控制寄存器用于系统初始化。

Flash操作控制器

对flash操作，例如checksum ，flash擦除，flash 编程和读flash，都有明确的控制时序。Flash操作控制器在收到flash 控制寄存器和flash初始化控制器的请求后, 就会产生这些控制时序。

片上Flash 存储器

片上flash存储器主要用于存储用户应用程序和参数。它包括了用户配置区，2K字节的LDROM ,512字节SPROM，与DATA Flash共享空间的32K APROM。页擦除大小为512字节，32位的可编程宽度。

6.4.4 功能描述

FMC 功能包括存储器组织，启动选择，IAP, ISP,片上flash编程，和checksum计算。在存储器组织中也介绍了flash存储器映射和系统存储器映射。

6.4.4.1 存储器组织

FMC存储包含了片上flash。片上存储器是可编程的，包括APROM, LDROM，SPROM，数据Flash和用户配置区。地址映射包括flash存储映射和4种地址映射：支持IAP功能的LDROM，不支持IAP功能的LDROM，支持IAP功能的APROM, 不支持IAP功能的APROM。

6.4.4.2 LDROM, APROM 和数据 Flash

LDROM 用于通过引导程序，来执行在系统编程(ISP)的功能。LDROM是一个2KB的片上flash存储器，flash地址是从0x0010_0000到0x0010_07FF。APROM是用户应用程序的主要存储器。APROM 的大小是32KB。数据Flash是用于存储应用参数的（不是指令）。数据FLASH与APROM共享存储空间，大小可配置。数据Flash的基地址由寄存器DFBA (CONFIG1[19:0])设定。所有片上flash存储器, 页擦除的大小都是512字节.

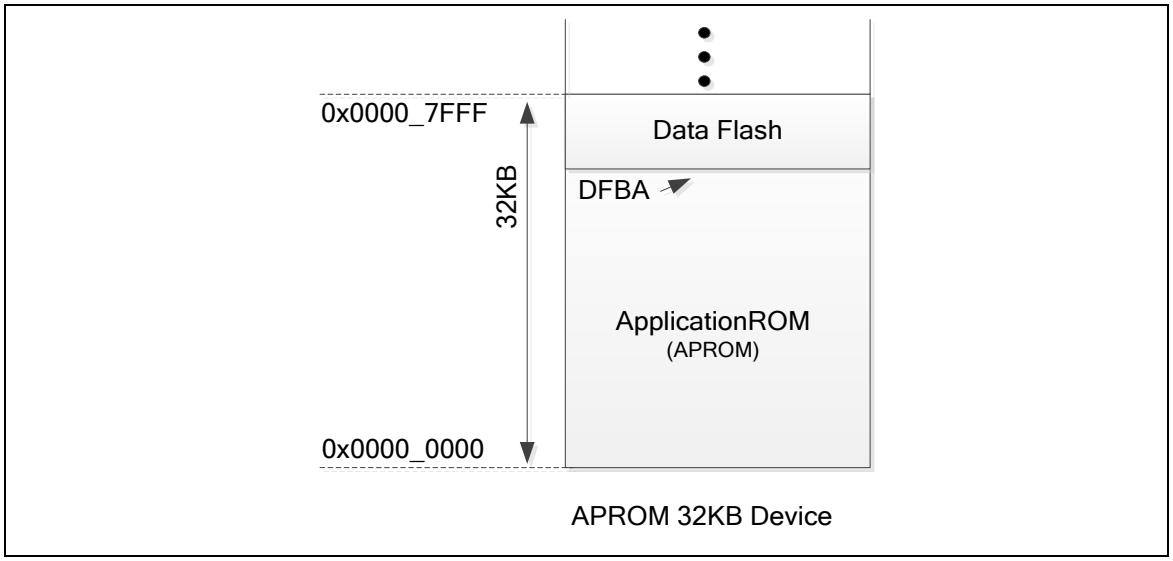


图 6.4-2 数据FLASH与APROM共享存储空间

6.4.4.3 用户配置区

用户配置区是一个内部可编程的配置区域，用于启动选项，比如flash安全锁，启动选择，欠电压电平设置和数据flash基地址设置。类似于保险丝在上电时的工作原理。在上电的时候，用户配置区设定被加载到相应的控制寄存器。用户可以通过不同的应用需求来设定寄存器。用户配置区可以通过ISP方式更新，位于地址0x0030_0000，有3个32位的寄存器 (CONFIG0、CONFIG1和CONFIG2). 用户更改后的配置将在系统重启后生效。

CONFIG0 (地址 = 0x0030_0000)

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

位	描述	
[31:24]	**Reserved**	保留.
[23]	**CBOVEXT**	**欠压选择扩展** 0 = 欠压选择包括 2.2V/2.7V/3.7V/4.4V 1 = 欠压选择包括 2.7V/3.7V/禁用模式 **注意：** 详细内容请查看 CBOV表中内容
[22:21]	**CBOV**	**欠压选择**

CBOVEXT	CBOV[1:0]	Brown-out voltage
0	00	2.2V
0	01	2.7V
0	10	3.7V
0	11	4.4V
1	00	2.7V
1	01	2.7V
1	10	3.7V
1	11	Disable

[20]	**CBORST**	**欠压复位使能** 0 =上电或复位后，使能欠压复位 1 =上电或复位后，禁用欠压复位
[19:11]	**Reserved**	保留.
[10]	**CIOINI**	**I/O初始状态选择** 0 = 上电或复位后，所有GPIO默认为准双向模式 1 = 上电或复位后，所有GPIO默认为三态输入模式
[9:8]	**Reserved**	保留.

[7:6]	**CBS**	**芯片启动选择** 在IAP模式下，当CBS[0] = 0，LDROM的基地址映射到0x100000, APROM的基地址映射到0x0. 用户既可以访问APROM也可以访问LDROM，无需启动切换。换句话说，如果是IAP模式，存放在LDROM与APROM的代码可以互相调用。 当MBS =1时，CBS生效。 00=由LDROM启动支持IAP功能 01=由LDROM启动不支持IAP功能 10=由APROM启动支持IAP功能 11=由APROM启动不支持IAP功能 注： 当CBS[0] = 1，BS (FMC_ISPCTL[1])只用于启动切换的控制。 当CBS[0] = 0，VECMAP (FMC_ISPSTS[23:9])只用于地址0x0~0x1ff重新映射。
[5:2]	**Reserved**	保留.
[1]	**LOCK**	**安全锁控制** 0 = 锁定FLASH存储器内容 1 = 当ALOCK (CONFIG2[7:0])不为0x5A时， 解锁Flash 存储器内容
[0]	**DFEN**	**数据FLASH使能位** 当DFEN=0时 APROM与数据flash共享，数据flash的基地址由DFBA (CONFIG1[19:0])设定。 0 = 使能数据FLASH 1 = 禁用数据FLASH

CONFIG1 (Address = 0x0030_0004)

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

位	描述	
[31:20]	**Reserved**	保留
[19:0]	**DFBA**	**数据Flash基地址** 只有当DFEN (CONFIG0[0])=0时，该寄存器才工作。如果DFEN (CONFIG0[0])=0，数据Flash基地址由用户定义。因为片上FLASH擦除单位为512字节，所以强制DFBA的第8-0位置为0。

CONFIG2 (Address = 0x0030_0008)

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

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**ALOCK**	**高级安全锁控制** 0x5A = 当 LOCK (CONFIG0[1])为1时，Flash存储器内容被解锁 . 其他 = Flash存储器内容被锁定 . **注意:** 在页擦除或整片擦除之后，ALOCK将被设置成 0x5A

6.4.4.4 安全保护存储器 (SPROM)

安全保护存储器（SPROM)用于存储安全指令，SPROM区域大小为512字节，地址为0x20_0000 ~ 0x20_01FF，不支持整片擦除命令。 下 图 6.4-3中SPROM最后一字节 (地址: 0x0020_01FF)，用于确定SPROM代码是处于非安全模式，还是调试安全模式，亦或安全模式下。

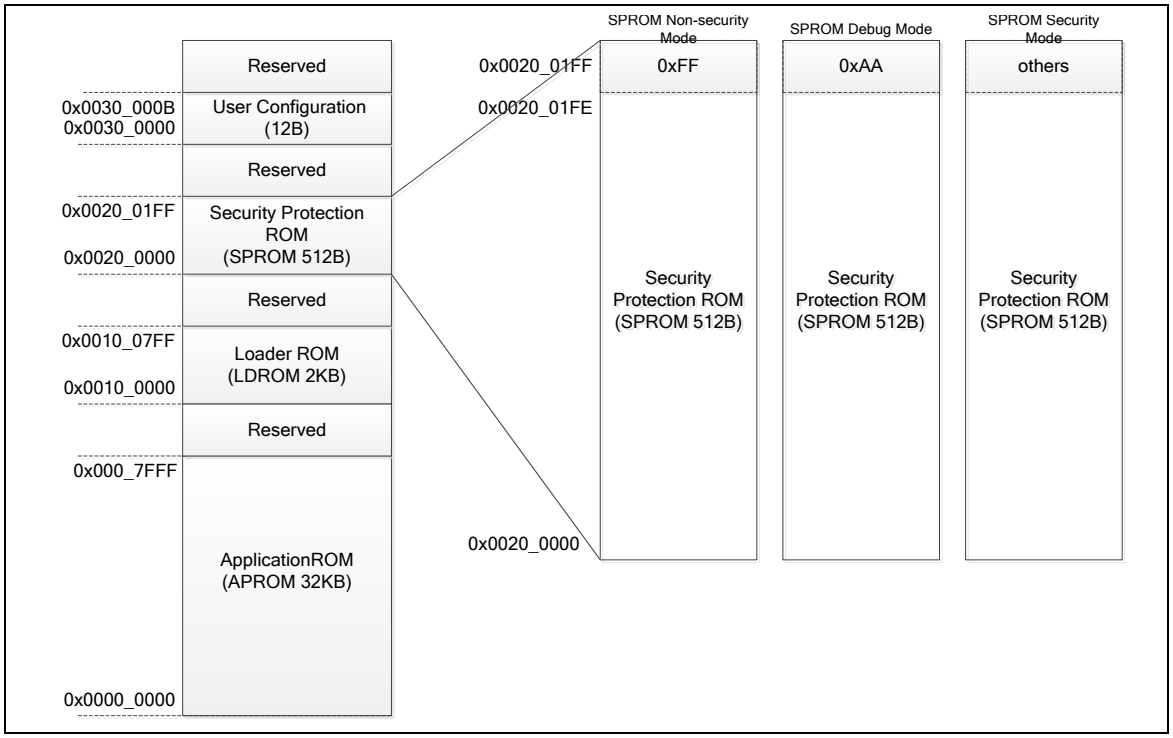


图 6.4-3 SPROM 安全锁模式

(1) SPROM非安全模式（最后一字节为0xFF）,SPROM的访问方式和APROM与LDROM的访问方式相同，所有区域皆可由CPU或ISP命令读取，并可由ISP命令擦除和编程。

(2) SPROM 调试安全模式 (最后一字节为0xAA)，为了方便调试，当Cortex®-M0 ICE端口连接到单片机时，FMC控制器支持SPROM编程。SPROM的其他操作和安全模式下的操作一致。

(3) SPROM安全模式(最后一字节既不是0xFF又不是0xAA)，安全模式下，为了保护SPROM中的代码，CPU运行在SPROM区域时，仅执行取指令以及从SPROM中获取数据的功能。其他情况下，在CPU进行数据访问时将读到全0（0x0000_0000）。在安全模式下，当Cortex®-M0 ICE 端口连接时，CPU也将读到0值，从而起到保护SPROM的作用。在这种模式下，SPROM仅仅支持使用数值为0x0055AA03的页擦除命令，不支持ISP编程和Flash读命令。

SCODE (FMC_ISPSTS[31])是SPROM的安全标志位，该位用来指示SPROM是处在何种模式下，是安全模式、调试模式还是非安全模式。在Flash初始化时，如果SPROM的最后一字节不是0xFF,该位将被置1，并且在页擦除完成后清零。在正常运行时，如果SPROM的末字节为0xFF，它可由用户置1，从而更加方便SPROM的安全模式的测试。

6.4.4.5 Flash存储器映射

Mini58系列中，flash存储器映射有别于系统存储器映射。当CPU从FMC存储器中获取代码或数据时，使用系统存储器映射。当用户用ISP功能去读取、编程、或者擦除FMC存储器时，使用flash存储器映射。Flash存储器映射如下图 6.4-4所示。

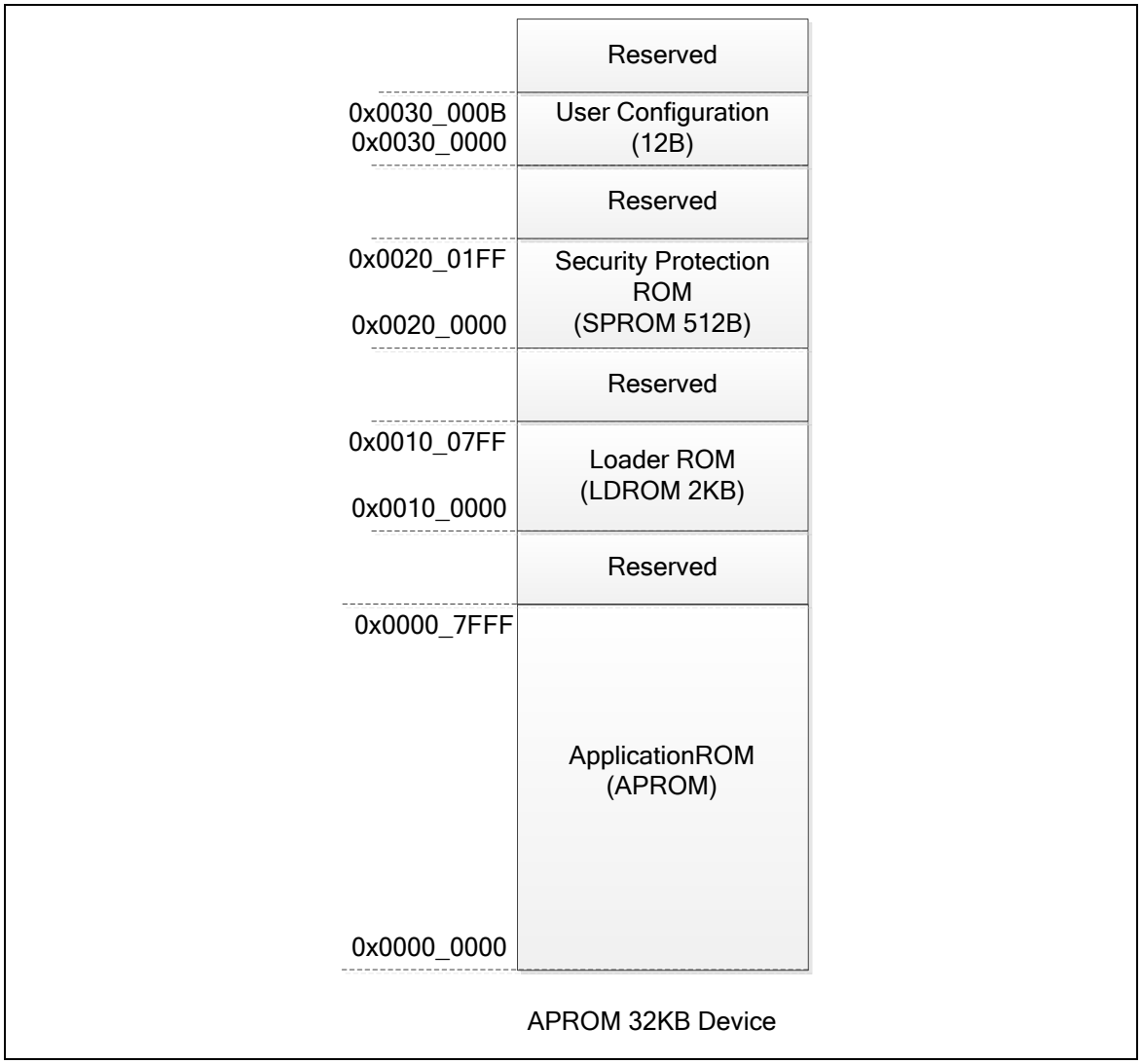


图 6.4-4 Flash存储器映射

6.4.4.6 支持IAP的系统存储器映射

在CPU访问FMC存储器获取代码或数据的时候，用到系统存储器映射。SPROM (0x0020_0000~0x0020_01FF)和LDROM(0x0010_0000~0x0010_07FF)的地址映射和在flash存储器映射中的地址映射相同。数据flash与APROM共享，数据flash的基地址由CONFIG1设定。在flash初始化时，CONFIG1的内容被加载到DFBA(数据Flash基地址寄存器)。DFBA~0x0000_7FFF是Cortex®-M0数据访问的数据Flash区，0x0000_0200~(DFBA-1)是Cortex®-M0指令存取的APROM区。

系统存储向量的地址在0x0000_0000 到 0x0000_01FF。在CPU启动期间，APROM和LDROM可以映射到系统存储向量区。当芯片启动的时候，有两种支持IAP的系统存储器映射模式：(1)支持IAP的LDROM (2)支持IAP的APROM .

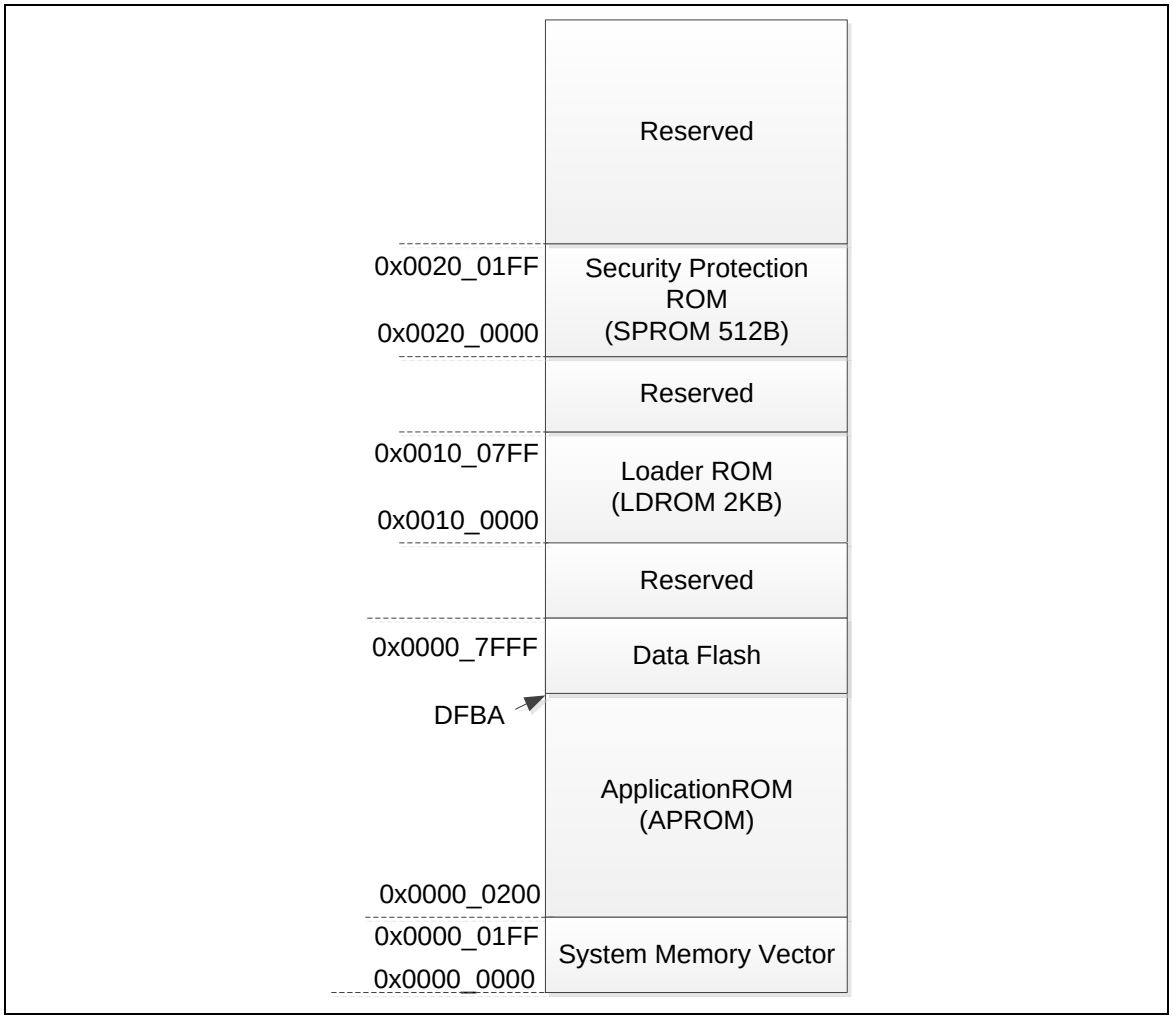


图 6.4-5 支持IAP的系统存储器映射

在支持IAP的LDROM模式中，{VECMAP[11:0], 9'h000}的默认值是0x100000， LDROM的第一页(0x0010_0000~0x0010_01FF)映射到了系统存储向量区，用于Cortex®-M0指令或数据存取。

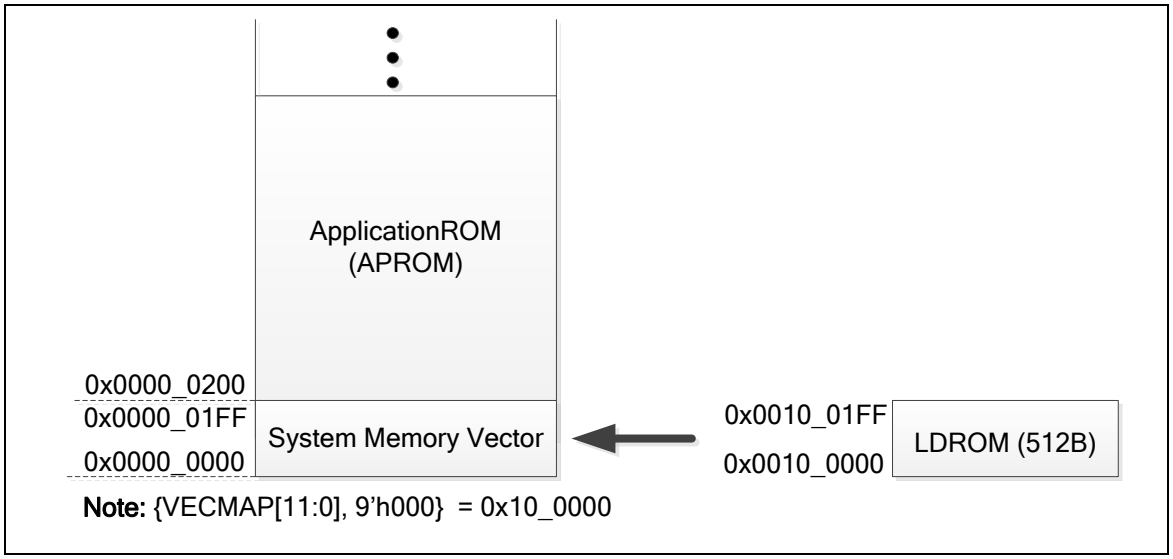


图 6.4-6 支持IAP的LDROM模式

在支持IAP的APROM模式中，{VECMAP[11:0], 9'h000}的默认值是0x100000，APROM 的第一页(0x0000_0000~0x0000_01FF)映射到系统存储向量区，用于Cortex®-M0指令或数据存取。

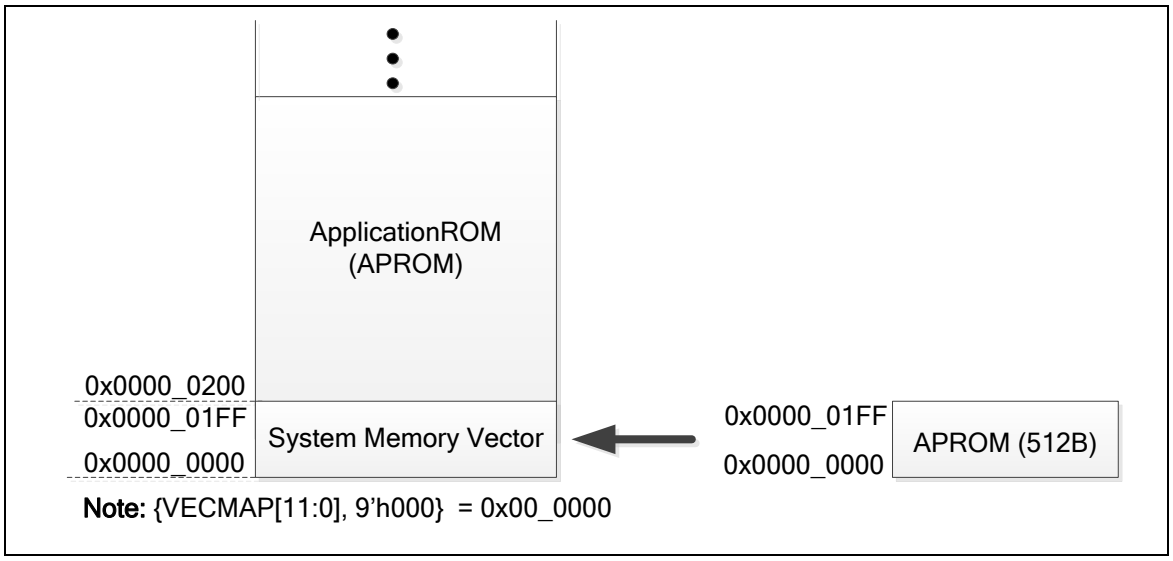


图 6.4-7 支持IAP的APROM模式

在支持IAP模式的系统存储映射中，当CPU运行时，APROM和LDROM可以重映射到系统存储向量区。用户可以写重映射的目标地址到寄存器FMC_ISPADDR，然后通过“向量重映射”命令(0x2E)触发ISP流程。在寄存器VECMAP (FMC_ISPSTS[23:9])中，可以得到最终的系统存储向量映射地址。

6.4.4.7 不支持IAP功能的系统内存映射

不支持IAP功能的系统内存映射，CPU仍然可以访问SPROM(0x0020_0000~0x0020_01FF)，但不支持系统存储向量映射。在芯片启动时，有两种不支持IAP功能的系统内存映射：(1)不支持IAP功能的LDROM (2)不支持IAP功能的APROM。在不支持IAP功能的LDROM模式中，LDROM基地址映射到0x0000_0000，CPU编程不能访问APROM。在不支持IAP功能的APROM中，APROM的基地址映射到0x0000_0000，CPU编程不能访问LDROM。数据Flash与APROM共享，数据Flash的基地址由CONFIG1设定。在flash初始化期间，CONFIG1的内容被加载到DFBA(数据Flash基地址)。DFBA0x0000_7FFF是Cortex®-M0的数据Flash区，0x0000_0000~(DFBA-1)是Cortex®-M0的APROM区,用于指令存取。

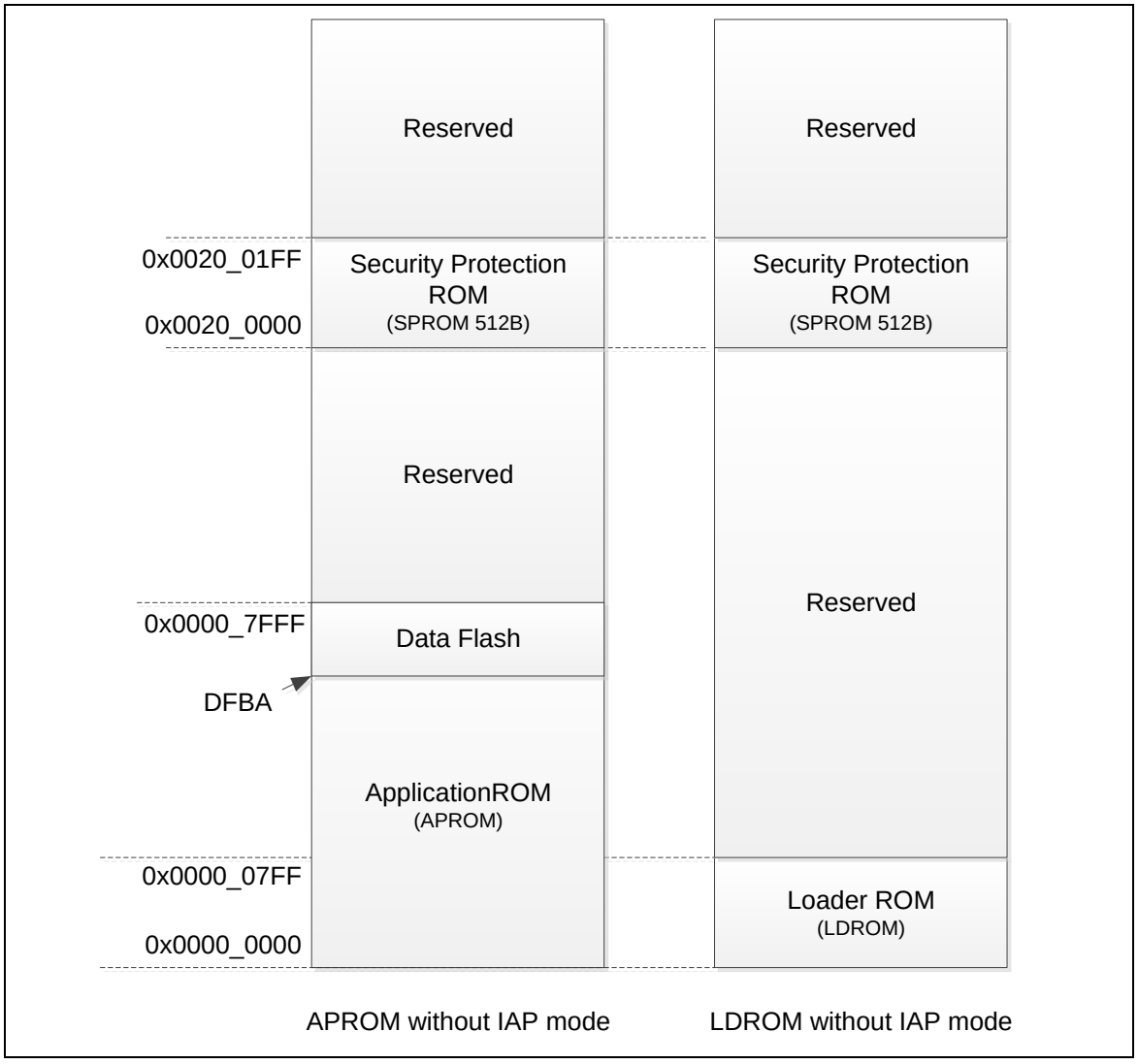


图 6.4-8 不支持IAP功能的系统内存映射

6.4.4.8 启动选择

Mini58提供了四种启动方式供用户选择，包括支持IAP功能的LDROM, 不支持IAP功能的LDROM, 支持IAP功能的APROM, 不支持IAP功能的APROM。启动源和系统内存映射由CBS (CONFIG0[7:6])控制 。

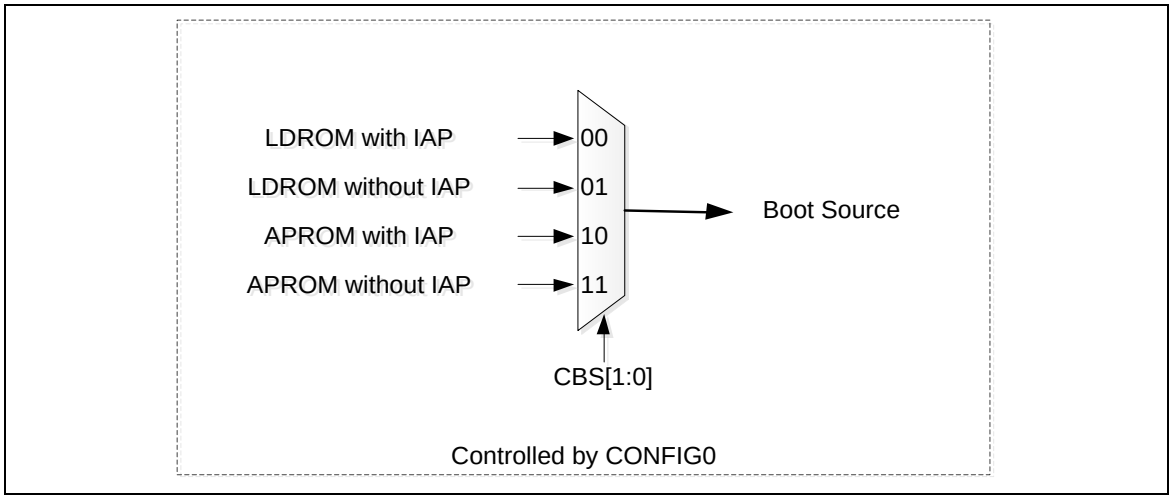


图 6.4-9 启动源选择

CBS[1:0]	启动选择/系统内存映射	向量映射支持
00	**LDROM 支持 IAP**	**是**
01	**LDROM 不支持 IAP**	**否**
10	**APROM 支持 IAP**	**是**
11	**APROM 不支持IAP**	**否**

表 6.4-1 向量映射支持表

6.4.4.9 在应用编程(IAP)

Mini58系列提供了在应用编程(IAP)功能，用户可以切换APROM, LDROM和 SPROM之间的代码。用户可以使能IAP功能，通过设定芯片的启动位寄存器CBS (CONFIG0[7:6]) 等于10 或00.

当支持IAP功能的芯片启动模式使能，任何可以执行的代码(512字节对齐)都可以随时映射到系统内存向量(0x0000_0000~0x0000_01FF)。用户可以改变重映射地址到FMC_ISPADDR，然后用”向量重映射”命令触发ISP流程。

6.4.4.10 在系统编程(ISP)

Mini58系列支持在系统编程(ISP)模式，可以通过软件控制重新烧写片上flash。使用ISP功能可以通过软件控制在目标板上直接编程，不需要将微控制器从系统板上移出，并可与多种接口相搭配，如UART, I2C和SPI.

Mini58 ISP 对片上flash操作提供如下功能：

- 支持flash页擦除功能
- 支持数据flash编程
- 支持读数据flash功能
- 支持读公司ID功能
- 支持读设备ID功能
- 支持读UID功能
- 支持内存checksum计算功能
- 支持系统内存向量重映射功能

ISP Commands

ISP Command	FMC_ISPCMD	FMC_ISPADDR	FMC_ISPDAT
FLASH页擦除	0x22	FLASH存储器的有效地址 它必须由512字节地址对齐	N/A
SPROM 页擦除	0x22	0x0020_0000	0x0055_AA03
FLASH 32-位编程	0x21	FLASH存储器的有效地址	FMC_ISPDAT :编程数据
FLASH 读	0x00	FLASH存储器的有效地址	FMC_ISPDAT: 返回数据
读公司 ID	0x0B	0x0000_0000	FMC_ISPDAT: 0x0000_00DA
读 Checksum	0x0D	保持"运行 Checksum计算"地址	FMC_ISPDAT: 返回 Checksum
运行 Checksum 计算	0x2D	有效的存储器起始地址，必须为512字节对齐	FMC_ISPDAT: 大小 必须为512字节
读 UID	0x04	0x0000_0000	FMC_ISPDAT: Unique ID 字0
		0x0000_0004	FMC_ISPDAT: Unique ID 字 1
		0x0000_0008	FMC_ISPDAT: Unique ID 字2
向量重映射	0x2E	APROM或 LDROM中的有效地址.必须为512字节对齐	N/A

表 6.4-2 ISP 命令列表

ISP流程

FMC控制器提供了片上flash内存的读，擦除和编程操作。一些FMC控制器的寄存器是写保护的，所以在设定之前要解锁。

在解锁了保护寄存器之后，用户需要设定FMC_ISPCTL控制寄存器来决定更新LDROM, APROM,SPROM或配置区，然后设定ISPEN (FMC_ISPCTL[0]) 来使能ISP功能。

一旦FMC_ISPCTL寄存器被设置成功，用户可以设定FMC_ISPCMD(参考上述ISP命令列表)来完成相应的

操作。设置FMC_ISPADDR作为flash内存的目标地址。FMC_ISPDAT可以作为设定数据来编程或作为读寄存器命令FMC_ISPCMD的返回数据。

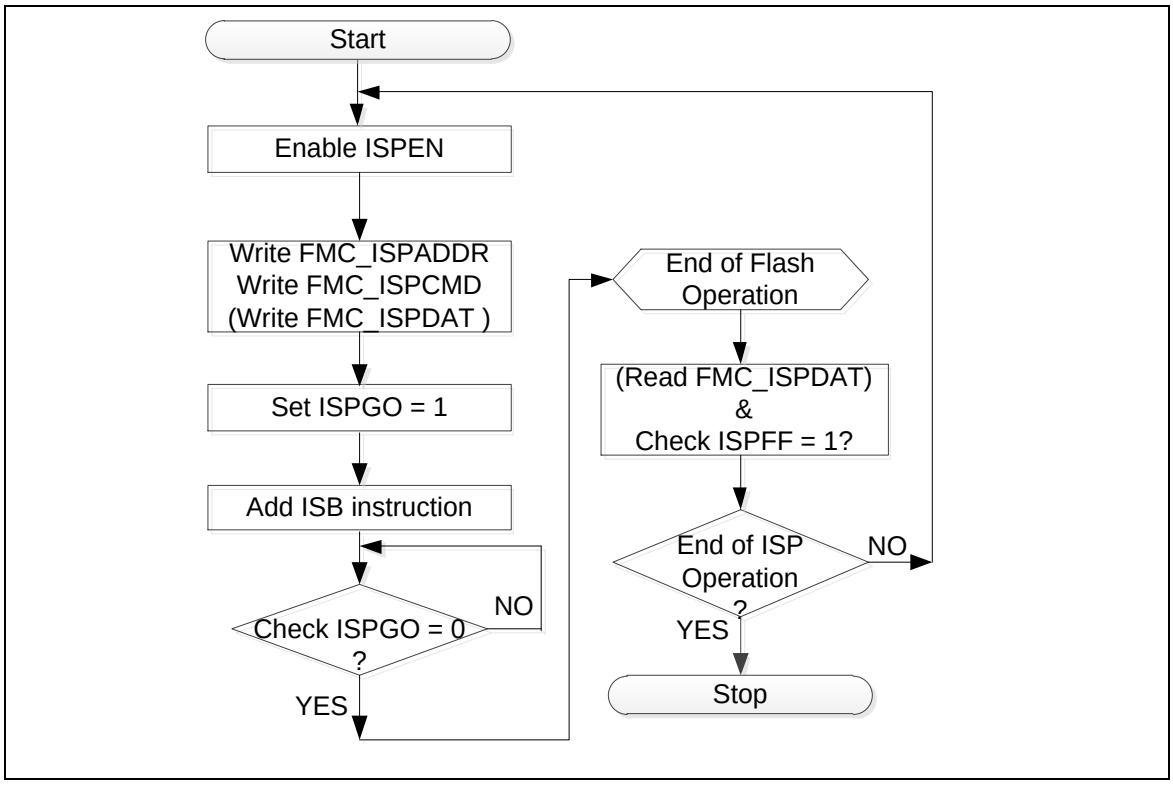


图 6.4-10 ISP流程举例

最终，设定ISPGO (FMC_ISPTRG[0])寄存器来执行相应的ISP功能。当ISP功能完成以后，ISPGO(FMC_ISPTRG[0])位自动清除。为确保ISP功能在CPU继续运行之前已经完成，在ISPGO(FMC_ISPTRG[0])设置完成后应立即使用ISB指令（指令同步隔离）。

ISP完成后，几个错误条件需要检查。如果出现错误，ISP操作就不会开始并且ISP失败标志会被置位。ISPFF(FMC_ISPSTS[6])标志只能由软件清除。就算ISPFF(FMC_ISPSTS[6])保持1，也可以开启下一个ISP流程，所以建议在ISP操作完成后检查ISPFF(FMC_ISPSTS[6])位，如果已经置1，则将其清0。

当ISPGO(FMC_ISPTRG[0])置位，CPU将一直等到ISP操作完成。这个时期内外围设备跟通常一样保持运行。任何中断请求都不会响应直到CPU完成ISP操作。当ISP操作完成ISPGO位将被硬件自动清0。用户可以通过ISPGO(FMC_ISPTRG[0])位来检查ISP操作是否完成。用户应该在ISPGO (FMC_ISPTRG[0])置1之后加ISB(指令同步隔离)指令，确保ISP操作之后的指令正确执行。

6.4.4.11 CRC32Checksum 计算公式

NuMicro® Mini58系列支持CRC-32 checksum计算功能，该功能能够帮助用户快速检查APROM, LDROM 和 SPROM内容. CRC-32 多项式如下：

$$\text{CRC-32: } X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1$$

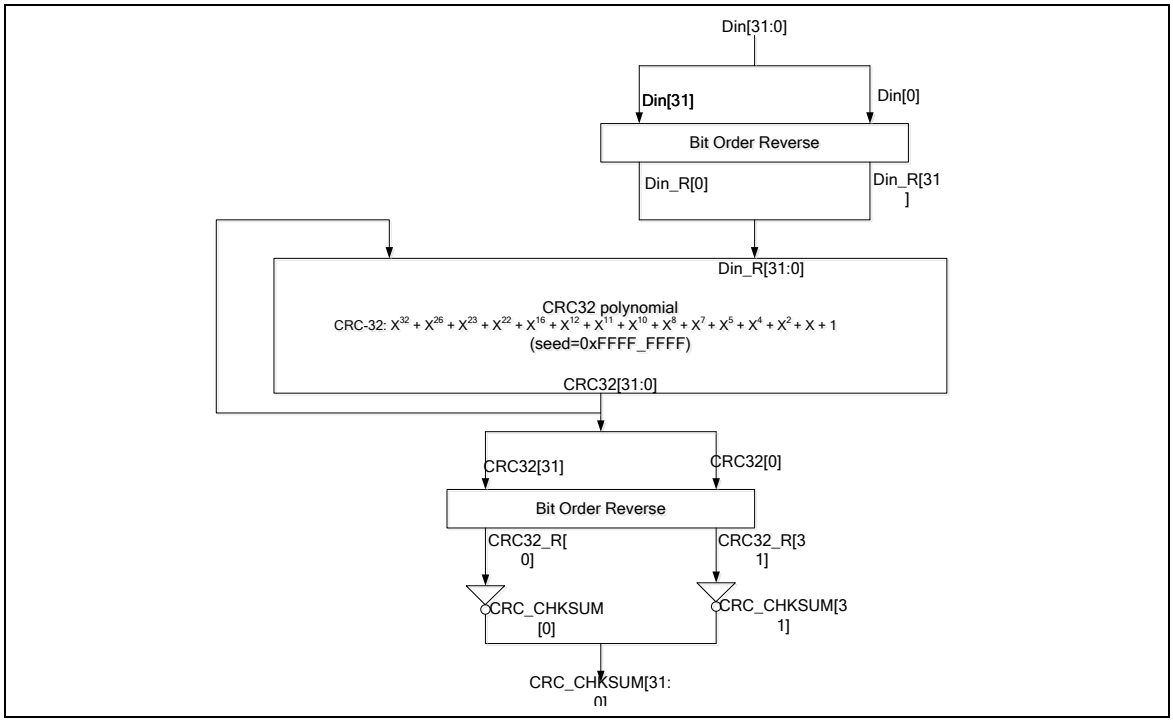


图 6.4-11 CRC-32 Checksum 计算流程

CRC-32 checksum 由以下三步完成：

1. 执行ISP存储器Checksum 操作：用户需要配置存储器的开始地址(FMC_ISPADDR)和数据大小(FMC_ISPDAT)，且地址和数据大小都必须为512字节的整数倍，开始地址可以在APROM, LDROM 和 SPROM中.
2. 执行ISP读存储器Checksum操作， FMC_ISPADDR的值应与第一步中的值保持一致。
3. 读取FMC_ISPDAT来得到 checksum: Checksum 由FMC_DAT中获得，如果checksum的值为0x0000_0000,表示下列两种情况之一 (1) Checksum正在计算中, (2)地址和大小超过设备限制。

当 SPROM 在安全模式下时， CPU和ISP不能直接从SPROM中读取内容，用户可以用Checksum功能来验证SPROM内容是否正确。

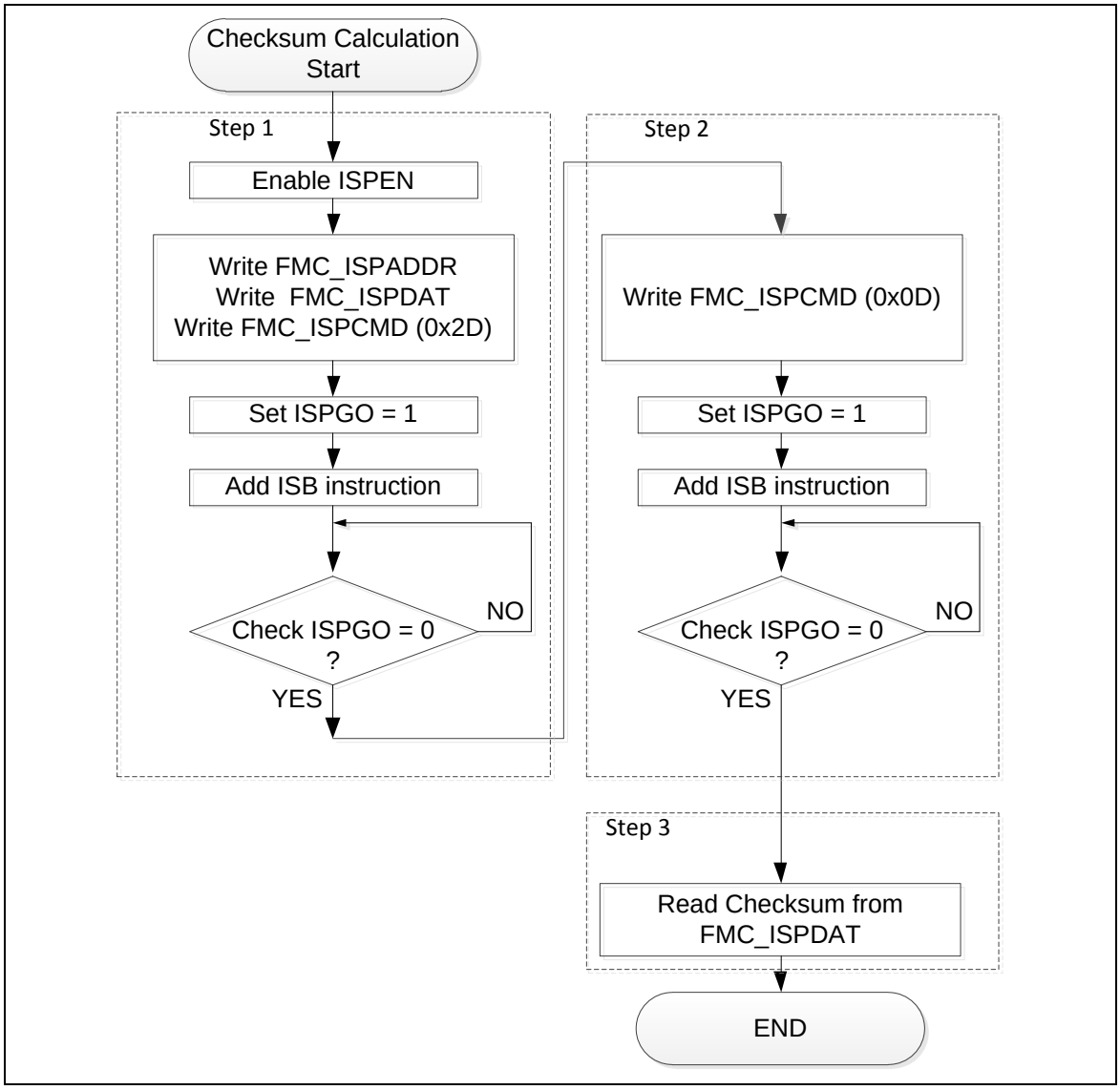


图 6.4-12 CRC-32 Checksum计算流程

6.4.5 Flash 控制寄存器

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移地址	读/写	描述	复位值
FMC基地址 **FMC_BA = 0x5000_C000**				
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP控制寄存器	0x0000_000X
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP地址寄存器	0x0000_0000
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP数据寄存器	0x0000_0000
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP命令寄存器	0x0000_0000
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP触发控制寄存器	0x0000_0000
FMC_DFBA	FMC_BA+0x14	R	数据Flash基地址	0xXXXX_XXXX
FMC_FATCTL	FMC_BA+0x18	R/W	Flash访问时间控制寄存器	0x0000_0000
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP状态寄存器	0xX0X0_000X

6.4.6 Flash 控制寄存器

ISP 控制寄存器 (FMC_ISPCTL)

寄存器	偏移地址	读/写	描述	复位值
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP 控制寄存器	0x0000_000X

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

位	描述	
[31:7]	**Reserved**	保留
[6]	**ISPFF**	**ISP失败标志 (写保护位)** 当已启动的ISP符合下面的任何一个条件时，该位由硬件置1： 该位需要通过写1清除。 (1) APUEN等于0时，APROM 写APROM. (2) LDUEN等于0时，LDROM 写LDROM. (3) CFGUEN等于0时，CONFIG 被擦除或编程. (4) SPUEN等于0时，SPU 被擦除或编程. (5)在SPROM安全模式下.编程SPROM. (6) ICE 连接状态中，LOCK模式时，进行页擦除命令 (7)欠压检测时，进行擦除或编程命令 (8)定义地址无效，比如超出正常范围 (9)无效的ISP 命令
[5]	**LDUEN**	**LDROM更新使能位(写保护位)** 0 = 禁止LDROM更新 1 = LDROM 可以更新
[4]	**CFGUEN**	**配置更新使能位(写保护位)** 0 = 禁止更新配置 1 = 使能更新配置
[3]	**APUEN**	**APROM 更新使能 位(写保护位)** 0 = 当芯片在APROM中运行时APROM 不能被更新. 1 = 当芯片在APROM中运行时APROM 可以被更新.

[2]	**SPUEN**	**SPROM 更新使能 (写保护位)** 0 = SPROM 不能被更新. 1 = SPROM 能被更新.
[1]	**BS**	**启动选择 (写保护位)** 置位/清零该位选择下次是由LDROM启动还是由APROM启动, 该位也可作为MCU启动状态的标志, 用于检查芯片是由LDROM还是APROM启动的. 除了CPU 复位(RSTS_CPU= 1)及system 复位 (RSTS_SYS)外的任何复位发生时，该位由CBS[1] (CONFIG0[7])的反转值初始化。 0 = APROM启动 1 = LDROM启动
[0]	**ISPEN**	**ISP 使能位(写保护位)** 设置该位可以使能ISP功能. 0 = 禁用 ISP 功能 1 = 使能 ISP 功能

ISP地址寄存器(FMC_ISPADDR)

寄存器	偏移地址	读/写	描述	复位值
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP 地址寄存器	0x0000_0000

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

位	描述	
[31:0]	**ISPADDR**	**ISP 地址** Mini58系列内嵌FLASH。ISP 32-位操作时，ISPADDR[1:0]必须为00.在执行向量页重映射命令时，ISPADR[8:0]必须为为0x00000000. 对于CRC32 Checksum计算命令，该地址是flashchecksum计算的起始地址，512字节对齐。

ISP数据寄存器(FMC_ISPDAT)

寄存器	偏移地址	读/写	描述	复位值
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP 数据寄存器	0x0000_0000

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

位	描述	
[31:0]	**ISPDAT**	**ISP 数据** ISP写操作前，写数据到该寄存器 ISP读操作后，从该寄存器读数据 执行CRC32 Checksum计算时，ISPDAT代表内存大小(字节)，512字节对齐。 对于ISP Checksum读命令，ISPDAT代表checksum的计算结果。 ISPDAT = 0x0000_0000，代表(1)还在checksum计算中，(2) checksum计算的内存范围出错.

ISP命令寄存器(FMC_ISPCMD)

寄存器	偏移地址	读/写	描述	复位值
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP 命令寄存器	0x0000_0000

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

位	描述	
[31:7]	**Reserved**	保留
[6:0]	**CMD**	**ISP 命令** 命令表如下: 0x00= 读 FLASH. 0x04= 读 UID. 0x0B= 读公司ID. 0x0C= 读设备 ID. 0x0D= 读CRC32 Checksum值. 0x21= FLASH 32-位编程. 0x22= FLASH 页擦除. 0x2D= 执行CRC32 Checksum 计算 0x2E=向量重映射 其他是无效指令

ISP 触发控制寄存器 (FMC_ISPTRG)

寄存器	偏移地址	读/写	描述	复位值
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP 触发控制寄存器	0x0000_0000

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

位	描述	
[31:1]	**Reserved**	保留.
[0]	**ISPGO**	**ISP触发寄存器（写保护）** 写 1 开始ISP操作，当ISP操作结束后，该位由硬件自动清零. 0 = ISP 操作结束 1 = ISP 正在执行

数据 Flash 起始地址寄存器(FMC_DFBA)

寄存器	偏移地址	读/写	描述	复位值
FMC_DFBA	FMC_BA+0x14	R	数据Flash起始地址寄存器	0xXXXX_XXXX

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

位	描述	
[31:0]	**DFBA**	**数据FLASH起始地址** 该寄存器为数据FLASH开始地址寄存器，只读。 数据flash与APROM共享存储空间。从CONFIG1加载该寄存器内容。 当DFEN (CONFIG0[0]) =0寄存器生效。

Flash 访问时间控制寄存器 (FMC_FATCTL)

寄存器	偏移地址	读/写	描述	复位值
FMC_FATCTL	FMC_BA+0x18	R/W	Flash访问时间控制寄存器	0x0000_0000

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

位	描述	
[31:9]	**Reserved**	保留
[8:7]	**Reserved**	保留
[6:4]	**FOM**	**频率优化模式（写保护位）** Mini58系列支持可调整的flash访问时间来优化不同工作频率下的flash访问周期时间。 0x1 = Frequency ≤ 24MHz. 其他 = Frequency ≤ 50MHz.
[3:0]	**Reserved**	保留.

ISP状态寄存器(FMC_ISPSTS)

寄存器	偏移地址	读/写	描述	复位值
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP 状态寄存器	0xX0X0_000X

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

位	描述	
[31]	**SCODE**	**安全码有效标志位** 在flash初始化时，如果检测到SPROM安全码有效，则该位由硬件置1。或由软件写1以确保安全码有效,该位仅可由SPROM页擦除操作清零。 0 = SPROM 完全码无效 1 = SPROM 安全码有效.
[30:21]	**Reserved**	保留.
[20:9]	**VECMAP**	**向量页映射地址（只读）** 当前 地址空间0x0000_0000~0x0000_01FF映射到地址{VECMAP[11:0], 9'h000} ~ {VECMAP[11:0], 9'h1FF}
[8:7]	**Reserved**	保留.
[6]	**ISPFF**	**ISP失败标志（写保护位）** 该位是寄存器ISPFF (FMC_ISPCTL[6])的镜像，需向FMC_ISPCTL[6] 或FMC_ISPSTS[6] 写1清零。当已启动的ISP符合下面的任何一个条件时，该位由硬件置1： (1) APUEN等于0时，APROM 写APROM. (2) LDUEN等于0时，LDROM 写LDROM. (3) CFGUEN等于0时，CONFIG 被擦除或编程. (4)SPUEN等于0时，SPROM 被擦除或编程. (5)在SPROM安全模式下编程SPROM. (6) ICE 连接状态中，LOCK模式时，进行页擦除命令 (7)欠压检测时，进行擦除或编程命令 (8)定义地址无效，比如超出正常范围 (9)无效的ISP 命令
[5:3]	**Reserved**	保留.

[2:1]	**CBS**	**启动配置（只读）** 除了CPU 复位(RSTS_CPU= 1)及system 复位 (RSTS_SYS)外的任何复位发生时，该位由CBS (CONFIG0[7:6])初始化。 00 = LDROM支持 IAP模式 01 = LDROM不支持 IAP模式 10 = APROM 支持IAP模式 11 = APROM 不支持IAP模式
[0]	**ISPBUSY**	**ISP 忙标志位（只读）** 0 = ISP 操作结束 1 = ISP 正在进行

6.5 通用I/O (GPIO)

6.5.1 概述

NuMicro® Mini58 系列有多达30个通用I/O管脚，这些管脚可以通过配置芯片和其他功能管脚共享。这30个管脚分配在P0, P1, P2, P3, P4 和 P5六个端口上。每个管脚都是独立的，都有相应的寄存器位来控制管脚的功能模式与数据。

I/O管脚的状态可由软件配置为输入，推挽输出，开漏或准双向模式。芯片复位之后，所有管脚都保持输入模式，每一个端口数据寄存器Px_DOUT[n]都被设置成1。在准双向模式下，每一个I/O管脚都有一个阻值为110K~300K的弱上拉电阻接到VDD 上，VDD范围为从5.0 V 到2.5 V。

6.5.2 特性

- 四种 I/O 模式:
  - 准双向模式
  - 推挽输出
  - 开漏输出
  - 高阻态输入
- TTL/Schmitt 触发输入模式，由SYS_Px_MFP[23:16]选择
- I/O可以配置成边沿/电平触发的中断源
- 仅在准双向模式下，I/O管脚内部上拉电阻才使能.
- 使能管脚中断功能将同时使能GPIO唤醒功能
- 支持高驱动力及高漏电流的I/O模式。
- 通过CIOINI (CONFIG0[10])复位，可配置所有I/O的默认模式。
  - 如果CIOIN = 0, 复位后所有的GPIO管脚是准双向模式
  - 如果CIOIN = 1, 复位后所有的GPIO管脚是三态输入模式

6.5.3 框图

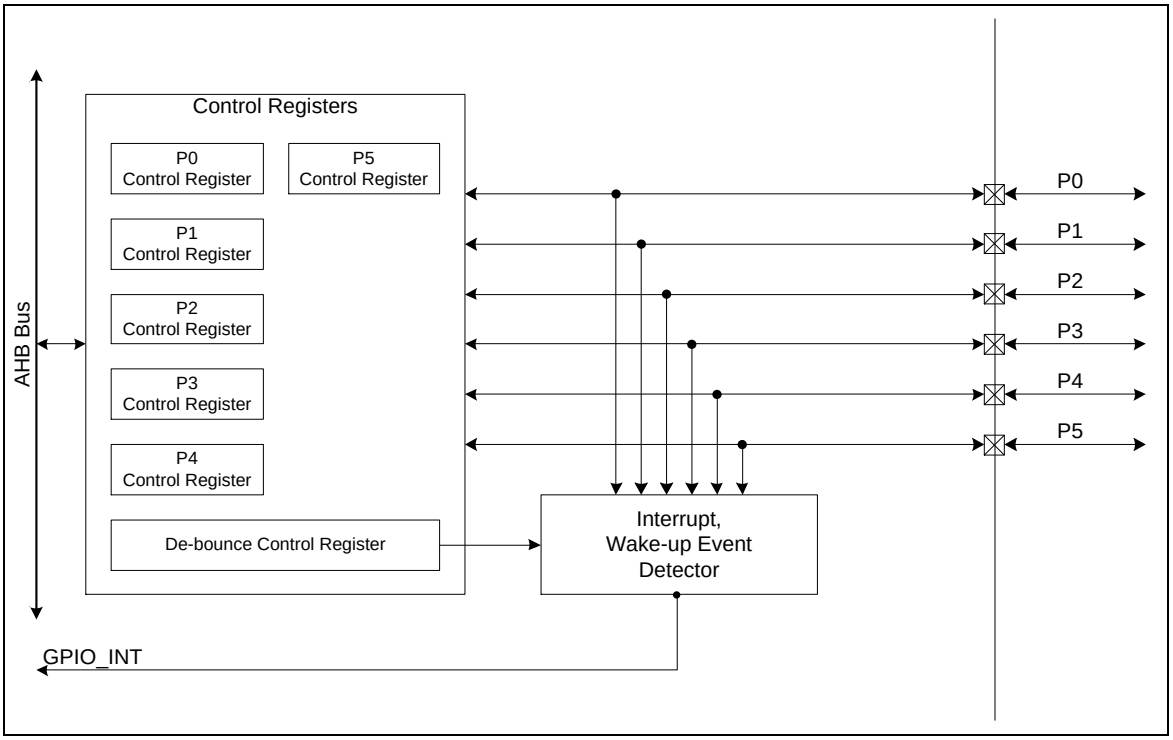


图 6.5-1 GPIO 控制器框图

6.5.4 基本配置

GPIO管脚通过 SYS_P0_MFP, SYS_P1_MFP, SYS_P2_MFP, SYS_P3_MFP, SYS_P4_MFP和 SYS_P5_MFP 寄存器进行配置。

6.5.5 功能描述

6.5.5.1 输入模式

设置MODEn (Px_MODE[2n+1:2n]) 为00, Px.n管脚为输入模式，I/O管脚为三态（高阻），没有输出驱动能力。管脚 (Px_PIN[n])的值反映相应端口的状态。

6.5.5.2 推挽输出模式

设置 MODEn (Px_MODE[2n+1:2n]) 为01，Px.n 管脚为推挽输出模式，I/O支持数字输出功能，有拉/灌电流能力。DOUT (Px_DOUT[n])相应位bit[n]的值被送到相应管脚上。

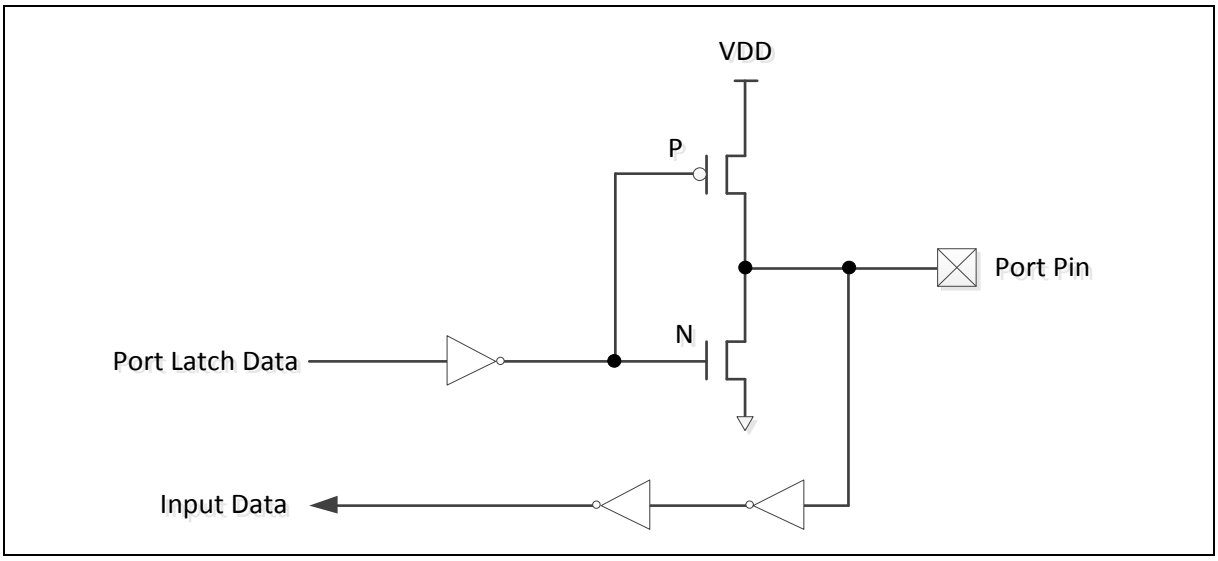


图 6.5-2 推挽输出

6.5.5.3 开漏输出模式

设置MODEn (Px_MODE[2n+1:2n]) 为10,Px.n 管脚为开漏模式，I/O管脚数字输出功能仅支持灌电流, 驱动高电平时需要外加上拉电阻。如果DOUT (Px_DOUT[n]) 相应位为‘0’, 管脚上输出低电平. 如果DOUT (Px_DOUT[n]) 相应位为‘1’, 该管脚输出为高电平时，由外部上拉电阻控制。

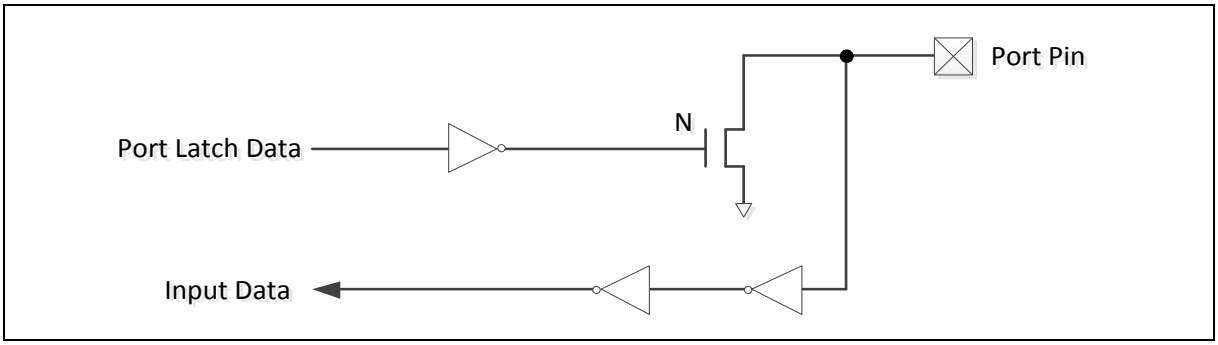


图 6.5-3 开漏输出

6.5.5.4 准双向模式

设置MODEn (Px_MODE[2n+1:2n])为11，Px.n 管脚为准双向模式，I/O同时支持数字输出和输入功能，但拉电流能力仅达数百uA。要实现数字输入，需要先将DOUT (Px_DOUT[n])相应位置1。准双向输出是80C51 及其派生产品常见的模式。若DOUT (Px_DOUT[n])相应位bit[n]为'0' ，管脚上输出为"低"。若DOUT (Px_DOUT[n])相应位bit[n]为'1'，该管脚将检测管脚值。若管脚值为高，没有任何动作，若管脚值为低，在该管脚上将有强输出驱动长度为2个时钟周期的高电平，然后关闭强输出驱动，其后管脚状态由内部上拉电阻控制。注意: 准双向模式的源电流大小仅有200 uA到30 uA(相应VDD的电压从5.0 V到2.5 V) 。

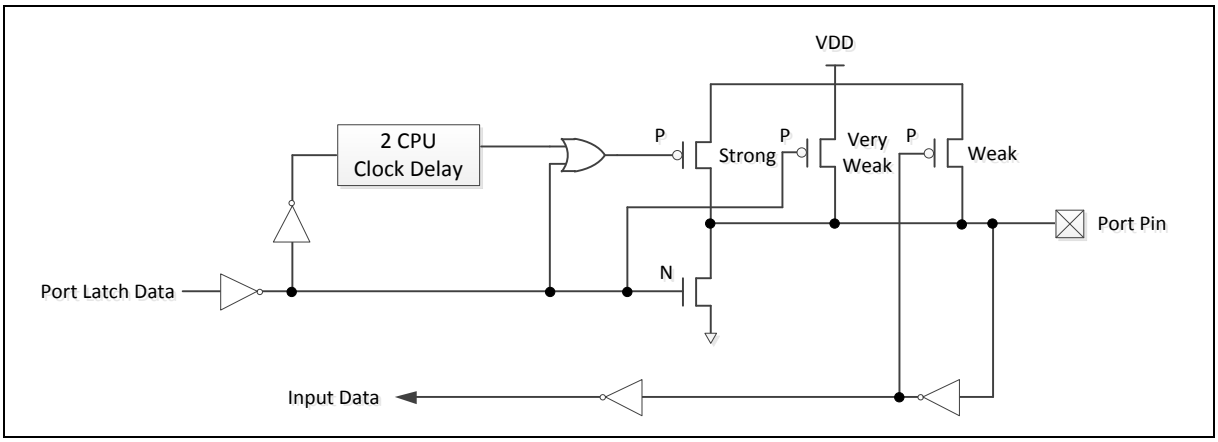


图 6.5-4 准双向I/O模式

6.5.6 GPIO 中断和唤醒功能

每个GPIO管脚都可以通过RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n])位和TYPE (Px_INTTYPE[n])设置成芯片的中断源。有五种中断触发条件可以设置：低电平触发、高电平触发、下降沿触发和上升沿触发以及上升与下降沿同时触发。在边沿触发中用户可以通过使能输入信号去抖功能来阻止由噪声引起的意外中断。去抖时钟源和采样周期可以通过DBCLKSRC (GPIO_DBCTL[4]) 和DBCLKSEL (GPIO_DBCTL[3:0])寄存器来设置。

当芯片进入Idle/Power-down模式时，GPIO也可以唤醒系统。设置GPIO为唤醒触发的条件与GPIO中断触发的条件相同。

6.5.7 寄存器映射

R: 只读, **W:** 只写, **R/W:** 读写

寄存器	偏移地址	R/W	描述	复位值
GPIO 基地址: **GPIO_BA = 0x5000_4000**				
P0_MODE	GP_BA+0x000	R/W	P0 I/O 模式控制	0x0000_XXXX
P0_DINOFF	GP_BA+0x004	R/W	P0关闭数字输入通路	0x0000_0000
P0_DOUT	GP_BA+0x008	R/W	P0数据输出	0x0000_00F3
P0_DATMSK	GP_BA+0x00C	R/W	P0数据输出写屏蔽	0x0000_0000
P0_PIN	GP_BA+0x010	R	P0管脚数值	0x0000_00XX
P0_DBEN	GP_BA+0x014	R/W	P0去抖动使能控制	0x0000_0000
P0_INTTYPE	GP_BA+0x018	R/W	P0中断模式控制	0x0000_0000
P0_INTEN	GP_BA+0x01C	R/W	P0中断使能控制	0x0000_0000
P0_INTSRC	GP_BA+0x020	R/W	P0中断源标志	0x0000_0000
P1_MODE	GP_BA+0x040	R/W	P1 I/O 模式控制	0x0000_XXXX
P1_DINOFF	GP_BA+0x044	R/W	P1关闭数字输入通路	0x0000_0000
P1_DOUT	GP_BA+0x048	R/W	P1数据输出寄存器	0x0000_003D
P1_DATMSK	GP_BA+0x04C	R/W	P1数据输出写屏蔽	0x0000_0000
P1_PIN	GP_BA+0x050	R	P1管脚数值	0x0000_00XX
P1_DBEN	GP_BA+0x054	R/W	P1去抖动使能控制	0x0000_0000
P1_INTTYPE	GP_BA+0x058	R/W	P1中断模式控制	0x0000_0000
P1_INTEN	GP_BA+0x05C	R/W	P1中断使能控制	0x0000_0000
P1_INTSRC	GP_BA+0x060	R/W	P1中断源标志	0x0000_0000
P2_MODE	GP_BA+0x080	R/W	P2 I/O 模式控制	0x0000_XXXX
P2_DINOFF	GP_BA+0x084	R/W	P2关闭数字输入通路	0x0000_0000
P2_DOUT	GP_BA+0x088	R/W	P2数据输出寄存器	0x0000_007C
P2_DATMSK	GP_BA+0x08C	R/W	P2数据输出写屏蔽	0x0000_0000
P2_PIN	GP_BA+0x090	R	P2管脚数值	0x0000_00XX
P2_DBEN	GP_BA+0x094	R/W	P2去抖动使能控制	0x0000_0000
P2_INTTYPE	GP_BA+0x098	R/W	P2中断模式控制	0x0000_0000
P2_INTEN	GP_BA+0x09C	R/W	P2中断使能控制	0x0000_0000

寄存器	偏移地址	R/W	描述	复位值
GPIO 基地址: **GPIO_BA = 0x5000_4000**				
P2_INTSRC	GP_BA+0x0A0	R/W	P2中断源标志	0x0000_0000
P3_MODE	GP_BA+0x0C0	R/W	P3 I/O 模式控制	0x0000_XXXX
P3_DINOFF	GP_BA+0x0C4	R/W	P3关闭数字输入通路	0x0000_0000
P3_DOUT	GP_BA+0x0C8	R/W	P3数据输出	0x0000_0077
P3_DATMSK	GP_BA+0x0CC	R/W	P3数据输出写屏蔽	0x0000_0000
P3_PIN	GP_BA+0x0D0	R	P3管脚数值	0x0000_00XX
P3_DBEN	GP_BA+0x0D4	R/W	P3去抖动使能控制	0x0000_0000
P3_INTTYPE	GP_BA+0x0D8	R/W	P3中断模式控制	0x0000_0000
P3_INTEN	GP_BA+0x0DC	R/W	P3中断使能控制	0x0000_0000
P3_INTSRC	GP_BA+0x0E0	R/W	P3中断源标志	0x0000_0000
P4_MODE	GP_BA+0x100	R/W	P4 I/O 模式控制	0x0000_XXXX
P4_DINOFF	GP_BA+0x104	R/W	P4关闭数字输入通路	0x0000_0000
P4_DOUT	GP_BA+0x108	R/W	P4数据输出	0x0000_00C0
P4_DATMSK	GP_BA+0x10C	R/W	P4数据输出写屏蔽	0x0000_0000
P4_PIN	GP_BA+0x110	R	P4管脚数值	0x0000_00XX
P4_DBEN	GP_BA+0x114	R/W	P4去抖动使能控制	0x0000_0000
P4_INTTYPE	GP_BA+0x118	R/W	P4中断模式控制	0x0000_0000
P4_INTEN	GP_BA+0x11C	R/W	P4中断使能控制	0x0000_0000
P4_INTSRC	GP_BA+0x120	R/W	P4中断源标志	0x0000_0000
P5_MODE	GP_BA+0x140	R/W	P5 I/O 模式控制	0x0000_XXXX
P5_DINOFF	GP_BA+0x144	R/W	P5关闭数字输入通路	0x0000_0000
P5_DOUT	GP_BA+0x148	R/W	P5数据输出	0x0000_003F
P5_DATMSK	GP_BA+0x14C	R/W	P5数据输出写屏蔽	0x0000_0000
P5_PIN	GP_BA+0x150	R	P5管脚数值	0x0000_00XX
P5_DBEN	GP_BA+0x154	R/W	P5去抖动使能控制	0x0000_0000
P5_INTTYPE	GP_BA+0x158	R/W	P5中断模式控制	0x0000_0000
P5_INTEN	GP_BA+0x15C	R/W	P5中断使能控制	0x0000_0000
P5_INTSRC	GP_BA+0x160	R/W	P5中断源标志	0x0000_0000
GPIO_DBCTL	GP_BA+0x180	R/W	去抖周期控制	0x0000_0020

寄存器	偏移地址	R/W	描述	复位值
GPIO 基地址: **GPIO_BA = 0x5000_4000**				
P00_PDIO	GP_BA+0x200	R/W	GPIO P0.0.端口数据输入/输出	0x0000_0001
P01_PDIO	GP_BA+0x204	R/W	GPIO P0.1.端口数据输入/输出	0x0000_0001
P04_PDIO	GP_BA+0x210	R/W	GPIO P0.4.端口数据输入/输出	0x0000_0001
P05_PDIO	GP_BA+0x214	R/W	GPIO P0.5.端口数据输入/输出	0x0000_0001
P06_PDIO	GP_BA+0x218	R/W	GPIO P0.6.端口数据输入/输出	0x0000_0001
P07_PDIO	GP_BA+0x21C	R/W	GPIO P0.7.端口数据输入/输出	0x0000_0001
P10_PDIO	GP_BA+0x220	R/W	GPIO P1.0.端口数据输入/输出	0x0000_0001
P12_PDIO	GP_BA+0x228	R/W	GPIO P1.2.端口数据输入/输出	0x0000_0001
P13_PDIO	GP_BA+0x22C	R/W	GPIO P1.3.端口数据输入/输出	0x0000_0001
P14_PDIO	GP_BA+0x230	R/W	GPIO P1.4.端口数据输入/输出	0x0000_0001
P15_PDIO	GP_BA+0x234	R/W	GPIO P1.5.端口数据输入/输出	0x0000_0001
P22_PDIO	GP_BA+0x248	R/W	GPIO P2.2.端口数据输入/输出	0x0000_0001
P23_PDIO	GP_BA+0x24C	R/W	GPIO P2.3.端口数据输入/输出	0x0000_0001
P24_PDIO	GP_BA+0x250	R/W	GPIO P2.4.端口数据输入/输出	0x0000_0001
P25_PDIO	GP_BA+0x254	R/W	GPIO P2.5.端口数据输入/输出	0x0000_0001
P26_PDIO	GP_BA+0x258	R/W	GPIO P2.6.端口数据输入/输出	0x0000_0001
P30_PDIO	GP_BA+0x260	R/W	GPIO P3.0.端口数据输入/输出	0x0000_0001
P31_PDIO	GP_BA+0x264	R/W	GPIO P3.1.端口数据输入/输出	0x0000_0001
P32_PDIO	GP_BA+0x268	R/W	GPIO P3.2.端口数据输入/输出	0x0000_0001
P34_PDIO	GP_BA+0x270	R/W	GPIO P3.4.端口数据输入/输出	0x0000_0001
P35_PDIO	GP_BA+0x274	R/W	GPIO P3.5.端口数据输入/输出	0x0000_0001
P36_PDIO	GP_BA+0x278	R/W	GPIO P3.6.端口数据输入/输出	0x0000_0001
P46_PDIO	GP_BA+0x298	R/W	GPIO P4.6.端口数据输入/输出	0x0000_0001
P47_PDIO	GP_BA+0x29C	R/W	GPIO P4.7.端口数据输入/输出	0x0000_0001
P50_PDIO	GP_BA+0x2A0	R/W	GPIO P5.0.端口数据输入/输出	0x0000_0001
P51_PDIO	GP_BA+0x2A4	R/W	GPIO P5.1.端口数据输入/输出	0x0000_0001
P52_PDIO	GP_BA+0x2A8	R/W	GPIO P5.2.端口数据输入/输出	0x0000_0001
P53_PDIO	GP_BA+0x2AC	R/W	GPIO P5.3.端口数据输入/输出	0x0000_0001

寄存器	偏移地址	R/W	描述	复位值
GPIO 基地址: **GPIO_BA = 0x5000_4000**				
P54_PDIO	GP_BA+0x2B0	R/W	GPIO P5.4.端口数据输入/输出	0x0000_0001
P55_PDIO	GP_BA+0x2B4	R/W	GPIO P5.5.端口数据输入/输出	0x0000_0001

注意:在使用QFN-33封装时，P5.5脚没有邦定，为了降低掉电模式功耗，须通过软件将管脚P5.5设置成输出模式。而在TSSOP20 时不需此操作。

6.5.8 寄存器 描述

端口 0-5 I/O 模式控制 (Px_MODE)

寄存器	偏移地址	读/写	描述	复位值
P0_MODE	GP_BA+0x000	R/W	P0 I/O 模式控制	0x0000_XXXX
P1_MODE	GP_BA+0x040	R/W	P1 I/O 模式控制	0x0000_XXXX
P2_MODE	GP_BA+0x080	R/W	P2 I/O 模式控制	0x0000_XXXX
P3_MODE	GP_BA+0x0C0	R/W	P3 I/O 模式控制	0x0000_XXXX
P4_MODE	GP_BA+0x100	R/W	P4 I/O 模式控制	0x0000_XXXX
P5_MODE	GP_BA+0x140	R/W	P5 I/O 模式控制	0x0000_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MODE7		**MODE6**		**MODE5**		**MODE4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MODE3		**MODE2**		**MODE1**		**MODE0**	

位	描述	
[31:16]	**Reserved**	保留.
[2n+1:2n] n=0,1..7	**MODEn**	**端口0-5 I/O Pin[N] 模式控制** 决定GPIOPx.n的I/O 类型. 00 = Px.n管脚为输入模式 01 = Px.n管脚为推挽输出模式 10 = Px.n 管脚为开漏模式 11 = Px.n管脚为准双向模式 **注意1**: 初始值由CIOINI (CONFIG0 [10])决定， CIOINI置0时默认值是0x0000_FFFF，上电后所有端口是准双向模式。 CIOINI置1时默认值是0x0000_0000，上电后所有端口是输入模式。 **注意2:** 端口0：Max. n=7, n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留

		端口5：Max. n=7, n=6,n=7保留

端口 0-5 管脚关闭数字通路寄存器 (Px_DINOFF)

寄存器	偏移地址	读/写	描述	复位值
P0_DINOFF	GP_BA+0x004	R/W	P0 关闭数字通路使能	0x0000_0000
P1_DINOFF	GP_BA+0x044	R/W	P1 关闭数字通路使能	0x0000_0000
P2_DINOFF	GP_BA+0x084	R/W	P2 关闭数字通路使能	0x0000_0000
P3_DINOFF	GP_BA+0x0C4	R/W	P3 关闭数字通路使能	0x0000_0000
P4_DINOFF	GP_BA+0x104	R/W	P4 关闭数字通路使能	0x0000_0000
P5_DINOFF	GP_BA+0x144	R/W	P5 关闭数字通路使能	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留.
[n+16] n=0,1..7	**DINOFF[n]**	**端口0-5 Pin[N] 关闭数字输入通道使能** 用于控制Px.n的数字输入通路是否使能. 如果输入为模拟信号，用户可以关闭Px.n输入通道,防止漏电. 0 = 使能IO数据输入通道 1 = 关闭IO的数字输入通道(数字输入拉低) **注意**: 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留
[15:0]	**Reserved**	保留.

端口 0-5 数据输出值 (Px_DOUT)

寄存器	偏移地址	读/写	描述	复位值
P0_DOUT	GP_BA+0x008	R/W	P0 数据输出值	0x0000_00F3
P1_DOUT	GP_BA+0x048	R/W	P1 数据输出值	0x0000_003D
P2_DOUT	GP_BA+0x088	R/W	P2 数据输出值	0x0000_007C
P3_DOUT	GP_BA+0x0C8	R/W	P3 数据输出值	0x0000_0077
P4_DOUT	GP_BA+0x108	R/W	P4 数据输出值	0x0000_00C0
P5_DOUT	GP_BA+0x148	R/W	P5 数据输出值	0x0000_003F

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

位	描述	
[31:8]	**Reserved**	保留.
[n] n=0,1..7	**DOUT[n]**	**端口0-5 Pin[N] 输出值** 在Px.n配置成推挽输出，开漏和准双向模式时，控制Px.n相应管脚的状态. 0 = GPIO配置成推挽输出，开漏和准双向模式时，Px.n为低 1 = GPIO配置成推挽输出，开漏和准双向模式时，Px.n为高 **注意：** 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留

端口 0-5 数据输出写屏蔽 (Px_DATMSK)

寄存器	偏移地址	读/写	描述	复位值
P0_DATMSK	GP_BA+0x00C	R/W	P0 数据输出写屏蔽	0x0000_0000
P1_DATMSK	GP_BA+0x04C	R/W	P1 数据输出写屏蔽	0x0000_0000
P2_DATMSK	GP_BA+0x08C	R/W	P2 数据输出写屏蔽	0x0000_0000
P3_DATMSK	GP_BA+0x0CC	R/W	P3 数据输出写屏蔽	0x0000_0000
P4_DATMSK	GP_BA+0x10C	R/W	P4 数据输出写屏蔽	0x0000_0000
P5_DATMSK	GP_BA+0x14C	R/W	P5 数据输出写屏蔽	0x0000_0000

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

位	描述	
[31:8]	**Reserved**	保留.
[n] n=0,1..7	**DATMSK[n]**	**端口0-5 Pin[N] 数据输出写屏蔽** 用于保护相应寄存器DOUT (Px_DOUT[n]). 当设置DATMSK (Px_DATMSK[n])为 '1'时，相应DOUT (Px_DOUT[n])位被保护，写信号被屏蔽时，不能向保护位写数据。 0 =相应的DOUT (Px_DOUT[n])位可以被更新 1 =保护相应的DOUT (Px_DOUT[n])位 **注意1**：该功能只保护相应的DOUT (Px_DOUT[n])位，不保护(PDIO (Pxn_PDIO[0]))相应位。 **注意2:** 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留

端口 0-5 管脚数据 (Px_PIN)

寄存器	偏移地址	读/写	描述	复位值
P0_PIN	GP_BA+0x010	R	P0 管脚数据	0x0000_00XX
P1_PIN	GP_BA+0x050	R	P1 管脚数据	0x0000_00XX
P2_PIN	GP_BA+0x090	R	P2 管脚数据	0x0000_00XX
P3_PIN	GP_BA+0x0D0	R	P3 管脚数据	0x0000_00XX
P4_PIN	GP_BA+0x110	R	P4 管脚数据	0x0000_00XX
P5_PIN	GP_BA+0x150	R	P5 管脚数据	0x0000_00XX

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

位	描述	
[31:8]	**Reserved**	保留.
[n] n=0,1..7	**PIN[n]**	**端口0-5 Pin[N] 管脚数据** 这些位的值为各个Px.n管脚真实状态的反映。如果值为'1'，表示相应管脚状态为高，否则为低 **注意：** 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留

端口 0-5 去抖动使能寄存器 (Px_DBEN)

寄存器	偏移地址	读/写	描述	复位值
P0_DBEN	GP_BA+0x014	R/W	P0 去抖动使能控制	0x0000_0000
P1_DBEN	GP_BA+0x054	R/W	P1 去抖动使能控制	0x0000_0000
P2_DBEN	GP_BA+0x094	R/W	P2 去抖动使能控制	0x0000_0000
P3_DBEN	GP_BA+0x0D4	R/W	P3 去抖动使能控制	0x0000_0000
P4_DBEN	GP_BA+0x114	R/W	P4 去抖动使能控制	0x0000_0000
P5_DBEN	GP_BA+0x154	R/W	P5 去抖动使能控制	0x0000_0000

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

位	描述	
[31:8]	**Reserved**	保留.
[n] n=0,1..7	**DBEN[n]**	**端口0-5 Pin[N] 输入信号去抖动使能位** DBEN[n]用于使能相应位的去抖动功能. 如果输入信号脉冲宽度不能被两个连续的去抖动采样周期所采样，则被视为信号抖动，从而不触发中断. 去抖动时钟源由DBCLKSRC (GPIO_DBCTL [4])控制, 一个去抖动采样周期由DBCLKSEL (GPIO_DBCTL [3:0])控制。 0 = 禁用Px.n去抖动功能 1 = 使能Px.n去抖动功能 去抖动功能仅对边沿触发中断有效, 对于电平触发中断模式，去抖动功能使能位不起作用. **注意1:**如果Px.n管脚选作掉电唤醒接口，用户在进入掉电模式之前应该禁用去抖功能，以避免系统唤醒之后第二个中断事件出现，该事件由Px.n的去抖功能引起。 **注意2:** 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留

端口 0-5 中断模式控制 (Px_ INTTYPE)

寄存器	偏移地址	读/写	描述	复位值
P0_INTTYPE	GP_BA+0x018	R/W	P0 中断模式控制	0x0000_0000
P1_INTTYPE	GP_BA+0x058	R/W	P1 中断模式控制	0x0000_0000
P2_INTTYPE	GP_BA+0x098	R/W	P2 中断模式控制	0x0000_0000
P3_INTTYPE	GP_BA+0x0D8	R/W	P3 中断模式控制	0x0000_0000
P4_INTTYPE	GP_BA+0x118	R/W	P4 中断模式控制	0x0000_0000
P5_INTTYPE	GP_BA+0x158	R/W	P5 中断模式控制	0x0000_0000

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

位	描述	
[31:8]	**Reserved**	保留.
[n] n=0,1..7	**TYPE[n]**	**端口0-5 Pin[N] 边沿或电平检测中断控制** TYPE (Px_INTTYPE[n])位用于控制电平触发或边沿触发中断. 若为边沿触发中断，触发源可由去抖动控制，如果是电平触发中断，输入源由一个HCLK时钟采样并产生中断。 0 = 边沿触发中断 1 = 电平触发中断 如果设置管脚为电平触发模式，则在寄存器RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n])中，只能设置一个电平(高电平或者低电平)；若设置了两个电平都触发中断，则设置被忽略，不会产生中断 去抖动功能对于边沿触发中断有效，对于电平触发中断无效 **注意:** 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留

端口 0-5中断使能控制(Px_INTEN)

寄存器	偏移地址	读/写	描述	复位值
P0_INTEN	GP_BA+0x01C	R/W	P0 中断使能控制	0x0000_0000
P1_INTEN	GP_BA+0x05C	R/W	P1 中断使能控制	0x0000_0000
P2_INTEN	GP_BA+0x09C	R/W	P2 中断使能控制	0x0000_0000
P3_INTEN	GP_BA+0x0DC	R/W	P3 中断使能控制	0x0000_0000
P4_INTEN	GP_BA+0x11C	R/W	P4 中断使能控制	0x0000_0000
P5_INTEN	GP_BA+0x15C	R/W	P5 中断使能控制	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留
[n+16] n=0,1..7	**RHIEN[n]**	**端口0-5 Pin[N] 输入上升沿或输入高电平中断使能位** RHIEN (Px_INTEN[n+16])用于使能相应GPIOPx.n输入的中断. 置‘1’也可以使能管脚唤醒功能 当设置RHIEN (Px_INTEN[n+16])位为‘1’: 如果中断是电平触发模式(TYPE (Px_INTTYPE[n])=1),，输入Px.n的状态为高电平时，产生中断. 如果中断是边沿触发模式(TYPE (Px_INTTYPE[n])=0),，输入Px.n的状态由低电平到高电平变化时，产生中断. 0 = 禁用Px.n高电平或由低电平到高电平变化的中断 1 = 使能Px.n高电平或由低电平到高电平变化的中断 **注意:** 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留
[15:8]	**Reserved**	保留.

[n] n=0,1..7	**FLIEN[n]**	**端口0-5 Pin[N] 输入下降沿或输入低电平中断使能位** FLIEN (Px_INTEN[n])用于使能相应GPIO Px.n]输入的中断. 置 '1' 也可以使能管脚唤醒功能 当设置FLIEN (Px_INTEN[n])位为 '1': 如果中断是电平触发模式(TYPE (Px_INTTYPE[n])=1),，输入Px.n的状态为低电平时，产生中断. 如果中断是边沿触发模式(TYPE (Px_INTTYPE[n])= 0),，输入Px.n的状态由高电平到低电平变化时，产生中断. 0 = 禁用Px.n低电平或由高电平到低电平变化的中断 1 = 使能Px.n低电平或由高电平到低电平变化的中断 **注意:** 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留

端口 0-5 中断源标志 (Px_INTSRC)

寄存器	偏移地址	读/写	描述	复位值
P0_INTSRC	GP_BA+0x020	R/W	P0中断源标志	0x0000_0000
P1_INTSRC	GP_BA+0x060	R/W	P1中断源标志	0x0000_0000
P2_INTSRC	GP_BA+0x0A0	R/W	P2中断源标志	0x0000_0000
P3_INTSRC	GP_BA+0x0E0	R/W	P3中断源标志	0x0000_0000
P4_INTSRC	GP_BA+0x120	R/W	P4中断源标志	0x0000_0000
P5_INTSRC	GP_BA+0x160	R/W	P5中断源标志	0x0000_0000

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

位	描述	
[31:8]	**Reserved**	保留.
[n] n=0,1..15	**INTSRC[n]**	**端口0-5 Pin[N]中断触发源标志** 写： 0= 无动作 1= 清相应的未处理中断标志 读： 0 = Px.n没有中断 1 = Px.n产生中断 **注意:** 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留

中断去抖周期控制 (GPIO_DBCTL)

寄存器	偏移地址	读/写	描述	复位值
GPIO_DBCTL	GP_BA+0x180	R/W	去抖周期控制	0x0000_0020

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

位	描述	
[31:6]	**Reserved**	保留.
[5]	**ICLKON**	**中断时钟 On 模式** 0 = 边沿侦测电路仅在 I/O管脚对应的RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n])位置 1 有效. 1 = 复位之后所有 I/O边沿侦测电路使能. **注意：**如果没有相关的特定应用，建议关掉该时钟以减少耗电
[4]	**DBCLKSRC**	**去抖动计数器时钟源选择** 0 = 去抖动计数器时钟源为 HCLK 1 = 去抖动计数器时钟源为内部 10 KHz(LIRC)时钟

[3:0]	**DBCLKSEL**	去抖动采样周期选择 0000 = 采样中断信号，每1 clocks一次 0001 = 采样中断信号，每2 clocks一次 0010 = 采样中断信号，每4 clocks一次 0011 = 采样中断信号，每8 clocks一次 0100 = 采样中断信号，每16 clocks一次 0101 = 采样中断信号，每32 clocks一次 0110 = 采样中断信号，每64 clocks一次 0111 = 采样中断信号，每128 clocks一次 1000 = 采样中断信号，每256 clocks一次 1001 = 采样中断信号，每2*256 clocks一次. 1010 = 采样中断信号，每4*256 clocks一次. 1011 = 采样中断信号，每8*256 clocks一次 1100 = 采样中断信号，每16*256 clocks一次 1101 = 采样中断信号，每32*256 clocks一次 1110 = 采样中断信号，每64*256 clocks一次 1111 = 采样中断信号，每128*256 clocks一次

GPIO Px.n数据输入输出 (Pxn_PDIO)

寄存器	偏移地址	读/写	描述	复位值
P00_PDIO	GP_BA+0x200	R/W	GPIO P0.0 管脚数据输入/输出	0x0000_0001
P01_PDIO	GP_BA+0x204	R/W	GPIO P0.1管脚数据输入/输出	0x0000_0001
P04_PDIO	GP_BA+0x210	R/W	GPIO P0.4管脚数据输入/输出	0x0000_0001
P05_PDIO	GP_BA+0x214	R/W	GPIO P0.5管脚数据输入/输出	0x0000_0001
P06_PDIO	GP_BA+0x218	R/W	GPIO P0.6管脚数据输入/输出	0x0000_0001
P07_PDIO	GP_BA+0x21C	R/W	GPIO P0.7管脚数据输入/输出	0x0000_0001
P10_PDIO	GP_BA+0x220	R/W	GPIO P1.0管脚数据输入/输出	0x0000_0001
P12_PDIO	GP_BA+0x228	R/W	GPIO P1.2管脚数据输入/输出	0x0000_0001
P13_PDIO	GP_BA+0x22C	R/W	GPIO P1.3管脚数据输入/输出	0x0000_0001
P14_PDIO	GP_BA+0x230	R/W	GPIO P1.4管脚数据输入/输出	0x0000_0001
P15_PDIO	GP_BA+0x234	R/W	GPIO P1.5管脚数据输入/输出	0x0000_0001
P22_PDIO	GP_BA+0x248	R/W	GPIO P2.2管脚数据输入/输出	0x0000_0001
P23_PDIO	GP_BA+0x24C	R/W	GPIO P2.3管脚数据输入/输出	0x0000_0001
P24_PDIO	GP_BA+0x250	R/W	GPIO P2.4管脚数据输入/输出	0x0000_0001
P25_PDIO	GP_BA+0x254	R/W	GPIO P2.5管脚数据输入/输出	0x0000_0001
P26_PDIO	GP_BA+0x258	R/W	GPIO P2.6管脚数据输入/输出	0x0000_0001
P30_PDIO	GP_BA+0x260	R/W	GPIO P3.0管脚数据输入/输出	0x0000_0001
P31_PDIO	GP_BA+0x264	R/W	GPIO P3.1管脚数据输入/输出	0x0000_0001
P32_PDIO	GP_BA+0x268	R/W	GPIO P3.2管脚数据输入/输出	0x0000_0001
P34_PDIO	GP_BA+0x270	R/W	GPIO P3.4管脚数据输入/输出	0x0000_0001
P35_PDIO	GP_BA+0x274	R/W	GPIO P3.5管脚数据输入/输出	0x0000_0001
P36_PDIO	GP_BA+0x278	R/W	GPIO P3.6管脚数据输入/输出	0x0000_0001
P46_PDIO	GP_BA+0x298	R/W	GPIO P4.6管脚数据输入/输出	0x0000_0001
P47_PDIO	GP_BA+0x29C	R/W	GPIO P4.7管脚数据输入/输出	0x0000_0001
P50_PDIO	GP_BA+0x2A0	R/W	GPIO P5.0管脚数据输入/输出	0x0000_0001
P51_PDIO	GP_BA+0x2A4	R/W	GPIO P5.1管脚数据输入/输出	0x0000_0001
P52_PDIO	GP_BA+0x2A8	R/W	GPIO P5.2管脚数据输入/输出	0x0000_0001
P53_PDIO	GP_BA+0x2AC	R/W	GPIO P5.3管脚数据输入/输出	0x0000_0001
P54_PDIO	GP_BA+0x2B0	R/W	GPIO P5.4管脚数据输入/输出	0x0000_0001

P55_PDIO	GP_BA+0x2B4	R/W	GPIO P5.5管脚数据输入/输出	0x0000_0001

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

位	描述	
[31:1]	**Reserved**	保留
[0]	**PDIO**	**GPIO Px.n管脚数据输入/输出** 写该位可以控制一个GPIO管脚的输出值 0 = 设置相应GPIO管脚为低 1 = 设置相应GPIO管脚为高 读该寄存器得到GPIO管脚状态 例如: 写P00_PDIO即把值写到DOUT (P0_DOUT[0])位上, 读P00_PDIO即读取PIN (P0_PIN[0])的值. **注意1**：写操作不受DATMSK (Px_DATMSK[n])影响 **注意2:** 端口0：Max. n=7,n=2,n=3保留 端口1：Max. n=7,n=1,n=6，n=7保留 端口2：Max. n=7,n=0,n=1，n=7保留 端口3：Max. n=7,n=3,n=7保留 端口4：Max. n=7,n=0,1,5保留 端口5：Max. n=7,n=6,n=7保留

6.6 定时器控制器(TIMER)

6.6.1 概述

定时器控制器包含 2 组 32-位定时器，TMR0 和 TMR1，提供用户便捷的计数定时功能。定时器可执行很多功能，如频率测量，时间延迟，时钟发生，外部输入管脚事件计数和外部捕捉管脚脉宽测量等。

6.6.2 特性

- 2组 32-位定时器，带24位向上计数器和一个8位的预分频计数器
- 每个定时器都有独立的时钟源
- 提供 one-shot, periodic, toggle 和 continuous 四种计数操作模式
- 通过CNT (TIMRTx_CNT[23:0])可读取内部 24 位向上计数器的值
- 支持事件计数功能
- 通过CAPDAT (TIMERx_CAP[23:0])可读取24位 捕捉值
- 支持外部管脚捕捉，可用于脉宽测量
- 支持外部引脚捕捉，可用于复位24位向上定时器
- 如果定时器中断信号产生，支持芯片从空闲/掉电模式唤醒
- 当内部ACMP输出信号产生后可触发捕捉脉宽

6.6.3 框图

定时器控制器的框图和时钟控制如图 6.6-1和图 6.6-2所示：

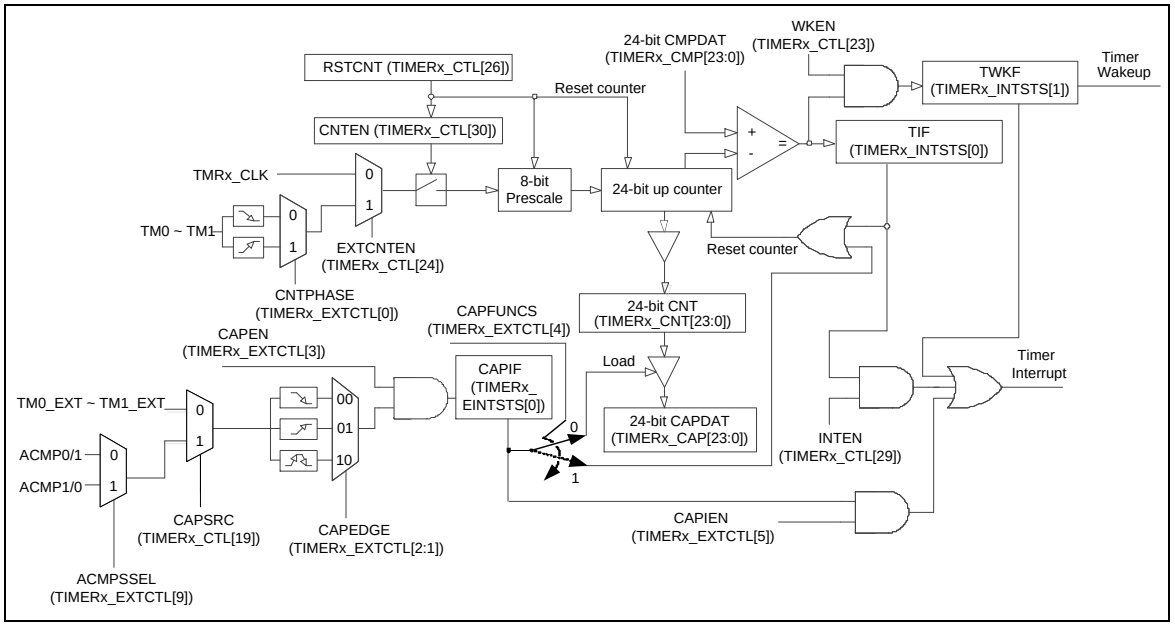


图 6.6-1 定时器控制器框图

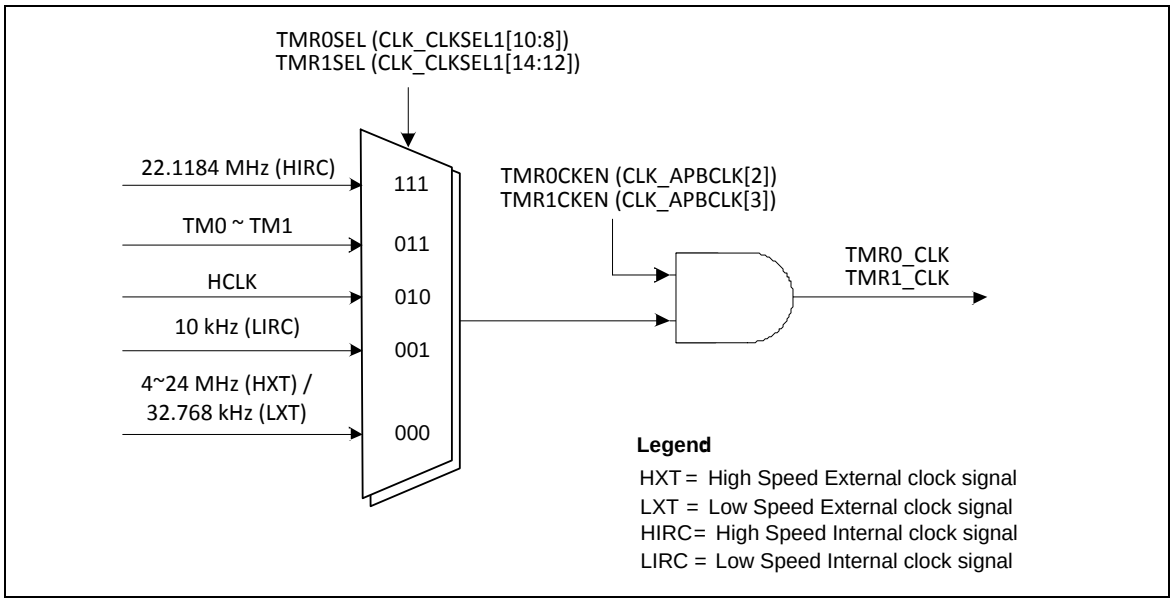


图 6.6-2 定时器控制器时钟源

6.6.4 基本配置

定时器TMR0和TMR1的时钟源可以在寄存器TMxCKEN (CLK_APBCLK[3:2])使能和在寄存器TMR0SEL (CLK_CLKSEL1[10:8]) (TMR0)，T(CLK_CLKSEL1[14:12]) (TMR1)选择不同时钟。

6.6.5 功能描述

6.6.5.1 定时器中断标志

定时器控制器支持两个中断标志：一个是TIF (TIMERx_INTSTS[0])标志，该标志在当定时器计数器值CNT (TIMERx_CNT[23:0])与定时器比较值CMPDAT (TIMERx_CMP[23:0])相匹配时置位，另一个是CAPIF (TIMERx_EINTSTS[0])标志，该标志在当TMx_EXT管脚的变化与CAPEDGE (TIMERx_EXTCTL[2:1])的设置一致时置位。

6.6.5.2 定时器计数模式

定时器控制器提供四种定时器计数模式： one-shot, periodic, toggle-output 和 continuous counting 计数模式。

6.6.5.3 One–shot模式

如果定时器工作在单周期 (one-shot) 模式(TIMERx_CTL[28:27] 为 2’b00) 且 CNTEN (TIMERx_CTL[30])置1)，则定时器的计数器开始计数。一旦CNT (TIMERx_CNT[23:0])达到CMPDAT (TIMERx_CMP[23:0])的值时，TIF (TIMERx_INTSTS[0])标志将变为1，CNT的值和 CNTEN位将由定时器控制器自动清零，然后定时器计数操作停止。与此同时，如果INTEN (TIMERx_CTL[29])位使能，则定时器中断信号产生并送到 NVIC 通知CPU。

6.6.5.4 Periodic 模式

如果定时器工作在周期 (periodic) 模式((TIMERx_CTL[28:27] 为 2'b01) 且 CNTEN (TIMERx_CTL[30])置1)，则定时器的计数器开始向上计数。一旦CNT (TIMERx_CNT[23:0])计数器的值达到CMPDAT (TIMERx_CMP[23:0])的值时，TIF (TIMERx_INTSTS[0])标志将变为1，CNT的值将由定时器控制器自动清零，然后定时器重新计数。与此同时，如果INTEN (TIMERx_CTL[29])使能，则定时器中断信号产生并送到 NVIC 通知 CPU 。在该模式，定时器控制器周期性地操作计数和 与CMPDAT的值比较，直到CNTEN位由用户软件清0。

6.6.5.5 Toggle-output 模式

如果定时器工作在触发输出 (toggle-out) 模式(TIMERx_CTL[28:27] 为 2'b10) ，且CNTEN (TIMERx_CTL[30]) 位置1，则定时器的计数器开始计数。toggle-out 模式的计数操作大部分与周期模式是一样的，除了该模式当TIF (TIMERx_INTSTS[0])被置位时，有相关的TM®0 ~ TM®1管脚会输出信号。因此，管脚TM®0 ~ TM®1上的触发输出信号以 50% 的占空周期反复改变。输出管脚可以是TM®0 ~ TM®1或TM®0_EXT ~ TM®1_EXT，这取决于TGLPINSEL (TIMERx_CTL[18])的设置。

6.6.5.6 Continuous Counting 模式

如果定时器工作在连续计数 (continuous counting) 模式(TIMERx_CTL[28:27] 为 2'b11) ，且CNTEN (TIMERx_CTL[30]) 位置1，则定时器的计数器开始计数。一旦CNT(TIMERx_CNT[23:0])的值达到CMPDAT (TIMERx_CMP[23:0])的值时，TIF(TIMERx_INTSTS[0])标志将变为1，但CNT的值继续保持向上计数。与此同时，如果INTEN (TIMERx_CTL[29])使能，则定时器中断信号产生并送到 NVIC 通知 CPU 。在该模式，用户可以立刻改变不同的CMPDAT值，而不需要停止定时器计数和重新开始定时器计数。

例如，CMPDAT的值设置为 80。当CNT 达到 80时,TIF标志将被置1，定时器计数器继续计数，而且CNT的值将不回到0，而是继续计数，81, 82, 83,⋯ 到 (2^{24} -1)，然后再一次 0, 1, 2, 3, ⋯ 到 2^{24} -1，如此往复。接下来，如果软件改变CMPDAT的值为200并且清除TIF标志位，当CNT的值达到200时，TIF标志将再次变为1,。最后，软件改变CMPDAT的值为500并且清除TIF标志，当CNT的值达到500时，TIF标志将再次变为1。这种模式就是定时器Continuous Counting 模式

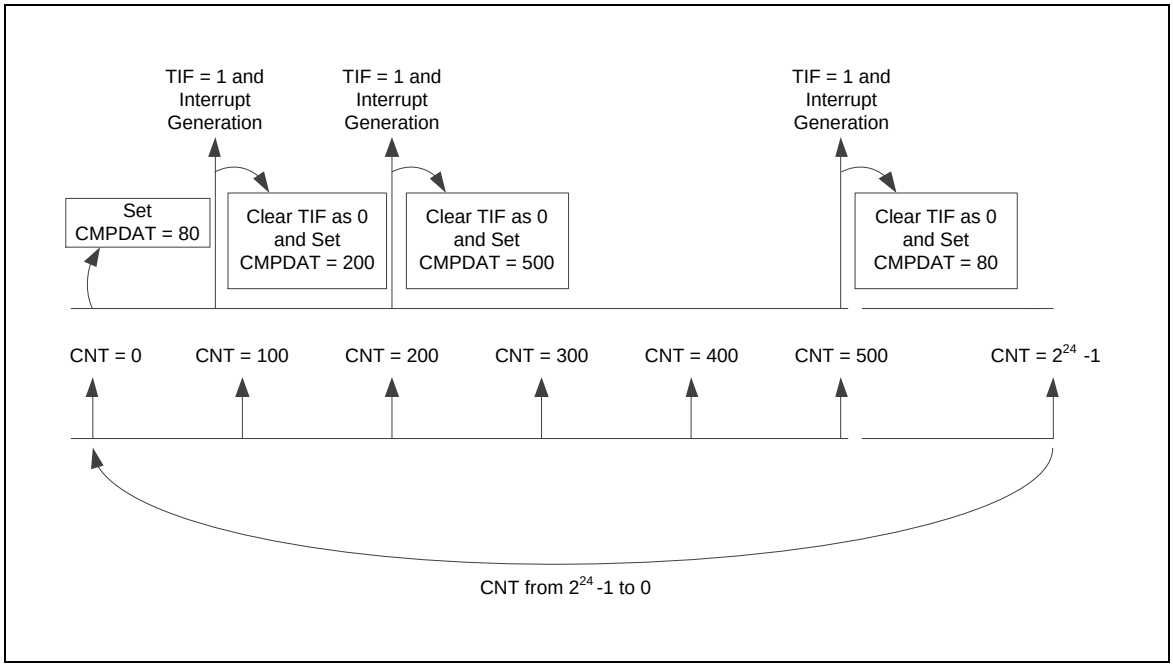


图 6.6-3 Continuous Counting 模式

6.6.5.7 事件计数模式

定时器控制器也提供这样的应用，能对来自管脚TMx (x= 0~1)输入事件计数并将事件的次数反应到CNT (TIMERx_CNT[23:0])的值。也可以称为事件计数功能。该功能下，EXTCNTEN (TIMERx_CTL[24])位需置位并且定时器外设时钟源必须设为 HCLK。

软件可以通过CNTDBEN (TIMERx_EXTCTL[7])位来使能或关闭TMx管脚消抖电路。如果TMx管脚的消抖电路关闭，输入事件频率必须少于1/3 HCLK，如果消抖电路打开，输入事件的频率须小于1/8 HCLK，以保证CNT的值是正确的。软件也可以通过设置CNTPHASE (TIMERx_EXTCTL[0])来选择边沿检测TMx管脚的相位。

事件计数模式下，定时器计数操作模式可以设置为one-shot, periodic 和 continuous counting模式，通过CNT (TIMERx_CNT[23:0])寄存器来计算来自TMx管脚的输入事件。

6.6.5.8 外部捕捉功能

输入捕捉或复位功能是用来捕捉或复位计数器的值。捕捉功能通过CAPSEL (TIMERx_EXTCTL[8])寄存器，可配置为free-counting和trigger-counting捕捉模式。free-counting捕捉模式,复位模式, trigger-counting捕捉模式详细描述如下。

6.6.5.9 Free-Counting 捕捉模式

事件捕捉功能是当检测到TMx_EXT管脚(x= 0~1)边沿电平有变化时，CNT (TIMERx_CNT[23:0])会送到CAPDAT (TIMERx_CAP[23:0])。在该模式下，需把CAPSEL (TIMERx_EXTCTL[8])和CAPFUNCS (TIMERx_EXTCTL[4])位设置为0，用来选择TMx_EXT变化时用作事件捕捉功能，而且定时器外设时钟源必须设为HCLK。

软件可以通过CAPDBEN (TIMERx_EXTCTL[6])位来使能或关闭TMx_EXT管脚消抖电路。在TMx_EXT的消抖电路关闭时，TMx_EXT管脚的转变频率必须少于1/3 HCLK，在TMx_EXT的消抖电路打开时，TMx_EXT管脚的转变频率必须少于1/8 HCLK，以保证捕捉功能能够正常工作。软件也可以通过设置CAPEDGE (TIMERx_EXTCTL[2:1])位来选择TMx_EXT管脚的边沿转变检测方式。

在事件捕捉模式，用户不用考虑定时器计数器工作模式的选择，只有当检测到TMx_EXT管脚有边沿变化时捕捉事件才会发生。

如果CPU不清除CAPIF (TIMERx_EINTSTS[0])状态标志，用户应知道此时的定时器会保持TIMERx_CAP寄存器的值不变，会丢掉新的捕捉值。这种操作模式如表 6.6-1所示。

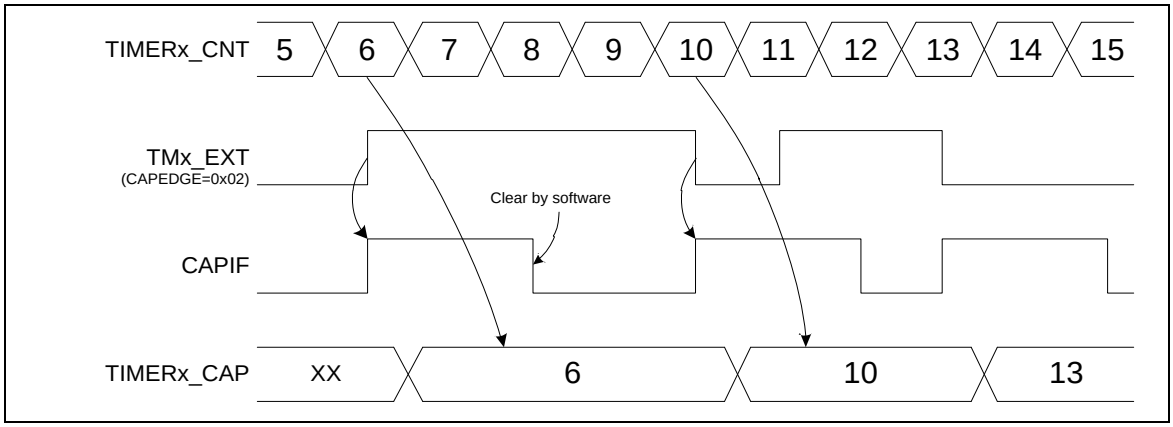


图 6.6-4 Free-Counting捕捉模式

6.6.5.10 外部复位计数模式

当检测到TMx_EXT (x= 0~1)有边沿转变时，定时器同样提供事件复位计数器功能来复位CNT (TIMERx_CNT[23:0])的值。在该模式，大部分设置与事件捕捉功能相同，除了CAPFUNCS (TIMERx_EXTCTL[4])位必须设置为1来选择TMx_EXT转变时用作为事件复位计数器。这种工作模式如表6.6-1所示。

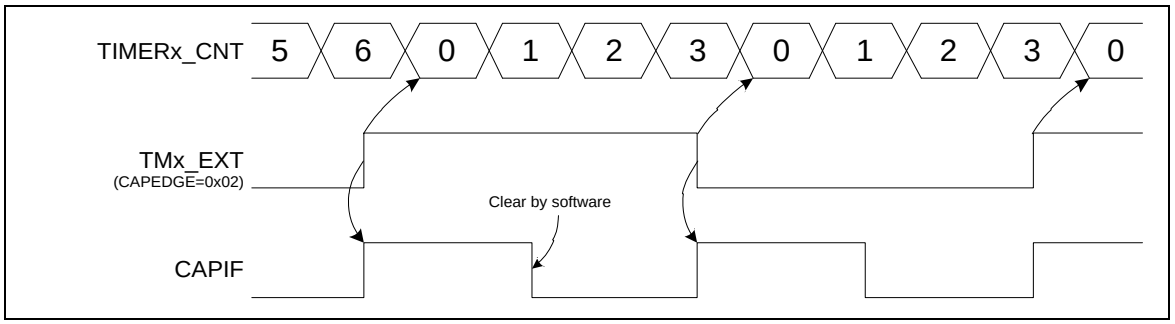


图 6.6-5 外部复位计数模式

6.6.5.11 Trigger-Counting 捕捉模式

如果CAPSEL (TIMERx_EXTCTL[8])位和CAPEN (TIMERx_EXTCTL[3])被置1，CAPFUNCS (TIMERx_EXTCTL[4])被清0，CNT将会被清零，然后当TMx_EXT (x= 0~1)管脚触发条件产生时捕捉到数据放入CAPDAT寄存器。TMx_EXT触发边沿可以通过CAPEDGE (TIMERx_EXTCTL[2:1])位来设置。具体的操作办法请参考表 6.6-1.当TMx_EXT管脚触发产生后，CAPIF (TIMERx_EINTSTS[0])被置1，内部产生一个信号被发送到NVIC 来告知CPU CAPIEN (TIMERx_EXTCTL[5])是否被置1. 在TMx_EXT的消抖电路关闭时，TMx_EXT管脚的转变频率必须少于1/3 HCLK，在TMx_EXT的消抖电路打开时，TMx_EXT管脚的转变频率必须少于1/8 HCLK，以保证捕捉功能能够正常工作。通过设置CAPDBEN (TIMERx_EXTCTL[6])可以使能或禁止TMx_EXT管脚的捕捉de-bounce功能。

功能	CAPSEL (TIMERx_EXTCTL[8])	CAPFUNCS TIMERx_EXTCTL[4])	CAPEDGE TIMERx_EXTCTL[2:1])	操作描述
Free-counting捕捉模式	0	0	00	,TMx_EXT (x= 0~1) 管脚检测到由1变0， CNT的值被捕捉到CAPDAT.
	0	0	01	TMx_EXT (x= 0~1) 管脚检测到由0变1. CNT 的值被捕捉到CAPDAT.
	0	0	10	TMx_EXT (x= 0~1) 管脚检测到由1变0或由0变1，CNT的值被捕捉到 CAPDAT.
	0	0	11	保留
外部复位计数模式	0	1	00	TMx_EXT (x= 0~1) 管脚检测到由1变0. CNT被清0.
	0	1	01	TMx_EXT (x= 0~1) 管脚检测到由0变1. CNT被清 0.
	0	1	10	TMx_EXT (x= 0~1) 管脚检测到由1变0或由0变1. CNT的值被清 0.

	0	1	11	保留
Trigger-Counting 捕捉模式	1	0	00	下降沿触发: TMx_EXT (x= 0~1) 管脚第一个由1变0，将使CNT 清0 ，然后开始计数, 当第2个由1变0产生，计数停止。
	1	0	01	上升沿触发: TMx_EXT (x= 0~1) 管脚第一个由0变1，将使CNT清0，然后开始计数，当第2个由0变1产生，计数停止.
	1	0	10	电平切换触发: TMx_EXT (x= 0~1) 脚检测到由1变0，将使CNT清0，然后开始计数, 当由0变1停止计数。
	1	0	11	电平切换触发: TMx_EXT (x= 0~1) 脚检测到由0变1，将使CNT清0然后开始计数，当由1变0停止计数.

表 6.6-1 输入捕捉模式操作流程

6.6.5.12 ACMP启动捕捉触发

外部捕捉功能可以被 ACMP0或ACMP1输出信号触发。用户可以设置ACMPSSEL (TIMERx_EXTCTL[9]) 来决定哪个ACMP 输出信号作为TMx_EXT (x= 0~1) 管脚的捕捉源. 具体的捕捉功能设置与之前外部捕捉模式是一样的。

6.6.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
TMR 基地址 **TMR_BA = 0x4001_0000**				
TIMER0_CTL	TMR_BA+0x00	R/W	Timer0 控制和状态寄存器	0x0000_0005
TIMER0_CMP	TMR_BA+0x04	R/W	Timer0比较寄存器	0x0000_0000
TIMER0_INTSTS	TMR_BA+0x08	R/W	Timer0中断状态寄存器	0x0000_0000
TIMER0_CNT	TMR_BA+0x0C	R	Timer0数据寄存器	0x0000_0000
TIMER0_CAP	TMR_BA+0x10	R	Timer0捕捉数据寄存器	0x0000_0000
TIMER0_EXTCTL	TMR_BA+0x14	R/W	Timer0外部控制寄存器	0x0000_0000
TIMER0_EINTSTS	TMR_BA+0x18	R/W	Timer0外部中断状态寄存器	0x0000_0000
TIMER1_CTL	TMR_BA+0x20	R/W	Timer1控制和状态寄存器	0x0000_0005
TIMER1_CMP	TMR_BA+0x24	R/W	Timer1比较寄存器	0x0000_0000
TIMER1_INTSTS	TMR_BA+0x28	R/W	Timer1中断状态寄存器	0x0000_0000
TIMER1_CNT	TMR_BA+0x2C	R	Timer1数据寄存器	0x0000_0000
TIMER1_CAP	TMR_BA+0x30	R	Timer1捕捉数据寄存器	0x0000_0000
TIMER1_EXTCTL	TMR_BA+0x34	R/W	Timer1外部控制寄存器	0x0000_0000
TIMER1_EINTSTS	TMR_BA+0x38	R/W	Timer1外部中断状态寄存器	0x0000_0000

6.6.7 寄存器描述

定时器控制寄存器(TIMERx_CTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CTL	TMR_BA+0x00	R/W	Timer0控制和状态寄存器	0x0000_0005
TIMER1_CTL	TMR_BA+0x20	R/W	Timer1控制和状态寄存器	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	CNTEN	INTEN	OPMODE		RSTCNT	ACTSTS	EXTCNTEN
23	22	21	20	19	18	17	16
WKEN	Reserved			CAPSRC	TGLPINSEL	CMPCTL	Reserved
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PSC							

位	描述	
[31]	**ICEDEBUG**	**仿真器 (ICE) 调试模式响应禁止位 （写保护位）** 0 = 仿真器调试模式响应影响定时器计数。 当调试器调试模式响应时，定时器计数器将被固定住。 1 = 仿真器调试模式响应禁用。 无论调试器调试模式响应与否，定时器计数器将持续计数下去。 **注意:** 该位为写保护 位。详情请参考SYS_REGLCTL 寄存器.
[30]	**CNTEN**	**定时器计数使能位** 0 = 停止/暂停计数 1 = 开始计数 **注意**1：在停止状态，设置CNTEN为 1 将使能 24-位向上计数器从上次停止的计数值继续计数。 **注意**2：在 one-shot 模式下(TIMERx_CTL[28:27] = 2'b00)，当相应的定时中断标志TIF (TIMERx_INTSTS[0])产生时，该位由硬件自动清零。
[29]	**INTEN**	**定时器中断使能位** 0 = 禁用定时器中断 1 = 使能定时器中断 注意：如果该位使能，当定时器中断标志TIF置 1， 定时器中断信号产生并通知CPU。
[28:27]	**OPMODE**	**定时器计数模式选择** 00 = 定时器工作在单触发模式 (one-shot) 01 = 定时器工作在周期模式 (Periodic) 10 = 定时器工作在Toggle 模式

位	描述	
		11 = 定时器工作在连续计数模式(Continuous Counting）
[26]	**RSTCNT**	**定时器复位位** 设置该位将复位24位向上计数器CNT (TIMERx_CNT[23:0])的值，且如果ACTSTS (TIMERx_CTL[25])位为1，则会把CNTEN (TIMERx_CTL[30])强制清0. 0 = 该位写 0 无效 1 =复位 8-位预分频计数器，内部 24-位向上计数器的值和 CNTEN 位
[25]	**ACTSTS**	**定时器激活状态位（只读）** 该位表示24位向上计数器的状态。 0 = 定时器计数器未激活 1 = 定时器计数器激活
[24]	**EXTCNTEN**	**事件计数模式使能位** 该位用来使能外部计数管脚功能。 0 = 禁用事件计数器模式 1 = 使能事件计数器模式 注意：当定时器用作事件计数器，该位需要设置成1.并选择HCLK作为定时器时钟源。
[23]	**WKEN**	**唤醒功能使能位** 如果该位置1，当定时器中断标志TIF(TIMERx_INTSTS[0])被置1，且INTEN (TIMERx_CTL[29])被使能，定时器中断信号将产生一个唤醒触发事件给CPU 0 = 如果定时器中断信号产生，唤醒功能禁止 1 = 如果定时器中断信号产生，唤醒功能使能
[22:20]	**Reserved**	保留.
[19]	**CAPSRC**	**捕捉管脚源选择位** 0 = 捕捉功能的源是来自 TMx_EXT (x= 0~1)管脚 1 = 捕捉功能的源是来自内部ACMP输出信号。用户可以设置CAPSRCMP (TIMERx_EXTCTL[9])来决定哪个ACMP信号是定时器捕捉源。
[18]	**TGLPINSEL**	**Toggle-Output 管脚选择** 0 = Toggle 模式输出到TMx 脚(定时器事件计数管脚). 1 = Toggle模式输出到TMx_EXT (定时器外部捕捉管脚).
[17]	**CMPCTL**	**定时器比较模式选择位** 0 = one-shot 或 periodic 模式选择禁止 当用户更新 CMPDAT，同时定时器运行在 one-shot 或 periodic 模式，CNT 将被复位到默认值 1 = one-shot 或 periodic 模式选择使能 当用户要更新 CMPDAT，同时定时器运行在 one-shot 或 periodic 模式，要求如下： 如果要更新 CMPDAT 的值 > CNT，CMPDAT 将被更新，CNT 的值将继续计数。 如果要更新 CMPDAT 的值 = CNT，定时器溢出中断将立即产生。 如果要更新CMPDAT 的值 < CNT，CNT将被复位到默认值
[16:8]	**Reserved**	保留.
[7:0]	**PSC**	**预分频计数器** 定时器的输入时钟源或事件计数源被(PSC+1)预分频，然后再输入到定时器。如果 该位为0(PSC = 0)，表示不分频。

定时器比较寄存器(TIMERx_CMP)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CMP	TMR_BA+0x04	R/W	Timer0比较寄存器	0x0000_0000
TIMER1_CMP	TMR_BA+0x24	R/W	Timer1比较寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留.
[23:0]	**CMPDAT**	**定时器比较值** CMPDAT是24-位比较寄存器。当内部 24-位向上计数器的值等于CMPDAT的值时，TIF (TIMERx_INTSTS[0]定时器中断标志)将被置1。 超时溢出周期 = (定时器输入时钟周期) * (8-bit PSC + 1) * (24-bit CMPDAT) **注意**1：不能向CMPDAT里写 0x0 或 0x1，否则内核将运行到未知状态。 **注意**2：当定时器工作在 continuous counting 模式，即使软件写一个新的值到CMPDAT，24-位向上计数定时器将保持继续计数。如果定时器工作在其他模式，如果软件写一个新的值到CMPDAT，定时器将使用新比较值并退出当前计数，开始重新计数。

定时器中断状态寄存器(TIMERx_INTSTS)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_INTSTS	TMR_BA+0x08	R/W	Timer0中断状态寄存器	0x0000_0000
TIMER1_INTSTS	TMR_BA+0x28	R/W	Timer1中断状态寄存器	0x0000_0000

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

位	描述	
[31:2]	**Reserved**	保留.
[1]	**TWKF**	**定时器唤醒标志** 该位表示定时器的中断唤醒标志状态。 0 = 定时器不会引起CPU 唤醒 1 = 如果定时器中断信号产生， CPU 从空闲或掉电模式唤醒 **注意:** 该位必须通过软件写1清0
[0]	**TIF**	**定时器中断标志** 当内部 24-位向上计数定时器CNT (TIMERx_CNT[23:0])的值与定时器比较值CMPDAT (TIMERx_CMP[23:0])匹配时，该位会显示其中断状态。. 0 = 无影响 1 = CNT的值与CMPDAT的值相匹配 注意：该位写 1 清零。

定时器数据寄存器(TIMERx_CNT)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CNT	TMR_BA+0x0C	R	Timer0数据寄存器	0x0000_0000
TIMER1_CNT	TMR_BA+0x2C	R	Timer1数据寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留.
[23:0]	**CNT**	**定时器数据寄存器** 该位可以反映内部24位定时器计数器的值或外部TMx(x=0~1)脚输入事件计数器的值 如果EXTCNTEN (TIMERx_CTL[24])为0, 用户可以读CNT 的值来获取当前24位向上计数器的值 如果EXTCNTEN (TIMERx_CTL[24])为1, 用户可以读 CNT 的值来获取当前24位事件计数器的值

定时器捕捉数据寄存器(TIMERx_CAP)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CAP	TMR_BA+0x10	R	Timer0捕捉数据寄存器	0x0000_0000
TIMER1_CAP	TMR_BA+0x30	R	Timer1捕捉数据寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留.
[23:0]	**CAPDAT**	**定时器捕捉数据寄存器** 当 CAPEN (TIMERx_EXTCTL[3])位被置 1, CAPFUNCS (TIMERx_EXTCTL[4])位为 0, TMx_EXT 管脚的变化与 CAPEDGE (TIMERx_EXTCTL[2:1])设定相匹配时, CAPIF (TIMERx_EINTSTS[0])将被置 1 且当前定时器计数器 CNT (TIMERx_CNT[23:0])的值将被自动载入到 CAPDAT.

定时器外部控制寄存器(TIMERx_EXTCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_EXTCTL	TMR_BA+0x14	R/W	Timer0外部控制寄存器	0x0000_0000
TIMER1_EXTCTL	TMR_BA+0x34	R/W	Timer1外部控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**ACMPSSEL**	**CAPSEL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNTDBEN	**CAPDBEN**	**CAPIEN**	**CAPFUNCS**	**CAPEN**	**CAPEDGE**		**CNTPHASE**

位	描述	
[31:10]	**Reserved**	保留.
[9]	**ACMPSSEL**	**用于触发捕捉功能的ACMP源选择** 对于 Timer 0: 0 = TIMER0的捕捉功能源是来自ACMP0 的输出信号. 1 = TIMER0的捕捉功能源是来自ACMP1 的输出信号. 对于 Timer 1: 0 = TIMER0的捕捉功能源是来自ACMP1 的输出信号. 1 = TIMER1的捕捉功能源是来自ACMP0 的输出信号.
[8]	**CAPSEL**	**捕捉模式选择位** 0 = 定时器计数复位功能或定时器捕捉功能free-counting 模式. 1 = 定时器捕捉功能触发计数模式
[7]	**CNTDBEN**	**定时器外部计数器输入管脚防抖动使能位** 0 = TMx (x= 0~1)脚防抖禁用 1 = TMx (x= 0~1)管脚防抖使能 注意：如果该位使能，TMx管脚边沿检测带防抖动电路。
[6]	**CAPDBEN**	**定时器外部捕捉输入管脚防抖动使能位** 0 = TMx_EXT (x= 0~1)管脚防抖禁用 1 = TMx_EXT (x= 0~1)管脚防抖使能 注意1：如果该位使能，TMx_EXT管脚边沿检测带防抖电路。 注意2：防抖电路不支持ACMP输出
[5]	**CAPIEN**	**定时器外部捕捉中断使能位**

位	描述	
		0 = TMx_EXT (x= 0~1)管脚检测中断禁止. 1 = TMx_EXT (x= 0~1)管脚检测中断使能 注意：CAPIEN用于使能定时器外部中断。如果CAPIEN使能，当CAPIF (TIMERx_EINTSTS[0])标志设为1时，定时器会产生一个外部捕捉中断信号。 例如, 当 CAPIEN = 1, CAPEN = 1, 且 CAPEDGE = 00, TMx_EXT 管脚上的由1变0变化，将 导致CAPIF 被置位，然后中断信号产生并发送到NVIC并告知CPU.
[4]	**CAPFUNCS**	**捕捉功能选择位** 0 =外部捕捉模式使能. 1 =外部复位模式使能. **注意1:** 当 CAPFUNCS为 0, TMx_EXT (x= 0~1)管脚上的变化用于保存 24-位定时器计数值到CAPDAT. **注意2:** 当 CAPFUNCS为 1, TMx_EXT (x= 0~1) 管脚上的变化用于复位复位24-位定时器计数器
[3]	**CAPEN**	**定时器外部捕捉管脚使能** 该位使能TMx_EXT管脚 0 = TMx_EXT (x= 0~1)管脚禁止。 1 = TMx_EXT (x= 0~1)管脚使能。
[2:1]	**CAPEDGE**	**定时器外部捕捉管脚边沿检测选择** 00 = TMx_EXT (x= 0~1)管脚上 下降沿将被检测。 01 = TMx_EXT (x= 0~1)管脚上 上升沿将被检测。 10 = TMx_EXT (x= 0~1)管脚上 上升沿或下降沿将被检测。 11 = 保留。
[0]	**CNTPHASE**	**定时器外部计数管脚相位检测选择** 该位表示TMx (x= 0~1)脚相位检测。 0 =管脚的下降沿将被统计。 1 =管脚的上升沿将被统计。

定时器外部中断状态寄存器(TIMERx_EINTSTS)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_EINTSTS	TMR_BA+0x18	R/W	Timer0外部中断状态寄存器	0x0000_0000
TIMER1_EINTSTS	TMR_BA+0x38	R/W	Timer1外部中断状态寄存器	0x0000_0000

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

位	描述	
[31:1]	**Reserved**	保留.
[0]	**CAPIF**	**定时器外部捕捉中断标志位** 该位表示定时器外部捕捉中断标志状态。. 当TEXEN (TEXCON[3]使能，且TMx_EXT管脚选择为外部捕捉功能，在TMx_EXT管脚边沿转变与TEX_EDGE (TEXCON[2:1]) 设定相匹配时，该标志将被硬件置位。 0 = TMx_EXT (x= 0~1)管脚中断没有发生. 1 = TMx_EXT (x= 0~1)管脚中断发生. **注意1**：该位写 1 清零。 **注意2:** 当CAPEN (TIMERx_EXTCTL[3]) 位被置1, CAPFUNCS (TIMERx_EXTCTL[4]) 位为 0, TMx_EXT (x= 0~1) 管脚的变化与CAPEDGE (TIMERx_EXTCTL[2:1])设定匹配, 该位将被硬件置1. **注意3:**CPU 清 CAPIF 状态前会检测到一个新的捕捉事件.如果以上条件发生, 定时器将保持TIMERx_CAP的值不变并丢掉新的捕捉值.

6.7 增强型PWM 发生器

6.7.1 概述

NuMicro® Mini58系列微控制器内置了针对马达驱动应用的PWM单元（PWM0通道）。PWM0支持6路PWM发生器，可以配置为相互独立的6路PWM输出，通道0到5，或配置成3对分别带有可编程死区发生器的互补PWM，通道0和1，2和3，4和5。

每一对PWM共用8位预分频器，有6组时钟分频器，分别提供5种分频系数(1, 1/2, 1/4, 1/8, 1/16)。每一路PWM输出有独立的16位计数器用以PWM控制，16位的比较器用以调节占空比。6组发生器提供12路独立PWM中断标志，相关PWM通道的周期和占空比如果与计数器相符，PWM中断将会被硬件置1。每一路PWM中断各有中断使能位。每个PWM发生器可以配置成单次模式（产生一个PWM信号周期）或者循环模式（连续输出PWM波形）。

为了防止PWM输出引脚输出不稳定波形，16位周期向下计数器和比较器带有双缓冲区。当用户向计数器和比较器寄存器写数据时，所写数据会在下一个周期加载。这种双缓存的特质保证了PWM稳定平滑的输出。

除了PWM，电机控制还需要Timer、ACMP和ADC共同工作。为了更精确的控制电机，我们提供一些寄存器，不仅可以配置PWM同时也可以配置Timer、ADC和ACMP。这样做，可以节省更多的CPU时间且便于控制电机，尤其是在直流无刷电机中。

6.7.2 特性

PWM0 有以下特性:

- 6个独立的16位PWM占空比控制单元控制最多6个PWM输出管脚:
  - 6路独立输出—PWM0_CH0, PWM0_CH1, PWM0_CH2, PWM0_CH3, PWM0_CH4, 和 PWM0_CH5
  - 三组互补PWM对--(PWM0_CH0, PWM0_CH1), (PWM0_CH2, PWM0_CH3)和 (PWM0_CH4, PWM0_CH5)。每组PWM对的一根引脚与另外一根引脚互补，且可以插入可编程的死区时间。
  - 三组同步PWM对--(PWM0_CH0, PWM0_CH1), (PWM0_CH2, PWM0_CH3)和 (PWM0_CH4, PWM0_CH5)，每组PWM对的引脚相位同步。
- 支持组控制，–PWM0_CH0,PWM0_CH2 ,PWM0_CH4 输出同步，PWM0_CH1, PWM0_CH3 ,PWM0_CH5 输出同步
- 有单次模式（只支持边沿对齐类型）或者自动装载模式
- 16位精度
- 支持边沿对齐，中心对齐和精准中心对齐三种模式
- 在中心对齐和精确中心对齐模式中，支持不对称发生模式
- 中心对齐模式和精确中心对齐模式中，支持中心点加载
- 一对PWM中，支持可编程死区发生器
- PWM0_CH0到PWM0_CH5每路PWM有独立的极性控制
- 硬件刹车保护
  - 支持软件触发

- 两种中断类型:
  - 向下计数器比较匹配（边沿对齐和中心对齐）或者下溢（边沿对齐模式）的时候，提出同步中断请求。
  - 发生外部刹车条件。
    - BKP0: EINT0或CPO1
    - BKP1: EINT1或CPO0

- PWM缺省状态下为正逻辑，PWM口的有效状态由极性控制寄存器来控制
- 支持屏蔽对齐功能
- 支持每个通道独立触发ADC中断：上升值比较，周期值比较，下降值比较（中心对齐），周期比较。
- 定时器比较事件可以触发PWM，实现直流无刷电机相切换
- 支持ACMP输出事件触发PWM强制输出一个周期的低电平，该特性可用于步进电机驱动。
- 支持中断中断累计功能。

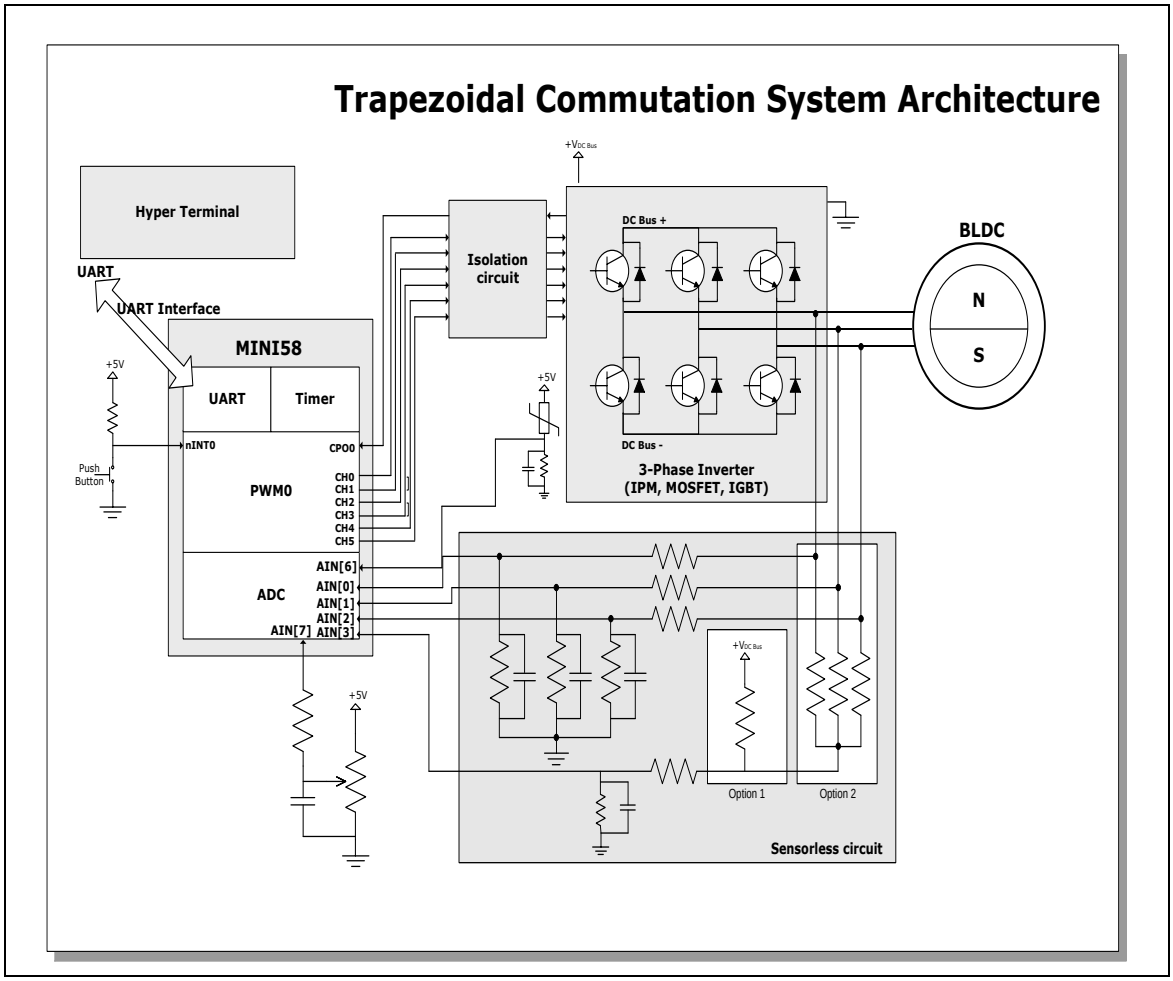


图 6.7-1 应用电路框图

6.7.3 方块图

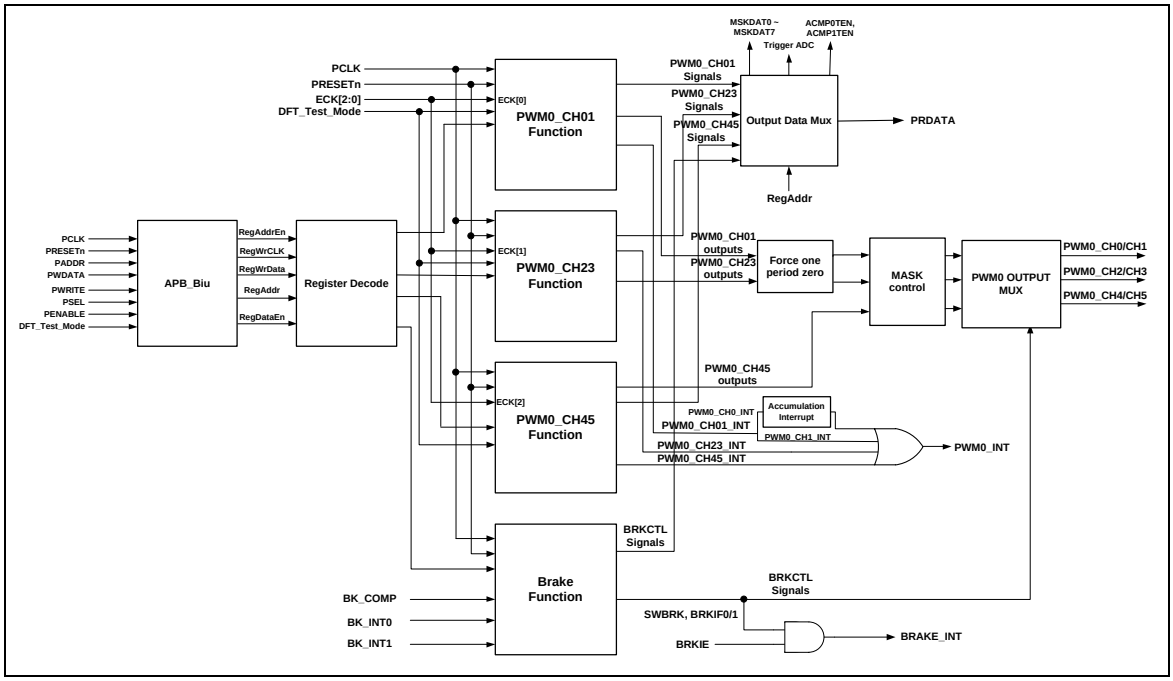


图 6.7-2 PWM 方框图

图 6.7-3 可以看出PWM的对架构，如通道0/1组成一对PWM，通道2/3是另一对。

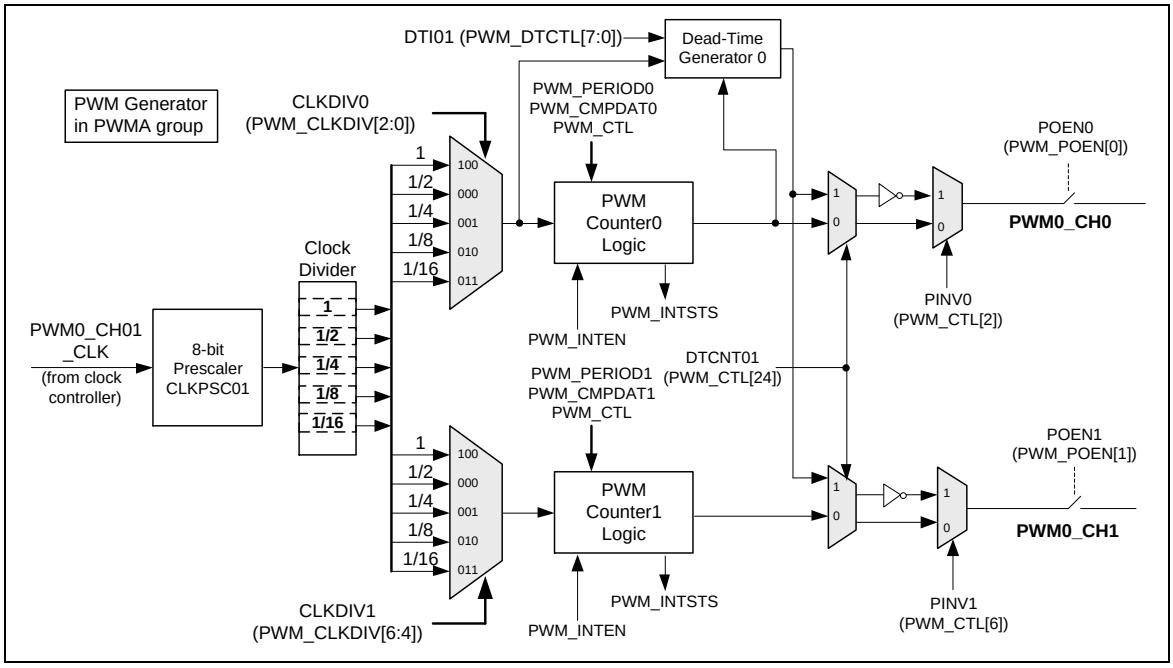


图 6.7-3 PWM CH0/1发生器架构图

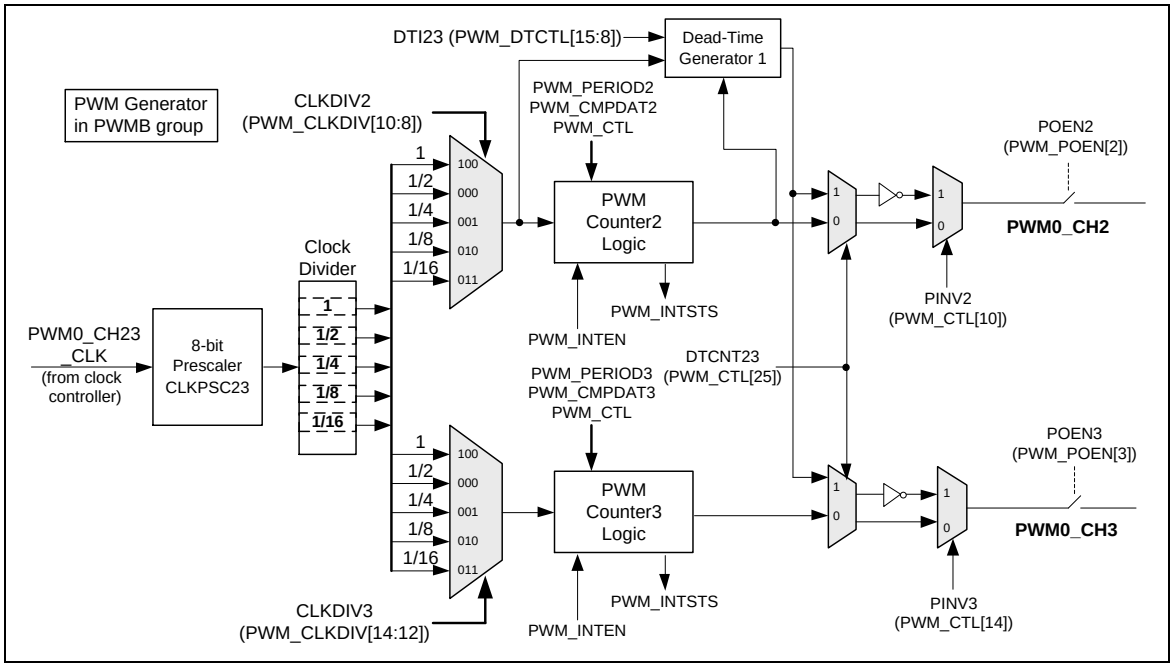


图 6.7-4 PWM CH2/3发生器架构图

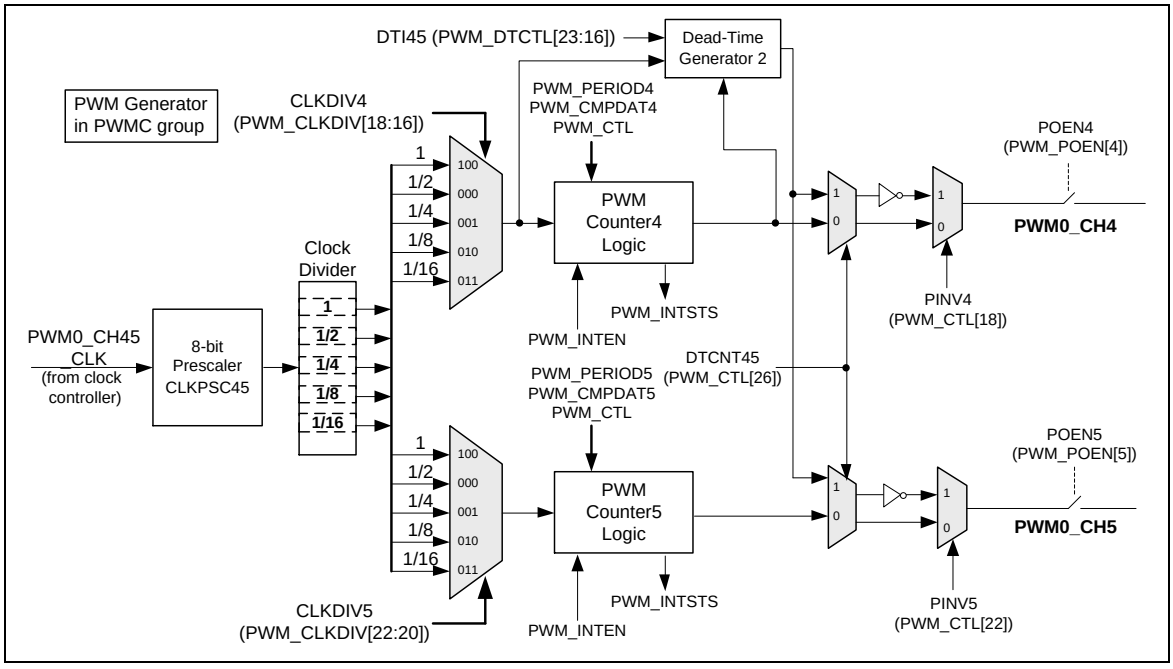


图 6.7-5 PWM CH4/5发生器架构图

6.7.4 基本配置

PWM0的管脚功能选择在 SYS_P0_MFP 和 SYS_P2_MFP 寄存器中.

PWM0 的时钟可以在 CLK_APBCLK[22:20]位配置. PWM0 时钟源需要选择为PCLK。

6.7.5 功能描述

6.7.5.1 PWM计数器工作模式

支持3种类型：边沿对齐，中心对齐，精确中心对齐。

以下是公式周期和占空比的计算公式。

<u>边沿对齐（向下计数）：</u>

占空比 = (CMP +1)/(PERIOD +1)

高电平时间 = (CMP +1) * (时钟周期)

周期= (PERIOD +1) * (时钟周期)

<u>中心对齐（上下计数）：</u>

占空比 = (PERIOD - CMP) / (PERIOD +1)

高电平时间= (PERIOD -CMP) * 2 * (时钟周期)

周期= (PERIOD +1) * 2 * (时钟周期)

<u>精确中心对齐（上下计数）：</u>

占空比 = (PERIOD - (CMP + 1) * 2) / PERIOD

高电平时间=(PERIOD - (CMP + 1) * 2)* (时钟周期)

周期= (PERIOD) * (时钟周期)

<u>边沿对齐**PWM**（向下计数）</u>

在边沿模式输出时，16位PWM计数器会在每个周期开始向下计数，与上个周期CMPn值进行比较，当比较值相符就会使对应通道PWM0_CHn输出为高电平，置位比较标志位CMPDIF。计数器会继续向下计数至0，这样会触发PWM0_CHn输出为低电平，当前CMPn和PERIODn 值在CNTMODEn=1情况下会更新，PIF周期中断标志位置位。

图 6.7-6, 图 6.7-7和 图 6.7-8 可以看出边沿对齐模式的时序图.

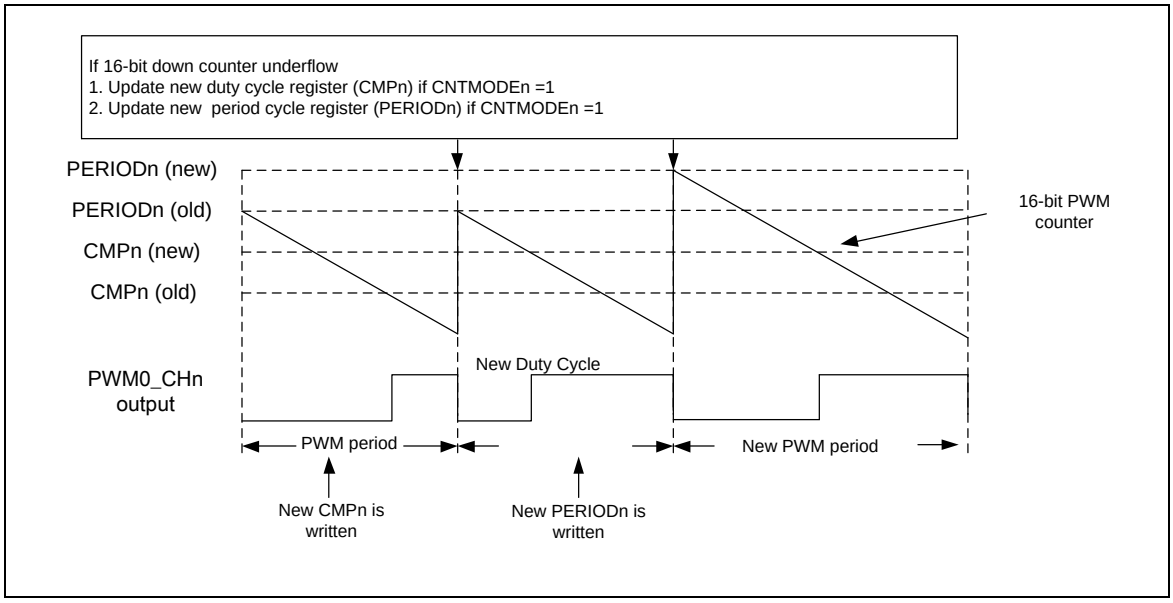


图 6.7-6 边沿对齐模式PWM

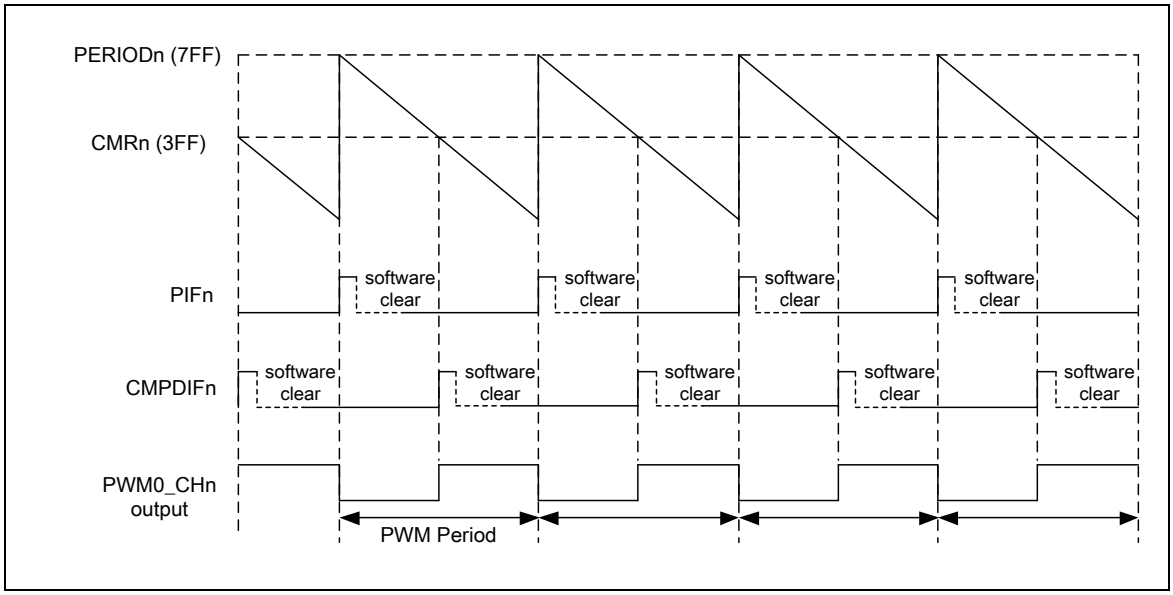


图 6.7-7 PWM边沿对齐时序图

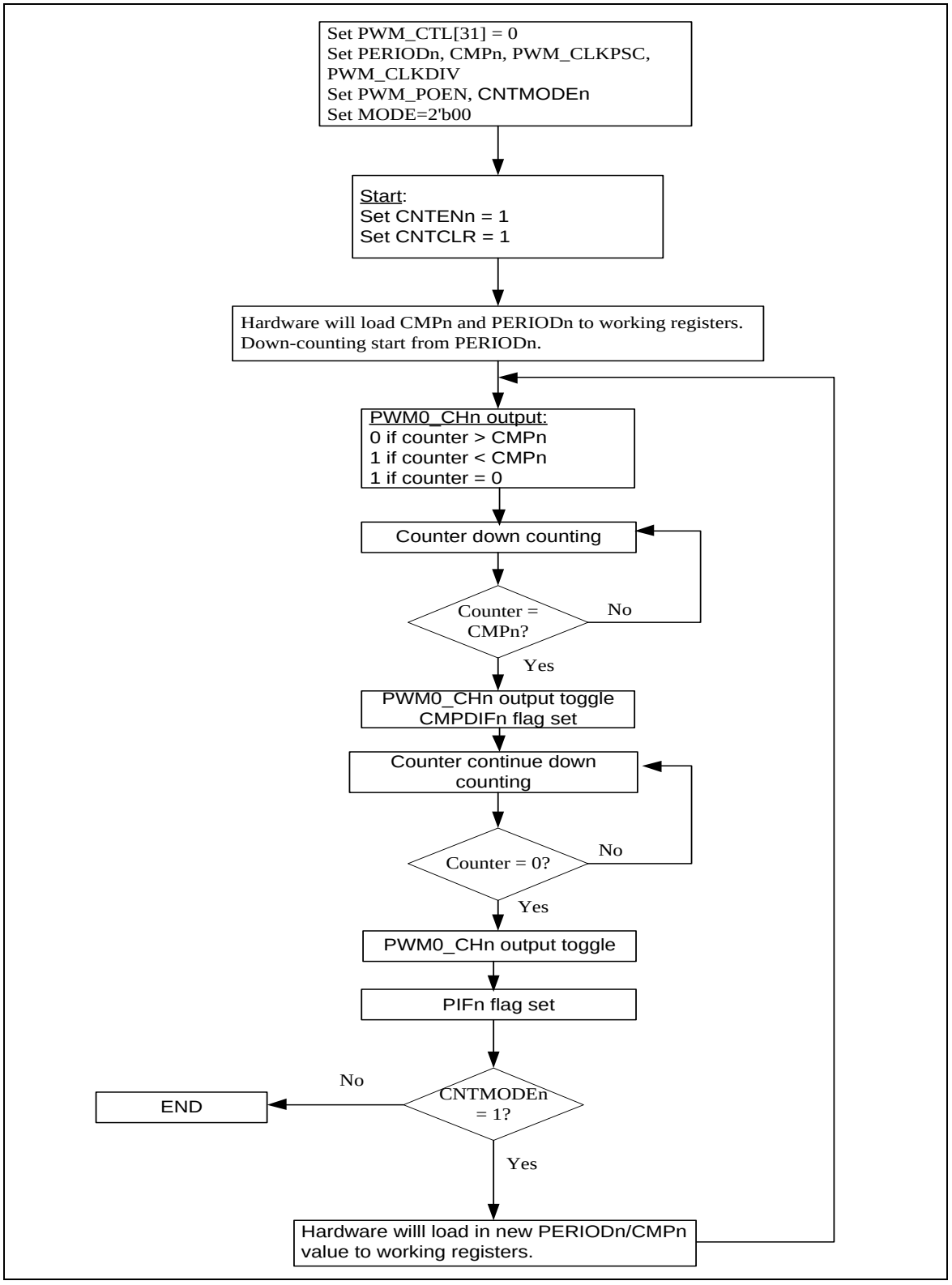


图 6.7-8 边沿对齐流程图

PWM的周期和占空比由PERIODn 和CMPn 决定。PWM计数器工作时序见图 6.7-10脉冲宽度调制遵循以下

公式，PWM比较器工作原理见图 6.7-9。注意相关的GPIO脚需要配置成PWM功能（使能PWM_POEN）。

PWM 频率 = HCLK / ((CLKPSCnm +1)*(时钟分频)) /(PERIOD +1); 这里 nm, 可以是 01, 23 或 45 ，这取决于PWM的通道选择

占空比 = (CMP +1)/(PERIOD +1)

CMP>= PERIOD: PWM 输出总是high

CMP<PERIOD: PWM low 宽度= (PERIOD -CMP) unit; PWM high 宽度 = (CMP+1) unit

CMP = 0: PWM low 宽度 = (PERIOD) unit; PWM high 宽度 = 1 unit

注意: 1. Unit =一个PWM时钟周期.

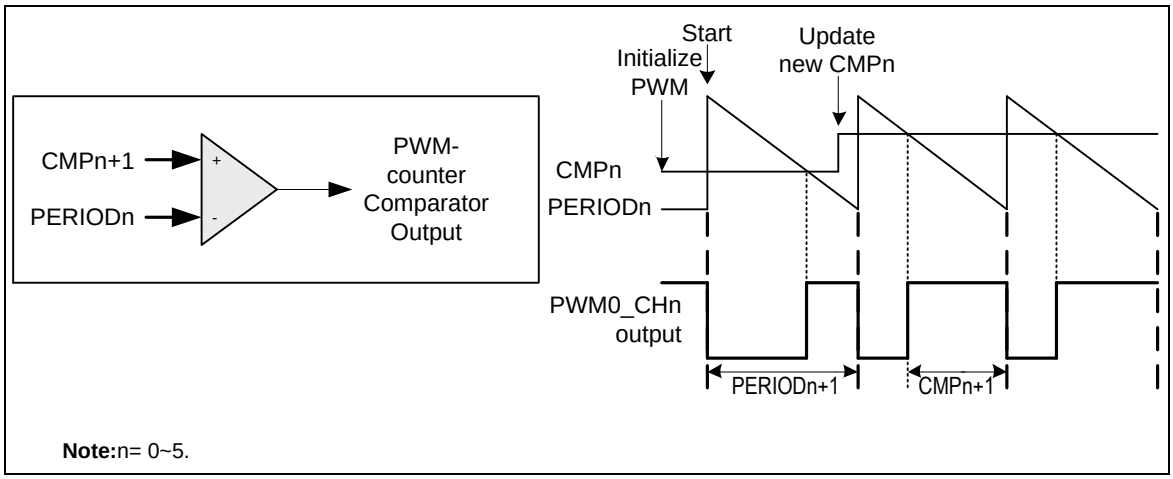


图 6.7-9 内部比较器工作原理

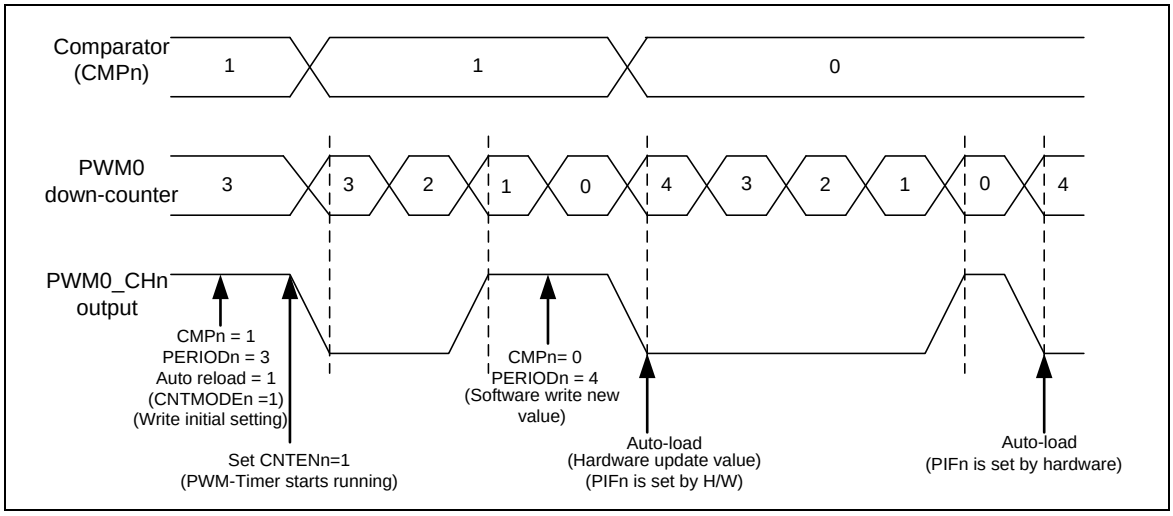


图 6.7-10 PWM工作时序

中心对齐模式（向上/向下计数）

中心对齐模式，PWM的上/下计数模式时间参数配置好以后将会产生相应的PWM波形。当PWM计数器的值计到CMPn (旧值)，将会从0重新开始计数。这将使PWM0_CHn通道发生器就输出高电平，并产生向上比较匹配中断标志CMPUIF。计数器将继续向上计数直到PERIODn (旧值)。然后，计数器会自动继续向下计数，当PWM计数器再次达到CMPn (旧值),产生PIF周期中断标志，PWM0_CHn产生输出翻转信号到低电平，并产生向下比较匹配中断标志CMPDIF。一旦PWM计数器计数到0，将会更新PWM计数器周期寄存器PERIODn (新值)和CMPn (新值)，且设置CNTMODEn = 1.

在中心对齐模式中，如果PINTTYPE (PWM_INTEN[17]) =0的话，PWM周期中断可以在向下计数溢出时提起中断。也就是在PWM周期的start (end)位置，或如果PINTTYPE (PWM_INTEN[17]) =1，向上计数器的值达到PERIODn，也就是PWM周期的中心点。

PWM 频率 = HCLK / ((CLKPSCnm +1)*(时钟分频)) /(PERIOD +1); 这里 nm, 可以是01, 23 或 45 ，它取决于所选的PWM通道。

占空比 = (PERIOD - CMP) / (PERIOD +1)

CMP>= PERIOD: PWM 输出总是 low

CMP<PERIOD: PWM low 宽度 = (CMP +1) * 2 units;PWM high宽度= (PERIOD - CMP) * 2 units

CMP = 0:PWM low宽度= 2 units; PWM high宽度= PERIOD * 2 units

注意: 1. Unit = 一个 PWM时钟周期.

图 6.7-11 ~图 6.7-13和图 6.7-14可以看出中心对齐PWM时序。

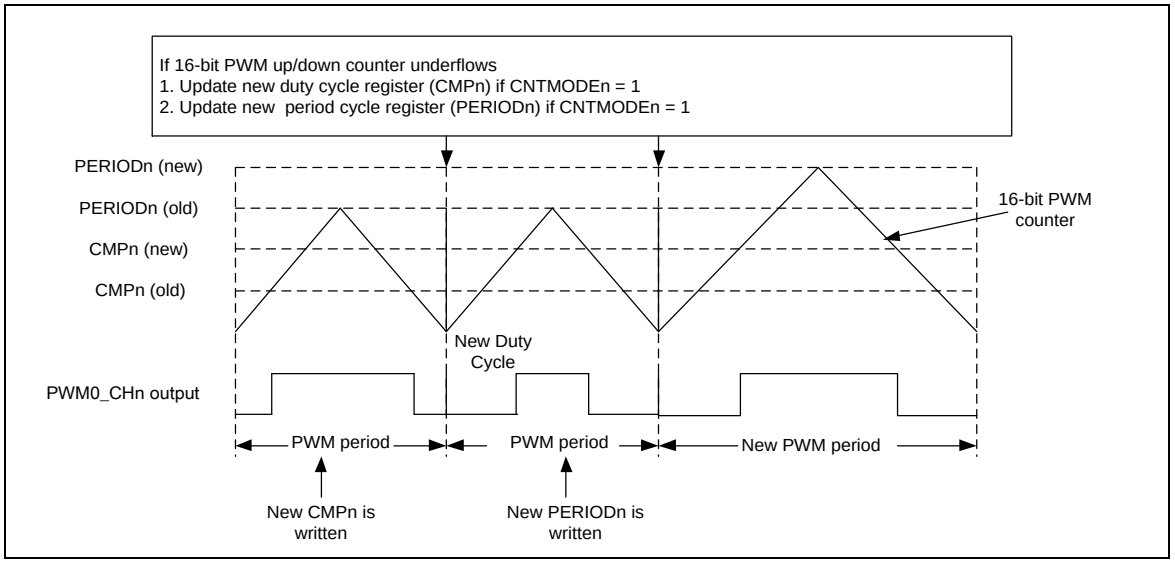


图 6.7-11 中心对齐模式

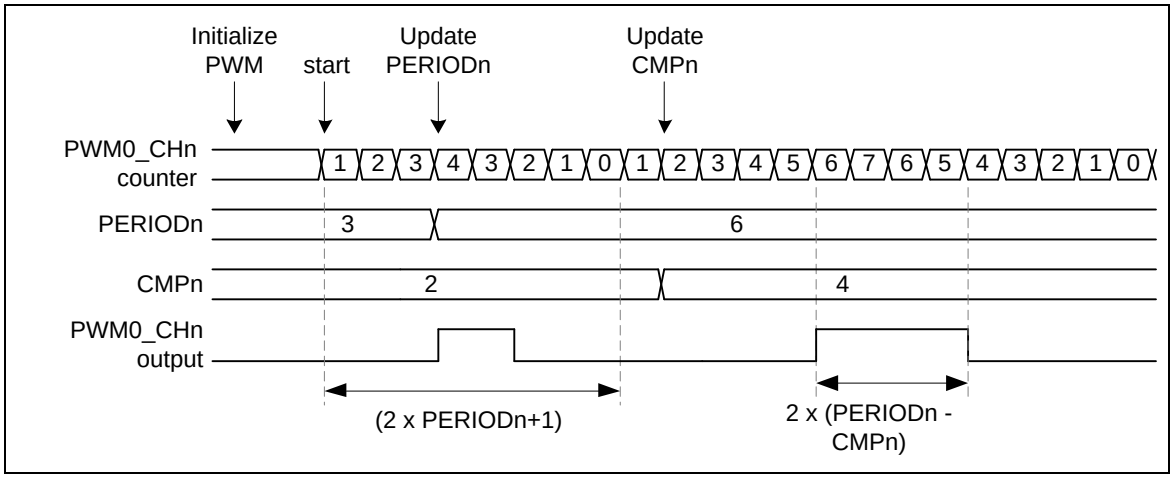


图 6.7-12 中心对齐模式时序图

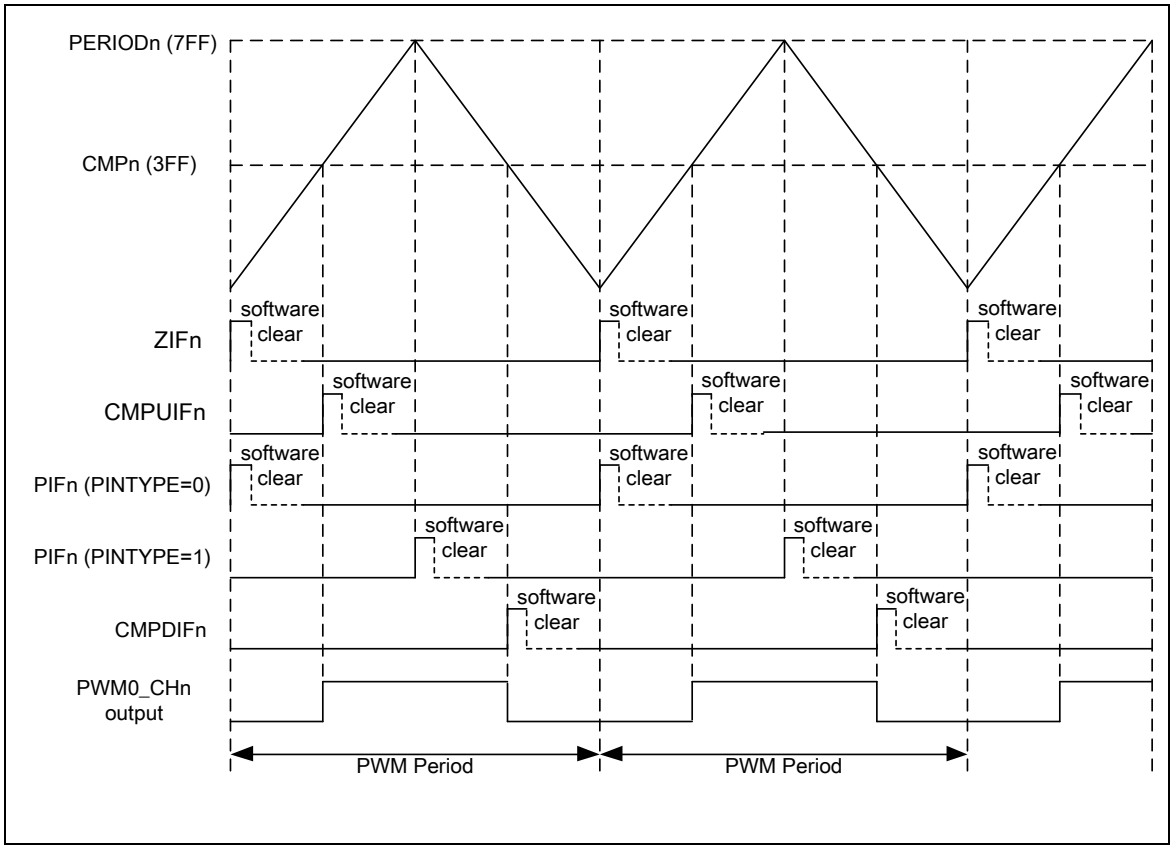


图 6.7-13 中心对齐模式波形图

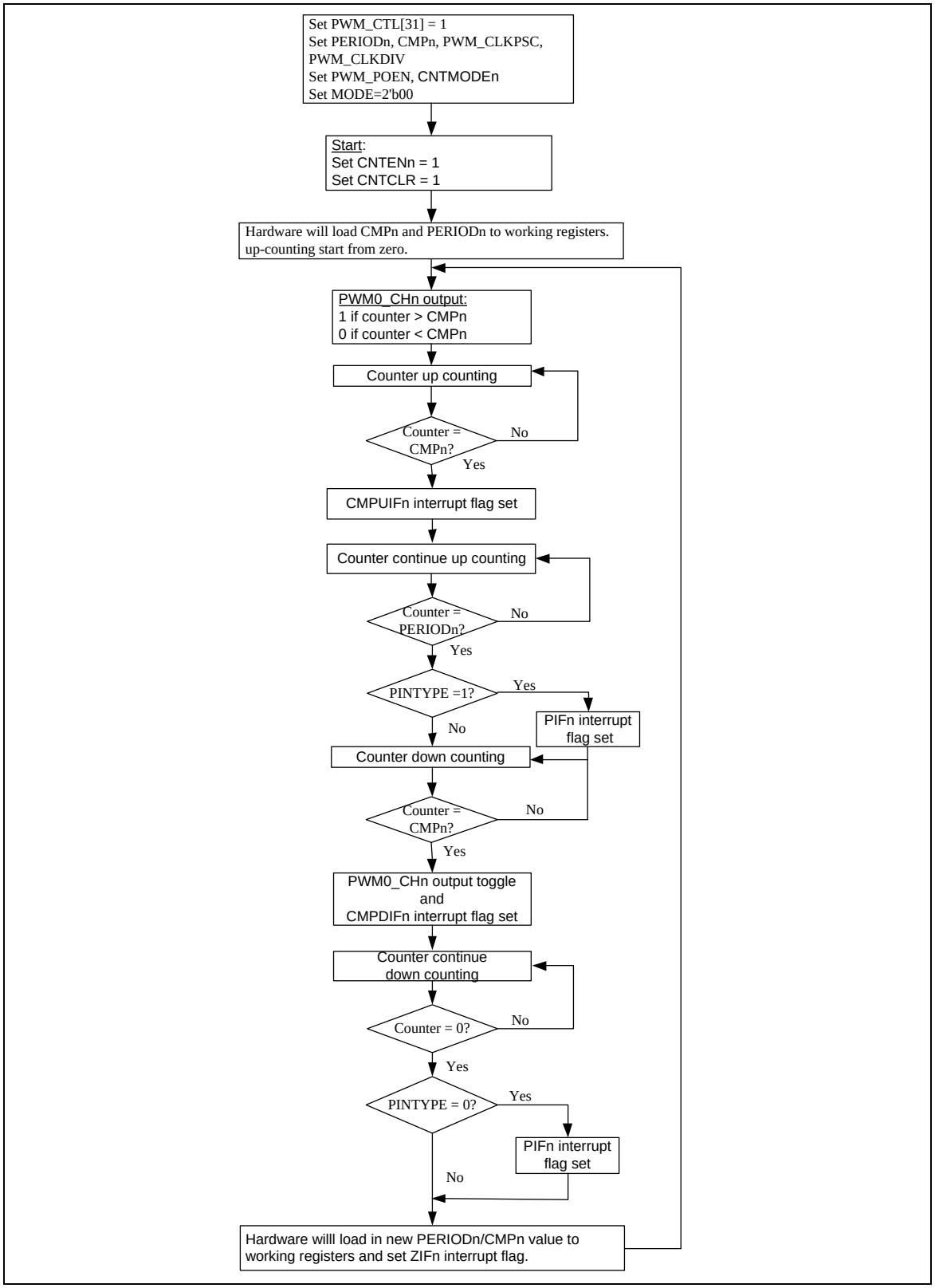


图 6.7-14 中心对齐模式流程图

精准中心对齐PWM模式 (向上/向下计数器)

在向上/向下计数类型当PWM的周期参数被配置，PCAEN (PWM_PCACTL[0])被使能，那么就有精准中心对齐PWM信号输出。PWM计数器将从0重新开始计数到CMPn (旧值)，这时候PWM0_CHn将输出高电平。计数器将继续计数到PERIODn (旧值)的一半。如果PERIODn周期值是奇数，计数器将继续计数超过电平分界点，且使得计数器的值满足PERIODn (旧值)的一半为整数，并继续计数2个时钟周期。计到该处后，计数器将自动向下计数，当PWM计数器的值再次达到CMPn(旧值)，PWM0_CHn通道将变为输出低电平。一旦PWM计数器的值计到0后，将更新周期寄存器PERIODn (新值)、占空比寄存器CMPn (新值)，且设置CNTMODEn = 1.

在精准中心对齐模式， 如果PINTTYPE (PWM_INTEN[17]) =0，也就是在每个PWM周期start (end) 处，或者当PINTTYPE (PWM_INTEN[17]) =1,向上计数到PERIODn，PWM将产生周期中断。

PWM 频率 = HCLK / ((CLKPSCnm +1)*(时钟分频)) /(PERIOD +1); 这里 nm, 可以是01, 23 or 45，它取决于所选的PWM通道。

占空比 = (PERIOD–(CMP+1)*2) / PERIOD

CMP>= PERIOD: PWM 输出总是low

CMP<PERIOD: PWM low 宽度 = (CMP +1) * 2 units;PWM high宽度= (PERIOD–(CMP+1)*2) units

CMP = 0:PWM low宽度= 2 units; PWM high宽度= PERIOD - 2 units

Note: 1. Unit = 一个 PWM 时钟周期

图 6.7-15, 图 6.7-16和图 6.7-17 展示了精确中心对齐模式的时序，流程图。

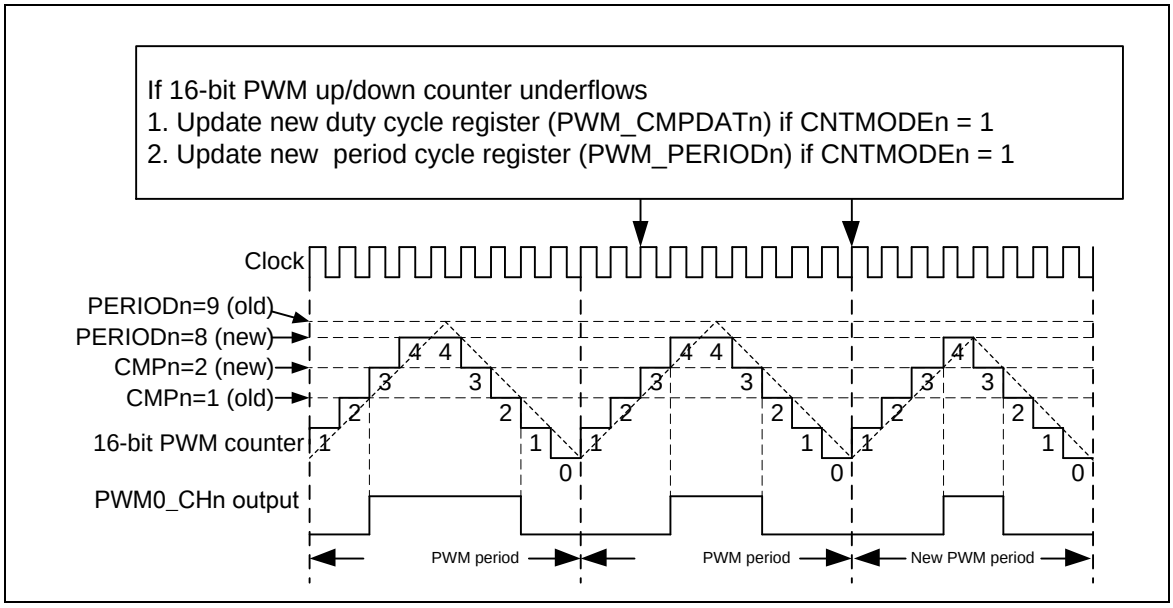


图 6.7-15 精确中心对齐模式

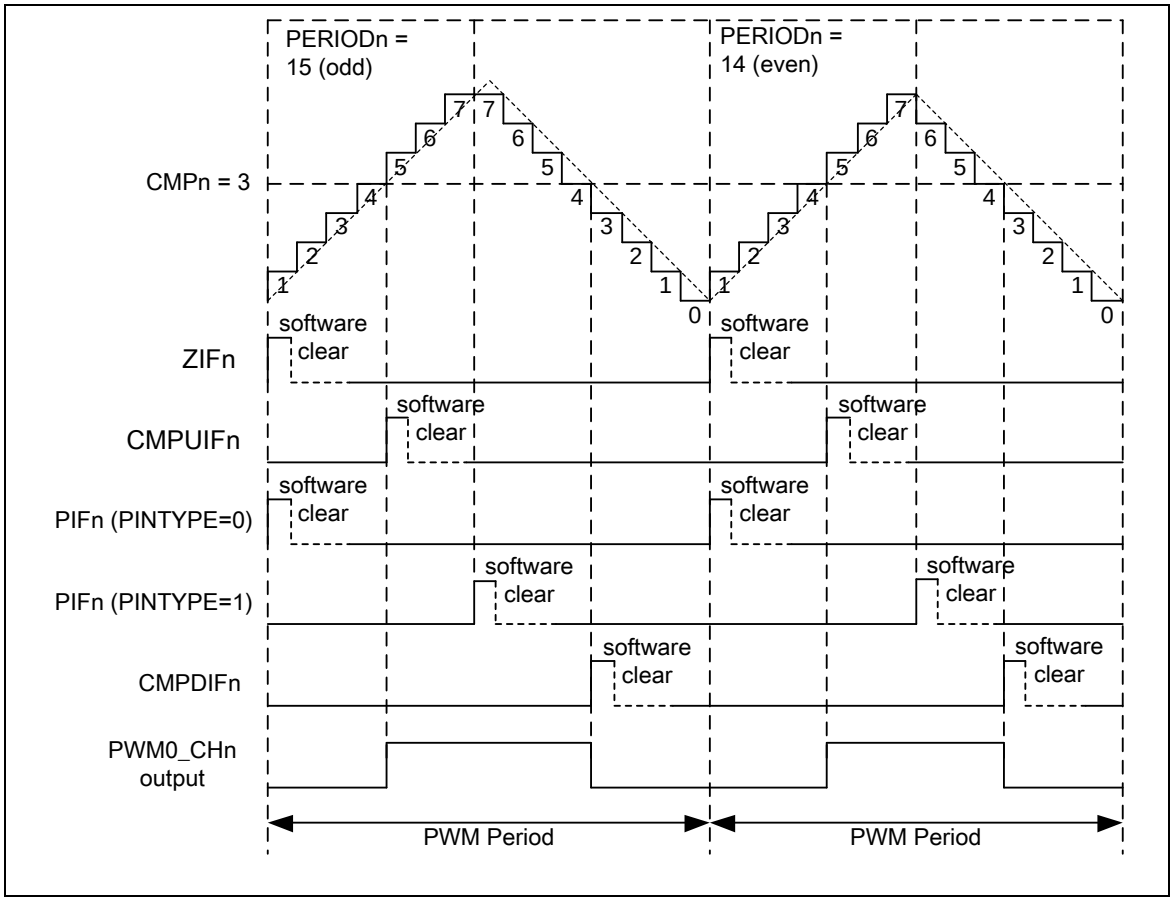


图 6.7-16 精确中心对齐波形图

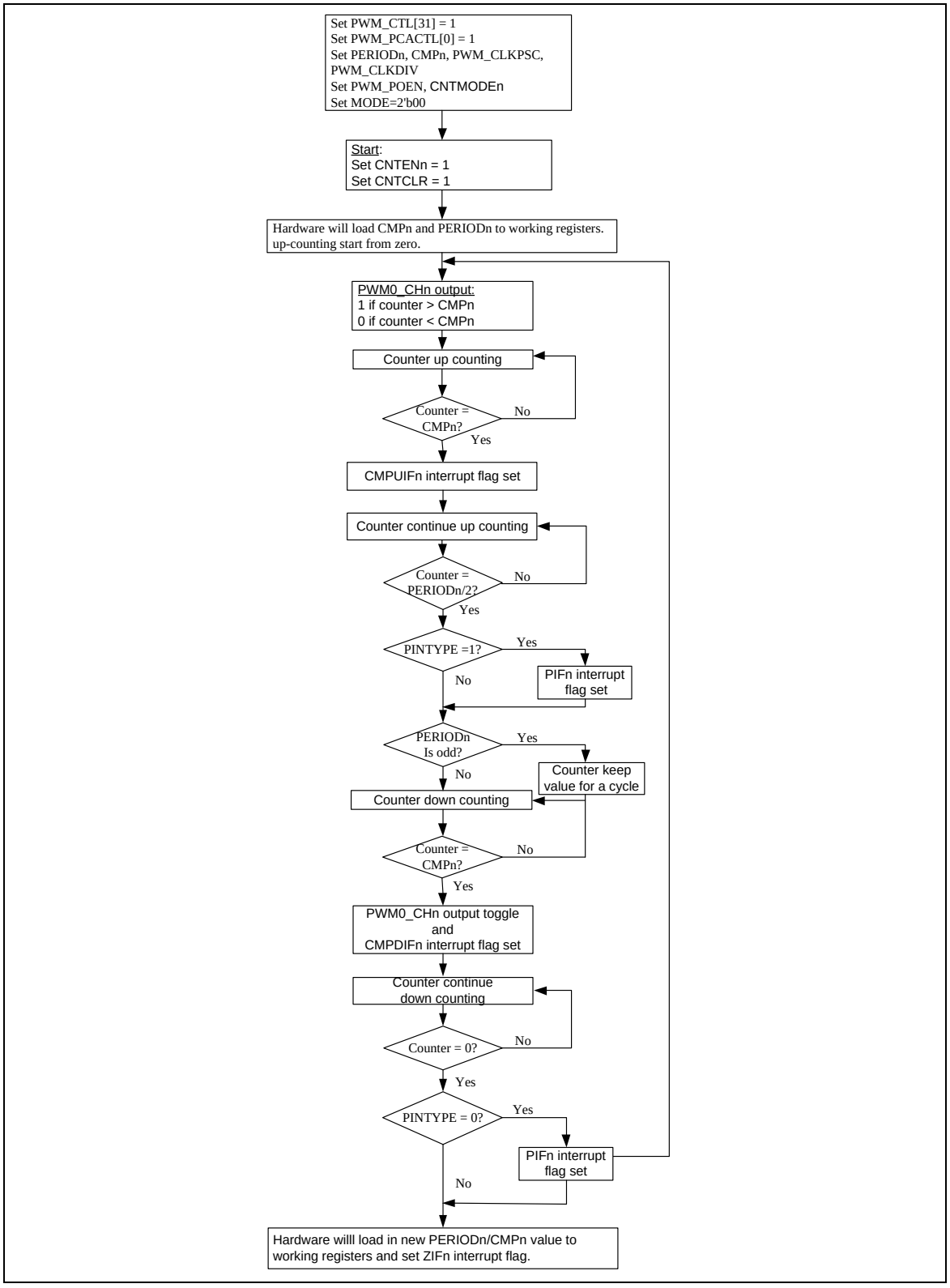


图 6.7-17 精确中心对齐流程图

6.7.5.2 PWM中心加载模式

在中心对齐或者精确中心对齐模式中，PWM支持加载新的PERIODn, CMPn 值。如果在同步模式中，

CMPDn 也支持中心加载操作。当计数器计数到周期中心点，设置HCUPDT (PWM_CTL[5])可以使能这个功能。图 6.7-18展示了一个中心加载模式的例子，计数器计数到第4个中心点，PERIODn 值更新为6。

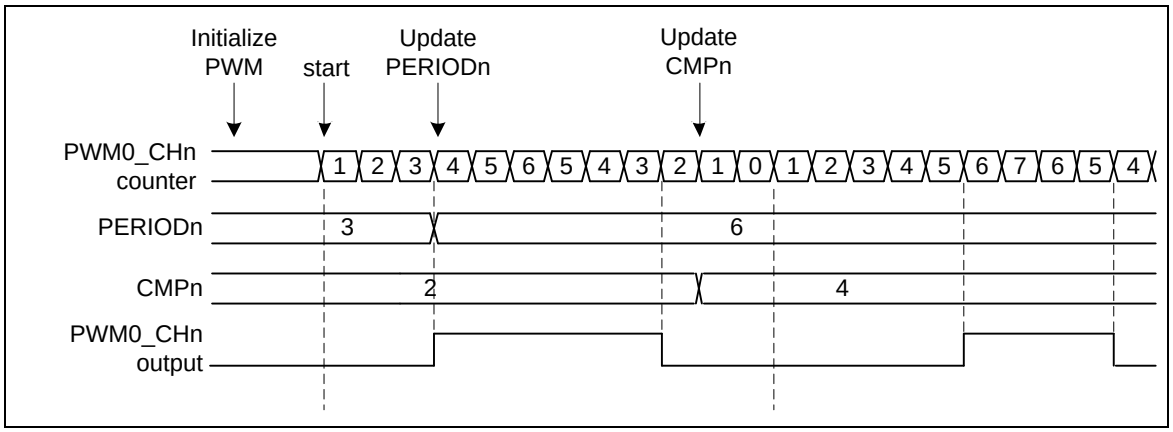


图 6.7-18 PWM中心加载时序

6.7.5.3 *PWM双Buffer，自动加载和单次模式*

NuMicro® Mini58系列微控制器，PWM带有双缓冲功能,重新加载配置值，在下一周期开始更新，不影响当前的计数器操作。这样PWM的设定周期值就被写到PERIODn.

如果CNTMODE0位设置为0，PWM0_CH0将配置成单次模式， 如果CNTMODE0位设置为1，将配置为自动重载模式。建议PWM0_CH0切换模式之前，先将CNTEN0位设置为1,使PWM0_CH0计数器运行起来。因为PWM0_CH0的模式一旦改变，PERIOD0和CMP0的内容将被清除为0，会重置PWM0_CH0周期和占空比设置。如果是PWM0_CH0运行在单次模式,应该先写入CMP0 PERIOD0的值,然后把CNTEN0置为1,使PWM0_CH0计数器开始运行。当PWM0_CH0计数器从PERIOD0值计数到0时,PERIOD0和CMP0将被硬件置0。新的CMP0和PERIOD0值需要由软件写入。重新启动下一个单次模式周期时,应该先写入CMP0值，因为一旦PERIOD0写一个非零值，PWM0_CH0计数器就开始自动重启计数了。PWM0_CH0运行在自动重载模式是,应该先写入CMP0和PERIOD0,然后把CNTEN0置为1,使PWM0_CH0计数器开始运行。PERIOD0的值当PWM0_CH0计数器向下计数达到0后，将重新加载。如果设置为0,PERIOD0 PWM0_CH0计数器将挂起。PWM0_CH1 ~ PWM0_CH5 与PWM0_CH0功能类似。

注意：单次模式只支持边沿对齐模式。

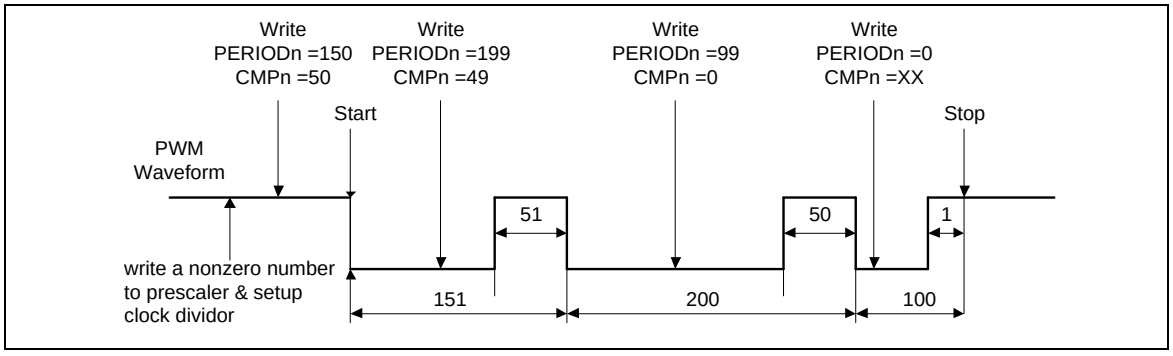


图 6.7-19 PWM双缓存图解

6.7.5.4 *调节占空比*

双缓存功能支持CMPn在当前周期的任何时间写入。当前写入值将在下个周期生效。

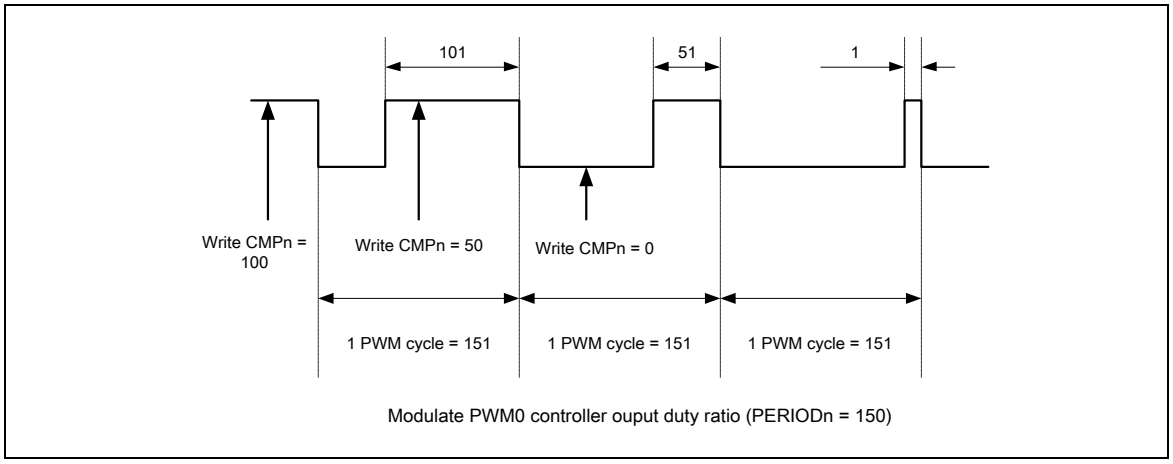


图 6.7-20 PWM控制器输出占空比

6.7.5.5 PWM工作模式

这种强大的PWM单元支持独立模式，可应用于直流或无刷直流电机系统,辅以死区功能可用于交流感应电机和同步电机的应用，同步模式使每组的两个引脚相位同相。此外，组模式可以强制使PWM0_CH2和PWM0_CH4与PWM0_CH0发生器同步，或许可以在直流和无刷直流电机应用及非对称模式（生成非对称PWM波形和中断时序）中简化更新占空比操作。

6.7.5.6 独立模式

当MODE (PWM_CTL[29:28]) = 00的时候，独立模式使能。

默认状况下，PWM工作在独立模式，6个PWM通道输出各自的周期和占空比。

6.7.5.7 互补模式

(PWM_CTL[29:28]) = 01时启用互补模式。

在这个模块下有三个专为互补模式提供的占空比-周期发生器，共有三对PWM输出。6个输出通道根据奇偶分组。互补模式下，奇偶PWM通道PWM0_CHn与相应的偶数PWM通道互补，如PWM0_CH0与PWM0_CH1互补，PWM0_CH3与PWM0_CH2互补，PWM0_CH5与PWM0_CH4互补。PWM的周期参数由其自身的16位计数器提供，它也可作为可选的预分频设置值。

6.7.5.8 插入死区时间

死区发生器在一组互补的PWM中，把一个管脚的波形从关闭状态到打开状态之间的偏移时间叫做“死区时间”。这样能对外接的电源切换设备起到保护作用。一对PWM输出通道有一个8位向下计数器产生死去时间。死区输出维持到计数器向下计数到0。

死区时间按照以下函数计算:

死区时间 = PWM_CLK * (DTInm+1). 这里 nm, 可以是01, 23, 45

下图示一对通道打开死区时间功能时序图。

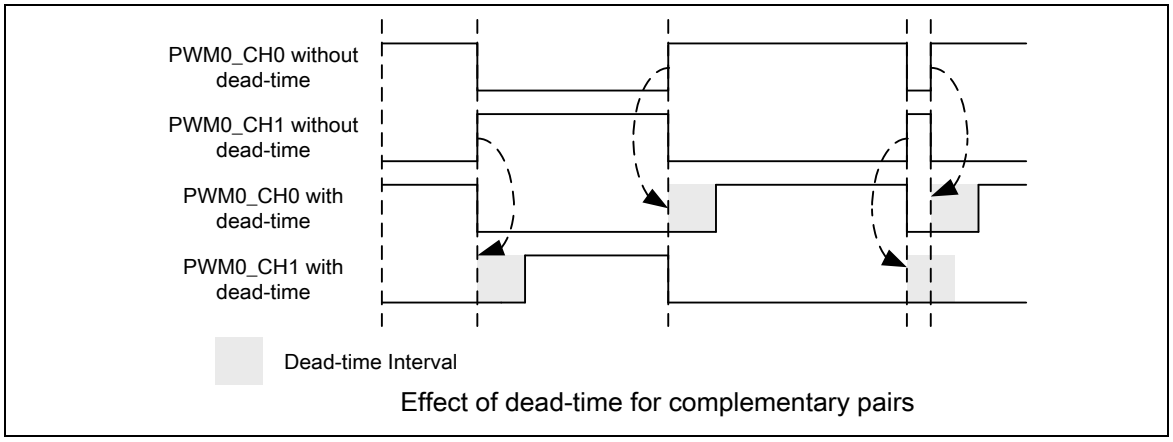


图 6.7-21 死区时间示意图

在电源翻转的应用中，死区时间插入避免上下半桥开关同时打开的情况。所以插入死区时间控制对系统正常工作是至关重要的。通道打开和关闭不能实现立即切换，所以在切换过程中需要留有一些时间，死区功能就恰好实现了这个延时。

6.7.5.9 同步模式

MODE (PWM_CTL[29:28]) = 10的时候同步模式打开。

同步模式的PWM应用于同一相。

PWM0_CH1=PWM0_CH0, PWM0_CH3=PWM0_CH2 and PWM0_CH5=PWM0_CH4.

6.7.5.10 组模式

GROUPEN (PWM_CTL[30]) = 1时，使能组模式

支持组模式控制，此模式中所有PWM通道输出占空比都由PWM0_CH0占空比寄存器控制。

如果GROUPEN = 1，那么(PWM0_CH2, PWM0_CH3) 和 (PWM0_CH4, PWM0_CH5)对会跟随(PWM0_CH0, PWM0_CH1)，意味着：

PWM0_CH4 = PWM0_CH2 = PWM0_CH0;

当MODE (PWM_CTL[29:28]) =01，如果互补模式被使能的话，PWM0_CH5 = PWM0_CH3 = PWM0_CH1 = (PWM0_CH0)的反向波形。

注意：对应用来说，不要同时用组模式和同步模式，这样同步模式实现不了。

6.7.5.11 不对称模式

不对称模式只在中心对齐模式中起作用。ASYMEN (PWM_CTL[21]) = 1时，不对称模式使能。在这个模式中PWM寄存器在向下计数时会与CMPDn (PWM_CMPDATn[31:16])比较。如果CMRDn 不等于CMRn，PWM会产生不对称波形并把相应通道标志CMPDIFn置位。

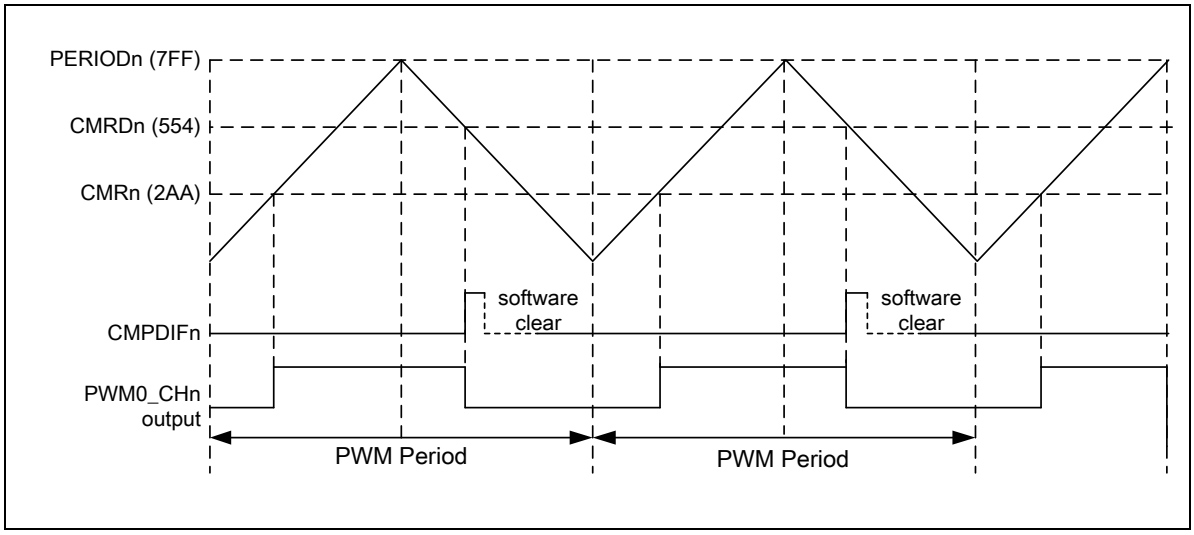


图 6.7-22 不对称模式时序图

6.7.5.12 极性控制

每个PWM通道都有独立的极性控制，默认极性是输出高电平。

图 6.7-23可以看出不同极性的PWM初始化状态

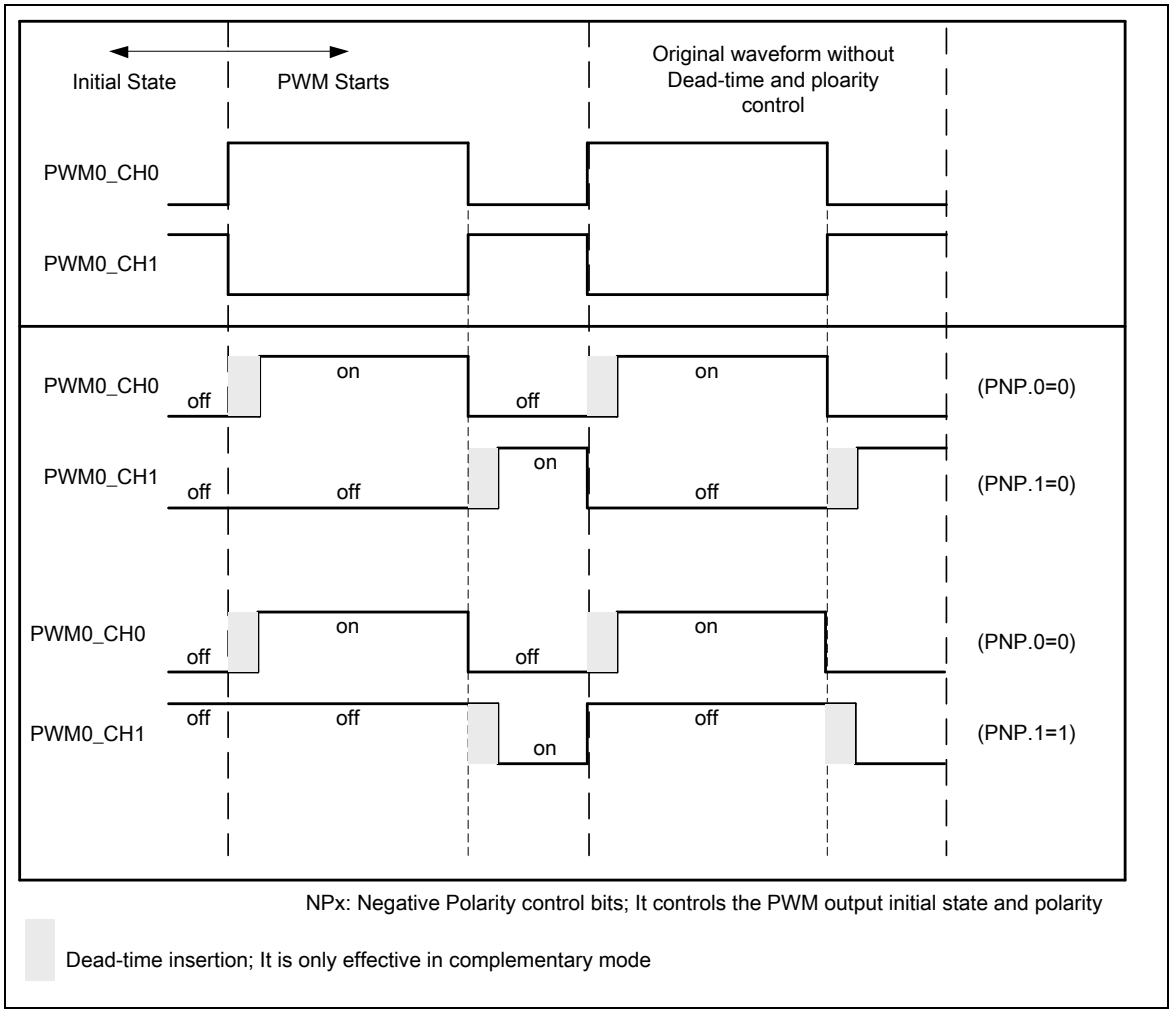


图 6.7-23 上升沿死区PWM极性控制初始化状态

6.7.5.13 PWM电机控制中断架构

PWM单元有七个中断源,ZIFn(PWM_INTSTS[5:0])PWM计数器计数为零时产生的中断标志。CMPUIFn(PWM_INTSTS[29:24])PWM计数器向上计数到CMPn(PWM_CMPDATn[15:0])中断标志。PIFn(PWM_INTSTS[23:18])PWM计数器边沿对齐计数中断标志，中心对齐计数中断标志。CMPDIFn(PWM_INTSTS[13])PWM计数器向下计数到CMPn(PWM_CMPDATn[15:0])中断标志，如果运行在在不对称类型，那么向下计数至CMPDn(PWM_CMPDATn[31:16]);BRKIF0(PWM_INTSTS[16])Brake0中断标志,BRKIF1(PWM_INTSTS[17])Brake1中断标志(BRKIF1)和SWBRK(PWM_BRKCTL[9]) 软件触发刹车中断。

BRKIEN(PWM_INTEN[16])位控制刹车中断使能。ZIENn(PWM_INTEN[5:0])位控制ZIFn中断使能。CMPUIENn(PWM_INTEN[29:24])位控制CMPUIFn中断使能。PIENn(PWM_INTEN[23:18])位控制PIFn中断使能。CMPDIENn(PWM_INTEN[13])位控制CMPDIFn中断使能。请注意,所有的中断标志都由硬件置位，且必须通过软件使能。

图 6.7-24展示了电机控制中PWM中断的架构.

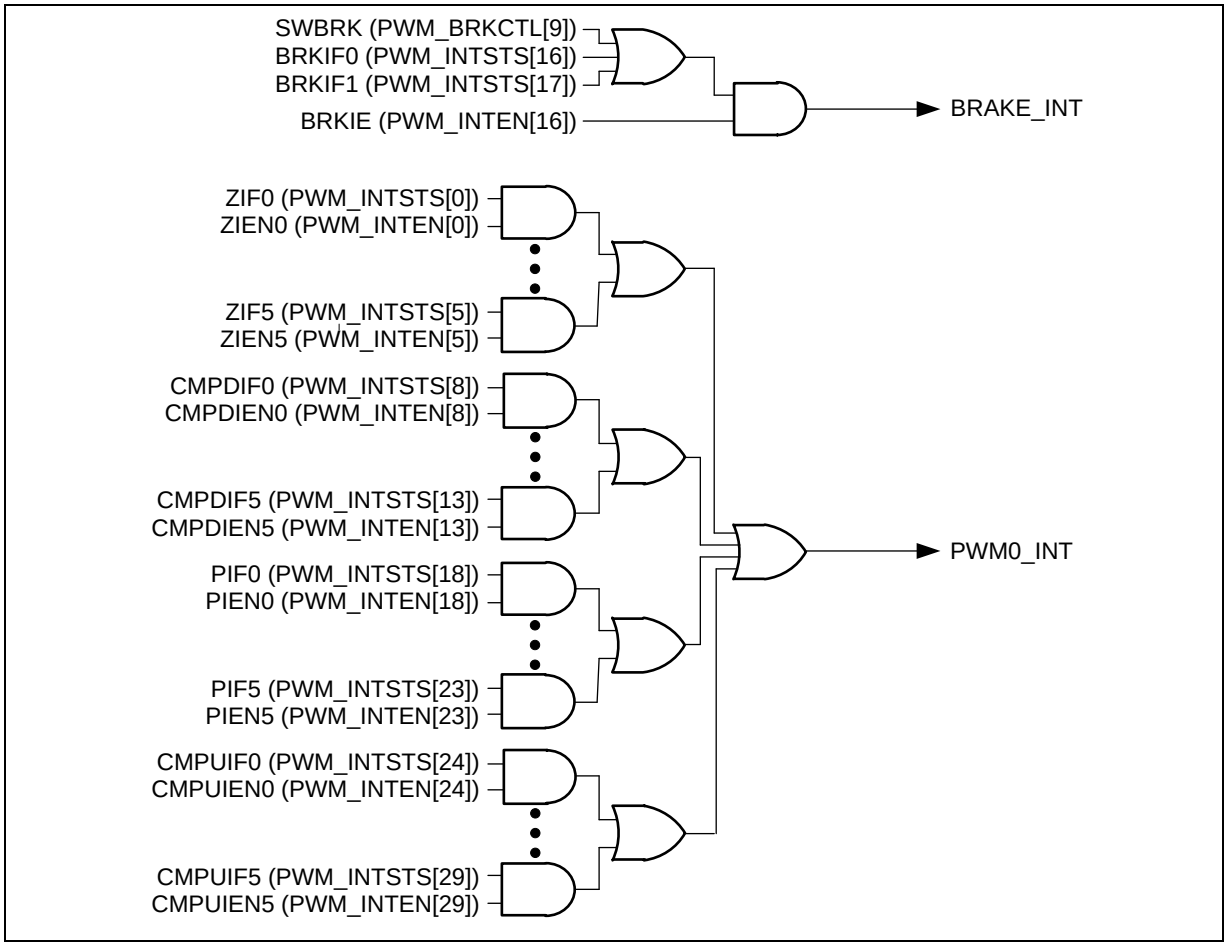


图 6.7-24 电机控制PWM中断架构

6.7.5.14 PWM刹车功能

PWM支持故障刹车功能。BKODn (PWM_BRKCTL[29:24])控制6个通道的刹车阀值，通道7 (GPIO P0.0)和通道6(GPIO P0.1)在刹车状态下分别输出D7BKOD，D7BKOD的值。

支持两路外部刹车管脚BKP0和BKP1，支持软件触发，刹车功能由PWM_BRKCTL寄存器控制。

因为刹车状况重置也会自动产生BRKSTS标志，用户可以使能相关刹车中断位来决定什么情况下产生刹车中断。

注意：当刹车事件发生时，PWM0_CH0 ~ PWM0_CH5使能位会被硬件清除。用户要先清掉刹车事件标志位，重新使能PWM0_CHn相关通道。

6.7.5.15 PWM换相功能

变相功能可以通过用定时器和ACMP触发PWM模块，此时TMR0TEN和TMR1TEN(PWM_PHCHG[30]和PWM_PHCHG[22])寄存器要使能。使用定时器触发PWM,通过配置PWM_PHCHG和PWM_PHCHGNXT寄存器。每次超时事件到来时, PWM_PHCHG的值会被PWM_PHCHG_NXT的值自动更新，每一次PWM_PHCHG更新，相关功能也将改变。它包括实现ACMP触发PWM使能,ACMP0TEN，ACMP1TEN(PWM_PHCHG[31]和PWM_PHCHG[23])。

有两个步骤选择ACMP来源。首先,设置POSCTL[1:0](PWM_PHCHGMSK[9:8])选择ACMP时钟源。如果控

制源是PWM_PHCHG寄存器(POSCTL[1:0)= 11),设置A0POSSEL(PWM_PHCHG[29:28])和A1POSSEL (PWM_PHCHG[21:20])选择ACMP0、ACMP1的时钟源。只要ACMP0或者ACMP1触发PWM，设置和ACMP0相关的AOFFENn0 (PWM_PHCHG[27:24]) 或和ACMP1相关的AOFFENn1 (PWM_PHCHG[19:16])将强制使PWM0_CHn（n表示PWM通道0~3）持续一个周期输出低。这在步进电机中很有用。

除了触发PWM，换相寄存器有相关的屏蔽控制位去改变PWM输出相位。将MSKENn (PWM_PHCHG[13:8])相关寄存器位置零去使能通道屏蔽功能，相关通道会输出MSKDATn (PWM_PHCHG[5:0])值。当MASKEND7和MASKEND6(PWM_PHCHGMSK[7:6])设置为1，MSKDAT7和MSKDAT6(PWM_PHCHG[7:6])是独立的输出数据，输出到GPIO P0.0 P0.1。

6.7.5.16 PWM换相屏蔽对齐功能

PWM支持相位改变屏蔽对齐功能来对齐每个PWM周期的屏蔽数据。通过设置相应通道的ALIGNn (PWM_MSKALIGN[21:16])，PWM通道n将不会立即改变输出电平为MSKDATn (PWM_MSKALIGN[5:0])，而在下一个PWM周期开始时更新。需要注意的是，PWM_MSKALIGN 和 PWM_PHCHG寄存器MSKENn 和 MSKDATn需要在同个寄存器里面配置。

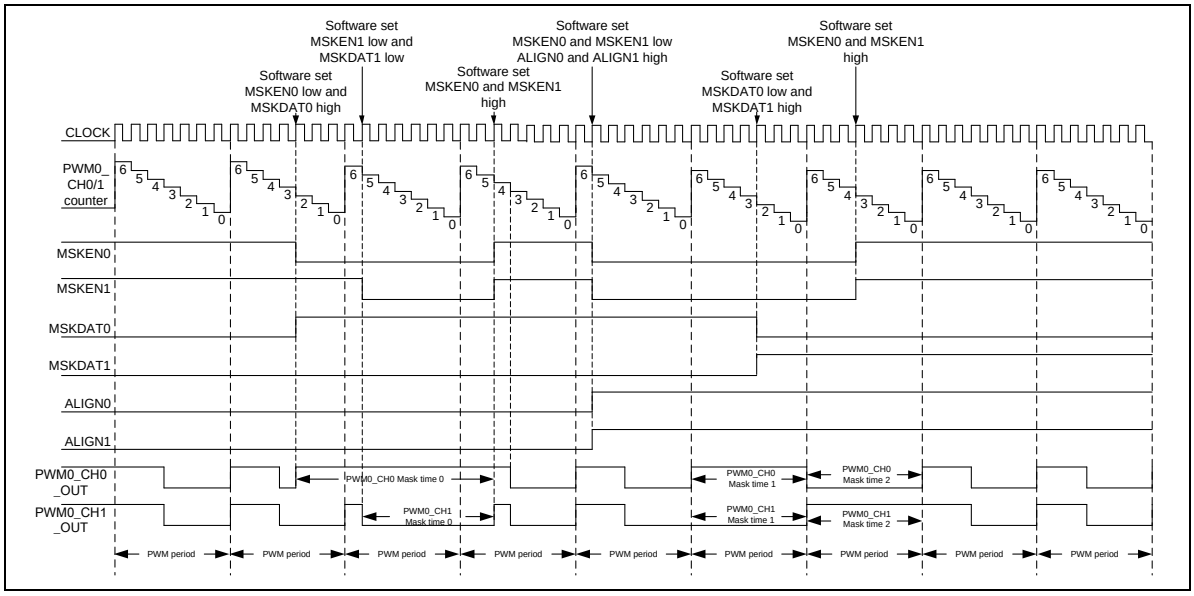


图 6.7-25 屏蔽和屏蔽对齐

6.7.5.17 PWM计数器配置过程

以下是PWM计数器配置过程

1. 配置预分频寄存器(PWM_CLKPSC) 以设置预分频CLKPSCnm.
2. 配置时钟选择寄存器(PWM_CLKDIV)选择时钟源(CLKDIVn).
3. 配置PWM控制寄存器(PWM_CTL)来使能自动加载模式(CNTMODEn = 1)，PWM计数器对齐类型选择(CNTTYPE)和关闭PWM计数器(CNTENn = 0)。
4. 配置PWM控制寄存器(PWM_CTL)选择翻转是否打开(PINVn)，死区时间发生器打开/关闭(DTCNTnm) (可选)
5. 配置PWM_DTCTL寄存器来选择死区时间（可选）。

6. 配置比较寄存器(PWM_CMPDATn) 来设置占空比。
7. 配置PWM计数器寄存器(PWM_PERIODn)，用以设置计数器加载值。
8. 配置PWM中断使能寄存器(PWM_INTEN)来设置PWM周期中断类型(INTTYPE)，PWM零点中断使能位(ZIENn), PWM向上计数匹配中断使能位(CMPUIENn), PWM周期中断使能位(PIENn), PWM向下计数匹配中断使能位(CMPDIENn). (可选)
9. 配置PWM输出使能寄存器(PWM_POEN)使能PWM输出通道
10. 配置PWM控制寄存器(PWM_CTL)使能PWM计数器(CNTENn = 1)

6.7.5.18 PWM 计数器停止过程

模式1:

设置16bit计数器寄存器(PERIODn)到0。当中断请求发生，关闭PWM计数器(PWM_CTL中的CNTENn位).(推荐做法)

模式2:

直接关闭PWM计数器（不推荐）

不推荐的原因是关闭CNTENn，会使PWM输出信号立即停止，会改变PWM输出的占空比，这在电机控制中将会带来灾难性后果。

6.7.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移量	R/W	描述	复位值
PWM0 基地址: **PWM0_BA = 0x4004_0000**				
PWM_CLKPSC	PWM0_BA+0x00	R/W	PWM时钟预分频寄存器	0x0000_0000
PWM_CLKDIV	PWM0_BA+0x04	R/W	PWM 时钟选择寄存器	0x0000_0000
PWM_CTL	PWM0_BA+0x08	R/W	PWM 控制寄存器	0x0000_0000
PWM_PERIOD 0	PWM0_BA+0x0C	R/W	PWM 计数器周期寄存器0	0x0000_0000
PWM_PERIOD 1	PWM0_BA+0x10	R/W	PWM 计数器周期寄存器 1	0x0000_0000
PWM_PERIOD 2	PWM0_BA+0x14	R/W	PWM 计数器周期寄存器2	0x0000_0000
PWM_PERIOD 3	PWM0_BA+0x18	R/W	PWM 计数器周期寄存器 3	0x0000_0000
PWM_PERIOD 4	PWM0_BA+0x1C	R/W	PWM计数器周期寄存器4	0x0000_0000
PWM_PERIOD 5	PWM0_BA+0x20	R/W	PWM 计数器周期寄存器5	0x0000_0000
PWM_CMPDAT 0	PWM0_BA+0x24	R/W	PWM比较寄存器 0	0x0000_0000
PWM_CMPDAT 1	PWM0_BA+0x28	R/W	PWM 比较寄存器1	0x0000_0000
PWM_CMPDAT 2	PWM0_BA+0x2C	R/W	PWM 比较寄存器2	0x0000_0000
PWM_CMPDAT 3	PWM0_BA+0x30	R/W	PWM 比较寄存器3	0x0000_0000
PWM_CMPDAT 4	PWM0_BA+0x34	R/W	PWM 比较寄存器 4	0x0000_0000
PWM_CMPDAT 5	PWM0_BA+0x38	R/W	PWM比较寄存器 5	0x0000_0000
PWM_INTEN	PWM0_BA+0x54	R/W	PWM中断使能寄存器	0x0000_0000
PWM_INTSTS	PWM0_BA+0x58	R/W	PWM中断状态寄存器	0x0000_0000
PWM_POEN	PWM0_BA+0x5C	R/W	PWM 输出使能寄存器	0x0000_0000
PWM_BRKCTL	PWM0_BA+0x60	R/W	PWM故障刹车控制寄存器	0x0000_0000
PWM_DTCTL	PWM0_BA+0x64	R/W	PWM死区控制寄存器	0x0000_0000
PWM_ADCTCTL0	PWM0_BA+0x68	R/W	PWM 触发控制寄存器 0	0x0000_0000

PWM_ADCTCTL1	PWM0_BA+0x6C	R/W	PWM 触发控制寄存器1	0x0000_0000
PWM_ADCTSTS0	PWM0_BA+0x70	R/W	PWM 触发状态寄存器0	0x0000_0000
PWM_ADCTSTS1	PWM0_BA+0x74	R/W	PWM触发状态寄存器 1	0x0000_0000
PWM_PHCHG	PWM0_BA+0x78	R/W	PWM 相位修改寄存器	0x0000_3F00
PWM_PHCHGNXT	PWM0_BA+0x7C	R/W	PWM 下一相位修改寄存器	0x0000_3F00
PWM_PHCHGMSK	PWM0_BA+0x80	R/W	PWM 相位修改屏蔽寄存器	0x0000_0000
PWM_IFA	PWM0_BA+0x84	R/W	PWM周期中断累加控制寄存器	0x0000_00F0
PWM_PCACTL	PWM0_BA+0x88	R/W	PWM精准中心对齐型控制寄存器	0x0000_0000
PWM_MSKALIGN	PWM0_BA+0x8C	R/W	PWM 相位修改屏蔽对齐寄存器	0x0000_3F00

6.7.7 寄存器描述

PWM预分频寄存器(PWM_CLKPSC)

寄存器	偏移地址	R/W	描述	复位值
PWM_CLKPSC	PWM0_BA+0x00	R/W	PWM 时钟预分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CLKPSC45							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CLKPSC23							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC01							

位	描述	
[31:24]	**Reserved**	保留.
[23:16]	**CLKPSC45**	**PWM 计数器 4和5的时钟预分频4** 输入时钟源输入到相应的PWM计数器前被(CLKPSC45 + 1)分频。 如图 CLKPSC45 = 0,时钟预分频 4 输出时钟将被停止。所以相应的PWM计数器将被停止。
[15:8]	**CLKPSC23**	**PWM 计数器 2和3的时钟预分频2** 输入时钟源输入到相应的PWM计数器前被(CLKPSC23 + 1)分频。 如图CLKPSC23 = 0,时钟预分频 2 输出时钟将被停止。所以相应的PWM计数器将被停止。
[7:0]	**CLKPSC01**	**PWM 计数器 0和1的时钟预分频0** 输入时钟源输入到相应的PWM计数器前被(CLKPSC01 + 1)分频。 如图CLKPSC01 = 0,时钟预分频 0 输出时钟将被停止。所以相应的PWM计数器将被停止。

PWM 时钟选择寄存器(PWM_CLKDIV)

寄存器	偏移地址	R/W	描述	复位值
PWM_CLKDIV	PWM0_BA+0x04	R/W	PWM时钟选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**CLKDIV5**			**Reserved**	**CLKDIV4**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**CLKDIV3**			**Reserved**	**CLKDIV2**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CLKDIV1**			**Reserved**	**CLKDIV0**		

位	描述	
[31:23]	**Reserved**	保留.
[22:20]	**CLKDIV5**	**计数器 5时钟分频选择** 选择PWM 计数器的时钟源. 000 = Clock input / (CLKPSC45/2). 001 = Clock input / (CLKPSC45/4). 010 = Clock input / (CLKPSC45/8). 011 = Clock input / (CLKPSC45/16). 100 = Clock input / CLKPSC45. Others = Clock input.
[19]	**Reserved**	保留.
[18:16]	**CLKDIV4**	**计数器 4时钟分频选择** 选择PWM 计数器的时钟源. 000 = Clock input / (CLKPSC45/2). 001 = Clock input / (CLKPSC45/4). 010 = Clock input / (CLKPSC45/8). 011 = Clock input / (CLKPSC45/16). 100 = Clock input / CLKPSC45. Others = Clock input.
[15]	**Reserved**	保留.

位	描述	
[14:12]	**CLKDIV3**	**计数器 3时钟分频选择** 选择PWM 计数器的时钟源. 000 = Clock input / (CLKPSC23/2). 001 = Clock input / (CLKPSC23/4). 010 = Clock input / (CLKPSC23/8). 011 = Clock input / (CLKPSC23/16). 100 = Clock input / CLKPSC23. Others = Clock input.
[11]	**Reserved**	保留.
[10:8]	**CLKDIV2**	**计数器 2时钟分频选择** 选择PWM 计数器的时钟源. 000 = Clock input / (CLKPSC23/2). 001 = Clock input / (CLKPSC23/4). 010 = Clock input / (CLKPSC23/8). 011 = Clock input / (CLKPSC23/16). 100 = Clock input / CLKPSC23. Others = Clock input.
[7]	**Reserved**	保留.
[6:4]	**CLKDIV1**	**计数器 1时钟分频选择** 选择PWM 计数器的时钟源. 000 = Clock input / (CLKPSC01/2). 001 = Clock input / (CLKPSC01/4). 010 = Clock input / (CLKPSC01/8). 011 = Clock input / (CLKPSC01/16). 100 = Clock input / CLKPSC01. Others = Clock input.
[3]	**Reserved**	保留.
[2:0]	**CLKDIV0**	**计数器0时钟分频选择** 选择PWM 计数器的时钟源. 000 = Clock input / (CLKPSC01/2). 001 = Clock input / (CLKPSC01/4). 010 = Clock input / (CLKPSC01/8). 011 = Clock input / (CLKPSC01/16). 100 = Clock input / CLKPSC01. Others = Clock input.

PWM Control Register (PWM_CTL)

寄存器	偏移地址	R/W	描述	复位值
PWM_CTL	PWM0_BA+0x08	R/W	PWM 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CNTTYPE	**GROUPEN**	**MODE**		**CNTCLR**	**DTCNT45**	**DTCNT23**	**DTCNT01**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CNTMODE5	**PINV5**	**ASYMEN**	**CNTEN5**	**CNTMODE4**	**PINV4**	**Reserved**	**CNTEN4**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNTMODE3	**PINV3**	**Reserved**	**CNTEN3**	**CNTMODE2**	**PINV2**	**Reserved**	**CNTEN2**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNTMODE1	**PINV1**	**HCUPDT**	**CNTEN1**	**CNTMODE0**	**PINV0**	**DBGTRIOFF**	**CNTEN0**

位	描述	
[31]	**CNTTYPE**	**PWM 计数器对齐方式选择位** 0 =边沿对齐方式 1 =中心对齐方式
[30]	**GROUPEN**	**成组功能使能位** 0 = 所有 PWM 通道信号时序相互对立 1 = 统一由PWM0_CH0控制的PWM0_CH0, PWM0_CH2 和 PWM0_CH4通道信号时序。统一由PWM0_CH1控制的PWM0_CH1, PWM0_CH3 和 PWM0_CH5通道信号时序。
[29:28]	**MODE**	**PWM 工作模式选择位** 00 = 独立模式. 01 = 互补模式. 10 = 同步模式. 11 = 保留.
[27]	**CNTCLR**	**清除PWM计数器控制位** 0 = 不清PWM计数器 1 = 所有 16-bit PWM 计数器清 0x0000. **注意:** 该位有硬件自动清除
[26]	**DTCNT45**	**死区4 计数器使能位 (PWM0_CH4 和 PWM0_CH5 为 PWMC 组的一对)** 0 = 死区4 发生器禁止. 1 = 死区4 发生器使能. **注意:** 当死区发生器被使能， PWM0_CH4 和 PWM0_CH5 对变成PWMC 组的互补对.
[25]	**DTCNT23**	**死区2 计数器使能位 (PWM0_CH2 和 PWM0_CH3为PWMB 组的一对)** 0 = 死区2发生器禁止. 1 = 死区2 发生器使能. **注意:** 当死区发生器被使能， PWM0_CH2 和 PWM0_CH3 对变成PWMB组的互补对.

位	描述	
[24]	**DTCNT01**	**死区0 计数器使能位 (PWM0_CH0 和 PWM0_CH1为PWMA组的一对)** 0 = 死区0发生器禁止. 1 = 死区0 发生器使能. **注意:** 当死区发生器被使能，PWM0_CH0 和 PWM0_CH1对变成PWMA组的互补对.
[23]	**CNTMODE5**	**PWM 计数器 5自动装载/One-Shot 模式** 0 = One-shot 模式. 1 = 自动装载模式 **注意:**如果该位由0变1，将使PERIOD5 和 CMP5 被清零.
[22]	**PINV5**	**PWM0_CH5 输出反向使能位** 0 = PWM0_CH5 输出反向禁止 1 = PWM0_CH5输出反向使能
[21]	**ASYMEN**	**中心对齐方式下的不对称模式** 0 = 中心对齐方式下的对称模式 1 = 中心对齐方式下的不对称模式.
[20]	**CNTEN5**	**PWM 计数器 5 使能开始运行** 0 = 相应的 PWM计数器停止. 1 = 相应的 PWM计数器开始运行.
[19]	**CNTMODE4**	**PWM 计数器 4 自动装载/One-Shot 模式** 0 = One-shot 模式. 1 = 自动装载模式 **注意:**如果该位由0变1，将使PERIOD4 和 CMP4被清零.
[18]	**PINV4**	**PWM0_CH4输出反向使能位** 0 = PWM0_CH4 输出反向禁止 1 = PWM0_CH4输出反向使能
[17]	**Reserved**	保留.
[16]	**CNTEN4**	**PWM 计数器 4 使能开始运行** 0 = 相应的 PWM计数器停止. 1 = 相应的 PWM计数器开始运行.
[15]	**CNTMODE3**	**PWM 计数器 3自动装载/One-Shot 模式** 0 = One-shot 模式. 1 = 自动装载模式 **注意:**如果该位由0变1，将使PERIOD3 和 CMP3被清零.
[14]	**PINV3**	**PWM0_CH3输出反向使能位** 0 = PWM0_CH3输出反向禁止 1 = PWM0_CH3输出反向使能
[13]	**Reserved**	保留.
[12]	**CNTEN3**	**PWM 计数器 3使能开始运行** 0 = 相应的 PWM计数器停止. 1 = 相应的 PWM计数器开始运行.

位	描述	
[11]	**CNTMODE2**	**PWM 计数器 2自动装载/One-Shot 模式** 0 = One-shot 模式. 1 = 自动装载模式 **注意:**如果该位由0变1，将使PERIOD2 和CMP2被清零.
[10]	**PINV2**	**PWM0_CH2输出反向使能位** 0 = PWM0_CH2输出反向禁止 1 = PWM0_CH2输出反向使能
[9]	**Reserved**	保留.
[8]	**CNTEN2**	**PWM 计数器 2使能开始运行** 0 = 相应的 PWM计数器停止. 1 = 相应的 PWM计数器开始运行.
[7]	**CNTMODE1**	**PWM 计数器 1自动装载/One-Shot 模式** 0 = One-shot 模式. 1 = 自动装载模式 **注意:**如果该位由0变1，将使PERIOD1 和 CMP1被清零.
[6]	**PINV1**	**PWM0_CH1输出反向使能位** 0 = PWM0_CH1输出反向禁止 1 = PWM0_CH1输出反向使能
[5]	**HCUPDT**	**中心对齐方式下半周期更新使能** 0 = 禁止半周期更新PERIOD & CMP. 1 = 使能半周期更新PERIOD & CMP.
[4]	**CNTEN1**	**PWM 计数器 1 使能/禁止运行** 0 = 相应的 PWM 计数器停止运行. 1 = 相应的 PWM 计数器开始运行
[3]	**CNTMODE0**	**PWM计数器 0自动装载/One-Shot 模式** 0 = One-shot 模式. 1 =自动装载模式 **注意:**如果该位由0变1，将使PERIOD0 和 CMP0被清零.
[2]	**PINV0**	**PWM0_CH0输出反向使能位** 0 = PWM0_CH0输出反向禁止 1 = PWM0_CH10输出反向使能
[1]	**DBGTRIOFF**	**禁止 PWM 在Debug 模式下输出三态 (仅在DEBUG模式下)** 0 = 安全模式: 计数器被停止，PWM输出被关闭，反相器处于安全状态。计数器可以重新计数。 1 = 正常模式:在某些特殊情况下，计数器继续正常运行可能非常危险，因为有一个固定占空比的波形输出到反向器 (没有了中断服务程序).
[0]	**CNTEN0**	**PWM 计数器 0 开始运行** 0 = 相应的 PWM 计数器停止运行. 1 = 相应的 PWM 计数器开始运行

PWM 计数器寄存器 0-5 (PWM_PERIOD0-5)

寄存器	偏移地址	R/W	描述	复位值
PWM_PERIOD0	PWM0_BA+0x0C	R/W	PWM 计数器周期寄存器 0	0x0000_0000
PWM_PERIOD1	PWM0_BA+0x10	R/W	PWM计数器周期寄存器1	0x0000_0000
PWM_PERIOD2	PWM0_BA+0x14	R/W	PWM计数器周期寄存器2	0x0000_0000
PWM_PERIOD3	PWM0_BA+0x18	R/W	PWM计数器周期寄存器3	0x0000_0000
PWM_PERIOD4	PWM0_BA+0x1C	R/W	PWM计数器周期寄存器4	0x0000_0000
PWM_PERIOD5	PWM0_BA+0x20	R/W	PWM计数器周期寄存器5	0x0000_0000

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

位	描述	
[31:16]	**Reserved**	保留.
[15:0] n=0,1..5	**PERIODn**	**PWM**计数器周期值 PERIODn 表示 PWM 计数器周期值. 边沿对齐方式: PWM 频率 = HCLK/((prescale+1)*(时钟分频))/(PERIODn+1); 这里 xy, 根据所选的PWM通道可以是01, 23, 45. 占空比= (CMPn+1)/(PERIODn+1). CMPn >= PERIODn: PWM 输出总是high. CMPn < PERIODn: PWM low宽度 = (PERIODn-CMPn) unit; PWM high 宽度 = (CMP+1) unit. CMPn = 0: PWM low 宽度 = (PERIODn) unit; PWM high 宽度 = 1 unit. 中心对齐方式: PWM 频率 = HCLK/((prescale+1)*(时钟分频))/ (2*PERIODn+1); 这里 xy, 根据所选的PWM通道可以是01, 23, 45 占空比 = (PERIODn - CMPn)/(PERIODn+1). CMPn >= PERIODn: PWM 输出总是 low. CMPn < PERIODn: PWM low 宽度 = (CMPn + 1) * 2 unit; PWM high 宽度 = (PERIODn -

位	描述	
		CMPn) * 2 unit. CMPn = 0: PWM low 宽度 = 2 unit; PWM high 宽度 = (PERIODn) * 2 unit. (Unit = One PWM 时钟周期). **注意:** 任何对 PERIODn写操作将会影响到下一个PWM周期.

PWM 比较器寄存器 0-5 (PWM_CMPDAT0-5)

寄存器	偏移地址	R/W	描述	复位值
PWM_CMPDAT0	PWM0_BA+0x24	R/W	PWM 比较器寄存器 0	0x0000_0000
PWM_CMPDAT1	PWM0_BA+0x28	R/W	PWM比较器寄存器1	0x0000_0000
PWM_CMPDAT2	PWM0_BA+0x2C	R/W	PWM 比较器寄存器 2	0x0000_0000
PWM_CMPDAT3	PWM0_BA+0x30	R/W	PWM 比较器寄存器3	0x0000_0000
PWM_CMPDAT4	PWM0_BA+0x34	R/W	PWM比较器寄存器4	0x0000_0000
PWM_CMPDAT5	PWM0_BA+0x38	R/W	PWM 比较器寄存器5	0x0000_0000

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

位	描述	
[31:16] n=0,1..5	**CMPDn**	**不对称模式下向下计数时的PWM**比较器寄存器 CMPn >= PERIODn:向上计数器PWM输出总是low. CMPDn >= PERIODn: 向下计数器PWM输出总是low. 其它: PWM 输出总是高.
[15:0] n=0,1..5	**CMPn**	**PWM 比较器寄存器** CMP 决定了PWM 的占空比. 边沿对齐模式: PWM 频率 = HCLK/((CLKPSCnm+1)*(时钟分频))/(PERIODn+1); 这里 nm, 根据所选的PWM通道可以是01, 23, 45. 占空比 = (CMPn+1)/(PERIODn+1). CMPn >= PERIODn: PWM输出总是high. CMPn < PERIODn: PWM low 宽度 = (PERIODn-CMPn) unit; PWM high 宽度 = (CMP+1) unit. CMPn = 0: PWM low 宽度 = (PERIODn) unit; PWM high 宽度 = 1 unit. 中心对齐方式: PWM 频率 = HCLK/((CLKPSCnm+1)*(时钟分频)) /(2*PERIODn+1); 这里 nm, 根据所选的

位	描述	
		PWM通道可以是01, 23, 45. 占空比 = (PERIODn - CMPn)/(PERIODn+1). CMPn >= PERIODn: PWM 输出总是 low. CMPn < PERIODn: PWM low 宽度 = (CMPn + 1) * 2 unit; PWM high 宽度 = (PERIODn - CMPn) * 2 unit. CMPn = 0: PWM low宽度= 2 unit; PWM high宽度= (PERIODn) * 2 unit. (Unit = 一个 PWM时钟周期). **注意:** 任何对CMPn写操作将会影响到下一个PWM周期

PWM 中断使能寄存器(PWM_INTEN)

寄存器	偏移地址	R/W	描述	复位值
PWM_INTEN	PWM0_BA+0x54	R/W	PWM中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPUIEN5	CMPUIEN4	CMPUIEN3	CMPUIEN2	CMPUIEN1	CMPUIEN0
23	22	21	20	19	18	17	16
PIEN5	PIEN4	PIEN3	PIEN2	PIEN1	PIEN0	PINTTYPE	BRKIEN
15	14	13	12	11	10	9	8
Reserved		CMPDIEN5	CMPDIEN4	CMPDIEN3	CMPDIEN2	CMPDIEN1	CMPDIEN0
7	6	5	4	3	2	1	0
Reserved		ZIEN5	ZIEN4	ZIEN3	ZIEN2	ZIEN1	ZIEN0

位	描述	
[31:30]	**Reserved**	保留.
[29:24] n=0,1..5	**CMPUIENn**	**PWM 向上比较中断使能位** 0 = PWM0_CHn 向上比较中断禁止. 1 = PWM0_CHn 向上比较中断使能.
[23:18] n=0,1..5	**PIENn**	**PWM周期中断使能位** 0 = PWM0_CHn **周期中断禁止** 1 = PWM0_CHn**周期中断使能**.
[17]	**PINTTYPE**	**PWM 中断类型选择** 0 = 如果 PWM 计数器下溢，ZIFn将被置1. 1 = 如果 PWM 计数器的值与 PERIODn 寄存器匹配，ZIFn将被置1. **注意: 当PWM为中心对齐方式时，该位有效**.
[16]	**BRKIEN**	**Fault Brake0 和 Fault Brake1中断使能位** 0 = BRKIF0 和 BRKIF1 触发PWM中断禁止. 1 = BRKIF0 和 BRKIF1触发PWM中断使能.
[15:14]	**Reserved**	保留.
[13:8] n=0,1..5	**CMPDIENn**	**PWM 比较向下中断使能位** 0 = PWM0_CHn 向下比较中断禁止. 1 = PWM0_CHn 向下比较中断使能.
[7:6]	**Reserved**	保留.
[5:0] n=0,1..5	**ZIENn**	**PWM 0点中断使能位** 0 = PWM0_CHn 0点中断禁止. 1 = PWM0_CHn0点中断使能.

PWM中断指示寄存器(PWM_INTSTS)

寄存器	偏移地址	R/W	描述	复位值
PWM_INTSTS	PWM0_BA+0x58	R/W	PWM中断指示寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
23	22	21	20	19	18	17	16
PIF5	PIF4	PIF3	PIF2	PIF1	PIF0	BRKIF1	BRKIF0
15	14	13	12	11	10	9	8
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
7	6	5	4	3	2	1	0
Reserved		ZIF5	ZIF4	ZIF3	ZIF2	ZIF1	ZIF0

位	描述	
[31:30]	**Reserved**	保留.
[29:24] n=0,1..5	**CMPUIFn**	**PWM 向上比较中断标志** 当 PWM0_CHn 计数器向上计数达到CMPn的值后标志被硬件置位. **注意:** 该位可以被软件写1清0.
[23:18] n=0,1..5	**PIFn**	**PWM周期中断标志** 当 PWM0_CHn计数器的值达到PERIODn的值后标志被硬件置位. **注意:** 该位可以被软件写1清0.
[17]	**BRKIF1**	**PWM Brake1 标志** 0 = BKP1管脚产生一个下降沿信号时，PWM 刹车不会产生刹车标志. 1 = 当 PWM 刹车在 BKP1管脚检测到一个下降沿信号，该标志将会被置1. **注意:** 该位可以被软件写1清0.
[16]	**BRKIF0**	**PWM Brake0 标志** 0 = BKP0管脚产生一个下降沿信号时，PWM 刹车不会产生刹车标志. 1 = 当 PWM 刹车在BKP0管脚检测到一个下降沿信号，该标志将会被置1. **注意:** 该位可以被软件写1清0.
[15:14]	**Reserved**	保留.
[13:8] n=0,1..5	**CMPDIFn**	**PWM 向下比较中断标志** 当 PWM0_CHn 计数器向上计数达到CMPn的值后标志被硬件置位. **注意:** 该位可以被软件写1清0.
[7:6]	**Reserved**	保留.
[5:0] n=0,1..5	**ZIFn**	**PWM 0点中断标志** 当 PWMn 计数器向下计数到0点时标志被硬件置1. **注意:** 该位可以被软件写1清0.

注意: 用户可以通过对PWM_INTSTS 相应位写1清中断标志.

PWM 输出控制寄存器 (PWM_POEN)

寄存器	偏移地址	R/W	描述	复位值
PWM_POEN	PWM0_BA+0x5C	R/W	PWM 输出使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**POEN5**	**POEN4**	**POEN3**	**POEN2**	**POEN1**	**POEN0**

位	描述	
[31:6]	**Reserved**	保留.
[5:0] n=0,1..5	**POENn**	**PWM 输出使能位** 0 = PWM 通道 n 输出禁止. 1 = PWM 通道 n 输出使能. **注意:**相应的GPIO 管脚必须切换到PWM功能

PWM 故障刹车控制寄存器(PWM_BRKCTL)

寄存器	偏移地址	R/W	描述	复位值
PWM_BRKCTL	PWM0_BA+0x60	R/W	PWM故障刹车可控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
D7BKOD	D6BKOD	BKODD5	BKOD4	BKOD3	BKOD2	BKOD1	BKOD0
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SWBRK	BRKACT
7	6	5	4	3	2	1	0
BRKSTS	Reserved			BRK1SEL	BRK0SEL	BRK1EN	BRK0EN

位	描述	
[31]	**D7BKOD**	**通道 7 刹车输出电平选择位** 0 = 当故障刹车产生后，通道 7 输出low 1 = 当故障刹车产生后，通道 7 输出high.
[30]	**D6BKOD**	**通道 6刹车输出电平选择位** 0 = 当故障刹车产生后，通道 6输出low 1 = 当故障刹车产生后，通道6 输出high.
[29:24] n=0,1..5	**BKODn**	**PWM 刹车输出电平选择位** 0 = 当故障刹车产生后，PWM 通道 n输出low 1 = 当故障刹车产生后，PWM 通道 n输出high.
[23:10]	**Reserved**	保留.
[9]	**SWBRK**	**软件刹车** 0 = 禁止PWM软件刹车并回到正常PWM功能。 1 = 立即产生PWM刹车.
[8]	**BRKACT**	**PWM刹车动作方式** 0 = 当刹车产生后，PWM 计数器停止. 1 = 当刹车产生后，PWM 计数器保持计数
[7]	**BRKSTS**	**PWM 故障刹车事件状态标志** 0 = 当故障刹车产生后，PWM输出初始状态。 1 = 当故障刹车产生后，PWM输出故障刹车状态. **注意:** 该位可以被软件写1清0，且必须在PWM计数器启动前清零.
[6:4]	**Reserved**	保留.

位	描述	
[3]	**BRK1SEL**	**BKP0 故障刹车功能源选择位** 0 = EINT0 作为 BKP0的一个刹车源. 1 = CPO1 作为 BKP0的一个刹车源.
[2]	**BRK0SEL**	**BKP1故障刹车功能源选择位** 0 = EINT1作为BKP1的一个刹车源. 1 = CPO0作为BKP1的一个刹车源.
[1]	**BRK1EN**	**使能 BKP1 管脚触发故障刹车功能1** 0 = 禁止 BKP1 管脚触发刹车功能 1 (EINT1 或 CPO0). 1 = 使能 BKP1 管脚的下降沿可以触发刹车功能1.
[0]	**BRK0EN**	**使能BKP0管脚触发故障刹车功能0** 0 = 禁止BKP0管脚触发刹车功能 0(EINT0 或 CPO1) 1 = 使能BKP0管脚的下降沿可以触发刹车功能0.

PWM 死区长度寄存器(PWM_DTCTL)

寄存器	偏移地址	R/W	描述	复位值
PWM_DTCTL	PWM0_BA+0x64	R/W	PWM 死区长度控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DTI45							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DTI23							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DTI01							

位	描述	
[31:24]	**Reserved**	保留.
[23:16]	**DTI45**	**通道4 和通道5 对(PWM0_CH4 和 PWM0_CH5 对)的死区长度寄存器** 该 8 位决定了死区长度. 死区长度的单位时间是由PWM_CLKDIV 位所决定.
[15:8]	**DTI23**	**通道2 和通道3 对(PWM0_CH2 和 PWM0_CH3 对)的死区长度寄存器** 该 8 位决定了死区长度. 死区长度的单位时间是由PWM_CLKDIV 位所决定.
[7:0]	**DTI01**	**通道0 和通道1 对(PWM0_CH0和 PWM0_CH1 对)的死区长度寄存器** 该 8 位决定了死区长度. 死区长度的单位时间是由PWM_CLKDIV 位所决定.

PWM 触发ADC控制寄存器(PWM_ADCTCTL0)

寄存器	偏移地址	R/W	描述	复位值
PWM_ADCTCTL0	PWM0_BA+0x68	R/W	PWM触发控制寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**ZPTRGEN3**	**CDTRGEN3**	**CPTRGEN3**	**CUTRGEN3**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**ZPTRGEN2**	**CDTRGEN2**	**CPTRGEN2**	**CUTRGEN2**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ZPTRGEN1**	**CDTRGEN1**	**CPTRGEN1**	**CUTRGEN1**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**ZPTRGEN0**	**CDTRGEN0**	**CPTRGEN0**	**CUTRGEN0**

位	描述	
[31:28]	**Reserved**	保留.
[27]	**ZPTRGEN3**	**通道3 在0点触发ADC使能位** 当通道3的计数值等于0时使能PWM触发ADC功能。 0 = PWM条件触发ADC功能禁止. 1 = PWM条件触发ADC功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[26]	**CDTRGEN3**	**通道 3 向下比较触发 ADC使能位** 当在向下计数方向时，通道3的计数值与CMP3 相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[25]	**CPTRGEN3**	**通道3 中心点触发 ADC使能位** 当通道3的计数值与PERIOD3相等时使能 PWM 触发ADC功能 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. 注意**:** 该位仅在PWM中心对齐模式下有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.
[24]	**CUTRGEN3**	**通道 3 向上比较触发 ADC使能位** 当在向上计数方向时，通道3的计数值与CMP3相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.
[23:20]	**Reserved**	保留.

位	描述	
[19]	**ZPTRGEN2**	**通道2 在0点触发ADC使能位** 当通道2的计数值等于0时使能PWM触发ADC功能。 0 = PWM条件触发ADC功能禁止. 1 = PWM条件触发ADC功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[18]	**CDTRGEN2**	**通道2 向下比较触发 ADC使能位** 当在向下计数方向时，通道2的计数值与CMP2 相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[17]	**CPTRGEN2**	**通道2 中心点触发 ADC使能位** 当通道2的计数值与PERIOD2相等时使能 PWM 触发ADC功能 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. 注意**:** 该位仅在PWM中心对齐模式下有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.
[16]	**CUTRGEN2**	**通道2 向上比较触发 ADC使能位** 当在向上计数方向时，通道2的计数值与CMP2相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.
[15:12]	**Reserved**	保留.
[11]	**ZPTRGEN1**	**通道1 在0点触发ADC使能位** 当通道1的计数值等于0时使能PWM触发ADC功能。 0 = PWM条件触发ADC功能禁止. 1 = PWM条件触发ADC功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[10]	**CDTRGEN1**	**通道1 向下比较触发 ADC使能位** 当在向下计数方向时，通道1的计数值与CMP1 相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[9]	**CPTRGEN1**	**通道1 中心点触发 ADC使能位** 当通道1的计数值与PERIOD1相等时使能 PWM 触发ADC功能 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. 注意**:** 该位仅在PWM中心对齐模式下有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.

位	描述	
[8]	**CUTRGEN1**	**通道1 向上比较触发 ADC使能位** 当在向上计数方向时，通道1的计数值与CMP1相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.
[7:4]	**Reserved**	保留.
[3]	**ZPTRGEN0**	**通道0 在0点触发ADC使能位** 当通道0的计数值等于0时使能PWM触发ADC功能。 0 = PWM条件触发ADC功能禁止. 1 = PWM条件触发ADC功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[2]	**CDTRGEN0**	**通道0 向下比较触发 ADC使能位** 当在向下计数方向时，通道0的计数值与CMP0 相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[1]	**CPTRGEN0**	**通道0中心点触发 ADC使能位** 当通道0的计数值与PERIOD0相等时使能 PWM 触发ADC功能 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. 注意**:** 该位仅在PWM中心对齐模式下有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.
[0]	**CUTRGEN0**	**通道0 向上比较触发 ADC使能位** 当在向上计数方向时，通道0的计数值与CMP0相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.

PWM 触发ADC控制寄存器(PWM_ADCTCTL1)

寄存器	偏移地址	R/W	描述	复位值
PWM_ADCTCTL1	PWM0_BA+0x6C	R/W	PWM触发控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ZPTRGEN5**	**CDTRGEN5**	**CPTRGEN5**	**CUTRGEN5**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**ZPTRGEN4**	**CDTRGEN4**	**CPTRGEN4**	**CUTRGEN4**

位	描述	
[31:12]	**Reserved**	保留.
[11]	**ZPTRGEN5**	**通道5在0点触发ADC使能位** 当通道5的计数值等于0时使能PWM触发ADC功能。 0 = PWM条件触发ADC功能禁止. 1 = PWM条件触发ADC功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[10]	**CDTRGEN5**	**通道5向下比较触发 ADC使能位** 当在向下计数方向时，通道5的计数值与CMP5 相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[9]	**CPTRGEN5**	**通道5中心点触发 ADC使能位** 当通道5的计数值与PERIOD5相等时使能 PWM 触发ADC功能 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. 注意**:** 该位仅在PWM中心对齐模式下有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.
[8]	**CUTRGEN5**	**通道5 向上比较触发 ADC使能位** 当在向上计数方向时，通道5的计数值与CMP5相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.
[7:4]	**Reserved**	保留.

位	描述	
[3]	**ZPTRGEN4**	**通道4在0点触发ADC使能位** 当通道4的计数值等于0时使能PWM触发ADC功能。 0 = PWM条件触发ADC功能禁止. 1 = PWM条件触发ADC功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[2]	**CDTRGEN4**	**通道4向下比较触发 ADC使能位** 当在向下计数方向时，通道4的计数值与CMP4相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效
[1]	**CPTRGEN4**	**通道4中心点触发 ADC使能位** 当通道4的计数值与PERIOD4相等时使能 PWM 触发ADC功能 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. 注意**:** 该位仅在PWM中心对齐模式下有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.
[0]	**CUTRGEN4**	**通道4 向上比较触发 ADC使能位** 当在向上计数方向时，通道4的计数值与CMP4相等时使能PWM触发ADC功能。 0 = PWM 触发 ADC 功能禁止. 1 = PWM触发 ADC 功能使能. **注意:** 该位在中心对齐模式和边沿对齐模式下均有效。当PWM 在边沿对齐模式下, 设置该位无意义也无影响.

PWM 触发状态寄存器(PWM_ADCTSTS0)

寄存器	偏移地址	R/W	描述	复位值
PWM_ADCTSTS0	PWM0_BA+0x70	R/W	PWM 触发状态寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				ZPTRGF3	CDTRGF3	CPTRGF3	CUTRGF3
23	22	21	20	19	18	17	16
Reserved				ZPTRGF2	CDTRGF2	CPTRGF2	CUTRGF2
15	14	13	12	11	10	9	8
Reserved				ZPTRGF1	CDTRGF1	CPTRGF1	CUTRGF1
7	6	5	4	3	2	1	0
Reserved				ZPTRGF0	CDTRGF0	CPTRGF0	CUTRGF0

位	描述	
[31:28]	**Reserved**	保留.
[27]	**ZPTRGF3**	**通道3 在0点触发ADC标志** 当通道3的计数器计数到0点, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[26]	**CDTRGF3**	**通道3 向下比较触发ADC标志** 当通道3的计数器计数到CMP3, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[25]	**CPTRGF3**	**通道3中心点触发ADC 标志** 当通道3的计数值与PERIOD3相等时, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[24]	**CUTRGF3**	**通道3向上比较触发ADC标志** 当通道3的计数值向上比较到CMP3时，该位将被置位来触发ADC.. **注意:** 该位可以被软件写1清0.
[23:20]	**Reserved**	保留.
[19]	**ZPTRGF2**	**通道2 在0点触发ADC标志** 当通道3的计数器计数到0点, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[18]	**CDTRGF2**	**通道2 向下比较触发ADC标志** 当通道2的计数器计数到CMP2, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.

位	描述	
[17]	**CPTRGF2**	**通道2中心点触发ADC 标志** 当通道2的计数值与PERIOD2相等时, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[16]	**CUTRGF2**	**通道2向上比较触发ADC标志** 当通道2的计数值向上比较到CMP2时，该位将被置位来触发ADC.. **注意:** 该位可以被软件写1清0.
[15:12]	**Reserved**	保留.
[11]	**ZPTRGF1**	**通道1 在0点触发ADC标志** 当通道1的计数器计数到0点, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[10]	**CDTRGF1**	**通道1向下比较触发ADC标志** 当通道1的计数器计数到CMP1, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[9]	**CPTRGF1**	**通道1中心点触发ADC 标志** 当通道1的计数值与PERIOD1相等时, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[8]	**CUTRGF1**	**通道1向上比较触发ADC标志** 当通道1的计数值向上比较到CMP1时，该位将被置位来触发ADC.. **注意:** 该位可以被软件写1清0.
[7:4]	**Reserved**	保留.
[3]	**ZPTRGF0**	**通道0 在0点触发ADC标志** 当通道0的计数器计数到0点, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[2]	**CDTRGF0**	**通道0向下比较触发ADC标志** 当通道0的计数器计数到CMP0, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[1]	**CPTRGF0**	**通道0中心点触发ADC 标志** 当通道0的计数值与PERIOD0相等时, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[0]	**CUTRGF0**	**通道0向上比较触发ADC标志** 当通道0的计数值向上比较到CMP0时，该位将被置位来触发ADC.. **注意:** 该位可以被软件写1清0.

PWM 触发状态寄存器 (PWM_ADCTSTS1)

寄存器	偏移地址	R/W	描述	复位值
PWM_ADCTSTS1	PWM0_BA+0x74	R/W	PWM触发状态寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ZPTRGF5**	**CDTRGF5**	**CPTRGF5**	**CUTRGF5**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**ZPTRGF4**	**CDTRGF4**	**CPTRGF4**	**CUTRGF4**

位	描述	
[31:12]	**Reserved**	保留.
[11]	**ZPTRGF5**	**通道5 在0点触发ADC标志** 当通道5的计数器计数到0点, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[10]	**CDTRGF5**	**通道5向下比较触发ADC标志** 当通道5的计数器计数到CMP5, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[9]	**CPTRGF5**	**通道5中心点触发ADC 标志** 当通道5的计数值与PERIOD5相等时, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[8]	**CUTRGF5**	**通道5向上比较触发ADC标志** 当通道5的计数值向上比较到CMP5时，该位将被置位来触发ADC.. **注意:** 该位可以被软件写1清0.
[7:4]	**Reserved**	保留.
[3]	**ZPTRGF4**	**通道4 在0点触发ADC标志** 当通道4的计数器计数到0点, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[2]	**CDTRGF4**	**通道4向下比较触发ADC标志** 当通道4的计数器计数到CMP4, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.
[1]	**CPTRGF4**	**通道4中心点触发ADC 标志** 当通道4的计数值与PERIOD4相等时, 该位将被置位来触发ADC. **注意:** 该位可以被软件写1清0.

位	描述	
[0]	**CUTRGF4**	**通道4向上比较触发ADC标志** 当通道4的计数值向上比较到CMP4时，该位将被置位来触发ADC.. **注意:** 该位可以被软件写1清0.

相位修改寄存器 (PWM_PHCHG)

寄存器	偏移地址	R/W	描述	复位值
PWM_PHCHG	PWM0_BA+0x78	R/W	PWM 相位修改寄存器	0x0000_3F00

31	30	29	28	27	26	25	24
ACMP0TEN	TMR0TEN	A0POSSEL		AOFFEN30	AOFFEN20	AOFFEN10	AOFFEN00
23	22	21	20	19	18	17	16
ACMP1TEN	TMR1TEN	A1POSSEL		AOFFEN31	AOFFEN21	AOFFEN11	AOFFEN01
15	14	13	12	11	10	9	8
AUTOCLR1	AUTOCLR0	MSKEN5	MSKEN4	MSKEN3	MSKEN2	MSKEN1	MSKEN0
7	6	5	4	3	2	1	0
MSKDAT7	MSKDAT6	MSKDAT5	MSKDAT4	MSKDAT3	MSKDAT2	MSKDAT1	MSKDAT0

位	描述	
[31]	**ACMP0TEN**	**ACMP0 触发PWM使能位** 0 = ACMP0触发PWM禁止. 1 = ACMP0触发PWM使能. **注意:** 如果 AUTOCLR0被置位，当 ACMP0 触发 PWM时该位被自动清零.
[30]	**TMR0TEN**	**TIMER0 触发PWM 使能位** 当该位被置1, 当TIMER0溢出时, PHCHG_NXT寄存器的内容将更新到PWM_PHCHG 0 = TIMER0触发PWM禁止. 1 = TIMER0 触发PWM使能.
[29:28]	**A0POSSEL**	**ACMP0 正极输入源选择位** 00 = 选择P1.5 作为ACMP0的正极输入. 01 = 选择P1.0作为ACMP0的正极输入. 10 = 选择P1.2作为ACMP0的正极输入. 11 = 选择P1.3作为ACMP0的正极输入.
[27]	**AOFFEN30**	**ACMP0 触发通道3输出一个周期使能位** 如果设置该位， ACMP0 触发该通道将强迫PWM 通道3 输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH3 禁止输出一个周期的低电平. 1 = PWM0_CH3 使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[26]	**AOFFEN20**	**ACMP0 触发通道2输出一个周期使能位** 如果设置该位， ACMP0 触发该通道将强迫PWM 通道2 输出至少一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH2 禁止输出一个周期的低电平. 1 = PWM0_CH2使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.

位	描述	
[25]	**AOFFEN10**	**ACMP0 触发通道1输出一个周期使能位** 如果设置该位， ACMP0 触发该通道将强迫PWM 通道1 输出至少一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH1 禁止输出一个周期的低电平. 1 = PWM0_CH1使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[24]	**AOFFEN00**	**ACMP0 触发通道0输出一个周期使能位** 如果设置该位， ACMP0 触发该通道将强迫PWM 通道0输出至少一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH0 禁止输出一个周期的低电平. 1 = PWM0_CH0使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[23]	**ACMP1TEN**	**ACMP1 触发PWM使能位** 0 = ACMP1触发PWM禁止. 1 = ACMP1触发PWM使能. **注意:** 如果AUTOCLR1被置位，当 ACMP1 触发 PWM时该位被自动清零.
[22]	**TMR1TEN**	**TIMER1 触发PWM 使能位** 当该位被置1, 当TIMER1溢出时, PHCHG_NXT寄存器的内容将更新到PWM_PHCHG 0 = TIMER1触发PWM禁止. 1 = TIMER1触发PWM使能.
[21:20]	**A1POSSEL**	**ACMP1 正极输入源选择位** 00 = 选择P3.1作为ACMP1的正极输入. 01 = 选择P3.2作为ACMP1的正极输入. 10 = 选择P3.3作为ACMP1的正极输入. 11 = 选择P3.4作为ACMP1的正极输入.
[19]	**AOFFEN31**	**ACMP1触发通道3输出一个周期使能位** 如果设置该位， ACMP1触发该通道将强迫PWM 通道3 输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH3 禁止输出一个周期的低电平. 1 = PWM0_CH3使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[18]	**AOFFEN21**	**ACMP1触发通道2输出一个周期使能位** 如果设置该位， ACMP1触发该通道将强迫PWM 通道2输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH2 禁止输出一个周期的低电平. 1 = PWM0_CH2使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.

位	描述	
[17]	**AOFFEN11**	**ACMP1触发通道1输出一个周期使能位** 如果设置该位， ACMP1触发该通道将强迫PWM 通道1输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH1 禁止输出一个周期的低电平. 1 = PWM0_CH1使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[16]	**AOFFEN01**	**ACMP1触发通道0输出一个周期使能位** 如果设置该位， ACMP1触发该通道将强迫PWM 通道0 输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH0 禁止输出一个周期的低电平. 1 = PWM0_CH0使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[15]	**AUTOCLR1**	**硬件自动清 ACMP1TEN 位** 0 = 当 ACMP1触发PWM时硬件将自动清除 ACMP1TEN. 1 = 当 ACMP1触发PWM时硬件不自动清除 ACMP1TEN
[14]	**AUTOCLR0**	**硬件自动清ACMP0TEN位** 0 = 当 ACMP0触发PWM时硬件将自动清除ACMP0TEN. 1 = 当 ACMP0触发PWM时硬件不自动清除ACMP0TEN
[13:8] n=0,1..5	**MSKENn**	**PWMn 输出屏蔽使能位** 0 = 输出 PWM_PHCHG 寄存器中MSKDATn设定的电平 1 = 输出通道n的原始波形
[7]	**MSKDAT7**	**PWM0_CH7 (GPIO P0.0) 屏蔽输出** 当 MASKEND7 为 1, 通道 7的输出电平为 MSKDAT7所设电平. 0 = PWM0_CH7 输出low. 1 = PWM0_CH7输出high.
[6]	**MSKDAT6**	**PWM0_CH6 (GPIO P0.1) 屏蔽输出** 当MASKEND6为 1, 通道6的输出电平为MSKDAT6所设电平. 0 = PWM0_CH6 输出low. 1 = PWM0_CH6输出high.
[5:0] n=0,1..5	**MSKDATn**	**PWM0_CHn 屏蔽输出** 当 MSKENn 为 0, 通道n的输出电平为 MSKDATn所设定的电平. 0 = PWM0_CHn输出low . 1 = PWM0_CHn输出high.

相位修改寄存器(PWM_PHCHGNXT)

寄存器	偏移地址	R/W	描述	复位值
PWM_PHCHGNXT	PWM0_BA+0x7C	R/W	PWM 下一相位修改寄存器	0x0000_3F00

31	30	29	28	27	26	25	24
ACMP0TEN	**TMR0TEN**	**A0POSSEL**		**AOFFEN30**	**AOFFEN20**	**AOFFEN10**	**AOFFEN00**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACMP1TEN	**TMR1TEN**	**A1POSSEL**		**AOFFEN31**	**AOFFEN21**	**AOFFEN11**	**AOFFEN01**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
AUTOCLR1	**AUTOCLR0**	**MSKEN5**	**MSKEN4**	**MSKEN3**	**MSKEN2**	**MSKEN1**	**MSKEN0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSKDAT7	**MSKDAT6**	**MSKDAT5**	**MSKDAT4**	**MSKDAT3**	**MSKDAT2**	**MSKDAT1**	**MSKDAT0**

位	描述	
[31]	**ACMP0TEN**	**ACMP0 触发PWM使能位** 0 = ACMP0触发PWM禁止. 1 = ACMP0 触发PWM使能. **注意:**如果AUTOCLR0为1,当ACMP0触发PWM，该位将自动被清零 .
[30]	**TMR0TEN**	**TMR0 触发PWM使能位** 当该位被置1, 当TIMER0溢出时, PHCHG_NXT寄存器的内容将更新到PWM_PHCHG 0 = TMR0触发PWM禁止. 1 = TMR0触发PWM使能
[29:28]	**A0POSSEL**	**ACMP0 正极输入源选择位** 00 = 选择 P1.5做ACMP0的正极输入. 01 = 选择P1.0做ACMP0的正极输入. 10 = 选择P1.2做ACMP0的正极输入. 11 = 选择P1.3做ACMP0的正极输入.
[27]	**AOFFEN30**	**ACMP0 触发通道3输出一个周期使能位** 如果设置该位， ACMP0 触发该通道将强迫PWM 通道3 输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH3 禁止输出一个周期的低电平. 1 = PWM0_CH3 使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.

位	描述	
[26]	**AOFFEN20**	**ACMP0 触发通道2输出一个周期使能位** 如果设置该位， ACMP0 触发该通道将强迫PWM 通道2输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH2 禁止输出一个周期的低电平. 1 = PWM0_CH2 使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[25]	**AOFFEN10**	**ACMP0 触发通道1输出一个周期使能位** 如果设置该位， ACMP0 触发该通道将强迫PWM 通道1输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH1禁止输出一个周期的低电平. 1 = PWM0_CH1 使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[24]	**AOFFEN00**	**ACMP0 触发通道0输出一个周期使能位** 如果设置该位， ACMP0 触发该通道将强迫PWM 通道0输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH0禁止输出一个周期的低电平. 1 = PWM0_CH0使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[23]	**ACMP1TEN**	**ACMP1 触发PWM使能位** 0 = ACMP1触发PWM禁止. 1 = ACMP1触发PWM使能. **注意:** 如果AUTOCLR1被置位，当 ACMP1 触发 PWM时该位被自动清零.
[22]	**TMR1TEN**	**TMR1触发PWM 使能位** 当该位被置1, 当TMR1溢出时, PHCHG_NXT寄存器的内容将更新到PWM_PHCHG 0 = TIMER1触发PWM禁止. 1 = TIMER1触发PWM使能.
[21:20]	**A1POSSEL**	**ACMP1 正极输入源选择位** 00 = 选择P3.1作为ACMP1的正极输入. 01 = 选择P3.2作为ACMP1的正极输入. 10 = 选择P3.3作为ACMP1的正极输入. 11 = 选择P3.4作为ACMP1的正极输入.
[19]	**AOFFEN31**	**ACMP1 触发通道3输出一个周期使能位** 如果设置该位， ACMP1 触发该通道将强迫PWM 通道3输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH3禁止输出一个周期的低电平. 1 = PWM0_CH3使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.

位	描述	
[18]	**AOFFEN21**	**ACMP1触发通道2输出一个周期使能位** 如果设置该位， ACMP1触发该通道将强迫PWM 通道2输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH2 禁止输出一个周期的低电平. 1 = PWM0_CH2使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[17]	**AOFFEN11**	**ACMP1触发通道1输出一个周期使能位** 如果设置该位， ACMP1触发该通道将强迫PWM 通道1输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH1 禁止输出一个周期的低电平. 1 = PWM0_CH1使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[16]	**AOFFEN01**	**ACMP1触发通道0输出一个周期使能位** 如果设置该位， ACMP1触发该通道将强迫PWM 通道0 输出一个周期的低电平， 该特性主要用于步进马达方面的应用 0 = PWM0_CH0 禁止输出一个周期的低电平. 1 = PWM0_CH0使能输出一个周期的低电平. **Note:** 该功能仅适用于 PWM0_CH0~ PWM0_CH3.
[15]	**AUTOCLR1**	**硬件自动清ACMP1TEN位** 0 = 当ACMP1触发PWM时硬件将自动清除ACMP1TEN. 1 = 当ACMP1触发PWM时硬件不自动清除ACMP1TEN
[14]	**AUTOCLR0**	**硬件自动清ACMP0TEN位** 0 = 当ACMP0触发PWM时硬件将自动清除ACMP0TEN. 1 = 当ACMP0触发PWM时硬件不自动清除ACMP0TEN
[13:8] n=0,1..5	**MSKENn**	**PWMn 输出屏蔽使能位** 0 = 输出PWM_PHCHG寄存器中MSKDATn设定的电平 1 = 输出通道n的原始波形
[7]	**MSKDAT7**	**PWM0_CH7 (GPIO P0.0) 屏蔽输出** 当MASKEND7为 1, 通道 7的输出电平为MSKDAT7所设电平. 0 = PWM0_CH7 输出low. 1 = PWM0_CH7输出high.
[6]	**MSKDAT6**	**PWM0_CH6 (GPIO P0.1) 屏蔽输出** 当MASKEND6为 1, 通道6的输出电平为MSKDAT6所设电平. 0 = PWM0_CH6 输出low. 1 = PWM0_CH6输出high
[5:0] n=0,1..5	**MSKDATn**	**PWM0_CHn 屏蔽输出** 当MSKENn为 0, 通道n的输出电平为MSKDATn所设定的电平. 0 = PWM0_CHn输出low . 1 = PWM0_CHn输出high.

相位修改屏蔽寄存器(PWM_PHCHGMSK)

寄存器	偏移地址	R/W	描述	复位值
PWM_PHCHGMSK	PWM0_BA+0x80	R/W	PWM 相位修改屏蔽寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**POSCTL**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MASKEND7	**MASKEND6**	**Reserved**					

位	描述	
[31:10]	**Reserved**	保留.
[9]	**POSCTL[1]**	**ACMP1的正极输入控制** 0 = ACMP的输入由CMP1CR控制. 1 = ACMP 的输入由PWM_PHCHG 寄存器的CMP1SEL位来控制. **注意:** 寄存器 CMP1CR的内容在比较器控制章节有描述
[8]	**POSCTL[0]**	**ACMP0 的正极输入控制** 0 = ACMP的输入由CMP0CR控制. 1 = ACMP 的输入由PWM_PHCHG寄存器的CMP0SEL位来控制. **注意:** 寄存器CMP0CR的内容在比较器控制章节有描述
[7]	**MASKEND7**	**PWM0_CH7 (GPIO P0.0)输出屏蔽使能位** 0 = 输出 GPIO P0.0的原始信号. 1 = 输出 寄存器 PWM_PHCHG 位7 MSKDAT7设定的电平.
[6]	**MASKEND6**	**PWM0_ CH6 (GPIO P0.1)输出屏蔽使能位** 0 = 输出GPIO P0.1的原始信号. 1 = 输出 寄存器PWM_PHCHG位6MSKDAT6设定的电平.
[5:0]	**Reserved**	保留.

中断累加控制寄存器 (PWM_IFA)

寄存器	偏移地址	R/W	描述	复位值
PWM_IFA	PWM0_BA+0x84	R/W	PWM 周期中断累加控制寄存器	0x0000_00F0

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

位	描述	
[31:8]	**Reserved**	保留.
[7:4]	**IFCNT**	**中断累加计数器** 当 IFAEN被置1, 当ZIFn标志置1时 IFCNT递减1， and 当 IFCNT递减到0, PWMn 中断将产生， IFCNT的值将重载.
[3:1]	**Reserved**	保留.
[0]	**IFAEN**	**中断累加使能位** 0 = 中断累加禁止 1 = 中断累加使能.

精准PWM中心对齐控制寄存器(PWM_PCACTL)

寄存器	偏移地址	R/W	描述	复位值
PWM_PCACTL	PWM0_BA+0x88	R/W	PWM精准中心对齐控制寄存器	0x0000_0000

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

位	描述	
[31:1]	**Reserved**	保留.
[0]	**PCAEN**	**PWM**精准中心对齐使能位 0 = **PWM**精准中心对齐禁止 1 = **PWM**精准中心对齐使能.

相位修改屏蔽对齐寄存器(PWM_MSKALIGN)

寄存器	偏移地址	R/W	描述	复位值
PWM_MSKALIGN	PWM0_BA+0x8C	R/W	PWM 相位修改屏蔽对齐寄存器	0x0000_3F00

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**ALIGN5**	**ALIGN4**	**ALIGN3**	**ALIGN2**	**ALIGN1**	**ALIGN0**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**MSKEN5**	**MSKEN4**	**MSKEN3**	**MSKEN2**	**MSKEN1**	**MSKEN0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**MSKDAT5**	**MSKDAT4**	**MSKDAT3**	**MSKDAT2**	**MSKDAT1**	**MSKDAT0**

位	描述	
[31:22]	**Reserved**	保留.
[21:16]	**ALIGNn**	**PWM0_CHn输出屏蔽对齐使能位** 0 = 当屏蔽功能使能后，PWM0_CHn输出将被立即屏蔽 1 = 当输出对齐到PWM周期PWM0_CHn 输出将屏蔽 .
[15:14]	**Reserved**	保留.
[13:8] n=0,1..5	**MSKENn**	**PWM 输出屏蔽使能位** 0 = 输出PWM_PHCHG 寄存器的位n MSKDATn指定的电平. 1 = 输出原始通道n的波形.
[7:6]	**Reserved**	保留.
[5:0] n=0,1..5	**MSKDATn**	**PWM0_CHn 屏蔽位** 当 MSKENn 为0, 通道 n的输出电平MSKDATn由控制 0 = PWM0_CHn 输出 low 电平. 1 = PWM0_CHn 输出 high 电平.

注意: PWM_PHCHG 和PWM_MSKALIGN寄存器的MSKENn 和 MSKDATn两个位都可以控制相同的位

6.8 看门狗定时器 (WDT)

6.8.1 概述

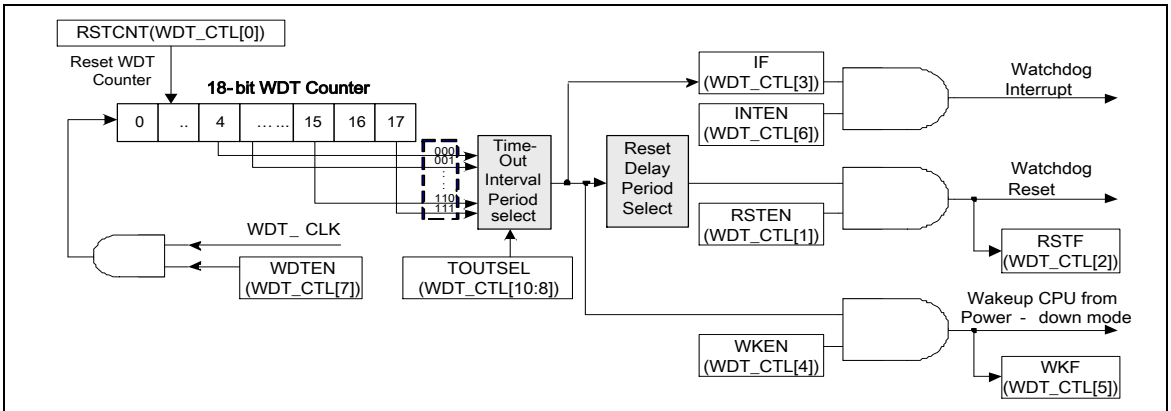
设计看门狗定时器的目的是，当系统运行到一个未知状态时，通过它来使系统复位。这种做法可以预防系统进入到无限期的死循环。此外，看门狗定时器还支持系统从空闲/掉电模式唤醒功能。

6.8.2 特性

- 18位的自由向上计数的看门狗定时器，可用来满足用户溢出时间间隔要求
- 溢出时间间隔(2^4 ~ 2^{18})个WDT_CLK时钟周期可选，如果WDT_CLK = 10 kHz，那么溢出时间间隔是1.6 ms ~ 26.214s
- 系统复位保持时间(1 / WDT_CLK) * 63
- 支持可选择的WDT复位延时周期，包括有1026、130、18 或 3 个WDT_CLK周期
- 配置Config0寄存器的CWDTEN[2:0]位，可以使芯片上电或复位条件下看门狗强制打开。
- 支持看门狗定时器溢出唤醒功能，此时时钟源必须选择为LIRC或LXT

6.8.3 框图

看门狗定时器模块图如图 6.8-1



注1: 看门狗复位CPU需持续63个WDT_CLK时钟

注2: 看门狗复位延时周期可以选择为3/18/130/1026 个WDT_CLK时钟

图 6.8-1 看门狗定时器模块图

6.8.4 时钟控制

看门狗定时器时钟控制如图 6.8-2

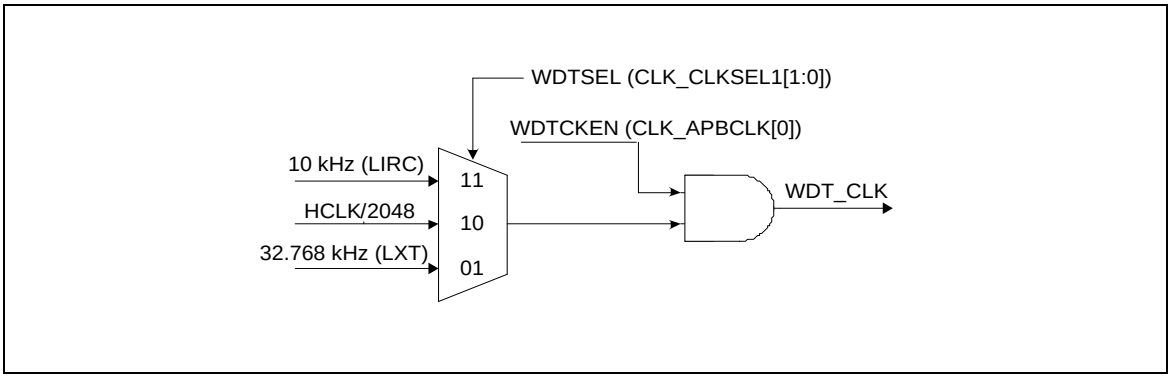


图 6.8-2 看门狗定时器时钟控制

6.8.5 基本配置

看门狗定时器的时钟通过WDTCKEN (CLK_APBCLK[0])位使能，时钟源由WDTSEL (CLK_CLKSEL1[1:0])位选择。

当CWDTEN[2:0]位没有配置成111时(CWDTEN[2] 对应Config0[31], CWDTEN[1:0] 对应Config0[4:3])，在芯片上电或复位后，系统将强制使能看门狗定时器，并会激活10k时钟。

6.8.6 功能描述

看门狗定时器(WDT)包含一个18位的可设置溢出时间间隔的自由向上计数器。表6.8-1显示了WDT溢出时间间隔选择，图 6.8-3 显示了WDT溢出时间间隔和复位周期时序。

6.8.6.1 WDT 定时溢出中断

对WDTEN (WDT_CTL[7])置位可以使能WDT功能，然后WDT计数器开始向上计数。通过设置TOUTSEL (WDT_CTL[10:8])，可以选择8个不同的溢出时间间隔。当WDT计数器计到TOUTSEL (WDT_CTL[10:8])设定值时，WDT将产生溢出中断，然后WDT的溢出中断标志IF (WDT_CTL[3])立即被置为1。

6.8.6.2 WDT 复位延时周期和复位系统

看门狗中断标志 IF (WDT_CTL[3])被置位后，会产生一个固定的复位延时周期T_{RSTD}。用户应在T_{RSTD} 复位延时计完之前，对RSTCNT (WDT_CTL[0])置1，来复位18位的WDT向上计数器的值，以防止产生WDT时间溢出复位信号。为清除WDT计数器，用户还需设置RSTDSEL (WDT_ALTCTL [1:0])，来选择合适的复位延时周期。如T_{RSTD} 时间计满以后，WDT向上计数器的值仍没有被清除，WDT控制器将对RSTF (WDT_CTL[2])置1，如果RSTEN (WDT_CTL[1])位有被使能，芯片立即进入复位状态。请参考图6.8-3看门狗定时器时间溢出间隔和复位周期时序，T_{RST}复位周期将保持至少63个WDT时钟周期，然后芯片从复位向量地址(0x0000_0000)重新开始执行程序。芯片由WDT定时溢出复位后，RSTF (WDT_CTL[2])标志将会保持为1.用户可以通过检查RSTF (WDT_CTL[2])标志来确认系统是否是因WDT定时溢出而产生的复位。

TOUTSEL	定时溢出间隔周期 T_{TIS}	复位延时周期 T_{RSTD}
000	$2^4 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
001	$2^6 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
010	$2^8 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
011	$2^{10} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
100	$2^{12} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
101	$2^{14} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
110	$2^{16} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
111	$2^{18} * T_{WDT}$	(3/18/130/1026) * T_{WDT}

表 6.8-1 看门狗定时器定时溢出间隔周期选择

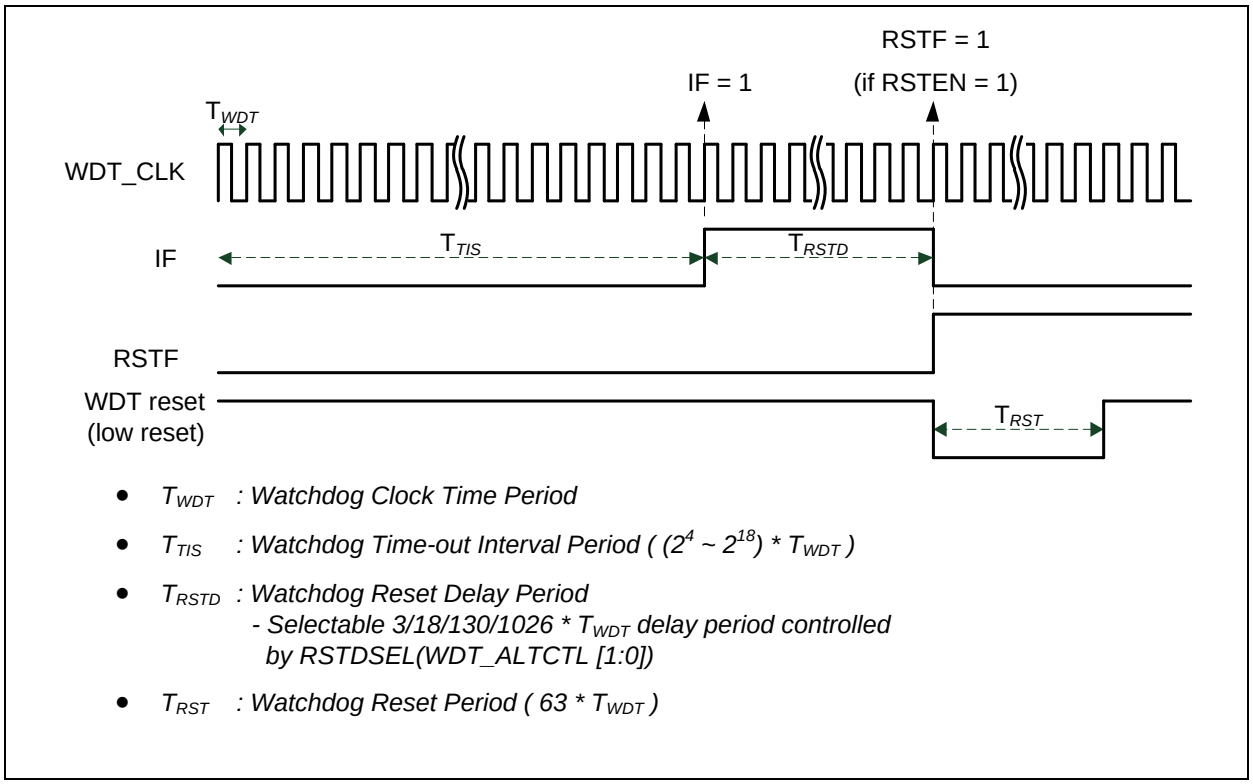


图 6.8-3 看门狗定时器定时溢出间隔和复位周期时序图

6.8.6.3 看门狗唤醒

如果看门狗时钟源选择为内部10k或者外部低速时钟, 而且WKEN (WDT_CTL[4])位被使能，当看门狗定时溢出中断产生后，系统能从Power-down模式下唤醒。同时，WKF (WDT_CTL[5])标志会被自动置1。用户可以通过查询WKF (WDT_CTL[5])标志知道系统是否是被看门狗定时溢出中断唤醒。

6.8.7 寄存器表

R: 只读, **W**: 只写, **R/W**: 可读写

寄存器	偏移地址	R/W	描述	复位值
WDT 基地址: **WDT_BA = 0x4000_4000**				
WDT_CTL	WDT_BA+0x00	R/W	看门狗定时器控制寄存器	0x0000_0700
WDT_ALTCTL	WDT_BA+0x04	R/W	看门狗定时器选择控制寄存器	0x0000_0000

6.8.8 寄存器描述

看门狗定时器控制寄存器 (WDT_CTL)

寄存器	偏移地址	R/W	描述	复位值
WDT_CTL	WDT_BA+0x00	R/W	看门狗定时器控制寄存器	0x0000_0700

31	30	29	28	27	26	25	24
ICEDEBUG	保留						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留					**TOUTSEL**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WDTEN	**INTEN**	**WKF**	**WKEN**	**IF**	**RSTF**	**RSTEN**	**RSTCNT**

位	描述	
[31]	**ICEDEBUG**	**ICE 调试模式下看门狗计数控制位(写保护)** 0 = ICE 调试模式影响看门狗计数，当ICE导致CPU运行停止时，看门狗计数器也停止。 1 = ICE 调试模式不影响看门狗计数。不论CPU由ICE停止还是运行，看门狗继续计数。 **注：** 该位是写保护位。参考SYS_REGLCTL寄存器。
[30:11]	**保留**	保留
[10:8]	**TOUTSEL**	**看门狗定时器溢出时间间隔选择 (写保护)** 以下三个位用于选择看门狗定时溢出周期 000 = 2^4 *T_{WDT}. 001 = 2^6 * T_{WDT}. 010 = 2^8 * T_{WDT}. 011 = 2^{10} * T_{WDT}. 100 = 2^{12} * T_{WDT}. 101 = 2^{14} * T_{WDT}. 110 = 2^{16} * T_{WDT}. 111 = 2^{18} * T_{WDT}. **注：** 该位是写保护位。参考SYS_REGLCTL寄存器
[7]	**WDTEN**	**看门狗定时器使能位 (写保护)** 0 = 看门狗定时器禁止. (该操作会复位看门狗计数器的值.) 1 = 看门狗定时器使能. **注：** 该位是写保护位。参考SYS_REGLCTL寄存器

位	描述	
[6]	**INTEN**	**看门狗定时器定时溢出中断使能位(写保护)** 如果该位使能,看门狗定时溢出后会产生中断信号，并告知CPU 0 = 看门狗定时溢出中断禁止. 1 = 看门狗定时溢出中断使能. **注：** 该位是写保护位。参考SYS_REGLCTL寄存器
[5]	**WKF**	**看门狗定时器溢出唤醒标志** 该位表示看门狗定时器中断唤醒标志的状态 0 = 看门狗定时器没有导致芯片唤醒 1 = 芯片从Idle或Power-down 模式由看门狗定时溢出中断唤醒 **注1:** 该位是写保护位。参考SYS_REGLCTL寄存器 **注2:** 该位为写1清零.
[4]	**WKEN**	**看门狗定时器定时溢出唤醒功能控制位 (写保护)** 如果此位被置1, 当定时溢出中断标志IF (WDT_CTL[3])被置1且中断使能位INTEN (WDT_CTL[6])被使能时，看门狗定时溢出中断信号将会触发事件唤醒MCU 0 = 唤醒触发事件禁止，即便看门狗定时器定时溢出中断发生 1 = 唤醒触发事件使能，如果看门狗定时溢出中断产生，将唤醒MCU **注1：** 该位是写保护位。参考SYS_REGLCTL寄存器 **注2:** MCU能被看门狗定时器定时溢出中断信号唤醒的条件是看门狗定时器时钟源必须为内部10k时钟或者是外部低速时钟LXT。
[3]	**IF**	**看门狗定时器定时溢出标志** 当看门狗定时器计数器的值达到溢出时间间隔时，该位将被置1 0 = 没有发生看门狗定时溢出中断 1 = 有看门狗定时溢出中断发生. **注意:** 该位写1清零
[2]	**RSTF**	**看门狗定时器复位标志** 该位用来指示系统是否因看门狗定时器定时溢出发生复位 0 = 没有发生看门狗定时溢出复位 1 = 发生了看门狗定时溢出复位. **注意:** 该位写1清零.
[1]	**RSTEN**	**看门狗定时器复位使能位 (写保护)** 该位是看门狗定时溢出复位功能使能位。如果看门狗计数器的值达到定时溢出的值，且固定的WDT复位延时时间已经计完，如果此时还没有对计数器清零，如果该位被设置，会发生看门狗定时溢出复位。 0 = 看门狗定时溢出复位功能禁止 1 = 看门狗定时溢出复位功能使能 **注：** 该位是写保护位。参考SYS_REGLCTL寄存器
[0]	**RSTCNT**	**清看门狗计数器 (写保护)** 0 = 不影响. 1 =18位看门狗定时器计数值复位清零. **注1:** 该位是写保护位。参考SYS_REGLCTL寄存器 **注2:** 该位硬件自动清零

看门狗定时器选择控制寄存器 (WDT_ALTCTL)

寄存器	偏移地址	R/W	描述	复位值
WDT_ALTCTL	WDT_BA+0x04	R/W	看门狗定时器选择控制寄存器	0x0000_0000

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

位	描述	
[31:2]	保留	保留
[1:0]	**RSTDSEL**	**看门狗定时器复位延时选择 (写保护)** 当看门狗定时器发生溢出后, 用户可以在看门狗复位延时的时段内，通过设置RSTCNT (WDT_CTL[0])来对看门狗计数器清零，以防止看门狗溢出复位。 对不同的看门狗溢出时间，用户也可以通过设置RSTDSEL来选择适当的看门狗复位延时时间。 00 = 看门狗定时器复位延时时间是1026 * WDT_CLK. 01 = 看门狗定时器复位延时时间是130 * WDT_CLK. 10 = 看门狗定时器复位延时时间是18 * WDT_CLK. 11 = 看门狗定时器复位延时时间是3 * WDT_CLK. **注1：**该位是写保护位。参考SYS_REGLCTL寄存 **注2：**如果看门狗定时器定时溢出复位发生，该寄存器的值将被清零

6.9 窗口看门狗定时器 (WWDT)

6.9.1 概述

窗口看门狗定时器(WWDT)用于在一个特定窗口时间内执行系统复位，以防止程序在不可预知条件下跑到一个不可控的状态

6.9.2 特性

- 6位向下计数值(CNTDAT) 和6位比较窗口值(CMPDAT)，使得窗口周期更加灵活
- 支持4-位值(PSCSEL)选择窗口看门狗预分频值，预分频计数器最大可达11位

6.9.3 框图

窗口看门狗定时器的模块图如图 6.9-1.

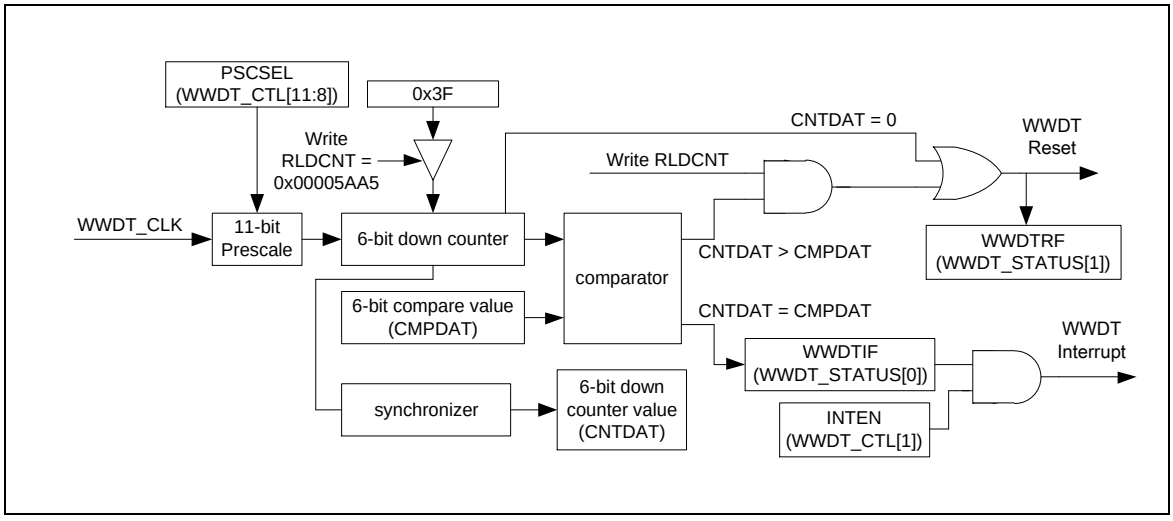


图 6.9-1 窗口看门狗定时器模块图

6.9.4 时钟控制

窗口看门狗定时器 的时钟控制如图图 6.9-2.

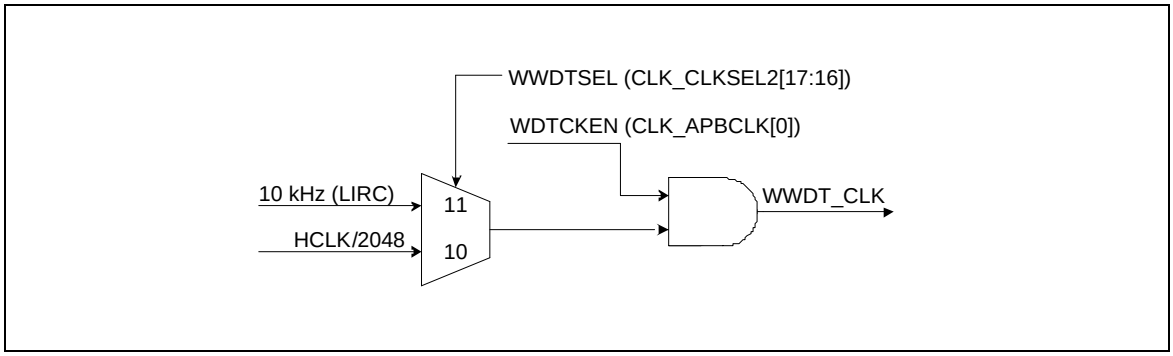


图 6.9-2 窗口看门狗定时器时钟控制图

6.9.5 基本配置

窗口看门狗定时器的外设时钟源通过WDTCKEN (CLK_APBCLK[0])来使能，时钟源通过WWDTSEL[1:0] (CLK_CLKSEL2[17:16])来选择。

6.9.6 功能描述

窗口看门狗定时器(WWDT)是一个 6位向下计数器，该计数器带一个可选择预分频值，不同的预分频值对应不同的看门狗定时溢出时间。6位窗口看门狗定时器的时钟源可以是系统时钟经2048 (HCLK/2048)分频的时钟，也可以是内部10k低速RC振荡时钟源(LIRC)，看门狗的时钟源带一个可选择的11位预分频值，该值可通过PSCSEL (WWDT_CTL[11:8])位来设置选择，对应预分频值如下表 6.9-1。

PSCSEL	预分频值	最大定时溢出间隔	最大定时溢出间隔 (WWDT_CLK=10 KHz)
0000	1	1 * 64 * T_{WWDT}	6.4 ms
0001	2	2 * 64 * T_{WWDT}	12.8 ms
0010	4	4 * 64 * T_{WWDT}	25.6 ms
0011	8	8 * 64 * T_{WWDT}	51.2 ms
0100	16	16 * 64 * T_{WWDT}	102.4 ms
0101	32	32 * 64 * T_{WWDT}	204.8 ms
0110	64	64 * 64 * T_{WWDT}	409.6 ms
0111	128	128 * 64 * T_{WWDT}	819.2 ms
1000	192	192 * 64 * T_{WWDT}	1.2288 s
1001	256	256 * 64 * T_{WWDT}	1.6384 s
1010	384	384 * 64 * T_{WWDT}	2.4576 s

1011	512	512 * 64 * T_{WWDT}	3.2768 s
1100	768	768 * 64 * T_{WWDT}	4.9152 s
1101	1024	1024 * 64 * T_{WWDT}	6.5536 s
1110	1536	1536 * 64 * T_{WWDT}	9.8304 s
1111	2048	2048 * 64 * T_{WWDT}	13.1072 s

表 6.9-1 窗口看门狗定时器预分频值选择

6.9.6.1 窗口看门狗定时器的计数

当WWDTEN (WWDT_CTL[0])位被使能，窗口看门狗向下计数器将会从0x3F向下递减计数到0。为了防止程序在非用户指定位置关闭窗口看门狗定时器，窗口看门狗定时器控制寄存器WWDT_CTL在芯片上电或复位后仅可写一次。当WWDTEN (WWDT_CTL[0])位被软件使能以后，用户不能禁止窗口看门狗定时器(WWDTEN), 不能修改计数器预分频周期(PSCSEL)，也不能修改窗口比较值(CMPDAT)，除非芯片复位。

6.9.6.2 窗口看门狗定时器比较匹配中断

窗口看门狗定时器向下计数过程中, 当窗口看门狗定时器计数值(CNTDAT) 等于比较值(CMPDAT)时，WWDTIF (WWDT_STATUS[0])会被置1，并且WWDTIF可以被软件清零。如果INTEN (WWDT_CTL[1])位被使能, 当WWDTIF位被硬件置1时，就会产生窗口看门狗比较匹配中断。

6.9.6.3 窗口看门狗定时器复位系统

当WWDTIF (WWDT_STATUS[0])产生后，用户必须通过对WWDT_RLDCNT寄存器写0x00005AA5进行重载使窗口看门狗计数器的值回到初始值0x3F，这样可以防止窗口看门狗定时器的值继续向下计数到0，从而产生窗口看门狗定时器复位系统信号通知系统复位。如果CNTDAT (WWDT_CNT[5:0])的当前值大于CMPDAT (WWDT_CTL[21:16])的值，此时用户对WWDT_RLDCNT寄存器写0x00005AA5，窗口看门狗定时器复位系统信号将立刻产生，并导致芯片复位。

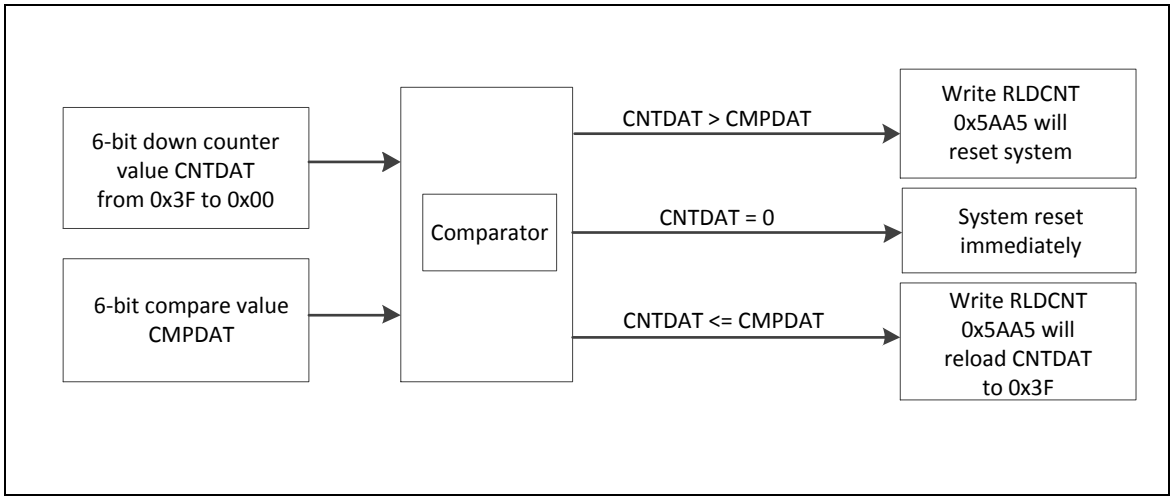


图 6.9-3 窗口看门狗定时器复位和重载过程

6.9.6.4 窗口看门狗定时器的窗口设置限制

当用户对WWDT_RLDCNT寄存器写0x00005AA5来重载窗口看门狗计数器的值到0x3F的时候，大概需要3个窗口看门狗定时器时钟来同步重载命令到实际执行重载操作。这意味着如果用户设置PSCSEL (WWDT_CTL[11:8])的值为0000，即计数器预分频值设置为1，CMPDAT(WWDT_CTL[21:16])的值必须大于2；否则，当WWDTIF (WWDT_STATUS[0])标志产生后，写WWDT_RLDCNT寄存器来重载窗口看门狗定时器的值为0x3F操作无效，将会一直出现窗口看门狗定时器引起的系统复位。

PSCSEL	预分频值	有效的CMPDAT比较值
0000	1	0x3 ~ 0x3F
0001	2	0x2 ~ 0x3F
其它	其它	0x0 ~ 0x3F

表 6.9-2 CMPDAT 设置限制

6.9.7 寄存器表

R: 只读, **W**: 只写, **R/W**: 可读写

寄存器	偏移地址	R/W	描写	复位值
WWDT 基地址: **WWDT_BA = 0x4000_4100**				
WWDT_RLDCNT	WWDT_BA+0x00	W	窗口看门狗定时器重载计数器寄存器	0x0000_0000
WWDT_CTL	WWDT_BA+0x04	R/W	窗口看门狗定时器控制寄存器	0x003F_0800
WWDT_STATUS	WWDT_BA+0x08	R/W	窗口看门狗定时器状态寄存器	0x0000_0000
WWDT_CNT	WWDT_BA+0x0C	R	窗口看门狗定时器计数器值寄存器	0x0000_003F

6.9.8 寄存器描述

窗口看门狗定时器重载计数器寄存器 (WWDT_RLDCNT)

寄存器	偏移地址	R/W	描述	复位值
WWDT_RLDCNT	WWDT_BA+0x00	W	窗口看门狗定时器重载计数器寄存器	0x0000_0000

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

位	描述	
[31:0]	**RLDCNT**	**窗口看门狗定时器重载计数器寄存器** 对此寄存器写0x00005AA5 将会重载窗口看门狗定时器计数器的值到0x3F。 **注:**用户只能在当窗口看门狗计数器的值在0到CMPDAT (WWDT_CTL[21:16])之间才可以写WWDT_RLDCNT寄存器来重载窗口看门狗计数器。如果窗口看门狗计数器的值大于CMPDAT时写WWDT_RLDCNT寄存器，窗口看门狗定时器复位信号会立即产生。

窗口看门狗定时器控制寄存器 (WWDT_CTL)

寄存器	偏移地址	R/W	描述	复位值
WWDT_CTL	WWDT_BA+0x04	R/W	窗口看门狗定时器控制寄存器	0x003F_0800

注意:此寄存器在芯片上电或复位后只能写一次

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

位	描述	
[31]	**ICEDEBUG**	**ICE 调试模式下窗口看门狗计数控制位** 0 = ICE 调试模式下影响窗口看门狗定时器计数，当CPU被ICE暂停后，窗口看门狗向下计数器计数值将保持不变 1 = ICE 调试模式下不影响窗口看门狗定时器计数，窗口看门狗定时器向下计数器将会保持继续计数，不管CPU是否被ICE暂停。
[30:22]	保留	保留
[21:16]	**CMPDAT**	**窗口看门狗定时器窗口比较值** 设置此寄存器来调整有效重载窗口 **注:** 当窗口看门狗定时器计数器的值在0到CMPDAT之间时，用户才可以写寄存器WWDT_RLDCNT来重载窗口看门狗定时器计数器的值。如果在当前窗口看门狗定时器计数器的值大于CMPDAT时用户去写WWDT_RLDCNT寄存器，窗口看门狗定时器复位信号会立即产生。
[15:12]	保留	保留
[11:8]	**PSCSEL**	**窗口看门狗定时器计数器预分频周期选择位** 0000 = 预分频为1; 最大定时溢出周期是 1 * 64 * WWDT_CLK. 0001 = 预分频为2; 最大定时溢出周期是2 * 64 * WWDT_CLK. 0010 = 预分频为4; 最大定时溢出周期是4 * 64 * WWDT_CLK. 0011 = 预分频为8; 最大定时溢出周期是8 * 64 * WWDT_CLK. 0100 = 预分频为16; 最大定时溢出周期是16 * 64 * WWDT_CLK. 0101 = 预分频为32; 最大定时溢出周期是32 * 64 * WWDT_CLK. 0110 = 预分频为64; 最大定时溢出周期是64 * 64 * WWDT_CLK. 0111 = 预分频为128; 最大定时溢出周期是128 * 64 * WWDT_CLK. 1000 = 预分频为192; 最大定时溢出周期是192 * 64 * WWDT_CLK. 1001 = 预分频为256; 最大定时溢出周期是256 * 64 * WWDT_CLK. 1010 = 预分频为384; 最大定时溢出周期是384 * 64 * WWDT_CLK. 1011 = 预分频为512; 最大定时溢出周期是512 * 64 * WWDT_CLK.

		1100 = 预分频为768; 最大定时溢出周期是768 * 64 * WWDT_CLK. 1101 = 预分频为1024; 最大定时溢出周期是1024 * 64 * WWDT_CLK. 1110 = 预分频为1536; 最大定时溢出周期是1536 * 64 * WWDT_CLK. 1111 = 预分频为2048; 最大定时溢出周期是2048 * 64 * WWDT_CLK.
[7:2]	**保留**	保留
[1]	**INTEN**	**窗口看门狗定时器中断使能位** 如果该位被使能，窗口看门狗定时器计数器比较匹配中断信号产生时，会通知CPU 0 = 窗口看门狗定时器计数器比较匹配中断禁止 1 = 窗口看门狗定时器计数器比较匹配中断使能
[0]	**WWDTEN**	**窗口看门狗定时器使能位** 设置该位来使能窗口看门狗定时器计数器计数 0 = 窗口看门狗定时器计数器停止. 1 = 窗口看门狗定时器计数器计数开始

窗口看门狗定时器状态寄存器 (WWDT_STATUS)

寄存器	偏移地址	R/W	描述	复位值
WWDT_STATUS	WWDT_BA+0x08	R/W	窗口看门狗定时器状态寄存器	0x0000_0000

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

位	描述	
[31:2]	保留	保留
[1]	**WWDTRF**	**窗口看门狗定时器溢出复位标志** 该位指示系统是否已被窗口看门狗定时器溢出复位 0 = 窗口看门狗定时器没有发生复位 1 = 窗口看门狗定时器发生了复位 **注:** 该位是写1清零
[0]	**WWDTIF**	**窗口看门狗定时器比较匹配中断标志** 该位指示当窗口看门狗定时器计数器的值与CMPDAT (WWDT_CTL[21:16])的值相匹配时，窗口看门狗定时器的比较匹配中断状态标志 0 = 窗口看门狗定时器计数器的值与CMPDAT的值不匹配 1 = 窗口看门狗定时器计数器的值与CMPDAT的值匹配 **注:** 该位是写1清零

窗口看门狗定时器计数器值寄存器 (WWDT_CNT)

寄存器	偏移地址	R/W	描述	复位值
WWDT_CNT	WWDT_BA+0x0C	R	窗口看门狗定时器计数器值寄存器	0x0000_003F

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

位	描述	
[31:6]	保留	保留
[5:0]	**CNTDAT**	窗口看门狗定时器计数值 CNTDAT将会被持续更新，以指示6位向下计数器的值

6.10 串口控制器 (UART)

6.10.1 概述

NuMicro® Mini58系列提供2个通用异步串行接口(UART)。UART0支持流量控制。UART0和UART1控制器的接收过程是把外设的串行数据转为并行数据，发送过程是把CPU的并行数据转成串行数据发送出去。UART0控制器还支持IrDA SIR功能，支持RS-485功能模式。UART0支持6种中断类型，而UART1只支持5种中断类型。UART0支持16个字节接收发送FIFO，而UART1仅支持 1 个字节的接收发送缓存。

6.10.2 特性

- 全双工，异步通讯口
- 独立的接收/发送16/16字节FIFO（仅适合UART0）
- 独立的接收/发送1/1字节数据缓存区(仅适合UART1)
- 支持硬件自动流控功能(CTS, RTS)，RTS自动流控触发电平可设(仅适合UART0)
- 接收缓存触发等级可设(仅适合UART0)
- 每个通道波特率可单独设置
- 支持 CTS管脚触发唤醒功能(仅UART0 支持此功能)
- 支持8位接收缓存定时溢出检测功能
- 可通过设置UART_TOUT[15:8]寄存器的相应位来设置传送两个数据间（从上一个stop 位到下一个start位之间）的时间间隔
- 支持break error, frame error, parity error和收发缓冲区溢出检测等功能
- 可编程串行接口特性
  - 数据位长度可设为5~8位
  - 校验位可设为奇、偶校验、无校验或固定校验位的产生和检测
  - 可设置停止位长度为1位,1.5位或2位
- 支持IrDA SIR 功能模式(仅适合UART0)
  - 支持正常模式下3/16-位宽功能
- 支持RS-485功能模式(仅适合UART0)
  - 支持RS-485 9-位模式
  - 支持RTS管脚软硬件控制RS-485传输方向

6.10.3 框图

串口0时钟控制和框图如图 6.10-1 和 图 6.10-3所示，串口1时钟控制和框图如图 6.10-2和图 6.10-4所示。

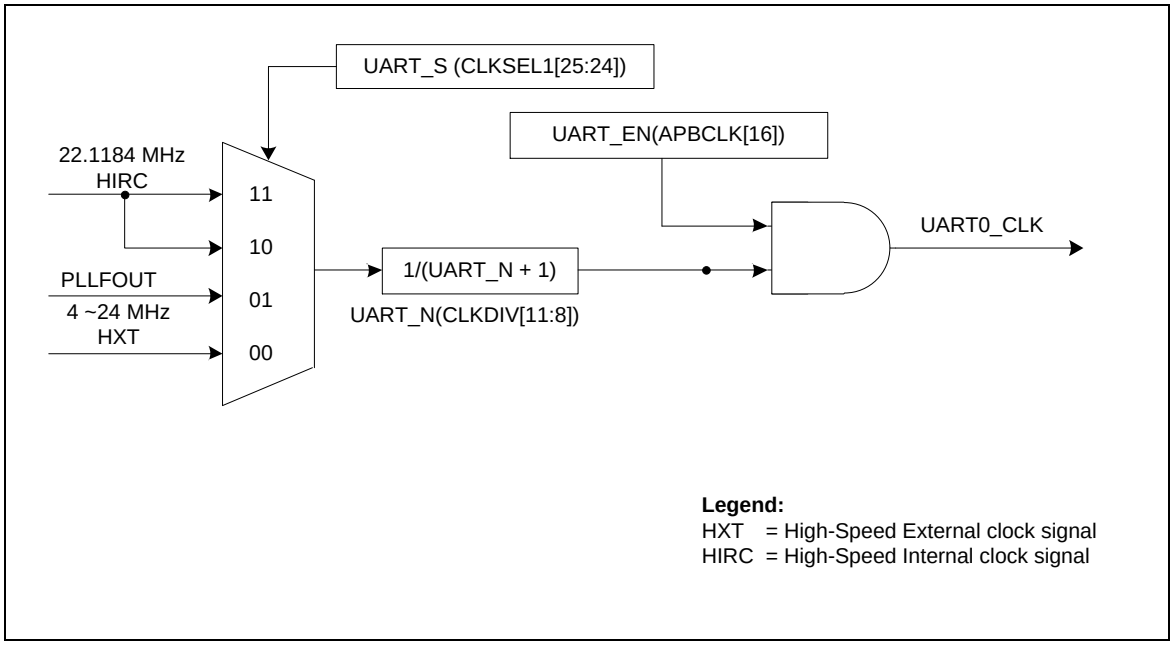


图 6.10-1 UART0 控制器时钟控制

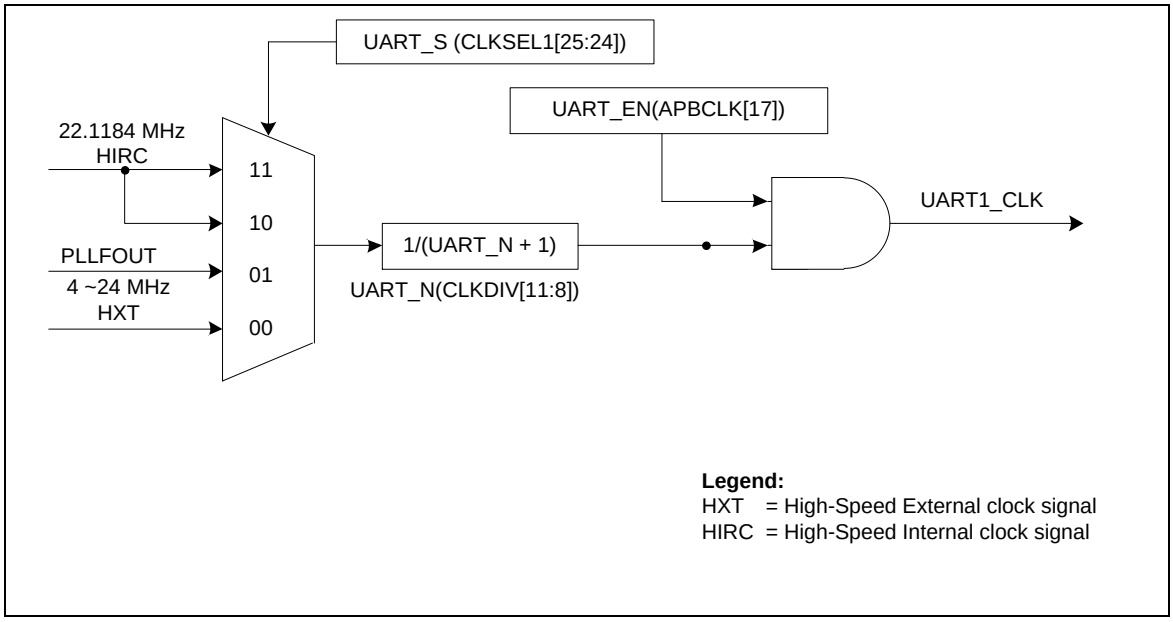


图 6.10-2 UART1 控制器时钟控制

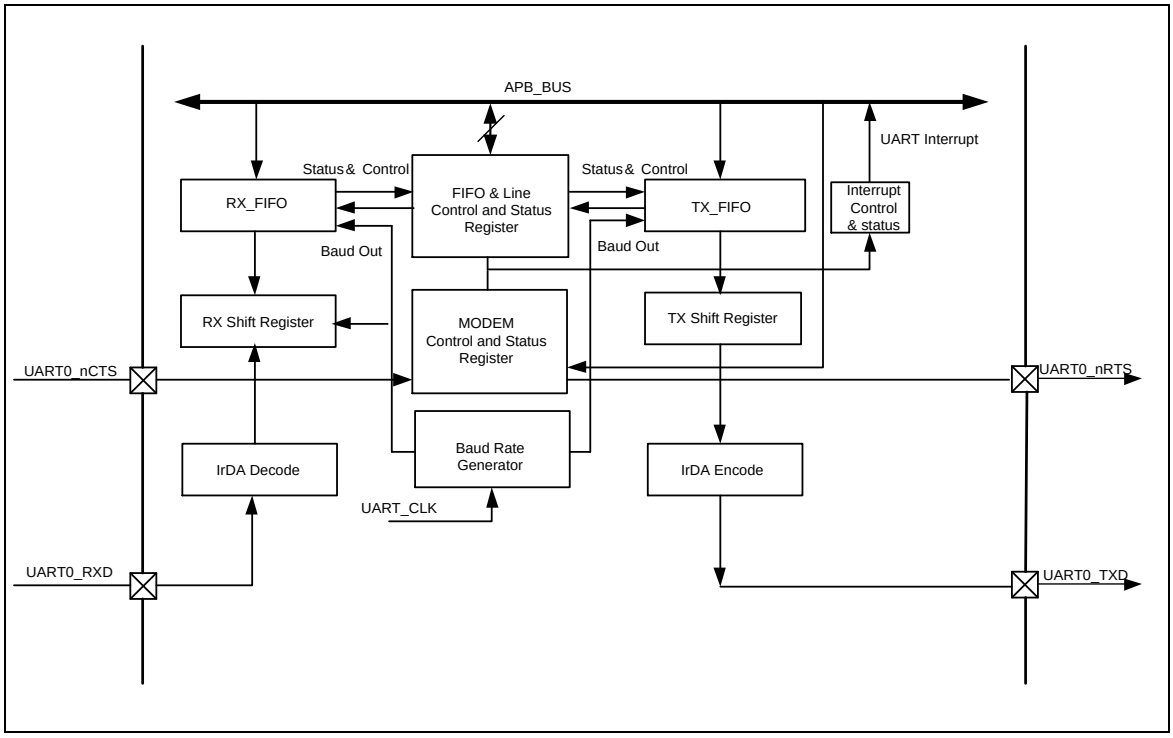


图 6.10-3 UART0 控制器框图

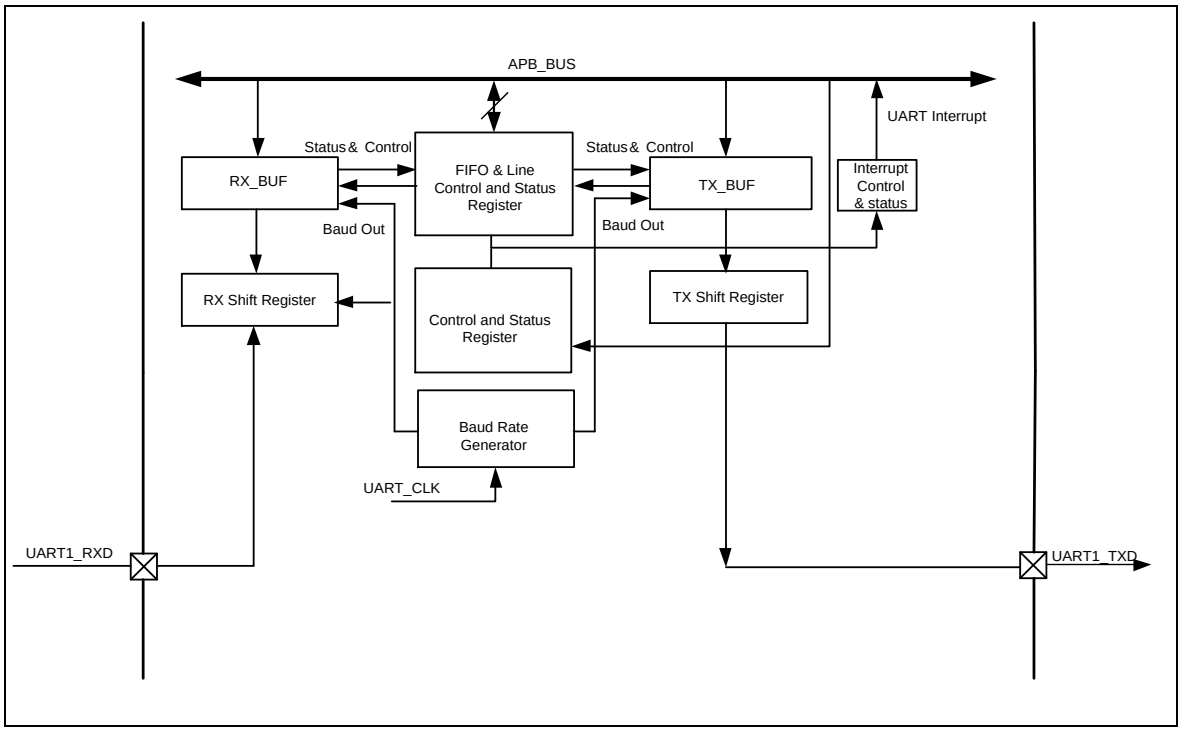


图 6.10-4 UART1 控制器框图

每个模块功能详细描述如下：

TX_FIFO/TX_BUF

UART0发送器带有一个16字节的FIFO缓冲区可减少CPU中断的频率。UART1发送器只有1个字节的缓存。

RX_FIFO/RX_BUF

UART0接收器带有一个16 字节的FIFO缓冲区可减少CPU中断的频率。UART1接收器只有1个字节的缓存(加上每字节有3个错误位)

发送移位寄存器

该模块用于控制把并行数据串行输出

接收移位寄存器

该模块用于控制把串行数据并行输入

波特率发生器.

通过把输入的外部时钟除频得到期望的波特率。详情请参考波特率公式

IrDA编码.

该模块为IrDA编码控制模块(仅适合UART0)

IrDA解码.

该模块为IrDA解码控制模块(仅适合UART0)

FIFO & 线控制和状态寄存器

该域的寄存器组有FIFO控制寄存器((UART_FIFO)，FIFO状态寄存器(UART_FIFOSTS)，和用于收发器的线控控制寄存器(UART_LINE)，时间溢出控制寄存器((UART_TOUT)配置时间溢出中断的条件。

中断控制和状态寄存器

一共有6种类型的中断，包括发送FIFO空中断(THERIF)，接收数据达到中断(RDAIF)中断，接收线状态中断(parity error 或 framing error 或 break interrupt) (RLSIF)，超时中断(RXTOINT)，缓存错误中断(BUFERRINT) 和nCTS唤醒中断(仅适合UART0)。中断使能寄存器(UART_INTEN) 用来使能或禁止相应中断，中断状态寄存器(UART_INTSTS)用以区分已发生的相应中断。

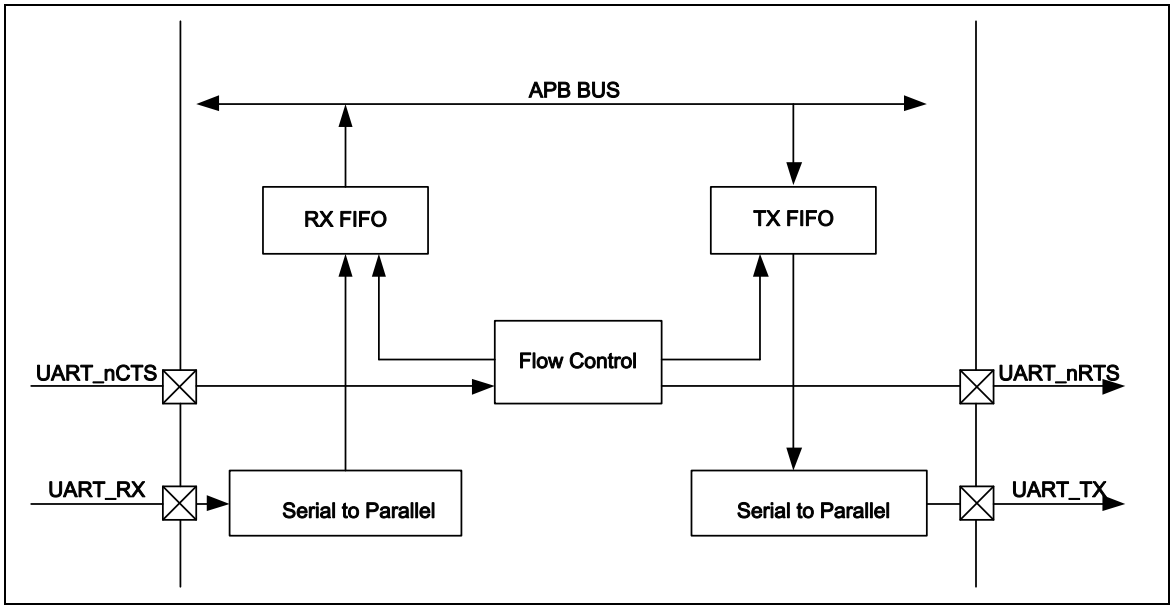


图 6.10-5 自动流控模块图

6.10.4 基本配置

UART控制器功能脚，在SYS_P0_MFP/SYS_P1_MFP寄存器中配置UART0，在SYS_P1_MFP/ SYS_P2_MFP/ SYS_P4_MFP寄存器中配置UART1。

UART控制器时钟在UART_EN (CLK_APBCLK[16])中使能。

UART控制器时钟源，在UARTSEL (CLKSEL[25:24])位来选择。

UART控制器时钟预分频通过UARTDIV (CLK_CLKDIV[11:8])位来设置。

6.10.5 功能描述

UART0控制器支持三个功能模式，包括UART, IrDA和RS-485模式。用户可以通过对UART_FUNSEL寄存器设置选择功能。UART1只支持UART数据发送和接收模式。

6.10.5.1 UART 控制器波特率发生器

UART控制器包含一个可编程波特率发生器，其通过分频器对输入时钟源分频而得到收发数据所需的串行时钟。波特率计算公式：波特率= UART_CLK / M * [BRD + 2]，这里M和BRD在波特率分频器寄存器(UART_BAUD)中有定义。图 6.10-6列出了各种条件下的UART波特率计算公式和串口波特率参数的设置。通过设置相应的参数和寄存器可以得到零误差的波特率。IrDA功能模式，波特率发生器的模式必须是模式0(仅适合UART0)。

模式	BAUDM1	BAUDM0	除数 X	BRD	M	波特率公式
模式 0	0	0	B	A	16	UART_CLK / [16 * (A+2)]
模式 1	1	0	B	A	B+1	UART_CLK / [(B+1) * (A+2)], B 必须 >= 8

模式 2	1	1	无关	A	1	UART_CLK / (A+2), A 必须 >=8

图 6.10-6 控制器波特率公式表

UART 外设时钟 = 22.1184 MHz			
波特率	**模式 0**	**模式 1**	**模式 2**
921600	不支持	A=0, B=11	A=22
460800	A=1	A=1, B=15 A=2, B=11	A=46
230400	A=4	A=4, B=15 A=6, B=11	A=94
115200	A=10	A=10, B=15 A=14, B=11	A=190
57600	A=22	A=22, B=15 A=30, B=11	A=382
38400	A=34	A=62, B=8 A=46, B=11 A=34, B=15	A=574
19200	A=70	A=126, B=8 A=94, B=11 A=70, B=15	A=1150
9600	A=142	A=254, B=8 A=190, B=11 A=142, B=15	A=2302
4800	A=286	A=510, B=8 A=382, B=11 A=286, B=15	A=4606

图 6.10-7 控制器波特率参数设置表

UART 外设设置 = 22.1184 MHz			
波特率	**模式 0**	**模式 1**	**模式 2**
921600	不支持	0x2B00_0000	0x3000_0016
460800	0x0000_0001	0x2F00_0001 0x2B00_0002	0x3000_002E
230400	0x0000_0004	0x2F00_0004 0x2B00_0006	0x3000_005E
115200	0x0000_000A	0x2F00_000A 0x2B00_000E	0x3000_00BE
57600	0x0000_0016	0x2F00_0016 0x2B00_001E	0x3000_017E

38400	0x0000_0022	0x2800_003E 0x2B00_002E 0x2F00_0022	0x3000_023E
19200	0x0000_0046	0x2800_007E 0x2B00_005E 0x2F00_0046	0x3000_047E
9600	0x0000_008E	0x2800_00FE 0x2B00_00BE 0x2F00_008E	0x3000_08FE
4800	0x0000_011E	0x2800_01FE 0x2B00_017E 0x2F00_011E	0x3000_11FE

图 6.10-8 控制器波特率寄存器设置表

6.10.5.2 UART控制器FIFO控制和状态

UART0内置一个16字节的发送FIFO (TX_FIFO)和一个16字节的接收FIFO (RX_FIFO)，通信中,使用这些收发FIFO可以减少对CPU的中断次数。CPU在任何时候都可以读到UART0的状态。状态信息包括UART0传输操作的类型和条件，以及接收过程有可能发生的3个错误状态（奇偶校验错误，帧错误，打断错误）。FIFO控制和状态也支持所有的功能模式，包括UART, IrDA, 和RS-485模式。

UART1内置一个字节的发送缓存和一个字节的接收缓存。CPU在任何时候都可以读到UART1的状态。状态信息包括UART1传输操作的类型和条件，以及接收过程有可能发生的3个错误状态（奇偶校验错误，帧错误，打断错误）。

6.10.5.3 UART控制器唤醒功能

只有UART0支持CTS唤醒功能。当芯片在Power-down模式下时，外部CTS管脚上的电平变化可以唤醒芯片。该功能在所有功能模式下都有效。用户必须使能WKCTSIEN(唤醒CPU中断功能控制位)来打开唤醒功能。如果WKCTSIEN (UART_INTEN[9])有被使能且CTS被激活，芯片可以从掉电模式中被唤醒。如果CTSWKIF (UART_INTSTS[16])被置位，表示芯片已由外部CTS管脚触发唤醒。一旦软件进入中断服务程序，CTSWKIF (UART_INTSTS[16])必须清零。UART1没有唤醒功能。

6.10.5.4 UART控制器中断和状态

UART0控制器支持6种类型的中断，UART1控制器支持5种类型。具体中断类型包括如下：

- 接收阈值水平达到后的中断(RDAINT)
- 发送FIFO空中断(THREINT)
- Line状态中断（奇偶校验错误，帧错误，打断中断）(RLSINT)
- MODEM/唤醒状态中断(MODEMINT)（仅UART0支持）
- 接收缓冲区定时溢出中断(RXTOINT)
- 缓冲区错误中断(BUFERRINT)

图 6.10-9描述了中断源和中断标志。当中断使能且有中断标志时就会产生中断。用户必须在中断产生后清中断标志。

中断源	中断指示	中断使能位	中断标志	清中断标志方法
接收数据有效中断	RDAINT	RDAIEN	**RDAIF**	读 UART_DAT
发送保持寄存器空中断	THREINT	THREIEN	**THREIF**	写 UART_DAT
接收Line状态中断	RLSINT	RLSIEN	**RLSIF** = (BIF 或 FEF 或 PEF)	写 '1' 到 BIF/FEF/ PEF
			RLSIF = ADDRDETF	写 '1' 到 ADDRDETF
Modem 状态中断 (仅 UART0支持)	MODEMINT	MODEMIEN	**MODEMIF** = CTSDETF	写 '1' 到 CTSDETF
RX 定时溢出中断	RXTOINT	RXTOIEN	**RXTOIF**	读 UART_DAT
缓冲区错误中断	BUFERRINT	BUFERRIEN	**BUFERRIF** = (TXOVIF 或 RXOVIF)	写 '1' 到 TXOVIF / RXOVIF

图 6.10-9 UART 控制器中断源和标志

6.10.5.5 UART 功能模式

UART0/UART1控制器提供了UART功能（用户须设置UART_FUNSEL [1:0]为"00"把控制器设置为UART功能模式）。UART 波特率最高速度是1 Mbps.

UART为全双工异步通讯接口。UART0收发器各包含一个16字节的FIFO缓冲区(仅UART0支持)。用户可以设置接收缓存触发水平和定时溢出检测时间(仅UART0支持)。发送数据帧间（即从上一帧停止位到下一帧起始位）的时间间隔可通过DLY (UART_TOUT [15:8])位设置(UART0/ UART0支持)。UART0支持硬件自动流控功能(CTS, RTS),且RTS流控的触发电平可设，全双工串行接口通讯参数可设(仅UART0支持)。

UART 线控制功能

UART控制器通过设置UART_LINE寄存器支持串行接口全部特性。可以通过对UART_LINE寄存器设置数据位和停止位的长度以及奇偶校验位。下图 6.10-10列出了UART数据位和停止位长度的设置以及UART奇偶校验位的设置。

NSB (UART_LINE[2])	WLS (UART_LINE[1:0])	数据位长度(位)	停止位长度 (位)
0	00	5	1
0	01	6	1
0	10	7	1
0	11	8	1
1	00	5	1.5
1	01	6	2

1	10	7	2
1	11	8	2

图 6.10-10 UART线控制的数据位和停止位的长度设置

奇偶校验类型	SPE (UART_LINE[5])	EPE (UART_LINE[4])	PBE (UART_LINE[3])	描述
无校验	x	x	0	无奇偶校验位输出
奇校验	0	0	1	奇校验位的计算方法是把数据流中的所有的1相加，使得包括校验位在内，1的总数为奇数个
偶校验	0	1	1	偶校验位的计算方法是把数据流中的所有的1相加，使得包括校验位在内，1的总数为偶数个.
奇偶校验位强制置1	1	0	1	奇偶校验位总是逻辑1 不管数据位中1的个数是多少（奇偶计数），奇偶校验位永远都是逻辑1
奇偶校验位强制置0	1	1	1	奇偶校验位总是逻辑0 不管数据位中1的个数是多少（奇偶计数），奇偶校验位永远都是逻辑0

图 6.10-11 UART 线控制校验位的设置

UART 自动流控功能 (仅UART0支持)

UART0支持自动流控功能，该功能用到两根信号线CTS (清零发送) 和RTS (请求发送)来控制UART与外部设备（如Modem）间的数据传输。当自动流控使能后，只有等到UART对外部设备发出有效的RTS信号后才允许接收数据，否则不接收。当RX FIFO接收到数据的数量达等于RTSTRGLV (UART_FIFO [19:16])位的值后，RTS信号会被取消。当UART0检测到外部设备给CTS信号脚有效信号后，UART0开始发送数据，否则 UART0不会发送数据。

下图 6.10-12展示了UART0 CTS自动流控功能模型。用户必须要先设置ATOCTSEN (UART_INTEN [13])来使能CTS自动流控功能。CTSACTLV (UART_MODEM [8])位可以设置CTS管脚为输入的有效状态。CTS管脚上任何电平变化将导致CTSDETF (UART_MODEMSTS [0])位被置1，然后TX数据将从TX FIFO自动发出。

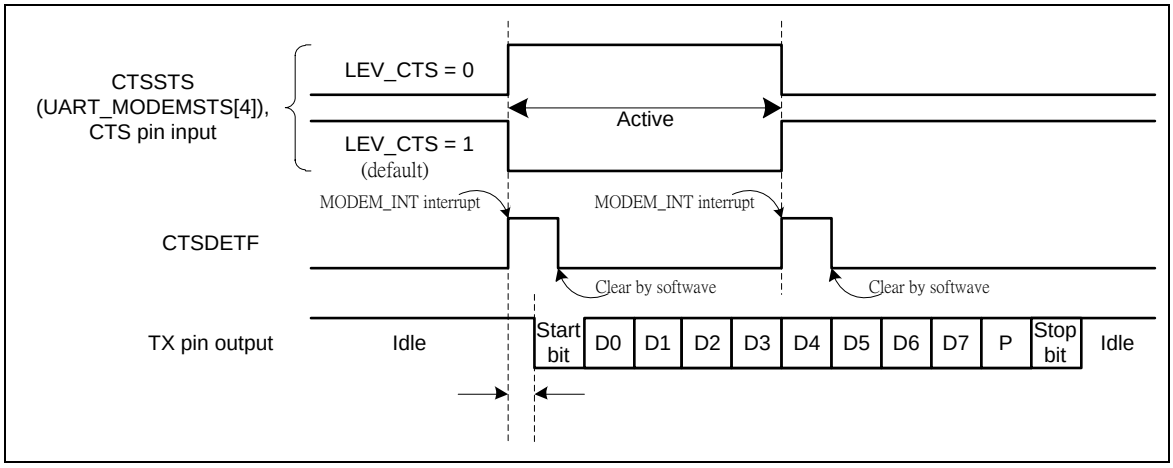


图 6.10-12 CTS 自动流控使能

如下图 6.10-13所示，UART0 RTS自动流控模式ATORTSEN(UART_INTEN[12])=1)中，nRTS内部信号的触发水平由UART0 FIFO控制器的寄存器RTS_RTI_LEV(UART_FIFO[19:16])位来控制。

设置RTSACTLV(UART_MODEM[9])位可以nRTS信号控制RTS 管脚的输出是反向或非反向。用户可以读RTSSTS(UART_MODEM[13])位来知道真实RTS脚输出电压的逻辑状态。

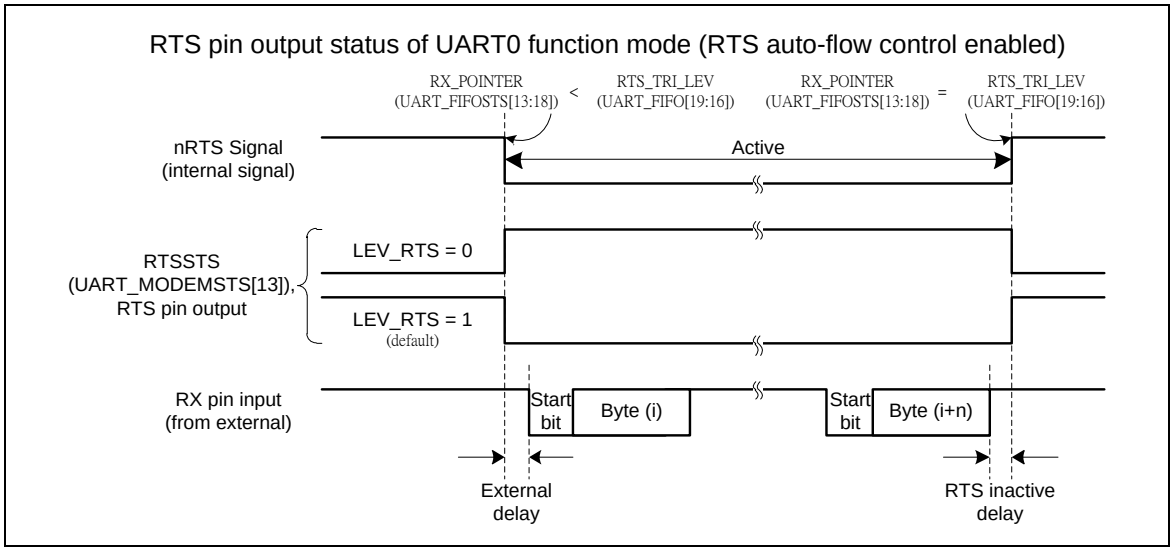


图 6.10-13 RTS 自动流控使能

如下图 6.10-14所示，在软件模式下(ATORTSEN(UART_INTEN[12])=0)，RTS流控直接由RTS(UART_MODEM[1])位来控制。

设置RTSACTLV(UART_MODEM[9])位可以控制RTS输出脚状态，RTS(UART_MODEM[1])可控制输出是同向还是反向。用户可以读RTSSTS(UART_MODEM[13])位来获取RTS脚的真实输出电平状态。

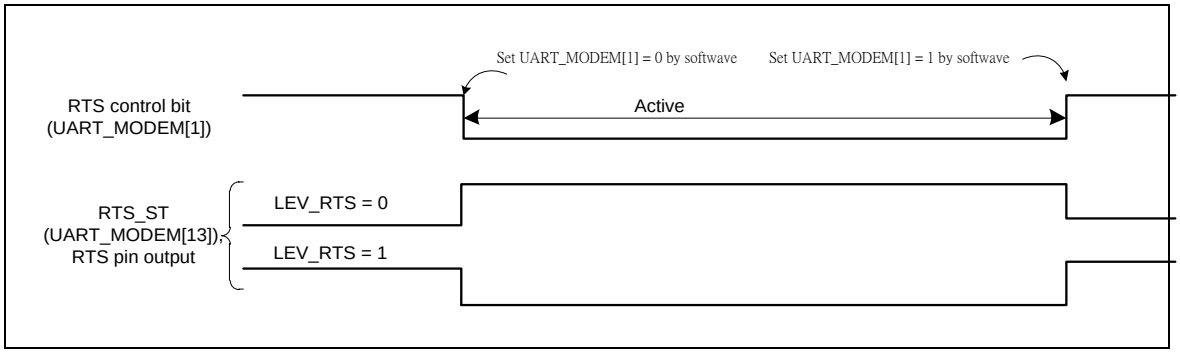


图 6.10-14 RTS 软件控制的流控

6.10.5.6 IrDA 功能模式(仅UART0支持)

UART0控制器也提供Serial IrDA (SIR, 串行红外)功能(用户必须设置IrDA_EN和 (UART_FUNSEL[1:0])为'10')来使能IrDA功能。SIR规范定义了一个短距离红外异步串行传输模式，包括一个起始位，8个数据位，一个停止位。最大速率115.2kbps。IrDA SIR模块包含一个IrDA SIR协议编/解码器。IrDA SIR协议是半双工工作模式。所以它不能同时收发数据。IrDA SIR物理层规定了发送与接收数据的时间上至少10ms的时间间隔，该延迟特性需通过软件来完成。

IrDA 模式下，寄存器BAUDM1 (UART_BAUD [29])位需被禁止。

波特率= Clock / (16 * BRD)，这里BRD是UART_BAUD寄存器中定义的波特率分频器。

以下图 6.10-15指示了IrDA控制模块框图：

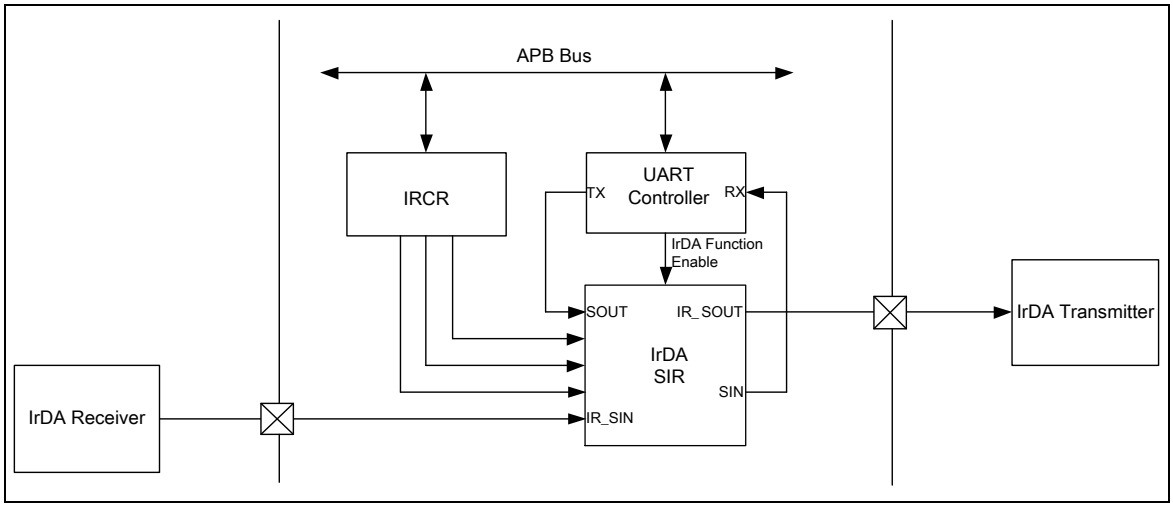


图 6.10-15 IrDA 控制器模块框图

IrDA SIR 发送编码

IrDA SIR 传送编码调制采用 Non-Return-to Zero (NRZ)编码，将数据流从 UART0输出。IrDA SIR 物理层指定使用归零反向调制编码 (Return-to-Zero, Inverted (RZI))，用一个红外光脉冲代表逻辑 0，被调整的脉冲输出到外部输出驱动器和红外线发光二极管。

在正常模式下，传输脉冲的宽度为3/16 波特率周期。

IrDA SIR 接收解码

IrDA SIR 接收解码器对输入管脚的(Return-to-Zero, Inverted (RZI))串行位流进行解调，并输出NRZ 串行位流到 UART0接收数据输入端。在空闲状态里，解码器输入端通常为高。（因此，RXINV (UART_IRCR[6])位默认设为1）。

当解码器输入端为低时，表明接收到一个起始位。

IrDA SIR 操作

IrDA SIR 编码/解码提供 UART 数据流和半双工串行 SIR 之间的转换。IrDA编码/解码波形图如图 6.10-16：

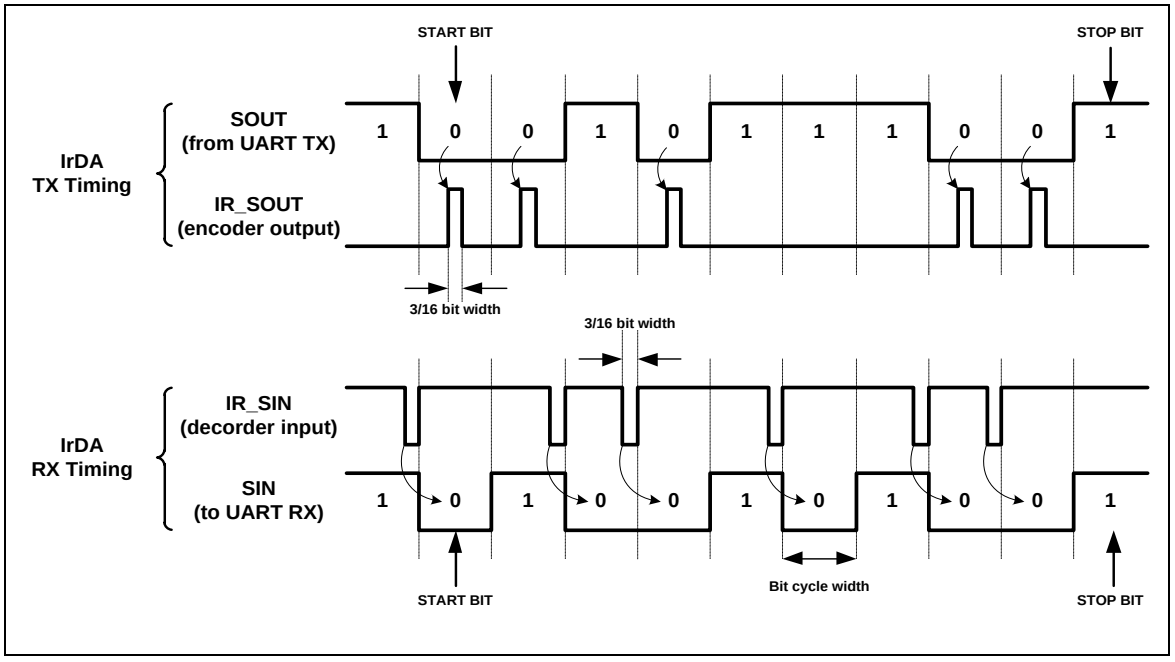


图 6.10-16 IrDA TX/RX 时序图

6.10.5.7 RS-485 功能模式(仅UART0支持)

UART控制器另一个可选择的功能是RS-485功能（用户必须设置UART_FUNSEL [1:0]为 “11”来使能RS-485功能），方向控制则RTS脚或可编程的GPIO(P0.3为RTS0和P0.1为RTS1)通过软件来控制。RS-485收发器的驱动控制是通过RTS控制信号来驱动的。RS-485模式下的RX和TX大多数特性与UART相同。

RS-485模式，控制器可以配置成 RS-485 可寻址的从机模式，RS-485 主机发送可通过设置奇偶检验位 (第9位) 为 1标识地址特性。对于数据特性，奇偶检验位设置为 0。软件可以设置寄存器UART_LINE 来控制第9位（当PBE, EPE 和 SPE置位时，第9位发送 0；当PBE 和 SPE 置位，EPE清零时，第 9 位发送1）。

该控制器支持三种操作模式：RS-485 普通多点操作模式 (NMM)，RS-485 自动地址识别模式 (AAD) 和 RS-485 自动方向控制模式 (AUD)，可通过UART_ALTCTL寄存器的设置选择其中一种工作模式，通过设置DLY (UART_TOUT [15:8])寄存器可以设置上一个停止位与下一个开始位之间的延迟时间。

RS-485 普通多点操作模式(NMM)

RS-485 普通多点操作模式，首先，软件必须决定在检测到地址字节之前的数据是否存储到 RX FIFO。如果软件想忽略在检测到地址之前的任何数据，流程是设置RXOFF (UART_FIFO [8])，然后使能RS485_NMM (UART_ALTCTL [8])，接收器将会忽略数据，直到检测到地址字节 (bit 9 =1) 并且地址字节数据存储到RX FIFO。如果软件想接收在检测到地址字节之前的任何数据，流程是禁用RXOFF (UART_FIFO [8])，然后使能RS485_NMM (UART_ALTCTL [8])，接收器将接收任何数据。

如果检测到地址字节 (bit 9 =1)，会产生一个中断到 CPU，软件可以通过设置RXOFF (UART_FIFO [8])，决定是否使能或禁用接收器接收数据。如果使能接收器，就会接收所有字节数据并存储到 RX-FIFO。如果设置RXOFF (UART_FIFO [8])位禁用接收器，会忽略所有接收到的字节数据，直到检测到下一个地址字节。当检测到地址字节后，控制器将清除RXOFF (UART_FIFO [8])位且地址字节数据将存储到 RX FIFO。

RS-485 自动地址识别工作模式(AAD)

RS-485 自动地址识别模式，接收器在检测到地址字节 (bit 9=1) 并且地址字节数据与ADDRMV (UART_ALTCTL[31:24])的值相匹配之前将忽略所有数据。地址字节数据将存储在 RX FIFO。所有接收字节数据将被接收并存储于 RX FIFO 直到地址字节数据与ADDRMV (UART_ALTCTL[31:24])的值不匹配。

RS-485 自动方向模式(AUD)

RS-485 控制器的另一个功能是 RS-485 自动方向控制功能。必须设置RS485_AUD(UART_ALTCTL[10])=1来使能RS485为自动方向模式。通过异步串口的RTS控制信号来使能RS-485 收发器。RTS线 连接到RS-485 收发器，设置RTS线为高（逻辑1）使能RS-485 收发器。设置 RTS 为低（逻辑0），使收发器进入 tri-state 状态。用户通过设置寄存器UART_MODEM中的RTSACTLV 位改变 RTS 的驱动电平

以下图 6.10-17 展示了AUD模式下RS-485 RTS的驱动电平。RTS管脚在TX数据发送阶段将被自动驱动。

设置RTSACTLV(UART_MODEM[9])位可以控制RTS脚的输出电平。用户可以通过读RTSSTS(UART_MODEM[13])位来知道RTS脚上实际的输出逻辑电平。

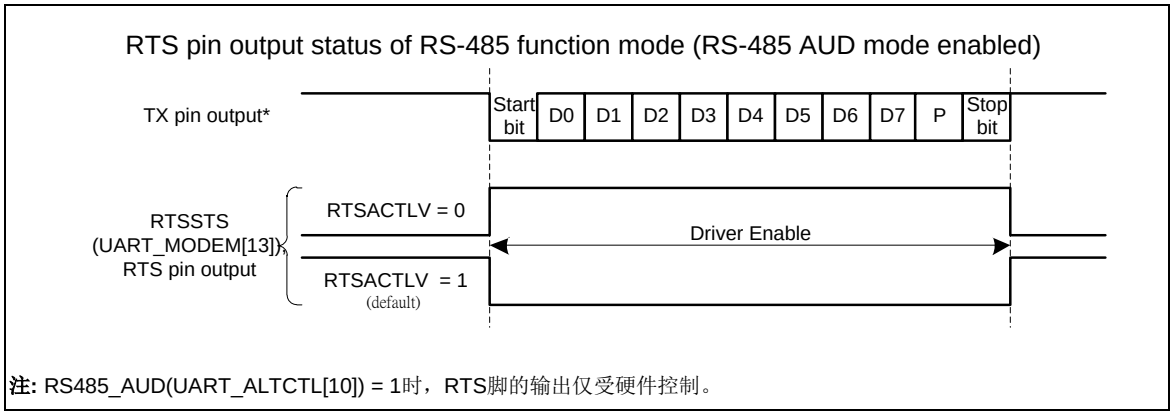


图 6.10-17 RS-485 自动方向模式下通过RTS 驱动电平

图 6.10-18展示了通过软件控制(RS485_AUD(UART_ALTCTL[10])=0) RS-485 RTS管脚的驱动电平。RTS驱动电平通过设置RTS(UART_MODEM[1])位来控制。

设置RTSACTLV(UART_MODEM[9])位可以控制RTS管脚的输出，通过设置RTS(UART_MODEM[1])可以控制输出是否反向。用户可以读RTSSTS(UART_MODEM[13])位来知道RTS管脚上实际的逻辑电平。

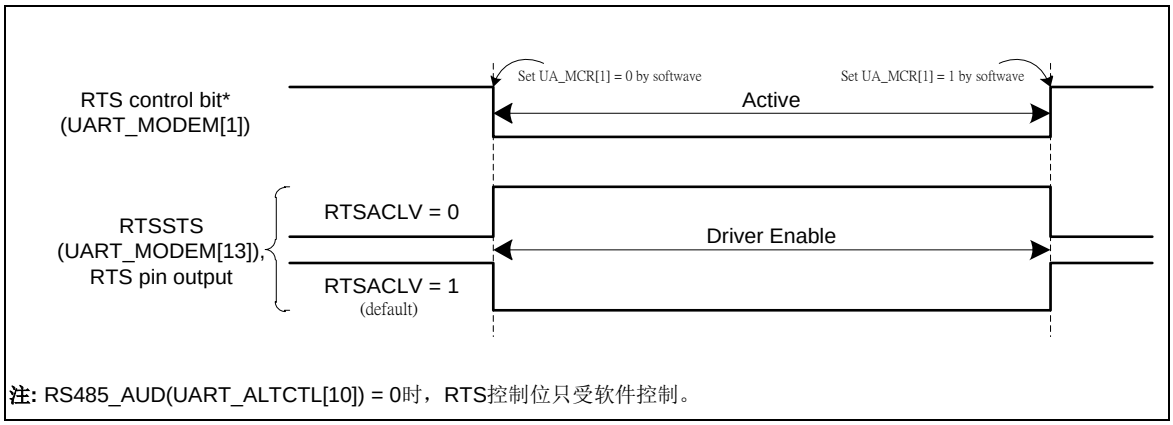


图 6.10-18 RS-485 软件控制下的RTS 驱动电平

编程流程示例：

1. 设置UART_FUNSEL (UART_FUNSEL [1:0]) 为 “11”来选择UART为RS-485 功能
2. 设置RXOFF (UART_FIFO[8])位决定使能或禁用 RS-485 接收器
3. 设置RS-485_NMM (UART_ALTCTL[8]) 或RS-485_AAD 模式
4. 如果选择RS-485_AAD (UART_ALTCTL[9])模式 ，ADDRMV(UART_ALTCTL[31:24])需设置成自动地址匹配值
5. 设置RS-485_AUD (UART_ALTCTL[10])来决定是否为自动方向控制

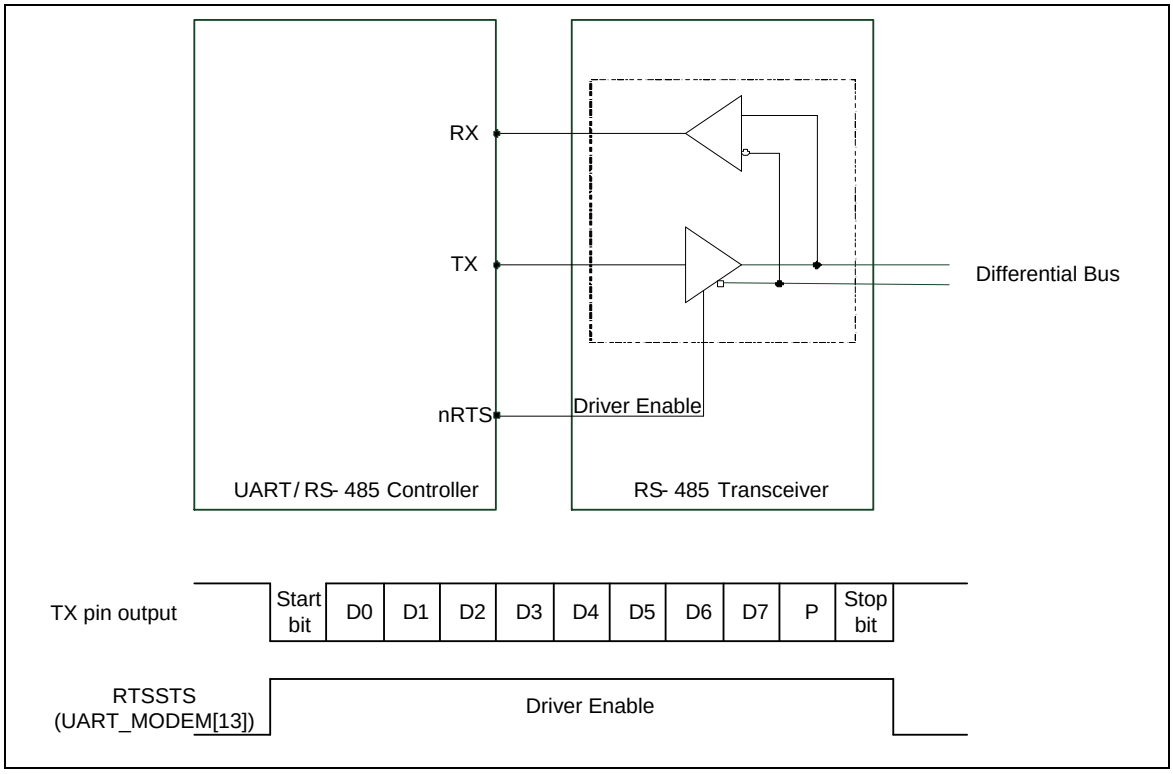


图 6.10-19 RS-485 帧结构

6.10.6 寄存器表

R: 只读, **W:** 只写, **R/W:**可读写

寄存器	偏移地址	R/W	描述	复位值
UART 基地址: **UART0_BA = 0x4005_0000** **UART1_BA = 0x4015_0000**				
UART_DAT **x = 0, 1**	UARTx_BA+0x00	R/W	UART 接收/发送缓冲寄存器	未定义
UART_INTEN **x = 0, 1**	UARTx_BA+0x04	R/W	UART 中断使能寄存器	0x0000_0000
UART_FIFO **x = 0, 1**	UARTx_BA+0x08	R/W	UART FIFO 控制寄存器	0x0000_0101
UART_LINE **x = 0, 1**	UARTx_BA+0x0C	R/W	UART 线控制寄存器	0x0000_0000
UART_MODEM **x = 0**	UARTx_BA+0x10	R/W	UART Modem 控制寄存器	0x0000_0200
UART_MODEMSTS **x = 0**	UARTx_BA+0x14	R/W	UART Modem 状态寄存器	0x0000_0110
UART_FIFOSTS **x = 0, 1**	UARTx_BA+0x18	R/W	UART FIFO 状态寄存器	0x1040_4000
UART_INTSTS **x = 0, 1**	UARTx_BA+0x1C	R/W	UART Interrupt 状态寄存器	0x0000_0002
UART_TOUT **x = 0, 1**	UARTx_BA+0x20	R/W	UART 定时溢出寄存器	0x0000_0000
UART_BAUD **x = 0, 1**	UARTx_BA+0x24	R/W	UART 波特率分频寄存器	0x0F00_0000
UART_IRDA **x = 0**	UARTx_BA+0x28	R/W	UART IrDA 控制寄存器	0x0000_0040
UART_ALTCTL **x = 0**	UARTx_BA+0x2C	R/W	UART 选择控制/状态寄存器	0x0000_0000
UART_FUNSEL **x = 0, 1**	UARTx_BA+0x30	R/W	UART 功能选择寄存器	0x0000_0000

6.10.7 寄存器描述

接收/发送缓冲区寄存器 (UART_DAT)

寄存器	偏移地址	R/W	描述	复位值
UART_DAT **x = 0, 1**	UARTx_BA+0x00	R/W	UART 接收/发送缓冲区寄存器	未定义

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

位	描述	
[31:8]	保留	保留
[7:0]	**DAT**	**接收/发送缓冲区寄存器** 写操作： 写一字节数据到该寄存器，数据将会保存到发送FIFO。 UART控制器将会通过UART_TXD管脚把存放在FIFO里的最前面的数据发送出去 读操作： 读该寄存器，UART 将返回接收到的来自接收缓存的8位数据。

中断使能寄存器 (UART_INTEN)

寄存器	偏移地址	R/W	描述	复位值
UART_INTEN **x = 0, 1**	UARTx_BA+0x04	R/W	UART 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留		**ATOCTSEN**	**ATORTSEN**	**TOCNTEN**	保留	**WKCTSIEN**	保留
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留		**BUFERRIEN**	**RXTOIEN**	**MODEMIEN**	**RLSIEN**	**THREIEN**	**RDAIEN**

位	描述	
[31:14]	描述	保留
[13]	**ATOCTSEN**	**CTS自动流控使能位 (UART0有效)** 0 = CTS自动流控禁止. 1 = CTS自动流控使能. **注：** 当 CTS 自动流控使能后，当CTS输入信号有效时， UART 将发送数据到外部设备（UART将不会发送数据到外部设备直到CTS有效）。
[12]	**ATORTSEN**	**RTS 自动流控使能位(UART0有效)** 0 = RTS自动流控禁止 1 = RTS自动流控使能 **注：** 当 RTS 自动流控使能后，如果RX FIFO中字节的数量等于RTSTRGLV (UART_FIFO [19:16])，UART会自动禁止RTS信号
[11]	**TOCNTEN**	**定时溢出计数器使能位** 0 = 定时溢出计数器禁止 1 = 定时溢出计数器使能.
[10]	保留	保留
[9]	**WKCTSIEN**	**唤醒CPU功能中断使能位 (UART0有效)** 0 = UART唤醒功能禁止 1 = UART唤醒功能使能 **注:** 当系统在Power-down模式时，CTS管脚电平的改变可以将系统从Power-down模式唤醒。
[8:6]	保留	保留
[5]	**BUFERRIEN**	**缓存错误中断使能位** 0 = 屏蔽缓存错误中断禁止

位	描述	
		1 = 屏蔽缓存错误中断使能
[4]	**RXTOIEN**	**RX 定时溢出中断使能位** 0 = RXTOINT 中断关闭 1 = RXTOINT 中断使能
[3]	**MODEMIEN**	**Modem 状态中断使能位** 0 = MODEMINT 状态中断关闭 1 = MODEMINT 状态中断使能
[2]	**RLSIEN**	**接收线状态中断使能位** 0 = RLSINT 关闭. 1 = RLSINT 使能
[1]	**THREIEN**	**发送保持寄存器空中断使能位** 0 = THREINT 关闭 1 = THREINT 使能
[0]	**RDAIEN**	**接收数据有效中断使能位** 0 = RDAINT 关闭 1 = RDAINT 使能

FIFO 控制寄存器 (UART_FIFO)

寄存器	偏移地址	R/W	描述	复位值
UART_FIFO **x = 0, 1**	UARTx_BA+0x08	R/W	UART FIFO 控制寄存器	0x0000_0101

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

位	描述	
[31:20]	保留	保留
[19:16]	**RTSTRGLV**	**RTS 自动流控触发阈值 (UART0有效)** 0000 = RTS 触发阈值为1 byte. 0001 = RTS 触发阈值为4 bytes. 0010 = RTS 触发阈值为8 bytes. 0011 = RTS 触发阈值为14 bytes. 其它 = 保留. **注：**该区域用于RTS自动流控控制
[15:9]	保留	保留
[8]	**RXOFF**	**接收器禁止设置位(UART0有效)** 是否禁用接收器（置 1 禁用接收器） 0 =接收器使能 1 =接收器禁止 **注：**该位用于 RS-485 普通多点模式。需要在RS-485_NMM (UART_ALTCTL [8])之前设置
[7:4]	**RFITL**	**RX FIFO 中断 (RDAINT) 触发级别 (UART0有效)** 当FIFO 接收字节数等于RFITL后，RDAIF将被置位（如果UART_INTEN寄存器的RDAIEN使能，将会产生中断）。 0000 = RX FIFO 触发中断阈值为 1 byte. 0001 = RX FIFO 触发中断阈值为 4 bytes 0010 = RX FIFO 触发中断阈值为 8 bytes. 0011 = RX FIFO 触发中断阈值为14 bytes. 其它 = 保留
[3]	保留	保留

位	描述	
[2]	**TXRST**	**TX 域软件复位** 当TX_RST置位， 发送 FIFO 内的所有数据和 TX 内部状态机将被清除。 0 = 不影响. 1 = 该位写 1 将复位 TX 内部状态机和指针 **注：**该位至少需要3个 UART 时钟周期后才会自动清零
[1]	**RXRST**	**RX 域软件复位** 当RX_RST被置位， 接收 FIFO 内的所有数据和 RX内部状态机将被清除。 0 = 不影响. 1 = 该位写 1 将复位 RX 内部状态机和指针. **注：**该位至少需要 3个 UART 时钟周期后才会自动清零。
[0]	**保留**	保留

线控制寄存器 (UART_LINE)

寄存器	偏移地址	R/W	描述	复位值
UART_LINE **x = 0, 1**	UARTx_BA+0x0C	R/W	UART 线控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留	**BCB**	**SPE**	**EPE**	**PBE**	**NSB**	**WLS**	

位	描述	
[31:7]	保留	保留
[6]	**BCB**	**Break 控制位** 当该位被置逻辑 1，串行数据输出 (TX) 将强制到 Spacing 状态 (logic 0)。该位仅作用于 TX，对传输逻辑不起作用。
[5]	**SPE**	**Stick 奇偶校验使能位** 0 = 禁用 Stick 奇偶校验 1 = 如果PBE (UART_LINE[3]) 和EPE (UART_LINE[4])都为逻辑 1，奇偶校验位发送和检验值为逻辑 0。如果PBE (UART_LINE[3])是1，EPE (UART_LINE[4])是 0，则奇偶校验位发送和检验值为 1。
[4]	**EPE**	**偶校验使能位** 0 = 逻辑 1的奇数数目在每个字节中被发送和检验 1 = 逻辑 1的偶数数目在每个字节中被发送和检验 该位只在PBE (UART_LINE[3])置位时有效
[3]	**PBE**	**校验位使能** 0 = 无奇偶校验位 1 = 每一个发送字符中都产生奇偶校验位，对每一个传进来的数据进行奇偶校验位检测
[2]	**NSB**	**“停止位”个数** 0 = 当发送数据时，产生 1 “ STOP bit” 1 = 当发送数据，选择 5-位 字长度时，产生 1.5个 “ STOP bit” 。当选择 6-, 7- 和 8-位字长度时，产生 2 个“STOP bit”
[1:0]	**WLS**	**字长度选择位** 00 = 字长是 5-bit 01 = 字长是 6-bit 10 = 字长是 7-bit

位	描述	
		11 = 字长是 8-bit

MODEM 控制寄存器(UART_MODEM) (UART0有效)

寄存器	偏移地址	R/W	描述	复位值
UART_MODEM **x = 0**	UARTx_BA+0x10	R/W	UART Modem 控制寄存器	0x0000_0200

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

位	描述	
[31:14]	保留	保留
[13]	**RTSSTS**	**RTS管脚状态(只读)** 该位的值对应于RTS管脚的输出电平逻辑状态 0 = RTS管脚为低电平逻辑状态 1 = RTS管脚为高电平逻辑状态.
[12:10]	保留	保留
[9]	**RTSACTLV**	**RTS 管脚的有效电平** 该位定义了RTS输出管脚的有效电平 0 = RTS 管脚输出高电平有效 1 = RTS 管脚输出低电平有效(默认值) **注1:** UART 功能模式请参考 图 6.10-13和图 6.10-14**注2:** RS-485 功能模式请参考图 6.10-17和图 6.10-18
[8:2]	保留	保留
[1]	**RTS**	**RTS (请求发送) 信号控制** 该位直接控制内部RTS管脚信号是否有效，然后使用RTSACTLV 位的配置驱动来RTS管脚输出 0 = RTS 信号有效. 1 = RTS 信号无效. **注1:** UART 功能模式下，当RTS自动流控被使能后，RTS信号控制位无效 **注2:** RS-485模式下，当RS-485自动方向模式(AUD)被使能后，RTS信号控制位无效
[0]	保留	保留

Modem 状态寄存器(UART_MODEMSTS) (UART0有效)

寄存器	偏移地址	R/W	描述	复位值
UART_MODEMSTS **x = 0**	UARTx_BA+0x14	R/W	UART Modem 状态寄存器	0x0000_0110

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

位	描述	
[31:9]	保留	保留
[8]	**CTSACTLV**	**CTS 管脚有效电平** 该位定义了CTS输入管脚的有效电平状态 0 = CTS输入管脚高电平有效 1 = CTS输入管脚低电平有效(默认状态)
[7:5]	保留	保留
[4]	**CTSSTS**	**CTS 管脚状态(只读)** 该位对应于CTS管脚输入电平逻辑状态 0 = CTS管脚输入状态为低电平 1 = CTS管脚输入状态为高电平 **注:**当UART控制器外围时钟被使能，且CTS功能被使能，该位才有效。
[3:1]	保留	保留
[0]	**CTSDETF**	**检测CTS 管脚电平改变标志 (只读)** 如果CTS输入管脚上有电平变化，该位将被置位，如果MODEMIEN (UART_INTEN [3])位被置位，将会产生Modem中断。 0 = CTS 输入管脚没有电平变化 1 = CTS 输入管脚有电平变化 **注:** 该位 写 “1” 清零

FIFO 状态寄存器 (UART_FIFOSTS)

寄存器	偏移地址	R/W	描述	复位值
UART_FIFOSTS **x = 0, 1**	UARTx_BA+0x18	R/W	UART FIFO 状态寄存器	0x1040_4000

31	30	29	28	27	26	25	24
保留			TXEMPTYF	保留			TXOVIF
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXFULL	TXEMPTY	TXPTR					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXFULL	RXEMPTY	RXPTR					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留	BIF	FEF	PEF	ADDRDETF	保留		RXOVIF

位	描述	
[31:29]	保留	保留
[28]	**TXEMPTYF**	**发送器空标志(只读)** 当TX FIFO (UART_DAT)为空，并且最后一个字节的STOP位也已经被发送完毕，那么该位将被置1 0 = TX FIFO 不为空 1 = TX FIFO 为空 **注：**当TX FIFO 不为空或最后一个字节没有全部被发送完，此位自动清零
[27:25]	保留	保留
[24]	**TXOVIF**	**TX 溢出错误中断标志** 如果TX FIFO(UART_DAT)满，此时如果再向UART_DAT写入数据，将会导致此位被置1. 0 = TX FIFO 未溢出 1 = TX FIFO 已溢出. **注：**此位可以写1清零。
[23]	**TXFULL**	**发送FIFO满 (只读)** 此位用于指示TX FIFO是否已满. 0 = TX FIFO 未满 1 = TX FIFO 已满 **注：**当TX FIFO 缓存中的字节数量等于1，此位将被置1。否则被硬件清零
[22]	**TXEMPTY**	**发送FIFO 空 (只读)** 此位指示TX FIFO是否为空 0 = TX FIFO 不为空 1 = TX FIFO 为空 **注意：**当TX FIFO中的最后一个字节被传送到发送移位寄存器时，硬件将把此位置1。当写

位	描述	
		入数据到THR中 (TX FIFO不为空)，此位被清零
[21:16]	**TXPTR**	**TX FIFO 指针(只读)** 此位指示TX FIFO缓冲区指针位置。当CPU写一个字节到UART_DAT寄存器时，TXPTR将累加1。当TX FIFO发送一个字节到发送移位寄存器中，TXPTR指针将减1。 TXPTR显示的最大值是15。当TX FIFO缓冲区所填充数据数量达到16时，TXFULL位将被置1，而且TXPTR将为零。此时，如果TX FIFO中发送一个字节到发送移位寄存器，TXFULL位将被清零，TXPTR将显示为15。
[15]	**RXFULL**	**接收 FIFO 满 (只读)** 该位指示RX FIFO 是否已满 0 = RX FIFO 未满 1 = RX FIFO 已满 **注意：**当RX FIFO缓冲区中的数据数量等于16 后，此位将被置1，否则被硬件清零。
[14]	**RXEMPTY**	**接收 FIFO 空 (只读)** 此位指示RX FIFO是否为空 0 = RX FIFO不为空. 1 = RX FIFO为空. **注意：**当RX FIFO中最后一个字节被CPU读取后，硬件将对此位置1，UART接收到新数据后此位将被清零。
[13:8]	**RXPTR**	**RX FIFO 指针(只读)** 此位指示RX FIFO缓冲区指针。当UART从外部设备接收到一个字节，RXPTR将累加1。当RX FIFO的数据被CPU读取一个字节，RXPTR将递减1。 RXPTR显示的最大值是15。当RX FIFO的数据达到16时，RXFULL位将被置1，RXPTR显示0。此时，如果RX FIFO 当中的数据被CPU读取一个后，RXFULL 将被清零，RXPTR显示15。
[7]	**保留**	保留
[6]	**BIF**	**Break中断标志位(只读)** 每当接收到数据输入 (RX) 维持在 “spacing state” (logic 0) 的时间长于一个全字的传输时间（即“start bit” + data bits + parity + stop bits 的总时间），该位置 1。 0 = 没有Break中断产生. 1 = 有Break中断产生. **注：**此位只读。但是软件可以写1清零
[5]	**FEF**	**帧错误标志(只读)** 每当接收到的字符没有有效的 “停止位”（即跟着最后的数据位或奇偶校验位的停止位检测为0）该位置 1 0 = 没有帧错误产生 1 = 有帧错误产生 **注：**该位只读，但是可以写 1 清零。
[4]	**PEF**	**奇偶校验错误标志(只读)** 每当接收到的字符没有有效的奇偶校验位时，该位置 1 0 = 没有奇偶校验错误产生 1 = 有奇偶校验错误产生 **注：**该位只读，但是可以写 1 清零。
[3]	**ADDRDETF**	**RS-485 地址字节数据检测标志 (UART0有效)**

位	描述	
		设置ADDRDEN (UART_ALTCTL[15])为1使能地址检测模式，当接收到带地址位(bit 9 =1)的数据时，该位将被置1。 **注1**：此域适用于RS-485功能模式 **注2**：该位可以写 1 清零。
[2:1]	**保留**	保留
[0]	**RXOVIF**	**RX 溢出错误中断标志** 此位当RX FIFO溢出时被置1 如果接收到的数据字节数量大于 RX_FIFO (UART_DAT) 的大小16个 字节，该位将被置位 0 = RX FIFO 未溢出. 1 = RX FIFO 溢出 **注**：该位可以写 1 清零

中断状态控制寄存器 (UART_INTSTS)

寄存器	偏移地址	R/W	描述	复位值
UART_INTSTS x = 0, 1	UARTx_BA+0x1C	R/W	UART 中断状态寄存器	0x0000_0002

31	30	29	28	27	26	25	24
保留							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留							**CTSWKIF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留		**BUFERRINT**	**RXTOINT**	**MODEMINT**	**RLSINT**	**THREINT**	**RDAINT**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留		**BUFERRIF**	**RXTOIF**	**MODEMIF**	**RLSIF**	**THREIF**	**RDAIF**

位	描述	
[31:17]	保留	保留
[16]	**CTSWKIF**	**nCTS 唤醒中断标志（只读）(UART0有效)** 0 = 芯片保持在power-down状态 1 = nCTS信号将芯片从power-down状态唤醒 **注1**：如果WKCTSIEN (UART_IER[9])被使能，将会产生唤醒中断 **注2**：该位为只读位，但可以通过写'1'清零
[15:14]	保留	保留
[13]	**BUFERRINT**	**缓存错误中断标志 (只读)** 如果BUFERRIEN (UART_INTEN[5])和BUFERRIF (UART_INTSTS[5])都被置1，该位置1 0 = 没有缓存错误中断产生. 1 = 有缓存错误中断产生
[12]	**RXTOINT**	**定时溢出中断标志 (只读)** 如果RXTOIEN (UART_INTEN[4])和RXTOIF (UART_INTSTS[4])都被置1，该位置 1 0 = 没有定时溢出中断产生 1 = 有定时溢出中断产生.
[11]	**MODEMINT**	**MODEM状态中断标志(只读)** 如果MODEMIEN (UART_INTEN[3])和MODENIF (UART_INTSTS[3])都被置1，该位置 1 0 = 没有Modem 中断产生. 1 = 有Modem 中断产生
[10]	**RLSINT**	**接收线状态中断标志 (只读)** 如果RLSIEN (UART_INTEN[2])和RLSIF (UART_INTSTS[2])都被置1，该位置 1 0 = 没有RLS中断产生 1 = 有RLS中断产生.

位	描述	
[9]	**THREINT**	**发送保持寄存器空中断标志 (只读)** 如果THREIEN (UART_INTEN[1]) 和 THREIF (UART_INTSTS[1])都被置1，该位置 1 0 = 没有THRE中断产生 1 = 有THRE中断产生
[8]	**RDAINT**	**接收有效数据中断标志 (只读)** 如果RDAIEN (UART_INTEN[0])和RDAIF (UART_INTSTS[0])都被置1，该位置 1 0 = 没有RDA中断产生. 1 = 有RDA中断产生.
[7:6]	保留	保留
[5]	**BUFERRIF**	**缓存错误中断标志 (只读)** 当TX FIFO 或 RX FIFO溢出标志 (TXOVIF (UART_FIFOSTS[24]或 RXOVIF(UART_FIFOSTS[0])置位，此时该位置1。 当BUFERRIF (UART_INTSTS[5])被置位，传输出错。如果BUFERRIEN (UART_INTEN[5])被使能, 缓存错误中断将产生 0 = 没有缓存 错误中断标志产生 1 = 有缓存 错误中断标志产生 **注:** 此位只读。但当TXOVIF (UART_FIFOSTS[24])和RXOVIF (UART_FIFOSTS[0]) 都被清零时，该位被清零.
[4]	**RXTOIF**	**定时溢出中断标志 (只读)** 当 RX FIFO 非空而且 RX FIFO无活动发生，当定时溢出计数器计数等于TOIC (UARTTOUT[7:0])时，该位被置位。如果RXTOIEN (UART_INTEN [4])使能，将产生定时溢出中断。 0 = 没有定时溢出中断标志产生 1 = 有定时溢出中断标志产生 **注：**该位只读，用户可以读UART_DAT(RX处于活动状态)清除该位
[3]	**MODEMIF**	**MODEM 中断标志(只读) (UART0有效)** 当CTS管脚有状态改变 (CTSDETF (UART_MODEMSTS[0]) = 1)，该位被置位。如果MODEMIEN (UART_INTEN[3])被使能，将产生Modem 中断。 0 = 没有Modem 中断标志产生. 1 = 有Modem 中断标志产生. **注：**该位只读，当向CTSDETF (UART_MODEMSTS[0])写1后，该位清除
[2]	**RLSIF**	**接收线中断标志 (只读)** 当 RX 接收数据有 parity error, framing error 或 break error（至少BIF, FEF 和 PEF 3位中有1位被置）该位置位。如果RLSIEN (UART_INTEN [2]) 使能，将产生RLS 中断 0 = 没有RLS中断标志产生 1 = 有RLS中断标志产生. **注1:** 在 RS-485功能模式时，当"接收检测任一接收到的地址字节字符 (bit9 = '1') 位"时，该位也会置1，同时，ADDRDETF (UART_FIFOSTS[3])位也被置1(UART0有效) **注2:** 该位只读，当BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4]) 和 ADDRDETF (UART_FIFOSTS[3])位都被清除后，该位清除为0(UART0有效)
[1]	**THREIF**	**发送保持寄存器空中断标志(只读)** 当TX FIFO 的最后数据传输到发送移位寄存器时，该位置位。如果THREIEN (UART_INTEN [1])被使能，将产生THRE 中断。

位	描述	
		0 = 没有THRE中断标志产生 1 = 有THRE中断标志产生 **注:** 该位只读，当写数据到UART_DAT(TX FIFO 非空)时，该位将被清除
[0]	**RDAIF**	**接收有效数据中断标志 (只读)** 当 RX FIFO 的数据字节数量等于RFITL(UART_FIFO[7:4])时，RDAIF(UART_INTSTS[0])将被置位。如果RDAIEN (UART_INTEN [0])被使能，将产生RDA 中断。 0 = 没有RDA中断标志产生 1 = 有RDA中断标志产生 **注：**该位只读，当 RX FIFO未读数据低于阈值(RFITL (UART_FIFO[7:4]))时，该位将被清除

定时溢出寄存器 (UART_TOUT)

寄存器	偏移地址	R/W	描述	复位值
UART_TOUT x = 0, 1	UARTx_BA+0x20	R/W	UART 定时溢出寄存器	0x0000_0000

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

位	描述	
[31:16]	保留	保留
[15:8]	**DLY**	**TX 延迟时间值** 该域用于设置上一次停止位和下一次开始位之间的传输延迟时间，单位为位时间。
[7:0]	**TOIC**	**定时溢出中断比较器** 当RX FIFO接收到一个新的数据字，定时溢出计数器重置并开始重新计数 (计数时钟 = 波特率)。一旦定时溢出计数器(TOUT_CNT)的内容等于定时溢出中断比较器(TOIC)，如果此时RXTOIEN (UART_INTEN[4])为使能，则接收定时溢出中断(RXTOINT)产生。新来的数据字或RX FIFO 为空清除RXTOINT。为了避免接收定时溢出中断一接收到一个字符就立即产生，TOIC的值必须设置在 40 到 255 之间。例如，如果TOIC为 40，当 UART传输设置为1 停止位和无奇偶校验位时，定时溢出中断将在4个字符没有收到之后产生。

波特率分频寄存器 (UART_BAUD)

寄存器	偏移地址	R/W	描述	复位值
UART_BAUD **x = 0, 1**	UARTx_BA+0x24	R/W	UART 波特率分频寄存器	0x0F00_0000

31	30	29	28	27	26	25	24
保留		**BAUDM1**	**BAUDM0**	**EDIVM1**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BRD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BRD							

位	描述	
[31:30]	保留	保留
[29]	**BAUDM1**	**分频 X 使能位** BRD = 波特率分频值，波特率的公式是 波特率= Clock / [M * (BRD + 2)]，M的默认值是16. 0 = 分频值X禁用 (M = 16). 1 = 分频值X使能 (M = X+1, 但是EDIVM1 [27:24] 必须>= 8). **注:** IrDA模式下，该位须禁止.
[28]	**BAUDM0**	**分频X 等于1** 0 = 分频值M = X (M = X+1, 但是EDIVM1[27:24] 必须>= 8) 1 = 分频值M = 1 (M = 1, 但是BRD [15:0] 必须>= 8) **注:** 详细信息参看章节”UART 控制器波特率发生器”
[27:24]	**EDIVM1**	**分频X** 波特率分频 M = X+1.
[23:16]	保留	保留
[15:0]	**BRD**	**波特率分频器** 该域表示波特率分频

IrDA 控制寄存器(UART_IRDA) (UART0有效)

寄存器	偏移地址	R/W	描述	复位值
UART_IRDA x = 0	UARTx_BA+0x28	R/W	UART IrDA 控制寄存器	0x0000_0040

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

位	描述	
[31:7]	保留	保留
[6]	**RXINV**	**IrDA 反向接收输入信号** 0 = 接收输入信号不反向 1 = 接收输入信号反向(默认)
[5]	**TXINV**	**IrDA 反向发送输出信号** 0 = 信号不反向发送(默认) 1 = 信号反向发送
[4:2]	保留	保留
[1]	**TXEN**	**IrDA 接收/发送选择使能位** 0 = IrDA 发送禁止，接收使能(默认) 1 = IrDA 发送使能，接收禁止
[0]	保留	保留

注: 在 IrDA 模式， BAUDM1 (UART_BAUD[29]) 寄存器必须禁止，波特率公式必须是Clock/16*(BRD+2)

UART 复用控制/状态寄存器 (UART_ALTCTL) (UART0有效)

寄存器	偏移地址	R/W	描述	复位值
UART_ALTCTL x = 0	UARTx_BA+0x2C	R/W	UART 复用控制/状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ADDRMV							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
ADDRDEN	保留				RS485_AUD	RS485_AAD	RS485_NMM
7	6	5	4	3	2	1	0
保留							

位	描述	
[31:24]	**ADDRMV**	**地址匹配值** 该域包括 RS-485 地址匹配值. **注：**该域用于 RS-485 自动地址检测模式
[23:16]	**保留**	保留
[15]	**ADDRDEN**	**RS-485 地址检测使能位** 该位用于使能 RS-485 地址检测使能模式. 0 = 地址检测模式禁止. 1 = 地址检测模式使能. **注：**该位用于 RS-485 各种操作模式.
[14:11]	**保留**	保留
[10]	**RS485_AUD**	**RS-485 自动方向模式 (AUD)控制位** 0 = RS-485自动方向操作模式AUD)禁止. 1 = RS-485自动方向操作模式AUD)使能 **注：**在RS485_NMM 操作模式下无效
[9]	**RS485_AAD**	**RS-485 自动地址检测操作模式 (AAD)** 0 = RS-485自动地址方向操作模式 (AAD)禁止. 1 = RS-485自动地址方向操作模式 (AAD)使能. **注：**在RS485_NMM操作模式下无效
[8]	**RS485_NMM**	**RS-485 普通多点操作模式 (NMM) 控制位** 0 = RS-485 普通多点操作模式 (NMM)禁止. 1 = RS-485 普通多点操作模式 (NMM)使能 **注：**在RS485_AAD 操作模式下无效.

位	描述	
[7:0]	保留	保留

UART 功能选择寄存器(UART_FUNSEL)

寄存器	偏移地址	R/W	描述	复位值
UART_FUNSEL **x = 0, 1**	UARTx_BA+0x30	R/W	UART 功能选择寄存器	0x0000_0000

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

位	描述	
[31:2]	保留	保留
[1:0]	**FUN_SEL**	功能选择使能位 00 = UART功能使能. 01 = 保留 10 = IrDA功能使能(UART0有效) 11 = RS-485功能使能(UART0有效)

6.11 I²C 串行接口控制器 (I²C)

6.11.1 概述

I²C是一种双线双向串行总线，为设备之间的数据交换提供了一种简单有效的连接方式。I²C标准是一个真正的多主机总线，包含了冲突检测和仲裁机制，用来在两个或多个主机同时尝试控制总线的情况下，防止数据损坏。该芯片有两组I²C控制器，只有I²C0支持低功耗下的唤醒功能。

6.11.2 特征

I²C 总线使用两条线 (SDA 和 SCL) 来传送连接在该总线上的设备之间的信息。总线的主要特征为：

- 支持两个 I²C 接口
- 支持主机/从机模式
- 主从机之间双向数据传送
- 多主机总线
- 多主机间同时传输数据仲裁，避免总线上串行数据损坏
- 总线采用串行同步时钟，可实现设备之间以不同的速率传输
- 串行同步时钟可以被用来作为握手机制实现挂起和恢复串行传输
- 内建14位溢出定时器，当I2C总线中止且定时器溢出，产生I2C中断
- 可编程的时钟可以用于多种速率控制
- 支持7位从地址模式
- 支持多地址识别（4组从机地址带mask选项）
- 支持唤醒模式 (仅 I2C0 通道支持该功能)
- 支持两级缓存功能

6.11.3 基本配置

I²C0的基本配置如下：

- I²C 管脚由寄存器 SYS_P3_MFP [13:12] 配置。
- 使能 I²C 时钟 (I2CCKEN) 由寄存器 CLK_APBCLK [8]配置。
- 复位 I²C 控制器 (I2CRST) 由寄存器 SYS_IPRST1 [8] 配置。

I²C1的基本配置如下：

- I²C 管脚由寄存器 SYS_P2_MFP [11:10] 或 SYS_P5_MFP [9:8] 配置。
- 使能 I²C 时钟 (I2CCKEN) 由寄存器CLK_APBCLK [9]配置。
- 复位 I²C 控制器 (I2CRST) 由寄存器SYS_IPRST1 [9]配置。

6.11.4 框图

I²C 控制器的框图如 图 6.11-1.

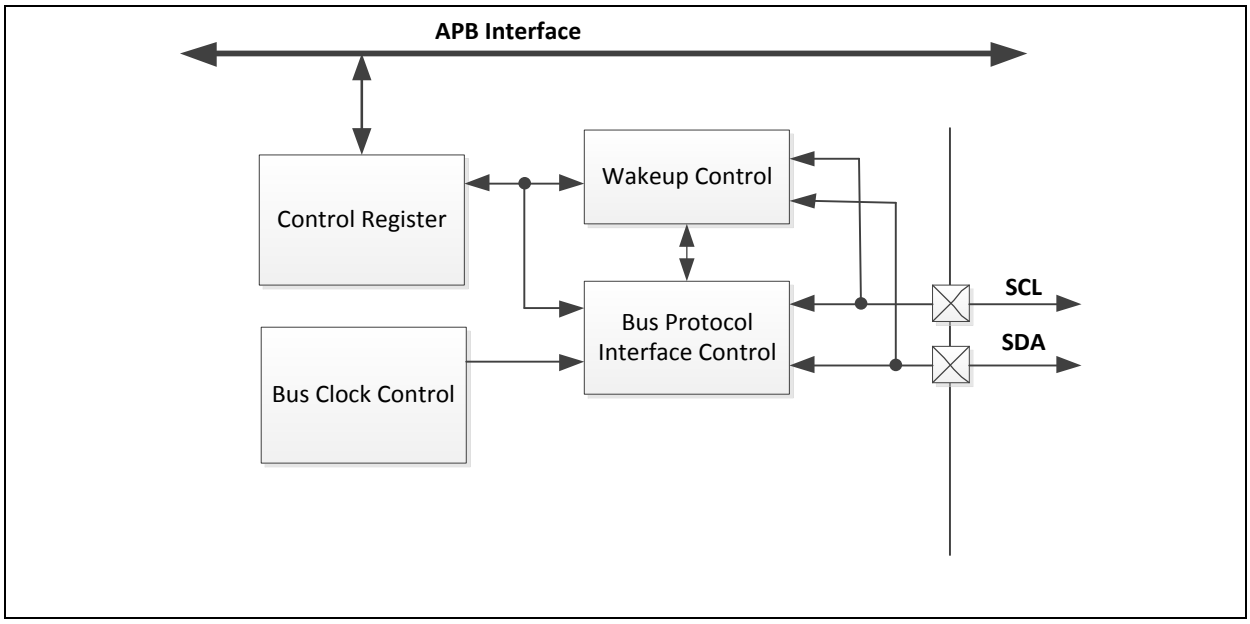


图 6.11-1 I²C 控制器框图

6.11.5 功能描述

在 I²C 总线中，数据是在主机和从机之间传送。数据在SCL和SDA上逐一字节同步传送。每一个数据的字节长度是8位。一个SCL 时钟脉冲传输一个数据位，数据由最高位MSB 开始传输，每个传输字节后跟随一个应答位，每个位在SCL 为高时采样；因此，SDA 线只有在SCL 为低时才可以改变，在SCL 为高时SDA 保持稳定。当SCL 为高时，SDA 线上的跳变视为一个命令(START or STOP)。更多关于I2C总线时序的细节请参考 图 6.11-2。

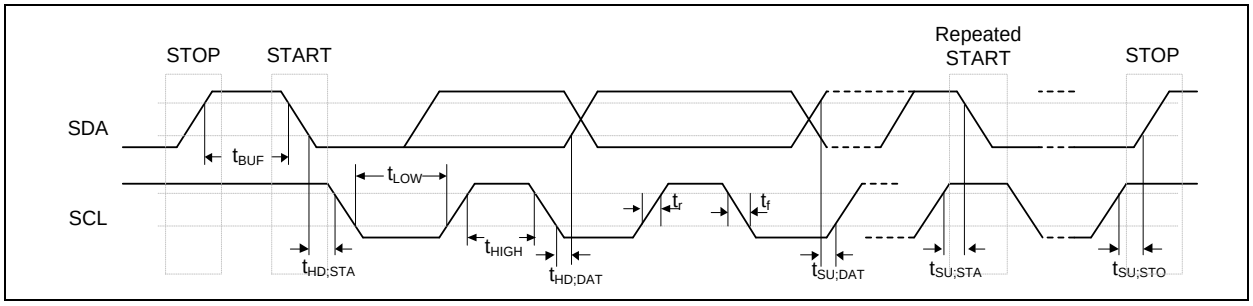


图 6.11-2 I²C 总线时序

该设备片上I²C提供了一个符合I²C总线规范的串行总线接口。I²C端口自动处理字节传输。使能该端口需要将寄存器I2C_CTL的I2CEN位置为'1'。I²C 硬件接口通过管脚SDA和SCL连接到I²C总线上。当I/O管脚作为I²C端口使用时，用户必须事先设定该管脚为I²C功能。

二级缓存提高了I²C总线的性能，在二级缓存模式里，即使在当前数据已被传送的情况或在上一个接收到的数据还未被读回的情况下，下一个传送的数据或上一个接收的数据仍处于活动状态。

当发生SI（I²C中断标志）事件，I²C SCL 总线会被拉低。寄存器I2C_CTL1的NSTRETCH位用来控制在发生SI事件时，I²C SCL 总线上的电平是否被拉低。

当通过TWOLVFIFO(I2C_CTL1[1])位使能二级缓存模式时，如果置位URIEN[4]和OVIEN[3]，就可以使能Under Run中断和Over Run中断。(Under Run指缓存为空时，继续发送数据到总线；Over Run是指接收数据时，缓存出现溢出)

注: 使用I²C时，需要给SDA和SCL引脚增加一组上拉电阻，因为这两个引脚为开漏脚。

6.11.5.1 I²C 协议

图 6.11-3 示意了典型的 I²C 协议。通常情况下，一个标准的通信由4部分组成：

1) 产生起始信号START 或重复起始信号 Repeated START
2) 从机地址和 R/W 位传输
3) 数据传输
4) 产生停止信号 STOP

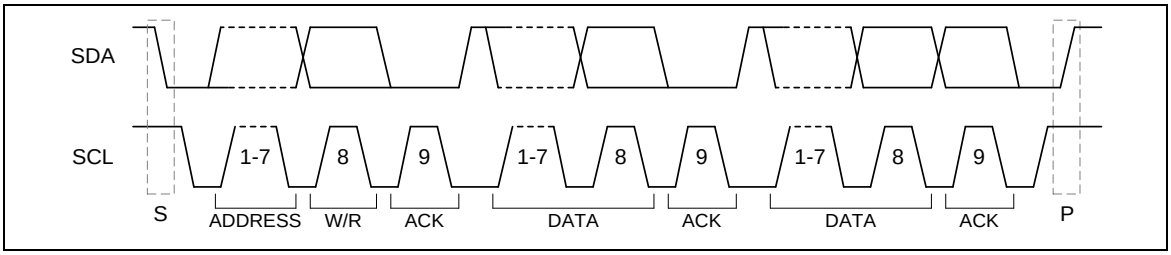


图 6.11-3 I²C 协议

6.11.5.1.1 起始信号或重复起始信号

当总线处于空闲状态下，说明没有主机设备占用总线（SCL和SDA都为高），主机可以通过发送起始信号来发起传输。起始信号，通常表示为S-bit，当SCL线为高时，SDA线上信号由高至低变化，就被定义为起始信号。起始信号表示一个新的数据传输的开始。

重复起始信号并不是两个起始信号之间的停止信号，通常表示为Sr-bit。主机用这个方式来与另一个从机进行通信或者与相同的从机进行相反的传输方向上的通信（从写设备到读设备），而不必释放总线。

6.11.5.1.2 停止信号

主机可以通过产生一个停止信号来终止数据传送。停止信号，通常表示为P-bit，当SCL线为高时，SDA线上信号由低至高变化，就被定义为停止信号。

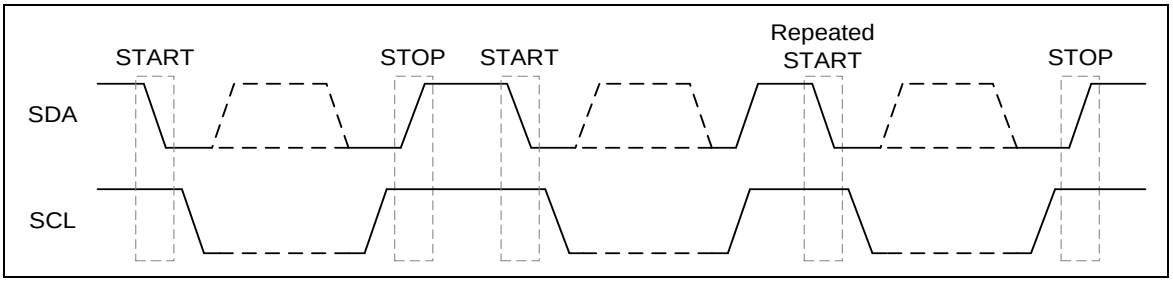


图 6.11-4 起始(START)和停止(STOP)条件

6.11.5.1.3 从机地址传输

在起始信号后，主机会立即传送的第一个字节的数据就是从机地址（SLA）。该字节是由称为从机地址的7位跟随1位读/写（R/W）位构成。R/W位，对于从机信号而言，表示着数据的传输方向。系统中不可能让两个从机有相同的地址，只有被主机寻址的从机设备才会通过在第9个SCL时钟周期内将SDA拉低作为应答信号。

6.11.5.1.4 数据传输

当从机接收到了带有读/写位的正确地址时，数据将会依照读/写位按规定传输。每一个传输的字节都会在SCL的第9个时钟周期跟随一个应答信号。如果从机产生一个非应答信号（NACK），主机可以产生一个停止信号来中止数据传输或者产生一个重复起始信号来开始新一轮的传输。

当主机作为一个接收设备时，给从机产生无应答信号（NACK），从机就会释放数据线来让主机产生停止或重复起始信号。

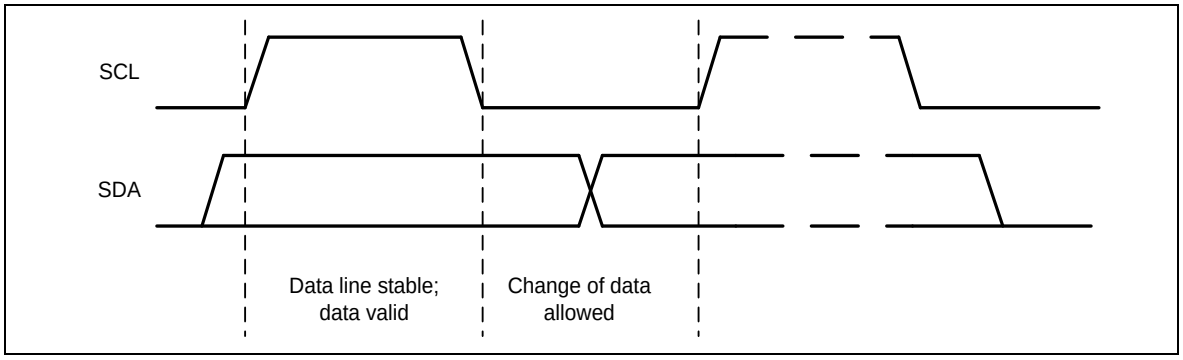


图 6.11-5 I^2C 总线上的位传输

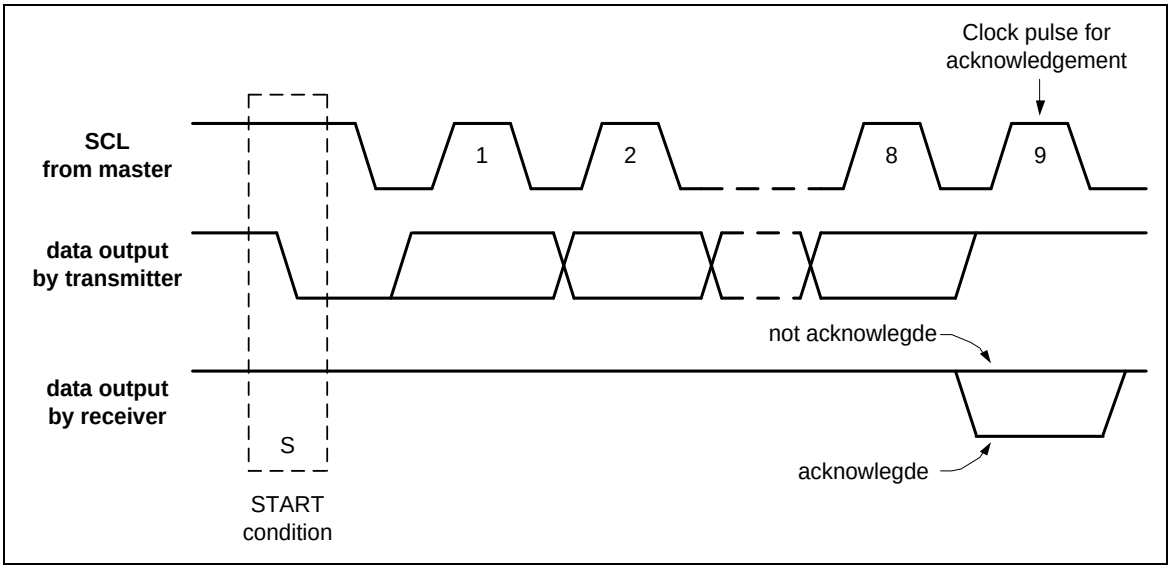


图 6.11-6 I²C 总线上的应答信号

6.11.5.1.5 I²C 总线上的数据传输

图 6.11-7表示了主机向从机传输数据。主机发送给从机7位地址和1位写命令，表示主机想要传送数据到从机。从机回复应答信号后，主机继续传送数据。

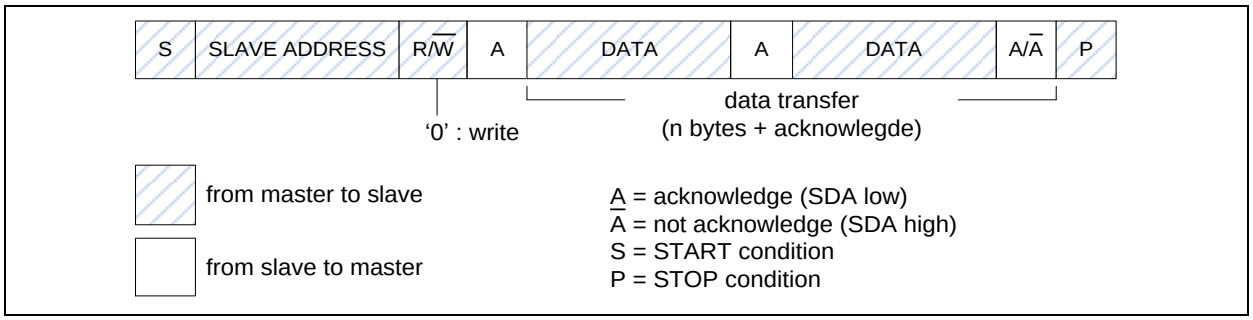


图 6.11-7 主机传送数据到从机

图 6.11-8 表示了主机向从机读取数据。主机发送给从机7位地址和1位读命令，表示主机想要向从机读取数据。从机回复应答信号后，就开始传送数据。

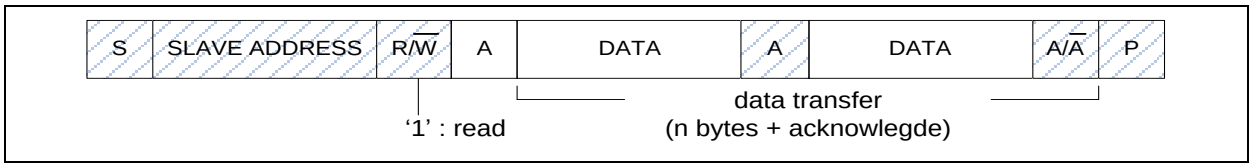


图 6.11-8 主机向从机读数据

6.11.5.1.6 I²C总线的二级缓存模式

使能二级缓存是用来临时存储I²C发送或接收的数据。这样，可以提高I²C总线的性能。如果TWOFF_EN设置为1，在当前SI清除之后，用于重复起始或停止的STA位应当被置位。

例如：4个数据被传输后再停止。STO位应该在第3个数据的SI事件清除后，进行置位。这时，第4个数据才可以被传输，在第4个数据传输完成后，I²C停止。

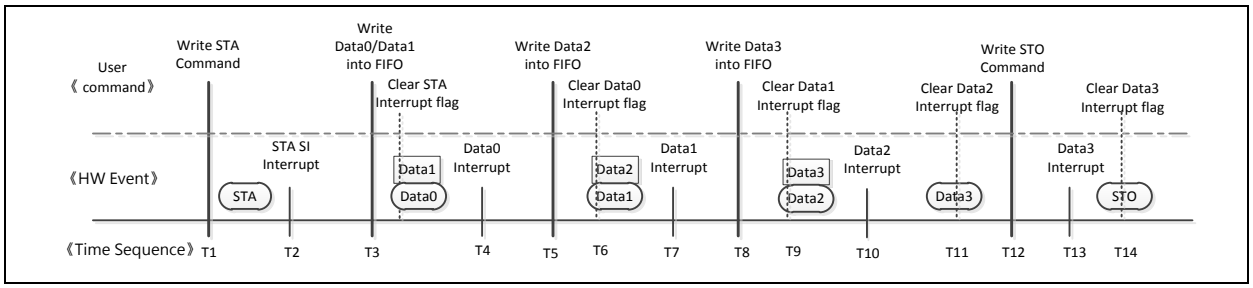


图 6.11-9 主机写时 二级缓存传输时序

例如：从机模式下，要收到4个数据。在第1个数据加载进接收缓存后，控制器才能从I²C总线上接收第2个数据，用户可以在第1个中断状态清除后，读到第1个数据。

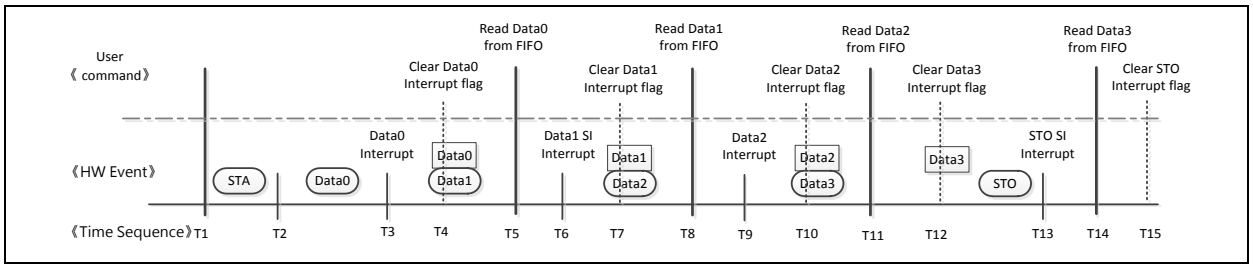


图 6.11-10 从机读时二级缓存传输时序

如果在总线停止（STO）后，缓存不是空的状态，用户可以设置TWOFF_EN为‘0’来让缓存控制状态器进入空闲状态，缓存将会被设置为默认值。

6.11.5.2 操作模式

片上I²C端口支持三种操作模式，主机、从机和广播呼叫模式。

在已给的应用中，I²C端口可以被操作为主机或者从机。在从机模式下，I²C端口硬件查找自身从机地址或广播呼叫地址，如果这两个地址的任一个被检测到，并且从机打算从主机接收或向主机发送数据(通过设置AA位)，应答脉冲将会在第9个时钟被发出，此时，如果中断被使能，则在主机和从机设备上都会发生一次中断请求。在主控芯片要成为总线主机时，在进入主机模式之前，硬件等待总线空闲使合理的从机动作不会被打断，在主机模式下，如果总线仲裁丢失，I²C立即切换到从机模式，并可以在同一次串行传输过程中检测自身从机地址。

为控制I²C总线在每种模式下的传输，用户需要依据I2C_STATUS的当前状态来设置I2C_CTL, I2C_DAT寄存器。换句话来说，I²C总线上的每一个动作，用户都需要查询I2C_STATUS寄存器的当前状态，然后设置I2C_CTL, I2C_DAT寄存器来让总线动作。最终，通过I2C_STATUS来检查响应状态。

在寄存器I2C_CTL [3]的SI标志清除后，I2C_CTL寄存器中STA、STO和AA用来控制I2C硬件的下一个状态，当完成一个新的动作，I2CSTATUS的状态代码将被更新，SI标志将被设置。如果I2C中断控制位EI (I2CON [7])被设置，新状态代码对应的动作或者软件将在中断服务程序中被执行。

图 6.11-11 显示了当前I^2C状态码是0x08， 然后设置I2CDATA=SLA+W和(STA,STO,SI,AA) = (0,0,1,x)发送I^2C总线地址。如果在总线上的从机地址相匹配则返回ACK，I2CSTATUS状态码更新为0x18。

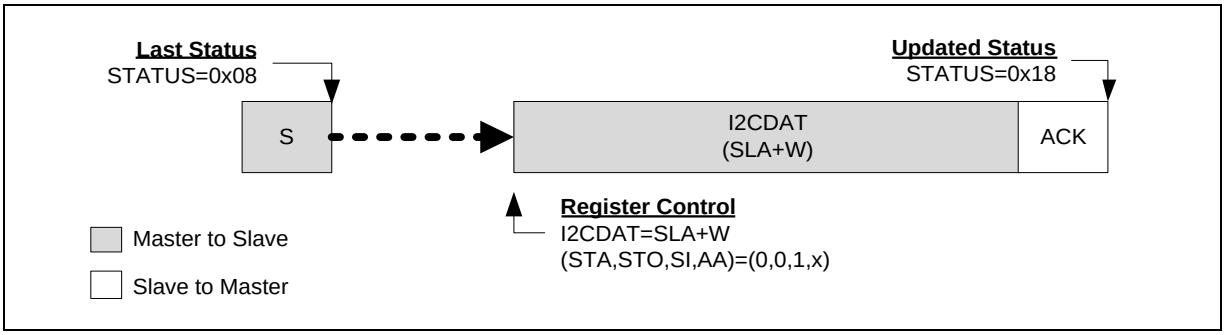


图 6.11-11 根据I^2C当前状态控制总线

6.11.5.2.1 主机模式

I^2C主机所有可能的协议如图 6.11-15 和 图 6.11-13，用户需要遵循恰当的流程来实现I^2C协议。

换句话说，用户可以发送一个起始信号到总线，I^2C将会被设置为主机传送模式（见图 6.11-15）或主机接收模式（见图 6.11-13），起始信号成功传输后，新的状态码将变为0x80。起始信号发送后，用户可以发送从机地址，读写位，数据，重复起始信号和停止信号来执行I^2C协议。

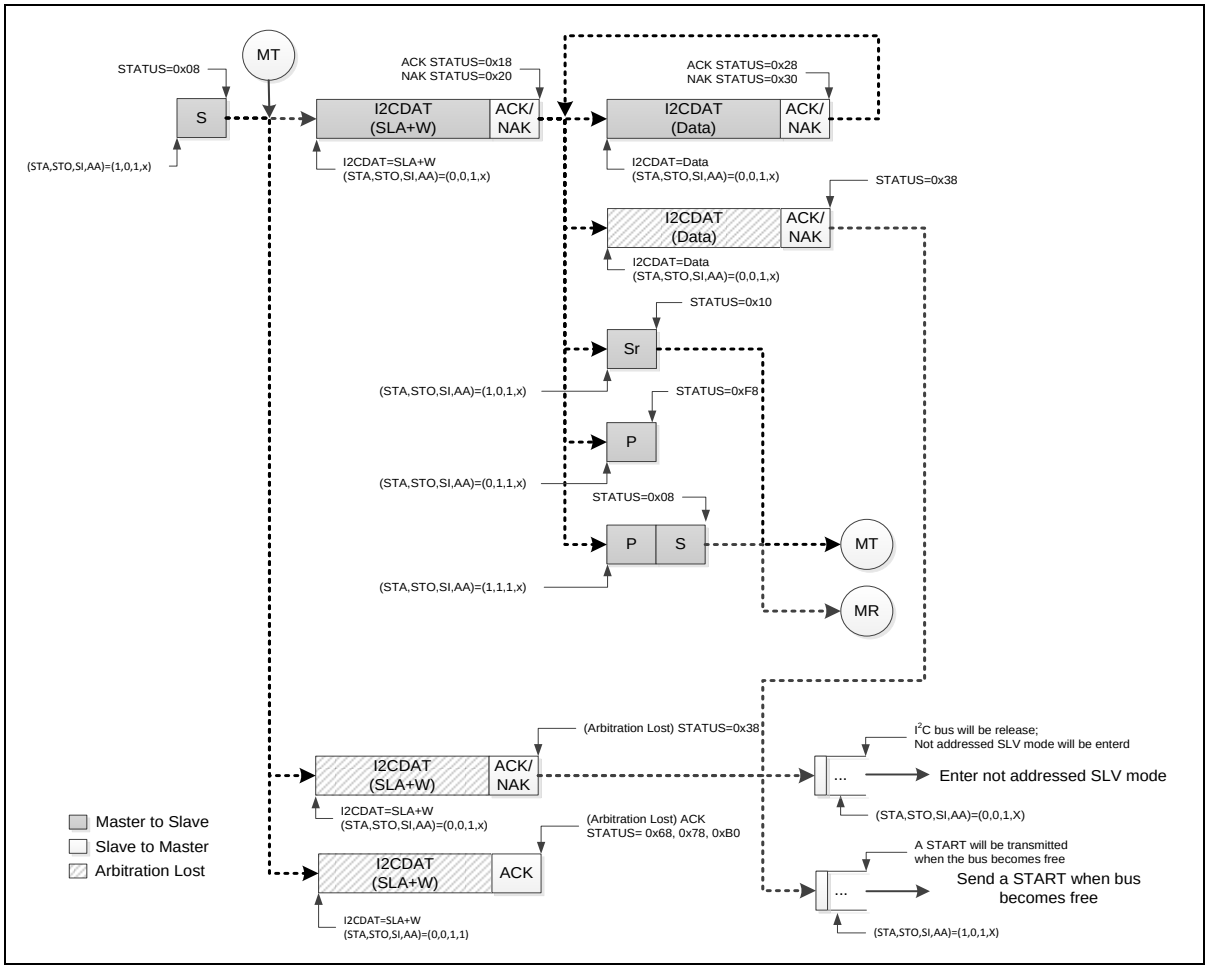


图 6.11-12 主机传送模式控制流程

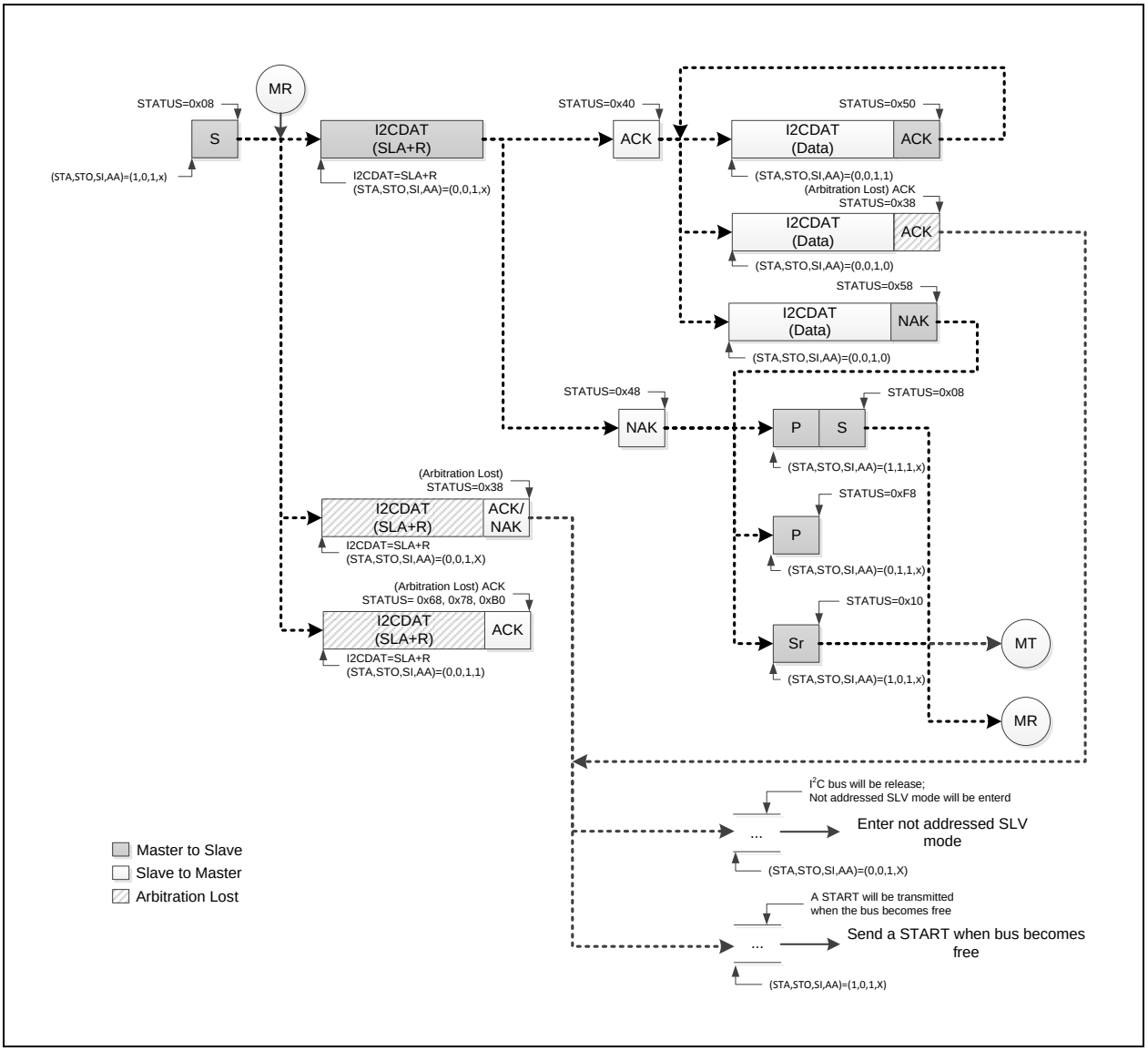


图 6.11-13 主机接收模式控制流程

如果I2C在主机模式并且仲裁丢失，状态码将置为0x38，在状态0x38时，用户可以设置(STA, STO, SI, AA) = (1, 0, 1, X)在总线空闲时发送起始信号来重新开始主机操作。另外，用户可以设置(STA, STO, SI, AA) = (0, 0, 1, X)来释放总线，并进入无地址从机模式。

6.11.5.2.2 从机模式

复位后默认情况下，I2C不会被寻址，并且不会识别I2C总线上的地址。用户可以通过设置从机地址I2CADDRx 和(STA, STO, SI, AA) = (0, 0, 1, 1)来让I2C识别主机发送的地址。图 6.11-15示意了从机模式的所有流程。用户需要遵循合适的流程（如图 6.11-15所示）来实现自己的I2C协议。

如果在主机模式仲裁丢失，I2C端口立即切换到从机模式并且在同一串口传输中识别自有的从机地址。如果在仲裁丢失后识别到地址是SLA+W（主机写数据到从机），状态码是0X68。如果在仲裁丢失后识别到地址是SLA+R（主机向从机读数据），状态码是0xB0。

注: I²C通讯时，在从机模式下当写“1”清除SI标志时，SCL时钟会被释放。

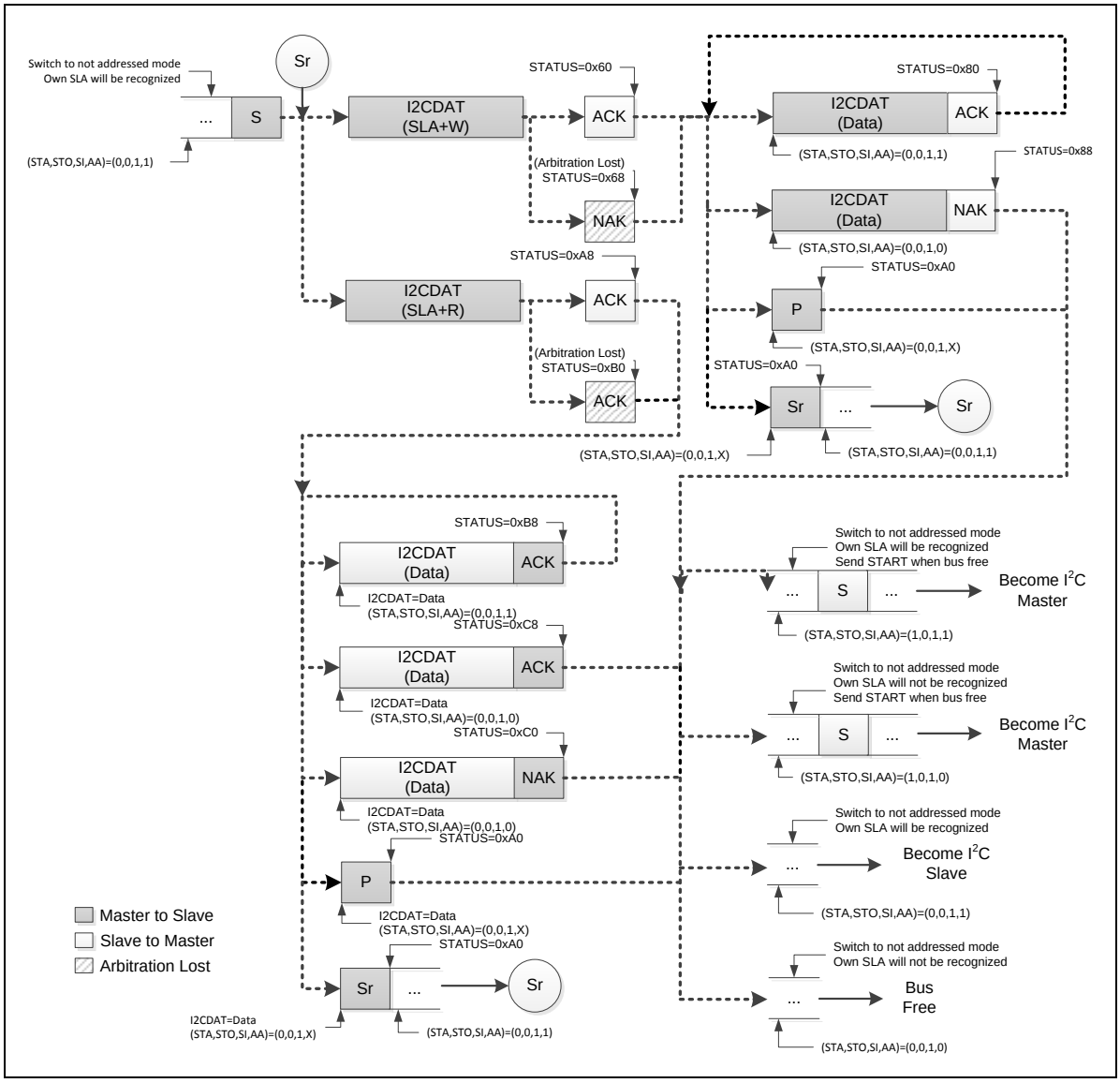


图 6.11-14 从机模式控制流程

如果I²C处在被寻址的从机模式下接收数据，却收到停止或重复起始信号，状态码是0xA0。此时，用户可以遵循如图 6.11-15状态码是0x88的操作。

如果I²C处在被寻址的从机模式下发送数据，却收到停止或重复起始信号，状态码是0xA0。此时，用户可以遵循如图 6.11-15状态码是0xC8的操作。

注: 从机获得0x88, 0xC8, 0xC0 和0xA0状态后,从机会切换到无地址模式，自身SLA不会被辨识。如果进入这种状态，从机不再从主机接收任何信号或地址。需要复位才能离开这种状态。

6.11.5.2.3 广播呼叫模式（GC）

如果GC位（I2C_ADDRx [0]）被置位,I²C 端口硬件将响应广播呼叫地址 (00H)。用户可以清除GC位来禁止广播呼叫功能。当GC bit被设置并且I²C在从模式时, 在主机发送广播呼叫地址到I²C总线上时， 它可以接收广播呼叫地址0x00,然后遵循广播呼叫模式的状态。

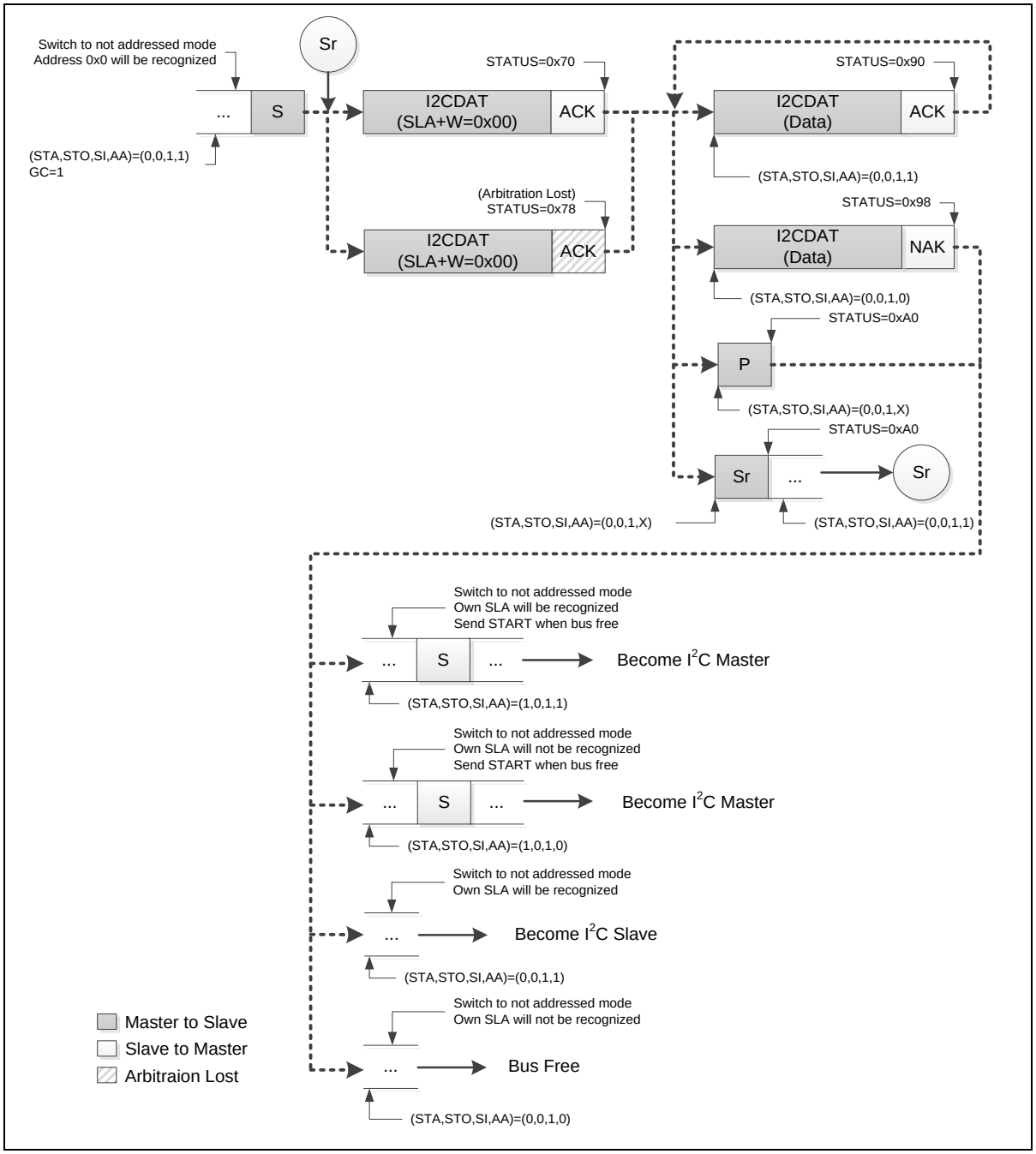


图 6.11-15 GC 模式

当I2C还在接收数据时，但是收到了STOP 或者重复 START信号，状态码将变成0xA0. 用户应该遵守上图6.11-15的状态码0x98来进行处理。

注: 从机得到0x98和0xA0状态码之后，从机会切到未寻址模式，并且自身的SLA也不会被识别。如果进入这个状态，从机不会接收来自主机的任何I2C信号和地址。可以通过重置I2C控制器以离开此状态。

6.11.5.2.4 多主机

在某些应用中， I^2C总线上同时有2个或者多个主机，主机可能同时发送数据。该I^2C具备冲突检测和裁决的功能，可以避免数据冲突，支持多主机模式。

- 当 I2C_STATUS = 0x38时，会发生“仲裁失败”。仲裁失败事件可能在发送START信号、数据信号或者STOP信号时，用户在总线空闲的时候，设置 (STA, STO, SI, AA) = (1, 0, 1, X) 来再次发送START 信号，或者设置(STA, STO, SI, AA) = (0, 0, 1, X) 来发送STOP信号返回未寻址从机模式。
- 当I2C_STATUS = 0x00时，会发生“总线错误”，为了将I2C从总线错误中恢复，可以设置STO并清除SI，之后STO会被清除总线也得以释放.
  - 设置 (STA, STO, SI, AA) = (0, 1, 1, X) 停止当前传输
  - 设置 (STA, STO, SI, AA) = (0, 0, 1, X) 释放总线

6.11.5.3 I^2C协议寄存器

可通过如下15个特殊功能寄存器来控制 I^2C端口： I2C_CTL （控制寄存器），I2C_STATUS （状态寄存器），I2C_DAT （数据寄存器）， I2C_ADDRn（地址寄存器，n=0~3），I2C_ADDRMSKn（地址屏蔽寄存器，n=0~3）,I2C_CLKDIV （时钟速率寄存器），I2C_TOCTL （超时计数寄存器），I2C_CTL1 （控制寄存器1）和 I2C_STATUS（状态寄存器1）。

6.11.5.3.1 地址寄存器(I^2C_ADDR)

该I^2C端口带有4个从机地址寄存器I2C_ADDRn (n=0~3)。 当I^2C处于主机模式时，这四个寄存器的值是无关的。 在从机模式下,位字段I2C_ADDRn [7:1]必须载入MCU自身的从机地址，当I2C_ADDRn地址与接收的从机地址符合时, I^2C硬件将会应答。

I^2C 端口支持广播呼叫功能。当GC位(I2C_ADDRn [0])被置位，I^2C端口硬件会响应广播呼叫地址(00H). 清除GC位可禁止广播呼叫功能。

当GC位被置位，且I^2C处于从机模式时，在主机发送广播呼叫地址到I^2C总线上之后，I^2C可以接收广播呼叫地址00H，然后它将跟随GC模式的状态。

6.11.5.3.2 从机地址屏蔽寄存器 (I2C_ADDRMSK)

I^2C总线控制器支持多地址识别，带有4组地址屏蔽寄存器I2C_ADDRMSKx （n=0~3）。当地址屏蔽寄存器的某一位被设置为1时，表示接收到的地址的相应位将被忽略，如果该位设置为0，表示接收到的地址的相应位应当与地址寄存器中相应位的值完全一致。

6.11.5.3.3 数据寄存器(I2C_DAT)

该寄存器存储的内容是准备发送或刚接收的一个字节的数据。不进行移位处理时，CPU可以直接读写这8位I2C_DAT [7:0]。当I^2C处于定义的状态下，且串行中断标志(SI)被置位时，I2C_DAT[7:0]中的数据将保持稳

定。在数据被移出的过程中，总线上的数据会同时被移入，I2CDAT[7:0]总是存放着呈现在总线上的最后一个字节的数据。

应答位由I²C的硬件控制，CPU不能访问。串行数据在SCL线上串行时钟脉冲的上升沿被移入I2C_DAT[7:0]。当一个字节被移入到I2C_DAT [7:0]后，I2C_DAT [7:0]中的串行数据就可以使用，应答位(ACK或NACK)，会在第 9 个时钟脉冲时通过控制逻辑返回。为了在发送数据的时候监控总线状态，当发送I2C_DATA[7:0]中数据到总线的时候，总线上的数据将同时被移入I2C_DATA[7:0]中。串行数据在SCL时钟脉冲的下降沿从I2C_DAT[7:0]中被移出，在SCL时钟脉冲的上升沿被移入I2C_DAT[7:0]。

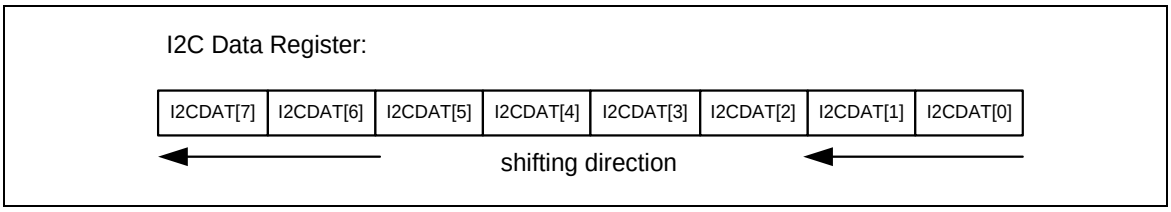


图 6.11-16 I2C 数据移动方向

6.11.5.3.4 控制寄存器(I²C_CTL)

CPU可以直接读写寄存器I2C_CTL [7:0]。当 I2C端口通过设置I2CEN (I2C_CTL [6])为1使能后，内部状态将会由I2C_CTL和I²C逻辑硬件控制。

当I²C硬件请求串行中断时SI位被置位，或总线上出现停止信号时STO位被清除时，这两位均会受到硬件的影响。当I2CEN = 0时，STO位也会被清除。

当有新的状态码产生并存储在I2C_STATUS寄存器中时，I²C中断标志位SI(I2C_CTL[3])将自动设置，如果使能中断位INTEN（ I2C_CTL［7］）也在此时被置位，则会产生I²C中断请求。内部状态码在I2C_STATUS[7:0]中保存，在SI被软件清除前，其内容保持不变。

6.11.5.3.5 状态寄存器(I2C_STATUS)

I2C_STATUS [7:0] 是一个8位只读寄器，I2C_STATUS[7:0]包含状态码。有26个可能的状态码。所有的状态都在图 6.11-1中列出。当I2C_STATUS [7:0]的内容为F8H时，则没有串口中断请求产生。所有其它的I2C_STATUS [7:0]的值都对应已定义的I2C状态。 所有的状态在下表列出。 如果I2CSTATUS [7:0]等于F8H，表示没有串行中断产生。当进入这些状态中的任一个，就会产生状态中断请求(SI=1)。在SI被硬件置位或被软件复位后的1个机器周期，有效状态码出现在I2C_STATUS [7:0]中。

此外，00H状态表示总线错误。该错误会在起始或停止信号处于帧结构不正确的位置时发生。 总线错误也有可能在串行传输地址字节，数据字节或应答位期间发生。为了将I²C从总线错误中恢复，需要置位STO，清除SI从而进入未寻址从机模式，然后STO会被清除从而释放总线并等待新的通信。I²C总线在总线错误时不能识别停止信号。

主机模式		从机模式	
状态	描述	状态	描述
0x08	开始	0xA0	从机发送重复开始或停止
0x10	主机重复开始	0xA8	从机发送地址 ACK
0x18	主机发送地址 ACK	0xB0	从机发送仲裁丢失
0x20	主机发送地址 NACK	0xB8	从机发送数据 ACK
0x28	主机发送地址ACK	0xC0	从机发送数据 NACK

0x30	主机发送数据NACK	0xC8	从机发送最后一个数据 ACK
0x38	主机仲裁丢失	0x60	从机接收地址 ACK
0x40	主机接收地址 ACK	0x68	从机接收仲裁丢失
0x48	主机接收地址 NACK	0x80	从机接收数据 ACK
0x50	主机接收数据 ACK	0x88	从机接收数据 NACK
0x58	主机接收数据 NACK	0x70	广播模式地址 ACK
0x00	总线错误	0x78	广播模式仲裁丢失
		0x90	广播模式数据 ACK
		0x98	广播模式数据 NACK
0xF8	总线释放 **注:** 状态“0xF8”存在于主机和从机模式，它不会引起任何中断		

表 6.11-1 I^{2}C 状态码描述表

6.11.5.3.6 I^{2}C时钟波特率位(I2C_CLKDIV)

当I^{2}C在主机模式下, I^{2}C数据的波特率由I2C_CLKDIV[7:0]寄存器决定。在从机模式下其数值并不重要。从机模式下，I^{2}C将自动与主机I^{2}C设备时钟频率同步。

数据波特率设定公式：I^{2}C的数据波特率 = (system clock)/(4x(I2C_CLKDIV[7:0]+1))，如果system clock=16MHz， I2C_CLKDIV[7:0]= 40(28H)， I2C的数据波特率 = 16MHz /(4* (40 +1)) = 97.5K比特/秒。

6.11.5.3.7 I^{2}C超时计数寄存器(I2C_TOCTL)

有一个14位的超时计数器可以用于处理I2C总线挂起。当计数功能使能后，计数器开始计数直至发生超时(TOIF=1),并对CPU发出 I^{2}C中断请求，或将I2CEN清除为0。当超时计数器使能后， 设定SI标志为高会复位计数器，清零SI之后计数器会重新开始计数。如果I^{2}C总线挂起，会导致I2C_STATUS及SI标志在一段时间内不能更新。该14位超时计数器可能溢出并发出I^{2}C中断请求。关于14位超时计数器可参考图 6.11-17，用户可通过对TOIF位写1清0该标志。

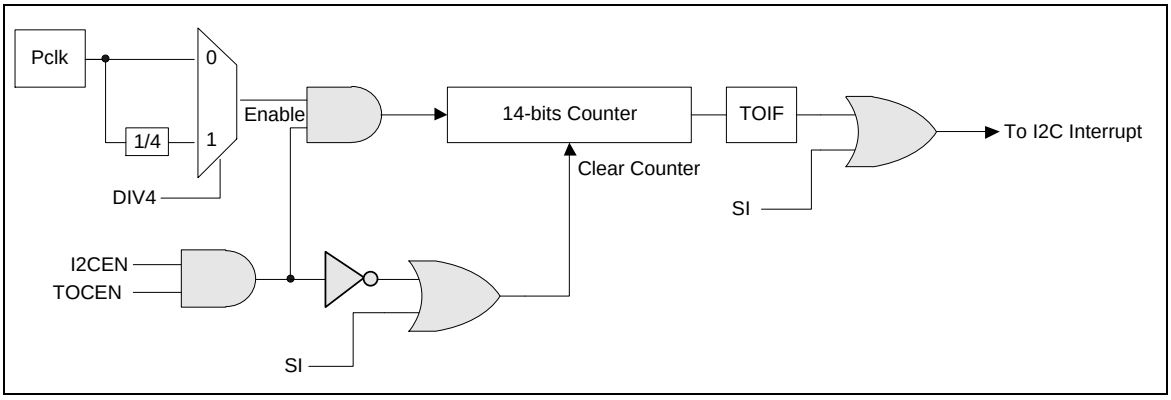


图 6.11-17 I^{2}C 超时计数器框图

6.11.5.3.8 I²C 控制寄存器1(I2C_CTL1)

当NSTRETCH(I2C_CTL1[2])位设置为1时，表示不对总线时钟进行控制。

关于TWOLVFIFO (I2C_CTL1[1])位，它用来使能I²C发送或接收的二级缓存，用来提升I²C总线的性能。如果该位被设为1，用来重复起始的START (I2C_CTL[5])控制位或STOP (I2C_CTL[4])控制位应该在当前中断被清除后置位。举个例子，假如要传送4个数据然后停止，STOP位应该在第3个数据中断被清除后置位，此时，第4个数据可以被传送，I²C将会在第4个数据传送完成后停止。

可以通过I2C_CTL1[4:3]来使能二级缓存状态中断，以便产生under run 和 over run 事件。

当进入低功耗模式，其它I²C主机可以通过I²C设备地址唤醒芯片，用户必须在进入低功耗模式前配置有关的设置。当芯片通过四个地址当中的一个匹配地址唤醒时，随后的数据此时将会被丢弃。注意，只有I²C0才支持唤醒功能。

6.11.5.3.9 I²C 状态寄存器 1 (I2C_STATUS1)

在该寄存器中可以包含的状态有：二级缓存状态、忙碌空闲状态、缓存空状态、over run等状态。

当通过其它I²C主机设备唤醒时，WKIF会被置位来表示该事件。用户需要去写1来清除该位。当TWOLVFIFO (I2C_CTL1[1])被置位时，其它的状态位用来表示当前缓存的状态。注意，只有I²C0才支持唤醒功能。

6.11.5.4 随机读取 EEPROM的例子

使用I2C读EEPROM时，使用下列步骤配置I2C相关寄存器：

1. 在“SYS_P3_MFP”寄存器中设置多功能引脚当作SCL和 SDA 引脚
2. 使能I2C APB 时钟：“CLK_APBCLK” 寄存器中I2CCKEN=1
3. 设置I2C_RST=1复位I²C控制器，然后设置“SYS_IPRST1”的I2C_RST=0使I²C控制器变成正常操作模式
4. “I2C_CTL” 寄存器中设置I2CEN =1 使能 I²C 控制器
5. 给I²C时钟速率寄存器“I2C_CLKDIV设置合适的比特率。
6. “NVIC_ISER”寄存器中设置 SETENA=0x00040000使能 I²C 中断
7. “I2C_CTL”寄存器中设置 EI=1 使能 I2C 中断.
8. 通过“I2C_ADDR0~I2C_ADDR3” 设置I²C地址寄存器

随机读操作是EEPROM中的一个访问方法。该方法允许主机访问EEPROM中的任何地址。图 6.11-18显示了EEPROM 随机读操作的流程.

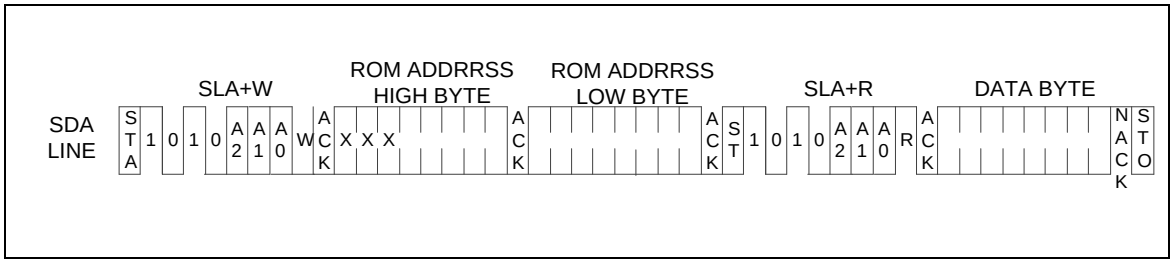


图 6.11-18 EEPROM 随机读操作

图 6.11-19显示了怎样使用I²C控制器来实现EEPROM随机读协议。

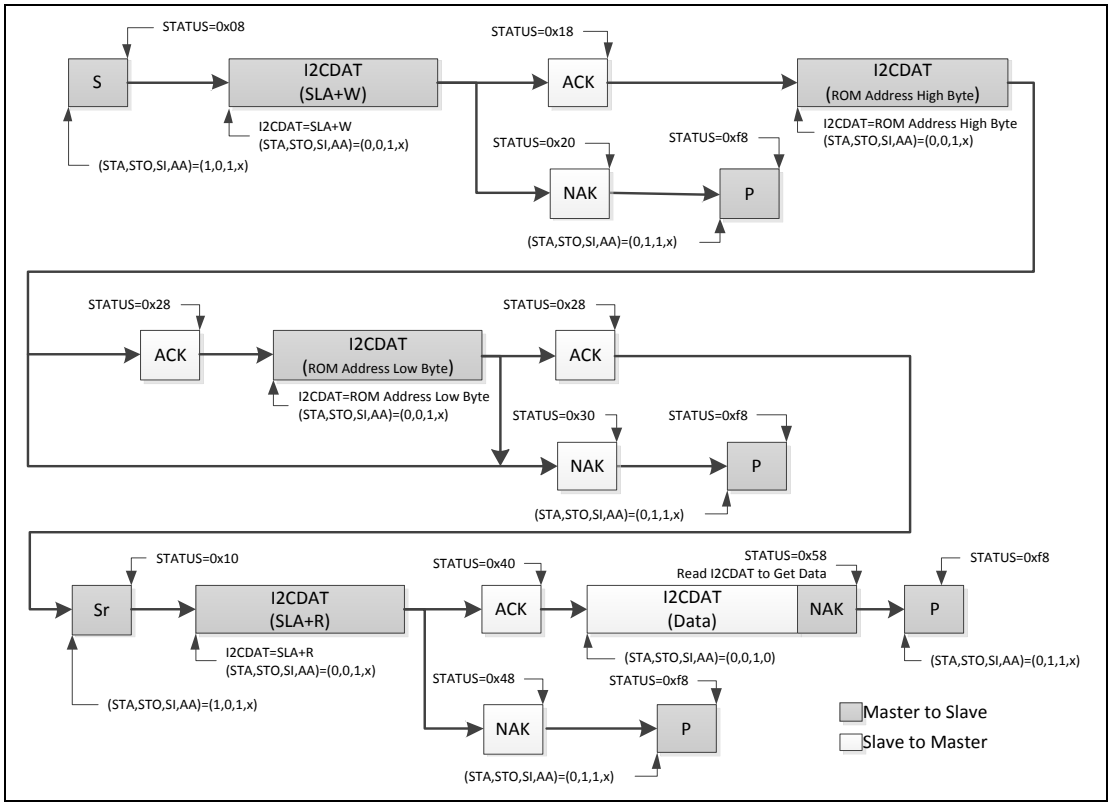


图 6.11-19 随机读协议

作为主机，I²C 控制器发送 START信号到总线上，然后发送SLA+W (从机地址+写)到EERPOM ，之后是2个字节的要读的EEPROM数据的地址。最后，重复START信号并发送SLA+R，之后就能从EEPROM传出要读取的数据了。

6.11.6 寄存器映射

R: 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移量	读/写	描述	复位值
I2C 基地址: **I2C0_BA = 0x4002_0000** **I2C1_BA = 0x4012_0000**				
I2C_CTL **x= 0, 1**	I2Cx_BA+0x00	R/W	I^2C 控制寄存器	0x0000_0000
I2C_DAT **x= 0, 1**	I2Cx_BA+0x08	R/W	I^2C 数据寄存器	0x0000_0000
I2C_STATUS **x= 0, 1**	I2Cx_BA+0x0C	R	I^2C 状态寄存器	0x0000_00F8
I2C_CLKDIV **x= 0, 1**	I2Cx_BA+0x10	R/W	I^2C 时钟分频寄存器	0x0000_0000
I2C_TOCTL **x= 0, 1**	I2Cx_BA+0x14	R/W	I^2C 超时控制寄存器	0x0000_0000
I2C_ADDR0 **x= 0, 1**	I2Cx_BA+0x04	R/W	I^2C 从机地址寄存器 0	0x0000_0000
I2C_ADDR1 **x= 0, 1**	I2Cx_BA+0x18	R/W	I^2C 从机地址寄存器 1	0x0000_0000
I2C_ADDR2 **x= 0, 1**	I2Cx_BA+0x1C	R/W	I^2C 从机地址寄存器 2	0x0000_0000
I2C_ADDR3 **x= 0, 1**	I2Cx_BA+0x20	R/W	I^2C 从机地址寄存器 3	0x0000_0000
I2C_ADDRMSK0 **x= 0, 1**	I2Cx_BA+0x24	R/W	I^2C 从机地址掩码寄存器 0	0x0000_0000
I2C_ADDRMSK1 **x= 0, 1**	I2Cx_BA+0x28	R/W	I^2C 从机地址掩码寄存器 1	0x0000_0000
I2C_ADDRMSK2 **x= 0, 1**	I2Cx_BA+0x2C	R/W	I^2C 从机地址掩码寄存器 2	0x0000_0000
I2C_ADDRMSK3 **x= 0, 1**	I2Cx_BA+0x30	R/W	I^2C 从机地址掩码寄存器 3	0x0000_0000
I2C_CTL1 **x= 0, 1**	I2Cx_BA+0x3C	R/W	I^2C 控制寄存器 1	0x0000_0000
I2C_STATUS1 **x= 0, 1**	I2Cx_BA+0x40	R/W	I^2C 状态寄存器 1	0x0000_0000

6.11.7 寄存器描述

I²C 控制寄存器 (I2C_CTL)

寄存器	偏移量	R/W	描述	复位值
I2C_CTL **x= 0, 1**	I2Cx_BA+0x00	R/W	I²C 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN	**I2CEN**	**STA**	**STO**	**SI**	**AA**	保留	

位	描述	
[31:8]	保留	保留
[7]	**INTEN**	**中断使能位** 0 = I²C 中断禁用 1 = I²C 中断使能
[6]	**I2CEN**	**I²C 控制器使能位** 0 = I²C 控制器禁用 1 = I²C 控制器使能 使能 I²C 串行功能控制器。 当 I2CEN=1 时，I²C 串行功能使能。多功能管脚的功能必须要预先设置为 I²C。
[5]	**STA**	**I²C START 控制位** 设置 STA 为 1 进入主机模式。当总线处于空闲状态时， I²C 硬件发送START 或重复START。
[4]	**STO**	**I²C STOP 控制位** 在主机模式，设置 STO 来传送一个 STOP 条件到总线，然后 I²C 硬件将会检查总线状况，如果检测到一个 STOP 状况，这个标志会被硬件自动清除。在 I²C 从机模式，设置 STO 复位 I²C 硬件来定义“无地址”从机模式，这表示在从机接收模式下不再接收从主机设备发送的数据。
[3]	**SI**	**I²C 中断标志** 当一个新的 I²C 状态出现在寄存器I2C_STATUS时，SI 标志由硬件置位，并且如果INTEN (I2C_CTL[7])位被置位，则产生 I²C 中断请求。SI 必须由软件通过向该位写 ’1’ 清零。

位	描述	
[2]	**AA**	**接收应答控制位** 当 AA=1 先于地址或数据被接收，在SCL上的应答时钟脉冲在以下两种情况时，应答（SDA低电平）将会被返回。1.) 从机正在应答主机发送的地址。2.) 接收设备正在应答发送设备发送的数据。当 AA = 0 先于地址或数据被接收时，则在SCL线上的应答时钟脉冲期间将返回一个非应答（SDA高电平）信号。
[1:0]	**保留**	保留

I²C 数据寄存器 (I2C_DAT)

寄存器	偏移量	R/W	描述	复位值
I2C_DAT **x= 0, 1**	I2Cx_BA+0x08	R/W	I²C 数据寄存器	0x0000_0000

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

位	描述	
[31:8]	保留	保留
[7:0]	**DAT[7:0]**	**I²C 数据寄存器** I²C串行端口8-位传输数据从头位于Bit [7:0]

I²C 状态寄存器 (I2C_STATUS)

寄存器	偏移量	R/W	描述	复位值
I2C_STATUS x= 0, 1	I2Cx_BA+0x0C	R	I²C 状态寄存器	0x0000_00F8

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

位	描述	
[31:8]	保留	保留.
[7:0]	**STATUS[7:0]**	**I²C 状态寄存器** 三个最低有效位总是0。五个最重要的位包含着状态码。共有26个可能状态码。当I2C_STATUS值是0xF8没有串行中断请求。所有其他的 I2C_STATUS的值对应 I2C 的状态。当进入每一状态时，就会产生状态中断请求 (SI= 1)。在 SI 被硬件置位或 SI 被软件复位后一个机器周期后，有效状态码出现在 I2C_STATUS中。此外，0x00状态表示总线错误。该错误会在起始或停止信号处于帧结构不正确的位置时发生。不正确的位置比如是在串行传输地址字节、数据字节或应答位期间。

I²C 时钟分频寄存器 (I2C_CLKDIV)

寄存器	偏移量	R/W	描述	复位值
I2C_CLKDIV x= 0, 1	I2Cx_BA+0x10	R/W	I²C 时钟分频寄存器	0x0000_0000

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

位	描述	
[31:8]	保留	保留.
[7:0]	**DIVIDER[7:0]**	**I²C 时钟分频寄存器** I²C 数据波特率 = (系统时钟) / (4 * (I2C_CLKDIV+1)) **注意:** I2C_CLKDIV 的最小值为 4

I²C 超时控制寄存器 (I2C_TOCTL)

寄存器	偏移量	R/W	描述	复位值
I2C_TOCTL **x= 0, 1**	I2Cx_BA+0x14	R/W	I²C 超时控制寄存器	0x0000_0000

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

位	描述	
[31:3]	保留	保留.
[2]	**TOCEN**	**超时计数使能位** 0 = 禁用超时计数 1 = 使能超时计数 **注:** 当14位超时计数器使能，它将会在SI清除后开始计数。设置SI为1将会复位计数器，在SI清除后会重新开始计数。
[1]	**TOCURIEN**	**超时定时器输入时钟除以 4** 0 = 禁用 1 = 使能 **注:** 该位使能后，超时时间扩大 4 倍。
[0]	**TOIF**	**超时标志** 当超时发生时，该位由硬件置位，如果此时 I²C 的中断使能位 （INTEN） 置为1，则可引发 CPU 的中断。 **注:** 写 1 清除该位。

I²C 从机地址 寄存器(I2C_ADDRx)

寄存器	偏移量	R/W	描述	复位值
I2C_ADDR0 x= 0, 1	I2Cx_BA+0x04	R/W	I²C 从机地址寄存器 0	0x0000_0000
I2C_ADDR1 x= 0, 1	I2Cx_BA+0x18	R/W	I²C从机地址寄存器 1	0x0000_0000
I2C_ADDR2 x= 0, 1	I2Cx_BA+0x1C	R/W	I²C从机地址寄存器 2	0x0000_0000
I2C_ADDR3 x= 0, 1	I2Cx_BA+0x20	R/W	I²C从机地址寄存器 3	0x0000_0000

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

位	描述	
[31:8]	保留	保留.
[7:1]	**ADDR[7:1]**	**I²C 地址** 主机模式下，该寄存器内容没有意义。从机模式下，高7位作为芯片本身的地址。如果任一地址符合，I²C 硬件将会自动应答。
[0]	**GC**	**广播呼叫模式控制** 0 = 禁用 1 = 使能

I²C 从机地址掩码寄存器 (I2C_ADDRMSKx)

寄存器	偏移量	R/W	描述	复位值
I2C_ADDRMSK0 **x= 0, 1**	I2Cx_BA+0x24	R/W	I²C 从机地址掩码寄存器 0	0x0000_0000
I2C_ADDRMSK1 **x= 0, 1**	I2Cx_BA+0x28	R/W	I²C从机地址掩码寄存器 1	0x0000_0000
I2C_ADDRMSK2 **x= 0, 1**	I2Cx_BA+0x2C	R/W	I²C从机地址掩码寄存器 2	0x0000_0000
I2C_ADDRMSK3 **x= 0, 1**	I2Cx_BA+0x30	R/W	I²C从机地址掩码寄存器 3	0x0000_0000

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

位	描述	
[31:8]	保留	保留.
[7:1]	**ADDRMSK[7:1]**	**I²C 地址掩码位** 0 = I²C 地址掩码禁用 (接收到的相应地址必须完全符合地址寄存器) 1 = I²C地址掩码使能(接收到的相应地址位不比对)
[0]	保留	保留.

I²C 控制寄存器 1 (I2C_CTL1)

寄存器	偏移量	R/W	描述	复位值
I2C_CTL1 **x= 0, 1**	I2Cx_BA+0x3C	R/W	I²C 控制寄存器 1	0x0000_0000

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

位	描述	
[31:5]	保留	保留.
[4]	**URIEN**	**I²C Under Run 中断控制** 当TWOLVFF使能且在发送缓存中发生under run事件时，设置URIEN为1发送中断到系统。 0 = Under run 中断禁用. 1 = Under run 中断使能.
[3]	**OVIEN**	**I²C Overrun 中断控制** 当TWOLVFF使能且在接收缓存中发生over run事件时，设置OVIEN为1发送中断到系统。 0 = Overrun 中断禁用. 1 = Overrun中断使能.
[2]	**NSTRETCH**	**I²C 总线上不受控** 0=如果主机模式下SI没有被清除，I²C SCL总线由硬件控制。 1=如果主机模式下SI没有被清除，I²C SCL总线不由硬件控制。
[1]	**TWOLVFIFO**	二级缓存使能位 0 = 禁用 1 = 使能 设置以使能I2C发送或接收缓存。用来提升I2C总线的性能。如果该位被设为1，用来重复起始的START (I2C_CTL[5])控制位或STOP (I2C_CTL[4])控制位应该在当前中断被清除后置位。举个例子，假如要传送4个数据然后停止，STOP位应该在第3个数据中断被清除后置位，此时，第4个数据可以被传送，I2C将会在第4个数据传送完成后停止。

位	描述	
[0]	**WKEN**	**唤醒使能位** 0 = 禁用 1 = 使能 当系统进入低功耗模式下，且接收到的数据与地址寄存器中相匹配，就可以通过I^2C总线将系统唤醒。 **注:** 只有I^2C0才支持该功能，该位对I^2C1无效。

I2C 状态寄存器 1 (I2C_STATUS 1)

寄存器	偏移量	R/W	描述	复位值
I2C_STATUS 1 **x= 0, 1**	I2Cx_BA+0x40	R/W	I^2C 状态寄存器 1	0x0000_0000

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

位	描述	
[31:5]	保留	保留.
[4]	**URIF**	**I^2C Under Run 状态** 当TWOLVFIFO = 1时，该位表示发送缓存处于under run状态。
[3]	**OVIF**	**I^2C Overrun 状态** 当TWOLVFIFO = 1时，该位表示接收缓存处于overrun 状态。
[2]	**EMPTY**	**I^2C 二级缓存空状态** 当TWOLVFIFO = 1时，该位表示接收缓存是否空。
[1]	**FULL**	**I^2C 二级缓存满状态** 当TWOLVFIFO = 1时，该位表示发送缓存是否满。
[0]	**WKIF**	**I^2C 唤醒中断标志** 当芯片从低功耗模式下被I^2C唤醒时，该位设置为1。软件写1清零。 **注:** 只有I^2C0才支持唤醒功能，该位对I^2C1无效。

6.12 串行外围总线 (SPI)

6.12.1 概述

串行外围设备接口(SPI) 是一个工作于全双工模式的同步串行数据通讯协议。 设备可工作在主/从模式，利用4线双向接口相互通讯。当从一个外围设备接收数据时，SPI执行串-并的转换，而在数据向外围设备发送时执行并-串的转换。该SPI控制器可以配置为主设备或从设备。

6.12.2 特征

- 支持主机或从机模式
- 可配置发送的位长度
- 提供四个32位FIFO缓存
- 传输时支持高位或低位优先
- 支持字节排序功能
- 支持字节或字休眠功能
- 支持从机三线模式

6.12.3 框图

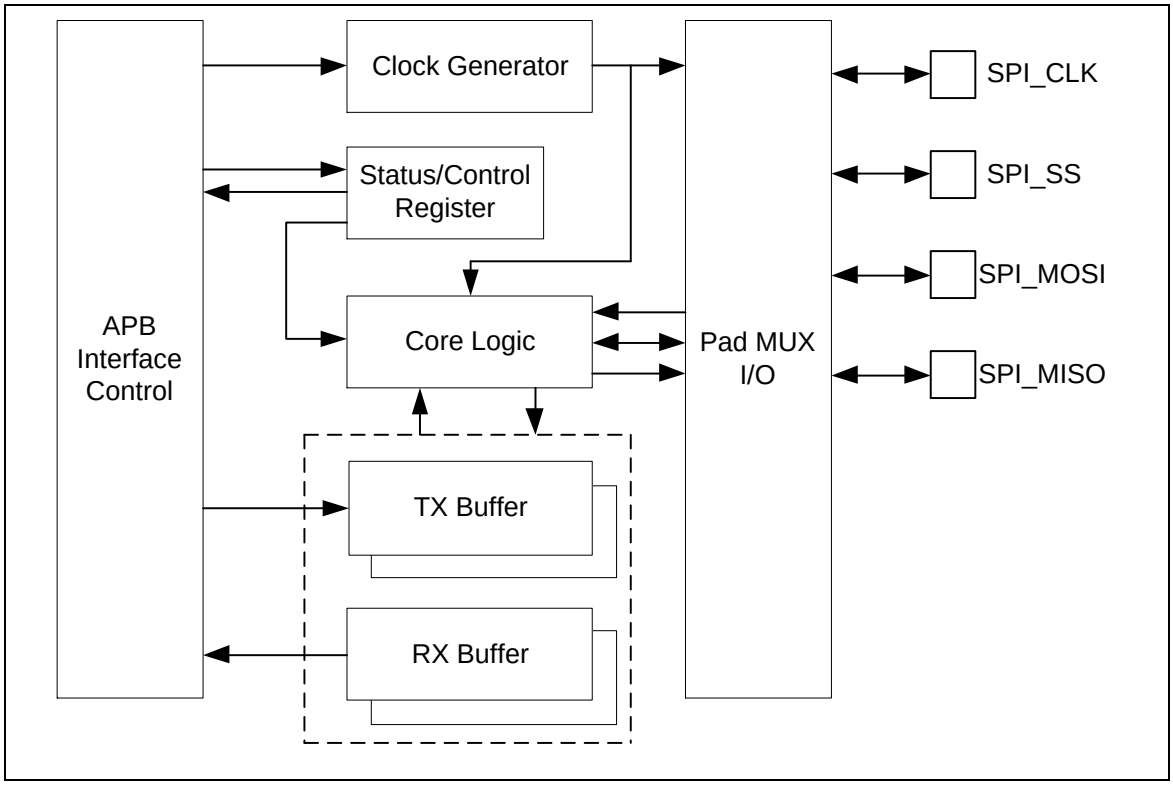


图 6.12-1 SPI 框图

6.12.4 基本配置

SPI引脚在SYS_P0_MFP寄存器中配置。SPI外围时钟可以在CLK_APBCLK[12]中使能，时钟源可以在CLK_CLKSEL1[5:4]选择。

6.12.5 功能描述

6.12.5.1 术语

SPI外设时钟和串行总线时钟

SPI控制器需要SPI外围时钟来驱动SPI逻辑单元从而执行数据传输。SPI总线时钟是SPI_CLK引脚上提供的时钟。SPI外设时钟频率是通过设置时钟源、DIVMOD (SPI_SLVCTL[31])和时钟除数(SPI_CLKDIV[7:0])来决定的。CLK_CLKSEL1寄存器中的SPISEL位决定了SPI外围设备时钟。设置DIVMOD位为0，可兼容早先产品的SPI时钟速率计算。通过设置SPI_CLKDIV寄存器中DIVIDER来决定时钟频率计算的除数。

在SPI主机模式下，SPI外围时钟等于SPI总线时钟。

在SPI从机模式下，SPI总线时钟由片外主机设置提供。从机设备的SPI外围时钟频率必须快于连接在一起的主设备的总线时钟频率。无论是在主机还是从机模式下，SPI外围时钟频率不能比APB时钟快。

主机/从机 模式

SPI控制器可以通过设置SLAVE 位 (SPI_CTL[18])配置成主机或从机模式，来与片外SPI主机或者从机来通讯，主机模式和从机模式的应用框图如下所示。

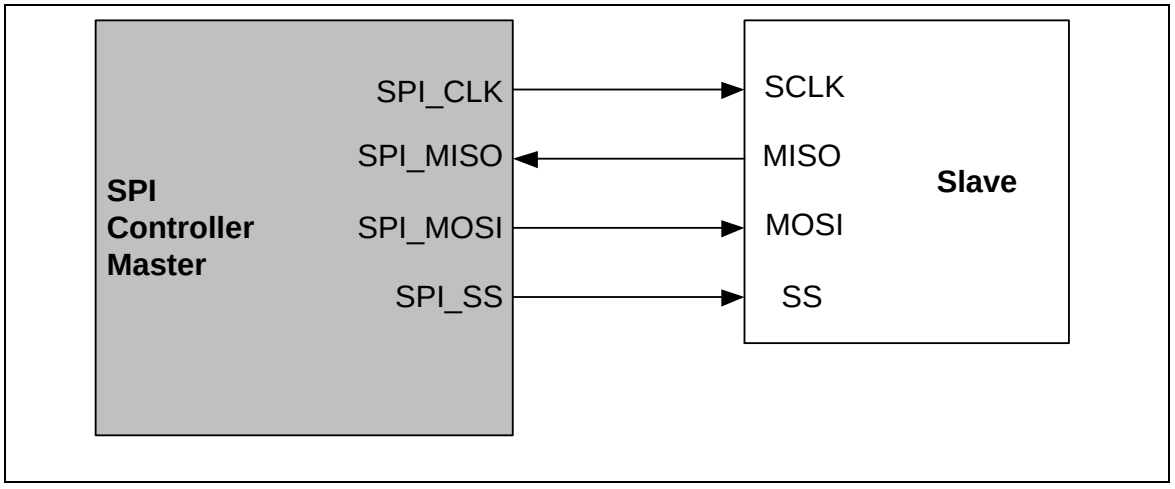


图 6.12-2 SPI 主机模式应用框图

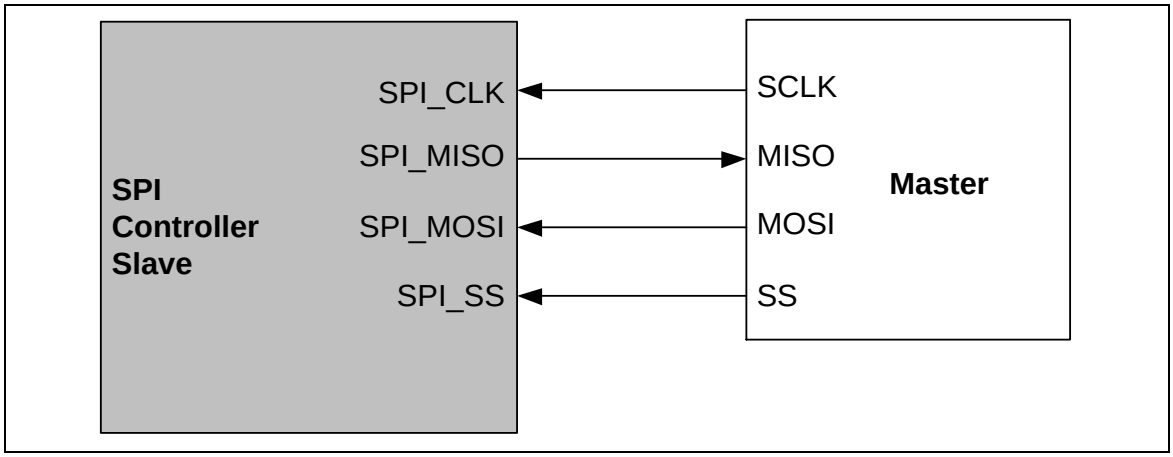


图 6.12-3 SPI 从机模式应用框图

时钟极性

CLKPOL (SPI_CTL[11])定义了总线时钟的空闲状态。如果 CLKPOL = 1，SPI_CLK输出为空闲高电平；否则CLKPOL = 0时，输出低电平。

发送/接收位长度

传输字的长度定义在DWIDTH (SPI_CTL[7:3])中。在传送和接收时，它可以最长被配置为32位。

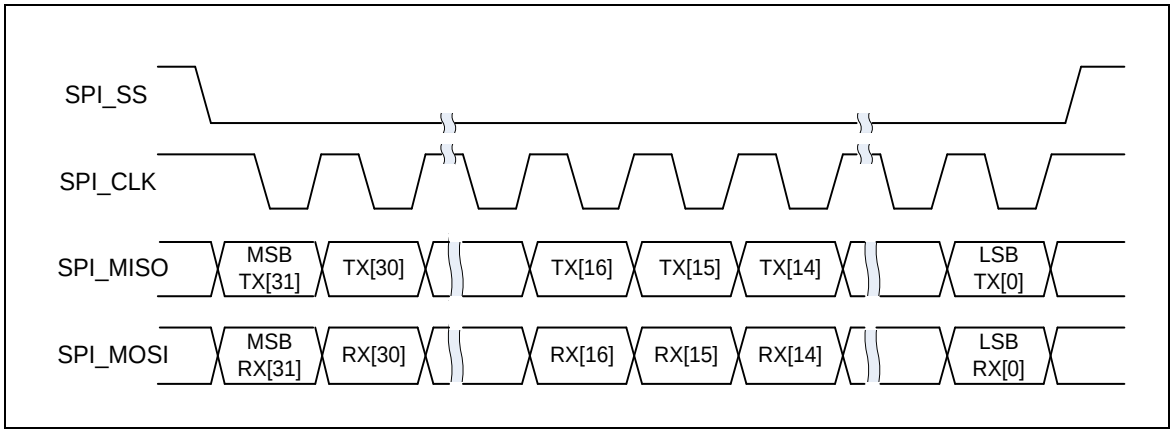


图 6.12-4 一次传输32-位

LSB/MSB 优先

LSB (SPI_CTL[10])定义了数据在发送/接收时是LSB优先还是MSB优先 。

发送沿

TXNEG (SPI_CTL[2]) 定义了数据在发送时，是在串行时钟SPI_CLK的下降沿还是上升沿。

接收沿

RXNEG (SPI_CTL[1]) 定义了数据在接收时，是在串行时钟SPI_CLK的下降沿还是上升沿。

注: TXNEG 和 RXNEG 的设置是互斥的。换句话说，就是发送的接收不能设置为相同的同名沿。

字休眠

在主机模式下，SUSPITV (SPI_CNTRL[15:12])可配置两个连续传输字之间的休眠间隔。休眠间隔定义为从前一次传输字的最后一个时钟沿到下一次传输字的第一个时钟沿。SUSPITV的缺省值为0x3 (3.5 个串行时钟周期)。如果软件关闭FIFO模式，SUSPITV的设定将不起作用。

从机选择

在主机模式下，该SPI控制器能通过从机选择输出脚SPI_SS来驱动片外从机设备。在从机模式下，片外主机设备通过SPI_SS输入端口驱动从机选择信号到SPI控制器。在主机/从机模式下，从机选择信号的有效状态可以通过编程寄存器SSACTPOL(SPI_SSCTL[2])来设定为低有效或高有效，SSLTEN(SPI_SSCTL[4])定义了从机选择信号SPI_SS为电平触发还是边沿触发。触发条件的选择取决于所连设备的从机/主机的类型。

在从机模式下，如果SSLTEN (SPI_SSCTL[4])位被配置成电平触发，则寄存器LTF (SPI_SSCTL[5])用来表示在一个传输完成后接收到的数据数量和位数量是否满足寄存器DWIDTH (SPI_CTL[7:3])的设定值（传输完成是指当从机选择信号未激活或SPI控制器完成了一次数据传输时，单位传输中断标志置1）。

电平触发 / 边沿触发

在从机模式下，从机选择信号可以被配置为电平触发或边沿触发。边沿触发模式，数据从一个有效的从机选择信号边沿开始，到一个无效的从机选择信号边沿结束。当检测到一个无效边沿，寄存器UNITIF (SPI_CTL[16])的单元传输中断标志将设置为1。如果主机没有发送一个无效边沿给从机，传输过程不会完成，从机的单元传输中断标志不会被置位。电平触发模式，当有下面的两个条件中的一个发生，从机的单元传输中断标志将会被置位。第一个条件是传输的数据位与DWIDTH (SPI_CTL[7:3])的设定值相匹配时，从机的单元传输中断标志将会被置位。第二个条件，如果主机正在传输期间的时候设置从机选择管脚为无效电平，将会强制从机设备终结当前传输，而不管已经传输了多少位数据，从机的单元传输中断标志将会被置位。用户可以读取LTF位来检查数据是否已经全部传完。

6.12.5.2 自动从机选择

在主机模式下，如果AUTOSS (SPI_SSCTL[3])设置为1，从机选择信号将会自动产生，并根据SS (SPI_SSCTL[0])是否使能，将从机选择信号输出到SPI_SS管脚上。这意味着当通过设置SPIEN bit (SPI_CTL[0])来开始数据传输时，在SS (SPI_SSCTL[0])使能的从机选择信号将由SPI控制器自动设置为有效状态，在数据传输结束后自动被设置为无效状态。如果AUTOSS被清零，从机选择输出信号需要通过手工置位/清除SS (SPI_SSCTL[0])，从而使从机进入激活或非激活状态。从机选择输出信号的激活电平状态由SSACTPOL (SPI_SSCTL[2])来定义。

6.12.5.3 字节重排序功能

当传输设置为MSB优先(LSB bit (SPI_CTL[10]) = 0)且使能字节重排序功能时，存储在Tx和Rx缓存中的数据将会按照[Byte0, Byte1, Byte2, Byte3]重新排序，前提是位长度被配置为32位(DWIDTH (SPI_CTL[7:3]) = 0)。发送/接收数据的顺序将会变为Byte0, Byte1, Byte2, Byte3。如果DWIDTH (SPI_CTL[7:3]))设置为24位，Tx和Rx缓存中的数据将会按照[未知Byte,Byte0, Byte1, Byte2]重新排序，SPI控制器将会按照Byte0, Byte1,Byte2的顺序发送/接收数据。每一个字节将会按照MSB优先的规则来发送/接收。16位模式的规则与上面相同。字节重排序功能只在 DWIDTH (SPI_CTL[7:3])为16, 24和 32 位时适用。

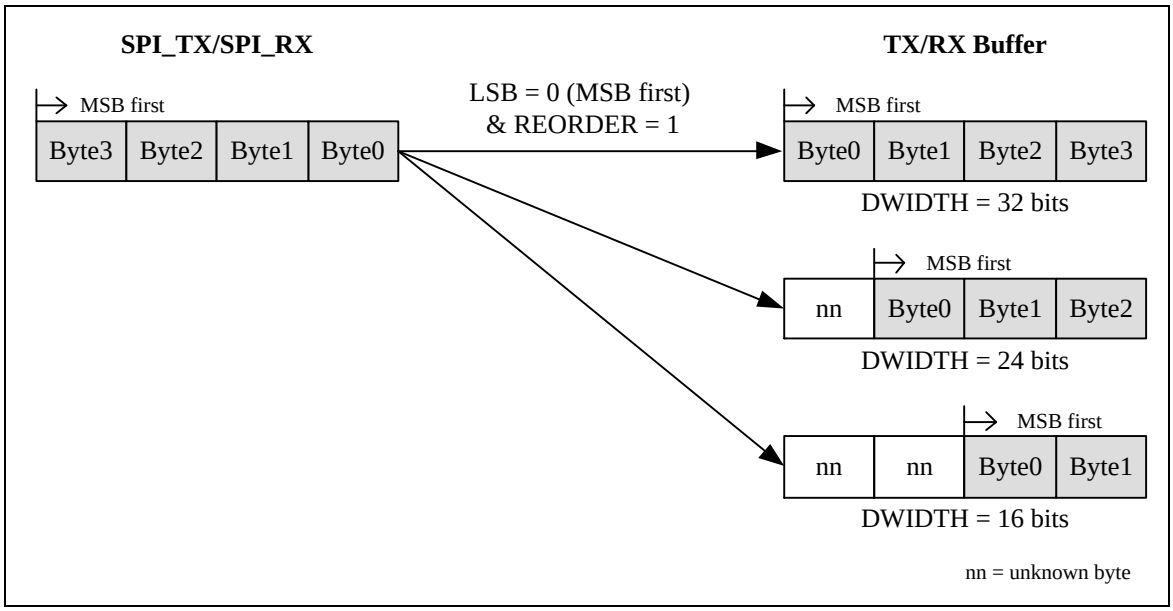


图 6.12-5 字节重排序

6.12.5.4 字节休眠功能

字节休眠功能与字休眠功能都在SUSPITV (SPI_CTL[15:12])中进行设置。在主机模式下，如果通过设置REORDER (SPI_CTL[19])为1来使能字节重排序功能，硬件将会在两个连续传输字节之间插入一个0.5 ~ 15.5个串行时钟周期的休眠间隔。DWIDTH (SPI_CTL[7:3])可以设置为16，24和32位。

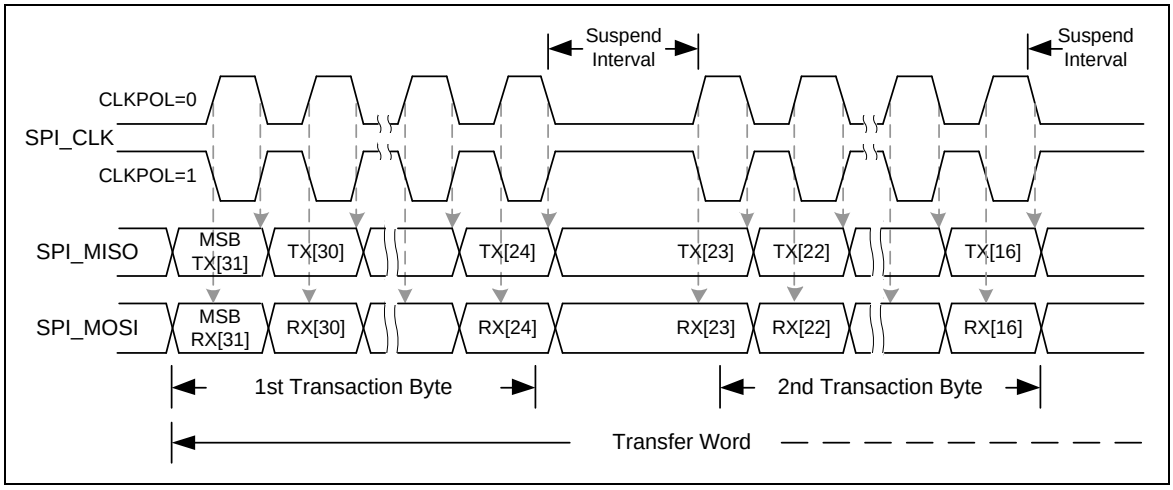


图 6.12-6 字节休眠时序波形

6.12.5.5 *从机3线模式*

当SLV3WIRE (SPI_SLVCTL[8]) 被软件置1使能从机3线模式时，在从机模式下SPI控制器可以在没有从机选择信号下工作。SLV3WIRE仅在从机模式下有效。在与一个主机通讯时，仅需要三个管脚，SPI_CLK, SPI_MISO和SPI_MOSI。SPI_SS脚可以配置成一个GPIO。当SLV3WIRE被设为 1时，在 SPIEN被置为1后，SPI从机将开始准备发送和接收数据。当接收数据位的个数满足DWIDTH (SPI_CTL[7:3])长度定义时，单元传输中断标志UNITIF (SPI_CTL[16])将会置位。

注: 在从机3线模式下，SSLTEN (SPI_SSCTL[4])应该被设置为1。

6.12.5.6 *FIFO 模式*

当 FIFOEN (SPI_CTL[21])被设为1时，SPI控制器支持FIFO模式。SPI控制器配备了4个32位发送和接收FIFO缓存。

发送FIFO缓存是一个4层深度，32位宽，先进先出的寄存器缓存。数据可以通过软件写SPI_TX寄存器写入发送FIFO缓存。存储在发送FIFO缓存的数据会被传输控制逻辑读取并发送出去。如果4层发送FIFO缓存满了，TXFULL(SPI_CTL[27])位会被置1。当SPI传输逻辑单元发送FIFO缓存中的最后一个数据时，那么4层发送FIFO缓存为空，TX_EMPTY(SPI_CTL[26])位被置1。注意最后一笔传输还在进行时TX_EMPTY (SPI_CTL[26]) 标志已被置1。

接收FIFO缓存也是一个4层深度，32位宽，先进先出的寄存器缓存。接收控制逻辑存储接收到的数据到该缓存。FIFO缓存数据可以通过软件从SPI_RX 寄存器读取。FIFO还带有一些相关的状态位，像RXEMPTY (SPI_CTL[24])和RXFULL (SPI_CTL[25])，用来表明当前FIFO缓存的状态。

在FIFO模式下，发送和接收阀值可以通过软件设置TXTH (SPI_FIFOCTL[29:28])和 RXTH (SPI_FIFOCTL[25:24])来设定。当存储在发送FIFO缓存的有效数据计数小于或等于TXTH设定，TXTHIF (SPI_STATUS[4])位会被置1。当存储在接收FIFO缓存的有效数据计数大于RXTH设定，RXTHIF (SPI_STATUS[0])位会被置1。

在 FIFO模式下, 4个数据可以事先通过软件写入SPI发送FIFO缓存。当SPI控制器工作在FIFO模式下时，SPI_CTL 寄存器中的SPIEN位由硬件控制，SPI_CTL 寄存器的内容不会被软件修改，除非FIFOEN位被清零使FIFO模式禁止。

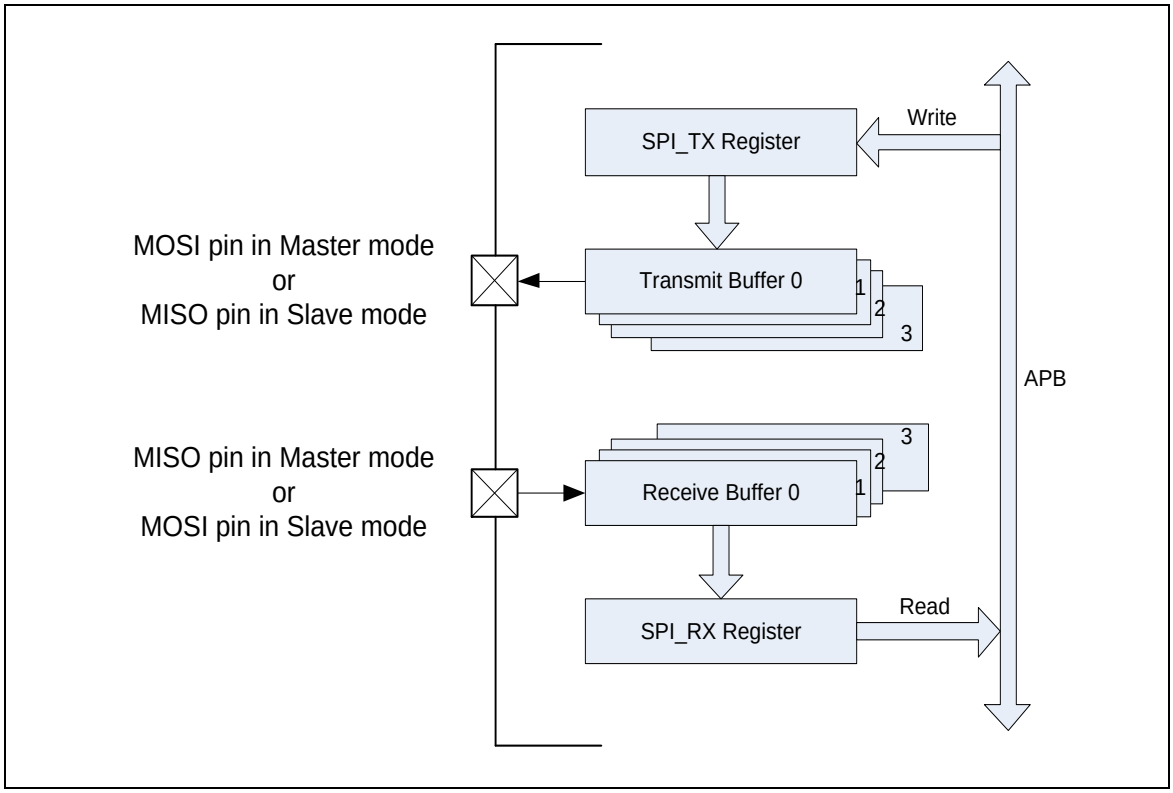


图 6.12-7 FIFO 模式框图

SPI 主机模式和使能FIFO 模式

在主机模式下，进行发送操作中，当FIFOEN 位 (SPI_CTL[21]) 设置为1，软件写第一个数据到SPI_TX 寄存器时，TXEMPTY (SPI_CTL[26])标志将会被清零。只要发送 FIFO 缓存非空，发送操作立即开始。用户可以立即写入下一个要发送的数据到SPI_TX中。在 FIFO 模式下，SPI 控制器将会在两个连续的事务之间插入一个休眠间隔，休眠间隔的长度由SUSPITV (SPI_CTL [15:12])的设定值决定。只要 TXFULL (SPI_CTL[27])标志为 0，用户就可以写数据到SPI_TX寄存器。

如果要发送的数据更新及时，接下来的事务将会被自动触发。如果在所有数据传输完成之后， SPI_TX 寄存器没有被更新，则传输停止。

在主机接收操作中，串行数据从SPI_MISO管脚接收并被存储在接收 FIFO 缓存。当接收 FIFO 缓存中包含未读的数据时，RXEMPTY (SPI_CTL[24])将会被清除为 0。只要RXEMPTY (SPI_CTL[24])标志为 0，软件就可以从SPI_RX寄存器读取接收到的数据。如果接收FIFO缓存包含4个未读数据，RXFULL (SPI_CTL[25])标志将会被设置为1。此时，SPI控制器将会停止接收数据，直到软件读取SPI_RX寄存器。

SPI从机模式和使能FIFO模式

从机模式下，当FIFOEN 位 (SPI_CTL[21])被设置为 1，SPIEN位(SPI_CTL[0])将会被硬件自动设置为 1。

在从机发送操作中，当软件写数据到SPI_TX寄存器，数据将会被加载到发送FIFO缓存，且 TXEMPTY (SPI_CTL[26]) 标志将会被清0。当从设备从主机接收到时钟信号，发送操作将会开始。只要 TXFULL(SPI_CTL[27])标志为 0，软件就可以写数据到SPI_TX寄存器。在所有数据都被 SPI 发送逻辑单元发送出去，且软件没有更新 SPI_TX0寄存器，TXEMPTY(SPI_CTL[26])标志将会被设置为 1。

在从机接收操作中，串行数据从SPI_MOSI管脚被接收，并被存储到接收 FIFO 缓存寄存器。接收机制与主机模式接收操作类似。

6.12.5.7 中断

SPI 单位传输中断

当SPI控制器完成一个单位传输，单位传输中断标志UNITIF (SPI_CTL[16])将会被置位。如果中断使能位UNITIEN (SPI_CTL[17])被置位，则单位传输中断事件将会给CPU产生中断。单位传输中断标志位只能写 1 清零。

SPI 从机3线模式起始中断

在3线模式下，当从机检测到SPI 时钟信号时，3 线模式会产生开始中断标志，SLVSTIF (SPI_SLVCTL[11])将会被置为 1。如果 SLVSTIEN (SPI_SLVCTL[10]) 被设置为 1，SPI控制器将会触发一个中断。如果接收到的数据位小于 DWIDTH (SPI_CTL[7:3])的设定要求，在由用户定义的期望的时间内再没有串行时钟输入，用户可以设置 SLVABT bit (SPI_SLVCTL[9])位来中止当前传输。如果软件设置SLVABT位为 1，单元传输中断标志UNITIF将会被置位。

接收FIFO超时中断

在FIFO 模式下，有超时功能通知用户。如果超时中断使能位RXTOIEN (SPI_FIFOCTL[21])置1，在FIFO里有一个接收到的数据，并且没有被软件读取。在主机模式下超过64个SPI外设时钟周期，或在从机模式下超过576个SPI外设时钟周期，会发出一个超时中断。

传送 FIFO 中断

在FIFO模式下，如果发送FIFO缓存的有效数据计数小于或等于TXTH (SPI_FIFOCTL[29:28])的设定值，发送FIFO中断标志会被置1。如果XTHIEN (SPI_FIFOCTL[3])置1，发送FIFO中断使能，SPI控制器会产生一个发送FIFO中断到系统。

接收 FIFO 中断

在FIFO 模式下，如果接收FIFO缓存的有效数据计数大于RXTH (SPI_FIFOCTL[25:24])的设定值，接收FIFO中断标志会被置1。如果RXTHIEN (SPI_FIFOCTL[2])置1，接收FIFO中断使能，SPI控制器会产生一个接收FIFO中断到系统。

6.12.6 时序图

从机选择信号的有效状态可以由 SSACTPOL(SPI_SSCTL[2])位和SSLTEN (SPI_SSCTL[4])位来定义。串行时钟 (SPI_CLK) 的空闲状态可以通过 CLKPOL (SPI_CTL[11]) 配置为高电平或低电平。传输字段长度在DWIDTH (SPI_CTL[7:3])中定义，发送/接收数据是以 MSB 或 LSB 优先由 LSB 位(SPI_CTL[10])定义。用户可以通过设置TXNEG/RXNEG (SPI_CTL[2:1])寄存器来选择发送/接收数据时串行时钟的边沿。四个 SPI 发送/接收及相关设置如下。

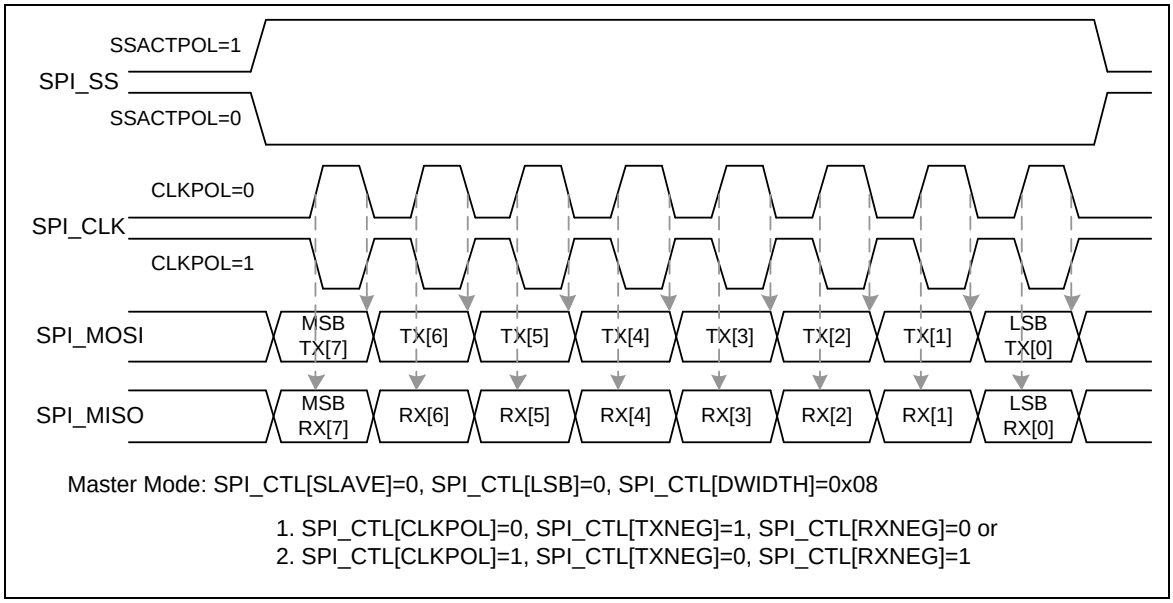


图 6.12-8 SPI 主机模式下的时序

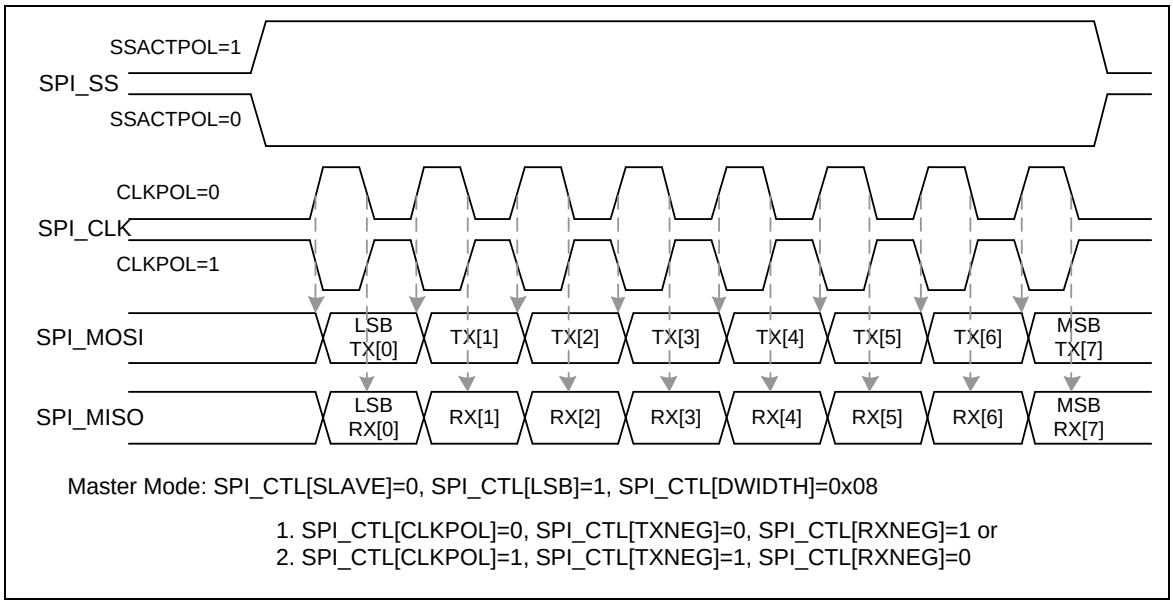


图 6.12-9 SPI主机模式下的时序 (交替SPI_CLK相位)

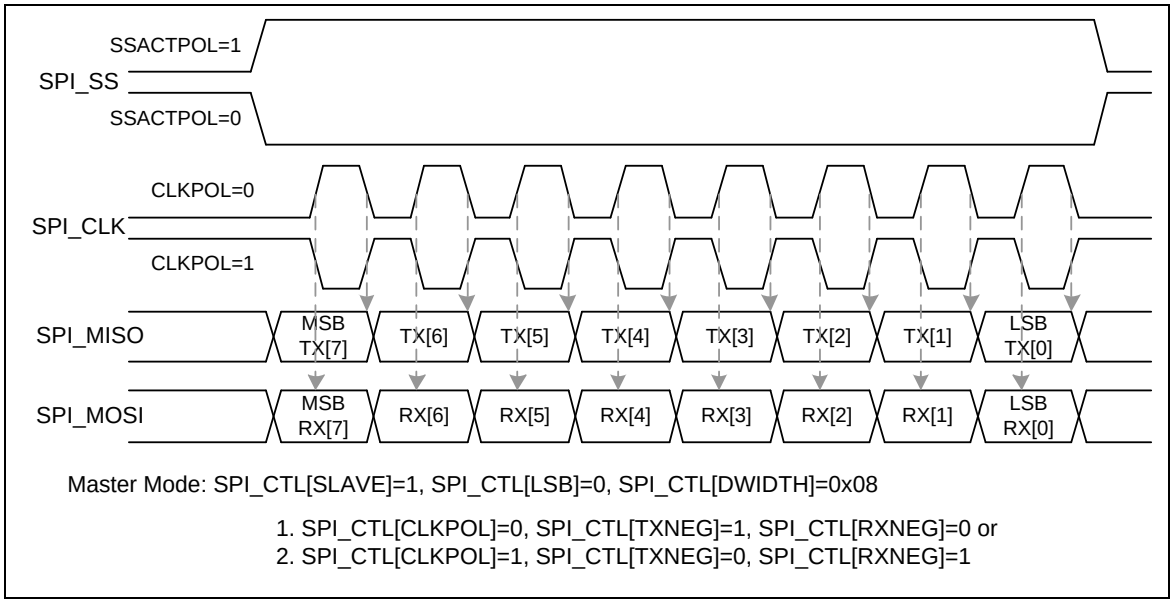


图 6.12-10 SPI从机模式下的时序

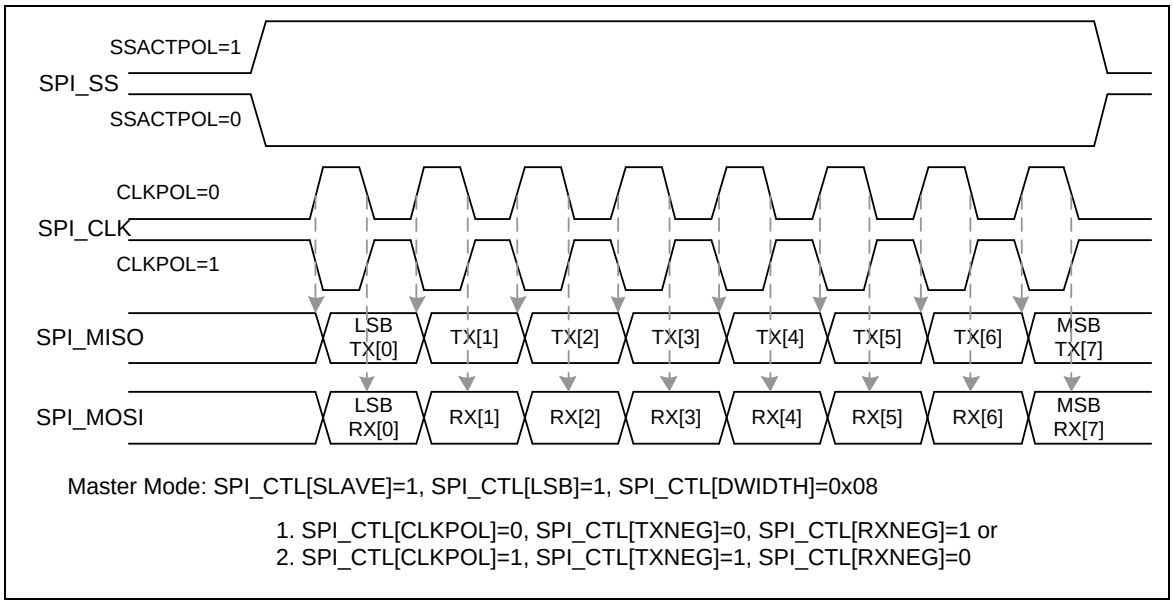


图 6.12-11 SPI从机模式下的时序(交替SPI_CLK相位)

6.12.7 编程示例

例 1: SPI 控制器作为主机去访问一个片外从机设备，过程如下：

- 数据在串行时钟上升沿锁存
- 数据在串行时钟下降沿传输
- MSB 先传输
- 数据宽度8位

- SPI_CLK空闲模式为低电平状态
- 每次只发送/接收一个字节
- 与一个片外从机相连。从机选择信号为低电平有效

操作流程如下：

1) 设置 DIVIDER (SPI_DIVIDER[7:0]) 寄存器来决定串行时钟输出频率。
2) 写一个合适的值到SPI_CTL寄存器来控制SPI主机操作
   1. 通过SLAVE (SPI_CTL[18] = 0) 设置SPI控制器作为主机
   2. 通过CLKPOL (SPI_CTL[11] = 0)设置串行时钟空闲状态为低电平
   3. 通过TXNEG (SPI_CTL[2] = 1)设置数据在串行时钟下降沿传输
   4. 通过RXNEG (SPI_CTL[1] = 0)设置数据在串行时钟上升沿锁存
   5. 通过DWIDTH(SPI_CTL[7:3] = 0x08)设置一次字传输的长度为8位
   6. 通过LSB (SPI_CTL[10] = 0)设置MSB优先， 不需要在意 SUSPITV (SPI_CTL[15:12]) 因为这种情况下它不是FIFO模式
3) 给SPI_SSCTL寄存器写入合适的值来实现主机的相关设置
   1. 清除自动从机选择位AUTOSS(SPI_SSCTL[3] = 0).

      在从机选择激活电平控制位SSACTPOL (SPI_SSCTL[2] = 0)设置低电平来触发从机选择信号。
   2. 通过设置从机选择控制位SS (SPI_SSCTL[0])来设置从机选择信号，来激活片外从机设备。
4) 若SPI主机尝试发送(写)一个字节的数据到片外从机设备，则将所要发送的数据写入SPI_TX寄存器。
5) 如果 SPI 主机只是要从片外从机设备接收（读）一个字节的数据，不必管被传输出去的数据是什么，寄存器SPI_TX不需要通过软件更新。
6) 使能SPIEN (SPI_CTL[0] = 1)来开始SPI数据传输。
7) 等待 SPI 中断发生（如果中断使能位UNITIEN bit (SPI_CTL[17])被使能）或轮询检测 SPIEN (SPI_CTL[0]) 位直到被硬件自动清零。
8) 从SPI_RX[7:0]中读出接收到的一个字节数据。
9) 重复步骤 4) 继续其他数据传输或设置SS (SPI_SSCTL [0])来停止片外从机设备。

例 2: SPI 控制器作为从机设备和一片外主机设备相连。外设主机通过SPI 接口与从机通信。过程如下：

- 数据在串行时钟上升沿锁存
- 数据在串行时钟下降沿传输
- LSB优先传输
- 数据宽度8位
- SPI_CLK空闲状态为高电平
- 每次发送/接收一个字节
- 从机选择信号为高电平触发

操作流程如下：

1) 设置DIVIDER (SPI_CLKDIV[7:0]) 来决定从机外设时钟频率。该频率必须大于SPI总线时钟频率。

2) 给寄存器SPI_SSCTL写入合适的值来实现从机模式的相关设置。

   设置从机选择有效电平位SSACTPOL (SPI_SSCTL[2] = 1)和从机选择触发电平位SSLTEN (SPI_SSCTL[4] = 1).来选择高电平触发作为从机选择信号。

3) 通过设置SPI_CTL寄存器来控制 SPI 从机的行为

   1. 通过SLAVE(SPI_CTL[18] = 1)设置SPI控制器作为从机设备
   2. 通过CLKPOL(SPI_CTL[11] = 1)设置串行时钟的空闲状态为高电平
   3. 通过TXNEG (SPI_CTL[2] = 1)设置数据在串行时钟的下降沿传输
   4. 通过RXNEG(SPI_CTL[1] = 0)设置数据在串行时钟的上升沿锁存
   5. 通过DWIDTH(SPI_CTL[7:3] = 0x08)设置一次字传输的长度为8位
   6. 通过LSB(SPI_CTL[10] = 1)设置LSB优先传输

4) 如果 SPI 从机要发送(被读)一个字节的数据到片外主机，则将所要发送的数据写入到SPI_TX寄存器。

5) 如果 SPI 从机只是要从片外主机接收(被写)一字节数据，用户不必关心什么数据将被传输，不需要软件更新SPI_TX寄存器。

6) 设置 SPIEN (SPI_CTL[0] = 1)来等待片外主机设备的从机选择触发输入和串行时钟输入，以便开始在 SPI 接口的数据传输。

7) 等待 SPI 中断发生（如果中断使能位UNITIEN(SPI_CTL[17])被使能）或轮询检测SPIEN bit (SPI_CTL[0])位直到被硬件自动清零。

8) 从寄存器SPI_RX[7:0]中读取接收到的一个字节数据。

9) 重复步骤4) 来继续其它数据传输或清除SPIEN(SPI_CTL[0]) 来停止数据传输。

6.12.8 寄存器映射

R: 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移量	读/写	描述	复位值
SPI 基地址: **SPI_BA = 0x4003_0000**				
SPI_CTL	SPI_BA+0x00	R/W	SPI 控制和状态寄存器	0x0500_3004
SPI_CLKDIV	SPI_BA+0x04	R/W	SPI 时钟分频寄存器	0x0000_0000
SPI_SSCTL	SPI_BA+0x08	R/W	SPI 从机选择寄存器	0x0000_0000
SPI_RX	SPI_BA+0x10	R	SPI 数据接收寄存器	0x0000_0000
SPI_TX	SPI_BA+0x20	W	SPI 数据发送寄存器	0x0000_0000
SPI_SLVCTL	SPI_BA+0x3C	R/W	SPI 从机控制和状态寄存器	0x0000_0000
SPI_FIFOCTL	SPI_BA+0x40	R/W	SPI FIFO 控制寄存器	0x2200_0000
SPI_STATUS	SPI_BA+0x44	R/W	SPI 状态寄存器	0x0500_0000

6.12.9 寄存器描述

SPI 控制和状态寄存器 (SPI_CTL)

寄存器	偏移量	R/W	描述	复位值
SPI_CTL	SPI_BA+0x00	R/W	SPI 控制和状态寄存器	0x0500_3004

31	30	29	28	27	26	25	24
保留				**TXFULL**	**TXEMPTY**	**RXFULL**	**RXEMPTY**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留		**FIFOEN**	保留	**REORDER**	**SLAVE**	**UNITIEN**	**UNITIF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SUSPITV				**CLKPOL**	**LSB**	保留	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DWIDTH					**TXNEG**	**RXNEG**	**SPIEN**

位	描述	
[31:28]	保留	保留.
[27]	**TXFULL**	**发送 FIFO 缓存满标志(只读)** 0 = 发送FIFO缓存没有满 1 = 发送FIFO缓存满 **注:** 该位是SPI_STATUS[27]的相互镜像位
[26]	**TXEMPTY**	**发送 FIFO 缓存空标志 (只读)** 0 = 发送 FIFO 缓存非空 1 = 发送 FIFO 缓存空 **注:** 该位是 SPI_STAUTS[26] 的相互镜像位
[25]	**RXFULL**	**Receive FIFO Buffer Full Indicator (Read Only)** 0 = 接收 FIFO 缓存没有满 1 = 接收 FIFO 缓存满 **注:** 该位是 SPI_STATUS[25] 的相互镜像位
[24]	**RXEMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** 0 = 接收 FIFO 缓存非空. 1 = 接收 FIFO 缓存空. **注:** 该位是 SPI_STATUS[24] 的相互镜像位
[23:22]	保留	保留.
[21]	**FIFOEN**	**FIFO 模式使能位** 0 = FIFO 模式禁用. 1 = FIFO 模式使能. **注 1:** 在使能 FIFO 模式前，其他相关的设置必须事先设定。

		注 2: 在主机模式下，如果 FIFO 模式被使能，在数据被写入4层深度的发送 FIFO 中之后，SPIEN将会自动被硬件设置为 1。当所有存储在发送FIFO缓存的数据全部发送出去，SPIEN会清零。
[20]	保留	保留.
[19]	**REORDER**	字节重排序功能 0 = 禁用. 1 = 使能. **注:**当 DWIDTH定义为16, 24或 32位时，该设置才有效。
[18]	**SLAVE**	从机模式控制 0 = 主机模式. 1 = 从机模式.
[17]	**UNITIEN**	单元发送中断使能位 0 = 禁用. 1 = 使能.
[16]	**UNITIF**	单元发送中断标志 0 = 传输还未完成. 1 = 单元传输完成. **注 1:** 向该位写1清零. **注 2:** 该位是 SPI_STATUS[16] 的相互镜像位
[15:12]	**SUSPITV**	休眠间隔 (仅用于主机) 该四位提供一个在传输过程中连续两个发送/接收事务之间可配置的休眠间隔。休眠间隔是指前一个事务的最后一个时钟边缘和后一个事务的第一个时钟边缘之间的间隔。默认值是 0x3. 休眠间隔长度根据如下公式获得： (SUSPITV[3:0] + 0.5) * SPI_CLK 时钟周期 Example: SUSPITV = 0x0 …. 0.5 SPI_CLK时钟周期 SUSPITV = 0x1 …. 1.5 SPI_CLK时钟周期 …… SUSPITV = 0xE …. 14.5 SPI_CLK时钟周期 SUSPITV = 0xF …. 15.5 SPI_CLK时钟周期
[11]	**CLKPOL**	时钟极性 0 = SPI_CLK 空闲时为低电平 1 = SPI_CLK 空闲时为高电平
[10]	**LSB**	**LSB 优先** 0 = MSB 优先发送/接收 1 = LSB 优先发送/接收
[9:8]	保留	保留.
[7:3]	**DWIDTH**	发送位长度 该位域指定在一个发送/接收事务中，多少个数据位将会被传输。最小位长是 8，最多可以达到 32 位。 DWIDTH = 0x01~0x07 …. 保留 (不使用).

		DWIDTH = 0x08 …. 8 位. DWIDTH = 0x09 …. 9 位. …… DWIDTH = 0x1F …. 31 位. DWIDTH = 0x00 …. 32 位.
[2]	**TXNEG**	**在下降沿发送** 0 = 在SPI_CLK的上升沿发送数据信号 1 = 在SPI_CLK的下降沿发送数据信号
[1]	**RXNEG**	**在下降沿接收** 0 = 在SPI_CLK的上升沿接收数据信号 1 = 在SPI_CLK的下降沿接收数据信号
[0]	**SPIEN**	**SPI 传输控制位和忙状态** 在 FIFO 模式下，该位由硬件控制并且为只读状态。 如果 FIFO 模式被禁止，在整个数据传输过程中，该位保持为 ‘1’。当传输结束，该位自动被清除。 0 = 如果SPI正在传输时，向该位写0来停止传输。 1 = 在主机模式下，写 1 到该位开始 SPI 数据传输。在从机模式下，写 1 到该位表示从机已经准备好与主机进行通信。 **注 1:**.当FIFO模式被禁用时，在写1到SPIEN位时，所有的配置必须事先配置好。 **注 2:** 在SPI从机模式下，如果FIFO模式被禁止，在数据传输时SPI总线时钟会保持在空闲状态，当从机选择信号处于非活动状态时，SPIEN不会被清零。

SPI 时钟分频寄存器 (SPI_CLKDIV)

寄存器	偏移量	R/W	描述	复位值
SPI_CLKDIV	SPI_BA+0x04	R/W	SPI 时钟分频寄存器	0x0000_0000

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

位	描述	
[31:8]	保留	保留.
[7:0]	**DIVIDER**	**时钟分频寄存器 (仅用于主机)** 该区域的值是一个频率除数，来决定SPI外设时钟频率f_{spi}，SPI主机的时钟频率由SPI_CLK管脚输出。该频率可由下面的公式获得： 当 DIVMOD, SPI_SLVCTL[31]为 0时， $f_{spi} = \frac{f_{SPI_clock_src}}{(DIVIDER+1)*2}$ 否则当 DIVMOD 为 1时, $f_{spi} = \frac{f_{SPI_clock_src}}{(DIVIDER+1)}$ 这里 $f_{SPI_clock_src}$ 是SPI外设时钟源，定义在寄存器CLK_CLKSEL1 中。

SPI 从机选择寄存器 (SPI_SSCTL)

寄存器	偏移量	R/W	描述	复位值
SPI_SSCTL	SPI_BA+0x08	R/W	SPI 从机选择寄存器	0x0000_0000

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

位	描述	
[31:6]	保留	保留.
[5]	**LTF**	**电平触发标志 (只读, 仅用于从机)** 当SSLTEN位在从机模式下被置位，读该位的值可用来表示一次传输完成后接收位的数量是否达到要求。 0 = 一次传输的个数或位长度不满足指定的要求 1 = 传输的个数及位长度满足 DWIDTH 定义的要求
[4]	**SSLTEN**	**从机选择电平触发使能位 (仅用于从机)** 0 = 输入从机选择信号为边沿触发。 1 = 输入从机选择信号为电平触发。
[3]	**AUTOSS**	**自动从机选择功能使能位(仅用于主机)** 0 = 置位/清除SS位，会使SPI_SS管脚信号将会被激活/失效。 1 = SPI_SS管脚信号由硬件自动产生，这意味着当通过设置SPIEN来开始发送/接收时，从机选择信号将会由SPI控制器激活，当发送/接收完成时，该信号又会处于无效状态。
[2]	**SSACTPOL**	**从机选择活动状态电平 (仅用于从机)** 该位定义了活动状态下的从机选择信号(SPI_SS) 如果SSLTEN 位为 1: 0 = 从机选择信号SPI_SS在低电平下处于活动状态 1 = 从机选择信号SPI_SS在高电平下处于活动状态 如果SSLTEN 位为 0: 0 = 从机选择信号SPI_SS在下降沿下处于活动状态 1 = 从机选择信号SPI_SS在上升沿下处于活动状态
[1]	保留	保留.
[0]	**SS**	**从机选择控制位 (仅用于主机)** 如果AUTOSS 位为 0,

		0 = 设置 SPI_SS 为非活动状态 1 = 设置 SPI_SS 为活动状态 如果 AUTOSS 位为 1, 0 = 保持 SPI_SS 为非活动状态 1 = SPI_SS在发送/接收的时间内被自动驱动到激活状态，而其他时间为非激活状态。SPI_SS的激活状态类型由SSACTPOL指定。

SPI 数据接收寄存器 (SPI_RX)

寄存器	偏移量	R/W	描述	复位值
SPI_RX	SPI_BA+0x10	R	SPI 数据接收寄存器	0x0000_0000

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

位	描述	
[31:0]	**RX**	**数据接收寄存器 (Read Only)** 数据接收寄存器内保存最后一次传输所接收到的数据。有效的位取决于SPI_CTL寄存器中的数据的长度DWIDTH内的配置来决定。 例如，如果DWIDTH配置为0x08，RX[7:0]内保存了收到的数据，其它位的数据未知，数据接收寄存器是只读寄存器。

SPI 数据发送寄存器 (SPI_TX)

寄存器	偏移量	R/W	描述	复位值
SPI_TX	SPI_BA+0x20	W	SPI 数据发送寄存器	0x0000_0000

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

位	描述	
[31:0]	**TX**	**数据传送寄存器** 数据发送寄存器保存下一次将要发送的数据。有效的位取决于SPI_CTL寄存器中的数据的长度DWIDTH内的配置来决定。 例如，如果DWIDTH为0x08，TX[7:0]内的值将会在下一次传输中被发送出去。

SPI 从机控制和状态寄存器 (SPI_SLVCTL)

寄存器	偏移量	R/W	描述	复位值
SPI_SLVCTL	SPI_BA+0x3C	R/W	SPI 从机控制和状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DIVMOD	保留						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留							**SSINAIEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留				**SLVSTIF**	**SLVSTIEN**	**SLVABT**	**SLV3WIRE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留							

位	描述	
[31]	**DIVMOD**	**时钟配置时钟向后兼容选项** 0 = 时钟配置向后兼容. 1 = 时钟配置不向后兼容. **注:** 详情参考 SPI_CLKDIV 中的描述
[30:17]	保留	保留.
[16]	**SSINAIEN**	**从机选择无效中断选(仅用于从机)** 0 = 当从机选择信号变为无效电平时， IF 位不会被置 1。 1 = 当从机选择信号变为无效电平时， IF 位会被置 1。 **注:**该设置仅在SPI控制器配置为电平触发从机设备时有效。
[15:12]	保留	保留.
[11]	**SLVSTIF**	**从机3线模式起始中断状态 (仅用于从机)** 该位用来表示在3线模式下，传输是否已经开始。 0 = 自从SLVSTIEN置1， 从机没有侦测到任何SPI 总线时钟。 1 = 在3线模式下，传输已经开始。 **注 1:** 该位在传输完成后自动清位或写 1 清位。 **注 2:** 它是SPI_STATUS[11]的相互镜像位。
[10]	**SLVSTIEN**	**从机3-线模式开始中断使能位 (仅用于从机)** 在3线模式下传输开始时，该位用于使能中断。如果在传输开始后由用户定义的时间内没有传输完成的中断，则用户可以置位SLVABT位强制完成传输。 0 = 禁用传输开始中断 1 = 使能传输开始中断 **注:** 该位在传输完成后清零或SLVSTIF位被清除后清零。
[9]	**SLVABT**	**从机3线模式终止控制位 (仅用于从机)** 在正常操作中，当接收到数据符合DWIDTH的要求位的值时，会有中断事件。

		如果接收到的位数少于要求的值而且在3线模式的从机模式下，在一次传输的时间后没有更多的总线时钟输入时，用户可以设定该位来强制完成当前的传输，然后用户就可以收到一个传输完成的中断事件。 0 = 当SLV3WIRE 设置为1时，不强制完成传输。 1 = 当SLV3WIRE 设置为1时，强制完成传输。 **注:**软件设置该位为1后，该位会被硬件自动清零。
[8]	**SLV3WIRE**	**从机 3-线模式使能位 (仅用于从机)** SPI控制器工作在3线接口，包含SPI_CLK, SPI_MISO和SPI_MOSI。 0 = 在从机模式下，控制器为4线双向接口。 1 = 在从机模式下，控制器为3线双向接口。控制器在SPIEN设置为1后，将会做好发送/接收准备。 **注:** 在3线模式下， SSLTEN (SPI_SSCTL[4]) 应该被置1.
[7:0]	保留	保留.

SPI FIFO 控制寄存器 (SPI_FIFOCTL)

寄存器	偏移量	R/W	描述	复位值
SPI_FIFOCTL	SPI_BA+0x40	R/W	SPI FIFO 控制寄存器	0x2200_0000

31	30	29	28	27	26	25	24
保留		**TXTH**		保留		**RXTH**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留		**RXTOIEN**	保留				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留	**RXOVIEN**	保留		**TXTHIEN**	**RXTHIEN**	**TXRST**	**RXRST**

位	描述	
[31:30]	保留	保留.
[29:28]	**TXTH**	**发送 FIFO阈值** 如果发送FIFO缓存的有效数据数量小于或者等于TXTH，TXTHIF 将会被设置为 1， 否则TXTHIF将会被设置为 0。
[27:26]	保留	保留.
[25:24]	**RXTH**	**接收 FIFO 阈值** 如果 接收FIFO缓存有效数据数量大于RXTH，RXTHIF将会被设置为 1， 否则RXTHIF将会被设置为 0。
[23:22]	保留	保留.
[21]	**RXTOIEN**	**接收 FIFO 超时中断使能 位** 0 = 禁用超时中断 1 = 使能超时中断
[20:7]	保留	保留.
[6]	**RXOVIEN**	**接收 FIFO Overrun中断使能位** 0 = 禁用接收FIFO overrun 中断 1 = 使能接收FIFO overrun 中断
[5:4]	保留	保留.
[3]	**TXTHIEN**	**发送阈值中断使能位** 0 = 禁用发送阈值中断 1 = 使能发送阈值中断
[2]	**RXTHIEN**	**接收阈值中断使能** 0 = 禁用接收阈值中断 1 = 使能接收阈值中断

[1]	**TXRST**	**清发送FIFO 缓存** 0 = 无影响 1 = 清除传送FIFO缓存 **注:**在软件写1和传送FIFO清除后，该位由硬件清零。
[0]	**RXRST**	**清接收FIFO 缓存** 0 = 无影响 1 = 清除接收FIFO缓存 **注:** 在软件写1和传送FIFO清除后，该位由硬件清零。

SPI 状态寄存器 (SPI_STATUS)

寄存器	偏移量	R/W	描述	复位值
SPI_STATUS	SPI_BA+0x44	R/W	SPI 状态寄存器	0x0500_0000

31	30	29	28	27	26	25	24
TXCNT				TXFULL	TXEMPTY	RXFULL	RXEMPTY
23	22	21	20	19	18	17	16
保留			SLVTOIF	保留			UNITIF
15	14	13	12	11	10	9	8
RXCNT				SLVSTIF	保留		
7	6	5	4	3	2	1	0
保留			TXTHIF	保留	RXOVIF	保留	RXTHIF

位	描述	
[31:28]	**TXCNT**	**发送 FIFO 数据计数 (只读)** 该位域表明发送FIFO缓存的有效数据计数。
[27]	**TXFULL**	**发送 FIFO 缓存满标志 (只读)** 0 = 发送 FIFO 缓存没满 1 = 发送 FIFO 缓存满 **注:** 该位是SPI_CTL[27]的相互镜像位
[26]	**TXEMPTY**	**发送 FIFO 缓存空标志 (只读)** 0 = 发送 FIFO 缓存非空 1 = 发送 FIFO 缓存空 **注:** 该位是SPI_CTL[26]的相互镜像位
[25]	**RXFULL**	**接收 FIFO 缓存满标志 (只读)** 0 = 接收 FIFO 缓存没满 1 = 接收 FIFO 缓存满 **注:** 该位是SPI_CTL[25]的相互镜像位
[24]	**RXEMPTY**	**接收 FIFO 缓存空标志 (只读)** 0 = 接收 FIFO 缓存没空 1 = 接收 FIFO 缓存空 **注:** 该位是SPI_CTL[24]的相互镜像位
[23:21]	保留	保留.
[20]	**SLVTOIF**	**超时中断标志** 0 = 没有收到 FIFO 超时事件 1 = 接收 FIFO 缓存非空且主机模式下超过64个SPI时钟周期或从机模式下超过576个SPI外设时钟周期没有读操作。当接收到的FIFO缓存被软件读取，超时状态会自动清零。 . **注:** 向该位写1清零。

[19:17]	保留	保留.
[16]	**UNITIF**	**SPI 单位传输中断标志** 0 = 传输未完成 1 = SPI 控制器已经完成一个单元传输. **注 1:** 向该位写1清零。 **注 2:** 该位是SPI_CTL[16]的相互镜像位
[15:12]	**RXCNT**	**接收 FIFO 数据计数 (只读)** 该域表明接收 FIFO 缓存有效数据计数.
[11]	**SLVSTIF**	**从机开始中断状态(仅用于从机)** It is used to dedicate that the transfer has started in slave 3-wire mode. 该位用来表明在3线模式下，是否已经开始了一次传输。 0 = 自动 SLVSTIEN 置1，从机没有侦测到任何SPI时钟传输。 1 = 在3线模式下，已经开始一次传输。 **注 1:** 当一次传输结束，或者写1到该位会清除该位。 **注 2**: 该位是SPI_SLVCTL[11]的相互镜像位
[10:5]	保留	保留.
[4]	**TXTHIF**	**发送 FIFO 阀值中断状态(只读)** 0 = 发送FIFO缓存的有效数据计数大于 TXTH 的设定值。 1 = 发送FIFO缓存的有效数据计数少于或等于 TXTH 的设定值。 **注:** 如果 TXTHIEN = 1 和 TXTHIF = 1，SPI 控制器会产生一个SPI中断请求。
[3]	保留	保留.
[2]	**RXOVIF**	**接收 FIFO Overrun 状态** 当接收 FIFO缓存满，上面的数据会丢掉，该位会置1。 0 = 接收 FIFO中没有 overrun 1 = 接收 FIFO中有 overrun **注:** 向该位写1清零。
[1]	保留	保留.
[0]	**RXTHIF**	**接收 FIFO 阀值中断状态(只读)** 0 = 接收FIFO缓存的有效数据计数小于或等于 RXTH 的设定值。 1 = 接收FIFO缓存的有效数据计数大于 RXTH 的设定值。 **注:** 如果RXTHIEN = 1 和 RXTHIF = 1， SPI 控制器会产生一个SPI中断请求。

6.13 模数转换(ADC)

6.13.1 概述

Mini58系列包含一个10位8通道逐次逼近型模数转换器（SAR A/D）。A/D转换器可由软件，外部引脚(STADC/P3.2)或PWM触发.

6.13.2 特性

- 模拟输入电压范围：0 ~ AV_{DD}
- 10位分辨率8位有效保证
- 多达8路单端模拟输入通道
- ADC外设频率最高可达6MHz，每次采样需14个ADC时钟
- 两种操作模式
  - 单次模式：对指定通道执行一次A/D转换
  - PWM 连续模式：当PWM触发，从0到2这三个通道中的两个通道将自动进行转换，转换顺序按照MODESEL (ADC_SEQCTL[3:2])中设定进行，或是通道[0,1]或是通道[1,2] 亦或是通道[0,2]。
- A/D转换触发条件：:
  - 软件写1到SWTRG位
  - 外部管脚(STADC)
  - PWM触发，可选择延时周期
- 每个通道的转换结果都存储在对应的数据寄存器中，并且带有有效/超载标志
- 转换结果可与指定的值进行比较。当转换值和指定比较寄存器中的设定值相同时，用户还可以选择是否产生一个中断请求
- 通道7支持2 路输入源：外部模拟电压和内部带隙电压

6.13.3 框图

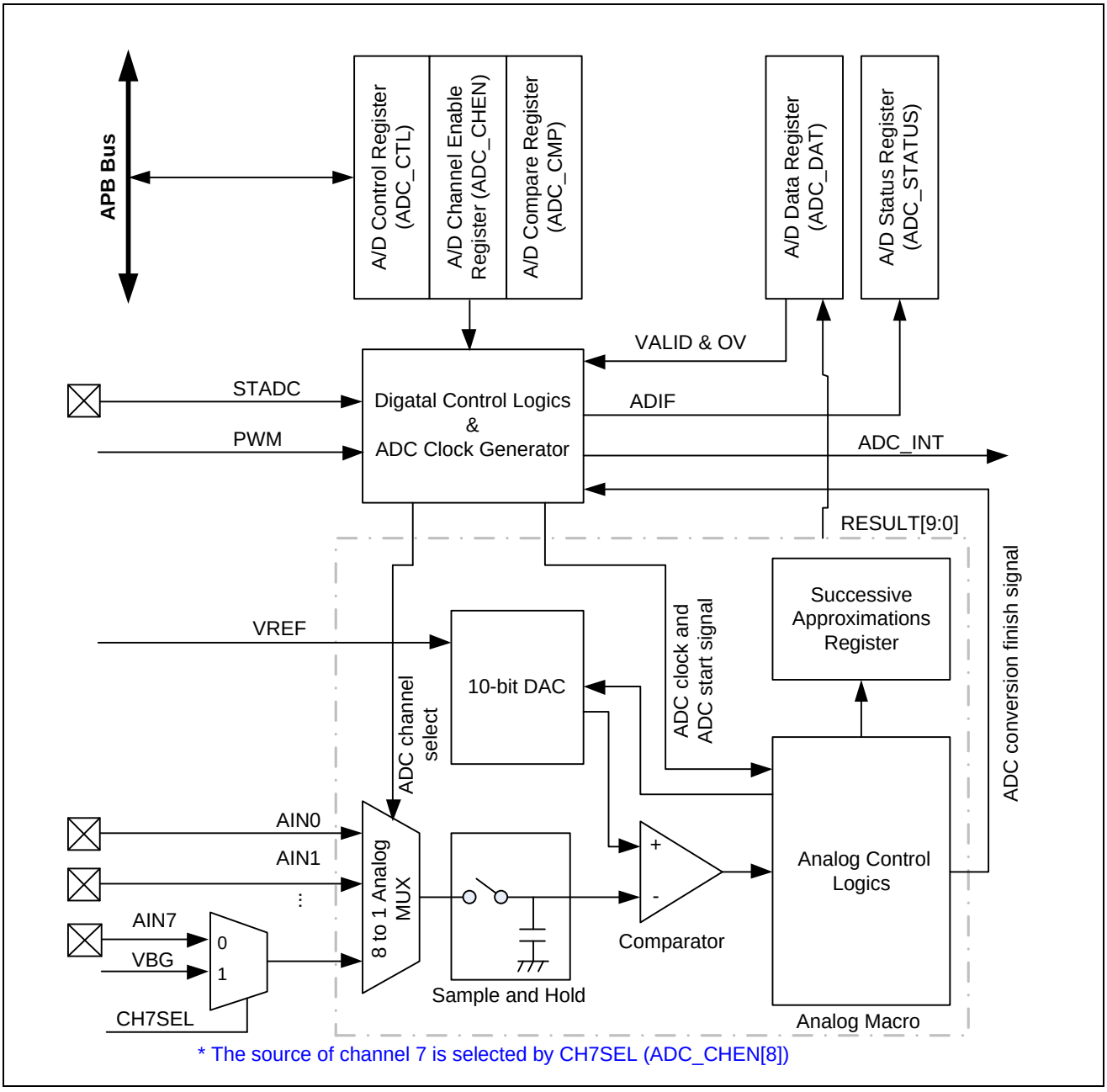


图 6.13-1 AD控制器框图

6.13.4 基本配置

ADC引脚功能是在SYS_P1_MFP 和 SYS_P3_MFP寄存器中配置的。建议将模拟信号输入管脚的数字输入通道关闭以防止漏电流，用户可以通过设置P1_DINOFF 和P3_DINOFF寄存器关闭数字输入通道。

ADC 模块控制器的时钟源是由ADCCKEN (CLK_APBCLK[28])控制使能的。ADC外设时钟源通过ADCSEL (CLK_CLKSEL1[3:2])选择。时钟分频器由ADCDIN (CLK_CLKDIV[23:16])决定

6.13.5 功能描述

A/D转换器采用逐次逼近转换方式，转换结果为10位数据。当需要改变模拟输入通道时，为防止错误操作，软件必须先将ADC_CTL 寄存器中的SWTRG位清为0。一旦SWTRG 位清零，A/D转换器将立刻抛弃当前的转换结果进入空闲状态。

6.13.5.1 ADC时钟发生器

ADC共有4个时钟源，可通过ADCSEL (CLK_CLKSEL1[3:2])进行选择。选择HXT 或 LXT时需使能XTLEN (CLK_PWRCTL[1:0])，使能外部时钟。ADC时钟频率按如下公式进行8位预分频：

ADC时钟频率 = (ADC 时钟源频率) / (ADCDIV +1);

8位的ADCDIV位于寄存器CLK_CLKDIV[23:16].

总之，软件可通过设置ADCSEL 和ADCDIV得到6 MHz或与之接近的频率。

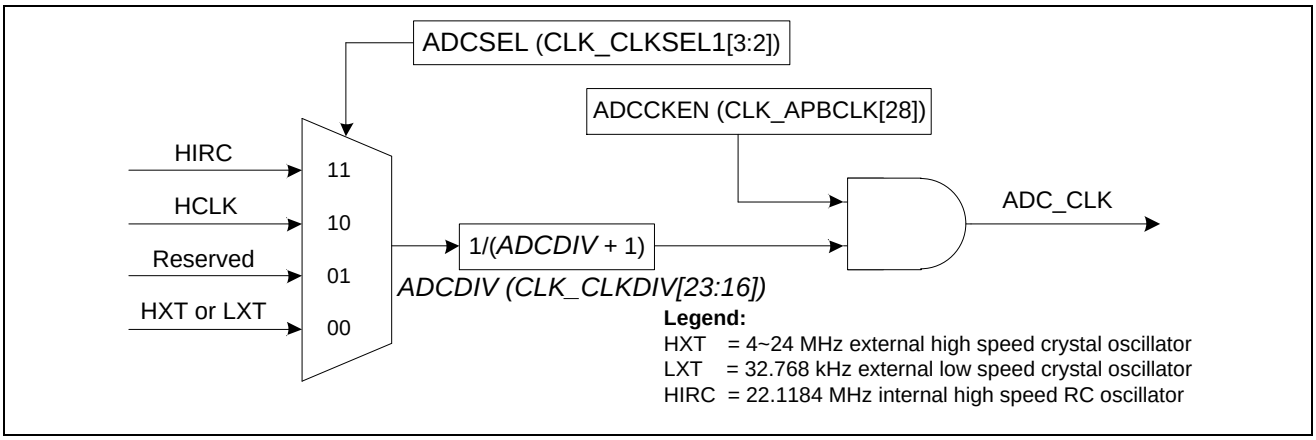


图 6.13-2 ADC时钟控制

6.13.5.2 ADC 单次操作

A/D转换器仅对指定的一个通道进行一次转换。操作流程为:

1. 当寄存器ADC_CTL中的SWTRG位被软件或外部触发置1时，A/D转换开始。
2. 当A/D转换完成，转换结果将存到A/D数据寄存器中
3. A/D转换完成后， ADC_STATUS中的ADIF位会被置1，如果此时寄存器ADC_CTL 中的ADCIEN =1, 则产生ADC中断。
4. 在A/D转换期间，SWTRG位一直保持为1。当A/D转换结束，SWTRG位会自动清0， A/D转换器进入空闲状态。
5. 图 6.13-3为单次模式的采样时序框图。

注：如果软件使能了多个通道，编号最小的通道将被选中而其他使能通道将被忽略。

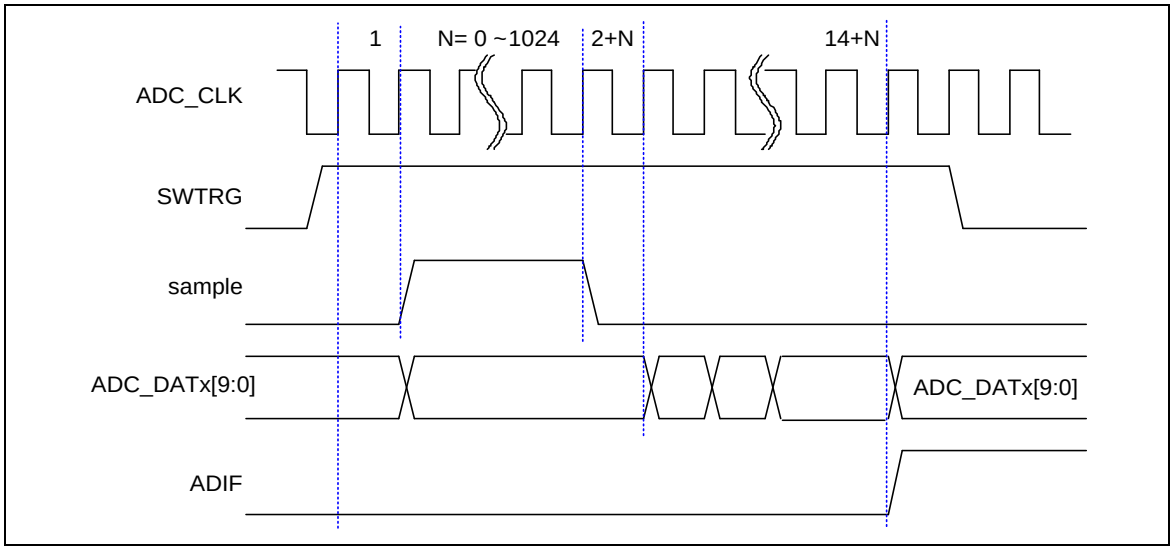


图 6.13-3 单一模式转换时序图

6.13.5.3 外部触发输入采样和A/D转换时间

A/D转换器可由外部管脚触发转换。当设置HWTRGEN (ADC_CTL[8])为1用来使能ADC外部触发功能时，设置HWTRGSEL (ADC_CTL[5:4]) 为 00b选择STADC管脚为外部触发管脚。再设定HWTRGCOND 来设置触发条件是上升沿还是下降沿触发。一个8位计数器用于去毛刺。如果选定边沿触发，高或低的状态至少需要保持4个PCLK，低于该值的脉冲将被忽略。

6.13.5.4 PWM 触发

A/D转换器还可由PWM触发转换。当设置HWTRGEN (ADC_CTL[8])为1用来使能ADC外部触发功能时，设置HWTRGSEL (ADC_CTL[5:4]) 为11b选择PWM为外部触发源。当使能PWM触发ADC功能时，设置DELAY (ADC_TRGDLY[7:0])可以在PWM触发条件和ADC启动转换之间加入一个延时。

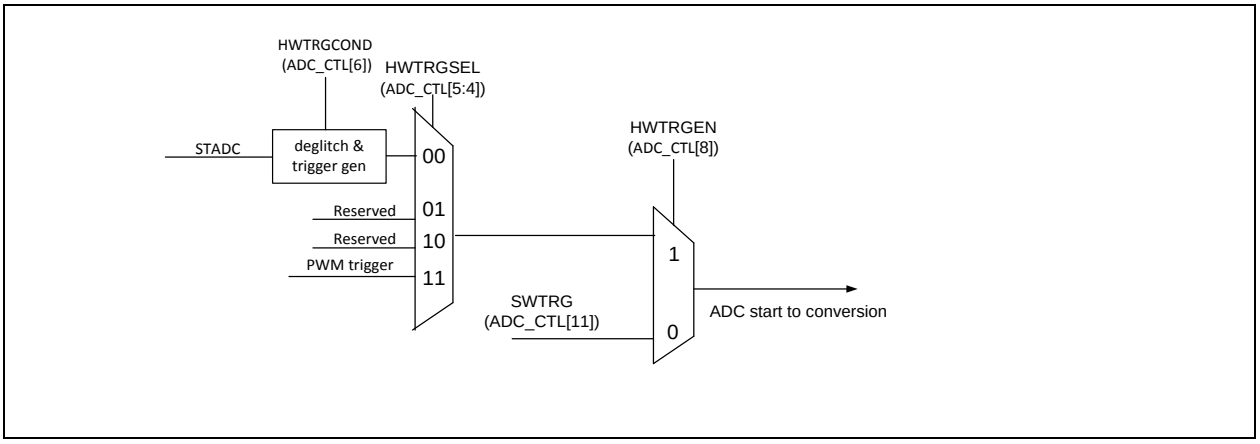


图 6.13-4 ADC启动转换条件

6.13.5.5 通过比较功能监测转换结果

Mini58系列的ADC控制器提供两组比较寄存器ADC_CMP0 和 ADC_CMP1，用于监控A/D通道（最多支持2路）的转换结果值。软件可通过设定CMPCH (ADC_CMPx[5:0])来选择监控哪路通道，而CMPCOND位则用于设置比较条件，如果CMPCOND为0，当转换结果小于CMPDAT (ADC_CMPx[25:16])设定值时，内部匹配计数器将会加1；如果CMPCOND为1，当转换结果大于等于CMPDAT[9:0]设定值时，内部匹配计数器将会加1。当CMPCH指定的通道转换完成时，比较行为会被自动触发一次。当比较结果和设定值相匹配，比较匹配计数器将加1，否则匹配计数器将会被清0。当计数器的值和设定值(CMPMCNT+1)匹配时，ADCMPF 位将会置1。如果ADCMPIE为1，将产生ADC_INT中断请求。使用这个功能就可在扫描模式下，无需软件介入就可以监控外部模拟输入管脚电压变化。图 6.13-5列出了监控逻辑框图。

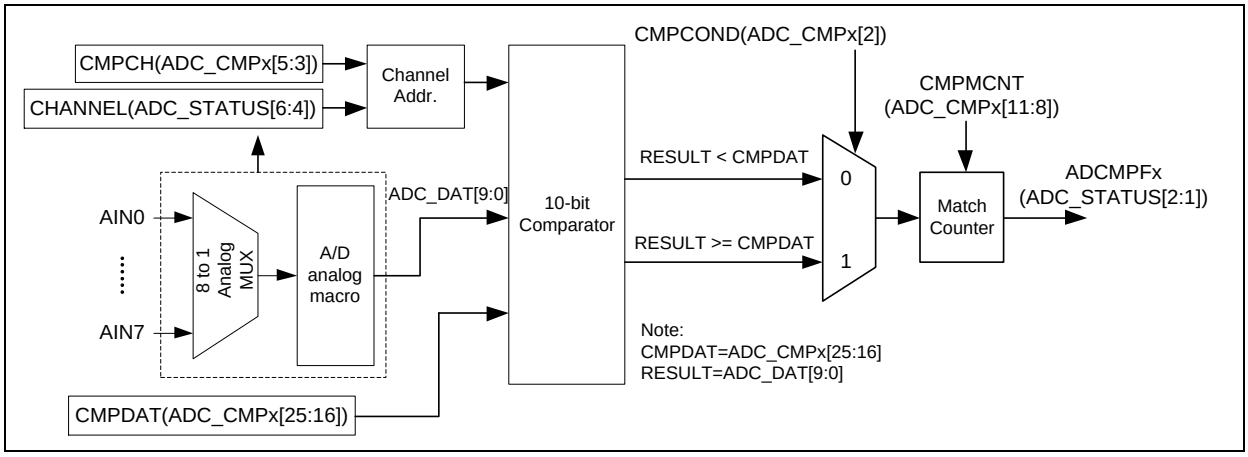


图 6.13-5 A/D转换结果监控逻辑框图

6.13.5.6 中断源

ADC中断共有三个中断源。当某个ADC操作模式结束转换时，A/D转换的结束标志ADF位将被置1。ADCMPF0 和 ADCMPF1是比较功能的比较标志。当A/D转换结果同ADC_CMP0/1寄存器设定值匹配时，相应的比较标志会被置1。当ADIF, ADCMPF0 和 ADCMPF1中任意一个标志被置1，且其相应中断使能位即ADC_CTL中的ADCIEN 及ADCMPIE中的ADC_CMP0/1置1时，将产生ADC中断。软件可通过清除标志位以撤销该中断请求。

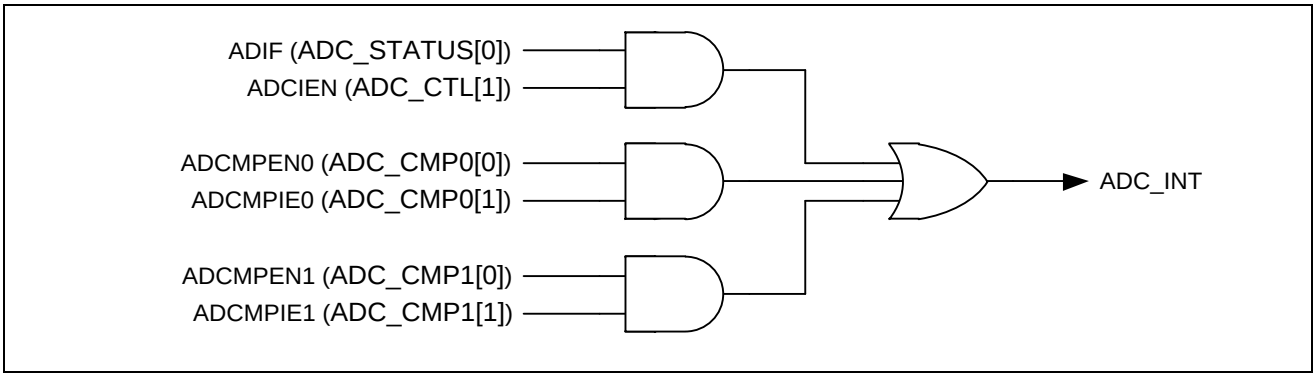


图 6.13-6 A/D控制器中断

6.13.5.7 转换结果

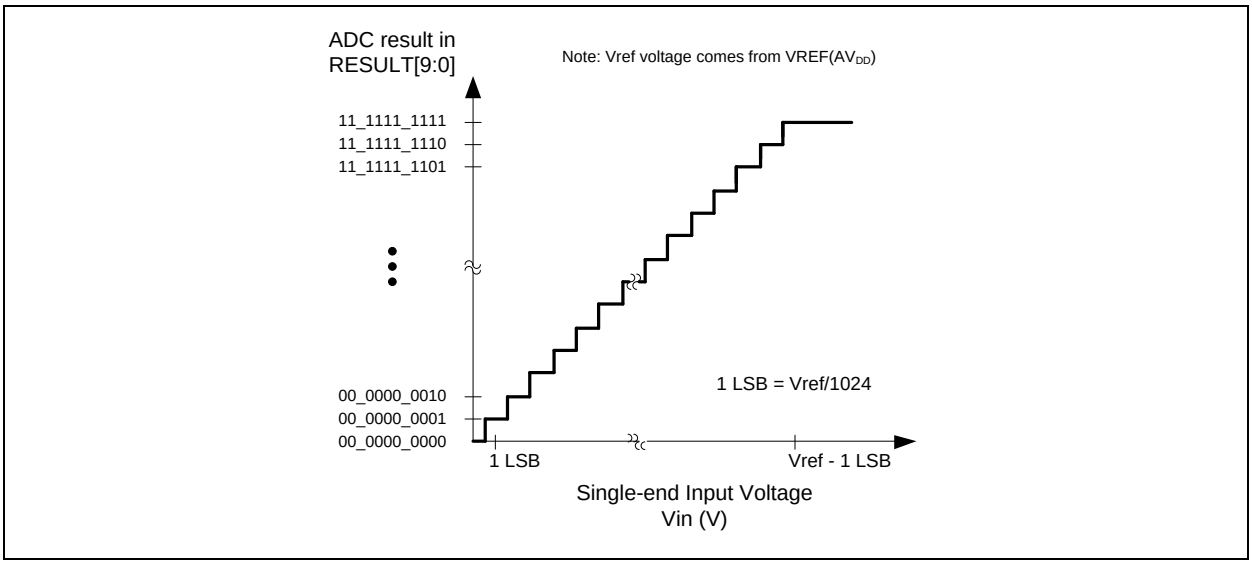


图 6.13-7 单端输入转换结果映射图

6.13.5.8 PWM 连续模式

支持2个通道的连续模式，这样中断频率可减低一半。当SEQEN (ADC_SEQCTL[0])置为1使能A/D PWM连续功能 时，设置TRG1CTL (ADC_SEQCTL[11:8]) 和 TRG2CTL (ADC_SEQCTL[19:16])用于选择外部触发输入来自PWM 通道 0/2/4，触发方式可以为上升沿/中心点/下降沿/周期点。当ADC连续模式使能时，从0到2 三个通道中的两个，按照MODESEL (ADC_SEQCTL[3:2])的设定或为通道 [0, 1] 或通道 [1,2] 或通道[0,2]， 将会自动进行模拟数据转换。而且，如果SEQTYPE (ADC_SEQCTL[1])被设置为低，ADC将只在第一次转换之前插入延迟时间，第二次转换则在第一次转换结束后立刻进行，这种方式应用于 2/3-shunt 类型，如果SEQTYPE (ADC_SEQCTL[1])被置为高，ADC将在每次转换开始前插入延迟时间，这种方式应用于 1-shunt 类型。2/3-shunt类型时，有效ADC通道为CH0~2。1-shunt类型时 ，有效通道为CH0~7。图 6.13-8 和 图 6.13-9为该功能框图。

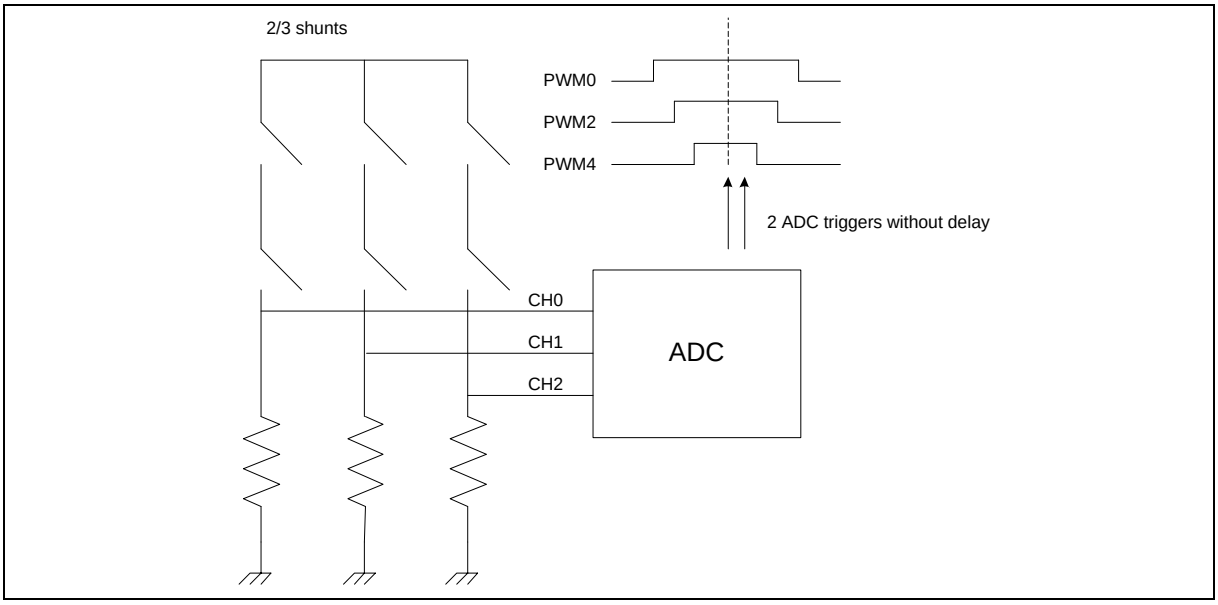


图 6.13-8 ADC 连续模式用于 2/3-shunt

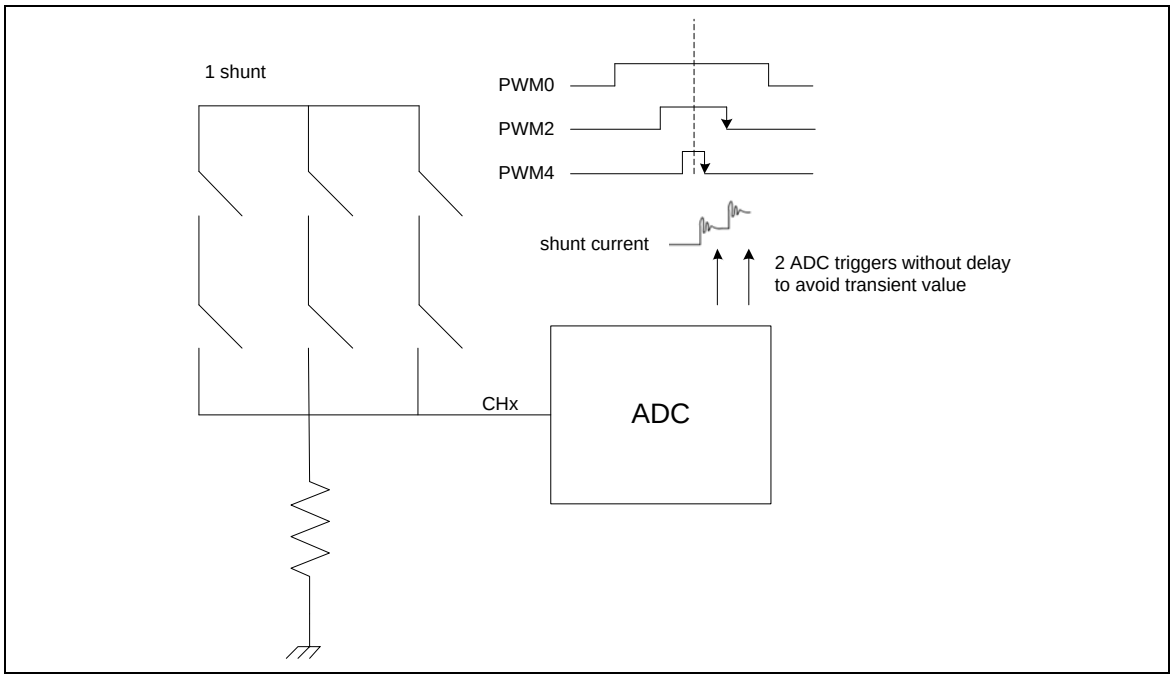


图 6.13-9 ADC连续模式用于1-shunt

6.13.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读写

寄存器	偏移量	R/W	描述	复位值
ADC基地址: **ADC_BA = 0x400E_0000**				
ADC_DAT	ADC_BA+0x00	R	A/D数据寄存器	0x0000_0000
ADC_CTL	ADC_BA+0x20	R/W	A/D控制寄存器	0x0000_0000
ADC_CHEN	ADC_BA+0x24	R/W	A/D 通道使能寄存器	0x0000_0000
ADC_CMP0	ADC_BA+0x28	R/W	A/D比较寄存器0	0x0000_0000
ADC_CMP1	ADC_BA+0x2C	R/W	A/D比较寄存器1	0x0000_0000
ADC_STATUS	ADC_BA+0x30	R/W	A/D状态寄存器	0x0000_0000
ADC_TRGDLY	ADC_BA+0x44	R/W	A/D触发延时控制寄存器	0x0000_0000
ADC_EXTSMPT	ADC_BA+0x48	R/W	A/D 采样时间计数器寄存器	0x0000_0000
ADC_SEQCTL	ADC_BA+0x4C	R/W	A/D PWM 连续模式控制寄存器	0x0000_0000
ADC_SEQDAT1	ADC_BA+0x50	R	A/D PWM 连续模式第一个结果寄存器	0x0000_0000
ADC_SEQDAT2	ADC_BA+0x54	R	A/D PWM连续模式第二个结果寄存器	0x0000_0000

6.13.7 寄存器描述

ADC数据寄存器(ADC_DAT)

寄存器	偏移量	R/W	描述	复位值
ADC_DAT	ADC_BA+0x00	R	A/D 数据寄存器	0x0000_0000

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

位	描述	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**有效标志** 当ADC转换结束时该位被置为1. 读ADC_DAT寄存器后,该位由硬件自动清零. 0 = RESULT[9:0]中的数据无效 1 = RESULT[9:0]中的数据有效
[16]	**OV**	**超载标志** 如果在新的转换结果载入到RESULT[9:0]寄存器之前,上一次的转换结果还没有被读取，则OV 标志将被置1。读ADC_DAT寄存器后，该位由硬件自动清零。 0 = RESULT[9:0]中的数据是最新的转换结果 1 = RESULT[9:0]中的数据被覆盖过
[15:10]	**Reserved**	Reserved.
[9:0]	**RESULT**	**A/D 转换结果** 该域存放ADC的转换结果

ADC控制寄存器(ADC_CTL)

寄存器	偏移量	R/W	描述	复位值
ADC_CTL	ADC_BA+0x20	R/W	A/D控制寄存器	0x0000_0000

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

位	描述	
[31:12]	**Reserved**	Reserved.
[11]	**SWTRG**	**软件触发启动A/D转换** SWTRG位被置1可通两种方式:软件和外部STADC管脚. 当转换结束时, SWTRG将由硬件自动清零. 0 = 转换停止, A/D转换器进入空闲状态 1 = 转换开始
[10:9]	**Reserved**	Reserved.
[8]	**HWTRGEN**	**硬件外部触发使能位**: 使能或是禁止外部STADC管脚触发A/D转换.如果使能外部触发,SWTRG位就可以通过选中的硬件触发源置为1. 0 = 外部触发禁止 1 = 外部触发使能
[7]	**Reserved**	Reserved.
[6]	**HWTRGCOND**	**硬件外部触发条件** 该位决定外部管脚STADC触发事件条件为下降沿还是上升沿. 边沿触发时, 触发信号必须保持稳定, 即高或低状态至少保持4个PCLK 0 = 下降沿 1 = 上升沿
[5:4]	**HWTRGSEL**	**硬件触发源选择位**: 00 = A/D转换器由外部STADC管脚触发 11 = A/D转换器由PWM触发 其它 = 保留 **注**:软件在改变TRGS之前,应该先禁止TRGEN和SWTRG.
[3:2]	**Reserved**	Reserved.

位	描述	
[1]	**ADCIEN**	**A/D中断使能位** 如果ADCIEN被置1，则A/D转换结束后将产生中断请求 0 = 禁止A/D中断 1 = 使能A/D中断
[0]	**ADCEN**	**A/D转换器使能位** 0 = 禁止 1 = 使能 **注**: 在开始A/D转换前，该位需设置为1。清除该位为0将禁用A/D转换器模拟电路从而节省功耗

ADC通道使能寄存器(ADC_CHEN)

寄存器	偏移量	R/W	描述	复位值
ADC_CHEN	ADC_BA+0x24	R/W	A/D通道使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**CH7SEL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHEN7	**CHEN6**	**CHEN5**	**CHEN4**	**CHEN3**	**CHEN2**	**CHEN1**	**CHEN0**

位	描述	
[31:9]	**Reserved**	Reserved.
[8]	**CH7SEL**	**模拟输入通道7选择:** 0 = 外部模拟输入 1 = 内部带隙电压(VBG) **注**:当软件选择带隙电压作为ADC通道7的模拟输入源,那么ADC时钟需要限制在300kHz以下.
[7]	**CHEN7**	**模拟输入通道7使能位** 0 = 通道7禁止 1 = 通道7使能
[6]	**CHEN6**	**模拟输入通道6使能位** 0 = 通道6禁止 1 = 通道6使能
[5]	**CHEN5**	**模拟输入通道5使能位** 0 = 通道5禁止 1 = 通道5使能
[4]	**CHEN4**	**模拟输入通道4使能位** 0 = 通道4禁止 1 = 通道4使能
[3]	**CHEN3**	**模拟输入通道3使能位** 0 = 通道3禁止 1 = 通道3使能
[2]	**CHEN2**	**模拟输入通道2使能位** 0 = 通道2禁止 1 = 通道2使能

位	描述	
[1]	**CHEN1**	**模拟输入通道1使能位** 0 = 通道1禁止 1 = 通道1使能
[0]	**CHEN0**	**模拟输入通道0使能位** 0 = 通道0禁止 1 = 通道0使能 **注:** 如果软件使能了多个通道, 那么只有通道号最小的通道会被选中,其它通道则被忽略.

A/D比较寄存器 0/1 (ADC_CMP0/1)

寄存器	偏移量	R/W	描述	复位值
ADC_CMP0	ADC_BA+0x28	R/W	A/D比较寄存器0	0x0000_0000
ADC_CMP1	ADC_BA+0x2C	R/W	A/D比较寄存器1	0x0000_0000

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

位	描述	
[31:26]	**Reserved**	Reserved.
[25:16]	**CMPDAT**	**比较数值** 此10位数据用于和指定通道的转换结果进行比较。
[15:12]	**Reserved**	Reserved.
[11:8]	**CMPMCNT**	**比较匹配计数** 当指定A/D通道的模拟转换值和比较条件CMPCOND[2]相匹配时，内部计数器将加1，在比较过程中，当内部计数器达到设定值(CMPMCNT+1)时，将置ADCMPFx为1。
[7:6]	**Reserved**	Reserved.
[5:3]	**CMPCH**	**比较通道选择** 设置该域用于选择对哪个通道结果进行比较 **注：**该域有效设定为0-7
[2]	**CMPCOND**	**比较条件** 0 = 设置比较条件为：当10位A/D转换结果小于10位CMPDAT (ADC_CMPx[25:16])，内部匹配计数器将加1 1 = 设置比较条件为：当10位A/D转换结果大于或等于 10位CMPDAT (ADC_CMPx[25:16])，内部匹配计数器将加1 **注意：**当内部计数器达到(CMPMCNT+1)，ADCMPFx将会被置1
[1]	**ADCMPIE**	**A/D比较中断使能位** 如果使能了比较功能，而且比较条件满足CMPCOND 和CMPMCNT中的设定，则ADCMPIE将会被置1，同时，如果ADCMPIE为1，将产生比较中断请求 0 = 禁止比较中断 1 = 使能比较中断

位	描述	
[0]	**ADCMPEN**	**A/D比较使能位** 设置此位为1可使能ADC控制器 将指定通道转换结果载入到ADC_DAT寄存器后与CMPDAT (ADC_CMPx[25:16])中数据进行比较 0 = 禁止比较功能 1 = 使能比较功能

A/D状态寄存器(ADC_STATUS)

寄存器	偏移量	R/W	描述	复位值
ADC_STATUS	ADC_BA+0x30	R/W	A/D状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**OV**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**VALID**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CHANNEL**			**BUSY**	**ADCMPF1**	**ADCMPF0**	**ADIF**

位	描述	
[31:17]	**Reserved**	Reserved.
[16]	**OV**	**超载标志（只读）** 该位为ADC_DAT 寄存器中OV 位的镜像
[15:9]	**Reserved**	Reserved.
[8]	**VALID**	**数据有效标志(只读)** 该位为ADC_DAT 寄存器中VALID 位的镜像
[7]	**Reserved**	Reserved.
[6:4]	**CHANNEL**	**当前转换通道（只读）** 当BUSY=1时，该域反应当前转换通道号。当BUSY=0时，它表示下一个要转换的通道号
[3]	**BUSY**	**忙/空闲(只读）** 该位为寄存器ADC_CTL中SWTRG位的镜像. 0 = A/D转换器处于空闲状态 1 = A/D转换器处于忙状态
[2]	**ADCMPF1**	**A/D比较标志1** 当所选择的通道的A/D转换结果和ADC_CMP1的设定条件匹配时，该位置1。 0 = ADC_DAT中的转换结果和ADC_CMP1 的设定值不匹配 1 = ADC_DAT中的转换结果和ADC_CMP1的设定值匹配 **注：**该位写1清除
[1]	**ADCMPF0**	**A/D 比较标志0** 当所选择的通道的A/D转换结果和ADC_CMP0的设定条件匹配时，该位置1。 0 = ADC_DAT中的转换结果和ADC_CMP0 的设定值不匹配 1 = ADC_DAT中的转换结果和ADC_CMP0的设定值匹配

位	描述	
		注：该位写1清除
[0]	**ADIF**	**A/D转换结束标志** 该状态标志表明A/D转换结束。当A/D转换结束时ADIF被置1。 **注**：软件写1清0

A/D 触发延时控制寄存器(ADC_TRGDLY)

寄存器	偏移量	R/W	描述	复位值
ADC_TRGDLY	ADC_BA+0x44	R/W	A/D 触发延时控制寄存器	0x0000_0000

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

位	描述	
[31:8]	**Reserved**	Reserved.
[7:0]	**DELAY**	**PWM触发延时定时器** 设置该域用于PWM触发后延时ADC采样开始时间，PWM触发延时为：(4 * DELAY) *系统时钟

A/D 采样寄存器(ADC_EXTSMPT)

寄存器	偏移量	R/W	描述	复位值
ADC_EXTSMPT	ADC_BA+0x48	R/W	A/D 采样时间计数器寄存器	0x0000_0000

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

位	描述	
[31:4]	**Reserved**	Reserved.
[3:0]	**EXTSMPT**	**附加ADC采样时钟** 如果ADC输入不稳定，用户可以设置该寄存器来增加采样时间以得到稳定的ADC输入信号。默认采样时间为1ADC 时钟。可插入附加时钟以拉长采样时钟 。 0 = 附加时钟周期为 0。 1 = 附加时钟周期为 1。 2 = 附加时钟周期为 2。 3 = 附加时钟周期为 4。 4 = 附加时钟周期为 8。 5 = 附加时钟周期为 16。 6 = 附加时钟周期为 32。 7 = 附加时钟周期为 64。 8 = 附加时钟周期为 128。 9 = 附加时钟周期为 256。 10 = 附加时钟周期为 512。 11 = 附加时钟周期为 1024。 12 = 附加时钟周期为 1024。 13 = 附加时钟周期为 1024。 14 = 附加时钟周期为 1024。 15 = 附加时钟周期为 1024。

A/D PWM 连续寄存器(ADC_SEQCTL)

寄存器	偏移量	R/W	描述	复位值
ADC_SEQCTL	ADC_BA+0x4C	R/W	A/D PWM 连续模式控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**TRG2CTL**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**TRG1CTL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**MODESEL**		**SEQTYPE**	**SEQEN**

位	描述	
[31:20]	**Reserved**	Reserved.
[19:16]	**TRG2CTL**	**TRG2CTL[3:2]**：PWM 触发源选择 00 = PWM触发源为PWM0 01 = PWM触发源为PWM2 10 = PWM触发源为PWM4 11 = 保留 **TRG2CTL[1:0]**：PWM 触发方式选择 00 = PWM的上升沿 01 = PWM的中心点 10 = PWM的下降沿 11 = PWM的周期点 **注**：PWM触发源对于1-shunt类型有效
[15:12]	**Reserved**	Reserved.
[11:8]	**TRG1CTL**	**TRG1CTL[3:2]**：PWM 触发源选择 00 = PWM触发源为PWM0 01 = PWM触发源为PWM2 10 = PWM触发源为PWM4 11 = 保留 **TRG1CTL[1:0]**：PWM 触发方式选择 00 = PWM的上升沿 01 = PWM的中心点 10 = PWM的下降沿 11 = PWM的周期点 **注**：PWM触发源对于1-shunt和2/3-shunt类型有效

位	描述	
[7:4]	**Reserved**	Reserved.
[3:2]	**MODESEL**	ADC 连续模式选择 00 = 当SEQEN =1时，先通道0 然后通道1转换结束后执行ADC_INT。 01 = 当SEQEN =1时，先通道1 然后通道2转换结束后执行ADC_INT。 10 = 当SEQEN =1时，先通道0 然后通道2转换结束后执行ADC_INT。 11 =保留
[1]	**SEQTYPE**	**ADC连续模式方式** 0 = ADC 延迟事件仅插入在首个转换之前。第二个转换将在首个转换结束后立刻进行（针对2/3 –shunt 类型） 1 = ADC延迟事件插入在每次转换之前。（针对1-shunt类型）
[0]	**SEQEN**	**ADC连续模式使能位** 当使能ADC连续模式时，通道0到2这三个通道中的两个通道将自动转换模拟数据。按照MODESEL (ADC_SEQCTL[3:2])的设置，或是通道 [0，1]或是通道[1，2]或是通道[0，2]。 0 = ADC连续模式禁止 1 = ADC连续模式使能

A/D PWM 连续模式结果寄存器(ADC_SEQDAT1/2)

寄存器	偏移量	R/W	描述	复位值
ADC_SEQDAT1	ADC_BA+0x50	R	A/D PWM 连续模式第一个结果寄存器	0x0000_0000
ADC_SEQDAT2	ADC_BA+0x54	R	A/D PWM 连续模式第二个结果寄存器	0x0000_0000

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

位	描述	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**有效标志** 当ADC转换结束时该位置1， 读寄存器ADC_DAT后，将由硬件清零。 0 = 在RESULT[9:0]中的数据无效 1 = 在RESULT[9:0]中的数据有效
[16]	**OV**	**超载标志** 如果RESULT[9:0]中的转换数据在新的转换结果载入之前还没有被读走，OV 将会被置1。读ADC_DAT寄存器后，硬件会将此位清零。 0 = 在RESULT[9:0]中的数据为最新转换结果 1 = 在RESULT[9:0]中的数据被覆盖过
[15:10]	**Reserved**	Reserved.
[9:0]	**RESULT**	**A/D PWM 连续模式转换结果** 该域包含有ADC的转换结果

6.14 模拟比较器(ACMP)

6.14.1 概述

NuMicro® Mini58系列包含两个比较器，可按照比较器配置应用于不同的场合。当正端输入大于负端输入时，比较器输出逻辑1，否则输出0。当比较器输出值改变时，每路比较器都可通过配置产生中断。

6.14.2 特性

- 模拟输入电压范围：0 ~ AV_{DD}
- 支持迟滞功能
- 每个比较器负极输入有多个内部参考电压源可选
- ACMP0比较器支持功能
  - 4路正极输入源
    - P1.5, P1.0, P1.2, 或P1.3
  - 3路负极输入源
    - P1.4
    - 比较器内部参考电压 (CRV)
    - 内部带隙电压(V_{BG})
- ACMP1比较器支持功能
  - 4 路正极输入源
    - P3.1, P3.2, P3.4, 或P3.5
  - 3 路负极输入源
    - P3.0
    - 比较器内部参考电压 (CRV)
    - 内部带隙电压(V_{BG})

6.14.3 框图

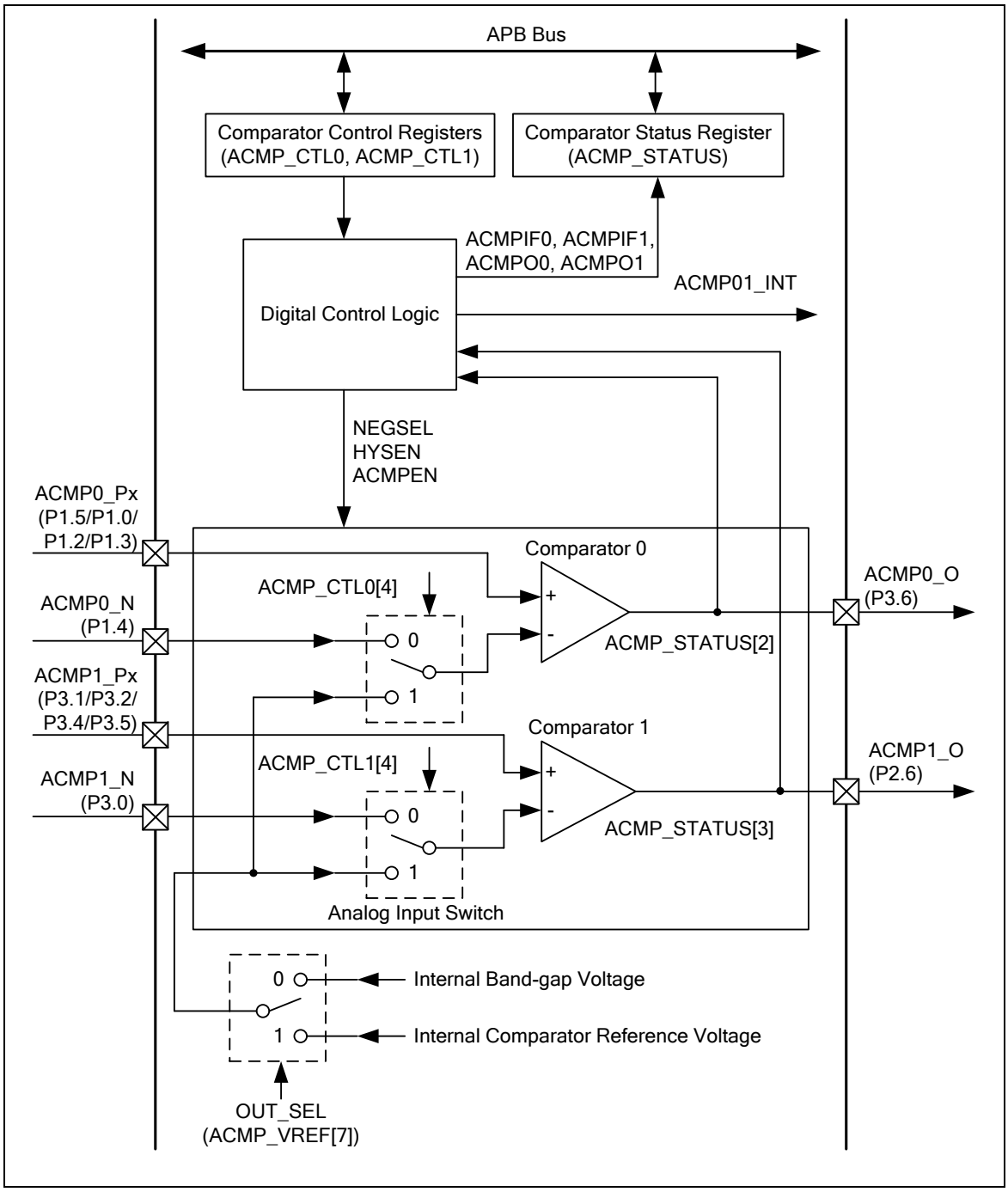


图 6.14-1 模拟比较器框图

6.14.4 基本配置

模拟比较器的功能引脚由寄存器SYS_P1_MFP, SYS_P2_MFP 和SYS_P3_MFP中的配置决定。如果被定

义为比较器模拟输入引脚，建议将该引脚的数字输入通道关闭，以避免产生漏电流。引脚数字输入通道可通过配置寄存器P1_OFFD 和P3_OFFD关闭。如果GPIO管脚配置为ACMP输入脚后，在Px_PMD寄存器中切忌将管脚配置为推挽输出模式。 输入模式是最安全的配置。如果为开漏，输出或准双向模式，该GPIO不可输出 0。GPIO默认输出值应为1。Px_PMD的默认设置是由用户配置的，应该配置它为输入模式或准双向模式。

置寄存器ACMPCKEN (CLK_APBCLK[30])为1，用于使能模拟比较器ACMP的模块时钟。

6.14.5 功能描述

6.14.5.1 中断源

比较器输出在通过 PCLK采样后，结果会被写入到 寄存器ACMPOx (ACMP_STATUS[3] 和 ACMP_STATUS[2])中。如果ACMPIE (ACMP_CRx[1])设置为1，则模拟比较器的中断将被使能。 当比较器的输出状态改变时，将会产生比较器中断请求，相应的比较器中断标志ACMPFx (ACMP_STATUS[1] 和 ACMP_STATUS[0])将被置位。软件可写1到标志位清除该位

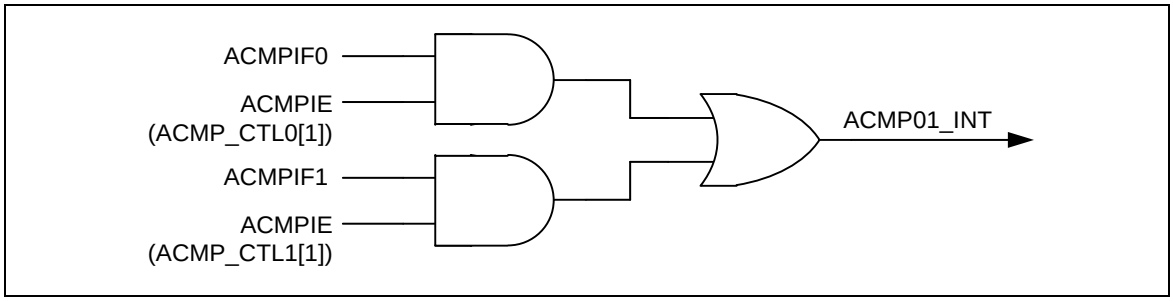


图 6.14-2 比较器控制器中断源

6.14.5.2 迟滞功能

模拟比较器提供迟滞功能以使比较器输出变化更平稳。当比较器输出为0时，比较器将会一直输出为0直到正极输入电压与负极输入电压的电压差大于迟滞电压为止。同样的，当比较器输出为1时，比较器将会一直输出为1直到正极输入电压值与负极输入电压的电压差小于迟滞电压为止。

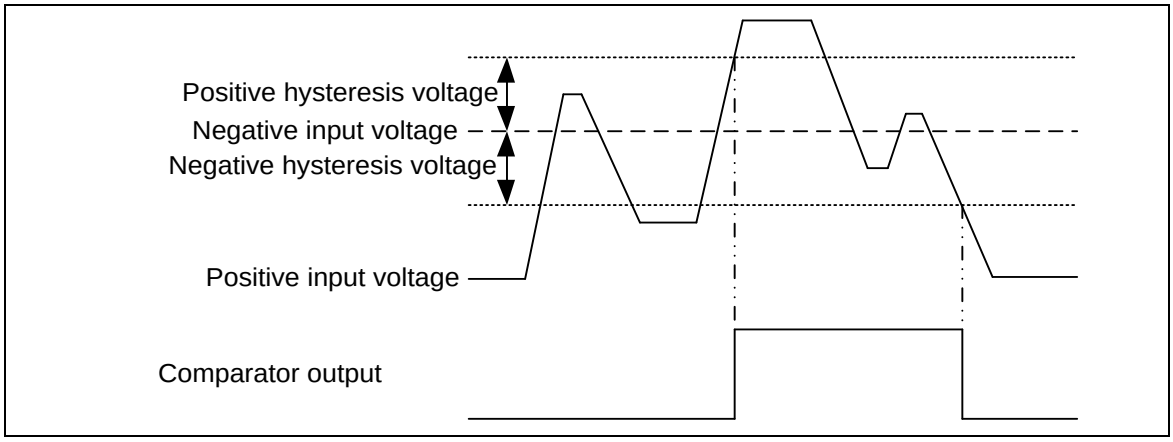


图 6.14-3 比较器的迟滞功能

6.14.5.3 滤波功能

模拟比较器提供滤波功能，用于避免比较器输出不稳定的情况。通过设定FILTSEL (ACMP_CTL0[23:20], ACMP_CTL1[23:20])，比较器输出将被连续的PCLK采样。 采样时间越长，比较器输出越平稳。但是比较器输出灵敏度将会降低。 滤波功能在去噪输出时非常有用。通过设定FILTSEL (ACMP_CTL0[23:20], ACMP_CTL1[23:20])，用户可使滤波器数量刚刚大于噪声时间。 例如，如果噪声时长小于8个PCLK周期，用户可配置FILTSEL (ACMP_CTL0[23:20], ACMP_CTL1[23:20])为0100，这样ACMP就可得到无噪输出。 因为，小于8个PCLK的噪声都被过滤掉了。如果噪声增至13个PCLK周长，用户可配置FILTSEL (ACMP_CTL0[23:20], ACMP_CTL1[23:20])为0101，ACMP即得出无噪输出。

6.14.6 比较器参考电压(CRV)

6.14.6.1 介绍

比较器参考电压(CRV)模块用于为比较器提供参考电压。CRV模块由阶梯电阻和模拟开关组成， 用户可以通过配置CRVCTL (ACMP_VREF[3:0])设置CRV输出电压，通过配置IREFSEL (ACMP_VREF[7])选择ACMP参考电压。

6.14.6.2 特性

- 用户可通过设置CRVCTL(ACMP_VREF[3:0)选择参考电压值
- 当设置IREFSEL (ACMP_VREF[7]) = 0（选择带隙电压时），阶梯电阻将被自动关闭，以降低电源功耗。

下图为CRV模块的框图:

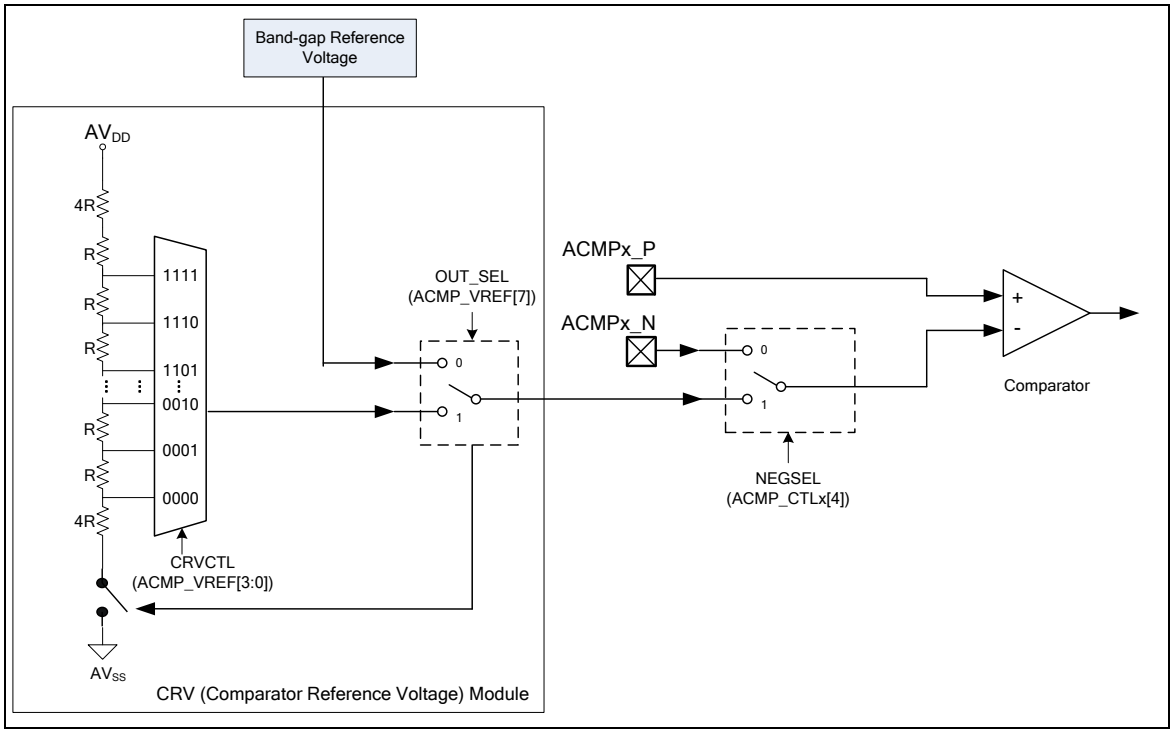


图 6.14-4 比较器参考电压框图

6.14.7 寄存器表

R：只读，W：只写， R/W：可读/写

寄存器	偏移量	R/W	描述	复位值
ACMP基地址: **ACMP_BA = 0x400D_0000**				
ACMP_CTL0	ACMP_BA+0x00	R/W	模拟比较器0控制寄存器	0x0000_0000
ACMP_CTL1	ACMP_BA+0x04	R/W	模拟比较器1控制寄存器	0x0000_0000
ACMP_STATUS	ACMP_BA+0x08	R/W	模拟比较器0/1状态寄存器	0x0000_0000
ACMP_VREF	ACMP_BA+0x0C	R/W	模拟比较器参考电压寄存器	0x0000_0000

6.14.8 寄存器描述

模拟比较器 0 控制寄存器(ACMP_CTL0)

寄存器	偏移量	R/W	描述	复位值
ACMP_CTL0	ACMP_BA+0x00	R/W	模拟比较器0控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**POSSEL**		**Reserved**				
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
FILTSEL				**Reserved**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**FTRGEN**	**RTRGEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**NEGSEL**	**Reserved**	**HYSSEL**	**ACMPIE**	**ACMPEN**

位	描述	
[31]	**Reserved**	Reserved.
[30:29]	**POSSEL**	**比较器ACMP0正极输入端选择** 00 = ACMP0_Px从ACMP0_P0 (P1.5)输入 01 = ACMP0_Px从ACMP0_P1 (P1.0)输入 10 = ACMP0_Px从ACMP0_P2 (P1.2)输入 11 = ACMP0_Px从ACMP0_P3 (P1.3)输入
[28:24]	**Reserved**	Reserved.
[23:20]	**FILTSEL**	**比较器输出滤波计数器选择** 0000 = 滤波器功能关闭. 0001 = ACMP0 输出经过1 个连续 PCLK采样 0010 = ACMP0输出经过2 个连续 PCLK采样. 0011 = ACMP0输出经过4 个连续 PCLK采样. 0100 = ACMP0输出经过8 个连续 PCLK采样. 0101 = ACMP0输出经过16 个连续 PCLK采样. 0110 = ACMP0输出经过32 个连续 PCLK采样. 0111 = ACMP0输出经过64 个连续 PCLK采样. 1000 = ACMP0输出经过128 个连续 PCLK采样. 1001 = ACMP0输出经过256 个连续 PCLK采样. 1010 = ACMP0输出经过512 个连续 PCLK采样. 其他 = 保留
[19:10]	**Reserved**	Reserved.

位	描述	
[9]	**FTRGEN**	**模拟比较器 0 下降沿触发使能** 0 = 使能模拟比较器0 下降沿触发PWM或TIMER 1 = 禁止模拟比较器0 下降沿触发 **注:**该位仅在模拟比较器0触发PWM或TIMER时有效
[8]	**RTRGEN**	**模拟比较器 0 上升沿触发使能** 0 = 使能模拟比较器0 上升沿触发PWM或TIMER 1 = 禁止模拟比较器0 上升沿触发 **注:**该位仅在模拟比较器0触发PWM或TIMER时有效
[7:5]	**Reserved**	Reserved.
[4]	**NEGSEL**	**模拟比较器0 负极输入选择位** 0 = 比较器负极输入来自CPN0管脚 1 =比较器负极输入来自内部带隙电压或比较器参考电压
[3]	**Reserved**	Reserved.
[2]	**HYSSEL**	**模拟比较器0 迟滞功能选择位** 0 = 禁止迟滞功能 1 = 使能迟滞功能
[1]	**ACMPIE**	**模拟比较器0 中断使能位** 0 = 禁止中断功能 1 = 使能中断功能
[0]	**ACMPEN**	**模拟比较器0使能位** 0 = 禁止模拟比较器0 1 = 使能模拟比较器0 **注**: 该位置1后,模拟比较器需要2 us等待时间稳定输出

模拟比较器 1 控制寄存器 (ACMP_CTL1)

寄存器	偏移量	R/W	描述	复位值
ACMP_CTL1	ACMP_BA+0x04	R/W	模拟比较器1 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**POSSEL**		**Reserved**				
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
FILTSEL				**Reserved**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**FTRGEN**	**RTRGEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**NEGSEL**	**Reserved**	**HYSSEL**	**ACMPIE**	**ACMPEN**

位	描述	
[31]	**Reserved**	Reserved.
[30:29]	**POSSEL**	**比较器ACMP1正极输入端选择** 00 = ACMP1_Px从ACMP1_P0 (P3.1)输入 01 = ACMP1_Px从ACMP1_P1 (P3.2)输入 10 = ACMP1_Px从ACMP1_P2 (P3.4)输入 11 = ACMP1_Px从ACMP1_P3 (P3.5)输入
[28:24]	**Reserved**	Reserved.
[23:20]	**FILTSEL**	**比较器输出滤波计数器选择** 0000 = 滤波器功能关闭. 0001 = ACMP1 输出经过1 个连续 PCLK采样 0010 = ACMP1输出经过2 个连续 PCLK采样. 0011 = ACMP1输出经过4 个连续 PCLK采样. 0100 = ACMP1输出经过8 个连续 PCLK采样. 0101 = ACMP1输出经过16 个连续 PCLK采样. 0110 = ACMP1输出经过32 个连续 PCLK采样. 0111 = ACMP1输出经过64 个连续 PCLK采样. 1000 = ACMP1输出经过128 个连续 PCLK采样. 1001 = ACMP1输出经过256 个连续 PCLK采样. 1010 = ACMP1输出经过512 个连续 PCLK采样. 其他 = 保留
[19:10]	**Reserved**	Reserved.
[9]	**FTRGEN**	**模拟比较器 1 下降沿触发使能**

位	描述	
		0 = 使能模拟比较器1 下降沿触发PWM或TIMER 1 = 禁止模拟比较器1 下降沿触发 **注**:该位仅在模拟比较器1 触发PWM或TIMER时有效
[8]	**RTRGEN**	**模拟比较器 1 上升沿触发使能** 0 = 使能模拟比较器1 上升沿触发PWM或TIMER 1 = 禁止模拟比较器1 上升沿触发 **注**:该位仅在模拟比较器1触发PWM或TIMER时有效
[7:5]	**Reserved**	Reserved.
[4]	**NEGSEL**	**模拟比较器1 负极输入选择位** 0 = 比较器负极输入来自CPN1管脚 1 = 比较器负极输入来自内部带隙电压或比较器参考电压
[3]	**Reserved**	Reserved.
[2]	**HYSSEL**	**模拟比较器1 迟滞功能选择位** 0 = 禁止迟滞功能 1 = 使能迟滞功能
[1]	**ACMPIE**	**模拟比较器1 中断使能位** 0 = 禁止中断功能 1 = 使能中断功能
[0]	**ACMPEN**	**模拟比较器1使能位** 0 = 禁止模拟比较器1 1 = 使能模拟比较器1 **注:** 该位置1后,模拟比较器需要2 us等待时间稳定输出

模拟比较器 0/1状态寄存器(ACMP_STATUS)

寄存器	偏移量	R/W	描述	复位值
ACMP_STATUS	ACMP_BA+0x08	R/W	模拟比较器0/1状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**ACMPO1**	**ACMPO0**	**ACMPIF1**	**ACMPIF0**

位	描述	
[31:4]	**Reserved**	Reserved.
[3]	**ACMPO1**	**比较器ACMP1输出** 同步APB时钟并允许软件访问。当ACMPEN (ACMP_CTL1[0]) = 0比较器1被禁用时该位被清除 0 = 模拟比较器1输出0 1 = 模拟比较器1输出1
[2]	**ACMPO0**	**比较器ACMP0输出** 同步APB时钟并允许软件访问。当ACMPEN (ACMP_CTL0[0]) = 0比较器0被禁用时该位被清除 0 = 模拟比较器0输出0 1 = 模拟比较器0输出1
[1]	**ACMPIF1**	**比较器1标志** 每当比较器1输出状态改变时，该位将被硬件置位。如果ACMPIE (ACMP_CTL1[1]) = 1，则产生中断。 0 = 模拟比较器1输出没有发生改变 1 = 模拟比较器1输出发生改变 **注**：写1清除该位为0。
[0]	**ACMPIF0**	**比较器0标志** 每当比较器1输出状态改变时，该位将被硬件置位。如果ACMPIE (ACMP_CTL0[1]) = 1，则产生中断。 0 = 模拟比较器0输出没有发生改变 1 = 模拟比较器0输出发生改变 **注**：写1清除该位为0。

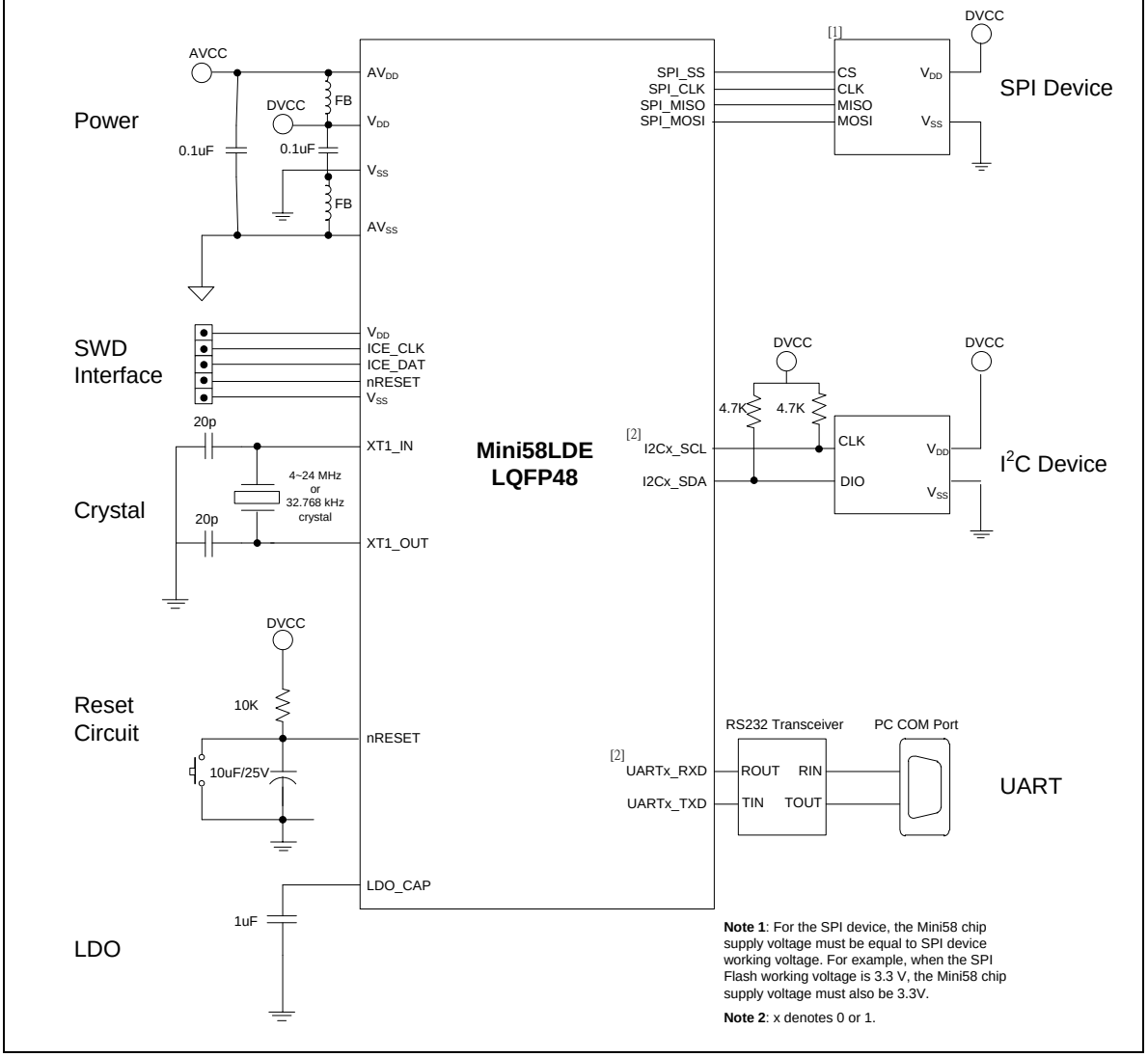
模拟比较器参考电压控制寄存器 (ACMP_VREF)

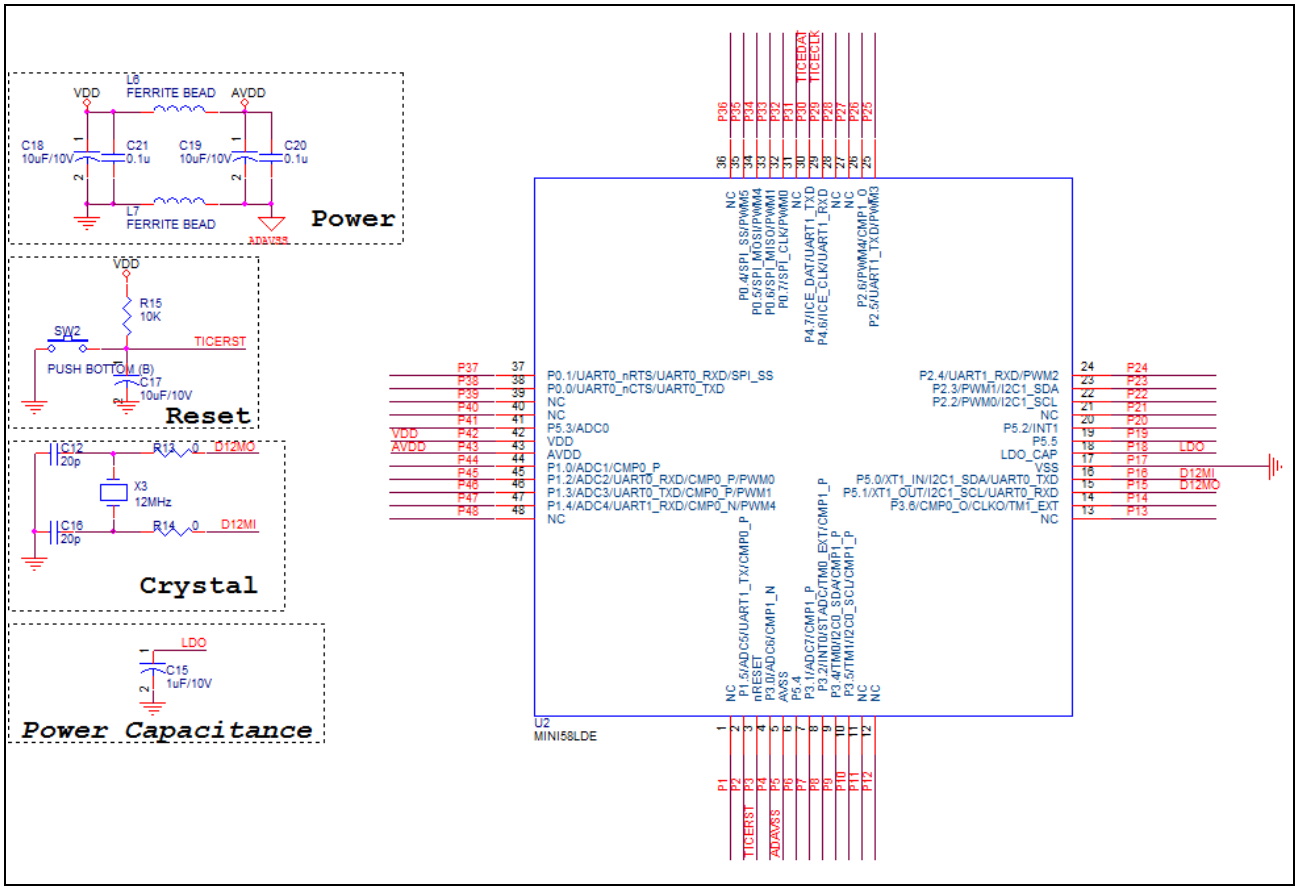
寄存器	偏移量	R/W	描述	复位值
ACMP_VREF	ACMP_BA+0x0C	R/W	模拟比较器参考电压控制寄存器	0x0000_0000

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

位	描述	
[31:8]	**Reserved**	Reserved.
[7]	**IREFSEL**	CRV内部参考电压选择 0 = 带隙电压 1 = 内部比较器参考电压
[6:4]	**Reserved**	Reserved.
[3:0]	**CRVCTL**	比较器参考电压控制 比较器参考电压 = AV_{DD} * (1 / 6 + CRVCTL(ACMP_VREF[3:0]) / 24).

7 应用电路


Power
AVCC
DVCC
FB
0.1uF
0.1uF
FB
AVDD
VDD
VSS
AVSS
SWD Interface
VDD
ICE_CLK
ICE_DAT
nRESET
VSS
Crystal
20p
20p
4~24 MHz or 32.768 kHz crystal
XT1_IN
XT1_OUT
Reset Circuit
DVCC
10K
10uF/25V
nRESET
LDO
1uF
LDO_CAP
Mini58LDE
LQFP48
SPI_SS
SPI_CLK
SPI_MISO
SPI_MOSI
[1]
CS
CLK
MISO
MOSI
VDD
VSS
DVCC
SPI Device
DVCC
DVCC
4.7K
4.7K
[2]
I2Cx_SCL
I2Cx_SDA
CLK
DIO
VDD
VSS
I2C Device
RS232 Transceiver
PC COM Port
[2]
UARTx_RXD
UARTx_TXD
ROUT
RIN
TIN
TOUT
UART
Note 1: For the SPI device, the Mini58 chip supply voltage must be equal to SPI device working voltage. For example, when the SPI Flash working voltage is 3.3 V, the Mini58 chip supply voltage must also be 3.3V.
Note 2: x denotes 0 or 1.

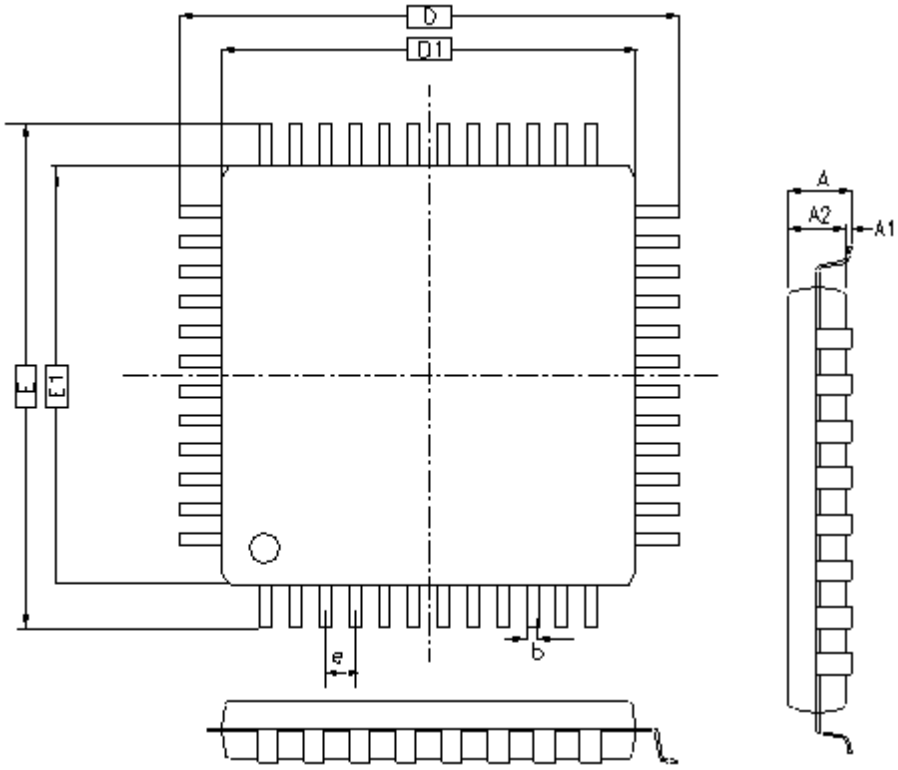

Power
VDD
L6
FERRITE BEAD
AVDD
C18
10uF/10V
C21
0.1u
C19
10uF/10V
C20
0.1u
L7
FERRITE BEAD
ADAVSS
Reset
VDD
R15
10K
SW2
TICERST
PUSH BOTTOM (B)
C17
10uF/10V
Crystal
C12
20p
R13
0
D12MO
X3
12MHz
C16
20p
R14
0
D12MI
Power Capacitance
LDO
C15
1uF/10V
U2
MINI58LDE
P0.1/UART0_nRTS/UART0_RXD/SPI_SS
P0.0/UART0_nCTS/UART0_TXD
NC
NC
P5.3/ADC0
VDD
AVDD
P1.0/ADC1/CMP0_P
P1.2/ADC2/UART0_RXD/CMP0_P/PWM0
P1.3/ADC3/UART0_TXD/CMP0_P/PWM1
P1.4/ADC4/UART1_RXD/CMP0_N/PWM4
NC
P2.4/UART1_RXD/PWM2
P2.3/PWM1/I2C1_SDA
P2.2/PWM0/I2C1_SCL
NC
P5.2/INT1
P5.5
LDO_CAP
VSS
P5.0/XT1_IN/I2C1_SDA/UART0_TXD
P5.1/XT1_OUT/I2C1_SCL/UART0_RXD
P3.6/CMP0_O/CLKO/TM1_EXT
NC
P0.4/SPI_SS/PWM5
P0.5/SPI_MOSI/PWM4
P0.6/SPI_MISO/PWM1
P0.7/SPI_CLK/PWM0
P4.7/ICE_DAT/UART1_TXD
P4.6/ICE_CLK/UART1_RXD
P2.6/PWM4/CMP1_O
P2.5/UART1_TXD/PWM3
P1.5/ADC5/UART1_TX/CMP0_P
nRESET
P3.0/ADC6/CMP1_N
AVSS
P5.4
P3.1/ADC7/CMP1_P
P3.2/INT0/STADC/TM0_EXT/CMP1_P
P3.4/TM0/I2C0_SDA/CMP1_P
P3.5/TM1/I2C0_SCL/CMP1_P
TICEDAT
TICECLK
TICERST
ADAVSS
D12MI
D12MO

8 电气特性

Mini58系列的电气特性请参考NuMicro® Mini58系列数据手册

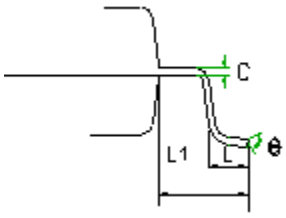
9 封装尺寸

9.1 48-pin LQFP

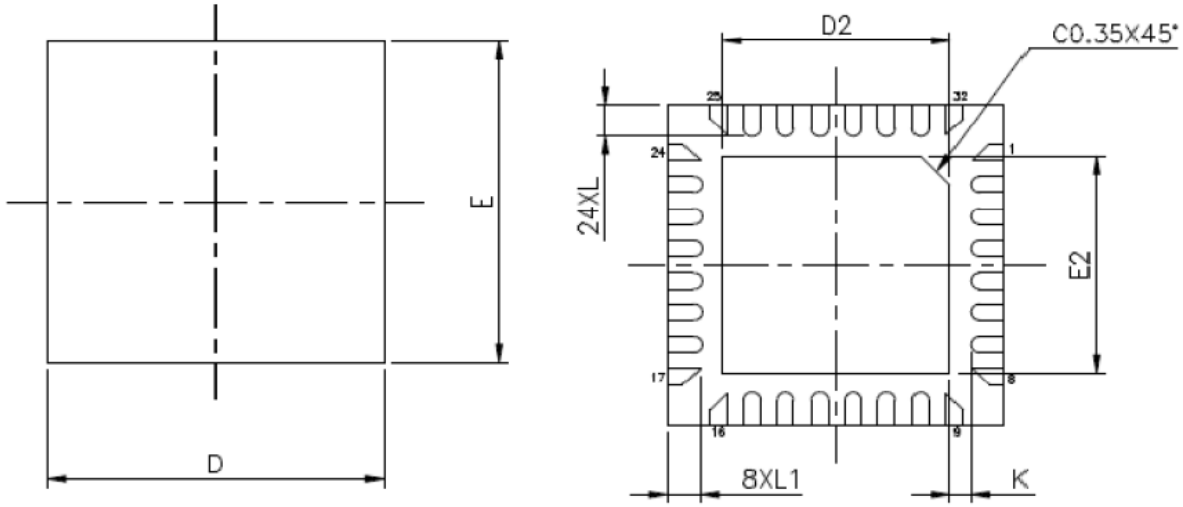


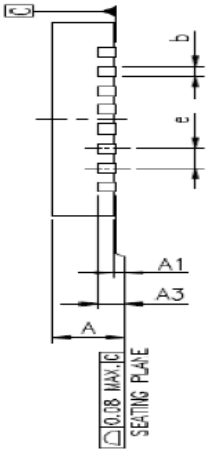
CONTROL DIMENSIONS ARE IN MILLIMETERS

SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	—	—	1.60	—	—	0.063
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
D1	6.90	7.00	7.10	0.272	0.276	0.280
E1	6.90	7.00	7.10	0.272	0.276	0.280
e	0.35	0.50	0.65	0.014	0.020	0.026
D	8.9	9.00	9.10	0.350	0.354	0.358
E	8.9	9.00	9.10	0.350	0.354	0.358
L	0.45	0.60	0.75	0.018	0.024	0.030
L1	—	1.00	—	—	0.039	—
C	0.09	—	0.20	0.004	—	0.0079
θ	0°	—	7°	0°	—	7°
b	0.17	0.22	0.27	0.007	0.0087	0.011



9.2 33-pin QFN (4 mm x 4 mm)

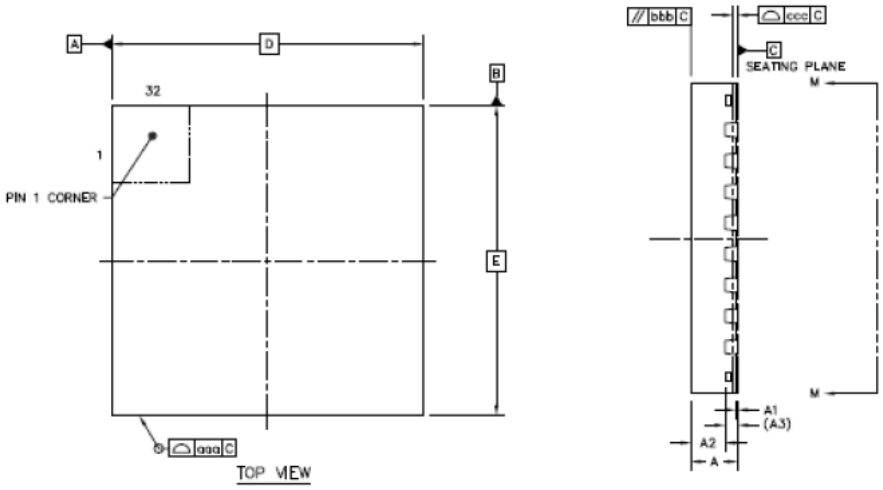


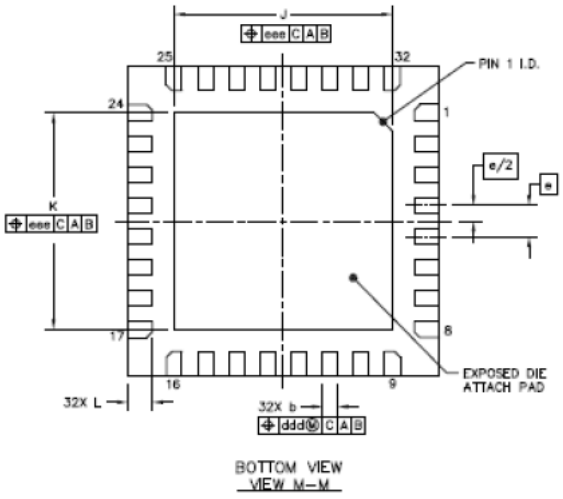


	Symbol	Min	Nom	Max
Total Thickness	A	0.7	0.75	0.8
Stand Off	A1	0.0	0.02	0.05
L/F Thickness	A3	0.2 REF.		
Lead Width	b	0.15	0.20	0.25
Body Size	D	4.00 BSC		
	E	4.00 BSC		
Lead Pitch	e	0.40 BSC		
Lead Length	L	0.35	0.4	0.45
	K	0.2	--	--

Symbol	L/F SIZE	Dimension in mm		
		MIN.	NOR.	MAX.
D2	114X114 mm	2.6	2.7	2.75
E2		2.6	2.7	2.75

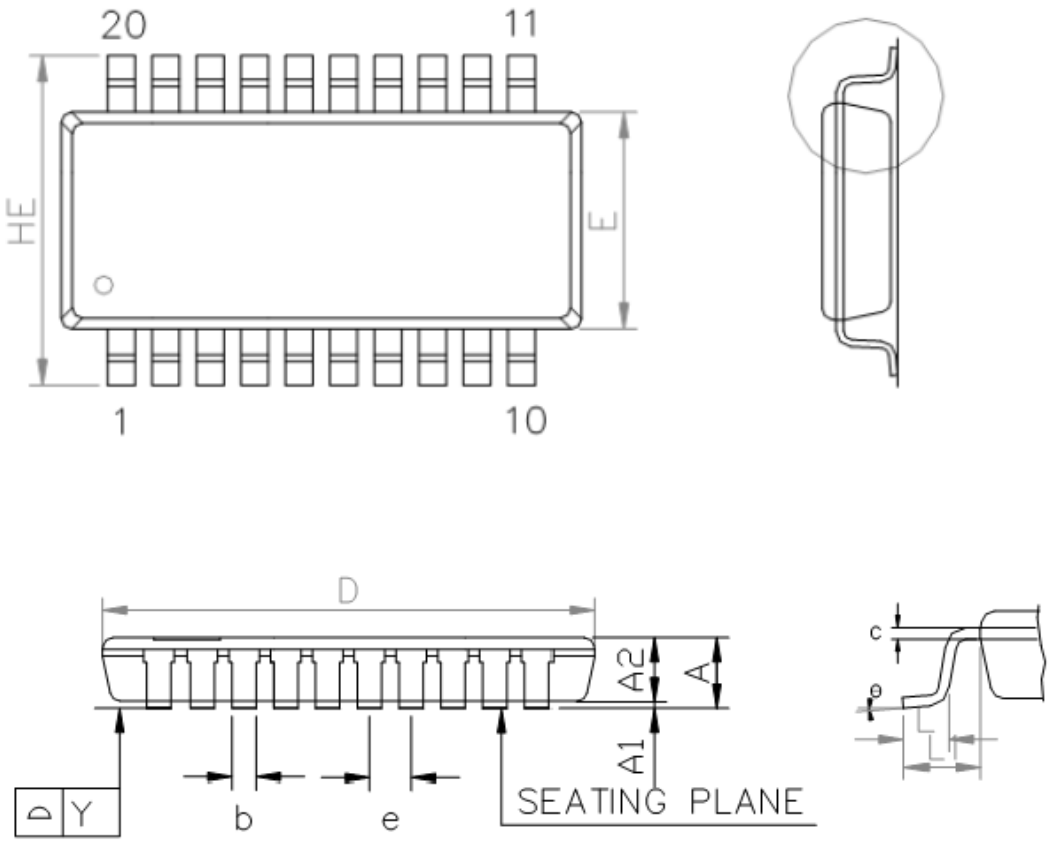
9.3 33-pin QFN (5 mm x 5 mm)





		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.7	0.75	0.8
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.55	0.57
L/F THICKNESS		A3	0.203 REF		
LEAD WIDTH		b	0.2	0.25	0.3
BODY SIZE	X	D	5 BSC		
	Y	E	5 BSC		
LEAD PITCH		e	0.5 BSC		
EP SIZE	X	J	3.4	3.5	3.6
	Y	K	3.4	3.5	3.6
LEAD LENGTH		L	0.35	0.4	0.45
PACKAGE EDGE TOLERANCE		aaa	0.1		
MOLD FLATNESS		bbb	0.1		
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.1		
EXPOSED PAD OFFSET		eee	0.1		

9.4 20-pin TSSOP



SYMBOL	DIMENSION (MM)			DIMENSION (INCH)		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	-	-	1.20	-	-	0.047
A1	0.05	-	0.15	0.002	-	0.006
A2	0.80	0.90	1.05	0.031	0.035	0.041
E	4.30	4.40	4.50	0.169	0.173	0.177
HE	6.40 BSC			0.252 BSC		
D	6.40	6.50	6.60	0.252	0.256	0.260
L	0.50	0.60	0.75	0.020	0.024	0.030
L1	1.00 REF			0.039 REF		
b	0.19	-	0.30	0.007	-	0.012
e	0.65 BSC			0.026 BSC		
c	0.09	-	0.20	0.004	-	0.008
θ	0°	-	8°	0°	-	8°
Y	0.10 BASIC			0.004 BASIC		

10 修订历史

日期	版本	描述
2015.06.11	1.00	第一版.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.